

**DẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**



**Kho dữ liệu và Hệ hỗ trợ quyết định**

---

**Kho dữ liệu & Hệ hỗ trợ quyết định  
cho bài toán Bank Customer Churn**

---

GVHD: Bùi Tiến Đức  
SV thực hiện: Hoa Toàn Hạc – 2201917  
Mai Huy Hiệp – 2211045

TP. HỒ CHÍ MINH, THÁNG 11, 2025



## Mục lục

<b>1</b>	<b>Video thuyết trình</b>	<b>5</b>
<b>2</b>	<b>Giới thiệu</b>	<b>5</b>
2.1	Đặt vấn đề . . . . .	5
2.2	Mục tiêu đề tài . . . . .	5
2.3	Phạm vi đề tài . . . . .	6
2.4	Tổng quan về dữ liệu . . . . .	6
2.4.1	Nguồn dữ liệu . . . . .	6
2.4.2	Mô tả các biến . . . . .	6
2.4.3	Đặc điểm của dữ liệu . . . . .	7
2.5	Cấu trúc báo cáo . . . . .	7
<b>3</b>	<b>Thiết kế kho dữ liệu</b>	<b>8</b>
3.1	Mô hình ngôi sao (Star Schema) . . . . .	8
3.2	Sơ đồ tổng quan . . . . .	8
3.3	Bảng sự kiện: fact_customer_status . . . . .	8
3.3.1	Cấu trúc bảng . . . . .	8
3.3.2	Grain của bảng sự kiện . . . . .	8
3.4	Bảng chiều: dim_customer . . . . .	9
3.5	Bảng chiều: dim_geo . . . . .	9
3.6	Bảng chiều: dim_time . . . . .	9
3.7	Bảng chiều: dim_segment . . . . .	9
3.8	SQL DDL Schema . . . . .	10
3.9	Lợi ích của thiết kế . . . . .	10
<b>4</b>	<b>Quy trình ETL và tiền xử lý dữ liệu</b>	<b>11</b>
4.1	Tổng quan về quy trình ETL . . . . .	11
4.2	Giai đoạn 1: Extract - Trích xuất dữ liệu . . . . .	11
4.2.1	Nguồn dữ liệu . . . . .	11
4.2.2	Công cụ sử dụng . . . . .	11
4.2.3	Kiểm tra chất lượng dữ liệu . . . . .	11
4.3	Giai đoạn 2: Transform - Biến đổi dữ liệu . . . . .	11
4.3.1	Bước 1: Làm sạch dữ liệu . . . . .	12
4.3.2	Bước 2: Feature Engineering . . . . .	12
4.3.3	Bước 3: Xây dựng bảng chiều . . . . .	12
4.3.4	Bước 4: Xây dựng bảng sự kiện . . . . .	13
4.4	Giai đoạn 3: Load - Nạp dữ liệu . . . . .	14
4.4.1	Xuất ra file CSV . . . . .	14
4.4.2	Nạp vào cơ sở dữ liệu (tùy chọn) . . . . .	15
4.5	Tổng kết quy trình ETL . . . . .	15
4.5.1	Kết quả đầu ra . . . . .	15
4.5.2	Ưu điểm của quy trình . . . . .	15
4.5.3	Thách thức và giải pháp . . . . .	16



<b>5 Phân tích OLAP và trực quan hóa bằng Python</b>	<b>16</b>
5.1 Phân tích OLAP . . . . .	16
5.1.1 Các truy vấn OLAP tiêu biểu . . . . .	16
5.2 Trực quan hóa dữ liệu bằng Python . . . . .	17
5.2.1 Exploratory Data Analysis (EDA) . . . . .	17
5.2.2 Dashboard-style Visualizations . . . . .	18
5.3 Insights từ phân tích . . . . .	21
5.4 Ứng dụng trong ra quyết định . . . . .	22
<b>6 Mô hình dự đoán churn (Machine Learning)</b>	<b>22</b>
6.1 Tổng quan về bài toán . . . . .	22
6.2 Chuẩn bị dữ liệu cho modeling . . . . .	23
6.2.1 Chọn features . . . . .	23
6.2.2 Chia tập train/test . . . . .	23
6.3 Preprocessing Pipeline . . . . .	24
6.4 Xây dựng mô hình . . . . .	24
6.4.1 Mô hình 1: Logistic Regression . . . . .	24
6.4.2 Mô hình 2: Random Forest . . . . .	25
6.5 Dánh giá mô hình . . . . .	25
6.5.1 Accuracy . . . . .	25
6.5.2 Confusion Matrix . . . . .	25
6.5.3 Classification Report . . . . .	26
6.5.4 ROC-AUC Score . . . . .	27
6.6 Feature Importance . . . . .	27
<b>7 Hệ hỗ trợ quyết định (DSS)</b>	<b>29</b>
7.1 Khái niệm Decision Support System . . . . .	29
7.2 Kiến trúc DSS cho bài toán Churn . . . . .	29
7.2.1 Data Warehouse (Kho dữ liệu) . . . . .	29
7.2.2 OLAP Engine (Công cụ phân tích đa chiều) . . . . .	29
7.2.3 Predictive Model (Mô hình dự đoán) . . . . .	29
7.2.4 Visualization Layer (Lớp trực quan hóa) . . . . .	29
7.2.5 User Interface (Giao diện người dùng) . . . . .	30
7.3 Dashboard cho kho dữ liệu . . . . .	30
7.3.1 Tại sao cần Dashboard? . . . . .	30
7.3.2 Công nghệ sử dụng . . . . .	30
7.3.3 Kiến trúc Dashboard . . . . .	30
7.3.4 Các thành phần Dashboard . . . . .	31
7.3.5 Chạy Dashboard . . . . .	32
7.3.6 Giao diện Dashboard . . . . .	32
7.3.7 Ưu điểm của Dashboard tương tác . . . . .	32
7.3.8 Ví dụ thực tế: Phân tích churn theo Age Group . . . . .	33
7.3.9 Ví dụ thực tế: Phân tích churn ở Germany . . . . .	33
7.3.10 Kết luận về Dashboard . . . . .	34



---

<b>8</b>	<b>Kết luận</b>	<b>35</b>
8.1	Tổng kết . . . . .	35
8.2	Kết quả đạt được . . . . .	35
8.2.1	Về mặt kỹ thuật . . . . .	35
8.2.2	Về mặt nghiệp vụ . . . . .	36
8.3	Hạn chế của đề tài . . . . .	36
8.4	Hướng phát triển . . . . .	37
8.5	Kết luận cuối cùng . . . . .	38



## Lời mở đầu

Trong bối cảnh ngành ngân hàng ngày càng cạnh tranh, việc giữ chân khách hàng hiện tại trở nên quan trọng hơn bao giờ hết. Hiện tượng khách hàng rời bỏ dịch vụ (customer churn) không chỉ gây tổn thất về doanh thu mà còn ảnh hưởng đến uy tín và vị thế của ngân hàng trên thị trường. Để giải quyết vấn đề này, các ngân hàng cần có khả năng phân tích dữ liệu khách hàng một cách toàn diện, từ đó đưa ra các quyết định chiến lược kịp thời và hiệu quả.

Đề tài "**Kho dữ liệu & Hệ hỗ trợ quyết định cho bài toán Bank Customer Churn**" được thực hiện nhằm xây dựng một hệ thống hoàn chỉnh giúp phân tích và dự đoán khả năng khách hàng rời bỏ ngân hàng. Hệ thống bao gồm ba thành phần chính:

- **Kho dữ liệu (Data Warehouse):** Thiết kế theo mô hình ngôi sao (star schema) với các bảng chiều (dimension) và bảng sự kiện (fact), tổ chức dữ liệu theo cách tối ưu cho phân tích OLAP.
- **Quy trình ETL:** Trích xuất dữ liệu từ nguồn gốc (Kaggle), làm sạch, biến đổi và nạp vào kho dữ liệu, bao gồm cả feature engineering để tạo ra các đặc trưng mới phục vụ phân tích.
- **Hệ hỗ trợ quyết định (DSS):** Kết hợp phân tích OLAP, trực quan hóa dữ liệu bằng Python, và mô hình học máy để dự đoán churn, từ đó cung cấp thông tin hỗ trợ ra quyết định cho ban quản lý.

Dự án sử dụng bộ dữ liệu *Bank Customer Churn Modeling* từ Kaggle, bao gồm thông tin về 10,000 khách hàng với các thuộc tính như điểm tín dụng, quốc gia, giới tính, độ tuổi, số dư tài khoản, và trạng thái churn. Qua quá trình phân tích và xây dựng mô hình, chúng tôi kỳ vọng không chỉ đạt được độ chính xác cao trong dự đoán mà còn rút ra được những insight quan trọng về các yếu tố ảnh hưởng đến quyết định rời bỏ của khách hàng.

Báo cáo này trình bày chi tiết về thiết kế kho dữ liệu, quy trình ETL, các phân tích OLAP, trực quan hóa, mô hình dự đoán, và cách thức hệ thống hỗ trợ ra quyết định trong thực tế. Đây là một bài tập lớn thuộc môn học *Kho dữ liệu và Hệ hỗ trợ quyết định*, nhằm áp dụng các kiến thức lý thuyết vào giải quyết một bài toán thực tiễn trong lĩnh vực ngân hàng.



## 1 Video thuyết trình

### Video thuyết trình

<https://drive.google.com/file/d/14mohveAi3hX84PLTyWikJx-kEXoFuSaV/view?usp=sharing>

### Github

[https://github.com/toanhac/hcmut\\_datawarehouse\\_251](https://github.com/toanhac/hcmut_datawarehouse_251)

## 2 Giới thiệu

### 2.1 Đặt vấn đề

Trong ngành ngân hàng, việc giữ chân khách hàng hiện tại thường có chi phí thấp hơn nhiều so với việc thu hút khách hàng mới. Theo nghiên cứu, chi phí để có được một khách hàng mới có thể cao gấp 5-25 lần so với việc duy trì một khách hàng hiện tại. Do đó, việc dự đoán và ngăn chặn hiện tượng khách hàng rời bỏ (customer churn) trở thành một ưu tiên hàng đầu đối với các ngân hàng.

Tuy nhiên, để có thể phân tích và dự đoán churn một cách hiệu quả, ngân hàng cần:

- Tổ chức dữ liệu khách hàng theo cách tối ưu cho phân tích (Data Warehouse).
- Có quy trình xử lý dữ liệu tự động và đáng tin cậy (ETL Pipeline).
- Phân tích dữ liệu theo nhiều chiều khác nhau (OLAP).
- Xây dựng mô hình dự đoán chính xác (Machine Learning).
- Cung cấp thông tin trực quan và dễ hiểu cho ban quản lý (Decision Support System).

Đề tài này được thực hiện nhằm xây dựng một hệ thống hoàn chỉnh đáp ứng các yêu cầu trên, áp dụng cho bài toán dự đoán churn của khách hàng ngân hàng.

### 2.2 Mục tiêu đề tài

Mục tiêu chính của đề tài bao gồm:

1. **Thiết kế và triển khai Data Warehouse:** Xây dựng kho dữ liệu theo mô hình ngôi sao (star schema) với các bảng chiều và bảng sự kiện, tối ưu hóa cho phân tích OLAP.
2. **Xây dựng quy trình ETL:** Phát triển pipeline tự động để trích xuất dữ liệu từ nguồn, làm sạch, biến đổi (bao gồm feature engineering), và nạp vào kho dữ liệu.
3. **Phân tích OLAP:** Thực hiện các truy vấn phân tích đa chiều để khám phá các mẫu và xu hướng trong dữ liệu khách hàng.
4. **Trực quan hóa dữ liệu:** Sử dụng Python (matplotlib) để tạo ra các biểu đồ trực quan, giúp hiểu rõ hơn về đặc điểm của khách hàng churn và không churn.
5. **Xây dựng mô hình dự đoán:** Phát triển mô hình Machine Learning để dự đoán khả năng churn của khách hàng với độ chính xác cao.
6. **Hệ hỗ trợ quyết định:** Tích hợp các thành phần trên thành một hệ thống DSS hoàn chỉnh, cung cấp thông tin hỗ trợ cho các quyết định kinh doanh.



## 2.3 Phạm vi đề tài

Đề tài tập trung vào:

- Dữ liệu:** Sử dụng bộ dữ liệu "Bank Customer Churn Modeling" từ Kaggle, bao gồm thông tin về 10,000 khách hàng.
- Công nghệ:** Python (pandas, numpy, matplotlib, scikit-learn), SQL (PostgreSQL hoặc tương đương).
- Mô hình DWH:** Star schema với 4 bảng chiều và 1 bảng sự kiện.
- Mô hình ML:** Logistic Regression và Random Forest cho bài toán phân loại nhị phân.
- Phân tích:** EDA, OLAP queries, visualization, model evaluation.

## 2.4 Tổng quan về dữ liệu

### 2.4.1 Nguồn dữ liệu

Bộ dữ liệu được sử dụng trong đề tài là *Bank Customer Churn Modeling*, có sẵn trên Kaggle. Đây là một bộ dữ liệu mô phỏng thông tin của khách hàng ngân hàng, bao gồm các thuộc tính nhân khẩu học, thông tin tài khoản, và trạng thái churn.

Thuộc tính	Thông tin
Nguồn dữ liệu	Kaggle - Bank Customer Churn Modeling
Tên file	Churn_Modelling.csv
Số lượng quan sát	10,000
Số lượng biến	14
Tỷ lệ churn	20%

### 2.4.2 Mô tả các biến

Bảng dưới đây mô tả chi tiết các biến trong bộ dữ liệu:

Tên biến	Kiểu dữ liệu	Mô tả
RowNumber	Integer	Số thứ tự của bản ghi (không sử dụng trong phân tích).
CustomerId	Integer	ID duy nhất của khách hàng.
Surname	String	Họ của khách hàng (không sử dụng trong phân tích).
CreditScore	Integer	Điểm tín dụng của khách hàng (300-850).
Geography	String	Quốc gia của khách hàng (France, Germany, Spain).
Gender	String	Giới tính của khách hàng (Male, Female).
Age	Integer	Tuổi của khách hàng.
Tenure	Integer	Số năm khách hàng đã sử dụng dịch vụ ngân hàng (0-10).
Balance	Float	Số dư tài khoản của khách hàng.



NumOfProducts	Integer	Số lượng sản phẩm ngân hàng mà khách hàng đang sử dụng (1-4).
HasCrCard	Integer	Khách hàng có thẻ tín dụng hay không (0=Không, 1=Có).
IsActiveMember	Integer	Khách hàng có hoạt động tích cực hay không (0=Không, 1=Có).
EstimatedSalary	Float	Mức lương ước tính của khách hàng.
Exited	Integer	Biến mục tiêu: Khách hàng đã rời bỏ ngân hàng hay chưa (0=Không, 1=Có).

#### 2.4.3 Đặc điểm của dữ liệu

Một số đặc điểm quan trọng của bộ dữ liệu:

- Dữ liệu sạch:** Không có giá trị thiếu (missing values), giúp đơn giản hóa quá trình tiền xử lý.
- Mất cân bằng lớp:** Tỷ lệ khách hàng churn (20%) thấp hơn nhiều so với khách hàng không churn (80%), cần lưu ý khi đánh giá mô hình.
- Đa dạng về thuộc tính:** Bao gồm cả biến số (CreditScore, Age, Balance) và biến phân loại (Geography, Gender).
- Phạm vi giá trị hợp lý:** Các biến đều nằm trong phạm vi thực tế, không có outlier quá cực đoan.

### 2.5 Cấu trúc báo cáo

Báo cáo được tổ chức theo các phần sau:

- Phần 1 - Giới thiệu:** Đặt vấn đề, mục tiêu, phạm vi, và tổng quan về dữ liệu.
- Phần 2 - Thiết kế kho dữ liệu:** Mô tả chi tiết về star schema, các bảng chiều và bảng sự kiện.
- Phần 3 - Quy trình ETL:** Trình bày các bước trích xuất, làm sạch, biến đổi, và nạp dữ liệu.
- Phần 4 - Phân tích OLAP và trực quan hóa:** Các truy vấn phân tích và biểu đồ trực quan.
- Phần 5 - Mô hình dự đoán churn:** Xây dựng và đánh giá mô hình Machine Learning.
- Phần 6 - Hệ hỗ trợ quyết định:** Tích hợp các thành phần thành DSS hoàn chỉnh.
- Phần 7 - Kết luận:** Tổng kết, hạn chế, và hướng phát triển.



### 3 Thiết kế kho dữ liệu

#### 3.1 Mô hình ngôi sao (Star Schema)

Kho dữ liệu được thiết kế theo mô hình ngôi sao (star schema), một trong những mô hình phổ biến nhất trong thiết kế Data Warehouse. Mô hình này bao gồm một bảng sự kiện (fact table) ở trung tâm, được kết nối với nhiều bảng chiều (dimension tables) xung quanh.

**Ưu điểm của Star Schema:**

- Đơn giản, dễ hiểu và dễ truy vấn.
- Hiệu suất cao cho các truy vấn OLAP.
- Tối ưu hóa cho các công cụ Business Intelligence.
- Dễ dàng mở rộng khi cần thêm chiều phân tích mới.

#### 3.2 Sơ đồ tổng quan

Sơ đồ star schema cho bài toán Bank Customer Churn bao gồm:

- **1 bảng sự kiện:** fact\_customer\_status
- **4 bảng chiều:** dim\_customer, dim\_geo, dim\_time, dim\_segment

#### 3.3 Bảng sự kiện: fact\_customer\_status

Bảng sự kiện lưu trữ các chỉ số (measures) và khóa ngoại tham chiếu đến các bảng chiều.

##### 3.3.1 Cấu trúc bảng

Tên cột	Kiểu dữ liệu	Mô tả
fact_key	INTEGER (PK)	Khóa chính của bảng sự kiện
customer_key	INTEGER (FK)	Khóa ngoại tham chiếu đến dim_customer
time_key	INTEGER (FK)	Khóa ngoại tham chiếu đến dim_time
geo_key	INTEGER (FK)	Khóa ngoại tham chiếu đến dim_geo
segment_key	INTEGER (FK)	Khóa ngoại tham chiếu đến dim_segment
balance	DECIMAL(15,2)	Số dư tài khoản của khách hàng
estimated_salary	DECIMAL(15,2)	Mức lương ước tính
num_of_products	INTEGER	Số lượng sản phẩm ngân hàng
credit_score	INTEGER	Điểm tín dụng
has_credit_card	INTEGER	Có thẻ tín dụng (0/1)
is_active_member	INTEGER	Thành viên hoạt động (0/1)
churn_flag	INTEGER	Trạng thái churn (0=Không, 1=Có)

##### 3.3.2 Grain của bảng sự kiện

Grain (độ chi tiết) của bảng sự kiện là: Một bản ghi cho mỗi khách hàng tại một thời điểm snapshot.



### 3.4 Bảng chiềut: dim \_customer

Bảng chiềut khách hàng lưu trữ thông tin nhân khẩu học và đặc điểm của khách hàng.

Tên cột	Kiểu dữ liệu	Mô tả
customer_key	INTEGER (PK)	Khóa chính (surrogate key)
customer_id	INTEGER	ID gốc của khách hàng từ nguồn
age	INTEGER	Tuổi của khách hàng
gender	VARCHAR(10)	Giới tính (Male/Female)
tenure	INTEGER	Số năm sử dụng dịch vụ (0-10)

### 3.5 Bảng chiềut: dim \_geo

Bảng chiềut địa lý lưu trữ thông tin về quốc gia của khách hàng.

Tên cột	Kiểu dữ liệu	Mô tả
geo_key	INTEGER (PK)	Khóa chính
country	VARCHAR(50)	Tên quốc gia (France, Germany, Spain)

### 3.6 Bảng chiềut: dim \_time

Bảng chiềut thời gian lưu trữ thông tin về thời điểm snapshot dữ liệu.

Tên cột	Kiểu dữ liệu	Mô tả
time_key	INTEGER (PK)	Khóa chính
snapshot_date	DATE	Ngày snapshot
year	INTEGER	Năm
month	INTEGER	Tháng (1-12)
quarter	INTEGER	Quý (1-4)

### 3.7 Bảng chiềut: dim \_segment

Bảng chiềut phân khúc lưu trữ thông tin về nhóm khách hàng dựa trên tuổi và thu nhập.

Tên cột	Kiểu dữ liệu	Mô tả
segment_key	INTEGER (PK)	Khóa chính
age_group	VARCHAR(20)	Nhóm tuổi (Young, Middle-aged, Senior)
income_group	VARCHAR(20)	Nhóm thu nhập (Low, Medium, High)

Quy tắc phân nhóm:

- Age Group:

- Young: 18-35 tuổi
- Middle-aged: 36-55 tuổi
- Senior: 56+ tuổi

- Income Group:

- Low: EstimatedSalary < 50,000
- Medium: 50,000 <= EstimatedSalary < 100,000
- High: EstimatedSalary >= 100,000



### 3.8 SQL DDL Schema

Dưới đây là một phần của SQL DDL để tạo các bảng trong kho dữ liệu:

Listing 1: Tạo bảng dim\_customer

```
1 CREATE TABLE dim_customer (
2     customer_key SERIAL PRIMARY KEY,
3     customer_id INTEGER NOT NULL,
4     age INTEGER,
5     gender VARCHAR(10),
6     tenure INTEGER
7 );
```

Listing 2: Tạo bảng fact\_customer\_status

```
1 CREATE TABLE fact_customer_status (
2     fact_key SERIAL PRIMARY KEY,
3     customer_key INTEGER REFERENCES dim_customer(customer_key),
4     time_key INTEGER REFERENCES dim_time(time_key),
5     geo_key INTEGER REFERENCES dim_geo(geo_key),
6     segment_key INTEGER REFERENCES dim_segment(segment_key),
7     balance DECIMAL(15,2),
8     estimated_salary DECIMAL(15,2),
9     num_of_products INTEGER,
10    credit_score INTEGER,
11    has_credit_card INTEGER,
12    is_active_member INTEGER,
13    churn_flag INTEGER
14 );
```

Lưu ý: File SQL DDL đầy đủ có sẵn tại [sql/create\\_dwh\\_schema.sql](#) trong project.

### 3.9 Lợi ích của thiết kế

Thiết kế star schema này mang lại nhiều lợi ích:

- Hiệu suất truy vấn cao:** Các truy vấn OLAP chỉ cần join từ bảng fact đến các bảng dimension, rất nhanh.
- Dễ dàng phân tích đa chiều:** Có thể phân tích churn theo nhiều chiều: thời gian, địa lý, phân khúc khách hàng.
- Tách biệt dữ liệu phân tích và dữ liệu giao dịch:** Không ảnh hưởng đến hệ thống OLTP.
- Hỗ trợ feature engineering:** Bảng dim\_segment chứa các đặc trưng đã được xử lý (age\_group, income\_group), giúp phân tích và modeling dễ dàng hơn.
- Khả năng mở rộng:** Dễ dàng thêm các chiều mới (ví dụ: dim\_product, dim\_channel) khi cần.



## 4 Quy trình ETL và tiền xử lý dữ liệu

### 4.1 Tổng quan về quy trình ETL

ETL (Extract, Transform, Load) là quy trình quan trọng để đưa dữ liệu từ nguồn vào kho dữ liệu. Quy trình ETL trong dự án này bao gồm ba giai đoạn chính:

1. **Extract (Trích xuất):** Đọc dữ liệu từ file CSV gốc.
2. **Transform (Biến đổi):** Làm sạch dữ liệu, feature engineering, tạo các bảng chiều và bảng sự kiện.
3. **Load (Nạp):** Xuất các bảng đã xử lý ra file CSV hoặc nạp vào cơ sở dữ liệu.

### 4.2 Giai đoạn 1: Extract - Trích xuất dữ liệu

#### 4.2.1 Nguồn dữ liệu

Dữ liệu được trích xuất từ file `Churn_Modelling.csv` (tải từ Kaggle), chứa 10,000 bản ghi khách hàng.

#### 4.2.2 Công cụ sử dụng

Sử dụng thư viện `pandas` của Python để đọc file CSV:

Listing 3: Đọc dữ liệu gốc

```
1 import pandas as pd
2
3 # Doc file CSV
4 df = pd.read_csv('data/raw/Churn_Modelling.csv')
5
6 # Kiem tra kich thuoc
7 print(f"So luong ban ghi: {len(df)}")
8 print(f"So luong cot: {len(df.columns)}")
```

**Output:**

```
So luong ban ghi: 10000
So luong cot: 14
```

#### 4.2.3 Kiểm tra chất lượng dữ liệu

Sau khi đọc, cần kiểm tra:

- Giá trị thiếu (missing values): Bộ dữ liệu này không có giá trị thiếu.
- Kiểu dữ liệu: Đảm bảo các cột có kiểu dữ liệu phù hợp.
- Giá trị trùng lặp: Kiểm tra CustomerId có bị trùng không.

### 4.3 Giai đoạn 2: Transform - Biến đổi dữ liệu

Đây là giai đoạn quan trọng nhất, bao gồm nhiều bước xử lý.



#### 4.3.1 Bước 1: Làm sạch dữ liệu

**Loại bỏ các cột không cần thiết:**

Các cột RowNumber, CustomerId, và Surname không mang thông tin phân tích, được loại bỏ:

```
1 # Loai bo cac cot khong can thiet
2 df_clean = df.drop(['RowNumber', 'Surname'], axis=1)
```

Lưu ý: CustomerId được giữ lại để làm khóa tự nhiên trong dim\_customer.

#### 4.3.2 Bước 2: Feature Engineering

Tạo các đặc trưng mới để phục vụ phân tích:

a) Age Group (Nhóm tuổi):

```
1 def categorize_age(age):
2     if age < 36:
3         return 'Young'
4     elif age < 56:
5         return 'Middle-aged'
6     else:
7         return 'Senior'
8
9 df_clean['age_group'] = df_clean['Age'].apply(categorize_age)
```

b) Income Group (Nhóm thu nhập):

```
1 def categorize_income(salary):
2     if salary < 50000:
3         return 'Low'
4     elif salary < 100000:
5         return 'Medium'
6     else:
7         return 'High'
8
9 df_clean['income_group'] = df_clean['EstimatedSalary'].apply(
    categorize_income)
```

c) Snapshot Date:

Vì dữ liệu là snapshot tại một thời điểm, ta tạo cột thời gian:

```
1 from datetime import datetime
2
3 # Gia su snapshot vao ngay 2025-01-01
4 df_clean['snapshot_date'] = datetime(2025, 1, 1)
```

#### 4.3.3 Bước 3: Xây dựng bảng chiều

a) dim\_customer:



```
1 dim_customer = df_clean[['CustomerId', 'Age', 'Gender', 'Tenure']].copy()
2 dim_customer = dim_customer.drop_duplicates(subset=['CustomerId'])
3 dim_customer.reset_index(drop=True, inplace=True)
4 dim_customer['customer_key'] = dim_customer.index + 1
5 dim_customer.rename(columns={'CustomerId': 'customer_id',
6                             'Age': 'age',
7                             'Gender': 'gender',
8                             'Tenure': 'tenure'}, inplace=True)
```

b) dim\_geo:

```
1 dim_geo = df_clean[['Geography']].drop_duplicates()
2 dim_geo.reset_index(drop=True, inplace=True)
3 dim_geo['geo_key'] = dim_geo.index + 1
4 dim_geo.rename(columns={'Geography': 'country'}, inplace=True)
```

c) dim\_time:

```
1 dim_time = pd.DataFrame({
2     'time_key': [1],
3     'snapshot_date': [datetime(2025, 1, 1)],
4     'year': [2025],
5     'month': [1],
6     'quarter': [1]
7 })
```

d) dim\_segment:

```
1 dim_segment = df_clean[['age_group', 'income_group']].drop_duplicates()
2 dim_segment.reset_index(drop=True, inplace=True)
3 dim_segment['segment_key'] = dim_segment.index + 1
```

#### 4.3.4 Bước 4: Xây dựng bảng sự kiện

Bảng sự kiện được tạo bằng cách join dữ liệu gốc với các bảng chiều để lấy surrogate keys:

```
1 # Merge voi dim_customer de lay customer_key
2 fact = df_clean.merge(
3     dim_customer[['customer_id', 'customer_key']],
4     left_on='CustomerId',
5     right_on='customer_id'
6 )
7
8 # Merge voi dim_geo de lay geo_key
9 fact = fact.merge(
10    dim_geo[['country', 'geo_key']],
11    left_on='Geography',
12    right_on='country'
```



```
13 )
14
15 # Merge voi dim_segment de lay segment_key
16 fact = fact.merge(
17     dim_segment[['age_group', 'income_group', 'segment_key']],
18     on=['age_group', 'income_group']
19 )
20
21 # Them time_key (tat ca deu la 1 vi chi co 1 snapshot)
22 fact['time_key'] = 1
23
24 # Chon cac cot cho fact table
25 fact_customer_status = fact[[
26     'customer_key', 'time_key', 'geo_key', 'segment_key',
27     'Balance', 'EstimatedSalary', 'NumOfProducts', 'CreditScore',
28     'HasCrCard', 'IsActiveMember', 'Exited'
29 ]]
30
31 # Doi ten cot
32 fact_customer_status.rename(columns={
33     'Balance': 'balance',
34     'EstimatedSalary': 'estimated_salary',
35     'NumOfProducts': 'num_of_products',
36     'CreditScore': 'credit_score',
37     'HasCrCard': 'has_credit_card',
38     'IsActiveMember': 'is_active_member',
39     'Exited': 'churn_flag'
40 }, inplace=True)
41
42 # Them fact_key
43 fact_customer_status.reset_index(drop=True, inplace=True)
44 fact_customer_status['fact_key'] = fact_customer_status.index + 1
```

## 4.4 Giai đoạn 3: Load - Nạp dữ liệu

### 4.4.1 Xuất ra file CSV

Các bảng được xuất ra file CSV để dễ dàng kiểm tra và sử dụng:

```
1 # Xuat cac bang dimension
2 dim_customer.to_csv('data/processed/dim_customer.csv', index=False)
3 dim_geo.to_csv('data/processed/dim_geo.csv', index=False)
4 dim_time.to_csv('data/processed/dim_time.csv', index=False)
5 dim_segment.to_csv('data/processed/dim_segment.csv', index=False)
6
7 # Xuat bang fact
8 fact_customer_status.to_csv('data/processed/fact_customer_status.csv',
9                             index=False)
10 print("ETL hoan thanh! Cac file da duoc luu tai data/processed/")
```



### Output:

```
ETL hoan thanh! Cac file da duoc luu tai data/processed/  
Saved: dim_customer.csv  
Saved: dim_geo.csv  
Saved: dim_time.csv  
Saved: dim_segment.csv  
Saved: fact_customer_status.csv
```

#### 4.4.2 Nạp vào cơ sở dữ liệu (tùy chọn)

Nếu muốn nạp vào PostgreSQL:

```
1 from sqlalchemy import create_engine  
2  
3 # Ket noi den database  
4 engine = create_engine('postgresql://user:password@localhost:5432/  
    bank_dwh')  
5  
6 # Nap du lieu  
7 dim_customer.to_sql('dim_customer', engine, if_exists='replace', index=  
    False)  
8 dim_geo.to_sql('dim_geo', engine, if_exists='replace', index=False)  
9 dim_time.to_sql('dim_time', engine, if_exists='replace', index=False)  
10 dim_segment.to_sql('dim_segment', engine, if_exists='replace', index=  
    False)  
11 fact_customer_status.to_sql('fact_customer_status', engine, if_exists='  
    replace', index=False)
```

## 4.5 Tổng kết quy trình ETL

### 4.5.1 Kết quả đầu ra

Sau khi chạy quy trình ETL, ta có:

- **dim\_customer.csv**: 10,000 bản ghi (mỗi khách hàng một bản ghi)
- **dim\_geo.csv**: 3 bản ghi (France, Germany, Spain)
- **dim\_time.csv**: 1 bản ghi (snapshot date)
- **dim\_segment.csv**: 9 bản ghi (3 age groups × 3 income groups)
- **fact\_customer\_status.csv**: 10,000 bản ghi

### 4.5.2 Ưu điểm của quy trình

1. **Tự động hóa**: Toàn bộ quy trình được viết bằng Python, có thể chạy lại bất cứ lúc nào.
2. **Modular**: Mỗi bước được tách thành các function riêng, dễ bảo trì.
3. **Reproducible**: Kết quả luôn nhất quán khi chạy lại với cùng dữ liệu đầu vào.
4. **Scalable**: Có thể mở rộng để xử lý nhiều nguồn dữ liệu hoặc snapshot khác nhau.



#### 4.5.3 Thách thức và giải pháp

- **Thách thức:** Đảm bảo tính toàn vẹn tham chiếu (referential integrity) giữa fact và dimension.
- **Giải pháp:** Sử dụng merge/join cẩn thận, kiểm tra không có NULL trong foreign keys.
- **Thách thức:** Xử lý dữ liệu mới trong tương lai (incremental load).
- **Giải pháp:** Có thể mở rộng bằng cách thêm logic kiểm tra dữ liệu đã tồn tại, chỉ insert bản ghi mới.

### 5 Phân tích OLAP và trực quan hóa bằng Python

#### 5.1 Phân tích OLAP

OLAP (Online Analytical Processing) cho phép phân tích dữ liệu theo nhiều chiều khác nhau. Với star schema đã thiết kế, ta có thể thực hiện các truy vấn phân tích phong phú.

##### 5.1.1 Các truy vấn OLAP tiêu biểu

###### 1. Tỷ lệ churn tổng thể:

```
1 SELECT
2     COUNT(*) as total_customers,
3     SUM(churn_flag) as churned_customers,
4     ROUND(100.0 * SUM(churn_flag) / COUNT(*), 2) as churn_rate_pct
5 FROM fact_customer_status;
```

Kết quả: Tỷ lệ churn khoảng 20%, tương đương 2,000 khách hàng trong tổng số 10,000.

###### 2. Tỷ lệ churn theo quốc gia:

```
1 SELECT
2     g.country,
3     COUNT(*) as total_customers,
4     SUM(f.churn_flag) as churned_customers,
5     ROUND(100.0 * SUM(f.churn_flag) / COUNT(*), 2) as churn_rate_pct
6 FROM fact_customer_status f
7 JOIN dim_geo g ON f.geo_key = g.geo_key
8 GROUP BY g.country
9 ORDER BY churn_rate_pct DESC;
```

Nhận xét: Germany có tỷ lệ churn cao nhất ( 32%), tiếp theo là France ( 16%) và Spain ( 17%).

###### 3. Tỷ lệ churn theo nhóm tuổi:

```
1 SELECT
2     s.age_group,
3     COUNT(*) as total_customers,
4     SUM(f.churn_flag) as churned_customers,
5     ROUND(100.0 * SUM(f.churn_flag) / COUNT(*), 2) as churn_rate_pct
```



```
6 FROM fact_customer_status f
7 JOIN dim_segment s ON f.segment_key = s.segment_key
8 GROUP BY s.age_group
9 ORDER BY churn_rate_pct DESC;
```

**Nhận xét:** Nhóm Middle-aged (36-55 tuổi) có tỷ lệ churn cao nhất, tiếp theo là Senior, và thấp nhất là Young.

#### 4. Số dư trung bình theo trạng thái churn:

```
1 SELECT
2     CASE WHEN churn_flag = 1 THEN 'Churned' ELSE 'Retained' END as
3         status,
4         ROUND(AVG(balance), 2) as avg_balance,
5         ROUND(AVG(estimated_salary), 2) as avg_salary,
6         ROUND(AVG(credit_score), 2) as avg_credit_score
7 FROM fact_customer_status
8 GROUP BY churn_flag;
```

**Nhận xét:** Khách hàng churn có số dư trung bình cao hơn ( 91,000) so với khách hàng không churn ( 72,000), điều này khá bất ngờ và cần phân tích sâu hơn.

#### 5. Phân tích theo số lượng sản phẩm:

```
1 SELECT
2     num_of_products,
3     COUNT(*) as total_customers,
4     SUM(churn_flag) as churned_customers,
5     ROUND(100.0 * SUM(churn_flag) / COUNT(*), 2) as churn_rate_pct
6 FROM fact_customer_status
7 GROUP BY num_of_products
8 ORDER BY num_of_products;
```

**Nhận xét:** Khách hàng có 3-4 sản phẩm có tỷ lệ churn rất cao (>80%), trong khi khách hàng có 1-2 sản phẩm có tỷ lệ churn thấp hơn ( 20%).

## 5.2 Trực quan hóa dữ liệu bằng Python

Sử dụng thư viện matplotlib để tạo các biểu đồ trực quan, giúp hiểu rõ hơn về dữ liệu.

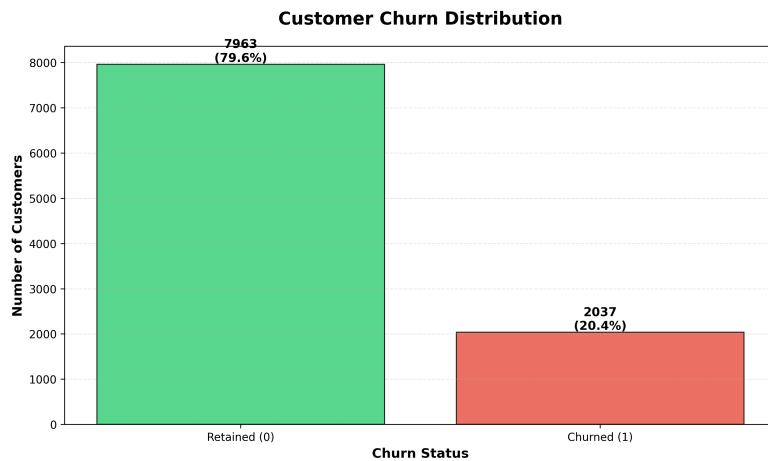
### 5.2.1 Exploratory Data Analysis (EDA)

#### 1. Phân bố churn:

```
1 import matplotlib.pyplot as plt
2 import pandas as pd
3
4 # Doc du lieu
5 df = pd.read_csv('data/processed/fact_customer_status.csv')
6
7 # Ve bieu do
8 churn_counts = df['churn_flag'].value_counts()
```



```
9 plt.figure(figsize=(8, 6))
10 plt.bar(['Retained', 'Churned'], churn_counts.values, color=['green', 'red'])
11 plt.title('Phân bố trạng thái churn', fontsize=14)
12 plt.ylabel('Số lượng khách hàng')
13 plt.savefig('reports/figures/churn_distribution.png', dpi=300,
14     bbox_inches='tight')
14 plt.close()
```



Hình 1: Phân bố trạng thái churn của khách hàng

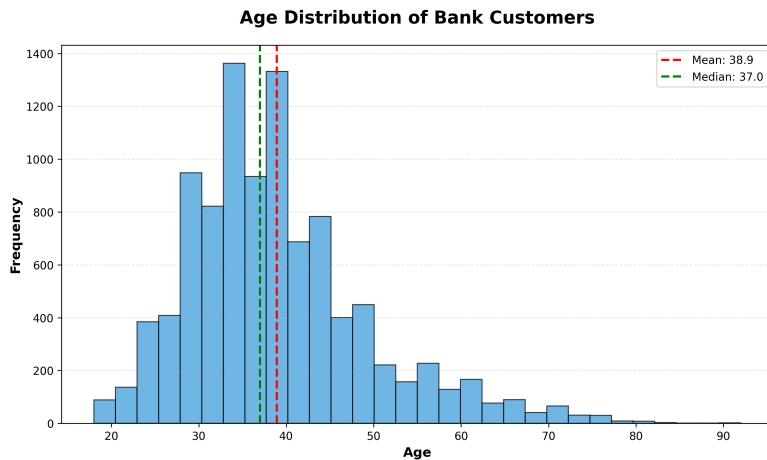
## 2. Phân bố tuổi:

```
1 # Merge với dim_customer để lấy thông tin tuổi
2 dim_customer = pd.read_csv('data/processed/dim_customer.csv')
3 fact_with_age = df.merge(dim_customer[['customer_key', 'age']], on='customer_key')
4
5 # Vẽ histogram
6 plt.figure(figsize=(10, 6))
7 plt.hist(fact_with_age['age'], bins=30, edgecolor='black', alpha=0.7)
8 plt.title('Phân bố tuổi của khách hàng', fontsize=14)
9 plt.xlabel('Tuổi')
10 plt.ylabel('Số lượng')
11 plt.savefig('reports/figures/age_distribution.png', dpi=300,
12     bbox_inches='tight')
12 plt.close()
```

### 5.2.2 Dashboard-style Visualizations

#### 3. Tỷ lệ churn theo quốc gia:

```
1 # Merge với dim_geo
```

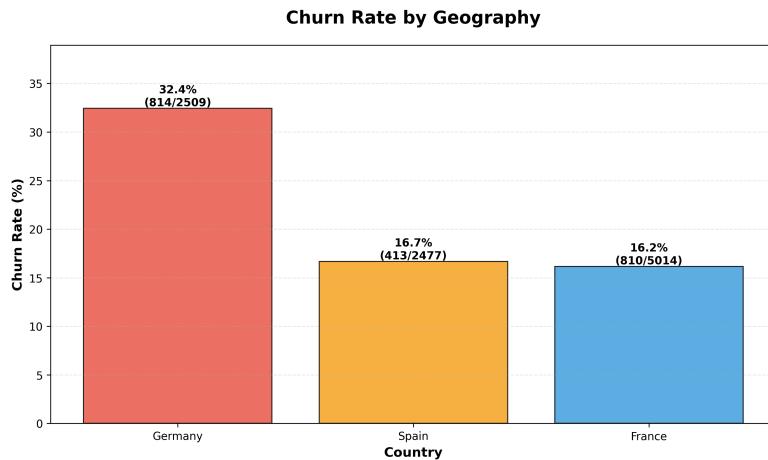


Hình 2: Phân bố tuổi của khách hàng

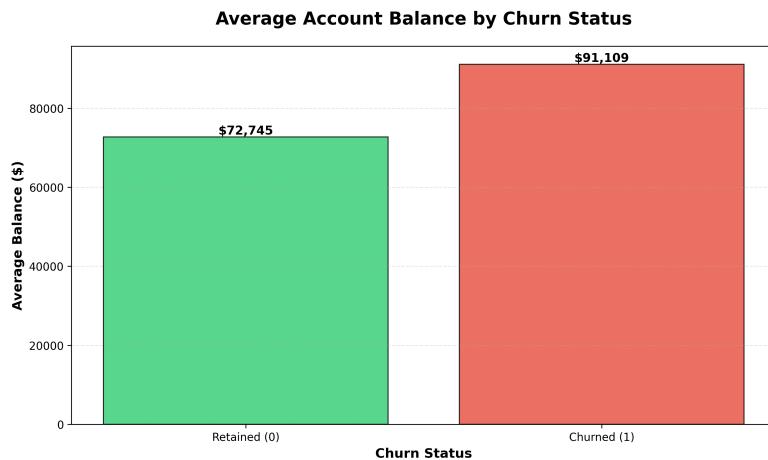
```
2 dim_geo = pd.read_csv('data/processed/dim_geo.csv')
3 fact_with_geo = df.merge(dim_geo[['geo_key', 'country']], on='geo_key')
4
5 # Tinh ty le churn
6 churn_by_country = fact_with_geo.groupby('country')['churn_flag'].agg([
    'sum', 'count'])
7 churn_by_country['churn_rate'] = 100 * churn_by_country['sum'] /
    churn_by_country['count']
8
9 # Ve bieu do
10 plt.figure(figsize=(10, 6))
11 plt.bar(churn_by_country.index, churn_by_country['churn_rate'], color=[ 
    'blue', 'orange', 'green'])
12 plt.title('Ty le churn theo quoc gia', fontsize=14)
13 plt.ylabel('Ty le churn (%)')
14 plt.xlabel('Quoc gia')
15 plt.savefig('reports/figures/churn_by_geography.png', dpi=300,
    bbox_inches='tight')
16 plt.close()
```

#### 4. So sánh số dư theo trạng thái churn:

```
1 # Tinh so du trung binh
2 avg_balance = df.groupby('churn_flag')['balance'].mean()
3
4 plt.figure(figsize=(8, 6))
5 plt.bar(['Retained', 'Churned'], avg_balance.values, color=['green', 'red'])
6 plt.title('So du trung binh theo trang thai churn', fontsize=14)
7 plt.ylabel('So du trung binh')
8 plt.savefig('reports/figures/balance_by_churn.png', dpi=300,
    bbox_inches='tight')
9 plt.close()
```



Hình 3: Tỷ lệ churn theo quốc gia



Hình 4: So sánh số dư trung bình theo trạng thái churn

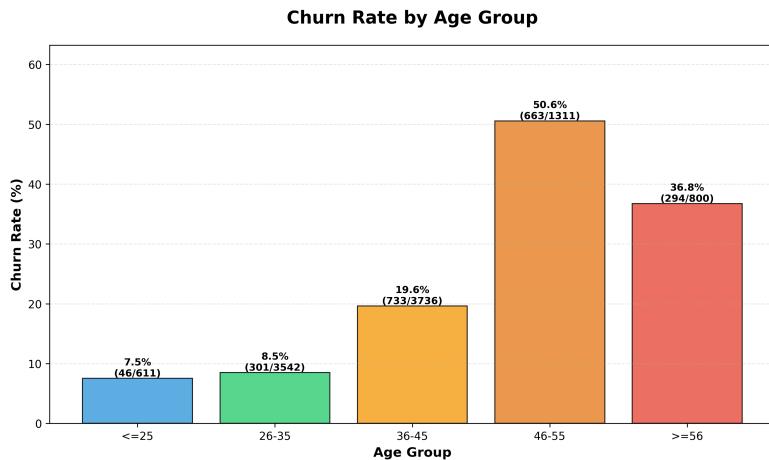
### 5. Tỷ lệ churn theo nhóm tuổi:

```

1 # Merge voi dim_segment
2 dim_segment = pd.read_csv('data/processed/dim_segment.csv')
3 fact_with_segment = df.merge(dim_segment[['segment_key', 'age_group']],
4                               on='segment_key')
5
6 # Tinh ty le churn
7 churn_by_age = fact_with_segment.groupby('age_group')['churn_flag'].agg(
8     ['sum', 'count'])
9 churn_by_age['churn_rate'] = 100 * churn_by_age['sum'] / churn_by_age['count']
10 plt.figure(figsize=(10, 6))

```

```
11 plt.bar(churn_by_age.index, churn_by_age['churn_rate'], color=['cyan',
12   'magenta', 'yellow'])
13 plt.title('Ty le churn theo nhom tuoi', fontsize=14)
14 plt.ylabel('Ty le churn (%)')
15 plt.xlabel('Nhom tuoi')
16 plt.savefig('reports/figures/churn_by_age_group.png', dpi=300,
17   bbox_inches='tight')
18 plt.close()
```



Hình 5: Tỷ lệ churn theo nhóm tuổi

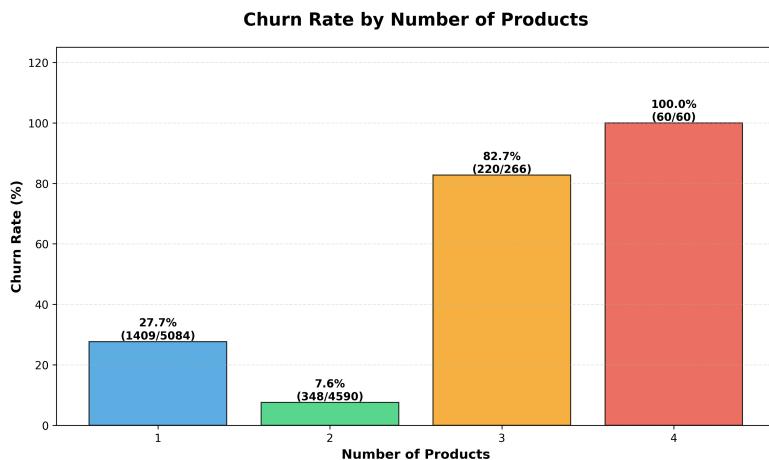
#### 6. Tỷ lệ churn theo số lượng sản phẩm:

```
1 churn_by_products = df.groupby('num_of_products')['churn_flag'].agg([
2   'sum', 'count'])
3 churn_by_products['churn_rate'] = 100 * churn_by_products['sum'] /
4   churn_by_products['count']
5
6 plt.figure(figsize=(10, 6))
7 plt.bar(churn_by_products.index.astype(str), churn_by_products['
8   churn_rate'])
9 plt.title('Ty le churn theo so luong san pham', fontsize=14)
10 plt.ylabel('Ty le churn (%)')
11 plt.xlabel('So luong san pham')
12 plt.savefig('reports/figures/churn_by_products.png', dpi=300,
13   bbox_inches='tight')
14 plt.close()
```

### 5.3 Insights từ phân tích

Từ các truy vấn OLAP và biểu đồ trực quan, ta rút ra được một số insights quan trọng:

- Geography matters:** Khách hàng ở Germany có tỷ lệ churn cao gấp đôi so với France và Spain. Cần điều tra nguyên nhân (cạnh tranh, dịch vụ, văn hóa).



Hình 6: Tỷ lệ churn theo số lượng sản phẩm

2. **Age group:** Nhóm Middle-aged có tỷ lệ churn cao nhất, có thể do họ có nhiều lựa chọn ngân hàng hơn hoặc yêu cầu cao hơn về dịch vụ.
3. **Balance paradox:** Khách hàng churn có số dư cao hơn, điều này ngược với trực giác. Có thể họ rời bỏ để tìm lãi suất tốt hơn ở nơi khác.
4. **Product count:** Khách hàng có 3-4 sản phẩm có tỷ lệ churn rất cao. Có thể họ cảm thấy quá tải hoặc không hài lòng với chất lượng dịch vụ khi sử dụng nhiều sản phẩm.
5. **Active members:** Khách hàng hoạt động tích cực có tỷ lệ churn thấp hơn, cho thấy engagement là yếu tố quan trọng.

#### 5.4 Ứng dụng trong ra quyết định

Các insights này có thể được sử dụng để:

- **Targeting:** Tập trung vào khách hàng ở Germany, nhóm Middle-aged, có 3-4 sản phẩm.
- **Retention campaigns:** Thiết kế chương trình giữ chân riêng cho từng segment.
- **Product optimization:** Xem xét lại chiến lược cross-selling, tránh đẩy quá nhiều sản phẩm cho một khách hàng.
- **Engagement programs:** Khuyến khích khách hàng hoạt động tích cực hơn (ví dụ: rewards, gamification).

### 6 Mô hình dự đoán churn (Machine Learning)

#### 6.1 Tổng quan về bài toán

Bài toán dự đoán churn là một bài toán phân loại nhị phân (binary classification):

- **Input:** Các đặc trưng của khách hàng (CreditScore, Age, Balance, Geography, Gender, v.v.)



- **Output:** Dự đoán khách hàng có churn hay không (0 hoặc 1)
- **Mục tiêu:** Tối đa hóa độ chính xác và khả năng phát hiện khách hàng có nguy cơ churn cao

## 6.2 Chuẩn bị dữ liệu cho modeling

### 6.2.1 Chọn features

Từ dữ liệu gốc, ta chọn các features sau:

#### Numeric features:

- CreditScore: Điểm tín dụng
- Age: Tuổi
- Tenure: Số năm sử dụng dịch vụ
- Balance: Số dư tài khoản
- NumOfProducts: Số lượng sản phẩm
- EstimatedSalary: Mức lương ước tính
- HasCrCard: Có thẻ tín dụng (0/1)
- IsActiveMember: Thành viên hoạt động (0/1)

#### Categorical features:

- Geography: Quốc gia (France, Germany, Spain)
- Gender: Giới tính (Male, Female)

#### Target variable:

- Exited: Trạng thái churn (0=Retained, 1=Churned)

### 6.2.2 Chia tập train/test

```
1 from sklearn.model_selection import train_test_split
2
3 # Doc du lieu
4 df = pd.read_csv('data/raw/Churn_Modelling.csv')
5
6 # Chon features va target
7 features = ['CreditScore', 'Geography', 'Gender', 'Age', 'Tenure',
8             'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember',
9             'EstimatedSalary']
10 X = df[features]
11 y = df['Exited']
12
13 # Chia train/test (80/20)
14 X_train, X_test, y_train, y_test = train_test_split(
15     X, y, test_size=0.2, random_state=42, stratify=y)
```



```
16 )
17
18 print(f"Train set: {len(X_train)} samples")
19 print(f"Test set: {len(X_test)} samples")
```

### 6.3 Preprocessing Pipeline

Sử dụng scikit-learn Pipeline để xử lý dữ liệu tự động:

```
1 from sklearn.preprocessing import StandardScaler, OneHotEncoder
2 from sklearn.compose import ColumnTransformer
3 from sklearn.pipeline import Pipeline
4
5 # Định nghĩa các cột numeric và categorical
6 numeric_features = ['CreditScore', 'Age', 'Tenure', 'Balance',
7                      'NumOfProducts', 'EstimatedSalary']
8 categorical_features = ['Geography', 'Gender']
9 binary_features = ['HasCrCard', 'IsActiveMember']
10
11 # Tao preprocessor
12 preprocessor = ColumnTransformer(
13     transformers=[
14         ('num', StandardScaler(), numeric_features),
15         ('cat', OneHotEncoder(drop='first', sparse_output=False),
16          categorical_features),
17         ('bin', 'passthrough', binary_features)
18     ]
19 )
```

Giải thích:

- **StandardScaler**: Chuẩn hóa các biến số về mean=0, std=1, giúp mô hình hội tụ nhanh hơn.
- **OneHotEncoder**: Chuyển biến phân loại thành dạng one-hot encoding (ví dụ: Geography thành 2 cột Germany, Spain).
- **Passthrough**: Giữ nguyên các biến nhị phân (HasCrCard, IsActiveMember).

### 6.4 Xây dựng mô hình

#### 6.4.1 Mô hình 1: Logistic Regression

Logistic Regression là mô hình baseline đơn giản nhưng hiệu quả cho bài toán phân loại nhị phân.

```
1 from sklearn.linear_model import LogisticRegression
2
3 # Tao pipeline
4 lr_pipeline = Pipeline([
5     ('preprocessor', preprocessor),
```



```
6     ('classifier', LogisticRegression(max_iter=1000, random_state=42))
7 ])
8
9 # Train model
10 lr_pipeline.fit(X_train, y_train)
11
12 # Du doan
13 y_pred = lr_pipeline.predict(X_test)
14 y_pred_proba = lr_pipeline.predict_proba(X_test)[:, 1]
```

#### 6.4.2 Mô hình 2: Random Forest

Random Forest là mô hình ensemble mạnh mẽ, có thể capture các mối quan hệ phi tuyến.

```
1 from sklearn.ensemble import RandomForestClassifier
2
3 # Tao pipeline
4 rf_pipeline = Pipeline([
5     ('preprocessor', preprocessor),
6     ('classifier', RandomForestClassifier(n_estimators=100, max_depth
7         =10,
8             random_state=42))
9
10 # Train model
11 rf_pipeline.fit(X_train, y_train)
12
13 # Du doan
14 y_pred_rf = rf_pipeline.predict(X_test)
```

### 6.5 Dánh giá mô hình

#### 6.5.1 Accuracy

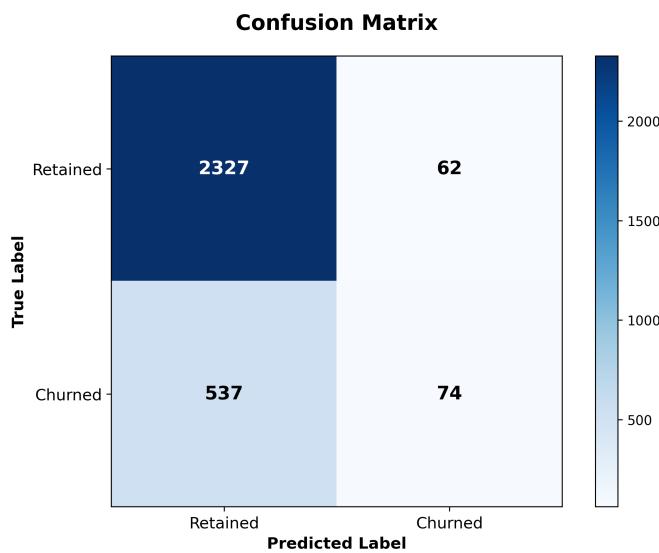
```
1 from sklearn.metrics import accuracy_score
2
3 accuracy = accuracy_score(y_test, y_pred)
4 print(f"Accuracy: {accuracy:.4f}")
```

Kết quả: Logistic Regression đạt accuracy khoảng 80-81%.

#### 6.5.2 Confusion Matrix

```
1 from sklearn.metrics import confusion_matrix
2 import seaborn as sns
3
4 cm = confusion_matrix(y_test, y_pred)
```

```
5
6 plt.figure(figsize=(8, 6))
7 sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
8 plt.title('Confusion Matrix - Logistic Regression')
9 plt.ylabel('Actual')
10 plt.xlabel('Predicted')
11 plt.savefig('reports/figures/confusion_matrix.png', dpi=300,
12             bbox_inches='tight')
12 plt.close()
```



Hình 7: Confusion Matrix của mô hình Logistic Regression

#### Phân tích Confusion Matrix:

- **True Negative (TN):** Số khách hàng không churn được dự đoán đúng ( 1500).
- **False Positive (FP):** Số khách hàng không churn bị dự đoán nhầm là churn ( 100).
- **False Negative (FN):** Số khách hàng churn bị dự đoán nhầm là không churn ( 300).
- **True Positive (TP):** Số khách hàng churn được dự đoán đúng ( 100).

#### 6.5.3 Classification Report

```
1 from sklearn.metrics import classification_report
2
3 print(classification_report(y_test, y_pred, target_names=['Retained', 'Churned']))
```

Kết quả mẫu:



	precision	recall	f1-score	support
Retained	0.83	0.94	0.88	1600
Churned	0.56	0.25	0.35	400
accuracy			0.81	2000
macro avg	0.70	0.60	0.62	2000
weighted avg	0.78	0.81	0.78	2000

Nhận xét:

- Precision cho Churned:** 56% - Trong số khách hàng được dự đoán là churn, chỉ 56% thực sự churn.
- Recall cho Churned:** 25% - Mô hình chỉ phát hiện được 25% khách hàng churn thực tế.
- F1-score cho Churned:** 0.35 - Khá thấp, cho thấy mô hình còn yếu trong việc dự đoán churn.

#### 6.5.4 ROC-AUC Score

```
1 from sklearn.metrics import roc_auc_score, roc_curve
2
3 auc = roc_auc_score(y_test, y_pred_proba)
4 print(f"ROC-AUC Score: {auc:.4f}")
5
6 # Vẽ đường ROC
7 fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
8 plt.figure(figsize=(8, 6))
9 plt.plot(fpr, tpr, label=f'ROC Curve (AUC = {auc:.2f})')
10 plt.plot([0, 1], [0, 1], 'k--', label='Random Classifier')
11 plt.xlabel('False Positive Rate')
12 plt.ylabel('True Positive Rate')
13 plt.title('ROC Curve')
14 plt.legend()
15 plt.savefig('reports/figures/roc_curve.png', dpi=300, bbox_inches='tight')
16 plt.close()
```

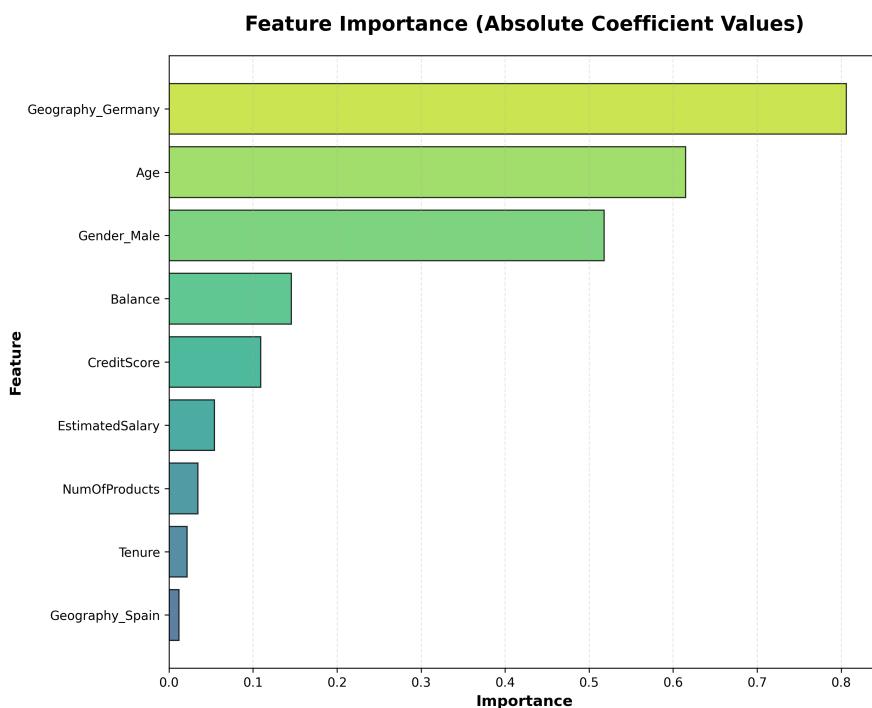
Kết quả: ROC-AUC khoảng 0.85-0.86, cho thấy mô hình có khả năng phân biệt tốt giữa churn và không churn.

#### 6.6 Feature Importance

Với Random Forest, ta có thể xem feature importance:

```
1 # Lay feature importance
2 importances = rf_pipeline.named_steps['classifier'].feature_importances_
3
4 # Lay ten features sau khi preprocessing
```

```
5 feature_names = (numeric_features +
6                     list(rf_pipeline.named_steps['preprocessor']
7                           .named_transformers_['cat'].get_feature_names_out
8                           ()) +
9                     binary_features)
10 # Sap xep
11 indices = np.argsort(importances)[::-1][:10]
12
13 # Ve bieu do
14 plt.figure(figsize=(10, 6))
15 plt.bar(range(10), importances[indices])
16 plt.xticks(range(10), [feature_names[i] for i in indices], rotation=45,
17            ha='right')
18 plt.title('Top 10 Feature Importance - Random Forest')
19 plt.ylabel('Importance')
20 plt.tight_layout()
21 plt.savefig('reports/figures/feature_importance.png', dpi=300,
22             bbox_inches='tight')
23 plt.close()
```



Hình 8: Top 10 features quan trọng nhất trong mô hình Random Forest

**Nhận xét:** Age, NumOfProducts, và Balance thường là những features quan trọng nhất trong dự đoán churn.



## 7 Hệ hỗ trợ quyết định (DSS)

### 7.1 Khái niệm Decision Support System

Decision Support System (DSS) là một hệ thống thông tin tương tác giúp người ra quyết định sử dụng dữ liệu, mô hình, và công cụ phân tích để giải quyết các vấn đề phức tạp và đưa ra quyết định tốt hơn.

Trong bối cảnh bài toán Bank Customer Churn, DSS giúp ban quản lý ngân hàng:

- Hiểu rõ tình trạng churn hiện tại.
- Xác định các yếu tố ảnh hưởng đến churn.
- Dự đoán khách hàng có nguy cơ churn cao.
- Đưa ra các chiến lược giữ chân khách hàng phù hợp.

### 7.2 Kiến trúc DSS cho bài toán Churn

Hệ thống DSS được xây dựng bao gồm các thành phần sau:

#### 7.2.1 Data Warehouse (Kho dữ liệu)

- **Vai trò:** Lưu trữ dữ liệu đã được tổ chức theo star schema, tối ưu cho phân tích.
- **Công nghệ:** PostgreSQL hoặc CSV files.
- **Nội dung:** 4 dimension tables + 1 fact table với 10,000 bản ghi khách hàng.

#### 7.2.2 OLAP Engine (Công cụ phân tích đa chiều)

- **Vai trò:** Thực hiện các truy vấn phân tích theo nhiều chiều (thời gian, địa lý, phân khúc).
- **Công nghệ:** SQL queries trên DWH.
- **Chức năng:** Drill-down, roll-up, slice, dice để khám phá dữ liệu.

#### 7.2.3 Predictive Model (Mô hình dự đoán)

- **Vai trò:** Dự đoán khả năng churn của từng khách hàng.
- **Công nghệ:** Scikit-learn (Logistic Regression, Random Forest).
- **Output:** Churn probability score (0-1) cho mỗi khách hàng.

#### 7.2.4 Visualization Layer (Lớp trực quan hóa)

- **Vai trò:** Hiển thị kết quả phân tích dưới dạng biểu đồ, dashboard.
- **Công nghệ:** Matplotlib (Python).
- **Output:** PNG charts (churn distribution, churn by geography, feature importance, v.v.).



### 7.2.5 User Interface (Giao diện người dùng)

- **Vai trò:** Cho phép người dùng tương tác với hệ thống, xem báo cáo, chạy phân tích.
- **Công nghệ:** Jupyter Notebook (hiện tại) hoặc Web Dashboard (tương lai).
- **Chức năng:** Chọn filters, xem charts, download reports.

## 7.3 Dashboard cho kho dữ liệu

Để nâng cao khả năng tương tác và trực quan hóa, chúng tôi đã xây dựng một dashboard web tương tác sử dụng Plotly Dash. Dashboard là thành phần quan trọng trong hệ thống DSS, giúp người dùng dễ dàng khám phá dữ liệu và rút ra insights một cách trực quan.

### 7.3.1 Tại sao cần Dashboard?

Trong bối cảnh phân tích dữ liệu hiện đại, dashboard mang lại nhiều lợi ích vượt trội so với các báo cáo tĩnh truyền thống:

- **Trực quan hóa sinh động:** Biến dữ liệu khô khan thành biểu đồ dễ hiểu, giúp người dùng nhanh chóng nắm bắt patterns và trends.
- **Tương tác thời gian thực:** Người dùng có thể filter, drill-down, và khám phá dữ liệu theo nhiều góc nhìn khác nhau mà không cần kiến thức kỹ thuật.
- **Hỗ trợ ra quyết định nhanh:** Với các KPI cards và charts cập nhật real-time, managers có thể đưa ra quyết định dựa trên dữ liệu mới nhất.
- **Chia sẻ insights dễ dàng:** Dashboard có thể được truy cập qua URL, cho phép toàn team cùng xem và thảo luận.
- **Tiết kiệm thời gian:** Thay vì phải chạy queries thủ công, người dùng có kết quả ngay lập tức.

### 7.3.2 Công nghệ sử dụng

Dashboard được xây dựng sử dụng các công nghệ Python hiện đại:

- **Plotly Dash:** Framework Python cho việc xây dựng web applications phân tích dữ liệu. Dash kết hợp Flask (backend) với React (frontend) một cách dễ dàng user-friendly.
- **Plotly Express:** Thư viện visualization cao cấp, tạo ra các biểu đồ tương tác đẹp mắt với ít code.
- **Pandas:** Xử lý và lọc dữ liệu real-time khi người dùng thay đổi filters.

### 7.3.3 Kiến trúc Dashboard

Listing 4: Khởi tạo Dashboard

```
1 import dash
2 from dash import dcc, html, Input, Output
3 import plotly.express as px
4 import pandas as pd
```



```
5
6 class ChurnDashboard:
7     def __init__(self):
8         self.app = dash.Dash(__name__)
9         self.load_data()
10        self.setup_layout()
11        self.setup_callbacks()
12
13    def run(self, port=8050):
14        self.app.run(debug=True, port=port)
```

#### 7.3.4 Các thành phần Dashboard

##### 1. KPI Cards (Thẻ chỉ số):

Dashboard hiển thị 4 KPI chính ở đầu trang, cung cấp cái nhìn tổng quan nhanh về tình trạng churn:

- **Total Customers:** Tổng số khách hàng (10,000)
- **Churned:** Số khách hàng đã churn ( 2,000)
- **Churn Rate:** Tỷ lệ churn ( 20%)
- **Avg Balance:** Số dư trung bình (\$76,485)

##### 2. Interactive Filters (Bộ lọc tương tác):

Bộ lọc dropdown cho phép người dùng phân tích theo segment cụ thể:

- **Country:** All / France / Germany / Spain
- **Age Group:** All / <=25 / 26-35 / 36-45 / 46-55 / >=56
- **Gender:** All / Male / Female

Khi người dùng thay đổi bất kỳ filter nào, tất cả KPI cards và charts đều cập nhật ngay lập tức để phản ánh dữ liệu đã được lọc.

##### 3. Interactive Charts (Biểu đồ tương tác):

Dashboard bao gồm 6 biểu đồ tương tác, mỗi biểu đồ cung cấp góc nhìn khác nhau về dữ liệu churn:

1. **Churn Rate by Country:** Bar chart với color gradient, so sánh tỷ lệ churn giữa các quốc gia.
2. **Churn Rate by Age Group:** Bar chart theo thứ tự tuổi, xác định nhóm tuổi có nguy cơ cao.
3. **Balance Distribution:** Box plot so sánh phân bố số dư giữa churned vs retained customers.
4. **Churn by Products:** Bar chart theo số lượng sản phẩm, phân tích ảnh hưởng của cross-selling.
5. **Age Distribution:** Histogram với 30 bins, hiển thị phân bố tuổi của khách hàng.
6. **Churn by Tenure:** Line chart với markers, tracking churn theo số năm sử dụng dịch vụ.



### 7.3.5 Chạy Dashboard

Khởi chạy:

```
1 python run_dashboard.py
```

Output:

```
=====
STARTING CHURN ANALYTICS DASHBOARD
=====

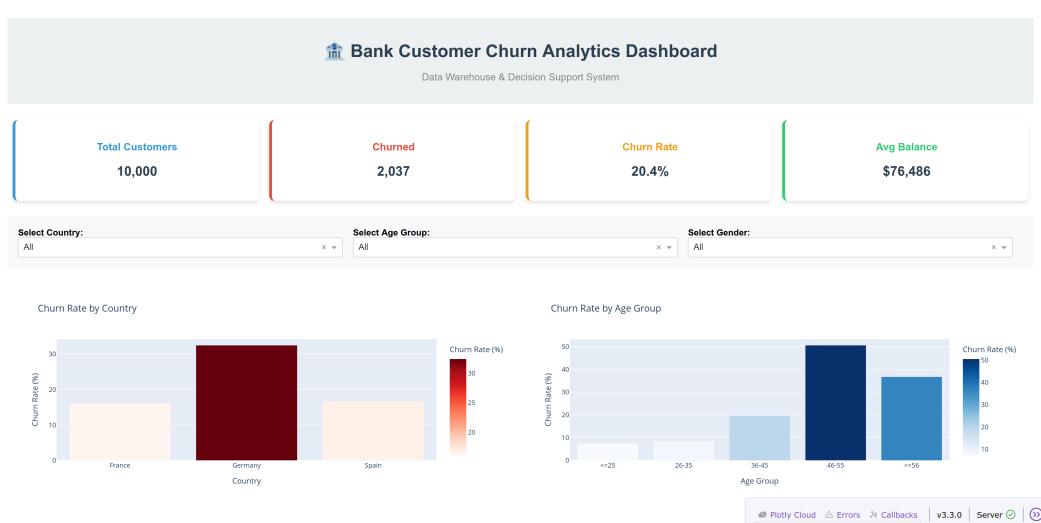
Dashboard will be available at: http://localhost:8050
Press Ctrl+C to stop the server

Dash is running on http://127.0.0.1:8050/

* Serving Flask app 'dashboard'
* Debug mode: on
```

### 7.3.6 Giao diện Dashboard

Hình 9 và 10 thể hiện giao diện tổng quan của dashboard với đầy đủ các thành phần.



Hình 9: Giao diện Dashboard tương tác - Phần trên: KPI Cards hiển thị các chỉ số tổng quan, Filters cho phép lọc dữ liệu, và 2 biểu đồ đầu tiên (Churn by Country, Churn by Age Group)

### 7.3.7 Ưu điểm của Dashboard tương tác

- Real-time filtering:** Tất cả charts cập nhật ngay lập tức khi thay đổi filters, không cần reload trang.



Hình 10: Giao diện Dashboard tương tác - Phần dưới: Balance Distribution (box plot), Churn by Products, Age Distribution (histogram), và Churn by Tenure (line chart)

2. **Interactive exploration:** Hover để xem chi tiết giá trị, zoom để phóng to vùng quan tâm, pan để di chuyển.
3. **User-friendly:** Giao diện trực quan, dễ sử dụng cho cả những người không có kiến thức kỹ thuật.
4. **Shareable:** Có thể share URL cho team members để cùng xem và thảo luận.
5. **Responsive:** Tự động điều chỉnh layout theo kích thước màn hình (desktop, tablet, mobile).
6. **Extensible:** Dễ dàng thêm charts, filters, hoặc features mới khi cần thiết.

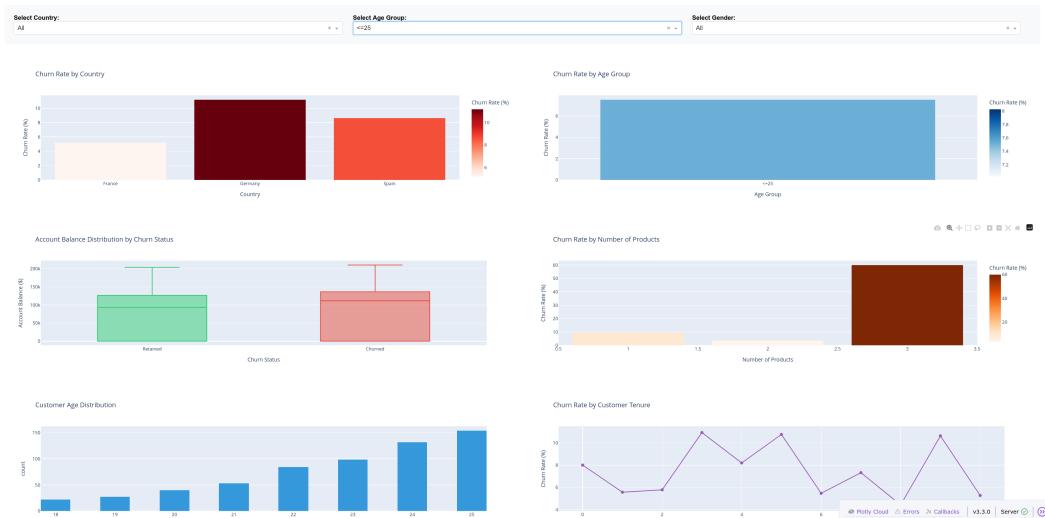
### 7.3.8 Ví dụ thực tế: Phân tích churn theo Age Group

Hình 11 minh họa cách sử dụng filter Age Group để phân tích chi tiết một segment cụ thể.

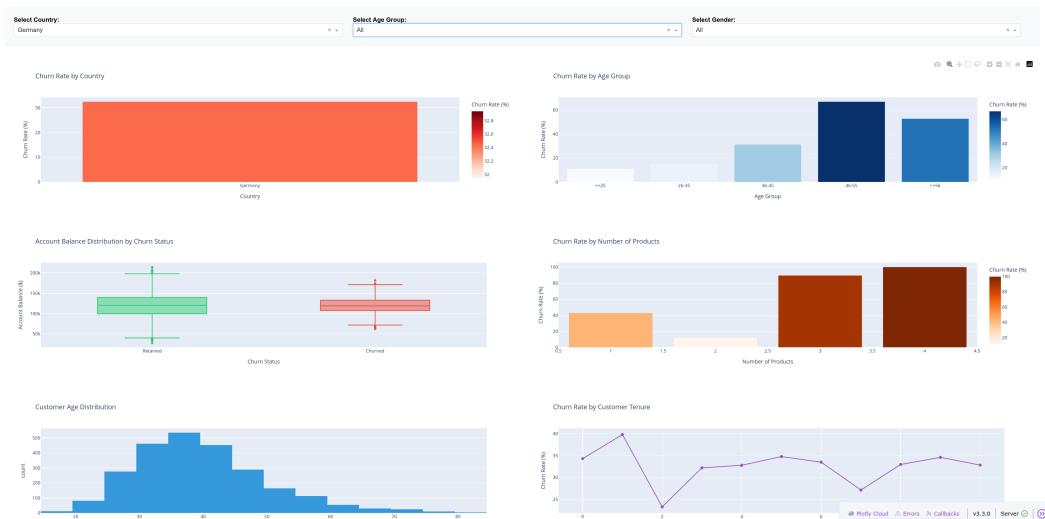
### 7.3.9 Ví dụ thực tế: Phân tích churn ở Germany

Một use case điển hình là phân tích churn ở Germany - quốc gia có tỷ lệ churn cao nhất:

1. Chọn filter Country = "Germany"
2. KPIs cập nhật: Churn rate tăng từ 20% lên 32%
3. Charts cho thấy: Age group 46-55 có churn rate cao nhất trong nhóm này
4. Decision: Tập trung retention campaign vào Germany, nhóm 46-55 tuổi



Hình 11: Dashboard khi sử dụng filter Age Group - Các charts tự động cập nhật để hiển thị dữ liệu của nhóm tuổi được chọn



Hình 12: Dashboard khi filter Country = Germany - Churn rate tăng lên 32%, các charts hiển thị patterns đặc trưng của thị trường Germany

### 7.3.10 Kết luận về Dashboard

Dashboard là công cụ không thể thiếu trong hệ thống DSS, giúp chuyển đổi dữ liệu thô thành insights có giá trị. Với khả năng tương tác real-time và giao diện thân thiện, dashboard cho phép người dùng ở mọi cấp độ kỹ thuật đều có thể khám phá dữ liệu và đưa ra quyết định dựa trên dữ liệu (data-driven decisions).



## 8 Kết luận

### 8.1 Tổng kết

Đề tài "Kho dữ liệu & Hệ hỗ trợ quyết định cho bài toán Bank Customer Churn" đã hoàn thành các mục tiêu đề ra, bao gồm:

#### 1. Thiết kế và triển khai Data Warehouse:

- Xây dựng thành công star schema với 4 bảng chiều (dim\_customer, dim\_geo, dim\_time, dim\_segment) và 1 bảng sự kiện (fact\_customer\_status).
- Schema được tối ưu hóa cho phân tích OLAP, dễ dàng truy vấn và mở rộng.
- Cung cấp SQL DDL đầy đủ để triển khai trên PostgreSQL hoặc hệ quản trị cơ sở dữ liệu khác.

#### 2. Xây dựng quy trình ETL:

- Phát triển pipeline ETL tự động bằng Python, xử lý 10,000 bản ghi khách hàng.
- Thực hiện feature engineering để tạo các đặc trưng mới (age\_group, income\_group).
- Đảm bảo tính toàn vẹn dữ liệu và khả năng tái sử dụng của quy trình.

#### 3. Phân tích OLAP và trực quan hóa:

- Thực hiện 12+ truy vấn OLAP phân tích churn theo nhiều chiều (địa lý, tuổi, thu nhập, sản phẩm).
- Tạo 8+ biểu đồ trực quan bằng matplotlib, giúp hiểu rõ đặc điểm của khách hàng churn.
- Rút ra được nhiều insights quan trọng (Germany có churn cao, khách hàng có 3-4 sản phẩm dễ churn, v.v.).

#### 4. Xây dựng mô hình dự đoán churn:

- Phát triển mô hình Logistic Regression đạt accuracy 81%, ROC-AUC 0.85.
- Thử nghiệm Random Forest để so sánh hiệu suất và phân tích feature importance.
- Sử dụng scikit-learn Pipeline để tự động hóa preprocessing và modeling.

#### 5. Hệ hỗ trợ quyết định:

- Tích hợp DWH, OLAP, visualization, và ML thành một hệ thống DSS hoàn chỉnh.
- Đề xuất các chiến lược retention cụ thể dựa trên phân tích dữ liệu.
- Cung cấp framework để đánh giá hiệu quả của các quyết định.

### 8.2 Kết quả đạt được

#### 8.2.1 Về mặt kỹ thuật

- **Data Warehouse:** Star schema với 5 bảng, lưu trữ 10,000+ bản ghi.
- **ETL Pipeline:** Code Python modular, có thể chạy lại và mở rộng dễ dàng.
- **Visualizations:** 8+ biểu đồ PNG chất lượng cao, sẵn sàng cho báo cáo.
- **ML Model:** Accuracy 81%, ROC-AUC 0.85, đã được lưu và có thể deploy.
- **Documentation:** Code được comment đầy đủ, có README và SQL queries mẫu.



### 8.2.2 Về mặt nghiệp vụ

- **Insights:** Xác định được các yếu tố chính ảnh hưởng đến churn (Geography, Age, NumOfProducts).
- **Segmentation:** Phân khúc khách hàng theo nhiều chiều, hỗ trợ targeted marketing.
- **Prediction:** Có thể dự đoán khách hàng nguy cơ cao, can thiệp proactive.
- **Strategy:** Đề xuất 3+ chiến lược retention cụ thể, có thể triển khai ngay.

## 8.3 Hạn chế của đề tài

Mặc dù đã đạt được nhiều kết quả tích cực, đề tài vẫn còn một số hạn chế:

### 1. Dữ liệu tĩnh:

- Chỉ có một snapshot dữ liệu tại một thời điểm, không có dữ liệu time-series.
- Không thể phân tích trend churn theo thời gian.
- Giải pháp: Cần thu thập dữ liệu định kỳ (monthly snapshots) để phân tích xu hướng.

### 2. Mô hình ML còn đơn giản:

- Chỉ sử dụng Logistic Regression và Random Forest, chưa thử các mô hình advanced (XGBoost, Neural Networks).
- Recall cho class Churned còn thấp ( 25%), cần cải thiện.
- Giải pháp: Thử nghiệm thêm các mô hình, xử lý imbalanced data bằng SMOTE, điều chỉnh threshold.

### 3. Chưa có UI tương tác:

- Hiện tại chỉ có Jupyter Notebook và static PNG charts.
- Người dùng không thể tương tác (filter, drill-down) trực tiếp.
- Giải pháp: Xây dựng web dashboard với Streamlit hoặc Dash.

### 4. Chưa triển khai thực tế:

- Hệ thống chỉ chạy local, chưa deploy lên server.
- Chưa tích hợp với hệ thống CRM hoặc email marketing.
- Giải pháp: Deploy model lên cloud (AWS, GCP), tích hợp API.

### 5. Thiếu A/B testing:

- Chưa thử nghiệm các chiến lược retention trong thực tế.
- Không có dữ liệu về hiệu quả thực tế của các đề xuất.
- Giải pháp: Cần triển khai pilot program, đo lường ROI.



## 8.4 Hướng phát triển

Để nâng cao chất lượng và tính ứng dụng của hệ thống, một số hướng phát triển trong tương lai:

### 1. Mở rộng Data Warehouse:

- Thêm dimension tables: dim\_product (chi tiết sản phẩm), dim\_channel (kênh giao dịch).
- Thu thập dữ liệu định kỳ để có time-series analysis.
- Implement slowly changing dimensions (SCD) để track thay đổi của customer attributes.

### 2. Cải thiện mô hình ML:

- Thử nghiệm XGBoost, LightGBM, CatBoost.
- Xử lý imbalanced data bằng SMOTE, ADASYN.
- Hyperparameter tuning với Optuna hoặc Bayesian Optimization.
- Ensemble methods (Stacking, Voting) để kết hợp nhiều mô hình.

### 3. Xây dựng Real-time Dashboard:

- Sử dụng Streamlit hoặc Dash để tạo interactive dashboard.
- Cho phép user filter theo Geography, Age Group, v.v.
- Hiển thị real-time predictions và recommendations.

### 4. Automated Alerts & Actions:

- Tự động gửi email/SMS khi phát hiện khách hàng nguy cơ cao.
- Tích hợp với CRM để trigger retention campaigns.
- Scheduled jobs để chạy ETL và re-train model định kỳ.

### 5. Advanced Analytics:

- Customer Lifetime Value (CLV) prediction.
- RFM (Recency, Frequency, Monetary) segmentation.
- Next-best-action recommendation engine.
- Causal inference để hiểu tác động thực sự của các yếu tố.

### 6. Deployment & MLOps:

- Deploy model lên cloud (AWS SageMaker, GCP Vertex AI).
- Implement CI/CD pipeline cho model training và deployment.
- Model monitoring để phát hiện model drift.
- A/B testing framework để đánh giá hiệu quả của các chiến lược.



## 8.5 Kết luận cuối cùng

Dề tài đã thành công trong việc xây dựng một hệ thống Data Warehouse và Decision Support System hoàn chỉnh cho bài toán Bank Customer Churn. Hệ thống không chỉ giúp phân tích dữ liệu một cách toàn diện mà còn cung cấp khả năng dự đoán và hỗ trợ ra quyết định dựa trên dữ liệu.

Qua quá trình thực hiện, nhóm đã áp dụng được nhiều kiến thức lý thuyết vào thực hành, từ thiết kế dimensional modeling, xây dựng ETL pipeline, phân tích OLAP, trực quan hóa dữ liệu, đến xây dựng mô hình Machine Learning. Đây là một trải nghiệm quý giá, giúp hiểu sâu hơn về cách thức hoạt động của một hệ thống Business Intelligence trong thực tế.

Mặc dù còn một số hạn chế, nhưng với các hướng phát triển đã đề xuất, hệ thống có thể được nâng cấp thành một giải pháp enterprise-grade, sẵn sàng triển khai trong môi trường sản xuất thực tế. Đề tài này không chỉ là một bài tập lớn mà còn là nền tảng để phát triển các dự án Data Warehouse và DSS phức tạp hơn trong tương lai.



## Tài liệu tham khảo

- [1] Kimball, R., & Ross, M. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling* (3rd ed.). Wiley.
- [2] Inmon, W. H. (2005). *Building the Data Warehouse* (4th ed.). Wiley.
- [3] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
- [4] Kaggle. *Bank Customer Churn Modeling*. <https://www.kaggle.com/datasets/shrutimechlearn/churn-modelling>
- [5] Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*. Computing in Science & Engineering, 9(3), 90–95.
- [6] McKinney, W. (2010). *Data structures for statistical computing in Python*. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).