

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

○-BOOK-○



HCMUTE

BÁO CÁO ĐỒ ÁN CUỐI KỲ
ĐỀ TÀI: PHÂN TÍCH DỰ ĐOÁN
KHÁCH HÀNG RÒI BỎ NGÂN HÀNG

Môn học: KHAI PHÁ DỮ LIỆU

Mã lớp học phần: DAMI330484_23_2_02

GVHD: ThS. Trần Trọng Bình

Nhóm sinh viên thực hiện: Nhóm 14

Nguyễn Tấn Phát	21133107
Trần Phan Quốc	21133108
Nguyễn Thị Thanh Hiền	21133032
Tăng Huỳnh Minh Tiến	21133088

Tp.HCM, tháng 5 năm 2024

BẢNG PHÂN CÔNG NHIỆM VỤ VÀ ĐÁNH GIÁ

MSSV	HỌ TÊN	ĐÓNG GÓP	% HOÀN THÀNH
21133107	Nguyễn Tân Phát	Tìm kiếm tập dữ liệu, Trích lọc đặc trưng - Đánh giá mức độ quan trọng của các thuộc tính bằng các mô hình học máy, Xây dựng mô hình.	100%
21133108	Trần Phan Quốc	Tìm kiếm tập dữ liệu, Phân tích dữ liệu, Chuẩn bị dữ liệu, Phân tích khám phá dữ liệu - Phân tích đa biến. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến liên tục với biến liên tục.	100%
21133032	Nguyễn Thị Thanh Hiền	Tìm kiếm tập dữ liệu, Phân tích khám phá dữ liệu - Phân tích đơn biến. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến phân loại với biến phân loại.	100%
21133088	Tăng Huỳnh Minh Tiên	Tìm kiếm tập dữ liệu, Khai phá luật kết hợp, Chuẩn hoá và mã hoá dữ liệu. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến phân loại với biến liên tục.	100%

MỤC LỤC

LỜI CẢM ƠN.....	1
LỜI MỞ ĐẦU.....	2
CHƯƠNG I: TỔNG QUAN VỀ TẬP DỮ LIỆU.....	3
1.1. Giới thiệu tổng quan về tập dữ liệu	3
1.2. Chi tiết các thuộc tính trong tập dữ liệu	3
1.3. Mục tiêu của báo cáo.....	4
1.4. Đọc dữ liệu	4
CHƯƠNG II: TIỀN XỬ LÝ DỮ LIỆU (EDA)	5
2.1. Kiểm tra kiểu dữ liệu của thuộc tính và dữ liệu bị null.....	5
2.2. Thông kê tổng hợp các biến.....	5
2.3. Phân bố của các giá trị.....	6
2.3.1 Phân bố của các biến liên tục	6
2.3.2. Phân bố của các biến phân loại	7
2.4. Làm sạch dữ liệu	7
2.4.1. Các vấn đề về tính đồng nhất của dữ liệu: Các ràng buộc về tính duy nhất	7
2.4.2. Phát hiện và xử lý ngoại lệ.....	8
2.4.3. Lọc các giá trị biến với DBSCAN.....	10
CHƯƠNG III: TRỰC QUAN HÓA VÀ PHÂN TÍCH DỮ LIỆU.....	12
3.1. Phân tích đơn biến.....	12
3.1.1. Phân bố của khách hàng theo điểm tín dụng.....	12
3.1.2. Sự phân bố của khách hàng theo địa lý.....	13
3.1.3. Sự phân bố khách hàng theo giới tính.....	14
3.1.4. Trực quan hóa dữ liệu theo độ tuổi.....	15
3.1.5. Trực quan thời gian khách hàng đồng hành với ngân hàng theo năm	16
3.1.6. Trực quan theo số dư tài khoản của khách hàng	17

3.1.7. Trực quan theo số lượng sản phẩm mà khách hàng sử dụng.....	18
3.1.8. Phân bổ theo việc sử dụng thẻ tín dụng của khách hàng	18
3.1.9. Phân bố theo khách hàng là thành viên thân thiết.....	19
3.1.10. Phân bố khách hàng theo lương ước tính	20
3.2. Phân tích đa biến	20
3.2.1. Phân phối Điểm tín dụng theo Trạng thái Churn với KDE	20
3.2.2. Phân bố của giới tính của khách hàng theo địa lý với trạng thái Churn	21
3.2.3. Phân bố của độ tuổi khách hàng với trạng thái Churn	22
3.2.4. Phân bố của số năm khách hàng đồng hành cùng ngân hàng với trạng thái Churn.....	23
3.2.5. Phân bố của số dư tài khoản khách hàng với trạng thái Churn.....	24
3.2.6. Phân bố số lượng sản phẩm khách hàng mua với tỉ lệ rời bỏ	25
3.2.7. Trực quan khách hàng có thẻ tín dụng với tỉ lệ Churn.....	26
3.2.8. Trực quan khách hàng thân thiết với tỉ lệ Churn	27
3.2.9. Trực quan tiền lương của khách hàng với tỉ lệ Churn.....	28
3.3. Khai phá luật kết hợp	30
3.3.1. Chuyển đổi các biến liên tục thành các biến phân loại.....	30
3.3.2. Sử dụng thuật toán Apriori để khai phá luật kết hợp	30
3.3.3. Sử dụng thuật toán FP-Growth để khai phá luật kết hợp.....	31
3.3.4. Giải thích các kết quả.....	32
3.3.5. Đánh giá.....	34
3.4. Trích lọc đặc trưng	36
3.4.1. Phân tích mối quan hệ giữa biến liên tục với biến liên tục.....	36
3.4.2 Phân tích mối quan hệ giữa biến phân loại với biến phân loại	37
3.4.3. Phân tích mối quan hệ giữa biến phân loại với biến liên tục	38
3.4.4. Đánh giá mức độ quan trọng của các thuộc tính để dự đoán Exited bằng RandomForest	39

3.4.5. Đánh giá mức độ quan trọng của các thuộc tính để dự đoán Exited bằng LogisticRegression.....	40
CHƯƠNG IV: XÂY DỰNG CÁC MÔ HÌNH.....	41
 4.1. Các mô hình được chọn (tập trung vào sự diễn).....	41
 4.1.1. Mô hình dự đoán Decision Tree	41
 4.1.2. Mô hình dự đoán Logistic Regression.....	51
 4.1.3. Mô hình ANN (kết hợp với SHAP để diễn giải).....	54
 4.2. Nhận xét chung.....	60
CHƯƠNG V: ĐÁNH GIÁ VÀ KẾT LUẬN.....	62
 5.1. Đánh giá.....	62
 5.2. Kết luận	63
 5.2.1. Kết quả đạt được	63
 5.2.2. Những hạn chế	63
TÀI LIỆU THAM KHẢO	64

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin gửi lời cảm ơn chân thành đến Thầy đã hướng dẫn, giúp đỡ và chỉ dẫn chúng em trong quá trình thực hiện đề tài này. Thầy đã tận tình chia sẻ kiến thức và kinh nghiệm của mình, giúp chúng em hiểu sâu hơn về đề tài trong quá trình làm đồ án kết thúc môn. Thầy cũng đã giúp đỡ chúng em giải quyết những khó khăn và trở ngại trong quá trình nghiên cứu, giúp chúng em hoàn thành đề tài này một cách tốt nhất có thể. Không có sự hỗ trợ đó, chúng em không thể hoàn thành đề tài một cách thành công.

Chúng em cũng xin gửi lời cảm ơn đến các thành viên trong nhóm. Mỗi thành viên đã đóng góp một phần công sức, tài năng và thời gian để hoàn thành đề tài này. Chúng em đã học hỏi và trau dồi thêm nhiều kỹ năng, kiến thức mới từ nhau. Đặc biệt, chúng em cảm ơn vì mối quan hệ tình cảm và sự đoàn kết của nhóm trong suốt quá trình làm việc.

Một lần nữa, chúng em xin chân thành cảm ơn tất cả những người đã giúp đỡ chúng em trong quá trình nghiên cứu và thực hiện đề tài này. Sự hỗ trợ của các bạn đã làm cho đề tài của chúng em trở nên hoàn hảo hơn.

LỜI MỞ ĐẦU

Trong bối cảnh ngành ngân hàng hiện nay, việc duy trì mối quan hệ với khách hàng và giảm thiểu tỷ lệ rời bỏ (churn) trở thành một thách thức lớn. Khách hàng rời bỏ không chỉ gây ra thiệt hại về doanh thu mà còn ảnh hưởng tiêu cực đến danh tiếng và sự phát triển bền vững của ngân hàng. Vì vậy, việc phân tích và dự đoán hành vi rời bỏ của khách hàng là cực kỳ quan trọng, giúp ngân hàng xây dựng các chiến lược hiệu quả để giữ chân khách hàng và cải thiện trải nghiệm người dùng.

Với mục tiêu khám phá và phân tích các yếu tố ảnh hưởng đến việc khách hàng rời bỏ, cũng như xây dựng các mô hình dự đoán để xác định khách hàng có nguy cơ rời bỏ, chúng em đã tiến hành một loạt các phân tích dữ liệu và áp dụng các kỹ thuật học máy trên tập dữ liệu này. Việc hiểu rõ các yếu tố then chốt dẫn đến hành vi rời bỏ sẽ giúp ngân hàng đưa ra những quyết định chiến lược nhằm cải thiện dịch vụ, nâng cao sự hài lòng của khách hàng và giảm thiểu tỷ lệ rời bỏ.

Báo cáo này sẽ trình bày chi tiết quá trình phân tích dữ liệu, từ việc khám phá các đặc trưng quan trọng đến việc xây dựng và đánh giá các mô hình dự đoán. Kết quả thu được không chỉ mang lại những hiểu biết sâu sắc về hành vi của khách hàng mà còn cung cấp các đề xuất thực tiễn để ngân hàng có thể cải thiện chất lượng dịch vụ và tăng cường sự gắn bó của khách hàng. Chúng em hy vọng rằng những phân tích và mô hình này sẽ trở thành công cụ hữu ích cho các nhà quản lý ngân hàng trong việc phát triển các chiến lược giữ chân khách hàng hiệu quả.

CHƯƠNG I: TỔNG QUAN VỀ TẬP DỮ LIỆU

1.1. Giới thiệu tổng quan về tập dữ liệu

Tập dữ liệu này chứa thông tin về khách hàng ngân hàng và tình trạng rời bỏ (churn) của họ, cho biết liệu họ đã rời khỏi ngân hàng hay chưa. Tập dữ liệu này phù hợp để khám phá và phân tích các yếu tố ảnh hưởng đến việc khách hàng rời bỏ tại các tổ chức ngân hàng và để xây dựng các mô hình dự đoán nhằm xác định khách hàng có nguy cơ rời bỏ.

1.2. Chi tiết các thuộc tính trong tập dữ liệu

Thuộc tính	Mô tả
Id	Số thứ tự được gán cho mỗi hàng trong tập dữ liệu
CustomerId	Một định danh duy nhất cho mỗi khách hàng
Surname	Họ của khách hàng
CreditScore	Điểm tín dụng của khách hàng
Geography	Vị trí địa lý của khách hàng (ví dụ: quốc gia hoặc khu vực)
Gender	Giới tính của khách hàng
Age	Tuổi của khách hàng
Tenure	Số năm khách hàng đã gắn bó với ngân hàng

Balance	Số dư tài khoản của khách hàng
NumOfProducts	Số lượng sản phẩm ngân hàng mà khách hàng sử dụng
HasCrCard	Cho biết liệu khách hàng có thẻ tín dụng hay không (nhị phân: có/không)
IsActiveMember	Cho biết liệu khách hàng là thành viên đang hoạt động hay không (nhị phân: có/không)
EstimatedSalary	Lương ước tính của khách hàng
Exited	Cho biết liệu khách hàng đã rời khỏi ngân hàng hay không (nhị phân: có/không)

1.3. Mục tiêu của báo cáo

Mục tiêu của báo cáo này là sử dụng để phân tích dữ liệu khám phá nhằm hiểu các yếu tố ảnh hưởng đến việc khách hàng rời bỏ ngân hàng, từ đó xây dựng các mô hình máy học để dự đoán việc khách hàng rời bỏ dựa trên các đặc trưng đã cho.

1.4. Đọc dữ liệu

```
data = pd.read_csv('playground-series-s4e1/train.csv')
data.head()
```

Python

	id	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	0	15674932	Okwudilichukwu	668	France	Male	33.0	3	0.00	2	1.0	0.0	181449.97	0
1	1	15749177	Okwudiliolisa	627	France	Male	33.0	1	0.00	2	1.0	1.0	49503.50	0
2	2	15694510	Hsueh	678	France	Male	40.0	10	0.00	2	1.0	0.0	184866.69	0
3	3	15741417	Kao	581	France	Male	34.0	2	148882.54	1	1.0	1.0	84560.88	0
4	4	15766172	Chiemenam	716	Spain	Male	33.0	5	0.00	2	1.0	1.0	15068.83	0

CHƯƠNG II: TIỀN XỬ LÝ DỮ LIỆU (EDA)

2.1. Kiểm tra kiểu dữ liệu của thuộc tính và dữ liệu bị null

data.info()		data.isna().sum()	
<pre><class 'pandas.core.frame.DataFrame'> RangeIndex: 165034 entries, 0 to 165033 Data columns (total 14 columns): # Column Non-Null Count Dtype --- 0 id 165034 non-null int64 1 CustomerId 165034 non-null int64 2 Surname 165034 non-null object 3 CreditScore 165034 non-null int64 4 Geography 165034 non-null object 5 Gender 165034 non-null object 6 Age 165034 non-null float64 7 Tenure 165034 non-null int64 8 Balance 165034 non-null float64 9 NumOfProducts 165034 non-null int64 10 HasCrCard 165034 non-null float64 11 IsActiveMember 165034 non-null float64 12 EstimatedSalary 165034 non-null float64 13 Exited 165034 non-null int64 dtypes: float64(5), int64(6), object(3) memory usage: 17.6+ MB</pre>		<pre>id 0 CustomerId 0 Surname 0 CreditScore 0 Geography 0 Gender 0 Age 0 Tenure 0 Balance 0 NumOfProducts 0 HasCrCard 0 IsActiveMember 0 EstimatedSalary 0 Exited 0 dtype: int64</pre>	

2.2. Thống kê tổng hợp các biến

data.describe()													Python
	id	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited		
count	165034.0000	1.650340e+05	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	
mean	82516.5000	1.569201e+07	656.454373	38.125888	5.020353	55478.086689	1.554455	0.753954	0.497770	112574.822734	0.211599		
std	47641.3565	7.139782e+04	80.103340	8.867205	2.806159	62817.663278	0.547154	0.430707	0.499997	50292.865585	0.408443		
min	0.0000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.000000	0.000000	11.580000	0.000000		
25%	41258.2500	1.563314e+07	597.000000	32.000000	3.000000	0.000000	1.000000	1.000000	0.000000	74637.570000	0.000000		
50%	82516.5000	1.569017e+07	659.000000	37.000000	5.000000	0.000000	2.000000	1.000000	0.000000	117948.000000	0.000000		
75%	123774.7500	1.575682e+07	710.000000	42.000000	7.000000	119939.517500	2.000000	1.000000	1.000000	155152.467500	0.000000		
max	165033.0000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.000000	1.000000	199992.480000	1.000000		

2.3. Phân bố của các giá trị

2.3.1 Phân bố của các biến liên tục

```
plt.figure(figsize=(15, 8))
plt.suptitle('Distribution of continuous variable', fontsize=20)

plt.subplot(3, 2, 1)
sns.histplot(data['CreditScore'], kde=True)
plt.xlabel('Credit Score variable', fontsize=15)

plt.subplot(3, 2, 2)
sns.histplot(data['Age'], kde=True)
plt.xlabel('Age variable', fontsize=15)

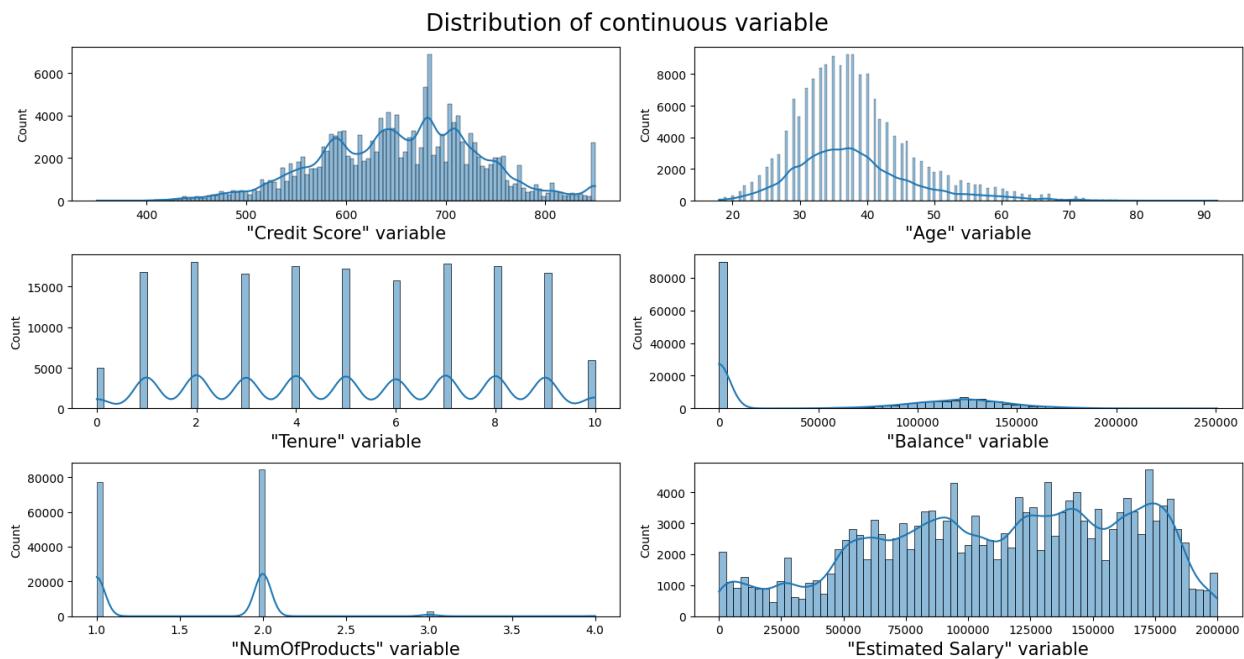
plt.subplot(3, 2, 3)
sns.histplot(data['Tenure'], kde=True)
plt.xlabel('Tenure variable', fontsize=15)

plt.subplot(3, 2, 4)
sns.histplot(data['Balance'], kde=True)
plt.xlabel('Balance variable', fontsize=15)

plt.subplot(3, 2, 5)
sns.histplot(data['NumOfProducts'], kde=True)
plt.xlabel('NumOfProducts variable', fontsize=15)

plt.subplot(3, 2, 6)
sns.histplot(data['EstimatedSalary'], kde=True)
plt.xlabel('Estimated Salary variable', fontsize=15)

plt.tight_layout()
plt.show()
```



2.3.2. Phân bố của các biến phân loại

```
fig, ax = plt.subplots(2, 3)
fig.set_size_inches(15, 6)
fig.suptitle('Distribution of categorical variable', fontsize=20)

sns.countplot(data=data[['Geography']], x='Geography', ax=ax[0, 0])

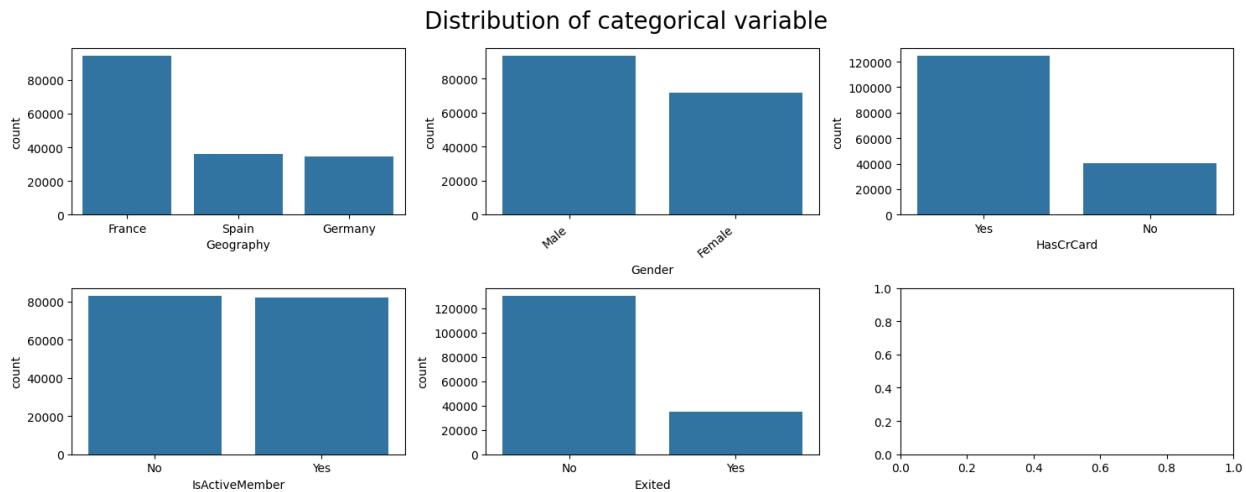
plot = sns.countplot(x='Gender', data=data[['Gender']], ax=ax[0, 1])
plot.set_xticklabels(plot.get_xticklabels(), rotation=40, ha="right")

sns.countplot(x='HasCrCard', data=data[['HasCrCard']].replace(yes_no_dict), ax=ax[0, 2])

sns.countplot(x='IsActiveMember', data=data[['IsActiveMember']].replace(yes_no_dict), ax=ax[1, 0])

sns.countplot(x='Exited', data=data[['Exited']].replace(yes_no_dict), ax=ax[1, 1])

fig.tight_layout()
plt.show()
```



2.4. Làm sạch dữ liệu

2.4.1. Các vấn đề về tính đồng nhất của dữ liệu: Các ràng buộc về tính duy nhất

Chúng tôi sẽ bỏ cột 'id', 'CustomerId', 'Surname' vì nó không có ý nghĩa nhiều khi phân tích hoặc lập mô hình.

```
new_data = data.drop(columns=['id', 'CustomerId', 'Surname'])
new_data.shape
```

(165034, 11)

Kiểm tra và xử lý trùng lặp

```
new_data.duplicated().sum()
```

123

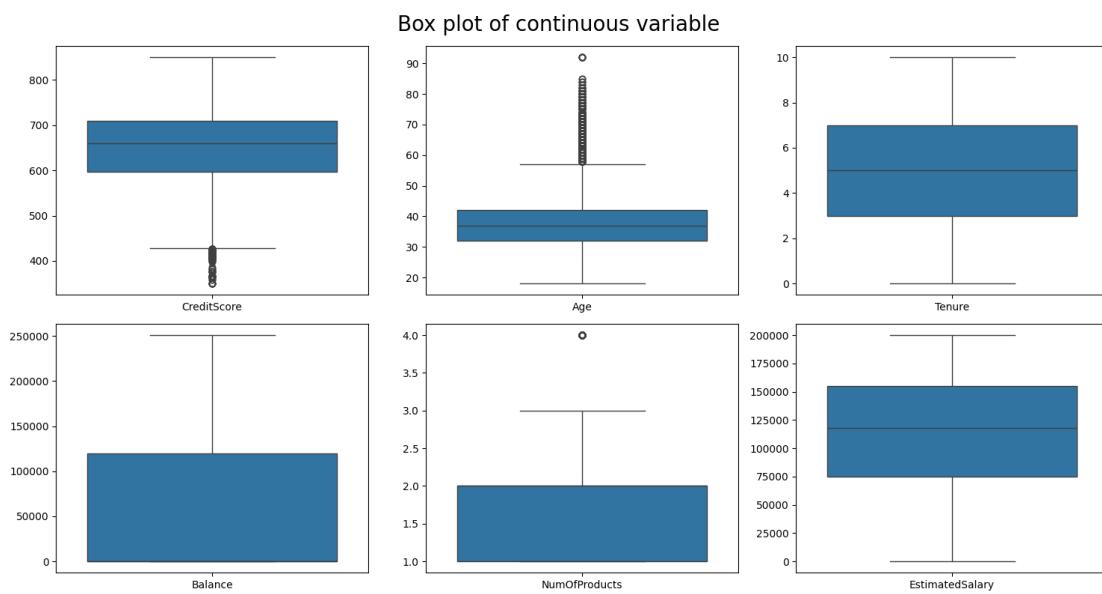
```
new_data = new_data.drop_duplicates()  
new_data.duplicated().sum()
```

0

2.4.2. Phát hiện và xử lý ngoại lệ

Trực quan các ngoại lệ trong biến liên tục

```
plt.figure(figsize=(15, 8))  
plt.suptitle('Box plot of continuous variable', fontsize=20)  
  
plt.subplot(2, 3, 1)  
sns.boxplot(new_data[['CreditScore']])  
  
plt.subplot(2, 3, 2)  
sns.boxplot(new_data[['Age']])  
  
plt.subplot(2, 3, 3)  
sns.boxplot(new_data[['Tenure']])  
  
plt.subplot(2, 3, 4)  
sns.boxplot(new_data[['Balance']])  
  
plt.subplot(2, 3, 5)  
sns.boxplot(new_data[['NumOfProducts']])  
  
plt.subplot(2, 3, 6)  
sns.boxplot(new_data[['EstimatedSalary']])  
  
plt.tight_layout()  
plt.show()
```



Xác định và loại bỏ các giá trị ngoại lai (outliers) trong các cột

```
columns = ['CreditScore', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'EstimatedSalary']
```

```
for col in columns:  
    print('*'*30)  
    print("Column:", col)  
    print("data before:", new_data.shape)  
    Q1 = np.percentile(new_data[col], 25)  
    Q3 = np.percentile(new_data[col], 75)  
    iqr = scipy.stats.iqr(new_data[col])  
    No_upper = new_data[new_data[col] > (Q3 + 1.5 * iqr)].shape[0]  
    No_lower = new_data[new_data[col] < (Q1 - 1.5 * iqr)].shape[0]  
    print(Q1 - 1.5 * iqr, Q3 + 1.5 * iqr, No_upper, No_lower,  
          | (No_lower + No_upper) / new_data.shape[0])  
    new_data = new_data[new_data[col] <= (Q3 + 1.5 * iqr)]  
    new_data = new_data[new_data[col] >= (Q1 - 1.5 * iqr)]  
    print("data after:", new_data.shape)
```

```
for col in columns:  
    print('*'*30)  
    print("Column:", col)  
    print("data before:", new_data.shape)  
    Q1 = np.percentile(new_data[col], 25)  
    Q3 = np.percentile(new_data[col], 75)  
    iqr = scipy.stats.iqr(new_data[col])  
    No_upper = new_data[new_data[col] > (Q3 + 1.5 * iqr)].shape[0]  
    No_lower = new_data[new_data[col] < (Q1 - 1.5 * iqr)].shape[0]  
    print(Q1 - 1.5 * iqr, Q3 + 1.5 * iqr, No_upper, No_lower,  
          | (No_lower + No_upper) / new_data.shape[0])  
    new_data = new_data[new_data[col] <= (Q3 + 1.5 * iqr)]  
    new_data = new_data[new_data[col] >= (Q1 - 1.5 * iqr)]  
    print("data after:", new_data.shape)
```

```
*****  
Column: CreditScore  
data before: (164911, 11)  
427.5 879.5 0 252 0.0015280969735190497  
data after: (164659, 11)  
*****  
Column: Age  
data before: (164659, 11)  
17.0 57.0 6369 0 0.03867993853964861  
data after: (158290, 11)  
*****  
Column: Tenure  
data before: (158290, 11)  
-3.0 13.0 0 0 0.0  
data after: (158290, 11)  
*****  
Column: Balance  
data before: (158290, 11)  
-179698.8975 299498.1625 0 0 0.0  
data after: (158290, 11)  
*****  
Column: NumOfProducts  
data before: (158290, 11)  
-0.5 3.5 434 0 0.002741803019773833  
data after: (157856, 11)  
...  
Column: EstimatedSalary  
data before: (157856, 11)  
-45705.99374999998 275777.27625 0 0 0.0  
data after: (157856, 11)  
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings..
```

Kiểm tra lại ngoại lệ

```
plt.figure(figsize=(15, 8))
plt.suptitle('Box plot of continuous variable', fontsize=20)

plt.subplot(2, 3, 1)
sns.boxplot(new_data[['CreditScore']])

plt.subplot(2, 3, 2)
sns.boxplot(new_data[['Age']])

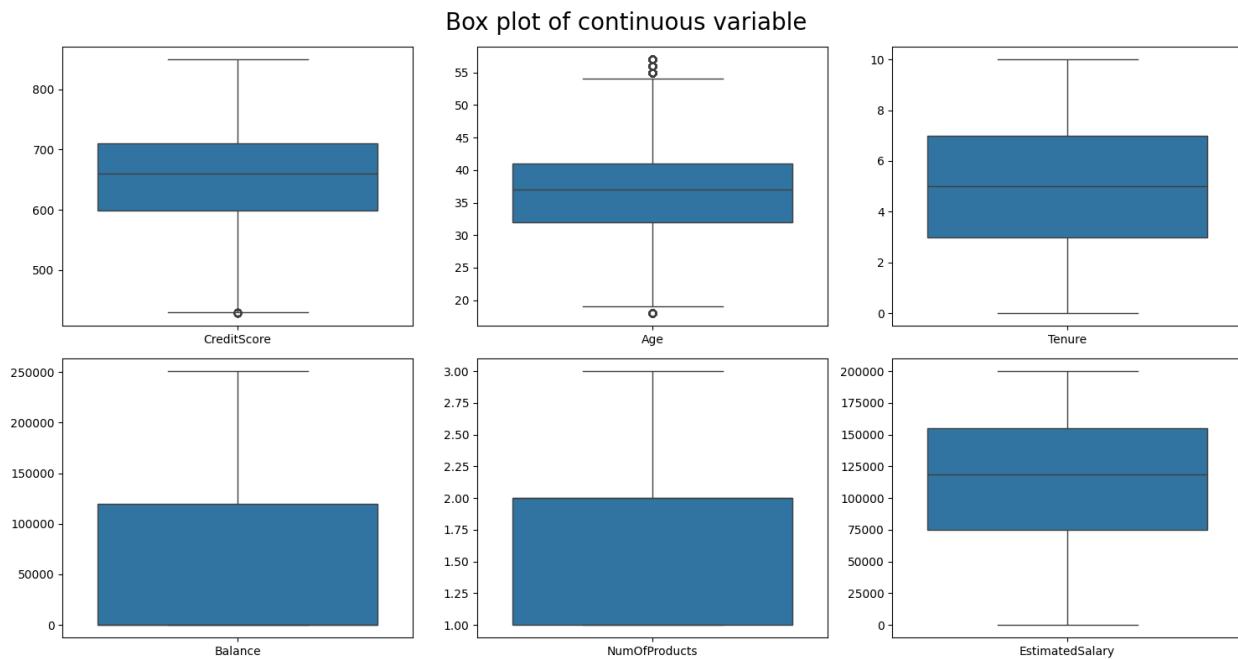
plt.subplot(2, 3, 3)
sns.boxplot(new_data[['Tenure']])

plt.subplot(2, 3, 4)
sns.boxplot(new_data[['Balance']])

plt.subplot(2, 3, 5)
sns.boxplot(new_data[['NumOfProducts']])

plt.subplot(2, 3, 6)
sns.boxplot(new_data[['EstimatedSalary']])

plt.tight_layout()
plt.show()
```



2.4.3. Lọc các giá trị biên với DBSCAN

Kiểm tra lại các cột và số dòng của data mới

```
new_data.shape  
7]
```

```
(157856, 11)
```

```
new_data.columns  
3]  
  
Index(['CreditScore', 'Geography', 'Gender', 'Age', 'Tenure', 'Balance',  
       'NumOfProducts', 'HasCrCard', 'IsActiveMember', 'EstimatedSalary',  
       'Exited'],  
      dtype='object')
```

CHƯƠNG III: TRỰC QUAN HÓA VÀ PHÂN TÍCH DỮ LIỆU

3.1. Phân tích đơn biến

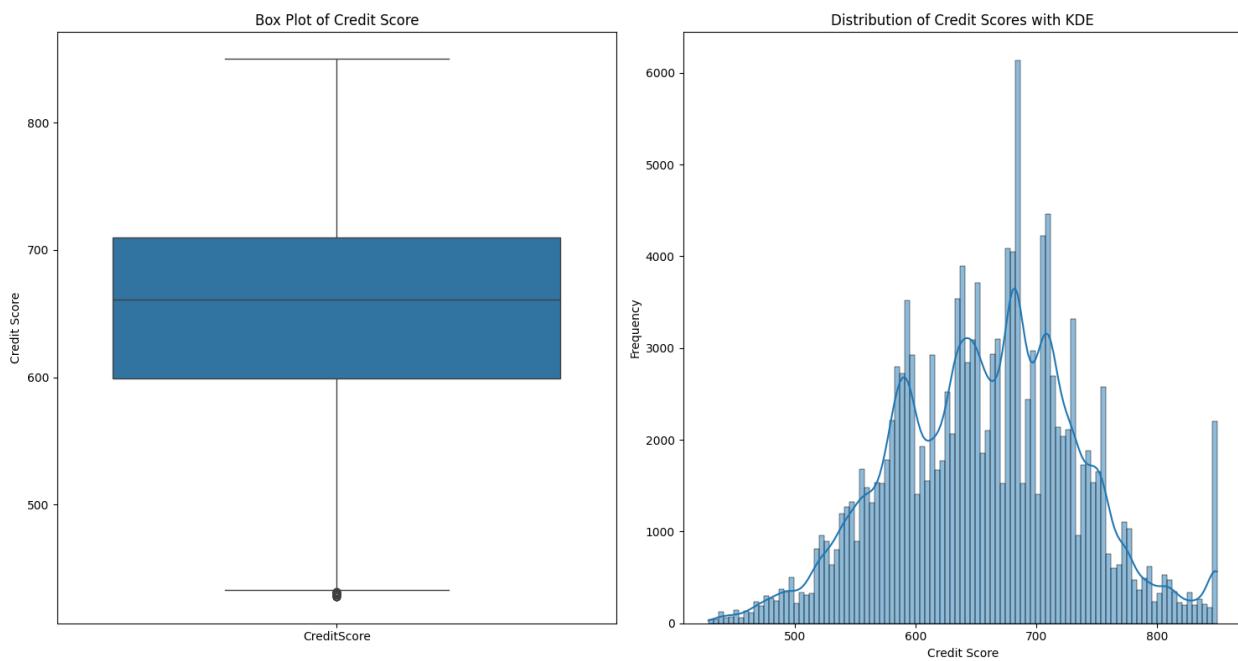
3.1.1. Phân bố của khách hàng theo điểm tín dụng

```
plt.figure(figsize=(15, 8))

plt.subplot(1, 2, 1)
sns.boxplot(new_data[['CreditScore']])
plt.title('Box Plot of Credit Score')
plt.ylabel('Credit Score')

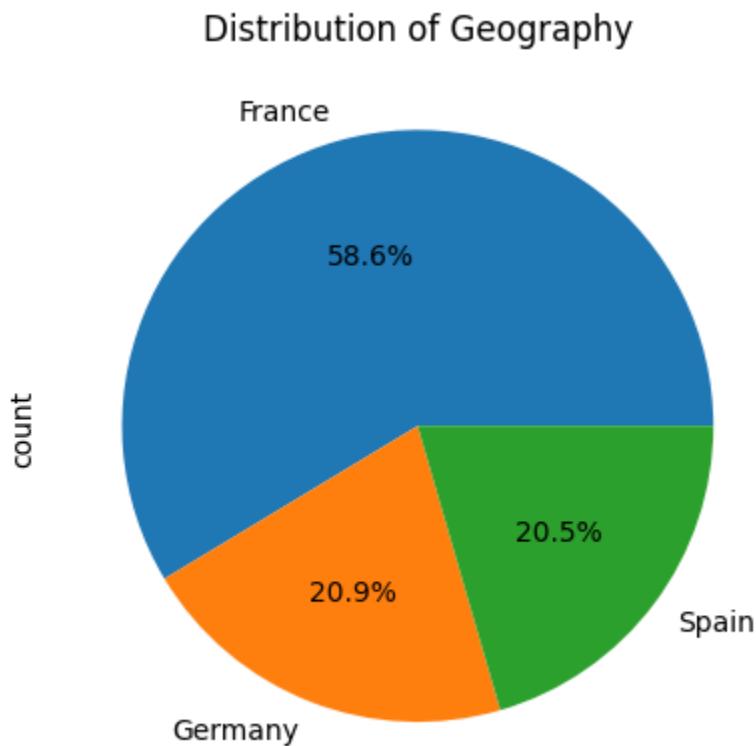
plt.subplot(1, 2, 2)
sns.histplot(new_data['CreditScore'], kde=True)
plt.title('Distribution of Credit Scores with KDE')
plt.xlabel('Credit Score')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```



3.1.2. Sự phân bố của khách hàng theo địa lý

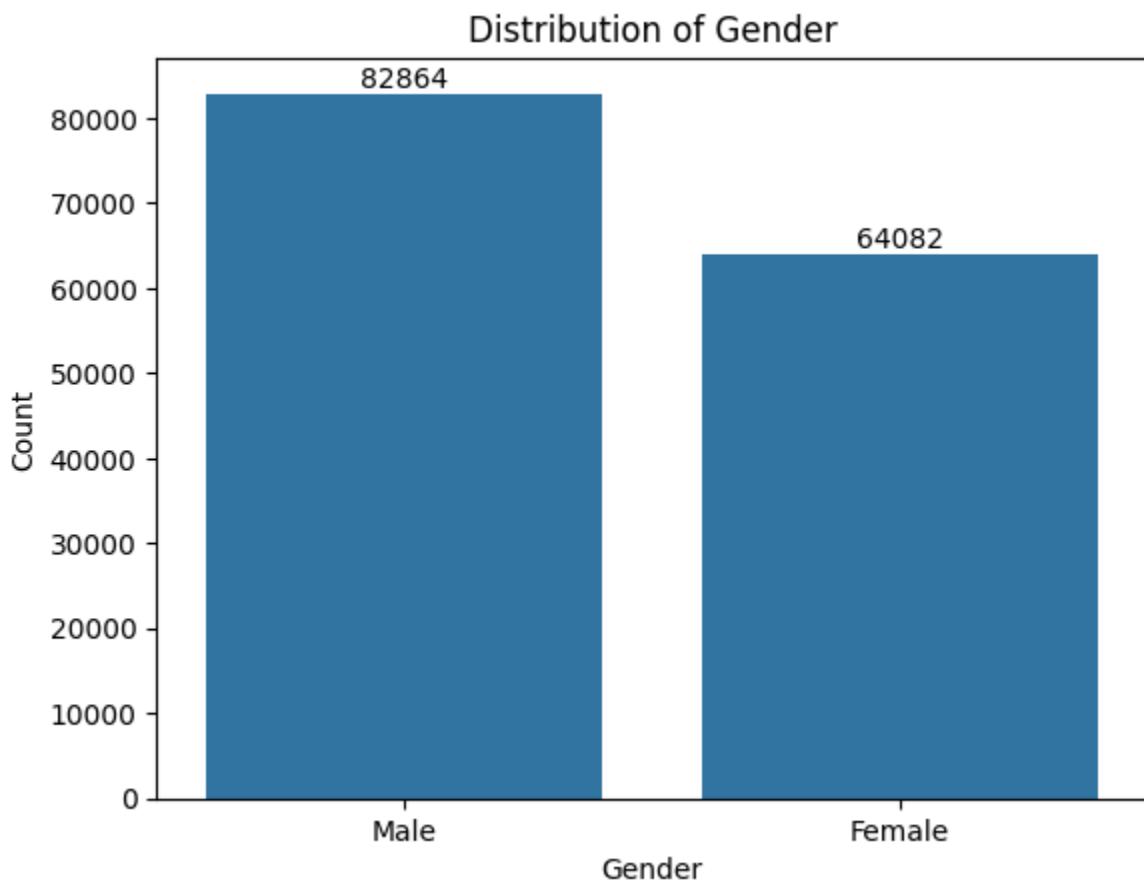
```
value_counts = new_data['Geography'].value_counts()
value_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Distribution of Geography')
plt.show()
```



Nhận xét: Số lượng khách hàng ở France (Pháp) chiếm tỷ lệ nhiều nhất với 58.6%.

3.1.3. Sự phân bố khách hàng theo giới tính

```
a = sns.countplot(data=new_data, x="Gender")
plt.title('Distribution of Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
for bars in a.containers:
    a.bar_label(bars)
plt.show()
```



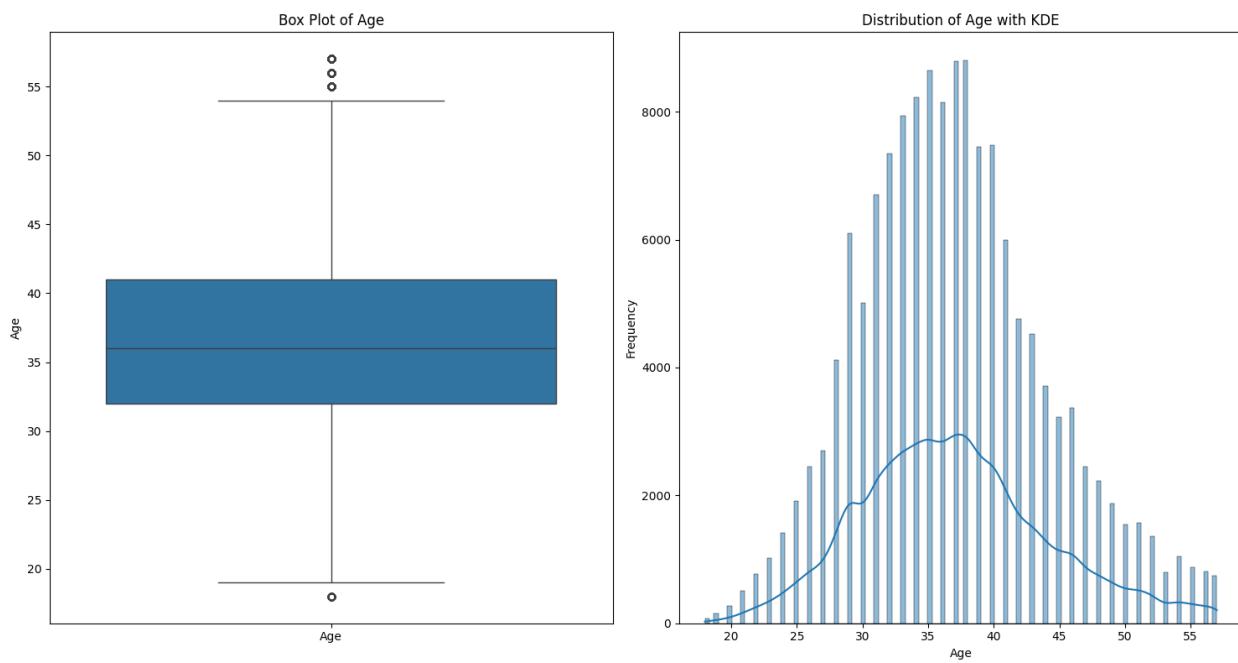
3.1.4. Trực quan hóa dữ liệu theo độ tuổi

```
plt.figure(figsize=(15, 8))

plt.subplot(1, 2, 1)
sns.boxplot(new_data[['Age']])
plt.title('Box Plot of Age')
plt.ylabel('Age')

plt.subplot(1, 2, 2)
sns.histplot(new_data['Age'], kde=True)
plt.title('Distribution of Age with KDE')
plt.xlabel('Age')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

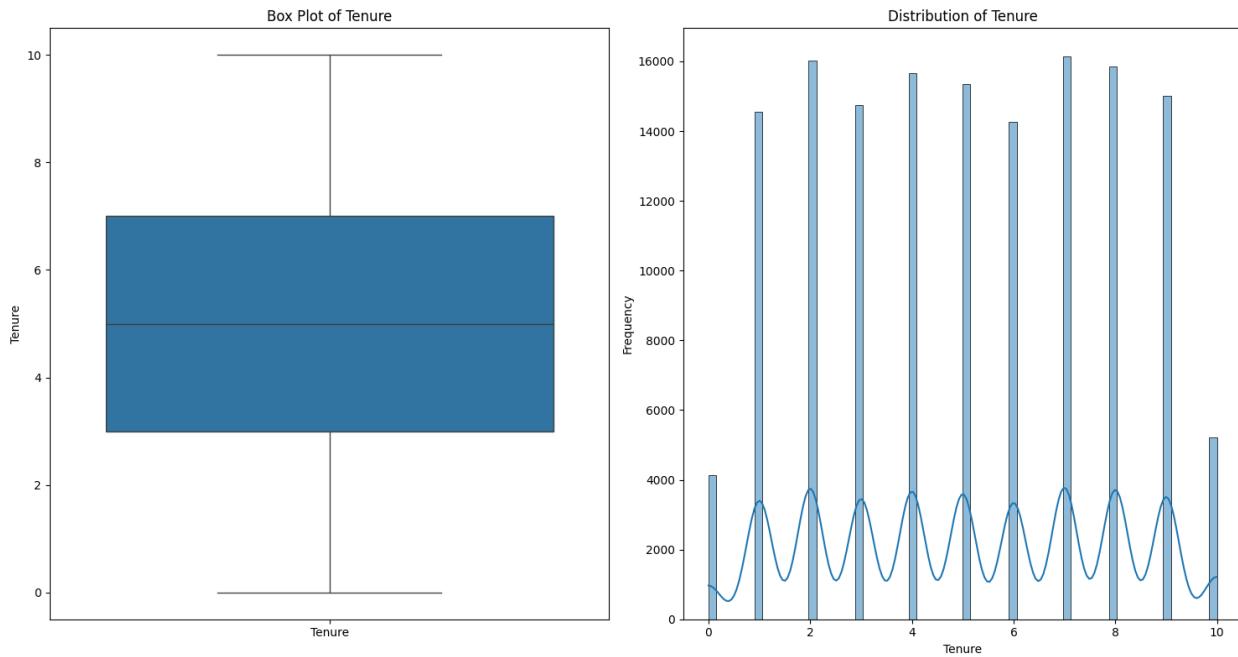


3.1.5. Trực quan thời gian khách hàng đồng hành với ngân hàng theo năm

```
plt.figure(figsize=(15, 8))

plt.subplot(1, 2, 1)
sns.boxplot(new_data[['Tenure']])
plt.title('Box Plot of Tenure')
plt.ylabel('Tenure')

plt.subplot(1, 2, 2)
sns.histplot(new_data['Tenure'], kde=True)
plt.title('Distribution of Tenure')
plt.xlabel('Tenure')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

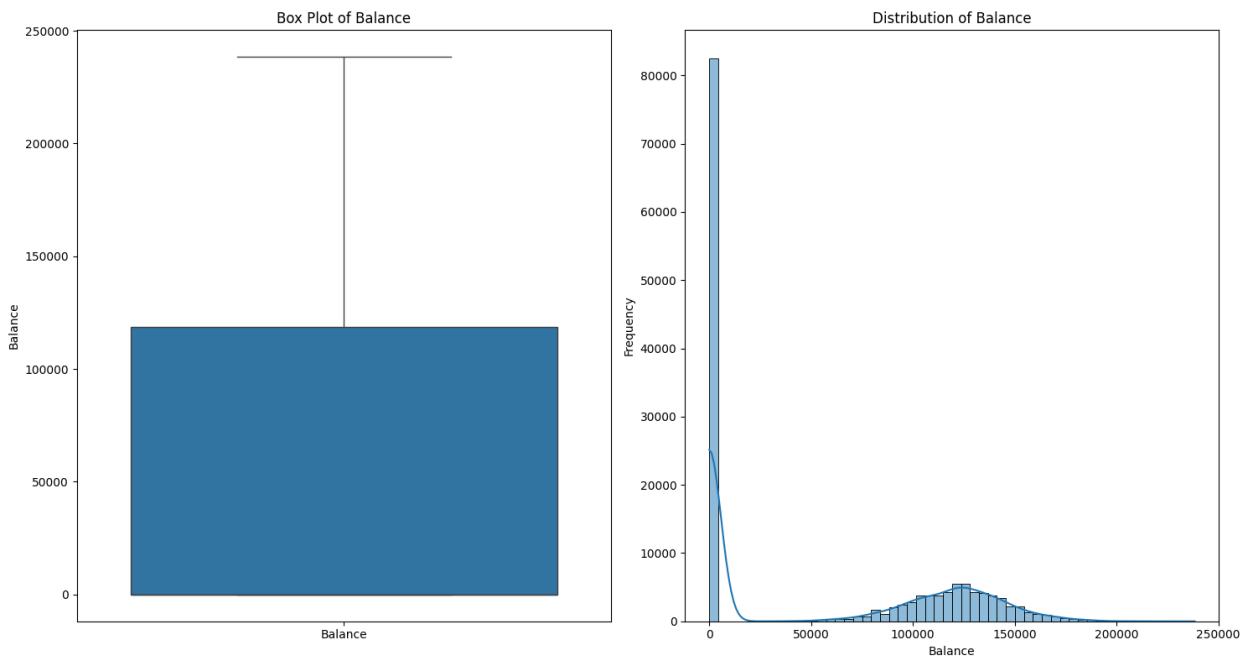


3.1.6. Trực quan theo số dư tài khoản của khách hàng

```
plt.figure(figsize=(15, 8))

plt.subplot(1, 2, 1)
sns.boxplot(new_data[['Balance']])
plt.title('Box Plot of Balance')
plt.ylabel('Balance')

plt.subplot(1, 2, 2)
sns.histplot(new_data['Balance'], kde=True)
plt.title('Distribution of Balance')
plt.xlabel('Balance')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```

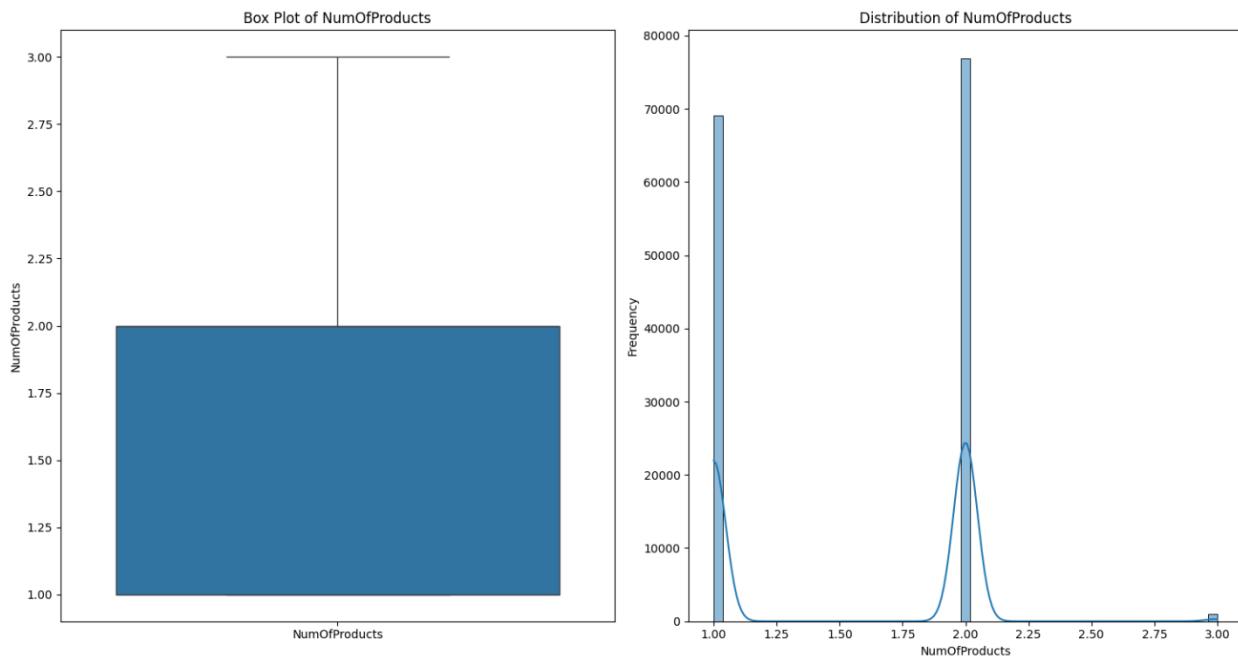


3.1.7. Trực quan theo số lượng sản phẩm mà khách hàng sử dụng

```
plt.figure(figsize=(15, 8))

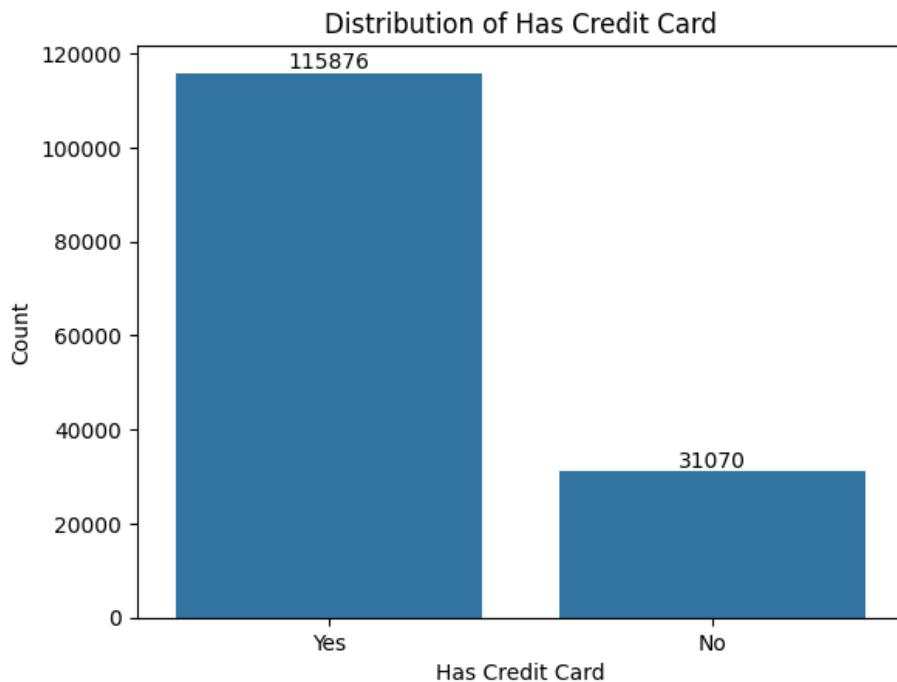
plt.subplot(1, 2, 1)
sns.boxplot(new_data[['NumOfProducts']])
plt.title('Box Plot of NumOfProducts')
plt.ylabel('NumOfProducts')

plt.subplot(1, 2, 2)
sns.histplot(new_data['NumOfProducts'], kde=True)
plt.title('Distribution of NumOfProducts')
plt.xlabel('NumOfProducts')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



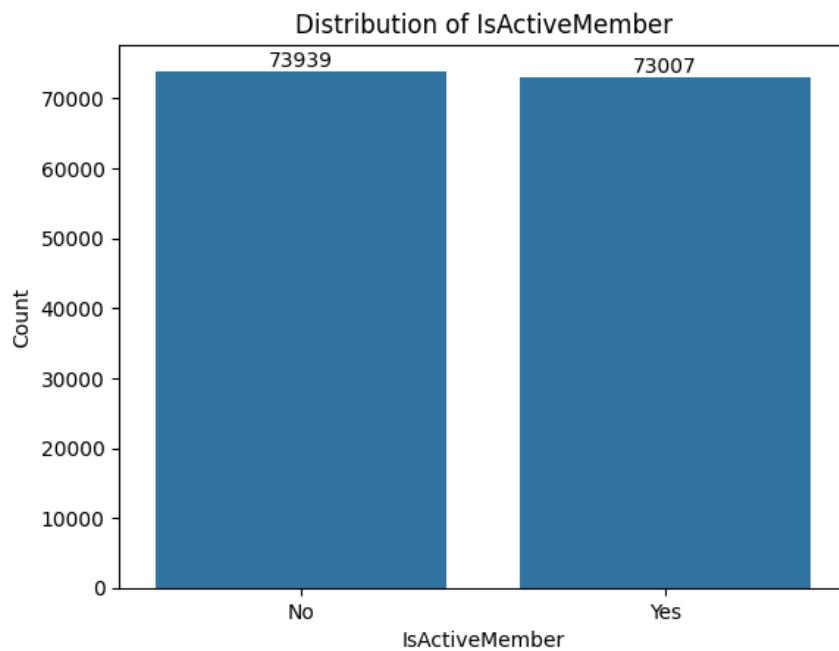
3.1.8. Phân bố theo việc sử dụng thẻ tín dụng của khách hàng

```
a = sns.countplot(x=new_data['HasCrCard'].replace(yes_no_dict))
plt.title('Distribution of Has Credit Card')
plt.xlabel('Has Credit Card')
plt.ylabel('Count')
for bars in a.containers:
    a.bar_label(bars)
plt.show()
```



3.1.9. Phân bố theo khách hàng là thành viên thân thiết

```
a = sns.countplot(x=new_data['IsActiveMember'].replace(yes_no_dict))
plt.title('Distribution of IsActiveMember')
plt.xlabel('IsActiveMember')
plt.ylabel('Count')
for bars in a.containers:
    a.bar_label(bars)
plt.show()
```

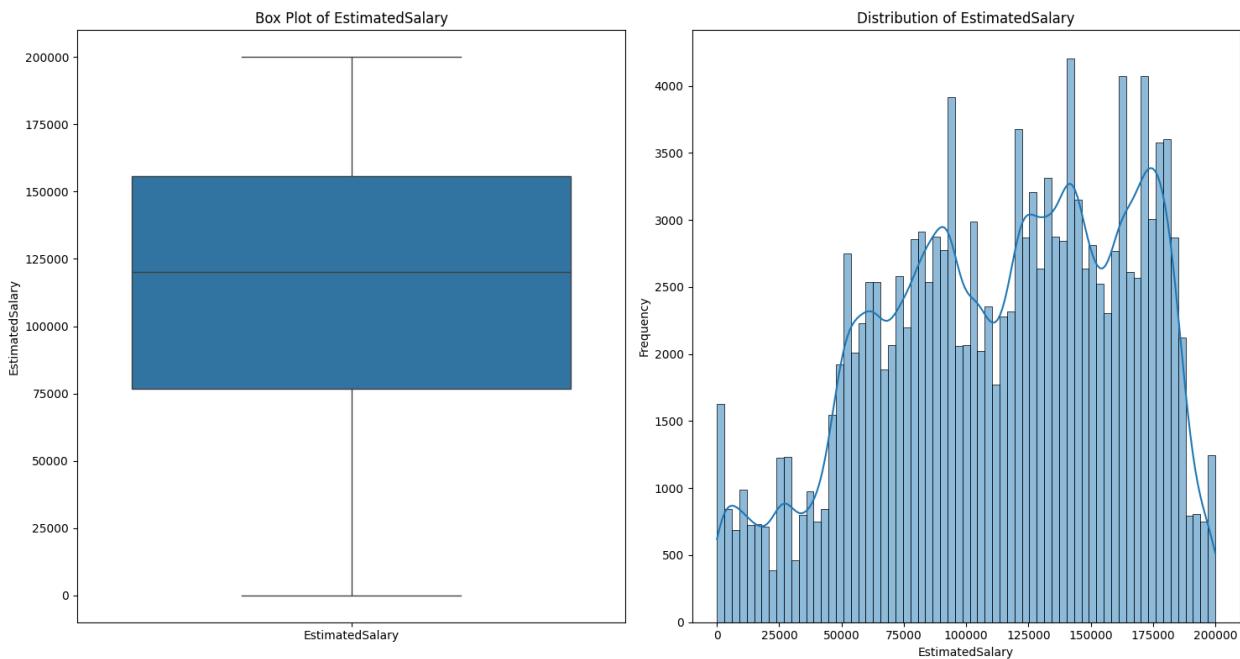


3.1.10. Phân bố khách hàng theo lương ước tính

```
plt.figure(figsize=(15, 8))

plt.subplot(1, 2, 1)
sns.boxplot(new_data[['EstimatedSalary']])
plt.title('Box Plot of EstimatedSalary')
plt.ylabel('EstimatedSalary')

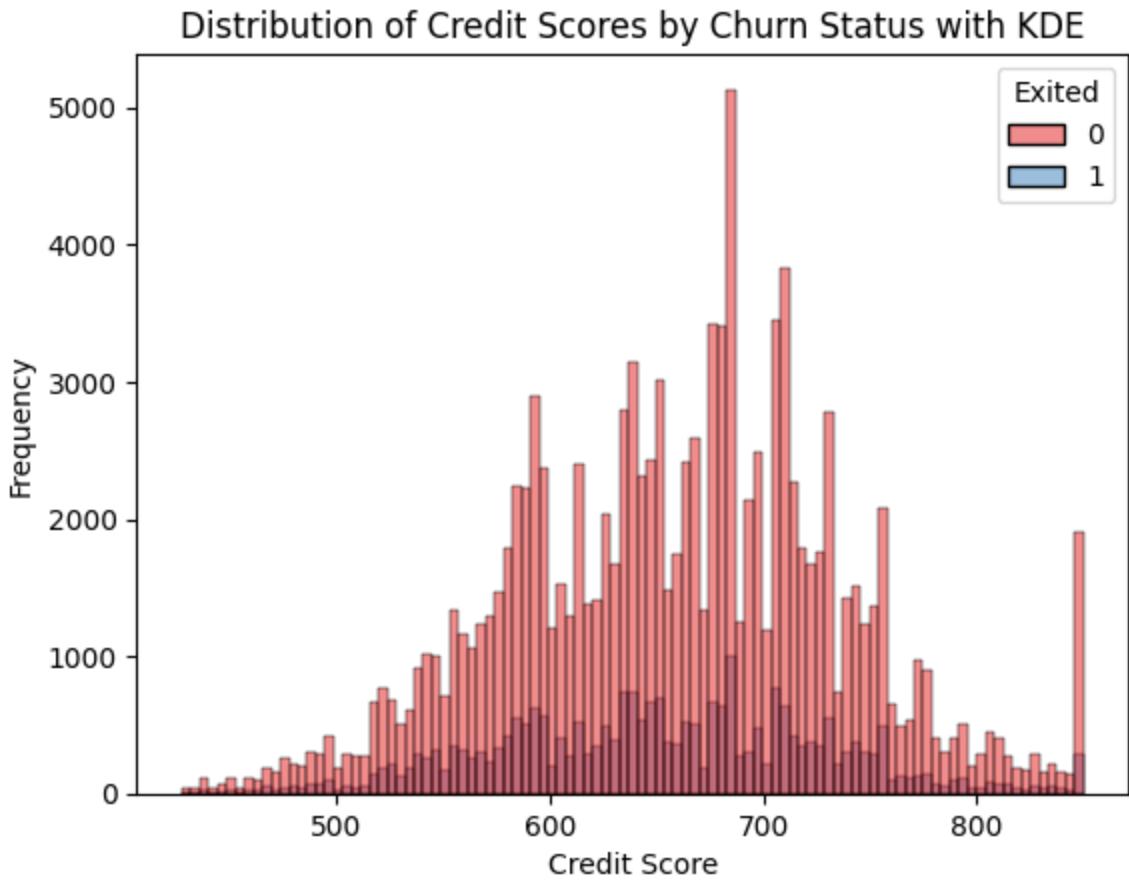
plt.subplot(1, 2, 2)
sns.histplot(new_data['EstimatedSalary'], kde=True)
plt.title('Distribution of EstimatedSalary')
plt.xlabel('EstimatedSalary')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



3.2. Phân tích đa biến

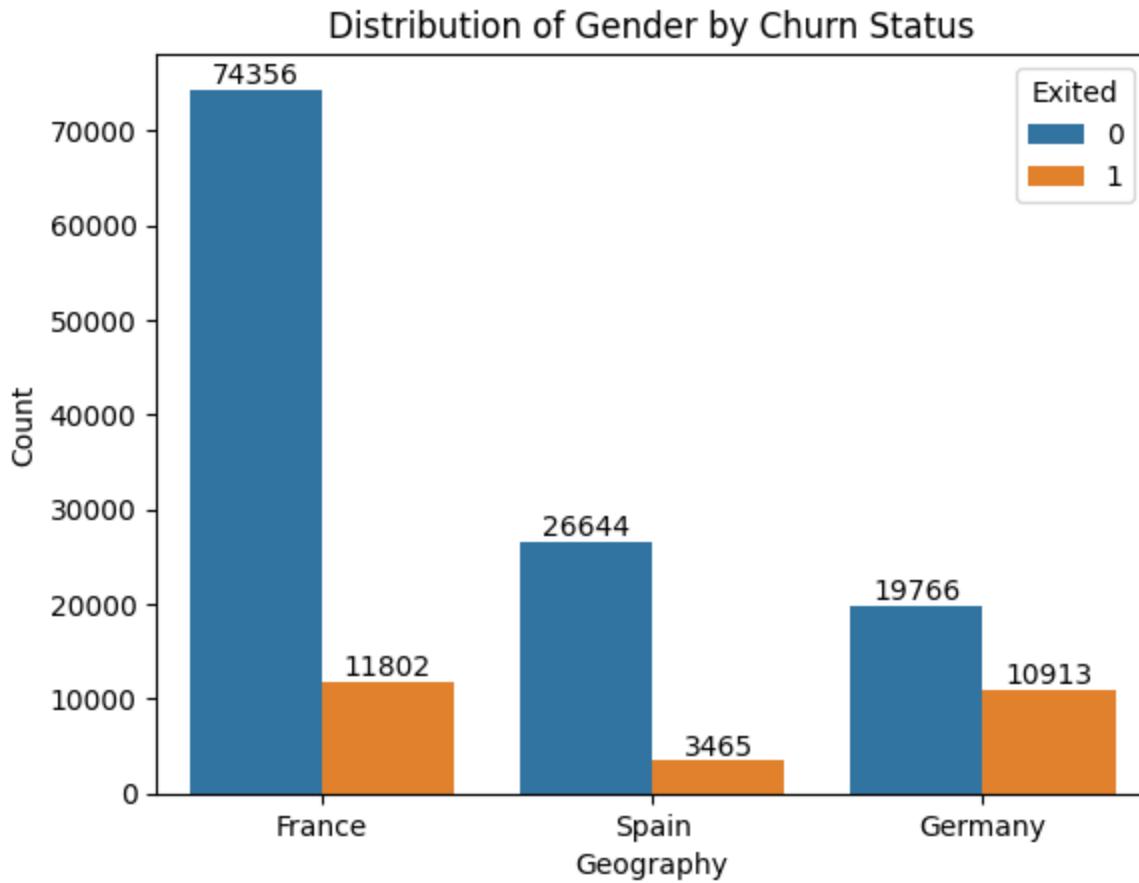
3.2.1. Phân phối Điểm tín dụng theo Trạng thái Churn với KDE

```
sns.histplot(data=new_data, x='CreditScore', hue='Exited', palette='Set1', kde=False)
plt.title('Distribution of Credit Scores by Churn Status with KDE')
plt.xlabel('Credit Score')
plt.ylabel('Frequency')
plt.show()
```



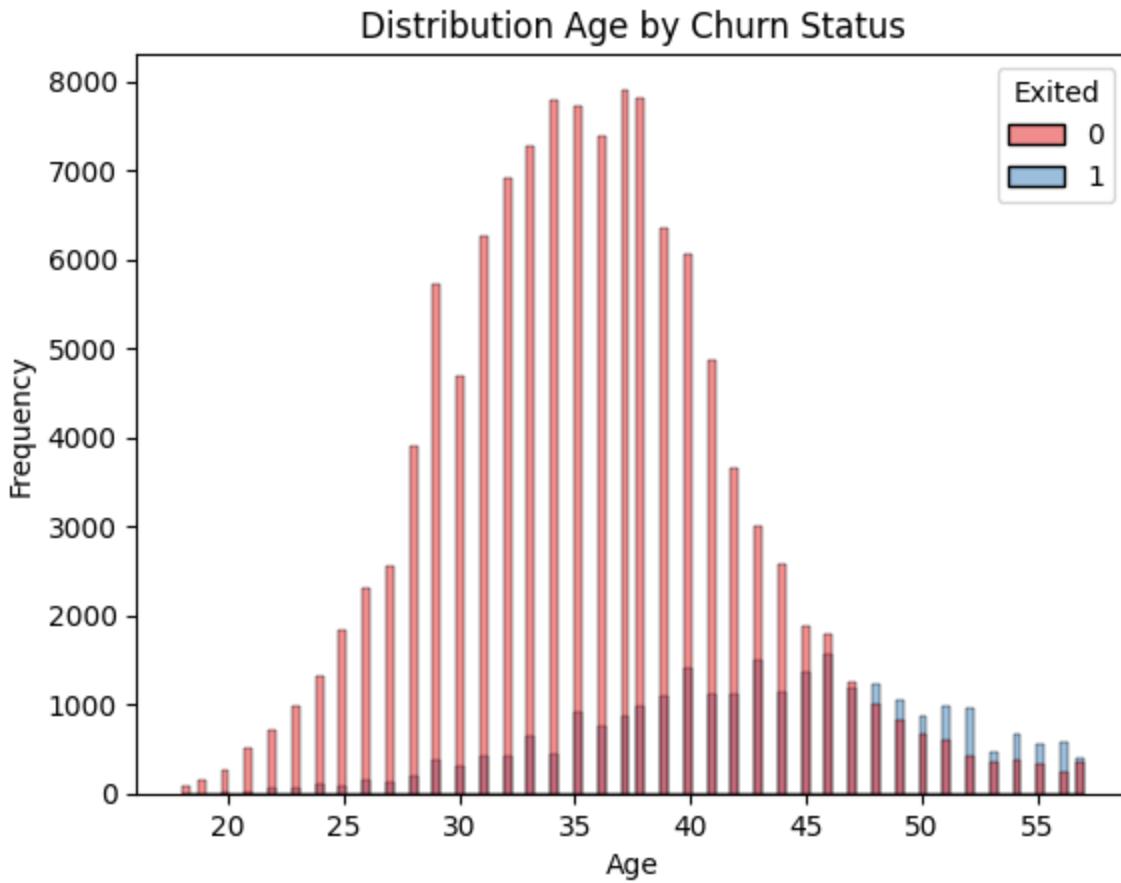
3.2.2. Phân bố của giới tính của khách hàng theo địa lý với trạng thái Churn

```
a= sns.countplot(
    data=new_data,
    x="Geography",
    hue="Exited"
)
plt.title('Distribution of Geography by Churn Status')
plt.xlabel('Geography')
plt.ylabel('Count')
for bars in a.containers:
    a.bar_label(bars)
plt.show()
```



3.2.3. Phân bố của độ tuổi khách hàng với trạng thái Churn

```
sns.histplot(data=new_data, x='Age', hue='Exited', palette='Set1', kde=False)
plt.title('Distribution Age by Churn Status')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



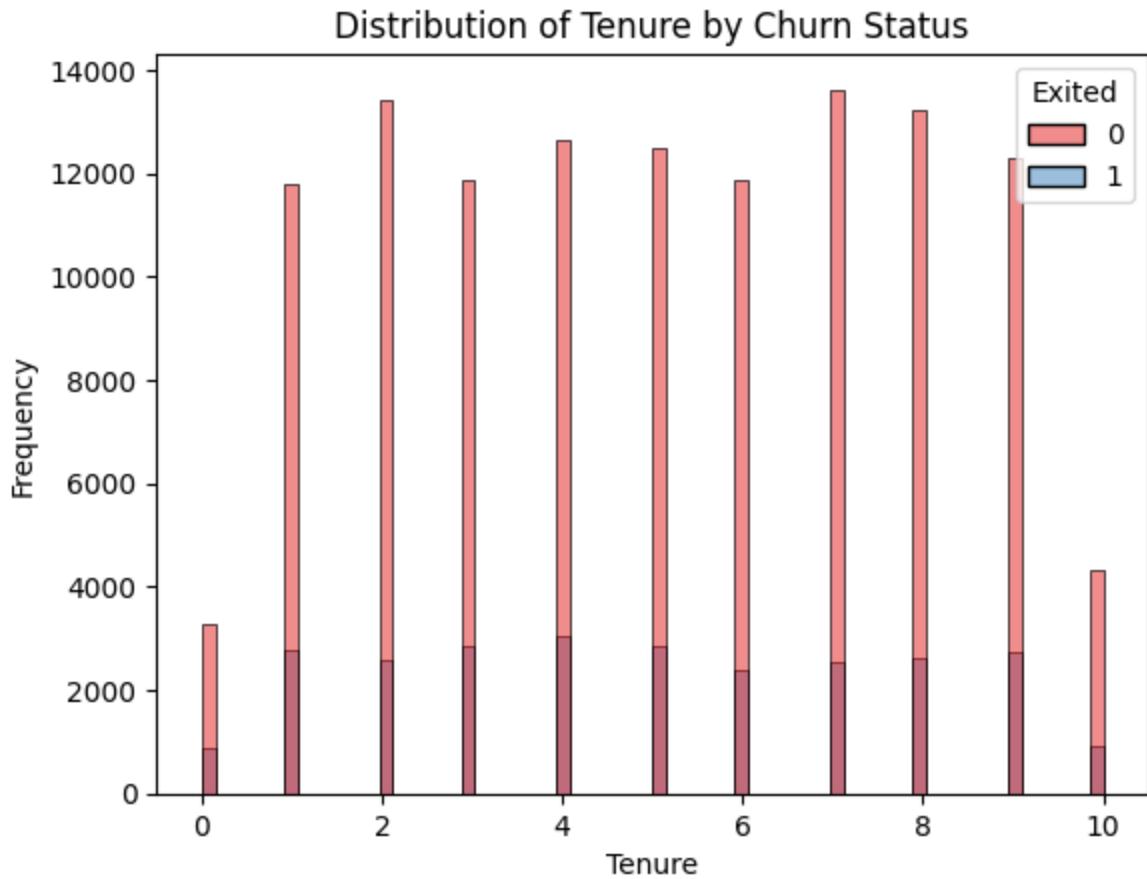
3.2.4. Phân bố của số năm khách hàng đồng hành cùng ngân hàng với trạng thái Churn

```

sns.histplot(data=new_data, x='Tenure', hue='Exited', palette='Set1', kde=False)

plt.title('Distribution of Tenure by Churn Status')
plt.xlabel('Tenure')
plt.ylabel('Frequency')
plt.show()

```

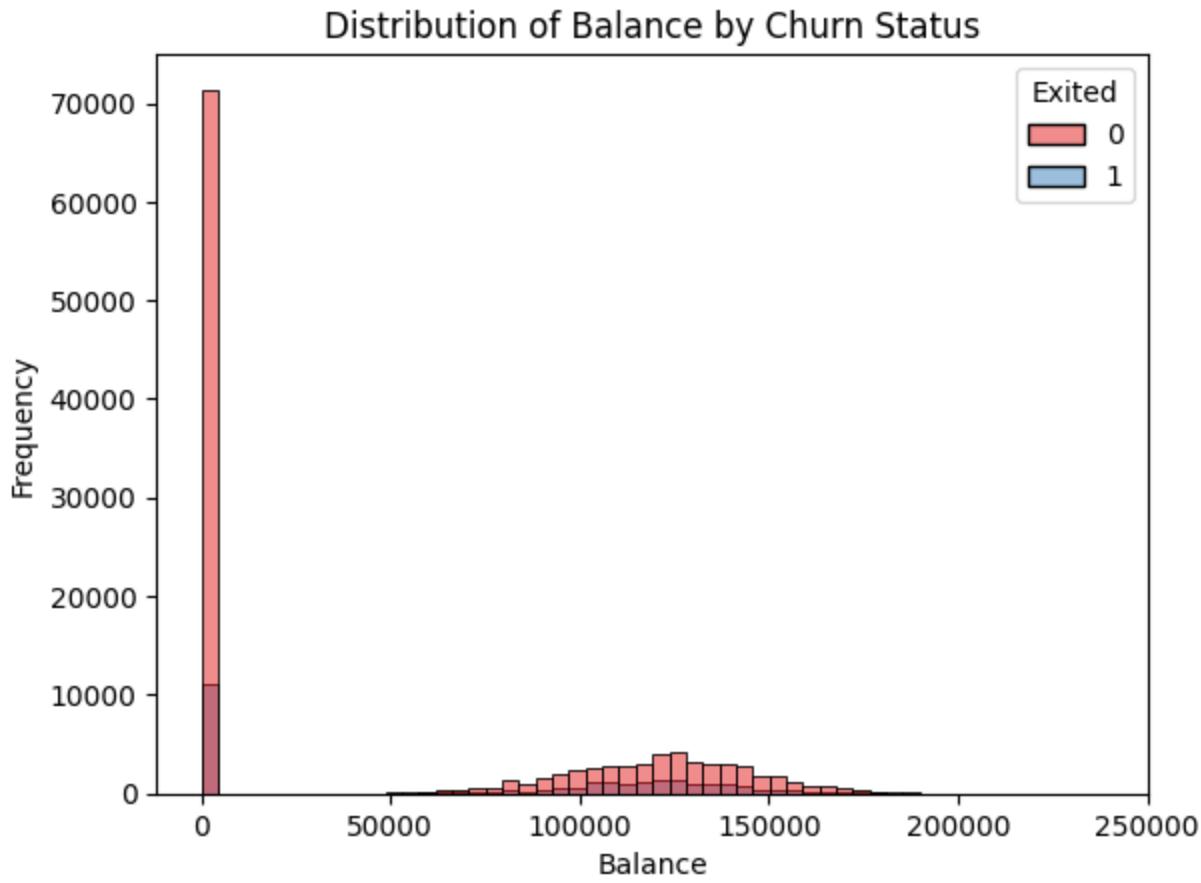


3.2.5. Phân bố của số dư tài khoản khách hàng với trạng thái Churn

```

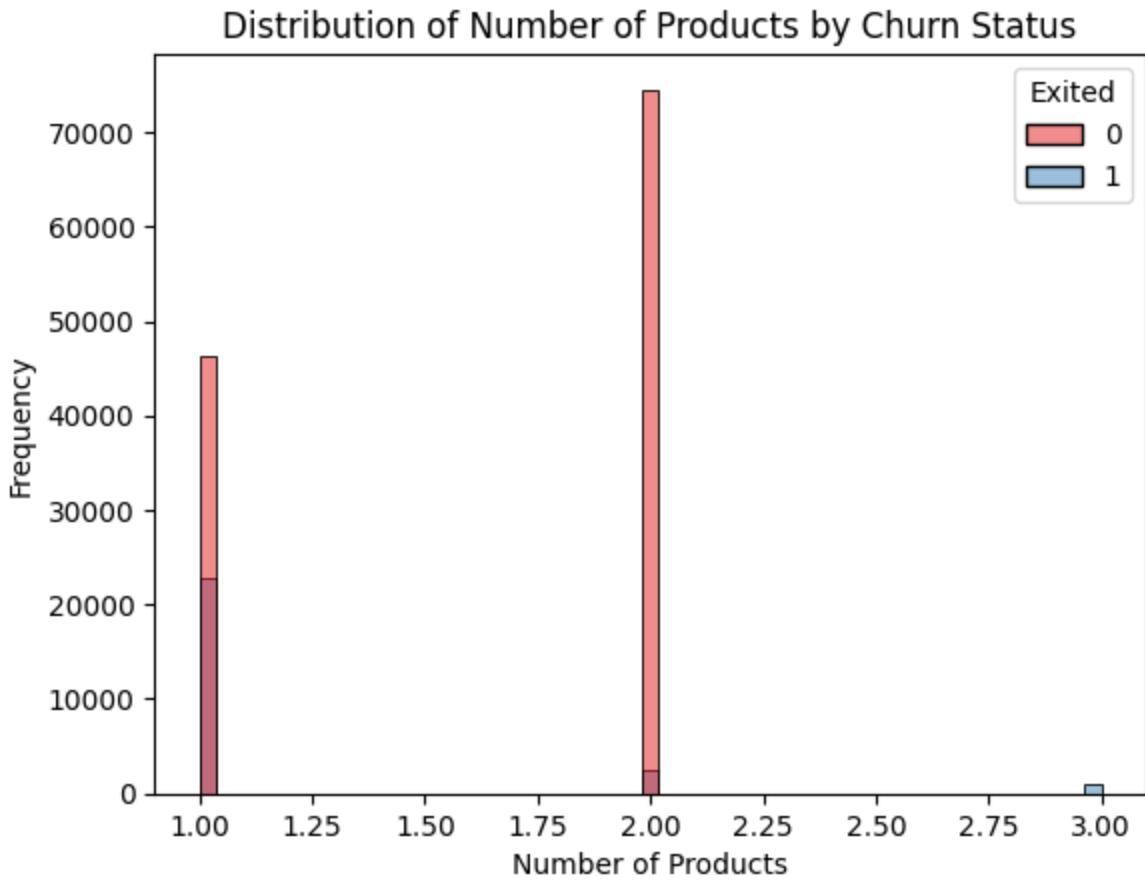
sns.histplot(data=new_data, x='Balance', hue='Exited', palette='Set1', kde=False)
plt.title('Distribution of Balance by Churn Status')
plt.xlabel('Balance')
plt.ylabel('Frequency')
plt.show()

```



3.2.6. Phân bố số lượng sản phẩm khách hàng mua với tỉ lệ rời bỏ

```
sns.histplot(data=new_data, x='NumOfProducts', hue='Exited', palette='Set1', kde=False)
plt.title('Distribution of Number of Products by Churn Status')
plt.xlabel('Number of Products')
plt.ylabel('Frequency')
plt.show()
```

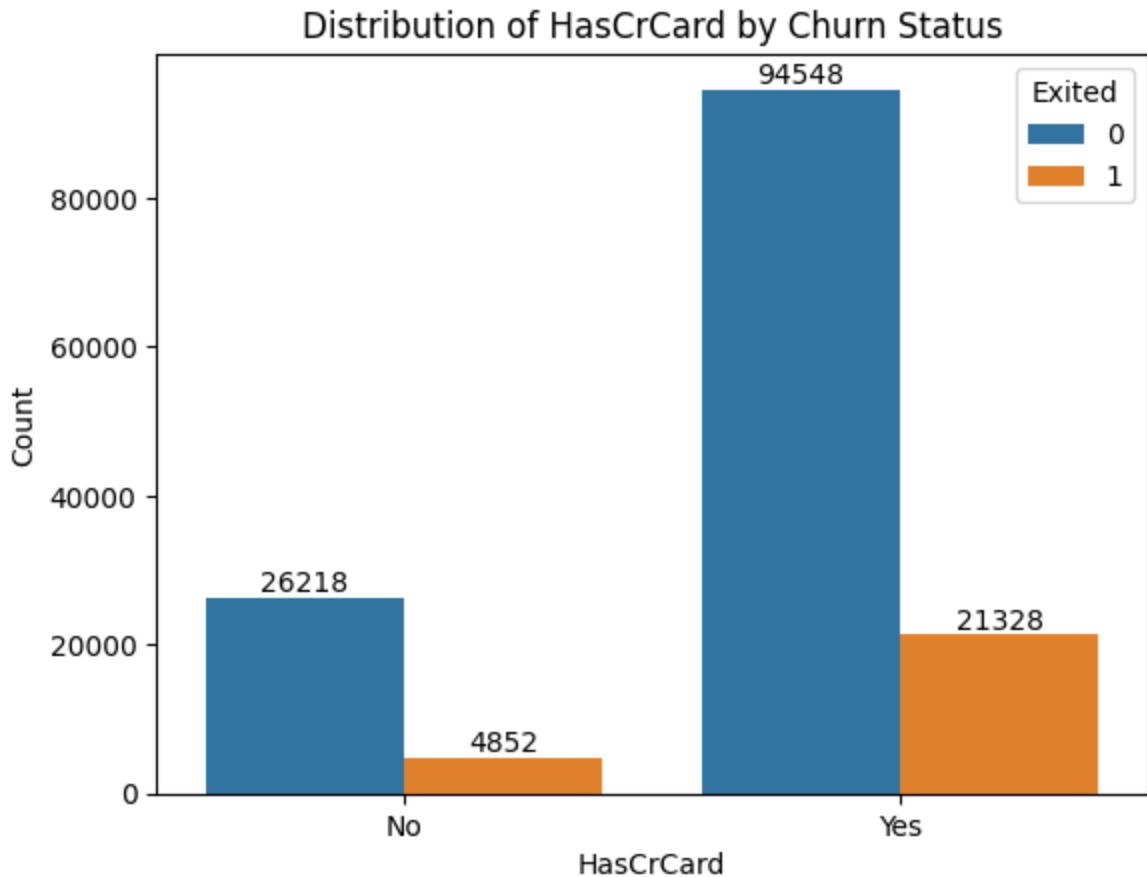


3.2.7. Trực quan khách hàng có thẻ tín dụng với tỉ lệ Churn

```

labels = ['No', 'Yes']
a= sns.countplot(data=new_data,x="HasCrCard",hue="Exited")
plt.title('Distribution of HasCrCard by Churn Status')
plt.xlabel('HasCrCard')
plt.ylabel('Count')
plt.xticks(ticks=[0, 1], labels=labels)
for bars in a.containers:
    a.bar_label(bars)
plt.show()

```

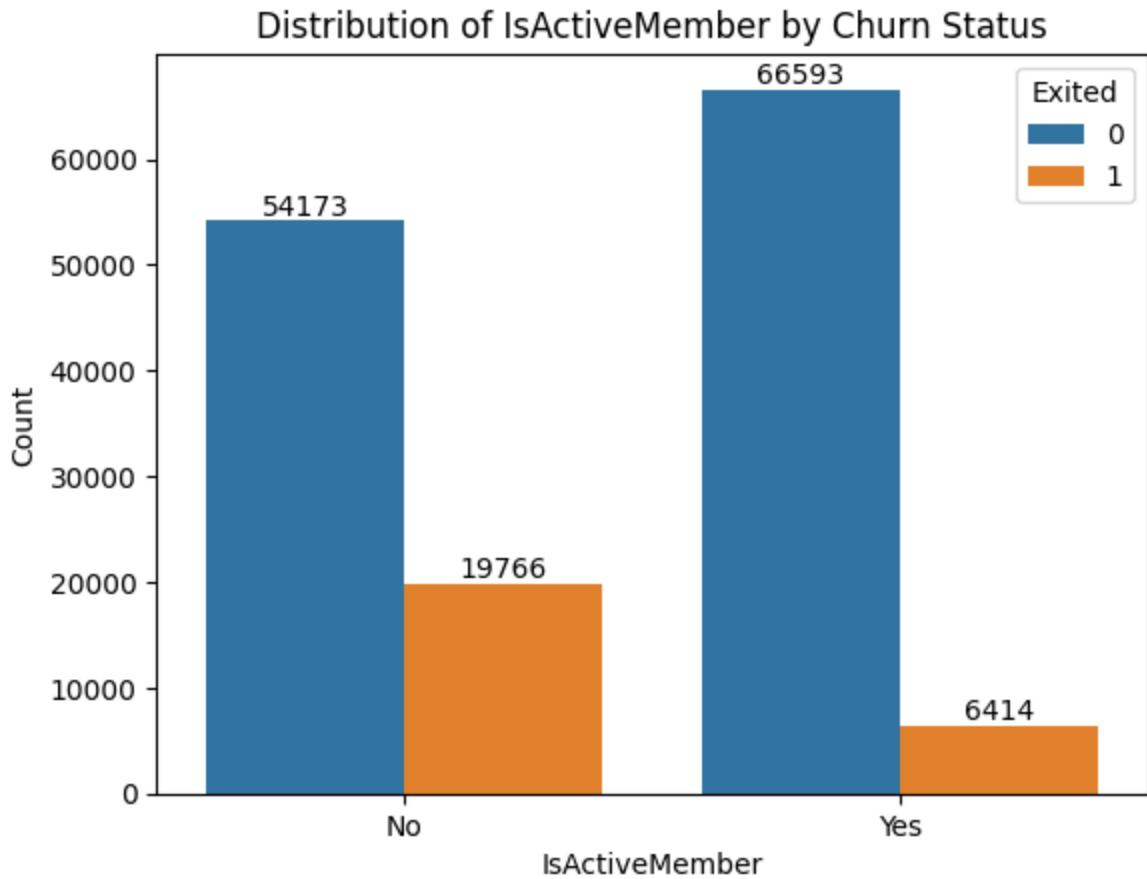


3.2.8. Trực quan khách hàng thân thiết với tỉ lệ Churn

```

labels = ['No', 'Yes']
a = sns.countplot(data=new_data, x="IsActiveMember", hue="Exited")
plt.title('Distribution of IsActiveMember by Churn Status')
plt.xlabel('IsActiveMember')
plt.ylabel('Count')
plt.xticks(ticks=[0, 1], labels=labels)
for bars in a.containers:
    a.bar_label(bars)
plt.show()

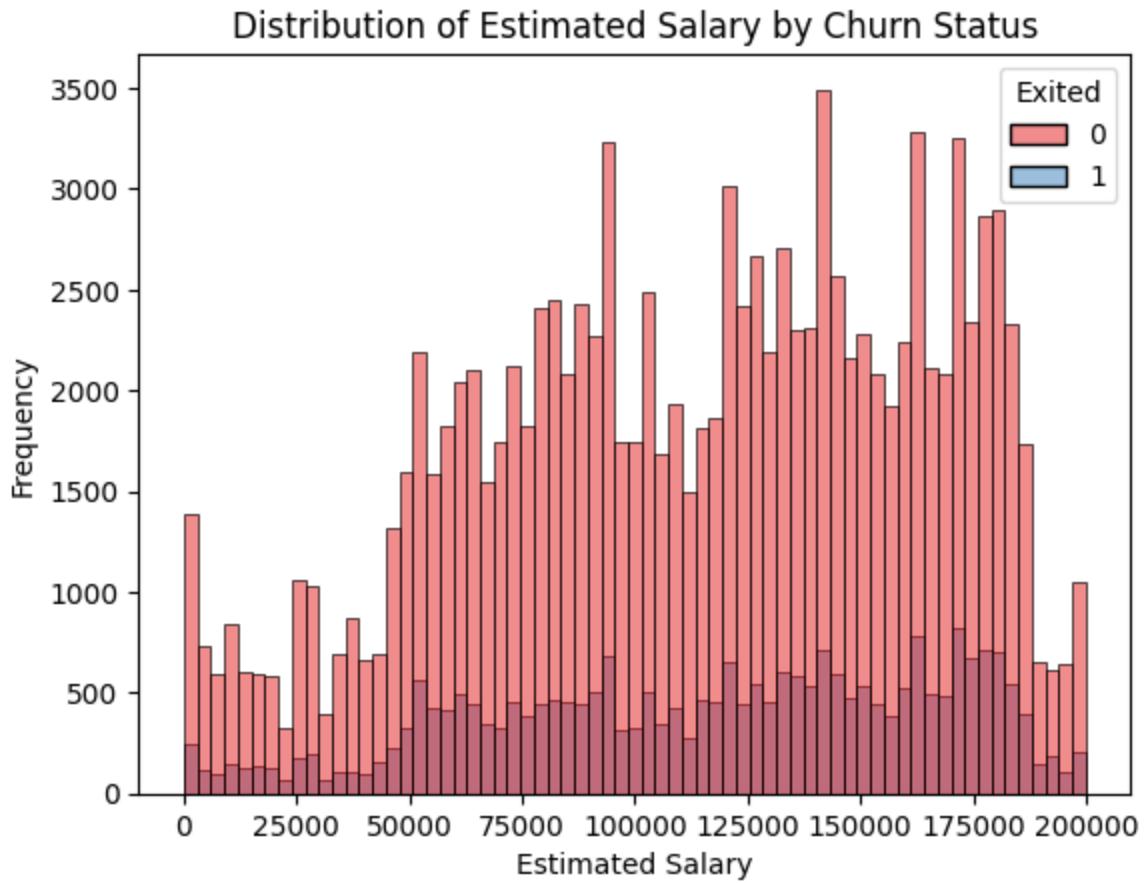
```



3.2.9. Trực quan tiền lương của khách hàng với tỉ lệ Churn

```
sns.histplot(data=new_data, x='EstimatedSalary', hue='Exited', palette='Set1', kde=False)

plt.title('Distribution of Estimated Salary by Churn Status')
plt.xlabel('Estimated Salary')
plt.ylabel('Frequency')
plt.show()
```



3.3. Khai phá luật kết hợp

3.3.1. Chuyển đổi các biến liên tục thành các biến phân loại

```
# Chuyển đổi các biến liên tục thành các nhóm
tempdf = data.copy()
tempdf['CreditScoreGroup'] = pd.cut(data['CreditScore'], bins=[300, 500, 700, 900], labels=['Low', 'Medium', 'High'])
tempdf['AgeGroup'] = pd.cut(data['Age'], bins=[18, 30, 60, 100], labels=['Young', 'Adult', 'Senior'])
tempdf['BalanceGroup'] = pd.cut(data['Balance'], bins=[0, 50000, 150000, 250000], labels=['Low', 'Medium', 'High'])
tempdf['EstimatedSalaryGroup'] = pd.cut(data['EstimatedSalary'], bins=[0, 50000, 150000, 200000], labels=['Low', 'Medium', 'High'])

# Chuyển đổi Tenure và NumOfProducts thành các nhóm nhị phân
tempdf['TenureGroup'] = pd.cut(data['Tenure'], bins=[0, 3, 6, 10], labels=['Short', 'Medium', 'Long'])
tempdf['NumOfProductsGroup'] = pd.cut(data['NumOfProducts'], bins=[0, 1, 2, 3, 4], labels=['One', 'Two', 'Three', 'Four'])

# Chuyển đổi HasCrCard IsActiveMember và Exited thành các bool
tempdf['HasCrCard'] = data['HasCrCard'].astype(bool)
tempdf['IsActiveMember'] = data['IsActiveMember'].astype(bool)
tempdf['Exited'] = data['Exited'].astype(bool)

# Chọn các cột cần thiết
cols = ['CreditScoreGroup', 'Geography', 'Gender', 'AgeGroup', 'TenureGroup', 'BalanceGroup',
        'NumOfProductsGroup', 'HasCrCard', 'IsActiveMember', 'EstimatedSalaryGroup', 'Exited']

# Tạo DataFrame chứa các đặc trưng đã chuyển đổi
df_encode = tempdf[cols]

# Chuyển đổi dữ liệu thành dạng phù hợp cho khai phá luật kết hợp
df_encode = pd.get_dummies(df_encode)

tempdf = None # Xoá dataframe tạm

df_encode
```

	HasCrCard	IsActiveMember	Exited	CreditScoreGroup_Low	CreditScoreGroup_Medium	CreditScoreGroup_High	Geography_France	Geography_Germany	Geography_Spain
0	True	False	False	False	True	False	True	False	False
1	True	True	False	False	True	False	True	False	False
2	True	False	False	False	True	False	True	False	False
3	True	True	False	False	True	False	True	False	False
4	True	True	False	False	False	True	False	False	True
...
165029	True	True	False	False	True	False	False	False	True
165030	False	False	False	False	False	True	True	False	False
165031	True	True	False	False	True	False	True	False	False
165032	False	True	False	False	True	False	False	False	True
165033	True	False	True	False	False	True	True	False	False

165034 rows × 27 columns

3.3.2. Sử dụng thuật toán Apriori để khai phá luật kết hợp

Áp dụng thuật toán Apriori

```
# Áp dụng thuật toán apriori để tìm các tập hợp thường xuyên
frequent_itemsets = apriori(df_encode, min_support=0.01, use_colnames=True)
```

- min_support=0.01: Đây là ngưỡng hỗ trợ tối thiểu (minimum support threshold). Tập hợp các mục phải xuất hiện trong ít nhất 1% tổng số record để được coi là tập hợp thường xuyên.

Khai phá luật kết hợp

```
# Khai phá Luật kết hợp
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)
```

- metric="confidence": Đây là thước đo để đánh giá độ mạnh của các luật kết hợp. confidence (độ tin cậy) của một luật $A \rightarrow B$ được tính bằng tỷ lệ số giao dịch chứa cả A và B so với số giao dịch chứa A.
- min_threshold=0.7: Đây là ngưỡng độ tin cậy tối thiểu. Các luật kết hợp phải có độ tin cậy ít nhất 70% để được đưa vào kết quả.

Trích lọc kết quả

```
# Lọc các rules có consequents là "Exited"
rules = rules[rules['consequents'] == frozenset({'Exited'})]

# Sắp xếp và hiển thị các luật
rules = rules[['antecedents', "consequents", "antecedent support", "consequent support", "support", "lift", "confidence"]]

# Sắp xếp rules giảm dần theo confidence, sau đó tăng dần theo antecedents_Length
rules = rules.sort_values(by=["confidence"], ascending=[False])

# Hiển thị các luật đã sắp xếp
rules
```

Kết quả

		antecedents	consequents	antecedent support	consequent support	support	lift	confidence
327		(AgeGroup_Adult, NumOfProductsGroup_Three)	(Exited)	0.015318	0.211599	0.013712	4.230525	0.895174
756		(HasCrCard, AgeGroup_Adult, NumOfProductsGroup...	(Exited)	0.011476	0.211599	0.010240	4.216902	0.892291
29		(NumOfProductsGroup_Three)	(Exited)	0.017536	0.211599	0.015476	4.170702	0.882516
88		(HasCrCard, NumOfProductsGroup_Three)	(Exited)	0.012967	0.211599	0.011386	4.149538	0.878037

3.3.3. Sử dụng thuật toán FP-Growth để khai phá luật kết hợp

Tương tự như cách sử dụng thuật toán Apriori, cách sử dụng và tham số điều chỉnh cho thuật toán cũng tương tự.

Áp dụng thuật toán FP-Growth

```
# Áp dụng thuật toán fpgrowth để tìm các tập hợp thường xuyên
frequent_itemsets = fpgrowth(df_encode, min_support=0.01, use_colnames=True)
```

- min_support=0.01: Đây là ngưỡng hỗ trợ tối thiểu (minimum support threshold). Tập hợp các mục phải xuất hiện trong ít nhất 1% tổng số record để được coi là tập hợp thường xuyên.

Khai phá luật kết hợp

```
# Khai phá Luật kết hợp
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.7)
```

- metric="confidence": Đây là thước đo để đánh giá độ mạnh của các luật kết hợp. confidence (độ tin cậy) của một luật $A \rightarrow B$ được tính bằng tỷ lệ số giao dịch chứa cả A và B so với số giao dịch chứa A.
- min_threshold=0.7: Đây là ngưỡng độ tin cậy tối thiểu. Các luật kết hợp phải có độ tin cậy ít nhất 70% để được đưa vào kết quả.

Trích lọc kết quả

```
# Lọc các rules có consequents là "Exited"
rules = rules[rules['consequents'] == frozenset({'Exited'})]

# Sắp xếp và hiển thị các Luật
rules = rules[["antecedents", "consequents", "antecedent support", "consequent support", "support", "lift", "confidence"]]

# Sắp xếp rules giảm dần theo confidence, sau đó tăng dần theo antecedents_Length
rules = rules.sort_values(by=["confidence"], ascending=[False])

# Hiển thị các Luật đã sắp xếp
rules
```

Kết quả

		antecedents	consequents	antecedent support	consequent support	support	lift	confidence
13319		(AgeGroup_Adult, NumOfProductsGroup_Three)	(Exited)	0.015318	0.211599	0.013712	4.230525	0.895174
13326		(HasCrCard, AgeGroup_Adult, NumOfProductsGroup...)	(Exited)	0.011476	0.211599	0.010240	4.216902	0.892291
13316		(NumOfProductsGroup_Three)	(Exited)	0.017536	0.211599	0.015476	4.170702	0.882516
13324		(HasCrCard, NumOfProductsGroup_Three)	(Exited)	0.012967	0.211599	0.011386	4.149538	0.878037

3.3.4. Giải thích các kết quả

Từ các kết quả trên, có thể thấy:

1. (**NumOfProductsGroup_Three, AgeGroup_Adult**) \rightarrow (**Exited**)

- **Confidence (Độ tin cậy): 0.895:** 89.5% khách hàng thuộc nhóm tuổi từ 30-60 và dùng 3 sản phẩm của ngân hàng sẽ rời khỏi ngân hàng.
- **Support (Độ hỗ trợ): 0.014:** 1.4% tổng số khách hàng thuộc nhóm này.
- **Lift (Độ nâng): 4.231:** Khả năng rời khỏi ngân hàng của khách hàng thuộc nhóm này cao gấp 4.231 lần so với trung bình.

2. (**NumOfProductsGroup_Three, AgeGroup_Adult, HasCrCard**) -> (**Exited**)

- **Confidence: 0.892:** 89.2% khách hàng thuộc nhóm tuổi từ 30-60, có thẻ tín dụng và dùng 3 sản phẩm của ngân hàng sẽ rời khỏi ngân hàng.
- **Support: 0.010:** 1.0% tổng số khách hàng thuộc nhóm này.
- **Lift: 4.217:** Khả năng rời khỏi ngân hàng của khách hàng thuộc nhóm này cao gấp 4.217 lần so với trung bình.

3. (**NumOfProductsGroup_Three**) -> (**Exited**)

- **Confidence: 0.883:** 88.3% khách hàng dùng 3 sản phẩm của ngân hàng sẽ rời khỏi ngân hàng.
- **Support: 0.015:** 1.5% tổng số khách hàng thuộc nhóm này.
- **Lift: 4.171:** Khả năng rời khỏi ngân hàng của khách hàng thuộc nhóm này cao gấp 4.171 lần so với trung bình.

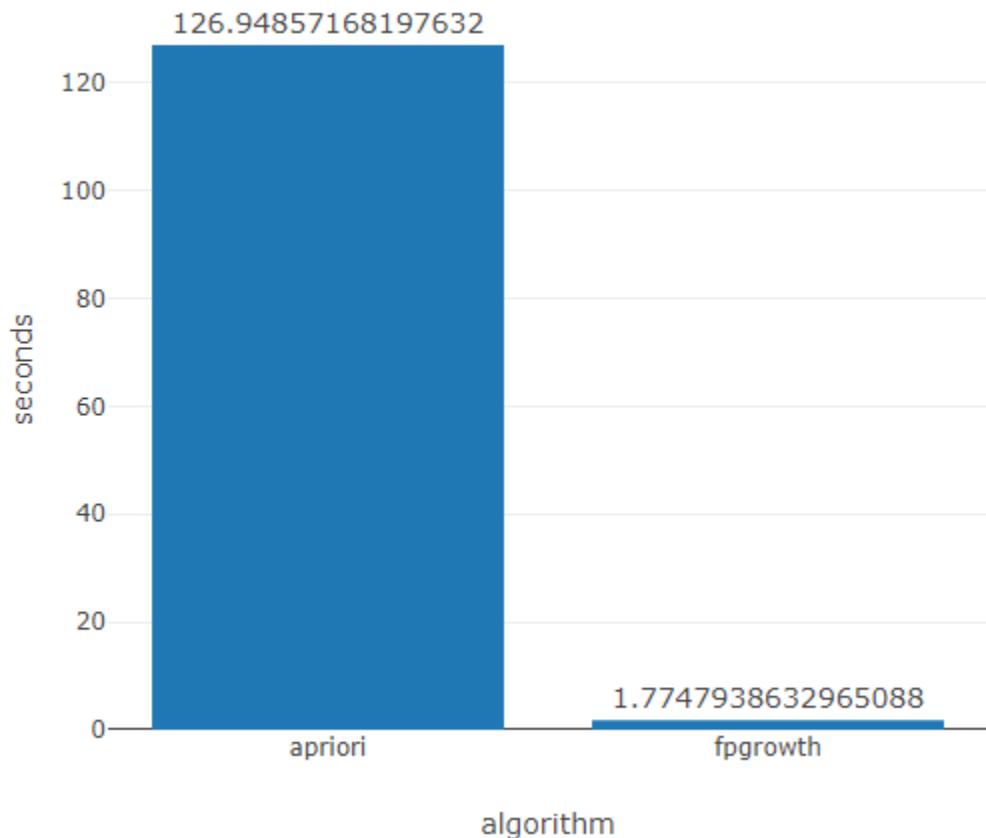
4. (**NumOfProductsGroup_Three, HasCrCard**) -> (**Exited**)

- **Confidence: 0.878:** 87.8% khách hàng có thẻ tín dụng và dùng 3 sản phẩm của ngân hàng sẽ rời khỏi ngân hàng.
- **Support: 0.011:** 1.1% tổng số khách hàng thuộc nhóm này.

- **Lift: 4.150:** Khả năng rời khỏi ngân hàng của khách hàng thuộc nhóm này cao gấp 4.150 lần so với trung bình.

Nhìn chung, cả 4 luật kết hợp trên đều nói về khách hàng dùng 3 sản phẩm của ngân hàng có khả năng cao sẽ rời khỏi ngân hàng.

3.3.5. Đánh giá



Kết quả trên được thực hiện từ ngày 30/05/2024, do điều kiện của môi trường chạy khác nhau, do đó kết quả thực tế có thể khác biệt.

Từ các kết quả luật kết hợp ở mục 3.3.2 và 3.3.3, có thể thấy rằng cả hai thuật toán Apriori và FP-Growth đều cho ra các kết quả tương tự về các luật kết hợp được tìm thấy. Tuy nhiên, thời gian thực hiện của thuật toán FP-Growth hoạt động tốt nhất với

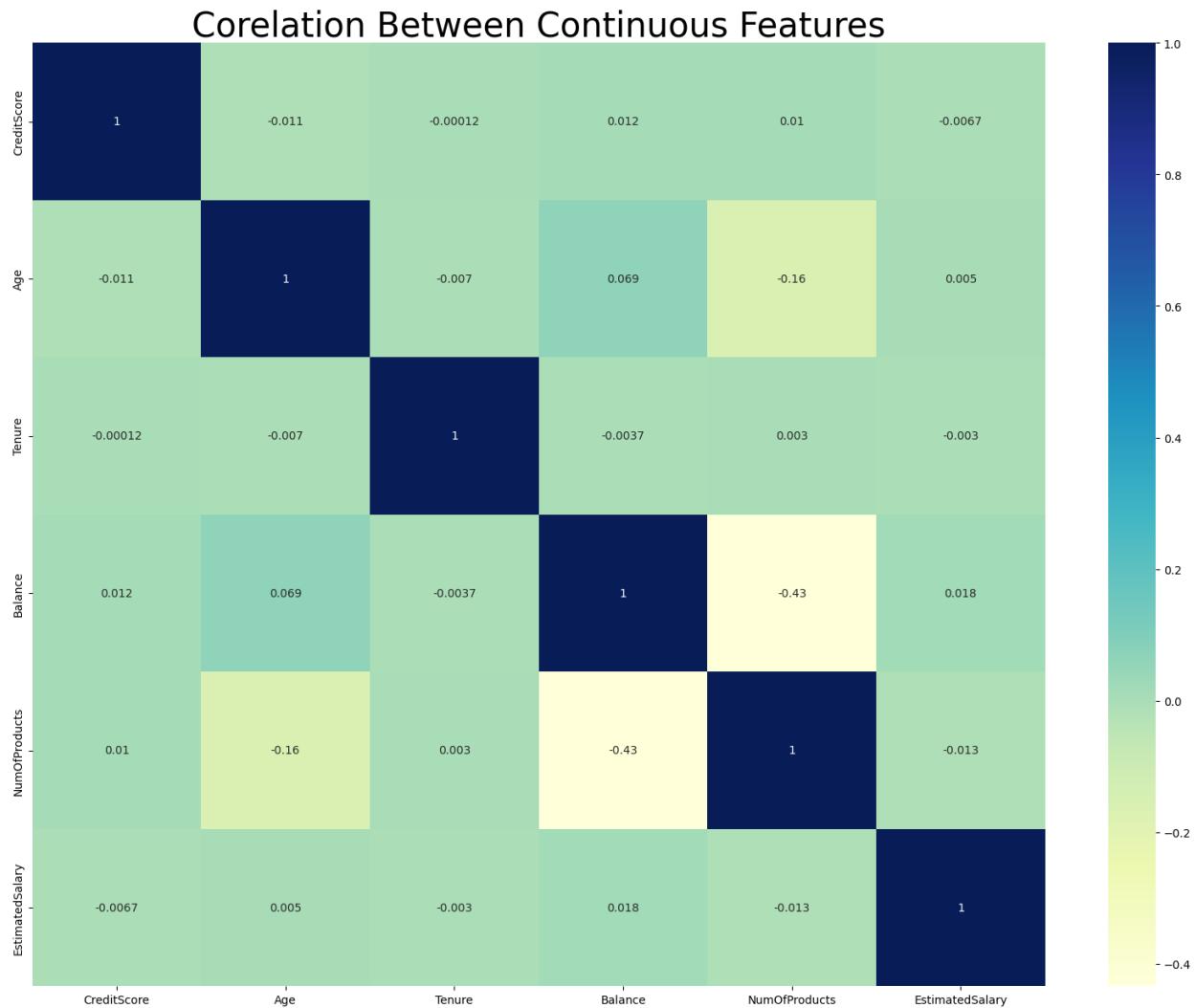
thời gian thực thi chỉ là 1.8 giây nhỏ hơn nhiều so với thời gian thực hiện của thuật toán Apriori, cho thấy ưu thế về hiệu suất so với thuật toán Apriori. Thuật toán Apriori mặc dù cũng tìm ra các luật kết hợp tương tự nhưng có thời gian thực thi dài hơn, khẳng định rằng FP-Growth là lựa chọn tốt hơn khi xét về tốc độ xử lý.

3.4. Trích lọc đặc trưng

3.4.1. Phân tích mối quan hệ giữa biến liên tục với biến liên tục

Sử dụng kỹ thuật Pearson

```
plt.figure(figsize=(20, 15))
sns.heatmap(new_data[continuous_variables].corr(method='spearman'), annot=True, cmap='YlGnBu')
plt.title('Corelation Between Continuous Features', fontsize=30)
plt.show()
```

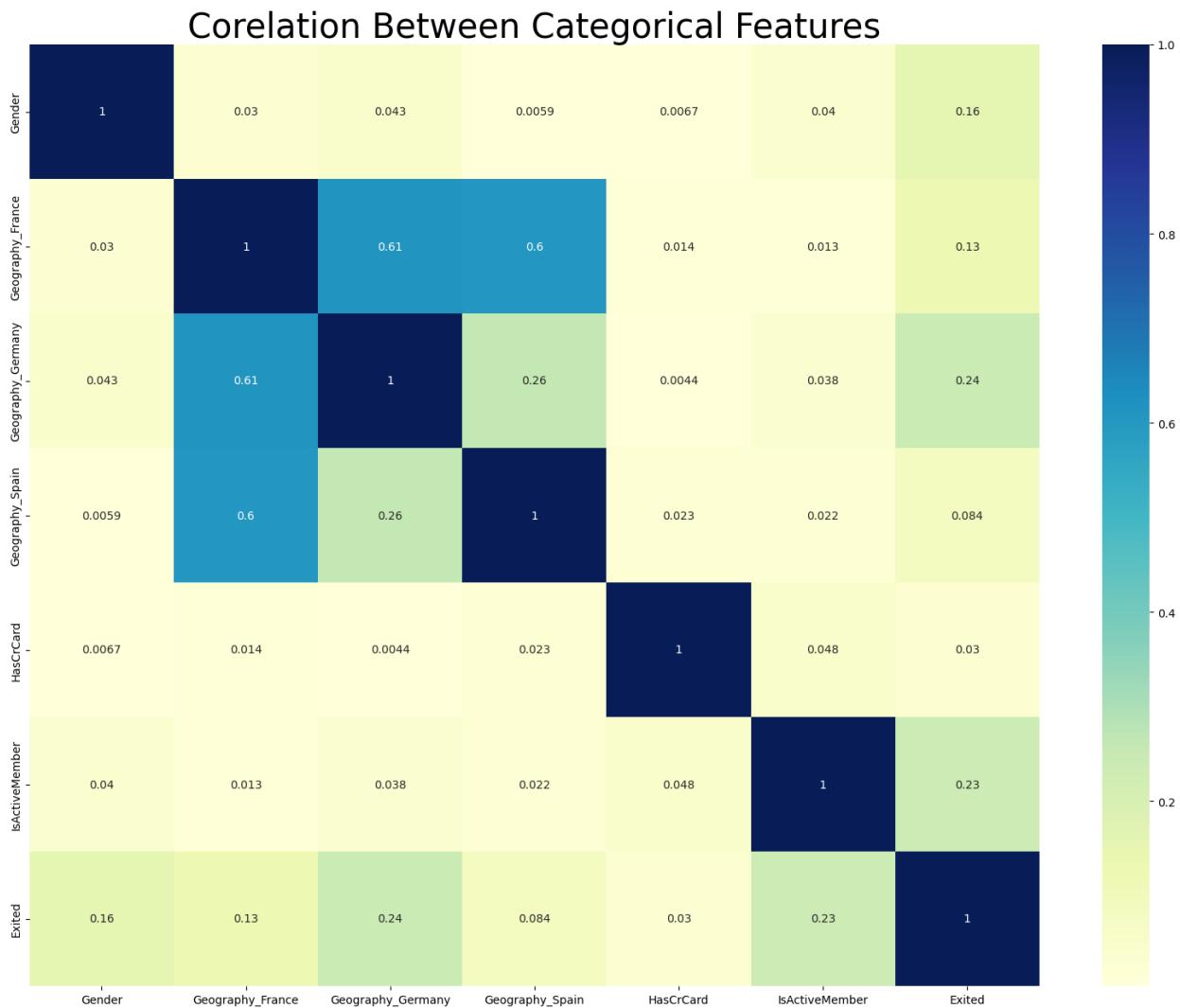


3.4.2 Phân tích mối quan hệ giữa biến phân loại với biến phân loại

Sử dụng phương thức Cramer's V

```
def cramers_v(confusion_matrix):
    chi2 = ss.chi2_contingency(confusion_matrix)[0]
    n = confusion_matrix.sum()
    phi2 = chi2 / n
    r, k = confusion_matrix.shape
    phi2corr = max(0, phi2 - ((k - 1) * (r - 1)) / (n - 1))
    rcorr = r - ((r - 1) ** 2) / (n - 1)
    kcorr = k - ((k - 1) ** 2) / (n - 1)
    return np.sqrt(phi2corr / min((kcorr - 1), (rcorr - 1)))
```

```
new_data[categorical_variables].corr(method=lambda x, y: cramers_v(pd.crosstab(x, y).values))
plt.figure(figsize=(20, 15))
sns.heatmap(new_data[categorical_variables].corr(method=lambda x, y: cramers_v(pd.crosstab(x, y).values)), annot=True, cmap='YlGnBu')
plt.title('Corelation Between Categorical Features', fontsize=30)
plt.show()
```



=> Biến mục tiêu 'Exited' có mối liên kết với 'Geography_Germany' và 'IsActiveMember'

3.4.3. Phân tích mối quan hệ giữa biến phân loại với biến liên tục

Sử dụng phương pháp kiểm thử ANOVA

```
from statsmodels.formula.api import ols
coefficient_df = pd.DataFrame(columns=["categorical_variables", "continuous_variables", "p-value", "select_feature"])

for cate_var in categorical_variables:
    for conti_var in continuous_variables:
        model = ols(f'{cate_var} ~ C({cate_var})', data=new_data).fit()
        anova_table = sm.stats.anova_lm(model, typ=1)
        row = {"categorical_variables": cate_var, "continuous_variables": conti_var, "p-value": anova_table["PR(>F)"][0], "select_feature": anova_table["PR(>F)"][0] <= 0.05}
        coefficient_df = coefficient_df.append(row, ignore_index = True)

coefficient_df
```

	categorical_variables	continuous_variables	p-value	select_feature
0	Gender	CreditScore	9.367959e-02	False
1	Gender	Age	8.266389e-175	True
2	Gender	Tenure	3.145242e-05	True
3	Gender	Balance	1.779630e-11	True
4	Gender	NumOfProducts	1.106840e-44	True
5	Gender	EstimatedSalary	1.386108e-01	False
6	Geography_France	CreditScore	4.221454e-04	True
7	Geography_France	Age	6.819559e-184	True
8	Geography_France	Tenure	4.102219e-01	False
9	Geography_France	Balance	0.000000e+00	True
10	Geography_France	NumOfProducts	2.472179e-190	True
11	Geography_France	EstimatedSalary	6.484983e-06	True
12	Geography_Germany	CreditScore	2.581990e-01	False
13	Geography_Germany	Age	0.000000e+00	True
14	Geography_Germany	Tenure	3.353762e-02	True
15	Geography_Germany	Balance	0.000000e+00	True

=> Biến mục tiêu 'Exited' có giá trị thống kê với các biến liên tục. Đối với biến phân loại 'Geography_Germany' thì nó có sự tương đồng với 'CreditScore' nên cần loại bỏ 'CreditScore'

=> Các cột được giữ lại 'Geography_Germany', 'IsActiveMember' và các biến liên tục (trừ 'CreditScore')

3.4.4. Đánh giá mức độ quan trọng của các thuộc tính để dự đoán Exited bằng RandomForest

```
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(new_data.drop(columns="Exited"), new_data['Exited'], test_size=0.2, random_state=42)

# Huấn luyện mô hình trên tập train
rdf_clf = RandomForestClassifier(
    max_depth = 4,
    max_leaf_nodes = 16,
    min_samples_split = 10,
    min_samples_leaf = 10
)

rdf_clf.fit(X_train, y_train)

# Dự báo trên tập test
y_pred = rdf_clf.predict(X_test)
scores = accuracy_score(y_pred, y_test)
print('RandomForest Accuracy: {:.03f}'.format(scores))

RandomForest Accuracy: 0.873
```

Tỉ lệ chính xác của mô hình Random Forest là 87%

Tỉ lệ quan trọng của các thuộc tính để dự đoán Exited

features	importance
2 Age	0.373430
5 NumOfProducts	0.319623
10 Geography_Germany	0.109559
7 IsActiveMember	0.106051
1 Gender	0.038213
4 Balance	0.030654
9 Geography_France	0.012994
11 Geography_Spain	0.006120
0 CreditScore	0.001310
6 HasCrCard	0.001174
8 EstimatedSalary	0.000629
3 Tenure	0.000245

Nhận xét: Mô hình Random Forest cho thấy rằng các đặc trưng như tuổi, số lượng sản phẩm, địa lý (đặc biệt là Đức), và tình trạng hoạt động của khách hàng là các yếu tố quan trọng nhất để dự đoán hành vi rời bỏ ngân hàng của khách hàng.

Các đặc trưng khác, mặc dù có ảnh hưởng nhất định, nhưng không quan trọng bằng và có thể ít tác động đến quyết định của mô hình.

3.4.5. Đánh giá mức độ quan trọng của các thuộc tính để dự đoán Exited bằng LogisticRegression

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression(solver='liblinear')

lr.fit(X_train, y_train)

rdf_clf.fit(X_train, y_train)

# Dự báo trên tập test
y_pred = lr.predict(X_test)
scores = accuracy_score(y_pred, y_test)
print('Logistic Regression Accuracy: {:.03f}'.format(scores))
```

Logistic Regression Accuracy: 0.882

Tỉ lệ chính xác của LogisticRegression là 88%

Tỉ lệ quan trọng của các thuộc tính để dự đoán Exited

features	contributed	
2	Age	5.648677
1	Gender	0.829481
10	Geography_Germany	0.429132
6	HasCrCard	0.295202
8	EstimatedSalary	0.243494
3	Tenure	-0.157101
0	CreditScore	-0.449894
9	Geography_France	-1.261427
7	IsActiveMember	-1.493958
11	Geography_Spain	-1.500970
4	Balance	-1.982780
5	NumOfProducts	-4.311988

=> Từ 3 phương thức trên những thuộc tính không quan trọng thường sẽ rời vào 'CreditScore', 'HasCrCard', 'Gender'. Và chúng sẽ khác nhau tùy theo mô hình chúng ta sử dụng. Nên tùy vào mô hình chúng ta sẽ lựa chọn thuộc tính lại.

Loại bỏ các cột không cần thiết

```
new_data = new_data.drop(columns=['Gender', 'HasCrCard', 'CreditScore'])
```

CHƯƠNG IV: XÂY DỰNG CÁC MÔ HÌNH

4.1. Các mô hình được chọn (tập trung vào sự diến)

- Logistic Regression
- Decision Tree
- ANN + SHAP

Chia tập dữ liệu thành 2 tập train và test với tỉ lệ 8:2 để xây dựng các mô hình.

```
x_train, x_test, y_train, y_test = train_test_split(data.drop(columns="Exited"), data['Exited'], test_size=0.2, random_state=42)
```

4.1.1. Mô hình dự đoán Decision Tree

Tiến hành tuning hyperparameters cho mô hình

```
parameters = {'criterion':['gini','entropy'],
              'max_depth':np.arange(2,10).tolist()[0::2],
              'min_samples_split':np.arange(3,10).tolist()[0::2],
              'max_leaf_nodes':np.arange(3,26).tolist()[0::2],
              'splitter' : ["best", "random"]}

dt = DecisionTreeClassifier()

g1 = GridSearchCV(dt, parameters, cv=5, n_jobs=-1, scoring='accuracy')
g1.fit(x_train,y_train)

[4] ✓ 2m 2.2s
```

...

```
GridSearchCV
estimator: DecisionTreeClassifier
    DecisionTreeClassifier
```

Python

Accuracy lớn nhất và các tham số tối ưu cho mô hình

```
print(g1.best_score_)
print(g1.best_params_)

[4] ✓ 0.0s
```

```
0.8865902101611592
{'criterion': 'gini', 'max_depth': 8, 'max_leaf_nodes': 25, 'min_samples_split': 3, 'splitter': 'best'}
```

Tiến hành training lại mô hình tốt nhất trên toàn bộ tập train

```
_clf = g1.best_estimator_
_clf.fit(x_train, y_train)
```

Tiến hành hậu cắt tỉa cho cây nhằm đặt được sự diễn giải (chiều sâu của cây thấp và số node ít) và độ chính xác cao.

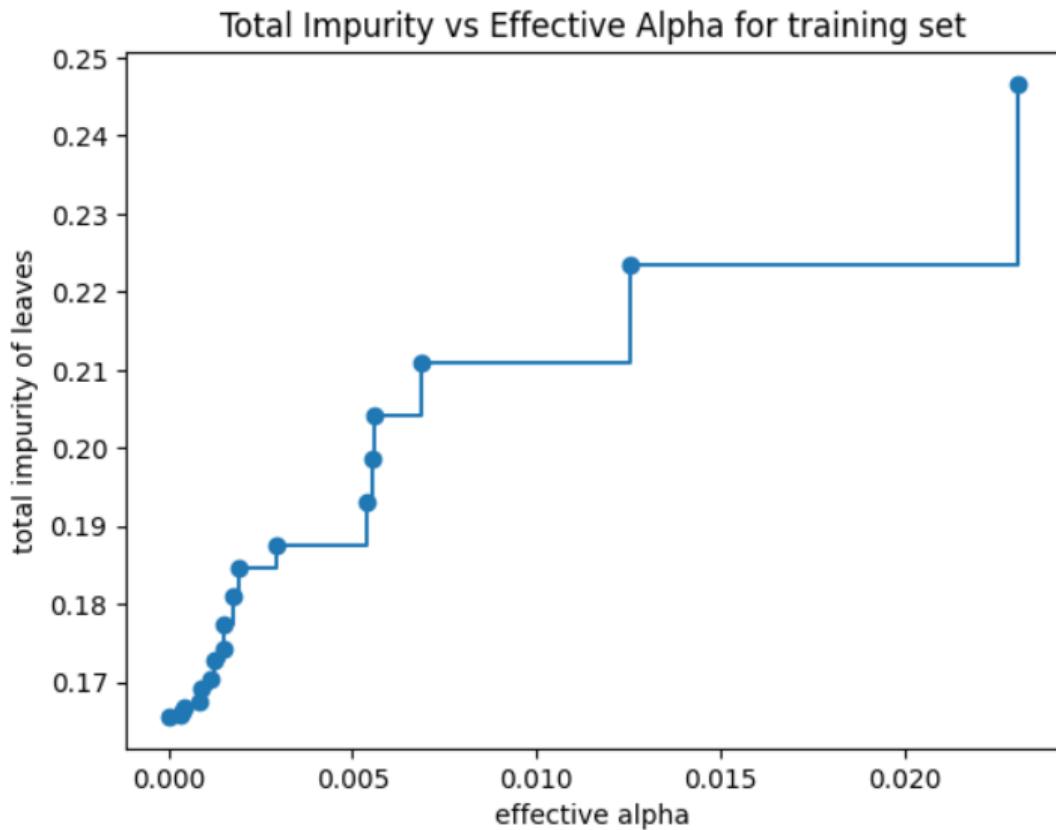
Phương pháp cắt tỉa phức tạp chi phí tối thiểu (Minimal Cost Complexity Pruning) sử dụng đệ quy để tìm ra nút có "liên kết yếu nhất". Liên kết yếu nhất được xác định bởi alpha hiệu quả, trong đó các nút có alpha hiệu quả nhỏ nhất sẽ được cắt tỉa trước.

Trục X: Hiển thị Effective Alpha, là một thước đo mức độ phức tạp của cây quyết định. Giá trị alpha cao hơn dẫn đến việc cắt tỉa nhiều hơn, làm cho cây đơn giản hơn.

Trục Y: Hiển thị Total Impurity của lá, là thước đo mức độ sai sót của cây quyết định. Giá trị cao hơn cho thấy cây có nhiều sai sót hơn.

```
path = _clf.cost_complexity_pruning_path(x_train, y_train)
ccp_alphas, impurities = path ccp_alphas, path impurities
```

```
Text(0.5, 1.0, 'Total Impurity vs Effective Alpha for training set')
```



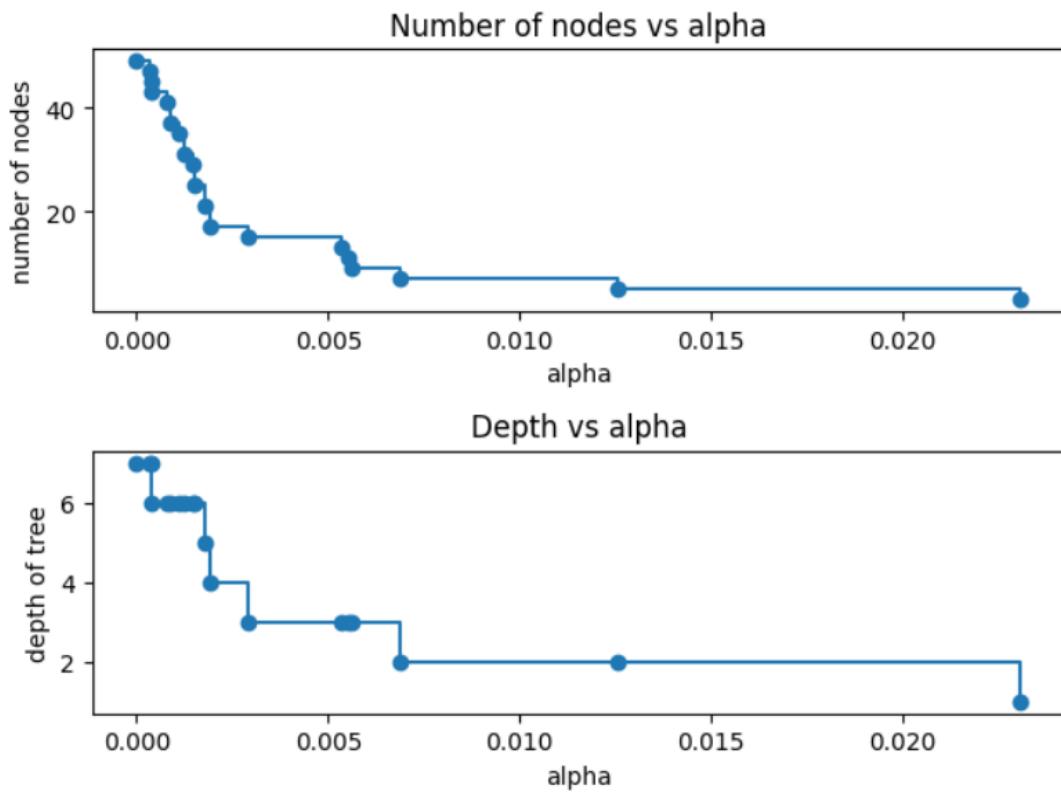
=> Cây khi cắt tía càng đơn giản thì càng nhiều sai sót.

Tiến hành tạo nhiều cây cắt tía theo từng giá trị Effective Alpha để kiểm tra khả năng diễn giải.

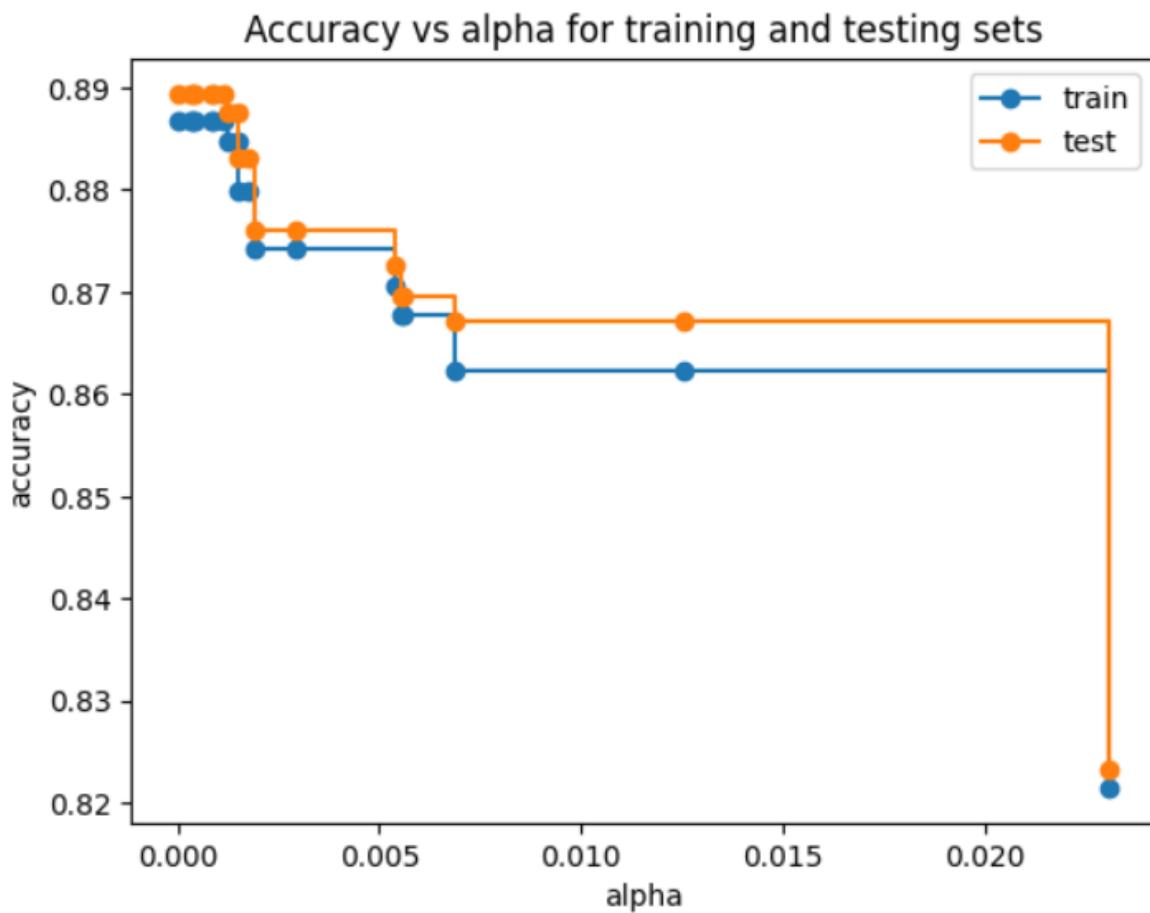
```
clfs = []
for ccp_alpha in ccp_alphas:
    clf = DecisionTreeClassifier(random_state=0, ccp_alpha=ccp_alpha,
                                 criterion='gini', max_depth= 8, max_leaf_nodes= 25, min_samples_split= 3, splitter= 'best')
    clf.fit(X_train, y_train)
    clfs.append(clf)
    print(
        "Number of nodes in the last tree is: {} with ccp_alpha: {}".format(
            clfs[-1].tree_.node_count, ccp_alphas[-1]
        )
    )
] ✓ 3.3s
```

Number of nodes in the last tree is: 1 with ccp_alpha: 0.04666410061221393

Vẽ đồ thị sự thay đổi của chiều sâu và số node khi cắt tía theo effective alpha



Sự thay đổi của accuracy:



Thống kê lại, chọn mô hình với index = 6, đạt độ chính xác cao, chiều giảm còn 6, số nodes 35

	test_scores	depth	node_counts
0	0.889418	7	49
1	0.889418	7	47
2	0.889418	7	45
3	0.889418	6	43
4	0.889418	6	41
5	0.889418	6	37
6	0.889418	6	35
7	0.887581	6	31
8	0.887581	6	29
9	0.883124	6	25
10	0.883124	5	21
11	0.876046	4	17
12	0.876046	3	15
13	0.872542	3	13
14	0.869650	3	11
15	0.869650	3	9
16	0.867064	2	7
17	0.867064	2	5
18	0.823307	1	3

Kết quả đánh giá của mô hình được chọn:

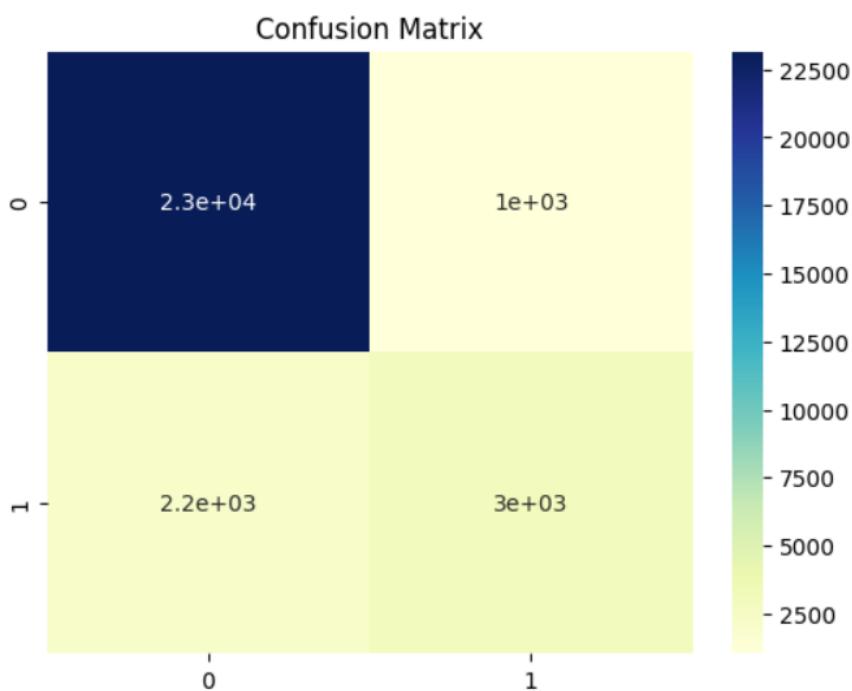
```

y_pred = selected_dt.predict(x_test)

sns.heatmap(confusion_matrix(y_test, selected_dt.predict(x_test)), annot=True, cmap='YlGnBu')
plt.title('Confusion Matrix')
plt.show()

```

✓ 0.1s



Precision: 0.741246585547554

Recall (Sensitivity): 0.5748122472559214

F1-Score: 0.6475054229934925

Nhận xét: Precision khá cao, cho thấy rằng khi mô hình dự đoán một khách hàng sẽ rời bỏ ngân hàng, khả năng dự đoán đúng là cao.

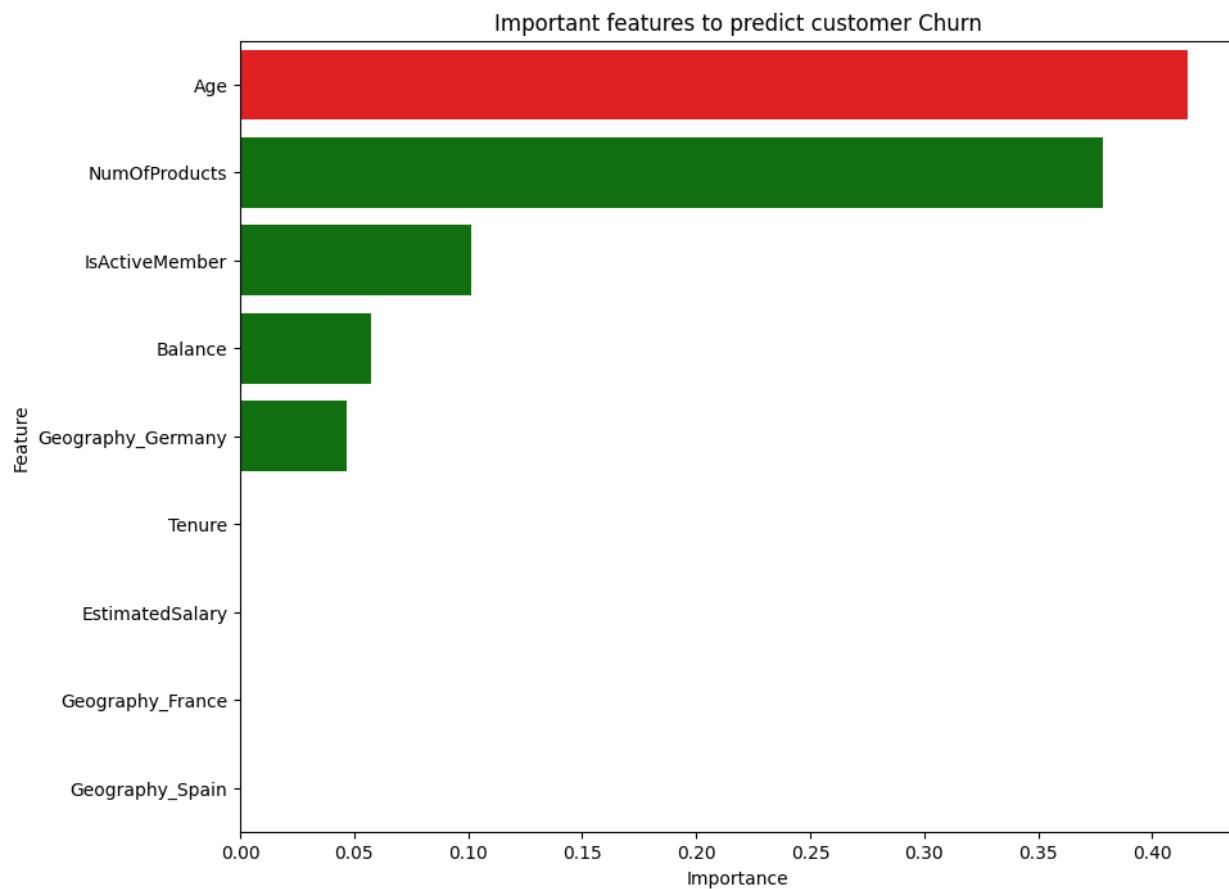
Recall không quá cao, cho thấy mô hình còn bỏ sót khá nhiều khách hàng thực sự rời bỏ ngân hàng.

F1-Score ở mức trung bình, phản ánh rằng mô hình hoạt động tốt hơn trong việc tránh các false positives (dự đoán sai khách hàng sẽ rời bỏ) hơn là tránh các false negatives (bỏ sót khách hàng sẽ rời bỏ).

Thống kê độ quan trọng của thuộc tính trong mô hình:

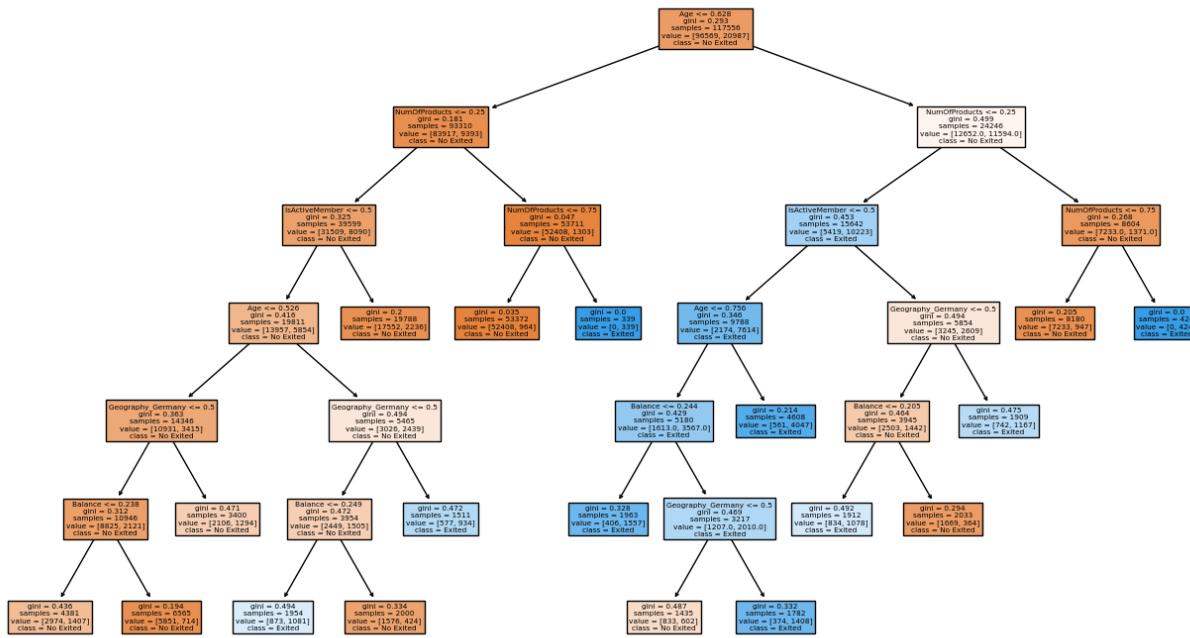
	Feature	Importance
0	Age	0.415587
1	NumOfProducts	0.378548
2	IsActiveMember	0.101606
3	Balance	0.057316
4	Geography_Germany	0.046943
5	Tenure	0.000000
6	EstimatedSalary	0.000000
7	Geography_France	0.000000
8	Geography_Spain	0.000000

=> Các thuộc tính có index từ 5 trở đi không tham gia vào việc đưa ra quyết định của mô hình.



Cây được tạo ra:

Decision tree trained



Điễn giải cây dưới dạng các luật

```

if (Age <= 0.628) and (NumOfProducts > 0.25) and (NumOfProducts <= 0.75) then class: No Exited (proba: 98.19%) | based on 53,372 samples
if (Age <= 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) then class: No Exited (proba: 88.7%) | based on 19,788 samples
if (Age > 0.628) and (NumOfProducts > 0.25) and (NumOfProducts <= 0.75) then class: No Exited (proba: 88.42%) | based on 8,180 samples
if (Age <= 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.526) and (Geography_Germany <= 0.5) and (Balance > 0.238) then class: No Exited (proba: 89.12%) | based on 6,465 samples
if (Age <= 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.510) and (Balance < 0.238) then class: Exited (proba: 87.83%) | based on 4,608 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age <= 0.526) and (Geography_Germany <= 0.5) and (Balance <= 0.238) then class: No Exited (proba: 67.88%) | based on 4,281 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.526) and (Geography_Germany <= 0.5) and (Balance < 0.205) then class: No Exited (proba: 61.94%) | based on 3,400 samples
if (Age > 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Geography_Germany <= 0.5) and (Balance > 0.205) then class: No Exited (proba: 82.1%) | based on 2,033 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age > 0.526) and (Geography_Germany <= 0.5) and (Balance > 0.249) then class: No Exited (proba: 78.8%) | based on 2,000 samples
if (Age > 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age <= 0.756) and (Balance <= 0.244) then class: Exited (proba: 79.32%) | based on 1,963 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.526) and (Geography_Germany <= 0.5) and (Balance <= 0.249) then class: Exited (proba: 55.32%) | based on 1,954 samples
if (Age > 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember > 0.5) and (Geography_Germany <= 0.5) and (Balance <= 0.205) then class: Exited (proba: 56.38%) | based on 1,912 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember > 0.5) and (Geography_Germany > 0.5) then class: Exited (proba: 61.13%) | based on 1,909 samples
if (Age > 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.756) and (Balance > 0.244) and (Geography_Germany > 0.5) then class: Exited (proba: 79.01%) | based on 1,782 samples
if (Age < 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age > 0.526) and (Geography_Germany > 0.5) then class: Exited (proba: 61.81%) | based on 1,511 samples
if (Age > 0.628) and (NumOfProducts <= 0.25) and (IsActiveMember <= 0.5) and (Age < 0.756) and (Balance > 0.244) and (Geography_Germany <= 0.5) then class: No Exited (proba: 58.05%) | based on 1,435 samples
if (Age > 0.628) and (NumOfProducts > 0.25) and (NumOfProducts <= 0.75) then class: Exited (proba: 100.0%) | based on 424 samples
if (Age < 0.628) and (NumOfProducts > 0.25) and (NumOfProducts > 0.75) then class: Exited (proba: 100.0%) | based on 339 samples
  
```

4.1.2. Mô hình dự đoán Logistic Regression

Tiến hành tuning các tham số mô hình

```
param_grid = {  
    'C': [0.001, 0.01, 0.1, 1, 10, 100],  
    'penalty': ['l1', 'l2', 'elasticnet'],  
    'solver': ['saga', 'liblinear'],  
    'max_iter': [100, 200, 300],  
    'class_weight':[None, 'balanced']  
}  
✓ 0.0s
```

```
from sklearn.linear_model import LogisticRegression  
  
logModel = LogisticRegression(random_state=10)  
lr_gs = GridSearchCV(logModel, param_grid = param_grid, cv = 5, n_jobs=-1, scoring='accuracy')  
lr_gs.fit(X_train, y_train)
```

Độ chính xác cao nhất và tham số của mô hình tốt nhất:

```
lr_gs.best_score_
```

```
| ✓ 0.0s
```

0.8788577126249131

```
lr_gs.best_params_
```

```
| ✓ 0.0s
```

```
{'C': 1,  
 'class_weight': None,  
 'max_iter': 100,  
 'penalty': 'l1',  
 'solver': 'liblinear'}
```

Tiến hành training lại toàn bộ tập train dựa trên tham số của tốt nhất:

```
best_lr = lr_gs.best_estimator_
```

✓ 0.0s

```
best_lr.fit(X_train, y_train)
```

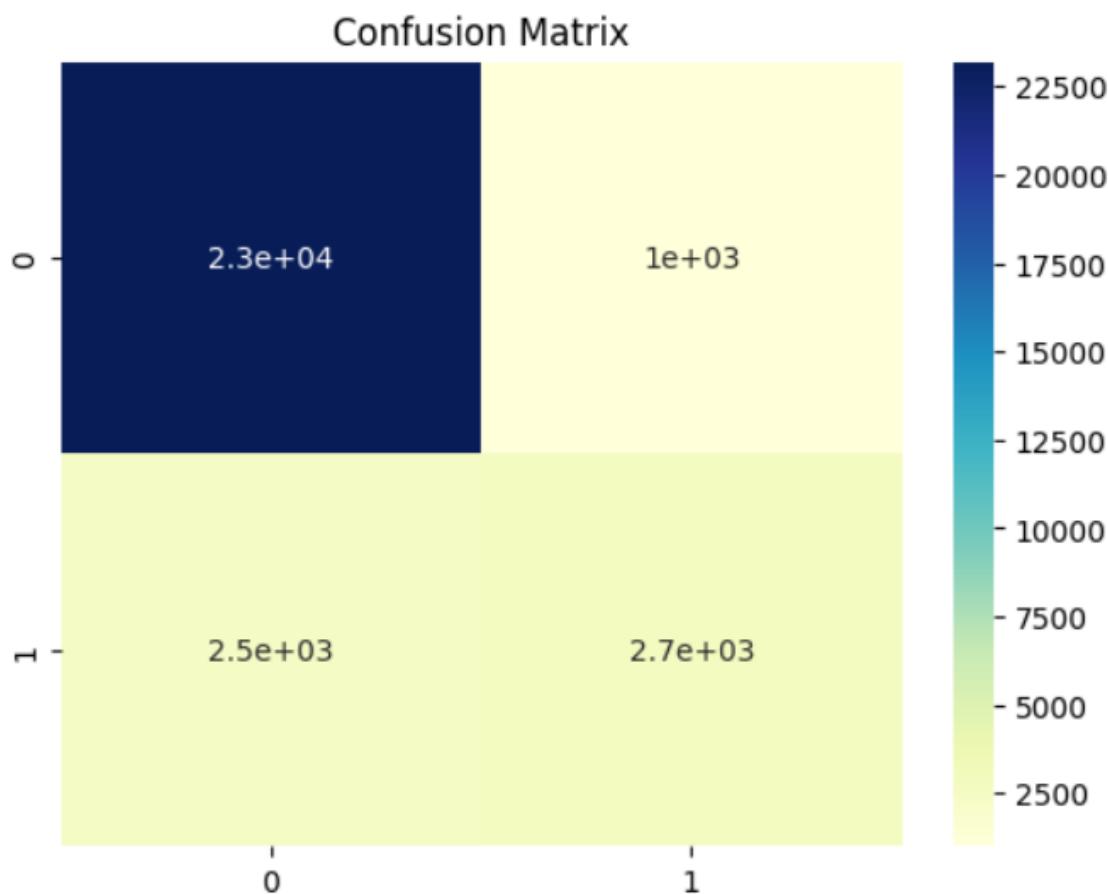
✓ 1.1s

LogisticRegression

```
LogisticRegression(C=1, penalty='l1', random_state=10, solver='liblinear')
```

Đánh giá mô hình:

Accuracy: 0.880



Precision: 0.7270742358078602

Recall (Sensitivity): 0.512998266897747

F1-Score: 0.6015580896466072

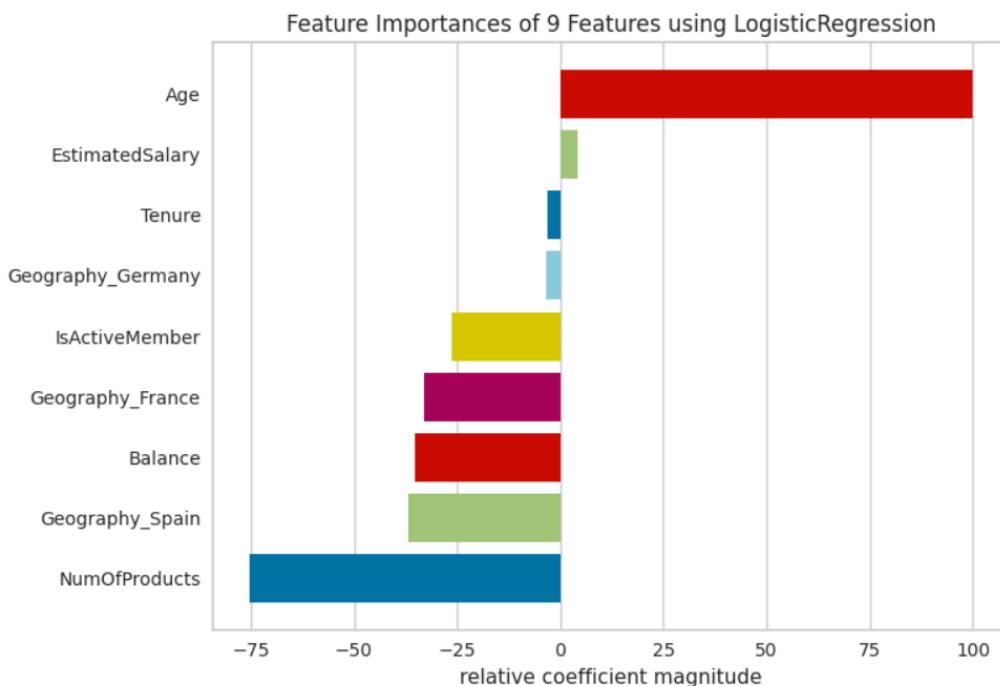
Nhận xét: Precision khá cao, cho thấy rằng khi mô hình dự đoán một khách hàng sẽ rời bỏ ngân hàng, khả năng dự đoán đúng là tương đối cao.

Recall khá thấp, cho thấy mô hình còn bỏ sót khá nhiều khách hàng thực sự rời bỏ ngân hàng.

F1-Score ở mức trung bình, phản ánh rằng mô hình hoạt động tốt hơn trong việc tránh các false positives (dự đoán sai khách hàng sẽ rời bỏ) hơn là tránh các false negatives (bỏ sót khách hàng sẽ rời bỏ).

Nhìn chung, mô hình mới có precision tương đối cao nhưng recall thấp hơn, dẫn đến việc bỏ sót nhiều khách hàng rời bỏ hơn so với mô hình trước đó.

Tầm quan trọng của các thuộc tính đóng góp vào mô hình



=> Các thuộc tính EstimatedSalary, Tenure, Geography_Germany không đóng góp nhiều vào quá trình đưa ra quyết định

4.1.3. Mô hình ANN (kết hợp với SHAP để diễn giải)

Tạo mô hình ANN với 4 lớp ReLU và 1 lớp sigmod làm đầu ra, hàm mất mát là BCEloss, tối ưu dựa trên stochastic gradient descent. Khởi tạo trọng số mean = 0, std = 0.05.

(Bordes và Yoshua Bengio đã đề xuất một giải pháp hiệu quả để giảm vấn đề Vanishing Gradients vào năm 2015 [11]. Phương pháp này được gọi là "Xavier/Glorot initialization" hoặc "Glorot initialization". Ý tưởng cơ bản là khởi tạo trọng số của mạng nơ-ron sao cho giữa các lớp có độ lớn phù hợp, giảm thiểu nguy cơ tiềm cận giá trị 0 hoặc 1 trong quá

trình lan truyền ngược. Cụ thể, các trọng số được khởi tạo ngẫu nhiên từ một phân phối Gaussian với trung bình 0 và phương sai được tính toán sao cho tổng phương sai của đầu ra và đầu vào của mỗi lớp là bằng nhau. Điều này giúp tránh tình trạng khi gradient truyền ngược quá nhỏ hoặc quá lớn qua các lớp, tối ưu hóa hiệu suất của mô hình và giảm vấn đề Vanishing Gradients.)

```

-----  

      Layer (type)          Output Shape       Param #  

=====  

      Linear-1              [-1, 1, 128]        1,280  

      ReLU-2                [-1, 1, 128]        0  

      Linear-3              [-1, 1, 128]        16,512  

      ReLU-4                [-1, 1, 128]        0  

      Linear-5              [-1, 1, 128]        16,512  

      ReLU-6                [-1, 1, 128]        0  

      Linear-7              [-1, 1, 128]        16,512  

      ReLU-8                [-1, 1, 128]        0  

      Linear-9              [-1, 1, 128]        16,512  

      ReLU-10               [-1, 1, 128]        0  

      Linear-11             [-1, 1, 128]        16,512  

      ReLU-12               [-1, 1, 128]        0  

      Linear-13             [-1, 1, 1]          129  

      Sigmoid-14            [-1, 1, 1]          0  

=====  

Total params: 83,969  

Trainable params: 83,969  

Non-trainable params: 0  

-----  

Input size (MB): 0.00  

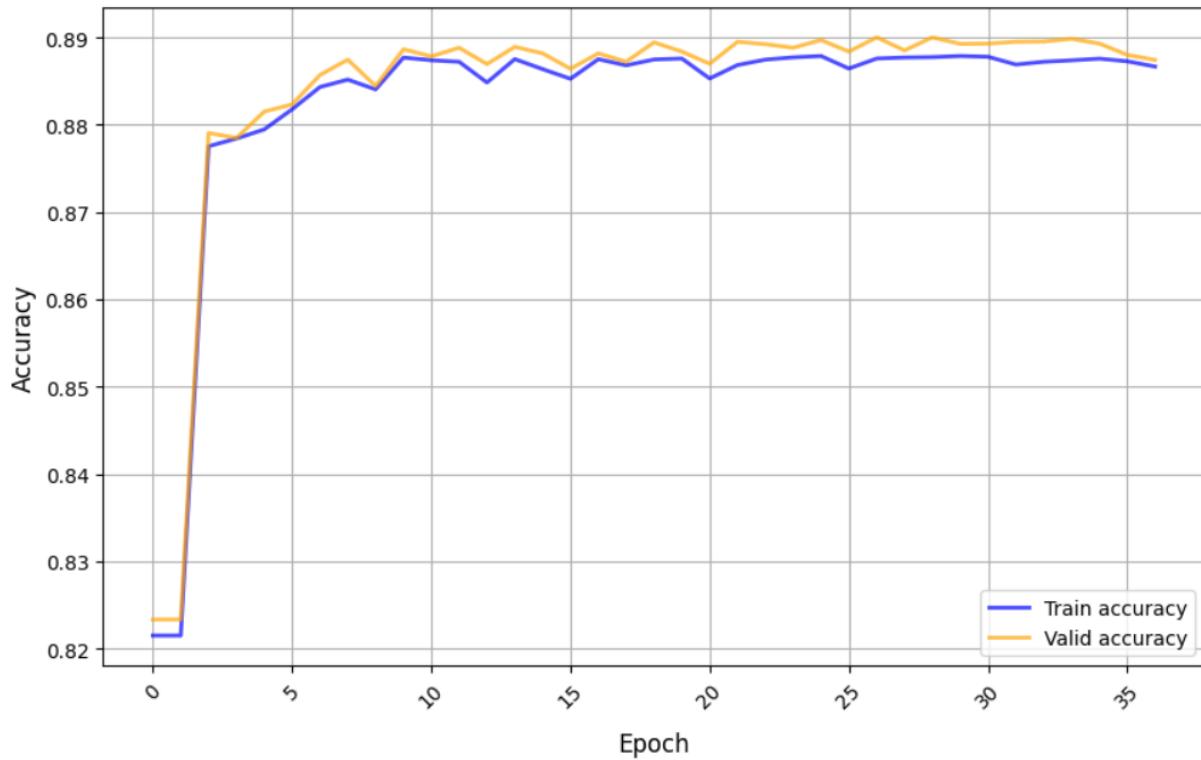
Forward/backward pass size (MB): 0.01  

Params size (MB): 0.32  

Estimated Total Size (MB): 0.33

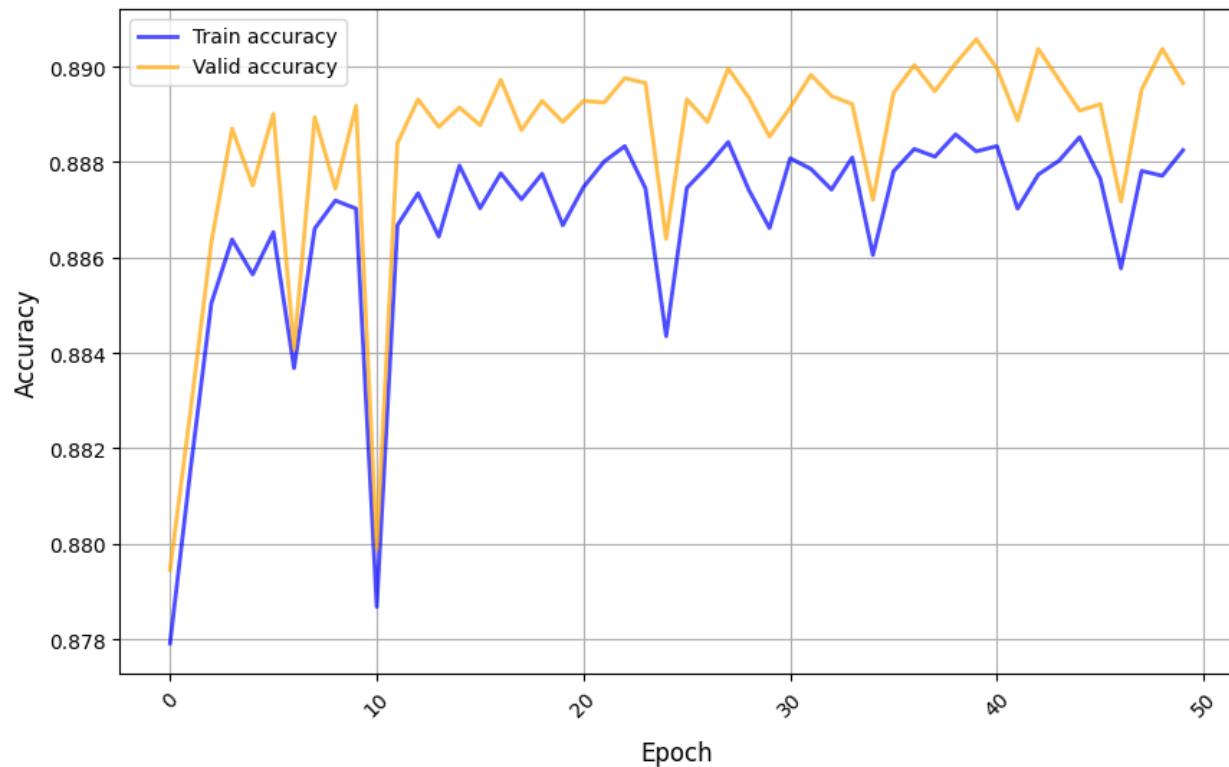
```

Kết quả:



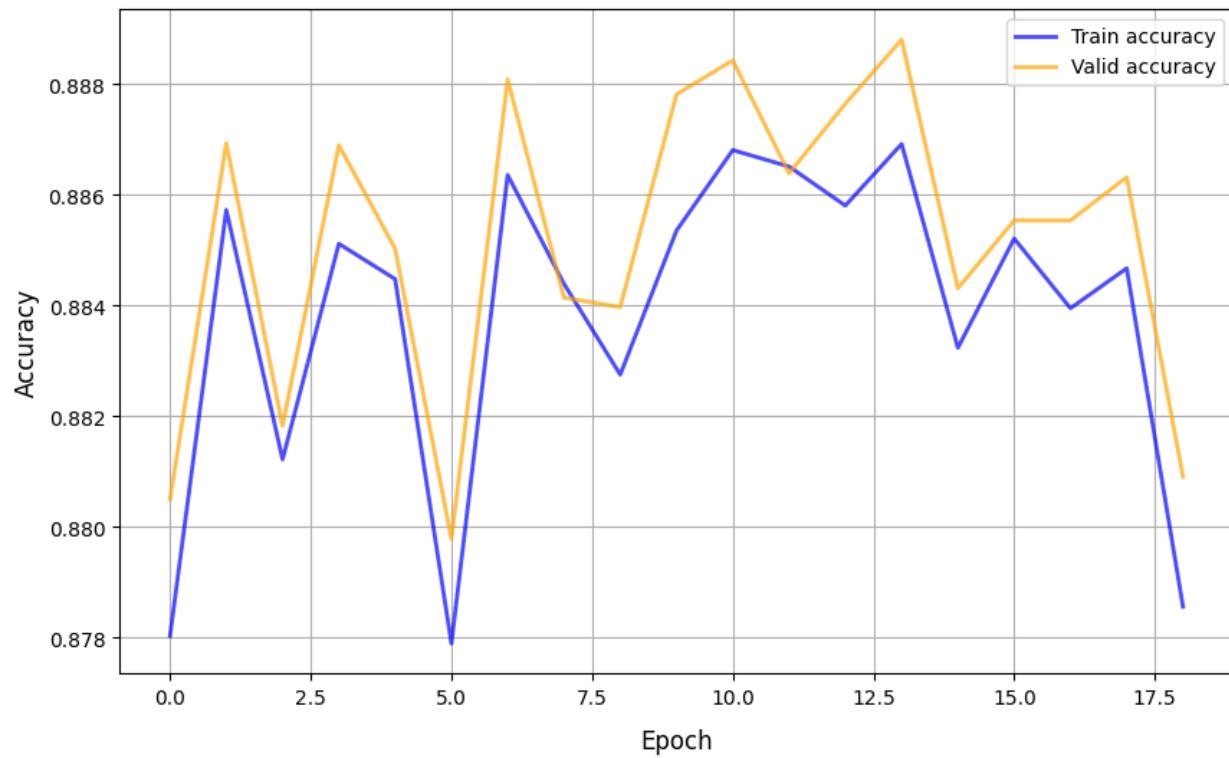
=> Accuracy trên 2 tập train và test dừng ở ngưỡng ~89%

Thử tăng phương sai lên 0.1



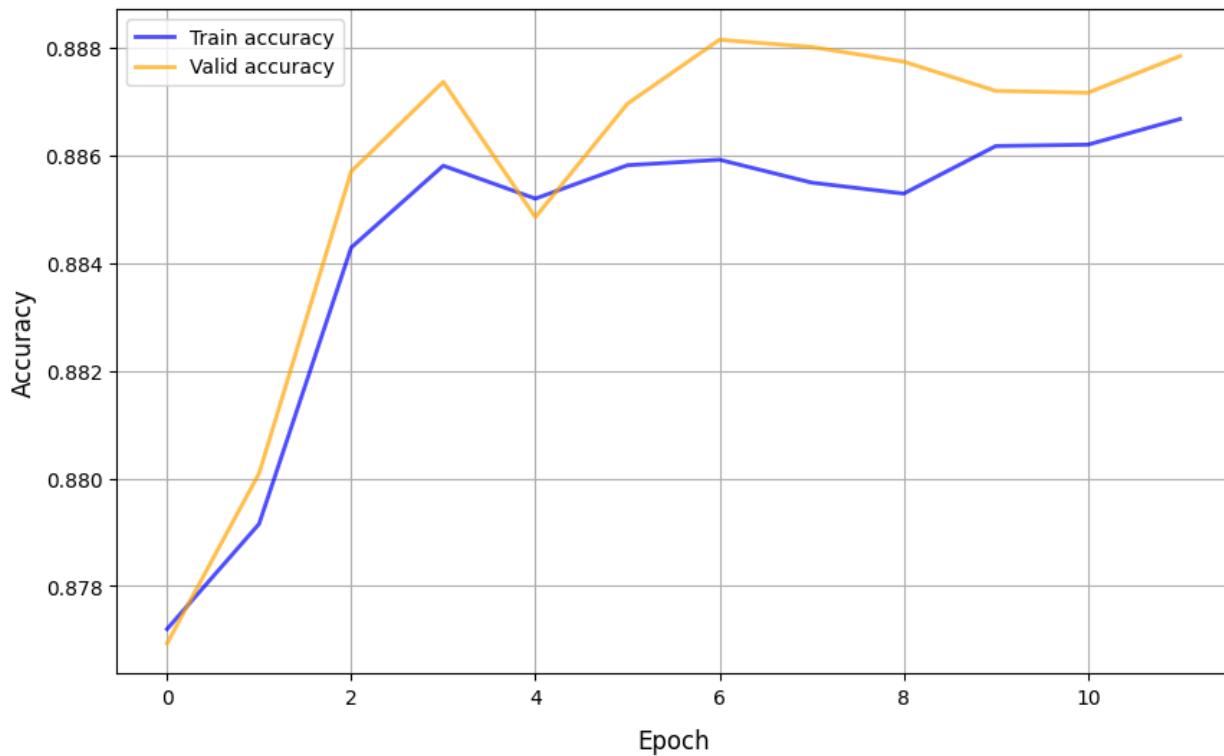
=> Kết quả cũng tương tự

Thử đổi optimizer sang adam và phương sai = 0.1:



=> Kết quả cũng tương tự

Thử đổi hàm kích hoạt từ ReLU sang Tanh:



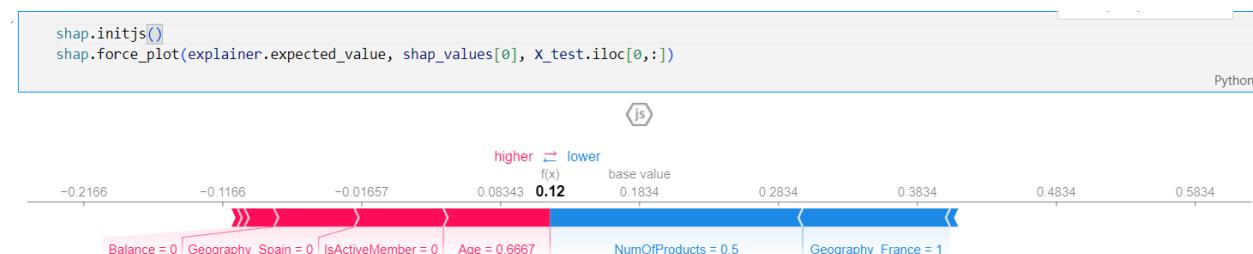
=> Kết quả cũng tương tự

Nhận xét: mô hình đạt ngưỡng accuracy tối đa khoảng ~89%

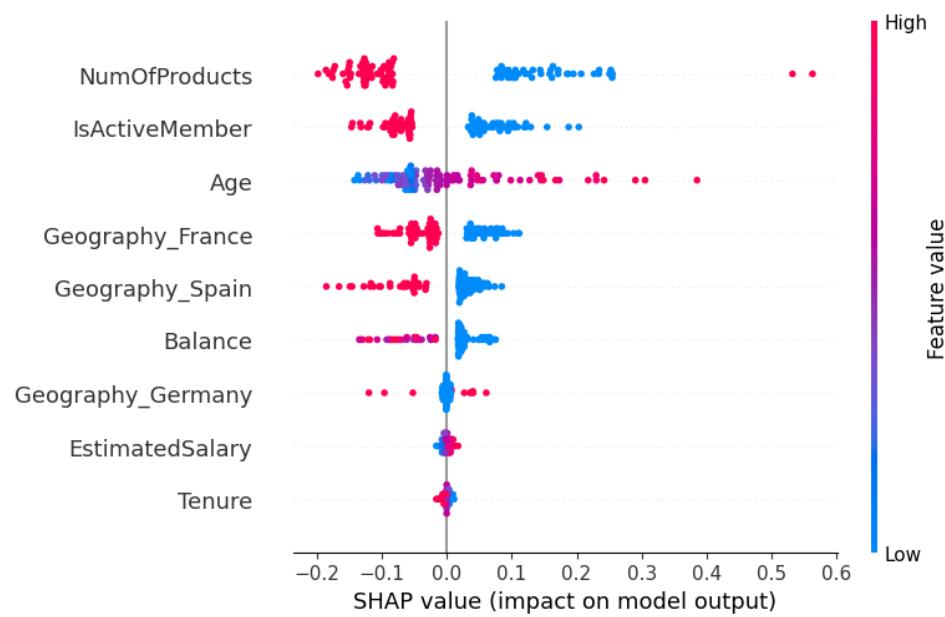
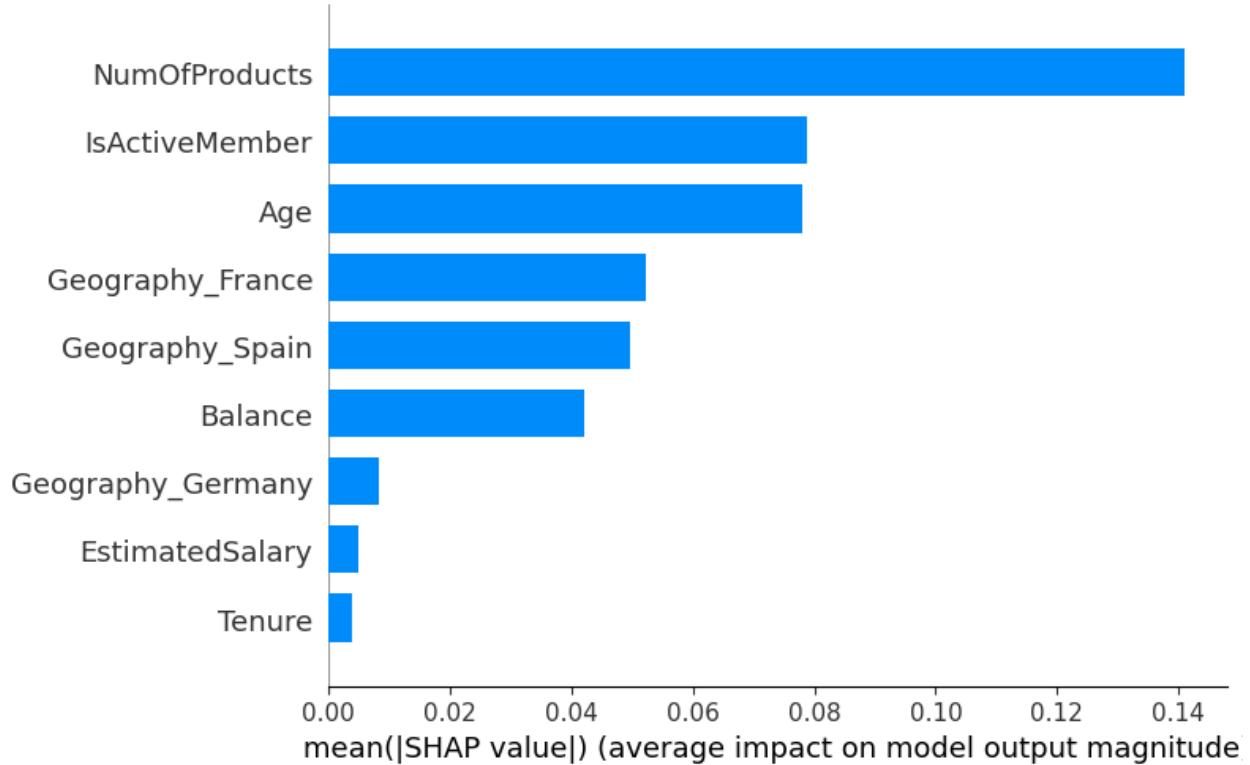
Dùng SHAP để tạo khả năng diễn giải theo mô hình tuyến tính cho ANN (ở đây lấy đầu ra của ANN từ hàm sigmod - giá trị thực trong đoạn [0,1] để giải thích)

Thử diễn giải trên 100 hàng dữ liệu đầu tiên của tập test.

Giải thích cục bộ trên 1 điểm dữ liệu



Giải thích toàn cục trên 100 điểm đã tính toán:



4.2. Nhận xét chung

Decision Tree, LogisticRegression là 2 mô hình khả thi giải dựa trên tập luật hoặc trọng số của thuộc tính, đồng thời cũng mang lại độ chính xác cao ~88%. Còn mô

hình ANN (được sử dụng trong bài toán) mang lại độ chính xác ~89% nhưng khá phức tạp, các hidden layers được xem như ‘black box’, không có khả năng diễn giải. Nhưng ta có thể tiến hành hậu diễn giải để diễn giải kết quả bằng các mô hình post-hoc, ở đây là SHAP.

CHƯƠNG V: ĐÁNH GIÁ VÀ KẾT LUẬN

5.1. Đánh giá

Thành viên	Đóng góp	Mức độ hoàn thành	Ưu điểm	Nhược điểm
Nguyễn Tân Phát - 21133107	Tìm kiếm tập dữ liệu, Trích lọc đặc trưng - Đánh giá mức độ quan trọng của các thuộc tính bằng các mô hình học máy, Xây dựng mô hình.	100%	Tích cực hoàn thành đúng thời gian và các nhiệm vụ được giao, có nhiều ý kiến đóng góp cho đề tài của nhóm.	Do thời gian hạn chế nên còn một số sai sót nhất định, Thiếu thời gian để nghiên cứu và tìm hiểu về đề tài.
Trần Phan Quốc - 21133108	Tìm kiếm tập dữ liệu, Phân tích dữ liệu, Chuẩn bị dữ liệu, Phân tích khám phá dữ liệu - Phân tích đa biến. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến liên tục với biến liên tục.	100%	Tích cực hoàn thành đúng thời gian và các nhiệm vụ được giao, có nhiều ý kiến đóng góp cho đề tài của nhóm.	Do thời gian hạn chế nên còn một số sai sót nhất định, Thiếu thời gian để nghiên cứu và tìm hiểu về đề tài.
Nguyễn Thị Thanh Hiền - 21133032	Tìm kiếm tập dữ liệu, Phân tích khám phá dữ liệu - Phân tích đơn biến. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến phân loại với biến phân loại.	100%	Tích cực hoàn thành đúng thời gian và các nhiệm vụ được giao, có nhiều ý kiến đóng góp cho đề tài của nhóm.	Do thời gian hạn chế nên còn một số sai sót nhất định, Thiếu thời gian để nghiên cứu và tìm hiểu về đề tài.
Tăng Huỳnh Minh Tiên - 21133088	Tìm kiếm tập dữ liệu, Khai phá luật kết hợp, Chuẩn hoá và mã hoá dữ liệu. Trích lọc đặc trưng - Phân tích mối quan hệ giữa biến phân loại với biến liên tục.	100%	Tích cực hoàn thành đúng thời gian và các nhiệm vụ được giao, có nhiều ý kiến đóng góp cho đề tài của nhóm.	Do thời gian hạn chế nên còn một số sai sót nhất định, Thiếu thời gian để nghiên cứu và tìm hiểu về đề tài.

5.2. Kết luận

5.2.1. Kết quả đạt được

Trong quá trình phân tích dữ liệu khách hàng ngân hàng, chúng em đã đạt được những kết quả quan trọng như tiền xử lý dữ liệu hiệu quả, phân tích chi tiết về sự phân bố và mối quan hệ giữa các thuộc tính với trạng thái Churn, cùng với khai phá luật kết hợp và xây dựng mô hình dự đoán. Các mô hình dự đoán như Decision Tree, Logistic Regression và ANN đã cung cấp cái nhìn sâu sắc về các yếu tố ảnh hưởng đến quyết định rời bỏ ngân hàng của khách hàng, giúp cải thiện chiến lược kinh doanh và tiếp thị.

5.2.2. Những hạn chế

Trong quá trình xây dựng mô hình cho tập dữ liệu, chúng em đã phải đối mặt với một số hạn chế. Có sự hạn chế về kiến thức và những kinh nghiệm thực tế. Ngoài ra, tập dữ liệu đầu vào chưa được hoàn chỉnh và vẫn còn nhiều dữ liệu chưa được khai thác hết tiềm năng. Sự thiếu sót trong quy trình triển khai và xây dựng đã làm cho quá trình này không đạt được mức độ hoàn thiện và hiệu quả như mong đợi.

TÀI LIỆU THAM KHẢO

1. Walter Reade, Ashley Chow. (2024). Binary Classification with a Bank Churn Dataset . Kaggle. <https://kaggle.com/competitions/playground-series-s4e1>
2. Tarandeep Singh 4 (2023). Hyperparameter tuning. geeksforgeeks. Truy cập ngày 30 tháng 05 năm 2024: <https://www.geeksforgeeks.org/hyperparameter-tuning>
3. Phạm Ngọc Giàu, Tống Lê Thanh Hải (12/2023), VẤN ĐỀ VANISHING GRADIENT VÀ CÁC PHƯƠNG PHÁP XỬ LÝ KHI LAN TRUYỀN NGUỒN TRONG HUÂN LUYỆN MÔ HÌNH HỌC SÂU. Tạp chí Khoa học và Công nghệ. Truy cập ngày 30 tháng 05 năm 2024: <https://jst-hau.edu.vn/media/31/uffile-upload-no-title31303.pdf>