Chapter 5 - Ex4: Customer Segmentation

Cho dữ liệu ageinc_g.csv chứa thông tin khách hàng.

- Phân tích thông tin sơ bộ về dữ liệu trên các thuộc tính income, age, gender. Xem xét mối quan hệ giữa hai thuộc tính là income và age. Trực quan hóa dữ liệu.
- Để thực hiện bài toán Customer Segmentation (Phân cụm/ nhóm khách hàng) cần phải kiểm tra và chuẩn hóa dữ liệu. Hãy chọn một phương pháp để chuẩn hóa dữ liệu dựa trên thông tin nêu trên. Trực quan hóa dữ liệu sau khi chuẩn hóa.

Gợi ý

In [2]:

```
# Tải các thư viện cần thiết
import pandas as pd
import numpy as np
import numpy as numpy
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler, StandardScaler
```

In [3]:

```
# Đọc dữ liệu. Tìm hiểu thông tin sơ bộ về dữ liệu
data = pd.read_csv('ageinc_g.csv', index_col=0)
```

In [4]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000 entries, 0 to 999
Data columns (total 3 columns):
income    1000 non-null int64
age       1000 non-null int64
gender    1000 non-null object
dtypes: int64(2), object(1)
memory usage: 31.2+ KB
```

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN TIN HỌC

In [5]:

```
data.head()
```

Out[5]:

	income	age	gender
0	101743	58	Female
1	49597	27	Female
2	36517	52	Male
3	33223	49	Male
4	72994	53	Female

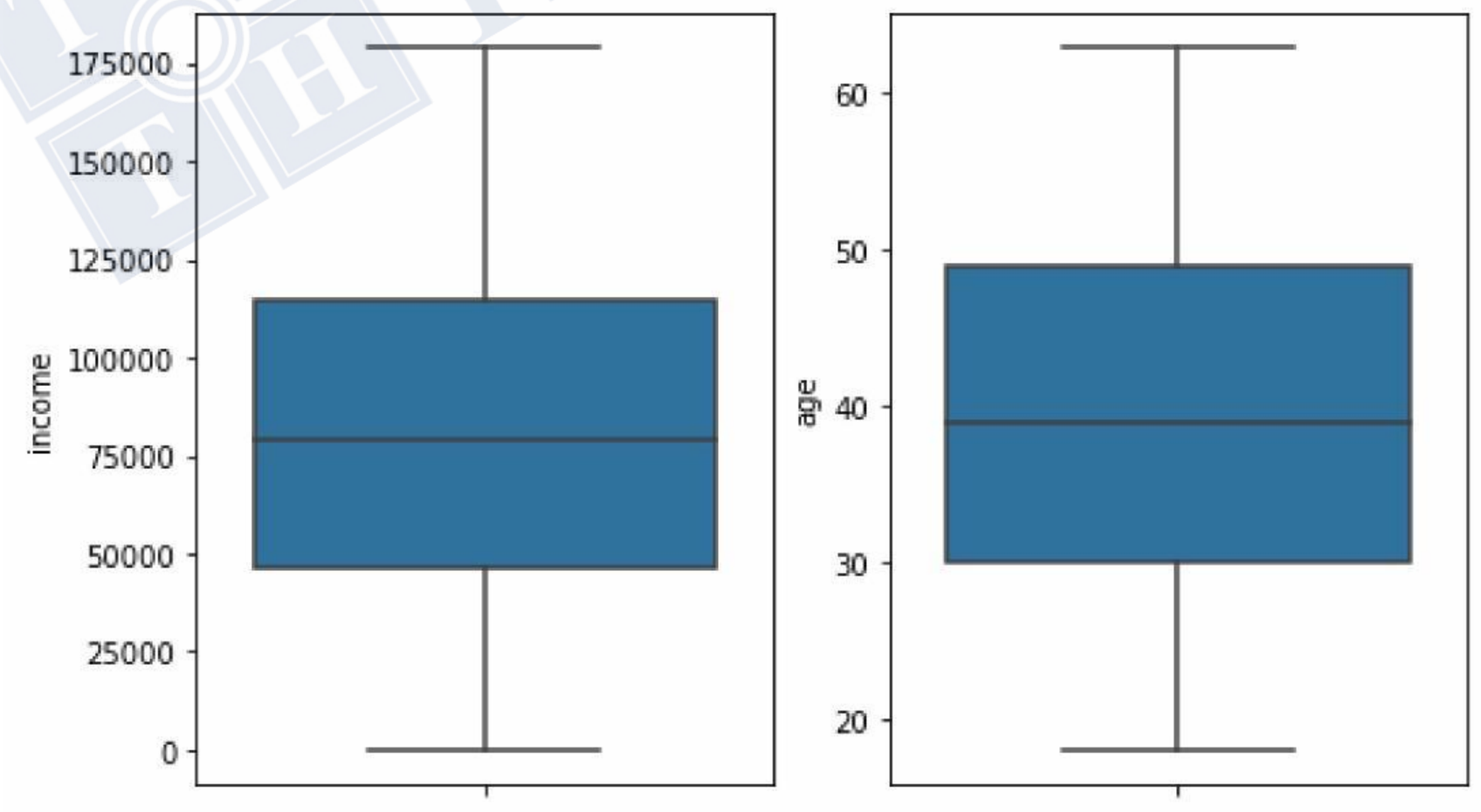
In [6]:

```
# Kiểm tra dữ liệu null
print(data.isnull().sum())
# => Không có dữ liệu null
```

```
income    0
age       0
gender    0
dtype: int64
```

In [7]:

```
# Phân tích đơn biến: trực quan hóa, kiểm tra dữ liệu outlier
# Trực quan hóa dữ liệu cho từng biến liên tục
plt.figure(figsize=(8,5))
plt.subplot(1,2,1)
sns.boxplot(data.income, orient="v")
plt.subplot(1,2,2)
sns.boxplot(data.age, orient="v")
plt.show()
# => Cả hai biến liên tục income và age đều không có outlier
```



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [8]:

```
# Dataset có cột gender: là thuộc tính phân loại kiểu chuỗi => tạo cột phân loại kiểu số
data = pd.get_dummies(data, drop_first=True)
```

In [9]:

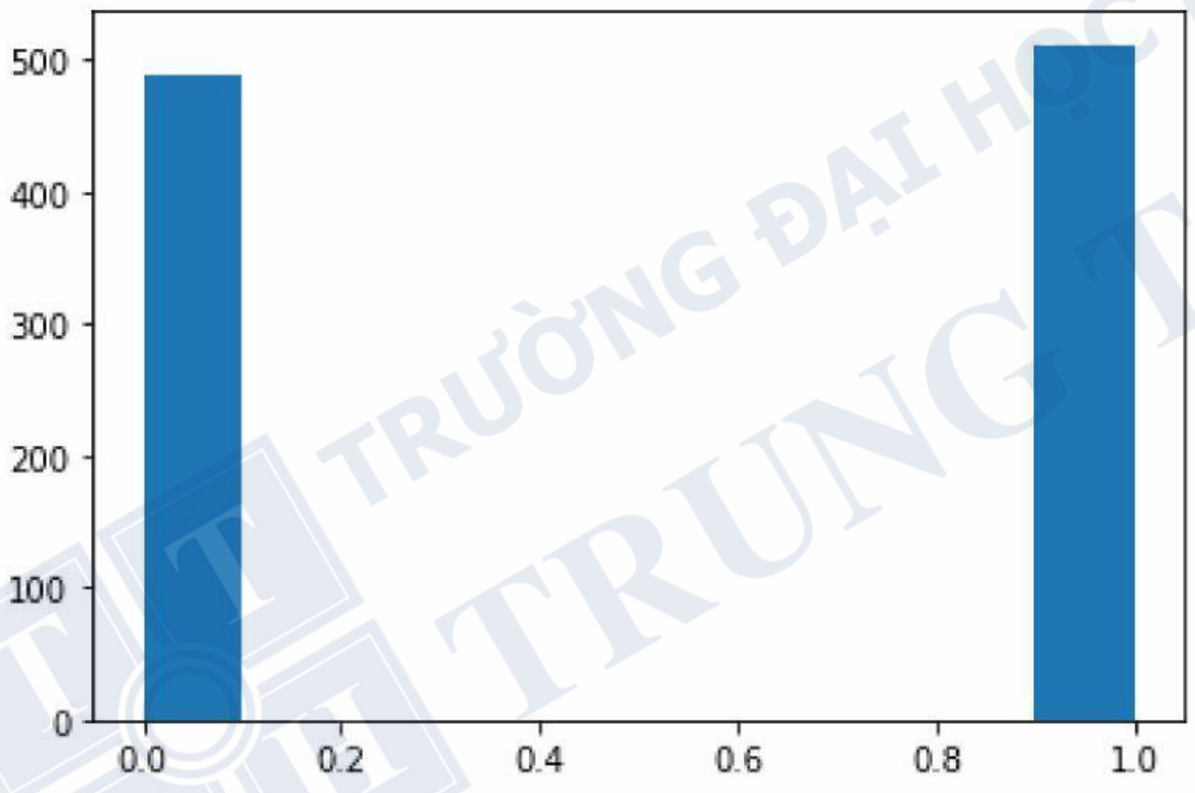
```
data.head()
```

Out[9]:

	income	age	gender_Male
0	101743	58	0
1	49597	27	0
2	36517	52	1
3	33223	49	1
4	72994	53	0

In [10]:

```
plt.hist(data.gender_Male)
plt.show()
```

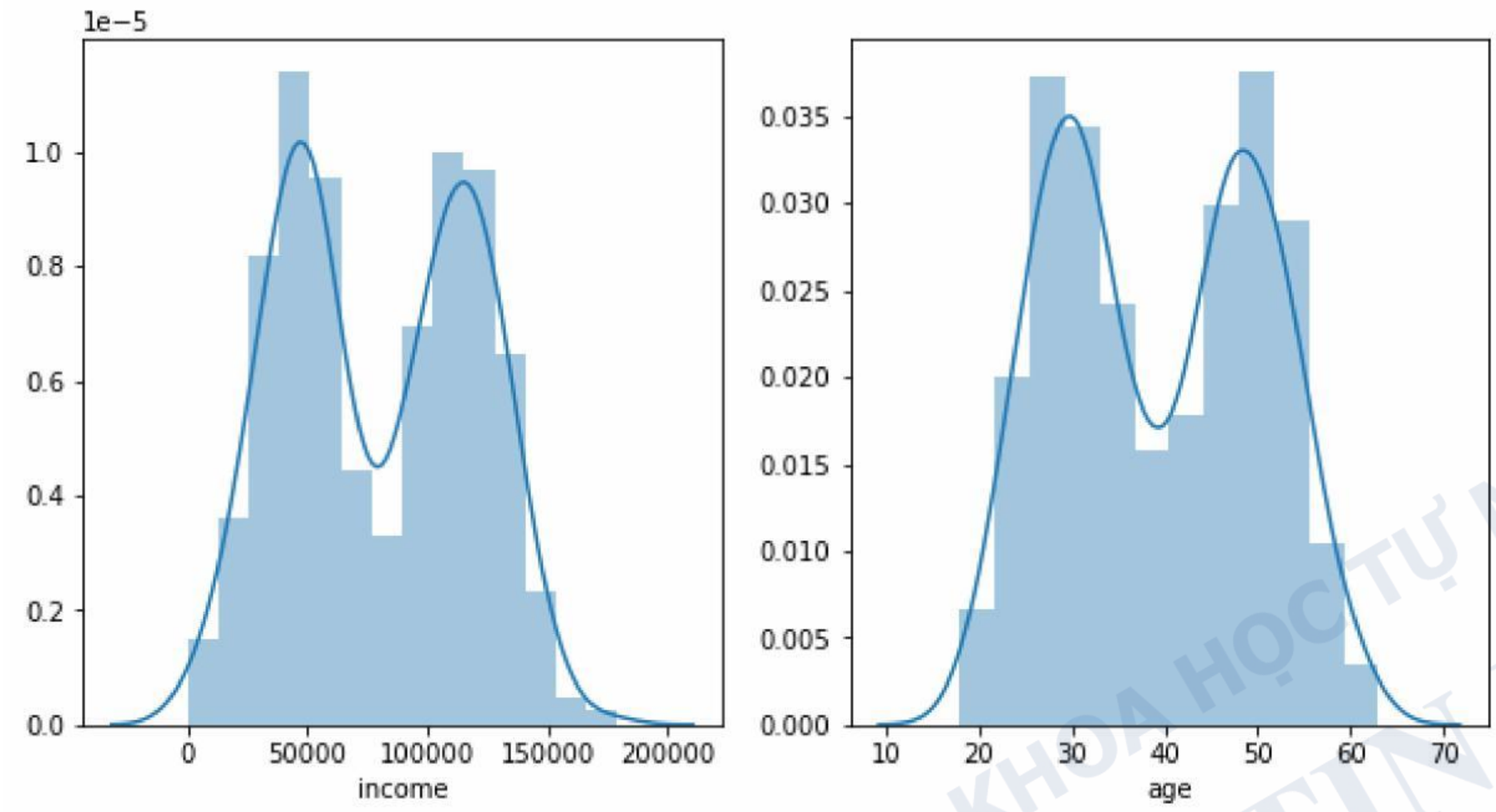


- Customer Segmentation là một bài toán phân nhóm dựa trên sự tương tự về các thuộc tính và sẽ tính khoảng cách để biết mẫu này “gần” mẫu kia hay không bằng công thức tính khoảng cách Euclide
- Đối với các thuật toán cần tính khoảng cách thì dữ liệu trên các cột phải đảm bảo tính công bằng khi tính toán. Tuy nhiên, quan sát thấy income và age có range dữ liệu khác biệt, chênh lệch nhiều => cần phải chuẩn hóa dữ liệu.
- Muốn chuẩn hóa dữ liệu: cần kiểm tra phân phối của dữ liệu.

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [11]:

```
plt.figure(figsize=(10,5))
plt.subplot(1, 2, 1)
sns.distplot(data.income)
plt.subplot(1, 2, 2)
sns.distplot(data.age)
plt.show()
```



In [12]:

```
data.skew()
```

Out[12]:

```
income         0.028753
age            0.049110
gender_Male   -0.044077
dtype: float64
```

In [13]:

```
data.kurtosis()
```

Out[13]:

```
income        -1.210079
age           -1.262649
gender_Male   -2.002065
dtype: float64
```

Từ những kết quả trên ta thấy:

- Dữ liệu không theo phân phối Gaussian
- Dữ liệu không có outlier
  - => Dùng MinMaxScaler để chuẩn hóa

MinMaxScaler

In [15]:

```
mmScaler = MinMaxScaler()
mmScaler.fit(data[['income', 'age']])
data_sub = mmScaler.transform(data[['income', 'age']])
```

In [16]:

```
data_sub
```

Out[16]:

```
array([[0.56746461, 0.88888889],
       [0.27662387, 0.2       ],
       [0.20367107, 0.75555556],
       ...,
       [0.23538434, 0.37777778],
       [0.0946769 , 0.28888889],
       [0.69080393, 0.57777778]])
```

In [17]:

```
data_sub_min_max_scaler = pd.DataFrame(data_sub,
                                       columns=['mm_income', 'mm_age'])
data = pd.concat([data.reset_index(drop=True),
                  data_sub_min_max_scaler], axis=1)
```

In [18]:

```
data.head()
```

Out[18]:

	income	age	gender_Male	mm_income	mm_age
0	101743	58	0	0.567465	0.888889
1	49597	27	0	0.276624	0.200000
2	36517	52	1	0.203671	0.755556
3	33223	49	1	0.185299	0.688889
4	72994	53	0	0.407119	0.777778

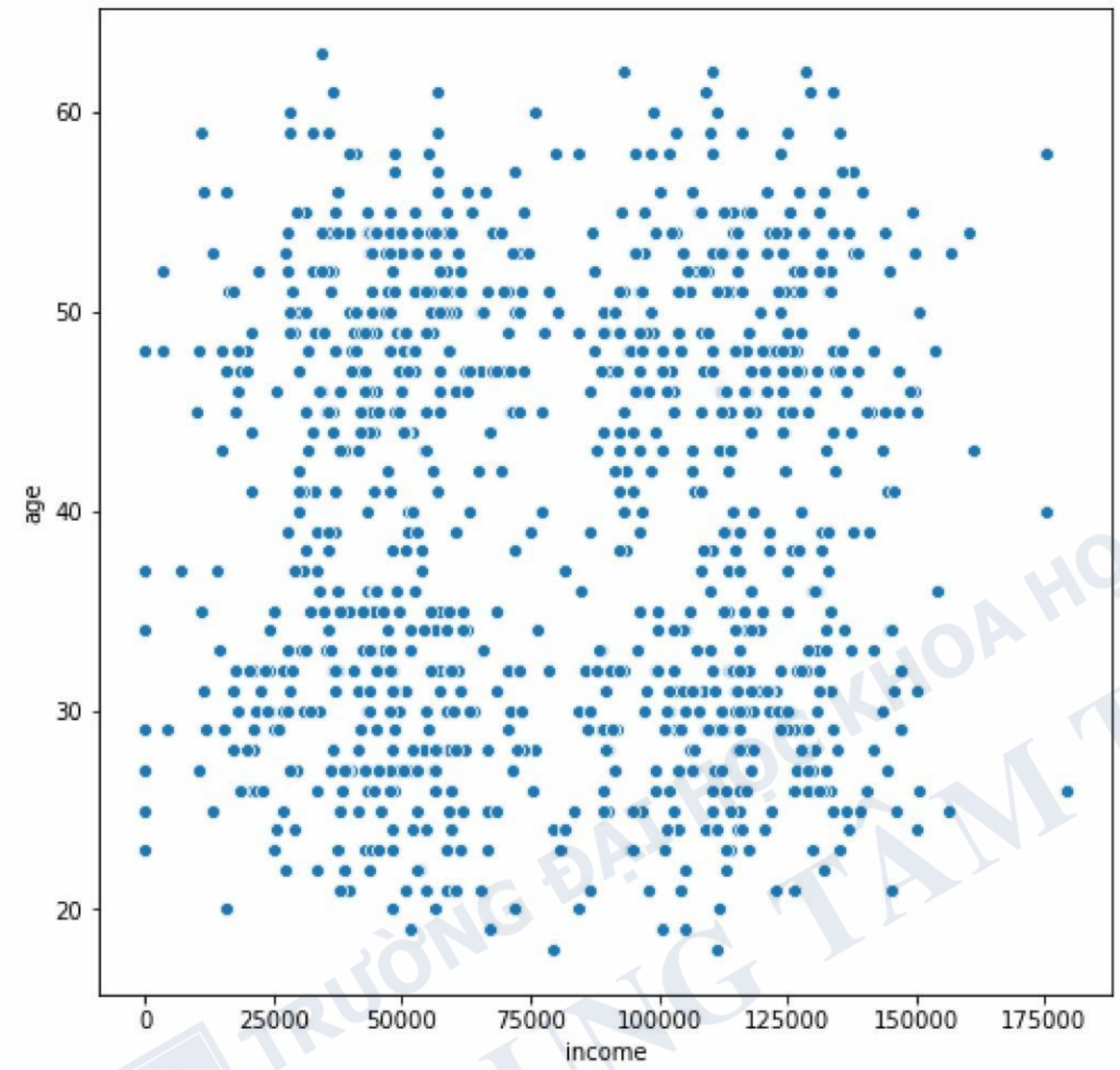
In [19]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 5 columns):
income          1000 non-null int64
age             1000 non-null int64
gender_Male     1000 non-null uint8
mm_income       1000 non-null float64
mm_age          1000 non-null float64
dtypes: float64(2), int64(2), uint8(1)
memory usage: 32.4 KB
```

In [24]:

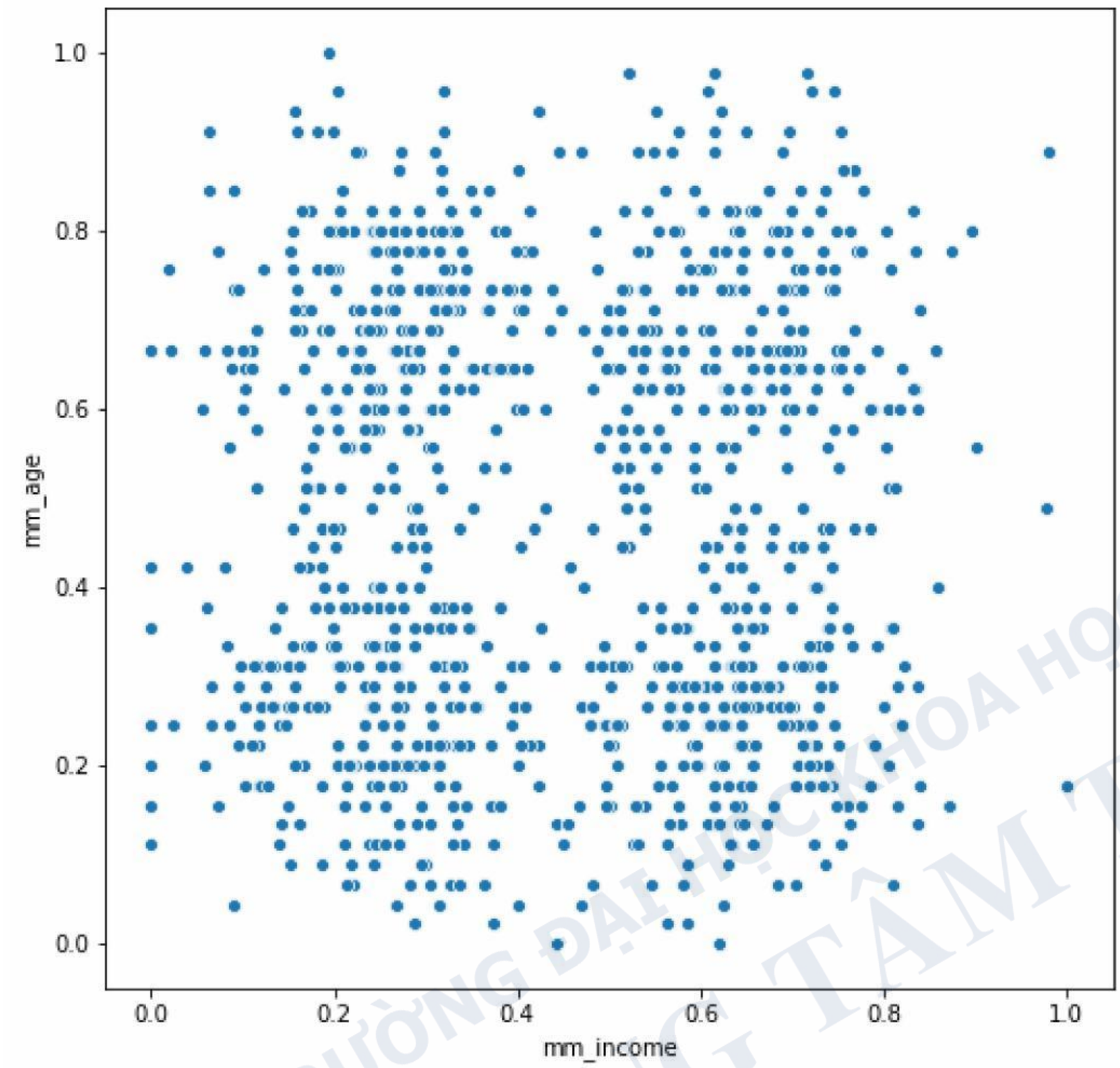
```
# Income & Age: trước khi Scaler
plt.figure(figsize=(8,8))
sns.scatterplot(data=data, x='income', y='age')
plt.show()
```



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC

In [26]:

```
# mm_income & mm_age: sau khi Scaler giá trị nằm trong khoảng [0,1]
plt.figure(figsize=(8,8))
sns.scatterplot(data=data, x='mm_income', y='mm_age')
plt.show()
```



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TRUNG TÂM TIN HỌC