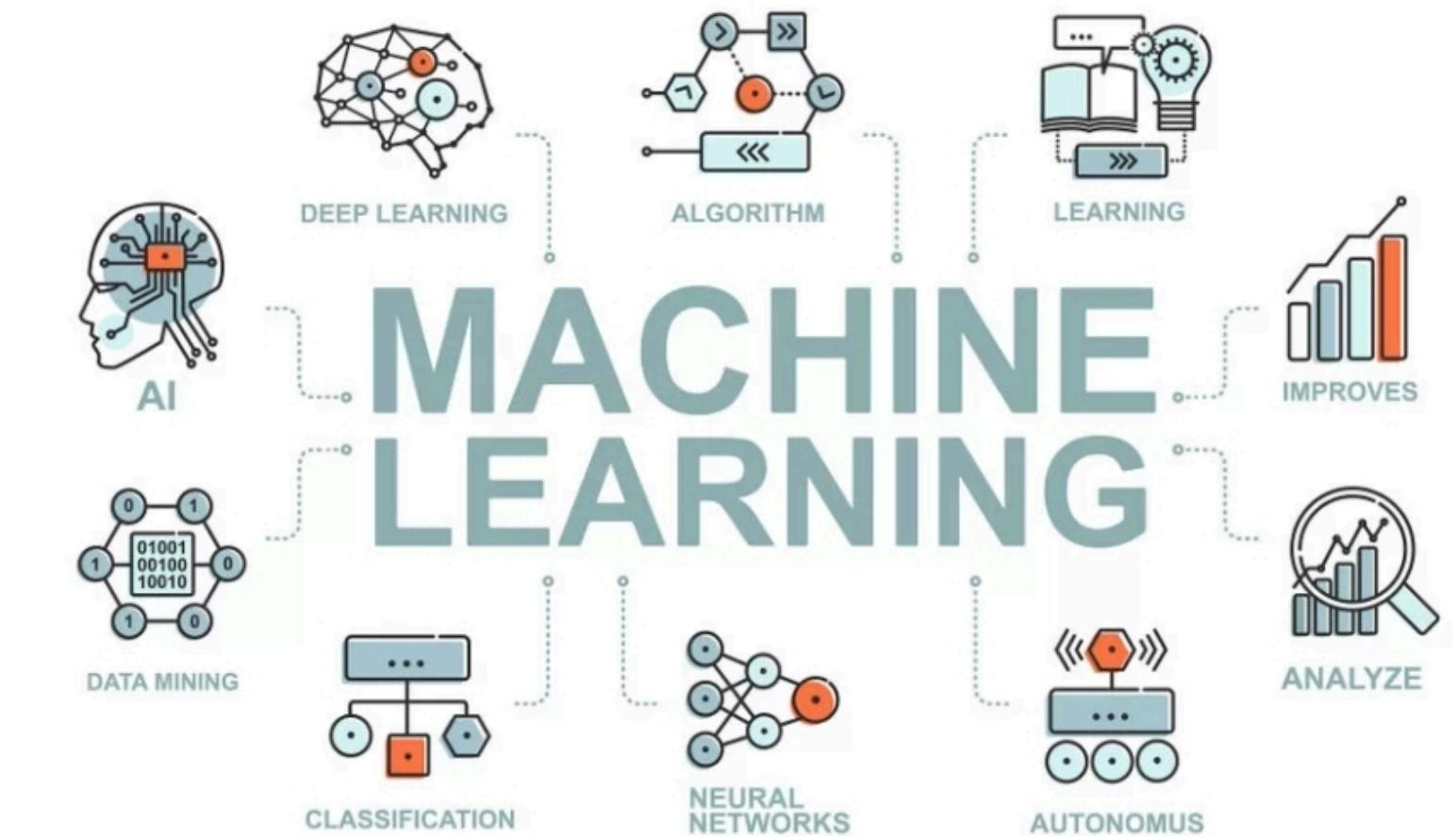


# XÂY DỰNG HỆ THỐNG DỰ ĐOÁN HỌC SINH BỎ HỌC GIỮA CHÙNG

Ứng dụng Machine Learning trong việc xây dựng hệ  
thống cảnh báo sớm

Người thực hiện: Dương Quốc Huy

Đặng Trung Hiếu



## Mục lục

- 1.Giới thiệu đề tài
- 2.Lí do chọn dataset
- 3.Tiền xử lí dữ liệu
- 4.Các mô hình training
- 5.Kết quả và demo

# Giới thiệu đê tài

# Lí do chọn đê tài

- Tỷ lệ sinh viên bỏ học (dropout) là vấn đề lớn của giáo dục đại học. Theo UNESCO và OECD, tỷ lệ này trung bình từ 20-40% tùy quốc gia.
- Dù chưa có thống kê chính thức toàn quốc, nhưng thực tế cho thấy rất nhiều sinh viên bỏ học do kinh tế, áp lực học tập, sức khỏe tâm lý, hoặc nhập học muộn (đi làm rồi mới đi học lại).
- Các trường thường chỉ phát hiện khi đã có dấu hiệu quá rõ ràng: nghỉ học nhiều, nợ học phí lâu, điểm quá thấp. Lúc này việc can thiệp thường đã muộn và hiệu quả thấp.



# Các hệ lụy khi một sinh viên bỏ học

- Mất cơ hội có bằng cấp, dễ rơi vào 'vòng luẩn quẩn' nợ học phí và khó tìm được việc làm ổn định trong tương lai.
- Lãng phí nguồn lực đào tạo đã đầu tư, bị giảm chỉ tiêu tuyển sinh cho các năm sau và ảnh hưởng trực tiếp đến uy tín.
- Mất đi một nguồn nhân lực chất lượng cao tiềm năng, ảnh hưởng đến năng suất lao động chung.



# Lí do chọn dataset

# Giới thiệu tập dữ liệu

## Tổng quan

Nguồn: dataset.csv

- Kích thước ban đầu: Số mẫu: 4.424 sinh viên
- Số đặc trưng(features): 35
- Nhân gốc: Dropout (32.12%), Graduate (49.93%), Enrolled (17.95%)
- Em gộp Graduate + Enrolled thành Non-Dropout để tập trung dự đoán “nguy cơ bỏ học”.

Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Nacionality	Mother's qualification	Father's qualification	Mother's occupation	Curricular units 2nd sem (credited)	Curricular units 2nd sem (enrolled)
0	1	8	5	2	1	1	1	13	10	6	0
1	1	6	1	11	1	1	1	3	4	...	0
2	1	1	5	5	1	1	1	22	27	10	...
3	1	8	2	15	1	1	1	23	27	6	0
4	2	12	1	3	0	1	1	22	28	10	0
...	...	...	...	...	...	...	...	...	...	...	...
4419	1	1	6	15	1	1	1	1	1	6	0
4420	1	1	2	15	1	1	19	1	1	10	0
4421	1	1	1	12	1	1	1	22	27	10	0
4422	1	1	1	9	1	1	1	22	27	8	0
4423	1	5	1	15	1	1	9	23	27	6	0

Mother's qualification	Father's qualification	Mother's occupation	Curricular units 2nd sem (credited)	Curricular units 2nd sem (enrolled)	Curricular units 2nd sem (evaluations)	Curricular units 2nd sem (approved)	Curricular units 2nd sem (grade)	Curricular units 2nd sem (without evaluations)	Unemployment rate	Inflation rate	GDP	Target
13	10	6	...	0	0	0	0.000000	0	10.8	1.4	1.74	Dropout
1	3	4	...	0	6	6	13.666667	0	13.9	-0.3	0.79	Graduate
22	27	10	...	0	6	0	0.000000	0	10.8	1.4	1.74	Dropout
23	27	6	...	0	6	10	5 12.400000	0	9.4	-0.8	-3.12	Graduate
22	28	10	...	0	6	6	13.000000	0	13.9	-0.3	0.79	Graduate
...	...	...	...	...	...	...	...	...	...	...	...	...
1	1	6	...	0	6	8	5 12.666667	0	15.5	2.8	-4.06	Graduate
1	1	10	...	0	6	6	2 11.000000	0	11.1	0.6	2.02	Dropout
22	27	10	...	0	8	9	1 13.500000	0	13.9	-0.3	0.79	Dropout
22	27	8	...	0	5	6	5 12.000000	0	9.4	-0.8	-3.12	Graduate
23	27	6	...	0	6	6	6 13.000000	0	12.7	3.7	-1.70	Graduate

# Mục tiêu của bài toán

Xây dựng mô hình dự đoán nhị phân

- 1 = Dropout: Sinh viên có nguy cơ cao bỏ học → cần hỗ trợ ưu tiên.
- 0 = Non-Dropout: Graduate hoặc Enrolled → ổn định.

Xử lý mất cân bằng nhẹ

- Supervised Binary Classification (lớp thiểu số ~32% → mất cân bằng nhẹ).



Vậy mất cân bằng lớp là gì và tại sao nó phổ biến trong Binary Classification?

- Trong binary classification, thường có 2 lớp: lớp đa số (majority class) và lớp thiểu số (minority class).
- Mất cân bằng xảy ra khi một lớp chiếm tỷ lệ áp đảo (68% Non-Dropout vs 32% Dropout).

Tại sao không xử lý mất cân bằng lại dẫn đến mô hình “ảo” (deceptive)?

Ví dụ:

- Nếu mô hình “ngu ngốc” luôn dự đoán “Non-Dropout” (lớp 0) cho mọi sinh viên → Accuracy = 68% (rất cao!).
- Nhưng Recall Dropout = 0% → mô hình KHÔNG PHÁT HIỆN ĐƯỢC BẤT KỲ sinh viên nào có nguy cơ bỏ học → hoàn toàn vô dụng trong thực tế!

→ Accuracy cao nhưng mô hình thất bại hoàn toàn với mục tiêu chính (phát hiện nguy cơ bỏ học).

Hậu quả nếu không xử lý	Lợi ích khi xử lý đúng cách
Mô hình thiên vị lớp đa số	Mô hình học được pattern của cả hai lớp
Recall lớp thiểu số rất thấp (bỏ sót nhiều)	Recall lớp thiểu số tăng → phát hiện nhiều nguy cơ hơn
Precision có thể cao nhưng không có ý nghĩa thực tế	Precision và Recall cân bằng hơn → báo động giả giảm
Đánh giá sai (Accuracy lừa dối)	Dùng metric phù hợp: F1, ROC-AUC, Precision-Recall Curve

## Tại sao đây là yếu tố QUAN TRỌNG NHẤT?

- Không xử lý mâu thuẫn → mô hình gần như vô dụng dù Accuracy cao.
- Xử lý đúng → mô hình có giá trị thực tiễn cao, giúp nhà trường/cơ quan can thiệp đúng đối tượng, tiết kiệm nguồn lực.

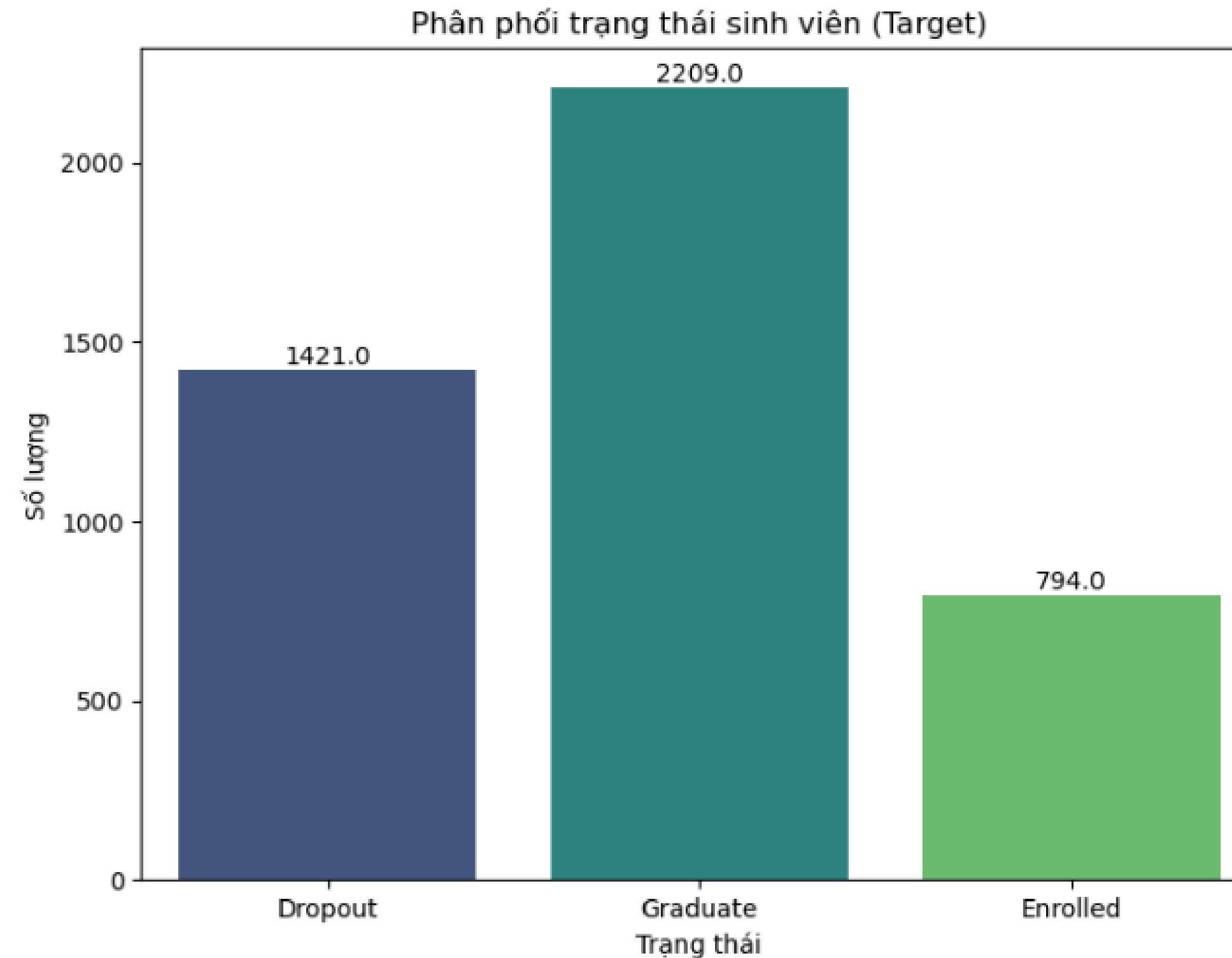
# Các kỹ thuật xử lý mất cân bằng em sử dụng

Kỹ thuật	Cách hoạt động	Áp dụng trong bài
SMOTE	Oversampling: tạo mẫu synthetic cho lớp thiểu số	Dùng cho Logistic Regression và KNN

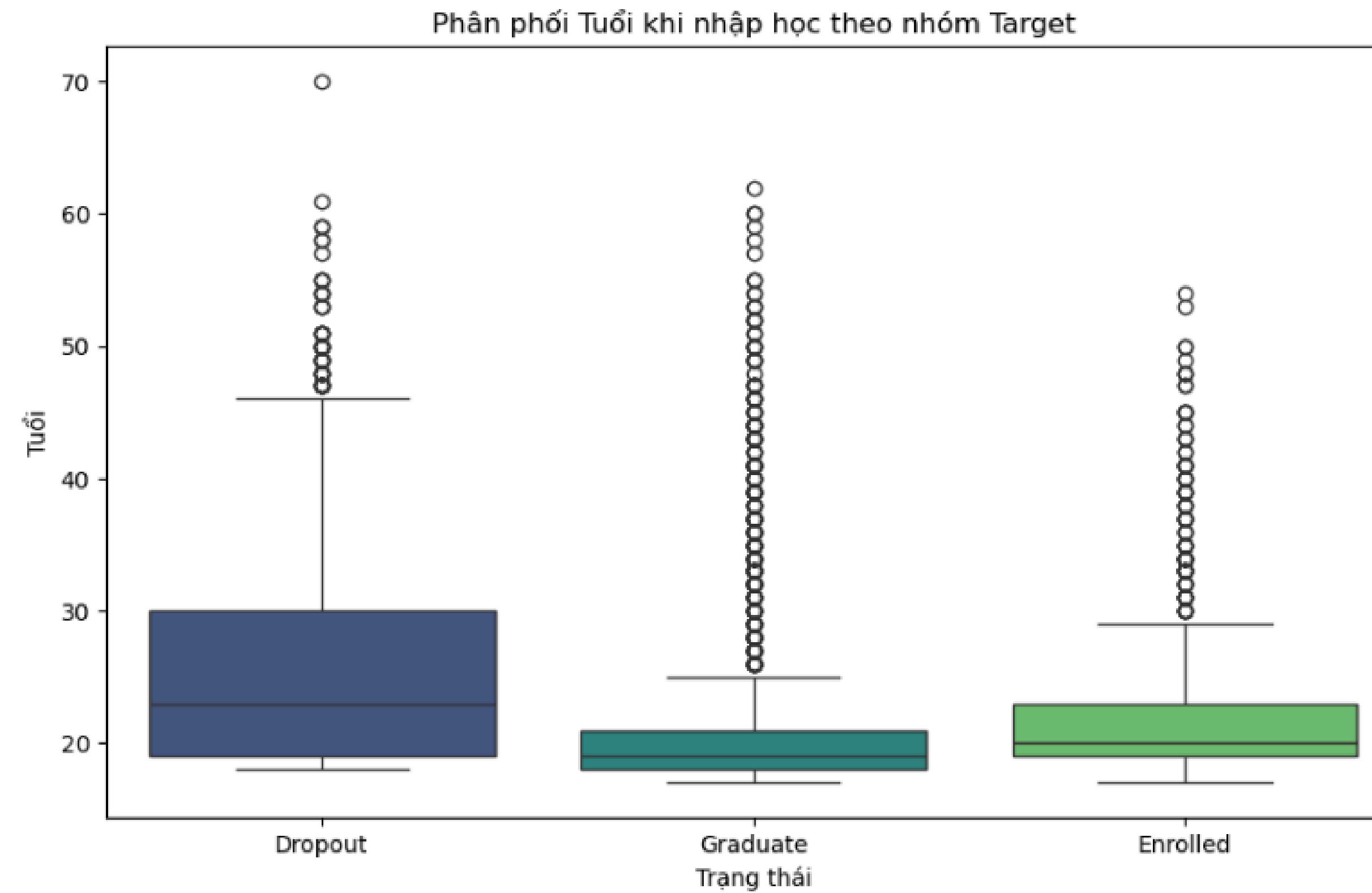
- Mục tiêu chính là phát hiện tối đa sinh viên có nguy cơ bỏ học (Recall Dropout cao).
- Đồng thời tránh báo động giả quá nhiều (Precision Dropout cao) để nhà trường không lãng phí nguồn lực hỗ trợ.

# Trực quan hóa dữ liệu

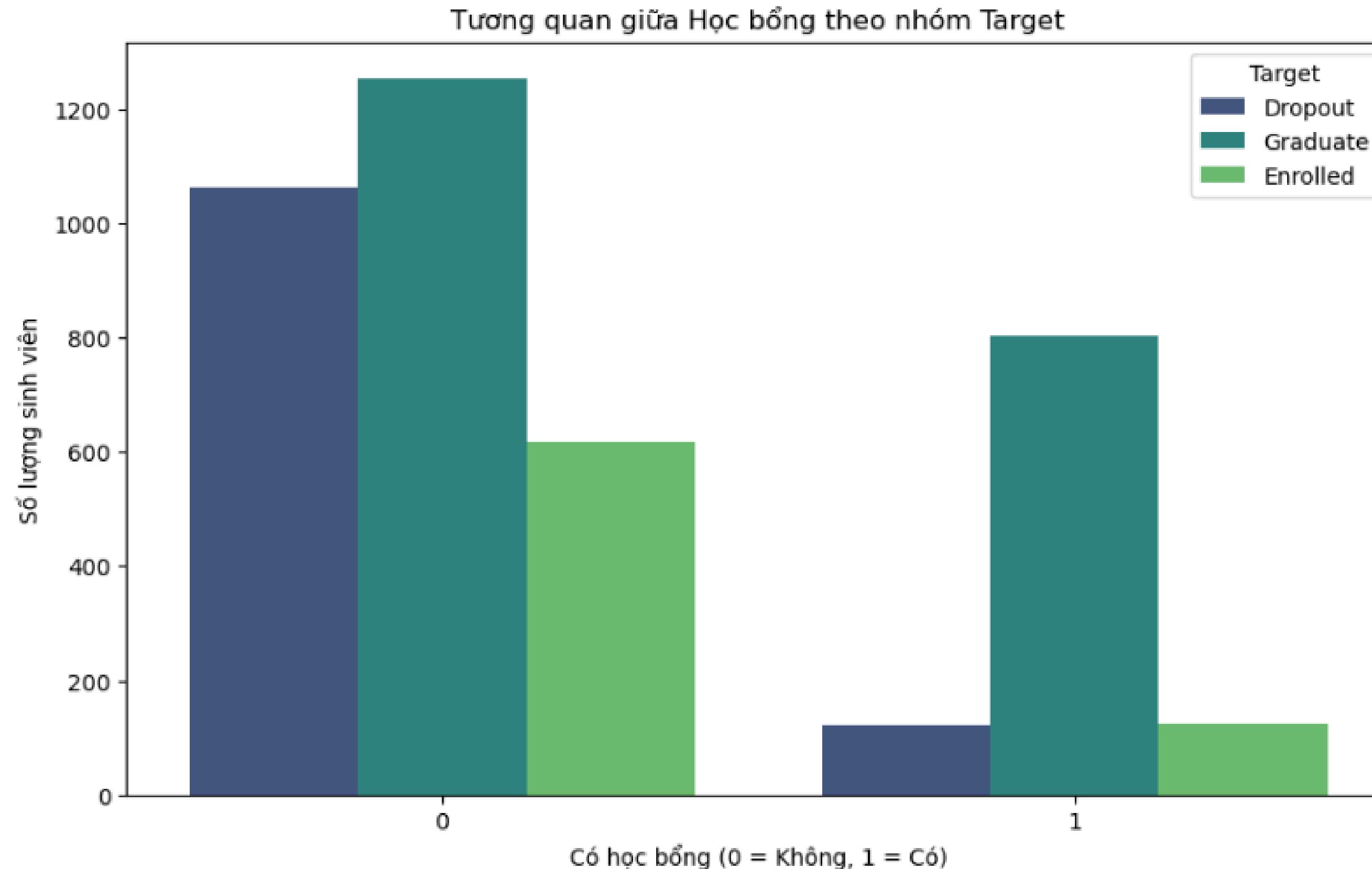
## 1. Biểu đồ phân phối học sinh bỏ học hay không bỏ học



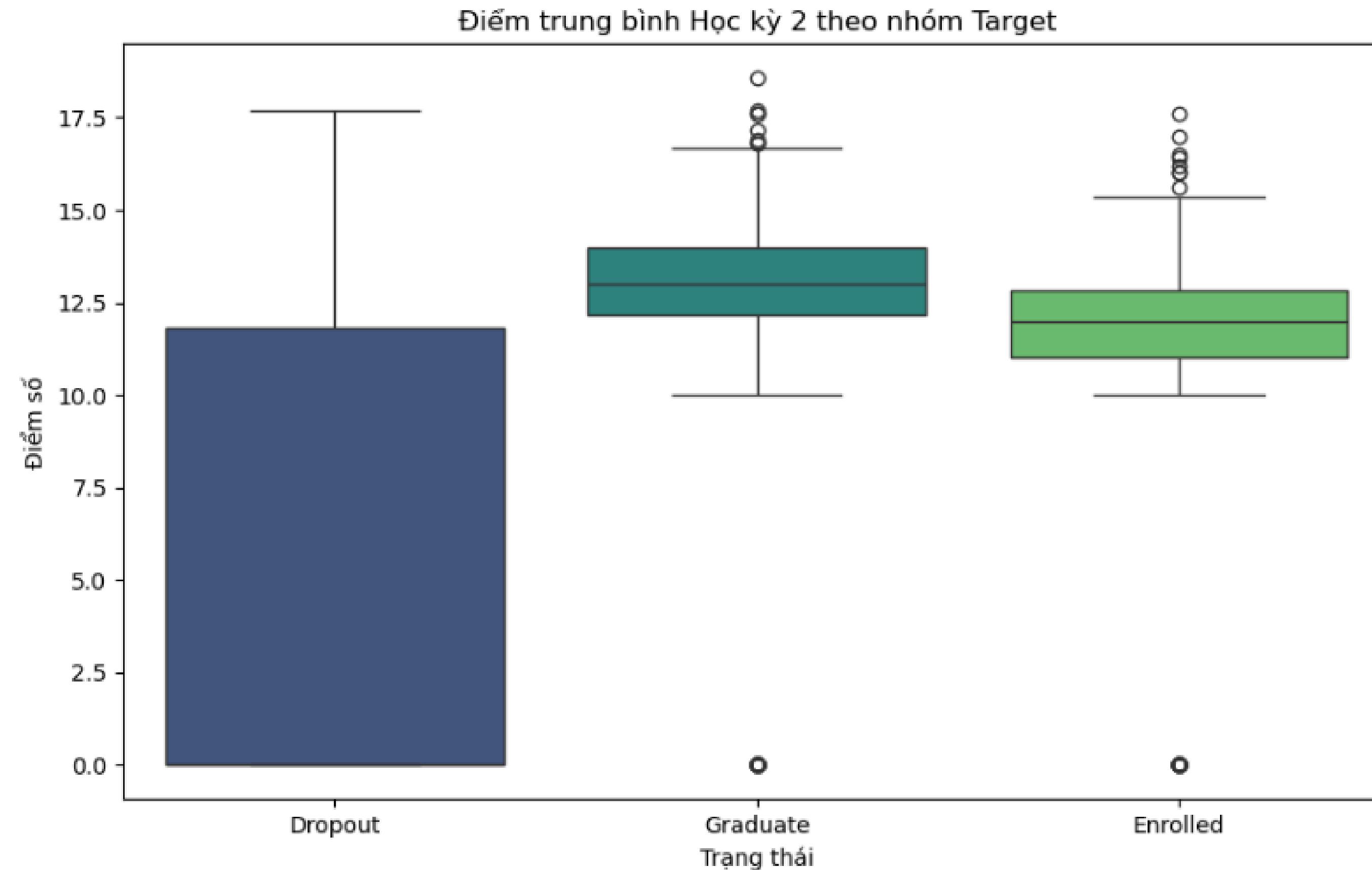
## 2.Biểu đồ phân phối tuổi nhập học theo nhóm target



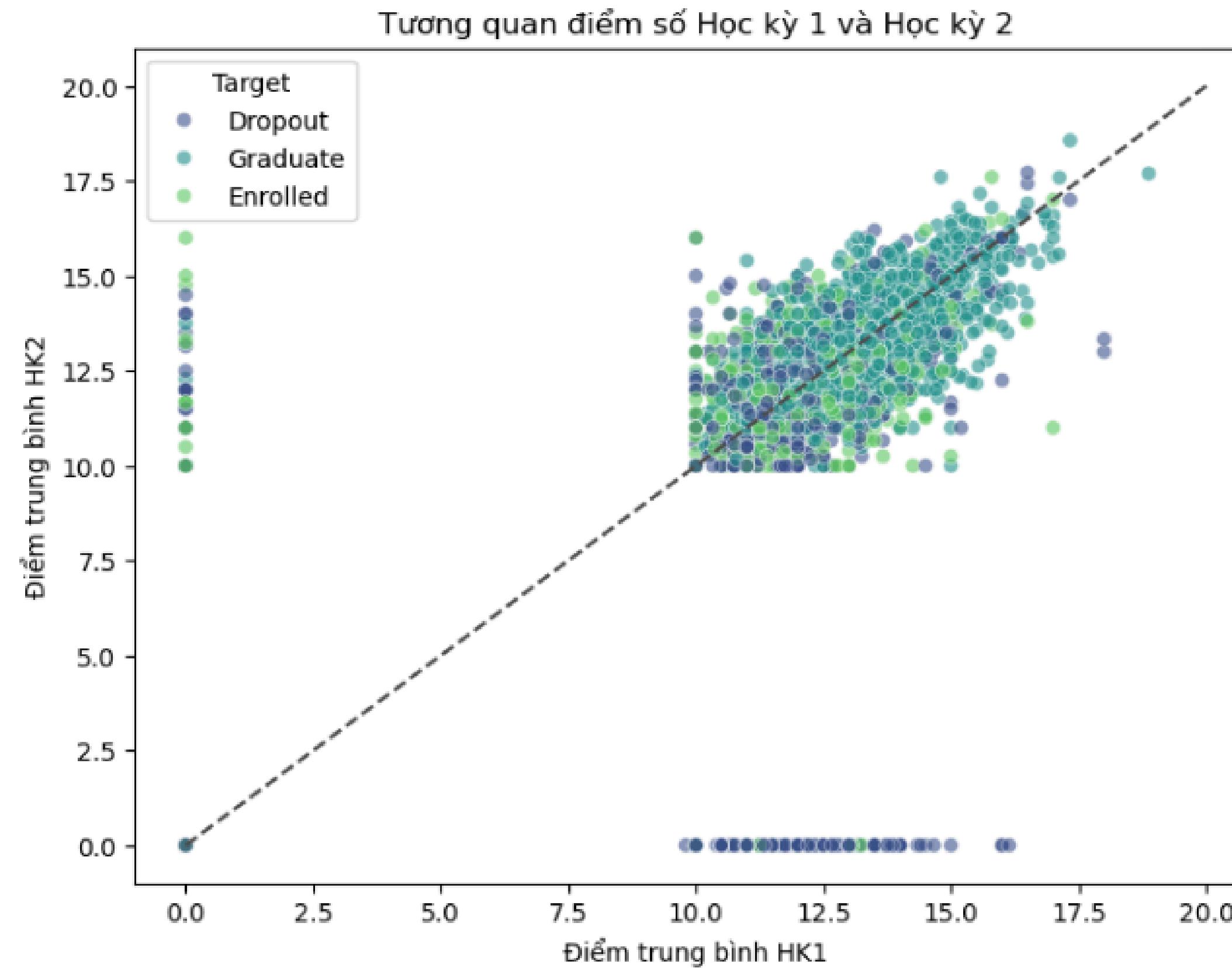
### 3. Biểu đồ tương quan về việc học bổng theo nhóm target



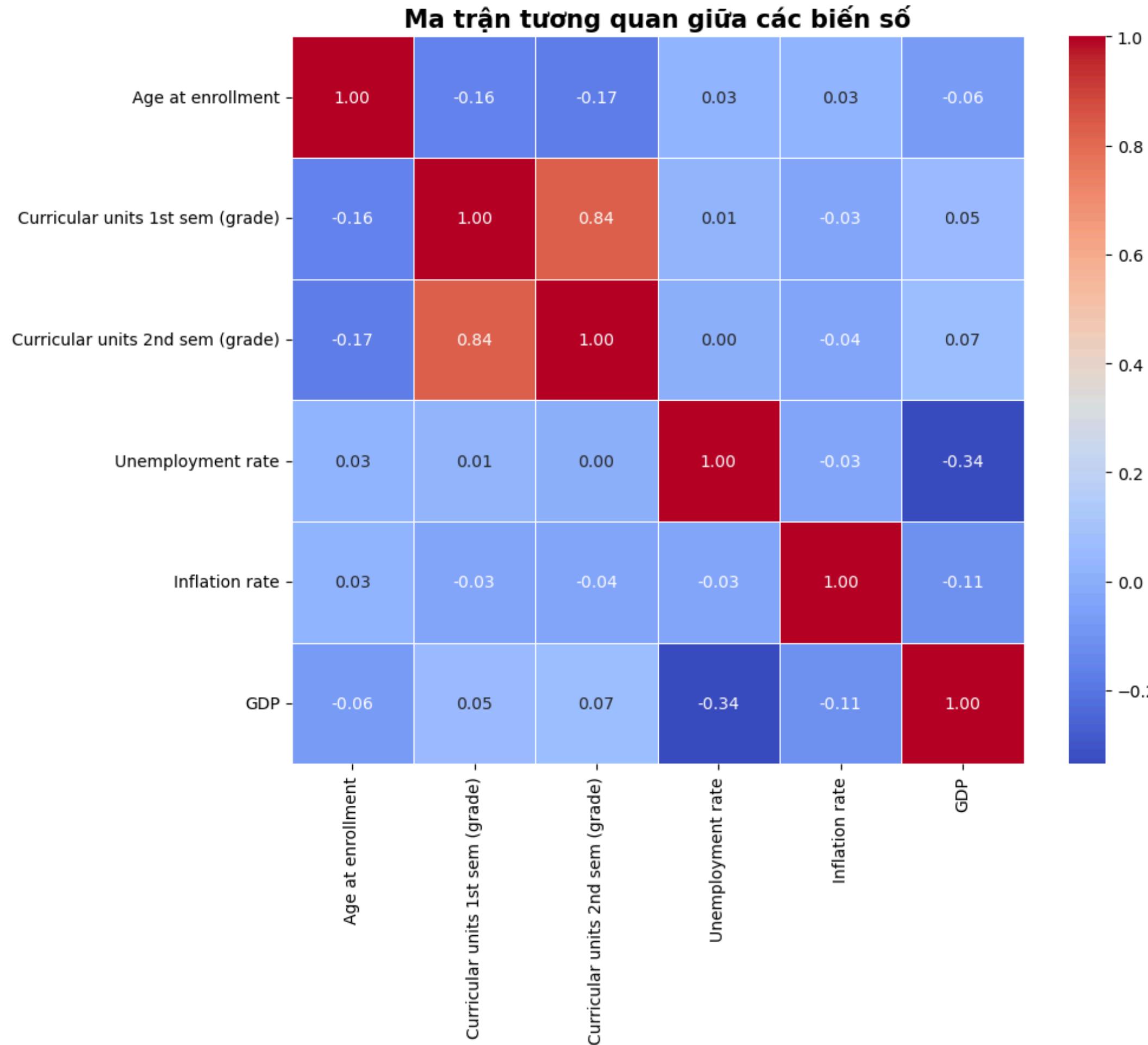
#### 4.Biểu đồ tương quan về điểm trung bình kì 2 theo nhóm target



## 5.Biểu đồ so sánh tương quan giữa điểm học kì 1 và học kì 2



# Ma trận tương quan



# Tiền xử lí dữ liệu

# Tiền xử lí dữ liệu

Chuyển nhãn Target thành binary

Nguyên bản: 3 lớp (Graduate, Enrolled, Dropout).

Chuyển thành:

- Dropout = 1 (lớp thiểu số – lớp quan tâm chính)
- Graduate + Enrolled = 0 (Non-Dropout – lớp đa số)

Lý do: Tập trung đúng vào mục tiêu cảnh báo sớm nguy cơ bỏ học, dễ đánh giá Precision/Recall cho lớp Dropout.

```
mapping = {
    'Dropout': 1,
    'Graduate': 0,
    'Enrolled': 0
}
df['Target'] = df['Target'].replace(mapping)
```

# Tiền xử lí dữ liệu

Xử lý đặc trưng Categorical:

```
cat_cols = [
    'Marital status', 'Application mode', 'Course', 'Previous qualification',
    'Nacionality', "Mother's qualification", "Father's qualification",
    "Mother's occupation", "Father's occupation"
]

num_cols = [col for col in df.columns if col not in cat_cols + ['Target']]

print(f"\nSố cột categorical: {len(cat_cols)}")
print(f"Số cột numerical: {len(num_cols)}")

preprocessor = ColumnTransformer([
    ('num', StandardScaler(), num_cols),                                # Chuẩn hóa numerical
    ('cat', OneHotEncoder(handle_unknown='ignore'), cat_cols)           # OneHot cho categorical
])
```

# Tiền xử lí dữ liệu

## Chia dữ liệu Train/Test

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42, stratify=y)
```

```
x_train.shape
```

```
(3539, 34)
```

```
x_test.shape
```

# Tiền xử lí dữ liệu

## Xử lí mất cân bằng

```
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train_processed, y_train)

fig, axes = plt.subplots(1, 2, figsize=(15, 6))
counts_before = pd.Series(y_train).value_counts().sort_index()
sns.barplot(x=counts_before.index, y=counts_before.values, ax=axes[0], palette='viridis')
axes[0].set_title('Trước khi SMOTE (Mất cân bằng)', fontsize=14, fontweight='bold')
axes[0].set_xlabel('Nhãn (0: Non-Dropout, 1: Dropout)')
axes[0].set_ylabel('Số lượng mẫu')
axes[0].grid(axis='y', linestyle='--', alpha=0.3)
for i, v in enumerate[Any](counts_before.values):
    axes[0].text(i, v + 10, str(v), ha='center', fontweight='bold')
counts_after = pd.Series(y_train_resampled).value_counts().sort_index()
sns.barplot(x=counts_after.index, y=counts_after.values, ax=axes[1], palette='deep')
axes[1].set_title('Sau khi SMOTE (Đã cân bằng)', fontsize=14, fontweight='bold')
axes[1].set_xlabel('Nhãn (0: Non-Dropout, 1: Dropout)')
axes[1].set_ylabel('Số lượng mẫu')
axes[1].grid(axis='y', linestyle='--', alpha=0.3)
for i, v in enumerate[Any](counts_after.values):
    axes[1].text(i, v + 10, str(v), ha='center', fontweight='bold')
plt.tight_layout()
plt.show()
```

# Các mô hình training

# Các thuật toán sử dụng



**KNN**

Dựa trên khoảng cách



**Naive Bayes**

Dựa trên xác suất



**Random Forest**

Dựa trên số đông



**Logistic  
Regression**

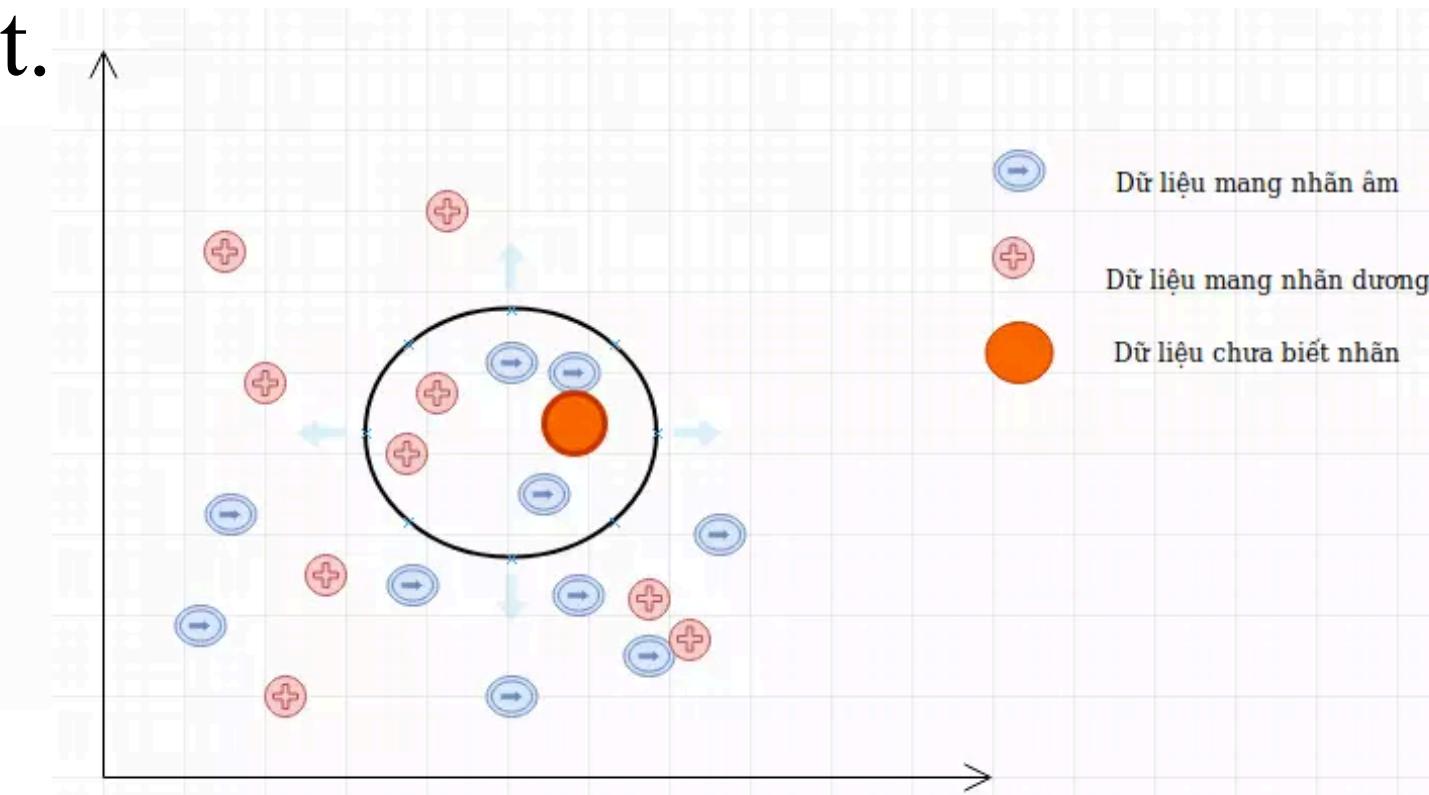
Dựa trên trọng số

# 1. KNN( K-Nearest Neighbors)

KNN là thuật toán non-parametric, instance-based, không học tham số mà lưu toàn bộ dữ liệu huấn luyện và dự đoán dựa trên k điểm gần nhất.

Công thức khoảng cách (Euclidean – phổ biến nhất)

$$d(x, x') = \sqrt{\sum_{i=1}^n (x_i - x'_i)^2}$$



Với một sinh viên mới cần dự đoán, tính khoảng cách từ sinh viên này đến tất cả sinh viên trong dữ liệu train. Sắp xếp khoảng cách tăng dần và chọn k sinh viên gần nhất (k là hyperparameter, em dùng k=5).

Bầu cử đa số (majority vote):

- Nhãn nào xuất hiện nhiều nhất trong k hàng xóm → làm dự đoán cuối cùng.
- Có thể dùng weighted vote: sinh viên gần hơn có trọng số cao hơn ( $1/d$  hoặc  $1/d^2$ ).

\*Hyperparameter (siêu tham số) là các tham số được thiết lập trước khi huấn luyện mô hình, không được học trực tiếp từ dữ liệu như trọng số (weights). Chúng kiểm soát cách thức học của mô hình, ảnh hưởng lớn đến hiệu suất cuối cùng.

# 1.KNN( K-Nearest Neighbors)

Ý tưởng cơ bản là “sinh viên nào giống nhau thì thường có kết quả giống nhau”. KNN không học quy tắc chung mà dựa trực tiếp vào kinh nghiệm quá khứ: tìm những sinh viên tương đồng nhất trong lịch sử và lấy kết quả của họ làm dự đoán.

Ưu điểm:

- Đơn giản, không giả định gì về dữ liệu
- Nắm bắt tốt pattern cục bộ (các nhóm sinh viên giống nhau)

Nhược điểm:

- Chậm khi dự đoán (tính khoảng cách với toàn bộ dữ liệu)
- Nhạy cảm với scale và nhiễu → dễ overfitting nếu k nhỏ

## 2.Naive Bayes

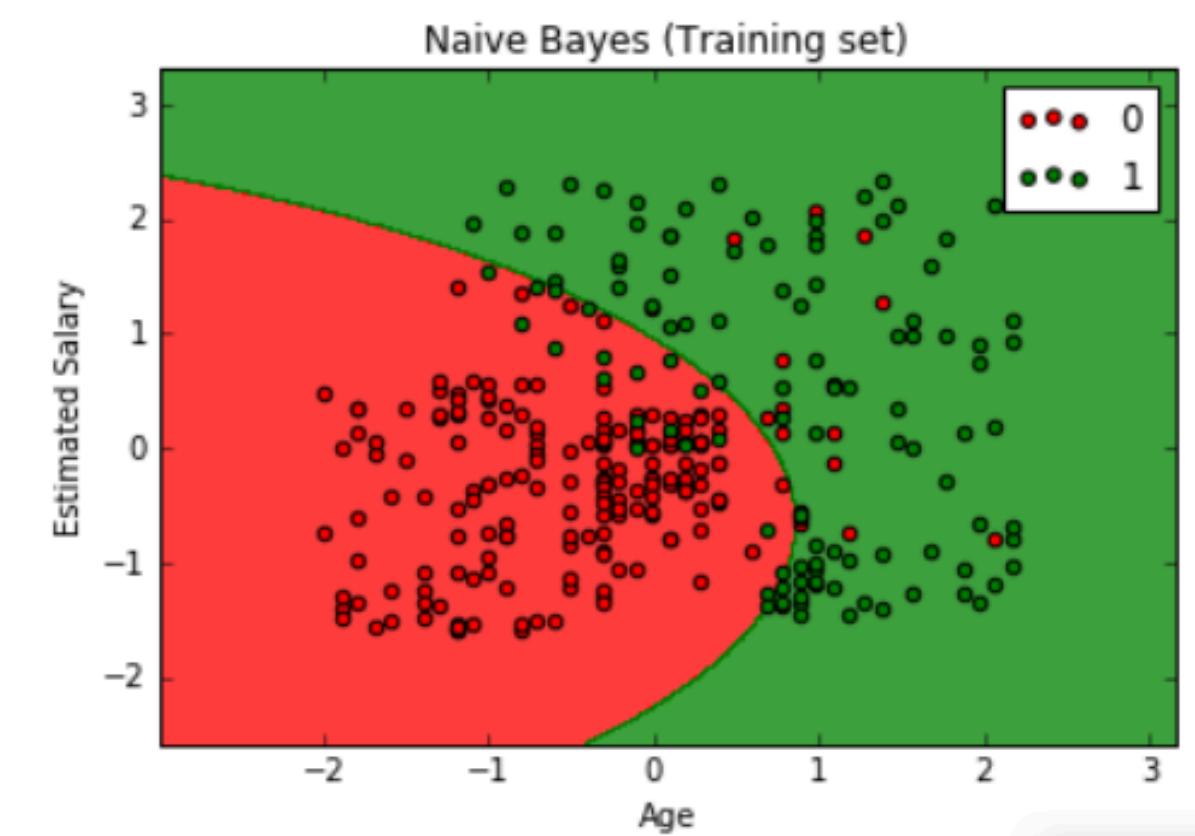
Naive Bayes là thuật toán xác suất, dựa trên định lý Bayes và giả định các đặc trưng độc lập với nhau khi đã biết nhãn lớp (Dropout hay Non-Dropout).

Ưu điểm:

- Rất nhanh, ít tài nguyên
- Ít tham số → ít overfitting
- Tốt với dữ liệu categorical

Nhược điểm:

- Giả định độc lập thường sai → nhiều báo động giả khi có tương tác mạnh
- Precision thấp nếu đặc trưng phụ thuộc nhau



## 2.Naive Bayes

Công thức tổng quát:

$$P(y | x) = \frac{P(x | y) \cdot P(y)}{P(x)}$$

Giả định độc lập có điều kiện (Naive assumption)

Sinh viên được biểu diễn bởi các đặc trưng:

$$x = (x_1, x_2, \dots, x_n)$$

Ví dụ:  $x_1$ : tuổi nhập học,  $x_2$ : đóng học phí đúng hạn,  $x_3$ : số môn đạt kỳ 2, ...

Naive Bayes giả định:

$$P(x | y) = \prod_{i=1}^n P(x_i | y)$$

**y (Nhãn - Label):** Trạng thái của học sinh → **Bỏ học** hoặc **Tiếp tục học**.

**x (Dữ liệu đầu vào):** Hồ sơ học sinh sau khi đã được số hóa (vector hóa), bao gồm các chỉ số như: tỷ lệ chuyên cần, điểm số GPA, số lần vi phạm kỷ luật, điều kiện kinh tế...

**P(y|x):** Xác suất học sinh đó sẽ **bỏ học** khi nhà trường biết được bộ hồ sơ/dữ liệu  $x$  của học sinh đó.

## 2.Naive Bayes

Công thức phân loại cuối cùng (dạng log)

$$\hat{y} = \arg \max_y \left[ \log P(y) + \sum_{i=1}^n \log P(x_i | y) \right]$$

- $\log P(y)$  (**Xác suất tiên nghiệm**): Đây là tỷ lệ học sinh bỏ học chung của toàn trường (hoặc toàn hệ thống) được thống kê từ trước khi xem xét hồ sơ cá nhân. Nó phản ánh mức độ phổ biến của việc bỏ học trong quá khứ.
- $\log P(x_i|y)$  (**Mức độ đặc trưng**): Cho biết một biểu hiện cụ thể  $x_i$  (ví dụ: nghỉ học quá 5 buổi) thường xuất hiện như thế nào trong nhóm học sinh đã từng bỏ học.

Smoothing (Laplace):

$$P(x_i | y) = \frac{\text{count}(x_i, y) + \alpha}{\sum_x \text{count}(x, y) + \alpha |V|}$$

Ý nghĩa:

- Tránh xác suất bằng 0 khi gặp giá trị chưa từng thấy trong train.
- Nếu một sinh viên có đặc trưng lạ → mô hình vẫn xử lý được

### 3. Random Forest

Random Forest là ensemble của nhiều Decision Tree, sử dụng kỹ thuật Bagging + random feature selection để tăng độ chính xác và chống overfitting.

Ensemble là gì?

Ensemble Learning là kỹ thuật kết hợp nhiều mô hình học máy (weak learners) → để tạo ra một mô hình mạnh hơn (strong learner).

Ý tưởng: Nhiều mô hình đoán chung → kết quả tốt và ổn định hơn một mô hình đơn lẻ.

Ba phương pháp ensemble phổ biến:

- Bagging (Bootstrap Aggregating) – Giảm variance
  - Huấn luyện nhiều mô hình độc lập
  - Mỗi mô hình học trên dữ liệu lấy mẫu khác nhau
  - Kết quả cuối cùng bằng trung bình hoặc đa số phiếu của model
- Boosting – Giảm Bias
  - Các mô hình học tuần tự
  - Mô hình sau tập trung vào mẫu mà mô hình trước đoán sai.
- Stacking
  - Kết hợp nhiều mô hình khác loại
  - Một mô hình khác (meta-model) học cách tổng hợp kết quả

### 3. Random Forest

Bagging là gì?

Bagging Là một kỹ thuật Ensemble Learning với ý tưởng:

- Huấn luyện nhiều mô hình độc lập trên các tập dữ liệu lấy mẫu ngẫu nhiên, rồi tổng hợp kết quả.

Các bước thực hiện :

Bước 1: Tạo ngẫu nhiên các N bags từ tập train set.

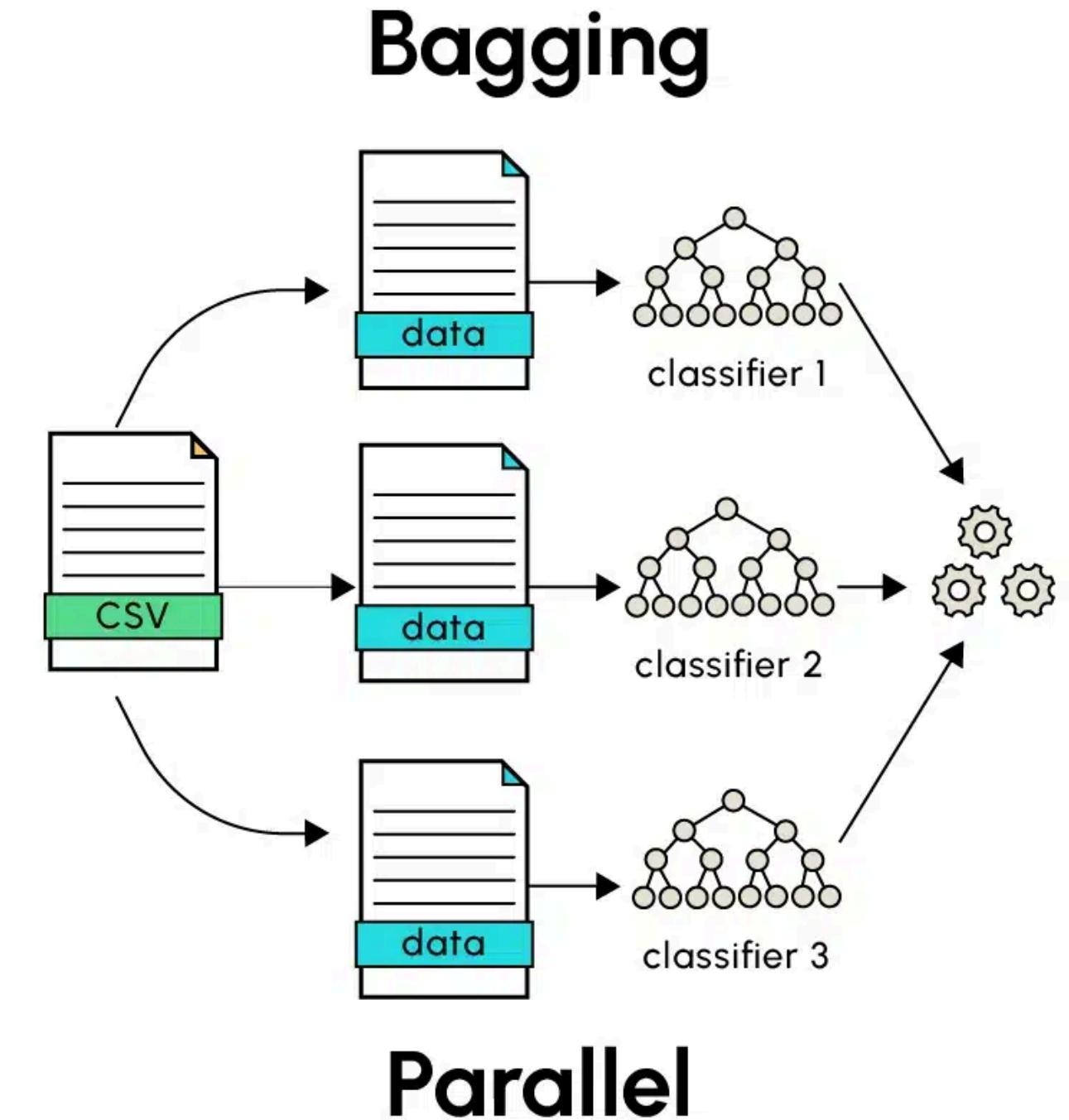
Bước 2: Tạo N model "yếu" và train trên mỗi bag, độc lập với nhau.

Bước 3: Sử dụng các objects đã trained để dự đoán

Các bag được dựng lên bởi 2 tham số:

Max\_samples: Số sample mỗi bag

Max\_features: Số features sử dụng trong các bags



### 3. Random Forest

Ý tưởng cơ bản là xây dựng hàng trăm cây quyết định. Mỗi cây học trên dữ liệu hơi khác (bootstrap) và xem xét góc nhìn khác (random features) → đa dạng ý kiến. Quyết định cuối cùng bằng bỏ phiếu đa số → chính xác, ổn định và ít bị ảnh hưởng bởi noise hơn nhiều so với một cây đơn lẻ.

Ưu điểm :

- Giảm overfitting so với Decision Tree nhờ trung bình hóa kết quả của nhiều cây độc lập
- Độ chính xác cao

Nhược điểm:

- Chậm hơn mô hình đơn
- Tốn tài nguyên bộ nhớ

# 4. Logistic Regression

Logistic Regression là mô hình tuyến tính cho phân loại nhị phân, mô hình hóa xác suất một sinh viên thuộc lớp Dropout bằng hàm sigmoid.

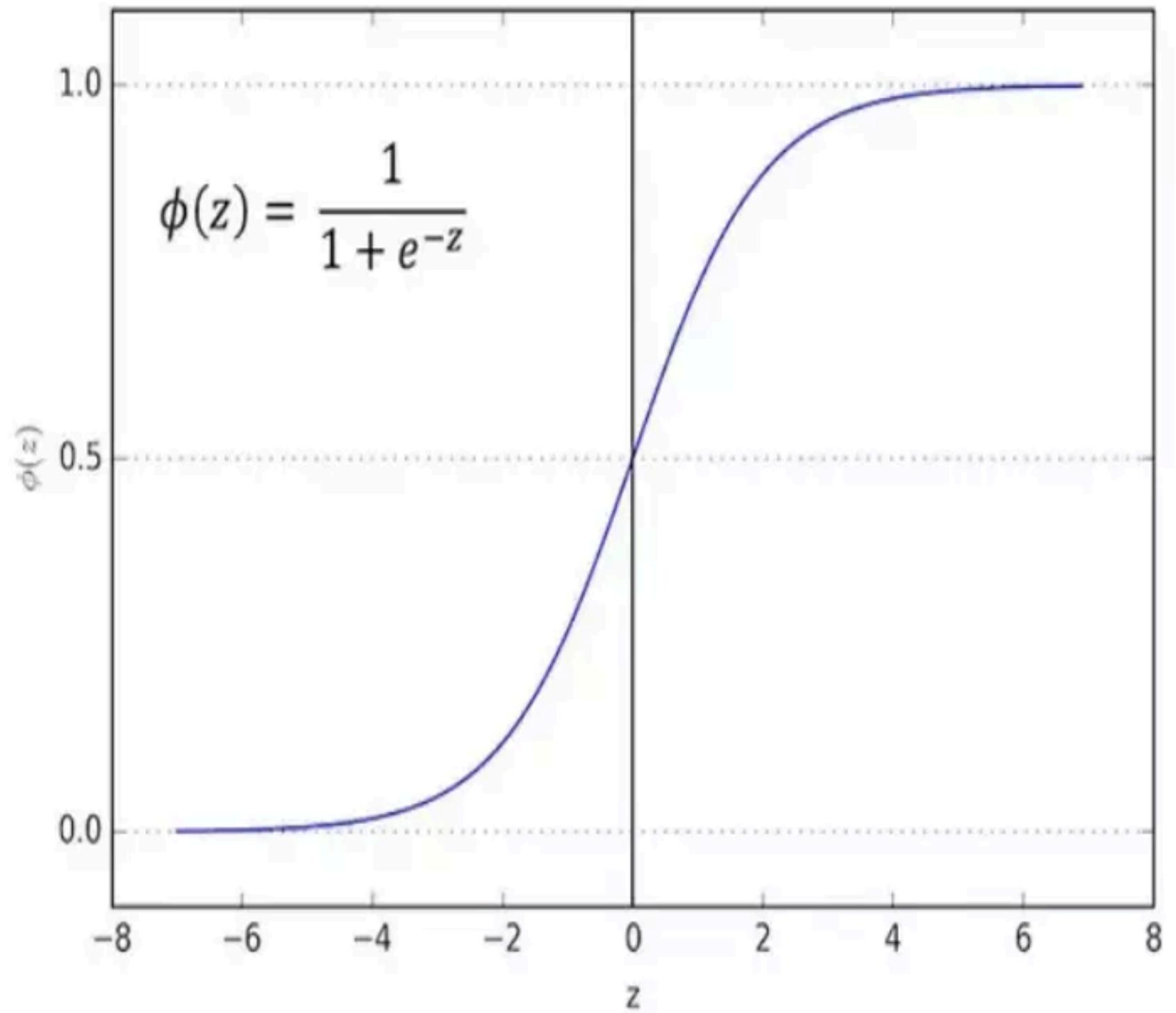
Hàm tuyến tính

$$z = w^T x + b$$

Hàm sigmoid

$$\hat{y} = \sigma(z)$$

trong đó:  $\sigma(z) = \frac{1}{1 + e^{-z}}$



# 4. Logistic Regression

## Ý tưởng thuật toán

Ý tưởng cơ bản là tìm một siêu phẳng (đường thẳng trong không gian đa chiều) phân biệt tốt nhất hai lớp Dropout và Non-Dropout. Mô hình học các trọng số w sao cho tổng có trọng số của các đặc trưng đẩy xác suất về đúng phía nhãn lớp. Nhờ giả định tuyến tính, Logistic Regression rất ổn định, dễ giải thích và thường được dùng làm baseline.

## Hàm mất mát (Loss): Cross-entropy (Log Loss)

$$L = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

Nếu mô hình dự đoán một sinh viên nguy cơ thấp nhưng thực tế bỏ học  $\rightarrow$  loss lớn  $\rightarrow$  phạt mạnh  $\rightarrow$  lần sau mô hình sẽ chú ý hơn đến các dấu hiệu tương tự.

# 4. Logistic Regression

## Regularization

$$L_{\text{total}} = L_{\text{cross-entropy}} + \lambda \sum_{j=1}^p w_j^2$$

Ngăn chặn overfitting – tình trạng mô hình học quá sát dữ liệu train (bao gồm cả nhiễu) → hiệu suất tốt trên train nhưng kém trên dữ liệu mới (sinh viên năm sau)

Regularization thêm phạt vào loss khi trọng số quá lớn → buộc mô hình đơn giản hơn, tập trung vào các yếu tố chính (tuổi, đóng phí, kỳ 2) thay vì “nhớ” các trường hợp cá biệt nhiều

# **Huân luyện mô hình**

**Chia 2 tập dữ liệu:**

- Train:80%
- Test:20%

# Kết quả chạy mô hình

Mô hình	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Logistic Regression	0,83	0,80	0,84	0,82	0,931
Naive Bayes	0,68	0,33	0,99	0,49	0,514
KNN (k=5)	0,79	0,63	0,80	0,71	0,860
Random Forest	0,85	0,89	0,71	0,79	0,930

# Các chỉ số đánh giá mô hình

## Confusion matrix

Là một phương pháp đánh giá kết quả của những bài toán phân loại với việc xem xét cả những chỉ số về độ chính xác và độ bao quát của các dự đoán cho từng lớp.

Đây là nền tảng để suy ra các chỉ số đánh giá.

- TP (True Positive): Sinh viên thực tế bỏ học và mô hình dự đoán đúng là có nguy cơ → phát hiện đúng, nhà trường can thiệp kịp thời → thành công lớn nhất.
- TN (True Negative): Sinh viên không bỏ học và mô hình dự đoán ổn định → đúng, không cần can thiệp.
- FP (False Positive): Sinh viên không bỏ học nhưng mô hình báo nguy cơ cao → báo động giả → nhà trường lãng phí nguồn lực hỗ trợ (học bổng, tư vấn) cho người không cần.
- FN (False Negative): Sinh viên thực tế bỏ học nhưng mô hình dự đoán ổn định → bỏ sót nguy cơ → hậu quả nghiêm trọng nhất (sinh viên bỏ học mà không được hỗ trợ).

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

# Các chỉ số đánh giá mô hình

Chỉ số	Công thức	Ý nghĩa	Diễn giải
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Độ chính xác tổng thể	Tỷ lệ mẫu được dự đoán đúng trên toàn bộ tập dữ liệu
Precision	$\frac{TP}{TP+FP}$	Độ chính xác của dự đoán Positive	Trong các mẫu được dự đoán là Positive, tỷ lệ dự đoán đúng
Recall (Sensitivity)	$\frac{TP}{TP+FN}$	Khả năng phát hiện Positive	Trong các mẫu Positive thực tế, tỷ lệ được mô hình phát hiện
F1-score	$2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$	Trung bình điều hòa Precision và Recall	Đánh giá mức cân bằng giữa Precision và Recall

# Confusion Matrix

Thực tế/dự đoán	Thực tế/dự đoán	Thực tế/dự đoán
Dropout (1)	<b>TP = 202</b>	<b>FN = 82</b>
Non-Dropout (0)	<b>FP = 25</b>	<b>TN = 576</b>

# Confusion Matrix

Ký hiệu	Ý nghĩa	Diễn giải thực tế trong cảnh báo sớm bỏ học
<b>TN</b>	Non-Dropout → Non-Dropout	Sinh viên không có nguy cơ bỏ học và mô hình dự đoán đúng là ổn định. → Không cần can thiệp – tiết kiệm nguồn lực.
<b>FP</b>	Non-Dropout → Dropout	Sinh viên thực tế ổn định nhưng mô hình báo nguy cơ cao. → Báo động giả – nhà trường có thể hỗ trợ nhầm (lãng phí học bổng, tư vấn).
<b>FN</b>	Dropout → Non-Dropout	Sinh viên thực tế bỏ học nhưng mô hình dự đoán ổn định. → Bỏ sót nguy cơ – hậu quả nghiêm trọng nhất (sinh viên bỏ học mà không được hỗ trợ kịp thời).
<b>TP</b>	Dropout → Dropout	Sinh viên thực tế bỏ học và mô hình dự đoán đúng nguy cơ cao. → Phát hiện đúng – nhà trường can thiệp kịp thời (hỗ trợ tài chính, tư vấn) → cứu được sinh viên.

# Kết quả và demo

# KNN

--- KẾT QUẢ KNN (SAU KHI SMOTE) ---

Độ chính xác (Accuracy): 0.7819

Độ nhạy (Recall Dropout): 0.8028

Báo cáo chi tiết (Classification Report):

	precision	recall	f1-score	support
0	0.89	0.77	0.83	601
1	0.62	0.80	0.70	284
accuracy			0.78	885
macro avg	0.76	0.79	0.77	885
weighted avg	0.81	0.78	0.79	885

# Navie bayes

--- KẾT QUẢ NAIVE BAYES

Độ chính xác (Accuracy): 0.3480

Độ nhạy (Recall Dropout): 0.9894

Báo cáo chi tiết (Classification Report):

	precision	recall	f1-score	support
0	0.90	0.04	0.09	601
1	0.33	0.99	0.49	284
accuracy			0.35	885
macro avg	0.61	0.52	0.29	885
weighted avg	0.72	0.35	0.22	885

# Random forest

--- KẾT QUẢ RANDOM FOREST

Độ chính xác (Accuracy): 0.8791

Độ nhạy (Recall Dropout): 0.7676

Báo cáo chi tiết (Classification Report):

	precision	recall	f1-score	support
0	0.89	0.93	0.91	601
1	0.84	0.77	0.80	284
accuracy			0.88	885
macro avg	0.87	0.85	0.86	885
weighted avg	0.88	0.88	0.88	885

# Logistic Regression

--- KẾT QUẢ LOGISTIC REGRESSION

Độ chính xác (Accuracy): 0.8802

Báo cáo chi tiết (Classification Report):

	precision	recall	f1-score	support
0	0.93	0.89	0.91	601
1	0.79	0.85	0.82	284
accuracy			0.88	885
macro avg	0.86	0.87	0.87	885
weighted avg	0.88	0.88	0.88	885

Tiêu chí đánh giá	Naive Bayes	Logistic Regression	KNN	Random Forest
Accuracy (Test)	Trung bình	Cao	Cao	Rất cao
Precision (Dropout)	Thấp	Cao	Trung bình - Cao	Rất cao
Recall (Dropout)	Rất cao	Cao	Cao	Trung bình - Cao
F1-score (Dropout)	Thấp	Cao & ổn định	Trung bình - Cao	Cao
Recall gap (Train - Test)	Nhỏ	Nhỏ	Trung bình	Trung bình
Nguy cơ Overfitting	Rất thấp	Rất thấp	Trung bình - Cao	Thấp
Thời gian huấn luyện	Rất nhanh	Nhanh	Nhanh	Trung bình
Khả năng tổng quát hóa	Trung bình	Tốt	Trung bình	Rất tốt
Phù hợp triển khai thực tế	Thấp	Cao	Trung bình	Rất cao

Random Forest là mô hình tối ưu nhất cho bài toán cảnh báo sớm bão học:

- Precision cao nhất → báo động đáng tin cậy.
- F1-score cao, tổng quát hóa tốt.
- Cung cấp feature importance rõ ràng (kỳ 2 quan trọng nhất) → hỗ trợ nhà trường rút insight thực tiễn.
- 

Logistic Regression là baseline rất mạnh, dễ giải thích – phù hợp nếu cần mô hình đơn giản, minh bạch.

KNN và Naive Bayes giúp hiểu dữ liệu tốt hơn nhưng không phù hợp triển khai chính (Precision thấp hoặc chậm).

**THANK YOU  
FOR LISTENING**