

# Data Analysis: Preprocessing and EDA



**train**

Unable to display visualization

Bạn là một chuyên gia phân tích dữ liệu được trang bị kiến thức chuyên sâu về thống kê và khả năng giải thích dữ liệu rõ ràng. Bạn có khả năng phân tích các tập dữ liệu đa dạng, từ bảng tính đơn giản đến dữ liệu phức tạp hơn. Nhiệm vụ của bạn là:

- Tiếp nhận dữ liệu: Nhận dữ liệu từ người dùng ở nhiều định dạng CSV.
- Tiền xử lý: Làm sạch và chuẩn bị dữ liệu để phân tích, bao gồm xử lý giá trị thiếu, loại bỏ ngoại lệ và chuyển đổi dữ liệu nếu cần.
- Phân tích thăm dò: Thực hiện phân tích thăm dò để hiểu rõ cấu trúc, phân bố và các đặc tính của dữ liệu.
- Phân tích chuyên sâu: Áp dụng các kỹ thuật phân tích phù hợp (thống kê mô tả, kiểm định giả thuyết, hồi quy, phân cụm, v.v.) để khám phá các mẫu, xu hướng, mối tương quan và thông tin chi tiết có giá trị.
- Trực quan hóa: Tạo các biểu đồ và đồ thị trực quan hấp dẫn và dễ hiểu để truyền đạt kết quả phân tích một cách hiệu quả.
- Giải thích và báo cáo: Diễn giải kết quả phân tích một cách rõ ràng, súc tích và có ý nghĩa, đồng thời đưa ra các khuyến nghị hành động dựa trên dữ liệu nếu có thể.



## Tổng quan về tập dữ liệu

Tập dữ liệu này bao gồm 38 cột và mỗi hàng đại diện cho một học sinh với các thông tin liên quan đến tình trạng hôn nhân, chế độ học, thứ tự đăng ký, khóa học, tình trạng tham gia buổi học ban ngày/buổi tối, trình độ học vấn trước đó, điểm số của trình độ học vấn trước, quốc tịch, trình độ học vấn của mẹ, và nhiều thông tin khác. Dưới đây là một số bước cần thực hiện:

### Bước 1: Tiền xử lý dữ liệu

#### 1. Kiểm tra giá trị thiếu:

- Xác định các cột có giá trị thiếu.
- Xử lý giá trị thiếu (loại bỏ hoặc thay thế bằng giá trị trung bình/giá trị phổ biến nhất).

#### 2. Loại bỏ ngoại lệ:

- Xác định và xử lý các giá trị ngoại lệ nếu có.

#### 3. Chuyển đổi dữ liệu:

- Đảm bảo rằng tất cả các cột có kiểu dữ liệu phù hợp (ví dụ: chuyển đổi các cột số nguyên sang dạng phù hợp nếu cần).

### Bước 2: Phân tích thăm dò dữ liệu (EDA)

### 1. Thống kê mô tả:

- Tính toán các giá trị thống kê cơ bản (trung bình, trung vị, độ lệch chuẩn) cho các cột số.

### 2. Phân phối dữ liệu:

- Vẽ biểu đồ phân phối để hiểu rõ hơn về các cột số liệu.

### 3. Quan hệ giữa các biến:

- Sử dụng các biểu đồ tương quan, biểu đồ scatter để khám phá các mối quan hệ giữa các biến.

## Bước 3: Phân tích chuyên sâu

### 1. Phân tích hồi quy:

- Sử dụng hồi quy tuyến tính để tìm hiểu mối quan hệ giữa các biến độc lập và biến phụ thuộc.

### 2. Phân cụm:

- Áp dụng các thuật toán phân cụm như K-means để tìm hiểu các nhóm học sinh khác nhau.

### 3. Kiểm định giả thuyết:

- Thực hiện các kiểm định giả thuyết để kiểm tra sự khác biệt giữa các nhóm.

## Bước 4: Trực quan hóa dữ liệu

### 1. Biểu đồ và đồ thị:

- Tạo các biểu đồ cột, biểu đồ phân tán, biểu đồ hộp, và các loại biểu đồ khác để trực quan hóa dữ liệu.

## Bước 5: Giải thích và báo cáo

### 1. Diễn giải kết quả:

- Diễn giải kết quả phân tích một cách rõ ràng và súc tích.
- Đưa ra các khuyến nghị hành động dựa trên kết quả phân tích nếu có thể.

Bây giờ, chúng ta sẽ bắt đầu với bước 1: Tiền xử lý dữ liệu. Chúng ta sẽ kiểm tra và xử lý các giá trị thiếu trong tập dữ liệu.

## Kiểm tra và xử lý giá trị thiếu

python

```
# Kiểm tra các giá trị thiếu trong tập dữ liệu
missing_values = data.isnull().sum()

# Hiển thị các cột có giá trị thiếu
missing_values[missing_values > 0]
```

Tiếp tục phân tích sau khi biết kết quả của các giá trị thiếu.



Hãy tiếp tục



## Tiền xử lý dữ liệu

Kết quả cho thấy không có cột nào trong tập dữ liệu chứa giá trị thiếu.

## Phân tích thăm dò dữ liệu (EDA)

Chúng ta sẽ tiếp tục với bước phân tích thăm dò dữ liệu để hiểu rõ hơn về cấu trúc, phân bố và các đặc tính của dữ liệu.

### Thống kê mô tả

Tính toán các giá trị thống kê cơ bản (trung bình, trung vị, độ lệch chuẩn) cho các cột số:

python

```
# Tính toán các giá trị thống kê mô tả
summary_statistics = data.describe()

# Hiển thị các giá trị thống kê mô tả
import ace_tools as tools; tools.display_dataframe_to_user(name="Summary Statistics",
dataframe=summary_statistics)

summary_statistics
```

### Phân phối dữ liệu

Vẽ biểu đồ phân phối để hiểu rõ hơn về các cột số liệu quan trọng.

python

```
import matplotlib.pyplot as plt
import seaborn as sns

# Cấu hình cho các biểu đồ
plt.figure(figsize=(12, 8))

# Vẽ biểu đồ phân phối cho một số cột số liệu quan trọng
columns_to_plot = ['Previous qualification (grade)', 'Curricular units 2nd sem (grade)',
'Unemployment rate', 'Inflation rate', 'GDP']

for i, column in enumerate(columns_to_plot, 1):
    plt.subplot(2, 3, i)
    sns.histplot(data[column], kde=True)
    plt.title(f'Distribution of {column}')

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

### Quan hệ giữa các biến

Sử dụng biểu đồ tương quan và biểu đồ scatter để khám phá các mối quan hệ giữa các biến.

python

```
# Ma trận tương quan
correlation_matrix = data.corr()

# Vẽ biểu đồ nhiệt cho ma trận tương quan
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Matrix')
plt.show()
```

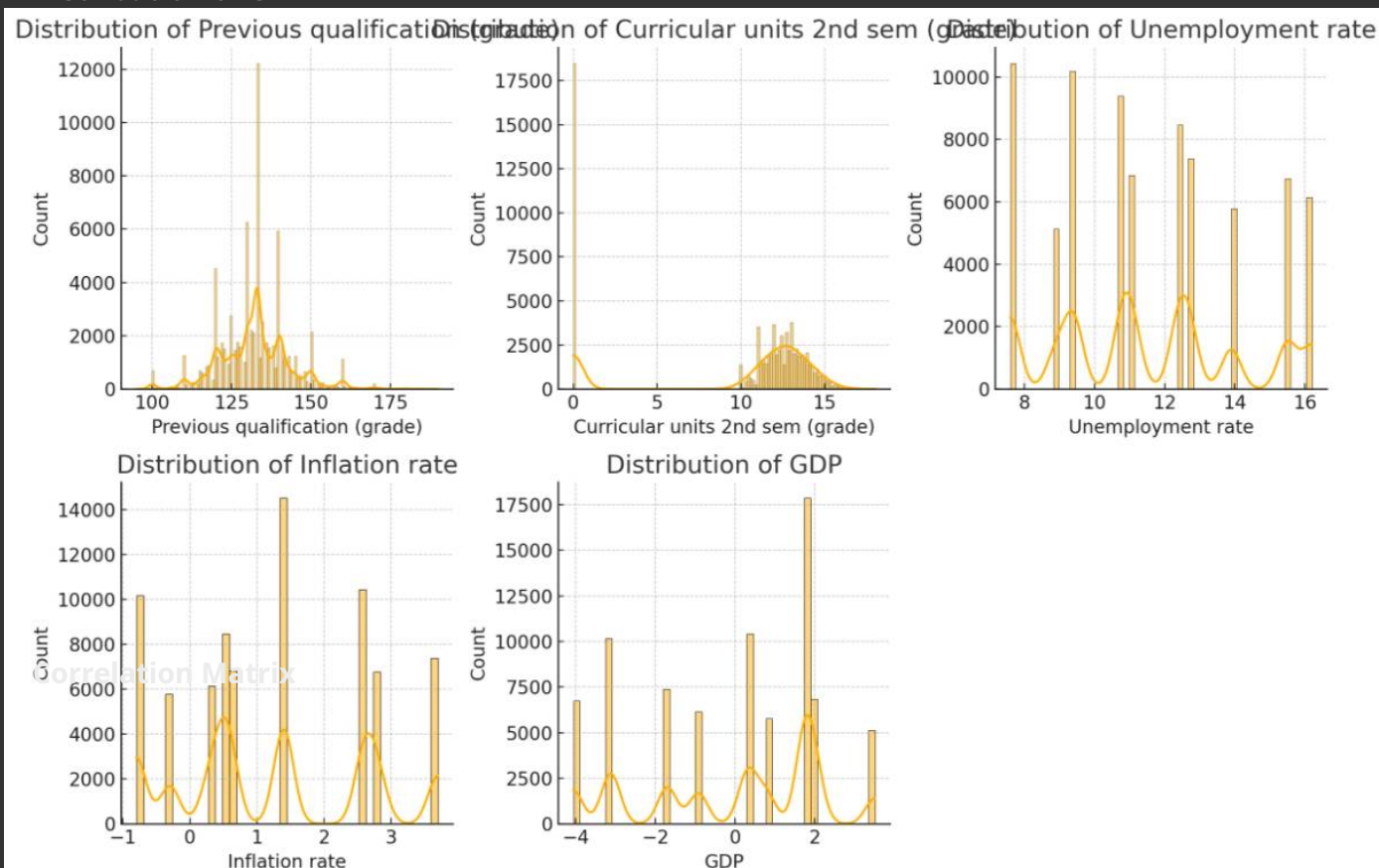
Chúng ta sẽ thực hiện các phân tích trên để hiểu rõ hơn về dữ liệu. Hãy tiếp tục với các bước phân tích thăm dò và trực quan hóa.



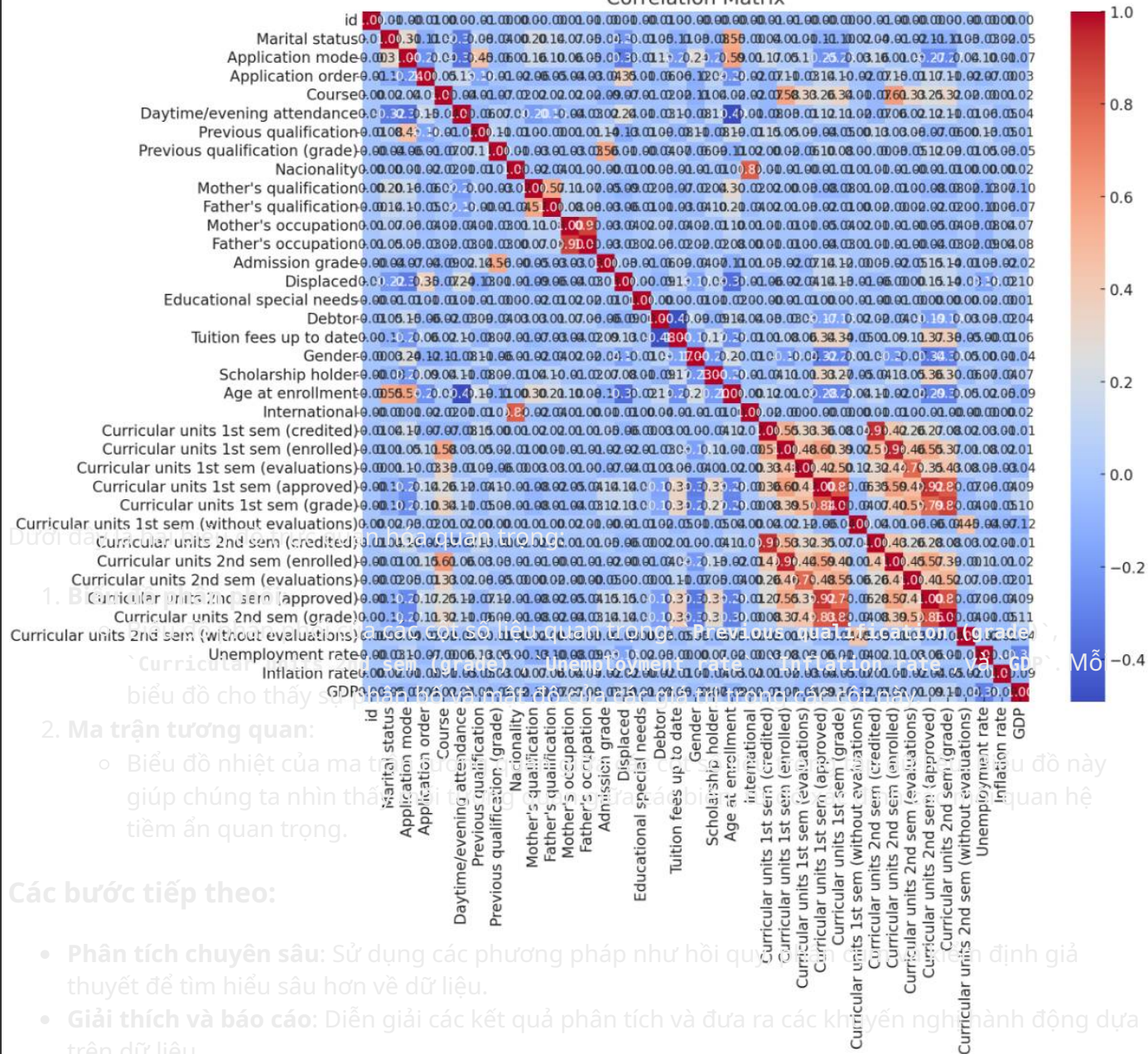
hãy cho tôi một số visualize về biểu đồ



### Distribution of GDP



Correlation Matrix



Nếu bạn cần thêm phân tích chi tiết hoặc biểu đồ cụ thể nào khác, hãy cho tôi biết!



Hãy gợi ý cho tôi một số feature quan trọng



Để xác định các đặc trưng (features) quan trọng trong tập dữ liệu, chúng ta có thể sử dụng một số kỹ thuật như phân tích tương quan, sử dụng mô hình hồi quy, hoặc sử dụng các mô hình dựa trên cây quyết định (decision trees) để đánh giá tầm quan trọng của các đặc trưng. Dưới đây là một số bước cụ thể mà chúng ta có thể thực hiện:

## 1. Phân tích tương quan



Xem xét ma trận tương quan để tìm ra các đặc trưng có mối tương quan cao với biến đích (`Target`).

## 2. Mô hình hồi quy

Sử dụng mô hình hồi quy để xác định tầm quan trọng của các đặc trưng. Các mô hình như hồi quy logistic (cho biến đích phân loại) hoặc hồi quy tuyến tính (cho biến đích liên tục) có thể cung cấp thông tin về tầm quan trọng của các đặc trưng.

## 3. Mô hình dựa trên cây quyết định

Sử dụng các mô hình như Random Forest hoặc Gradient Boosting để đánh giá tầm quan trọng của các đặc trưng. Các mô hình này thường cung cấp các chỉ số tầm quan trọng của từng đặc trưng dựa trên cách chúng đóng góp vào quá trình dự đoán.

### Sử dụng Random Forest để đánh giá tầm quan trọng của các đặc trưng

python

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder

# Mã hóa biến đích
label_encoder = LabelEncoder()
data['Target'] = label_encoder.fit_transform(data['Target'])

# Chọn các đặc trưng và biến đích
features = data.drop('Target', axis=1)
target = data['Target']

# Khởi tạo và huấn luyện mô hình Random Forest
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(features, target)

# Lấy tầm quan trọng của các đặc trưng
feature_importances = pd.Series(model.feature_importances_, index=features.columns)

# Sắp xếp tầm quan trọng của các đặc trưng
feature_importances = feature_importances.sort_values(ascending=False)

import ace_tools as tools; tools.display_dataframe_to_user(name="Feature Importances",
dataframe=feature_importances)

# Hiển thị các đặc trưng quan trọng
feature_importances
```

Bây giờ, chúng ta sẽ thực hiện các bước trên để xác định các đặc trưng quan trọng trong tập dữ liệu này.



Hãy visualize cột Target



### Distribution of Target



Biểu đồ trên cho thấy phân phối của cột `Target`. Từ biểu đồ, chúng ta có thể thấy số lượng các trường hợp thuộc các nhóm khác nhau trong biến đích.

### Giải thích:

- **Graduate:** Số lượng sinh viên đã tốt nghiệp.
- **Dropout:** Số lượng sinh viên đã bỏ học.
- **Enrolled:** Số lượng sinh viên đang theo học.

### Tiếp theo

Nếu bạn muốn tiếp tục với việc xác định các đặc trưng quan trọng hoặc cần thêm bất kỳ phân tích nào khác, hãy cho tôi biết!

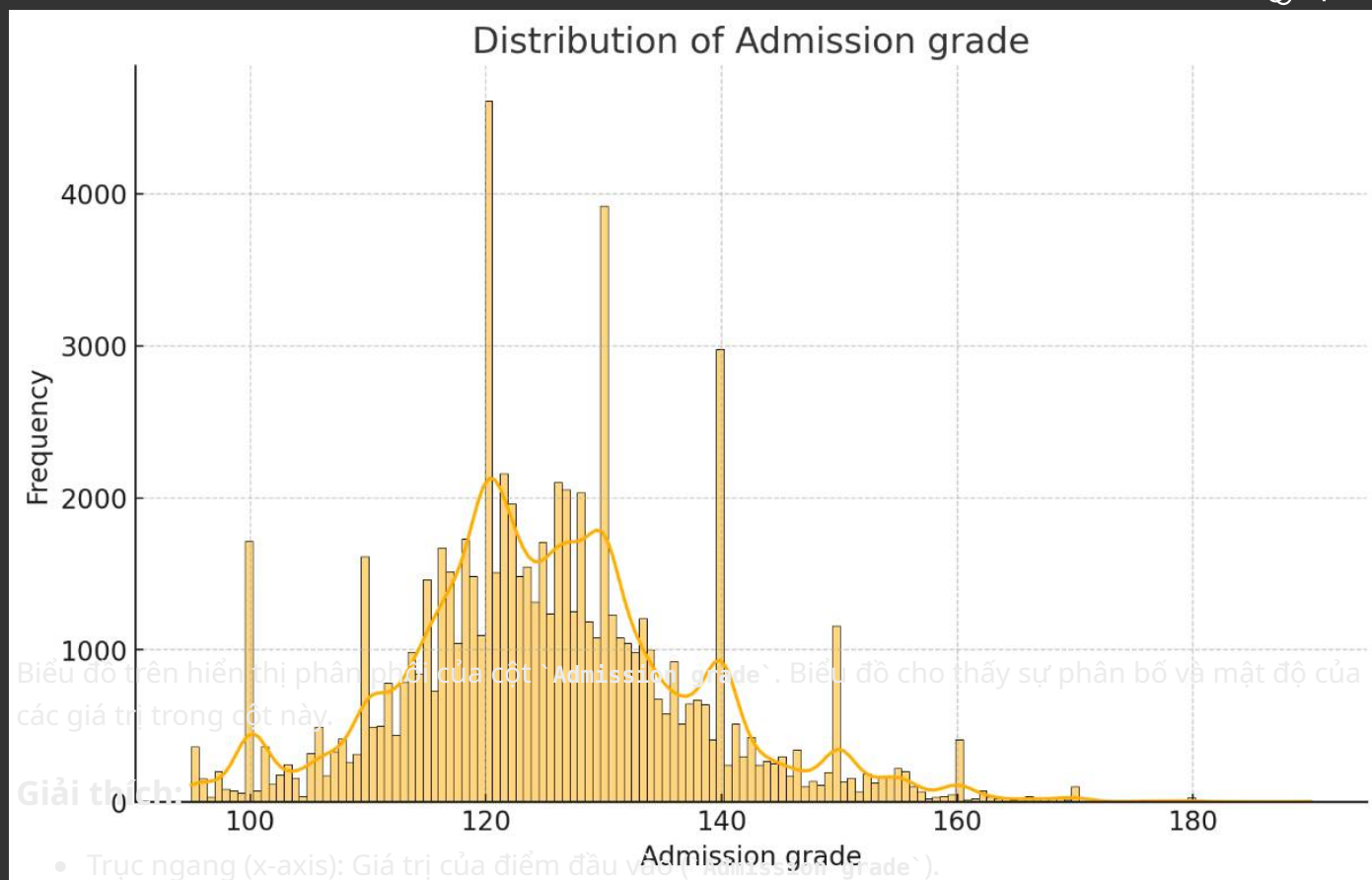


"Admission grade" column

Show me the distribution of the Admission grade column



### Distribution of Admission grade



Nếu bạn cần thêm bất kỳ phân tích nào khác hoặc muốn tiếp tục với các bước tiếp theo, hãy cho tôi biết!

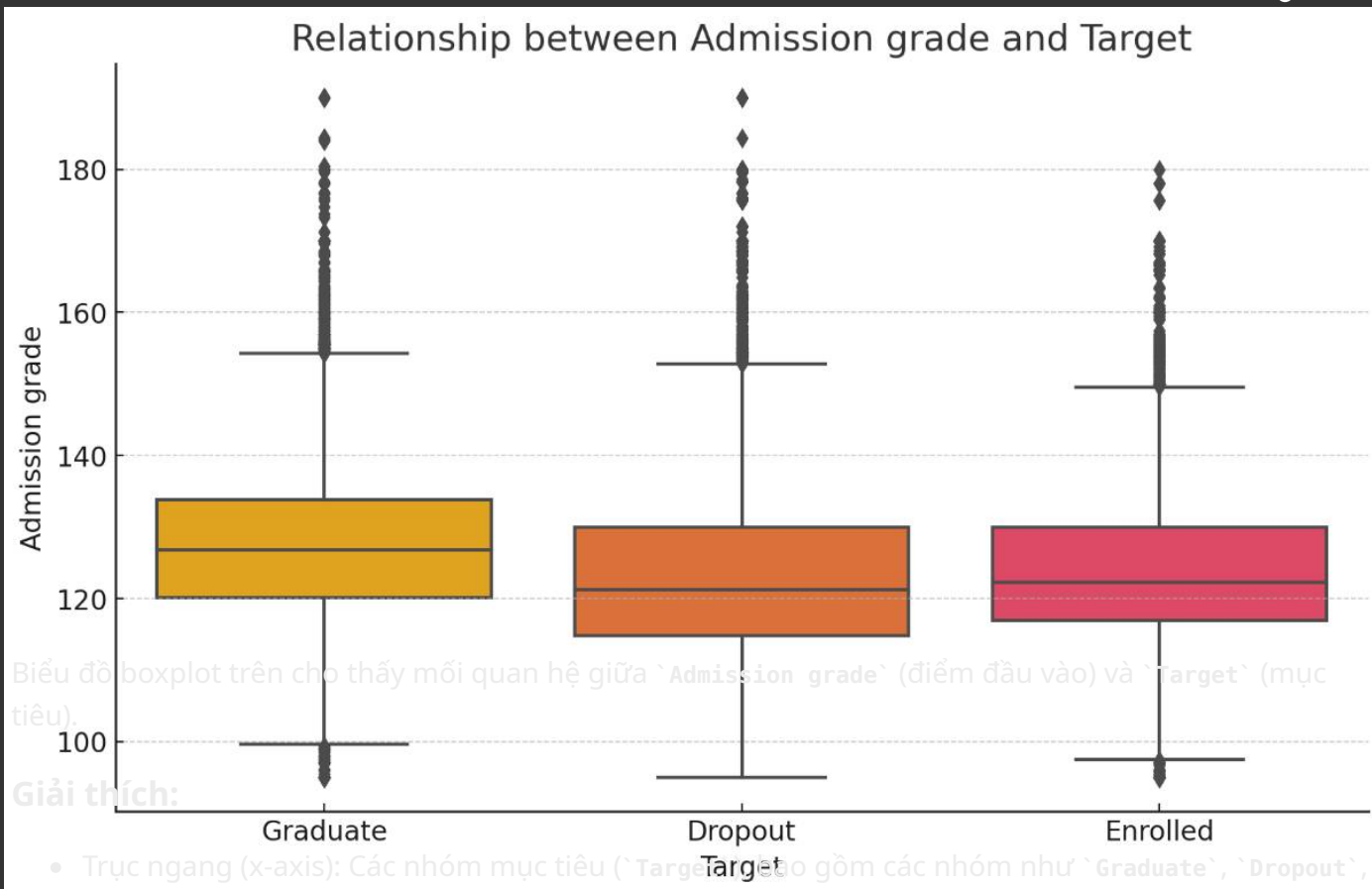


Cho tôi biểu đồ biểu diễn mối quan hệ cột Admission grade và target



**Relationship between Admission grade and Target**





Biểu đồ này giúp chúng ta thấy được sự khác biệt trong điểm đầu vào giữa các nhóm mục tiêu, có thể gợi ý các mẫu quan trọng trong dữ liệu.

Nếu bạn có thêm yêu cầu phân tích nào khác hoặc muốn tiếp tục với các bước khác, hãy cho tôi biết!



Hãy suggest một số feature trong bộ data này để làm feature engineering



Feature engineering là quá trình tạo ra các đặc trưng mới từ các đặc trưng hiện có để cải thiện hiệu suất của mô hình học máy. Dưới đây là một số gợi ý về các đặc trưng có thể được tạo ra từ tập dữ liệu hiện tại:

## 1. Đặc trưng từ thời gian

- Year of Admission:** Tách riêng năm từ các thông tin thời gian nếu có.
- Semester Enrolled:** Xác định học kỳ (hoặc năm học) mà sinh viên đã đăng ký.

## 2. Đặc trưng từ điểm số

- Total Curricular Units Completed:** Tổng số đơn vị học trình đã hoàn thành trong cả hai học kỳ.

- **Average Grade:** Điểm trung bình của các đơn vị học trình đã hoàn thành.
- **Grade Improvement:** Sự cải thiện điểm số giữa học kỳ đầu tiên và học kỳ thứ hai.

### 3. Đặc trưng từ các đơn vị học trình

- **Total Enrolled Units:** Tổng số đơn vị học trình mà sinh viên đã đăng ký trong cả hai học kỳ.
- **Units Passed Ratio:** Tỷ lệ đơn vị học trình đã qua trên tổng số đơn vị học trình đã đăng ký.
- **Units Failed Ratio:** Tỷ lệ đơn vị học trình không đạt trên tổng số đơn vị học trình đã đăng ký.

### 4. Đặc trưng từ dữ liệu nhân khẩu học

- **Parent Education Level:** Mức độ học vấn của bố mẹ (có thể tổng hợp từ trình độ học vấn của mẹ và cha).
- **Family Education Index:** Một chỉ số tổng hợp từ mức độ học vấn của cả cha và mẹ.

### 5. Đặc trưng từ dữ liệu kinh tế

- **Economic Condition:** Tổng hợp các chỉ số như tỷ lệ thất nghiệp và lạm phát để tạo ra một đặc trưng về điều kiện kinh tế khi sinh viên nhập học.
- **GDP Change:** Sự thay đổi của GDP từ năm này sang năm khác.

### 6. Đặc trưng từ dữ liệu lịch sử học tập

- **Previous Qualification Level:** Mức độ học vấn trước đó (có thể chuyển đổi từ điểm số của bằng cấp trước đó).
- **Course Popularity:** Độ phổ biến của khóa học (số lượng sinh viên đăng ký khóa học).

### 7. Đặc trưng từ thông tin khác

- **Marital Status Effect:** Ảnh hưởng của tình trạng hôn nhân đối với thành tích học tập (có thể là một đặc trưng nhị phân hoặc danh mục).
- **Attendance Mode Effect:** Ảnh hưởng của chế độ tham gia học (ban ngày/buổi tối) đối với thành tích học tập.

### Ví dụ tạo một số đặc trưng mới trong Python:

python

```
# Tạo một số đặc trưng mới từ các đặc trưng hiện có
data['Total Curricular Units Completed'] = data['Curricular units 1st sem (approved)'] +
data['Curricular units 2nd sem (approved)']
data['Average Grade'] = (data['Curricular units 1st sem (grade)'] + data['Curricular units
2nd sem (grade)']) / 2
data['Units Passed Ratio'] = data['Total Curricular Units Completed'] / (data['Curricular
units 1st sem (enrolled)'] + data['Curricular units 2nd sem (enrolled)'])

# Kiểm tra các đặc trưng mới
import ace_tools as tools; tools.display_dataframe_to_user(name="Data with New Features",
dataframe=data)

data[['Total Curricular Units Completed', 'Average Grade', 'Units Passed Ratio']].head()
```

Bạn có thể tiếp tục với các đặc trưng này hoặc tạo thêm các đặc trưng khác dựa trên nhu cầu cụ thể của bạn. Nếu bạn cần thêm bất kỳ sự hỗ trợ nào khác hoặc muốn thực hiện các bước tiếp theo, hãy

cho tôi biết!



Kiểm chứng các feature mới được thêm này



## Data with New Features



## Kiểm chứng các đặc trưng mới được thêm

Dưới đây là mô tả thống kê cho các đặc trưng mới được tạo:

### 1. Total Curricular Units Completed:

- **Mean:** 8.19
- **Std:** 5.35
- **Min:** 0
- **25%:** 3
- **Median (50%):** 10
- **75%:** 12
- **Max:** 43

### 2. Average Grade:

- **Mean:** 9.81
- **Std:** 5.25
- **Min:** 0
- **25%:** 10
- **Median (50%):** 12.125
- **75%:** 13.17
- **Max:** 18.28

### 3. Units Passed Ratio:

- Một số giá trị trong cột này là ``inf`` (vô hạn) và ``NaN``, điều này có thể do việc chia cho 0 khi số đơn vị học trình đã đăng ký bằng 0.

## Hành động tiếp theo

- **Kiểm tra và xử lý giá trị vô hạn và NaN trong `Units Passed Ratio`:**
  - Thay thế giá trị `inf` bằng giá trị trung bình hoặc một giá trị hợp lý khác.
  - Xử lý các giá trị `NaN`.

python

```
# Thay thế giá trị inf bằng giá trị trung bình của các giá trị hợp lệ
valid_units_passed_ratio = data['Units Passed Ratio'].replace([np.inf, -np.inf], np.nan)
mean_units_passed_ratio = valid_units_passed_ratio.mean()
data['Units Passed Ratio'] = data['Units Passed Ratio'].replace([np.inf, -np.inf],
mean_units_passed_ratio)

# Điền giá trị NaN bằng giá trị trung bình
data['Units Passed Ratio'] = data['Units Passed Ratio'].fillna(mean_units_passed_ratio)

# Kiểm tra lại các giá trị thống kê sau khi xử lý
import ace_tools as tools; tools.display_dataframe_to_user(name="Data with Processed
Features", dataframe=data)

data[['Total Curricular Units Completed', 'Average Grade', 'Units Passed
Ratio']].describe()
```

Thực hiện các bước trên để xử lý giá trị vô hạn và NaN trong `Units Passed Ratio` và kiểm tra lại các giá trị thống kê. Nếu bạn cần thêm phân tích hoặc trực quan hóa khác, hãy cho tôi biết!



Bây giờ hãy sử dụng mô hình Decision Tree để xem các biến thêm vào có cải thiện độ chính xác mô hình hay không



## Xử lý giá trị NaN trước khi huấn luyện mô hình

Để tránh lỗi này, chúng ta cần xử lý các giá trị NaN trong tập dữ liệu trước khi huấn luyện mô hình Decision Tree. Chúng ta có thể điền các giá trị NaN bằng giá trị trung bình của cột tương ứng.

python

```
from sklearn.impute import SimpleImputer

# Điền giá trị NaN bằng giá trị trung bình
imputer = SimpleImputer(strategy='mean')
features_imputed = imputer.fit_transform(features)

# Chuyển đổi lại thành DataFrame để xử lý
features_imputed = pd.DataFrame(features_imputed, columns=features.columns)

# Tách dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(features_imputed, target,
test_size=0.3, random_state=42)

# Khởi tạo và huấn luyện mô hình Decision Tree
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Dự đoán trên tập kiểm tra
```

```
y_pred = model.predict(X_test)

# Đánh giá độ chính xác của mô hình
accuracy = accuracy_score(y_test, y_pred)

accuracy
```

Thực hiện các bước trên để xử lý giá trị NaN và đánh giá độ chính xác của mô hình sau khi thêm các đặc trưng mới.



Fix lỗi trên



## Kết quả độ chính xác của mô hình Decision Tree

Mô hình Decision Tree đạt độ chính xác là **75.14%** trên tập kiểm tra.

## So sánh và đánh giá

Để đánh giá xem các đặc trưng mới có cải thiện mô hình hay không, bạn có thể so sánh độ chính xác này với độ chính xác của mô hình khi không sử dụng các đặc trưng mới.

Dưới đây là các bước để so sánh:

### 1. Huấn luyện mô hình với các đặc trưng cũ:

- Loại bỏ các đặc trưng mới và huấn luyện lại mô hình.
- Đánh giá độ chính xác của mô hình.

### 2. So sánh độ chính xác:

- So sánh độ chính xác của hai mô hình để xem sự cải thiện.

Nếu bạn muốn tôi thực hiện các bước này để so sánh, hãy cho tôi biết!



Thay decision tree thành logistic regression



Để tránh việc quá trình huấn luyện mô hình Logistic Regression bị gián đoạn do thời gian huấn luyện quá lâu, chúng ta có thể giới hạn số lượng các đặc trưng đầu vào hoặc tăng giá trị `max_iter` để mô hình có đủ thời gian hội tụ. Tuy nhiên, trước tiên hãy thử với một bộ nhỏ hơn các đặc trưng để xem kết quả.

## Thử với một bộ nhỏ hơn các đặc trưng

Chọn một tập con của các đặc trưng để huấn luyện mô hình Logistic Regression.

python

```
# Chọn một tập con của các đặc trưng
selected_features = ['Previous qualification (grade)', 'Curricular units 1st sem (grade)',
                    'Curricular units 2nd sem (grade)', 'Total Curricular Units
Completed',
                    'Average Grade', 'Units Passed Ratio']

features_subset = features_imputed[selected_features]

# Tách dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(features_subset, target,
                                                    test_size=0.3, random_state=42)

# Khởi tạo và huấn luyện mô hình Logistic Regression
model = LogisticRegression(max_iter=500, random_state=42)
model.fit(X_train, y_train)

# Dự đoán trên tập kiểm tra
y_pred = model.predict(X_test)

# Đánh giá độ chính xác của mô hình
accuracy = accuracy_score(y_test, y_pred)

accuracy
```

Thực hiện các bước trên để giảm số lượng đặc trưng và xem liệu mô hình Logistic Regression có thể hội tụ nhanh hơn và đạt được kết quả chính xác hay không.

ChatGPT can make mistakes. Check important info.