TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG CÔNG THÔNG TIN

BÁO CÁO ĐỒ ÁN 3

Data Fitting

Chủ đề: Toán ứng dụng – thống kê

Lớp: 22CLC01

Sinh viên: **Trương Thuận Kiệt – 22127224**

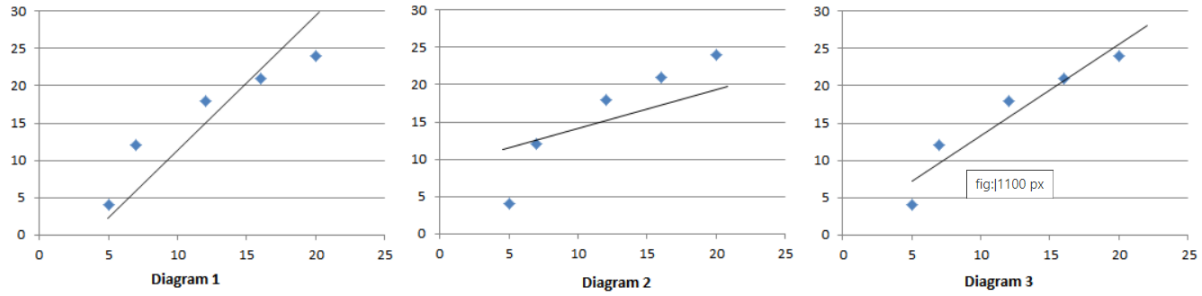
TP. Hồ Chí Minh, tháng 8 năm 2024

Mục lục

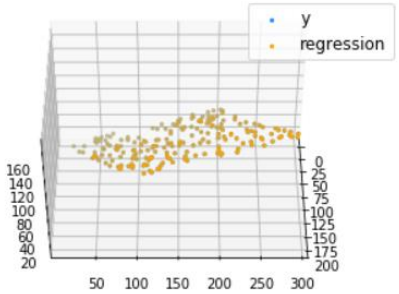
1. Giới thiệu về đồ án

a. Linear Regression

- Đây là một loại thuật toán machine learning để tính toán quan hệ tuyến tính giữa các biến phụ thuộc lẫn nhau và 1 hoặc nhiều tính năng độc lập bằng cách fit vào linear equation với dữ liệu từ tập training
- Khi có 1 feature thì được gọi là Simple Linear Regression và khi có nhiều hơn 1 features thì được gọi là Multiple Linear Regression
- Trong đó, Simple Linear Regression có dạng phương trình như sau:
  - Y = a + bx
  - Y: là giá trị cần dự đoán và là biến phụ thuộc
  - a: là intercept (Hệ số chặn)
  - b: là hệ số của x
  - x: biến độc lập
  - Ví dụ về đồ thị:



- Kế đến, Multiple Linear Regression có dạng phương trình như sau:
  - Y = b0 + b1 * x1 + b2 * x2 + b3 * x3 + …… bn * xn
  - Y: là giá trị cần dự đoán và là biến phụ thuộc
  - a: là intercept (Hệ số chặn)
  - b_i: là các hệ số của x_i
  - x_i: là các biến độc lập
  - Ví dụ về đồ thị:

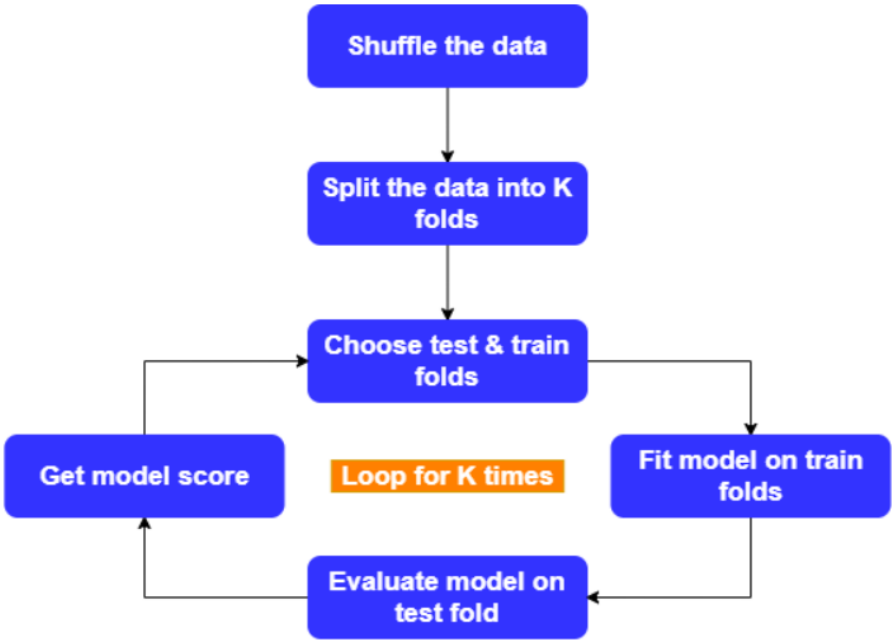


b. MAE (Mean absolute error)

- Là sai số tuyệt đối trung bình để đánh giá độ chính xác của mô hình giữa giá trị dự đoán và thực tế
- Công thức: $MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
  - N: số lần quan sát (Ở trong project này thì n sẽ tương ứng với k fold mà sẽ nói ở dưới và k ở đây sẽ mặc định là 5)
  - y_i: giá trị thực tế của y
  - $\hat{y}_i$: giá trị được dự đoán của y

c. K-Fold Cross Validation

- Trong đó, chúng ta sẽ chia ra làm 2 phần **K-Fold** và **Cross Validation**
- **K-Fold** dùng để chia data training thành k số tập nhỏ hơn để xác thực model. Sau đó model sẽ được train lại trên k-1 folds của data training, còn phần còn lại được dùng như tập validation để đánh giá model
- **Cross validation** dùng để train model trên tập train và evaluate nó trên tập test và lần lượt như vậy cho hết k fold, mỗi lần như vậy sẽ chọn fold riêng biệt cho test data, cuối cùng lấy sum MAE và chia trung bình
- **Mô hình hoạt động:**



- Vậy từ đó, chúng ta có thể thấy được K-Fold-Cross-Validation giúp khái quát hóa model từ đó có thể đưa ra dự đoán tốt hơn về data
- **Mô hình hoạt động:**

d. Ý tưởng chung cho việc tìm ra mô hình tốt nhất

- Đầu tiên ta sẽ shuffle training data
- Sau đó chia training data thành k nhóm nhỏ với các X_train và y_Train
- Với bộ dữ liệu X_train và y_Train với mô hình M_i ta sẽ đi tìm giá trị x trong phương trình ma trận X_train * x ≈ y_train
- Sau khi tìm ra x thì dùng x tính MAE bằng X_test và y_test của mô hình M_i
- Sau khi tính toán thì so sánh độ lỗi MAE của các mô hình để tìm ra mô hình có MAE thấp nhất nghĩa là mô hình đó có độ tối ưu nhất

2. Các thư viện đã sử dụng và mục đích sử dụng

a. NumPy

- Dùng để xử lý các tính toán liên quan đến ma trận

b. Pandas

- Dùng để đọc dữ liệu từ file và biểu thị bảng so sánh MAE giữa các mô hình

c. Maplotlib

- Dùng để xây dựng và hiển thị đồ thị

d. Seaborn

- Dùng để vẽ biểu đồ thể hiện mối quan hệ giữa các features

3. Ý nghĩa của các hàm/class tự cài đặt và từ thư viện có sẵn đã sử dụng

a. Class OLSLinearRegression

- **Hàm fit(self, X, y):**
  - **Ý nghĩa**

Dùng để tìm nghiệm thông qua việc huấn luyện model linear regression theo cách OLS (Ordinary Least Squares)

- **Input:**
  X: Là ma trận đặc trưng (feature matrix), trong đó mỗi hàng là một mẫu (sample) và mỗi cột là một đặc trưng (feature).
  y: Là vector mục tiêu (target vector), chứa các giá trị đầu ra tương ứng với các mẫu trong X.
- **Output:**
  Mô hình đã huấn luyện (self), với trọng số w được lưu trữ trong đối tượng.
- **Các bước thực hiện:**
  X.T @ X: Tính tích vô hướng của ma trận X với chính nó (transpose). Đây là một phần của công thức OLS để tìm trọng số w.
  np.linalg.inv(X.T @ X): Tính nghịch đảo của ma trận này. Lưu ý rằng ma trận này phải không suy biến (invertible).
  X_pinv: Tính nghịch đảo Moore-Penrose của X để tìm trọng số w
  self.w = X_pinv @ y: Tính trọng số w bằng cách nhân nghịch đảo với y. Đây là trọng số của mô hình sau khi huấn luyện.

- **Hàm get_params(self):**
  - **Ý nghĩa:**
    Trả về trọng số w đã được tính trong hàm fit
  - **Output:**
    Trọng số w
- **Hàm predict(self, X):**
  - **Ý nghĩa:**
    Dùng để dự đoán đầu ra **y_hat** từ ma trận đặc trưng **X** dựa trên trọng số **w** đã tính
  - **Input:**
    X: ma trận đặc trưng
  - **Output:**
    Y_hat: giá trị dự đoán
  - **Các bước thực hiện:**
    X @ self.w: Tính tích vô hướng của ma trận X với trọng số w để dự đoán giá trị y_hat

b. Hàm mae(y, y_hat)

- **Ý nghĩa:**
  Dùng để tính sai số tuyệt đối trung bình (MAE) giữa giá trị thực tế y và giá trị dự đoán y_hat.
- **Input**:
  Y: giá trị thực tế (Y_test)
- **Output:**
  Gía trị mae

- **Các bước thực hiện:**

  np.abs(y.ravel() - y_hat.ravel()): Tính sai số tuyệt đối giữa y và y_hat, đồng thời chuyển y và y_hat thành vector một chiều bằng ravel()
  np.mean(...): Tính giá trị trung bình của sai số tuyệt đối

c. Hàm preprocess(X)

- **Ý nghĩa:**

  Chuẩn bị dữ liệu đầu vào bằng cách thêm một cột 1 vào ma trận đặc trưng X. Điều này cần thiết cho mô hình hồi quy tuyến tính, vì cột 1 đại diện cho hệ số chặn (bias) trong phương trình tuyến tính
- **Ví dụ:**

Trước khi thêm cột bias	Sau khi thêm cột bias
[[79] [65] [60] ... [44] [98] [92]]	[[1. 79.] [1. 65.] [1. 60.] ... [1. 44.] [1. 98.] [1. 92.]]

- **Input:**

  X: ma trận đặc trưng
- **Output:**

  X: ma trận X ban đầu nhưng đã thêm cột 1
- **Các bước thực hiện:**

  np.hstack((np.ones((X.shape[0], 1)), X)): Ghép thêm một cột 1 vào trước ma trận X

d. Hàm k_fold_cross_validation(X, k=5):

- **Ý nghĩa:**

  Dùng để chia tập data thành k phần bằng nhau và sau đó thực hiện quy trình training và test k lần
- **Input:**

  X : np.ndarray:
  Ma trận dữ liệu đầu vào, mỗi hàng là một mẫu và mỗi cột là một đặc trưng.
  k : int, optional:
  Số lượng fold để chia dữ liệu (mặc định là 5).
- **Output:**

  List[Tuple[np.ndarray, np.ndarray]]:
  Danh sách chứa k phần tử, mỗi phần tử là một tuple (train_indices, test_indices) với:

+ train_indices: Mảng chứa các chỉ số của các mẫu dữ liệu được dùng để huấn luyện mô hình.
+ test_indices: Mảng chứa các chỉ số của các mẫu dữ liệu được dùng để kiểm tra mô hình.

- **Các bước thực hiện:**
  indices = np.arange(n): Tạo một mảng indices chứa các chỉ số từ 0 đến n-1, tương ứng với các mẫu trong tập dữ liệu.
  train_indices = np.concatenate([indices[:i * fold_size], indices[(i + 1) * fold_size:]]): Phần còn lại của dữ liệu (tất cả các chỉ số không nằm trong test_indices) được chọn làm tập huấn luyện (train_indices). Các chỉ số này được ghép lại với nhau bằng np.concatenate

e. Model1(train), Model2(train), Model3(train), Model4(train):

- **Ý nghĩa:**
  Dùng để tạo mô hình mới thông qua tạo ra các tập dữ liệu huấn luyện khác nhau bằng cách lựa chọn và chuyển đổi các đặc trưng từ tập dữ liệu gốc
- **Input:**
  train (pd.DataFrame): DataFrame chứa dữ liệu đầu vào, với mỗi hàng là một mẫu và các cột đại diện cho các đặc trưng và mục tiêu
- **Output:**
  X (np.ndarray): Ma trận numpy chứa đặc trưng đã được chọn và chuyển đổi.

f. Hàm pd.read_csv (Hàm từ thư viện numPy):

- **Ý nghĩa:**
  Dùng để đọc nội dung từ tập tin dạng .csv
- **Input:**
  Link của tập
- **Output:**
  Nội dung của tập tin

g. Hàm sns.heatmap (Hàm từ thư viện seaborn):

- **Ý nghĩa:**
  Dùng để vẽ ra biểu thị mối quan hệ giữa các features, Các ô vuông trên heatmap sẽ có màu sắc tương ứng với giá trị tương quan:
  + Màu sắc càng đậm hoặc sáng thể hiện mối tương quan càng mạnh (dương hoặc âm).
  + Màu sắc nhạt hoặc trung tính thể hiện mối tương quan yếu hoặc không tồn tại.
- **Input:**
  Train.corr(): Tương quan là một thước đo thống kê thể hiện mối quan hệ giữa hai biến số, thường có giá trị trong khoảng từ -1 đến 1.
  + Giá trị +1 thể hiện mối tương quan dương hoàn hảo (khi một biến tăng, biến kia cũng tăng).

+ Giá trị -1 thể hiện mối tương quan âm hoàn hảo (khi một biến tăng, biến kia giảm).
+ Giá trị 0 thể hiện rằng không có mối tương quan giữa hai biến.

- **Output:**
  Biểu đồ nhiệt

h. Hàm sns.pairplot (Hàm từ thư viện seaborn):

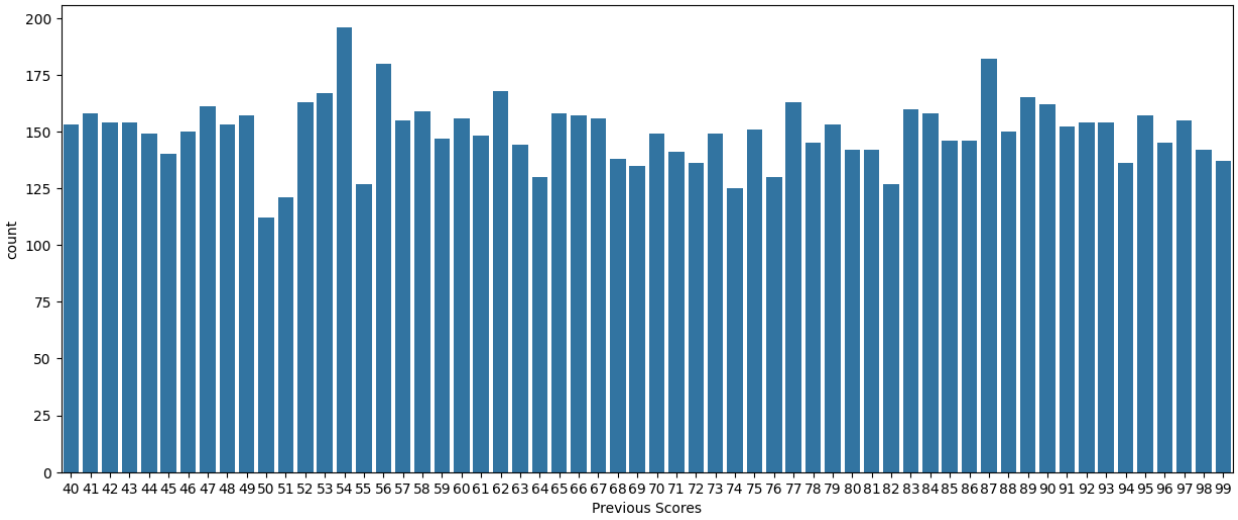
- **Ý nghĩa:**
  Tạo ra một lưới các biểu đồ, trong đó:
  + **Trên đường chéo chính**: Mỗi ô sẽ hiển thị biểu đồ phân phối (histogram hoặc KDE) của từng biến số trong DataFrame train.
  + **Ngoài đường chéo chính**: Mỗi ô sẽ là một biểu đồ phân tán (scatter plot) giữa hai biến số khác nhau trong DataFrame
- **Input**:
  Train: dataset
- **Output**:
  Biểu đồ

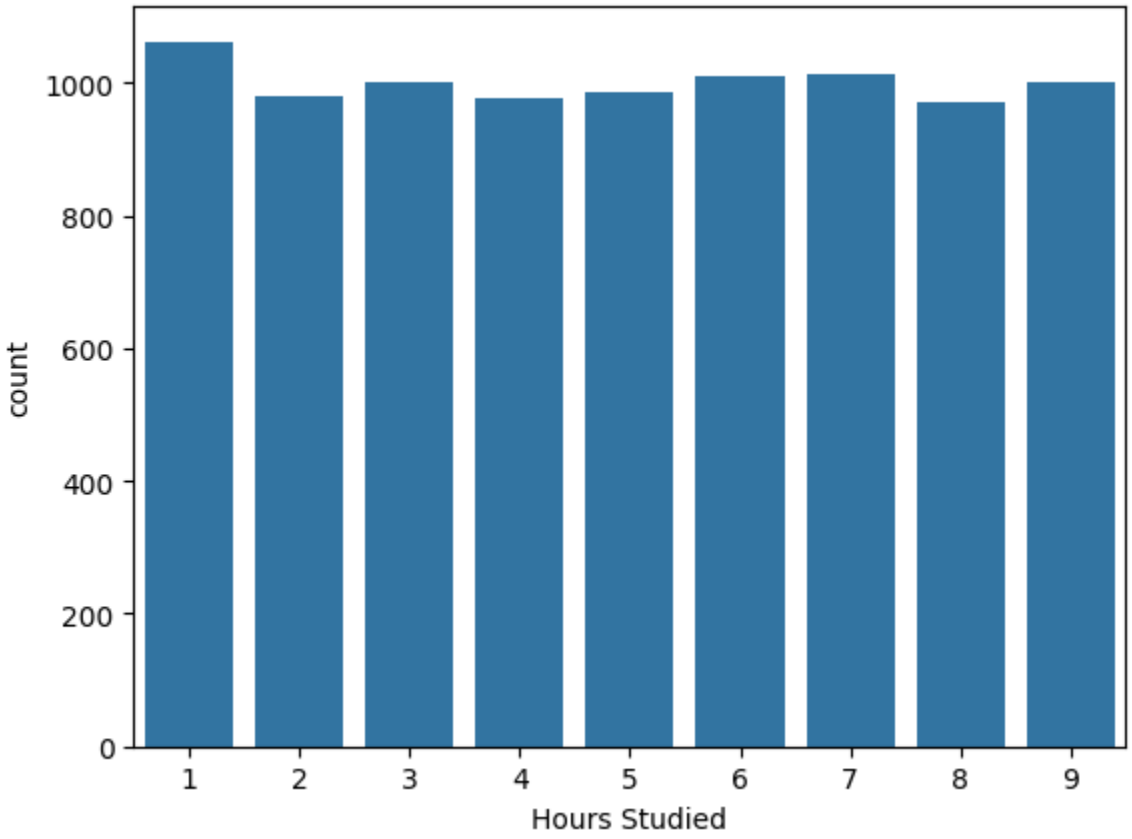
i. Hàm sns.barplot (Hàm từ thư viện seaborn):

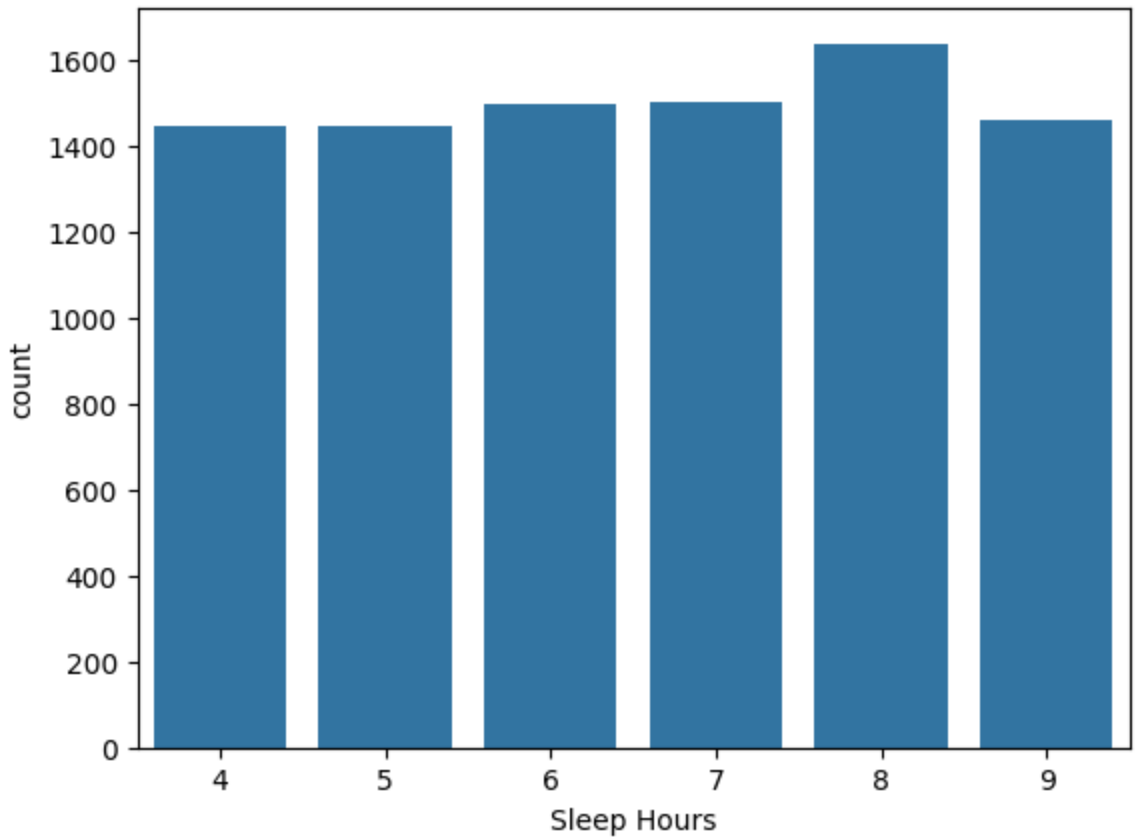
- **Ý nghĩa:**
  Tạo ra đồ thị cột, ngoài ra nhằm só sánh giá trị của các feature có sự giao động mạnh và nhiều
- **Input:**
  X, y: features muốn biểu thị
  Data: dataset
- **Output:**
  Đồ thị dạng cột

j. Hàm sns.countplot (Hàm từ thư viện seaborn):

- **Ý nghĩa:**
  Nhằm cho thấy sự phân phối tần suất của các giá trị của một feature nhất định **(Dùng cho feature Previous Scores, Hours Studied và Sleep Hours)**
- **Input:**
  X: feature muốn thể hiện
- **Output:**
  Đồ thị dạng cột

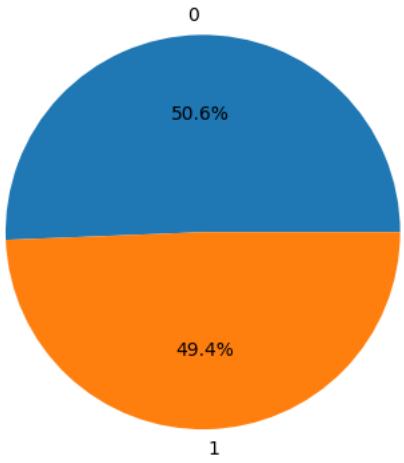

count
Previous Scores

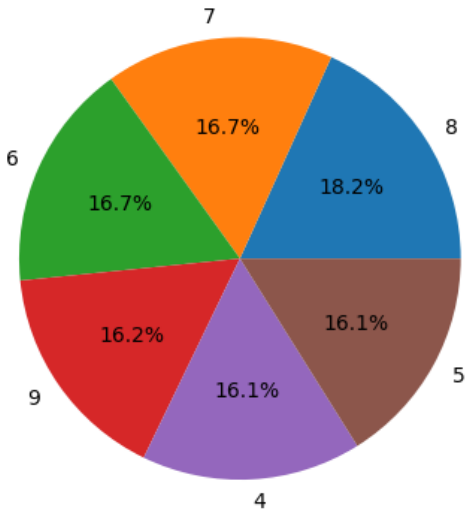

count
Hours Studied



k. Hàm plt.pie (Hàm từ thư viện mathplotlib):

- **Ý nghĩa:**
  Dùng để vẽ biểu đồ tròn, hiển thị tỷ lệ phần trăm của các giá trị khác nhau, ngoài ra dùng trực quan hóa tỷ lệ các phần khác nhau trong tổng thể **(Dùng cho feature Extracurricular Activities và Sleep Hours)**
- **Input:**
  Số lượng giá trị của một feature
- **Output:**
  Biểu đồ tròn





l. Hàm sns.boxplot (Hàm từ thư viện seaborn):

- **Ý nghĩa:**
  Vẽ một biểu đồ hộp, trong đó:
  + Mỗi hộp biểu diễn phân phối của 1 feature cho một giá trị cụ thể của 1 feature khác.
  + Hộp biểu diễn khoảng từ phần tư thứ nhất (Q1) đến phần tư thứ ba (Q3) của dữ liệu, với một đường nằm ngang bên trong hộp biểu diễn giá trị trung vị (median).
  + Các "râu" (whiskers) của hộp kéo dài từ Q1 - 1.5 * IQR (interquartile range) đến Q3 + 1.5 * IQR, biểu diễn phạm vi dữ liệu mà không có các giá trị ngoại lai.
  + Các điểm nằm ngoài các "râu" là các giá trị ngoại lai (outliers).
- **Input**:
  X,y: Các features muốn hiển thị

- **Output**:
  Biểu đồ hộp

4. Kết quả và nhận xét

4.1 Yêu cầu 1

a. Thông tin cơ bản của data

- Đầu tiên, chúng ta sẽ cần biết sơ lược thông tin về data:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9000 entries, 0 to 8999
Data columns (total 6 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Hours Studied                   9000 non-null   int64
 1   Previous Scores                 9000 non-null   int64
 2   Extracurricular Activities      9000 non-null   int64
 3   Sleep Hours                     9000 non-null   int64
 4   Sample Question Papers Practiced  9000 non-null   int64
 5   Performance Index               9000 non-null   float64
dtypes: float64(1), int64(5)
```

- Ở đây, ta có thể thấy được rằng data có tổng cộng **6 cột, 9000 dòng** và **không có bất kì giá trị null nào**, trong đó tên của các cột sẽ là **Hours Studied Previous Scores, Extracurricular Activities, Sleep Hours, Sample Question Papers Practiced, Performance Index**

b. Thông tin chi tiết của data

- Sau đó, chúng ta sẽ tìm hiểu thêm về giá trị của từng feature:

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
count	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000	9000.000000
mean	4.976444	69.396111	0.493667	6.535556	4.590889	55.136333
std	2.594647	17.369957	0.499988	1.695533	2.864570	19.187669
min	1.000000	40.000000	0.000000	4.000000	0.000000	10.000000
25%	3.000000	54.000000	0.000000	5.000000	2.000000	40.000000
50%	5.000000	69.000000	0.000000	7.000000	5.000000	55.000000
75%	7.000000	85.000000	1.000000	8.000000	7.000000	70.000000
max	9.000000	99.000000	1.000000	9.000000	9.000000	100.000000

- **Hours Studied:**
  Trung bình (mean): 4.98 giờ.
  Độ lệch chuẩn (std): 2.59, cho thấy mức độ biến động vừa phải trong số giờ học giữa các học sinh.
  Phạm vi giá trị: Từ 1 đến 9 giờ.
  Phân vị 50% (median): 5 giờ, tức là 50% học sinh học dưới 5 giờ và 50% học sinh học trên 5 giờ.

=> Nhận xét: Số giờ học của học sinh phân bố khá đều, với mức trung bình gần 5 giờ. Đa số học sinh học từ 3 đến 7 giờ (từ phân vị 25% đến 75%).

- **Previous Scores:**
  Trung bình: 69.4 điểm.
  Độ lệch chuẩn: 17.37, cho thấy sự biến động khá lớn trong điểm số của học sinh.
  Phạm vi giá trị: Từ 40 đến 99 điểm.
  Phân vị 50% (median): 69 điểm.
  => Nhận xét: Điểm số trước đó của học sinh tập trung chủ yếu quanh mức 69, nhưng có sự phân tán khá rộng, với một số học sinh có điểm rất thấp hoặc rất cao.
- **Extracurricular Activities:**
  Trung bình: 0.49, gần bằng 0.5, cho thấy số lượng học sinh tham gia và không tham gia hoạt động ngoại khóa gần như bằng nhau.
  Phân vị 50%: 0, tức là ít nhất 50% học sinh không tham gia hoạt động ngoại khóa.
  Phạm vi giá trị: Giá trị tối thiểu là 0 và tối đa là 1, thể hiện chỉ có hai giá trị cho biến này (có hoặc không tham gia).
  => Nhận xét: Có sự cân bằng giữa số lượng học sinh tham gia và không tham gia hoạt động ngoại khóa.
- **Sleep Hours:**
  Trung bình: 6.54 giờ.
  Độ lệch chuẩn: 1.70, cho thấy có sự biến động vừa phải về số giờ ngủ.
  Phân vị 50%: 7 giờ, tức là hầu hết học sinh ngủ từ 5 đến 8 giờ.
  Phạm vi giá trị: Từ 4 đến 9 giờ.
  => Nhận xét: Phần lớn học sinh ngủ từ 5 đến 8 giờ mỗi ngày, với mức trung bình là 6.54 giờ.
- **Sample Question Papers Practiced:**
  Trung bình: 4.59 đề mẫu.
  Độ lệch chuẩn: 2.86, cho thấy sự biến động khá lớn về số lượng đề mẫu mà học sinh đã luyện tập.
  Phân vị 50%: 5 đề mẫu, tức là đa số học sinh luyện tập từ 2 đến 7 đề mẫu.
  Phạm vi giá trị: Từ 0 đến 9 đề mẫu.
  => Nhận xét: Số lượng đề mẫu mà học sinh đã luyện tập khá đa dạng, với một số học sinh không luyện tập đề mẫu nào và một số khác đã luyện tập tối đa 9 đề mẫu.
- **Performance Index:**
  Trung bình: 55.14 điểm.
  Độ lệch chuẩn: 19.19, cho thấy sự biến động khá lớn về chỉ số hiệu suất.

Phân vị 50%: 55 điểm, tức là 50% học sinh có chỉ số hiệu suất dưới 55 điểm và 50% có chỉ số hiệu suất trên 55 điểm.

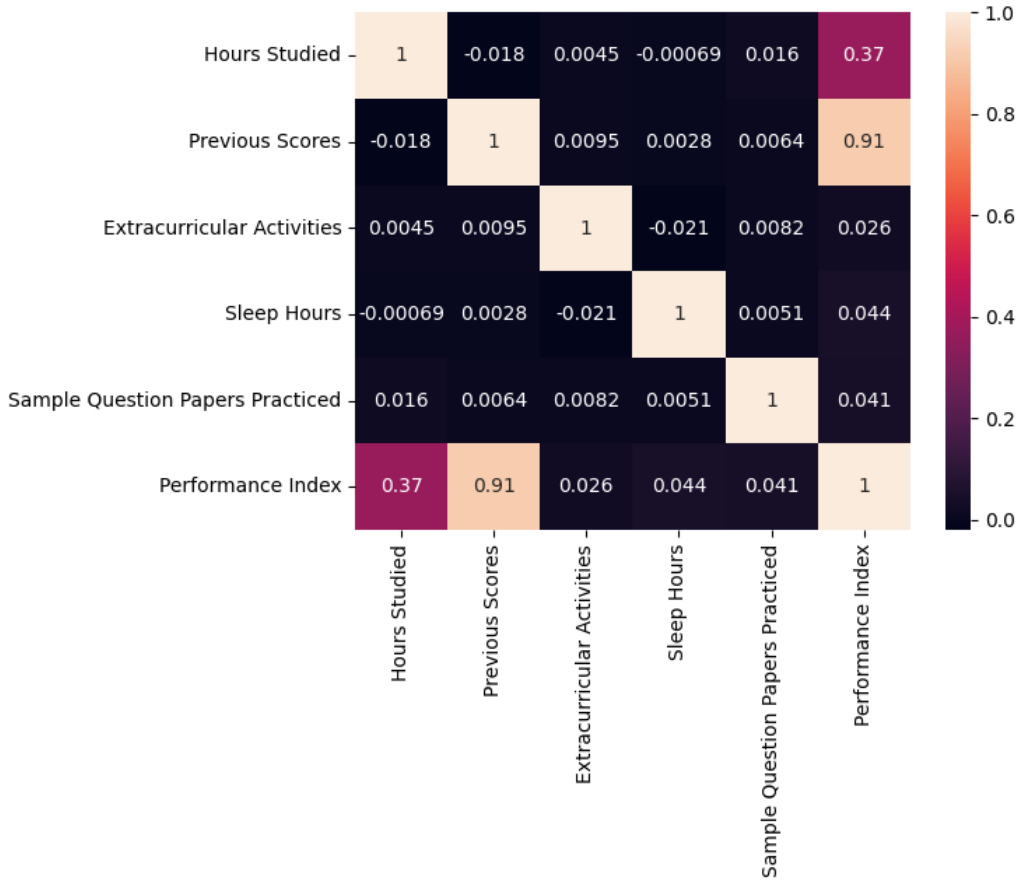
Phạm vi giá trị: Từ 10 đến 100 điểm.

=> Nhận xét: Chỉ số hiệu suất của học sinh phân bố rộng, từ 10 đến 100 điểm, với mức trung bình là 55.14 điểm. Độ lệch chuẩn cao cho thấy sự khác biệt đáng kể trong hiệu suất của học sinh.

- **Tổng quan:**

  Các biến trong tập dữ liệu đều có sự phân bố rõ ràng, với một số biến như **Hours Studied, Previous Scores, Sample Question Papers Practiced và Sleep Hours** có sự phân tán đáng kể, cho thấy rằng có sự khác biệt lớn giữa các học sinh trong các khía cạnh này. Điều này có thể ảnh hưởng đến mô hình dự đoán và cần được xem xét kỹ lưỡng trong quá trình phân tích dữ liệu.

- Kế đến, chúng ta sẽ dùng các đồ thị để có cái nhìn tổng quát hơn về correlation giữa các feature:



- **Mức độ tương quan giữa các biến:**

  Previous Scores và Performance Index có mức độ tương quan rất cao, với hệ số tương quan lên đến 0.91.

⇨ **Điều này cho thấy rằng điểm số trước đó có ảnh hưởng rất lớn đến Performance Index, và có thể dự đoán chỉ số hiệu suất khá chính xác chỉ dựa trên điểm số trước đó.**

Hours Studied cũng có mức độ tương quan đáng kể với Performance Index, với hệ số tương quan 0.37.

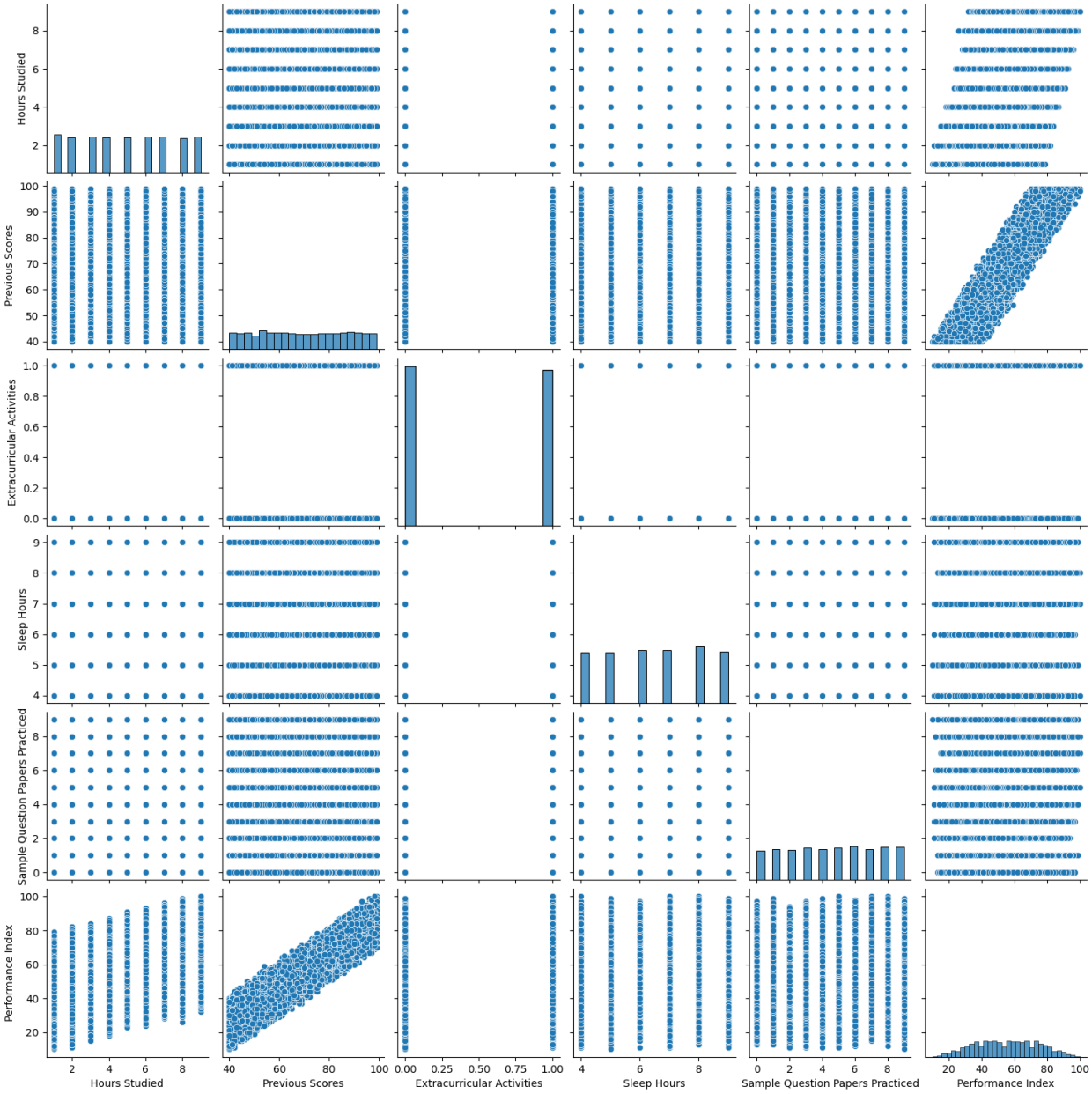
⇨ **Điều này chỉ ra rằng số giờ học cũng là một yếu tố quan trọng ảnh hưởng đến hiệu suất học tập, mặc dù không mạnh bằng điểm số trước đó.**

- **Mối tương quan thấp hoặc không đáng kể:**
  Các biến như Extracurricular Activities, Sleep Hours và Sample Question Papers Practiced đều có mối tương quan rất thấp với Performance Index, với các hệ số tương quan lần lượt là 0.026, 0.044, và 0.041.

  ⇨ **Điều này cho thấy những yếu tố này không đóng vai trò quan trọng trong việc dự đoán chỉ số hiệu suất của học sinh.**

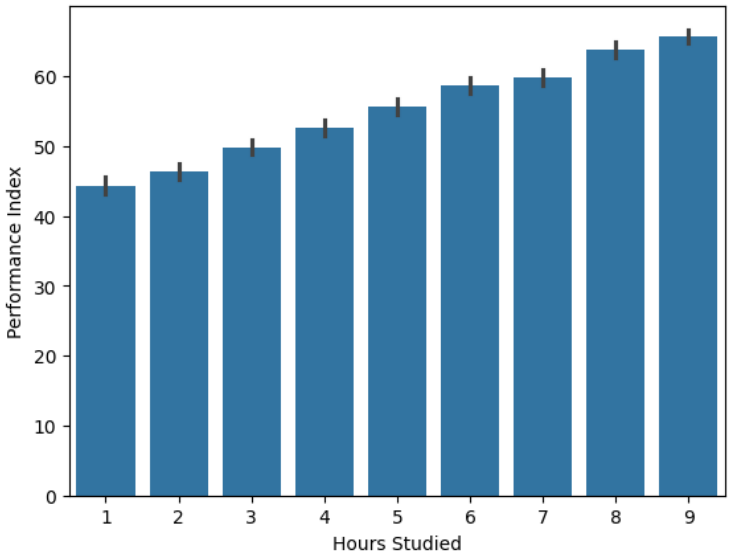
  Mức độ tương quan giữa các biến khác (ví dụ: giữa Hours Studied và Previous Scores) cũng rất thấp, với hệ số tương quan gần bằng 0, cho thấy các biến này hoạt động khá độc lập với nhau.
- **Tính đa cộng tuyến:**
  Không có dấu hiệu rõ ràng của tính đa cộng tuyến giữa các biến, ngoại trừ mối quan hệ giữa Previous Scores và Performance Index. Điều này gợi ý rằng các biến còn lại có thể không ảnh hưởng quá mạnh mẽ lẫn nhau, ngoại trừ trường hợp trên.
- **Tổng kết:**
  Previous Scores là yếu tố dự đoán mạnh nhất đối với Performance Index, theo sau đó là Hours Studied.
  Các yếu tố khác như hoạt động ngoại khóa, số giờ ngủ, và số lượng đề mẫu luyện tập có tác động rất nhỏ đến chỉ số hiệu suất và có thể ít quan trọng hơn trong việc xây dựng mô hình dự đoán hiệu suất học tập.

- Cuối cùng, để có thể củng cố những thông tin mà chúng ta có được chúng ta tiếp tục sử dụng pairplot để có cái nhìn chi tiết hơn về quan hệ của từng cặp feature

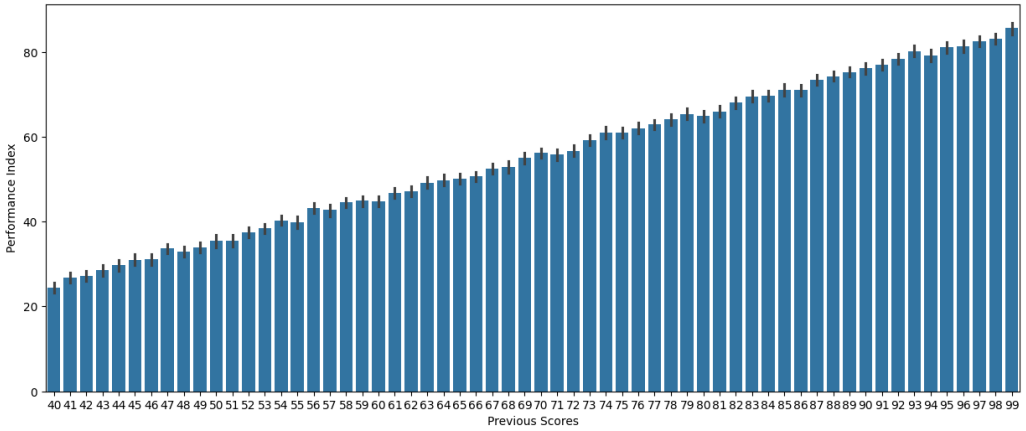


c. Phân tích các feature riêng biệt

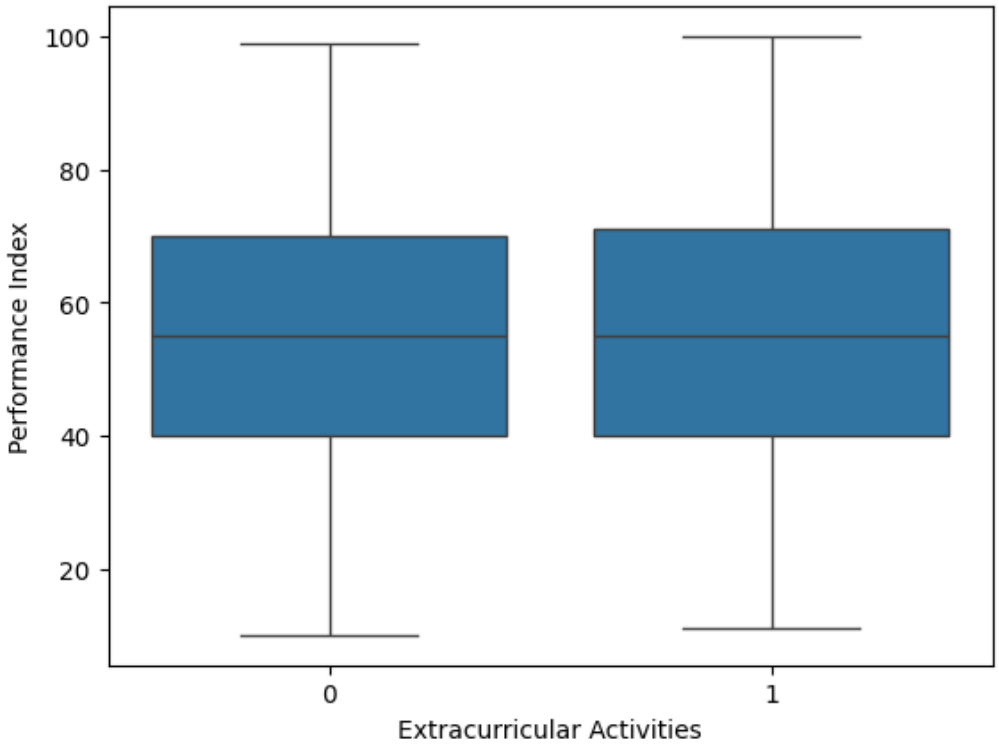
- **Feature Hours Studied**



  - Từ biểu đồ ta có thể thấy được sự phân bố khá đều giữa các Hours Studied và Performance Index, tuy nhiên một điều dễ nhận ra chính là Hours Studied càng cao thì đồng nghĩa đến việc Performance Index cũng cao
- **Feature Previous Scores**

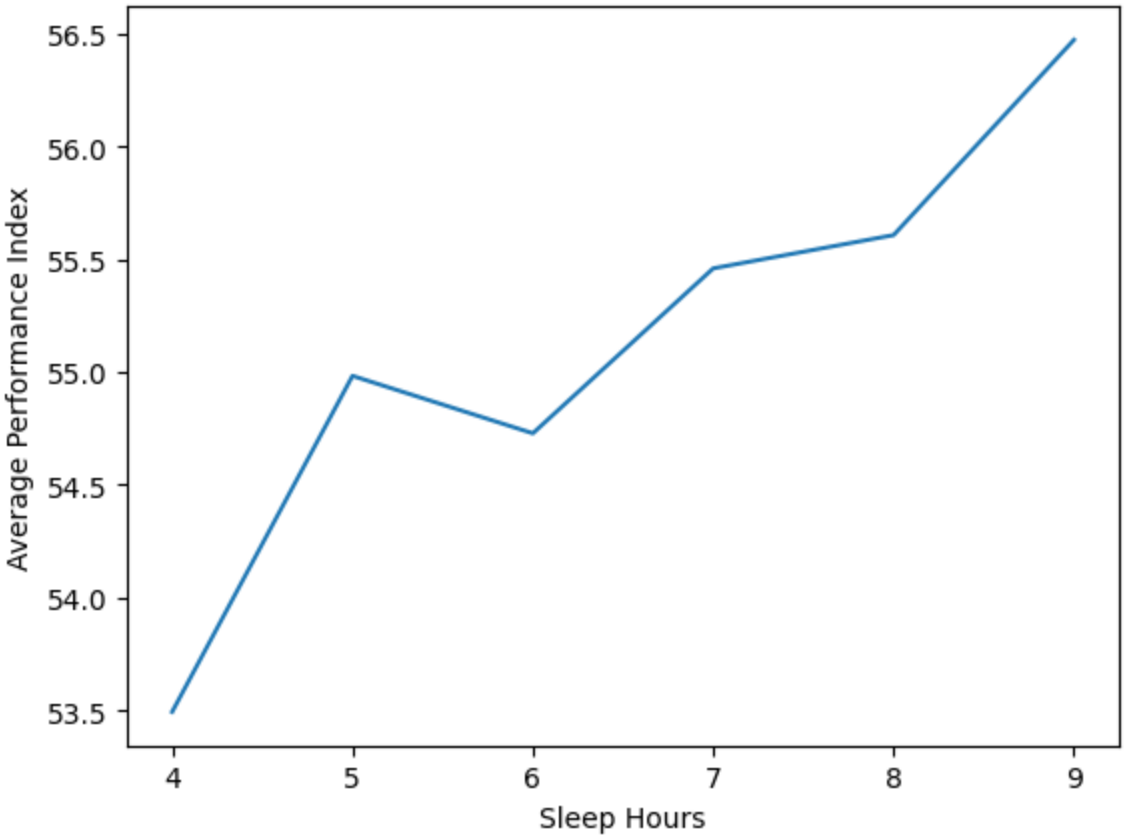


  - Từ biểu đồ ta có thể cũng có thể thấy được sự phân bố khá đều giữa Previous Scores và Performance Index, và đồng thời Previous Scores thì Performance Index càng cao theo
- **Feature Extracurricular Activities**



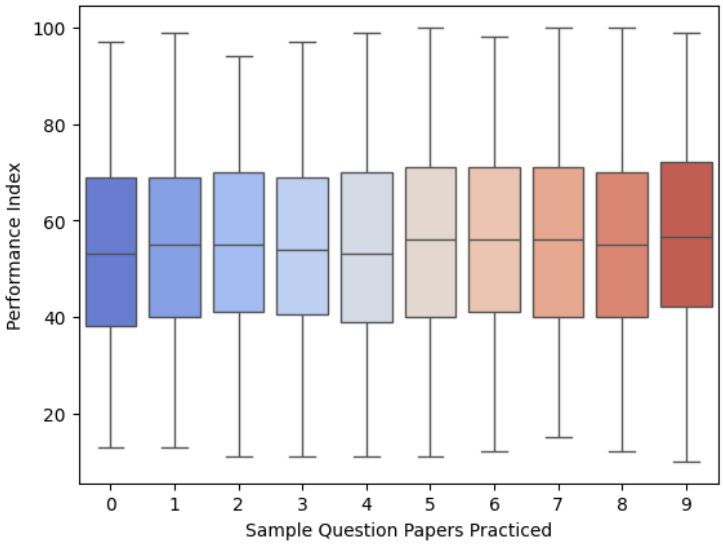
- Từ biểu đồ ta có thể thấy được không có quá nhiều sự chuyển biến giữa việc có hay không có tham gia Extracurricular Activities và chỉ đóng góp rất nhỏ đến Performance Index

- **Feature Sleep Hours**



- Từ đồ thị trên, ta có thể thấy được việc ngủ càng nhiều thì Performance Index cũng sẽ tăng theo

- **Feature Sample Question Papers Practiced**



- Từ đồ thị trên, ta có thể thấy được sự phân bố giữa số lượng Sample Question Papers Practiced khá đồng đều, không có quá nhiều sự khác biệt giữa các số lượng

4.2 Yêu cầu 2a

a. Bước thực hiện:

- **B1**: Đầu tiên, chúng ta cần phải tiền xử lý X_train và X_test bằng cách dùng hàm preprocess cho cả 2 để thêm cột 1 vào ma trận
- **B2:** Huấn luyện mô hình trên tập train bằng hàm fit và sau đó lấy ra hệ số hồi quy (coefficient) và hệ số chặn (intercept) bằng hàm get_params(), trong đó công thức hồi quy:

$$y = intercept + \sum_{i=1}^{5} (coefficients[i] \times feature_i)$$

```
array([-33.96928368,   2.85202007,   1.01786957,   0.60428174,
         0.47356583,   0.19237624])
```

- **B3**: Sử dụng hàm predict để dự đoán mô hình trên X_test và hàm mae để tính độ lỗi trên tập test, và kết quả như sau

```
# Gọi hàm MAE (tự cài đặt hoặc từ thư viện) trên tập kiểm tra
y_hat = model.predict(X_test1a)
mae(y_test, y_hat)
```

✓ 0.0s

```
C:\Users\ACER\AppData\Local\Temp\ipykernel_22392\2223892123.py:22
  return np.mean(np.abs(y.ravel() - y_hat.ravel()))

1.595648688476289
```

b. Kết quả từ mô hình:

- **Student Performance = 2.852* Hours Studied + 1.018* Previous Scores + 0.604* Extracurricular Activities + 0.474* Sleep Hours + 0.192* Sample Question Papers Practiced -33.969**
- **MAE = 1.5956**

c. Nhận xét kết quả:

- **Hiệu suất mô hình:**
  Giá trị MAE là 1.5956 cho biết rằng trung bình, mô hình dự đoán sai lệch khoảng 1.6 đơn vị so với giá trị thực tế trên thang đo của Performance Index. Với một bài toán hồi quy, MAE càng thấp thì mô hình càng tốt. Tuy nhiên, mức độ chấp nhận được của MAE phụ thuộc vào phạm vi giá trị của biến mục tiêu (ở đây là Performance Index).
- **Độ phù hợp của mô hình:**
  Do Performance Index có giá trị dao động từ 10 đến 100 nên một MAE khoảng 1.6 là khá nhỏ và cho thấy mô hình đang dự đoán khá chính xác.
  ⇨ **Với MAE 1.5956, mô hình sử dụng tất cả 5 đặc trưng (Hours Studied, Previous Scores, Extracurricular Activities, Sleep Hours, Sample**

Question Papers Practiced) đang hoạt động tốt rong phạm vi giá trị của Performance Index

4.3 Yêu cầu 2b

a. Bước thực hiện:

- **B1:** Đầu tiên, chúng ta cần shuffle data training 1 lần
- **B2:** Sau đó với mỗi đặc trưng, chúng ta sẽ huấn luyện cho đặc trưng đó k lần bằng K-Fold-Cross-Validation **(Ở đây chúng ta sẽ dùng mặc định k = 5)** trong đó, chúng ta sẽ chia tập train và tập test thông qua indices được lấy ra từ K-Fold-Cross-Validation và preprocess cả 2 tập data
- **B3:** Tính giá trị trung bình MAE của đặc trưng đó
- **B4:** So sánh giá trị trung bình MAE của các đặc trưng nào nhỏ nhất nghĩa là tốt nhất
- **B5:** Huấn luyện lại mô hình trên tập train để lấy ra hệ số tự do và hệ số chặn
- **B6:** Cuối cùng tính MAE trên tập test

b. Kết quả:

```
Feature: Hours Studied
15.301165707629366
15.601946875332242
15.288403179376859
15.71681437628033
15.368196989161511
Feature: Previous Scores
6.523562446609711
6.778770828630133
6.642265057512893
6.6043596471380095
6.555578832470456
Feature: Extracurricular Activities
15.712319974399659
16.25800244094246
16.30112573480831
16.62699737222247
16.079852087772057
Feature: Sleep Hours
15.751256007011005
16.26336065928619
16.26548205195893
16.610507227224936
16.064024533993827
Feature: Sample Question Papers Practiced
15.717876944886985
16.23110962410841
16.278320668700808
16.647716655080888
16.05857860552269
The best feature is: Previous Scores
```

	Feature	MAE
0	Hours Studied	15.455305
1	Previous Scores	6.620907
2	Extracurricular Activities	16.195660
3	Sleep Hours	16.190926
4	Sample Question Papers Practiced	16.186720

- Với k = 5, mô hình tốt nhất có đặc trưng là Previous Scores
- Công thức hồi quy:

```
# Huấn luyện lại mô hình best_feature_model với đặc trưng tốt nhất trên toàn bộ tập huấn luyện

best_feature = min_mae[0]
best_feature_index = label.index(best_feature)
X_best_feature = X_train1[:, [best_feature_index]]

print('Best feature:', X_best_feature)
X_best_feature = preprocess(X_best_feature)

print(X_best_feature)
# Train the model using the best feature
best_feature_model = OLSLinearRegression()
best_feature_model.fit(X_best_feature, y_train1)
best_feature_model.get_params()
```

✓ 0.0s

```
Best feature: [[67]
 [68]
 [95]
 ...
 [77]
 [79]
 [76]]
[[ 1. 67.]
 [ 1. 68.]
 [ 1. 95.]
 ...
 [ 1. 77.]
 [ 1. 79.]
 [ 1. 76.]]

array([-14.98864578,   1.01050301])
```

Student Performance = 1.010* Previous Scores -14.989

- MAE trên tập test = 6.5442772934525015

```
# Gọi hàm MAE (tự cài đặt hoặc từ thư viện) trên tập kiểm tra với mô hình best_feature_model

# B7: Kiểm tra mô hình tốt nhất với D_test

X_test_2b = test.iloc[:, [best_feature_index]].values
X_test_2b = preprocess(X_test_2b)
y_hat_best_feature = best_feature_model.predict(X_test_2b)
mae_best_feature = mae(y_test, y_hat_best_feature)
print('MAE on test set using the best feature:', mae_best_feature)
```

✓ 0.0s

```
MAE on test set using the best feature: 6.5442772934525015
```

c. Nhận xét:

- Sự chênh lệch giữa MAE trên tập huấn luyện (6.620907) và tập kiểm tra (6.544277) là rất nhỏ. Điều này cho thấy mô hình không bị overfitting và có thể tổng quát hóa tốt trên dữ liệu mới.
- Các đặc trưng khác như Hours Studied, Extracurricular Activities, Sleep Hours, và Sample Question Papers Practiced đều có MAE lớn hơn (từ khoảng 15.45 đến 16.19). **Điều này chỉ ra rằng các đặc trưng này không có khả năng dự đoán tốt như Previous Scores. Tuy nhiên cũng có thể thấy được đặc trưng Hours Studied có MAE thấp thứ 2 trong các đặc trưng nên cũng có thể đáng cân nhắc được sử dụng cho câu sau**, gần như tương đồng với sự phân tích dữ liệu từ câu 1

4.4 Yêu cầu 2c

a. Bước thực hiện:

- **B1:** Phân tích dữ liệu và quyết định số lượng đặc trưng sẽ chọn
  - Từ yêu cầu 2b và yêu cầu 1, ta biết được rằng đặc trưng Hours Studied và Previous Scores là hai đặc trưng có ảnh hưởng lớn nhất đến Performance Index (Dựa trên MAE được tính ở yêu câu 2b)
  - ⇨ **Tất cả model sẽ có hai đặc trưng này**
  - Sau đó, chúng ta sẽ tiếp tục thêm các đặc trưng khác dựa trên các phân tích dữ liệu có từ yêu cầu 1
  - **Model1:**
    - **Đặc trưng sử dụng:**
      Hours Studied
      Previous Scores
      Một đặc trưng kết hợp: ((Extracurricular Activities * Sample Question Papers Practiced) / Sleep Hours) ^ (1/2)
    - **Lý do chọn đặc trưng:**
      Kết hợp Extracurricular Activities và Sample Question Papers Practiced được chia bởi Sleep Hours tạo ra một đặc trưng mới nhằm thể hiện mức độ hiệu quả khi kết hợp giữa hoạt động ngoại khóa và luyện tập với số giờ ngủ. Đặc trưng này được chuyển đổi bằng căn bậc hai nhằm giảm sự ảnh hưởng của các giá trị quá lớn và tạo ra một phân phối đều hơn.
      Ngoài ra bởi vì cả 3 đặc trưng này đều có MAE gần như tương đồng nhau và khá lớn nên để giảm thiểu độ sai thì chúng ta sẽ căn bậc 2 đặc trưng kết hợp
    - **Công thức:**

$$\text{Performance Index} = a * \text{Hours Studied} + b * \text{Previous Scores} + c * \sqrt{\left(\frac{\text{Extracurricular Activities} * \text{Sample Question Papers Practiced}}{\text{Sleep Hours}}\right)} + E$$

  - **Model2:**
    - **Đặc trưng sử dụng:**
      Hours Studied
      Previous Scores
    - **Lý do chọn đặc trưng:**
      Đây là hai đặc trưng cơ bản và quan trọng nhất đã được chứng minh là có tác động mạnh nhất đến kết quả học tập thông qua phân tích MAE. Mô hình này đơn giản nhưng tập trung vào các yếu tố cốt lõi nhất.
    - **Công thức:**

$$\text{Performance Index} = a * \text{Hours Studied} + b * \text{Previous Scores} + E$$

- **Model3:**
  - **Đặc trưng sử dụng:**
    Hours Studied
    Previous Scores
    Một đặc trưng kết hợp: (Sample Question Papers Practiced / Sleep Hours)^(1/2)
  - **Lý do chọn đặc trưng:**
    Hours Studied và Previous Scores vẫn là những đặc trưng quan trọng.
    Đặc trưng mới (Sample Question Papers Practiced / Sleep Hours) tạo ra một chỉ số về hiệu quả luyện tập dựa trên số giờ ngủ và số lượng bài mẫu đã luyện. Việc lấy căn bậc hai nhằm giảm sự khác biệt giữa các giá trị và làm cho mô hình trở nên ổn định hơn.
    Ngoài ra ở đây, chúng ta bỏ đi đặc trưng Extracurricular Activities vì đặc trưng này có MAE cao nhất
  - **Công thức:**

$$\text{Performance Index} = a * \text{Hours Studied} + b * \text{Previous Scores} + c * \sqrt{\left(\frac{\text{Sample Question Papers Practiced}}{\text{Sleep Hours}}\right)} + E$$

- **Model4:**
  - **Đặc trưng sử dụng:**
    Hours Studied
    Previous Scores
    Sleep Hours
    Sample Question Papers Practiced
  - **Lý do chọn đặc trưng:**
    Mô hình này bao gồm tất cả các đặc trưng chính, nhưng không tạo ra các đặc trưng kết hợp hay biến đổi phức tạp. Nó cho phép kiểm tra tác động của từng đặc trưng một cách trực tiếp mà không cần kết hợp hoặc biến đổi.
  - **Công thức:**

$$\text{Performance Index} = a * \text{Hours Studied} + b * \text{Previous Scores} + c * \text{Sleep Hours} + d * \text{Sample Question Papers Practiced} + E$$

- **B2:** Sau đó với mỗi model, chúng ta sẽ huấn luyện cho model đó k lần bằng K-Fold-Cross-Validation **(Ở đây chúng ta sẽ dùng mặc định k = 5)** trong đó, chúng ta sẽ chia tập train và tập test thông qua indices được lấy ra từ K-Fold-Cross-Validation và preprocess cả 2 tập data
- **B3:** Tính giá trị trung bình MAE của model đó
- **B4:** So sánh giá trị trung bình MAE của các model nào nhỏ nhất nghĩa là tốt nhất
- **B5:** Huấn luyện lại model tốt nhất trên tập train để lấy ra hệ số tự do và hệ số chặn
- **B6:** Cuối cùng tính MAE trên tập test

b. Kết quả:

- **Model1:**

```
Model 1
1.8194425438672048
1.786907243179609
1.8152939054317225
1.7387071787943182
1.813029860809761
Mean Absolute Error: 1.794676146416523
```

- **Model2:**

```
Model 2
1.8412266845779208
1.8021999905026878
1.8369624986360524
1.7694435063211151
1.833621497041334
Mean Absolute Error: 1.816690835415822
```

- **Model3:**

```
Model 3
1.8261369652082888
1.7994222565075169
1.817370851758808
1.7547850599977248
1.8271853498581534
Mean Absolute Error: 1.8049800966660985
```

- **Model4:**

```
Model 4
1.6365022274856937
1.6347056508715474
1.651692171203499
1.6290014518904963
1.6519778289436167
Mean Absolute Error: 1.6407758660789706
```

STT	Mô hình		MAE
1	Sử dụng 5 đặc trưng (Hours Studied, Previous Scores, Sleep Hours, Extracurricular Activities, Sample Question Papers Practiced)		1.794676
2	Sử dụng 2 đặc trưng (Hours Studied, Previous Scores)		1.816691
3	Sử dụng 4 đặc trưng(Hours Studied, Previous Scores, Sleep Hours, Sample Question Papers Practiced)	Cộng thông thường	1.641944
		Có sử dụng đặc trưng hỗn hợp từ 3 đặc trưng	1.804980

- Với k = 5, thì mô hình 4 là tối ưu nhất
- Công thức hồi quy:

```python
# Huấn luyện lại mô hình my_best_model trên toàn bộ tập huấn luyện
if (best_model_index == 'Model 1'):
    X_train_2c, y_train_2c = model1(train_shuffle)
elif (best_model_index == 'Model 2'):
    X_train_2c, y_train_2c = model2(train_shuffle)
elif (best_model_index == 'Model 3'):
    X_train_2c, y_train_2c = model3(train_shuffle)
else:
    X_train_2c, y_train_2c = model4(train_shuffle)
# Gọi hàm MAE (tự cài đặt hoặc từ thư viện) trên tập kiểm tra với mô hình best_feature_model
X_test_2c, y_test_2c = model1(test) if best_model_index == 'Model 1' else model2(test) if best_model_index == 'Model 2' else model3(test) if best_model_index == 'Model 3' else
model4(test)

X_train_2c = preprocess(X_train_2c)
X_test_2c = preprocess(X_test_2c)


my_best_model = OLSLinearRegression()
my_best_model.fit(X_train_2c, y_train_2c)
params = my_best_model.get_params()
print(f'Model parameters: {params}')
```

✓ 0.0s Python

```
Model parameters: [-33.66490795   2.85254922   1.01803696   0.46985264   0.19323805]
```

Student Performance = 2.852* Hours Studied + 1.018* Previous Scores + 0.470* Sleep Hours + 0.193* Sample Question Papers Practiced -33.665

- MAE trên tập test = 1.6136875904819854

```
# Gọi hàm MAE (tự cài đặt hoặc từ thư viện) trên tập kiểm tra với mô hình my_best_model

y_hat_best_model = my_best_model.predict(X_test_2c)
mae_best_model = mae(y_test_2c, y_hat_best_model)
print(f'MAE on test set using the best model: {mae_best_model}')
```

✓ 0.0s

```
MAE on test set using the best model: 1.6136875904819854
```

c. Nhận xét

- **Phân tích kết quả:**
  - **Model 1:**

MAE trên tập huấn luyện: 1.794676

Đây là một mô hình có hiệu suất khá tốt, tuy nhiên không phải là tốt nhất trong số các mô hình được thử nghiệm.

  - **Model 2:**

MAE trên tập huấn luyện: 1.816691

Mô hình này chỉ sử dụng hai đặc trưng đơn giản (Hours Studied và Previous Scores), và cho thấy kết quả MAE gần với Model 1, chứng tỏ rằng hai đặc trưng này đã nắm bắt được phần lớn thông tin cần thiết cho việc dự đoán.

  - **Model 3:**

MAE trên tập huấn luyện: 1.804980

Mô hình này sử dụng một biến chuyển đổi mới, nhưng không cải thiện nhiều so với Model 1 và 2.

  - **Model 4:**

MAE trên tập huấn luyện: 1.641944

Đây là mô hình có hiệu suất tốt nhất cả trên tập huấn luyện và tập kiểm tra. mà không tạo ra các biến chuyển đổi phức tạp.

- **Nhận xét tổng quát:**
  - **Hiệu suất của Model 4:**
    Model 4 có MAE thấp nhất trên cả tập huấn luyện và tập kiểm tra, cho thấy rằng mô hình này tổng quát tốt hơn các mô hình khác. Nó có khả năng dự đoán ổn định và ít bị overfitting, đặc biệt là khi sử dụng tất cả các đặc trưng chính.
  - **Sự ổn định giữa tập huấn luyện và tập kiểm tra:**
    Sự chênh lệch giữa MAE trên tập huấn luyện (1.641944) và tập kiểm tra (1.6136875904819854) của Model 4 là rất nhỏ, cho thấy mô hình này không bị overfitting và có khả năng dự đoán chính xác trên dữ liệu mới.
  - **Vai trò của các đặc trưng:**
    Việc sử dụng cả 4 đặc trưng (Hours Studied, Previous Scores, Sleep Hours, Sample Question Papers Practiced) mà không tạo ra các biến chuyển đổi phức tạp đã giúp Model 4 có được hiệu suất tốt nhất. Điều này

cho thấy rằng các đặc trưng gốc đã chứa đủ thông tin quan trọng để dự đoán chính xác mà không cần phải biến đổi thêm.

- **Kết luận:**

  Model 4 là lựa chọn tốt nhất dựa trên cả hiệu suất huấn luyện và kiểm tra. Nó tận dụng tối đa thông tin từ các đặc trưng gốc và đạt được kết quả MAE thấp nhất, cho thấy khả năng dự đoán chính xác và ổn định.

5. Tài liệu tham khảo

- **Linear Regression**

  https://www.ncl.ac.uk/webtemplate/ask-assets/external/maths-resources/statistics/regression-and-correlation/simple-linear-regression.html#:~:text=7%20See%20Also-,Definition,this%20is%20known%20as%20interpolation.

  https://www.geeksforgeeks.org/simple-linear-regression-using-r/

  https://www.geeksforgeeks.org/ml-multiple-linear-regression-using-python/

- **K-Fold-Cross-Validation**

  https://www.kaggle.com/code/satishgunjal/tutorial-k-fold-cross-validation