ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Linear Regression

Môn học: Toán Ứng Dụng Và Thống Kê Cho Công Nghệ Thông Tin

MTH00057-23CLC08

Sinh viên thực hiện:

Nguyễn Phương Thảo

23127306

23CLC08

Giảng viên:

Vũ Quốc Hoàng

Trần Thị Thảo Nhi

Nguyễn Văn Quang Huy

Ngày 15 tháng 8 năm 2025

Mục lục

1 Ý tưởng thực hiện

1.1 Tổng quan về đồ án

Đồ án tập trung vào việc xây dựng mô hình hồi quy tuyến tính nhằm phân tích và dự đoán chỉ số thành tích học tập của sinh viên (Academic Student Performance Index). Mục tiêu là xác định các yếu tố ảnh hưởng và mức độ tác động của từng yếu tố tới kết quả học tập, từ đó xây dựng mô hình dự đoán Performance Index.

1.2 Input / Output

Input

Bộ dữ liệu Student Performance gồm 10.000 dòng dữ liệu (được tỉ lệ 9:1 cho train và test) và 6 cột:

1. Hours Studied – Tổng số giờ học của sinh viên.
2. Previous Scores – Điểm số sinh viên đạt được trong các bài kiểm tra trước đó.
3. Extracurricular Activities – Sinh viên có tham gia hoạt động ngoại khóa hay không.
4. Sleep Hours – Số giờ ngủ trung bình mỗi ngày của sinh viên.
5. Sample Question Papers Practiced – Số lượng đề thi mẫu đã luyện tập.
6. Performance Index – Chỉ số thành tích học tập (biến mục tiêu).

Output

- Phân tích dữ liệu từ đó nhận xét tầm quan trọng và ảnh hưởng của từng đặc trưng.
- Mô hình dự đoán Performance Index với sai số nhỏ nhất có thể.
- Bảng đánh giá gồm:
  - Kết quả Cross-validation để tránh overfitting.
  - MAE (Mean Absolute Error) trên tập kiểm tra.

1.3 Mục tiêu

Mục tiêu chính của đồ án là xây dựng và lựa chọn một mô hình dự đoán tối ưu cho biến mục tiêu Performance Index dựa trên các yếu tố đầu vào của sinh viên. Ngoài ra, đồ án yêu cầu tạo thêm đặc trưng mới (feature engineering) nhằm nâng cao khả năng dự đoán. Các mục tiêu cụ thể gồm:

- Xây dựng mô hình hồi quy tuyến tính và các phiên bản mở rộng.
- Thực hiện so sánh hiệu quả giữa các mô hình thông qua Cross-validation và đánh giá trên tập kiểm tra (test set).
- Lựa chọn mô hình có độ chính xác cao nhất (MAE thấp nhất).

1.4 Ý tưởng giải quyết

1.4.1 Hồi quy tuyến tính với phương pháp OLS

Trong bài toán hồi quy tuyến tính, mục tiêu là tìm vector trọng số $\mathbf{w}$ sao cho mô hình:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$$

dự đoán giá trị $\hat{\mathbf{y}}$ gần nhất với giá trị thực tế $\mathbf{y}$.
Phương pháp **Bình phương tối thiểu** (Ordinary Least Squares - OLS) [2] giải quyết bài toán này bằng cách tối thiểu hoá tổng bình phương sai số:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

hay viết gọn dưới dạng ma trận:

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Giải bài toán trên bằng cách lấy đạo hàm theo $\mathbf{w}$ và đặt bằng 0, ta thu được **công thức nghiệm đóng**:

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

Quy trình áp dụng:

1. **Tiền xử lý dữ liệu**: Chuẩn hoá dữ liệu và thêm một cột toàn giá trị 1 vào ma trận $\mathbf{X}$ để mô hình học hệ số chặn (intercept).
2. **Huấn luyện mô hình**: Sử dụng công thức OLS để tính toán vector trọng số $\mathbf{w}$ từ tập huấn luyện.
3. **Dự đoán**: Với dữ liệu kiểm tra $\mathbf{X}_{\text{text}}$, dự đoán đầu ra được tính:

$$\hat{\mathbf{y}}_{\text{pred}} = \mathbf{X}_{\text{test}}\mathbf{w}$$

4. **Đánh giá**: Sử dụng chỉ số MAE kết hợp với Cross-Validation để đánh giá hiệu suất mô hình.

1.4.2 Tạo đặc trưng mới

Để cải thiện khả năng dự đoán, các đặc trưng mới được tạo ra từ dữ liệu ban đầu: [3]

- **Biến bậc hai (Polynomial)**: Bình phương các biến hiện tại để nắm bắt quan hệ phi tuyến.
- **Biến tương tác (Interaction)**: Kết hợp hai hoặc nhiều đặc trưng để phản ánh tác động phối hợp.
- **Biến tùy chỉnh (Custom)**: tạo ra các chỉ số phản ánh hiệu quả hoặc quan hệ giữa các biến, được tính toán dựa trên các cột hiện có trong dữ liệu để tăng khả năng dự đoán của mô hình.

1.4.3 Đánh giá mô hình với K-Fold Cross-Validation

Để so sánh hiệu suất giữa các tập đặc trưng, phương pháp **K-Fold Cross-Validation** [4][5] được sử dụng. Quy trình như sau:

1. Chia dữ liệu thành k phần (folds) có kích thước gần bằng nhau.
2. Với mỗi lần lặp, chọn 1 fold làm tập kiểm tra (validation), các fold còn lại làm tập huấn luyện.
3. Huấn luyện mô hình trên tập huấn luyện và đánh giá trên tập kiểm tra, lưu lại chỉ số MAE.
4. Sau k lần lặp, tính MAE trung bình của tất cả các lần để đánh giá hiệu suất mô hình.

2 Chi tiết thực hiện

Các thư viện chính sử dụng gồm:

- **Pandas**: đọc dữ liệu từ file CSV (`pd.read_csv`), xử lý dữ liệu dạng bảng (`DataFrame`), chọn cột đặc trưng và biến mục tiêu, tạo đặc trưng mới, và thực hiện các thao tác tiền xử lý dữ liệu.
- **NumPy**: thao tác với mảng số liệu (`array`), thực hiện các phép toán đại số tuyến tính (như nhân ma trận, nghịch đảo ma trận), tạo chỉ số cho k-fold cross-validation, và xử lý dữ liệu đầu vào/ra cho các hàm huấn luyện và dự đoán.
- **Matplotlib**: vẽ biểu đồ trực quan hóa dữ liệu như histogram, scatter plot, boxplot, heatmap; sử dụng các hàm như `plt.figure`, `plt.show`, và thiết lập kích thước, tiêu đề, nhãn trục.
- **Seaborn**: trực quan hóa dữ liệu nâng cao, vẽ histogram có đường KDE, scatter plot, boxplot, countplot, và heatmap cho ma trận tương quan; giúp biểu đồ đẹp hơn và dễ đọc hơn so với Matplotlib.

2.1 Các hàm phân tích khám phá dữ liệu

Các hàm chính:

1. **show_dataset_info(df)**: In ra cấu trúc dataset: số dòng, số cột, tên cột, kiểu dữ liệu và số lượng giá trị không bị thiếu. In bảng thống kê mô tả các biến số, bao gồm: giá trị trung bình, độ lệch chuẩn, giá trị nhỏ nhất, giá trị lớn nhất và các phần tư (25%, 50%, 75%).
2. **analyze_target(df, target_col)**: Phân tích phân phối của biến mục tiêu bằng biểu đồ histogram kết hợp đường mật độ (KDE).
3. **analyze_features(df, target_col)**: Phân tích mối quan hệ giữa từng đặc trưng và biến mục tiêu.
    - Biến phân loại nhị phân (1/0): Vẽ boxplot so sánh giá trị biến mục tiêu giữa các nhóm. Vẽ countplot để hiển thị tần suất xuất hiện của các nhóm.

- Biến liên tục: Vẽ scatter plot giữa đặc trưng và biến mục tiêu. Vẽ histogram + KDE để mô tả phân phối giá trị của đặc trưng.

4. **show_correlation_matrix(df)**: Tính toán ma trận hệ số tương quan (Pearson correlation) giữa các đặc trưng và Vẽ heatmap bằng Seaborn.

5. **rank_features_by_importance(df, target_col)**: Tính toán hệ số tương quan tuyệt đối giữa từng đặc trưng và biến mục tiêu, từ đó xếp hạng tầm quan trọng của các đặc trưng.

2.2 Các hàm phục vụ xây dựng mô hình hồi quy tuyến tính

2.2.1 a. Mô hình sử dụng toàn bộ 5 đặc trưng [1]

`preprocess(x)`

Chức năng: Tiền xử lý dữ liệu trước khi huấn luyện mô hình. Thêm một cột toàn giá trị 1 (đại diện cho intercept) vào dữ liệu đầu vào.

Tham số: `x` (`numpy.ndarray`): Ma trận dữ liệu đặc trưng gốc.

Trả về: `X` (`numpy.ndarray`): Ma trận đặc trưng đã thêm cột intercept.

Ý nghĩa: Chuẩn bị dữ liệu ở dạng phù hợp cho mô hình hồi quy tuyến tính.

`OLSLinearRegression`

Chức năng: Triển khai mô hình hồi quy tuyến tính sử dụng phương pháp bình phương tối thiểu (Ordinary Least Squares).

Các phương thức:

1. `fit(X, y)`

   Mục đích: Mô hình ước lượng vector trọng số w bằng cách giải nghiệm đóng:

   $$w = (X^T X)^{-1} X^T y$$

   Tham số:

   - `X` (`numpy.ndarray`): Ma trận đặc trưng có kích thước (n, m), với n là số mẫu, m là số đặc trưng (bao gồm intercept nếu có).
   - `y` (`numpy.ndarray`): Vector đầu ra thực tế kích thước $(n,)$ hoặc $(n, 1)$.

Kết quả: Lưu vector trọng số w vào thuộc tính `self.w`.

Trả về: `self`

2. `get_params()`

   Mục đích: Lấy vector trọng số w đã được huấn luyện.

   Trả về: `numpy.ndarray`: Vector trọng số kích thước $(m,)$.

   Ý nghĩa: Cho phép truy xuất trực tiếp hệ số chặn và các hệ số của đặc trưng.

3. `predict(X)`

   Mục đích: Tính toán giá trị dự đoán $\hat{y}$ từ dữ liệu đầu vào.

   Tham số: `X` (`numpy.ndarray`): Ma trận đặc trưng kích thước (n, m) (đã được tiền xử lý với cột intercept nếu cần).

   Trả về: `numpy.ndarray`: Vector dự đoán kích thước $(n,)$ hoặc $(n, 1)$.

   Công thức:

$$\hat{y} = Xw$$

`train_model(X, y, preprocess_fn=preprocess, model_cls=OLSLinearRegression)`

Chức năng: Huấn luyện mô hình hồi quy tuyến tính trên tập dữ liệu đầu vào, kèm bước tiền xử lý đặc trưng.

Tham số:

- `X` (`numpy.ndarray`): Ma trận đặc trưng đầu vào kích thước $(n\_samples, n\_features)$.
- `y` (`numpy.ndarray`): Vector mục tiêu kích thước $(n\_samples,)$.
- `preprocess_fn` (`function`, mặc định: `preprocess`): Hàm tiền xử lý dữ liệu, ví dụ thêm cột bias hoặc chuẩn hóa.
- `model_cls` (`class`, mặc định: `OLSLinearRegression`): Lớp mô hình cần huấn luyện, phải có phương thức `fit(X, y)`.

Trả về:

- `model`: Đối tượng mô hình đã được huấn luyện, có thể dùng `predict()` để dự đoán.

Mô tả:

1. Tiền xử lý dữ liệu bằng hàm `preprocess_fn`.

2. Khởi tạo mô hình từ lớp `model_cls`.
3. Gọi `fit()` để huấn luyện mô hình.
4. Trả về mô hình đã được huấn luyện.

`calc_mae(y, y_hat)`

Chức năng: Tính toán sai số tuyệt đối trung bình MAE.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Tham số:

- `y` (`numpy.ndarray`): Giá trị thực tế.
- `y_hat` (`numpy.ndarray`): Giá trị dự đoán.

Trả về: `float` – Giá trị MAE.

`print_regression_equation(model, feature_names, target_name="y", decimals=3)`

Chức năng: In phương trình hồi quy tuyến tính từ mô hình đã huấn luyện.

Tham số:

- `model`: Đối tượng mô hình đã huấn luyện, có thuộc tính `w`.
- `feature_names` (`list`): Danh sách tên đặc trưng.
- `target_name` (`str`): Tên biến mục tiêu (mặc định "y").
- `decimals` (`int`): Số chữ số thập phân làm tròn.

2.2.2 Thực hiện yêu cầu 2a

1. **Lựa chọn đặc trưng:** Sử dụng toàn bộ 5 biến đầu vào ban đầu.
2. **Huấn luyện mô hình:** Gọi hàm `train_model` với các tham số:
   - `X_train`: dữ liệu huấn luyện.
   - `y_train`: biến mục tiêu.

- `preprocess_fn=preprocess`: tiền xử lý dữ liệu (thêm cột intercept).
- `model_cls=OLSLinearRegression`: lớp mô hình hồi quy tuyến tính.

Hàm này trả về mô hình đã huấn luyện (`model_2a`).

3. **In phương trình hồi quy:** Sử dụng hàm `print_regression_equation` để hiển thị phương trình hồi quy với các hệ số tương ứng cho từng đặc trưng.
4. **Dự đoán trên tập kiểm tra:**
   - Tiền xử lý tập kiểm tra bằng hàm `preprocess(X_test)`.
   - Dự đoán giá trị $\hat{y}$ bằng phương thức `model_2a.predict()`.
   - Lưu kết quả vào `y_pred_2a`.
5. **Đánh giá mô hình:** Tính MAE giữa giá trị thực tế và giá trị dự đoán bằng hàm `calc_mae(y_test, y_pred_2a)`.

2.2.3 b. Mô hình sử dụng duy nhất 1 đặc trưng

1. `kfold_split(n, k=5, random_state=42)` [4]

Chức năng: Chia dữ liệu thành k tập train/validation phục vụ Cross-Validation.

Tham số:

- `n` (`int`): Số lượng mẫu.
- `k` (`int`): Số lượng folds.
- `random_state` (`int`): Seed để tái lập kết quả.

Trả về: Danh sách `[(train_idx, val_idx), ...]` chứa chỉ số train/val cho từng fold. **Mô tả:**

1. Sinh ngẫu nhiên chỉ số mẫu với seed `random_state` để tái lập kết quả.
2. Tính kích thước mỗi fold sao cho các fold có số mẫu gần bằng nhau.
3. Chia dữ liệu thành k fold.
4. Với mỗi fold, tính index sao cho:

- Tập *validation* là fold hiện tại.
- Tập *train* là toàn bộ các fold còn lại ghép lại.

2. `cross_validate_models(X, y, models_dict, folds=None)` [4]

Chức năng: Thực hiện **k-fold cross-validation** cho nhiều mô hình và trả về lỗi MAE trung bình của từng mô hình.

Tham số:

- `X` (`pandas.DataFrame`): Tập dữ liệu đặc trưng gốc (chỉ dùng để lấy số mẫu).
- `y` (`pandas.Series` hoặc `numpy.ndarray`): Biến mục tiêu.
- `models_dict` (`dict`): Từ điển dạng `{tên_mô_hình: DataFrame_đặc_trưng}`.
- `folds` (`list[tuple]`): Danh sách các cặp (`train_idx`, `val_idx`) từ hàm `kfold_split`.

Trả về: `dict` dạng `{tên_mô_hình: MAE_trung_bình}`.

Cách hoạt động:

1. Duyệt qua từng mô hình trong `models_dict`.
2. Với mỗi fold trong `folds`:
   - Tách dữ liệu *train* và *validation* theo chỉ số đã cho.
   - Tiền xử lý dữ liệu bằng hàm `preprocess`
   - Huấn luyện mô hình hồi quy OLS trên tập train.
   - Dự đoán trên tập validation và tính lỗi MAE.
3. Lấy trung bình lỗi MAE qua tất cả các folds.

3. `format_cv_results(results_dict)`

Chức năng: Chuyển kết quả cross-validation thành bảng `DataFrame` dễ đọc.

Tham số: `results_dict` (`dict`): {`tên mô hình`: MAE}. **Trả về:** `pandas.DataFrame` – Bảng kết quả sắp xếp theo MAE tăng dần.

2.2.4 Thực hiện yêu cầu 2b

1. **Xác định đặc trưng đơn tốt nhất:**

- Tạo các mô hình hồi quy tuyến tính chỉ với từng đặc trưng đơn lẻ từ danh sách `features`.
- Tách dữ liệu huấn luyện thành 5 folds bằng hàm `kfold_split`.
- Thực hiện cross-validation cho từng mô hình bằng hàm `cross_validate_models` và lưu kết quả MAE trung bình vào `results_2b`.
- Chọn đặc trưng có MAE thấp nhất làm đặc trưng đơn tốt nhất (`best_feature`).

2. **Huấn luyện mô hình với đặc trưng tốt nhất:**
   - Lấy ma trận đặc trưng huấn luyện chỉ gồm `best_feature`: `X_2b = X_train[[best_feature]]`.
   - Gọi hàm `train_model` để huấn luyện mô hình hồi quy tuyến tính.
   - Lưu mô hình đã huấn luyện vào `best_feature_model`.
   - In phương trình hồi quy với hàm `print_regression_equation`.
3. **Dự đoán và đánh giá:**
   - Tiền xử lý tập kiểm tra với đặc trưng tốt nhất bằng `preprocess(X_test[[best_feature]])`.
   - Dự đoán giá trị $\hat{y}$ bằng `best_feature_model.predict(X_test_proc)`.
   - Tính MAE trên tập kiểm tra bằng hàm `calc_mae(y_test, y_pred_2b)` để đánh giá độ chính xác của mô hình đơn.

2.2.5 c. Mô hình tự thiết kế

`add_features(df)`

Chức năng: Mở rộng tập dữ liệu ban đầu bằng cách tạo ra các đặc trưng mới nhằm giúp mô hình học tốt hơn. **Tham số:** `df` (`pandas.DataFrame`): Bảng dữ liệu gốc.
Trả về: `pandas.DataFrame` mới chứa cả đặc trưng gốc và đặc trưng được tạo thêm.

2.2.6 Thực hiện yêu cầu 2c:

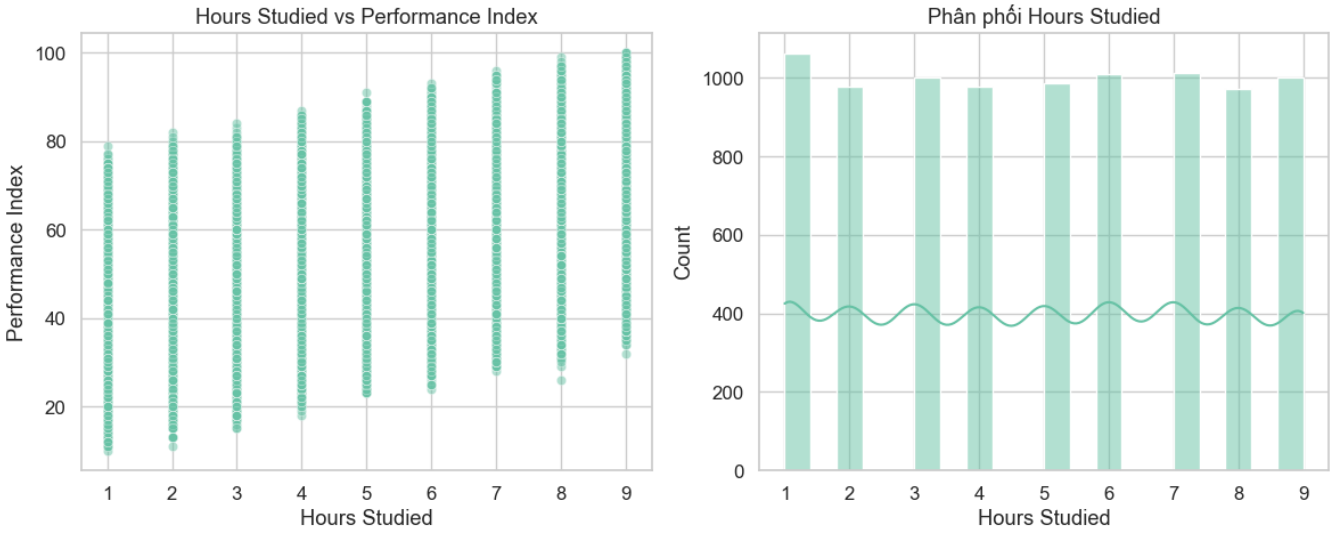
1. **Tạo các đặc trưng mở rộng:** Áp dụng hàm `add_features(df)` để sinh thêm các đặc trưng polynomial, interaction và custom feature từ tập huấn luyện và kiểm tra.
2. **Định nghĩa các mô hình tự thiết kế**
3. Các bước tiếp theo thực hiện tương tự yêu cầu 2b

3 Kết quả và kết luận

3.1 Phân tích khám phá dữ liệu

3.1.1 Nhận xét từng đặc trưng

a. Hours Studied



Hình 1: Biểu đồ tương quan giữa Hours Studied và Performance Index

Thống kê mô tả:

- Trung bình 5 giờ/ngày, với độ lệch chuẩn 2.59 giờ.
- 25% số học sinh học <= 3 giờ, 50% học 5 giờ và 75% học >= 7 giờ.
- Giờ học dao động từ 1 giờ đến 9 giờ.

Biểu đồ scatter: Xu hướng tăng rõ rệt – học càng nhiều giờ thì Performance Index càng cao.
Biểu đồ phân phối: Giờ học phân bố đều, không lệch quá nhiều.

Nhận xét: Số giờ học càng cao thì Performance Index càng lớn. Tuy nhiên, dữ liệu có độ phân tán, không phải tất cả các học sinh học nhiều giờ đều đạt điểm cao, nhưng vẫn tồn tại mối quan hệ tích cực rõ ràng.

b. Previous Scores

Hình 2: Biểu đồ tương quan giữa Previous Scores và Performance Index

Thống kê mô tả:

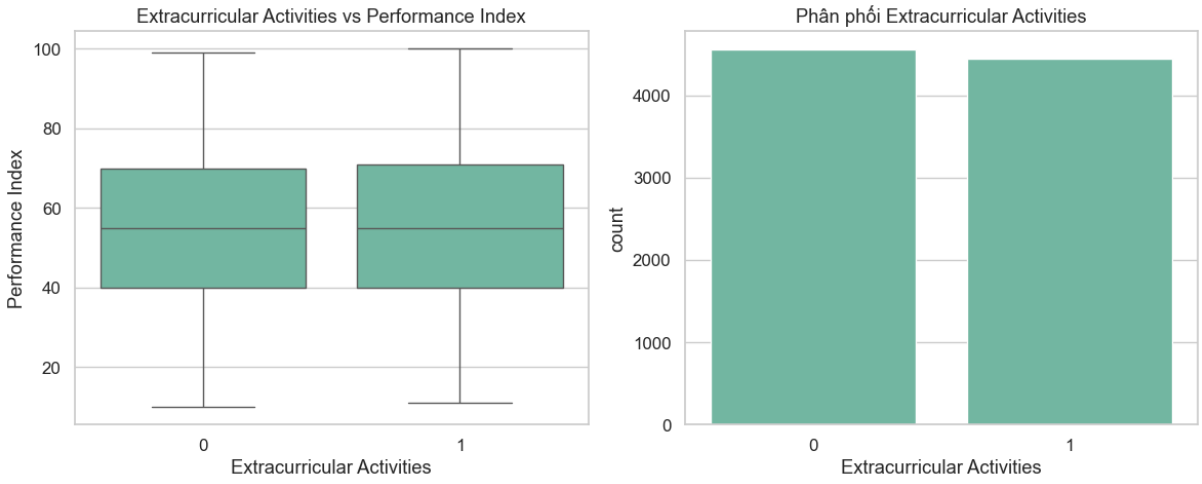
- Trung bình 69, độ lệch chuẩn 17.37.
- 25% học sinh có điểm <= 54, 75% có điểm >= 85.
- Phân bố khá rộng từ 40 đến 99.

Biểu đồ scatter: Mối quan hệ tuyến tính gần như hoàn hảo với Performance Index: điểm số trước đây cao thường dẫn đến Performance Index cao.

Biểu đồ phân phối: Previous Scores tập trung ở 40–100.

Nhận xét: Đây là đặc trưng mạnh nhất, có khả năng quyết định lớn đến kết quả mô hình.

c. Extracurricular Activities



Hình 3: Biểu đồ tương quan giữa Extracurricular Activities và Performance Index

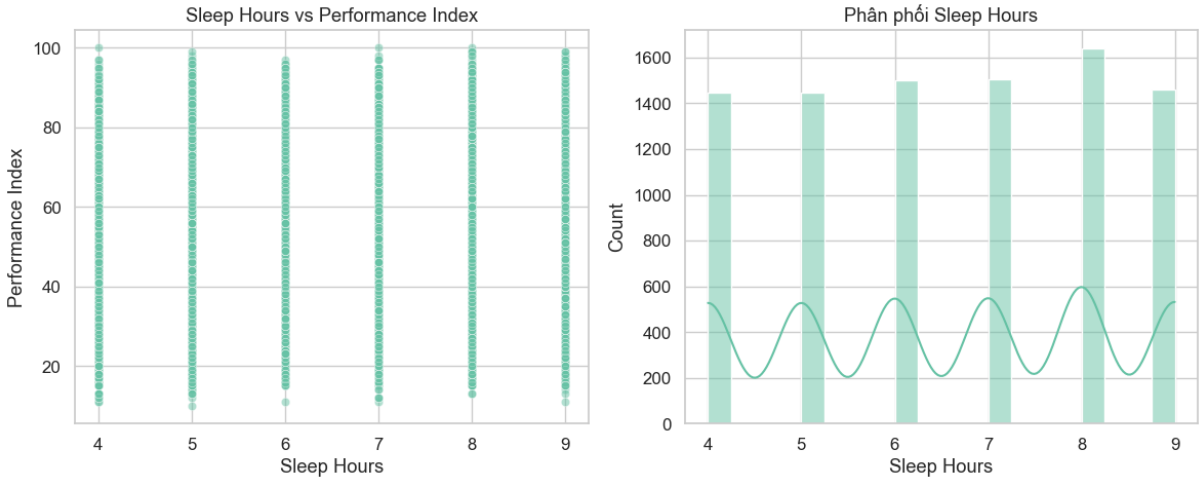
Thống kê mô tả: Khoảng 49.36% học sinh tham gia hoạt động ngoại khóa.

Biểu đồ boxplot: Trung vị Performance Index của hai nhóm gần như giống nhau, phân bố chồng lấn.

Biểu đồ phân phối: Số lượng học sinh tham gia và không tham gia khá cân bằng.

Nhận xét: Tác động yếu hoặc không rõ rệt, có thể chỉ có ý nghĩa nếu kết hợp với biến khác.

d. Sleep Hours

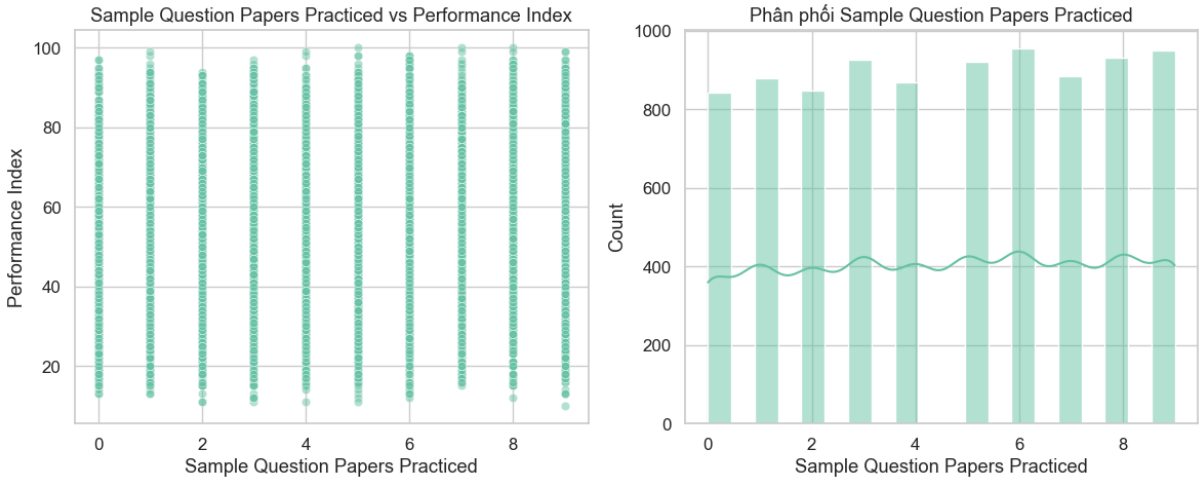


Hình 4: Biểu đồ tương quan giữa Sleep Hours và Performance Index

Thống kê mô tả: Trung bình 6.53 giờ, dao động từ 4 đến 9 giờ, độ lệch chuẩn 1.70 giờ.
Biểu đồ scatter: Không có xu hướng rõ ràng; Performance Index trải đều ở mọi giờ ngủ từ 4–9h.
Biểu đồ phân phối: Số giờ ngủ phân bố khá đều, lệch nhẹ về 8 giờ.

Nhận xét: Các mức giờ ngủ khác nhau có phân bố điểm gần giống nhau => có thể tác động của biến này là nhỏ hoặc phi tuyến tính.

e. Sample Question Papers Practiced



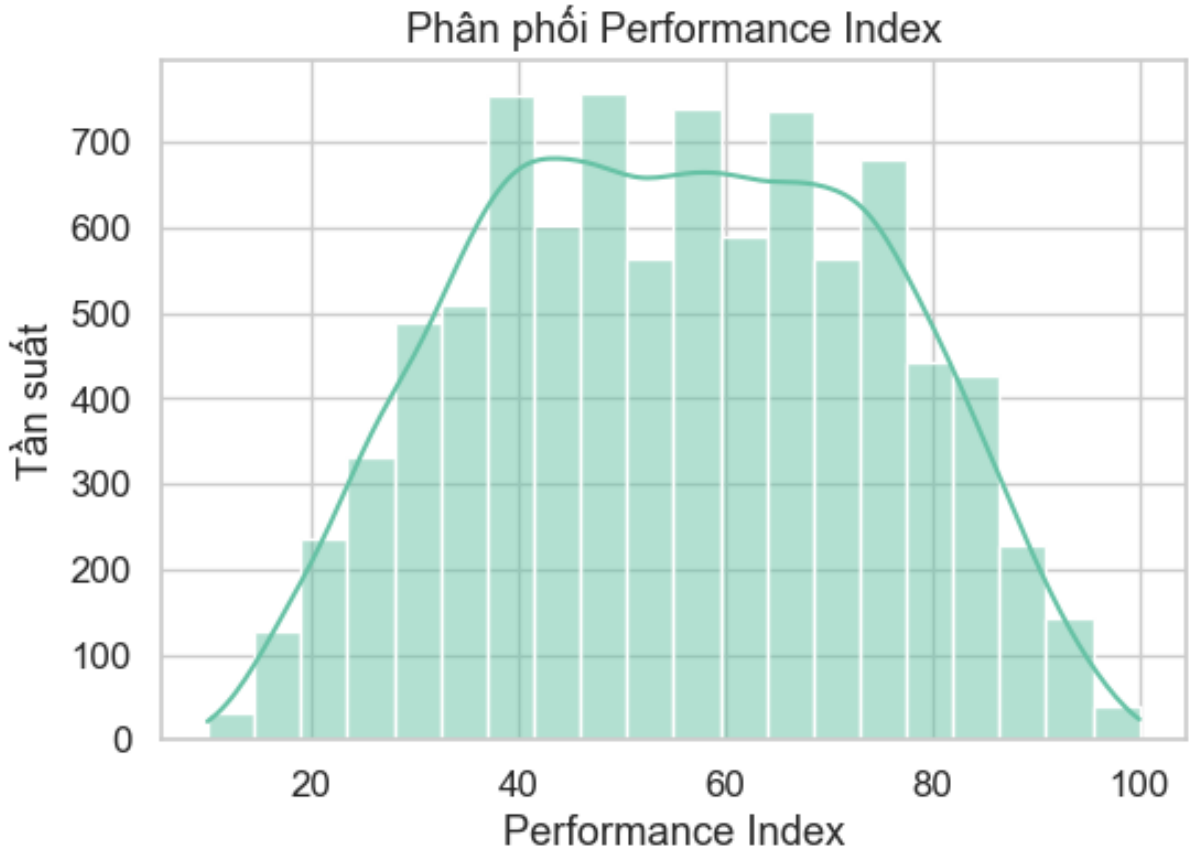
Hình 5: Biểu đồ tương quan giữa Sample Question Papers Practiced và Performance Index

Thống kê mô tả:

- Trung bình 4.59 đề, độ lệch chuẩn 2.86.
- Phân bố từ 0 đến 9 đề.

Biểu đồ scatter: Có xu hướng tăng nhẹ Performance Index khi luyện nhiều đề, nhưng ảnh hưởng yếu hơn Hours Studied và Previous Scores.
Biểu đồ phân phối: Học sinh thực hành số lượng đề khá đều nhau, không tập trung vào số lượng nào cụ thể.

f. Performance Index



Thống kê mô tả:

- Trung bình 55, độ lệch chuẩn 19.19.
- Phân bố trải từ 10 đến 100, khá rộng → mô hình hồi quy có khả năng dự đoán tốt nếu chọn biến phù hợp.

3.1.2 Nhận xét chung và đánh giá mức độ ảnh hưởng

Đặc trưng	**Hệ số tương quan**
Previous Scores	0.914775
Hours Studied	0.369148
Sleep Hours	0.043980
Sample Question Papers Practiced	0.041088
Extracurricular Activities	0.025637

Bảng 1: Bảng xếp hạng các đặc trưng số ảnh hưởng mạnh nhất tới Performance Index

Hình 6: Ma trận tương quan giữa các biến trong tập dữ liệu

- **Previous Scores** có mối tương quan rất mạnh với **Performance Index** ($r \approx 0.915$), được thể hiện rõ trên heatmap với màu đỏ đậm. Điều này cho thấy điểm số trước đây là đặc trưng quan trọng nhất quan trọng nhất cho hiệu suất hiện tại.

- **Hours Studied** có tương quan mức trung bình với **Performance Index** ($r \approx 0.369$), nghĩa là việc tăng số giờ học thường có xu hướng cải thiện kết quả, nhưng tác động không mạnh bằng Previous Scores.

- Các biến **Sleep Hours** ($r \approx 0.044$), **Sample Question Papers Practiced** ($r \approx 0.041$) và **Extracurricular Activities** ($r \approx 0.026$) có tương quan rất thấp với **Performance Index**. Điều này cho thấy các yếu tố này có thể không ảnh hưởng trực tiếp hoặc tuyến tính đến hiệu suất học tập, nhưng vẫn có thể có mối quan hệ phi tuyến hoặc tương tác giữa các biến.

- Ma trận tương quan cũng cho thấy giữa các biến độc lập hầu như không có hiện tượng đa cộng tuyến mạnh (ngoại trừ mối quan hệ rất mạnh giữa **Previous Scores** và **Performance Index** vì đây là biến mục tiêu).

Nhận xét: Từ các kết quả trên, có thể thấy mô hình dự báo cần ưu tiên các biến **Previous Scores** và **Hours Studied** để đạt hiệu quả cao, đồng thời xem xét các đặc trưng tương tác hoặc phi tuyến cho các biến có tương quan thấp nhằm khai thác tối đa thông tin.

3.2 Các mô hình hồi quy tuyến tính

3.2.1 a. Mô hình sử dụng toàn bộ 5 đặc trưng

Phương trình hồi quy

$$Performance\ Index = -33.969 + 2.852\,x_1 + 1.018\,x_2 + 0.604\,x_3 + 0.474\,x_4 + 0.192\,x_5$$

Trong đó:

- x_1: Hours Studied
- x_2: Previous Scores
- x_3: Extracurricular Activities
- x_4: Sleep Hours
- x_5: Sample Question Papers Practiced

Giá trị MAE trên tập test:

$$\text{MAE}_{\text{test}} = 1.59565$$

3.2.2 b. Mô hình sử dụng duy nhất 1 đặc trưng

Kết quả chạy cross-validation

Feature	**MAE**
Previous Scores	6.618215
Hours Studied	15.448599
Sleep Hours	16.187007
Sample Question Papers Practiced	16.188386
Extracurricular Activities	16.195873

Bảng 2: MAE của từng mô hình đặc trưng đơn lẻ

Từ bảng 3.2.2 có thể thấy rằng **Previous Scores** là đặc trưng có hiệu suất tốt nhất với giá trị MAE thấp nhất (MAE = 6.618).

Phương trình hồi quy

$$Performance\ Index = -14.989 + 1.011 * Previous\ Scores$$

Giá trị MAE trên tập test

$$\text{MAE}_{\text{test}} = 6.544277$$

Kết luận:

- **Previous Scores** là đặc trưng dự báo tốt nhất cho *Student Performance*, với MAE thấp nhất (6.618). Kết quả phản ánh đúng với tầm quan trọng của các đặc trưng theo bảng Table 1
- **Hours Studied** có MAE cao hơn đáng kể (15.449), cho thấy mặc dù thời gian học quan trọng nhưng không phải là yếu tố dự báo mạnh nhất khi đứng một mình.
- **Sleep Hours**, **Sample Question Papers Practiced** và **Extracurricular Activities** có MAE tương đối gần nhau (khoảng 16.18–16.20).
- Thứ tự xếp hạng dựa trên MAE phù hợp với phân tích tầm quan trọng đặc trưng trong Table 1.

3.2.3 c. Mô hình tự thiết kế

Ý tưởng tạo ra các đặc trưng mới (feature engineering): Mục tiêu của việc bổ sung đặc trưng là khai thác thông tin tiềm ẩn từ các biến gốc, giúp mô hình nắm bắt tốt hơn các mối quan hệ phi tuyến và tác động kết hợp giữa các yếu tố. Các nhóm đặc trưng bổ sung bao gồm:

- **Biến đa thức (Polynomial)**: Bắt được mối quan hệ phi tuyến giữa đặc trưng và mục tiêu.
  - `Hours_sq`: Bình phương số giờ học, giúp nhận diện các trường hợp học quá nhiều hoặc quá ít ảnh hưởng khác nhau đến hiệu suất.
  - `PrevScores_sq`: Bình phương điểm số trước đây.
  - `Sleep_sq`: Bình phương số giờ ngủ, xem tác động của việc ngủ quá nhiều hoặc quá ít.
  - `Sample_Papers_sq`: Bình phương số đề luyện tập đã thực hiện.

- **Biến tương tác (Interaction)**: Mô tả tác động kết hợp giữa hai yếu tố.
  - `Hours_x_Sleep`: Tích của giờ học và giờ ngủ, xem xét mối quan hệ giữa học tập và nghỉ ngơi.
  - `Hours_x_Prev`: Kết hợp số giờ học với điểm số trước đây, để kiểm tra khả năng bù đắp của việc học nhiều đối với thành tích quá khứ.
- **Biến tùy chỉnh (Custom)**: Các chỉ số phản ánh hiệu quả hoặc cân bằng giữa các yếu tố.
  - `Practice_effort`: Kết hợp số đề luyện tập và số giờ học, phản ánh "nỗ lực thực tế" của học sinh.
  - `Paper_score_ratio`: So sánh số đề luyện tập với điểm số trước, giúp nhận biết mức độ luyện tập so với năng lực hiện tại; học sinh điểm thấp nhưng luyện tập nhiều có thể cải thiện hiệu suất.
  - `Study_Over_Rest`: Tỷ lệ giữa số giờ học và thời gian nghỉ ngơi điều chỉnh, tính bằng:

$$\text{Study\_Over\_Rest} = \frac{\text{Hours Studied}}{\text{Total\_Rest} + \epsilon}$$

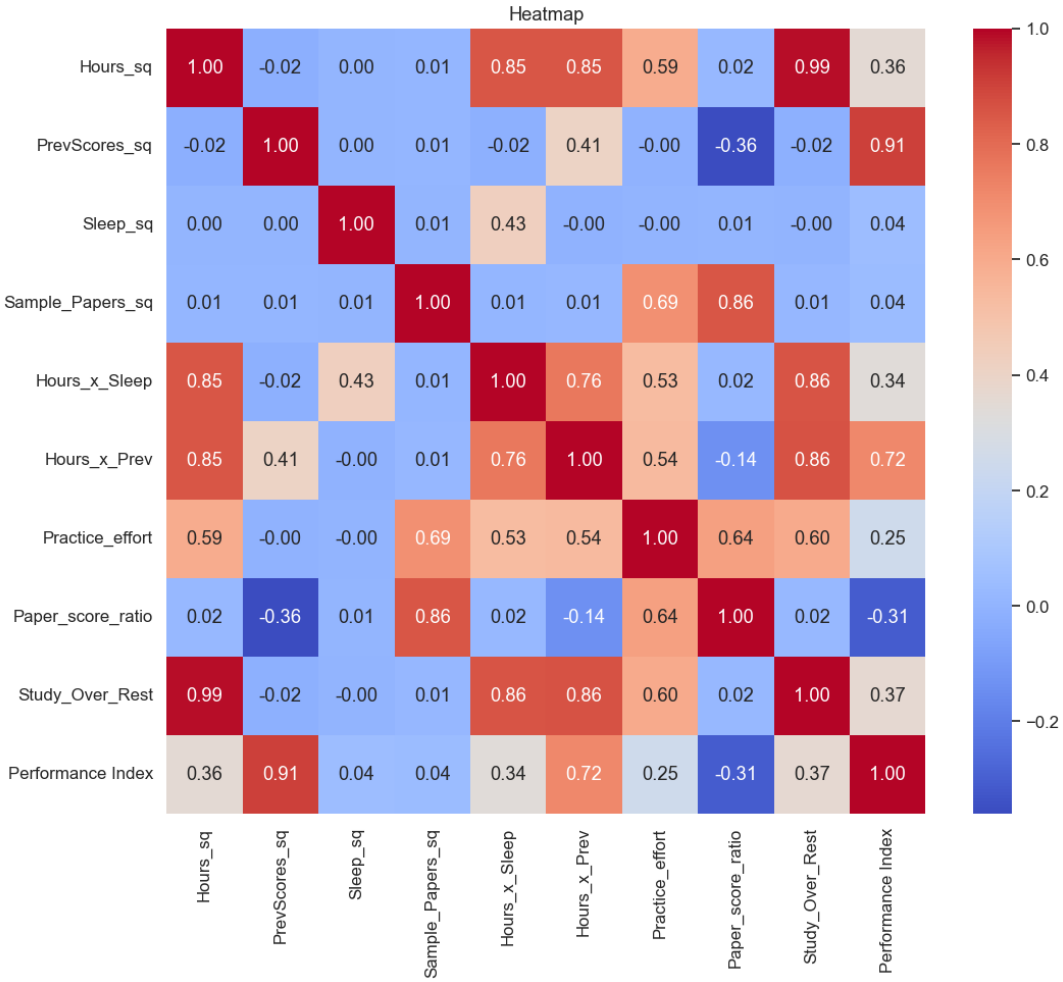
    với $\epsilon = 10^{-6}$ để tránh chia cho 0.
    Giả sử sinh viên có tham gia hoạt động ngoại khóa sẽ dành 2 giờ/ngày để dành cho hoạt động, thời gian nghỉ thật sự của sinh viên sẽ là:

$$Total\_Rest = 24 - HoursStudied - 2 * (ExtracurricularActivities)$$

    Đo mức độ "căng thẳng học tập" so với thời gian nghỉ ngơi; giá trị cao cho thấy học sinh dành nhiều thời gian học so với nghỉ ngơi, có thể ảnh hưởng đến hiệu suất học tập.

Ma trận tương quan

Dựa trên ma trận tương quan giữa các đặc trưng mới và **Performance Index**, lựa chọn các mô hình thử nghiệm với các chiến lược chọn đặc trưng khác nhau. Hình 6 minh họa correlation heatmap giữa các đặc trưng và target:



Hình 7: Correlation heatmap giữa các đặc trưng và **Performance Index**.

Chiến lược lựa chọn mô hình

Model 1 - Top 3 Most Influential Features: Chọn 3 đặc trưng gốc có tương quan cao nhất với target: `Previous Scores`, `Hours Studied`, `Sleep Hours`. Như minh họa trong Hình 6, các đặc trưng này có hệ số tương quan lần lượt là (0.915, 0.369, 0.044)

Model 2 - Top 2 Most Influential Features + Squared Terms: Chọn 2 đặc trưng gốc có correlation cao nhất (`Hours Studied`, `Previous Scores`) và thêm các bình phương của chúng (`Hours_sq`, `PrevScores_sq`). Thử nghiệm khả năng nắm bắt các mối quan hệ phi tuyến giữa những đặc trưng quan trọng và target. Bằng cách sử dụng bình phương Hours_sq và PrevScores_sq vì chúng có hệ số tương quan cao nhất trong số các biến bình phương.

Model 3 - Interaction-Based Features: Chọn các đặc trưng kết hợp hoặc ratio dựa trên correlation cao trong heatmap: `Paper_score_ratio`, `Practice_effort`, `Study_Over_Rest`, `Hours_x_Prev`.

Lý do: Kiểm tra xem các đặc trưng thể hiện tương tác giữa các yếu tố (ví dụ thời gian học và số lượng bài tập) có cải thiện dự đoán không, đặc biệt khi các mối quan hệ phi tuyến và tác động phối hợp quan trọng.

Model 4 - Combined Polynomial + Interaction: Kết hợp tất cả đặc trưng cơ bản, 2 bình phương có correlation cao nhất (`Hours_sq`, `PrevScores_sq`) và 2 interaction mạnh nhất (`Hours_x_Sleep`, `Hours_x_Prev`). Lý do: Mô hình đầy đủ nhất, kết hợp tuyến tính, phi tuyến và tương tác, tận dụng toàn bộ thông tin quan sát được từ dữ liệu. Mục tiêu là so sánh sức mạnh dự đoán của mô hình đầy đủ với các mô hình đơn giản hơn.

Kết quả chạy cross-validation cho các mô hình 2c

Model	**MAE**
Model 4: Combined Polynomial + Interaction	1.621631
Model 1: Top 3 Most Influential Features	1.702118
Model 2: Top 2 Most Influential Features + Squared Terms	1.815747
Model 3: Interaction-Based Features	6.809077

Bảng 3: MAE trung bình tính bằng cross-validation cho từng mô hình 2c

Giải thích kết quả cross-validation

Như Bảng 3 cho thấy, **Model 4: Combined Polynomial + Interaction** đạt **MAE thấp nhất** trên tập dữ liệu huấn luyện. Lý do chính bao gồm:

- **Kết hợp đầy đủ các đặc trưng quan trọng:** Model 4 sử dụng tất cả các đặc trưng cơ bản, kết hợp với các biến bình phương có correlation cao và các interaction mạnh nhất. Việc này giúp mô hình tận dụng tối đa thông tin từ dữ liệu.
- **Bắt được mối quan hệ phi tuyến:** Các biến bình phương (polynomial) cho phép mô hình nắm bắt các ảnh hưởng phi tuyến giữa đặc trưng và target.
- **Bắt được tác động phối hợp giữa các yếu tố:** Các biến interaction phản ánh mối liên hệ đồng thời giữa hai đặc trưng, giúp mô hình dự đoán chính xác hơn khi nhiều yếu tố thay đổi đồng thời.

Do đó, Model 4 tận dụng cả thông tin tuyến tính, phi tuyến và tương tác, giúp MAE thấp nhất trong số các mô hình thử nghiệm.

Phương trình hồi quy

$$\begin{aligned} Performance\ Index = &-34.097 \quad &+2.787\,x_1 \\ &+1.029\,x_2 \quad &+0.605\,x_3 \\ &+0.444\,x_4 \quad &+0.193\,x_5 \\ &+0.004\,x_6 \quad &-0.0001\,x_7 \\ &+0.006\,x_8 \quad &-0.0002\,x_9 \end{aligned}$$

Trong đó:

- x_1: Hours Studied
- x_2: Previous Scores
- x_3: Extracurricular Activities
- x_4: Sleep Hours
- x_5: Sample Question Papers Practiced
- x_6: Hours_sq (Hours Studied^2)
- x_7: PrevScores_sq (Previous Scores^2)
- x_8: Hours_x_Sleep (Hours Studied $\times$ Sleep Hours)
- x_9: Hours_x_Prev (Hours Studied $\times$ Previous Scores)

Giá trị MAE trên tập test:

$$\text{MAE}_{\text{test}} = 1.59373$$

Tài liệu

[1] Lab 04: Ols linear regression. File hướng dẫn Lab04 của môn, 2025. Truy cập ngày 6/8/2025.

[2] GeeksforGeeks. Ordinary least squares (ols). https://www.geeksforgeeks.org/machine-learning/ordinary-least-squares-ols/, 2025. Truy cập ngày 7/8/2025.

[3] Stephen Boyd and Lieven Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares*. Stanford University / University of California, Los Angeles, 2024. Chương 13.3: Feature engineering, Truy cập ngày 7/8/2025.

[4] GeeksforGeeks. Cross validation in machine learning. https://www.geeksforgeeks.org/machine-learning/cross-validation-machine-learning/, 2025. Truy cập ngày 8/8/2025.

[5] Stephen Boyd and Lieven Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares*. Stanford University / University of California, Los Angeles, 2024. Chương 13.2: Validation, Truy cập ngày 7/8/2025.

4 Acknowledgement

- Em xin chân thành cảm ơn cô Trần Thị Thảo Nhi đã giao đề tài thú vị, tạo cơ hội để em tìm hiểu sâu về các phương pháp hồi quy tuyến tính, đặc biệt là thuật toán **Ordinary Least Squares (OLS)** và kỹ thuật **K-fold Cross-Validation** để chọn mô hình dự đoán Performance Index phù hợp. Sự hướng dẫn tận tình của cô giúp em nâng cao kiến thức về mô hình hóa, phân tích dữ liệu, cũng như cách trình bày báo cáo khoa học một cách mạch lạc và chuyên nghiệp.

- Em xin cảm ơn các nguồn tài liệu tham khảo:

  – Trang **GeeksforGeeks** đã cung cấp thông tin và ví dụ minh họa về OLS, K-fold Cross-Validation và cách triển khai trong Python.

  – Sách **Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares** của Stephen Boyd và Lieven Vandenberghe giúp em hiểu rõ các khái niệm lý thuyết nền tảng và kỹ thuật feature engineering.

- Em cũng xin cảm ơn **ChatGPT** đã hỗ trợ em trong quá trình phát triển và hoàn thiện báo cáo, cụ thể:

  – Gợi ý cách thiết kế và cải tiến các hàm kiểm tra mô hình, giúp đánh giá hiệu quả các giá trị đặc trưng và so sánh kết quả dự đoán.

  – Viết docstrings các hàm gọn gàng

  – Viết chú thích code rõ ràng, dễ hiểu, hỗ trợ việc bảo trì và mở rộng sau này.

  – Hướng dẫn cách trình bày LaTeX báo cáo, giúp sắp xếp nội dung khoa học, logic và chuyên nghiệp.

Nhờ sự hỗ trợ từ giảng viên, tài liệu tham khảo uy tín và công cụ AI như ChatGPT, em đã hoàn thiện đồ án đúng tiến độ, nâng cao kỹ năng lập trình, phân tích dữ liệu và trình bày kết quả khoa học.