TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN 3: LINEAR REGRESSION

TOÁN ỨNG DỤNG VÀ THỐNG KÊ

Triệu Nhật Minh — 21127112 — 21CLC02

Giảng viên hướng dẫn

Vũ Quốc Hoàng
Lê Thanh Tùng
Nguyễn Văn Quang Huy
Phan Thị Phương Uyên

Ngày 22 tháng 8 năm 2023

Mục lục

1 Thư viện sử dụng

1.1 pandas

Thư viện cho phép thao tác với dữ liệu dạng bảng. pandas cung cấp các đối tượng DataFrame và Series, cho phép lưu trữ, truy xuất, lọc, nhóm, biến đổi và thống kê dữ liệu một cách hiệu quả và dễ dàng. Trong đồ án này, pandas được sử dụng để đọc dữ liệu từ file csv và lưu trữ dữ liệu dưới dạng DataFrame.

1.2 numpy

Thư viện cho phép thao tác với mảng nhiều chiều. Với bài toán data fitting sử dụng phương pháp bình phương tối thiểu (OLS Linear Regression), để giải nghiệm x cho hệ phương trình được tính bằng công thức $x = (A^TA)^{-1}A^Tb$. Nhằm tối ưu thời gian tính toán, ta sử dụng hàm có sẵn từ thư viện này. Hầu hết các hàm có sẵn đã quen thuộc từ những đồ án trước, duy có hàm *numpy.ravel* và *numpy.triu* sẽ được giải thích rõ hơn ở phần liệt kê hàm.

1.3 matplotlib

Thư viện matplotlib (cụ thể là module pyplot) cho phép ta tạo ra các biểu đồ dạng 2D. matplotlib.pyplot cũng cho phép điều chỉnh các thuộc tính của đồ thị, như màu sắc, kích thước, chú thích và tiêu đề cho đồ thị.

1.4 seaborn

Thư viện cho phép vẽ heatmap. Heatmap là một loại biểu đồ 2D biểu diễn giá trị của một ma trận bằng cách sử dụng các ô có màu sắc khác nhau. seaborn cung cấp các hàm để vẽ heatmap từ các đối tượng DataFrame hoặc numpy array, cũng như điều chỉnh các thuộc tính như bản đồ màu, khoảng giá trị, nhãn và tiêu đề. Heatmap là thành phần không thể thiếu để tìm hiểu mối quan hệ giữa các biến trong bộ dữ liệu và là tiền đề để thực hiện tìm mô hình cho yêu cầu 1d.

1.5 sklearn

Thư viện cho phép chia dữ liệu thành các fold để thực hiện cross-validation. Cross-validation là một kỹ thuật kiểm tra hiệu năng của mô hình học máy bằng cách sử dụng một phần của dữ liệu làm tập kiểm tra và phần còn lại làm tập huấn luyện. *sklearn.model_selection.KFold* cho phép chia dữ liệu thành k fold có kích thước bằng nhau và lặp qua từng fold để sử dụng làm tập kiểm tra hoặc tập huấn luyện.

sklearn.model_selection.KFold là một lớp trong thư viện scikit-learn, cung cấp các chỉ số để chia dữ liệu thành các tập huấn luyện và kiểm tra. Nó chia tập dữ liệu thành k fold liên tiếp. Mỗi fold được sử dụng một lần làm tập kiểm tra trong khi các fold còn lại được sử dụng làm tập huấn luyện.

Hàm tự cài đặt có thể thực hiện chức năng tương tự như KFold, nhưng có thể khác biệt về hiệu suất và tính năng. Việc sử dụng KFold từ scikit-learn có thể đảm bảo tính ổn định và độ tin cậy của kết quả, do nó được sử dụng rộng rãi trong cộng đồng khoa học dữ liệu, nhất là khi bộ dữ liệu được sử dụng trong đồ án khó có thể kiểm tra thủ công. Tuy nhiên, một hàm tự cài đặt có thể được tùy chỉnh để phù hợp với nhu cầu đặc biệt của người dùng, thậm chí có thể có hiệu suất tốt hơn so với KFold trong một số ít trường hợp.

1.6 IPython

Thư viện này không đóng góp vào việc giải quyết bài toán, nhưng vẫn được sử dụng vì khả năng hiển thị ngôn ngữ LaTeX để trình bày công thức hồi quy tuyến tính cho yêu cầu 1a do số lượng biến lớn. Module IPython.display.Latex cho phép chèn các biểu thức LaTeX vào Jupyter Notebook.

2 Hàm sử dụng

2.1 Hàm built-in từ thư viện

2.1.1 Hàm pandas.read_csv

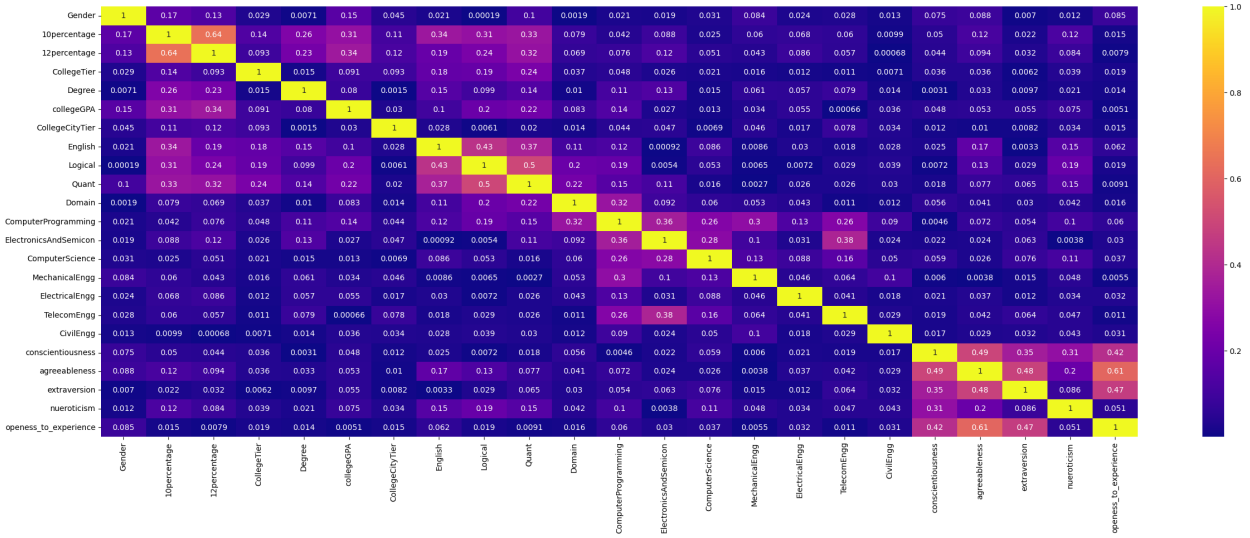
Input: Đường dẫn đến file csv.
Output: DataFrame chứa dữ liệu từ file csv.

Mô tả: Hàm *pandas.read_csv* [5] được sử dụng để đọc dữ liệu từ file csv và lưu trữ dữ liệu dưới dạng DataFrame. Hàm này có thể nhận thêm các tham số để tùy chỉnh cách đọc dữ liệu, nhưng trong đồ án này ta giữ nguyên các tham số khác xem như mặc định, chỉ tùy chỉnh đường dẫn đến file csv.

2.1.2 Hàm pandas.drop

Input: Tên cột cần xóa.
Output: DataFrame sau khi đã xóa cột.

Mô tả: Trong yêu cầu 1d, để xây dựng mô hình chứa các đặc trưng chứa ít sự tương quan nhát (least correlation features), sau khi đã tìm được ma trận tương quan giữa các cột, ta sẽ xóa các cột có độ tương quan cao hơn 0.6. Mặc dù trong cộng đồng khoa học dữ liệu, các đặc trưng có độ tương quan cao hơn 0.95 mới được xem là có sự tương quan cao, nhưng trong đồ án này, dựa vào ma trận tương quan thì giá trị lớn nhất là 0.64 nên ta sẽ xóa các cột có độ tương quan cao hơn 0.6. Và hàm *pandas.drop* [4] được gọi để xóa cột dựa trên tên cột thoả yêu cầu đã đề cập.



Hình 1: Ma trận tương quan giữa các cột trong yêu cầu 1d

2.1.3 Hàm pandas.corr

Input: Các tham số để tùy chỉnh cách tính ma trận tương quan.
Output: DataFrame chứa ma trận tương quan giữa các cột.

Mô tả: Ma trận tương quan (correlation matrix) là một ma trận vuông chứa các hệ số tương quan giữa nhiều biến. Mỗi ô trong bảng cho biết mối tương quan giữa hai biến cụ thể. Ma trận tương quan thường được sử dụng để tóm tắt dữ liệu, làm đầu vào cho phân tích nâng cao hơn và làm chẩn đoán cho phân tích nâng cao

Một số điểm cần lưu ý khi đọc ma trận tương quan:

- Các hệ số tương quan trên đường chéo của bảng đều bằng 1 vì mỗi biến hoàn toàn tương quan với chính nó.
- Chỉ một nửa của ma trận tương quan cần được hiển thị vì nửa còn lại của các hệ số tương quan trong ma trận là dư thừa và không cần thiết.
- Ta có thể tô màu ma trận tương quan sẽ được như một bản đồ nhiệt (sử dụng tham số *cmap* trong hàm heatmap) để làm cho các hệ số tương quan dễ đọc hơn.

Trong đồ án này, phương pháp Pearson [6] được sử dụng để tính ma trận tương quan. Phương pháp này được sử dụng để đo lường mối quan hệ giữa hai biến ngẫu nhiên X và Y. Giá trị của hệ số tương quan Pearson nằm trong khoảng [-1, 1]. Hệ số tương quan bằng 1 nếu có mối quan hệ tuyến tính thuận hoàn hảo giữa hai biến, bằng -1 nếu có mối quan hệ tuyến tính nghịch hoàn hảo giữa hai biến. Hệ số tương quan bằng 0 nếu không có mối quan hệ tuyến tính giữa hai biến.

Công thức tính hệ số tương quan Pearson giữa hai biến X và Y:

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{1}$$

Trong đó:

- $r_{X,Y}$ là hệ số tương quan Pearson giữa hai biến X và Y.
- x_i là giá trị của biến X tại điểm dữ liệu thứ i.
- y_i là giá trị của biến Y tại điểm dữ liệu thứ i.
- $\bar{x}$ là giá trị trung bình của biến X.
- $\bar{y}$ là giá trị trung bình của biến Y.
- n là số lượng điểm dữ liệu.

Do ta cần quan tâm độ tương quan giữa hai đặc trưng chứ không quan tâm chúng có tương quan thuận hay nghịch, nên ta sẽ lấy giá trị tuyệt đối của hệ số tương quan Pearson để tính ma trận tương quan. Bằng cách gọi hàm *abs()* trên DataFrame chứa ma trận tương quan, ta sẽ được ma trận tương quan giữa các cột như hình 1.

2.1.4 Hàm numpy.linalg.inv

Input: Ma trận vuông cần tính ma trận nghịch đảo.
Output: Ma trận nghịch đảo của ma trận đầu vào.

Mô tả: Hàm *numpy.linalg.inv* [2] được sử dụng để tính ma trận nghịch đảo của ma trận vuông. Trong đồ án này, hàm này được sử dụng để tính ma trận nghịch đảo của ma trận A^TA để tìm nghiệm của hệ phương trình tuyến tính $x = (A^TA)^{-1}A^Tb$ (hàm *fit* trong class OLSLinearRegression tự cài đặt, song để tối ưu hoá thời gian tính toán, ta sẽ sử dụng hàm *numpy.linalg.inv* để tính ma trận nghịch đảo thay vì tự cài đặt).

2.1.5 Hàm numpy.ravel

Input: Mảng NumPy được đọc theo thứ tự được chỉ định bởi tham số order và được đóng gói thành một mảng 1 chiều và được đọc theo thứ tự C.

Output: Mảng NumPy một chiều.

Mô tả: Trong numpy, tham số order được sử dụng để chỉ định cách mà các phần tử của một mảng được lưu trữ trong bộ nhớ C trong *order='C'*, có nghĩa là mảng được lưu trữ theo thứ tự liên tục của C, hay chỉ số cuối cùng thay đổi nhanh nhất [1]. Điều này có nghĩa là khi bạn duyệt qua các phần tử của một mảng nhiều chiều theo thứ tự C, bạn sẽ duyệt qua các phần tử của chỉ số cuối cùng trước, sau đó tăng chỉ số kế cuối lên 1 và tiếp tục duyệt qua các phần tử của chỉ số cuối cùng, và cứ tiếp tục như vậy cho đến khi duyệt hết các phần tử của mảng.

Trong bài toán hồi quy tuyến tính, chúng ta cần tìm một ma trận trọng số self.w sao cho tổng bình phương sai số giữa giá trị dự đoán và giá trị thực tế là nhỏ nhất. Để làm được điều này, chúng ta cần biến đổi ma trận trọng số thành một vector có kích thước bằng với số lượng tham số trong mô hình. Hàm *numpy.ravel()* [3] giúp chúng ta thực hiện việc này một cách dễ dàng và nhanh chóng. Bằng cách sử dụng hàm này, chúng ta có thể tính toán tích vô hướng giữa vector trọng số và ma trận đầu vào X bằng cách nhân từng phần tử tương ứng và cộng lại bằng cách gọi hàm *numpy.sum*. Cuối cùng ta thu được giá trị dự đoán cho từng quan sát trong ma trận đầu vào X.

Hàm *numpy.ravel* cũng được sử dụng trong hàm mae 2.2.4 để làm phẳng các mảng đầu vào y và y_hat thành các mảng 1 chiều. Điều này cho phép tính toán trực tiếp sự khác biệt tuyệt đối giữa các phần tử tương ứng của hai mảng bằng cách trừ chúng và lấy giá trị tuyệt đối. Sau đó, giá trị trung bình của các sự khác biệt tuyệt đối được tính toán bằng hàm *numpy.mean* để trả về giá trị lỗi trung bình tuyệt đối (MAE) giữa hai mảng.

2.1.6 Hàm numpy.sum

2.1.7 Hàm numpy.mean

2.1.8 Hàm numpy.triu

2.1.9 Hàm numpy.where

2.1.10 Hàm numpy.ones

2.1.11 Hàm seaborn.heatmap

2.1.12 Hàm sklearn.model_selection.KFold

2.1.13 Hàm IPython.display.Latex

2.2 Hàm tự cài đặt

2.2.1 Hàm OLSLinearRegression.fit

2.2.2 Hàm OLSLinearRegression.predict

2.2.3 Hàm OLSLinearRegression.get_params

2.2.4 Hàm mae

2.2.5 Hàm latex_text

2.2.6 Hàm kfold_cross_model

2.2.7 Hàm scientific_notation_converter

3 Đánh giá kết quả mô hình

3.1 Yêu cầu 1a

3.2 Yêu cầu 1b

3.3 Yêu cầu 1c

3.4 Yêu cầu 1d

4 Tài liệu tham khảo

[1] *Cheapest way to get a numpy array into C-contiguous order? — stackoverflow.com*. https://stackoverflow.com/questions/29947639/cheapest-way-to-get-a-numpy-array-into-c-contiguous-order. [Accessed 11-08-2023].

[2] *numpy.linalg.inv; NumPy v1.25 Manual — numpy.org*. https://numpy.org/doc/stable/reference/generated/numpy.linalg.inv.html. [Accessed 11-08-2023].

[3] *numpy.ravel; NumPy v1.25 Manual — numpy.org*. https://numpy.org/doc/stable/reference/generated/numpy.ravel.html. [Accessed 11-08-2023].

[4] *pandas.DataFrame.drop; pandas 2.0.3 documentation — pandas.pydata.org*. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html. [Accessed 19-08-2023].

[5] *pandas.read_csv; pandas 2.0.3 documentation — pandas.pydata.org*. https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html. [Accessed 11-08-2023].

[6] *Pearson correlation coefficient - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/Pearson_correlation_coefficient. [Accessed 19-08-2023].