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 24 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

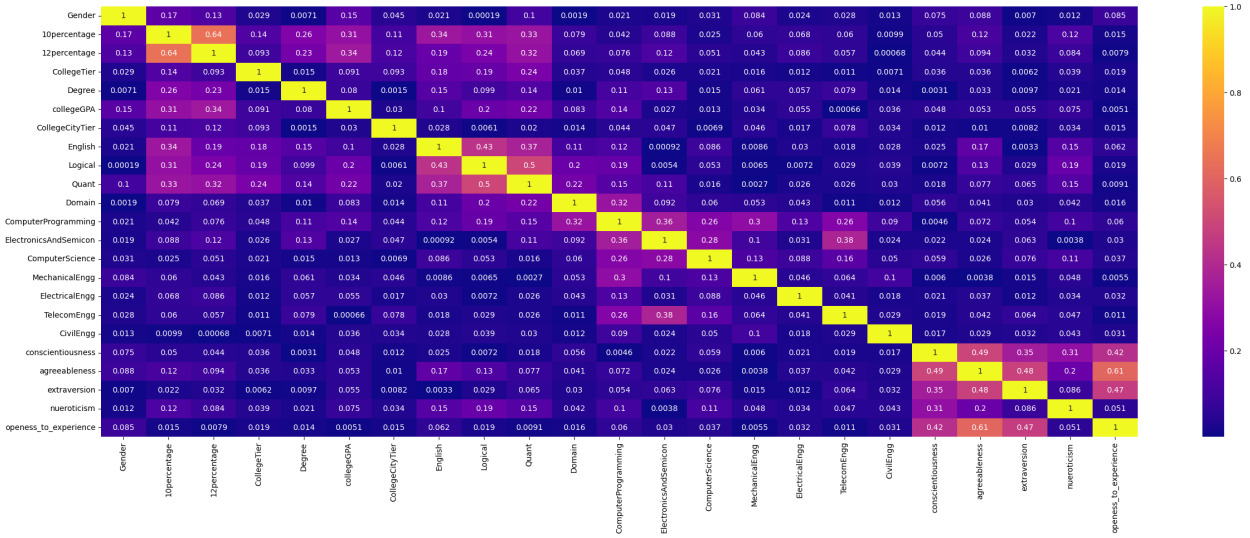
Input: Đường dẫn đến file csv.
Output: DataFrame chứa dữ liệu từ file csv.

Mô tả: Hàm *pandas.read_csv* [19] 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* [18] 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 [20] 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

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* [11] 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 [3]. 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()* [14] 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 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

Input: Mảng NumPy cần tính tổng, cột cần tính tổng
Output: Hàm này trả về kết quả là tổng của các phần tử trong mảng đầu vào theo các tham số đã xác định. Hàm có thể xử lý các mảng có kích thước và chiều khác nhau, và có thể thực hiện broadcasting nếu các kích thước tương thích.

Mô tả: Hàm *numpy.sum* [15] được sử dụng để tính tổng các phần tử trong mảng. Trong đồ án này, hàm này được sử dụng để tính tổng các phần tử trong mảng đầu vào X bằng cách nhân từng phần tử tương ứng của vector trọng số và ma trận đầu vào X và cộng lại. Cuối cùng ta thu được giá trị dự đoán cho từng quan sát trong ma trận đầu vào X.

2.1.7 Hàm numpy.mean

Input: Mảng NumPy cần tính trung bình, cột cần tính trung bình
Output: Hàm này trả về kết quả là trung bình của các phần tử trong mảng đầu vào theo các

tham số đã xác định. Hàm có thể xử lý các mảng có kích thước và chiều khác nhau, và có thể thực hiện broadcasting nếu các kích thước tương thích.

Mô tả: Hàm *numpy.mean* [12] được sử dụng để tính giá trị trung bình của các phần tử trong mảng. Trong đồ án này, hàm này được sử dụng để tính giá trị trung bình của các 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.

2.1.8 Hàm numpy.triu

Input: Mảng NumPy cần tìm ma trận tam giác trên, kích thước của ma trận tam giác trên
Output: Mảng NumPy chứa ma trận tam giác trên của mảng đầu vào.

Mô tả: Hàm này trả về một bản sao của mảng đầu vào với các phần tử dưới đường chéo thứ k bị đưa về 0. Đối với các mảng có số chiều lớn hơn 2, triu sẽ áp dụng cho hai trục cuối cùng [16].

2.1.9 Hàm numpy.where

Input: Mảng NumPy cần tìm vị trí, giá trị cần tìm vị trí
Output: Mảng NumPy chứa các vị trí của giá trị cần tìm.

Mô tả: Hàm *numpy.where* [17] được sử dụng để lọc các phần tử của một mảng dựa trên một điều kiện cho trước. Cụ thể hơn, *np.triu(np.ones(corr_matrix.shape), k=1).astype(bool)* sẽ tạo ra một mảng có cùng kích thước với corr_matrix, trong đó các phần tử nằm trên đường chéo chính (k=1) sẽ có giá trị là True, còn các phần tử còn lại có giá trị là False.

Sau đó, mảng này được sử dụng như một mặt nạ để lọc các phần tử của corr_matrix bằng cách sử dụng phương thức where. Kết quả cuối cùng là một mảng mới chỉ chứa các phần tử nằm trên đường chéo chính của corr_matrix, còn các phần tử còn lại sẽ bị loại bỏ (có giá trị là NaN).

Nói cách khác, đoạn code trên lọc ra ma trận tam giác trên của ma trận tương quan corr_matrix, loại bỏ các phần tử nằm dưới đường chéo chính và giữ lại các phần tử nằm trên đường chéo chính.

2.1.10 Hàm numpy.ones

Input: Kích thước của mảng, kiểu dữ liệu của mảng
Output: Mảng NumPy chứa các phần tử có giá trị là 1.

Mô tả: Hàm *numpy.ones* [13] được sử dụng để tạo ra một mảng có kích thước và kiểu dữ liệu nhất định, trong đó các phần tử có giá trị là 1.

2.1.11 Hàm seaborn.heatmap

Input: Ma trận tương quan, các tham số để tùy chỉnh cách vẽ heatmap.
Output: Biểu đồ heatmap.

Mô tả: Hàm *seaborn.heatmap* [24] được sử dụng 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.

Tham số *annot* trong hàm *sns.heatmap* được sử dụng để kiểm soát việc hiển thị giá trị của các ô trong biểu đồ heatmap. Nếu *annot=True*, thì giá trị của mỗi ô sẽ được hiển thị bên trong ô đó.

Biểu đồ heatmap với ma trận tương quan corr_matrix, sử dụng bản đồ màu plasma, và hiển thị giá trị của mỗi ô bên trong ô đó. Nếu tham số này là False, thì giá trị của mỗi ô sẽ không được hiển thị bên trong ô đó, chỉ hiển thị các màu tương ứng với giá trị của ô đó.

2.1.12 Lớp sklearn.model_selection.KFold

Input: Các tham số để tùy chỉnh cách chia dữ liệu.
Output: Đối tượng KFold chứa các chỉ số để chia dữ liệu thành các tập huấn luyện và kiểm tra.

Mô tả: Chiến lược phổ biến nhất trong học máy là chia tập dữ liệu thành tập huấn luyện và tập kiểm tra. Tỷ lệ chia có thể là 70:30 hoặc 80:20 [6]. Một trong số những phương pháp là sử dụng k-fold Cross Validation. [26]

Trong đồ án này, lớp *sklearn.model_selection.KFold* [25] được sử dụng để chia dữ liệu thành các fold. Nó có thể nhận thêm các tham số để tùy chỉnh cách chia dữ liệu, nhưng trong đồ án này ta giữ nguyên các tham số khác xem như mặc định, chỉ tùy chỉnh số lượng fold, kiểu chia dữ liệu và trộn dữ liệu trước khi chia (lần lượt các tham số *n_splits*, *shuffle* và *random_state*). Tham số *n_splits* được sử dụng để chỉ định số lượng fold (k = 20 với các yêu cầu 1b, 1c, 1d), tham số *shuffle* được sử dụng để chỉ định cách chia dữ liệu, và tham số *random_state* được sử dụng để chỉ định cách trộn dữ liệu trước khi chia.

Tham số *shuffle* được sử dụng để chỉ định cách chia dữ liệu. Nếu *shuffle=True*, thì dữ liệu sẽ được trộn ngẫu nhiên trước khi chia. Nếu *shuffle=False*, thì dữ liệu sẽ không được trộn ngẫu nhiên trước khi chia. Trong đồ án này, ta sử dụng *shuffle=True* để trộn dữ liệu ngẫu nhiên trước khi chia.

Đến với tham số *random_state*, nếu *random_state=None*, thì mỗi lần chạy, lớp *KFold* sẽ cho ra kết quả khác nhau. Trên thực tế, việc truyền bất kì số nguyên nào (kể cả số 0) thì cũng không

thành vấn đề. Đây là một random seed trong thuật toán random của máy tính. [21] Các thí nghiệm có cùng random seed và cùng các tham số khác sẽ cho ra kết quả giống nhau. 42 là một con số đến từ cuốn sách Hướng dẫn du lịch vũ trụ. Câu trả lời cho cuộc sống vũ trụ và mọi thứ và được coi là một trò đùa. Nó không có ý nghĩa khác ngoài việc là một con số ngẫu nhiên. [22] Trong đồ án này, ta sử dụng *random_state=42* để đảm bảo kết quả của các thí nghiệm là nhất quán.

Việc chia dữ liệu thành k phần giúp chúng ta có thể kiểm tra độ chính xác và ổn định của mô hình trên nhiều tập dữ liệu khác nhau, tránh hiện tượng quá khớp (overfitting) hoặc thiếu khớp (underfitting) mô hình. Việc lựa chọn giá trị k trong k-fold cross validation là tùy thuộc vào nhiều yếu tố. Một giá trị k lớn sẽ cho phép mô hình được huấn luyện trên nhiều dữ liệu hơn, nhưng cũng sẽ tăng thời gian tính toán. Một giá trị k nhỏ sẽ giảm thời gian tính toán, nhưng cũng có thể làm tăng sai số trong việc đánh giá hiệu suất của mô hình. [4]

2.1.13 Hàm IPython.display.Latex

Input: Chuỗi kí tự cần hiển thị ở dạng LaTeX.
Output: Hiển thị chuỗi kí tự ở dạng LaTeX.

2.2 Hàm tự cài đặt

2.2.1 Hàm OLSLinearRegression.fit

Input: Ma trận chứa các giá trị của đặc trưng, y là một vector chứa các giá trị của biến mục tiêu.
Output: Chính nó (self) để có thể gọi phương thức khác trên cùng đối tượng OLSLinearRegression.

Mô tả: Hàm này sử dụng phương pháp bình phương nhỏ nhất để tính toán trọng số w cho mô hình tuyến tính. Đầu tiên, nó tính toán ma trận giả nghịch đảo của ma trận A^TA bằng hàm *numpy.linalg.inv* (thay vì tự cài đặt để tối ưu hoá thời gian tính toán).

Mục đích hàm fit của lớp OLSLinearRegression là tìm ra ma trận trọng số w ứng với mô hình tuyến tính. Để làm được điều này, ta cần giải hệ phương trình tuyến tính $A^TAw = A^Tb$ để tìm ra giá trị của w. Để giải hệ phương trình này, ta nhân cả hai vế của phương trình với ma trận nghịch đảo của ma trận A^TA, ta được $w = (A^TA)^{-1}A^Tb$. Do đó, ta sẽ tính ma trận nghịch đảo của ma trận A^TA bằng hàm *numpy.linalg.inv* và nhân với ma trận A^Tb để tìm ra giá trị của w.

2.2.2 Hàm OLSLinearRegression.predict

Input: Ma trận chứa các giá trị của đặc trưng.
Output: Mảng NumPy chứa các giá trị dự đoán.

Mô tả: Trong hàm *predict*, ta tính toán giá trị dự đoán bằng cách nhân ma trận X với vector trọng số w (được tính toán trong hàm và cộng các giá trị lại với nhau theo chiều thứ nhất (axis=1) để thu được một vector kết quả. Cụ thể, ta sử dụng phép toán *np.sum(self.w.ravel() * X, axis=1)* để thực hiện việc này.

Sở dĩ ta phải gọi hàm *ravel* được sử dụng để chuyển đổi vector trọng số w thành một mảng 1 chiều liên tục trước khi thực hiện phép nhân với ma trận đầu vào X. Điều này cần thiết để đảm bảo rằng kích thước của w và X phù hợp với nhau và phép nhân. Trái lại, nếu không sử dụng hàm *ravel*, vector trọng số w có kích thước không phù hợp và dẫn đến lỗi khi thực hiện phép nhân.

2.2.3 Hàm OLSLinearRegression.get_params

Input: Không có.
Output: Mảng NumPy chứa các giá trị của vector trọng số w.

Mô tả: Hàm *get_params* được sử dụng để trả về các giá trị của vector trọng số w. Mô hình OLSLinearRegression sau khi được huấn luyện sẽ có một vector trọng số w duy nhất, và hàm *get_params* được sử dụng để trả về các giá trị của vector trọng số này.

2.2.4 Hàm mae

Input: Vector chứa các giá trị thực tế từ tập dữ liệu cho trước, vector chứa các giá trị dự đoán tính được từ mô hình.
Output: Giá trị lỗi trung bình tuyệt đối (MAE).

Mô tả: Phương pháp bình phương tối thiểu khi sử dụng để tìm mô hình tuyến tính sẽ tìm ra mô hình có giá trị trung bình của tổng bình phương sai số là nhỏ nhất. Tuy nhiên, trong thực tế, chúng ta thường quan tâm đến giá trị trung bình của tổng sai số tuyệt đối, hay còn gọi là lỗi trung bình tuyệt đối (MAE). MAE được tính bằng cách lấy giá trị tuyệt đối của sự khác biệt giữa các giá trị thực tế và giá trị dự đoán, sau đó lấy giá trị trung bình của các sự khác biệt tuyệt đối này.

Trong hàm mae, ta sử dụng hàm *numpy.ravel* 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.

2.2.5 Hàm latex_text

Input: Mảng NumPy chứa các tham số của mô hình tuyến tính, từ điển chứa các tên cột tương ứng với các tham số trong mảng NumPy.
Output: Hiển thị chuỗi kí tự ở dạng LaTeX.

Mô tả: Hàm latex_text có hai đầu vào là params và dict. Đầu vào params là một mảng numpy chứa các tham số của mô hình tuyến tính, trong khi đầu vào dict là một từ điển chứa các tên cột tương ứng với các tham số trong params. Hàm này trả về một chuỗi LaTeX biểu diễn phương trình tuyến tính của mô hình dựa trên các tham số và tên cột được cung cấp.

Trong hàm này, ta bắt đầu bằng cách khởi tạo chuỗi text với giá trị ban đầu là phần đầu của phương trình LaTeX: Hàm latex_text có hai đầu vào là params và dict. Đầu vào params là một mảng numpy chứa các tham số của mô hình tuyến tính, trong khi đầu vào dict là một từ điển chứa các tên cột tương ứng với các tham số trong params. Hàm này trả về một chuỗi LaTeX biểu diễn phương trình tuyến tính của mô hình dựa trên các tham số và tên cột được cung cấp.

Do LaTeX trên Jupyter Notebook không tự xuống dòng khi độ dài của một chuỗi vượt quá chiều rộng của trang, ta sẽ thêm đoạn mã xuống dòng mỗi 4 tên cột được hiển thị (không tính cột đầu tiên).

2.2.6 Hàm kfold_cross_model

Input: Mảng NumPy chứa các giá trị của đặc trưng, mảng NumPy chứa các giá trị của biến mục tiêu, đối tượng KFold, mảng để lưu trữ các giá trị MAE.
Output: Không có.

Mô tả: Trong hàm này, ta sử dụng vòng lặp để duyệt qua các phần chia của dữ liệu huấn luyện khi thực hiện phương pháp kfold cross validation. Mỗi phần tử của iterator là một tuple chứa hai mảng chỉ số: *train_index* và *test_index*. Mảng *train_index* chứa các chỉ số của phần huấn luyện, trong khi mảng *test_index* chứa các chỉ số của phần kiểm tra1.

Vòng lặp *for train_index, test_index in kf.split(X_train_np)* được sử dụng để duyệt qua các phần chia của dữ liệu huấn luyện. Đối với mỗi phần chia, ta sử dụng chỉ số của phần huấn luyện và phần kiểm tra để trích xuất dữ liệu tương ứng từ X_train_np và y_train_np. Sau đó, ta có thể huấn luyện mô hình với dữ liệu huấn luyện đã trích xuất và đánh giá hiệu suất của mô hình trên dữ liệu kiểm tra. Cuối cùng, ta thêm giá trị MAE vào danh sách mae_arr để lưu trữ kết quả. Nếu X_train_np là một vector 1 chiều, ta thêm một chiều mới vào cuối bằng cách sử dụng cú pháp [:, None]. Ngược lại, ta chỉ cần trích xuất dữ liệu bình thường. Sau đó, ta huấn luyện một mô hình hồi quy tuyến tính bằng cách sử dụng phương thức fit của class OLSLinearRegression với dữ liệu huấn luyện đã trích xuất. Ta sử dụng mô hình đã huấn luyện để dự đoán giá trị đầu ra cho phần

kiểm tra của dữ liệu và tính toán giá trị MAE giữa giá trị thực tế và giá trị dự đoán. Cuối cùng, ta thêm giá trị MAE vào danh sách mae_arr để lưu trữ kết quả.

Hàm này không có giá trị trả về, nhưng nó thay đổi nội dung của danh sách mae_arr bằng cách thêm các giá trị MAE tính toán được trong quá trình kiểm định chéo.

2.2.7 Hàm scientific_notation_converter

Input: Mảng chứa các trọng số của mô hình tuyến tính.
Output: Mảng chứa các trọng số của mô hình tuyến tính ở dạng ký hiệu khoa học với 3 chữ số thập phân.

Mô tả: Trong đồ án yêu cầu ta phải trình bày mô hình dự đoán mức lương với tham số được làm tròn 3 chữ số thập phân. Ta có thể sử dụng hàm *round* trong phương thức *get_params*. Song nếu làm tròn ngay từ bước huấn luyện, ta sẽ mất đi độ chính xác của mô hình. Do đó, ta sẽ sử dụng hàm *scientific_notation_converter* để chuyển đổi các trọng số của mô hình.

Trong hàm này, ta sử dụng kỹ thuật list comprehension để duyệt qua các phần tử trong params và chuyển đổi chúng thành dạng ký hiệu khoa học (scientific notation). Đối với mỗi phần tử *param* trong params, sau đó định dạng nó thành chuỗi ký hiệu khoa học với 3 chữ số thập phân bằng cách sử dụng phương thức format: *"%.3f"*. Kết quả cuối cùng là một danh sách mới chứa các chuỗi ký hiệu khoa học tương ứng với các phần tử trong params.

3 Đánh giá kết quả mô hình

3.1 Thông tin cấu hình

- Bộ xử lý: AMD Ryzen 7 5800H
- Hệ điều hành: Windows 10 Pro 64-bit
- Phiên bản Python: 3.10.9

3.2 Ghi chú

Trước khi thực hiện tìm mô hình hồi quy tuyến tính cho tất cả yêu cầu đề bài, ta cần đọc dữ liệu từ *train.csv* và *test.csv* ứng với tập dữ liệu huấn luyện (**train**) và tập dữ liệu kiểm tra (**test**), đồng thời xử lý DataFrame vừa đọc được để nhận biết đâu là tập các đặc trưng (**X**) và đâu là tập biến mục tiêu (**y**). Sau đó, ta sẽ chia tập dữ liệu huấn luyện thành hai phần: tập huấn luyện (**X_train**, **y_train**) và tập kiểm tra (**X_test**, **y_test**). Do tập mục tiêu là cố định (Salary ứng với **y_train** và **y_test**), nên ta dùng chung hai tập này cho các câu 1a, 1b, 1c, 1d.

Sở dĩ phải chuyển về NumPy array với tất cả DataFrame vừa đọc được là vì khi thực hiện yêu cầu tìm đặc trưng tốt nhất trong số các đặc trưng (yêu cầu 1b, 1c), ma trận đặc trưng X chỉ có 1 cột, nên khi thực hiện các phép toán trên ma trận, ta cần mở rộng 1 chiều mới có thể thực hiện phép nhân (phương thức *fit* của lớp OLSLinearRegression). Nếu không mở rộng 1 chiều mới, ta sẽ gặp lỗi khi thực hiện phép nhân.

Do đồ án yêu cầu khi sử dụng cross-validation, ta chỉ được phép xáo trộn dữ liệu 1 lần duy nhất và thực hiện trên m mô hình [10], ta sẽ gọi lại đối tượng KFold đã được khởi tạo trước đó để sử dụng lại các chỉ số đã được tạo ra. Điều này giúp đảm bảo rằng các chỉ số được tạo ra sẽ giống nhau trong mỗi lần chạy, từ đó đảm bảo tính nhất quán của kết quả.

3.3 Yêu cầu 1a

3.3.1 Các bước thực hiện

1. Thực hiện lấy 11 đặc trưng đầu tiên đề bài cung cấp.
2. Thực hiện chia tập dữ liệu huấn luyện thành hai phần: tập huấn luyện (**X_1a_train**, **y_train**) và tập kiểm tra (**X_test**, **y_test**).
3. Huấn luyện mô hình hồi quy tuyến tính với tập huấn luyện (**X_1a_train**, **y_train**) thu được tập mục tiêu dự đoán (**y_pred_1a**).
4. Gọi hàm **mae** để tính giá trị lỗi trung bình tuyệt đối (MAE) giữa tập mục tiêu thực tế (**y_test**) và tập mục tiêu dự đoán (**y_pred_1a**).

3.3.2 Công thức hồi quy

$$\text{Salary} = -22756.513 \times \text{Gender} + 804.503 \times \text{10percentage} + 1294.655 \times \text{12percentage} - 91781.898 \times \text{CollegeTier} + 23182.389 \times \text{Degree} + 1437.549 \times \text{collegeGPA} - 8570.662 \times \text{CollegeCityTier} + 147.858 \times \text{English} + 152.888 \times \text{Logical} + 117.222 \times \text{Quant} + 34552.286 \times \text{Domain}$$

3.3.3 Kết quả mô hình

MAE = 104863.77754033195

3.3.4 Nhận xét

Trong một mô hình hồi quy tuyến tính, việc thêm nhiều thuộc tính có thể giúp giảm sai số trung bình tuyệt đối (MAE) bởi vì nó cho phép mô hình nắm bắt được nhiều thông tin hơn về mối quan hệ giữa các biến độc lập và biến phụ thuộc. Tuy nhiên, điều này không phải lúc nào cũng đúng. Việc thêm quá nhiều thuộc tính có thể dẫn đến hiện tượng quá khớp (overfitting), khi đó

mô hình sẽ hoạt động tốt trên dữ liệu huấn luyện nhưng không hoạt động tốt trên dữ liệu kiểm tra. Việc này giống như để mô hình "học tủ" các quan hệ giữa các biến độc lập và biến phụ thuộc trong dữ liệu huấn luyện, khiến cho mô hình không thể áp dụng được cho các dữ liệu mới. Do đó, cách làm này làm tăng sai số trung bình tuyệt đối (MAE) trên dữ liệu kiểm tra.

3.4 Yêu cầu 1b

3.4.1 Các bước thực hiện

1. Thực hiện lấy 5 đặc trưng tính cách yêu cầu
2. Đặt giá trị k (trong k-fold cross validation) là 20.
3. Gọi đối tượng thuộc lớp KFold trong thư viện scikit-learn nhằm cung cấp các chỉ mục để chia dữ liệu thành các tập huấn luyện và kiểm tra.
4. Tạo mảng chứa các giá trị MAE ứng với mỗi đặc trưng.
5. Lần lượt duyệt qua các đặc trưng, với mỗi đặc trưng, ta thực hiện các bước sau:
   (a) Gọi hàm **kfold_cross_model** 2.2.6 với các tham số là mảng đặc trưng tương ứng, tập mục tiêu y_train_np, đối tượng KFold, mảng chứa các giá trị MAE.
   (b) Tính giá trị trung bình của các giá trị MAE thu được từ k-fold cross validation.
   (c) Thêm giá trị trung bình của các giá trị MAE thu được vào mảng chứa các giá trị MAE.
6. Tìm ra đặc trưng tốt nhất bằng cách tìm ra giá trị MAE nhỏ nhất trong mảng chứa các giá trị MAE, ở đây đặc trưng tốt nhất là *nueroticism*.
7. Với đặc trưng tốt nhất vừa tìm được, ta thực hiện các bước sau:
   (a) Thêm chiều mới vào cuối mảng đặc trưng bằng cách sử dụng cú pháp [:, None].
   (b) Huấn luyện mô hình hồi quy tuyến tính với tập huấn luyện (**X_1b_best_train**, **y_train_np**) thu được tập mục tiêu dự đoán (**y_pred**). Với **X_1b_best_train** là mảng đặc trưng tốt nhất vừa tìm được, **y_train_np** là tập mục tiêu huấn luyện.
   (c) Đánh giá mô hình với tập kiểm tra (**X_1b_best_test**, **y_test**) thu được tập mục tiêu dự đoán (**y_pred**).
   (d) Gọi hàm **mae** để tính giá trị lỗi trung bình tuyệt đối (MAE) giữa tập mục tiêu thực tế (**y_test**) và tập mục tiêu dự đoán (**y_pred**).

3.4.2 Công thức hồi quy

$$\text{Salary} = -56546.304 \times \text{nueroticism}$$

3.4.3 Kết quả mô hình

STT	Mô hình với 1 đặc trưng	MAE
1	conscientiousness	306221.75562170806
2	agreeableness	300857.53034685535
3	extraversion	307059.3991665598
4	neuroticism	299376.1964789984
5	openness_to_experience	303054.52630103764

Kết quả MAE của mô hình với 1 đặc trưng tốt nhất là **291019.693226953**

3.4.4 Nhận xét

- Các tính cách này là một phần của mô hình Big Five, một khung tham chiếu phổ biến để đánh giá nhân cách của con người. Các tính cách này được đo bằng các điểm số trong bài kiểm tra AMCAT, một cổng việc làm trực tuyến. [5]
- Các tính cách này có thể ảnh hưởng đến mức lương và công việc của các kỹ sư tốt nghiệp ở Ấn Độ. Ví dụ, neuroticism có thể liên quan đến khả năng chịu áp lực và thích nghi với môi trường làm việc. Agreeableness có thể liên quan đến khả năng hợp tác và giao tiếp với đồng nghiệp và khách hàng. Openness_to_experience có thể liên quan đến khả năng sáng tạo và học hỏi những điều mới. Conscientiousness có thể liên quan đến khả năng tổ chức và quản lý thời gian. Extraversion có thể liên quan đến khả năng giao tiếp và thuyết phục.
- Để ý rằng MAE của agreeableness và neuroticism không chênh lệch nhau quá nhiều.
  Theo một số nghiên cứu, các đặc trưng tính cách có thể ảnh hưởng đến mức lương và sự thăng tiến trong công việc. Tuy nhiên, mức độ ảnh hưởng của các đặc trưng tính cách có thể khác nhau tùy thuộc vào ngành nghề và công việc cụ thể. [1] Một báo cáo được công bố bởi trang web tìm kiếm việc làm Joblist cho thấy những người có tính cách tự phụ (conscientiousness) có xu hướng kiếm được ít nhất 75.000 đô la Mỹ mỗi năm, trong khi những người có tính cách neuroticism có xu hướng kiếm được 34.999 đô la Mỹ hoặc ít hơn mỗi năm. Tuy nhiên, báo cáo này không đề cập đến mối quan hệ giữa tính cách agreeableness và mức lương. [23]
- Mức lương của một người có thể phụ thuộc vào nhiều yếu tố khác nhau, chẳng hạn như nghề nghiệp, kinh nghiệm làm việc, nơi làm việc, v.v. Do đó, việc dự đoán mức lương của một người dựa trên một đặc trưng tính cách là không chính xác. Do đó, chỉ số MAE của mô hình tốt nhất **best_personality_feature_model** dù là tốt nhất nhưng vẫn có sai số khá lớn.

3.4.5 Giả thuyết

Một nghiên cứu được công bố vào năm 2001 trong Tạp chí Hành vi Nghề nghiệp đã khảo sát các chiều tính cách “Big Five”: neuroticism, conscientiousness, extroversion, agreeableness và openness để hiểu mối quan hệ của chúng với kết quả nghề nghiệp. Các nhà nghiên cứu tại Đại

học Cleveland State đã khảo sát 496 nhân viên (318 nam và 178 nữ). Kết quả của một phân tích thống kê cho thấy nhân viên hướng ngoại có sự hài lòng lớn hơn với lương, thăng tiến và sự hài lòng về sự nghiệp tổng thể; trong khi những người có điểm cao về neuroticism (ví dụ, tính khí thất thường, lo lắng, lo âu, sợ hãi hoặc bực bội) ít có khả năng hài lòng với sự nghiệp của họ.

Những người có điểm cao về agreeableness có ít sự hài lòng về sự nghiệp, và điểm cao về openness có mối quan hệ tiêu cực với mức lương. Nhóm nghiên cứu phát hiện ra một mối quan hệ tiêu cực đáng kể giữa agreeableness và mức lương đối với những người trong các nghề nghiệp liên quan đến con người nhưng không có mối quan hệ đối với những người trong các nghề nghiệp không liên quan đến yếu tố "con người" mạnh.

Một nghiên cứu thứ hai được công bố trong số tháng 12 năm 2020 của Tạp chí Khoa học Tâm lý cho thấy nếu ta thay đổi tính cách, nó có thể dẫn đến mức độ thành công cao hơn trong công việc. Tiến sĩ Kevin Hoff tại Đại học Houston và nhóm nghiên cứu của ông đã theo dõi hai nhóm thanh niên trong khoảng 12 năm từ 17 tuổi đến khoảng 29 tuổi. Sự phát triển về ổn định cảm xúc, tự phụ và hướng ngoại là các đặc trưng tính cách dự đoán sự hài lòng và thành công trong sự nghiệp.

Những người có mức độ tự phụ và ổn định cảm xúc cao hơn cho thấy sự thành công nhiều hơn trong sự nghiệp tổng thể. Đây là nghiên cứu đầu tiên cho thấy sức mạnh dự đoán của sự thay đổi tính cách cho một loạt các nghề nghiệp trong hơn một thập kỷ. Nói chung, các kết quả cho thấy tính cách có tác động quan trọng đến kết quả sự nghiệp sớm - cả thông qua các mức độ đặc trưng bền vững và cách mà con người thay đổi theo thời gian. Tin tốt là những kết quả này cho thấy các đặc trưng tính cách có thể được điều chỉnh theo thời gian và hỗ trợ khoa học thần kinh rằng não bộ có tính linh hoạt. [2]

Tóm lại, neuroticism đạt kết quả tốt nhất vì nó có mối quan hệ tiêu cực với mức lương. Các đặc trưng tính cách khác có thể ảnh hưởng đến mức lương và sự thăng tiến trong công việc, nhưng mức độ ảnh hưởng này có thể khác nhau tùy thuộc vào ngành nghề và công việc cụ thể.

3.5 Yêu cầu 1c

3.5.1 Các bước thực hiện

1. Thực hiện lấy 3 đặc trưng tính cách yêu cầu
2. Đặt giá trị k (trong k-fold cross validation) là 20.
3. Gọi đối tượng thuộc lớp KFold trong thư viện scikit-learn nhằm cung cấp các chỉ mục để chia dữ liệu thành các tập huấn luyện và kiểm tra.
4. Tạo mảng chứa các giá trị MAE ứng với mỗi đặc trưng.
5. Lần lượt duyệt qua các đặc trưng, với mỗi đặc trưng, ta thực hiện các bước sau:

(a) Gọi hàm **kfold_cross_model** 2.2.6 với các tham số là mảng đặc trưng tương ứng, tập mục tiêu y_train_np, đối tượng KFold, mảng chứa các giá trị MAE.

(b) Tính giá trị trung bình của các giá trị MAE thu được từ k-fold cross validation.

(c) Thêm giá trị trung bình của các giá trị MAE thu được vào mảng chứa các giá trị MAE.

6. Tìm ra đặc trưng tốt nhất bằng cách tìm ra giá trị MAE nhỏ nhất trong mảng chứa các giá trị MAE, ở đây đặc trưng tốt nhất là *Quant*.

7. Với đặc trưng tốt nhất vừa tìm được, ta thực hiện các bước sau:

(a) Thêm chiều mới vào cuối mảng đặc trưng bằng cách sử dụng cú pháp [:, None].

(b) Huấn luyện mô hình hồi quy tuyến tính với tập huấn luyện (**X_1c_best_train**, **y_train_np**) thu được tập mục tiêu dự đoán (**y_pred**). Với **X_1c_best_train** là mảng đặc trưng tốt nhất vừa tìm được, **y_train_np** là tập mục tiêu huấn luyện.

(c) Đánh giá mô hình với tập kiểm tra (**X_1c_best_test**, **y_test**) thu được tập mục tiêu dự đoán (**y_pred**).

(d) Gọi hàm **mae** để tính giá trị lỗi trung bình tuyệt đối (MAE) giữa tập mục tiêu thực tế (**y_test**) và tập mục tiêu dự đoán (**y_pred**).

3.5.2 Công thức hồi quy

$$\text{Salary} = 585.895 \times \text{Quant}$$

3.5.3 Kết quả mô hình

STT	Mô hình với 1 đặc trưng	MAE
1	English	121901.64167007094
2	Logical	120304.93112741373
3	Quant	118126.44763193061

Kết quả MAE của mô hình với 1 đặc trưng tốt nhất là **106819.5776198967**

3.5.4 Nhận xét

- Các kỹ năng này là một phần của bài kiểm tra AMCAT, một cổng việc làm trực tuyến. [5]
- Các kỹ năng này có thể ảnh hưởng đến mức lương và công việc của các kỹ sư tốt nghiệp ở Ấn Độ. Chẳng hạn, Logical có thể liên quan đến khả năng giải quyết vấn đề, Quant có thể liên quan đến khả năng tính toán, English có thể liên quan đến khả năng giao tiếp. 3 kỹ năng Tiếng Anh, Lý luận, Số học là một số phần trong bài kiểm tra AMCAT đo khả năng sử dụng ngôn ngữ, lý luận logic và khả năng số học của ứng viên. Những kỹ năng này quan trọng cho kỹ sư để giao tiếp hiệu quả, giải quyết vấn đề và phân tích dữ liệu.

- Mô hình dự đoán lương của sinh viên tốt nghiệp kỹ thuật với đặc trưng Quant có MAE tốt nhất so với các đặc trưng English và Logical, với MAE lần lượt là 121901.64167007094 và 120304.93112741373. Điều này cho thấy đặc trưng Quant có thể giải thích sự biến động trong lương của sinh viên tốt nghiệp kỹ thuật tốt hơn so với các đặc trưng English và Logical. Điều này cho thấy khả năng giải quyết định lượng (Quant) có thể có mối liên hệ mật thiết với mức lương của sinh viên tốt nghiệp kỹ thuật, trong khi khả năng giao tiếp bằng tiếng Anh (English) và khả năng suy luận logic (Logical) có thể không quan trọng bằng. Tuy nhiên, điều này chỉ là một giả thuyết và cần được kiểm chứng thêm bằng cách sử dụng dữ liệu và phân tích thêm.
- Ba giá trị MAE không chênh lệch nhiều cho thấy mỗi đặc trưng đều có một mức độ ảnh hưởng nhất định đến mức lương của sinh viên tốt nghiệp kỹ thuật. Điều này có thể cho thấy rằng cả ba kỹ năng đều quan trọng trong ngành kỹ sư và có thể cần được phát triển để đạt được thành công trong sự nghiệp.

3.5.5 Giả thuyết

Một giả thuyết có thể là bài kiểm tra định lượng (Quantitative Ability Test) có mối liên hệ mật thiết với khả năng thành công trong một số ngành kỹ thuật. Điều này có thể dẫn đến việc một mô hình dự đoán lương sử dụng kết quả bài kiểm tra định lượng làm đặc trưng có thể có độ chính xác cao hơn, được thể hiện qua giá trị MAE thấp hơn.

Hai kỹ năng còn lại là English và Logical cũng quan trọng không kém. Trong ngành kỹ sư, khả năng giải quyết vấn đề định lượng có mối liên hệ mật thiết hơn với mức lương so với khả năng giao tiếp bằng tiếng Anh. Điều này có thể dẫn đến việc một mô hình dự đoán lương sử dụng kết quả bài kiểm tra định lượng làm đặc trưng có thể có độ chính xác cao hơn so với một mô hình sử dụng khả năng giao tiếp bằng tiếng Anh làm đặc trưng.

Ví dụ, trong các ngành kỹ thuật liên quan đến toán học và khoa học máy tính, khả năng giải quyết vấn đề định lượng rất quan trọng để đánh giá khả năng của một ứng viên. Do đó, một ứng viên có kết quả cao trong bài kiểm tra định lượng có thể được xem là tốt hơn trong ngành kỹ thuật này và do đó có thể được trả lương cao hơn. Trong khi đó, khả năng giao tiếp bằng tiếng Anh có thể không được coi là yếu tố then chốt để đánh giá khả năng thành công của một ứng viên trong các ngành kỹ thuật này.

Còn kỹ năng Logical không ảnh hưởng đến mức lương bằng Quant là vì trong một số ngành kỹ thuật, khả năng giải quyết vấn đề định lượng có thể được coi là một yếu tố quan trọng để đánh giá khả năng thành công của một ứng viên. Nói cách khác, với bộ dữ liệu nhận được, khả năng giải quyết vấn đề định lượng có thể có mối liên hệ mật thiết hơn với mức lương của sinh viên tốt nghiệp kỹ thuật so với khả năng suy luận logic. Một số nhóm ngành kỹ thuật có thể khiến cho Quant tốt hơn Logical là kỹ sư phần mềm, kỹ sư điện, kỹ sư cơ khí, kỹ sư hóa học, kỹ sư môi

trường . . .

3.6 Yêu cầu 1d

- **Mô hình 1 (AMCAT)**

3.6.1 Ý tưởng

AMCAT là một bài kiểm tra trực tuyến được sử dụng để đánh giá các kỹ năng cơ bản của một ứng viên. Bài kiểm tra này được sử dụng rộng rãi trong các công ty Ấn Độ để đánh giá các ứng viên cho các vị trí kỹ thuật. Bài kiểm tra này đo các kỹ năng cơ bản của ứng viên, chẳng hạn như kỹ năng giao tiếp bằng tiếng Anh, khả năng giải quyết vấn đề định lượng, khả năng suy luận logic, v.v. [5].

AMCAT quan trọng vì nó mang lại một bài kiểm tra năng lực công bằng, tiêu chuẩn và điểm số dễ so sánh để lọc ra danh sách các ứng viên có hiệu suất tốt nhất. Điều này tiết kiệm thời gian cho cả ứng viên và công ty và giúp ứng viên tìm thấy công việc phù hợp nhất cho họ trên khắp đất nước. AMCAT được hơn 1000 công ty sử dụng và được đăng ký dự thi bởi hơn 2 triệu sinh viên [8].

Vì vậy, mô hình dự đoán lương của sinh viên tốt nghiệp kỹ thuật sử dụng kết quả bài kiểm tra AMCAT có thể là một mô hình đáng tin cậy để thử nghiệm. Ta bỏ qua các đặc trưng tính cách do các đặc trưng tính cách đã được lựa chọn ở yêu cầu 1b.

3.6.2 Các bước thực hiện

1. Các đặc trưng có liên quan đến AMCAT được sử dụng trong mô hình là: ComputerProgramming, ElectronicsAndSemicon, ComputerScience, MechanicalEngg, ElectricalEngg, TelecomEngg, CivilEngg. Ta sẽ lấy các đặc trưng này từ tập dữ liệu huấn luyện (**X_AMCAT_train**) và mảng chứa các giá trị MAE.
2. Gọi hàm *kfold_cross_model* 2.2.6 với các tham số là mảng đặc trưng tương ứng, tập mục tiêu y_train_np, đối tượng KFold, mảng chứa các giá trị MAE. Với mỗi giá trị MAE thu được từ 1 fold, thêm vào mảng chứa các giá trị MAE.
3. Tính giá trị trung bình của các giá trị MAE thu được từ k-fold cross validation.

- **Mô hình 2**

3.6.3 Ý tưởng

Độ tương quan là một phương pháp thống kê được sử dụng để đo lường mối quan hệ giữa hai hoặc nhiều biến. Nó có thể giúp ta xác định các biến có mối liên hệ mạnh với biến phụ

thuộc và do đó có thể được sử dụng làm đặc trưng cho mô hình dự đoán. Điều này có thể giúp cải thiện hiệu suất của mô hình và giúp ta đưa ra dự đoán chính xác hơn.

Một trong những lợi ích của việc sử dụng độ tương quan để xây dựng mô hình là nó cung cấp một bản tóm tắt ngắn gọn và rõ ràng về mối quan hệ giữa các biến. Điều này có thể giúp ta hiểu rõ hơn về dữ liệu và xây dựng các mô hình chính xác hơn. Ngoài ra, việc sử dụng độ tương quan cũng có thể giúp ta loại bỏ các biến không liên quan hoặc không cần thiết, giúp tối ưu hóa hiệu suất của mô hình.

Tuy nhiên, độ tương quan không phải là phương pháp duy nhất để xây dựng mô hình và có thể không phù hợp trong mọi trường hợp. Các phương pháp khác như lựa chọn đặc trưng, phân tích thành phần chính và lựa chọn mô hình cũng có thể được sử dụng để xây dựng các mô hình chính xác. Việc lựa chọn phương pháp phù hợp sẽ phụ thuộc vào nhiều yếu tố khác nhau, bao gồm loại dữ liệu, số lượng biến và yêu cầu của bài toán [7] [9].

Trong khoa học dữ liệu, độ tương quan càng lớn có thể cho thấy sự quá khớp (overfitting) của mô hình huấn luyện, dẫn đến khi đưa mô hình vào tập kiểm tra sẽ có sai số lớn. Với cách tạo heatmap đã được trình bày, ta nhận thấy max{corr} = 0.64. Và ta sẽ tiến hành loại bỏ những đặc trưng có độ tương quan > 0.6 để xây dựng mô hình.

Trong bài toán này, ta sẽ sử dụng độ tương quan để xây dựng mô hình dự đoán lương của sinh viên tốt nghiệp kỹ thuật. Ta sẽ lựa chọn các đặc trưng có độ tương quan thấp nhất với nhau để xây dựng mô hình. Ta sẽ sử dụng hàm **corr** trong thư viện pandas để tính độ tương quan giữa các đặc trưng. Ta sẽ lựa chọn các đặc trưng có độ tương quan thấp nhất với nhau để xây dựng mô hình.

3.6.4 Các bước thực hiện

1. Tạo ma trận tương quan: *corr_matrix* bằng cách sử dụng phương thức corr() của X_train để tính toán ma trận tương quan giữa các cột của X_train. Phương thức abs() được sử dụng để lấy giá trị tuyệt đối của các hệ số tương quan và vẽ heatmap.
2. Lấy ma trận tam giác trên với các giá trị 1 ở trên đường chéo chính và các giá trị 0 ở dưới đường chéo chính. Phương thức astype(bool) sau đó được sử dụng để chuyển đổi các giá trị 1 thành True và các giá trị 0 thành False.

   Kết quả là một mảng bool với các giá trị True ở trên đường chéo chính và các giá trị False ở dưới. Mảng bool này sau đó được sử dụng như là tham số cho phương thức where() của *corr_matrix*. Phương thức where() lọc ra các giá trị của DataFrame tại các vị trí mà mảng bool có giá trị True và đặt các giá trị còn lại thành NaN. Kết quả là DataFrame upper chỉ chứa các giá trị của tam giác trên của ma trận tương quan.
3. Chọn các đặc trưng có độ tương quan lớn hơn 0.6 sử dụng list comprehension để lặp qua các cột của DataFrame upper và chọn ra các cột có bất kỳ giá trị nào lớn hơn 0.6. Các cột này được lưu vào danh sách to_drop.

4. Gọi hàm *kfold_cross_model* 2.2.6 với các tham số là mảng đặc trưng tương ứng, tập mục tiêu y_train_np, đối tượng KFold, mảng chứa các giá trị MAE. Với mỗi giá trị MAE thu được từ 1 fold, thêm vào mảng chứa các giá trị MAE.
5. Tính giá trị trung bình của các giá trị MAE thu được từ k-fold cross validation.

- **Mô hình 3**

3.6.5 Ý tưởng

Một ý tưởng để huấn luyện mô hình là sử dụng toàn bộ các cột có sẵn trong dữ liệu huấn luyện. Lợi ích của việc sử dụng toàn bộ các cột để huấn luyện mô hình là nó đơn giản và nhanh chóng. Việc sử dụng toàn bộ các cột có thể giúp mô hình học được những mối quan hệ phức tạp giữa các đặc trưng và biến mục tiêu.

Tuy nhiên, việc này cũng có thể có những hạn chế. Một số đặc trưng có thể không liên quan hoặc không cần thiết cho việc dự đoán biến mục tiêu, và việc sử dụng chúng có thể làm giảm hiệu suất của mô hình. Ngoài ra, việc sử dụng nhiều đặc trưng có thể làm tăng độ phức tạp của mô hình và khiến nó khó hiểu và khó diễn giải hơn.

Đồng thời, ta sẽ không thể loại bỏ những đặc trưng không liên quan/ không cần thiết cho việc dự đoán biến mục tiêu. Dẫn đến có thể làm cho mô hình được huấn luyện không tối ưu.

Ở mô hình 3, ta sẽ loại bỏ đặc trưng CollegeCityTier và xử lý các giá trị thiếu của các đặc trưng còn lại như sau:

- $\text{Grading} = 5\%10\text{percentage} + 10\%12\text{percentage} + 85\%(\text{collegeGPA} \times \text{CollegeTier}^2 \times \text{Degree}^2)$
- $\text{AMCAT Point} = 2(\text{English} + \text{Logical} + \text{Quant})$
  $+ \text{Domain}(\text{ComputerProgramming} + \text{ElectronicsAndSemicon} + \text{ComputerScience}$
  $+ \text{MechanicalEngg} + \text{ElectricalEngg} + \text{TelecomEngg} + \text{CivilEngg}) +$
  $(\text{conscientiousness} + \text{agreeableness} + \text{extraversion} + \text{nueroticism} + \text{openess\_to\_experience})^2$

Các tham số được truyền vào với nhiều nguyên nhân. Đầu tiên, điểm số lớp 10 và lớp 12 không ảnh hưởng quá nhiều đến mức lương khi thực tế rằng rất nhiều sinh viên chưa có định hướng và đầu tư cho tương lai nghề nghiệp từ những năm trung học phổ thông. Do đó điểm lớp 10 chiếm 5% và điểm lớp 12 chiếm 10% trong tổng điểm. Điểm số của sinh viên trong trường đại học được tính bằng công thức có thêm hệ số CollegeTier và Degree. Cấp độ của trường đại học có thể ảnh hưởng đến mức lương của sinh viên tốt nghiệp kỹ thuật. Điều này có thể là do các trường đại học hàng đầu có chất lượng giảng dạy tốt hơn và có thể đào tạo ra những kỹ sư tốt hơn và do chúng chỉ có giá trị 1 hoặc 2 nên ta bình phương để tạo ra sự khác biệt rõ ràng hơn. Degree cũng có thể ảnh hưởng đến mức lương của sinh viên tốt nghiệp kỹ thuật. Ứng viên có bằng thạc sĩ hoặc tiến sĩ có thể được trả lương cao hơn so với ứng viên có bằng cử nhân. Do đó, ta cũng thực hiện bình phương.

Đối với bài kiểm tra AMCAT, ba phần kỹ năng chính và tất cả sinh viên dự thi đều tham gia (tất cả giá trị khác -1), ta nhân điểm của từng bài thi thành phần cho 2 vừa để tăng độ quan trọng của bài thi thành phần. Với các điểm số các phần kiểm tra tiếp theo, có thể có sinh viên không tham gia và đặt giá trị -1, ta quy đổi về 0 để không ảnh hưởng đến tổng điểm. Và cuối cùng nhân tổng điểm tính được với điểm số Domain.

Đối với các đặc trưng tính cách, ta thực hiện bình phương để tạo ra sự khác biệt rõ ràng hơn cũng như loại bỏ các giá trị âm. Và cuối cùng cộng tổng điểm tính được với điểm số của các đặc trưng tính cách.

3.6.6 Các bước thực hiện

1.

3.6.7 Kết quả mô hình

STT	Mô hình tìm được	MAE
1	Sử dụng 7 đặc trưng (ComputerProgramming, ElectronicsAndSemicon, ComputerScience, MechanicalEngg, ElectricalEngg, TelecomEngg, CivilEngg)	135972.48928102275
2	Sử dụng 21 đặc trưng ít tương quan nhất (Least correlated features), loại bỏ những đặc trưng có tương quan > 0.6	110608.51614315098
3	Sử dụng 22 đặc trưng (tất cả trừ College City Tier, có thay đổi về điểm số đánh giá và điểm bài kiểm tra AMCAT)	115699.55196244389

3.6.8 Nhận xét mô hình tốt nhất

3.6.9 Giả thuyết mô hình tốt nhất

4 Tài liệu tham khảo

[1] University of Arkansas. *Personality traits predict performance differently across different jobs — sciencedaily.com*. https://www.sciencedaily.com/releases/2021/12/211213181545.htm. [Accessed 23-08-2023]. 2021.

[2] Ph.D. Bryan Robinson. *Scientists Discover The Link Between Your Personality And Degree Of Career Success — forbes.com*. https://www.forbes.com/sites/bryanrobinson/2020/12/05/scientists-discover-the-link-between-your-personality-and-degree-of-career-success/?sh=723511ac7c4c. [Accessed 23-08-2023].

[3] *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].

[4] *Choice of K in K-fold cross-validation — stats.stackexchange.com*. https://stats.stackexchange.com/questions/27730/choice-of-k-in-k-fold-cross-validation. [Accessed 16-08-2023].

[5] *Engineering Graduate Salary Prediction — kaggle.com*. https://www.kaggle.com/datasets/manishkc06/engineering-graduate-salary-prediction. [Accessed 23-08-2023].

[6] *How to Implement K fold Cross-Validation in Scikit-Learn — section.io*. https://www.section.io/engineering-education/how-to-implement-k-fold-cross-validation/. [Accessed 11-08-2023].

[7] *How to Use Correlation to Make Predictions — hbr.org*. https://hbr.org/2022/04/how-to-use-correlation-to-make-predictions. [Accessed 16-08-2023].

[8] *Is AMCAT Useful for Freshers? - Getmyuni — getmyuni.com*. https://www.getmyuni.com/articles/how-useful-is-amcat. [Accessed 16-08-2023].

[9] *Is correlation needed when building a model? — datascience.stackexchange.com*. https://datascience.stackexchange.com/questions/29051/is-correlation-needed-when-building-a-model. [Accessed 16-08-2023].

[10] *Learning Management System - fit@hcmus — courses.ctda.hcmus.edu.vn*. https://courses.ctda.hcmus.edu.vn/pluginfile.php/108900/mod_folder/content/0/MSSV.ipynb?forcedownload=1. [Accessed 16-08-2023].

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

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

[13] *numpy.ones; NumPy v1.25 Manual — numpy.org*. https://numpy.org/doc/stable/reference/generated/numpy.ones.html. [Accessed 19-08-2023].

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

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

[16] *numpy.triu; NumPy v1.25 Manual — numpy.org*. https://numpy.org/doc/stable/reference/generated/numpy.triu.html. [Accessed 19-08-2023].

[17] *numpy.where; NumPy v1.25 Manual — numpy.org*. https://numpy.org/doc/stable/reference/generated/numpy.where.html. [Accessed 19-08-2023].

[18] *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].

[19] *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].

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

[21] *Random seed - Wikipedia — en.wikipedia.org*. https://en.wikipedia.org/wiki/Random_seed. [Accessed 16-08-2023].

[22] *Random state (Pseudo-random number) in Scikit learn — stackoverflow.com*. https://stackoverflow.com/questions/28064634/random-state-pseudo-random-number-in-scikit-learn. [Accessed 16-08-2023].

[23] Jade Scipioni. *These 2 personality traits can help determine whether you get a promotion or high salary — cnbc.com*. https://www.cnbc.com/2022/04/24/report-personality-traits-correlate-with-promotions-high-salaries.html. [Accessed 23-08-2023]. 2022.

[24] *seaborn.heatmap; seaborn 0.12.2 documentation — seaborn.pydata.org*. https://seaborn.pydata.org/generated/seaborn.heatmap.html. [Accessed 19-08-2023].

[25] *sklearn.model_selection.KFold — scikit-learn.org*. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html. [Accessed 11-08-2023].

[26] Isheunesu Tembo. *Cross-Validation Using K-Fold With Scikit-Learn — isheunesu48.medium.com*. https://isheunesu48.medium.com/cross-validation-using-k-fold-with-scikit-learn-cfc44bf1ce6. [Accessed 11-08-2023].