ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐHQG HCM
KHOA CÔNG NGHỆ THÔNG TIN

Toán Ứng Dụng Và Thống Kê

Cho Công Nghệ Thông Tin

Báo Cáo

Đồ Án 3: Linear Regression

Giảng viên:
Vũ Quốc Hoàng

Trợ giảng:
Phan Thị Phương Uyên
Lê Thanh Tùng
Nguyễn Văn Quang Huy

Sinh viên:

21127657 - Nguyễn Khánh Nhân

TPHCM, tháng 8 năm 2023

Mục Lục

I. Thông tin sinh viên

Ho tên: Nguyễn Khánh Nhân
MSSV: 21127657
Lớp: 21CLC02
Email: nknhan21@clc.fitus.edu.vn

II. Thông tin đồ án

Mục tiêu của đồ án là tìm hiểu các yếu tố quyết định mức lương và việc làm của các kỹ sư ngay sau khi tốt nghiệp. Các yếu tố như điểm số ở các cấp/trường đại học, kỹ năng của ứng viên, sự liên kết giữa trường đại học và các khu công nghiệp/công ty công nghệ, bằng cấp của sinh viên và điều kiện thị trường cho các ngành công nghiệp cụ thể sẽ ảnh hưởng đến điều này.

Bộ dữ liệu được sử dụng trong đồ án này thu thập tại Ấn Độ, nơi có hơn 6000 cơ sở đào tạo kỹ thuật công nghệ với khoảng 2,9 triệu sinh viên đang học tập. Mỗi năm, trung bình có 1,5 triệu sinh viên tốt nghiệp chuyên ngành Công nghệ/Kỹ thuật, tuy nhiên do thiếu kỹ năng cần thiết, ít hơn 20% trong số họ có việc làm phù hợp với chuyên môn của mình. Bộ dữ liệu này không chỉ giúp xây dựng công cụ dự đoán mức lương mà còn cung cấp thông tin về các yếu tố ảnh hưởng đến mức lương và chức danh công việc trên thị trường lao động. Sinh viên sẽ được khám phá những thông tin này trong phạm vi đồ án.

Trong đồ án này, ta sẽ thực hiện xây dựng mô hình hồi quy tuyến tính để dự đoán mức lương của các kỹ sư. Không chỉ đánh giá các mô hình có sẵn mà ta còn phải tìm cách/phương pháp xây dựng cho mình một mô hình riêng sao cho việc dự đoán mức lương chính xác với thực tế nhất có thể

III. Chi tiết mã nguồn của chương trình

1. Thư viện sử dụng

- **Numpy:** là một thư viện Python mạnh mẽ cho tính toán khoa học và toán học. Nó cung cấp một cấu trúc dữ liệu mảng đa chiều (ndarray) hiệu quả và các hàm toán học để thực hiện các phép toán trên mảng đó. Mô hình tuyến tính thường yêu cầu xử lý và tính toán trên mảng dữ liệu đầu vào. NumPy cung cấp các công cụ mạnh mẽ để thực hiện các phép toán ma trận, tính toán vector hóa và xử lý nhanh chóng các mảng số. Điều này giúp tăng tốc độ tính toán và hiệu suất của mô hình tuyến tính.
- **Pandas:** là một thư viện dựa trên NumPy, cung cấp cấu trúc dữ liệu và công cụ phân tích dữ liệu mạnh mẽ. Pandas giúp xử lý và làm việc với dữ liệu dạng bảng (như dữ liệu từ file CSV) dễ dàng. Nó cung cấp đối tượng DataFrame, cho phép bạn thực hiện các thao tác như lọc dữ liệu, xử lý dữ liệu thiếu, ghép nối dữ liệu từ nhiều nguồn, và truy vấn dữ liệu theo các điều kiện. Khi xây dựng mô hình tuyến tính, Pandas thường được sử dụng để nạp dữ liệu, thực hiện các phép biến đổi dữ liệu và chuẩn bị dữ liệu cho mô hình.
- **Statsmodels [1]:** là một thư viện Python được sử dụng rộng rãi cho các mô hình thống kê và hồi quy. Nó cung cấp các công cụ mạnh mẽ để thực hiện phân tích thống kê, kiểm định giả thuyết, mô hình hồi quy và các phương pháp khác liên quan đến thống kê. Ở câu **1D**, ta sẽ xây dựng mô hình có các **đặc trưng** nào dựa vào giá trị **p-value** của chúng (cụ thể sẽ được trình bày bên dưới), vì vậy thư viện **statsmodels** sẽ hỗ trợ ta tìm ra được **p-value**.

2. Class

- **OLSLinearRegression:** Class này sẽ cung cấp các phương thức để thực hiện việc xây dựng mô hình hồi quy tuyến tính, trả về hệ số của các đặc trưng có trong mô hình được xây dựng và cuối cùng dựa vào mô hình vừa xây dựng để thực hiện việc dự đoán mức lương.
- **k_folds_cross_validation:** Class sẽ hỗ trợ các phương thức để thực hiện kĩ thuật K-fold Cross Validation, một thuật toán hỗ trợ đánh giá xem mô hình nào sẽ tốt hơn.

* **Kĩ thuật k-fold cross validation**:
  - Thực hiện xáo trộn dữ liệu.
  - Chia dữ liệu thành k fold có kích thước bằng nhau.
  - Thực hiện xét qua từng fold, với mỗi fold sẽ thực hiện huấn luyện nó trên tất cả các mô hình để tính giá trị mae tương ứng.
  - Tính giá trị **mae** trung bình của mỗi mô hình sau khi được huấn luyện qua k fold.
  - Dựa vào giá trị **mae** trung bình để tìm ra mô hình tốt nhất. Mô hình tốt nhất là mô hình có giá trị **mae** trung bình nhỏ nhất.

3. Hàm và phương thức của các class

- Phương thức fit() của class OLSLinearRegression
  - **INPUT:** mảng X chứa các giá trị đặc trưng (**feature**), mảng y chứa các giá trị mục tiêu (**target**).
  - **OUTPUT:** trả về chính đối tượng mô hình, bao gồm mảng chứa các trọng số của từng **feature** trong mô hình.

- **Chi tiết:** mô hình tính toán ma trận ngược đảo của X nhân với chính nó chuyển vị (X.T @ X), sau đó nhân với chuyển vị của X (X.T). Kết quả cuối cùng là ma trận **pseudo-inverse** của X (hoặc còn gọi là **ma trận giả nghịch đảo**). Tiếp theo, mô hình tính toán vector trọng số self.w bằng cách nhân ma trận giả nghịch đảo này với mảng y.

- Phương thức get_params() của class OLSLinearRegression
  - **INPUT:** không có.
  - **OUTPUT:** trả về mảng chứa các trọng số (self.w) của từng feature trong mô hình được huấn luyện.
- Phương thức predict() của class OLSLinearRegression
  - **INPUT:** mảng X chứa các giá trị đặc trưng (**feature**).
  - **OUTPUT:** Trả về mảng y chứa các giá trị dự đoán.
- Phương thức __init__() của class k_folds_cross_validation
  - **INPUT:** số lượng fold k, các đặc trưng sẽ sử dụng trong những mô hình được đem đi so sánh features, các mô hình models.
  - **OUTPUT:** không có.
  - **Chi tiết:** Thực hiện gán các input vào các thuộc tính của class.
- Phương thức shuffle_data() của class k_folds_cross_validation
  - **INPUT:** Không có.
  - **OUTPUT:** Không có.
  - **Chi tiết:** Cố định seed trước khi shuffle để dễ cho việc kiểm tra. Sau đó chuyển dữ liệu từ kiểu DataFrame sang array để thực hiện xáo trộn dữ liệu bằng hàm numpy.random.shuffle(). Chuyển lại dữ liệu đã được xáo trộn từ array về

DataFrame.

- Phương thức split_to_k_folds() của class k_folds_cross_validation
  - **INPUT:** Không có.
  - **OUTPUT:** Không có.
  - **Chi tiết:** Sử dụng kĩ thuật list comprehension chia dữ liệu thành k fold sao cho mỗi fold có kích thước như nhau và lưu các fold vào 1 list self.folds.
- Phương thức cross_validation() của class k_folds_cross_validation
  - **INPUT:** Không có.
  - **OUTPUT:** Không có.
  - **Chi tiết:** Với mỗi fold, ta sẽ thực hiện đem fold đó đi huấn luyện trên tất cả mô hình và tính giá trị mae tương ứng. Lưu các giá trị **mae** có được sau khi huấn luyện fold đang xét qua các mô hình vào list mae_list. List này sẽ như một ma trận 2 chiều để lưu trữ các giá trị **mae**, các phần tử cùng một hàng sẽ là các giá trị **mae** của một fold xác định khi được huấn luyện trên n mô hình. Cuối cùng sẽ tính giá trị **mae trung bình** của từng mô hình sau khi được huấn luyện trên k fold bằng hàm np.mean(), tham số axis = 1 là để thực hiện tính trung bình của các phần tử trên cùng một cột vì các phần tử này là giá trị **mae** của một mô hình sau khi huấn luyện trên k fold.
- Phương thức best_model() của class k_folds_cross_validation
  - **INPUT:** Không có.
  - **OUTPUT:** Trả về kết quả mae trung bình của từng mô hình sau khi thực hiện kĩ thuật k-folds cross validation và mô hình được xem là tốt nhất.
- Hàm mae()

- **INPUT:** Mảng chứa các giá trị dự đoán có được từ mô hình y_hat và mảng chứa các giá trị dự đoán thực sự trên tập test y.
- **OUTPUT:** Trả về giá trị **mae** (độ lỗi tuyệt đối trung bình) của mô hình.
- **Chi tiết:** Lấy trung bình của tổng các sai số (độ lệch giữa giá trị dự đoán với giá trị thực sự). Hàm ravel() được sử dụng để chuyển đổi ma trận thành một mảng 1 chiều để tính toán cho thuận tiện.

- Hàm calc_mae_current_model()
  - **INPUT:** Các giá trị của các đặc trưng và mục tiêu của một mô hình cụ thể trên tập train và test X_train, y_train, X_test, y_test.
  - **OUTPUT:** Giá trị **mae** tương ứng của mô hình.
  - **Chi tiết:** Có chứ năng tương ứng như hàm mae(), chỉ sửa lại đôi chút để dễ sử dụng hơn cho câu **1D**.
- Hàm backward_elimination()
  - **INPUT:** Tập dữ liệu train, tập dữ liệu test, giá trị SL (Significance Level).
  - **OUTPUT:** Các đặc trưng có trong mô hình sau khi được chọn lọc.
  - **Chi tiết:**

    Bước 1: Xác định ngưỡng giá trị tối đa của **p-value** **[8]** hay còn gọi là **significance value (SL)**. Nó là một ngưỡng quyết định để xác định xem giả thuyết không có tác động có thể bị bác bỏ hay không. Các giá trị phổ biến cho mức ý nghĩa bao gồm 0.05 (5%) và 0.01 (1%). Lý do sử dụng **p-value** trong phương pháp **backward elimination** là để đánh giá mức độ ảnh hưởng của từng **đặc trưng** đến **biến phụ thuộc**. Nếu một biến có **p-value** lớn hơn ngưỡng đáng chú ý, có nghĩa là **không** có đủ bằng chứng để cho rằng biến đó có ảnh

hưởng đáng kể đến biến phụ thuộc. Do đó, việc loại bỏ biến đó khỏi mô hình **không** gây mất mát đáng kể về khả năng dự đoán.

Bước 2: Xây dựng mô hình hồi quy tuyến tính với tất cả các đặc trưng có trong selected_features bằng sm.OLS(endog = y_train, exog = X_train).fit(). Sử dụng vòng lặp while True để thực hiện đến khi đạt điều kiện dừng.

Bước 3: Sử dụng các công cụ, thư viện có sẵn (cụ thể là statsmodels.api) để tính toán ra **p-value** của các đặc trưng và lưu trữ chúng trong mảng bằng regressor_OLS.pvalues.to_numpy(). Lấy đặc trưng có giá trị **p-value** cao nhất. Nếu nó cao hơn **SL** thì sang **bước 4**. Nếu không thì sẽ kết thúc quá trình.

- Bước 4: Loại bỏ đặc trưng này ra khỏi mô hình để có mô hình mới. Tính giá trị **mae (Mean Absolute Error)** của mô hình mới này. Nếu nhỏ hơn **mae** của mô hình cũ thì ta sẽ giữ lại mô hình cũ và kết thúc phương pháp. Nếu không thì ta sẽ quay lại **Bước 2** với các đặc trưng còn được giữ lại trong selected_features.

- Hàm forward_elimination()
  - **INPUT:** Tập dữ liệu train, tập dữ liệu test, giá trị SL (Significance Level).
  - **OUTPUT:** Các đặc trưng có trong mô hình sau khi được chọn lọc.
  - **Chi tiết:**

Bước 1: Xác định ngưỡng giá trị tối đa của **p-value** hay còn gọi là **significance value (SL)** như ở phương pháp **Backward Elimination** để xác định xem đặc trưng nào sẽ được thêm vào mô hình.

Bước 2: Xây dựng các mô hình với chỉ một đặc trưng lấy ra từ tập hợp các đặc trưng remain_features. Mô hình nào có đặc trưng với **p-value** nhỏ nhất sẽ được giữ lại và thêm vào selected_features.

Bước 3: Lần lượt bổ sung một đặc trưng từ tập các đặc trưng còn lại remain_features vào mô hình đang có.

Bước 4: Xét **p-value** của các đặc trưng mới thêm vô trong mô hình mới. Tìm ra **p-value** nhỏ nhất. Nếu nó nhỏ hơn giá trị **SL** và **mae** của mô hình mới này nhỏ hơn **mae** của mô hình trước đó thì sẽ chính thức thêm đặc trưng mới này vào mô hình hiện tại bằng selected_features.append(), còn **mae** của mô hình mới này lớn hơn **mae** của mô hình trước đó thì sẽ không thêm đặc trưng này vào và loại bỏ nó hoàn toàn khỏi tập các đặc trưng còn lại remain_features, sau đó sẽ quay lại **Bước 3**. Nếu **p-value** nhỏ nhất lớn hơn giá trị **SL** thì sẽ giữ lại mô hình cũ (mô hình trước khi thêm đặc trưng mới) và kết thúc quá trình.

IV. Báo cáo và nhận xét kết quả cho từng yêu cầu

1. Câu 1a

- **Yêu cầu:**

- Sử dụng 11 đặc trưng đầu tiên: **Gender**, **10percentage**, **12percentage**, **CollegeTier**, **Degree**, **collegeGPA**, **CollegeCityTier**, **English**, **Logical**, **Quant**, **Domain** để xây dựng mô hình hồi quy tuyến tính dự đoán mức lương của các kĩ sư ở Ấn Độ.
- Huấn luyện **1 lần** duy nhất cho mô hình với 11 đặc trưng trên cho toàn bộ tập train (train.csv).
- Báo cáo kết quả (mae) trên tập test (test.csv) cho mô hình vừa huấn luyện.

- **Cách triển khai:**

- Lưu các đặc trưng trong mô hình vào features_1a.
- Từ tập train chỉ lấy ra các giá trị trên các cột đặc trưng tương ứng X_train_1a

= train.loc[:, features_1a] và các giá trị mục tiêu y_train_1a = train.iloc[:, -1].

- Làm tương tự trên tập test.
- Bắt đầu xây dựng mô hình hồi quy tuyến tính eleven_features_model và huấn luyện nó 1 lần trên tập huấn luyện (train).
- Lấy và in các trọng số tương ứng với từng đặc trưng của mô hình vừa huấn luyện thông qua phương thức get_params() của class OLSLinearRegression.
- Tính giá trị dự đoán của tập kiểm tra dựa vào mô hình vừa có nhờ phương thức predict() của class OLSLinearRegression và từ đó tính giá trị **mae (mean absolute error)** của mô hình.

- **<u>Kết quả:</u>**

- Công thức hồi quy của mô hình:

$$\begin{aligned}\mathbf{Salary} = &-\mathbf{22756.513} \times \mathbf{Genger} + \mathbf{804.503} \times \mathbf{10percentage} \\ &+ \mathbf{1294.655} \times \mathbf{12percentage} - \mathbf{91781.897} \times \mathbf{CollegeTier} \\ &+ \mathbf{23182.389} \times \mathbf{Degree} + \mathbf{1437.549} \times \mathbf{collegeGPA} \\ &- \mathbf{8570.662} \times \mathbf{CollegeCityTier} + \mathbf{147.858} \times \mathbf{English} \\ &+ \mathbf{152.888} \times \mathbf{Logical} + \mathbf{117.222} \times \mathbf{Quant} \\ &+ \mathbf{34552.286} \times \mathbf{Domain}\end{aligned}$$

- Giá trị **mae** của mô hình:

$$\mathbf{MAE} = \mathbf{104863.777}$$

- **<u>Nhận xét:</u>**

- Với giá trị **mae** của mô hình thì ta có thể thấy mô hình với 11 đặc trưng đầu tiên không phải là mô hình thực sự tốt, tuy nhiên vẫn có thể tạm chấp nhận vì

mô hình này xây dựng một cách ngẫu nhiên chứ không hề được tạo ra do có sự tính toán ha phương pháp khoa học cụ thể.

- Một điều làm cho mô hình này không tốt vì nó vẫn chứa những đặc trưng **"rác"**, đây là những đặc trưng không có đóng góp đáng kể cho độ chính xác của việc dự đoán của mô hình. Ta có thể dựa vào **hệ số (coefficient)** của đặc trưng trong mô hình hoặc giá trị **p-value** của chúng thông qua phương pháp kiểm định thống kê. Cụ thể ta sẽ có thể kiểm định thống kê mô hình của ta để có được **p-value** của các đặc trưng nhờ class api có trong thư viện statsmodels.

OLS Regression Results

Dep. Variable:	Salary	R-squared (uncentered):	0.686
Model:	OLS	Adj. R-squared (uncentered):	0.684
Method:	Least Squares	F-statistic:	444.4
Date:	Thu, 24 Aug 2023	Prob (F-statistic):	0.00
Time:	00:19:56	Log-Likelihood:	-30763.
No. Observations:	2248	AIC:	6.155e+04
Df Residuals:	2237	BIC:	6.161e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Gender	-2.276e+04	1.09e+04	-2.087	0.037	-4.41e+04	-1377.825
10percentage	804.5032	612.025	1.314	0.189	-395.693	2004.700
12percentage	1294.6546	548.376	2.361	0.018	219.276	2370.033
CollegeTier	-9.178e+04	1.38e+04	-6.675	0.000	-1.19e+05	-6.48e+04
Degree	2.318e+04	1.63e+04	1.419	0.156	-8848.409	5.52e+04
collegeGPA	1437.5487	582.872	2.466	0.014	294.522	2580.576
CollegeCityTier	-8570.6620	9999.387	-0.857	0.391	-2.82e+04	1.1e+04
English	147.8583	49.517	2.986	0.003	50.754	244.963
Logical	152.8885	62.037	2.464	0.014	31.233	274.544
Quant	117.2218	45.647	2.568	0.010	27.707	206.736
Domain	3.455e+04	9898.125	3.491	0.000	1.51e+04	5.4e+04

- Qua bảng thống kê trên, ta có thể thấy được các đặc trưng **"rác"** là 10percentage, Degree và CollegeCityTier vì giá trị **p-value** của chúng cao hơn hẳn so với các đặc trưng khác, vì vậy việc bỏ chúng khỏi mô hình hiện tại sẽ giúp mô hình của ta tốt hơn. Tuy nhiên **p-value** vẫn sẽ có điểm yếu mà ta sẽ đề cập cụ thể hơn ở câu **1d**.

2. Câu 1b

- **<u>Yêu cầu:</u>**

- Xét 5 mô hình với mỗi mô hình chỉ có 1 đặc trưng duy nhất trong các đặc trưng: conscientiousness, agreeableness, extraversion, neuroticism, openess to_experience.
- Phân tích tầm ảnh hưởng của các đặc trưng này tới **Salary** bằng cách so sánh giá trị **mae** của từng mô hình với đặc trưng tương ứng.
- Dùng kĩ thuật **k-folds cross validation** để chọn ra mô hình tốt nhất. Huấn luyện nó trên toàn bộ tập train và tính **mae** của mô hình này.
- Báo cáo kết quả **mae trung bình** của 5 mô hình khi sử dụng **k-folds cross validation** và **mae** của mô hình tốt nhất.
- Thể hiện công thức hồi quy tuyến tính của mô hình tốt nhất.

- **<u>Cách triển khai:</u>**

- Lưu các đặc trưng sẽ sử dụng vào features_1b.
- Với mỗi mô hình sẽ có đặc trưng nào thì sẽ lưu vào 1 list để quản lí, tất cả mô hình sẽ được lưu vào list models_1b.
- Sử dụng các phương thức shuffle_data(), split_to_k_folds(), cross_validation() của class **k_folds_cross_validation** để thực hiện xáo trộn dữ liệu, chia thành k

fold như nhau và huấn luyện 5 mô hình trên từng fold. Cuối cùng tính giá trị mae trung bình của từng mô hình và tìm ra mô hình tốt nhất nhờ phương thức best_model().

- Có được mô hình tốt rồi thì ta sẽ quay về lại như câu **1a** đó chính là đem mô hình best_personality_feature_model đi huấn luyện trên toàn bộ tập train. Tìm ra được công thức hồi quy tuyến tính của mô hình tốt nhất và cuối cùng tính ra được giá trị **mae** của mô hình này.

- **Kết quả:**

- **Mae trung bình** sau khi thực hiện **k-folds cross validation**

STT	Mô hình với đặc trưng	MAE
1	conscientiousness,	304945.954
2	agreeableness	299281.089
3	extraversion	305733.283
4	neuroticism,	**297666.275**
5	openess_to_experience	302362.676

- **Nhận xét:** Từ 5 kết quả mae trung bình có từ k-folds cross validation, ta có thể nhận ra được đặc trưng neuroticism là đặc trưng ảnh hưởng nhất tới Salary, tuy nhiên kết quả trên cũng không quá chính xác 100% vì ta thấy chênh lệch giữa mae trung bình của agreeableness và neuroticism không quá cao. Nếu ta trộn bộ dữ liệu của ta theo cách khác thì có thể agreeableness sẽ là đặc trưng tốt hơn, tuy nhiên thì neuroticism vẫn sẽ là một đặc trưng có tầm ảnh hưởng hơn cả so với 4 đặc trưng còn lại. Tuy vậy những chỉ số mae này có thể coi là khá cao. Từ đó có thể nhận xét chung là những đặc trưng

này không giải thích được sự thay đổi của mục tiêu Salary. Vì thế mức độ ảnh hưởng của những đặc trưng này lên giá trị mục tiêu khi đứng riêng lẻ là không thực sự đáng tin.

- **<u>Kết quả mô hình tốt nhất:</u>**

- Mô hình tốt nhất là mô hình với đặc trưng: **neuroticism**.
- **MAE** của mô hình tốt nhất:

$$\mathbf{MAE} = \mathbf{291019.693}$$

- Mô hình hồi quy tuyến tính:

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

- **<u>Nhận xét và nêu giả thuyết:</u>**

- Kết quả mae của mô hình cho thấy mô hình này không phải là một mô hình tốt, nó tệ đi rất nhiều so với mô hình 11 đặc trưng ở câu 1a, qua đó cho ta một góc nhìn rỏ hơn về mô hình một đặc trưng thực sự tệ như thế nào. Ngoài ra trọng số của neuroticism còn cho thấy đây là một đặc trưng ảnh hưởng mạnh mẽ theo hướng tiêu cực đến tiền lương salary.
- Có rất nhiều giả thuyết để giải thích cho lý do trong 5 đặc trưng trên thì neuroticism lại là đặc trưng có tầm ảnh hưởng lớn nhất đối với giá trị mục tiêu Salary.
- Dựa vào công thức của mô hình trên, với **trọng số âm** thì ta dễ dàng nhận ra người có mức độ neuroticism cao thường có xu hướng khó lòng hài lòng với công việc của mình, đó chính là nguyên nhân ảnh hưởng đến lượng (salary) của họ giảm một cách tiêu cực. Họ có thể tỏ ra lo lắng, căng thẳng và dễ bị ảnh hưởng bởi các vấn đề tiêu cực trong môi trường làm việc. Sự không hài

lòng với công việc có thể dẫn đến việc không đạt được sự phát triển nghề nghiệp và khả năng tăng lương.

- Neuroticism có thể ảnh hưởng đến khả năng tương tác xã hội và quan hệ làm việc. Người có mức độ neuroticism cao có thể có xu hướng khó khăn trong việc xây dựng và duy trì quan hệ làm việc tích cực với đồng nghiệp, cấp trên hoặc khách hàng. Điều này có thể ảnh hưởng đến khả năng tạo ra mạng lưới quan hệ và cơ hội nghề nghiệp, có thể làm giảm khả năng tiến xa trong công việc và tăng lương.
- Neuroticism có thể làm tăng khả năng cảm thấy căng thẳng và lo lắng trước áp lực công việc. Những người có mức độ neuroticism cao có thể gặp khó khăn trong việc quản lý stress và thích ứng với các tình huống khó khăn trong công việc. Điều này có thể ảnh hưởng đến hiệu suất làm việc và khả năng đạt được thành công nghề nghiệp.
- Theo bài viết **The Personality Traits That Increase Your Salary [2]**, yếu tố neuroticism có mức độ ảnh hưởng đến tiền lương là 5-9%, cao hơn so với tất cả 4 yếu tố còn lại.

3. Câu 1c

- **Yêu cầu:**

- Xét 3 mô hình với mỗi mô hình chỉ có 1 đặc trưng duy nhất trong các đặc trưng: English, Logical, Quant.
- Phân tích tầm ảnh hưởng của các đặc trưng này tới **Salary** bằng cách so sánh giá trị **mae** của từng mô hình với đặc trưng tương ứng.
- Dùng kĩ thuật **k-folds cross validation** để chọn ra mô hình tốt nhất. Huấn

luyện nó trên toàn bộ tập train và tính **mae** của mô hình này.

- Báo cáo kết quả **mae trung bình** của 3 mô hình khi sử dụng **k-folds cross validation** và **mae** của mô hình tốt nhất.
- Thể hiện công thức hồi quy tuyến tính của mô hình tốt nhất.

- **Cách triển khai:**

- Lưu các đặc trưng sẽ sử dụng vào features_1c.
- Với mỗi mô hình sẽ có đặc trưng nào thì sẽ lưu vào 1 list để quản lí, tất cả mô hình sẽ được lưu vào list models_1c.
- Sử dụng các phương thức shuffle_data(), split_to_k_folds(), cross_validation() của class **k_folds_cross_validation** để thực hiện xáo trộn dữ liệu, chia thành k fold như nhau và huấn luyện 3 mô hình trên từng fold. Cuối cùng tính giá trị mae trung bình của từng mô hình và tìm ra mô hình tốt nhất nhờ phương thức best_model().
- Có được mô hình tốt rồi thì ta sẽ quay về lại như câu **1a** đó chính là đem mô hình best_skill_feature_model đi huấn luyện trên toàn bộ tập train. Tìm ra được công thức hồi quy tuyến tính của mô hình tốt nhất và cuối cùng tính ra được giá trị **mae** của mô hình này.

- **Kết quả:**

- **Mae trung bình** sau khi thực hiện **k-folds cross validation:**

STT	Mô hình với đặc trưng	MAE
1	English	121714.645
2	Logical	120323.481

3	Quant	**117863.434**

- **<u>Nhận xét:</u>** Từ 3 kết quả mae trung bình có từ k-folds cross validation, ta có thể nhận ra được đặc trưng Quant là đặc trưng ảnh hưởng nhất tới Salary, Tuy vậy những chỉ số mae này có thể coi là khá ổn mặc dù các đặc trưng này phải nằm riêng lẻ. Từ đó có thể nhận xét chung là những đặc trưng này thực sự đóng góp sự ảnh hưởng tương đối lớn với giá trị mục tiêu Salary và sự ảnh hưởng của chúng có thể xem là đáng tin cậy hơn so với những đặc trưng ở câu 1b.

- **<u>Kết quả của mô hình tốt nhất:</u>**

- Mô hình tốt nhất là mô hình với đặc trưng: **Quant.**
- **MAE** của mô hình tốt nhất:

$$\mathbf{MAE} = \mathbf{106819.578}$$

- Mô hình hồi quy tuyến tính:

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

- **<u>Nhận xét và nêu giả thuyết:</u>**

- So sánh MAE của mô hình tính cách tốt nhất (Quant) với mô hình 11 đặc trưng ở câu 1a có thể thấy, có thể thấy 2 chỉ số không quá chênh lệch nhau. Điều này lại càng chứng tỏ Quant là một đặc trưng quan trọng góp phần giải thích được Salary vì dù mô hình chỉ có 1 đặc trưng, nhưng đã mang lại hiệu suất tương đương với mô hình 11 đặc trưng.
- Trọng số của Quant cho thấy tuy Quant không ảnh hưởng quá nhiều đến sự thay đổi của Salary nhưng nếu có thì sẽ theo chiều hướng tích cực (đồng biến).
- Đặc trưng Quant có thể liên quan đến khả năng và kiến thức về tính toán, số

học và phân tích dữ liệu. Trong nhiều ngành nghề, nhất là trong lĩnh vực kỹ thuật, tài chính và công nghệ thông tin, khả năng làm việc với dữ liệu số và phân tích số liệu là yêu cầu quan trọng. Do đó, sự thành thạo trong đặc trưng Quant có thể đóng góp đáng kể đến khả năng thực hiện công việc và giá trị chuyên môn của cá nhân, từ đó ảnh hưởng tích cực đến mức lương của ta.

- Các kỹ năng liên quan đến đặc trưng Quant, chẳng hạn như khả năng phân tích số liệu và xử lý thông tin số, có thể khan hiếm trong một số ngành nghề hoặc thị trường lao động. Khi một đặc trưng hiếm và có giá trị cao, như Quant, các chuyên gia có khả năng về đặc trưng này có thể có lợi thế trong việc tìm kiếm việc làm và đàm phán mức lương cao hơn.
- Một số công việc hoặc vị trí có tính chất cụ thể yêu cầu một mức độ cao về đặc trưng Quant. Ví dụ, trong lĩnh vực tài chính, quản lý rủi ro, phân tích dữ liệu, hoặc trong lĩnh vực nghiên cứu khoa học, các kỹ năng liên quan đến đặc trưng Quant có thể là yếu tố quyết định để đảm bảo hiệu suất công việc và thành công. Do đó, mức độ ảnh hưởng của đặc trưng Quant đối với Salary có thể cao hơn trong các lĩnh vực này.
- Theo topic thảo luận **What is the average salary of a Quant in US/UK and Europe as a beginner and how does it vary with experience** trên Quora **[4]**, mức lương của những người có cho mình phẩm chất "Quant" tốt đều hay các chuyên gia quant (những người chuyên về phân tích số liệu, xử lí dữ liệu,...) được nhận lương rất cao dù chỉ mới từ 0-2 năm kinh nghiệm ($70,000 to $100,000 mỗi năm) và con số còn tăng nhiều hơn nữa cho thấy đặc trưng Quant có tầm ảnh hưởng quan trọng như thế nào với Salary .

4. Câu 1d

- **Yêu cầu:**

- Tự xây dựng cho mình m mô hình (m >= 3) rồi từ đó sử dụng kĩ thuật k-folds cross validation để tìm ra mô hình tốt nhất.
- Báo cáo kết quả mae trung bình của m mô hình từ k-folds cross validation.
- Báo cáo kết quả mae của mô hình tốt nhất và công thức hồi quy tuyến tính của mô hình này.

- **Cách triển khai**:

- Có rất nhiều phương pháp để xây dựng một mô hình hồi quy tuyến tính, dưới đây sẽ là 3 phương pháp được sử dụng phổ biến và dễ thực hiện: **All-in, Backward Elimination, Forward Elimination**.
- Cả 2 phương pháp cuối đều sử dụng một yếu tố để đánh giá xem 1 đặc trưng có thực sự ảnh hưởng, đóng góp tới kết quả của biến phụ thuộc hay không đó chính là **p-value**, một giá trị mà ta đã học ở môn **xác suất thống kê**, được sử dụng để đánh giá sự tin cậy của kết quả và đưa ra quyết định về việc có chấp nhận hay từ chối giả thuyết. Thông thường, ngưỡng đáng chú ý (đặt trước) được chọn và so sánh với **p-value** để đưa ra quyết định. Tuy nhiên nếu không sử dụng **p-value**, ta vẫn còn có nhiều giá trị khác để đánh giá, có thể xem tham khảo tại video **[5]**.
- Vậy câu hỏi đặt ra là **Tại sao trong các phương pháp trên lại sử dụng p-value cho từng feature thì có thể xác định mô hình tốt nhất nên có feature ấy hay không?** Lý do sử dụng **p-value** là để đánh giá mức độ ảnh hưởng của từng đặc trưng đến giá trị dự đoán. Nếu một biến có **p-value** lớn hơn **ngưỡng đáng chú ý SL (significance level)**, có nghĩa là không có đủ

bằng chứng để cho rằng đặc trưng đó có ảnh hưởng đáng kể đến biến phụ thuộc. Do đó, việc loại bỏ biến đó khỏi mô hình không gây mất mát đáng kể về khả năng dự đoán.

- **Phương pháp All-in**: Đây không hẳn là một kĩ thuật/phương pháp vì nhiệm vụ của ta là chỉ đơn giản dùng tất cả các đặc trưng trong mô hình. Đây là điều không nên làm vì ta biết sẽ có những **garbage feature** làm cho mô hình của ta tệ đi, tuy nhiên trong đồ án này thì việc sử dụng toàn bộ đặc trưng mà ta có trong một mô hình sẽ cho ta một kết quả khả quan. Vì vậy mô hình sử dụng toàn bộ các đặc trưng cũng không phải là một lựa chọn quá tồi đối với bộ dataset trong đồ án này.
- **Lí do sử dụng hai phương pháp Backward/Forward Elimination**: trong dữ liệu của chúng ta có rất nhiều đặc trưng X dùng để dự đoán cho giá trị cần tìm y, tuy nhiên việc sử dụng tất cả các đặc trưng X trong một mô hình là không phải một điều hay vì một trong số chúng có thể là **garbage feature**. Các **garbage feature** có thể làm cho mô hình của ta trở nên tệ hơn vì vậy khi xây dựng mô hình hồi quy tuyến tính cần xem xét kĩ nên giữ hay loại bỏ các đặc trưng nào để giúp cho mô hình của ta tốt nhất có thể.
- **Phương pháp Backward Elimination [7]**: **Bước 1:** Xác định ngưỡng giá trị tối đa của **p-value** hay còn gọi là **significance value (SL)**. Nó là một ngưỡng quyết định để xác định xem giả thuyết không có tác động có thể bị bác bỏ hay không. Các giá trị phổ biến cho mức ý nghĩa bao gồm 0.05 (5%) và 0.01 (1%). **Bước 2**: Xây dựng mô hình hồi quy tuyến tính với tất cả các **biến độc lập** có trong dữ liệu. **Bước 3**: Sử dụng các công cụ, thư viện có sẵn (cụ thể là statsmodels.api) để tính toán ra **p-value** của các đặc trưng. Lấy đặc trưng có

giá trị **p-value** cao nhất. Nếu nó cao hơn **SL** thì sang **bước 4**. Nếu không thì sẽ kết thúc quá trình. **Bước 4**: Loại bỏ đặc trưng này ra khỏi mô hình để có mô hình mới. Tính giá trị **mae (Mean Absolute Error)** của mô hình mới này. Nếu nhỏ hơn **mae** của mô hình cũ thì ta sẽ giữ lại mô hình cũ và kết thúc phương pháp. Nếu không thì ta sẽ quay lại **Bước 2** với các đặc trưng còn được giữ lại. Nếu vẫn còn chưa rỏ, ta có thể tham khảo cách làm qua các video hướng dẫn Backward Elimination.

- **Phương pháp Forward Elimination**: **Bước 1:** Xác định ngưỡng giá trị tối đa của **p-value** hay còn gọi là **significance value (SL)** như ở phương pháp **Backward Elimination** để xác định xem đặc trưng nào sẽ được thêm vào mô hình. **Bước 2:** Xây dựng các mô hình với chỉ một đặc trưng lấy ra từ tập hợp các đặc trưng. Mô hình nào có đặc trưng có **p-value** nhỏ nhất sẽ được giữ lại. **Bước 3:** Lần lượt bổ sung một đặc trưng từ tập các **đặc trưng còn lại** vào mô hình đang có. **Bước 4:** Xét **p-value** của các đặc trưng mới thêm vô trong mô hình mới. Tìm ra **p-value** nhỏ nhất. Nếu nó nhỏ hơn cả giá trị **SL** và **mae** của mô hình mới này nhỏ hơn **mae** của mô hình trước đó thì sẽ chính thức thêm đặc trưng mới này vào mô hình hiện tại còn không thì sẽ không thêm đặc trưng này vào và loại bỏ nó hoàn toàn khỏi tập các đặc trưng còn lại , sau đó sẽ quay lại **Bước 3**. Nếu không sẽ giữ lại mô hình cũ (mô hình trước khi thêm đặc trưng mới) và kết thúc quá trình.
- Việc có được mô hình từ phương pháp **All-in** rất đơn giản `first_model_1d = train.columns.to_list()[:-1]`.
- Mô hình có được từ phương pháp **Backward Elimination** sẽ được trả về từ hàm `backward_elimination()` và sẽ được lưu vào `second_model_1d`.

- Mô hình có được từ phương pháp **Forward Elimination** sẽ được trả về từ hàm forward_elimination() và sẽ được lưu vào third_model_1d.
- Có được 3 mô hình từ 3 phương pháp trên rồi thì sẽ áp dụng phương pháp **k-folds cross validation** để xác định mô hình nào tốt nhất.
- In ra giá trị **mae** trung bình của 3 mô hình từ **k-folds cross validation**.
- Lấy mô hình tốt nhất đi huấn luyện để tìm ra công thức hồi quy tuyến tính cũng như giá trị **mae**.

- **<u>Kết quả:</u>**

- Mô hình có được từ phương pháp All-in:

$$\begin{gathered}\text{Model1:}\\ \text{Salary} = w1 \times Gender + w2 \times 10percentage + w3 \times 12percentage + w4 \times CollegeTier \\ + w5 \times Degree + w6 \times collegeGPA + w7 \times CollegeCityTier + w8 \times English + w9 \times Logical \\ + w10 \times Quant + w11 \times Domainw12 \times ComputerProgramming + w13 \times ElectronicsAndSemicon \\ + w14 \times ComputerScience + w15 \times MechanicalEngg + w16 \times ElectricalEngg + w17 \times TelecomEngg \\ + w18 \times CivilEngg + w19 \times conscientiousness + w20 \times agreeableness + w21 \times extraversion \\ + w22 \times nueroticism + w23 \times openesstoexperience\end{gathered}$$

- Mô hình có được từ phương pháp Backward Elimination:

$$\begin{gathered}\text{Model2:}\\ \text{Salary} = w1 \times Gender + w2 \times 10percentage + w3 \times 12percentage + w4 \times CollegeTier + w5 \times collegeGPA \\ + w6 \times English + w7 \times Logical + w8 \times Quant + w9 \times Domain + w10 \times ComputerProgramming \\ + w11 \times ElectronicsAndSemicon + w12 \times ComputerScience + w13 \times ElectricalEngg \\ + w14 \times TelecomEngg + w15 \times CivilEngg + w16 \times conscientiousness + w17 \times agreeableness \\ + w18 \times extraversion + w19 \times nueroticism + w20 \times openesstoexperience\end{gathered}$$

- Mô hình có được từ phương pháp Forward Elimination:

$$\begin{gathered}\text{Model3:}\\ \text{Salary} = w1 \times Gender + w2 \times Logical + w3 \times Quant + w4 \times ComputerScience \\ + w5 \times ComputerProgramming + w6 \times 10percentage + w7 \times CollegeTier + w8 \times collegeGPA \\ + w9 \times ElectricalEngg + w10 \times Domain + w11 \times conscientiousness + w12 \times ElectronicsAndSemicon \\ + w13 \times 12percentage\end{gathered}$$

- Kết quả có được từ k-folds cross validation

STT	Mô hình có được từ phương pháp	MAE

1	All-in	110180.473
2	Backward Elimination	**109938.754**
3	Forward Elimination	111724.380

- Mô hình tốt nhất là mô hình thứ 2, có được từ phương pháp Backward Elimination.
- Công thức hồi quy tuyến tính của mô hình tốt nhất:

$$\begin{aligned}\boldsymbol{textSalary} &= \boldsymbol{-24571.327 \times Gender + 880.012 \times 10percentage} \\ &\boldsymbol{+ 1176.514 \times 12percentage - 80548.712 \times CollegeTier} \\ &\boldsymbol{+ 1778.113 \times collegeGPA + 153.544 \times English} \\ &\boldsymbol{+ 124.193 \times Logical + 103.584 \times Quant + 29779.497 \times Domain} \\ &\boldsymbol{+ 68.792 \times ComputerProgramming} \\ &\boldsymbol{- 56.398 \times ElectronicsAndSemicon} \\ &\boldsymbol{- 180.668 \times ComputerScience - 159.574 \times ElectricalEngg} \\ &\boldsymbol{- 70.042 \times TelecomEngg153.809 \times CivilEngg} \\ &\boldsymbol{- 19782.870 \times conscientiousness} \\ &\boldsymbol{+ 15167.240 \times agreeableness + 5031.589 \times extraversion} \\ &\boldsymbol{- 10670.594 \times nueroticism} \\ &\boldsymbol{- 5619.404 \times openess\_to\_experience}\end{aligned}$$

- MAE của mô hình:

$$\text{MAE} = \mathbf{101421.379}$$

- **<u>Nhận xét:</u>**

- Từ mae trung bình có được nhờ kĩ thuật k-folds cross validation thì mô hình

có được từ phương pháp Backward Elimination 'tạm' được xem là mô hình tốt nhất trong 3 mô hình ở câu 1d này. Tuy nhiên chắc chắn nó sẽ là mô hình tốt nhất trong đồ án này. Lí do vì sao dùng từ "tạm" là vì mae trung bình của 3 mô hình rất sát sao nhau. Có thể với cách trộn dữ liệu khác thì mô hình tốt nhất sẽ không còn là mô hình của phương pháp Backward Elimination. Nhưng ta vẫn sẽ chấp nhận xem đây là mô hình tốt nhất có thể có trong đồ án này.

- Phương pháp All-in thì không có gì để bàn vì đây là phương pháp đơn giản, độ hiệu quả của nó có thể xem là "hên xui" tùy thuộc vào bộ dữ liệu. Vì thế ta sẽ chỉ nhận xét về ưu điểm cũng như nhược điểm của hai phương pháp còn lại.
- **Ưu điểm của Backward Elimination:** bắt đầu với một mô hình đầy đủ bao gồm tất cả các biến độc lập, sau đó loại bỏ từng biến một cách tuần tự dựa trên giá trị p-value hoặc các tiêu chí khác. Điều này giúp tiết kiệm thời gian và công sức so với việc xây dựng lại các mô hình từ đầu. Qua quá trình loại bỏ biến một cách tuần tự, phương pháp backward elimination giúp tìm ra một mô hình đơn giản nhất có thể với các biến mang lại ảnh hưởng ít đáng kể hoặc không đáng kể đến biến phụ thuộc. Điều này giúp giảm hiện tượng overfitting và làm cho mô hình dễ hiểu và dễ diễn giải.
- **Ưu điểm của Forward Elimination:** phương pháp Forward Elimination xây dựng mô hình bằng cách thêm từng biến một vào mô hình ban đầu và đánh giá tác động của từng biến tới biến phụ thuộc. Điều này giúp xác định được tác động độc lập của từng biến và đánh giá tính quan trọng của chúng trong mô hình. Nó cho phép thêm từng biến một vào mô hình, do đó, có khả năng

phát hiện và xác định các tương tác giữa các biến. Điều này rất hữu ích khi các biến tương tác có thể có ảnh hưởng đáng kể đến biến phụ thuộc.

- **Nhược điểm:** Cả 2 đều có điểm chung là dựa rất nhiều vào p-value, tuy nhiên p-value lại có nhược điểm là phụ thuộc vào mẫu dữ liệu. Kết quả có thể khác nhau nếu bạn sử dụng mẫu dữ liệu khác nhau. Hai phương pháp này không xem xét tương quan giữa các biến độc lập. Một biến có p-value cao có thể vẫn quan trọng nếu nó liên quan mật thiết đến các biến khác trong mô hình. Việc loại bỏ một biến có thể ảnh hưởng đến mô hình toàn bộ. Việc xóa một biến cần được xem xét kỹ lưỡng để đảm bảo không mất mát quan trọng trong khả năng dự đoán hay khả năng giải thích của mô hình. Các phương pháp này chỉ tạo ra một mô hình tương đối tốt dựa trên các tiêu chí xác định trước, và việc lựa chọn biến nên được kết hợp với kiểm định và đánh giá mô hình để đảm bảo tính chính xác và hiệu quả của mô hình.

- **Giả thuyết cho mô hình tốt nhất:**

- Như vậy mô hình có được từ **Backward Elimination** chính là mô hình tốt nhất ta đang có. Xét mô hình trước và sau khi thực hiện **Bakcward Elimination** thì có 3 đặc trưng bị loại ra khỏi: **CollegeCityTier, Degree, MechanicalEngg**.
- Đầu tiên nói về **CollegeCityTier**, đúng là yếu tố thứ hạng của thành phố có trường đại học mà một người đang theo có thể ảnh hưởng đến lương người đó vì đó sẽ là nơi mà người đó làm việc sau khi kết thúc việc học. Nếu yếu tố **CollegeCityTier** càng cao chứng tỏ đó là nơi tập trung nhiều nguồn lực, cơ hội việc làm cao cũng như điều kiện kinh tế phát triển. Theo thống kê của tòa báo **Thư Viện Pháp Luật [9]**, mức lương tối thiểu của người sống ở Thành

phố Hồ Chí Minh là **4.680.000VND** còn lương tối thiểu của những người ở các tỉnh miền Tây như Long An, Tiền Giang chỉ vào **khoảng hơn 4 triệu VND**, qua đó cho thấy phần nào sự ảnh hưởng của **CollegeCityTier** đến **Salary**. Tuy nhiên đó là nếu ta xếp hạng, so sánh nhiều thành phố, còn đằng này trong dữ liệu của ta chỉ có 2 cấp bậc là 0 và 1 thì sự ảnh hưởng của **CollegeCityTier** gần như là vô hại. Lấy ví dụ giữa Thành phố Hồ Chí Minh và Hà Nội, dựa vào dân số ở năm 2019 **[10]** thì dĩ nhiên **CollegeCityTier** của Hà Nội sẽ thấp hơn của Thành phố Hồ Chí Minh nên nếu ta kết luận lương của các kĩ sư ở TPHCM cao hơn ở Hà Nội thì đó là một sự sai lầm lớn. Vì vậy nếu miền giá trị của **CollegeCityTier** nhỏ thì đây sẽ là một **garbage feature**, vì nó không chỉ không tác động đến **Salary** mà còn làm giảm độ chính xác của mô hình dự đoán tiền lương.

- Tiếp theo là về **Degree**, đây là một yếu tố đi liền với **CollegeCityTier** và **CollegeTier**, vì thực tế bằng cấp "cử nhân" hay "thạc sĩ" của những trường đại học khác nhau thì sẽ có giá trị khác nhau. Bằng cấp "cử nhân" và "thạc sĩ" thể hiện trình độ học vấn cao hơn so với các bằng cấp khác. Những người có trình độ học vấn cao thường có kiến thức chuyên môn sâu hơn và kỹ năng đặc thù trong lĩnh vực họ theo đuổi. Điều này làm tăng khả năng ứng dụng và đóng góp của họ trong công việc và do đó có thể tạo ra mức lương cao hơn. Tuy nhiên không thể chỉ dựa vào giá trị bằng cấp đó mà cho rằng bằng cấp ảnh hưởng lớn đến tiền lương được. Vì đi đôi với bằng cấp **(Degree)** còn phải xem tấm bằng ấy là từ trường Đại học nào (**CollegeTier**), nơi mà trường Đại học ấy đang ở **(CollegeCityTier)** thế nên ở phương pháp **Backward Elimination** ta đã loại đi **CollegeCityTier** rồi nên sẽ làm ảnh hưởng rất nhiều

đến **Degree**, làm tầm ảnh hưởng của nó đến **Salary** giảm đi hoặc bị phân tán, đây cũng chính là một **điểm yếu chí mạng** của phương pháp này (không thể đánh giá được mối tương quan giữa các đặc trưng). Tuy nhiên suy cho cùng thì nếu chỉ xét riêng **Degree** mà nói nó có ảnh hưởng lớn đến Salary thì vẫn không đúng, do ta đã chấp nhận loại bỏ **CollegeCityTier** nên việc loại bỏ **Degree** ra khỏi mô hình vẫn là chấp nhận được vì bây giờ tầm ảnh hưởng của **Degree** đã giảm hoặc bị sai lệch đi rất nhiều rồi.

- Cuối cùng là **MechanicalEngg**, điểm số của phần kĩ thuật cơ khí có thể có ảnh hưởng đến mức lương của một kỹ sư, nhưng không phải là yếu tố quyết định duy nhất. Tuy nhiên nếu nhìn vào dữ liệu ta sẽ thấy có rất nhiều giá trị '-1', chứng tỏ rất nhiều sinh viên trong bộ dữ liệu này không thực sự thích thứ với kĩ thuật cơ khí mà thay vào đó họ lại chú tâm hơn vào các mảng khác hiện đại, có sức hút hơn như **ComputerProgramming, ElectronicsAndSemicon, ComputerScience.** Vì thực sự **MechanicalEngg** rất kén người học (nhất là nữ), đó chính là lí do đặc trưng này chứa rất nhiều giá trị vô nghĩa '-1' làm cho đặc trưng này gần như không có ảnh hưởng quá lớn hay có thể nói là vô hại đến **Salary** nên việc loại đặc trưng này khỏi mô hình là một việc nên làm để giúp mô hình của ta trở nên tốt và chính xác hơn trong việc dự đoán lương.

V. References

[1] S. S. J. T. s.-d. Josef Perktold, "statsmodels.org," 5 5 2023. [Online]. Available: https://www.statsmodels.org/stable/. [Accessed 23 8 2023].

[2] J.-A. Min, "The Personality Traits That Increase Your Salary," 13/5/2015.

[3] M. Mauro, "Neurotic Personalities Earn Lower Salaries," *Psychology Today*, 2010.

[4] Unknown, "What is the average salary of a Quant in US/UK and Europe as a beginner and how does it vary with experience?," in *Quora*.

[5] B. Foltz, "Youtube," 10 8 2020. [Online]. Available: https://www.youtube.com/watch?v=-inJu1jHqb8. [Accessed 19 8 2023].

[6] "simplilearn," [Online]. Available: https://www.simplilearn.com/what-is-backward-elimination-technique-in-machine-learning-article#:~:text=Backward%20elimination%20is%20a%20method,is%20removed%20from%20the%20model.. [Accessed 19 8 2023].

[7] unknown, Composer, *Backward Elimination.* [Sound Recording].

[8] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/P-value. [Accessed 19 8 2023].

[9] T. v. p. luật, "Bảng tra cứu lương tối thiểu vùng 2023, áp dụng từ 01/01/2023," Thư viện Pháp Luật, TP Hồ Chí Minh, 2023.

[10] Unknown, "Dân số Việt Nam," DanSo.org, 2023.

HẾT