

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

****

**BÀI TẬP CHƯƠNG 3**

**Giảng Viên: Nguyễn Thị Anh Thư**

**Mã Lớp: CS313.O21**

**Nhóm 04**

| *Trương Văn Khải* | *21520274* |
| --- | --- |
| *Đoàn Nhật Sang* | *21522542* |
| *Lê Ngô Minh Đức* | *21520195* |
| *Lê Minh Quang* | *21522510* |
| *Lê Yến Nhi* | *21522427* |
| *Hoàng Thị Mỹ Hạnh* | *21522044* |
| *Hoàng Tiến Đạt* | *21520696* |

**TP. Hồ Chí Minh, 3/2024**

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................ ............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................ .................................................................................................................................... .................................................................................................................................... .................................................................................................................................... .................................................................................................................................... ...................................................................................................................................

**PHÂN CÔNG CÔNG VIỆC**

| **Chi tiết công việc** | | **Trương Văn Khải (NT)** | **Đoàn Nhật Sang** | **Lê Ngô Minh Đức** | **Lê Minh Quang** | **Lê Yến Nhi** | **Hoàng Thị Mỹ Hạnh** | **Hoàng Tiến Đạt** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Phân công** | |  |  |  |  |  |  |  |
| **Thống kê mô tả** | |  |  |  |  |  |  |  |
| **Trực quan hóa dữ liệu** | **2.2.1 2.2.2** |  |  |  |  |  |  |  |
| **2.2.3** |  |  |  |  |  |  |  |
| **Xử lý dữ dữ liệu** | **2.3.1**  **2.3.2** |  |  |  |  |  |  |  |
| **2.3.3**  **2.3.4** |  |  |  |  |  |  |  |
| **Phân tích thống kê** | |  |  |  |  |  |  |  |
| **Khai phá tri thức** | |  |  |  |  |  |  |  |
| **Xử lý dữ liệu trùng lặp** | |  |  |  |  |  |  |  |
| **Quay video** | |  |  |  |  |  |  |  |
| **Format docs** | |  |  |  |  |  |  |  |
| **Kiểm tra** | |  |  |  |  |  |  |  |
| **Mức độ hoàn thành** | | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** |

Bảng 0.1. Phân công công việc

**MỤC LỤC**

[**1. KHÁM PHÁ DỮ LIỆU 1**](#_kfimwbcic6m7)

[1.1. THỐNG KÊ MÔ TẢ: 3](#_dd32bnmmio95)

[1.1.1. Tính toán các thống kê cơ bản như trung bình, trung vị, độ lệch chuẩn, tứ phân vị, v.v 3](#_4m2k72qo5o0d)

[1.1.2. Phân tích phân phối dữ liệu bằng cách sử dụng biểu đồ histogram, boxplot, v.v. 3](#_ssqssrpwrnu7)

[1.1.3. Xác định các giá trị ngoại lai (outlier) có thể ảnh hưởng đến kết quả phân tích: 5](#_vygt5qgggs01)

[1.2. TRỰC QUAN HÓA DỮ LIỆU: 6](#_77eytik4kw3z)

[1.2.1. Sử dụng các biểu đồ như scatter plot, heatmap, line plot, v.v. để thể hiện mối quan hệ giữa các biến: 6](#_8obb8k8ei93j)

[1.2.2. Tạo các dashboard trực quan để theo dõi các chỉ số quan trọng và khám phá các mẫu dữ liệu theo thời gian: 11](#_nwz14bpv6l8r)

[1.2.3. Sử dụng các kỹ thuật như dimensionality reduction để giảm bớt số lượng biến và dễ dàng trực quan hóa dữ liệu: 14](#_oy6ldsm9y8m5)

[1.3. XỬ LÝ DỮ LIỆU: 22](#_aeo6ot34exzs)

[1.3.1. Xử lý dữ liệu thiếu: 22](#_9eu4xnr38umc)

[1.3.2. Xử lý dữ liệu trùng lặp: 23](#_pzq2ds6o8tqq)

[1.3.3. Xử lý dữ liệu nhiễu 23](#_ujcoqagvr6nt)

[1.3.4. Xử lý dữ liệu không nhất quán 30](#_txxd78xw2v)

[1.4. PHÂN TÍCH THỐNG KÊ: 33](#_wwg9euf9pba8)

[1.4.1. Kiểm định giả thuyết: 33](#_4iq9jazbp0zi)

[1.4.2. Tương quan: 38](#_5450v3z94isi)

[1.4.3. Sử dụng mô hình máy học để dự đoán giá trị: 41](#_puxt8j6a3s03)

[1.5. KHAI PHÁ TRI THỨC: 43](#_oj6xkvcc626l)

[1.5.1. Overview 43](#_1q3v9ntngjt3)

[1.5.2. Chuẩn bị Transaction và Encode 44](#_bet6i9xyy3xd)

[1.5.3. Apriori algorithm 45](#_hkifgxkzsfqy)

[1.5.4. Association rule 48](#_huzwyc3x3y1y)

[1.5.5. Nhận xét 51](#_5c5xgo9gzyb0)

[**2. LÀM SẠCH DỮ LIỆU: 51**](#_97tgi7ninbfr)

[2.1. Xử lý dữ liệu trùng lặp: 51](#_pxb5o5pm0koo)

NỘI DUNG

**Link video thuyết trình**: <https://youtu.be/zBsqGR8MbP0>

Bảng dữ liệu sử dụng: problem.json

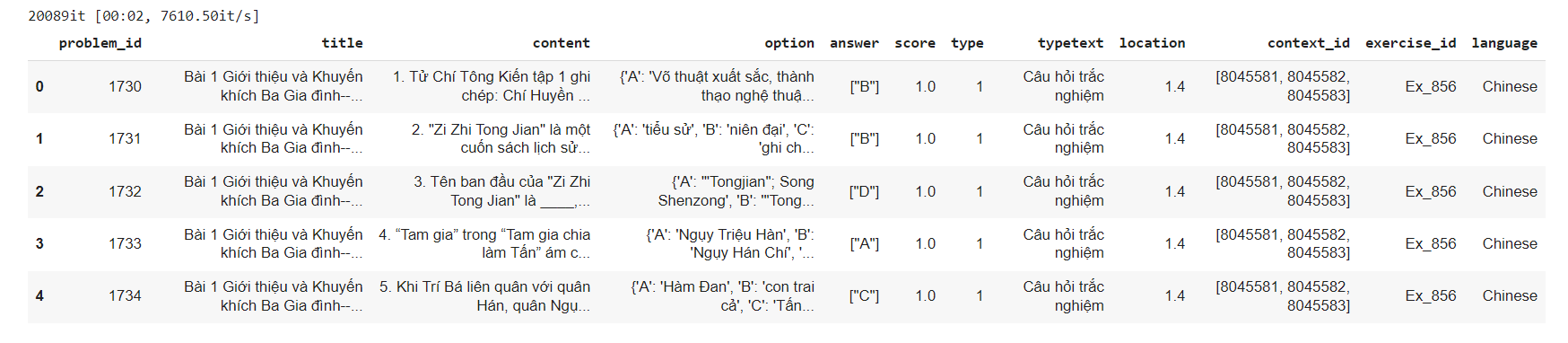
Số lượng: 2454422 mẫu

Ngôn ngữ: chủ yếu là tiếng Trung, có một vài mẫu có tiếng Anh

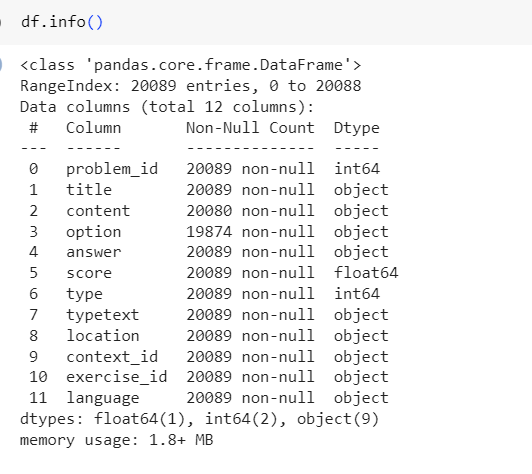
Nhóm sử dụng Google Translate API thông qua package deep\_translator (Python) để dịch dữ liệu sang tiếng Việt. Do lượng dữ liệu trong problem.json rất nhiều, nên khi dịch sẽ tốn rất nhiều thời gian. Vì vậy, nhóm chỉ sử dụng một phần dữ liệu để làm mẫu. Cụ thể, nhóm sẽ lấy 20000 mẫu đầu tiên trong bảng này để thực hiện.

# 1. KHÁM PHÁ DỮ LIỆU

Trước khi khám phá dữ liệu, chúng tôi đã thực hiện một số quan sát cơ bản về dữ liệu như sau:

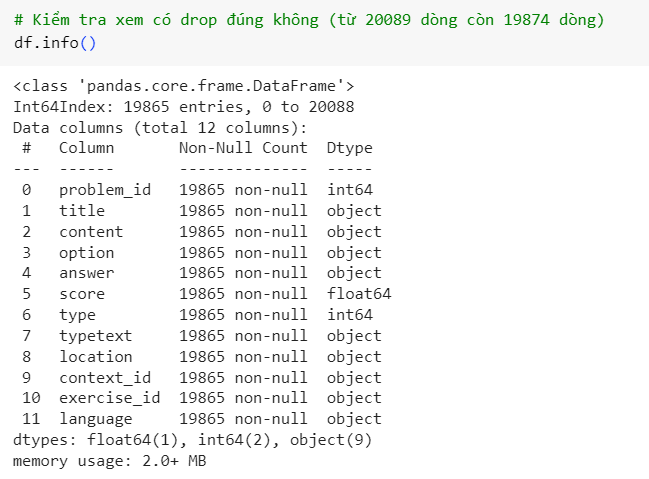


Hình 1.1. Entities/problem.json



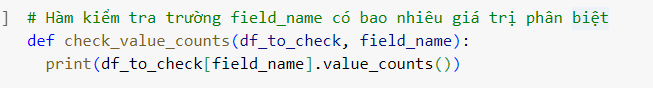
Hình 1.2. Thông tin chung của dữ liệu

Dễ thấy, dữ liệu có 20089 dòng, trong đó trường **option** và **content** bị null ở một số chỗ. Ở những dòng này, câu hỏi (content) được đặt ra, ta có đáp án (answer) nhưng lại không biết nó là gì do không có option, và người làm sạch dữ liệu cũng không có cách nào trả lời và khôi phục lại các options ban đầu cho câu hỏi này, do đó ta sẽ bỏ đi các hàng không có option bởi vì những câu hỏi này không có câu trả lời, do đó không cần thiết phải tìm cách điền dữ liệu giả vào để thay thế. Xử lý tương tự với **content**.



Hình 1.3. Thông tin chung của dữ liệu sau khi bỏ các dòng null ở trường option và content

Tiếp theo, để nhận biết đặc điểm của từng trường, ta sử dụng hàm *check\_value\_counts* như sau:



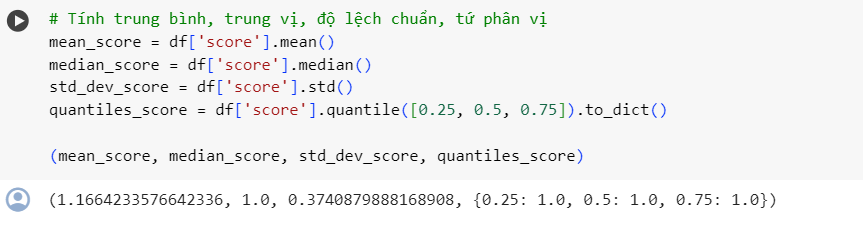
Hình 1.4. Hàm kiểm tra số lượng giá trị phân biệt cho trường bất kì

Sau khi thay các trường typetext, type, title, score, location, exercise\_id, language chúng tôi nhận thấy các trường **type**, **typetext, score** và **language** là dữ liệu dạng CATEGORICAL. Bây giờ, chúng tôi bắt đầu khám phá dữ liệu theo tuần tự các bước như ở dưới:

## 1.1. THỐNG KÊ MÔ TẢ:

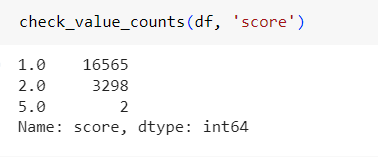
Đầu tiên chúng tôi tiến hành thực hiện trực quan trên bộ dữ liệu **entities/problem.json**, và tiến hành một số thao tác cơ bản:

### 1.1.1. Tính toán các thống kê cơ bản như trung bình, trung vị, độ lệch chuẩn, tứ phân vị, v.v

****

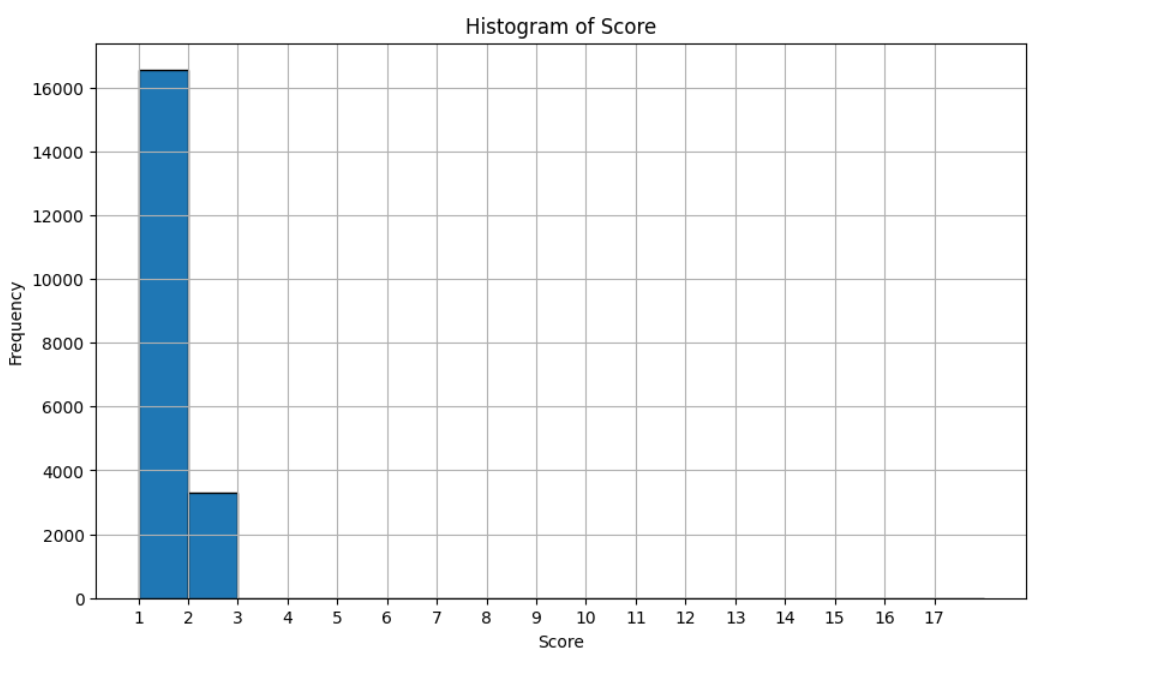
Hình 1.5. Các thống kê cơ bản của bảng dữ liệu entities/problem.json

Như ta thấy, lower quartile, median và upper quartile đều bằng 1, điều này đơn giản là do giá trị score=1 chiếm rất nhiều trong dữ liệu:

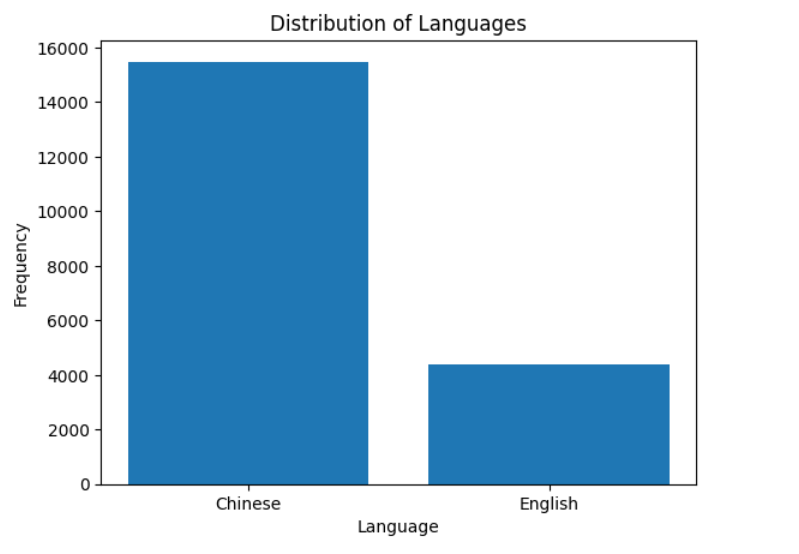


Hình 1.6. Đếm tần số xuất hiện của các giá trị score phân biệt

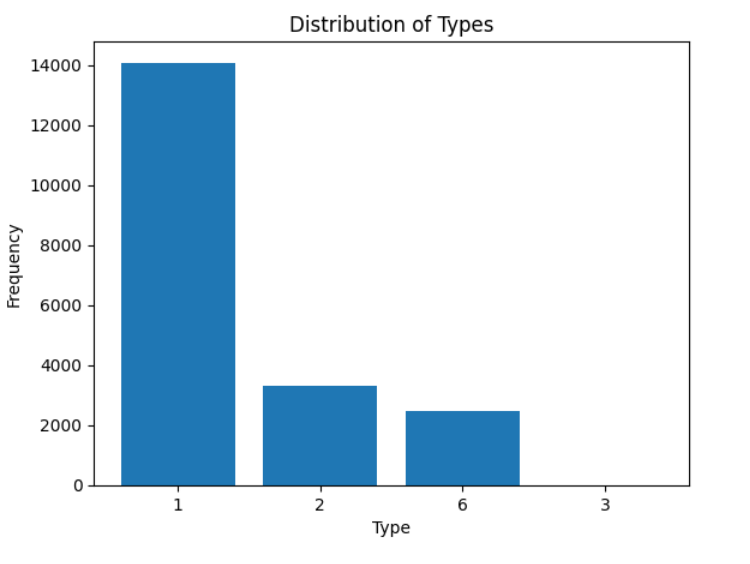
### 1.1.2. Phân tích phân phối dữ liệu bằng cách sử dụng biểu đồ histogram, boxplot, v.v.

Nhằm để chắc chắn việc thống kê là chính xác, chúng tôi đã vẽ histogram cho các trường categorical, cụ thể ở đây chúng tôi xây dựng biểu đồ cho các trường **score**, **language** và **type** như sau:

Hình 1.7. Biểu đồ histogram cho trường score



Hình 1.8. Biểu đồ histogram cho trường language



Hình 1.9. Biểu đồ histogram cho trường type

**Nhận xét:**

* Số câu hỏi có giá trị score bằng 1 chiếm hầu hết câu hỏi (problem) trong dữ liệu, tuy nhiên sự mất cân bằng này không ảnh hưởng tới ý nghĩa của dữ liệu, do số điểm sẽ tương ứng với số câu trả lời đúng trong đáp án của câu hỏi đó (ví dụ: câu hỏi trắc nghiệm 2 đáp án sẽ có số điểm tối đa là 2 thay vì 1 như thông thường).
* Số câu hỏi tiếng Trung chiếm phần lớn dữ liệu so với tiếng Anh, điều này hiển nhiên vì nền tảng này là trang web dạy học của nội địa Trung Quốc.
* Các loại câu hỏi cũng có phân bố khá chênh lệch, với phần lớn là loại 1 (câu hỏi trắc nghiệm 1 hoặc nhiều đáp án) so với loại 2 (đúng hay sai), 3 (điền vào chỗ trống) và 6 (câu hỏi chủ quan). Tuy nhiên tất cả các câu hỏi này đều nằm ở hình thức trắc nghiệm nên cũng không ảnh hưởng đến ý nghĩa của dữ liệu.

### 1.1.3. Xác định các giá trị ngoại lai (outlier) có thể ảnh hưởng đến kết quả phân tích:

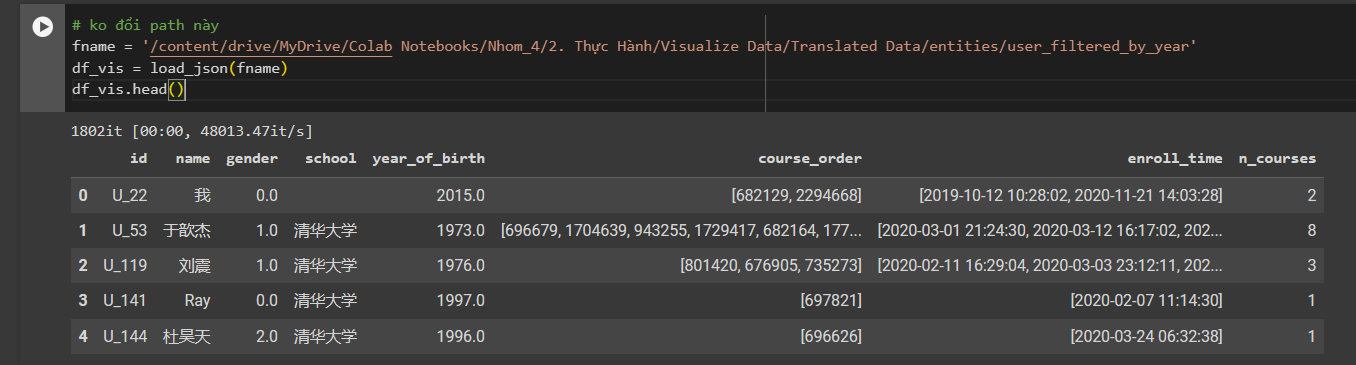
Sau khi thống kê sơ bộ, chúng tôi thấy ở bảng dữ liệu problem.json chỉ có trường **score** là có ý nghĩa số học (do có kiểu dữ liệu float64) trong khi các dữ liệu còn lại là chuỗi hoặc là categorical, do đó các trường này không có giá trị ngoại lai. Hơn nữa, như đã nói ở trên, mặc dù trường score có sự mất cân bằng trong phân bố, tuy nhiên các giá trị thiểu số (score bằng 2) không ảnh hưởng đến dữ liệu do nó phụ thuộc vào tính chất của câu hỏi, và do đó không thể xem như giá trị ngoại lai (outlier).

Do đó, để thực hiện đầy đủ các thao tác xử lí dữ liệu, chúng tôi chuyển sang sử dụng bảng dữ liệu **entities/user.json** do nó chứa nhiều trường kiểu số hơn, tức là có thể dễ dàng sử dụng các phương pháp toán học để trực quan hóa.

## 1.2. TRỰC QUAN HÓA DỮ LIỆU:

### 1.2.1. Sử dụng các biểu đồ như scatter plot, heatmap, line plot, v.v. để thể hiện mối quan hệ giữa các biến:

Chúng tôi tiến hành thực hiện trực quan trên bộ dữ liệu **entities/user.json**, tuy nhiên chúng tôi đã có một chút biến thể để phù hợp với bài toán của tôi ở mục 1.4.



Hình 1.10. Bảng dữ liệu entities/user.json được lọc theo năm.

Bây giờ, chúng tôi sẽ thực hiện dùng các biểu đồ để biểu diễn mối quan hệ giữa năm sinh (year\_of\_birth) và số khóa học đăng ký (n\_courses).

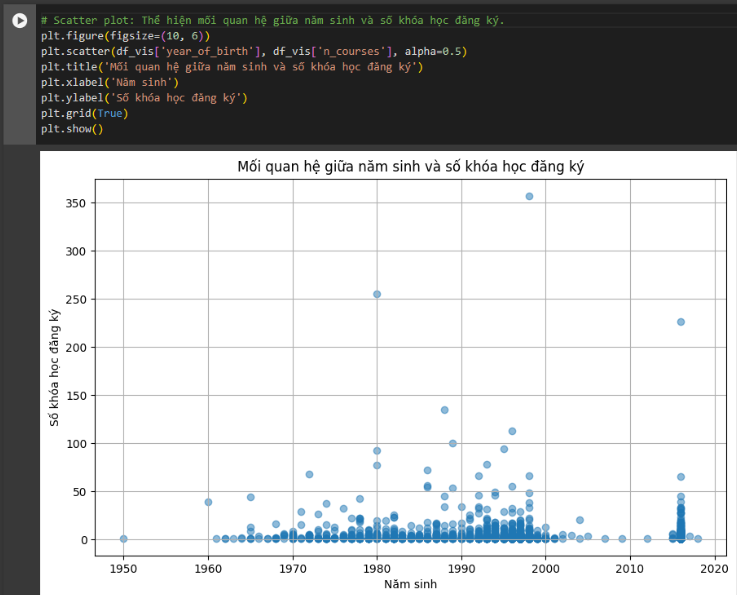
* **Scatter plot** để thể hiện mối quan hệ giữa năm sinh và số khóa học đăng ký:

**Chức năng**: Tạo ra biểu đồ phân tán từ dữ liệu được cung cấp. Hiển thị mối quan hệ giữa hai hoặc nhiều biến.

**Ứng dụng**: Thường được sử dụng để hiển thị mối quan hệ giữa hai biến trong dữ liệu. Hữu ích trong việc phát hiện các xu hướng, mối quan hệ và ngoại lai trong dữ liệu.

**Cú pháp**: plt.scatter() của thư viện matplotlib

**Code**:



Hình 1.11. Hình vẽ biểu diễn mới quan hệ giữa năm sinh và số khóa học đăng ký thông qua hàm Scatter Plot.

**Nhận xét**:

Mối tương quan: Biểu đồ phân tán cho thấy mối tương quan dương giữa năm sinh và số khóa học đăng ký. Nhìn chung, những người sinh trước (kể từ năm 2000 trở về trước) có xu hướng đăng ký nhiều khóa học hơn những người sinh trước.

Sự phân bố: Hầu hết các điểm dữ liệu nằm trong khu vực có số khóa học đăng ký từ 50 đến 250. Có một số điểm dữ liệu nằm ngoài khu vực này, với số khóa học đăng ký cao hơn hoặc thấp hơn đáng kể.

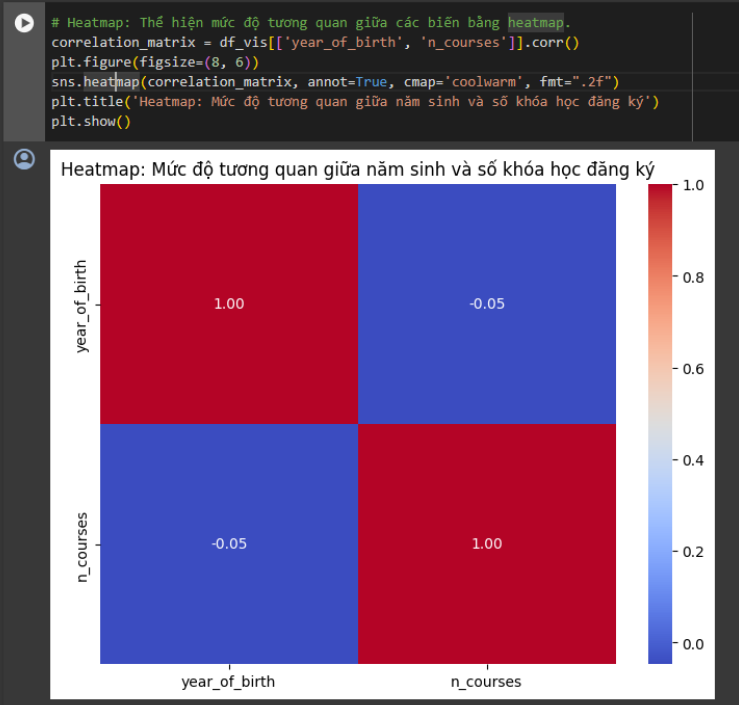
* **Heatmap** thể hiện mức độ tương quan giữa các biến bằng heatmap.

**Chức năng**: Tạo biểu đồ heatmap từ dữ liệu ma trận. Cho phép hiển thị sự biến đổi, tương quan, hoặc phân bố của dữ liệu dưới dạng màu sắc trên một lưới ô.

**Ứng dụng**: Thường được sử dụng trong phân tích dữ liệu để hiển thị mức độ tương quan giữa các biến.

**Cú pháp**: heatmap() của thư viện sns

**Code**:



Hình 1.12. Hình vẽ biểu diễn mới quan hệ giữa năm sinh và số khóa học đăng ký thông qua hàm Heatmap.

**Nhận xét**:

Giá -0.05 cho thấy có một mối quan hệ rất nhỏ hoặc gần như không liên quan giữa năm sinh và số khóa học đăng ký.

Giữa năm sinh và năm sinh, hay số khóa học đăng ký với nhau thì nó tương đồng 100%, nên giá trị ở đây là 1.00.

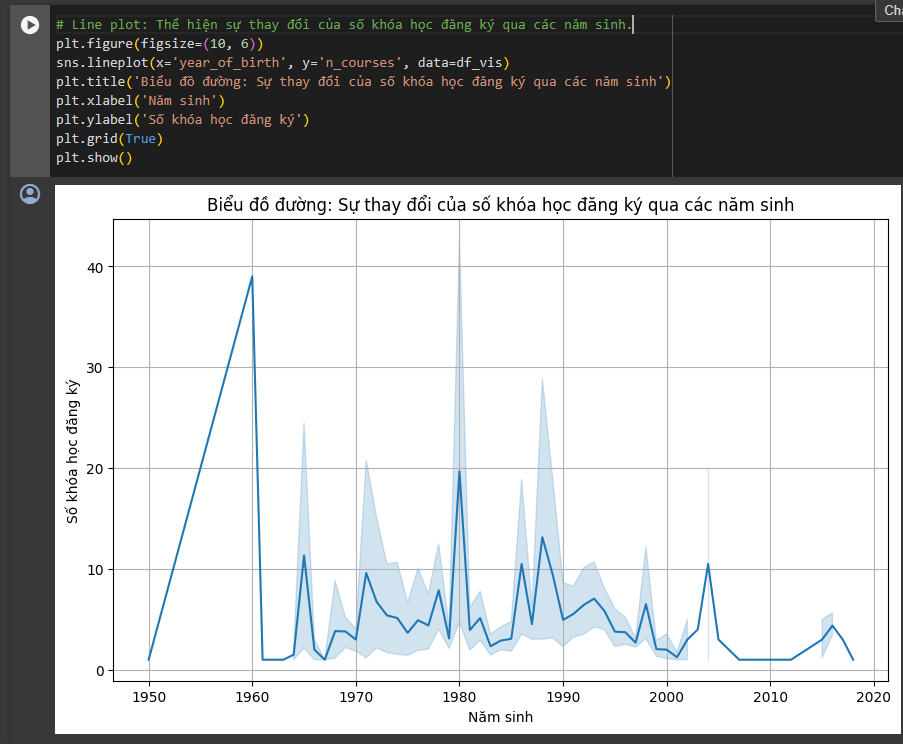
* **Line plot** thể hiện sự thay đổi của số khóa học đăng ký qua các năm sinh.

**Chức năng**: được sử dụng để tạo biểu đồ đường. Nó tạo ra một biểu đồ thể hiện mối quan hệ giữa các biến dữ liệu dạng chuỗi (ví dụ: thời gian) và một biến số liên tục (ví dụ: giá trị).

**Ứng dụng**: Hiển thị sự biến thiên của một biến liên tục qua thời gian hoặc không gian. So sánh xu hướng giữa các nhóm dữ liệu khác nhau. Phát hiện sự biến động, xu hướng hoặc mối quan hệ giữa các biến.

**Cú pháp**: lineplot() của thư viện sns

**Code**:



Hình 1.13. Hình vẽ biểu diễn mới quan hệ giữa năm sinh và số khóa học đăng ký thông qua hàm Line Plot.

**Nhận xét**:

Số khóa học đăng ký có xu hướng biến động nhiều và không ổn định theo thời gian.

Có sự gia tăng đột biến số khóa học đăng ký từ năm sinh 1950 - 1960, 1980.

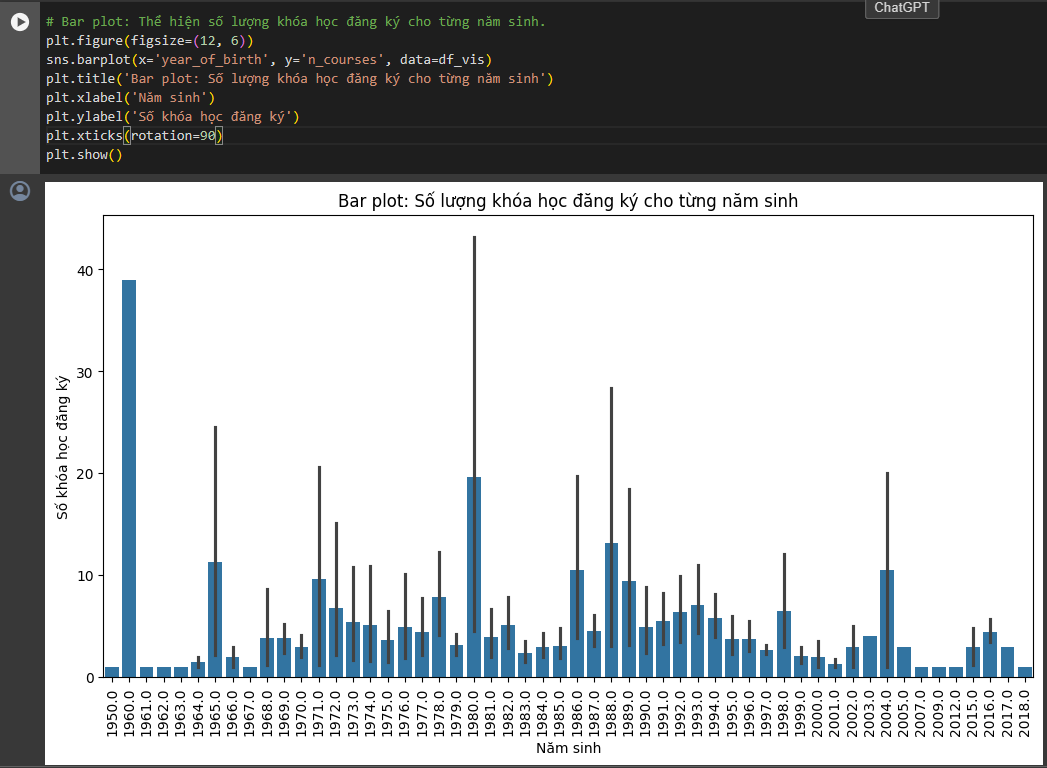
* **Bar plot** thể hiện số lượng khóa học đăng ký cho từng năm sinh.

**Chức năng**: Hiển thị dữ liệu dưới dạng các cột dọc hoặc ngang, với chiều cao (hoặc chiều rộng) của mỗi cột đại diện cho giá trị của biến.

**Ứng dụng**: Dùng để so sánh giá trị trung bình hoặc phân phối của một biến dưới các điều kiện khác nhau.

**Cú pháp**: barplot() của thư viện sns

**Code**:



Hình 1.14. Hình vẽ biểu diễn mới quan hệ giữa năm sinh và số khóa học đăng ký thông qua hàm Barplot.

**Nhận xét**:

Số lượng khóa học đăng ký có xu hướng không ổn định theo thời gian.

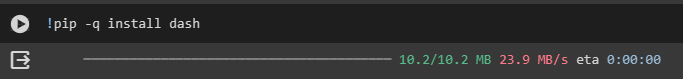
Có sự biến động trong số lượng khóa học đăng ký qua các năm.

### 1.2.2. Tạo các dashboard trực quan để theo dõi các chỉ số quan trọng và khám phá các mẫu dữ liệu theo thời gian:

Dashboard cung cấp một cái nhìn tổng quan và dễ hiểu về các chỉ số quan trọng hoặc các thước đo khác liên quan đến mục tiêu kinh doanh hoặc hoạt động của tổ chức. Việc này giúp cho việc theo dõi và đánh giá hiệu suất trở nên dễ dàng và nhanh chóng hơn. Vì nên, chúng tôi sẽ thử tạo 1 dashboard giản được chạy trên Google Colab để phục vụ mục đích bài tập tìm hiểu này.

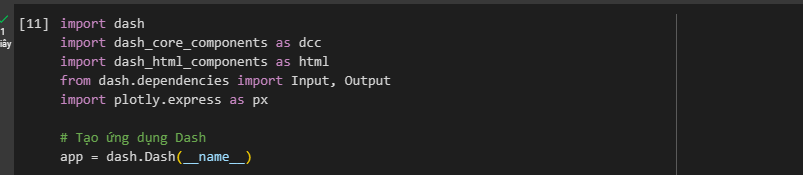
**Code**:

* Cài đặt thư viện dash: Thư viện dash là một framework Python cho việc xây dựng ứng dụng web tương tác trực quan bằng Python mà không cần sử dụng HTML, CSS hoặc JavaScript. Nó kết hợp các thành phần của React.js để tạo ra các ứng dụng web đơn giản, dễ sử dụng.



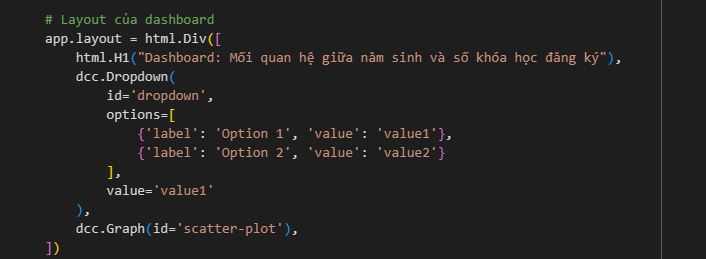
Hình 1.15. Chạy đoạn lệnh để cài đặt thư viện dash.

* Import các thư viện cần thiết và khởi tạo 1 ứng dụng app mới:



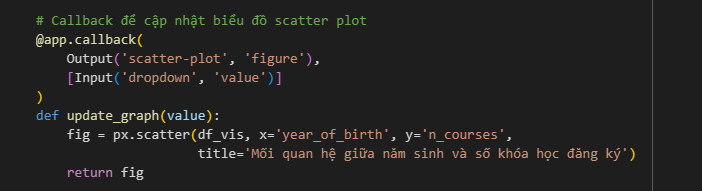
Hình 1.16. Import các thư viện cần thiết và khởi tạo 1 ứng dụng app.

* Định nghĩa Layout và các thành phần HTML, Components của Dashboard:



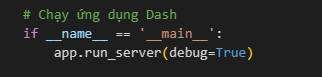
Hình 1.17. Định nghĩa Layout và các thành phần HTML, Components của Dashboard.

* Tạo 1 callback trong Dashboard, Callback là cách Dash kết nối các thành phần của ứng dụng với nhau và cho phép cập nhật nội dung của các thành phần dựa trên sự tương tác của người dùng:

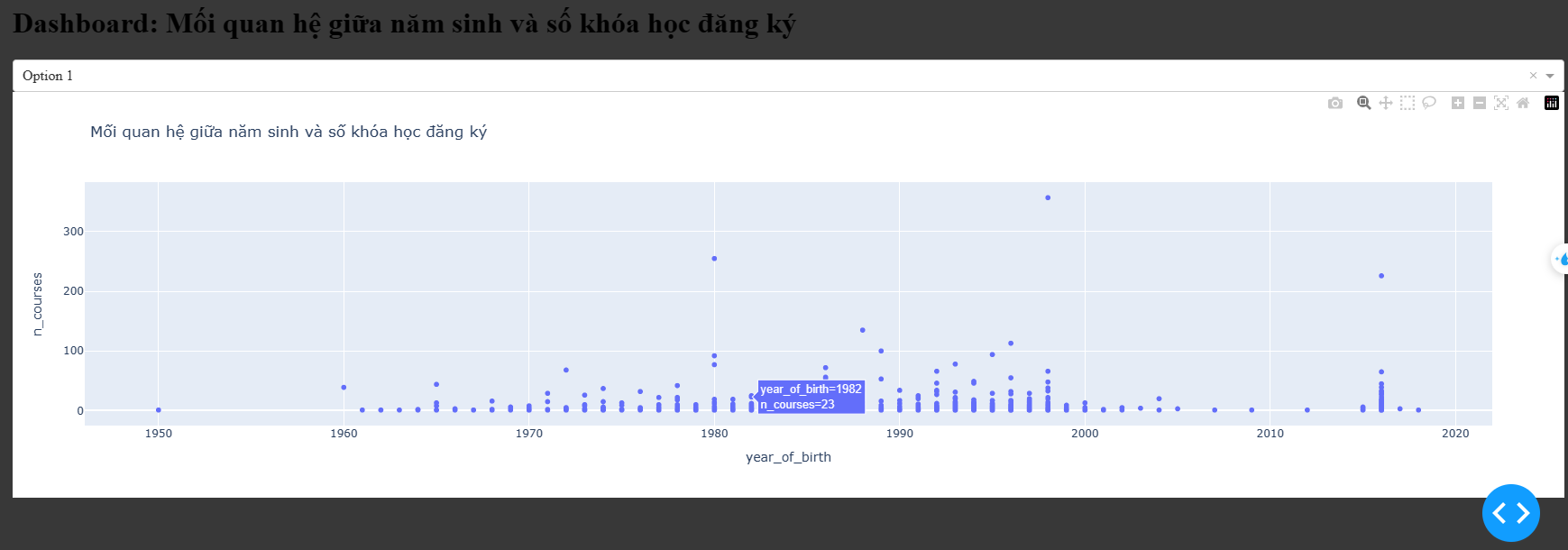


Hình 1.18. Tạo callback để cập nhật biểu đồ scatter plot dựa trên giá trị của dropdown.

* Chạy Dashboard và kết quả đạt được:



Hình 1.19. Code thực thi Dashboard tương tác.

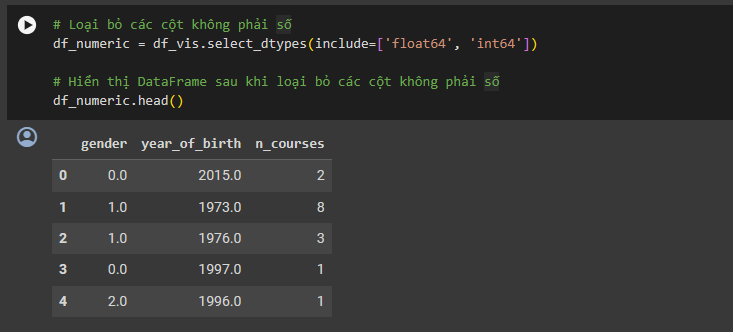


Hình 1.20. Dashboard đạt được và có thể tương tác với người dùng.

### 1.2.3. Sử dụng các kỹ thuật như dimensionality reduction để giảm bớt số lượng biến và dễ dàng trực quan hóa dữ liệu:

Phương pháp giảm chiều dữ liệu là quá trình giảm số chiều của bộ dữ liệu bằng cách chọn ra một tập hợp con của các biến hoặc tính năng (features) hoặc tạo ra các biến mới từ các biến ban đầu. Mục tiêu của việc giảm chiều dữ liệu là giữ lại các thông tin quan trọng trong dữ liệu ban đầu trong khi giảm độ phức tạp của dữ liệu.

Ở bài này, chúng tôi thử sử dụng các phương pháp giảm chiều dữ liệu như PCA (Principal Component Analysis), SVD (Singular Value Decomposition) và t-SNE (t-distributed Stochastic Neighbor Embedding). Các phương pháp này thường được áp dụng với dữ liệu cần phải ở dạng số, không thể có các cột dữ liệu không phải số như 'name', 'school' và 'enroll\_time'. Trong trường hợp này, chúng tôi sẽ bỏ một số cột dữ liệu không phải số như 'name', 'school' và 'enroll\_time' trước khi áp dụng PCA, SVD hoặc t-SNE. Kết quả thu được của quá trình lọc bỏ trên như sau:



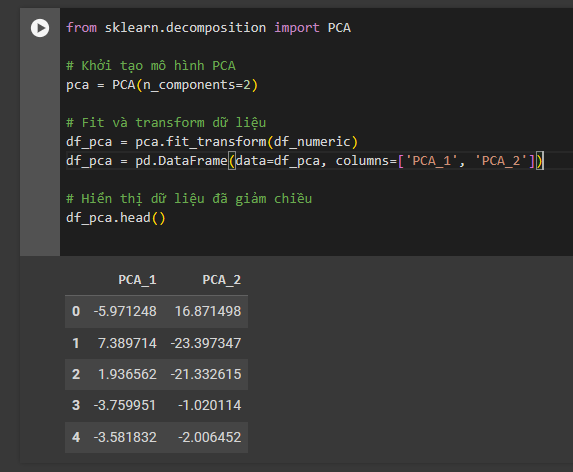
Hình 1.21. Dữ liệu thu được sau khi xử lý để áp dụng thử các phương pháp giảm chiều dữ liệu.

#### 1.2.3.1. PCA (Principal Component Analysis) - Phân tích thành phần chính:

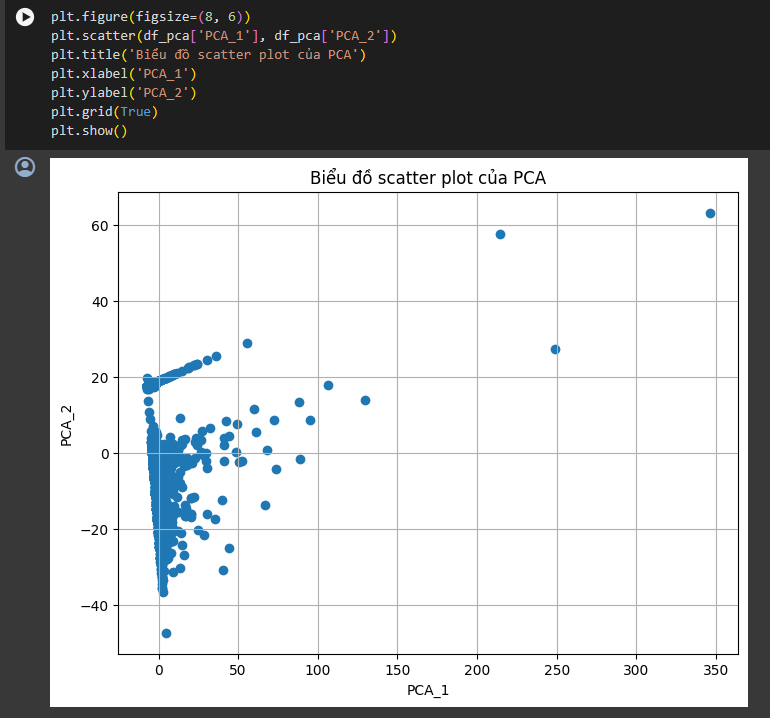
**Chức năng**: Phương pháp này dựa trên quan sát rằng dữ liệu thường không phân bố ngẫu nhiên trong không gian mà thường phân bố gần các đường/mặt đặc biệt nào đó. PCA xem xét một trường hợp đặc biệt khi các mặt đặc biệt đó có dạng tuyến tính là các không gian con (subspace).

**Cú pháp**: PCA() của thư viện sklearn.decomposition

**Code**: Chúng tôi tiến hành phương pháp PCA để giảm chiều dữ liệu từ 3 cột thuộc tính xuống còn 2.



Hình 1.22. Đoạn code PCA giảm chiều dữ liệu xuống còn 2 và kết quả dữ liệu thu được dưới dạng bảng.



Hình 1.23. Kết quả dữ liệu thu được sau khi được biểu diễn trên miền không gian 2 chiều sau khi áp dụng PCA.

**Nhận xét**:

* Phân bố dữ liệu: Các điểm dữ liệu phân bố khá hẹp, có thể thấy rằng các điểm có giá trị PCA\_1 và PCA\_2 nằm trong một phạm vi hẹp và lệch về phía bên trái,trải từ giá trị âm đến dương.
* Phân bố tương đối của các điểm: Có thể thấy rằng các điểm dữ liệu không phân bố đều trên không gian 2 chiều sau khi giảm chiều. Điều này có thể phản ánh sự biến động lớn trong dữ liệu gốc và việc giảm chiều không thể giữ lại toàn bộ thông tin về phân bố của dữ liệu ban đầu.

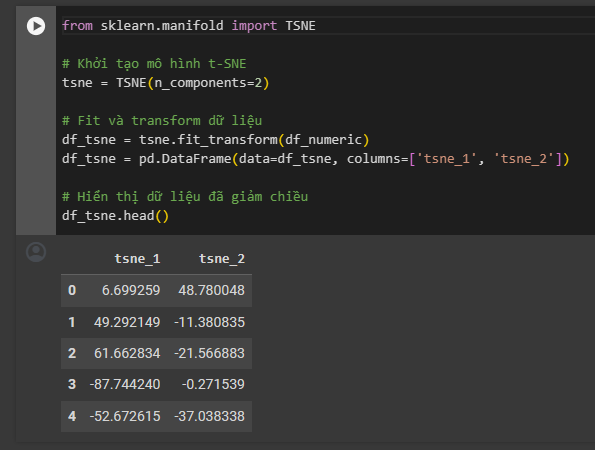
=> Có thể, các nguyên nhân trên đều có liên quan đến việc dữ liệu ở các cột thuộc tính "gender", "year\_of\_birth", "n\_courses" không hề có liên quan hay có sự ảnh hưởng đến nhau.

#### 1.2.3.2. t-SNE (t-distributed Stochastic Neighbor Embedding):

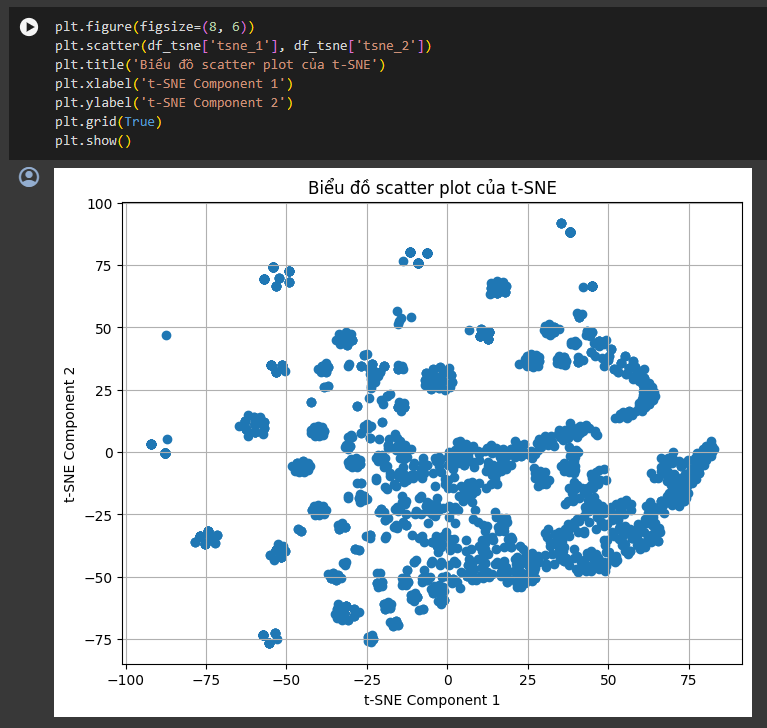
**Chức năng**: t-SNE (t-distributed Stochastic Neighbor Embedding) là một kỹ thuật giảm chiều dữ liệu không tuyến tính được sử dụng chủ yếu để trực quan hóa dữ liệu nhiều chiều lên không gian chiều thấp (thường là hai hoặc ba chiều). T-SNE có khả năng tạo ra một biểu đồ phân cụm (cluster) hoặc phân phối của các điểm dữ liệu sao cho các điểm dữ liệu tương đồng trong không gian nhiều chiều cũng sẽ được bảo toàn khoảng cách tương đồng trong không gian chiều thấp.

**Cú pháp:** TSNE() của thư viện sklearn.manifold

**Code**: Chúng tôi tiến hành phương pháp t-SNE để giảm chiều dữ liệu từ 3 cột thuộc tính xuống còn 2.



Hình 1.24. Đoạn code t-SNE giảm chiều dữ liệu xuống còn 2 và kết quả dữ liệu thu được dưới dạng bảng.



Hình 1.25. Kết quả dữ liệu thu được sau khi được biểu diễn trên miền không gian 2 chiều sau khi áp dụng t-SNE.

**Nhận xét**:

* Phân bố dữ liệu: Các điểm dữ liệu được phân bố khá rộng và phân tán trên không gian 2 chiều sau khi giảm chiều bằng t-SNE.
* Phân bố tương đối của các điểm: Các điểm dữ liệu có phân bố không đều trên không gian 2 chiều sau khi giảm chiều bằng t-SNE. Điều này có thể phản ánh sự phức tạp của mối quan hệ giữa các điểm dữ liệu trong dữ liệu gốc.

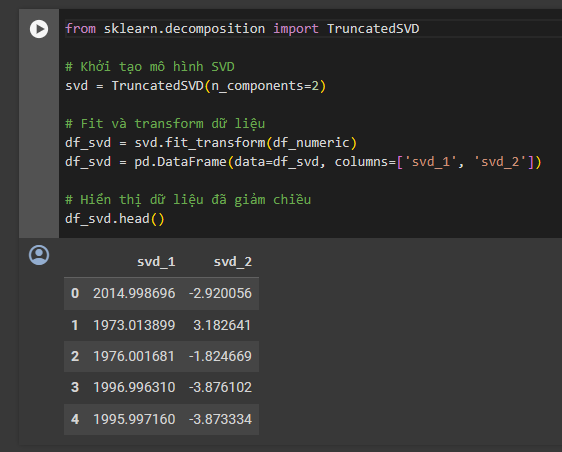
=> Có thể, các nguyên nhân trên đều có liên quan đến việc dữ liệu ở các cột thuộc tính "gender", "year\_of\_birth", "n\_courses" không hề có liên quan hay có sự ảnh hưởng đến nhau.

#### 1.2.3.3. SVD (Singular Value Decomposition):

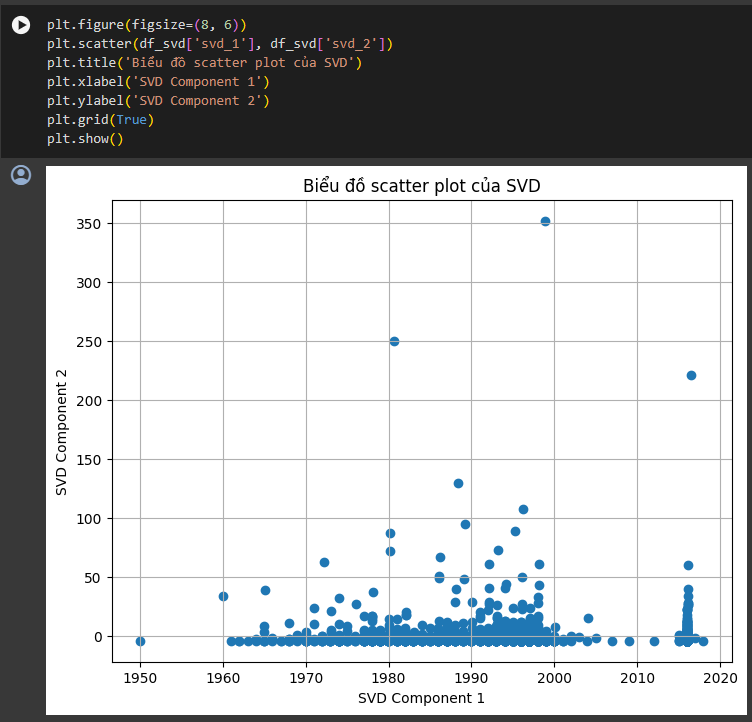
**Chức năng**: SVD có thể được sử dụng để giảm chiều dữ liệu bằng cách loại bỏ hoặc giảm thiểu các singular values nhỏ, từ đó giảm kích thước của ma trận ban đầu trong quá trình biểu diễn dữ liệu.

**Cú pháp**: TruncatedSVD() của thư viện sklearn.decomposition

**Code**: Chúng tôi tiến hành phương pháp t-SNE để giảm chiều dữ liệu từ 3 cột thuộc tính xuống còn 2.



Hình 1.26. Đoạn code SVD giảm chiều dữ liệu xuống còn 2 và kết quả dữ liệu thu được dưới dạng bảng.



Hình 1.27. Kết quả dữ liệu thu được sau khi được biểu diễn trên miền không gian 2 chiều sau khi áp dụng SVD.

**Nhận xét**:

* Phân bố dữ liệu: Các điểm dữ liệu sau khi giảm chiều bằng SVD có phân bố rất khác biệt so với PCA hoặc t-SNE. Cụ thể, giá trị của SVD\_1 chứa thông tin về năm sinh trong khi giá trị của SVD\_2 không có ý nghĩa rõ ràng. Điều này thể hiện sự khác biệt trong cách SVD giảm chiều dữ liệu so với các phương pháp khác.
* Phân bố tương đối của các điểm: Các điểm dữ liệu có phân bố không đều trên không gian 2 chiều sau khi giảm chiều bằng SVD. Các giá trị SVD\_1 phản ánh sự khác biệt lớn giữa các điểm dữ liệu về năm sinh trong khi SVD\_2 không mang lại nhiều thông tin có ý nghĩa.

=> Có thể, các nguyên nhân trên đều có liên quan đến việc dữ liệu ở các cột thuộc tính "gender", "year\_of\_birth", "n\_courses" không hề có liên quan hay có sự ảnh hưởng đến nhau.

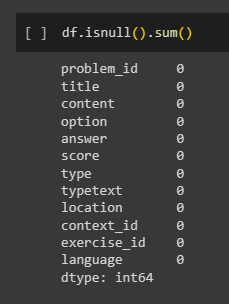
## 1.3. XỬ LÝ DỮ LIỆU:

### 1.3.1. Xử lý dữ liệu thiếu:

**Mô tả**: Kiểm tra các cột có dữ liệu còn thiếu.

**Cú pháp**: dataframe.isnull() của thư viện pandas: kiểm tra giá trị NaN trong DataFrame.

Code:



Hình 1.28. Kết quả trả về sau khi dùng hàm để kiểm tra dữ liệu còn thiếu.

**Nhận xét**:

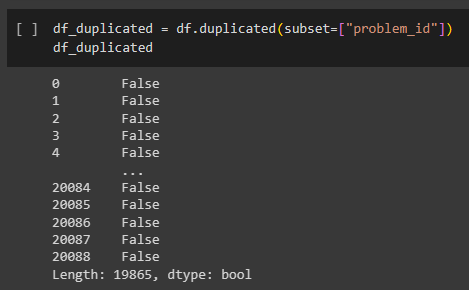
* Vì ở mục 2.1. Thống kê mô tả, chúng tôi đã thực hiện bỏ đi các hàng không có option (những câu hỏi này không có câu trả lời, tức là chúng tôi sẽ không cần phải tìm cách điền dữ liệu giả vào để thay thế). Xử lý tương tự với content.
* Kết quả của df.isnull().sum() cho thấy không có giá trị thiếu (null) nào trong DataFrame. Mỗi cột đều có số lượng giá trị thiếu bằng 0, điều này có nghĩa là mọi cột đều có đầy đủ dữ liệu.

### 1.3.2. Xử lý dữ liệu trùng lặp:

**Mô tả**: Loại bỏ đi các điểm dữ liệu trùng nhau trong dữ liệu.

**Cú pháp**: dataframe.duplicated(): xác định bản ghi trùng lặp trong pandas.

**Code**:



Hình 1.29. Kết quả trả về sau khi dùng hàm để kiểm tra dữ liệu bị trùng lặp.

Nhận xét:

Vì tất cả các giá trị ở mỗi index đều là False nên Dataframe không có chứa bất kỳ giá trị bị trùng lặp.

### 1.3.3. Xử lý dữ liệu nhiễu

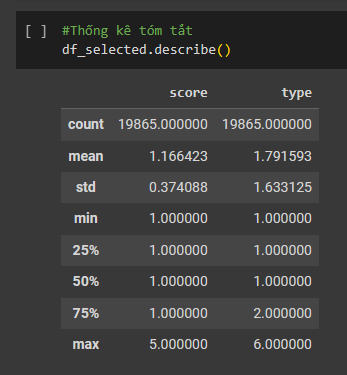
Mô tả: Xử lý dữ liệu nhiễu là quá trình loại bỏ hoặc giảm thiểu ảnh hưởng của các điểm dữ liệu không chính xác, không đáng tin cậy hoặc không phù hợp trong bộ dữ liệu.

#### 1.3.3.1. Xử lý dữ liệu nhiễu ở dạng số:

**Mô tả**: Dữ liệu ở dạng số thường sẽ là các dữ liệu có kiểu dữ liệu thuộc [int32, int64, float32, float64, …]. Xử lý dữ liệu nhiễu ở dạng số thường sẽ là lọc các Outliers để giảm thiểu sự ảnh hưởng của các điểm dữ liệu này hay Regularization để tăng tính tổng quát cho mô hình.

**Cú pháp**: dataframe.describe() của thư viện pandas: Thực hiện thống kê tóm tắt về dữ liệu, đưa ra thông tin về giá trị trung bình, độ lệch chuẩn, giá trị tối thiểu, giá trị tối đa và các phần vị.

**Code**:



Hình 1.30. Kết quả trả về sau khi dùng hàm để thống kê mô tả các dữ liệu ở dạng số.

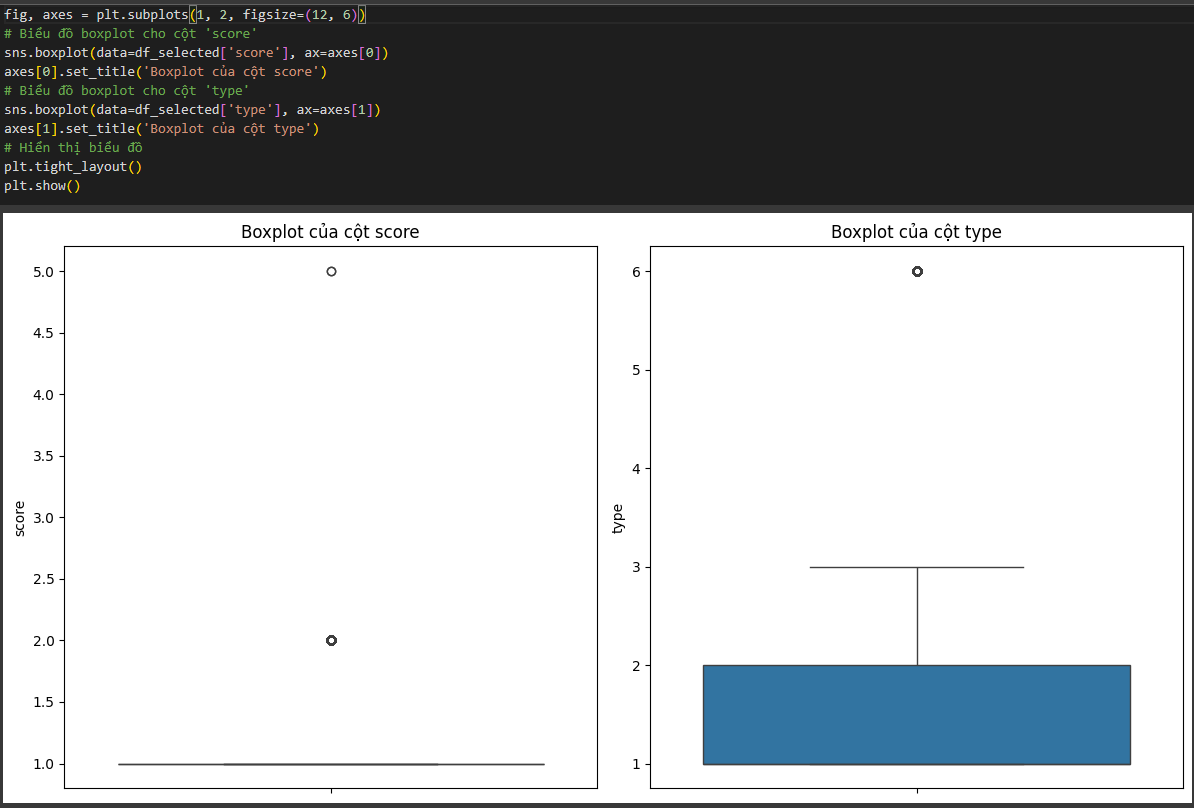
**Nhận xét**:

* Lý do hàm df\_selected.describe() chỉ trả về 2 cột thuộc tính là "score" và "type", trong khi dataframe df\_selected lại có nhiều hơn là vì: describe() chỉ tính toán thống kê mô tả cho các cột dữ liệu số, và cột thuộc tính còn lại các cột thuộc tính có chứa các mẫu dữ liệu ở dạng string.
* Với việc chưa xử lý dữ liệu ngoại lai nên ở cột "score" có giá trị max là 5.0, tương tự cột "type" có giá trị max là 6.0
* **Vẽ biểu đồ Boxplot để biểu diễn sự phân bố dữ liệu khi chưa xử lý Outliers:**

**Mô tả**: Biểu đồ hộp (boxplot) hay còn gọi là biểu đồ hộp và râu (box and whisker plot) là một công cụ trực quan trong thống kê để thể hiện sự phân bố của dữ liệu. Biểu đồ này giúp người xem so sánh các tập dữ liệu khác nhau, đánh giá độ lệch, độ biến thiên và giá trị ngoại lệ của dữ liệu.

**Cú pháp**: boxplot() của sns.

**Code**:



Hình 1.31. Boxplot của dữ liệu ở 2 cột thuộc tính “score” và “type” khi chưa xử lý Outliers.

**Nhận xét**:

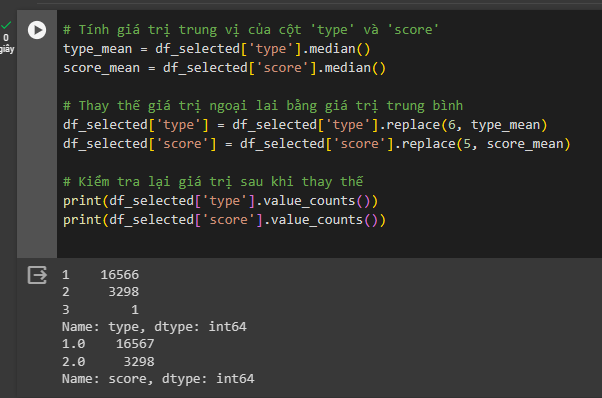
* Dữ liệu của cột score tập trung hơn, phân bố cân xứng và có ít giá trị ngoại lệ hơn so với dữ liệu của cột type.
* Dữ liệu của cột type phân tán hơn, có nhiều giá trị ngoại lệ hơn.
* **Xử lý các giá trị ngoại vi:**

Vì các các giá trị ngoại vi có thể gây ảnh hưởng không tốt đến kết quả mô hình, nên chúng ta có thể lọc hay thay thế các giá trị ngoại vi này để giảm thiểu sự ảnh hưởng đến dữ liệu.

Ở bài này, chúng tôi sẽ thay thế các giá trị ngoại vi bằng giá trị trung vị để xử lý các giá trị ngoại vi này. Lý do chúng tôi sử dụng thay thế các giá trị ngoại vi bằng giá trị trung vi thay vì các giá trị khác là bởi vì: Các mẫu dữ liệu ở hai cột “score”, “type” là các dữ liệu categorical, tức là các giá trị này sẽ được thiết lập cứng cho từng loại mẫu tương ứng, chứ không phải là các mẫu dữ liệu có tính tuyến tính hay tuyến.

**Cú pháp**: dataframe.replace() của thư viện pandas.

**Code**:



Hình 1.32. Thay thế các giá trị ngoại vi lần lượt ở các cột “score”, “type”.

* **Xử lý các miền giá trị:**

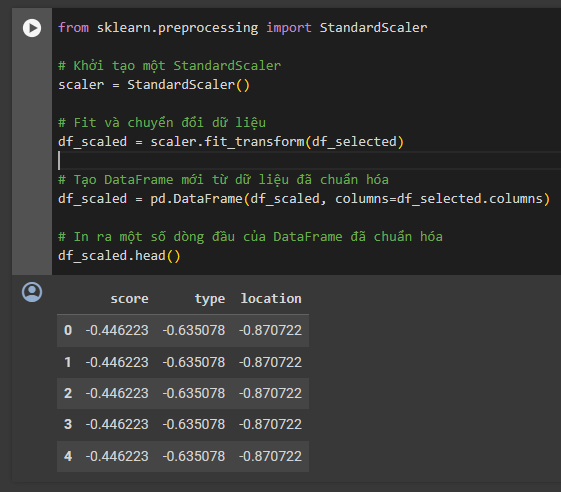
Lý do chúng ta phải xử lý miền giá trị của từng cột dữ liệu là để đảm bảo được sự thống nhất của dữ liệu. Ví dụ như dữ liệu của bạn có hai cột dữ liệu là “diện tích” và “giá nhà”, khi đó diện tích nếu tính theo đơn vị m2 thì chỉ giao động 0-10000, tuy nhiên cột giá nhà nếu tính theo đơn vị đồng thì lại giao động từ 100.000.000 đến 10.000.000.000 đồng. Vì vậy có thể thấy hai miền giá trị này có sự khác nhau rất lớn, và có thể cột “giá nhà” sẽ có ảnh hưởng hơn rất nhiều lần so với với cột “diện tích”. Tóm lại, để có thể đảm bảo sự công bằng của các cột dữ liệu khác nhau thì việc xử lý các miền giá trị là một trong những các tốt nhất để làm điều này.

Hiện nay, có rất nhiều các để xử lý miền giá trị của dữ liệu, ví dụ như: MinMaxScaler(), hay StandardScaler(). Ở bài này chúng tôi sẽ ví dụ thử việc sử dụng StandardScaler().

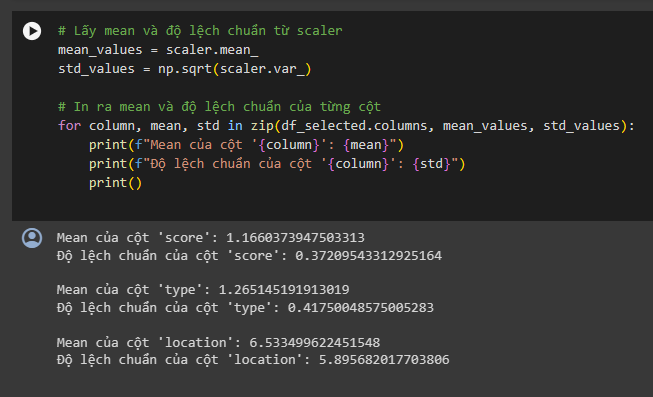
Mô tả: Là một phương pháp chuẩn hóa dữ liệu, chuyển đổi dữ liệu thành dạng có mean=0 và độ lệch chuẩn=1. Giúp giảm thiểu ảnh hưởng của nhiễu và làm cho mô hình dự đoán tốt hơn.

**Cú pháp**: StandardScaler() của thư viện sklearn.preprocessing.

**Code**:



Hình 1.33. Xử lý độ lớn của miền giá trị bằng việc sử dụng hàm StandardScaler().



Hình 1.34. Kiểm tra lại giá trị mean và độ lệch chuẩn của các cột dữ liệu sau khi áp dụng hàm StandardScaler().

#### 1.3.3.2. Xử lý dữ liệu nhiễu dưới dạng chuỗi

Thông thường, việc xử lý dữ liệu nhiễu ở dạng chuỗi thường có các bước như sau:

* Loại bỏ và thay thế ký tự không hợp lệ: Loại bỏ các ký tự không hợp lệ hoặc thay thế chúng bằng các ký tự thích hợp. Ví dụ, trong dữ liệu văn bản, bạn có thể loại bỏ dấu câu hoặc ký tự đặc biệt không cần thiết.
* Sử dụng kỹ thuật mã hóa: Sử dụng kỹ thuật mã hóa như mã hóa one-hot hoặc mã hóa dựa trên tần số để biến đổi dữ liệu chuỗi thành dạng số, từ đó giúp giảm ảnh hưởng của nhiễu.
* Regular expression (Regex): Sử dụng các biểu thức chính quy để loại bỏ hoặc tìm kiếm các mẫu không mong muốn trong dữ liệu chuỗi.
* Lọc và sàng lọc: Sử dụng các kỹ thuật lọc hoặc sàng lọc để loại bỏ nhiễu từ dữ liệu chuỗi, ví dụ như sàng lọc Savitzky-Golay trong xử lý tín hiệu.

Cụ thể, chúng tôi sẽ tiền xử lý dữ liệu dạng chuỗi theo từng bước sau:

* Xoá bỏ các tag html.

VD: “<p class=”comment>Xấu đẹp gì ko biết nhưng rất ưng TGdđ phục vụ rất tuyệt vời<\p>

⇒ “Xấu đẹp gì ko biết nhưng rất ưng TGdđ phục vụ rất tuyệt vời”

* Dùng từ điển teencode để ánh xạ một số từ viết tắt, sai chính tả thông dụng thành từ đầy đủ.

VD: “cx dc" ⇒ “cũng được"

* Dùng tử điển emoji để ánh xạ một số emoji thành từ có đầy đủ nghĩa.

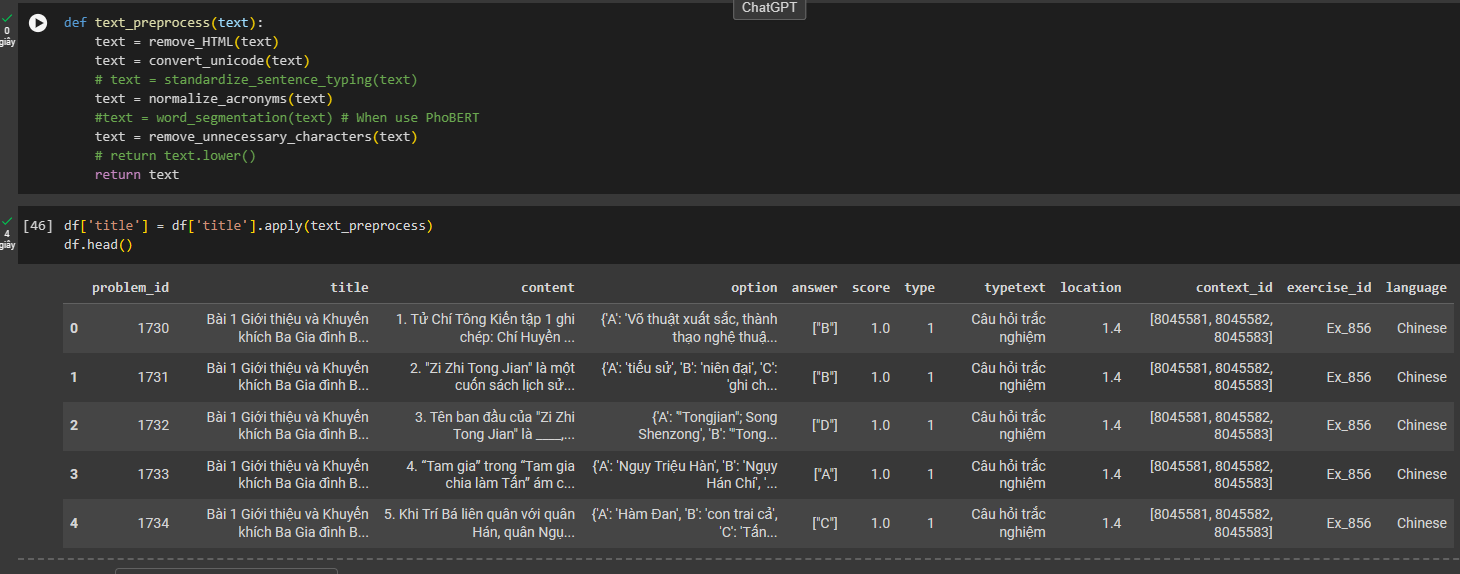
VD: '😔': 'tiêu cực', '😓': 'tiêu cực', '⭐': 'star', '\*': 'star', '🌟': 'star'

* Chuyển đổi các ký tự unicode1252 sang unicode utf8.
* Chuẩn hoá từ theo quy tắc đặt dấu thanh của chữ quốc ngữ kiểu cũ.

VD: gỉa, giả ⇒ giả; kià ⇒ kìa

* Xoá những ký tự không cần thiết như: `, #, @, ~, …

**Code**: Ở đây, chúng tôi sẽ cố gắng tạo 1 hàm text\_preprocess() để xử lý nhiễu ở dữ liệu dạng chuỗi, hàm này sẽ bao gồm các hàm con được xử lý theo đúng pipiline ở trên, sau đó chúng tôi sẽ áp dụng hàm này ở cột “title” ở dữ liệu dataframe của chúng tôi để xử lý nhiễu.



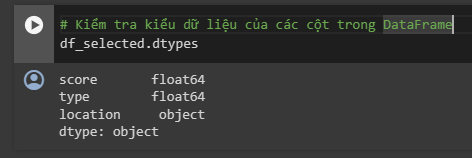
Hình 1.35. Hàm xử lý nhiễu ở dữ liệu dạng chuỗi và kết quả thu được.

### 1.3.4. Xử lý dữ liệu không nhất quán

* **df.dtypes()** của thư viện pandas

**Chức năng**: Kiểm tra kiểu dữ liệu của các cột trong DataFrame.

**Code**:

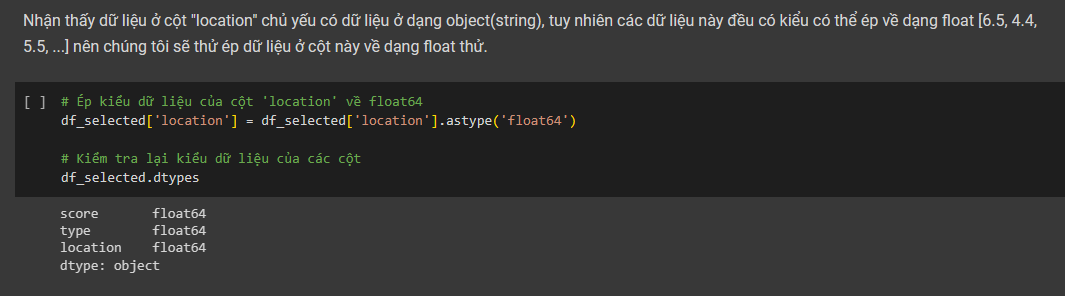


Hình 1.36. Kiểu dữ liệu của các cột trong dataframe sau khi dùng hàm dtypes().

* **df.astype()** của thư viện pandas

**Chức năng**: Chuyển đổi kiểu dữ liệu của các cột thành kiểu dữ liệu khác.

**Code**:



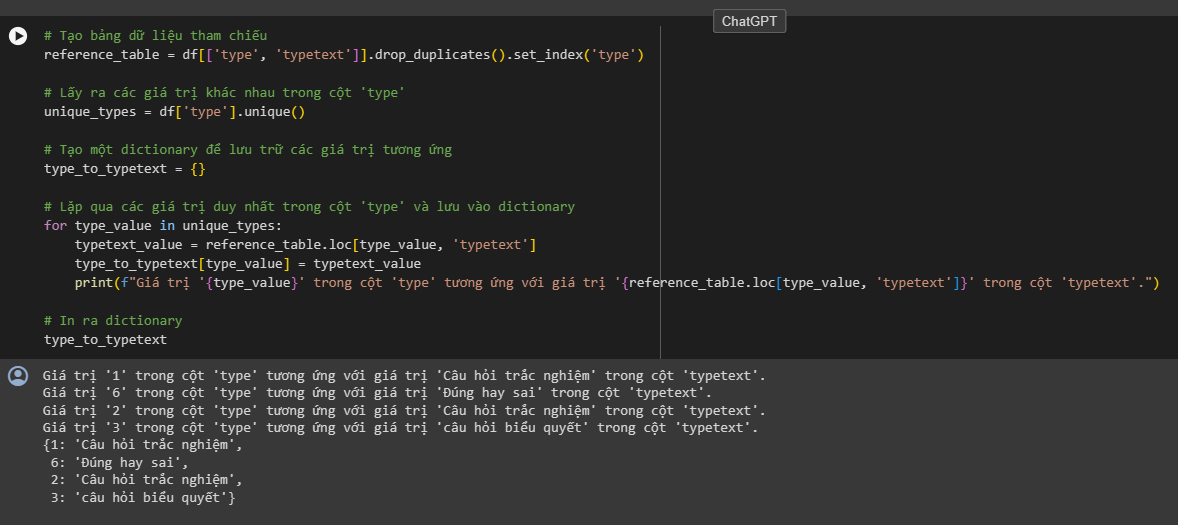
Hình 1.37. Chuyển đổi dữ liệu dạng string ở cột “location” về dạng “float64” bằng cách dùng hàm asypes().

* **dict():**

**Chức năng**: Tạo dictionary để ánh xạ các giá trị không nhất quán thành giá trị hợp lệ.

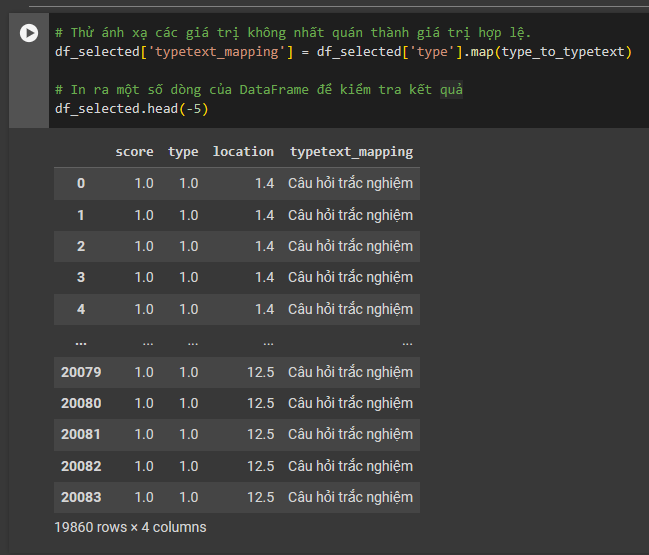
**Code**:

* + Đầu tiên, chúng tôi sẽ tạo 1 dictionary để tạo cơ sở cho việc ánh xạ các dữ liệu. Nhận thấy dataframe có hai cột là “type” (ở dạng số để phân loại các loại câu hỏi) và cột “typetext” (ở dạng string là các loại câu hỏi). Vì vậy, ý tưởng ở đây sẽ là lọc qua lần lượt các dòng dữ liệu để tạo ra được một dictionary với key là dữ liệu dạng số ở cột “type” và ứng với nó là khái niệm, tức là tên của các loại câu hỏi này (ở cột “typetext”). Kết quả thu được như sau:



Hình 1.38. Tạo ra 1 dictionary dựa trên hai cột dữ liệu “type” và “typetext”.

* + Sau đó tiến ánh xạ các mẫu dữ liệu ở cột “type” sang thành một cột dữ liệu mới, lưu ở cột “typetext\_mapping” để kiểm chứng tính đúng sai.



Hình 1.39. Thực hiện ánh xạ từ cột dữ liệu “type” sang một cột dữ liệu mới là “typetext\_mapping”.

## 1.4. PHÂN TÍCH THỐNG KÊ:

### 1.4.1. Kiểm định giả thuyết:

#### 1.4.1.1. Kiểm định t cho 2 mẫu độc lập

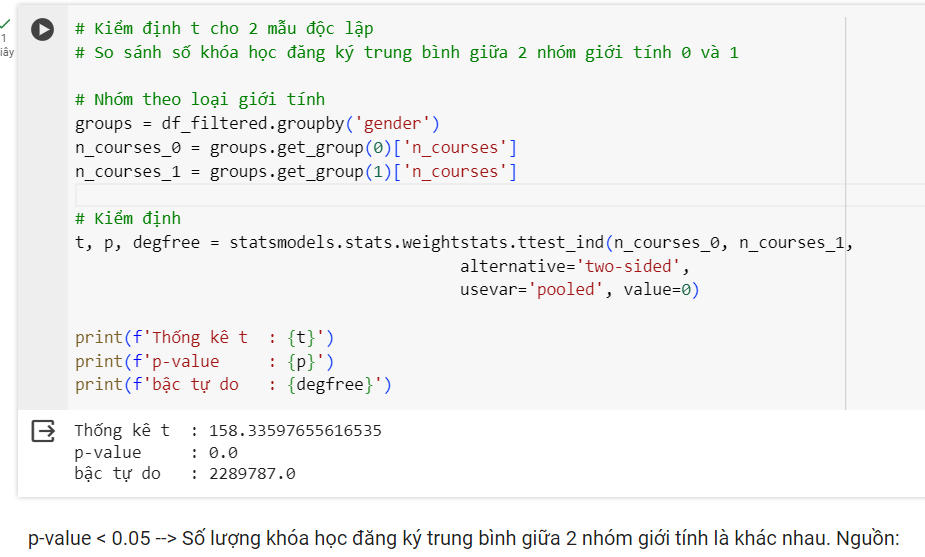
Hàm: statsmodels.stats.weightstats.ttest\_ind

Công dụng: So sánh giá trị trung bình của 2 tổng thể độc lập

Cú pháp: statsmodels.stats.weightstats.ttest\_ind(x1, x2, alternative='two-sided', usevar='pooled', weights=(None, None), value=0)

* Parameters:
  + x1: array\_like, 1-D hoặc 2-D. Thành phần đầu tiên trong 2 mẫu độc lập.
  + x2: array\_like, 1-D hoặc 2-D. Thành phần thứ 2 trong 2 mẫu độc lập
  + alternative: str. Định nghĩa giả thuyết đối, H1 và nhận 1 trong các giá trị sau:
    - ‘two-sided’: Khác biệt giữa các trung bình không bằng value
    - ‘larger’: Khác biệt giữa các trung bình lớn hơn value
    - ‘smaller’: Khác biệt giữa các trung bình nhỏ hơn value
  + uservar: str, ‘pooled’ hoặc ‘unequal’. Nếu pooled, độ lệch chuẩn giữa các mẫu được giả thiết là bằng nhau. Nếu unequal, sử dụng Welch ttest với bậc tự do Satterthwait.
  + weights: tuple hoặc None hoặc ndarrays. Trọng số của 2 mẫu.
  + value: float. Sự sai khác giữa các trung bình trong giả thuyết null.
* Return:
  + tstat: float. Thống kê test
  + pvalue: float. p-value của t-test. Nếu p-value < (thường là 0.05) thì bác bỏ giả thuyết null, nghĩa là chấp nhận giả thuyết đối.  *Ngược lại* *thì không đủ cơ sở để bác bỏ giả thuyết null.*
  + df: int hoặc float. Bậc tự do được sử dụng trong t-test

Demo:



Hình 1.40. Demo cho kiểm định t trên 2 mẫu độc lập

#### 1.4.1.2. Kiểm định t cho 2 mẫu ghép đôi

Hàm: scipy.stats.ttest\_rel

Công dụng: So sánh giá trị trung bình của hai nhóm tổng thể riêng biệt A và B, với đặc điểm mỗi phần tử của tổng thể A có điểm tương đồng theo cặp với mỗi phần tử của tổng thể B. VD: So sánh điểm kiểm tra chất lượng đầu vào của một nhóm học viên sau 3 tháng nhập học so với thời điểm bắt đầu. Điều kiện: 2 biến phải tuân theo phân phối chuẩn.

Cú pháp: scipy.stats.ttest\_rel(*a*, *b*, *axis=0*, *nan\_policy='propagate'*, *alternative='two-sided'*, *\**, *keepdims=False*)

* Parameters:
  + a, b: array\_like. Các arrays phải cùng shape
  + axis: int hoặc None, mặc định là 0.
  + nan\_policy: {‘propagate’, ‘omit’, ‘raise’}. Định nghĩa cách xử lý NaN:
    - propagate: Thành phần tương ứng với NaN trả về NaN
    - omit: Bỏ qua NaN để thực hiện thống kê.
    - raise: Báo lỗi khi có NaN
  + alternative: {‘two-sided’, ‘less’, ‘greater’}, tùy chọn. Định nghĩa giả thuyết đối:
    - ‘two-sided’: trung bình của các phân phối trong các mẫu không bằng nhau.
    - ‘less’: trung bình của phân phối trong mẫu thứ nhất nhỏ hơn trung bình của phân phối trong mẫu thứ hai.
    - ‘greater’: trung bình phân phối trong mẫu thứ nhất lớn hơn trung bình của phân phối trong mẫu thứ hai.
  + keepdims: bool, mặc định là False. Nếu để là True, các trục bị tiêu giảm sẽ được giữ lại trong kết quả như là các chiều với size 1.
* Return: Đối tượng chứa:
  + statistic: float hoặc array. Thống kê t
  + pvalue: float or array. p-value liên quan với giả thuyết đối.
  + df: float hoặc array. Bậc tự do

#### 1.4.1.3. Phân tích phương sai (ANOVA)

Hàm: scipy.stats.f\_oneway

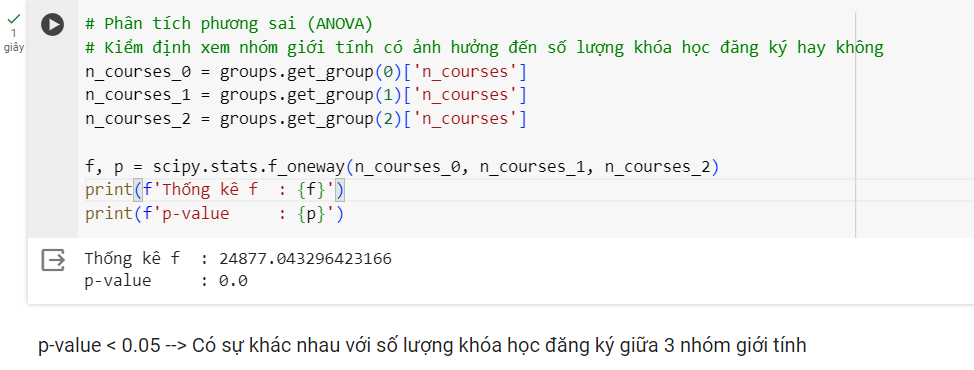
Công dụng: Phương pháp phân tích phương sai ANOVA được dùng khi nghiên cứu ảnh hưởng của biến nguyên nhân định tính lên biến kết quả định lượng, phương pháp này so sánh trung bình của nhiều nhóm (3 nhóm trở lên).Giả thuyết null: biến nguyên nhân định tính không có tác động gì lên kết quả của biến định lượng. Giả thuyết đối: biến nguyên nhân định tính ảnh hưởng đến biến định lượng. ANOVA có các giả thiết quan trọng cần được thỏa mãn để p-value khả dụng:

* Các mẫu phải độc lập.
* Mỗi mẫu được lấy từ quần thể có phân phối chuẩn.
* Độ lệch chuẩn giữa các nhóm phải bằng nhau.

Cú pháp: scipy.stats.f\_oneway(*\*samples*, *axis=0*)

* Parameters:
  + sample1, sample2,...: array like. Các đo lường mẫu cho mỗi nhóm. Phải có ít nhất 2 đối số. Các array có thể có nhiều chiều.
  + axis: int, tùy chọn, mặc định là 0. Axis của các arrays đầu vào mà theo đó kiểm định được áp dụng.
* Return:
  + statistic: float. Thống kê F của kiểm định.
  + pvalue: float. p-value.

Demo:



Hình 1.41. Demo cho phân tích phương sai

#### 1.4.1.4. Kiểm định Chi bình phương

Hàm: statsmodels.stats.chi2\_contingency

Công dụng: Thực hiện kiểm định Chi bình phương. Kiểm định Chi bình phương được sử dụng khi chúng ta muốn đánh giá xem liệu có mối quan hệ giữa hai biến định tính hay biến phân loại (categorical variables) trong một tập dữ liệu hay không. VD: Việc lựa chọn ngành học có phụ thuộc vào giới tính hay không.

Cú pháp: **scipy.stats.chi2\_contingency(*observed*, *correction=True*, *lambda\_=None*)**

* Parameters:
  + observed: array\_like. Contingency table, nghĩa là bảng chứa tần số xuất hiện trong mỗi thể loại.
  + correction: bool, tùy chọn. Nếu True, và các bậc tự do là 1, áp dụng chỉnh sửa Yate cho sự liên tục. Hiệu ứng của việc chỉnh sửa là điều chỉnh mỗi giá trị quan sát một đại lượng 0.5 hướng về giá trị kỳ vọng tương ứng.
  + lambda\_: float hoặc str, tùy chọn.
* Return: Đối tượng chứa:
  + statistic: float. Giá trị thống kê kiểm định
  + pvalue: float. p-value của kiểm định
  + dof: int. Các bậc tự do.
  + expected\_freq: ndarray, có cùng kích thước với observed. Các tầng số kỳ vọng.

### 1.4.2. Tương quan:

#### 1.4.2.1. Hệ số tương quan Pearson:

Hàm: scipy.stats.pearsonr

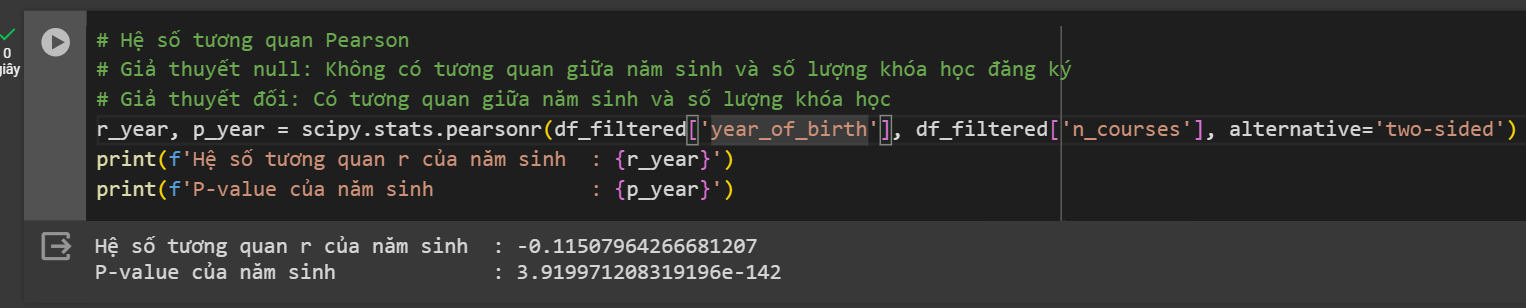
Công dụng: Tính hệ số tương quan Pearson giữa 2 biến ngẫu nhiên liên tục. Hệ số tương quan Pearson ( là thước đo mối quan hệ tuyến tính giữa hai biến và không phụ thuộc vào các đơn vị đo lường của hai biến này. Nếu hệ số tương quan càng gần -1 hoặc 1 thì tương quan tuyến tính càng mạnh, nếu càng gần 0 thì tương quan càng yếu. Nếu hệ số tương quan < 0 thì thể hiện sự nghịch biến, > 0 thì thể hiện sự đồng biến.

Cú pháp: scipy.stats.pearsonr(*x*, *y*, *\**, *alternative='two-sided'*, *method=None*)

* Parameters
  + x (array\_like): array của biến thứ 1
  + y (array\_like): array của biến thứ 2
  + alternative (string): Giả thuyết thay thế. Nhận 1 trong các giá trị sau:
    - ‘two-sided’: tương quan khác không
    - ‘less’: tương quan âm (<0)
    - ‘greater’: tương quan dương (>0)
  + method (ResamplingMethod): Hàm định nghĩa các tính p-value. Nếu method=None (mặc định) thì sẽ tính theo công thức mặc định trong note của scipy.stats.pearsonr
* Return: một đối tượng chứa 2 giá trị sau:
  + statistic (float): hệ số tương quan r
  + pvalue (float): p-value. Nếu p-value < (thường là 0.05) thì bác bỏ giả thuyết null, nghĩa là chấp nhận giả thuyết đối.  *Ngược lại* *thì không đủ cơ sở để bác bỏ giả thuyết null.*

Demo:

* Xét mối tương quan giữa 2 biến year\_of\_birth (năm sinh) và n\_courses (số lượng khóa học đăng ký của học sinh)

**

Hình 1.42. Demo cho hệ số tương quan Pearson

* Nhận xét: Hệ số tương quan r gần bằng 0 → Tương quan tuyến tính yếu. Hệ số p-value < 0.05 → Tương quan có thật (có ý nghĩa thống kê)

#### 1.4.2.2. Hệ số tương quan Spearman:

Hàm: scipy.stats.spearmanr

Công dụng: Tính hệ số tương quan spearman giữa 2 biến liên tục (ví dụ: cân nặng, chiều cao, …) hay thứ bậc (thứ hạng vận động viên, thứ hạng trên lớp). Hệ số tương quan Spearman (Spearman rank correlation) là thước đo phi tham số về tương quan thứ hạng (sự phụ thuộc thống kê giữa thứ hạng của hai biến). Nó đánh giá mối quan hệ giữa hai biến có thể được mô tả tốt như thế nào bằng cách sử dụng một hàm đơn điệu. Nếu hệ số tương quan càng gần -1 hoặc 1 thì tương quan thứ hạng càng mạnh, nếu càng gần 0 thì tương quan càng yếu. Nếu hệ số tương quan < 0 thì thể hiện sự nghịch biến, > 0 thì thể hiện sự đồng biến.

Cú pháp: scipy.stats.spearmanr(*a*, *b=None*, *axis=0*, *nan\_policy='propagate'*, *alternative='two-sided'*)

* Parameters:
  + a, b: 1D hoặc 2D array\_like, b là tùy chọn. Một hoặc hai 1-D hay 2-D arrays chứa nhiều biến và quan sát. Khi chúng là 1-D array, mỗi cái sẽ đại diện cho 1 vector của các quan sát của 1 biến đơn. Trong trường hợp 2-D, xem chi ở parameter axis bên dưới. Cả 2 arrays cần phải có cùng độ dài trong chiều axis.
  + axis: int hoặc None, tùy chọn. Nếu axis=0 (mặc định), mỗi cột biểu diễn 1 biến với các quan sát trên các hàng. Nếu axis=1 thì ngược lại. Nếu axis=None, cả hai arrays sẽ bị chia cắt.
  + nan\_policy: {‘propagate’, ‘raise’, ‘omit’}, tùy chọn. Định nghĩa cách xử lý khi đầu vào chứ nan. Các cách xử lý như sau:
    - propagate: trả về nan
    - raise: báo lỗi
    - omit: thực hiện phép tính trong khi bỏ qua nan.
  + alternative: {‘two-sided’, ‘less’, ‘greater’}, tùy chọn. Định nghĩa giả thuyết đối. Mặc định là ‘two-sided’. Các lựa chọn bao gồm:
    - ‘two-sided’: tương quan khác 0
    - ‘less’: tương quan là âm (<0)
    - ‘greater’: tương quan là dương (>0)
* Return: một đối tượng chứa 2 giá trị sau:
  + statistic: hệ số tương quan Spearman (nếu chỉ 2 có biến) hoặc ma trận tương quan Spearman. Ma trận là vuông với chiều dài bằng tổng số biến trong a và b gộp lại.
  + pvalue: float. Nếu p-value < (thường là 0.05) thì bác bỏ giả thuyết null, nghĩa là chấp nhận giả thuyết đối.  *Ngược lại* *thì không đủ cơ sở để bác bỏ giả thuyết null.*

**Demo:**

****

Hình 1.43. Demo cho hệ số tương quan Spearman

### 1.4.3. Sử dụng mô hình máy học để dự đoán giá trị:

#### **1.4.3.1. Hồi quy tuyến tính**

Công dụng: sklearn.linear\_model.LinearRegression

Cú pháp: Tạo mô hình hồi quy tuyến tính

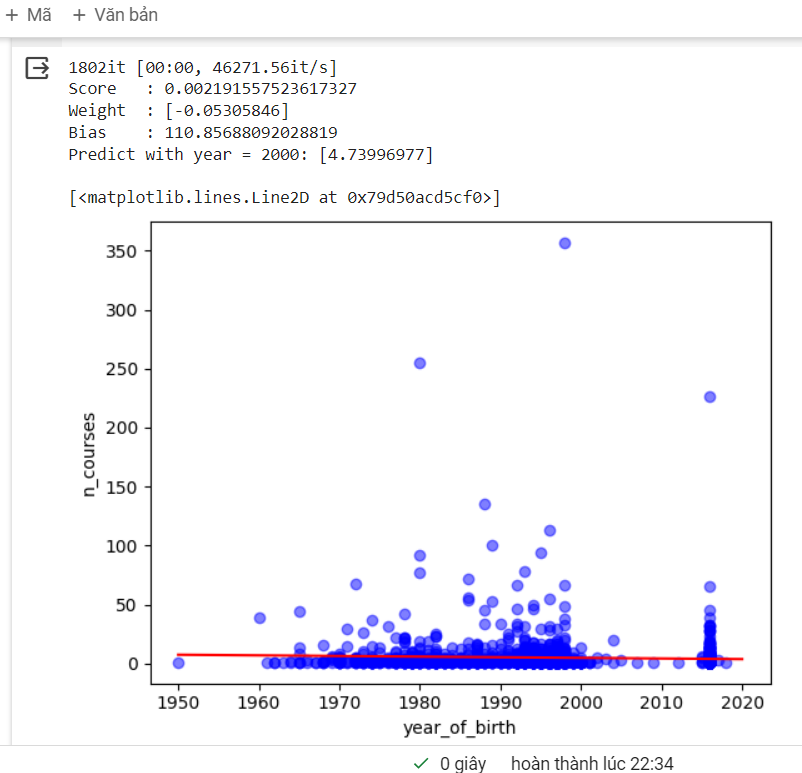
* Parameters:
  + fit\_intercept: bool, mặc định là True. Có tính hệ số cho mô hình hay không. Nếu đặt là False, không có hệ số nào được sử dụng trong tính toán.
  + copy\_X: bool, mặc định là True. Nếu True, X sẽ được copy, ngược lại, nó sẽ bị ghi đè.
  + n\_jobs: int, mặc định là None. Điều chỉnh lượng tài nguyên cho tính toán. Nếu đặt là -1, toàn bộ processors sẽ được sử dụng. Nếu đặt là None thì có nghĩa là 1 nếu không trong ngữ cảnh joblib.parallel\_backend.
  + positive\_value: bool, mặc định là False. Khi đặt là True, bắt mọi hệ số phải dương.
* Return:
  + Linear regression model

Demo:

* Ta sẽ xây dựng một mô hình linear regression để dự đoán số lượng khóa học đăng đăng ký dựa trên năm sinh. Mặc dù cả 2 biến này không có mối tương quan với nhau nhưng ta sẽ tạm sử dụng 2 biến này để làm demo.



Hình 1.44. Code demo cho sử dụng linear regression

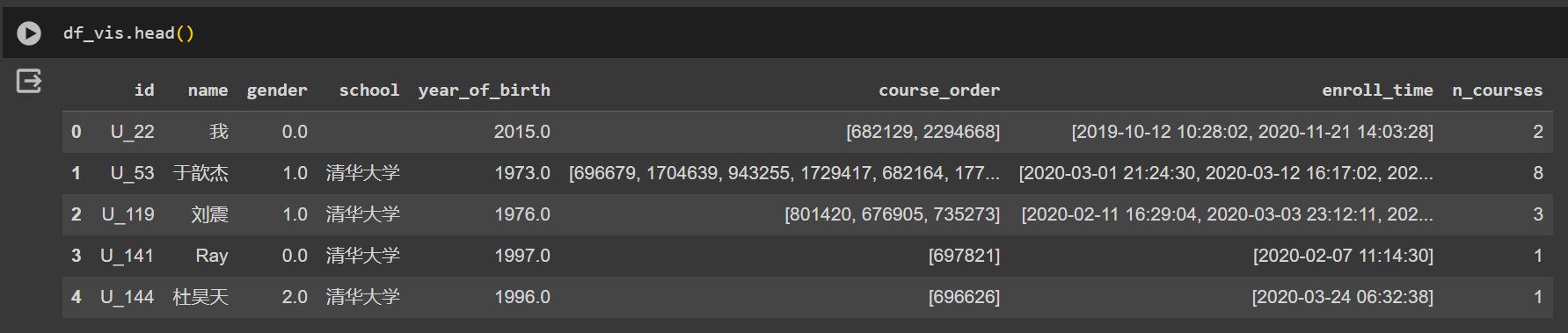
****

Hình 1.45. Kết quả khi chạy code demo linear regression

## 1.5. KHAI PHÁ TRI THỨC:

### 1.5.1. Overview

Nhìn lại một chút về bộ dữ liệu **entities/user.json.**



Hình 1.46. Bảng dữ liệu entities/user.json

**Nhận xét:**

* Khai phá tri thức là việc chúng ta xác định các mẫu, xu hướng và mối quan hệ ẩn trong bộ dữ liệu.
* Dựa vào các trường dữ liệu trong bộ dữ liệu có thể thấy: trường id chứa giá trị id của học viên tham gia khóa học - đây là những giá trị duy nhất nên không thể dùng trường id để tiến hành khai phá tri thức; Các trường thông tin khác như: name, gender, school, year\_of\_birth, enroll\_time là những trường mang thông tin đi kèm với học viên do đó các trường này cũng không thể dùng để khai thác tri thức.
* Dựa vào bảng dữ liệu trên có thể thấy thông tin chứa ở trường course\_order là thông tin về những khóa học mà học viên đã lựa chọn, dựa vào thông tin này chúng ta có thể khai thác được xu hướng cũng như mối quan hệ trong việc đăng ký khóa học của học viên. Từ đó có thể giới thiệu hay đề xuất một khóa học mới dựa trên những khóa học mà học viên đã đăng ký trước đó.

### 1.5.2. Chuẩn bị Transaction và Encode

Transaction và Encode là quá trình chuyển đổi dữ liệu tập hợp (còn gọi là transaction data) thành dạng một ma trận 2 chiều, trong đó mỗi hàng đại diện cho 1 transaction và mỗi cột đại diện cho 1 mục (item) trong bộ dữ liệu.

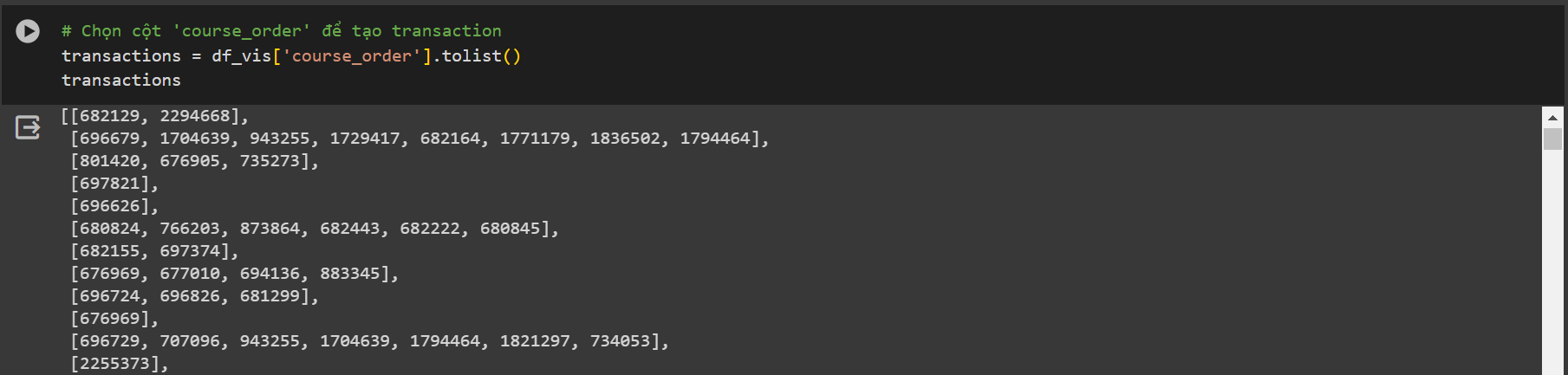
**Mục đích:** là tạo ra một kiểu biểu diễn số hóa của dữ liệu tập hợp, làm cho nó phù hợp cho việc áp dụng thuật toán Apriori để tìm kiếm các mẫu kết hợp trong dữ liệu.

#### 1.5.2.1. Tạo Transaction

Như đã phân tích ở trên, chúng tôi sử dụng trường course\_order để tạo 1 transaction duy nhất. Thực chất của việc tạo transaction là việc nhóm các giá trị trong trường course\_order vào cùng một nhóm

**Cú pháp:** dataframe[‘<feature>’].tolist()

**Code:**

****

Hình 1.47. Đoạn code và kết quả tạo transaction

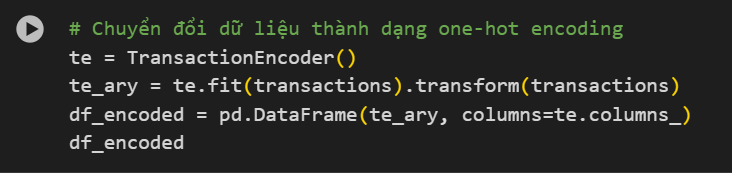
**Nhận xét:** Mỗi dòng trong hình 1.47 đại diện cho giá trị trong trường course\_order của một học viên.

#### 1.5.2.2. Encode

Tiếp theo chúng tôi tiến hành chuyển đổi transaction trên thành dạng one-hot encoding

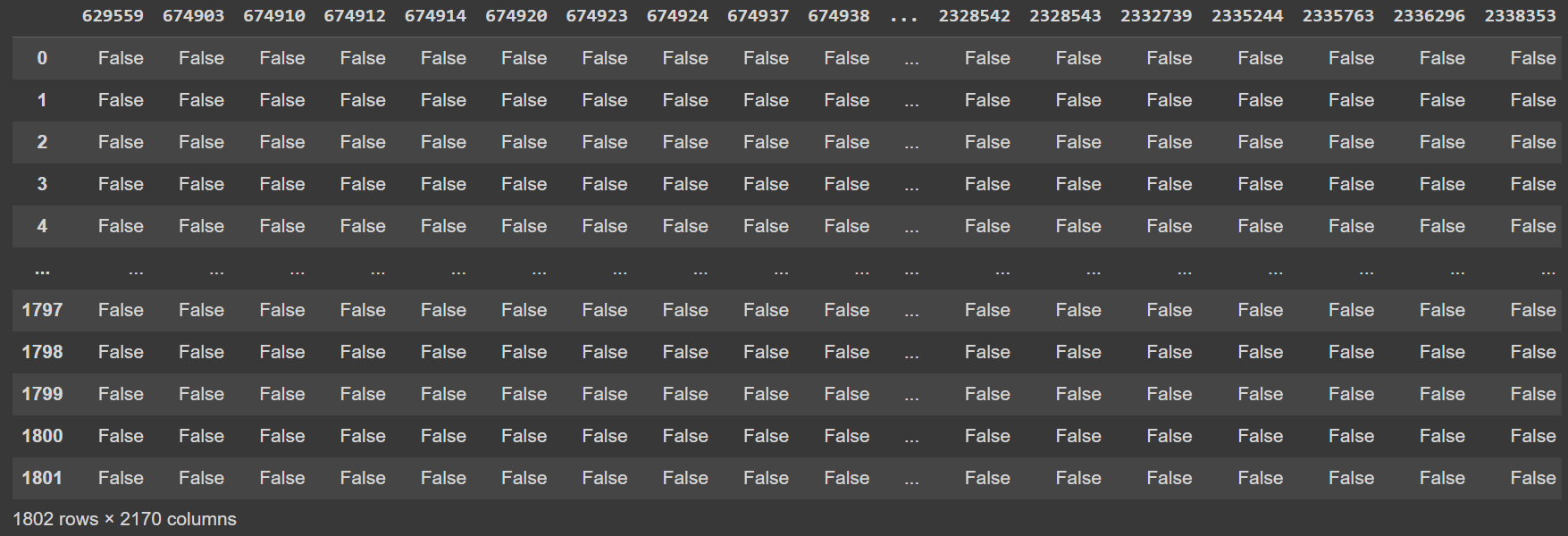
**Cú pháp:** sử dụng hàm TransactionEncoder() của thư viện mlxtend.preprocessing với thao tác fit() và transform()

**Code:**



Hình 1.48. Đoạn code chuyển đổi transaction data thành dạng one-hot encoding

**Kết quả:**

****

Hình 1.49. Kết quả thu được sau khi chạy TransactionEncoder

**Nhận xét:** Sau khi chạy TransactionEncoder chúng tôi thu được bảng kết quả như hình 1.49, trong đó mỗi cột là mỗi khóa học, mỗi dòng chứa các tập khóa học.

### 1.5.3. Apriori algorithm

#### 1.5.3.1. Giới thiệu thuật toán

Thuật toán apriori đã trở thành một trong những thuật toán được sử dụng rộng rãi nhất trong Association rule mining. Nó đã được áp dụng cho nhiều ứng dụng khác nhau, bao gồm phân tích giỏ hàng thị trường, hệ thống khuyến nghị và phát hiện gian lận, đồng thời đã truyền cảm hứng cho sự phát triển của nhiều thuật toán khác cho các nhiệm vụ tương tự.

**Cách hoạt động của thuật toán:**

Thuật toán apriori bắt đầu bằng cách đặt ngưỡng hỗ trợ tối thiểu (min\_support). Đây là số lần tối thiểu một mục phải xuất hiện trong cơ sở dữ liệu để nó được coi là một tập phổ biến. Sau đó, thuật toán sẽ lọc ra bất kỳ tập mục ứng cử viên nào không đáp ứng ngưỡng hỗ trợ tối thiểu.

Sau đó, thuật toán tạo ra một danh sách tất cả các kết hợp có thể có của các tập phổ biến và đếm số lần mỗi kết hợp xuất hiện trong cơ sở dữ liệu. Sau đó, thuật toán tạo ra một danh sách các quy tắc kết hợp dựa trên các kết hợp tập phổ biến.

Quy tắc kết hợp là một tuyên bố có dạng "nếu mục A có trong giao dịch thì mục B cũng có khả năng xuất hiện". Độ mạnh của sự liên kết được đo bằng cách sử dụng độ tin cậy của quy tắc, đó là xác suất mà mục B có mặt khi có mục A.

Sau đó, thuật toán sẽ lọc ra bất kỳ quy tắc kết hợp nào không đáp ứng ngưỡng tin cậy tối thiểu. Những luật này được gọi là luật kết hợp mạnh. Cuối cùng, thuật toán trả về danh sách các quy tắc kết hợp mạnh làm đầu ra.

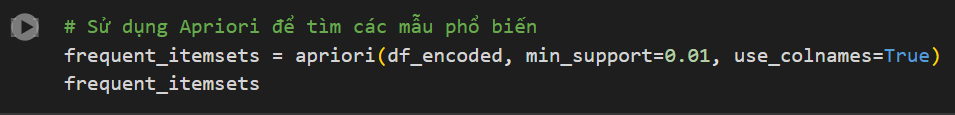
Apriori sử dụng cách tiếp cận "từ dưới lên", bắt đầu với các mục riêng lẻ và dần dần kết hợp chúng thành các tập mục ngày càng lớn hơn khi nó tìm kiếm các mẫu phổ biến. Nó cũng sử dụng phương pháp "xóa-nhãn lại" để cắt bớt không gian tìm kiếm một cách hiệu quả bằng cách loại bỏ các tập mục không thường xuyên khỏi việc xem xét.

#### 1.5.3.2. Áp dụng thuật toán Apriori

**Cú pháp:** sử dụng hàm apriori của thư viện mlxtend.frequent\_patterns các tham số sau:

* dataframe: đây là ma trận dữ liệu sau khi được mã hóa one-hot encoding.
* min\_support: đây là tham số biểu thị ngưỡng tối thiểu cho mức hỗ trợ của các itemset để được coi là phổ biến.
* use\_colnames: tham số này để hiển thị tên của bảng kết quả.

**Code:** Áp dụng thuật toán Apriori trên với các tham số sau:

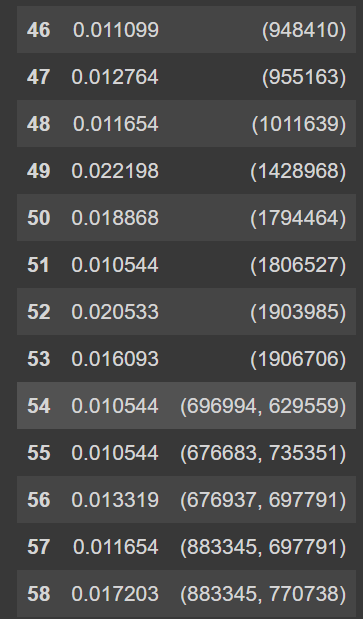
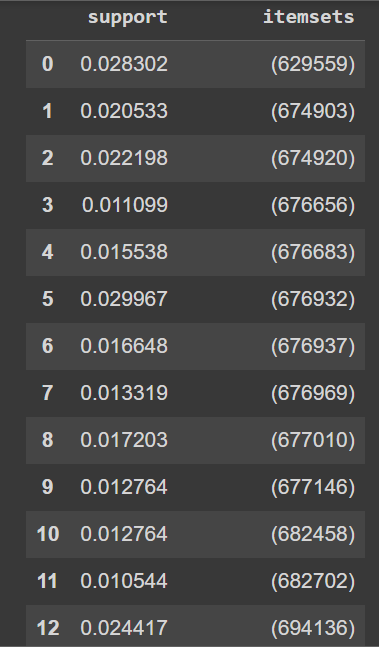


Hình 1.50. Đoạn code sử dụng thuật toán Apriori

**Giải thích các parameters:**

* dataframe: df\_encoded (dữ liệu đã được one-hot encoding trước đó)
* min\_support=0.01, vì tần số xuất hiện của các itemset ứng với bộ dữ liệu là không nhiều chính vì thế nếu chọn min\_support quá lớn thì kết quả của thuật toán sẽ là rỗng, ngược lại nếu đặt quá nhỏ thì thuật toán sẽ không còn đúng theo ý nghĩa của nó là tìm những itemset có tần số xuất hiện lớn.
* use\_colnames=True để hiển thị tên của bảng kết quả.

**Kết quả:**



Hình 1.51. Kết quả sau khi chạy thuật toán Apriori

**Nhận xét:** có thể thấy tần suất xuất hiện của các itemset trong bộ dữ liệu là không nhiều được thể hiện ở cột support, và đồng thời các itemset có tần suất xuất hiện cao hơn 0.01 phần lớn có 1 phần tử, chỉ tồn tại 1 vài itemset có 2 phần tử. Điều này cho thấy các khóa học mà các học viên đăng ký là tự do, không có tính xâu chuỗi cao.

### 1.5.4. Association rule

#### 1.5.4.1. Giới thiệu

Khai thác luật kết hợp là một kỹ thuật được sử dụng để xác định các mẫu trong các tập dữ liệu lớn. Nó liên quan đến việc tìm kiếm mối quan hệ giữa các biến trong dữ liệu và sử dụng các mối quan hệ đó để đưa ra dự đoán hoặc quyết định.

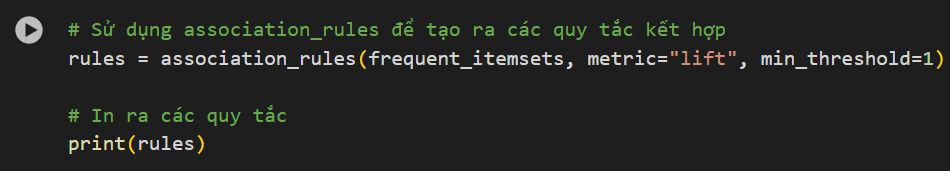
**Mục đích:** là khám phá các luật mô tả mối quan hệ giữa các itemset khác nhau trong tập dữ liệu.

#### 1.5.4.2. Áp dụng Association rule

**Cú pháp:** sử dụng hàm association\_rules của thư viện mlxtend.frequent\_patterns các tham số sau:

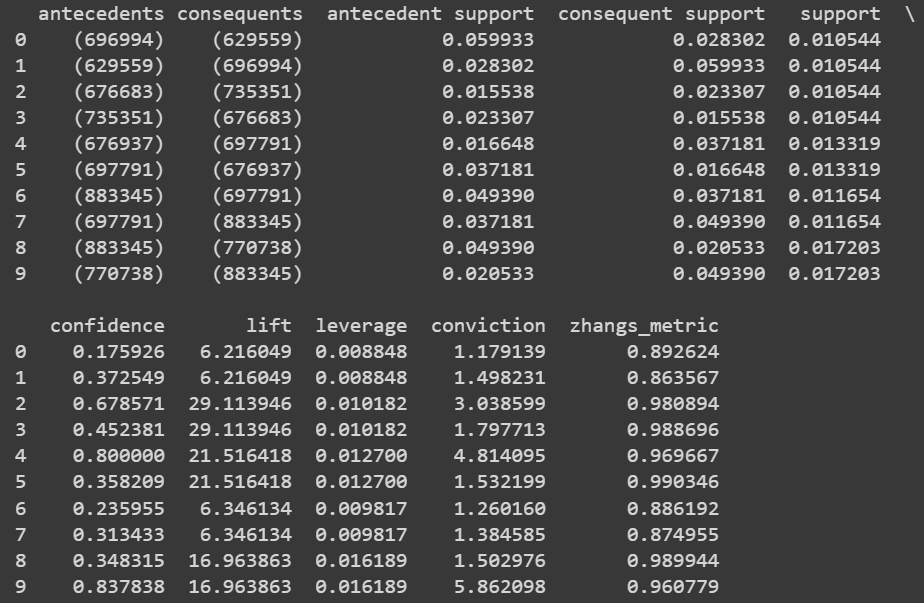
* frequent\_itemset: đây là tập itemset phổ biến mà đã được tìm thấy sau khi sử dụng thuật toán Apriori.
* metric: đây là phương pháp được sử dụng để đánh giá các quy tắc kết hợp. Các phương pháp thường được sử dụng bao gồm support, confident, lift,...
* min\_threshold: đây là chặn dưới của metric, chỉ các quy tắc nào có giá trị lớn hơn mới được chấp nhận.

**Code:** Áp dụng Association rule để tạo các rules:



Hình 1.52. Đoạn code sử dụng association\_rules

**Kết quả:**

****

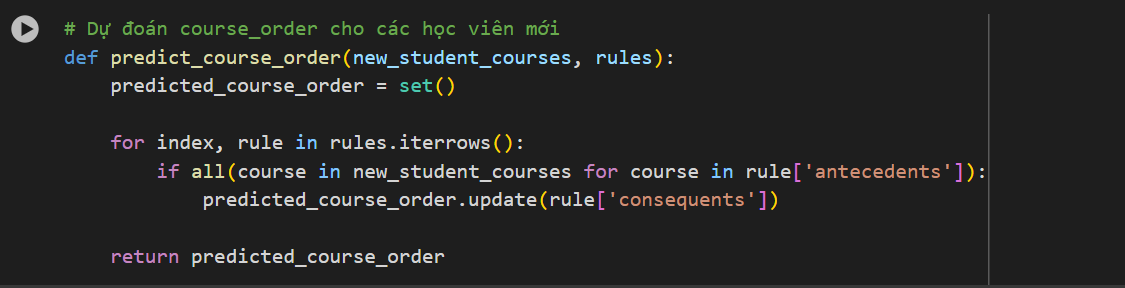
Hình 1.53. Kết quả sau khi chạy association\_rules

**Nhận xét:**

* Cột antecedents: là tập hợp các items trong quy tắc kết hợp.
* Cột consequents: là tập hợp các items trong kết quả của quy tắc kết hợp.
* Các cột antecedent support, consequent support, confident, lift, leverage, conviction, zhangs\_metric là các độ đo của quy tắc kết hợp.
* Dựa vào kết quả từ cột lift có thể thấy khả năng giới thiệu khóa học mới khi biết khóa học trước đó.

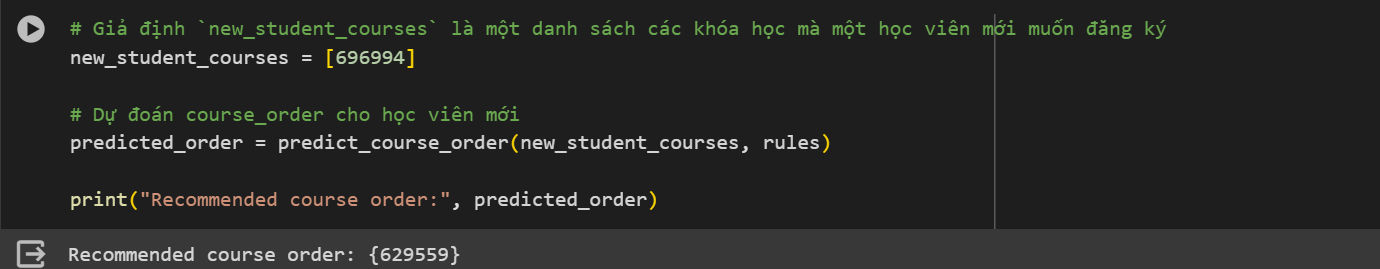
#### 1.5.4.3. Tiến hành gợi ý khóa học mới

**Code:**



Hình 1.54. Đoạn code xây dựng hàm gợi ý khóa học dựa trên các rules được tạo ra

**Kết quả:**



Hình 1.55. Đoạn code kết quả của hàm gợi ý khóa học

### 1.5.5. Nhận xét

Sau khi tiến hành các thao tác khai phá tri thức từ bảng **entities/user.json**, cụ thể là sử dụng association\_rules ở trường course\_order để tìm ra các cách kết hợp, chúng tôi nhận thấy có thể sử dụng những thông tin kết hợp của trường này để tiến hành dự đoán xu hướng hay gợi ý những khóa học cho những học viên mới.

# 2. LÀM SẠCH DỮ LIỆU:

## 2.1. Xử lý dữ liệu trùng lặp:

Trong việc xử lí dữ liệu trùng lặp, tuỳ vào mục đích sử dụng mà ta có thể giữ lại hoặc loại bỏ những dữ liệu này. Thường thì việc loại bỏ sẽ được lựa chọn nhiều hơn khi nó có thể làm sạch dữ liệu và hỗ trợ tốt hơn cho những tác vụ xử lí sau này.

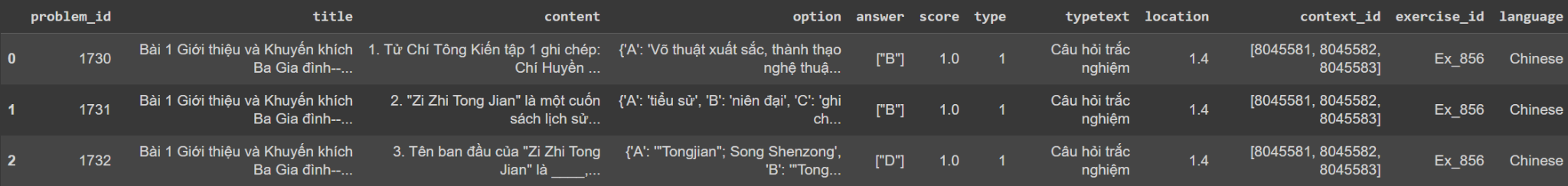
**Hàm sử dụng:** df.duplicated() và df.drop\_duplicates()

**Công dụng:**

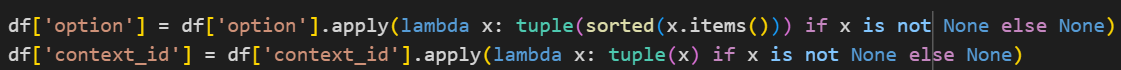
* df.duplicated(): là hàm có trong thư viện pandas được dùng để kiểm tra dữ liệu bị trùng lặp trong dataframe. Hàm này có các tham số (subset: quy định các cột cần kiểm tra, keep: có các giá trị {first, last, False} để quy định cách đánh dấu dữ liệu trùng lặp)
* df.drop\_duplicates(): được dùng để xoá những dữ liệu trùng lặp trong dataframe. Hàm có các tham số (subset, keep, inplace: quy định việc có tạo ra dataframe mới hay chỉnh sửa trên dataframe có sẵn)

**Áp dụng:** Đối với bộ dữ liệu **entities/problem.json**, có các cột mang kiểu dữ liệu **dict** (cột option) và **list** (cột context\_id). Những kiểu dữ liệu này không thể so sánh được khi sử dụng những hàm nêu trên trên, do đó, ta sẽ chuyển kiểu thành tuple để sử dụng được hàm *duplicated()* và *drop\_duplicates()*

**Trước khi đổi kiểu:**

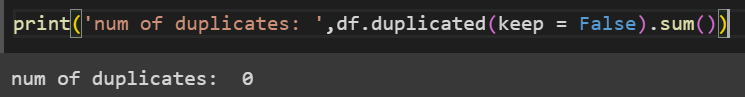


**Code:**

**

**Kết quả:**



Sau đó, ta tiến hành sử dụng hàm *duplicated()* để kiểm tra dữ liệu trùng lặp: 

Với kết quả này, việc sử dụng hàm *drop\_duplicates()* là không cần thiết vì bộ dữ liệu không bị trùng lặp.