

BÀI 1: CHUẨN BỊ DỮ LIỆU

I. Mục tiêu:

Sau khi thực hành xong, sinh viên nắm được:

- Các bước làm sạch dữ liệu và tiền xử lý dữ liệu
- Sử dụng các thư viện Pandas và scikit-learn.
- Chuẩn hóa dữ liệu.
- Rời rạc hóa dữ liệu.
- PCA

II. Tóm tắt lý thuyết:

Tập dữ liệu nhiều chiều D là một tập hợp gồm n bản ghi $\overline{X}_1, \overline{X}_2, \dots, \overline{X}_n$, sao cho mỗi \overline{X}_i là một tập hợp chứa d đặc trưng được ký hiệu bởi (x_i^1, \dots, x_i^d) .

1. Chuẩn hóa dữ liệu:

- Xét trường hợp thuộc tính thứ j có trung bình (mean) là μ_j và độ lệch chuẩn (standard deviation) σ_j . Khi đó, x_i^j (giá trị thuộc tính thứ j) của \overline{X}_i (bản ghi thứ i) có thể được chuẩn hóa như sau:

$$z_i^j = \frac{x_i^j - \mu_j}{\sigma_j}$$

- Xấp xỉ thứ 2 sử dụng min-max scaling để ánh xạ tất cả thuộc tính thành vùng $[0,1]$. Đặt \min_j và \max_j là các giá trị nhỏ nhất và lớn nhất của thuộc tính j . Khi đó, x_i^j của \overline{X}_i có thể được scale như sau:

$$y_i^j = \frac{x_i^j - \min_j}{\max_j - \min_j}$$

2. Rời rạc hóa dữ liệu:

- a. **Equi-width ranges:** là chia các giá trị này thành các khoảng bằng nhau về độ rộng (width) hoặc bin. Cụ thể hơn là, các bin có độ rộng bằng nhau với mỗi bin

được xác định như sau: $[min, min + w - 1], [min + w, min + 2w - 1], \dots [min + kw, max]$ với $w = \frac{(max - min)}{k}$, k là số bin.

Ví dụ: Cho các giá trị 10, 15, 18, 20, 31, 34, 41, 46, 51, 53, 54 chia thành 4 bin.

- **Bước 1:** Sắp xếp dữ liệu theo thứ tự tăng dần:

10, 15, 18, 20, 31, 34, 41, 46, 51, 53, 54.

- **Bước 2:** Xác định độ rộng của mỗi bin sử dụng công thức $w = \frac{(max - min)}{k} = \frac{54 - 10}{4} = 11$, $k = 4$ là số bin. Các bin có dạng là

Bin 1 : $[(min), (min + w - 1)] = [10, 20]$

Bin 2 : $[(min + w), (min + 2w - 1)] = [21, 31]$

Bin 3 : $[(min + 2w), (min + 3w - 1)] = [32, 42]$

Bin 4 : $[(min + 3w), (max)] = [43, 54]$

Khi đó, ta có:

Bin 1 : 10, 15, 18, 20

Bin 2 : 31

Bin 3 : 34, 41

Bin 4 : 46, 51, 53, 54

- b. Equi-depth (or frequency) ranges:** Trong trường hợp này, các vùng được chọn sao cho mỗi vùng có số các bản ghi bằng nhau. Một thuộc tính có thể được chia thành các vùng equi-depth bằng việc sắp xếp nó đầu tiên và sau đó việc lựa chọn các điểm phân chia trong giá trị thuộc tính được sắp xếp, sao cho mỗi vùng chứa số các bản ghi bằng nhau.

Ví dụ: Cho các giá trị 15, 10, 18, 20, 31, 34, 41, 46, 51, 53, 54, 60 chia thành 4 khoảng cùng số điểm dữ liệu.

- **Bước 1:** Sắp xếp dữ liệu theo thứ tự tăng dần:

10, 15, 18, 20, 31, 34, 41, 46, 51, 53, 54, 60.

- **Bước 2:** tổng số các phần điểm dữ liệu là 12, số bin yêu cầu là 4 nên mỗi bin sẽ có 3 điểm dữ liệu. Khi đó, các bin có dạng là

Bin 1 : 10, 15, 18

Bin 2 : 20, 31, 34

Bin 3 : 41, 46, 51

Bin 4 : 53, 54, 60

3. PCA:

Cho C là ma trận hiệp phương sai đối xứng ($d \times d$) của ma trận dữ liệu D ($n \times d$). Khi đó, phần tử c_{ij} của C ký hiệu hiệp phương sai giữa cột i và cột j (số chiều) của ma trận dữ liệu D . Cho μ_i biểu diễn trung bình theo chiều thứ i . Nếu x_k^m là bản ghi thứ k của chiều thứ m thì giá trị của phần tử hiệp phương sai c_{ij} được tính như sau:

$$c_{ij} = \frac{\sum_{k=1}^n x_k^i x_k^j}{n} - \mu_i \mu_j \quad \forall i, j \in \{1, \dots, d\}.$$

Cho $\bar{\mu} = (\mu_1 \dots \mu_d)$ là vector dòng d chiều biểu diễn các giá trị trung bình theo các chiều khác nhau. Khi đó, các tính toán $d \times d$ ở trên của phương trình cho các giá trị khác nhau của i và j có thể được biểu diễn trong dạng ma trận $d \times d$ như sau:

$$C = \frac{D^T D}{n} - \bar{\mu}^T \bar{\mu}$$

Chú ý rằng d phần tử đường chéo của ma trận C tương ứng với d phương sai. Ma trận hiệp phương sai C nửa xác định dương vì nó có thể được chứng minh rằng với vector cột d chiều \bar{v} , giá trị của $\bar{v}^T C \bar{v}$ bằng với phương sai của phép chiếu 1 chiều $D\bar{v}$ của tập dữ liệu D trong \bar{v} .

$$\bar{v}^T C \bar{v} = \frac{(D\bar{v})^T D\bar{v}}{n} - (\bar{\mu}\bar{v})^2 = \text{phương sai của các điểm 1 chiều trong } D\bar{v} \geq 0.$$

Thật vậy, mục tiêu của PCA là để xác định liên tục các vector trực giao \bar{v} cực đại $\bar{v}^T C \bar{v}$. Bởi vì ma trận hiệp phương sai là đối xứng và nửa xác định dương nên nó có thể được chéo hóa như sau:

$$C = P \Lambda P^T$$

Các cột của ma trận P chứa các vector trực giao của C , Λ là ma trận đường chéo chứa các giá trị riêng không âm. Phần tử Λ_{ii} là trị riêng tương ứng với vector riêng thứ i

(hoặc cột) của ma trận P . Các vector riêng này biểu diễn các lời giải trực giao liên tục nhau thành mô hình tối ưu hóa cực đại phương sai $\bar{v}^T C \bar{v}$ theo phương thống nhất \bar{v} . Một tính chất thú vị của sự chéo hóa này là cả vector riêng và trị riêng đều có một thể hiện hình học liên quan tới phân phối dữ liệu cơ bản. Đặc biệt, nếu hệ trục của biểu diễn dữ liệu được xoay thành tập trực giao của các vector riêng trong các cột của P thì nó có thể được chứng minh rằng $\binom{d}{2}$ hiệp phương sai của các giá trị đặc trưng được biến đổi mới là 0. Mặt khác, các phương lưu trữ phương sai lớn nhất cũng là các phương tương quan loại bỏ. Hơn nữa, các trị riêng biểu diễn các phương sai của dữ liệu theo các vector riêng tương ứng. Thật vậy, ma trận đường chéo Λ là ma trận phương sai mới sau khi xoay trục. Do đó, các vector riêng với các trị riêng lớn bảo toàn phương sai lớn hơn và cũng được nhắc đến như các thành phần chính. Bởi vì sự tự nhiên của công thức tối ưu hóa thường sử dụng để suy ra sự biến đổi này, một hệ thống trục mới chỉ chứa các vector riêng với các trị riêng lớn nhất được tối ưu thành phương sai lớn nhất liên tục trong một số chiều được cố định.

III. Nội dung thực hành:

1. Làm sạch và tiền xử lý dữ liệu:

Làm sạch và tiền xử lý dữ liệu với Pandas là các bước có tính quyết định trong phân tích dữ liệu khi chúng ta đảm bảo rằng dữ liệu là chất lượng cao và sẵn sàng cho phân tích. Dưới đây là một vài bước cơ bản cho làm sạch và tiền xử lý dữ liệu với Pandas:

- Load dữ liệu thành DataFrame Pandas

Tạo file "data.csv" như sau:

ID	First Name	Last Name	Age	Gender	Departme	Salary	Date of Joining
1	John	Doe	25	M	Sales	50000	01/01/2020
2	Jane	Smith	30	F	Marketing	60000	06/01/2018
3	Bod	Johnson	45	M	HR	70000	09/01/2016
4	Alice	Williams	33	F	IT	80000	02/01/2017
5	James	Brown	27	M	Sales	55000	03/01/2019
6	Sarah	Lee		F	Marketing	65000	12/01/2018
7	Michael	Davis	39	M	HR		08/01/2015
8	Susan	Miller	42	F	IT	90000	11/01/2014
9	David	Wilson	28	M	Sales	60000	05/01/2020
10	Emily	Brown	35	F	Marketing	55000	04/01/2017
11	John	Doe	25	M	Sales	50000	01/01/2020
12	John	Doe	25	M	Sales	50000	01/01/2020

Load dữ liệu

```
#- Load dữ liệu thành DataFrame Pandas
import pandas as pd
import numpy as np
from sklearn import preprocessing

df = pd.read_csv('D:\\Huynh\\DataMining_Lab\\data\\tuan1\\data.csv',
                 delimiter=',')
#đặt lại index
df = df.set_index('ID')
print(df)
```

ID	First Name	Last Name	Age	Gender	Department	Salary	Date of Joining
1	John	Doe	25.0	M	Sales	50000.0	01/01/2020
2	Jane	Smith	30.0	F	Marketing	60000.0	06/01/2018
3	Bod	Johnson	45.0	M	HR	70000.0	09/01/2016
4	Alice	Williams	33.0	F	IT	80000.0	02/01/2017
5	James	Brown	27.0	M	Sales	55000.0	03/01/2019
6	Sarah	Lee	NaN	F	Marketing	65000.0	12/01/2018
7	Michael	Davis	39.0	M	HR	NaN	08/01/2015
8	Susan	Miller	42.0	F	IT	90000.0	11/01/2014
9	David	Wilson	28.0	M	Sales	60000.0	05/01/2020
10	Emily	Brown	35.0	F	Marketing	55000.0	04/01/2017
11	John	Doe	25.0	M	Sales	50000.0	01/01/2020
12	John	Doe	25.0	M	Sales	50000.0	01/01/2020

- Xử lý các giá trị missing: sử dụng các hàm `isnull()`, `fillna()` (làm đầy các giá trị missing với một giá trị đặc biệt) và `dropna()` (bỏ các dòng với các giá trị missing)

```
#2. Kiểm tra các giá trị missing
print(df.isnull().sum())
#làm đầy các giá trị missing bằng giá trị trung bình
df.fillna(df.mean(), inplace=True)
```

```
First Name    0
Last Name     0
Age           1
Gender        0
Department    0
Salary        1
Date of Joining 0
dtype: int64
2
```

- Xử lý các giá trị giống nhau: sử dụng các hàm `duplicated()` (trả về giá trị luận lý cho biết mỗi dòng là giống nhau hay không) và `drop_duplicates()` (bỏ các dòng giống nhau).

```
#3. Kiểm tra giá trị giống nhau
print(df.duplicated().sum())

##loại bỏ các các dòng giống nhau
df=df.drop_duplicates()
print(df)
```

	First Name	Last Name	Age	...	Department	Salary	Date of Joining
ID				...			
1	John	Doe	25.000000	...	Sales	50000.000000	01/01/2020
2	Jane	Smith	30.000000	...	Marketing	60000.000000	06/01/2018
3	Bod	Johnson	45.000000	...	HR	70000.000000	09/01/2016
4	Alice	Williams	33.000000	...	IT	80000.000000	02/01/2017
5	James	Brown	27.000000	...	Sales	55000.000000	03/01/2019
6	Sarah	Lee	32.181818	...	Marketing	65000.000000	12/01/2018
7	Michael	Davis	39.000000	...	HR	62272.727273	08/01/2015
8	Susan	Miller	42.000000	...	IT	90000.000000	11/01/2014
9	David	Wilson	28.000000	...	Sales	60000.000000	05/01/2020
10	Emily	Brown	35.000000	...	Marketing	55000.000000	04/01/2017

- Việc mã hóa các biến categorical: sử dụng các hàm như `get_dummies()` (khởi tạo một cột nhị phân cho mỗi category) và `LabelEncoder()` (gán các giá trị số thành mỗi category).

```
#- Mã hóa dữ liệu categorical
df = pd.get_dummies(df, columns=['Gender', 'Department'])
```

- Xử lý dữ liệu Datetime: sử dụng các hàm như sau `to_datetime()` (chuyển một giá trị chuỗi hoặc số thành một đối tượng DateTime) và `strftime()` (định dạng một đối tượng Datetime thành một chuỗi).

```
#- Xử lý dữ liệu datetime
##chuyển cột date thành một đối tượng datetime
df['Date of Joining']=pd.to_datetime(df['Date of Joining'])
##extract month and day of week from date column
df['month'] = df['Date of Joining'].dt.month
df['day_of_week'] = df['Date of Joining'].dt.day_name()
## Drop the original 'Date' column
df = df.drop('Date of Joining', axis=1)
print(df)
```

	First Name	Last Name	Age	...	Department_Sales	month	day_of_week
ID				...			
1	John	Doe	25.000000	...	1	1	Wednesday
2	Jane	Smith	30.000000	...	0	6	Friday
3	Bod	Johnson	45.000000	...	0	9	Thursday
4	Alice	Williams	33.000000	...	0	2	Wednesday
5	James	Brown	27.000000	...	1	3	Friday
6	Sarah	Lee	32.181818	...	0	12	Saturday
7	Michael	Davis	39.000000	...	0	8	Saturday
8	Susan	Miller	42.000000	...	0	11	Saturday
9	David	Wilson	28.000000	...	1	5	Friday
10	Emily	Brown	35.000000	...	0	4	Saturday

[10 rows x 12 columns]

- Xử lý các giá trị ngoại lai:

- Chuẩn hóa và scaling dữ liệu: sử dụng các phương pháp khác nhau như min-max scaling, chuẩn hóa z-score (standard) và biến đổi log.

```
#- xử lý các giá trị ngoại lai
#- chuẩn hóa và scale dữ liệu
df1 = df.drop(['First Name', 'Last Name', 'day_of_week'], axis=1)
array = df1.values
print(array)

###sử dụng RobustScaler() để loại bỏ những giá trị ngoại lai
scaler = preprocessing.RobustScaler()
robust_df = scaler.fit_transform(array)
robust_df = pd.DataFrame(robust_df)

###Chuẩn hóa dữ liệu bằng phương pháp z-score (Standard)
scaler = preprocessing.StandardScaler()
standard = scaler.fit_transform(array)
standard_df = pd.DataFrame(standard, index = df.index)
print('Chuan hoa du lieu:\n', standard_df)

###scale dữ liệu bằng phương pháp minmax
scaler = preprocessing.MinMaxScaler()
minmax = scaler.fit_transform(array)
minmax_df = pd.DataFrame(minmax, index = df.index)
print('Scaling dữ liệu:\n', minmax_df)
```

Chuan hoa du lieu:

	0	1	2	3	4	5	6	7	8
ID									
1	-1.369782	-1.264391	-1.0	1.0	-0.5	-0.5	-0.654654	1.527525	-1.420508
2	-0.575077	-0.405854	1.0	-1.0	-0.5	-0.5	1.527525	-0.654654	-0.027853
3	1.809037	0.452683	-1.0	1.0	2.0	-0.5	-0.654654	-0.654654	0.807740
4	-0.098254	1.311220	1.0	-1.0	-0.5	2.0	-0.654654	-0.654654	-1.141977
5	-1.051900	-0.835122	-1.0	1.0	-0.5	-0.5	-0.654654	1.527525	-0.863446
6	-0.228297	0.023415	1.0	-1.0	-0.5	-0.5	1.527525	-0.654654	1.643333
7	0.855391	-0.210732	-1.0	1.0	2.0	-0.5	-0.654654	-0.654654	0.529209
8	1.332214	2.169757	1.0	-1.0	-0.5	2.0	-0.654654	-0.654654	1.364802
9	-0.892959	-0.405854	-1.0	1.0	-0.5	-0.5	-0.654654	1.527525	-0.306384
10	0.219627	-0.835122	1.0	-1.0	-0.5	-0.5	1.527525	-0.654654	-0.584915

Scaling dữ liệu:

	0	1	2	3	4	5	6	7	8
ID									
1	0.000000	0.000000	0.0	1.0	0.0	0.0	0.0	1.0	0.000000
2	0.250000	0.250000	1.0	0.0	0.0	0.0	1.0	0.0	0.454545
3	1.000000	0.500000	0.0	1.0	1.0	0.0	0.0	0.0	0.727273
4	0.400000	0.750000	1.0	0.0	0.0	1.0	0.0	0.0	0.090909
5	0.100000	0.125000	0.0	1.0	0.0	0.0	0.0	1.0	0.181818
6	0.359091	0.375000	1.0	0.0	0.0	0.0	1.0	0.0	1.000000
7	0.700000	0.306818	0.0	1.0	1.0	0.0	0.0	0.0	0.636364
8	0.850000	1.000000	1.0	0.0	0.0	1.0	0.0	0.0	0.909091
9	0.150000	0.250000	0.0	1.0	0.0	0.0	0.0	1.0	0.363636
10	0.500000	0.125000	1.0	0.0	0.0	0.0	1.0	0.0	0.272727

Rời rạc hóa dữ liệu:

- 10 equi-width ranges (sử dụng mục 2.a): dùng hàm `pd.cut()`


```
#####10 equi-width ranges với cột đầu tiên của standard_df
df2=standard_df.copy()
df2['equi-width_column0'] = pd.cut(x=df2[0], bins=10)
print('Roi rac hoa cot 0 bang 10 equi-width ranges:\n',df2)
```

```
Roi rac hoa cot 0 bang 10 equi-width ranges:
      0      1      2      ...      7      8  equi-width_column0
ID
1 -1.369782 -1.264391 -1.0    ...  1.527525 -1.420508  (-1.373, -1.052]
2 -0.575077 -0.405854  1.0    ... -0.654654 -0.027853  (-0.734, -0.416]
3  1.809037  0.452683 -1.0    ... -0.654654  0.807740   (1.491, 1.809]
4 -0.098254  1.311220  1.0    ... -0.654654 -1.141977  (-0.0983, 0.22]
5 -1.051900 -0.835122 -1.0    ...  1.527525 -0.863446  (-1.373, -1.052]
6 -0.228297  0.023415  1.0    ... -0.654654  1.643333  (-0.416, -0.0983]
7  0.855391 -0.210732 -1.0    ... -0.654654  0.529209   (0.538, 0.855]
8  1.332214  2.169757  1.0    ... -0.654654  1.364802   (1.173, 1.491]
9 -0.892959 -0.405854 -1.0    ...  1.527525 -0.306384  (-1.052, -0.734]
10 0.219627 -0.835122  1.0    ... -0.654654 -0.584915  (-0.0983, 0.22]

[10 rows x 10 columns]
Roi rac hoa cot 0 bang 10 equi-depth ranges:
```

– 10 equi-depth ranges (sử dụng mục 2.b): dùng hàm `pd.qcut()`

```
#####10 equi-depth ranges với cột đầu tiên của standard_df
df3=standard_df.copy()
df3['equi-depth_column0'] = pd.qcut(df3[0], q=10)
print('Roi rac hoa cot 0 bang 10 equi-depth ranges:\n',df3)
```

```
Roi rac hoa cot 0 bang 10 equi-depth ranges:
      0      1      2      ...      7      8  equi-depth_column0
ID
1 -1.369782 -1.264391 -1.0    ...  1.527525 -1.420508  (-1.371, -1.084]
2 -0.575077 -0.405854  1.0    ... -0.654654 -0.027853  (-0.67, -0.367]
3  1.809037  0.452683 -1.0    ... -0.654654  0.807740   (1.38, 1.809]
4 -0.098254  1.311220  1.0    ... -0.654654 -1.141977  (-0.163, 0.0289]
5 -1.051900 -0.835122 -1.0    ...  1.527525 -0.863446  (-1.084, -0.925]
6 -0.228297  0.023415  1.0    ... -0.654654  1.643333  (-0.367, -0.163]
7  0.855391 -0.210732 -1.0    ... -0.654654  0.529209   (0.41, 0.951]
8  1.332214  2.169757  1.0    ... -0.654654  1.364802   (0.951, 1.38]
9 -0.892959 -0.405854 -1.0    ...  1.527525 -0.306384  (-0.925, -0.67]
10 0.219627 -0.835122  1.0    ... -0.654654 -0.584915  (0.0289, 0.41]

[10 rows x 10 columns]
```

2. Bài tập vận dụng:

- Download Arrhythmia từ UCI Machine Learning Repository
(<https://archive.ics.uci.edu/dataset/5/arrhythmia>)

archive.ics.uci.edu/dataset/5/arrhythmia

UC Irvine Machine Learning Repository

Datasets Contribute Dataset About Us

Search datasets...

Arrhythmia

Donated on 12/31/1997

Distinguish between the presence and absence of cardiac arrhythmia and classify it in one of the 16 groups.

Dataset Characteristics	Subject Area	Associated Tasks
Multivariate	Life	Classification

Attribute Type	# Instances	# Attributes
Categorical, Integer, Real	452	279

Information

DOWNLOAD

CITE

19 citations
21215 views

Creators

- H. Guvenir
- Burak Acar
- Haldun Muderrisoglu
- R. Quinlan

- Cài đặt thư viện scikit-learn:

```
C:\Users\Huynh>pip install -U scikit-learn
Collecting scikit-learn
  Downloading scikit_learn-1.0.2-cp37m-win_amd64.whl (7.1 MB)
    Requirement already satisfied: scipy>=1.1.0 in c:\users\huynh\appdata\local\programs\python\python37\lib\site-packages (from scikit-learn) (1.3.1)
    Requirement already satisfied: numpy>=1.14.6 in c:\users\huynh\appdata\local\programs\python\python37\lib\site-packages (from scikit-learn) (1.17.2)
    Collecting joblib>=0.11
      Downloading joblib-1.2.0-py3-none-any.whl (297 kB)
    Collecting threadpoolctl>=2.0.0
      Downloading threadpoolctl-3.1.0-py3-none-any.whl (14 kB)
Installing collected packages: threadpoolctl, joblib, scikit-learn
Successfully installed joblib-1.2.0 scikit-learn-1.0.2 threadpoolctl-3.1.0
```

- Đọc dữ liệu từ file "arrhythmia.data":

```
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import KBinsDiscretizer

df = pd.read_csv('D:\\Huynh\\DataMining_Lab\\data\\tuan1\\arrhythmia.data',
                 delimiter=',', header = None)
print(df)
```

	0	1	2	3	4	5	6	...	273	274	275	276	277	278	279
0	75	0	190	80	91	193	371	...	0.0	0.0	0.9	2.9	23.3	49.4	8
1	56	1	165	64	81	174	401	...	0.0	0.0	0.2	2.1	20.4	38.8	6
2	54	0	172	95	138	163	386	...	0.0	0.0	0.3	3.4	12.3	49.0	10
3	55	0	175	94	100	202	380	...	0.0	0.0	0.4	2.6	34.6	61.6	1
4	75	0	190	80	88	181	360	...	0.0	0.0	-0.1	3.9	25.4	62.8	7
...
447	53	1	160	70	80	199	382	...	0.0	0.0	0.7	0.6	-4.4	-0.5	1
448	37	0	190	85	100	137	361	...	0.0	0.0	0.4	2.4	38.0	62.4	10
449	36	0	166	68	108	176	365	...	0.0	0.0	1.5	1.0	-44.2	-33.2	2
450	32	1	155	55	93	106	386	...	0.0	0.0	0.5	2.4	25.0	46.6	1
451	78	1	160	70	79	127	364	...	0.0	0.0	0.5	1.6	21.3	32.8	1

[452 rows x 280 columns]

- Làm sạch và tiền xử lý dữ liệu tương tự như ở mục 1
- Rời rạc hóa mỗi thuộc tính số hóa thành
 - 10 equi-width ranges
 - 10 equi-depth ranges

3. PCA:

Download the Musk data set from the UCI Machine Learning Repository [213]. Apply PCA to the data set, and report the eigenvectors and eigenvalues.

4. Yêu cầu:

- Cài đặt và thực thi mục 1 trên máy tính
- Làm tiếp những chỗ chưa hoàn chỉnh ở mục 2
- Sử dụng mục 1 để làm sạch dữ liệu và tiền xử lý dữ liệu, và sử dụng PCA trong thư viện sklearn để làm mục 3.
- Viết file báo cáo.