Open in Colab

CHỦ ĐỀ: CHUẨN BỊ DỮ LIỆU

1. Mục tiêu
2. Đọc dữ liệu
3. LỰa chọn dữ liệu
4. Lọc dữ liệu
5. Lọc các giá trị sai
6. Thao tác dữ liệu
7. Sắp xếp dữ liệu
8. Biểu diễn trực quan dữ liệu

Đọc dữ liệu

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
```

Đọc tệp dữ liệu từ file csv

```
edu =pd.read_csv('educ_figdp_Data.csv',na_values=':',
   usecols=['TIME', 'GEO', 'Value'])

# xem dữ liệu
edu
```

Out[3]:

	TIME	GEO	Value
0	2000	European Union (28 countries)	NaN
1	2001	European Union (28 countries)	NaN
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
...	...	...	...
379	2007	Finland	5.90
380	2008	Finland	6.10
381	2009	Finland	6.81
382	2010	Finland	6.85
383	2011	Finland	6.76

> Chú ý:

- na_values là từ khóa biểu diễn dữ liệu không có (trống)
- để đọc file ta còn có các read_excel(), read_table(), read_clipboard()

Đọc dữ liệu đầu và cuối tệp

```
edu.head() #đầu
edu.tail() #cuối
```

Hàm miêu tả dữ liệu

- Xem dữ liệu:
  - Đếm
  - std (độ lệch)
  - xem giá trị nhỏ nhất (**min**)
  - xem giá trị nhỏ nhất (**max**)

- xem số phần trăm dữ liệu

```
edu.describle()
```

	TIME	Value
count	384.000000	361.000000
mean	2005.500000	5.203989
std	3.456556	1.021694
min	2000.000000	2.880000
25%	2002.750000	4.620000
50%	2005.500000	5.060000
75%	2008.250000	5.660000
max	2011.000000	8.810000

Name: Value, dtype: float64

Lựa chọn dữ liệu

- Mục tiêu:
  - chọn thuộc tính/cột trên dữ liệu để đưa vào bài toán học máy
  - chọn dữ liệu có ý nghĩa
- Ví dụ chọn cột **Value**

```
edu ['Value']
```

Out[5]:

```
0    NaN
1    NaN
2    5.00
3    5.03
4    4.95
...  ...
380 6.10
381 6.81
382 6.85
383 6.76
Name: Value, dtype: float64
```

- Lựa chọn dữ liệu theo dòng

```
# chọn dữ liệu từ dòng 15 đến dòng 17
edu[15:17]
# tạo ra bảng có các chỉ số dòng và cột
edu.ix[90:94, ['TIME'], ['GEO']]
```

Out[7]:

	TIME	GEO
90	2006	Belgium
91	2007	Belgium
92	2008	Belgium
93	2009	Belgium
94	2010	Belgium

Lọc dữ liệu

- Lọc dữ liệu thỏa mãn yêu cầu

```
edu[edu['Value']>6.5]
# xem 5 dòng cuối
```

> Chú ý: các phép toán >=, ⇐, !=

Lọc dữ liệu sai

- Mục tiêu: Tìm kiếm và loại các dữ liệu sai, NaN là một kiểu giá trị số thực động, sử dụng phương thức isnull()

```
edu[edu['Value'].isnull].head()
```

Thao tác với dữ liệu

- Mục tiêu: Lựa chọn dữ liệu mong muốn, sử dụng các hàm có sẵn trong python

1. count() # đếm các mẫu không trống
2. sum() # tính tổng các giá trị
3. mean() # trung bình các giá trị
4. median() phép toán tính các giá trị trung bình
5. min() # nhỏ nhất
6. max()
7. prod() # tích vô hướng
8. std() # độ lệch chuẩn
9. var() # phương sai

Ví dụ cách sử dụng:

```
edu[edu['Value'].max()].head()
```

```
data1 = edu["Value"]/100 # giá trị Value trên mỗi dòng sẽ được chia cho 100
data1.head()
```

Áp dụng một phép toán trên một trường

```
data2 = edu["Value"].apply(np.sqrt) # lấy căn bậc 2
data2.head()
```

```
# lamda là từ khóa định nghĩa một hàm mà người dùng tự định nghĩa
data3 = edu["Value"].apply(lamda binhphuong: binhphuong**2)
data3.head()
```

Thêm một cột mới sử dụng tham chiếu đến kết quả từ cột đã có

```
edu['vNorm'] = edu['Value']/edu['Value'].max()
# thêm cột 'vNorm' vào edu
```

	TIME	GEO	Value	vNorm
379	2007	Finland	5.90	0.669694
380	2008	Finland	6.10	0.692395
381	2009	Finland	6.81	0.772985
382	2010	Finland	6.85	0.777526
383	2011	Finland	6.76	0.767310

Loại bỏ những dòng có dữ liệu sai

```
data =edu.dropna()
data
```

Thêm một dòng dữ liệu

```
edu = edu.append({'TIME': 2001, 'Value': 5.00, 'GEO': 'a'},
    ignore_index=True)
# xem dữ liệu được thêm vào
edu.tail()
```

Loại bỏ cột trên dữ liệu

```
edu.drop('vNorm', axis=1, inplace=True)
# xem dữ liệu
edu.head()
```

> Chú ý: axis = 0 nghĩa là loại bỏ dòng, axis = 1 là loại bỏ cột

Loại bỏ dữ liệu trên dòng theo điều kiên

```
#loại bỏ dòng cuối cùng của dữ liệu
edu.drop(max(edu.index), axis=0, inplace=True)
# xem dữ liệu
edu.tail()
```

Loại bỏ dòng sử dụng từ khóa any

- Ví dụ: để loại bỏ các giá trị NaN, thay vì sử dụng hàm dropna(), ta muốn bất kì dòng nào có chứa NaN sẽ được loại bỏ và lưu sang một tập con.

```
edu2 = edu.dropna(how='any', subset=['Value],axis=0)
# xem dữ liệu
edu.head()
```

	TIME	GEO	Value
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
5	2005	European Union (28 countries)	4.92
6	2006	European Union (28 countries)	4.91

Thay thế dòng chứa dữ liệu NaN bởi giá trị khác

```
edu3 = edu.fillna(value ={'Value':0})
# xem dữ liệu
edu3.head()
```

> value: là tham số

	TIME	GEO	Value
0	2000	European Union (28 countries)	0.00
1	2001	European Union (28 countries)	0.00
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95

Sắp xếp dữ liệu

- Mục tiêu: Sắp xếp dữ liệu trên DataFrame theo cột, có thể sử dụng bất kì cột nào, theo chiều tăng dần (giảm dần)

```
edu.sort_values(by ='Value', ascending =False,inplace=True)
#xem dữ liệu
edu.head()
```

> Chú ý: by ='Value' là tham số chỉ tên cột cần được sắp xếp

	TIME	GEO	Value
130	2010	Denmark	8.81
131	2011	Denmark	8.75
129	2009	Denmark	8.74
121	2001	Denmark	8.44
122	2002	Denmark	8.44

Sắp xếp dữ liệu theo chỉ số

- Mục tiêu: khôi phục sắp xếp theo dữ liệu gốc bởi các chỉ số

```
edu.sort_index(axis=0, ascending=True, inplace=True)
#xem
edu.head()
```

Nhóm dữ liệu

- Mục tiêu: Nhóm dữ liệu theo tiêu chí
- Ví dụ: Ta muốn nhóm các quốc gia và tính trung bình Value

```
data_group = edu[edu['GEO', 'Value']].groupby('GEO').mean()
#xem
data_group.head()
```

	Value
GEO	
Austria	5.618333
Belgium	6.189091
Bulgaria	4.093333
Cyprus	7.023333
Czech Republic	4.16833

Tổ chức lại dữ liệu

- Mục tiêu: Xây dựng bảng dữ liệu mới theo yêu cầu mới
- Ví dụ: Xây dựng bảng dữ liệu mới lọc dữ liệu có thời gian lớn hơn 2005, được tổ chức theo chỉ số dòng là 'GEO', chỉ số cột là 'TIME'

```
filter_data = edu[edu["TIME"]>2005]
pivot_edu = dp.pivot_table(filter_data,values = 'Value',
    index = ['GEO'],
    columns = ['TIME'])
# xem
pivot_edu.head()
```

> Chú ý: filter_data là bảng dữ liệu đã lọc theo tiêu chí

Trích dữ liệu trong bảng

- Mục tiêu: lấy dữ liệu theo tiêu chí dòng bởi các nhãn, sử dụng phép toán **ix**
- Ví dụ: lấy dữ liệu 'Value' năm 2006 và 2011 của nước Spain và Portuagal

```
pivot_edu.ix[['Spain', 'Portuagal'], [2006,2011]]
```

TIME	2006	2011
GEO		
Spain	4.26	4.82
Portugal	5.07	5.27

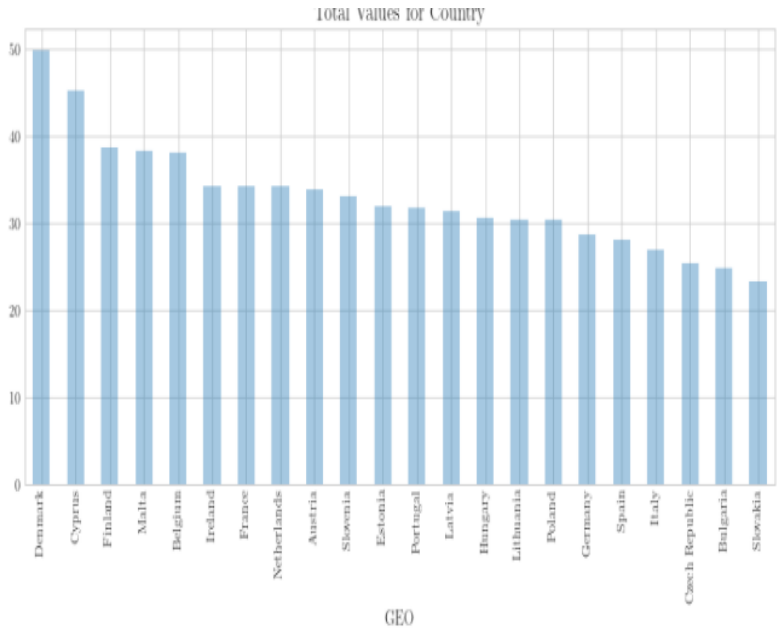
Xếp hạng dữ liệu theo tiêu chí

TIME	2006	2007	2008	2009	2010	2011
GEO						
Austria	10.0	7.0	11.0	7.0	8.0	8.0
Belgium	5.0	4.0	3.0	4.0	5.0	5.0
Bulgaria	21.0	21.0	20.0	20.0	22.0	22.0
Cyprus	2.0	2.0	2.0	2.0	2.0	3.0
Czech Republic	19.0	20.0	21.0	21.0	20.0	19.0

```
pivot_edu = pivot_edu.drop(['Euro area (13 countries)',

                            'Euro area (15 countries)',
                            'Euro area (17 countries)',
                            'Euro area (18 countries)',
                            'European Union (25 countries)',
                            'European Union (27 countries)',
                            'European Union (28 countries)'
                            ], axis=0)
pivot_edu =  pivot_edu.rename(index ={'Germany (until 1990 former territory of the FRG)': 'Germany'})

pivot_edu = pivot_edu.dropna()
pivot_edu.rand(ascending=False, method='first')
#xem
pivot_edu.head()
```

Vẽ biểu diễn dữ liệu

```
fig = plt.figure(figsize(12,5)) #khơi tạo
total_dataSum = pivot_edu.sum(axis=1).sort_values(ascending=False)

#vẽ dữ liệu
total_dataSum.plot(kind ='bar', style ='b', alpha = 0.4,
                   title = "Tổng giá trị cho mỗi nước")

# lưu biểu đồ sang ảnh
plt.savefig('Ten_Anh.png', dpi=300, bbox_inches='tight')
```



CHỦ ĐỀ: CHUẨN BỊ DỮ LIỆU

1. Mục tiêu
2. Đọc dữ liệu
3. LỰa chọn dữ liệu
4. Lọc dữ liệu
5. Lọc các giá trị sai
6. Thao tác dữ liệu
7. Sắp xếp dữ liệu
8. Biểu diễn trực quan dữ liệu

Đọc dữ liệu

```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
```

Đọc tệp dữ liệu từ file csv

```
edu =pd.read_csv('educ_figdp_Data.csv',na_values=':',
   usecols=['TIME', 'GEO', 'Value'])

# xem dữ liệu
edu
```

Out[3]:

	TIME	GEO	Value
0	2000	European Union (28 countries)	NaN
1	2001	European Union (28 countries)	NaN
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
...	...	...	...
379	2007	Finland	5.90
380	2008	Finland	6.10
381	2009	Finland	6.81
382	2010	Finland	6.85
383	2011	Finland	6.76

> Chú ý:

- na_values là từ khóa biểu diễn dữ liệu không có (trống)
- để đọc file ta còn có các read_excel(), read_table(), read_clipboard()

Đọc dữ liệu đầu và cuối tệp

```
edu.head() #đầu
edu.tail() #cuối
```

Hàm miêu tả dữ liệu

- Xem dữ liệu:
  - Đếm
  - std (độ lệch)
  - xem giá trị nhỏ nhất (**min**)
  - xem giá trị nhỏ nhất (**max**)
  - xem số phần trăm dữ liệu

```
edu.describle()
```

	TIME	Value
count	384.000000	361.000000
mean	2005.500000	5.203989
std	3.456556	1.021694
min	2000.000000	2.880000
25%	2002.750000	4.620000
50%	2005.500000	5.060000
75%	2008.250000	5.660000
max	2011.000000	8.810000

Name: Value, dtype: float64

Lựa chọn dữ liệu

- Mục tiêu:
  - chọn thuộc tính/cột trên dữ liệu để đưa vào bài toán học máy
  - chọn dữ liệu có ý nghĩa
- Ví dụ chọn cột **Value**

```
edu ['Value']
```

Out[5]:

```
0    NaN
1    NaN
2   5.00
3   5.03
4   4.95
...  ...
380 6.10
381 6.81
382 6.85
383 6.76
Name: Value, dtype: float64
```

- Lựa chọn dữ liệu theo dòng

```
# chọn dữ liệu từ dòng 15 đến dòng 17
edu[15:17]
# tạo ra bảng có các chỉ số dòng và cột
edu.ix[90:94, ['TIME'], ['GEO']]
```

Out[7]:

	TIME	GEO
90	2006	Belgium
91	2007	Belgium
92	2008	Belgium
93	2009	Belgium
94	2010	Belgium

Lọc dữ liệu

- Lọc dữ liệu thỏa mãn yêu cầu

```
edu[edu['Value']>6.5]
# xem 5 dòng cuối
```

> Chú ý: các phép toán >=, ⇐, !=

Lọc dữ liệu sai

- Mục tiêu: Tìm kiếm và loại các dữ liệu sai, NaN là một kiểu giá trị số thực động, sử dụng phương thức isnull()

```
edu[edu['Value'].isnull].head()
```

Thao tác với dữ liệu

- Mục tiêu: Lựa chọn dữ liệu mong muốn, sử dụng các hàm có sẵn trong python

1. count() # đếm các mẫu không trống
2. sum() # tính tổng các giá trị
3. mean() # trung bình các giá trị
4. median() phép toán tính các giá trị trung bình
5. min() # nhỏ nhất
6. max()
7. prod() # tích vô hướng
8. std() # độ lệch chuẩn
9. var() # phương sai

Ví dụ cách sử dụng:

```
edu[edu['Value'].max()].head()
```

```
data1 = edu["Value"]/100 # giá trị Value trên mỗi dòng sẽ được chia cho 100
data1.head()
```

Áp dụng một phép toán trên một trường

```
data2 = edu["Value"].apply(np.sqrt) # lấy căn bậc 2
data2.head()
```

```
# lamda là từ khóa định nghĩa một hàm mà người dùng tự định nghĩa
data3 = edu["Value"].apply(lamda binhphuong: binhphuong**2)
data3.head()
```

Thêm một cột mới sử dụng tham chiếu đến kết quả từ cột đã có

```
edu['vNorm'] = edu['Value']/edu['Value'].max()
# thêm cột 'vNorm' vào edu
```

	TIME	GEO	Value	vNorm
379	2007	Finland	5.90	0.669694
380	2008	Finland	6.10	0.692395
381	2009	Finland	6.81	0.772985
382	2010	Finland	6.85	0.777526
383	2011	Finland	6.76	0.767310

Loại bỏ những dòng có dữ liệu sai

```
data =edu.dropna()
data
```

Thêm một dòng dữ liệu

```
edu = edu.append({'TIME': 2001, 'Value': 5.00, 'GEO': 'a'},
    ignore_index=True)
# xem dữ liệu được thêm vào
edu.tail()
```

Loại bỏ cột trên dữ liệu

```
edu.drop('vNorm', axis=1, inplace=True)
# xem dữ liệu
edu.head()
```

> Chú ý: axis = 0 nghĩa là loại bỏ dòng, axis = 1 là loại bỏ cột

Loại bỏ dữ liệu trên dòng theo điều kiên

```
#loại bỏ dòng cuối cùng của dữ liệu
edu.drop(max(edu.index), axis=0, inplace=True)
# xem dữ liệu
edu.tail()
```

Loại bỏ dòng sử dụng từ khóa any

- Ví dụ: để loại bỏ các giá trị NaN, thay vì sử dụng hàm dropna(), ta muốn bất kì dòng nào có chứa NaN sẽ được loại bỏ và lưu sang một tập con.

```
edu2 = edu.dropna(how='any', subset=['Value],axis=0)
# xem dữ liệu
edu.head()
```

	TIME	GEO	Value
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
5	2005	European Union (28 countries)	4.92
6	2006	European Union (28 countries)	4.91

Thay thế dòng chứa dữ liệu NaN bởi giá trị khác

```
edu3 = edu.fillna(value ={'Value':0})
# xem dữ liệu
edu3.head()
```

> value: là tham số

	TIME	GEO	Value
0	2000	European Union (28 countries)	0.00
1	2001	European Union (28 countries)	0.00
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95

Sắp xếp dữ liệu

- Mục tiêu: Sắp xếp dữ liệu trên DataFrame theo cột, có thể sử dụng bất kì cột nào, theo chiều tăng dần (giảm dần)

```
edu.sort_values(by ='Value', ascending =False,inplace=True)
#xem dữ liệu
edu.head()
```

> Chú ý: by ='Value' là tham số chỉ tên cột cần được sắp xếp

	TIME	GEO	Value
130	2010	Denmark	8.81
131	2011	Denmark	8.75
129	2009	Denmark	8.74
121	2001	Denmark	8.44
122	2002	Denmark	8.44

Sắp xếp dữ liệu theo chỉ số

- Mục tiêu: khôi phục sắp xếp theo dữ liệu gốc bởi các chỉ số

```
edu.sort_index(axis=0, ascending=True, inplace=True)
#xem
edu.head()
```

Nhóm dữ liệu

- Mục tiêu: Nhóm dữ liệu theo tiêu chí
- Ví dụ: Ta muốn nhóm các quốc gia và tính trung bình Value

```
data_group = edu[edu['GEO', 'Value']].groupby('GEO').mean()
#xem
data_group.head()
```

	Value
GEO	
Austria	5.618333
Belgium	6.189091
Bulgaria	4.093333
Cyprus	7.023333
Czech Republic	4.16833

Tổ chức lại dữ liệu

- Mục tiêu: Xây dựng bảng dữ liệu mới theo yêu cầu mới
- Ví dụ: Xây dựng bảng dữ liệu mới lọc dữ liệu có thời gian lớn hơn 2005, được tổ chức theo chỉ số dòng là 'GEO', chỉ số cột là 'TIME'

```
filter_data = edu[edu["TIME"]>2005]
pivot_edu = dp.pivot_table(filter_data,values = 'Value',
    index = ['GEO'],
    columns = ['TIME'])
# xem
pivot_edu.head()
```

> Chú ý: filter_data là bảng dữ liệu đã lọc theo tiêu chí

TIME	2006	2007	2008	2009	2010	2011
GEO						
Austria	5.40	5.33	5.47	5.98	5.91	5.80
Belgium	5.98	6.00	6.43	6.57	6.58	6.55
Bulgaria	4.04	3.88	4.44	4.58	4.10	3.82
Cyprus	7.02	6.95	7.45	7.98	7.92	7.87
Czech Republic	4.42	4.05	3.92	4.36	4.25	4.51

Trích dữ liệu trong bảng

- Mục tiêu: lấy dữ liệu theo tiêu chí dòng bởi các nhãn, sử dụng phép toán **ix**
- Ví dụ: lấy dữ liệu 'Value' năm 2006 và 2011 của nước Spain và Portuagal

```
pivot_edu.ix[['Spain', 'Portuagal'], [2006,2011]]
```

TIME	2006	2011
GEO		
Spain	4.26	4.82
Portugal	5.07	5.27

Xếp hạng dữ liệu theo tiêu chí

TIME	2006	2007	2008	2009	2010	2011
GEO						
Austria	10.0	7.0	11.0	7.0	8.0	8.0
Belgium	5.0	4.0	3.0	4.0	5.0	5.0
Bulgaria	21.0	21.0	20.0	20.0	22.0	22.0
Cyprus	2.0	2.0	2.0	2.0	2.0	3.0
Czech Republic	19.0	20.0	21.0	21.0	20.0	19.0

```
pivot_edu = pivot_edu.drop(['Euro area (13 countries)',

                            'Euro area (15 countries)',
                            'Euro area (17 countries)',
                            'Euro area (18 countries)',
                            'European Union (25 countries)',
                            'European Union (27 countries)',
                            'European Union (28 countries)'
                            ], axis=0)
pivot_edu =  pivot_edu.rename(index ={'Germany (until 1990 former territory of the FRG)': 'Germany'})

pivot_edu = pivot_edu.dropna()
pivot_edu.rand(ascending=False, method='first')
#xem
pivot_edu.head()
```

Vẽ biểu diễn dữ liệu

```
fig = plt.figure(figsize(12,5)) #khởi tạo
total_dataSum = pivot_edu.sum(axis=1).sort_values(ascending=False)

#vẽ dữ liệu
total_dataSum.plot(kind ='bar', style ='b', alpha = 0.4,
                   title = "Tổng giá trị cho mỗi nước")

# lưu biểu đồ sang ảnh
plt.savefig('Ten_Anh.png', dpi=300, bbox_inches='tight')
```