


Class period 16

บทที่ 6 การแสดงผลการกระจายของข้อมูล (ต่อ)
timestamp - datetime


1



Quiz

- https://drive.google.com/drive/folders/1AztVMSDdZwJDHfSI0T51VSCVRYQZ3?fbclid=IwAR1Ta6xSC4Yck3-SPxK4jY2EABxYFW5HTBwt_xFDtDl5IP1N4e4kpunh8
- โหลดไฟล์ example_timestamp.csv
- เปรียบเทียบผลรวมของข้อมูล alpha และ beta ก่อนวันที่ 2 มิถุนายน 2020 และตั้งแต่วันที่ 2 มิถุนายน 2020

2



Timestamp - datetime

timestamp ใช้แปลงรูปแบบข้อมูลวันเดือนปีต่างๆ ให้เป็นรูปแบบ timestamp
การใช้งาน timestamp จะช่วยให้สามารถใช้ในการชี้หรือเรียงข้อมูลวันเดือนปีโดยจะต่างจากการชี้หรือเรียงข้อมูลแบบปกติในรูปแบบ string
timestamp จะสามารถกำหนดรูปแบบข้อมูลวันเดือนปีที่ต้องการและแปลงรูปแบบข้อมูลเป็นรูปแบบ timestamp การใช้งานจะยืดหยุ่นกว่า เพราะบางคนอาจจะใช้รูปแบบในการเก็บข้อมูลวันเดือนปีต่างกัน
datetime ใช้ในการชี้ข้อมูลรูปแบบ timestamp

3



ยกตัวอย่างการชี้วันเดือนปีแบบ string ธรรมดา

ตัวอย่างข้อมูลวันเดือนปีจากคอลัมน์ Unnamed: 0 จะเรียงลำดับตามหลักการเรียงของข้อมูล string พื้นฐาน การใช้การชี้หรือการเรียงข้อมูลแบบ string จึงสามารถใช้ได้ เช่น


```
be = df[df['Unnamed: 0'] < '2020-06-01']
be
```

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 2020-05-29 | 8.78 | 24 |
| 1 | 2020-05-30 | 13.00 | 25 |
| 2 | 2020-05-31 | 0.44 | 25 |

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 2020-05-29 | 8.78 | 24 |
| 1 | 2020-05-30 | 13.00 | 25 |
| 2 | 2020-05-31 | 0.44 | 25 |

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 2020-05-29 | 8.78 | 24 |
| 1 | 2020-05-30 | 13.00 | 25 |
| 2 | 2020-05-31 | 0.44 | 25 |
| 3 | 2020-06-01 | 1.94 | 28 |
| 4 | 2020-06-02 | 5.40 | 20 |
| 5 | 2020-06-03 | 5.68 | 21 |
| 6 | 2020-06-04 | 2.64 | 16 |

4



ยกตัวอย่างการชี้วันเดือนปีแบบ string ธรรมดา

เปลี่ยนรูปแบบวันเดือนปี และลองชี้และเรียงแบบ string

```
be = df[df['Unnamed: 0'] < '02-06-2020']
be
```

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 3 | 01-06-2020 | 1.94 | 28 |

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 29-05-2020 | 8.78 | 24 |
| 1 | 30-05-2020 | 13.00 | 25 |
| 2 | 31-05-2020 | 0.44 | 25 |
| 3 | 01-06-2020 | 1.94 | 28 |
| 4 | 02-06-2020 | 5.40 | 20 |
| 5 | 03-06-2020 | 5.68 | 21 |
| 6 | 03-06-2020 | 2.64 | 16 |

จะเห็นว่า เรียงไม่ถูก เพราะการเรียงแบบพื้นฐานของ string จะเรียงจากหน้าไปหลัง, 00 ถึง 10 หรือ 000 ถึง 999
ถ้าเป็นตัวอย่างจะเริ่มจาก Aa หรือ กก ดังนั้นจึงมีแค่ 01-06-2020 ที่น้อยกว่า 02-06-2020

5



การใช้ timestamp (.to_datetime)

- การใช้งาน
 - import pandas as pd
 - pd.to_datetime('ข้อมูลคอลัมน์ที่ต้องการ', format='%d-%m-%Y')
 - โดย Input: format จะใช้กำหนดรูปแบบวันเดือนปีของข้อมูล input
 - %d คือ วัน
 - %m คือ เดือน
 - %Y คือ ปี
 - รูปแบบข้อมูลในคอลัมน์คือ 01-06-2020 format='%d-%m-%Y'
 - รูปแบบข้อมูลในคอลัมน์คือ 2020-06-01 format='%Y-%m-%d'
 - รูปแบบข้อมูลในคอลัมน์คือ 01/06/2020 format='%d/%m/%Y'

6

ดึงข้อมูลตัวอย่าง

```

import pandas as pd
df = pd.read_csv('/content/example_timestamp_03.csv')
df

```

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 29/05/2020 | 8.78 | 24 |
| 1 | 30/05/2020 | 13.00 | 25 |
| 2 | 31/05/2020 | 0.44 | 25 |
| 3 | 1/6/2020 | 1.94 | 28 |
| 4 | 2/6/2020 | 5.40 | 20 |
| 5 | 3/6/2020 | 5.68 | 21 |
| 6 | 3/6/2020 | 2.64 | 16 |

df.iloc[0,0] → '29/05/2020'

7

ตัวอย่างการใช้ .to_datetime แปลงรูปแบบข้อมูล

```

df['Unnamed: 0'] = pd.to_datetime(df['Unnamed: 0'], format='%d/%m/%Y')
df

```

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 2020-05-29 | 8.78 | 24 |
| 1 | 2020-05-30 | 13.00 | 25 |
| 2 | 2020-05-31 | 0.44 | 25 |
| 3 | 2020-06-01 | 1.94 | 28 |
| 4 | 2020-06-02 | 5.40 | 20 |
| 5 | 2020-06-03 | 5.68 | 21 |
| 6 | 2020-06-03 | 2.64 | 16 |

df.iloc[0,0] → Timestamp('2020-05-29 00:00:00')

8

การซึ่ข้อมูล timestamp ด้วย datetime

- การซึ่ข้อมูล timestamp จะใช้คำสั่ง datetime สามารถกำหนดวันเดือนปีในการซึ่
- datetime(day=วันที่ต้องการซึ่, month=เดือนที่ต้องการซึ่, year=ปีที่ต้องการซึ่)
- เช่น

```

from datetime import datetime
df[df['Unnamed: 0'] < datetime(day=2, month=6, year=2020)]

```

| | Unnamed: 0 | alpha | beta |
|---|------------|-------|------|
| 0 | 2020-05-29 | 8.78 | 24 |
| 1 | 2020-05-30 | 13.00 | 25 |
| 2 | 2020-05-31 | 0.44 | 25 |
| 3 | 2020-06-01 | 1.94 | 28 |

9

การใช้ .sum() รวมข้อมูลตาม timestamp

- ซึ่ข้อมูล timestamp ที่ต้องการโดยใช้คำสั่ง datetime ตามด้วย .sum() และกำหนดตัวแปรเก็บค่า เช่น
- A = data[data['Unnamed: 0'] < datetime(day=2, month=6, year=2020)].sum()
- print(A)

```

alpha    24.16
beta    102.00
dtype: float64

```

A['alpha']

```

24.160000000000004

```

10

การใช้ .sum() รวมข้อมูลตาม timestamp

- ซึ่ข้อมูล timestamp ที่ต้องการโดยใช้คำสั่ง datetime ตามด้วย .sum() และกำหนดตัวแปรเก็บค่า เช่น
- BF = data[data['Unnamed: 0'] < datetime(day=2, month=6, year=2020)].sum()
- AT = data[data['Unnamed: 0'] >= datetime(day=2, month=6, year=2020)].sum()
- print(BF)

```

alpha    24.16
beta    102.00
dtype: float64

```

print(AT)

```

alpha    13.72
beta     57.00
dtype: float64

```

11