



Class period 7

บทที่ 5 การเตรียมข้อมูลสำหรับการแสดงผล 1

Pandas 101 part2



เฉลย Homework class period 6

- คำนวณ อายุเฉลี่ย ของผู้หญิง และผู้ชาย ของข้อมูลทั้งหมด
- `this_data = data_covid[['sex', 'age', 'province_of_onset']]`
- `this_data`
- เลือกข้อมูลเฉพาะคอลัมน์ที่ต้องการใช้งานและเก็บไว้ในตัวแปร `this_data`
- `female = this_data[this_data['sex']=='หญิง']`
- เลือกแถวข้อมูลที่มีข้อมูลในคอลัมน์ `sex` เท่ากับ หญิง เก็บไว้ในตัวแปร `female`
- `female['age']`
- เลือกแสดงข้อมูลในตัวแปร `female` เฉพาะคอลัมน์ `age` ก็จะได้ข้อมูลอายุของเพศหญิงทั้งหมด



เฉลย Homework class period 6

- จากนั้นวนลูปเพื่อหาอายุเฉลี่ย
- `sum = 0`
- `N = 0`
- `for a in female['age']:`
- `if a > 0:`
- `sum += a # sum = sum + a`
- `N += 1`
- `print(f'อายุเฉลี่ย ของ ผู้ป่วยหญิง {sum/N}')`
- กำหนดตัวแปร `sum=0` และ `N=0` เพื่อใช้ในการเก็บค่าจากการบวกในการวนลูปแต่ละรอบจนถึงรอบสุดท้าย โดย `sum` จะใช้เก็บค่าอายุ และ `N` ใช้เก็บค่าจำนวนผู้หญิง



เฉลย Homework class period 6

- `sum = 0`
- `N = 0`
- กำหนดตัวแปร `sum=0` และ `N=0` เพื่อใช้ในการเก็บค่าจากการบวกในการวนลูปแต่ละรอบจนถึงรอบสุดท้าย โดย `sum` จะใช้เก็บค่าอายุ และ `N` ใช้เก็บค่าจำนวนผู้หญิง
- `for a in female['age']:`
- วนลูปอ่านค่าอายุของผู้หญิงที่แต่ละคนเก็บไว้ในตัวแปร `a`
- `if a > 0:`
- ตั้งเงื่อนไขในค่าอายุมากกว่า 0 ถึงจะนำค่าอายุมาบวกคำนวณหาค่าเฉลี่ย เพื่อหลีกเลี่ยงค่า `missing`(บางคนไม่มีข้อมูลอายุ)



เฉลย Homework class period 6

- `sum += a` # `sum = sum + a`
- `N += 1`
- นำตัวแปร `sum` มาบวกค่าอายุของผู้หญิงทีละคน จบรูป 1 รอบก็จะเอาผลลัพธ์จากการบวกรอบที่แล้วมาบวกต่อไปเรื่อยๆ เพื่อหาค่าอายุรวม
- นำตัวแปร `N` มาบวก 1 เพื่อใช้นับจำนวนผู้หญิง
- `print(f'อายุเฉลี่ย ของ ผู้ป่วยหญิง {sum/N}')`
- นำตัวแปร `sum` และ `N` มาหารกันเพื่อหาค่าเฉลี่ย ผลลัพธ์จะได้



การจัดการ Missing Value

- มีทั้งหมด 3 แบบ
- 1. ลบ record ที่เป็น missing
- 2. แทนที่ ค่า missing ด้วยค่าที่เหมาะสม mean, default, category-unknown
- 3. ใช้ ค่าจาก columns อื่นๆ ช่วยประมาณค่า ค่าใน column ที่หายไป (regression, deep learning, etc.)



ลบ record (dropna)

- missing = None, NA(not authorized), NaN (not a number)
- .dropna() เป็นคำสั่งที่ใช้ในการลบข้อมูลแถวที่ไม่มีค่าหรือไม่มีข้อมูล ตัวอย่างเช่น
- `data_covid.shape` ผลลัพธ์จะได้ขนาดของข้อมูล data_covid
- `(839771, 11)`
- `data_covid.dropna().shape` ผลลัพธ์จะได้ขนาดของข้อมูล data_covid ที่ลบแถวข้อมูลที่มีค่าเป็น None
- `(599988, 11)`

การใช้งาน .dropna()



- สามารถเลือกลบข้อมูลที่เป็น None เฉพาะในคอลัมน์ที่ต้องการใช้งาน แทนที่จะเลือกลบจากข้อมูลทั้งหมด เช่น
- `this_data = data_covid[['sex', 'age', 'province_of_onset']]`
- `this_data.shape` ผลลัพธ์จะได้
- `(839771, 3)`
- `this_data.dropna().shape` ผลลัพธ์จะได้
- `(674906, 3)`
- จะเห็นว่าเมื่อเทียบกับ `data_covid.dropna().shape` ที่เป็นข้อมูลทั้งหมด `(599988, 11)`
- ข้อมูลที่เลือกเฉพาะคอลัมน์ที่ต้องการใช้งานจะมีจำนวนข้อมูลมากกว่า

การใส่ตัวแปรเพื่อรับค่า



- `this_data.dropna()`
- `print(this_data.shape)` ผลลัพธ์จะได้
- `(839771, 3)`
- ซึ่งไม่ใช่ผลลัพธ์ที่ได้จากการใช้ `.dropna()` เพื่อลบข้อมูลแถวที่มีค่าเป็น `None` เนื่องจากไม่ได้มีตัวแปรเข้ารับค่า เช่น
- `This_data_dn = this_data.dropna()`
- `print(This_data_dn.shape)` ผลลัพธ์จะได้
- `(674906, 3)`



Parameter: inplace ของ .dropna()

- inplace จะเป็นการอัปเดตค่าในตารางเลย โดยที่ไม่จำเป็นต้องมีตัวแปรที่มารับค่า เช่น
- `this_data.dropna(inplace=True)`
- `print(this_data.shape)`
- `(674906, 3)`

pandas.DataFrame.dropna

```
DataFrame.dropna(*, axis=0, how=_NoDefault.no_default, thresh=_NoDefault.no_default, subset=None, inplace=False, ignore_index=False) \[source\]
```

Remove missing values.

See the [User Guide](#) for more on which values are considered missing, and how to work with missing data.

Parameters:

axis : {0 or 'index', 1 or 'columns'}, default 0

Determine if rows or columns which contain missing values are removed.

- 0, or 'index' : Drop rows which contain missing values.
- 1, or 'columns' : Drop columns which contain missing value.

Only a single axis is allowed.

how : {'any', 'all'}, default 'any'

Determine if row or column is removed from DataFrame, when we have at least one NA or all NA.

- 'any' : If any NA values are present, drop that row or column.
- 'all' : If all values are NA, drop that row or column.

thresh : int, optional

Require that many non-NA values. Cannot be combined with how.

subset : column label or sequence of labels, optional

Labels along other axis to consider, e.g. if you are dropping rows these would be a list of columns to include.

inplace : bool, default False

Whether to modify the DataFrame rather than creating a new one.

ignore_index : bool, default False

If True, the resulting axis will be labeled 0, 1, ..., n - 1.

Parameter: subset ของ .dropna()

- subset จะเป็นการเลือกเฉพาะคอลัมน์ที่ต้องการลบแถวข้อมูลที่เป็น None เฉพาะคอลัมน์ที่เลือก เช่น
- `this_data = data_covid[['sex', 'age', 'province_of_onset']]`
- `this_data.shape` ขนาดของข้อมูลตาราง
- `(839771, 3)`
- `this_data.dropna().shape` ขนาดของข้อมูลตารางที่มีการลบข้อมูลแถวที่เป็น None แบบปกติ
- `(674906, 3)`
- `this_data.dropna(subset=['age']).shape`
- ขนาดของข้อมูลตารางที่มีการกำหนด subset ลบข้อมูลแถวที่เป็น None เฉพาะ subset ที่กำหนด
- `(763606, 3)`

แทน missing ด้วยค่าที่เหมาะสม .fillna()

- .fillna() เป็นคำสั่งที่ใช้ในการแทนที่ค่า missing หรือค่า None ด้วยค่าที่กำหนด เช่น
- `this_data = data_covid[['sex', 'age', 'province_of_onset']]`
- `this_data_updated = this_data.fillna(value={'sex': 'ไม่รู้', 'age': -1, 'province_of_onset': 'ไม่รู้'})`
- หมายความว่า ให้แทนที่ข้อมูลที่เป็น None
- ในคอลัมน์ sex แทนด้วย 'ไม่รู้'
- ในคอลัมน์ age แทนด้วย -1
- และในคอลัมน์ province_of_onset ให้แทนที่ค่า None ด้วย 'ไม่รู้'
- ผลลัพธ์จะได้ข้อมูลตาราง this_data_updated ที่ไม่มีค่า None



แทน missing ด้วยค่าที่เหมาะสม .fillna()

- สามารถตรวจสอบว่าค่า None ถูกแทนที่ด้วยค่าที่กำหนดไว้แล้วรียัง
- `this_data_updated[this_data_updated['sex']=='ไม่รู้']`
- ผลลัพธ์จะเห็นว่าข้อมูลในคอลัมน์ sex ที่ถูกแทนที่ด้วย 'ไม่รู้'

`this_data_updated[this_data_updated['sex']=='ไม่รู้']`

	sex	age	province_of_onset
4397	ไม่รู้	-1	สมุทรสาคร
4398	ไม่รู้	-1	สมุทรสาคร
4399	ไม่รู้	-1	สมุทรสาคร
4400	ไม่รู้	-1	สมุทรสาคร
4401	ไม่รู้	-1	สมุทรสาคร
...
838332	ไม่รู้	3.0	กรุงเทพมหานคร
839174	ไม่รู้	47.0	กรุงเทพมหานคร
839304	ไม่รู้	-1	ไม่รู้
839396	ไม่รู้	-1	นครราชสีมา
839397	ไม่รู้	-1	นครราชสีมา

54895 rows x 3 columns

การใช้ logical expression จากข้อมูลตารางอื่น

- `data_covid[this_data_updated['province_of_onset']=='ไม่รู้']`
- จะเห็นได้ว่าในส่วนที่กำหนดเงื่อนไขของ logical expression มาจากตัวแปร `this_data_updated` ซึ่งนำมาใช้ในข้อมูลตารางของตัวแปร `data_covid`
- สาเหตุที่สามารถนำมาใช้ด้วยกันได้และผลออกมาถูกต้อง 2 ตัวแปรที่เป็นข้อมูลตารางต้องมีจำนวนแถวเท่ากันและในแต่ละแถวมีข้อมูลเหมือนกันตำแหน่งเดียวกัน
- ถ้าหาก 2 ตัวแปรที่มีจำนวนแถวเท่ากันแต่ในแต่ละแถวมีข้อมูลไม่เหมือนกันก็สามารถใช้งานได้ แต่ผลลัพธ์ที่ได้อาจจะไม่ถูกต้องเพราะไม่ใช่ข้อมูลเดียวกัน

การวนลูป record ในตาราง .iterrows()

- .iterrows() เป็นคำสั่งที่ช่วยในสามารถวนลูปอ่านข้อมูลในตาราง
- `this_data = data_covid[['sex', 'age', 'province_of_onset']]`
- `for each_row in this_data.iterrows():`
- `if (each_row[1]['age'] == 20) and (each_row[1]['province_of_onset'] == 'ขอนแก่น'):`
- `print(each_row)`
- หมายความว่า
- ให้วนลูปอ่านค่าในข้อมูลตารางตัวแปร this_data ทีละแถวและเก็บในตัวแปร each_row
- ส่วนทำงานภายในลูปกำหนดเงื่อนไขโดยกำหนดให้เลือก print ข้อมูลเฉพาะแถวที่มีข้อมูลในตาราง age=20 และ province_of_onset=ขอนแก่น