Tkinter and Pandas

授課老師：邱淑怡

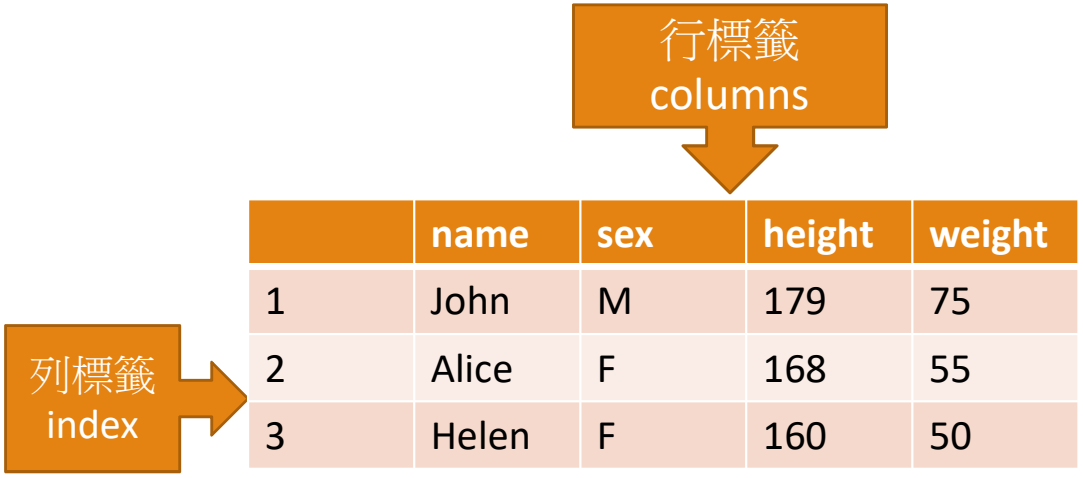
DATE:11/29/2024

Outline

- Pandas module
  - Introduce dataframe
  - Dataframe operation
  - 10 minutes to pandas (https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html)
- Combine tkinter(user interface) with pandas(select data) for recommendation system/query

DataFrame

- Stored integer, float, string .....
- 2-dimensional type
- Dataframe contains 3 parts:
  1. index(列標籤)
  2. columns(行標籤)
  3. data



	name	sex	height	weight
1	John	M	179	75
2	Alice	F	168	55
3	Helen	F	160	50

```
import pandas as pd
```

df1=pd.DataFrame(data, index=None, columns=None,其他選擇性參數), data用來指定DataFrame的資料, index用來指定資料的列標籤, column用來指定資料的行標籤(變數名稱)

Basic information about the dataframe

```
display(Df2)

Df2.info()

Df2.describe()

Df2.columns

Df2.sort_values(by='a').head()
```

Read txt, csv, excel url files

1. Excel file: read_df = pd.read_excel("data2.xlsx",sheet_name='工作表2',header=0, index_col=0)
2. csv file: US2020_df = pd.read_csv("D:\\temp\\governors_county.csv",header=0)
3. File from website: df1=pd.read_csv(url_link)
4. Txt file:
   - txt_url = 'http://people.apache.org/~edwardyoon/kmeans.txt'
   - iris_df = pd.read_table(txt_url, sep = "\t")

Selection

Use indexing or label to select a single/multiple data

Methods	Introduction
dataframe.at	Use row/column labels to select a single data (selection by label)
dataframe.iat	Use 0-based indexing to select a single data (selection by position)
dataframe.loc	Use row/column labels to select multiple data (selection by label)
dataframe.iloc	Use 0-based indexing to select multiple data (selection by position)

Selection data

df1.iat[0,0] #first row and first column

df1.at[1,'column_1']

df1.iloc[0:2,0:3]

df1.loc[0:2,'column_1':'column_3']

Selection multiple data

1. dataframe_name.iloc[0:3,2:5]:
   - 先篩選第幾筆到第幾筆資料，再選欄位
2. per_df.loc[‘2':'4’,column_1’:’column_3’])
   - loc: location
   - first index range, then column_name range

Selection data by columns

- A single column

  df_1[‘column_name’] or df_1.column_name

  ex: df_1.type.tolist() # 將type欄位的資料抓出來轉成list

- multiple columns

  df_1[[‘column_1’,’column_2’]]

Selection by conditions

- a single condition

```
print(df[df['salary']>50000])
```

- multiple conditions

```
c1 = df['class']=='class0'
c2 = df['height'] > 170
temp_df=df[(c1 & c2)]
(or) temp_df1=df[ (df[‘class’]==‘class0’) & (df[‘height’] > 170) ]
```

Delete some data by column/row labels

```
new_df1 = df.drop([“class"], axis=1)
display(new_df1)
new_df2= df.drop([‘3’,‘5’]) # axis預設值為0
new_df2=df2.drop(new_df2.index[0])  #remove first row
new_df2=df2.drop(new_df2.index[-1])  #remove last row
display(new_df2)
# 刪除空值(nan)
new_df3 = df.dropna()
```

Tkinter: Entry (文字方塊)

```
import tkinter
def add_num():
    result.set(num1.get() + num2.get())
win = tkinter.Tk()
win.title('加法視窗程式')
num1=tkinter.DoubleVar()
num2=tkinter.DoubleVar()
result=tkinter.DoubleVar()
```

```
item1=tkinter.Entry(win, width=10, textvariable=num1)
label1=tkinter.Label(win, width=5, text='+')
item2=tkinter.Entry(win, width=10, textvariable=num2)
btn=tkinter.Button(win, width=5, text='=', command=add_num)
label2=tkinter.Label(win, width=10, textvariable=result)
item1.pack(side='left')
label1.pack(side='left')
item2.pack(side='left')
btn.pack(side='left')
label2.pack(side='left')
win.mainloop()
```

Exercise

Question

- Use restaurant_nccu.csv (https://www.cs.nccu.edu.tw/~sichiu/restaurant_nccu.csv)
- 建立查詢介面，查詢的欄位有餐廳類型(type)、設定與餐廳的距離(利用Entry widget 讓使用者輸入距離)
- Show result on messagebox

Example 1 of TreeView

```
import pandas as pd
from tkinter import ttk
import tkinter as tk
def Start():
    fp=pd.read_csv("countries.csv")
    for i in range(fp.shape[0]):
        tree.insert("","end",value=tuple(fp.iloc[i,].values))

win=tk.Tk()
ttk.Button(win, text="Import file",command=Start).pack()
columns=("country","year","population")

tree=ttk.Treeview(win,show="headings",columns=columns)
tree.column("country",width=100,anchor='center')
tree.column("year",width=100,anchor='center')
tree.column("population",width=200,anchor='center')
tree.heading("country",text="country")
tree.heading("year",text="year")
tree.heading("population",text="population")
tree.pack()

win.mainloop()
```

Example 2

```
import tkinter as tk
import tkinter.ttk as ttk  #ttk 加強模組
win = tk.Tk()
win.title('cuteluluWindow')
win.configure(bg="#7AFEC6")
win.geometry('500x200')
tree=ttk.Treeview(win,columns=("節日"))
tree.heading("#0",text="節日")
tree.heading("#1",text="日期")
tree.insert("",index="end",text="國慶日",values="10/10")
tree.insert("",index="end",text="聖誕節",values="12/25")
tree.insert("",index="end",text="元旦",values="1/1")
tree.insert("",index="end",text="愚人節",values="4/1")
tree.insert("",index="end",text="兒童節",values="4/4")
tree.pack()
win.mainloop()
```

Example 3

```
import tkinter as tk
from tkinter import ttk
from tkinter.messagebox import showinfo

root = tk.Tk()
root.title('Treeview demo')
root.geometry('620x200')

# define columns
columns = ('first_name', 'last_name', 'email')
tree = ttk.Treeview(root, columns=columns, show='headings')

# define headings
tree.heading('first_name', text='First Name')
tree.heading('last_name', text='Last Name')
tree.heading('email', text='Email')

# generate sample data
contacts = []   # a list of tuples
for n in range(1, 100):
    contacts.append(('first %d'%n, 'last %d'%n,
'email%d@example.com'%n))

# add data to the treeview
for contact in contacts:
    tree.insert('', tk.END, values=contact)
```

```
def item_selected(event):
    for selected_item in tree.selection():
        item = tree.item(selected_item)
        record = item['values']
        # show a message
        showinfo(title='Information', message=','.join(record))


tree.bind('<<TreeviewSelect>>', item_selected)

tree.grid(row=0, column=0, sticky='nsew')  # place the Treeview widget on the
root window

# add a scrollbar
scrollbar = ttk.Scrollbar(root, orient=tk.VERTICAL, command=tree.yview)
tree.configure(yscroll=scrollbar.set)
scrollbar.grid(row=0, column=1, sticky='ns')

# run the app
root.mainloop()
```