

# Matplotlib topic

## Instructions

- Create new PyCharm project
- Delete unwanted code in main.py
- Go to page 272 / 321
- Go to terminal
  - pip install numpy, pandas, matplotlib
- Go back to main.py
- Import numpy, pandas and pyplot
- Create case06() function in your main.py
- Copy the source code in page 272 to your case06()
- Add plt.show()
- Make sure that main.py call case06()
- Try changing "-10" to "-20"
- Check the content in Series
  - print(x) and print(y)
- Continue 11.1 → Change color of the graph
- About color
  - Goto "color.adobe.com"
  - Try pickup 6-digit color and try it in your code (RGB)
  - #XXXXXX → XX XX XX → x can be 0 1 2 3 ... 9 A B C D E F

Until 11.1.1.2 (Break 10:15 - 10:30)

- Go to page 285 / 321
- Create case07()
- Put the source code in case07()
- Create case08()
- Try Histogram → the frequency of data occur in event
- Create case09()
- Try Barchart
- Create case10()
- Try Stacked bar chart
- Go to 11.1.3.3 page (314 / 321)
- Try Legends
- Save file
  - You need to have your dedicated location on your computer.
  - In my case, the position is "C:\data\output"
  - In mac, it is similar to /Users/<user name>/Desktop/<user name>
  - Do not forget to double backslash (Windows only)

eg

- Data of student score
- so histogram will show which score appear most
- How many ppl walk in to shopping mall

```
main.py
1 import matplotlib.pyplot as plt
2 import numpy as np
3 import pandas as pd
4
5 usage
6 def case06():
7     x = np.arange(-10, 11)
8     y = x ** 2
9     #plt.plot(x, y) #X is list, Y is result of list
10    plt.show() #need to put everytime we plot
11    #print(x) # if going to check number's list '#' plt command
12    #print(y)
13
14    # put '#' for plt. command before or else will erase
15    y_2 = -x ** 2
16    plt.plot('args: x, y, "-") # when wanna show
17    plt.plot('args: x, y_2, "-")
18    plt.show()
19
20 if __name__ == '__main__':
21     case06()

#Scatter plot
x = np.random.rand(30)
y = np.random.rand(30)
z = np.random.rand(30)
colours = np.random.choice(["blue", "black", "red"], 30)
plt.scatter(x, y, marker="o", color=colours, s=z * 100
            size
```

Bins → arrage

Bins = Grouping numbers that is in same categorise  
and set a no. that u want to categorize.

e.g. sales in 100 shops Bins in 100,000 = 1  
0 - 100,000, 100,000 - 200,000 etc...

• (-4, 4, 0.1) → Bins

-4 -- -3.5, -3.5 -- -3 etc...  
category 1

```
#####
# case08 is about Histogram
1 usage
def case08():
    x = np.random.randn(1000)
    plt.hist(x)
    plt.show()
    #print(x) ###for checking list number

if __name__ == '__main__':
    #case06()
    #case07()
    case08()

if __name__ == '__main__':
    #case06()
    #case07()
    case08()
```

```
#####
# case08 is about Histogram
1 usage
def case08():
    x = np.random.randn(1000) ### 这里是随机数字
    bins = np.arange(-4, 4, .1)
    plt.hist(x, bins=bins)
    plt.hist(x)
    plt.show()
    #print(x) ###for checking list number

if __name__ == '__main__':
    #case06()
    #case07()
    case08()
```

```
#####
# case08 is about Histogram
1 usage
def case08():
    x = np.random.randn(1000) ### 这里是随机数字
    bins = np.arange(-4, 4, 0.5)
    plt.hist(x, edgecolor='white', linewidth=1.5, bins=bins)
    plt.hist(x)
    plt.show()
    #print(x) ###for checking list number

if __name__ == '__main__':
    #case06()
    #case07()
    case08()
```

## CHAPTER 11. DATA VISUALIZATION

to make it look better

```
##### about Bar chart
1 usage
def case09():
    pays = ["France", "Italie", "Belgique", "Allemagne"]
    unemployment = [9.3, 9.7, 6.5, 3.4]
    plt.bar(pays, unemployment)
    plt.show()

if __name__ == '__main__':
    #case06()
    #case07()
    #case08()
    case09()

# can change according to the dataset
# I'm using > the categories I want to represent
eg, categories = Pays, Represent = unemployment
```

can also use "countries"

Country	Unemployment Rate
France	9.3
Italie	9.7
Belgique	6.5
Allemagne	3.4

list of country names  
"countries."

Breakdown:

```
def case09():
    countries = ["France", "Italie", "Belgique", "Allemagne"]
    unemployment = [9.3, 9.7, 6.5, 3.4]
    plt.bar(countries, unemployment)
    plt.show()

if __name__ == '__main__':
    #case06()
    #case07()
    #case08()
```

Country	Unemployment Rate
France	9.3
Italie	9.7
Belgique	6.5
Allemagne	3.4

```
##### Below is for Several data/series
countries = ["France", "Italie", "Belgique", "Allemagne"]
unemp_f = [9.1, 11.2, 6.4, 2.9] #unemployment rate of Males & Females
unemp_h = [9.5, 9.2, 6.6, 3.8] # Position on the x-axis for each label
position = np.arange(len(countries))

# Bar widths
width = .35
# Creating the figure and a set of subgraphics
fig, ax = plt.subplots()
r1 = ax.bar(position - width / 2, unemp_f, width)
r2 = ax.bar(position + width / 2, unemp_h, width)
# Modification of the marks on the x-axis and their labels
ax.set_xticks(position)
ax.set_xticklabels(countries)
-- = Left side
+ = Right side
```

Country	Males (unemp_f)	Females (unemp_h)
France	9.1	9.5
Italie	11.2	9.2
Belgique	6.4	6.6
Allemagne	2.9	3.8

R1 = Bar for 1st unemployment data set

```
### about Stacked bar chart
1 usage
def case10():
    countries = ["France", "Italie", "Belgique", "Allemagne"]
    no_unemp_f = [1.307, 1.185, .577, .148]
    no_unemp_h = [1.46, 1.338, .878, .179]
    plt.bar(countries, no_unemp_f, bottom=no_unemp_h)
    plt.show()
```

Country	No. Unemployment Females (no_unemp_f)	No. Unemployment Males (no_unemp_h)
France	1.307	1.46
Italie	1.185	1.338
Belgique	.577	.878
Allemagne	.148	.179

To save pictures of each result

```
## Below is Legends
x = np.arange(-10, 11)
y = x ** 2
y_2 = x ** 3
plt.plot(*args, x, y, label="square ($x^2$)")
plt.plot(*args, x, y_2, label="cube ($x^3$)")
plt.legend()
plt.legend()

plt.show()

if __name__ == '__main__':
    x**2 = use
    x**3 = use
```

square = 2  
cube = 3

x	square (\$x^2\$)	cube (\$x^3\$)
-10	100	-1000
-5	25	-125
0	0	0
5	25	125
10	100	1000

```
## Below is Legends
x = np.arange(-10, 11)
y = x ** 2
y_2 = x ** 3
plt.plot(*args, x, y, label="square ($x^2$)")
plt.plot(*args, x, y_2, label="cube ($x^3$"))
plt.legend()
plt.legend()
path = C:\Users\YourName\PycharmProjects\Final practice.png
plt.savefig(path)

if __name__ == '__main__':
    #case06()
    #case07()
    #case08()
    #case09()
    case10()
```

## Json

```
import numpy as np
import pandas as pd

usage
def case11():
    path = 'C:\\Users\\User\\OneDrive\\[删除]\\02-customer_missing.csv'
    df = pd.read_csv(path)
    print(df) —————> checking data (will show after "run")

if __name__ == '__main__':
    case11()
```

```
"C:\Users\User\PycharmProjects\Final practice2\.venv\\
    cust_id          name   gender  national
0      1001  missing data     Nan    Male     USA
1      1002  missing data     Nan    Female    USA
2      1003  missing data     Nan    Male     USA
3      1004  missing data     Nan    Female    USA
4      1005  missing data     Nan    Male     USA
```

— Check here →

# Data Cleaning

w1

But NaN and other types of missing data still there

```
## Data_cleansing
df2 = df.replace( to_replace: '-' , value: 'No data')
#print(df2)

df3 = df2.fillna('No data')
print(df3) ↳ till not a number

0
First roll is "0"

rmProjects\Final practice2\.env\Scripts\python.exe"
    name   gender   national   date_join
  No data     Male      USA 15/05/2022
Jane Smith  Female    Canada 16/05/2022

### Data_cleansing
df2 = df.replace( to_replace: '-' , value: 'No data')
#print(df2)
df3 = df2.fillna('No data')
df4 = df3.replace( to_replace: '--' , value: 'No data')
df5 = df4.replace( to_replace: ' ' , value: 'No data')
print(df5)

) ↳ Continue replace missing data to
```

# Replace command is cleaning "all" but can't specify column

# if want to replace many data → python dicunary { }

W2

```
### For cleaning "Name" column ONLY (in specific column)
1 usage
def case12():
    path = 'C:\\Users\\User\\OneDrive\\[เอกสาร]\\02-customer_missing.csv'
    df = pd.read_csv(path)
    df['name'] = df['name'].fillna('No data')
    df['name'] = df['name'].replace({
        '-': 'No data',
        '--': 'No data',
        ' ': 'No data'})
    print(df)

    " ; when continue putting data
    ↗ dicurary
    ↗ Replace missing data
```

Change for 1 column

The screenshot shows a PyCharm interface with a code editor and a terminal. The code editor has a yellow arrow pointing from the terminal output to the 'case13()' function definition. A blue box highlights the 'from datetime import date' import statement. Handwritten annotations include: 'give birth date → பிறந்து' above the import statement; 'def \_\_\_\_\_():' next to the 'def case13():' line; 'det \_\_\_\_\_():' next to the 'def calculateAge(birthDate):' line; 'Get new source code create new function & paste' with an arrow pointing to the 'calculateAge' function; '# Don't put / merger in ur own's' with an arrow pointing to the same line; and '27 years' in the terminal output.

```
Project ▾ main.py x
Final practice2
  .venv library
    Lib
    Scripts
    .gitignore
    pyvenv.cfg
  main.py
External Libraries
Scratches and C

1 import numpy as np
2 import pandas as pd
3 from datetime import date
4
5 def case13():
6     print(calculateAge(date( year: 1997, month: 2, day: 3)), "years")
7
8     usage
9
10 def calculateAge(birthDate):
11     days_in_year = 365.2425
12     age = int((date.today() - birthDate).days / days_in_year)
13     return age

Run main x
C:UsersUserPycharmProjectsFinal practice2.venvScriptspython.exe C:UsersUserPy
27 years
```

```

def case13():
    print(calculateAge(date( year: 1997, month: 2, day: 3)), "years")
    text = '15/05/2022' → must change to 年,月,日
    cust_id,name,gender,national,date_join
    1001,,Male,USA,15/05/2022
    Usage
    def case13():
        print(calculateAge(date( year: 1997, month: 2, day: 3)), "years")
        text = '15/05/2022'
        y = text[0:4] → get element from 0 to 4 (first four elements)
        m = text[5:7]
        d = text[8:10]
        print(y)
        print(m)
        print(d)
case13()

```

For trying the code only

```

"C:\Users\USER\PycharmProjects\Final practice2\.venv\Scripts\python.exe" "C:/Users/USER/PycharmProjects/Final practice2/main.py"
27 years
2022
05
15

```

## Finding the work duration of the employee

```

### you want to know the work duration of the employee (one by one)

path = 'C:\\\\Users\\\\User\\\\OneDrive\\\\LearnView\\\\02-customer_missing.csv'
df = pd.read_csv(path)
print(df)

→ join date
# below is runningg info 1 by 1 [run into 1 by 1]
for x in df.index:
    jdate = df.at[x, 'date_join'] → pick up 'date join' to jdate 1 by 1

```

data transformation

```

jdate = df.at[x, 'date_join']
y = int(jdate[0:4])
m = int(jdate[5:7])
d = int(jdate[8:10])
duration_value = calculateAge(date(y, m, d))
df.at[x, 'duration'] = duration_value (put value back to data frame)
print(df)

```

joined at year 2



	Daniel Davis	Male	Mexico	2022-06-08
124	Grace Johnson	Female	South Korea	2022-06-09
125	Ethan Garcia	Male	France	2022-06-10
126	Ava Taylor	Female	Germany	2022-06-11
127		Male	India	2022-06-12
128				