HKUSPACE
香港大學專業進修學院
HKU School of Professional and Continuing Education

Business Process Automation with VBA and Python

Lecture 5 – Business Process Automation with Python (Part 3)

Jackie Liu - Jul 2024
College of Business and Finance (CBF)

Course Outline

Lecture	Topics	Activities

#1
07-Sep
 	
#2
14-Sep
 	
#3
21-Sep
 	
#4
28-Sep
 	
#5
05-Oct
	(3) Business process automation with Python
	
	• Fundamentals of Python syntax and related packages
	• Development of user cases
	  • Data source management
	  • Automation development for browser-based functions
	  • Automation development for application-based functions
	  • Automated reporting: tabular and graphic formats
	  • Common algorithm used in automation: fuzzy matching and standard deviation
	• Business and finance applications
		In-class Practice

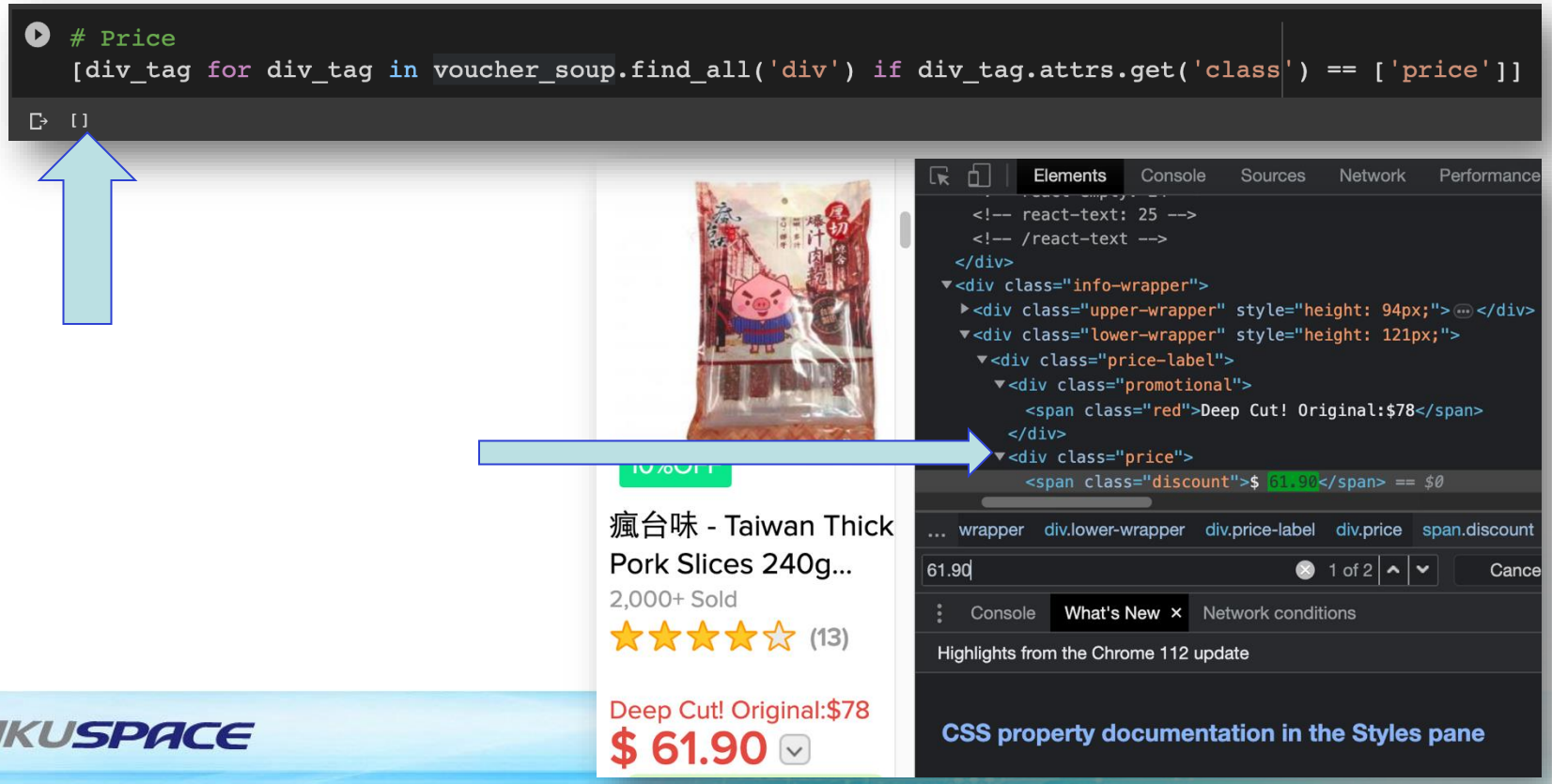
HKUSPACE
香港大學專業進修學院
HKU School of Professional and Continuing Education

Browser Automation

Business Process Automation with Python

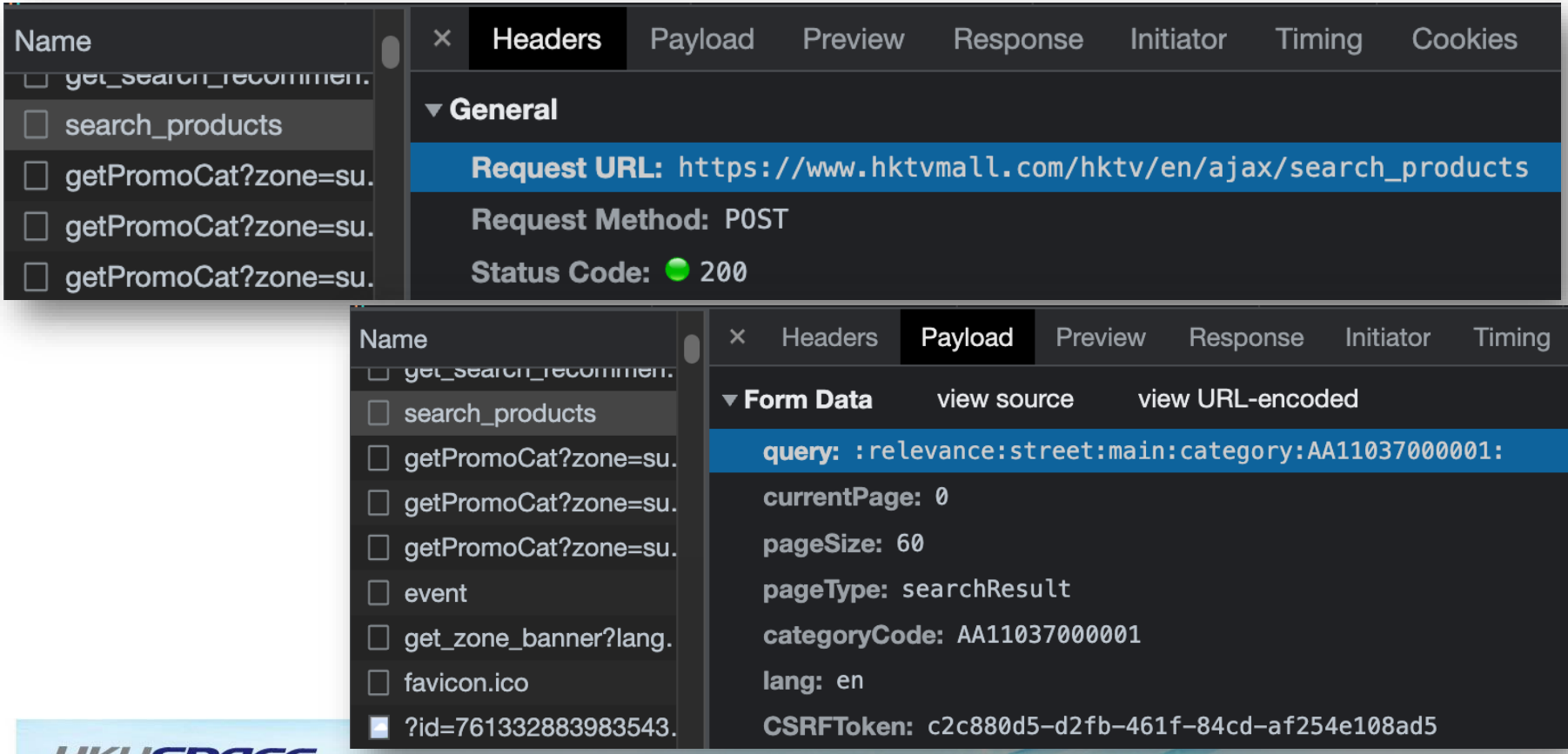
Browser Automation

- Motivation
  - Modern websites may use JavaScript
  - Content are “injected” when browser runs the scripts



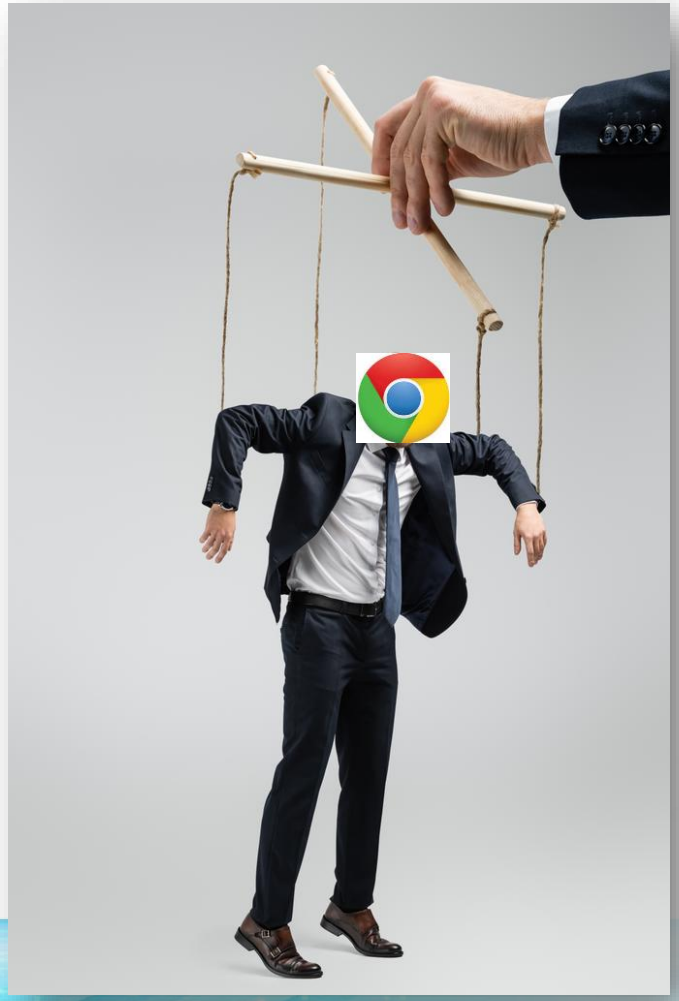
Browser Automation

- Motivation
  - Replicating what the JavaScript does ➔ troublesome



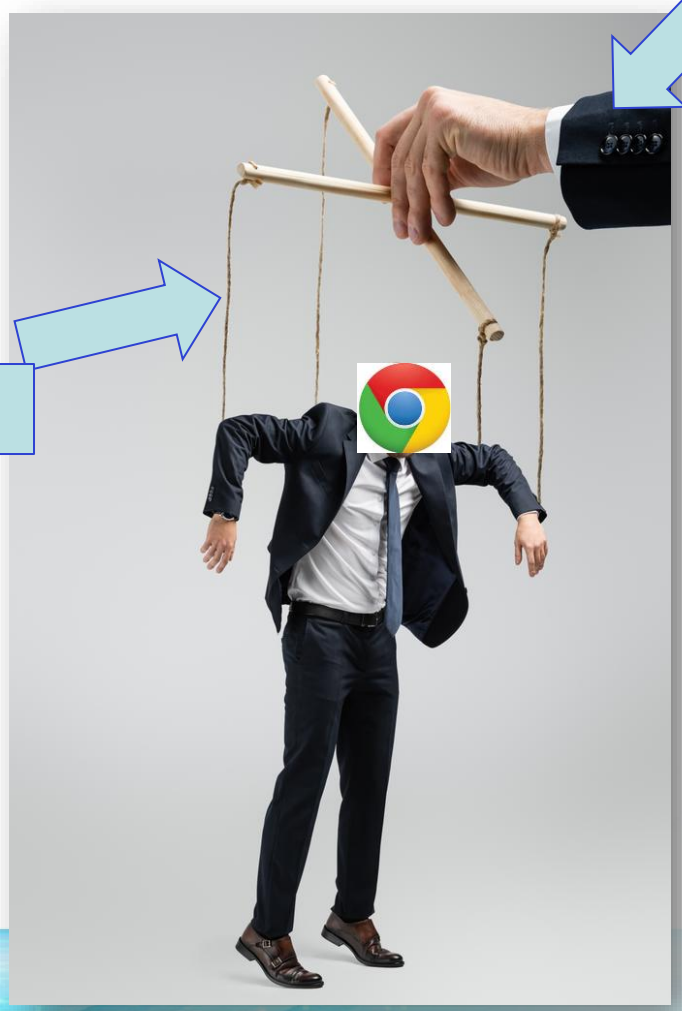
Browser Automation

- Q: If browsers can run JavaScript, can I control the browser with Python?



Browser Automation

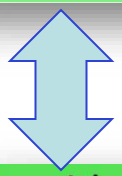
- Yes!
  - Selenium talks to drivers
  - Drivers control browsers



Browser Automation (Chrome)

- Install the latest Chrome driver
  - https://chromedriver.chromium.org/downloads
- **ChromeDriver** version should match **Chrome** version
  - https://googlechromelabs.github.io/chrome-for-testing/#stable

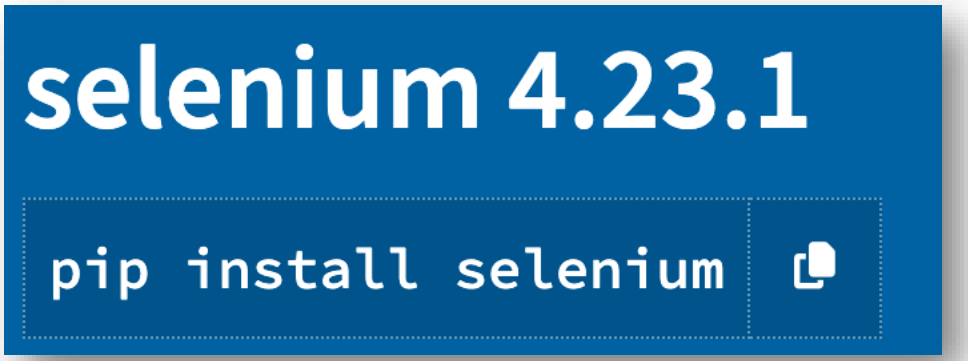
chrome	**win64**	https://storage.googleapis.com/chrome-for-testing-public/127.0.6533.119/win64/chrome-win64.zip



chromedriver	**win64**	https://storage.googleapis.com/chrome-for-testing-public/127.0.6533.119/win64/chromedriver-win64.zip

Browser Automation

- Install Selenium
  - https://pypi.org/project/selenium/



- Import before use

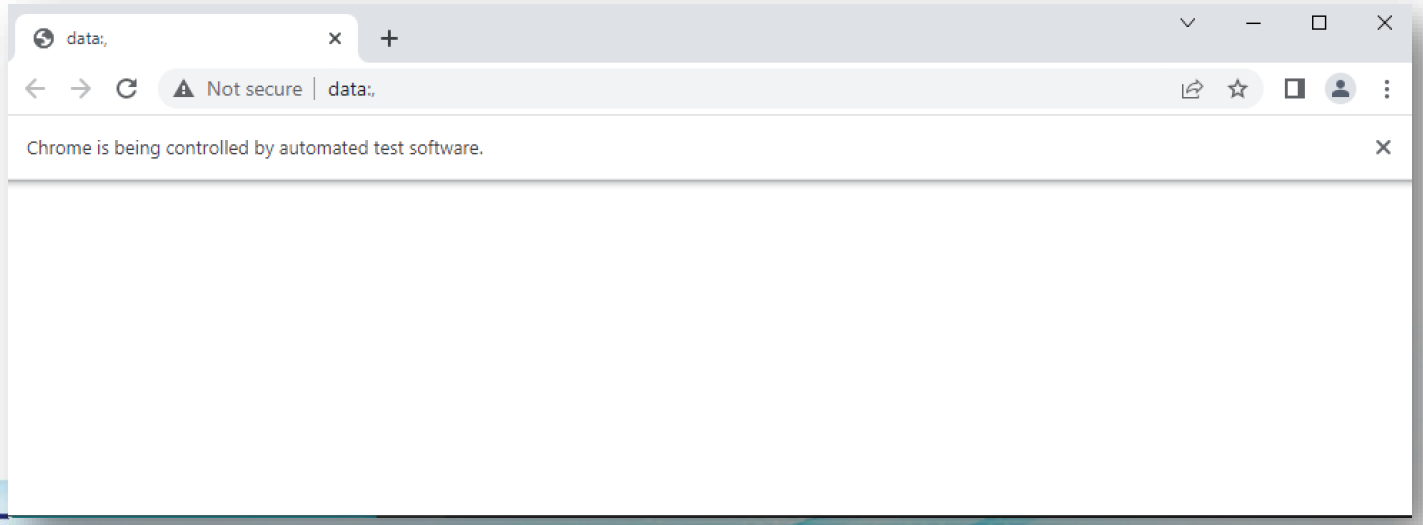
```
from selenium import webdriver
from selenium.webdriver.common.by import By
```

Browser Automation

- Creating the driver object
  - driver = webdriver.Chrome()

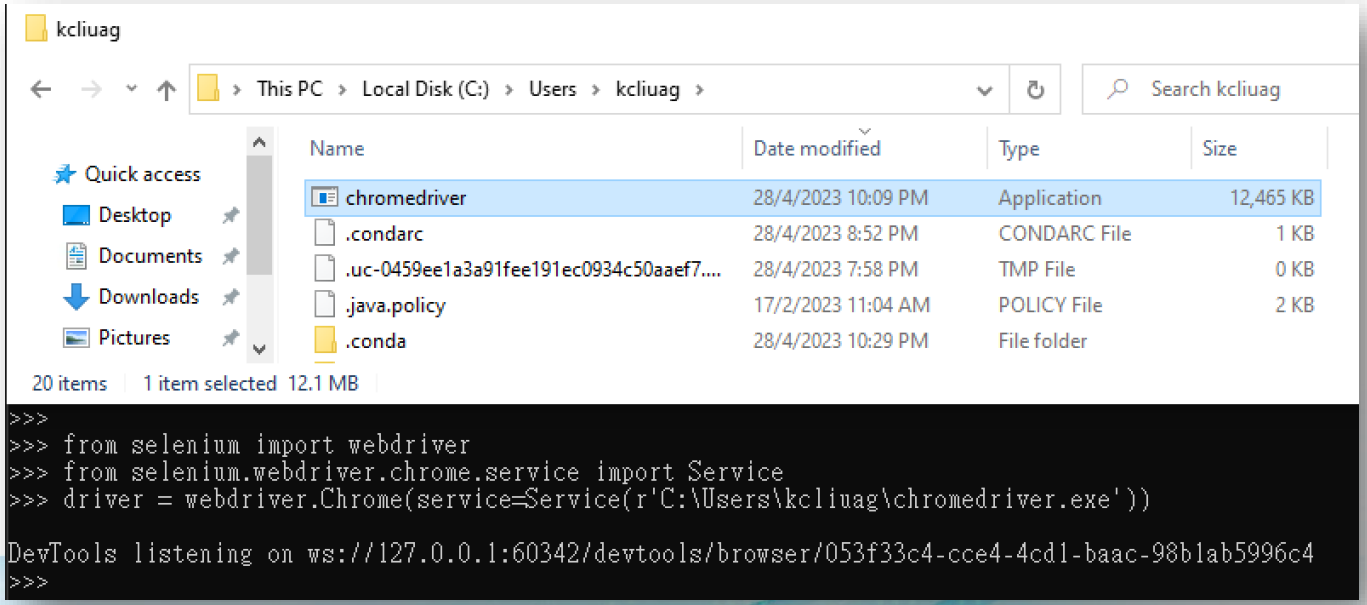
```
# Create the driver
driver = webdriver.Chrome()
```

- This launches a Chrome browser!



Browser Automation

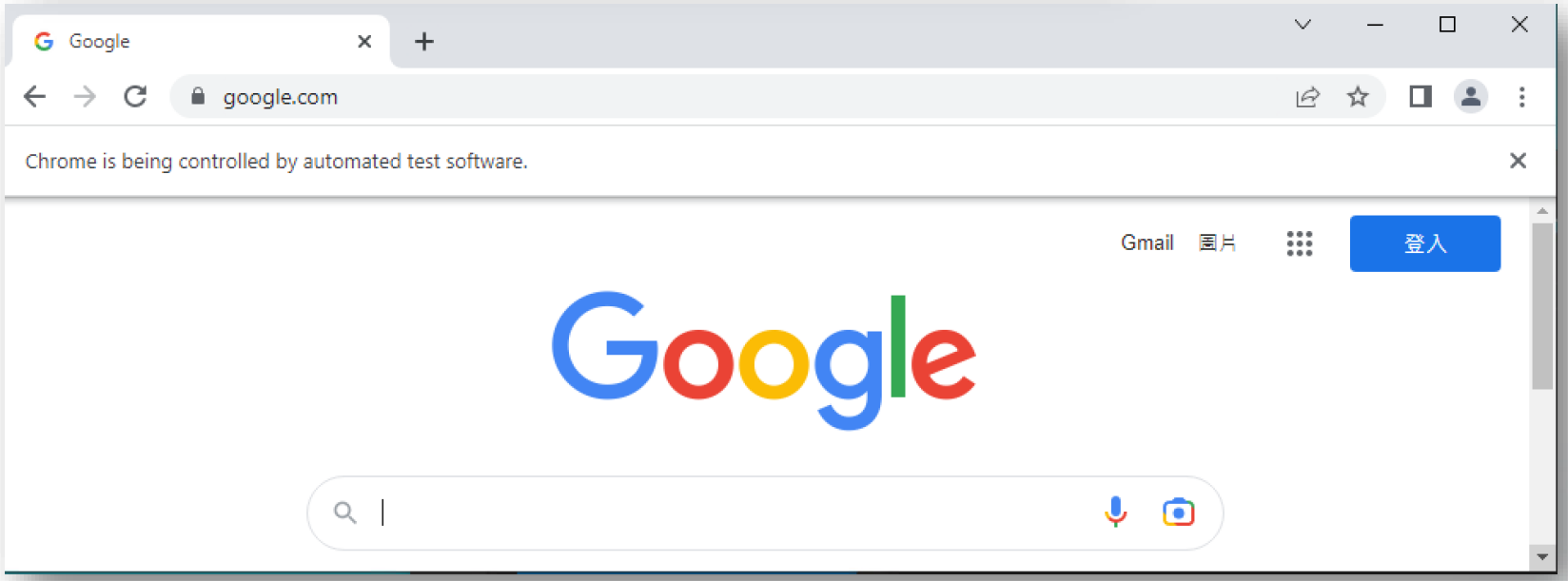
- For Windows
  - If it cannot detect your chrome driver
  - Specify the path of the driver
    - from selenium.webdriver.chrome.service import Service
    - driver = webdriver.Chrome(service=Service(path))



Browser Automation

- Basics - Getting a webpage
  - driver.get(url)

```
driver.get('http://www.google.com')
```



Browser Automation

- Basics - Turning it off
  - driver.close()
    - Close current tab (page), exit if only 1 tab exists
  - driver.quit()
    - Close all tabs and exit browser

Browser Automation

- Basics - Options
  - For automation ➔ no need to see the browser
    - options = webdriver.ChromeOptions()
    - options.add_argument('--headless')
    - driver = webdriver.Chrome(options=options)
  - The browser will silently run in the background

Naruto – Disguise Jutsu

Browser Automation

- Basics - Options
  - Other options

```
# Set options
options = webdriver.ChromeOptions()
options.add_argument('--disable-gpu')
options.add_argument("--window-size=1920, 1200")
options.add_argument('--disable-dev-shm-usage')

# Create the driver
driver = webdriver.Chrome(options=options)
```

1. *List of Chromium Command Line Switches. Peter Beverloo.* https://peter.sh/experiments/chromium-command-line-switches

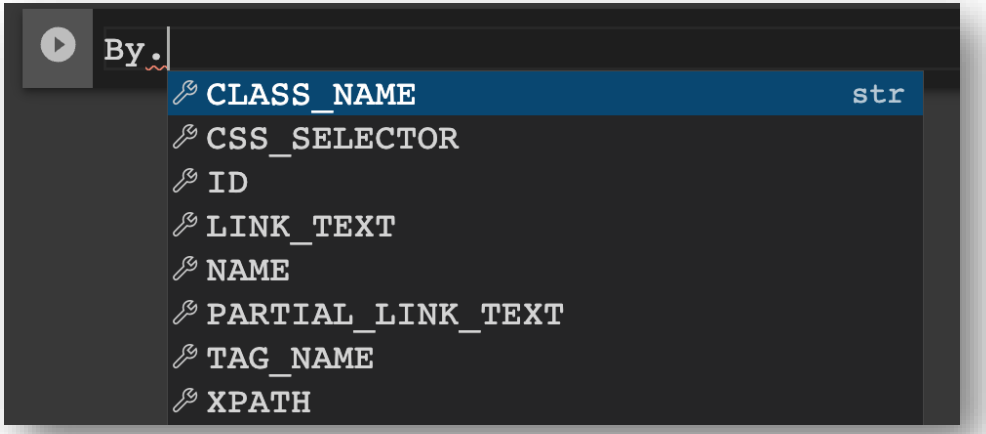
Browser Automation

- Getting the HTML
  - Requests ➔ response.content
  - Selenium ➔ driver.page_source

```
driver.get(voucher_url)

[59] driver.page_source[:100]

'<html lang="en" class=" svgfilters inlinesvg supports cssfilters flexbox flexboxlegacy flexwrap no-f'
```

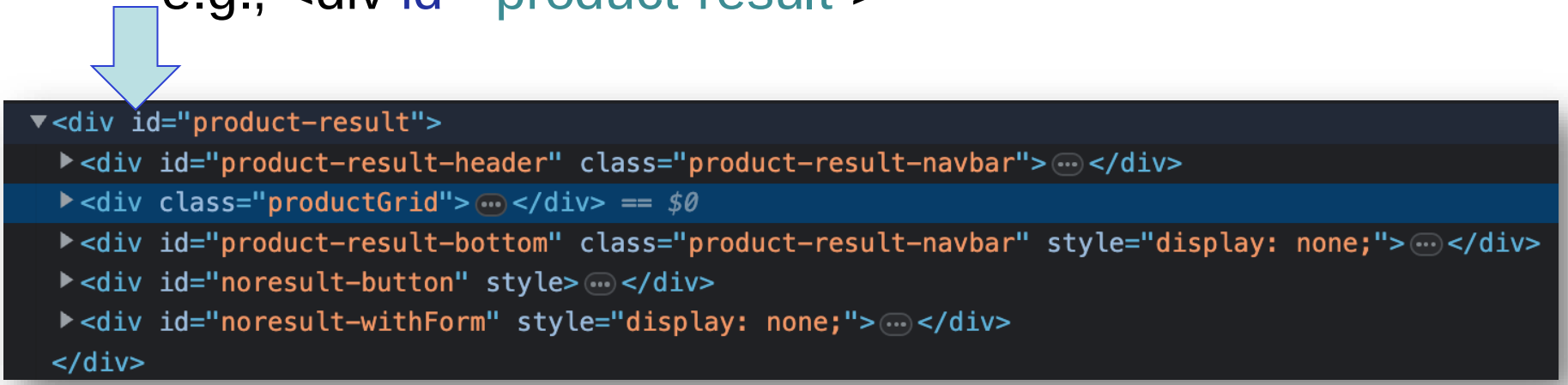
Browser Automation

- Navigation
  - BeautifulSoup ➔ find() and find_all()
  - Selenium ➔ find_element() and find_elements()
  - Must specify “By”



Browser Automation

- Navigation – example
  - e.g.,



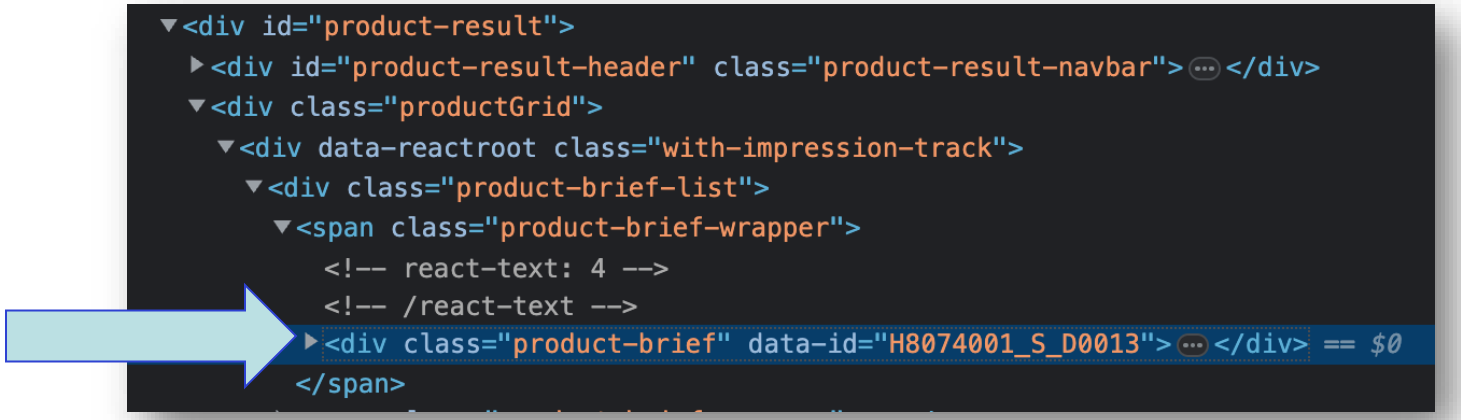
  - driver.find_element(By.ID, value="product-result")

```
product_result_tag = driver.find_element(By.ID, value='product-result')
product_result_tag
```

```
<selenium.webdriver.remote.webelement.WebElement (session="32f8a759bde20942680ad468d9c7b285", element=
```

Browser Automation

- Navigation – More example
  - e.g.,



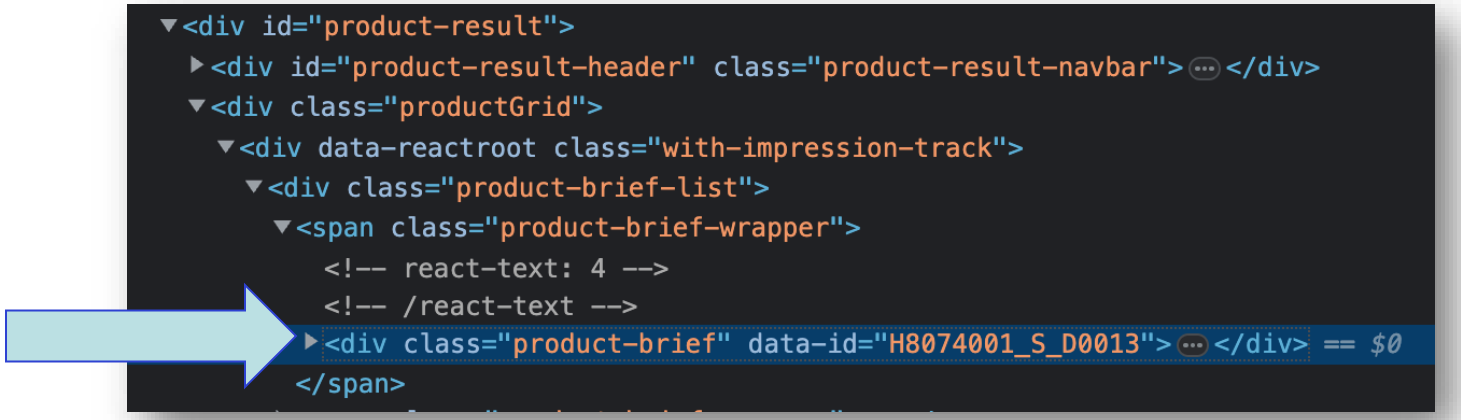
  - element.find_element(By.CLASS_NAME, value="product-brief")

```
product_result_tag.find_element(By.CLASS_NAME, value='product-brief')
```

```
<selenium.webdriver.remote.webelement.WebElement (session="32f8a759bde20942680ad468d9c7b285", element="55a237db-4f4b-42b5-8e9e-658a4adc9f72")>
```

Browser Automation

- WebElement - Attributes
  - element.get_attribute(name)



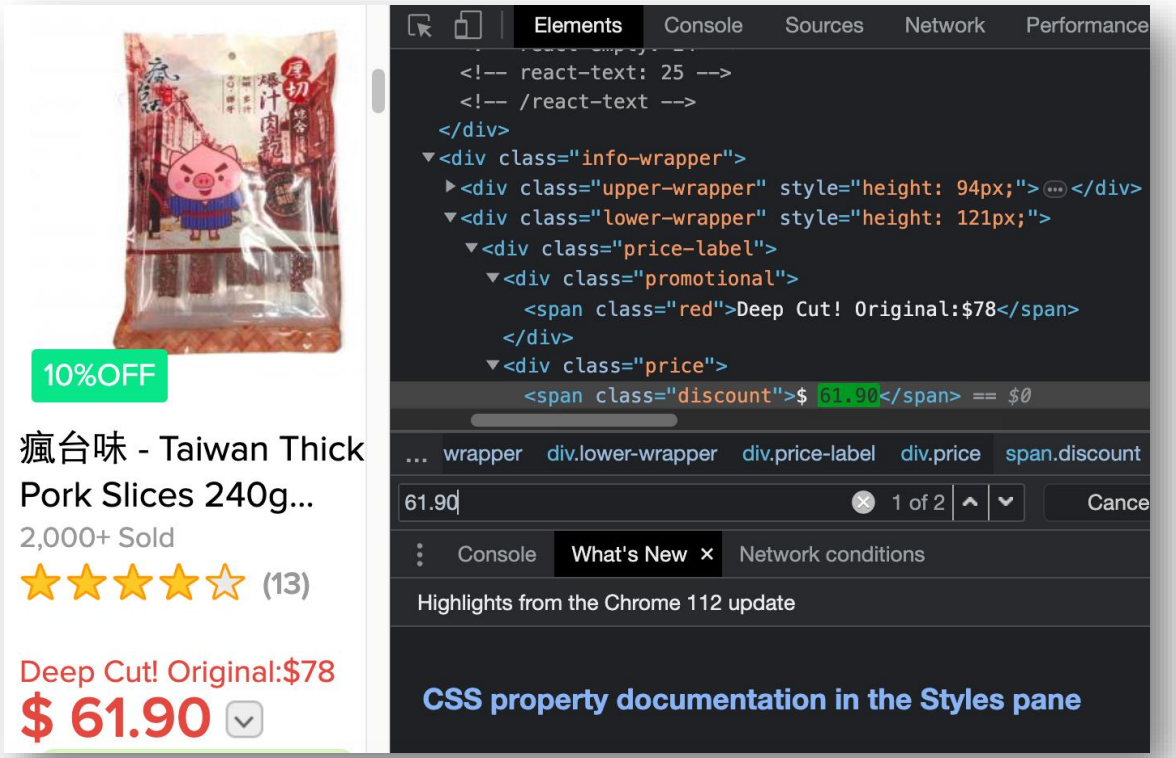
  - Getting data-id attribute
    - element.get_attribute('data-id')

```
brief_element = product_result_tag.find_element(By.CLASS_NAME, value='product-brief')
brief_element.get_attribute('data-id')
```

```
'H8074001_S_D0013'
```

Browser Automation

- WebElement - Text
  - .text
  - Simple, right?




```
e.find_element(By.CLASS_NAME, value='discount').text
```

```
'$ 61.90'
```

Extended Learning - Auto scroll

- Some websites only load more if you scroll down
  - Can simulate that by running a script
  - driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")

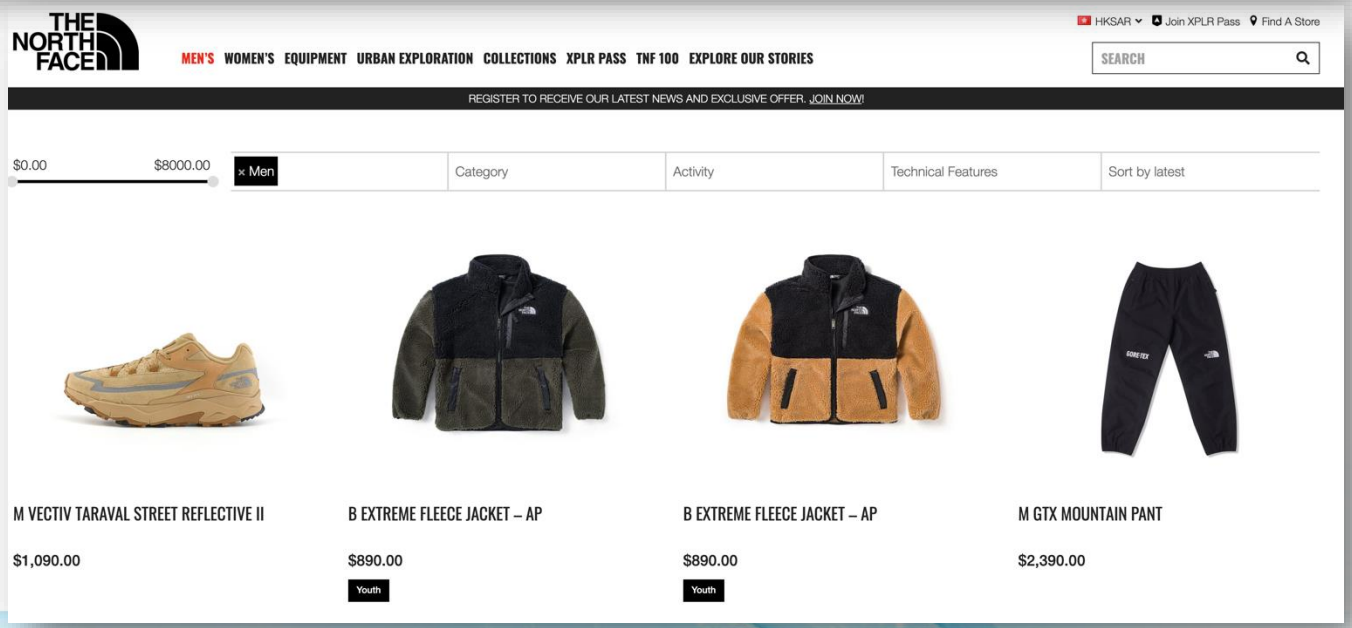
Extended Learning - Auto scroll

- Example - Before

```
nf_url = 'https://www.thenorthface.com.hk/shop/?filters=gender%5Bmen%5D'
driver.get(nf_url)
```

```
[41] len(driver.page_source)
```

```
739503
```



Extended Learning - Auto scroll

- Example - After

```
nf_url = 'https://www.thenorthface.com.hk/shop/?filters=gender%5Bmen%5D'
driver.get(nf_url)
```

```
[41] len(driver.page_source)
```

```
739503
```

```
# Scroll down
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
# Wait for browser to load
time.sleep(3)
# Check length again
len(driver.page_source)
```

```
871579
```

Extended Learning - Auto scroll

- Looping till end of page
  - Use the break statement if there is no height change

```
for i in range(100):
    # Record current page height
    current_height = driver.execute_script("return document.body.scrollHeight")
    # Scroll down
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
    time.sleep(3)
    # Check new page height
    new_height = driver.execute_script("return document.body.scrollHeight")
    # Stop if there is no change
    if new_height == current_height:
        break

# Check length again
len(driver.page_source)
```

```
2343197
```

Browser Automation

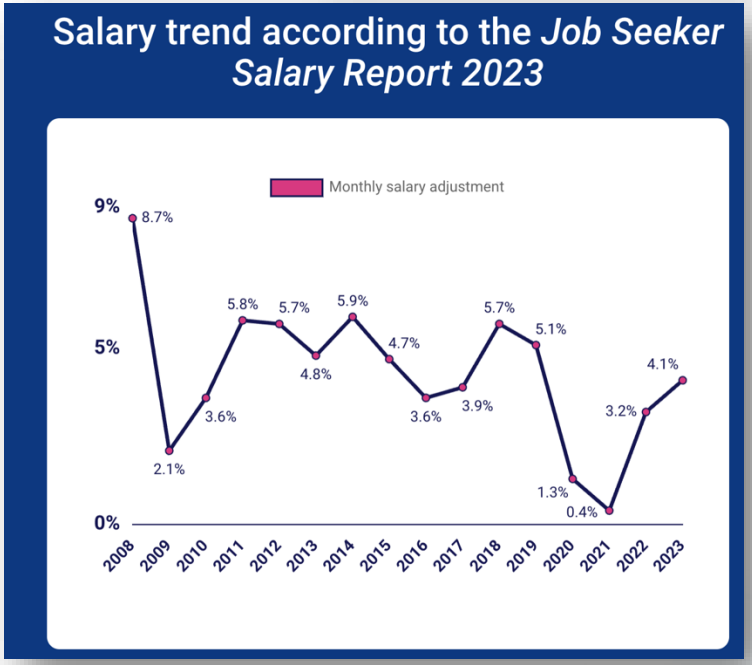
- **Q: Why not just use Selenium?**
  - Performance ➔ Slow / require more resources
  - Driver compatibility ➔ Lose support / require updates

- Example notebook
  - https://colab.research.google.com/drive/1JTNPmRV63eVE1GZve2PmVenBNgGAoBHl?usp=sharing

Basic Visualization

Charting

- Motivation
  - Datasets could be large / boring
  - Charts help to draw insights[1]

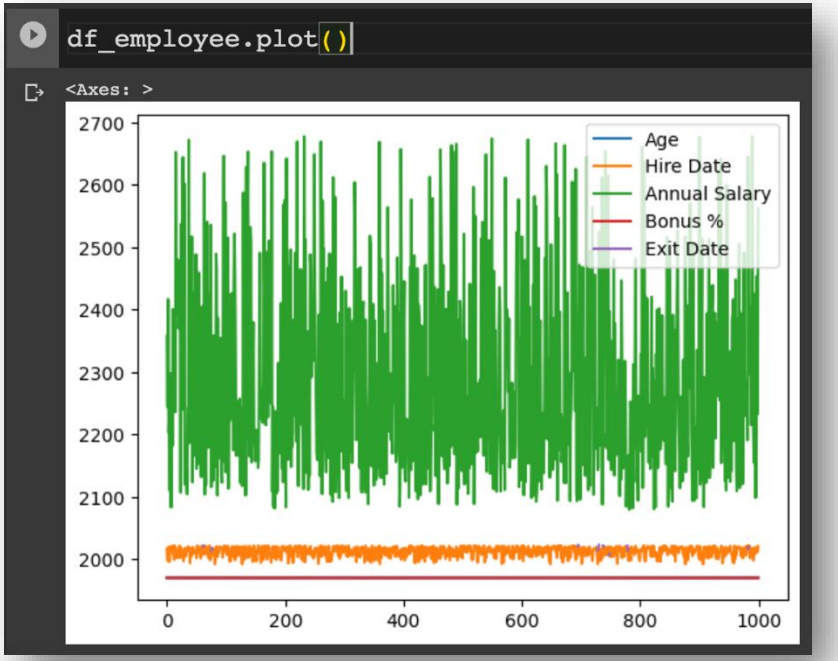


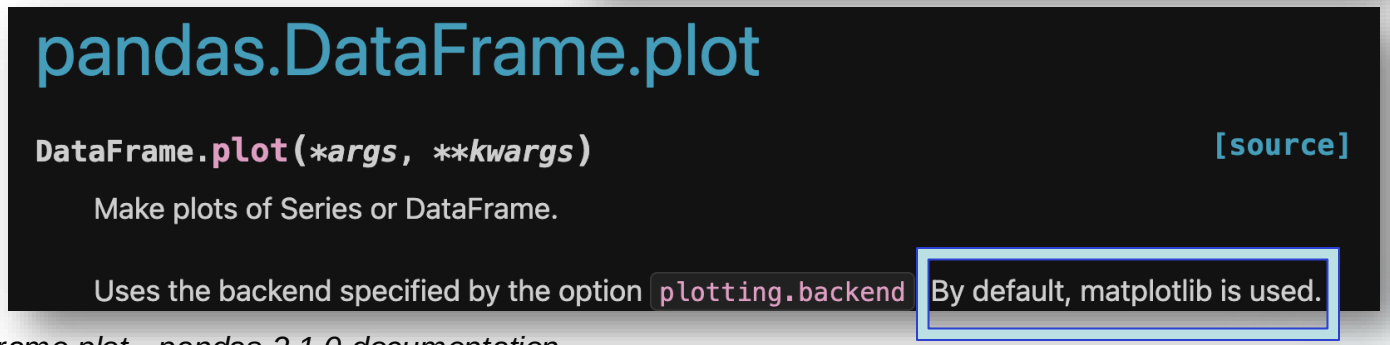
1. *Job seeker salary report 2023* (2023) *JobsDB Hong Kong*. https://hk.jobsdb.com/en-hk/pages/job-seeker-salary-report-2023/

Charting

- From DataFrame
  - Try ➔ **df.plot()**

- "**Matplotlib** is used" [1]
  - What is that?



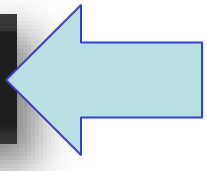


1. *pandas.DataFrame.plot - pandas 2.1.0 documentation*. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.html

Matplotlib

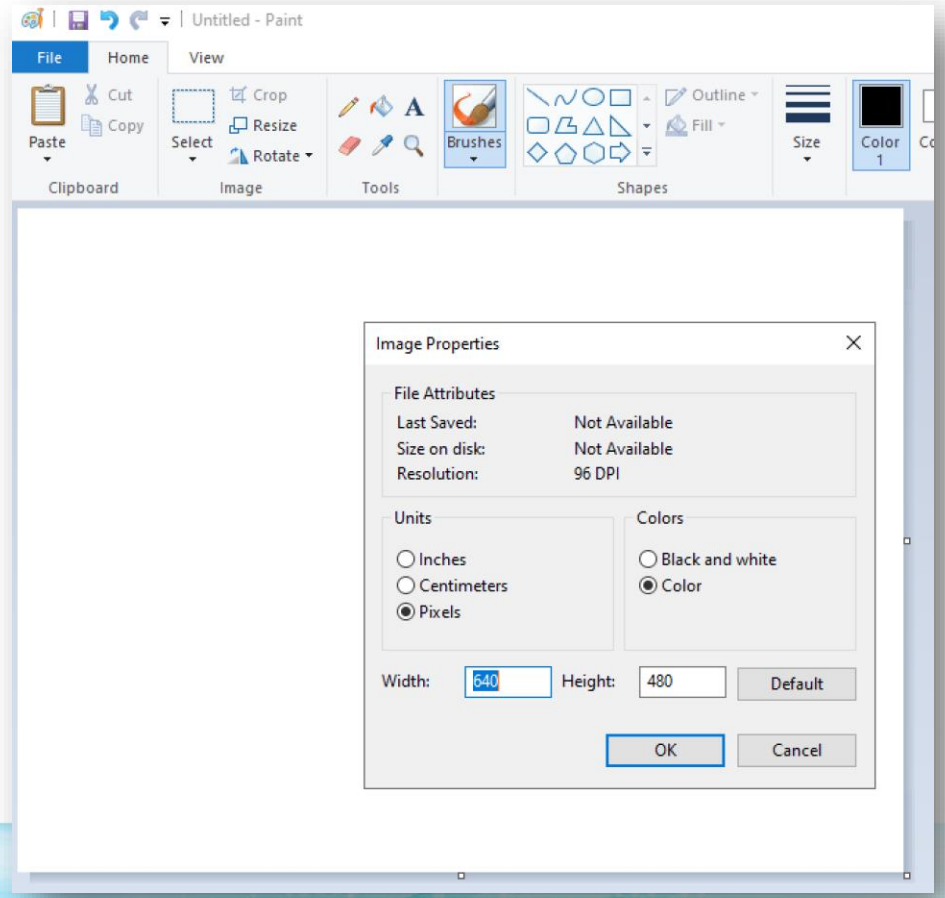
- Import before usage
- It is a package with many indirectly referenced modules
- Main module ➔ **PyPlot**
  - Commonly imported as "**plt**"

```
import matplotlib.pyplot as plt
```



Matplotlib - Figure

- Imagine charting ➔ drawing
- Start with a canvas / drawing board

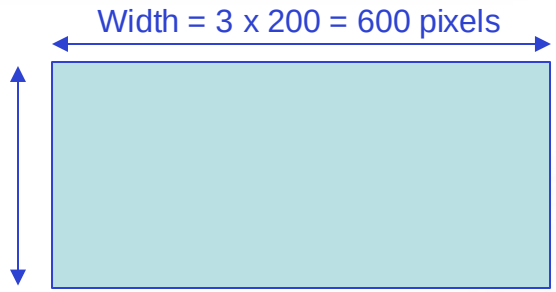


Matplotlib - Figure

- Create ➔ **plt.figure()**
  - figsize: Tuple of width and height (Required)
  - dpi: dot per inch (default: 100)
  - facecolor: Background colour (default: white)

```
# Canvas with 600 x 400 pixels
plt.figure(figsize=(3.0, 2.0), dpi=200, facecolor='white')
```

```
<Figure size 600x400 with 0 Axes>
```

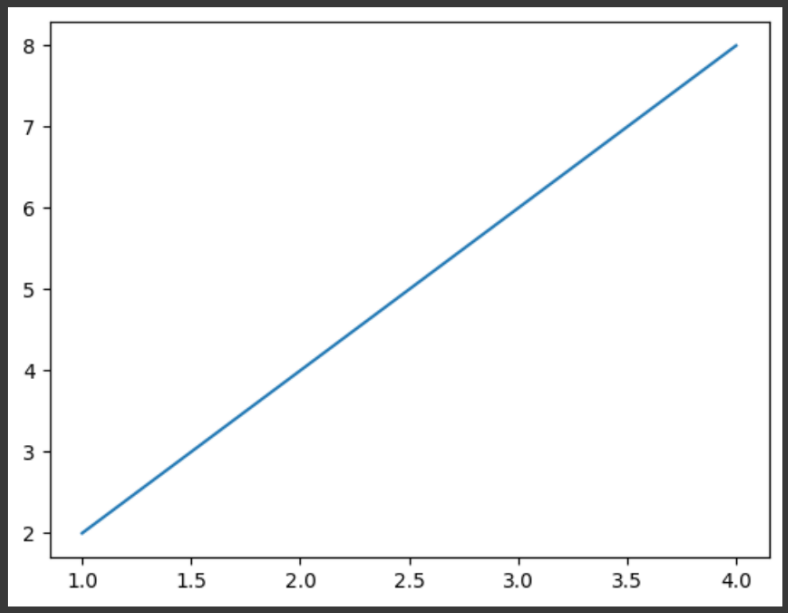


1. *matplotlib.figure - Matplotlib 3.7.3 documentation*. https://matplotlib.org/stable/api/figure_api.html

Matplotlib - Plot

- Drawing a line graph
  - plt.**plot**(x_axis, y_axis)
- In IDEs ➔ needs plt.**show**()

```
# Plotting a line
x_axis = [1, 2, 3, 4]
y_axis = [2, 4, 6, 8]
plt.plot(x_axis, y_axis)

plt.show()
```

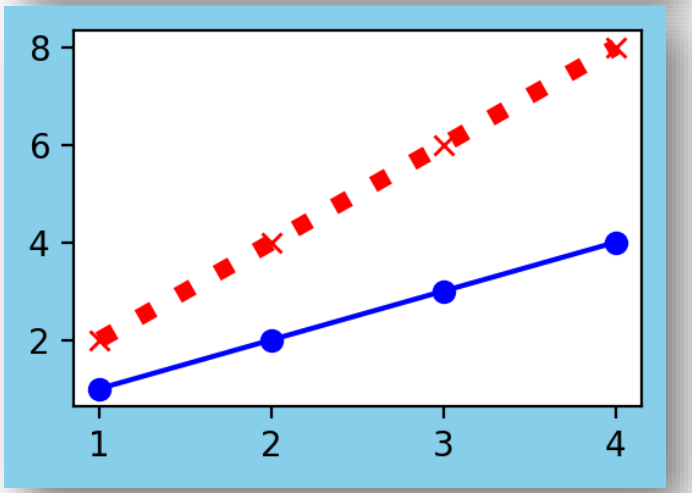


1. *matplotlib.pyplot.plot - Matplotlib 3.8.0 documentation.* https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html

Matplotlib - Plot

- Styling (Format string)
  - Marker
    - x / o / v
  - Line
    - - (solid) / : (dotted)
  - Color
    - g (green)
    - r (red)
    - k (black)

```
fmt = '[marker][line][color]'
```

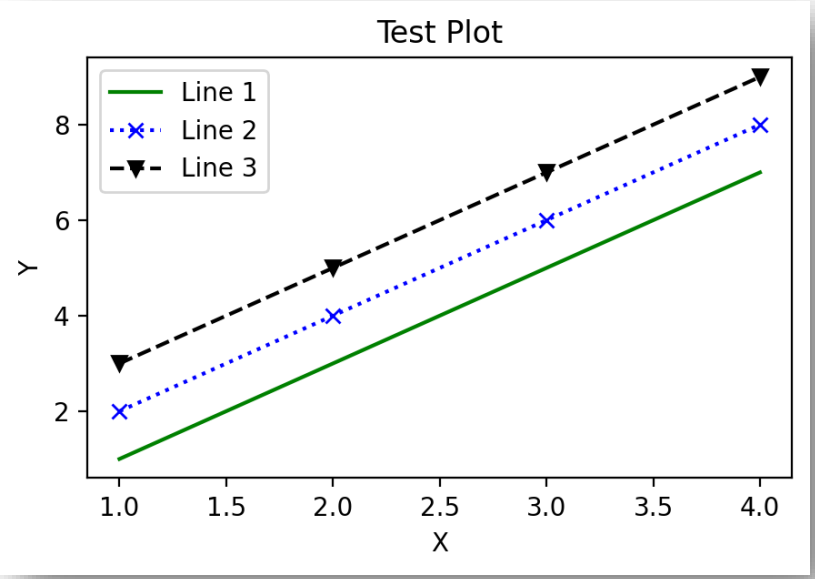



```
# More options
plt.figure(figsize=(3.0, 2.0), dpi=200, facecolor='skyblue')

# : => Dotted (Line)
# x => x (Marker)
# r => Red colour
plt.plot([1, 2, 3, 4], [2, 4, 6, 8], ':xr', linewidth=5)
plt.plot([1, 2, 3, 4], [1, 2, 3, 4], '-ob')

plt.show()
```


1. *matplotlib.pyplot.plot - Matplotlib 3.8.0 documentation.* https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html
2. *List of named colors* – Matplotlib 3.8.0 documentation. https://matplotlib.org/stable/gallery/color/named_colors.html

Matplotlib - Plot

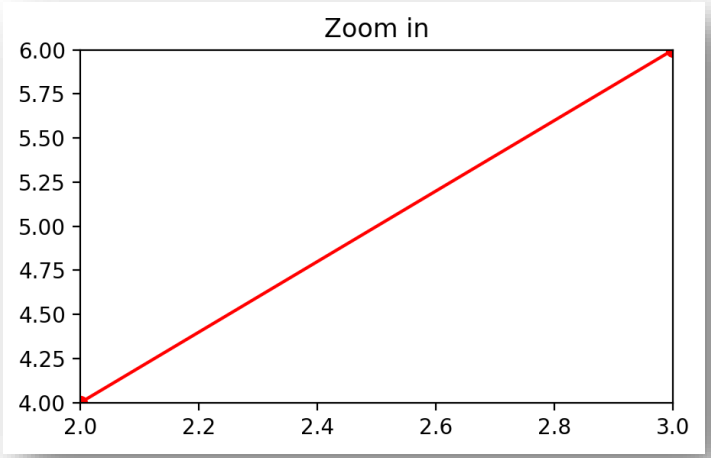
- Labels
  - plot(label='abc')
    - Need plt.legend()
  - plt.title()
  - plt.xlabel()
  - plt.ylabel()




```
# Labels
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

plt.title('Test Plot')
plt.plot([1, 2, 3, 4], [1, 3, 5, 7], '-g',   label='Line 1')
plt.plot([1, 2, 3, 4], [2, 4, 6, 8], ':xb',  label='Line 2')
plt.plot([1, 2, 3, 4], [3, 5, 7, 9], '--vk', label='Line 3')
plt.xlabel('X')
plt.ylabel('Y')
plt.legend(loc='best') # loc: 'upper right' / 'center left' / 'lower center'

plt.show()
```

Matplotlib - Plot

- Zoom
  - plt.xlim((x_min, x_max))
  - plt.ylim((y_min, y_max))

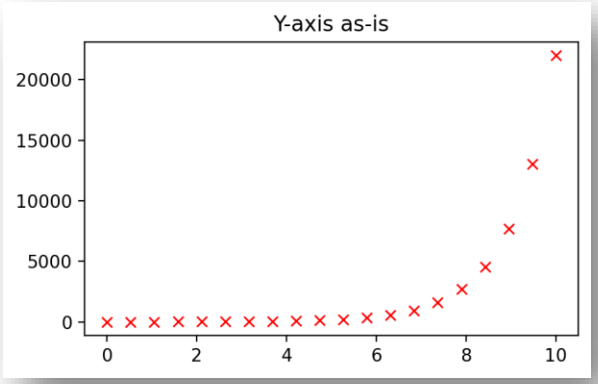



```
# Zoom
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

plt.title('Zoom in')
plt.plot([1, 2, 3, 4], [2, 4, 6, 8], '-or')
plt.xlim((2, 3))
plt.ylim((4, 6))

plt.show()
```

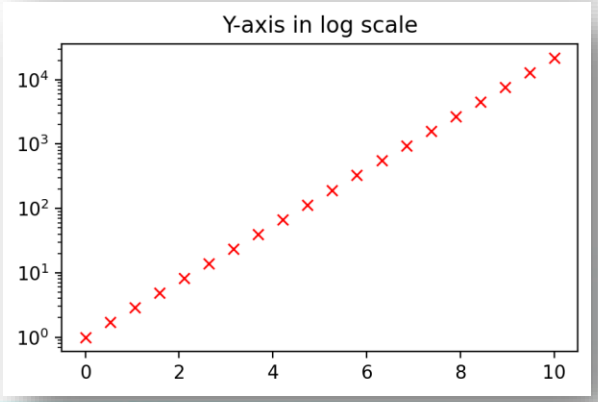
Matplotlib - Plot

- Scale
  - plt.xscale() / plt.yscale() ➔ linear / log / logit

```
import numpy as np
x_axis_linear = np.linspace(0, 10, num=20)
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')
plt.title('Y-axis as-is')
plt.plot(x_axis_linear, np.exp(x_axis_linear), 'xr')
plt.show()
```



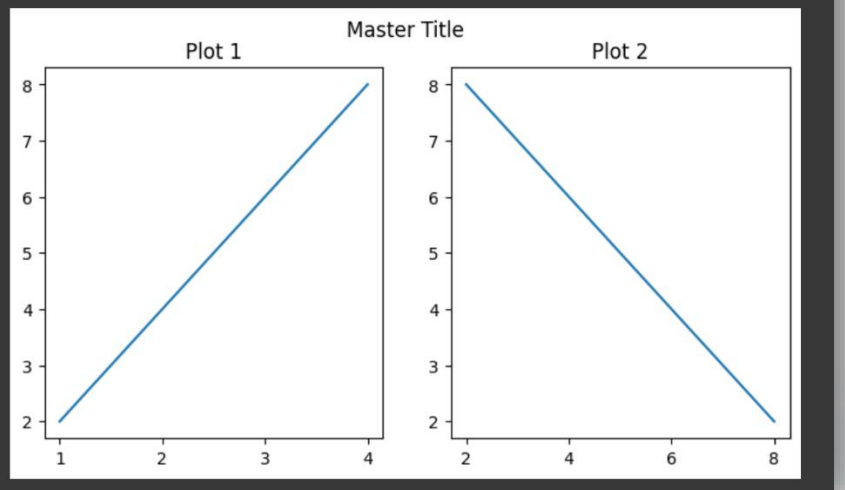

```
# Scaling
x_axis_linear = np.linspace(0, 10, num=20)
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')
plt.title('Y-axis in log scale')
plt.yscale('log')
plt.plot(x_axis_linear, np.exp(x_axis_linear), 'xr')
plt.show()
```



Matplotlib – Multiple Figures

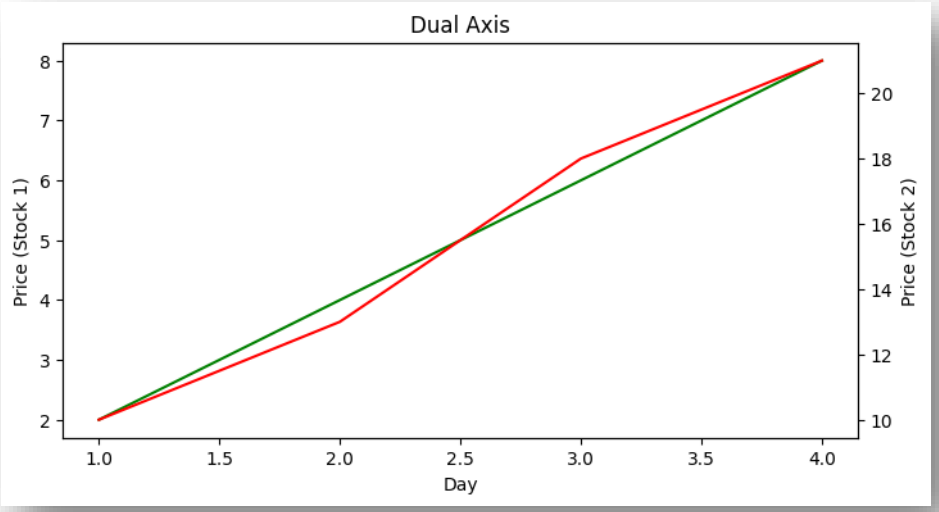
- **plt.subplots**(rows, columns)
  - ➔ fig (figure)
    ➔ ax (Axes ➔ plot)
  - Figure-level title
    - fig.suptitle()
  - Plot-level properties
    - Add “set_”
    - plt.title()
      ➔ ax[i].set_title()

```
fig, ax = plt.subplots(1, 2, figsize=(8.0, 4.0))

# Super title
fig.suptitle('Master Title')

# Plot 1
ax[0].set_title('Plot 1')
ax[0].plot([1, 2, 3, 4], [2, 4, 6, 8])
# Plot 2
ax[1].set_title('Plot 2')
ax[1].plot([2, 4, 6, 8], [8, 6, 4, 2])

fig.show()
```



Matplotlib – Multiple Axes

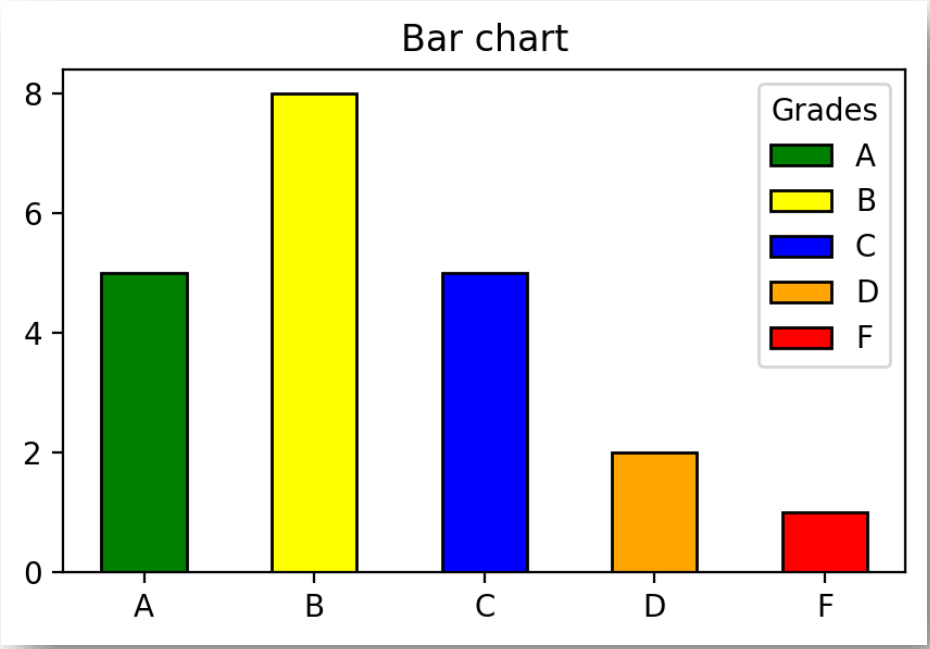
- **ax2 = ax.twinx()**
- As if there is a subplot




```
fig, ax1 = plt.subplots(figsize=(8.0, 4.0))

# Plot 1
ax1.set_title('Dual Axis')
ax1.plot([1, 2, 3, 4], [2, 4, 6, 8], '-g', label='Stock 1')
ax1.set_xlabel('Day')
ax1.set_ylabel('Price (Stock 1)')

# Plot 2
ax2 = ax1.twinx()
ax2.plot([1, 2, 3, 4], [10, 13, 18, 21], '-r', label='Stock 2')
ax2.set_ylabel('Price (Stock 2)')
fig.show()
```

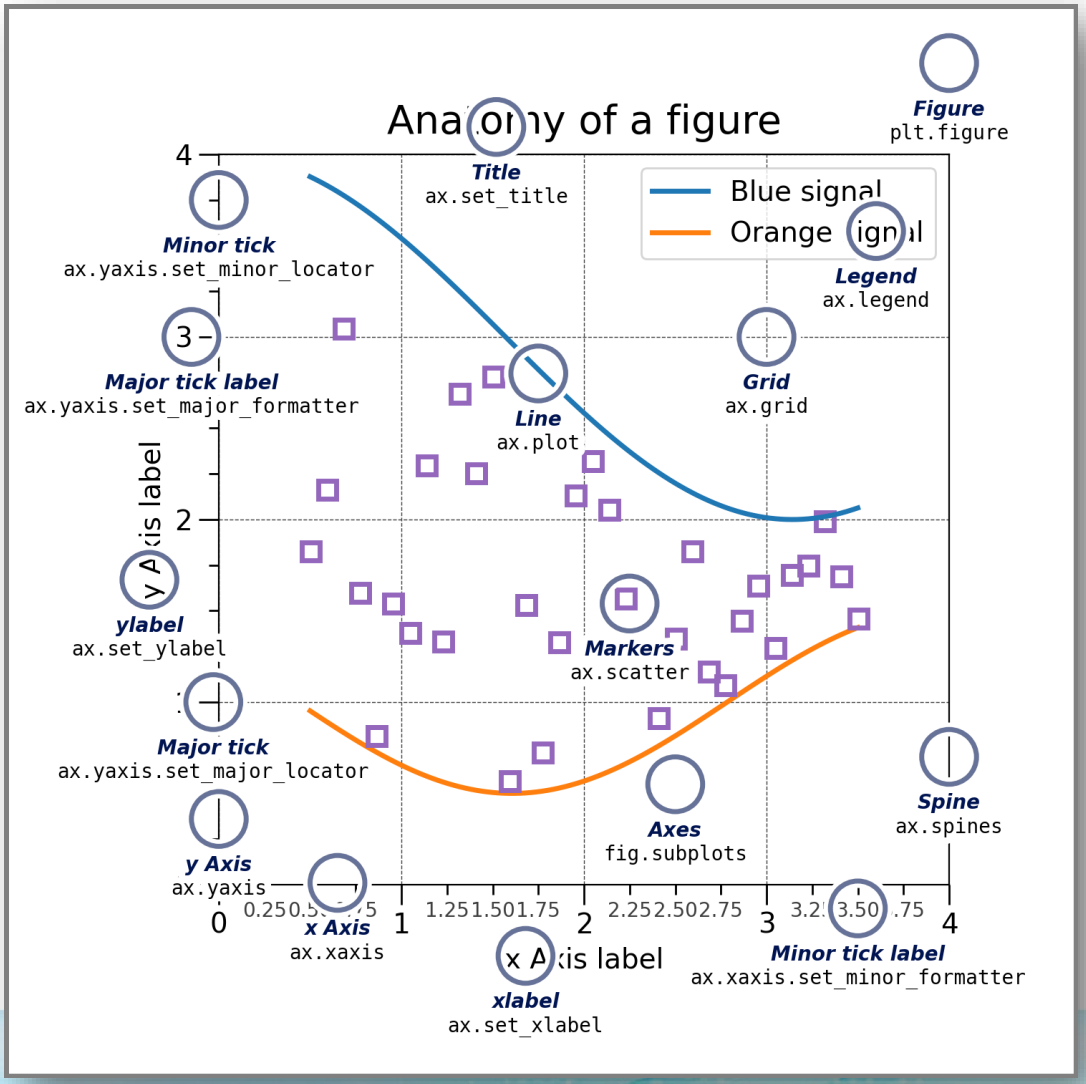
Matplotlib – Bar Chart

- **plt.plot() ➔ plt.bar()**
- Great for category counts



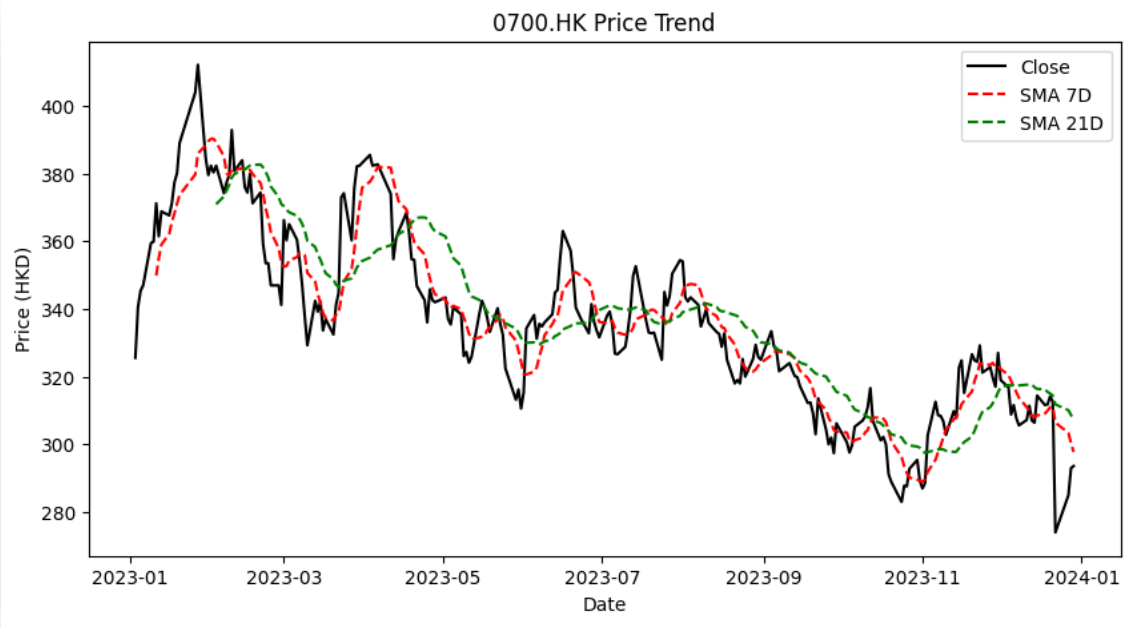

```
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

plt.title('Bar chart')
categories = ['A', 'B', 'C', 'D', 'F']
values = [5, 8, 5, 2, 1]
bar_colors = ['green', 'yellow', 'blue', 'orange', 'red']

plt.bar(categories, values, width=0.5,
        label=categories, color=bar_colors, edgecolor="black")
plt.legend(title='Grades', loc='best')
plt.show()
```

Matplotlib – Summary



Practice 7 – Visualizing Stock Price Trends

- Try out what you have learnt using real stock data!



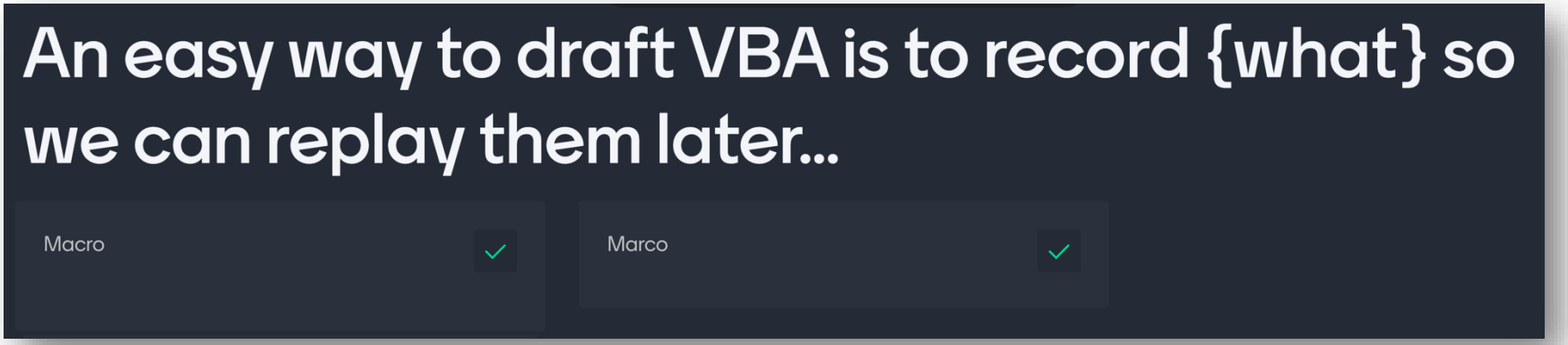
- **Starting notebook**
  https://drive.google.com/file/d/1rBE8g-BcQNIQ_8GZj9P3Z966MmwMuHJt/view?usp=sharing

HKUSPACE
香港大學專業進修學院
HKU School of Professional and Continuing Education

Fuzzy Matching

Fuzzy Match

- Motivation
  - Real-world inputs may be different from expected



Fuzzy Match

- Exact matching
  - May not be best

```
'RECORD MACRO' == 'RECORD MARCO'
```

```
False
```

Fuzzy Match

- Edit Distance (Going from X to Y)
  - What is the unit of difference?
    - In real life ➔ inch / meter
    - String comparison ➔ ???

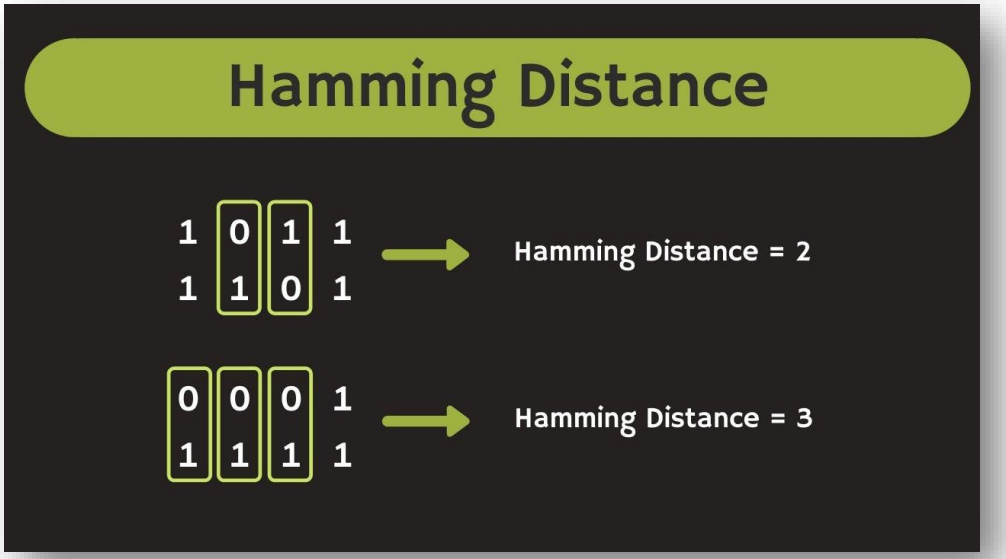
Fuzzy Match

- Clue from childhood
  - Spot the difference ➔ Number of replacement?

Fuzzy Match

- Minimum number of replacement
  - C ➔ R, R ➔ C
  - Distance = 2
  - This is called Hamming distance

```
X = 'RECORD MACRO'
Y = 'RECORD MARCO'
```



Fuzzy Match

- Minimum number of replacement
  - Q: How to measure it in Python?

```
X = 'RECORD MACRO'
Y = 'RECORD MARCO'
```

Fuzzy Match

- Hamming distance
  - Option 1

```
X = 'RECORD MACRO'
Y = 'RECORD MARCO'
```

```
X = 'RECORD MACRO'
Y = 'RECORD MARCO'

replacement_count = 0
for i in range(len(X)):
  if X[i] != Y[i]:
    replacement_count = replacement_count + 1

print(f'{replacement_count} letters need to be replaced')
```

```
2 letters need to be replaced
```

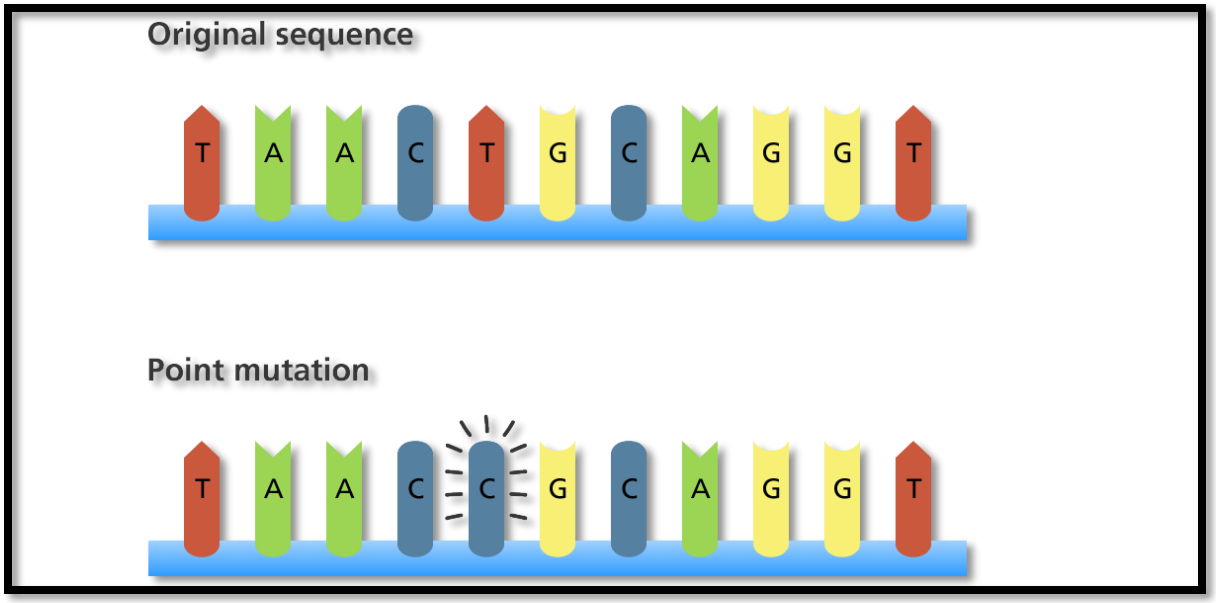
  - Option 2

```
sum([1 for i in range(len(X)) if X[i]!=Y[i]])
```

```
2
```

Fuzzy Match

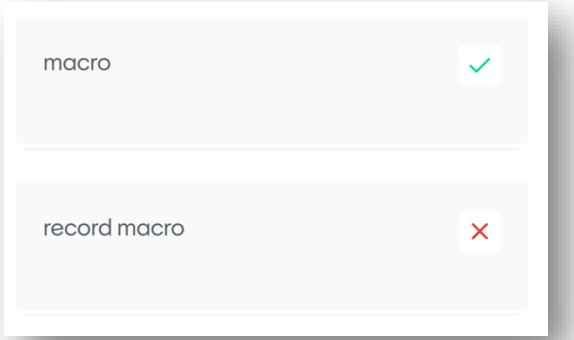
- Hamming distance
  - Can be used in error detection
    - Computer memory / Digital signals / DNA / etc



1. Mohammadi-Kambs, M., Hölz, K., Somoza, M. M., & Ott, A. (2017, April 5). *Hamming Distance as a Concept in DNA Molecular Recognition*. PubMed Central (PMC). https://doi.org/10.1021/acsomega.7b00053

Fuzzy Match

- Hamming distance
  - What if length is different?




```
X = 'RECORD MACRO'
Y = 'MACRO'
sum([1 for i in range(len(X)) if X[i]!=Y[i]])
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-12-edb8fe2cb8f1> in <cell line: 3>()
      1 X = 'RECORD MACRO'
      2 Y = 'MACRO'
----> 3 sum([1 for i in range(len(X)) if X[i]!=Y[i]])

<ipython-input-12-edb8fe2cb8f1> in <listcomp>(.0)
      1 X = 'RECORD MACRO'
      2 Y = 'MACRO'
----> 3 sum([1 for i in range(len(X)) if X[i]!=Y[i]])

IndexError: string index out of range
```

Fuzzy Match

- Hamming distance
  - New considerations ➔ Addition & Deletion

Fuzzy Match

- Levenshtein distance
  - New considerations ➔ Addition & Deletion
    - From X: Delete “RECORD ”
    - From Y: Add “RECORD ”

```
X = 'RECORD MACRO'
Y = 'MACRO'
```

  - Each add/delete ➔ +1 distance

Fuzzy Match

- Q: How to find the Levenshtein distance?
  - X = Sunday
  - Y = Saturday
- **Replace / Add / Delete**

S	u	n	d	a	y		
S	a	t	u	r	d	a	y

Fuzzy Match

- Levenshtein Algorithm
  - Initialize from 0 to maximum length

			Y							
			S	a	t	u	r	d	a	y
		0	1	2	3	4	5	6	7	8
X	S	1	0	1	2	3	4	5	6	7
	u	2	1	1	2	2	3	4	5	6
	n	3	2	2	2	3	3	5	5	6
	d	4	3	3	3	3	4	3	4	5
	a	5	4	3	4	4	4	4	3	4
	y	6	5	4	4	5	5	5	4	3

Fuzzy Match

- Levenshtein Algorithm
  - X=S vs Y=S ➔ distance is 0

		Y							
		S	**a**	**t**	**u**	**r**	**d**	**a**	**y**
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	5	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

X

Fuzzy Match

- Levenshtein Algorithm
  - X=S vs Y=SA ➔ why distance is 1?

			Y							
			S	a	t	u	r	d	a	y
		0	1	2	3	4	5	6	7	8
X	S	1	0	1	2	3	4	5	6	7
	u	2	1	1	2	2	3	4	5	6
	n	3	2	2	2	3	3	5	5	6
	d	4	3	3	3	3	4	3	4	5
	a	5	4	3	4	4	4	4	3	4
	y	6	5	4	4	5	5	5	4	3

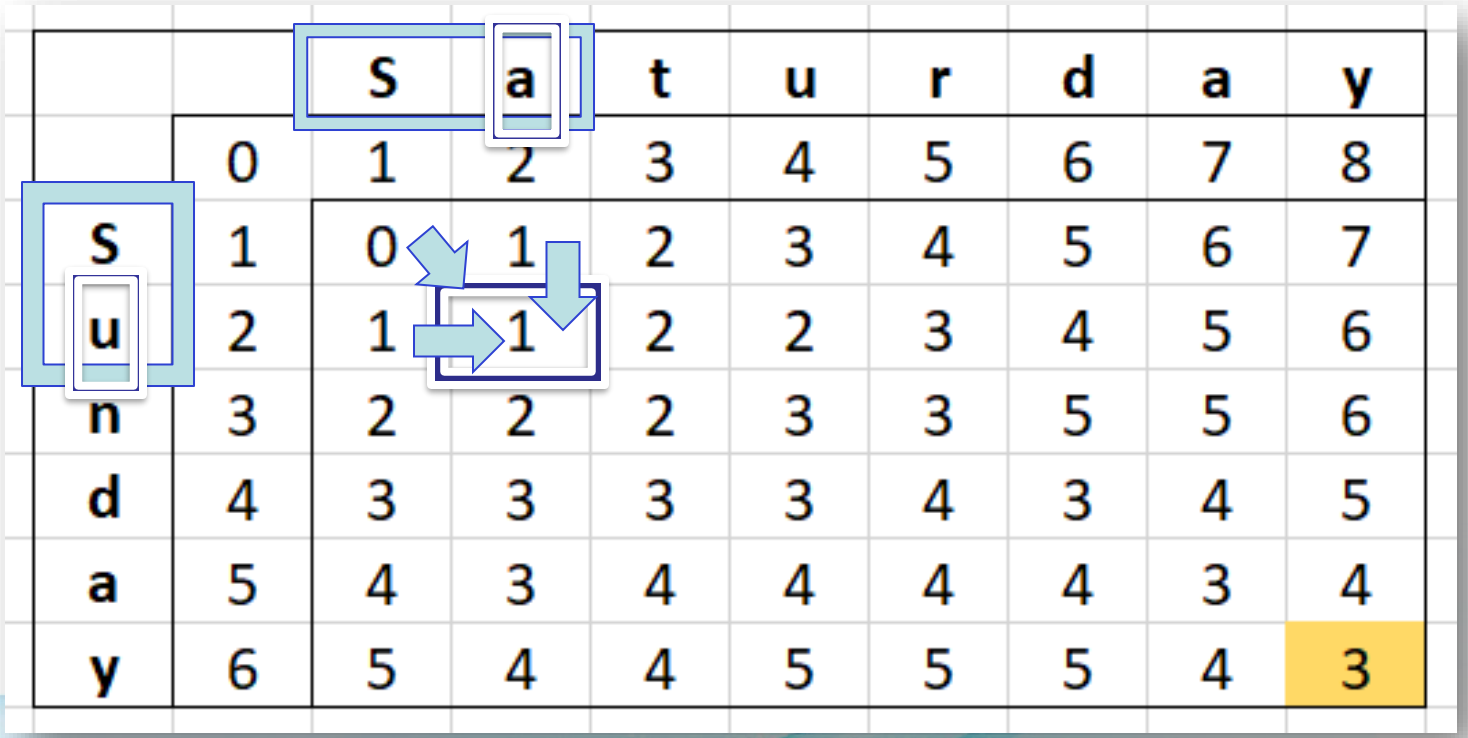
Fuzzy Match

- Levenshtein Algorithm
  - Each step: minimum of 3 directions + add 1 if mismatch

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	5	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Fuzzy Match

- Levenshtein Algorithm – More example
  - min(1, 0, 1) = 0
  - Add 1 because u is not a
  - 0 + 1 ➔ 1



		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	5	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Fuzzy Match

- Levenshtein Algorithm – More example
  - min(4, 3, 5) ➔ 3
  - No need to add 1 because d vs d
  - Answer: 3

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	5	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Fuzzy Match

- Levenshtein Algorithm
  - min(4, 3, 4) ➔ 3
  - No need to add 1 because y vs y
  - Answer: 3

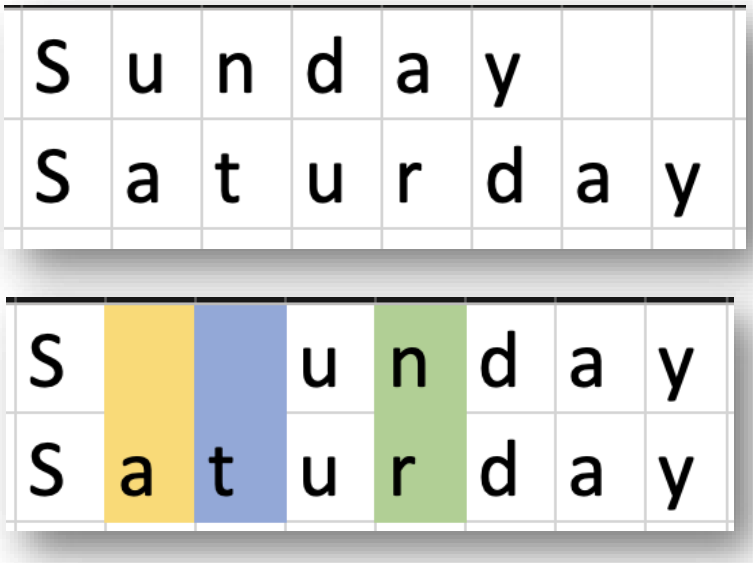
S	u	n	d	a	y		
S	a	t	u	r	d	a	y

S			u	n	d	a	y
S	a	t	u	r	d	a	y

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7
u	2	1	1	2	2	3	4	5	6
n	3	2	2	2	3	3	5	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

Fuzzy Match

- Levenshtein Algorithm
  - Q: How to write it in Python?



Fuzzy Match

- Levenshtein Algorithm
  - Interview question (not for this course ☺)
  - Explainer from NeetCode if you are interested: https://youtu.be/XYi2-LPrwm4

```python
class Solution:
    def minDistance(self, word1: str, word2: str) -> int:
        cache = [[float("inf")] * (len(word2) + 1) for i in range(len(word1) + 1)]

        for j in range(len(word2) + 1):
            cache[len(word1)][j] = len(word2) - j
        for i in range(len(word1) + 1):
            cache[i][len(word2)] = len(word1) - i

        for i in range(len(word1) - 1, -1, -1):
            for j in range(len(word2) - 1, -1, -1):
                if word1[i] == word2[j]:
                    cache[i][j] = cache[i + 1][j + 1]
                else:
                    cache[i][j] = 1 + min(cache[i + 1][j], cache[i][j + 1], cache[i+1][j+1])

        return cache[0][0]
```

Fuzzy Match

- Levenshtein Algorithm
  - https://pypi.org/project/python-Levenshtein/
  - pip install python-Levenshtein

  - Import Levenshtein

```
import Levenshtein
```

Fuzzy Match

- Levenshtein Algorithm
  - Levenshtein.distance(X, Y)
  - Result is symmetric

```
X = 'SUNDAY'
Y = 'SATURDAY'
Levenshtein.distance(X, Y)
```

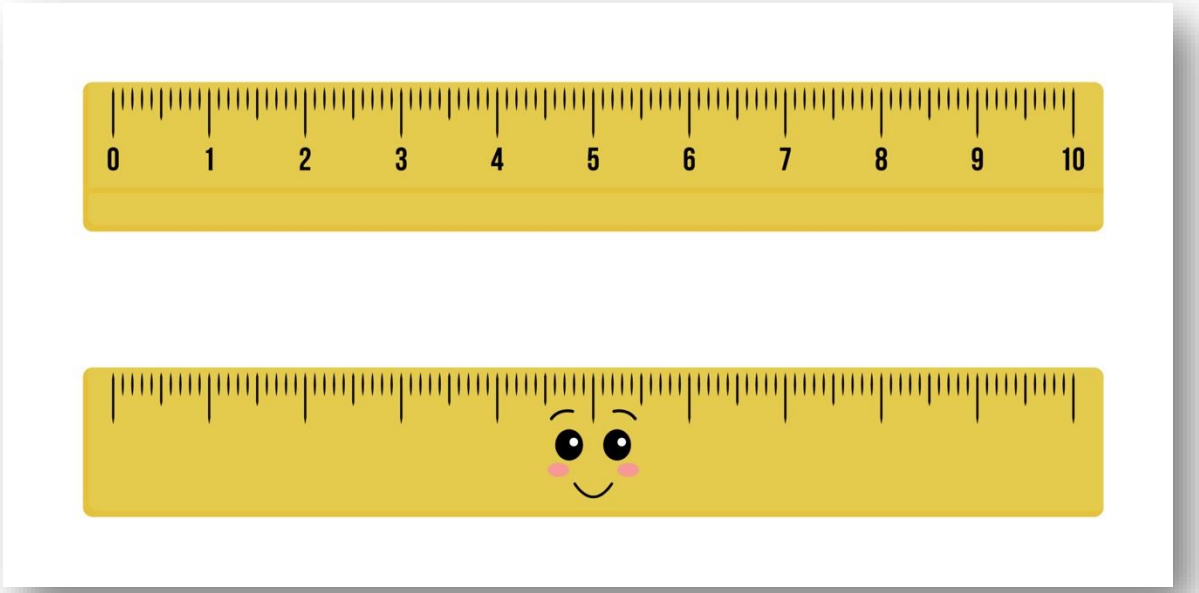
```
3
```

```
Levenshtein.distance(Y, X)
```

```
3
```

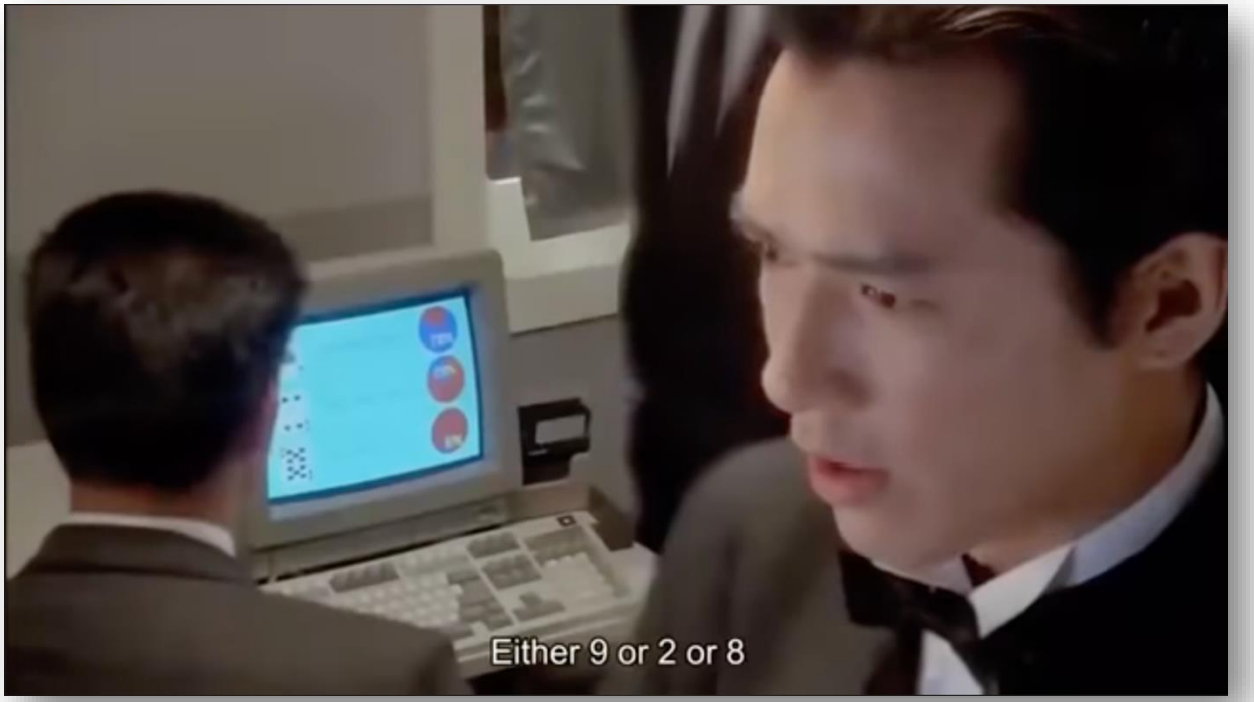
Fuzzy Match

- **Q: Wait, what were we trying to do?**
  - Measure similarity of 2 string



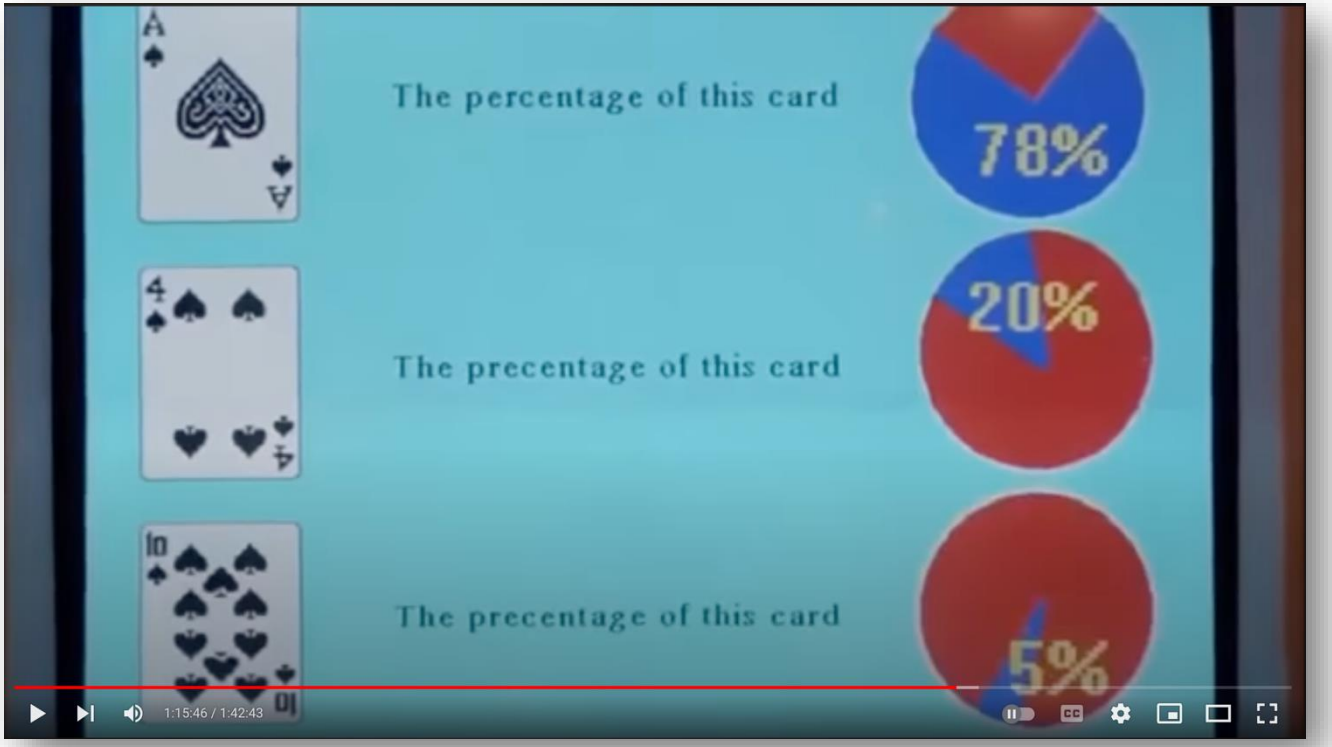
Fuzzy Match

- Clues from God of Gamblers 2 (1990)
  - https://youtu.be/Nl8PkOiFfBw?si=cujCSkENbXfvITfC&t=235



Fuzzy Match

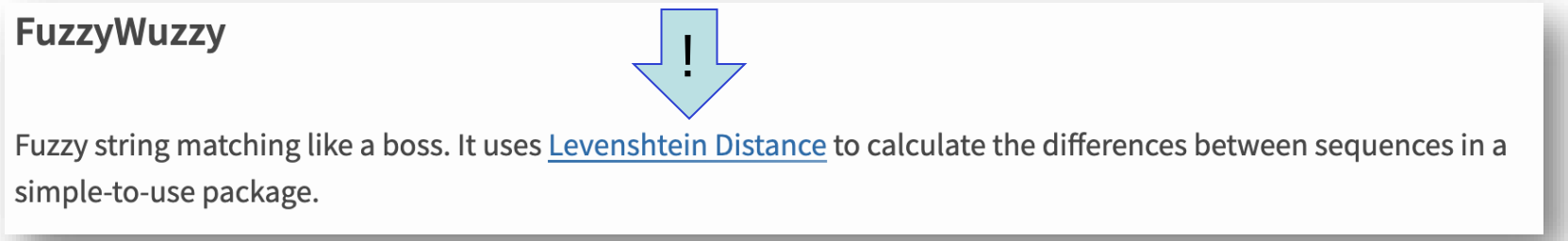
- Distance ➔ Similarity scores
  - Convert the distance into a number from 0 to 100%



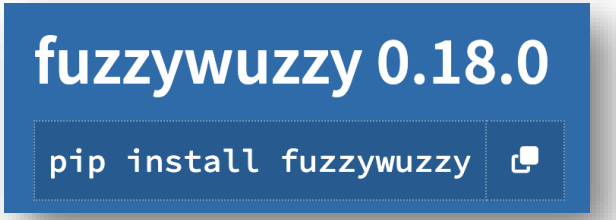
^ sum of possibilities may not be 100%

Fuzzy Match

- FuzzyWuzzy
  - https://pypi.org/project/fuzzywuzzy/



  - pip install fuzzywuzzy



  - Import before use

```
from fuzzywuzzy import fuzz, process
```

Fuzzy Match

- process.extract() ➔ Use partial ratio by default
  - Useful ➔ matching substring (e.g., start/end)

```
process.extract('hku', choices=['cuhk', 'hkust', 'hkuspace'])
```

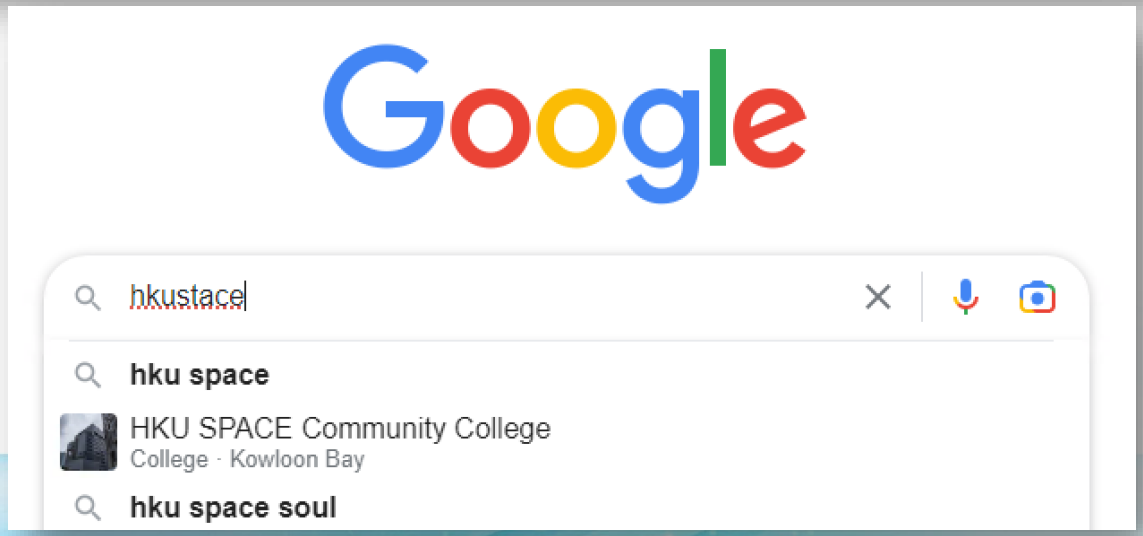
```
[('hkust', 90), ('hkuspace', 90), ('cuhk', 57)]
```

Fuzzy Match

- Similarity ratio
  - Length factor: hku ➔ hkust > hkuspace
  - Block factor (continuous): hku ➔ hkust > cuhk
  - Useful ➔ Comparing words of similar length

```
# Similarity Ratio
process.extract('hku', choices=['cuhk', 'hkust', 'hkuspace'], scorer=fuzz.ratio)
```

```
[('hkust', 75), ('cuhk', 57), ('hkuspace', 55)]
```



Fuzzy Match

- Partial token sort ratio
  - Sort into a word list to compare
  - Useful ➔ compare names (first/last name order)

```
process.extract('Hello Hong Kong',
                choices=['Kong Hello Hong', 'Hello King Kong'],
                scorer=fuzz.partial_token_sort_ratio)
```

```
[('Kong Hello Hong', 100), ('Hello King Kong', 87)]
```

Fuzzy Match

- Partial token set ratio
  - Considers unique tokens with loose comparison
  - Useful ➔ Check against plagiarism

```
process.extract('Ding Ding',
                choices=['Ding', 'Dong'],
                scorer=fuzz.partial_token_set_ratio)
```

```
[('Ding', 100), ('Dong', 75)]
```

```
process.extract('Nanae Aoyama',
                choices=['Aoyama Nanae', 'Aoyama Nanako'],
                scorer=fuzz.partial_token_set_ratio)
```

```
[('Aoyama Nanae', 100), ('Aoyama Nanako', 100)]
```

```
[35] process.extract('Nanae Aoyama',
                choices=['Aoyama Nanae', 'Aoyama Nanako'],
                scorer=fuzz.partial_token_sort_ratio)
```

```
[('Aoyama Nanae', 100), ('Aoyama Nanako', 92)]
```

Fuzzy Match

- More practical application
  - Clean up data (e.g., Survey response)
    1. Input ➔ Hong Kong
    2. Expected ➔ Hong Kong SAR
    3. New Column ➔ Fuzzy match

Input	Fuzzy Matched
Hong Kong	Hong Kong SAR

FirstName	Surname	CompanyName	Address1	Town
Lee	Leighton	Alter Image	131 Sundown Close	
L	Leighton	Alter Image Ltd	131 Sundown Rd	Leicester
Emma	Wrighte	Write Way	280 Bath road	Birmingham
E	Wright	The Write Way	280 Bath rd	
Emma	Wright	The Write Way Ltd	280 Bath road	Birmingham
William	Giles	Social & Military Club	78 Ford St	Bolton
Bill	Giles	S & M Club	78, Ford Street	Bolton
Alan	Baker	Bakery Baker Ltd	7 main road	Lowton
Al	Baker	Bakery Baker	7 main Rd	Lowton

HKUSPACE
香港大學專業進修學院
HKU School of Professional and Continuing Education

THANK YOU