環境安裝：Python程式語言

請大家光明正大的拿出手機 / 電腦

開始之前

打開

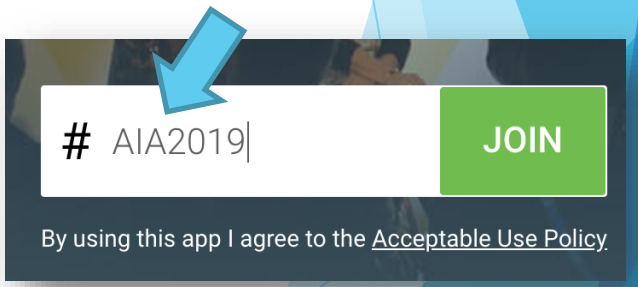
瀏覽器

輸入網址

sli.do

繼續輸入

#FBA



我先舉個例子給大家參考

一個男友跟女友說：「嫁給我，我保證結婚後天天洗碗！」

女友把對話截圖下來，發佈到臉書、IG

再透過手機傳給100個朋友，這叫做「**物聯網**」

而有圖有真相，人手一張，以後不能後悔，這叫「**區塊鏈**」。

我先舉個例子給大家參考

女友用“我保證”，搜尋兩人交往以來所有的對話紀錄

發現男友總共講了八千次“我保證”這叫作「**文字探勘**」

事後發現總共有六千次沒有做到！

所以男友的“我保證” --> 達成率為25%，這叫「**大數據**」。

我先舉個例子給大家參考

然後調查各大網站發現，一般男人說話可信度為50%

但這個男友可信度卻只有25%，這叫作「**網路爬蟲**」

而最後的決策顯示「不可以嫁給他」，這叫「**AI**」。

我先舉個例子給大家參考

女友不管數據分析，還是決定嫁給男友

這叫「**人為因素操作不當**」

資料科學程序

- 1. 資料取得
- 2. 資料工程
- 3. 資料儲存
- 4. 資料分析
- 5. 資料建模（機器學習）
- 6. 資料洞察
- 7. 自動化程序與反饋機制

大數據、區塊鏈、物聯網、網路爬蟲、
文字探勘、特徵工程、人工智慧等等

FBA教材規劃

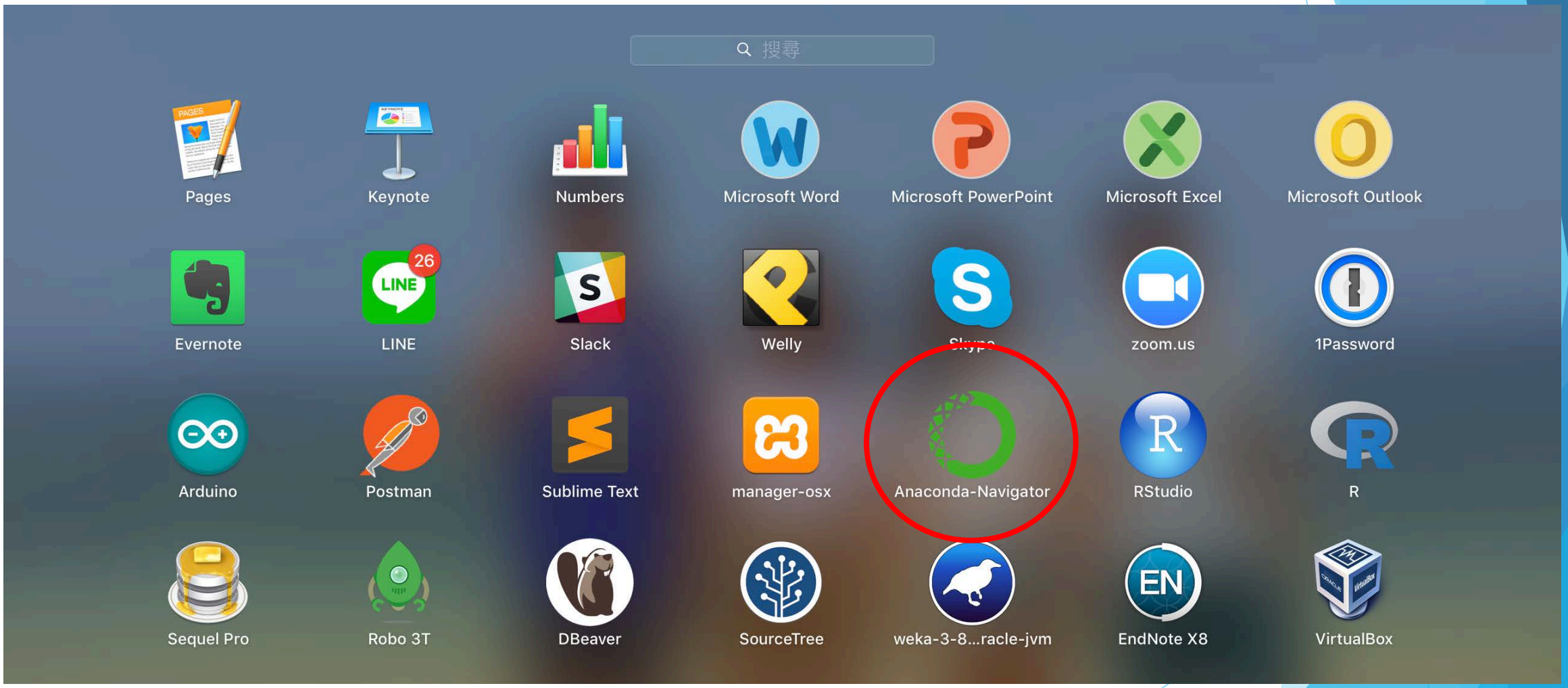
- FBA0-環境安裝：Python程式語言
- FBA1-資料取得：認識 Facebook API
- FBA2-簡單料理：EDA探索式資料分析
- FBA3-資料工程：文字探勘Text mining
- FBA4-進階分析：資料建模Data modeling

環境安裝：Python程式語言

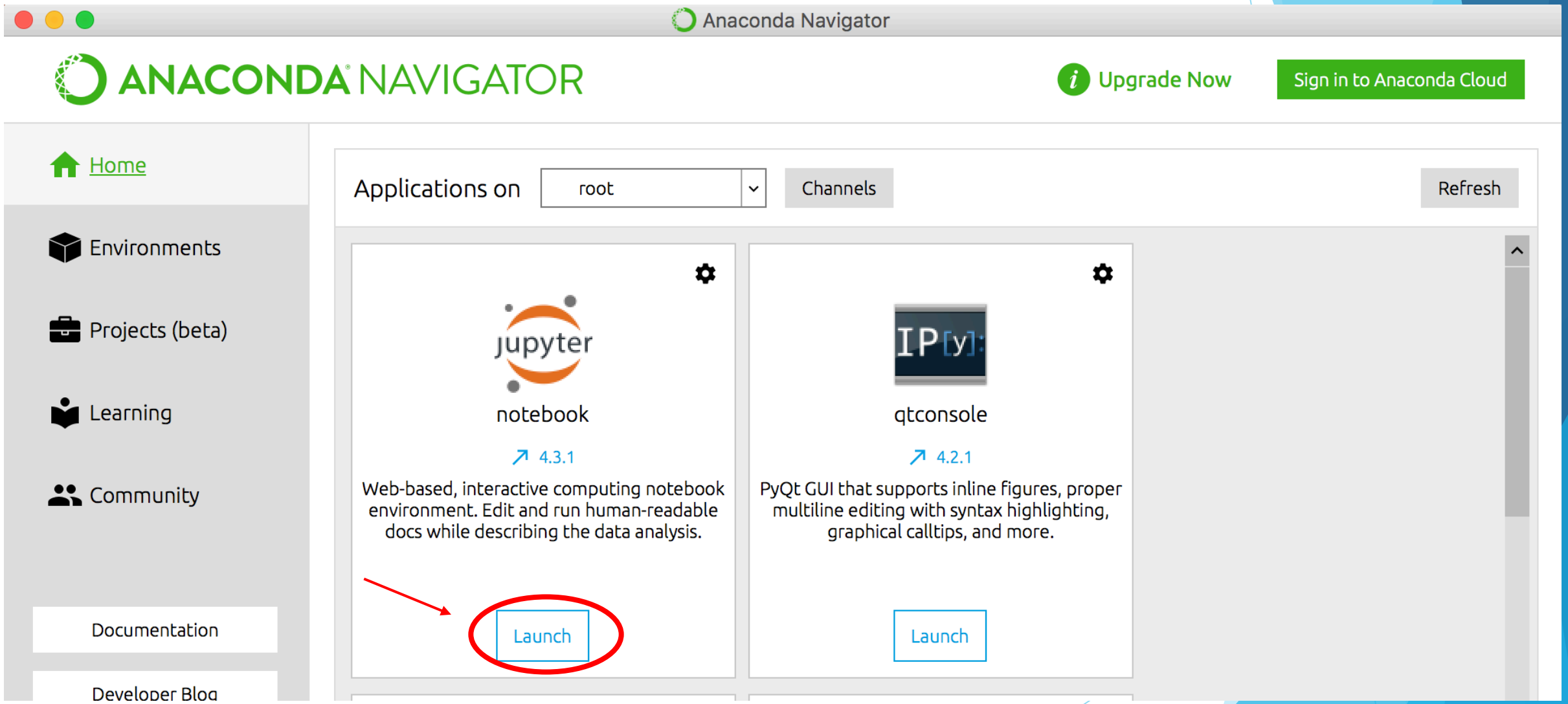
Anaconda

- 集合許多工具的軟件包
- 使用 jupyter notebook 編譯器
- 編譯器還有很多種 Ex. Sublime, spyder…
- https://www.anaconda.com/download/

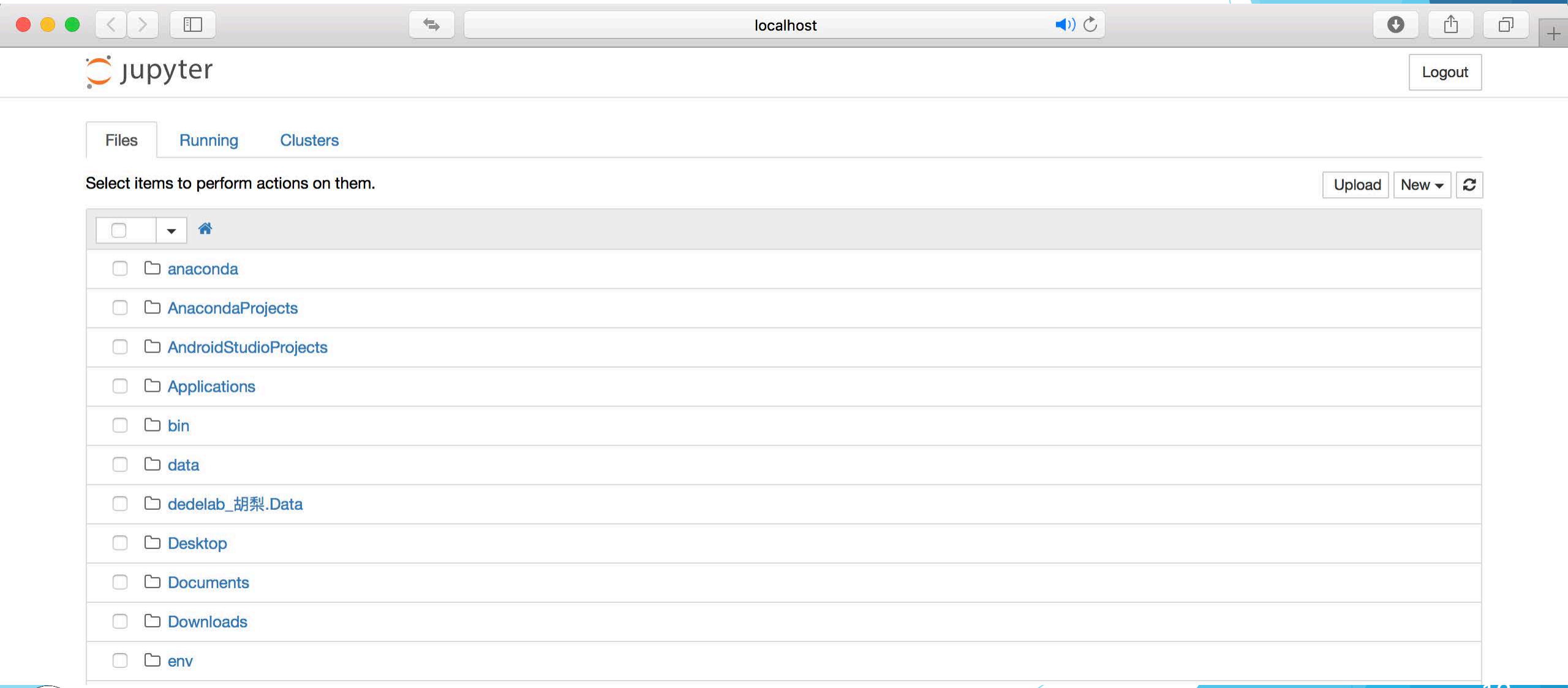
Anaconda Navigator



Anaconda Navigator

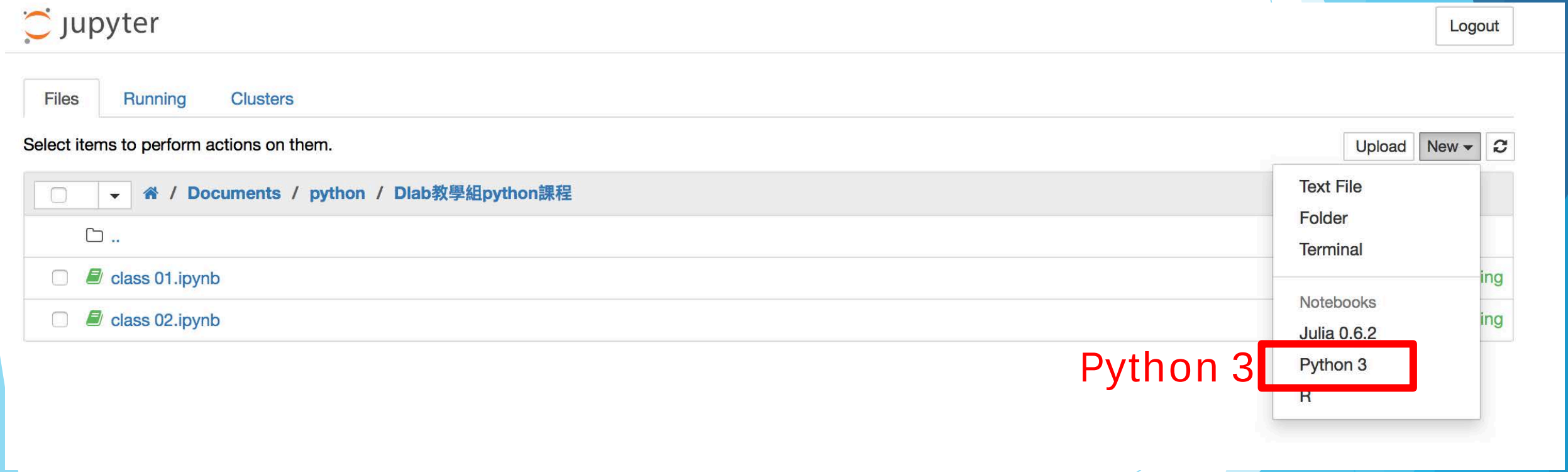


Jupyter Notebook



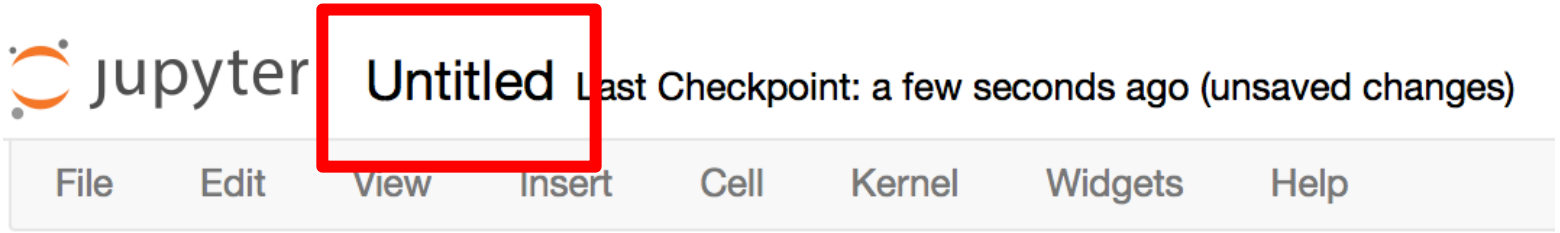
Jupyter Notebook

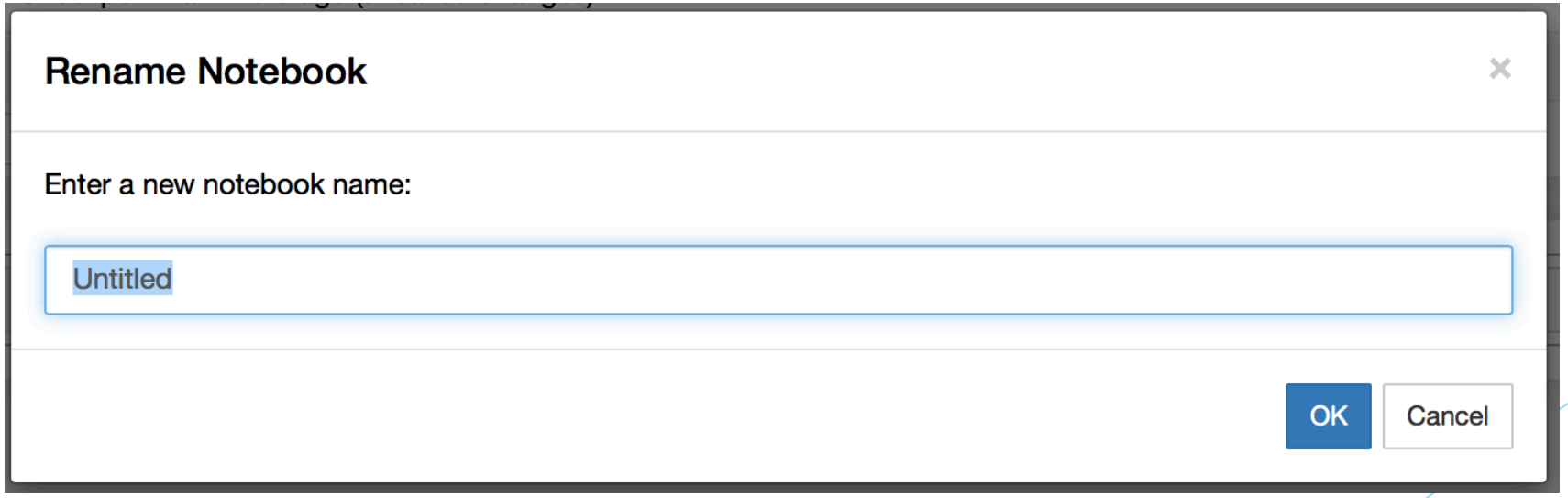
- 新增一個Notebook



Jupyter Notebook

- 重新命名

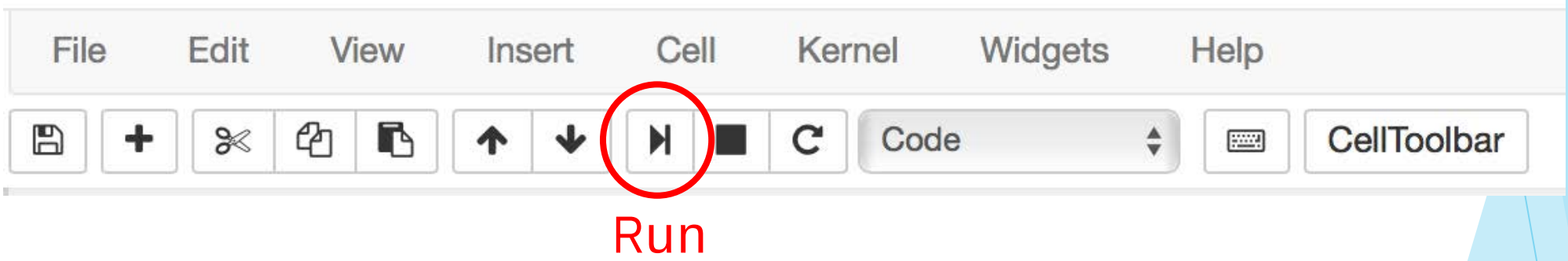




Jupyter Notebook

程式執行

- 快捷鍵1：Ctrl + Enter
- 快捷鍵2：Shift + Enter




```
print("Hello World!")
```

```
Hello World!
```

Jupyter Notebook

- Cell介紹
  - 可以分很多Cell
  - 每一個Cell獨立儲存結果

```
In [2]: # 這裡是第2個cell
        print("Hello Jurassic Park!")

        Hello Jurassic Park!
```

Jupyter Notebook

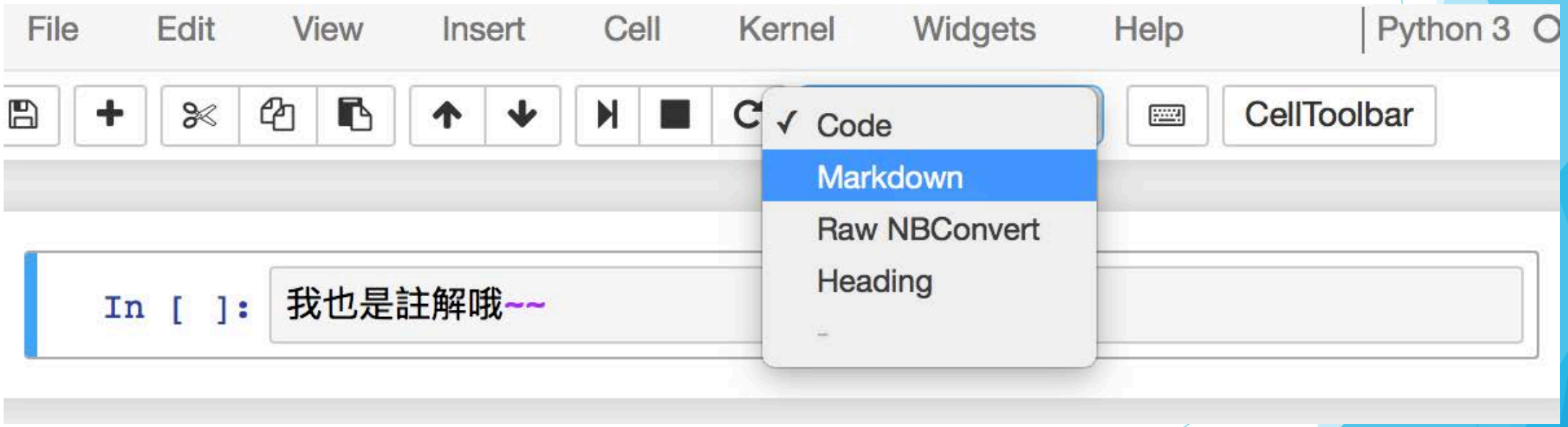
- 註解介紹
  - 註解不會被執行
  - 方便使用者 or 他人閱讀
  - 寫筆記的概念

```
# 這是註解
# 我的第一支程式
print("Hello World!")
```

```
Hello World!
```

Jupyter Notebook

- Markdown介紹
  - Jupyter notebook 獨有功能
  - 更好看的註解



Jupyter Notebook

- Markdown介紹

```
我也是註解哦~~
```

執行後

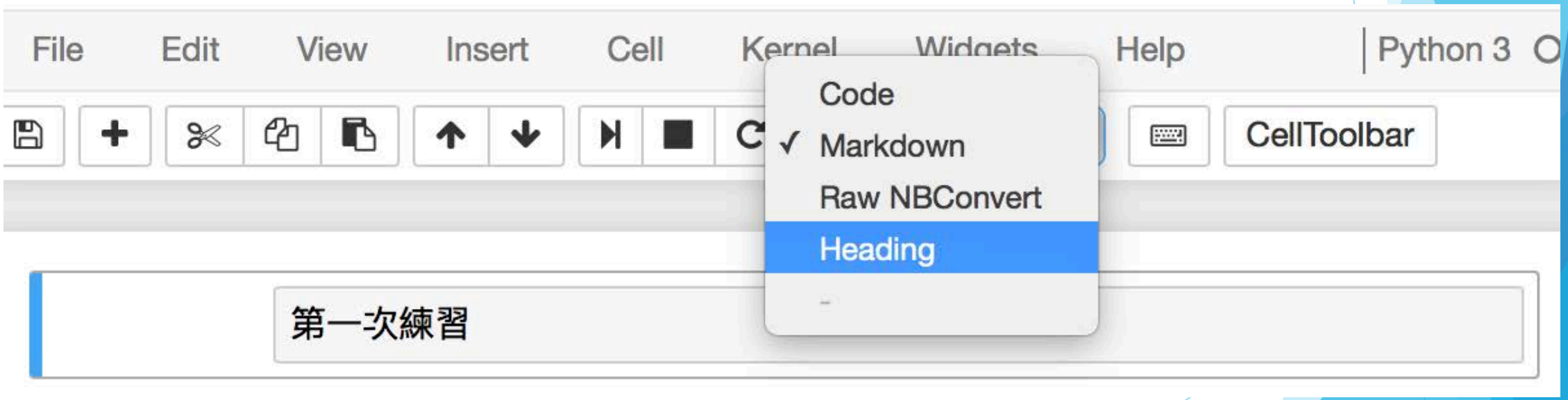
我也是註解哦~~

```
In [1]: print("Hello world!")
```

```
Hello world!
```

Jupyter Notebook

- Heading介紹
  - Jupyter notebook 獨有功能
  - 更好看的註解 Part2



Jupyter Notebook

- Heading介紹

第一次練習

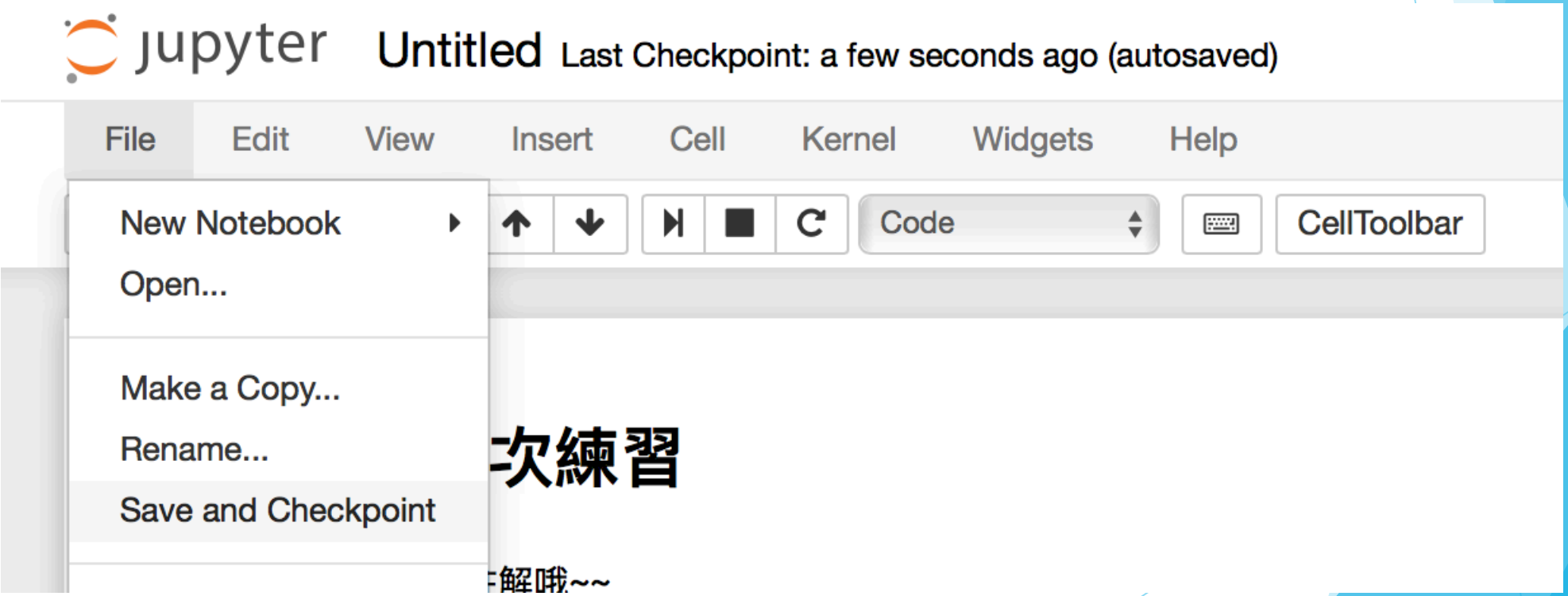
我也是註解哦~~

In [1]:
```
print("Hello world!")
```

```
Hello world!
```

Jupyter Notebook

- 儲存檔案
  - 快捷鍵：Ctrl + S



Jupyter Notebook

- 檔案位置
  - 安裝時的預設路徑
  - 可以自行修改

```
In [3]: import os
        os.getcwd()

Out[3]: '/Users/andy/Documents/python'
```

Jupyter Notebook

- 下載第三方套件 (for mac)
  - 打開終端機，輸入pip install 套件名稱

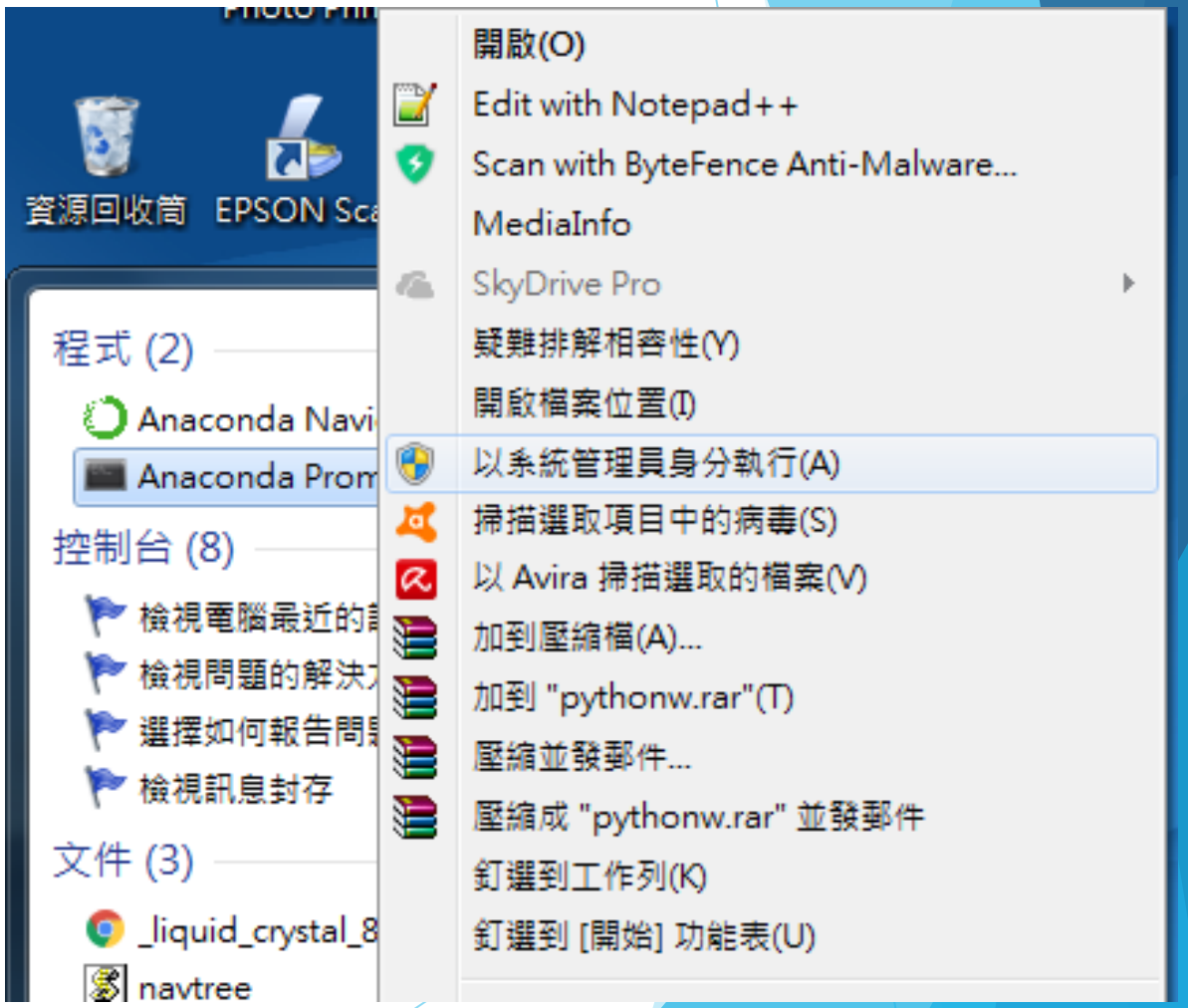
andy — -bash — 80×21

~ — jupyter-notebook ▸ python | ~ — -bash

```
Last login: Mon Jul  9 08:34:01 on ttys000
andy@Andyde-MacBook-Pro-2:~$ pip install pandas
Requirement already satisfied: pandas in ./anaconda/lib/python3.6/site-packages
(0.20.3)
Requirement already satisfied: python-dateutil>=2 in ./anaconda/lib/python3.6/si
te-packages (from pandas) (2.6.1)
Requirement already satisfied: pytz>=2011k in ./anaconda/lib/python3.6/site-pack
ages (from pandas) (2018.3)
Requirement already satisfied: numpy>=1.7.0 in ./anaconda/lib/python3.6/site-pac
kages (from pandas) (1.14.0)
Requirement already satisfied: six>=1.5 in ./anaconda/lib/python3.6/site-package
s (from python-dateutil>=2->pandas) (1.11.0)
andy@Andyde-MacBook-Pro-2:~$
```

Jupyter Notebook

- 下載第三方套件 (for windows)
  - 搜尋Anaconda Prompt (以系統管理員身份執行)
  - 輸入pip install 套件名稱



Jupyter Notebook

- 下載第三方套件 (第三種方法)
  - 在任何一個cell裡輸入pip install 套件名稱

```
!pip install pandas
```

```
Requirement already satisfied: pandas in /Users/andy/anacon
da/lib/python3.6/site-packages (0.20.3)
Requirement already satisfied: python-dateutil>=2 in /Users
/andy/anaconda/lib/python3.6/site-packages (from pandas) (2
.6.1)
Requirement already satisfied: pytz>=2011k in /Users/andy/a
naconda/lib/python3.6/site-packages (from pandas) (2018.3)
Requirement already satisfied: numpy>=1.7.0 in /Users/andy/
anaconda/lib/python3.6/site-packages (from pandas) (1.14.0)
Requirement already satisfied: six>=1.5 in /Users/andy/anac
onda/lib/python3.6/site-packages (from python-dateutil>=2->
pandas) (1.11.0)
```

資料取得：認識 Facebook API

PART 01
API介紹

什麼是API

- Application Programming Interface
- 中文是"應用程式介面”
- 負責接洽聯絡的窗口

什麼是API

- 舉例：自動販賣機
- 就像自動販賣機裡面的飲料
- 而面板和按鈕就是API
- 透過這個面板就可以拿到飲料了

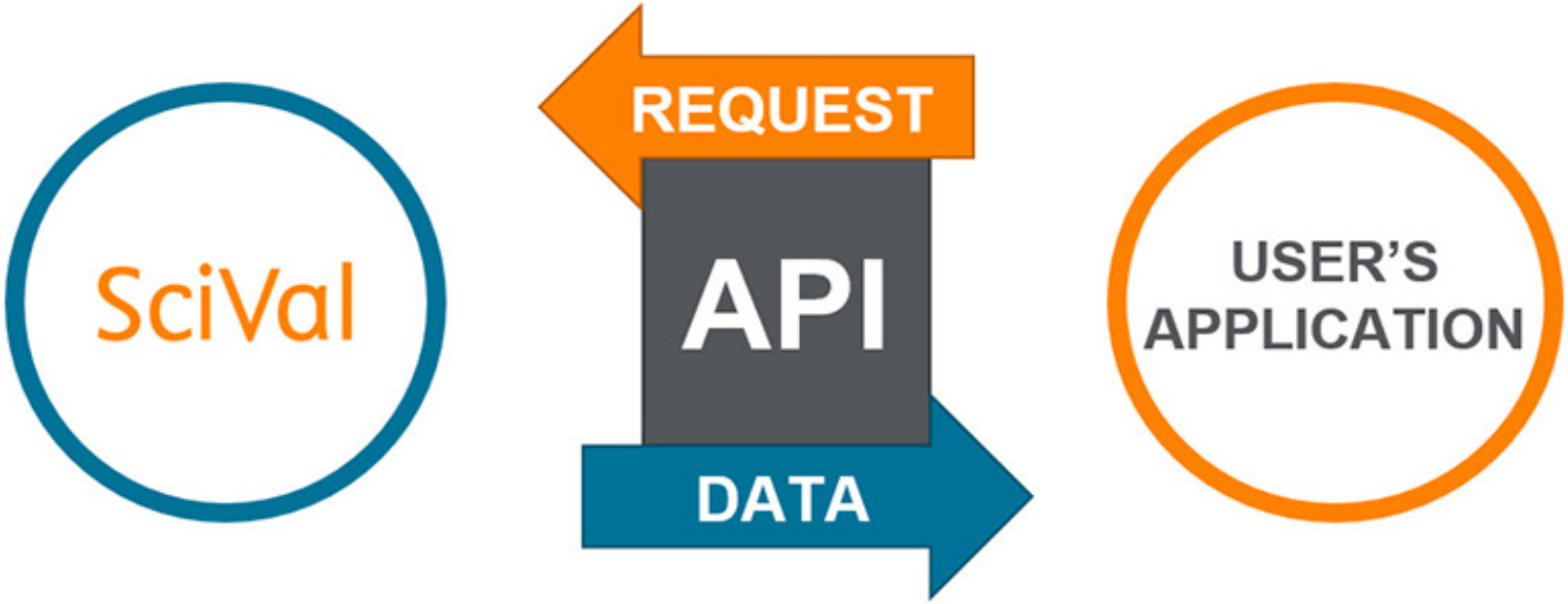
什麼是API

- 如圖說明

1. 想要一瓶可樂 (想要的資料)
2. 投幣按下按鈕 (送出請求)
3. 販賣機掉出可樂 (取得資料)

什麼是API

- API就是中間接洽的窗口



什麼是API

使用上注意

- 明明知道有賣水蜜桃口味的可樂，可是販賣機裡沒有
- API不開放、沒有 = 販賣機裡沒有賣的飲料
- 雖然我們知道Facebook有這樣的資料可是他們不開放，也就無法取得
- 有些公司的API可能也會有每月取得上限，就像販賣機的飲料賣完一樣

什麼是API

Why API ?

- 有些公司的網站不想讓你輕易爬取資料 Ex. Facebook, Twitter
- 但大家又想要取得他的資料，因此開放 API 當作窗口
- 原因是使用的網頁的 Javascript 元素放在網頁程式碼內

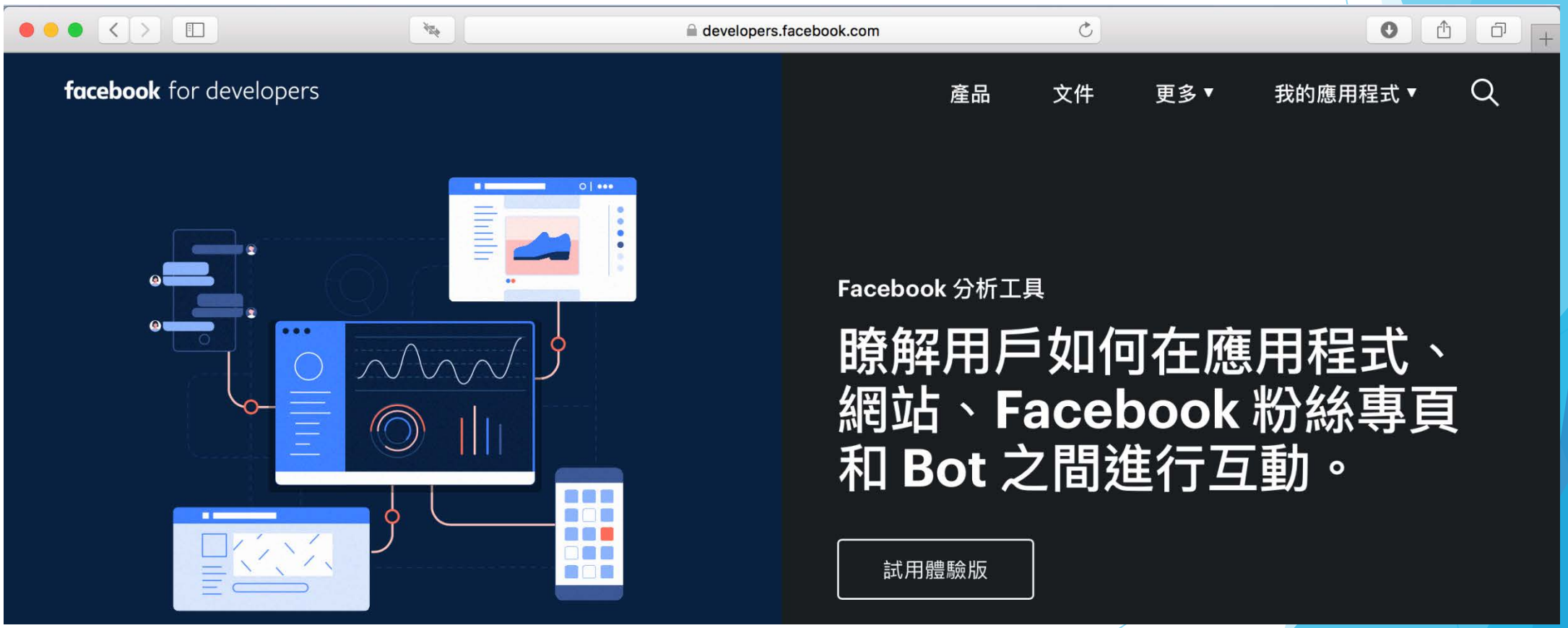
什麼是API

- Facebook API
  - 由Facebook公司向大眾釋出的資料取得窗口
  - 目前僅開放粉絲專頁、公開社團、好友的資料
  - 非好友的使用者資料因涉及隱私權問題並無開放
  - https://developers.facebook.com

PART 02
Facebook API

Facebook API

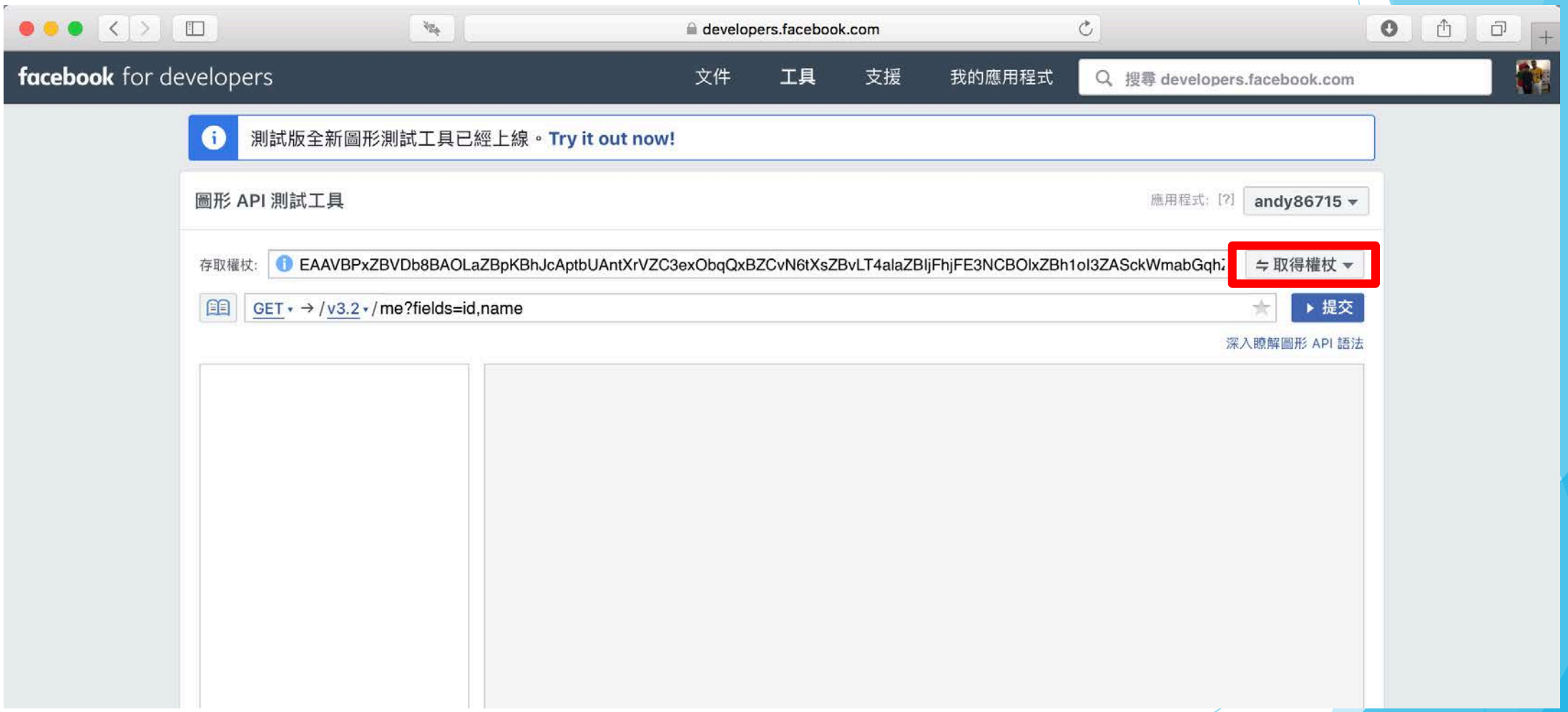
https://developers.facebook.com



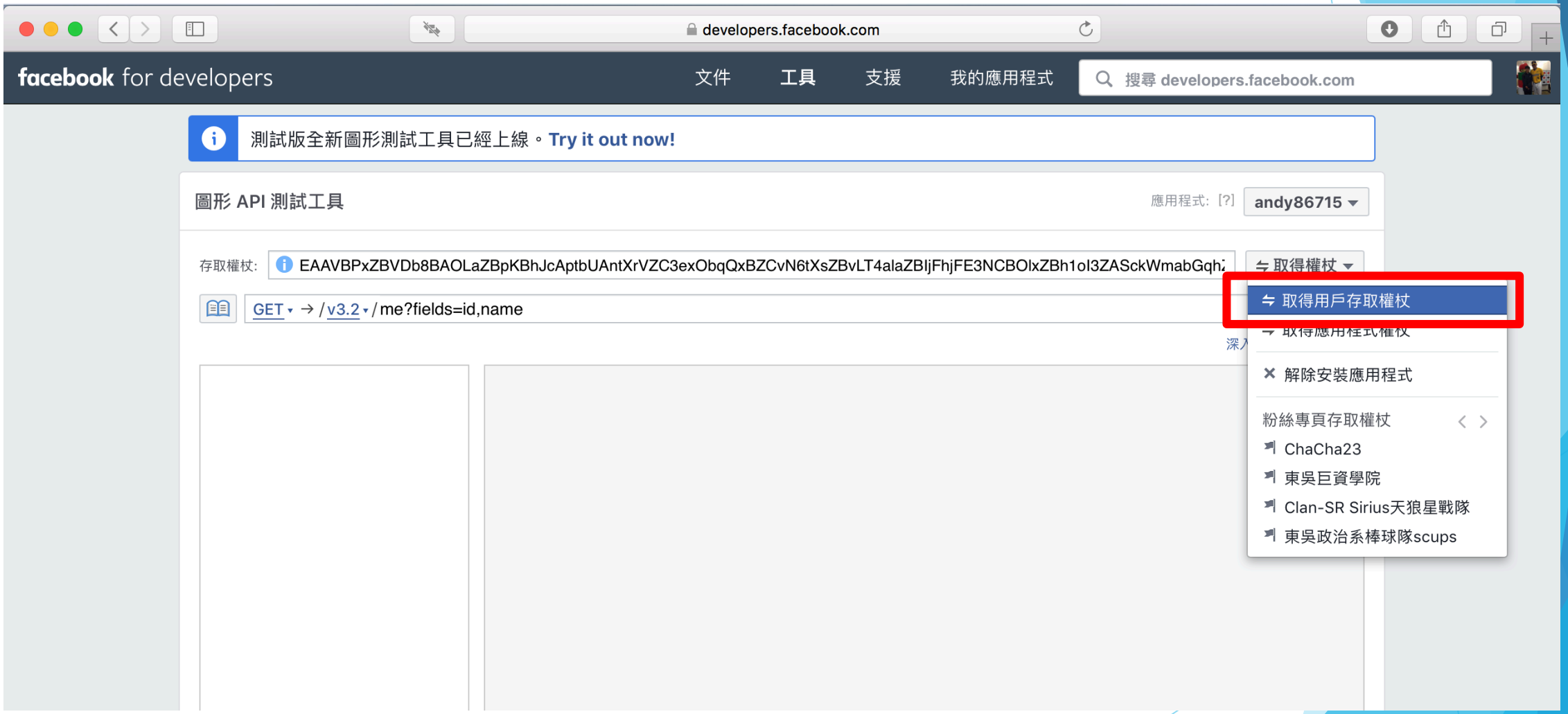
Facebook API

Facebook API

Facebook API



Facebook API



developers.facebook.com

facebook for developers　文件　工具　支援　我的應用程式　搜尋 developers.facebook.com

測試版全新圖形

圖形 API 測試工具　andy86715

存取權杖: EAAVBPxZ　取得權杖

GET → /v3.2/　提交

深入瞭解圖形 API 語法

選擇權限

v3.2

用戶數據權限

- [ ] email
- [ ] user_age_range
- [x] user_birthday
- [ ] user_friends
- [ ] user_gender
- [x] user_hometown
- [x] user_likes
- [ ] user_link
- [x] user_location
- [ ] user_photos
- [ ] user_posts
- [x] user_status
- [ ] user_tagged_places
- [ ] user_videos

活動、社團和粉絲專頁

- [x] ads_management
- [x] ads_read
- [x] business_management
- [x] groups_access_member_info
- [x] manage_pages
- [x] pages_manage_cta
- [x] pages_manage_instant_articles
- [x] pages_messaging
- [x] pages_messaging_phone_number
- [x] pages_messaging_subscriptions
- [x] pages_show_list
- [x] publish_pages
- [x] publish_to_groups
- [x] read_page_mailboxes
- [x] user_events

其他

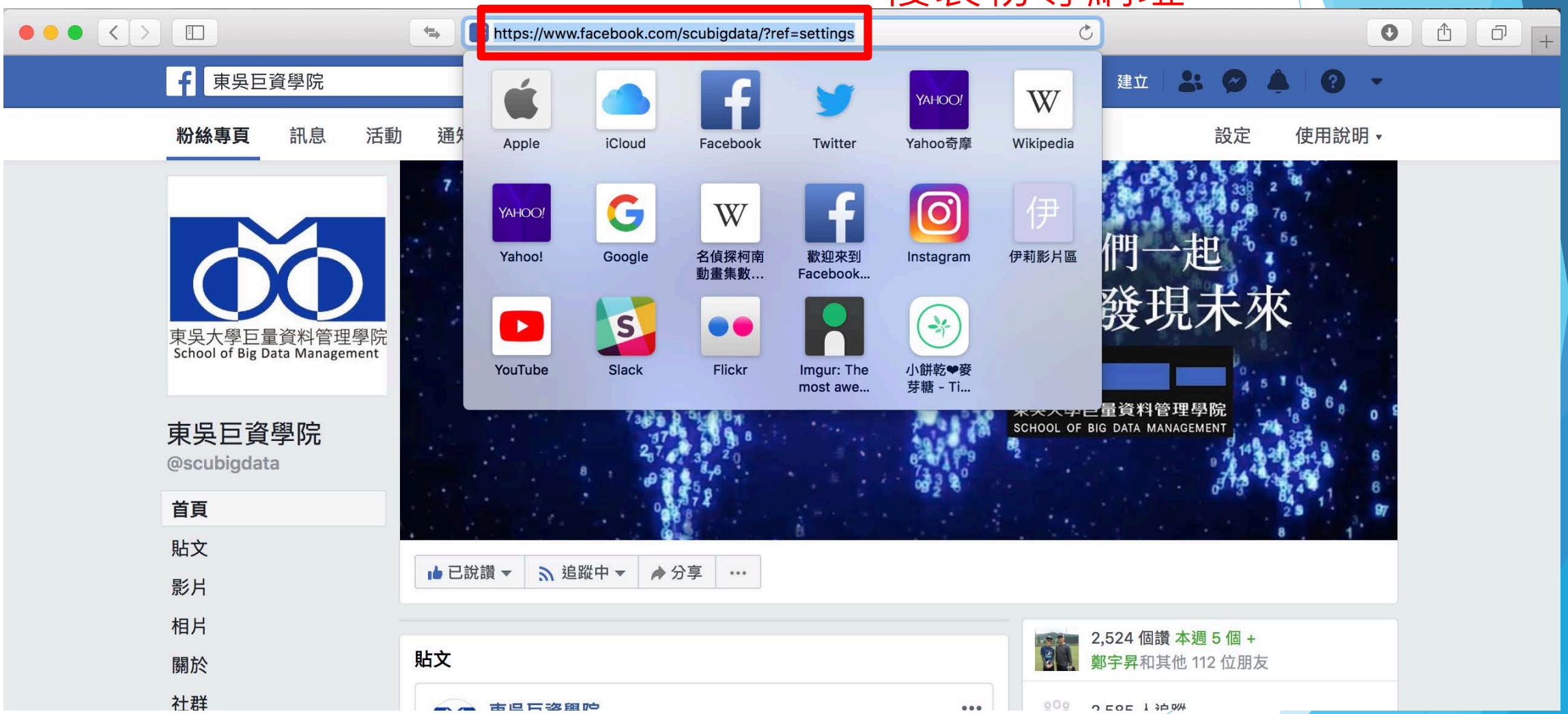
- [ ] instagram_basic
- [ ] instagram_manage_comments
- [ ] read_audience_network_insights
- [ ] leads_retrieval
- [ ] publish_video
- [ ] read_insights
- [ ] instagram_manage_insights

預設已包含公開的個人檔案。

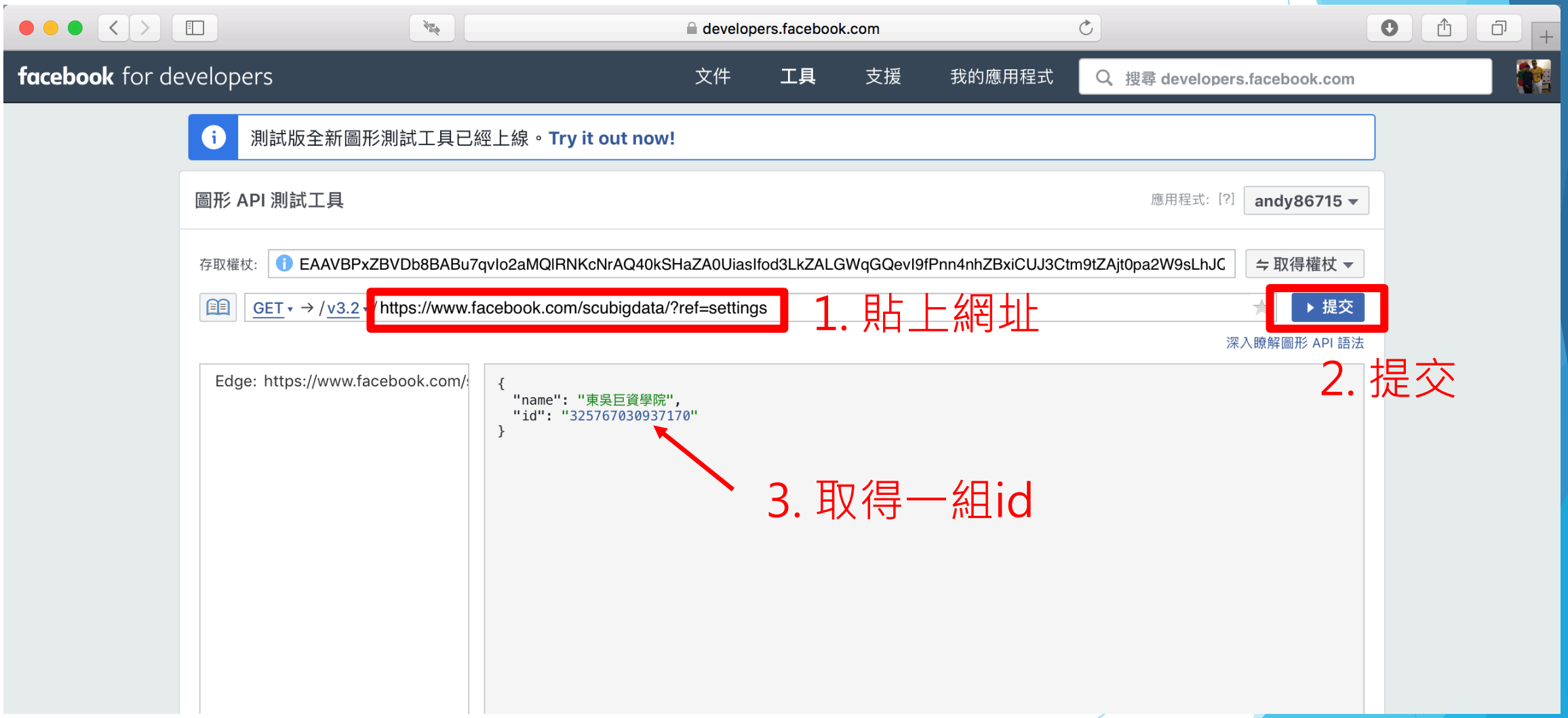
取得存取權杖　清除　取消

Facebook API

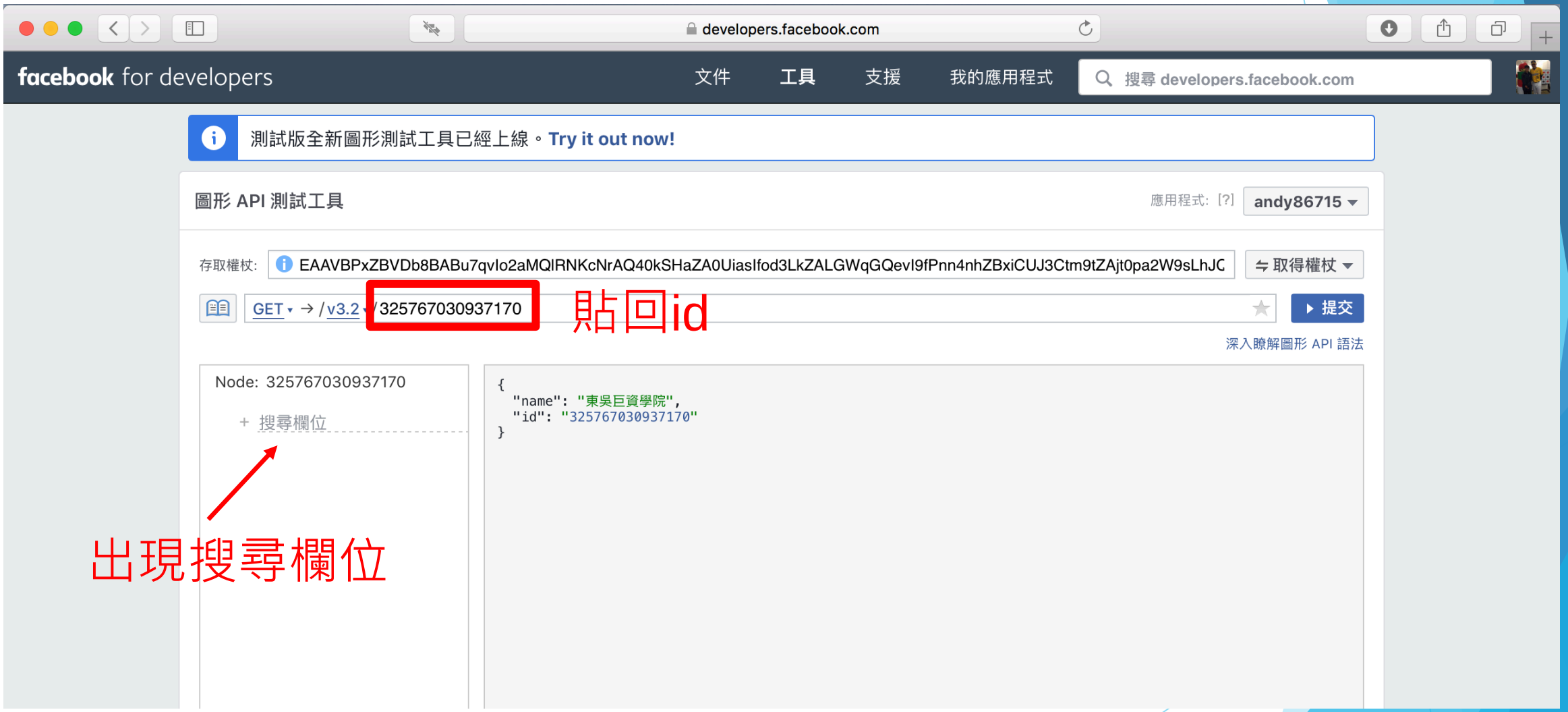
複製粉專網址



Facebook API



Facebook API



貼回id

出現搜尋欄位

Facebook API

developers.facebook.com

facebook for developers　文件　工具　支援　我的應用程式　搜尋 developers.facebook.com

測試版全新圖形測試工具已經上線。Try it out now!

圖形 API 測試工具　應用程式: [?] andy86715

存取權杖: EAAVBPxZBVDb8BALgUZCUiBQrT5fXdXavyEWn2H8hrpxMnTQZCOG69sqP9yx9VHi5K0rr857tCjhir7BXjZBvs1IGolIPuDA13EysjT(取得權杖

GET → /v3.2 / 325767030937170　提交

深入瞭解圖形 API 語法

Node: 325767030937170

+ 搜尋欄位

notifications
page_backed_instagram_acc
personas
photos
picture
place_topics
posts
product_catalogs
promotable_posts
published_posts

搜尋posts

```
{
  "name": "東吳巨資學院",
  "id": "325767030937170"
}
```

Facebook API

developers.facebook.com

facebook for developers　文件　工具　支援　我的應用程式　搜尋 developers.facebook.com

測試版全新圖形測試工具已經上線。Try it out now!

圖形 API 測試工具　應用程式: [?] andy86715

存取權杖: EAAVBPxZBVDb8BADZCFKB1MU46vniKrBGdpaqE6YkP2dZAWZA8qdcbuCtOmxs75FRvUmIkylvLGsZBdrfuR5ZCGMkCHdwSFEw　取得權杖

GET → /v3.2 / 325767030937170?fields=posts　提交

深入瞭解圖形 API 語法

Node: 325767030937170

posts

+ 搜尋欄位


```
{
  "posts": {
    "data": [
      {
        "created_time": "2018-11-06T02:55:08+0000",
        "message": "【徵才】
原友蘭教授 誠徵 #專任全職助理

➡️工作地點：東海大學
工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度
➡️工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮
➡️工作福利：
＊ 有3-4次出國機會(印尼與菲律賓)
＊ 包含住宿與機票
➡️薪資待遇：
＊ 32000-34000之間+勞健保+一個月年終

➡️其他事項：
＊ 有碩士學會就可申請在東海兼課
＊ 會幫忙申請教師證
```

Facebook API

developers.facebook.com

facebook for developers　文件　工具　支援　我的應用程式　搜尋 developers.facebook.com

測試版全新圖形測試工具已經上線。Try it out now!

圖形 API 測試工具　應用程式: [?] andy86715

存取權杖: EAAVBPxZBVDb8BAKLoaGe3ZBX57uSZBzZAVoxGYpfpwyaflZAk1FjFaJZBsaxuNoOye140yZBo1EwurQUcd3QInGjyD4tZBSy0qhD　取得權杖

GET → /v3.2 / 325767030937170?fields=posts,likes　提交

深入瞭解圖形 API 語法

Node: 325767030937170

- posts
- likes

+ 搜尋欄位

leadgen_legal_content
leadgen_qualifiers
leadgen_whitelisted_users
likes
live_encoders
live_videos
locations
media_fingerprints


```
{
  "posts": {
    "data": [
      {
        "created_time": "2018-11-06T02:55:08+0000",
        "message": "【徵才】
原友蘭教授 誠徵 #專任全職助理

工作地點：東海大學
工時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度
工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮
工作福利：
3一次出國機會(印尼與菲律賓)
含住宿與機票
薪資待遇：
2000-34000之間+勞健保+一個月年終

其他事項：
碩士學會就可申請在東海兼課
幫忙申請教師證
```


其他還有按讚數量等等

Facebook API

developers.facebook.com

facebook for developers　文件　工具　支援　我的應用程式　搜尋 developers.facebook.com

測試版全新圖形測試工具已經上線。Try it out now!

圖形 API 測試工具　應用程式: [?] andy86715

存取權杖: EAAVBPxZBVDb8BADZCFKB1MU46vniKrBGdpaqE6YkP2dZAWZA8qdcbuCtOmxs75FRvUmIkyIvLGsZBdrfuR5ZCGMkCHdwSFEw　取得權杖

GET → /v3.2 / 325767030937170?fields=posts,likes　提交

深入瞭解圖形 API 語法

小箭頭

Node: 325767030937170

- posts
- likes

+ 搜尋欄位

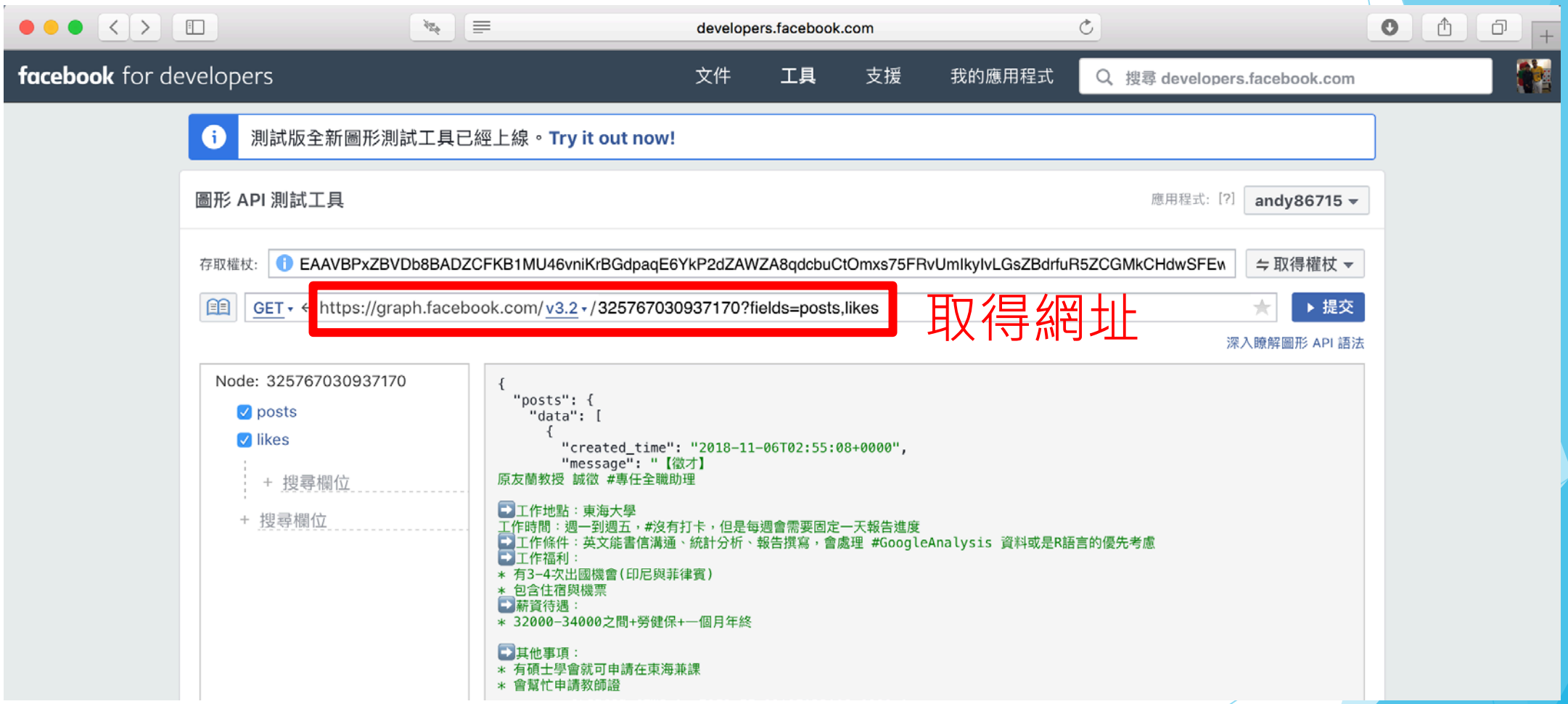
+ 搜尋欄位


```
{
  "posts": {
    "data": [
      {
        "created_time": "2018-11-06T02:55:08+0000",
        "message": "【徵才】
原友蘭教授 誠徵 #專任全職助理

➡工作地點：東海大學
工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度
➡工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮
➡工作福利：
＊ 有3-4次出國機會(印尼與菲律賓)
＊ 包含住宿與機票
➡薪資待遇：
＊ 32000-34000之間+勞健保+一個月年終

➡其他事項：
＊ 有碩士學會就可申請在東海兼課
＊ 會幫忙申請教師證
```

Facebook API



Facebook API

- 官方文件：https://developers.facebook.com/docs/graph-api/using-graph-api/#fieldexpansion
- 欄位文件：https://developers.facebook.com/docs/graph-api/reference

PART 03
Python串接API

Python串接API

- 匯入套件
  - pip install requests

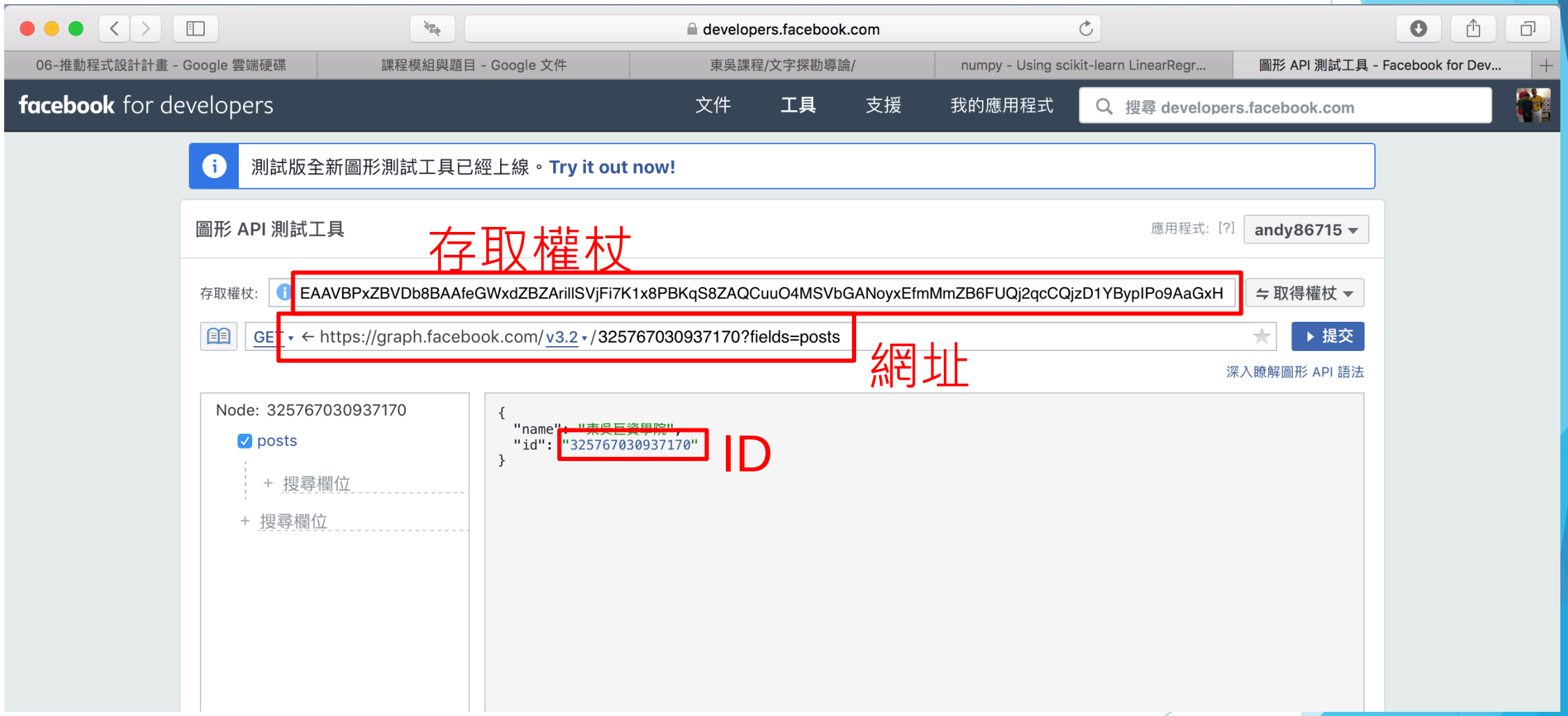
```
import json
import requests
```

Python串接API

- 串接API
  - 需要存取權杖、網址以及粉專ID

```
token = '存取權杖'
res = requests.get('搜尋欄位的網址'+'&access_token={}'.format('粉專ID', token))
```

Python串接API



Python串接API

- 串接API
  - 實際的網址非常長一段
  - 因爲除了文章以外還要取得按讚數、分享數、留言數等等資料

```
token =
'EAAVBPxZBVDb8BAIk78UfZBuID04j8VrrPT29IOOVUB8Dmo5WZCa7uFIRlhpGjiW1ZBmiKLwS769
xukxNUZAzoNri57UrL5ELd9ZBkY9xlre6WgYXWl49QHK0qiBZBAG5e35bUTWVGhEPDhoxv5tbehZB
eDybBZBO6Qr9K3C28yfQrQtHnmuFv8QQZCeDws22ZBB1NcZD'
res =
requests.get('https://graph.facebook.com/v2.9/{}/posts&access_token={}'.forma
t(325767030937170, token))
```

https://graph.facebook.com/v2.9/{}/posts?fields=id,message,type,created_time,link,shares,comments.limit(0).summary(true),likes.limit(0).summary(total_count).as(reaction_like),reactions.type(LOVE).limit(0).summary(total_count).as(reactions_love),reactions.type(WOW).limit(0).summary(total_count).as(reactions_wow),reactions.type(HAHA).limit(0).summary(total_count).as(reactions_haha),reactions.type(SAD).limit(0).summary(total_count).as(reactions_sad),reactions.type(ANGRY).limit(0).summary(total_count).as(reactions_angry)&access_token={}

Python串接API

串接API

- 使用 json 讀取抓取格式

```
fanpage = json.loads(res.text)
print(fanpage)
```


```
{'data': [{'id': '325767030937170_1050616578452208', 'message': '【徵才】\n原友蘭教授 誠徵 #專任全職助理\n\n→工作地點：東海大學\n工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度\n→工作條件：英文能書信溝通、統計分析、報告撰寫，會處理#GoogleAnalysis 資料或是R語言的優先考慮\n→工作福利：\n* 有3-4次出國機會(印尼與菲律賓)\n* 包含住宿與機票\n→薪資待遇：\n* 32000-34000之間+勞健保+一個月年終\n\n→其他事項：\n* 有碩士學會就可申請在東海兼課\n* 會幫忙申請教師證\n\n→聯繫方式：\n* yoyoyuan@go.thu.edu.tw\n* 0910-659134', 'type': 'status', 'created_time': '2018-11-06T02:55:08+0000', 'shares': {'count': 1}, 'comments': {'data':
```

Python串接API

串接API

- 資料位在 Key 值為 data 的名稱內

```
fanpage['data']
```

```
[{'comments': {'data': [],
    'summary': {'can_comment': True, 'order': 'ranked', 'total_count': 0}},
  'created_time': '2018-11-06T02:55:08+0000',
  'id': '325767030937170_1050616578452208',
  'message': '【徵才】\n原友蘭教授 誠徵 #專任全職助理\n\n→工作地點：東海大學\n工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度\n→工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮\n→工作福利：\n* 有3-4次出國機會(印尼與菲律賓)\n* 包含住宿與機票\n→薪資待遇：\n* 32000-34000之間+勞健保+一個月年終\n\n→其他事項：\n* 有碩士學會就可申請在東海兼課\n* 會幫忙申請教師證\n\n→聯繫方式：\n* yoyoyuan@go.thu.edu.tw\n* 0910-659134',
```

Python串接API

串接API

- 一筆 Json 內共有25筆資料

```
len(fanpage['data'])
```

```
25
```

Python串接API

串接API

- 查看第一筆資料

```
fanpage['data'][0]
```

```
{'comments': {'data': [],
  'summary': {'can_comment': True, 'order': 'ranked', 'total_count': 0}},
 'created_time': '2018-11-06T02:55:08+0000',
 'id': '325767030937170_1050616578452208',
 'message': '【徵才】\n原友蘭教授 誠徵 #專任全職助理\n\n→工作地點：東海大學\n工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度\n→工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮\n→工作福利：\n* 有3-4次出國機會(印尼與菲律賓)\n* 包含住宿與機票\n→薪資待遇：\n* 32000-34000之間+勞健保+一個月年終\n\n→其他事項：\n* 有碩士學會就可申請在東海兼課\n* 會幫忙申請教師證\n\n→聯繫方式：\n* yoyoyuan@go.thu.edu.tw\n* 0910-659134',
```

Python串接API

串接API

- 發文時間

```
fanpage['data'][0]['created_time']
```

```
'2018-11-06T02:55:08+0000'
```

- 文章ID

```
fanpage['data'][0]['id']
```

```
'325767030937170_1050616578452208'
```

Python串接API

串接API

- 文章內容

```
fanpage['data'][0]['message']
```

```
'【徵才】\n原友蘭教授 誠徵 #專任全職助理\n\n→工作地點：東海大學\n工作時間：週一到週五，#沒有打卡，但是每週會需要固定一天報告進度\n→工作條件：英文能書信溝通、統計分析、報告撰寫，會處理 #GoogleAnalysis 資料或是R語言的優先考慮\n→工作福利：\n* 有3-4次出國機會(印尼與菲律賓)\n* 包含住宿與機票\n→薪資待遇：\n* 32000-34000之間+勞健保+一個月年終\n\n→其他事項：\n* 有碩士學會就可申請在東海兼課\n* 會幫忙申請教師證\n\n→聯繫方式：\n* yoyoyuan@go.thu.edu.tw\n* 0910-659134'
```

Python串接API

- 串接API
  - 不同的按讚數量 Ex.讚, 驚訝, 愛心, 笑, 生氣, 難過

```
print(fanpage['data'][0]['reaction_like']['summary']['total_count'])
print(fanpage['data'][0]['reactions_wow']['summary']['total_count'])
print(fanpage['data'][0]['reactions_love']['summary']['total_count'])
print(fanpage['data'][0]['reactions_haha']['summary']['total_count'])
print(fanpage['data'][0]['reactions_angry']['summary']['total_count'])
print(fanpage['data'][0]['reactions_sad']['summary']['total_count'])
```

```
12
0
0
0
0
0
```

Python串接API

串接API

- 分享數量

```
fanpage['data'][0]['shares']['count']
```

```
1
```

- 留言數量

```
fanpage['data'][0]['comments']['summary']['total_count']
```

```
2
```

Python串接API

串接API

- 把資訊存入 list 裡

```
time, ID, context, like, wow, love, haha, angry, sad, share, comment = [], [], [], [],
[], [], [], [], [], [], []
for i in fanpage['data']:
    time.append(i['created_time'])
    ID.append(i['id'])
    context.append(i['message'])
    like.append(i['reaction_like']['summary']['total_count'])
    wow.append(i['reactions_wow']['summary']['total_count'])
    love.append(i['reactions_love']['summary']['total_count'])
    haha.append(i['reactions_haha']['summary']['total_count'])
    angry.append(i['reactions_angry']['summary']['total_count'])
    sad.append(i['reactions_sad']['summary']['total_count'])
    try:
        share.append(i['shares']['count'])
    except:
        share.append(0)
    comment.append(i['comments']['summary']['total_count'])
```

Python串接API

串接API

- 抓取存有下一個25筆資料的 Json
- 換頁資訊位在Key值為 paging 和 next 欄位裡

```
fanpage['paging']['next']
```


```
'https://graph.facebook.com/v2.9/325767030937170/posts?access_token=EAAVBPxZB
VDb8BAJib0SAxWnOtZCPBjXVnuPtUx0iVMYdq0GlcRrhxwU5sZCGv3idx0ObuWgW3UVWzHXTpk3ZB
B4gi3aIldxrgUdyyRYoZCQIvl9ZCAoZB7jXnZA8cT74gQZAXEMHj3XpK7rZCZAUeSLySdOpZCHIQN
i2Bab8Dh4HZAr0zqkYWARIYo0LpDzNms3dIQGUY746LfRfDsgZDZD&fields=id%2Cmessage%2Ct
ype%2Ccreated_time%2Clink%2Cshares%2Ccomments.limit%280%29.summary%28true%29%
2Clikes.limit%280%29.summary%28total_count%29.as%28reaction_like%29%2Creactio
ns.type%28LOVE%29.limit%280%29.summary%28total_count%29.as%28reactions_love%2
9%2Creactions.type%28WOW%29.limit%280%29.summary%28total_count%29.as%28reacti
ons_wow%29%2Creactions.type%28HAHA%29.limit%280%29.summary%28total_count%29.
```

Python串接API

- 串接API
  - 使用while迴圈設定要爬取的頁數
  - 分享數若為0並不會出現在Json檔裡，因此需做例外處理
  - 完整程式碼如下頁

```
page=2
url = fanpage['paging']['next']
while page < 5:
    res = requests.get(url)
    fanpage2 = json.loads(res.text)
    for i in fanpage2['data']:
        time.append(i['created_time'])
        ID.append(i['id'])
        context.append(i['message'])
        like.append(i['reaction_like']['summary']['total_count'])
        wow.append(i['reactions_wow']['summary']['total_count'])
        love.append(i['reactions_love']['summary']['total_count'])
        haha.append(i['reactions_haha']['summary']['total_count'])
        angry.append(i['reactions_angry']['summary']['total_count'])
        sad.append(i['reactions_sad']['summary']['total_count'])
        try:
            share.append(i['shares']['count'])
        except:
            share.append(0)
        comment.append(i['comments']['summary']['total_count'])
    url = fanpage2['paging']['next']
    page += 1
```

Python串接API

- 轉成 DataFrame 格式
  - 使用 zip() 函數將不同的 lists 合併

```
information = list(zip(time, ID, context, like, wow, love, haha, angry, sad,
share, comment))
```

  - 匯入 pandas 套件轉成 DataFrame 格式

```
import pandas as pd
df = pd.DataFrame(information,
columns=['time','id','context','like','wow','love','haha','angry','sad','shar
e','comment'])
```

PS: pip install pandas

Python串接API

轉成 DataFrame 格式

```
df.head()
```

	time	id	context	like	wow	love	haha	angry	sad	share	comment
0	2018-11-06T02:55:08+0000	325767030937170_1050616578452208	【徵才】\n原友蘭教授 誠徵 #專任全職助理\n\n→工作地點：東海大學\n工作時間：週一...	12	0	0	0	0	0	1	0
1	2018-10-26T15:30:42+0000	325767030937170_1045065325674000	【東吳巨資碩士甄試報名】\n網路蓬勃發展的時代，KOL的人數也逐漸上升，\n但究竟要如何經營...	15	0	0	0	0	0	0	0
2	2018-10-24T04:50:17+0000	325767030937170_1043766419137224	【東吳巨資碩士甄試報名】\n想要了解時下深度學習、AI究竟是什麼嗎？\n就是現在！東吳巨資碩...	36	0	2	3	0	0	8	9
3	2018-10-19T16:07:39+0000	325767030937170_1041456549368211	【東吳巨資碩士甄試報名】\n還在煩惱報名步驟繁瑣嗎？\n還在擔心是不是少準備了什麼東西嗎？\...	13	0	0	0	0	0	2	0
4	2018-10-18T11:00:01+0000	325767030937170_1040710716109461	【東吳巨資碩士甄試報名】\n現在是數據化的時代，你卻還一片徬徨嗎？\n想要學以致用，習得一些...	50	0	1	0	0	0	11	0

Python串接API

- 轉成 DataFrame 格式
  - 連同第一次換頁前爬的共爬取了5頁，100筆資料

```
df.info()
```

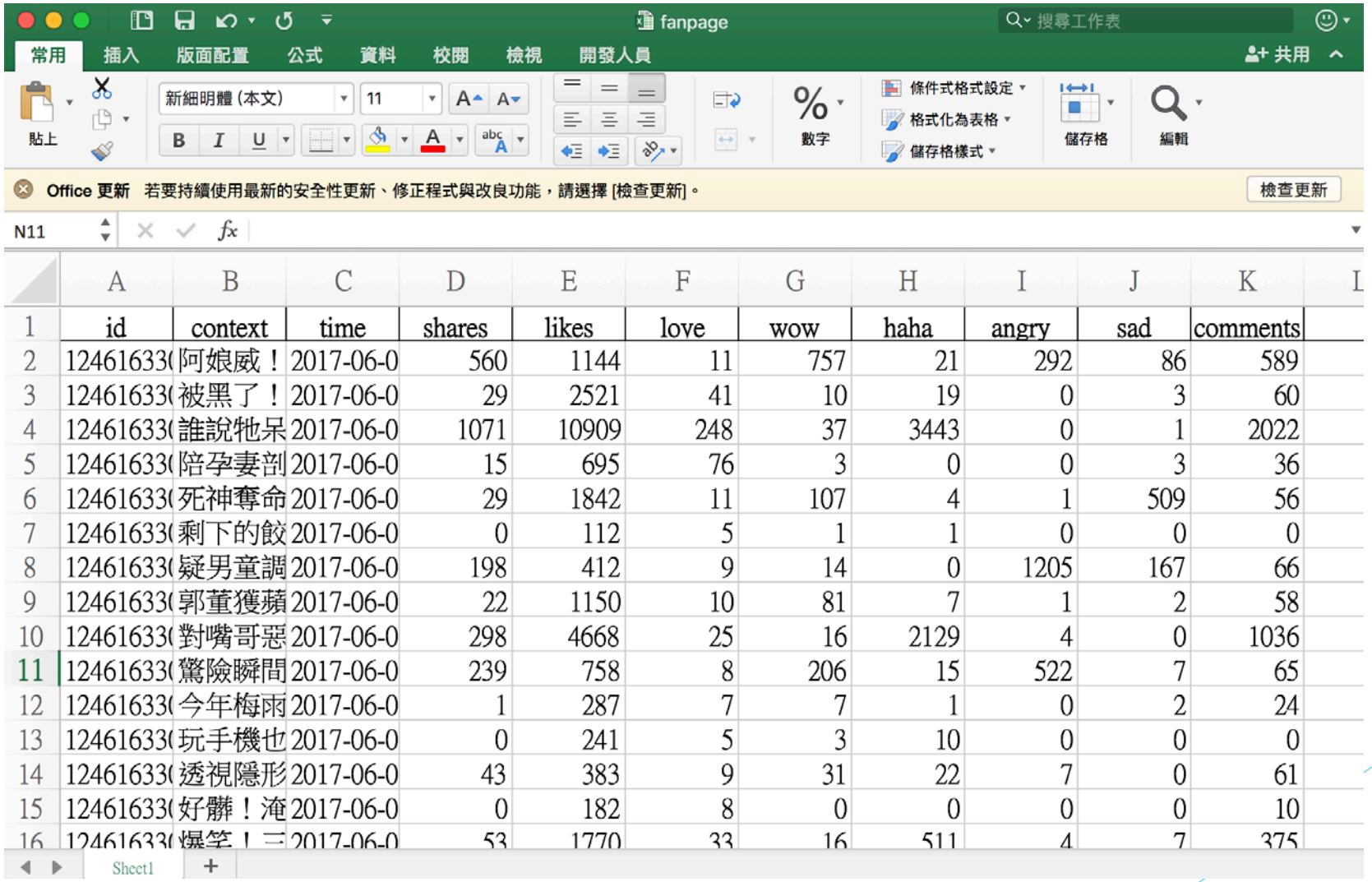
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 11 columns):
time        100 non-null object
id          100 non-null object
context     100 non-null object
like        100 non-null int64
wow         100 non-null int64
love        100 non-null int64
haha        100 non-null int64
angry       100 non-null int64
sad         100 non-null int64
share       100 non-null int64
comment     100 non-null int64
dtypes: int64(8), object(3)
memory usage: 8.7+ KB
```

Python串接API

- 轉成 DataFrame 格式
  - 匯出成 Excel 檔案

```
df.to_excel("粉絲專頁資料.xlsx", index=False)
```

Python串接API



	A	B	C	D	E	F	G	H	I	J	K
1	id	context	time	shares	likes	love	wow	haha	angry	sad	comments
2	12461633(	阿娘威！	2017-06-0	560	1144	11	757	21	292	86	589
3	12461633(	被黑了！	2017-06-0	29	2521	41	10	19	0	3	60
4	12461633(	誰說牠呆	2017-06-0	1071	10909	248	37	3443	0	1	2022
5	12461633(	陪孕妻剖	2017-06-0	15	695	76	3	0	0	3	36
6	12461633(	死神奪命	2017-06-0	29	1842	11	107	4	1	509	56
7	12461633(	剩下的餃	2017-06-0	0	112	5	1	1	0	0	0
8	12461633(	疑男童調	2017-06-0	198	412	9	14	0	1205	167	66
9	12461633(	郭董獲蘋	2017-06-0	22	1150	10	81	7	1	2	58
10	12461633(	對嘴哥惡	2017-06-0	298	4668	25	16	2129	4	0	1036
11	12461633(	驚險瞬間	2017-06-0	239	758	8	206	15	522	7	65
12	12461633(	今年梅雨	2017-06-0	1	287	7	7	1	0	2	24
13	12461633(	玩手機也	2017-06-0	0	241	5	3	10	0	0	0
14	12461633(	透視隱形	2017-06-0	43	383	9	31	22	7	0	61
15	12461633(	好髒！淹	2017-06-0	0	182	8	0	0	0	0	10
16	12461633(	爆笑！三	2017-06-0	53	1770	33	16	511	4	7	375

簡單料理：EDA探索式資料分析

先來個小範例

先做觀察

東森新聞
27 分鐘 ·

每月加班可達54小時 影響35萬勞工
#躲邊編：又有正當理由讓員工再拚命上班了
記者：李頂立 盧柏璁 台北報導
#加班 #時數 #勞工 東森財經

FNC.EBC.NET.TW
每月加班可達54小時 影響35萬勞工
勞基法3月開始給予加班時數鬆綁空間，相關備查資料也曝光了，目前...

東森新聞
37 分鐘 ·

墨西哥客機墜毀瞬間曝光 乘客崩潰哭喊「快開門」
#少在那編：從機艙逃出來的畫面太震撼了 恭喜你們活著回來
#墜機 #生還 #目擊 #逃生

NEWS.EBC.NET.TW | 作者：東森新聞
直擊！墨西哥客機墜毀瞬間曝光 乘客崩潰哭喊「快開門」
墨西哥國際航空一架客機在週二在起飛後遇到狂風冰雹失事墜毀，機...

收集資料：規模與範圍

東森新聞的粉絲專頁利用Python的Facebook API所抓取的9975篇文章

觀測期間：2017/3~2017/6

資料欄位	p_id	資料編號
	id	文章編號
	message	文章內文
	link	文章連結
	created_time	發文時間
	type	發文類別
	picture	圖片連結
	source	影片連結
	porperties	影片長度
	shares	分享數量
	likes	按讚數量
	love	愛心數量
	wow	驚奇數量
	haha	大笑數量
	angry	生氣數量
	sad	傷心數量
	comments	留言數量
	linktype	新聞類別

然後，列出你的想法

- 如何取得 # 後面的文字
- 排名前10%的 # 是哪些？
- 還可以做些什麼分析？
- …… 依此類推……

不小心發現

- 這些排名前10%的 # 是這些...

躲邊編、閃編、西瓜挖大編、我在這編、少在那編、在你身編、掛在嘴編.....

不小心發現...東森小編！
接著你會想分析什麼？

東森小編績效分析

定義小編的經營績效

- 按讚總數
  - 特點：可用來說明該小編對於整體經營績效標獻的總貢獻程度。
- 平均按讚數
  - 特點：可用來說明所發佈之文章是否成功與閱聽眾產生互動與共鳴。
- 發文總數
  - 特點：可用來反映小編的產值。

東森小編績效分析

定義「好」小編

- 我們透過平均按讚數、發文總數以及按讚總數來挑選這三項指標的前20名並作交集，找出績效最好的小編。
- 績效好的小編結果依序為：哈姆編、哩厝編、西瓜挖大編、吐司切編、車部編、周二編、內褲穿反編

然後，繼續列出你的想法

- 還可以做些什麼分析？
- 哪一天發文績效好？
- 表現好的小編是哪裡好？
- 每個人都適合寫同類型的文章嗎？
- 請繼續發想...

PART 01
資料前處理

資料前處理

- 匯入資料

```
import pandas as pd
df = pd.read_excel("fanpage.xlsx")
```

資料前處理

查看資料

```
df.head(2)
```

	id	context	time	shares	likes	love	wow	haha	angry	sad	comments
0	1246163309068 00_1560501197318299	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	2017-06-05T03:09:40+0000	560	1144	11	757	21	292	86	589
1	1246163309068 00_1560454417322977	被黑了！李毓芬演唱「大落拍」 網友卻意外發現「亮點」\n#條紋編：這一段應該是昨天的亮點表演...	2017-06-05T03:00:00+0000	29	2521	41	10	19	0	3	60

資料前處理

- 查看資料
  - 共有9975筆資料
  - 且發現有遺失值

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9975 entries, 0 to 9974
Data columns (total 11 columns):
id          9975 non-null object
context     9968 non-null object
time        9975 non-null object
shares      9975 non-null int64
likes       9975 non-null int64
love        9975 non-null int64
wow         9975 non-null int64
haha        9975 non-null int64
angry       9975 non-null int64
sad         9975 non-null int64
comments    9975 non-null int64
dtypes: int64(8), object(3)
memory usage: 857.3+ KB
```

資料前處理

- 處理遺失值
  - context 內容為空值的填入字串無

```
df['context'] = df['context'].fillna("無")
```

資料前處理

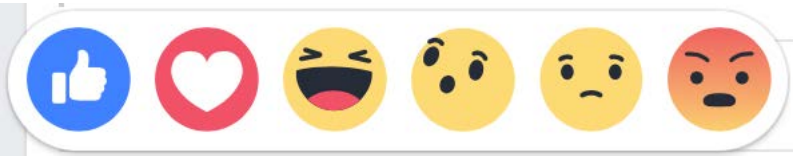
- 處理遺失值
  - 已去除空值

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9975 entries, 0 to 9974
Data columns (total 11 columns):
id          9975 non-null object
context     9975 non-null object
time        9975 non-null object
shares      9975 non-null int64
likes       9975 non-null int64
love        9975 non-null int64
wow         9975 non-null int64
haha        9975 non-null int64
angry       9975 non-null int64
sad         9975 non-null int64
comments    9975 non-null int64
dtypes: int64(8), object(3)
memory usage: 857.3+ KB
```

資料前處理

- 整理資料
  - 計算按讚數量加總
  - 新增欄位 likes_count，為五種按讚指標的加總



```
df['likes_count'] =
df['likes']+df['love']+df['wow']+df['haha']+df['angry']+df['sad']
```

資料前處理

整理資料

```
df.head(1)
```

	id	context	time	shares	likes	love	wow	haha	angry	sad	comments	likes_count
0	12461633090680 0_15605011973182 99	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	2017-06-05T03:09:40+0000	560	1144	11	757	21	292	86	589	2311

資料前處理

- 整理時間資料
  - 查看原先的時間格式

```
df['time'][0]
```

```
'2017-06-05T03::09:40+0000'
```

資料前處理

- 整理時間資料
  - 去除時間符號以及去除秒數後四位

'2017-06-05T03::09:40+0000'

```
df['time'] = df["time"].str.replace("T",'').str.replace("-",'')
df['time'] = df['time'].str.replace(':','').str.split('+').str[0]

df['time'][0]
```

'20170605030940'

資料前處理

- 整理時間資料
  - 轉換成 DataFrame 中的時間格式

```
df['time'] = pd.to_datetime(df['time'],format='%Y%m%d%H%M%S')

df['time'][0]
```

```
Timestamp('2017-06-05 03:09:40')
```

資料前處理

- 整理時間資料
  - 轉換成 DataFrame 中的時間格式

```
import datetime
df['time'] = df['time']+datetime.timedelta(hours = 8)
df['time'][0]
```

```
Timestamp('2017-06-05 11:09:40')
```

資料前處理

- 整理時間資料
  - 新增欄位 hour，取出小時
  - 轉為 object 資料型態

```
df['hour'] = df["time"].dt.hour
df['hour'] = df['hour'].astype('object')

df['hour'][0]
```

```
11
```

資料前處理

- 整理時間資料
  - 新增欄位 weekday，取出星期 (數字0~6)
  - 數字從0~6代表星期一至日，再取代成英文字串

```
df['weekday'] = df["time"].dt.weekday
df['weekday'] = df['weekday'].replace([0, 1, 2, 3, 4, 5, 6],
['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'])

df['weekday'][0]
```

```
'Monday'
```

資料前處理

- 整理時間資料

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9975 entries, 0 to 9974
Data columns (total 14 columns):
id              9975 non-null object
context         9975 non-null object
time            9975 non-null
datetime64[ns]
shares          9975 non-null int64
likes           9975 non-null int64
love            9975 non-null int64
wow             9975 non-null int64
haha            9975 non-null int64
angry           9975 non-null int64
sad             9975 non-null int64
comments        9975 non-null int64
likes_count     9975 non-null int64
hour            9975 non-null object
weekday         9975 non-null object
dtypes: datetime64[ns](1), int64(9),
object(4)
memory usage: 1.1+ MB
```

資料前處理

整理資料

```
df.head(1)
```

	id	context	time	shares	likes	love	wow	haha	angry	sad	comments	likes_count	hour	weekday
0	1246163309068001560501197318299	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	2017-06-05 11:09:40	560	1144	11	757	21	292	86	589	2311	11	Monday

資料前處理

- 取出小編名

東森新聞正在直播 — 在台北市政府。
41 分鐘 · 台北市 ·

【東森大直播】柯文哲造勢力拼陸戰！選前市府路萬人應援
#不開統編：還有學姐大跳街舞應援催票！
#台北市長 #柯文哲 #造勢 #市府 #學姐

東森新聞
3 分鐘 ·

專屬拖鞋大使！萌柴「屎盃秀」服務 羨煞眾網友
#麻花編：你們家毛小孩做過什麼貼心舉動嗎？
●柴柴越獄中！為衝前座怒鑽網 慘變菠蘿麵包 網笑翻
https://news.ebc.net.tw/News/fun/132857…… 更多

資料前處理

- 取出小編名
  - 每篇貼文的最後都會出現 #XX編，因此加以切開取出，存入 list 裡

```
curator = []
for i in df['context'].str.split('#').str[1].str.split('：').str[0]:
    try:
        if i[-1] == '編':
            curator.append(i)
        else:
            curator.append(None)
    except:
        curator.append(None)
```

資料前處理

- 取出小編名
  - 新增欄位 curator 加入小編名稱

```
df['curator'] = pd.DataFrame(curator)
```

資料前處理

- 取出小編名

```
df.head(1)
```

	id	context	time	shares	likes	love	wow	haha	angry	sad	comments	likes_count	curator
0	1246163309068000_156050119731829 9	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	2017-06-05 11:09:40	560	1144	11	757	21	292	86	589	2311	要打統編

PART 02
描述性統計

描述性統計

- 設定中文字型
  - matplotlib & seaborn 皆為Python的繪圖套件
  - pip install matplotlib
  - pip install seaborn

```
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
```

描述性統計

- 設定中文字型
  - 繪圖套件本身並不支援中文

```
from matplotlib.font_manager import FontProperties
font = FontProperties(fname=r"電腦中的中文路徑")
```

描述性統計

- 設定中文字型
  - for mac

```
font = FontProperties(fname=r"/System/Library/Fonts/STHeiti Light.ttc")
```

  - for windows

```
font = FontProperties(fname=r"C:/windows/Fonts/msjh.ttc")
```

描述性統計

基本統計

- 連續型資料統計指標

```
df.describe()
```

	shares	likes	love	wow	haha	angry	sad	comments	likes_count
count	9975.000000	9975.000000	9975.000000	9975.000000	9975.000000	9975.000000	9975.000000	9975.000000	9975.000000
mean	277.510877	3125.289925	115.573634	118.844411	262.752381	147.826065	111.721704	585.611830	3882.008120
std	1171.178025	6770.009492	500.730071	323.255901	1105.812423	942.289195	967.558397	2545.403304	8278.568139
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	6.000000	384.000000	7.000000	8.000000	3.000000	0.000000	0.000000	10.000000	465.000000
50%	40.000000	1017.000000	15.000000	28.000000	11.000000	1.000000	1.000000	52.000000	1246.000000
75%	160.000000	2795.500000	43.000000	95.000000	77.000000	10.000000	8.000000	254.000000	3515.000000
max	39140.000000	144475.000000	11394.000000	8474.000000	30265.000000	30751.000000	48997.000000	64206.000000	156850.000000

描述性統計

- 基本統計
  - 類別型資料統計
  - 小編出現次數排名

```
df['curator'].value_counts()
```

```
B編          410
內編          397
條紋編         383
悠悠編         299
哩厝編         276
惡魔在身編       276
M編          275
哈姆編         266
周二編         266
閃編          265
西瓜挖大編       248
```

描述性統計

基本統計

- 計算小編數量

```
len(df['curator'].value_counts().index)
```

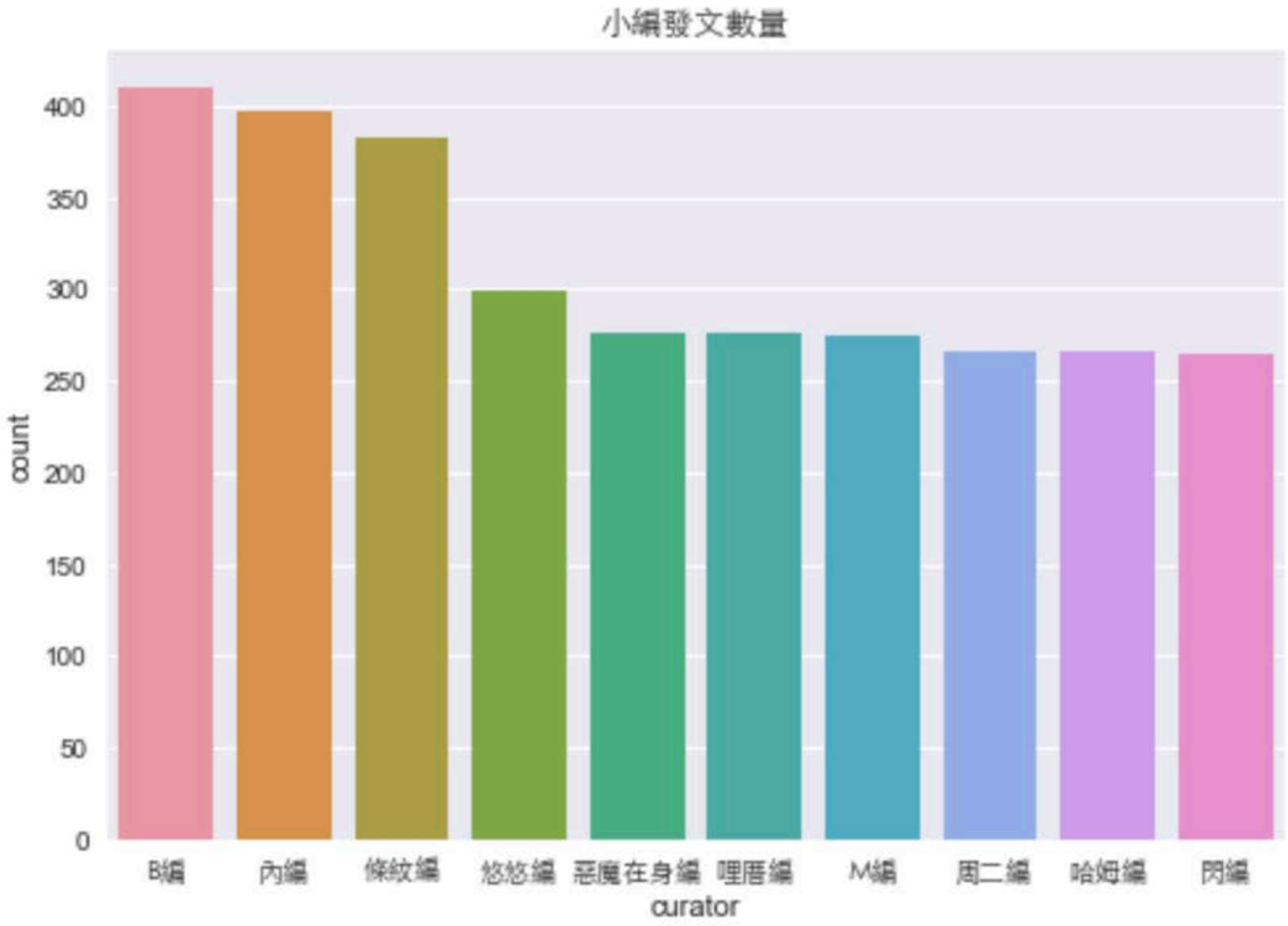
```
86
```

描述性統計

觀察小編

- 查看發文數量前10名的小編為誰

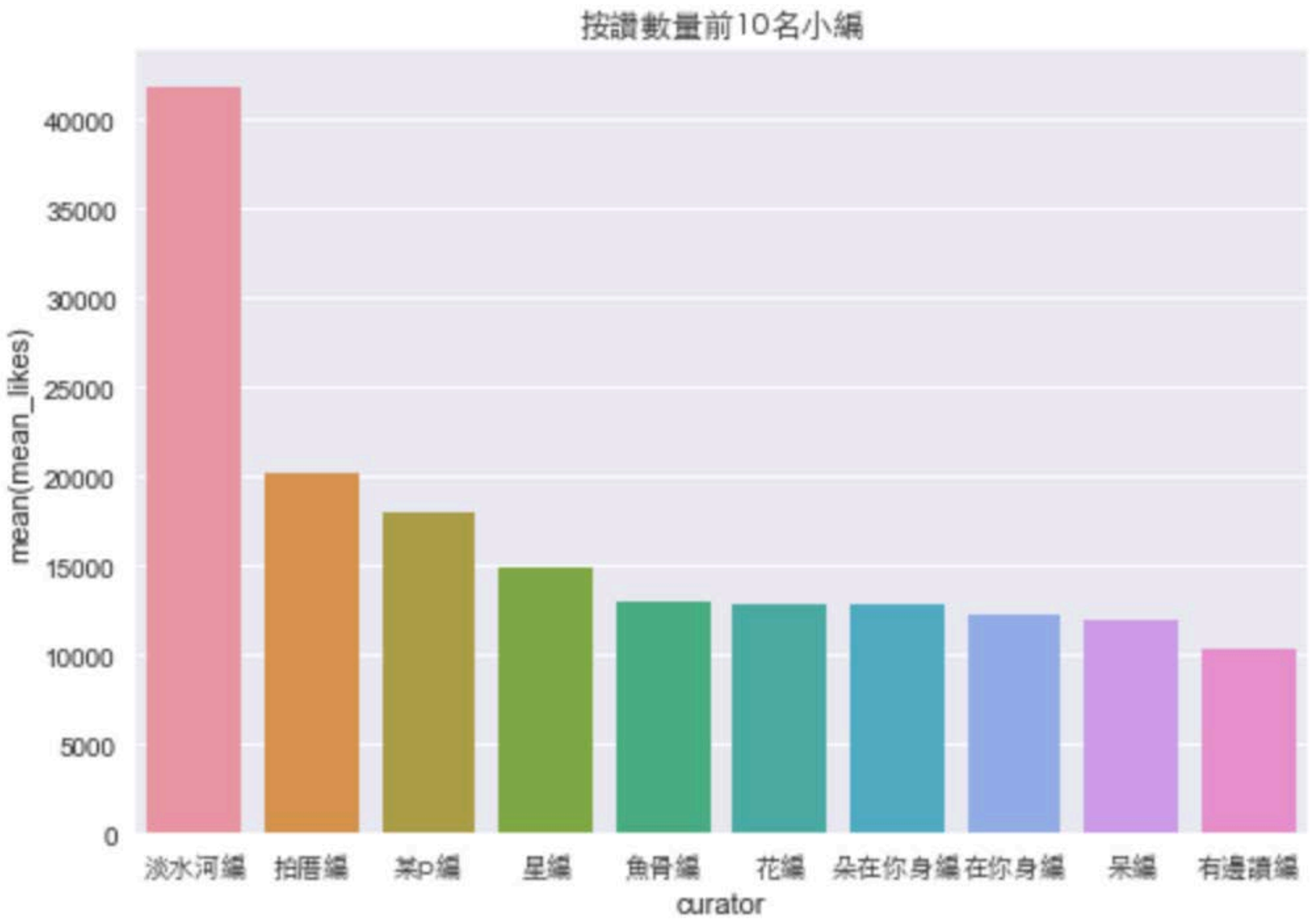
```
sns.countplot(data=df, x='curator',
              order=df['curator'].value_counts().iloc[:10].index)
plt.xticks(fontproperties=font, size=10)
plt.title("小編發文數量", fontproperties=font, size=12)
plt.show()
```


小編發文數量
count
400
350
300
250
200
150
100
50
0
B編
內編
條紋編
悠悠編
惡魔在身編
哩唇編
M編
周二編
哈姆編
閃編
curator

描述性統計

- 觀察小編
  - 查看發文按讚數量平均前10名的小編為誰

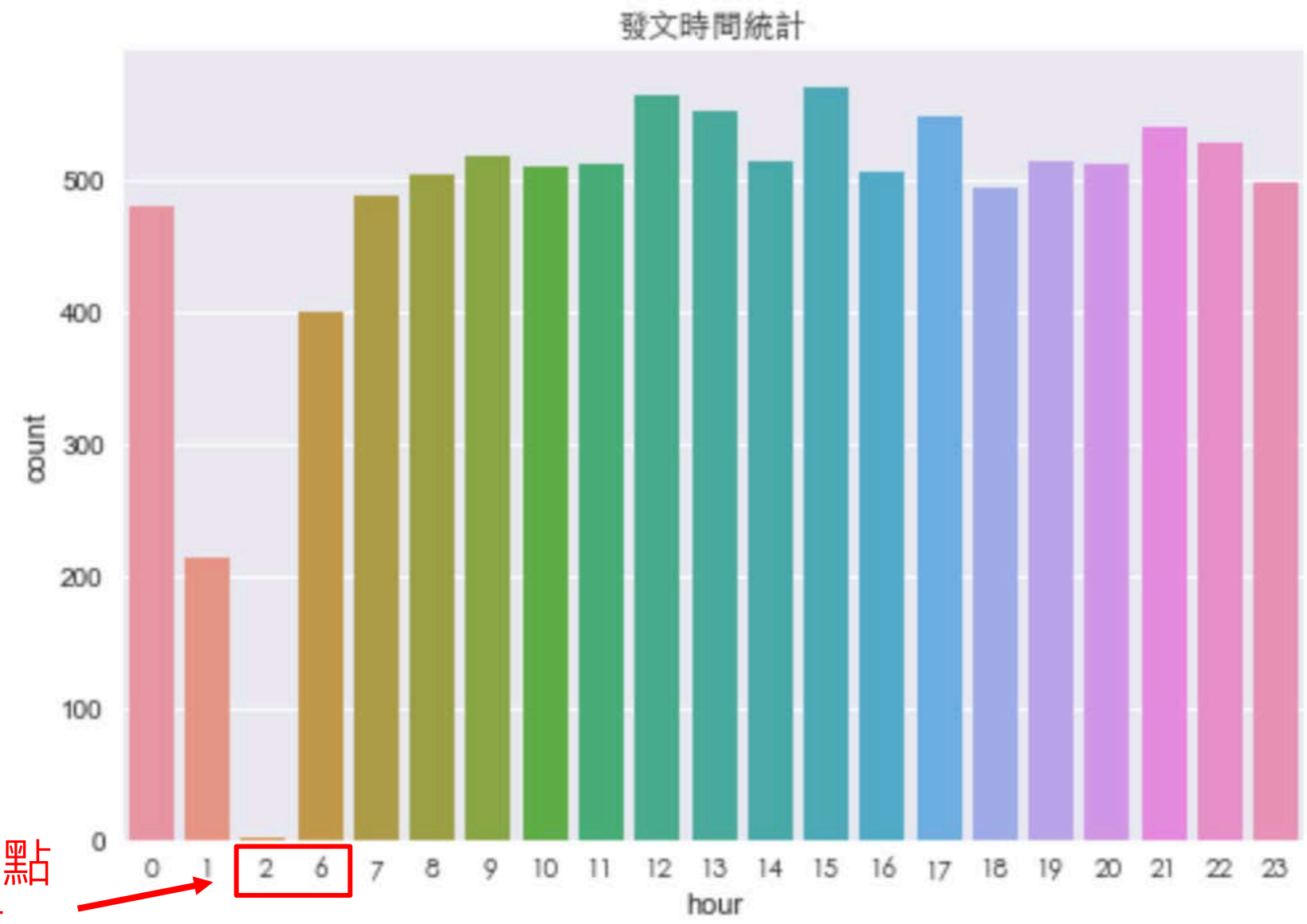
```
likes_avg = []
for i in df['curator'].value_counts().index:
    likes_avg.append([i,(df[df['curator']==i])["likes_count"].mean()])
df2 = pd.DataFrame(likes_avg, columns=['curator','mean_likes'])
df3 = df2.sort_values('mean_likes',ascending=False).head(10)
sns.barplot(x='curator', y='mean_likes', data=df3)
plt.xticks(fontproperties=font, size=10)
plt.title("按讚數量前10名小編", fontproperties=font, size=12)
plt.show()
```


按讚數量前10名小編
mean(mean_likes)
40000
35000
30000
25000
20000
15000
10000
5000
0
淡水河編
拍唇編
某p編
星編
魚骨編
花編
朵在你身編
在你身編
呆編
有邊讀編
curator

描述性統計

- 探討時間因子
  - 查看發文時間統計

```
sns.countplot(data=df, x='hour')
plt.xticks(fontproperties=font, size=10)
plt.title("發文時間統計", fontproperties=font, size=12)
plt.show()
```

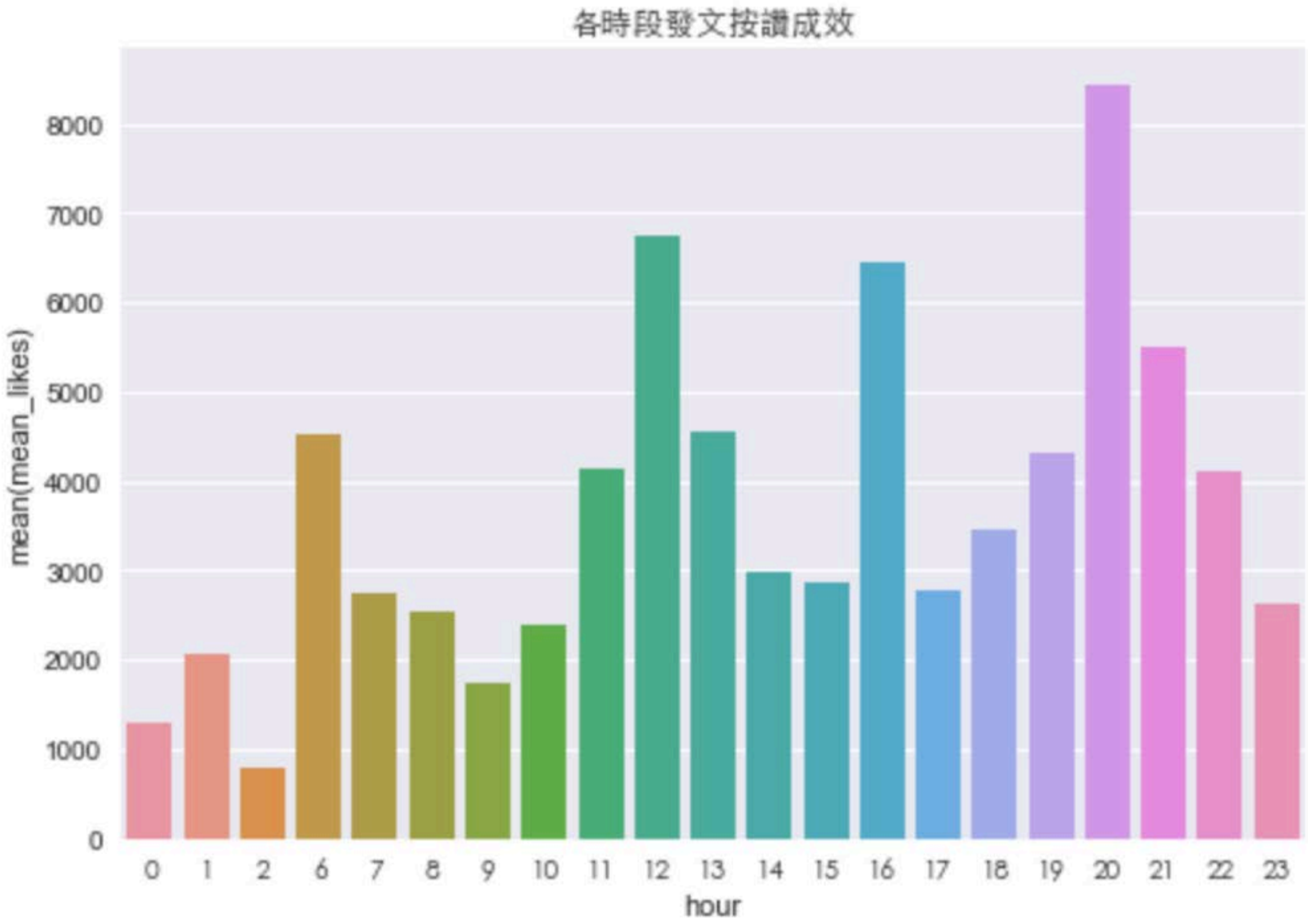

發文時間統計
count
0
100
200
300
400
500
0 1 2 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
hour

半夜3~5點
無發過文

描述性統計

- 探討時間因子
  - 查看各個時段發文的按讚成效

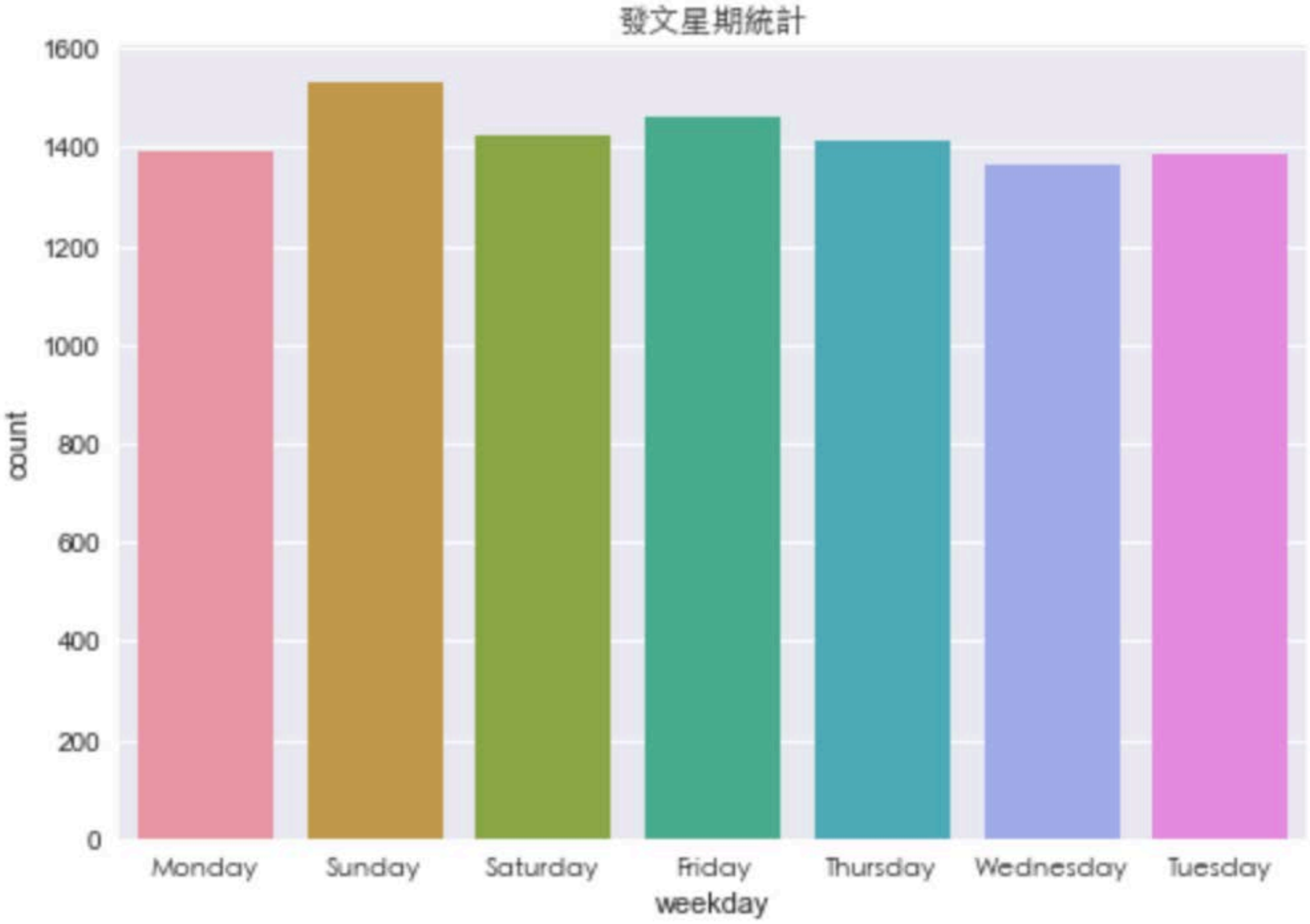
```
likes_avg_hour = []
for i in df['hour'].value_counts().index:
    ikes_avg_hour.append([i,(df[df['hour']==i])["likes_count"].mean()])
df4 = pd.DataFrame(likes_avg_hour, columns=['hour','mean_likes'])
sns.barplot(x='hour', y='mean_likes', data=df4)
plt.xticks(fontproperties=font, size=10)
plt.title("各時段發文按讚成效", fontproperties=font, size=12)
plt.show()
```


各時段發文按讚成效
mean(mean_likes)
8000
7000
6000
5000
4000
3000
2000
1000
0
0 1 2 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
hour

描述性統計

- 探討時間因子
  - 查看發文星期統計

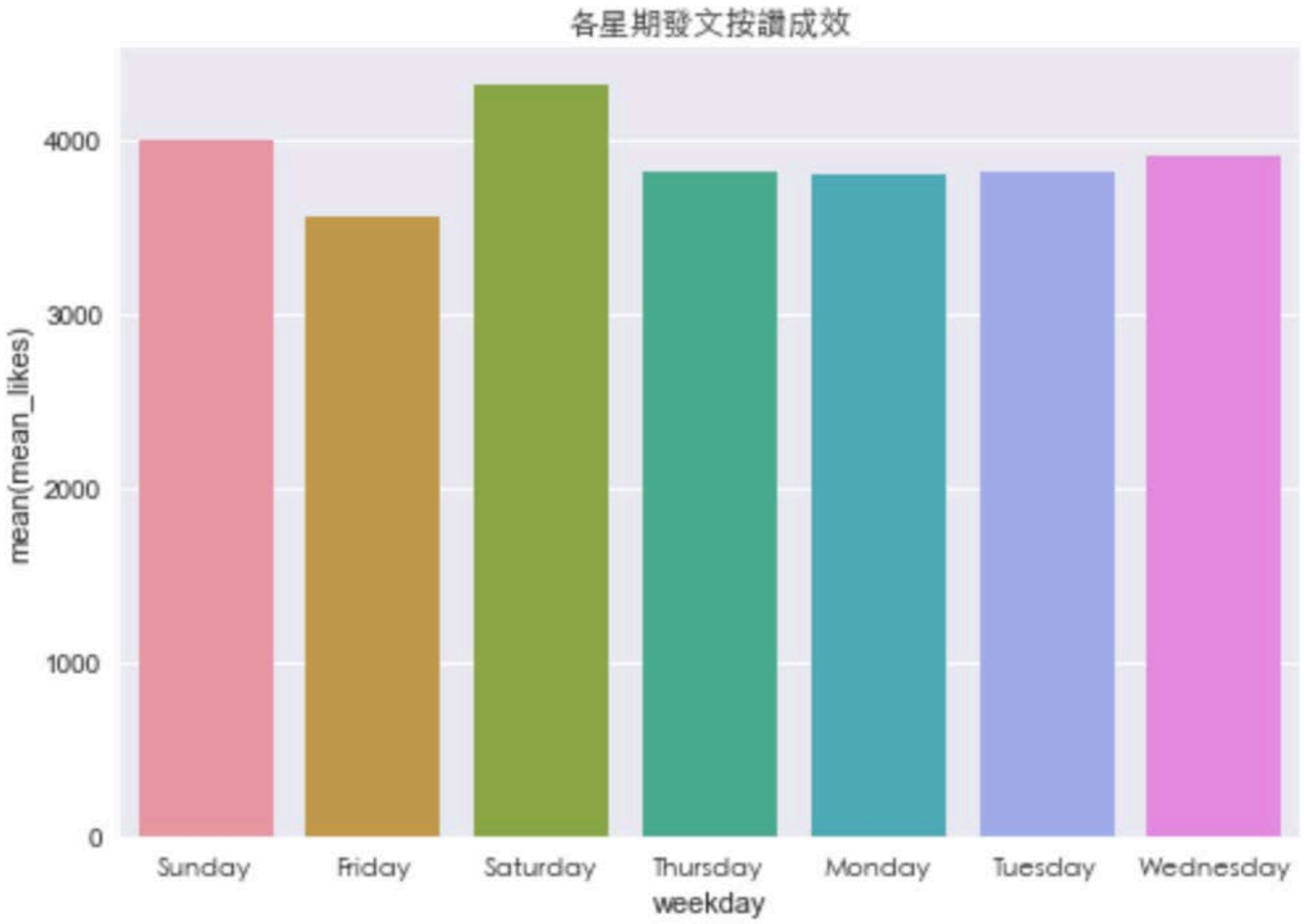
```
sns.countplot(data=df, x='weekday')
plt.xticks(fontproperties=font, size=10)
plt.title("發文星期統計", fontproperties=font, size=12)
plt.show()
```


發文星期統計
1600
1400
1200
1000
800
600
400
200
0
count
Monday
Sunday
Saturday
Friday
Thursday
Wednesday
Tuesday
weekday

描述性統計

- 探討時間因子
  - 查看各個星期發文按讚成效

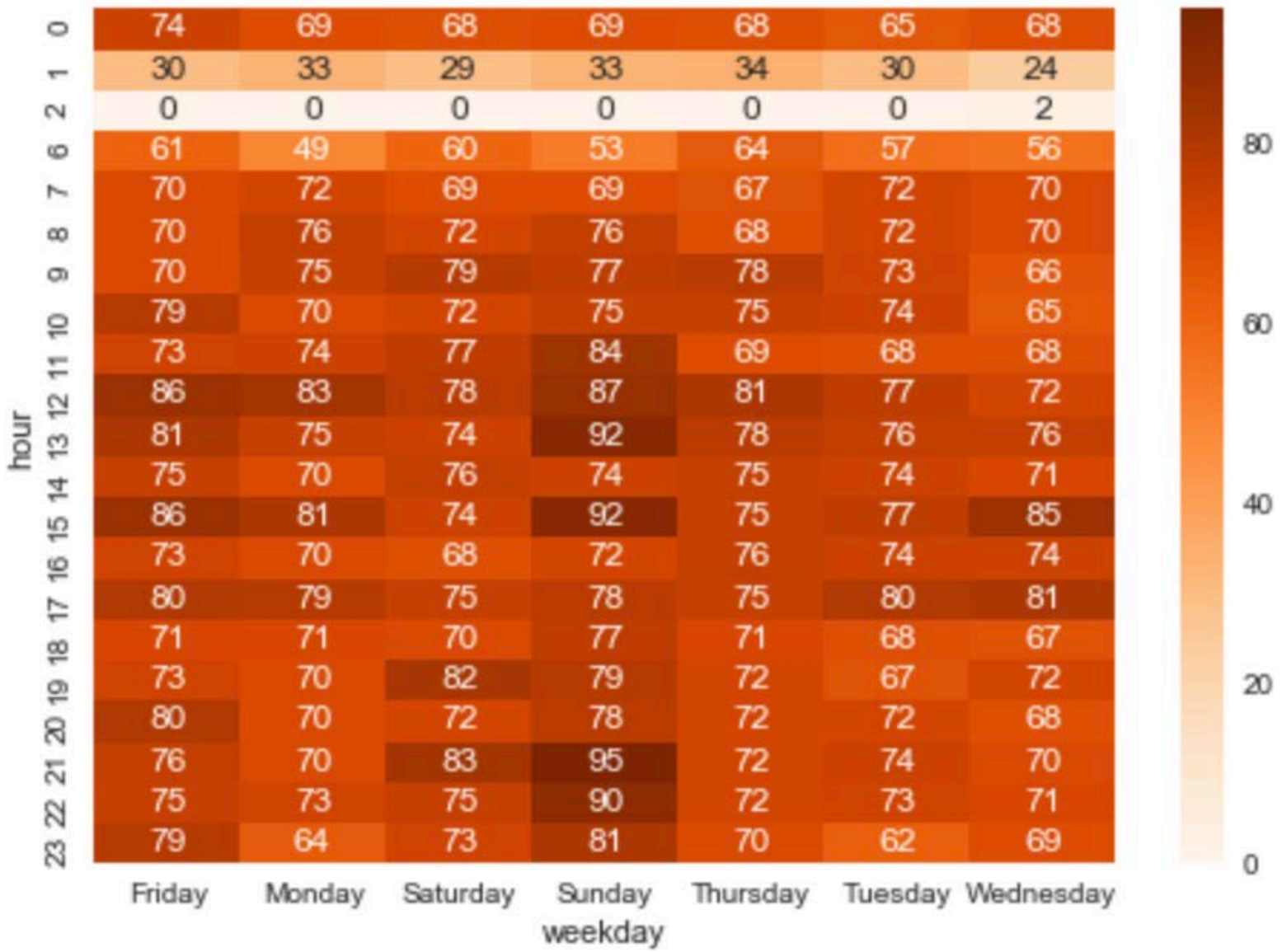
```
likes_avg_week = []
for i in df['weekday'].value_counts().index:
    likes_avg_week.append([i,(df[df['weekday']==i])["likes_count"].mean()])
df4 = pd.DataFrame(likes_avg_week, columns=['weekday','mean_likes'])
sns.barplot(x='weekday', y='mean_likes', data=df4)
plt.xticks(fontproperties=font,size=10)
plt.title("各星期發文按讚成效",fontproperties=font,size=12)
plt.show()
```


各星期發文按讚成效
mean(mean_likes)
4000
3000
2000
1000
0
Sunday
Friday
Saturday
Thursday
Monday
Tuesday
Wednesday
weekday

描述性統計

- 熱點圖 Heatmap
  - 各個星期 vs 各個時間點的發文數量

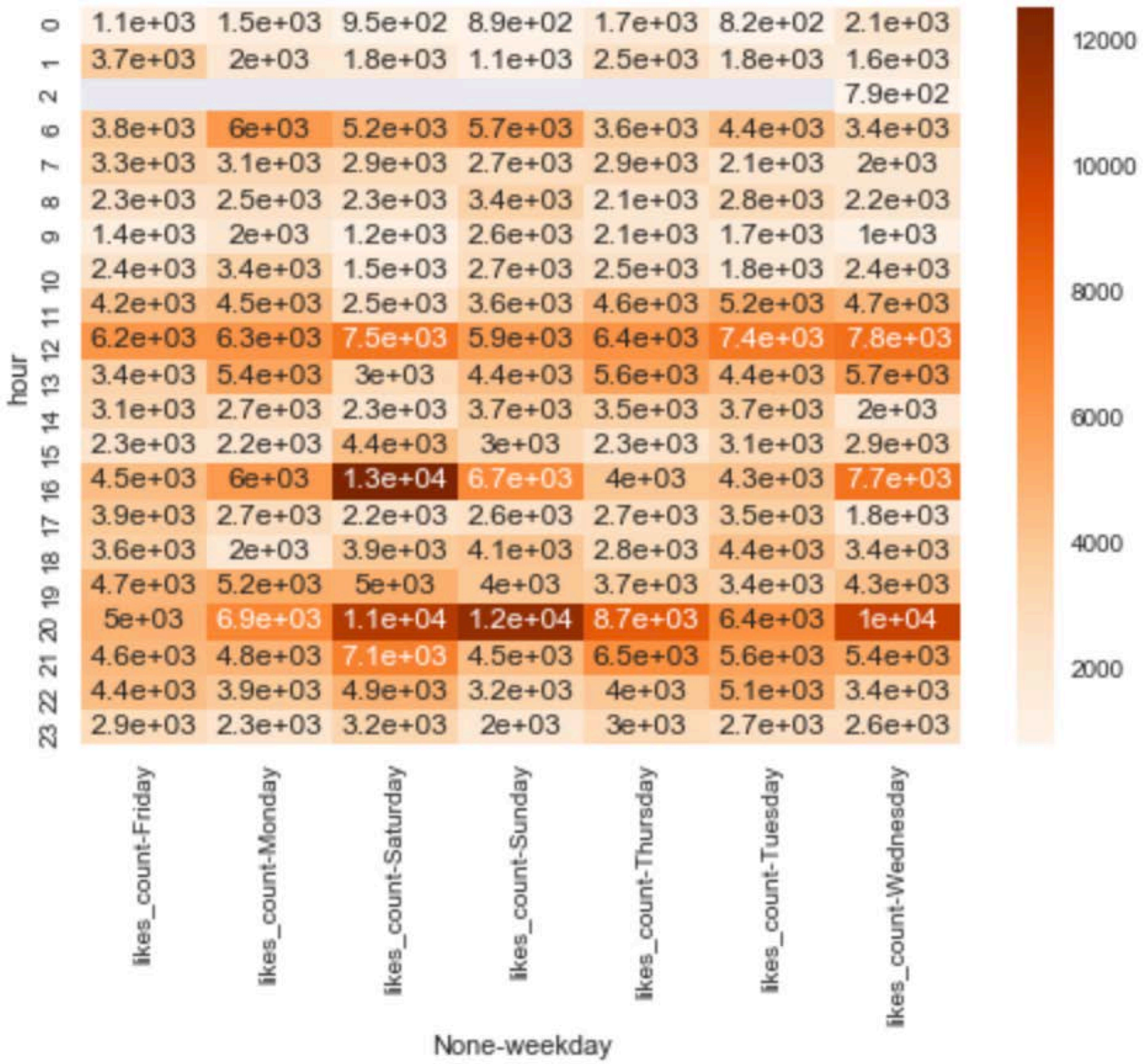
```
df6 = pd.crosstab(df["hour"], df["weekday"])
sns.heatmap(df6, annot=True,cmap="Oranges")
plt.show()
```


hour
0 1 2 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
Friday Monday Saturday Sunday Thursday Tuesday Wednesday
weekday
74 69 68 69 68 65 68
30 33 29 33 34 30 24
0 0 0 0 0 0 2
61 49 60 53 64 57 56
70 72 69 69 67 72 70
70 76 72 76 68 72 70
70 75 79 77 78 73 66
79 70 72 75 75 74 65
73 74 77 84 69 68 68
86 83 78 87 81 77 72
81 75 74 92 78 76 76
75 70 76 74 75 74 71
86 81 74 92 75 77 85
73 70 68 72 76 74 74
80 79 75 78 75 80 81
71 71 70 77 71 68 67
73 70 82 79 72 67 72
80 70 72 78 72 72 68
76 70 83 95 72 74 70
75 73 75 90 72 73 71
79 64 73 81 70 62 69
80
60
40
20
0

描述性統計

- 熱點圖 Heatmap
  - 各個星期 vs 各個時間點的按讚數量

```
df7 = pd.pivot_table(df, index=['hour'], columns=['weekday'],
                     values=['likes_count'])
sns.heatmap(df7, annot=True, cmap="Oranges")
plt.show()
```


hour
0 1.1e+03 1.5e+03 9.5e+02 8.9e+02 1.7e+03 8.2e+02 2.1e+03
1 3.7e+03 2e+03 1.8e+03 1.1e+03 2.5e+03 1.8e+03 1.6e+03
2 7.9e+02
6 3.8e+03 6e+03 5.2e+03 5.7e+03 3.6e+03 4.4e+03 3.4e+03
7 3.3e+03 3.1e+03 2.9e+03 2.7e+03 2.9e+03 2.1e+03 2e+03
8 2.3e+03 2.5e+03 2.3e+03 3.4e+03 2.1e+03 2.8e+03 2.2e+03
9 1.4e+03 2e+03 1.2e+03 2.6e+03 2.1e+03 1.7e+03 1e+03
10 2.4e+03 3.4e+03 1.5e+03 2.7e+03 2.5e+03 1.8e+03 2.4e+03
11 4.2e+03 4.5e+03 2.5e+03 3.6e+03 4.6e+03 5.2e+03 4.7e+03
12 6.2e+03 6.3e+03 7.5e+03 5.9e+03 6.4e+03 7.4e+03 7.8e+03
13 3.4e+03 5.4e+03 3e+03 4.4e+03 5.6e+03 4.4e+03 5.7e+03
14 3.1e+03 2.7e+03 2.3e+03 3.7e+03 3.5e+03 3.7e+03 2e+03
15 2.3e+03 2.2e+03 4.4e+03 3e+03 2.3e+03 3.1e+03 2.9e+03
16 4.5e+03 6e+03 1.3e+04 6.7e+03 4e+03 4.3e+03 7.7e+03
17 3.9e+03 2.7e+03 2.2e+03 2.6e+03 2.7e+03 3.5e+03 1.8e+03
18 3.6e+03 2e+03 3.9e+03 4.1e+03 2.8e+03 4.4e+03 3.4e+03
19 4.7e+03 5.2e+03 5e+03 4e+03 3.7e+03 3.4e+03 4.3e+03
20 5e+03 6.9e+03 1.1e+04 1.2e+04 8.7e+03 6.4e+03 1e+04
21 4.6e+03 4.8e+03 7.1e+03 4.5e+03 6.5e+03 5.6e+03 5.4e+03
22 4.4e+03 3.9e+03 4.9e+03 3.2e+03 4e+03 5.1e+03 3.4e+03
23 2.9e+03 2.3e+03 3.2e+03 2e+03 3e+03 2.7e+03 2.6e+03
likes_count-Friday
likes_count-Monday
likes_count-Saturday
likes_count-Sunday
likes_count-Thursday
likes_count-Tuesday
likes_count-Wednesday
None-weekday
12000
10000
8000
6000
4000
2000

描述性統計

- 匯出檔案
  - 最後匯出經過整理好的資料

```
df.to_excel("fanpage_clean.xlsx", index=False)
```

資料工程：文字探勘Text mining

推動大學程式設計教學計畫。分項六：資料分析領域教學研發推廣團隊（東吳大學巨量資料管理學院胡筱薇老師主編）

PART 01
什麼是文字探勘

什麼是文字探勘

自然語言處理 (NLP)

- Natural Language Process
- 讓機器懂我們的語言
- Ex. 機器翻譯、輸入法選字等

什麼是文字探勘

- 範疇其實很廣.....

1. 文本朗讀（Text to speech）/語音合成（Speech synthesis）
2. 語音識別（Speech recognition）
3. 自動分詞（word segmentation）
4. 詞性標註（Part-of-speech tagging）
5. 句法分析（Parsing）
6. 自然語言生成（Natural language generation）
7. 文本分類（Text categorization）
8. 信息檢索（Information retrieval）
9. 信息抽取（Information extraction）
10. 文字校對（Text-proofing）
11. 問答系統（Question answering）
12. 機器翻譯（Machine translation）
13. 自動摘要（Automatic summarization）
14. 文字蘊涵（Textual entailment）

什麼是文字探勘

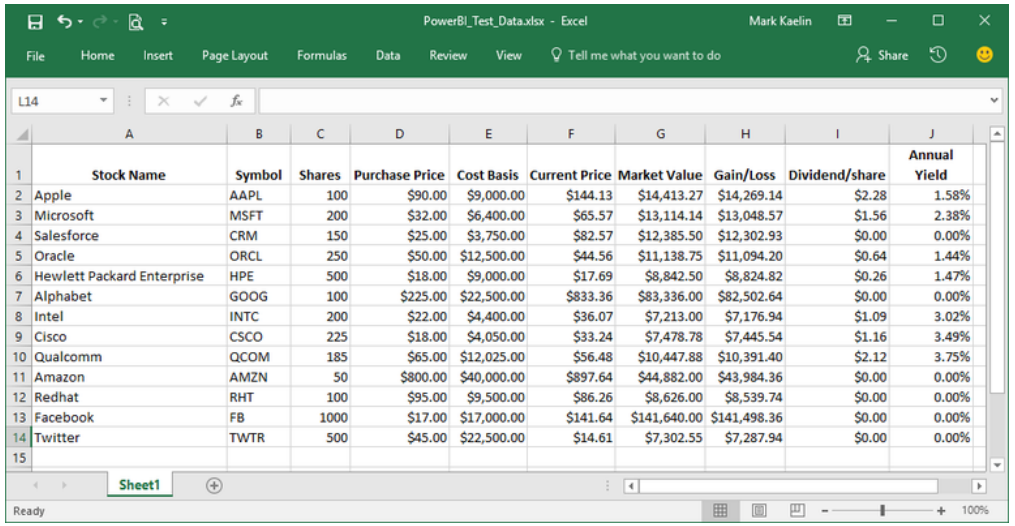
文字探勘 Text mining

- 也被稱為文本挖掘、文字採礦、智慧型分析等
- 從非結構化的文字資訊中，萃取出重要的資訊或知識
- 從大量的文章中找出隱含的寶藏

什麼是文字探勘

結構化 vs 非結構化

- 結構化資料
  --> Excel, csv, json 檔...



	Stock Name	Symbol	Shares	Purchase Price	Cost Basis	Current Price	Market Value	Gain/Loss	Dividend/share	Annual Yield
2	Apple	AAPL	100	$90.00	$9,000.00	$144.13	$14,413.27	$14,269.14	$2.28	1.58%
3	Microsoft	MSFT	200	$32.00	$6,400.00	$65.57	$13,114.14	$13,048.57	$1.56	2.38%
4	Salesforce	CRM	150	$25.00	$3,750.00	$82.57	$12,385.50	$12,302.93	$0.00	0.00%
5	Oracle	ORCL	250	$50.00	$12,500.00	$44.56	$11,138.75	$11,094.20	$0.64	1.44%
6	Hewlett Packard Enterprise	HPE	500	$18.00	$9,000.00	$17.69	$8,842.50	$8,824.82	$0.26	1.47%
7	Alphabet	GOOG	100	$225.00	$22,500.00	$833.36	$83,336.00	$82,502.64	$0.00	0.00%
8	Intel	INTC	200	$22.00	$4,400.00	$36.07	$7,213.00	$7,176.94	$1.09	3.02%
9	Cisco	CSCO	225	$18.00	$4,050.00	$33.24	$7,478.78	$7,445.54	$1.16	3.49%
10	Qualcomm	QCOM	185	$65.00	$12,025.00	$56.48	$10,447.88	$10,391.40	$2.12	3.75%
11	Amazon	AMZN	50	$800.00	$40,000.00	$897.64	$44,882.00	$43,984.36	$0.00	0.00%
12	Redhat	RHT	100	$95.00	$9,500.00	$86.26	$8,626.00	$8,539.74	$0.00	0.00%
13	Facebook	FB	1000	$17.00	$17,000.00	$141.64	$141,640.00	$141,498.36	$0.00	0.00%
14	Twitter	TWTR	500	$45.00	$22,500.00	$14.61	$7,302.55	$7,287.94	$0.00	0.00%

- 非結構化資料
  --> 文字檔，圖片檔，影音檔...

PART 02
文字探勘進行的方式

文字探勘進行的方式

Step 1

處理文字前一定要先能分辨一個句子 or 文章中的所有單字

- 以英文為例

  單字與單字中間有空格分開，容易分辨

  Ex. I am a boy and she is a girl.

文字探勘進行的方式

Step 1

處理文字前一定要先能分辨一個句子 or 文章中的所有單字

- 那中文的處理.....?

Ex. 我來自台灣東吳大學

我 / 來自 / 台灣 / 東吳大學

文字探勘進行的方式

Step 1

處理文字前一定要先能分辨一個句子 or 文章中的所有單字

- 斷字斷詞字典
- 字典裡有單字、權重、詞性
- 權重越高代表優先切開的字詞

dict.txt.big.txt

```
製造場 3 n
製造家 26 n
製造局 3 n
製造廠 194 n
製造廠商 3 n
製造懸念 3 n
製造業 847 n
製造業者 3 n
製造機 2 n
製造矛盾 3 l
製造糾紛 3 z
製造者 71 n
製造術 3 n
製造費用 3 n
```

文字探勘進行的方式

Step 1

處理文字前一定要先能分辨一個句子 or 文章中的所有單字

- Ex. XX是開發中國家

  1. XX / 是 / 開發中國家
  2. XX / 是 / 開發中 / 國家
  3. XX / 是 / 開發 / 中國 / 家

dict.txt.big.txt

```
製造場 3 n
製造家 26 n
製造局 3 n
製造廠 194 n
製造廠商 3 n
製造懸念 3 n
製造業 847 n
製造業者 3 n
製造機 2 n
製造矛盾 3 l
製造糾紛 3 z
製造者 71 n
製造術 3 n
製造費用 3 n
```

文字探勘進行的方式

Step 2

去除停止詞 stopwords

- 停止詞 (沒有意義的字)

  Ex. 我，你，它，當，的

- 以 "這堂課是文字探勘"為例

  這堂 / 課 / 文字探勘

停止詞字典

stop_words2.txt

可
可以
可是
可能
可見
各
各個
各人
各位
各地
各種
各級
各自
合理

文字探勘進行的方式

下載字典

- 斷字斷詞字典

  https://github.com/fxsjy/jieba/raw/master/extra_dict/dict.txt.big

- 停止詞字典

  https://github.com/chdd/weibo/blob/master/stopwords/中文停用詞庫.txt

PART 03
文字探勘套件介紹

文字探勘套件介紹

Python套件

- pip install jieba
- pip install nltk

```
import jieba
import nltk
```

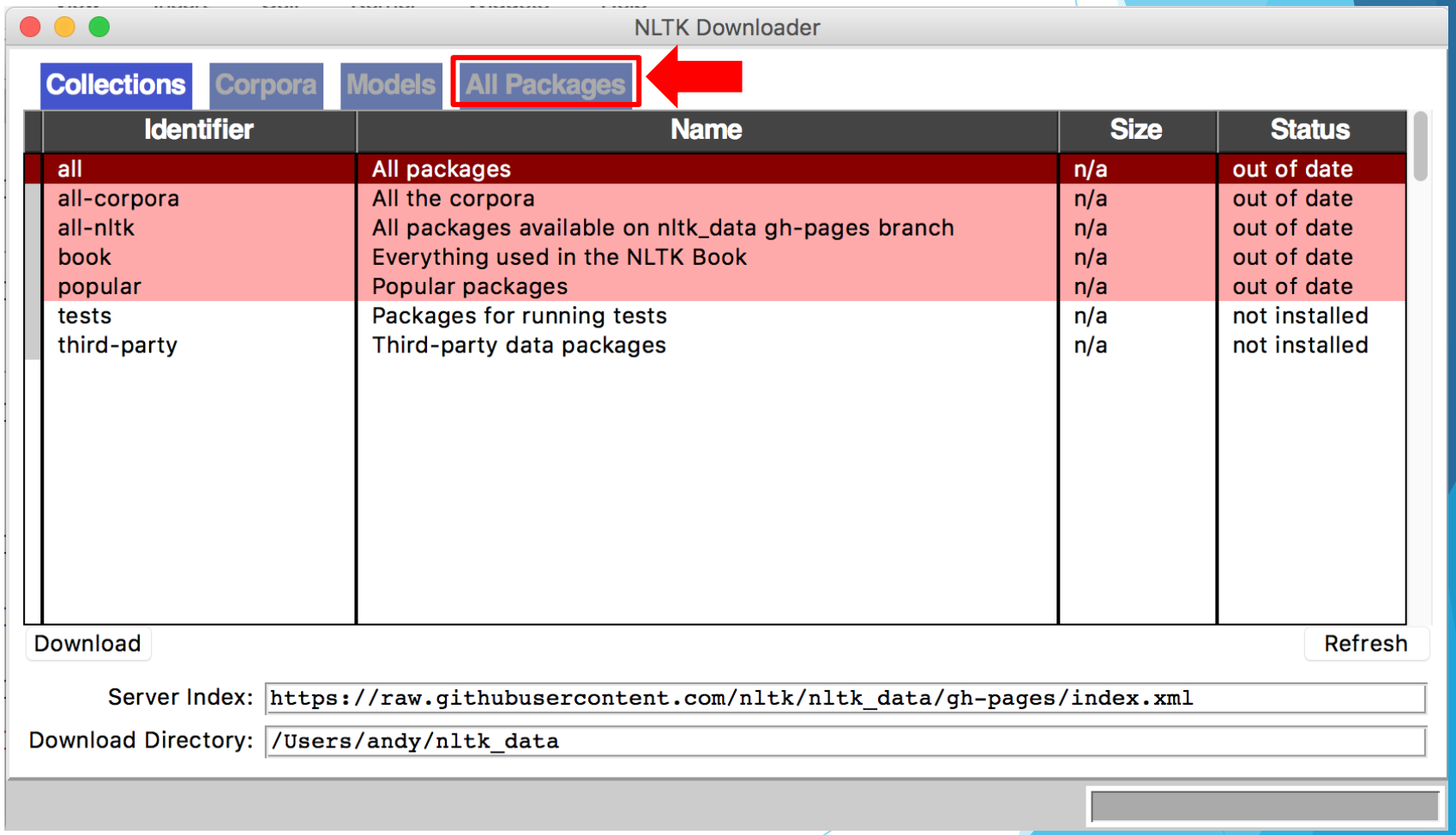
- nltk 是 Python 非常大的自然語言處理套件庫
- 包含的套件非常多，因此需個別下載

```
nltk.download()
```

文字探勘套件介紹

Python套件

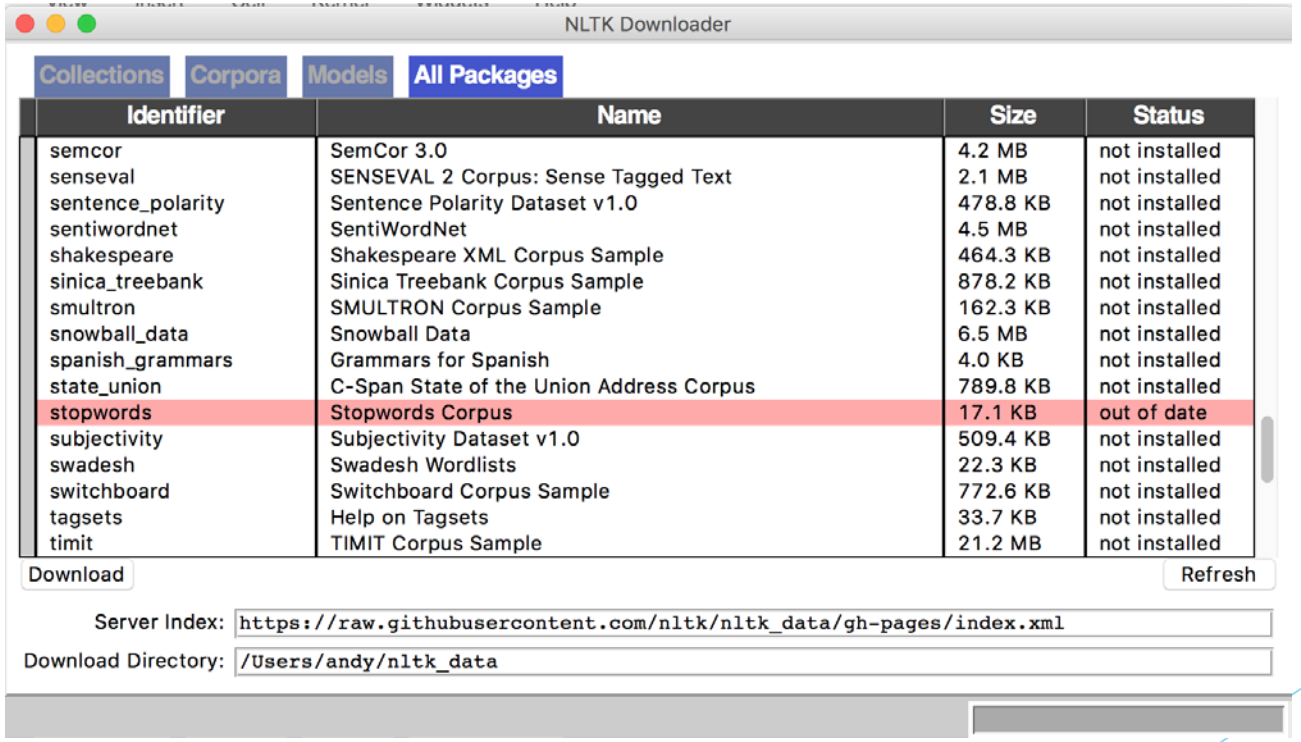
- 跳出新的下載視窗
- 點選All Packages



文字探勘套件介紹

Python套件

- 往下滑找到stopwords，點選左下角Downloads即下載完成



PART 04

Jieba套件介紹

Jieba套件介紹

匯入套件

- 同時載入兩份下載好的字典

```
import jieba
from nltk.corpus import stopwords
jieba.set_dictionary('dict.txt.big.txt')
stop=stopwords.words('stop_words2.txt')
```

Jieba套件介紹

Jieba模式

- 1. 全模式：會把所有認為可能是單字的詞都切出來

```
import jieba
word = jieba.cut("我來到台灣台北東吳大學", cut_all=True)
print("/".join(word))
```

我/來到/台/灣/台北/北東/東吳/東吳大學/大學

Jieba套件介紹

Jieba模式

- 2. 精確模式：只會切出最有可能在這個句子中的單字

```
word = jieba.cut("我來到台灣台北東吳大學", cut_all=False)
print("/".join(word))
```

我/來到/台灣/台北/東吳大學

PS：若無設定 cut_all 參數，默認為精確模式

Jieba套件介紹

Jieba模式

- 3. 搜尋模式：切出有可能會被搜尋的字詞

```
word = jieba.cut_for_search("我來到台灣台北東吳大學")
print("/".join(word))
```

我/來到/台灣/台北/東吳/大學/東吳大學

Jieba套件介紹

- 到網路上找一篇長篇文章，使用精確模式看切出來的字詞是否

```
word = jieba.cut("""目前暫居龍頭的中信兄弟今天在主場迎戰富邦悍將，1局下黃衫軍精神領袖彭政閔就擊出3分砲，助隊在開賽就攻下3比0領先。
8月6日是恰恰的40歲生日，在年紀越來越大的情況下，近年不斷傳出他可能從球員身分退役的消息，雖已接近不惑之年，彭政閔本季目前為止打擊率仍有3成76，
彭政閔本季至今天賽前229打數擊出72支安打包含4發全壘打、貢獻36打點、打擊率3成76，上回恰恰開轟是6月14日面對富邦悍將投手張耿豪所擊出。今天同樣是
(中時電子報)""")
print("/".join(word))
```

```
目前/暫居/龍頭/的/中信/兄弟/今天/在/主場/迎戰/富邦/悍將/，/1/局下/黃/衫/軍/精神領袖/彭政閔/就/擊出/3/分/砲/，/助隊/在/開賽/就/攻下/3/
比/0/領先/。/
/8/月/6/日/是/恰恰/的/40/歲/生日/，/在/年紀/越來越/大/的/情況/下/，/近年/不斷/傳出/他/可能/從/球員/身分/退役/的/消息/，/雖/已/接近/
不惑之年/，/彭政閔/本季/目前/為止/打擊率/仍/有/3/成/76/，/還看/得到/年輕/時期/「/4/割/男/」/的/影子/。/
/彭政閔/本季/至/今天/賽前/229/打數/擊出/72/支安打/包含/4/發/全壘打/、/貢獻/36/打點/、/打擊率/3/成/76/，/上/回/恰恰/開轟是/6/月/14/
日/面對/富邦/悍將/投手/張耿豪/所/擊出/。/今天/同樣/是/面對/富邦/，/苦主/則/換成/前/隊友/伍鐸/(/Bryan/ /Woodall/)/。/
/(/中時/電子報/)
```

- 發現有些字切的沒有很正確

```
目前/暫居/龍頭/的/中信/兄弟/今天/在/主場/迎戰/富邦/悍將/，/1/局下/黃/衫/軍/精神領袖/彭政閔/就/擊出/3/分/砲/，/助隊/在/開賽/就/攻下/3/
比/0/領先/。/
```

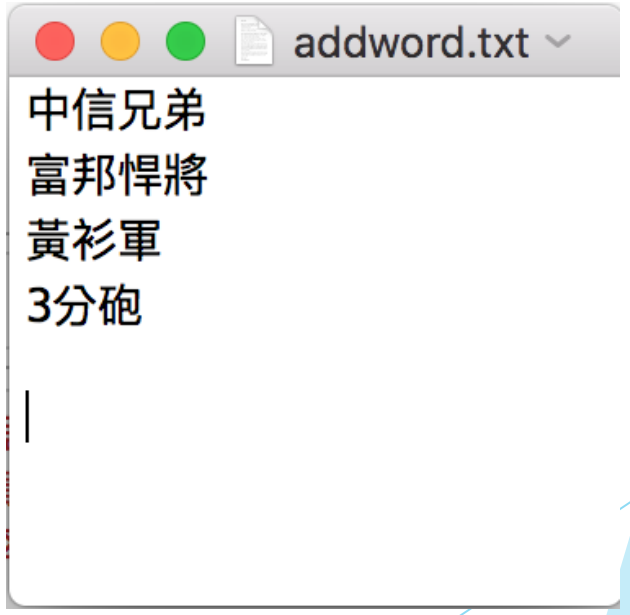
Jieba套件介紹

Jieba功能　加入自定義字典

1. jieba除了有內建辭庫外，也可以針對沒有在字典的詞自行加入

2. 不同的專業會有不同的語料庫

3. 打開記事本，輸入沒切好的字



Jieba套件介紹

4. 載入自定義的字典

```
jieba.load_userdict("addword.txt")
```

5. 重跑一次剛剛的 jieba 可以發現正確切開了加入的字詞

```
word = jieba.cut("""目前暫居龍頭的中信兄弟今天在主場迎戰富邦悍將，1局下黃衫軍精神領袖彭政閔就擊出3分砲，助隊在開賽就攻下3比0領先。
8月6日是恰恰的40歲生日，在年紀越來越大的情況下，近年不斷傳出他可能從球員身分退役的消息，雖已接近不惑之年，彭政閔本季目前為止打擊率仍有3成76，
彭政閔本季至今天賽前229打數擊出72支安打包含4發全壘打、貢獻36打點、打擊率3成76，上回恰恰開轟是6月14日面對富邦悍將投手張耿豪所擊出。今天同樣是
(中時電子報)""")
print("/".join(word))
```

```
目前/暫居/龍頭/的/中信兄弟/今天/在/主場/迎戰/富邦悍將/，/1/局下/黃衫軍/精神領袖/彭政閔/就/擊出/3分砲/，/助隊/在/開賽/就/攻下/3/比/0/領
先/。/
/8/月/6/日/是/恰恰/的/40/歲/生日/，/在/年紀/越來越/大/的/情況/下/，/近年/不斷/傳出/他/可能/從/球員/身分/退役/的/消息/，/雖/已/接近/
不惑之年/，/彭政閔/本季/目前/為止/打擊率/仍/有/3/成/76/，/還看/得到/年輕/時期/「/4/割/男/」/的/影子/。/
/彭政閔/本季/至/今天/賽前/229/打數/擊出/72/支安打/包含/4/發/全壘打/、/貢獻/36/打點/、/打擊率/3/成/76/，/上/回/恰恰/開轟是/6/月/14/
日/面對/富邦悍將/投手/張耿豪/所/擊出/。/今天/同樣/是/面對/富邦/，/苦主/則/換成/前/隊友/伍鐸/(/Bryan/ /Woodall/)/。/
/(/中時/電子報/)
```

Jieba套件介紹

Jieba功能　去除停止詞

1. 去除前

```
word = jieba.cut("目前暫居龍頭的中信兄弟今天在主場迎戰富邦悍將")
print("/".join(word))
```

目前/暫居/龍頭/的/中信兄弟/今天/在/主場/迎戰/富邦悍將

2. 去除後

```
word = jieba.cut("目前暫居龍頭的中信兄弟今天在主場迎戰富邦悍將")
stoptext = ''
for words in word:
    if words not in stop:
        stoptext += "/"+words
print(stoptext)
```

/暫居/龍頭/中信兄弟/主場/迎戰/富邦悍將

PART 05
分析粉絲專頁資料

分析粉絲專頁資料

匯入檔案

```
Import pandas as pd
df = pd.read_excel('fanpage_clean.xlsx')
df.head()
```

	id	message
0	124616330906800_1560501197318299	阿娘威！披羊皮的狼？竟大口嚼小雞 #要打統編：小編真的是快嚇死了...😬😬 影片來源：騰訊視頻 #草食性 #羊
1	124616330906800_1560454417322977	被黑了！李毓芬演唱「大落拍」網友卻意外發現「亮點」 #條紋編：這一段應該是昨天的亮點表演之一吧～ #李毓芬 #落拍 #唱歌
2	124616330906800_1559870414048044	誰說牠呆？心機月月調虎離山 網友讚影帝 #樂無編：最萌心機鬼～(*ˆ∀ˆ)~❤ 影片來源：秒拍 #萌 #心機 #調虎離山計

分析粉絲專頁資料

- 將匯入的資料做斷字斷詞

```
jieba_text = []
for index in range(len(df)):
    words = jieba.cut(str(df['context'][index]))
    text, text2 = [], ''
    for word in words:
        if word not in stop:
            text.append(word)
            text2 += " "+word
    jieba_text.append([text,text2,len(text)])
```

100% 9975/9975 [00:20<00:00, 498.10it/s]

分析粉絲專頁資料

結果呈現

```
df['jieba_text'] = pd.DataFrame(jieba_text)[0]
df['jieba_text2'] = pd.DataFrame(jieba_text)[1]
df['jieba_count'] = pd.DataFrame(jieba_text)[2]
```

	id	message	**jieba_text**	**jieba_count**	**jieba_text2**
0	124616330906800_1560501197318299	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	[阿娘, 威, 披, 羊皮, 狼, 竟大口, 嚼, 小雞, \n, #, 統編, 小編, 真...	30	阿娘 威 披 羊皮 狼 竟大口 嚼 小雞 \n # 統編 小編 真 的 快 嚇死 ... 😬...
1	124616330906800_1560454417322977	被黑了！李毓芬演唱「大落拍」網友卻意外發現「亮點」\n#條紋編：這一段應該是昨天的亮點表演...	[黑, 李毓芬, 演唱, 「, 大落, 拍, 」, , 網友, 卻, 意外, 發現, 「,...	37	黑 李毓芬 演唱 「 大落 拍 」 網友 卻 意外 發現 「 亮點 」 \n # 條紋...
2	124616330906800_1559870414048044	誰說牠呆？心機月月調虎離山 網友讚影帝\n#樂無編：最萌心機鬼～(*´∀`)~❤\n\n影片...	[誰, 說, 牠, 呆, 心機, 月, 月, 調虎離山, , 網友, 讚, 影帝, \n,...	43	誰 說 牠 呆 心機 月 月 調虎離山 網友 讚 影帝 \n # 樂無編 最萌 心機 ...

PART 06

TF-IDF 演算法

TF-IDF 演算法

TF-IDF

Term x within document y

TF-IDF

- TF-IDF 其實是TF和IDF的結合
- 公式看起來很複雜

$$w_{t,d} = (1 + \log(\mathrm{tf}_{t,d})) \times \log(N / \mathrm{df}_t)$$

TF-IDF 演算法

TF-IDF

Term x within document y

Term Frequency (TF)

- ✓ 計算一個字詞在一篇文章中所出現的次數 (count)，並取log降維

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d} & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- ✓ 但文字出現的越多不代表越重要，有可能是不重要的字詞 Ex.我，你，它……

TF-IDF 演算法

TF-IDF

Term x within document y

Inverse Document Frequency (IDF)

- ✓ 算法是 [(總文件數) / 文字出現在文件內的次數]取log
- ✓ 把上面不重要的字詞降低權重

$$\mathrm{idf}_t = \log_{10} \frac{1{,}000{,}000}{\mathrm{df}_t}$$

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

TF-IDF 演算法

TF-IDF

Term x within document y

- ✓ 所以TF-IDF就是將兩個相乘
  (其中一種算法，其實還有很多衍伸型)公式看起來很複雜)

$$w_{t,d} = (1+\log(tf_{t,d}))\times\log(N/df_t)$$

- ✓ 看不懂公式……看圖

Word	出現總次數	出現文件數
ferrari	10422	17←較高的稀有性 (高資訊量)
insurance	10440	3997

TF-IDF 演算法

安裝python套件

- ✓ pip install scikit-learn
- ✓ 匯入TF-IDF模組

```
from sklearn import feature_extraction
from sklearn.feature_extraction.text import TfidfVectorizer
```

- ✓ 將文字轉換成矩陣模式

```
vectorizer = TfidfVectorizer()
tfidf = vectorizer.fit_transform(df['jieba_text2'])
```

TF-IDF 演算法

✓ 計算TFIDF，每個文章中的前五個關鍵字匯入TF-IDF模組

```
words = []
words2 = vectorizer.get_feature_names()
for i in tqdm_notebook(range(len(df['jieba_text2']))):
    print("======= Post"+str(i+1)+" =======")
    temp_array = tfidf[i,:].toarray()
    for l in temp_array:
        print([(words2[x],l[x]) for x in (l*-1).argsort()][:5])
```

```
======= Post1 =======
[('草食性', 0.3886164663330653), ('竟大口', 0.3886164663330653), ('小雞',
0.37205588933099115), ('羊皮', 0.37205588933099115), ('阿娘', 0.31216568557942365)]
======= Post2 =======
[('李毓芬', 0.5393296772291432), ('亮點', 0.4742848799368196), ('大落',
0.28166788829793676), ('昨天', 0.2611485393335714), ('演唱', 0.2611485393335714)]
======= Post3 =======
[('心機', 0.6875944679814824), ('調虎離山', 0.5748856520127635), ('影帝',
0.25649296680540506), ('最萌', 0.22919815599382748), ('樂無編', 0.2181098701502329)]
```

TF-IDF 演算法

✓ 將結果存到變數裡面

```
words = []
words2 = vectorizer.get_feature_names()
for i in tqdm_notebook(range(len(df['jieba_text2']))):
    temp_array = tfidf[i,:].toarray()
    for l in temp_array:
        words.append([(words2[x],l[x]) for x in (l*-1).argsort()][:5])
```

TF-IDF 演算法

✓ 轉成 DataFrame 形式

```
# 每篇文章的前五關鍵詞
df2 =
pd.DataFrame(words,columns=['Keyword1','Keyword2','Keyword3','Keyword4','Keyword5'])
df2.head()
```

	Keyword1	Keyword2	Keyword3	Keyword4	Keyword5
0	(草食性, 0.41657326701385844)	(竟大口, 0.41657326701385844)	(小雞, 0.3988213335189045)	(羊皮, 0.3988213335189045)	(阿娘, 0.33462266979698757)
1	(李毓芬, 0.5487664937801285)	(亮點, 0.48258358774415294)	(大落, 0.2865963176100915)	(演唱, 0.26571793531213106)	(昨天, 0.26571793531213106)
2	(心機, 0.6939407964222015)	(調虎離山, 0.5801917057019975)	(影帝, 0.25886033403402775)	(最萌, 0.23131359880740054)	(樂無編, 0.2201229708027141)
3	(感動, 0.32316350526125726)	(2roeqto, 0.2504423351739234)	(our, 0.2504423351739234)	(陪孕妻, 0.2504423351739234)	(living, 0.2504423351739234)
4	(畫家, 0.546678780884177)	(樓亡, 0.28957812644546865)	(要當, 0.2804329445867179)	(挑食, 0.2804329445867179)	(死神, 0.2733393904420885)

TF-IDF 演算法

- ✓ 與原本的資料合併
- ✓ pandas合併有兩種方法 (concat & merge)
- ✓ https://pandas.pydata.org/pandas-docs/stable/merging.html

	id	message	jieba_message	Keyword1	Keyword2	Keyword3	
0	124616330906800_1560501197318299	阿娘威！披羊皮的狼？竟大口嚼小雞\n#要打統編：小編真的是快嚇死了...😬😬\n\n影片來源...	/阿娘/威/披/羊皮/狼/竟大口/嚼/小雞/\n/統編/小編/真的/嚇死/\n/\n/\n/...	(草食性, 0.41657326701385844)	(竟大口, 0.41657326701385844)	(羊皮, 0.3988213335189045)	(小雞, 0.3988213

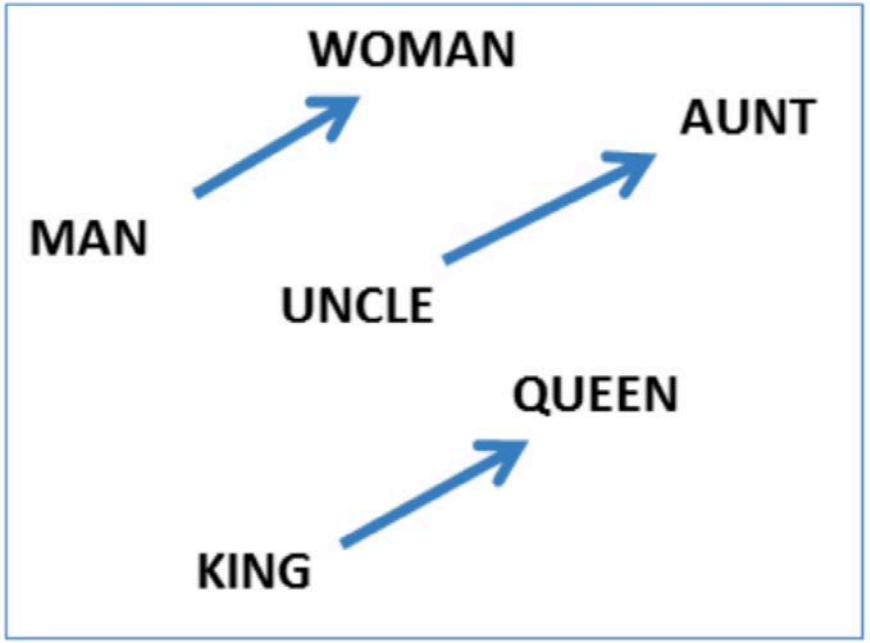
PART 07

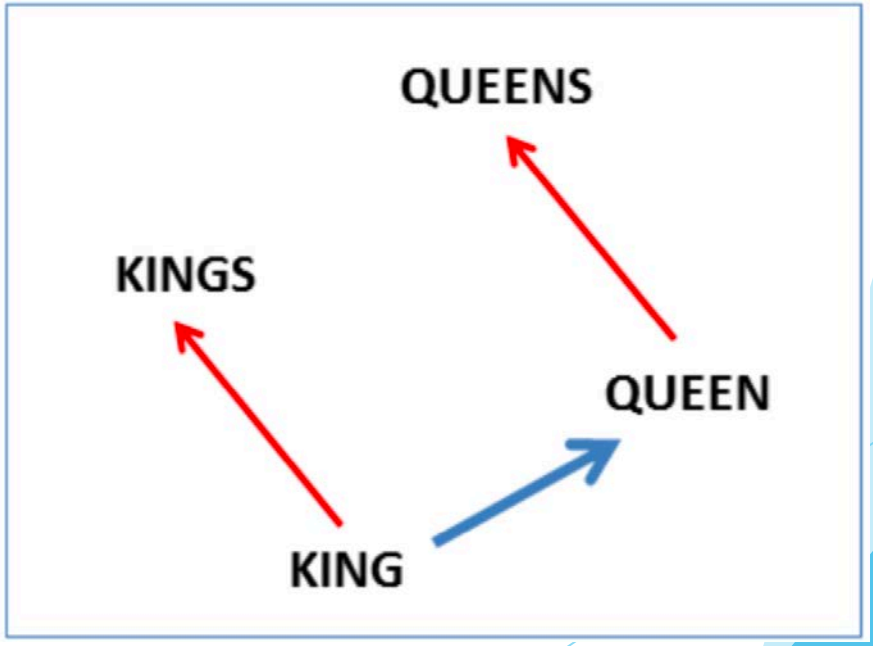
Word2Vec 演算法

Word2Vec 演算法

Word2Vec概念

✓ 把詞轉換成向量投射到高維度空間

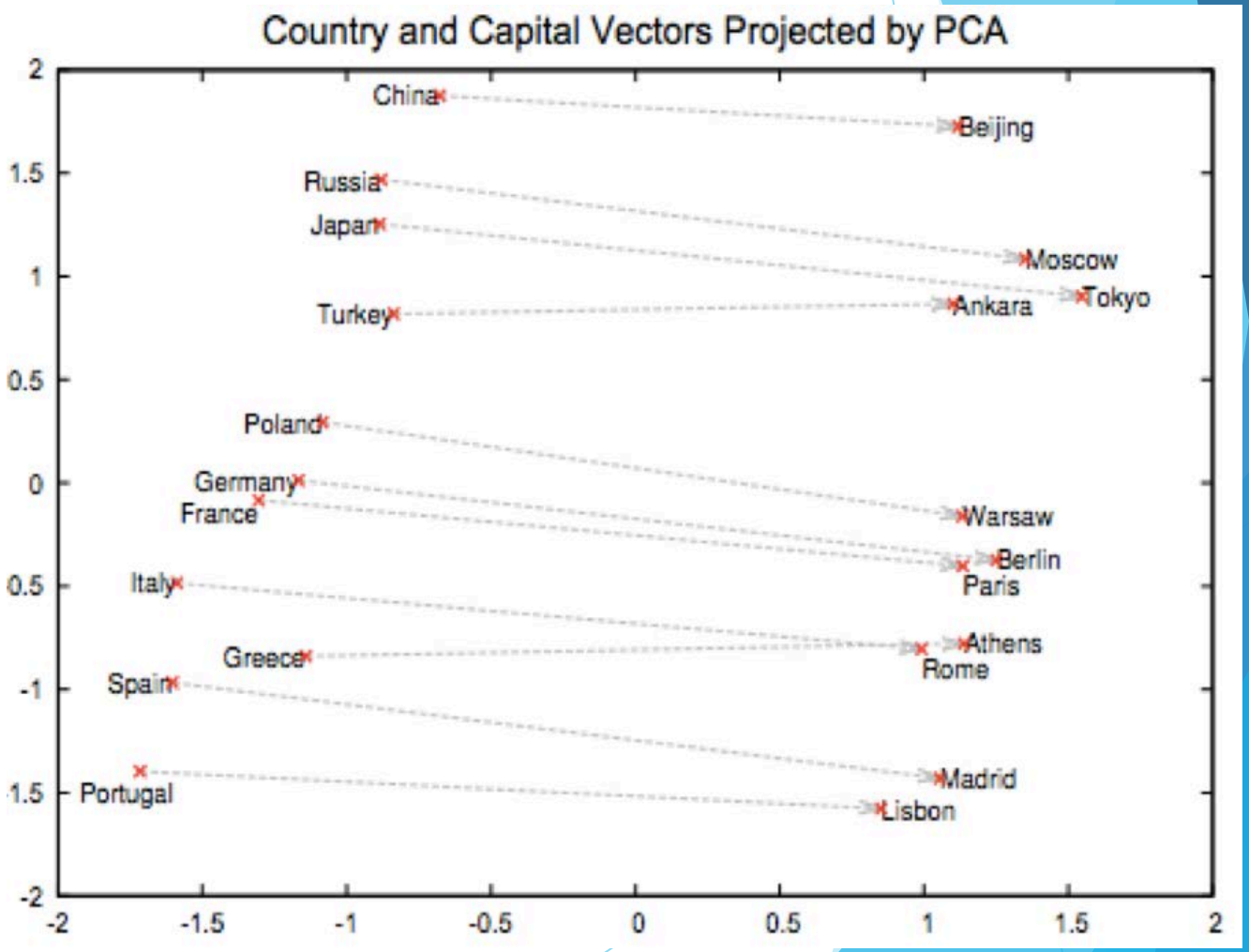




Word2Vec 演算法

Word2Vec概念

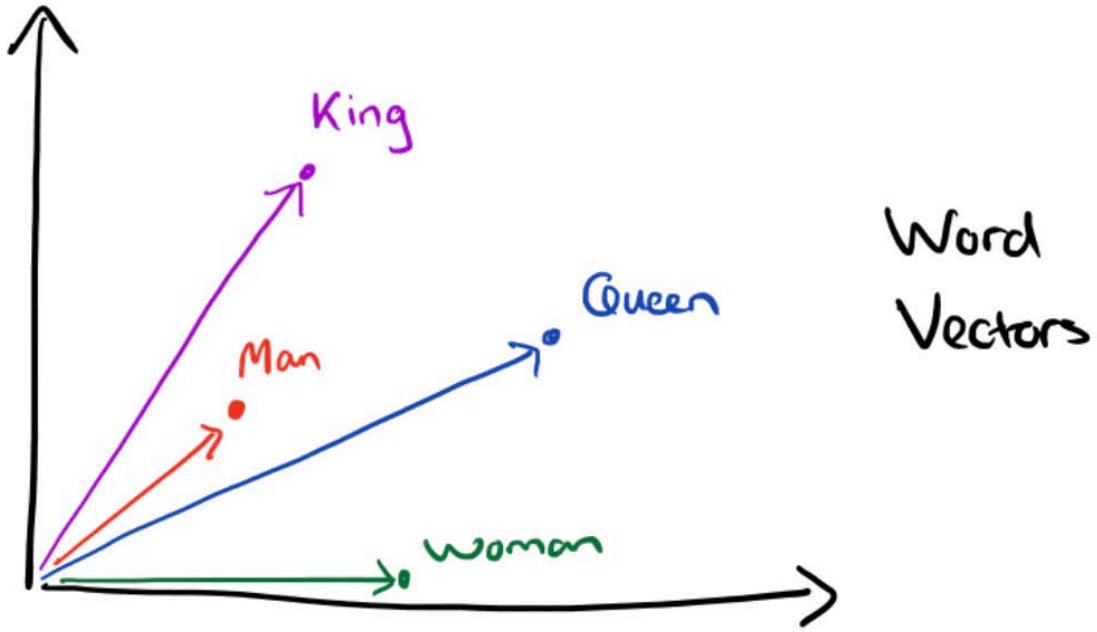
- ✓ 找尋字與字之間的關聯
- ✓ 相同概念的詞距離會相似



Word2Vec 演算法

Word2Vec概念

✓ 算出來的值會是向量間的cos值，也就是夾角角度



Word2Vec 演算法

安裝 Python 套件

- ✓ `pip install genism`
- ✓ 匯入Word2Vec模組

```
from gensim.models.word2vec import Word2Vec
```

Word2Vec 演算法

建立模型

- ✓ Word2Vec 吃的格式是 Dataframe 中的 list 型態

```
model = Word2Vec(df['jieba_text'])
```

```
def most_similar(w2v_model, words, topn=10):
    similar_df = pd.DataFrame()
    for word in words:
        try:
            similar_words = pd.DataFrame(w2v_model.wv.most_similar(word, topn=topn),
                                         columns=[word, 'cos'])
            similar_df = pd.concat([similar_df, similar_words], axis=1)
        except:
            print(word, "not found in Word2Vec model!")
    return similar_df
```

Word2Vec 演算法

查看結果

```
most_similar(model, ['新聞', '體育', '娛樂', '政治', 'XDD', '小編', '夜市', '明星', '女孩'])
```

體育 not found in Word2Vec model!

	新聞	cos	娛樂	cos	政治	cos	XDD	cos	小編	cos	夜市	cos	明星	cos	女孩	cos
0	直播	0.967324	S	0.998950	地方	0.994920	哩厝編	0.994200	好	0.984673	環島	0.999220	技巧	0.999714	噴發	0.998853
1	中心	0.965926	櫻花	0.998900	社會	0.994582	狗狗	0.991577	萌死	0.978975	飯店	0.999144	髮	0.999616	爸爸	0.998728
2	綜合	0.956663	起司	0.998894	T	0.990361	柯基	0.991063	曉	0.977118	高空	0.999091	最美	0.999566	Q	0.998684
3	育樂中心	0.943752	役	0.998769	NCC	0.990037	XDDD	0.990727	想	0.972329	餐廳	0.999069	安娜	0.999490	抱	0.998549
4	愛玩	0.932010	害怕	0.998734	壯壯	0.989895	誰准	0.990455	人	0.971244	團	0.999040	茶	0.999484	洋蔥	0.998500
5	吳念樺	0.928818	飛行	0.998684	禽流感	0.987717	好笑	0.989693	寶寶	0.970022	鍋貼	0.998937	V	0.999439	心	0.998497
6	朱姵慈	0.928699	手工	0.998654	肇逃	0.987193	愛	0.988566	吃	0.963738	驚悚	0.998870	倍	0.999332	床	0.998367
7	台	0.917264	抓到	0.998643	侵權	0.986938	養樂多	0.988533	真的	0.962447	巴黎	0.998865	父	0.999247	有趣	0.998241
8	提醒您	0.909181	性別	0.998492	毒品	0.986920	牠	0.988344	Jay	0.962223	跑車	0.998857	編髮	0.999242	瞬間	0.998178
9	李欣	0.907590	約會	0.998433	巨蛋	0.986402	Baby	0.988283	長	0.959666	板車	0.998834	馬尾	0.999203	挑戰	0.998119

Word2Vec 演算法

修改參數

```
model_d250 = Word2Vec(df['jieba_text'], size=250, iter=10)
most_similar(model_d250, ['新聞', '體育', '娛樂', '政治', 'XDD', '小編', '夜市', '明星',
'女孩'])
```

```
體育 not found in Word2Vec model!
```

	新聞	cos	娛樂	cos	政治	cos	XDD	cos	小編	cos	夜市	cos	明星	cos	女孩	cos
0	Nick	0.908591	Keigo	0.995358	監獄	0.978106	愛	0.963409	好	0.927528	逢甲	0.979087	原味	0.994482	掛在	0.968459
1	節目	0.908183	Youtube	0.991555	香港	0.972059	牠	0.961356	想	0.902362	溫泉	0.971208	跟風	0.994479	汪汪	0.964284
2	直播	0.898731	邦	0.990808	美國	0.972043	可愛	0.958527	感覺	0.901491	步道	0.968835	上身	0.994263	卡哇伊	0.956831
3	愛玩	0.891666	紙牌	0.990400	青瓦台	0.971166	xD	0.957824	真的	0.870075	牛肉	0.967882	出奇	0.994263	這招	0.956034
4	房趣	0.887777	<	0.989025	谷	0.967853	XDDDD	0.948542	人	0.869911	殺手	0.967501	出招	0.993517	主人	0.955768
5	調查	0.821492	媒合	0.988894	汙	0.967761	嘗試	0.943750	媽媽	0.849948	沖繩	0.966578	境界	0.993509	表情	0.955542
6	中心	0.813243	ι	0.988487	中部	0.966782	der	0.940438	厲害	0.846849	品牌	0.966445	完美	0.993505	挑戰	0.953859
7	挑	0.810430	Mihara	0.988282	政府	0.966702	嘴編	0.937272	耶	0.843780	駁火	0.966388	放閃	0.993127	小	0.953715
8	車	0.797249	咖哩	0.988211	跨界	0.964899	脆嘀叔	0.933207	好吃	0.841185	商圈	0.965920	材料	0.993045	隻	0.952203
9	馨	0.788445	錕	0.988182	車站	0.964738	萌死	0.930417	統編	0.840771	麻辣鍋	0.965484	電眼	0.992904	肚臍	0.951553

Word2Vec 演算法

修改參數

- ✓ size 代表詞向量大小
- ✓ iter 代表訓練的次數

其他參數

- ✓ 官方文件：https://radimrehurek.com/gensim/models/word2vec.html
- ✓ 中文參考：https://www.kaggle.com/jerrykuo7727/word2vec

Word2Vec 演算法

儲存模型

- ✓ 資料量很大時避免每次都要重複訓練模型

```
model.save('word2vec.model')
```

- ✓ 下次可以直接匯入模型，再做參數調整

```
from gensim.models.word2vec import Word2Vec
model = Word2Vec.load('word2vec.model')
```

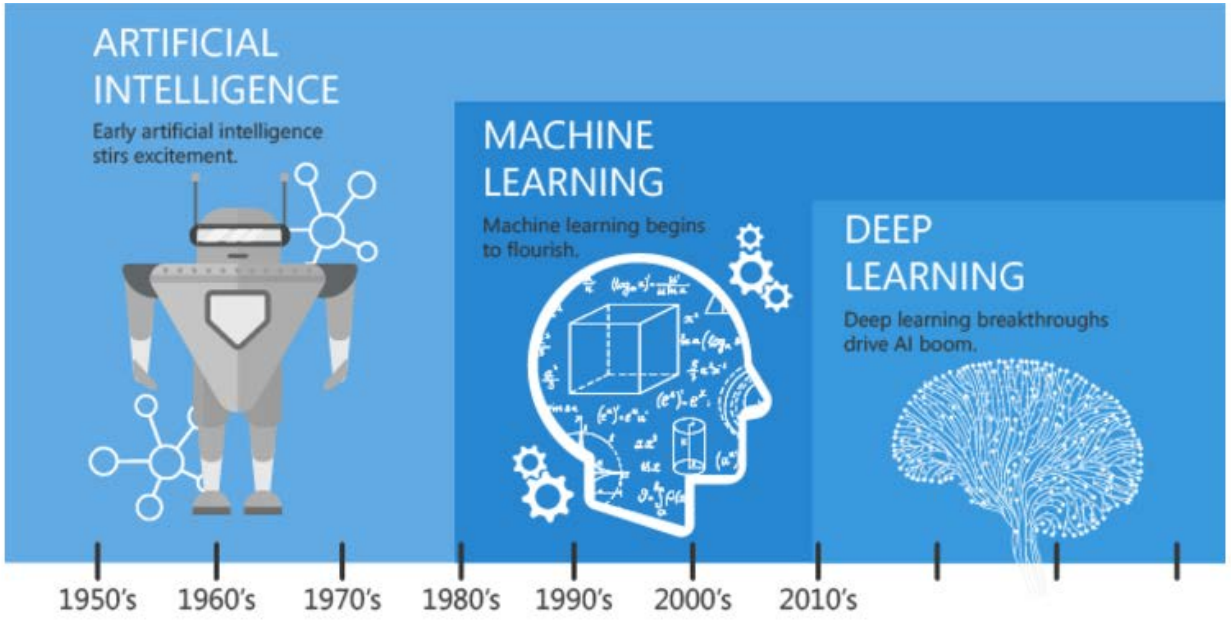
進階分析：資料建模Data modeling

PART 01
什麼是機器學習

什麼是機器學習

用一張圖說明

- 可以發現早在1950年代就出現AI的想法
- 而陸續出現機器學習、深度學習等技術

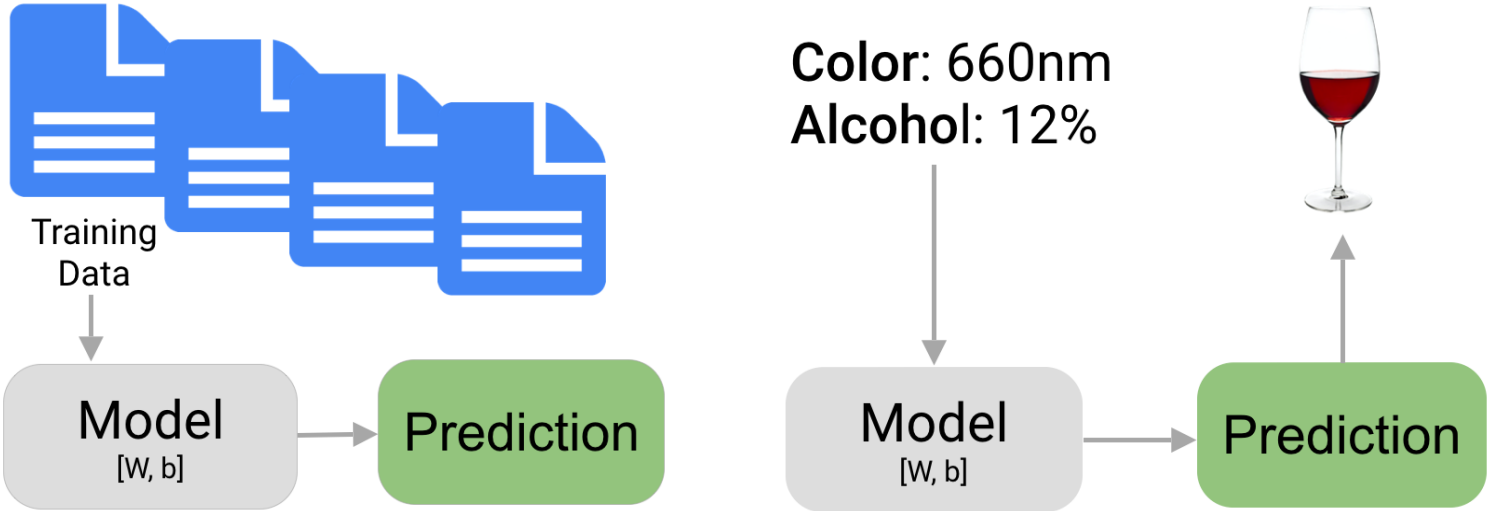


Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

什麼是機器學習

機器學習的概念

- 透過從過往的資料和經驗中學習並找到其運行規則，最後達到人工智慧的方法
- 而其中這些過往的資料我們稱之為『訓練資料 Training Data』
- 訓練資料中會包含兩部分『特徵 Feature』&『目標 lable』



什麼是機器學習

機器學習又分為...

監督式學習(Supervised learning)

- 分類 Classification

✓ 二元分類 —>
希望預測的目標 (label) 只有兩種　Ex. 男生or女生

✓ 多元分類 —>
希望預測的目標 (label) 有多個選項　Ex. 酒的品種

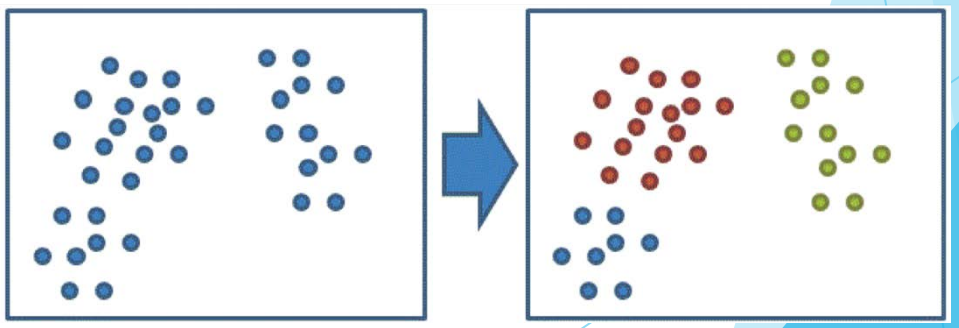
- 回歸 Regression

✓ 回歸分析 —>
希望預測的目標 (label) 是連續型資料
Ex. 商品的價格

非監督式學習 (Unsupervised learning)

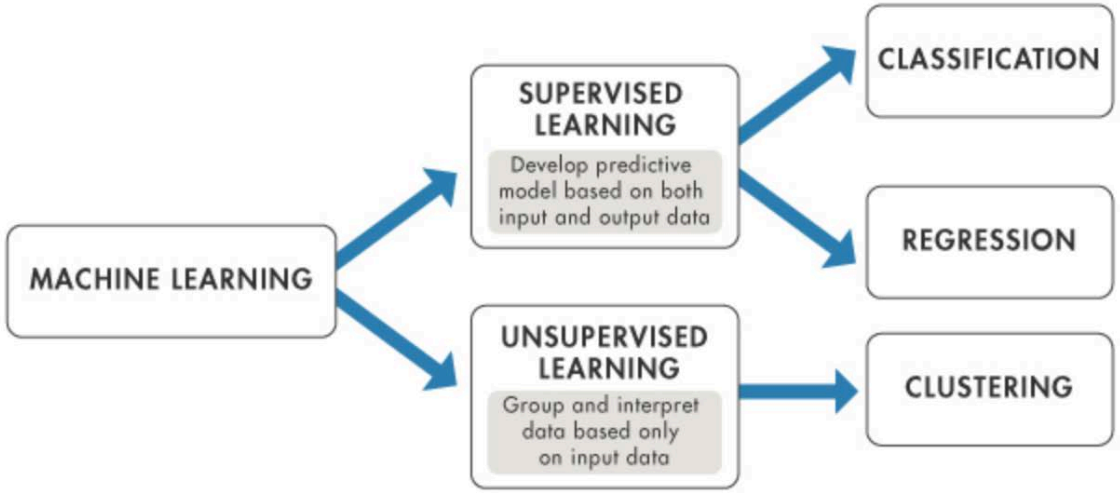
- 分群 Cluster

✓ 群集分析 —>
不知道要預測的答案，所以沒有目標 (label)



什麼是機器學習

架構圖

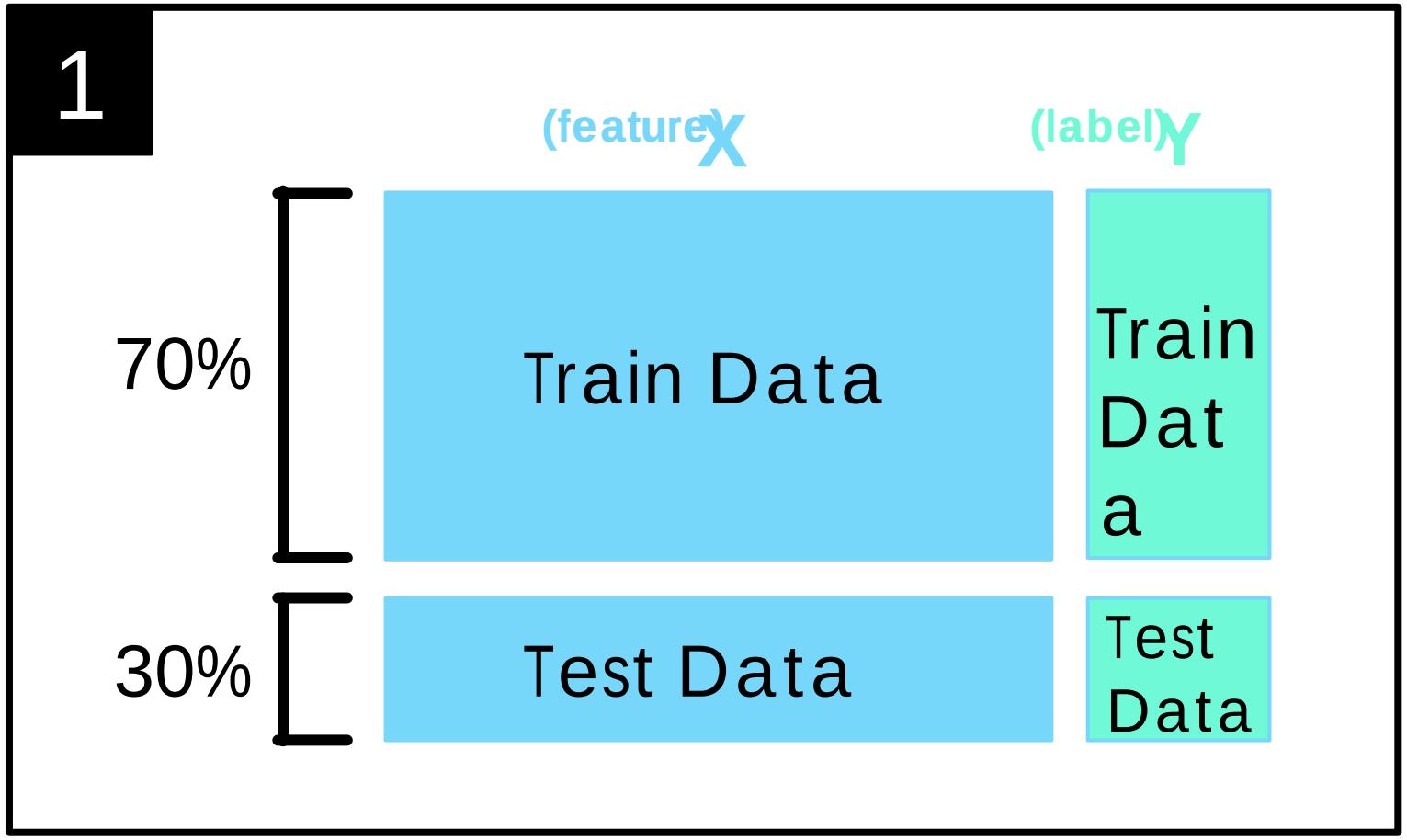


CLASSIFICATION	REGRESSION	CLUSTERING
Support Vector Machines	Linear Regression, GLM	K-Means, K-Medoids Fuzzy C-Means
Discriminant Analysis	SVR, GPR	Hierarchical
Naive Bayes	Ensemble Methods	Gaussian Mixture
Nearest Neighbor	Decision Trees	Hidden Markov Model
Neural Networks	Neural Networks	Neural Networks

分別對應不同的演算法

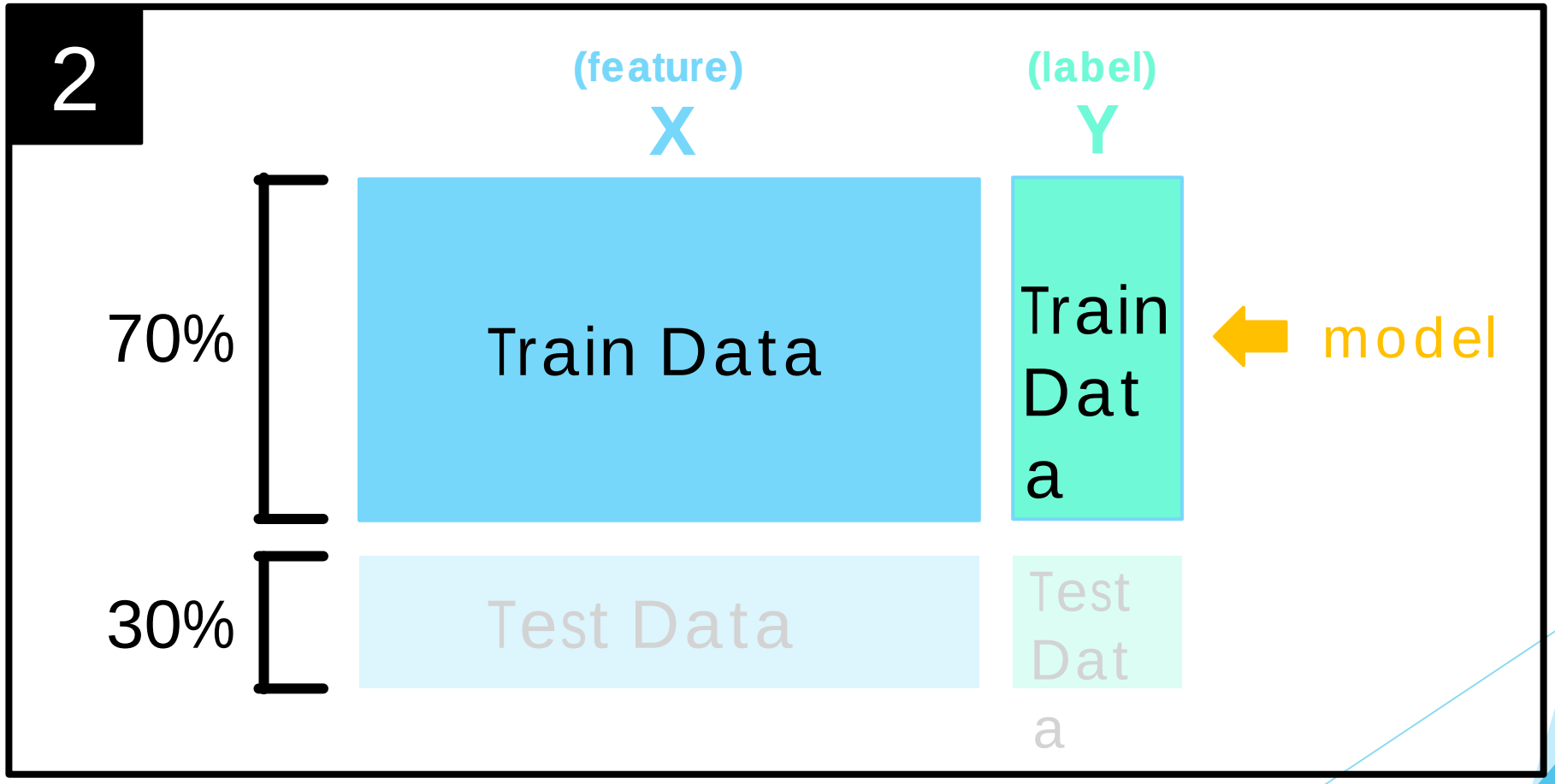
什麼是機器學習

機器學習的流程



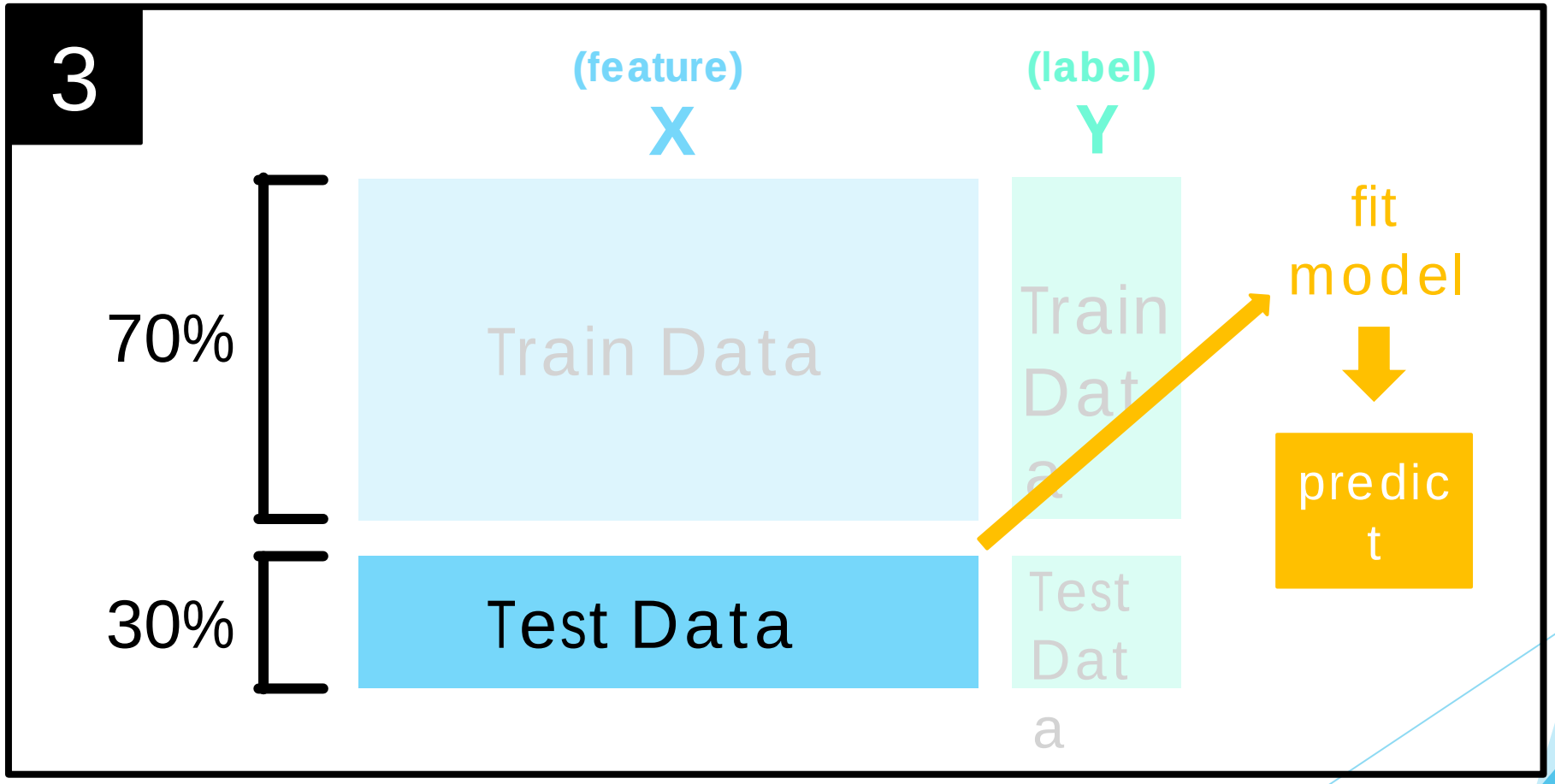
什麼是機器學習

機器學習的流程



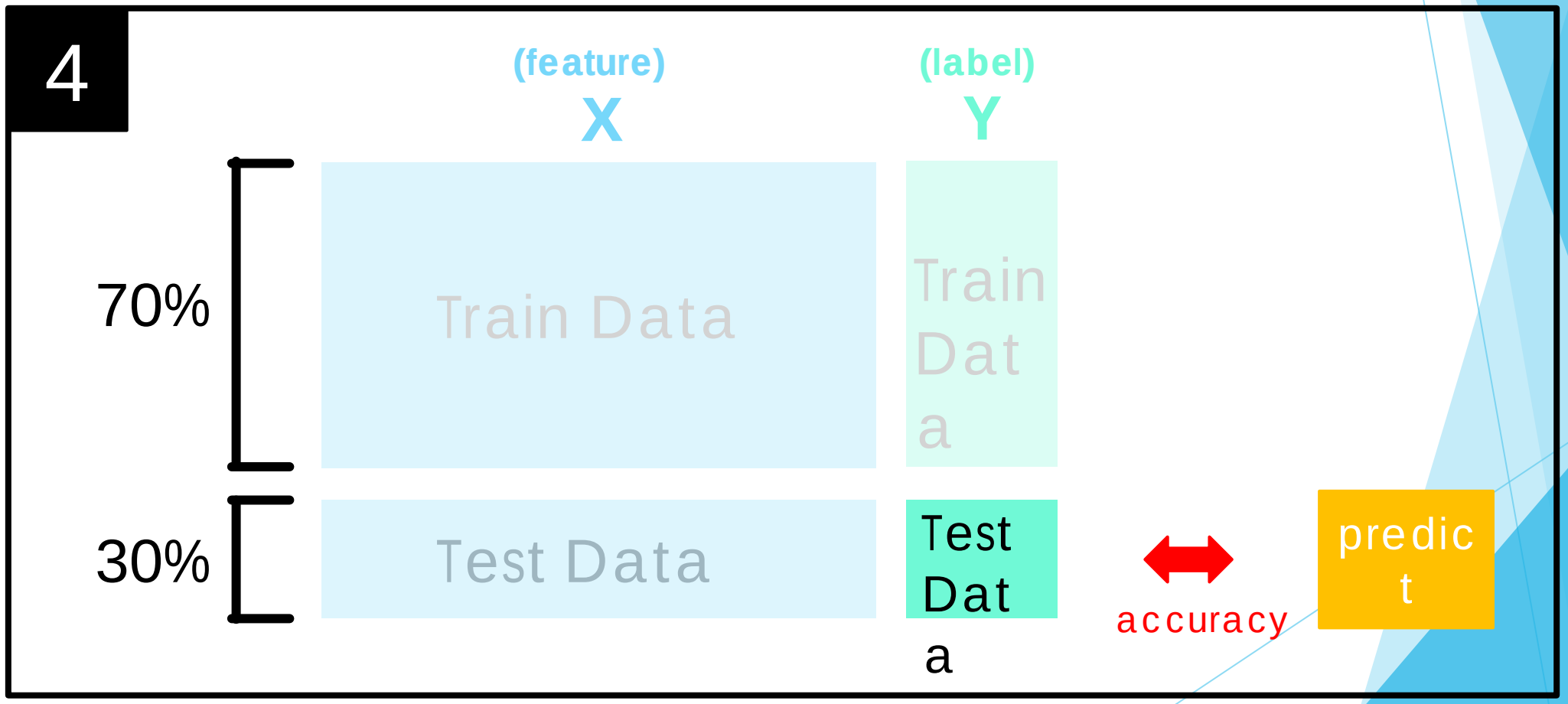
什麼是機器學習

機器學習的流程



什麼是機器學習

機器學習的流程



PART 02
資料前處理

資料前處理

匯入檔案

```
import pandas as pd
df = pd.read_excel('fanpage_clean.xlsx')
```

```
df.head()
```

	id	message
0	124616330906800_1560501197318299	阿娘威!披羊皮的狼?竟大口嚼小雞\n#要打統編:小編真的是快嚇死了... \n\n影片來源..
1	124616330906800_1560454417322977	被黑了!李毓芬演唱「大落拍」網友卻意外發現「亮點」\n#條紋編:這一段應該是昨天的亮點表演..
2	124616330906800_1559870414048044	誰說牠呆?心機月月調虎離山網友讚影帝\n#124616330906800_1559870414048044樂無編:最萌心機鬼~(*^Y)~y\n\n影片..

資料前處理

X & Y

- 使用分享數、留言數、小編、發文時間、發文星期預測按讚數量

```
X = df[['shares','comments','curator','hour','weekday']]
y = df['likes_count']
```

資料前處理

檢查資料

```
X.info()
```

```
<class 'pandas.core. frame. DataFrame’>
Range Index: 9975 entries, 0 to 9974
Data columns (total 5 columns):
shares 9975 non-null int64
comments 9975 non-null int64
curator 7232 non-null object
hour 9975 non-null int64
weekday 9975 non-null object
dtypes: int64(3), object(2)
memory usage: 389.7+ KB
```

資料前處理

處理遺失值

- 空值的小編填入未知

```
X['curator'] = X['curator'].fillna('未知')
```

```
X.info()
```

```
<class 'pandas.core. frame. DataFrame’>
Range Index: 9975 entries, 0 to 9974
Data columns (total 5 columns):
shares 9975 non-null int64
comments 9975 non-null int64
curator 7232 non-null object
hour 9975 non-null int64
weekday 9975 non-null object
dtypes: int64(3), object(2)
memory usage: 389.7+ KB
```

資料前處理

資料標準化

- pip install sklearn
- 將連續型的資料標準化

```
from sklearn import preprocessing
standard = preprocessing.StandardScaler()
X[['shares','comments']] = standard.fit_transform(X[['shares','comments']])
```

資料前處理

資料標準化

- 將類別型的資料轉成 Dummy Variable

```
X_1 = pd.get_dummies(X)
```

```
X_1.head()
```

	shares	comments	hour	curator_BG編	Curator_B編	curator_M編	curator_七條編	curator_人間四月編	curator_什麼編	curator_傻編	…	curator_高光編
0	0.241213	0.001331	11	0	0	0	0	0	0	0	…	0
1	-0.212199	-0.206505	11	0	0	0	0	0	0	0	…	0
2	0.677548	0.564335	10	0	0	0	0	0	0	0	…	0
3	-0.224154	-0.215934	10	0	0	0	0	0	0	0	…	0
4	-0.212199	-0.208076	10	0	0	0	0	0	0	0	…	0

資料前處理

Train & Test

- 將資料分成 Training data & Testing data

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_1, y, test_size = 0.3,
random_state = 2)
```

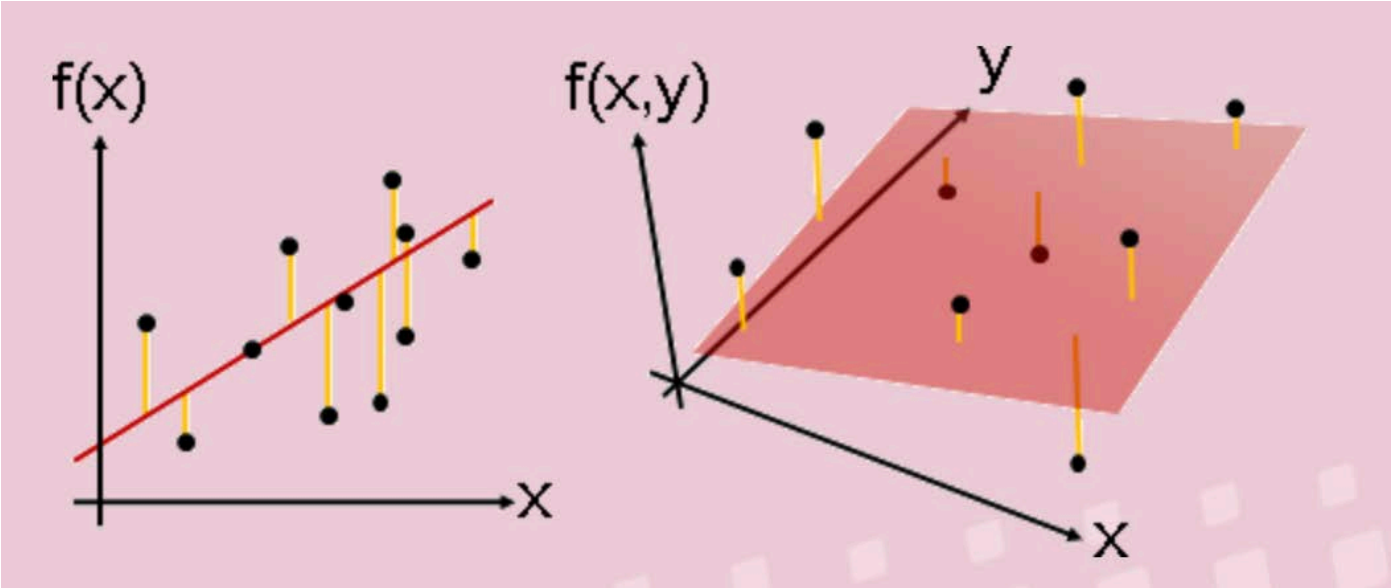
70%Training, 30%Testing

PART 03
預測模型：
Linear Regression

Linear Regression

線性迴歸介紹

- 迴歸就是找一個函數，盡量符合手邊的一堆數據



- 二維：直線
- 三維：平面
- 多維：超平面

(圖片來源：http://www.csie.ntnu.edu.tw/~u91029/Regression.html)

Linear Regression

線性迴歸介紹

- 簡單線性迴歸公式

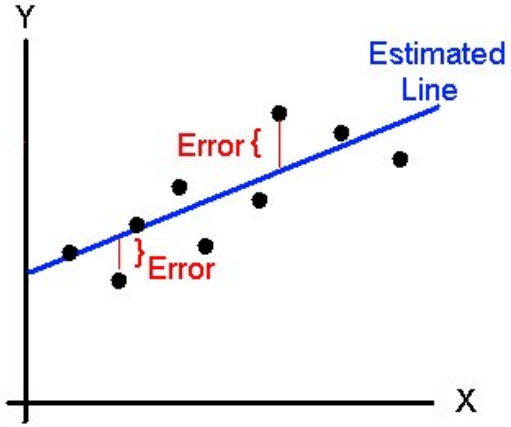
Estimated (or predicted) Y value for observation i

Estimate of the regression intercept

Estimate of the regression slope

Value of X for observation i

$$\hat{Y}_i = b_0 + b_1 X_i$$



- 複回歸

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \varepsilon_i$$

Linear Regression

匯入模型套件

- Linear Regression Model

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
```

- 模型訓練

```
lm.fit(X_train.values, y_train.values)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1
, normalize=False)
```

Linear Regression

查看模型

- 迴歸模型的截距項，也就是公式中的$\beta 0$

```
print(lm.intercept_)
```

```
3620.626025474225
```

- 迴歸係數

```
print(lm.coef_)
```

```
[ 3.00462596e+03  4.72062851e+03  2.01721953e+01 -2.28276444e+03
  2.07920813e+03  1.48858548e+02 -1.46823617e+03 -2.81825555e+03
  3.51436775e+03 -1.33455026e+03  4.27693492e+02  3.67418636e+02
 -1.60739471e+03 -3.07750639e+02 -1.51851326e+03  1.48574584e+03
```

Linear Regression

預測模型

- 進行預測

```
y_predict = lm.predict(X_test)
```

Linear Regression

預測模型

- 比較預測值 vs 真實值

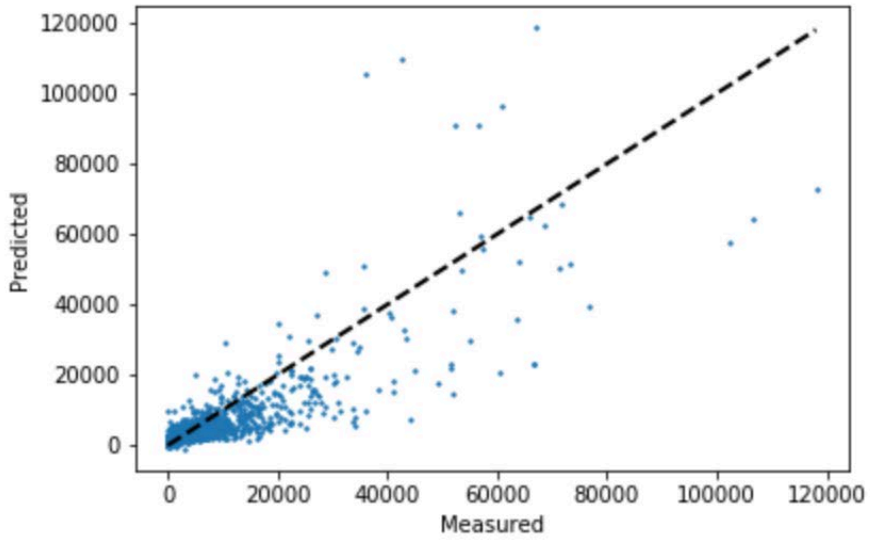
```
pd.DataFrame(list(zip(y_test.values, y_predict)),
             columns=['Measured','Predicted']).head()
```

	Measured	Predicted
0	6938	221.857708
1	860	2083.246286
2	128	273.179627
3	4851	5543.968663
4	491	12672.784547

Linear Regression

繪圖

```
import matplotlib.pyplot as plt
plt.scatter(y_test.values,y_predict,s=2)
plt.plot([y_test.values.min(), y_test.values.max()],
         [y_test.values.min(), y_test.values.max()], 'k--', lw=2)
plt.ylabel('Predicted')
plt.xlabel('Measured')
plt.show()
```



Linear Regression

檢驗迴歸模型

- 均方誤差MSE

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test.values, y_predict)
print("MSE : ",mse)
```

MSE : 23446608.538704727

- 均方根誤差RMSE

```
from math import sqrt
rms = sqrt(mean_squared_error(y_test.values, y_predict))
print("RMSE : ",rms)
```

RMSE : 4842.169817210537

Linear Regression

檢驗迴歸模型

- 判定係數 R-square

```
R_2 = lm.score(X_train, y_train)
print("R-squared  : ",R_2)
```

```
R-squared: 0.6991904295741282
```

- 調整後的判定係數 Adjusted R-square

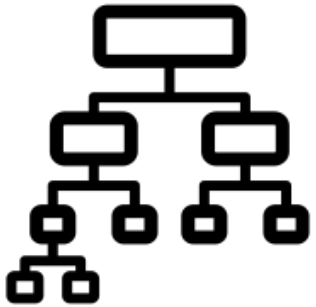
```
adj_R_2 = R_2 - (1 - R_2) * (X_train.shape[1] / (X_train.shape[0] - X_train.shape[1] - 1))
print("Adjusted R-squared : ",adj_R_2)
```

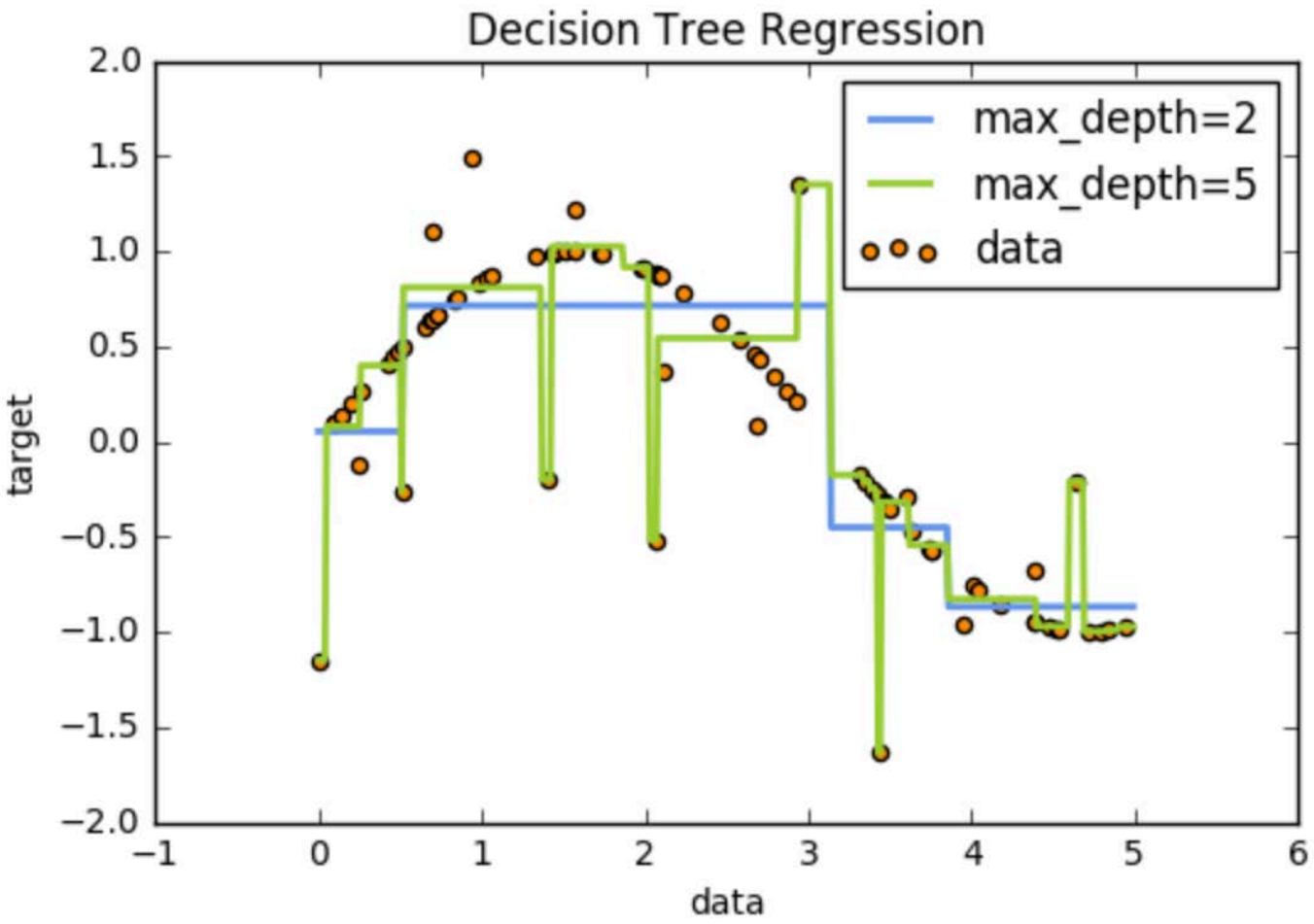
```
Adjusted R-squared:  0.6949518287125203
```

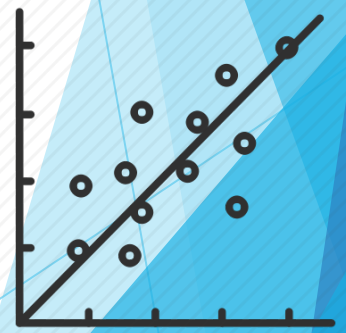
PART 04

Random Forest Regression

Random Forest Regression







Random Forest Regression

隨機森林迴歸介紹

- 一種結合分類和迴歸的演算法
- 從第二章的EDA分析時可以發現PO文的小編以及發文的時間確實會影響到按讚的數量
- 因此設想採用此模型效果更好

Random Forest Regression

匯入模型套件

- Random Forest Regression Model

```
from sklearn.ensemble import RandomForestRegressor
rfr = RandomForestRegressor()
```

- 模型訓練

```
rfr.fit(X_train.values, y_train.values)
```

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                      max_features='auto', max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                      min samples leaf=1, min samples split=2,
                      min weight fraction leaf=0.0, n estimators=10, n jobs=1,
```

Random Forest Regression

預測模型

- 進行預測

```
y_predict_rfr = rfr.predict(X_test)
```

Random Forest Regression

預測模型

- 比較預測值 vs 真實值

```
pd.DataFrame(list(zip(y_test.values, y_predict_rfr)),
             columns=['Measured','Predicted']).head()
```

	Measured	Predicted
0	**6938**	**3221.5**
1	860	**755.8**
2	128	292.6
3	4851	5626.1
4	**491**	730.9

Random Forest Regression

檢驗迴歸模型

- 均方誤差MSE

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test.values, y_predict_rfr)
print("MSE : ",mse)
```

MSE :19021269.861873765

- 均方根誤差RMSE

```
from math import sqrt
rms = sqrt(mean_squared_error(y_test.values, y_predict_rfr))
print("RMSE : ",rms)
```

RMSE：4361.338081583881 → 顯著下降

Random Forest Regression

檢驗迴歸模型

- 判定係數 R-square

```
R_2 = rfr.score(X_train, y_train)
print("R-squared  : ",R_2)
```

R-squared : 0.9449581808295875

- 調整後的判定係數 Adjusted R-square

```
adj_R_2 = R_2 - (1 - R_2) * (X_train.shape[1] / (X_train.shape[0] - X_train.shape[1] - 1))
print("Adjusted R-squared : ",adj_R_2)
```

Adjusted R-squared: 0.9441826060969423 → 顯著提升