Psychoinformatics - Week 3 (Exercises)

黃品綺 b09705035@ntu.edu.tw

1 Analyze what videos go viral? (8 points)

Please use YouTube APIs to carry out a data-driven or hypothesis-driven microstudy about the characteristics of viral videos.

You need to present, here in this notebook, AT LEAST two **statistical** figures or tables as supporting evidence for your arguments. Each of these figures/tables deserves 4 points.

In [ ]:
```
#API key
key = "AIzaSyC41URREPlNwjTd0V2JzEi51a0XaRO2jcs"
```

In [ ]:
```
from apiclient.discovery import build
import pandas as pd
import csv
```

In [ ]:
```
#將 video_response 轉成 dataframe 的函數
def table(video_response):

    #頻道
    channelId = []
    channelTitle = []
    country = []
    subscriberCount = []
    #影片
    title = []
    publishedDate = []
    categoryId = []
    videoId = []
    tags = []
    #統計
    viewCount = []
    likeCount = []
    commentCount = []
    favoriteCount = []

    for result in video_response.get("items", []): #item 一個搜尋結果 #[] 空列表，

        channelId.append(result['snippet']['channelId'])
        channelTitle.append(result['snippet']['channelTitle'])

        youtube = build('youtube','v3',developerKey=key)
        channel_response = youtube.channels().list(
            part='statistics, snippet',
            id=result['snippet']['channelId']).execute()
        if 'country' in channel_response['items'][0]['snippet'].keys():
            country.append(channel_response['items'][0]['snippet']['country'])
        else:
```

```
            country.append([])
        if 'subscriberCount' in channel_response['items'][0]['statistics'].keys(
            subscriberCount.append(channel_response['items'][0]['statistics']['s
        else:
            subscriberCount.append([])

        title.append(result['snippet']['title'])
        publishedDate.append(result['snippet']['publishedAt'])
        categoryId.append(result['snippet']['categoryId'])
        videoId.append(result['id'])
        if 'tags' in result['snippet'].keys():
            tags.append(result['snippet']['tags'])
        else:
            tags.append([])

        viewCount.append(result['statistics']['viewCount'])
        if 'likeCount' in result['statistics'].keys():
            likeCount.append(result['statistics']['likeCount'])
        else:
            likeCount.append([])
        if 'commentCount' in result['statistics'].keys():
            commentCount.append(result['statistics']['commentCount'])
        else:
            commentCount.append([])
        favoriteCount.append(result['statistics']['favoriteCount'])

    youtube_dict = {'channelId': channelId,'channelTitle': channelTitle,'country
                    'title': title,'publishedDate': publishedDate,'categoryId':
                    'viewCount': viewCount,'likeCount': likeCount,'commentCount'
    df = pd.DataFrame(data=youtube_dict)

    return df
```

In [ ]:

```
#搜尋並將 search_response 轉成 dataframe 的函數
def youtube_search(q, l, r, o, max_results=50, token=None):

    youtube = build('youtube','v3',developerKey=key)

    search_response = youtube.search().list( #list 定搜尋條件（給定參數）
        q = q,               #搜尋關鍵字
        type = "video",      #搜尋類型
        pageToken = token,   #分頁標記，用於分段擷取搜索結果
        order = o,           #排序方式（relevance 相關度）
        part = "id,snippet", #搜尋結果類型 #snippet 摘要訊息，包含標題、描述、頻道標
        publishedAfter="2023-01-01T00:00:00Z",
        maxResults = max_results, #搜尋結果最大數量
        location = l, #搜尋地點 #radius 方圓幾里的概念
        locationRadius = r).execute()

    videos = []

    for search_result in search_response.get("items", []):
        if search_result["id"]["kind"] == "youtube#video":
            response = youtube.videos().list(
                part='statistics, snippet',
                id=search_result['id']['videoId']).execute()

            videos.append(response)

    df = table(videos[0])
```

```
    return df
```

SEJ 的一篇文章 How To Make A Video Go Viral 中，提及 Ian Forrester（DAVID 的創辦人兼執行長）曾說過 viral vedios 至少具有以下其一特徵：

- A video which has a high share rate (shares/views x 100)
- A video which gets picked up by a platform algorithm and shown to a large number of viewers

---

TMR 臺灣行銷研究的一篇文章 透過資料視覺化，找出成為 Youtube 發燒影片的關鍵 中，使用美國區（regionCode='US'）的發燒影片做為資料集，分析發燒影片具有以下特徵：

- 不一定需要相當高的觀看次數、按讚數、評論數才能夠成為發燒影片（中位數與平均數進行比較）
- 標題長度多分布在 30 字至 60 字之間
- Official、Video、Trailer、How、2018 等是較為常見的詞，分別對應到「官方」、「串流影片預告」、「教學影片」、「當年度」
- 在週四、五發布影片，成為發燒影片的數目最高
- 在下午 2 點到 6 點發布影片，成為發燒影片的數目最高

根據以上兩篇文章，在此作業中將以台灣區（regionCode='TW'）的 Youtube trending videos 作為 viral videos 進行分析，驗證台灣區的發燒影片是否符合 **第二篇文章** 所述的特徵（因為 Youtube API v3 無法取得影片分享次數，所以無法驗證第一篇文章的第一點）。

由於 youtube.video.list(chart='mostPopular') 每次只會抓取 50 部影片，因此我將分別收集以下三個時間點的發燒影片做為資料集並比較。

- 10 月 3 日下午
- 10 月 4 日下午
- 10 月 5 日下午

In [ ]:
```
#收集資料集（利用 API 抓取發燒影片）
youtube = build('youtube', 'v3', developerKey=key)
video_response = youtube.videos().list(
    part='snippet,statistics',
    chart='mostPopular',
    regionCode='TW',maxResults=50).execute()

trending = table(video_response)
display(trending.head(3))

# trending.to_csv('03_TW_trending/231005.csv', index=False,encoding='utf-8-sig')
```

	channelId	channelTitle	country	subscriberCount	title	p
0	UCCQvP4hsRW9emj0meGk15jg	愛爾達體育家族 ELTA Sports	TW	641000	【中華隊滑輪溜冰】這就是對勝利的渴望！男子接力隊不放棄衝線0.01秒之差擊敗南韓摘金！	
1	UCR3asjvr_WAaxwJYEDV_Bfw	東森新聞 CH51	TW	2180000	3行人並排走被叭提醒！他神巧遇怒衝理論：在叭什麼 @newsebc #shorts	
2	UC2t5bjwHdUX4vM2g8TRDq5g	League of Legends	US	15000000	GODS ft. NewJeans (뉴진스) (Official Music Video)...	

In [ ]:
```
third = pd.read_csv('03_TW_trending/231003.csv')
forth = pd.read_csv('03_TW_trending/231004.csv')
fifth = pd.read_csv('03_TW_trending/231005.csv')
```

三天影片的比較（相同影片的數量）

In [ ]:
```
thfo, fofi, fith = 0, 0, 0

for i in range(50):
    if third['videoId'][i] in list(forth['videoId']):
        thfo += 1
    if forth['videoId'][i] in list(fifth['videoId']):
        fofi += 1
    if fifth['videoId'][i] in list(third['videoId']):
        fith += 1

compare = pd.DataFrame([[thfo, fofi, fith], [50 - thfo, 50 - fofi, 50 - fith], [
display(compare)
```

	3 vs 4	4 vs 5	3 vs 5
same	38.00	29.00	21.00
different	12.00	21.00	29.00
same rate	0.76	0.58	0.42

Youtube 發燒影片的更新頻率為 15 分鐘，但每次更新後的變動幅度不大。 由於這三天的影片重複率皆超過四成，因此不能合併三天的資料，故分開分析，結果也僅供參考（樣本數量不大）。

觀看次數、按讚數、評論數、訂閱數

In [ ]:
```
view, like, comment, subscriber, day = [], [], [], [], [third, forth, fifth]
tmp = []

for i in day:
    view.append(i['viewCount'].mean())
    view.append(i['viewCount'].median())
    view.append(round(i['viewCount'].mean() / i['viewCount'].median(), 2))

    tmp.append(sum(i['likeCount'] == "[]"))
    like.append(i[i['likeCount'] != "[]"]['likeCount'].astype(int).mean())
    like.append(i[i['likeCount'] != "[]"]['likeCount'].astype(int).median())
    like.append(round(i[i['likeCount'] != "[]"]['likeCount'].astype(int).mean()

    comment.append(i['commentCount'].mean())
    comment.append(i['commentCount'].median())
    comment.append(round(i['commentCount'].mean() / i['commentCount'].median(),

    subscriber.append(i['subscriberCount'].mean())
    subscriber.append(i['subscriberCount'].median())
    subscriber.append(round(i['subscriberCount'].mean() / i['subscriberCount'].m

print(tmp)
```

```
[1, 3, 5]
```

In [ ]:
```
third1 = pd.DataFrame(data={'View': view[0:3], 'Like': like[0:3], 'Comment': com
forth1 = pd.DataFrame(data={'View': view[3:6], 'Like': like[3:6], 'Comment': com
fifth1 = pd.DataFrame(data={'View': view[6:], 'Like': like[6:], 'Comment': comme

for i, j in zip([third1, forth1, fifth1], [3, 4, 5]):
    print("10 月 {} 日下午".format(j))
    display(i)
```

```
10 月 3 日下午
```

	View	Like	Comment	Subscriber
Mean	1681871.96	116235.081633	5867.78	5615734.0
Median	315102.00	6853.000000	374.50	677000.0
Ratio	5.34	16.960000	15.67	8.3

```
10 月 4 日下午
```

	View	Like	Comment	Subscriber
Mean	2014542.12	128391.659574	6473.4	5723616.00
Median	331519.50	7113.000000	449.5	852500.00
Ratio	6.08	18.050000	14.4	6.71

```
10 月 5 日下午
```

	View	Like	Comment	Subscriber
Mean	2180860.34	113665.266667	6438.62	3051850.00
Median	307411.00	6837.000000	379.00	659000.00
Ratio	7.09	16.630000	16.99	4.63

由三天的結果可見，**平均數皆為中位數的四到六倍以上**，特別是在按讚數與評論數有很大的差異，可知平均數明顯大於中位數，因此可以推論 不一定需要相當高的觀看次數、按讚數、評論數才能夠成為發燒影片 的特徵亦成立，**訂閱數也是**。

影片標題

In [ ]:
```
import re

#移除特殊字元的函數
def remove_special_characters(input_string):
    special_characters = re.findall(r'[^\w\s]', input_string)
    clean_string = re.sub(r'[^\w\s]', '', input_string)
    return clean_string, special_characters
```

In [ ]:
```
#計算字數的函數
def wordCount(df):
    word, CwordCnt, EwordCnt, total, sign = [], [], [], [], []
    for i in df['title']:
        tmp, tmp1 = remove_special_characters(i)
        word.append(tmp)
        C = len(re.findall(r'[\u4e00-\u9fff]', tmp))
        CwordCnt.append(C)

        english_characters = re.sub(r'[\u4e00-\u9fff]', '', tmp)
        tmp2 = english_characters.split(" ")
        tmp2 = [x for x in tmp2 if x != '']
        E = len(tmp2)
        EwordCnt.append(E)

        total.append(C + E)
        sign.append(tmp1)

    return pd.DataFrame(data={'Word': word, 'Chinese Word Count': CwordCnt, 'Eng
```

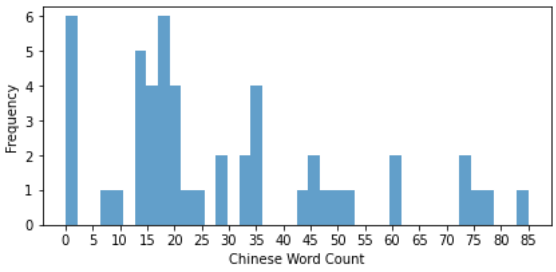
In [ ]:
```
third2 = wordCount(third)
forth2 = wordCount(forth)
fifth2 = wordCount(fifth)
display(third2.head(3))
```

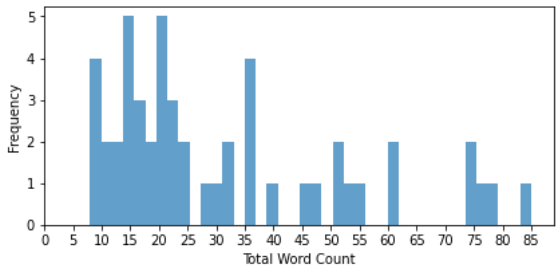
	Word	Chinese Word Count	English Word Count	Total Word Count	Sign
0	中華隊滑輪溜冰這就是對勝利的渴望男子接力隊不放棄衝線 001秒之差擊敗南韓摘金	35	1	36	[【, 】, ！, ., ！]
1	葬送的芙莉蓮 首播100分鐘 第0104話Muse木棉花 動畫 線上看	20	2	22	[-, |]
2	杭州亞運 韓國 VS 台灣看球閒聊直播	14	1	15	[【, 】]

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
```

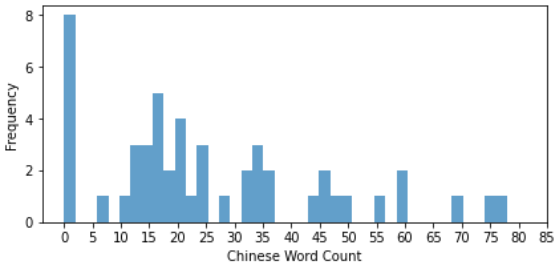
In [ ]:
```
for i, j in zip([third2, forth2, fifth2], [3, 4, 5]):
    print("10 月 {} 日下午".format(j))
    fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(15, 3))

    for k, m in zip(['Chinese', 'Total'], range(2)):
        axes[m].hist(i[k + ' Word Count'], bins=40, alpha=0.7)
        axes[m].set_xlabel(k + ' Word Count')
        axes[m].set_ylabel('Frequency')
        axes[m].set_xticks(np.arange(0, 90, 5))

    plt.show()
```

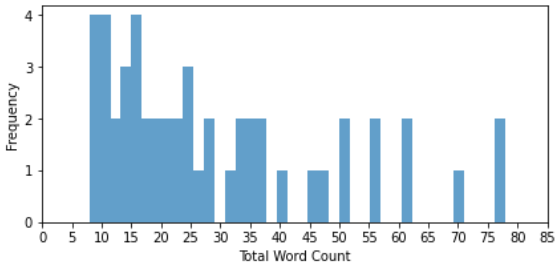
10 月 3 日下午



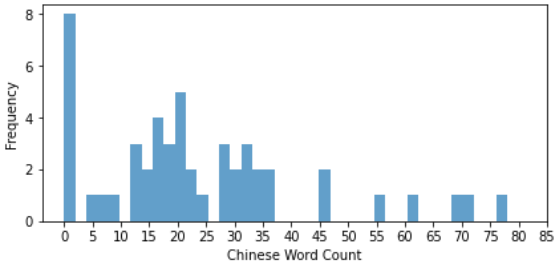


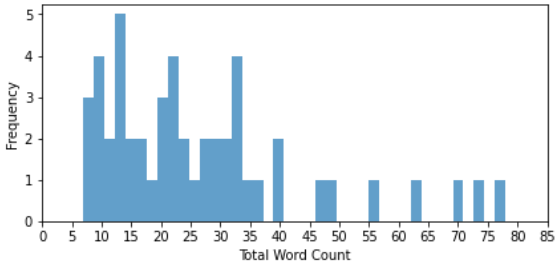
10 月 4 日下午





10 月 5 日下午





由三天的結果可見，字數主要分布在大約 **10 到 25 個字之間（正負 5 個字）**，與 標題長度多分布在 30 字至 60 字之間 的特徵不相符，很可能是因為語言所造成的差異。另外，由圖表所見，可以發現標題字數對是否成為發燒影片並沒有決定性的影響。

In [ ]:
```
from collections import Counter

for i, j in zip([third2, forth2, fifth2], [3, 4, 5]):
    print("10 月 {} 日下午".format(j))
    tokens = []
    for k in i["Sign"]:
        tokens = tokens + k

    bag_of_words = Counter(tokens)
    print(bag_of_words.most_common(5))

    tmp1 = []
    for k in bag_of_words.most_common(5):
        tmp = 0
        for n in i['Sign']:
            if str(k[0]) in n:
                tmp = tmp + 1
        tmp1.append(tuple([k[0], tmp]))

    print(tmp1)
    if j != 5: print("\n")
```

```
10 月 3 日下午
[('#', 75), ('！', 37), ('.', 24), ('・', 24), ('【', 21)]
[('#', 17), ('！', 23), ('.', 13), ('・', 12), ('【', 20)]


10 月 4 日下午
[('#', 64), ('.', 32), ('！', 28), ('【', 22), ('】', 22)]
[('#', 16), ('.', 15), ('！', 22), ('【', 21), ('】', 21)]


10 月 5 日下午
[('#', 73), ('.', 38), ('！', 26), ('【', 21), ('】', 21)]
[('#', 21), ('.', 17), ('！', 18), ('【', 21), ('】', 21)]
```

由於中文字詞的重複性不高，因此改以符號分析。每日的兩行分別為總出現次數以及出現在標題的次數。

由三天的結果可見，在影片標題中標點符號使用的前三名為 # 與 . 與 ！。 由結果可推測，在標題中使用 ！、【】和 # 可能有助於成為發燒影片（突顯標題和幫助搜尋）。

發布時間

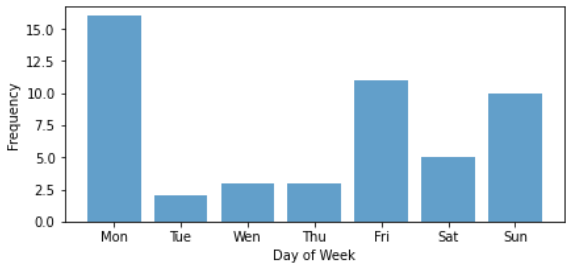
In [ ]:
```
#解析日期的函數
from datetime import datetime, timedelta

conversion_map = {
        "Monday": 'Mon',
        "Tuesday": 'Tue',
        "Wednesday": 'Wen',
        "Thursday": 'Thu',
        "Friday": 'Fri',
```

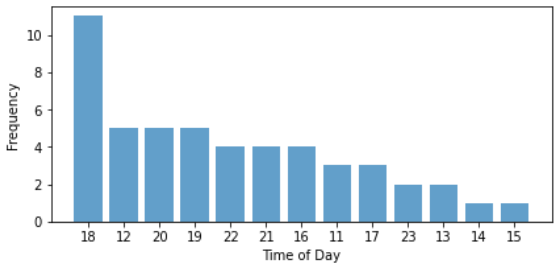
```
        "Saturday": 'Sat',
        "Sunday": 'Sun'
    }

def date_trans(date_str):
    date_obj = datetime.strptime(date_str, "%Y-%m-%dT%H:%M:%SZ")
    taiwan_offset = timedelta(hours=8)
    taiwan_time = date_obj + taiwan_offset

    day_of_week = taiwan_time.strftime('%A')
    week = conversion_map.get(day_of_week, None)
    time = taiwan_time.strftime('%H')

    return week, time
```

In [ ]:

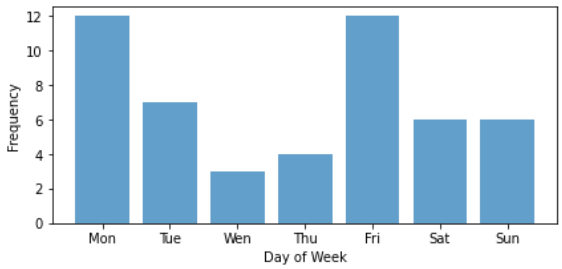
```
third3, forth3, fifth3 = [], [], []
label_w = ['Mon', 'Tue', 'Wen', 'Thu', 'Fri', 'Sat', 'Sun']
# label_t = ['00', '01', '02', '03', '04', '05', '06', '07', '08', '09', '10',
#            '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '

for i, j, k in zip([third3, forth3, fifth3], [third, forth, fifth], [3, 4, 5]):
    print("10 月 {} 日下午".format(k))

    for m in j['publishedDate']:
        i.append(date_trans(m))
    i = pd.DataFrame(data=i, columns=['Week', 'Time'])
    cnt_w = i['Week'].value_counts()[label_w]
    cnt_t = i['Time'].value_counts()

    fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(15, 3))

    axes[0].bar(label_w, list(cnt_w.values), alpha=0.7)
    axes[0].set_xlabel('Day of Week')
    axes[0].set_ylabel('Frequency')

    axes[1].bar(cnt_t.index, list(cnt_t.values), alpha=0.7)
    axes[1].set_xlabel('Time of Day')
    axes[1].set_ylabel('Frequency')

    plt.show()
```

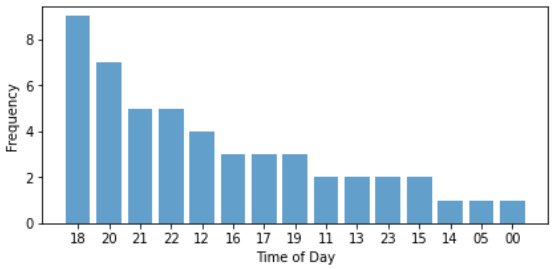
```
10 月 3 日下午
```



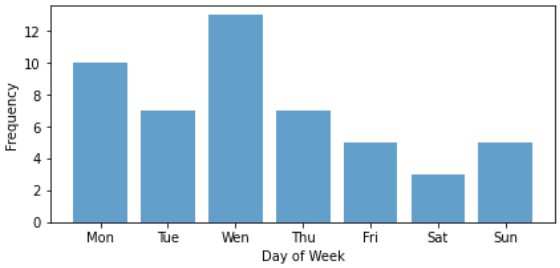


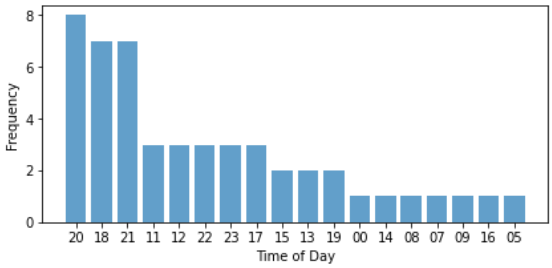

```
10 月 4 日下午
```





```
10 月 5 日下午
```





星期方面

- 10 月 3 日（二）**週一** 發布影片成為發燒影片的數目最高，**週二** 則最低。
- 10 月 4 日（三）**週一、五** 發布影片成為發燒影片的數目最高，**週三** 則最低。
- 10 月 5 日（四）**週三** 發布影片成為發燒影片的數目最高，**週六** 則最低。

由此結果，難以評判與 在週四、五發布影片成為發燒影片的數目最高 的特徵是否相符，因為樣本數太少且嚴重受到資料收集時間的影響。

時間方面

- 10 月 3 日 **下午 6 點** 發布影片成為發燒影片的數目最高，次高則為 **下午 12、8、7 點**
- 10 月 4 日 **下午 6 點** 發布影片成為發燒影片的數目最高，次高則為 **下午 8 點**
- 10 月 5 日 **下午 8 點** 發布影片成為發燒影片的數目最高，次高則為 **下午 6、9 點**

由此結果可見，整體而言在 **下午 6 點到 10 點** 發布影片成為發燒影片的數目最高，這與 在下午 2 點到 6 點發布影片成為發燒影片的數目最高 的特徵不相符，很可能是因為地區文化所造成的差異。另外，可以發現幾乎沒有在早上 0 點到 10 點間發布的影片，可以推測原因與這段為大多數人上班和睡眠的時間有關。

總結以上的分析，可以發現僅有第一點的特徵（不一定需要相當高的觀看次數、按讚數、評論數才能夠成為發燒影片）是相同的，其餘可能因為語言、地區文化的而有所差異。

以下是台灣地區 10 月 3 日到 10 月 5 日，Youtube 發燒影片的特徵整理：

- 不一定需要相當高的觀看次數、按讚數、評論數、訂閱數才能夠成為發燒影片
- 標題長度多分布在 10 字至 25 字之間（包含中英文）
- 在標題中使用 ！、【 】 和 # 的符號可能有助於成為發燒影片
- 因為樣本數太少且嚴重受到資料收集時間的影響，難以評判在星期幾發部影片成為發燒影片的數目會最高
- 在下午 6 點到 10 點發布影片，成為發燒影片的數目較高；在上午 0 點到 10 點發布影片，成為發燒影片的數目較低