粉絲專頁觀察分析

台灣科技大學社群媒體分析實務期末報告

2017.1.15 | 電機工程系 | M10407323 | 王靖煊

Outline

- 背景與觀察
- 分析方法與目的
- 資料統計
  - 分析專頁資訊
  - 專頁基本資訊
  - 貼文響應廂型圖
  - 粉絲行為分析
  - 迴響度變化量
- 資料分析結果
- 結論

背景與觀察

全聯小編 聽說真的很猛

不論是電視上的全聯先生，還是粉絲專頁的文案撰寫，全聯福利中心的廣告文案總是讓人印象深刻。

全聯粉絲專頁的貼文目的，與其他賣場的性質相同，不外乎就是促銷活動、本期強檔或是節慶採購資訊。

但全聯的小編在內文裡加入了更多巧思：標記朋友獲得抽獎資格、精美的設計圖文吸引注意、甚至連看似純粹的喇豬屎，都是廣告文宣的一部份。

究竟強大的全聯粉絲專頁，與其他同樣經營賣場，卻較少被提及的「大潤發」、「家樂福」粉絲專頁，有哪些巨大的差異呢？

分析方法與目的

分析方法

我採用毛敬豪老師提供的 Facebook 資料集，並使用 ElasticSearch 進行存放。

在開發工具方面，我使用 Anaconda 中提拱的 iPython Notebook，以及下列套件：
json、matplotlib.pyplot、numpy、pandas、datetime、seaborn、pyes

主要分析項目包含：
- 粉絲專頁基本公開資訊
- 專頁貼文與回應變化趨勢
- 粉絲行為分析（好友 Tag 分析）
- 專頁發文的迴響度變化量

並且都使用 Function 方式建立，只要輸入粉絲專頁 id 即可進行分析。

資料統計

Step 1. 分析專頁資訊

首先，我要看看這份資料集當中有哪些粉絲專頁的資料。

所以先列出所有粉絲專頁的基本資訊，以確保我們要分析的目標專頁（大潤發、全聯、家樂福）的資訊量足夠。

欲分析專頁的 id 分別為：

大潤發 RT-mart
156435698616

全聯福利中心
134004310003557

家樂福 Carrefour Taiwan
170290829685289

觀察 162 個粉絲專頁資訊與名稱

In [8]:
```
q = pyes.query.MatchAllQuery()

result = conn.search(query=q , indices='facebook_nested' , doc_types='fanpage')

print "TalkCount\tLikesCount\tHereCount\tfid\t\t\tname"
for x in result:
    print x['talking_about_count'],"".ljust(5), "\t", x['likes_count'],"".ljust(5), "\t", x['were_here_count'],"".ljust(
```

TalkCount	LikesCount	HereCount	fid	name
2	70	0	158512494180943	祥和老人養護中心
58	173	277	342383462532625	慈暉老人養護中心
5	364	168	162761550493760	台北市私立龍江老人長期照顧中心(養護型)
0	7	0	1563716147234837	忠祥老人養護中心
5	297	23	760574903991825	大愛老人養護中心
7067	122421	0	157941490891336	Ford Taiwan
5	100	226	636929313051699	臺北市私立祥寶尊榮老人長期照顧中心-養護型
4	231	134	535072236552536	大坪林老人長期照顧中心
0	229	0	1633445946934758	登革熱防疫大作戰
5258	65734	0	154929428406	疾病管制署 - 1922防疫達人
0	57	0	899855000096200	成大防疫志工隊
31	552	0	387140921383751	慈園老人 長期照顧中心
0	148	0	683316918367656	建安長期照顧中心
16	1072	138	153448974709254	長祐老人長期照顧中心（養護型） • 聖祐護理之家
10	1491	0	157812647606634	社團法人台灣長期照護管理學會
547	13648	0	398088637066969	台灣長期照顧關懷協會 TLTCA
8272	586570	0	76625396025	CDC
0	184	54	608371179281556	吉安長期照顧中心
0	337	44	170609949662329	友緣長期照顧中心
3770	104080	0	77322222971	Acer Taiwan
15039	5531482	0	9924322884	P&G
0	159	3	738370826273530	財團法人臺北市私立恆安老人長期照顧中心 - 長期照護型
2	190	44	1425784827690511	臺北市私立景興老人長期照顧中心（養護型）
75	424	293	435102099875111	佳南老人長期照顧中心
2	161	0	781385955304973	長照營養大師
75	704	0	179334535508784	臺北市私立慧華老人長期照顧中心-養護型
1	114	0	949478508432591	台灣長期照顧居家服務員資訊
68	1072	0	308714755874341	銀髮樂齡健康與長期照護資訊網 Health and Long-term care
6316	652859	0	210121695693833	中國信託優惠情報“讚”
38976	645087	0	248954465174600	TOYOTA Taiwan
13541	175728	204	202586376996	義美食品
1702	30529	0	180080052032125	光泉"HOT"鮮奶
112144	81176236	0	7270241753	YouTube
31442	777271	0	304834096975	Sony Mobile TW
1505	1482037	0	134278719946186	Bayer

Step 2. 專頁基本資訊

我使用 Calculation 函式，印出三個目標專頁的粉絲數量、貼文數量與評論數量。

這時候就可以感受到數據上的差異：

1. 全聯福利中心擁有最多的粉絲。
2. 在貼文數量只有大潤發、家樂福的一半的情況下，還能獲得將近 10 倍大潤發的留言 (Comments) 數量，貼文成效相當驚人！

可謂：貼文重「質」不重「量」！

```
def Calculation(id1, id2, id3):
    names.append(Name(id1))
    names.append(Name(id2))
    names.append(Name(id3))
    fans.append(Fans(id1))
    fans.append(Fans(id2))
    fans.append(Fans(id3))
    posts.append(Posts(id1))
    posts.append(Posts(id2))
    posts.append(Posts(id3))
    comments.append(Comments(id1))
    comments.append(Comments(id2))
    comments.append(Comments(id3))

    s0 = pd.Series(fans, index = names, name = 'Fans count')
    s1 = pd.Series(posts,  index = names, name = 'Posts count')
    s2 = pd.Series(comments, index = names, name = 'Comments count')
    barResult = pd.concat([s0, s1, s2], axis=1)
    return barResult

print Calculation(156435698616, 1340043100003557, 170290829685289)
```

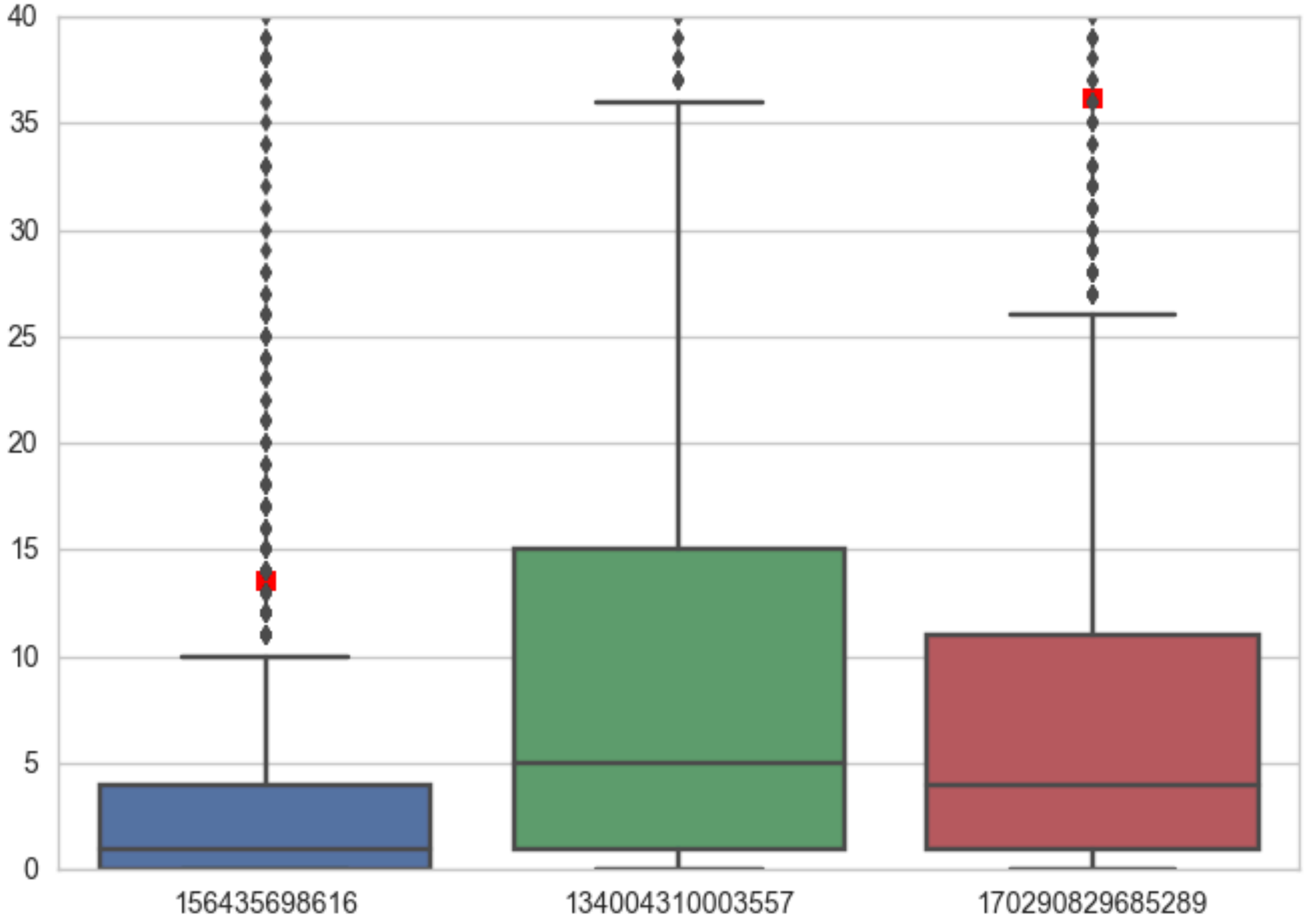
	Fans count	Posts count	Comments count
大潤發 RT-mart	82187	2175	27278
全聯福利中心	864812	1497	276667
家樂福 Carrefour Taiwan	745430	2312	76174

Step 3. 貼文響應廂型圖

粉絲專頁貼文最重要的目標之一，就是讓粉絲們可以參與迴響！

所以我撰寫了一個 Function，可以在輸入粉專 id 以後，直接匯出圖表。

這邊我發現一件事：
post 資料集中 Query 的 field = "id" 與 comment 資料集中 Query 的 field = "fid_pid" 是同一個東西，所以我可以從 post 資料集中的 comment_count 欄位直接得知 comments 數量，也可從 comment 資料集中的每個 fid_pid 計算 comments 數量。



Step 4. 粉絲行為分析

粉絲專頁貼文的第二個目的，就是要讓粉絲們分享訊息，加速傳播！

所以程式中的這個 Function，可以在輸入粉專 id 以後，直接匯出「分享數量」與「標記數量」的報表與百分比。

資料集當中有 3 種 Tag:
message_tags: Profiles tagged in message. This is an object with a unique key for each tag in the message.
story_tags: Deprecated field, same as message_tags.
with_tags: Profiles tagged as being 'with' the publisher of the post.
故我採用 message_tags 為標準！

```
def ShowResult(fid):
    resultBuffer = [Posts(fid), Shares(fid), Tags(fid)]
    print "The page id", fid, "has:"
    print "All posts count:", resultBuffer[0]
    print "Posts including share:", resultBuffer[1], "(", resultBuffer[1]*100/resultBuffer[0], "% )"
    print "Posts including tags:", resultBuffer[2], "(", resultBuffer[2]*100/resultBuffer[0], "% )\n"

ShowResult(156435698616)
ShowResult(134004310003557)
ShowResult(170290829685289)
```

```
The page id 156435698616 has:
All posts count: 2175
Posts including share: 1545 ( 71 % )
Posts including tags: 444 ( 20 % )

The page id 134004310003557 has:
All posts count: 1497
Posts including share: 1467 ( 97 % )
Posts including tags: 234 ( 15 % )

The page id 170290829685289 has:
All posts count: 2312
Posts including share: 2203 ( 95 % )
Posts including tags: 324 ( 14 % )
```

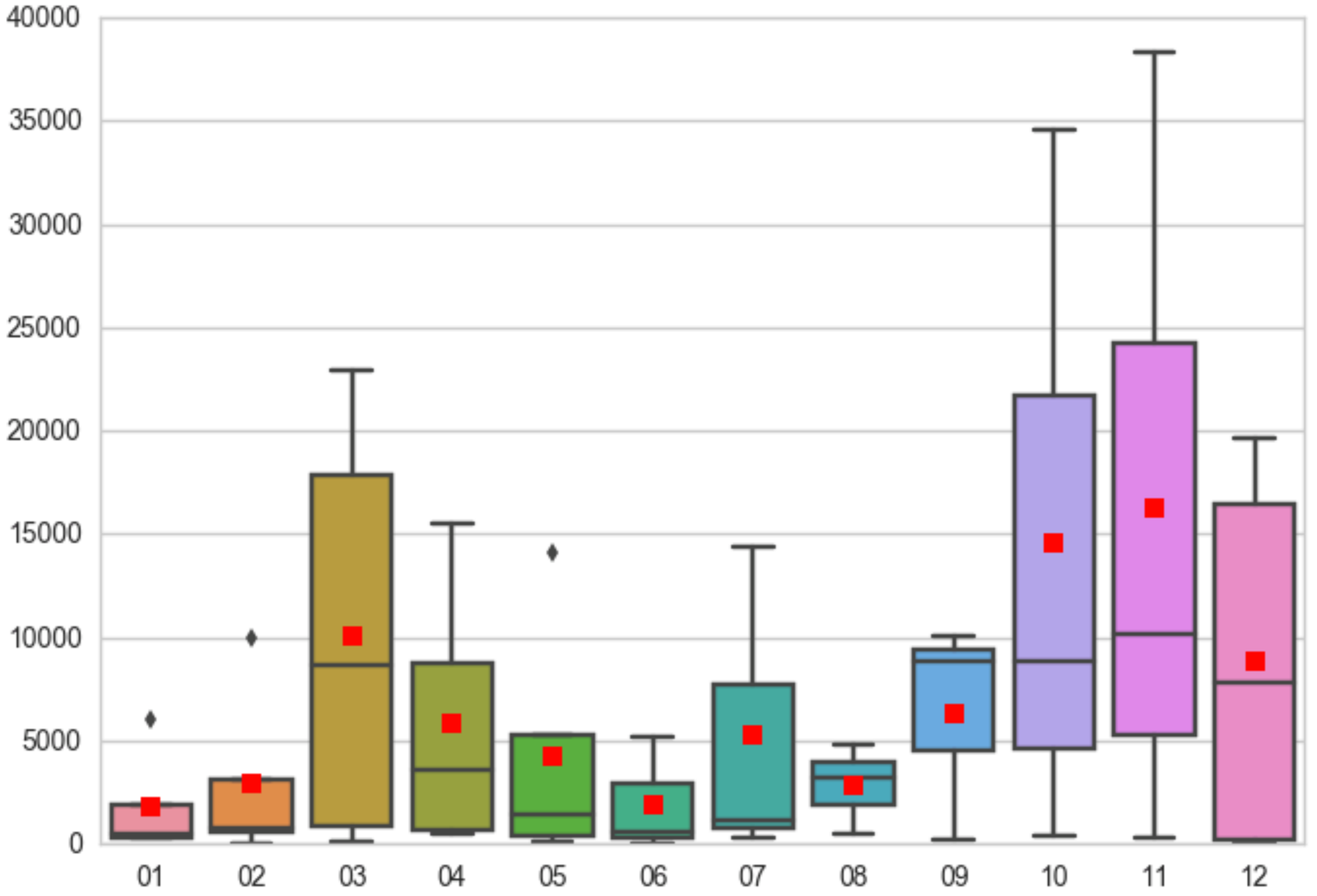
我後來透過非常多的方法，想找出貼文之間 Tag 的數量，不過資料集當中並沒有提供所謂的 Tag Count，而且從 Query 下來的資料，到 ElasticsSearch 的查詢介面，都看不出有無標記朋友的差別，於是詳細項目便無法分析了！

Step 5. 迴響度變化量

最後，我對粉專每月 Po 文的迴響度變化量進行分析。

這邊我第一次嘗試使用 Aggregation 函式來進行運算。之前使用的方法，都是透過大量的迴圈，直接在 Query 的資料中進行數量的加總；不過透過 pyes 好用的 DateHistogramAgg 函式，我們可以輕鬆輸出某時間區間總和的 JSON 格式資料，再透過解 JSON 的方法，直接取得每月的貼文迴響數據資料。

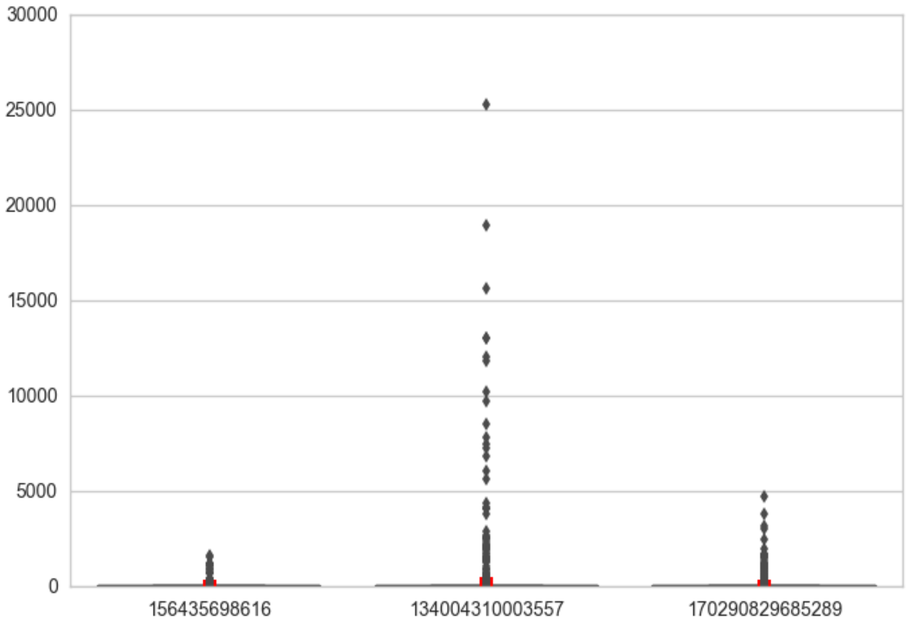
從圖表中可以發現，每年 3 月以及年底，都是粉絲回文數大增的時期！



資料分析結果

分析結果

1. 從 Step 1. 輸出的表格來看，這份資料集裡面怎麼有這麼多老人照護與安養中心呢？這些粉絲專頁真的是全國百大之一嗎哈哈哈！
2. 從 Step 2. 的資料查詢結果來看，「全聯」粉絲團的小編真不簡單，使用最少的貼文，卻能達到最多的粉絲與迴響度。
3. 再進行如 Step 3. 的廂型圖表分析時，一定要懂得把 y 軸調小，Zoom in 至少到顯示平均值的地方，不然預設的結果圖就會像右邊這張，簡直慘不忍睹。
4. Step 4. 中的 Tag 數量分析雖然失敗，但過程還挺有趣的！甚至曾經還試過 Query “@” 符號，以及其他千奇百怪的方法，但當然是行不通啦！
5. Step 5. 的結果，是最有感的一張圖表，如果能有更長期的分析（如 5 年以上的資料）相信會有更明顯的趨勢！



結語

結語

必須說！這門課絕對是我研究所修課（甚至是大學時修的課）最有收穫的前三名！

當初在修課前，完全沒有 Python 與資料分析的實務經驗，只有聽學長說「好課推推」就來了，

如果一樣遇到學弟，我一定會說同一句話：「好課推推」

兩次堅持自己完成的作業，讓我從完全零基礎的情況，變得開始有點感覺了！Python 不再是那麼生疏又陌生的東西，甚至到後來才發現：天哪！我想做某件事，網路上就會有文件告訴我們怎麼做，而且破碎的想法與程式東拼西湊，竟然還能跑起來！（錯誤示範）真是太神奇了！

總之，這學期不管在技術層面，還是思考的方式，都有些不一樣，非常感謝毛老師的教導！

比較可惜的是，在機器學習方面的分析，我還沒能力完成，相信加入老師提到的 u–p 矩陣、p–t 矩陣、Deep learning model、Naive 之類的工具，一定能對社群資料有非常透徹的分析！

期待還能夠再次參與老師的課程，謝謝老師！