Amplification Factor - AF Score

Tuan Tran

June 21, 2019

Giới Thiệu Trong tài liệu này sẽ giới thiệu định nghĩa, ý tưởng và giải thích các scripts dùng để tính toán điểm Content Creation (CC) score.

1 Giới Thiệu về Content Creation score

Định Nghĩa CC score dùng để đánh giá chất lượng bài viết và khả năng tạo nội dung của người dùng. Chất lượng một bài viết bao gồm hai yếu tố: hàm lượng giá trị nội dung và hình thức diễn đạt. Điểm số đánh giá bài viết là tổng hợp của hai yếu tố này. Nó giúp người dùng giải quyết một trong hai vấn đề sau:

- Các thông tin nóng bỏng trên mạng xã hội (hot news): tin giật gân, tin liên quan đến các sự kiện kinh tế, chẳng hạn như việc tăng giá điện, tăng giá xăng, các tin nóng trên mạng xã hội.
- Các xu hướng tiếp cận nội dung (trending): giới trẻ ngày càng có xu hướng thích xem các clip ngắn thay vì đọc các bài viết dưới dạng văn bản.

Phương pháp tiếp cận Việc đánh giá chất lượng bài viết và khả năng tạo nội dung của người dùng gồm hai phần: mức ảnh hưởng của người viết và giá trị của nội dung. Độ đo của việc đánh giá này được thể hiện thông qua mức ảnh hưởng của một bài viết với giá trị AF score của bài viết. Thông thường một người dùng (user) sẽ có một vài bài post hầu như không có giá trị về nội dung (baseline posts). Những tương tác của bài viết đó đơn thuần là do ảnh hưởng (quan hệ) của người viết. Từ đây, ta có thể định nghĩa điểm nội dung của một bài viết bất kỳ bằng cách so sánh nó với các baseline posts như sau:

$$CC_i = \frac{AF_i}{AF_b}$$

Tuy nhiên việc xác định một baseline post AF_b sẽ gặp nhiều khó khăn, vì có thể đó là một bài viết không có nội dung, chẳng hạn như các phát biểu "Tôi mệt quá !", mặt khác, có thể có những bài viết rất dài nhưng cũng không có nội dung. Việc xác định một bài baseline là một bài toán khó trong việc xử lý ngôn ngữ tự nhiên và hiện nay vẫn chưa có lời giải tốt.

Vì vậy, trong dự án, để giải quyết bài toán này bằng cách tiếp cận khác. Chúng ta có thể xem AF_b là AF nhỏ nhất trong tất cả bài viết của người dùng.

$$AF_b = min\{AF_i\}$$

Mặt khác, trong thực tế những bài viết của một người dùng luôn có những bài có AF score rất thấp (bằng 0) vì một lý do đặc biệt nào đó. Do đó, để tránh những trường hợp ngoại lệ, chúng ta đặt:

$$AF_b = \text{mean of bottom } 10\% \text{ of all posts of the user}$$

Trong quá trình thực hiện và kiểm chứng CC score thì nhóm thấy do người dùng sẽ thuộc nhiều nhóm người khác nhau. Do đó, nhóm đề suất xài AF_b là sử dụng giá trị trung bình của 10% bài posts có AF score thấp nhất của 1 nhom thay vì của 1 người dùng riêng rẽ. Đồng thời để tránh việc thay đổi liên tục của AF_i thì, nhóm sử dụng AF_t là giá trị trung bình của 10% bài có điểm AF cao nhất thay cho AF_i. Việc sử dụng này sẽ tăng tính hợp lý và độ chính xác cho CC score của 1 người trong 1 khoảng thời gian nhất định.

$$AF_t = \text{mean of top } 10\% \text{ of all posts of the user.}$$
$$AF_b = \text{mean of bottom } 10\% \text{ of all posts of the cluster that users belong}$$

Từ những bước phát triển trên, CC score của một người dùng trong một giai đoạn nhất định sẽ được tính như sau:

$$CC_u = \frac{AF_t}{AF_b}$$

$$Where: AF_t = mean\ of\ top\ 10\%of\ all\ posts\ of\ the\ user.$$
$$AF_b = mean\ of\ bottom\ 10\%of\ all\ posts\ of\ the\ cluster\ that\ users\ belong$$

Về hướng đi tương lai khi dữ liệu đủ lớn, Để tính CC score chính xác hơn. Ta cần phải thực hiện 1 step clustering, để gom các người dùng tương đồng về cùng 1 nhóm, Sau đó sử dụng công thức trên, sẽ mang đến độ chính xác tốt hơn.

2 Diễn giải về script

Trước hết cần phải import các library cần thiết đặc biệt là pymongo bởi vì database là MongoDB. Và sau đó là tạo kết nối với database, trong đây cần phải kết nối 2 database vì sẽ cần sử dụng database đã đánh mentioned.

```
import pandas as pd
import numpy as np
import datetimedatabase
```

```
from pymongo import MongoClient


client1 = MongoClient('45.122.223.198:27017',       #IP address of database
                  username = 'kapiReadOnly',        #Username
                  password = 'pl2oieAt9#tnWV!Yc0',  #Password
                  authSource = 'kapi',              #name of database
                  authMechanism = 'SCRAM-SHA-1')
kapi = client1['kapi']


client = MongoClient('45.122.223.198:27017',       #IP address of database
                  username = 'kpi-v2R',             #Username
                  password = 'EecvKJxdTQ1JEK8J2FA', #Password
                  authSource = 'kpi-v2',            #name of database
                  authMechanism = 'SCRAM-SHA-1')
kpitest = client['kpi-v2']
```

Tiếp đến là query data cần thiết để tính, trong đó cần hai function query data. Function đầu tiên dùng để query thông tin cần thiết có liên quan tới tất cả các bài post của một người dùng trong một khoảng thời gian nhất định.

```
def get_ge_post(users_list, start_date, end_date):
    cluster = kapi['posts'].find({'to_user': {'$in': users_list},
                                  'parent_id': None,'created_date': {'$gte':
                                     start_date, '$lt': end_date}},
                                 {'_id': 0, 'fid': 1, 'to_user': 1, 'created_date': 1,
                                  'comments_count': 1, 'shares_count': 1,
                                     'likes_count': 1})

    tb = pd.DataFrame()
    for a in cluster:
        tmp = pd.DataFrame([a])
        tb = tb.append(tmp, ignore_index=True)

    tb['total_int'] = [np.nansum([x, y, z]) for x, y, z in
                       zip(tb['comments_count'], tb['likes_count'],
                           tb['shares_count'])]

    return tb
```

Function thứ hai là dùng để query các post có mentioned tới một industry hay một brand cụ thể nào đó

```
def get_me_post(users_list, industry, start_date, end_date):
    cluster = kpitest['posts'].find({'to_user': {'$in': users_list},
                                'industry': industry, 'parent_id': None,
                                'created_date': {'$gte': start_date, '$lt':
                                    end_date}},
                               {'_id': 0, 'fid': 1, 'to_user': 1, 'created_date': 1,
                                'comments_count': 1, 'shares_count': 1,
                                    'likes_count': 1})

    tb = pd.DataFrame()
    for a in cluster:
        tmp = pd.DataFrame([a])
        tb = tb.append(tmp, ignore_index=True)

    tb['total_int'] = [np.nansum([x, y, z]) for x, y, z in
                    zip(tb['comments_count'], tb['likes_count'],
                        tb['shares_count'])]

    return tb
```

Bước tiếp theo là query data để chuẩn bị tính toán. Function query tất cả post của một người dùng đc lưu về "ge_tol". Còn Function query những post mà đã được đánh mention tới một industry hay brand nào đó được lưu về "me_tol".

```
user_list = ["100001997853167", "100003714391961", "567112096",
    "100004095850481", "100003798186002"]

start_date = datetime.datetime(2018,7,1)
end_date = datetime.datetime(2018,12,31)
industry = 'baby_milk_powder'
brand = 'optimum' #'apple', 'huawei', 'oppo', 'samsung' # 'nan', 'friso',
    'similac', 'enfa', 'optimum'

ge_tol = get_ge_post(user_list, start_date, end_date)
me_tol = get_me_post(user_list, industry, start_date, end_date)
```

Từ bảng "me_tol", sủ dụng hàm aggregate để tạo ra bảng "cc_tol_tb", đây là bảng dùng để tính CC score. Sau đó là tạo ra function "cc_fun" để tính CC score trên bảng "cc_tol_tb".

```
cc_tol_tb = (me_tol
            .groupby(['to_user'])
            .agg({'fid': 'count'})
```

```
            .rename(columns={'fid': 'number_pst'})
            .reset_index())


def cc_fun(to_user, num_me_post):
   cc = (me_tol[me_tol.to_user == to_user ].nlargest(int(num_me_post*.1 + .9),
       'total_int').total_int.mean()/
     ge_tol.nsmallest(int(ge_tol.shape[0]*.1 + .9),
         'total_int').total_int.mean())*np.log10(num_me_post + 1)

   return cc
```

Bước cuối là tính CC score, sủ dụng function "cc_fun" đã được define ở trên. Sau đó xuất ra file "cc_scr_test.csv"

```
cc_tol_tb['cc score'] = cc_tol_tb.apply(lambda x: cc_fun(x['to_user'],
    x['number_pst']), axis=1)

cc_tol_tb = cc_tol_tb.sort_values('cc score', ascending=False)

cc_tol_tb.to_csv('cc_scr_test.csv')
```