ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO BÀI TẬP LỚN

XỬ LÍ VÀ PHÂN TÍCH DỮ LIỆU FANPAGE FACEBOOK

Sinh viên: **Phạm Thành Long**

Mã sinh viên: **22022604**

Môn: Lập trình xử lí dữ liệu với Python

Mã học phần: AIT2003 1

Học kỳ: I

Năm học: 2023-2024

Mục Lục

I - Giới thiệu

Đây là bài báo cáo về việc xử lí và phân tích dữ liệu của fanpage trên facebook. Giúp sinh viên nắm vững kiến thức và kĩ năng về thu thập, làm sạch và xử lí dữ liệu. Đồng thời rèn luyện các khả năng quan sát, nhận xét tình huống cùng việc viết báo cáo hỗ trợ cho việc học sau này.

Fanpage được lựa chọn là **Clash of Clans** (https://www.facebook.com/ClashofClans).

Đôi nét về Clash of Clans:

Fanpage của Clash of Clans trên Facebook là nguồn thông tin chính thức và cộng đồng sôi động dành cho những người yêu thích trò chơi chiến lược nổi tiếng này. Với mục tiêu tạo ra một không gian tương tác và giao lưu, Fanpage không chỉ cung cấp thông tin về những cập nhật mới nhất của Clash of Clans mà còn là nơi để người chơi chia sẻ kinh nghiệm, chiến thuật, và tìm hiểu về cộng đồng đa dạng của mình.

Nơi đây, người chơi có cơ hội tham gia vào các sự kiện đặc biệt, giải đấu, và cuộc thi, giúp tạo ra một không khí sôi động và hứng khởi. Fanpage còn là nơi người chơi có thể tìm thấy thông tin hữu ích về các bản vá mới, chiến thuật hiệu quả, và những câu chuyện thú vị từ cộng đồng.

Với sự tương tác tích cực, Fanpage không chỉ là nơi để nhận thông tin mà còn là cơ hội để thể hiện đam mê, góp ý và kết nối với hàng triệu người chơi khác trên khắp thế giới. Khám phá và tham gia Fanpage để trở thành một phần không thể thiếu của cộng đồng Clash of Clans trên Facebook!

II - Thu thập dữ liệu

II.1 - Frameworks

Chúng ta sẽ sử dụng framework hỗ trợ việc crawl dữ liệu từ facebook tự động đó là facebook-scraper
⇒ *Tài liệu hướng dẫn sử dụng:* https://github.com/kevinzg/facebook-scraper

II.2 - Thiết đặt

Install frameworks

```
[ ]
# %pip install facebook_scraper pandas
```
Python

Import frameworks

```
[ ]
from facebook_scraper import *
import pandas as pd
import time
import numpy as np
```
Python

Setup

```
[ ]
FANPAGE_LINK ="ClashofClans" # Fanpage to crawl
FOLDER_PATH = "../Data/Raw/" # Folder to save data
COOKIE_PATH = "../config/cookies.txt" # Path to cookies file

PAGES_NUMBER = 30 # Number of pages to crawl
```
Python

```
[ ]
results = [] # List of results
start_url = None # Url to start crawling from facebook
def handle_pagination_url(url): # Handle pagination url
    global start_url
    start_url = url
```
Python

II.3 - Thu thập

Do nếu chúng ta crawl dữ liệu với nhiều trường cùng một lúc sẽ dẫn đến khả năng bị khóa tài khoản hoặc bị thiếu sót dữ liệu sẽ rất lớn ⇒ Vì thế ta sẽ chia ra crawl từng thành phần sau bằng cách điều chỉnh tham số trong API crawl dữ liệu

- bình luận (*comments*)
- cảm xúc (*reactions*)
- người bày tỏ cảm xúc (*reactors*)

Tiến hành crawl comment sử dụng cách dưới đây:

Crawling

```
set_cookies(COOKIE_PATH)
while True:
    try:
        for post in get_posts(FANPAGE_LINK, pages=PAGES_NUMBER , start_url=start_url, request_url_callback=handle_pagination_url,
                              options={"comments": True, "reactions": False,
                                       "reactors": False, "shared_text": False, "post_url": False}):
            results.append(post)
        print("All done")
        break
    except exceptions.TemporarilyBanned:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("start_url.txt", "a")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
    except exceptions.AccountDisabled:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("start_url.txt", "w")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
```

Save URL

```
f = open("start_url.txt", "w")
f.write(start_url + "\n")
f.close()
```

Để có thể crawl dữ liệu một cách liên tục mà không cần phải crawl lại từ đầu mỗi khi xảy ra lỗi, ta sẽ lưu lại đường dẫn tuyệt đối của bài viết được chọn làm điểm bắt đầu (*start_url*).

Hơn nữa, cookies cũng cần được thay đổi trong phần xử lí ngoại lệ mỗi khi frameworks xảy ra lỗi để tránh việc dừng chương trình khi ta muốn crawl trong khoảng thời gian dài.

Sau khi crawl được dữ liệu về *results*, ta sẽ lưu data lại dưới dạng .csv và .npy (dạng của numpy array)

Save data

```
# Initialize dataframe to scrape Facebook post
post_df_full = pd.DataFrame(columns=results[0].keys(), index=range(len(results)), data=results)
# To df
path=FOLDER_PATH + 'ClashOfClans_Comments'
post_df_full.to_csv(path + '.csv', index=False)
# print(path)
post_np_full = np.array(post_df_full)
np.save(path + ".npy", post_np_full)
```

Python

Việc lựa chọn những định dạng file trên cũng cần được xem xét kĩ càng bởi nếu ta sử dụng file excel thì một số trường dữ liệu có thể bị mất mát do giới hạn kí tự của một ô excel vì vậy thay vào đó chúng ta sử dụng định dang .csv để tránh rủi ro trên.

Sau khi thử nghiệm thì có một vài trường có định dạng dictionary nhưng không chuyển được về đúng kiểu của chúng (đã test qua json.load() và ast.literal_eval()) nên chúng ta sử dụng định dạng .npy được sử dụng để lưu trữ kiểu dữ liệu của các trường.

Tương tự ta có thể crawl reactions và reactors như với comments (chỉ cần thay một vài tham số và đường dẫn)

```python
set_cookies(COOKIE_PATH)
while True:
    try:
        for post in get_posts(FANPAGE_LINK, pages=PAGES_NUMBER , start_url=start_url, request_url_callback=handle_pagination_url,
                              options={"comments": False, "reactions": True,
                                       "reactors": False, "allow_extra_requests": True}):
            results.append(post)
        print("All done")
        break
    except exceptions.TemporarilyBanned:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("start_url.txt", "a")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
    except exceptions.AccountDisabled:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("start_url.txt", "w")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
```

Python

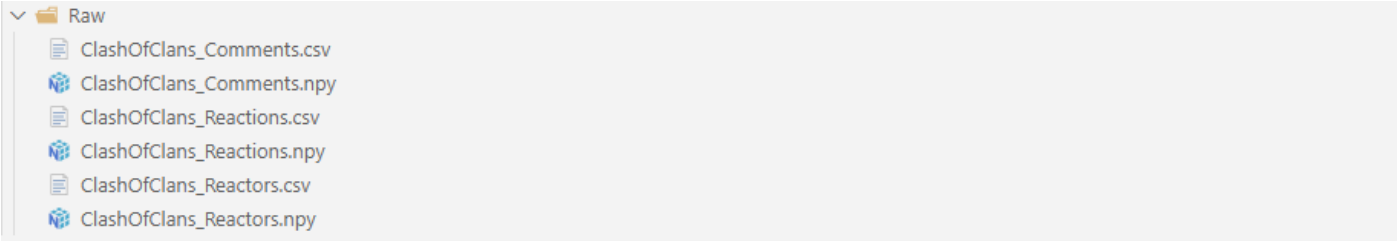
Với reactions ta để tham số reactions = True. Còn reactors cũng sẽ làm tương ứng.

```python
set_cookies(COOKIE_PATH)
while True:
    try:
        for post in get_posts(FANPAGE_LINK, pages=PAGES_NUMBER , start_url=start_url, request_url_callback=handle_pagination_url,
                              options={"comments": False, "reactions": False,
                                       "reactors": True, "allow_extra_requests": True}):
            results.append(post)
        print("All done")
        break
    except exceptions.TemporarilyBanned:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("start_url.txt", "a")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
    except exceptions.AccountDisabled:
        print("Temporarily banned, sleeping for 10m")
        if len(results) > 0:
            f = open("../start_url.txt", "w")
            f.write(start_url + "\n")
            f.close()
        time.sleep(600)
        set_cookies(COOKIE_PATH)
```

Python

```
All done
```

Sau khi crawl thành công, bộ dữ liệu thô mà ta thu thập được bao gồm như sau:



Tiếp đó, ta sẽ tiến hành tiền xử lí để có data sạch cho việc phân tích của chúng ta.

III - Tiền xử lí dữ liệu

III.1 - Xử lí data *reactions* thô

Ở bước này, chúng ta chỉ xử lí tổng quan phần dữ liệu hiện tại để chuẩn bị cho việc xử lí ở bước sau.

Ta cần tạo một dataframe để thực hiện tiền xử lí dữ liệu của mình

```
[97]
import pandas as pd
import numpy as np
```
Python

```
[98]
cols = np.array(pd.read_csv('../Data/Raw/ClashOfClans_Reactions.csv').columns)
arr = np.load('../Data/Raw/ClashOfClans_Reactions.npy', allow_pickle=True)
```
Python

```
[99]
df = pd.DataFrame(arr, columns=cols)
```
Python

Về tổng thể thì data thu được có thông tin như sau

```
df.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 511 entries, 0 to 510
Data columns (total 51 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   post_id                         511 non-null    object
 1   text                            511 non-null    object
 2   post_text                       511 non-null    object
 3   shared_text                     486 non-null    object
 4   original_text                   20 non-null     object
 5   time                            511 non-null    datetime64[ns]
 6   timestamp                       511 non-null    object
 7   image                           258 non-null    object
 8   image_lowquality                511 non-null    object
 9   images                          510 non-null    object
 10  images_description              510 non-null    object
 11  images_lowquality               511 non-null    object
 12  images_lowquality_description   511 non-null    object
 13  video                           149 non-null    object
 14  video_duration_seconds          0 non-null      object
 15  video_height                    0 non-null      object
 16  video_id                        149 non-null    object
 17  video_quality                   0 non-null      object
 18  video_size_MB                   0 non-null      object
 19  video_thumbnail                 149 non-null    object
...
 49  was_live                        511 non-null    object
 50  fetched_time                    375 non-null    datetime64[ns]
dtypes: datetime64[ns](3), object(48)
memory usage: 203.7+ KB
```

Tổng cộng có 51 trường, tuy nhiên trường chúng ta cần quan tâm trong bước xử lí này đó là *reactions* và *reaction_count*.

Tiếp theo, chúng ta sẽ kiểm tra sơ lược xem data của chúng ta có sai sót chỗ nào hay không.

Đầu tiên, data thu được có 511 bản ghi nhưng có đến 11 bản ghi trùng lặp. Việc trùng lặp này xảy ra trong khi chúng ta crawl dư liệu nhiều lần với start_url là url của bài viết cuối cùng của lần crawl trước đó.

```
df.shape
```
Python

```
(511, 51)
```

```
df['post_id'].duplicated().sum()
```
Python

```
11
```

Thứ hai, trường *reactions* có một vài bản ghi có *key* là Tiếng Việt (chúng ta thống nhất dùng bản ghi Tiếng Anh). Nguyên nhân của việc này là do tài khoản được sử dụng để thu thập là tài khoản tiếng Việt nên các dữ liệu lấy được sẽ không phải là tiếng Anh. Ví dụ như bản ghi thứ 300 dưới đây:

```python
df['reactions'].iloc[300]
```

```
{'thích': 4119,
 'yêu thích': 997,
 'haha': 23,
 'wow': 8,
 'thương thương': 47,
 'buồn': 10,
 'phẫn nộ': 19}
```

Hơn nữa, có một số bản ghi của *reactions* bị thiếu một vài loại cảm xúc do bài viết không có người bày tỏ cảm xúc đó hoặc do lỗi của frameworks. Lấy mẫu từ bản ghi thứ 288 có thể thấy bản ghi này không có 'buồn'.

```python
df['reactions'].iloc[288]
```

```
{'thích': 2073,
 'yêu thích': 368,
 'haha': 8,
 'wow': 13,
 'thương thương': 27,
 'phẫn nộ': 5}
```

Sau khi tìm ra được những sai sót, ta cần tiến hành xử lí những vấn đề trên theo các cách tiếp theo đây.

Việc *thứ nhất* đó là chuyển ngôn ngữ cho các loại cảm xúc của bài viết, đồng thời đặt các cảm xúc không được liệt kê có giá trị bằng 0.

```python
def trans(dic):
    if dic is None:
        return np.nan
    new_dic = {}
    if 'thích' in dic.keys():
        new_dic['like'] = dic['thích']
    elif 'like' in dic.keys():
        new_dic['like'] = dic['like']
    else :
        new_dic['like'] = 0
    if 'yêu thích' in dic.keys():
        new_dic['love'] = dic['yêu thích']
    elif 'love' in dic.keys():
        new_dic['love'] = dic['love']
    else :
        new_dic['love'] = 0
    if 'haha' in dic.keys():
        new_dic['haha'] = dic['haha']
    else :
        new_dic['haha'] = 0
    if 'wow' in dic.keys():
        new_dic['wow'] = dic['wow']
    else :
        new_dic['wow'] = 0
    if 'thương thương' in dic.keys():
        new_dic['care'] = dic['thương thương']
    elif 'care' in dic.keys():
        new_dic['care'] = dic['care']
    else :
        new_dic['care'] = 0
    if 'buồn' in dic.keys():
        new_dic['sad'] = dic['buồn']
    elif 'sad' in dic.keys():
        new_dic['sad'] = dic['sad']
    else :
        new_dic['sad'] = 0
    if 'phẫn nộ' in dic.keys():
        new_dic['angry'] = dic['phẫn nộ']
    elif 'angry' in dic.keys():
        new_dic['angry'] = dic['angry']
    else :
        new_dic['angry'] = 0
    return new_dic
```

0.0s Python

```python
df.reactions = df['reactions'].apply(trans)
```

0.0s Python

Kiểm thử và nhận được kết quả như ta kì vọng

```
df.reactions.iloc[300]
```
✓ 0.0s

```
{'like': 4119,
 'love': 997,
 'haha': 23,
 'wow': 8,
 'care': 47,
 'sad': 10,
 'angry': 19}
```

```
df['reactions'].iloc[288]
```
✓ 0.0s

```
{'like': 2073,
 'love': 368,
 'haha': 8,
 'wow': 13,
 'care': 27,
 'sad': 0,
 'angry': 5}
```

Việc còn lại thì ta chỉ cần loại bỏ đi các bài viết bị trùng có trong data.

```
df.drop_duplicates(subset=['post_id'], inplace=True)
df.index = range(df.shape[0])
```
✓ 0.0s Python

```
df['post_id'].duplicated().sum()
```
✓ 0.0s Python

```
0
```

```
df.shape
```
✓ 0.0s Python

```
(500, 51)
```

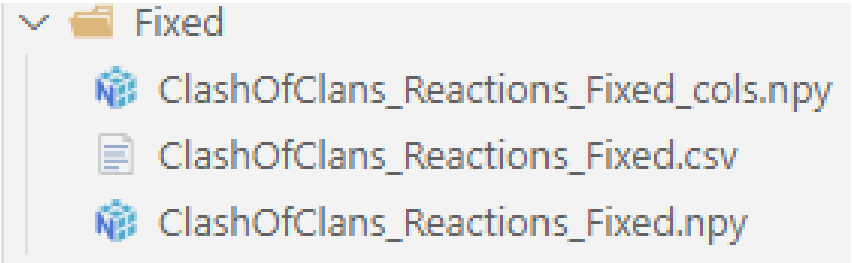
Cuối cùng sau khi xử lí thì ta lưu trữ dữ liệu đó lại.

```
# # save
# path='../Data/Fixed/ClashOfClans_Reactions_Fixed'
# df.to_csv(path + '.csv', index=False)

# arr = np.array(df)
# np.save(path + ".npy", arr)

# np.save(path + "_cols.npy", cols)
```
✓ 0.0s

Chúng ta đã có được kho dữ liệu mới – dữ liệu đã được sửa như dưới đây



Như vậy, phần data đầu tiên đã được tiền xử lí, hãy tiến tới phần tiếp theo.

III.2 - Xử lí data *reactors* thô

Tương tự như phần xử lí *reactions*, bước này ta cần quan tâm đến trường *reactors* của data.

```
import pandas as pd
import ast
import numpy as np
```

```
# df = pd.read_csv('../Data/Raw/ClashOfClans_Reactors.csv')
```

```
cols = np.array(pd.read_csv('../Data/Raw/ClashOfClans_Reactors.csv').columns)
arr = np.load('../Data/Raw/ClashOfClans_Reactors.npy', allow_pickle=True)
```

```
df = pd.DataFrame.from_records(arr, columns=cols)
```

```
df['post_id'].duplicated().sum()
```

```
0
```

Bước đầu kiểm tra có thể thấy được rằng data không có bài viết trùng lặp. Tuy nhiên, có một vấn đề xuất hiện cũng giống như phần trước đó là lỗi về ngôn ngữ. Ví dụ như bản ghi dưới đây:

```
df['reactors'][0][66]
```

```
{'name': 'Wesly Garsia',
 'link': 'https://facebook.com/profile.php?id=100089893557024&eav=AfZaWDMMNIqD_zgf6S1JUu1_aAs8p4KLZRcPM9bVmoW5qSewemIsLOmu7_l56ANLi7w&fref=pb&paipv=0',
 'type': 'yêu thích'}
```

Do đó chúng ta chỉ việc xử lí như dưới đây để có thể nhận được data chuẩn.

```
def f(x):
    if x is None:
        return np.nan

    for y in x:
        if y['type'] == 'thích':
            y['type'] = 'like'
        elif y['type'] == 'yêu thích':
            y['type'] = 'love'
        elif y['type'] == 'thương thương':
            y['type'] = 'care'
        elif y['type'] == 'phẫn nộ':
            y['type'] = 'angry'
        elif y['type'] == 'buồn':
            y['type'] = 'sad'
    return x
```

```
df['reactors'] = df['reactors'].apply(f)
```

```
df['reactors'][0][66]
```

```
{'name': 'Wesly Garsia',
 'link': 'https://facebook.com/profile.php?id=100089893557024&eav=AfZaWDMMNIqD_zgf6S1JUu1_aAs8p4KLZRcPM9bVmoW5qSewemIsLOmu7_l56ANLi7w&fref=pb&paipv=0',
 'type': 'love'}
```

```
# # save
# path='../Data/Fixed/ClashOfClans_Reactions_Fixed'
# df.to_csv(path + '.csv', index=False)

# arr = np.array(df)
# np.save(path + ".npy", arr)

# np.save(path + "_cols.npy", cols)
```

III.3 - Xử lí data *comments* thô

Cũng như 2 bước ở trên, chúng ta sẽ kiểm tra phần comments của dữ liệu

```
df.info()
```

✓ 0.0s Python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 352 entries, 0 to 351
Data columns (total 53 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   post_id                         352 non-null    object
 1   text                            352 non-null    object
 2   post_text                       352 non-null    object
 3   shared_text                     339 non-null    object
 4   original_text                   0 non-null      object
 5   time                            352 non-null    datetime64[ns]
 6   timestamp                       348 non-null    float64
 7   image                           172 non-null    object
 8   image_lowquality                352 non-null    object
 9   images                          352 non-null    object
 10  images_description              352 non-null    object
 11  images_lowquality               352 non-null    object
 12  images_lowquality_description   352 non-null    object
 13  video                           133 non-null    object
 14  video_duration_seconds          0 non-null      object
 15  video_height                    0 non-null      object
 16  video_id                        133 non-null    object
 17  video_quality                   0 non-null      object
 18  video_size_MB                   0 non-null      object
 19  video_thumbnail                 133 non-null    object
...
 51  videos                          4 non-null      object
 52  header                          44 non-null     object
dtypes: bool(3), datetime64[ns](2), float64(1), int64(4), object(43)
memory usage: 138.7+ KB
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

```
df['post_id'].duplicated().sum()
```

✓ 0.0s Python

```
49
```

```
df.comments.mean()
```

✓ 0.0s Python

```
561.3465346534654
```

Dữ liệu của chúng ta bị trùng lặp khá là nhiều (lên đến 14%) do page có lượng comments khá lớn nên việc crawl dữ liệu có thể dẫn đến tài khoản bị khóa và việc crawl nhiều lần trở lên rất cần thiết nên sẽ có một số bài viết bị lặp lại.

Bước này chỉ đơn giản là loại bỏ trùng lặp và rồi lưu trữ lại dữ liệu.

```
df.drop_duplicates(subset='post_id', inplace=True)
```

✓ 0.0s

```
df.shape
```

✓ 0.0s

```
(303, 53)
```

```
# # save
# path='../Data/Fixed/ClashOfClans_Comments_Fixed'
# df.to_csv(path + '.csv', index=False)

# post_np_full = np.array(df)
# np.save(path + ".npy", post_np_full)

# cols = np.array(df.columns)
# np.save(path + "_cols.npy", cols)
```

✓ 0.0s

III.4 - Gộp và xử lí data tổng hợp

Chuẩn bị dataframe tổng hợp

Ở bước này, chúng ta cần tổng hợp các data đã tiền xử lí sơ bộ ở trên để có một data hoàn chỉnh.

```
import pandas as pd
import numpy as np
```
✓ 0.4s Python

```
comments = np.load('../Data/Fixed/ClashOfClans_Comments_Fixed.npy', allow_pickle=True)
cmt_cols = np.load('../Data/Fixed/ClashOfClans_Comments_Fixed_cols.npy', allow_pickle=True)

reactions = np.load('../Data/Fixed/ClashOfClans_Reactions_Fixed.npy', allow_pickle=True)
rct_cols = np.load('../Data/Fixed/ClashOfClans_Reactions_Fixed_cols.npy', allow_pickle=True)

reactors = np.load('../Data/Fixed/ClashOfClans_Reactors_Fixed.npy', allow_pickle=True)
rtr_cols = np.load('../Data/Fixed/ClashOfClans_Reactors_Fixed_cols.npy', allow_pickle=True)
```
✓ 0.4s Python

```
print('Comments: ', comments.shape)
print('Reactions: ', reactions.shape)
print('Reactors: ', reactors.shape)
```
✓ 0.0s Python

```
Comments:  (303, 53)
Reactions:  (500, 51)
Reactors:  (300, 51)
```

```
df = pd.DataFrame.from_records(comments, columns=cmt_cols)
reactions = pd.DataFrame.from_records(reactions, columns=rct_cols)
reactors = pd.DataFrame.from_records(reactors, columns=rtr_cols)
```
✓ 0.0s Python

```
df['reactions'] = reactions['reactions']
df['reaction_count'] = reactions['reaction_count']
df['reactors'] = reactors['reactors']
```
✓ 0.0s Python

Có thể thấy là comments data có số bản ghi gần như ít nhất và cùng với việc kiểm tra trong file csv rằng *comments* ít bị lỗi ngôn ngữ nhất thì chúng ta quyết định dùng *comments* data làm data chuẩn.

Vì vậy, ta sao chép các trường cần thiết của 2 bản data còn lại vào df để có data tổng hợp.

Xử lí data

Nhìn chung thì data có 53 trường và 303 bản ghi nhưng đây là data còn chưa được xử lí hoàn toàn.

```
df.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 53 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   post_id                         303 non-null    object
 1   text                            303 non-null    object
 2   post_text                       303 non-null    object
 3   shared_text                     293 non-null    object
 4   original_text                   0 non-null      object
 5   time                            303 non-null    datetime64[ns]
 6   timestamp                       303 non-null    float64
 7   image                           147 non-null    object
 8   image_lowquality                303 non-null    object
 9   images                          303 non-null    object
 10  images_description              303 non-null    object
 11  images_lowquality               303 non-null    object
 12  images_lowquality_description   303 non-null    object
 13  video                           113 non-null    object
 14  video_duration_seconds          0 non-null      object
 15  video_height                    0 non-null      object
 16  video_id                        113 non-null    object
 17  video_quality                   0 non-null      object
 18  video_size_MB                   0 non-null      object
 19  video_thumbnail                 113 non-null    object
...
 51  videos                          3 non-null      object
 52  header                          36 non-null     object
dtypes: bool(3), datetime64[ns](2), float64(1), int64(4), object(43)
memory usage: 119.4+ KB
```

Số lượng dữ liệu bị missing khá là nhiều. Chỉ có 28 trường có số lượng missing dưới 50%, trong đó reactors có tới 135 bản ghi missing data (do fanpage có lượng reactions khá lớn nên khả năng crawl dữ liệu bị thiếu cũng cao).

```
check = df.isna().sum()
```
✓ 0.0s

```
check.loc[check < 150]
```
✓ 0.0s

```
post_id                          0
text                             0
post_text                        0
shared_text                     10
time                             0
timestamp                        0
image_lowquality                 0
images                           0
images_description               0
images_lowquality                0
images_lowquality_description    0
likes                            0
comments                         0
shares                           0
post_url                         0
links                            0
user_id                          0
username                         0
user_url                         0
is_live                          0
available                        0
comments_full                    0
reactors                       135
reactions                       11
reaction_count                   0
page_id                          0
image_ids                        0
was_live                         0
dtype: int64
```

```
check.loc[check < 150].shape
```
✓ 0.0s

```
(28,)
```

Do đó chúng ta sẽ lọc bỏ data bị thiếu theo một số trường cần quan tâm sau

```
df = df.dropna(subset=['post_id', 'time', 'timestamp',
                       'post_text', 'shares', 'comments',
                       'comments_full', 'reaction_count', 'reactions', 'reactors'], axis=0)
```
✓ 0.0s

```
df.shape
```
✓ 0.0s

```
(167, 53)
```

Lượng data còn lại chỉ bằng khoảng một nửa so với data thu thập được. Con số 167 bản ghi không quá nhiều nhưng cũng có thể dùng để phân tích một cách đa dạng.

```
# # save
# path='../Data/Final/ClashOfClans'
# df.to_csv(path + '.csv', index=False)

# post_np_full = np.array(df)
# np.save(path + ".npy", post_np_full)

# cols = np.array(df.columns)
# np.save(path + "_cols.npy", cols)
```
✓ 0.6s

Quá trình tiền xử lí khá là dài dòng tuy nhiên nó giúp chúng ta có được data "sạch" để có thể phân tích và xây dựng mô hình mà không gặp phải vấn đề về dữ liệu.

IV - Phân tích và trực quan hóa dữ liệu

Để có thể phân tích và biểu thị dữ liệu, ở đây chúng ta sẽ sử dụng thư viện seaborn và các thư viện thiết yếu đó là pandas và matplotlib.

IV.1 – Tổng quan

Sau khi kiểm tra, dữ liệu của chúng ta có rất nhiều trường, việc sử dụng tất cả các trường là điều không cần thiết. Vì thế, chúng ta sẽ chỉ chọn một số trường hữu ích cho việc phân tích của mình.

```
# %pip install matplotlib pandas numpy seaborn
```
✓ 0.0s Python

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```
✓ 1.2s Python

+ Code + Markdown

```
cols = np.load('../Data/Final/ClashOfClans_cols.npy', allow_pickle=True)
arr = np.load('../Data/Final/ClashOfClans.npy', allow_pickle=True)
df = pd.DataFrame(arr, columns=cols)
```
✓ 0.2s Python

```
df.columns
```
✓ 0.0s Python

```
Index(['post_id', 'text', 'post_text', 'shared_text', 'original_text', 'time',
       'timestamp', 'image', 'image_lowquality', 'images',
       'images_description', 'images_lowquality',
       'images_lowquality_description', 'video', 'video_duration_seconds',
       'video_height', 'video_id', 'video_quality', 'video_size_MB',
       'video_thumbnail', 'video_watches', 'video_width', 'likes', 'comments',
       'shares', 'post_url', 'link', 'links', 'user_id', 'username',
       'user_url', 'is_live', 'factcheck', 'shared_post_id', 'shared_time',
       'shared_user_id', 'shared_username', 'shared_post_url', 'available',
       'comments_full', 'reactors', 'w3_fb_url', 'reactions', 'reaction_count',
       'with', 'page_id', 'sharers', 'image_id', 'image_ids', 'was_live',
       'video_ids', 'videos', 'header'],
      dtype='object')
```

```
df = df[['post_id', 'time', 'post_text',
         'images_description', 'shares', 'comments',
            'comments_full', 'reaction_count',
            'reactions', 'reactors']]
```
✓ 0.0s Python

Trong đó:

- post-id: id của bài viết.
- time: thời gian đăng tải bài viết.
- post-text: nội dung của bài viết.
- images-description: mô tả hình ảnh của bài viết.
- shares: số lượt chia sẻ.
- comments: số lượt bình luận.
- comments-full: thông tin chi tiết của từng bình luận.
- reaction-count: số lượt bày tỏ cảm xúc.
- reactors: người bày tỏ cảm xúc.

Nhìn chung, data của chúng ta có đầy đủ dữ liệu với 1 trường thời gian còn lại là object. Tuy nhiên, theo thực tế thì *shares*, *comments*, *reaction_count* phải ở dạng số.

```
df.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   post_id             167 non-null    object
 1   time                167 non-null    datetime64[ns]
 2   post_text           167 non-null    object
 3   images_description  167 non-null    object
 4   shares              167 non-null    object
 5   comments            167 non-null    object
 6   comments_full       167 non-null    object
 7   reaction_count      167 non-null    object
 8   reactions           167 non-null    object
 9   reactors            167 non-null    object
dtypes: datetime64[ns](1), object(9)
memory usage: 13.2+ KB
```

Vậy nên, chúng ta cần chuyển kiểu dữ liệu của chúng sang dạng số và thực hiện thống kê data sử dụng phương thức describe() của thư viện pandas.

```
df[['shares', 'comments', 'reaction_count']] = df[['shares', 'comments', 'reaction_count']].astype(int)
df.describe()
```

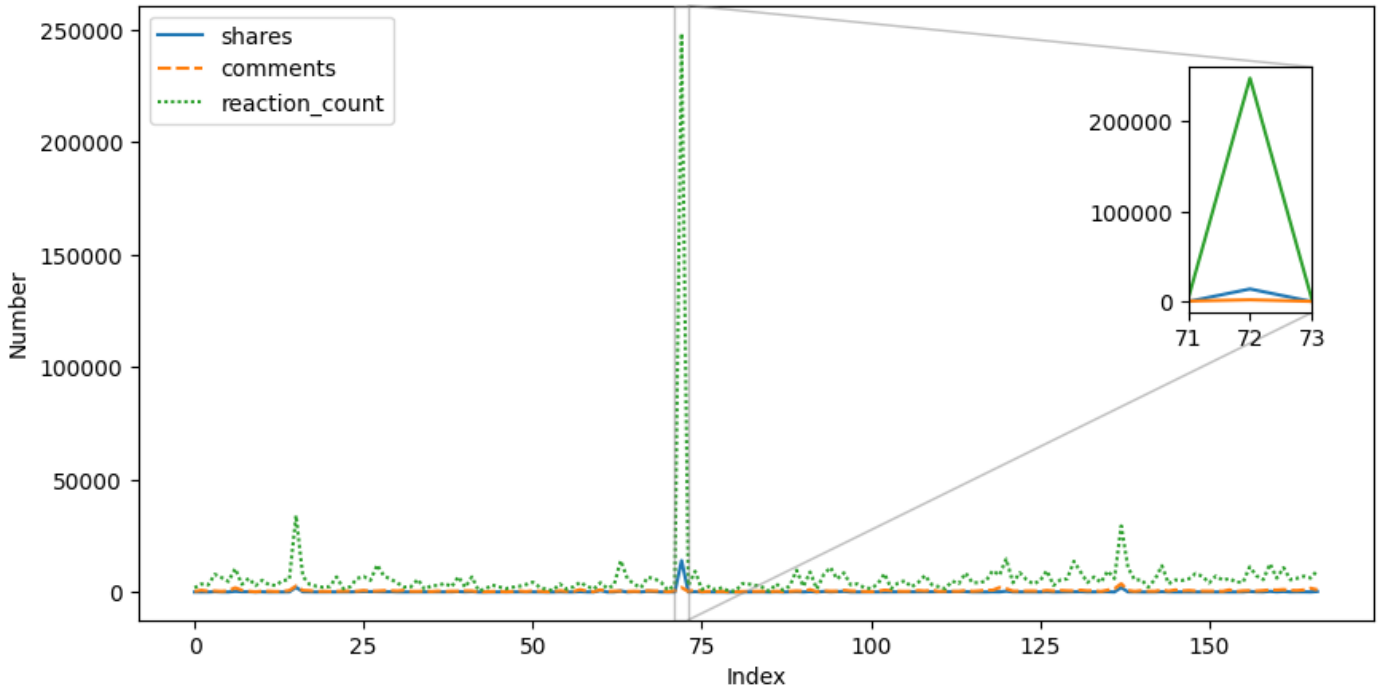
✓ 0.0s

	time	shares	comments	reaction_count
count	167	167.000000	167.000000	167.000000
mean	2023-07-10 06:43:55.670658816	187.215569	543.634731	6608.401198
min	2022-12-12 17:57:11	3.000000	31.000000	358.000000
25%	2023-04-24 05:01:35	29.000000	299.000000	2547.000000
50%	2023-08-15 15:00:49	42.000000	429.000000	4234.000000
75%	2023-09-30 09:07:58	101.000000	672.000000	6631.000000
max	2023-11-07 20:15:02	14000.000000	3700.000000	248302.000000
std	NaN	1100.436557	456.975789	19276.037022

Theo như thống kê trên, số lượt chia sẻ, bình luận và bày tỏ cảm xúc trung bình (mean) khá là cao, cho thấy rằng Clash Of Clans có lượt tương tác không nhỏ.

Tuy nhiên, nhìn vào min, max và std (độ lệch chuẩn so với mean), chúng ta cần phải xem xét lại. Chênh lệch giữa min với max cùng std thể hiện sự bất thường của dữ liệu tại một vài điểm nào đó.

Quan sát biểu đồ sau đây để có một cái nhìn trực quan hơn.



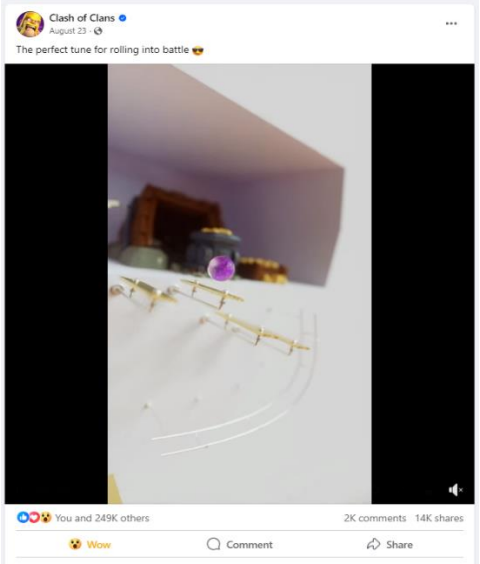
Dữ liệu tăng đột biến tại vị trí thứ 72, hãy xem bài viết này như thế nào mà lại có lượng tương tác ‘khủng’ đến vậy.

```
print('url = https://www.facebook.com/ClashofClans/posts/' + df['post_id'].iloc[72])
```

✓ 0.0s

```
url = https://www.facebook.com/ClashofClans/posts/849314546550406
```

https://www.facebook.com/ClashofClans/posts/849314546550406



Bài viết này là một video khá thú vị với nhạc nền có giai điệu êm tai và vui nhộn cùng với video rất thu hút. Theo như tựa đề “The perfect tune rolling into battle” (giai điệu hoàn hảo để lao vào cuộc chiến) thì đây là bản nhạc nền dành cho việc tìm kiếm trận đấu của game.

IV.2 – Nội dung bài viết

Đầu tiên, hãy truy xuất xem chúng ta có những dữ liệu gì.

```
df_text = df[['post_id','post_text', 'images_description']]
print(df_text.info())
df_text
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 3 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   post_id             167 non-null    object
 1   post_text           167 non-null    object
 2   images_description  167 non-null    object
dtypes: object(3)
memory usage: 4.0+ KB
None
```

	post_id	post_text	images_description
0	890116715803522	Books of Clash Vol 2 is on sale! 🎉 Get it now ...	[May be an image of text]
1	889593269189200	edit: the issues have been fixed, thanks for y...	[]
2	889556962526164	Hey Chief! We've got a brief maintenance happe...	[May be an image of text that says 'MAINTENANC...
3	888646695950524	Before Barcher, Hog Wizard, Witch Golem and La...	[May be an image of text, May be an image of t...
4	888172002664660		[May be an image of 1 person]
...	...	...	...
162	6340880625936207	Hey Chiefs! We've got some balance changes we'...	[No photo description available.]
163	6337372302953706	Season's greetings from your friendly neighbor...	[No photo description available.]
164	6323016887722581	Get yourself into the Clashmas mood with some ...	[No photo description available.]
165	6319757081381895	Hey Chief! We'll take a maintenance break in ~...	[No photo description available.]
166	6311032642254339	UPDATE DAY is here! 😍 Maintenance will start s...	[No photo description available.]

167 rows × 3 columns

Cột post_text chứa nội dung (captions) của bài viết, còn cột images_description là mô tả về những bức ảnh có trong bài (do máy tính tính toán). Do đó, chúng ta có thể bỏ qua phần description và tập chung phân tích phần nội dung.

Để phân tích nội dung, ta sẽ sử dụng thư viện wordcloud. Đây là một thư viện hỗ trợ cho việc xử lí ngôn ngữ của Python. *Tài liệu được cung cấp tại*: https://github.com/amueller/word_cloud.

Bức ảnh trên được sinh từ Wordcloud, cho ta cái nhìn trực quan về những từ và mức độ xuất hiện của nó. Có thể thấy nổi bật là ‘Clash’, ‘Clan’, ‘Clan Games’, ‘Hey Chief’,… là những từ ngữ xuất hiện nhiều trong các

bài viết của trang. Đó thường là những từ liên quan đến trò chơi như những sự kiện mới, vật phẩm mới, những thông báo hay cũng có thể là cách mà nhà phát hành gọi người chơi.

Phần nội dung bài viết không có nhiều điều có thể phân tích hoặc đi sâu, phân tích ở trên có thể cho thấy phần nào về những chủ đề mà nội dung bài viết hướng tới.

IV.3 - Phân tích *reactions*

IV.3.1 - Reactions

Đầu tiên, hãy kiểm tra xem data của chúng ta có những cột nào liên quan đến cảm xúc của bài viết.

```
df.columns
```
✓ 0.0s

```
Index(['post_id', 'time', 'post_text', 'images_description', 'shares',
       'comments', 'comments_full', 'reaction_count', 'reactions', 'reactors'],
      dtype='object')
```

Có thể thấy đó là '*reaction_count*', '*reactions*' và '*reactors*'. Kế đến, tạo một dataframe để phân tích các phần dữ liệu này.

```
reactions = pd.DataFrame(df[['reaction_count', 'reactions', 'reactors']])
```
✓ 0.0s

Tiếp tới, hãy xem một số bản ghi của các cột trên có những gì.

	reaction_count	reactions	reactors
0	1876	{'like': 1470, 'love': 353, 'haha': 12, 'wow':...	[{'name': 'Di Mo', 'link': 'https://facebook.c...
1	3570	{'like': 2790, 'love': 524, 'haha': 185, 'wow'...	[{'name': 'Di Mo', 'link': 'https://facebook.c...
2	3162	{'like': 2520, 'love': 538, 'haha': 29, 'wow':...	[{'name': 'Di Mo', 'link': 'https://facebook.c...
3	7860	{'like': 5406, 'love': 2209, 'haha': 30, 'wow'...	[{'name': 'Di Mo', 'link': 'https://facebook.c...
4	6389	{'like': 4458, 'love': 1772, 'haha': 11, 'wow'...	[{'name': 'Di Mo', 'link': 'https://facebook.c...

Có thể thấy, các bản ghi trong cột 'reactions' có kiểu dữ liệu là một dictionary. Chúng ta sẽ tách nó ra thành các cột từng loại cảm xúc như sau:

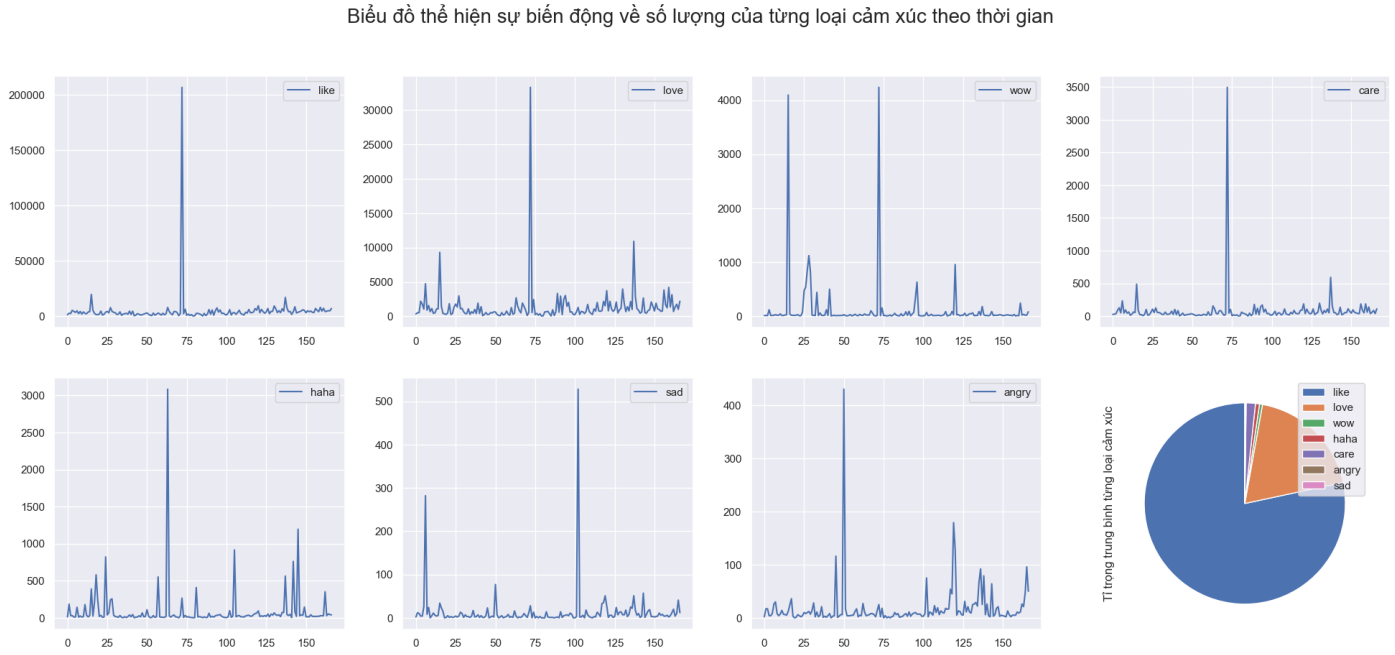
```
reactions['like'] = reactions['reactions'].map(lambda x: x['like'])
reactions['haha'] = reactions['reactions'].map(lambda x: x['haha'])
reactions['love'] = reactions['reactions'].map(lambda x: x['love'])
reactions['wow'] = reactions['reactions'].map(lambda x: x['wow'])
reactions['sad'] = reactions['reactions'].map(lambda x: x['sad'])
reactions['angry'] = reactions['reactions'].map(lambda x: x['angry'])
reactions['care'] = reactions['reactions'].map(lambda x: x['care'])
```
✓ 0.0s

```
reactions[['like', 'love', 'wow', 'haha', 'care', 'angry', 'sad']].describe()
```
✓ 0.0s

	like	love	wow	haha	care	angry	sad
count	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000	167.000000
mean	4907.514970	1382.167665	113.341317	91.071856	83.808383	17.526946	12.970060
std	15936.195342	2819.438079	477.783913	286.438466	274.421142	39.996508	46.737208
min	277.000000	56.000000	1.000000	0.000000	3.000000	0.000000	0.000000
25%	1947.000000	466.500000	10.000000	12.500000	26.500000	4.000000	2.000000
50%	3258.000000	780.000000	15.000000	23.000000	45.000000	7.000000	4.000000
75%	4691.500000	1476.500000	32.500000	41.500000	85.000000	17.000000	10.500000
max	206958.000000	33290.000000	4240.000000	3085.000000	3495.000000	430.000000	528.000000

Thực hiện thống kê các loại cảm xúc, có thể thấy 'like' chiếm số lượng chủ yếu trong cảm xúc của bài viết, tiếp đến là 'love'. Còn xếp cuối cùng đó là 'angry' và 'sad'.

Để có cái nhìn trực quan hơn, chúng ta sẽ biểu thị đường biểu diễn của chúng thông qua sns.lineplot().



Như biểu diễn ở trên, dựa vào biểu đồ thể hiện tỉ trọng trung bình từng loại cảm xúc thì có thể xác nhận được nhận xét về tỉ trọng đã nói ở trên. Đồng thời, theo như biểu đồ đường phát triển của mỗi loại, có thể thấy từng loại cảm xúc có xu hướng không ổn định, cùng với đó là những điểm có số lượng cao đột biến. Do đó, chúng ta cần xem xét chúng là những bài viết như thế nào để đưa ra kết luận cụ thể.

Thứ nhất, về LIKE, LOVE và CARE, hai loại cảm xúc này đều cùng có điểm đột biến tại bài viết có id là 849314546550406. Đây là điểm dị thường mà ta đã tìm hiểu ở phần IV.1.

Thứ hai, nhìn vào đồ thị có thể thấy WOW có 02 điểm dị thường với số lượng bày tỏ trên 4000 và có thông tin như sau:

```
reactions.loc[reactions['wow'] > 4000]
```
✓ 0.0s

	post_id	time	reaction_count	reactions	reactors	like	haha	love	wow	sad	angry	care
15	882200249928502	2023-10-23 22:00:07	34074	{'like': 19748, 'love': 9303, 'haha': 390, 'wo...	[{'name': 'Noorullah Ramazani', 'link': 'https...	19748	390	9303	4097	34	12	490
72	849314546550406	2023-08-23 23:48:43	248302	{'like': 206958, 'love': 33290, 'haha': 266, '...	[{'name': 'Mubasher Ali', 'link': 'https://fac...	206958	266	33290	4240	28	25	3495

Cũng như ở trên, chúng ta sẽ bỏ qua bài viết số 72 và tập trung nhận xét bài viết số 15 có id 882200249928502.

Bài viết này có nội dung nói đến việc phát hành phiên bản game dành cho máy tính – điều mà trước đây chưa có. Vì vậy, rất nhiều người chơi cảm thấy ngạc nhiên khi biết điều này cho nên số lượt WOW đã có một bước nhảy vọt lớn.

https://www.facebook.com/ClashOfClans/posts/882200249928502

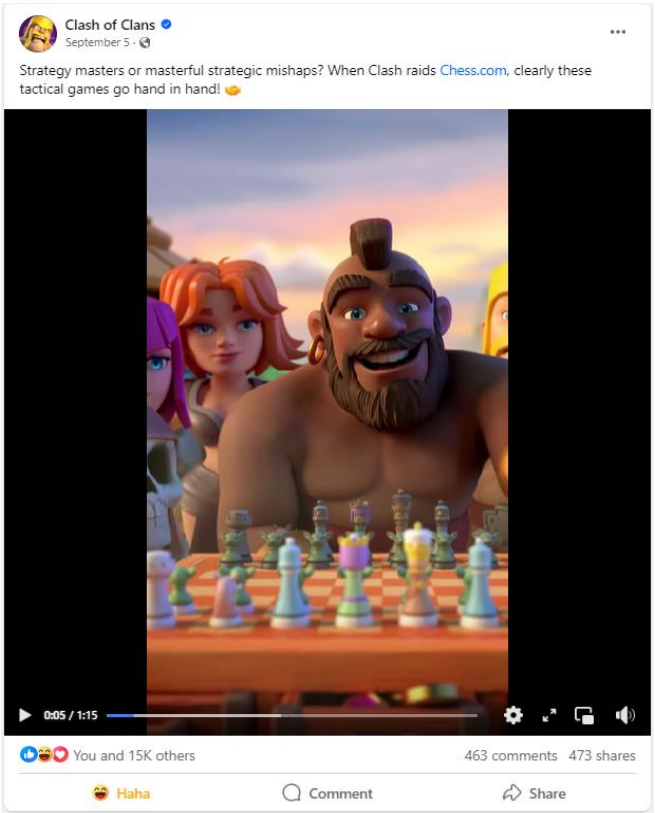
Tiếp theo, bản ghi chứa giá trị đột biết của HAHA có thông tin như dưới đây:

```
reactions.loc[reactions['haha'] > 3000]
```

0.0s

	post_id	time	reaction_count	reactions	reactors	like	haha	love	wow	sad	angry	care
63	855786275903233	2023-09-05 19:10:54	14024	{'like': 8023, 'love': 2704, 'haha': 3085, 'wo...	[{'name': 'Ke Shara', 'link': 'https://faceboo...	8023	3085	2704	40	4	11	157

Có thể thấy rằng cảm xúc này không đạt điểm đột biến tại bài viết thứ 72 như các loại trên mà lại xuất hiện tại bài số 63.



https://www.facebook.com/ClashOfClans/posts/855786275903233

Đây là một clip hài hước về chơi cờ vua giữa các nhân vật (quân lính) trong game. Đồng thời quảng bá về việc hợp tác với Chess.com.

Thứ tư, điểm nhảy vọt của ANGRY có tại 02 bài viết dưới đây:

https://www.facebook.com/clashOfClans/posts/864489661699561 https://www.facebook.com/clashOfClans/posts/6804789822878616

Hai bài viết trên có nội dung về phần thưởng cho sự kiện và việc bảo trì trò chơi. Chắc hẳn ai cũng cảm thấy khó chịu khi phần thưởng không được như mong đợi hoặc phải chờ cho tới khi việc bảo trì kết thúc để có thể quay lại trò chơi.

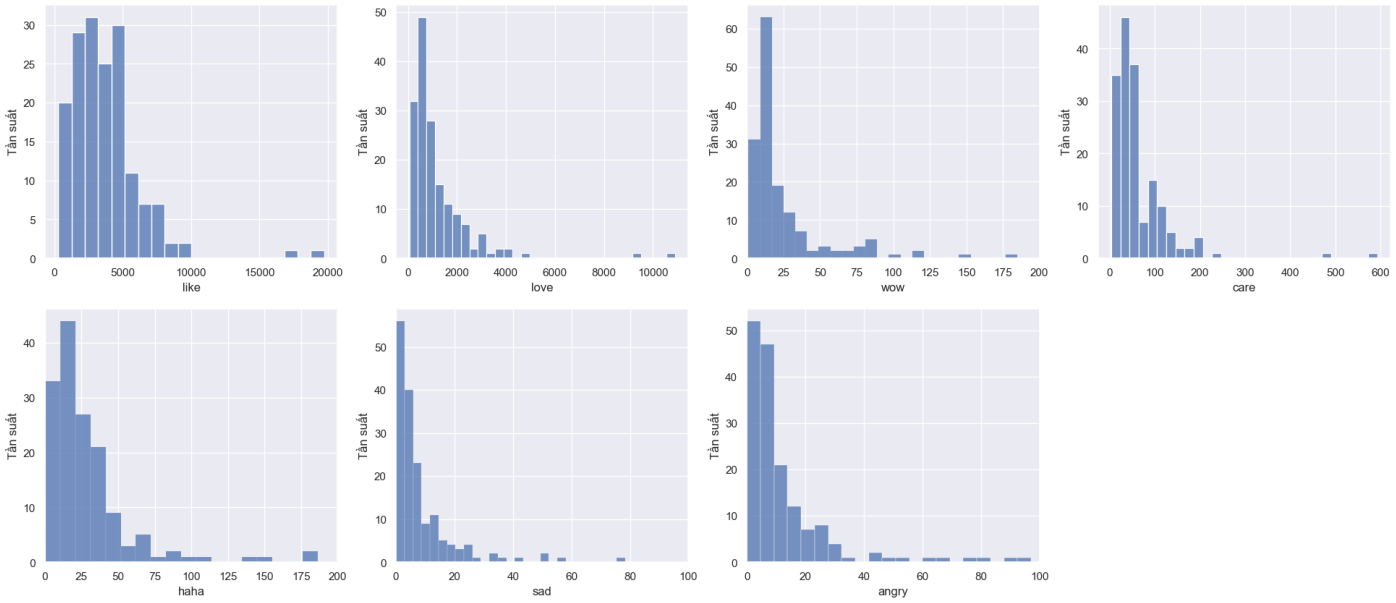
Cuối cùng là SAD, từ biểu đồ đường thì các điểm cao bất thường là ở bài viết về việc xóa bỏ nhân vật được thêm trong sự kiện Halloween và một bài viết về thưởng sự kiện.

https://www.facebook.com/clashOfClans/posts/887129712768889 https://www.facebook.com/clashOfClans/posts/832161388265722

⇨ **Như vậy, theo như những phân tích trên, có thể kết luận rằng những cảm xúc được bày tỏ đạt những điểm cao đột biến phụ thuộc vào nội dung của bài viết. Có thể là về việc cập nhật mới, một clip thú vị, một thông báo nào đó của nhà phát hành.**

Để theo dõi phân bố số lượng của các cảm xúc, chúng ta sẽ quan sát biểu đồ sau (Lưu ý: Điều cần thiết là trước đó ta cần loại bỏ điểm dị thường đã nói tại phần IV.1).

Biểu đồ thể hiện phân bố theo số lượng của từng loại cảm xúc

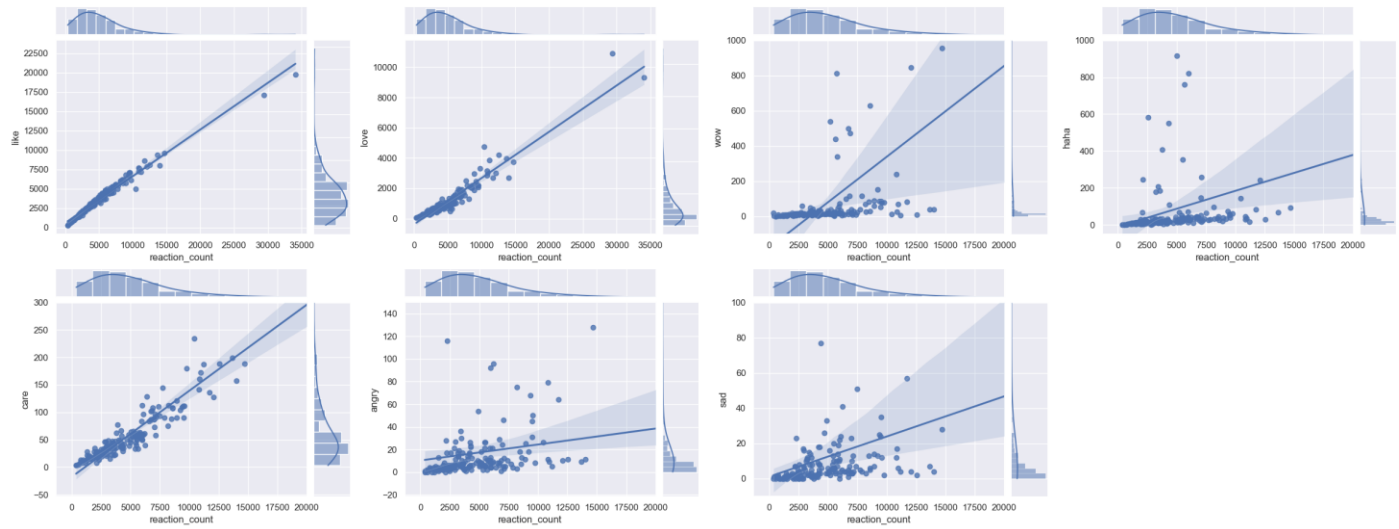


Ta có thể kết luận một vài điều về phân bổ số lượng những cảm xúc như sau:

- LIKE: chủ yếu trong khoảng 1000 đến 6000.
- LOVE: chủ yếu trong khoảng 400 đến 1600.
- WOW: khoảng 0 đến 23.
- CARE: từ 20 đến 60.
- HAHA: khoảng từ 0 đến 45.
- SAD: chủ yếu nằm ở khoảng 0 tới 10.
- ANGRY: khoảng từ 0 đến 20.

Tiếp sau đây, chúng ta sẽ quan sát biểu đồ thể hiện phân bố của các loại reactions theo tổng số lượng reactions của các bài viết. Điều cần thiết là trước đó ta cần loại bỏ điểm dị thường đã nói tại phần IV.1.

Biểu đồ thể hiện phân bố của từng loại reaction theo tổng số reactions



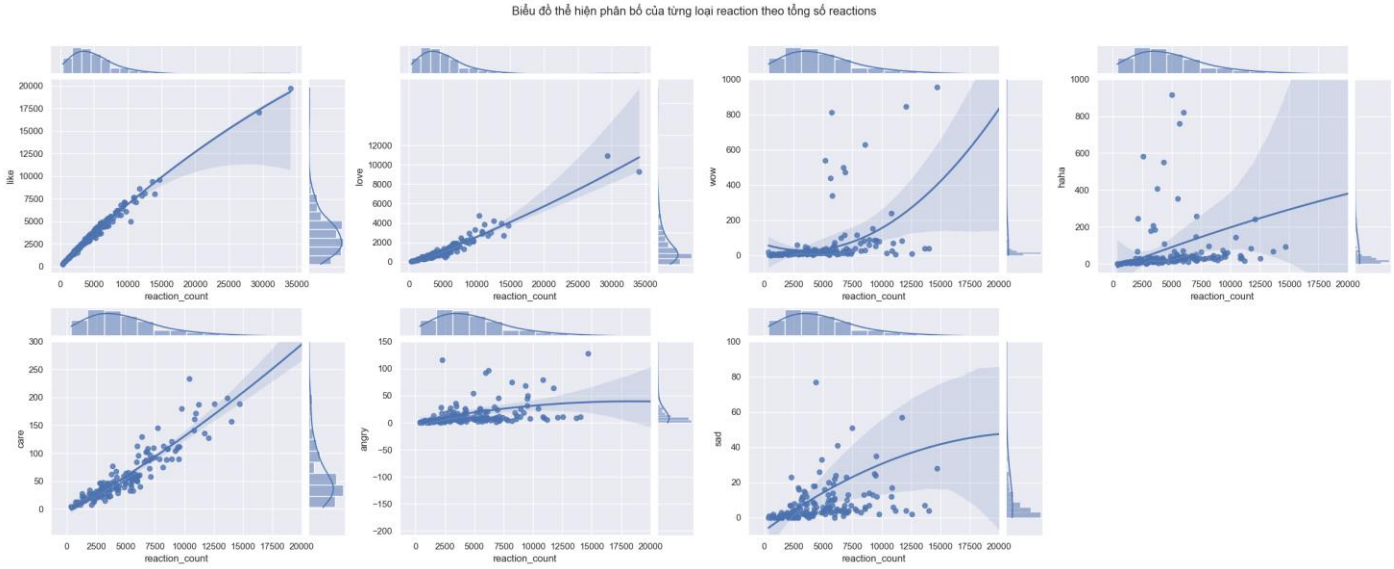
Để xây dựng được biểu đồ trên, ta cần sử dụng seaborn.jointplot() ngoài ra còn có SeabornFig2Grid (source-code và cách sử dụng chi tiết được tại đây:

https://stackoverflow.com/questions/35042255/how-to-plot-multiple-seaborn-jointplot-in-subplot).

Theo như biểu đồ trên, các cảm xúc đều có đường hồi quy thể hiện xu hướng tăng theo số lượng cảm xúc tổng thể. Tuy nhiên từng loại cảm xúc lại có những phân bố khác nhau.

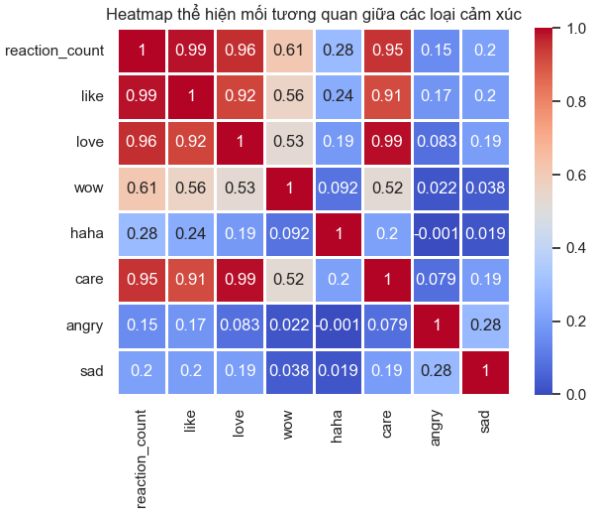
Về phần LIKE, LOVE và CARE, chúng có phân bố khá ổn định (chiếm số lượng chủ yếu) cùng đường hồi quy gần đúng nhất. Các cảm xúc còn lại có phân bố khá hỗn loạn (chủ yếu chiếm số lượng nhỏ) bởi vậy đường hồi quy của những loại cảm xúc này không được tốt cho lắm.

Có thể xét đến được hồi quy bậc 2 có biểu đồ dưới đây.



Đường hồi quy có vẻ 'gần' hơn một chút với dữ liệu của chúng ta.

Thêm nữa, sự tương quan của chúng được biểu diễn như sau.



IV.3.2 - Reactors

Trong mỗi bài viết sẽ có danh sách những người bày tỏ cảm xúc. Mỗi người sẽ có thông tin sau.

```
{'name': 'Di Mo',
 'link': 'https://facebook.com/profile.php?id=100095717490800&eav=AfZ2FUoxX2PgxTn4XSgGbIHR2X-tXfBztxof2m03YtIgziPK73GhoQWXLSTGzdHEYhY&fref=pb&paipv=0',
 'type': 'like'}
```

Có tên, đường dẫn đến trang cá nhân và cảm xúc đã bày tỏ.

Vì số lượng người bày tỏ là rất lớn, nên ta sẽ **không** phân tích quá sâu vào phần này mà chỉ phân tích sơ lược như trên và dành việc phân tích người tương tác cho phần phía sau.

Như vậy, qua một vài phân tích ở trên, ta đã có được những nhận xét và kết luận về một số vấn đề liên quan đến reaction.

IV.4 – Comments và Shares

Đầu tiên, hãy kiểm tra thông tin và thống kê comments và shares trong dữ liệu.

```
comments.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   post_id        167 non-null    object
 1   time           167 non-null    datetime64[ns]
 2   comments       167 non-null    int32
 3   comments_full  167 non-null    object
 4   shares         167 non-null    int32
dtypes: datetime64[ns](1), int32(2), object(2)
memory usage: 5.3+ KB
```

```
comments.describe()
```
✓ 0.0s

	time	comments	shares
count	167	167.000000	167.000000
mean	2023-07-10 06:43:55.670658816	543.634731	187.215569
min	2022-12-12 17:57:11	31.000000	3.000000
25%	2023-04-24 05:01:35	299.000000	29.000000
50%	2023-08-15 15:00:49	429.000000	42.000000
75%	2023-09-30 09:07:58	672.000000	101.000000
max	2023-11-07 20:15:02	3700.000000	14000.000000
std	NaN	456.975789	1100.436557

Theo như thống kê thì *comments* có số lượng chủ yếu trong khoảng 400 đến 700 trên mỗi bài viết, còn lượng *shares* thì thấp hơn, chủ yếu trong khoảng 40 đến 100 mỗi bài.

Tiếp theo, hãy xem qua một phần đầu của data về *comments* và *shares*.

```
comments.head()
```
✓ 0.1s Python

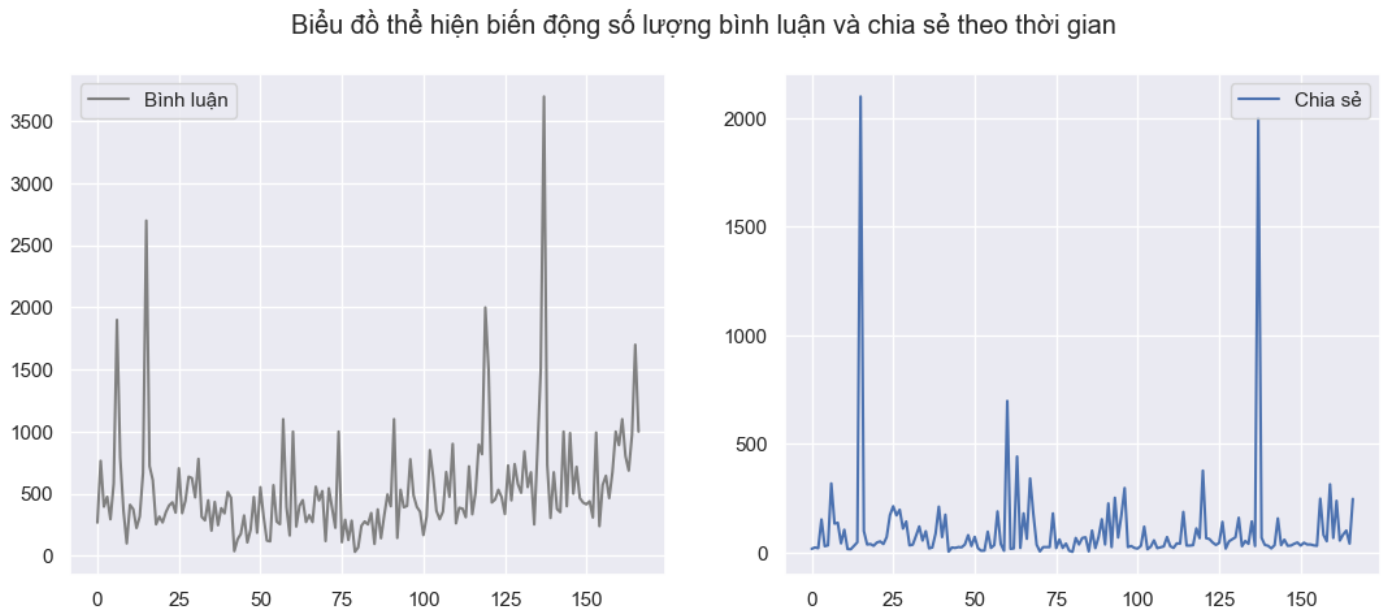
	post_id	time	comments	comments_full	shares
0	890116715803522	2023-11-07 20:15:02	267	[{'comment_id': '1483156222461521', 'comment_u...	17
1	889593269189200	2023-11-06 19:15:55	764	[{'comment_id': '358059689925658', 'comment_ur...	23
2	889556962526164	2023-11-06 17:36:19	395	[{'comment_id': '1259825128044288', 'comment_u...	20
3	888646695950524	2023-11-04 19:54:48	475	[{'comment_id': '285704867769829', 'comment_ur...	152
4	888172002664660	2023-11-03 20:35:02	294	[{'comment_id': '592877136242752', 'comment_ur...	29

```
comments['comments_full'].iloc[0]
```
✓ 0.0s Python

```
[{'comment_id': '1483156222461521',
  'comment_url': 'https://facebook.com/1483156222461521',
  'commenter_id': '1547348997',
  'commenter_url': 'https://facebook.com/candi.warren?eav=AfbiNpmqys47oOcpozaaZ3-1fujQk0SS2bcFyRNYACc39Nx-5mGmYaUdXYJPithtcxw&fref=nf&rc=p&refid=52&__tn__=R&paipv=0',
  'commenter_name': 'Candi Warren',
  'commenter_meta': None,
  'comment_text': 'I'm maxed out on everything…. Can you make it so I can gift resources to my own clan. Otherwise, I'm a sitting duck for attacks… I still want to enjoy the game, some how',
  'comment_time': datetime.datetime(2023, 11, 8, 0, 0),
  'comment_image': None,
  'comment_reactors': [],
  'comment_reactions': None,
  'comment_reaction_count': None,
  'replies': [{'comment_id': '649800960690490',
    'comment_url': 'https://facebook.com/649800960690490',
    'commenter_id': '100004529382881',
    'commenter_url': 'https://facebook.com/adam.genez?eav=AfZCK-UleIF6a4cv3xxoeTM-QMa7sxHrV0MqIqNTvkhGl1tPKBfSBcfhKucQ5IYcP6I&fref=nf&rc=p&__tn__=R&paipv=0',
    'commenter_name': 'Adam Genez',
    'commenter_meta': None,
    'comment_text': 'Candi Warren this is a ruled out idea for 2 reasons:\n\n1: people will create a bunch of "mini" accounts and pool their resources onto their main account\n\n2: people will create black markets and attempt to sell
    'comment_time': datetime.datetime(2023, 11, 8, 0, 0),
    'comment_image': 'https://scontent.fhan20-1.fna.fbcdn.net/m1/v/t6/An_UvxJXg9tdnLU3Y5qjPi0200MLilhzPXUgxzGjQzUMaNcmjdZA6anyrngvkdub33NZzZhd51fpCAEzNHFhkoSaKRFP5fS1w_lKwYrzcNLupv27.png?ccb=10-5&oh=00_AfDL0NFvPS37sJzbMJ0FK3sxzAPxovKg
    'comment_reactors': [],
    'comment_reactions': None,
    'comment_reaction_count': None}]},
 {'comment_id': '317532330989943',
...
  'comment_image': None,
  'comment_reactors': [],
  'comment_reactions': None,
```

Có thể thấy, phần thông tin về bình luận của bài viết được crawl về khá là chi tiết, có trường id comment, đường dẫn đến comment, nội dung, hình ảnh, người comment,…. Đây là nguồn thông tin hữu ích cho các phân tích tiếp theo.

Theo dõi đường phát triển của số lượng bình luận và chia sẻ, ta có được biểu đồ như sau:



Biểu đồ cho thấy số lượng mỗi loại biến động liên tục nhưng vẫn xê dịch quanh một khoảng nhất định. Tuy nhiên, có một vài điểm nhảy vọt khác thường.

```
comments.loc[(comments['comments'] > 1500) | (comments['shares'] > 500)]
```

✓ 0.1s

	post_id	time	comments	comments_full	shares
6	887129712768889	2023-11-01 22:32:35	1900	[{'comment_id': '1746006649181843', 'comment_u...	318
15	882200249928502	2023-10-23 22:00:07	2700	[{'comment_id': '1015783102805700', 'comment_u...	2100
60	859122382236289	2023-09-11 19:49:05	1000	[{'comment_id': '793930129176016', 'comment_ur...	698
119	6804789822878616	2023-05-16 14:41:30	2000	[{'comment_id': '958144218755939', 'comment_ur...	66
137	6586857294671871	2023-03-09 00:14:04	3700	[{'comment_id': '895985071731568', 'comment_ur...	2000
165	6319757081381895	2022-12-14 19:40:13	1700	[{'comment_id': '1345743539529532', 'comment_u...	41

Hãy lấy bài số 137 làm đại diện. Bài viết có chủ đề chúc mừng ngày quốc tế phụ nữ. Hình ảnh của bài đều là những nữ nhân vật có trong trò chơi. Do đó, bài viết thu hút khá nhiều lượt hưởng ứng cả về bình luận, chia sẻ cũng như bày tỏ cảm xúc.

https://www.facebook.com/clashofClans/posts/6586857294671871

Bức ảnh ngay trên mô tả các từ phổ biến trong nội dung bình luận của người tương tác. Chủ yếu là những câu từ liên quan đến trò chơi như game, clan, clash, troop, pekka, defense, heroes,…. Có thể thấy người chơi chủ yếu nói về những vấn đề xoay quanh trò chơi này.

IV.4.1 – Commenters

Hãy tìm hiểu thông tin về những người bình luận của fanpage.

IV.4.1.1 – Thu thập dữ liệu

Chúng ta sẽ thu thập id của những người bình luận có trong các bài viết, sau đó sử dụng get_profile của facebook-scraper để lấy dữ liệu về. Do vấn đề của framework (hoặc có thể do người dùng không công khai), nên một số thông tin cần thiết không được thu thập.

```
from facebook_scraper import get_profile
```

```
commenter = []
```

```
max, index = 0, 0
cookie_path = '../config/cookies.txt'
for i in range(0, len(id_cmters)):
    try:
        profile = get_profile(id_cmters[i], cookies=cookie_path)
        if len(profile.keys()) > max:
            max = len(profile.keys())
            index = i
        commenter.append(profile)
        print(f"Added profile no.{len(commenter)}")
        if len(commenter) % 100 == 0:
            # To df
            post_df_full = pd.DataFrame(columns=commenter[7].keys(), index=range(len(commenter)), data=commenter)

            path='../Data/Cmters/Commenterss'
            # post_df_full.to_csv(path + '.csv')
    except Exception as e:
        # To df
        if len(commenter) > 0:
            post_df_full = pd.DataFrame(columns=commenter[7].keys(), index=range(len(commenter)), data=commenter)

            path='../Data/Cmters/Commenterss'
            # post_df_full.to_csv(path + '.csv')
```

IV.4.1.2 – Tiền xử lí

Đầu tiên, loại bỏ trùng lặp và những người thiếu thông tin.

```python
post_df_full.drop_duplicates(subset=['id'], inplace=True)
post_df_full.dropna(subset=['Places lived', 'Relationship'], inplace=True)
```

Tiếp theo, trích xuất địa điểm và định vị quốc gia (sử dụng geopy và googletrans).

```python
from geopy.geocoders import Nominatim
from googletrans import Translator
```

```python
geolocator = Nominatim(user_agent="crawl_page")

def country(place):
    place = place.split(',')[-1].strip()
    try:
        location = geolocator.geocode(place, language='en')
        return location.address.split(',')[-1].strip()
    except:
        try:
            translator = Translator()
            location = geolocator.geocode(translator.translate(place, dest='en').text, language='en')
            return country(place)
        except:
            print("Error at", place)
        return 1

print(country("Thanh hoa"))

def extract_location(place):
    if place is np.nan:
        return None
    if type(place) is not list:
        return None
    if 'text' not in place[0].keys():
        return None
    return place[0]['text']

post_df_full['location'] = post_df_full['Places lived'].apply(lambda x: extract_location(x))

# Get country of location
post_df_full['country'] = post_df_full['location'].apply(lambda x: country(x) if x else None)
```

```
Vietnam
```

Sau đó, xác định châu lục dựa vào quốc gia của người dùng.

```python
% pip install pycountry_convert
```

```python
import pycountry_convert as pc
```

```python
def continent(country_name):
    country_alpha2 = pc.country_name_to_country_alpha2(country_name)
    country_continent_code = pc.country_alpha2_to_continent_code(country_alpha2)
    country_continent_name = pc.convert_continent_code_to_continent_name(country_continent_code)
    return country_continent_name

post_df_full['continent'] = post_df_full['country'].apply(lambda x: continent(x) if x else None)
```

Cuối cùng là xử lí tình trạng quan hệ của người bình luận.

```python
def rel(x):
    if type(x) is dict:
        return x['type']
    return x
```

```python
post_df_full['rel'] = post_df_full['Relationship'].apply(lambda x: rel(x) if x else None)
```

```python
def f(x):
    if 'since' in x and '\n' in x:
        return x[x.index('\n') + 1 : x.index('since')].strip()
    if 'since' in x:
        return x[: x.index('since')].strip()
    if '\n' in x:
        return x[x.index('\n') + 1 :].strip()
    return x
```

```python
post_df_full['relationship'] = post_df_full['rel'].apply(lambda x: f(x) if x else None)
```

IV.4.1.3 – Phân tích

Sau một vài bước tiền xử lí, ta sẽ thu được data về commenters có thông tin như sau:

```
commenters_df.info()
```
0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 132 entries, 0 to 131
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               132 non-null    int64
 1   Name             132 non-null    object
 2   Friend_count     47 non-null     float64
 3   Follower_count   53 non-null     float64
 4   Following_count  11 non-null     float64
 5   location         132 non-null    object
 6   country          132 non-null    object
 7   continent        132 non-null    object
 8   relationship     122 non-null    object
dtypes: float64(3), int64(1), object(5)
memory usage: 9.4+ KB
```

Ta sẽ tập trung phân tích 3 trường cuối cùng: *country*, *continent*, *relationship*. Tuy nhiên, trước đó hãy xem phần đầu của dữ liệu có những thông tin gì.

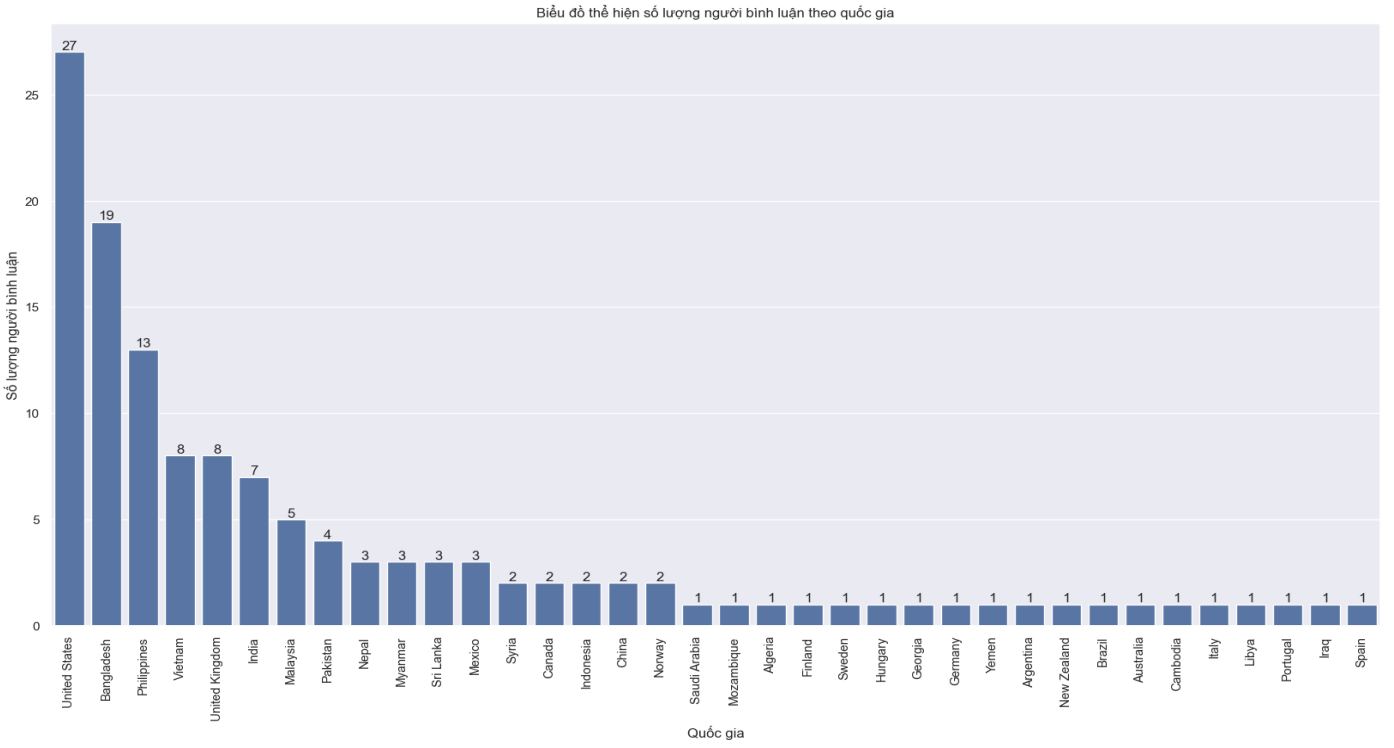
```
commenters_df.head()
```
0.0s

	id	Name	Friend_count	Follower_count	Following_count	location	country	continent	relationship
0	100051419944069	Huy Anh	NaN	12838.0	178.0	Luc Ngan	Vietnam	Asia	Single
1	100004094897649	Trịnh Trung	NaN	2360.0	371.0	Thanh Hóa	Vietnam	Asia	Single
2	100090428256818	Minh Thiệnn	264.0	2.0	218.0	Can Tho	Vietnam	Asia	Single
3	100024020196271	Tă M Ĩm	NaN	NaN	NaN	Khulna	Bangladesh	Asia	Single
4	100002797914077	Muhamad Bukhari	NaN	NaN	NaN	Penang, Malaysia	Malaysia	Asia	Single

Country

```
print(commenters_df['country'].value_counts().shape)
commenters_df['country'].value_counts().plot(kind='bar', figsize=(20, 10))
```
0.3s

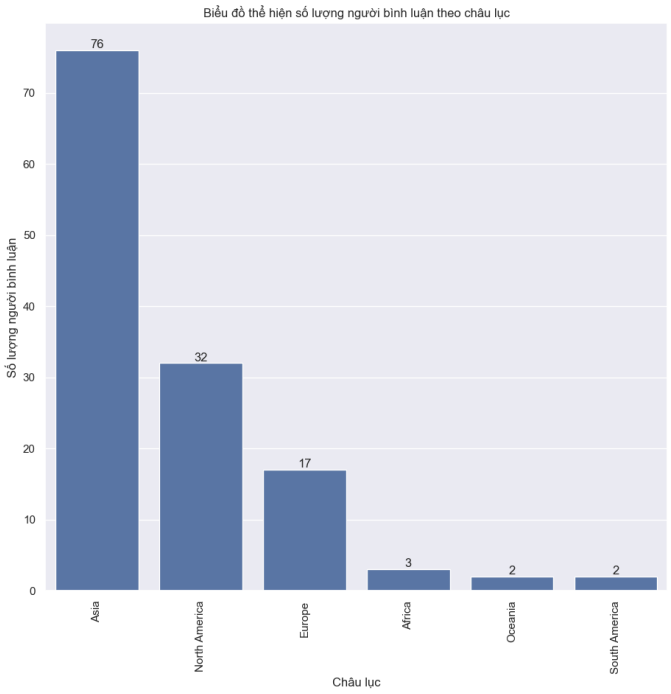
```
(36,)
```

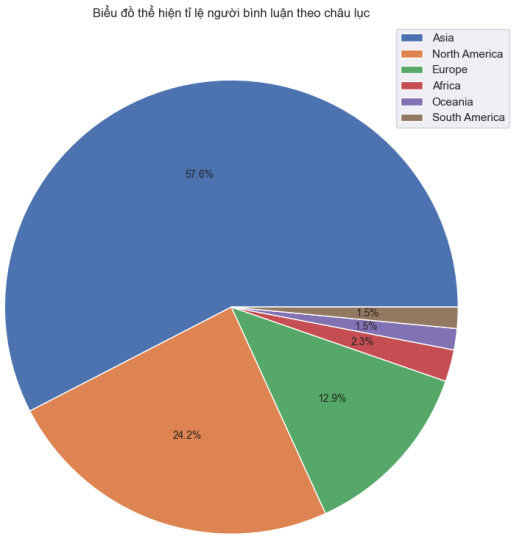


Người bình luận đến từ rất nhiều quốc gia (trong file dữ liệu ta có 36 nước).

Căn cứ theo biểu đồ thể hiện số lượng người bình luận theo quốc gia, Mĩ có số lượng nhiều nhất, sau đó là Bangladesh. Việt Nam chỉ xếp thứ tư trong danh sách trên.

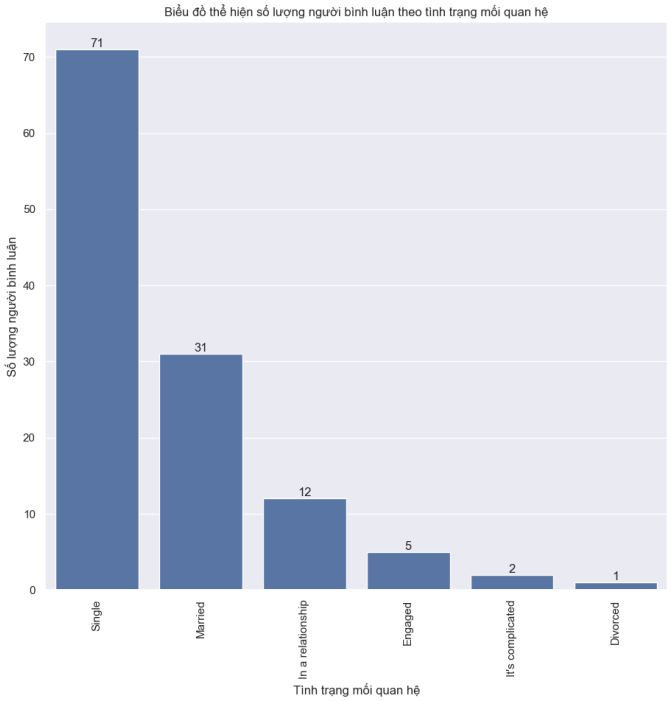
Continent

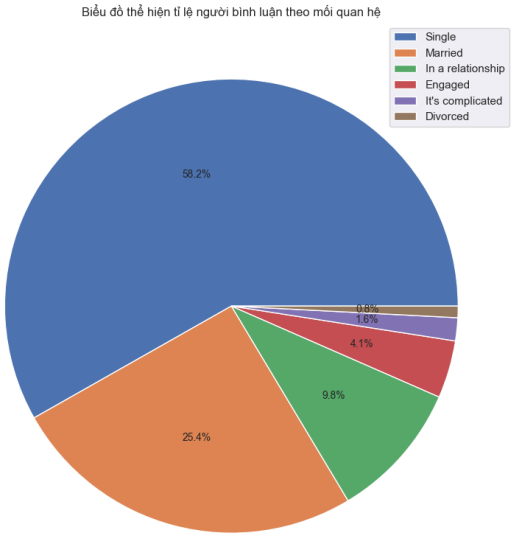




Tuy Mĩ dẫn đầu về số lượng người chơi theo quốc gia, nhưng châu Á mới là nơi đứng đầu về lượng người theo châu lục (57,6%). Châu Mĩ chỉ chiếm 26,5% tính cả Bắc Mĩ và Nam Mĩ.

Relationship



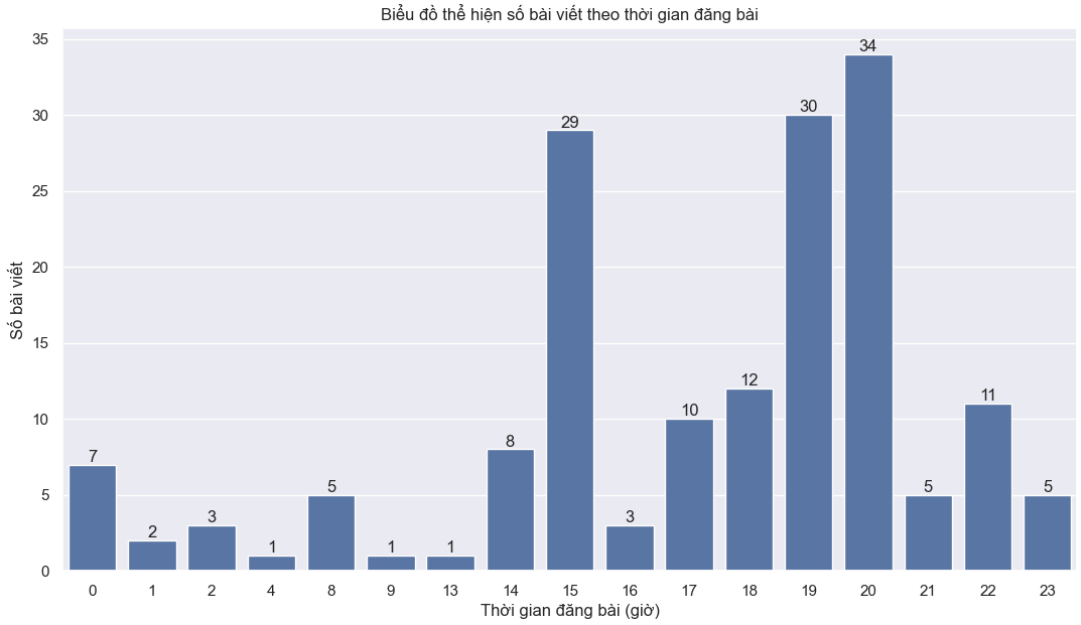


Phần lớn người chơi đều độc thân, xếp tiếp theo là số lượng người đã kết hôn. Để đưa ra kết luận chính xác so với thực tế là điều khó khăn bởi có thể do người dùng đặt sai tình trạng quan hệ nhưng dựa trên dữ liệu chúng ta có thì người bình luận độc thân chiếm đa số, có thể nhận xét rằng họ trong độ tuổi thanh niên dựa vào tình trạng quan hệ này.

IV.5 – Các mối quan hệ, sự tương quan giữa các thông tin có trong dữ liệu.

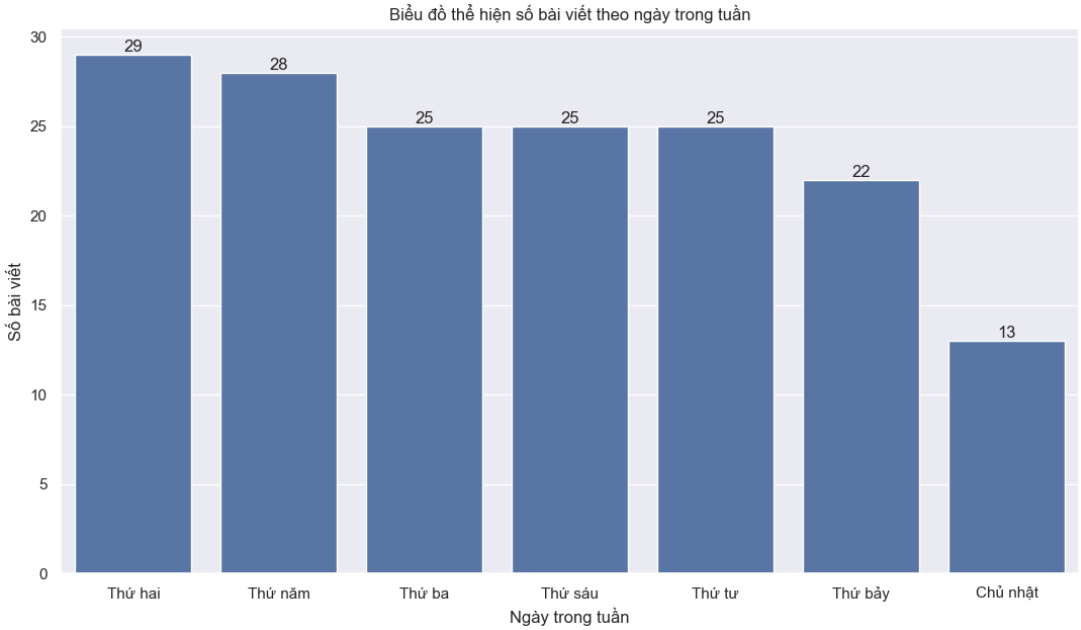
IV.5.1 – Mối liên kết giữa số lượng bài viết theo thời gian

IV.5.1.1 – Theo giờ trong ngày



Sự chênh lệch được thể hiện rất rõ ràng thông qua biểu đồ cột được vẽ ở trên. Thời gian đăng bài thường xuyên của 'Clash of Clans' rơi vào 15h, 19h và 20h trong ngày. Rất ít khi đăng bài vào khoảng trưa (9h đến 13h).

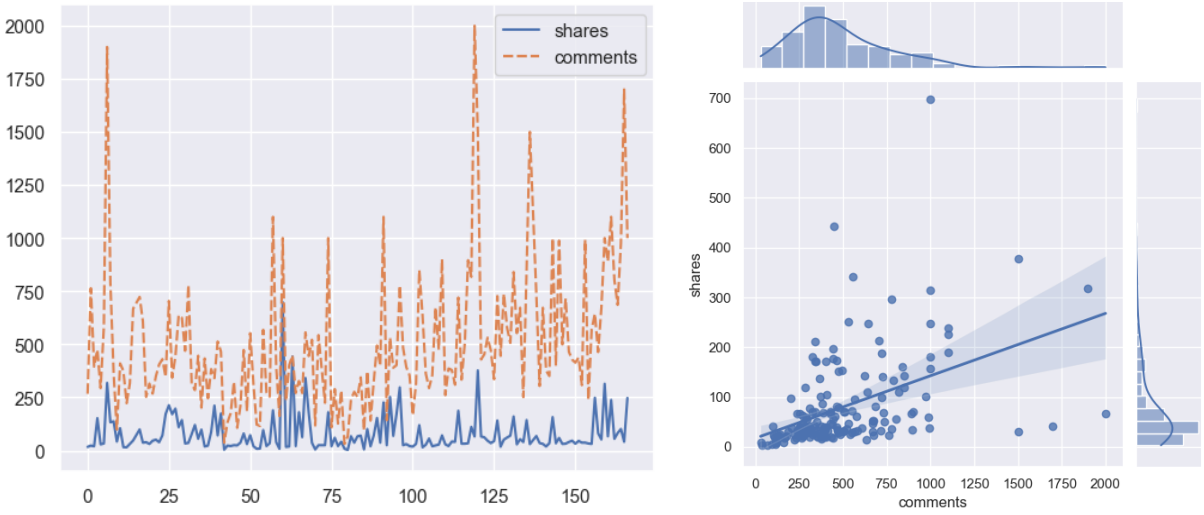
IV.5.1.2 – Theo ngày trong tuần



Dựa vào biểu đồ trên, có thể nhận xét rằng các bài viết được phân bố khá đều theo các ngày trong tuần, trừ Chủ nhật có số lượng bài thấp hơn hẳn. Nguyên do có thể là vì đó là ngày nghỉ nên lượng bài sẽ ít đi.

Chú ý: Chúng ta sẽ **không** phân tích sự phụ thuộc của comments, reactions và shares vào thời gian. Bởi vì, 3 đại lượng trên không có mối liên hệ với thời gian, hay có thể nói là độc lập với thời gian.

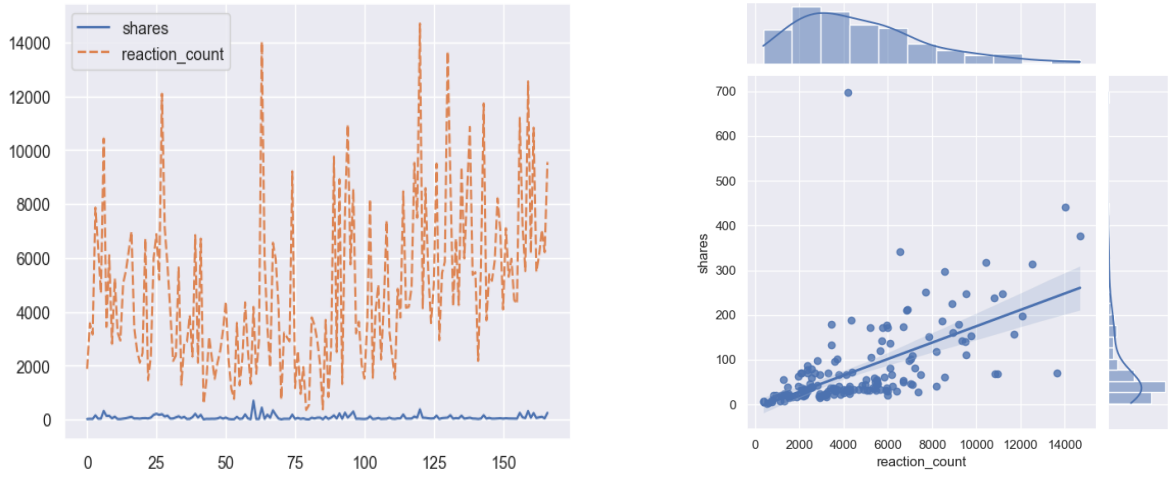
IV.5.2 – Mối liên hệ giữa comments và shares



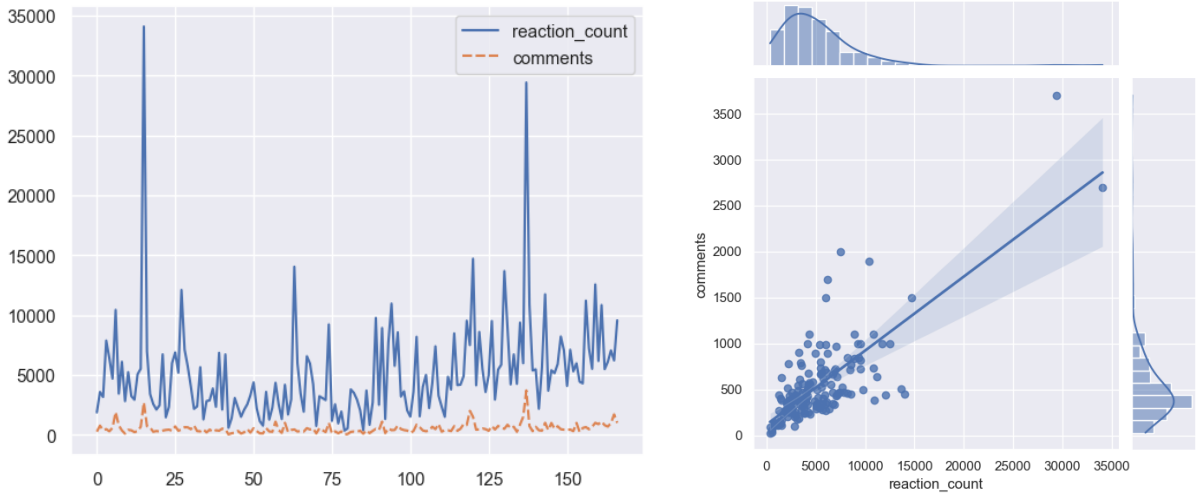
Đường hồi quy tuyến tính bắt rất gần với dữ liệu mà ta có. Tuy nhiên, dường như giữa bình luận và chia sẻ có mối liên hệ không chặt chẽ cho lắm. Thật vậy, hệ số tương quan mà chúng ta tính được chỉ là 0.444 cho thấy chúng không đồng nhất.

IV.5.2 – Mối liên hệ giữa reactions và shares

Hai đại lượng này có hệ số tương quan khá hơn một chút, vào khoảng 0.592 với biểu diễn sau đây:



IV.5.2 – Mối liên hệ giữa reactions và comments



Hệ số tương quan giữa reactions và comments cao hơn hẳn (rơi vào khoảng 0.761). Đồng thời, phân bố của chúng có biểu diễn bớt hỗn loạn hơn.

V - Tổng kết

Sau khi thực hiện những công việc trên, sinh viên đã có thể có những kĩ năng sau:

- **Kĩ năng thu thập dữ liệu**: biết cách sử dụng framework, sửa chữa những lỗi sai trong quá trình crawl dữ liệu, lưu trữ dữ liệu.
- **Kĩ năng tiền xử lí và làm sạch dữ liệu**: biết cách xử lí trùng lặp, xử lí thiếu sót data, xử lí những lỗi có trong dữ liệu
- **Kĩ năng phân tích và trực quan hóa dữ liệu**: biết cách thống kê, xây dựng các biểu đồ thể hiện dữ liệu, tìm ra những đặc điểm của dữ liệu và đưa ra những nhận xét, kết luận từ những biểu đồ hoặc bảng thống kê đó.

VI - Định hướng tương lai

1. **Xây dựng mô hình Học máy**: Xây dựng mô hình để dự đoán dữ liệu.
2. **Xây dựng Dashboard**: Tạo trang web có dash để thao tác giao diện với dữ liệu một cách sinh động.
3. **Xử lí chuỗi thời gian**: tìm hiểu thêm về chuỗi thời gian trong mô hình.

Github project: https://github.com/longluv1605/final-project-Python-DataAnalyst

---------- **HẾT** ----------