

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ - ĐHQGHN
VIỆN TRÍ TUỆ NHÂN TẠO



LẬP TRÌNH XỬ LÝ DỮ LIỆU
BÁO CÁO BÀI TẬP LỚN

NHÓM 7: ĐÁNH ĐÁU LỖ ĐÓ

Phân tích bài viết trên diễn đàn chứng khoán FireAnt và tương quan của diễn đàn với thị trường chứng khoán Việt Nam

Thành viên nhóm: Phạm Nhật Quang 23020413
Bùi Minh Quân 23020415
Phan Quang Trường 23020443

Lời mở đầu

Đường dẫn

- GitHub Source code: [Link GitHub](#)
- Dữ liệu phân tích: [Link Google Drive](#)

Lời đầu tiên, nhóm tác giả chúng em xin gửi lời cảm ơn tới thầy cô trong Viện Trí tuệ nhân tạo, vì đã tạo điều kiện cho chúng em thử sức với nhiều điều mới mẻ.

Ngày nay, thị trường chứng khoán Việt Nam là nơi diễn ra các hoạt động mua bán cổ phiếu của các công ty, nó đóng vai trò quan trọng trong việc huy động vốn cho nền kinh tế. Nó còn là kênh đầu tư, đầu cơ vô cùng hấp dẫn để gia tăng thu nhập. Thị trường đã phát triển nhanh chóng, thu hút ngày càng nhiều nhà đầu tư trong và ngoài nước.

FireAnt (<https://fireant.vn/>) là một trong những diễn đàn lớn tại Việt Nam dành cho các nhà đầu tư. Đây là nơi chia sẻ thông tin phong phú về thị trường, thu hút nhiều thành phần khác nhau từ các nhà đầu tư cá nhân đến các chuyên gia. Sự phổ biến của **FireAnt** làm tăng mức độ ảnh hưởng của nó lên thị trường chứng khoán Việt Nam.

Có thể nói, các diễn đàn như **FireAnt** có tác động mạnh đến tâm lý của các nhà đầu tư, từ đó ảnh hưởng đến quyết định mua bán cổ phiếu. Việc nghiên cứu tác động của diễn đàn có thể giúp hiểu rõ hơn về những yếu tố tâm lý chi phối thị trường.

Trong bản báo cáo này, chúng em sẽ tiến hành khai thác và phân tích dữ liệu từ **FireAnt** cũng như thị trường chứng khoán Việt Nam theo từng phần sau:

1. Cách thức thu thập dữ liệu từ **FireAnt**, dữ liệu từ các Công ty chứng khoán, cùng phương án xử lý và làm sạch dữ liệu;
2. Những phân tích về **FireAnt**: khai thác, phân tích và đào sâu vào nội dung đăng tải của người dùng trên nền tảng, bao gồm các bài đăng, bình luận;
3. Những phân tích về thị trường chứng khoán Việt Nam;
4. Liên hệ, xây dựng giả thuyết, tìm tương quan giữa diễn đàn **FireAnt** với diễn biến của thị trường chứng khoán Việt Nam;
5. Mô hình: Sử dụng Định luật Bayes, Xích Markov để Dự đoán giá dựa trên thống kê bài đăng; Mô hình Học máy Phân tích Quan điểm dựa trên nội dung bài viết;
6. Kết luận những thông tin, góc nhìn đã được đưa ra từ những phân tích trên.

Chúng em hi vọng những phát hiện và nghiên cứu của mình sẽ khẳng định được những dự đoán, tầm nhìn của chúng em về tâm lý thị trường chứng khoán Việt Nam. Qua đó, có thể ứng dụng vào nhiều khía cạnh trọng điểm trong thực tế.

Mục lục

Lời mở đầu, đường dẫn	1
1 Thu thập và Tiền xử lý dữ liệu	4
1.1 Phương án thu thập	4
1.1.1 Dữ liệu FireAnt	4
1.1.2 Dữ liệu Chứng khoán	5
1.1.3 Công cụ	5
1.2 Quá trình thu thập	5
1.2.1 Dữ liệu FireAnt	5
1.2.2 Dữ liệu Chứng khoán	10
1.3 Tiền xử lý, làm sạch dữ liệu	12
2 Phân tích Dữ liệu Diễn đàn FireAnt	17
2.1 Tinh chỉnh Dữ liệu để phân tích	17
2.2 Phân tích dữ liệu bài đăng	19
2.2.1 Số bài đăng theo ngày	19
2.2.2 Số bài đăng theo ngày trong tuần và giờ	19
2.2.3 Phân bố Quan điểm người dùng	20
2.2.4 Wordcloud bài đăng	20
2.2.5 Số lần xuất hiện của link	21
2.2.6 Quan điểm người dùng khi chèn đường link	22
2.3 Phân tích phản hồi/bình luận	23
2.3.1 Số phản hồi theo ngày	23
2.3.2 Số phản hồi theo ngày trong tuần và giờ	24
2.3.3 Wordcloud phản hồi	25
2.4 Phân tích các mã cổ phiếu được đề cập	26
2.4.1 Quan điểm của người dùng	26
2.4.2 So sánh số bài đăng đề cập đến của các mã theo ngày	27
2.4.3 Xếp hạng các mã cổ phiếu được đề cập nhiều nhất	28
2.5 Phân tích Cộng đồng người dùng FireAnt	29
2.5.1 Thống kê cơ bản	29
2.5.2 Thống kê các hoạt động nổi bật của người dùng	32
2.5.3 Phân tích Văn hóa ứng xử của người dùng	35
3 Phân tích Dữ liệu Thị trường Chứng khoán Việt Nam	38
3.1 Phân tích các mã cổ phiếu theo ngành	38
3.1.1 Phân bổ giữa các ngành	38
3.1.2 Vốn hóa thị trường, số cổ phiếu lưu hành	39
3.1.3 Quan điểm người dùng	40
3.2 Phân tích Tương quan	41
3.2.1 Tương quan trong Thị trường Nội địa	41
3.2.2 Tương quan với Thị trường Quốc tế	42
4 Phân tích Tương quan giữa FireAnt và Thị trường Chứng khoán Việt Nam	43
4.1 Khối lượng giao dịch cổ phiếu với Số lượng bài viết	43
4.2 Xu hướng của chỉ số VNINDEX và Hoạt động trên diễn đàn FireAnt	44

5 Mô hình	48
5.1 Mô hình Dự đoán giá trong ngày sử dụng Định luật Bayes	48
5.1.1 Chuẩn bị dữ liệu tính toán	48
5.1.2 Áp dụng Định luật Bayes	49
5.1.3 Kiểm chứng Phương pháp	50
5.2 Mô hình Dự đoán giá Phiên kế tiếp sử dụng Xích Markov	53
5.2.1 Chuẩn bị dữ liệu tính toán	54
5.2.2 Áp dụng Xích Markov tính xác suất chuyển đổi trạng thái	54
5.2.3 Kiểm chứng Phương pháp	55
5.3 Mô hình Học máy Phân tích Quan điểm dựa trên Nội dung Bài viết (Sentiment Analysis) sử dụng Random Forest	57
5.3.1 Chuẩn bị dữ liệu	57
5.3.2 Huấn luyện mô hình	59
5.3.3 Cải thiện dữ liệu đầu vào	61
5.3.4 Sử dụng thư viện imbalanced-learn	62
5.3.5 Kết luận, hướng phát triển	63
Tổng kết	64
Tài liệu tham khảo	67

Chương 1

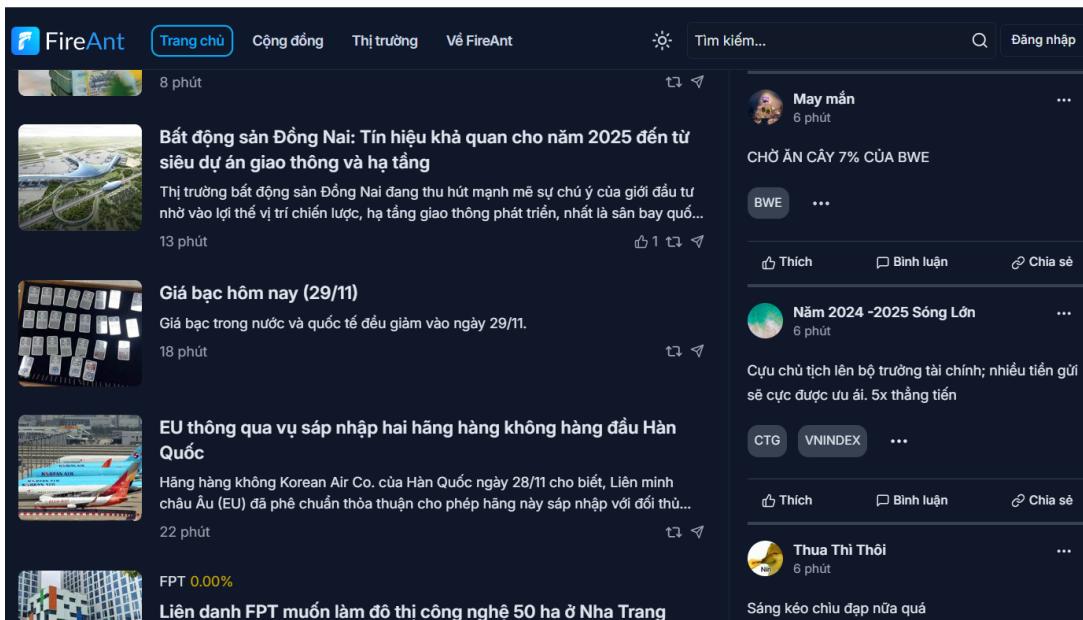
Thu thập và Tiền xử lý dữ liệu

1.1 Phương án thu thập

1.1.1 Dữ liệu FireAnt

Để thực hiện phân tích diễn đàn, ta cần cào dữ liệu là những bài đăng của người dùng lên diễn đàn đó.

FireAnt là diễn đàn đã tồn tại lâu đời, với hơn 3 triệu tài khoản đã được đăng ký [5]. Do đó, chúng ta hoàn toàn có thể kỳ vọng một lượng bài đăng cũng như tương tác ổn định ở diễn đàn này.



Hình 1.1: Trang chủ của FireAnt

Tất nhiên, do lượng dữ liệu từ toàn bộ các bài đăng của trang này là quá khổng lồ, chúng ta sẽ chỉ thu thập những bài đăng gần đây nhất. Ở đây ta hi vọng có thể thu được dữ liệu trong khoảng thời gian **hai tháng**, đặc biệt tại thời điểm thị trường có nhiều biến động.

Nền tảng FireAnt cung cấp cho người dùng một bộ API tương đối đầy đủ, có thể truy cập bằng cách gửi request vào đường dẫn <https://api.fireant.vn/>.

API này yêu cầu một Authentication Token của người dùng, tuy nhiên ta có thể tạo một tài khoản và lấy Token của tài khoản đó một cách dễ dàng.

Như vậy, công việc thu thập dữ liệu từ FireAnt bao gồm các công đoạn sau:

1. Cung cấp các thông tin cần thiết (Auth Token);

2. Tiến hành gọi request, thu thập dữ liệu từ những thông tin được cung cấp;
3. Xử lý dữ liệu, tái tổ chức và lưu trữ.

1.1.2 Dữ liệu Chứng khoán

Tương tự, để khớp với dữ liệu trong FireAnt, ta cũng cần cào dữ liệu của thị trường trong khoảng thời gian đó. Ta hi vọng có thể thu được dữ liệu của toàn bộ các mã chứng khoán, các chỉ số nổi tiếng (VNINDEX, DJI, N225, ...), trong khoảng thời gian hai tháng, trùng với khoảng thời gian bên trên.

Một vài công ty chứng khoán như TCBS, Vietcap hay các trang tin như MSN Finance có sẵn bảng giá và lịch sử giá của các mã chứng khoán, và ta có thể lấy được chúng thông qua API và dữ liệu được cào trực tiếp trong trang.

Hơn nữa, xuất hiện một số thư viện có thể gọi API cũng như cào dữ liệu một cách tự động, tiêu biểu là thư viện `vnstocks` [13].

Như vậy, công việc thu thập dữ liệu chứng khoán bao gồm:

1. Tải và cài đặt các thư viện cần thiết;
2. Lên danh sách các mã cần quan tâm;
3. Tiến hành chạy thư viện, thư viện sẽ gọi API và cào dữ liệu tương ứng;
4. Xử lý dữ liệu, tái tổ chức và lưu trữ.

1.1.3 Công cụ

Để hỗ trợ trong việc thu thập, xử lý dữ liệu, ta sử dụng những thư viện:

- `vnstocks` cung cấp công cụ cần thiết để tự động cào, trích xuất dữ liệu từ các công ty chứng khoán;
- `requests`, `selenium` cung cấp khả năng gọi API, trích xuất dữ liệu từ trang web;
- `pandas`, `numpy`, `json`, `csv` cung cấp khả năng tổ chức, biến đổi dữ liệu, đọc dữ liệu JSON, đồng thời lưu trữ dữ liệu dưới dạng file CSV.

1.2 Quá trình thu thập

1.2.1 Dữ liệu FireAnt

Dầu tiên, ta tiến hành lấy Auth Token trong một request bằng chế độ Inspect trên trình duyệt.

Name	Headers	Payload	Preview	Response	Initiator	Timing
<code>X-Aspnet-Version:</code>	4.0.50319					
<code>X-Powered-By:</code>	ASP.NET					
<code>Request Headers</code>						
<code>:authority:</code>	restv2.fireant.vn					
<code>:method:</code>	GET					
<code>:path:</code>	<code>/posts?type=0&offset=0&limit=20</code>					
<code>:scheme:</code>	https					
<code>Accept:</code>	application/json, text/plain, */*					
<code>Accept-Encoding:</code>	gzip, deflate, br, zstd					
<code>Accept-Language:</code>	en-US,en;q=0.9,vi;q=0.8					
<code>Authorization:</code>	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiL... RwczovL2FjY291bnRzLmZpcmVhbnQudm4 ZW1haWwicLJhY2NvdW50cy1yZWFlkiwiYV cmVhZCIsInN5bWJvbHMtcmVhZCIsInVzZX					

Dãy token này sẽ được sử dụng trong suốt quá trình thu thập.

```
# Dãy Authentication Bearer Token cần sử dụng để gọi API
AUTH_BEARER = 'eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6IkdYdExONzViZlZQakd
[REDACTED]

Hàm gọi API

import requests
import json

def get_request(url, params=None):
    headers = {
        'Authorization': f'Bearer {AUTH_BEARER}', # Thêm dãy token vào header
    }
    response = requests.get(url, headers=headers, params=params)
    if response.status_code != 200:
        print(f'Fail: {response.status_code}')
        return None # Trả về None nếu request thất bại
    return response.json() # Trả về dữ liệu json nếu request thành công
```

Để thực hiện cào bài viết của người dùng, ta sẽ request tới <https://api.fireant.vn/posts> cùng 3 tham số là `type`, `offset`, `limit`.

```
def get_posts(offset = 0, limit = 10):
    """
    Hàm lấy danh sách các bài viết

    type: tham số không quan trọng,
    offset: vị trí bắt đầu lấy, limit: số lượng bài viết cần lấy
    """

    url = 'https://api.fireant.vn/posts'
    params = [
        'type': 0,
        'offset': offset,
        'limit': limit,
    ]
    return get_request(url, params)
```

Để đảm bảo dữ liệu không xảy ra trùng lặp, sau mỗi lần gọi hàm, ta cần xét số bài viết đã lấy được và tính lại `offset`. Ngoài ra cần xét nếu dữ liệu nằm ngoài khoảng thời gian yêu cầu thì không lấy. Ta sẽ ghi lại dữ liệu này vào một file CSV.

```

def crawl_posts(offset = 0, limit = 1000, number_of_entries = 999999, end_date_str = '2024-09-06T00:00:00+07:00'):
    """
    Hàm cào dữ liệu bài viết từ FireAnt

    offset: vị trí bắt đầu, limit: số lượng bài viết lấy một lúc, number_of_entries: số lượng bài viết tối đa cần lấy
    end_date_str: ngày kết thúc lấy dữ liệu
    """

    result_posts_id = {}      # Dict tổng hợp ID các bài viết đã lấy, tránh trùng lặp
    count_posts = 0            # Số lượng bài viết đã lấy
    count_api_call = 0         # Số lần gọi API

    # Lặp cho đến khi lấy đủ số lượng bài viết cần lấy, hoặc hết dữ liệu
    while count_posts < number_of_entries:
        count_api_call += 1
        data = get_posts(offset, limit) # Lấy dữ liệu từ API

        if not data: # Nếu dữ liệu rỗng, thoát vòng lặp
            break

        # Nếu thời gian của bài viết cuối cùng nhỏ hơn ngày end_date, thoát vòng lặp
        # Ta chỉ lấy dữ liệu đến ngày end_date
        cur_date = dateutil.parser.isoparse(data[-1]['date']) # Thời gian của bài viết cuối cùng
        end_date = dateutil.parser.isoparse(end_date_str) # Thời gian kết thúc lấy dữ liệu
        if cur_date < end_date:
            break

        count_overlap = 0 # Số lượng bài viết trùng lặp
        result_posts = [] # Danh sách bài viết cần lấy

        for post in data:
            # Nếu bài viết đã được lấy, tăng biến đếm và bỏ qua
            if post['postID'] in result_posts_id:
                count_overlap += 1
                continue

            # Thêm ID bài viết vào danh sách đã lấy
            result_posts_id[post['postID']] = True
            result_posts.append(process_post(post))
            count_posts += 1

        # Tính lại offset cho lần lấy tiếp theo
        offset += limit + count_overlap

        # Ghi dữ liệu vào file CSV
        keys = result_posts[0].keys() # Lấy danh sách key của dữ liệu
        with open('posts.csv', 'a', newline='', encoding='utf-8') as csvfile:
            writer = csv.DictWriter(csvfile, fieldnames=keys) # Tạo writer
            if(count_api_call == 1): # Nếu là lần ghi đầu tiên, ghi header
                writer.writeheader()
            writer.writerows(result_posts) # Ghi dữ liệu vào file

    print(f'Call no. {count_api_call}, this time got: {len(result_posts)} posts, total crawled {count_posts} posts, offset: {offset}, end={end_date_str}')

```

Ta thực hiện lấy 1000 bài viết một lúc, lấy những bài viết kể từ 06/9/2024 cho tới 06/11/2024. Tổng cộng cào được 272979 bài viết.

```

# Cào dữ liệu bài viết
crawl_posts(limit=1000, end_date_str='2024-09-06T00:00:00+07:00')

```

```
Crawled 272979 posts, end of data posts, total crawled 272979 posts, offset: 280021
```

Dữ liệu trả về có dạng JSON, là một list gồm các phần tử là object đại diện cho bài viết, trong đó có 60 trường dữ liệu, nổi bật có ID bài viết, ngày đăng, thông tin người đăng, nội dung bài viết, tổng số lượt thích/chia sẻ/bình luận, mã chứng khoán được nhắc đến, tầm nhìn tích cực/tiêu cực (sentiment), link, hình ảnh, video (nếu có).

```

{
  "postID": 29520643,
  "user": {
    "id": "325fd365-ae20-4a49-b301-a0b516d338b3",
    "name": "Chứng Khoán Lướt Sóng Thần"
  },
  "originalContent": "CHÚC MỪNG CÁC BÁC THEO DŌI\n",
  "date": "2024-11-29T23:29:40.247+07:00",
  "hasImage": true,
  "hasFile": false,
  "link": null,
  "sentiment": 1,
  "totalLikes": 3,
  "totalReplies": 8,
  "totalShares": 0,
  "replyToPostID": null,
  "images": [
    {
      "imageID": 2707158,
      "base64Image": null,
      "imageUrl": null
    }
  ],
  "taggedSymbols": [
    {
      "symbol": "CMG",
      "price": 56.8,
      "change": 2.4,
      "percentChange": 4.4117647058823533,
      "changeSince": 0.0,
      "percentChangeSince": 0.0
    }
  ]
}

```

Hình 1.2: Ví dụ một bài viết

Tuy nhiên, request trên không bao gồm dữ liệu về phần bình luận của bài viết. Với mỗi một bài viết, ta cần gửi một request để lấy bình luận của bài viết đó. Đường dẫn cần gửi tới là <https://api.fireant.vn/posts/<post-id>/replies>, với <post-id> là ID bài viết. Cấu trúc request tương tự như trên.

```

def get_replies(post_id, offset = 0, limit = 10):
    """
    Hàm lấy danh sách các bình luận của một bài viết

    post_id: id của bài viết cần lấy, offset: vị trí bắt đầu lấy
    limit: số lượng bình luận cần lấy
    """

    url = f'https://api.fireant.vn/posts/{post_id}/replies'
    params = {
        'offset': offset,
        'limit': limit,
    }
    return get_request(url, params)

```

Ta cũng thực hiện gọi hàm này tương tự cách gọi hàm lấy bài viết. Do cấu trúc của bình luận và bài viết là như nhau (bình luận là một dạng bài viết), ta không cần xử lý thêm nhiều. Ta sẽ ghi lại dữ liệu này vào một file CSV khác.

Tuy nhiên, với số lượng bài viết lên tới 270 nghìn, việc gọi requests cho từng bài viết mất rất nhiều thời gian và có nguy cơ gây quá tải. Theo thống kê, **3418 bài viết có ít nhất 15 bình luận**. Vì vậy, ta chỉ cần dữ liệu bình luận trong những bài viết này. Ngoài ra, giữa các request sẽ có khoảng delay là 0.2 giây để tránh bị chặn IP.

```

def crawl_replies(offset = 0, limit = 1000, skip = 0, range_of_replies = (20, 1000)):
    """
    Hàm cào dữ liệu bình luận từ FireAnt, sử dụng danh sách bài viết `posts.csv`.

    offset: vị trí bắt đầu (bình luận), limit: số lượng bình luận lấy một lúc
    skip: số lượng bài viết ban đầu được bỏ qua
    range_of_replies: bài có số lượng bình luận nằm trong khoảng này mới được lấy
    """

    df = pd.read_csv('posts.csv') # Dataframe chính từ danh sách bài viết trong file
    count_replies = 0 # Số lượng bình luận đã lấy

    # Xử lý dữ liệu bình luận
    for post in data:
        result_replies.append(process_post(post))
        count_replies += 1

    # Ghi dữ liệu vào file CSV
    keys = result_replies[0].keys() # Lấy danh sách key của dữ liệu
    with open('replies.csv', 'a', newline='', encoding='utf-8') as csvfile:
        writer = csv.DictWriter(csvfile, fieldnames=keys) # Tạo writer
        if(post_idx == 0): # Nếu là lần ghi đầu tiên, ghi header
            writer.writeheader()
        writer.writerows(result_replies) # Ghi dữ liệu vào file

    print(f'Crawled post no. {postID} ({post_idx+1}/{len(post_list)}), got {delay(0.2)} # Delay 0.2 giây giữa các bài viết, tránh bị block')

    print(f'\nCrawled {len(post_list)} posts, end of data')

```

Tổng cộng cào được **75510** bình luận có trong **3418** bài viết.

```

# Cào dữ liệu bình luận, những bài từ 20 bình luận trở lên
crawl_replies()

Found 1508 posts with totalReplies ≥ 20
Failed to get replies for post no. 29090463 (12/1508), retrying... (1/3)
Failed to get replies for post no. 29090463 (12/1508), retrying... (2/3)
Failed to get replies for post no. 29090463 (12/1508), retrying... (3/3)
Failed to get replies for post no. 29090463 (12/1508), skipping...
Crawled post no. 28005695 (1508/1508), got 22 replies, total crawled 43707 replies
Crawled 1508 posts, end of data

# Cào dữ liệu bình luận, lần 2 (những bài có từ 15-19 bình luận)
crawl_replies(range_of_replies=(15, 19))

Found 1910 posts with totalReplies between 15 and 19
Failed to get replies for post no. 29099118 (1/1910), retrying... (1/3)
Failed to get replies for post no. 29099118 (1/1910), retrying... (2/3)
Failed to get replies for post no. 29099118 (1/1910), retrying... (3/3)
Failed to get replies for post no. 29099118 (1/1910), skipping...
Failed to get replies for post no. 29088819 (25/1910), retrying... (1/3)replies
Failed to get replies for post no. 29088819 (25/1910), retrying... (2/3)
Failed to get replies for post no. 29088819 (25/1910), retrying... (3/3)
Failed to get replies for post no. 29088819 (25/1910), skipping...
Crawled post no. 28005609 (1910/1910), got 18 replies, total crawled 31407 replies
Crawled 1910 posts, end of data

```

1.2.2 Dữ liệu Chứng khoán

Với việc có thư viện hỗ trợ, công việc cào dữ liệu chứng khoán tương đối đơn giản.

```
from vnstock3 import Vnstock
import pandas as pd
import json

stock = Vnstock().stock(source='VCI')      # Nguồn dữ liệu VCI
# Danh sách mã chứng khoán theo ngành
df_list_stock = pd.DataFrame(stock.listing.symbols_by_industries())
# Danh sách các bài viết
df_list_post = pd.read_csv('cleaned_posts.csv')
```

Thông số cơ bản

Với mỗi cổ phiếu, ta dễ dàng đến những dữ liệu đáng chú ý:

- Dữ liệu về tên công ty, ngành nghề:

```
def get_info_industry(symbol: str):
    # Lấy hàng chứa thông tin của mã cổ phiếu cần tìm
    df_row = df_list_stock[df_list_stock['symbol'] == symbol]
    df_row = df_row[['symbol', 'organ_name', 'icb_code2', 'icb_name2']]
    df_row.columns = ['symbol', 'name', 'indus_code', 'indus_name']

    return df_row

get_info_industry('VCI')

symbol          name  indus_code  indus_name
1460  Công ty Cổ phần Chứng khoán Vietcap       8700  Dịch vụ tài chính
```

- Dữ liệu về số lượng cổ đông, nhân viên, đánh giá của TCBS:

```
def get_info_overview(symbol: str):
    # Lấy thông tin tổng quan của mã cổ phiếu cần tìm, từ TCBS
    df_info_ow = pd.DataFrame(Vnstock(show_log=False).stock(symbol=symbol, source='TCBS').company.overview())
    df_info_ow['symbol'] = symbol
    df_info_ow = df_info_ow[['symbol', 'exchange', 'no_shareholders', 'no_employees', 'stock_rating']]

    # Fill NaN thành -1
    df_info_ow = df_info_ow.fillna(-1).infer_objects(copy=False)

    # Đổi tên cột
    df_info_ow.columns = ['symbol', 'exc', 'm_stholder', 'm_employee', 'r_tcbsrating']

    return df_info_ow

get_info_overview('VCI')

symbol   exc  m_stholder  m_employee  r_tcbsrating
0      VCI    HOSE        15669         386           2.3
```

- Dữ liệu về thông tin tài chính công ty, các chỉ số tài chính:

```

df_info_finance = pd.DataFrame()

# Lấy các chỉ tiêu cơ bản trong năm gần nhất (2023)
df_row = df_row.head(1)

LIST_KEYS = [(
    ('Meta', 'CP'), 'symbol'),
    ('Meta', 'Năm'), 'year'),
    ('Chỉ tiêu định giá', 'Vốn hóa (Tỷ đồng)'), 'm_cap' ), # m_cap
    ('Chỉ tiêu định giá', 'Số CP lưu hành (Triệu CP)'), 'm_share', # m_share
    ('Chỉ tiêu định giá', 'P/B'), 'r_pb' ), # r_pb = giá thị trường / BVPS
    ('Chỉ tiêu định giá', 'P/E'), 'r_pe' ), # r_pe = giá thị trường / lợi nhuận sau thuế
    ('Chỉ tiêu định giá', 'EPS (VND)'), 'r_eps' ), # r_eps = lợi nhuận sau thuế / số CP lưu hành
    ('Chỉ tiêu định giá', 'BVPS (VND)'), 'r_bvps' ), # r_bvps = vốn chủ sở hữu / số CP lưu hành
    ('Chỉ tiêu khả năng sinh lời', 'ROE (%)'), 'p_roe' ), # p_roe = lợi nhuận sau thuế / vốn chủ sở hữu
    ('Chỉ tiêu khả năng sinh lời', 'Tỷ suất cổ tức (%)'), 'p_div' ), # p_div = Tỷ suất cổ tức = cổ tức / giá thị trường
]

# Kiểm tra xem các chỉ tiêu có trong df_info_finance không
for key, col_name in LIST_KEYS:
    if key not in df_row.columns:
        df_row[key] = 0

    # Đổi tên cột thành col_name
    df_info_finance[col_name] = df_row[key]

return df_info_finance

get_info_finance('VCI')

```

	symbol	year	m_cap	m_share	r_pb	r_pe	r_eps	r_bvps	p_roe	p_div
0	VCI	2023	15093750000000	437500000	2.047657	30.6843	1124.353507	16848.52459	0.070948	0.0

Kết quả có **1528 mã cổ phiếu**, ta thực hiện lấy và gộp 3 miền dữ liệu bên trên:

```

for i, symbol in enumerate(list_symbol):
    # Lấy thông tin của mã cổ phiếu ở 3 hàm
    df_info_industry = get_info_industry(symbol)
    df_info_overview = get_info_overview(symbol)
    df_info_finance = get_info_finance(symbol)

    # Gộp
    df_info_temp = pd.merge(df_info_industry, df_info_overview, on='symbol', how='inner')
    df_info_temp = pd.merge(df_info_temp, df_info_finance, on='symbol', how='inner')

    df_info = pd.concat([df_info, df_info_temp], ignore_index=True)

    # Lưu vào file CSV
    df_info_temp.to_csv('list_stock.csv', index=False, encoding='utf-8', header=(False if i > 0 else True),
    print(f'Processed {symbol} ({i+1}/{len(list_symbol)})!', end='\r')

print(f'\nProcessed {len(df_info)} stocks, end of data')

```

Python																		
Processed XMC (1532/1532)! Processed 1528 stocks, end of data																		
		symbol	name	indus_code	indus name	exc	m_stholder	m_employee	r_tbcrating	year	m_cap	m_share	r_pb	r_pe	r_eps	r_bvps	p_roe	p_div
1508	X77	Công ty Cổ phần Thành An 77	2300	Xây dựng và Vật liệu	UPCOM	183	2	-1.0	2023	4.022632e+08	1340877.0	0.000000	4.136201	72.530346	0.000000	-0.000508	0.000000	
1509	DLT	Công ty Cổ phần Du lịch và Thương mại - Vinacomin	5700	Du lịch và Giải trí	UPCOM	200	605	-1.0	2023	1.550035e+10	2500056.0	0.262400	5.080386	1987.025164	23628.046929	0.084106	0.161290	
1510	WTC	Công ty Cổ phần Vận tải thủy - Vinacomin	2700	Hàng & Dịch vụ Công nghiệp	UPCOM	511	208	2.3	2023	1.150000e+11	10000000.0	0.508799	7.639684	1615.297791	22602.224455	0.071419	0.104348	

Lịch sử giá

Ta lấy danh sách các mã cổ phiếu được đề cập trong danh sách bài viết. Ta chỉ thực hiện lấy lịch sử giá của các mã cổ phiếu, chỉ số nổi tiếng (VNINDEX, VN30, DJI, ...). Ta lấy dữ liệu từ ngày **01/9/2024** cho tới ngày **08/11/2024**.

Ta sẽ lấy lịch sử giá theo **khung thời gian 30 phút**. Mỗi mục trong lịch sử giá sẽ bao gồm giá khởi điểm (open), giá đóng khung (close), giá cao nhất trong khung (high), giá thấp nhất trong khung (low), khối lượng giao dịch (volume). Một số mã do không đủ dữ liệu, nên chỉ lấy trong khung thời gian 1 ngày.

```

# Lấy danh sách mã cổ phiếu được đề cập
set_taggedSymbols = set()

for post in df_list_post['taggedSymbols']:
    symb_array = json.loads(post)
    for symb in symb_array:
        set_taggedSymbols.add(symb['symb'])

list_taggedSymbols = list(set_taggedSymbols)

# Xử lý từng mã cổ phiếu
for i, symb in enumerate(list_taggedSymbols):

    # Phân loại mã cổ phiếu
    if symb[0] == '$': # Tiền điện tử (crypto)
        crypto = Vnstock().crypto(source='MSN')
        count_processed += process_symbol(crypto, symb[1:])
    elif symb[0] == '^': # Chỉ số (index)
        index = Vnstock().world_index(source='MSN')
        count_processed += process_symbol(index, symb[1:])
    elif '-' in symb: # Tỷ giá (forex)
        forex = Vnstock().fx(source='MSN')
        count_processed += process_symbol(forex, symb.replace('-', ''))

    elif len(symb) > 3 and symb[0] == 'C': # Chứng quyền (warrant), bỏ qua
        continue
    elif len(symb) > 3 and (symb[-2::] == '24' or symb[-2::] == '25'): # Hợp đồng tương lai (future), bỏ qua
        continue
    elif '=' in symb or '.' in symb: # Một số mã cổ phiếu khác, bỏ qua
        continue
    else:
        count_processed += process_symbol(stock, symb, '30m') # Mã cổ phiếu thông thường, lấy dữ liệu 30 phút
print(f'Processed {symb} ({i+1}/{len(list_taggedSymbols)}), saved to price/{symb}.csv', end='\r')

```

Kết quả ta lấy được lịch sử giá của **1154** mã, với mỗi mã đều có cấu trúc như trên:

	time	open	high	low	close	volume
1	2024-09-04 09:00:00	20.3	20.55	20.3	20.5	1078100
2	2024-09-04 09:30:00	20.5	20.6	20.45	20.55	1190500
3	2024-09-04 10:00:00	20.55	20.55	20.35	20.4	1409100
4	2024-09-04 10:30:00	20.4	20.4	20.2	20.2	2259200
5	2024-09-04 11:00:00	20.2	20.3	20.2	20.2	1465800
6	2024-09-04 13:00:00	20.2	20.3	20.2	20.3	1003100
7	2024-09-04 13:00:00	20.2	20.3	20.2	20.3	1003100

1.3 Tiềm xử lý, làm sạch dữ liệu

Dữ liệu FireAnt

Trong 60 trường dữ liệu, ta chỉ phân tích một vài trường dữ liệu:

```

# Dãy các key giá trị cần lấy từ API
USEFUL_KEYS = ['postID', 'user', 'originalContent', 'date', 'images', 'files', 'link', 'linkTitle', 'sentiment', 'totalLikes',
'totalReplies', 'totalShares', 'replyToPostID', 'taggedSymbols', 'linkTitle', 'totalFiles', 'isTop', 'isExpertIdea']

```

Một vài trường dữ liệu chỉ có một giá trị, ví dụ như `totalShares` chỉ có giá trị 0 (?), `files`, `totalFiles` cũng không có giá trị, `isTop`, `isExpertIdea` toàn bộ là False...

```

def clean_data(csvfile):
    """
    Hàm xử lý dữ liệu sau khi cào, xóa các bài viết trùng lặp và các cột không cần thiết

    csvfile: file CSV cần xử lý
    """

    # Đọc file CSV
    df = pd.read_csv(csvfile)

    # Xóa các bài viết trùng lặp
    df = df.drop_duplicates(subset=['postID'])

    # Xóa các cột không cần thiết (các cột này chỉ có 1 giá trị, chi tiết trong báo cáo)
    df = df.drop(columns=['totalShares', 'linkTitle', 'totalFiles'])

    # Ghi dữ liệu vào file CSV mới (cleaned_{csvfile})
    df.to_csv(f'cleaned_{csvfile}', index=False)

    print(f'Cleaned data saved to cleaned_{csvfile}, total {len(df)} entries')

```

Một số trường dữ liệu có thể đơn giản hơn nữa, ví dụ như danh sách các mã chứng khoán được nhắc đến (`taggedSymbols`), thay vì lưu toàn bộ dữ liệu, ta thực sự chỉ quan tâm đến tên cổ phiếu và giá của cổ phiếu tại thời điểm đăng bài. Một số khác như `images`, `files`, ta chỉ cần quan tâm đến số lượng.

```

def process_post(post: dict):
    """
    Hàm xử lý dữ liệu bài viết, xử lý qua các key và bỏ các key không cần thiết
    Trả về bài viết đã được xử lý
    """

    post = {k: v for k, v in post.items() if k in USEFUL_KEYS} # Chỉ giữ lại các key cần thiết

    list_symbol = post['taggedSymbols'] # Xử lý các mã được đề cập trong key 'taggedSymbols'
    list_processed_symbol = [] # Danh sách các mã sau khi xử lý
    for symbol in list_symbol:
        # symb: mã cổ phiếu
        # price: giá cổ phiếu tại thời điểm bài viết được đăng
        try:
            list_processed_symbol.append({
                'symb': symbol['symbol'],
                'price': round(float(symbol['price']),2),
            })
        except:
            pass

    post['taggedSymbols'] = json.dumps(list_processed_symbol) # Gán lại giá trị mới cho key 'taggedSymbols'
    # Xử lý các key cần thiết
    post['username'] = post['user']['name']
    post['userid'] = post['user']['id']
    post['totalImages'] = len(post['images'])
    post['totalFiles'] = len(post['files'])
    post['totalSymbols'] = len(list_processed_symbol)

```

Bằng cách trên, kích cỡ dữ liệu đã giảm 4 lần, từ 417MB xuống còn 105MB.

```

# Làm sạch dữ liệu trong 2 file CSV
clean_data('posts.csv')
clean_data('replies.csv')

Cleaned data saved to cleaned_posts.csv, total 272979 entries
Cleaned data saved to cleaned_replies.csv, total 75510 entries

```

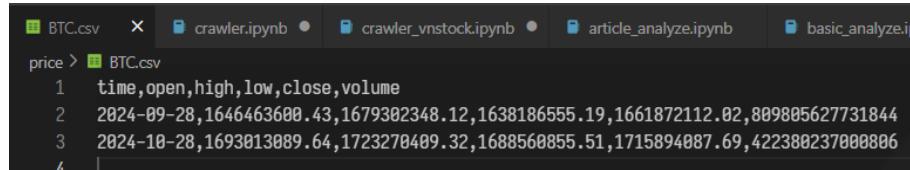
Ngoài ra, một số bài viết bị nhảy, mất dữ liệu. Tuy nhiên có thể giải quyết bằng cách gửi request ID bài viết bị hut đến API <https://api.fireant.vn/posts/<post-id>>.

Với dữ liệu bình luận cũng xảy ra tình trạng tương tự. Nếu dữ liệu trả về Null, ta sẽ thử lại 3 lần, giữa mỗi lần delay sẽ tăng thêm. Nếu vẫn không lấy được dữ liệu, hoặc bị lỗi 403 Forbidden thì khả năng cao bị chặn IP, cần phải thử lại sau (và thực tế chúng em đã bị chặn IP hai lần).

Dữ liệu chứng khoán

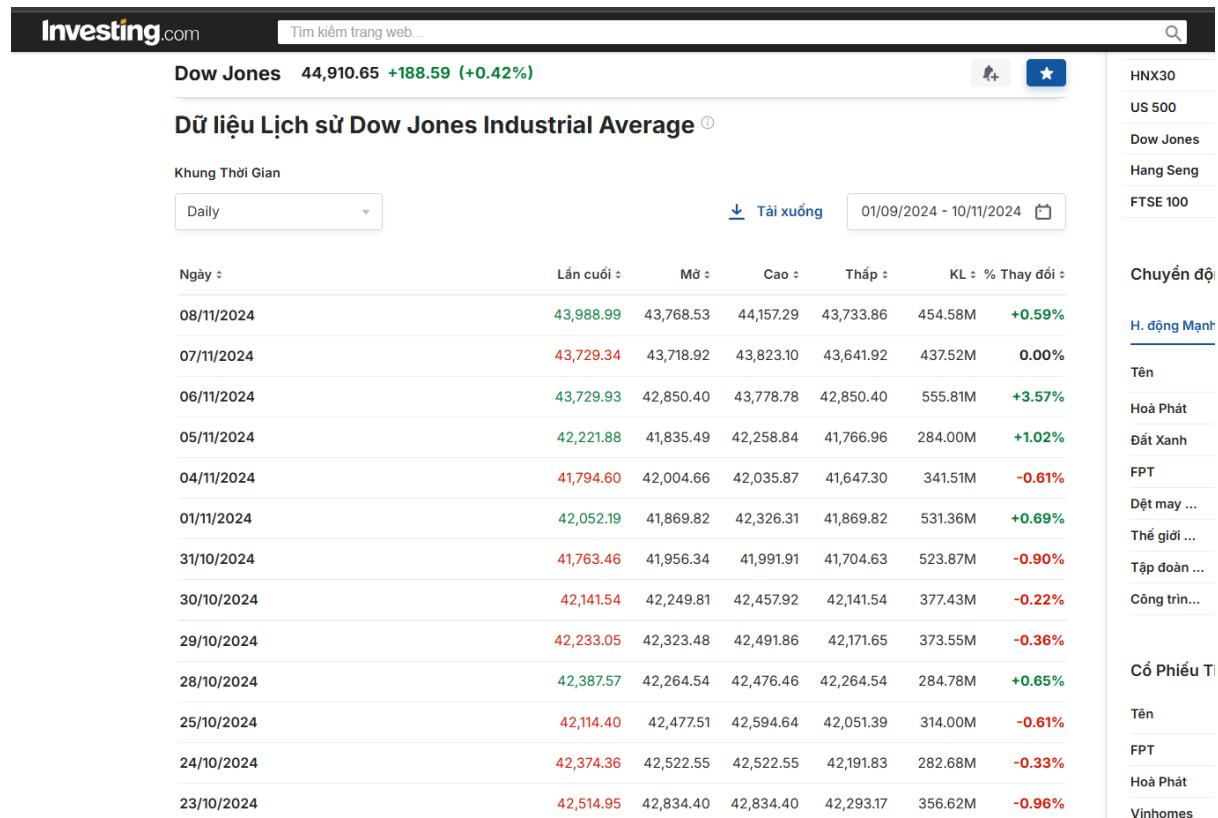
Dữ liệu thông số cơ bản của các công ty tương đối sạch.

Dữ liệu lịch sử giá tại các công ty chứng khoán Việt cũng rất ổn, tuy nhiên có một số mã phải cào ở nguồn MSN bị thiếu (các chỉ số quốc tế như DJI, giá tiền ảo như BTC...), chỉ có dữ liệu theo từng tháng.



```
price > BTC.csv
1   time,open,high,low,close,volume
2   2024-09-28,1646463600.43,1679302348.12,1638186555.19,1661872112.02,809805627731844
3   2024-10-28,1693013889.64,1723270409.32,1688560855.51,1715894087.69,422380237000806
4
```

Ta cần tìm nguồn dữ liệu khác. Qua quá trình tìm hiểu, chúng em phát hiện trang <https://vn.investing.com> có cung cấp giá lịch sử của loại mã này:



Ngày	Lần cuối	Mở	Cao	Thấp	KL	% Thay đổi
08/11/2024	43,988.99	43,768.53	44,157.29	43,733.86	454.58M	+0.59%
07/11/2024	43,729.34	43,718.92	43,823.10	43,641.92	437.52M	0.00%
06/11/2024	43,729.93	42,850.40	43,778.78	42,850.40	555.81M	+3.57%
05/11/2024	42,221.88	41,835.49	42,258.84	41,766.96	284.00M	+1.02%
04/11/2024	41,794.60	42,004.66	42,035.87	41,647.30	341.51M	-0.61%
01/11/2024	42,052.19	41,869.82	42,326.31	41,869.82	531.36M	+0.69%
31/10/2024	41,763.46	41,956.34	41,991.91	41,704.63	523.87M	-0.90%
30/10/2024	42,141.54	42,249.81	42,457.92	42,141.54	377.43M	-0.22%
29/10/2024	42,233.05	42,323.48	42,491.86	42,171.65	373.55M	-0.36%
28/10/2024	42,387.57	42,264.54	42,476.46	42,264.54	284.78M	+0.65%
25/10/2024	42,114.40	42,477.51	42,594.64	42,051.39	314.00M	-0.61%
24/10/2024	42,374.36	42,522.55	42,522.55	42,191.83	282.68M	-0.33%
23/10/2024	42,514.95	42,834.40	42,834.40	42,293.17	356.62M	-0.96%

Hình 1.3: Thông tin lịch sử giá trên trang Investing

Sử dụng công cụ Inspect, ta tìm ra được API để lấy giá lịch sử:

The screenshot shows a browser developer tools Network tab with a selected request for the URL <https://api.investing.com/api/financialdata/historical/169?start-date=2024-09-01&end-date=2024-11-08&time-frame=Daily&add-missing-rows=false>. The Response tab displays a JSON object containing historical financial data for index 169. The JSON includes fields such as "rowDate", "last_close", "last_max", "last_min", "volume", and "change_percent". The response body is as follows:

```
{
  "rowDate": "08/11/2024",
  "rowDateRaw": 1731024000,
  "rowDateTimestamp": "2024-11-08T00:00:00Z",
  "last_close": "43,988.99",
  "last_open": "43,768.53",
  "last_max": "44,157.29",
  "last_min": "43,733.86",
  "volume": "454.58M",
  "volumeRaw": 454575008,
  "change_percent": "0.59",
  "last_closeRaw": "43988.9882125000000",
  "last_openRaw": "43768.5312500000000",
  "last_maxRaw": "44157.2890625000000",
  "last_minRaw": "43733.8593750000000",
  "change_percentRaw": 0.5937625366121555
}
```

At the bottom left, it says "2 requests | 4.8 kB transferred | 23.3 kB resources".

Ta tích hợp API này vào Notebook (`crawl_indexes.ipynb`), xử lý kết quả để nó có dạng giống các mã khác, sau đó lưu vào một file CSV trong thư mục mới.

```

import pandas as pd

# (Mã truy vấn Investing, Tên mã viết tắt, Tên đầy đủ)
INDEX_LIST = [(179, 'HSI', 'Hang Seng Index (Hồng Kông)'),
               (169, 'DJI', 'Dow Jones Industrial Average (Mỹ)'),
               (27, 'FTSE', 'FTSE 100 UK100 (Anh)'),
               (37426, 'KS11', 'KOSPI (Hàn Quốc)'),
               (48820, 'SSE', 'Shanghai Composite (Trung Quốc)'),
               (178, 'N225', 'Nikkei 225 (Nhật Bản)'),
               (1057391, 'BTC', 'Bitcoin'),
               (41063, 'VNI', 'VN-Index (Việt Nam)'),
               (41064, 'VN30', 'VN30-Index (Việt Nam)'),
               (995072, 'HNX30', 'HNX30-Index (Việt Nam)],]

```

0.5s

Tương tự, ta sẽ lấy dữ liệu từ ngày 01/9/2024 đến 08/11/2024.

```

import requests

def get_price(index: tuple, startDate, endDate):
    url = f'https://api.investing.com/api/financialdata/historical/{index[0]}'
    headers = { # Header trong request
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/131.0.0',
        'Domain-Id': 'vn'
    }
    params = { # Thông số truy vấn
        'startDate': startDate, # Ngày bắt đầu
        'endDate': endDate, # Ngày kết thúc
        'time-frame': 'Daily', # Khoảng thời gian
        'add-missing-rows': 'false' # Không thêm dữ liệu thiếu
    }

    response = requests.get(url, headers=headers, params=params) # Gửi request
    data = response.json() # Dữ liệu trả về là JSON

    df = pd.DataFrame(data['data'])
    # Chỉ lấy các cột cần thiết
    df = df[['rowDateTimestamp', 'last_openRaw', 'last_maxRaw', 'last_minRaw', 'last_openRaw', 'volumeRaw']]
    df.columns = ['time', 'open', 'high', 'low', 'close', 'volume']
    df['time'] = df['time'].str[:10] # 2024-09-12T00:00:00Z → 2024-09-12

    return df

get_price(INDEX_LIST[0], '2024-09-01', '2024-11-08').head(20)

```

1.2s

	time	open	high	low	close	volume
0	2024-11-08	21199.9609375000000	21355.4394531250000	20705.1191406250000	21199.9609375000000	4153511168
1	2024-11-07	20386.3593750000000	20986.3105468750000	20370.4394531250000	20386.3593750000000	4317469184
2	2024-11-06	20791.8007812500000	20859.6601562500000	20361.9199218750000	20791.8007812500000	3903889920

```

import os
if not os.path.exists('price_index'):
    os.makedirs('price_index')

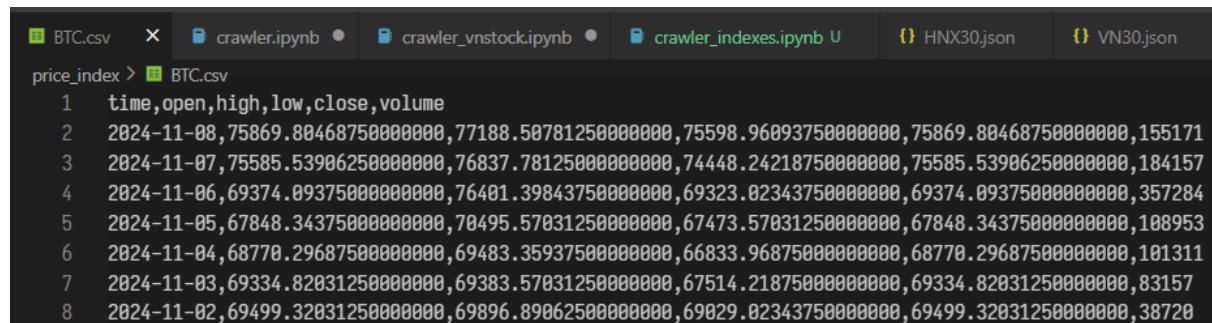
for index in INDEX_LIST:
    df = get_price(index, '2024-09-01', '2024-11-08')
    df.to_csv(f'price_index/{index[1]}.csv', index=False)
    print(f'Done processing {index[1]}')

3.0s

Done processing HSI
Done processing DJI
Done processing FTSE
Done processing KS11

```

Dữ liệu đã được bổ sung đầy đủ, ngoài ra còn có thêm dữ liệu của một số chỉ số không có trong dữ liệu ban đầu.



```

price_index > BTC.csv
1 time,open,high,low,close,volume
2 2024-11-08,75869.8046875000000,77188.5078125000000,75598.9609375000000,75869.8046875000000,155171
3 2024-11-07,75585.5390625000000,76837.781250000000,74448.2421875000000,75585.5390625000000,184157
4 2024-11-06,69374.0937500000000,76401.3984375000000,69323.0234375000000,69374.0937500000000,357284
5 2024-11-05,67848.3437500000000,70495.5703125000000,67473.5703125000000,67848.3437500000000,108953
6 2024-11-04,68770.2968750000000,69483.3593750000000,66833.9687500000000,68770.2968750000000,101311
7 2024-11-03,69334.8203125000000,69383.5703125000000,67514.2187500000000,69334.8203125000000,83157
8 2024-11-02,69499.3203125000000,69896.8906250000000,69029.0234375000000,69499.3203125000000,38720

```

Ta kết thúc quá trình thu thập và tiền xử lý dữ liệu ở đây. Tuy nhiên với một vài trường hợp cụ thể, ta vẫn cần phải xử lý thêm dữ liệu, để dữ liệu phù hợp với hướng phân tích đó.

Chương 2

Phân tích Dữ liệu Diễn đàn FireAnt

Sau khi thu thập những dữ liệu của diễn đàn FireAnt, ta thu về được những con số, những dữ liệu giàu giá trị phân tích. Chúng ta sẽ cùng đi qua toàn bộ những khía cạnh của dữ liệu, từ nội dung bài viết, các bình luận cũng như các người dùng, để có những thống kê thú vị!

2.1 Tinh chỉnh Dữ liệu để phân tích



```
1 import pandas as pd
2 import numpy as np
3 import json
4 from dateutil.parser import isoparser
5
6 posts_df = pd.read_csv("cleaned_posts.csv")
7 replies_df = pd.read_csv('cleaned_replies.csv')
8
9
10 posts_df.head(3)
```

Đây là nơi ta bắt đầu. Ta đọc dữ liệu từ hai tệp CSV: `cleaned_posts.csv` và `cleaned_replies.csv`. Các thư viện `numpy`, `pandas` cũng được import để hỗ trợ các phép toán số học, xử lý dữ liệu. Ngoài ra, một số thư viện khác như `dateutil`, `json` được khai báo để thực hiện hỗ trợ biến đổi dữ liệu.



```
1 # Convert data type from 'object' to a Python string
2 posts_df['taggedSymbols'] = posts_df['taggedSymbols'].astype("string")
3 posts_df['postID'] = posts_df['postID'].astype('string')
4 posts_df['originalContent'] = posts_df['originalContent'].astype("string")
5 posts_df['link'] = posts_df['link'].astype("string")
6
7 posts_df['sentiment'] = posts_df['sentiment'].astype('string')
8 posts_df['sentiment'] = posts_df['sentiment'].map({'1' : 'positive', '0' : 'neutral', '-1' : 'negative'})
9
10 posts_df['totalLikes'] = pd.to_numeric(posts_df['totalLikes'], errors='coerce').astype('Int64')
11 posts_df['totalReplies'] = pd.to_numeric(posts_df['totalReplies'], errors='coerce').astype('Int64')
12
13 posts_df['date'] = posts_df['date'].astype("string")
14 posts_df['date'] = posts_df['date'].apply(lambda x: isoparser().isoparse(x) if pd.notnull(x) else pd.NaT)
15
16
17 posts_df['totalImages'] = pd.to_numeric(posts_df['totalImages'], errors='coerce').astype('Int64')
18 posts_df['totalSymbols'] = pd.to_numeric(posts_df['totalSymbols'], errors = 'coerce').astype('Int64')
19
20 posts_df['replyToPostID'] = posts_df['replyToPostID'].astype("string")
21 posts_df['username'] = posts_df['username'].astype("string")
22 posts_df['userid'] = posts_df['userid'].astype('string')
```

Ta thực hiện chuyển đổi kiểu dữ liệu về các dạng phù hợp cho các cột trong dataframe `posts_df`. Các bước xử lý bao gồm việc định lại kiểu cột, gán nhãn dữ liệu, dịch thời gian sang dạng DateTime, ...

Xử lý cột `taggedSymbols`

Cột `taggedSymbols` là một cột dữ liệu đặc biệt cần được xử lý riêng. Đây là cột chứa thông tin về các mã chứng khoán và giá của chúng tại thời điểm được nhắc tới. Lưu ý rằng mỗi bài viết có thể đề cập đến nhiều mã chứng khoán cùng một lúc, khiến dữ liệu trong cột này có dạng danh sách, chứa nhiều phần tử.

Ta sẽ giải quyết bằng cách sử dụng `.explode()` trong pandas, hàm này có tác dụng tách một hàng có cột dạng list thành nhiều hàng:

```
def json_to_dict(x):
    try:
        data = json.loads(x.replace("'", ""))
        return data
    except json.JSONDecodeError:
        return None

posts_df['taggedSymbols'] = posts_df['taggedSymbols'].apply(json_to_dict)
posts_df = posts_df.explode('taggedSymbols', ignore_index = True)

posts_df['taggedSymbols']

0      {'symb': '^DJI', 'price': 43496.43}
1          NaN
2      {'symb': '^DJI', 'price': 43506.86}
3      {'symb': 'VNINDEX', 'price': 1261.28}
4      {'symb': '^DJI', 'price': 43499.54}
...
487263      {'symb': 'GVR', 'price': 34.0}
487264      {'symb': 'DIG', 'price': 22.75}
487265      {'symb': 'FPT', 'price': 131.3}
487266      {'symb': 'OIL', 'price': 13.9}
487267      {'symb': 'NKG', 'price': 20.7}
Name: taggedSymbols, Length: 487268, dtype: object
```

Sau khi tách các mã chứng khoán trong mỗi bài viết thành các hàng riêng biệt, ta thực hiện tách phần này thành hai cột riêng biệt: mã chứng khoán được đề cập (`symbol`) và giá tại thời điểm đề cập (`price`). Việc tách này giúp việc theo dõi và phân tích dữ liệu trở nên dễ dàng và trực quan hơn. Dữ liệu các bài viết được lưu trong dataframe `posts_df`, dữ liệu bình luận được lưu trong `replies_df`.

```
def extract_price(x):
    if isinstance(x, dict) and 'price' in x:
        return x['price']
    return None

posts_df['price'] = posts_df['taggedSymbols'].apply(extract_price)

def extract_symbol(x):
    if isinstance(x, dict) and 'symb' in x:
        return x['symb']
    return None
posts_df['symbol'] = posts_df['taggedSymbols'].apply(extract_symbol)
posts_df.head(2)
```

2.2 Phân tích dữ liệu bài đăng

2.2.1 Số bài đăng theo ngày



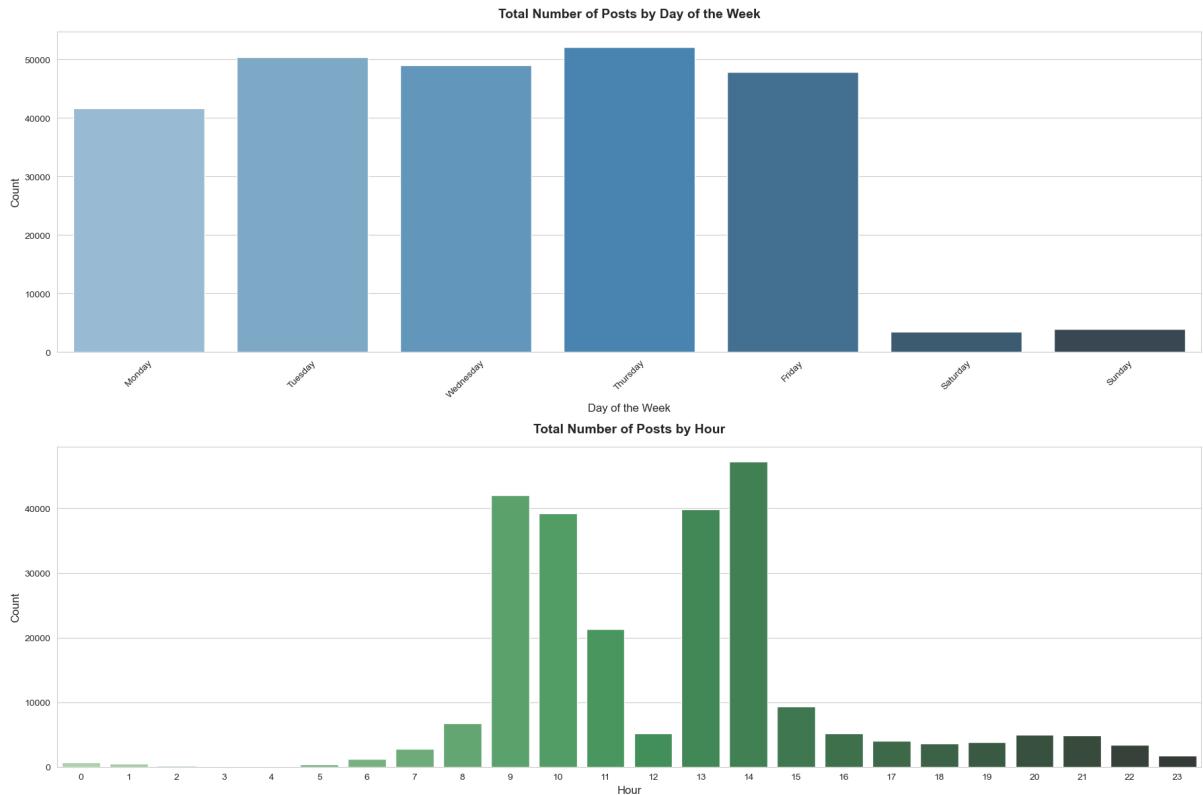
Hình 2.1: Số lượng bài đăng hàng ngày

Nhận xét:

- Biểu đồ cho thấy sự chênh lệch đáng kể giữa các ngày. Một số ngày có số bài đăng rất thấp (dưới 1000 bài), trong khi những ngày khác vượt quá 6000 bài, và có chu kỳ lặp lại.
- Những ngày ít bài đăng là những ngày cuối tuần, khi mà thị trường không mở cửa để nhà đầu tư giao dịch. Như vậy sẽ có ít bài viết để nói hơn.
- Đường trung bình nằm dưới đường trung vị, cho thấy phân phối có một số ngày có lượng bài đăng rất thấp kéo giá trị trung bình xuống.
- Đường trung vị cao hơn đường trung bình, điều này phản ánh rằng phần lớn các ngày có số bài đăng cao hơn giá trị trung bình, nhưng có một vài giá trị ngoại lai thấp (outliers) làm giảm giá trị trung bình.

2.2.2 Số bài đăng theo ngày trong tuần và giờ

Có thể thấy diễn đàn có chu kỳ tương tác tương đối thu vị, vậy thì thời gian hoạt động chủ yếu của người dùng là gì?



Hình 2.2: Số lượng bài đăng theo tuần, theo giờ

Người dùng hoạt động mạnh nhất vào Thứ 5 và Thứ 3 và hoạt động ít nhất vào các ngày nghỉ là Thứ 7 và Chủ Nhật. Khung giờ vàng hay khung giờ cao điểm của người dùng nói chung là từ 9-11h và 13-15h. Tuy nhiên, vào giữa khung giờ vàng, lúc 12h, lượng tương tác đột ngột giảm mạnh, và ngoài các khung giờ này, mức độ tương tác cũng rất thấp.

Lý do là bởi thị trường chứng khoán Việt Nam hoạt động từ Thứ 2 - Thứ 6, mở cửa vào lúc 9h-11h30, nghỉ trưa từ 11h30-13h, sau đó mở cửa từ lúc 13h-15h. Ta dễ dàng thấy khung giờ mà số lượng bài viết nhiều nhất trùng khớp với khoảng thời gian này.

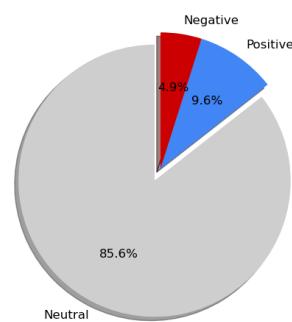
2.2.3 Phân bố Quan điểm người dùng

Theo hình (2.3), hầu hết (85.6%) bài đăng đều mang quan điểm Trung lập. Điều này phần lớn là do thói quen không đặt quan điểm cho bài viết của người dùng. Ngoài ra, số lượng bài Tích cực (9.6%) lớn gần gấp đôi so với Tiêu cực (4.9%), cho thấy xu hướng chung của diễn đàn là tương đối tích cực, ít nhất là trong khoảng dữ liệu hai tháng.

2.2.4 Wordcloud bài đăng

Về nội dung bài viết, ta có thể tìm ra những từ khóa nào được người dùng sử dụng nhiều. Bằng cách sử dụng word cloud ta có mô phỏng như sau (hình (2.4)):

Như vậy có thể thấy, ngoài **cổ phiếu, giao dịch, thị trường** thì có một số keyword đáng chú ý khác là **thanh khoản, điểm mua, kỳ vọng** là những keyword đáng chú ý. Có thể thấy diễn đàn có xu hướng khá tích cực, khi sử dụng nhiều từ như **hỗ trợ, tăng trưởng, tích lũy, tiếp tục, xu hướng**, ...



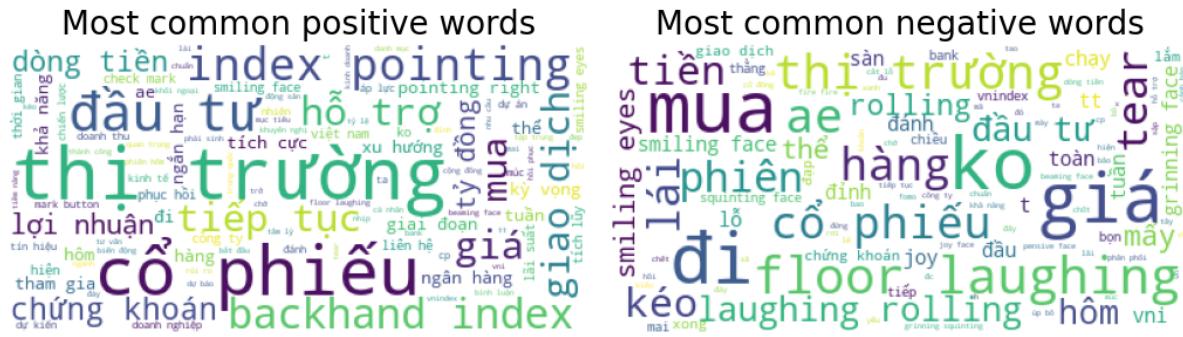
Hình 2.3: Phân bố Quan điểm người dùng



Hình 2.4: Wordcloud bài viết

Biểu diễn keyword dựa trên quan điểm của người dùng

Trong dữ liệu của mỗi bài viết có mục về quan điểm (sentiment) của người dùng trong bài viết đó, vì vậy ta có thể sử dụng chúng để chia thành hai word cloud theo quan điểm của từng người như sau:



Hình 2.5: Một số từ hàm ý Tích cực và Tiêu cực xuất hiện nhiều nhất

Nhận xét:

- Giữa 2 wordcloud ta có thể thấy có một số keyword trùng lặp như **cổ phiếu**, **giá**, **thị trường**, **đầu tư**.
- Các bài tiêu cực, lại không có nhiều từ mang hàm ý tiêu cực. Trong khi các bài tích cực thì vẫn theo xu hướng chung của diễn đàn.
- Trong các bài thường có xu hướng xuất hiện nhiều từ như **index pointing**, **backhand index**, **laughing rolling**, ... là các emoji. Điều này cho thấy mức độ sử dụng emoji trong các bài viết là tương đối lớn.

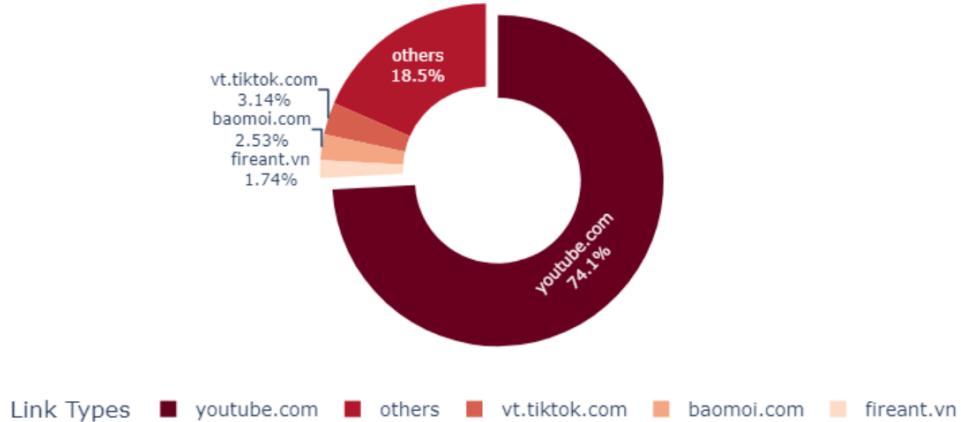
2.2.5 Số lần xuất hiện của link

```

    ● Lấy 5 loại link phổ biến nhất
    top_5_links = link_type_counts['link_type'][:5].tolist()
    # Thay thế các loại link ít phổ biến bằng 'others'
    links_df['link_type'] = links_df['link_type'].apply(lambda x: x if x in top_5_links else 'others')

```

Top 5 Link Types by Appearance



Hình 2.6: Top 5 loại link xuất hiện nhiều nhất

Nhận xét:

- Có thể thấy rằng đa số các liên kết trong bài viết đều dẫn đến các trang web bên ngoài, trong đó có nhiều liên kết đến **youtube.com**.
- Giải thích cho điều này là vì YouTube là nền tảng cho phép upload video và là nền tảng livestream với số lượng người theo dõi rất đông đảo, đồng thời cho phép kiếm tiền vậy nên các video thường có xu hướng cập nhật thông tin rất nhạy so với thị trường.
- Người dùng thường hay lấy những thông tin được dẫn chứng từ các video YouTube, cũng như các chuyên gia hay chia sẻ nhận định hoặc livestream trên nền tảng này.
- Các trang còn lại có sự phân bố tỉ lệ khá đều và nhỏ lẻ với 3.14% từ **tiktok.com**, 2.53% từ **baomoi.com** và 1.74% từ **fireant.vn**.
- Còn lại là các mã có tỉ lệ xuất hiện ít hơn tổng hợp lại trong **Others** với 18.5%.

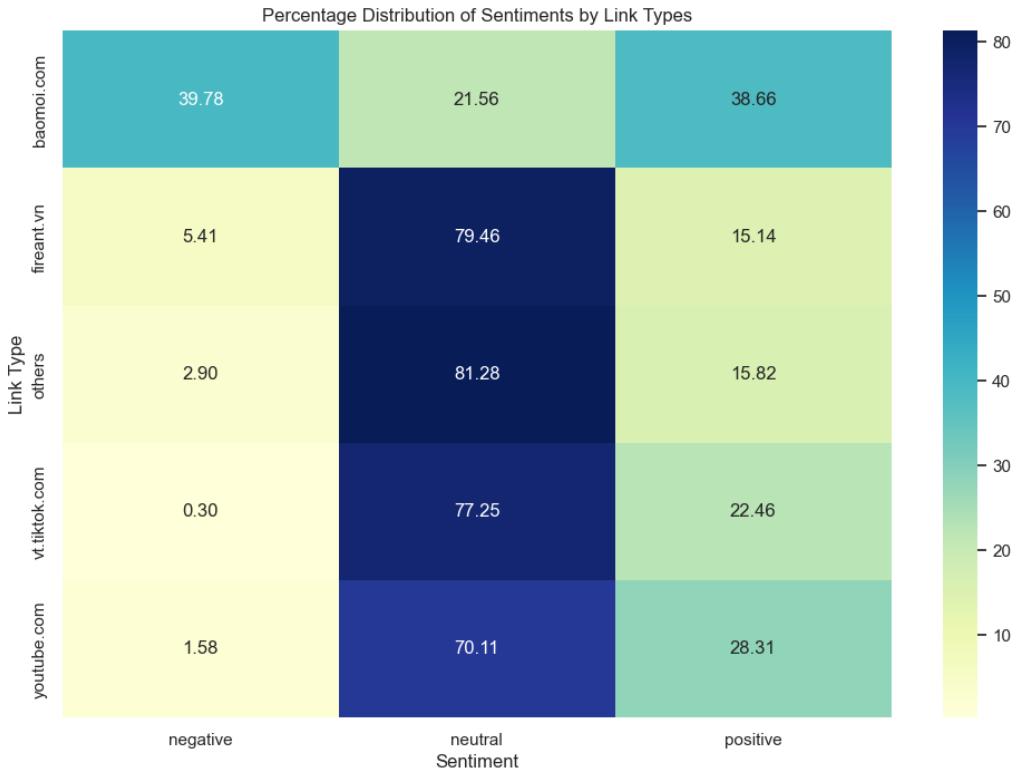
2.2.6 Quan điểm người dùng khi chèn đường link

```
● Chuẩn bị dữ liệu
1 # Chuẩn bị dữ liệu
2 link_sentiment_correlation = links_df.pivot_table(index='link_type', columns='sentiment', aggfunc='size', fill_value=0)
3 link_sentiment_correlation_percentage = link_sentiment_correlation.div(link_sentiment_correlation.sum(axis=1), axis=0) * 100 # Chuẩn hóa tỷ lệ %
4 link_sentiment_correlation_percentage = link_sentiment_correlation_percentage.reset_index()
5
```

Đặt câu hỏi về phản ứng của người đăng bài khi họ nhắc đến các đường dẫn như nào. Để biểu diễn kĩ hơn, ta sử dụng heatmap để miêu tả sự phân bố của các sentiment (hình (2.7)). Từ biểu đồ heatmap (2.7), ta có thể rút ra một số nhận xét:

- **youtube.com**: Nền tảng chia sẻ video, đây là loại liên kết phổ biến nhất và có sự phân bố lệch hẳn về phía Trung lập.
- **vt.tiktok.com**: Nền tảng chia sẻ video ngắn, loại liên kết này chủ yếu có quan điểm trung lập và tích cực, với số lượng quan điểm trung lập chiếm phần lớn.
- **fireant.vn**: Liên kết trong FireAnt, loại liên kết này chủ yếu có sentiment trung lập, với số lượng quan điểm tiêu cực và tích cực ít hơn.

- **baomoi.com:** Trang báo tổng hợp thông tin, số lượng liên kết từ baomoi.com có quan điểm tiêu cực và tích cực gần như tương đương, trong khi sentiment trung lập ít hơn.
- Những liên kết khác: Các liên kết khác ngoài top 4 chủ yếu có quan điểm trung lập, nhưng cũng có một số lượng đáng kể quan điểm tích cực và tiêu cực.



Hình 2.7: Heatmap tỉ lệ phản ứng người dùng với liên kết

Các bài viết kèm link phần lớn là Tích cực hoặc Trung lập. Tuy nhiên các liên kết từ **baomoi.com** có sự phân bố quan điểm đa dạng hơn. Lý do có thể tới từ việc **baomoi.com** là một trang báo tổng hợp thông tin, nhiều thông tin kiểm chứng từ báo dài được đưa ra, cung cấp cho luận điểm đa dạng của người dùng.

2.3 Phân tích phản hồi/bình luận

Ta có cấu trúc dataframe `replies_df` giống với `posts_df`, do bình luận được lưu trong Cơ sở dữ liệu của FireAnt như một loại bài viết đặc biệt. Vì vậy ta không phải viết lại nhiều code.

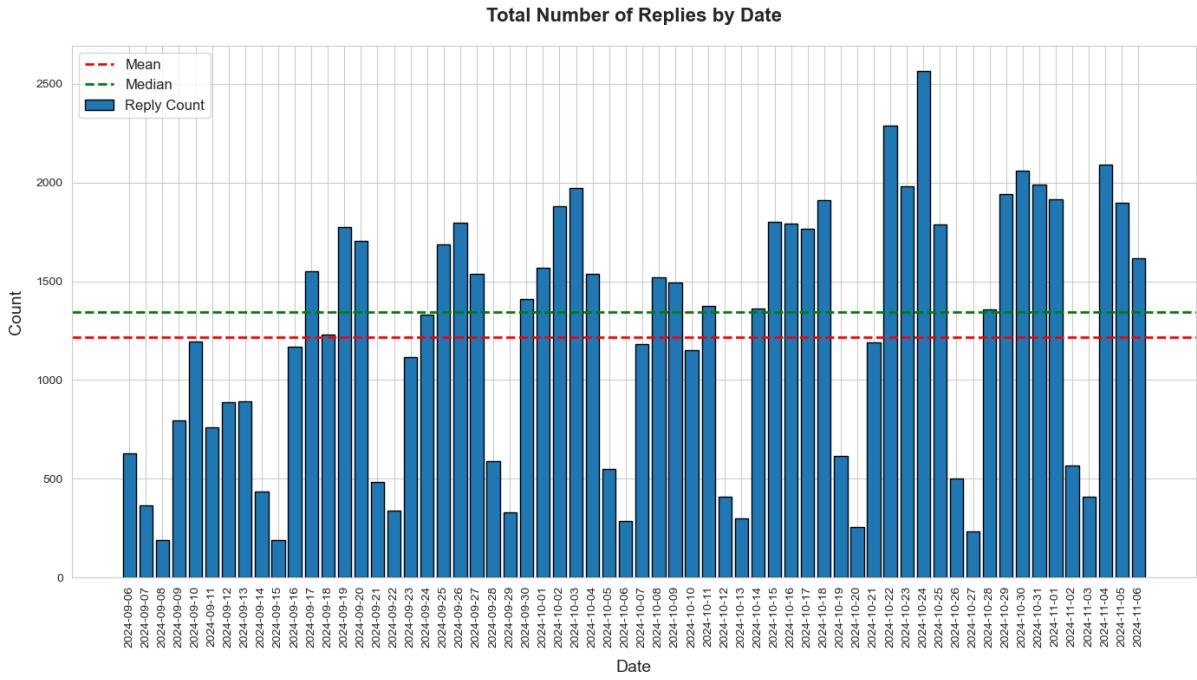
2.3.1 Số phản hồi theo ngày

Ta miêu tả số lượng phản hồi theo ngày dưới dạng chart với code sau:

```

● ● ●
1 # Chuyển đổi cột 'date' sang định dạng datetime
2 replies_df['date'] = pd.to_datetime(replies_df['date'], format='ISO8601')
3
4 # Tạo dataframe mới chứa ngày, số lượng phản hồi
5 reply_counts_by_date = replies_df.groupby(pd.Grouper(key='date', freq='D')).size().reset_index()
6 reply_counts_by_date.columns = ['date', 'reply_count']
7 reply_counts_by_date['date'] = reply_counts_by_date['date'].dt.strftime('%Y-%m-%d')
8 reply_mean_value = reply_counts_by_date['reply_count'].mean()
9 reply_median_value = reply_counts_by_date['reply_count'].median()

```



Hình 2.8: Số lượng các phản hồi theo ngày

Nhận xét:

- Nhìn chung, số lượng liên kết dao động trong khoảng từ dưới 50 đến hơn 250, với một số ngày có số lượng rất thấp.
- Số lượng bình luận có sự tương quan rất lớn với số lượng bài viết, khi cũng có cùng một “chu kỳ” với lượng bài viết, là nhiều vào những ngày trong tuần, và ít đi ở những ngày cuối tuần.
- Giá trị trung bình thấp hơn giá trị trung vị, cho thấy có những ngày số lượng liên kết rất thấp làm giảm giá trị trung bình. Đường trung vị cao hơn đường trung bình, cho thấy rằng phần lớn các ngày có số lượng liên kết lớn hơn giá trị trung bình, nhưng có một vài ngày với số lượng liên kết rất thấp (outliers).

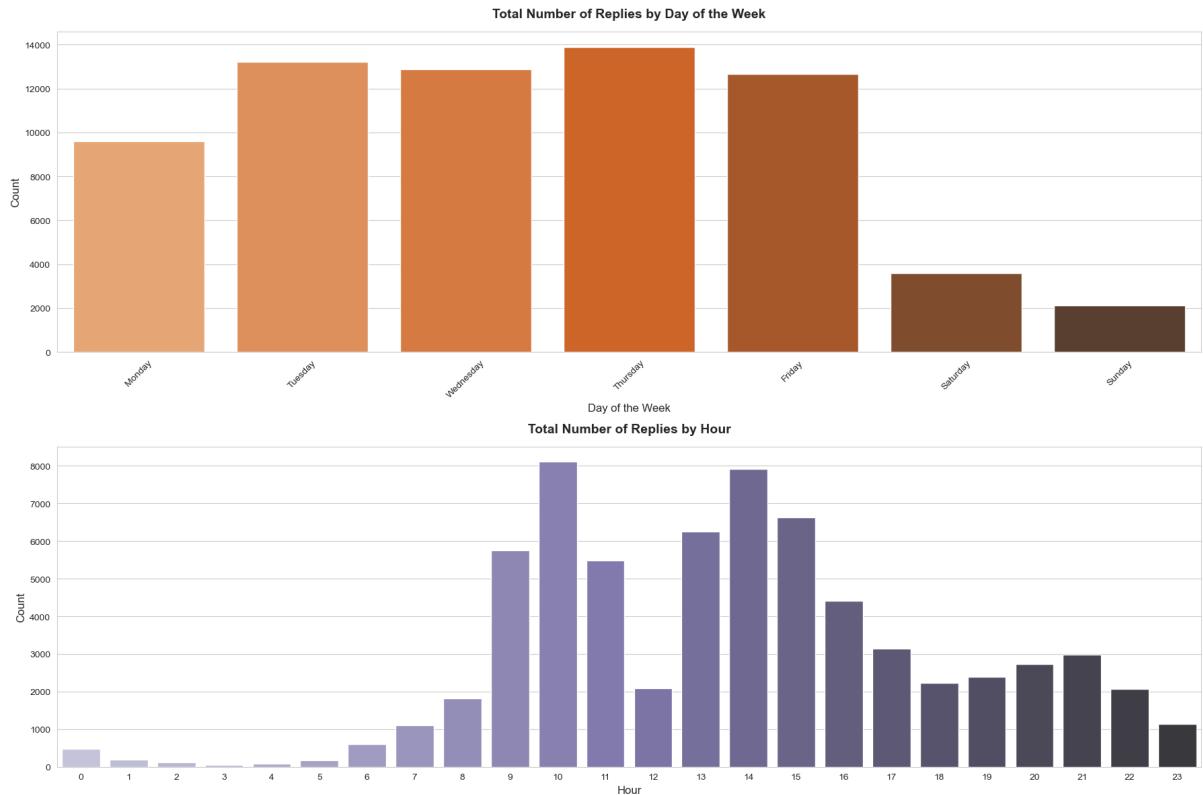
2.3.2 Số phản hồi theo ngày trong tuần và giờ

```

1 start_date = '2024-09-08'
2 end_date = '2024-11-02'
3
4 posts_df = posts_df[(posts_df['date'] >= start_date) & (posts_df['date'] <= end_date)]
5 replies_summary = replies_summary[(replies_summary['date'] >= start_date) & (replies_summary['date'] <= end_date)]
6 # 3. Số lượng phản hồi theo thứ trong tuần
7 replies_summary['day_of_week'] = replies_summary['date'].dt.day_name()
8 replies_summary['day_of_week'] = pd.Categorical(replies_summary['day_of_week'], categories=[
9     'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'], ordered=True)
10 replies_by_day = replies_summary['day_of_week'].value_counts().reset_index()
11 replies_by_day.columns = ['day_of_week', 'count']
12 # 4. Số lượng phản hồi theo giờ
13 reply_counts_by_hour = replies_summary.groupby(replies_summary['date'].dt.hour).size().reset_index()
14 reply_counts_by_hour.columns = ['hour', 'reply_count']

```

Như phân tích trên đã phân tích, dù đồ thị của lượt phản hồi theo tuần và theo giờ có sự tương đồng lớn với đồ thị của số lượng bài viết, nhưng cũng có vài điểm khá nổi bật.



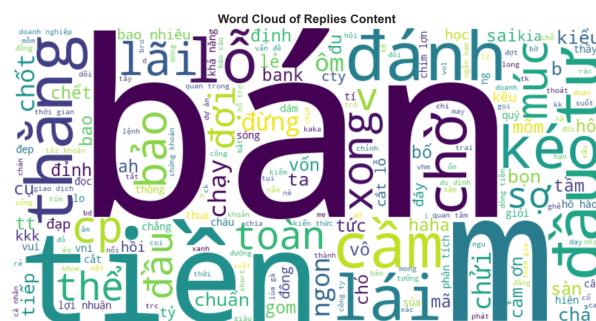
Hình 2.9: Số lượng phản hồi theo ngày trong tuần và giờ

Nhận xét:

- Người dùng hoạt động mạnh nhất vào Thứ 5 và Thứ 3 và hoạt động ít nhất vào các ngày nghỉ là Thứ 7 và Chủ Nhật. Khung giờ vàng hay khung giờ cao điểm của người dùng là 9-12h và 13-15h và giữa khung giờ vàng là 12h đột ngột giảm, tương tự như đồ thị của bài viết.
- Đặc biệt, khác với đồ thị của nội dung bài viết, khoảng thời gian từ 15h đến 21h có sự giảm đều, nhẹ và tăng nhẹ lên và đạt đỉnh vào 21h rồi giảm. Có nhiều lý do, thứ nhất, khung thời gian từ 19h-22h là khoảng thời gian nghỉ tối, sẽ có nhiều người có khả năng trực tuyến hơn. Thứ hai, thời điểm 21h là thời điểm sàn chứng khoán Mỹ NASDAQ, NYSE hoạt động, vì vậy nhiều thông tin ở đó được đem ra bàn luận hơn.

2.3.3 Wordcloud phản hồi

Tương tự như phần trên, với wordcloud, ta biểu diễn các keyword mà người dùng nói đến trong phần nội dung của bình luận.



Hình 2.10: Wordcloud phản hồi (sau khi lọc stopword)

Các keyword không có giá trị sử dụng quá nhiều, thường là các từ được sử dụng trong giao tiếp, cũng như liên quan đến chứng khoán.

2.4 Phân tích các mã cổ phiếu được đề cập

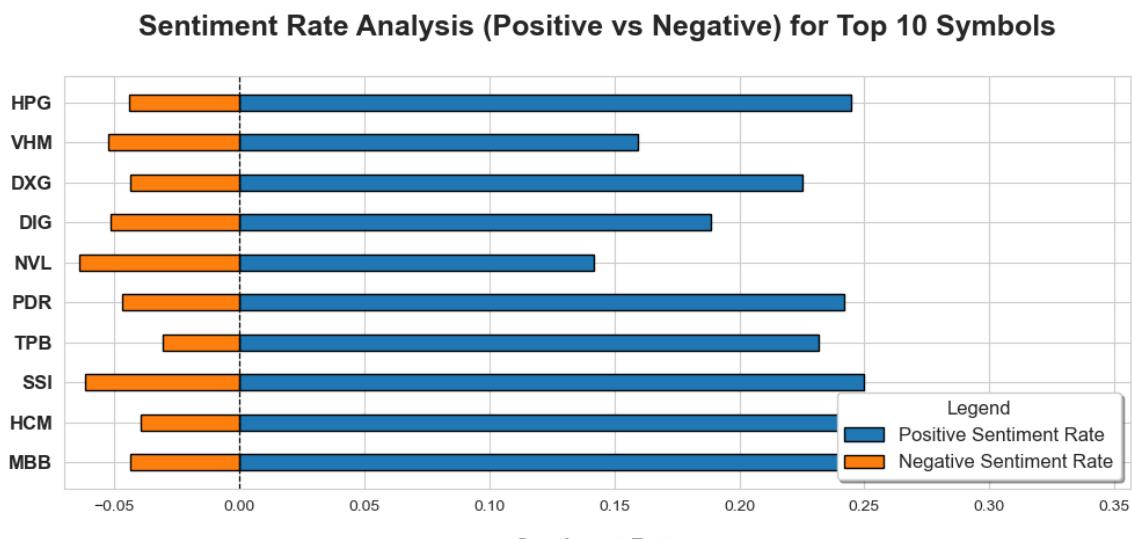
2.4.1 Quan điểm của người dùng

Lập một biểu đồ biểu diễn gồm 50 mã có số lần đề cập đến nhiều nhất, tương ứng là số quan điểm của người dùng khi nhắc đến cổ phiếu đó:



Hình 2.11: Top 50 mã cổ phiếu, và quan điểm của người dùng

Từ biểu đồ trên ta biểu thị phần trăm tiêu cực và tích cực của 10 mã chứng khoán trên theo thứ tự từ trên xuống dưới.



Hình 2.12: Tỉ lệ quan điểm người dùng của 10 mã có lượt đề cập nhiều nhất

Nhận xét:

Từ 2 plot trên ta thấy rằng:

- Các mã được nhắc đến đều là những mã có nhiều giao dịch, và đa số là những công ty dãy dầu ngành tương ứng, từ ngành Thép, cho tới Bất động sản, Ngân hàng, Chứng khoán. Nhìn chung là có độ phân bố Tích cực cao hơn nhiều so với Tiêu cực, có lẽ do dữ liệu được lấy trong khoảng thời gian biến động chưa đủ mạnh.
- **HPG, VHM, DXG** là các mã được chú ý nhiều nhất, với nhận định chủ yếu là Trung lập và Tích cực. Đây cũng chính là 3 mã có thanh khoản rất lớn trên thị trường.

2.4.2 So sánh số bài đăng đề cập đến của các mã theo ngày

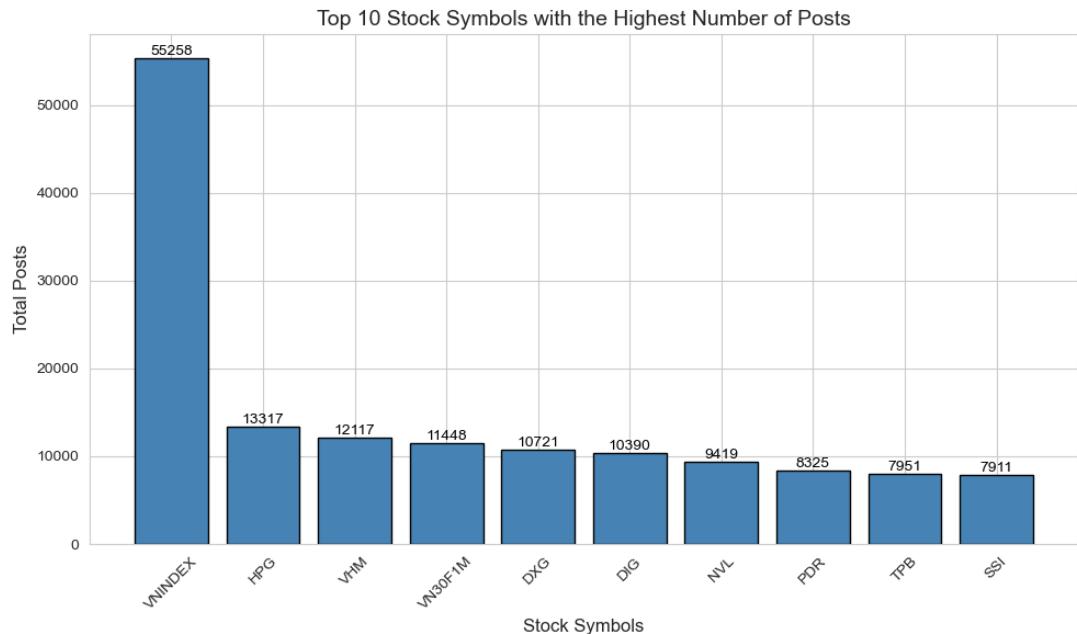


Hình 2.13: Số lượt được đề cập của các mã cổ phiếu theo ngày

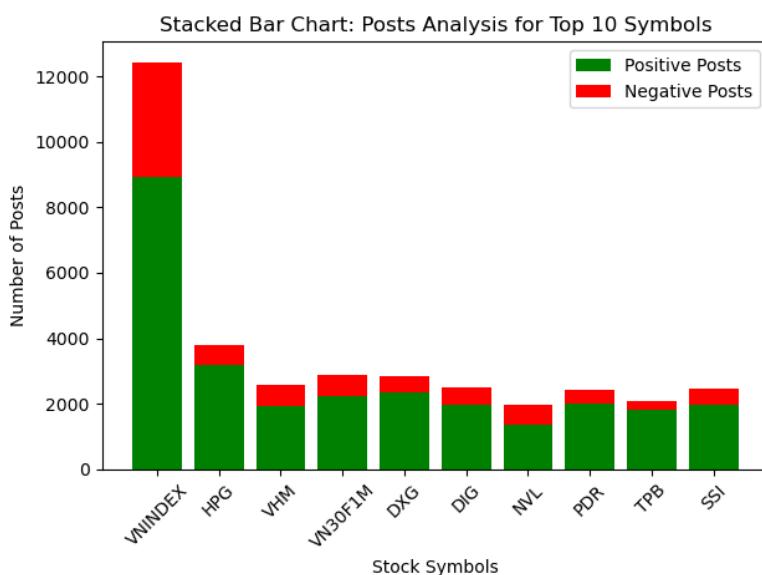
Nhận xét:

- Dễ dàng nhận thấy từ hai biểu đồ trên, VNINDEX vượt trội về số lượt đề cập hàng ngày so với các mã cổ phiếu khác.
- Không chỉ có vậy, số lượt đề cập đến mã cổ phiếu này trong suốt 2 tháng qua cũng rất đều đặn, cho thấy sự quan tâm rất lớn đến từ cộng đồng chứng khoán FireAnt.
- Điều này hoàn toàn dễ hiểu vì VNINDEX là chỉ số đại diện cho toàn bộ thị trường chứng khoán Việt Nam. Biến động của VNINDEX có ảnh hưởng lớn đến các mã chứng khoán khác, thu hút sự chú ý mạnh mẽ từ các nhà đầu tư.
- Bên cạnh VNINDEX, các mã chứng khoán đáng chú ý khác như HPG, VHM, VN30F1M, DXG và DIG cũng có số lượt đề cập ở mức khá cao.

2.4.3 Xếp hạng các mã cổ phiếu được đề cập nhiều nhất



Hình 2.14: Biểu đồ xếp hạng các mã cổ phiếu phổ biến nhất



Hình 2.15: Biểu đồ so sánh số bài viết tích cực và tiêu cực của các mã cổ phiếu phổ biến

Nhận xét:

- Từ 2 biểu đồ trên, ta một lần nữa khẳng định sự phổ biến của VNINDEX, khi chỉ số này không chỉ chiếm ưu thế về số lượng bài đăng, với tổng cộng 55258 bài, mà còn cả các bài viết tích cực và tiêu cực với lần lượt 8919 và 3518 bài.
- Chính vì nhận được sự quan tâm lớn, VNINDEX cũng là chỉ số chịu nhiều bài viết tiêu cực cao nhất. Tỷ lệ giữa số bài viết tích cực và tiêu cực về VNINDEX chỉ là 2.5, trong khi tỷ lệ này với các mã chứng khoán khác thường trên 5.

2.5 Phân tích Cộng đồng người dùng FireAnt

2.5.1 Thống kê cơ bản

Tổng số lượng người dùng xuất hiện trong bài đăng và phản hồi: 24993 người.

```
users = posts_df['userid'].unique().tolist()
users.extend(replies_df['userid'].unique().tolist())

users = set(users)
print(f"Total number of unique users: {len(users)}")
✓ 0.0s

Total number of unique users: 24993
```

Thoạt nhìn, con số 24.993 người dùng có các hoạt động đăng bài và phản hồi trong vòng 2 tháng có vẻ là lớn. Tuy nhiên, theo như dữ liệu công bố trên trang chủ, hiện tại FireAnt đã đạt mốc hơn 3 triệu người dùng [5]. Do đó, số lượng người dùng tham gia hoạt động thực tế chỉ chiếm một tỷ lệ rất nhỏ so với tổng số người dùng đã đăng ký trên nền tảng.

Điều này thể hiện rằng phần lớn người dùng có xu hướng theo dõi thông tin hơn là tương tác. Hoạt động trên nền tảng có thể tập trung vào một nhóm người dùng tích cực, trong khi đa số còn lại thụ động. Ngoài ra, số liệu này chỉ phản ánh trong 2 tháng, có thể chưa đủ để đánh giá toàn diện mức độ tham gia trên nền tảng.

Tổng số người dùng có trên 100 bài viết: 387 người.

Tổng số người dùng có trên 100 lượt phản hồi: 87 người.

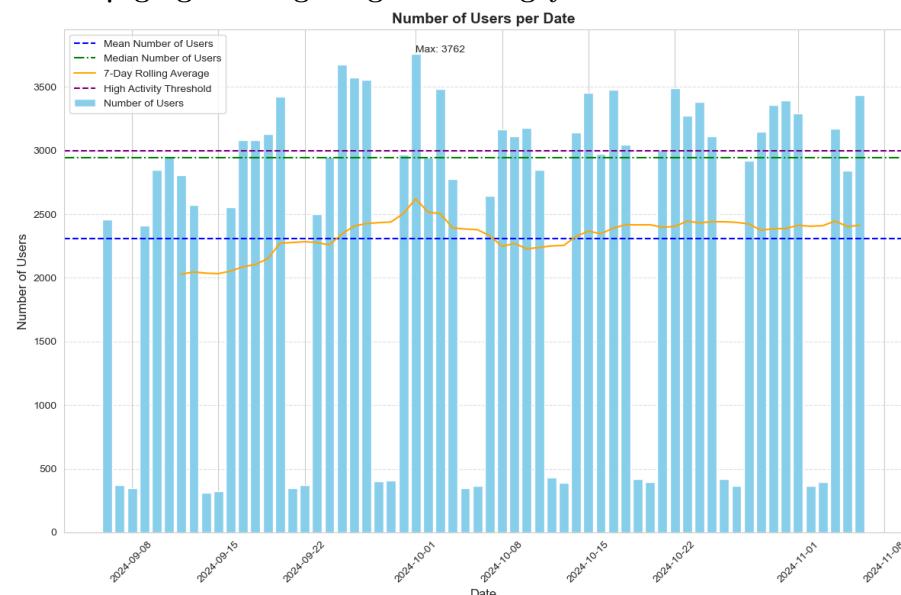
```
# number of users with more than 100 posts
active_users = user_activity[user_activity['number_of_posts'] > 100]
print(f"Number of users with more than 100 posts: {len(active_users)}")

active_repliers = user_activity_replies[user_activity_replies['number_of_replies'] > 100]
print(f"Number of users with more than 100 replies: {len(active_repliers)}")
] ✓ 0.0s

Number of users with more than 100 posts: 387
Number of users with more than 100 replies: 87
```

Điều này phản ánh rằng nội dung chủ yếu được tạo bởi một nhóm nhỏ người dùng tích cực, ngoài ra cũng khá thú vị khi số người bình luận lại ít hơn số người đăng bài.

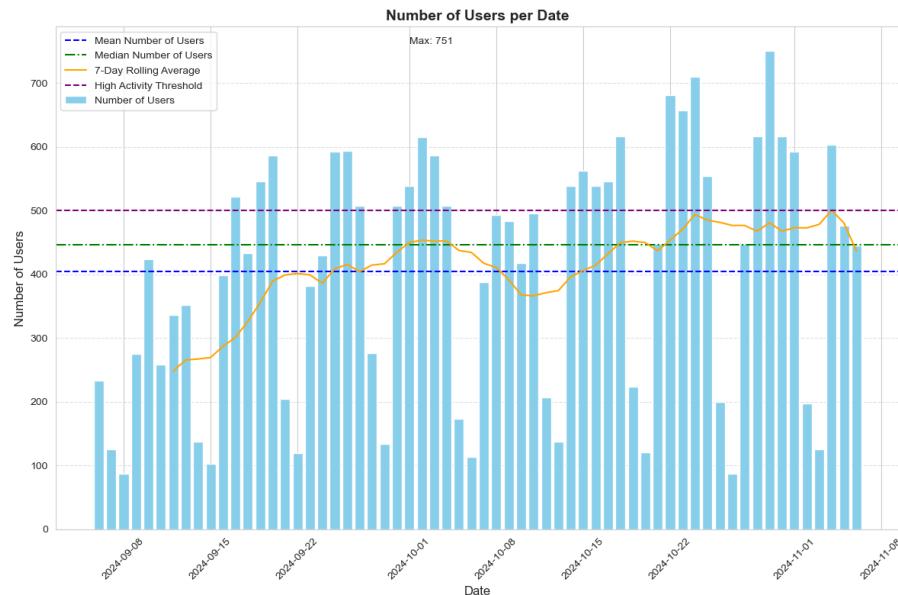
Thống kê số lượng người dùng đăng bài mỗi ngày



Hình 2.16: Biểu đồ Thống kê lượng người dùng đăng bài mỗi ngày

Dễ thấy, số lượng người hoạt động trên diễn đàn FireAnt có sự thay đổi theo chu kỳ, với 5 ngày cao điểm và 2 ngày thấp điểm. Tương quan lớn với biểu đồ số lượng bài viết/phản hồi (2.1), (2.8).

Thống kê số lượng người tham gia phản hồi theo ngày



Hình 2.17: Số lượng người tham gia phản hồi theo ngày

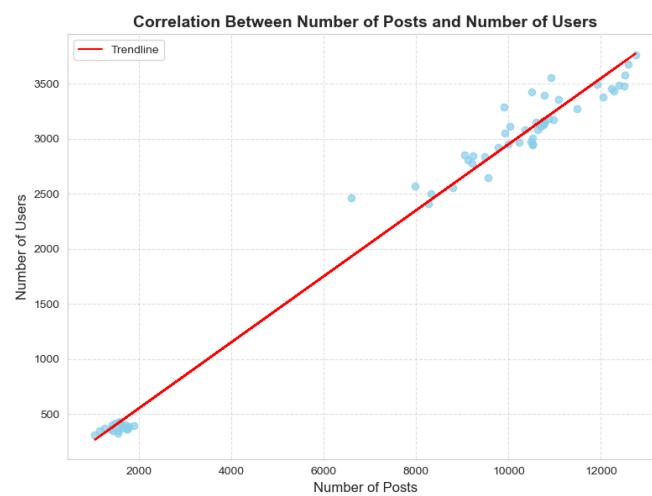
Tương tự với số người đăng bài, số người phản hồi cũng thay đổi theo chu kỳ, với 5 ngày cao điểm và giảm mạnh vào hai ngày cuối tuần (Thứ Bảy và Chủ Nhật). Tuy nhiên, số người tham gia phản hồi nhìn chung vẫn thấp hơn đáng kể so với số người đăng bài.

Hệ số tương quan giữa số người dùng và số bài đăng

```

152 |     print('The correlation between the number of posts and the number of users is:', user_per_date['number_of_posts'].corr(user_per_date['number_of_users']))
152 |     ✓ 0.0s
...   The correlation between the number of posts and the number of users is: 0.9935814881404628

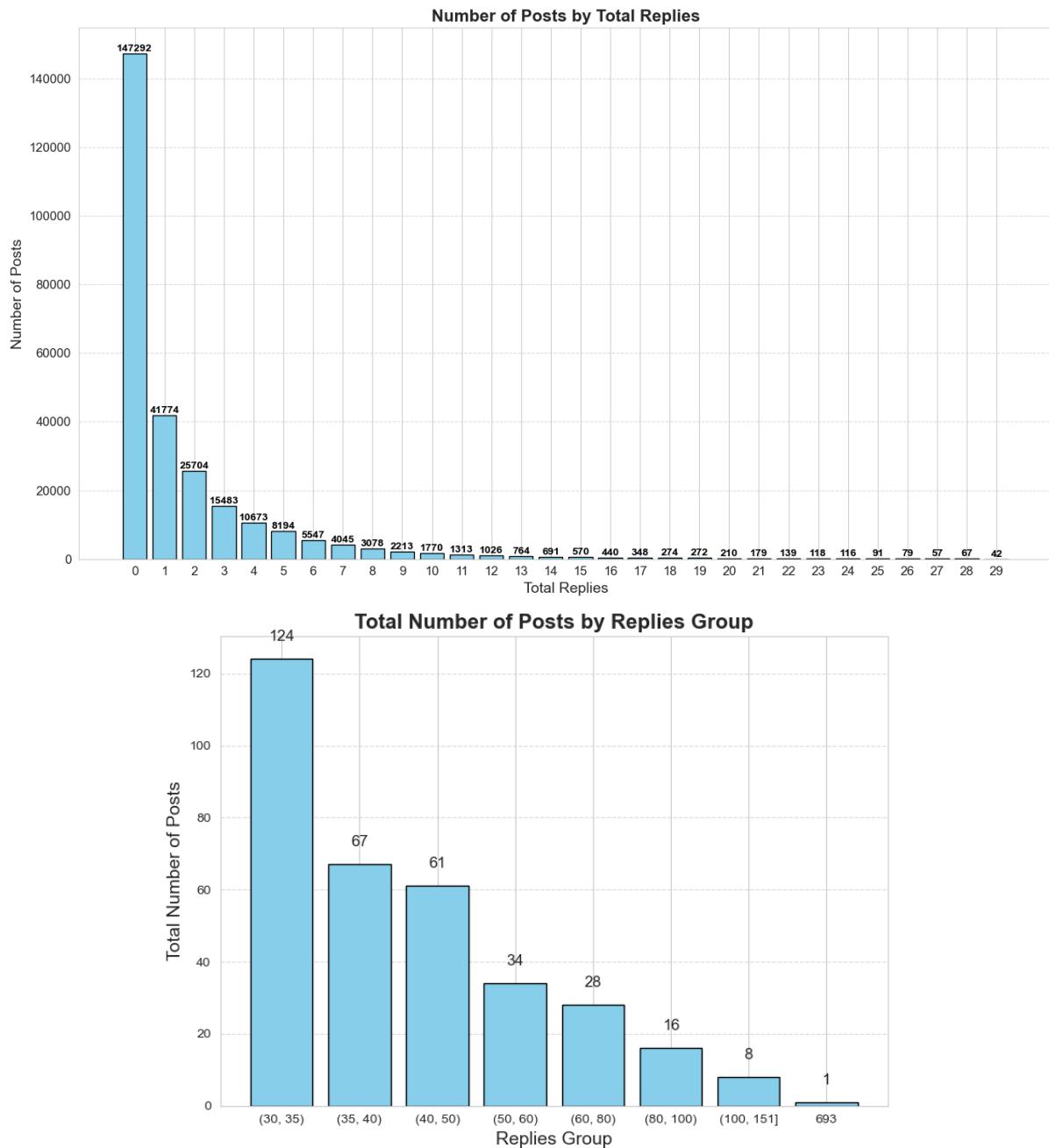
```



Hình 2.18: Tương quan giữa Số lượng bài viết và Số người dùng

Ta thấy số lượng bài đăng có mối liên hệ chặt chẽ với số lượng người đăng bài, thể hiện qua hệ số tương quan lên tới 0.9936. Đây là điều hoàn toàn hiển nhiên, vì số lượng bài đăng tăng đồng nghĩa với việc có nhiều người tham gia hoạt động hơn.

Thống kê Số phản hồi trong mỗi bài đăng



Hình 2.19: Thống kê Số phản hồi trong mỗi bài đăng

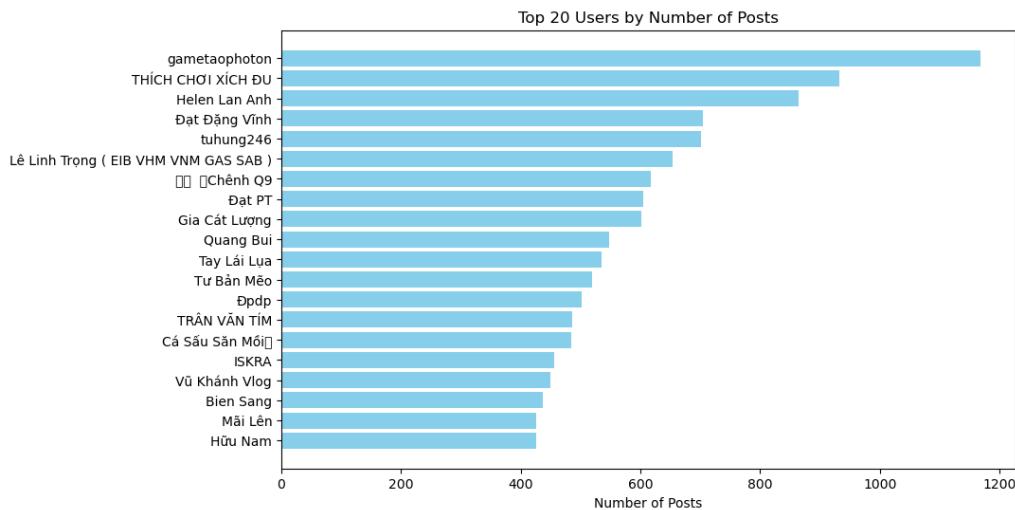
Ta thấy phần lớn các bài đăng trên diễn đàn FireAnt đều không có hoặc có rất ít phản hồi. Cụ thể trong tổng số hơn 270 nghìn bài viết mà ta đã cào được, có đến 147.292 ($\approx 54\%$) bài viết là không có phản hồi. Số bài có từ 5 phản hồi trở xuống chiếm đến 91.26%. Điều này cho thấy mức độ tương tác của người dùng là tương đối thấp.

Một số bài viết đạt từ 50 lượt phản hồi trở lên, hầu hết là những bài viết “tóm tắt diễn biến thị trường trong ngày”, thu hút được nhiều người dùng vào bình luận và trao đổi.

Tuy vậy vẫn có một trường hợp ngoại lệ xảy ra, vào ngày 26/9/2024, người dùng **Doan Nhan** đã đăng **một bài viết** chỉ với 5 chữ “Thức dậy đi chú Đạt” nhưng có đến 693 lượt phản hồi. Thực ra hầu hết đoạn phản hồi ở đây là cuộc tranh cãi qua lại giữa người này và một người dùng khác trong 5 tuần, còn “chú Đạt” ở đây là Chủ tịch của công ty Phát Đạt (PDR).

2.5.2 Thống kê các hoạt động nổi bật của người dùng

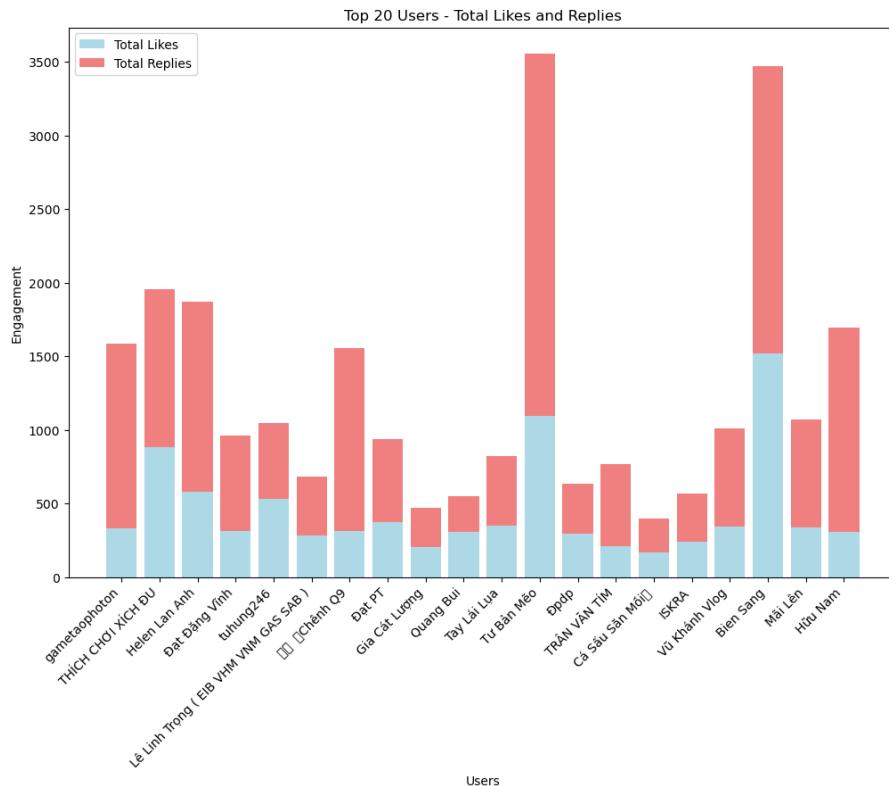
Xếp hạng người dùng có số bài đăng lớn nhất



Hình 2.20: Thống kê Top 20 người dùng đăng bài nhiều nhất

Dễ thấy, gam**ton, Thí**đu, hay Hel**Anh là những người dùng có hoạt động tích cực nhất, nổi trội hơn hẳn so với những người còn lại trong danh sách. Tuy vậy, các tài khoản này thường xuyên đăng tải nội dung với tần suất rất cao, chủ yếu là các bài viết có tính chất spam hoặc chia sẻ những thông tin lặp đi lặp lại. Những hành vi này cho thấy hoạt động của các tài khoản trên không thực sự mang lại giá trị lớn cho cộng đồng, mà chủ yếu chỉ để tăng sự hiện diện cá nhân hoặc phục vụ mục đích riêng.

Xếp hạng người dùng có số lượt phản hồi và lượt thích cao nhất

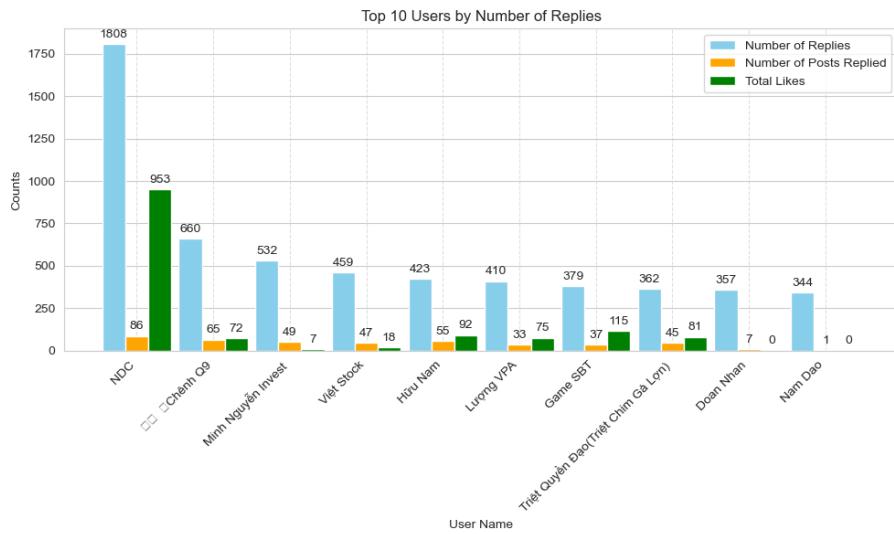


Hình 2.21: Thống kê Top 20 người dùng có nhiều lượt thích và bình luận nhất

Ở biểu đồ này, ta có thể thấy không phải gam**ton hay Hel**Anh là những người đứng đầu về tầm ảnh hưởng, mà chính Tư**Méo và Biên Sang mới thực sự nổi bật. Mặc dù số bài đăng của hai người dùng này chưa bằng một nửa so với gam**ton, nhưng tổng số lượt tương tác mà họ nhận được lại gấp đôi.

Điều này phần lớn là do các bài viết của Tư**Méo và Biên Sang thường mang tính phân tích sâu sắc về thị trường chứng khoán và chia sẻ kinh nghiệm đầu tư thực tiễn (Tư**Méo đã cập nhật 46 mã cổ phiếu trong các bài đăng của mình).

Xếp hạng người dùng tích cực phản hồi các bài viết nhất



Hình 2.22: Thống kê Top 10 người dùng bình luận nhiều nhất

Ta thấy người dùng NDC vượt trội cả về số bài phản hồi và tổng lượt thích. Mặc dù chỉ bình luận cho 86 bài viết, nhưng đã viết hơn 1800 bài và nhận được 953 lượt thích, điều này cho thấy mức độ tương tác rất cao với cộng đồng. Có nhiều người dùng chỉ bình luận, chứ không thường xuyên đăng bài hay thích một bài.

Xếp hạng người dùng có độ chính xác quan điểm cao nhất

Ta sẽ đặt ra một phép đo, thể hiện tầm suất bài đăng của một người dùng cụ thể “dự đoán đúng” giá cổ phiếu trong phiên giao dịch tiếp theo. Ví dụ, khi bài đăng thể hiện quan điểm Tiêu cực, và giá cổ phiếu đi xuống; hay khi bài viết Tích cực và giá cổ phiếu đi lên; lúc đó bài đăng này được coi là “dự đoán đúng”.

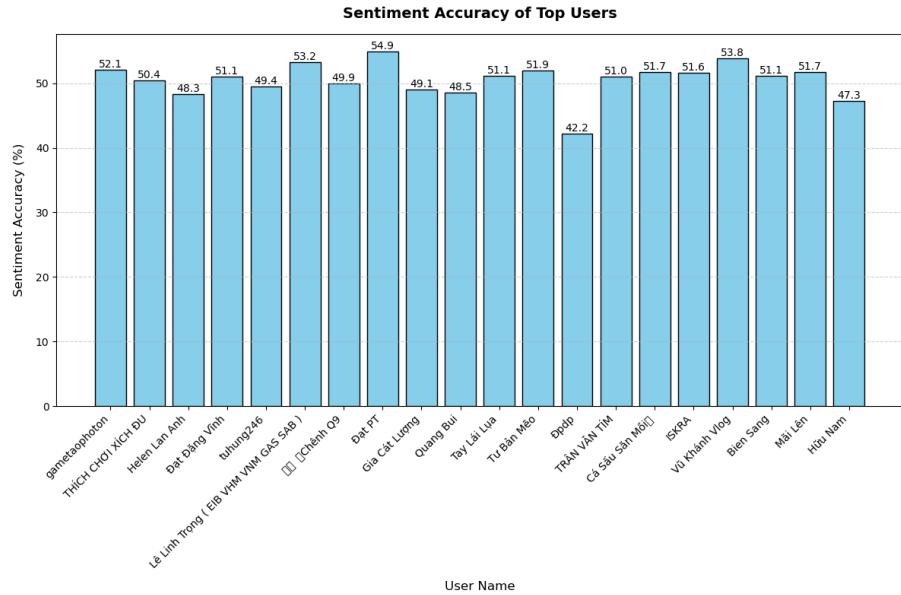
$$\text{Độ chính xác} = \frac{\text{Số bài dự đoán đúng}}{\text{Tổng số bài có Quan điểm}}$$

```
def calculate_sentiment_accuracy(symbol_name=None, user_name=None):
    if (symbol_name is None and user_name is None) or (symbol_name is not None and user_name is not None): # Kiểm tra đầu vào
        return None
    # Lấy dữ liệu cần thiết
    if symbol_name:
        df = symbols_data_dict[symbol_name]
    else:
        df = new_posts_df[new_posts_df['username'] == user_name]

    df = df.dropna(subset=['next_price', 'next_time']) # Loại bỏ các dòng không có giá trị giá cổ phiếu phiên tới
    sentiment_map = {'positive': 1, 'negative': -1} # Chuyển đổi sang dạng số
    df['sentiment_numeric'] = df['sentiment'].map(sentiment_map)

    df['price_change_category'] = ((df['next_price'] - df['price']) >= 0).astype(int).replace(1, -1) # Phân loại giá cổ phiếu tăng hay giảm
    df['sentiment_accuracy'] = (df['sentiment_numeric'] == df['price_change_category']) # So sánh quan điểm bài dự đoán với thực tế

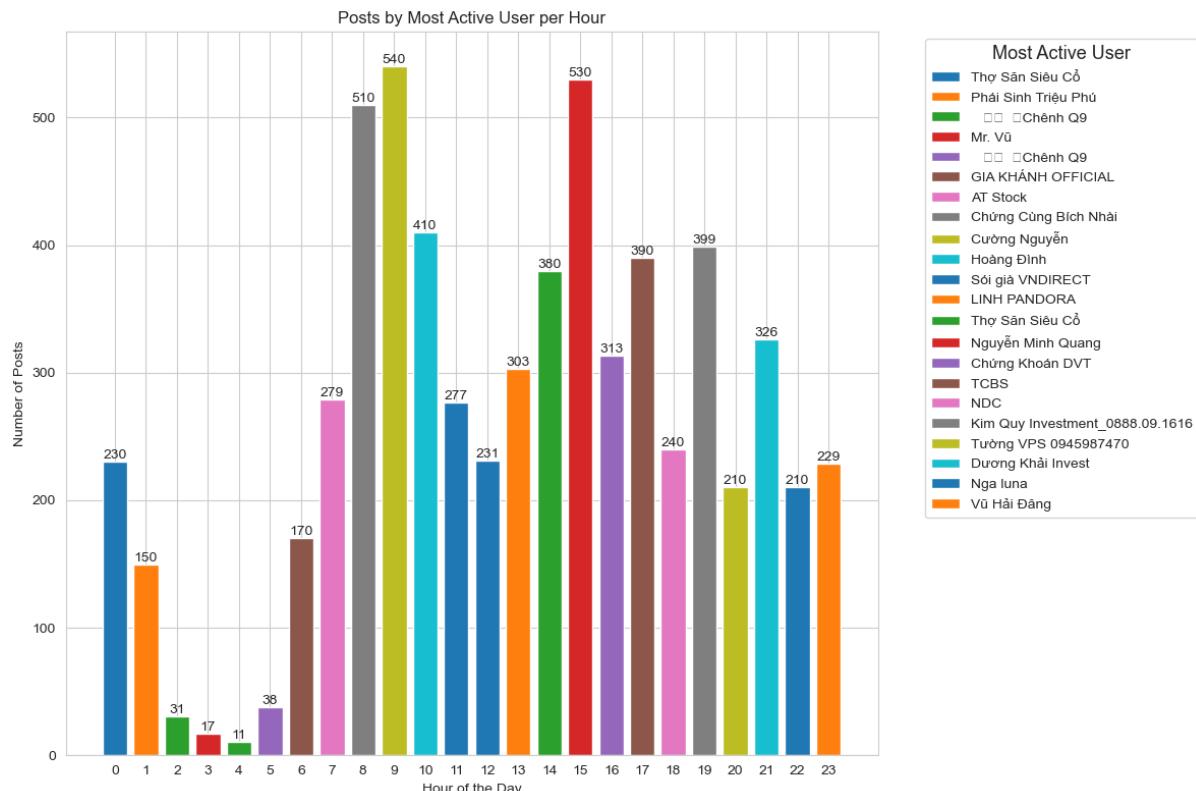
    return df['sentiment_accuracy'].mean() # Trả về tỷ lệ chính xác
```



Hình 2.23: So sánh Độ chính xác quan điểm của người dùng hay đăng bài

Ta nhận thấy độ chính xác của các người đăng bài nhiều nhất đều không cao. Người dùng đứng đầu chỉ có độ chính xác là 54.9%. Nhìn chung các người dùng khác cũng chỉ có độ chính xác loanh quanh 50%, do đó ta có thể kết luận, việc dựa trên các bài viết tiêu cực và tích cực của người dùng hay đăng bài để đoán giá là khó khả thi.

Thống kê những người dùng phổ biến nhất theo từng khung giờ



Hình 2.24: Thống kê những người dùng phổ biến nhất theo từng khung giờ

Ở đây, ta có thể thấy một số cái tên quen thuộc đã xuất hiện từ trước. Đặc biệt, Ché**Q9 nổi bật với việc đăng bài vào những khung giờ rất muộn như 2h và 4h sáng.

2.5.3 Phân tích Văn hóa ứng xử của người dùng

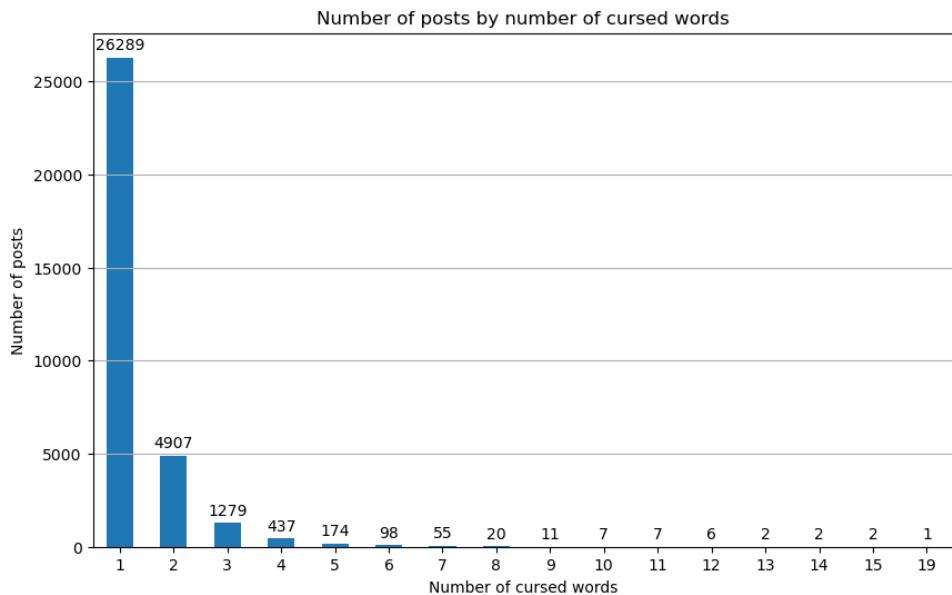
Sự văn minh và thân thiện giữa người dùng và người sáng tạo nội dung là yếu tố quan trọng quyết định sự phát triển bền vững của một cộng đồng. Vì vậy, chúng ta sẽ cùng phân tích xem FireAnt có thực sự là một môi trường giao tiếp tích cực và xây dựng, nơi các cuộc thảo luận diễn ra trong không khí tôn trọng và xây dựng, hay liệu vẫn tồn tại các vấn đề liên quan đến hành vi không phù hợp, như việc sử dụng ngôn từ tục tĩu hoặc thiếu văn minh.

Ta sẽ sử dụng nguồn từ chửi tục trong file Vietnamese_cursed_words.txt, biến đổi từ file gốc [1]. Trước hết ta cần phải lọc dấu câu, biến toàn bộ ký tự trong cột originalContent thành chữ thường. Sau đó chỉ cần làm một hàm đếm số lượng từ chửi tục:

```
cursed_words_list = (pd.read_csv('Vietnamese_cursed_words.txt'))['word'].tolist() # Danh sách từ tục tiếng Việt
cursed_words_list = set(cursed_words_list)

def count_cursed_words(text):
    count = 0
    for word in text.split():
        if word in cursed_words_list: # Tách từ trong câu
            count += 1
    return count
post_content['cursed_words_count'] = post_content['originalContent'].apply(count_cursed_words) # Apply hàm cho từng dòng
```

Số lượng bài đăng dựa trên số lượng các từ chửi tục

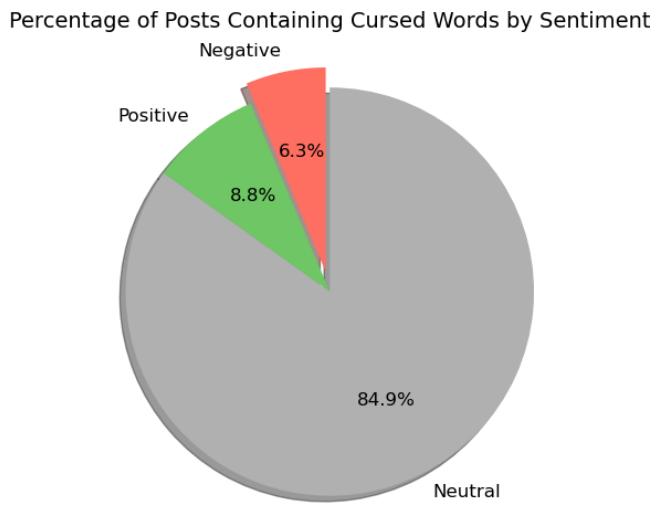


Hình 2.25: Số lượng bài đăng dựa trên số lượng các từ chửi tục

Nhận xét:

- Ta thấy trong các bài đăng kém văn minh, phần lớn chỉ có 1 từ chửi thề, số từ chửi thề càng lớn, số bài đăng càng giảm.
- Số từ chửi tục lớn nhất có trong 1 bài đăng là 19, cho thấy mức độ cay cú của người viết bài.

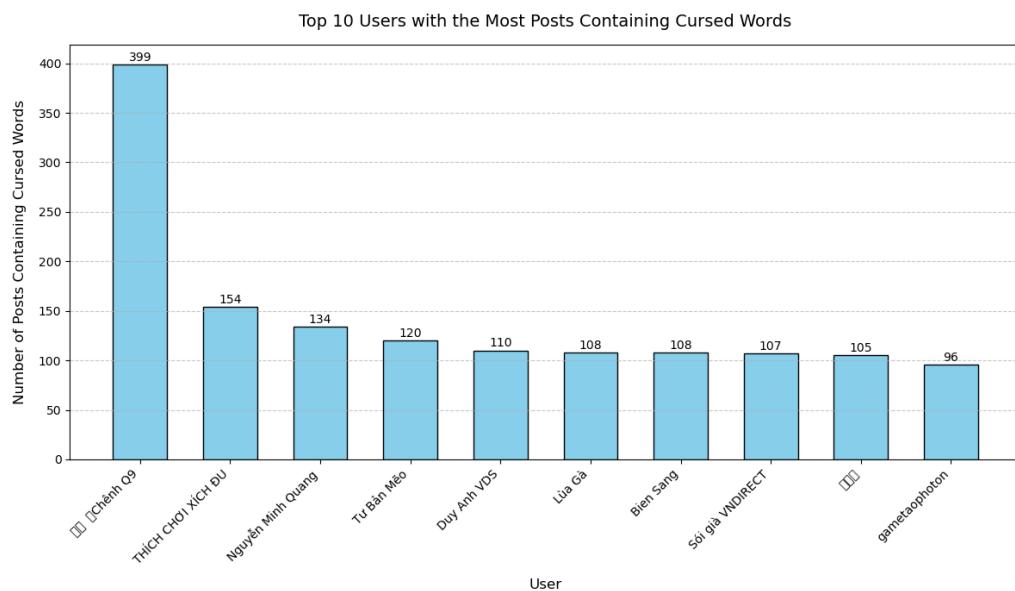
Tỉ lệ Quan điểm của những bài có chửi tục



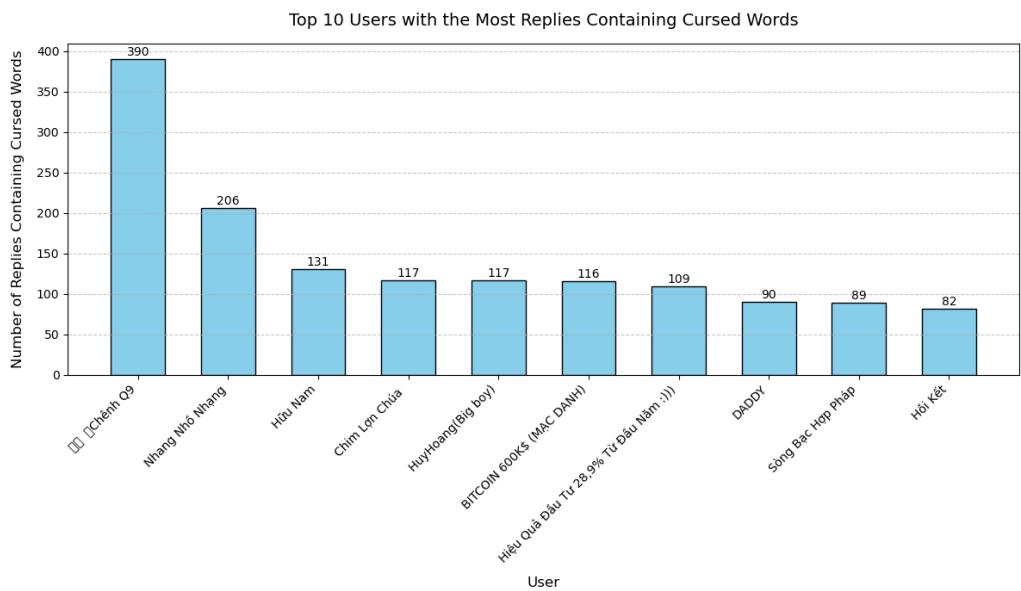
Hình 2.26: Tỉ lệ Quan điểm của những bài có chửi tục

Dễ thấy đa phần các bài viết, đều là các bài viết trung lập. Điều này đa phần do người dùng ở diễn đàn FireAnt quên không phân định quan điểm bài viết trước khi đăng. Điều khá thú vị là số lượng bài Tích cực chửi tục lại nhiều hơn Tiêu cực, điều này chứng tỏ rằng, người dùng không chỉ mỗi chửi tục khi thể hiện quan điểm Tiêu cực.

Xếp hạng các người dùng kém văn minh nhất



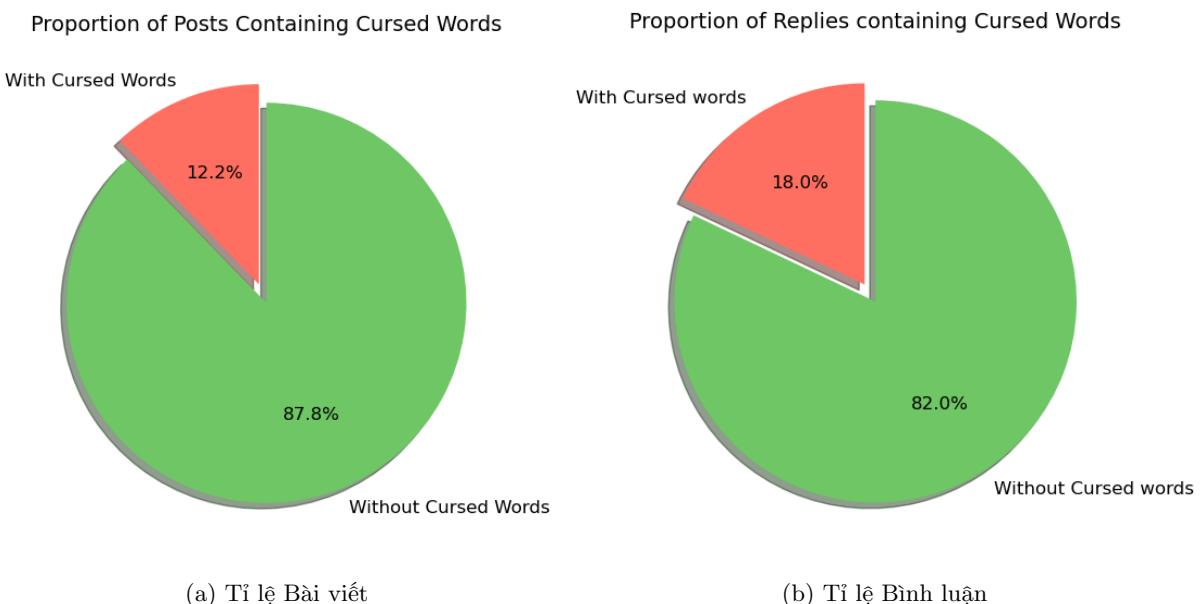
Hình 2.27: Xếp hạng những người dùng có bài viết chửi tục nhiều



Hình 2.28: Xếp hạng những người dùng có bình luận chửi tục nhiều

Người dùng Chèn Q9 nổi bật nhất với 399 bài viết và 390 bình luận chứa các từ ngữ chửi tục. Mặc dù đây là những người có số lượng bài viết ...vô văn hoá nhiều nhất, nhưng họ cũng là những người có số lượng lượt thích và bình luận cao. Có thể những nội dung tiêu cực, hoặc sử dụng ngôn ngữ mạnh thường thu hút sự chú ý và tương tác cao từ cộng đồng.

So sánh tỷ lệ các bài đăng văn minh và các bài đăng kém văn minh



Hình 2.29: Tỉ lệ chửi tục trong bài viết/bình luận

Nhận xét:

- Từ hai biểu đồ so sánh trên, ta có thể ghi nhận một dấu hiệu tích cực về mức độ lịch sự trong các bài viết của người dùng.
- Cụ thể, 88% các bài đăng gốc và 82% các bài phản hồi là những bài viết lịch sự, không chứa từ ngữ chửi tục.
- Điều này cho thấy phần lớn người dùng vẫn giữ được thái độ văn minh trong giao tiếp trên mạng, đặc biệt trong các bài đăng gốc.

Chương 3

Phân tích Dữ liệu Thị trường Chứng khoán Việt Nam

Ngoài dữ liệu từ diễn đàn FireAnt, thị trường chứng khoán Việt Nam là đối tượng phân tích tiếp theo của ta. Thị trường cũng cung cấp lượng data “màu mỡ” để ta có thể khai thác, và củng cố giả thuyết.

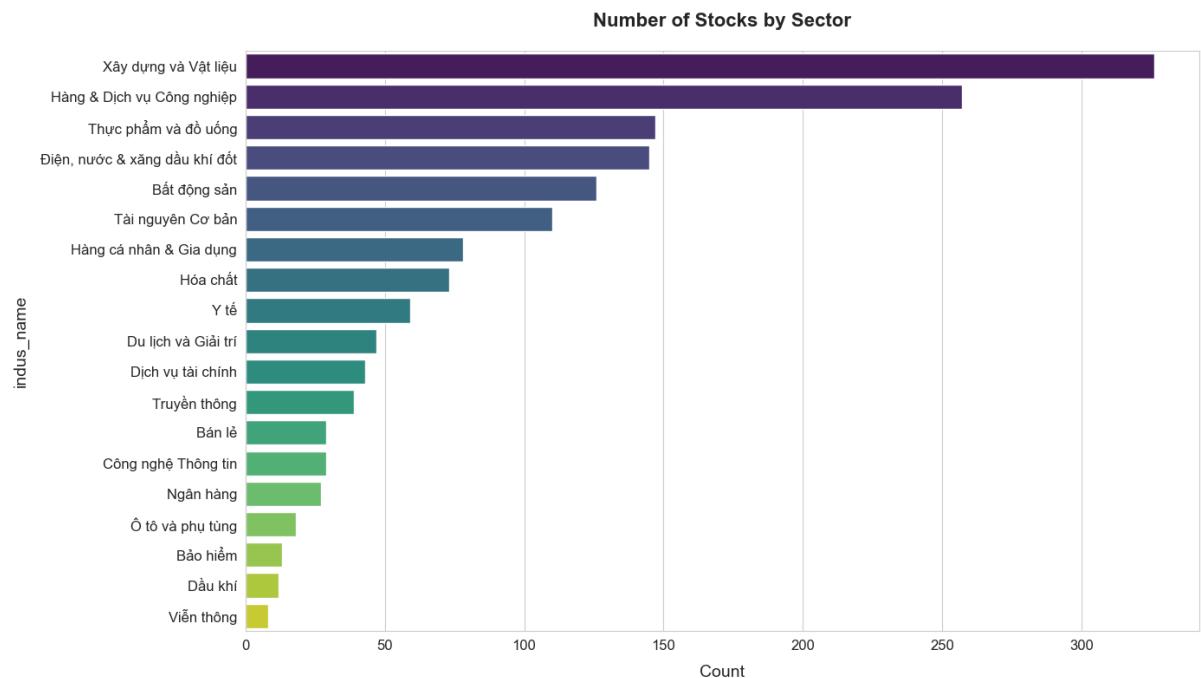
3.1 Phân tích các mã cổ phiếu theo ngành

3.1.1 Phân bố giữa các ngành



```
# Liệt ngành trong list_stock.csv và số lượng mã chứng khoán của mỗi ngành
sectors = pd.read_csv('list_stock.csv')
sectors['indus_name'] = sectors['indus_name'].str.strip()
sector_counts = sectors['indus_name'].value_counts().reset_index()
sector_counts.columns = ['indus_name', 'count']
sector_counts
```

Ta sẽ bắt đầu bằng việc nhập file `list_stock.csv`. Ta sẽ sử dụng dữ liệu về nhóm ngành của từng công ty niêm yết trong danh sách.



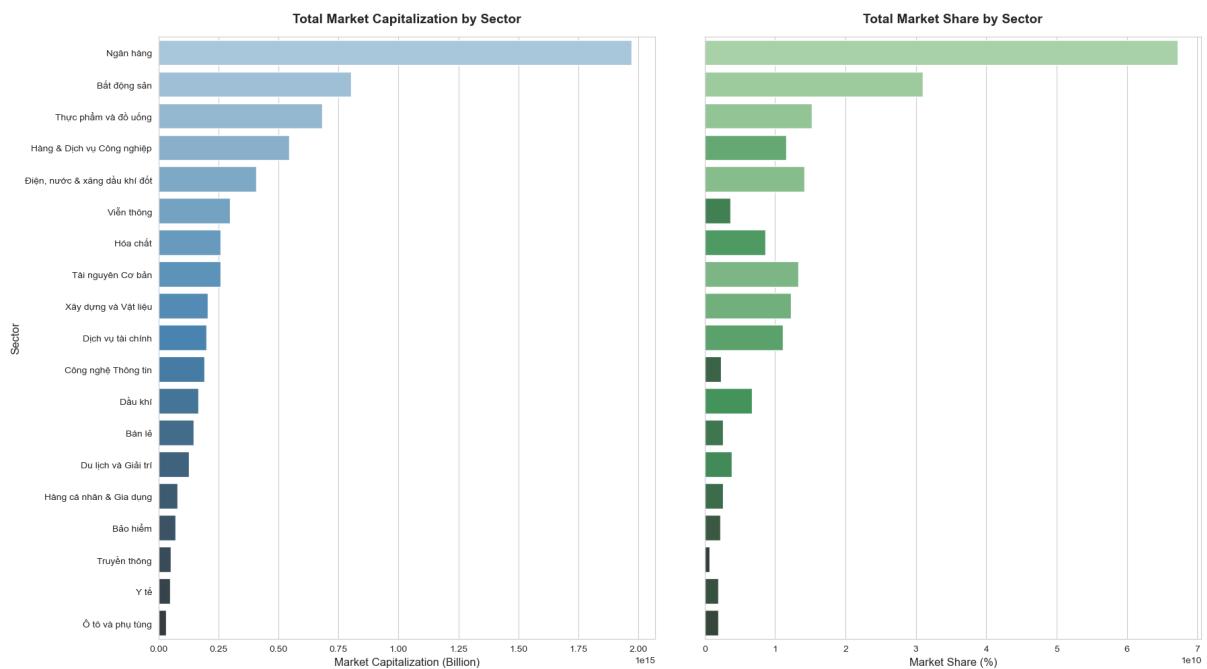
Hình 3.1: Số lượng mã cổ phiếu trong ngành

Nhận xét:

- Ngành **Xây dựng và Vật liệu** là ngành có số lượng cổ phiếu cao nhất, vượt trội so với các ngành khác. Sau đó là **Hàng Dịch vụ Công nghiệp**, có số lượng cổ phiếu gần tương đương với ngành dẫn đầu. Ngành **Thực phẩm và đồ uống và Điện, nước xăng dầu khí đốt** cũng có số lượng cổ phiếu đáng kể, nhưng thấp hơn so với hai ngành đầu tiên. Các ngành như "**Viễn thông**", "**Dầu khí**", và "**Bảo hiểm**" có số lượng cổ phiếu thấp nhất trong biểu đồ.
- Tóm lại, có sự chênh lệch rõ rệt giữa các ngành, và trong phân bố ngành kinh tế Việt Nam. Một số ngành có số lượng cổ phiếu áp đảo trong khi nhiều ngành khác tương đối thấp.

3.1.2 Vốn hóa thị trường, số cổ phiếu lưu hành

Vốn hóa và số cổ phiếu lưu hành luôn là những tiêu chí có sức ảnh hưởng lớn tới việc mua bán mã chứng khoán. Dưới đây là biểu đồ mô tả 2 giá trị quan trọng đó với từng ngành:



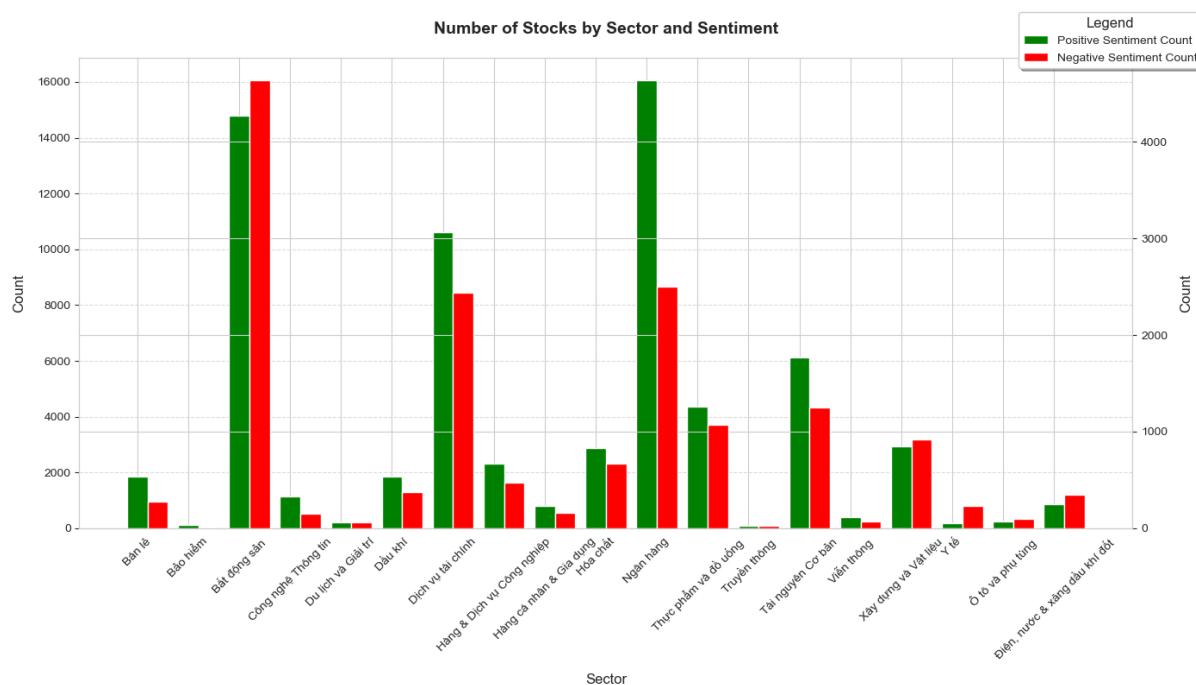
Hình 3.2: Vốn hóa thị trường và Số cổ phiếu lưu hành

Nhận xét:

- Ngân hàng** là ngành có lượng vốn và cổ phiếu lưu hành nhiều nhất dẫu cho số mã chứng khoán của ngành này khá khiêm tốn so với tất cả các ngành khác. Theo sau **Ngân hàng** là các ngành như **Bất động sản**, **Thực phẩm và đồ uống**, ...
- Có thể thấy rằng không phải vốn hóa cao cũng đồng nghĩa với số cổ phiếu lưu hành cao.

3.1.3 Quan điểm người dùng

Tương tự, ta lấy cột tên ngành từ `list_stock.csv` kết hợp để lập biểu đồ với số lượng bài viết thể hiện quan điểm Tích cực hoặc Tiêu cực.



Hình 3.3: Biểu đồ phản ánh ngành và Quan điểm của người dùng

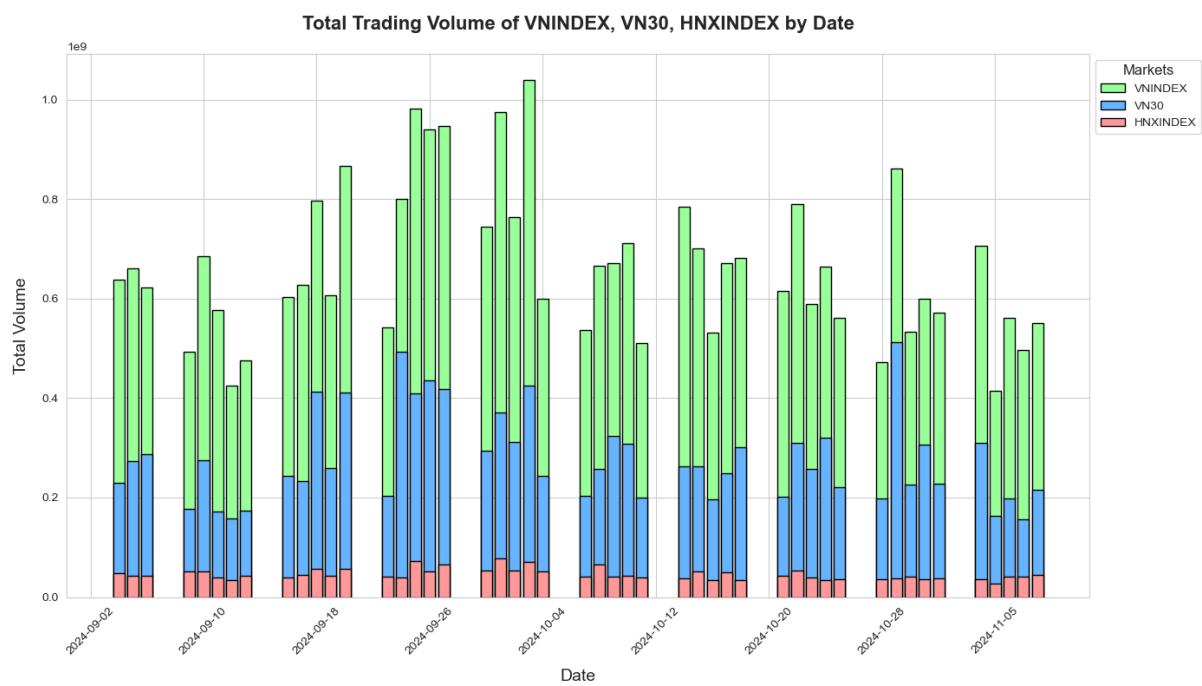
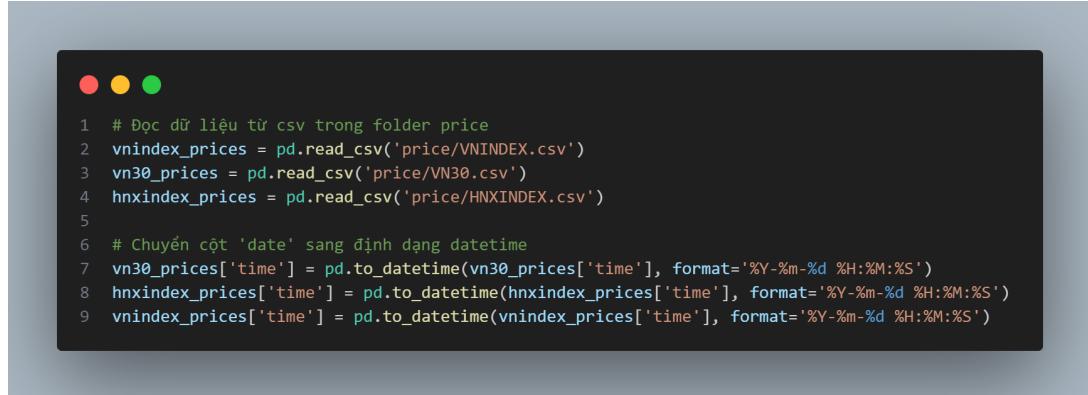
Nhận xét:

- Các ngành có số lượng bài viết nhiều là **Bất động sản**, **Dịch vụ tài chính (Chứng khoán)**, **Ngân hàng**, **Tài nguyên**, **Xây dựng và Vật liệu**.
- Đa phần các ngành đều có lượng bài viết Tích cực cao hơn Tiêu cực, nhưng có một vài ngoại lệ như ngành **Bất động sản**, do đa số các mã trong ngành này chịu ảnh hưởng xấu trong khoảng thời gian lấy dữ liệu.
- Ngành **Ngân hàng** có chỉ số Tích cực gấp đôi, do đây là ngành được hưởng lợi và các mã trong ngành đều tăng trưởng rất tốt trong khoảng thời gian lấy dữ liệu.

3.2 Phân tích Tương quan

3.2.1 Tương quan trong Thị trường Nội địa

Trên thị trường chứng khoán Việt Nam, có 3 chỉ số thị trường lớn là VNINDEX, VN30 và HNXINDEX. Dưới đây là khối lượng giao dịch của 3 mã theo ngày.



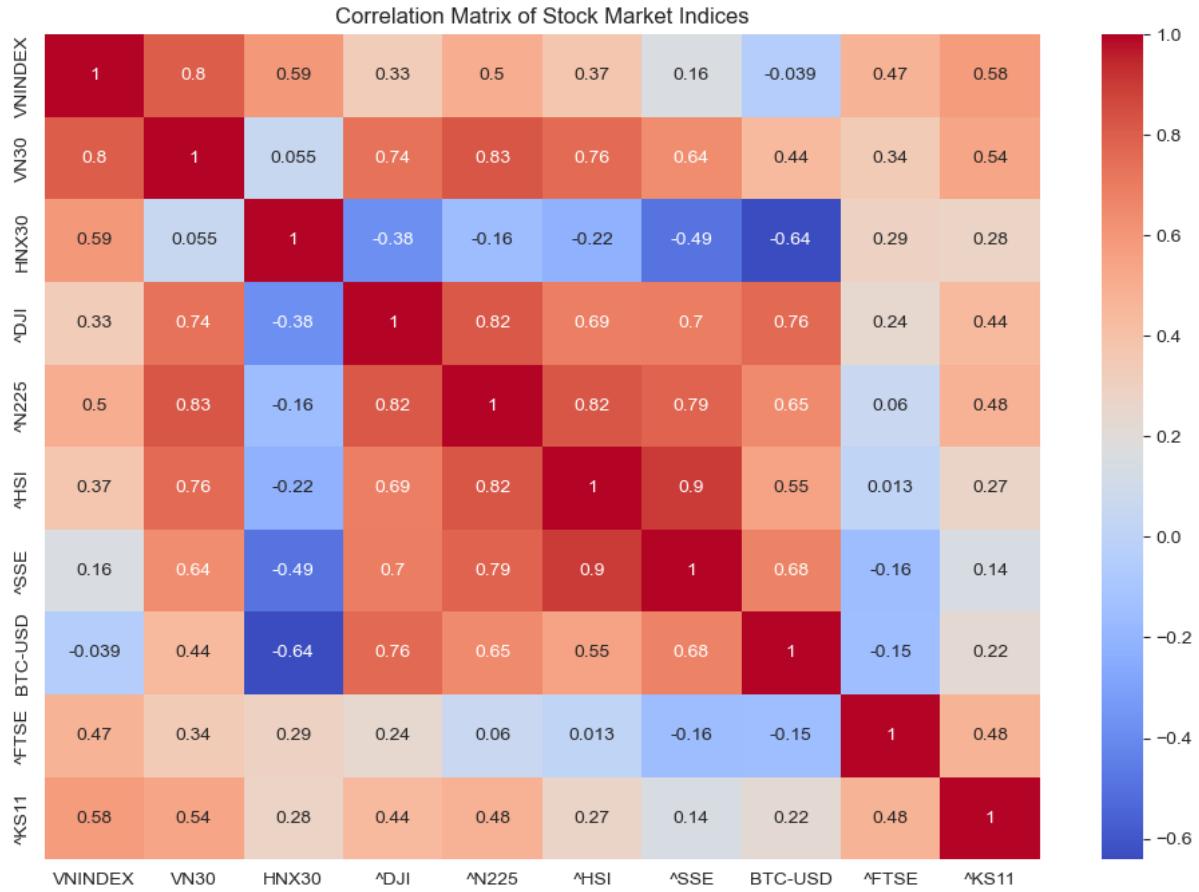
Hình 3.4: Khối lượng giao dịch của VN30, VNINDEX, HNXINDEX

Nhận xét:

- Có xảy ra nhiều biến đổi mạnh, diễn ra vào cuối tháng 9 đầu tháng 10, khi chỉ số điểm VNINDEX giảm từ 1300 xuống 1265.
- Nhìn chung khối lượng của cả 3 chỉ số thị trường không có sự biến đổi quá rõ rệt, do có sự liên quan tới nhau và cùng chịu ảnh hưởng ở chung một nền kinh tế. Có thể thấy khá rõ sự đồng pha của các chỉ số thị trường chứng khoán Việt Nam.

3.2.2 Tương quan với Thị trường Nước ngoài

Có thể thấy rằng nhịp đập của thị trường chứng khoán Việt Nam là một bản giao hưởng hài hòa, ma trận tương quan dưới đây sẽ vừa thể hiện rõ điều đó, cũng như là bổ sung thêm những thông tin quan trọng về mối tương quan với thị trường ngoài nước. Ta sẽ sử dụng dữ liệu từ [investing.com](https://www.investing.com), chúng là phần dữ liệu được cào thêm:



Hình 3.5: Ma trận tương quan giữa mã thị trường trong nước và quốc tế

Nhận xét:

- VN30 và N225 (0.83): Mối tương quan cao giữa VN30 và Nikkei 225 cho thấy sự liên kết của thị trường Việt Nam với thị trường Nhật Bản.
- VN30 và DJI (0.74): Chỉ số VN30 có mối tương quan mạnh với Dow Jones, cho thấy sự ảnh hưởng của thị trường Mỹ đến thị trường châu Á.
- VN30 và HSI (0.76): Thị trường Việt Nam có mối liên hệ khá mạnh với Hang Seng (Hồng Kông).
- VN30 và SSE (0.64): VN30 cũng có sự gắn kết với thị trường Trung Quốc.
- VNINDEX và VN30 (0.8): Chỉ số chung của Việt Nam có mối tương quan chặt chẽ với chỉ số nhóm vốn hóa lớn (VN30), điều này dễ hiểu do VN30 chiếm tỷ trọng cao trong VNINDEX.
- HNX30 và BTC (-0.64): Có sự đối lập hiệu suất của HNX30 và tiền điện tử (Bitcoin).
- HNX30 và DJI (-0.38): HNX30 cũng có mối tương quan âm yếu với chỉ số Dow Jones, cho thấy ít liên hệ giữa sàn chứng khoán Hà Nội và thị trường Mỹ.
- Nhìn chung, các thị trường trong khu vực Châu Á (VN30, HSI, N225, SSE, KS11) có sự đồng pha mạnh, cho thấy sự ảnh hưởng qua lại trong khu vực. Nhưng HNX30 lại tương đối khác biệt so với các chỉ số khác.

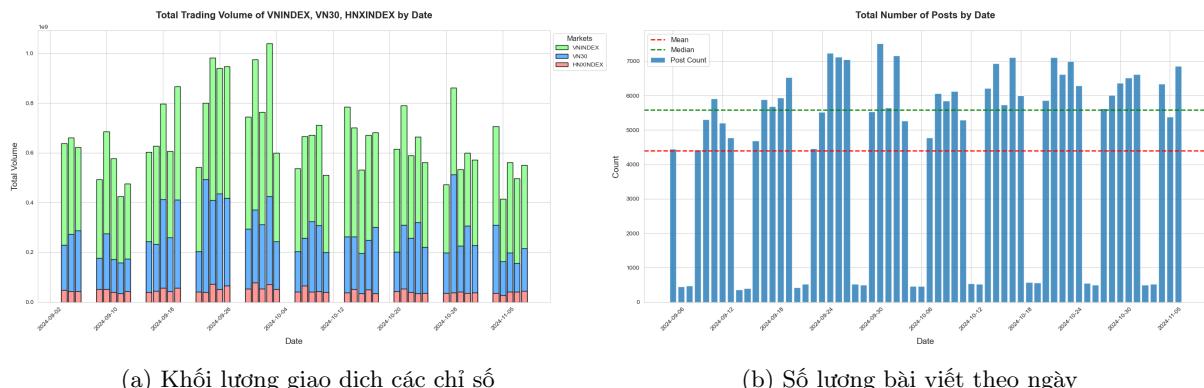
Chương 4

Phân tích Tương quan giữa FireAnt và Thị trường Chứng khoán Việt Nam

4.1 Khối lượng giao dịch cổ phiếu với Số lượng bài viết

Đến với yếu tố đầu tiên mà chúng em muốn đề cập đến ở đây là liệu rằng có tồn tại mối tương quan nào giữa số lượng bài viết và khối lượng giao dịch của 3 chỉ số chứng khoán hàng đầu Việt Nam như VN30, VNINDEX, HNXINDEX. hay không.

Ta cùng nhìn qua 2 biểu đồ đã được đề cập từ trước, biểu đồ (3.4) thể hiện khối lượng giao dịch của VNINDEX, VN30, HNXINDEX và biểu đồ (2.1) thể hiện số lượng bài viết theo ngày:

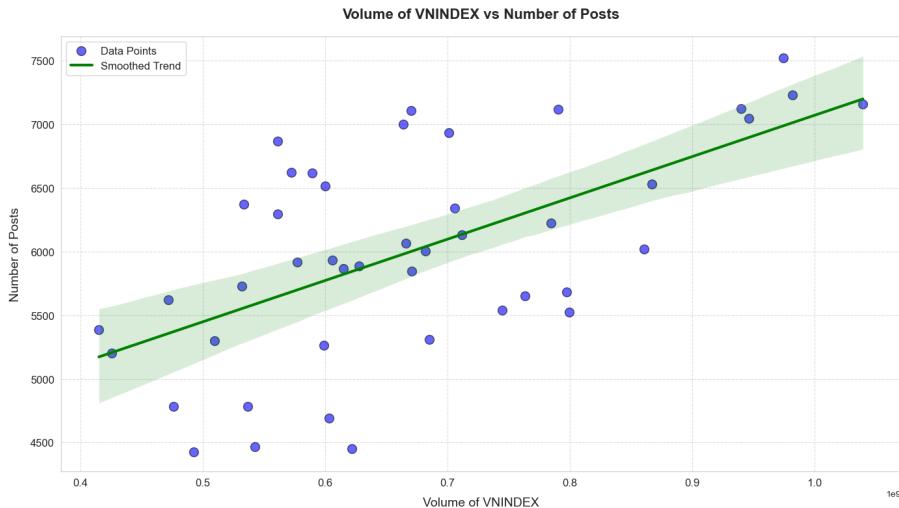


Dễ dàng nhận thấy có một sự tương đồng. Ta nhận xét và có thể đưa ra giả định rằng, khi số lượng bài viết tăng (đặc biệt nếu là ngày quan trọng), khối lượng giao dịch có thể tăng do nhà đầu tư chú ý và phản ứng với thông tin hay có xuất hiện sự kiện thu hút dòng vốn mới vào cổ phiếu liên quan. Ngược lại, nếu một cổ phiếu ít được chú ý (số lượng bài viết ít), khả năng khối lượng giao dịch sẽ thấp do sự quan tâm ít hơn.

Ta sẽ đặt giả thuyết rằng: Số lượng bài viết trong ngày có tương quan thuận với khối lượng giao dịch của thị trường. Khi sử dụng Scatter plot, ta dễ thấy xu hướng đồng thuận hiện hữu, kết quả khá khả quan (hình 4.2):

Nhưng có vấn đề chỉ thực sự xuất hiện khi ta cố gắng chứng minh giả thuyết của mình đúng bằng cách tính hệ số tương quan Pearson của các chỉ số chứng khoán và số bài viết. Ta có hệ số tương quan Pearson r :

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$



Hình 4.2: Scatter plot với VNINDEX

Khi đó ta tính được:

- Hệ số tương quan Pearson giữa count và max_volume_vnindex: 0.61
- Hệ số tương quan Pearson giữa count và max_volume_vn30: 0.45
- Hệ số tương quan Pearson giữa count và max_volume_hnxindex: 0.4
- Hệ số tương quan Pearson giữa count và tổng khối lượng giao dịch của cả 3 mã: 0.56

Mức độ tương quan cho cả 3 chỉ số ở mức trung bình ≈ 0.56 cho thấy rằng giả thuyết chưa hoàn toàn chính xác. Điểm sáng ở đây ở VNINDEX với hệ số Pearson lớn nhất là ≈ 0.61 . Tuy nhiên, nó vẫn khá ổn định xu hướng rằng **khi số lượng bài viết tăng thì khối lượng giao dịch cũng tăng tương ứng**.

4.2 Xu hướng của chỉ số VNINDEX và Hoạt động trên diễn đàn FireAnt

Như đã nhiều lần đề cập trước đó, VNINDEX là một chỉ số quan trọng vì nó đại diện cho toàn bộ thị trường chứng khoán Việt Nam, bao gồm 30 mã cổ phiếu lớn và có tính thanh khoản cao nhất. Do đó, việc ta tính toán được sự liên kết hay thậm chí là dự đoán giá trị chỉ số VNINDEX dựa trên các hoạt động bàn luận trên diễn đàn FireAnt sẽ mang lại nhiều lợi ích.

Trước hết, ta phân loại các ngày theo mức độ biến động giá của VNINDEX và tính toán số lượng bài viết trung bình (bao gồm bài viết tích cực và tiêu cực) cho từng nhóm. Cụ thể như sau:

- **Ngày Stable:** Giá thay đổi ít hơn 0.25%, tức là thay đổi nhỏ hoặc không có thay đổi đáng kể.
- **Ngày Slight Change:** Biến động giá trong khoảng từ 0.25% đến 2%.
- **Ngày Significant Increase:** Biến động giá lớn hơn 4% (giá tăng mạnh).
- **Ngày Significant Decrease:** Biến động giá nhỏ hơn -4% (giá giảm mạnh).

Sau khi phân loại các ngày, ta tiến hành tính toán số lượng bài viết trung bình cho từng loại ngày, bao gồm:

- Số lượng bài viết tổng cộng (*Average Number of Posts*).
- Số lượng bài viết tích cực (*Average Number of Positive Posts*).
- Số lượng bài viết tiêu cực (*Average Number of Negative Posts*).

```

# Define thresholds
stability_threshold = 0.25 # Stable days with minimal change
slight_change_threshold = 2 # Slight change days

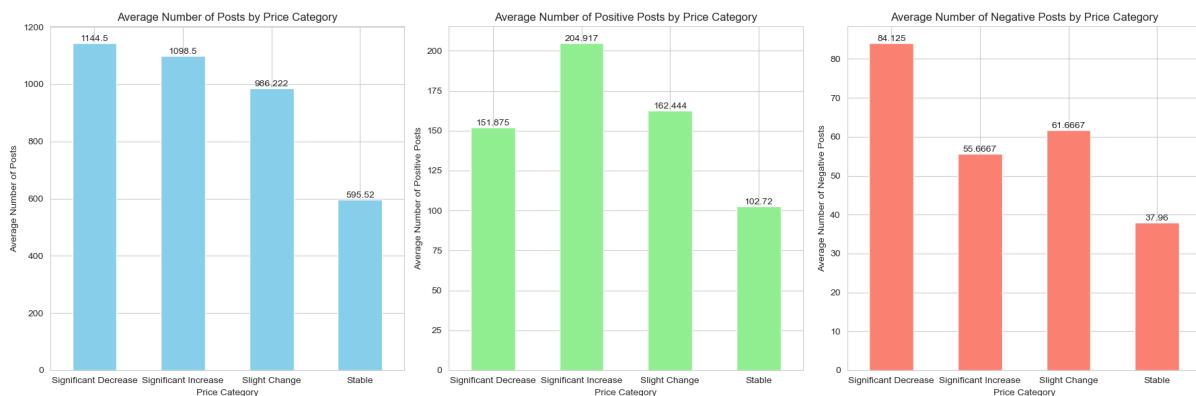
# Categorize days based on price change
VNINDEX_analyse['price_category'] = 'Stable'
VNINDEX_analyse.loc[(VNINDEX_analyse['price_change'] > stability_threshold) &
                     (VNINDEX_analyse['price_change'] <= slight_change_threshold), 'price_category'] = 'Slight Change'
VNINDEX_analyse.loc[VNINDEX_analyse['price_change'] > 4, 'price_category'] = 'Significant Increase'
VNINDEX_analyse.loc[VNINDEX_analyse['price_change'] < -4, 'price_category'] = 'Significant Decrease'

# Calculate average number of posts for each category
avg_posts = VNINDEX_analyse.groupby('price_category')['NumberofPosts'].mean()
avg_positive_posts = VNINDEX_analyse.groupby('price_category')['NumberofPositivePosts'].mean()
avg_negative_posts = VNINDEX_analyse.groupby('price_category')['NumberofNegativePosts'].mean()

# Print the results
print("\nAverage Number of Posts by Price Category:")
for category in avg_posts.index: # Loop only through existing categories
    print(f"{category} Days:")
    print(f" - Average Number of Posts: {avg_posts[category]:.2f}")
    print(f" - Average Number of Positive Posts: {avg_positive_posts[category]:.2f}")
    print(f" - Average Number of Negative Posts: {avg_negative_posts[category]:.2f}")
    print("")

✓ 0.0s

```



Hình 4.3: Biểu đồ số lượng bài viết theo sự thay đổi về giá

Kết quả cho thấy, vào những ngày VNINDEX giảm mạnh (*Significant Decrease Days*), số lượng bài viết trung bình đạt mức cao nhất (1144.50). Điều này phản ánh sự quan tâm đặc biệt từ cộng đồng đầu tư khi thị trường có biến động giảm đáng kể. Đáng chú ý, số lượng bài viết tích cực (151.88) và tiêu cực (84.12) đều tăng cao, cho thấy các ý kiến thảo luận về biến động giảm không chỉ tập trung vào mặt tiêu cực mà còn bao hàm cả những cảm xúc lạc quan.

Ngược lại, vào những ngày VNINDEX tăng mạnh (*Significant Increase Days*), số lượng bài viết tích cực đạt mức cao nhất (204.92), phản ánh sự lạc quan và kỳ vọng lớn từ cộng đồng đầu tư trong bối cảnh thị trường khởi sắc. Đồng thời, số lượng bài viết tiêu cực lại rất thấp (55.67), cho thấy tâm lý chung khá tích cực trong các phiên tăng mạnh.

Trong khi đó, vào những ngày thị trường ổn định (*Stable Days*), số lượng bài viết trung bình chỉ đạt 595.52, thấp nhất trong tất cả các loại ngày. Điều này cho thấy khi thị trường không có biến động lớn, mức độ quan tâm từ cộng đồng giảm đáng kể. Số lượng bài viết tích cực (102.72) và tiêu cực (37.96) cũng duy trì ở mức thấp, thể hiện sự trầm lắng trong thảo luận khi không có sự kiện nổi bật xảy ra.

Hệ số tương quan giữa VNINDEX và số bài đăng

Dầu tiên ta cần biết giá trị trung bình và độ lệch chuẩn của VNINDEX trong dữ liệu:

```

# Calculate the average and standard deviation of the VNINDEX price
average_vnindex_price = VNINDEX_analyse['price'].mean()
standard_price_deviation = VNINDEX_analyse['price'].std()

```

```
Average VNINDEX Price: 1270.77
Standard Deviation of Price: 14.54
```

Kết quả cho thấy giá trị trung bình của VNINDEX trong khoảng thời gian 2 tháng là 1270.77, với độ lệch chuẩn 14.54. Điều này cho thấy VNINDEX có sự biến động không quá lớn trong giai đoạn này, mức giá trung bình tương đối ổn định và độ lệch chuẩn cho thấy sự dao động giá vừa phải. Ta thực hiện tính toán hệ số tương quan giữa phần trăm thay đổi của số lượng bài đăng và giá trị cổ phiếu, sử dụng hàm `pct_change()` và `corr()` có sẵn trong pandas.

```
VNINDEX_analyse['price_pct_change'] = VNINDEX_analyse['price'].pct_change() * 100
# tương quan giữa %thay đổi giá và số lượng bài
VNINDEX_analyse['NumberOfPosts'].corr(VNINDEX_analyse['price_pct_change'])
✓ 0.0s
-0.007997333474793247
```

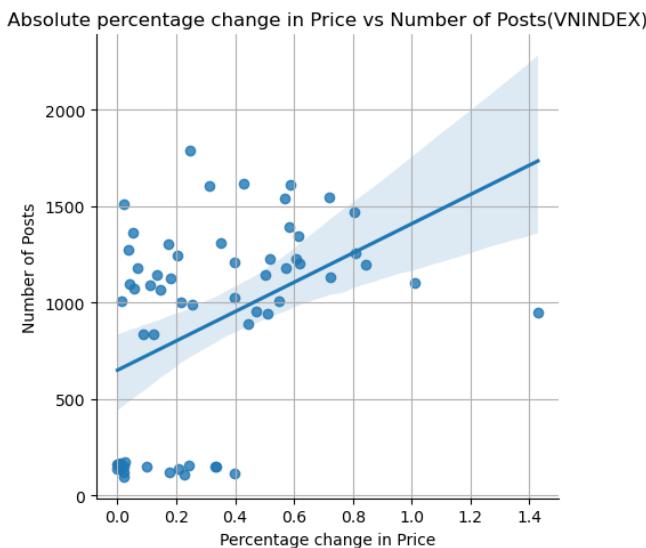
Kết quả cho thấy hệ số tương quan giữa sự thay đổi phần trăm giá trị và số lượng bài đăng là -0.008, cho thấy chẵng có mối quan hệ giữa hai yếu tố này.

Kết quả cho thấy tính toán của ta có thể đã có vấn đề, bởi theo như quan sát từ trước sự thay đổi trong giá trị cổ phiếu thường dẫn theo số lượng bài đăng sẽ tăng cao. Ta nhận thấy, số lượng bài viết đều có xu hướng tăng khi giá tăng hoặc giá giảm. Việc chỉ dùng giá trị thay đổi thuần để tính hệ số tương quan như các lần thực hiện trước đó là chưa chính xác, vì nó bỏ qua yếu tố rằng cả biến động tăng và giảm đều có thể kích thích sự thảo luận từ phía cộng đồng nhà đầu tư. Vậy nên ta phải tính theo **giá trị tuyệt đối phần trăm thay đổi giá trong ngày**, tức phải bổ sung giá trị tuyệt đối.

```
VNINDEX_analyse['price_pct_change'] = VNINDEX_analyse['price'].pct_change() * 100
# tương quan giữa (giá trị tuyệt đối %thay đổi giá) và số lượng bài
VNINDEX_analyse['NumberOfPosts'].corr(VNINDEX_analyse['price_pct_change'].abs())
✓ 0.0s
0.4317948478927149
```

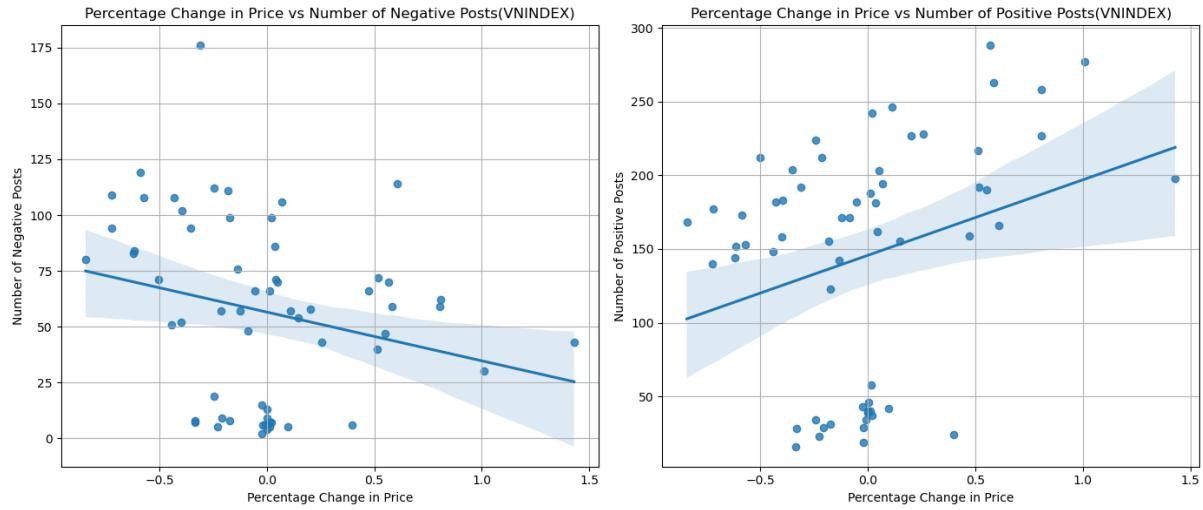
Đúng như những nhận định và dự đoán trước đó, số lượng bài viết giờ đã mối liên kết rõ ràng hơn với chỉ số VNINDEX nói riêng và các mã cổ phiếu khác nói chung. Xu hướng này cho thấy sự quan tâm của nhà đầu tư đối với thị trường chứng khoán được thể hiện rõ ràng thông qua hoạt động thảo luận trên các diễn đàn. Tuy nhiên, giá trị chỉ 0.432 cũng chỉ ra rằng mối liên kết này chưa phải mối liên kết mạnh mẽ.

Để thể hiện rõ hơn xu hướng, ta có thể vẽ biểu đồ Scatter plot:



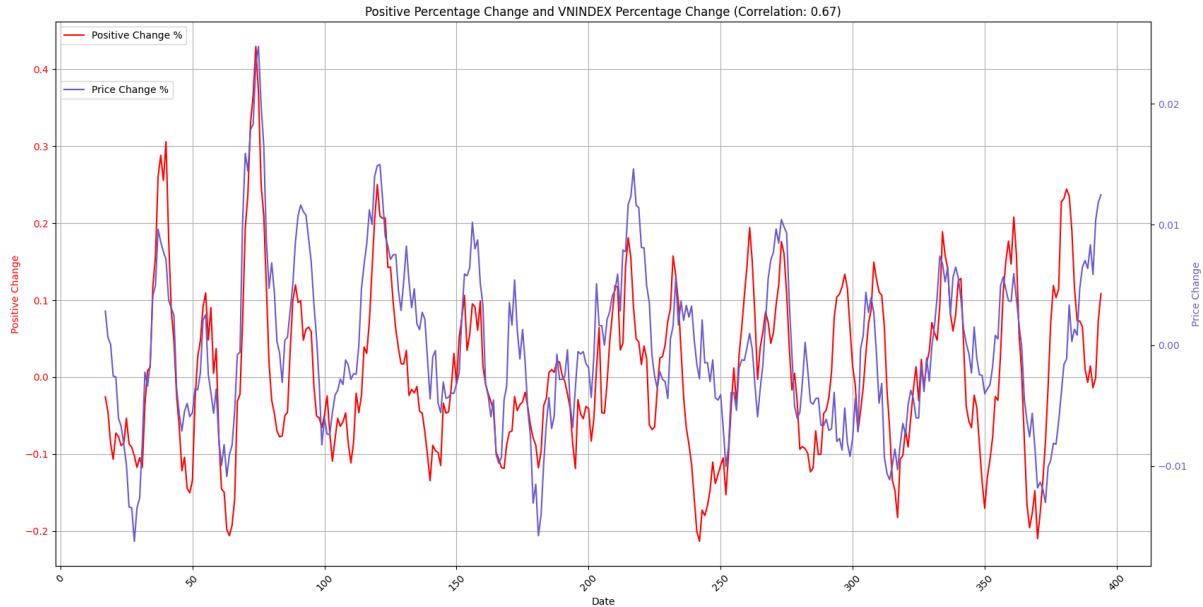
Hình 4.4: Scatter plot thể hiện Tương quan giữa Phần trăm thay đổi giá và Số lượng bài viết

Thực hiện tương tự với số lượng bài đăng tích cực và tiêu cực với giá trị cổ phiếu:



Hình 4.5: Tương quan giữa Phần trăm thay đổi giá và Số lượng bài viết Tích cực/Tiêu cực

Để ý rằng, trong ba biểu đồ trên luôn có một cụm chấm đứng rời rạc hẳn so với phần còn lại. Những vùng chấm đó là khi giá VNINDEX không đổi và số lượng bài đăng thấp. Ta dễ dàng thấy điều này hay xảy ra vào lúc cuối tuần. Như vậy, nếu ta chỉ xét khoảng thời gian giao dịch (tức từ 9h-15h hàng ngày, từ Thứ 2 đến Thứ 6), sự đồng pha thể hiện còn rõ ràng hơn với giá trị tương quan 0.67:



Hình 4.6: Diễn biến Phần trăm thay đổi giá và Số lượng bài viết Tích cực (Trong phiên)

Tóm lại, kết quả từ các biểu đồ cho thấy một xu hướng khá rõ ràng. Đường hồi quy của số lượng bài viết tiêu cực có độ dốc đi xuống, phản ánh mối quan hệ nghịch giữa số lượng bài viết tiêu cực và giá trị VNINDEX. Điều này cho thấy **khi VNINDEX tăng, số lượng bài viết tiêu cực có xu hướng giảm**.

Ngược lại, đường hồi quy của số lượng bài viết tích cực có độ dốc đi lên, chỉ ra mối quan hệ thuận với giá trị VNINDEX. **Khi VNINDEX tăng, số lượng bài viết tích cực cũng tăng theo**, phản ánh tâm lý lạc quan của nhà đầu tư trong những phiên thị trường khởi sắc.

Những xu hướng này **hoàn toàn phù hợp** với các dự đoán trước đó và được thể hiện một cách trực quan, rõ ràng qua biểu đồ, dù độ tương quan chưa phải rõ rệt.

Chương 5

Mô hình

5.1 Mô hình Dự đoán giá trong ngày sử dụng Định luật Bayes

Ở chương 4, ta xét thấy sự tương quan tương đối lớn giữa hoạt động của người dùng diễn đàn FireAnt và diễn biến của thị trường, ta hãy cùng phân tích và áp dụng một số mô hình để kiểm chứng lại sự tương quan. Ở phần này, ta sẽ tập trung vào việc dự đoán sự tăng hoặc giảm của giá cổ phiếu dựa trên những bài viết trên diễn đàn FireAnt. Dưới đây là diễn giải của một mô hình xác suất thống kê sử dụng Định luật Bayes.

5.1.1 Chuẩn bị dữ liệu tính toán



```
 1 # Create a directory to store the output files
 2 symbol_list = posts_df['symbol'].unique()
 3 symbol_list = symbol_list.tolist()
 4
 5 valid_symbols = []
 6 invalid_symbols = []
 7 # Remove None from the list
 8 symbol_list = [symbol for symbol in symbol_list if symbol is not None]
 9 symbol_price_data_dict = {}
10 symbol_price_data_dict = {}
11 for symbol in symbol_list:
12     clean_symbol = re.sub(r'[^a-zA-Z0-9]', '', symbol)
13     # check if the file exists
14     if not os.path.exists(f"price/{clean_symbol}.csv"):
15         symbol_price_data_dict[symbol] = None
16         invalid_symbols.append(symbol)
17         continue
18     data = pd.read_csv(f"price/{clean_symbol}.csv")
19     symbol_price_data_dict[symbol] = data
20     valid_symbols.append(symbol)
21
22 symbol_groups = posts_df.groupby('symbol')
23
24 symbols_data_dict = {symbol : data for symbol, data in symbol_groups}
```

Ta lấy danh sách các mã chứng khoán từ file CSV `cleaned_posts.csv` và đọc dữ liệu lịch sử giá của các mã đó vào `symbol_price_data_dict`. Tiếp theo, ta chuẩn bị một dataframe gồm các chỉ số thống kê bài viết được nhóm theo ngày, bao gồm Số lượng bài viết; Số bài viết Tích cực; Số bài viết Tiêu cực; Số từ trung bình của bài viết; Tổng số lượt Thích; Tổng số lượt Bình luận.

Đồng thời kết hợp cùng dữ liệu giá cổ phiếu theo ngày, bao gồm Giá mở cửa; Giá cao nhất; Giá đóng cửa; Sự thay đổi giá trong ngày. Tất nhiên, ta chỉ lấy những ngày có diễn ra giao dịch.

```

1 # Aggregate post data for each date related to the symbol
2 posts_analyze_symbol = symbols_data_dict[symbol_name].groupby('date').agg(
3     number_of_posts=('postID', 'count'),
4     number_of_positive_posts=('sentiment', lambda x: (x == 'positive').sum()),
5     number_of_negative_posts=('sentiment', lambda x: (x == 'negative').sum()),
6     average_posts_len=('word_count', 'mean'),
7     total_likes=('totalLikes', 'sum'),
8     total_replies=('totalReplies', 'sum'),
9 ).reset_index()

```

Kết quả là một dataframe mới với các cột:

`date, number_of_posts, number_of_positive_posts, number_of_negative_posts
average_posts_len, total_likes, total_replies, open, high, close, daily_change.`

# check if data in price folders are loaded successfully symbols_posts_data['VNINDEX'].head(20)												
✓	0.0s	date	number_of_posts	number_of_positive_posts	number_of_negative_posts	average_posts_len	total_likes	total_replies	open	high	close	daily_change
0	2024-09-04		0.0	0.0	0.0	0.00	0	0	1273.86	1277.27	1275.80	0.001523
1	2024-09-05		0.0	0.0	0.0	0.00	0	0	1276.46	1282.21	1268.21	-0.006463
2	2024-09-06		707.0	80.0	48.0	38.41	978	1053	1268.22	1274.44	1273.96	0.004526
3	2024-09-09		891.0	148.0	51.0	54.94	1321	1528	1273.96	1267.73	1255.23	-0.004890
4	2024-09-10		1203.0	144.0	83.0	41.94	1734	1923	1270.31	1271.83	1255.23	-0.011871

5.1.2 Áp dụng Định luật Bayes

Trước hết ta sẽ xây dựng hàm `bayesProbPosts` để thực hiện tính toán xác suất giá trị cổ phiếu thay đổi trong ngày. Dữ liệu trong hàm bao gồm ba cột: `date` (ngày giao dịch), `daily_change` (phần trăm thay đổi giá cổ phiếu hàng ngày), và `number_of_posts` (số lượng bài viết trong ngày). Các cột này được kết hợp vào một dataframe duy nhất để dễ dàng xử lý và phân tích.

Tiếp theo ta xác định ngưỡng để phân định một ngày bất kỳ có số lượng bài viết là nhiều, ít, hay trung bình. Ta sẽ tính trung bình số lượng bài viết, sau đó phân định:

- Nếu số lượng bài viết ngày **lớn hơn 15% mức trung bình**: Ngày này có lượng bài viết **nhiều**;
- Nếu số lượng bài viết trong ngày **nhỏ hơn 15% mức trung bình**: Ngày này có lượng bài viết **ít**;
- Trường hợp còn lại: Ngày này có lượng bài viết **trung bình**.

Tương tự, với phần trăm thay đổi giá (`price_column`):

- Nếu phần trăm thay đổi giá **lớn hơn 0.2%**: Ngày này có giá **tăng**;
- Nếu phần trăm thay đổi giá **nhỏ hơn -0.2%**: Ngày này có giá **giảm**;
- Trường hợp còn lại: Ngày này có giá **ít biến động**.

Như vậy, một ngày giao dịch có thể có **9 trường hợp kết quả xảy ra**. Với mỗi trường hợp, hàm `bayesProbPosts` sẽ tính ra giá trị xác suất cụ thể. Bằng cách sử dụng Định luật Bayes, giả sử với trường hợp (Giá tăng, Bài nhiều), ta có những xác suất sau:

$$P(\text{Giá tăng} | \text{Bài nhiều}) = \frac{P(\text{Bài nhiều} | \text{Giá tăng}) \times P(\text{Giá tăng})}{P(\text{Bài nhiều})}$$

$$P(\text{Bài nhiều} | \text{Giá tăng}) = \frac{P(\text{Số ngày Bài nhiều và Giá tăng})}{P(\text{Giá tăng})}$$

$$P(\text{Bài nhiều}) = \frac{\sum(\text{dataframe}[\text{'Số bài trong ngày'}] \geq \text{Ngưỡng xác định})}{\text{Số ngày giao dịch}}$$

$$P(\text{Giá tăng}) = \frac{\sum(\text{dataframe}[\text{'Thay đổi giá trong ngày'}] \geq \text{Ngưỡng xác định})}{\text{Số ngày giao dịch}}$$

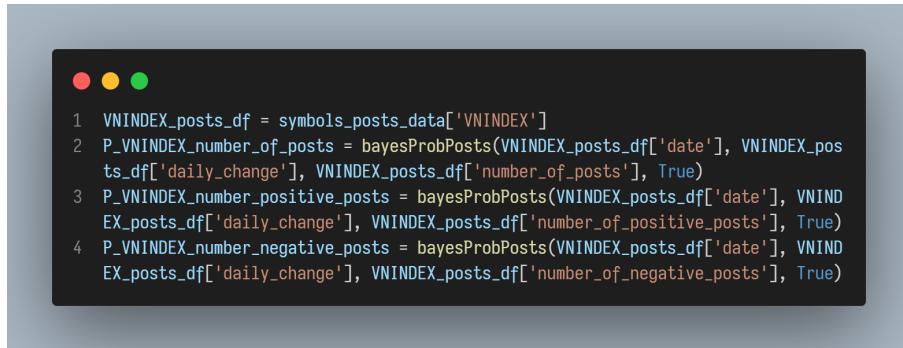
```

PRICE_SIGNIFICANT_THRESHOLD = 0.002 # If the price change is greater than 0.2%, it is considered significant
POSTS_SIGNIFICANT_THRESHOLD = 0.15 # If the number of posts change is 15% more than the mean, it is considered significant

def bayesProbPosts(date_column, price_column, posts_column, verbose=False):
    # Retrieve the data frame for the given stock symbol
    symbol_data_frame = pd.DataFrame({
        'date' : date_column,
        'daily_change' : price_column,
        'number_of_posts' : posts_column
    })
    # Prepare the conditions for price movements (up, down, stable)
    price_up = symbol_data_frame['daily_change'] > PRICE_SIGNIFICANT_THRESHOLD # Price up
    price_down = symbol_data_frame['daily_change'] < -PRICE_SIGNIFICANT_THRESHOLD # Price down
    price_stable = ~price_up & ~price_down # Price stable
    # Define the thresholds for categorizing the number of posts (low, high, normal)
    low_threshold = mean_posts*(1-POSTS_SIGNIFICANT_THRESHOLD) # Low posts:
    high_threshold = mean_posts*(1+POSTS_SIGNIFICANT_THRESHOLD) # High posts:

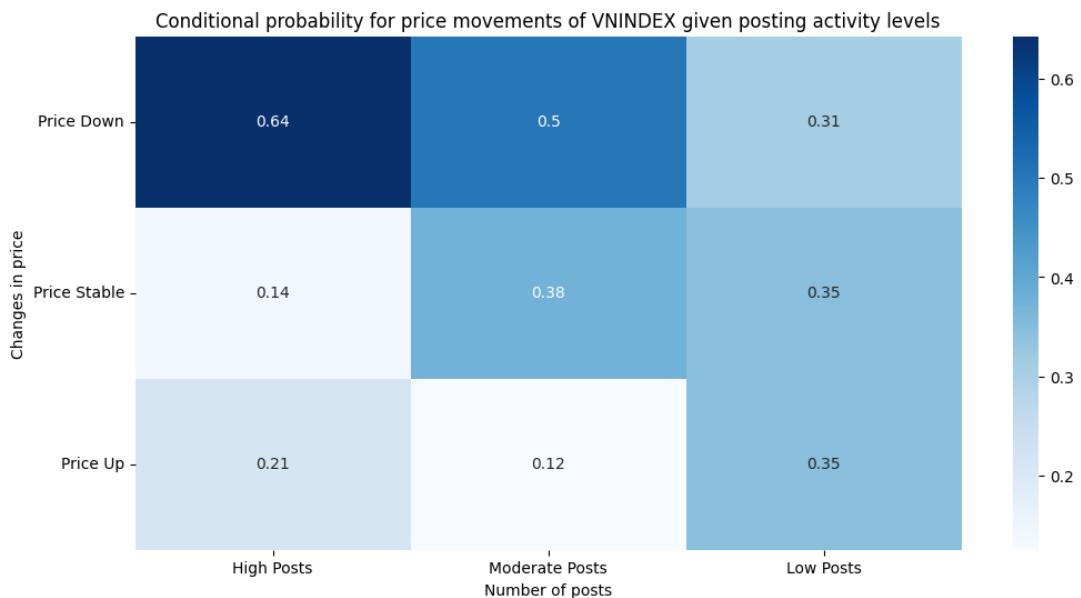
```

Ta tiến hành áp hàm này vào 3 yếu tố: Số bài viết trong ngày; Số bài viết Tích cực trong ngày; Số bài viết Tiêu cực trong ngày. Sau đó ta sẽ trực quan hóa kết quả bằng các biểu đồ Heatmap.



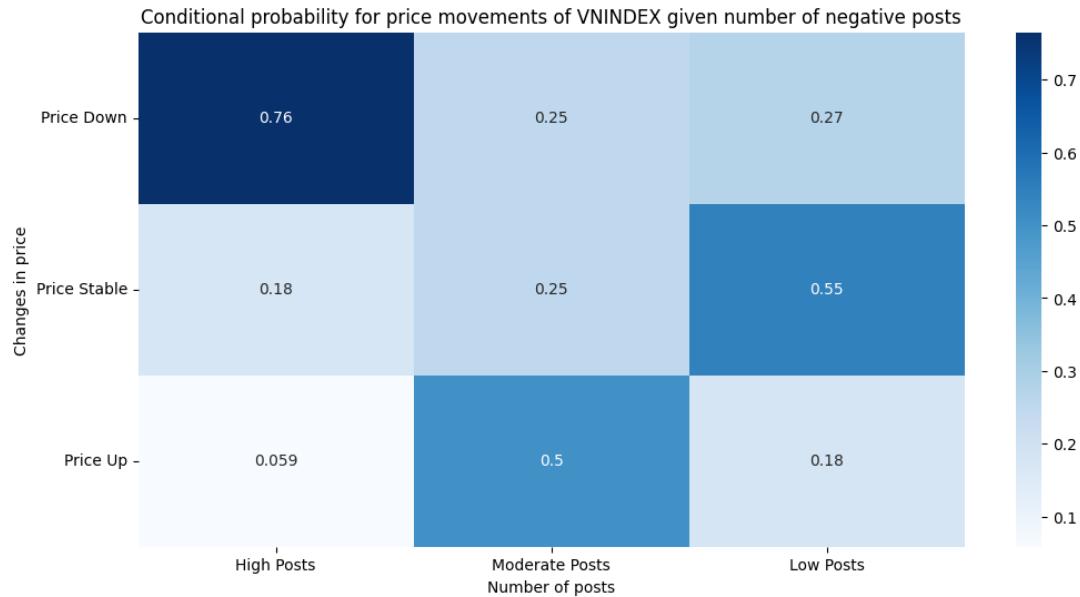
5.1.3 Kiểm chứng Phương pháp

Nhận xét Phương pháp trên VNINDEX



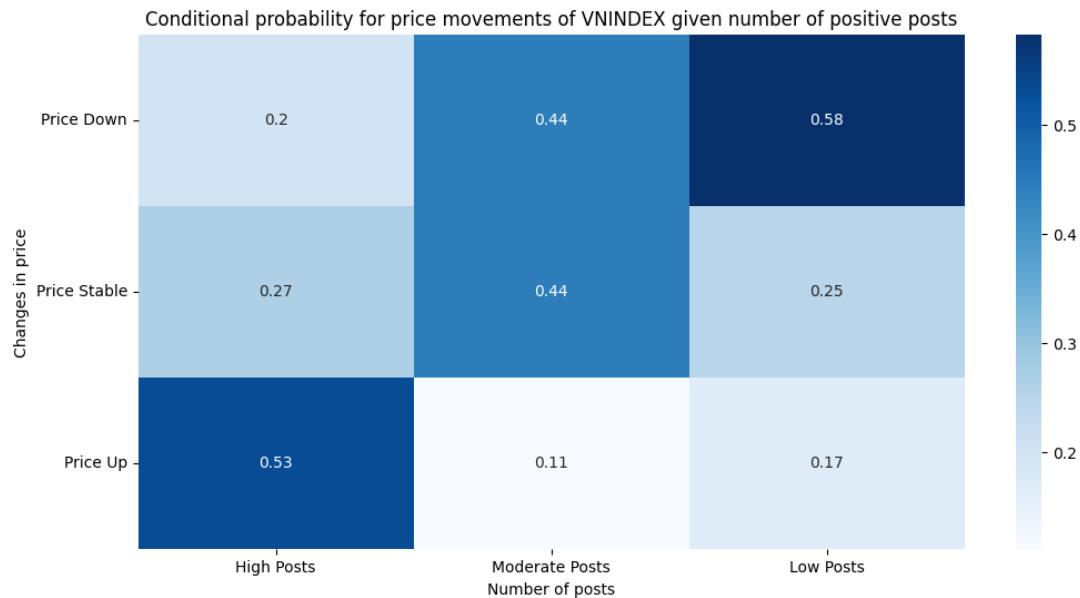
Hình 5.1: Heatmap Xác suất sự thay đổi của VNINDEX dựa trên Số bài đăng hàng ngày

Ta sẽ đánh giá phương pháp này dựa trên kết quả của nó khi áp với giá trị của chỉ số VNINDEX. Với biểu đồ trên, ta có thể nhận xét khi có nhiều bài đăng trong ngày, thì VNINDEX sẽ có 64% khả năng giảm giá. Khi lượng bài viết ở mức trung bình, xu hướng giảm giá vẫn tương đối rõ rệt, khi chỉ có 12% xác suất giá sẽ tăng. Cuối cùng, khi có ít bài viết trong ngày, kết quả sẽ tương đối khó đoán định, có thể do đó là những ngày thị trường chán nản, thanh khoản thấp.



Hình 5.2: Heatmap Xác suất sự thay đổi của VNINDEX dựa trên Số bài đăng Tiêu cực hàng ngày

Khi xét số lượng bài tiêu cực, ta dễ dàng nhận thấy khi có nhiều bài đăng tiêu cực trong ngày, thì VNINDEX sẽ có khả năng giảm giá cao. Tuy vậy, khi lượng bài viết tiêu cực ở mức thấp, xác suất tăng giá lại rõ rệt hơn (55%). Cuối cùng, khi có số lượng vừa phải các bài viết tiêu cực trong ngày, kết quả sẽ nghiêng về phía đứng giá. Tụt chung, xác suất giảm giá sẽ giảm khi số lượng bài viết Tiêu cực giảm. Điều đó cho thấy, có sự tương quan giữa số lượng bài viết Tiêu cực và diễn biến của thị trường.



Hình 5.3: Heatmap Xác suất sự thay đổi của VNINDEX dựa trên Số bài đăng Tích cực hàng ngày

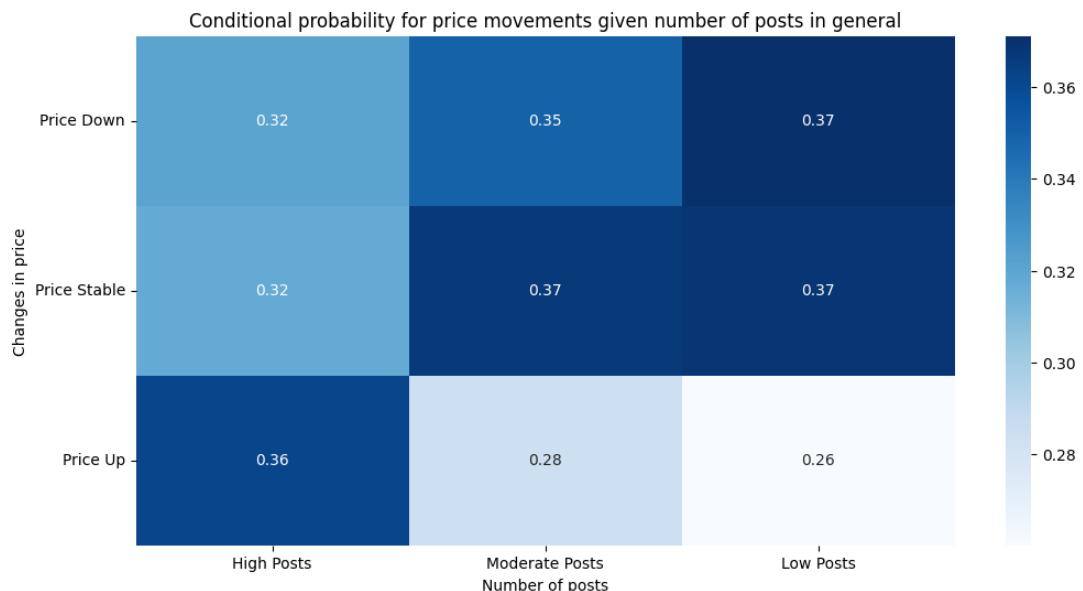
Với trường hợp tích cực, xu hướng được biểu diễn tương đối rõ ràng. Xác suất chỉ ra rằng VNINDEX sẽ có khả năng tăng giá cao khi có nhiều bài Tích cực, và càng trở nên khó đoán định khi số lượng bài

Tích cực giảm dần. Khi số lượng bài Tích cực càng ít, thì xu hướng giảm giá sẽ lộ rõ ràng. Tựu chung, xác suất giảm giá sẽ tăng khi số lượng bài viết Tích cực giảm.

Nhìn chung, từ cả 3 biểu đồ, ta có thể đưa ra nhận xét rằng, các xu hướng trên đều đúng với giả thuyết, rằng tâm lý của các nhà đầu tư, thể hiện qua số lượng bài đăng trong ngày, có sự tương quan tương đối với diễn biến của thị trường thực tế trong ngày đó. Khi sử dụng số lượng bài viết Tích cực/Tiêu cực để đánh giá, xu hướng sẽ hiện ra rõ rệt hơn so với việc sử dụng số lượng bài viết chung. Đồng thời, ta nhận thấy xu hướng giảm giá sẽ biểu lộ rõ rệt hơn so với xu hướng tăng giá, một phần lý do có thể đến từ việc dữ liệu đầu vào biến động nghiêng về phía giảm nhiều hơn.

Nhận xét Phương pháp trên Toàn bộ mã cổ phiếu

Để thực hiện được điều này, trước hết ta cho chạy hàm `bayesProbPosts` cho từng mã chứng khoán. Sau đó, với từng trường hợp kết quả, ta tính trung bình giá trị của trường hợp kết quả đó trong toàn bộ mã chứng khoán.



Hình 5.4: Heatmap Xác suất của Toàn bộ cổ phiếu dựa trên Số bài đăng hàng ngày

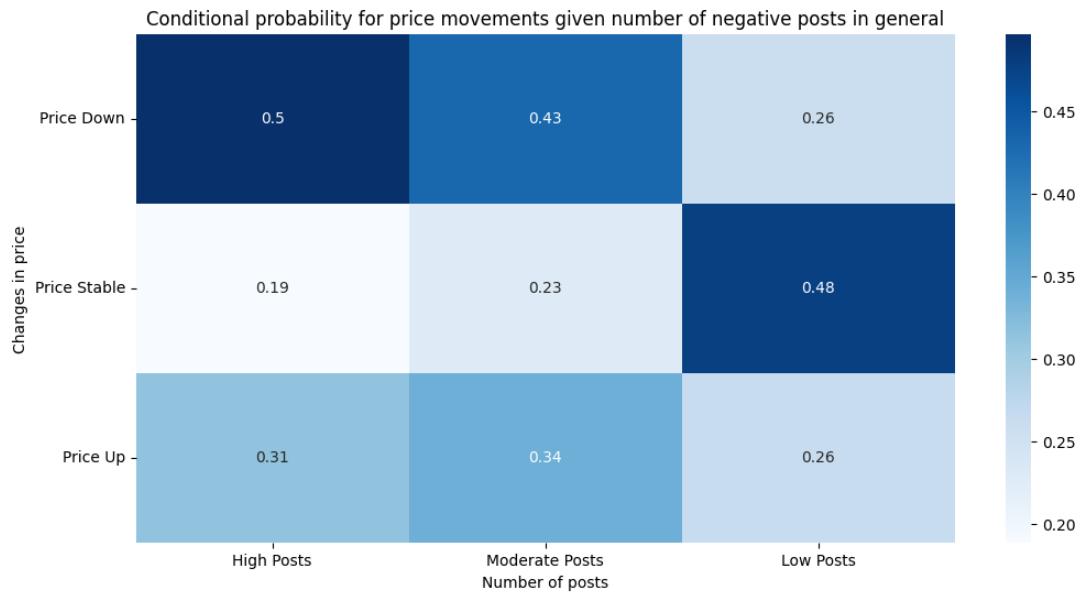
Điểm đặc biệt của biểu đồ (5.4) là xác suất rất đều, khi toàn bộ các trường hợp chỉ khác nhau chưa tới 9%. Điều này khẳng định luận điểm rằng khi sử dụng số lượng bài viết chung để tính xác suất, xu hướng thị trường sẽ không lộ rõ.

Xu hướng đã lộ rõ hơn ở biểu đồ (5.5), và xác suất giảm giá vẫn đi xuống khi số lượng bài viết Tiêu cực giảm. Tuy nhiên, điều này không thể hiện rõ bằng biểu đồ (5.2), khi ta xét trên VNINDEX.

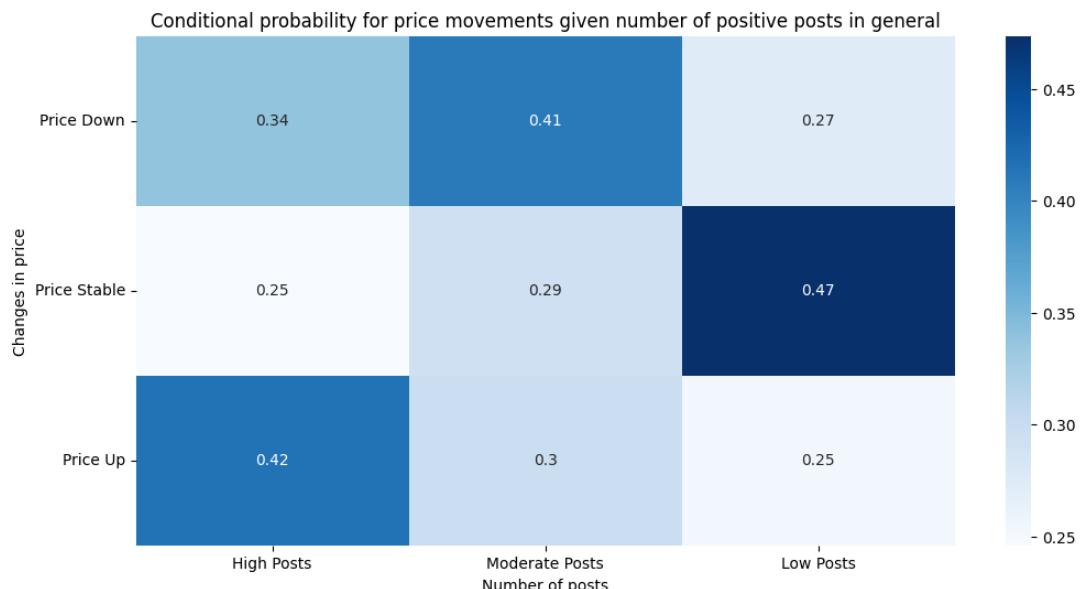
Cuối cùng là biểu đồ (5.6). Nhìn chung những luận điểm từ trước của ta vẫn chính xác.

Từ 3 biểu đồ heatmap trên, có thể thấy việc gộp chung dữ liệu từ tất cả các mã không phải là một phương án tối ưu, khi mà xác suất để giá trị cổ phiếu tăng, giảm hoặc duy trì ổn định đều không quá chắc chắn, dù nó vẫn thỏa mãn những luận điểm khi ta xét trên VNINDEX. Sẽ tốt hơn nếu ta thực hiện chọn lọc các cổ phiếu có xác suất tăng rõ ràng hơn để tính toán.

Nhận xét chung về phương pháp dự đoán giá cổ phiếu sử dụng định luật Bayes: Mặc dù cho nhiều kết quả khả quan, tuy nhiên phương pháp này gần như không mang lại giá trị thực tiễn, do xác suất tăng hay giảm cụ thể là chưa đáng kể. Ngoài ra, ta cũng đang dùng số lượng bài đăng để dự đoán sự tăng/giảm của cổ phiếu trong ngày, nhưng giá trị cổ phiếu có biến động (tăng/giảm) thì số lượng bài đăng mới tăng. **Số lượng bài đăng trên diễn đàn không nguyên nhân chính khiến giá biến động.** Dẫu vậy, ta có thể dùng số lượng bài đăng như một hình thức tham khảo hoặc loại cảnh báo.



Hình 5.5: Heatmap Xác suất của Toàn bộ cổ phiếu dựa trên Số bài đăng Tiêu cực hàng ngày



Hình 5.6: Heatmap Xác suất của Toàn bộ cổ phiếu dựa trên Số bài đăng Tích cực hàng ngày

Hướng cải thiện: Có thể bổ sung thêm dữ liệu như nội dung các bài viết, thời gian và một vài chỉ số tài chính để làm giàu đặc trưng đầu vào. Hơn nữa, ta có thể kết hợp với các mô hình học máy khác để có được độ chính xác tốt hơn. Đồng thời, cần thêm nhiều dữ liệu để giảm nhiễu và kiểm tra lại các giả định cũng như hiệu chỉnh thông số để tăng độ chính xác.

5.2 Mô hình Dự đoán giá Phiên kế tiếp sử dụng Xích Markov

Như đã trình bày về hạn chế về phương pháp dự đoán giá trị cổ phiếu áp dụng Định luật Bayes, ta sẽ thử nghiệm với cách tiếp cận sử dụng Xích Markov. Xích Markov là một quá trình ngẫu nhiên trong đó trạng thái của hệ thống tại thời điểm tiếp theo chỉ phụ thuộc vào trạng thái hiện tại, không phụ thuộc vào các trạng thái trước đó. Quá trình này được đặc trưng bởi tính chất “không nhớ” (memoryless property).

5.2.1 Chuẩn bị dữ liệu tính toán

Gần tương tự như phần Chuẩn bị dữ liệu tính toán sử dụng Định luật Bayes (phần 5.1.1), ta cũng sẽ có một dataframe mới, lần này với nhiều cột (features) hơn:

```
time, number_of_posts, number_positive_posts, number_negative_posts, number_neutral_posts
open, high, low, close, volume, average_post_length, average_price_mentioned
```

	time	number_of_posts	number_positive_posts	number_of_negative_posts	number_neutral_posts	average_price_mentioned	open	high	low	close
0	2024-09-04 09:00:00	0	0	0	0	0.0	1273.86	1273.86	1268.53	1271.34 76
1										

Tương tự như phần trước, để tránh lặp lại mã nguồn cho các tính toán và thống kê tương tự nhau, ta xây dựng các hàm chức năng:

- **Hàm categorize_posts()**: Hàm này phân loại số lượng bài viết vào ba nhóm: cao, trung bình, hoặc thấp, dựa trên độ lệch so với số lượng bài viết trung bình.
- **Hàm define_state(change, threshold=0.01)**: Hàm này xác định trạng thái của giá cổ phiếu là Tăng giá, Giảm giá, hoặc Ổn định giá, dựa trên ngưỡng biến động 1%.
- **Hàm df_to_transitions_matrix(df)**: Tạo ba ma trận chuyển trạng thái: ma trận chuyển trạng thái tổng thể, ma trận chuyển trạng thái bài viết tích cực và ma trận chuyển trạng thái bài viết tiêu cực.
- **Hàm heatmap_illustration(matrix, title, xlabel, ylabel)**: Hàm này thực hiện trực quan hóa các ma trận chuyển trạng thái dưới dạng biểu đồ nhiệt, với các chú thích về xác suất.

5.2.2 Áp dụng Xích Markov tính xác suất chuyển đổi trạng thái

Công thức xác suất chuyển trạng thái trong xích Markov được mô tả như sau:

$$P(X_{t+1} = j \mid X_t = i) = P_{ij}$$

Trong đó:

- P_{11} là xác suất chuyển từ Tăng giá sang Tăng giá,
- P_{12} là xác suất chuyển từ Tăng giá sang Giảm giá,
- P_{13} là xác suất chuyển từ Tăng giá sang Ổn định giá, và tương tự cho các phần tử còn lại.

Trong trường hợp phân tích cổ phiếu, ta có thể xem xét ba trạng thái chính của giá cổ phiếu: **Tăng giá**, **Giảm giá**, và **Ổn định giá**. Các trạng thái này có thể được xác định dựa trên mức thay đổi giá cổ phiếu trong một khoảng thời gian nhất định, ở đây là 1 ngày.

Ta có thể xây dựng ma trận chuyển trạng thái P của xích Markov, trong đó mỗi phần tử P_{ij} là xác suất chuyển từ trạng thái i sang trạng thái j . Ma trận này có thể được xác định từ dữ liệu lịch sử giá cổ phiếu, ví dụ, thông qua việc tính toán tần suất xuất hiện của sự chuyển đổi giữa các trạng thái:

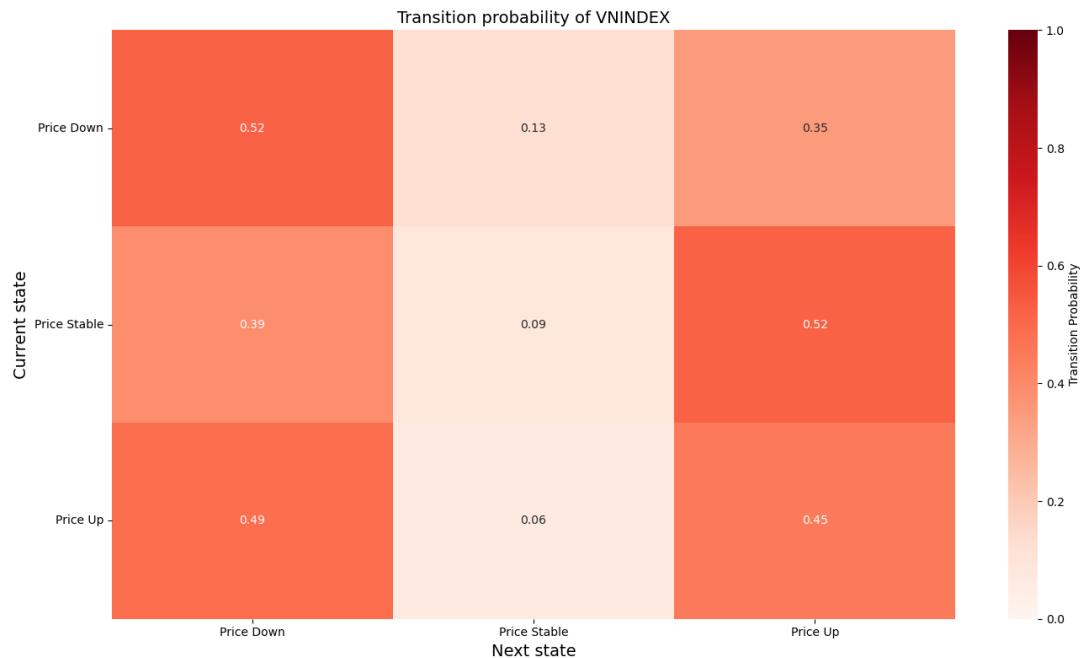
$$P = \begin{pmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{pmatrix}$$

Trong đó:

- P_{11} là xác suất chuyển từ Tăng giá sang Tăng giá,
- P_{12} là xác suất chuyển từ Tăng giá sang Giảm giá,
- P_{13} là xác suất chuyển từ Tăng giá sang Ổn định giá, và tương tự cho các phần tử còn lại.

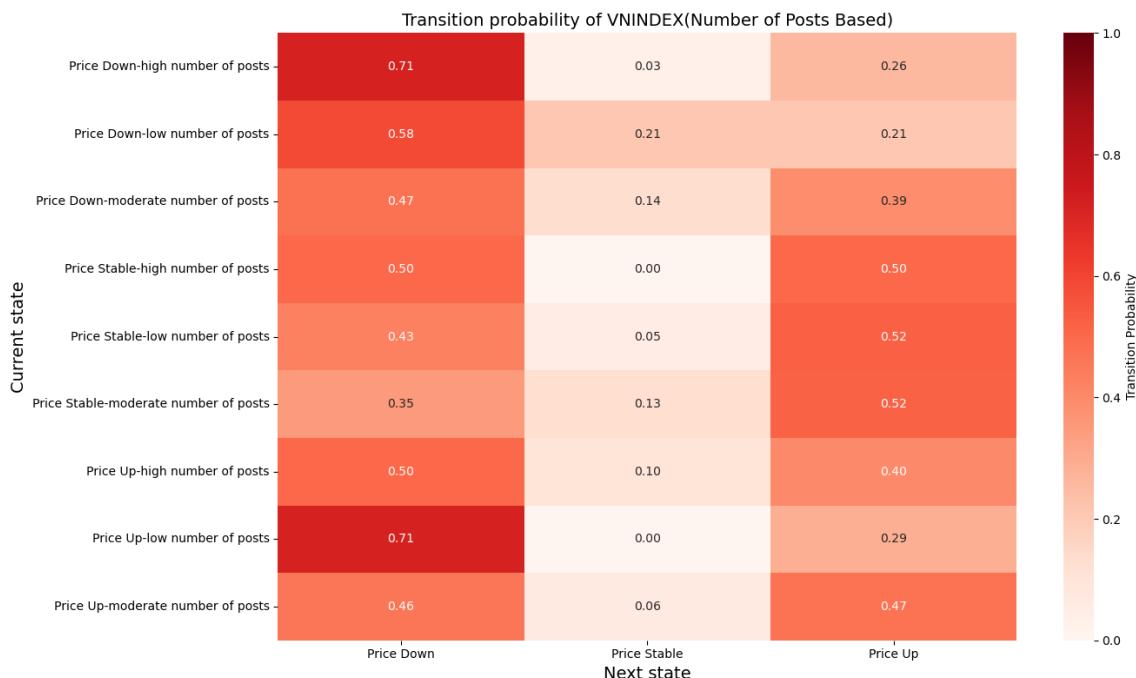
5.2.3 Kiểm chứng Phương pháp

Ta sẽ lập ma trận chuyển đổi của xích Markov với giá trị của VNINDEX, cho 4 trường hợp sau.



Hình 5.7: Heatmap Xác suất chuyển đổi trạng thái của giá trị VNINDEX

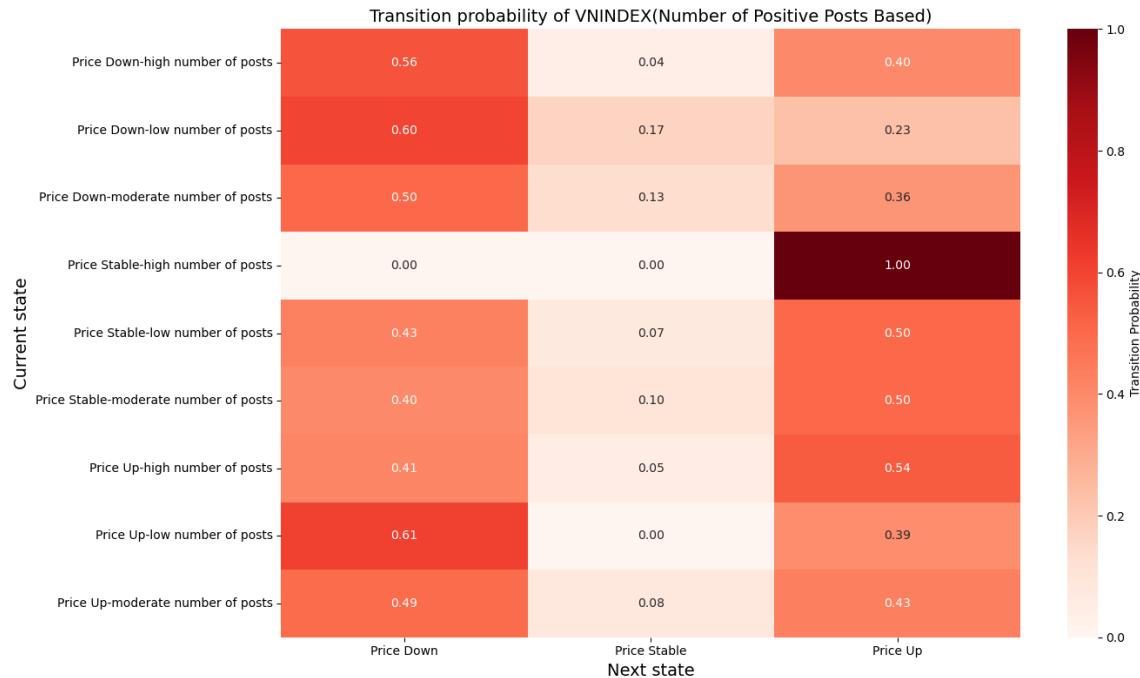
Dễ thấy xác suất để giá trị VNINDEX chuyển từ trạng thái hiện tại sang trạng thái tăng hoặc giảm, nhìn chung đều không chắc chắn, xoay quanh 50%. Đơn giản là vì ta chưa xét các yếu tố nào khác mà chỉ dựa hoàn toàn lịch sử thay đổi giá trị trước đó.



Hình 5.8: Heatmap Xác suất chuyển đổi giá trị VNINDEX (Với yếu tố Số lượng bài đăng)

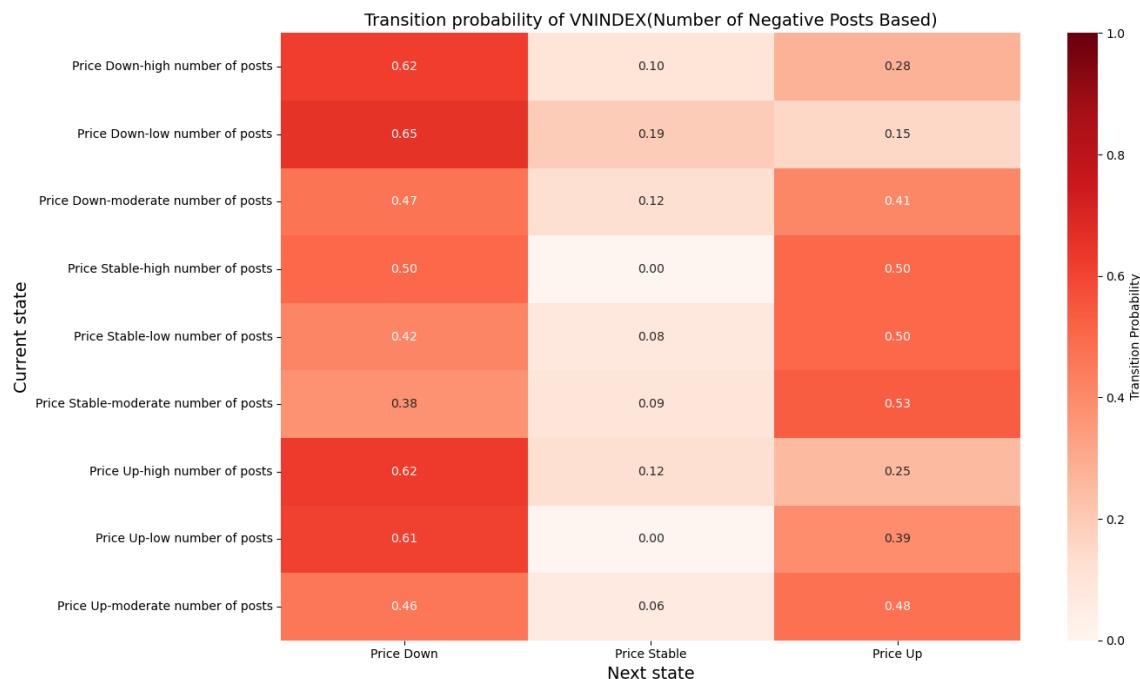
Khắc phục hạn chế từ kết quả ở biểu đồ (5.7), sau khi thêm vào số lượng các bài đăng trong từng khoảng thời gian vào một trong các yếu tố thống kê, ta thu được kết quả khả quan hơn. Cụ thể, nếu ở

đợt giao dịch hiện tại giá trị cổ phiếu giảm và số lượng bài viết cao, thì ở lần giao dịch tiếp theo, 71% giá trị VNINDEX sẽ giảm. Hay giá trị cổ phiếu có khả năng giảm khoảng 71% nếu ở đợt giao dịch hiện tại giá trị cổ phiếu tăng, số lượng bài viết thấp.



Hình 5.9: Heatmap Xác suất chuyển đổi giá trị VNINDEX (Với yếu tố Số lượng bài đăng Tích cực)

Tương tự như bảng heatmap (5.8), khi số lượng bài đăng tích cực tăng cao, mà giá trị cổ phiếu lại duy trì ở mức ổn định, giá trị cổ phiếu khả năng cao sẽ tăng ở lượt giao dịch tiếp theo. Tuy vậy, trong hầu hết các trường hợp, giá trị cổ phiếu có khả năng giảm nhiều hơn tăng.



Hình 5.10: Heatmap Xác suất chuyển đổi giá trị VNINDEX (Với yếu tố Số lượng bài đăng Tiêu cực)

Khác với khi sử dụng số lượng bài viết nói chung và số lượng bài viết Tích cực, khi dựa vào số bài viết Tiêu cực, ta thấy xác giá trị cổ phiếu giảm chỉ cao nhất là 65%. Điều này trái với dự đoán rằng, khi số lượng bài viết tiêu cực cao, xác suất lượt giao dịch tiếp theo giá giảm cũng sẽ cao. Tương tự xác suất

chuyển đổi trạng thái sang giá trị cổ phiếu tăng cũng chỉ cao nhất là 53% và thấp nhất là 25%. Nhìn chung, số lượng bài viết tiêu cực không thể hiện quá nhiều về khả năng tăng hoặc giảm của mã cổ phiếu sau này.

Nhận xét chung về phương pháp dự đoán giá sử dụng xích Markov: Dẫu mang lại nhiều kết quả tích, Xích Markov vẫn tồn tại nhiều hạn chế. Cụ thể, một trong những nhược điểm lớn của phương pháp xích Markov trong dự đoán giá cổ phiếu là sự giả định về tính độc lập của các trạng thái, tức là mỗi trạng thái chỉ phụ thuộc vào trạng thái ngay trước đó mà không tính đến các yếu tố dài hạn hay các biến động ngoại lai. Điều này có thể dẫn đến việc bỏ qua các yếu tố quan trọng như tin tức, sự kiện vĩ mô, hay tâm lý thị trường, vốn có thể ảnh hưởng mạnh mẽ đến biến động giá cổ phiếu.

Thêm vào đó, phương pháp này không thể dự đoán chính xác trong các thị trường biến động mạnh hoặc khi có những sự kiện bất ngờ, vì mô hình chỉ dựa vào dữ liệu lịch sử mà không thể dự đoán các yếu tố ngoài dữ liệu này. Tựu chung lại, **mô hình không thể được sử dụng để dự đoán giá của cổ phiếu**.

Hướng cải thiện: Ta có thể kết hợp xích Markov với các mô hình học máy phức tạp hơn, như mạng nơ-ron, hoặc cây quyết định, để tận dụng thêm các yếu tố ngoài lịch sử giá và số lượng bài đăng, chẳng hạn như dữ liệu tin tức, các chỉ số kinh tế vĩ mô, hoặc cảm xúc thị trường. Việc sử dụng các dữ liệu không phải chuỗi thời gian này có thể giúp mô hình nhận diện các yếu tố tác động dài hạn và các sự kiện ngoại lệ mà xích Markov không thể mô phỏng được. Hơn nữa, việc bổ sung các tính năng như phân tích xu hướng hoặc các chỉ số kỹ thuật có thể cải thiện độ chính xác của dự đoán, đặc biệt trong các giai đoạn thị trường thay đổi mạnh.

5.3 Mô hình Học máy Phân tích Quan điểm dựa trên Nội dung Bài viết (Sentiment Analysis) sử dụng Random Forest

Với tập dữ liệu là những bài viết của người dùng, một khía cạnh thú vị mà ta có thể áp dụng máy học chính là phân tích quan điểm của bài viết. Ta sẽ sử dụng Học máy, cùng những kỹ thuật trong Xử lý ngôn ngữ tự nhiên nhằm tối ưu độ chính xác của kết quả.

Ta sẽ sử dụng thư viện Scikit-learn (`sklearn`) nhằm thực hiện những tác vụ liên quan đến Học máy. Thư viện `pandas` vẫn được sử dụng để làm việc với dữ liệu.

5.3.1 Chuẩn bị dữ liệu

Dữ liệu gốc của ta là dữ liệu của toàn bộ 270 nghìn bài viết, với hàng chục cột dữ liệu. Tuy nhiên ta chỉ quan tâm đến hai cột: nội dung bài viết (`originalContent`) và cột Quan điểm (`sentiment`).

Ta sẽ không lấy những bài có quan điểm Trung lập (`sentiment = 0`), do ta đang tìm cách để tạo mô hình phân loại một bài là Tích cực, hay Tiêu cực. Tiếp theo, ta sẽ bỏ những bài có gán link, do nó làm loãng dữ liệu và không đem lại giá trị khi phân định quan điểm. Sau quá trình xử lý nội dung trong bài, ta sẽ tiếp tục loại bỏ bài có độ dài ≤ 5 .

```
1 # only keep the originalContent sentiment link columns
2 posts_after_cleaned = posts_df[['originalContent', 'sentiment', 'link']]
3 # not interested in neutral sentiment
4 posts_after_cleaned = posts_after_cleaned[posts_after_cleaned['sentiment'] != 0]
5 # not interested in posts with link
6 posts_after_cleaned = posts_after_cleaned[posts_after_cleaned['link'].isnull()]
7 posts_after_cleaned = posts_after_cleaned[~posts_after_cleaned['originalContent'].str.contains('http|https|www|.com')]
8 posts_after_cleaned.drop(columns=['link'], inplace=True)
9 # process the text
10 posts_after_cleaned['originalContent'] = posts_after_cleaned['originalContent'].apply(preprocess_text_optimized)
11 # remove rows with content length <= 5
12 posts_after_cleaned = posts_after_cleaned[posts_after_cleaned['originalContent'].apply(lambda x: len(x) > 5)]
13 posts_after_cleaned.dropna(inplace=True)
```

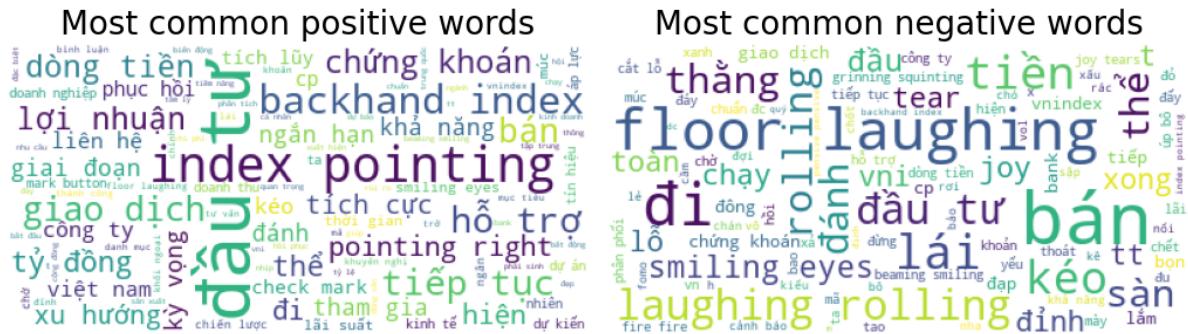
Về nội dung trong bài, trước hết ta sẽ biến mọi chữ về dạng in thường. Tiếp theo ta sẽ thực hiện bỏ những dấu câu, cũng như bỏ hết các chữ số. Với một số ký tự dạng Emojo, thay vì xóa chúng, ta giả

định rằng chúng cũng sẽ góp phần thể hiện quan điểm rõ ràng hơn. Vì vậy, ta sẽ biến nó về dạng chữ (miêu tả Emoji đó bằng lời).

Cuối cùng là công đoạn lọc stopword. Stopword là những từ không mang đậm ý nghĩa tiêu cực hay tích cực. Việc để lọt stopword sẽ ảnh hưởng xấu đến kết quả của mô hình, vậy nên ta cần lọc chúng. Ban đầu ta sẽ sử dụng một tập dữ liệu stopword chung chung [9], rồi sẽ dần dần thêm bớt hoặc bỏ đi các từ để phù hợp với tập dữ liệu của ta.

```
1 stop_words = (pd.read_csv('vietnamese-stopwords.csv'))['word'].tolist()
2 stop_words = set(stop_words) # Faster check, also remove duplicates
3
4 def preprocess_text_optimized(text):
5     text = text.lower()                                     # Make all text lowercase
6     text = text.translate(str.maketrans('', '', string.punctuation)) # Remove all symbols
7     text = demoji.replace_with_desc(text, sep = ' ')        # Replace all emojis with their descriptions
8     text = ''.join([i for i in text if not i.isdigit()])    # Remove numbers and other special characters
9     return ' '.join(word for word in text.split() if word not in stop_words) # Remove stop words
```

Kết quả, ta có **21159 bài Tích cực**, **11771 bài Tiêu cực**. Dữ liệu này đủ đa dạng để ta tiếp tục.



Hình 5.11: Một số từ hàm ý Tích cực và Tiêu cực xuất hiện nhiều nhất

Công đoạn tiếp theo ta sẽ thực hiện kỹ thuật TF-IDF. TF-IDF (term frequency-inverse document frequency) được dùng nhằm phản ánh tầm quan trọng của một từ đối với một văn bản trong một tập hợp hay một ngữ liệu văn bản. Ta sử dụng nó để biến văn bản thành vector chứa trọng số của các từ xuất hiện trong đoạn văn bản đó. Thư viện `scikit-learn` có chứa sẵn code cần thiết cho tác vụ này:

```
from sklearn.feature_extraction.text import TfidfVectorizer

def vectorize(data,tfidf_vect_fit: TfidfVectorizer):
    X_tfidf = tfidf_vect_fit.transform(data) # Biến đổi dữ liệu thành vector, theo TF-IDF
    words = tfidf_vect_fit.get_feature_names_out() # Các từ sẽ là các feature, lấy các từ được sử dụng trong TF-IDF
    X_tfidf_df = pd.DataFrame(X_tfidf.toarray()) # Chuyển dữ liệu về dạng DataFrame
    X_tfidf_df.columns = words # Đặt tên cột là các từ
    return(X_tfidf_df)

# Tạo TF-IDF Vectorizer
tfidf = TfidfVectorizer(analyzer = 'word', min_df=40, max_df=0.2) # Tạo TF-IDF Vectorizer
tfidf_vect_fitted=tfidf.fit(posts_after_cleaned_orig['originalContent']) # Thực hiện TF-IDF với dữ liệu

# Vector hóa toàn bộ dữ liệu
X_train = vectorize(X_train_pre['originalContent'],tfidf_vect_fitted)
y_train=y_train_pre.copy()

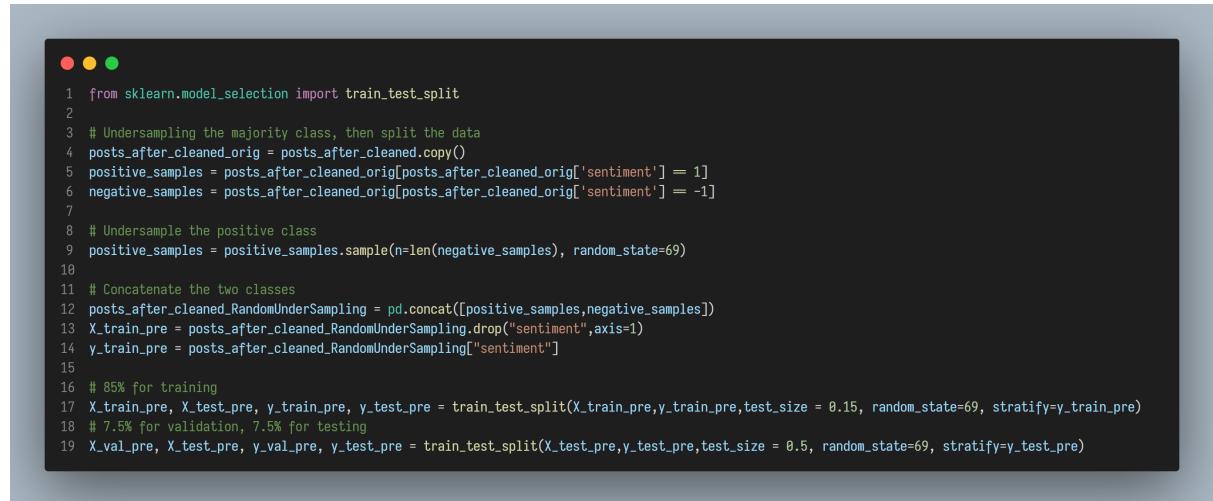
X_train.shape
```

Ta sẽ đặt các tham số cho kỹ thuật TF-IDF. Đầu tiên, ta thực hiện phân tách các từ bằng dấu cách (' '), bằng việc đặt `analyzer = 'word'`. Nói cách khác ta sẽ chỉ làm việc với những từ đơn. Điều này tương đối tệ khi áp dụng với tiếng Việt, do có nhiều từ cần phải là từ ghép mới mang được đầy đủ ý nghĩa. Ta có thể tối ưu bằng việc sử dụng một Tokenizer được làm đặc biệt cho tiếng Việt, tuy nhiên

với độ phức tạp của dự án, ta sẽ chỉ sử dụng phân tách bằng dấu cách.

Tiếp theo ta sẽ đặt ngưỡng tối thiểu để một từ được xuất hiện trong danh sách từ để xét TF-IDF là 40 lần (`min_df = 40`); tương tự, ngưỡng tối đa là xuất hiện 20% trong văn bản (`max_df = 0.2`). Đặt ngưỡng tối thiểu sẽ giúp tăng độ chính xác, vì những từ xuất hiện ít ảnh hưởng không đáng kể đến kết quả, hơn nữa còn giúp huấn luyện mô hình nhanh hơn, do phải làm việc với ít từ. Đặt ngưỡng tối đa cũng sẽ ngăn tình trạng để lọt stopword, giúp thuật toán chính xác hơn. Kết quả, ta có **1626 từ được dùng** để tạo vector trọng số cho nội dung bài viết.

Tiếp theo ta sẽ tiến hành phân tách tập dữ liệu thành 3 phần: tập Train (85%), tập Validation (7.5%), và tập Test (7.5%). Ta sử dụng hàm `train_test_split` trong `sklearn` để thực hiện tách tập dữ liệu:



```
1 from sklearn.model_selection import train_test_split
2
3 # Undersampling the majority class, then split the data
4 posts_after_cleaned_orig = posts_after_cleaned.copy()
5 positive_samples = posts_after_cleaned_orig[posts_after_cleaned_orig['sentiment'] == 1]
6 negative_samples = posts_after_cleaned_orig[posts_after_cleaned_orig['sentiment'] == -1]
7
8 # Undersample the positive class
9 positive_samples = positive_samples.sample(n=len(negative_samples), random_state=69)
10
11 # Concatenate the two classes
12 posts_after_cleaned_RandomUnderSampling = pd.concat([positive_samples,negative_samples])
13 X_train_pre = posts_after_cleaned_RandomUnderSampling.drop("sentiment",axis=1)
14 y_train_pre = posts_after_cleaned_RandomUnderSampling["sentiment"]
15
16 # 85% for training
17 X_train_pre, X_test_pre, y_train_pre, y_test_pre = train_test_split(X_train_pre,y_train_pre,test_size = 0.15, random_state=69, stratify=y_train_pre)
18 # 7.5% for validation, 7.5% for testing
19 X_val_pre, X_test_pre, y_val_pre, y_test_pre = train_test_split(X_test_pre,y_test_pre,test_size = 0.5, random_state=69, stratify=y_test_pre)
```

Vấn đề lớn nhất của ta trong tập dữ liệu chính là sự **mất cân bằng của dữ liệu đầu vào**. Có tới 64.2% bài viết là Tích cực trong khi chỉ có 35.8% bài viết Tiêu cực. Sự ảnh hưởng lớn này dễ tạo ra sự thiên vị cho mô hình, đồng thời kết quả kiểm tra mô hình sẽ bị ảnh hưởng mạnh, đặc biệt với mô hình kiểu như Random Forest. Có một vài cách xử lý cho tập dữ liệu này, thậm chí còn có sẵn thư viện có thể giúp chúng ta giải quyết vấn đề (sẽ được nói ở phần 5.3.4). Ta sẽ sử dụng **phương pháp Random Under-sampling**, nghĩa là ta sẽ chọn **ngẫu nhiên** các bài Tích cực, và chọn cho tới khi số bài được chọn bằng với số bài Tiêu cực có trong tập dữ liệu.

Sau khi Under-sampling, số bài Tích cực bằng với số bài Tiêu cực và bằng **11771 bài**. Cuối cùng, tập Train có **20010 bài**; tập Validation và Test đều có **1766 bài**. Tất cả dữ liệu đều có tỉ lệ bài Tích cực và Tiêu cực **bằng nhau**.

5.3.2 Huấn luyện mô hình

Sau khi có dữ liệu, ta sẽ thực hiện huấn luyện cho mô hình. Ta sẽ sử dụng **mô hình Random Forest** để thực hiện tác vụ này. Sở dĩ ta sử dụng Random Forest, thứ nhất vì tính đơn giản của nó, cũng như khả năng làm việc với ma trận nhiều chiều và tập dữ liệu lớn, phù hợp với những tác vụ liên quan đến ngôn ngữ. Random Forest cũng có thể xử lý các giá trị khuyết thiếu, outliers, điều dễ có thể xảy ra trong tập dữ liệu của ta. Để kiểm chứng độ hiệu quả của mô hình, ta sẽ sử dụng kỹ thuật Cross-validation, với tiêu chí chấm điểm là độ chính xác (accuracy):



```
# RF chưa tính chính
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_validate

scoring = ["accuracy"] # Tiêu chí đánh giá
rf = RandomForestClassifier(n_jobs=-1) # Khởi tạo model Random Forest
cv_res = cross_validate(rf, X_train, y_train, scoring=scoring) # Cross validation
print(f"General accuracy: {cv_res['test_accuracy'].mean()}")
```

General accuracy: 0.7187346326836581

Kết quả khá ổn khi mô hình có độ chính xác bình quân là 71.8% trên tập Train.

Tiếp theo ta sẽ tìm ra tham số tốt nhất để mô hình chạy tốt hơn. Ta sẽ sử dụng hàm `GridSearchCV` có trong `sklearn`. `GridSearchCV` sẽ thực hiện thử từng cặp tham số, sau đó đánh giá sử dụng Cross-validation để chọn ra cặp tham số hiệu quả nhất. Ta sẽ thực hiện thử với 3 loại tham số chính, là `n_estimators` (số lượng Cây quyết định), `max_features` (số lượng feature (đầu vào) sẽ lấy để chia cây), và `max_depth` (độ sâu tối đa của Cây).

```
# Tinh chỉnh tham số sử dụng GridSearchCV
from sklearn.model_selection import GridSearchCV
rf = RandomForestClassifier()

parameters = {
    'n_estimators': [50,100,200],           # Số cây
    'max_features': ['sqrt', 'log2'],      # Số features tối đa mỗi cây
    'max_depth': [30,None]                # Độ sâu tối đa mỗi cây
}

cv = GridSearchCV(rf, parameters, n_jobs=-1) # Thủ với model Random Forest
cv.fit(X_train,y_train.values.ravel())        # Fit với dữ liệu train
print_results(cv)                           # Hàm phụ trợ in kết quả

BEST PARAMS: {'max_depth': None, 'max_features': 'log2', 'n_estimators': 200}

0.706 (+/-0.011) for {'max_depth': 30, 'max_features': 'sqrt', 'n_estimators': 50}
0.703 (+/-0.015) for {'max_depth': 30, 'max_features': 'sqrt', 'n_estimators': 100}
0.702 (+/-0.017) for {'max_depth': 30, 'max_features': 'sqrt', 'n_estimators': 200}
0.678 (+/-0.01) for {'max_depth': 30, 'max_features': 'log2', 'n_estimators': 50}
0.684 (+/-0.01) for {'max_depth': 30, 'max_features': 'log2', 'n_estimators': 100}
0.686 (+/-0.012) for {'max_depth': 30, 'max_features': 'log2', 'n_estimators': 200}
0.732 (+/-0.013) for {'max_depth': None, 'max_features': 'sqrt', 'n_estimators': 50}
0.732 (+/-0.013) for {'max_depth': None, 'max_features': 'sqrt', 'n_estimators': 100}
0.736 (+/-0.013) for {'max_depth': None, 'max_features': 'sqrt', 'n_estimators': 200}
0.736 (+/-0.01) for {'max_depth': None, 'max_features': 'log2', 'n_estimators': 50}
0.738 (+/-0.014) for {'max_depth': None, 'max_features': 'log2', 'n_estimators': 100}
0.744 (+/-0.014) for {'max_depth': None, 'max_features': 'log2', 'n_estimators': 200}
```

Như vậy, kết quả chỉ ra rằng, việc không giới hạn độ sâu mỗi cây, sử dụng tối đa log2 lượng feature, và có 200 cây quyết định sẽ đưa ra kết quả tốt nhất với độ chính xác 74.4% trên tập Train.

Để kiểm chứng những tham số tốt nhất, ta cần thử trên tập Validation. Ta chọn ra 3 cặp tham số tốt nhất, huấn luyện chúng qua tập Train, và cho chúng thử trên tập Validation. Ngoài ra ta cũng lưu lại thời gian huấn luyện để trợ cho quyết định so sánh.

```
from sklearn.metrics import balanced_accuracy_score, precision_recall_fscore_support
import time

# 3 model tốt nhất
rf1 = RandomForestClassifier(n_estimators=200, max_features = 'log2', max_depth=None, n_jobs=-1)
rf2 = RandomForestClassifier(n_estimators=100, max_features = 'log2', max_depth=None, n_jobs=-1)
rf3 = RandomForestClassifier(n_estimators=200, max_features = 'sqrt', max_depth=None, n_jobs=-1)

for model in [rf1,rf2,rf3]:                      # Thủ từng model
    start = time.time()                          # Bắt đầu đếm thời gian
    model.fit(X_train, y_train.values.ravel())    # Fit với tập train
    y_pred = model.predict(X_val)                # Dự đoán tập validation
    end = time.time()                           # Kết thúc đếm thời gian
    print_results_each(model, time = end-start)   # Hàm phụ trợ in kết quả

MAX FEATURES: log2 / NUMBER OF EST: 200 -- Time: 6.302 / Acc: 0.746 / Precision: 0.746 / Recall: 0.746
MAX FEATURES: log2 / NUMBER OF EST: 100 -- Time: 3.422 / Acc: 0.743 / Precision: 0.744 / Recall: 0.743 / F1: 0.743
MAX FEATURES: sqrt / NUMBER OF EST: 200 -- Time: 15.339 / Acc: 0.737 / Precision: 0.737 / Recall: 0.737 / F1: 0.737
```

Kết quả tương đồng với dữ liệu trong tập Train, hơn nữa thời gian huấn luyện mô hình log2 nhanh hơn hẳn so với mô hình `sqrt`. Vậy, ta sẽ sử dụng mô hình `rf1` làm mô hình chính để thử với tập Test.

Kết quả cuối cùng của mô hình có độ chính xác **74.2%**, với các chỉ số recall và f1-score đều đẹp.

```

from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))

MAX FEATURES: log2 / NUMBER OF EST: 200 -- Acc: 0.754 / Precision: 0.753 / Recall: 0.75 / F1: 0.754
precision    recall   f1-score   support
      -1       0.74      0.77      0.76      883
       1       0.76      0.73      0.75      883

accuracy          0.75      1766

MAX FEATURES: log2 / NUMBER OF EST: 200 -- Time: 0 / Acc: 0.742 / Precision: 0.742 / Recall: 0.742 / F1: 0.742
precision    recall   f1-score   support
      -1       0.74      0.74      0.74      883
       1       0.74      0.74      0.74      883

accuracy          0.74      1766
macro avg       0.74      0.74      0.74      1766
weighted avg     0.74      0.74      0.74      1766

```

5.3.3 Cải thiện dữ liệu đầu vào

Ta có thể cải thiện dữ liệu đầu vào bằng nhiều cách. Ví dụ, ta có thể xét thêm tham số là số lượng từ chửi thề có trong đoạn văn bản. Bài nào nhiều từ chửi thề, khả năng cao sẽ mang ý Tiêu cực.

```

1 cursed_words = (pd.read_csv('Vietnamese_cursed_words.txt'))['word'].tolist()      # Danh sách từ tục tiếng Việt
2 cursed_words = set(cursed_words)
3
4 def contains_cursed_words(text):
5     words = text.lower().split()                                              # Chuyển thành chữ thường
6     number_of_cursed_words = sum(1 for word in words if word in cursed_words)  # Đếm số từ tục
7     return number_of_cursed_words
8
9 # Thêm cột 'noCursedWord' vào DataFrame
10 posts_after_cleaned[['noCursedWord']] = posts_after_cleaned[['originalContent']].apply(contains_cursed_words).apply(pd.Series)

```

Hơn nữa, ta có thể tinh chỉnh danh sách stopword cho phù hợp với tập dữ liệu của ta. Ta xét top 40 từ ảnh hưởng nhiều nhất đến quyết định của mô hình (hình (5.12)).

Ta nhận thấy ngoài những từ mang hàm ý Tích cực (múc, mua, tím, trần, gom, chạy...), và những từ mang ý Tiêu cực (đi, sàn, lỗ, chốt, đập, ...), vẫn còn xuất hiện những từ không mang nhiều ý nghĩa phân định (face, ae, hàng, phiên, hôm, nay, mai, ...). Ta sẽ thực hiện loại bỏ những từ này bằng cách thêm nó vào danh sách stopword (hình (5.13)).

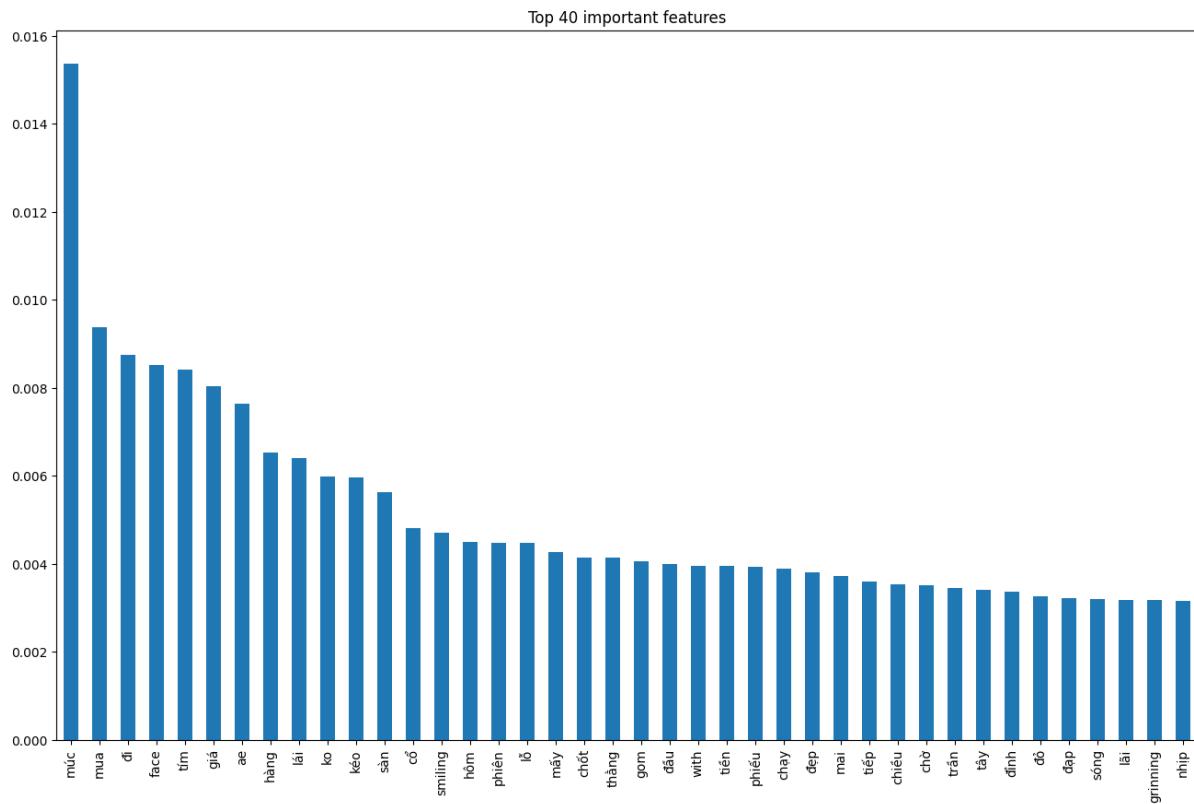
Sau khi thực hiện các cách trên, độ chính xác đã được cải thiện khá tốt, từ **74.2%** lên **75.4%**.

```

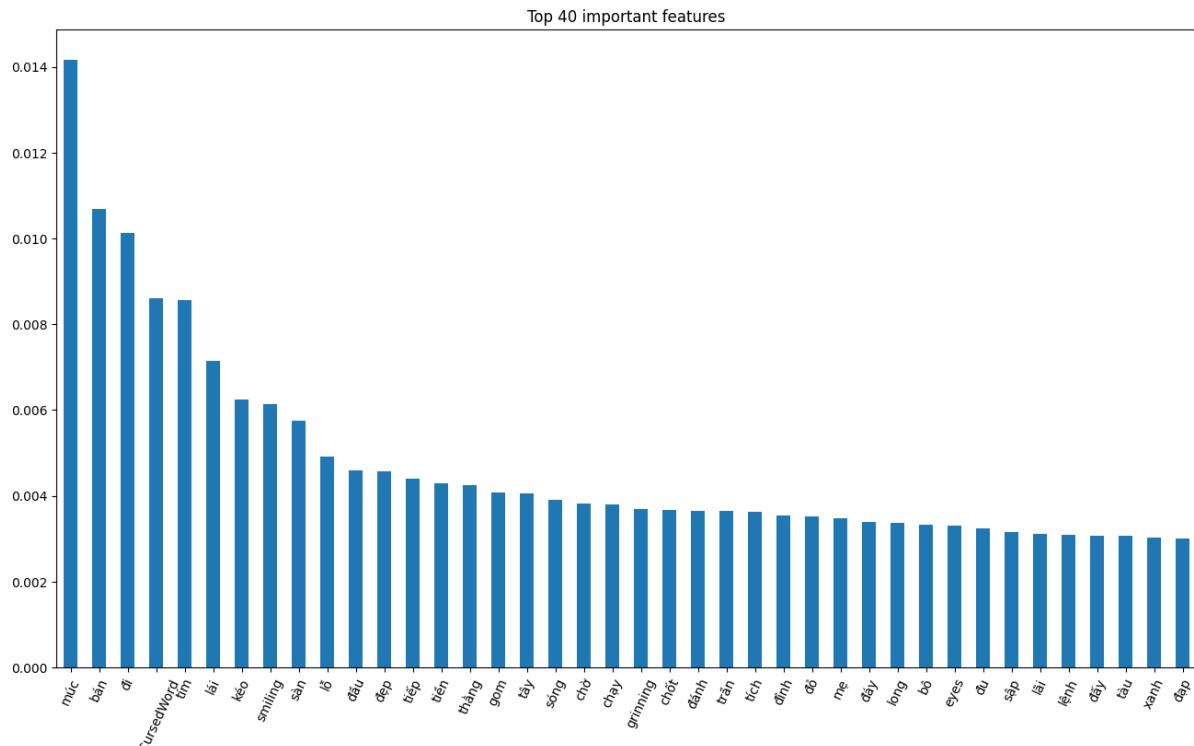
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))

MAX FEATURES: log2 / NUMBER OF EST: 200 -- Acc: 0.754 / Precision: 0.753 / Recall: 0.75 / F1: 0.754
precision    recall   f1-score   support
      -1       0.74      0.77      0.76      883
       1       0.76      0.73      0.75      883

```



Hình 5.12: Top 40 từ ảnh hưởng nhiều nhất tới mô hình (trước khi lọc)



Hình 5.13: Top 40 từ ảnh hưởng nhiều nhất tới mô hình (sau khi lọc một số từ)

5.3.4 Sử dụng thư viện imbalanced-learn

Đặc trưng dữ liệu của ta là sự thiếu cân bằng giữa số bài viết Tích cực và Tiêu cực. Để làm việc thuận tiện hơn với loại dữ liệu này, thư viện [imbalanced-learn](#) [8] ra đời, dựa trên cơ sở là thư viện [scikit-learn](#). Ta sẽ thử nghiệm mô hình `BalancedRandomForestClassifier` có trong thư viện này. Bản chất của nó vẫn là Random Forest, tuy nhiên nó sẽ tự động thực hiện các bước Random Under-sampling mà không cần ta phải cài đặt.

Ta sẽ phân lại dữ liệu đầu vào bằng cách lấy sẵn khoảng 5% số dữ liệu làm tập Test, và 95% còn lại là tập Train. Như vậy tập Test có 1800 bài, còn tập Train có 31130 bài. Tập Test sẽ có số lượng bài Tích cực và Tiêu cực bằng nhau (900 bài), còn tập Train thì không. Kết quả rất ấn tượng, với **76.2%**:

```

from imblearn.ensemble import BalancedRandomForestClassifier
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import cross_validate

rf_clf = make_pipeline(
    BalancedRandomForestClassifier(
        sampling_strategy="all", replacement=True, bootstrap=False,      # Xét toàn bộ mẫu
        n_estimators=200, max_features="log2", max_depth=None,           # Tham số tối ưu từ trước
        random_state=42, n_jobs=-1,
    ),
)
cv_res = cross_validate(rf_clf, X_train_imb, y_train_imb, scoring=scoring)  # Cross validation
print(f"General accuracy: {cv_res['test_accuracy'].mean()}")                 # Trung bình accuracy

rf_clf.fit(X_train_imb, y_train_imb)                                         # Fit với tập train
y_pred = rf_clf.predict(X_test_imb)                                           # Dự đoán tập test
print_results_each2(test_set=y_test_imb)                                       # Hàm phụ trợ in kết quả
] 39.5s

General accuracy: 0.7398329585608737
Acc: 0.762 / Precision: 0.762 / Recall: 0.762 / F1: 0.762
      precision   recall   f1-score   support
      -1       0.76     0.76     0.76     900
       1       0.76     0.77     0.76     900

      accuracy          0.76     1800
      macro avg       0.76     0.76     0.76     1800
  weighted avg       0.76     0.76     0.76     1800

```

Ngoài ra, còn có nhiều kỹ thuật khác trong thư viện `imbalanced-learn` dùng để xử lý dữ liệu mất cân bằng. Ta có thể đặt thêm giả thuyết về dữ liệu hiện tại, và thử nghiệm với nhiều kỹ thuật hơn.

5.3.5 Kết luận, hướng phát triển

Như vậy, ta đã áp dụng nhiều phương pháp để huấn luyện một mô hình học máy thực hiện tác vụ phân tích quan điểm Tiêu cực hoặc Tích cực, và độ chính xác 76.2% đạt được là tương đối tốt. Có rất nhiều tiềm năng để phát triển mô hình này. Đầu tiên, ta có thể sử dụng các mô hình học máy phức tạp hơn, sử dụng Transformer [3], cơ chế Attention cùng các mô hình Học sâu như mạng Nơ-ron, v.v. Phần hiệu chỉnh thông số có thể thử với nhiều thông số hơn, áp dụng nhiều kỹ thuật trong Xử lý ngôn ngữ tự nhiên hơn.

Ngoài ra, ta có thể cải thiện thêm ở phần dữ liệu đầu vào. Trên hết là một Tokenizer để phân tách các từ tiếng Việt tốt hơn, phương pháp vector hóa dữ liệu tốt hơn [4], sau đó là tối ưu hóa danh sách stopword. Ta cũng có thể thử với nhiều phương pháp giảm sự mất cân bằng trong dữ liệu, và cuối cùng, ta cần thêm dữ liệu để có thể dự đoán tốt hơn.

Tổng kết

Trong bản báo cáo này, nhóm chúng em đã tiến thành thu thập dữ liệu của FireAnt trong khoảng hai tháng, vào thời điểm mà thị trường xảy ra nhiều biến động. Nhóm đã cào được **272979 bài viết**, cùng với đó là **75510 bình luận**, được cào từ ngày **06/9/2024** đến ngày **06/11/2024**; kết hợp cùng dữ liệu các mã cổ phiếu. Chúng em cũng đã thực hiện việc tiền xử lý và làm sạch dữ liệu, từ đó rút gọn trường dữ liệu, đồng thời, bổ sung thêm dữ liệu về các chỉ số trong nước và quốc tế.

Sau đó, chúng em cũng đã đưa ra phân tích, nhận định khái quát những dữ liệu bài đăng và phản hồi, đưa ra giá trị trung vị, giá trị trung bình của số bình luận và bài viết, thống kê được thời gian vàng mà người dùng có lượng tương tác đạt mức kỉ lục cũng như các từ khóa mà người dùng hay nhắc đến. Qua đó, thực hiện đánh giá cộng đồng người dùng FireAnt ta có được thống kê lượng người phản hồi và đăng bài mỗi ngày, tìm ra được sự tương quan của người dùng và số bài đăng cũng như là đánh giá được lượng phản hồi của người dùng. Tỉ lệ bài viết xuất hiện từ thiếu văn minh cũng như tỉ lệ quan điểm tích cực và tiêu cực cũng đã phản ánh được khái quát về văn hóa và tính chất của cộng đồng FireAnt nói riêng và Việt Nam nói chung.

Thị trường chứng khoán Việt Nam cũng được phân tích và đánh giá một cách tổng quan, sơ bộ. Nhìn chung, sự phân bố các ngành và các yếu tố như vốn hóa thị trường, số cổ phiếu lưu hành hay là quan điểm người dùng về ngành đã được phân tích và trực quan hóa ở dạng biểu đồ và dataframe, từ đó nhận xét quy mô thị trường của các ngành nước nhà. Trong đó, 3 chỉ số thị trường lớn là **VNINDEX**, **VN30** và **HNXINDEX** có sự tương quan sâu sắc lẫn nhau cùng với các chỉ số thị trường nước ngoài với những con số vượt ngoài mong đợi.

Qua hai phân tích trên về thị trường chứng khoán Việt Nam và FireAnt, điều đầu tiên chúng em rút ra được là xu hướng khối lượng giao dịch có thể tăng do nhà đầu tư chú ý và phản ứng với thông tin hay có xuất hiện sự kiện thu hút dòng vốn mới vào cổ phiếu liên quan, dẫn đến khẳng định rằng khi số lượng bài viết tăng thì khối lượng giao dịch cũng tăng tương ứng. Đồng thời chúng em đã biểu diễn được số lượng bài viết theo phản ứng người dùng khi thị trường biến động mạnh hay yếu được phản ánh bởi các scatter plot giữa phần trăm biến động của VNINDEX và số bài viết, giải thích cho các điểm dị thường trong đồ thị và khẳng định sự tương tác người dùng có sự đồng nhất với biến đổi thị trường (biến đổi cùng mạnh thì tương tác càng nhiều và ngược lại), qua đó khẳng định xu hướng rằng **khi giá tăng, số lượng bài viết tiêu cực có xu hướng giảm và số lượng bài viết tích cực cũng tăng theo**.

Ngoài ra, chúng em cũng đã sử dụng Bayes và xích Markov để ứng dụng vào dự đoán giá trong ngày. Nhìn chung, tuy không thể áp dụng vào để dự đoán chính xác hoàn toàn sự biến động của thị trường trong thực tế, số lượng bài đăng trên diễn đàn không phải nguyên nhân chính khiến giá biến động nhưng có thể được sử dụng như một hình thức tham khảo hoặc loại cảnh báo và có thể kết hợp với các thuật toán phức tạp khác để tăng sự chính xác.

Nổi bật hơn cả là mô hình học máy phân tích quan điểm dựa trên nội dung bài viết sử dụng Random Forest. Tại đó, với dữ liệu được trích từ toàn bộ 270 nghìn bài viết, chúng em đã phân loại các bài tích cực và tiêu cực, lọc các stopword và split dữ liệu. Thực hiện train lần một, sau đó cải thiện dữ liệu đầu vào và tiến hành train lại lần hai đã có kết quả khả quan hơn với độ chính xác ở mức tương đối tốt (**76.2%**), vậy nên việc có thể áp dụng mô hình vào phân tích quan điểm là khả quan. Chúng em cũng đã đưa ra hướng cải thiện, như thử nhiều phương pháp khác, áp dụng Học sâu, để tăng độ chính xác của mô hình.

Dự định tương lai

Mục tiêu chính của chúng em, không chỉ dừng lại ở đây với tư cách là một bài tập lớn mà sẽ là một dự án đầy đủ, phân tích sâu hơn tâm lý thị trường và xu hướng chung của những nhà đầu tư. Vậy nên trong tương lai, chúng em dự định sẽ cào thêm nhiều dữ liệu, tối ưu mô hình, phân tích sâu sắc hơn để có những số liệu chính xác. Trong quá trình hiện thực hóa ý tưởng này, nhóm chúng em sẽ liên tục cải thiện trình độ của bản thân, cũng như lôi kéo các nguồn lực để phát triển.

Chúng em chân thành cảm ơn thầy, cô và các độc giả đã dành thời gian để đọc bản báo cáo này của chúng em. Một lần nữa chúng em xin cảm ơn thầy cô về một đề tài mở thú vị, giúp chúng em nâng cao hiểu biết trong lĩnh vực Lập trình Xử lý dữ liệu bằng Python. Nhóm chúng em hi vọng sẽ nhận được những góp ý cũng như phản hồi tích cực từ phía thầy cô.

Trân trọng,

Nhóm 7, Đánh dấu lỗ đó.

Tài liệu tham khảo

- [1] GitHub: BlueEyesVN. Danh mục từ thô tục của Việt Nam. <https://github.com/blue-eyes-vn/vietnamese-offensive-words>, 2021.
- [2] Mamata Das, Selvakumar K., and P. J. A. Alphonse. A Comparative Study on TF-IDF feature Weighting Method and its Analysis using Unstructured Dataset. <https://arxiv.org/abs/2308.04037>, 2023.
- [3] Pinak Datta. Building a Multimodal Sentiment Analysis Model using Transformers. <https://medium.com/@pinakdatta/mastering-multimodal-sentiment-analysis-with-transformers-a7d0acf47569>, 2024.
- [4] An Long Doan and Son T. Luu. Improving Sentiment Analysis By Emotion Lexicon Approach on Vietnamese Texts. In *2022 International Conference on Asian Language Processing (IALP)*, page 39–44. IEEE, October 2022. <http://dx.doi.org/10.1109/IALP57159.2022.9961318>.
- [5] FireAnt. Trang chủ FireAnt. <https://fireant.vn>, 2024.
- [6] GeeksForGeeks. Markov Chains Python Implement. <https://www.geeksforgeeks.org/markov-chain/>, 2021. [Online; truy cập 11/2024].
- [7] Yan Holtz and Conor Healy. From data to viz. <https://www.data-to-viz.com/>, 2018. [Online; truy cập 11/2024].
- [8] imbalanced-learn developers. imbalanced-learn documentation. <https://imbalanced-learn.org/stable/index.html>, 2024. [Online; truy cập 11/2024].
- [9] Duyet Le. vietnamese-stopwords. <https://github.com/stopwords/vietnamese-stopwords/blob/master/vietnamese-stopwords.txt>, 2017.
- [10] Melvin Mokhtari, Ali Seraj, Niloufar Saeedi, and Adel Karshenas. The Impact of Twitter Sentiments on Stock Market Trends. <https://arxiv.org/abs/2302.07244>, 2023.
- [11] Kaggle: Ona_Gilbert. Sentiment Analysis with TFIDF and Random Forest. <https://www.kaggle.com/code/onadegibert/sentiment-analysis-with-tfidf-and-random-forest>, 2019.
- [12] Nguyễn Phan Trúc Phương, Hồ Thị Lam, and Võ Xuân Vinh. Mối quan hệ giữa thị trường chứng khoán các nước và Việt Nam. <https://ktpt.neu.edu.vn/Uploads/Bai%20bao/2022/So%20299/380642.pdf>, 2022.
- [13] VNStocks team. Tài liệu thư viện vnstocks. <https://vnstocks.com/docs/tai-lieu/huong-dan-nhanh>, 2024.

- [14] Tri thức cộng đồng. Sự khác biệt giữa Tương quan (Correlation) và Hồi quy. <https://trithuccongdong.net/hoi-dap/tuong-quan-la-gi-su-khac-biet-giu-a-tuong-quan-correlation-va-hoi-quy.html>, 2024. [Online; truy cập 11/2024].
- [15] VNExpress. VN-Index, VN30-Index và cách tính toán. <https://vnexpress.net/vn-index-vn30-index-va-cach-tinh-toan-4297351.html>, 2022. [Online; truy cập 11/2024].