TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO

LƯU TRỮ VÀ XỬ LÝ DỮ LIỆU LỚN

Đề tài:

Phân tích dữ liệu các trang báo online

Giáo viên hướng dẫn: PGS. Nguyễn Bình Minh

TS. Đào Thành Chung

Học viên thực hiện: Đỗ Minh Khang – 20211030M

Nguyễn Ngọc Khiêm – 20211031M

Nguyễn Văn Tiến – 20211036M

Trần Đoan Trang – 20211038M

HÀ NỘI - 2022

MỤC LỤC

DANH MỤC HÌNH ẢNH

Phần 1. Tổng quan bài toán

1. Tên chương trình

Phân tích dữ liệu các trang báo online.

2. Mục đích

Chương trình được xây dựng với mục đích:

- Tìm hiểu các thông tin từ dữ liệu thu thập được từ các trang báo online.
- Đưa ra được một số nhận định từ dữ liệu được trực quan hoá.

3. Công nghệ sử dụng

Các công nghệ sử dụng trong bài toán:

- Spark với thư viện PySpark (v2.4.5)
- Zeppelin Notebook làm editor (v0.9.0)
- GitHub để quản lý mã nguồn
- Ứng dụng chạy trên nền Docker
- Thu thập dữ liệu với Scrapy

4. Phân Công công việc

Công việc được phân chia cho các thành viên trong nhóm như sau:

- **Tiến, Trang:**
  - Thu thập dữ liệu
  - Tiền xử lý dữ liệu
  - Cấu hình phân cụm
- **Khiêm, Khang:**
  - Tiền xử lý dữ liệu
  - Phân tích một số thông tin khác từ dữ liệu

Phần 2: Thiết kế hệ thống

1. Thu thập dữ liệu

a. Công cụ và nguồn dữ liệu.

- Dữ liệu được thu thập bằng công cụ Scrapy với Python
- Các nguồn báo được thu thập là:
  - 24h
  - Vietnamnet
  - VNExpress
- Kích thước dữ liệu: 24h: 700M, VietNamnet : 60M, VNExpress : 150M
- Crawl dữ liệu: Sử dụng thư viện Scrapy

Scrapy là một framework hỗ trợ việc khai thác thông tin từ các website. Quá trình crawl dữ liệu sử dụng Scrapy như sau:

  - Scrapy gửi request get tới các các địa chỉ trang báo để nhận về các tài liệu HTML.
  - Từ HTML này có thể tìm các href tới các bài báo khác của cùng trang Web, đồng thời lấy các thông tin cần về bài báo.

- Chi tiết Crawl dữ liệu :

- Tạo project scrapy: sử dụng câu lệnh

```
scrapy startproject tutorial
```

Sau khi tạo project có cấu trúc:

```
    tutorial/
    scrapy.cfg       # deploy configuration file
    tutorial/        # project's Python module, you'll import
your code from here
    __init__.py
    items.py          # project items definition file
    middlewares.py    # project middlewares file
    pipelines.py      # project pipelines file
    settings.py       # project settings file
```

```
    spiders/        # a directory where you'll later put your
spiders
    __init__.py
```

- Tại thư mục spider chúng ta có thể tạo thêm ứng dụng với mục đích crawl dữ liệu:

Viết một lớp Scrapy kết thừa lớp `CrawlSpider`

```
    class MySpider(CrawlSpider):
    name = 'vnexpress' // tên cửa ứng dụng
    allowed_domains = ["vnexpress.net"]
    start_urls = ['https://vnexpress.net /'] //địa chỉ bắt
đầu crawl
    rules = (
    Rule(LinkExtractor(allow=(r'\.htm$')),callback='parse
_item',follow=True)
    )
    // các luật để lọc ra các trang html cần thiết
    def parse_item(self, response):
    // hàm triển khai việc khai thác thông tin từ file
html
```

- Mã nguồn crawl trang Web VnExpress:

```
    def parse_item(self, response):
            #vexpress
            self.logger.info('Hi, this is an item page! %s',
response.url)
            headline        =        response.css("h1.title-
detail::text").get()
            if headline is not None:
                 headline = headline.rstrip().strip()
            print("////////////////////",headline)
            des = response.css("p.description::text").get()
            if des is not None:
```

```
                des = des.rstrip().strip()
            print("////////////////////",des)
            content = response.css("p.Normal::text").getall()
            if content is not None:
                content = " ".join(content)
            print("////////////////////",content)
            date = response.css("span.date::text").get()
            if date is not None:
                des = date.split(",")[1]
            if date is not None:
                des = date.rstrip().strip()
            print("////////////////////",date)
            cate = response.css("ul.breadcrumb >li
>a::text").get()
            if cate is not None:
                cate = cate.rstrip().strip()
            print("////////////////////",cate)
            if (headline is not None) and (des is not None) and
(content is not None) and (date is not None) and (cate is not
None):
                writer.writerow({"headline": headline,
"description": des, "content": content, "datepublish":
date,"cate":cate})
```

- Mã nguồn crawl trang Web 24h:

```
    class MySpider(CrawlSpider):
       name = '24h'
       start_urls = ['https://www.24h.com.vn/']
       rules =
(Rule(LinkExtractor(allow=(r'\.html$')),callback='parse_item'
,follow=True),)
```

```
       def parse_item(self, response):


           #24h
           self.logger.info('Hi, this is an item page! %s',
response.url)
           headline = response.xpath("//h1[@class='clrTit bld
tuht_show']/text()").get()
           if headline is not None:
                headline = headline.rstrip().strip()
           print("////////////////////",headline)
           des       =       response.xpath("//h2[@class='ctTp
tuht_show']/text()").get()
           if (des is None):
               des        =        response.css("h2.tuht_show
>strong::text").get()
           if des is not None:
                des = des.rstrip().strip()
           print("////////////////////",des)
           content = response.xpath("//p/text()").getall()
           if content is not None:
               content = " ".join(content)
           print("////////////////////",content)
           date  =    response.xpath("//time[@class='cate-24h-
foot-arti-deta-cre-post']/text()").get()
           if (date is None):
                date   =   response.xpath("//div[@class='updTm
updTmD mrT5']/text()").get()
           if date is not None:
                date = date.split(",")[1]
                date = date.split(" ")[2]
```

```
            print("////////////////////",date)
            cate   =   response.css("ul.brm   >li   >a.brmItem
>span::text").get()
            if (cate is None):
                cate     =     response.css("ul.d-flex     >li
>a.active::text").get()
            if cate is not None:
                 cate = cate.rstrip().strip()
            print("////////////////////",cate)
            if (headline is not None) and (des is not None) and
(content is not None) and (date is not None) and (cate is not
None):
                writer.writerow({"headline":          headline,
"description":   des,   "content":   content,   "datepublish":
date,"cate":cate})                writer.writerow({"headline":
headline,    "description":    des,    "content":    content,
"datepublish": date,"cate":cate})
```

- Mã nguồn crawl trang Web Vietnamnet:

```
    class MySpider(CrawlSpider):
       name = 'vietnamnet'
       allowed_domains = ["vietnamnet.vn"]
       start_urls = ['https://vietnamnet.vn/']
       rules                                                =
(Rule(LinkExtractor(allow=(r'\.htm$')),callback='parse_item',
follow=True),)
       def parse_item(self, response):

           #Vietnamnet
           self.logger.info('Hi, this is an item page! %s',
response.url)
```

```
            headline = response.xpath("//h1[@class='title-page
detail']/text()").get()
            if headline is not None:
                 headline = headline.rstrip().strip()
            print("////////////////////",headline)
            des    =    response.xpath("//h2[@class='singular-
sapo']/text()").get()
            if des is not None:
                 des = des.rstrip().strip()
            print("////////////////////",headline)
            content = response.xpath("//p/text()").getall()
            if content is not None:
                content = " ".join(content)
            print("////////////////////",content)
            date = response.css("time.author-time::text").get()

            if date is not None:
                 date = date.split(",")[1]
                 date = date.split(" ")[1]
            if date is not None:
                 date = date.rstrip().strip()
            print("////////////////////",date)
            cate     =     response.css("ul.breadcrumbs     >li
>a::text").get()
            print("////////////////////",cate)
            if (headline is not None) and (des is not None) and
(content is not None) and (date is not None) and (cate is not
None):
```

```
                writer.writerow({"headline":          headline,
"description":   des,   "content":   content,   "datepublish":
date,"cate":cate})
```

- Cấu hình trong khi crawl dữ liệu:

Trong file setting ta điều chỉnh các trường:

```
    Redirect_enabled = True # cho phép redirect url
    DOWNLOAD_MAXSIZE = 0 #Không giới hạn kích thước dữ liệu đã
tải về
    CONCURRENT_REQUESTS = 20 # gửi các request cùng lúc
     # request theo breadth first, ưu tiên các trang web ở gần
     đây
    DEPTH_PRIORITY = 1
    SCHEDULER_DISK_QUEUE                                        =
'scrapy.squeues.PickleFifoDiskQueue'
    SCHEDULER_MEMORY_QUEUE = 'scrapy.squeues.FifoMemoryQueue'
```

b. Mô tả dữ liệu

Các bài báo được thu thập với các thông tin sau:

Name	Type	Desc
headline	string	Tiêu đề (title) của bài báo
description	string	Mô tả (subtitle) của bài báo, 1 câu ngắn gọn để mô tả bài báo.
datepublish	string	Thời gian đăng bài viết
content	string	Nội dung bài báo
category	string	Category của bài báo

2. Tiền xử lý dữ liệu

Có một số thông tin trong dữ liệu thu thập về cần được tiền xử lý để cho quá trình phân tích dữ liệu được dễ dàng hơn.

a. Load data

Đầu tiên load data từ file csv đã thu thập được từ quá trình crawl data. Bao gồm 3 file 24h.csv, vnexpress.csv, vietnamenet.csv

```
    # Check data file path
    %spark.pyspark
    DATA_24h_PATH = '../../data/24h.csv'
    DATA_VNEXPRESS_PATH = '../../data/vnexpress.csv'
    DATA_VIETNAMNET_PATH = '../../data/vietnamnet.csv'

    # load data
    data_24h_df = spark.read.options(header=True, inferSchema=True, multiline=True, quote='"', escape='"', encoding="UTF-8").csv(DATA_24h_PATH)
    data_VNE_df = spark.read.options(header=True, inferSchema=True, multiline=True, quote='"', escape='"', encoding="UTF-8").csv(DATA_VNEXPRESS_PATH)
    data_VNN_df = spark.read.options(header=True, inferSchema=True, multiline=True, quote='"', escape='"', encoding="UTF-8").csv(DATA_VIETNAMNET_PATH)
```

Kết quả thu được:

```
24h records: 169357
VNExpress records: 43798
VietNamNet count: 13946

Took 35 sec. Last updated by anonymous at July 30 2022, 9:50:59 PM.
```

Hình 1. Kết quả load data từ 3 websites

b. Tổng hợp các data source:

- **Desc**: Do quá trình crawl data, dữ liệu của 3 trang được lưu lại 3 files khác nhau, khi đọc vào cũng được lưu dưới 3 dataframe khác nhau. Cần gộp 3 dataframe này lại với nhau để thực hiện các truy vấn được thuận tiện hơn.

- **Xử lý**:

```
%spark.pyspark
# init source
data_24h_df = data_24h_df.withColumn('source', F.lit("24h"))
data_VNE_df = data_VNE_df.withColumn('source', F.lit("VNExpress"))
data_VNN_df = data_VNN_df.withColumn('source', F.lit("VietNamNet"))

# Merge
data_df = data_VNE_df.union(data_VNN_df).union(data_24h_df)
print("Total records:",data_df.count())
```

- **Kết quả**: `Total records: 227101`

c. Xử lý date_published

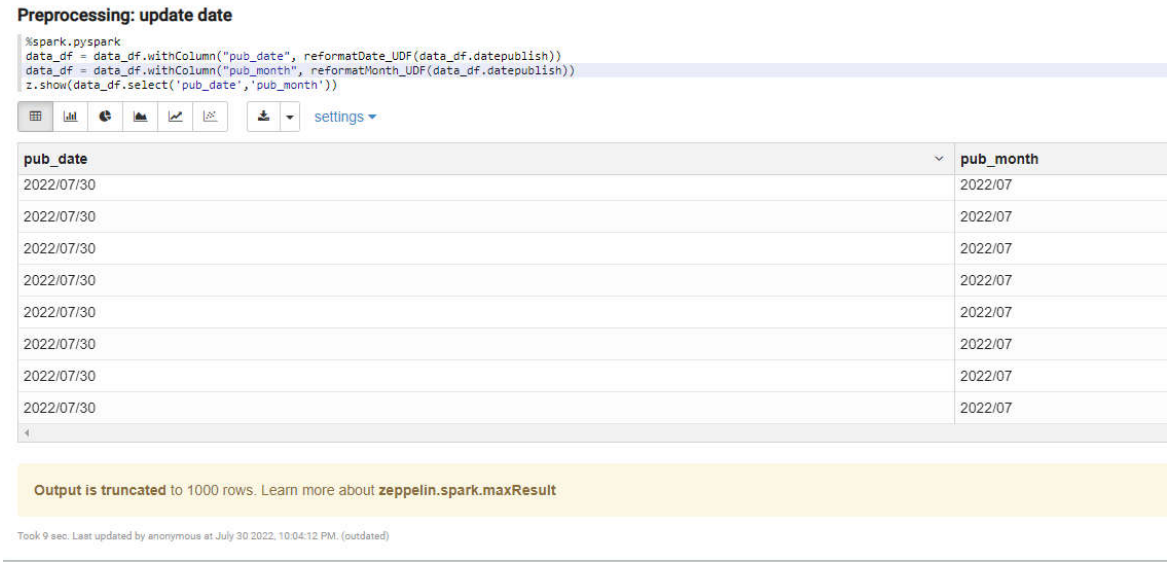
- **Desc**: Do một số bài viết ngày tháng được viết dưới dạng 1/1/22 một số khác lại có dạng đầy đủ 1/1/2022, do vậy cần thống nhất lại cách viết dạng đầy đủ 1/1/2022, đồng thời sẽ đảo ngược thứ tự YYYY/MM/dd để thuận lợi trong việc sắp xếp theo thứ tự thời gian.
  - `pub_date`: YYYY/MM/dd
  - `pub_month`: YYYY/MM
- **Xử lý**:

```
%spark.pyspark
# Change all date to type: YYYY-MM-dd
def reformatDate(date):
    arr = date.split('/')
    if(len(arr) == 3):
        if len(arr[2]) < 4:
```

```
                return '20'+arr[2] + '/' + arr[1].zfill(2) +
'/' + arr[0].zfill(2)
            else:
                return  arr[2]+'/'+  arr[1].zfill(2)  +  '/'  +
arr[0].zfill(2)
        else:
            return "1970/01/01"
        return "1970/01/01"
    reformatDate_UDF = F.udf(lambda date: reformatDate(date))

    # Change all date to type: YYYY-MM
    def reformatMonth(date):
        arr = date.split('/')
        if(len(arr) == 3):
            if len(arr[2]) < 4:
                return '20'+arr[2] + '/' + arr[1].zfill(2)
            else:
                return arr[2]+'/'+ arr[1].zfill(2)
        else:
            return "1970/01"
        return "1970/01"
    reformatMonth_UDF        =        F.udf(lambda        date:
reformatMonth(date))

    data_df          =          data_df.withColumn("pub_date",
reformatDate_UDF(data_df.date_published))

    data_df          =          data_df.withColumn("pub_month",
reformatMonth_UDF(data_df.date_published))
```

```
z.show(data_df)
```

- **Kết quả**: OK



Hình 2. Kết quả xử lý date_published

d. Xử lý category

- **Desc**: Do mỗi một trang báo sẽ có một cách phân chia category khác nhau, nên chúng ta cần gộp những category giống hoặc gần giống nhau vào làm 1 để data được đồng nhất. (Hiện tại có 142 categories)

Hình 3. Danh sách category sau khi xử lý

- **Xử lý**: Các category sau sẽ được gộp lại thành 10 categories theo bảng sau:

Name	Sub-Categories
Tin tức	'Thế giới','Tin tức trong ngày','Pháp luật','Thời sự','Xã hội','Ý kiến','Góc nhìn','Xu hướng tiêu dùng độc lạ','Tấm lòng nhân ái','Bạn đọc','Video An ninh','Tết 2022','Tin tức du lịch','News','Nhân tài Đất Việt','Tuyến đầu chống dịch','Tư vấn','Tổng hợp','Đánh giá sản phẩm','Máy tính để

	bàn','VnExpress 20 năm','Bảng giá','Tết Rộn Ràng','Giới trẻ','Du học','Tuyển sinh','Thủ thuật - Tiện ích','Đánh giá'
Sức khoẻ- Đời sống	'Bạn trẻ - Cuộc sống','Sức khỏe đời sống','Ẩm thực','Sức khỏe','Thời trang Hi-tech','Du lịch','Làm đẹp','Thị trường - Tiêu dùng','Đời sống','An sinh','Nhịp sống trẻ','Tình yêu - Giới tính','Tin tức sức khỏe','Dịch viêm phổi virus corona','Chuyên mục test','Tin tức Ẩm thực','Sống khỏe hậu COVID','Điểm du lịch','Du lịch Việt Nam','Tâm sự','Sức khỏe tình dục','Bệnh trẻ em','Sức khỏe dinh dưỡng','Ung thư','Bệnh đàn ông','An toàn thực phẩm','Gan','Bác sĩ của bạn','Sợ Virus ???'
Giải trí	'Giải trí','Đời sống Showbiz','Phi thường - kỳ quặc','Media','Cười 24H','Văn hóa','Du Lịch','Blog','Hài','Clip hài hước','Movies - Muzik','Thơ vui','Hội quán 24H','Tranh vui','Đố vui','Truyện tiếu lâm'
Kinh doanh	'Kinh doanh','Bất động sản','Sản xuất - Tiêu dùng','Lao động - Việc làm','Tài chính','Doanh nghiệp','Chứng khoán','Giá vàng hôm nay','Doanh nhân','Giá cả hàng hóa','Ngân hàng','Khởi nghiệp','Tiền tiền tiền','Tỷ giá ngoại tệ','Tin giá vàng','Thị trường 24h','Kinh tế cho tương lai','Tin Tài chính - Nhà đất - BĐS','Thời cuộc','Tìm việc làm Tuyển dụng'
Thể thao	'Bóng đá','Thể thao','EURO 2020','Asiad 2018','SEA Games 2019','Olympic 2016','World Cup 2022','Tin tức thể thao','Top ghi bàn','Bóng đá Đức - Bundesliga','Lịch bóng đá TV','SEA Games 31','Kết quả bóng đá','World Cup 2018','Lịch thi đấu tennis','Olympic Tokyo 2021','AFF Cup 2021','Tennis','Bóng rổ NBA - VBA','Xếp hạng FIFA','Lịch thi đấu thể thao','Bóng đá Ngoại hạng Anh','Olympic 2012','Ấn tượng SEA Games'
Giáo dục	'Giáo dục - du học','Khoa học','Giáo dục','Giáo dục - Hướng nghiệp','Tin tức Giáo dục'

Công nghệ	'Công nghệ thông tin','Số hóa','Sức mạnh số','Khoa học - Công nghệ','Khám phá công nghệ','Thế giới công nghệ','Tin tức công nghệ','Điện tử gia dụng'
Thời trang	'Thời trang','Style','Stars'
Xe	'Ô tô','Xe máy - Xe đạp','Ô tô - Xe máy','Xe','Tin tức ô tô','Thế giới xe','Bảng giá xe','Xe xịn','Bảng giá xe ô tô'
Others	

```
%spark.pyspark
# validate function
News = ['Thế giới','Tin tức trong ngày','Pháp luật','Thời sự','Xã hội','Ý kiến','Góc nhìn','Xu hướng tiêu dùng độc lạ','Tấm lòng nhân ái','Bạn đọc','Video An ninh','Tết 2022','Tin tức du lịch','News','Nhân tài Đất Việt','Tuyến đầu chống dịch','Tư vấn','Tổng hợp','Đánh giá sản phẩm','Máy tính để bàn','VnExpress 20 năm','Bảng giá','Tết Rộn Ràng','Giới trẻ','Du học','Tuyển sinh','Thủ thuật - Tiện ích','Đánh giá']
SucKhoeDoiSong = ['Bạn trẻ - Cuộc sống','Sức khỏe đời sống','Ẩm thực','Sức khỏe','Thời trang Hi-tech','Du lịch','Làm đẹp','Thị trường - Tiêu dùng','Đời sống','An sinh','Nhịp sống trẻ','Tình yêu - Giới tính','Tin tức sức khỏe','Dịch viêm phổi virus corona','Chuyên mục test','Tin tức Ẩm thực','Sống khỏe hậu COVID','Điểm du lịch','Du lịch Việt Nam','Tâm sự','Sức khỏe tình dục','Bệnh trẻ em','Sức khỏe dinh dưỡng','Ung thư','Bệnh đàn ông','An toàn thực phẩm','Gan','Bác sĩ của bạn','Sợ Virus ???']
GiaiTri = ['Giải trí','Đời sống Showbiz','Phi thường - kỳ quặc','Media','Cười 24H','Văn hóa','Du Lịch','Blog','Hài','Clip hài hước','Movies - Muzik','Thơ vui','Hội quán 24H','Tranh vui','Đố vui','Truyện tiếu lâm']
```

```
    KinhDoanh = ['Kinh doanh','Bất động sản','Sản xuất - Tiêu dùng','Lao động - Việc làm','Tài chính','Doanh nghiệp','Chứng khoán','Giá vàng hôm nay','Doanh nhân','Giá cả hàng hóa','Ngân hàng','Khởi nghiệp','Tiền tiền tiền','Tỷ giá ngoại tệ','Tin giá vàng','Thị trường 24h','Kinh tế cho tương lai','Tin Tài chính - Nhà đất - BĐS','Thời cuộc','Tìm việc làm Tuyển dụng']
    TheThao = ['Bóng đá','Thể thao','EURO 2020','Asiad 2018','SEA Games 2019','Olympic 2016','World Cup 2022','Tin tức thể thao','Top ghi bàn','Bóng đá Đức - Bundesliga','Lịch bóng đá TV','SEA Games 31','Kết quả bóng đá','World Cup 2018','Lịch thi đấu tennis','Olympic Tokyo 2021','AFF Cup 2021','Tennis','Bóng rổ NBA - VBA','Xếp hạng FIFA','Lịch thi đấu thể thao','Bóng đá Ngoại hạng Anh','Olympic 2012','Ấn tượng SEA Games']
    GiaoDuc = ['Giáo dục - du học','Khoa học','Giáo dục','Giáo dục - Hướng nghiệp','Tin tức Giáo dục']
    CongNghe = ['Công nghệ thông tin','Số hóa','Sức mạnh số','Khoa học - Công nghệ','Khám phá công nghệ','Thế giới công nghệ','Tin tức công nghệ','Điện tử gia dụng']
    ThoiTrang = ['Thời trang','Style','Stars']
    Xe = ['Ô tô','Xe máy - Xe đạp','Ô tô - Xe máy','Xe','Tin tức ô tô','Thế giới xe','Bảng giá xe','Xe xịn','Bảng giá xe ô tô']
    def updateCategory(cate):
        if cate in News:
            return "Tin tức"
        elif cate in SucKhoeDoiSong:
            return "Sức khoẻ - Đời sống"
        elif cate in GiaiTri:
            return "Giải trí"
```

```
        elif cate in KinhDoanh:
            return "Kinh doanh"
        elif cate in TheThao:
            return "Thể thao"
        elif cate in GiaoDuc:
            return "Giáo dục"
        elif cate in CongNghe:
            return "Công nghệ"
        elif cate in ThoiTrang:
            return "Thời trang"
        else:
            return "Khác"

    updateCategory_UDF = F.udf(lambda cate:
updateCategory(cate))
    data_df = data_df.withColumn("category",
updateCategory_UDF(data_df.category))
```

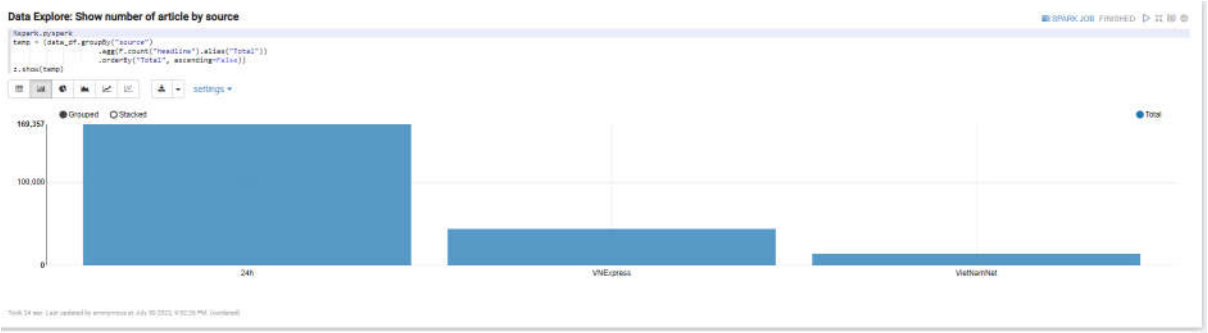
3. Một số thống kê từ dữ liệu

a. Thống kê về nguồn bài báo

- **Desc**: Thống kê số lượng bài báo theo nguồn.
- **Code**:

```
    %spark.pyspark
    temp = (data_df.groupBy("source")

.agg(F.count("headline").alias("Total"))
                        .orderBy("Total", ascending=False))
    z.show(temp)
```

- **Kết quả**:



Hình 4. Kết quả thống kê theo nguồn bài báo

- **Nhận xét**:
  - Số lượng bài báo thu được chủ yếu quyết định dựa trên quá trình crawl.
  - Lượng bài báo đến từ 3 trang hiện đang ngang nhau, mỗi trang có khoảng 5100 bài viết.

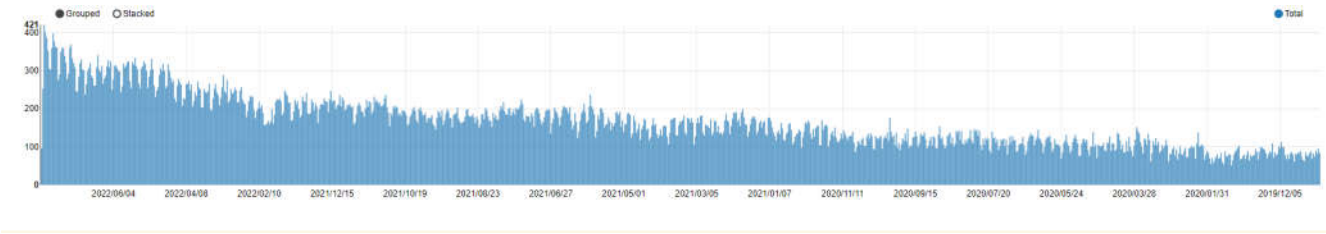
b. Thống kê số lượng bài viết theo ngày / tháng

- **Desc**: Thống kê số lượng bài viết theo ngày / tháng.
- **Code**: Dựa vào trường "pub_date" và "pub_month" đã được tiền xử lý trước đó.

```
%spark.pyspark
temp = (data_df.groupBy("pub_date")
          .agg(F.count("headline").alias("Total"))
          .orderBy("pub_date", ascending=False))
z.show(temp)
%spark.pyspark
temp = (data_df.groupBy("pub_month")
           .agg(F.count("headline").alias("Total"))
           .orderBy("pub_month", ascending=False))
z.show(temp)
```

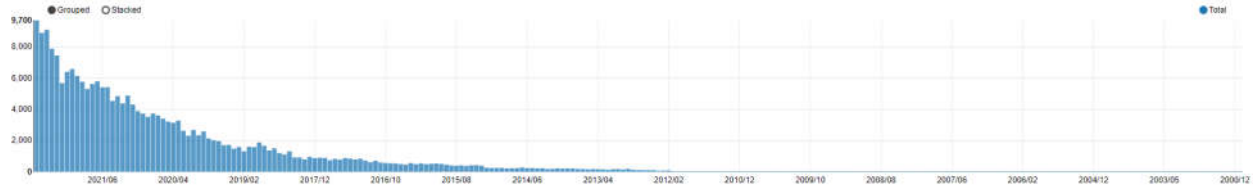
- **Kết quả**:

Theo ngày:



Hình 5. Kết quả thống kê số lượng bài báo theo ngày

Theo tháng:

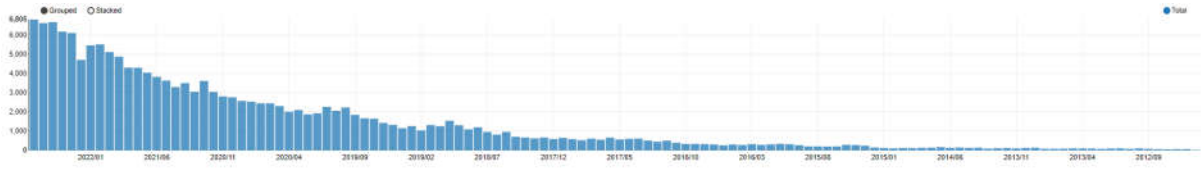


Hình 6. Kết quả thống kê số lượng bài báo theo tháng

- **Nhận xét**:
  - Data cần gần với hiện tại thì càng nhiều, có 2 nguyên nhân:
    - Quá trình crawl data được thực hiện dựa theo các bài viết mới nhất rồi tới các bài viết trước đó.
    - Ngoài ra còn một phần do số lượng bài viết cũng ngày càng nhiều lên.
  - Ngày nhiều bài viết nhất: 22/07/2021 với 421 bài.
  - Tháng nhiều bài viết nhất: 07/2021 với 9700 bài.

Thống kê lại theo từng trang cũng cho kết quả tương tự:

- 24h:



Hình 7. Thống kê số lượng bài báo trên trang 24h

- VietnamNet:

Hình 8. Thống kê số lượng bài báo trên trang VietnamNet

- VNExpress:

Hình 9. Thống kê số lượng bài báo trên trang VNExpress

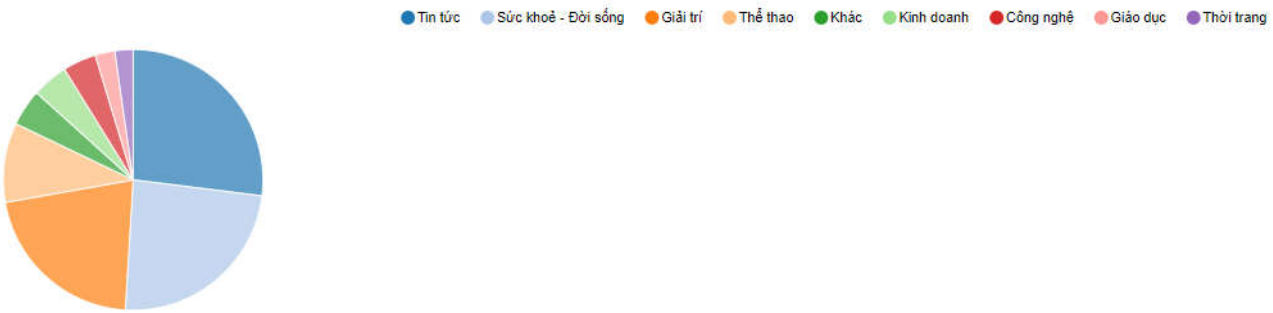
***NOTE**: Chính vì lý do này nên một số truy vấn phía sau sẽ chỉ xét trong khoảng thời gian gần đây để đảm bảo tính khách quan nhất có thể!*

c. Thống kê category

- **Desc**: Thống kê số lượng bài viết theo category
- **Code**:

```
%spark.pyspark
temp = (data_df.groupBy("category")
            .agg(F.count("headline").alias("Total"))
            .orderBy("Total", ascending=False))
z.show(temp)
```

- **Kết quả**:



Hình 10. Kết quả thống kê theo category

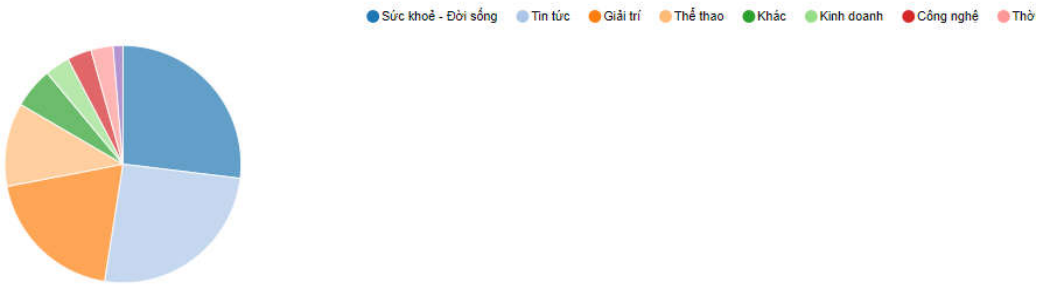
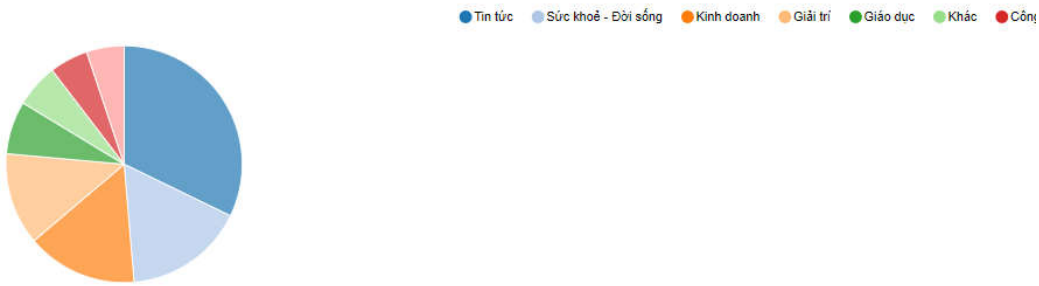
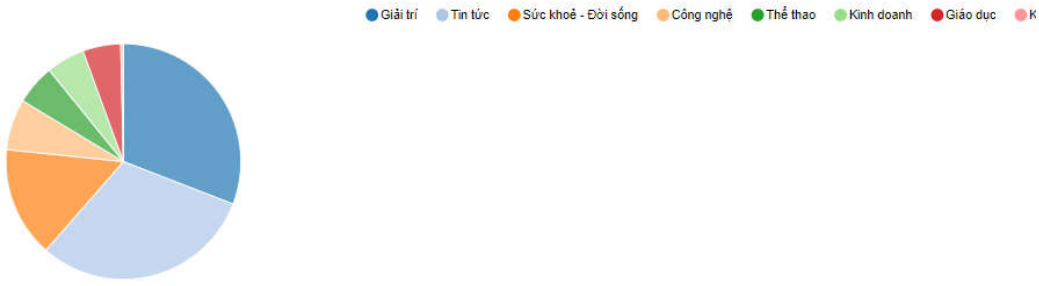
- **Nhận xét**:
  - Số lượng bài viết lớn nhất thuộc thể loại Tin tức với 27%
  - Tiếp theo sau là 2 mảng: Sức khoẻ - Đời sống (24%) và Giải trí (21%)
  - Ta sẽ xem xét kỹ hơn theo từng nguồn trang một phía sau đây.

d. Thống kê category dựa theo nguồn trang

- **Desc**: Thống kê số lượng bài viết theo category theo từng nguồn trang. Để xem trang nào có xu hướng đăng bài viết theo category nào.
- **Code**:

```
%spark.pyspark
temp1 = (data_df.where(data_df.source == '24h')
            .groupBy("category")
            .agg(F.count("headline").alias("Total"))
            .orderBy("Total", ascending=False))
z.show(temp1)
```

- **Kết quả**:

24h	
Vietnam Net	
VNExpress	

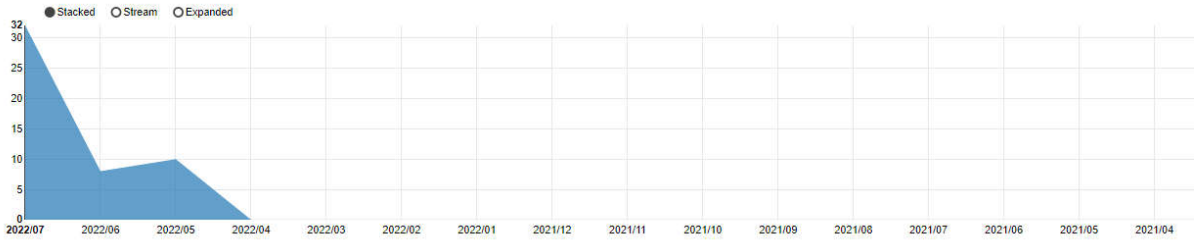
- **Nhận xét**:
  - 24h: tập trung 3 mảng chính là sức khỏe – đời sống(27%), tin tức (26%), giải trí (19%)
  - VietNamNet: Đăng chủ yếu tin tức (32%) và kinh doanh, sức khoẻ
  - VNExpress: giải trí và tin tức đều 31%

e. Thông tin về đậu mùa khỉ

- **Desc**: Kiểm tra số lượng các bài báo có nói về đậu mùa khỉ
- **Code**:

```
%spark.pyspark
t = (data_df.withColumn('DauMua', F.when(
data_df.content.contains('đậu mùa khỉ')
                                        |
data_df.content.contains('đậu mùa khỉ'),1).otherwise(0)))
 t1 = (t.groupBy('pub_month')
                    .agg(F.sum("DauMua").alias("Total"))
                    .orderBy("pub_month",
ascending=False))
 print(t1.count())
 z.show(t1.limit(24))
```

- **Kết quả**:



Hình 11. Kết quả thông tin các bài báo về dịch đậu mùa khỉ

- **Nhận xét**:
  - Bài báo về đậu mùa khỉ xuất hiện từ tháng 4/2022

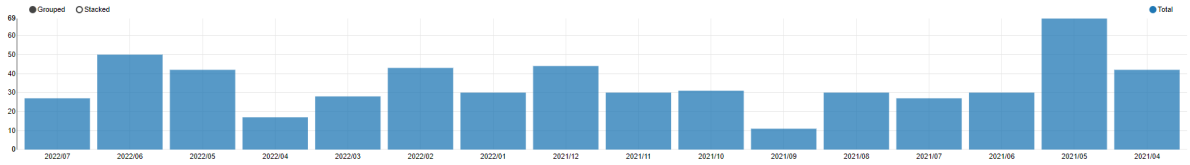
f. Thông tin về đồng tiền Bitcoin

- **Desc**: Kiểm tra số lượng các bài báo có nói về Bitcoin
- **Code**:

```
%spark.pyspark
t = (data_df.withColumn('Bitcoin', F.when(
data_df.content.contains('Bitcoin')
                                    |
```

```
data_df.content.contains('bitcoin'),1).otherwise(0)))
    t1 = (t.groupBy('pub_month')
                    .agg(F.sum("Bitcoin").alias("Total"))
                    .orderBy("pub_month", ascending=False))
    print(t1.count())
    z.show(t1.limit(16))
```

- **Kết quả**:



Hình 12. Kết quả thông tin các bài báo về Bitcoin

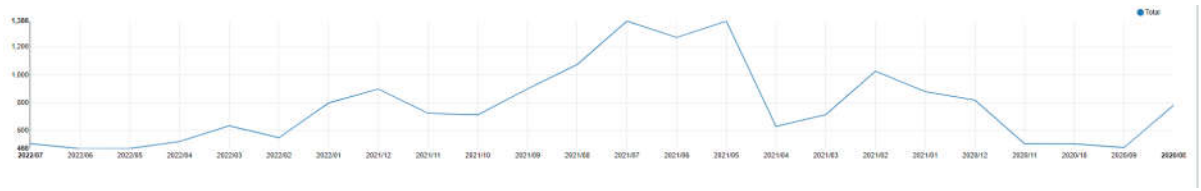
- **Nhận xét**:
  - Bitcoin đã được nhắc đến thường xuyên hơn

g. Thông tin về Covid

- **Desc**: Đếm số lượng các bài báo nói về Covid
- **Code**:

```
    %spark.pyspark
    t       =       (data_df.withColumn('Covid',       F.when(
data_df.content.contains('Covid')
                                        |
data_df.content.contains('covid'),1).otherwise(0)))
    t1 = (t.groupBy('pub_month')
                    .agg(F.sum("Covid").alias("Total"))
                    .orderBy("pub_month", ascending=False))
    print(t1.count())
    z.show(t1.limit(24))
```

- **Kết quả**:



Hình 13. Kết quả thông tin các bài báo về COVID-19

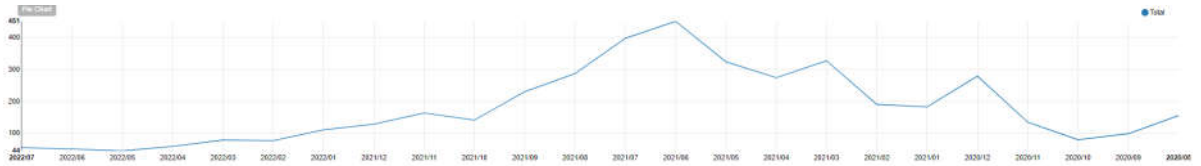
- **Nhận xét**:
  - Hiện tại thì có ít bài báo về covid hơn năm 2021.

h. Thông tin về vaccine Covid

- **Desc**: Đếm các bài báo nói về vaccine và Covid
- **Code**:

```
    %spark.pyspark
    t      =      (data_df.withColumn('Vaccine',      F.when(
(data_df.content.contains('Vaccine')
                                                |
data_df.content.contains('vaccine'))
                                               &
(data_df.content.contains('Covid')
                                                |
data_df.content.contains('covid') ),1).otherwise(0)))
    t1 = (t.groupBy('pub_month')
                    .agg(F.sum("Vaccine").alias("Total"))
                    .orderBy("pub_month", ascending=False))
    print(t1.count())
    z.show(t1.limit(24))
```

- **Kết quả**:



Hình 14. Kết quả thông tin các bài báo về vaccine COVID-19

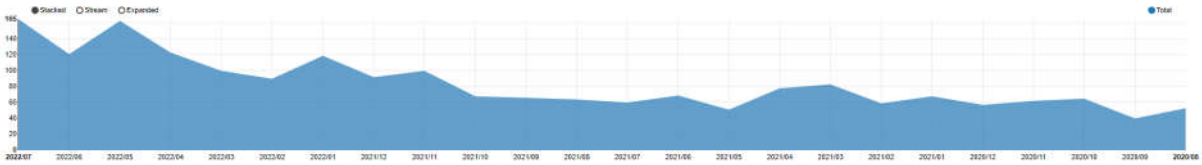
- **Nhận xét**:
  - Hiện tại có ít bài báo về vaccine covid hơn năm 2021.

i. Thông tin về những điều không tốt

- **Desc**: Thống kê các bài báo có nhắc đến: "cướp", "giết", "tai nạn"
- **Code**:

```
%spark.pyspark
t = (data_df.withColumn('Bad', F.when(
data_df.description.contains('cướp')
                                    |
data_df.description.contains('giết')
                                    |
data_df.description.contains('tai nạn') ,1).otherwise(0)))
t1 = (t.groupBy('pub_month')
                .agg(F.sum("Bad").alias("Total"))
                .orderBy("pub_month", ascending=False))
print(t1.count())
z.show(t1.limit(24))
```

- **Kết quả**:



Hình 15. Kết quả các thông tin về các bài báo phản ánh về nội dung không tốt

- **Nhận xét**:
  - Dựa trên kết quả biểu đồ trên thấy rằng những chuyện xấu lúc nào cũng sảy ra đều đặn.

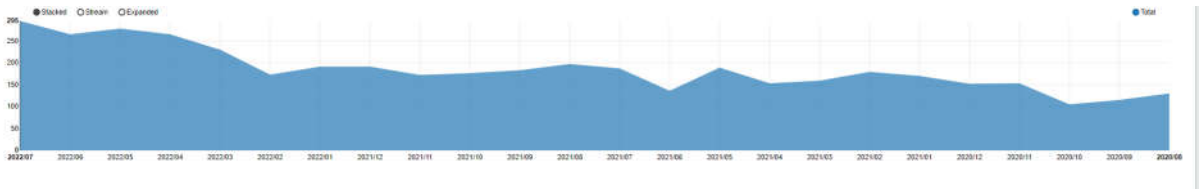
j. Thông tin về những điều tốt

- **Desc**: Thống kê các bài báo có nhắc đến: "yêu", "tình yêu", "yêu thương", "niềm vui", "hạnh phúc"
- **Code**:

```
%spark.pyspark
t       =       (data_df.withColumn('Good',       F.when(
```

```
data_df.description.contains('tình yêu')
                                        |
data_df.description.contains('yêu')
                                        |
data_df.description.contains('hạnh phúc')
                                        |
data_df.description.contains('niềm vui')
                                        |
data_df.description.contains('yêu thương') ,1).otherwise(0)))
    t1 = (t.groupBy('pub_month')
                    .agg(F.sum("Good").alias("Total"))
                    .orderBy("pub_month", ascending=False))
    print(t1.count())
    z.show(t1.limit(24))
```

- **Kết quả**:



Hình 16. Kết quả các thông tin về các bài báo phản ánh về nội dung tốt

- **Nhận xét**:
  - Số lượng bài viết tốt cũng rất nhiều.
  - Như vậy khả năng tăng là do số lượng bài viết tăng, hoặc lượng data chưa được khách quan (tức là đầy đủ) để thể hiện.