Ask Company

크롤링 차근차근 시작하기 (2/E) - 중급편

인스타그램에서 태그 검색하여 포스팅의 해시태그 및 이미지 수집하고 시각화하기

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

기본 시나리오

다음 주소의 페이지에서

- https://www.instagram.com/explore/tags/태그명/

그 태그의 전체 포스팅 개수를 알아내고

원하는 최대 개수를 크롤링할 때까지 Page Down을 수행하며 크롤링하여

각 포스팅의 주소를 획득

각 포스팅 페이지를 방문하여 해시태그를 알아내기

해시태그 빈도 시각화

유틸리티 클래스 (1/2)

```
import time
from selenium import webdriver
from bs4 import BeautifulSoup


class Browser:
    def __init__(self, driver_path='chromedriver'):
        self.driver = webdriver.Chrome(driver_path)
        self.driver.implicitly_wait(5)

    def __enter__(self):
        return self

    def __exit__(self, type, value, traceback):
        self.driver.quit()

    def visit(self, url):
        self.driver.get(url)

    def bs4_select(self, selector):
        html = self.driver.page_source
        soup = BeautifulSoup(html, 'html.parser')
        return soup.select(selector)
```

유틸리티 클래스 (2/2)

이전 페이지에 이어

```
def bs4_select_one(self, selector):
    html = self.driver.page_source
    soup = BeautifulSoup(html, 'html.parser')
    return soup.select_one(selector)

def webdriver_select_one(self, selector):
    return self.driver.find_element_by_css_selector(selector)

def scroll_page_down(self, wait=0.5):
    self.driver.execute_script('window.scrollTo(0, document.body.scrollHeight)')
    time.sleep(wait)

def is_bottom(self):
    return self.driver.execute_script('''
        const doc_height = Math.max(
            document.body.scrollHeight,
            document.body.offsetHeight,
            document.documentElement.clientHeight,
            document.documentElement.scrollHeight,
            document.documentElement.offsetHeight);

        return (window.innerHeight + window.scrollY) >= doc_height;
    ''')
```

각 포스팅의 id 알아내기

클래스 코드 (1/3)

```
import time
import re
from utils import Browser

class Instagram:
    POST_ID_PATTERN = re.compile(r'\/p\/(.+?)\/')
    TAG_BASE_URL = "https://www.instagram.com/explore/tags/{}/"

    def __init__(self, driver_path):
        self.browser = Browser(driver_path)

    def __enter__(self):
        return self

    def __exit__(self, type, value, traceback):
        self.browser.quit()

    def get_total(self):
        total_str = self.browser.webdriver_select_one('main header span').text
        matched = re.match(r'([\d,]+)', total_str)
        total = int(matched.group(1).replace(',', ''))
        return total
```

클래스 코드 (2/3)

```
def get_current_post_list(self):
    for post_tag in self.browser.bs4_select('.v1Nh3 a'):
        post_url = post_tag['href']
        matched = re.search(self.POST_ID_PATTERN, post_url)
        post_id = matched.group(1)

        img_tag = post_tag.select_one('img')
        try:
            img_url, width = img_tag['srcset'].split(',')[-1].rsplit(' ', 1)
        except KeyError:
            img_url = None

        post = {'id': post_id, 'img_url': img_url}
        yield post
```

클래스 코드 (3/3)

```
def get_post_dict(self, tag_name, max_size=100):
    post_dict = dict()

    page_url = self.TAG_BASE_URL.format(tag_name)
    self.browser.visit(page_url)

    total = self.get_total()
    limit = min(max_size, total)

    print('total: {} -> limit: {}'.format(total, limit))

    while len(post_dict) < limit:
        self.browser.scroll_page_down(0.5)
        self.browser.scroll_page_down(0.5)
        is_added = False

        for post in self.get_current_post_list():
            if post['id'] not in post_dict:
                post_dict[post['id']] = post
                is_added = True

        if not is_added:
            time.sleep(3)
            is_bottom = self.browser.is_bottom()
            if is_bottom:
                break

        print('\r{} / {}'.format(len(post_dict), limit), end='')

    return post_dict
```

수행결과

```
%%time

driver_path = './drivers/chromedriver-mac64-74.0.3729.6'

with Instagram(driver_path) as insta:
    post_dict = insta.get_post_dict('배낭여행', max_size=1000)
```

```
total: 867941 -> limit: 1000
1017 / 1000CPU times: user 2.66 s, sys: 122 ms, total: 2.78 s
Wall time: 1min 52s
```

1017개 수집하는 데에
1분 52초 소요

```
len(post_dict)
```

```
1017
```

각 포스팅의 해쉬태그 읽어오기

다수 포스팅의 해쉬태그를 비동기로 빠르게 읽어오기

pip install requests-futures

```
from bs4 import BeautifulSoup
from concurrent.futures import ThreadPoolExecutor
from requests_futures.sessions import FuturesSession

session = FuturesSession(executor=ThreadPoolExecutor(max_workers=30))

future_list = []

for post_id in post_dict:
    post_url = 'https://www.instagram.com/p/{}/'.format(post_id)
    future = session.get(post_url)
    future_list.append((post_id, future))

print('{}개의 작업을 대기 중 ...'.format(len(future_list)))

for post_id, future in future_list:
    response = future.result()
    html = response.text
    soup = BeautifulSoup(html, 'html.parser')
    hashtag_set = {tag['content'] for tag in soup.select('meta[property=instapp:hashtags]')}
    post_dict[post_id]['hashtag'] = hashtag_set

print('완료.')
```

```
1002개의 작업을 대기 중 ...
완료.
CPU times: user 14.3 s, sys: 1.3 s, total: 15.6 s
Wall time: 22.3 s
```

1002개 페이지 요청하고
해쉬태그 추출에
22초 소요

해시태그만 따로 모아서 → 빈도 세기 → pd.Series로 변환

```
import pandas as pd
from collections import Counter

hashtag_list = []

for post in post_dict.values():
    hashtag_list.extend(post['hashtag'])

hashtag_series = pd.Series(Counter(hashtag_list)).sort_values(ascending=False)
print(hashtag_series.size)
hashtag_series.head(20)
```

```
 4660

배낭여행              715
travel              311
여행                  310
여행에미치다            285
세계여행              212
여행스타그램            204
유럽여행              154
trip                121
세계일주              120
backpacking         103
traveler             97
유디니                 97
backpacker           89
일상                   82
europe               81
남미여행               80
travelgram           72
travelholic          71
유럽                   68
여행사진               68
dtype: int64
```

시각화 (Bar Plot, Word Cloud)

Bar Plot으로 시각화

Magic Command

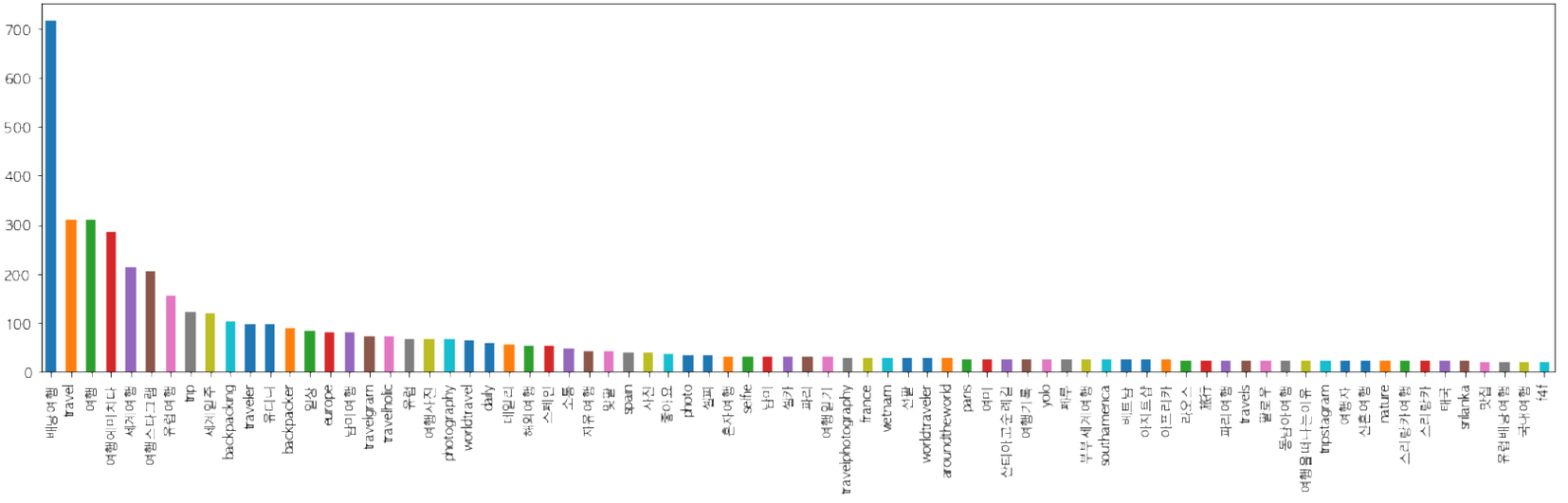
```
%matplotlib inline
# %config InlineBackend.figure_format = 'retina'  # 맥의 경우

mask = hashtag_series > 20

import matplotlib.pyplot as plt

plt.rc('font', family='Malgun Gothic')   # 맥의 경우 'AppleGothic'
plt.rc('axes', unicode_minus=False)

hashtag_series[mask].plot.bar(figsize=(20, 5))
```



Word Cloud

pip install wordcloud

```
from matplotlib import font_manager
from wordcloud import WordCloud

font_dict = { font.name: font.fname for font in font_manager.fontManager.ttflist }
font_path = font_dict['Malgun Gothic'] # 맥의 경우 'AppleGothic'

word_cloud = WordCloud(font_path=font_path).generate_from_frequencies(hashtag_series)

fig = plt.figure(figsize=(10, 10))
plt.imshow(word_cloud.to_array(), interpolation='bilinear')
plt.show()
fig.savefig('word_cloud.png')
```

인생은 짧습니다.
파이썬/장고를 쓰세요.

여러분의 파이썬/장고 페이스메이커가 되겠습니다.

- Ask Company