파이썬X웹크롤링 2강

beautifulsoup를 통한 네이버뉴스 크롤링 프로그램 만들어보기

- Beautifulsoup로 네이버뉴스 크롤링 프로그램 만들어보기
- 크롤링뿐만 아니라 데이터시각화 / 데이터분석까지 해보기

네이버뉴스 크롤링 프로그램 (1)

beautifulsoup를 통한 네이버뉴스 크롤링 프로그램 만들어보기

- 네이버뉴스홈(http://news.naver.com/main/list.nhn?mode=LSD&mid=sec&sid1=100)에 접속

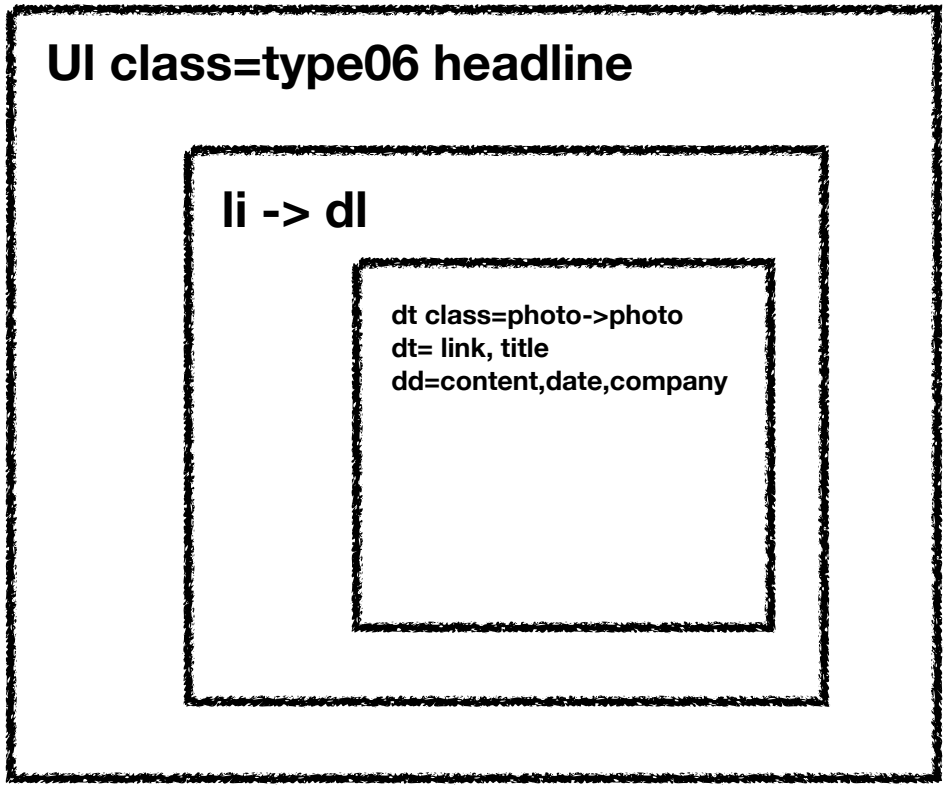
- 기사제목, 기사내용, 언론사, 날짜 크롤링 해오기

네이버 뉴스 크롤링 프로그램 1

여기 데이터들을 크롤링 하는것이 목표

- 우리가 원하는 데이터들이 어떤 태그로 구성되어있는가 확인 !

```
                    <ul class="type06_headline">


                <li>
                    <dl>

                        <dt class="photo">
                            <a href="http://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=100&oid=033&aid=0000035641" class="nclicks(fls.list)">
                             <img src="http://imgnews.naver.net/image/thumb70/033/2017/08/16/35641.jpg" alt="[시시 2분4강]고생했기"
onError="javascript:this.src='http://static.news.naver.net/image/news/2009/noimage_70x50.gif';this.style.width=70"/>
                            </a>
                        </dt>


                        <dt>
                            <a href="http://news.naver.com/main/read.nhn?mode=LSD&mid=sec&sid1=100&oid=033&aid=0000035641" class="nclicks(fls.list)">
                             [시시 2분4강]고생했기
                            </a>
                        </dt>
                        <dd>
                            공관형 선생님이 오늘 시작성어 문제를 내주셨는데, 답을 알아? 나쁜한 뜻데? 유에스 큰소리를 질러서 주변을 시고럽게 하는 짓을 …<span class="writing">주간김환</span><
class="date">2017-08-16 10:05</span>
                            </dd>
                    </dl>
                </li>
```



- 우리가 가져올 데이터들의 구성 태그

지금부터는 라이브코딩 !

```
import requests
from bs4 import BeautifulSoup

# Week1 / 1주차 참고
r=requests.get("http://news.naver.com/main/list.nhn?mode=LSD&mid=sec&sid1=100")
c=r.content
soup=BeautifulSoup(c,"html.parser")

# ul 이며 class 는 type06_headline 태그를 찾는다
all=soup.find("ul",{"class":"type06_headline"})
```

```
# 이미지 주소 가져오는 함수
def getImageSrc():
    dl=all.find_all("dl") # dl 태그를 모두 찾는다
    for item2 in dl:
        try: # 에러가 없을 시(dl태그를 찾을 시)
            # 찾은 dl태그중에서 dt 태그(class :photo) 를 찾고 다시 그 안에서 img 태그를 찾는다
            img=item2.find("dt",{"class":"photo"}).find("img")
            # 찾은 img 태그 중 속성 'src'의 내용을 가져와 출력한다.
            print(img['src'])
        except :
            # 에러 발생시(dl 태그 존재 안함)
            print("No image")
```

```
# 주소와 기사제목을 가져오는 함수
def getLinkAndTitle():
    dl = all.find_all("dl")
    for item2 in dl:
        link = item2.find("dt",{"class":""}).find("a")
        print(link['href'])
        # replace를 통해서 탭,줄띄움을 모두 공백으로 바꾸고
        # 문장들 중 첫번째 자리(스페이스)를 짤라낸다
        print(link.text.replace("\t","").replace("\n","")[1:len(link.text)+1])
```

```
# 기사내용을 가져오는 함수
def getContent ():
    dl = all.find_all("dl")
    for item2 in dl:
        try:
            content = item2.find("dd")
            print(content.text.replace("\t","").replace("\n","").split("...")[0])
            print(content.find("span",{"class":"writing"}).text)
            print(content.find("span", {"class": "date"}).text)
        except:
            print("No Content")

# 함수 호출
getContent()
```

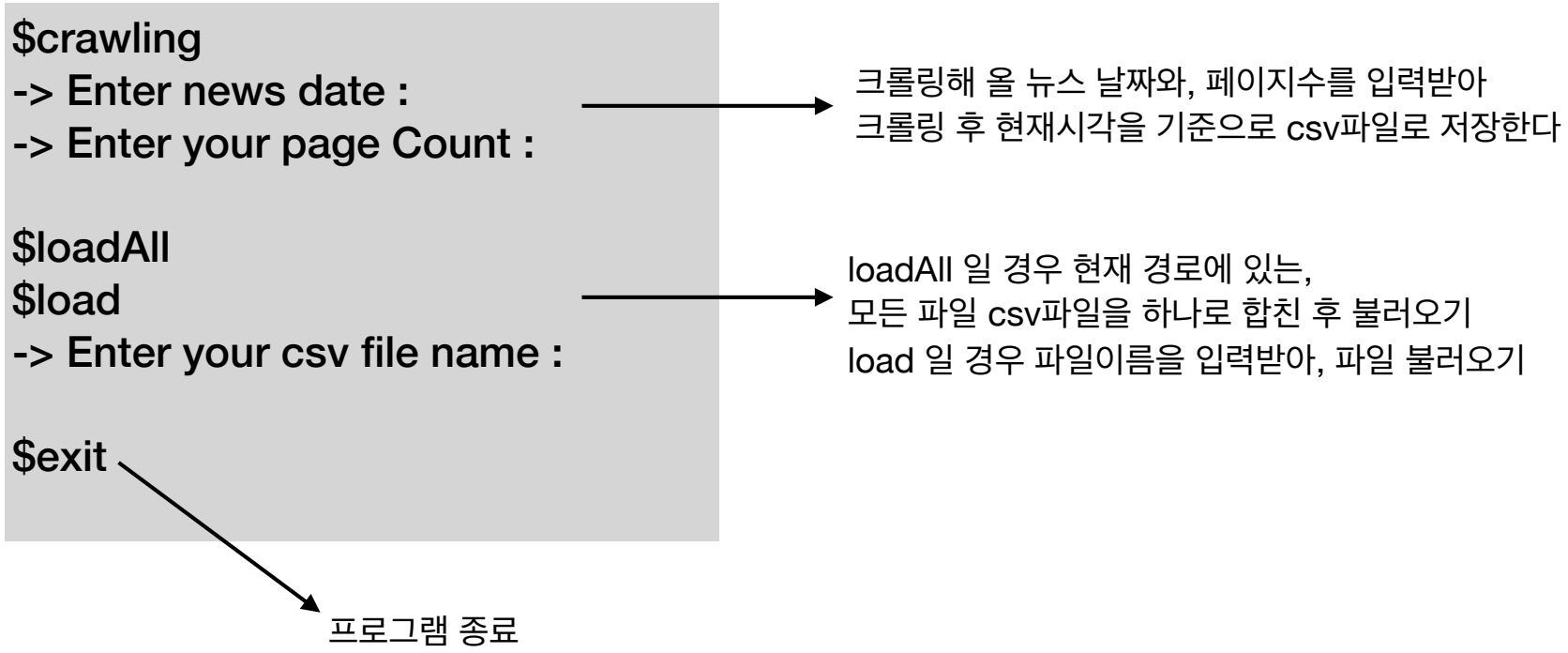
https://github.com/etilelab/webcrawling/blob/master/Week2/webcrawling0201.py

네이버뉴스 크롤링 프로그램 (2)

beautifulsoup를 통한 네이버뉴스 크롤링 프로그램 만들어보기

- 날짜, 페이지 설정으로 다수의 뉴스데이터 크롤링 해오기
- 함수 정리, 프로그램화 하기 -> 일반화(general)
- pandas, glob2 라이브러리 사용으로 csv파일로 저장하고 불러오기

네이버 뉴스 크롤링 프로그램 2 실행 예



pip를 통한 pandas, glob2 설치

- pip3 install pandas
- pip3 install glob2

지금부터는 라이브코딩 !

```
import requests,operator,pandas,glob2
from bs4 import BeautifulSoup

# 메인 세팅 함수, 사용자로부터 값을 입력받아 함수를 호출
def mainSetting():
    while (1):
        kb = input("$ ")
        if kb == "exit":
            break
        elif kb == "crawling":
            date = input("Enter news date : ")
            page = input("Enter your pageCount : ")
            crawlingData(date, page)
        elif kb == "loadAll":
            loadFile("all")
        elif kb == "load":
            fileName = input("Enter your csv file name : ")
            loadFile(fileName)
        else:
            print("command error")


mainSetting()
```

네이버 뉴스 크롤링 프로그램 2 소스코드

```
# 크롤링 함수(날짜와 페이지수를 입력받아 그날짜의 그 페이지수만큼 크롤링 해옴)
def crawlingData(date, pageCount):

    # 현재 시각을 now라는 변수에 저장
    now = datetime.now()
    l = [] # 리스트 l

    # pagecount는 1페이지부터 사용자가 입력한 페이지 수까지 됨
    for pagecount in range(1, int(pageCount)):

        # 동적으로, 사용자가 입력한 날짜와 뉴스 페이지에 접속
        r = requests.get("http://news.naver.com/main/list.nhn?mode=LSD&mid=sec&sid1=100&date=" + str(date) + "&page=" + str(pagecount))
        c = r.content
        soup = BeautifulSoup(c, "html.parser")
        all = soup.find_all("li")

        for item in all:
            for item2 in item.find_all("dl"):
                d = {} # 사전 d
                try:
                    linkTag = item2.find("dt", {"class": ""}).find("a")
                    d["LinkSrc"] = linkTag['href'] # 사전 d의 LinkSrc라는 키에 href 내용을 가져와 저장
                    d["Title"] = linkTag.text.replace("\t", "").replace("\n", "").replace(",", "").replace('"',"").replace("\r", "")[1:len(linkTag.text) + 1]
                except:
                    d["LinkSrc"] = "None"
                    d["Title"] = "None"

                try:
                    contentTag = item2.find("dd")
                    d["Content"] = \
                    contentTag.text.replace("\t", "").replace("\n", "").replace("\r", "").replace(",", "").replace('"',"").split("…")[0]
                    d["Company"] = contentTag.find("span", {"class": "writing"}).text
                    d["Date"] = contentTag.find("span", {"class": "date"}).text
                    print(d["Content"])
                except:
                    d["Content"] = "None"
                    d["Company"] = "None"
                    d["Date"] = "None"

                try:
                    imgTag = item2.find("dt", {"class": "photo"}).find("img")
                    d["imgSrc"] = imgTag["src"]
                except:
                    d["imgSrc"] = "No image"

                l.append(d) # 리스트에 사전 추가 / 한 행마다 사전에 추가

    df = pandas.DataFrame(l) # pandas 사용 l의 데이터프레임화

    # now(현재시각)을 이용해 csv 파일로 저장
    df.to_csv('%s-%s-%s-%s-%s-%s.csv' % (now.year, now.month, now.day, now.hour, now.minute, now.second),
              encoding='utf-8-sig', index=False)
    print("Success Get DataFIle and Save Data")
```

```
# loadFile 함수
def loadFile(fileName):
    # checkFileName함수를 호출, 파일이 존재하나 존재하지
않는가 확인
    outputFileName = checkFileName(fileName)

    if outputFileName is not -1:
        df = pandas.read_csv(outputFileName)
        content = df["Content"]
        title = df["Title"]
        company = df["Company"]
        print(company)

        print("csv FIle Load Success")
    else:
        print("Error csv File")
```

```
# checkFileName 함수
# 사용자가 입력한 파일명이 존재하지 않을시 -1 리턴, 존재시 파일명 리턴
# 사용자 입력값이 all이면 같은 경로의 모든 csv파일을 하나로 합치고, csv파일을 새로 만듦
# 그리고 만들어진 csv 파일을 리턴
def checkFileName(fileName):
    now = datetime.now()

    # 같은 경로에 csv 파일이 하나도 없다면 -1 리턴
    if len(glob2.glob("*.csv")) == 0:
        print("No file found in this di¬¬rectory")
        return -1
    else:
        # 사용자가 입력한 값이 all 이라면
        if fileName == "all":
            result = []
            # for 문을 돌며 모든 csv 파일을 가져와 읽은 후 result라는 리스트에 저장
            for i in glob2.glob("*.csv"):
                result.append(pandas.read_csv(i))

            # 새로 만들 파일이름 지정
            outputFileName = '%s-%s-%s-%s-%s-%s merging.csv' % (
                now.year, now.month, now.day, now.hour, now.minute, now.second)

            # 리스트에 저장한 csv파일들을 resultDf 라는 변수에 저장
            resultDf = pandas.concat(result, ignore_index=True)
            # csv 파일 생성
            resultDf.to_csv(outputFileName, encoding='utf-8-sig')
            # 새로만든 csv 파일의 이름을 리턴
            return outputFileName

        else:
            return fileName
```

https://github.com/etilelab/webcrawling/blob/master/Week2/webcrawling0202.py

네이버뉴스 크롤링 프로그램 (3)

beautifulsoup를 통한 네이버뉴스 크롤링 프로그램 만들어보기

- Csv 파일을 불러와/크롤링한 데이터 , 단어별 쪼개기, 단어 분류, 단어빈도 수 매기기
- 키워드 검색해서, 해당 키워드 갯수 파악하기
- 데이터 시각화, 워드클라우드

네이버 뉴스 크롤링 프로그램 3 실행 예

버젼2와 거의 동일하다. 하지만 데이터 시각화, 데이터분석 부분이 추가되었다.

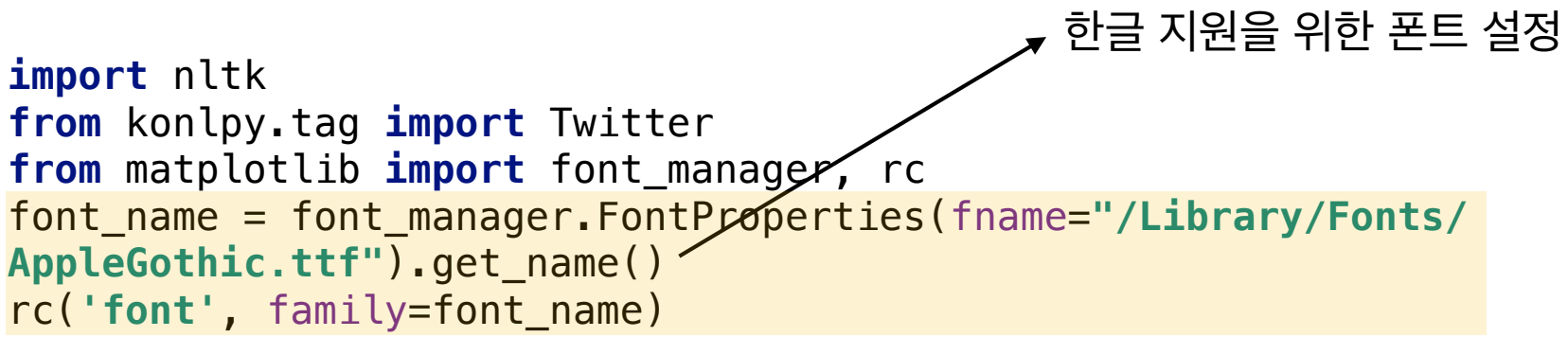
```
$crawling
-> Enter news date :
-> Enter your page Count :

$loadAll
$load
-> Enter your csv file name :

$analyze
-> Enter your csv file name :

$exit
```

기사제목, 기사내용 분석하기
단어로 쪼개어 단어빈도수, 빈도수를 시각화하기

- pip3 install wordcloud ⟶ 데이터를 시각화하기 위한 word cloud
- pip3 install konlpy ⟶ 한글 형태소/단어 분리를 위한 라이브러리
- pip3 install JPype1-py3
- pip3 install nltk ⟶ 자연어 처리 라이브러리
- pip3 install Twitter ⟶ 형태소/단어 분리에 도움을 주는 라이브러리
- Pip3 install matplotlib


중앙
한국
조선
평양
조선민주주의인민공화국
연합뉴스
중앙
조선
통신
평양
조선민주주의인민공화국

시각화 자연어 처리를 위한 라이브러리 import




```
import nltk
from konlpy.tag import Twitter
from matplotlib import font_manager, rc
font_name = font_manager.FontProperties(fname="/Library/Fonts/
AppleGothic.ttf").get_name()
rc('font', family=font_name)

from wordcloud import WordCloud
import matplotlib.pyplot as plt

t=Twitter()
```

Analyze def

각 내용들(문장)을 단어형태로 변경

가장 빈도수가 높은 단어 100개를 선택

```
def analyze(content):

    nouns=t.nouns(str(content))
    ko=nltk.Text(nouns,name="분석")

    ranking=ko.vocab().most_common(100)

    tmpData=dict(ranking)

    wordcloud=WordCloud(font_path="/Library/Fonts/
AppleGothic.ttf",relative_scaling=0.2,background_color="white",)
.generate_from_frequencies(tmpData)

    plt.figure(figsize=(16,8))
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()
```

워드크라우드 형태로 나타내기 위해
폰트와 배경화면 기타 설정

지금부터는 라이브코딩 !

```
import requests,operator,pandas,glob2
from bs4 import BeautifulSoup
from datetime import datetime

import nltk
from konlpy.tag import Twitter
from matplotlib import font_manager, rc
font_name = font_manager.FontProperties(fname="/Library/Fonts/
AppleGothic.ttf").get_name()
rc('font', family=font_name)

from wordcloud import WordCloud
import matplotlib.pyplot as plt

t=Twitter()
```

```
# 메인 세팅 함수, 사용자로부터 값을 입력받아 함수를 호출
def mainSetting():
    while (1):
        kb = input("$ ")
        if kb == "exit":
            break
        elif kb == "crawling":
            date = input("Enter news date : ")
            page = input("Enter your pageCount : ")
            crawlingData(date, page)
        elif kb == "loadAll":
            loadFile("all")
        elif kb == "load":
            fileName = input("Enter your csv file name : ")
            loadFile(fileName,0)

        elif kb=="analyze":
            fileName = input("Enter your csv file name : ")
            loadFile(fileName,1)

        else:
            print("command error")


mainSetting()
```

```
# 크롤링한 데이터 분석 함수
def analyze(content):
    # 매게변수로 전달받은 content 를 string 형태로 변경하고
    # 명사형으로 바꾸어 nouns 라는 변수에 저장
    nouns=t.nouns(str(content))
    ko=nltk.Text(nouns,name="분석")

    # 저장 후 가장 빈도수가 높은 단어 100개를 뽑아 ranking 이라는 변수에 저장
    ranking=ko.vocab().most_common(100)

    # ranking이라는 변수를 사전형으로 변경
    tmpData=dict(ranking)

    # 워드크라우드 설정
    wordcloud=WordCloud(font_path="/Library/Fonts/
AppleGothic.ttf",relative_scaling=0.2,background_color="white",).generate_from_frequencies(tmpData)

    # 만들어진 워드크라우드를 mataplotlib 를 통해 image 형태로 보여줌
    plt.figure(figsize=(16,8))
    plt.imshow(wordcloud)
    plt.axis("off")
    plt.show()
```