중간 보고서

통합 리뷰를 기반으로 한 제품 안내 어플

Vol.1

제출일	2020. 01, 24	전공	컴퓨터공학과
과목	졸업작품 프로젝트	학번	2015722084 2015722083
담당교수	이기훈	이름	한승주 김성종

목 차

I. 개요

1. 배경 및 필요성

- 서로 다른 플랫폼과 웹사이트는 사용하는 연령대나, 나이 대 등이 달라 제품의 정보를 단 하나의 사이트만 보고는 신뢰성 있는 정보를 통한 구매가 불가능하다.

- 실제 국내 페이스북 유저 비율은 남성이 여성 대비 14% 많고, 인스타그램은 여성이 남성 대비 4% 많은 비율을 가지고 있다. 또한 페이스북은 연령대가 고른 반면, 인스타그램은 20~30대 비율이 상대적으로 높다.

- 페이스북, 인스타그램, 트위터와 같은 소셜미디어에 자주 노출되는 광고와 함께 실제 사용자가 올리는 리뷰, 그리고 제품 구매처인 쇼핑몰(G 마켓, 옥션 등), 제품 가격 비교 사이트(다나와, 네이버 비교 쇼핑 등)은 여러 존재하지만, 이렇게 많은 웹사이트와 플랫폼에 퍼져 있는 사용자의 구매 후기를 통합적으로 파악할 수 있는 곳은 없다.

2. 목적

- **정보의 신뢰성**

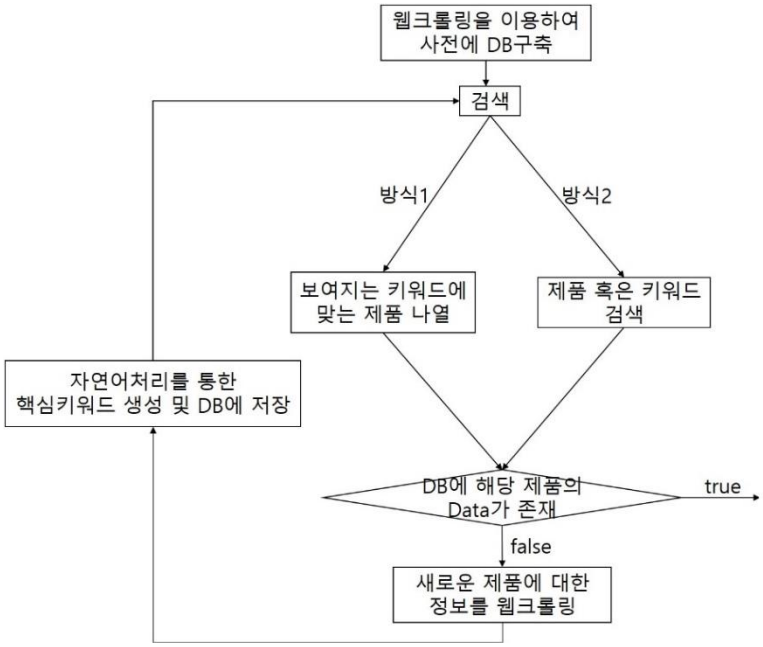
  여러 플랫폼에 퍼져 있는 제품의 리뷰를 중요 키워드를 중심으로 데이터베이스를 구축하면 이렇게 만들어진 데이터베이스를 통해 키워드만으로 사용자가 원하는 조건의 제품을 소개하고 찾아볼 수 있다. 긍정적 리뷰와 부정적 리뷰를 다 같이 한번에 보여줌으로써 제품의 장점뿐만 아니라 단점을 파악하고 같은 제품군을 비교 구매하고 싶어 하는 사용자에게 편의성을 제공할 수 있다.

- **접근의 용이성**

  이런 결과를 어플리케이션(혹은 웹사이트)으로 추가적인 구현을 하면 사용자가 언제 어디서든 쉽게 접근하고 사용할 수 있다.

3. 설계내용

ㄱ. Flow Chart



ㄴ. 개념 설계

1) 플랫폼의 웹사이트 코드 분석 및 크롤링(Beautiful soup)

- DB 를 구축하기 위해 해당 웹사이트 접속
- 제품 정보와 사용자 및 구매자의 리뷰가 담긴 웹사이트 코드 분석
- BS4를 이용해 페이지 데이터 호출
- 제품의 기본 정보(이름, 가격, 리뷰 등) tag 를 찾아 추출
- 각 사이트와 페이지별로 링크를 재귀적으로 검색하여 데이터를 추출

2) 크롤링된 정보를 이용한 DB 구축(Mysql)

- MySQL 서버에 접속하여 데이터베이스 생성
- cursor 를 추출하여 execute 메서드로 SQL 을 실행, 테이블 생성
- Execute 메서드에 데이터를 계속 확장

3) Kkma, Hannanum 을 이용한 KoNLP(키워드 생성)

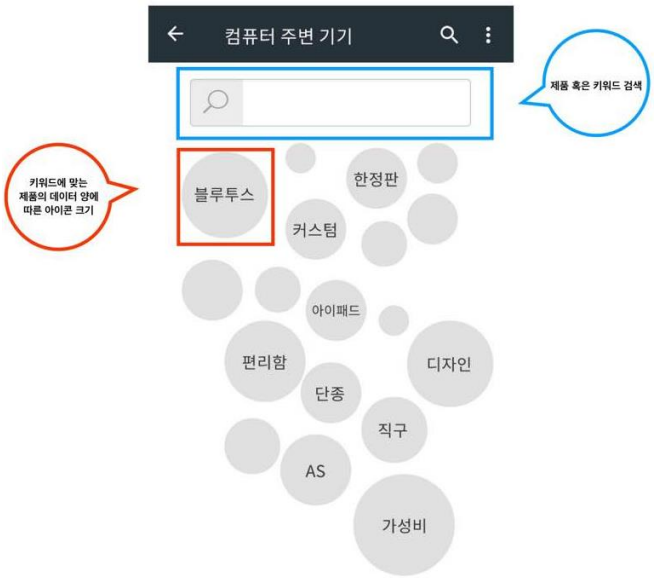
- Kkma 나 Hannanum 모듈을 이용하여, 해당 모듈에 맞추어 입력된 문자열에서 키워드로 표현할 품사 추출

- 가장 빈도수가 높은 단어(키워드로 설정할 단어)를 DB 에 저장

4) Android UI 제작

- 자신이 구매하고자 하는 제품의 리뷰를 보기 위한 제품의 검색창과 위에서 나타낸 키워드 중 전체 제품에서 가장 많은 비중을 차지하는 몇 개의 키워드를 다음과 같이 제품 검색창 아래에 사용자가 보기 편하도록 UI 로 구현

- 검색 시, 제품 검색 및 키워드(조건)를 검색할 수 있게 구현



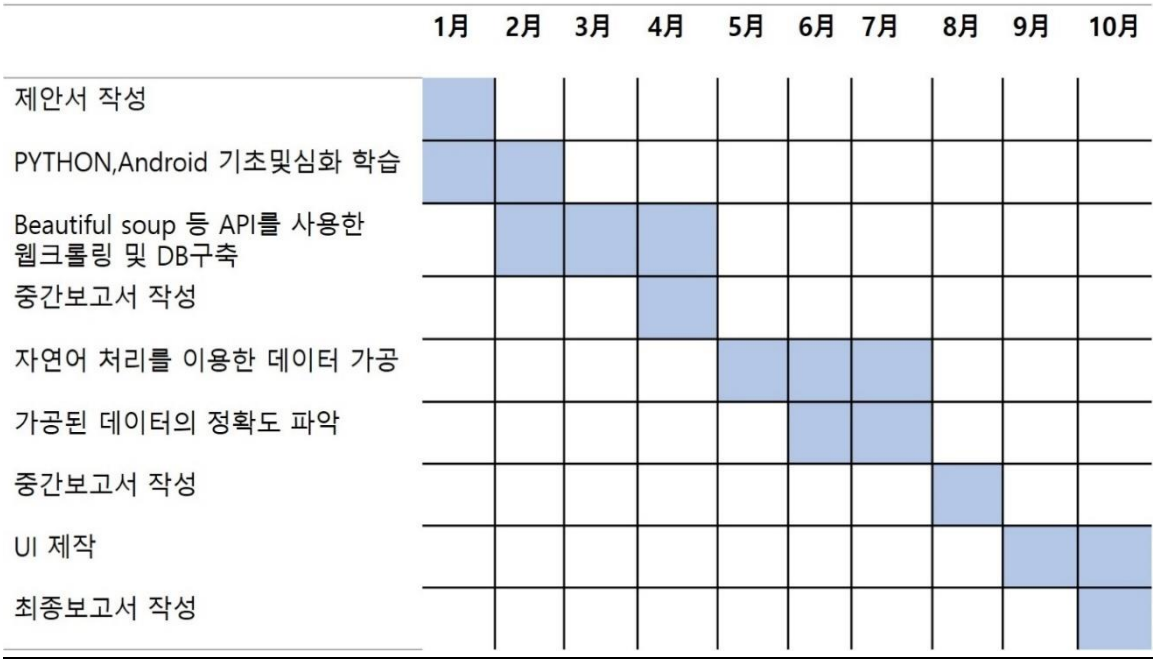
- 제품 검색 시, 제품의 가격 정보와 리뷰의 중요 키워드를 노출

- 실제 리뷰를 사이트 별로 노출

- 어플 자체 리뷰를 남겨 사용자가 어플을 더 다양하게 사용하도록 유도

II. 과제 수행

1. 수행 일정



2. 웹 크롤링

```python
def get_page_products(url):
    driver = webdriver.Chrome("C:/chromedriver.exe")
    driver.get(url)

    bs_obj = bs4.BeautifulSoup(driver.page_source, "html.parser")

    ul = bs_obj.find("ul", {"class": "prdList grid4"})
    boxes = ul.findAll("div", {"class": "box"})

    product_info_list = [get_product_info(box) for box in boxes]
    return product_info_list
```

```python
def get_product_info(box):
    p_tag = box.find("p", {"class": "name"})
    spans_name = p_tag.find("span")
    f_ul = box.find("ul")
    spans_price = f_ul.findAll("span")

    name = spans_name.text
    price = spans_price[1].text

    a_tag=box.find("a")
    link=a_tag["href"]

    return {"name": name, "price": price, "link":link}
```

Request와 selenium중 selenium을 사용한 방식이다. 페이지 하나만 사용했을 때는 request를 이용해도 문제가 없었으나, 다수의 페이지를 한번에 스크래핑 하려다보니 find 함수에서 none 값을 return하여 selenium을 사용하였다. 다만 추후에, parsing을 다른 것을 사용하면 되지 않을까 생각하여 request방식을 사용해보고자 한다.

페이지별로 각각의 상품의 이름, 가격, 하이퍼링크 주소를 각각 딕셔너리 형식으로 출력하였고, 다음은 그 결과이다.

```
==================================
1 페이지 총 상품 수 : 48
[{'name': 'Isntree Hyaluronic Acid Toner 200ml (Renewal)',
'price': 'USD 19.00', 'link': '/product/detail.html?
product_no=22560&cate_no=1019&display_group=1'},
{'name': 'ETUDE HOUSE Soon Jung pH 5.5 Relief Toner
180ml', 'price': 'USD 18.57', 'link': '/product/detail.html?
product_no=11647&cate_no=1019&display_group=1'},
{'name': 'SON&PARK BEAUTY WATER 340ml', 'price': 'USD
29.40', 'link': '/product/detail.html?
product_no=10953&cate_no=1019&display_group=1'},
{'name': 'SOME BY MI AHA BHA PHA 30 Days Miracle
Toner 150ml', 'price': 'USD 22.20', 'link':
'/product/detail.html?
product_no=16241&cate_no=1019&display_group=1'},
{'name': 'COSRX Light Fit Real Water Toner to Cream
130ml', 'price': 'USD 21.00', 'link': '/product/detail.html?
product_no=17525&cate_no=1019&display_group=1'},
'USD 16.00', 'link': '/product/detail.html?
product_no=20039&cate_no=1019&display_group=1'}]

5 페이지 총 상품 수 : 48
[{'name': 'MISSHA Super Aqua Ultra Hyalron Skin Essence
200ml', 'price': 'USD 22.66', 'link': '/product/detail.html?
product_no=19894&cate_no=1019&display_group=1'},
{'name': 'JUICYFUL Daily Vitalizer Vitamin Peeling Tissue
22ea', 'price': 'USD 12.00', 'link': '/product/detail.html?
product_no=19755&cate_no=1019&display_group=1'},
{'name': 'JUICYFUL Daily Moisture Matcha Peeling Tissue
22ea', 'price': 'USD 12.00', 'link': '/product/detail.html?
product_no=19754&cate_no=1019&display_group=1'},
{'name': 'JUICYFUL Daily Nourishing Black Bee Propolis
Peeling Tissue 22ea', 'price': 'USD 12.00', 'link':
'/product/detail.html?
product_no=19753&cate_no=1019&display_group=1'},
```

```python
import bs4
import requests

url = "https://movie.naver.com/movie/bi/mi/point.nhn?code=162249"
res = requests.get(url)
obj = bs4.BeautifulSoup(res.text, "html.parser")
main_content = obj.find("div",{"class":"score_result"})
lis=main_content.findAll("li")

count=0
for li in lis:
    score = li.find("em").text
    reple = li.find("p").text

    print(count+1,". 평점 : ",score,"\n","리플 : ",reple)
    count+=1
```

리뷰를 스크래핑하는 방식 또한 위와 유사하다. Find와 findAll함수를 사용하여 해당 위치의 정보를 가져오는 방식이다. 다만, 지금 url에서 링크를 얻고 연속적으로 링크안의 정보를 가져오는 것을 할 수 있다는 것을 확인하여 이를 해보고자 하며, 이번에는 영화정보의 리뷰를 긁어오는 실습을 해보았다.

```
"C:\Users\KIM SEONGJONG\PycharmProjects\untitled\venv\Scripts\pyt
1. 평점 : 4
리플 : 철거용역 대행전문
2. 평점 : 5
리플 : 드웨인 존슨과 함께라면 어디나 놀이동산
3. 평점 : 6
리플 : <쥬라기 월드>의 영향이 엿보이는, 인간과 고릴라의 브로맨스
4. 평점 : 4
리플 : 아무 '괴수', 아무 '파괴' 대잔치
```

그 외에도 뉴스기사나 블로그 글 제목, 환율정보 등 여러 사이트에서 각기 다른 타입의 데이터를 스크래핑하는 작업들을 해보았는데, html을 구성하는 요소가 다르기 때문에 어느 부분을 검색해야 하는지의 차이일 뿐 하는 방식은 계속해서 유사하다는 점을 확인하였다.

3. DB 구축

(진행도가 있을 때 추가 작성예정)

4. 단어 사용 빈도 추출

(진행도가 있을 때 추가 작성예정)

5. UI 제작

(진행도가 있을 때 추가 작성예정)

III. 과제 평가

1. 개선방안

- Request를 사용했을 때보다 selenium을 사용했을 때 동작에 있어 시간이 다소 오래걸리는 것을 확인했는데, html을 파싱하는데에 있어서 다소 문제가 있어서 selenium을 사용했던 것인데 이 파싱하는 부분을 바꿔서 코드를 수정해보고자 한다. 또한, 딕셔너리로 return을 받을 때 나중에 데이터를 사용할때는 반점과 괄호로 구분이 되어서 크게 상관은 없지만, 직접적인 view에 있어서는 다소 불편함이 있어서 중간중간 '₩n'가 삽입이 가능한지 확인해 보고자 한다.
- 이번 같은 경우에는 상품정보는 외국 몰에서, 리뷰는 영화로 대체를 했는데, 국내 쇼핑몰의 경우 오른쪽클릭이 안되어서 html소스를 못보는줄 알았는데, F12를 사용해서 할 수 있는 것을 확인하고 추가로 해보고자 한다. 특히, 이 경우 위에서처럼 제품정보를 얻으면서 링크를 구할 수 있고, 그 링크로부터 리뷰를 뽑아내는 것까지 해보고자 한다.
- 자연어처리 konlpy설치하는데에 있어서 pip upgrade 등의 시작부터 문제가 있는 듯하여 해결하고자 한다..

2. 기대효과

ㄱ. 기업적 측면

- 제품에 대한 긍정적인 리뷰를 많이 보이는 제품의 경우, 구매자로 하여금 다양한 웹사이트의 공통적인 리뷰를 확인가능함으로써 리뷰의 신뢰성이 높아져 구매력이 증가할 것이다.

- 해당 제품의 실제 이용고객이 제품을 사용하고 실제로 좋은 제품이라면 회사의 신뢰성이 증가하여 차기 출시될 제품의 신규 구매력도 증가할 것이다.

ㄴ. 사용자 측면

- 제품에 대한 리뷰를 알아보기 위해 번거롭게 여러 플랫폼을 일일이 찾아 검색하지 않아도 한번에 파악할 수 있다.

- 리뷰에서 자주 언급된 단어를 통해 중요 키워드를 산출해내기 때문에 신뢰가 가는 제품인지 쉽게 파악할 수 있다.

- 비슷한 조건의 다른 좋은 선택지가 없는지 한 눈에 알아볼 수 있다.

참고문헌

- 파이썬을 활용한 클로러 개발과 스크레이핑 입문, 2019, 카토 카츠야, 요코야마 유우키, 위키북스
- 파이썬 데이터 수집 자동화 한방에 끝내기 한입에 웹크롤링, 2018, 김경록, 서영덕, 비제이퍼블릭
- 파이썬을 이용한 웹크롤링과 스크레이핑, 2018, 카토 코타, 위키북스
- 파이썬을 이용한 머신러닝, 딥러닝 실전 개발 입문, 2019, 쿠지라 히코우즈쿠에, 위키북스
- Web Scraping with Python, 2019, 라이언미첼, 한빛미디어
- https://www.crummy.com/software/BeautifulSoup/bs4/doc.ko/