코사인 유사도 분석을 이용한 최저가 매칭 서비스

[1)]유송은, [1)]강병오, [1)]김지민, [1)]이강혁, [2)]이민우, [*,1)]고석주
[1)]경북대학교, [2)]㈜ 아이티스코
songeuny98@gmail.com, rkdquddh1208@naver.com, meaning2415@gmail.com,
rkdgur5381@gmail.com, itsco@itsco.co.kr, *sjkoh@knu.ac.kr

The Lowest Price Matching Service Using Cosine Similarity Analysis

[1)]Songeun Yoo, [1)]Byungoh Kang, [1)]Jimin Kim, [1)]Ganghyeok Lee, [2)]Minwoo Lee, [*,1)]Seokju Koh
[1)]KyungPook University, [2)]ITSCO

요 약

최근 온라인 쇼핑 시장이 커지면서 소비자들은 다양한 물건을 온라인에서 쉽게 접근하고 구매할 수 있게 되었다. 이와 함께 인터파크의 '톡집사', 네이버 쇼핑 등에서는 다양한 쇼핑몰의 가격 정보를 모아서 소비자들이 합리적인 가격에 상품을 구매할 수 있도록 도와주고 있다.

이에 본 논문에서는 이러한 가격 비교 시스템을 활용하여 판매자들을 대상으로 서비스하는 시스템을 제안한다. 문서 유사도를 비교하기 위하여 쓰이던 코사인 유사도 분석 기법을 쇼핑몰 상품명 분석에 이용할 수 있도록 한다. 실제 상품명 정보를 이용해 코사인 유사도 분석을 실행하고 코사인 유사도 분석 결괏값으로 관련성이 낮은 상품을 배제한다. 나머지 상품의 정보를 바탕으로 최저가 분석을 수행하여 적정 판매가격을 추출하여 제시한다. 따라서 제안하는 방식을 적용하여 상품 분석을 시행하면 비슷한 범주에 있는 상품들을 추출한 뒤 최적의 가격을 제시할 수 있을 것이다.

1. 서론

현재 소비자들이 다양한 물건을 온라인에서 쉽게 접근하고 구매할 수 있는 인터넷 쇼핑몰의 수가 늘어나고 있다. 이와 함께 인터파크의 '톡집사', 네이버 쇼핑 등에서는 다양한 쇼핑몰의 가격정보를 모아서 소비자들이 합리적인 가격에 상품을 구매할 수 있도록 도와주고 있다. 이렇게 소비자에게 제공되고 있는 서비스를 판매자에게 제공한다면 어떨까? 보통 상품은 공장에서 가져올 때의 비용에 따라서 판매자가 결정하는 가격이 달라진다. 따라서 소비자가 구매하는 시장에서는 같은 상품이라도 가격이 다양할 수 있다. 본 논문에서는 이러한 정보를 모아서 판매자에게 상품의 가격 추이와 최저가를 보여주는 것을 목표로 한다.

상품별로 가격을 수집하기 위하여 사용되는 기술은 크롤링과 문장유사도 분석이다. 가격을 수집하기 위해서는 인터넷상에 있는 상품들에 대한 상품명, 모델명, 가격 등의 정보가 필요하다. 상품의 모델명을 기준으로 하여 크롤링 한 결과 중 같은 모델명이지만 전혀 다른 상품이 함께 검색되기도 한다. 이러한 결과들을 제외하고 최저가 매칭을 목표로 하는 상품에 대해서만 가격 분석을 해야 더욱 정확한 결과를 낼 수 있으므로 유사도 분석을 수행하였다. 이때, 유사도 분석은 형태소 분석과 코사인 유사도를 통해 수행하였는데 코사인 유사도를 통해서 각 상품명에 대한 유사도를 비교할 수 있지만, 이 방식은 위치와 띄어쓰기 때문에 영향을 많이 받기 때문에 형태소 분석을 통해 이를 제어해 주었다.

가격 최저가 시스템을 가시적으로 보여주기 위해 산출물의 형태는 웹으로 선정하였다. 웹 서버는 Flask 로 구성하며 크롤링한 데이터는 DB 에 저장하여 사용하게 된다. DB 상에서 관리되고 있는 상품들을 테이블 형식으로 보여주고 선택에 따라 상품의 최저가를 업데이트할 수 있도록 한다. 최저가 업데이트는 크롤링과 문장유사도 분석을 수행한 결괏값으로 제공된 상품들의 리스트에서 가격을 비교하여 최저가를 찾아내는 방식으로 수행된다. 최저가는 DB 에 업데이트되며 이를 가공하여 차트 형식으로 출력함으로써 유의미한 결괏값을 내도록 하였다.

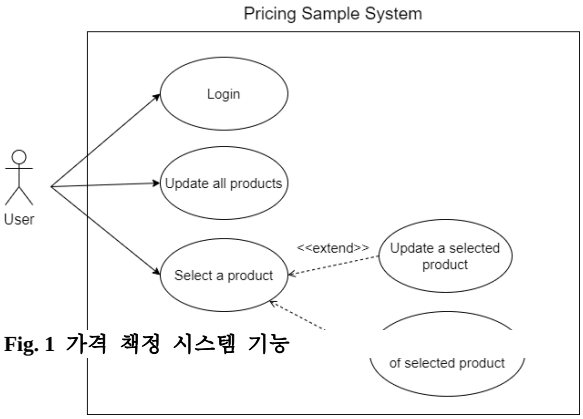


Fig. 1 가격 책정 시스템 기능

2. 관련 연구

본 논문에서는 모델명을 기준으로 한 상품명 데이터를 해당 모델명과 일치하는 데이터만 추출하는 것이 주목표다. 유사 상품 추출을 통해 현재 '아이티스코'에서 서비스 중인 '09 미플'(이하 '미플') 데이터베이스와 검색을 통해 얻은 데이터를 코사인 유사도를 통해 비교한다.

코사인 유사도 기반 비교를 위해 문서-단어 행렬을 작성 후 TF-IDF 비교를 한다. 해당 테이블을 기준으로 문장들을 벡터화한 후 기준 문장에 대한 코사인 유사도 값들을 구한다[1]

문서 단어 행렬(Document-Term Matrix, DTM)이란 다수의 문서에서 등장하는 각 단어의 빈도를 행렬로 표현한 것이다. 만약 가지고 있는 전체 문서가 방대한 데이터라면 문서 벡터의 차원은 수백만의 차원을 가질 수도 있다. 또한, 많은 문서 벡터가 대부분의 값이 0 을 가질 수도 있다. 각 문장에는 중요한 단어와 불필요한 단어들이 혼재되어 있으므로 불용어는 의미를 거의 갖지 못한다.

위와 같은 문제를 해결하기 위해 TF-IDF 가중치를 준다. TF-IDF(Term Frequency-Inverse Document Frequency)는 단어의 빈도와 역 문서 빈도(문서의 빈도에 특정식을 취함)를 사용하여 DTM 내의 각 단어마다 중요한 정도를 가중치로 준다. DTM 을 만든 후, TF-IDF 가중치를 부여한다. TF-IDF 는 주로 문서의 유사도를 구하는 작업, 검색 시스템에서 검색 결과의 중요도를 정하는 작업, 문서 내에서 특정 단어의 중요도를 구하는 작업 등에 쓰일 수 있다.[2]

정확한 DTM 을 만들기 위해 단어를 토큰화하는 과정이 필요하다. 영어와 다르게 한국어는 교착어이기 때문에 조사를 분리해주는 과정이 필요하고 한국어는 띄어쓰기가 지켜지지 않아도 문장을 이해하는 데 문제가 없어 위의 이유로 인해 한국어는 토큰화에 어려움이 존재한다. 품사 분리와 띄어쓰기 문제를 해결하기 위해 형태소 분석기를 이용해 토큰화를 한다.[3]

3. 수행 결과

본 논문에서는 일정 시각마다 자동으로 업데이트하는 기능과, 사용자가 수동으로 요청하여 확인 및 업데이트하는 기능 두 가지가 존재한다. 사용자는 기존의 미플 시스템에서 사용자가 등록한 전체 제품, 혹은 단일 제품을 업데이트하는 기능과 각 제품에 대한 최저가 분석을 보여주는 기능을 사용할 수 있다

사용자가 전체, 혹은 한 제품에 대한 가격 정보 업데이트를 요청할 경우 쇼핑몰 사이트에서 데이터를 가져온다. 가져온 데이터를 기반하여 최저가를 찾아 사용자의 화면에 표시한다. 단일 제품의 가격 분포도가 차트로 변환되어 사용자에게 보인다.

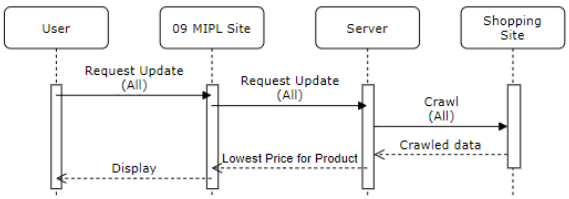


Fig. 2 전체 상품 업데이트

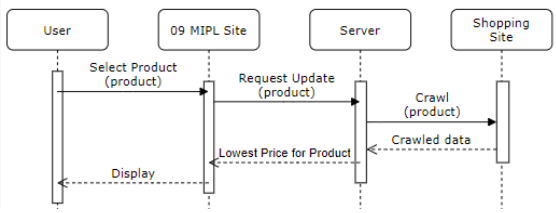


Fig. 3 단일 상품 업데이트

사용자가 특정 제품에 대한 데이터를 요청할 경우, 데이터베이스에 저장해둔 해당 제품의 날짜별 최저가 데이터를 가져와 꺾은선 그래프로 사용자에게 보여준다.

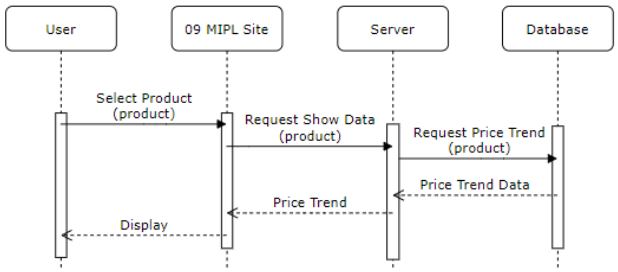


Fig. 4 상품 최저가 추이 표기

3-1. 크롤링

본 논문에서는 크롤링을 위해 파이썬 셀레니움 라이브러리를 이용한다. 자바스크립트를 이용해 동적으로 생성되는 사이트의 데이터를 사용하는 데 유용하므로 상품 정보를 크롤링하기 위한 사이트의 특성과 부합해 사용하게 되었다.

페이지 로딩 이후 생성되는 소스를 읽어 파이썬 BS4 라이브러리로 알맞은 태그를 가져와 파싱한다. 저장할 데이터는 상품명, 가격, 배송비, 판매자 링크이다. 해당 데이터들을 서버와 연동해서 실제로 사용할 수 있도록 한다.

3-2. 문장유사도 검사

크롤링으로 수집한 자료에 모델명은 같지만, 상품 유형이 아예 다른 예도 있었다. 이런 자료가 있으면 가격을 분석할 때 악영향이 있을 것으로 판단하고 문장유사도 검사를 통해 제거하기로 하였다.

[YEEZY] 이지 크레페 로우탑 스니커즈 KM5001 037이지 크레페 로	318300
Yeezy 이지 크레페 로우탑 스니커즈 KM5001 037 KM50010..(18S	299500
[마리오몰][이지]이지 크레페 로우탑 스니커즈 KM5001 037 KM50	290520
Yeezy 이지 크레페 로우탑 스니커즈 KM5001 037 KM50010..(18S	290510
[Yeezy]이지 크레페 로우탑 스니커즈 KM5001 037	282900
이지 크레페 로우탑 스니커즈 KM5001 037 KM5001037 (18SSKM	257480
이지 크레페 로우탑 스니커즈 KM5001 037 (18SSKM5001037)	213400
[탠디]BY미셸 남성 카멜 페니로퍼 정장구두 ML416T037KM	185000
[탠디]BY미셸 남성 카멜 페니로퍼 정장구두 ML416T037KM	185000
[탠디]BY미셸 남성 카멜 페니로퍼 정장구두 ML416TT037KM	185000
[탠디]BY미셸 남성 카멜 페니로퍼 정장구두 ML416TT037KM	185000
[미즈노 골프] MIZUNO GOLF 골프 슈즈 GENEM002 45KM037 3E	157800
베스토 디지털 워킹메타 B-DWM160D 100Km 쌍발타입.	89260
베스토 381-1212 디지털워킹메타 B-DWM160D 10000km-15.9cm	82710

Fig. 5 모델명은 같지만, 상품 유형이 다른 경우

크롤링으로 받아오는 데이터는 리스트이므로 먼저 데이터 프레임으로 변환한다. 그리고 데이터 프레임에 문장 유사도 검사의 기준이 될 상품명을 데이터베이스에서 받아서 입력하고 나머지 정보는 '-1'을 저장해서 다른 자료와 차별성이 있도록 한다.

형태소 분석기를 사용하여 상품명의 특수문자를 제거하고 완성된 데이터 프레임의 새로운 열에 추가하는 과정을 거친다. 형태소 분석기는 Konlpy 한글 형태소 분석기 모듈에 있는 클래스 중 테스트를 거친 결과 Mecab 을 이용해 가장 정확도가 높은 Okt 와 비슷한 정확도를 확보하면서 처리 속도를 향상시킬 수 있었다.

```
from konlpy.tag import Mecab

mecab = Mecab(dicpath="C:₩₩mecab₩₩mecab-ko-dic")

print(mecab.morphs(u'공부를하면할수록모르는게많다는것을알게됩니다.'))
print(mecab.morphs(u'공부를 하면할수록 모르는게 많다는 것을 알게 됩니다.'))
print(mecab.morphs(u'텐바이텐 [텐바이텐]kami et muse Comfort backless sneakers_KM17s03
print(mecab.morphs(u'텐바이텐[텐바이텐]kamietmuseComfortbacklesssneakers_KM17s037 '))

['공부', '를', '하', '면', '할수록', '모르', '는', '게', '많', '다는', '것', '을',
'알', '게', '됩니다', '.']
['공부', '를', '하', '면', '할수록', '모르', '는', '게', '많', '다는', '것', '을',
'알', '게', '됩니다', '.']
['텐', '바이텐', '[', '텐', '바이텐', ']', 'kami', 'et', 'muse', 'Comfort', 'backles
s', 'sneakers', '_', 'KM', '17', 's', '037']
['텐', '바이텐', '[', '텐', '바이텐', ']', 'k', 'a', 'm', 'i', 'e', 't', 'm', 'u',
'seComfortbacklesssneakers', '_', 'KM', '17', 's', '037']
```

Fig. 6 Mecab 형태소 분석기 예시

형태소 분석이 끝난 데이터 프레임은 코사인 유사도를 계산한다. 초기에 코사인 유사도를 계산할 때는 각 상품명의 TF-IDF 를 계산해서 모든 문장에 대해서 코사인 유사도를 계산했다. 이런 방식을 사용하면 $O(N^2)$만큼의 시간과 N^2만큼의 벡터 공간이 필요했다. 따라서 데이터 개수가 10 만 개 이상일 때부터 30GB 이상의 메모리를 요구하는 문제가 발생하였다.

이 문제는 코사인 계산을 한 행에 대해서만 함으로써 해결할 수 있었다. 실제로 필요한 코사인값은 처음 데이터 프레임에 추가한 DB 에서 가져온 이름이었기 때문에 해당 데이터의 행을 찾아서 나머지 문장과 코사인 유사도를 계산하는 방식으로 변경하였다. 이렇게 함으로써 $O(N)$으로 시간을 줄이고 필요한 공간도 N으로 줄어들 수 있었다. 실제로 테스트했을 때도 기존 방식과 비교해 더 나은 성능을 보여주었다.

```
print(cosine_nxn)

[[1.         1.         1.         ... 0.         0.         0.15626555]
 [1.         1.         1.         ... 0.         0.         0.15626555]
 [1.         1.         1.         ... 0.         0.         0.15626555]
 ...
 [0.         0.         0.         ... 1.         0.0668026  0.01387115]
 [0.         0.         0.         ... 0.0668026  1.         0.02224689]
 [0.15626555 0.15626555 0.15626555 ... 0.01387115 0.02224689 1.        ]]

print(cosine_sim)

[[0.15626555 0.15626555 0.15626555 ... 0.01387115 0.02224689 1.        ]]
```

Fig. 7 코사인 유사도 계산하는 방식을 변경한 결과

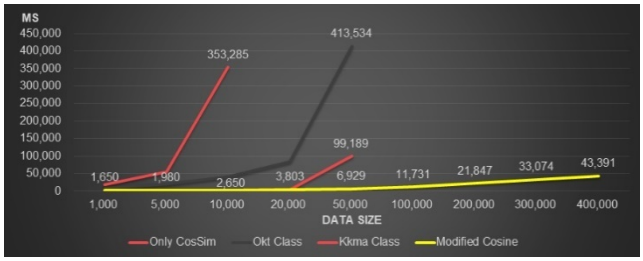


Fig. 8 데이터양에 따른 코사인 연산 시간. Modified Cosine 이 한 행에 대해서만 연산하는 방식이다.

코사인 유사도 계산까지 마친 후 코사인 유사도 값이 0.06 이하로 나온 행은 모두 삭제하는 과정을 거친다. 하위 30%를 자르거나 가격 차이가 두 배 이상 나는 곳을 기준으로 제거하는 등의 방법을 사용해보았지만 한두 개 정도 상관없는 상품이 남는 경우가 생길 때도 있고 같은 모델이라도 가격이 두세 배 차이 나는 경우가 있어서 코사인값을 기준으로 자르는 방식을 채택하였다. 대신 정확한 결과를 얻으려면 서로 다른 상품마다 다른 기준이 필요하다고 생각해서 기준을 정하는 공식을 유도해보려고 했지만, 값의 증가하는 형태가 아래의 그래프처럼 나와서 값이 갑자기 증가하는 부분을 특정하는 것이 불가능하다고 판단하고 공식 유도하기는 포기하였다. 대신 여러 상품에 대한 분석을 통해서 0.06 이 적당한 값이라고 결론을 내리고 코사인값의 기준으로 0.06 을 설정하였다.

Fig. 11 DB 다이어그램

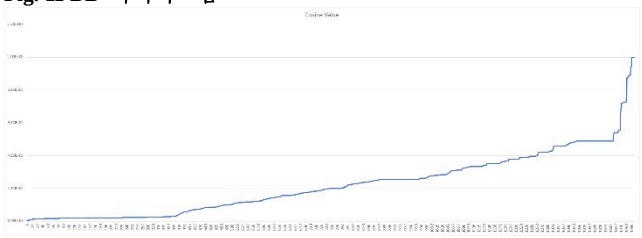


Fig. 9 코사인값 증가 추세선

코사인값에 따라 데이터 프레임에서 상품 제거까지 마친 후에는 처음에 DB 에서 가져온 행을 삭제하고 다시 리스트 자료형으로 바꿔서 반환한다.

3-3. 서버

파이썬 코드로 작성된 여러 모듈과의 호환성을 위해 파이썬 flask 웹서버를 이용했다. 실시간으로 크롤링 된 데이터를 DB, 코사인 유사도 모듈과 연동하여 웹페이지 렌더링 전에 전처리해서 출력한다. DB 와 연동하여 로그인 이후 해당 사용자가 등록 중인 상품들을 가져올 수 있도록 한다.

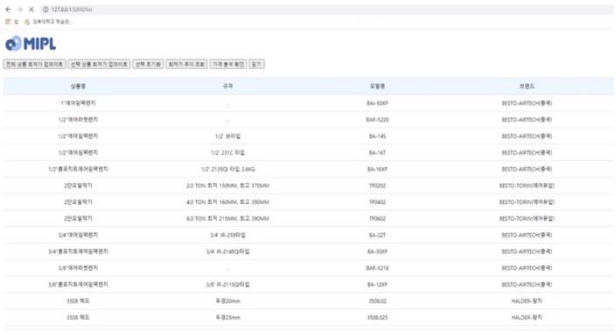
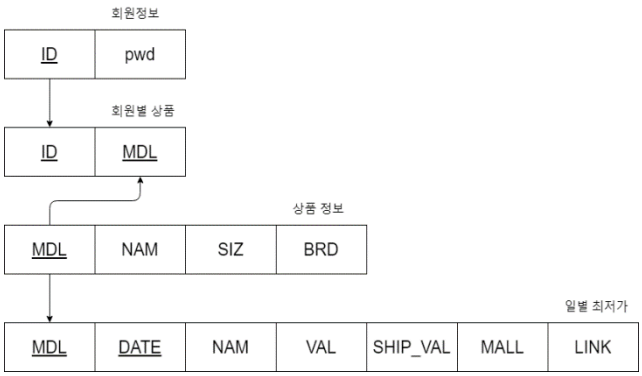


Fig. 10 렌더링 된 메인 페이지

해당 상품 관련 가격정보를 얻기 위해 실시간으로 해당 모델명에 대한 크롤링 이후 코사인 유사도를 통해 필터링 후 결과물을 차트와 표 형식으로 페이지를 렌더링한다.

최저가 업데이트 시 크롤링을 통해 날짜별로 업데이트를 진행해 DB 에 저장한다.



3-4. DB

본 프로그램에선 최저가 추이 저장과 이용자 등록상품 조회를 위해 DB 를 사용한다. MySQL 을 이용해 제작했고 서버와는 pymysql 라이브러리를 이용해 연동한다.

회원정보 테이블에서는 기본적인 ID, 비밀번호와 같은 정보를 저장한다. 테스트용으로 만들었기 때문에 이메일과 전화번호 같은 회원 개인정보 열은 추후에 추가할 수 있다. 로그인하면 가장 먼저 보이는 화면에 회원별 상품들을 출력한다.

상품정보 테이블에는 실제 공구미플 사이트에서 판매 중인 제품들의 정보를 저장한다. 회사의 지원을 통해 실제 판매 중인 공구들의 데이터를 받을 수 있었다. 이 테이블에서 모델명을 외래키로 받아와서 회원 ID 와 MDL 을 외래키로 가지는 회원별 상품 테이블을 만든다. 시연 과정에서는 모든 제품을 가지고 있는 test 계정을 만들어 시연하였다.

상품별 일별 최저가 조회를 위해 일별 최저가를 저장한다. 모델

명과 날짜를 복합키로 가지는 테이블이다. 해당 상품의 그 날 최저가를 저장한다. 최저가 조회, 갱신 시에 모델명과 날짜가 겹칠 경우 UPDATE 구문을 통해 가격을 업데이트한다.

3-5. 차트

웹페이지 상에서 데이터를 효과적으로 나타내기 위하여 구글 차트를 이용한다. 크롤링한 데이터를 구글 데이터 테이블에 저장하고, 해당 정보를 구글 차트를 이용하여 팝업 창으로 출력한다. 차트에서 제공하는 클릭 이벤트를 이용하여 해당 영역별로 이벤트를 지정하였다. 판매자가 가격 분포를 확인할 수 있도록 해당하는 분포 내의 모든 항목을 테이블로 출력한다.

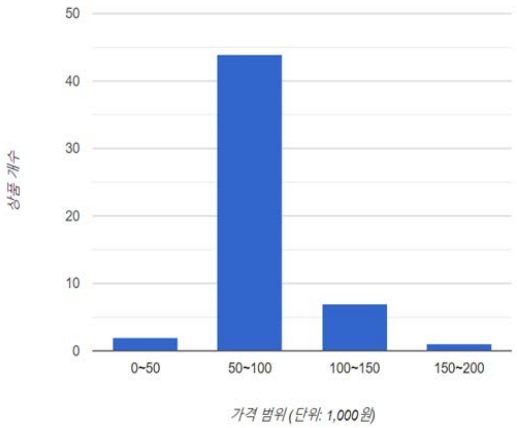


Fig. 12 실제 웹 사이트에서 구글 차트 표시

	Name	Price	Delivery Price	Shop
1	베스토 에어임팩렌치 BA-14S 쇼트타입	60,200	3,000	G마켓
2	베스토 에어 임팩 렌치 BA-14S 1/2인치 14mm 숏타입	62,700	2,500	11번가
3	베스토 에어임팩렌치 BA-14S 쇼트타입	63,000	3,000	11번가
4	[베스토] 1/2인치 에어임팩렌치(숏타입) BA-14S	63,370	3,000	11번가
5	BESTO-AIRTECH 1/2inch에어임팩렌치 BA-14S	64,330	0	옥션
6	BESTO 1 2인치에어임팩렌치 BA-14S	65,360	2,500	G마켓

Fig. 13 구글 데이터 테이블 차트 표시

또한, 날짜별 최저가를 상품별로 데이터베이스에 저장해 두어 판매자가 시장가격의 추이를 한눈에 알아볼 수 있도록 꺾은 선 그래프를 이용해 나타낸다.

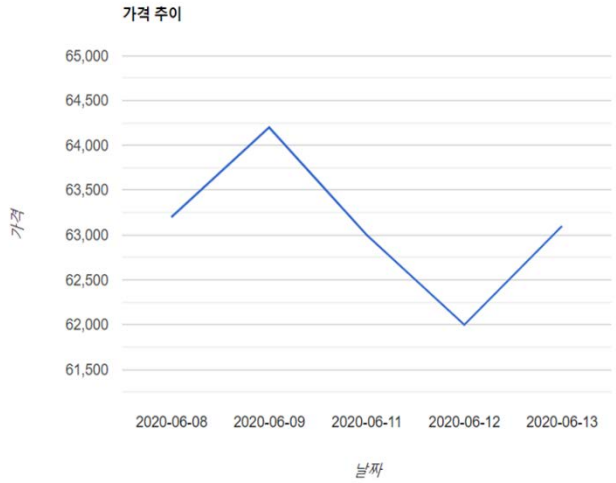


Fig. 14 날짜별 가격 변동 추이 표시

5. 기대효과

본 논문에서는 기존 상품 관리 서비스의 경쟁력 향상을 위해 문장 유사도로 상품을 분류하고 최저가를 찾아주는 서비스를 제시하였다. DB 에 상품 정보를 저장해 두고 선택한 상품명을 기준으로 형태소 분석과 코사인 유사도를 수행하여 유효한 상품 리스트를 도출했다. 상품 리스트의 가격을 분석하여 최저가를 찾아 DB에 업데이트하였고, 차트를 이용해 상품의 가격 분포를 보여주며 누적된 가격 데이터를 통해 가격의 변동 추이를 보여주었다.

본 논문에서 제시한 최저가 매칭 서비스를 통해 판매자가 판매하는 물품에 대한 처리를 자동화한다. 기존 아이티스코에서 서비스하는 '09 미플' 서비스는 제품의 내용을 09 미플 사이트에 한 번만 입력하면 다양한 쇼핑 플랫폼에 상품을 연동하는 기능을 갖고 있다. 이에 더하여 물품에 대한 등록이 끝난 뒤에는 판매자가 신경 쓰지 않아도 시장의 가격에 맞추어 자동으로 가격을 조정을 도움으로써, 완전히 자동화된 물건 관리 시스템 체계를 구축할 수 있을 것이다.

성공적으로 판매자의 편리를 위한 소프트웨어가 적용된다면 향후 판매자를 위한 타 소프트웨어의 개발 역시 기대해볼 수 있다. 본 논문에서 제시하는 프로그램의 향상 방향 역시 제시해둔 것처럼 다양하다. 동일 모델명이 아닌 동일류 제품의 가격을 살펴 가격 경쟁력을 알려주는 기능이나 가격 추이를 이용하여 추후 최저가를 예측하는 기능을 추가할 수도 있다.

Acknowledgement

"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW 중심대학사업의 연구결과로 수행되었음"(2015-0-00912)

참고문헌

[1] 김민재, 이상진. (2014). 코사인 유사도 기반의 인터넷 댓글상 이상 행위 분석 방법. 정보보호학회논문지, 24(2), 335-343.

[2] 원준, 『딥 러닝을 이용한 자연어 처리 입문. 위키독스』, 2020.(https://wikidocs.net/21690)

[3] 이유진, 김세빈, 홍현석, 김장원. (2019). 특허 문서를 위한 형태소 분석기 비교 평가. 한국정보기술학회 종합학술발표논문집, (), 264-265.