기술문서 최종 보고서	'15. 11. 04. 작성

기계 학습 기반의 한글 댓글 분석을 통한 페이스북 유해 게시물 탐지 모델 개발

참 여 인 원 : 지 한 별 ghb202@seoultech.ac.kr
송 윤 성 ys_9814@seoultech.ac.kr
서 관 영 tjrhks12@seoultech.ac.kr

KUCIS 대학정보보호동아리연합회

서울과학기술대학교 SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

목 차

Abstract

현재, 정보화시대가 급속도로 진행되면서 인터넷을 기반으로 한 소셜 네트워크 서비스(Social Network Service)나 블로그 등의 플랫폼이 일상생활 속의 필수적인 의사소통 및 정보 공유수단으로 자리 잡게 되었다. 이로 인하여 인터넷 사용자들은 네트워크상에서 실시간으로 정보를 공유하거나 의사소통을 하는 것이 가능하게 되었지만, 한편으로는 익명성이나 전파성 등의 특성을 이용하여 유해물을 유포하거나 인터넷상에서의 명예훼손 같은 문제가 심각하게 대두되고 있다.

방송통신심의위원회는 위와 같은 문제에 대처하기 위해 관련법에 따라 음란 게시물에 대한 삭제 또는 접속차단, 이용자에 대한 이용정지 해지 등의 조치를 하고 있다. 하지만 이러한 조치가 국내에 국한되어 있어 Facebook, Twitter 등의 해외기반 서비스 대해서는 즉각적인 조치를 할 수 없는 한계를 지니고 있다. 본 논문에서는 이런 한계를 극복하고 SNS상의 유해물 유포에 대한 해결책을 제시하기 위해 사용자들의 댓글 데이터를 기반으로 데이터 마이닝 기법을 활용한 유해물 탐지 모델을 개발한다.

Ⅰ. Introduction

21세기 정보화 사회는 컴퓨터나 멀티미디어, 통신 분야가 주요 매체이고 정보의 생산과 전달을 중심으로 전개되는 탈공업화 사회이다.〔1〕 정보화 사회를 주도하고 있는 인터넷이 발달하면서 사용자들은 다양한 서비스를 받고 있다.

1) 2004년 조사부터 인터넷에 모바일 인터넷을 포함시켰으며, 인터넷 이용자 정의도 '월평균 1회 이상 인터넷 이용자'에서 '최근 1개월 이내 인터넷 이용자'로 변경함

2) 2006년부터 조사대상을 만 3세 이상 인구로 확대함(2000년-2001년 : 만7세 이상 인구, 2002년-2005년 : 만 6세 이상 인구)

〔그림 1-1〕 인터넷 이용률 및 이용자수 변화 추이(%, 천명)

커뮤니티, E-mail, 블로그, 자료검색 등 인터넷 서비스가 발달하면서 국내 인터넷 사용자 수는 점진적으로 증가하였다. 〔그림 1-1〕에서 볼 수 있듯이 2004년에 31,580,000명에서 10년 후인 2014년에는 41,118,000명으로 인구의 83.6%가 인터넷을 사용하게 되었다.〔2〕

1.1 SNS상의 유해물 유포 현황

인터넷 서비스가 모바일에서도 가능해짐에 따라 매일 수많은 정보들이 공유된다. SNS에서는 광고, 동영상, 뉴스, 생활정보 등 다양한 장르의 게시물이 쏟아진다. 최근에는 일간베스트, 디시인사이드, 네이트판 등의 커뮤니티에서 불법광고, 도박, 음란물 등 유해 게시물이 늘어나고 있다. SNS에서도 역시 별도의 인증 없이 음란 게시물을 접할 수 있어 사용자들이 유해 게시물에 쉽게 노출되고 있는 상황이다.

구분	위반유형	'12년	'13년	'14년	합계
시정 요구	도박	1,000	872	1,352	3,224
	불법 식·의약품 판매	2,601	597	86	3,284
	성매매·음란	250	4,448	15,821	20,519
	권리침해	164	17	77	258
	기타 법령 위반 (문서위조, 불법 명의거래 등)	439	469	215	1,123
합계		4,454	6,403	17,551	28,408

〔표 1-1〕 SNS 시정요구 현황
(기간 : 2012. 1. 1. ~ 2014. 12. 31. /단위 : 건)

사용자들은 인터넷 게시물 중 도박, 음란물, 불법 식·의약품 판매, 권리침해 등의 유해성 게시물들에 대해 2012년부터 2014년까지 약 28,408건의 시정요구를 신청했다. 〔표 1-1〕에서 알 수 있듯이 이 중 성매매·음란 관련 게시물 시정요구가 2014년 기준 15,821건으로 가장 많다.〔3〕

1.2 문제인식 및 해결방안

현재 SNS의 수많은 음란게시물은 별도의 성인인증 없이도 열람가능하다. 또한, 전 세계적으로 게시물을 공유하는 인터넷에서는 유해 게시물 역시 쉽게 접할 수 있다. SNS 중 이용자가 가장 많은 Facebook 타임라인에서도 유해게시물을 접하기 쉽다.〔2〕 이 중 대부분은 〔그림 1-2〕와 같이 적은 텍스트를 포함하며 사진 또는 영상으로 이루어져 있다. 따라서 게시물의 텍스트 내용만으로는 음란성 판단이 어렵다.

〔그림 1-2〕 SNS에서의 음란물 유포 현황

유해성 판단을 위해 게시물의 이미지나 동영상에서 픽셀단위의 색상 비교를 통해 사람의 피부 색상이 많이 포함되는 경우를 확인하여 유해물을 판단하는 연구가 활발히 진행되고 있다. 하지만 이는 영상이 어둡고 정확하지 않으면 분석 효율성이 떨어진다. 이와 같은 경우에는 분석 모델의 성능적인 면에서 한계가 있다고 판단한다. 따라서 본 연구에서는 게시물의 내용이 아닌 게시물의 댓글을 이용한 유해물 탐지 기법을 통해 기존 유해물 탐지 시스템의 성능 향상방안을 제시한다.

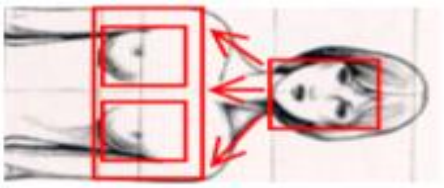
Ⅱ. Related Works

2.1 선행연구 분석

이 절에서는 본격적인 데이터 마이닝에 앞서 선행 연구 사례를 살펴본다. 유해물 탐지 기법인 하르 분류기를 통한 유해물 탐지 기법과 등급에 따른 웹 유해 문서분류 기법에 대하여 서술한다.

2.1.1 하르 분류기를 통한 유해물 탐지

하르 분류기는 다양한 표본으로 분류기를 훈련하는 데이터 마이닝의 일종이다. 주로 카메라에서 사람의 얼굴을 인식하는 기술로 사용된다. 이를 이용하여 음란물에서 인식된 인체 특정 부분의 기하학적 관계를 이용하여 음란 이미지를 탐지하는 연구가 진행되고 있다.〔4〕



〔그림 2-1〕 객체의 기하학적 관계

〔그림 2-2〕 얼굴 인식 예시

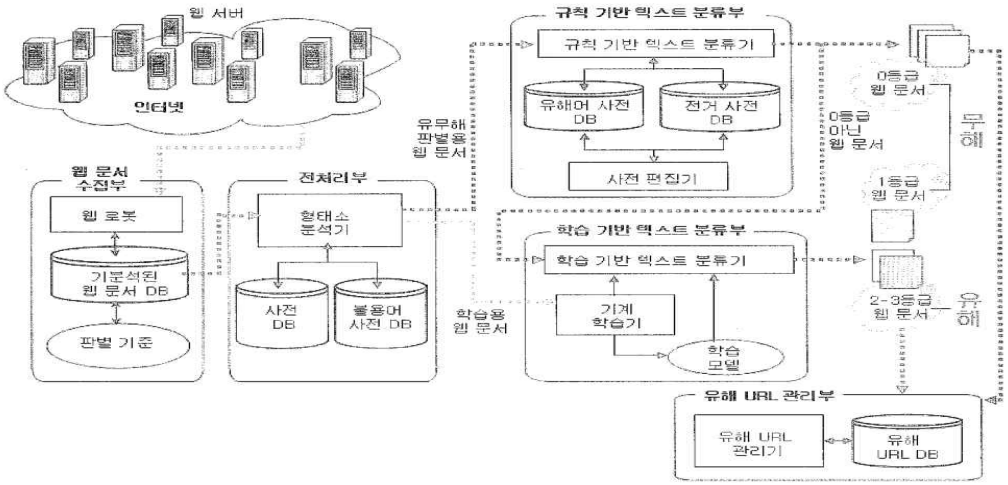
하지만 이 연구에서는 얼굴 부분이 없는 이미지나 가슴부위가 가려진 이미지 등은 인식할 수 없다는 한계가 있다. 이러한 예외적인 경우에 대해서도 추가적인 연구가 필요하다고 결론지었다. 이처럼, 모든 선정적 이미지와 동영상을 자동으로 탐색하기는 매우 어렵다. 따라서 본 연구에서는 음란게시물을 판단할 때 게시물의 미디어 파일을 판단하기보다는 해당 게시물의 댓글에서 선정적인 내용을 판별함으로써 음란게시물을 선별하는 것이 더 효과적이라 판단한다.

2.1.2 등급에 따른 웹 유해 문서분류

이 연구는 웹에서 유해정보의 내용 자체에 기반을 두어 유해 여부를 자동으로 판별할 수 있는 효율적인 유해 웹 문서 텍스트 판별 시스템을 제시한다. 등급에 따른 웹 유해 문서분류는 자칭 추출 기술을 사용한다. 이 기술은 판별에 유용하게 사용될 만한 내용어를 선택한다. 또한, 아래와 같은 기술을 이용하여 추출된 자질을 통해 텍스트를 판별한다.〔5〕

- TF(Term Frequency) 이용: 특정 용어가 출현한 문서의 수
- DT(Document Frequency) 이용: 특정 용어가 출현한 문서의 수
- MT(Mutual Information) 이용: 두 용어 중 한 용어가 출현했다는 사건이 다른 용어의 출현 여부를 예측하는 데 기여하는 정도
- IG(Information Gain) 이용: 문서 분류에 기여하는 정도 계산
- CHI(x2 statistics) 이용: 용어와 범주 간의 의존선을 측정해 용어의 중요도를 구함

텍스트 판별에서는 지지 벡터 기계(SVM: Support Vector Machine)를 적용한다. 이는 최근의 학습 기법으로 두 개의 클래스 구성 데이터들을 가장 잘 분리할 수 있는 결정면(decision surface)을 찾는 모델이다.



〔그림 2-3〕 효율적인 유해 웹 문서 판별 시스템

〔그림 2-3〕은 효율적인 유해 웹 문서 판별 시스템의 전개이다. 웹 문서 수집부, 전처리부, 규칙 기반 텍스트 분류부, 학습 기반 텍스트 분류부, 유해 URL 관리부 등 총 다섯 부분으로 구성된다. 이 중 전처리부의 핵심 기능은 형태소 분석이다. 불용어 사전을 이용한 기호 및 불용어 제거 작업을 한 후 형태소 사전을 통하여 웹 문서의 내용을 형태소 단위로 구분한다.

이 연구에서는 텍스트의 내용을 판별하여 유해문서를 분류하였다. 이와 달리 본 논문은 대부분의 유해게시물은 텍스트 없이 사진이나 영상으로만 이루어졌기 때문에 정확한 텍스트 분석은 어렵다고 판단하여 텍스트의 내용이 아닌 댓글의 내용으로 분석을 진행하였다.

2.2 한국어 정보처리 패키지

본 논문에서는 Facebook의 댓글이 한글이므로 성능 높은 한국어 형태소 분류기로 데이터 마이닝을 진행하였다. 이 절에서는 KOMORAN, MeCab, KoNLPy의 특징과 분류기 선택이유를 서술한다.

2.2.1 KOMORAN

KOMORAN은 SHINEWARE에서 개발된 한국어 형태소 분석기로 JAVA 라이브러리 형태(.jar)의 무료 오픈 소스 패키지다. 여러 어절을 하나의 품사로 분석할 수 있고 공백이 포함된 형태소 단위도 분석 가능하다. 또 외부 라이브러리와의 의존성 문제가 없으며 트리구조 기반의 형태소 사전을 이용해서 저장 공간 부담이 적다. 일반 텍스트 파일 형태로 작업이 가능하므로 가독성이 좋은 형태소 분석기이다.〔6〕

2.2.2 MeCab

MeCab은 인볼 NTT 커뮤니케이션 과학 기초 연구소 공동 연구 유닛 프로젝트를 통해 개발된 오픈 소스 형태소 분석 엔진이다. 언어 사전, 말뭉치에 의존하지 않아 범용성이 뛰어난 설계로 되어있다. 조건부 확률 주차(Conditional Random Fields)를 사용하여 정밀도가 높으며 작업속도가 빠르다. MeCab은 각종 스크립트 언어를 호환하여 다양성을 확보하기 때문에 다양한 형태소 분석에 사용된다.〔7〕

2.2.3 자연언어 처리 패키지 KoNLPy

〔그림 2-4〕 KoNLPy

KoNLPy 패키지는 기존에 C/C++, JAVA로 개발된 형태소 분석기를 Python에서 사용할 수 있도록 처리했다. 이로 인해 NLTK라는 매우 우수한 형태소 분석 도구를 Python으로 작성하여 한국어 형태소 분석이 가능하다. KoNLPy는 KOMORAN, MeCab-ko 등 많은 한국어 정보처리 패키지와 달리 특별한 장점들이 있다.〔8〕〔9〕

쉽고 간단한 사용법과 직관적인 함수명을 사용한다.
다양한 자연어 처리 기능과 말뭉치를 포괄한다.
Python패키지와 응용하여 사용 가능하다.

〔표 2-1〕 KoNLPy패키지의 장점

본 연구에서는 세 가지 분류기 중에서 KoNLPy로 한국어 형태소 분석을 하였다. KOMORAN은 JAVA 라이브러리 형태(.jar)로 제공되어 Python의 다른 패키지와 응용할 수 없지만 KoNLPy는 응용이 가능하다. 또한, KoNLPy 0.3.3 버전부터 MeCab을 Class의 형태로 사용할 수 있다. 형태소 분석기뿐만 아니라 다양한 자연어 처리 기능과 말뭉치를 포괄하기 때문에 본 연구의 목적에 부합하여 KoNLPy를 한글 형태소 분석기로 사용하였다.

2.3 기계 학습

기계 학습(Machine Learning)은 데이터 마이닝 또는 패턴 인식이라고 불리며, 어떠한 에 대해 꾸준한 경험을 통하여 현상에 대한 성능을 높이는 것이다. 크게 지도학습(Supervised Learning)과 자율학습(Unsupervised Learning)으로 분류한다.

2.3.1 지도 학습(Supervised Learning)

어떠한 입력(x)에 대해 레이블(y)을 달아놓은 데이터를 전달하면 컴퓨터가 그것을 학습하는 것이다. 입력 과정에서 사람이 개입하므로 정확도가 높은 데이터를 사용 가능하다. 지도학습의 종류는 크게 분류(Classification)와 회귀(Regression)로 나눌 수 있다.[10] [표 2-2]는 지도 학습의 대표적인 예들을 나타낸다.

Artificial Neural Network	Decision Tree
Regression	Nearest Neighbor Algorithm
Support Vector machine	Random Forests
Naive Bayes Classification	Ensembles of classifiers

[표 2-2] 지도 학습의 대표적인 예들

2.3.2 자율 학습(Unsupervised Learning)

입력이 있지만 정해진 출력이 없는 경우이며 순수하게 데이터들이 갖고 있는 속성들을 이용해 그룹으로 나누는 경우이다. 즉 y없이 x만 이용해서 학습하는 것이며 학년, 성별 등의 속성별로 단순한 분류만을 한다. 군집화(Clustering)가 대표적인 예다. [표 2-3]는 자율 학습의 대표적인 예들을 나타낸다.[10]

Clustering	Dimensionality Reduction
PCA (Pinciple Component Analysis)	ICA (Idependent Component Analysis)
Non-negative Matrix Factorization	Singlular Value Decomposition

[표 2-3] 자율 학습의 대표적인 예들

Ⅲ. Methodology

3.1 데이터 마이닝 절차

본 연구에서는 '유해물 댓글 데이터 마이닝'으로 기존 유해물 탐지 기법과는 차별화된 방법을 제시한다. 페이스북의 GRAPHAPI를 통한 데이터 수집부터, KoNLPy 패키지를 이용한 Tokenizing(토큰화), 4가지 방법의 분석, 각 분석에 대한 결과 해석으로 연구가 진행되었다.

〔그림 3-1〕 데이터 마이닝 절차

3.2 분석 도구

본 연구는 Windows 8.1 k 환경에서 진행하였으며 SQLite3, KoNLPy 0.4.3, Python2.7.0을 분석 도구로서 사용하였다. 데이터 수집 및 분석 코드는 Python 언어로 작성하였다.

절차	사용 툴		사용 목적
데이터 수집	SQLite3	데이터베이스 파일을 생성하고 디자인하기 편리한 오픈소스 툴이다.	데이터베이스 파일을 생성하여 페이스북 댓글을 저장한다.
데이터 전처리	KoNLPy 0.4.3	KoNLPy를 통해 형태소 분석기 NLTK를 Python에서 사용한다.	한글로 이루어진 댓글을 형태소로 토큰화한다.
학습 및 분석	Python 2.7.0	Python의 문법은 아주 간결하고 가독성이 좋고, 다양한 패키지를 사용할 수 있다.	데이터 수집, 분석알고리즘 코드를 Python 언어로 작성하고 실행한다.
	Python 라이브러리	sklearn : 분석 함수를 포함한다. numpy : 배열을 생성한다. matplotlib : 분석 결과를 시각화한다.	Python의 라이브러리를 분석에 사용하였다.

〔표 3-1〕 절차 별 사용 툴과 사용 목적

3.3 단계별 방법론 소개

이 절에서는 데이터 수집부터 전처리까지의 단계를 서술한다. Facebook의 게시물 수집, Tokenizing을 통한 속성 정의, Ground truth 생성의 단계로 진행하였다. 일련의 과정은 Python 코드로 이루어졌다.

3.3.1 데이터 수집 및 탐구

Facebook으로부터 개발자 등록 후 페이지의 게시물에 달린 댓글들을 수집하였다. 데이터 수집에는 Python2.7.0와 SQLite를 도구로써 사용하였다.

3.3.1.1 분석 대상 데이터 선정 기준

본 연구에서는 한국 대법원에서 규정한 음란성 기준을 적용하여 유해성 페이지를 선정하였다. 한국 대법원에서는 음란성 개념을 다음의 세 가지 요소로 일관되게 규정한다.

당해 행위 및 기타 매체를 통하여 성욕의 흥분 및 자극이 가해진다.
그로 인하여 일반인의 정상적 성적 수치심을 해한다.
결과적으로 성적 도의관념에 반하게 된다.

〔표 3-2〕 음란성 개념의 세 가지 요소

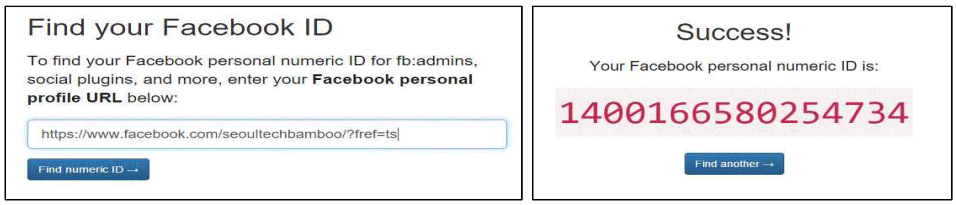
유해성 페이지는 〔표 3-2〕의 요소를 만족하며 1만 개 이상의 페이지 '좋아요' 수를 보유하고 있고, 게시물에 50개 이상의 댓글이 있는 페이지로 선정하였다.

남자들의 ****	**** 야한얘기	*** 야한거 **
야**툰	*** 성인	19세 ***
남자** **창고	남자의 ***	남자를 위한 ******
무조건 **** 19금	**들을 위한 19***	단언컨대 *** ***

〔표 3-3〕 유해성 페이지 목록

〔표 3-3〕의 유해성 페이지 12개에서 총 2,300개의 게시물을 선정하였다. 비유해성 페이지는 다양한 내용이 등장하는 '서울과학기술대학교 대나무숲' 페이지에서 총 2,300개의 게시물을 선정하였다.

3.3.1.2 페이지 위치 및 고유 ID 파악



〔그림 3-2〕 페이지 고유 ID 파악

Facebook에는 수많은 유·무해 게시물을 포함한 페이지가 있다. 〔그림 3-2〕처럼 'Find my Facebook ID(http://findmyfbid.com/)'라는 페이지를 통해 페이지 고유의 ID 값을 얻어냈다. 같은 방법으로 게시물의 고유한 ID를 얻어 코드에서 데이터의 위치를 나타내는 값으로 사용하였다.

```
keys = []
ids = ['1400166580254734', '1637982839806439'] #if a page name contains multiple words,
# separated by space (e.g. SPOT Coffee); it will show up as words connected by hyphens in URL.
# For example, SPOT Cofee as in "Spot-Coffee-Elmwood." If so, it is recommended that you use page id.
# You can find page id in the URL - it is the string of numbers after page name in URL.
# (e.g. https://www.facebook.com/pages/Spot-Coffee-Elmwood/316579834919)
```

〔그림 3-3〕 페이지와 게시물 각각의 ID를 입력

3.3.1.3 데이터 수집 권한 획득

Facebook은 개발자 등록을 한 사용자에게 개발자 Token을 부여한다. 이렇게 부여받은 토큰으로 수집 권한을 얻음으로써 원하는 데이터를 수집하였다.

```
url = "https://graph.facebook.com/%s/posts?access_token=CAACE...h2SAZDZD&include_hidden=true"
```

〔그림 3-4〕 개발자 Token 입력

3.3.1.4 데이터 수집 및 저장

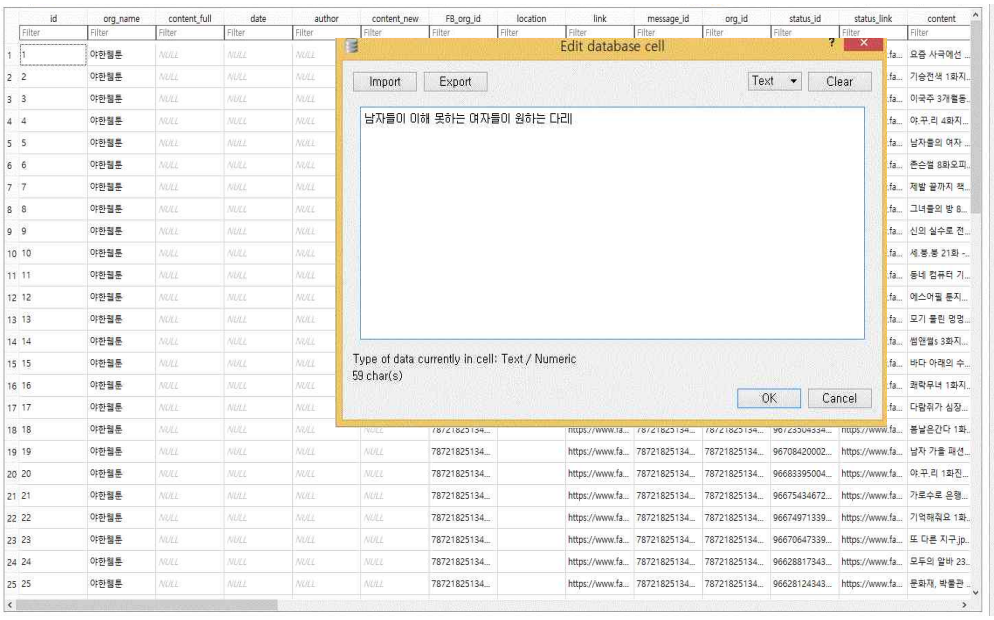
페이지와 게시물 각각의 ID와 개발자 Token을 Python 코드에 적용하여 페이스북 페이지 게시물의 데이터 수집을 실시한다. 〔그림 3-5〕처럼 SQLite파일을 생성하여 수집된 데이터를 저장한다.

```
class Scrape:
    def __init__(self):
        engine = sqlalchemy.create_engine("sqlite:///gogo.sqlite", echo=False)

        Session = sessionmaker(bind=engine)

        self.session = Session()
        Base.metadata.create_all(engine)
```

〔그림 3-5〕 SQLite 파일을 생성하여 데이터 저장

3.3.1.5 DB에 저장된 데이터 변환

〔그림 3-6〕은 수집한 데이터를 SQLite DB에 저장한 내용이다. 이와 같은 방법으로 Facebook의 유해성 페이지와 비유해성 페이지의 게시물을 수집하였다.



〔그림 3-6〕 SQLite에 저장된 게시물 내용과 댓글이 담긴 cell

SQLite DB에 저장된 데이터들을 csv파일 형태로 추출하여 분석데이터로 활용하였다.

id	org_name	content_fu	date	author	content_n	FB_org_id	location	link	message_i	org_id	status_id	status_link	content	published_
1	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.19E+14	https://ww	음란마귀 (	2015-10-0
2	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.19E+14	https://ww	손만 씻으	2015-10-0
3	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.19E+14	https://ww	아프리카	2015-10-0
4	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.15E+14	https://ww	북한군들으	2015-09-2
5	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.15E+14	https://ww	다시보는	2015-09-2
6	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.15E+14	https://ww	썸남의 위	2015-09-2
7	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.15E+14	https://ww	네스호의	2015-09-2
8	웃기고 야한거 좋아?					4.95E+14		https://ww	495020397	4.95E+14	7.15E+14	https://ww	흔한 42세	2015-09-2

〔그림 3-7〕 SQLite DB에서 추출한 csv파일

추출된 csv파일은 게시물의 id와 페이지 이름, 주소, 게시 글 내용, 댓글 내용 등을 포함한다. 본 연구에서 사용될 데이터는 댓글의 내용이므로, 댓글에 해당하는 column을 추출하여 분석에 사용하였다.

3.3.2 KoNLPy 패키지를 통한 데이터 전처리

이 절에서는 수집데이터 전처리 과정에 대하여 서술한다. 게시물의 댓글을 형태소로 분리하여 형식형태소는 제거하고, 실질형태소를 '단어'로 정의하여 단어별 빈도수를 추출하였다. 단어의 빈도수를 이용해 분석의 기준이 되는 속성을 정의하고 데이터를 전처리하여 분석 데이터와 정답 데이터(Ground truth)를 생성하였다.

3.3.2.1 빈도수 추출

변수를 추출하기 위해서 1,700개의 유해성 게시물과 1,700개의 비유해성 게시물을 선정하였다. KoNLPy를 이용하여 선정된 게시물에서 유해성 단어와 비유해성 단어 각각의 빈도수를 추출하였다.

```
In [1]: from konlpy.corpus import kobill

In [2]: files_ko = kobill.fileids()

In [3]: doc_ko = kobill.open('comment1.txt').read()

In [4]: from konlpy.tag import Twitter;

In [5]: t = Twitter()

In [6]: tokens_ko = t.nouns(doc_ko)
```

〔그림 3-8〕 댓글 Tokenizing

〔그림 3-8〕는 KoNLPy 말뭉치 기능으로 댓글을 읽어내는 코드이다. 그다음 Twitter class에 포함된 nouns(명사 추출기) 함수를 이용하여 형태소로 분리하였다. 텍스트 문서의 댓글들을 단어별로 토큰화한 것을 변수 tokens_ko에 저장하였다.

```
In [7]: import nltk

In [8]: ko = nltk.Text(tokens_ko, name = '음란1호')

In [9]: data = ko.vocab().items()

In [10]: import csv

In [11]: with open('words.csv', 'w') as f:
   ....:     f.write('word,freq₩n')
   ....:     writer = csv.writer(f)
   ....:     writer.writerows(data)
```

〔그림 3-9〕 csv파일로 단어 빈도수 저장

〔그림 3-9〕는 nltk 모듈을 이용하여 구문을 분석하는 코드다. tokens_ko을 ko 변수에 '음란 1호'라는 이름을 지정해서 저장하였다. 그 후, 형태소와 빈도수를 목록화한 뒤 'words.csv' 파일로 저장하였다.

3.3.2.2 단어 탐색 및 정제

'words.csv' 파일은 형태소의 빈도수를 나타낸다. 이 형태소들에서 속성을 선정하기 위해 정제 단계를 거쳤다. 속성은 유해 속성과 비유해 속성으로 구분되며 형식 형태소를 제외한 실질 형태소를 활용한다. 합리적이고 유효한 변수를 선택하여 속성으로 결정하는 것은 성능이 좋은 분류기를 구현함에 있어서 매우 중요한 요소이다. 따라서 더 정확한 속성 정의를 위해 〔그림 3-10〕과 같은 가중치 식을 이용하였다.

$$\text{단어의 가중치} = \frac{\text{해당 유해 단어 빈도수}}{\text{총 유해 단어수}} \times \log\left(\frac{\text{총 비유해 단어수}}{\text{해당 유해 단어의 비유해 게시물에서의 빈도수}}\right)$$

〔그림 3-10〕 단어 가중치 계산

로그단어빈도 가중치 공식은 빈도가 지나치게 낮은 단어의 영향력을 보충하고, 빈도가 높은 단어의 지나친 영향력을 낮추기 위해 사용한다.〔11〕 본 연구에서는 게시물의 특정 내용 때문에 발생한 빈도수가 높게 나타난 단어는 유해 여부를 결정하기에 부적절하다고 판단하여 로그단어빈도 가중치 공식을 〔그림 3-10〕의 식에서 활용하였다.

	A	B	C	D	E	F
1	word	freq	비율	1-비율	가중치	비중
2	년	703	0.03168	0.96832	5.2591	0.1666
3	사다리	631	0.02936	0.97064	4.72047	0.13861
4	와	573	0.02825	0.97175	4.28658	0.1211
5	개	576	0.02762	0.97238	4.30902	0.11899
6	여자	479	0.0243	0.9757	3.58337	0.08709
7	존나	460	0.02392	0.97608	3.44123	0.08232
8	놀이터	428	0.0228	0.9772	3.20184	0.07301
9	검증	386	0.02104	0.97896	2.88764	0.06077
10	지급	352	0.0196	0.9804	2.63329	0.05162
11	이벤트	333	0.01892	0.98108	2.49115	0.04712
12	남자	328	0.01899	0.98101	2.45375	0.0466
13	시발	315	0.01859	0.98141	2.3565	0.04381
14	먹튀	289	0.01738	0.98262	2.16199	0.03758
15	오	266	0.01628	0.98372	1.98993	0.0324
16	새끼	239	0.01487	0.98513	1.78795	0.02659
17	대박	221	0.01396	0.98604	1.65329	0.02308
18	일본	158	0.01012	0.98988	1.18199	0.01196
19	소리	140	0.00906	0.99094	1.04733	0.00949
20	실시간	134	0.00875	0.99125	1.00245	0.00877
21	병신	118	0.00777	0.99223	0.88275	0.00686
22	가슴	109	0.00724	0.99276	0.81542	0.0059
23	소름	102	0.00682	0.99318	0.76306	0.0052
24	쓰레기	100	0.00673	0.99327	0.74809	0.00504
25	지랄	99	0.00671	0.99329	0.74061	0.00497
26	배팅	98	0.00669	0.99331	0.73313	0.0049
27	몸매	97	0.00666	0.99334	0.72565	0.00484
28	쿵	89	0.00616	0.99384	0.6658	0.0041
29	쌍년	88	0.00612	0.99388	0.65832	0.00403
30	욕	87	0.00613	0.99387	0.65084	0.00399
31	소개	87	0.00609	0.99391	0.65084	0.00396

〔그림 3-11〕 유해 단어 비중 계산

	A	B	C	D	E	F
1	word	freq	비율	1-비율	가중치	비중
2	사람	898	0.03935	0.96065	5.98153	0.23535
3	학교	395	0.01802	0.98198	2.63107	0.0474
4	동아리	388	0.01802	0.98198	2.58445	0.04658
5	생각	379	0.01793	0.98207	2.5245	0.04526
6	말	374	0.01801	0.98199	2.49119	0.04487
7	댓글	366	0.01795	0.98205	2.4379	0.04376
8	우리	296	0.01478	0.98522	1.97164	0.02915
9	익명	287	0.01455	0.98545	1.91169	0.02781
10	친구	270	0.01389	0.98611	1.79845	0.02498
11	시간	251	0.01309	0.98691	1.6719	0.02189
12	학생	222	0.01173	0.98827	1.47873	0.01735
13	고수	211	0.01129	0.98871	1.40546	0.01586
14	정말	209	0.01131	0.98869	1.39214	0.01574
15	수업	190	0.0104	0.9896	1.26558	0.01316
16	마음	185	0.01023	0.98977	1.23227	0.0126
17	문제	164	0.00916	0.99084	1.09239	0.01001
18	학기	163	0.00919	0.99081	1.08573	0.00998
19	학생회	156	0.00888	0.99112	1.03911	0.00922
20	공부	134	0.00769	0.99231	0.89257	0.00687
21	대술	130	0.00752	0.99248	0.86592	0.00651
22	학우	121	0.00705	0.99295	0.80597	0.00568
23	공감	120	0.00704	0.99296	0.79931	0.00563
24	개인	119	0.00704	0.99296	0.79265	0.00558
25	학점	116	0.00691	0.99309	0.77267	0.00534
26	사랑	114	0.00683	0.99317	0.75935	0.00519
27	시험	114	0.00688	0.99312	0.75935	0.00523
28	화이팅	113	0.00687	0.99313	0.75269	0.00517
29	관심	108	0.00661	0.99339	0.71938	0.00476
30	학년	106	0.00653	0.99347	0.70606	0.00461
31	연애	106	0.00657	0.99343	0.70606	0.00464

〔그림 3-12〕 비유해 단어 비중 계산

3.3.2.3 속성 선정

단어별로 빈도수, 비율, 1-비율, 가중치, 비중을 계산하여 속성을 정의하는 기준으로 삼았다. 유해성, 비유해성 게시물에서 (가중치*비율) 값인 비중이 높은 순으로 분석에 있어 기준이 되는 단어를 선정하였다. 유해 단어 30개, 비유해 단어 30

개를 속성으로 정의하였다.

유해 변수	'년','사다리','개','와','여자','존나','놀이터','검증','지급','이벤트','남자','시발','먹튀','오','새끼','대박','일본','소리','실시간','병신','가슴','소름','쓰레기','지랄','배팅','몸매','좆','쌍년','소개','욕'
비유해 변수	'사람','학교','동아리','생각','말','댓글','우리','익명','친구','시간','학생','교수','정말','수업','마음','문제','학기','학생회','공부','대숲','학우','공감','개인','학점','시험','사랑','화이팅','관심','연애','학년'

〔표 3-4〕 유해성 속성과 비유해성 속성

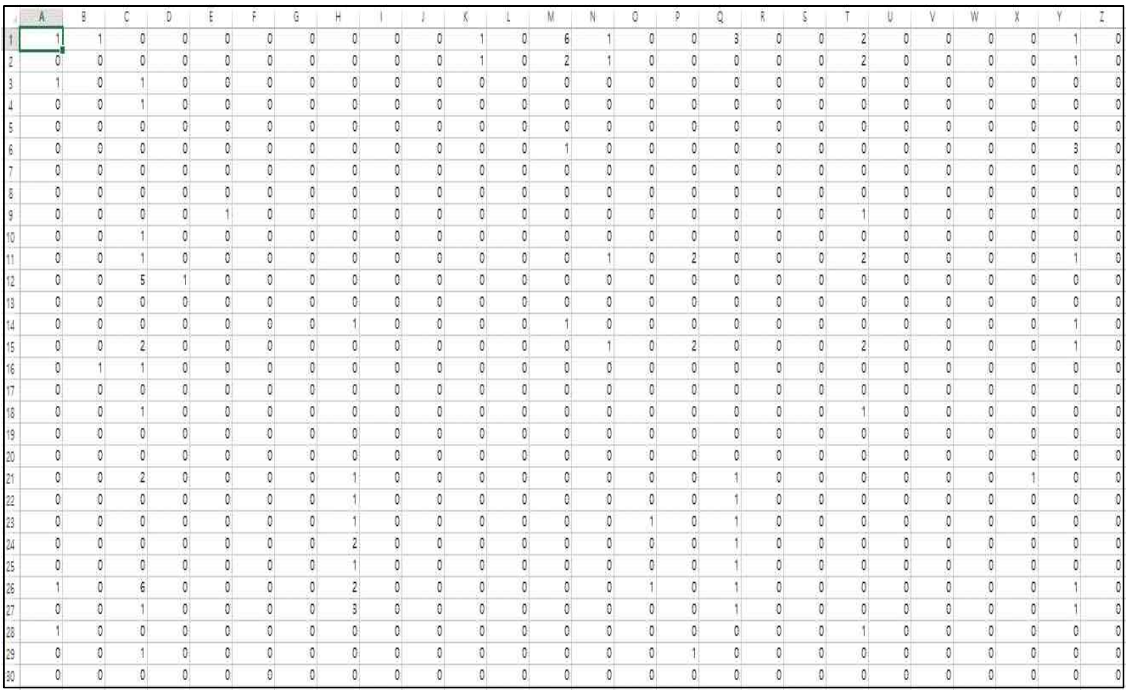
3.3.2.4 Ground truth

분석에서 사용될 총 4,600개의 게시물(유해 2,300개+비유해 2,300개)을 새롭게 수집하여 해당 게시물의 속성 빈도수와 해당 게시물의 유해성 여부에 대한 정답 파일을 생성하였다. 정답 파일에는 유해성 게시물은 1로, 비유해성 게시물은 0으로 표기된다.

```
#-*-coding:utf-8-*-
import csv
from sklearn import datasets
import numpy as np
import operator
from functools import reduce
import matplotlib.pyplot as plt
from sklearn import linear_model, datasets
f = open('유해_모음.csv', 'r')
a = ['우리','사람','나','진짜','너무','오늘','하나','돈','오빠','지금','학교','대박','만원'
result=[]
y=[]
t=[]
print(a)
csv_f = csv.reader(f)
k=0
for row in csv_f:
    j=0
    temp=[]
    for i in a:
        temp.append(row[48].count(i)+row[49].count(i))
        j=j+1
    result.append(temp)
    k=k+1


f = open('비유해_모음.csv', 'r')
csv_f = csv.reader(f)
k=0
for row in csv_f:
    j=0
    temp=[]
    for i in a:
        temp.append(row[48].count(i)+row[49].count(i))
        j=j+1
    result.append(temp)
    k=k+1

y.append([0]*2400+[1]*2400)

x=np.array(result)
y=np.array(y)

f = open('x_value.csv', 'w')
cw=csv.writer(f, delimiter=',', quotechar=' ', quoting=csv.QUOTE_ALL)
for row in x:
    cw.writerow(row)

f = open('y_value.csv', 'w')
cw=csv.writer(f, delimiter=',', quotechar=' ', quoting=csv.QUOTE_ALL)
for row in y:
    cw.writerow(row)
cw.writerow(y)
f.close()
```

〔그림 3-13〕 Ground truth 생성 코드

〔그림 3-13〕의 Python 코드에서 'x_value.csv'는 4,600개의 각 게시물에서 60개의 속성에 대한 빈도수를 나타내는 파일이다. 'y_value.csv'는 2,300개의 유해성 게시물(1)과 2,300개의 비유해성 게시물(0)의 정답을 나타낸다.



〔그림 3-14〕 각각 게시물에서의 속성 빈도수(x_value.csv)

'x_value.csv'는 총 4,600개의 열을 포함한 배열 데이터로 이루어져 있다. row는 60개의 속성을, column은 4,600개의 게시물을 포함한다. 이 데이터는 분석 데이터로 70%는 Training_Data(훈련데이터)로, 30%는 Test_Data(테스트데이터)로 사용된다.

3.4 분석 알고리즘

본 연구에서는 지도 학습 알고리즘 중 Logistic Regression, Decision Tree, K-NN Clustering, Naïve Bayes Classification을 분석기법으로 사용하였다. 각각의 분석기법을 통해 정확도를 확인하고 최적의 방법을 찾아낸다.

3.4.1 Logistic Regression

로지스틱 회귀분석(Logistic Regression)은 존재(1)/부재(0) 형태로 되어 있는 종속변수에 대하여 특정 조건을 의미하는 한 개 이상의 독립변수들을 가지고 이를 추측하기 위해 회귀분석을 하는 것이다. 일반적인 회귀분석과 유사하지만, 종속변수가 구간 비율척도 변수가 아니라 0과 1의 값을 가지는 것이 다른 점이다.

종속변수가 범주형 변수라는 점은 판별분석과 같지만 독립변수들이 다변량 정규분포를 따르지 않아도 되기 때문에 판별분석 보다 덜 제한적이어서 다양하게 사용할 수 있다.

3.4.2 K-NN(K-Nearest Neighbors) Clustering

패턴 인식에서 K-NN(최근접 이웃 알고리즘)은 분류나 회귀에 사용되는 비모수 방식이다. Outlier 때문에 잘못 분류될 가능성을 방지해 주는 장점이 있지만 모든 변수에 대해 거리를 계산한다는 단점이 있다. 또한, 모수 k를 정하는 방법이 어렵고 데이터의 지역구조에 민감하다.

3.4.3 Decision tree

의사결정나무(Decision Tree)는 의사결정규칙(Decision Rule)을 트리구조로 도표화 하여 분류와 예측을 수행하는 분석 방법이다. 분류와 예측의 과정이 나무구조에 의한 추론규칙에 의해서 표현되기 때문에 다른 방법(신경망, 판별분석, 회귀분석 등)에 비하여 연구자가 그 과정을 쉽게 이해하고 설명할 수 있다는 장점이 있다.

3.4.4 Naive Bayes Classification

나이브 베이즈 분류(Naive Bayes Classification)는 베이즈 정리(특성들 사이의 독립을 가정)를 적용한 확률 분류기의 일종이다. 이는 분류기를 만들 수 있는 간단한 기술로써 일반적인 원칙에 근거한 여러 알고리즘을 이용하여 훈련한다. 지도 학습에서 매우 효율적이고, 트레이닝 데이터의 양이 매우 적으며 복잡한 실제 상황에서 잘 작동한다.

Ⅳ. Experiment

4.1 분석 결과

본 연구에서 사용된 4가지 분석 방법은 sklearn라이브러리로부터 추출된 모델 함수로써 배열 형태의 데이터를 입력 값으로 요구한다. 때문에 앞서 csv파일을 배열(numpy.ndarray) 형태로 변환했다. 각각의 분석은 총 4,600개에 해당하는 데이터를 70%는 훈련데이터, 30%는 테스트데이터로 사용하였다.

4.1.1 논리적 회귀분석 (Logistic Regression)

〔그림 4-1〕은 논리적 회귀분석하기 위한 Python 코드를 나타낸 것이다. sklearn.linear_model에서 Logistic Regression 함수를 import 하여 분석하였다.

```
from sklearn import cross_validation
X_train, X_test, y_train, y_test = cross_validation.train_test_split(x,y,test_size=0.3, random_state=1020)
from sklearn.linear_model import LogisticRegression
lr= LogisticRegression(random_state=1020)

lr.fit(X_train, y_train)

print(lr.score(X_train, y_train))
print(lr.score(X_test, y_test))
```

〔그림 4-1〕 Logistic Regression 분석 코드

random_state=1020으로 설정하였을 때 다음과 같이 Training_Data와 Test_Data의 정확도를 추출하였다.

```
C:₩datamining>python test_logic.py
{'C': 1.0, 'verbose': 0, 'intercept_scaling': 1, 'fit_intercept': True, 'max_ite
r': 100, 'penalty': 'l2', 'multi_class': 'ovr', 'random_state': 1020, 'dual': Fa
lse, 'tol': 0.0001, 'solver': 'liblinear', 'class_weight': None}
0.804347826087
0.780434782609
```

〔그림 4-2〕 Training_Data와 Test_Data의 정확도

Training_Data의 정확도가 0.78 정도의 수치를 보였다. Training_Data의 결과와 Test_Data의 결과가 0.02 정도의 차이를 보였다.

4.1.2 K-최근접 이웃 알고리즘 (K-nearest neighbors)

K-최근접 이웃 알고리즘에서는 K(최근접 이웃의 수)의 설정에 따라 결과의 값이 달라진다. 주로 Training_Data 개수의 제곱근을 한 값을 많이 사용하므로, k의 값은 57로 설정하였다. 〔그림 4-3〕은 K-최근접 이웃 알고리즘을 실행하기 위한 Python 코드를 나타낸 것이다. sklearn.neighbors에서 KNeighborsClassifier 함수를 import 하여 분석하였다.

```
from numpy import genfromtxt
X_bin = genfromtxt('x_value.csv', delimiter=',')
y_bin = genfromtxt('y_value.csv', delimiter=',')
print(type(X_bin), type(y_bin))
print(X_bin.shape, y_bin.shape)


from sklearn import decomposition
pca = decomposition.PCA(n_components=2)
pca.fit(X_bin)
Z=pca.transform(X_bin)


from sklearn import cross_validation
X_train, X_test, y_train, y_test = cross_validation.train_test_split(Z, y_bin, test_size=0.3, random_state=1234)
from sklearn.neighbors import KNeighborsClassifier

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.neighbors import NearestCentroid
nbrs = KNeighborsClassifier(n_neighbors=57)
nbrs.fit(X_train,y_train)
print(nbrs.score(X_train,y_train))
print(nbrs.score(X_test,y_test))
```

〔그림 4-3〕 K-nearest neighbors 분석 코드

random_state=1234으로 설정하였을 때 다음과 같이 Training_Data와 Test_Data의 정확도를 추출하였다.

```
C:₩datamining>python test_knn_nongraph.py
(<type 'numpy.ndarray'>, <type 'numpy.ndarray'>)
((4600L, 60L), (4600L,))
0.647826086957
0.621014492754
```

〔그림 4-4〕 Training_Data와 Test_Data의 정확도

Test_Data의 정확도는 0.62 정도의 수치를 보였다. 로지스틱 회귀분석의 결과보다 0.16 낮은 결과를 보였다.

4.1.3 나이브 베이지 분류(Naive Bayes Classification)

〔그림 4-5〕는 나이브 베이지 분류를 실행하기 위한 Python 코드를 나타낸 것이다. sklearn.naive_bayes에서 Gaussian NB 함수를 import 하여 분석하였다.

```
print(X_bin.shape, y_bin.shape)

X_train, X_test, y_train, y_test = cross_validation.train_test_split(X_bin, y_bin, test_size=0.3, random_state=1234)

clf = GaussianNB()
clf = clf.fit(X_train, y_train)

print(clf.score(X_train, y_train))
print(clf.score(X_test, y_test))
```

〔그림 4-5〕 Naive Bayes Classification 코드

random_state=1234으로 설정하였을 때 다음과 같이 Training_Data와

Test_Data의 정확도를 추출하였다.

```
C:₩datamining>python test_naive.py
((4600L, 60L), (4600L,))
0.748447204969
0.739855072464
```

〔그림 4-6〕 Training_Data와 Test_Data의 정확도

Test_Data의 정확도가 0.73 정도의 수치를 보였고, Training_Data의 결과와 Test_Data의 결과가 0.01 정도의 차이를 보였다.

4.1.4 의사결정나무(Decision-Tree)

〔그림 4-7〕은 의사결정나무 분석을 실행하기 위한 Python 코드를 나타낸 것이다. sklearn.tree에서 DecisionTreeRegressor 함수를 import 하여 분석하였다.

```
X_bin = genfromtxt('x_value.csv', delimiter=',')
y_bin = genfromtxt('y_value.csv', delimiter=',')

print(X_bin.shape, y_bin.shape)

X_train, X_test, y_train, y_test = cross_validation.train_test_split(X_bin, y_bin, test_size=0.3, random_state=1020

clf = tree.DecisionTreeRegressor()
clf = clf.fit(X_train, y_train)
export_graphviz(clf)
print(clf.score(X_train, y_train))
print(clf.score(X_test, y_test))
```

〔그림 4-7〕 Decision-Tree 분석 코드

random_state=1020으로 설정하였을 때 다음과 같이 Training_Data와 Test_Data의 정확도 값을 추출하였다.

```
C:₩datamining>python test_decision.py
((4600L, 60L), (4600L,))
0.65184714817
0.190635643004
```

〔그림 4-8〕 Training_Data와 Test_Data의 정확도

Test_Data의 정확도가 0.19 정도의 수치를 보였다. 4가지 분석기법 중 가장 낮은 분석결과를 보이며, 50% 수준에도 미치지 못했다.

4.2 결과 해석

본 절에서는 4가지 분석결과에 대한 해석을 풀이한다. 4.2절에서 분석한 결과를 정리하면 〔표 4-1〕과 같다.

Methods	정확도
Logistic Regression	78.04%
K-nearest neighbors	62.10%
Naive Bayes Classification	73.99%
Decision-Tree	19.06%

〔표 4-1〕 Test_Data의 분석 결과

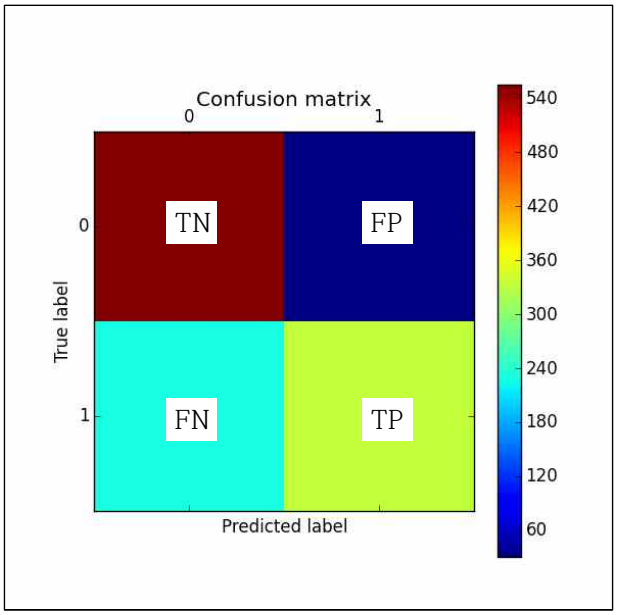
분석 결과 Decision-Tree를 제외한 나머지 3가지 분석기법에서 유의한 결과를 얻을 수 있었다. 가장 높은 정확도를 보인 분석은 Logistic Regression으로, 연구 결과 78%의 음란물 탐지능력을 보였다.

따라서 Logistic Regression 모델을 최적의 분석기법으로 선정하였다. Logistic Regression 모델의 분석 결과를 평가하기 위해서 Confusion Matrix를 작성해보았다.

```
from sklearn import decomposition
pca = decomposition.PCA(n_components=2)
pca.fit(x)
Z=pca.transform(x)

from sklearn import cross_validation
X_train, X_test, y_train, y_test = cross_validation.train_test_split(Z,y,test_size=0.3, random_state=1020)
from sklearn.linear_model import LogisticRegression
lr= LogisticRegression(random_state=1020)
h=.02
lr.fit(X_train, y_train)
X=X_test
y=y_test
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = lr.predict(np.c_[xx.ravel(), yy.ravel()])

Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(4, 3))
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap=plt.cm.Paired)
plt.xlabel('X')
plt.ylabel('Y')

plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())

plt.show()
```

〔그림 4-9〕 Logistic Regression에서의 Confusion Matrix 코드

〔그림 4-9〕의 코드에 따라 진행된 결과는 〔그림 4-10〕과 같다.



〔그림 4-10〕 Confusion Matrix

〔그림 4-10〕은 Logistic Regression에서의 Confusion Matrix를 나타낸 것이다. Confusion Matrix란 데이터 마이닝의 평가 단계에서 사용하는 기법으로, 전체 Test_Data에 대한 결과를 한꺼번에 보여준다. True label은 Actual Data를 의미하고, Predicted label은 Predicted Data를 의미한다. 또한 0은 비유해를, 1은 유해를 의미한다. 데이터 마이닝에서는 분석 모델의 평가를 위해 Confusion Matrix를 사용하는데, 모델의 Accuracy는 (TP+TN)/(TP+TN+FP+FN)로 구한다. 로지스틱 회귀분석의 경우 0.78 정도의 정확도를 얻음으로써 앞서 언급했듯이 SNS 댓글 수집을 통한 데이터 분석이 유해물 탐지 성능 향상에 충분한 도움이 될 수 있음을 증명한 실험이었다.

Ⅴ. Evaluation

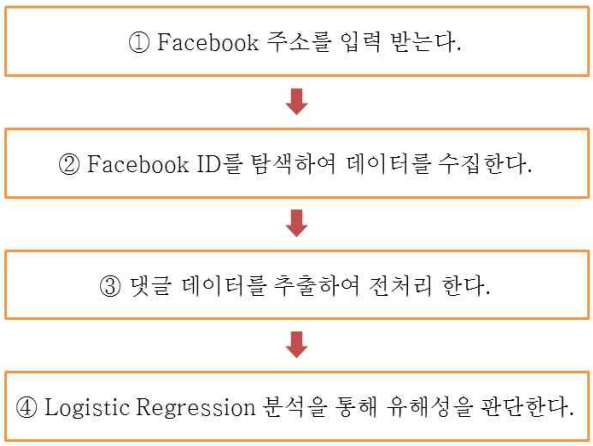
5.1 연구 의의

Logistic Regression 분석 결과 정확도 78%의 수준으로, Naive Bayes Classification 결과 정확도 74%의 수준으로 유해성 페이지를 유해 페이지로 옳게 판단하였다. 향후 연구에서는 더 많은 데이터를 수집하고 속성을 정의하는 데 사용한다면 정확도를 높일 수 있을 것이라 생각한다. 이로써 댓글 형태소를 이용한 속성정의 및 분석이 유해물 판단에서 발전 가능성이 있는 모델임을 증명한다.

5.2 향후 연구계획

본 연구에서 제시한 분석 모델은 속성정의에 따라 분석의 정확도가 달라질 수 있다. 환경에 따른 정확도의 변동 차이를 줄이기 위해서는 유해와 비유해를 대표하는 속성을 명확히 설정해야 한다. 이를 위해, 다수의 유해성 페이지를 선정해 빈번히 출현하는 단어들을 추출하여 유해 속성으로 정의해야 한다. 의미 있는 속성을 정의하는 것은 모델의 정확도를 향상하기에 효율적인 방법이다. 따라서 유해성 속성과 비유해성 속성이 각각 유해성 페이지와 비유해성 페이지를 잘 대표한다면 더 발전된 모델이 될 것이다.

모델의 성능을 향상한 후, 본 연구에서 진행된 분석을 자동화하는 새로운 연구를 진행할 예정이다. Facebook 게시물의 주소를 입력하면 즉시 유해성 여부를 판단해주는 소프트웨어를 개발할 것이다. 〔그림 5-1〕는 향후 연구 계획인 프로그램의 알고리즘을 나타낸 것이다.



〔그림 5-1〕 프로그램 알고리즘

Facebook의 주소를 입력하면 Facebook ID를 탐색하고 그 페이지에 대해서 데이터 수집을 진행한다. 그리고 수집된 데이터에서 댓글 데이터만을 추출하여 전처리 과정을 거친 후, 미리 정의된 속성들을 통해서 분석을 진행하여 유해성 판단 결과를 출력하는 프로그램이다.

Ⅵ. Conclusion

본 연구에서는 댓글의 형태소 분석을 통한 Facebook 유해 게시물 탐지 모델을 개발하였다. Facebook의 타임라인에서도 음란물을 쉽게 접할 수 있듯이 현재 SNS에서의 유해물 유포현황은 심각한 수준에 이른다. 이뿐만 아니라 Instagram에서도 음란 해시태그로 청소년들이 유해물에 무방비로 노출되고 있다. 방송통신심의위원회 청소년 보호법에 의해 국내 기업 음란물 규제는 엄격하지만, 해외에 서버를 둔 기업들은 국내법을 적용할 수가 없으므로 음란물 탐지 기술을 적용해서 판별하여 별다른 조치를 취해야 한다.

그리고 등급에 따른 문서 분류기술, 하르 분류기를 이용한 분석 등 기존의 분석 기법과는 다른 모델을 제시하였다. 이는 게시글 자체가 아닌, 게시글의 댓글 내용을 데이터마이닝 함으로써 유해성을 판단하는 기법이다. Facebook 대부분의 댓글이 한글임을 고려하여 한국어 형태소 분류와 데이터 마이닝을 진행하였다. 데이터 전처리를 거친 후 4가지 분석을 통해 이 모델의 실현 가능성과 당위성에 대하여 증명하였다.

분석 알고리즘의 성능은 지속적인 피드백을 통하여 향상할 수 있다. 또한, 텍스트가 존재하지 않는 게시물도 유해성 판단이 가능하므로 이는 유해물의 특성을 잘 반영한 분석모델이라 할 수 있다. 이 모델은 언제든 속성의 재정의가 가능하므로 변화가 많은 SNS상의 게시물 분석에서 유용하게 쓰일 것으로 기대된다.

Reference

〔1〕 국립국어원 표준국어대사전. “정보화사회.” http://stdweb2.korean.go.kr/.

〔2〕 한국인터넷진흥원 (2014). “2014년 인터넷이용실태조사 요약보고서.” http://www.kisa.or.kr. (검색일: 2015.09.19.).

〔3〕 한국방송통신심의위원회 (2014), “SNS 시정요구 현황.” http://www.kocsc.or.kr/. (검색일: 2015.09.20.).

〔4〕 이정환・김현정・원일용. (2011). “하르 분류기가 인식한 인체특정부분의 기하학적 관계를 이용한 음란 이미지 탐지.” 제36회 한국정보처리학회 추계학술발표대회 논문집 제18권 제2호.

〔5〕 김영수・남택용・원동호 (2006), “등급에 따른 웹 유해 문서 분류 기술.” 한국정보처리학회 정보처리학회논문지C.

〔6〕 신준수 (2013). “KOMORAN 2.0 - 자바 기반의 한국어 형태소 분석기.” http://www.shineware.co.kr. (검색일: 2015.11.02.).

〔7〕 MeCab : Yet Another Part-of-Speech and Morphological Analyzer. (연도불명). “MeCab (和布蕪)とは.” http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html. (검색일: 2015.11.02.).

〔8〕 박은정・조성준 (2014). “KoNLPy: 쉽고 간결한 한국어 정보처리 파이썬 패키지.” 제26회 한글 및 한국어정보처리 학술대회 논문집.

〔9〕 박은정 (2014). “KoNLPy: 파이썬 한국어 NLP - KoNLPy 0.4.4 documentation.” http://konlpy.org/ko/v0.4.4/.

〔10〕 나무위키 (연도불명). “기계학습.” https://namu.wiki/w/. (검색일자: 2015.11.04.)

〔11〕 김호성・정경호・황도삼 (2001). “단어 가중치를 이용한 스팸메일 필터링.” 제13회 한글 및 한국어정보처리 학술대회 논문집.