비지도 학습 기반 감성분석 모델 비교

: EDA와 전처리를 중심으로

우영석  
*인공지능융합학과   
성균관대학교 소프트웨어융합대학*2024711029

*Abstract*—본 프로젝트는 TextBlob, VADER, Pattern의 세 가지 비지도 학습 기반 감성분석 모델을 사용하여 트위터 데이터의 감성을 분석했고 이들의 성능을 비교했습니다. 특히 각 모델의 성능을 라벨링 기준에 따라, 텍스트 전처리 전/후로 비교했습니다. 본 연구를 진행하면서 데이터의 양이 적고 텍스트 길이가 짧은 경우 모델의 성능을 위해 최소한의 전처리를 고려할 필요가 있다는 점을 발견했습니다.

Keywords—비지도 학습, 감성분석, 트위터 데이터셋, EDA, 텍스트 전처리

# Introduction

본 프로젝트는 세 종류의 비지도 학습 기반 감성분석을 진행하는 것입니다. 감성분석은 주로 텍스트 데이터에 대해 이뤄지고 해당 데이터셋의 감성을 파악하는 것이 중요한 작업으로 여겨질 때 사용됩니다. 단, 본 프로젝트는 특정 작업 수행을 목표로 하기보다 텍스트 데이터에 대한 EDA, 전처리 그리고 감성분석 라이브러리를 적용해보는 것을 목표로 합니다. 특히 세 종류의 감성분석 라이브러리를 적용한 결과를 비교하는 것이 핵심 목표입니다.

본 프로젝트는 목표 수행을 위해 감성분석에 적합한 트위터 데이터를 분석합니다. 소셜 미디어는 개인의 감정과 의견을 표현하는 공간입니다. 특히 트위터는 페이스북과 같은 여타 다른 소셜미디어보다 훨씬 affordance[[1]](#footnote-1)가 높습니다. 따라서 트위터의 텍스트는 더 다양하고 풍부한 감성을 가지고 있을 것으로 기대됩니다. 요약하면 트위터 데이터셋은 감성분석을 적용하기에 매우 적합합니다. 따라서 본 프로젝트는 감성분석 라이브러리를 적용해보고 성능을 비교해보는 목표를 수행하기에 잘 구성되어 있습니다.

# Body

1. 모델 설명

본 프로젝트에서는 세 가지 유형의 비지도 학습 기반 감성분석 모델(TextBlob, VADER, Pattern)을 사용하였습니다.

첫째, TextBlob는 텍스트 데이터를 처리하기 위한 Python 라이브러리로, NLP의 일반적인 작업을 손쉽게 다룰 수 있도록 간단한 API를 제공합니다. 특히 TextBlob은 NLTK 및 Pattern 라이브러리를 기반으로 구축되었습니다. 본 프로젝트에서 사용하는 TextBlob의 감성분석 기능은 텍스트가 긍정적인지 부정적인지를 평가합니다. 각 문장의 감성 극성을 측정하여 문장이 전달하는 감정의 성향을 분석할 수 있습니다.[5]

둘째, VADER(Valence Aware Dictionary and sEntiment Reasoner)는 NLTK(Natural Language Toolkit)의 서브모듈로서 감성분석 라이브러리입니다. 특히 소셜 미디어 텍스트에 최적화되어 있습니다. 따라서 줄임말, 이모티콘, 대문자 사용, 구두점 등을 고려하여 감성을 분석합니다.[3]

셋째, Pattern은 다양한 자연어 처리 작업을 위해 사용되는 Python 라이브러리입니다. 그 중 sentiment() 함수가 텍스트의 감성을 분석하는 기능을 합니다. 감성분석 기능은 다른 라이브러리와 마찬가지로 주어진 문장을 -1.0(매우 부정적) ~ +1.0(매우 긍정적) 사이의 극성 값으로 표현해줍니다.[4]

2. 코드 분석 및 결과 분석

- EDA

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

데이터셋을 로드한 후 데이터프레임 형태로 변환한 결과는 위와 같습니다. 초기 데이터프레임에서 각 column은 순서에 따라 숫자로 표시되어 있습니다. rename을 사용하여 column의 이름을 변경하는 것은 이 데이터가 무엇을 나타내는지 더 쉽게 이해할 수 있게 했습니다. column 2는 'sentiment', column 3은 'text'로 변경하여 각각 감성 레이블과 텍스트 데이터라는 것을 명확히 표현했습니다. 또한 column 0과 1은 실험에서 불필요하기 때문에 제거했습니다.

# column명 바꾸기

df.rename(columns={2: 'sentiment', 3: 'text'}, inplace=True)

# 0,1 column 없애기

df.drop(columns=[0, 1], inplace=True)

# actual sentiment 값의 분포 확인

df['sentiment'].value\_counts()

# 실제 sentiment가 "Irrelevant"가 아닌 데이터만 남긴다.

df = df[df['sentiment'] != 'Irrelevant']

df['sentiment'].value\_counts()

아래 그래프는 데이터 전처리 후 감성 분석 데이터셋의 레이블 별 분포를 시각화한 결과입니다. 세 가지 유형의 감성 레이블인 'Neutral', 'Negative'과 'Positive'은 비슷한 비율로 분포되어 있다는 것을 알 수 있습니다.

텍스트, 스크린샷, 직사각형, 도표이(가) 표시된 사진

자동 생성된 설명

- 라벨 별 중요단어 EDA(예: ‘Positive’)

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

자동 생성된 설명

위 워드 클라우드는 ‘Positive’ 라벨의 트위터 게시글에 대한 중요 단어를 시각화한 것입니다. 워드 클라우드를 사용하는 주된 이유는 각 라벨에 대해 주로 사용된 핵심 단어들을 파악하여, 텍스트 데이터 내에서 감성의 주요 표현들을 식별하기 위함입니다.[2] 텍스트 전처리를 진행하기 전인데 모두 영어 알파벳이 주를 이루고 있다는 것을 알 수 있습니다.

-텍스트 전처리

def remove\_special\_characters\_and\_stopwords(text):

# 이모티콘 정규식을 가져옴

emoji\_list = emoji.emoji\_list(text)

emoji\_chars = [e['emoji'] for e in emoji\_list]

# 영어 알파벳, 숫자, 공백, 이모티콘을 제외한 나머지 문자 제거

cleaned\_text = []

for char in text:

if char.isalnum() or char.isspace() or char in emoji\_chars:

cleaned\_text.append(char)

cleaned\_text = ''.join(cleaned\_text)

# 불용어 처리

filtered\_words = [word for word in cleaned\_text.split() if word.lower() not in stop\_words]

return ' '.join(filtered\_words)

# 데이터 전처리

df['text'] = df['text'].str.strip() # 앞뒤 공백 제거

df['text'] = df['text'].str.lower() # 소문자로 변환

df['text'] = df['text'].apply(remove\_special\_characters\_and\_stopwords)

# 중복값 제거

df.drop\_duplicates(inplace=True)

위 코드는 감성 분석을 진행하기 전의 ‘text’ 열의 텍스트 전처리에 해당됩니다.

영어 알파벳, 숫자는 남기고 특수문자들을 제거했습니다. 이때 이모티콘도 함께 남겼습니다. 왜냐하면 이모티콘은 텍스트의 감성을 표현하는 중요한 요소로 감성 분석에서 무시할 수 없는 정보이기 때문입니다.

그 후 NLTK의 영어 불용어 리스트를 통해 텍스ㅌ의 중요하지 않은 부분을 줄였습니다. 또한 strip() 함수를 사용해 텍스트의 앞뒤 공백을 제거했고, lower() 함수를 통해 모든 텍스트를 소문자로 변환했습니다.

마지막으로 본 보고서에서는 생략했지만 EDA 결과에 따라 중복값과 결측치를 제거했습니다. drop\_duplicates() 함수를 사용하여 중복된 데이터를 제거했습니다. 결측치는 존재하지 않으므로 진행하지 않았습니다.

-모델링 및 결과 확인

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

데이터프레임에 각 모델 별로 감성분석 결과를 열로 추가했습니다. 데이터프레임의 위아래 5개 텍스트의 극성값을 살펴보면 TextBolb와 Pattern 라이브러리가 전부 같은 감성 분석 결과를 보이고 있습니다.

극성값으로 표현된 각 열을 원본 데이터의 'sentiment' 열과 같이 ‘Positive’, ‘Neutral’, ‘Negative’의 라벨로 변환해줬습니다. 즉, 극성값은 -1과 1 사이의 값을 가지므로 ‘Neutral’ 라벨에 해당하는 극성값의 절댓값 범위를 설정하여 라벨링을 진행해야 합니다. 아래 그래프는 ‘Neutral’ 라벨의 절댓값 범위를 어떻게 설정했는지에 따른 모델의 성능을 시각화한 결과입니다.

텍스트, 스크린샷, 도표, 그래프이(가) 표시된 사진

자동 생성된 설명

세 모델 모두 ‘Neutral’ 라벨의 범위를 0.1보다 작거나 같게 설정했을 때 가장 높은 성능을 보였습니다. 따라서 ‘Neutral’ 라벨을 해당 범위로 설정하고 전처리 후 성능도 비교해봤습니다.

텍스트, 스크린샷, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

위 그래프는 세 가지 감성분석 모델들의 성능을 전처리 전과 후에 어떻게 달라지는지 시각화한 것입니다.

전처리를 수행한 후 모든 모델의 정확도가 감소했습니다. 이는 전처리 과정에서 필요한 정보가 제거되었기 때문일 가능성이 큽니다. 특히 본 데이터셋은 EDA 과정에서 텍스트의 길이를 확인하니 0부터 350까지의 짧은 글 위주였습니다. 따라서 간단한 전처리도 정보 손실로 이어질 수 있는 상황이었습니다.

Pattern 모델이 전처리 전후를 통틀어 가장 높은 정확도를 보였습니다. 소셜 미디어 텍스트 분석에 특화된 것으로 알려진 VADER보다도 더 나은 성능을 보였다는 점은 다소 의외였습니다.

# Conclusion

본 프로젝트는 TextBlob, VADER, Pattern 세 가지 비지도 학습 기반 감성분석 모델을 사용하여 트위터 데이터의 감성을 분석하고, 이들의 성능을 비교하였습니다. 모델 성능이 좋지 못하게 보고된 만큼 EDA와 전처리에 노력을 기울였습니다.

텍스트 데이터는 모델 학습에 활용하기 위해 임베딩을 통해 모델이 이해할 수 있는 형태로 변환해야 합니다. 따라서 텍스트 데이터는 고차원의 데이터이므로 효율적인 학습을 위해 텍스트 전처리에 많은 심혈을 기울여야 합니다. 하지만 본 프로젝트는 사용한 데이터의 양이 많지 않고 각 텍스트의 길이가 짧은 편이어서 전처리 없이도 모델 학습에 큰 무리가 없었습니다. 이를 통해 모델 성능을 위해 데이터 전처리를 최소화하는 것을 우선 고려할 필요가 있다고 생각하게 되었습니다.

# References

[1] Guo, Y., Hu, C., & Yang, Y. (2023). Predict the Future from the Past? On the Temporal Data Distribution Shift in Financial Sentiment Classifications. *arXiv preprint arXiv:2310.12620*.

[2] Kevin, M. (2024, June 6). Twitter sentiment analysis – Logistic Regression. *Kaggle*. <https://kaggle.com/code/kevinmorgado/twitter-sentiment-analysis-logistic-regression>

[3] *NLTK Documentation*. (2024, June 6). <https://www.nltk.org/_modules/nltk/sentiment/vader.html>

[4] *Pattern.en Official Documentation*. (2024, June 6). <https://digiasset.org/html/pattern-en.html#sentiment>

[5] *TextBlob Github*. (2024, June 6). <https://github.com/sloria/textblob>

1. 한국어로 ‘행동유도성’, 사물이 사람으로 하여금 자신을 어떻게 다룰지 알려주는 속성입니다. 즉, 소셜 미디어 플랫폼 어포던스는 플랫폼이 개인의 표현에 영향을 미치는 속성을 의미합니다. 왜냐하면 플랫폼(SNS)이 유도하는 행동은 의견 표현이기 때문입니다.[1] [↑](#footnote-ref-1)