금융 뉴스 헤드라인 감성분석을 통한 주가 종목 변동성 예측 (NLP Sentiment Analysis)

딥러닝 파이널 프로젝트 5조

고선욱
장현영

A Table of Contents

1 프로젝트 주제 및 데이터 소개

2 데이터 EDA 및 전처리

3 모델링 과정 및 학습 모델 구축

4 학습 모델을 통한 결과 분석

5 한계 및 보완점

6 참고문헌

Part 1,

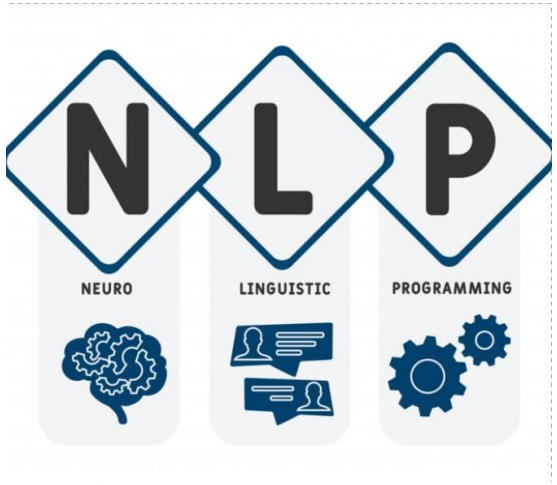
프로젝트 주제 및 데이터 소개

Part 1,

프로젝트 주제 및 데이터 소개

프로젝트 주제

딥러닝 파이널 프로젝트 5조



+

=

ML/DL에서 자연어 처리 기술

음성 인식, 요약, 사용자 감성분석 , 텍스트 분류 작업

자연어 처리의 통계 기반 언어 모델, 언어분류 딥러닝 모델 성장으로 자연어처리의 비약적인 성능 향상

매일 쏟아지는 금융 뉴스 기사

종목의 현재 상황을 가장 직관적으로 파악할 수 있는 뉴스 기사

주가 기사동향이 긍정적인 문장이 많이 있다면 해당 주가가 올랐을까?

뉴스를 보고 주식시장 등락을 알 수 있을까?

가정 : 기자가 작성한 오피니언 어휘는 '주가의 동향을 판단하는 척도로 사용할 수 있다.

헤드라인의 긍/부정이라는 의미는 시간적, 사회적 그룹 내에서 빈번하게 통용되는 문맥의 패턴으로 판단

뉴스 헤드라인 감성분석을 통해 감성 분류와 주가 변동성 분류가 매칭 가능한지 확인

Part 1,

프로젝트 주제 및 데이터 소개

데이터 선정 및 크롤링

딥러닝 파이널 프로젝트 5조

코스피 코스닥

N	종목명	현재가	전일비	등락률	액면가	시가총액
1	삼성전자	71,300	▼ 2,000	-2.73%	100	4,256,455
2	LG에너지솔루션	505,000	▼ 92,000	-15.41%	500	1,181,700
3	SK하이닉스	113,500	▼ 4,000	-3.40%	5,000	826,283
4	삼성전자우	65,600	▼ 1,200	-1.80%	100	539,814
5	NAVER	303,000	▼ 10,000	-3.19%	100	497,069
6	삼성바이오로직스	712,000	▼ 45,000	-5.94%	2,500	471,095
7	LG화학	610,000	▼ 54,000	-8.13%	5,000	430,613
8	삼성SDI	594,000	▼ 39,000	-6.16%	5,000	408,461
9	현대차	186,500	▼ 3,500	-1.84%	5,000	398,491
10	카카오	82,600	▼ 4,300	-4.95%	100	368,318
11	기아	79,100	▲ 1,400	+1.80%	5,000	320,642
12	KB금융	58,000	▼ 1,100	-1.86%	5,000	241,169
13	POSCO	257,500	▼ 12,000	-4.45%	5,000	224,506
14	현대모비스	227,000	▼ 2,000	-0.87%	5,000	214,681
15	셀트리온	147,500	▼ 8,500	-5.45%	1,000	203,472
16	LG전자	124,000	▼ 5,000	-3.88%	5,000	202,923
17	SK이노베이션	215,500	▼ 16,500	-7.11%	5,000	199,263
18	신한지주	38,100	▼ 200	-0.52%	5,000	196,824

뉴스검색

삼성전자 검색

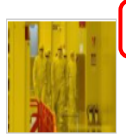
범위 : 제목과 본문 제목에서만 포토/TV뉴스만

기간 : 전체 최근 한달간 최근 일주일간 직접입력 2011-12-31~2021-12-31

"삼성전자"(으)로 검색결과 "1,215,487"개의 기사가 검색되었습니다.

'따상' 실패했어도... LG엔솔, 상장하자마자 시총 2위로

시가총액은 118조1700억원으로 삼성전자(425조6455억원)에 이어 시총 2위로 올라선다. SK하이닉스(82조6283억원)를 밀어냈다. 이날 LG그룹(232조7084억... 조선일보 2022-01-27 23:10



반도체 날개 단 삼성전자… 올해 매출 300조원 시대 열까

지난해 사상 최대 매출을 갈아치운 삼성전자가 올해에도 실적 경신행진을 이어갈 전망이다. 신종 코로나바이러스 감염증(코로나19) 장기화로 비대면 ... 머니S 2022-01-27 22:10

'따상' 기대했던 LG엔솔… 데뷔하는 날 증시 대폭락 '불운'

뉴시스 LG에너지솔루션의 상장 첫날 거래대금은 8조864억원으로 2위 삼성전자(1조5929억원)보다 5배 이상 많았다. 이날 코스피 전체 시장에... 세계일보 2022-01-27 22:02

```
# 크레프톤의 기업정보를 확인하는 예시
raw[raw['Symbol'] == '259960']
```

	Symbol	Market	Name	Sector	Industry	ListingDate	Settle
4237	259960	KOSPI	KRAFTON	소프트웨어 개발 및 공급업	응용 소프트웨어	2021-08-10	

```
# 크레프톤 예시, 기준일을 2018년부터로 설정해도 첫 상장일이
fdr.DataReader('259960', '2018')
```

Date	Open	High	Low	Close	Volume	Change
2021-08-10	448500	480000	400500	454000	5121520	NaN
2021-08-11	444500	446000	405000	407000	1647759	-0.103524
2021-08-12	414000	420500	402000	406000	947958	-0.002457
2021-08-13	415000	445500	408500	437000	1669847	0.076355
2021-08-17	433000	460000	423000	451500	1189500	0.033181
...	...	...	...	...	...	...
2022-01-13	364000	365000	345000	346500	570811	-0.057143
2022-01-14	338500	354000	333500	343500	462339	-0.008658
2022-01-17	351500	354500	338000	341000	298123	-0.007278
2022-01-18	339000	343000	334000	340500	251142	-0.001466
2022-01-19	341000	342500	323000	325500	388273	-0.044053

110 rows × 6 columns

	date	change	name
0	2012-01-02	1	삼성전자
1	2012-01-03	1	삼성전자
2	2012-01-04	0	삼성전자
3	2012-01-05	0	삼성전자
4	2012-01-06	0	삼성전자
...	...	...	...
24725	2022-01-10	1	신한지주
24726	2022-01-11	1	신한지주
24727	2022-01-12	1	신한지주
24728	2022-01-13	0	신한지주
24729	2022-01-14	0	신한지주

시가 총액 상위 10개 종목 확인

기사양이많은종목선정필요

2021년 시가총액 기준 계열사 제외 상위 10개 종목
(삼성전자, 삼성전자우, 삼성바이오로직스)—삼성전자

종목 검색 시 헤드라인, 날짜 크롤링

최근 10년간종목당해당날짜의헤드라인
총 10-20개씩selenium으로크롤링

Trainset: 2012-2019년, Testset: 2020-2021년
(총419444개headline추출)

FinanceDataReader

KRX(한국종목)의특정종목특정일가격정보확인

주식시장 종목명과 일치하기 위해 Symbol (종목 코드)로 접근하여
총 10개종목의데이터를수집 후매핑

프로젝트 주제 및 데이터 소개

데이터 매핑

Score	Review
10	NaN
10	어른이봐도 재밌네요 른 즐귀탱:)
6	영상, 배경음악은 좋았는데, 연출실력은 높게 못사겟네요. 1만년뒤에도? 사람 사는게...
8	사실 내용은 좀 어렵고 재미가 있을랑말랑 있을랑말랑 그런 상태였지만 일단 주인공이 ...
7	이혜영 배우와 홍상수 감독의 만남이 신선했다. 홍감독의 영화는 최근 스타일이 많이 ...

< 네이버 영화 리뷰 - 평점>

부정(-1) : ★~★★★★ (score: 1~4)

중립 (0) : ★★★★★~★★★★★★ (score: 5~6)

긍정(1) : ★★★★★★★~★★★★★★★★★★ (score: 7~10)

	name	headline	date	change
0	삼성전자	[TV] 재계 총수들, 일자리 창출과 투자 확대 강조	2012-01-02	1
1	삼성전자	재계 총수들의 올해 첫 화두는 '위기' '투자' '인재'	2012-01-02	1
2	삼성전자	[경제 굿&노굿] CEO들 신년사로 보는 2012 경제전망	2012-01-02	1
3	삼성전자	재계 빅3 공격적 투자..나머지는 '흐림'	2012-01-02	1
4	삼성전자	오너 3세를 '전진배치' 명과 암	2012-01-02	1
...	...	...	...	...
419439	신한지주	신한 디지털 vs 국민 변화... 카드 리딩금융 승부수	2021-12-30	0
419440	신한지주	순혈주의 이어 유리천장도 깨진다...은행권에 부는 '여풍'	2021-12-30	0
419441	신한지주	"리딩금융 승자, 비은행이 가른다" 신한 vs KB, '보험 전쟁'	2021-12-30	0
419442	신한지주	은행 개발자 몸값 더 높아진다...부서 넘나들며 플랫폼 특명 완수	2021-12-30	0
419443	신한지주	배당락에 3000선 내줬지만..."코스피 하락폭 예상보다 작았다"	2021-12-30	0

< 헤드라인- 종목 주가 등락 mapping >

라벨링

감성 분석 모델을 사용할 예정이기에 평점으로 긍정/부정 지표를 추출할 수 있는 영화 리뷰 데이터와 다르게 헤드라인에 대한 객관적 라벨링 지표 선정에 어려움

종목 검색 시 해당 일자 종목의 등락을 0, 1로 판단하고, 해당 날짜의 기사를 등락으로 라벨링 진행

Part 2,

데이터 EDA 및 전처리

데이터 EDA 및 전처리

텍스트 1차 전처리 및 EDA


```
# 중복으로 제거할 모든 rows의 index 확인 후 제거

temp = df.duplicated(subset='headline',keep=False)

df_duplicate = df.loc[temp,:]
erase_idx = df_duplicate.index.tolist()
erase_idx[:5], len(erase_idx)
```

```
([9, 10, 11, 12, 13], 174725)
```

```
df = df.drop(index=erase_idx)
df.reset_index(inplace=True)
del df['index']
df
```

	name	headline	date	change
0	삼성전자	[TV] 재계 총수들, 일자리 창출과 투자 확대 강조	2012-01-02	1
1	셀트리온	[시황]상승동력 부재... 코스피, 하락 전환	2012-01-02	0
2	셀트리온	코스닥, 새해 첫 거래일 소폭 오름세 지속	2012-01-02	0
3	셀트리온	코스닥 새해 첫 거래일 소폭 오름세 지속	2012-01-02	0
4	셀트리온	[특징주]차바이오,알앤엘 등 바이오株, 연초 '강세'	2012-01-02	0
...	...	...	...	...
244714	SK하이닉스	"코로나 봉쇄로 삼성 시안공장 생산 줄여...메모리 공급사 주가에는 긍정적 영향"	2021-12-30	1
244715	SK하이닉스	SK하이닉스, 인텔 낸드 1단계 인수	2021-12-30	1
244716	SK하이닉스	SK하이닉스, 특별성과급 300% 쏜다	2021-12-30	1
244717	셀트리온	삼성바이오로직스-셀트리온, 바이오 대장주 희비	2021-12-30	0
244718	신한지주	배당락에 3000선 내줬지만..."코스피 하락폭 예상보다 작았다"	2021-12-30	0

244719 rows × 4 columns

< 헤드라인 중복 제거>

1. 동일 헤드라인 중복 제거

전체 42만여개 데이터 중 동일한 헤드라인의 라벨링 차이가 발생

EX) "리딩금융 승자, 비은행이 가른다" 신한 vs KB 보험 전쟁-신한지주 1, KB금융 0

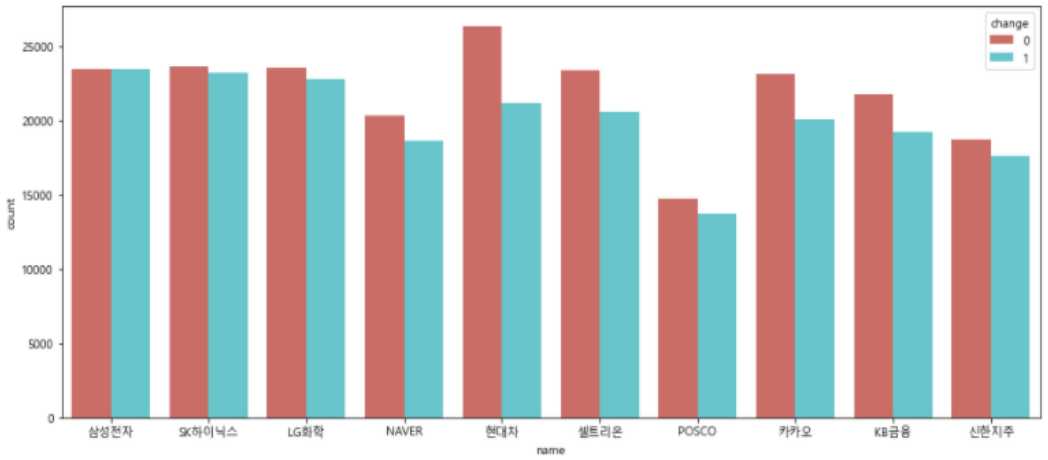
중복이 된 모든 데이터 제거 (17만 여개) - 노이즈 제거

Part 2,

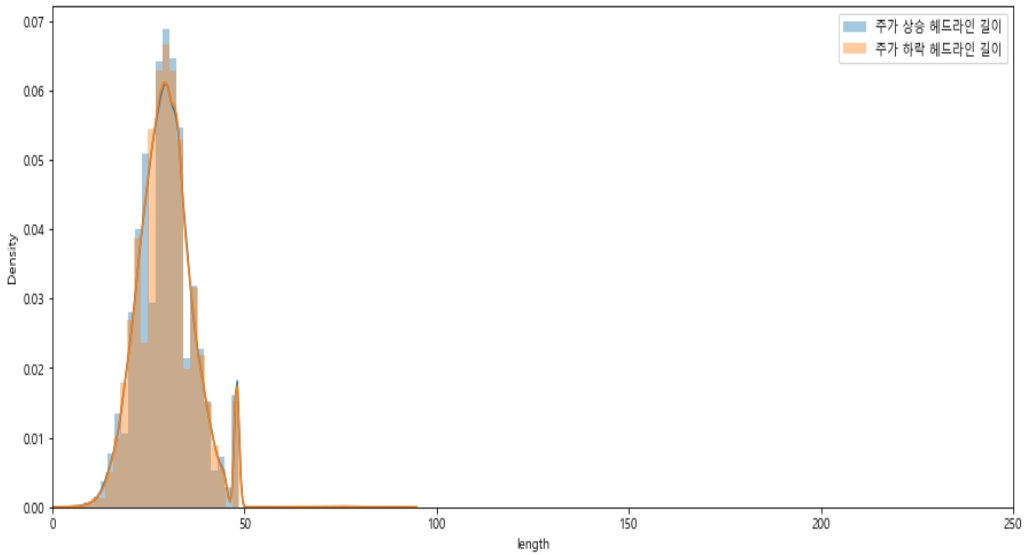
데이터 EDA 및 전처리

텍스트 1차 전처리 및 EDA

딥러닝 파이널 프로젝트 5조



< 종목 별 상승/ 하락 헤드라인 개수>



< 상승/ 하락 헤드라인 길이 분포>

change	0	1
date		
2012	26.137991	26.144073
2013	27.128090	27.225518
2014	27.998248	27.953421
2015	28.483687	28.649886
2016	29.868026	29.809824
2017	30.495178	30.486866
2018	30.998732	30.878947
2019	31.381180	31.395573
2020	31.576210	31.450013
2021	32.292991	32.066119

< 연도별 상승/하락 헤드라인 길이 >

name	KB금융	LG화학	NAVER	POSCO
date				
2012	26.556716	26.644502	23.911111	26.817862
2013	26.854575	27.803517	25.790628	27.439474
2014	27.859060	28.185903	28.672190	28.216119
2015	28.122522	28.780409	29.150383	28.591638
2016	29.372237	30.002291	30.534836	30.262664
2017	29.759450	30.988383	31.661284	31.968229
2018	31.504898	32.066667	31.128459	32.611760
2019	30.295023	31.946481	32.046512	32.479913
2020	31.275727	31.934245	32.127133	32.715000
2021	31.407027	32.487236	33.877025	33.082335

< 연도별 종목당 헤드라인 길이 >

2. 헤드라인 - 주가 변동성 분류 연관성 찾기

종목별, 연도별, 종목 연도 별 뉴스 헤드라인 개수, 길이 확인

헤드라인의 개수와 길이 패턴만으로는 상승/하락 패턴 확인이 어려움

언어적 특성을 텍스트로 직접 확인하면서 노이즈로 판단되는 요소 제거 필요

데이터 EDA 및 전처리

텍스트 1차 전처리

	name	headline	date	change
36973	POSCO	[생톡관심종목] 김성남전문가 - 포스코ict	2013-06-24	0
38409	POSCO	[생톡관심종목] 김성남전문가 - 포스코ICT	2013-07-11	1
39446	카카오	[생톡관심종목] 차대웅 전문가 - 영우통신(051390)	2013-07-24	0
39822	SK하이닉스	[생톡관심종목] 이지한 전문가 - 아이테스트(089530)	2013-07-30	1
41065	POSCO	[생톡관심종목] 전희승 전문가 - DSR(155660)	2013-08-16	0
41411	삼성전자	[생톡관심종목] 김두호 전문가 - 아이테스트(089530)	2013-08-22	0
46060	현대차	[생톡관심종목] 황병우 전문가 - 한일이화(007860)	2013-11-01	1
47355	현대차	[생톡관심종목] 황병우 전문가 - 현대위아(011210)	2013-11-21	0
485	현대차	[사진]'2012 북미 올해의 차'에 선정된 '아반떼'	2012-01-09	1
1374	삼성전자	[사진]호텔신라, 커피-베이커리 사업철수	2012-01-26	0

< 노이즈라고 판단한 삭제 키워드 헤드라인 일부>

	name	headline	date	change
244649	셀트리온	[마감시황] 올해 마지막날 코스피 2977.65 마감, 코스닥 1000선 지켜	2021-12-30	0
244703	LG화학	[마감시황] 코스피 2977.65로 올해 증시 마감	2021-12-30	0

< 현 상황에 대한 중계 알려주는 시황 키워드 일부>

3. 특정 종목의 상승/하락과 무관하다고 판단되는 헤드라인 제거

- 전체 헤드라인 중 특정 키워드로 시작하는 헤드라인 제거

- 삭제 키워드

**[생톡관심종목], [사진], [포토], [부고], [인사], [부음],
<표>, [프로필], [오늘의 메모], [집중관찰], <오늘의 화제주>,
[특징주], [게임소개], [업&다운]**

전체 헤드라인에서 3000여개 정도 삭제

+ 해당일의 코스피 가격,시장을 말그대로 중계해주는 키워드인 시황 키워드도 노이즈로 판단 8000여개 정도 삭제 (n-gram 기준 삭제)

데이터 EDA 및 전처리

텍스트 2차 전처리


```
def apply_regular_expression(headline):
  # 한글 추출 규칙 : 띄어쓰기(1 개)를 포함한 한글과 대문자, ... …는 반드시 공백으로 처리해야 띄어쓰기 유지
  headline = headline.replace("…"," ") # 구두점으로 표기하고 띄어쓰기가 반영이 되지 않은 것 처리
  headline = headline.replace(".."," ") # 구두점으로 표기하고 띄어쓰기가 반영이 되지 않은 것 처리
  headline = headline.replace("·"," ") # 구두점으로 표기하고 띄어쓰기가 반영이 되지 않은 것 처리

  # 국가나 기관등을 한자로 표기하는 경우가 많아 전처리 진행
  headline = headline.replace("中企","중소기업") # 중국과 혼동 피하기 위해 먼저 전처리(중국과 구분짓기 위함)
  headline = headline.replace("中","중국")
  headline = headline.replace("日","일본")
  headline = headline.replace("韓","한국")
  headline = headline.replace("美","미국")
  headline = headline.replace("銀","은행")
  headline = headline.replace('株','종목') # 많이 쓰이는 한자
  # 대통령을 한자로 표기하는 경우가 많아 전처리 진행
  headline = headline.replace('文','문재인')
  headline = headline.replace('朴','박근혜')

  headline = headline.replace("↓"," 하락")
  headline = headline.replace("↑", "상승")

  headline = headline.replace("미리보는","") # 미리보는 삭제
  headline = re.sub("₩s₩s+" , " ", headline) # 다중 공백 제거

  hangeul = re.compile('[^ ㄱ-ㅣ 가-힣a-zA-Z]') # 숫자는 배제!
  result = hangeul.sub('', headline)
  return result
```

4. 1차 전처리 후 한글, 영어 띄어쓰기를 제외한 모든 표현 삭제

예외 처리 기준

- 구두점

"공모가 10% 프리미엄에도 청약경쟁률 663대 1…POSCO 관련종목 흥행 성공"

- 한자

"韓中 무역갈등 현실화? 원자재 수급 비상"

"文,삼성전자 반도체 공장 방문 이재용 부회장과 대면"

"한국銀,사중금리 인상고려"

- 영어

"SK하이닉스OCI머리티얼스M&A협약… 반도체 역량 강화 중점"
영어가 포함된헤드라인은 M&A, R&D, GDP 등주요경제용어가다수포함되어있어 전처리대상제외

- 숫자

경제관련기사에숫자는헤드라인논조에중요한역할이라고판단
단위,시간등에따라다른숫자를명확한기준으로전처리할수없기에제거

- 특수문자

"외국인대량매도에바이오관련주일제히 ↓"
↓ , ↑ 와 같은 경우, 특수문자 자체로도 의미가 형성되므로 하락과 상승으로 전처리

- 기타

헤드라인에미리보는,이데일리등상승/하락과무관한지표제거
다중공백제거

데이터 EDA 및 전처리

텍스트 3차 전처리(형태소 분석기)

Okt(Twitter)

오픈 소스 한국어 분석기

띄어쓰기 강점, 어근화

이모티콘, 비표준어 등
비정제 데이터 강점

한계: 미등록어 처리, 사용가능한 형태소 적음

KoNLPy

- 한국어 단어(형태소) 토큰화 작업

- 띄어쓰기 단위의 단어 분석이 아닌 형태소 분석으로 문맥 파악 가능

Time analysis [1]

1. 로딩 시간: 사전 로딩을 포함하여 클래스를 로딩하는 시간.
   - Kkma: 5.6988 secs
   - Komoran: 5.4866 secs
   - Hannanum: 0.6591 secs
   - Twitter: 1.4870 secs
   - Mecab: 0.0007 secs
2. 실행시간: 10만 문자의 문서를 대상으로 각 클래스의 pos 메소드를 실행하는데 소요되는 시간.
   - Kkma: 35.7163 secs
   - Komoran: 25.6008 secs
   - Hannanum: 8.8251 secs
   - Twitter: 2.4714 secs
   - Mecab: 0.2838 secs

Mecab

오픈 소스 형태소 분석 엔진 MeCab

말뭉치 학습과 사전 목록은 21세기 세종계획의 성과물을 사용

은전한닢 프로젝트 사전 구축

가장 빠른 로딩/ 형태소 분류 처리

한계 : 미등록어 처리 , 동음이의어 처리

출처: https://konlpy.org/ko/v0.4.3/morph/

데이터 EDA 및 전처리

텍스트 3차 전처리(형태소 분석기)

Okt(Twitter)

```
okt.pos('SK공모가 프리미엄에도 청약경쟁률 대 흥행 성공')
```

```
[('SK', 'Alpha'),
 ('공모', 'Noun'),
 ('가', 'Josa'),
 ('프리미엄', 'Noun'),
 ('에도', 'Josa'),
 ('청약', 'Noun'),
 ('경쟁률', 'Noun'),
 ('대', 'Verb'),
 ('흥행', 'Noun'),
 ('성공', 'Noun')]
```

Mecab으로 형태소 분석 진행

Okt와 mecab 비교할 때 더 상세한 형태소 분석 가능
(mecab : 세종 품사 태그 기준)

본 프로젝트 : 정제된 단어로 사용된 헤드라인을 데이터로 사용

성능/ 속도면에서 빠른 mecab을 기준으로 token화 진행

Mecab

```
mecab.pos('SK공모가 프리미엄에도 청약경쟁률 대 흥행 성공')
```

```
[('SK', 'SL'),
 ('공모', 'NNG'),
 ('가', 'JKS'),
 ('프리미엄', 'NNG'),
 ('에', 'JKB'),
 ('도', 'JX'),
 ('청약', 'NNG'),
 ('경쟁', 'NNG'),
 ('률', 'XSN'),
 ('대', 'XPN'),
 ('흥행', 'NNG'),
 ('성공', 'NNG')]
```

출처: https://konlpy.org/ko/v0.4.3/morph/

데이터 EDA 및 전처리

텍스트 3차 전처리(형태소 분석기)

실질의미유무	대분류(5언 + 기타)	세종 품사 태그		mecab-ko-dic 품사 태그	
		태그	설명	태그	설명
실질형태소	체언	NNG	일반 명사	NNG	일반 명사
		NNP	고유 명사	NNP	고유 명사
		NNB	의존 명사	NNB	의존 명사
				NNBC	단위를 나타내는 명사
		NR	수사	NR	수사
		NP	대명사	NP	대명사
	용언	VV	동사	VV	동사
		VA	형용사	VA	형용사
		VX	보조 용언	VX	보조 용언
		VCP	긍정 지정사	VCP	긍정 지정사
		VCN	부정 지정사	VCN	부정 지정사
	수식언	MM	관형사	MM	관형사
		MAG	일반 부사	MAG	일반 부사
		MAJ	접속 부사	MAJ	접속 부사
	독립언	IC	감탄사	IC	감탄사

	name	headline	date	change	token_mecab
244698	LG화학	개미 울리는 물적분할에 고승범 "법적 측면 포함 대책 검토"	2021-12-30	0	개미 울리 물적분할 고승범 측면 포함 대책 검토
244699	LG화학	올해 증시 마지막 날 코스피 2970선으로 마감…한 해 동안 3.63% 상승	2021-12-30	0	올해 증시 마지막 코스피 마감 동안 상승
244700	LG화학	코스피 마지막 거래일 2,970선 마감..삼성전자 이틀째 하락 흐름	2021-12-30	0	코스피 마지막 거래일 마감 삼성전자 이틀 하락 흐름
244701	LG화학	3000선 끝내 못 지킨 코스피…올해 2977.65 마감	2021-12-30	0	끝내 코스피 올해 마감
244702	LG화학	JW중외제약, 아토피신약 개발 박차…유한양행·LG화학도 경쟁	2021-12-30	0	JW 중외 제약 아토피 신약 개발 박차 유한양행 LG 화학 경쟁
244704	LG화학	악재 덮친 LG화학, 주가 뚝뚝…"50만원대 추락" 최악 시나리오까지	2021-12-30	0	악재 LG 화학 뚝뚝 추락 최악 시나리오
244705	LG화학	금융위원장 "물적분할 규제 검토, 인터넷은행 대출 상한도 업권 특성 고려할 것"	2021-12-30	0	금융 위원장 물적분할 규제 검토 인터넷 은행 대출 상한 특성 고려

Mecab-ko-dic품사태그

- 체언 : 일반명사, 고유명사, 의존명사, 수사, 대명사
  - ['NNG', 'NNP', 'NNB', 'NR', 'NP']
- 용언 : 동사, 형용사, 긍정지정사, 부정지정사
  - ['VV', 'VA', 'VCP', 'VCN']
- 수식언 : 관형사, 일반부사
  - ['MM', 'MAG']
- 외국어: 영어만 포함
  - ['SL']

※ 한 글자 token은 제거

Part 2,

데이터 EDA 및 전처리

텍스트 3차 전처리(형태소 분석기)

딥러닝 파이널 프로젝트 5조

```
print(len(df[df.token_mecab.isnull()]))
df[df.token_mecab.isnull()].tail(20)
```

31

	name	headline	date	change	token_mecab
50794	카카오	돈으로 사야만 가질 수 있나요?	2014-03-11	0	NaN
51151	KB금융	V	2014-03-17	0	NaN
55491	NAVER	수공 "2017년까지 빚 36% 줄일 것"	2014-05-26	0	NaN
56381	현대차	원高, 피할 수 없다면 즐겨볼까	2014-06-10	0	NaN
62936	NAVER	비슷하면 지는 거다	2014-09-17	1	NaN
80646	LG화학	한경硏 "엔저 2~3년 더 간다"	2015-05-29	0	NaN
81929	KB금융	돈? 안 생겨요, 돈 있어야 돈 버는걸요	2015-06-16	0	NaN
84305	NAVER	일, 삶의 빛인가 짐인가	2015-07-20	1	NaN
93929	현대차	'더 깨끗하고 더 똑똑한' 차들이 온다	2015-11-19	1	NaN
100476	카카오	살(買)것인가, 살(居)것인가	2016-02-25	0	NaN
102948	삼성전자	G5, 53만~57만원이면 산다	2016-03-31	1	NaN
105790	현대차	50% 싸게	2016-05-13	0	NaN
113470	삼성전자	'잘 해봅시다'	2016-09-01	0	NaN
113610	KB금융	'신기하네'	2016-09-02	0	NaN
164197	KB금융	번돈 30% 빚 갚는데 쓴다	2018-11-16	0	NaN
173949	LG화학	"죄송합니다"	2019-04-22	1	NaN
175957	NAVER	와!신기하다...	2019-05-23	0	NaN
175986	NAVER	와!이겼다...	2019-05-23	0	NaN
200071	NAVER	먹음직스런 갓 구워낸 빵	2020-07-17	1	NaN
215771	삼성전자	'꽃혀야 산다'	2021-03-24	0	NaN

Token_mecab null값 제거

전체헤드라인중 token_mecab으로잡히지않은 31개의데이터제거

헤드라인이짧거나종목의등락과관련없는기사로판단

V

서울경제

기사입력 2014-03-17 10:22 기사원문

공감 0

▷ DGB금융지주(139130) : (주)대구은행, (주)DGB캐피탈, 유페이먼트(주), 대구신용정보(주), (주)DGB데이터시스템 등을 자회사로 보유한 금융지주회사. 주 수익원은 자회사의 배당수익임. 은행 섹터 : 신한지주, KB금융, 우리금융, 하나금융지주, 기업은행, BS금융지주, JB금융지주

▷ LG디스플레이(034220) : LG그룹 계열의 TFT-LCD 패널 제조 전문업체. TFT-LCD, LTPS-LCD 및 OLED 등의 기술을 활용한 LCD 및 OLED관련 제품을 제조, 판매하고 있음. 스마트폰 섹터 : 바른전자, 토비스, 이엘케이, 멜파스, 시노펙스, 크루셜텍

와!이겼다...

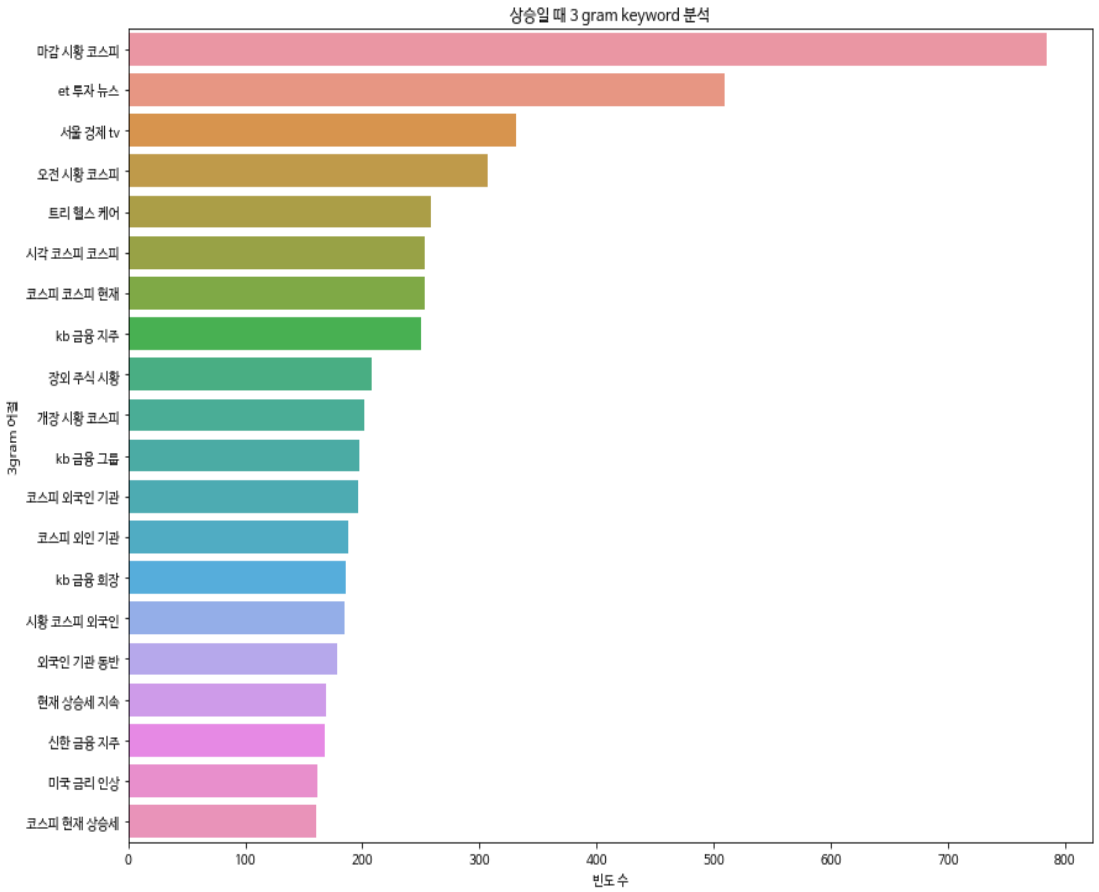
NEWSIS

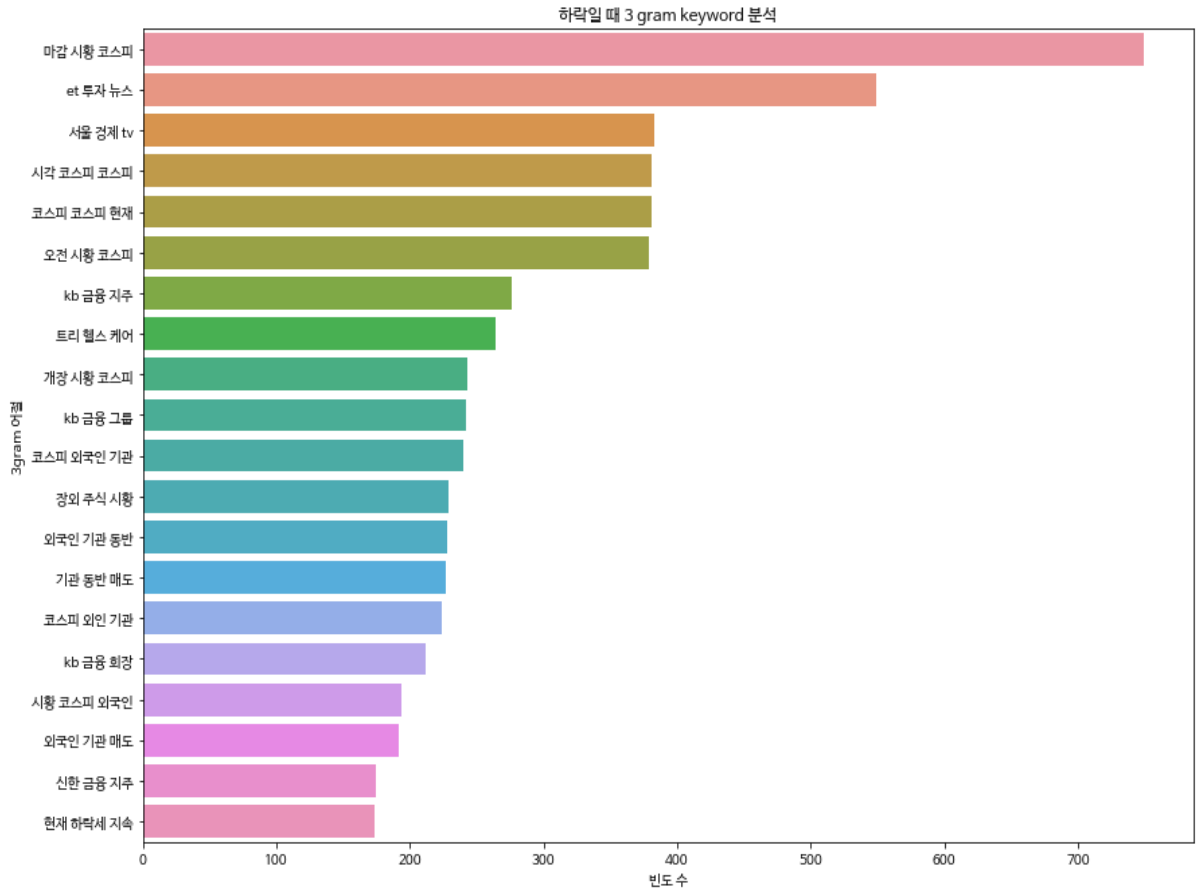
기사입력 2019-05-23 12:58 기사원문

공감 0

데이터 EDA 및 전처리

텍스트 3차 전처리(형태소 분석기)



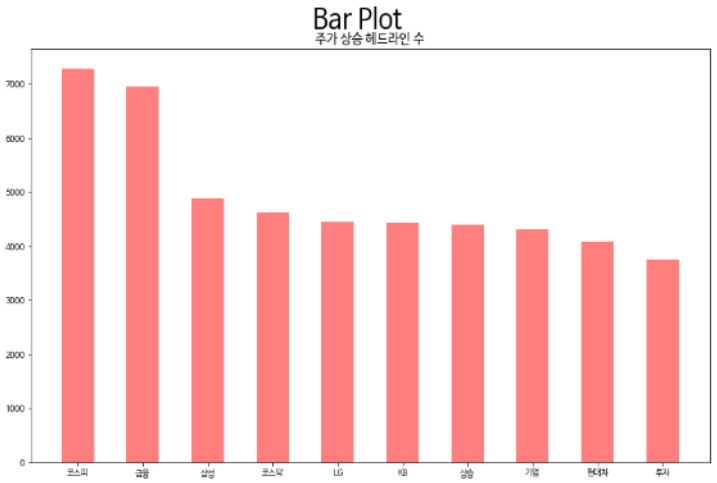


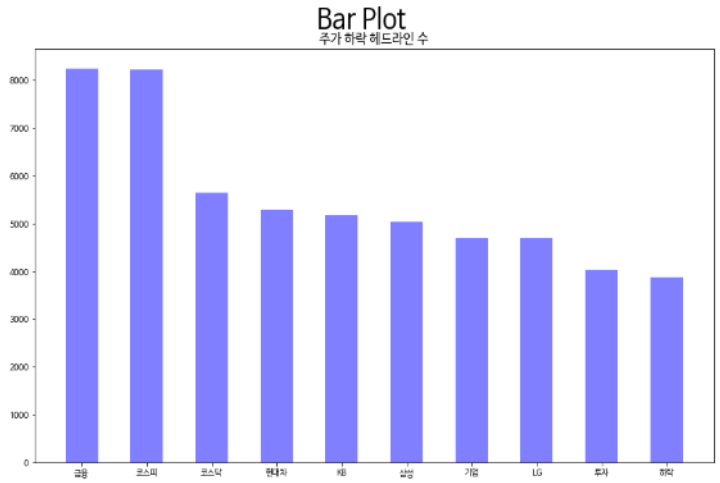
N-gram으로 조합 확인, 상승/하락 중립 단어

시황이라는 키워드가 포함된 헤드라인은 말그대로 현재 종목가의 상황을 중계하는 것으로 노이즈라 판단,
시황이 포함된 헤드라인 모두 제거 (8000여개)

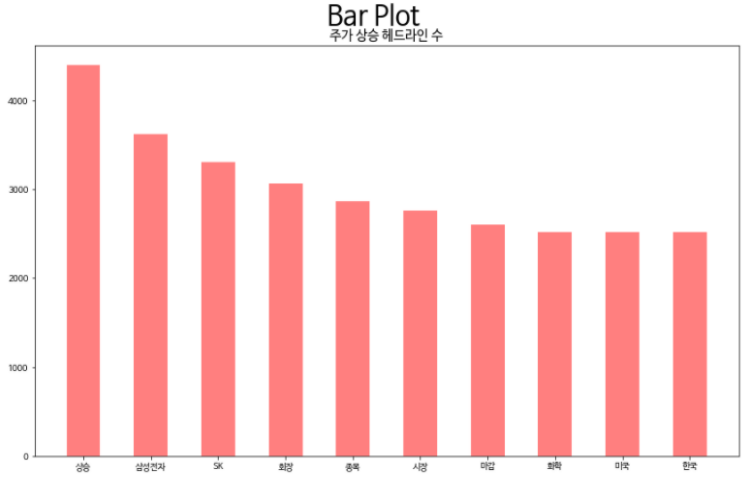
데이터 EDA 및 전처리

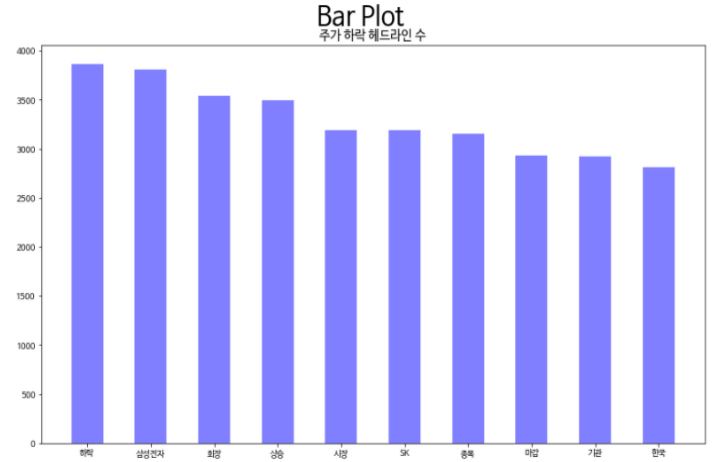
텍스트 4차 전처리





1st Common words: ['코스피', '금융', '삼성', '코스닥', 'LG', 'KB', '기업', '현대차', '투자']





2nd Common words: ['삼성전자', 'SK', '종목', '시장', '마감', '한국']

단어 등장 빈도 EDA 후 전처리

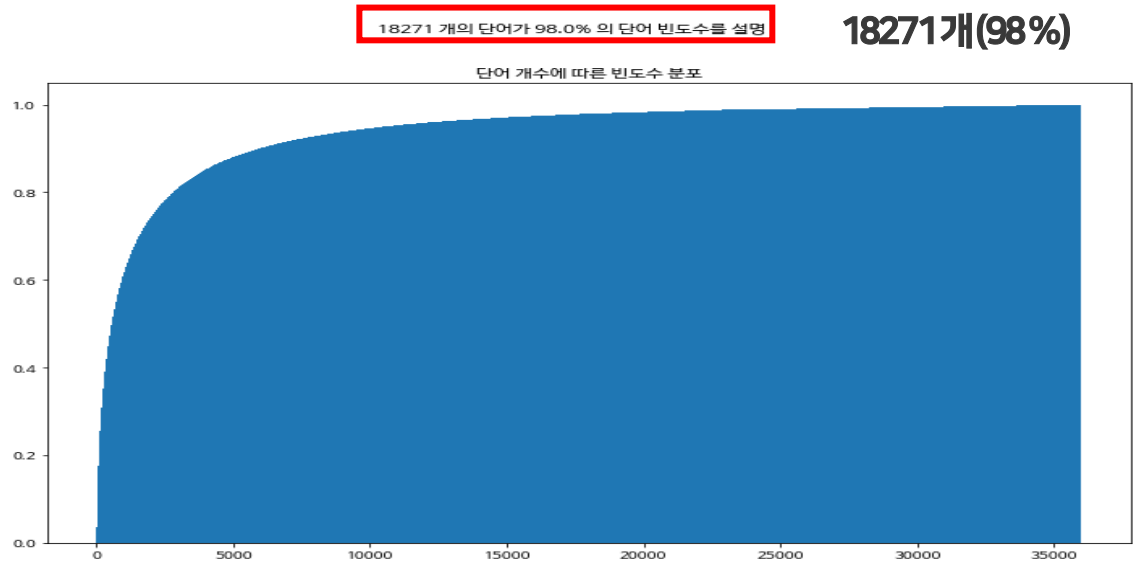
상승/하락 헤드라인의 토큰 빈도수를 확인

상승 상위 10, 하락 상위 10 단어 중 겹치는 단어를 중립어라고 판단

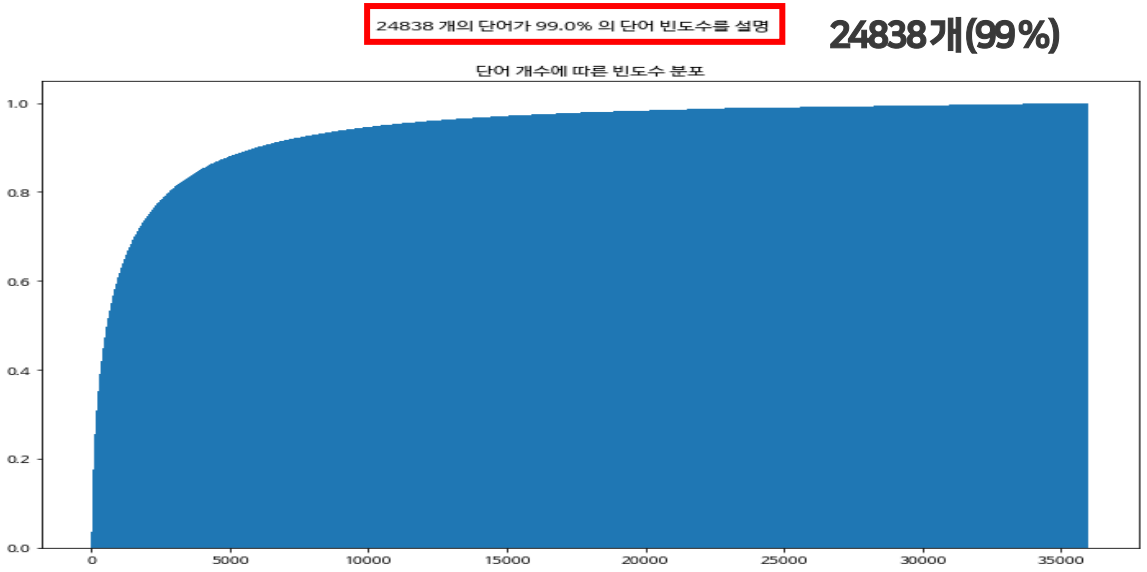
2번에 걸쳐서 총 16개 단어는 token_mecab에서 제거 후 사용

데이터 EDA 및 전처리

텍스트 4차 전처리



18271개(98%)



24838개(99%)

희귀 단어 전처리

전체 사용 단어 35000여개 중 빈도 누적합을 통해 98%, 99%를 설명할 수 있는 단어 수 찾기

전체 사용 단어 중 약 18000개~24800여개의 단어 선택을 위해 희귀단어 확인하기

데이터 EDA 및 전처리

텍스트 4차 전처리


```
단어 집합(vocabulary)의 크기 : 35738
등장 빈도가 1번 이하인 희귀 단어의 수: 12942
단어 집합에서 희귀 단어의 비율: 36.213554200011195
전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 1.1593543376353006
------------------------------
단어 집합(vocabulary)의 크기 : 35738
등장 빈도가 2번 이하인 희귀 단어의 수: 17761
단어 집합에서 희귀 단어의 비율: 49.697800660361516
전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 2.022733808051699
------------------------------
단어 집합(vocabulary)의 크기 : 35738
등장 빈도가 3번 이하인 희귀 단어의 수: 20355
단어 집합에서 희귀 단어의 비율: 56.95618109575242
전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 2.7198513675848397
------------------------------
단어 집합(vocabulary)의 크기 : 35738
등장 빈도가 4번 이하인 희귀 단어의 수: 22149
단어 집합에서 희귀 단어의 비율: 61.97604790419161
전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 3.3626829799222615
------------------------------
```

희귀 단어 전처리

어휘 집합 수를 약 2만개정도 사용한다고 가정하고, 최소 등장빈도가 2 이상이면 토큰 손실우려

등장 빈도가 1회 이하인 토큰들만 정수 인코딩 과정에서 배제하도록 설계

전체 비중에서 1.15% 등장하는 빈도를 보임

희귀 단어 전처리 후 단어 집합의 크기는 **22797개**

데이터 EDA 및 전처리

텍스트 4차 전처리


```
# 길이가 0인 샘플들의 인덱스 반환
drop_train = [index for index, sentence in enumerate(X_train) if len(sentence) < 1]
drop_test = [index for index, sentence in enumerate(X_test) if len(sentence) < 1]
print(f'빈 train 개수 : {len(drop_train)}')
print(f'빈 test 개수 : {len(drop_test)}')
```

```
빈 train 개수 : 115
빈 test 개수 : 31
```

train, test 모두 빈 샘플들을 제거한다.

```
# 빈 샘플들을 제거
X_train = np.delete(X_train, drop_train, axis=0)
y_train = np.delete(y_train, drop_train, axis=0)
X_test = np.delete(X_test, drop_test, axis=0)
y_test = np.delete(y_test, drop_test, axis=0)
len(X_train), len(X_test)
```

```
(188386, 45141)
```

최종 Null 데이터 전처리

약 12000개의 토큰 삭제로 희귀 토큰들로만 구성된 헤드라인은 결측치로 변경됨

Train, test의 115, 31개의 결측치 샘플을 제거

최종 train과 test 개수: 188386, 45141개

Part 3,

모델링 과정 및 학습 모델 구축

Part 3,

모델링 과정 및 학습 모델 구축

학습 모델 종류

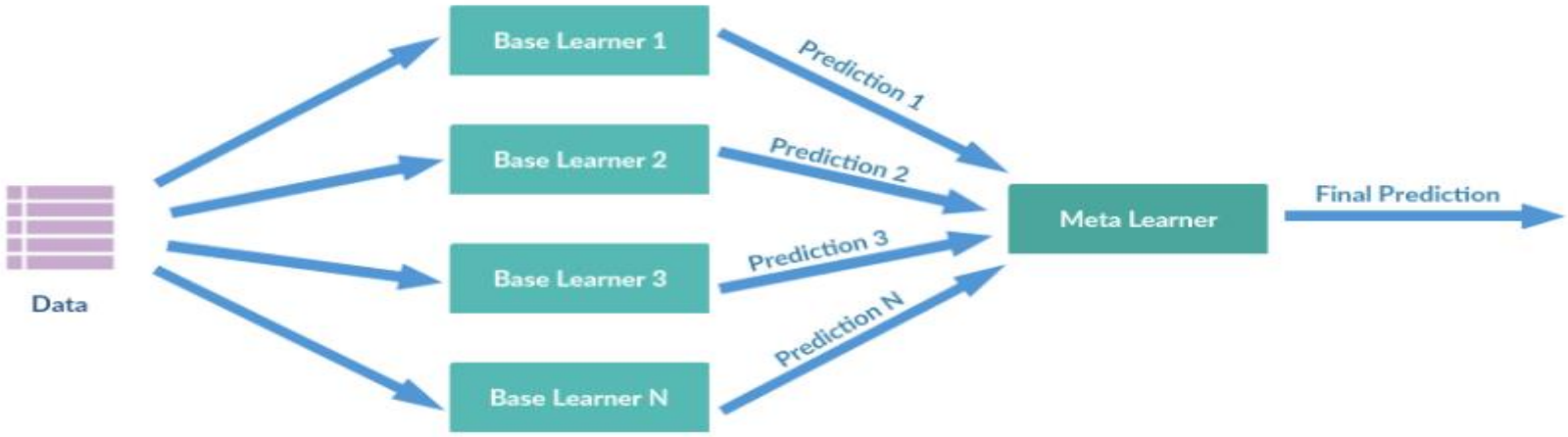
학습 모델 종류

1	2	3	4	5
Stacking Ensemble (Naïve Bayes) (SGD) (LGBM) (XGB)	RNN (LSTM)	RNN (GRU)	Hugging Face (BERT)	SKT Brain (KoBERT)

모델링 과정 및 학습 모델 구축

TF-IDF Stacking Ensemble (Naïve Bayes, SGD, LGBM, XGB)

모델 선정 배경



- 지도 학습을 통한 텍스트(감정) 분류 모델 선정
- Naïve Bayes: 데이터셋의 모든 특징들이 독립적이며, 학습을 많이 시킬수록 분류 능력이 향상되는 장점
- SGD: 랜덤하게 추출된 일부 데이터를 사용하여 학습이 빠르며, 텍스트 분류에서 자주 발생하는 대규모 및 희소 기계 학습 문제에 좋음
- LGBM: 적은 메모리를 사용, 빠른 모델 생성, 병렬 처리를 통해 매우 빠른 속도로 학습하면서 높은 성능을 보여줌
- XGB: LGBM의 Leaf-wise 방식의 단점인 과적합에 취약하다는 점, 데이터 수가 적음으로 인한 과적합 발생으로 인해 비교 선정

모델링 과정 및 학습 모델 구축

TF-IDF Stacking Ensemble (Naïve Bayes, SGD, LGBM, XGB)

TF-IDF Encoding

```
def get_pipe(model, model_name: str) -> Pipeline:
    "TfidfVectorizer와 모델을 연결한 파이프라인을 반환하는 함수"
    tfidf = TfidfVectorizer(analyzer="word", ngram_range=(1, 3))
    pipe = Pipeline([
        ("tfidf", tfidf),
        (model_name, model)
    ])
    return pipe
```

- 단어의 빈도와 역문서 빈도를 사용하여 DTM 내의 각 단어들마다 중요한 정도를 가중치로 주는 TF-IDF encoding 진행
- 상대적으로 긍/부정 용어가 많은 리뷰 텍스트와 다르게 주식 종목 관련 기사이므로 관련 전문 용어가 많이 포함되있음
- analyzer를 word로 설정하여 학습의 단위를 단어로 함
- 앞에서 EDA를 통해 상승/하락 n_gram을 확인했을 때 3_gram정도 묶였을 때 각 라벨별로 의미를 가지는 단어들을 볼 수 있었음
- ngram_range를 (1, 3)으로 설정

모델링 과정 및 학습 모델 구축

TF-IDF Stacking Ensemble (Naïve Bayes, SGD, LGBM, XGB)

Classifier(Naïve Bayes, SGD, LGBM, XGB) 학습

```python
def get_pipe(model, model_name: str) -> Pipeline:
    "TfidfVectorizer와 모델을 연결한 파이프라인을 반환하는 함수"
    tfidf = TfidfVectorizer(analyzer="word", ngram_range=(1, 3))
    pipe = Pipeline([
        ("tfidf", tfidf),
        (model_name, model)
    ])
    return pipe
```

```python
def return_kfold_accuarcy(model, k: int = 5) -> float:
    "모델을 입력받아 KFold 예측 후 accuracy score를 반환하는 함수"
    kfold = StratifiedKFold(k, shuffle=True, random_state=42)
    result = []
    cnt = 0
    for train_idx, test_idx in kfold.split(train["token_mecab"], train["change"]):
        train_new, val = train.iloc[train_idx], train.iloc[test_idx]
        model.fit(train_new["token_mecab"], train_new["change"])
        pred = model.predict(val["token_mecab"])
        acc = accuracy_score(val["change"], pred)
        result.append(acc)
        cnt += 1
        print(f'{cnt}th complete!')

    return np.mean(result)
```

```python
models = [
    ("naive_bayes", BernoulliNB()),
    ("SGD", SGDClassifier(random_state=42, n_jobs=-1)),
    ("lgbm", LGBMClassifier(random_state=42)),
    ("xgb", XGBClassifier(random_state=42))
]

model_pipes = [(name, get_pipe(model, name)) for name, model in models]
```

```python
table = Table(title="Model Comparison Table")
table.add_column("Model Name", justify="left", style="green")
table.add_column("Accuracy", justify="right")

for model_name, model in tqdm(model_pipes, leave=False):
    print(f'###################### {model_name} ###################### ')
    acc = return_kfold_accuarcy(model)
    table.add_row(model_name, f"{acc:0.3f}")

rich.print(table)
```

```
  0%|          | 0/4 [00:00<?, ?it/s]
###################### naive_bayes ######################
1th complete!
2th complete!
3th complete!
4th complete!
5th complete!
###################### SGD ######################
1th complete!
2th complete!
3th complete!
4th complete!
5th complete!
###################### lgbm ######################
1th complete!
2th complete!
3th complete!
4th complete!
5th complete!
###################### xgb ######################
1th complete!
2th complete!
3th complete!
4th complete!
5th complete!
  Model Comparison Table
```

Model Name	Accuracy
naive_bayes	0.597
SGD	0.542
lgbm	0.553
xgb	0.544

- TF-IDF Encoding에 이어 각 분류기를 통해 학습하는 함수를 묶어 모델 파이프라인을 구성

- 각 분류기를 통해 train set을 학습 후 모델 별 accuracy를 추출하여 비교해 봄

모델링 과정 및 학습 모델 구축

TF-IDF Stacking Ensemble (Naïve Bayes, SGD, LGBM, XGB)

Stacking Ensemble

```
from sklearn.ensemble import StackingClassifier

stack_models = [(name, get_pipe(model, name)) for name, model in models]

stacking = StackingClassifier(stack_models)
acc = return_kfold_accuarcy(stacking)
rich.print(acc)
```

```
1th complete!
2th complete!
3th complete!
4th complete!
5th complete!
0.5492119465482872
```

```
stacking.fit(train['token_mecab'], train['change'])
```

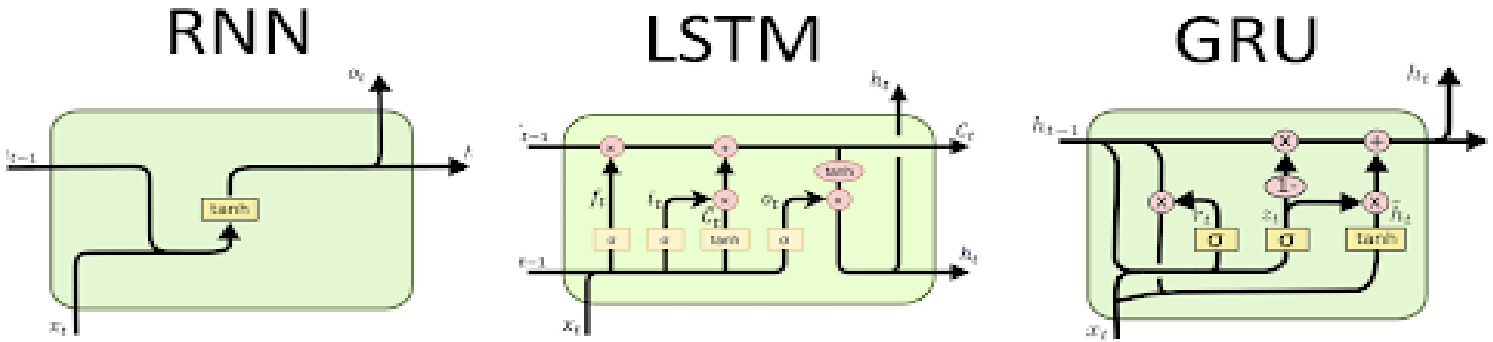
- 4개의 모델을 각각 train set으로 학습한 후 나온 4개의 예측 값을 학습데이터로 사용해 accuracy 추출

- stacking된 최종 model을 다시 train set으로 학습

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

모델 선정 배경



- 텍스트를 처리할 수 있는 딥러닝 RNN 모델 선정 (감성분석)
- SimpleRNN은 실전에 쓰기에는 너무 단순 (긴 시간에 걸친 의존성은 학습할 수 없음)
- 층이 많은 일반 네트워크에서 나타나는 것과 비슷한 현상인 기울기 소실 문제
- Tensorflow keras의 LSTM, GRU 모델 사용

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

정수 인코딩

```
# 빈도 확인해보기
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train)

threshold = 2
total_cnt, rare_cnt = check_frequency(threshold, tokenizer)
vocab_size = total_cnt - rare_cnt + 1
print('단어 집합의 크기 :',vocab_size)

# tokenizer
myTokenizer = Tokenizer(vocab_size)
myTokenizer.fit_on_texts(X_train)

# 정수 sequences로 encoding
X_train = myTokenizer.texts_to_sequences(X_train)
X_test = myTokenizer.texts_to_sequences(X_test)

# check_data
print(X_train[0])
print(X_test[0])
```

```
단어 집합(vocabulary)의 크기 : 35737
등장 빈도가 1번 이하인 희귀 단어의 수: 12942
단어 집합에서 희귀 단어의 비율: 36.2145675350477706
전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 1.15936576189196 44
단어 집합의 크기 : 22796
[49, 208, 350, 224, 682, 39, 1273]
[57, 227, 7466, 2866]
```

- 전처리된 텍스트 시퀀스를 정수 시퀀스로 변환하여 encoding

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

빈 샘플 제거

```
len(X_train), len(y_train)
```

```
(188501, 188501)
```

```
# 길이가 0인 샘플들의 인덱스 반환
drop_train = [index for index, sentence in enumerate(X_train) if len(sentence) < 1]
drop_test = [index for index, sentence in enumerate(X_test) if len(sentence) < 1]
print(f'빈 train 개수 : {len(drop_train)}')
print(f'빈 test 개수 : {len(drop_test)}')
```

```
빈 train 개수 : 115
빈 test 개수 : 31
```

```
# 빈 샘플들을 제거
X_train = np.delete(X_train, drop_train, axis=0)
y_train = np.delete(y_train, drop_train, axis=0)
X_test = np.delete(X_test, drop_test, axis=0)
y_test = np.delete(y_test, drop_test, axis=0)
len(X_train), len(X_test)
```

```
(188386, 45141)
```

- 전체 데이터에서 빈도수가 1이하인 단어가 삭제되었을 때 빈 샘플이 있을 수 있음

- 빈 샘플들은 어떤 레이블이 붙어있던 의미가 없으므로 제거

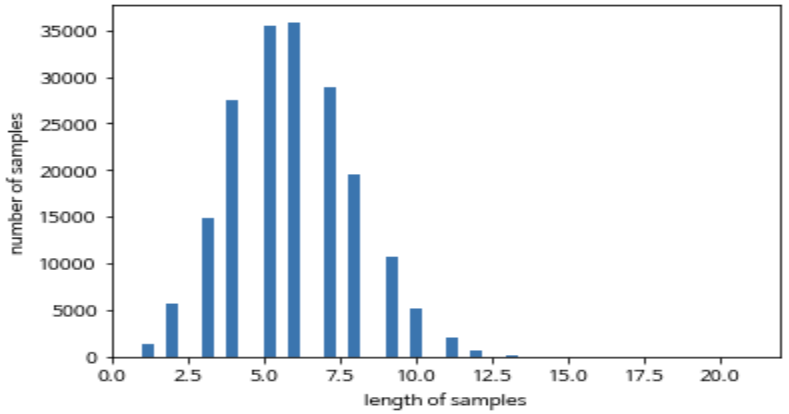
모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

패딩

```
print('헤드라인의 최대 길이 :',max(len(review) for review in X_train))
print('헤드라인의 평균 길이 :',sum(map(len, X_train))/len(X_train))
plt.hist([len(review) for review in X_train], bins=50)
plt.xlabel('length of samples')
plt.ylabel('number of samples')
plt.show()
```

```
헤드라인의 최대 길이 : 21
헤드라인의 평균 길이 : 5.856900194281953
```




```
def below_threshold_len(max_len, nested_list):
  count = 0
  for sentence in nested_list:
    if(len(sentence) <= max_len):
        count = count + 1
  print('전체 샘플 중 길이가 %s 이하인 샘플의 비율: %s'%(max_len, (count / len(nested_list))*100))
```

```
max_len = 12
below_threshold_len(max_len, X_train)
```

```
전체 샘플 중 길이가 12 이하인 샘플의 비율: 99.8375675474823
```

- 전체 train 데이터 중 약 99%의 헤드라인이 12이하의 길이를 가지므로 max_len을 12로 설정

```
X_train = pad_sequences(X_train, maxlen=max_len)
X_test = pad_sequences(X_test, maxlen=max_len)
```

- 패딩을 통해 문장의 길이를 통일 -> 데이터 셋의 병렬 연산을 위해

- 헤드라인 길이의 분포를 확인하여 max_len보다는 전체 샘플 중 길이가 12 이하인 샘플의 비율이 99.8% max_len 12로 설정

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

모델 학습 - LSTM

```
from tensorflow.keras.layers import Embedding, Dense, LSTM, Bidirectional
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import load_model
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

embedding_dim = 100
hidden_units = 128

model = Sequential()
model.add(Embedding(vocab_size, embedding_dim))
model.add(LSTM(hidden_units))
model.add(Dense(1, activation='sigmoid'))

es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=4)
mc = ModelCheckpoint('best_model.h5', monitor='val_acc', mode='max', verbose=1, save_best_only=True)

model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['acc'])
history = model.fit(X_train, y_train, epochs=15, callbacks=[es, mc], batch_size=64, validation_split=0.2)
```

```
Epoch 1/15
2348/2355 [============================>.] - ETA: 0s - loss: 0.6857 - acc: 0.5461
Epoch 00001: val_acc improved from -inf to 0.52139, saving model to best_model.h5
2355/2355 [==============================] - 19s 5ms/step - loss: 0.6857 - acc: 0.5461 - val_loss: 0.6939 - v
al_acc: 0.5214
Epoch 2/15
2353/2355 [============================>.] - ETA: 0s - loss: 0.6671 - acc: 0.5936
Epoch 00002: val_acc improved from 0.52139 to 0.52232, saving model to best_model.h5
2355/2355 [==============================] - 13s 6ms/step - loss: 0.6671 - acc: 0.5936 - val_loss: 0.7020 - v
al_acc: 0.5223
Epoch 3/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.6450 - acc: 0.6274
Epoch 00003: val_acc did not improve from 0.52232
2355/2355 [==============================] - 15s 6ms/step - loss: 0.6450 - acc: 0.6274 - val_loss: 0.7107 - v
al_acc: 0.5222
Epoch 4/15
2353/2355 [============================>.] - ETA: 0s - loss: 0.6205 - acc: 0.6585
Epoch 00004: val_acc did not improve from 0.52232
2355/2355 [==============================] - 15s 6ms/step - loss: 0.6205 - acc: 0.6585 - val_loss: 0.7309 - v
al_acc: 0.5138
Epoch 5/15
2347/2355 [============================>.] - ETA: 0s - loss: 0.5935 - acc: 0.6833
Epoch 00005: val_acc did not improve from 0.52232
2355/2355 [==============================] - 14s 6ms/step - loss: 0.5936 - acc: 0.6832 - val_loss: 0.7679 - v
al_acc: 0.5137
Epoch 00005: early stopping
```

- 임베딩 벡터의 차원은 100, 은닉층의 크기를 128로 적절히 설정

- 출력은 상승/하락 중 하나를 예측하는 이진 분류 문제이므로 활성화함수로 sigmoid 사용

- 손실함수로 binary_crossentropy 설정, batch size 64, epoch 15

- 과적합 방지 차원으로 val_loss가 4회 증가하면 정해진 epoch에 도달하지 못하였어도 학습을 조기 종료 시킴

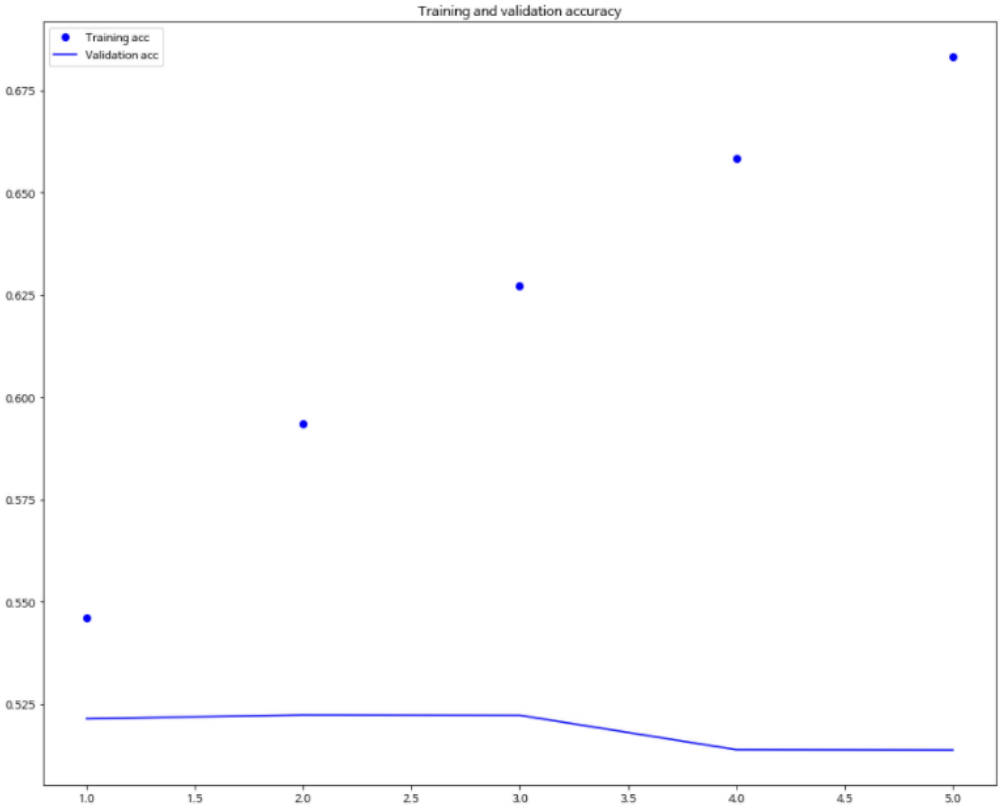
- ModelCheckpoint를 통해 val_acc가 개선될 경우 모델을 저장

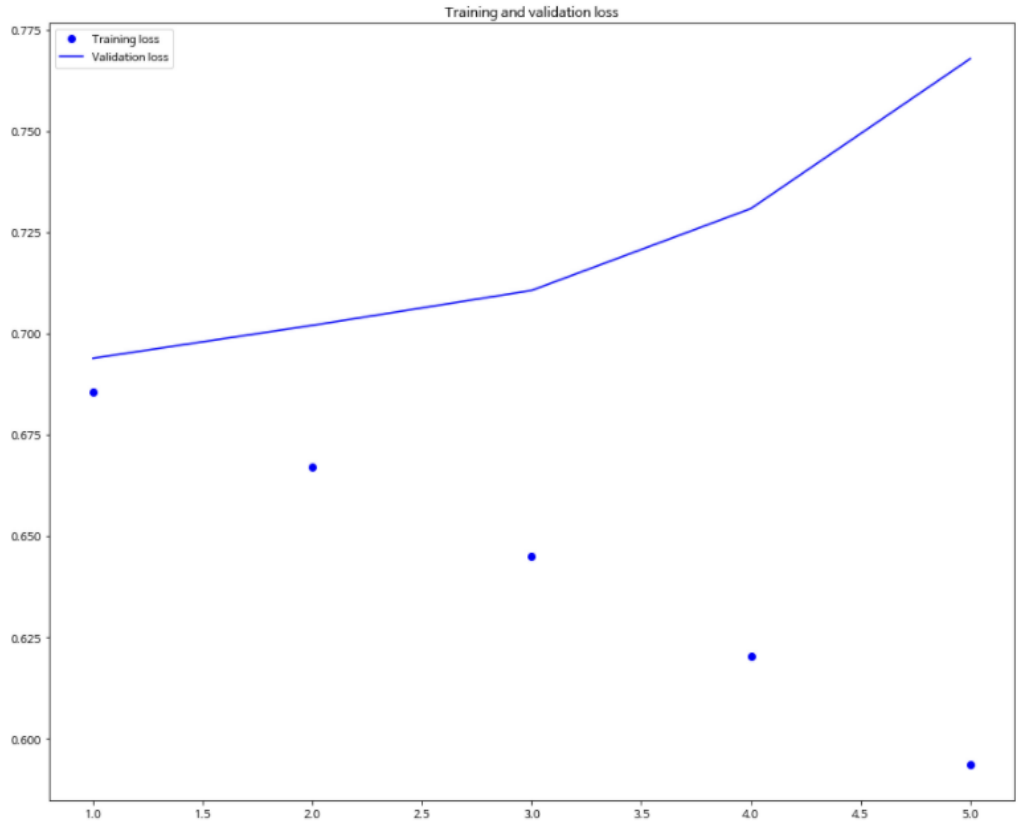
- train set의 20%를 validation set으로 사용

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

학습 결과 – LSTM





- 과적합 발생

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

모델 학습 - GRU

```
from tensorflow.keras.layers import Embedding, Dense, GRU
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import load_model
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

embedding_dim = 100
hidden_units = 128

model = Sequential()
model.add(Embedding(vocab_size, embedding_dim))
model.add(GRU(hidden_units))
model.add(Dense(1, activation='sigmoid'))

es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=4)
mc = ModelCheckpoint('best_model.h5', monitor='val_acc', mode='max', verbose=1, save_best_only=True)

model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['acc'])
history = model.fit(X_train, y_train, epochs=15, callbacks=[es, mc], batch_size=64, validation_split=0.2)
```

```
Epoch 1/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.6859 - acc: 0.5459
Epoch 00001: val_acc improved from -inf to 0.52951, saving model to best_model.h5
2355/2355 [==============================] - 105s 44ms/step - loss: 0.6858 - acc: 0.5459 - val_loss: 0.6902 -
val_acc: 0.5295
Epoch 2/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.6658 - acc: 0.5964
Epoch 00002: val_acc did not improve from 0.52951
2355/2355 [==============================] - 91s 39ms/step - loss: 0.6658 - acc: 0.5964 - val_loss: 0.7026 -
val_acc: 0.5162
Epoch 3/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.6431 - acc: 0.6314
Epoch 00003: val_acc did not improve from 0.52951
2355/2355 [==============================] - 93s 39ms/step - loss: 0.6431 - acc: 0.6313 - val_loss: 0.7155 -
val_acc: 0.5185
Epoch 4/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.6171 - acc: 0.6622
Epoch 00004: val_acc did not improve from 0.52951
2355/2355 [==============================] - 110s 47ms/step - loss: 0.6171 - acc: 0.6622 - val_loss: 0.7486 -
val_acc: 0.5093
Epoch 5/15
2354/2355 [============================>.] - ETA: 0s - loss: 0.5889 - acc: 0.6886
Epoch 00005: val_acc did not improve from 0.52951
2355/2355 [==============================] - 91s 39ms/step - loss: 0.5889 - acc: 0.6886 - val_loss: 0.7676 -
val_acc: 0.5182
Epoch 00005: early stopping
```

- 임베딩 벡터의 차원은 100, 은닉층의 크기를 128로 적절히 설정

- 출력은 상승/하락 중 하나를 예측하는 이진 분류 문제이므로 활성화함수로 시그모이드 사용

- 손실함수로 binary_crossentropy 설정, batch size 64, epoch 15

- 과적합 방지 차원으로 val_loss가 4회 증가하면 정해진 epoch에 도달하지 못하였어도 학습을 조기 종료 시킴

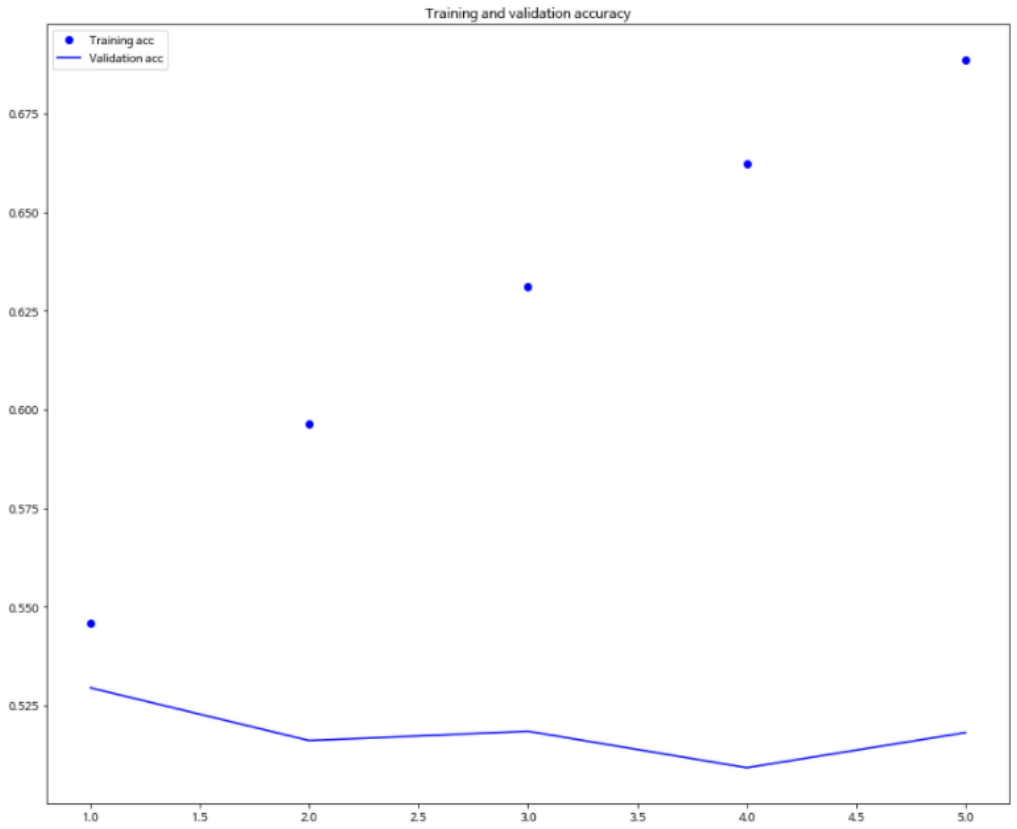
- ModelCheckpoint를 통해 val_acc가 개선될 경우 모델을 저장

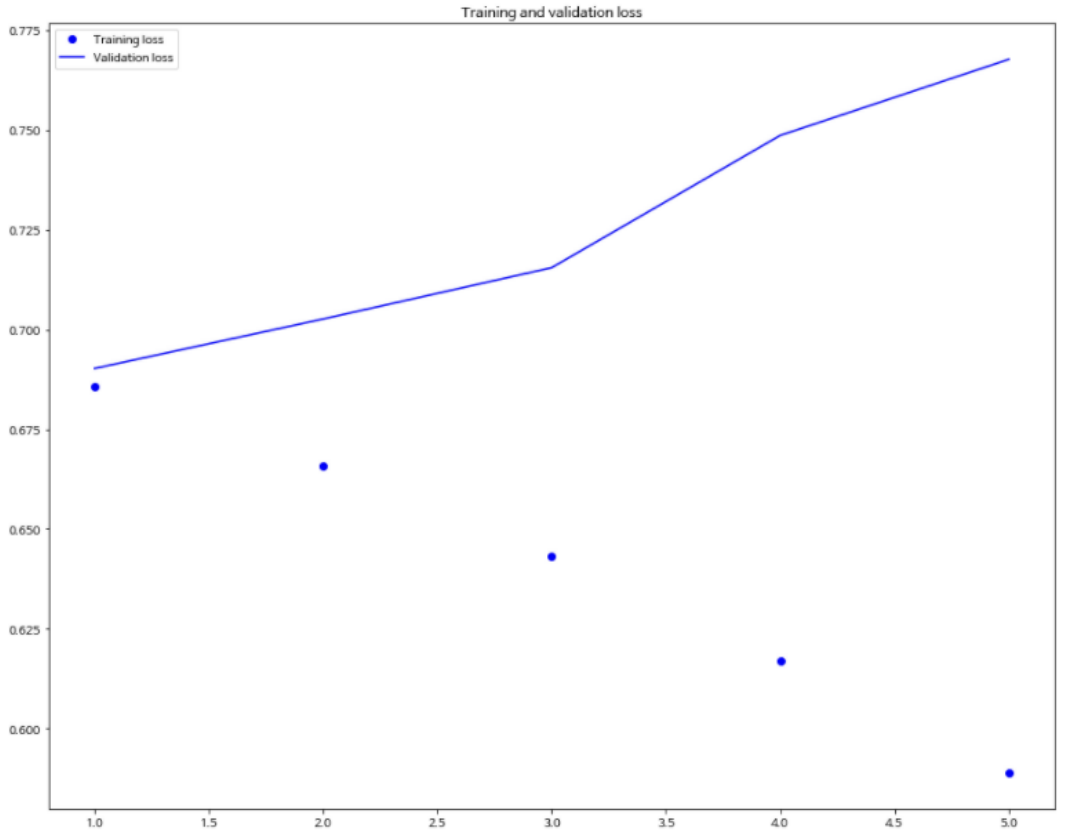
- train set의 20%를 validation set으로 사용

모델링 과정 및 학습 모델 구축

Keras text to sequence LSTM & GRU

학습 결과 – GRU



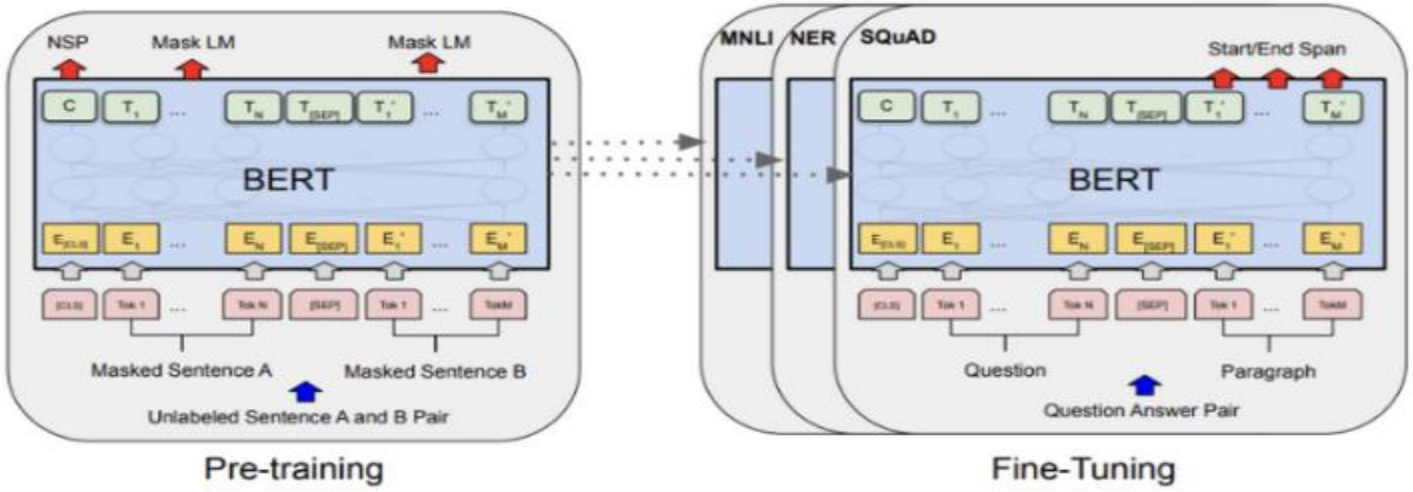


- 과적합 발생

모델링 과정 및 학습 모델 구축

Hugging Face BERT

BERT 모델 선정 배경



- Hugging Face의 transformer models는 transformer 기반의 다양한 모델을 Pytorch, Tensorflow로 구현해 놓은 모듈
- Hugging Face의 transformer model 중 하나인 Bert 모델을 사용
- 이미 광범위한 글을 학습한 모델 feature를 보유하고 있는 모델이라서 새로운 문장에 대한 분류 학습 수렴이 굉장히 빠름
- 전략적인 주가 예측기로서 감정기반 분류학습을 선택했다기 보다는 딥러닝으로 자연어 처리 분야를 뒤집었던 Bert 모델을 사용해보고 싶었음
- 약 18만개의 종목 관련 헤드라인 train set을 전이학습하여 새로운 분류기를 만듬

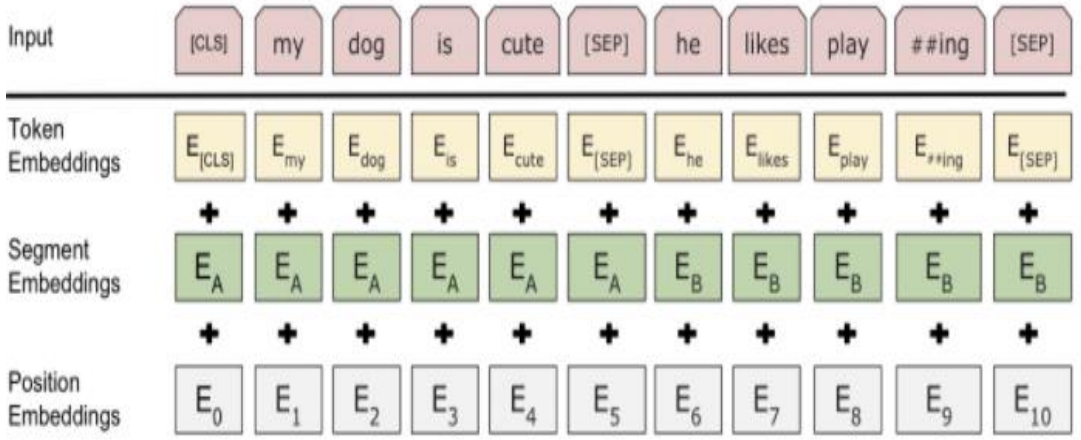
모델링 과정 및 학습 모델 구축

Hugging Face BERT

전처리 - BERT의 입력 형식에 맞게 변환

```
# BERT의 입력 형식에 맞게 변환
sentences = ["[CLS] " + str(sentence) + " [SEP]" for sentence in sentences]
sentences[:10]
```

```
['[CLS] [TV] 재계 총수들, 일자리 창출과 투자 확대 강조 [SEP]',
 '[CLS] 코스닥, 새해 첫 거래일 소폭 오름세 지속 [SEP]',
 '[CLS] 코스닥 새해 첫 거래일 소폭 오름세 지속 [SEP]',
 "[CLS] 코스닥, 개인·기관 '사자'에 상승 지속…저출산株 강세 [SEP]",
 "[CLS] 코스피, 외인 '팔자'에 약세 전환…방향성 탐색 [SEP]",
 '[CLS] 코스닥, 2012년 첫 거래일 상승세..0.91%↑ [SEP]',
 '[CLS] 코스닥, 새해 첫 거래일 개인 매수에 3.7p↑ [SEP]',
 '[CLS] [IR52 장영실상] LG화학, 일회용장갑 소재 `NBR 라텍스` [SEP]',
 '[CLS] [2012년 주목할 CEO 12인] 김반석 LG화학 부회장 [SEP]',
 '[CLS] 정유·화학주 하락 [SEP]']
```



- Classification을 뜻하는 [CLS] 심볼이 제일 앞에 삽입됨
- Fine-tuning시 출력에서 이 위치의 값을 사용하여 분류를 함
- [SEP]은 Seperation을 가리키는데, 두 문장을 구분하는 역할
- 헤드라인 문장은 하나이므로 [SEP]도 하나만 넣어줌

모델링 과정 및 학습 모델 구축

Hugging Face BERT

전처리 - BERT의 토크나이저로 문장을 토큰으로 분리

```
tokenizer = BertTokenizer.from_pretrained('bert-base-multilingual-cased', do_lower_case=False)
tokenized_texts = [tokenizer.tokenize(sent) for sent in sentences]

print(sentences[0])
print(tokenized_texts[0])
```

```
Downloading:   0%|          | 0.00/972k [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/29.0 [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/1.87M [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/625 [00:00<?, ?B/s]
[CLS] [TV] 재계 총수들, 일자리 창출과 투자 확대 강조 [SEP]
['[CLS]', '[', 'TV', ']', '재', '##계', '총', '##수', '##들', ',', '일', '##자', '##리', '창', '##출', '##
과', '투', '##자', '확', '##대', '강', '##조', '[SEP]']
```

- BERT는 형태소 분석으로 토큰을 분리하지 않음
- WordPiece라는 통계적인 방식을 사용
- 한 단어내에서 자주 나오는 글자들을 붙여서 하나의 토큰으로 만듬 -> 언어에 상관없이 토큰을 생성할 수 있다는 장점이 있음
- 또한 신조어 같이 사전에 없는 단어를 처리하기도 좋음
- 위 결과에서 '##' 기호는 앞 토큰과 이어진다는 표시
- 토크나이저는 여러 언어의 데이터를 기반으로 만든 'bert-base-multilingual-cased'를 사용 -> 한글도 처리 가능

모델링 과정 및 학습 모델 구축

Hugging Face BERT

전처리 - 패딩

```
# 입력 토큰의 최대 시퀀스 길이
MAX_LEN = 64

# 토큰을 숫자 인덱스로 변환
input_ids = [tokenizer.convert_tokens_to_ids(x) for x in tokenized_texts]

# 문장을 MAX_LEN 길이에 맞게 자르고, 모자란 부분을 패딩 0으로 채움
input_ids = pad_sequences(input_ids, maxlen=MAX_LEN, dtype="long", truncating="post", padding="post")

input_ids[0]
```

```
array([  101,   164, 10813,   166,  9659, 21611,  9761, 15891, 27023,
         117,  9641, 13764, 12692,  9736, 52363, 11882,  9881, 13764,
        9994, 14423,  8853, 20626,   102,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0,     0,     0,     0,     0,     0,     0,     0,     0,
           0])
```

- 보통 딥러닝 모델에는 토큰 자체를 입력으로 넣을 수 없음
- 임베딩 레이어에는 토큰을 숫자로 된 인덱스로 변환하여 사용
- BERT의 토크나이저는 {단어토큰:인덱스}로 구성된 단어사전을 가지고 있음
- 이를 참조하여 토큰을 인덱스로 바꿈

모델링 과정 및 학습 모델 구축

Hugging Face BERT

모델링

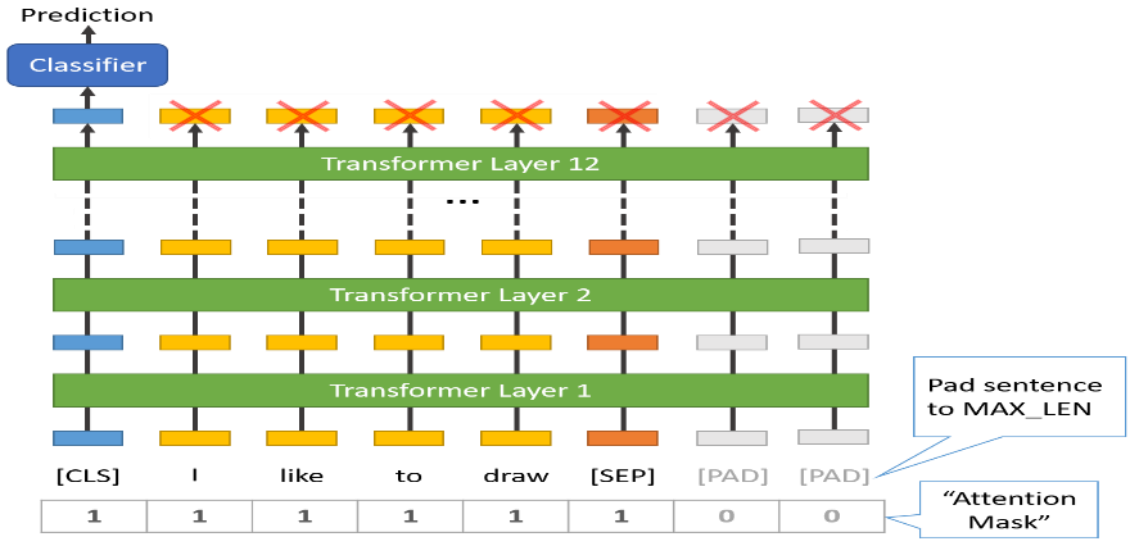
```
# GPU 디바이스 이름 구함
device_name = tf.test.gpu_device_name()

# GPU 디바이스 이름 검사
if device_name == '/device:GPU:0':
    print('Found GPU at: {}'.format(device_name))
else:
    raise SystemError('GPU device not found')
```

```
Found GPU at: /device:GPU:0
```

```
# 디바이스 설정
if torch.cuda.is_available():
    device = torch.device("cuda")
    print('There are %d GPU(s) available.' % torch.cuda.device_count())
    print('We will use the GPU:', torch.cuda.get_device_name(0))
else:
    device = torch.device("cpu")
    print('No GPU available, using the CPU instead.')
```

```
There are 1 GPU(s) available.
We will use the GPU: Tesla P100-PCIE-16GB
```

```
# 분류를 위한 BERT 모델 생성
model = BertForSequenceClassification.from_pretrained("bert-base-multilingual-cased", num_labels=2)
model.cuda()
```



- 사전 훈련된 BERT는 다양한 문제로 전이학습이 가능
- 여기서는 위 그림과 같이 한 문장을 분류하는 방법을 사용(헤드라인)
- 헤드라인 문장이 입력으로 들어가면, 긍정(상승)/부정(하락)으로 구분
- 모델의 출력에서 [CLS] 위치인 첫 번째 토큰에 새로운 레이어를 붙여서 fine-tuning
- Huggning Face는 BertForSequenceClassification() 함수를 제공하기 때문에 쉽게 구현할 수 있음

모델링 과정 및 학습 모델 구축

Hugging Face BERT

모델 학습

```
# 재현을 위해 랜덤시드 고정
seed_val = 13
random.seed(seed_val)
np.random.seed(seed_val)
torch.manual_seed(seed_val)
torch.cuda.manual_seed_all(seed_val)

# 그래디언트 초기화
model.zero_grad()

# 에폭만큼 반복
for epoch_i in range(0, epochs):

    # ========================================
    #               Training
    # ========================================

    print("")
    print('======== Epoch {:} / {:} ========'.format(epoch_i + 1, epochs))
    print('Training...')

    # 시작 시간 설정
    t0 = time.time()

    # 로스 초기화
    total_loss = 0

    # 훈련모드로 변경
    model.train()

    # 데이터로더에서 배치만큼 반복하여 가져옴
    for step, batch in enumerate(train_dataloader):
        # 경과 정보 표시
        if step % 500 == 0 and not step == 0:
            elapsed = format_time(time.time() - t0)
            print('  Batch {:>5,}  of  {:>5,}.    Elapsed: {:}.'.format(step, len(train_dataloader), elapsed

        # 배치를 GPU에 넣음
        batch = tuple(t.to(device) for t in batch)

        # 배치에서 데이터 추출
        b_input_ids, b_input_mask, b_labels = batch

        # Forward 수행
        outputs = model(b_input_ids,
                        token_type_ids=None,
                        attention_mask=b_input_mask,
                        labels=b_labels)
```

```
======== Epoch 1 / 4 ========
Training...
  Batch   500  of  4,713.    Elapsed: 0:01:49.
  Batch 1,000  of  4,713.    Elapsed: 0:03:39.
  Batch 1,500  of  4,713.    Elapsed: 0:05:28.
  Batch 2,000  of  4,713.    Elapsed: 0:07:17.
  Batch 2,500  of  4,713.    Elapsed: 0:09:06.
  Batch 3,000  of  4,713.    Elapsed: 0:10:55.
  Batch 3,500  of  4,713.    Elapsed: 0:12:44.
  Batch 4,000  of  4,713.    Elapsed: 0:14:33.
  Batch 4,500  of  4,713.    Elapsed: 0:16:22.

  Average training loss: 0.69
  Training epcoh took: 0:17:09

Running Validation...
  Accuracy: 0.53
  Validation took: 0:01:14

======== Epoch 2 / 4 ========
Training...
  Batch   500  of  4,713.    Elapsed: 0:01:49.
  Batch 1,000  of  4,713.    Elapsed: 0:03:38.
  Batch 1,500  of  4,713.    Elapsed: 0:05:27.
  Batch 2,000  of  4,713.    Elapsed: 0:07:16.
  Batch 2,500  of  4,713.    Elapsed: 0:09:05.
  Batch 3,000  of  4,713.    Elapsed: 0:10:55.
  Batch 3,500  of  4,713.    Elapsed: 0:12:44.
  Batch 4,000  of  4,713.    Elapsed: 0:14:33.
  Batch 4,500  of  4,713.    Elapsed: 0:16:22.

  Average training loss: 0.69
  Training epcoh took: 0:17:09

Running Validation...
  Accuracy: 0.53
  Validation took: 0:01:14

======== Epoch 3 / 4 ========
Training...
  Batch   500  of  4,713.    Elapsed: 0:01:49.
  Batch 1,000  of  4,713.    Elapsed: 0:03:38.
  Batch 1,500  of  4,713.    Elapsed: 0:05:28.
  Batch 2,000  of  4,713.    Elapsed: 0:07:17.
  Batch 2,500  of  4,713.    Elapsed: 0:09:06.
  Batch 3,000  of  4,713.    Elapsed: 0:10:55.
  Batch 3,500  of  4,713.    Elapsed: 0:12:45.
  Batch 4,000  of  4,713.    Elapsed: 0:14:34.
  Batch 4,500  of  4,713.    Elapsed: 0:16:23.
```

모델링 과정 및 학습 모델 구축

SKT Brain KoBERT

모델 선정 배경

KoBERT

Korean BERT (Bidirectional Encoder Representations from Transformers)

KoBERT는 기존 BERT의 한국어 성능 한계를 극복하기 위해 개발되었다. 위키피디아나 뉴스 등에서 수집한 수백만 개의 한국어 문장으로 이루어진 대규모말뭉치(corpus)를 학습하였으며, 한국어의 불규칙한 언어 변화의 특성을 반영하기 위해 데이터 기반 토큰화 (Tokenization) 기법을 적용하여 기존 대비 27%의 토큰만으로 2.6% 이상의 성능 향상을 이끌어 냈다.

대량의 데이터를 빠른시간에 학습하기 위해 링 리듀스(ring-reduce) 기반 분산 학습 기술을 사용하여, 십억 개 이상의 문장을 다수의 머신에서 빠르게 학습한다. 더불어, 파이토치 (PyTorch), 텐서플루(TensorFlow), ONNX, MXNet을 포함한 다양한 딥러닝 API를 지원함으로써, 많은 분야에서 언어 이해 서비스 확산에 기여하고 있다.

- GitHub : https://github.com/SKTBrain/KoBERT

- 국내 주식 종목의 뉴스 헤드라인 텍스트 데이터
- 한국어 데이터셋을 바탕으로 사전 학습된 모델인 KoBERT

모델링 과정 및 학습 모델 구축

SKT Brain KoBERT

모델 학습

```
class BERTDataset(Dataset):
    def __init__(self, dataset, sent_idx, label_idx, bert_tokenizer, max_len,
                 pad, pair):
        transform = nlp.data.BERTSentenceTransform(
            bert_tokenizer, max_seq_length=max_len, pad=pad, pair=pair)

        self.sentences = [transform([i[sent_idx]]) for i in dataset]
        self.labels = [np.int32(i[label_idx]) for i in dataset]

    def __getitem__(self, i):
        return (self.sentences[i] + (self.labels[i], ))

    def __len__(self):
        return (len(self.labels))
```

```
print(train_data.shape)
print(test_data.shape)
```

```
(188530, 2)
(45174, 2)
```

```
dataset_train = nlp.data.TSVDataset('./train_file.tsv',field_indices=[1,2], num_discard_samples=1)
dataset_test = nlp.data.TSVDataset('./test_file.tsv',field_indices=[1,2], num_discard_samples=1)
```

```
class BERTClassifier(nn.Module):
    def __init__(self,
                 bert,
                 hidden_size = 768,
                 num_classes=2,
                 dr_rate=None,
                 params=None):
        super(BERTClassifier, self).__init__()
        self.bert = bert
        self.dr_rate = dr_rate

        self.classifier = nn.Linear(hidden_size , num_classes)
        if dr_rate:
            self.dropout = nn.Dropout(p=dr_rate)

    def gen_attention_mask(self, token_ids, valid_length):
        attention_mask = torch.zeros_like(token_ids)
        for i, v in enumerate(valid_length):
            attention_mask[i][:v] = 1
        return attention_mask.float()

    def forward(self, token_ids, valid_length, segment_ids):
        attention_mask = self.gen_attention_mask(token_ids, valid_length)

        _, pooler = self.bert(input_ids = token_ids, token_type_ids = segment_ids.long(), attention_mask = a
        if self.dr_rate:
            out = self.dropout(pooler)
        return self.classifier(out)
```

```
device = torch.device("cuda:0")

model = BERTClassifier(bertmodel,  dr_rate=0.5).to(device)
final_md = torch.load('/content/drive/MyDrive/final_project/kobert_v2.pt',map_location=device)
final_md.load_state_dict(model.state_dict())
final_md.eval()
```

- 전처리 과정은 위에서 했던 BERT와 동일
- 15만개의 영화 평론 긍/부정이 라벨링된 데이터 + 18만개의 뉴스 헤드라인 상승(긍정)/하락(부정) train set 전이학습

Part 4,

학습 모델을 통한 결과 분석

학습 모델을 통한 결과 분석

Test set 예측 값 도출

감성 지수 추출

```
loaded_model = load_model('best_model.h5') # best model 불러오기
```

```
score = loaded_model.predict(X_test)
```

```
test['sentiment_prob'] = score
test.head()
```

	name	headline	date	change	token_mecab	sentiment_prob
0	카카오	"기술 사대주의로는 AI 정복 힘들다"	2020-01-02	0	기술 사대주의 AI 정복 힘들	0.517778
1	카카오	[해설]中 디지털 종속 우려…동남아 국가 '中 QR결제 금지' 초강수	2020-01-02	0	해설 중국 디지털 종속 우려 동남아 국가 중국 QR 결제 금지 강수	0.488098
2	카카오	씨티은행, 1.3%포인트 금리인하 직장인 신용대출..한달 연장 이벤트	2020-01-02	0	씨티 은행 포인트 금리 인하 직장 신용 대출 연장 이벤트	0.485674
3	카카오	'소비자분쟁 통지서' 카톡으로 제공된다	2020-01-02	0	소비자 분쟁 통지서 카톡 제공	0.430201
4	카카오	[이슈분석]키 잡은 공정위…배민-DH 합병 경우의 수	2020-01-02	0	이슈 분석 공정위 배민 DH 합병 경우	0.415983

학습 모델을 통한 결과 분석

Test set 예측 값 도출

최종 예측 값 도출

일별 감성 주가 예측 예

create_dt	Document	Prediction
2014-03-10	장중 2만6650원(3.5%)까지 올랐다.	긍정 0.73
2014-03-10	올해 3분기부터 회복세를 보인 D램 수요는 4분기에도 ...	부정 0.64
2014-03-10	이날 오전 11시 20분 기준 외국인과 기관은 SK하이닉스를 각각....	긍정 0.92
2014-03-10	최대 실적 D램 공급 부족 지속 될 것으로 전망된다.	긍정 0.96

▪ S_{PNt} = (0.73+0.36+0.92+0.96)/4 = 74.25% 긍정(상승)

```
from tqdm import tqdm

stock_prob_list = []
for name in tqdm(test['name'].unique()): # 10개 종목
    temp_list = []
    for date in tqdm(test['date'].unique()): # 일자 별
        avg_prob = np.mean(test[(test['date'] == date) & (test['name'] == name)]['sentiment_prob'])
        temp_list.append(avg_prob)
    stock_prob_list.append(temp_list)
```

```
# 각각 데이터프레임 생성

ten_df = []
def make_per_df(name):
  df = pd.DataFrame({'predict':result_df[name],'label': change_df[name]}, index = test['date'].unique())
  df['name'] = name
  df = df.sort_index()
  df.dropna(subset=['predict'],inplace=True)
  return df

def labeling(df):
  df['predict'][df['predict'] < 0.5] = 0
  df['predict'][df['predict'] >= 0.5] = 1
  return df
```

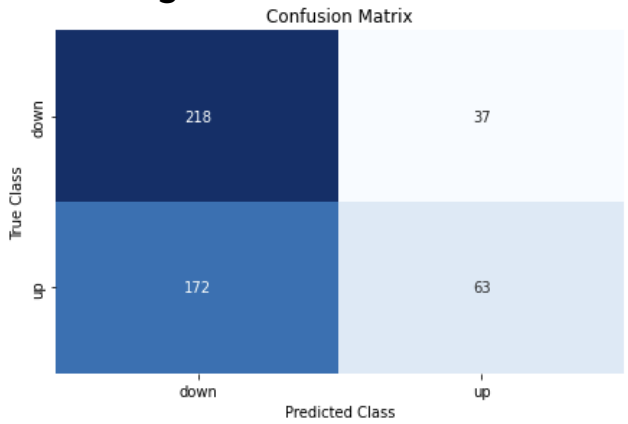
- 예측기로서 동작할 수 있는 Selection Function을 만들어야 함
- 특정날짜의 주가 기사문장이 N개라고 했을 때,
- N개의 문장의 감성 분류스코어를 계산한 확률평균을 그날 주식 종가의 업다운을 결정하는 예측 값으로 사용
- 문장 판별 확률의 평균으로 선택한 이유 -> 일반화된 테스트 데이터가 아님

학습 모델을 통한 결과 분석

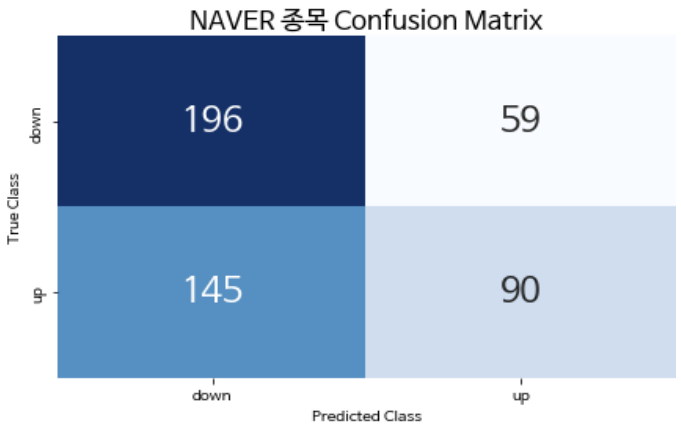
Test set **평가 및 분석**

모델 별 confusion matrix (ex. NAVER)

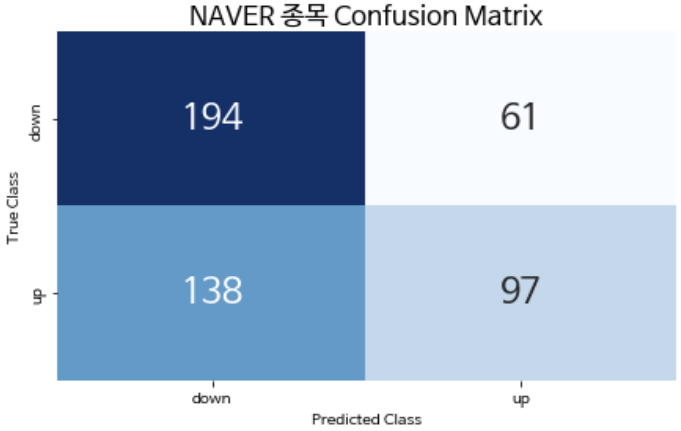
1. Stacking Classifier



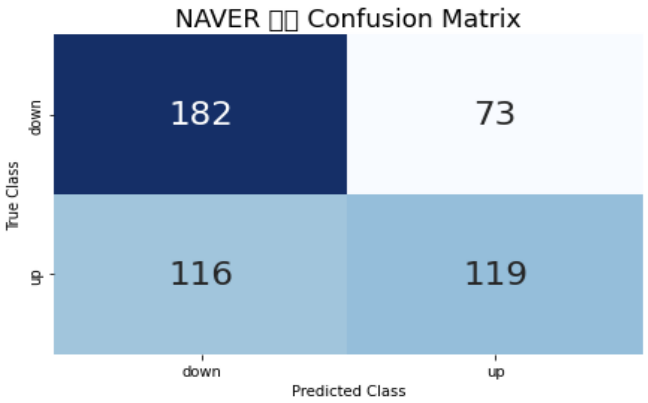
2. LSTM



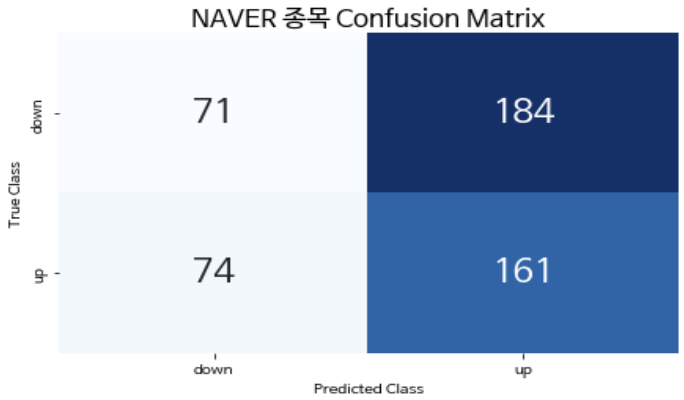
3. GRU



4. BERT



5. KoBERT



학습 모델을 통한 결과 분석

Test set **평가 및 분석**

모델 별 classification report (ex. NAVER)

1. Stacking Classifier

	precision	recall	f1-score	support
down	0.56	0.85	0.68	255
up	0.63	0.27	0.38	235
accuracy			0.57	490
macro avg	0.59	0.56	0.53	490
weighted avg	0.59	0.57	0.53	490

2. LSTM

	precision	recall	f1-score	support
down	0.57	0.77	0.66	255
up	0.60	0.38	0.47	235
accuracy			0.58	490
macro avg	0.59	0.58	0.56	490
weighted avg	0.59	0.58	0.57	490

3. GRU

	precision	recall	f1-score	support
down	0.58	0.76	0.66	255
up	0.61	0.41	0.49	235
accuracy			0.59	490
macro avg	0.60	0.59	0.58	490
weighted avg	0.60	0.59	0.58	490

4. BERT

	precision	recall	f1-score	support
down	0.61	0.71	0.66	255
up	0.62	0.51	0.56	235
accuracy			0.61	490
macro avg	0.62	0.61	0.61	490
weighted avg	0.62	0.61	0.61	490

5. KoBERT

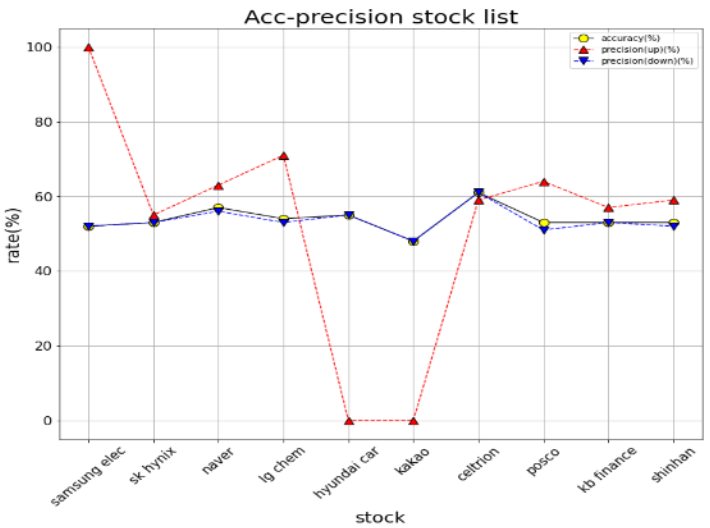
	precision	recall	f1-score	support
down	0.49	0.28	0.35	255
up	0.47	0.69	0.56	235
accuracy			0.47	490
macro avg	0.48	0.48	0.46	490
weighted avg	0.48	0.47	0.45	490

학습 모델을 통한 결과 분석

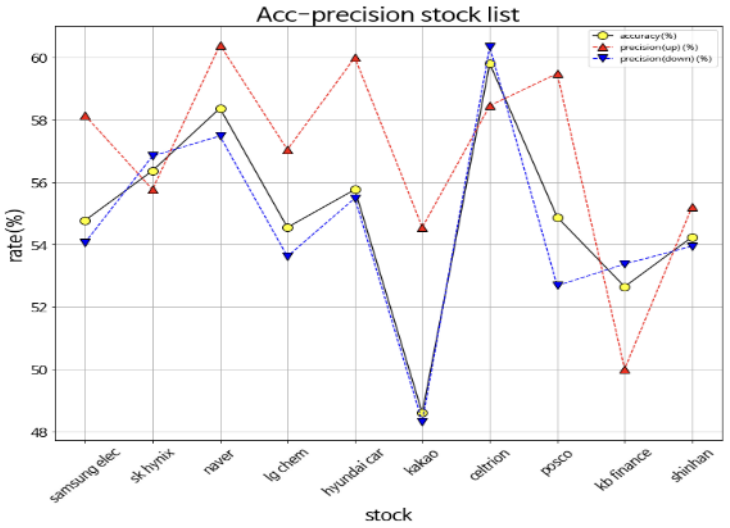
Test set **평가 및 분석**

모델 별 10개 종목 Accuracy-Precision 비교

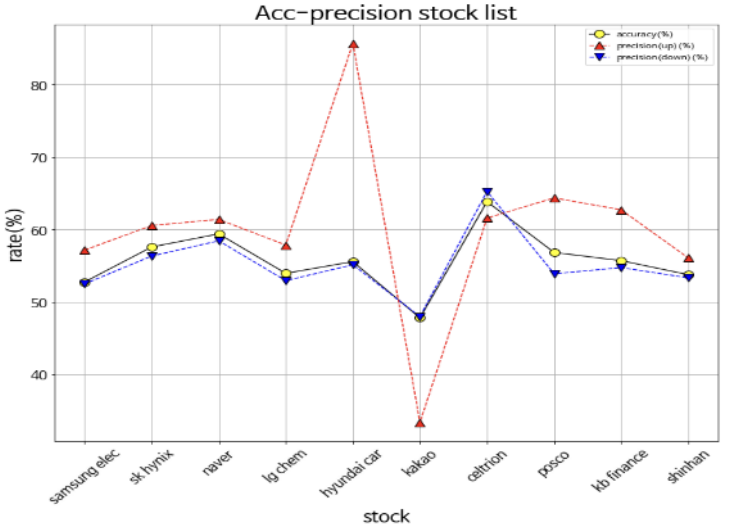
1. Stacking Classifier



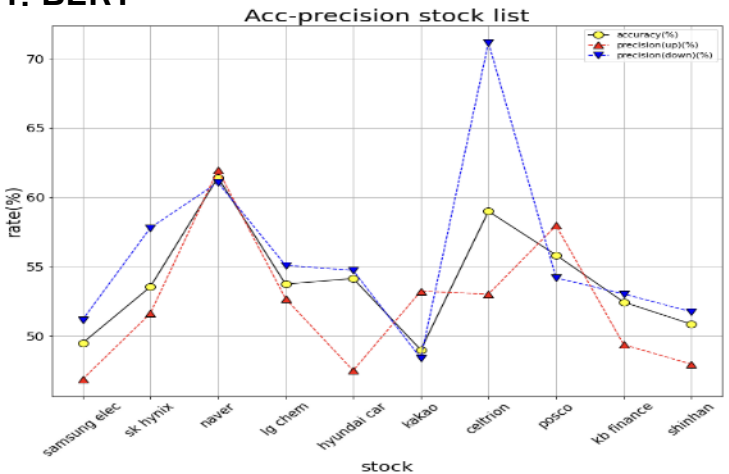
2. LSTM



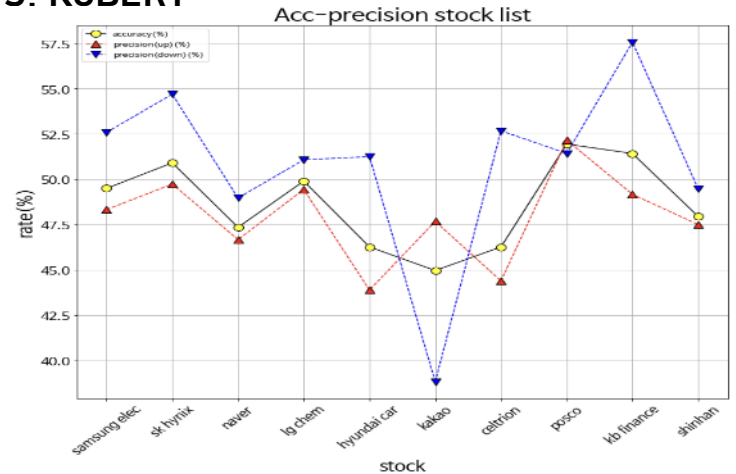
3. GRU



4. BERT



5. KoBERT



Part 5,

한계 및 보완점

한계 및 보완점

1. (크롤링 데이터 과정) 본문 학습 X, 주식 종목과 연관성이 낮다고 판단되는 기사 헤드라인 수집

모델 학습과 크롤링 과정에서 물리적 시간의 한계

- 본문까지 활용한 자연어분석을 진행하여 평균을 내는 방식으로 develop 가능

네이버 금융 뉴스 내 카카오 종목 검색 시 카카오와 연관이 낮은 헤드라인이 크롤링

- “00제과 신제품 출시, **라이언과 니니즈**에 힘입어 매출 상승” — 카카오 종목과 관련없는 헤드라인(카카오 캐릭터에 대한 내용)에 대한 전처리에 대한 한계 존재

- 본문 추가학습이나 종목과 관련있는 전처리 후 반복 크롤링 작업 수행 필요

2. (라벨링 과정) 특정 분야(금융, 유통, 바이오 등)에 대한 감성 분석의 한계

금융 뉴스 감성 분석은 관련 전문용어와 라벨링된 데이터가 존재하지 않으므로, 특정 종목에 대한 상승, 하락 지표로 임의로 선정함 (감성 라벨 지표 ✕)

학습된 딥러닝 모델에선 특정 도메인에서 사용되는 전문 용어를 기반으로 학습이 진행되었기에 명확한 지표를 얻기가 어려움

한계 및 보완점

3. 하이퍼파라미터 튜닝 최적화, 과적합 원인 미해결, layer 수정, 다양한 embedding 기법(glove,fasttext) 필요

모델 구현에 앞서 기존 텍스트에 전처리하는 과정이 많이 소요

- 모델에 대한 성능을 높이기 위해 하이퍼파라미터 튜닝 최적화를 진행할 수 있었으나, 낮은 test 평가 지표로 인해 전처리 과정에 집중됨
- 모델에 대한 성능에 초점을 두고 layer 구성 및 다양한 embedding 기법을 조합하여 develop 가능

Part 6,

참고문헌

참고문헌

•KoBERT로 감성 분석을 해보자(Text Classification)

- https://tech-diary.tistory.com/31?category=952584

•네이버 영화 평점 감성분석

- https://github.com/e9t/nsmc/

•[NLP] 텍스트 분류와 감성(Sentiment)분석 구현하기

- https://techblog-history-younghunjo1.tistory.com/111?category=924148

•네이버 영화 리뷰 감성 분류하기(Naver Movie Review Sentiment Analysis)

- https://wikidocs.net/44249

•[Python, KoBERT] 다중 감정 분류 모델 구현하기 (huggingface로 이전 방법 O)

- https://hoit1302.tistory.com/159#6.앞으로의기대

•KoBERT 네이버 영화 리뷰 감성분석 with Hugging Face

- https://complexoftaste.tistory.com/2

•SKTBrain - KoBERT 오픈소스

- https://github.com/SKTBrain/KoBERT

•데이터 사이언티스트 한국어 자연어처리 방법

- https://blog.naver.com/reasoninlife/222574624867

•뉴스데이터를 활용한 주가 예측2 (감성분석)

- https://blog.naver.com/dalgoon02121/222051184805

•[Keras]기사 제목을 가지고 긍정 / 부정 / 중립으로 분류하는 모델 만들어보기

- https://somjang.tistory.com/entry/Keras기사-제목을-가지고-긍정-부정-중립-분류하는-모델-만들어보기#google_vignette

감사합니다
THANK YOU
MERCI
GRACIAS
ありがとう
谢谢