비정형 데이터 활용 세미 프로젝트

서비스 산업 데이터분석가 취업캠프

팀 | 금요일
(김하은, 고현서, 권지은, 김두규, 방민준, 안재훈, 이정한)

2023.06.19

목차

1. 프로젝트 개요

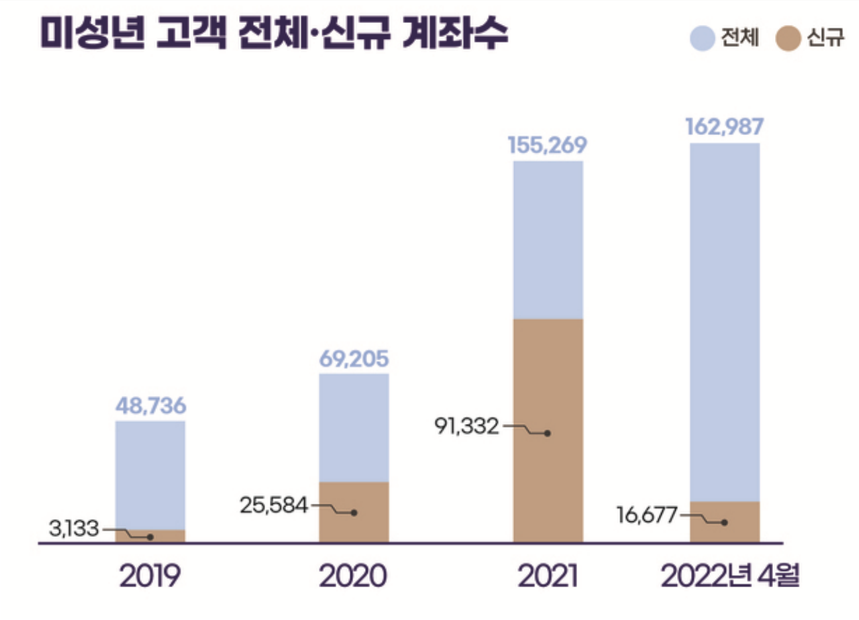
2. 모델 플로우

3. 요약 및 시사점

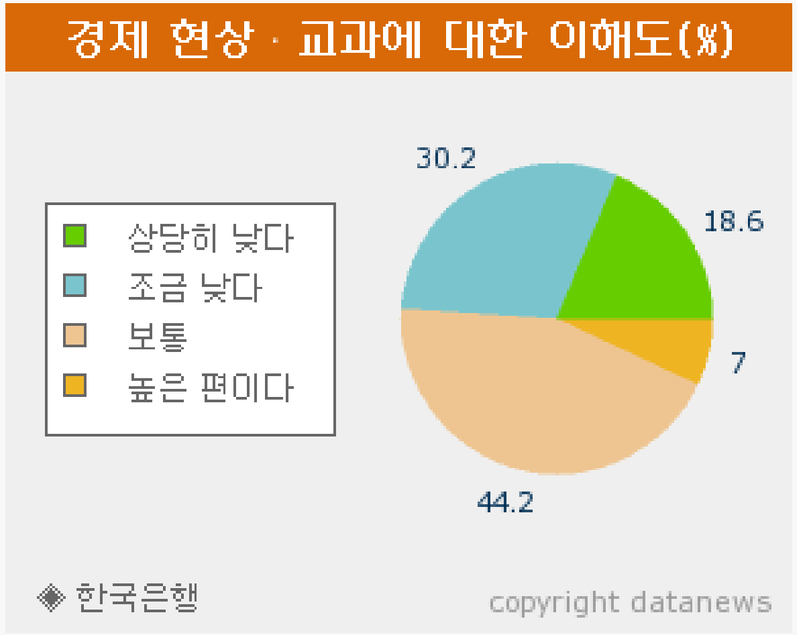
4. 개발 후기

I 프로젝트 개요

1) 프로젝트 배경 / 문제정의



1 미성년자 비대면 증권계좌 개설이 허용되며
미성년 주식 투자자들 급증



2 미성년자들의 낮은 경제 이해도

Problem

미성년 투자자들이 급증했지만
주식 정보를 이해하는데
어려움을 겪고 있음

Solution

뉴스제목만 보고도 상향/하향을
알려주는 숏리뷰 서비스 개발

I 프로젝트 개요

2) 역할 분담

기획서 2

작업 이름	상태	담당자
세미프로젝트 기획안 초안	완료	현서 고, 김하은, 안재훈
세미프로젝트 기획안 완성	완료	지은 권

PPT 4

작업 이름	상태	담당자
PPT 레퍼런스	완료	김하은, 현서 고
PPT 초안 1	완료	김하은, 현서 고
PPT 디자인	완료	김하은
PPT 발표	완료	민준 방

EDA 및 데이터 전처리 6

작업 이름	상태	담당자
A조	완료	현서 고, 민준 방
B조	완료	안재훈
C조	완료	김하은, 지은 권, 정한 이
SentenceBert	완료	김하은
test/train 데이터셋 분리	완료	민준 방, 안재훈
코드 검수	완료	Kyoo Kim

모델링 7

작업 이름	상태	담당자
RandomForest	완료	정한 이
AdaBoost	완료	지은 권
나이브베이즈	완료	현서 고, 지은 권
LSTM	완료	민준 방
BERT	완료	김하은
뉴스 제목 모델에 적용	완료	안재훈, 현서 고
모델링 코드 검수	완료	Kyoo Kim

II 모델 플로우

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

[데이터 선정]

미래에셋증권 미성년자 보유 상위 종목 (2023.04.25 기준)

순위	국내주식	해외주식	ETF
1 위	삼성전자	애플	TIGER 미국 S&P500
2 위	카카오	테슬라	TIGER 미국나스닥 100
3 위	삼성전자 우선주	마이크로소프트	TIGER 차이나전기차

=> 미성년자 보유 국내주식 종목 1위인 삼성전자 데이터를 수집

[데이터 수집 방법]

- 크롤링 사이트 : 한경 컨세서스 / 네이버 증권 뉴스

- 한경 컨센서스에서 삼성전자로 필터링 된 리포트 크롤링
  => 상향/하향 분류 별 리포트 제목, 작성일 등 수집
  (총 237개/기간:5년치)

- 네이버 주식 종목에서 삼성전자로 필터링 된 뉴스 크롤링
  => 기사 제목, 작성일 등 수집(4000개)

II 모델 플로우

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

[한경 컨센서스 크롤링 데이터]

	작성일	종목명	종목코드	제목
0	2023-04-10	삼성전자	5930	삼성전자 업황이 나쁠수록 감산은 확대된다
1	2023-03-15	삼성전자	5930	삼성전자 뻔하지만 당연한 최고의 선택
2	2023-01-09	삼성전자	5930	삼성전자 4Q22 잠정실적 리뷰
3	2022-11-21	삼성전자	5930	삼성전자 경쟁력 격차 복구 여부에 주목 저점 분할 매수 권고
4	2022-07-29	삼성전자	5930	삼성전자 시험은 어렵지만 변별력은 높아진다
5	2022-04-08	삼성전자	5930	삼성전자 1분기 디스플레이 기대 이상
6	2022-02-14	삼성전자	5930	삼성전자 메모리 업황 개선 본격화
7	2022-02-08	삼성전자	5930	삼성전자 리스크는 주가에 반영 성장성은 미반영
8	2022-01-10	삼성전자	5930	삼성전자 IT Set 실판매 강도에 주목 필요
9	2022-01-10	삼성전자	5930	삼성전자 반도체 업종 조정기의 끝자락
10	2022-01-05	삼성전자	5930	삼성전자 2022년은 퀀텀 점프가 될 한 해
11	2021-12-20	삼성전자	5930	삼성전자 반도체와 모바일 사업부문 실적 호조
12	2021-11-24	삼성전자	5930	삼성전자 23년 연간 매출액 300조원
13	2021-10-25	삼성전자	5930	삼성전자 파운드리 사업부 가치 반영
14	2021-07-08	삼성전자	5930	삼성전자 3Q21 하이퍼스케일러들의 서버 DRAM 재고 감소 예상
15	2021-07-01	삼성전자	5930	삼성전자 2Q21 영업이익 11 3조원 전망
16	2021-06-18	오이솔루션	138080	오이솔루션 삼성전자 5G 장비 수주 확대 수혜 기대
17	2021-03-23	삼성전자	5930	삼성전자 TSMC 에 대적할 수 있는 유일한 반도체 기업
18	2021-03-08	원익IPS	240810	원익IPS 삼성전자 반도체 부분 투자 확대 예상
19	2021-01-29	삼성전자	5930	삼성전자 1Q21 영업이익 9 0조원 전망
20	2021-01-28	삼성물산	28260	삼성물산 4Q20 Review 삼성전자 배당 확대 수혜 기대

(총 237개/기간:5년치)

[네이버증권 뉴스 크롤링 데이터]

	종목코드	종목명	title
0	5930	삼성전자	'월가의 황제'가 한국에 왔다…제일 먼저 만난 사람은? [투자360]
1	5930	삼성전자	삼성 '유연한 조직 만들기' 시동…전 뉴스위크 회장 초청강연
2	5930	삼성전자	코스피, 기관 대량 매수에 2,610대 마감…원·달러 환율 1,308원 10전
3	5930	삼성전자	연중최고치' 코스피 2600선 안착..삼성🏗·LG전자 '52주 신고가'
4	5930	삼성전자	'8만전자' 불 지피는 증권가 "삼성전자 좋아질 일만"…SK하이닉스 목표가도 ↑↑
5	5930	삼성전자	삼성, 리자드 스미스 핑커턴재단 CEO 초청 특강 실시
6	5930	삼성전자	삼성전자, 외인 매도에 주가 숨고르기…증권가는 "9만전자, BUY"
7	5930	삼성전자	공생 혹은 동상이몽…'우호적 행동주의'는 성공할 수 있을까
8	5930	삼성전자	[조점] 30년 전 이건희가 외친 '신경영'…이재용의 '뉴 삼성'에 쏠리는 눈
9	5930	삼성전자	삼성전자 치솟자…조용한 질주에 남몰래 웃었다는 '이 기업'
10	5930	삼성전자	다이먼 JP모건 회장, 5년 만에 방한…KIC 등 금융기관장들 만나
11	5930	삼성전자	LG전자 'RE100' 가입…축구장 3개 크기 태양광 발전소 건설도
12	5930	삼성전자	[마켓PRO 칼럼]실적 뒷받침 교육주?…소외된 기업 옥석가릴 때
13	5930	삼성전자	[마감시황] 코스피, 기관 대량 매수에 연중 '최고치'경신…2610대 안착
14	5930	삼성전자	유연한 조직' 이재용 특명에 삼성 임원 대상 특강
15	5930	삼성전자	[시황] 코스피, 기관 매수세 유입에 상승…2615.41 마감
16	5930	삼성전자	코스피, 기관 매수에 0.5% 상승 마감…반도체는 조정
17	5930	삼성전자	코스피, 2610선 안착…가치주 강세 주도[마감시황]
18	5930	삼성전자	"하반기 3000 가나"… 코스피 연중 최고치 기록에 기대감 확산
19	5930	삼성전자	대구창조경제혁신센터 2022 성과평가 최우수 기관 선정
20	5930	삼성전자	코스피, 기관 매수에 0.5% 상승 마감…반도체주는 조정(종합)

(총 4000개)

II 모델 플로우

데이터 수집 | **데이터 전처리** | train/test 분리 | 모델링 | 결과

[공통 전처리 과정]

① 한경 컨센서스 크롤링 데이터 -> 상향 :1 / 하향 :0 라벨링

	작성일	종목명	종목코드	제목	label
0	2023-04-10	삼성전자	5930	삼성전자 업황이 나쁠수록 감산은 확대된다	1
1	2023-03-15	삼성전자	5930	삼성전자 뻔하지만 당연한 최고의 선택	1
2	2023-01-09	삼성전자	5930	삼성전자 4Q22 잠정실적 리뷰	1
127	2023-04-10	삼성전자	5930	삼성전자 1Q23 잠정실적 리뷰	0
128	2023-04-10	삼성전자	5930	삼성전자 1Q23P Samsung Pivot 발생 그 이후	0
129	2023-02-01	삼성전자	5930	삼성전자 좀더 구체화된 감산 계획	0

② 삼성전자 종목 외의 데이터 제거

③ '제목' 컬럼의 데이터 값 안의 '삼성전자' 단어 삭제

II 모델 플로우

데이터 수집 → **데이터 전처리** → train/test 분리 → 모델링 → 결과

[개별 전처리 과정]

	1) EDA	2) 조건 설정	3) 형태소 분석	4) 벡터화
A조	Word Cloud	• 한글 외 모든 문자,기호 삭제 • 한글자 형태소 삭제	Okt / Mecab	Keras / TF-IDF CV / Word2Vec
B조	Word Cloud	• 한글, 영어 외 모든 기호 삭제 • 한글자 형태소 삭제	Okt / Mecab	Keras / TF-IDF CV / Word2Vec
C조	Word Cloud	• 한글, 영어 외 모든 기호 삭제 • 한글자 형태소 보존 • 자체 불용어 리스트 적용	Okt / Mecab / Hannanum	Keras / TF-IDF CV / Word2Vec
Sentence Transformer	Word Cloud	ko-sroberta-multitask 사전모델 적용		

Ⅱ 모델 플로우

데이터 수집 → **데이터 전처리** → train/test 분리 → 모델링 → 결과

[Code Review]

: C조 - 자체 불용어 리스트 적용 코드

```
# 불용어 리스트 생성
f = open('stop_words.txt', encoding = 'utf8')
reader = csv.reader(f)
stopwords = []
for i in reader:
    stopwords.append(i)
f.close()
```

Rank NL 사이트의 Korean Stopwords

Korean Stopwords

아	어찌됐든	하기보다는
휴	그위에	차라리
아이구	게다가	하는 편이 낫다
아이쿠	점에서 보아	흐흐
아이고	비추어 보아	놀라다
어	고려하면	상대적으로 말하
나	하게될것이다	자면
우리	일것이다	마치
저희	비교적	아니라면
따라	좀	쉿
의해	보다더	그렇지 않으면
을	비하면	그렇지 않다면
를	시키다	안 그러면
에	하게하다	아니었다면
의	할만하다	하든지
가	의해서	아니면

+

탐색 후 추가한 Stopwords

ㄱ	ㅓ	
ㄴ	ㅋ	
ㄷ	ㅗ	
ㄹ	ㅛ	
ㅁ	ㅜ	
ㅂ	ㅠ	
ㅅ	ㅡ	
ㅇ	ㅣ	
ㅈ	ㅔ	
ㅊ	ㅐ	
ㅋ	ㅒ	
ㅌ	ㅖ	
ㅍ	ㄴ다	
ㅎ	ㅂ니다	
ㅏ	ㄹ까	
ㅑ	ㄹ수록	
	ㅂ시다	

총 715개의 stopwords

II 모델 플로우

데이터 수집 → 데이터 전처리 → **train/test 분리** → 모델링 → 결과

```
### train_test_split
from sklearn.model_selection import train_test_split

# 평가용과 테스트 세트로 나누기
X_train, X_test, y_train, y_test = train_test_split(df_final.제목, df_final.label, test_size=0.2, random_state=11)

# 출력을 확인하기 위해 데이터 개수 출력
print("Train set 개수:", len(X_train))
print("Test set 개수:", len(X_test))
```

```
print(X_train.index)
print(y_train.index)
```

```
train_index = list(X_train.index)
test_index = list(X_test.index)
```

```
b_y_train = df_final.label[train_index]
b_y_test = df_final.label[test_index]
```

```
b_y_train = 'b_y_train.csv'
b_y_test = 'b_y_test.csv'

b_y_train_df = pd.DataFrame(y_train)
b_y_train_df.to_csv(b_y_train, index=False)
b_y_test_df = pd.DataFrame(y_test)
b_y_test_df.to_csv(b_y_test, index=False)
```

```
b_X_train = df_final.label[train_index]
b_X_test = df_final.label[test_index]
```

```
b_X_train = 'b_X_train.csv'
b_X_test = 'b_X_test.csv'

b_X_train_df = pd.DataFrame(b_X_train)
b_X_train_df.to_csv(b_X_train, index=False)
b_X_test_df = pd.DataFrame(b_X_test)
b_X_test_df.to_csv(b_X_test, index=False)
```

① 개별 전처리한 데이터 => 80% Train set / 20% Test set
(test_size=0.2. random_state=11)

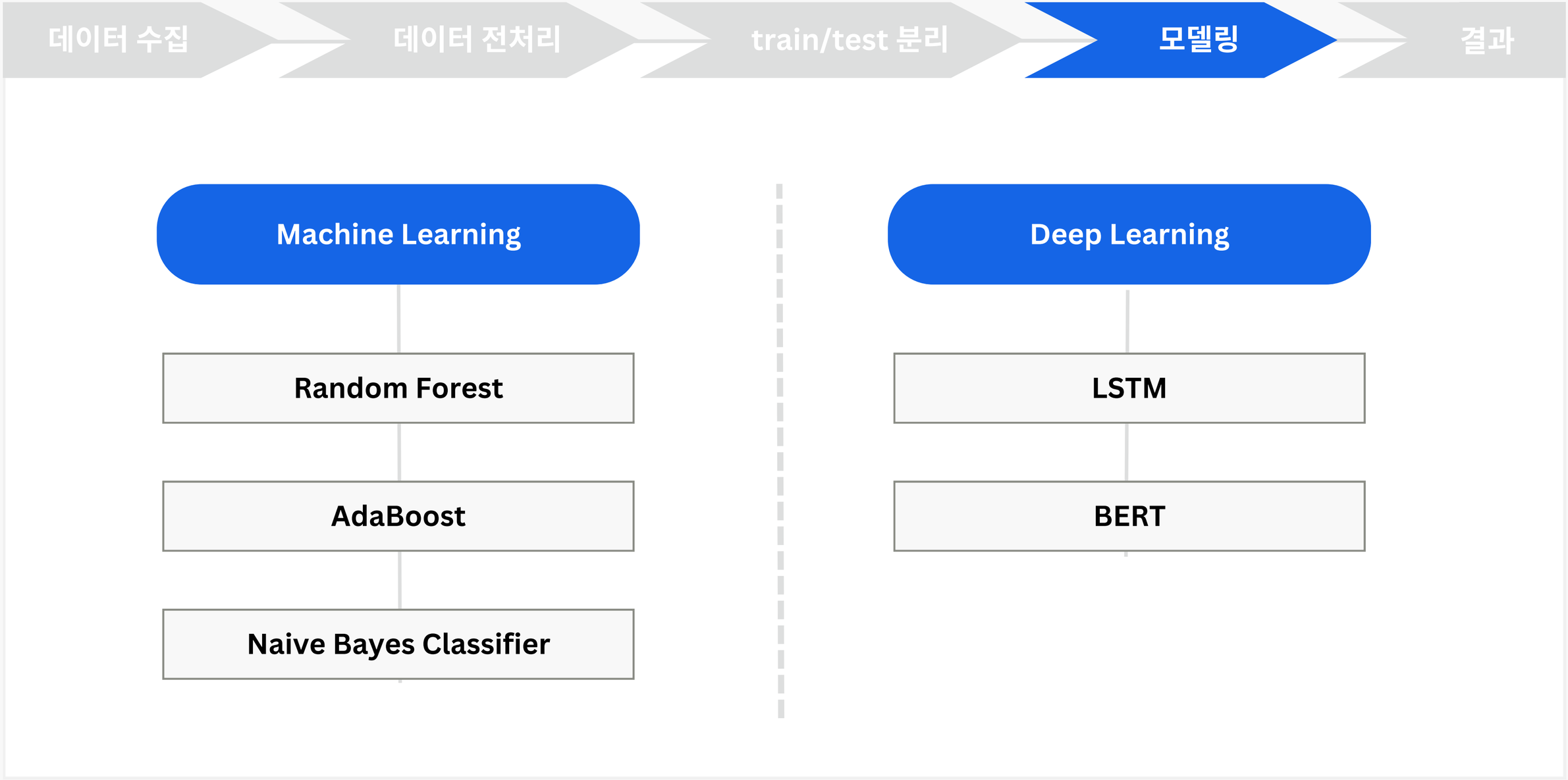
② 모델 별 train/test 데이터 통일을 위해 인덱스 일치 작업 진행

③ Bert 모델용 => 형태소 분석 데이터 ①과 동일하게 분리

▼

A조 : train set-174개, test set-44개
B조 : train set-180개, test set-46개
C조: train set-180개, test set-46개
SentenceTransformer
: train set-180개, test set-46개

II 모델 플로우


데이터 수집
데이터 전처리
train/test 분리
모델링
결과
Machine Learning
Random Forest
AdaBoost
Naive Bayes Classifier
Deep Learning
LSTM
BERT

II 모델 플로우

데이터 수집 → 데이터 전처리 → train/test 분리 → **모델링** → 결과

[Code Review]

: Naive Bayes Classifier

```
### 나이브 베이즈 분류 모델 생성

# 필요한 라이브러리 임폴트
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
X_train_wv = scaler.fit_transform(X_train_wv)
X_test_wv = scaler.fit_transform(X_test_wv)

# 모델 객체 생성
nb = MultinomialNB()
```

[MinMaxScaler를 사용한 이유]

‘Negative values in data passed to MultinomialNB (input X)’
라는 음수값이 존재한다는 오류가 발생

▼

MinMaxScaler를 사용해서 0에서 1까지의 범위로
정규화하여 음수값을 제거함.

II 모델 플로우

Random Forest

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → **결과**

step1	step2	step3	accuracy
a	okt218	keras	0.8182
		tfidf	0.5682
		countvector	0.5682
		word2vec	0.6136
	mecab218	keras	0.8409
		tfidf	0.5909
		countvector	0.5682
		word2vec	0.6364
b	okt226	keras	0.8478
		tfidf	0.5652
		countvector	0.6522
		word2vec	0.6522
	mecab226	keras	0.7391
		tfidf	0.6522
		countvector	0.6957
		word2vec	0.6739
c	okt226	keras	0.6304
		tfidf	0.6087
		countvector	0.6739
		word2vec	0.6522
	mecab226	keras	0.6087
		tfidf	0.6087
		countvector	0.6739
		word2vec	0.5652
	hannaum226	keras	0.6522
		tfidf	0.5870
		countvector	0.6087
		word2vec	0.5652
sentencetransformer			0.6956521739

Best Result

B조 전처리 + Okt + Keras

II 모델 플로우

AdaBoost

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

step1	step2	step3	accuracy
a	okt218	keras	0.8636
		tfidf	0.5682
		countvector	0.5682
		word2vec	0.5000
	mecab218	keras	0.8182
		tfidf	0.5000
		countvector	0.5682
		word2vec	0.5909
b	okt226	keras	0.8478
		tfidf	0.5870
		countvector	0.6739
		word2vec	0.6304
	mecab226	keras	0.9130
		tfidf	0.5870
		countvector	0.5870
		word2vec	0.6304
c	okt226	keras	0.6087
		tfidf	0.6087
		countvector	0.6957
		word2vec	0.4348
	mecab226	keras	0.5652
		tfidf	0.5870
		countvector	0.5870
		word2vec	0.6304
	hannaum226	keras	0.5870
		tfidf	0.5870
		countvector	0.5652
		word2vec	0.6522
sentencetransformer			0.6739130435

Best Result

B조 전처리 + Mecab + Keras

II 모델 플로우

Naive Bayes Classifier

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → **결과**

step1	step2	step3	accuracy
a	okt218	keras	0.4091
		tfidf	0.5682
		countvector	0.5682
		word2vec	0.5909
	mecab218	keras	0.5909
		tfidf	0.5909
		countvector	0.5682
		word2vec	0.6591
b	okt226	keras	0.4348
		tfidf	0.6522
		countvector	0.6957
		word2vec	0.6739
	mecab226	keras	0.5435
		tfidf	0.5870
		countvector	0.6304
		word2vec	0.6739
c	okt226	keras	0.5217
		tfidf	0.6522
		countvector	0.7174
		word2vec	0.5870
	mecab226	keras	0.5217
		tfidf	0.7174
		countvector	0.6522
		word2vec	0.6522
	hannaum226	keras	0.5217
		tfidf	0.6304
		countvector	0.6304
		word2vec	0.5435
sentencetransformer			0.7173913043

Best Result

SentenceTransformer

II 모델 플로우

LSTM

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

step1	step2	step3	accuracy
a	okt218	keras	0.5538
		tfidf	0.5308
		countvector	0.5308
		word2vec	0.5308
	mecab218	keras	0.5615
		tfidf	0.5308
		countvector	0.5308
		word2vec	0.5308
b	okt226	keras	0.5556
		tfidf	0.5333
		countvector	0.5333
		word2vec	0.5333
	mecab226	keras	0.5333
		tfidf	0.5333
		countvector	0.5333
		word2vec	0.5333
c	okt226	keras	0.5333
		tfidf	0.5333
		countvector	0.5333
		word2vec	0.5333
	mecab226	keras	0.5333
		tfidf	0.5333
		countvector	0.5333
		word2vec	0.5333
	hannaum226	keras	0.5333
		tfidf	0.5333
		countvector	0.5333
		word2vec	0.5333
sentencetransformer			

Best Result

A조 전처리 + Mecab + Keras

II 모델 플로우

BERT

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

step1	step2	step4	step5 결과: 평가 지표	
		train_test_split	accuracy	loss
a	okt218	a_okt_bert_train a_okt_bert_test	0.6364	0.6182
	mecab218	a_mecab_bert_train a_mecab_bert_test	0.7045	0.6265
b	okt226	b_okt_bert_train b_okt_bert_test	0.6957	0.5276
	mecab226	b_mecab_bert_train b_mecab_bert_train	0.6957	0.5893
c	okt226	c_okt_bert_train c_okt_bert_test	0.6739	0.658
	mecab226	c_mecab_bert_train c_mecab_bert_test	0.6522	0.6745
	hannaum226	c_hannanum_bert_train c_hannanum_bert_test	0.7609	0.503

Best Result

C조 전처리 + Hannanum

II 모델 플로우

성능 개선

데이터 수집 → 데이터 전처리 → train/test 분리 → 모델링 → 결과

1차 사이클 설정값					
step	models	params			
3	keras	max_len=9			
	word2vec	vector_size=300	window=5	min_count=1	sg=1
	countvectorizer	max_features=10000			
4	train_test_split	test_size=0.2	random_state=11		
5	RF	random_state=11			
	나이브베이즈	random_state=11			
	AdaBoost	random_state=11			
	LSTM	batch_zise=256	epochs=20	validation_split = 0.25	max_len의 경우 형태소별로 다름
	Bert	batch_zise=128	epochs=5	random_seed=11	max_len의 경우 형태소별로 다름

▼ 매개변수 조정 & GridSearchCV를 통한 성능 개선 시도 ▼

2차 사이클 설정값					
step	models	params			
3	keras	max_len=9			
	word2vec	vector_size=300	window=5	min_count=1	sg=1
	countvectorizer	max_features=10000			
4	train_test_split	test_size=0.2	random_state=11		
5	RF	random_state=11	n_estimators = 50~500	max_depth= 1~10	min_samples_leaf = 1~ 6
	나이브베이즈	alpha=0.01~1	cv=15		
	AdaBoost	random_state=11	n_estimators = 50~500		
	LSTM	batch_zise=128	epochs=30	validation_split = 0.25	max_len의 경우 형태소별로 다름
	Bert	batch_zise=16	epochs=10	random_seed=11	max_len의 경우 형태소별로 다름

Result

모든 모델에서
유의미한 성능 개선 X

II 모델 플로우

실전 적용 모습


```
news_predict(news, model='rf')

모델: rf
분류 할 제목: 뚝뚝 떨어진 D램 가격… 삼성전자·SK하이닉스, 2분기도 어둡다
토큰화 된 제목: ['뚝뚝', '떨어지다', '램', '가격', '삼성', '전자', '하이닉스', '분기도', '어둡다']
임베딩 된 제목: [[493 494 495  48 496  78 497 498 499]]
하향

news_predict(news, model='nb')

모델: nb
분류 할 제목: 뚝뚝 떨어진 D램 가격… 삼성전자·SK하이닉스, 2분기도 어둡다
토큰화 된 제목: 토큰화 별도로 없음
임베딩 된 제목: 길어서 생략
하향

news_predict(news, model='adaboost')

모델: adaboost
분류 할 제목: 뚝뚝 떨어진 D램 가격… 삼성전자·SK하이닉스, 2분기도 어둡다
토큰화 된 제목: ['뚝뚝', '떨어진', '램', '가격', '삼성전자', '하이닉스', '분기', '도', '어둡', '다']
임베딩 된 제목: [[440 441  59 442 443   2 444 445 446]]
하향

[ ] news_predict(news, model='lstm')

모델: lstm
분류 할 제목: 뚝뚝 떨어진 D램 가격… 삼성전자·SK하이닉스, 2분기도 어둡다
토큰화 된 제목: ['뚝뚝', '떨어진', '램', '가격', '삼성전자', '하이닉스', '분기', '도', '어둡', '다']
임베딩 된 제목: [[371 372  51 373 374   2 375 376 377]]
1/1 [==============================] - 1s 853ms/step
하향

news_predict(news, model='bert')

모델: bert
분류 할 제목: 뚝뚝 떨어진 D램 가격… 삼성전자·SK하이닉스, 2분기도 어둡다
토큰화 된 제목: ['떨 지 D 램 가격… 삼성전자·SK하이닉스, 2분기 어둡 다']
임베딩 된 제목: 없음
1/1 [==============================] - 9s 9s/step
하향
```

II 모델 플로우

실전 적용 모습

[Code Review]

```
def news_predict(news, model):


  if model == 'rf':

    print(f'모델: rf')

    # 텍스트 토큰화
    okt = Okt()
    news_title = re.sub(r'[^ㄱ-ㅎㅏ-ㅣ가-힣 ]', '', news)
    news_title = okt.morphs(news_title, norm=True, stem=True)

    print(f'분류 할 제목: {news}')
    print(f'토큰화 된 제목: {news_title}')


    # 모델 경로
    classifier_path = '/content/drive/MyDrive/KDT/비정형텍스트분석/KDT_1차프로젝트/뉴스 모델링 적용/RandomForest 모델/b_okt_keras_rf_saved_model1.pkl'
    prepro_path = '/content/drive/MyDrive/KDT/비정형텍스트분석/KDT_1차프로젝트/뉴스 모델링 적용/b_okt_keras.pickle'

    # 학습된 임베딩 모델 불러오기
    with open(prepro_path, 'rb') as f:
      a_prepro_model = pickle.load(f)

    # 불러온 임베딩 모델로 처리
    a_prepro_model.fit_on_texts(news_title)
    encoded = a_prepro_model.texts_to_sequences([news_title])
    final_sentence = pad_sequences(encoded, maxlen=9)
    print(f'임베딩 된 제목: {final_sentence}')

    # 학습된 분류 모델 불러오기
    classifier_model = joblib.load(classifier_path)

    # 불러온 분류모델에 임베딩된 뉴스 제목 삽입
    result = float(classifier_model.predict(final_sentence))

    if(result == 1):
      print("상향")
    else:
      print("하향")

```

[실전 적용 함수 코드 작성]

① 학습된 분류 모델, 토크나이저, 데이터셋, 불용어 리스트 등 준비

② input 데이터 입력
- 사용자가 입력한 뉴스 기사 제목 -> news 변수
- 사용자가 선택한 모델 -> model 변수

③ if 문으로 입력된 모델 선택

④ 선택된 모델과 동일한 방식으로 뉴스 제목 전처리

⑤ 학습된 분류 모델 불러오기 및 학습

⑥ label=1 -> 상향 / label=0 -> 하향 출력

Ⅲ요약 및 시사점

SUMMARY

본 프로젝트는 뉴스 제목으로 주식의 상향/하향을 분류하는 모델을 개발하는 것이 주제

3개의 조로 나누고 SentenceTransformer를 이용해 총 4가지 방식으로 데이터 전처리를 진행함

RandomForest, 나이브베이즈, AdaBoost, LSTM, BERT 모델을 이용하여 각 모델별로 성능을 비교함

실험결과, b조의 전처리+Mecab+Keras+AdaBoost모델의 가장 높은 accuracy 점수를 보임.
그러나 실전 적용에서는 SentenceTransformer+Naive Bayes Classifier가 가장 잘 분류함.

Ⅲ요약 및 시사점

IMPLICATION

GridSearchCV와 파라미터 조정 등 성능 개선을 시도했지만 유의미한 결과가 나오지 않음.

데이터셋이 237개로 작아 학습이 잘 되지 않아 전반적으로 모델 성능이 뛰어나지 않음.

학습에 활용된 한경 컨센서스의 데이터 중 상향/하향 또한 "예측"에 불과하고 실제 상향/하향과 차이가 있음

리포트와 뉴스의 차이점을 잘 반영하지 못해 분류가 잘 되지 않는 모델이 있음

보완점 : 본 프로젝트에선 데이터셋을 늘려 진행하면 성능이 나아질 것으로 예상

Ⅳ 개발 후기

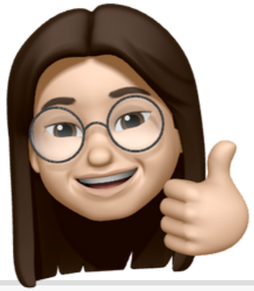
보고있나 세종대왕
보고있나 뉴턴

텍스트 전처리 방법을 다양하게 쓴 나머지 경우의 수가 많아져서 관리하기가 어려웠습니다...

텍스트 데이터 분석은 양이 많을 때 하는게 가장 좋다는 걸 배웠네요!

수업에서 배웠던 코드를 사용해 분석해볼 수 있어서 좋았습니다!

앞으로 할 본 프로젝트에서는 배운 것들을 더 적용시켜 세미프로젝트보다 더 즐기면서 하고 싶습니다



세미프로젝트지만 저에게는 첫 프로젝트인만큼 그 성취감이 배로 다가오는 것 같습니다.

적절한 데이터를 수집하는 것이 가장 중요하고 어렵다는 것을 알 수 있었습니다.

python을 배우고 처음했던 프로젝트여서 걱정이 많았지만, 팀원 분들이 잘 이끌어주셔서 프로젝트를 잘 마무리할 수 있었습니다.

1조 팀원분들 너무 감사했습니다!

Q&A