Fake News Detection

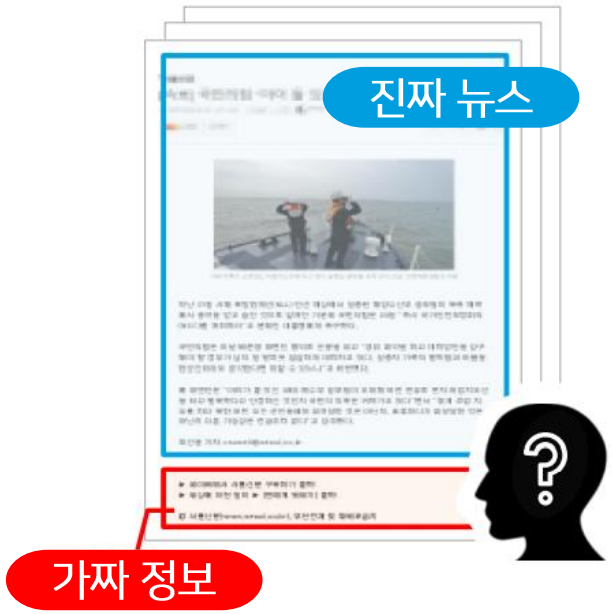
AI야, 진짜 뉴스를 찾아줘!

이정민 이주희 이혜림

contents

1. 프로젝트 소개

프로젝트 소개



AI야, 진짜 뉴스를 찾아줘!

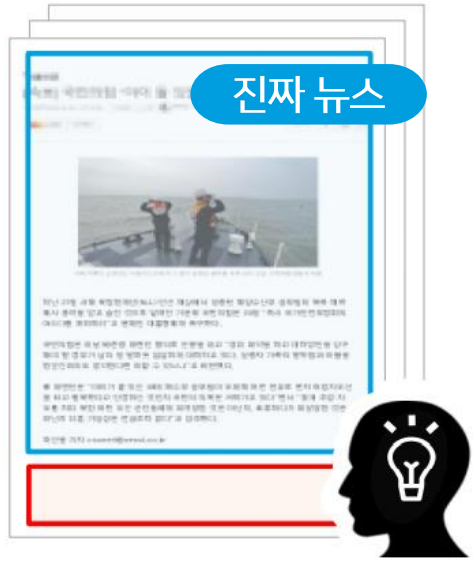
AI 뉴스 필터링 알고리즘

진짜 뉴스 내용 연관성

가짜 정보 위치

가짜 정보 키워드

….



가짜 정보 제거 !

뉴스 본문에서 **가짜 정보를 제거**하고 진짜 뉴스만 찾는 알고리즘 개발

→ AI 뉴스 필터링 알고리즘 개발을 통해 고객이 필요로 하는 **" 진짜 뉴스 " 만 제공**

학습 데이터

[5개의 컬럼]
- 뉴스 Index 번호
- 발행 날짜
- 제목
- 내용
- 뉴스 내용 순서

[Target] 진짜 뉴스 유무 0 → 진짜
1 → 가짜

n_id	date	title	content	ord	info
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	[이데일리 MARKETPOINT]15:32 현재 코스닥 기관 678억 순매도	1	0
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	"실적기반" 저가에 매집해야 할 8월 급등유망주 TOP 5 전격공개	2	1

진짜 뉴스

가짜 뉴스

⋮

NEWS09727	20200626	롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참	한편 앞서 중기부에 따르면 지난 25일 글로벌 쇼트 비디오 앱 틱톡(Tiktok)의 공익캠페인…	12	0
NEWS09727	20200626	롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참	"실적기반" 저가에 매집해야 할 8월 급등유망주 TOP 5 전격공개	13	1
NEWS09727	20200626	롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참	하이스탁론, 선취수수료 없는 월 0.4% 최저금리 상품 출시	14	1

2. EDA & Feature 추가

EDA & feature 추가

[content에 자주 나오는 단어]

REAL NEWS

FAKE NEWS

EDA & feature 추가

title_noise

FAKE NEWS의 title 에 자주 나오는 단어

- 특수문자: '※', '■', '★', '』'
- 영어: 'TOP', 'BEST'
- 단어: '전문가의 눈', '전문가선정', '전문가의견', '전문가 추천', '주요이슈'
- 뒤에 주로 나오는 단어: '종목', '관련주'
- 문장: '후속주도 감사합니다'

content_noise

FAKE NEWS의 content 에 자주 나오는 단어

- 특수문자: '#', '00%'
- 영어: 'TOP', 'BEST'
- 단어: '긴급공개', '긴급 공개', '임상3상', '대장株', '대장주', '카톡', '원"만', '관련기사', '관련 테마분석', '코스피', '코스닥'

EDA & feature 추가

title_noise

FAKE NEWS의 title 에 자주 나오는 단어

- 특수문자: '※', '■', '★', '』'
- 영어: 'TOP', 'BEST'
- 단어: '전문가와 논', '전문가선정', '전문가인터', '전문가 추천', '주요이슈'
- 뒤에 주로 나오는 단어: '종목', '관련주'
- 문장: '후속주도 감사합니다'

content_noise

FAKE NEWS의 content 에 자주 나오는 단어

- 특수문자: '#', '00%'
- 영어: 'TOP', 'BEST'
- 단어: '긴급공개', '긴급 공개', '임상3상', '대장株', '대장주', '카톡', '원"만', '관련기사', '관련 테마분석', '코스피', '코스닥'

→ 위의 단어들이 나오는 경우 90% 이상이 FAKE NEWS

EDA & feature 추가

[Feature 추가]

"info1_title"

FAKE NEWS의 title에 자주 나오는 단어인 title_noise를 포함하고 있을 경우, 1로 채우기
(title_nosise를 포함할 경우, 약 90% 이상이 FAKE news)

"info1_content"

FAKE NEWS의 content에 자주 나오는 단어인 content_noise를 포함하고 있을 경우, 1로 채우기

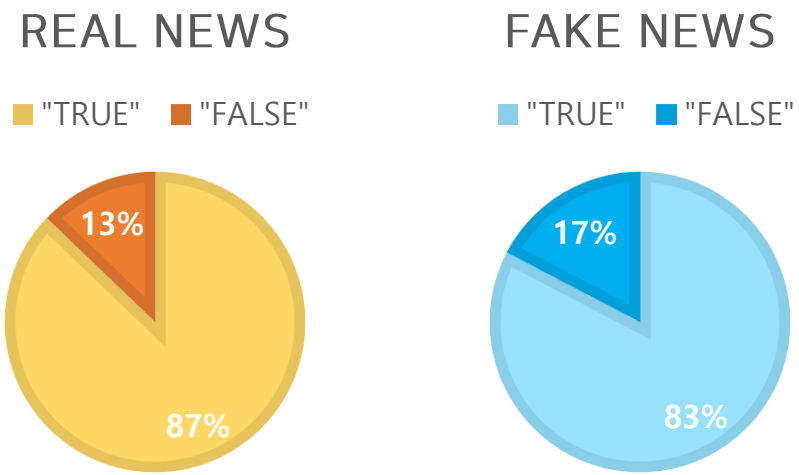
EDA & feature 추가

[인접한 기사들끼리 info가 유사]

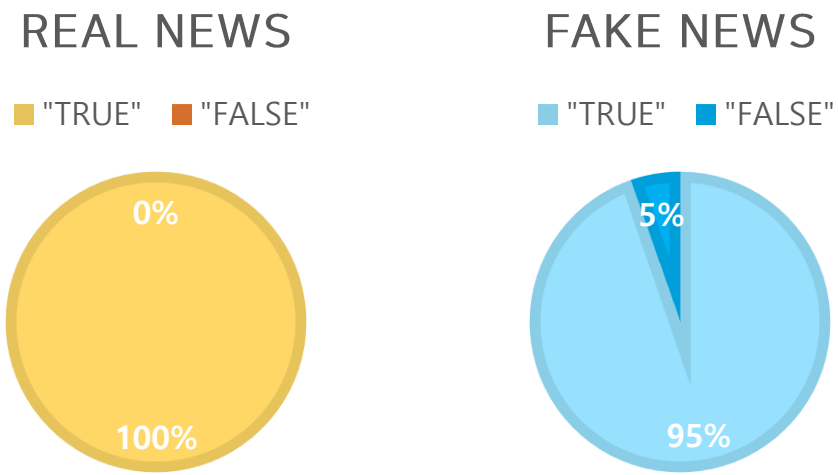
n_id	date	title	content	ord	info
NEWS00695	20200522	BMW코리아, 온라인 한	"실적기반" 저가에 매집해야 할 8월 급등	13	1
NEWS00695	20200522	BMW코리아, 온라인 한	하이스탁론, 선취수수료 없는 월 0.4% 최	14	1
NEWS00695	20200522	BMW코리아, 온라인 한	종합 경제정보 미디어 이데일리 - 무단전	15	0
NEWS07957	20200414	온라인 결제株, 코로나19	온라인결제 관련주가 코로나19 사태의 초	1	1
NEWS07957	20200414	온라인 결제株, 코로나19	한편, 스탁론에 대한 관심이 날로 높아지	2	1
NEWS07957	20200414	온라인 결제株, 코로나19	"실적기반" 저가에 매집해야 할 8월 급등	3	1
NEWS07957	20200414	온라인 결제株, 코로나19	하이스탁론, 선취수수료 없는 월 0.4% 최	4	1
NEWS07957	20200414	온라인 결제株, 코로나19	종합 경제정보 미디어 이데일리 - 무단전	5	0
NEWS03718	20200515	거래소 "이에스브이 실질	[이데일리 박태진 기자] 한국거래소 코스	1	0
NEWS03718	20200515	거래소 "이에스브이 실질	거래소는 코스닥시장상장규정 제38조제2	2	0
NEWS03718	20200515	거래소 "이에스브이 실질	이에 따라 거래소는 다음달 5일까지 동사	3	0
NEWS03718	20200515	거래소 "이에스브이 실질	"실적기반" 저가에 매집해야 할 8월 급등	4	1
NEWS03718	20200515	거래소 "이에스브이 실질	하이스탁론, 선취수수료 없는 월 0.4% 최	5	1
NEWS09727	20200626	롯데·공영 등 7개 TV 홈	현장 이원 생방송의 다음 주자로는, 공영	10	0
NEWS09727	20200626	롯데·공영 등 7개 TV 홈	박영선 장관은 이번 동행세일 행사에 TV	11	0
NEWS09727	20200626	롯데·공영 등 7개 TV 홈	한편 앞서 중기부에 따르면 지난 25일 글	12	0

EDA & feature 추가

title이 [또는 (로 시작하는 경우

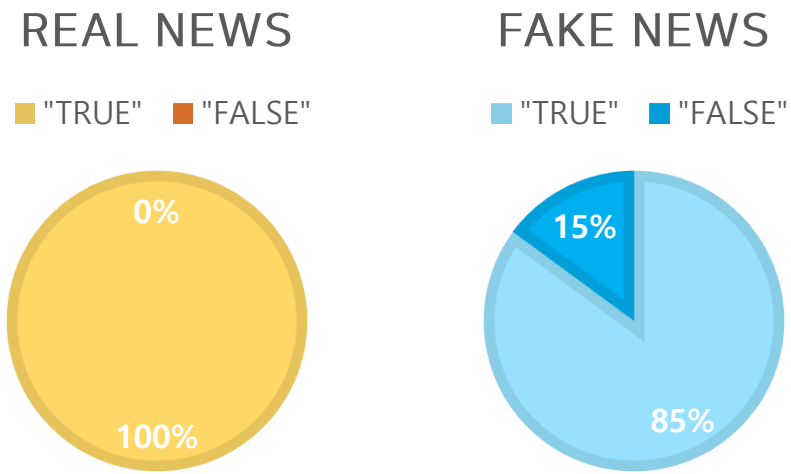


content가 [또는 (로 시작하는 경우



EDA & feature 추가

Content가 “제목“으로 시작하는 경우



+

“content_startswith"

content가 “[” 또는 “(” 또는 “제목” 으로 시작할 경우,

REAL news인 0으로 채우고,

아닐 경우 FAKE news인 1로 채우기

EDA & feature 추가

"new_ord"

= x["ord"]/title_group[x["title"]]

title_group

같은 title을 가진 경우, 각 title에 대해 그 수를 count

```
title
"10세 미만 어린이, 날 음식 먹이지 마세요" 서울대병원 의사의 당부      31
"3년 전 분양가로 3가구 공급" vs "아파트명 바꾸고 완판"          39
"AR · VR 적용" 실감교육 체험학교 모집                        21
"G7 경제수장 코로나19 대응 성명 초안에 금리인하 요구 없어"         16
"K-스토리 일냈다" 카카오페이지 IP 하루거래액 20억 돌파           28
                                                    ..
```

+

	title	ord	new_ord
[마감]코스닥 기관 678억 순매도		1	0.250000
[마감]코스닥 기관 678억 순매도		2	0.500000
[마감]코스닥 기관 678억 순매도		3	0.750000
[마감]코스닥 기관 678억 순매도		4	1.000000
롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참		1	0.066667
롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참		2	0.133333
롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참		3	0.200000
롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참		4	0.266667
롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참		5	0.333333

EDA & feature 추가

[모델링에 사용된 최종 train 데이터 셋]

제공 데이터

추가한 feature

n_id	date	title	content	ord	info	content_startswith_[	info1_title	info1_content	new_ord
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	[이데일리 MARKETPOINT]15:32 현재 코스닥 기관 678억 순매도	1	0	1	0	0	0.250000
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	"실적기반" 저가에 매집해야 할 8월 급등유망주 TOP 5 전격공개	2	1	0	0	1	0.500000
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	하이스탁론, 선취수수료 없는 월 0.4% 최저금리 상품 출시	3	1	0	0	0	0.750000
NEWS02580	20200605	[마감]코스닥 기관 678억 순매도	종합 경제정보 미디어 이데일리 - 무단전재 & 재배포 금지	4	0	0	0	0	1.000000
NEWS09727	20200626	롯데·공영 등 7개 TV 홈쇼핑들, 동행세일 동참	전국적인 소비 붐 조성에 기여할 예정	1	0	0	0	0	0.066667

y

3. 전처리

불용어 제거
Tokenization
Vectorization
Embedding

불용어 제거

영문 및 숫자 제거

-> 특수문자의 경우 가짜뉴스 판별에 유의미한 요소이기 때문에 삭제하지 않음

단 10초만에 계좌를 붉게만들기 가능하다.=>정보 확인하기 최소 800% 수익 보장. 적극매수
정확하게 매매 타점 제시해드립니다....기회 왔을때 잡으셔야 합니다. 딱 하루. 오늘 하루만 투
무료체험 회원 추천주 "000" 폭등.. 6/26 금요일 개장 즉시 계좌를 바꿔줄 종목 10초면 확인
-씨젠 186% 上 적중.-레몬 93% 上 적중.-YBM넷 223% 上 적중.-파미셀 120% 上 적중.-크리스
금요일 개장즉시 폭발 종목... 오늘까지만 제공하고 조기마감합니다==>신청즉시 문자발송 신

<가짜 뉴스의 특수문자 예시>

기자 이름 제거

-> 기자 이름은 유의미한 token이라고 볼 수 없음

[이데일리 박태진 기자] 한국거래소 코스닥시장본부는 이에스브이(223310)에 대해

<기자 이름 예시>

Tokenization

Mecab

굉장히 속도가 빠르며
좋은 분석 결과를 보여준다.

Komoran

댓글과 같이 정제되지 않은 글에 대해서 먼저 사용해보면 좋다.
(오탈자를 어느정도 고려해준다.)

Kkma

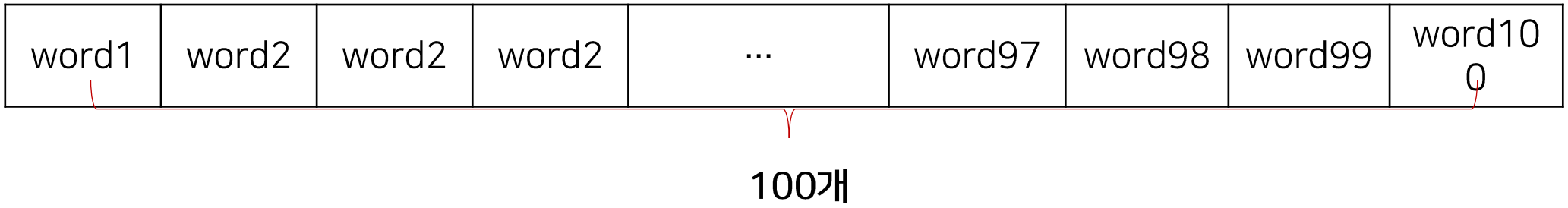
분석 시간이 오래 걸리기 때문에
잘 이용하지 않는다.

Okt

품사 태깅 결과를 Noun, Verb등 알아보기 쉽게 반환해준다.

Vectorization

한 문장 당 100개의 단어를 사용



if1) 100개보다 작다면 앞에서부터 0으로 채움(padding)
if2) 100개보다 많다면 앞의 단어를 반영

-> 총 33519개의 단어 사용

Embedding

Embedding size : 300

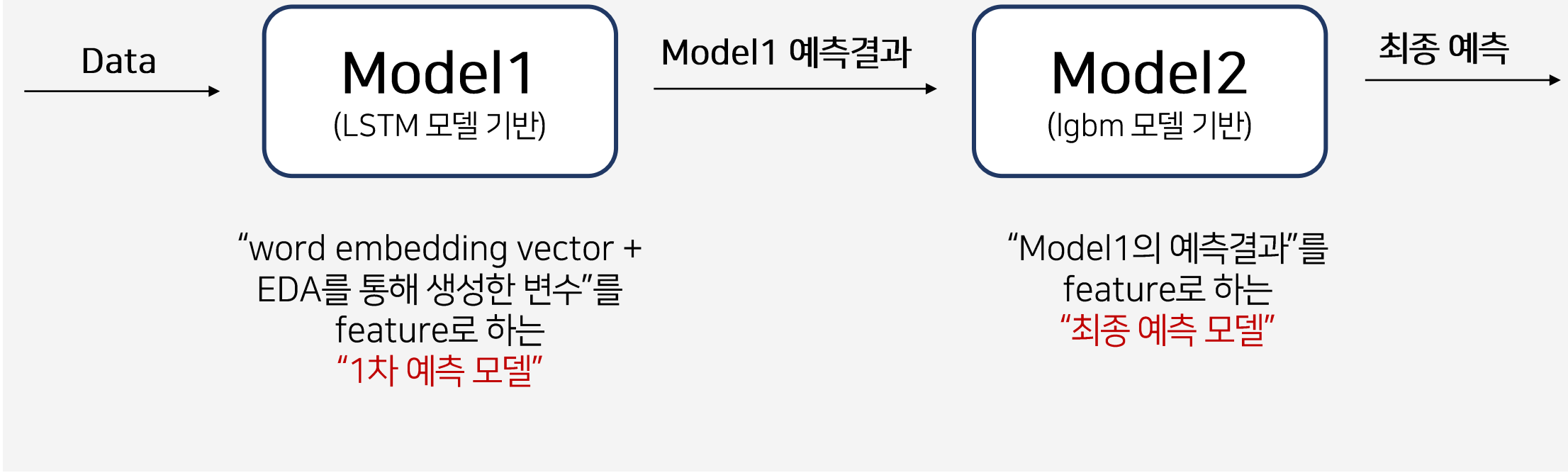
LSTM 모델 내에서 Embedding matrix 학습

- 기존 embedding matrix 사용 혹은 word2vec/glove model 훈련을 통해 생성한 word embedding matrix 보다 더 우수한 성능을 보였기 때문(Base Model 기준)

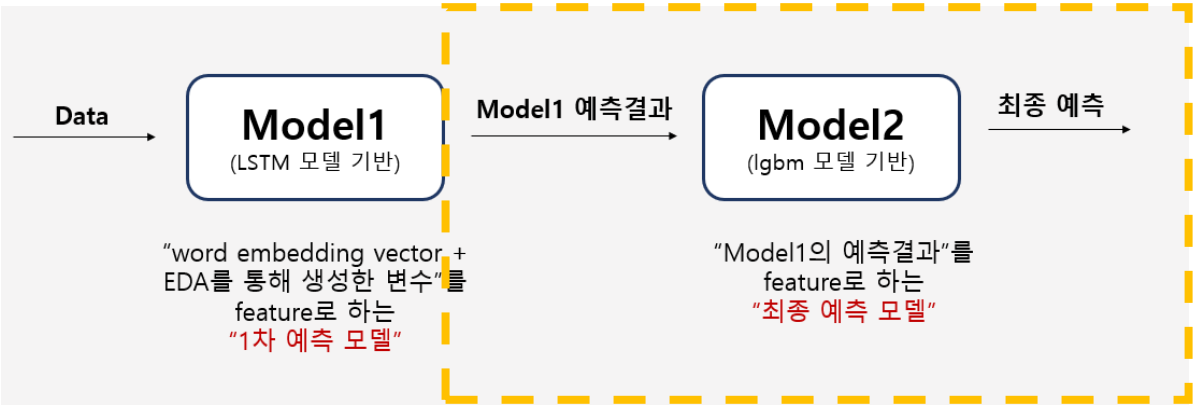
[전처리 과정에서의 Word matrix 크기]

4. 모델링

모델링 개요


Data
Model1
(LSTM 모델 기반)
Model1 예측결과
Model2
(lgbm 모델 기반)
최종 예측
"word embedding vector + EDA를 통해 생성한 변수"를 feature로 하는 "1차 예측 모델"
"Model1의 예측결과"를 feature로 하는 "최종 예측 모델"

모델링 개요



"Model2는 왜 필요할까?"

EDA결과 "인접한 기사들끼리 info가 유사"

↓

인접한 데이터의 info를 이용하면 더 잘 예측가능

↓

Model1의 예측결과를 사용하여 각 데이터의 인접한 데이터의 예측결과를 feature로 하는 데이터셋을 생성, Model2를 통해 최종 예측

RNN & LSTM

RNN이란?

- 시퀀스(Sequence) 모델로, 입력과 출력을 시퀀스 단위로 처리하는 모델.

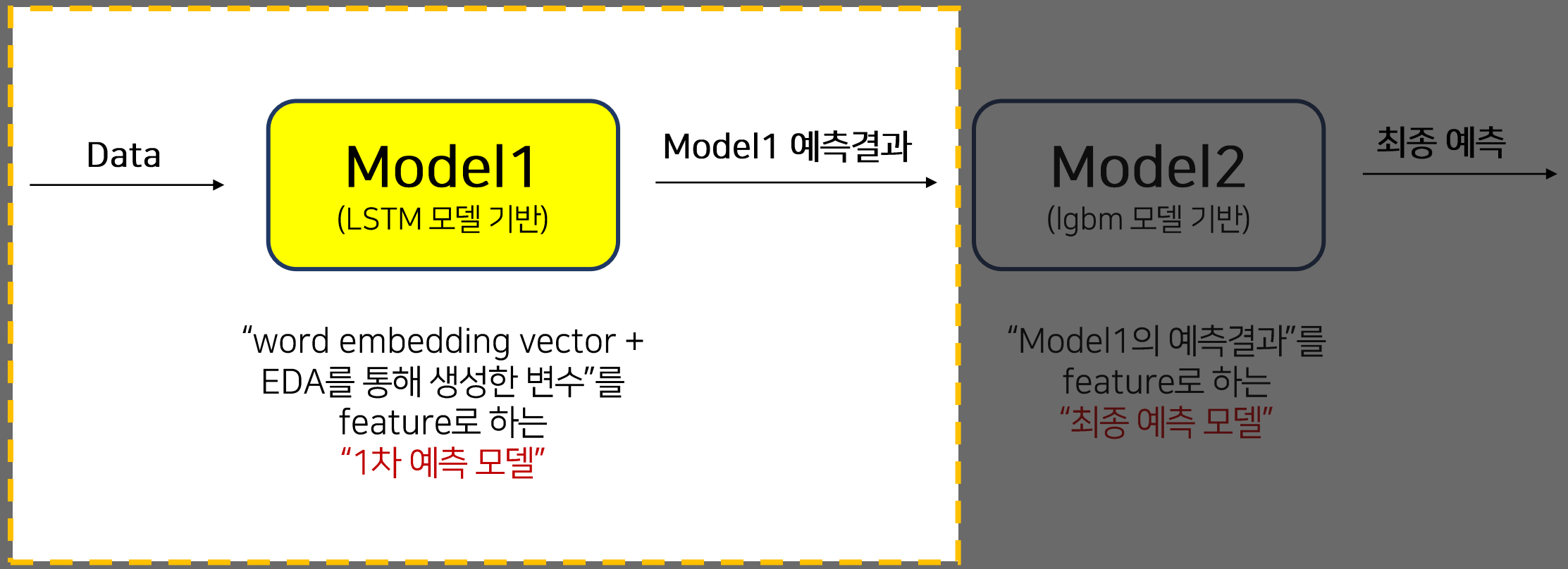
- 은닉층 node의 활성화 함수를 통해 나온 결과값을 출력층 방향으로도 보내면서, 다시 은닉층 노드의 다음 계산의 입력을 보내는 특징

왜 RNN 모델 중 LSTM을 사용했을까?

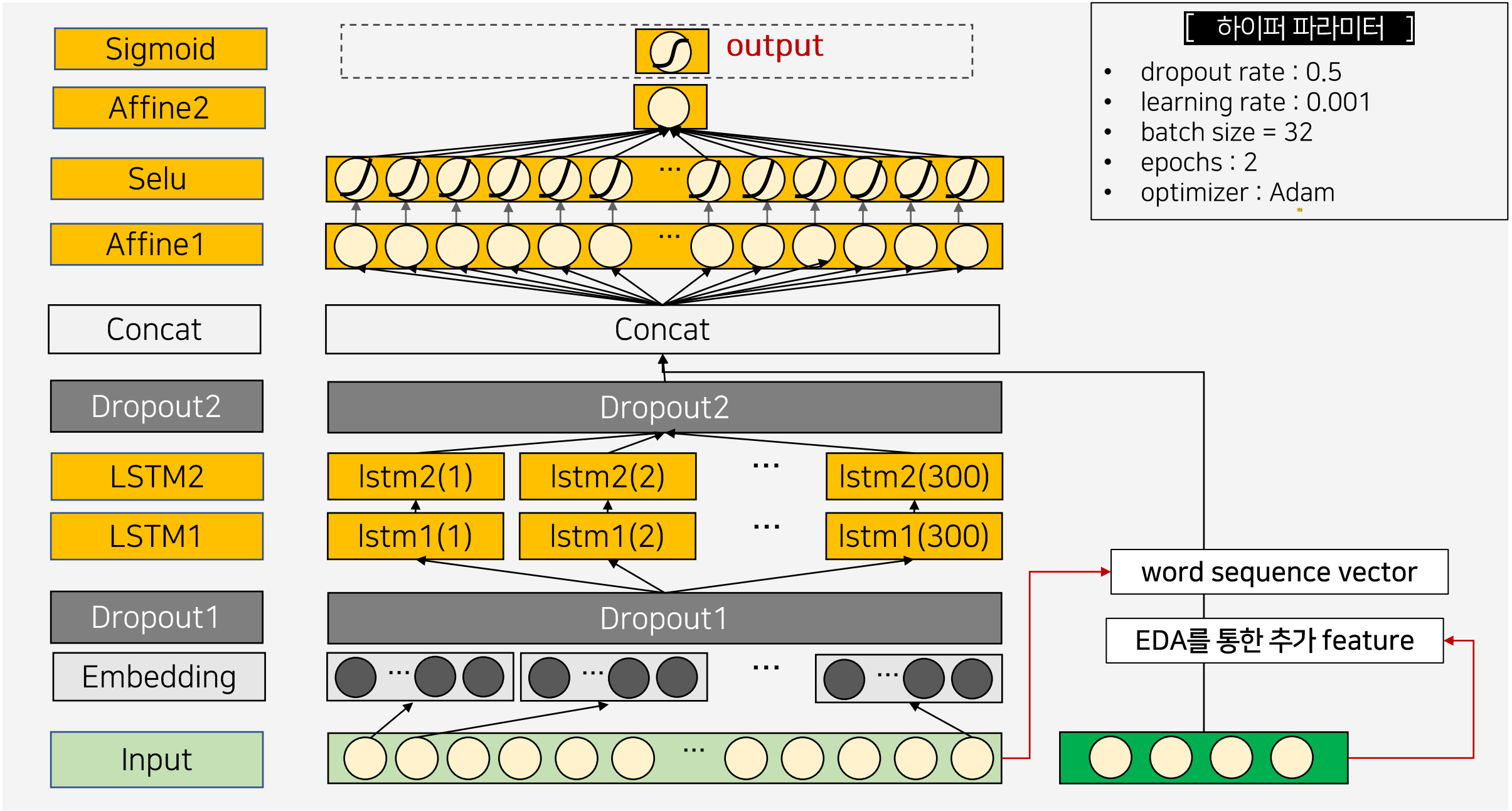
- embedding size가 300으로, 관련 지점과 그 정보를 사용하는 지점 사이의 거리가 멀기 때문에 RNN 사용 시 vanishing gradient problem이 나타날 우려가 있음

-> vanishing gradient problem을 개선한 모델인 LSTM사용

Model1


Data
Model1
(LSTM 모델 기반)
Model1 예측결과
Model2
(lgbm 모델 기반)
최종 예측
"word embedding vector +
EDA를 통해 생성한 변수"를
feature로 하는
"1차 예측 모델"
"Model1의 예측결과"를
feature로 하는
"최종 예측 모델"

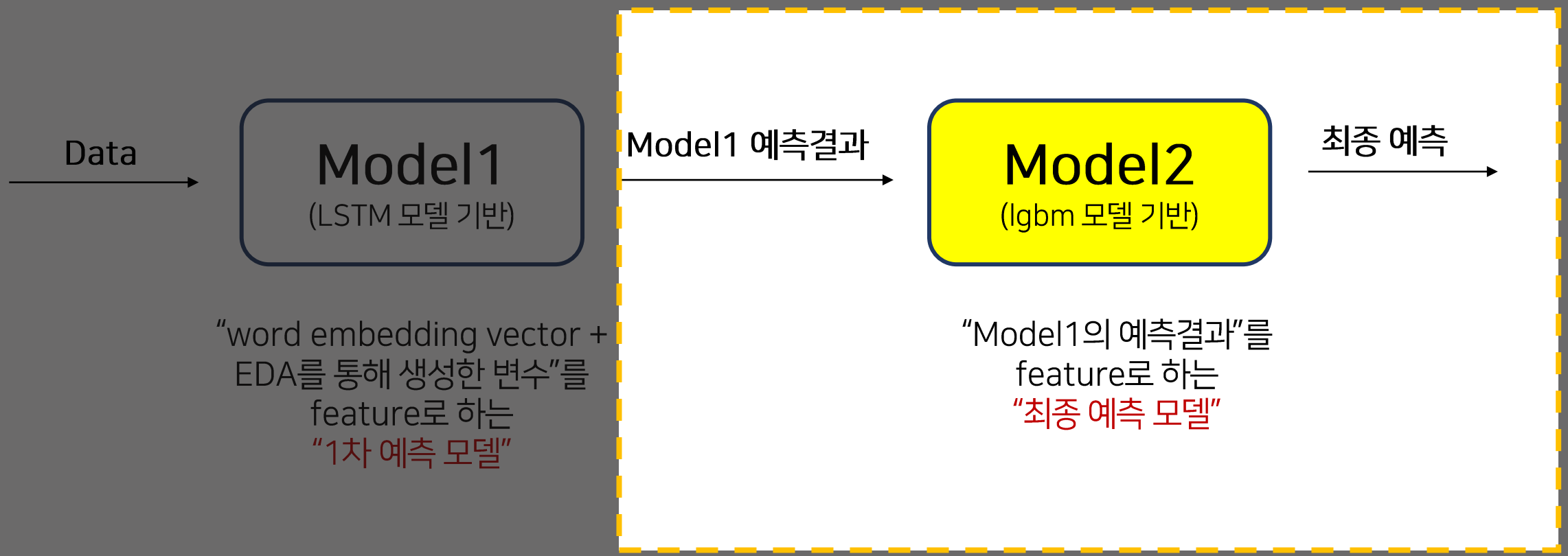
[Model1 구조]



[하이퍼 파라미터]

- dropout rate : 0.5
- learning rate : 0.001
- batch size = 32
- epochs : 2
- optimizer : Adam

Model2


Data
Model1
(LSTM 모델 기반)
Model1 예측결과
Model2
(lgbm 모델 기반)
최종 예측
"word embedding vector + EDA를 통해 생성한 변수"를 feature로 하는 "1차 예측 모델"
"Model1의 예측결과"를 feature로 하는 "최종 예측 모델"

Model2(사용모델)

기존의 Boosting 개념에서
오분류된 데이터들에
더 큰 가중치를 주는 개념 추가

Adaboosting

경사 하강법을 이용하는
기존 Gradient Boosting 모델보다
속도가 빠르고 정규화 항을
추가함으로써 과적합 방지

XGBoosting

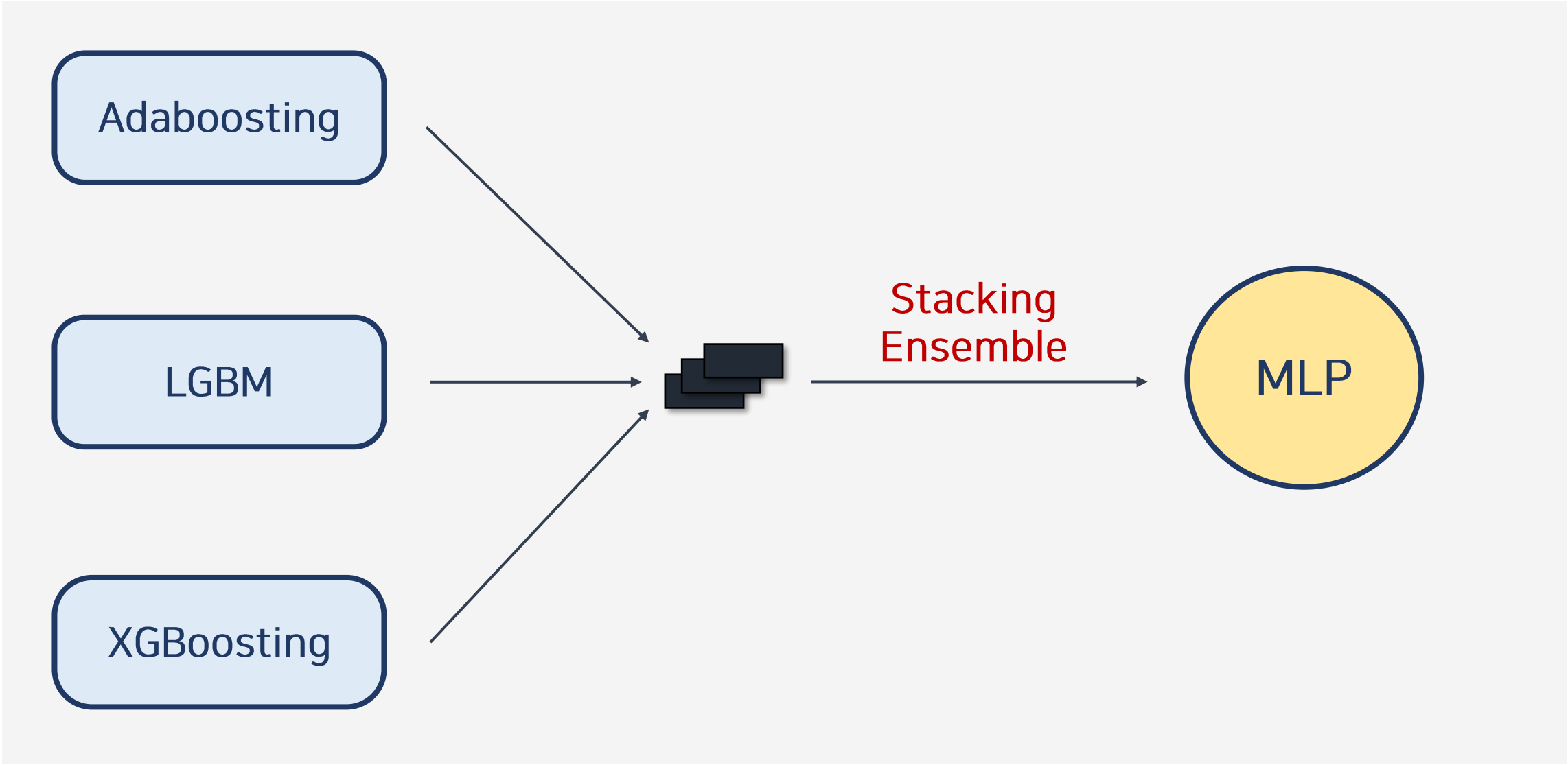
LGBM

기존 Boosting 모델들보다
속도, 성능적으로 개선된 모델

MLP

다층 퍼셉트론으로, 은닉층을
이용한 심층 신경망 모델

Model2(Stacking Ensemble)



Model2(Final model)

But, Stacking Ensemble에서 “과적합” 문제 발생.

최종적으로 LGBM 단일 모델 사용

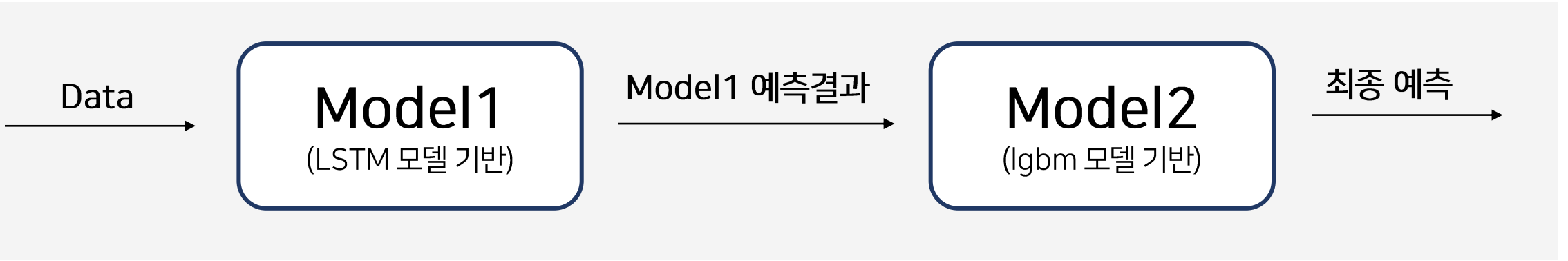
LGBM Classifier

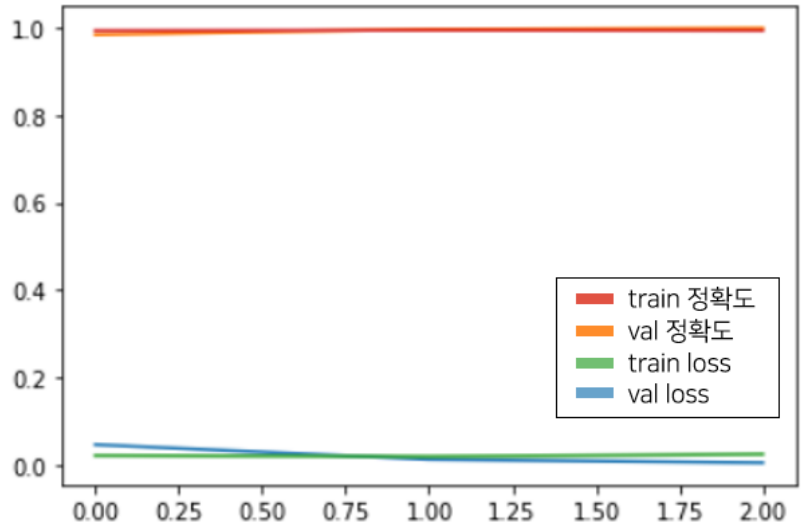
[Grid search를 통해 찾은 하이퍼 파라미터]

- boosting_type : “gbdt”
- learning_rate : 1.0
- max_depth : -1
- min_child_sample : 20
- n_estimators : 100

5. 결과

최종 성능





<model 훈련 과정>

train 정확도 : 99.91
val 정확도 : 99.89

프로젝트 의의

가짜 뉴스 판독에 적합한 Embedding matrix 활용

가짜 뉴스 판독에 적합한 Embedding matrix를 생성했기 때문에,
후에 비슷한 task에서 이를 활용할 수 있다.

고객 편리 제고

개발한 AI 뉴스 필터링 알고리즘을 이용하여 가짜 뉴스를 제거,
고객이 필요로 하는 "진짜 뉴스"만 제공함으로써 사용자의 편리를 제고할 수 있다.

Code

Code

content 맨앞이 [이거나 (이거나 제목이면 0

```
train["content_startswith_["]=train.content.apply(lambda x : str(x).startswith("[" ) or str(x).startswith("(") or str(x).startswith
("제목"))+0
```

타이틀을 이용한 feature

- 해당 title에 몇가지 단어가 들어갈 경우 약 90% 이상이 info가 1

```
title_noise = ['적중 100%', '글로벌 주요 뉴스', '[전문가 의견]',
               '[포커스]', '※','■', '★',' TOP', 'BEST',
'전문가의 눈', '전문가선정', '전문가의견','】 ','후속주도 감사합니다',
               '전문가추천', '주요이슈']

def title_choose(x):
  if ("종목" in x[-6:]) or ("관련주" in x[-5:]):
    return 1
  for noise in title_noise:
    if noise in x.upper():
        return 1
  return 0
```

```
train["info1_title"]=train['title'].apply(title_choose)
```

Code

content를 이용한 feature

```
content_noise = ['00%','긴급공개', '긴급 공개','임상3상','# ','대장株','대장주','카톡','원"만','TOP','BEST']

def content_choose(x):
  if (x=='관련기사') or (x=="관련 테마분석") or (x=="코스피") or (x=="코스닥"):
    return 1
  for noise in content_noise:
    if noise in x.upper():
        return 1
  return 0
```

```
train["info1_content"]=train["content"].apply(content_choose)
```

Order을 이용한 feature

```
title_group = (train.groupby(["title"]).count())["n_id"]
train["new_ord"]=train.apply(lambda x: x["ord"]/title_group[x["title"]], axis=1)
```

Code

tokenization

```
from konlpy.tag import Mecab
import re

def text_preprocessing(text_list):

    tokenizer = Mecab() #형태소 분석기

    token_list = []

    for text in text_list:
        txt = re.sub("[a-zA-ZO-9]", ' ', text) #영문, 숫자 제거 -> 특수문자는 제거하지 않음
        txt = re.sub('[가-힣]+ㅠs기자','기자', txt) #기자 이름 제거
        token = tokenizer.morphs(txt) #형태소 분석

        token = [t for t in token]
        token_list.append(token)

    return token_list, tokenizer

#형태소 분석기를 따로 저장한 이유는 후에 test 데이터 전처리를 진행할 때 이용해야 되기 때문입니다.
train['new_article'], mecab = text_preprocessing(train['content'])
```

```
# 결측치 제거
train = train[train["new_article"].apply(lambda x: False if len(x)==0 else True)]
```

Code

Vectorization

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
max_len = 100

def text2sequence(train_text, max_len=100):

    tokenizer = Tokenizer()
    tokenizer.fit_on_texts(train_text)
    train_X_seq = tokenizer.texts_to_sequences(train_text)
    vocab_size = len(tokenizer.word_index) + 1
    print('vocab_size : ', vocab_size)
    X_train = pad_sequences(train_X_seq, maxlen = max_len, truncating="pre") # 길이를 맞춰줌
    return X_train, vocab_size, tokenizer

train_y = train['info']
train_X, vocab_size, vectorizer = text2sequence(train['new_article'], max_len = max_len)

print(train_X.shape, train_y.shape)
```

```
vocab_size :  33519
(118676, 100) (118676,)
```

```
# vectorizer 저장
joblib.dump(vectorizer, "vectorizer.sav")
```

```
['vectorizer.sav']
```

Code

모델 생성

```
# EDA기반으로 만든 feature 예측변수로 추가한 모델 생성
from keras import regularizers

def LSTM_add_feature(vocab_size, embedding_size = 100, max_len=100):
  input1 = keras.layers.Input(shape = [max_len,]) #문장 단어 input
  input2 = keras.layers.Input(shape = [feature_num,]) # EDA기반 feature input

  # LSTM
  embedding = keras.layers.Embedding(vocab_size, embedding_size, input_length = max_len)(input1) # 임베딩 가중치 훈련
  dropout1 = keras.layers.SpatialDropout1D(0.5)(embedding)
  lstm1 = keras.layers.LSTM(32, return_sequences = True)(dropout1)
  lstm2 = keras.layers.LSTM(32)(lstm1)
  dropout2 = keras.layers.Dropout(0.5)(lstm2)

  # MLP
  concat = keras.layers.concatenate([dropout2,input2])
  hidden = keras.layers.Dense(32, activation = "selu")(concat)
  output = keras.layers.Dense(1, activation = "sigmoid")(hidden)

  model = keras.Model(inputs = [input1, input2], outputs = [output])

  model.compile(optimizer=keras.optimizers.Adam(lr=learning_rate), loss="binary_crossentropy", metrics = "accuracy")
  model.summary()
  return model
```

Code

train test split

```
#concat
feature_num = 4
train_X = np.concatenate([train_X,train[["info1_title","info1_content","new_ord","content_startswith_["]].values.reshape(-1,feature_
num)], axis=1)

# 문장별로 train_test set 분리
from sklearn.model_selection import train_test_split

X_train, X_valid, y_train, y_valid = train_test_split(train_X, train_y, random_state = 42, test_size = 0.3)
```

모델 훈련 및 검증

```
# 훈련 시
tf.random.set_seed(42)
embedding_size = 300

checkpoint_cb = keras.callbacks.ModelCheckpoint("hyerim_add_feature_best_model2.h5",
                                                save_best_only = True)

# 하이퍼파라미터
max_epoch = 3
batch_size = 32
learning_rate = 0.001

model = LSTM_add_feature(vocab_size, max_len = max_len, embedding_size = embedding_size)
history = model.fit(x=[X_train[:,:max_len],X_train[:,-feature_num:]], y=y_train, epochs=max_epoch,
                    batch_size = batch_size,  validation_data = ((X_valid[:,:max_len],X_valid[:,-feature_num:]),y_valid), validation_bat
ch_size = batch_size,
                     callbacks = [checkpoint_cb])
```

Code

전문장, 전전문장의 예측결과값을 이용하여 예측값 보정

```
lstm_model = keras.models.load_model("lstm_model.h5")
```

```
# LSTM을 통한 예측
train_predicted = lstm_model.predict((train_X[:,:max_len],train_X[:,-feature_num:]))

# 각 문장의 바로 앞문장과, 그 앞 문장의 예측값 생성
train_pre_predicted1 = np.array([train_predicted[idx-1][0] for idx in range(len(train_predicted))]).reshape(-1,1) # 앞문장
train_pre_predicted2 = np.array([train_predicted[idx-2][0] for idx in range(len(train_predicted))]).reshape(-1,1) # 앞앞문장

# 학습을 위한 데이터셋 생성
train["predicted"]=train_predicted
train["pre_predicted1"]=train_pre_predicted1
train["pre_predicted2"]=train_pre_predicted2

# ord가 1,2 인 것은 학습에서 제외 후 최종 lgbm 훈련을 위한 데이터셋 생성
final_X = train[["predicted","pre_predicted1","pre_predicted2"]][(train["ord"]!=1)&(train["ord"]!=2)]
final_y = train["info"][(train["ord"]!=1)&(train["ord"]!=2)]
final_X_train, final_X_valid, final_y_train, final_y_valid = train_test_split(final_X, final_y,test_size = 0.3, random_state = 42)
```

```
# LGBM 모델 불러오기
from lightgbm import LGBMClassifier

# index 랜덤으로 섞음
np.random.seed(42)
random_index = np.random.randint(0,len(final_X),len(final_X))

# LGBM model 예측
lgbm_model=LGBMClassifier(random_state=42)
lgbm_model.fit(final_X.iloc[random_index], y=final_y.iloc[random_index])
```

Code

test 예측

```
path = "/gdrive/My Drive/dacon_news/"
feature_num = 4
test = pd.read_csv(path+"data/news_test.csv")

# EDA 기반 feature 추가
test["info1_title"]=test["title"].apply(title_choose)
test["info1_content"]=test["content"].apply(content_choose)
test["content_startswith_["]=test.content.apply(lambda x : str(x).startswith("[") or str(x).startswith("(") or str(x).startswith("제목"))+0
title_group = (test.groupby(["title"]).count())["n_id"]
test["new_ord"]=test.apply(lambda x: x["ord"]/title_group[x["title"]], axis=1)

# 전처리
test['new_article'], okt = text_preprocessing(test['content'])
test_X_seq = vectorizer.texts_to_sequences(test["new_article"])
test_X = pad_sequences(test_X_seq, maxlen = max_len) # 길이를 맞춰줌

# concat
test_X = np.concatenate([test_X,test[["info1_title","info1_content","new_ord","content_startswith_["]].values.reshape(-1,feature_num)], axis=1)
```

```
# lstm 모델을 통한 예측
predicted = lstm_model.predict([test_X[:,:max_len],test_X[:,-feature_num:]])
```

```
# 앞문장, 앞앞문장으로 최종 예측
pre_predicted1 = np.array([predicted[idx-1][0] for idx in range(len(predicted))]).reshape(-1,1) # 앞문장
pre_predicted2 = np.array([predicted[idx-2][0] for idx in range(len(predicted))]).reshape(-1,1) # 앞앞문장

test["predicted"]=predicted
test["pre_predicted1"]=pre_predicted1
test["pre_predicted2"]=pre_predicted2

lgbm_final_predicted = lgbm_model.predict(test[["predicted","pre_predicted1","pre_predicted2"]])
```

```
# ord가 1,2인 것은 lstm모델만을 통해 예측, threshold 0.6으로 분류
test["info"]=lgbm_final_predicted
test["info"][test["ord"]==1]=(test["predicted"][test["ord"]==1]>=0.6)+0
test["info"][test["ord"]==2]=(test["predicted"][test["ord"]==2]>=0.6)+0
test["info"][test["content"].apply(lambda x: True if ('http://etoday.bujane.co.kr/' in x) or ('http://bit.ly/2XrAuGJ_itoozanews' in x) or ('htt
p://www.hisl.co.kr/0306/' in x) or ('https://www.hankyung.com/election2020/' in x) or (x==']]') else False)]=1
```