

2024.01

MUSINSA

회귀 모델을 이용한 판매량 예측

제로베이스 데이터 취업 스쿨 20기

발표 - 하승현 / 팀원 - 이청하, 이병찬, 김지현, 김주희

Contents

01

프로젝트 소개

- 패션 e커머스 현황
- 프로젝트 주제

02

데이터 수집 및 전처리

- 데이터 정의
- 웹 크롤링
- 리뷰감성분석 파생변수 생성
- 데이터 전처리

03

EDA

04

모델 학습과 평가

- OLS
- Multiple Regression + Lasso
- Lasso
- LGBM
- XGBoost
- 모델 평가 결과

05

결론

- 새로운 제품에 대한 예측 (기대효과)
- 결론 및 결과
- lesson & learned



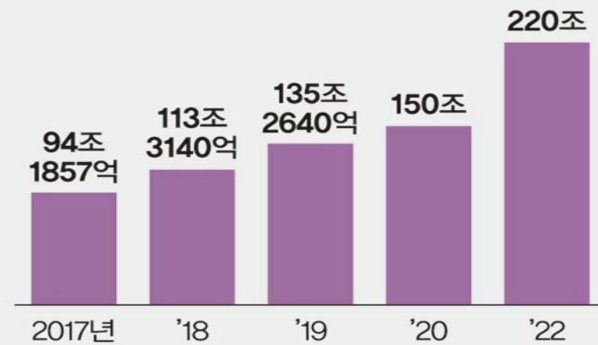
1. 프로젝트 소개



한국 e커머스 내 무신사의 위치

1. 국내 e커머스 시장은 5년 이상 꾸준히 성장세
2. 그 중 패션 분야에서 트렌드로 자리잡은 무신사

국내 온라인 쇼핑 시장 규모 (단위 : 원)



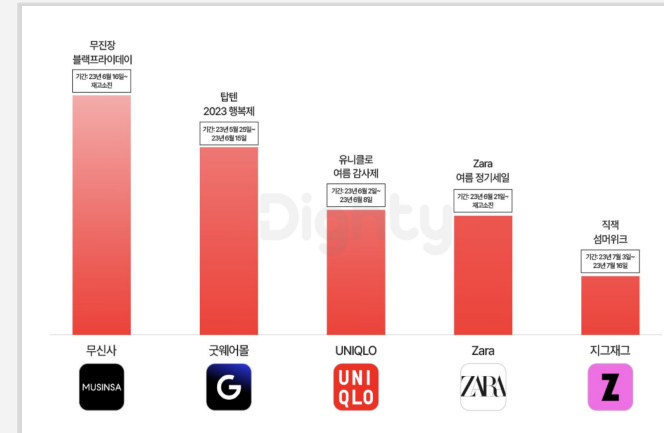
성장하는 시장규모

1. 220조 규모의 온라인 쇼핑시장
2. 팬데믹 상황에서도 꾸준한 성장

2022년		2023년	
MUSINSIA	무신사 42.4	MUSINSIA	무신사 48.5
ZIGZAG	지그재그 20.6	ABLY	에이블리 22.2
ABLY	에이블리 18.2	ZIGZAG	지그재그 21.5
BRAND	브랜드 11.7	CRIM	크림 11.2
TAMTAM	탐텐물 10.7	29CM	29CM 11.1
29CM	29CM 8.4	SSF SHOP	SSF SHOP 8.7
SSF SHOP	SSF SHOP 7.6	WCONCEPT	W컨셉 7.3
WCONCEPT	W컨셉 6.5	BRAND	브랜드 7.0
LF	LF물 6.1	LF	LF물 5.7
CRIM	크림 5.0	H	하이버 5.1

3개월 이내 사용경험

1. 소비자 대상 3개월 설문
2. 높아진 무신사의 비율



실제 설치경험

1. 사용자 대상 실제 설치 경험
2. 가장 높은 비율의 무신사



“
무신사의 제품 정보를 크롤링하여,
제품의 1년 누적 판매량을
예측해 보자
”

1

2

3

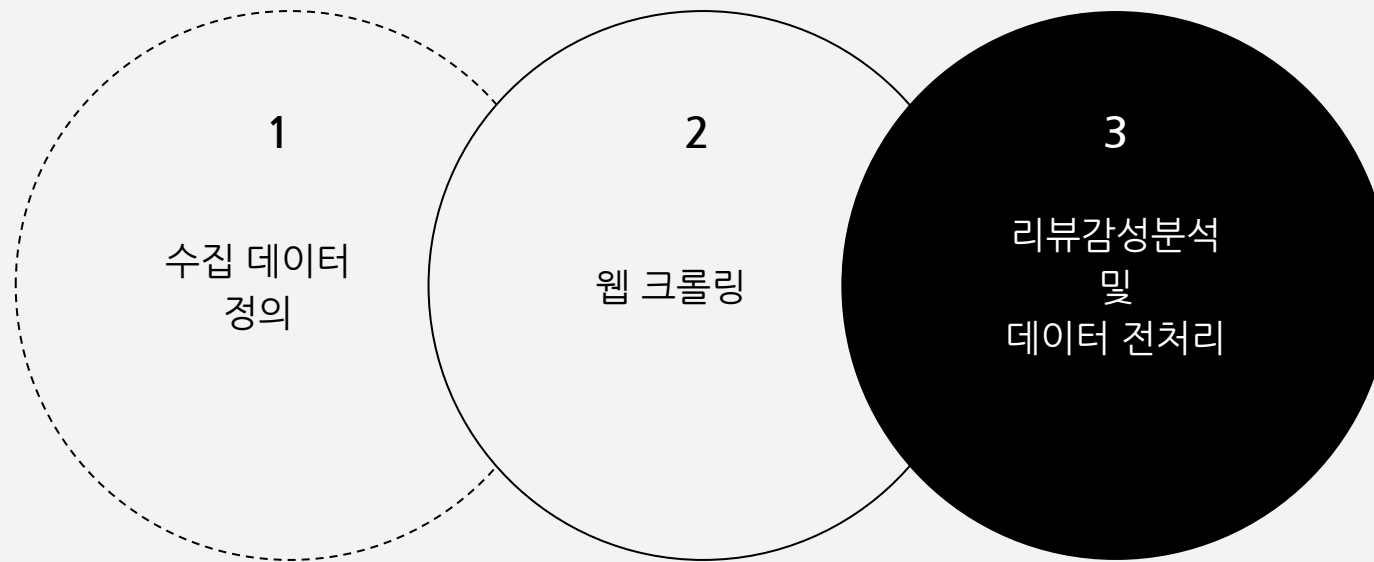
4

5

2. 데이터 수집 및 전처리



Steps



1

2

3

4

5


1. 수집 데이터 정의

1. 1) 모델링에 사용할 피처가 될 제품 데이터 종류 정의

바지 > 데님 팬츠 (토피)

한정 판매

헤이즈 워시드 와이드 데님 팬츠 (VINTAGE INDIGO) | Haze washed wide denim pants (VINTAGE INDIGO)



비슷한 스타일 >

한정 판매

무신사 스토어를 포함한 제한된 판매처에서만 구매 가능한 상품입니다.

Product Info

제품정보

브랜드 / 품번

TOFFEE / T3F-HWWDP204VI

시즌

2023 F/W 남

조회수(1개월)

13.5만 회 이상

누적판매(1년)

1.1만 개 이상 (결제완료-반품)

좋아요

25,640

구매 후기

★★★★★ 4.9 후기 1,847개 보기

#위싱진

#와이드팬츠

#와이드데님

#도피바지

#도피와이드

#데님팬츠

#장바지

#데님팬츠맨

#데님와이드핏

#데님

19~28세, 남성

에게 인기 많은 상품

Delivery Info

배송정보

출고 정보

결제 3일 이내 출고

배송 정보

국내 배송 / 입점사 배송 / CJ대한통운

1/8(월) 도착 예정

Price Info

가격정보

무신사 판매가

69,000원

무신사 회원가

44,850원

35% 이상 할인

무신사 적립금

최대 3,794원

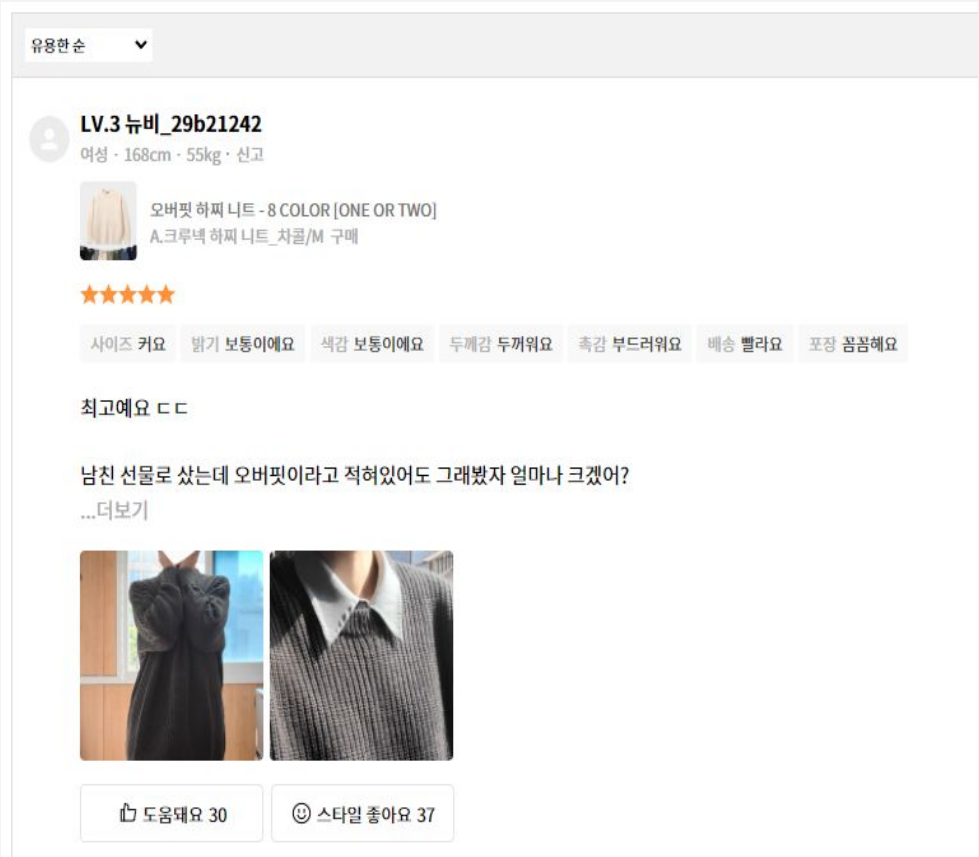
전 상품 무료배송 (가구 등 일부 상품 제외)

회원 특별 혜택

Features	Type	Example
상위 카테고리	String	바지
하위 카테고리	String	데님 팬츠(토피)
아이템 품번	String	T3F-HWWDP204VI
아이템 이름	String	헤이즈 워시드 와이드...
한정판매(숨겨두기)	String	T/F
단독판매(숨겨두기)	String	T/F
타겟 성별	String	남성
구매 성별	String	남성
구매 나이	INT	19~28세
1개월 조회수	INT	13.5만 회
좋아요 수	INT	25,640
가격(회원이 기준)	INT	44,850
할인률(회원이 기준 상시)	INT	35
배송정보	INT	3(일)
평점	Float	4.9
후기 수	INT	1,847
1년 누적 판매량	INT	1.1만개 이상

1. 수집 데이터 정의

1. 2) 리뷰감성분석점수 파생변수 생성을 위한 데이터 종류 정의



Features	Type	Example
아이템 품번	String	T3F-HWWDPT204VI
아이템 이름	String	헤이즈 워시드 와이드...
유저 ID	String	뉴비_29b21242
유저 평점	Float	5.0
유저 리뷰	String	최고예요 ㄷㄷ...



리뷰감성분석점수(score) 파생변수 생성

1
2
3
4
5

2. 웹크롤링

1. 상품 데이터 크롤링

- 대분류가 상의, 바지, 아우터인 카테고리의 데이터 크롤링 진행
- 무신사 추천순으로 상의, 바지, 아우터 카테고리별로 총 8000여개 수집

major_category	middle_category	name		number	limit	exclusive	target_gender	buy_gender	buy_age	view	like	price	discount_rate	delivery_date	rating	review	buy
0	상의	셔츠/블라우스	오버사이즈 16골 에센셜 코 듀로이 셔츠 블랙	22FWGS28BK	1	0	2	0.0	('29~33세', 0)	1200.0	493	39800.0	20	19.0	4.6	36	50.0
1	상의	니트/스웨터	TD5-SW01 램스울 라운드 니 트-만다린오렌지	TD5-SW01_MAO	1	0	0	0.0	('29~33세', 0)	4900.0	650	53550.0	10	5.0	4.9	61	200.0
2	상의	후드 티셔츠	[SET UP] PPP 플라워 후드 셋 업_멜란지 그레이	KBCS1TH003MGKBCS1PL003MGKBCS1PS003MG	1	0	2	1.0	('19~23세', 0)	3600.0	2536	49900.0	49	5.0	4.8	41	150.0
3	상의	후드 티셔츠	TIE DYE 후드티 Olive Green	CTTZPHD01UG4	0	0	0	0.0	('40세~', 0)	1000.0	569	19800.0	80	5.0	4.8	62	150.0
4	상의	긴소매 티셔츠	essential turtleneck logo top - pink	BT23WTS002PIKF	1	0	1	1.0	('24~28세', 0)	11000.0	2561	43000.0	0	28.0	4.8	34	100.0
...
8043	아우터	숏패딩/숏헤비 아우터	숏 미니멀 덕다운 패딩 블랙	3001	0	1	0	0.0	('29~33세', 0)	3300.0	2028	238400.0	20	7.0	4.8	87	100.0
8044	아우터	아노락 재킷	Pocket Tidy Hood Anorak H7 Navy/Green	409	1	0	2	1.0	('34~39세', 0)	400.0	368	53000.0	40	5.0	5.0	22	50.0
8045	스포츠/용품	아우터	아웃런 더 스톱 재킷 1376794-002	1376794-002	0	0	0	0.0	('24~28세', 0)	1100.0	228	129000.0	0	5.0	5.0	37	50.0
8046	아우터	레더/라이더스 재킷	비건 레더 싱글 카 코트 (블 랙)	CXC2JK06K-BLK	0	0	2	NaN	('29~33세', 0)	1000.0	500	175500.0	10	5.0	4.8	20	NaN
8047	아우터	패딩 베스트	컴포트 베스트_BROWN	BR	0	0	0	NaN	('24~28세', 0)	1700.0	154	193000.0	0	5.0	4.9	25	NaN

8048 rows × 17 columns

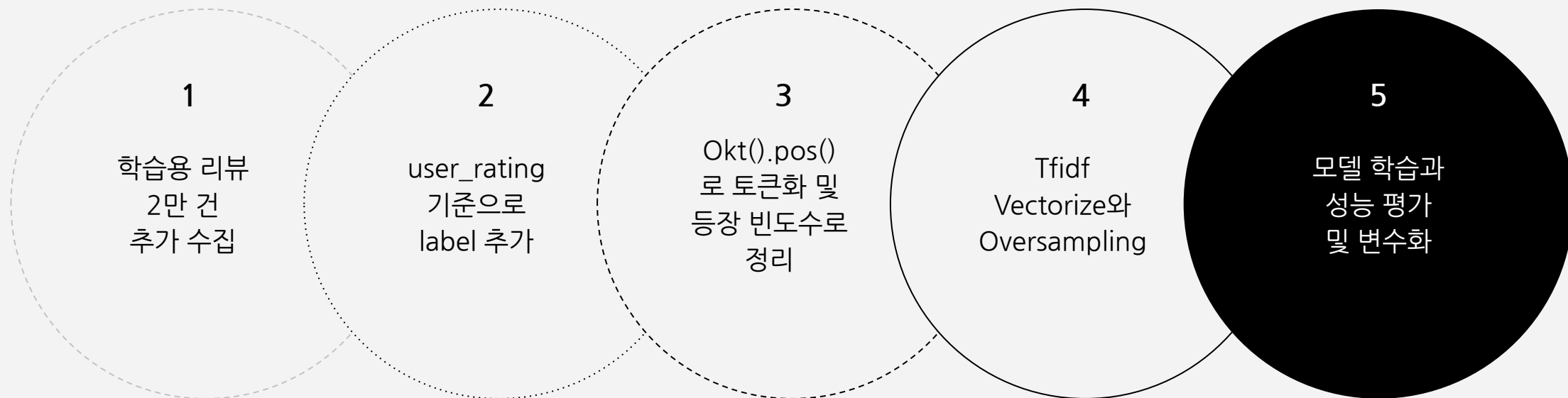
2. 웹크롤링

2. 리뷰 데이터 크롤링

- 수집한 각 상품별 리뷰 데이터 데이터 크롤링 진행
- 각 상품 별 스타일/이미지/일반 각각 최대 15개의 리뷰 수집
- 총 180,000여개 수집

	product_names	product_ids	user_names	user_ratings	user_reviews
0	오버사이즈 16골 에센셜 코듀로이 셔츠 블랙	22FWGS28BK	김오복	80	깔끔하고 입기 편안해요 손이 자주가는 편한 느낌
1	오버사이즈 16골 에센셜 코듀로이 셔츠 블랙	22FWGS28BK	web	100	오버한 사이즈고 입으면 더 예뻐니다. 부드럽고 적당한 두께감이라 겨울에 아주 잘입을...
2	오버사이즈 16골 에센셜 코듀로이 셔츠 블랙	22FWGS28BK	퓨블릭a.	100	많이 길고 오버할 줄 알았는데 그렇지 않아서 좋구여 예뻐니다. 다른 색깔도 살듯..
3	오버사이즈 16골 에센셜 코듀로이 셔츠 블랙	22FWGS28BK	YANGMAL2	80	굿굿 생각한거보다 이쁘네영 자주 손이가네요 추천합니다
4	오버사이즈 16골 에센셜 코듀로이 셔츠 블랙	22FWGS28BK	.-----.	100	살짝 얇은 원단이 아쉽지만 전체적으로 가격을 뛰어 넘는 품질입니다.
...
179870	UNISEX Vegan Leather Goose-Down Puffer Blouson...	MS22FWUJK01BK	뉴비_a5e777a1e703	100	굳~!!!아주아주이쁘고좋아여~~~이브랜드 옷을참잘만드네여
179871	UNISEX Vegan Leather Goose-Down Puffer Blouson...	MS22FWUJK01BK	GoodLuck_DK	100	블프 할인으로 좋은 가격에 샀습니다. 생각 이상으로 따뜻하고 예뻐요!!
179872	UNISEX Vegan Leather Goose-Down Puffer Blouson...	MS22FWUJK01BK	배넷	100	오늘따라 좀 색.시하고 싶다 싶을때 입으면 좋을것같아요
179873	UNISEX Vegan Leather Goose-Down Puffer Blouson...	MS22FWUJK01BK	3335	100	안에 두꺼운 옷은 못 입고 티 두개 레이어드나 두겹지 않은 니트정도만 입을 수 있는...
179874	UNISEX Vegan Leather Goose-Down Puffer Blouson...	MS22FWUJK01BK	개짖는소리좀안나게하라	100	이전부터 눈여겨봤던 패딩인데블프 +랜덤쿠폰으려 거의18만원에 구매했구요 디테일 핏 ...

3. 리뷰감성분석 파생변수 생성



1. 무신사에서 학습용 리뷰 데이터 약 2만 건을 추가 수집

- pos, neu, neg 3가지 클래스를 골고루 수집하기 위해 '평점 높은 순', '평점 낮은 순' 필터 이용
- 중복 제거

2. 지도 학습 진행을 위해 user_rating 기준으로 label 부여

- 80점 이상이면 'pos', 60점이면 'neu', else 'neg'

1
2
3
4
5

3. 리뷰감성분석 파생변수 생성

3. 1) 한국어 분석을 위해 Okt().pos()를 이용해 형태소별 토큰화
- 한국어의 경우 동음이의어 구별을 위해 품사를 함께 사용하면 분석 정확도가 향상됨
 - stopwords를 지정하여 사용

	product_names	product_ids	user_names	user_ratings	user_reviews	label	token
0	[기모버전추가]시그니처 오버핏 맨투맨(4col)	P00000NE	뉴비_cc4a349fa2d6	100	색상무난하고 기본스타일의 맨투맨입니다편해요	pos	[색상/Noun, 무난/Noun, 기/Modifier, 본/Modifier, 스타일...
1	[기모버전추가]시그니처 오버핏 맨투맨(4col)	P00000NE	은비파파	100	기모인줄 알고 구입했는데 기모가 아니네요. 그럼에도 두께감도있고 기모가 아니라 부해...	pos	[기모/Noun, 인/Josa, 줄/Noun, 알/Noun, 고/Josa, 구입/N...
2	[기모버전추가]시그니처 오버핏 맨투맨(4col)	P00000NE	거누임니닷	100	옷 색감이 딱 생각한거라 너무 좋아요 추천합니다!	pos	[옷/Noun, 색감/Noun, 딱/Adverb, 생각/Noun, 한/Determi...

```
stopwords =  
['의/Josa', '가/Josa', '이/Josa', '은/Josa', '들/Josa', '는/Josa', '좀/Noun', '강/Noun', '과/Josa', '도/Josa', '를/Josa', '으로/Josa', '에/Josa', '와/Josa', '한/Noun', '하다/Verb', '을/Josa', '에서/Josa', '에게/Josa', '하고/Josa', '이다/Verb', ' ./Punctuation']
```

1
2
3
4
5

3. 리뷰감성분석 파생변수 생성

3. 2) 등장 빈도수가 낮은 토큰 정리

- threshold = 3
- 등장 빈도가 threshold 미만인 희귀 단어 조사
- 희귀 단어의 수는 많지만, 훈련 데이터에서 희귀 단어 등장 빈도 비율은 매우 적은 수치인 2.35%
- 등장 빈도가 3회 미만 단어들은 제거

```
ko.vocab()
```

```
-----  
FreqDist({'좋다/Adjective': 10164, '입다/Verb': 8431, '같다/Adjective': 5719, '너무/Adverb': 5146, '있다/Adjective': 3870, '이쁘다/Adjective': 3762, '핏/Noun': 3597, '사이즈/Noun': 3416, '예쁘다/Adjective': 3329, '이다/Adjective': 3305, ...})
```

단어 집합(vocabulary)의 크기 : 11359

등장 빈도가 2번 이하인 희귀 단어의 수: 6560

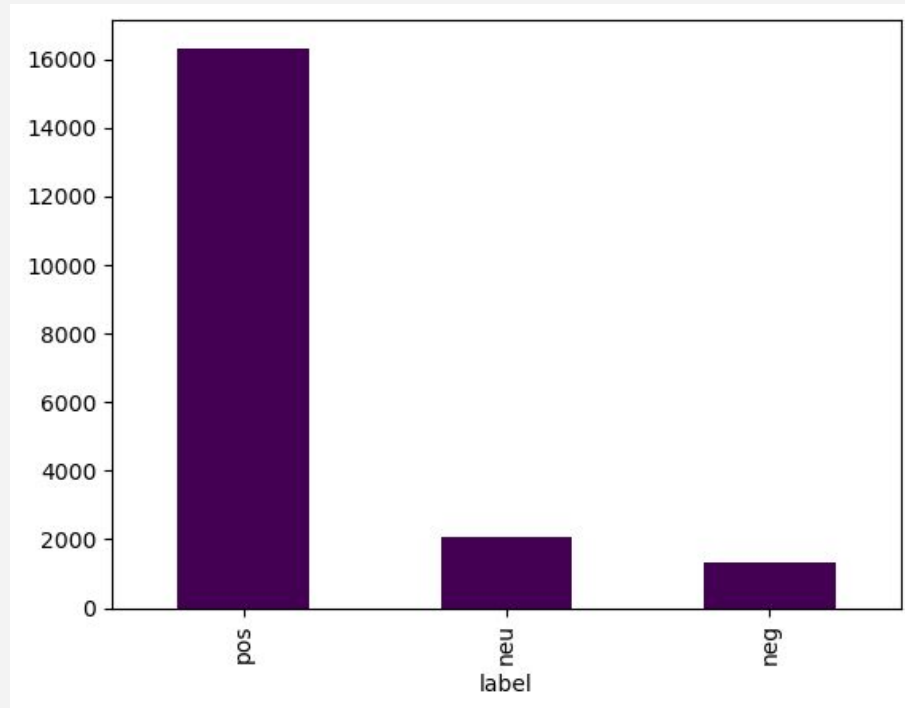
단어 집합에서 희귀 단어의 비율: 57.75156263755612

전체 등장 빈도에서 희귀 단어 등장 빈도 비율: 2.352789995385541

3. 리뷰감성분석 파생변수 생성

4. TfidfVectorizer와 Oversampling

- 토큰을 하나의 text로 합치고 TfidfVectorizer 진행
- 클래스 간 비율 차이가 큰 것을 보완하기 위해 학습데이터에 SMOTE Oversampling 진행



X_train_vectorized.shape	(15770, 3831)
y_train.shape	(15770,)
X_train_vectorized_over.shape	(39231, 3831)
y_train_over.shape	(39231,)

```
print(np.unique(y_train_over, return_counts=True))
```

```
(array(['neg', 'neu', 'pos'], dtype=object), array([13077, 13077, 13077], dtype=int64))
```

3. 리뷰감성분석 파생변수 생성

5. 1) 모델 학습과 성능 평가

- MultinomialNB, LightGBMClassifier, LogisticRegression 학습 후 성능 평가
- 모델 별로 Precision과 Recall에서 큰 차이가 없기 때문에 Accuracy가 가장 높은 LightGBMClassifier 모델 채택

	model_names	Accuracy_score	Precision	Recall	F1
0	MultinomialNB	0.7208	0.5276	0.6087	0.5472
1	LightGBMClassifier	0.8428	0.6179	0.5244	0.5540
2	LogisticRegression	0.7446	0.5303	0.6095	0.5552

5. 2) 변수화

- 실제 제품 리뷰를 위 LightGBMClassifier를 이용해 예측값 도출
- 'pos'는 1, 'neu'는 0.5, 'neg'는 0으로 치환하여 제품별 리뷰스코어(score) 파생변수 생성 및 추가

4. 데이터 전처리

1. feature별 NaN값 처리

	Feature	NaN값 처리	처리 근거
0	buy	<ul style="list-style-type: none"> - 0 - drop() 	<ul style="list-style-type: none"> • 평점이 없고 리뷰수가 '0' 이어도, 조회수가 200 이하인 상품: 정말 판매가 이루어지지 않은 제품이기 때문에 buy가 없는 것으로 판단하여 0 처리 • 그 외 판매량의 NaN값은 삭제(drop)
1	buy_gender	<ul style="list-style-type: none"> - target_gender 값 사용 	<ul style="list-style-type: none"> • buy_gender 칼럼은 구매 성별, target_gender 칼럼은 상품 타겟 성별을 의미 • 상품 생산 시 타겟으로 한 성별이 구매에 가장 큰 영향을 줄 것이라고 봄
2	delivery_date	<ul style="list-style-type: none"> - 5(median) 	<ul style="list-style-type: none"> • 해당 컬럼의 통계량 확인 • max값이 60, mean 5.6, median 5.0으로 이상치를 제거하면 평균이 감소할 것으로 예상 • 중앙값인 5로 처리
3	rating	<ul style="list-style-type: none"> - 0 - 4.8(median) 	<ul style="list-style-type: none"> • buy 값이 '0' 이 아니면서 rating이 '0' 인 값: 중앙값 4.8점 처리 • buy 값이 '0' 이 면서 rating이 '0' 인 값: 0 처리
4	view	<ul style="list-style-type: none"> - RandomForestRegressor 적용 	<ul style="list-style-type: none"> • RandomForestRegressor 모델을 이용한 view 예측값 사용 • 여러가지 파라미터 조정 결과 <ul style="list-style-type: none"> - Best r2_Score: 0.47624328080184863 - Best trial parameter {'n_estimators': 281, 'max_depth': 20, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_features': 0.8499303073699949}



```
df.isnull().sum()

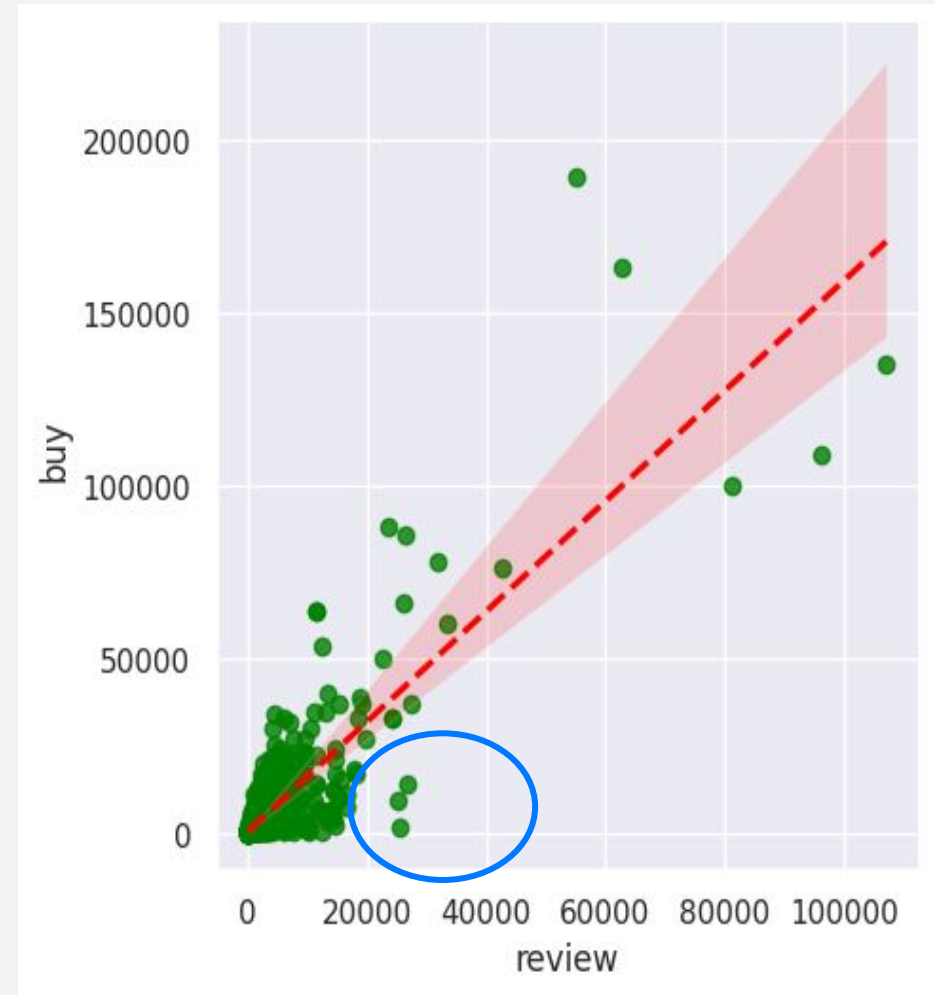
major_category    0
middle_category   0
name              0
number            0
limit             0
exclusive          0
target_gender      0
buy_gender         0
buy_age           0
view              0
like              0
price             0
discount_rate      0
delivery_date      0
rating            0
review            0
buy               0
score             0
dtype: int64
```

1
2
3
4
5

4. 데이터 전처리

2. 논리적 이상치 제거

- 리뷰수(review)가 판매량(buy)보다 많은 경우
웹에서 표기가 잘못된 데이터는 고쳐주되
그렇지 않은 데이터들에 한에서 제거를 진행하였음



1

2

3

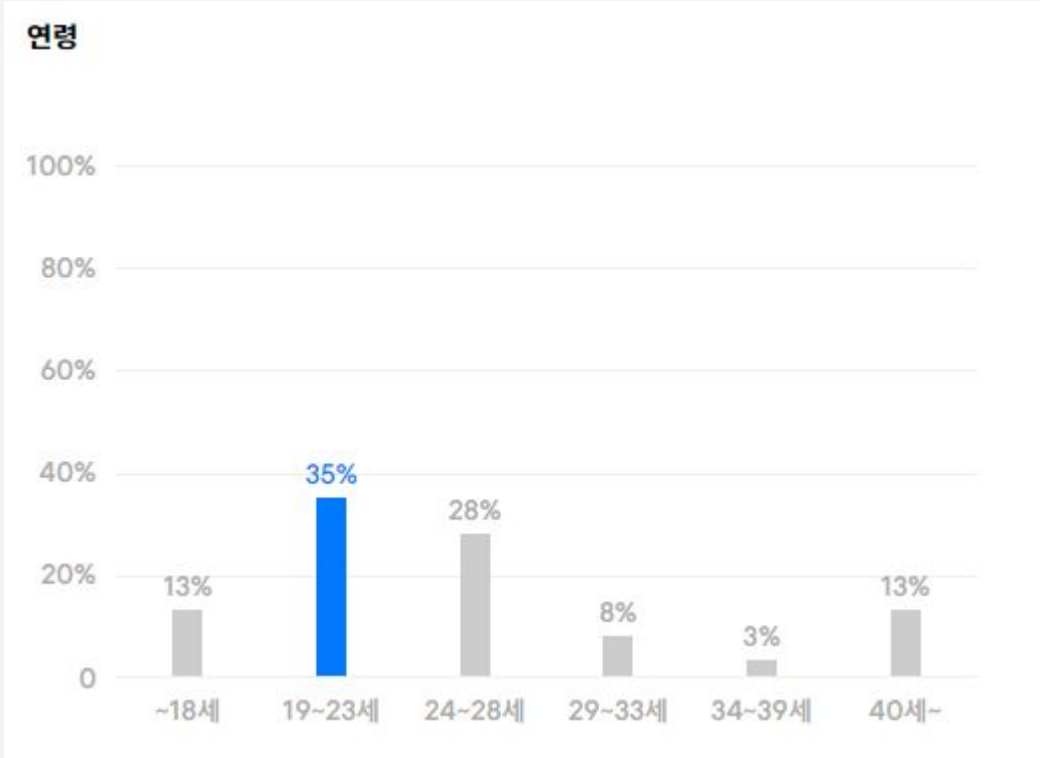
4

5

4. 데이터 전처리

3. buy_age 범주형 데이터 매핑

- 해당 상품을 가장 많이 구매한 연령대를 나타내는 buy_age의 값
- 동물이 아닐 경우 ('나이구간', 0)이고 동물일 경우 ('나이구간', '나이구간')으로 수집
- buy_age의 경우 동물인 경우가 있기 때문에 buy_age1, buy_age2로 컬럼을 나눔
- 범위를 0(NaN),1,2,3,4,5,6으로 구분지어주었음

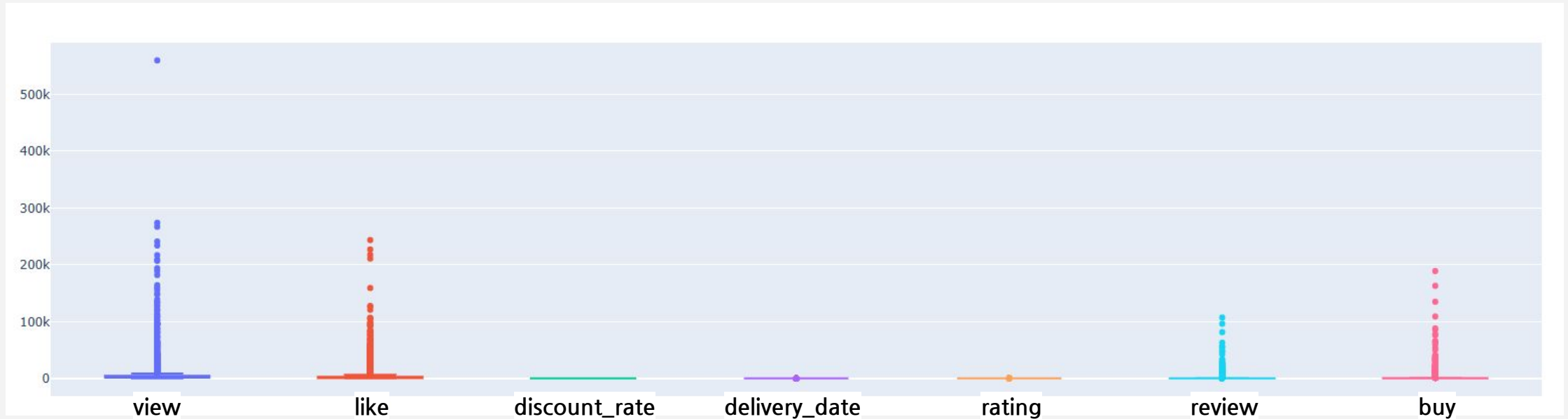


	buy_age	buy_age1	buy_age2
0	(' 19~23세 ' , 0)	2	0
1	(' 19~23세 ' , ' 24~28세 ')	2	3
2	(' 29~33세 ' , 0)	3	0

3. EDA



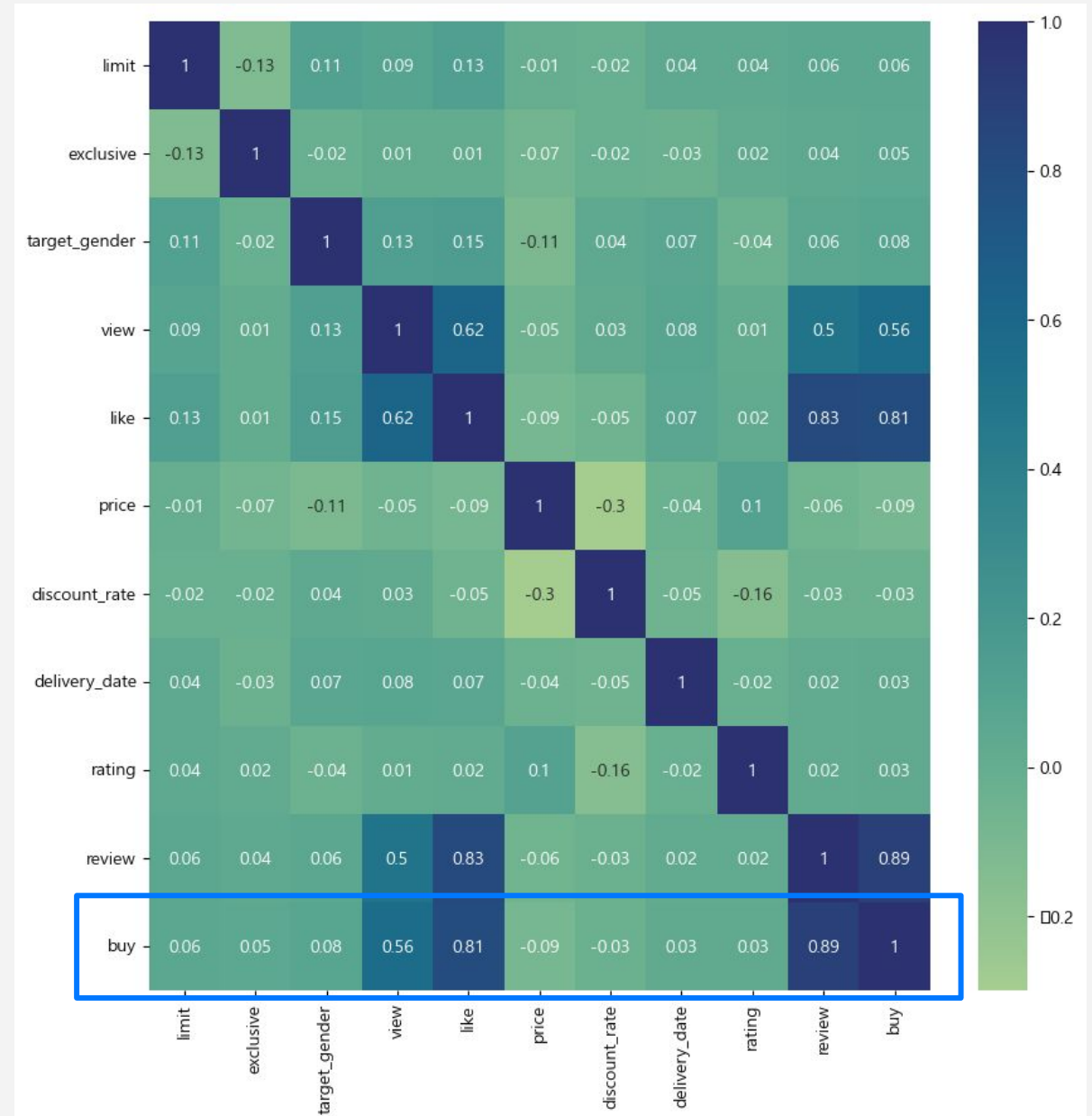
1. 데이터 분포



- feature별 값의 단위 차이가 크다.
- 모델링을 위해 스케일러 또는 로그 변환 등의 처리가 필요할 수 있다.
- view 있는 이상치를 확인할 수 있으며, 사분위수들을 초과하는 값이 많이 보이기 때문에 IQR을 이용한 이상치 정리가 필요할 수 있다.

2. 상관관계

- target 데이터인 누적 판매량(buy)과 상관도가 높은 feature:
 - 조회수(view)
 - 좋아요(like)
 - 후기수(review)



1

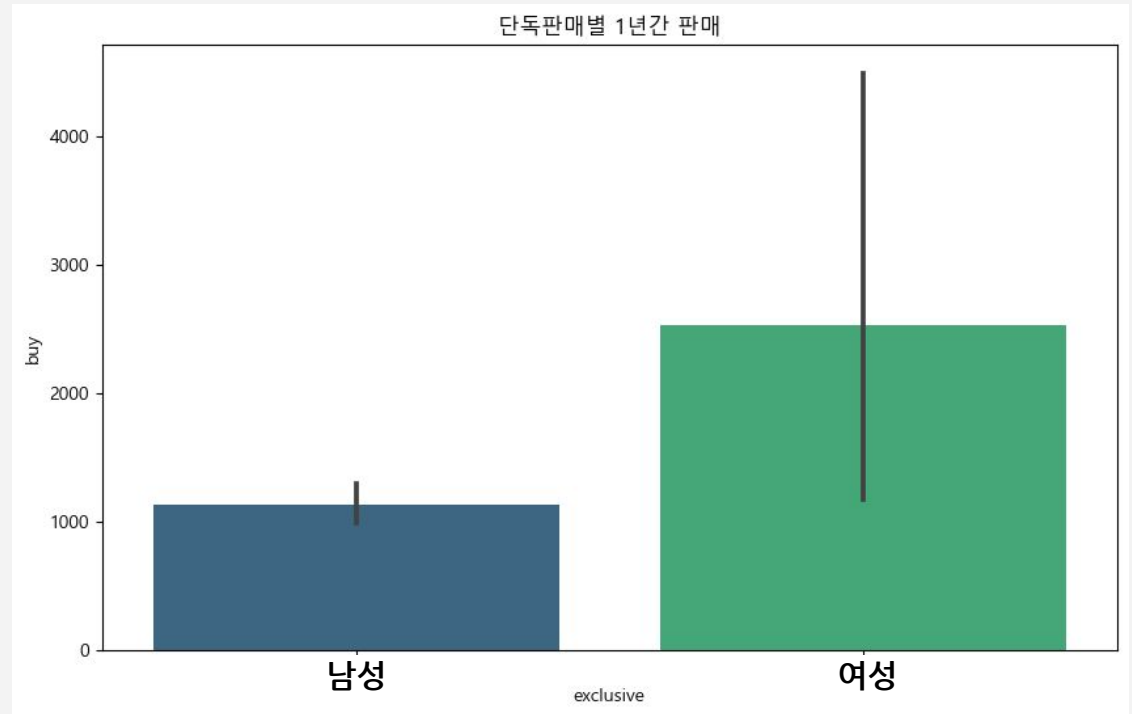
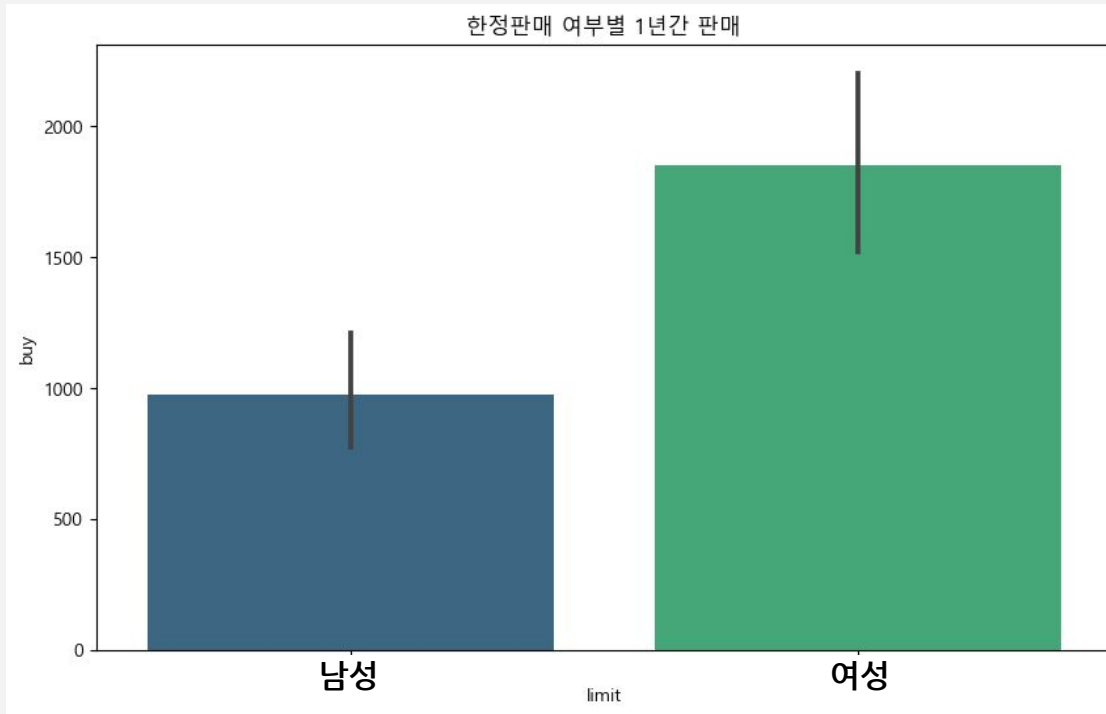
2

3

4

5

3. feature 탐색

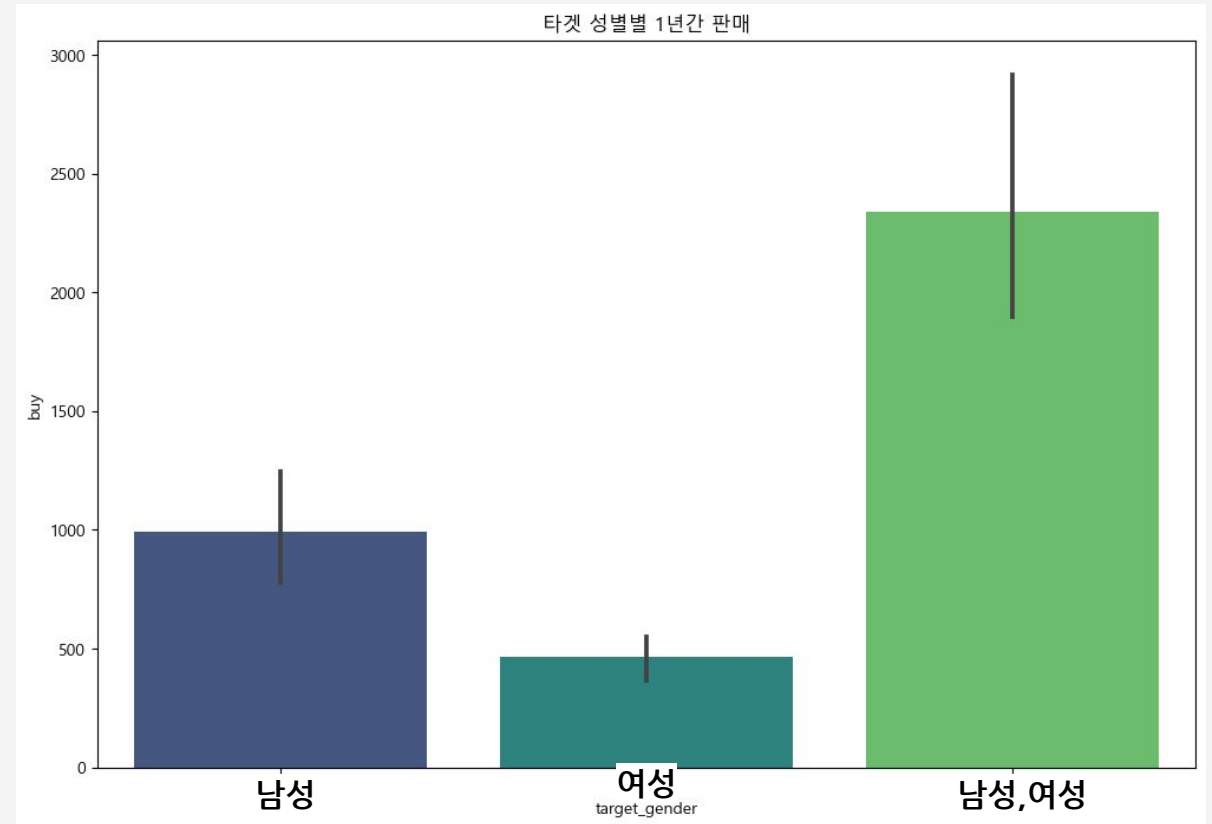


- 한정판매 또는 단독판매인 상품들이 아닌 상품들보다 누적판매량의 평균이 약 2배 더 높다.

3. feature 탐색

- 타겟 성별: 상품 생산 시 타겟으로 한 성별 (예: 스커트, 여자)
- 누적 판매량 평균이 가장 높은 상품: 남녀 공용 타겟
- 누적 판매량 평균이 가장 낮은 상품: 여성 타겟

💡 단, 수집한 데이터 카테고리가 상의, 아우터, 바지로 여성 전용 제품들은 포함되어 있지 않기 때문에 모든 카테고리로 확대 적용할 수 없다.



1

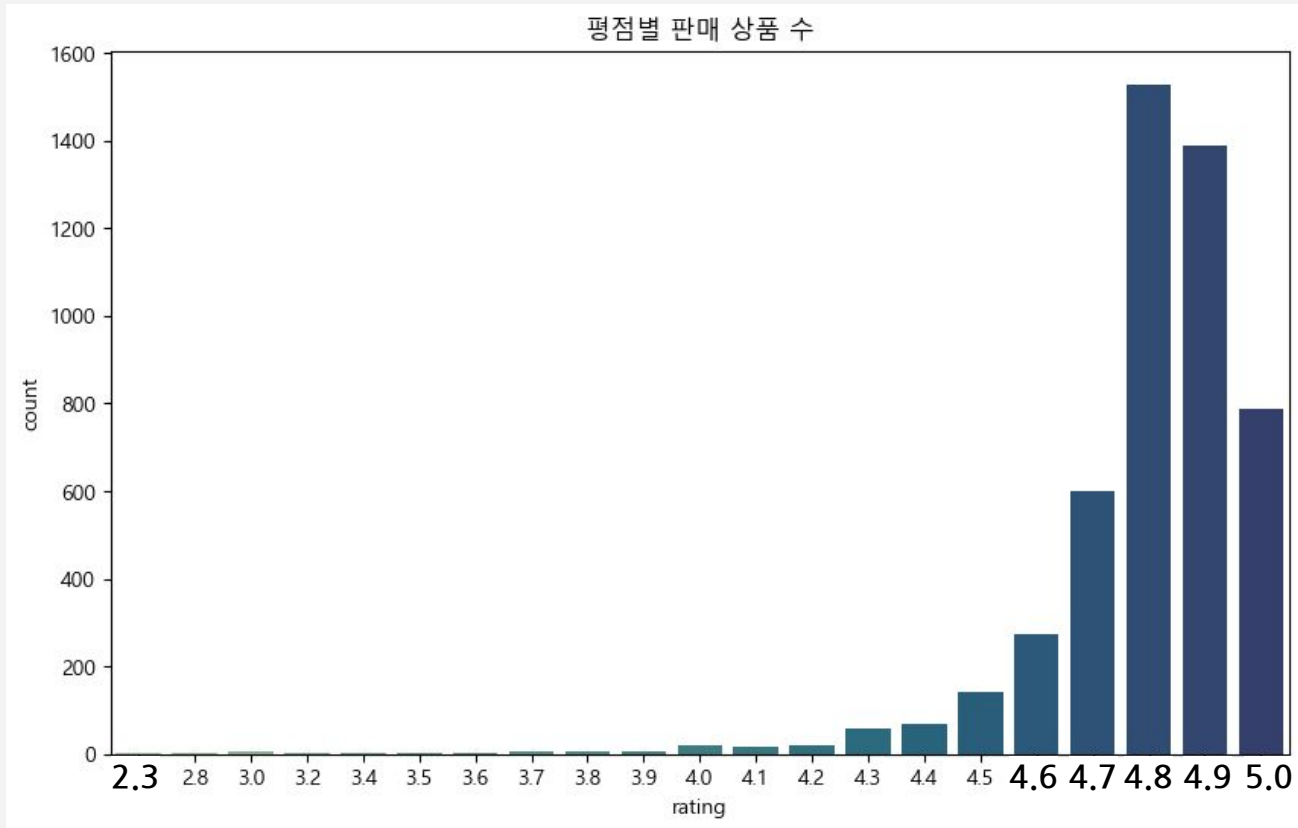
2

3

4

5

3. feature 탐색



- 상품들은 2.3 ~ 5.0까지의 평점을 받았다.
- 4.7 이상 좋은 평점을 받은 상품들이 많은 것으로 보아 무신사 고객들은 구매한 상품에 대한 만족도가 높다.

1

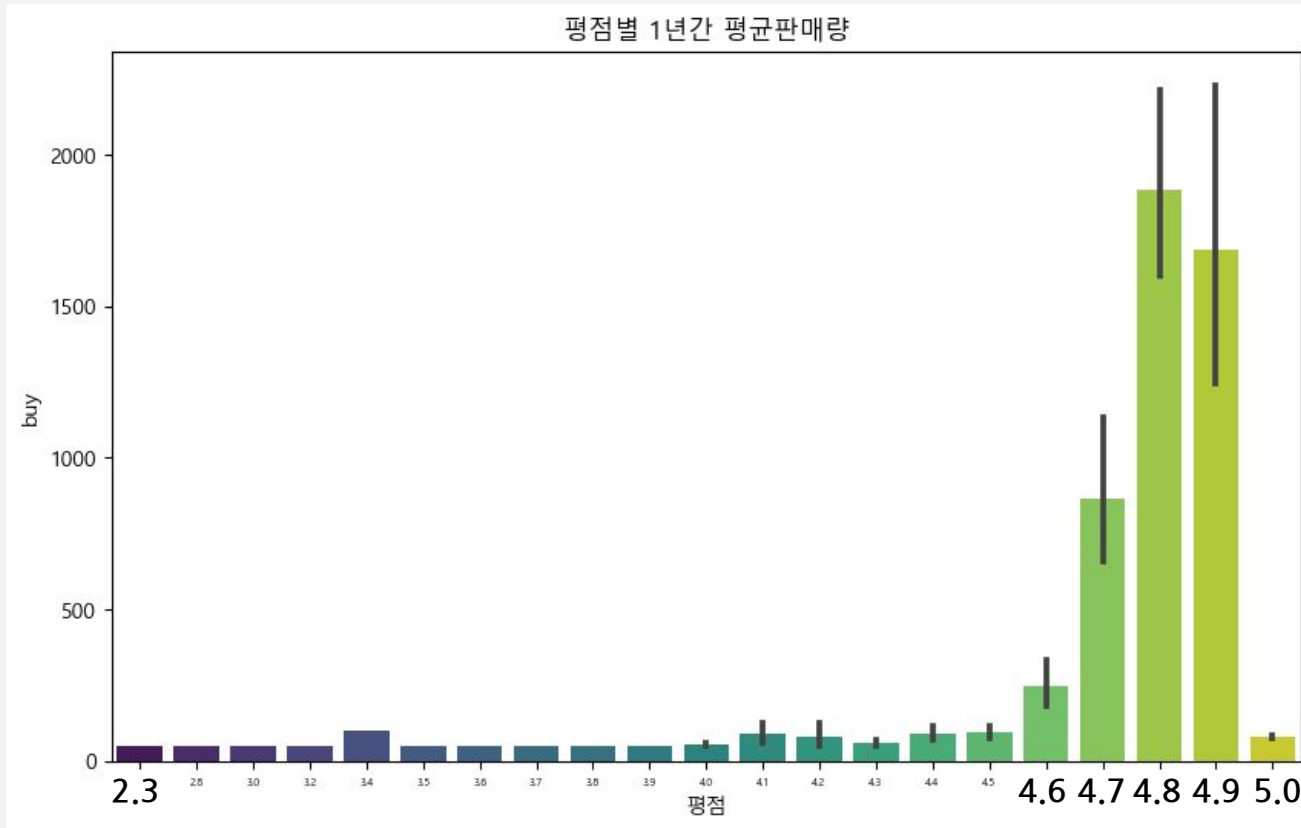
2

3

4

5

3. feature 탐색



- 판매량과 평점은 상관관계 0.03으로 높지 않음
- 판매량이 적더라도 평점이 높을 수 있기 때문
- 하지만 평점이 높은 제품에서는 판매량이 높게 나타남.

💡 평점 5점인 상품의 수는 많았지만 평균 판매량은 낮은 것으로 보아, 아직 다양한 고객 의견이 반영되지 않은 것이며, 판매량도 높으면서 5점인 상품은 존재하기 힘들다는 것을 알 수 있다.

1

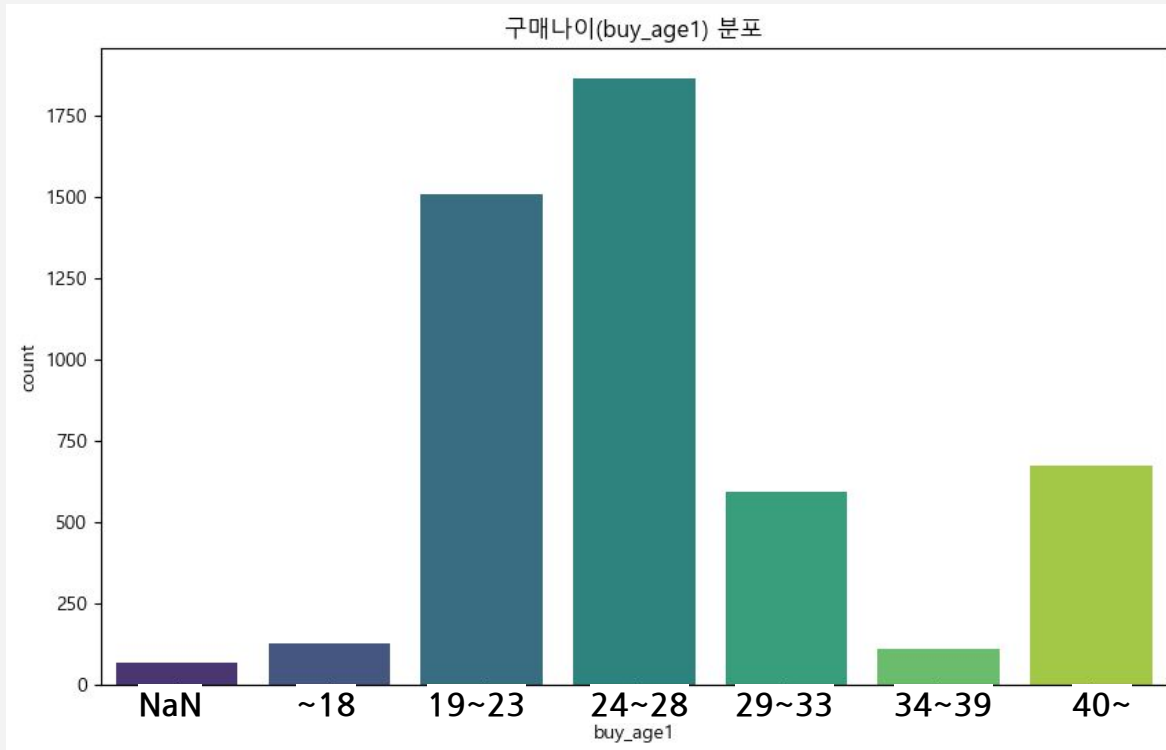
2

3

4

5

3. feature 탐색

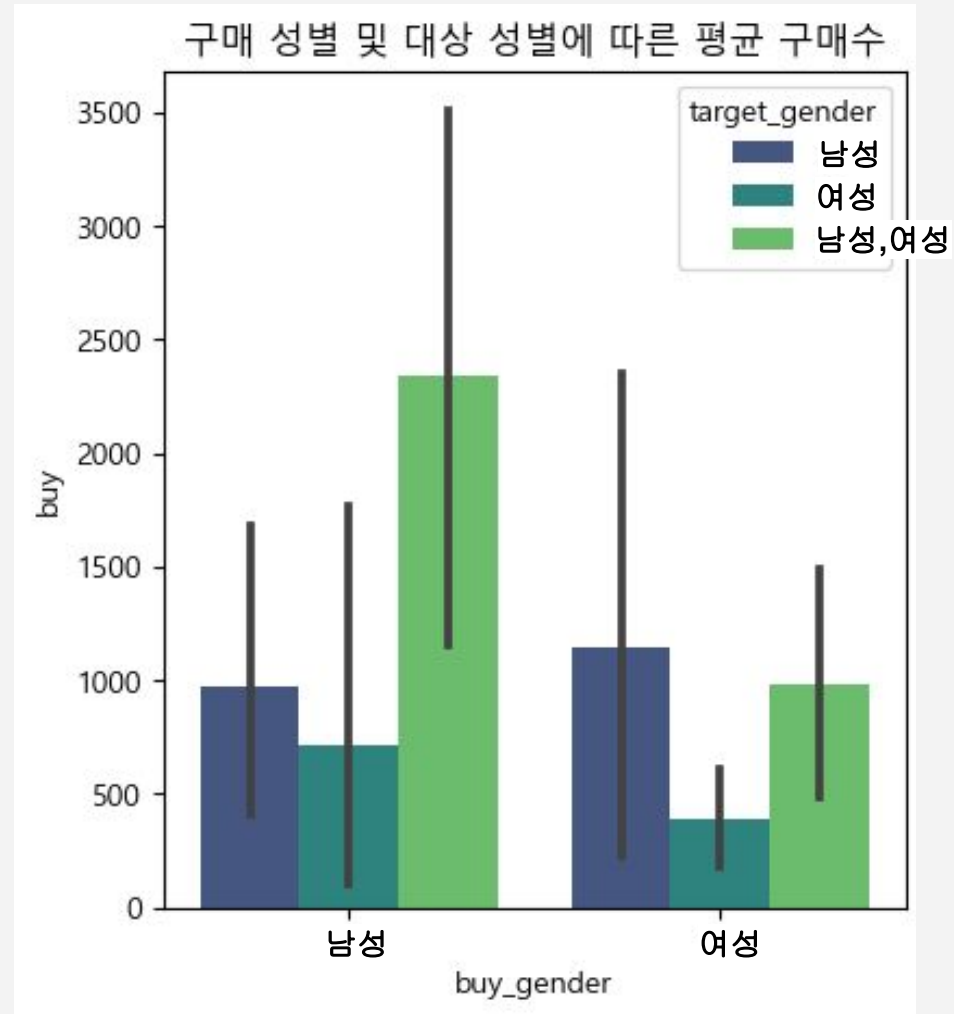


- 주로 20대인 고객을 만족시키는 상품이 많은 것으로 보인다.
- 30대에서 줄어들다가 40대 이상이 높아지는 이유로 ‘힙한 브랜드’를 판매하는 무신사의 이미지를 고려하여 학부모 계정으로 구매한 미성년자를 고려할 수 있다.

3. feature 탐색

- 남성은 남성 제품보다 공용 제품을 더 많이 구매
- 남성이 여성 제품을 구매한 경우도 적지 않다.
- 여성도 여성 겨냥 제품보다 남자 혹은 공용제품을 더 많이 구매

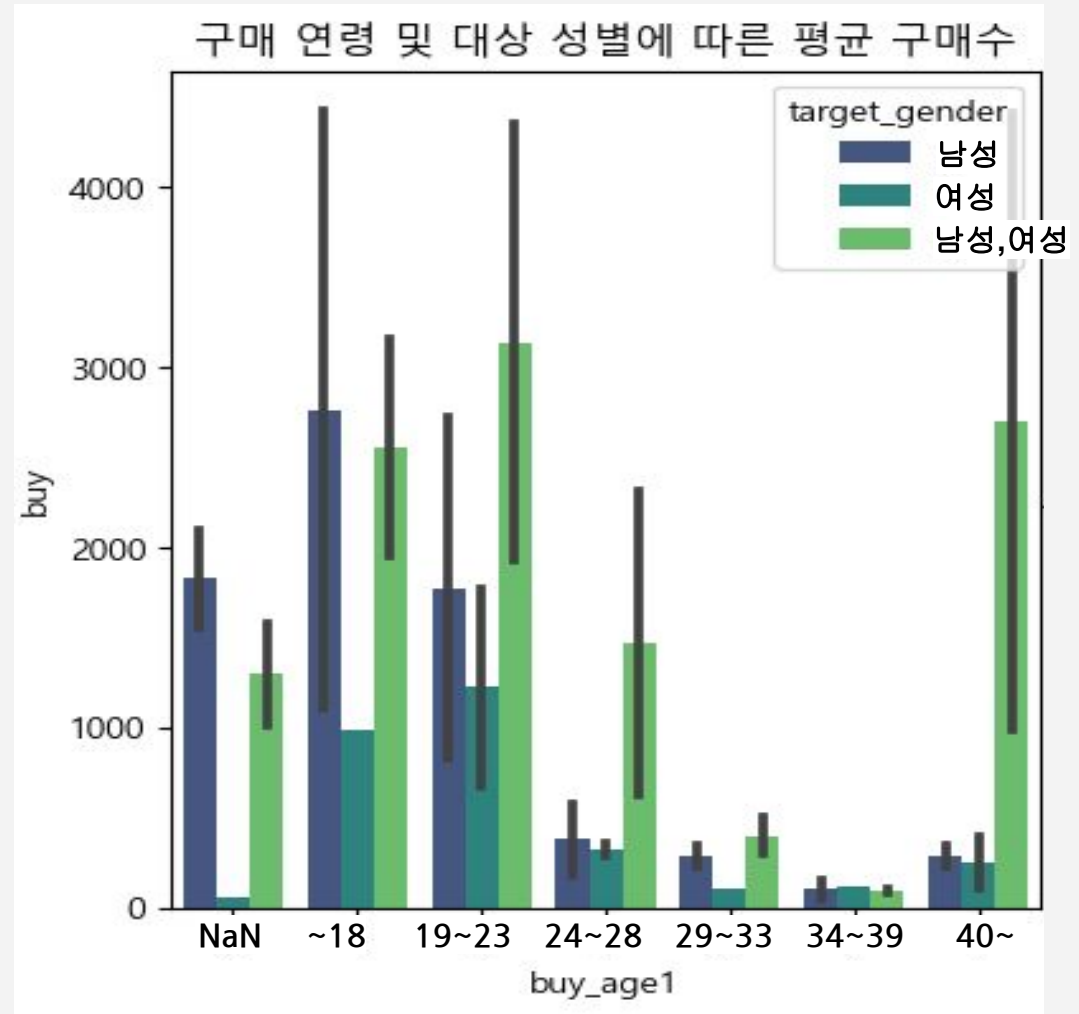
💡 단, 수집한 데이터 카테고리가 상의, 아우터, 바지로 여성 전용 제품들은 포함되어 있지 않기 때문에 모든 카테고리로 확대 적용할 수 없다.



3. feature 탐색

- 20대 이상의 모든 연령층에서 공용 제품 판매량이 가장 많다.
- 10대에서는 남성 타겟 제품이 가장 많다
- 여성 타겟 제품은 비교적 판매량이 떨어진다.

💡 단, 수집한 데이터 카테고리가 상의, 아우터, 바지로 여성 전용 제품들은 포함되어 있지 않기 때문에 모든 카테고리로 확대 적용할 수 없다.



1

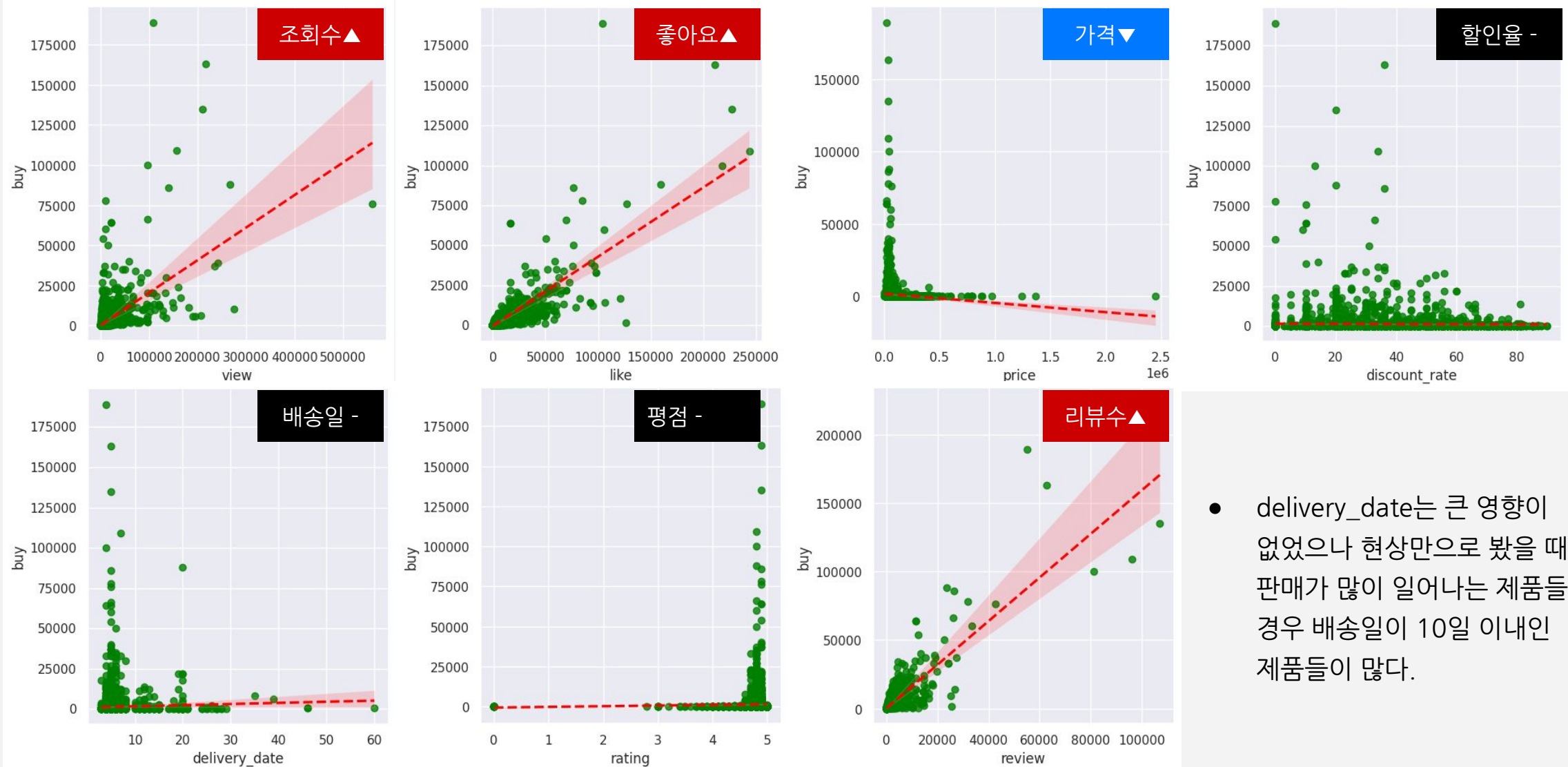
2

3

4

5

3. feature 탐색



- delivery_date는 큰 영향이 없었으나 현상만으로 봤을 때, 판매가 많이 일어나는 제품들의 경우 배송일이 10일 이내인 제품들이 많다.

1

2

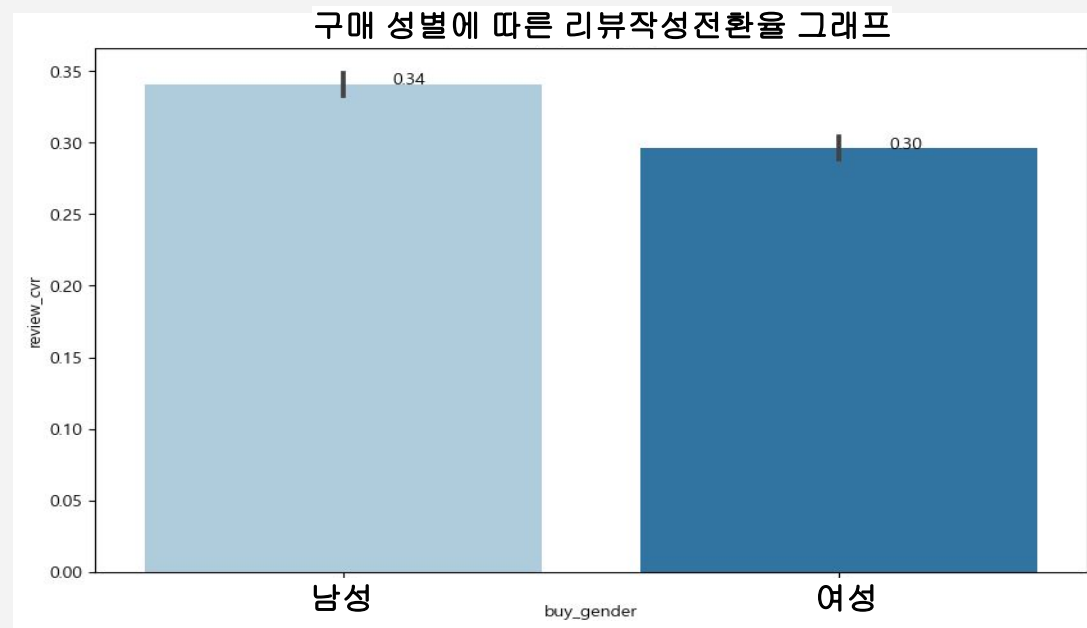
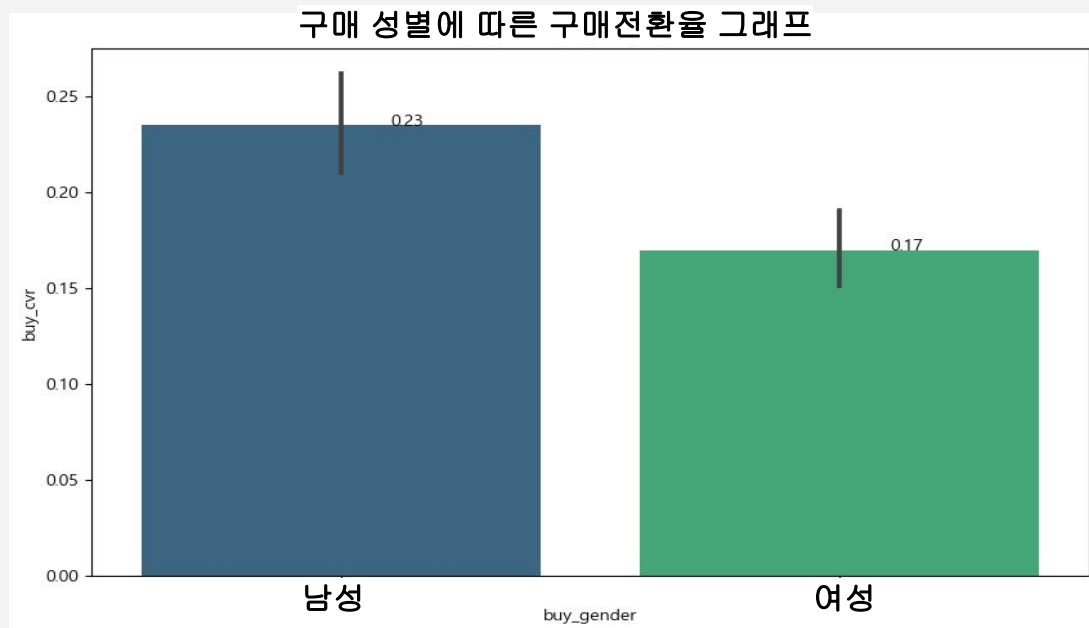
3

4

5

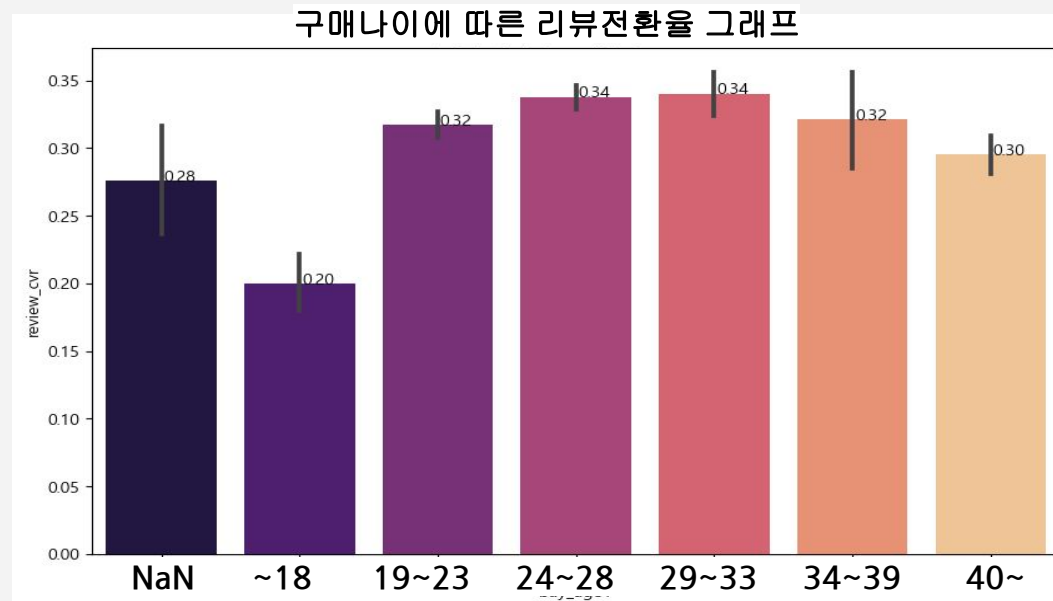
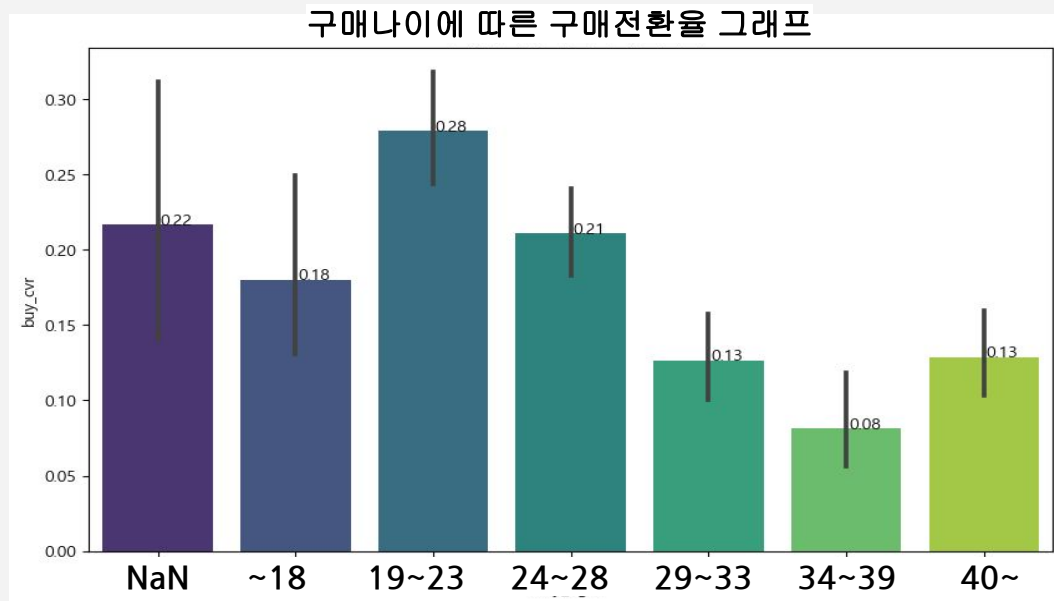
4. 구매성별에 따른 구매전환율 및 리뷰작성전환율

- 구매전환율: 판매량/조회수
- 남성의 구매전환율이 여성의 구매전환율보다 약 0.07% 정도 더 높음.
- 여성이 남성보다 실제 구매를 하기까지 더 많은 시간과 노력을 투자한다고 추측
- 남성의 리뷰작성전환율이 여성의 리뷰작성전환율보다 0.04% 더 높음.
- 남성이 여성의 경우보다 리뷰 작성에 좀 더 긍정적이거나 조금 더 성실하다고 볼 수 있음.
- 리뷰작성전환율: 리뷰수/판매량



4. 구매나이에 따른 구매전환율 및 리뷰작성전환율

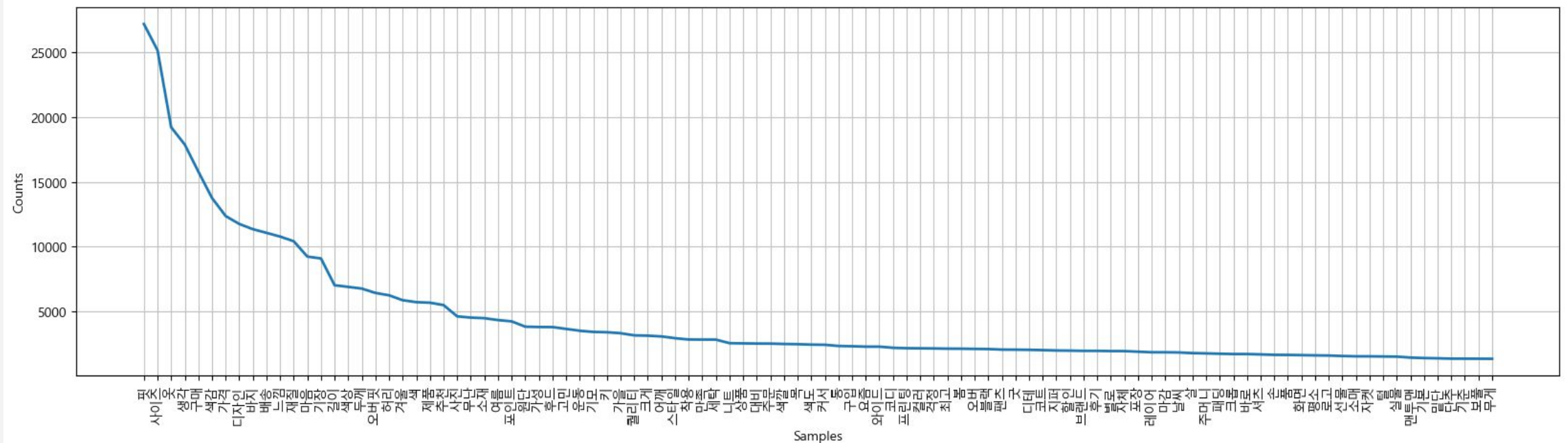
- 구매 전환율을 10대 후반에서 20대 초반이 가장 높고, 이후로 점점 떨어지다가 40세 이상에서 다시 상승
- 리뷰 작성은 10대를 제외하고 대부분 비슷한 것을 확인
- 구매 나이에 따른 리뷰작성전환율은 구매 나이에 따른 구매전환율과 다소 상반됨



5. 리뷰 분석

1. 등장 빈도수

- stopword 반영, 100위까지 추출



```
stopwords_for_wc = ['.', '(', ')', ',', '"', '%', '-', 'X', ')', ' ', 'x', '의', '자', '에', '안', '번', '호', '을', '이', '다', '만', '로', '가', '를', '것',
                    '좀', '더', '때', '맘', '거', '입', '조금', '진짜', '티', '안', '살짝', '수', '제', '아주', '부분', '감', '정말', '정도', '저', '완전', ...]
```

💡 100개를 추출을 반복하면서 계속 stopwords를 추가하며 유의미한 단어만 나올 수 있도록 진행

5. 리뷰 분석

1. 워드클라우드로 시각화

- 150위까지 추출



{ '핏': 27162, '사이즈': 25136, '옷': 19212, '생각': 17872, '구매': 15783, '색감': 13758, '가격': 12371, '디자인': 11760, '바지': 11357, '배송': 11075, ... }

1

2

3

4

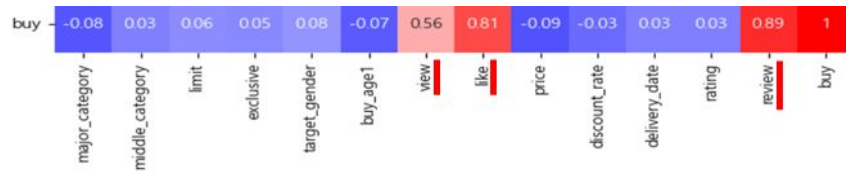
5

4. 모델 학습과 평가



1. Ridge

- “buy”와 상관계수가 높았던 피처만 선정



```
1 df_copy_corr = df_copy[['view', 'like', 'review', 'buy']]
2 df_copy_corr
```

	view	like	review	buy
0	1200.0	493	36	50.0
1	4900.0	650	61	200.0
2	3600.0	2536	41	150.0
3	1000.0	569	62	150.0
4	11000.0	2561	34	100.0
...
4926	2900.0	771	17	50.0
4927	2100.0	1275	222	300.0
4928	1400.0	198	33	100.0
4929	3300.0	2028	87	100.0
4930	400.0	368	22	50.0

4931 rows x 4 columns

- StandardScaler 적용

```
1 X = df_copy_corr.drop('buy', axis=1)
2 y = df_copy_corr['buy']
3
4 X.shape, y.shape
```

✓ 0.0s

((4931, 3), (4931,))

```
1 from sklearn.preprocessing import StandardScaler
2
3 SS = StandardScaler()
4
5 X_ss = SS.fit_transform(X)
6 X_ss_pd = pd.DataFrame(X_ss, columns=X.columns)
```

✓ 0.0s

- test_size=0.2, random_state=13 으로 X, y 트레인
테스트 데이터 스플릿

1. Ridge

- “Ridge” 모델 및 GridSearch를 통한 최적의 alpha 값 적용
 - alpha=100

```
1 from sklearn.linear_model import Ridge
2 from sklearn.model_selection import GridSearchCV
3
4 ridge = Ridge()
5 params = {'alpha' : [0.0001, 0.001, 0.01, 0.1, 1, 10, 100] }
6 gridsearch = GridSearchCV(ridge, param_grid = params)
7 # 적절한 알파값을 찾기 위해 그리드 서치를 사용
8 gridsearch.fit(X_train, y_train)
9
10 r_ridge_estimator = gridsearch.best_estimator_
11 r_y_train_pred = r_ridge_estimator.predict(X_train)
12 r_y_test_pred = r_ridge_estimator.predict(X_test)
13
```

```
14 print(gridsearch.best_estimator_)
✓ 0.2s
Ridge(alpha=100)
```

- 데이터 학습 및 결과 도출

```
train
MAE : 533.7925999537798
MSE : 7844584.865188178
RMSE : 2800.8186062628506
R-Squared : 0.8171773500903057
=====
test
MAE : 498.20317805229104
MSE : 3059689.1776001677
RMSE : 1749.1967235277361
R-Squared : 0.8205681389537077
```

- StandardScaler 적용 및 중요 feature 추출 df
 - r2: 0.8205681389537077
 - RMSE: 1749.1967235277361

2. OLS

- LabelEncoder로 범주형 변수 매핑

- 대분류(major_category)

['바지' , '상의' , '아우터'] => [1, 0, 2]

- 중분류(middle_category)

['기타바지' , '기타상의' , '긴소매 티셔츠' ...]
=> [7, 8, 10, 2, 9, ...]

- 구매나이(buy_gender)

[('19~23세' , '24~28세') , ('19~23세' , '29~33세') ,
...]
=> [11, 4, 14, 8, 0, 19, 13, ...]

- feature 추출

```
[ 'major_category' , 'middle_category' , 'limit' ,  
  'exclusive' , 'target_gender' , 'buy_gender' , 'view' ,  
  'like' , 'price' , 'discount_rate' , 'delivery_date' ,  
  'rating' , 'review' , 'buy_age1' ]
```

- OLS 모델에 학습시킬 데이터 set

1. name, number, buy_age를 제외한 모든 칼럼을 feature로 두고 상수항을 추가한 경우
2. 1에서 데이터를 분리하여 4개 칼럼 모두 변수로 고려한 경우
3. 상위 3개 상관관계를 feature로 두고 상수항을 추가한 경우
4. 3번에서 데이터를 분리하여 4개 칼럼 모두 변수로 고려한 경우

2. OLS

- OLS

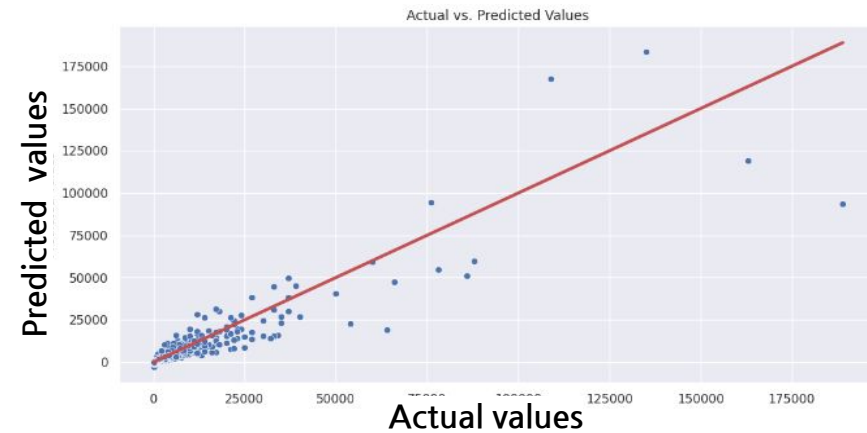
```
lm = sm.OLS(y, sm.add_constant(X)).fit()
y_pred = lm.predict(sm.add_constant(X))

mae = mean_absolute_error(y, y_pred)
mse = mean_squared_error(y, y_pred)
rmse = np.sqrt(mse)
```

- 성능 평가

- r2: 0.821
- MAE: 582.847048883963
- MSE: 8787599.0122890175
- RMSE: 2695.8885429958785

- 실제값과 예측값 대조 그래프



💡 4가지를 돌려본 결과 1번 data set의 결과가 우수했다.

3. Multiple Regression + Lasso

- 모델링에 사용할 피쳐 총 14개 추출

```
X = df.drop(['name', 'number', 'buy_age', 'buy', 'major_category', 'middle_category'], axis=1)
print(X)
y = df["buy"]
print('\n')
print(y)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4931 entries, 0 to 4930
Data columns (total 14 columns):
#   Column          Non-Null Count  Dtype
---  ---
0    limit          4931 non-null   int64
1    exclusive      4931 non-null   int64
2    target_gender   4931 non-null   int64
3    buy_gender      4931 non-null   float64
4    view           4931 non-null   float64
5    like           4931 non-null   int64
6    price          4931 non-null   float64
7    discount_rate   4931 non-null   int64
8    delivery_date   4931 non-null   float64
9    rating         4931 non-null   float64
10   review         4931 non-null   int64
11   score          4931 non-null   float64
12   buy_age1       4931 non-null   int64
13   buy_age2       4931 non-null   int64
dtypes: float64(6), int64(8)
memory usage: 577.9 KB
```

- 데이터 분할 후 다항회귀PolynomialFeatures

```
# 데이터 분할
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=13)

from sklearn.preprocessing import PolynomialFeatures

poly=PolynomialFeatures(include_bias=False)
poly.fit(X_train)
train_poly=poly.transform(X_train)
print(train_poly.shape)

(3698, 119)
```

```
from sklearn.linear_model import LinearRegression

# 모델링
reg = LinearRegression()
reg.fit(train_poly, y_train)

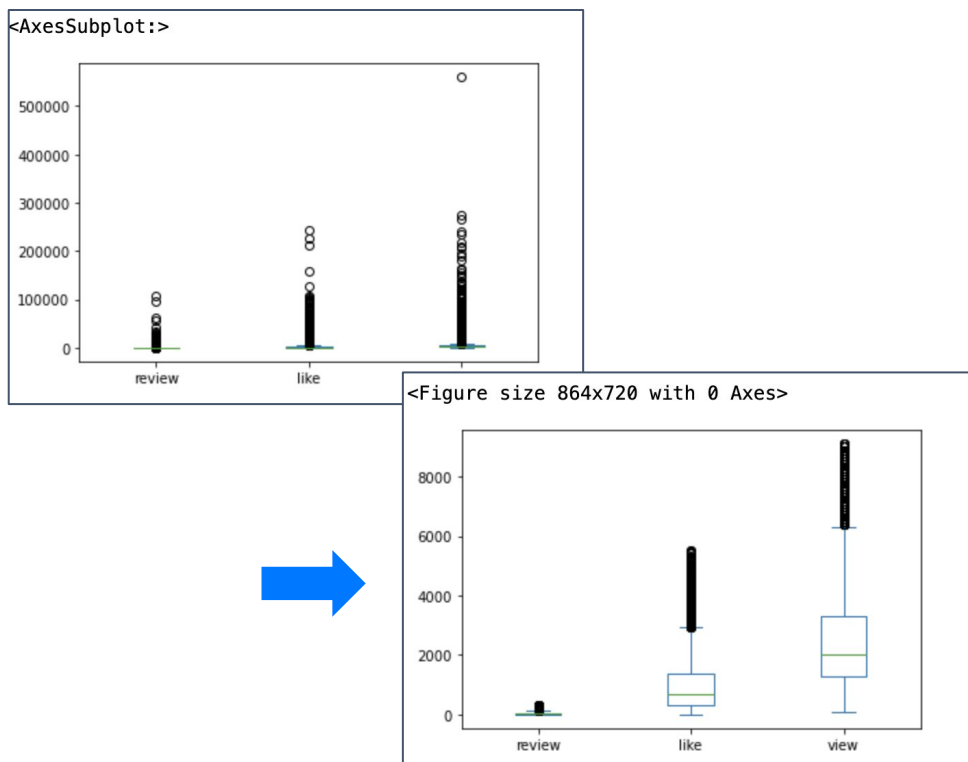
print(reg.score(train_poly, y_train))
print(reg.score(test_poly, y_test))

0.955703045779865
-1.5662402481561664
```


3. Multiple Regression + Lasso

- 상관계수 상위 3개 피쳐 이상치 제거

- view: 0.562318
- like: 0.814722
- review: 0.890346



이상치 제거

- StandardScaler 적용 및 라쏘회귀

```
from sklearn.preprocessing import StandardScaler

ss = StandardScaler()
ss.fit(train_poly)

train_scaled = ss.transform(train_poly)
test_scaled = ss.transform(test_poly)
```

```
lasso = Lasso(alpha=1)
lasso.fit(train_scaled, y_train)

print(lasso.score(train_scaled, y_train))
print(lasso.score(test_scaled, y_test))
```

0.7519617771509419
0.7401587127140274

- StandardScaler 적용 df
 - r2: 0.740158712740274
 - RMSE train: 110.564242905521
 - RMSE test: 112.51141758526

라쏘회귀 적용

1

2

3

4

5

4. LGBM Regressor

- Original df

	limit	exclusive	...	view	price	like
0	2	0	...	1200.0	39800.0	493
1	2	3	...	4900.0	53550.0	650

- StandardScaler 적용 df

	limit	exclusive	...	ss_view	ss_price	ss_like
0	2	0	...	-0.276859	-0.506499	-0.293186
1	2	3	...	-0.067121	-0.350067	-0.278775

- Log scale 적용 df

	limit	exclusive	...	log_view	log_price	log_like
0	2	0	...	7.090910	10.591647	6.202536
1	2	3	...	8.497195	10.888390	6.478510

- LGBM Regressor + GridSearchCV

```
...
lgbmr = LGBMRegressor(n_estimators= 10,
max_depth= 10, num_leaves= 30,
learning_rate=0.1, reg_alpha=0.1, n_jobs=-1,
verbosity=-1)

params = {
    'n_estimators': [100, 500, 1000],
    'max_depth': [10, 20, 30],
    'num_leaves': [30, 60, 100],
    'learning_rate': [0.05, 0.1],
    'reg_alpha': [0.1, 0.5, 0.8, 1.0],
}

gridsearch = GridSearchCV(estimator=lgbmr,
param_grid=params, cv=5,
scoring='neg_mean_absolute_error')
gridsearch.fit(X_train, y_train)
...
```

- Original df
 - o r2: 0.69, RMSE: 2296.00
- StandardScaler df
 - o r2: 0.69, RMSE: 2281.20
- Log scale df
 - o r2: 0.69, RMSE: 2299.72

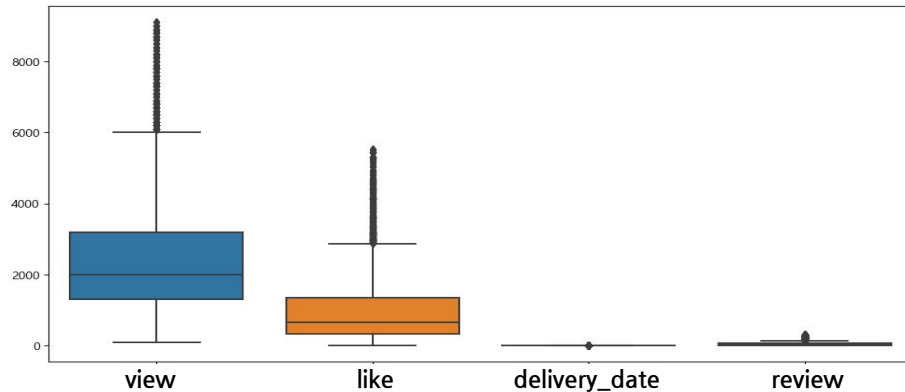
4. LGBM Regressor

- Original df에서 'view', 'like', 'delivery_date', 'review' feature 이상치 제거

```
...
Q1 = df_refined[['view', 'like', 'delivery_date',
'review']].quantile(q=0.25)
Q3 = df_refined[['view', 'like', 'delivery_date',
'review']].quantile(q=0.75)

IQR = Q3-Q1

IQR_df = df_refined[(df_refined['view'] <=
Q3['view']+1.5*IQR['view']) & (df_refined['view']
>= Q1['view']-1.5*IQR['view'])]
...
```



이상치 제거

- IQR 기준 이상치 제거된 데이터

```
print('original_df :', len(df_refined))
print('IQR_df :', len(IQR_df))
```

- original_df : 4931
- IQR_df : 3690

- LGBM Regressor + GridSearchCV 최종

- IQR 이상치 제거된 Original df:
 - MAE: 51.89890706639263
 - MSE: 12835.762996669655
 - r2_score_train: 0.9389638944944941
 - r2_score_test: 0.7625633360665581**
 - RMSE train: 54.513610478476295
 - RMSE test: 113.29502635451239

💡 상관관계가 높았던 feature만 선별해서 학습해도 성능은 유사했다.

학습 및 성능 평가

1

2

3

4

5

5. XGBoost Regressor

- 총 14개 feature 사용

```
'limit', 'exclusive', 'target_gender', 'buy_gender',
'buy_age1', 'buy_age2', 'view', 'like', 'price',
'discount_rate', 'delivery_date', 'rating',
'review', 'score'
```

- 총 8개의 데이터 set으로 모델 학습

- 아래에서 각각 하이퍼파라미터 적용/미적용

0	Original df
1	상관관계 상위 10개 feature 추출 df
2	이상치 제거 df
3	Log scale 적용 df

💡 Original df에 하이퍼파라미터 적용한 경우의 성능이 가장 우수했다.

- RandomSearch를 이용한 하이퍼파라미터 튜닝

```
param = {'n_estimators': 277, 'max_depth': 77,
'subsample': 0.7955167781453465, 'colsample_bytree':
0.5842153210611298, 'gamma': 0.7819910241112622,
'min_child_weight': 1,}
```

5. XGBoost Regressor

• 모델 생성 및 cross_val_score

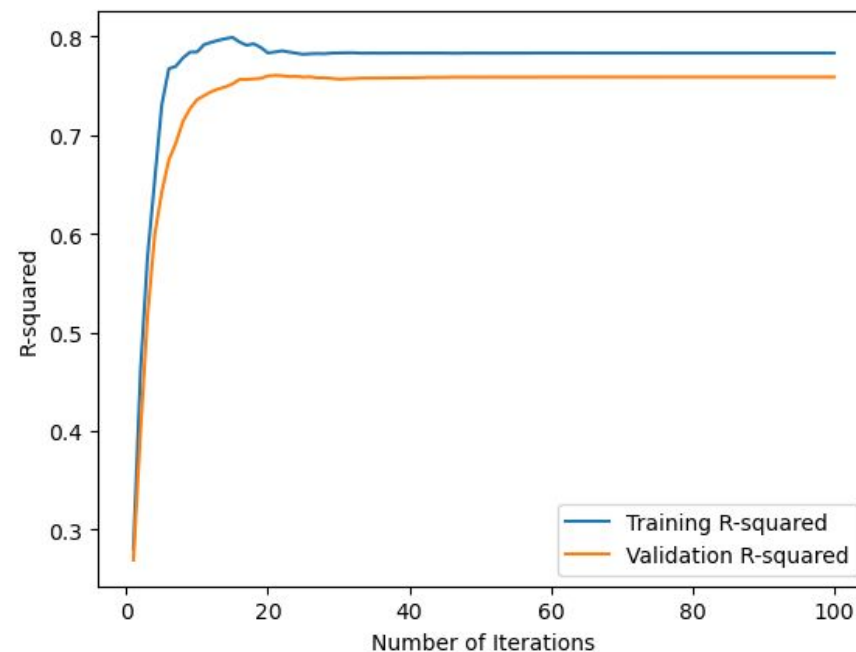
```
model = xgb.XGBRegressor(**param)

kf = KFold(n_splits=5, shuffle=True, random_state=42)

train_r2_scores = cross_val_score(model, X_train,
                                   y_train,
                                   cv=kf, scoring='r2')
val_r2_scores = cross_val_score(model, X_val, y_val,
                                 cv=kf, scoring='r2')
train_mae_scores = cross_val_score(model, X_train,
                                    y_train, cv=kf, scoring='neg_mean_absolute_error')
val_mae_scores = cross_val_score(model, X_val, y_val,
                                  cv=kf, scoring='neg_mean_absolute_error')
train_mse_scores = cross_val_score(model, X_train,
                                    y_train, cv=kf, scoring='neg_mean_squared_error')
val_mse_scores = cross_val_score(model, X_val, y_val,
                                  cv=kf, scoring='neg_mean_squared_error')
```

• 평가 및 과적합 그래프

- Original df + 하이퍼파라미터 튜닝
 - o **r2: 0.7588801888271577**
 - o mae: 633.1560355491251
 - o mse: 11986432.662931258
 - o rmse: 3462.1427848849994



모델 성능 비교

모델링	이상치 제거	R2_score	MAE	MSE	RMSE
OLS	X	0.82	563	6740000	2595.69
Multiple Regression + Lasso	0	0.74	58.56	12658.82	112.51
Ridge	X	0.82	498.20	3059689.18	1749.20
LGBM Regressor	0	0.77	51.86	12423.76	111.46
XGB00ST Regressor	X	0.76	633.16	11986432.66	3462.14

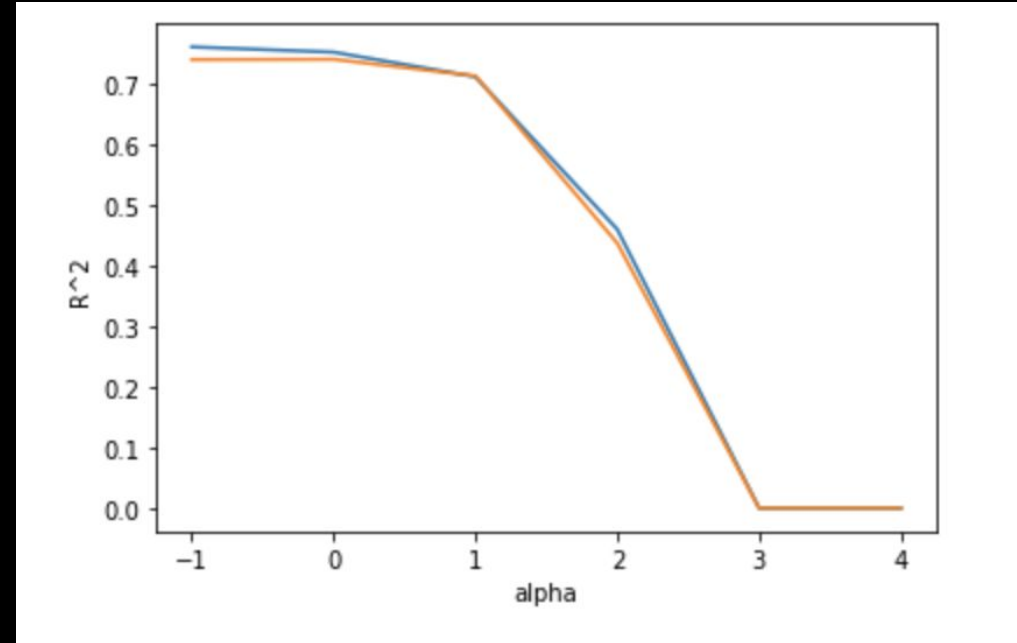
5. 결론



1. 최종 모델 Multiple Regression + Lasso

- R^2 _score 만으로 보았을 때 설명력이 가장 높은 모델은 아니지만 종합적인 성능지표를 비교했을 때 우수한 것으로 판단
- 트레인, 테스트 데이터의 과적합을 가장 효과적으로 해결한 모델
 - Lasso의 α 값 도출 그래프 참고
 - train, test set의 RMSE

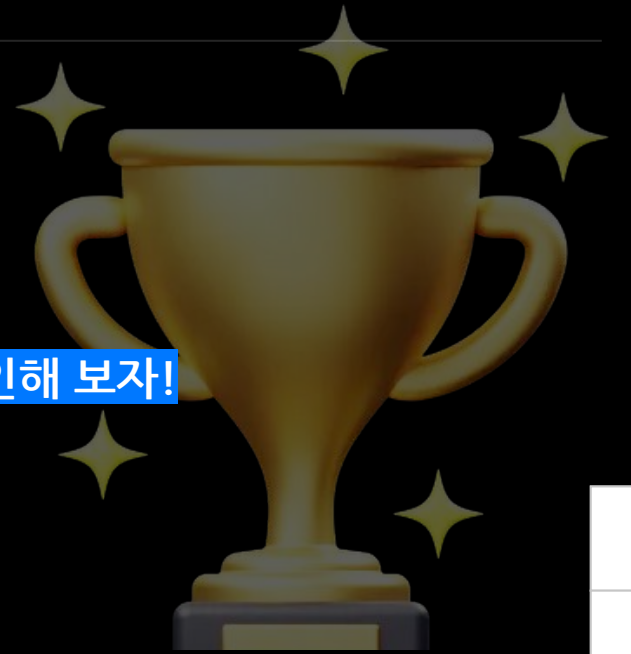
- StandardScaler 적용 df
 - r^2 : 0.740158712740274
 - RMSE train: 110.564242905521
 - RMSE test: 112.51141758526



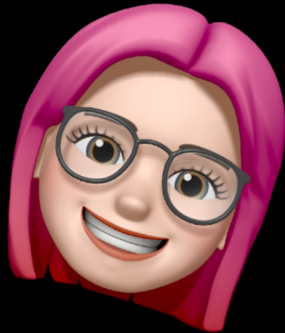
모델링	이상치 제거	R^2 _score	MAE	MSE	RMSE
Multiple Regression + Lasso	0	0.74	58.56	12658.82	112.51

2. Battle! 어떤 제품이 가장 잘 팔릴까?

- 이전 프로젝트의 목적은 판매량을 예측하여 제조사와 무신사 플랫폼의 리스크를 줄이는 것.
- 팀원이 모두 제조사가 되어 Lasso 모델을 이용해 가상 제품 feature를 넣어 예측값을 확인해 보자!



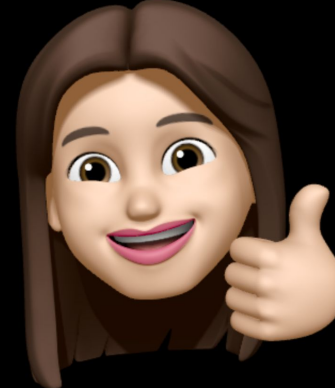
주희



지현



병찬



청하



승현

1

2

3

4

5

2. Battle! 어떤 제품이 가장 잘 팔릴까?

브랜드 명	limit	exclusive	target_gender	buy_gender	view	like	price	discount_rate	delivery_date	rating	review	score	buy_age1	buy_age2
하승현(a)	1	1	2	1	5000	600	10000	30	4	4.8	2000	0.9	2	2
이청하(b)	1	1	2	1	10000	10000	35000	26	4	4.8	5000	0.9	2	2
이병찬(c)	1	1	0	1	5000	700	45000	25	5	4.8	700	0.9	2	2
김지현(d)	1	1	2	1	16875	9450	34500	20	5	4.8	7420	0.96	2	0
김주희(e)	1	0	2	1	4000	500	37500	35	5	4.7	500	0.94	2	2

Winner is,

브랜드 명	predicted buy
하승현(a)	73
이청하(b)	284
이병찬(c)	37
김지현(d)	445
김주희(e)	5



지현



1
2
3
4
5

3. Lesson & learned

주요 이슈와 문제

- 결측치 처리에 대한 기준 설정
- 무신사 웹 내의 태그값과 웹 구조 변동
- 웹크롤링할 때 수집할 범위 설정의 중요성
- 수집한 데이터에서 예측값이 없는 데이터가 대량 수집되는 경우 많은 양의 데이터가 drop 되었음
- 모델의 과적합 판단

교훈과 배운 점

- 실제로 머신러닝에 적용시킬 수 있는 데이터를 빨리 확보하는 것이 중요
- XGBoost나 LGBM과 같은 앙상블 모델이 성능이 가장 좋을 것이라 생각했으나 그렇지 않았음
- 데이터에 맞는 모델을 사용하는 것이 바람직함
- 예측값이 포함된 파생변수를 생성한 경우 다루는 데 있어 주의 필요
- 모델 학습 과정에서는 data leakage(학습 데이터에 이미 예측 대상이 반영된 데이터가 흘러들어가는 것) 방지 필수

회고

- 데이터 수집의 양이 좀 더 많았으면 과적합을 방지할 수 있을까?
- 데이터 수집을 일찍 끝내고 더 많은 종류의 모델과 하이퍼 파라미터를 학습 및 테스트하면 좋은 성능의 모델을 찾을 수 있지 않을까?
- 데이터를 좀 더 수집하고 전처리를 좀 더 효과적으로 했다면 어땠을까?

MUSINSA 2024.01

END
감사합니다.