

2023 BDA 데이터 분석·활용 공모전

Track 2 모델링 고도화

KHUDA 심화트랙 컨퍼런스

쿠벤져스 팀 | 강지원, 노명은, 송예지, 임소영

CONTENTS

1 프로젝트 개요

- 공모전 소개
- Track 2 소개

2 데이터 전처리

- 데이터 소개
- 데이터 분석 과정
- 데이터 전처리

3 모델링

- 모델링 과정
- 모델링 결과

4 최종 결과 및 계획

- F1-score
 - 보완할 점
-

프로젝트 개요

1.1 공모전 소개

1.2 track 2 소개

01

공모전 소개

2023 BDA 데이터 분석·활용 공모전

주관: BDA

track 1 / track2

예선심사 (~5/9)

분석계획서 제출

본선심사(5/18~6/1)

실제 모델링 후 결과제출

발표심사(6/4)

CJ제일제당센터 방문 후 PPT 발표심사



2023 BDA 데이터 분석· 활용 공모전

2023.04.18 ~ 06.04

주최 : (AI)한국빅데이터학회

주관 : BDA (Big Data Analytics)

후원 : CJ제일제당

협력 학회 및 동아리 : KUBIG, KHEDA, ESC, ESAA, DAri-B, DSA

참가자격 : BDA 57L 6기 우수학회원 / 협력 학회 및 동아리 개인 또는 4명 이하 팀

대상분야 : 데이터분석기초, 종급, 고급반

대상분야

	Track 1	Track 2
대상(1팀)	250,000원	250,000원
활무수상(1팀)	150,000원	150,000원
우수상(1팀)	100,000원	100,000원

공모분야 및 내용

Track1: 시작화 인사이트

Track2: 모델링 고도화

참가방법

구글폼 링크로 2023년 4월 18일 ~ 23일 접수
(<https://forms.gle/94UJ5yphiTxdVjfxkdwu8>)

2023 BDA 데이터 분석·활용 공모전

Track 2 소개

목표

2023년도 1월 CJ더마켓 고객 주문 데이터를 활용해

1) 일반회원과 2) 임직원의 프라임회원을 예측하는 모델링 진행

모델1 일반회원 중 프라임회원을 구분하는 모델링

모델2 임직원 중 프라임회원을 구분하는 모델링

-> 총 2개의 모델링 진행

POINT

프라임회원인지 아닌지를 구분하는

"Binary Classification 문제"

프라임회원이란?

CJ더마켓의 스페셜멤버십인

the프라임에 가입한 회원



매일매일 더 큰 혜택

the프라임 멤버십

2,000원에 이 모든 혜택을!

the프라임은 월회비 2,000원으로 전용혜택을 누릴 수 있는
프리미엄 멤버십 서비스입니다.

the프라임 멤버십만의
특별한 혜택 6가지



무료배송

매월 3회 제공



7% 무제한

횟수 무제한 상시할인



3천원 쿠폰

전용상품 할인쿠폰



회원전용DAY

주 1일 특가할인



창고형마켓

2개월마다 BIG세일



전용체험단

신제품 및 히트상품

데이터 전처리

- 1.1 데이터 소개
- 1.2 데이터 분석 과정
- 1.3 데이터 전처리

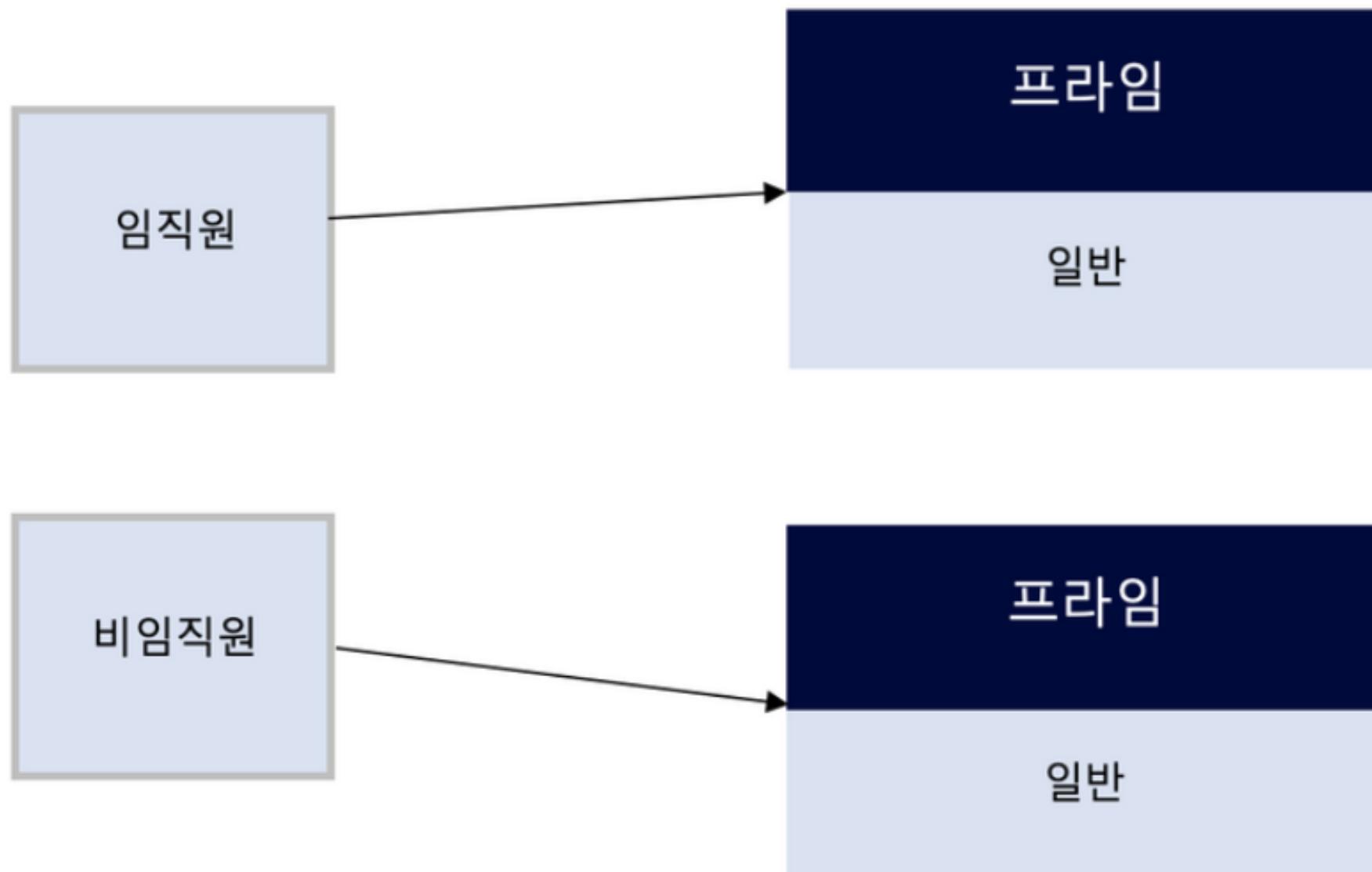
02

데이터 소개

2023년도 1월 CJ 더마켓 고객 주문 데이터

scd(주문번호)	product_name	net_order_qty	net_order_amt	gender	age_grp	데이터 분리	Target Att
주문번호	상품명	주문량	주문 금액 (스케일링)	성별	나이대	임직원 여부	프라임 회원 여부
20230124153976	잔칫집 식혜 240ml 30입	0.521	0.523	M/F	10/20/30/40/50	Y/N	20230102 Y/N

과제 소개



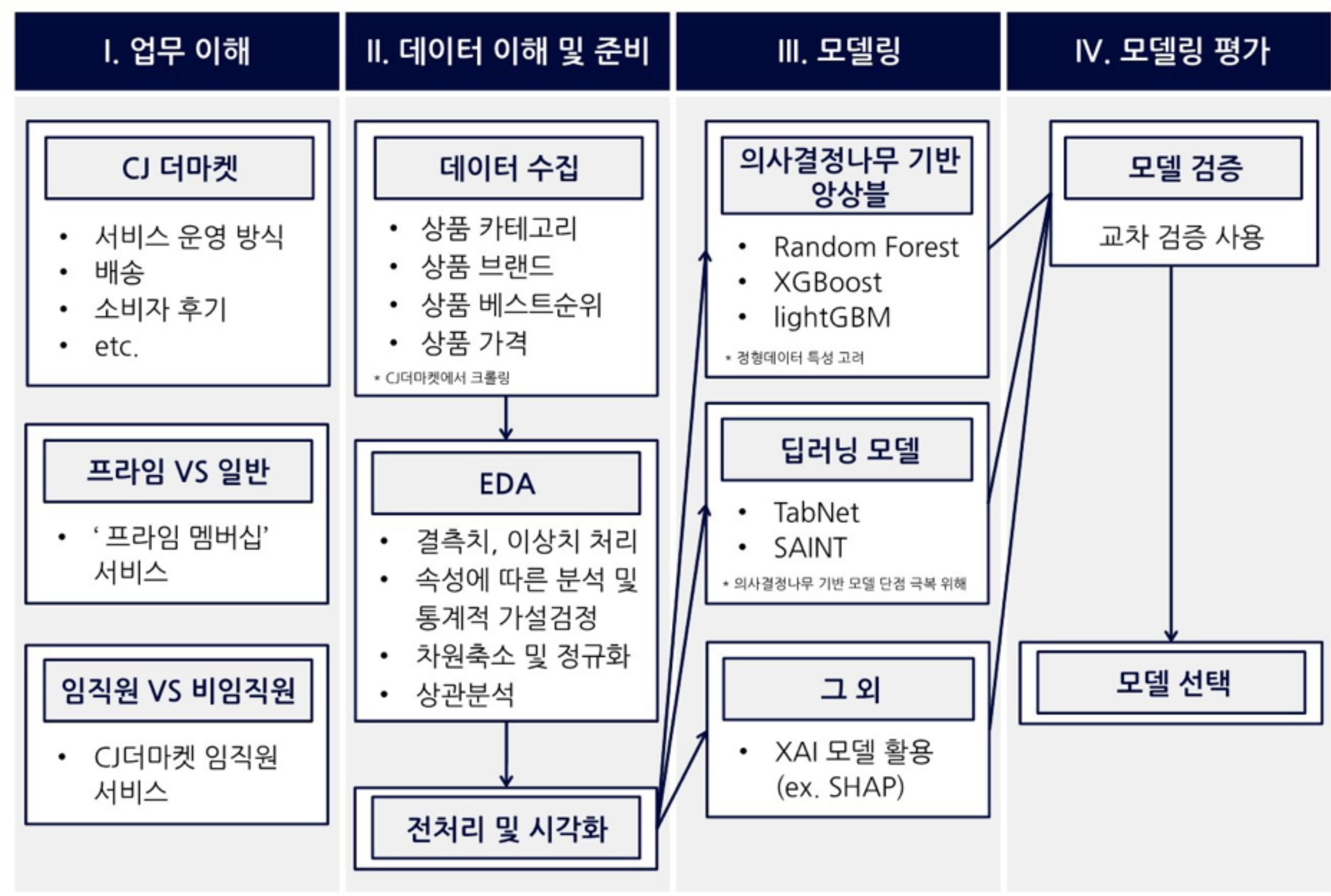
임직원 여부에 따라 각각 프라임회원과 일반 회원을 예측하는 모델링

- F1 스코어도 중요하지만,
- 해당 모델을 사용한 근거와 로직이 가장 중요

평가 항목	배점
실현가능성	30
분석방법 우수성	25
창의성	20
과제 이해도	10
발표	15

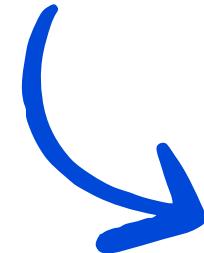
본선 심사 기준

진행 과정 소개



Aggregation

scd	product_name net_order_qty net_order_amt gender age_grp employee_yn order_date prime_yn
20230101963226	
20230101963226	
20230101963226	
20230101963226	데이터 공개 금지



한 고객의
여러 레코드

- 한 명의 고객이 하루에 여러 상품을 구매한 경우
- 이렇게 여러 행(레코드)로 이루어진 것을 알 수 있음

모델링을 위한
Aggregation

- 이 그대로 모델링의 인풋으로 사용할 수 없기에
- 집계(Aggregation)를 하여 하나의 레코드로 표현하기로 함

Aggregation

Aggregation 전

- 고객 한 명에 여러 주문 레코드

scd	product_name	net_order_qty	net_order_amt	gender	age_grp	employee_yn	order_date	prime_yn
20230101963226								N
20230101963226								N
20230101963226								N
20230101963226							데이터 공개 금지	N

Aggregation 후

- 고객 한 명에 하나의 레코드

scd	gender	order_date	pdn_unique_cnt	net_order_qty_sum	net_order_qty_mean	diff_meanPrice	...	category_1	brand_1
unique 한 값	성별	구매 날짜	scd 중복이 얼마나 됐는지 구매한 상품 종류 개수	상품 주문 개수	상품 주문 개수 평균	구매가격 편차	...(생략)	category_1을 산 개수	brand_1를 산 개수

Aggregation 예시

카테고리 칼럼 생성

CJ더마켓

카테고리 크롤링

데이터 결합 및
결측치 채우기

- 셀레니움 활용
 - 크롤링 로직
 - 각 카테고리를 순서대로 클릭하는 반복문
 - 총 페이지를 계산하여 하나씩 클릭후, 그 안에 있는 상품명 크롤링
 - 데이터프레임으로 저장 후, 주어진 데이터와 결합
-
- 주어진 데이터와 결합했을 때 결측치가 약 47%
 - 상품명이 완전히 같은 경우만 카테고리를 분류하였기 때문
 - 결측치를 채우기 위해 특정 단어가 들어간 상품명을 특정 카테고리에 해당하도록 설계
 - 이후 결측치 2.1%

```
# 카테고리 리스트에 포함되지 않는 예외 처리 함수
def categoryName(product_name):
    if any(keyword in product_name for keyword in ['반찬', '김치']):
        return '국/김치/김/반찬/두부'
```

예외 함수 처리 예시

☰ 전체 카테고리

추천 테마

밥/죽/면

국/김치/김/반찬/두부

만두/피자/치킨

핫도그/떡볶이/간식

돈까스/함박/구이

스팸/닭가슴살/소시지

양념/소스/가루/오일

건강식품

신선식품

음료/생수/시럽

대용량 식자재

CJ더마켓 카테고리 분류

product_name category

잔칫집 식혜 240ml 30입 음료/생수

백설 한입쏙 비엔나 120g*2 스팸/닭가슴살/소시지

카테고리 생성 결과

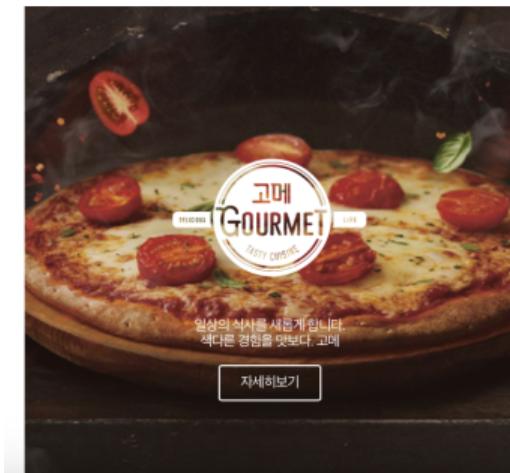
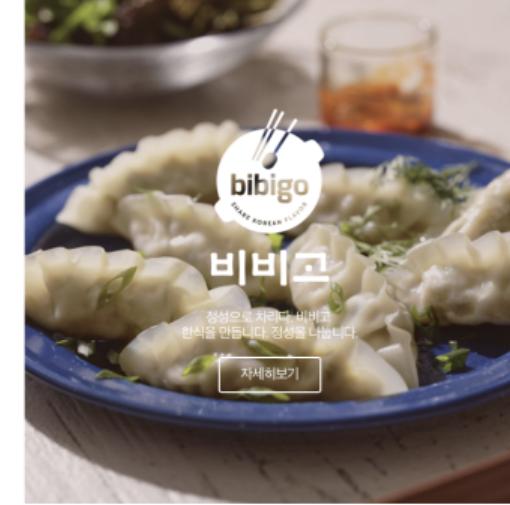
브랜드 칼럼 생성

상품명의 브랜드로 분류

상품명에 브랜드가 없는 경우 & 브랜드명이 다르게 적힌 경우

- 상품명에 브랜드 명이 존재
- 브랜드 기준은 CJ더마켓 브랜드관을 참고
- '햇반'이 존재하면 브랜드 속성값을 '햇반'으로 분류

- 상품명에 브랜드 명이 통일되어 있지 않은 경우가 존재
- 브랜드 명이 적히지 않은 경우도 존재
- 예외인 것을 처리하는 함수 적용



CJ더마켓 브랜드관

```
# 브랜드 리스트에 포함되지 않는 예외 처리 함수
def brandName(product_name):
    if any(keyword in product_name for keyword in ['더건강한', 'The 더건강한']):
        return 'The더건강한'
```

예외 처리 함수 예시

product_name	brand
햇반 소프트밀 단호박죽 420g	햇반
[앱전용특가]비비고 진국육수 소고기양지 500gX6개	비비고
비비고 수제진한김치만두 400gx2	비비고
[700개한정]고메 빅크리스피핫도그520gX2개	고메
비비고 왕교자 1.05kg	비비고

브랜드 구분 결과 예시

유의미한 브랜드 찾기

목적

- 프라임 회원 여부를 구분하는데 큰 영향을 주는, 유의미한 브랜드를 찾고자 함.

카이제곱검정

- '프라임 회원 유무가 어떤 브랜드를 샀는지에 영향을 받을 것이다'를 가정
- 기대 빈도가 5미만인 셀이 전체의 20%를 초과하면 검정 결과가 정확하지 않을 수 있으므로 이를 확인 후 진행
- 임직원, 비임직원 각각 나눠서 진행

p-value

chi2 : 107.54610395271439
p-value : 4.232798674616353e-10

카이제곱검정의 p-value

- p-value가 굉장히 작으므로
'프라임 회원 유무가 어떤 브랜드를 샀는지에 영향을 받는다'고 결론내릴 수 있음

	brand	N	y
0	비비고	1708	2596
1	햇반	1016	1547
2	고메	731	1176
3	백설	681	1200
4	스팸	391	698
5	The더건강한	241	428
6	CJ명가	188	325
7	삼호어묵	140	328
8	한뿌리	141	287
9	행복한콩	110	302
10	쁘띠첼	160	188
11	해찬들	118	199
12	리턴업	103	186
13	즐거운동행	112	176
14	다담	95	189

카이제곱검정을 위한 교차표

유의미한 브랜드 찾기

목적

- 프라임 회원 여부를 구분하는 데 큰 영향을 주는, 유의미한 브랜드를 찾고자 함.

과정

- 프라임 여부에 따라(N, Y) 각각 브랜드의 비율을 파악 후 그 비율의 차이가 큰 것부터 내림차순

```
# 임직원  
empY_df = df[df['employee_yn']=='Y']  
  
# 개수가 아닌 비율 - column 기준  
empY_brand_ratio = pd.crosstab(empY_df['brand'], empY_df['prime_yn'], normalize = 'columns')  
  
# 프라임 유무 비율 차이  
empY_brand_ratio['dff_emp_ratio'] = empY_brand_ratio['N'] - empY_brand_ratio['Y']  
  
# 차이가 음수인 걸 양수로  
empY_brand_ratio = empY_brand_ratio.abs()  
  
# 차이를 기준으로 내림차순  
empY_brand_ratio = empY_brand_ratio.sort_values(by=['dff_emp_ratio'], ascending=[False])  
  
# 그 중 dfff_emp_ratio 가 큰 10개 뽑기  
empY_brand_ratio = empY_brand_ratio[0:10]
```

brand	prime_yn	N	Y	dfff_emp_ratio
비비고	0.259930	0.236408		0.023522
햇반	0.154619	0.140880		0.013739
행복한콩	0.016740	0.027502		0.010762
삼호어묵	0.021306	0.029870		0.008564
쁘띠첼	0.024349	0.017120		0.007229
하선정	0.010957	0.017212		0.006254
백설	0.103637	0.109280		0.005642
한뿌리	0.021458	0.026136		0.004678
고메	0.111246	0.107094		0.004152
스팸	0.059504	0.063564		0.004060

유의미한 순으로 내림차순

유의미한 카테고리 찾기

목적

- 프라임 회원 여부를 구분하는 데 큰 영향을 주는, 유의미한 카테고리를 찾고자 함.

카이제곱검정

- '프라임 회원 유무가 어떤 브랜드를 샀는지에 영향을 받을 것이다' 를 가정
- 기대 빈도가 5미만인 셀이 전체의 20%를 초과하면 검정 결과가 정확하지 않을 수 있으므로 이를 확인 후 진행
- 임직원, 비임직원 각각 나눠서 진행

p-value

chi2 : 30.307441133761863
p-value : 0.0007628438961501412

카이제곱검정의 p-value

- p-value가 0.001보다도 작으므로
'프라임 회원 유무가 어떤 카테고리를 샀는지에 영향을 받는다' 고 결론내릴 수 있음

	category	N	Y
0	국/김치/김/반찬/두부	3016	5299
1	밥/죽/면	1077	1747
2	핫도그/떡볶이/간식	697	1016
3	양념/소스/가루/오일	543	894
4	만두/피자/치킨	548	813
5	스팸/닭가슴살/소시지	378	714
6	음료/생수	320	597
7	돈까스/함박/구이	340	557
8	건강식품	205	362
9	신선식품	79	106
10	밀키트	5	21

카이제곱검정을 위한 교차표

유의미한 카테고리 찾기

목적

- 프라임 회원 여부를 구분하는 데 큰 영향을 주는, 유의미한 카테고리를 찾고자 함.

과정

- 프라임 여부에 따라(N, Y) 각각 카테고리의 비율을 파악 후 그 비율의 차이가 큰 것부터 내림차순

```
# 임직원
empY_df = df[df['employee_yn']=='Y']

# 개수가 아닌 비율 - column 기준
empY_category_ratio = pd.crosstab(empY_df['category'], empY_df['prime_yn'], normalize = 'columns')

# 프라임 유무 비율 차이
empY_category_ratio['dff_emp_ratio'] = empY_category_ratio['N'] - empY_category_ratio['Y']

# 차이가 음수인 걸 양수로
empY_category_ratio = empY_category_ratio.abs()

# 차이를 기준으로 내림차순
empY_category_ratio = empY_category_ratio.sort_values(by=['dff_emp_ratio'], ascending=[False])

# 그 중 dff_emp_ratio 가 큰 10개 뽑기
empY_category_ratio = empY_category_ratio[0:10]
```

category	prime_yn	N	Y	dff_emp_ratio
국/김치/김/반찬/두부	0.418424	0.436995		0.018571
핫도그/떡볶이/간식	0.096698	0.083787		0.012911
만두/피자/치킨	0.076027	0.067046		0.008981
스팸/닭가슴살/소시지	0.052442	0.058882		0.006440
밥/죽/면	0.149417	0.144071		0.005347
음료/생수	0.044395	0.049233		0.004838
신선식품	0.010960	0.008742		0.002218
양념/소스/가루/오일	0.075333	0.073726		0.001607
건강식품	0.028441	0.029853		0.001413
돈까스/함박/구이	0.047170	0.045934		0.001235

유의미한 순으로 내림차순

구매가격 편차

가설

- 상품 구매 가격은 할인율이 적용된 가격으로, 프라임 회원은 상품의 평균가격보다 싼 가격으로 구매했을 것

과정

- 각 상품에 대해 평균 가격을 구함
- diff_meanPrice라는 새로운 칼럼 추가
 - 상품가격 평균과의 차이로, -0.1인 경우 평균보다 0.1만큼 더 싸게(할인) 구매했다는 것을 의미

product_name	net_order_amt	diff_meanPrice
훈제대란 20구	9.407386	0.107433
훈제대란 20구	9.311904	0.011950
훈제대란 20구	9.311904	0.011950
훈제대란 20구	10.052812	0.752859
훈제대란 20구	9.465060	0.165107

동일한 제품이 구매자에 따라 구매 가격이 다른 모습과 diff_meanPrice 칼럼 추가한 모습

- 실제로 모델의 feature_importance에서 높은 중요도를 보인 칼럼

feature_importance	Feature Id	Importances
0 net_order_amt_diff	30.179418	
1 diff_meanPriceMean	16.050006	

상품 주문량, 주문금액 처리

가설

- 프라임 회원과 일반 회원은 상품의 주문량과 총 주문금액에 차이가 존재할 것

과정

- 상품 주문량, 총 주문금액의 평균을 구함
- 각 평균에 얼마나 차이나는지를 나타내는 칼럼 추가

net_order_amt_diff net_order_qty_diff

-28.412975

-6.237367

예시

- net_order_amt_diff
 - 평균 주문 금액과의 편차
- net_order_qty_diff
 - 평균 주문 수량과의 편차

```
# 점이연 상관분석
from scipy import stats
a = plot_df['net_order_qty_diff']
b = plot_df['prime_yn']

stats.pointbiserialr(a, b)
```

SignificanceResult(statistic=0.04390748938116266, pvalue=0.0025701289440467913

점이연 상관분석

- 범주형(프라임 여부)과 연속형(net_order_qty_diff)의 점이연 상관분석 시행결과
- p-value가 0.0025로 작으므로, 주문금액 편차와 프라임 여부 간에 상관관계가 있다고 할 수 있음

최종 input data

```
df_scd_list = df['scd'].value_counts().index.to_list()

scd = []                      # unique 한 값
prime_yn = []                  # 프라임 회원 여부 (타겟속성)
gender = []                     # 성별
age_grp = []                    # 나이대
employee_yn = []                # 임직원 유무
order_date = []                 # 주문 일자
pdn_unique_cnt = []             # 상품 종류 개수, scd 중복횟수. f1score
net_order_qty_sum = []          # 상품 주문 개수
net_order_qty_mean = []          # 상품 주문 개수 평균
net_order_qty_diff = []          # 상품 주문 개수 평균과의 차이
net_order_amt_sum = []           # 구매 금액 총합
net_order_amt_mean = []           # 구매 금액 평균
net_order_amt_diff = []           # 구매 금액 평균과의 차이
diff_meanPriceMean = []          # 할인된 금액의 평균
diff_meanPriceSum = []            # 할인된 금액의 합
weekday = []                     # 요일

for i in df_scd_list:
    scd.append(i)
    prime_yn.append(df[df['scd'] == i]['prime_yn'].iloc[0])
    gender.append(df[df['scd'] == i]['gender'].iloc[0])
    age_grp.append(df[df['scd'] == i]['age_grp'].iloc[0])
    employee_yn.append(df[df['scd'] == i]['employee_yn'].iloc[0])
    order_date.append(df[df['scd'] == i]['order_date'].iloc[0])
    pdn_unique_cnt.append(len(df[df['scd'] == i]['net_order_qty']))
    net_order_qty_sum.append(df[df['scd'] == i]['net_order_qty'].sum())
    net_order_qty_mean.append(df[df['scd'] == i]['net_order_qty'].mean())
    net_order_qty_diff.append(df[df['scd'] == i]['net_order_qty_diff'].iloc[0])
    net_order_amt_sum.append(df[df['scd'] == i]['net_order_amt'].sum())
    net_order_amt_mean.append(df[df['scd'] == i]['net_order_amt'].mean())
    net_order_amt_diff.append(df[df['scd'] == i]['net_order_amt_diff'].iloc[0])
    diff_meanPriceMean.append(df[df['scd'] == i]['diff_meanPrice'].mean())
    diff_meanPriceSum.append(df[df['scd'] == i]['diff_meanPrice'].sum())
    weekday.append(df[df['scd'] == i]['weekday'].iloc[0])
```

예측 모델링

3.1 모델링 과정

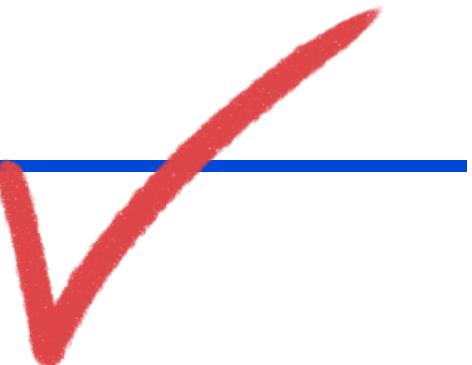
3.2 모델링 결과

03

모델링

$$F1\ Score = 2 \times \frac{recall \times precision}{recall + precision}$$

모델링 평가 기준 : F1-score



캐글 대회 1등 모델

XGBoost

학습에 오랜 시간
하이퍼파라미터 수정이 관건

정형데이터에서의 딥러닝

TabNET

정형데이터에 특화되었다는 딥러닝 모델
하지만 우리 데이터에서는 낮은 결과

GBM의 문제점 개선

CatBoost

기존 GBM의 과적합과 학습속도를 개선한 모델
Symmetric tree 구조
결측치가 많은 sparse한 데이터에는 부적합

Threshold 지정

지정 전보다 0.1-0.2 f1-score 상승

모델링

$$F1 Score = 2 \times \frac{recall \times precision}{recall + precision}$$

모델링 평가 기준 : F1-score

임직원 모델링 결과

CatBoost 모델 사용

- Threshold 지정 전 : 0.7734
- Threshold 지정 후 : 0.78

비임직원 모델링 결과

CatBoost 모델 사용

- Threshold 지정 전 : 0.65
- Threshold 지정 후 : 0.70

평가 방식

- 교차검증 사용, 5-Fold 사용
- Aggregation 되어 있는 데이터에 예측을 진행하므로, 이를 다시 원 데이터프레임에서의 결과를 확인해야 함
 - 정확한 f1-score를 내는 함수를 따로 만들어 평가를 진행

모델링 결과

4.1 F1-score

4.2 보완할 점

04

최종 F1 Score

BDA 공모전 측에서 확인한 F1 Score

쿠엔져스 팀의 F1-Score는 0.7321 입니다.

BDA 공모전 F1 Score 분포 상황

최고점 : 0.8158

최저점 : 0.4007

평균 : 0.6316

중앙값 : 0.6613

나쁘지 않은 F1 Score!

이제 발표심사만 잘 준비하면 된다!

BDA 본선 일정

본선 결과물 제출

본선 결과물 제출

1. 작성코드 (.ipynb 최종본 제출)
2. test데이터 (.csv 파일)
3. 발표용 PPT 자료 : (.pdf 파일, 30p 이내)

6/1(목) 23:59

본선 발표심사

결과물 발표 및 심사

CJ제일제당 본사에서 발표 진행

6/4(일)

컨퍼런스

연합 컨퍼런스 및 시상식

BDA 공모전에 참여하는 데이터 분야 학회 및 동아리의 연합 학술 교류를 위한 컨퍼런스
시상식과 뒷풀이가 있을 예정으로 네트워킹 가능

6/24(토)

보완할 점

부족한 시간으로 다음과 같은 아쉬운 점, 보완할 점이 존재.
더욱 보완하여 내일까지 최종 결과물 제출

인풋 데이터에 대한 적정성 고려

트리기반 모델의 결과에 Feature Importance가 높은 칼럼이 상관분석을 해보면 낮게 나오거나, 그 반대의 경우가 발생. 인풋 데이터에 대한 고려를 조금 더 해서 모델링을 다시 진행후, 성능 비교

모델링의 아쉬움.. 여러 모델로 모델링 진행

시간상의 문제로 CatBoost밖에 진행하지 못한 점이 아쉬움. 다른 모델을 이용해서 모델링을 진행후, 성능 비교

고객 세그멘테이션 분석

F1-score을 위해 모델링에만 치중한 경향이 있음. 단순히 예측 모델링을 어떻게 만들었는지 뿐만 아니라 고객을 세분화하는 과정을 필요로 하므로, 남은 시간동안 이 부분에 집중

THANK YOU

쿠벤저스

KHUDA 심화트랙 컨퍼런스