Today’s

Team Project By PinaColada

Table of Contents

Index

001 Introduction

002 Data

003 Web

004 Conclusion

005 QnA

Part 1
Introduction

Introduction

프로젝트 선정 조건

데이터의 양?

학습한 기술스택 구현?

생활과 밀접한가?

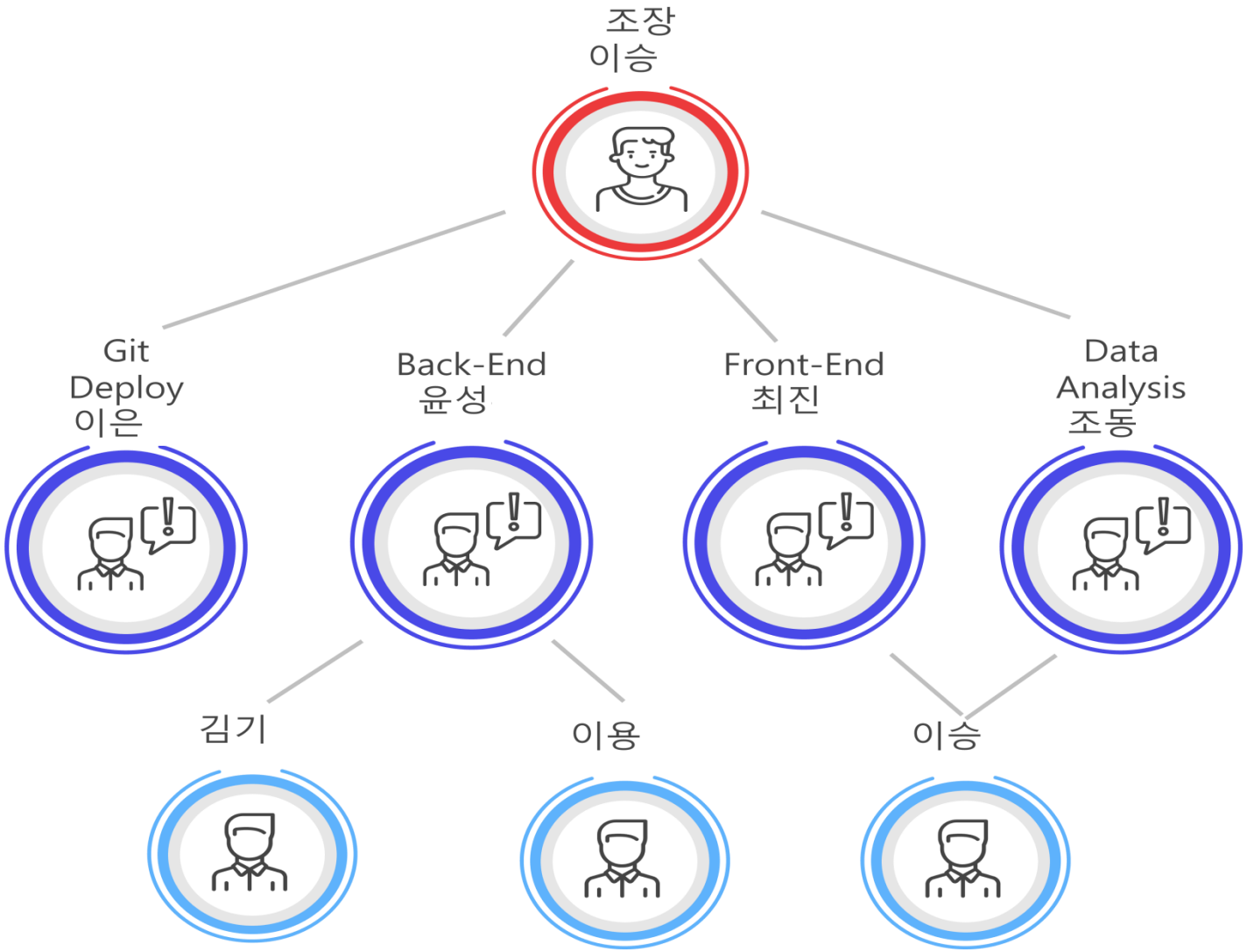
Introduction

서비스 소개

네이버, 다음, 카카오 등의 소셜 페이지에서의 리뷰 Crawling, 분석
기상청 날씨 데이터 API를 활용하여
사용자가 검색하고자 하는 지역의 카페들을 추천 받는 서비스

Introduction

역할 및 담당



Introduction

역할 및 담당

이름	세부 역할
이승	프로젝트 일정관리, 데이터 수집(네이버크롤링), UI 디자인
조동	데이터수집, 데이터 가공, 데이터 모델링
이은	프로젝트 조율, 서비스화, 백엔드 (DB설계, 로그인/아웃)
김기	백앤드 (예외처리, 1차 프로젝트 보안)
윤성	백엔드 총괄(DB설계, 비즈니스 로직 총괄)
이용	백앤드 (DB설계, 북마크, 마이페이지)
최진	UI 디자인 및 프론트 총괄(Layout, Design, Logo)

Introduction

SKILL STACK

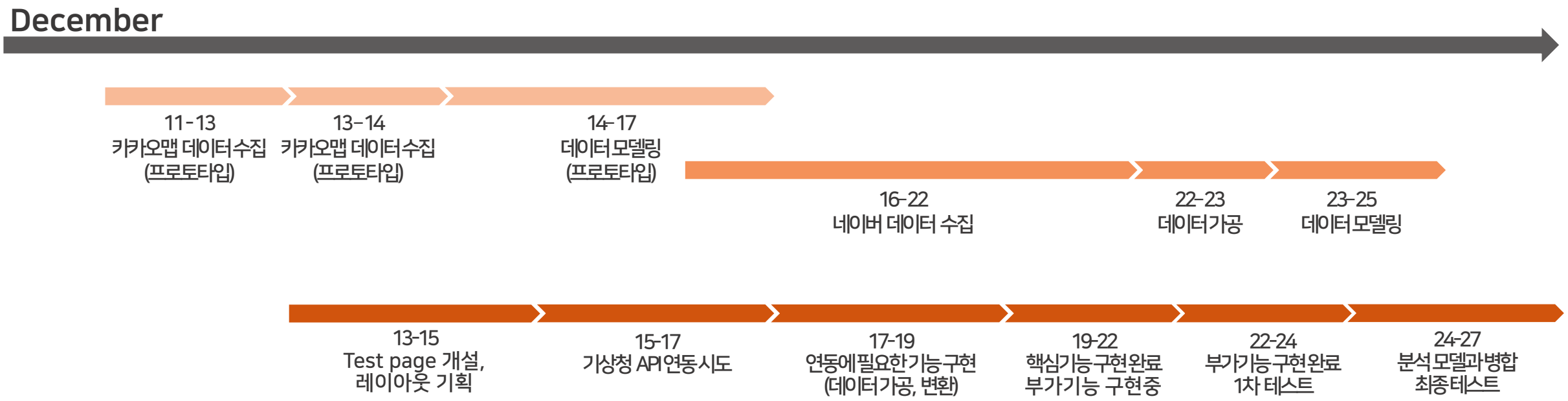
Language Framework

Data Analysis

Database Manage Tool

Introduction

프로젝트 진행 일정



Part 2
Data

1) 네이버맵 카페 데이터 크롤링

동적 크롤링(Selenium+BeautifulSoup)

Selenium

+

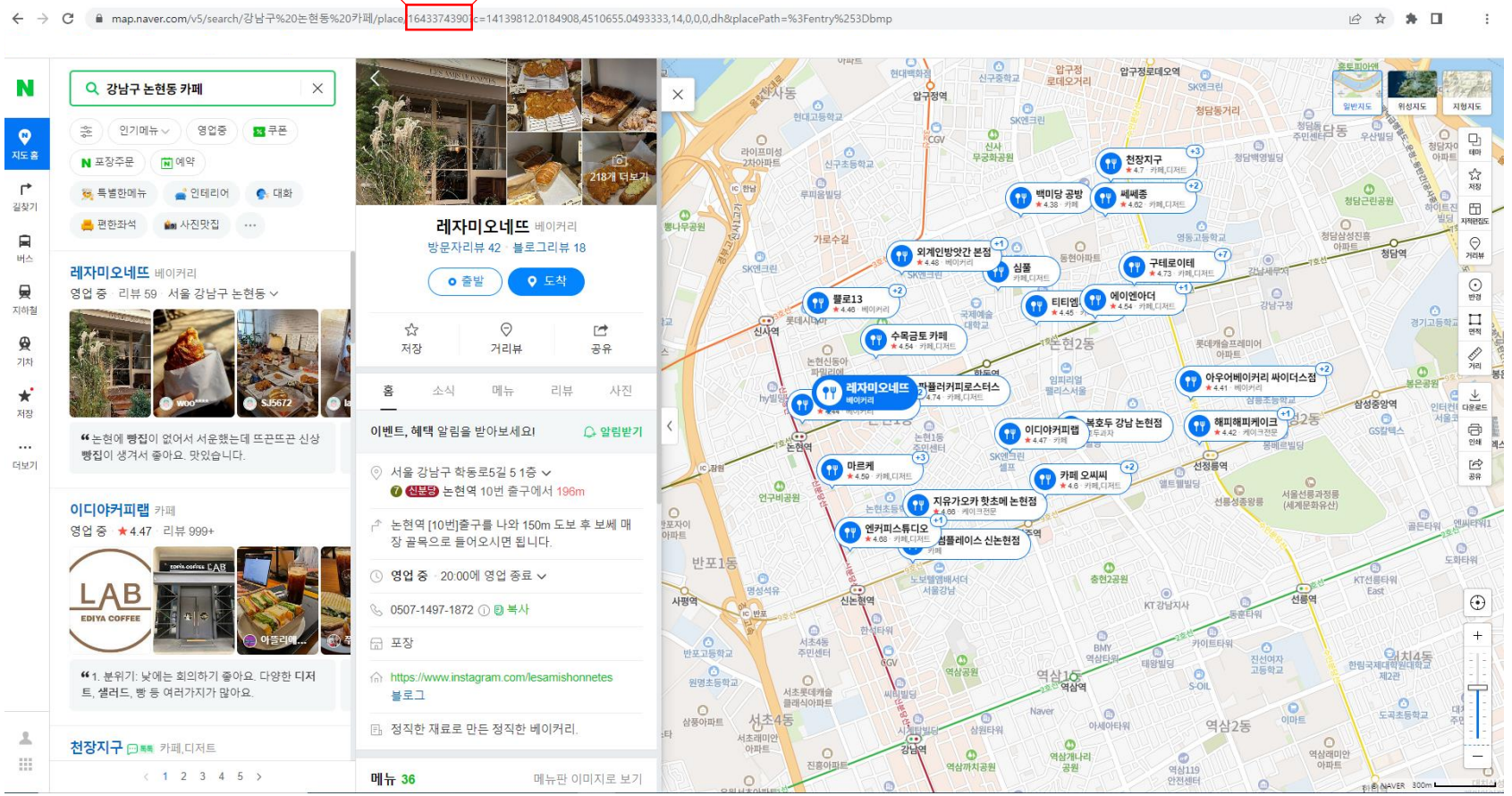
BeautifulSoup

1) 네이버맵 카페 데이터 크롤링

1차 크롤링 방식

1. 네이버맵 URL 주소 GET	2. 고유 넘버 추출 후, 넘버링 매핑	3. 네이버 플레이스 URL 주소 GET

1643374390 → 넘버링 매핑 → pcmap.place.naver.com/restaurant/1643374390/home



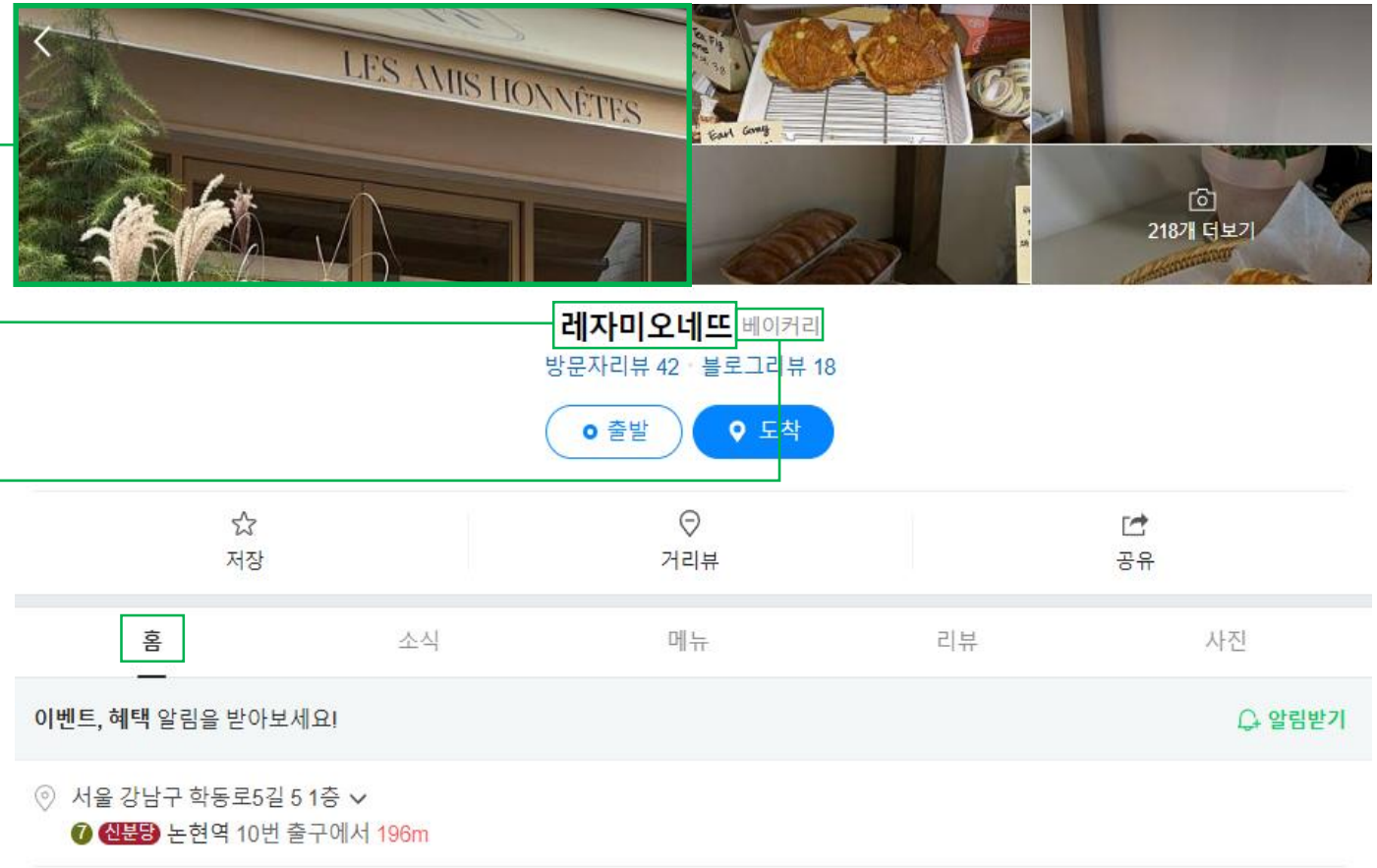
1) 네이버맵 카페 데이터 크롤링

2차 크롤링 방식

1. 네이버 플레이스 '홈' 탭 스크래핑

- BeautifulSoup 모듈을 이용하여 카페 정보 Scarping -

수집 방법	수집 데이터 소개
스크래핑	카페 대문 이미지 주소
	카페명
	카페 카테고리
검색 키워드	카페 주소(자치구, 행정동) Ex) 강남구 역삼동 카페



1) 네이버맵 카페 데이터 크롤링

2차 크롤링 방식

2. 네이버 플레이스 '리뷰' 탭 동적 크롤링

- Selenium + BeautifulSoup 모듈을 이용하여 개별 리뷰 데이터 Crawling -

수집 방법	수집 데이터 소개
동적 크롤링	개별 리뷰
	개별 리뷰 게시일

1) 네이버맵 카페 데이터 크롤링

데이터 크롤링 결과

[데이터프레임 정보]	
칼럼명	데이터 타입
카페명	object
자치구	
행정동	
카테고리	
이미지 주소	
리뷰 날짜	
리뷰	
총 개수 : 565,329 개	

	카페명	자치구	행정동	카테고리	이미지 주소	리뷰 날짜	리뷰
0	블랑커피	강남구	개포동	카페,디저트	https://search.pstatic.net/common/?autoRotate=...	2022-11-24	개포동에 세련된 카페 발견하고 홀린듯이 들어와봤는데요\n여기 스콘 맛집에 커피 맛집...
1	블랑커피	강남구	개포동	카페,디저트	https://search.pstatic.net/common/?autoRotate=...	2022-12-04	분위기 너무좋아서 지나가다온곳.개포동에 이런곳이 있었다니~ 커피원두향 너무좋고\n생...
2	블랑커피	강남구	개포동	카페,디저트	https://search.pstatic.net/common/?autoRotate=...	2022-11-24	맛있는 스콘 냄새가 넘넘 좋은 카페🌹 특히 플라워스콘! 모양도 너무 이쁜데 위에 ...
3	블랑커피	강남구	개포동	카페,디저트	https://search.pstatic.net/common/?autoRotate=...	2022-11-28	매장이 진짜 청결한데 이뻐요!!!스콘도 안은 촉촉하고 달달하면서 너무 맛있습니다!베...
4	블랑커피	강남구	개포동	카페,디저트	https://search.pstatic.net/common/?autoRotate=...	2022-11-24	디저트가 종류별로 다 맛있어요~ 런던스콘 정말 맛있고요, 선물용으로도 좋아요!
...	...	...	...	...	...	...	...
565324	파리바게뜨 묵동삼거리점	중랑구	중화동	베이커리	https://search.pstatic.net/common/?autoRotate=...	2022-12-03	꿀토스트 진짜 너무 맛있어요~~
565325	파리바게뜨 묵동삼거리점	중랑구	중화동	베이커리	https://search.pstatic.net/common/?autoRotate=...	2022-12-03	호밀호두식빵만 매주 구입합니다.
565326	파리바게뜨 묵동삼거리점	중랑구	중화동	베이커리	https://search.pstatic.net/common/?autoRotate=...	2022-11-12	빠바가 spc인 걸 순간 잊고 방문했네요 \n하지만 매장 직원분은 친절하고 좋았습니...
565327	파리바게뜨 묵동삼거리점	중랑구	중화동	베이커리	https://search.pstatic.net/common/?autoRotate=...	2022-11-19	호밀호두식빵 매주 구입해요~
565328	파리바게뜨 묵동삼거리점	중랑구	중화동	베이커리	https://search.pstatic.net/common/?autoRotate=...	2022-12-03	친절 빵 맛있고 종류 많음\n오랜만에 샌드위치랑 쿠키 사먹었는데 너무 맛있었습니다\...

565329 rows × 7 columns

2) 데이터 결측치, 중복값 제거

결측치 제거

1. 데이터프레임 결측치 확인

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

```
카페명          0
자치구          0
행정동          0
카테고리         0
이미지 주소    14048
리뷰 날짜        0
리뷰             4
dtype: int64
```

→ 이미지 주소 칼럼 : 14,048개

→ 리뷰 칼럼 : 4개

2. 데이터프레임 결측치 제거

```
df = df.dropna(subset=['이미지 주소', '리뷰'])
df = df.reset_index()
df.drop(['index'], axis=1, inplace=True)
print(df.isnull().sum(), end='₩n₩n')
print('결측치 제거 후, 데이터 개수 :', len(df))
```

```
카페명        0
자치구        0
행정동        0
카테고리       0
이미지 주소    0
리뷰 날짜      0
리뷰           0
dtype: int64
```

→ 이미지 주소, 리뷰 칼럼 결측값 제거 완료

3. 결측값 제거 후, 데이터 개수 확인

```
결측치 제거 후, 데이터 개수 : 551277
```

중복값 제거

1. 중복값 제거

```
print('중복 제거 전, 데이터 개수 : ', len(df))
df.drop_duplicates(keep='first', ignore_index=True, inplace=True)
print('중복 제거 후, 데이터 개수 : ', len(df))
```

```
중복 제거 전, 데이터 개수 :  551277
중복 제거 후, 데이터 개수 :  549323
```

→ drop_duplicates() 메소드를 사용하여 전체 칼럼 기준으로 중복된 값을 제거

3) 카페별 리뷰 개수 산포도

카페별 리뷰 개수 확인

1. 카페별 리뷰 개수

- value_counts() 메소드 사용 -

```
우지커피          277
카페인           251
경성꽈배기         240
바바플           229
사생활           220
              ...
써니라이브           1
초콜렛 7080       1
감나무집 카페          1
어너더빈스            1
꽃보다츄러스           1
Name: 카페명, Length: 25245, dtype: int64
```

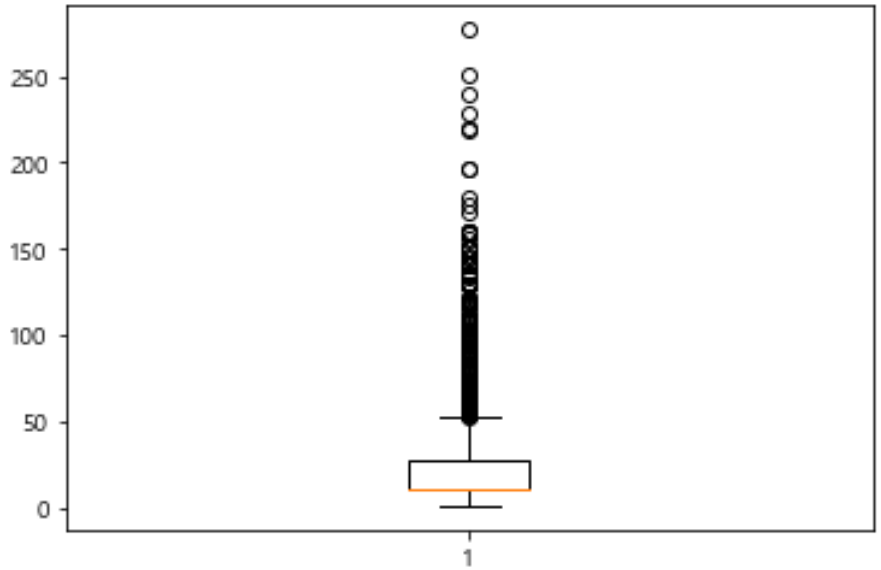
→ 카페별 리뷰 개수 편차가 큼

→ 서울 지역 기준, 총 25,245개의 카페 존재

3) 카페별 리뷰 개수 산포도

카페별 리뷰 개수 산포도 그래프

2. 카페별 리뷰 개수 Boxplot 그래프



1) 제 4분위수에서 벗어난 카페 데이터가 존재

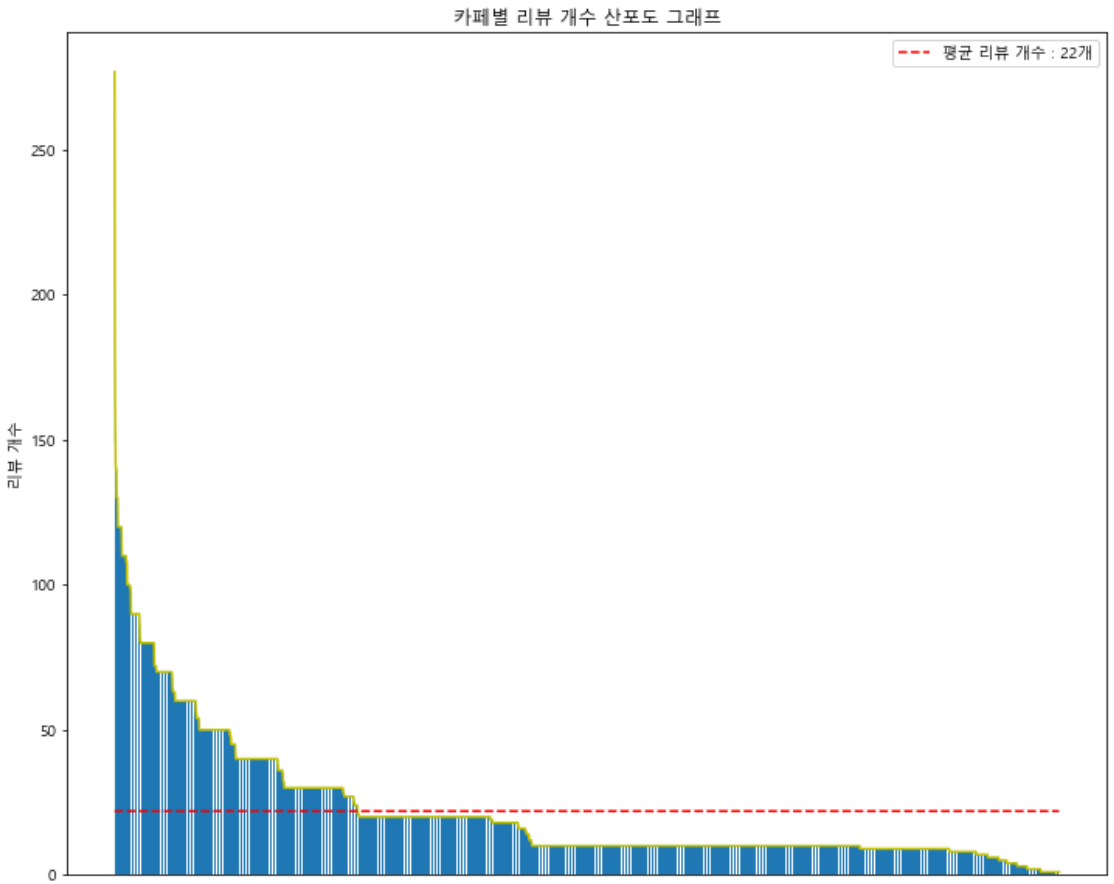
2) 최소, 최대 데이터의 편차가 매우 큼

→ Plot 그래프로 더 자세히 확인

3) 카페별 리뷰 개수 산포도

카페별 리뷰 개수 산포도 그래프

3. 카페별 리뷰 개수 Plot + Bar 그래프



1) 최소 개수와 최대 개수 편차 크기를 잘 나타냄

2) 카페별 평균 리뷰 개수는 20~22개

[리뷰 개수가 많은 카페 특징]

- 인기 많은 카페일 확률이 높음
- 광고성 리뷰가 달려있을 확률이 높음

[리뷰 개수가 적은 카페 특징]

- 형태소 분석을 통해 특징 추출의 어려움 발생
- 하지만 개별 리뷰 길이가 길수록 유의미한 추출 가능

[결론]

→ 리뷰 개수 많은 카페 : 광고성 리뷰 패턴 파악 후, 광고성 리뷰 제거

→ 리뷰 개수 적은 카페 : 개별 리뷰 길이가 짧고 총 리뷰 길이가 기준 미만인 카페 제거

4) 리뷰 개수 기반 데이터 전처리

리뷰 개수가 많은 카페 전처리

1. 광고성 리뷰 패턴 분석 결과

[패턴 분석 결과]

1) 카페별 평균 리뷰 개수(20~22개) 이상인 카페에 대해 전처리 적용

2) 카페 오픈일 당일에 달린 리뷰에서 대략 80%는 광고, 20%는 일반 리뷰라고 가정

3) 광고성 리뷰의 100%를 제거하기 보단, 일부 제거를 통한 데이터 신뢰성 증진 목적

4) 카페별 가장 빠른 일자에 해당하는 리뷰들을 전부 제거

4) 리뷰 개수 기반 데이터 전처리

리뷰 개수가 많은 카페 전처리

2. 평균 리뷰 개수를 초과하는 카페 비율 확인

```
many_review_cafes = list(df['카페명'].value_counts()[df['카페명'].value_counts() > 20].index)
print('리뷰 개수가 20개를 초과하는 카페 비율 :', np.round(len(many_review_cafes)/len(df['카페명'].unique()), 2))
```

```
리뷰 개수가 20개를 초과하는 카페 비율 : 0.26
```

→ 평균 리뷰 개수(20개)를 초과하는 카페 비율은 26%이다.

→ 그렇게 많은 비율을 차지하지 않는다.

4) 리뷰 개수 기반 데이터 전처리

리뷰 개수가 많은 카페 전처리

3. 리뷰 개수가 20개를 초과하는 카페 전처리

```
dataframe = pd.DataFrame(columns=df.columns)

print('기존 데이터프레임 개수 : ', len(df))
for cafe in df['카페명'].unique():
    if df[df['카페명'] == f'{cafe}']['카페명'].value_counts()[0] > 20:
        date_list = df[df['카페명'] == cafe]['리뷰 날짜'].unique()
        date_list.sort()
        date = date_list[0]
        test_df = df[df['카페명'] == cafe]
        test_df = test_df[test_df['리뷰 날짜'] != date]
        dataframe = pd.concat([dataframe, test_df], axis=0)
        dataframe.reset_index(inplace=True)
        dataframe.drop(['index'], axis=1, inplace=True)
    elif df[df['카페명'] == f'{cafe}']['카페명'].value_counts()[0] <= 20:
        test_df = df[df['카페명'] == cafe]
        dataframe = pd.concat([dataframe, test_df], axis=0)

dataframe.reset_index(inplace=True)
dataframe.drop(['index'], axis=1, inplace=True)
print('마지막 일자 데이터 제거 후, 데이터프레임 개수 : ', len(dataframe))
```

```
기존 데이터프레임 개수 :  549323
마지막 일자 데이터 제거 후, 데이터프레임 개수 :  514678
```

[전처리 알고리즘 설명]

1) 카페 별로 리뷰 개수가 20개를 초과하는 카페의 전체 리뷰 날짜 GET

2) 가장 빠른 리뷰 날짜의 리뷰 이외의 데이터를 GET

3) 새로운 데이터프레임에 병합

[데이터프레임 개수 변화]

- 기존 데이터 개수 : 549,323개

↓

- 제거 후, 데이터 개수 : 514,678개

(약 35,000개 row 삭제)

4) 리뷰 개수 기반 데이터 전처리

리뷰 개수가 적은 카페 전처리

1. 평균 리뷰 개수 미만의 카페 비율 확인

```
min_review_cafes = df['카페명'].value_counts()[df['카페명'].value_counts() < 10].index
print('리뷰 개수가 10개 미만인 카페 비율 : ', np.round(len(min_review_cafes)/len(df['카페명'].unique()), 2))
```

```
리뷰 개수가 10개 미만인 카페 비율 :  0.21
```

→ 평균 리뷰 개수(20개) 미만의 개수를 갖는 카페 비율은 21%이다.

→ 그렇게 많은 비율을 차지하지 않는다.

4) 리뷰 개수 기반 데이터 전처리

리뷰 개수가 적은 카페 전처리

2. 카페별 총 리뷰 길이가 350개 미만인 카페 제거

```
unuse_min_cafes = []

for cafe in min_review_cafes:
    if df[df['카페명'] == cafe]['카페명'].value_counts()[0] < 10:
        review_len = 0
        for review in df[df['카페명'] == cafe]['리뷰']:
            review_len += len(review)
        if review_len < 350:
            unuse_min_cafes.append(cafe)
```

```
print('카페 삭제 전, 데이터 개수 :', len(df))
df = df[df['카페명'].isin(unuse_min_cafes) == False]
print('카페 삭제 후, 데이터 개수 :', len(df))
```

```
카페 삭제 전, 데이터 개수 : 514678
카페 삭제 후, 데이터 개수 : 486851
```

[전처리 알고리즘 설명]

1) 리뷰 개수가 10개 미만인 카페들을 대상으로 잡는다.

2) 카페별 전체 리뷰 길이를 구한다.

3) 전체 리뷰 길이가 350 미만이면 제거한다.

[데이터프레임 개수 변화]

- 기존 데이터 개수 : 514,678 개

↓

- 제거 후, 데이터 개수 : 486,851개

(약 28,000개 row 삭제)

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

데이터 전처리 절차 및 방식

1. 절차 및 방식 소개

[절차 및 방식]

1) 각 카페의 리뷰들을 카페 별 하나의 문자열로 병합 (총 20,824개 카페 사용)

2) 카페 별 리뷰에 TF-IDF를 사용하여 상위 10개 단어 추출

3) Word2Vec을 이용해 상위 10개 단어들에 임베딩 수행

4) 각 단어마다 1차원 값으로 축소

5) 카페 별 상위 10개 단어들 PCA를 이용해 2차원으로 축소

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 병합 및 형태소 분석

1. 형태소 분석을 이용해 형용사 추출 후, 카페 별 리뷰 병합

	카페명	자치구	행정동	카테고리	이미지 주소	리뷰 날짜	리뷰
0	블랑커피	NaN	NaN	NaN	NaN	NaN	세련된 입니다 있어서 좋았어요 맛있어요 맛있어서 친절하시고 자세하게 맛있는 있었습니...
1	Glyn.	NaN	NaN	NaN	NaN	NaN	남다른 친절하셔서 즐겁게 있어서 좋아요 예쁘네요 편안한 맛있는 좋아요 맛있고 조용해...
2	PEEK	NaN	NaN	NaN	NaN	NaN	좋습니다 멋진 맛있었습니다 맛있고 이쁘네요 좋음 비싸지만 괜찮은 좋은데 있어 많아도...
3	연리담	NaN	NaN	NaN	NaN	NaN	고소하고 맛있어요 부드럽고 좋고 같아요 같은 많아서 같아요 더라고요 예뻐서 친절해요...
4	억셉트커피 양재점	NaN	NaN	NaN	NaN	NaN	고소한 괜찮았어요 그랬고 포근한 있고 넓어요 있어서 같아 좋아요 있어서 맛있음 맛있...
...	...	...	...	...	...	...	...
20819	카페111 본점	NaN	NaN	NaN	NaN	NaN	편안한 좋았어요 좋습니다 좋아요 맛있네요 맛있어요 친절합니다 좋 아늑하고 좋네요 야...
20820	픽미픽미아이스 중화점	NaN	NaN	NaN	NaN	NaN	많아서 좋아요 있어서 맛있오용 좋아요 친절하고 깔끔하고 좋아요 많아용 맛있는게 많아...
20821	뚝방다방	NaN	NaN	NaN	NaN	NaN	괜찮고 친절하시고 무섭게 그래요 시원한 많아 시원하게 유일하게 있는 맛있어요 좋은 ...
20822	카페줌마	NaN	NaN	NaN	NaN	NaN	맛있어요 좋아요 친절하시고 좋아요 굳 좋아요 좋아요 맛있어요
20823	포트오브모카 중화점	NaN	NaN	NaN	NaN	NaN	좋아요 친절하셔서 좋아요 맛있어요 좋아요 좋아요 친절하시네요 조용히 좋았음

20824 rows × 7 columns

→ 카페 별 리뷰들에 형태소 분석을 수행한 후, 형용사만 추출하여 전체 병합시킴

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 병합 및 형태소 분석

2. 추출한 형용사 문자열에서 빈 문자열 row 제거

```
unuse_idx = cafe_review[cafe_review['리뷰'] == ''].index[0]
print('리뷰 내용이 없는 row 인덱스 :', unuse_idx)
```

```
리뷰 내용이 없는 row 인덱스 : 81
```

```
print('제거 전, 데이터 개수 :', len(cafe_review))
cafe_review.drop([unuse_idx], axis=0, inplace=True)
print('제거 후, 데이터 개수 :', len(cafe_review))
```

```
제거 전, 데이터 개수 : 20824
제거 후, 데이터 개수 : 20823
```

→ 81번 인덱스는 형태소 추출 및 문자열 병합 후 빈 문자열이 추출됨

→ drop() 메소드를 이용해 row 제거

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 병합 및 형태소 분석

3. 결측치 제거

```
print(cafe_review.isnull().sum())
```

```
Unnamed: 0        0
카페명               0
자치구           20823
행정동           20823
카테고리          20823
이미지 주소        20823
리뷰 날짜         20823
리뷰               12
dtype: int64
```

```
cafe_review.dropna(subset=['리뷰'], axis=0, inplace=True)
```

```
print(cafe_review.isnull().sum())
```

```
Unnamed: 0        0
카페명               0
자치구           20811
행정동           20811
카테고리          20811
이미지 주소        20811
리뷰 날짜         20811
리뷰                0
dtype: int64
```

[결측치 확인 및 제거]

1) 리뷰 칼럼에 12개 결측치 확인

2) dropna() 메소드를 사용해 결측치 제거

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 TF-IDF 수행

1. TF-IDF 수행 결과

	카페명	상위 리뷰
0	블랑커피	[고소해서, 청결한데, 좋구, 배불러도, 자세하게, 이쁜데, 세련된, 있었다니, 아...
1	Glyn.	[조용해수, 평범하네요, 남다른, 즐겁게, 예쁘네요, 솔직히, 좋아요, 맛있는, 없...
2	PEEK	[좋음, 있다, 멋져요, 맛있었습니다, 싱거운, 다양한, 많아도, 있어, 기대했는데...
3	연리담	[더라고요, 맛있더라구요, 같아요, 예뻐서, 고소하고, 부드럽고, 같은, 많아서, ...
4	억셉트커피 양재점	[억셉, 그랬고, 맛있음, 포근한, 시원하고, 괜찮았어요, 같아, 넓어요, 있어서,...
...	...	...
20806	카페111 본점	[야할, 친절합니다, 편안한, 아늑하고, 맛있어요, 같은, 좋습니다, 좋네요, 맛있...
20807	픽미픽미아이스 중화점	[맛있오용, 그러네요, 많아용, 맛있는게, 오래된, 좋아요, 많아요, 많아서, 깔끔...
20808	뚝방다방	[무섭게, 유일하게, 그래요, 가깝고, 시원하게, 시원한, 괜찮고, 많아, 좋아요,...
20809	카페줌마	[좋아요, 맛있어요, 친절하시고]
20810	포트오브모카 중화점	[좋았음, 친절하시네요, 좋아요, 조용히, 친절하셔서, 맛있어요]

20811 rows × 2 columns

→ 각 카페를 대표할 수 있는 상위 리뷰 10개 추출

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 TF-IDF 수행

2. 상위 단어 개수 10개 미만인 카페 추출

```
unuse_idx = list(rank_df['상위 리뷰'][rank_df['상위 리뷰'].apply(lambda x: len(x)) < 10].index)
print('최상위 단어 개수 10 미만의 카페 개수 :', len(unuse_idx))
```

```
최상위 단어 개수 10 미만의 카페 개수 : 3196
```

→ 총 20,811개 카페 데이터 중에서

상위 단어 개수 10개 미만인 카페 데이터 3,196개 추출

5)카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 리뷰 TF-IDF 수행

3. 상위 단어 개수 10개 미만인 카페 제거

```
print('제거 전, 데이터 개수 : ', len(rank_df))
rank_df.drop(unuse_idx, axis=0, inplace=True)
print('제거 후, 데이터 개수 : ', len(rank_df))
```

```
제거 전, 데이터 개수 :  20811
제거 후, 데이터 개수 :  17615
```

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 별 상위 리뷰 단어에 임베딩 수행

1. 임베딩 수행

	카페명	상위 리뷰
0	블랑커피	[[0.21636295, 0.09980916, 0.07972772, 0.370629...
1	Glyn.	[[-0.016417548, 0.027127746, 0.030885674, 0.02...
2	PEEK	[[0.19458222, -0.27224186, 1.4078854, 0.908932...
3	연리담	[[0.1453362, 0.13460557, -0.10296305, 0.175386...
4	억셉트커피 양재점	[[0.050961845, -0.030419229, 0.004930825, 0.02...
...	...	...
20801	트리플에이커피 중화역점	[[0.026063096, -0.0016550375, 0.0020405357, 0....
20803	컴포즈커피 중화장미점	[[-0.028127631, 0.045010634, 0.010464004, -0.0...
20806	카페111 본점	[[0.14545651, 0.17786981, -0.111999184, 0.3137...
20807	픽미픽미아이스 중화점	[[-0.01575429, -0.014423155, -0.016126608, 0.0...
20808	뚝방다방	[[0.006796795, 0.07928051, -0.0052865404, 0.07...

17615 rows × 2 columns

→ 상위 10개 단어에서 각 단어는 10차원의 float 데이터로 임베딩 된다.

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 별 상위 리뷰 단어에 임베딩 수행

2. 임베딩 벡터 축소 후, 각 단어를 칼럼 별로 나누기

1 순위 키워드 개수 : 17615
2 순위 키워드 개수 : 17615
3 순위 키워드 개수 : 17615
4 순위 키워드 개수 : 17615
5 순위 키워드 개수 : 17615
6 순위 키워드 개수 : 17615
7 순위 키워드 개수 : 17615
8 순위 키워드 개수 : 17615
9 순위 키워드 개수 : 17615
10 순위 키워드 개수 : 17615

	카페명	상위 리뷰	1 순위 키워드	2 순위 키워드	3 순위 키워드	4 순위 키워드	5 순위 키워드	6 순위 키워드	7 순위 키워드	8 순위 키워드	9 순위 키워드	10 순위 키워드
0	블랑커피	[[0.21636295, 0.09980916, 0.07972772, 0.370629...	2.282888	0.050866	3.563681	0.650245	1.040168	1.194994	1.129744	1.937320	1.702361	1.737271
1	Glyn.	[[-0.016417548, 0.027127746, 0.030885674, 0.02...	0.057244	0.195919	0.638602	2.373752	2.387179	2.245899	6.504819	5.578884	3.113813	4.304359
2	PEEK	[[0.19458222, -0.27224186, 1.4078854, 0.908932...	5.192621	3.978342	2.351507	4.380652	0.511575	4.846098	1.183328	5.043743	1.673676	1.890040
3	연리담	[[0.1453362, 0.13460557, -0.10296305, 0.175386...	1.004142	3.109971	5.998544	3.995976	4.721123	5.049824	4.793315	5.087853	5.875030	6.504819
4	억셉트커피 양재점	[[0.050961845, -0.030419229, 0.004930825, 0.02...	0.086947	1.030250	5.044117	1.907249	3.395527	3.122540	3.019410	3.704708	5.721091	4.700218

[전처리 결과]

1) 단어 별로 구분한 후, 키워드 개수 파악

2) 10개의 단어 키워드 개수가 일치하는지 확인

3) 상위 단어 순으로 총 10개의 칼럼으로 분리

5) 카페 분위기 추출 모델(1차 모델) 데이터 전처리

카페 별 임베딩된 상위 단어에 PCA 축소

1. PCA 분석

```
# PCA 객체 선언
pca = PCA(n_components=2)

# PCA를 통한 10차원 데이터 2차원으로 차원 축소
encoded_x = pca.fit_transform(x)
print(encoded_x)
```

```
[[-3.99288658  1.66913246]
 [ 1.67060446 -2.31824193]
 [-0.10291916  5.27847484]
 ...
 [ 7.65277879  2.06269135]
 [ 3.90378804 -2.35897163]
 [ 2.806547   -1.44199069]]
```

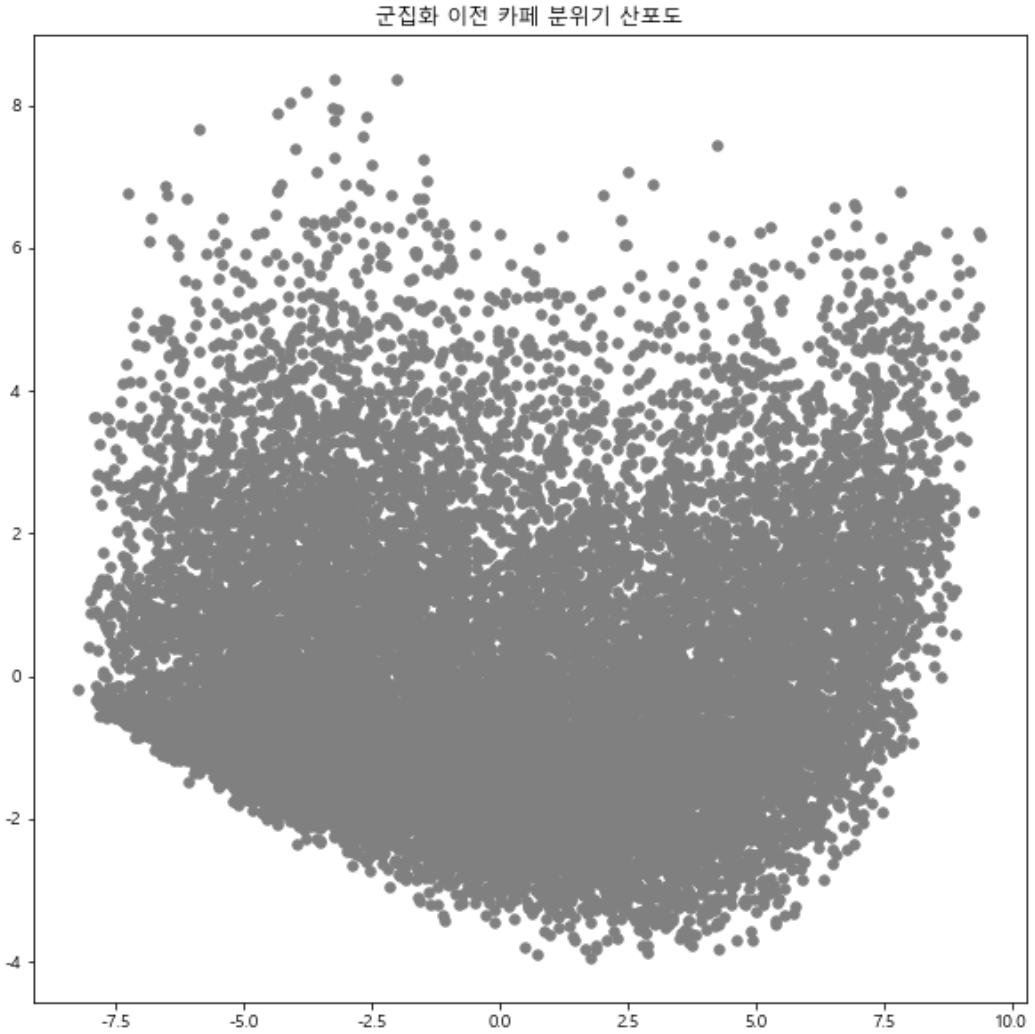
[PCA 축소 결과]

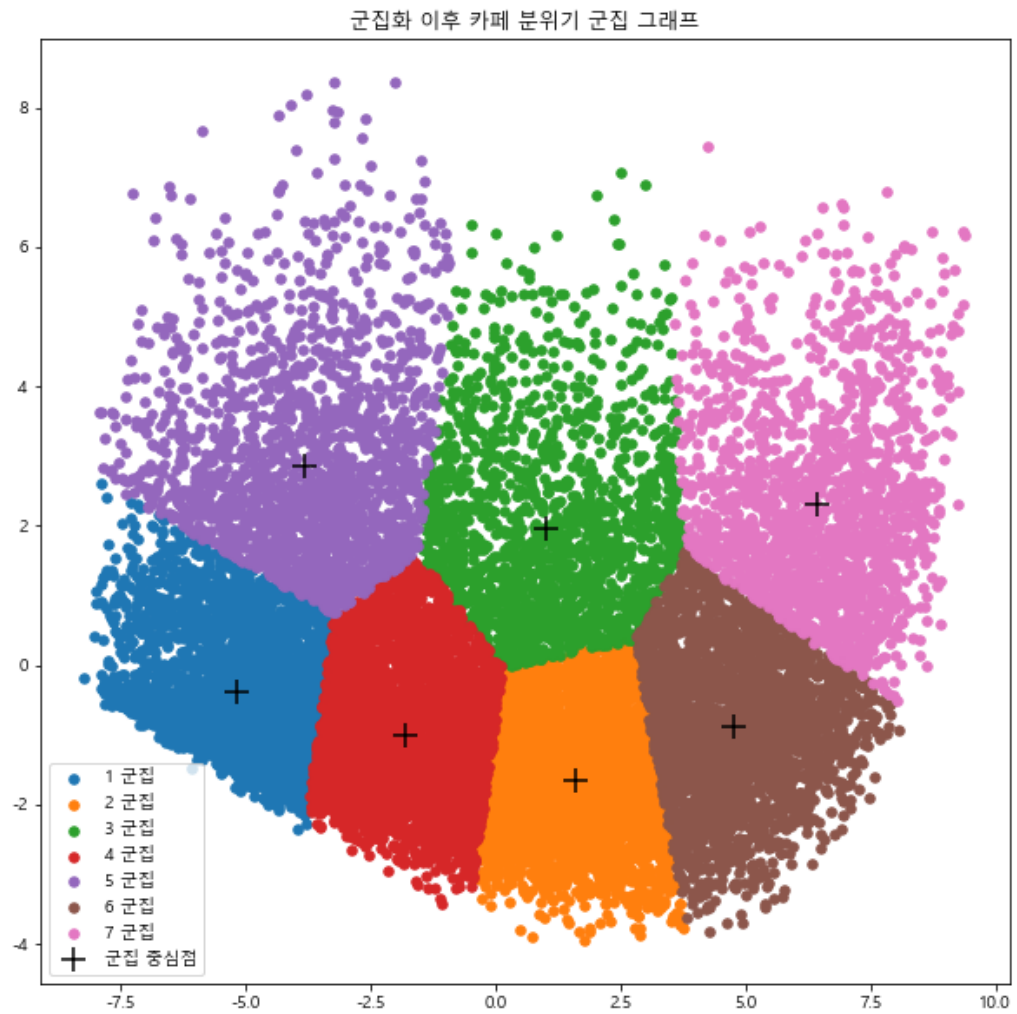
- 10차원 벡터를 2차원 벡터로 차원 축소 진행

6) 카페 분위기 추출 클러스터링

카페 별 리뷰 데이터로 클러스터링 수행

1. 군집화 수행 전, 후 그래프 확인





6)카페 분위기 추출 클러스터링

카페 별 리뷰 데이터로 클러스터링 수행

2. 군집화 결과 레이블 매핑

	카페명	자치구	행정동	카테고리	#
0	로슈베이커리	강남구	개포동	베이커리	
1	로슈베이커리	강남구	개포동	베이커리	
2	로슈베이커리	강남구	개포동	베이커리	
3	로슈베이커리	강남구	개포동	베이커리	
4	로슈베이커리	강남구	개포동	베이커리	
...	...	...	...	...	
430461	카페111 본점	중랑구	중화동	카페,디저트	
430462	카페111 본점	중랑구	중화동	카페,디저트	
430463	카페111 본점	중랑구	중화동	카페,디저트	
430464	카페111 본점	중랑구	중화동	카페,디저트	
430465	카페111 본점	중랑구	중화동	카페,디저트	

	이미지 주소	리뷰 날짜	분위기
0	https://search.pstatic.net/common/?autoRotate=...	2022-10-30	0
1	https://search.pstatic.net/common/?autoRotate=...	2022-07-28	0
2	https://search.pstatic.net/common/?autoRotate=...	2022-09-01	0
3	https://search.pstatic.net/common/?autoRotate=...	2022-09-29	0
4	https://search.pstatic.net/common/?autoRotate=...	2022-07-31	0
...	...	...	..
430461	https://search.pstatic.net/common/?autoRotate=...	2022-10-30	6
430462	https://search.pstatic.net/common/?autoRotate=...	2022-10-01	6
430463	https://search.pstatic.net/common/?autoRotate=...	2021-12-03	6
430464	https://search.pstatic.net/common/?autoRotate=...	2022-02-28	6
430465	https://search.pstatic.net/common/?autoRotate=...	2022-08-26	6

[430466 rows x 7 columns]

7) 1차 전처리 데이터와 기상청 데이터 매핑

기상청 데이터 확인

1. 기상청 데이터 확인

	지점	지점명	일시	평균기온(°C)	일강수량(mm)	평균 풍속(m/s)	평균 상대습도(%)	ㅆ
0	108	서울	2017-01-01	2.7	NaN	1.5	75.9	
1	108	서울	2017-01-02	5.0	0.3	2.1	77.8	

	합계 일사량(MJ/m2)	일 최심적설(cm)	평균 전운량(1/10)
0	4.38	NaN	7.0
1	5.11	NaN	7.6

7) 1차 전처리 데이터와 기상청 데이터 매핑

기상청 데이터와 연결

1. 기상청 데이터와 연결

	카페명	자치구	행정동	카테고리	이미지 주소	₩
0	로슈베이커리	강남구	개포동	베이커리	https://search.pstatic.net/common/?autoRotate=...	
1	로슈베이커리	강남구	개포동	베이커리	https://search.pstatic.net/common/?autoRotate=...	

	리뷰 날짜	분위기	평균기온(° C)	일강수량(mm)	평균 풍속(m/s)	평균 상대습도(%)	합계 일사량(MJ/m2)	₩
0	2022-10-30	0	14.9	NaN	1.9	64.5	13.64	
1	2022-07-28	0	29.7	NaN	2.4	63.8	25.20	

	일 최심적설(cm)	평균 전운량(1/10)
0	NaN	1.1
1	NaN	3.5

8) 데이터 전처리

결측치 제거

1. 결측치 확인

```
카페명              0
자치구              0
행정동              0
카테고리             0
이미지 주소           0
리뷰 날짜            0
분위기              0
평균기온(° C)         2
일강수량(mm)       280726
평균 풍속(m/s)        2
평균 상대습도(%)        2
합계 일사량(MJ/m2)     359
일 최심적설(cm)      369893
평균 전운량(1/10)       2
dtype: int64
```



```
카페명              0
자치구              0
행정동              0
카테고리             0
이미지 주소           0
리뷰 날짜            0
분위기              0
평균기온(° C)         0
일강수량(mm)         0
평균 풍속(m/s)        0
평균 상대습도(%)        0
합계 일사량(MJ/m2)     0
일 최심적설(cm)        0
평균 전운량(1/10)       0
dtype: int64
```

[결측치 제거]

- 평균기온 칼럼 : 2개
- 평균 풍속 : 2개
- 평균 상대습도 : 2개
- 평균 전운량 : 2개

→ 실제 데이터가 없는 경우 이므로 결측치 제거 진행

[결측치 채우기]

- 일강수량 : 280,726개
- 합계 일사량 : 359개
- 일 최심적설 : 369,893개

→ 0 값이거나 최솟값인 데이터가 NaN 값으로 표시된 칼럼이 므로 결측치를 최솟값으로 채우기

8) 데 이 터 전 처 리

데 이 터 형 변 환

1. Object 타입 칼럼 레이블 인코딩

[레이블 인코딩]

- 카페명 칼럼 : 카페 별 unique 값을 숫자형 데이터로 변환

- 자치구 칼럼 : 자치구 별 unique 값을 숫자형 데이터로 변환

- 행정동 칼럼 : 자치구 별 행정동 unique 값을 숫자형 데이터로 변환

- 카테고리 칼럼 : 카테고리 별 unique 값을 숫자형 데이터로 변환

8) 데 이 터 전 처 리

데 이 터 형 변 환

2. 데이터 변환 결과

	카페명	자치구	행정동	카테고리	이미지 주소	₩
0	3586	0	0	16	https://search.pstatic.net/common/?autoRotate=...	
1	3586	0	0	16	https://search.pstatic.net/common/?autoRotate=...	

	분위기	평균기온(° C)	일강수량(mm)	평균 풍속(m/s)	평균 상대습도(%)	합계 일사량(MJ/m2)	일 최심적설(cm)	₩
0	0	14.9	0.0	1.9	64.5	13.64	0.0	
1	0	29.7	0.0	2.4	63.8	25.20	0.0	

	평균 전운량(1/10)
0	1.1
1	3.5

8) 데 이 터 전 처 리

데 이 터 형 변 환

3. 연속형 데이터를 범주형 데이터로 변환

- 일강수량(mm)
  → 0.0mm : 0(강수없음)
  → 0.1~1.0mm : 1(1.0mm 미만)
  → 1.0~30.0mm : 2(30.0mm 미만)
  → 30.0~50.0mm : 3(50.0mm 미만)
  → 50.0mm 이상 : 4(50.0mm 이상)

- 일 최심적설(cm)
  → 0.0cm : 0(적설없음)
  → 0.1~1.0cm : 1(1.0cm 미만)
  → 1.0~5.0cm : 2(5.0cm 미만)
  → 5.0cm 이상 : 3(5.0cm 이상)

- 평균 전운량(1/10)
  → 0.0~4.9 : 1(맑음)
  → 5.0~7.9 : 3(구름많음)
  → 8.0~10.0 : 4(흐림)

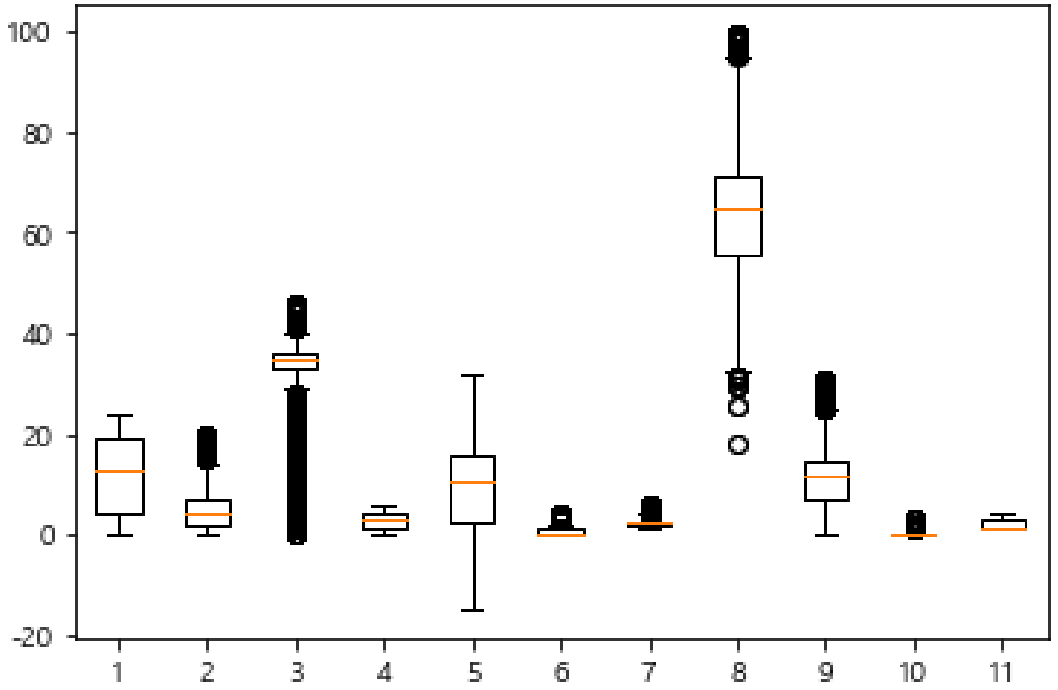
8) 데 이 터 전 처 리

이 상 치 확 인

1. 이상치 확인

[칼럼명]

- 1번 인덱스 : 자치구
- 2번 인덱스 : 행정동
- 3번 인덱스 : 카테고리
- 4번 인덱스 : 분위기
- 5번 인덱스 : 평균기온
- 6번 인덱스 : 일강수량
- 7번 인덱스 : 평균 풍속
- 8번 인덱스 : 평균 상대습도
- 9번 인덱스 : 합계 일사량
- 10번 인덱스 : 일 최심적설
- 11번 인덱스 : 평균 전운량



→ 카테고리, 평균기온 칼럼 : MinMax 정규화 수행

→ 평균 상대습도 : 100으로 나누어 정규화 수행

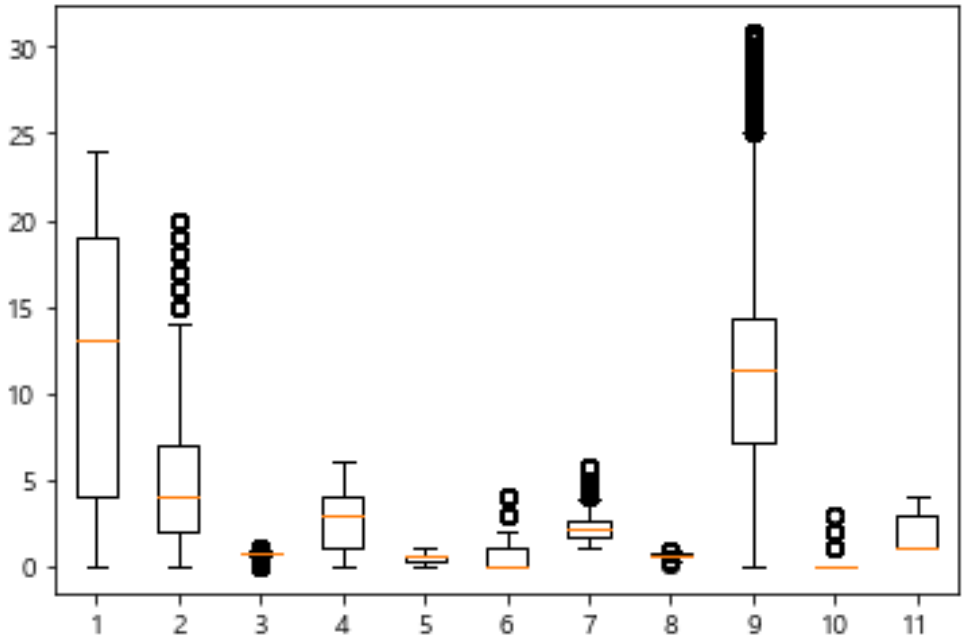
8) 데 이 터 전 처 리

이 상 치 확 인

2. 이상치 처리 결과

[칼럼명]

- 1번 인덱스 : 자치구
- 2번 인덱스 : 행정동
- 3번 인덱스 : 카테고리
- 4번 인덱스 : 분위기
- 5번 인덱스 : 평균기온
- 6번 인덱스 : 일강수량
- 7번 인덱스 : 평균 풍속
- 8번 인덱스 : 평균 상대습도
- 9번 인덱스 : 합계 일사량
- 10번 인덱스 : 일 최심적설
- 11번 인덱스 : 평균 전운량



→ 전체 칼럼의 범주를 0~40 사이의 범주로 줄임

9) 특성과 레이블 관계 분석

관계 분석 정리

1. 관계 분석 정리

- feature
  - 자치구, 행정동, 카테고리, 분위기, 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10) : 범주형 데이터(명목 척도)
  - 평균기온(°C), 평균 풍속(m/s), 평균 상대습도(%), 합계 일사량(MJ/m2) : 수치형 데이터(등간 척도)
- Label
  - 카페명 : 범주형 데이터(명목 척도)
- 범주형 feature - 범주형 label : 카이제곱 검정, Cramer's V
- 수치형 feature - 범주형 label : 로지스틱 회귀

9) 특성과 레이블 관계 분석

범주형, 범주형 관계 분석

1. 카이제곱 검정 가설 수립

- 귀무가설 : 자치구, 행정동, 카테고리, 분위기, 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)별로 카페 분위기 차이가 없다.
- 대립가설 : 자치구, 행정동, 카테고리, 분위기, 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)별로 카페 분위기 차이가 있다.

9)특성과 레이블 관계 분석

범주형, 범주형 관계 분석

2. 카이제곱 검정 수행

[자치구 - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 25
종속변수 unique value 개수 : 17613

chi2 : 7794459.744712437
p-value : 0.0

[행정동 - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 21
종속변수 unique value 개수 : 17613

chi2 : 2290915.6128467657
p-value : 0.0

[카테고리 - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 47
종속변수 unique value 개수 : 17613

chi2 : 19541294.926991746
p-value : 0.0

[분위기 - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 7
종속변수 unique value 개수 : 17613

chi2 : 2582772.0
p-value : 0.0

[일강수량(mm) - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 5
종속변수 unique value 개수 : 17613

chi2 : 210489.69698876713
p-value : 0.0

[일 최심적설(cm) - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 4
종속변수 unique value 개수 : 17613

chi2 : 205726.1139846219
p-value : 0.0

[평균 전운량(1/10) - 카페명 변수 카이제곱 수행]
독립변수 unique value 개수 : 3
종속변수 unique value 개수 : 17613

chi2 : 109661.865149727
p-value : 0.0

9) 특성과 레이블 관계 분석

범주형, 범주형 관계 분석

3. 카이제곱 검정 결과 분석

[카이제곱 검정 결과 분석]

- 전체 독립변수에 대한 종속변수 카이제곱
- p-value(0.0) < 유의수준(0.05)

- 유의확률이 유의수준 내에 존재하므로 독립변수(자치구, 행정동, 카테고리, 분위기, 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10))를 통해 종속변수(카페명)를 추출하는데 유의미한 차이를 낼 수 있다고 판단

  → 귀무가설 기각
  → 대립가설 채택

  대립가설 : 자치구, 행정동, 카테고리, 분위기, 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)별로 카페 이용 현황 차이가 있다.

→ 카이제곱 검정에서는 각 변수간의 관계만 파악할 수 있음

→ 관계의 크기를 확인하기 위해서는 상관관계 분석이 필요

→ Cramer's V 상관관계 분석 진행

9)특성과 레이블 관계 분석

범주형, 범주형 관계 분석

4. 상관관계 분석(Cramer's V) 수행

[자치구 - 카페명 변수 Cramer's V 분석]

자치구 Cramer's V 계수 : 0.8686003075107768

[행정동 - 카페명 변수 Cramer's V 분석]

행정동 Cramer's V 계수 : 0.5158484602866348

[카테고리 - 카페명 변수 Cramer's V 분석]

카테고리 Cramer's V 계수 : 0.9934141558445946

[분위기 - 카페명 변수 Cramer's V 분석]

분위기 Cramer's V 계수 : 1.0

[일강수량(mm) - 카페명 변수 Cramer's V 분석]

일강수량(mm) Cramer's V 계수 : 0.34963750802061067

[일 최심적설(cm) - 카페명 변수 Cramer's V 분석]

일 최심적설(cm) Cramer's V 계수 : 0.3991321158847885

[평균 전운량(1/10) - 카페명 변수 Cramer's V 분석]

평균 전운량(1/10) Cramer's V 계수 : 0.3568990621044261

9) 특성과 레이블 관계 분석

범주형, 범주형 관계 분석

5. 상관관계 분석(Cramer's V) 결과 분석

[Cramer's V 결과 분석]

- 전체 독립변수에 대한 종속변수 Cramer's V
  → 아주 강한 상관관계 리스트 : [자치구, 카테고리, 분위기]
  → 강한 상관관계 리스트 : [행정동]
  → 약한 상관관계 리스트 : [일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)]

9)특성과 레이블 관계 분석

수치형, 범주형 관계 분석

1. 로지스틱 회귀 분석 수행

- feature : 평균기온(°C), 평균 풍속(m/s), 평균 상대습도(%), 합계 일사량(MJ/m2)
- label : 카페명

```
==============================================================================
Dep. Variable:                  카페명   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                  0.001
Method:                 Least Squares   F-statistic:                     70.36
Date:                Sat, 24 Dec 2022   Prob (F-statistic):           1.13e-59
Time:                        00:23:39   Log-Likelihood:            -4.2848e+06
No. Observations:              430462   AIC:                         8.570e+06
Df Residuals:                  430457   BIC:                         8.570e+06
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       9014.9828     64.528    139.707      0.000    8888.511    9141.455
평균기온        -655.5398     61.730    -10.619      0.000    -776.530    -534.550
평균풍속         -21.1449     12.565     -1.683      0.092     -45.773       3.483
평균상대습도       221.9144     96.965      2.289      0.022      31.865     411.964
합계일사량         -3.0872      1.990     -1.551      0.121      -6.988       0.813
==============================================================================
Omnibus:                   323340.541   Durbin-Watson:                   0.113
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            25048.237
Skew:                           0.018   Prob(JB):                         0.00
Kurtosis:                       1.819   Cond. No.                         201.
==============================================================================
```

9) 특성과 레이블 관계 분석

수치형, 범주형 관계 분석

2. 로지스틱 회귀 분석 수행 결과

[다중 로지스틱 회귀 분석 결과]

- 모델 p-value : 1.13e-59
  → 모델 유의확률이 유의수준(0.05) 내에 존재하므로 유의미한 결과를 낸다고 판단

- features p-value : 0.000, 0.092, 0.022, 0.121
  → '평균기온', '평균상대습도'의 유의확률은 유의수준(0.05) 내에 존재하므로 각 feature를 통해 유의미한 label을 추출할 수 있다고 판단
  → '평균풍속'의 유의확률은 유의수준(0.05)에서 크게 벗어난 값이 아니므로 해당 feature를 통해 유의미한 label을 추출할 수 있다고 판단
  → '합계일사량'의 유의확률은 유의수준(0.05)에서 크게 벗어난 값이므로 해당 feature는 유의미한 label을 추출할 수 없다고 판단

- features 표준오차 : 0.001~0.037
  → 각 feature 모두 표준오차 값이 소숫점 둘째 or 셋째 자리의 값을 갖으므로 낮은 오차를 보인다.

[결과]

- '합계 일사량(MJ/m2)' 칼럼 제거에 대해 고려해보아야 한다.

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

1. 학습, 테스트 데이터 분리

```
from sklearn.model_selection import train_test_split

features = df.iloc[:, 1:]
label = df['카페명']
x_train, x_test, y_train, y_test = train_test_split(features, label, test_size=0.4, stratify=label, random_state=1)
print(x_train.shape, x_test.shape, y_train.shape, y_test.shape)
```

```
(12470, 10) (8314, 10) (12470,) (8314,)
```

→ 테스트 데이터 비율 : 40%

→ stratify를 적용하여 label 값을 적절하게 분리

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

2. 모델 선언 및 학습

```
rf_model = RandomForestClassifier(n_estimators=50, criterion='entropy', random_state=10)
rf_model.fit(x_train, y_train)
```

→ 내부 결정 트리 개수 : 50개

→ 표준 분류 방법 : entropy(무질서한 정도를 수치화한 값)

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

3. 분류 모델 성능 지표 – 정확도, 정밀도, 재현도

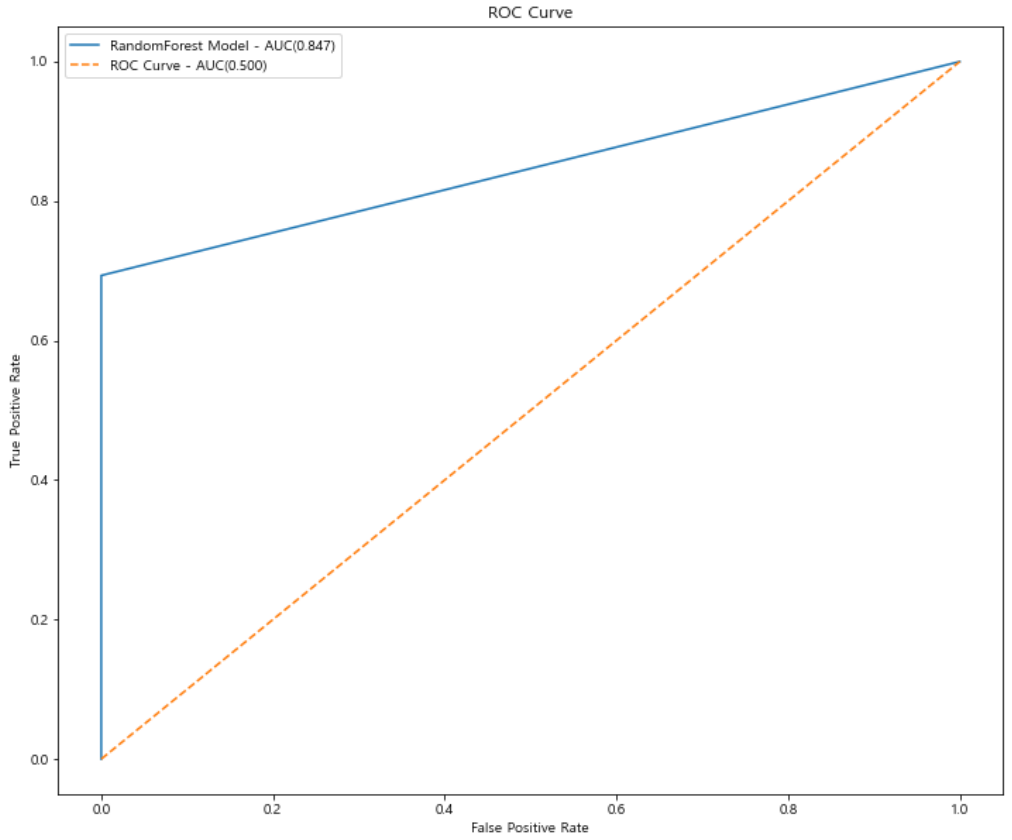
```
report = classification_report(y_test, y_pred)
print(report)
```

정확도 : 0.69
재현도 : 0.69
정밀도 : 0.71
→ 정확도, 재현도, 정밀도 모두 0.69~0.71 수준의 성능을 보이므로 모델 성능은 평균정도라고 판단

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

4. 분류 모델 성능 지표 – ROC Curve, AUC

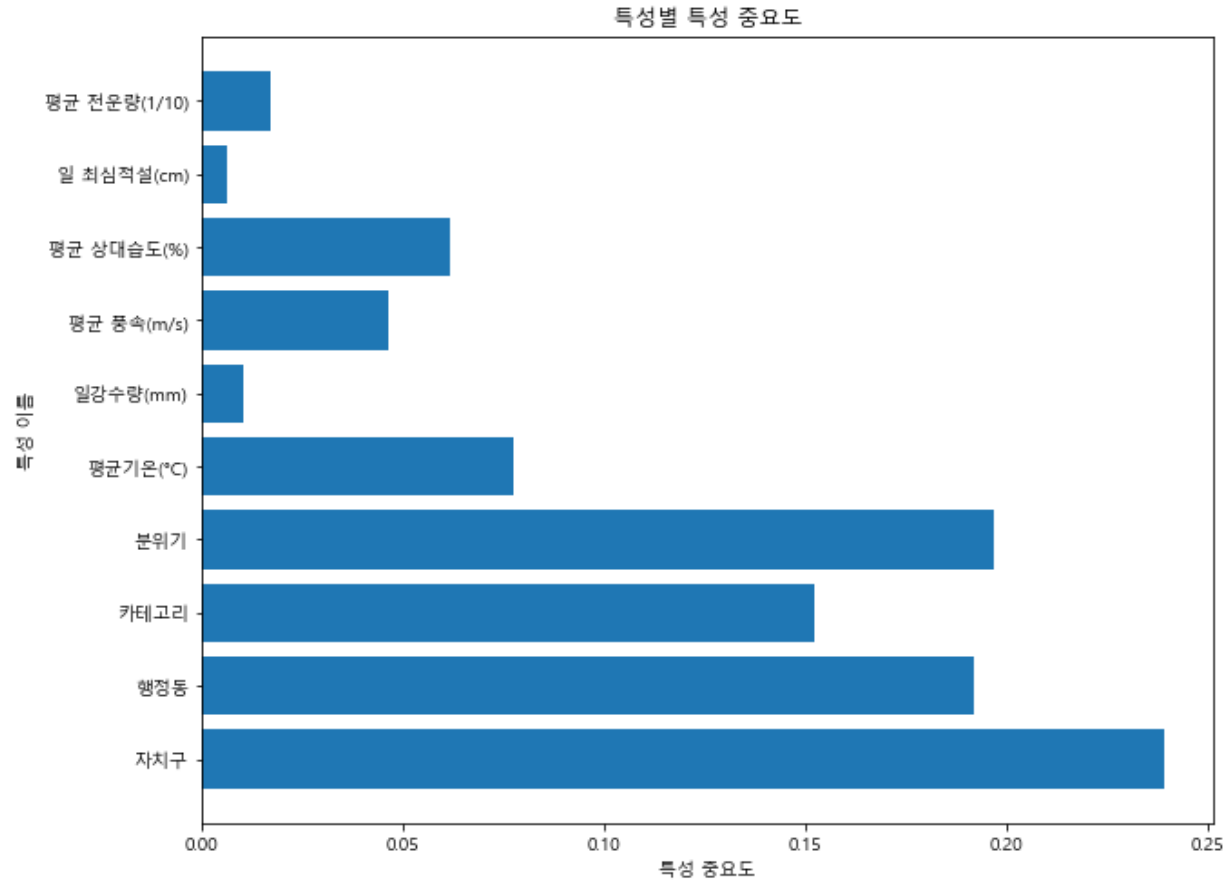


→ AUC 값 = 0.847 정도의 수치를 보이므로 평균 이상의 성능을 가진 모델이라고 판단 가능

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

5. 특성 중요도 추출



→ 제거 대상 특성들 : 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

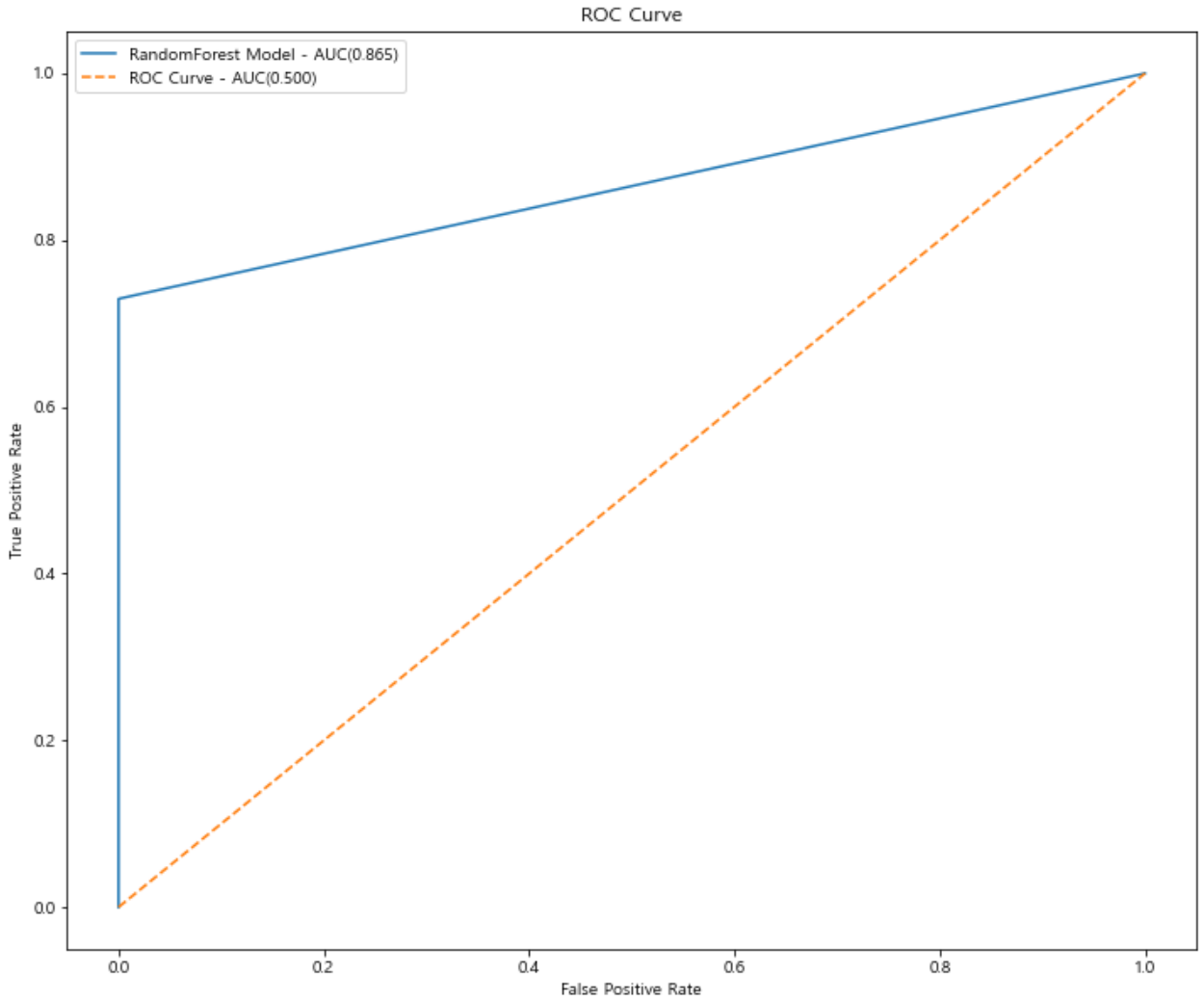
6. 특성 제거 후, 새로운 렌덤 포레스트 모델 학습 결과 – 정확도, 정밀도, 재현도 확인

정확도 : 0.73
재현도 : 0.72
정밀도 : 0.75
→ 정확도, 재현도, 정밀도 모두 0.72~0.75 수준의 성능을 보이므로 특정 칼럼을 제거하지 않은 모델보다 성능이 0.04 가량 상승함

10)전통적인 모델 ML 학습

랜덤 포레스트 다중 분류 모델

7. 특성 제거 후, 새로운 렌덤 포레스트 모델 학습 결과 – ROC Curve, AUC 확인



AUC 값 : 0.865

→ 원본 모델 AUC : 0.847
→ 특성 제거 모델 AUC : 0.865
→ 특성을 제거한 모델의 전반적인 성능이 조금 우수하다.

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

1. XGBoost 학습 레이블 설정

```
from sklearn.preprocessing import LabelEncoder

features = df.iloc[:, 1:]
label = df['카페명']

encoder = LabelEncoder()
encoded_label = encoder.fit_transform(label)
```

→ XGBoost 모델을 학습할 때는 레이블 범주가 0부터 순차적으로 시작해야 한다.

→ 위 조건을 만족하기 위해 LabelEncoder를 이용해 재작업한다.

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

2. XGBoost 모델 선언 및 학습

```
xgb_model = XGBClassifier(n_estimators=100, random_state=1)
xgb_model.fit(
    x_train, y_train,
    eval_metric='auc',
    eval_set=[[x_train, y_train], [x_test, y_test]],
    early_stopping_rounds=2
)
```

→ 내부 결정 트리 개수 : 100개

→ 학습 시, 검증 평가 방식 : AUC 값 사용

→ 조기 종료 : 2번 유지 시, 조기 종료 수행

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

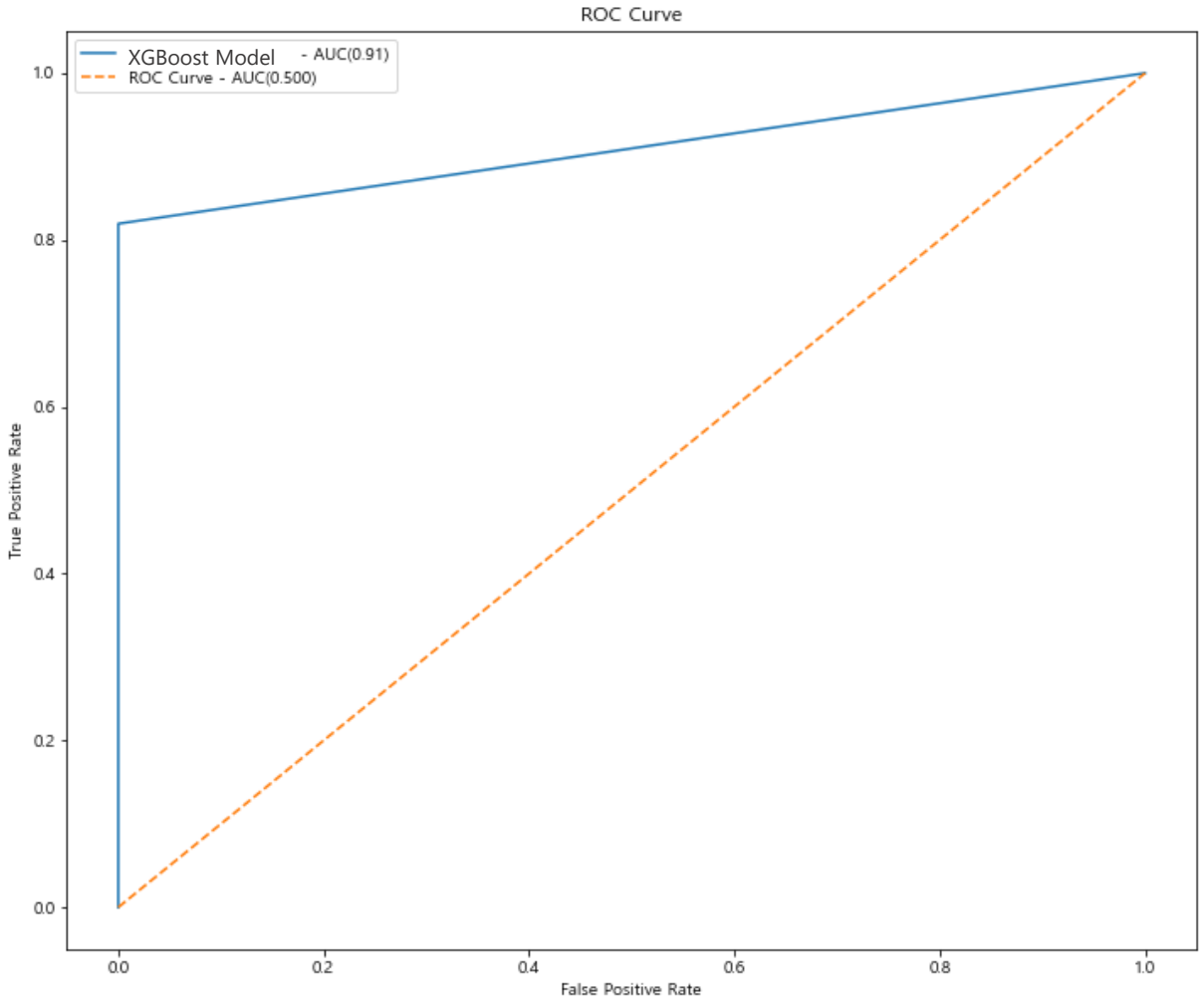
3. 분류 모델 성능 지표 – 정확도, 정밀도, 재현도

정확도 : 0.82
재현도 : 0.81
정밀도 : 0.83
→ 정확도, 재현도, 정밀도 모두 0.81~0.83 수준의 성능을 보이므로 높은 성능의 모델이라고 판단 가능
→ 랜덤포레스트 모델보다 0.14 정도 높은 정확도, 재현도, 정밀도를 보인다.

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

4. 분류 모델 성능 지표 – ROC Curve, AUC



AUC 값 : 0.91

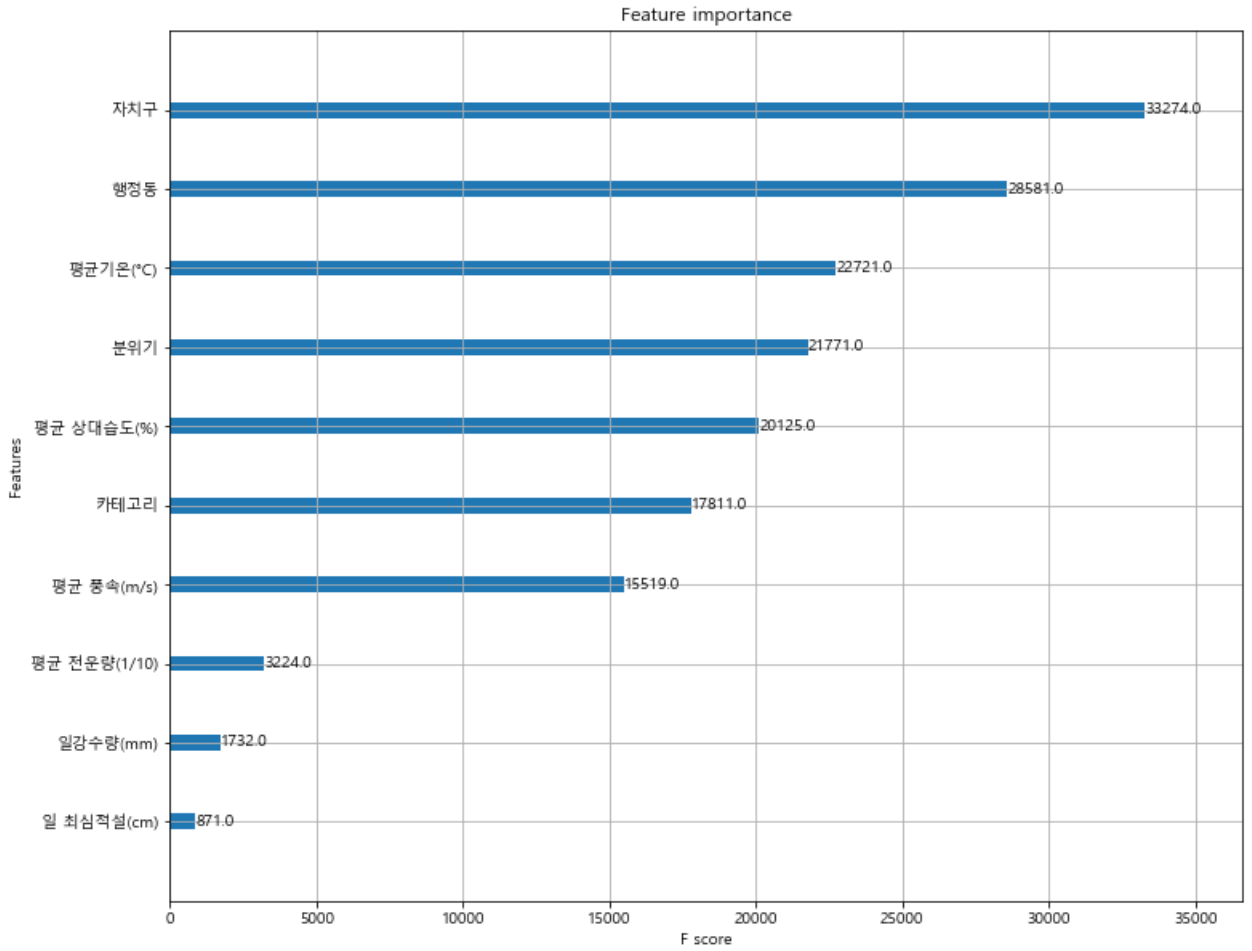
→ 랜덤포레스트 모델0.865

→ 랜덤포레스트 모델보다 0.05 정도 높은 AUC 값을 갖는다.

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

5. 특성 중요도 추출



→ 제거 대상 특성들 : 일강수량(mm), 일 최심적설(cm), 평균 전운량(1/10)

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

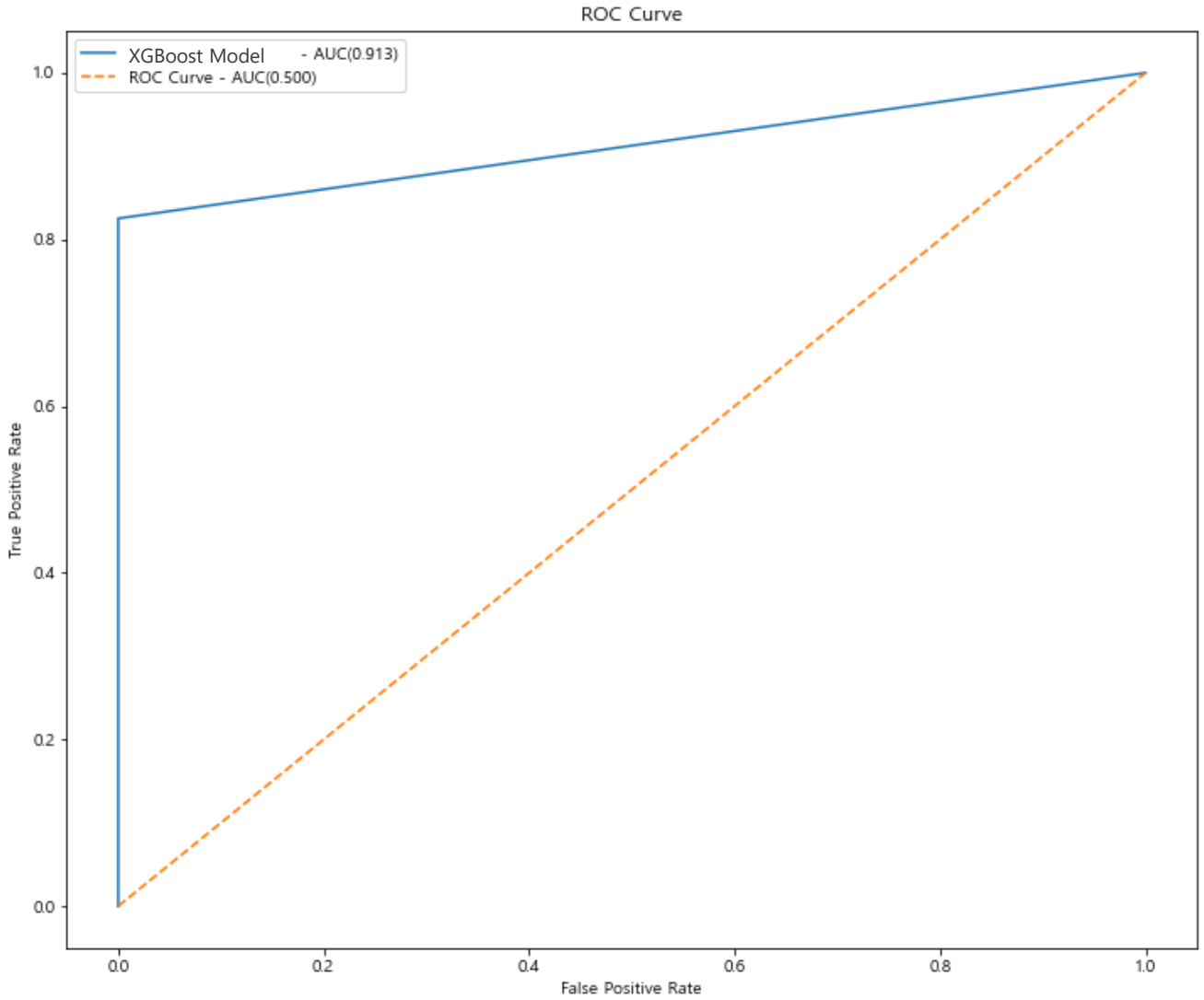
6. 특성 제거 후, 새로운 XGBoost 모델 학습 결과 – 정확도, 정밀도, 재현도

정확도 : 0.83
재현도 : 0.82
정밀도 : 0.83
→ 정확도, 재현도, 정밀도 모두 0.82~0.83 수준의 성능을 보이므로 원본 모델보다 0.01 정도의 성능이 향상됨
→ 랜덤포레스트 모델보다 0.15 정도 높은 정확도, 재현도, 정밀도를 보인다.

10)전통적인 모델 ML 학습

XGBoost 다중 분류 모델

7. 특성 제거 후, 새로운 XGBoost 모델 학습 결과 – ROC Curve, AUC



AUC 값 : 0.913

→ 원본 XGBoost 모델 AUC : 0.910
→ 특정 칼럼 제거 XGBoost 모델 AUC : 0.913

→ 원본 XGBoost 모델보다 0.003 정도 높은 AUC 값을 갖는다.

11)딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

1. 레이블 원-핫 인코딩

```
import pickle
from tensorflow.keras.utils import to_categorical
```

```
with open('./데이터 처리 객체/카페명_encoder.pickle', 'rb') as f:
    encoder = pickle.load(f)
```

```
y_train = encoder.transform(y_train)
y_valid = encoder.transform(y_valid)
y_test = encoder.transform(y_test)
```

```
y_train = to_categorical(y_train)
y_valid = to_categorical(y_valid)
y_test = to_categorical(y_test)
```

```
print(y_train.shape, y_valid.shape, y_test.shape)
```

```
(8727, 17615) (3741, 17615) (8312, 17615)
```

→ 딥러닝 다중 분류 모델의 레이블 형태는 인코딩된 상태의 2차원 레이블이다.

→ 위 조건을 만족하기 위해 to_categorical() 모듈을 사용하여 원-핫 인코딩을 수행

11) 딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

2. 딥러닝 모델 Layer 구축

[모델 Layer 실제 구축]

```
input_dim = x_train.shape[-1]
tag_size = y_train.shape[-1]
dropout_ratio = 0.3
```

```
# 입력층
inputs = Input(shape=(7,))

# 은닉층
net = Dense(units=1024, activation='relu')(inputs)
net = Dense(units=512, activation='relu')(net)
net = Dense(units=256, activation='relu')(net)
net = Dense(units=128, activation='relu')(net)
net = Dense(units=64, activation='relu')(net)

# 출력층
outputs = Dense(units=tag_size, activation='softmax')(net)


# 모델 선언
model = Model(inputs, outputs)
```

[모델 Summary 확인]

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 7)]               0

 dense (Dense)               (None, 1024)              8192

 dense_1 (Dense)             (None, 512)               524800

 dense_2 (Dense)             (None, 256)               131328

 dense_3 (Dense)             (None, 128)               32896

 dense_4 (Dense)             (None, 64)                8256

 dense_5 (Dense)             (None, 17615)             1144975

=================================================================
Total params: 1,850,447
Trainable params: 1,850,447
Non-trainable params: 0
_________________________________________________________________
None
```

딥러닝 MLP 다중 분류 모델

3. 모델 학습 시, 최적화 설정

```
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])
```

[최적화 방법 설명]

- 최적화 함수(optimizer) : Adam 알고리즘 + 학습률 0.001
- 손실 함수(loss) : 다중 분류 설정(categorical_crossentropy)
- 평가 방식 : 분류 정확도(accuracy)

딥러닝 MLP 다중 분류 모델

4. 조기 종료, 체크 포인트 설정

```
es = EarlyStopping(
    monitor='val_loss',
    patience=10
)
```

```
cp = ModelCheckpoint(
    monitor='val_loss',
    filepath='save_model.hdf5',
    verbose=0,
    save_best_only=True
)
```

[조기 종료]

- 모니터링 기준 : 검증 loss 값
- 조기 종료 라운드 : 10번 지속 시, 수행

[모델 체크 포인트]

- 모니터링 기준 : 검증 loss 값
- 최고 상태의 모델을 중간 저장

딥러닝 MLP 다중 분류 모델

5. 모델 학습

```
history = model.fit(
    x=x_train,
    y=y_train,
    batch_size=64,
    epochs=200,
    validation_data = [x_valid, y_valid],
    callbacks=[es, cp],
    verbose=2
)
```

[모델 학습 파라미터 설명]

- batch_size : 64 (32는 너무 적어 오래 걸림, 128은 너무 커서 불안정함)

- epochs : 총 200번 학습 반복

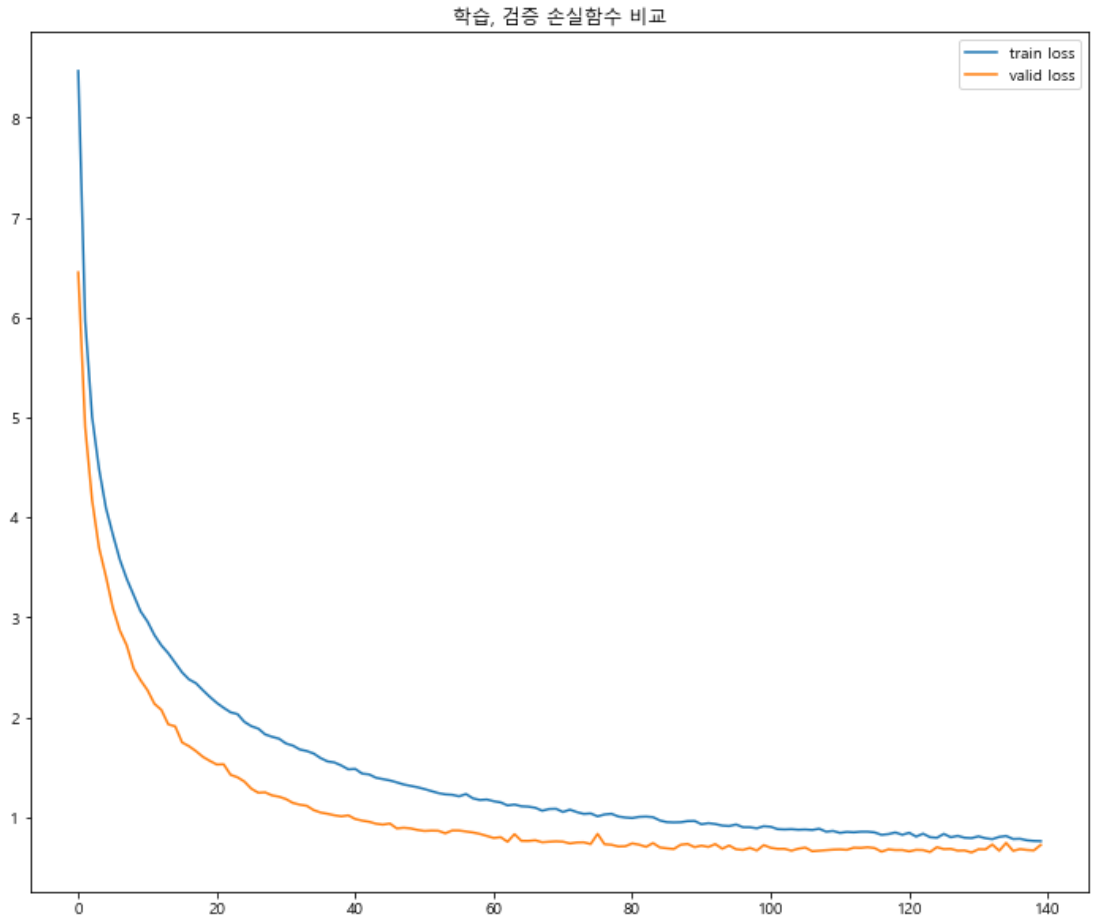
- validation_data : 검증 데이터 지정

- callbacks : 조기 종료, 체크 포인트 객체 지정

11)딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

6. 모델 손실 함수 그래프

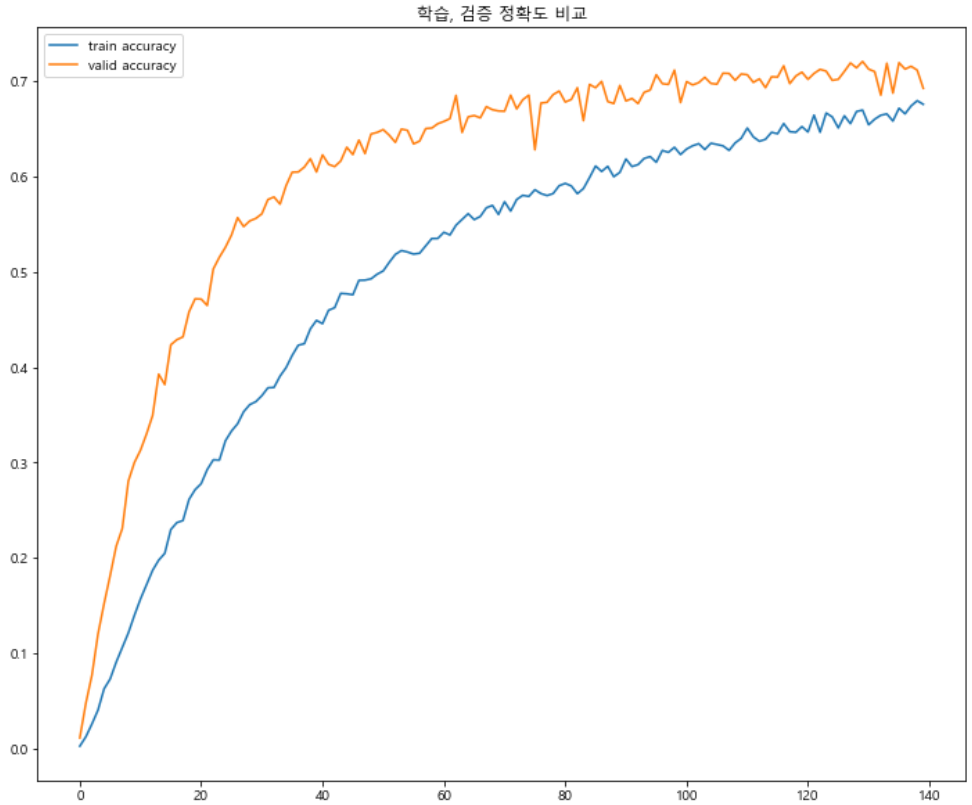


→ 학습이 진행될수록 학습 loss와 검증 loss는 하나의 값으로 수렴해간다.

11)딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

7. 모델 정확도 그래프



→ 학습을 진행하면서 검증 정확도에서 노이즈가 많이 발생하지만,

학습 정확도와 검증 정확도 모두 하나의 값으로 수렴하게 된다.

11)딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

8. 분류 모델 성능 평가 – 정확도, 재현도, 정밀도

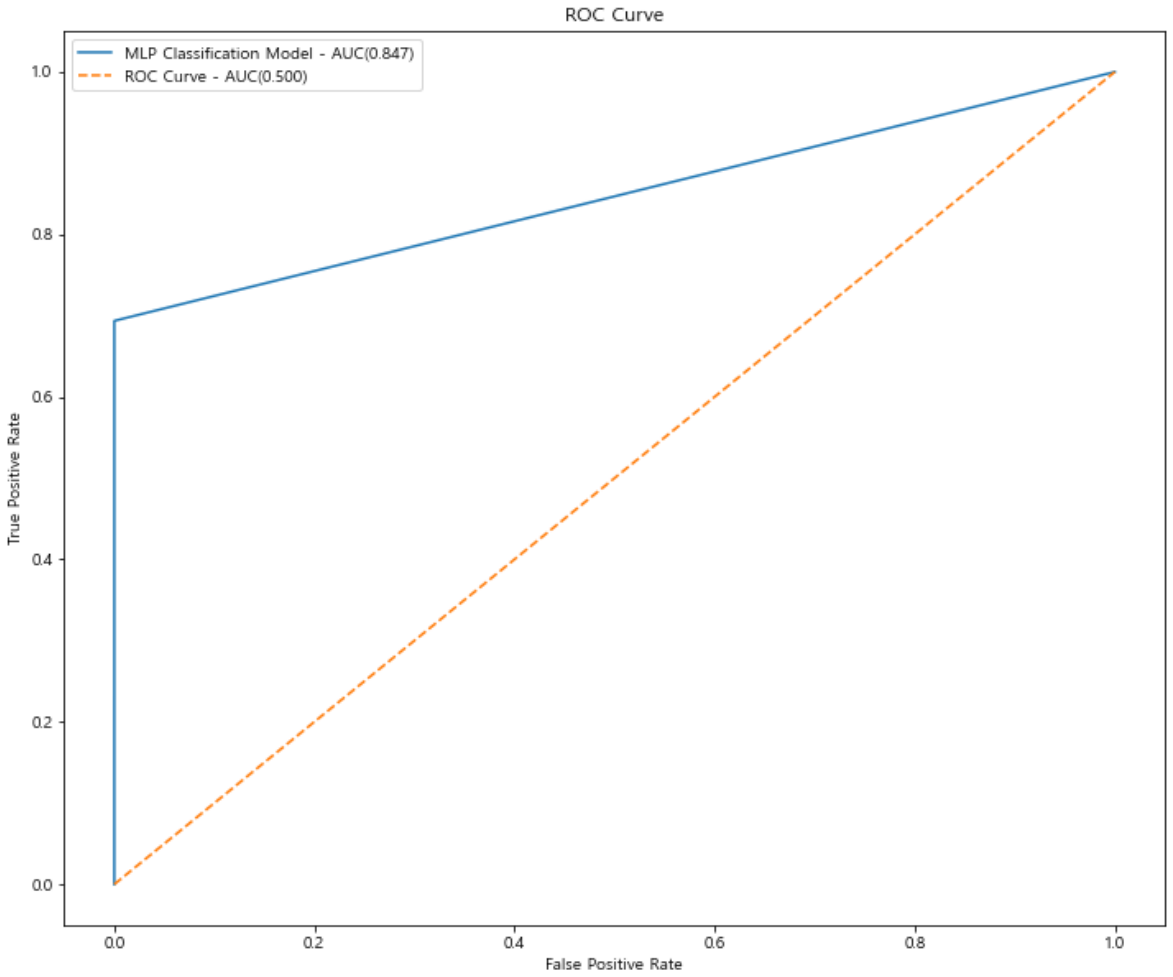
→ 정확도 : 0.69
→ 정밀도 : 0.61
→ 재현도 : 0.69

- 정확도, 정밀도, 재현도에서 전체적으로 0.61~0.69 사이의 값을 갖으므로 분류 모델의 성능은 평균이거나 평균 이하의 성능을 갖는 것으로 판단

11) 딥러닝 모델 DL 학습

딥러닝 MLP 다중 분류 모델

9. 분류 모델 성능 평가 – ROC Curve, AUC



AUC 값 : 0.847

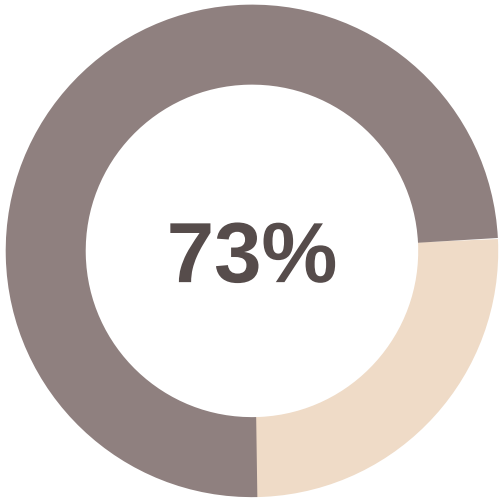
→ MLP 분류 모델 : 0.847

→ 랜덤포레스트 모델보다는 높은 값이 나온다.
→ XGBoost 모델보다는 낮은 값이 나온다.

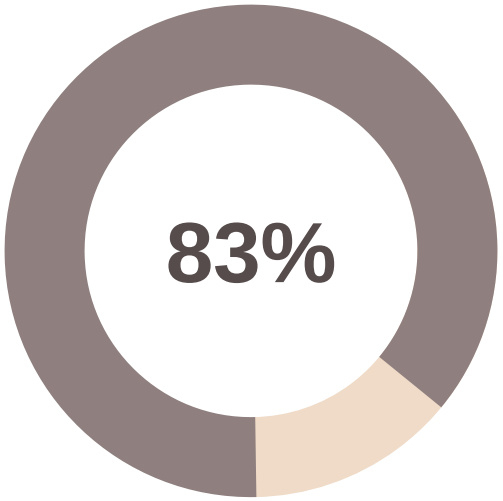
12)사용 모델 확정

사용할 모델 확정을 위한 평가

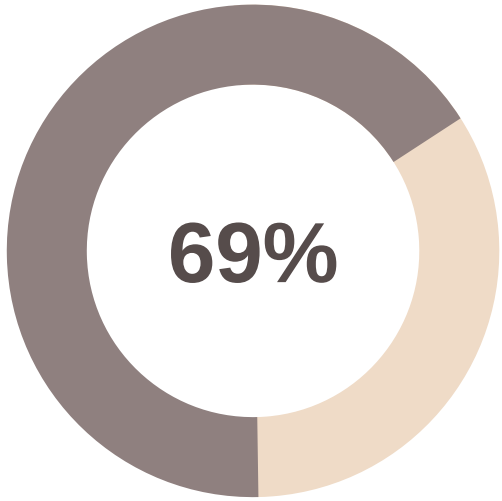
1. 분류 모델 성능 평가 지표 - 정확도



RandomForest



XGBoost

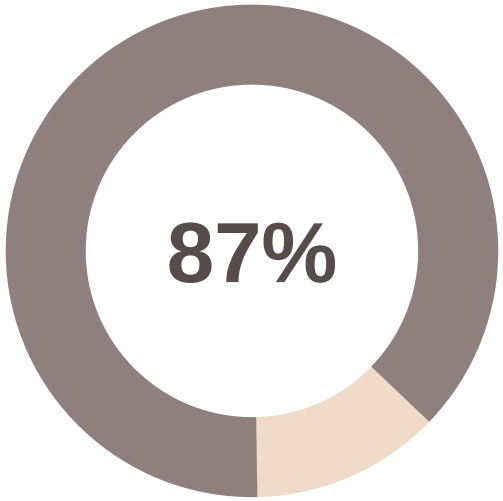


MLP

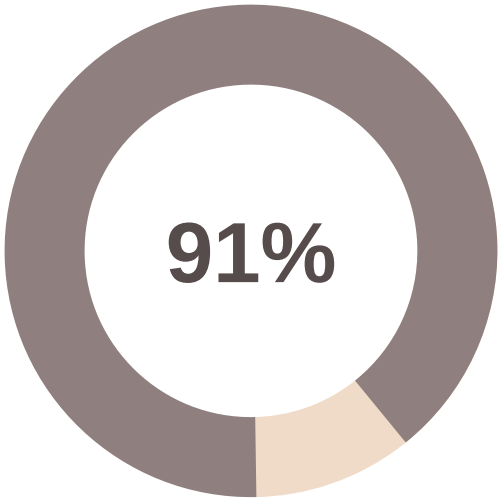
12)사용 모델 확정

사용할 모델 확정을 위한 평가

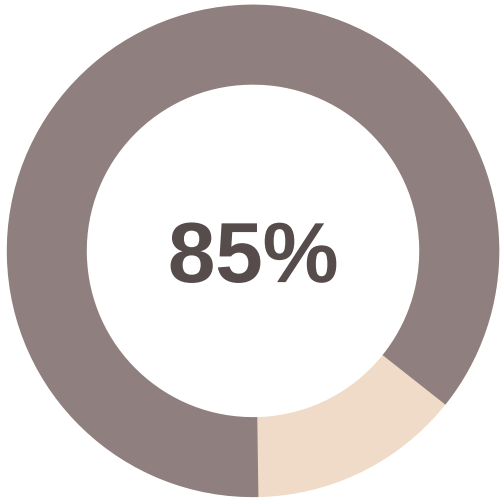
2. 분류 모델 성능 평가 지표 – AUC



RandomForest



XGBoost



MLP

12) 사용 모델 확정

사용할 모델 확정을 위한 평가

3. 분류 모델 비교 결과

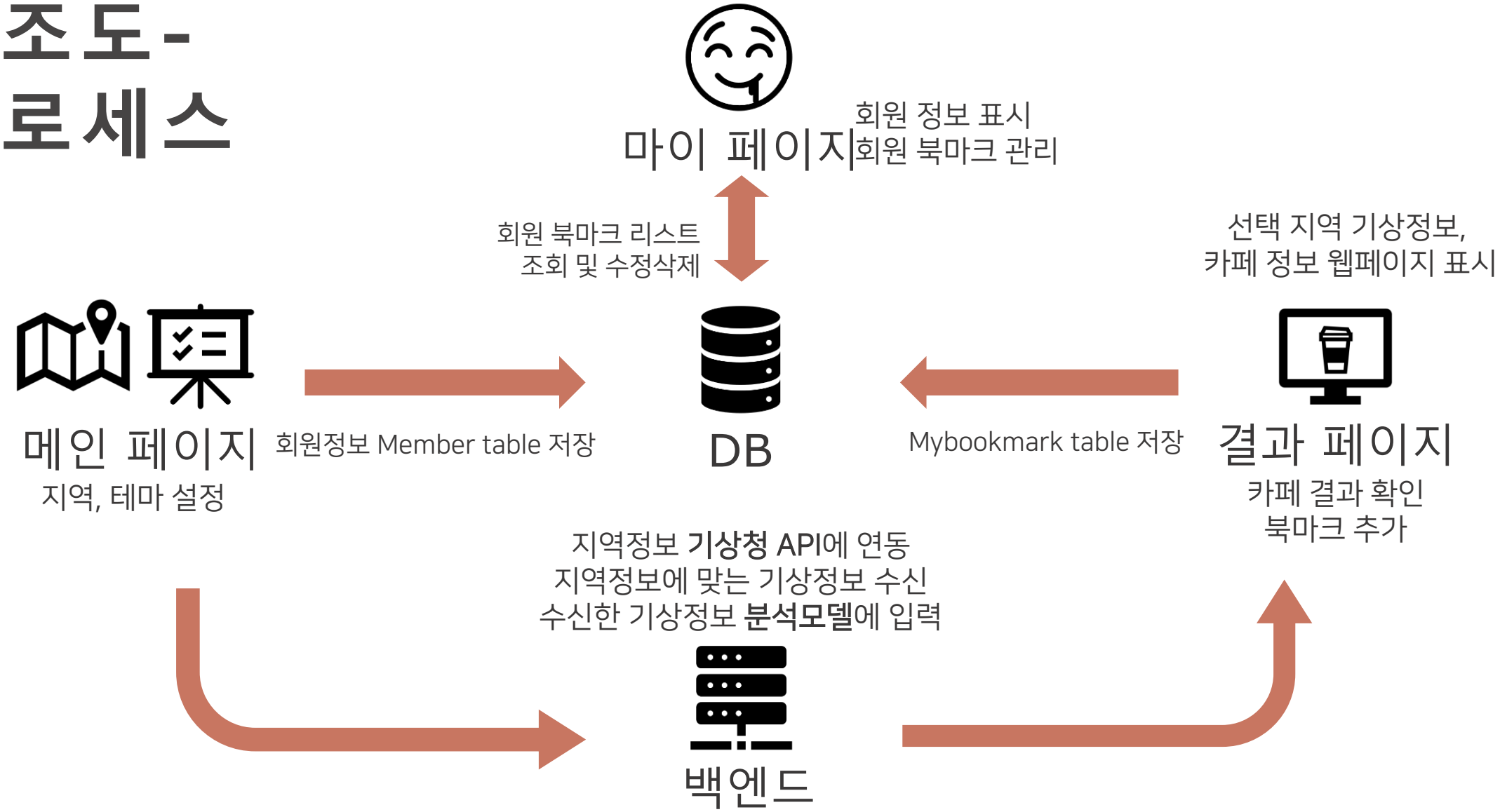
	RandomForest	XGBoost	MLP
사용 편리성	상	하	중
잠재성	중	중	상
응용력	중	중	상
성능	중	상	중
결과	X	X	O

Part 3

Web

003 - Web

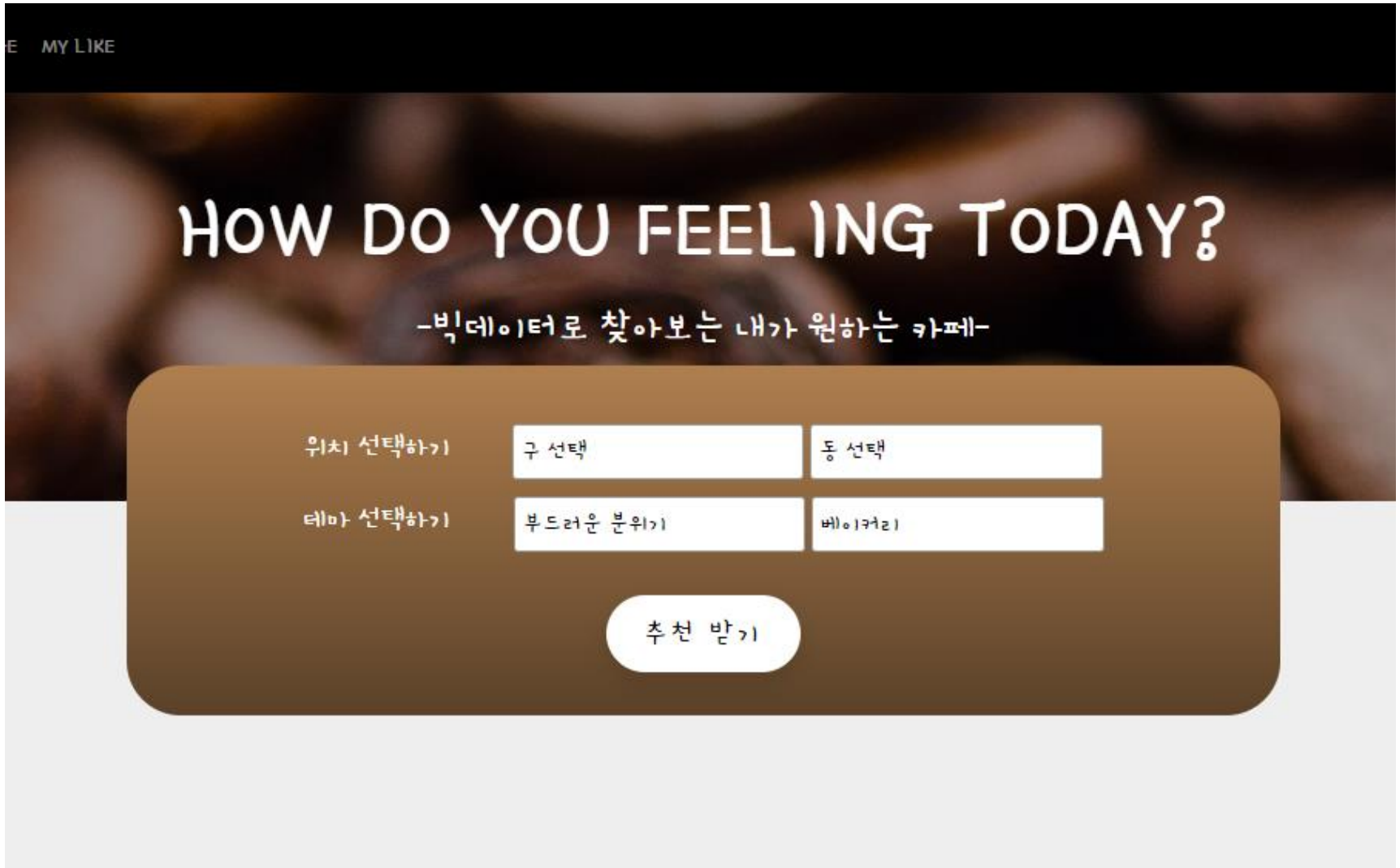
구조도-프로세스



1) 메 인 페 이 지

지 역 및 테 마 선 택

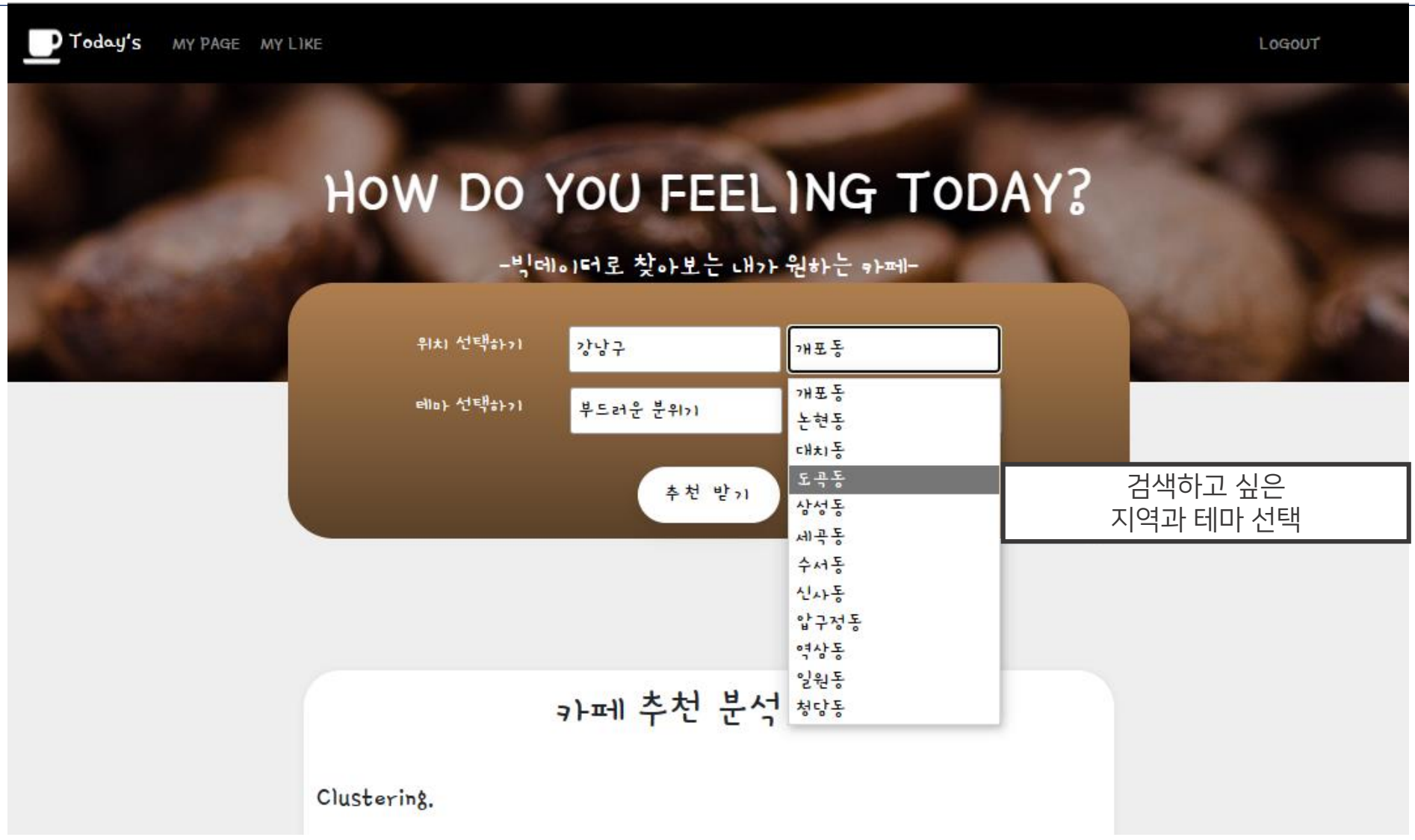
구와 동을 선택하여 위치 선택 및 테마는 데이터 분석을 통해 카테고리 분류



1) 메인 페이지

지역 및 테마 선택

구와 동을 선택하여 위치 선택 및 테마는 데이터 분석을 통해 카테고리 분류



검색하고 싶은
지역과 테마 선택

2) 백엔드

기상청 API 연동 & 기상정보모델 입력

데이터 출처 : 공공데이터 포털 / API명 : 기상청_단기예보 조회서비스

2) 백 엔 드

기 상 청 A P I 연 동 & 기 상 정 보 모 델 입 력

API 연동을 위한 모델 설계 및 연동 후 필요 데이터 추출을 위한 모델 입력, 수정

2) 백 엔 드

기상청 API 연동 & 기상정보모델 입력

전달된 정보 처리 과정

```
result(request):
if 'Members' not in request.session:
    return render(request, 'users/loginform.html')

if request.method == 'POST':
    gu = request.POST.get('sido')
    dong = request.POST.get('gugun')
    mood = request.POST.get('mood')
    food = request.POST.get('food')
    print('responded value:', gu, dong, mood, food)
    try:

        # 선택된 날짜 기반
        global sel_lat_long
        sel_lat_long = func.location_to_coord(gu, dong)
        sel_nx_ny = func.grid(sel_lat_long['lat'], sel_lat_long['long'])
        global selected_weather
        selected_weather = func.dangi_api(sel_nx_ny['x'], sel_nx_ny['y'])
        background = func.set_background(hour, selected_weather['code'])

        maxim = func.maxim()

        # 모델 예측을 위한 데이터 전처리
        data = {}

        # [ 자치구 칼럼 전처리 ]
        with open('./todays/mainapp/encoders/자치구_encoder.pickle', 'rb')
            administ_encoder = pickle.load(f)
        administ = administ_encoder.transform([gu])[0]
        data['자치구'] = [administ]

        # [ 행정동 칼럼 전처리 ]
        with open('./todays/mainapp/encoders/자치구별_행정동_encoders.pickle
            building_encoders = pickle.load(f)
        building_encoder = building_encoders[administ]
        building = building_encoder.transform([dong])[0]
        data['행정동'] = [building]

        # [ 카테고리 칼럼 전처리 ]
        with open('./todays/mainapp/encoders/카테고리_encoder.pickle', 'rb
```

전달된 지역정보를
위도/경도계로 변환

API 값 전달 위해
좌표값 한번 더 변환

단기예보 API에서
기상정보 출력, 저장

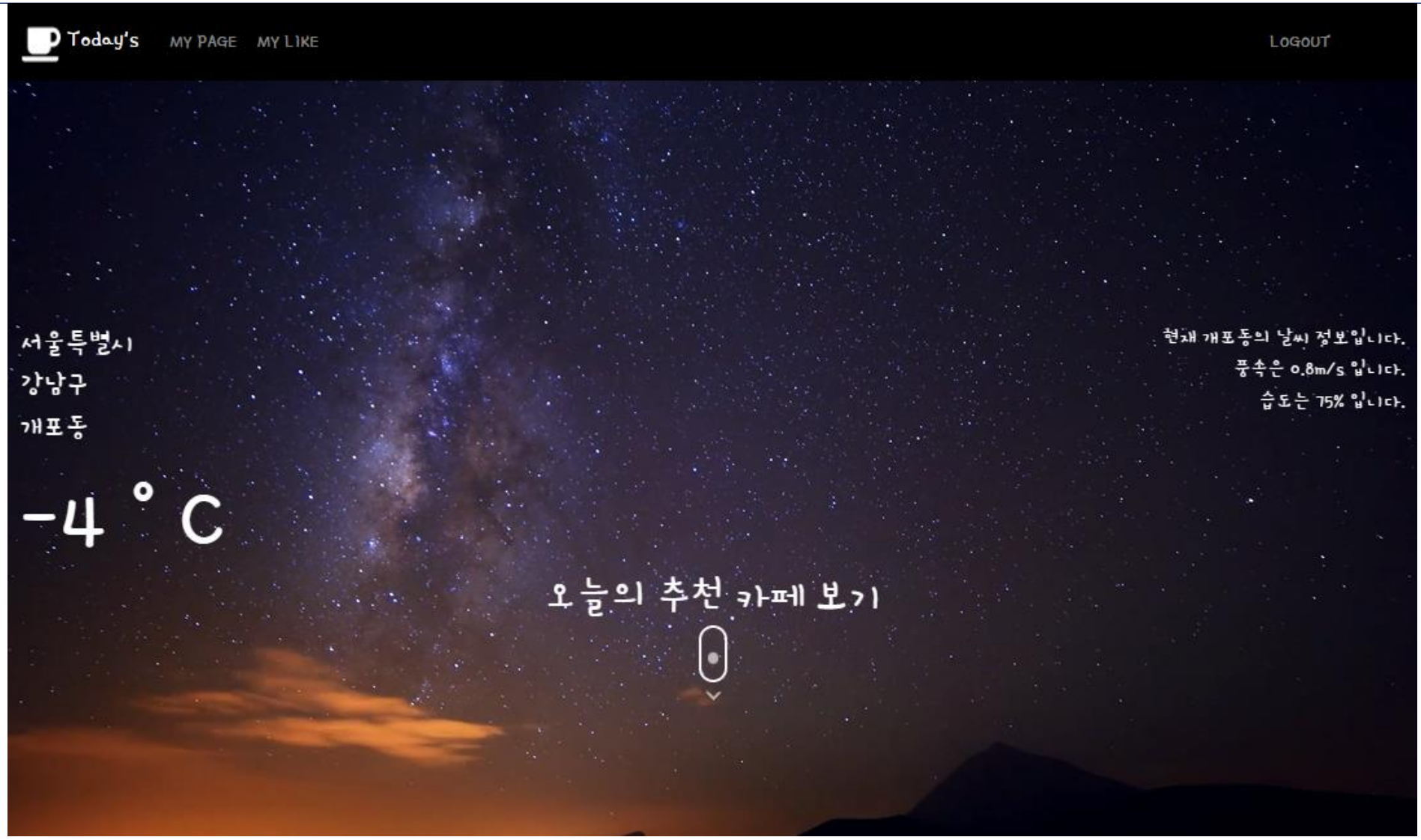
지역정보, 테마 컬럼 전처리

```
# 모델 예측
with open('./todays/mainapp/encoders/카페명_encoder.pickle', 'r
    cafe_encoder = pickle.load(f)

data_list_df = pd.read_csv('./todays/mainapp/datasets/Django에
data_list_df.drop(['Unnamed: 0'], axis=1, inplace=True)

model = models[0]
new_y = model.predict(new_x)
new_y = new_y.tolist()[0]
result_df = pd.DataFrame(columns=data_list_df.columns)

for i in range(5):
    idx = new_y.index(np.max(new_y))
    new_y[idx] = 0.0
    cafe_name = cafe_encoder.inverse_transform([idx])[0]
    test_df = data_list_df[data_list_df['카페명'] == cafe_name]
    result_df = pd.concat([result_df, test_df], axis=0)

result_df.reset_index(inplace=True)
result_df.drop(['index'], axis=1, inplace=True)
# print(result_df)

# 결과 데이터프레임 loop
loop = []
for i, data in enumerate(result_df.values):
    test_dict = {
        'num': i + 1,
        'name': data[0],
        'addr': data[1] + ' ' + data[2],
        'category': data[3],
        'mood': data[4],
        'img_addr': data[-1]
    }
    loop.append(test_dict)
loop_cnt = len(loop)

context = {
    'background': background,
    'latitude': nx_ny['x'],
```

3) 결과페이지

기상정보 표시

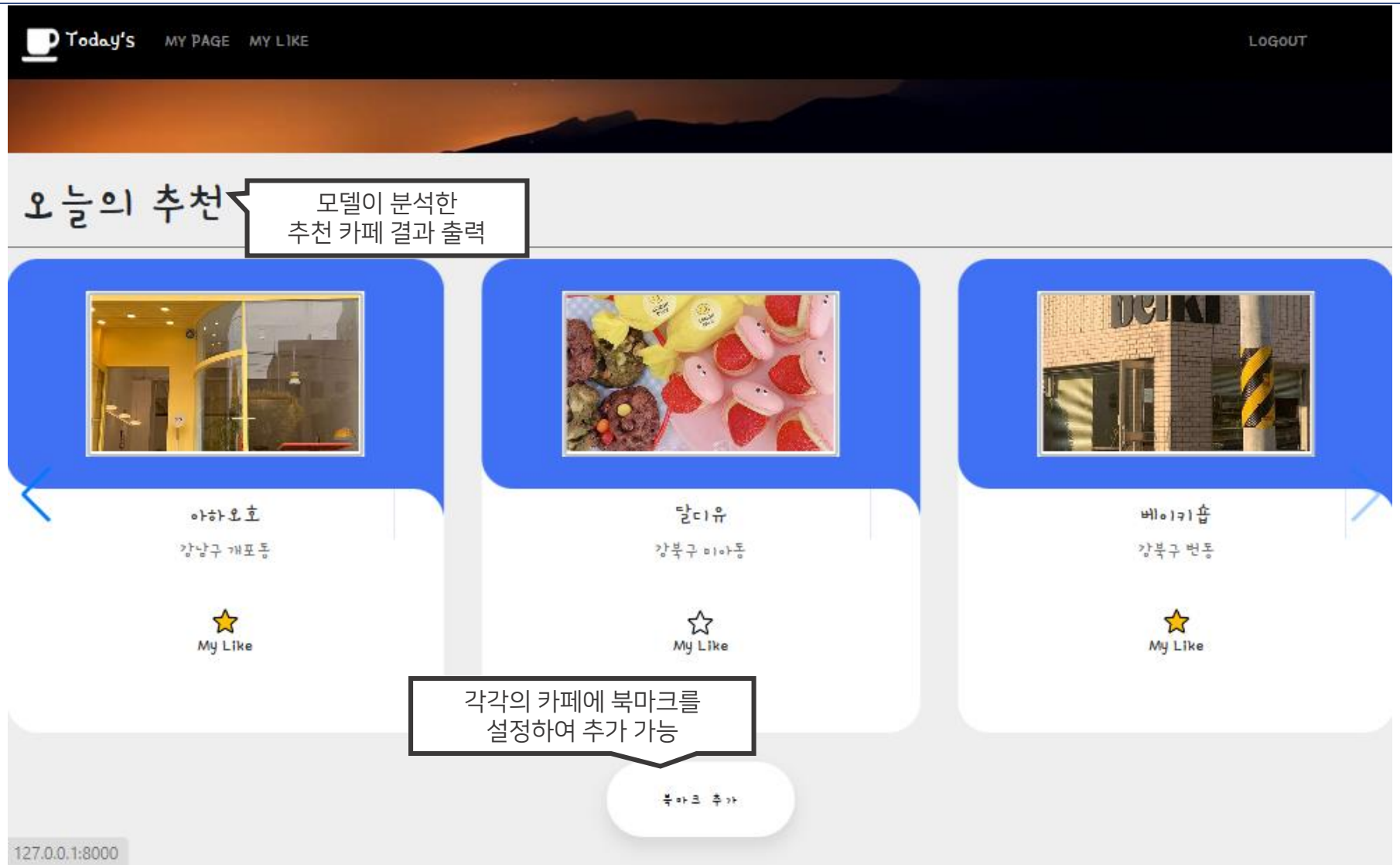
API와 모델을 통해 수집된 기상정보 표시



3) 결과페이지

모델 출력 결과(카페) & 북마크 기능

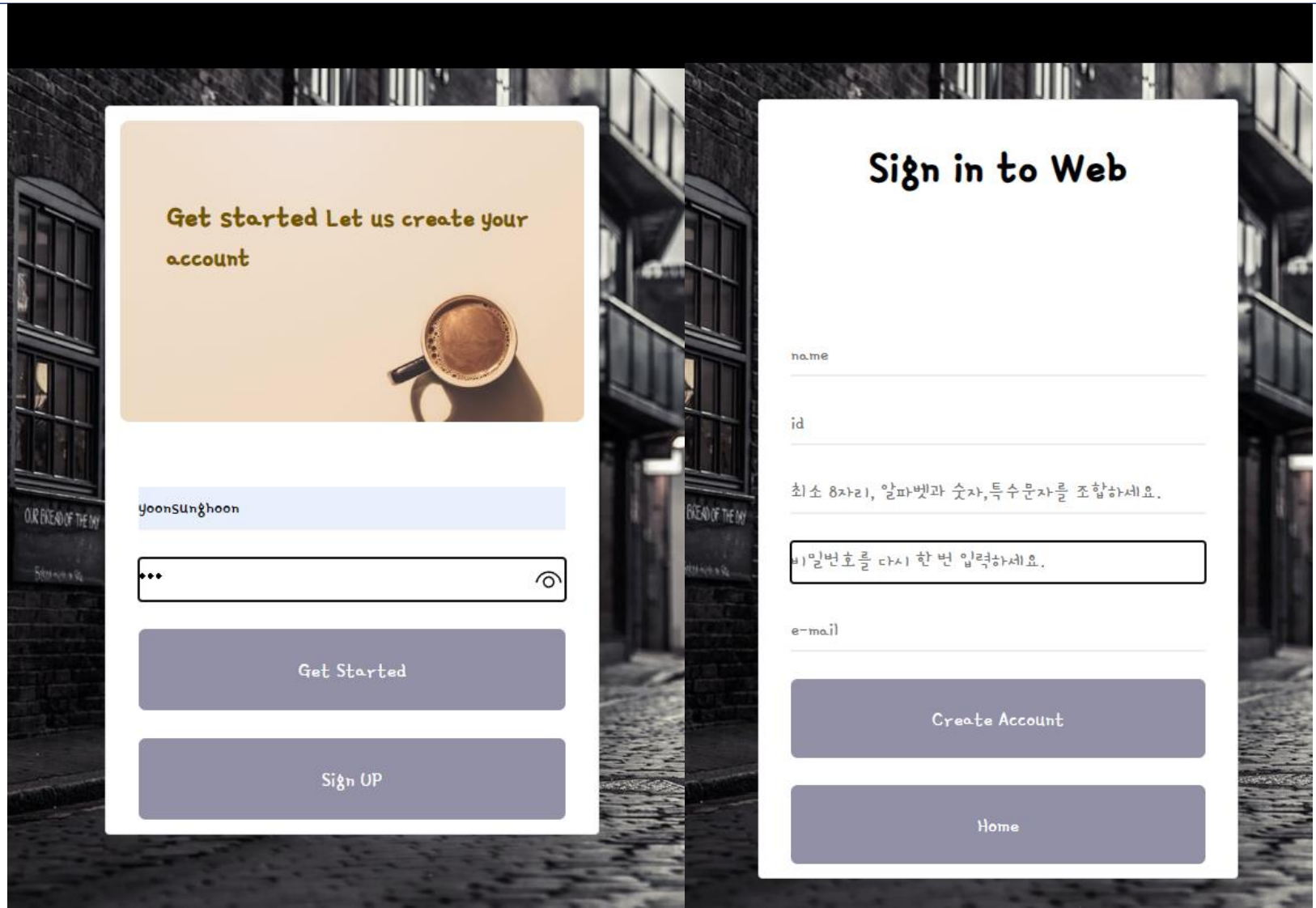
모델을 통해서 추천한 카페 리스트 및 북마크 기능 구현



4) 회원정보

회원가입 & 로그인/로그아웃

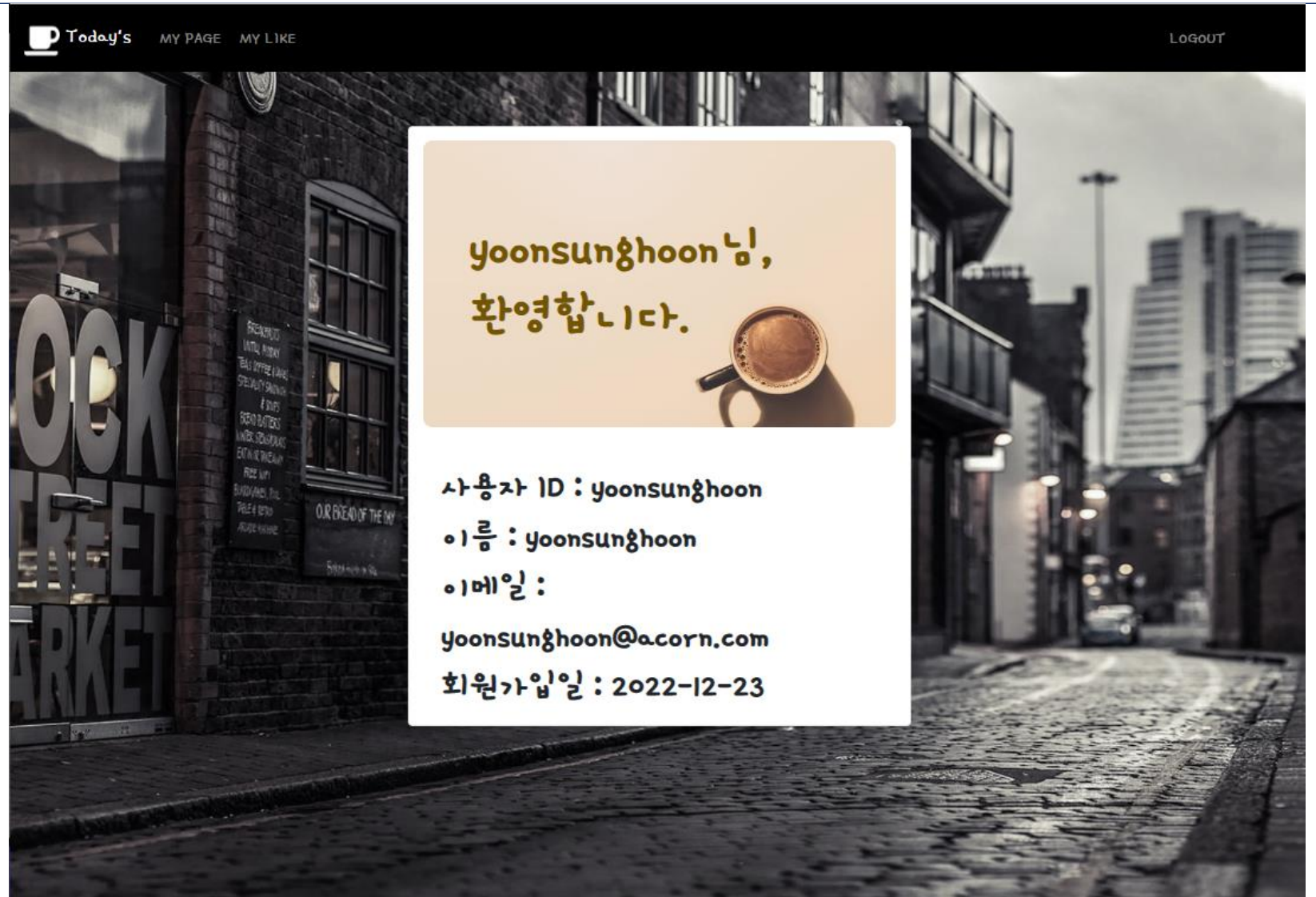
회원가입을 통해 입력된 정보를 DB에 저장하고 로그인/로그아웃 기능 구현



4) 회원정보

내 정보 보기

내 정보를 확인할 수 있는 마이페이지 구현



4) 회원정보

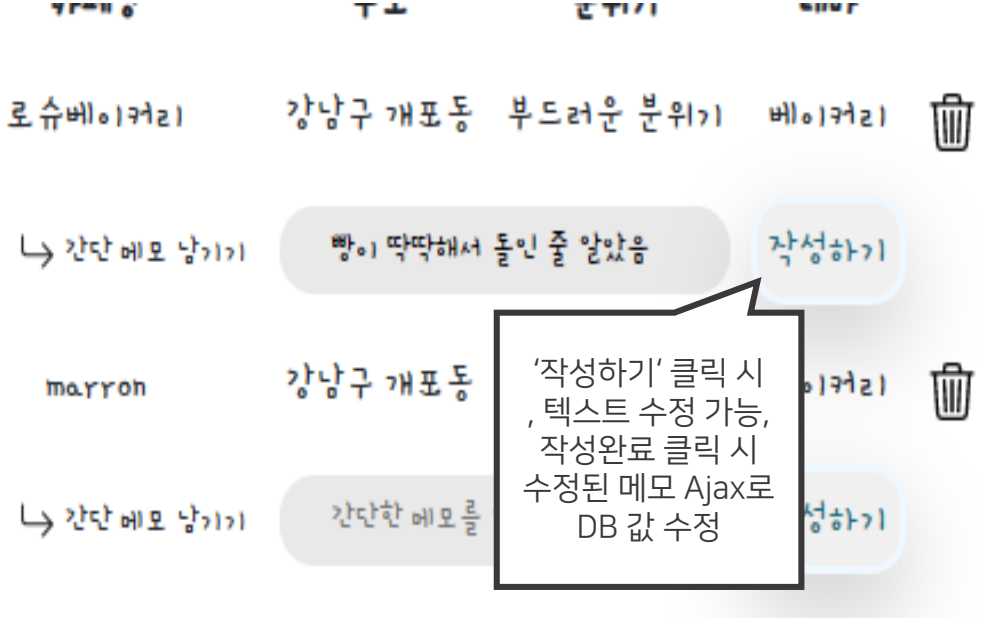
내 북마크 보기

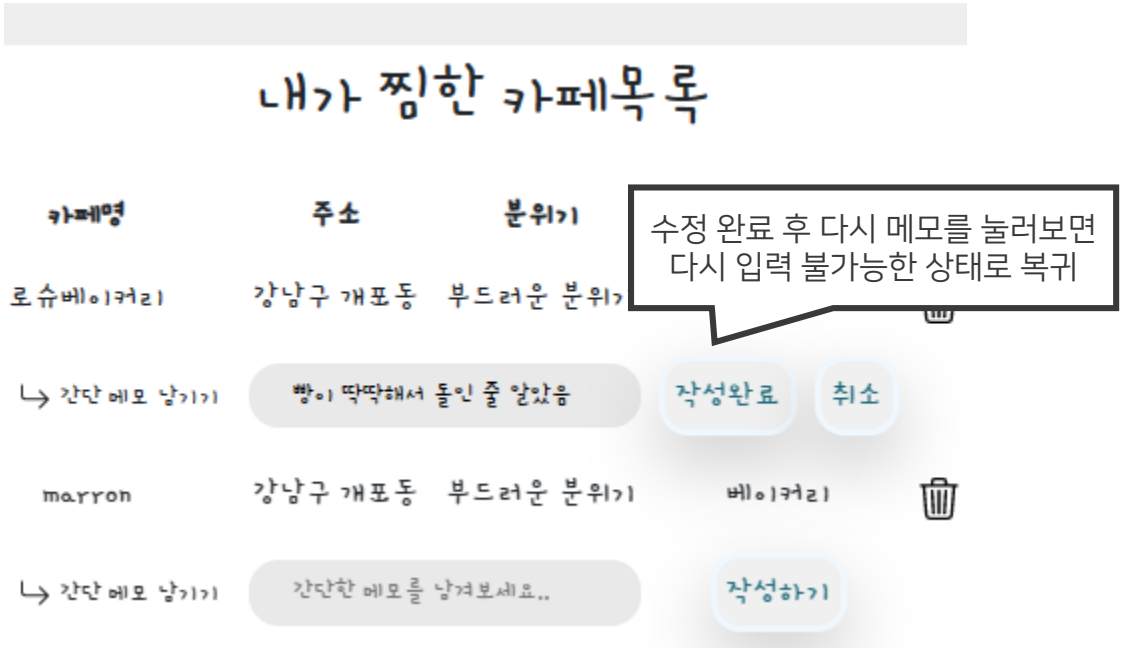
추가된 북마크 목록을 리스트식으로 출력 및 간단한 메모 기능 구현

4) 회원정보

내 북마크 보기

추가된 북마크 목록을 리스트식으로 출력 및 간단한 메모 기능 구현





5) 배포

AWS EC2 인스턴스

AWS 계성 생성 및 인스턴스 생성 후 실행

i-01f04003dd6e7988f (Django_WebServer)에 대한 인스턴스 요약 정보

less than a minute 전에 업데이트됨

연결 | 인스턴스 상태 | 작업

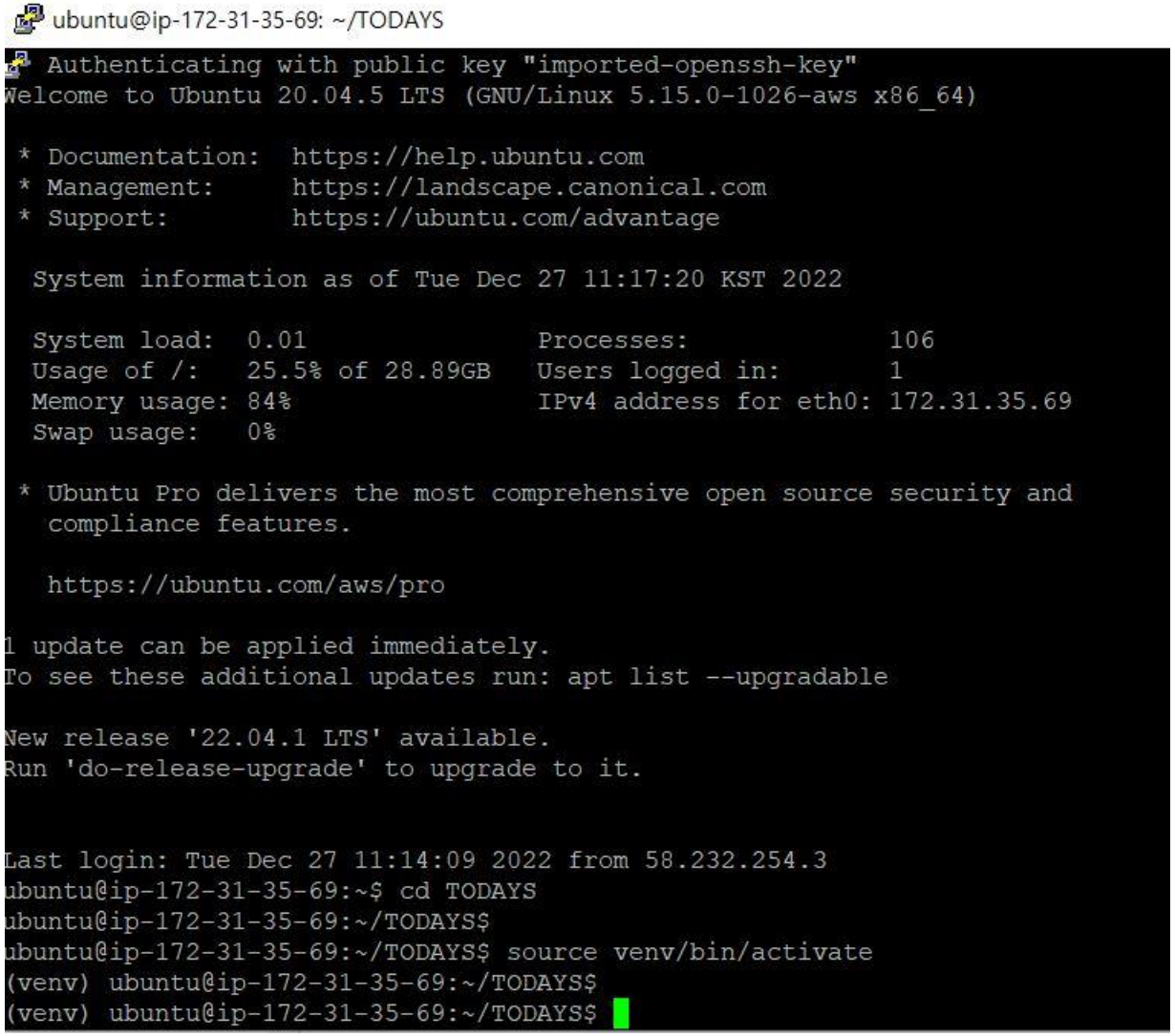
인스턴스 ID i-01f04003dd6e7988f (Django_WebServer)	퍼블릭 IPv4 주소 3.36.207.105 | 개방 주소법	프라이빗 IPv4 주소 172.31.35.69
IPv6 주소 –	인스턴스 상태 실행 중	퍼블릭 IPv4 DNS ec2-3-36-207-105.ap-northeast-2.compute.amazonaws.com | 개방 주소법
호스트 이름 유형 IP 이름: ip-172-31-35-69.ap-northeast-2.compute.internal	프라이빗 IP DNS 이름(IPv4만 해당) ip-172-31-35-69.ap-northeast-2.compute.internal	
프라이빗 리소스 DNS 이름 응답 IPv4(A)	인스턴스 유형 t2.micro	탄력적 IP 주소 3.36.207.105 [퍼블릭 IP]
자동 할당된 IP 주소 –	VPC ID vpc-0f851f160a7663f22	AWS Compute Optimizer 찾기 권장 사항을 위해 AWS Compute Optimizer에 옵트인합니다. | 자세히 알아보기
IAM 역할 –	서브넷 ID subnet-007d2e5482e84158d	Auto Scaling 그룹 이름 –

Name	인스턴스 ID	인스턴스 상태	인스턴스 유형	상태 검사	경보 상태	가용 영역
Django_WebS...	i-01f04003dd6e7988f	실행 중	t2.micro	2/2개 검사 통과	경보 없음	ap-northeast-2c

5) 배포

Ubuntu 환경으로 EC2 접속

PuTTy Client를 사용하여 원격으로 Ubuntu 환경 서버 접속



5) 배포

Local과 연동하기 위한 설정

Local 컴퓨터의 Django Project와 연동하기 위한 설정 진행

```
[Unit]
Description=uWSGI Emperor service

[Service]
ExecStart=/usr/local/bin/uwsgi --emperor /etc/uwsgi/sites
Restart=on-failure
KillSignal=SIGQUIT
Type=notify
NotifyAccess=all
StandardError=syslog

[Install]
WantedBy=multi-user.target
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/systemd/system/uwsgi.service" 13L, 236C
```

ubuntu@ip-172-31-35-69: ~/TODAYS

```
[uwsgi]
chdir=/home/ubuntu/TODAYS
module=todays.wsgi:application
master=True
pidfile=/tmp/project-master.pid
vacuum=True
max-requests=5000
daemonize=/home/ubuntu/TODAYS/django.log
home=/home/ubuntu/TODAYS/venv
virtualenv=/home/ubuntu/TODAYS/venv
socket=/home/ubuntu/TODAYS/uwsgi.sock
chmod-socket=666
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"uwsgi.ini" 12L, 301C
```

5) 배포

Socket 설정 후 sock파일 생성

Socket 파일 생성 및 실행을 통한 웹서비스 실행

```
ubuntu@ip-172-31-35-69:~$ cd TODAYS
ubuntu@ip-172-31-35-69:~/TODAYS$
ubuntu@ip-172-31-35-69:~/TODAYS$ source venv/bin/activate
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$ sudo vim /etc/systemd/system/uwsgi.service
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$ sudo vim uwsgi.ini
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$ ls
Web_Page_Layout_v1.pdf  django.log  manage.py  todays     uwsgi.sock
dbconnect.py            mainapp     mydb.dat   uwsgi.ini  venv
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$
```

5) 배포

AWS

Nginx config 설정과 Nginx-sites-enable 설정을 통한 고정IP와의 연결

ubuntu@ip-172-31-35-69: ~/TODAYS

```
user root;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
        worker_connections 768;
        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##
        upstream django{
            server unix:/home/ubuntu/TODAYS/uwsgi.sock;
        }

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;
```

ubuntu@ip-172-31-35-69: ~/TODAYS

```
        # include snippets/snakeoil.conf;

        root /var/www/html;

        # Add index.php to the list if you are using PHP
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                include /etc/nginx/uwsgi_params;
                uwsgi_pass django;
        }

        location /static/ {
                alias /home/ubuntu/TODAYS/mainapp/static/;
        }
        # pass PHP scripts to FastCGI server
        #
        #location ~ \.php$ {
        #       include snippets/fastcgi-php.conf;
        #
        #       # With php-fpm (or other unix sockets):
        #       fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        #}

        # deny access to .htaccess files, if Apache's document root
-- INSERT --
```

5) 배포

AWS

Nginx 와 Uwsgi 설정 재시작 후 상태 확인

ubuntu@ip-172-31-35-69: ~/TODAYS

```
^C
(venv) ubuntu@ip-172-31-35-69:~/TODAYS$ sudo systemctl status uwsgi
● uwsgi.service - uWSGI Emperor service
     Loaded: loaded (/etc/systemd/system/uwsgi.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-27 10:39:31 KST; 43min ago
   Main PID: 3364 (uwsgi)
     Status: "The Emperor is governing 0 vassals"
      Tasks: 1 (limit: 1143)
     Memory: 1.2M
     CGroup: /system.slice/uwsgi.service
             └─3364 /usr/local/bin/uwsgi --emperor /etc/uwsgi/sites

Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: detected binary path: /usr/local/bin/uwsgi
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: !!! no internal routing support, rebuild with pcre support !!!
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: uWSGI running as root, you can use --uid/--gid/--chroot options
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: *** WARNING: you are running uWSGI as root !!! (use the --uid >
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: *** WARNING: you are running uWSGI without its master process >
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: your processes number limit is 3811
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: your memory page size is 4096 bytes
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: detected max file descriptor number: 1024
Dec 27 10:39:31 ip-172-31-35-69 systemd[1]: Started uWSGI Emperor service.
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: *** starting uWSGI Emperor ***
lines 1-20/20 (END)...skipping...
● uwsgi.service - uWSGI Emperor service
     Loaded: loaded (/etc/systemd/system/uwsgi.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-27 10:39:31 KST; 43min ago
   Main PID: 3364 (uwsgi)
     Status: "The Emperor is governing 0 vassals"
      Tasks: 1 (limit: 1143)
     Memory: 1.2M
     CGroup: /system.slice/uwsgi.service
             └─3364 /usr/local/bin/uwsgi --emperor /etc/uwsgi/sites

Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: detected binary path: /usr/local/bin/uwsgi
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: !!! no internal routing support, rebuild with pcre support !!!
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: uWSGI running as root, you can use --uid/--gid/--chroot options
Dec 27 10:39:31 ip-172-31-35-69 uwsgi[3364]: *** WARNING: you are running uWSGI as root !!! (use the --uid flag) ***
```

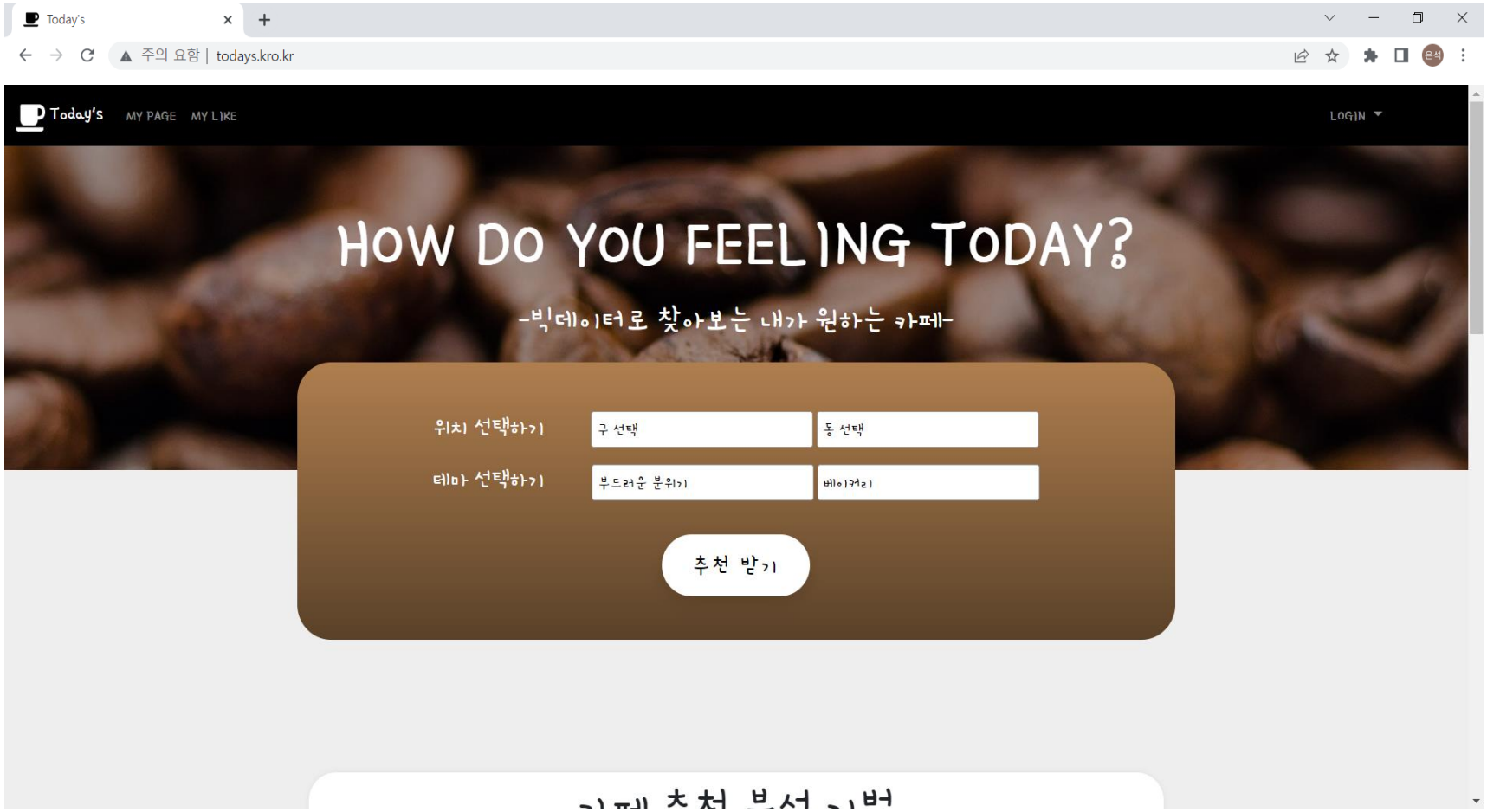
5) 배포

Web page 배포

웹사이트 배포 및 도메인 설정 진행

도메인 주소 : www.todays.kro.kr

고정 IP : 3.36.207.105



003 - Web

기술설명-시연

Part 4
Conclusion

004 Conclusion

결론 및 보완점

1. Framework 응용 능력
   - 장고를 조금만 더 익숙하게 다룰 수 있었으면 더 많은 기능을 구현할 수 있었을 것.
   - DB를 더 다양하게 활용할 수 있었을 텐데 json으로 데이터를 처리했던 점이 아쉽다.

2. 데이터 분석 환경
   - 대량의 데이터를 수집을 했지만 물리적인 한계(RAM 용량 한계 등)로 인한 방대한 데이터를 모두 사용하지 못한 점이 아쉽다
   - 시간이 더 있었다면 BERT 모델과 같은 고급 모델링을 통해 성능을 조금 더 끌어올릴 수 있었을 것

3. AWS를 활용한 배포
   - 배포를 하기 위한 준비 시간이 짧아서 모델까지 적용된 사이트를 구현하는 것에 어려움을 겪음
   - 처음이다 보니 예상치 못한 시행착오가 많았음

Part 5

QnA

Thank You
감사합니다