



분석보고서 (포트폴리오 용)

- 1. 프로젝트 개요
- 3. 프로젝트 수행 절차 및 방법
- 4. 프로젝트 수행 절차 및 전처리
- 5. 데이터 분석
- 6. 모델링 예측 분석
- 7. 인사이트 정리 및 결론
- 8. 프로젝트 회고 및 개선점
- 9. 부록

1. 프로젝트 개요

1-1. 주제



"커피템플 : 오프라인 성공을 온라인으로 이어나가는 전략"

- 메뉴 최적화
 - 연관 분석을 통한 세트 메뉴 구성
 - 온/오프라인 진열상품 최적화
- 트렌드 예측
 - 시계열 분석
 - 회귀 분석

1-2. 주제 선정의 배경

커피템플의 오프라인 매출은 준수하지만 온라인 매출이 상대적으로 낮은 상황으로 파악되고 있습니다.

또한, 대표님의 오프라인의 경험이 온라인 매출로 이어지지 않는다는 말씀에서 아이디어를 얻었습니다.

이에 따라 저희는 "오프라인의 특별한 경험이 온라인 매출으로 이어지게 하는 솔루션"을 제시하겠습니다.

커피템플은 제주 안에서 커피/카페로서 원앤온리 가치를 가지고 찾게 만들고자 하는 욕심이 있습니다.

무언가를 파는 것보다 좋은 원두를 제공하는 것에 가치를 둔다는 점을 강조하여 방향성을 유지하고자 합니다.

그리고 이를 통해 향후 고물가 시대 제주 방문객 감소의 리스크를 튼튼한 온라인 매출로 이겨낼 수 있는 환경을 조성할 수 있을거라 기대합니다.

1-3. 본 프로젝트의 활용 방안 제시

1. 오프라인 세트메뉴 추가구성

- 연관(장바구니)분석을 통해 관련이 깊은 메뉴들을 엮어 **세트메뉴**로 구성함으로써 고객들의 메뉴 선택 부담을 덜고, 다양한 디저트와 함께 즐기도록 유도함으로써 오프라인 매출 증진에 기여합니다.
- 향후 온라인 고객데이터를 분석해서 온라인에도 적용이 가능합니다.

2. 오프라인 진열상품 최적화

- 메뉴들을 더 효과적으로 소개하고 고객들의 관심과 선호도를 높이기 위해 사용합니다.
- 진열상품의 매출량을 기준으로 진열 구성을 재구성하여 매출 증진에 기여합니다.
- 진열상품의 관심을 온라인으로 손쉽게 가져오기 위해 QR코드를 메뉴 설명란과 매장 탁자에 명기해두어 직접 매장에서 사가지 않더라도 온라인 스토어로 스페셜티를 주문해 즐길 수 있는 편리성을 제공합니다.

3. 온라인 고객 타겟 분석 및 마케팅 제안

- 스마트스토어 이용 고객의 연령별, 성별로 CRM데실 분석을 적용하여 타겟 계층을 선정합니다.
- 주요 고객을 대상으로 한 마케팅을 제안하여, 충성고객 확보 및 온라인 매출 증진을 기대합니다.

이러한 방안들을 통해 커피템플은 온라인 매출을 증대시키는 동시에 브랜드 가치와 개성을 유지할 수 있습니다.

커피템플은 단순히 무언가를 파는 것이 아니라 **고객들에게 좋은 원두를 제공하는 커피/카페의 가치**를 강조함으로써, 고객들의 관심과 신뢰를 얻을 수 있을 것입니다.

3. 프로젝트 수행 절차 및 방법

3-1. 데이터 설명 (활용 데이터에 대해 조사한 지식을 바탕으로 작성해주세요.)

- 오프라인 (페이히어 결제시스템)
- 온라인 데이터
- ▼ 추가 데이터(날씨, 차량 수, 검색량)

▼ **날씨 데이터** → '일시', '평균기온', '강수량'

지점명	일시	평균기온(℃)	최고기온(℃)	최저기온(℃)	최저기온시각	일교차	평균습도(%rh)	최저습도(%rh)	강수량(mm)	1시간최다강수량(mm)	1시간최다강수량시각	
제주	2022-02-10	6.9	9.8	12:57	4.9	4:23	4.9	61.8	40	7.2	NaN	NaN
제주	2022-02-11	7.5	11.4	13:08	2.8	7:07	8.6	50.4	37	NaN	NaN	NaN
제주	2022-02-12	10.0	12.6	14:36	7.6	2:04	5.0	54.5	36	NaN	NaN	NaN
제주	2022-02-13	10.1	11.9	12:35	8.1	23:57	3.8	69.6	46	0.0	NaN	NaN
제주	2022-02-14	9.3	13.7	12:23	7.3	5:16	6.4	70.9	40	0.0	NaN	NaN
...
제주	2023-05-27	23.4	26.4	12:02	20.7	1:05	5.7	71.3	58	NaN	NaN	NaN
제주	2023-05-28	24.2	26.9	13:08	21.3	5:07	5.6	79.1	72	NaN	NaN	NaN
제주	2023-05-29	24.3	29.9	13:50	21.0	23:43	8.9	86.1	66	6.4	5.1	20:16
제주	2023-05-30	21.0	23.6	11:59	19.6	20:15	4.0	96.9	90	32.8	10.5	16:45
제주	2023-05-31	20.5	24.4	14:12	18.4	5:26	6.0	86.1	67	10.2	5.8	2:27

▼ **차량 데이터** → 카카오맵 '일별 도착 차량수'

	일자	차량 수
9	2022-02-10	15
10	2022-02-11	31
11	2022-02-12	40
12	2022-02-13	43
13	2022-02-14	19
...
480	2023-05-27	51
481	2023-05-28	53
482	2023-05-29	25
483	2023-05-30	43
484	2023-05-31	15

▼ 네이버 검색량 데이터 → '날짜', '검색량'

	날짜	검색량
9	2022-02-10	33.81147
10	2022-02-11	32.99180
11	2022-02-12	48.56557
12	2022-02-13	59.01639
13	2022-02-14	61.27049
...
480	2023-05-27	64.54918
481	2023-05-28	55.12295
482	2023-05-29	72.33606
483	2023-05-30	66.39344
484	2023-05-31	47.13114

4. 프로젝트 수행 절차 및 전처리

4-0. 팀원 별 역할분담

역할 분담 리스트

Aa Name	📅 Date	⋮ Tags
<u>데이터 전처리</u>		팀원_김예린
<u>데이터 시각화</u>		팀원_김예린
<u>모델링 - 회귀 분석</u>		팀원_김예린
<u>모델링 - 연관 분석</u>		팀원_김예린
<u>데이터 전처리</u>		팀원_정우용
<u>데이터 시각화</u>		팀원_정우용
<u>모델링 - 시계열 분석</u>		팀원_정우용
<u>기획서 작성</u>	@2023년 8월 5일 → 2023년 8월 7일	팀원_정우용

Aa Name	📅 Date	☰ Tags
<u>모델링 - 회귀분석</u>	@ 2023년 7월 29일 → 2023년 8월 6일	팀장_함승주
<u>대시보드 기획</u>		팀장_함승주
<u>팀 전원 역할 수행 관리</u>		팀장_함승주
<u>WBS 관리</u>		팀장_함승주
<u>데이터분석</u>		팀원_이윤지
<u>모델링 - 연관 분석</u>		팀원_이윤지
<u>대시보드 기획</u>		팀원_이윤지
<u>발표자료 작성</u>	@ 2023년 8월 5일 → 2023년 8월 7일	팀원_이윤지
<u>아이디어 회의</u>		공통
<u>자료조사</u>		공통
<u>데이터 탐색</u>		공통
<u>분석 보고서 작성</u>		공통
<u>제목 없음</u>		

4-1. 데이터 클리닝 계획

- 오프라인 데이터
 - 일/시간을 key값으로 정렬하여 카테고리, 상품명, 상품별 단가, 상품별 합계를 간단하게 전처리함
- 추가 데이터
 - 날씨, 차량, 검색량 데이터를 추가로 불러와서 기존 df와 merge 시킴

4-2. 데이터 샘플

▼ 🗄 Dataset

1. 오프라인 데이터

```
df_off = pd.read_excel('/content/offline.xlsx')
```

```
df_off.shape = (145010, 26)
```

2. 날씨 데이터

```
weather = pd.read_csv('/content/weather.csv', encoding='cp949')
weather['일시'] = pd.to_datetime(weather['일시']).dt.date
weather = weather[(weather['일시']>pd.to_datetime('2022-02-10')) &
                  (weather['일시']<pd.to_datetime('2023-05-31'))]
```

지점명	일시	평균기온(℃)	최고기온(℃)	최저기온(℃)	최저기온시각	일교차	평균습도(ℳh)	최저습도(ℳh)	강수량(mm)	1시간최다강수량(mm)	1시간최다강수량시각
40	제주 2022-02-10	6.9000	9.8000	12.57	4.9000	4.23	4.9000	61.8000	40	7.2000	NaN
41	제주 2022-02-11	7.5000	11.4000	13.08	2.8000	7.07	8.6000	50.4000	37	NaN	NaN
42	제주 2022-02-12	10.0000	12.6000	14.36	7.6000	2.04	5.0000	54.5000	36	NaN	NaN
43	제주 2022-02-13	10.1000	11.9000	12.35	8.1000	23.57	3.8000	69.6000	46	0.0000	NaN
44	제주 2022-02-14	9.3000	13.7000	12.23	7.3000	5.16	6.4000	70.9000	40	0.0000	NaN
...
511	제주 2023-05-27	23.4000	26.4000	12.02	20.7000	1.05	5.7000	71.3000	58	NaN	NaN
512	제주 2023-05-28	24.2000	26.9000	13.08	21.3000	5.07	5.6000	79.1000	72	NaN	NaN
513	제주 2023-05-29	24.3000	29.9000	13.50	21.0000	23.43	8.9000	86.1000	66	6.4000	5.1000
514	제주 2023-05-30	21.0000	23.6000	11.59	19.6000	20.15	4.0000	96.9000	90	32.8000	10.5000
515	제주 2023-05-31	20.5000	24.4000	14.12	18.4000	5.26	6.0000	86.1000	67	10.2000	5.8000

476 rows x 13 columns

3. 차량수 데이터

```
car = pd.read_csv('/content/car.csv', encoding='cp949')
car['일자'] = pd.to_datetime(car['일자']).dt.date
car = car[(car['일자']>=pd.to_datetime('2022-02-10')) &
          (car['일자']<=pd.to_datetime('2023-05-31'))]
```

일자 차량 수		
9	2022-02-10	15
10	2022-02-11	31
11	2022-02-12	40
12	2022-02-13	43
13	2022-02-14	19
...		
480	2023-05-27	51
481	2023-05-28	53
482	2023-05-29	25
483	2023-05-30	43
484	2023-05-31	15

476 rows × 2 columns

4. 검색량 데이터

```
naver = pd.read_excel('/content/datalab_220201-230531.xlsx')
naver['날짜'] = pd.to_datetime(naver['날짜']).dt.date
naver = naver[(naver['날짜']>=pd.to_datetime('2022-02-10')) &
              (naver['날짜']<=pd.to_datetime('2023-05-31'))]
```

날짜 검색량		
9	2022-02-10	33.8115
10	2022-02-11	32.9918
11	2022-02-12	48.5656
12	2022-02-13	59.0164
13	2022-02-14	61.2705
...		
480	2023-05-27	64.5492
481	2023-05-28	55.1230
482	2023-05-29	72.3361
483	2023-05-30	66.3934
484	2023-05-31	47.1311

476 rows × 2 columns

4-3. 데이터 수집 및 전처리

[오프라인 데이터 - 전처리]

▼ 결제일/결제시간 결측치 처리 후 정렬

- '결제일', '결제시간' Nan 값 인 경우 → [xxx 외 3건, 2건] 등의 형태
- 아래 2행, 3행, 4행의 결측치를 맨위 행으로 대체

```
df_off['결제일'].fillna(method='ffill', inplace=True)
df_off['결제시간'].fillna(method='ffill', inplace=True)

df_off['결제일'] = pd.to_datetime(df_off['결제일']).dt.date
df_off.sort_values(by=['결제일', '결제시간'], inplace=True)
df_off.reset_index(drop=True, inplace=True)
```

▼ 환불 건 제거 `df_off2`

```
# 상품별 합계가 '-' 인 행은 환불 건
df_off2 = df_off[df_off['상품별 합계'] != '-']
```

▼ 상품별 단가 0, 1 제거

```
# 상품별 단가가 0인 것 중에 어린이 우유를 제외하고 제거
df_off2 = df_off2[(df_off2['상품별 단가'] != 0) |
                  (df_off2['상품명'].str.contains('어린이 우유'))]

# 상품별 단가가 1인 것 제거
df_off2 = df_off2[df_off2['상품별 단가'] != 1]

# 사업자 카테고리 제거
df_off2 = df_off2[df_off2['카테고리'] != '사업자']
```

▼ 포장, 커스텀 카테고리 설정

- 보냉백, 종이백, 종이백_고급 → 포장
- 샷추가, 바닐라시럽, 오토리, 오토사이드 → 커스텀

```
df_off2.loc[df_off2['상품명']=='샷추가', '상품명'] = '1샷 추가'
df_off2.loc[df_off2['상품명']=='바닐라 시럽', '상품명'] = '바닐라시럽'

df_off2.loc[df_off2['상품별 단가']==100, '카테고리'] = '포장'
df_off2.loc[df_off2['상품별 단가']==500, '카테고리'] = '커스텀'
df_off2.loc[df_off2['상품별 단가']==1000, '카테고리'] = '커스텀'
df_off2.loc[df_off2['상품별 단가']==2000, '카테고리'] = '포장'
df_off2.loc[df_off2['상품명']=='보냉백', '카테고리'] = '포장'
```

▼ 디저트 상품명 이상치 처리

결제일		결제시간	주문	재분	결제내역	합계	상품명	할인	결제	할인	카드	결제	현금	결제	간편	결제	...	환불	환불	일시	배달당(배출 포함)	카테고리	상품명	옵션	수량	상품별	단가	상품별	합계	결제	메모
55031	2022-07-27	10:08:45	결제	○ 외 3건	23500	-	-	23500	-	-	-	-	-	-	-	-	-	-	-	-	-	디저트	○	-	1	3000	3000	-	-	-	
55043	2022-07-27	10:11:51	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	디저트	○	-	1	3000	3000	NaN	NaN	NaN	

2 rows × 25 columns

2 rows × 25 columns

	결제일	결제시간	주문	재	결제내역	합계	상품별	할인	결제	할인	카드	결제	현금	결제	간편	결제	...	환불	환불	일시	배달당(배출 포함 x)	카테고리	상품	옵션	수량	상품별	단가	상품별	합계	결제	메모
55040	2022-07-27	10:11:09	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	디저트	○	-	1	3300	3300	NaN	NaN	NaN	
55045	2022-07-27	10:13:10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	디저트	○	-	1	3300	3300	NaN	NaN	NaN	

2 rows x 25 columns

2 rows × 25 columns

```
df_off2.loc[df_off2['상품명']=='○', '상품명'] = '플레이н 휘낭시에'
df_off2.loc[df_off2['상품명']=='○', '상품명'] = '무화과 휘낭시에'
```

▼ 판매상품/etc 분류 → 컬럼 생성

```
# 따로 구매한 것이 아닌 것 etc로 분류
etc = ['종이백', '종이백', '종이백_고급',
       '오토리', '바닐라 시럽', '오토사이드', '바닐라 시럽',
       '샷추가', '1샷 추가', '2샷 추가', '보냉백']
df_off2['분류'] = df_off2['상품명'].apply(lambda x: 'etc' if x in etc else '판매상품')
```

▼ Hot/Ice 구분 → 컬럼 생성

```
df_off2['Hot/Ice'] = df_off2.apply(lambda row:
    'Hot' if row['카테고리'] == 'Basic' else
    'Ice' if row['카테고리'] == 'Basic _ ice' else
    'Ice' if (row['카테고리'] == '비버리지' and
              any(word in row['상품명'] for word in ['(I)', '아이스', '차가운', '주스'])) else
    'Hot' if (row['카테고리'] == '비버리지' and
              any(word not in row['상품명'] for word in ['(I)', '아이스', '차가운', '주스'])) else
    'Hot' if (row['상품명'] == '공연 텐더린카푸치노') else
    'Ice' if ('아이스' in row['상품명']) else
    'Ice' if (row['카테고리'] == '세트' and '(I)' in row['상품명']) else
    'Hot' if (row['카테고리'] == '세트' and '(I)' not in row['상품명']) else
    'Ice' if (row['카테고리'] == '시그니처' and '아이스' in row['상품명']) else
    'Hot' if (row['카테고리'] == '시그니처' and '아이스' not in row['상품명']) else
    'Hot' if row['카테고리'] == '에스프레소' else
    'Ice' if (row['카테고리'] == '핸드드립' and '(I)' in row['상품명']) else
```

```
'Hot' if (row['카테고리'] == '핸드드립' and '(I)' not in row['상품명']) else
'해당없음', axis=1)
```

▼ 불필요 컬럼 drop

```
df_off2.drop(['Unnamed: 0', '주문 채널', '결제내역', '합계', '상품별 할인',
              '결제 할인', '카드 결제', '현금 결제', '간편 결제', '기타 결제',
              '온라인 스토어', '사용 포인트', '적립 포인트', '사용 선불권',
              '환불', '환불 일시', '배달팁(매출 포함x)', '결제메모'],
             axis=1, inplace=True)
```

▼ '미분류' 카테고리 ⇒ 'MD'

```
# 미등록된 MD 상품
df_off2.loc[df_off2['카테고리']=='미분류', '카테고리'] = 'MD'
```

▼ 옵션 Hot/Ice 값 반영 후 drop

```
df_off2.loc[df_off2['옵션']=='Ice', 'Hot/Ice'] = 'Ice'
df_off2.loc[df_off2['옵션']=='Hot', 'Hot/Ice'] = 'Hot'
df_off2.loc[df_off2['옵션']=='연하게, 설탕시럽, Hot', 'Hot/Ice'] = 'Hot'

df_off2.drop(['옵션'], axis=1, inplace=True)
```

▼ 원두 구분 → 컬럼 생성

- 주시, 디카프, 클래식, 싱글, 게바(게스트 바리스타) + 해당없음

```
df_off2['원두'] = '싱글'

df_off2.loc[df_off2['카테고리'] == '시그니처', '원두'] = '주시'
df_off2.loc[df_off2['상품명'].str.contains('디카프'), '원두'] = '디카프'
df_off2.loc[df_off2['상품명'].str.contains('클래식'), '원두'] = '클래식'
df_off2.loc[df_off2['상품명'].str.contains('주시'), '원두'] = '주시'
df_off2.loc[df_off2['상품명'].str.contains('댄싱|이터널|레전드|시티트래블러|노스텔리아|팍트로 콜롬비아'), '원두'] = '게바'

unknown_bean = df_off2['카테고리'].isin(['MD', '포장', '디저트', '커스텀', '비버리지'])
df_off2.loc[unknown_bean, '원두'] = '해당없음'
```

▼ 상품명 일반화(정규표현식) df_off3

```
df_off3 = df_off2.copy()

# 정규 표현식 사용
import re
df_off3['상품명'] = df_off2['상품명'].str.replace(r'\\(I\\)', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'\\(H\\)', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'아이스', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'차가운', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'따뜻한', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'주시', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'클래식', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'디카프', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'싱글', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'_', '')
df_off3['상품명'] = df_off3['상품명'].str.replace(r'^\s+|\s+$', '', regex=True)
```

```
df_off3.loc[df_off3['상품명'] == '텐라', '상품명'] = '텐저린 라떼'
df_off3.loc[df_off3['상품명'] == '텐저린카푸치노', '상품명'] = '텐저린 카푸치노'

df_off3.loc[df_off3['상품명'] == '바닐라 라떼', '상품명'] = '바닐라라떼'
df_off3.loc[df_off3['상품명'] == '(템플) 플랫', '상품명'] = '(템플)플랫'

df_off3.loc[df_off3['상품명'] == 'Set. 아메리카노', '상품명'] = 'Set.아메리카노'
df_off3.loc[df_off3['상품명'] == 'Set. 카페라떼', '상품명'] = 'Set.카페라떼'
```

```
# 유자 아메리카노 - Only Ice
df_off3.loc[df_off3['상품명'] == '유자 아메리카노', 'Hot/Ice'] = 'Ice'

# 카테고리 '핸드드립'에 Null값 -> 오늘의 커피로 대체
df_off3.loc[119675, '상품명'] = '오늘의 커피'

# 카테고리 '싱글원두'에 상품명 '-' 제거
df_off3 = df_off3[~df_off3['상품명'].str.contains('-')]
```

▼ 결제일_년월 → 컬럼 생성 (시각화용)

```
df_off3['결제일'] = pd.to_datetime(df_off3['결제일'])
df_off3['결제일_년월'] = df_off3['결제일'].dt.strftime('%y-%m')
```

▼ 일별 수량, 일별 매출 → 컬럼 생성

```
day_count = df_off3.groupby('결제일')['수량'].sum().reset_index()
day_sum = df_off3.groupby('결제일')['상품명_합계'].sum().reset_index()

day_count.rename(columns={'수량': '일별 수량'}, inplace=True)
day_sum.rename(columns={'상품명_합계': '일별 매출'}, inplace=True)

df_off3 = pd.merge(df_off3, day_count[['결제일', '일별 수량']],
                  left_on='결제일', right_on='결제일', how='left')
df_off3 = pd.merge(df_off3, day_sum[['결제일', '일별 매출']],
                  left_on='결제일', right_on='결제일', how='left')
```

[추가 데이터 - 전처리]

▼ 날씨 데이터 : '일시', '평균기온', '강수량'

```
# 강수량 결측치 0으로 대체
weather['강수량(mm)'].fillna(0, inplace=True)

df_off2 = pd.merge(df_off2, weather[['일시', '평균기온(°C)', '강수량(mm)']],
                  left_on='결제일', right_on='일시', how='left')
df_off2 = df_off2.drop(columns=['일시'])
```

▼ 차량 데이터 (카카오맵 일별 도착 차량수) : '일자', '차량 수'

```
df_off2 = pd.merge(df_off2, car[['일자', '차량 수']],
                  left_on='결제일', right_on='일자', how='left')
df_off2 = df_off2.drop(columns=['일자'])
```

▼ 네이버 검색량 데이터 : '날짜', '검색량'

```
df_off2 = pd.merge(df_off2, naver[['날짜', '검색량']],
                  left_on='결제일', right_on='날짜', how='left')
df_off2 = df_off2.drop(columns=['날짜'])
```

[오프라인 데이터 - 최종]

- offline_0808.xlsx

[온라인 데이터]

- [원두], [성별], [연령], [결제금액] → 통계

4-4. 활용 라이브러리 등 기술적 요소

- ☒ pandas
- ☒ numpy
- ☒ matplotlib (koreanize_matplotlib)
- ☒ seaborn
- ☒ plotly
- ☐ cufflinks
- ☐ folium
- ☒ 워드클라우드 : konlpy, platform, IPython, wordcloud, PIL,
- ☒ 연관분석 : apriori
- ☒ 모델링 : scikit learn, lightgbm, catboost
- ☒ 기타 : statsmodels, scipy, Tableau, Excel

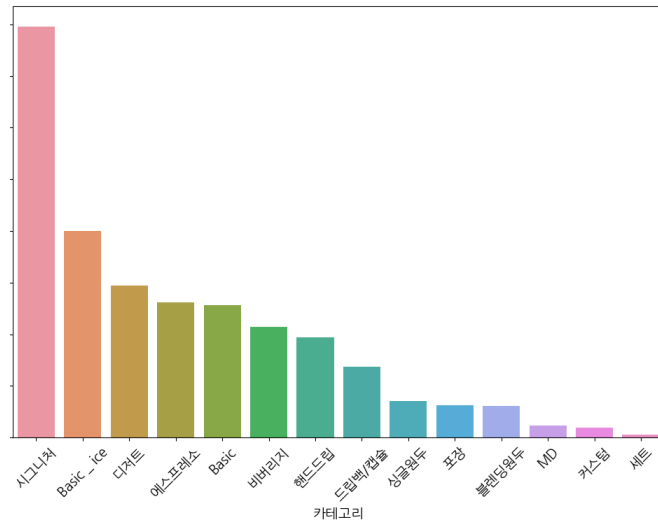
4-5. 프로젝트에서 분석한 내용

- ☒ 결측치 확인
- ☒ 중복값 확인
- ☒ 데이터 타입 확인
- ☒ 이상치 확인
- ☒ 전체 수치 변수의 히스토그램 그리기
- ☒ 수치 데이터 기술 통계 구하기
- ☒ 범주 데이터 기술 통계 구하기
- ☒ 파생변수 만들기
- ☒ 데이터프레임 병합
- ☒ 상관관계 구하기
- ☒ 빈도수 구하기
- ☒ groupby, pivot_table 등을 통한 데이터 집계
- ☐ 기타:

4-6. EDA 결과물

[오프라인 데이터]

1. 카테고리별 주문 수



그래프를 보면 **시그니처**의 판매량이 가장 많은 것을 알 수 있습니다.

고객들은 다른 매장엔 없는 특별한 커피를 경험하고 싶어한다는 것을 쉽게 파악할 수 있었습니다.

▼ 코드

```
plt.figure(figsize=(12, 8))
total_count = len(df_off3) # 전체 행 수

sns.countplot(x='카테고리', data=df_off3, order=df_off3['카테고리'].value_counts().index)

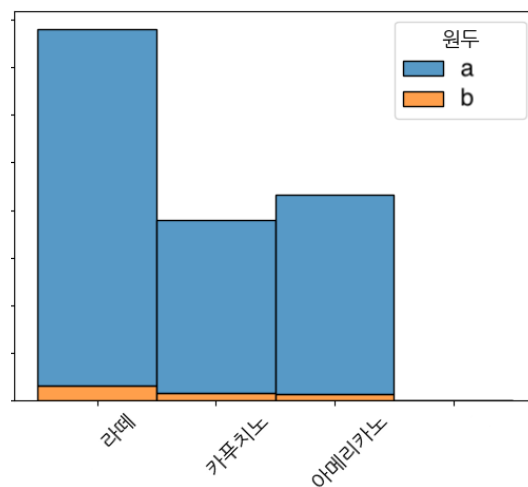
plt.xticks(rotation=45)
plt.show()
```

2. 카테고리 별 원두 비율

시그니처 커피를 마시는 사람들의 원두 선택 비율을 보여줍니다.

시그니처 커피의 기본 원두인 **a 원두**의 비율이 큰 것을 알 수 있고,

간혹 **b 원두**를 찾는 고객들도 있다는 점을 알 수 있습니다.

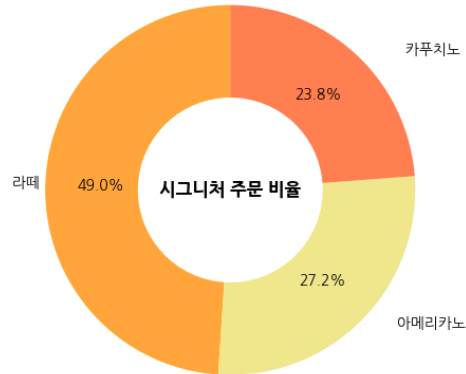


▼ 코드

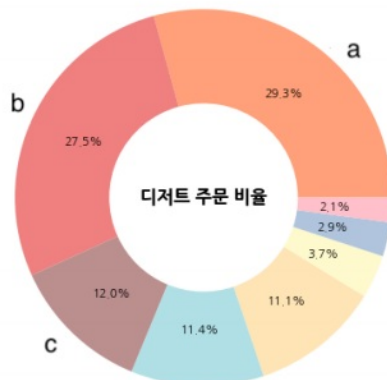
```
sns.histplot(x='상품명', data=df_off3[df_off3['카테고리'] == '시그니처'],
             hue='원두', multiple='stack')
```

```
plt.xticks(rotation=45)
```

3. 카테고리별 주문수 비율



시그니처 메뉴 중 **라떼**의 비율이 거의 절반에 달하는 모습을 볼 수 있습니다.
 라떼와 아메리카노는 **아이스** 메뉴이기 때문에
 시그니처 메뉴 중 시원한 음료를 선호하는 고객들이 많다는 것을 시사합니다.
 반면, 10명 중 2명 정도만 따뜻한 카푸치노를 즐깁니다.



디저트 메뉴의 판매 비율은 어떻게 될까요?

a와 **b**가 가장 높은 비중을 차지하는 것을 확인할 수 있습니다.

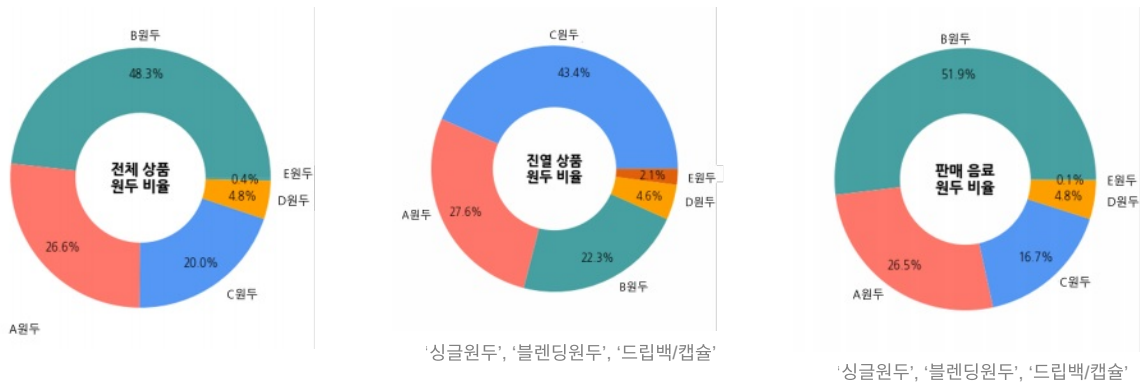
나중에 메뉴 연관분석을 진행하면 디저트와 커피의 관계를 분석해봐도 좋겠다는 생각이 듭니다.

▼ 코드

```
df_dessert = df_off3[df_off3['카테고리'] == '디저트']
product_counts = df_dessert['상품명'].value_counts()

plt.figure(figsize=(8, 5))
plt.pie(product_counts, labels=product_counts.index,
        autopct='%1.1f%%', textprops={'fontsize': 8}, startangle=0,
        # shadow=True, explode=(0.1, 0.1, 0, 0, 0, 0, 0, 0),
        pctdistance=0.7, labeldistance=1.03,
        wedgeprops=dict(width=0.5),
        colors=['lightsalmon', 'lightcoral', 'rosybrown', 'powderblue',
               'moccasin', 'lemonchiffon', 'lightsteelblue', 'pink'])
plt.title('디저트 카테고리의 상품명 별 주문수')
plt.axis('equal')
plt.show()
```

4. 원두 비율



전체상품, 진열된 상품, 판매 커피에 대한 원두 별 판매 비율을 분석했습니다.

전체상품과 판매 커피는 **b 원두**가 가장 잘 나가고,

진열상품은 **c 원두**가 가장 잘 판매되는 모습을 보입니다.

나중에 진열상품 관련 분석을 진행할 때 활용이 가능해 보입니다.

▼ 코드

```
plt.pie(df_off2[df_off2['원두']!='해당없음']['원두'].value_counts(),
       labels=df_off2[df_off2['원두']!='해당없음']['원두'].value_counts().index,
       autopct='%1.1f%%', explode=(0.1, 0, 0, 0, 0), shadow=True, startangle=90,
       colors=['cadetblue', 'salmon', 'cornflowerblue', 'orange'])

plt.title('전체 상품에 대한 원두 비율')
```

```
bean = df_off2[df_off2['카테고리'].isin(['싱글원두', '블렌딩원두', '드립백/캡슐'])]

plt.pie(bean['원두'].value_counts(), labels=bean['원두'].value_counts().index,
       autopct='%1.1f%%', explode=(0.1, 0, 0, 0, 0), shadow=True, startangle=90,
       colors=['cornflowerblue', 'salmon', 'cadetblue', 'orange', 'chocolate'])

plt.title('진열 상품에 대한 원두 비율')
```

```
bean2 = df_off2[df_off2['카테고리'].isin(['시그니처', 'Basic _ ice', 'Basic', '핸드드립', '에스프레소', '세트'])]

plt.pie(bean2[bean2['원두']!='해당없음']['원두'].value_counts(),
       labels=bean2[bean2['원두']!='해당없음']['원두'].value_counts().index,
       autopct='%1.1f%%', explode=(0.1, 0, 0, 0, 0), shadow=True, startangle=90,
       colors=['cadetblue', 'salmon', 'cornflowerblue', 'orange'])

plt.title('판매 음료에 대한 원두 비율')
```

5. 기온에 따른 Hot/Ice 매출 추이

막대 그래프가 해당 월의 평균 기온을 나타냅니다.

특히, 22년 12월과 23년 1월, 2월의 그래프가 눈에 띕니다.

추워지니 따뜻한 음료의 판매량이 늘고, 차가운 음료의 판매량이 줄어드는 모습을 보입니다.

기온이 약 영상 25도 이상으로 올라가면 Hot의 판매량이 매우 저조한 모습을 보입니다.

기온이 약 영상 10도 이하로 떨어지면 Ice의 판매량이 적어지는 모습입니다.

- 22년 3월~12월까지의 ICE 주문량은 기온의 증가와 하락에 따른 선호도를 잘 보여주지 못합니다.
→ 이는 일정한 매출이 발생하지 않고 지속적으로 감소하였기 때문이라고 생각합니다
- 날씨 변화에 따른 hot ice 주문 매출 추이
→ HOT에 대한 매출을 보는 것이 조금 더 정확 할 수 있다고 봅니다.
- 22년 7월과 8월에 최저점을 형성하고 23년 1월까지 기온이 내려감에 따라 지속적인 hot 증가를 보입니다.
→ 얼죽아 민족인 한국에서는 날씨와 HOT/ICE 의 뚜렷한 주문 구분을 하기 어렵다고 판단됩니다.

즉, Hot은 **기온의 영향**을 많이 받는 반면, Ice는 기온의 영향을 덜 받는 **개인의 취향**으로 비춰집니다.

이를 염두에 두고 매장 운영을 하시면 더 효율적으로 매출을 관리할 수 있어보입니다.

▼ 코드

```
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

def percentage(x, pos):
    return f'{0.000001 * x:.1f}%'

plt.figure(figsize=(14, 6))

ax1 = df_off2.groupby('결제일_년월')['평균기온(°C)'].mean().plot(kind='bar',
    color='gray', label='평균기온(°C)')

subset_hot = df_off2[df_off2['Hot/Ice']=='Hot']
subset_ice = df_off2[df_off2['Hot/Ice']=='Ice']

ax2 = subset_hot.groupby('결제일_년월')['상품별 합계'].sum().plot(
    secondary_y=True, marker='o', color='r', label='Hot')
ax3 = subset_ice.groupby('결제일_년월')['상품별 합계'].sum().plot(
    secondary_y=True, marker='o', color='b', label='Ice')

plt.title('평균 기온에 따른 년월별 Hot & Ice 매출 추이')
plt.xlabel('년월')

# y축 레이블 변환
formatter = FuncFormatter(percentage)
plt.gca().yaxis.set_major_formatter(formatter)

plt.xticks(rotation=45)
plt.legend()
plt.show()
```

6. 원두 상품 주요 키워드 - 워드 클라우드



온/오프라인의 진열 판매 상품명의 언급 건수를 기반으로 워드 클라우드를 만들었습니다.

많이 언급된 상품일수록 글씨가 큼니다.

오프라인 진열대에서는 게이샤, 온라인 스토어에서는 블렌딩 상품이 많이 팔린 것으로 보입니다.

▼ 코드

```
wordcloud = WordCloud(width=800, height=400, background_color='white',
                        font_path = font_path, colormap='plasma'
                        ).generate_from_text(' '.join(bean_name))
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('오프라인 매장 - 원두')
plt.show()
```

```
import PIL
icon = PIL.Image.open('/content/coffee_.png')

img = PIL.Image.new('RGB', icon.size, (255,255,255))
img.paste(icon, icon)
img = np.array(img)

wordcloud = WordCloud(width=800, height=400, background_color='white',
                        font_path = font_path, mask=img, colormap='plasma'
                        ).generate_from_text(' '.join(on_bean))

plt.figure(figsize=(8, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('온라인 매장 - 원두')
plt.show()
```

7. 매출 추이에 따른 차량수와 검색량과의 관계

‘커피템플’ 키워드에 대한 네이버 **검색량** 추이와 카카오톡 기반의 커피템플에 도착하는 **차량수**의 추이를 보여줍니다. 그리고 막대그래프는 매출의 추이를 보여줍니다.

검색량과 차량 도착수의 증감 추이가 매출의 증감 추이와 비슷한 것으로 파악되어,
향후 매출 예측 모델링을 할 때 지표로서 활용이 가능해보입니다.

▼ 코드

```
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

def percentage(x, pos):
    return f'{0.0001 * x:.1f}%'

plt.figure(figsize=(13, 6))

ax1 = df_off2.groupby('결제일_년월')['상품별_합계'].count().plot(kind='bar',
    color='gray', label='상품별_합계')

ax2 = df_off2.groupby('결제일_년월')['차량 수'].sum().plot(
    secondary_y=True, marker='o', color='pink', label='차량 수')

ax3 = df_off2.groupby('결제일_년월')['검색량'].sum().plot(
    secondary_y=True, marker='o', color='purple', label='검색량')

plt.title('년월별 매출 합계')
plt.xlabel('년월')

# y축 레이블 변환
formatter = FuncFormatter(percentage)
plt.gca().yaxis.set_major_formatter(formatter)

plt.ylabel('상품별_합계')
plt.xticks(rotation=45)
plt.legend()
plt.show()
```

일별 수량 / 일별 매출 column 추가

```

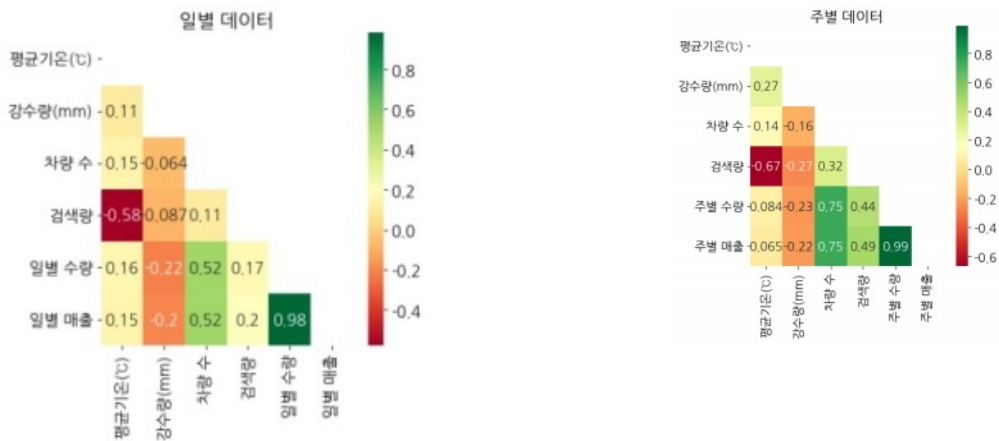
day_count = df_off3.groupby('결제일')['수량'].sum().reset_index()
day_sum = df_off3.groupby('결제일')['상품별 합계'].sum().reset_index()

day_count.rename(columns={'수량': '일별 수량'}, inplace=True)
day_sum.rename(columns={'상품별 합계': '일별 매출'}, inplace=True)

df_off3 = pd.merge(df_off3, day_count[['결제일', '일별 수량']],
                   left_on='결제일', right_on='결제일', how='left')
df_off3 = pd.merge(df_off3, day_sum[['결제일', '일별 매출']],
                   left_on='결제일', right_on='결제일', how='left')

```

8. 변수 간 상관관계(히트맵)



각 외부데이터와 내부데이터의 특성 별 상관관계를 살펴봤습니다.

역시 일별 수량(주문수)와 일별 매출과는 매우 깊은 상관관계를 보입니다.

차량 도착 수와도 꽤 유의미한 상관관계를 보입니다.

의외로 검색량과 평균기온 간에도 높은 음의 상관관계를 보여줍니다.

그래서 뒤에서 EDA를 추가로 해보겠습니다.

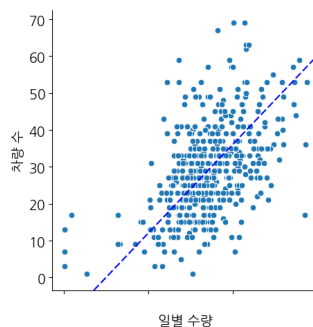
▼ 코드

```

plt.figure(figsize=(8, 5))
sns.heatmap(df_off3.corr(), annot=True, cmap='RdYlGn')

```

9. 차량 수에 따른 일별 주문 수량



'카카오맵 기반의 커피템플에 도착하는 차량수'와 '주문수량'의 상관관계를 보다 깊게 살펴봤습니다.

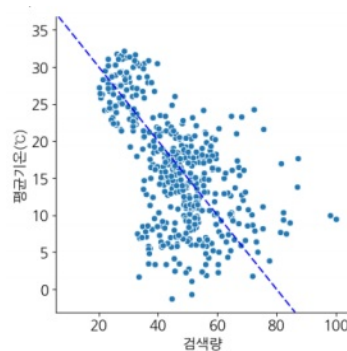
- 차량 수가 늘어날수록, 주문수도 많아지는 추세를 보입니다.

차량을 타고 오는 손님이 대부분인 매장으로, 매출과 직결되는 사실을 확인할 수 있습니다.

▼ 코드

```
g = sns.relplot(data=df_off3, x="일별 수량", y="차량 수")
g.ax.axline(xy1=(100, 0), slope=.12, color="b", dashes=(5, 2))
```

10. 커피템플 네이버 검색량과 평균 기온의 관계

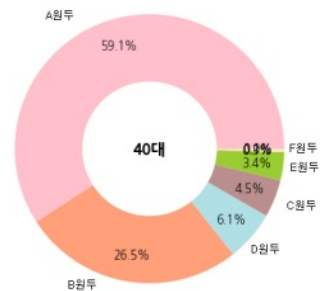
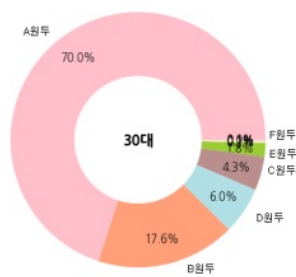
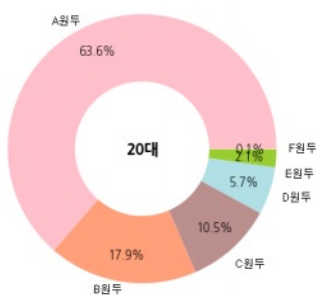


- 평균기온이 내려갈수록 검색량이 늘어가는 추세를 보입니다.

▼ 코드

```
sns.jointplot(data=df_off3, x="검색량", y="평균기온(°C)")
```

11. 온라인 스마트스토어 연령별 원두 선호도



나이가 들수록 공통적으로

- a 원두 선호비율 감소
- b 원두 선호비율 증가
- c 원두 선호비율 감소

하는 것으로 보입니다.

▼ 코드

```
c20 = custom[(custom['연령대']=='20~25')|(custom['연령대']=='26~30')].groupby('분류')['매출액'].sum()
plt.pie(c20.sort_values(ascending=False), labels=c20.sort_values(ascending=False).index,
```



```

        autopct='%1.1f%%', textprops={'fontsize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'rosybrown', 'powderblue', 'yellowgreen',
                'moccasin', 'lemonchiffon', 'lightsteelblue']
    )

plt.text(0, 0, '20대', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')

```

```

c30 = custom[(custom['연령대']=='31~35')|(custom['연령대']=='36~40')].groupby('분류')['매출액'].sum()

plt.pie(c30.sort_values(ascending=False), labels=c30.sort_values(ascending=False).index,
        autopct='%1.1f%%', textprops={'fontsize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'powderblue', 'rosybrown', 'yellowgreen',
                'moccasin', 'lemonchiffon', 'lightsteelblue']
    )

plt.text(0, 0, '30대', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')

```

```

c40 = custom[(custom['연령대']=='41~45')|(custom['연령대']=='46~50')].groupby('분류')['매출액'].sum()

plt.pie(c40.sort_values(ascending=False), labels=c40.sort_values(ascending=False).index,
        autopct='%1.1f%%', textprops={'fontsize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'powderblue', 'rosybrown', 'yellowgreen',
                'moccasin', 'lemonchiffon', 'lightsteelblue']
    )

plt.text(0, 0, '40대', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')

```

5. 데이터 분석

5-1. 연관 분석을 통한 세트 메뉴 구성

5-2. 오프라인 진열상품 최적화

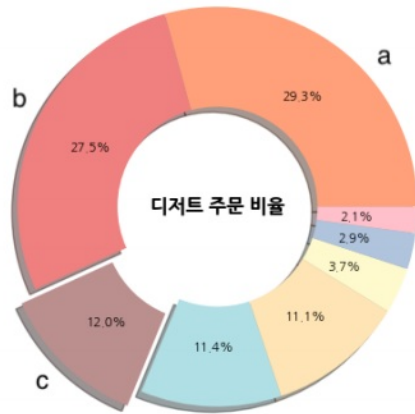
5-3. 온라인 고객 타겟 분석 및 마케팅 제안

5-1. 연관 분석을 통한 세트 메뉴 구성

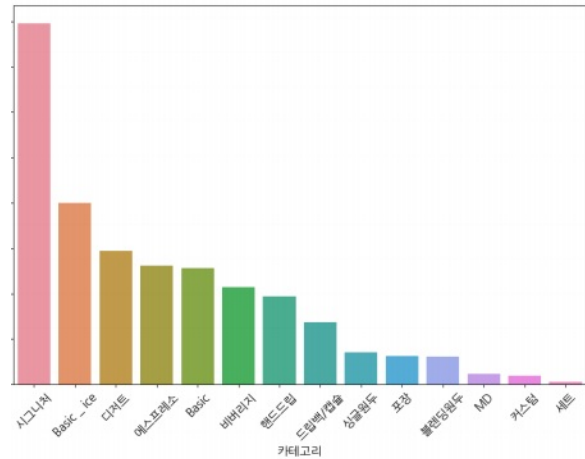
세트 메뉴의 가장 큰 장점은 메뉴를 개별적으로 구매했을 때보다 약간 더 저렴한 가격에 구입할 수 있는 점입니다. 이러한 이유로 세트 메뉴는 합리적인 소비로 간주됩니다. 세트 메뉴를 선택함으로써 소비자는 더 많은 가치를 누릴 뿐만 아니라 지출을 효율적으로 관리할 수 있습니다.

이러한 소비 방식은 매우 설득력 있습니다. 소비자는 동일한 맛과 만족감을 느끼면서도 자신의 예산을 더 효과적으로 활용할 수 있습니다. 이로 인해 많은 사람들이 세트 메뉴를 선호하게 되며, 이는 결국 가게의 매출을 증가시키는 유리한 전략이 될 것입니다.

- 기존 세트 메뉴 = C + 커피 (오전 9시~10시 판매/2000원 할인)



디저트 인기 순위 : 1위 a, 2위 b, 3위 c

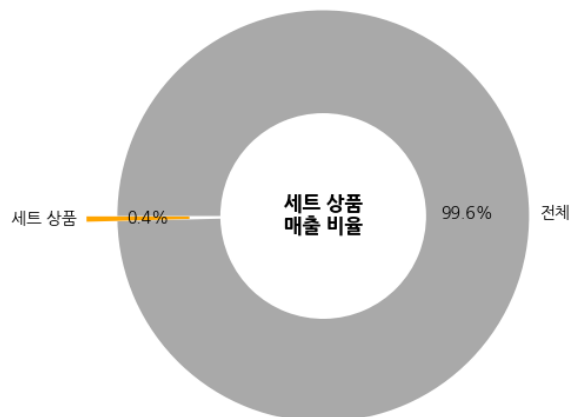


음료 카테고리별 인기 순위: 1위 시그니처, 2위 Basic_ice

위 그래프에서 디저트 카테고리에서 c의 인기 순위는 3위이고,

음료도 Basic/Basic_ice보다는 시그니처가 훨씬 더 선호도가 높은 것을 확인할 수 있습니다.

또한 기존 세트 메뉴의 매출은 전체 매출의 0.4% 밖에 안된다는 것을 확인하였습니다.



▼ 코드

```
# 전체 매출 계산
전체_매출 = df[df['카테고리'].isin(['Basic', 'Basic _ ice', '디저트', '바버라지', '시그니처', '에스프레소', '핸드드립'])['상품별 합계'].sum()

# 세트 상품 매출 계산
세트_상품_매출 = df[df['카테고리'] == '세트']['상품별 합계'].sum()

# 파이 그래프 그리기
labels = ['전체 매출', '세트 상품 매출']
sizes = [전체_매출, 세트_상품_매출]
colors = ['darkgray', 'orangered']
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%',
        explode=(0.1, 0), shadow=True, startangle=140)
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.
plt.title('전체 매출과 세트 상품 매출 비율')
plt.show()
```



현황분석

1. 기존 세트 메뉴의 추가/변경이 필요합니다.
2. 기존 세트 메뉴의 경우 2인이 주문하기 어렵습니다.
3. 음료 2잔 혹은 음료 2잔 + 디저트 1개 구성의 세트 메뉴를 제안할 수 있습니다.

apriori 모델

- 지지도 0.06 이상 설정 → 상품 및 조합 17건 추출

결제일시	상품명
2022-02-10 11:39:20	얼그레이 밀크티
2022-02-10 11:39:20	아메리카노
2022-02-10 11:39:20	브라우니
2022-02-10 12:02:12	아메리카노
2022-02-10 12:02:12	잠봉빅르
2022-02-10 12:02:38	브라우니
2022-02-10 12:05:23	플랫화이트
2022-02-10 12:05:23	잠봉빅르
2022-02-10 12:51:06	시그니처 카푸치노
2022-02-10 12:51:06	시그니처 라떼

주문내역에 디저트가 한 개라도 포함된 결제일
시만 추출하여 분석을 진행하였습니다.

	support	itemsets
1	0.3564	(바스크 치즈케이크)
2	0.3352	(브라우니)
7	0.3133	(시그니처 라떼)
3	0.3020	(아메리카노)
8	0.1696	(시그니처 카푸치노)
4	0.1635	(시그니처 아메리카노)
5	0.1461	(잠봉빅르)
9	0.1388	(플레인 휘낭시에)
0	0.1346	(무화과 휘낭시에)
11	0.1169	라떼, 바스크 치즈케이크
13	0.1129	(아메리카노, 브라우니)
10	0.1097	(아메리카노, 바스크 치즈케이크)
14	0.1073	(시그니처 라떼, 브라우니)
6	0.1063	(카페라떼)
16	0.0737	(시그니처 라떼, 시그니처 아메리카노)
15	0.0682	(시그니처 라떼, 아메리카노)
12	0.0617	(시그니처 카푸치노, 바스크 치즈케이크)

association_rules 규칙 추출

- 리프트 값 1 이상 설정, 내림차순 정렬 → 12건 추출

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	shangs_metric
0	(시그니처 라떼)	(시그니처 아메리카노)	0.3133	0.1635	0.0737	0.2352	1.4384	0.0225	1.0937	0.4438
1	(시그니처 아메리카노)	(시그니처 라떼)	0.1635	0.3133	0.0737	0.4506	1.4384	0.0225	1.2500	0.3643
2	(브라우니)	(아메리카노)	0.3352	0.3020	0.1129	0.3370	1.1156	0.0117	1.0527	0.1559
3	(아메리카노)	(브라우니)	0.3020	0.3352	0.1129	0.3739	1.1156	0.0117	1.0619	0.1485
4	(시그니처 라떼)	(바스크 치즈케이크)	0.3133	0.3564	0.1169	0.3732	1.0471	0.0053	1.0268	0.0656
5	(바스크 치즈케이크)	(시그니처 라떼)	0.3564	0.3133	0.1169	0.3281	1.0471	0.0053	1.0220	0.0699
6	(시그니처 라떼)	(브라우니)	0.3133	0.3352	0.1073	0.3425	1.0219	0.0023	1.0112	0.0312
7	(브라우니)	(시그니처 라떼)	0.3352	0.3133	0.1073	0.3202	1.0219	0.0023	1.0101	0.0322
8	(시그니처 카푸치노)	(바스크 치즈케이크)	0.1696	0.3564	0.0617	0.3638	1.0208	0.0013	1.0117	0.0246
9	(바스크 치즈케이크)	(시그니처 카푸치노)	0.3564	0.1696	0.0617	0.1731	1.0208	0.0013	1.0043	0.0317
10	(아메리카노)	(바스크 치즈케이크)	0.3020	0.3564	0.1097	0.3632	1.0193	0.0021	1.0108	0.0271
11	(바스크 치즈케이크)	(아메리카노)	0.3564	0.3020	0.1097	0.3079	1.0193	0.0021	1.0084	0.0294

순위	💖 베스트 커플
1	시그니처 아메리카노 - 시그니처 라떼
2	브라우니 - 아메리카노
3	바스크 치즈케이크 - 시그니처 라떼
4	브라우니 - 시그니처 라떼

위의 순서대로 메뉴를 같이 주문하는 경향이 있습니다.



시각화하면 왼쪽의 그림과 같습니다. (↔: 상품 간 연관성)

이를 통해 세트 메뉴를 제안합니다.

세트 A: 시그니처 라떼 + 시그니처 아메리카노

세트 B: 브라우니 + 아메리카노 + 시그니처 라떼

세트 C: 바스크 치즈케이크 + 시그니처 라떼 + 시그니처 카푸치노

▼ 코드

```
# 판매음료와 디저트 카테고리
df = df[df['카테고리'].isin(['시그니처', 'Basic _ ice', '디저트', 'Basic', '비버리지'])]

# 디저트를 한개라도 주문한 결제일시만 포함시킴
datetime = df[df['카테고리']=='디저트']['결제일시']
df2 = df[df['결제일시'].isin(datetime)]

df2 = df2[['결제일시', '상품명', '수량']]
w1 = df2.groupby(['결제일시', '상품명'])['수량'].sum()
w2 = w1.unstack().reset_index().fillna(0).set_index('결제일시')
basket_df = w2.apply(lambda x: x>0)

w3 = df2[['결제일시', '상품명']].drop_duplicates()
w3 = w3.set_index('결제일시')
```

```
freq_items1 = apriori(basket_df, min_support = 0.06, use_colnames = True)
display(freq_items1.sort_values('support', ascending = False))
```

```
a_rules1 = association_rules(freq_items1, metric = "lift", min_threshold = 1)
a_rules1 = a_rules1.sort_values('lift', ascending = False).reset_index(drop=True)
display(a_rules1)
```

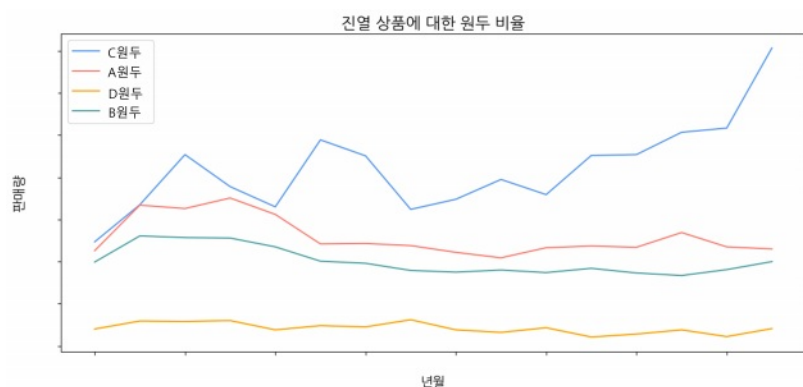
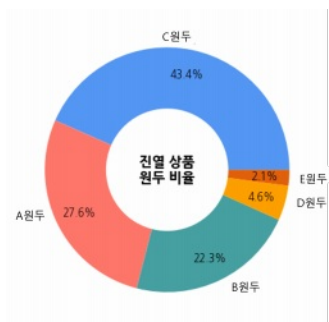
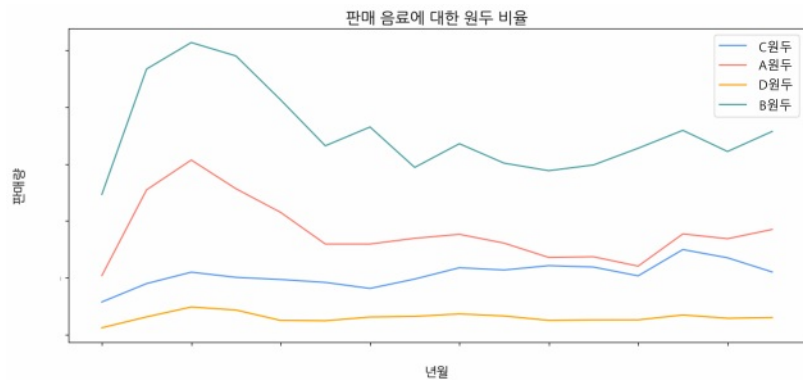
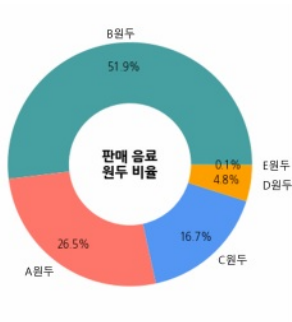
5-2. 오프라인 진열상품 최적화

스마트스토어의 리뷰를 읽어보면

“매장에서 마신 커피가 맛있어서 스토어에서 구매해갑니다!”

는 후기를 심심치 않게 읽어볼 수 있습니다.

그래서 매장의 고객들은 어떤 커피를 마시고, 어떤 커피 상품을 사갈까 궁금해서 분석을 진행했습니다.



위의 차트와 그래프를 보면, 고객은 매장에 방문해서만 즐겨볼 수 있는 시그니처 메뉴의 특성상 판매음료의 원두 비율은 B원두가 가장 높았습니다.

그리고 진열상품은 고객의 취향에 따라 C원두를 주로 사셨습니다.

고객이 선호하는 비율로 상품을 진열해야 매출을 증진할 수 있다고 생각합니다.



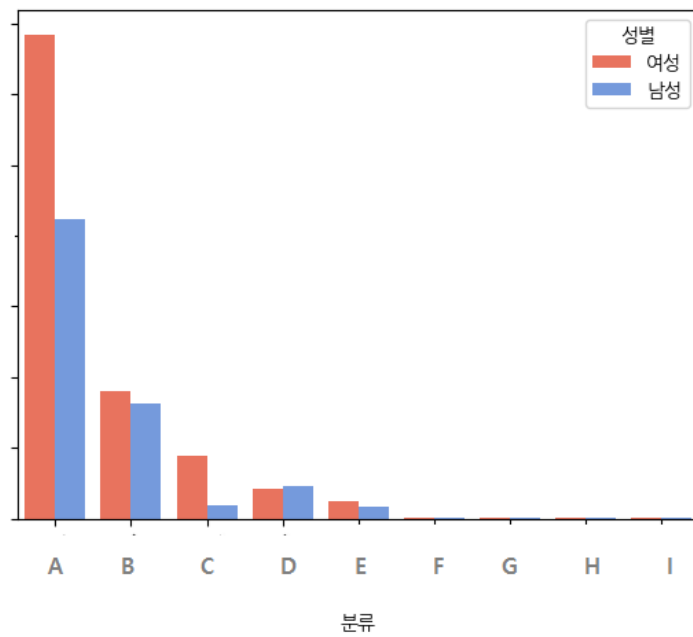
그래서 오프라인 매장에서 어떤 원두의 선호도가 가장 높았는지 파악하기 위해 카테고리가 '드립백/캡슐', '블렌딩원두', '싱글원두', '핸드드립' 인 카테고리의 워드 클라우드를 만들어보았습니다.

위 사진으로 '게이샤 페루', '게이샤 콜롬비아', '게이샤 과테말라'와 같이 '게이샤' 원두의 선호도가 높은 것을 확인할 수 있었습니다.

싱글 원두 상품의 진열 비중을 늘리고, 그 중 게이샤 원두에 대한 고객의 니즈를 좀 더 충족하면 매출 증진에 도움이 될 것 입니다.



각 상품 설명의 좌측 하단에 추천 연령을 제시하여 상품선택에 도움을 줄 수 있다.



“매장에서 마신 커피가 맛있어서 스토어에서 구매합니다!”

스마트스토어 QR코드를 메뉴 설명란과 매장 탁자에 붙여두길 권장드립니다.

5-3. 온라인 고객 타겟 분석 및 마케팅 제안

선호도 1위는 **A원두**가 차지했습니다. 2위는 **B원두**였으며, 3위는 남성의 경우 D커피가, 여성의 경우 **C커피**가 차지했습니다.

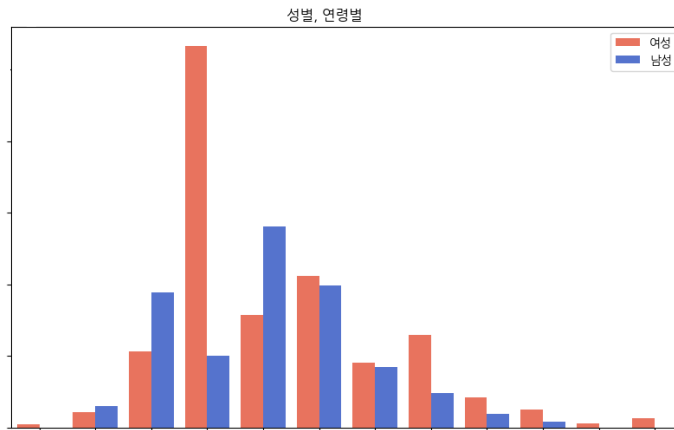


그림을 통해 ‘블렌딩 커피템플’, ‘클래식 블렌딩’, ‘쥬시 블렌딩’의 수요가 높은 것을 확인할 수 있었습니다.

- 고객 관심 유도: 메뉴별 순위를 통해 가장 인기 있는 메뉴를 강조함으로써 고객들의 관심을 끌고, 선택을 도와줍니다.
- 매출 증대: 고객들이 선호하는 메뉴들을 더 주목시킴으로써 해당 메뉴들의 판매량이 증가하게 됩니다. 이는 매출 증대로 이어질 수 있습니다.
- 효율적인 마케팅: 제품 포지셔닝과 시즌별 메뉴를 활용하여 메뉴별로 타겟팅된 마케팅 전략을 구축할 수 있습니다. 이는 고객들의 니즈를 충족시키고 더 효과적인 마케팅을 가능하게 합니다.
- 제품 다양성 강조: 다양한 메뉴들을 순위별로 소개하면서 전체 메뉴의 다양성을 강조할 수 있습니다. 이는 고객들에게 카페에서 다양한 선택지를 제공하는 인상을 심어줍니다.

즉, 아래 표의 3개의 계층이 온라인 매출의 절반 이상을 견인하고 있는 모습입니다.

순위	연령대	성별
----	-----	----



순위	연령대	성별
1	31~35	여성
2	36~40	남성
3	41~45	남성
4	36~40	여성
5	41~45	여성
6	26~30	남성
7	31~35	남성
8	51~55	여성
9	26~30	여성
10	46~50	여성

온라인에서의 마케팅은 물론이고, 오프라인에서의 마케팅 전략도 이 세 계층을 주요 타겟으로 설정하면 더 나은 매출 증진을 기대할 수 있을 겁니다.

우리는 이 분석을 통해 스마트스토어에서의

- 핵심 고객의 연령, 성별에 따른 마케팅 커뮤니케이션 조정
- 타겟 마케팅 진행을 위한 핵심 타겟 커뮤니티 파악 및 제휴 마케팅 제안

을 할 수 있습니다.



Action Point

- **게스트 바리스타** 활동을 활용해 주요 타겟 계층을 대상으로 인스타그램 주소와 스마트스토어를 홍보하고 이를 자체 커뮤니티로 활용해 자생적으로 성장하도록 유도합니다.
- 메뉴에 대한 피드백이나, 커피에 대한 관심도 증진을 통해 매출에 기여하도록 하고, 브랜드의 가치를 공유하여 성공적으로 충성 고객을 확보함을 목표로 합니다.

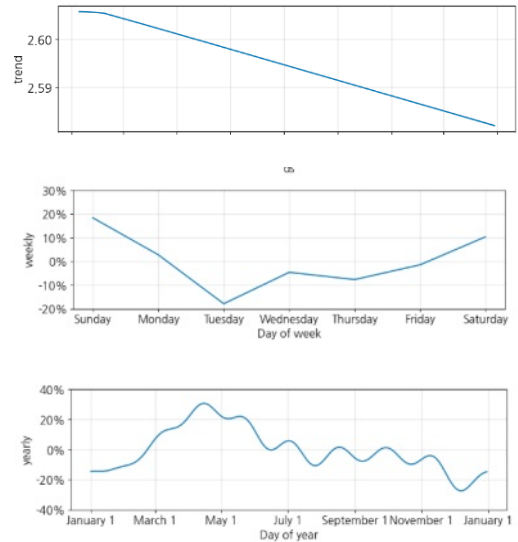
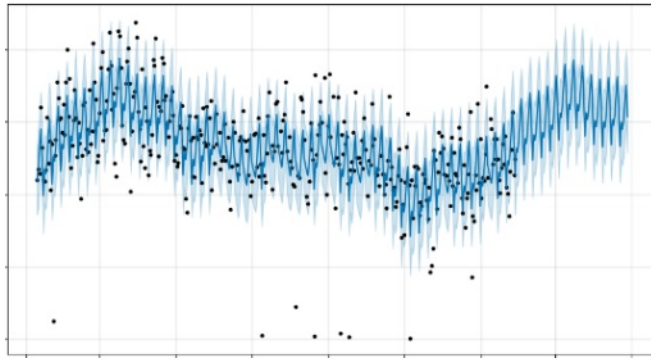
6. 모델링 예측 분석

6-1. 시계열 분석

- 공통
 - 시계열 예측 라이브러리로 **Prophet**을 사용했습니다.
 - `mday` = 2023-03-01
 - `train` : 2022-02-01 ~ 2023-02-28
 - `test` : 2023-03-01 ~ 2023-05-31
 - 예측기간 D = 90

- 데이터

ds	y
결제일	상품별 합계
날짜	상품별 합계



trend는 하락하는 추세를 보입니다. 일별 데이터로 **Changepoint = 500** 으로 설정 했으며 변화를 잘 감지 하도록 설정하였습니다.

두번째 '주 계절성' 차트인 **weekly**를 보면 금, 토 일이 많은 것을 알 수 있습니다.

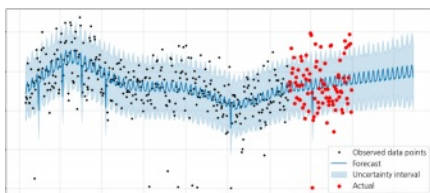
세번째 '연 계절성' 차트인 **yearly**에서는 **trend**와 마찬가지로 지속적으로 감소하는 것으로 보입니다.

1) 하이퍼 파라미터 최적화

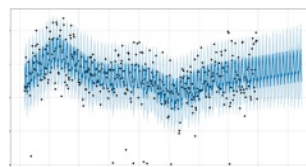
예측 확률 증가를 위해 최적화를 진행하였습니다.

하지만 하이퍼파라미터 튜닝을 할 수록 **과적합**이 되고 **예측 정확도가 하락**하였습니다.

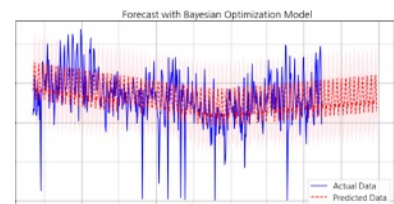
튜닝 모델	R2
그리드 서치(Grid search)	-0.0223
랜덤탐색(Random search)	0.1974
베이지안 최적화	0.2787
수동 하이퍼파라미터 튜닝 및 휴무일 추가	0.2718



그리드 탐색(Grid search)



랜덤 탐색(Random search)

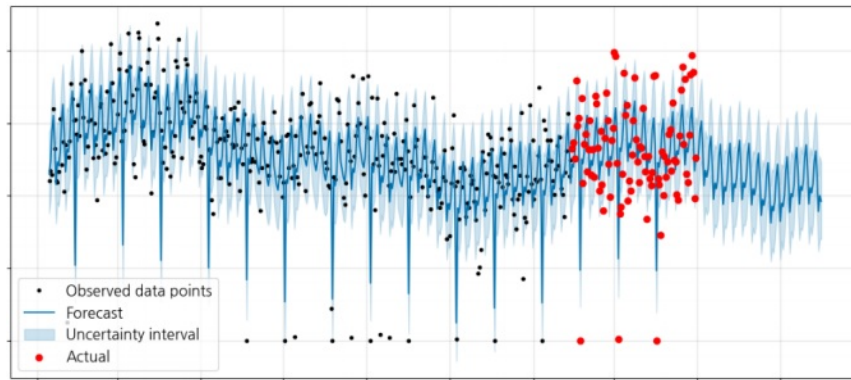


베이지안 최적화

2) 하이퍼 파라미터 튜닝

하이퍼 파라미터 튜닝을 직접해주고

커피템플 휴무일(매 월 첫째주 화요일) 반영을 위해 설정해주었습니다.



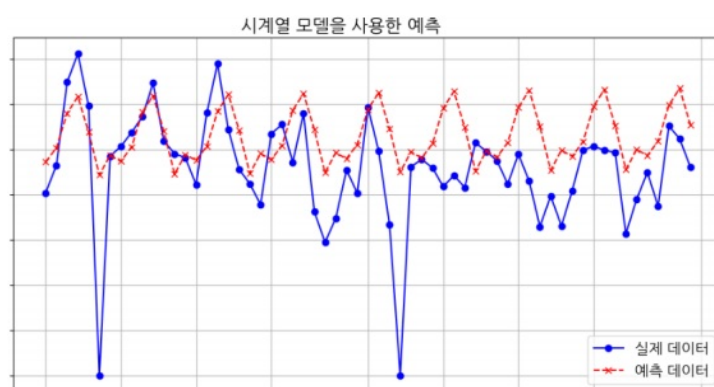
▼ 코드

```
custom_holidays = pd.DataFrame({
    'holiday': 'custom_holiday',
    'ds': pd.date_range(start='2022-02-01', end='2023-05-31', freq='WOM-1TUE')
})
```

```
model = Prophet(
    growth='linear',
    # changepoints=None,
    n_changepoints=500,
    changepoint_range=0.8,
    changepoint_prior_scale=0.1,
    seasonality_mode='multiplicative',
    seasonality_prior_scale=10.0,
    yearly_seasonality=True,
    weekly_seasonality=True,
    daily_seasonality=False,
    holidays=custom_holidays, # 커스텀 휴무일 추가
    holidays_prior_scale=10.0,
    interval_width=0.7,
)
```

R2값 = 0.2718

예측값 추출 후 6~7월 데이터와 비교했습니다.



시계열 예측 패키지 Prophet를 이용했고 기대에 미치지 못한 결과를 얻었습니다.

6-2. 회귀 분석

- 추가 검증 예측을 하기에 앞서, 기존 데이터를 `train` 과 `test` 로 나누어 **적합한 모델을 선정**하는 과정을 거쳤습니다.
- `mday` = 2023-03-01
- `train` : 2022-02-10 ~ 2023-02-28

- **test** : 2023-03-01 ~ 2023-05-31
- 사용 변수 : 평균기온(°C), 강수량(mm), 차량 수, 검색량
- ▼ 코드 - 데이터 분할

```
# 분할 기준 날짜 mday
mday = pd.to_datetime('2023-03-01')

cols = ['평균기온(°C)', '강수량(mm)', '차량 수', '검색량']
target = '일별 매출'

# 학습용 데이터의 index와 검증용 데이터의 index를 생성
train_index = df['결제일'] < mday
test_index = df['결제일'] >= mday

# 입력 데이터 분할
x_train = df[train_index][cols]
x_test = df[test_index][cols]

y_train = df[train_index][target]
y_test = df[test_index][target]
```

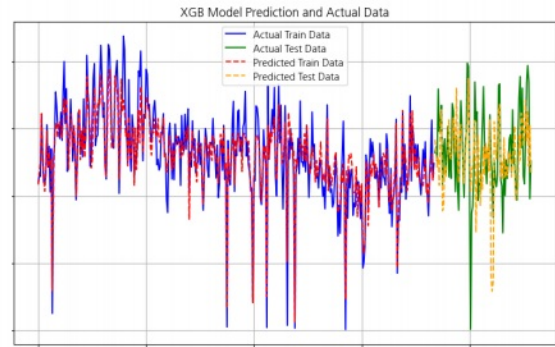
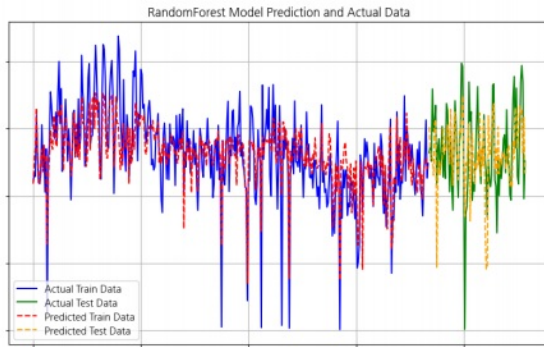
1) 일별 예측

1-1. 모델 적용 선택

	평균기온(°C)	강수량(mm)	차량 수	검색량	일별	일별
0	6.9000	7.2000	15.0000	33.8115		
1	7.5000	0.0000	31.0000	32.9918		
2	10.0000	0.0000	40.0000	48.5656		
3	10.1000	0.0000	43.0000	59.0164		
4	9.3000	0.0000	19.0000	61.2705		
...		
453	23.4000	0.0000	51.0000	64.5492		
454	24.2000	0.0000	53.0000	55.1230		
455	24.3000	6.4000	25.0000	72.3361		
456	21.0000	32.8000	43.0000	66.3934		
457	20.5000	10.2000	15.0000	47.1311		

- 매달 첫째주 화요일 휴무로 인한 이상치로 인해 (주별에 비하여) 정확도가 높지 않았습니다.

모델	train RMSE	valid RMSE	train RMSLE	valid RMSLE
Random Forest (그리드서치)	420260.7869	596558.5438	0.4311	0.5308
XGBoost (그리드서치)	280402.3507	644767.6705	0.3011	0.5703
Cat Boost (그리드서치)	445080.2392	587972.2457	0.4337	0.5388
AdaBoost	461819.9368	595152.6116	0.4125	0.5299



여러 모델을 적용해 보았을 때, **Random Forest** 와 **XGBoost** 의 학습 및 테스트 결과가 가장 좋은 것을 확인하였습니다.

1-2) 변수 선택

우선 각 컬럼 별 상관계수를 보면 다음과 같습니다.

이 중 일별 매출량을 잘 설명해줄 수 있는 지표는 무엇일까요?



Random Forest 모델을 기준으로 지표(특성) 별 모델링을 진행해 봤습니다. 지표 별 결과는 아래와 같습니다.

지표	train RMSE	valid RMSE	train RMSLE	valid RMSLE
검색량	592024.6349	667002.5746	0.5341	0.5639
강수량	639095.6065	604161.8914	0.5531	0.5427
기온	588438.3803	668522.7939	0.536	0.571
검색량, 강수량	578407.4991	623784.8835	0.5358	0.554
검색량, 평균기온	504576.2336	717406.6973	0.5009	0.5834
검색량, 차량수	466590.391	635158.942	0.4454	0.541
강수량, 기온, 검색량	489582.8857	643021.9762	0.5	0.562
강수량, 기온, 검색량, 차량수	420382.4947	596102.7888	0.4321	0.53
일별 수량(주문수)	117771.2465	118288.5841	0.0682	0.0737
차량수	524282.6085	616175.0682	0.4709	0.5047

지표를 어떤 것을 쓸지 고민하기보단,

지표를 쓰고나서 나온 실질적 추세를 지표와 연관지어서 어떻게 스토리텔링 할 지 고민해봤습니다.

강수량, 기온, 검색량, 차량수 총 4가지의 지표를 사용한 값을 사용하기로 했습니다.



변수해석

1. 강수량 🌧️

실내 좌석 > 야외 좌석

- 매장 외부에서 먹는 손님이 많다보니, 날씨의 영향을 받을 수 밖에 없을 것 입니다.
- 비가 많이 올 경우 야외 활동 제한되어 영향을 줄 것입니다.

2. 평균 기온 ☀️

- 기온이 쾌적한 봄/가을에는 상승되는 것으로 보입니다.
- 평균기온-검색량 간 상관관계(-0.58)가 높아 변수로 넣으면 더 유의미할 것이라는 판단도 있었습니다.

3. 네이버 '커피템플' 키워드 검색량 🔍

- (잠재적) 고객의 관심도를 나타내는 지표라고 생각됩니다.
- 검색량과 증감추이가 비슷한 양상으로 진행되는 것을 보였습니다.
- 아직 스마트 스토어가 활발하지 않은 상태이기 때문에, 검색량이 나중에 직접적으로 매장 방문으로 이어질 수 있는 지표라는 생각이 들었습니다.

4. 카카오맵 '커피템플' 차량 도착수 🚗

- 차량 수와 검색량은 밀접한 상관 관계가 있다고 그래프를 해석 할 수 있습니다.
 - 다만 모든 검색 포털, 모든 내비게이션 플랫폼에서 취합한 결과가 아니기 때문에 추세는 거의 정확하지만 1:1 매치는 어렵다는 한계가 보입니다.
- 다중공선성 문제를 우려했으나, 상관관계 0.11으로 고민하지 않아도 됩니다.
- 고객이 차량으로 올 수 밖에 없는 지리적, 환경적 요인을 고려하면 거의 차량을 가져오므로 예측과 직결될 가능성 높습니다.

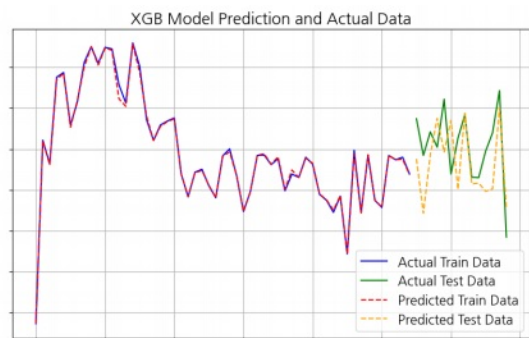
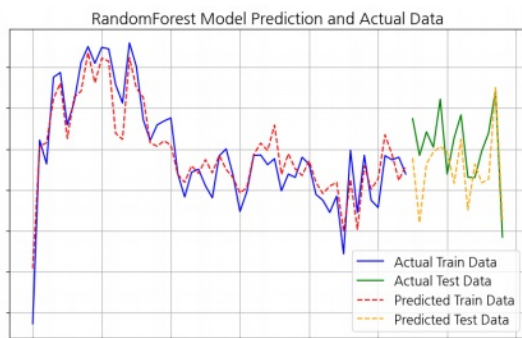
2) 주별 예측

주별 예측도 **Random Forest** 모델을 기준으로 지표(특성) 별 모델링을 진행해 봤습니다. 지표 별 결과는 아래와 같습니다.

지표	train RMSE	valid RMSE	train RMSLE	valid RMSLE
검색량	2146029.5468	3784033.7799	0.1293	0.2189
강수량	3080134.1643	2552071.9216	0.1935	0.1547
기온	2621753.3966	2592197.7636	0.169	0.156
검색량, 강수량	2217772.6672	3566526.8062	0.1434	0.2068
검색량, 평균기온	1184824.5508	4050295.5883	0.0858	0.2263
검색량, 차량수	842032.6773	2512075.4993	0.0597	0.1546
강수량, 기온, 검색량	1284314.2234	3740839.8383	0.0912	0.2104
강수량, 기온, 검색량, 차량수	1129480.9161	2082547.9335	0.0819	0.1261
차량수	1705236.1151	2304575.3668	0.1069	0.1455

모델	train RMSE	valid RMSE	train RMSLE	valid RMSLE	과적합 여부
Random Forest	1129480.9161	2082547.9335	0.0819	0.1261	
XGBoost	158936.4963	2345606.8571	0.008	0.1402	▲ (의심)

week	평균기온 (°C)	강수량 (mm)	차량 수	검색량	주별	주별
0	8.1333	2.4000	86.0000	115.3688		
1	5.4143	1.9857	218.0000	424.5901		
2	4.7000	0.0143	201.0000	454.7131		
3	9.1571	0.5429	223.0000	499.7950		
4	11.4857	0.0000	231.0000	525.4098		
...		
64	18.4667	24.7333	174.0000	323.9754		
65	16.5571	4.0857	161.0000	317.0082		
66	19.2143	6.0857	147.0000	312.7049		
67	19.8000	0.0429	261.0000	353.0737		
68	22.5000	12.3500	136.0000	240.9836		



17 일별 예측이 아닌 주별 예측을 사용하기로 한 이유

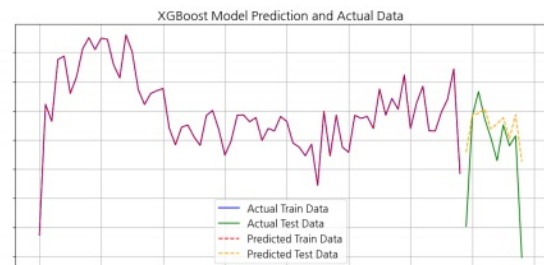
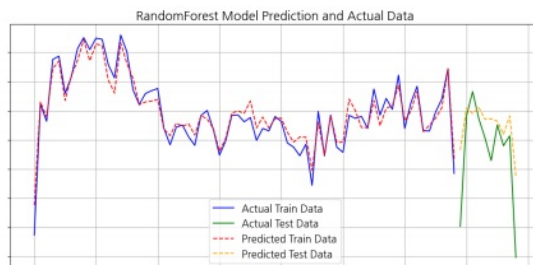
- 휴무일을 이상치로 처리하지 않고 예측에 반영할 수 있습니다.
- 변동성이 큰 일별 예측보다 그래프가 단순해져 예측력이 올라갑니다.
- valid RMSLE 값도 비교적 좋습니다. (약 0.5 → 0.1)

3) 예측일 주별 예측 후 실제(검증용) 데이터와 비교

- mday = 2023-06-01
- train : 2022-02-10 ~ 2023-05-31
- test : 2023-06-01 ~ 2023-07-31
- 사용 변수 : 평균기온(°C), 강수량(mm), 차량 수, 검색량

week	평균기온 (°C)	강수량 (mm)	차량 수	검색량	주별	주별
0	8.1333	2.4000	86.0000	115.3688		
1	5.4143	1.9857	218.0000	424.5901		
2	4.7000	0.0143	201.0000	454.7131		
3	9.1571	0.5429	223.0000	499.7950		
4	11.4857	0.0000	231.0000	525.4098		
...		
64	18.4667	24.7333	174.0000	323.9754		
65	16.5571	4.0857	161.0000	317.0082		
66	19.2143	6.0857	147.0000	312.7049		
67	19.8000	0.0429	261.0000	353.0737		
68	22.5000	12.3500	136.0000	240.9836		

모델	train RMSE	valid RMSE	train RMSLE	valid RMSLE	과적합 여부
Random Forest	806324.7929	3406174.7107	0.0588	<u>0.1402</u>	
XGBoost	8.1588	3555217.6342	0.0	0.3754	●



과적합



RandomForest 모델 결정 이유

1. valid RMSLE 값 비교

- RandomForest 0.1402 < XGBoost 0.3754

2. 과적합 가능성

- RandomForest 의 train RMSLE과 valid RMSLE 값의 차이가 크지 않음
- XGBoost 의 train RMSLE 값이 0.0 → 학습 데이터에 지나치게 최적화되어 과적합된 것으로 보임

3. 모델 특성

- RandomForest 는 앙상블 기법 중 하나로, 다양한 특성을 고려하여 예측을 수행하므로 일반적으로 다양한 데이터에 대해 좋은 성능을 보임
- XGBoost 는 Gradient Boosting 알고리즘으로 높은 예측 성능을 보이지만, 하이퍼파라미터 튜닝과 규제화를 제대로 적용하지 않으면 과적합될 수 있음

등의 특징으로 인하여 RandomForest 가 XGBoost 보다 예측에 조금 더 적합하다고 판단했습니다.

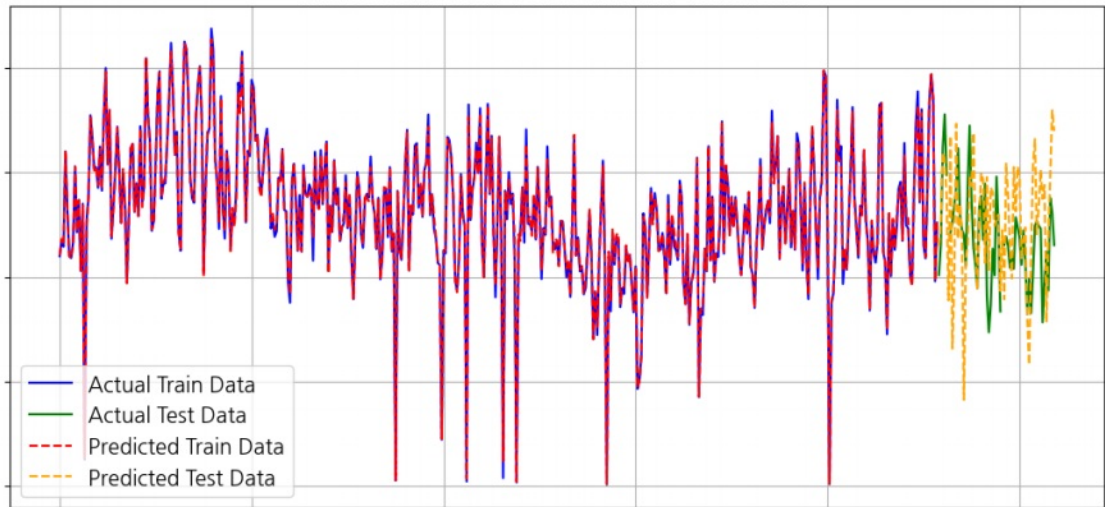


마지막 주의 실제값이 급하락한 이유는 7월 31일이 월요일이기 때문

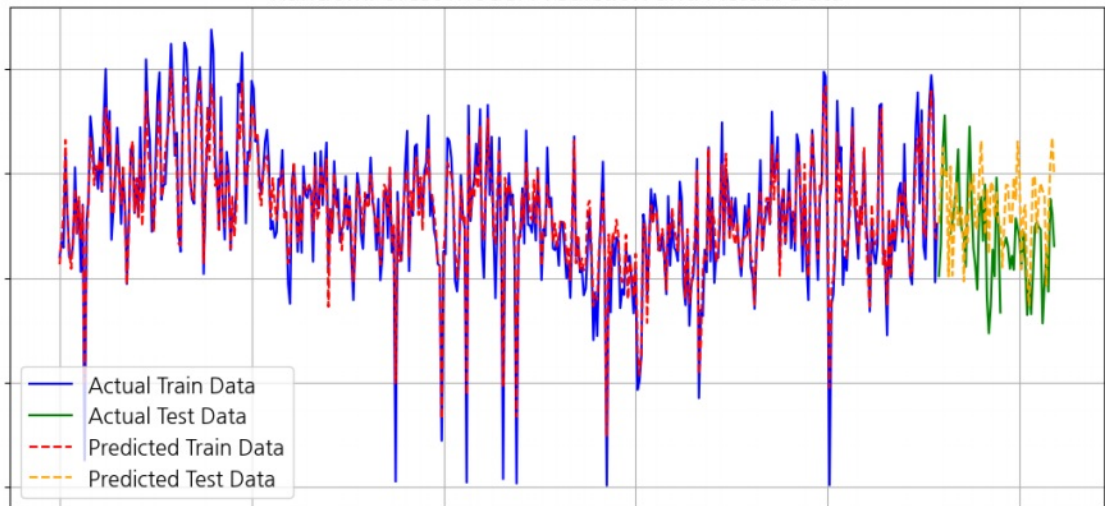
▼ (추가) 일별 예측

	평균기온 (°C)	강수량 (mm)	차량 수	검색량	일별	일별
0	6.9000	7.2000	15.0000	33.8115		
1	7.5000	0.0000	31.0000	32.9918		
2	10.0000	0.0000	40.0000	48.5656		
3	10.1000	0.0000	43.0000	59.0164		
4	9.3000	0.0000	19.0000	61.2705		
...		
453	23.4000	0.0000	51.0000	64.5492		
454	24.2000	0.0000	53.0000	55.1230		
455	24.3000	6.4000	25.0000	72.3361		
456	21.0000	32.8000	43.0000	66.3934		
457	20.5000	10.2000	15.0000	47.1311		

XGBoost Model Prediction and Actual Data



RandomForest Model Prediction and Actual Data



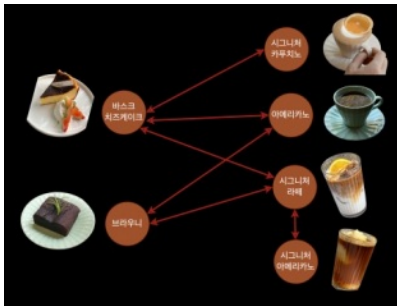
7. 인사이트 정리 및 결론

7-1. 인사이트 정리

1. 연관 분석을 통한 세트 메뉴 구성
2. 오프라인 진열상품 최적화
3. 온라인 고객 타겟 분석 및 마케팅 제안

7-2. 결론 (비즈니스 전략 수립)

- **상품 및 가격 전략:** 매출에 큰 영향을 미치는 상품을 강화 생산하거나 개선하여 수익을 높일 수 있습니다. 상품 다양화나 업그레이드를 고려해 볼 수 있습니다. 가격이 매출에 미치는 영향을 고려하여 경쟁력 있는 가격을 설정하거나 프로모션 전략을 수립합니다.
 - 세트메뉴 최적화



좌측의 그림을 통해 세트 메뉴를 제안합니다.

(↔ : 상품 간 연관성)

세트 A: 시그니처 라떼 + 시그니처 아메리카노

세트 B: 브라우니+아메리카노+시그니처 라떼

세트 C: 바스크치즈케이크 + 시그니처 라떼 + 시그니처 카푸치노

세트 메뉴 판매 시, 합계 금액에서 1,000원 정도 가격 할인을 하는 전략을 세우시는 것을 추천합니다.

- 매장 구매 고객에게 온라인스토어 할인 바우처 지급 마케팅.

매장 상품 구매 고객에게 온라인 스마트스토어의 상품을 구매를 유도하기 위해, 매장 탁자에 스마트스토어의 QR을 붙여 두고, 온라인스토어 할인 바우처 지급.

- 스마트스토어 내, 스토어찜 기능 사용.

신규고객 유입과 전체고객에 대한 프로모션 행사로 스마트스토어 충성 고객 확보 차원.

예시 : 카카오 스타일 쿠폰 효과

- 스마트스토어 내 '게이샤' 원두 관련 상품 개발



오프라인 상에서 게이샤 커피에 대한 선호도와 수요가 높은 반면,

온라인 상에서는 게이샤 커피 상품의 선호도가 전혀 반영이 되어있지 않은 상황입니다.

무로배송 | 혜택정보 | 배송유형 | 상품유형 | 가격대

✓ 정확도순 | 인기도순 | 최신등록순 | 낮은가격순 | 높은가격순 | 팔린순 | 리뷰많은순

40개씩 보기

SOLD OUT

[원두커피] 파나마 로스 나랑호스 골드 게이샤 / 100g / 커피템플

35% 22,000원 34,000원

파나마 로스 나랑호스 골드 게이샤 PANAMA Los Naranjos Gold Gesha 골드 로트의 커피는 소규모 수작업 컨셉의 게이샤 커피입니다. ...

평점 5.0

리뷰 3

포장부품 | 포장료

SOLD OUT

[원두커피] 온두라스 라 피후다 게이샤 / 100g / 커피템플

24,000원

온두라스 라 피후다 게이샤 HONDURAS La Pijuda Gesha 농장의 이름인 La Pijuda 는 품질이 우수하고 예쁜, 최상품을 의미합니다. La ...

평점 5.0

리뷰 1

포장부품 | 포장료

BEST

[원두커피] 페루 베야 비스타 게이샤 / 100g / 커피템플

28,000원

SINGLE ORIGIN 페루 베야 비스타 게이샤 Peru Bella Vista / Geisha / Washed 투영하연서도 향긋한 캐모마일의 향미와 달콤한 과일향의 ...

평점 5.0

리뷰 0

포장부품 | 포장료



[원두커피] 페루 베야 비스타 게이샤 / 100g / 커피템플

28,000원

상세정보

리뷰 0

Q&A 0

관련 태그

#싱글오리지커피

#싱글오리진

#핸드드립커피

상품정보 제공고시

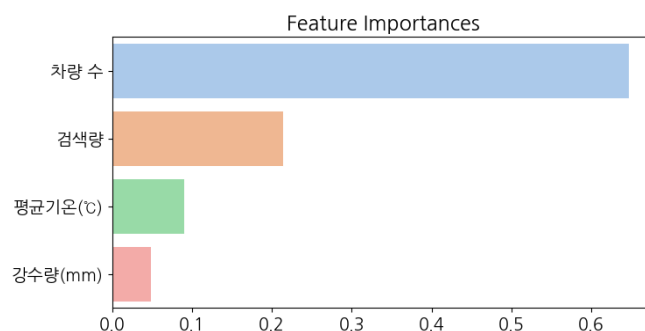
식품 위생법에 따른 표시사항	· 제품명	커피템플 싱글오리진 / 과테말라 산 안토니오 차기테 200g
	· 식품의 유형	원두커피
	· 생산자	커피템플
	· 소재지	커피템플 제주점
	· 제조연월일	주문일로부터 10일 이내
	· 소비기한 또는 품질유지기한	직접입력
	· 포장단위별 내용물의 용량(중량), 수량	200g
	· 포장단위별 수량	1
	· 원재료명 (농수산물의 원산지 표시 등 에 관한 법률)에 따른 원산지 표시 포함)	커피원두100%
	· 함량	

판매 제품명은 '페루 비야비스타 게이샤 100g'이고, 상품정보에는 '과테말라 산 안토니오 차기테 200g'으로 나와있어 원산지와 제품명의 미스매치가 보입니다.

소비자가 신뢰할 수 있는 스마트스토어 관리가 시급합니다.

또한, 게이샤 원두 기반의 커피캡슐과 드립백 상품의 개발이 필요하다고 생각합니다.

- **마케팅 전략:** 예측에 큰 영향을 미치는 변수를 중심으로 마케팅 전략을 수립합니다.



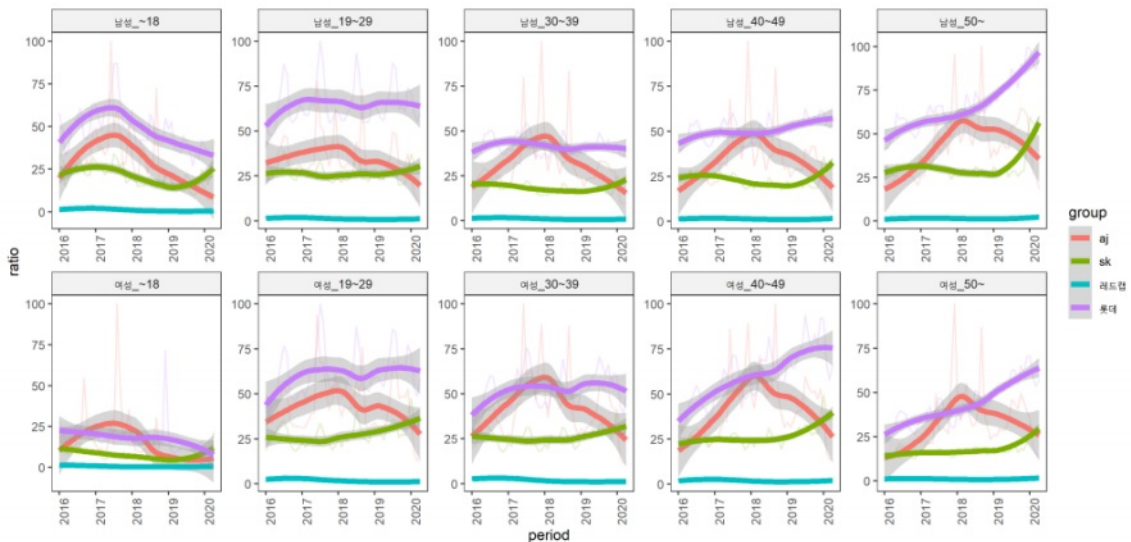
▼ 코드

```
modelRF = RandomForestRegressor()
modelRF.fit(x_train, y_train)

grb_importances_values = modelRF.feature_importances_
grb_importances = pd.Series(grb_importances_values, index = x_train.columns)
grb_top10 = grb_importances.sort_values(ascending=False)

# plt.rcParams["font.family"] = 'NanumGothicCoding'
plt.figure(figsize=(8,4))
plt.title('Feature Importances')
sns.barplot(x=grb_top10, y=grb_top10.index, palette = "pastel")
plt.show()
```

- 차량 도착 수가 예측에 가장 큰 영향을 미치는 변수라고 판단하였고, 이에 따라 다음과 같은 마케팅 전략을 수립했습니다.



30대 이상에서 롯데렌터카, SK렌터카의 검색량(2016~2020년 기준)이 증가한다. (출처: 하단 참고자료 표시)

- 차량으로 방문하는 고객 특성 상, 향후 추가 오픈 매장을 제휴하는 GS칼텍스의 MOU 업체인 롯데렌터카와의 렌터카 제휴로 매장 방문 할인 등 마케팅에 활용할 수 있습니다.

[보도자료] GS칼텍스, 전기차 렌터카 충전 서비스도 품는다 | GS칼텍스 미디어허브 - 아카이브 사이트
GS칼텍스, 8천대 전기차 보유 롯데렌탈과 충전 서비스 관련 업무협약 체결 전기차 충전 요금 및 세차 할인, QR코드 간편결제 등 고객 서비스 제공 친환경 렌터카 보급 및 인프라 구축을 위한 신사업 발굴 협력

<https://archive.gscaletxmediahub.com/story/rentalcar-charge-service/>



- **고객 타겟팅:** 고객 세분화를 실시하여 각 그룹에 맞는 맞춤 전략을 수립합니다. 특정 고객 그룹을 위한 상품 또는 서비스를 개발할 수 있습니다.

그래서 우리는 CRM 데실분석을 통해 고객계층을 10계층으로 나누었습니다.

주요 3계층을 타겟으로 전략을 수립하는 것을 제안합니다.

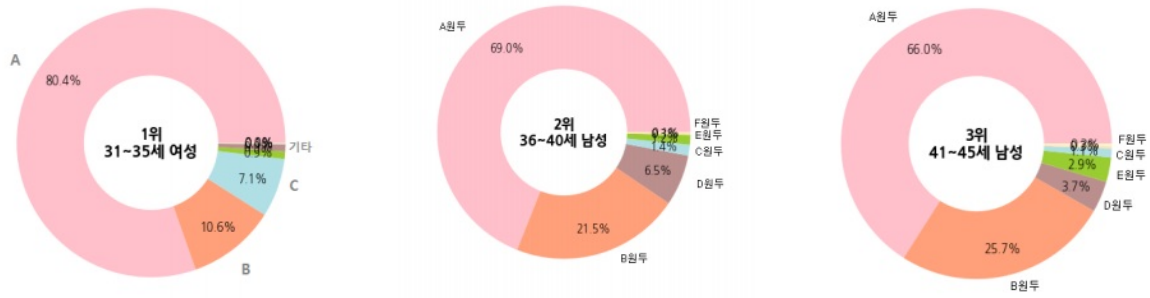
순위	연령대	성별	매출비율
1	31~35	여성	26.2%
2	36~40	남성	12.4%
3	41~45	남성	9.3%

주요 3계층은 다음과 같습니다.

1위 계층인 31~35세의 여성의 경우, C 원두 선호 비율이 눈에 띄는 편이니, C 원두의 상품을 새로 개발해도 좋을 것으로 판단됩니다.

2위 계층인 온라인 스마트스토어 고객의 매출 절반 이상을 차지하는 주요 고객 계층입니다.

3위 계층이 선호하는 원두 위주의 상품을 개발, 관리하는 방향을 추천 드립니다.



▼ 코드

```
one = custom[(custom['연령대']=='31~35') & (custom['성별']=='여성')].groupby('분류')['매출액'].sum()

plt.pie(one.sort_values(ascending=False), labels=one.sort_values(ascending=False).index,
        autopct='%1.1f%%', textprops={'fontSize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'powderblue', 'yellowgreen', 'rosybrown',
                'moccasin', 'lemonchiffon', 'lightsteelblue'])

plt.text(0, 0, '1위\n31~35세 여성', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')
```

```
two = custom[(custom['연령대']=='36~40') & (custom['성별']=='남성')].groupby('분류')['매출액'].sum()

plt.pie(two.sort_values(ascending=False), labels=two.sort_values(ascending=False).index,
        autopct='%1.1f%%', textprops={'fontSize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'rosybrown', 'powderblue', 'yellowgreen',
                'moccasin', 'lemonchiffon', 'lightsteelblue'])

plt.text(0, 0, '2위\n36~40세 남성', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')
```

```
three = custom[(custom['연령대']=='41~45') & (custom['성별']=='남성')].groupby('분류')['매출액'].sum()

plt.pie(three.sort_values(ascending=False), labels=three.sort_values(ascending=False).index,
        autopct='%1.1f%%', textprops={'fontSize': 9},
        # startangle=0, explode=(0.1, 0, 0, 0, 0), shadow=True,
        pctdistance=0.8, labeldistance=1.05,
        wedgeprops=dict(width=0.5),
        colors=['pink', 'lightsalmon', 'rosybrown', 'yellowgreen', 'powderblue',
                'moccasin', 'lemonchiffon', 'lightsteelblue'])

plt.text(0, 0, '3위\n41~45세 남성', fontsize=12, fontweight='bold',
        color='black', ha='center', va='center')
```

8. 프로젝트 회고 및 개선점

8-1. 피드백

8-2. 회고

--	--

팀원	회고
함승주	
정우용	
이윤지	
김예린	

- 온라인 고객 타겟 분석 : 온/오프라인 통틀어 정확한 고객 데이터가 있었으면 더 많고 다양한 CRM분석을 시도해 볼 수 있었을 것으로 생각되어 아쉬운 부분이 있습니다.
- 연관분석
- 오프라인 진열상품 최적화

8-3. 개선점

8-4. 추후 개선 계획

프로젝트는 여전히 살아 숨쉬고 있습니다.

개선점에 대한 회고 이후, 가능하다면 실제 액션 계획도 세워보세요. 포트폴리오에서 '개선 시도/경험'은 아주 긍정적인 요소로 작용한답니다.


9. 부록

9-1. 참고자료

도메인 조사:

coffeetemple

스페셜티 커피 전문기업 커피템플

 <http://www.coffeetemple.co.kr/>



네이버 지도

제주 월평동 카페

<https://map.naver.com/v5/search/%EC%A0%9C%EC%A3%BC%20%EC%9B%94%ED%8F%89%EB%8F%99%20%EC%B9%B4%ED%8E%98?c=16.3,0,0,0,dha&p=7TOVoa9EdYjnzsswNbEb5g,111.15,-3.68,80,Float>



데이터 분석:

남동욱 / 데이터 분석 결과를 액서너블한 비즈니스 인사이트로 만... | 커리어리

데이터 분석을 열심히 하고 결과를 공유하면 "그래서 뭐 어쩔?" 같은 느낌의 표정을 마주하게 됩니다. 데이터...

 https://careerly.co.kr/comments/87527?utm_campaign=user-share



영업력 강화로 이어지는 고객 관리를 적절하게 하기 위해서는?

영업 프로세스에서 CRM의 효과는 널리 알려져 있습니다. 그러나, 데이터를 축적하기만 하고 분석하지 않으면 CRM을 도입하는 의미가 없습니다. 고객 분석에 관한 비즈니스 개념과 대표적인 방법에 대해 소개하겠습니다.

 <https://www.salesforce.com/kr/hub/crm/crm-analysis/>



모델링 :

Jonas Kim / 머신러닝 문제 정의 단계에서 절대 하지 말아야 할 일 ... | 커리어리

ML에서 문제 정의는 중요성이 과소평가되며, 적은 작업 시간이 투입되는 단계입니다. 그러나 첫 단추를 잘못 ...

https://careerly.co.kr/comments/87830?utm_campaign=user-share



XGBoost(eXtra Gradient Boost)

XGBoost는 트리 기반의 앙상블 학습에서 가장 주목받고 있는 알고리즘 중 하나입니다. XGBoost는 GBM의 단점인 느린 수행 시간과 과적합 규제(Regularization) 부재 등의 문제를 해결해서 매우 주목을 받고 있습니다.

XGBoost의 장점들을 정리해보도록 하겠습니다. 뛰어난 예측 : 분류와 회귀 영역에서 뛰어난 예측 성능을 발휘함

https://jaaamj.tistory.com/37



추가분석:

2020 제주 렌터카시장 점유율, 나이대(남성/ 여성)렌터카 선호도

‘핀팬’ 불러 모으는 커뮤니티 마케팅

안녕하세요, 애드옵스 플랫폼 아드리엘입니다. 최근 시장에는 셀 수 없이 많은 브랜드와 서비

https://www.openads.co.kr/content/contentDetail?contsId=9774

핀팬 모으는 비결은?

커뮤니티
마케팅



9-2. 출처

외부데이터:

기상자료개방포털[기후통계분석:통계분석:조건별통계]

기상청, 기상자료개방포털

https://data.kma.go.kr/climate/RankState/selectRankStatisticsDivisionList.do?pgmNo=179

날씨 데이터. 평균기온, 강수량

제주관광공사

제주관광공사는 관광을 통한 지역경제 발전과 관광산업 육성 및 주민복지 증진을 도모합니다. 해외 관광객 유치 기반 조성, 제주관광공사 지정면세점, 신규수익사업 발굴, 글로벌 통합마케팅 강화, 관광상품 질적 패러다임 변화 선도, 자립기반 사업 다각화, 경영시스템 효율화

https://ijto.or.kr/korean/

차량 도착수

네이버 데이터랩 : 검색어트렌드

네이버 통합검색에서 검색된 검색어와 검색횟수를 기간별/연령별/성별로 조회할 수 있습니다.

http://datalab.naver.com/keyword/trendResult.naver?hashKey=N_505ea45cb6dc7718d57de9434b64fef5

NAVER
데이터랩

‘커피템플’ 키워드 검색량

오프라인 진열상품 분석:

[제주여행] 커피템플, 한국 바리스타 챔피언 김사홍 바리스타의 고즈넉한 카페 ☕

커피 템플 COFFEE TEMPLE 📍 제주 제주시 영평길 269 중선농원 커피템플 ☎ 0507-1311-4050...

<https://blog.naver.com/abaec74/223162676962>



사진자료 출처

제주 카페 추천 커피템플

제주 카페 추천 커피템플 사실 난 커피를 1도 안먹지만 같이 여행오는 사람들을 위하여 커피 맛있다고 유명...

<https://blog.naver.com/md3544/223124719910>



사진자료 출처