NS SHOP+ 판매실적 예측을 통한 편성 최적화 방안(모형) 도출

이수필 (chmk95456@gmail.com)

목차

데이터 전처리

데이터 전처리

```
shop.drop(shop.loc[shop['취급액'].isnull()].index, inplace = True)
```

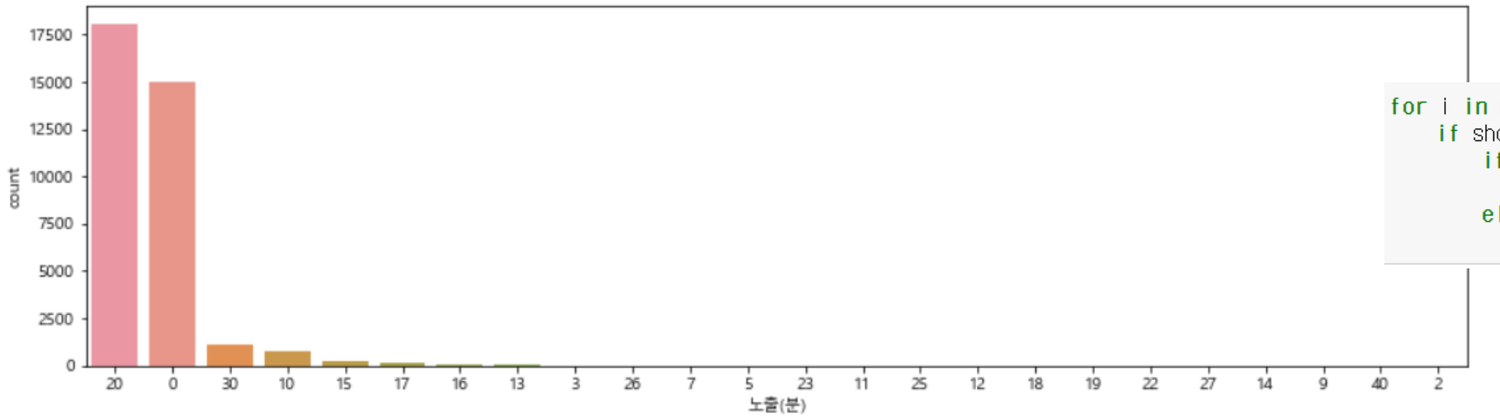
```
shop_t = shop_t[shop_t['상품군'] != '무형']
```

제공데이터와 평가데이터에서 취급액 값이 0인 상품군 중 무형에 속하는 데이터들을 제거해준다.

```
def year_month(방송일시):
    return "{0}-{1}".format(방송일시.year, 방송일시.month)
shop['년월'] = shop['방송일시'].apply(year_month)
shop['년'] = shop['방송일시'].dt.year
shop['월'] = shop['방송일시'].dt.month
shop['일'] = shop['방송일시'].dt.day
shop['시'] = shop['방송일시'].dt.hour
shop['분'] = shop['방송일시'].dt.minute
shop['요일'] = shop['방송일시'].dt.dayofweek
shop['분기'] = shop['방송일시'].dt.quarter
shop['계절'] = 'nan'
season = ['겨울', '겨울', '봄', '봄', '봄', '여름', '여름', '여름', '가을', '가을'
for i in range(len(shop)):
    shop['계절'].iloc[i] = season[(shop['월'].iloc[i])-1]
```

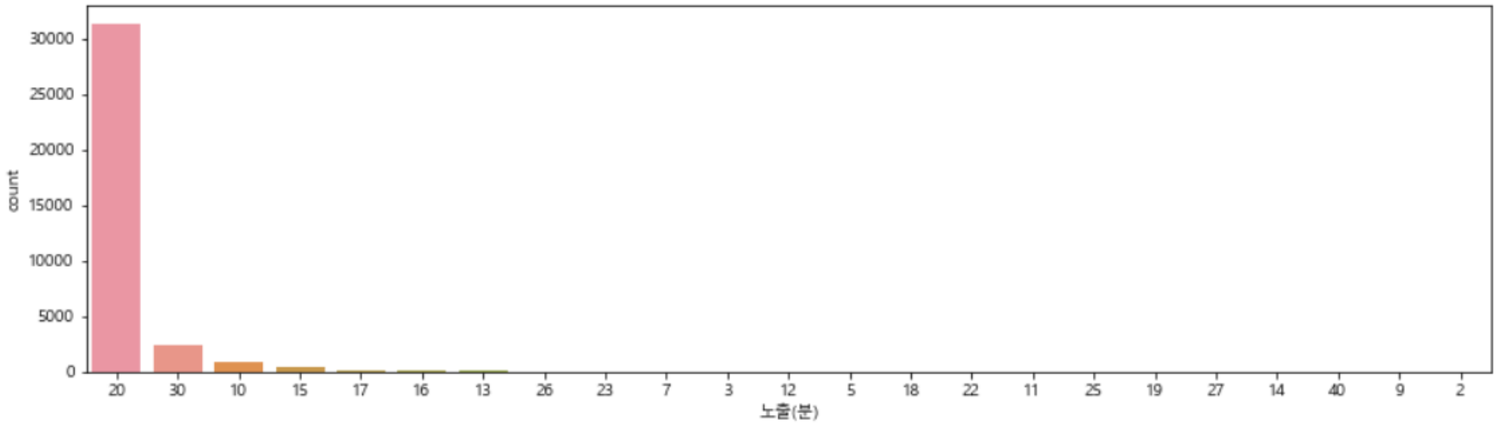
```
def year_month(방송일시):
    return "{0}-{1}".format(방송일시.year, 방송일시.month)
shop_t['년월'] = shop_t['방송일시'].apply(year_month)
shop_t['년'] = shop_t['방송일시'].dt.year
shop_t['월'] = shop_t['방송일시'].dt.month
shop_t['일'] = shop_t['방송일시'].dt.day
shop_t['시'] = shop_t['방송일시'].dt.hour
shop_t['분'] = shop_t['방송일시'].dt.minute
shop_t['요일'] = shop_t['방송일시'].dt.dayofweek
shop_t['분기'] = shop_t['방송일시'].dt.quarter
shop_t['계절'] = 'nan'
season = ['겨울', '겨울', '봄', '봄', '봄', '여름', '여름', '여름', '가을', '가을', '가을', '겨울']
for i in range(len(shop_t)):
    shop_t['계절'].iloc[i] = season[(shop_t['월'].iloc[i])-1]
```

제공데이터와 평가데이터에서 방송 일시 컬럼을 이용하여 년 월, 월, 시, 요일, 분기, 계절 등의 컬럼들을 추가해준다.




```
for i in range(len(shop)):
    if shop.iloc[i]['노출(분)'] == 0:
        if grouped[shop['방송일시'][i]] != 0:
            shop.iloc[i, shop.columns.get_loc('노출(분)')] = grouped[shop['방송일시'][i]]
        else:
            shop.iloc[i, shop.columns.get_loc('노출(분)')] = 20
```

노출(분) 컬럼을 countplot으로 시각화 해 본 결과 nan 값인 0이 두번째로 많은 것을 확인 할 수 있다.
노출(분) 컬럼을 모델의 피처로 사용하기 방송 일시가 같다면 그 방송 일시와 같은 값을 넣어 주고
방송 일시가 같은 데이터가 없다면 평균 값과 가장 비슷한 20으로 넣어 주기로 했다.



코드 실행 후 시각화 해 본 결과 0값이 모두 사라진 것을 확인할 수 있다.

shop

	방송일시	노출(분)	마더코드	상품코드	상품명	상품군	판매단가	취급액	년월	년	월	일	시	분	요일	분기	계절
0	2019-01-01 06:00:00	20	100346	201072	테이트 남성 셀린니트3종	의류	39900	2099000.00000	2019-1	2019	1	1	6	0	1	1	겨울
1	2019-01-01 06:00:00	20	100346	201079	테이트 여성 셀린니트3종	의류	39900	4371000.00000	2019-1	2019	1	1	6	0	1	1	겨울
2	2019-01-01 06:20:00	20	100346	201072	테이트 남성 셀린니트3종	의류	39900	3262000.00000	2019-1	2019	1	1	6	20	1	1	겨울
3	2019-01-01 06:20:00	20	100346	201079	테이트 여성 셀린니트3종	의류	39900	6955000.00000	2019-1	2019	1	1	6	20	1	1	겨울
4	2019-01-01 06:40:00	20	100346	201072	테이트 남성 셀린니트3종	의류	39900	6672000.00000	2019-1	2019	1	1	6	40	1	1	겨울
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
35374	2019-12-31 23:40:00	20	100448	201391	일시불쿠첸압력밥솥 6인용	주방	148000	10157000.00000	2019-12	2019	12	31	23	40	1	4	겨울
35375	2020-01-01 00:00:00	20	100448	201383	무이자쿠첸압력밥솥 10인용	주방	178000	50929000.00000	2020-1	2020	1	1	0	0	2	1	겨울
35376	2020-01-01 00:00:00	20	100448	201390	일시불쿠첸압력밥솥 10인용	주방	168000	104392000.00000	2020-1	2020	1	1	0	0	2	1	겨울
35377	2020-01-01 00:00:00	20	100448	201384	무이자쿠첸압력밥솥 6인용	주방	158000	13765000.00000	2020-1	2020	1	1	0	0	2	1	겨울
35378	2020-01-01 00:00:00	20	100448	201391	일시불쿠첸압력밥솥 6인용	주방	148000	46608000.00000	2020-1	2020	1	1	0	0	2	1	겨울

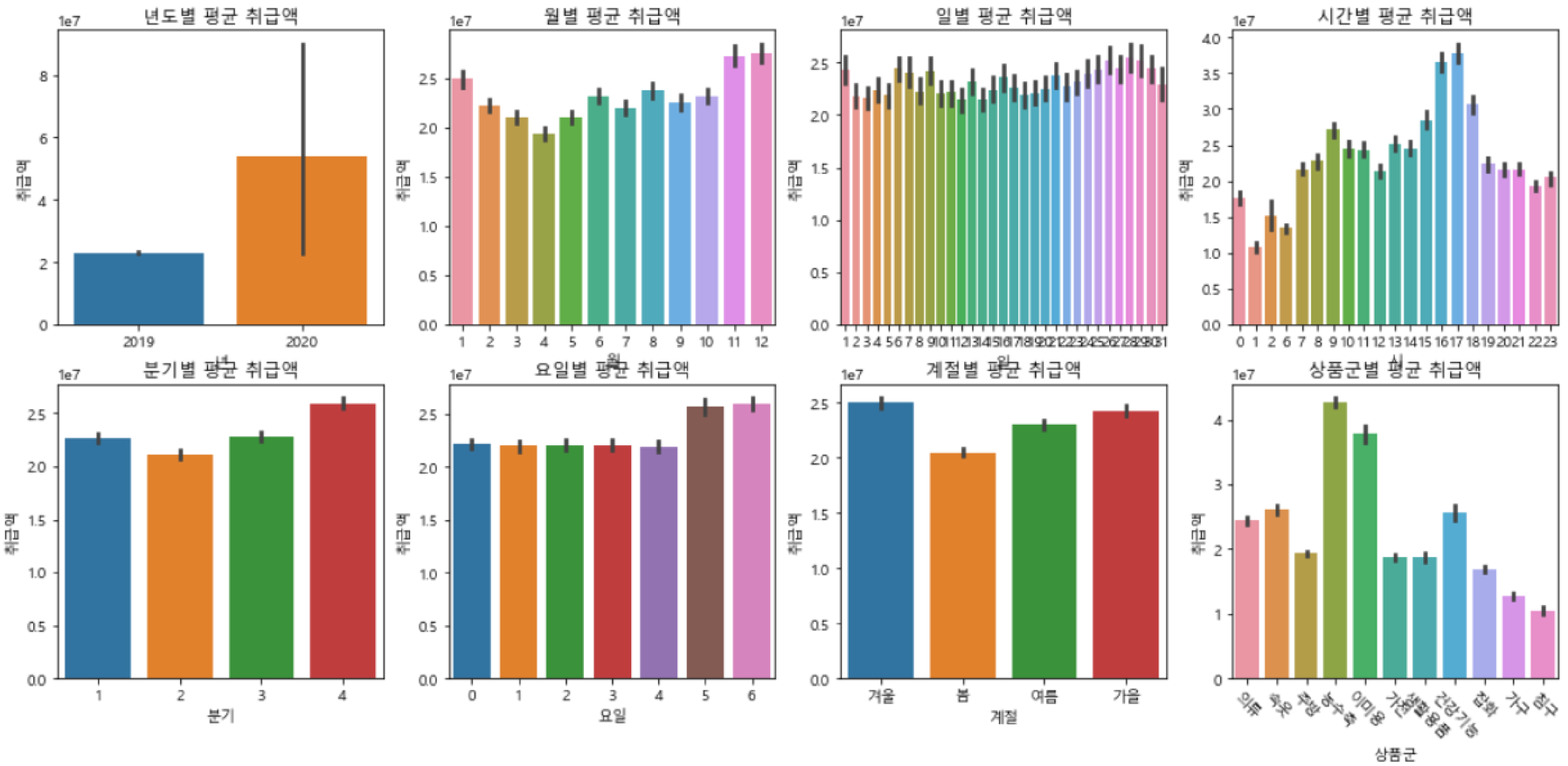
35379 rows × 17 columns

shop_t

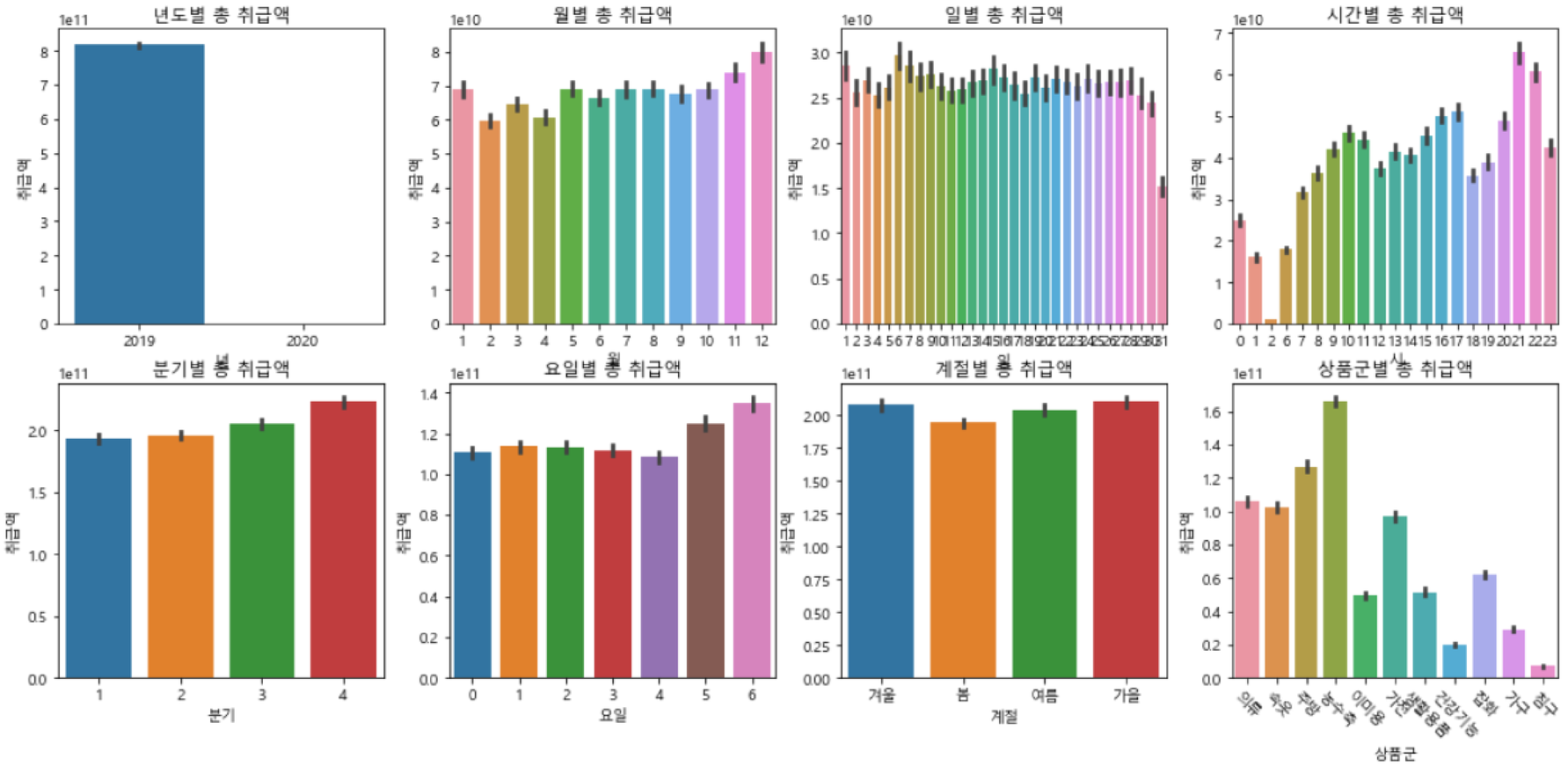
	방송일시	노출(분)	마더코드	상품코드	상품명	상품군	판매단가	취급액	년월	년	월	일	시	분	요일	분기	계절
0	2020-06-01 06:20:00	20	100650	201971	잭필드 남성 반팔셔츠 4종	의류	59800	0.00000	2020-6	2020	6	1	6	20	0	2	여름
1	2020-06-01 06:40:00	20	100650	201971	잭필드 남성 반팔셔츠 4종	의류	59800	0.00000	2020-6	2020	6	1	6	40	0	2	여름
2	2020-06-01 07:00:00	20	100650	201971	잭필드 남성 반팔셔츠 4종	의류	59800	0.00000	2020-6	2020	6	1	7	0	0	2	여름
3	2020-06-01 07:20:00	20	100445	202278	쿠미투니카 쿨 레이시 란쥬쉐이퍼&팬티	속옷	69900	0.00000	2020-6	2020	6	1	7	20	0	2	여름
4	2020-06-01 07:40:00	20	100445	202278	쿠미투니카 쿨 레이시 란쥬쉐이퍼&팬티	속옷	69900	0.00000	2020-6	2020	6	1	7	40	0	2	여름
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2711	2020-07-01 00:10:00	10	100099	200273	[일시불]라쉬반 FC바로셀로나 드로즈 패키지	속옷	99000	0.00000	2020-7	2020	7	1	0	10	2	3	여름
2712	2020-07-01 00:10:00	10	100099	200272	[무이자]라쉬반 FC바로셀로나 드로즈 패키지	속옷	119000	0.00000	2020-7	2020	7	1	0	10	2	3	여름
2713	2020-07-01 00:10:00	10	100099	200274	라쉬반 FC바로셀로나 드로즈 8종	속옷	119000	0.00000	2020-7	2020	7	1	0	10	2	3	여름
2714	2020-07-01 01:20:00	20	100261	200875	아놀드파마 티셔즈레깅스세트	의류	69900	0.00000	2020-7	2020	7	1	1	20	2	3	여름
2715	2020-07-01 01:40:00	16	100261	200875	아놀드파마 티셔즈레깅스세트	의류	69900	0.00000	2020-7	2020	7	1	1	40	2	3	여름

바뀐 데이터를 확인해보면 새로운 컬럼들이 추가된 것을 확인할 수 있다.

데이터 분석

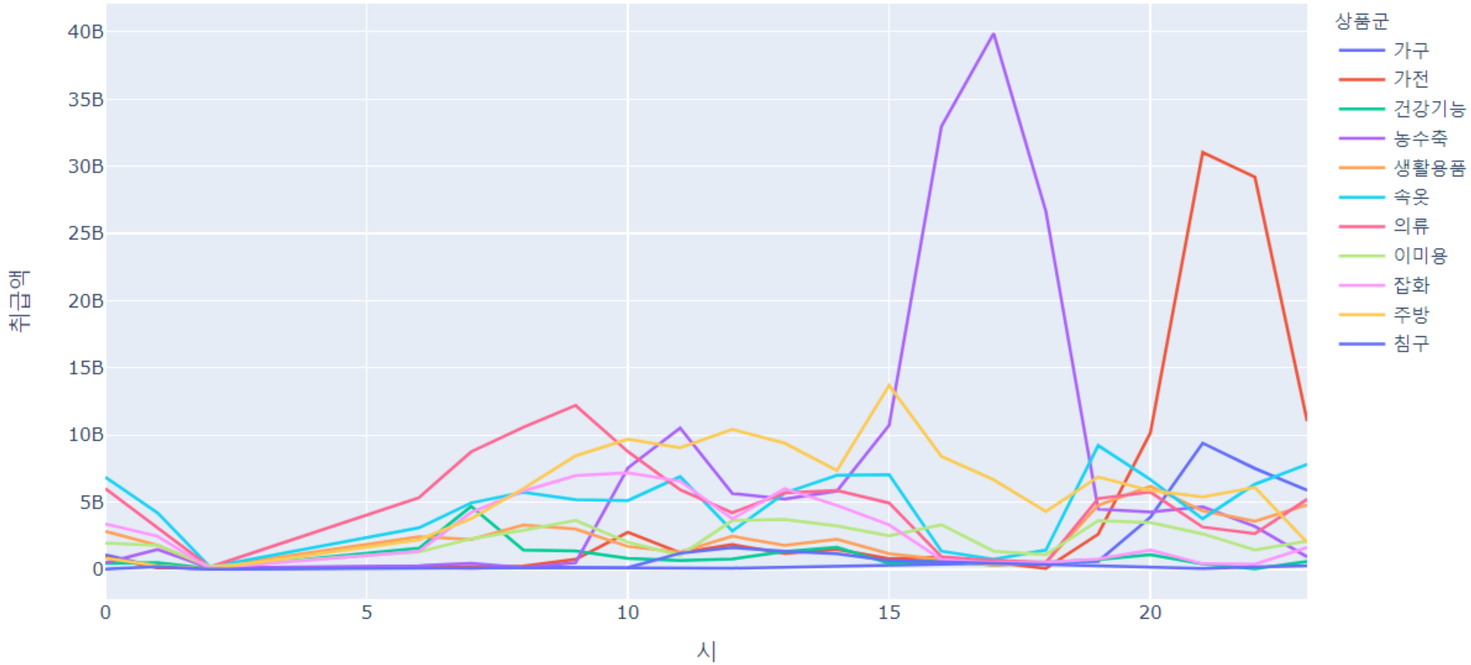


평균 취급액을 기준으로 데이터를 시각화 해 본 결과 겨울인 11, 12월, 저녁 식사 시간 전인 4, 5시, 주말인 토, 일, 상품군별로는 농수축과 이미용 분야가 평균 취급액이 가장 높은 것을 확인할 수 있다.



총 취급액을 기준으로 데이터를 시각화 해 본 결과 저녁 식사 시간 후인 9, 10시에 취급액이 가장 높았고 상품군별로는 농수축과 주방 분야가 총 취급액이 가장 높은 것을 알 수 있다.

```
pdt_time = shop.groupby(["시", "상품군"])[['상품군', '취급액']].sum()
pdt_time = pdt_time.reset_index(level=[0,1])
fig = px.line(pdt_time, x = '시', y = '취급액', color = '상품군')
fig.show()
```



시, 상품군, 취급액 컬럼을 groupby로 묶어준 후 더해줘 시간대 별 각 상품군의 총 취급액 값을 가지고 있는 pdt_time 이라는 데이터 프레임을 만들고 시각화 해본 결과 저녁 식사 전 시간대에 농수축 상품들의 취급액이 높고,저녁 식사 이후 시간대에 가전 제품들의 취급액이 높은 것을 확인할 수 있다.

```
pdt_group = pdt_time['상품군'].drop_duplicates()
pdt_group = list(pdt_group)
pdt_group
```

```
['가구', '가전', '건강기능', '농수축', '생활용품', '속옷', '의류', '이미용', '잡화', '주방', '침구']
```

```
pdt_group_re = pd.DataFrame(np.arange(22).reshape(11, 2),
                            index = pdt_group, columns = ['시', '취급액'])
```

```
pdt_idx = []
```

```
for i in range(len(pdt_group)):
    pdt_idx.append(pdt_time[pdt_time['상품군']==pdt_group[i]].취급액.idxmax())
```

```
pdt_idx
```

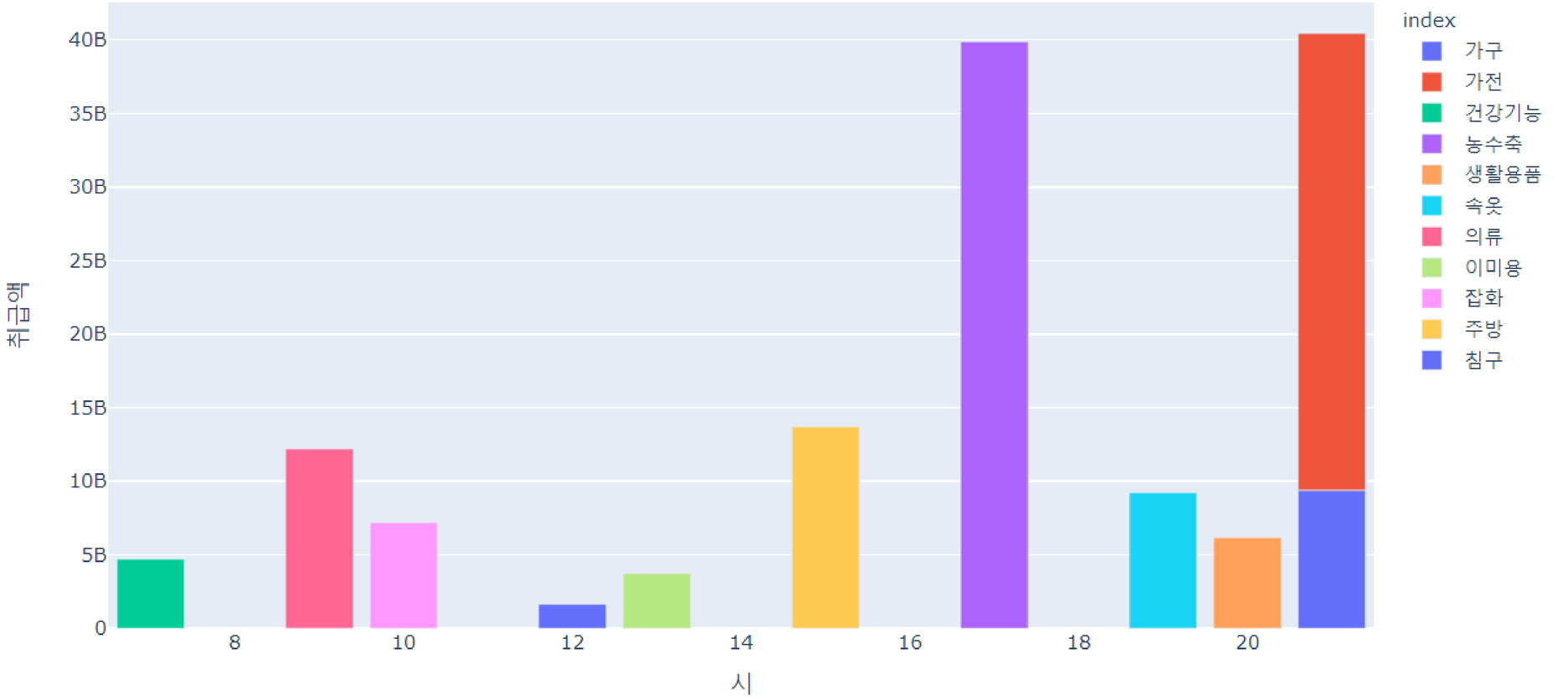
```
[181, 182, 44, 145, 175, 166, 68, 110, 80, 132, 103]
```

```
for i in range(len(pdt_idx)):
    pdt_group_re.loc[pdt_group[i], ['시']] = pdt_time.iloc[pdt_idx[i]].시
    pdt_group_re.loc[pdt_group[i], ['취급액']] = pdt_time.iloc[pdt_idx[i]].취급액
```

```
pdt_group_re
```

	시	취급액
가구	21	9396720000.00000
가전	21	31030382000.00000
건강기능	7	4690377000.00000
농수축	17	39880541000.00000
생활용품	20	6178662000.00000
속옷	19	9223430000.00000
의류	9	12203216000.00000
이미용	13	3716337000.00000
잡화	10	7189207000.00000
주방	15	13699423000.00000
침구	12	1627799000.00000

전에 만든 pdt_time 이라는 데이터 프레임을 이용하여 각 상품군별 최대 취급액과 그 시간을 넣은 데이터 프레임을 만들어 준다.



위 데이터를 시각화 하면 다음과 같은 결과가 나타난다.

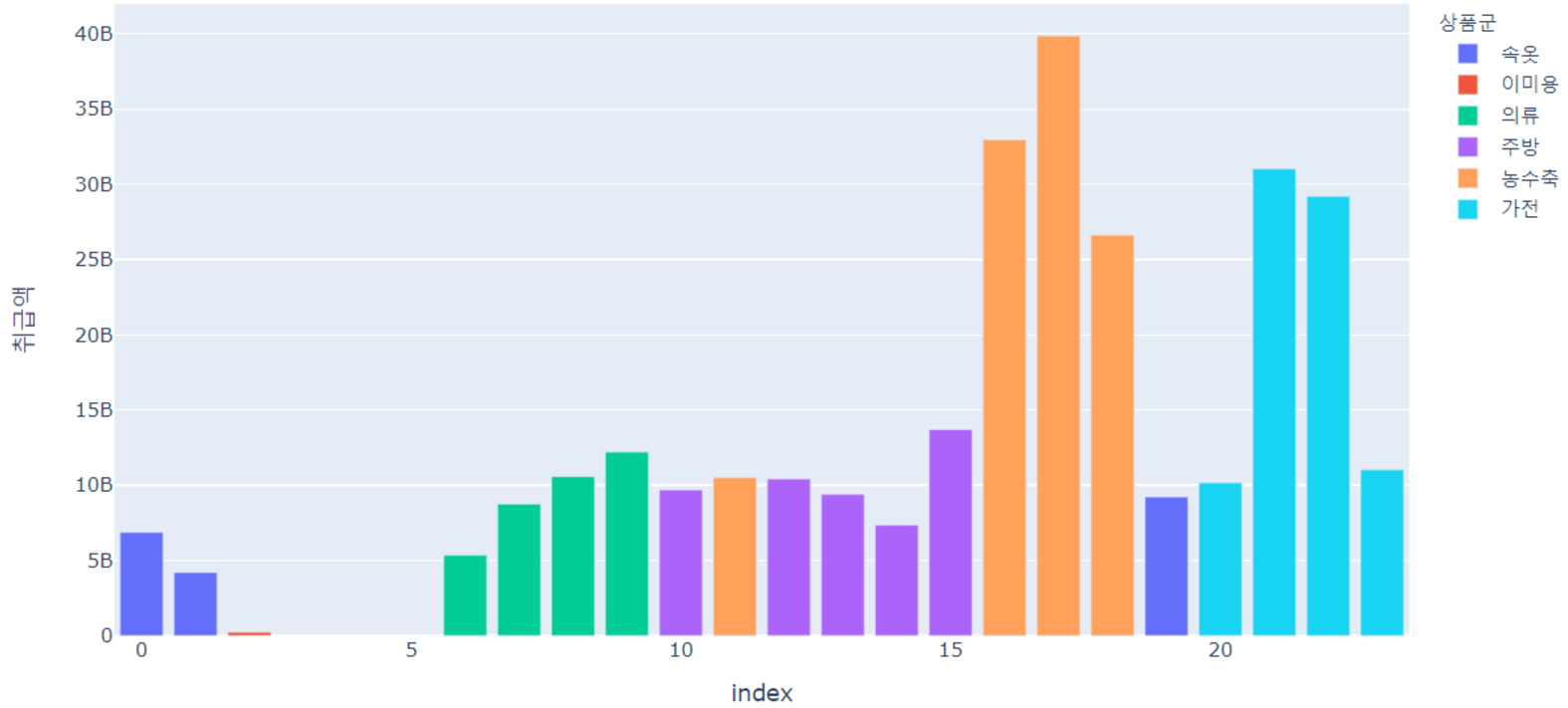
```
pdt_hour = pdt_time['시'].drop_duplicates()
pdt_hour = list(pdt_hour)
pdt_hour
```

```
[0, 1, 2, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23]
```

```
pdt_idx1 = []
for n in pdt_hour:
    pdt_idx1.append(pdt_time[pdt_time['시']==n].취급액.idxmax())
pdt_idx1
```

```
pdt_group_time = pd.DataFrame(np.arange(42).reshape(21, 2),
                              index = pdt_hour, columns = ['상품군', '취급액'])
pdt_group_time
```

```
for i in range(len(pdt_idx1)):
    pdt_group_time.iloc[i, 0] = pdt_time.iloc[pdt_idx1[i]].상품군
    pdt_group_time.iloc[i, 1] = pdt_time.iloc[pdt_idx1[i]].취급액
pdt_group_time
```



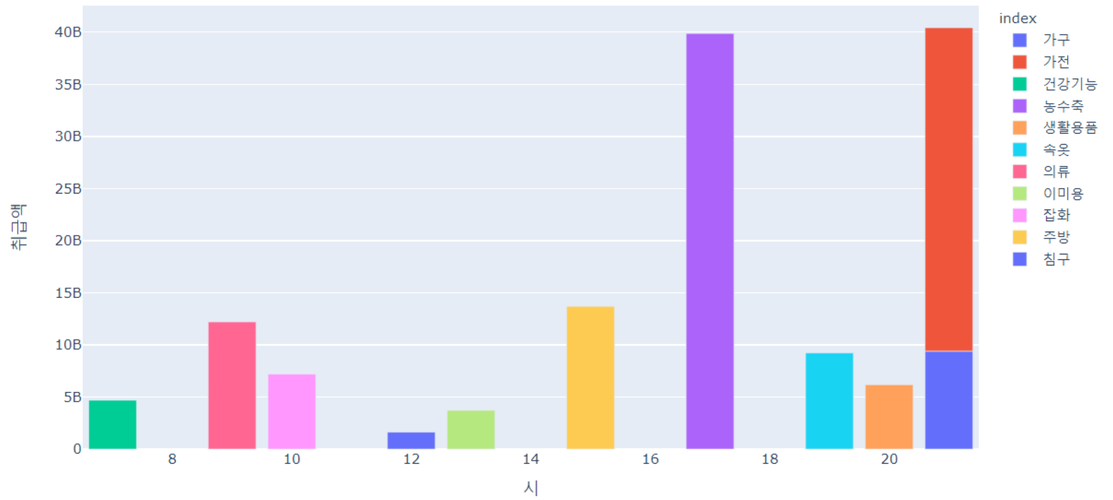
	상품군	취급액
0	속옷	6864607000.00000
1	속옷	4202392000.00000
2	이미용	231643000.00000
6	의류	5341645000.00000
7	의류	8754229000.00000
8	의류	10580369000.00000
9	의류	12203216000.00000
10	주방	9683624000.00000
11	농수축	10515826000.00000
12	주방	10418813000.00000
13	주방	9392184000.00000
14	주방	7356592000.00000
15	주방	13699423000.00000
16	농수축	32969415000.00000
17	농수축	39880541000.00000
18	농수축	26636253000.00000
19	속옷	9223430000.00000
20	가전	10169152000.00000
21	가전	31030382000.00000
22	가전	29193334000.00000
23	가전	11034628000.00000

다시 pdt_time 이라는 데이터 프레임을 이용하여 시간대별 최대 취급액과 취급액에 해당하는 상품군을 담고있는 데이터 프레임을 만들고 시각화 해준다.

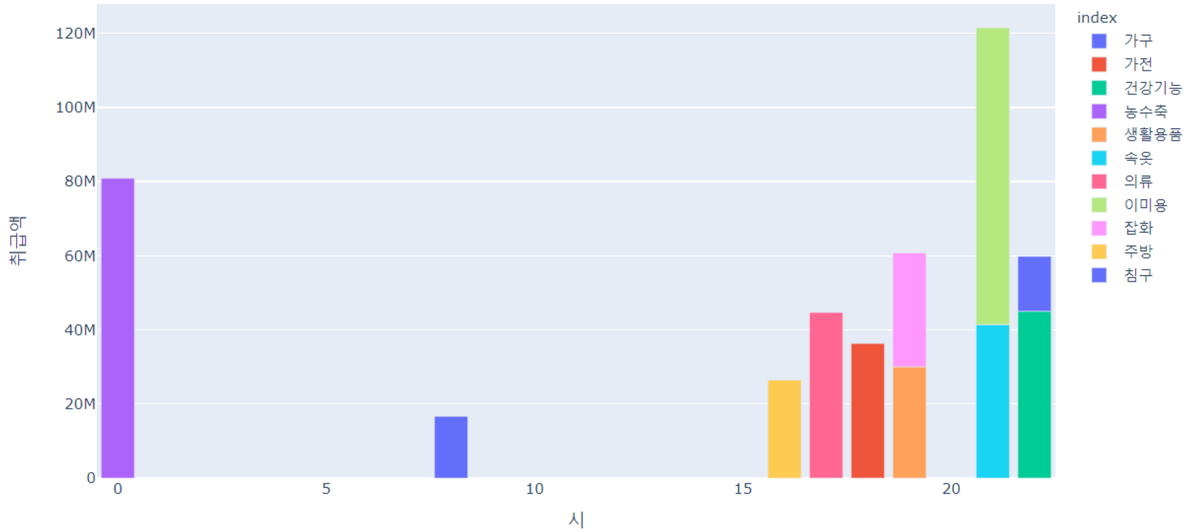
```
pdt_time_m = shop.groupby(["시", "상품군"])[['상품군', '취급액']].mean()
pdt_time_m = pdt_time_m.reset_index(level=[0,1])
```

위의 과정을 취급액의 총 값에서 취급액의 평균값으로만 바꿔서 똑같이 실행해보았다.

총 취급액

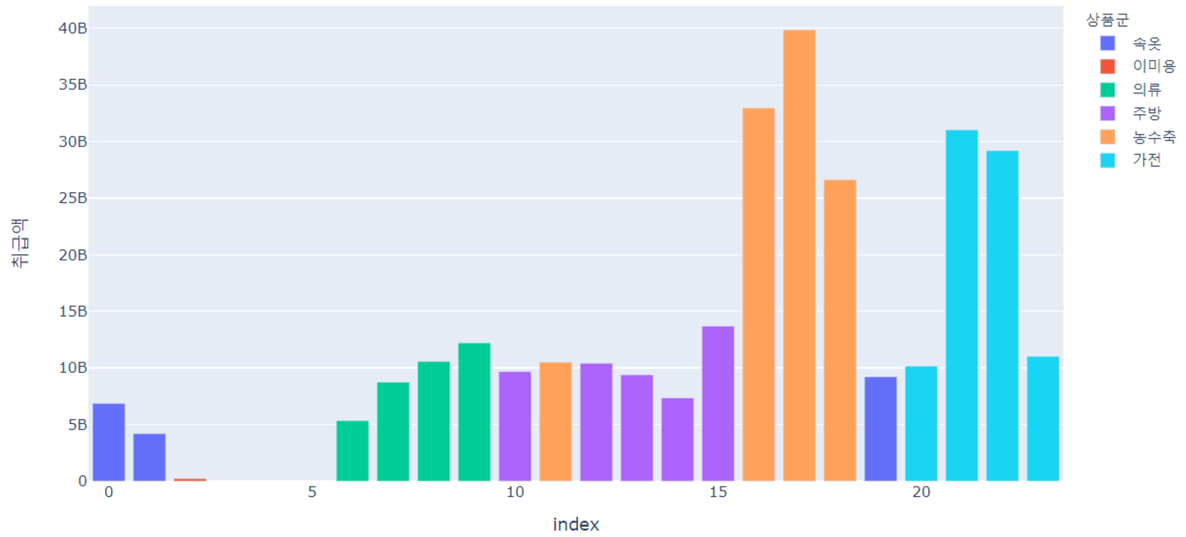


평균 취급액

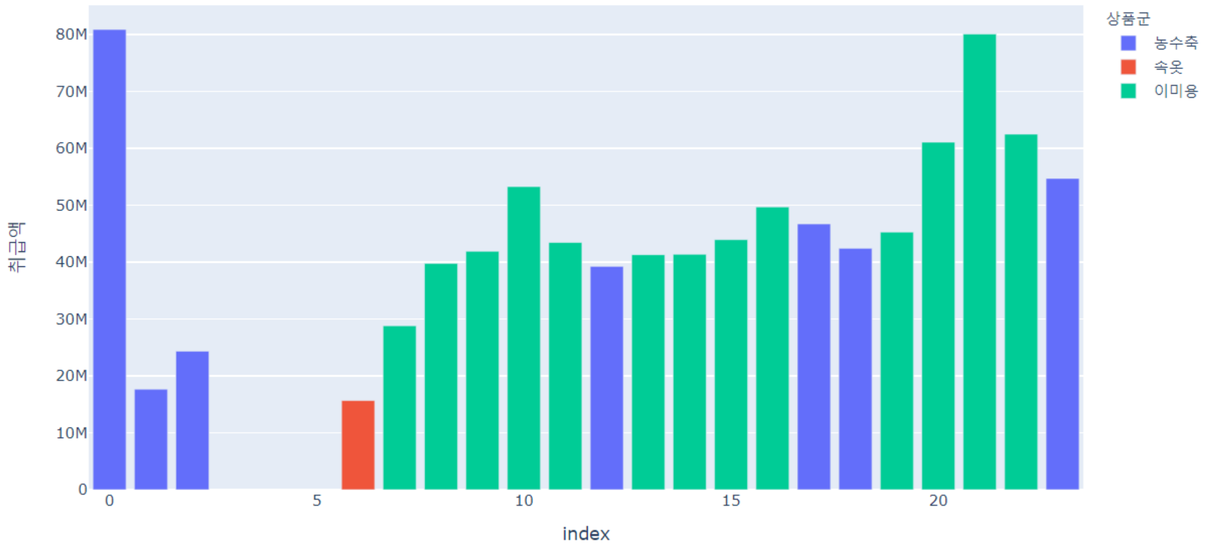


두 그래프의 상관관계는 크게 찾아볼 수 없으므로 총 취급액에 미치는 영향이 큰 것은 방송 횟수 임을 알 수 있다.

총 취급액



평균 취급액

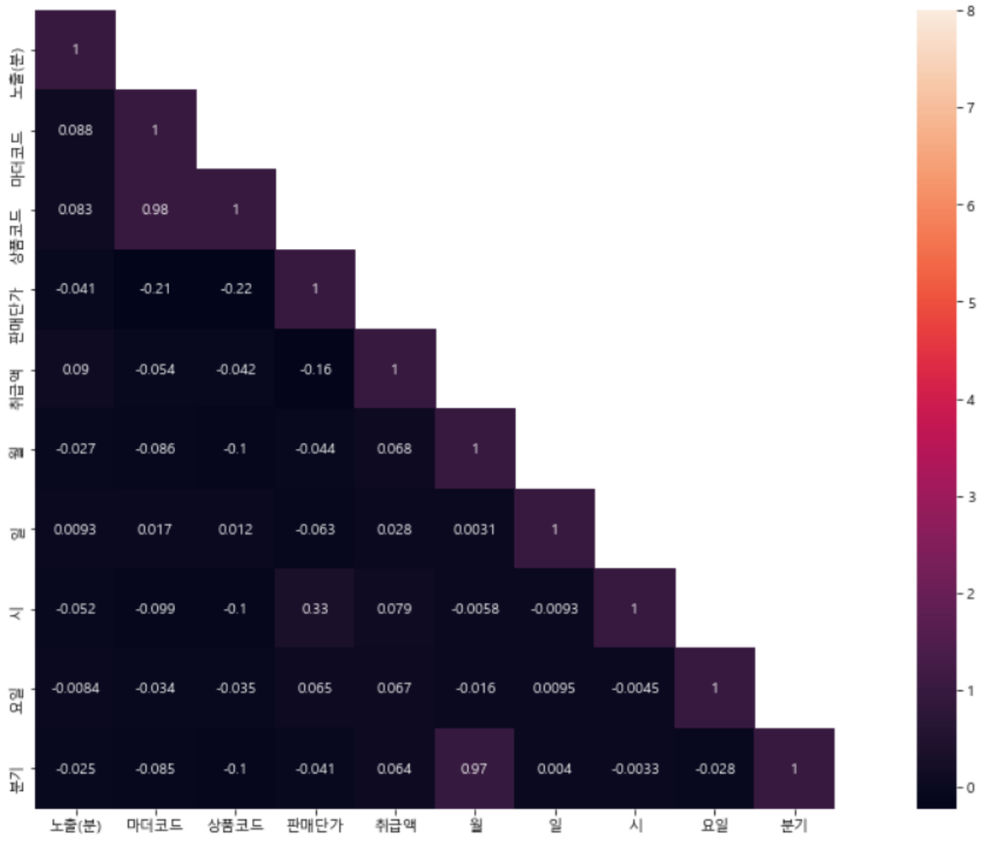


이미용 상품들은 총 취급액 에선 큰 비중을 차지하지 못하지만 평균 취급액에서는
큰 비중을 차지하므로 이는 고객들의 수요는 많으나 공급이 적은 것으로 해석할 수 있고,
이를 통해 이미용 관련 상품들의 판매 비중을 늘리는 것이 전체 취급액의 상승으로
이어질 수 있을 것이라고 예상할 수 있다.

XGBoost 모델을 통한 6월 데이터 취급액 예측

모델 선택 이유

- 병렬 처리를 사용하여 학습과 예측이 빠르다.
- 욕심쟁이 알고리즘을 사용한 자동 가지치기가 가능하여 과 적합이 잘 일어나지 않는다.




```
X_train = shop[['월', '요일', '시', '일', '노출(분)']]
y_train = shop['취급액']
```

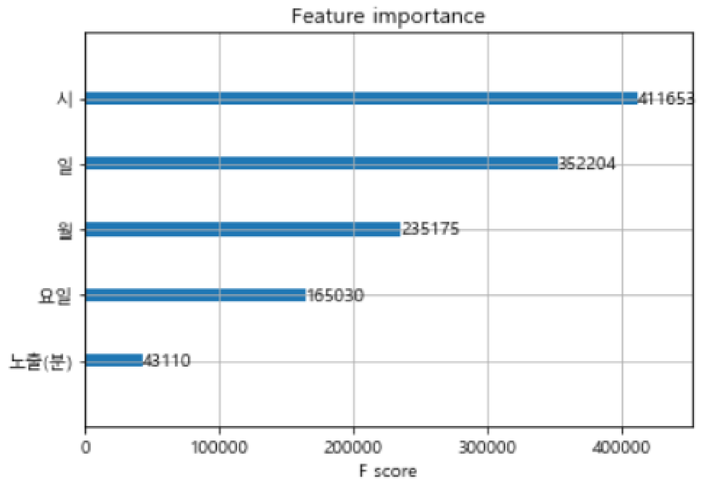
히트맵을 이용하여 취급액과 상관관계가 있는 컬럼들을 모델의 피처로 이용하기로 한다

```
model = XGBRegressor(n_estimators = 20000)
model.fit(X_train, y_train)
pred = model.predict(X_train)
```

```
mae = mean_absolute_error(y_train, pred)
mae
```

```
8850205.955291275
```

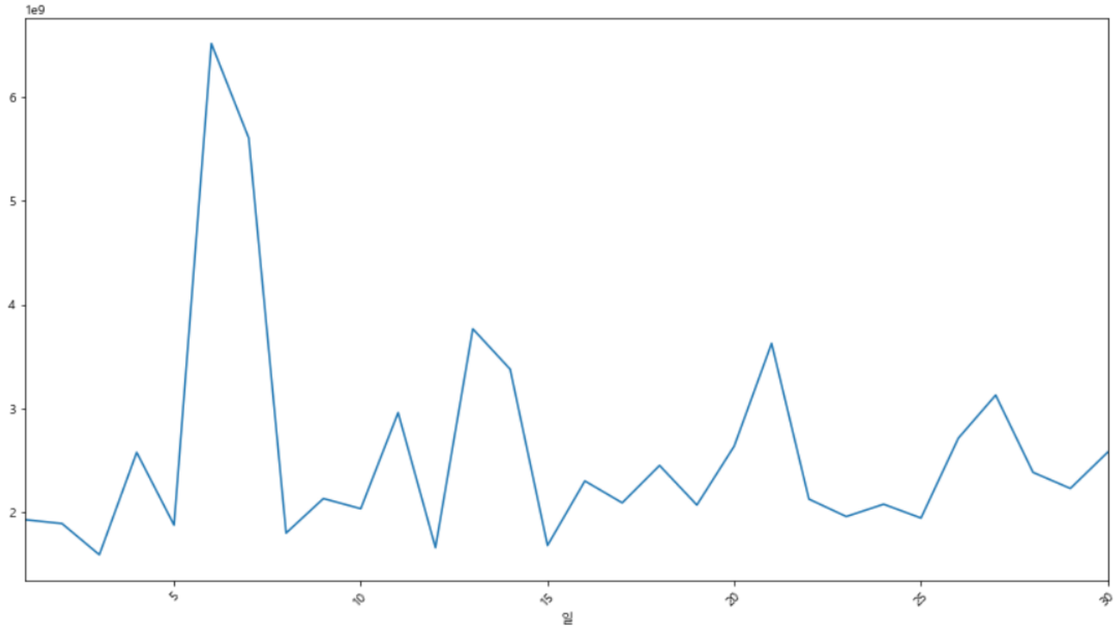
XGBRegressor 모델을 불러오고 트리 개수는
20000개로 설정해 준 후 제공데이터를 이용하여
학습시켜 주었다. 그 결과를 mae(평균 절대 오차)로
예측값과 기존 값을 비교해 보니 880만원 가량
차이 나는 것을 확인할 수 있다.



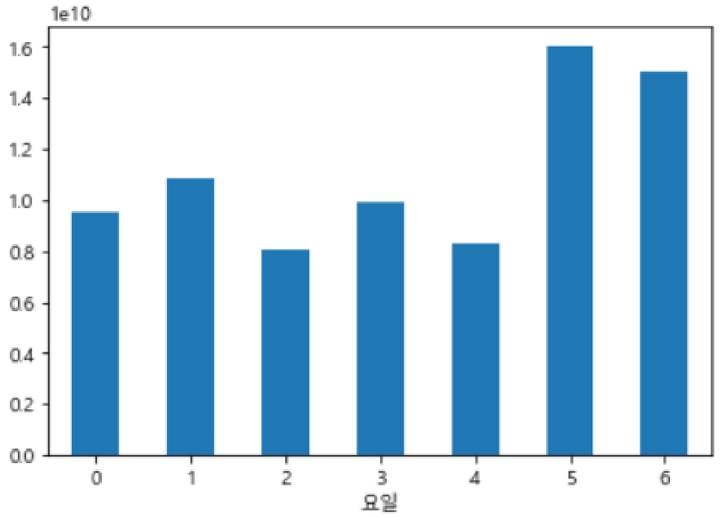
모델의 피처 중요도를 시각화 해보니
시, 일, 월, 요일, 노출(분) 순으로 중요도를 매긴 것을 알 수 있다.

```
pred_t = model.predict(test)
pred_t
```

훈련한 모델을 이용하여 평가 데이터를 예측해 보았다.

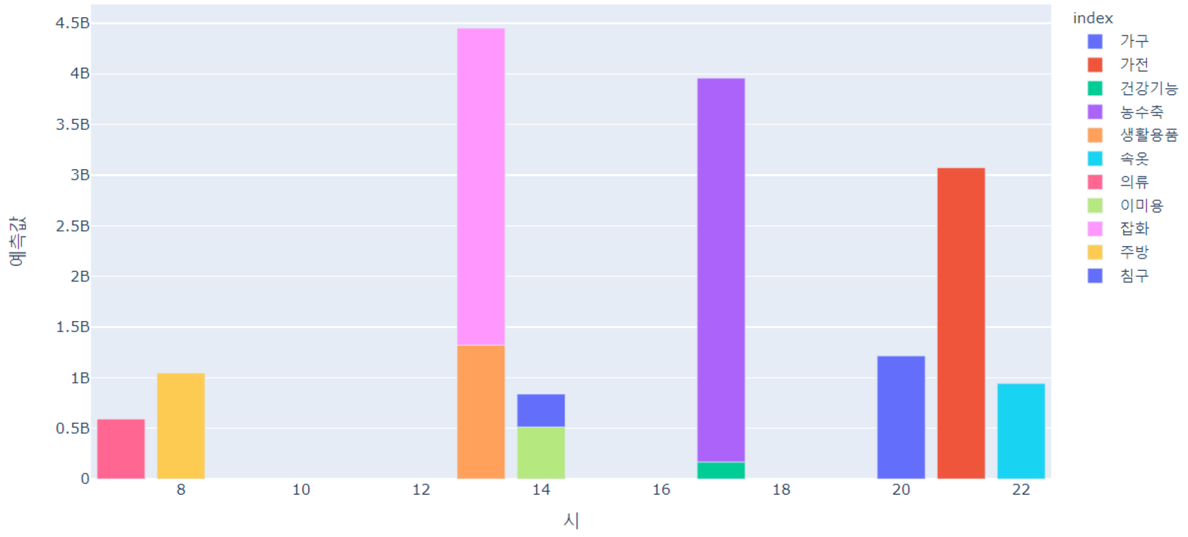


예측 값을 이용하여 일별 총 예상 취급액을
시각화 해보았다.

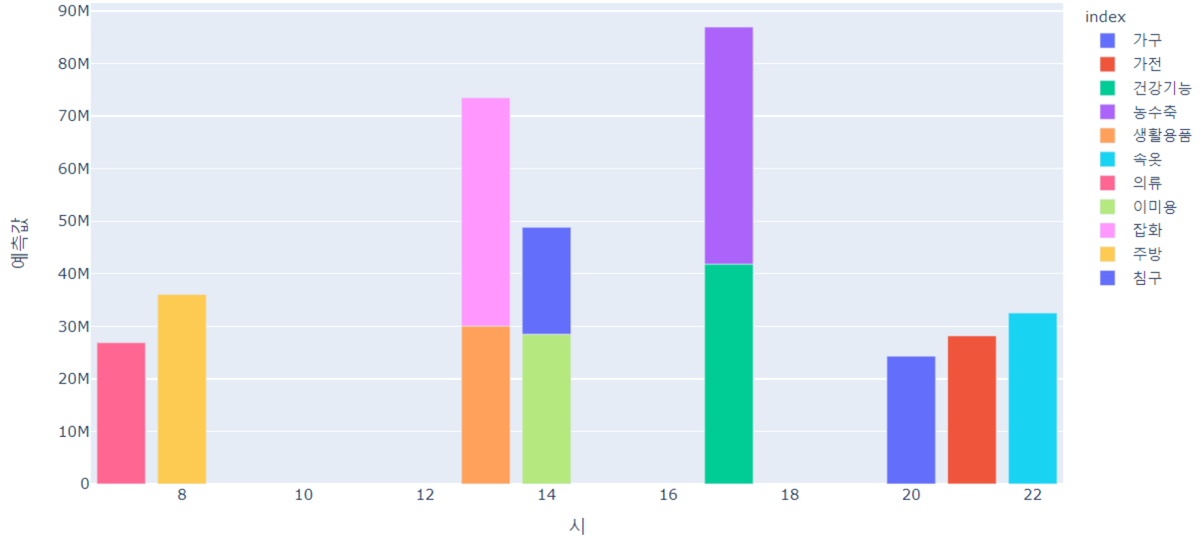


제공 데이터와 동일하게 주말인 토, 일에
총 취급액이 가장 높은 것으로 예상 할 수 있다.

총 취급액



평균 취급액



위 그래프는 어떤 상품군의 상품들이 어느 시간대에 최대 취급액을 달성했는지 보여주는 그래프로,
오전에는 의류 및 주방 관련 상품, 낮 시간대에는 잡화, 생활용품, 이미용, 침구, 건강기능,
농수축 관련상품, 저녁부터 밤까지는 가구, 가전, 속옷 등의 상품의 판매가 잘 될 것으로 예측된다.

활용 방안 및 기대효과

- 총 취급액은 낮지만 평균 취급액이 높은 이미용 상품들을 이미용 상품들의 평균 취급액이 가장 높은 시간인 9시에 추가 배치한다.
- 사람들이 배고파 충동 구매를 할 가능성이 높고 농수축 상품들의 평균 취급액이 가장 높은 시간인 밤 12시에 농수축 상품들 배치를 추가한다.
- 수요는 높지만 공급이 적은 상품들 파악을 통해 고객의 니즈를 충족시킬 수 있고 이는 곧 더 많은 매출을 불러올 수 있을 것이다.