필요한 라이브러리 불러오기

In [1]:
```
import pandas as pd
import numpy as np
import datetime
```

In [14]:
```
import random
```

In [2]:
```
# 특정한 값만 불러오고 싶을 때
from datetime import datetime, timedelta
```

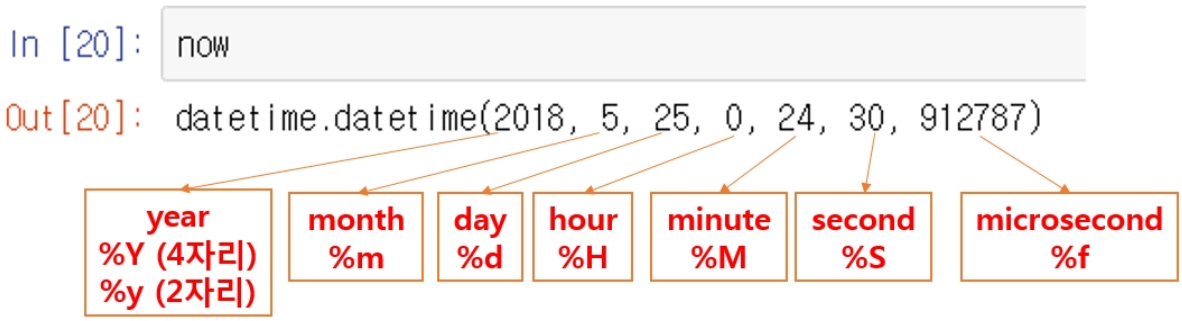
1. 시계열 데이터 소개 (datetime)

In [4]:
```
# 현재 날짜
now = datetime.now()
now
```

Out[4]: datetime.datetime(2020, 5, 14, 19, 12, 13, 770288)

In [5]:
```
# 현재 일
now.day
```

Out[5]: 14



In [6]:
```
# 4주뒤에 날짜 계산
now + timedelta(weeks = 4)
```

Out[6]: datetime.datetime(2020, 6, 11, 19, 12, 13, 770288)

In [7]:
```
# 10일 뒤에 날짜 계산
now + timedelta(days = 10)
```

Out[7]: datetime.datetime(2020, 5, 24, 19, 12, 13, 770288)

In [8]:
```
# 3시간 30분 뒤에 계산
now + timedelta(hours = 3, minutes = 30)
```

Out[8]: datetime.datetime(2020, 5, 14, 22, 42, 13, 770288)

date_range(start, end, periods)

start : 시작 기간

end : 끝 기간

periods : 기간

periods는 start부터 시작해서 periods기간 만큼

freq : 'D' or 'M' or 'Y' or 'W-요일(영어로 앞에 3글자)' or 'B'

In [9]:
```
# freq : 'D'
pd.date_range(start = '19981122', end = '20200514', freq = 'D')
```

Out[9]:
```
DatetimeIndex(['1998-11-22', '1998-11-23', '1998-11-24', '1998-11-25',
               '1998-11-26', '1998-11-27', '1998-11-28', '1998-11-29',
               '1998-11-30', '1998-12-01',
               ...
               '2020-05-05', '2020-05-06', '2020-05-07', '2020-05-08',
               '2020-05-09', '2020-05-10', '2020-05-11', '2020-05-12',
               '2020-05-13', '2020-05-14'],
              dtype='datetime64[ns]', length=7845, freq='D')
```

In [10]:
```
# freq: 'M'
pd.date_range(start = '19981122', end = '20200514', freq = 'M')
```

Out[10]:
```
DatetimeIndex(['1998-11-30', '1998-12-31', '1999-01-31', '1999-02-28',
               '1999-03-31', '1999-04-30', '1999-05-31', '1999-06-30',
               '1999-07-31', '1999-08-31',
               ...
               '2019-07-31', '2019-08-31', '2019-09-30', '2019-10-31',
               '2019-11-30', '2019-12-31', '2020-01-31', '2020-02-29',
               '2020-03-31', '2020-04-30'],
              dtype='datetime64[ns]', length=258, freq='M')
```

In [11]:
```
# freq: 'Y'
len(pd.date_range(start = '19981122', end = '20200514', freq = 'Y'))
```

Out[11]:
```
22
```

In [12]:
```
# freq: 'W-MON'
# 특정 요일
pd.date_range(start = '19981122', end = '20200514', freq = 'W-MON')
```

Out[12]:
```
DatetimeIndex(['1998-11-23', '1998-11-30', '1998-12-07', '1998-12-14',
               '1998-12-21', '1998-12-28', '1999-01-04', '1999-01-11',
               '1999-01-18', '1999-01-25',
               ...
               '2020-03-09', '2020-03-16', '2020-03-23', '2020-03-30',
               '2020-04-06', '2020-04-13', '2020-04-20', '2020-04-27',
               '2020-05-04', '2020-05-11'],
              dtype='datetime64[ns]', length=1121, freq='W-MON')
```

In [13]:
```
# freq: 'B'
pd.date_range(start = '19981122', end = '20200514', freq = 'B')
```

Out[13]:
```
DatetimeIndex(['1998-11-23', '1998-11-24', '1998-11-25', '1998-11-26',
               '1998-11-27', '1998-11-30', '1998-12-01', '1998-12-02',
               '1998-12-03', '1998-12-04',
               ...
               '2020-05-01', '2020-05-04', '2020-05-05', '2020-05-06',
               '2020-05-07', '2020-05-08', '2020-05-11', '2020-05-12',
               '2020-05-13', '2020-05-14'],
              dtype='datetime64[ns]', length=5604, freq='B')
```

In [16]:
```
# 2020년 sample dataframe 생성
sample = pd.DataFrame()
sample['date'] = pd.date_range(start = '20200101', periods=366)
sample['count'] = random.sample(range(1, 1000), 366)
sample
```

Out[16]:

	date	count
0	2020-01-01	39
1	2020-01-02	556
2	2020-01-03	218
3	2020-01-04	260
4	2020-01-05	587
...	...	...
361	2020-12-27	367
362	2020-12-28	519
363	2020-12-29	308
364	2020-12-30	985
365	2020-12-31	564

366 rows × 2 columns

In [17]:
```
# type 확인
sample.dtypes
```

Out[17]:
```
date     datetime64[ns]
count             int64
dtype: object
```

In [20]:
```
# 2020년 5월달만 불러오기
pd.date_range(start = '20200501', end = '20200531')
```

Out[20]:
```
DatetimeIndex(['2020-05-01', '2020-05-02', '2020-05-03', '2020-05-04',
               '2020-05-05', '2020-05-06', '2020-05-07', '2020-05-08',
               '2020-05-09', '2020-05-10', '2020-05-11', '2020-05-12',
               '2020-05-13', '2020-05-14', '2020-05-15', '2020-05-16',
               '2020-05-17', '2020-05-18', '2020-05-19', '2020-05-20',
               '2020-05-21', '2020-05-22', '2020-05-23', '2020-05-24',
               '2020-05-25', '2020-05-26', '2020-05-27', '2020-05-28',
               '2020-05-29', '2020-05-30', '2020-05-31'],
              dtype='datetime64[ns]', freq='D')
```

In [23]:
```
sample.isin(pd.date_range(start = '20200501', end = '20200531'))
```

Out[23]:

	date	count
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
...	...	...
361	False	False
362	False	False
363	False	False
364	False	False
365	False	False

366 rows × 2 columns

In [24]:
```
sample.date.isin(pd.date_range(start = '20200501', end = '20200531'))
```

Out[24]:
```
0       False
1       False
2       False
3       False
4       False
        ...
361     False
362     False
363     False
364     False
365     False
Name: date, Length: 366, dtype: bool
```

In [25]:
```
sample[sample.date.isin(pd.date_range(start = '20200501', end = '20200531'))]
```

Out[25]:

	date	count
121	2020-05-01	700
122	2020-05-02	666
123	2020-05-03	775
124	2020-05-04	209
125	2020-05-05	365
126	2020-05-06	757
127	2020-05-07	381
128	2020-05-08	893
129	2020-05-09	807
130	2020-05-10	487
131	2020-05-11	272

In [29]:
```
# date를 index로 설정하기
# 원본 손상 -> reset 해야 함
# sample.set_index('date', inplace = True)

sample2 = sample.set_index('date')
sample2
```

Out[29]:

date	count
2020-01-01	39
2020-01-02	556
2020-01-03	218
2020-01-04	260
2020-01-05	587
...	...
2020-12-27	367
2020-12-28	519
2020-12-29	308
2020-12-30	985

In [30]:
```
# 2020년 5월만 뽑아오기
sample2['2020-05']
```

Out[30]:

date	count
2020-05-01	700
2020-05-02	666
2020-05-03	775
2020-05-04	209
2020-05-05	365
2020-05-06	757
2020-05-07	381
2020-05-08	893
2020-05-09	807
2020-05-10	487

In [31]:
```
# 2020년 뽑아오기
sample2['2020']
```

Out[31]:

date	count
2020-01-01	39
2020-01-02	556
2020-01-03	218
2020-01-04	260
2020-01-05	587
...	...
2020-12-27	367
2020-12-28	519
2020-12-29	308
2020-12-30	985

In [33]:
```
sample2.loc['2020-03-11']
```

Out[33]:
```
count    705
Name: 2020-03-11 00:00:00, dtype: int64
```

In [35]:
```
sample2.loc['2020-11-22']
```

Out[35]:
```
count    258
Name: 2020-11-22 00:00:00, dtype: int64
```

food_order data

In [36]:
```
# 데이터 불러오기
food = pd.read_excel("data/food_order.xlsx")
```

In [37]:
```
# 상위 10개 데이터 확인하기
food.head(10)
```

Out[37]:

	date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff_
0	2013-08-06	dinner	grill & fry	훈제연어벤또(200개)	F	213	0	0	0	0	0	
1	2013-08-06	dinner	rice & soup 1	뚝배기우거지갈비탕	F	261	0	0	0	0	0	
2	2013-08-06	lunch	noodle bar	판모밀정식	F	267	0	0	0	0	0	
3	2013-08-06	breakfast	rice & soup 1	설렁탕정식	F	1	0	0	0	0	0	
4	2013-08-06	lunch	rice & soup 1	A. 부대찌개정식	F	376	0	0	0	0	0	
5	2013-08-12	dinner	noodle bar	차슈라멘	F	179	0	0	0	0	0	
6	2013-08-12	breakfast	rice & soup 1	뚝배기순대국	F	26	0	0	0	0	0	
7	2013-08-12	lunch	rice & soup 1	A:누룽지장각백숙	F	504	0	0	0	0	0	
8	2013-08-12	lunch	noodle bar	유니자장면	F	167	0	0	0	0	0	
9	2013-08-12	dinner	rice & soup 1	김치날치알밥	F	216	0	0	0	0	0	

In [38]:
```
# 데이터프레임 요약 통계
food.describe()
```

Out[38]:

	use_count	pred_count	additional	good	ok	bad	diff_use_pred	
count	6825.000000	6825.000000	6825.000000	6825.000000	6825.000000	6825.000000	6825.000000	682
mean	189.974652	72.377582	8.569963	6.894799	3.179048	0.792088	-109.027106	201
std	118.012211	99.659671	13.765606	22.679150	10.652367	3.202065	152.218334	
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-787.000000	201
25%	101.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-206.000000	201
50%	175.000000	0.000000	0.000000	0.000000	0.000000	0.000000	-58.000000	201
75%	259.000000	130.000000	20.000000	0.000000	0.000000	0.000000	10.000000	201
max	814.000000	720.000000	110.000000	198.000000	99.000000	52.000000	434.000000	201

In [39]:
```
food.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6825 entries, 0 to 6824
Data columns (total 16 columns):
date             6825 non-null datetime64[ns]
dine_type        6825 non-null object
corner           6825 non-null object
menu             6825 non-null object
is_sold_out      6825 non-null object
use_count        6825 non-null int64
pred_count       6825 non-null int64
additional       6825 non-null int64
good             6825 non-null int64
ok               6825 non-null int64
bad              6825 non-null int64
diff_use_pred    6825 non-null int64
year             6825 non-null int64
month            6825 non-null int64
day              6825 non-null int64
wday             6825 non-null int64
dtypes: datetime64[ns](1), int64(11), object(4)
memory usage: 853.2+ KB
```

In [40]:
```
# date index 설정하기
food.set_index('date', inplace = True)
```

In [41]:
```
food.head()
```

Out[41]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff_use
2013-08-06	dinner	grill & fry	훈제연어벤또(200개)	F	213	0	0	0	0	0	
2013-08-06	dinner	rice & soup 1	뚝배기우거지갈비탕	F	261	0	0	0	0	0	
2013-08-06	lunch	noodle bar	판모밀정식	F	267	0	0	0	0	0	
2013-08-06	breakfast	rice & soup 1	설렁탕정식	F	1	0	0	0	0	0	
2013-08-06	lunch	rice & soup 1	A. 부대찌개정식	F	376	0	0	0	0	0	

In [42]:
```
# 2014년 데이터만 불러오기
food['2014']
```

Out[42]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff
2014-01-02	breakfast	take out	쿡살라미샌드위치	F	51	0	0	0	0	0	
2014-01-02	dinner	noodle bar	오징어비빔국수	F	111	0	0	0	0	0	
2014-01-02	dinner	take out	날치알캘리포니아롤	F	386	0	0	0	0	0	
2014-01-02	lunch	burger&pizza	갈릭오일파스타	T	142	0	0	0	0	0	
2014-01-02	dinner	burger&pizza	참치김치그라	T	151	0	0	0	0	0	

In [43]:
```
# 2015년 1월 데이터 불러오기
food['2015-01']
```

Out[43]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff
2015-01-02	dinner	after school	참깨라면&볶음김치김밥	F	39	80	10	0	0	0	
2015-01-02	lunch	after school	오징어김치볶음밥&계란후라이	F	158	170	40	0	0	0	
2015-01-02	breakfast	rice & soup 1	감자호박국&소고기야채볶음	F	35	20	10	0	0	0	
			청국장								

In [45]:
```
# 14년 8월 1일 부터 14년 9월 25일까지 데이터 불러오기
food['20140801':'20140925']
```

Out[45]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff_use_
2014-08-01	dinner	after school	햄모듬라면&콩도너츠	F	137	0	0	0	0	0	
2014-08-01	lunch	after school	카레라면&비빔채소만두	T	129	0	0	0	0	0	
2014-08-01	dinner	grill & fry	취나물밥&양념장	F	68	0	0	0	0	0	
2014-08-01	breakfast	take out	7곡빵샌드위치	F	39	0	0	0	0	0	

In [46]:
```
# 2015년 6월 30일 데이터 불러오기
food.loc['2015-06-30']
```

Out[46]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff
2015-06-30	dinner	take out	비엔나소시지컵밥	F	116	110	10	0	0	0	
2015-06-30	lunch	take out	콥샐러드	F	281	300	0	3	0	0	
2015-06-30	lunch	rice & soup 1	감자탕	F	230	250	10	63	46	30	
2015-06-30	lunch	after school	짜파게티	F	147	190	0	85	21	2	
2015-06-30	lunch	burger&pizza	콤비네이션피자	F	93	140	0	44	15	3	
2015-	..	...	불고기	F	145	200	0	31	11	0	

In [47]:
```
# index 최소값
food.index.min()
```

Out[47]:
```
Timestamp('2013-08-06 00:00:00')
```

In [48]:
```
# index 최대값
food.index.max()
```

Out[48]:
```
Timestamp('2015-07-27 00:00:00')
```

In [50]:
```
# 월요일만 뽑아오기
monday = pd.date_range(start = food.index.min(),
                       end = food.index.max(),
                       freq = 'W-MON')
monday
```

Out[50]:
```
DatetimeIndex(['2013-08-12', '2013-08-19', '2013-08-26', '2013-09-02',
               '2013-09-09', '2013-09-16', '2013-09-23', '2013-09-30',
               '2013-10-07', '2013-10-14',
               ...
               '2015-05-25', '2015-06-01', '2015-06-08', '2015-06-15',
               '2015-06-22', '2015-06-29', '2015-07-06', '2015-07-13',
               '2015-07-20', '2015-07-27'],
              dtype='datetime64[ns]', length=103, freq='W-MON')
```

In [51]:
```
# tolist() : list로 바꾸기
monday.tolist()
```

Out[51]:
```
[Timestamp('2013-08-12 00:00:00', freq='W-MON'),
 Timestamp('2013-08-19 00:00:00', freq='W-MON'),
 Timestamp('2013-08-26 00:00:00', freq='W-MON'),
 Timestamp('2013-09-02 00:00:00', freq='W-MON'),
 Timestamp('2013-09-09 00:00:00', freq='W-MON'),
 Timestamp('2013-09-16 00:00:00', freq='W-MON'),
 Timestamp('2013-09-23 00:00:00', freq='W-MON'),
 Timestamp('2013-09-30 00:00:00', freq='W-MON'),
 Timestamp('2013-10-07 00:00:00', freq='W-MON'),
 Timestamp('2013-10-14 00:00:00', freq='W-MON'),
 Timestamp('2013-10-21 00:00:00', freq='W-MON'),
 Timestamp('2013-10-28 00:00:00', freq='W-MON'),
 Timestamp('2013-11-04 00:00:00', freq='W-MON'),
 Timestamp('2013-11-11 00:00:00', freq='W-MON'),
 Timestamp('2013-11-18 00:00:00', freq='W-MON'),
 Timestamp('2013-11-25 00:00:00', freq='W-MON'),
 Timestamp('2013-12-02 00:00:00', freq='W-MON'),
 Timestamp('2013-12-09 00:00:00', freq='W-MON'),
 Timestamp('2013-12-16 00:00:00', freq='W-MON'),
```

In [52]:
```
# 매주 월요일 데이터만 불러오기
food[food.index.isin(monday)]
```

Out[52]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff_use_
2013-08-12	dinner	noodle bar	차슈라멘	F	179	0	0	0	0	0	
2013-08-12	breakfast	rice & soup 1	뚝배기순대국	F	26	0	0	0	0	0	
2013-08-12	lunch	rice & soup 1	A:누룽지장각백숙	F	504	0	0	0	0	0	
2013-08-12	lunch	noodle bar	유니자장면	F	167	0	0	0	0	0	
2013-08-12	dinner	rice & soup 1	김치날치알밥	F	216	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...
			돼지고								

In [53]:
```
# wday: 월화수목금을 숫자로 01234표시
food['wday'].value_counts()
```

Out[53]:
```
0    1433
1    1432
2    1373
3    1298
4    1289
Name: wday, dtype: int64
```

In [59]:
```
# 평일만 불러오기
weekday = pd.date_range(start = food.index.min(),
                        end = food.index.max(),
                        freq = 'B')
weekday
```

Out[59]:
```
DatetimeIndex(['2013-08-06', '2013-08-07', '2013-08-08', '2013-08-09',
               '2013-08-12', '2013-08-13', '2013-08-14', '2013-08-15',
               '2013-08-16', '2013-08-19',
               ...
               '2015-07-14', '2015-07-15', '2015-07-16', '2015-07-17',
               '2015-07-20', '2015-07-21', '2015-07-22', '2015-07-23',
               '2015-07-24', '2015-07-27'],
              dtype='datetime64[ns]', length=515, freq='B')
```

In [60]:
```
# 실제 데이터 날짜 길이
food.index.unique().size
```

Out[60]:
```
480
```

In [61]:
```
food.index.unique()
```

Out[61]:
```
DatetimeIndex(['2013-08-06', '2013-08-12', '2013-08-13', '2013-08-14',
               '2013-08-19', '2013-08-20', '2013-08-21', '2013-08-22',
               '2013-08-23', '2013-08-26',
               ...
               '2015-07-14', '2015-07-15', '2015-07-16', '2015-07-17',
               '2015-07-20', '2015-07-21', '2015-07-22', '2015-07-23',
               '2015-07-24', '2015-07-27'],
              dtype='datetime64[ns]', name='date', length=480, freq=None)
```

In [62]:
```
# 평일 날짜 길이
weekday.size
```

Out[62]:
```
515
```

In [63]:
```
# 포함되지 않는 날짜 확인
weekday[~weekday.isin(food.index.unique())]
```

Out[63]:
```
DatetimeIndex(['2013-08-07', '2013-08-08', '2013-08-09', '2013-08-15',
               '2013-08-16', '2013-09-18', '2013-09-19', '2013-09-20',
               '2013-10-03', '2013-10-09', '2013-12-25', '2014-01-01',
               '2014-01-30', '2014-01-31', '2014-04-04', '2014-05-01',
               '2014-05-05', '2014-05-06', '2014-06-04', '2014-06-06',
               '2014-08-15', '2014-09-08', '2014-09-09', '2014-09-10',
               '2014-10-03', '2014-10-09', '2014-12-25', '2015-01-01',
               '2015-02-18', '2015-02-19', '2015-02-20', '2015-04-03',
               '2015-05-01', '2015-05-05', '2015-05-25'],
              dtype='datetime64[ns]', freq=None)
```

In [64]:
```
# groupby를 사용하여 date별로 use_count의 합 구하기
food.groupby('date')['use_count'].sum()
```

Out[64]:
```
date
2013-08-06    1118
2013-08-12    1787
2013-08-13    1714
2013-08-14    1450
2013-08-19    2647
              ...
2015-07-21    2737
2015-07-22    2755
2015-07-23    2480
2015-07-24    2193
2015-07-27    1901
Name: use_count, Length: 480, dtype: int64
```

In [66]:
```
# column이 하나인 경우 시각적으로 잘 보이게 DataFrame으로 씌워줌
pd.DataFrame(food.groupby('date')['use_count'].sum())
```

Out[66]:

date	use_count
2013-08-06	1118
2013-08-12	1787
2013-08-13	1714
2013-08-14	1450
2013-08-19	2647
...	...
2015-07-21	2737
2015-07-22	2755
2015-07-23	2480
2015-07-24	2193
2015-07-27	1901

480 rows × 1 columns

In [65]:
```
# pivot_table을 사용하여 date별로 use_count의 합 구하기
food.pivot_table(index = 'date', aggfunc = 'sum', values = 'use_count')
```

Out[65]:

date	use_count
2013-08-06	1118
2013-08-12	1787
2013-08-13	1714
2013-08-14	1450
2013-08-19	2647
...	...
2015-07-21	2737
2015-07-22	2755
2015-07-23	2480
2015-07-24	2193

실습

1. 전체 기간 동안, 아침, 점심, 저녁 메뉴로 가장 인기 있었던 메뉴를 각각 착기

In [74]:
```
food.groupby('dine_type').max()
```

Out[74]:

dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	diff_use_pred
breakfast	take out	후레쉬 듬뿍햄 샌드위치	F	101	75	20	34	10	5	49
dinner	take out	흑미밥 &콩나물국	T	531	270	110	108	40	33	252
lunch	take out	흰살생선스테이크& 크림소스	T	814	720	100	198	99	52	434

In [77]:
```
food.groupby(['dine_type', 'menu'])['use_count'].sum()
```

Out[77]:
```
dine_type  menu
breakfast  (뚝)설렁탕                          66
           7곡빵샌드위치                       220
           A: NC 모닝샌드위치                   61
           A: NC모닝샌드위치(542kcal)           71
           A: 닭가슴살브로컬리샌드위치(582kcal)     62
                                             ...
lunch      훈제오리와부추생채                  1147
           훈제오리채소볶음                     207
           흑미버거                            328
           흑임자두부스테이크&생선가스              165
           흰살생선스테이크&크림소스                258
Name: use_count, Length: 3250, dtype: int64
```

In [78]:
```
pd.DataFrame(food.groupby(['dine_type', 'menu'])['use_count'].sum())
```

Out[78]:

		use_count
dine_type	**menu**	
	(뚝)설렁탕	66
	7곡빵샌드위치	220
breakfast	A: NC 모닝샌드위치	61
	A: NC모닝샌드위치(542kcal)	71
	A: 닭가슴살브로컬리샌드위치(582kcal)	62
...	...	...
	훈제오리와부추생채	1147
	훈제오리채소볶음	207
lunch	흑미버거	328
	흑임자두부스테이크&생선가스	165
	흰살생선스테이크&크림소스	258

3250 rows × 1 columns

In [81]:
```
temp = pd.DataFrame(food.groupby(['dine_type', 'menu'])['use_count'].sum().unst
temp
```

Out[81]:

dine_type	breakfast	dinner	lunch
menu			
참치롤밥 450	NaN	470.0	NaN
해물볶음밥컵밥	NaN	NaN	157.0
(뚝)감자고추장찌개	NaN	344.0	257.0
(뚝)감자탕	NaN	551.0	NaN
(뚝)고등어무조림	NaN	NaN	212.0
...	...	...	...
흑미밥&찜닭	NaN	90.0	NaN
흑미밥&콩나물국	NaN	44.0	NaN
흑미버거	NaN	NaN	328.0
흑임자두부스테이크&생선가스	NaN	NaN	165.0
흰살생선스테이크&크림소스	NaN	NaN	258.0

2776 rows × 3 columns

In [82]:
```
# 아침
temp.sort_values(by = 'breakfast', ascending = False)
```

Out[82]:

dine_type	breakfast	dinner	lunch
menu			
샌드위치콤보&음료&과일	7754.0	NaN	NaN
A: 샌드위치 SET 510kcal	1593.0	NaN	NaN
A: 샌드위치 SET 510	1377.0	NaN	NaN
샌드위치콤보&음료&사과	1202.0	NaN	NaN
NC모닝	1198.0	NaN	NaN
...	...	...	...
흑미밥&찜닭	NaN	90.0	NaN
흑미밥&콩나물국	NaN	44.0	NaN
흑미버거	NaN	NaN	328.0
흑임자두부스테이크&생선가스	NaN	NaN	165.0

In [83]:
```
# 점심
temp.sort_values(by = 'lunch', ascending = False)
```

Out[83]:

dine_type	breakfast	dinner	lunch
menu			
콥샐러드	NaN	NaN	13747.0
훈제오리&단호박샐러드	NaN	NaN	13252.0
돈가스샐러드	NaN	NaN	11279.0
연어레몬샐러드	NaN	NaN	8628.0
리코타치즈&치아바타샐러드	NaN	NaN	8041.0
...	...	...	...
흑미밥&돼지갈비찜	NaN	195.0	NaN
흑미밥&돼지고기사태편육	NaN	87.0	NaN
흑미밥&쇠고기청경채굴소스볶음	NaN	67.0	NaN
흑미밥&찜닭	NaN	90.0	NaN
흑미밥&콩나물국	NaN	44.0	NaN

2776 rows × 3 columns

In [85]:
```
# 저녁
temp.sort_values(by = 'dinner', ascending = False)
```

Out[85]:

dine_type	breakfast	dinner	lunch
menu			
로제컵파스타&새우튀김	NaN	7560.0	NaN
미트볼라이스	NaN	6315.0	NaN
토마토미트볼컵파스타	NaN	5391.0	NaN
비엔나소시지컵밥*스크램블에그	NaN	3742.0	NaN
참치김치찌개	383.0	3692.0	5175.0
...	...	...	...
훈제오리와부추생채	NaN	NaN	1147.0
훈제오리채소볶음	NaN	NaN	207.0
흑미버거	NaN	NaN	328.0
흑임자두부스테이크&생선가스	NaN	NaN	165.0
흰살생선스테이크&크림소스	NaN	NaN	258.0

2776 rows × 3 columns

2. 2014년 한 해 동안, 각 코너 별 월별 판매량 구해보기

In [84]:
```
# 2014년 데이터 불러오기
food2014 = food['2014']
```

In [88]:
```
food2014.pivot_table(index = 'corner', columns = 'month', aggfunc = 'sum', valu
```

Out[88]:

month	1	2	3	4	5	6	7	8	9	10	
corner											
after school	6799.0	6731.0	7180.0	6826.0	5220.0	5389.0	6702.0	5161.0	4276.0	5014.0	4
burger&pizza	5549.0	5497.0	6447.0	6145.0	4505.0	4898.0	7228.0	6436.0	5855.0	6441.0	5
grill & fry	8896.0	8591.0	9316.0	9218.0	8317.0	8684.0	11008.0	8740.0	7600.0	8261.0	8
noodle bar	7985.0	7437.0	7622.0	7159.0	6614.0	6772.0	9254.0	7229.0	5751.0	6303.0	6
rice & soup 1	19027.0	16191.0	12832.0	11837.0	9877.0	11046.0	13793.0	11623.0	9232.0	11550.0	10
rice & soup 2	NaN	798.0	5148.0	4820.0	4665.0	4649.0	6052.0	5368.0	4272.0	4222.0	4
take out	14963.0	14499.0	15634.0	15354.0	13481.0	13263.0	15816.0	11125.0	6540.0	7907.0	7

3. 매진을 가장 많이 기록한 상위 10개 메뉴와 매진횟수를 출력

In [89]:
```
food['is_sold_out']
```

Out[89]:
```
date
2013-08-06    F
2013-08-06    F
2013-08-06    F
2013-08-06    F
2013-08-06    F
              ..
2015-07-27    F
2015-07-27    F
2015-07-27    F
2015-07-27    F
2015-07-27    F
Name: is_sold_out, Length: 6825, dtype: object
```

In [90]:
```
# 매진을 기록한 것만 불러옴
food_sold_out = food[food.is_sold_out == 'T']
food_sold_out
```

Out[90]:

date	dine_type	corner	menu	is_sold_out	use_count	pred_count	additional	good	ok	bad	di
2013-10-23	dinner	noodle bar	바지락칼국수	T	170	0	0	0	0	0	
2013-10-28	dinner	after school	어묵떡볶이 set	T	147	0	0	0	0	0	
2013-10-28	dinner	burger&pizza	마늘새우볶음밥	T	138	0	0	0	0	0	
2013-10-29	dinner	burger&pizza	더블버거	T	120	0	0	0	0	0	
2013-10-29	lunch	after school	떡만두라면 SET	T	175	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...
2015-07-15	dinner	burger&pizza	새우또디아랩	T	90	80	10	18	10	6	
2015-07-20	dinner	burger&pizza	라자냐	T	159	0	0	47	7	4	
2015-07-23	lunch	grill & fry	*인도요리*	T	307	0	0	127	9	5	
2015-07-24	dinner	rice & soup 1	돼지고기고추장볶음	T	148	0	0	0	0	0	
2015-07-27	lunch	burger&pizza	매콤돈가스	T	282	0	0	0	0	0	

1184 rows × 15 columns

In [92]:
```
food_sold_out['menu'].value_counts()
```

Out[92]:
```
치즈돈가스              14
등심돈가스              13
김치볶음밥&계란후라이      11
고구마돈가스              9
고르곤졸라피자             9
                       ..
왕새우튀김덮밥             1
햄김치라면               1
새우크림파스타             1
설렁탕&소면              1
해물토마토라이스그라탕       1
Name: menu, Length: 797, dtype: int64
```