캐글의 Store Sales - Time Series Forecasting 데이터 전처리

데이터: https://www.kaggle.com/competitions/store-sales-time-series-forecasting/code?competitionId=29781&sortBy=commentCount

데이터 전처리 참고 링크

https://www.kaggle.com/code/ekrembayar/store-sales-ts-forecasting-a-comprehensive-guide#3.-Transactions

0. 패키지 불러오기

```
# BASE
# ------------------------------------------------------
import numpy as np
import pandas as pd
import os
import gc
import warnings

# PACF - ACF
# ------------------------------------------------------
import statsmodels.api as sm

# DATA VISUALIZATION
# ------------------------------------------------------
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px


# CONFIGURATIONS
# ------------------------------------------------------
pd.set_option('display.max_columns', None)
pd.options.display.float_format = '{:.2f}'.format
warnings.filterwarnings('ignore')
```

1. 데이터 불러오기

- **데이터 소개**
- 애콰도르에 기반을 둔 대형 식료품 소매 업체 Corporación Favorita의 상점 매출 데이터 > 시계열 예측하기
- 날빠, 매장 및 품목 정보, 프로모션 및 단위 매출에 대한 학습 데이터 세트

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
holidays = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/holidays_events.csv')
oil = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/oil.csv')
stores = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/stores.csv')
transactions = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/transactions.csv')
train = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/train.csv')
test = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/test.csv')
```

```
# Datetime 타입으로 변경하기
train["date"] = pd.to_datetime(train.date)
test["date"] = pd.to_datetime(test.date)
transactions["date"] = pd.to_datetime(transactions.date)
```

```
# merge 가능하도록 data type 고정
train.onpromotion = train.onpromotion.astype("float16")
train.sales = train.sales.astype("float32")
stores.cluster = stores.cluster.astype("int8")
```

1. holidays

```
holidays = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/holidays_events.csv')
```

```
holidays.head()
```

	date	type	locale	locale_name	description	transferred
0	2012-03-02	Holiday	Local	Manta	Fundacion de Manta	False
1	2012-04-01	Holiday	Regional	Cotopaxi	Provincializacion de Cotopaxi	False
2	2012-04-12	Holiday	Local	Cuenca	Fundacion de Cuenca	False
3	2012-04-				Cantonizacion de	

```
holidays.type.unique()
# 실제로 쉬는 날에만 영향이 있지 않을까? work day때 유의미한가???
# 수빈님 데이터 구조 참고...
```

```
array(['Holiday', 'Transfer', 'Additional', 'Bridge', 'Work Day', 'Event'],
      dtype=object)
```

```
holidays.locale.unique()
# National은 가게 전체에, regional과 Local은 그 지역에 있는 가게에만 적용해야 할 것 같음...
```

```
array(['Local', 'Regional', 'National'], dtype=object)
```

```
holidays.locale_name.unique()
```

```
array(['Manta', 'Cotopaxi', 'Cuenca', 'Libertad', 'Riobamba', 'Puyo',
       'Guaranda', 'Imbabura', 'Latacunga', 'Machala', 'Santo Domingo',
       'El Carmen', 'Cayambe', 'Esmeraldas', 'Ecuador', 'Ambato',
       'Ibarra', 'Quevedo', 'Santo Domingo de los Tsachilas',
       'Santa Elena', 'Quito', 'Loja', 'Salinas', 'Guayaquil'],
      dtype=object)
```

```
holidays.description.unique()
# 아마도 지역명.....과의 결합??
# 설명이니 이후 drop해주기
```

```
array(['Fundacion de Manta', 'Provincializacion de Cotopaxi',
       'Fundacion de Cuenca', 'Cantonizacion de Libertad',
       'Cantonizacion de Riobamba', 'Cantonizacion del Puyo',
       'Cantonizacion de Guaranda', 'Provincializacion de Imbabura',
       'Cantonizacion de Latacunga', 'Fundacion de Machala',
       'Fundacion de Santo Domingo', 'Cantonizacion de El Carmen',
       'Cantonizacion de Cayambe', 'Fundacion de Esmeraldas',
       'Primer Grito de Independencia', 'Fundacion de Riobamba',
       'Fundacion de Ambato', 'Fundacion de Ibarra',
       'Cantonizacion de Quevedo', 'Independencia de Guayaquil',
       'Traslado Independencia de Guayaquil', 'Dia de Difuntos',
       'Independencia de Cuenca', 'Provincializacion de Santo Domingo',
       'Provincializacion Santa Elena', 'Independencia de Guaranda',
       'Independencia de Latacunga', 'Independencia de Ambato',
       'Fundacion de Quito-1', 'Fundacion de Quito', 'Fundacion de Loja',
       'Navidad-4', 'Cantonizacion de Salinas', 'Navidad-3', 'Navidad-2',
       'Puente Navidad', 'Navidad-1', 'Navidad', 'Navidad+1',
       'Puente Primer dia del ano', 'Primer dia del ano-1',
       'Primer dia del ano', 'Recupero puente Navidad',
       'Recupero puente primer dia del ano', 'Carnaval', 'Viernes Santo',
       'Dia del Trabajo', 'Dia de la Madre-1', 'Dia de la Madre',
       'Batalla de Pichincha', 'Fundacion de Guayaquil-1',
       'Fundacion de Guayaquil', 'Inauguracion Mundial de futbol Brasil',
       'Mundial de futbol Brasil: Ecuador-Suiza',
       'Mundial de futbol Brasil: Ecuador-Honduras',
       'Mundial de futbol Brasil: Ecuador-Francia',
       'Mundial de futbol Brasil: Octavos de Final',
       'Mundial de futbol Brasil: Cuartos de Final',
       'Mundial de futbol Brasil: Semifinales',
       'Mundial de futbol Brasil: Tercer y cuarto lugar',
       'Mundial de futbol Brasil: Final', 'Black Friday', 'Cyber Monday',
       'Recupero Puente Navidad', 'Recupero Puente Primer dia del ano',
       'Terremoto Manabi', 'Terremoto Manabi+1', 'Terremoto Manabi+2',
       'Terremoto Manabi+3', 'Terremoto Manabi+4', 'Terremoto Manabi+5',
       'Terremoto Manabi+6', 'Terremoto Manabi+7', 'Terremoto Manabi+8',
       'Terremoto Manabi+9', 'Terremoto Manabi+10', 'Terremoto Manabi+11',
       'Terremoto Manabi+12', 'Terremoto Manabi+13',
       'Terremoto Manabi+14', 'Terremoto Manabi+15',
       'Terremoto Manabi+16', 'Terremoto Manabi+17',
       'Terremoto Manabi+18', 'Terremoto Manabi+19',
       'Terremoto Manabi+20', 'Terremoto Manabi+21',
       'Terremoto Manabi+22', 'Terremoto Manabi+23',
       'Terremoto Manabi+24', 'Terremoto Manabi+25',
       'Terremoto Manabi+26', 'Terremoto Manabi+27',
```

```
       'Terremoto Manabi+28', 'Terremoto Manabi+29',
       'Terremoto Manabi+30', 'Traslado Batalla de Pichincha',
       'Traslado Fundacion de Guayaquil',
       'Traslado Primer Grito de Independencia', 'Puente Dia de Difuntos',
       'Recupero Puente Dia de Difuntos', 'Traslado Primer dia del ano',
       'Traslado Fundacion de Quito'], dtype=object)
```

```
# 결측치 확인하기
holidays.isnull().sum()
```

```
date           0
type           0
locale         0
locale_name    0
description    0
transferred    0
dtype: int64
```

```
# datetime으로 바꾸기
holidays["date"] = pd.to_datetime(holidays.date)
```

```
# 옮겨진 기념일 확인하기 (실제로 기념 X)
holidays[holidays['transferred']==True]
```

	date	type	locale	locale_name	description	transferred
19	2012-10-09	Holiday	National	Ecuador	Independencia de Guayaquil	True
72	2013-10-09	Holiday	National	Ecuador	Independencia de Guayaquil	True
135	2014-10-09	Holiday	National	Ecuador	Independencia de Guayaquil	True
255	2016-05-24	Holiday	National	Ecuador	Batalla de Pichincha	True
266	2016-07-25	Holiday	Local	Guayaquil	Fundacion de Guayaquil	True
268	2016-08-10	Holiday	National	Ecuador	Primer Grito de Independencia	True
297	2017-01-01	Holiday	National	Ecuador	Primer dia del ano	True
303	2017-04-12	Holiday	Local	Cuenca	Fundacion de Cuenca	True

```
# Transferred Holidays
# tr1 > 실제로는 휴일이 아닌 날
tr1 = holidays[(holidays.type == "Holiday") & (holidays.transferred == True)].drop("transferred", axis = 1).reset_index(drop = True)
# tr2 > type에서 'Transfer'인 날, 실제로 기념됨
tr2 = holidays[(holidays.type == "Transfer")].drop("transferred", axis = 1).reset_index(drop = True)
tr2.head()
```

	date	type	locale	locale_name	description
0	2012-10-12	Transfer	National	Ecuador	Traslado Independencia de Guayaquil
1	2013-10-11	Transfer	National	Ecuador	Traslado Independencia de Guayaquil
2	2014-10-10	Transfer	National	Ecuador	Traslado Independencia de Guayaquil
3	2016-05-27	Transfer	National	Ecuador	Traslado Batalla de Pichincha
4	2016-07-24	Transfer	Local	Guayaquil	Traslado Fundacion de Guayaquil

```
# 실제로 휴일인 날만 모아보자!
holidays = holidays[(holidays.transferred == False)].drop("transferred", axis = 1)
```

```
holidays.head()
```

	date	type	locale	locale_name	description
0	2012-03-02	Holiday	Local	Manta	Fundacion de Manta
1	2012-04-01	Holiday	Regional	Cotopaxi	Provincializacion de Cotopaxi
2	2012-04-12	Holiday	Local	Cuenca	Fundacion de Cuenca
3	2012-04-14	Holiday	Local	Libertad	Cantonizacion de Libertad
4	2012-04-21	Holiday	Local	Riobamba	Cantonizacion de Riobamba

```
# work day는 원래 토요일 등 근무하지 않는 날이나, bridge의 휴일을 보충하기 위한 날
# work day는 따로 분리, bridge 유지!
work_day = holidays[holidays.type == "Work Day"]
holidays = holidays[holidays.type != "Work Day"]
```

```
# Additional Holidays > 일반적인 연휴(기념되는 날에 1일 더 쉬는 등...)
# "description"에 그와 관련된 묘사 적혀있음! > 유지!
```

```
holidays[holidays['type']=='Additional']
```

150	2014-12-21	Additional	National	Ecuador	Navidad-4
152	2014-12-22	Additional	National	Ecuador	Navidad-3
153	2014-12-23	Additional	National	Ecuador	Navidad-2
154	2014-12-24	Additional	National	Ecuador	Navidad-1
157	2014-12-26	Additional	National	Ecuador	Navidad+1
158	2014-12-31	Additional	National	Ecuador	Primer dia del ano-1
171	2015-05-09	Additional	National	Ecuador	Dia de la Madre-1
200	2015-12-05	Additional	Local	Quito	Fundacion de Quito-1
203	2015-12-21	Additional	National	Ecuador	Navidad-4
204	2015-12-22	Additional	National	Ecuador	Navidad-3
206	2015-12-23	Additional	National	Ecuador	Navidad-2
207	2015-12-24	Additional	National	Ecuador	Navidad-1
209	2015-12-26	Additional	National	Ecuador	Navidad+1
210	2015-12-31	Additional	National	Ecuador	Primer dia del ano-1
242	2016-05-07	Additional	National	Ecuador	Dia de la Madre-1
264	2016-07-24	Additional	Local	Guayaquil	Fundacion de Guayaquil-1
286	2016-12-05	Additional	Local	Quito	Fundacion de Quito-1
289	2016-12-21	Additional	National	Ecuador	Navidad-4
290	2016-12-22	Additional	National	Ecuador	Navidad-3
292	2016-12-23	Additional	National	Ecuador	Navidad-2
293	2016-12-24	Additional	National	Ecuador	Navidad-1
295	2016-12-26	Additional	National	Ecuador	Navidad+1
296	2016-12-31	Additional	National	Ecuador	Primer dia del ano-1
310	2017-05-13	Additional	National	Ecuador	Dia de la Madre-1
321	2017-07-24	Additional	Local	Guayaquil	Fundacion de Guayaquil-1
322	2017-07-25	Additional	Local	Guayaquil	Fundacion de Guayaquil
339	2017-12-05	Additional	Local	Quito	Fundacion de Quito-1
343	2017-12-21	Additional	National	Ecuador	Navidad-4
345	2017-12-22	Additional	National	Ecuador	Navidad-3
346	2017-12-23	Additional	National	Ecuador	Navidad-2
347	2017-12-24	Additional	National	Ecuador	Navidad-1
349	2017-12-26	Additional	National	Ecuador	Navidad+1

```
# Events: 국경일
events = holidays[holidays.type == "Event"].drop(["type", "locale", "locale_name", 'description'], axis = 1).rename({"description":"events"}, axis
# 국경일인지, 휴일인지, 지역적인지 국가적인지 나누기!
holidays = holidays[holidays.type != "Event"].drop('description', axis=1)
regional = holidays[holidays.locale == "Regional"].drop(["type","locale"], axis = 1)
national = holidays[holidays.locale == "National"].drop(["type","locale", "locale_name"], axis = 1)
local = holidays[holidays.locale == "Local"].drop(["type","locale"], axis = 1)
```

```
regional.head()
```

	date	locale_name
1	2012-04-01	Cotopaxi
7	2012-06-25	Imbabura
23	2012-11-06	Santo Domingo de los Tsachilas
24	2012-11-07	Santa Elena
47	2013-04-01	Cotopaxi

2. oil

- 오일 가격은 이송 가격에 영향을 미치며, 즉 오일 가격이 높아질수록 상품들의 가격에, 특히 수입 상품들의 가격에 영향을 미칠 수 있음을 짐작할 수 있다.

```
oil = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/oil.csv')
oil.head()
```

	date	dcoilwtico
0	2013-01-01	NaN
1	2013-01-02	93.14
2	2013-01-03	92.97
3	2013-01-04	93.12
4	2013-01-07	93.20

```
# 마지막 날짜 확인하기
oil.tail()
```

	date	dcoilwtico
1213	2017-08-25	47.65
1214	2017-08-28	46.40
1215	2017-08-29	46.46
1216	2017-08-30	45.96
1217	2017-08-31	47.26

```
# 데이터 타입 확인하기
oil.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1218 entries, 0 to 1217
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   date        1218 non-null   object 
 1   dcoilwtico  1175 non-null   float64
dtypes: float64(1), object(1)
memory usage: 19.2+ KB
```

```
# Date로 데이터 타입 바꾸어주기
oil["date"] = pd.to_datetime(oil.date)
oil.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1218 entries, 0 to 1217
```

```
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   date        1218 non-null   datetime64[ns]
 1   dcoilwtico  1175 non-null   float64       
dtypes: datetime64[ns](1), float64(1)
memory usage: 19.2 KB
```

결측치 처리하기

```
# 결측치 확인하기
oil.isnull().sum()
```

```
date          0
dcoilwtico    43
dtype: int64
```

결측치의 위치 확인하기

- 1월 1일, 12월 25일 등 국제적인 휴일에 오일 가격이 책정되어 있지 않음을 확인할 수 있다.

```
# 결측치 위치 확인하기
oil[oil['dcoilwtico'].isnull()==True]
```

294	2014-02-17	NaN
338	2014-04-18	NaN
364	2014-05-26	NaN
393	2014-07-04	NaN
434	2014-09-01	NaN
497	2014-11-27	NaN
517	2014-12-25	NaN
522	2015-01-01	NaN
534	2015-01-19	NaN
554	2015-02-16	NaN
588	2015-04-03	NaN
624	2015-05-25	NaN
653	2015-07-03	NaN
699	2015-09-07	NaN
757	2015-11-26	NaN
778	2015-12-25	NaN
783	2016-01-01	NaN
794	2016-01-18	NaN
814	2016-02-15	NaN
843	2016-03-25	NaN
889	2016-05-30	NaN
914	2016-07-04	NaN
959	2016-09-05	NaN
1017	2016-11-24	NaN
1039	2016-12-26	NaN
1044	2017-01-02	NaN
1054	2017-01-16	NaN
1079	2017-02-20	NaN
1118	2017-04-14	NaN
1149	2017-05-29	NaN
1174	2017-07-03	NaN
1175	2017-07-04	NaN

결측치 '보간'하기

- 근접한 날짜와 오일 가격은 밀접한 관련이 있는 이런 시계열 데이터에서, 결측치를 전체 평균값이 아닌, 이어지듯이, 즉 그라데이션처럼 결측값을 채워나가는 것을 의미한다.
- 그 중, 시계열데이터의 값에 선형으로 비례하는 방식으로 결측값 보간을 보간하기 위해서는, Python pandas의 interpolate()를 사용하면 된다.

- https://rfriend.tistory.com/264 참조.

```
# Interpolate
oil["dcoilwtico"] = np.where(oil["dcoilwtico"] == 0, np.nan, oil["dcoilwtico"])
oil["dcoilwtico_interpolated"] =oil.dcoilwtico.interpolate()
```

```
# 결측치를 채운 값으로 그래프 그려보기
p = oil.melt(id_vars=['date']+list(oil.keys()[5:]), var_name='Legend')
px.line(p.sort_values(["Legend", "date"], ascending = [False, True]), x='date', y='value', color='Legend',title = "Daily Oil Price" )
```

> 그라데이션으로 잘 이어진 것을 확인가능하다!

```
# 결측치 존재하는지 다시 한번 확인하기
oil[oil["dcoilwtico_interpolated"].isnull()]
```

	date	dcoilwtico	dcoilwtico_interpolated
0	2013-01-01	NaN	NaN

```
# 결측치 처리해주기 - 같은 년도의 바로 다음 날짜의 것으로!
oil.dcoilwtico_interpolated[oil["dcoilwtico_interpolated"].isnull()==True] = 93.14
```

```
oil.head()
```

	date	dcoilwtico	dcoilwtico_interpolated
0	2013-01-01	NaN	93.14
1	2013-01-02	93.14	93.14
2	2013-01-03	92.97	92.97
3	2013-01-04	93.12	93.12
4	2013-01-07	93.20	93.20

> 보다시피, date 사이에 빈 날짜들도 존재!

> merge 이후 이것도 결측치로 처리되어 보간이 필요해보인다.

'sales'와의 상관관계 확인하기

```
oil = oil.drop('dcoilwtico', axis=1)
oil.head()
```

	date	dcoilwtico_interpolated
0	2013-01-01	93.14
1	2013-01-02	93.14
2	2013-01-03	92.97
3	2013-01-04	93.12
4	2013-01-07	93.20

```
oil["date"] = pd.to_datetime(oil.date)
oil.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1218 entries, 0 to 1217
Data columns (total 2 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   date                    1218 non-null   datetime64[ns]
 1   dcoilwtico_interpolated  1218 non-null   float64
dtypes: datetime64[ns](1), float64(1)
memory usage: 19.2 KB
```

```
# train의 sales와의 상관관계 확인하기
# date대로 groupby후 merge 해주기
temp = pd.merge(train.groupby(["date"]).sales.sum().reset_index(), oil, how = "left")
print("Spearman Correlation between Total Sales and oil price: {:,.4f}".format(temp.corr("spearman").sales.loc["dcoilwtico_interpolated"]))
```

```
Spearman Correlation between Total Sales and oil price: -0.6526
```

```
px.scatter(temp, x = "dcoilwtico_interpolated", y = "sales", trendline = "ols", trendline_color_override = "red")
# 상관관계가 존재함을 확인할 수 있다!
```

3. stores

```
stores = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/stores.csv')
stores.head()
```

	store_nbr	city	state	type	cluster
0	1	Quito	Pichincha	D	13
1	2	Quito	Pichincha	D	13
2	3	Quito	Pichincha	D	8
3	4	Quito	Pichincha	D	9
4	5	Santo Domingo	Santo Domingo de los Tsachilas	D	4

```
# 결측치 확인하기
stores.isnull().sum()
```

```
store_nbr    0
city         0
state        0
type         0
cluster      0
dtype: int64
```

4. transactions

```
transactions = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/transactions.csv')
transactions.head()
```

	date	store_nbr	transactions
0	2013-01-01	25	770
1	2013-01-02	1	2111
2	2013-01-02	2	2358
3	2013-01-02	3	3487
4	2013-01-02	4	1922

```
# 맨 나중의 정보 확인하기
transactions.tail()
```

	date	store_nbr	transactions
83483	2017-08-15	50	2804
83484	2017-08-15	51	1573
83485	2017-08-15	52	2255
83486	2017-08-15	53	932
83487	2017-08-15	54	802

```
# 결측치 확인하기
transactions.isnull().sum()
```

```
date            0
store_nbr       0
transactions    0
dtype: int64
```

```
# 정렬한 것으로 저장해주기
transactions = transactions.sort_values(["store_nbr", "date"])
```

타깃 정보인 'sales'와 어떻게 연관이 있는지 확인해보자.

- transactions: 그날 이루어진 거래의 양 혹은 방문한 고객의 수를 의미한다.(Transactions means how many people came to the store or how many invoices created in a day)

> 즉, 'sales'와 밀접한 연관이 있을 관련성! (sales는 그날에 판매된 물품들의 총 금액이다.)

```
train.info()
```

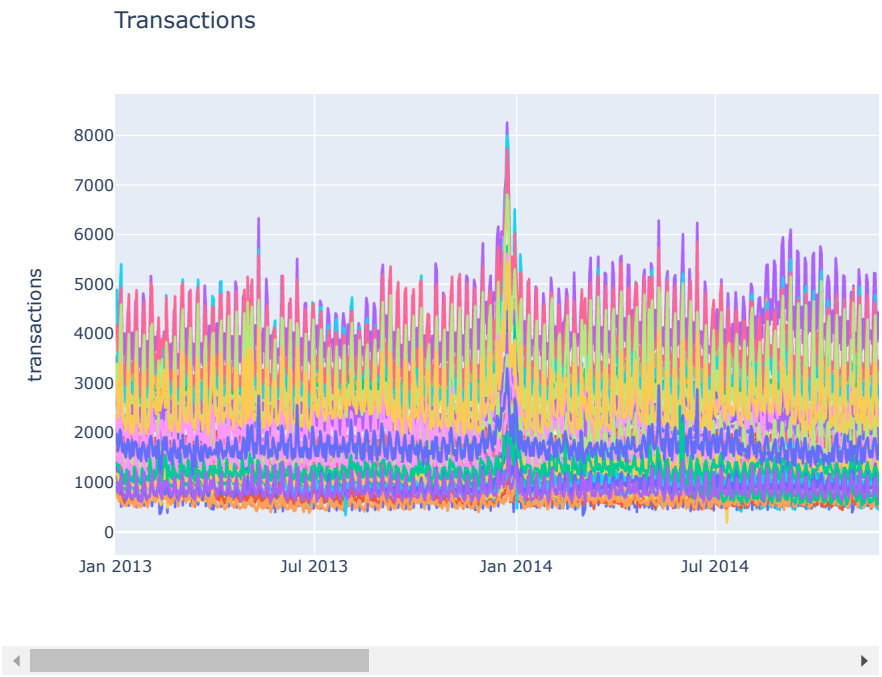
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000888 entries, 0 to 3000887
Data columns (total 6 columns):
 #   Column       Dtype
---  ------       -----
 0   id           int64
 1   date         datetime64[ns]
 2   store_nbr    int64
 3   family       object
 4   sales        float32
 5   onpromotion  float16
dtypes: datetime64[ns](1), float16(1), float32(1), int64(2), object(1)
memory usage: 108.8+ MB
```

```
transactions["date"] = pd.to_datetime(transactions.date)
transactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 83488 entries, 1 to 83487
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   date          83488 non-null  datetime64[ns]
 1   store_nbr     83488 non-null  int64         
 2   transactions  83488 non-null  int64         
dtypes: datetime64[ns](1), int64(2)
memory usage: 2.5 MB
```

```
# train의 sales와의 상관관계 확인하기
# date, stor_nbr 대로 groupby후 merge 해주기
temp = pd.merge(train.groupby(["date", "store_nbr"]).sales.sum().reset_index(), transactions, how = "left")
print("Spearman Correlation between Total Sales and Transactions: {:,.4f}".format(temp.corr("spearman").sales.loc["transactions"]))
px.line(transactions.sort_values(["store_nbr", "date"]), x='date', y='transactions', color='store_nbr',title = "Transactions" )
```

```
Spearman Correlation between Total Sales and Transactions: 0.8175
```

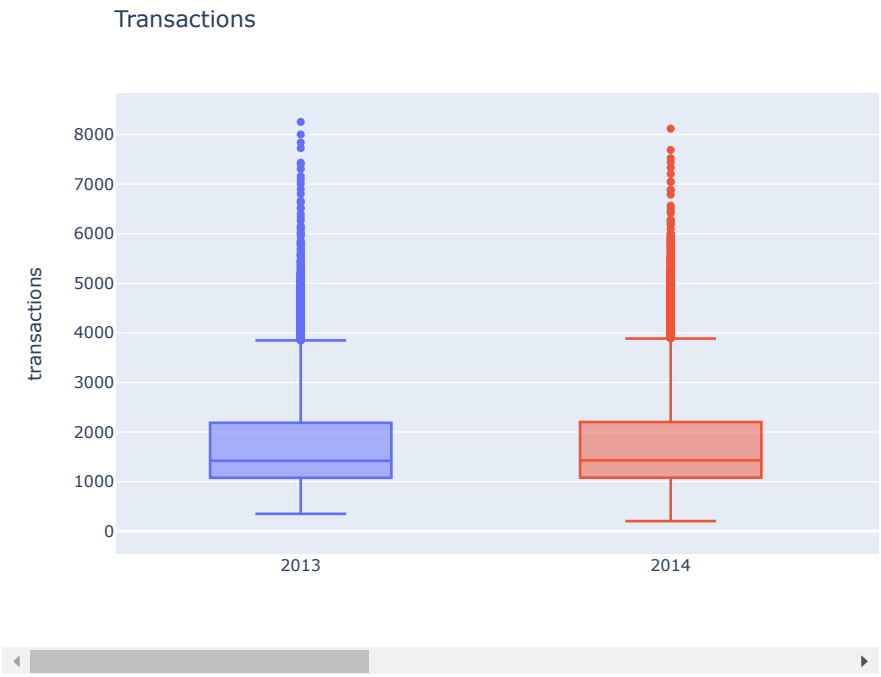


날짜에 따라 어떠한 패턴이 나타나는 것을 확인할 수 있다.

년별로 특징이 다른지 확인하기

대개 비슷한 형태가 나타난다.

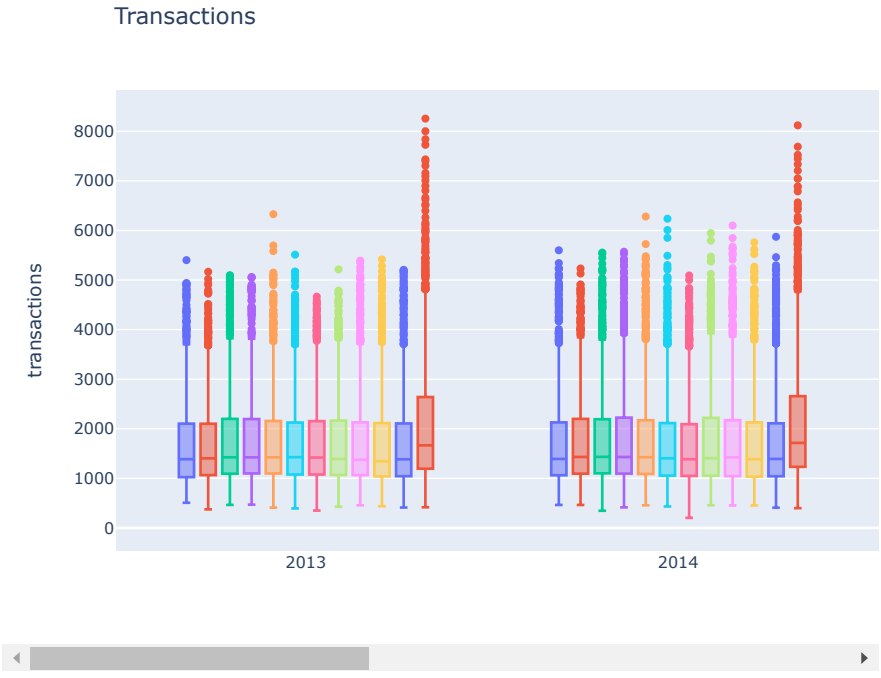
```
a = transactions.copy()
a["year"] = a.date.dt.year
px.box(a, x="year", y="transactions" , color = "year", title = "Transactions")
```



월별로 특징이 다른지 확인해보기

> 아래의 그래프를 보아, 월별마다 비슷한 특징이 나타남을 확인할 수 있다!

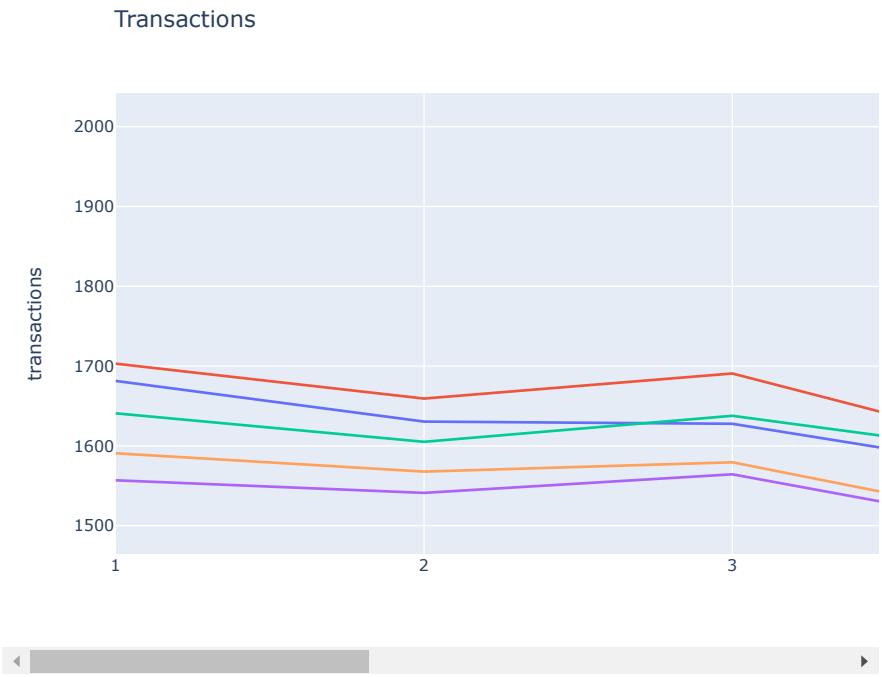
```
a = transactions.copy()
a["year"] = a.date.dt.year
a["month"] = a.date.dt.month
px.box(a, x="year", y="transactions" , color = "month", title = "Transactions")
```



요별로 특징이 다른지 확인해보기

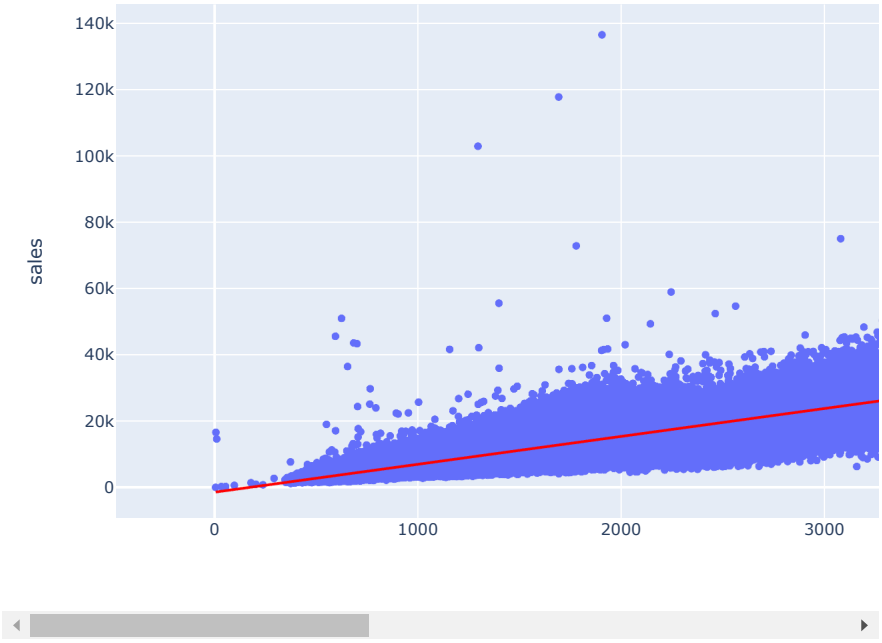
> 아래의 그래프를 보아, 요별마다 비슷한 특징이 나타남을 확인할 수 있다!

```
a = transactions.copy()
a["year"] = a.date.dt.year
a["dayofweek"] = a.date.dt.dayofweek+1
a = a.groupby(["year", "dayofweek"]).transactions.mean().reset_index()
px.line(a, x="dayofweek", y="transactions" , color = "year", title = "Transactions")
```



sales-transactions 간 상관관계 나타내기

```
px.scatter(temp, x = "transactions", y = "sales", trendline = "ols", trendline_color_override = "red")
# 상관관계가 상당히 높음을 알 수 있다.
```



5. train, test

```
train = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/train.csv')
train.head()
```

	id	date	store_nbr	family	sales	onpromotion
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0
1	1	2013-01-01	1	BABY CARE	0.00	0
2	2	2013-01-01	1	BEAUTY	0.00	0
3	3	2013-01-01	1	BEVERAGES	0.00	0
4	4	2013-01-01	1	BOOKS	0.00	0

```
# 결측치 확인하기
train.isnull().sum()
```

```
id            0
date          0
store_nbr     0
family        0
sales         0
onpromotion   0
dtype: int64
```

```
test = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ESAA 학회_OB/OB_project_vaca/data/test.csv')
test.head()
```

	id	date	store_nbr	family	onpromotion
0	3000888	2017-08-16	1	AUTOMOTIVE	0
1	3000889	2017-08-16	1	BABY CARE	0
2	3000890	2017-08-16	1	BEAUTY	2
3	3000891	2017-08-16	1	BEVERAGES	20
4	3000892	2017-08-16	1	BOOKS	0

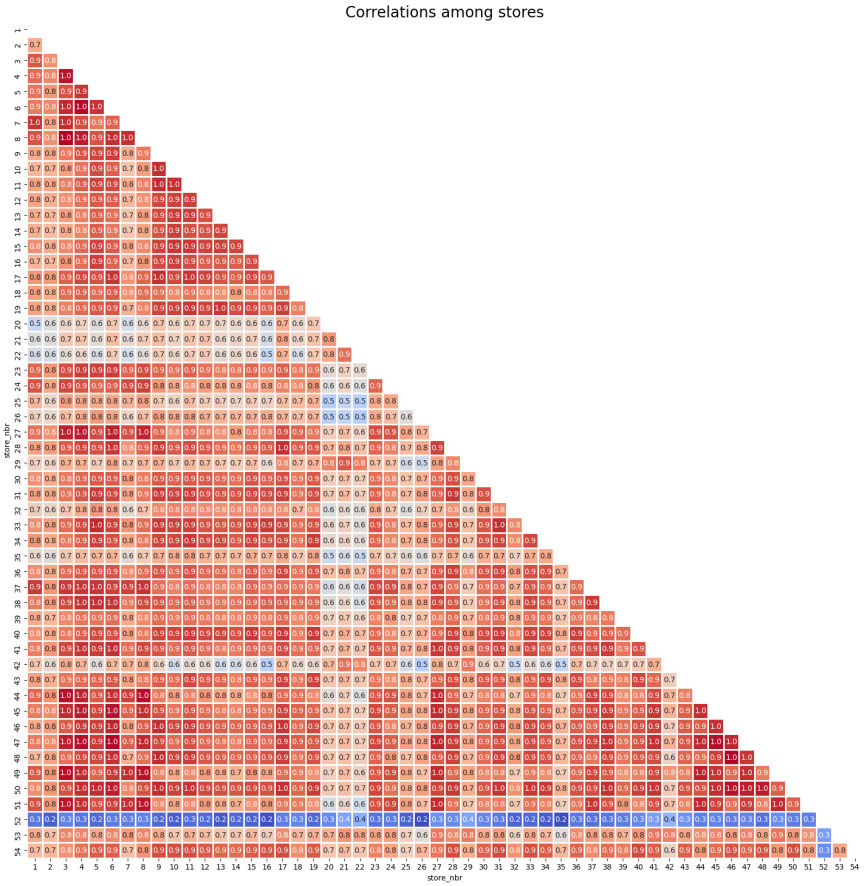
```
# 결측치 확인하기
test.isnull().sum()
```

```
id            0
date          0
store_nbr     0
family        0
onpromotion   0
dtype: int64
```

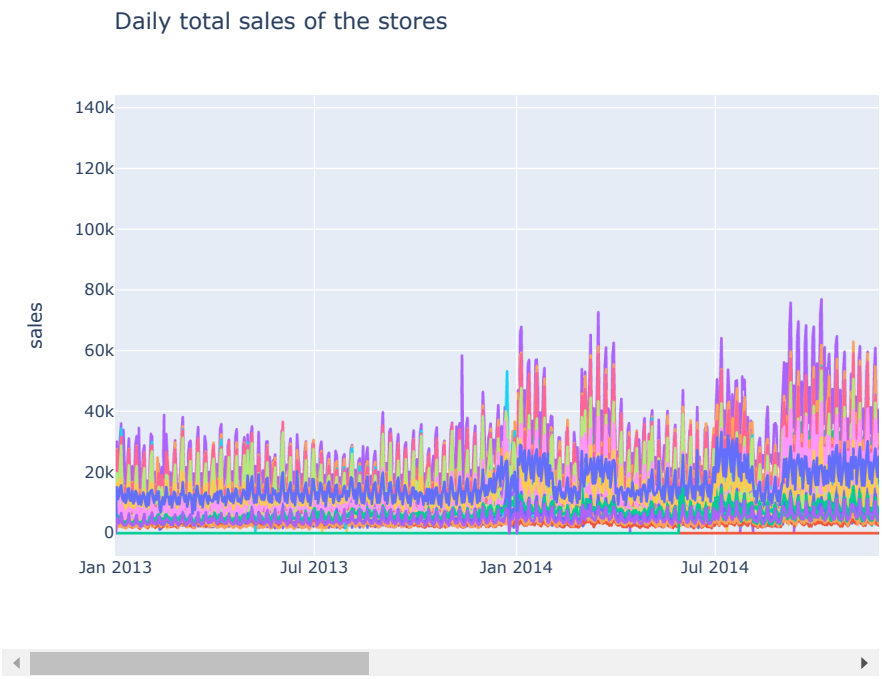
```
train["date"] = pd.to_datetime(train.date)
test["date"] = pd.to_datetime(test.date)
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000888 entries, 0 to 3000887
Data columns (total 6 columns):
 #   Column       Dtype
---  ------       -----
 0   id           int64
 1   date         datetime64[ns]
 2   store_nbr    int64
 3   family       object
 4   sales        float64
 5   onpromotion  int64
dtypes: datetime64[ns](1), float64(1), int64(3), object(1)
memory usage: 137.4+ MB
```

```
# 상관관계 확인하기!
# 가게번호 합쳐준 뒤에 계산!
a = train[["store_nbr", "sales"]]
a["ind"] = 1
a["ind"] = a.groupby("store_nbr").ind.cumsum().values
a = pd.pivot(a, index = "ind", columns = "store_nbr", values = "sales").corr()
mask = np.triu(a.corr())
plt.figure(figsize=(20, 20))
sns.heatmap(a,
        annot=True,
        fmt='.1f',
        cmap='coolwarm',
        square=True,
        mask=mask,
        linewidths=1,
        cbar=False)
plt.title("Correlations among stores",fontsize = 20)
plt.show()
```

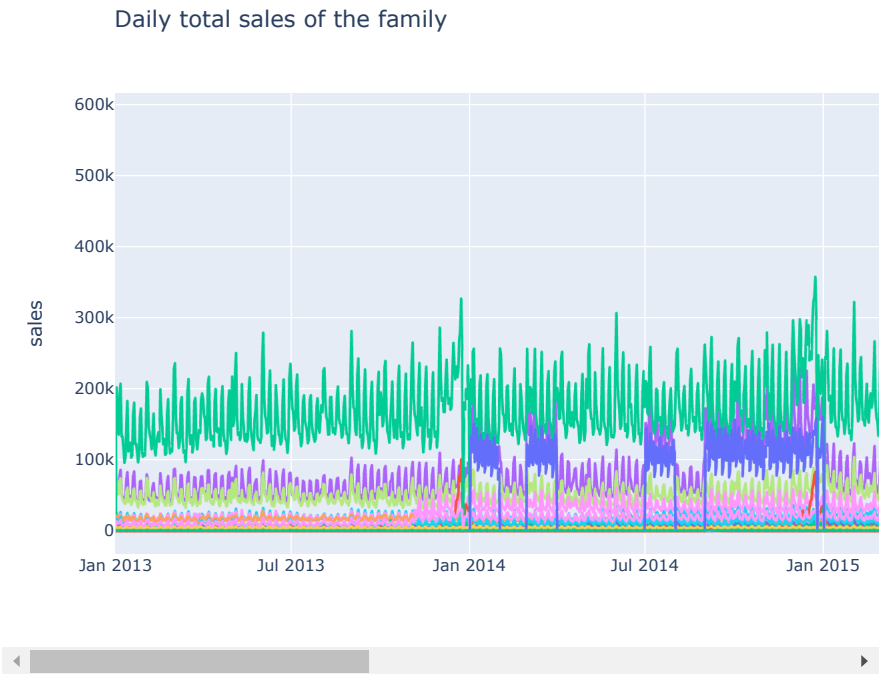

Correlations among stores
store_nbr
store_nbr

```
# 가게별 매일 sale양 확인하기
a = train.set_index("date").groupby("store_nbr").resample("D").sales.sum().reset_index()
px.line(a, x = "date", y= "sales", color = "store_nbr", title = "Daily total sales of the stores")
```

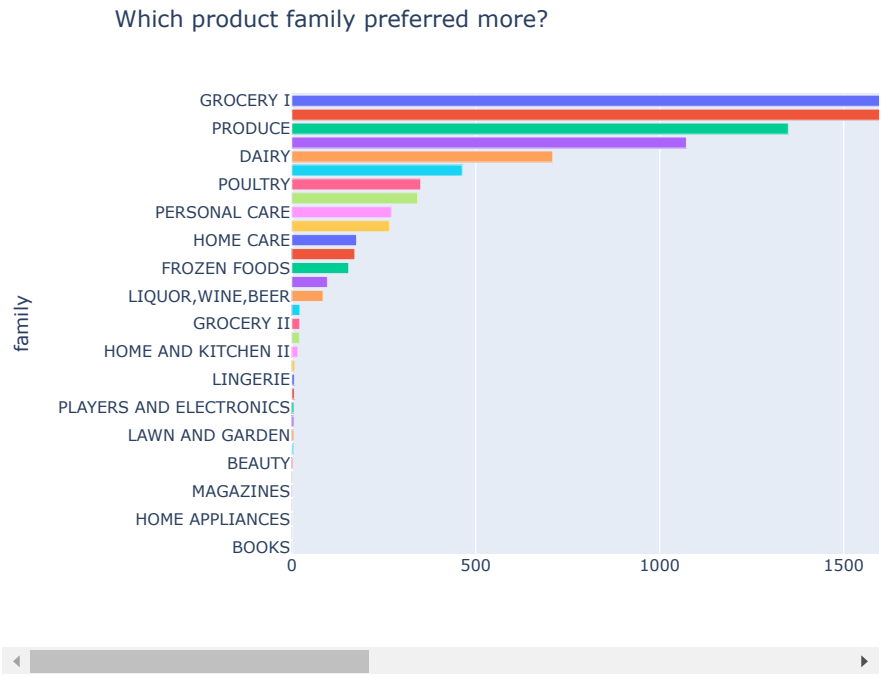


물품 종류별 매일 매출량 확인하기

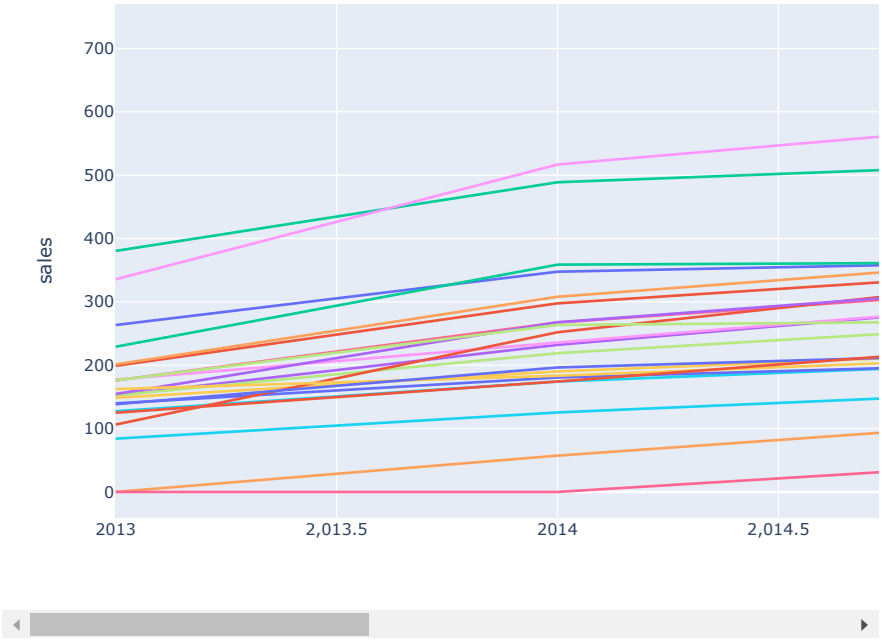
```
a = train.set_index("date").groupby("family").resample("D").sales.sum().reset_index()
px.line(a, x = "date", y= "sales", color = "family", title = "Daily total sales of the family")
```




```
# 바 그래프로 물품별 총 매출량 확인하기
a = train.groupby("family").sales.mean().sort_values(ascending = False).reset_index()
px.bar(a, y = "family", x="sales", color = "family", title = "Which product family preferred more?")
```




```
# 도시별 매일 매출량 확인하기
d = pd.merge(train, stores)
d["store_nbr"] = d["store_nbr"].astype("int8")
d["year"] = d.date.dt.year
px.line(d.groupby(["city", "year"]).sales.mean().reset_index(), x = "year", y = "sales", color = "city")
```



6. merge하기

1. transaction과 merge하기

> 안하기로 회의에서 이야기했었음...

2. oils와 merge

```
merged = pd.merge(train, oil, on='date')
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
merged["oil_price"] =merged.dcoilwtico_interpolated.interpolate()
```

3. holidays와 merge

```
events['events'] = 1
events.head()
```

	date	events
55	2013-05-12	1
103	2014-05-11	1
106	2014-06-12	1
107	2014-06-15	1
108	2014-06-20	1

```
events.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 56 entries, 55 to 311
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    56 non-null     datetime64[ns]
 1   events  56 non-null     int64
dtypes: datetime64[ns](1), int64(1)
memory usage: 1.3 KB
```

```
events.date.unique()
```

```
array(['2013-05-12T00:00:00.000000000', '2014-05-11T00:00:00.000000000',
       '2014-06-12T00:00:00.000000000', '2014-06-15T00:00:00.000000000',
       '2014-06-20T00:00:00.000000000', '2014-06-25T00:00:00.000000000',
       '2014-06-28T00:00:00.000000000', '2014-06-29T00:00:00.000000000',
       '2014-06-30T00:00:00.000000000', '2014-07-01T00:00:00.000000000',
       '2014-07-04T00:00:00.000000000', '2014-07-05T00:00:00.000000000',
       '2014-07-08T00:00:00.000000000', '2014-07-09T00:00:00.000000000',
       '2014-07-12T00:00:00.000000000', '2014-07-13T00:00:00.000000000',
       '2014-11-28T00:00:00.000000000', '2014-12-01T00:00:00.000000000',
       '2015-05-10T00:00:00.000000000', '2015-11-27T00:00:00.000000000',
       '2015-11-30T00:00:00.000000000', '2016-04-16T00:00:00.000000000',
       '2016-04-17T00:00:00.000000000', '2016-04-18T00:00:00.000000000',
       '2016-04-19T00:00:00.000000000', '2016-04-20T00:00:00.000000000',
       '2016-04-21T00:00:00.000000000', '2016-04-22T00:00:00.000000000',
       '2016-04-23T00:00:00.000000000', '2016-04-24T00:00:00.000000000',
       '2016-04-25T00:00:00.000000000', '2016-04-26T00:00:00.000000000',
       '2016-04-27T00:00:00.000000000', '2016-04-28T00:00:00.000000000',
       '2016-04-29T00:00:00.000000000', '2016-04-30T00:00:00.000000000',
       '2016-05-01T00:00:00.000000000', '2016-05-02T00:00:00.000000000',
       '2016-05-03T00:00:00.000000000', '2016-05-04T00:00:00.000000000',
       '2016-05-05T00:00:00.000000000', '2016-05-06T00:00:00.000000000',
       '2016-05-07T00:00:00.000000000', '2016-05-08T00:00:00.000000000',
       '2016-05-09T00:00:00.000000000', '2016-05-10T00:00:00.000000000',
       '2016-05-11T00:00:00.000000000', '2016-05-12T00:00:00.000000000',
       '2016-05-13T00:00:00.000000000', '2016-05-14T00:00:00.000000000',
       '2016-05-15T00:00:00.000000000', '2016-05-16T00:00:00.000000000',
       '2016-11-25T00:00:00.000000000', '2016-11-28T00:00:00.000000000',
       '2017-05-14T00:00:00.000000000'], dtype='datetime64[ns]')
```

```
merged = pd.merge(merged, events, on='date', how='left')
```

```
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# events 아닌 것 0으로 채워주기!
merged['events'].fillna(value = 0, inplace=True)
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
merged[merged['events'] == 1].date.unique()
```

```
array(['2014-06-12T00:00:00.000000000', '2014-06-20T00:00:00.000000000',
       '2014-06-25T00:00:00.000000000', '2014-06-30T00:00:00.000000000',
       '2014-07-01T00:00:00.000000000', '2014-07-04T00:00:00.000000000',
       '2014-07-08T00:00:00.000000000', '2014-07-09T00:00:00.000000000',
       '2014-11-28T00:00:00.000000000', '2014-12-01T00:00:00.000000000',
       '2015-11-27T00:00:00.000000000', '2015-11-30T00:00:00.000000000',
       '2016-04-18T00:00:00.000000000', '2016-04-19T00:00:00.000000000',
       '2016-04-20T00:00:00.000000000', '2016-04-21T00:00:00.000000000',
       '2016-04-22T00:00:00.000000000', '2016-04-25T00:00:00.000000000',
       '2016-04-26T00:00:00.000000000', '2016-04-27T00:00:00.000000000',
       '2016-04-28T00:00:00.000000000', '2016-04-29T00:00:00.000000000',
       '2016-05-02T00:00:00.000000000', '2016-05-03T00:00:00.000000000',
       '2016-05-04T00:00:00.000000000', '2016-05-05T00:00:00.000000000',
       '2016-05-06T00:00:00.000000000', '2016-05-09T00:00:00.000000000',
       '2016-05-10T00:00:00.000000000', '2016-05-11T00:00:00.000000000',
       '2016-05-12T00:00:00.000000000', '2016-05-13T00:00:00.000000000',
       '2016-05-16T00:00:00.000000000', '2016-11-25T00:00:00.000000000',
       '2016-11-28T00:00:00.000000000'], dtype='datetime64[ns]')
```

```
# 국가적으로 쉬는 날 merge하기
national['national'] = 1
national.head()
```

	date	national
14	2012-08-10	1
20	2012-10-12	1
21	2012-11-02	1
22	2012-11-03	1
31	2012-12-21	1

```
merged = pd.merge(merged, national, on='date', how='left')
```

```
# national 아닌 것 0으로 채워주기!
merged['national'].fillna(value = 0, inplace=True)
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

3. store과 merge하기

```
merged = pd.merge(merged, stores, on='store_nbr', how='left')
merged
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_in
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	
1	1	2013-01-01	1	BABY CARE	0.00	0	
2	2	2013-01-01	1	BEAUTY	0.00	0	
3	3	2013-01-01	1	BEVERAGES	0.00	0	
4	4	2013-01-01	1	BOOKS	0.00	0	
...	...	...	...	...	...	...	...
2145523	3000883	2017-08-15	9	POULTRY	438.13	0	
2145524	3000884	2017-08-15	9	PREPARED FOODS	154.55	1	

이제 regional, local과 결합 가능!

```
regional.locale_name.head()
```

```
1                        Cotopaxi
7                        Imbabura
23    Santo Domingo de los Tsachilas
24                     Santa Elena
47                       Cotopaxi
Name: locale_name, dtype: object
```

```
merged.state.unique()
```

```
array(['Pichincha', 'Cotopaxi', 'Chimborazo', 'Imbabura',
       'Santo Domingo de los Tsachilas', 'Bolivar', 'Pastaza',
       'Tungurahua', 'Guayas', 'Santa Elena', 'Los Rios', 'Azuay', 'Loja',
       'El Oro', 'Esmeraldas', 'Manabi'], dtype=object)
```

```
regional.rename(columns={'locale_name':'state'}, inplace=True)
regional['regional'] = 1
regional.head()
```

	date	state	regional
1	2012-04-01	Cotopaxi	1
7	2012-06-25	Imbabura	1
23	2012-11-06	Santo Domingo de los Tsachilas	1
24	2012-11-07	Santa Elena	1
47	2013-04-01	Cotopaxi	1

```
merged = pd.merge(merged, regional, on=['date', 'state'], how='left')
```

```
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# regional 아닌 것 0으로 채워주기!
```

```
merged['regional'].fillna(value = 0, inplace=True)
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

local과 결합해주기

```
local.locale_name.head()
```

```
0       Manta
2      Cuenca
3    Libertad
4    Riobamba
5        Puyo
Name: locale_name, dtype: object
```

```
merged.city.unique()
```

```
array(['Quito', 'Cayambe', 'Latacunga', 'Riobamba', 'Ibarra',
       'Santo Domingo', 'Guaranda', 'Puyo', 'Ambato', 'Guayaquil',
       'Salinas', 'Daule', 'Babahoyo', 'Quevedo', 'Playas', 'Libertad',
       'Cuenca', 'Loja', 'Machala', 'Esmeraldas', 'Manta', 'El Carmen'],
      dtype=object)
```

```
local.rename(columns={'locale_name':'city'}, inplace=True)
local['local'] = 1
local.head()
```

	date	city	local
0	2012-03-02	Manta	1
2	2012-04-12	Cuenca	1
3	2012-04-14	Libertad	1
4	2012-04-21	Riobamba	1
5	2012-05-12	Puyo	1

```
merged = pd.merge(merged, local, on=['date', 'city'], how='left')
```

```
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# local 아닌 것 0으로 채워주기!
merged['local'].fillna(value = 0, inplace=True)
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# 결측치 확인하기
merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2145528 entries, 0 to 2145527
Data columns (total 16 columns):
 #   Column                  Dtype
---  ------                  -----
 0   id                      int64
 1   date                    datetime64[ns]
 2   store_nbr               int64
 3   family                  object
 4   sales                   float64
 5   onpromotion             int64
 6   dcoilwtico_interpolated float64
 7   oil_price               float64
 8   events                  float64
 9   national                float64
 10  city                    object
 11  state                   object
 12  type                    object
 13  cluster                 int64
 14  regional                float64
 15  local                   float64
dtypes: datetime64[ns](1), float64(7), int64(4), object(4)
memory usage: 278.3+ MB
```

```
merged.isnull().sum()
```

```
id                      0
date                    0
store_nbr               0
family                  0
sales                   0
onpromotion             0
dcoilwtico_interpolated 0
oil_price               0
events                  0
national                0
city                    0
state                   0
type                    0
cluster                 0
regional                0
local                   0
dtype: int64
```

요일 변수 추가하기

```
merged['weekday'] = merged['date'].dt.weekday
```

```
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# work_day는 원래 토요일 등 휴일이나, 근무하는 날
work_day['work_day'] = 1
work_day.head()
```

	date	type	locale	locale_name	description	work_day
42	2013-01-05	Work Day	National	Ecuador	Recupero puente Navidad	1
43	2013-01-12	Work Day	National	Ecuador	Recupero puente primer dia del ano	1
149	2014-12-20	Work Day	National	Ecuador	Recupero Puente Navidad	1
161	2015-01-	Work			Recupero Puente Primer	

```
work_day.locale.unique()  # 모두 national이므로 정보 삭제가 가능하다!
```

```
array(['National'], dtype=object)
```

```
work_day = work_day.drop(['type', 'locale', 'locale_name', 'description'], axis=1)
```

```
work_day.head()
```

	date	work_day
42	2013-01-05	1
43	2013-01-12	1
149	2014-12-20	1
161	2015-01-10	1
283	2016-11-12	1

```
merged = pd.merge(merged, work_day, on='date', how='left')
```

```
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
# events 아닌 것 0으로 채워주기!
merged['work_day'].fillna(value = 0, inplace=True)
merged.head()
```

	id	date	store_nbr	family	sales	onpromotion	dcoilwtico_interpolated
0	0	2013-01-01	1	AUTOMOTIVE	0.00	0	93.14
1	1	2013-01-01	1	BABY CARE	0.00	0	93.14
2	2	2013-01-01	1	BEAUTY	0.00	0	93.14

```
merged.isnull().sum()
```

```
id                        0
date                      0
store_nbr                 0
family                    0
sales                     0
onpromotion               0
dcoilwtico_interpolated   0
oil_price                 0
events                    0
national                  0
city                      0
state                     0
type                      0
cluster                   0
regional                  0
local                     0
weekday                   0
work_day                  0
dtype: int64
```

```
merged[7100: 7130]
# work day에는 마트 매출이 존재하지 않는다 > 주말이라 존재하지 않는 것인가?
```

		01-04				
7110	7110	2013-01-04	9	HOME AND KITCHEN I	0.00	0
7111	7111	2013-01-04	9	HOME AND KITCHEN II	0.00	0
7112	7112	2013-01-04	9	HOME APPLIANCES	0.00	0
7113	7113	2013-01-04	9	HOME CARE	0.00	0
7114	7114	2013-01-04	9	LADIESWEAR	0.00	0
7115	7115	2013-01-04	9	LAWN AND GARDEN	1.00	0
7116	7116	2013-01-04	9	LINGERIE	7.00	0
7117	7117	2013-01-04	9	LIQUOR,WINE,BEER	54.00	0
7118	7118	2013-01-04	9	MAGAZINES	0.00	0
7119	7119	2013-01-04	9	MEATS	310.88	0
7120	7120	2013-01-04	9	PERSONAL CARE	324.00	0
7121	7121	2013-01-04	9	PET SUPPLIES	0.00	0
7122	7122	2013-01-04	9	PLAYERS AND ELECTRONICS	0.00	0
7123	7123	2013-01-04	9	POULTRY	332.67	0
7124	7124	2013-01-04	9	PREPARED FOODS	57.00	0
7125	7125	2013-01-04	9	PRODUCE	0.00	0
7126	7126	2013-01-04	9	SCHOOL AND OFFICE SUPPLIES	0.00	0
7127	7127	2013-01-04	9	SEAFOOD	11.00	0
7128	10692	2013-01-07	1	AUTOMOTIVE	0.00	0
7129	10693	2013-01-07	1	BABY CARE	0.00	0