Team ID : PNT2022TMID18585 Date : 18 November 2022

```
# data operation libraries
import numpy as np
import pandas as pd

#·importing·visualisation·libraries
import·matplotlib.pyplot·as·plt
import·seaborn·as·sns
%matplotlib inline

# for chloroplath plotting
import chart_studio.plotly as py
import plotly.graph_objs as go
import plotly
import cufflinks as cf
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)
cf.go_offline()

# for datetime operations
import datetime as dt

# pandas general settings
pd.options.display.max_columns = None
```

```
data = pd.read_csv('/content/Global_Superstore2.csv', encoding='windows-1252')
```

```
import pandas as pd
import io
```

```
pip install chart_studio
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Collecting chart_studio
  Downloading chart_studio-1.1.0-py3-none-any.whl (64 kB)
     |████████████████████████████████| 64 kB 1.7 MB/s
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from c
Collecting retrying>=1.3.3
  Downloading retrying-1.3.3.tar.gz (10 kB)
Requirement already satisfied: plotly in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-pa
Building wheels for collected packages: retrying
  Building wheel for retrying (setup.py) ... done
  Created wheel for retrying: filename=retrying-1.3.3-py3-none-any.whl size=11448 sh
```

```
  Stored in directory: /root/.cache/pip/wheels/f9/8d/8d/f6af3f7f9eea3553bc2fe6d53e4b
Successfully built retrying
Installing collected packages: retrying, chart-studio
Successfully installed chart-studio-1.1.0 retrying-1.3.3
```

```
data.head(2) #taking a look at the dataframe structure
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	Sta
0	32298	CA-2012-124891	31-07-2012	31-07-2012	Same Day	RH-19495	Rick Hansen	Consumer	New York City	N Y
1	26341	IN-2013-77878	05-02-2013	07-02-2013	Second Class	JR-16210	Justin Ritter	Corporate	Wollongong	N Sou Wa

```
# correcting 'Order Date' variable
data[['order_day','order_month','order_year']] = data['Order Date'].str.split('-', expand=
data['Order Date'] = data['order_year'] + '/' + data['order_month'] + '/' + data['order_da
data['Order Date'] = pd.to_datetime(data['Order Date'])


# doing likewise for 'Ship Date'
data[['ship_day','ship_month','ship_year']] = data['Ship Date'].str.split('-', expand=True
data['Ship Date'] = data['ship_year'] + '/' + data['ship_month'] + '/' + data['ship_day']
data['Ship Date'] = pd.to_datetime(data['Ship Date'])


# dropping the support columns
data.drop(columns=['order_day','order_month','order_year','ship_day','ship_month','ship_ye


data.info() #checkout the data types/ null rows and memory consumption
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          51290 non-null  int64
 1   Order ID        51290 non-null  object
 2   Order Date      51290 non-null  datetime64[ns]
 3   Ship Date       51290 non-null  datetime64[ns]
 4   Ship Mode       51290 non-null  object
 5   Customer ID     51290 non-null  object
```

```
 6   Customer Name   51290 non-null  object
 7   Segment         51290 non-null  object
 8   City            51290 non-null  object
 9   State           51290 non-null  object
 10  Country         51290 non-null  object
 11  Postal Code     9994 non-null   float64
 12  Market          51290 non-null  object
 13  Region          51290 non-null  object
 14  Product ID      51290 non-null  object
 15  Category        51290 non-null  object
 16  Sub-Category    51290 non-null  object
 17  Product Name    51290 non-null  object
 18  Sales           51290 non-null  float64
 19  Quantity        51290 non-null  int64
 20  Discount        51290 non-null  float64
 21  Profit          51290 non-null  float64
 22  Shipping Cost   51290 non-null  float64
 23  Order Priority  51290 non-null  object
dtypes: datetime64[ns](2), float64(5), int64(2), object(15)
memory usage: 9.4+ MB
```

```
# let's check out the columns which are suitable category column type

data.nunique()
```

```
Row ID            51290
Order ID          25035
Order Date         1430
Ship Date          1464
Ship Mode             4
Customer ID        1590
Customer Name       795
Segment               3
City               3636
State              1094
Country             147
Postal Code         631
Market                7
Region               13
Product ID        10292
Category              3
Sub-Category         17
Product Name       3788
Sales             22995
Quantity             14
Discount             27
Profit            24575
Shipping Cost     10037
Order Priority        4
dtype: int64
```

```
data['Ship Mode'] = data['Ship Mode'].astype('category')
data['Segment'] = data['Segment'].astype('category')
data['Country'] = data['Country'].astype('category')
data['Market'] = data['Market'].astype('category')
data['Region'] = data['Region'].astype('category')
data['Category'] = data['Category'].astype('category')
```

```
data['Sub-Category'] = data['Sub-Category'].astype('category')
data['Order Priority'] = data['Order Priority'].astype('category')

data.info() #check the reduction in memory consumption
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51290 entries, 0 to 51289
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          51290 non-null  int64
 1   Order ID        51290 non-null  object
 2   Order Date      51290 non-null  datetime64[ns]
 3   Ship Date       51290 non-null  datetime64[ns]
 4   Ship Mode       51290 non-null  category
 5   Customer ID     51290 non-null  object
 6   Customer Name   51290 non-null  object
 7   Segment         51290 non-null  category
 8   City            51290 non-null  object
 9   State           51290 non-null  object
 10  Country         51290 non-null  category
 11  Postal Code     9994 non-null   float64
 12  Market          51290 non-null  category
 13  Region          51290 non-null  category
 14  Product ID      51290 non-null  object
 15  Category        51290 non-null  category
 16  Sub-Category    51290 non-null  category
 17  Product Name    51290 non-null  object
 18  Sales           51290 non-null  float64
 19  Quantity        51290 non-null  int64
 20  Discount        51290 non-null  float64
 21  Profit          51290 non-null  float64
 22  Shipping Cost   51290 non-null  float64
 23  Order Priority  51290 non-null  category
dtypes: category(8), datetime64[ns](2), float64(5), int64(2), object(7)
memory usage: 6.7+ MB
```

```
# making sure neither of our category columns have leading spaces

def remove_leading_spaces(df):
    for cols in df.columns:
        if df[cols].dtypes in ['object','category']:
            df[cols] = df[cols].str.strip()
        return df


data = remove_leading_spaces(data)


data.head(2)
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	City	St
0	32298	CA-2012-124891	2012-07-31	2012-07-31	Same Day	RH-19495	Rick Hansen	Consumer	New York City	N Y
1	26341	IN-2013-	2013-	2013-	Second	JR-16210	Justin	Corporate	Wollongong	N So

```
# generating years from our 'Order_year' variable because we are going
# to need this in future analysis

data['Order_year'] = data['Order Date'].dt.year
```

```
# also total unique customer count is something we need in our future analysis

print('Number of unique customers made purchase in 2011: {}'.format(data[data['Order_year'
print('Number of unique customers made purchase in 2012: {}'.format(data[data['Order_year'
print('Number of unique customers made purchase in 2013: {}'.format(data[data['Order_year'
print('Number of unique customers made purchase in 2014: {}'.format(data[data['Order_year'
```

```
Number of unique customers made purchase in 2011: 795
Number of unique customers made purchase in 2012: 795
Number of unique customers made purchase in 2013: 795
Number of unique customers made purchase in 2014: 794
```

```
def total_purchase_in_year(row):
    Order_year = row[24]

    if Order_year in [2011,2012,2013]:
        return 795
    else:
        return 794


# generating  'unique_customers_within_year' based on associated year value
# for that particular row

data['unique_customers_within_year'] = data.apply(total_purchase_in_year, axis='columns')


# Generating 'Revenue' column
data['Revenue'] = data['Sales'] * data['Quantity']


purchase_frequency = data.groupby(['Order_year','Customer Name'])


purchase_frequency.agg({'Customer Name': 'count',
                        'unique_customers_within_year': 'min',
```

```
                    'Revenue': 'sum',
                    'Profit': 'sum'})
```

Order_year	Customer Name	Customer Name	unique_customers_within_year	Revenue	Prof
2011	**Aaron Bergman**	14	795	2693.78200	189.264
	Aaron Hawkins	15	795	47418.92150	1528.255
	Aaron Smayling	8	795	12117.70000	180.540
	Adam Bellavance	6	795	9210.63210	370.652
	Adam Hart	19	795	25909.57552	322.349
...	**...**	...	...	...	
2014	**Xylona Preis**	13	794	13240.71500	210.672
	Yana	20	794	47494.72200	2175.742

```
analysis_result = purchase_frequency.agg({'Customer Name': 'count',
                    'unique_customers_within_year': 'min',
                    'Revenue': 'sum',
                    'Profit': 'sum'})
```

```
analysis_result.rename(mapper={'Customer Name': 'Purchase_during_year'}, axis=1, inplace=T
```

```
analysis_result['Customer_purchase_frequency'] = analysis_result['Purchase_during_year']/a
```

```
analysis_result.head(5)
```

		Purchase_during_year	unique_customers_within_year	Reve
Order_year	Customer Name			

```
tmp_df = analysis_result.reset_index()
```

```
tmp_df.head()
```

	Order_year	Customer Name	Purchase_during_year	unique_customers_within_year	Re
0	2011	Aaron Bergman	14	795	2693.
1	2011	Aaron Hawkins	15	795	47418.
2	2011	Aaron Smayling	8	795	12117.
3	2011	Adam Bellavance	6	795	9210.
4	2011	Adam Hart	19	795	25909.

```
grouped_object = tmp_df.groupby(['Order_year'])
```

```
freq_df = pd.DataFrame(columns=tmp_df.columns)
```

```
for g,d in grouped_object:
    highest_freq_customers = d.nlargest(1, 'Customer_purchase_frequency')
    freq_df = pd.concat([freq_df, highest_freq_customers])
```

```
def highlight_cols(x):
    df = x.copy()
    df.loc[:, ['Customer Name','Customer_purchase_frequency']] = 'background-color: green'
    df[['Order_year','Purchase_during_year','unique_customers_within_year','Revenue','Prof
    return df
```

```
display(freq_df.style.apply(highlight_cols, axis = None))
```

	Order_year	Customer Name	Purchase_during_year	unique_customers_within_year	
210	2011	David Philippe	31	795	375(
1433	2012	Rob Dowd	42	795	418

```
rev_df = pd.DataFrame(columns=tmp_df.columns)


for g,d in grouped_object:
    highest_rev_customers = d.nlargest(1, 'Revenue')
    rev_df = pd.concat([rev_df, highest_rev_customers])


def highlight_cols(x):
    df = x.copy()
    df.loc[:, ['Customer Name','Revenue']] = 'background-color: green'
    df[['Order_year','Purchase_during_year','unique_customers_within_year','Profit','Custo
    return df


display(rev_df.style.apply(highlight_cols, axis = None))
```

	Order_year	Customer Name	Purchase_during_year	unique_customers_within_year	
687	2011	Sean Miller	15	795	15
1481	2012	Sean Christensen	21	795	11
1596	2013	Adrian Barton	15	795	13
3142	2014	Tom Ashbrook	23	794	14

```
profit_df = pd.DataFrame(columns=tmp_df.columns)


for g, d in grouped_object:
    highest_profit = d.nlargest(1, 'Profit')
    profit_df = pd.concat([profit_df,highest_profit])


def highlight_cols(x):
    df = x.copy()
    df.loc[:, ['Customer Name','Profit']] = 'background-color: green'
    df[['Order_year','Purchase_during_year','unique_customers_within_year','Revenue','Cust
    return df


display(profit_df.style.apply(highlight_cols, axis = None))
```

	Order_year	Customer Name	Purchase_during_year	unique_customers_within_year	
672	2011	Sanjit Chand	18	795	
1337	2012	Mike Gockenbach	13	795	
2321	2013	Tamara Chand	23	795	1
3007	2014	Raymond Buch	22	794	

```
segment_group = data.groupby(['Order_year','Segment'])
```

```
high_profit_df = segment_group.agg({'Profit':'sum'}).unstack()
high_profit_df.style.background_gradient(cmap='Spectral', subset=pd.IndexSlice[:, pd.Index
```

	Profit		
Segment	**Consumer**	**Corporate**	**Home Office**
Order_year			
2011	117337.494060	84746.935740	46856.381740
2012	165799.190940	90556.699920	51059.388240
2013	208427.733980	125707.939080	72799.557120
2014	257675.363080	140196.753920	106293.853460

```
country_group = data.groupby(['Country'])
```

```
customer_distribution = country_group.agg({'Customer ID':'count'})
customer_distribution.columns = ['Customer_count']
customer_distribution.reset_index(inplace=True)
customer_distribution
```

	Country	Customer_count
0	Afghanistan	55
1	Albania	16
2	Algeria	196
3	Angola	122

```
country_map = dict(type='choropleth',
           locations=customer_distribution['Country'],
           locationmode='country names',
           z=customer_distribution['Customer_count'],
            reversescale = True,
           text=customer_distribution['Country'],
           colorscale='earth',
           colorbar={'title':'Customer Count'})
```

```
layout = dict(title='Customer Distribution over Countries',
             geo=dict(showframe=False,projection={'type':'mercator'}))


choromap = go.Figure(data = [country_map],layout = layout)
iplot(choromap)
```

```
country_group = data.groupby('Country')
```

```
import squarify
```

```
year_category_group = data.groupby(['Order_year','Sub-Category'])
```

```
pip install squarify
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/
Collecting squarify
  Downloading squarify-0.4.3-py3-none-any.whl (4.3 kB)
Installing collected packages: squarify
Successfully installed squarify-0.4.3
```

```
year_category_proft_df = year_category_group.agg({'Profit':'sum'})
year_category_proft_df
```

		Profit
Order_year	**Sub-Category**	
2011	**Accessories**	15719.8606
	Appliances	22838.4413
	Art	10399.0233
	Binders	11447.2053
	Bookcases	27518.8575
...	**...**	...
2014	**Paper**	20975.8306
	Phones	70657.6413
	Storage	39016.9521
	Supplies	7365.4090
	Tables	-30545.9084

68 rows × 1 columns

```
year_category_proft_df.reset_index(inplace=True)
category_yearly_profit = year_category_proft_df.groupby('Order_year')
top5_profit_category = pd.DataFrame(columns=year_category_proft_df.columns)
```

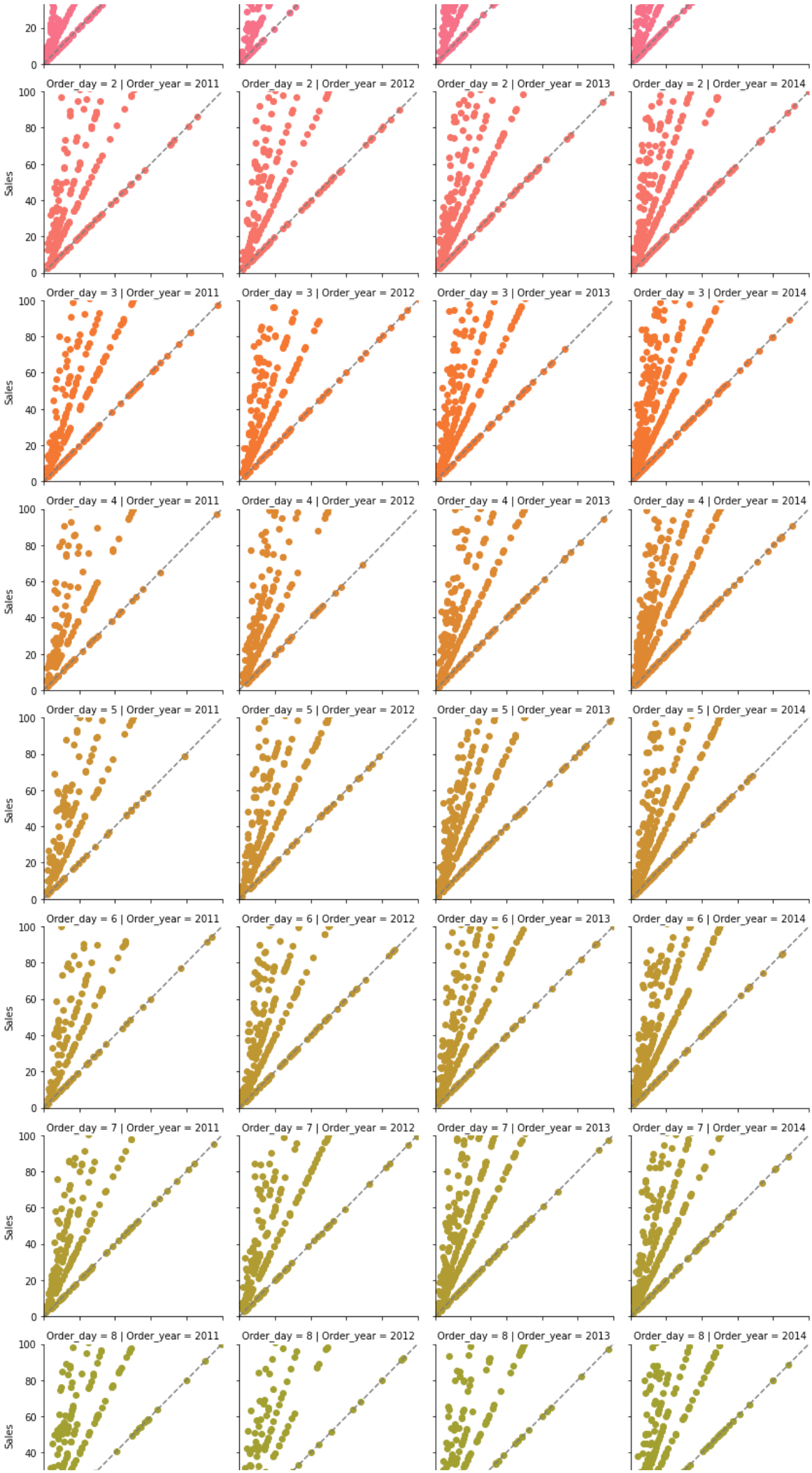
```
for g, d in category_yearly_profit:
    high_profit_categories = d.nlargest(5, 'Profit')
    top5_profit_category = pd.concat([top5_profit_category,high_profit_categories])
```

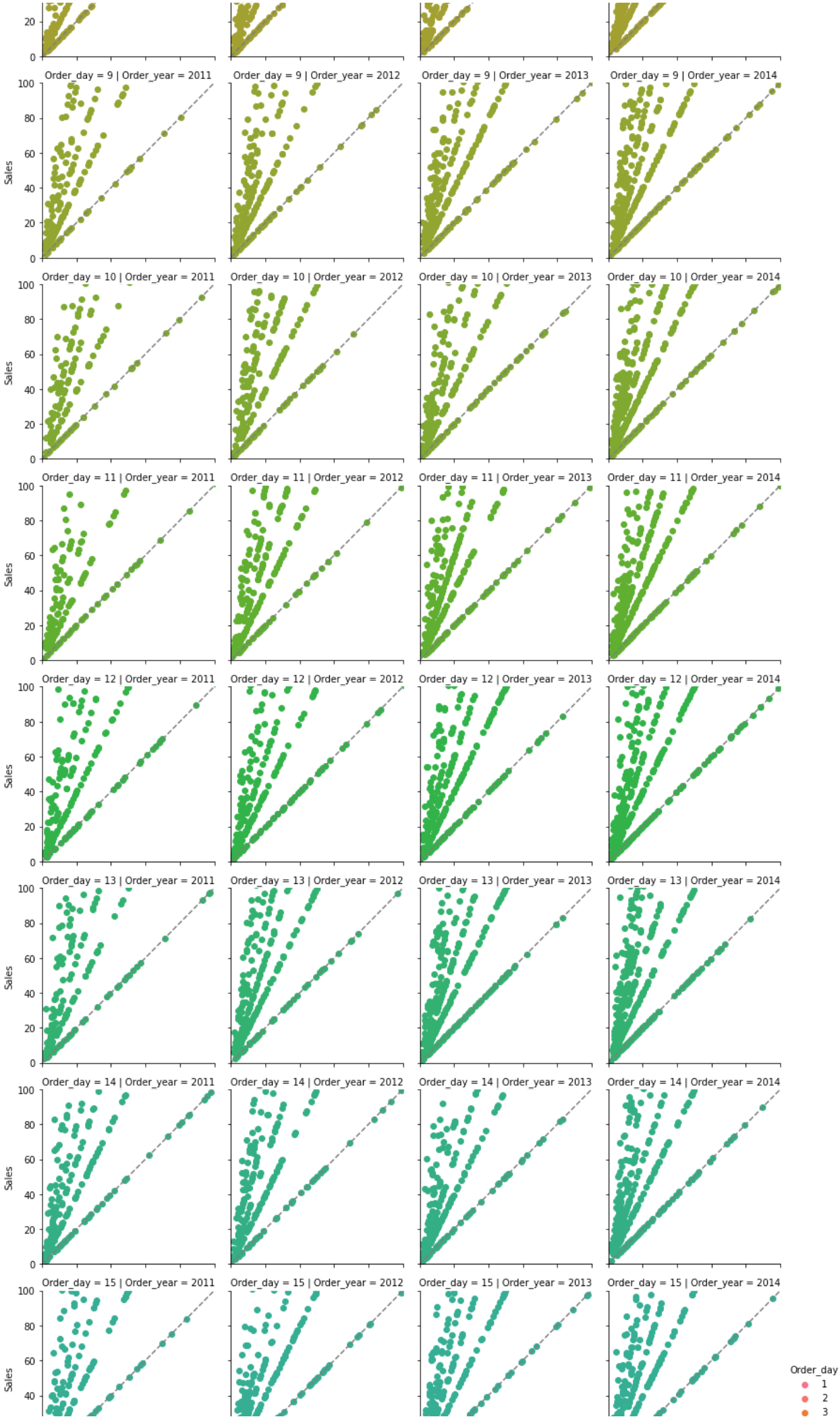
```
top5_profit_category.style.background_gradient(cmap='Spectral', subset=pd.IndexSlice[:, 'P
```

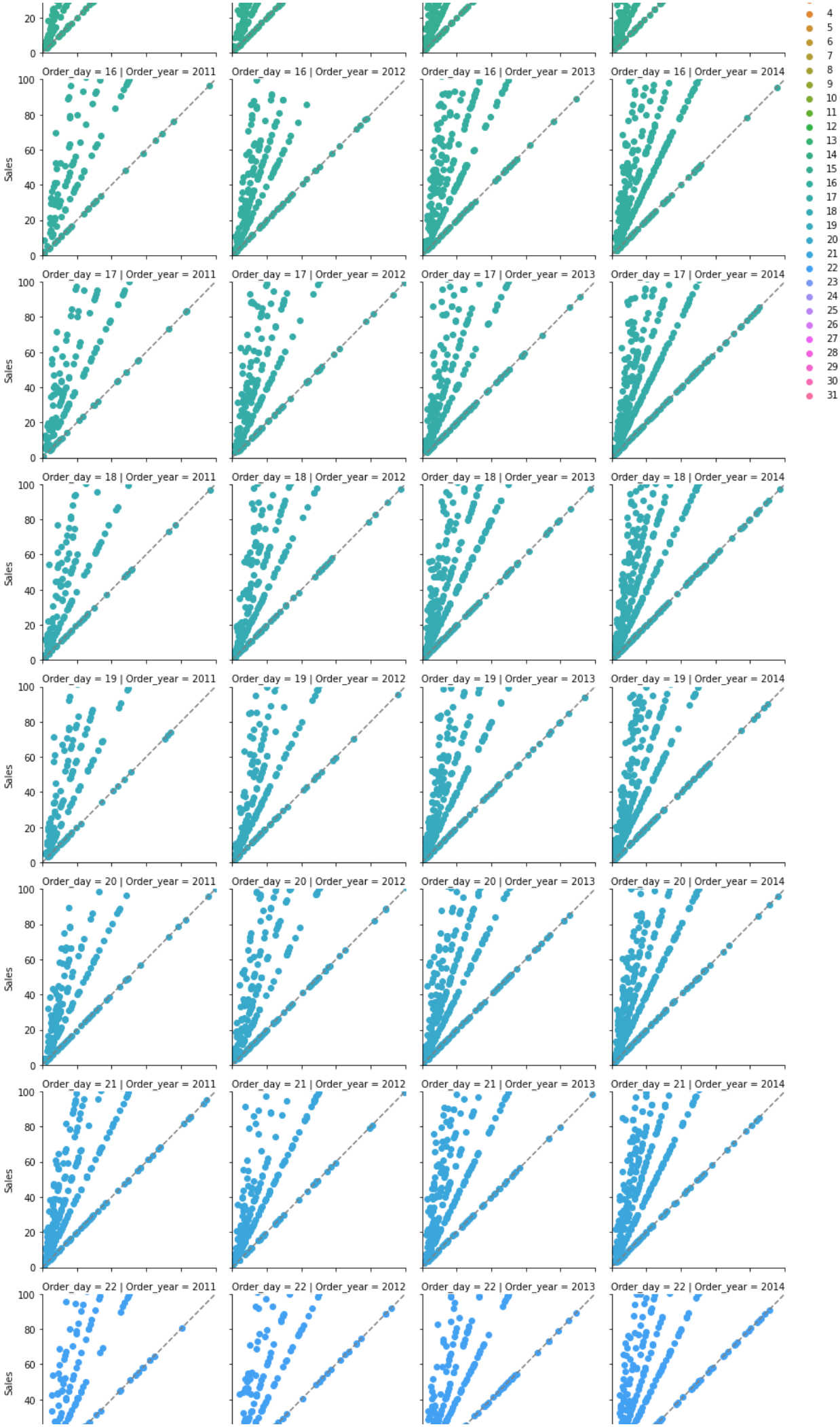
	Order_year	Sub-Category	Profit
13	2011	Phones	53927.489500
6	2011	Copiers	30375.093440
5	2011	Chairs	29943.157100
4	2011	Bookcases	27518.857500
1	2011	Appliances	22838.441300
23	2012	Copiers	51843.227600
30	2012	Phones	45223.049800
17	2012	Accessories	33507.100200
22	2012	Chairs	28755.346700
21	2012	Bookcases	28137.267100
40	2013	Copiers	72300.691180
47	2013	Phones	46908.825200
38	2013	Bookcases	43049.244400
35	2013	Appliances	41485.516000
39	2013	Chairs	40449.492100
57	2014	Copiers	104048.535960
64	2014	Phones	70657.641300
55	2014	Bookcases	63219.050500
52	2014	Appliances	53040.500500
51	2014	Accessories	41593.928600

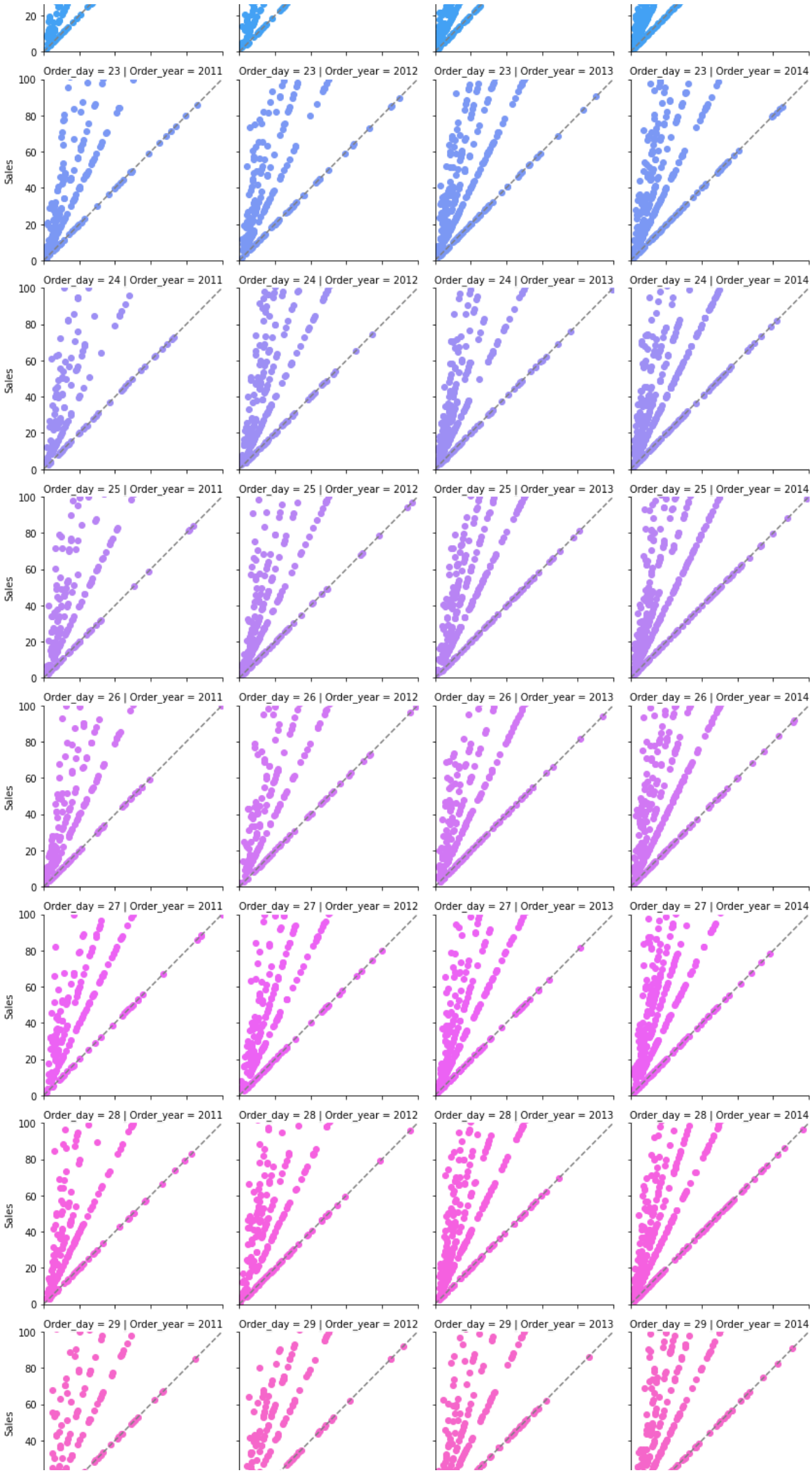
```
data['Unit_price'] = data['Sales']/data['Quantity']
data['Order_day'] = data['Order Date'].dt.day
```

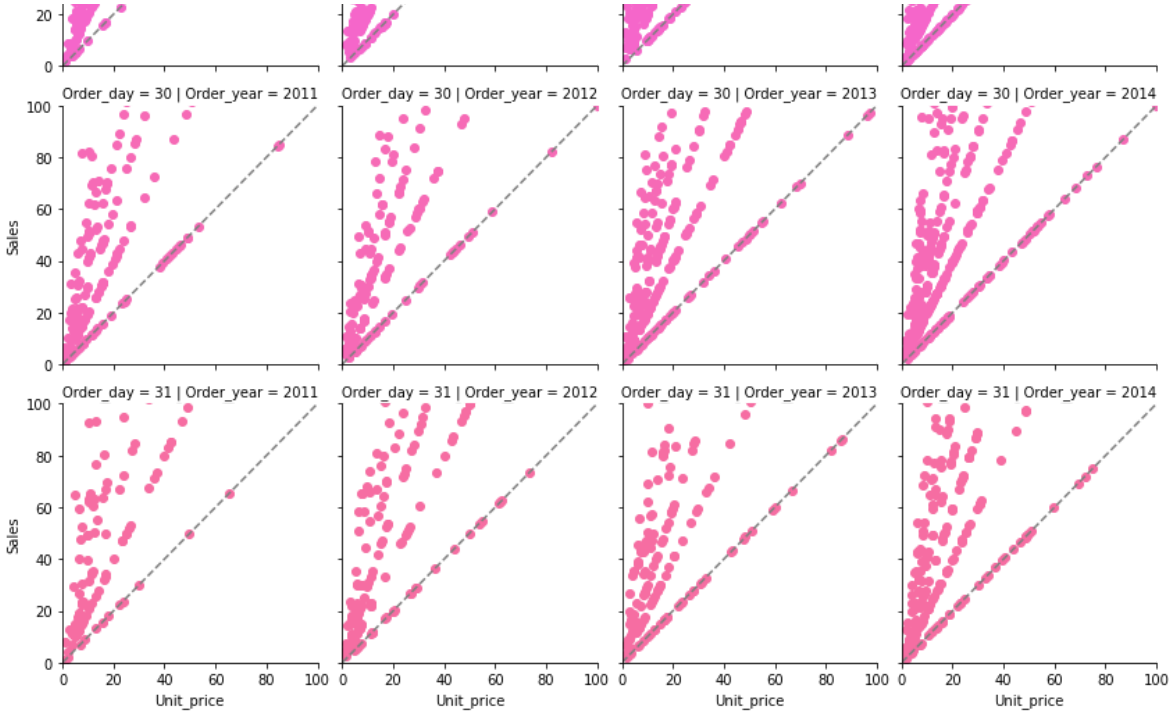
```
g5 = sns.FacetGrid(data, row = 'Order_day', col = 'Order_year', hue = 'Order_day')
kwe = dict(s = 50, linewidth = 0.5, edgecolor = 'black')
g5 = g5.map(plt.scatter, 'Unit_price', 'Sales')
g5.set(xlim=(0,100), ylim=(0,100))
for ax in g5.axes.flat:
    ax.plot((0,100),(0,100), c = 'gray', ls = '--')
g5.add_legend()
```


Order_day = 2 | Order_year = 2011
Order_day = 2 | Order_year = 2012
Order_day = 2 | Order_year = 2013
Order_day = 2 | Order_year = 2014
Order_day = 3 | Order_year = 2011
Order_day = 3 | Order_year = 2012
Order_day = 3 | Order_year = 2013
Order_day = 3 | Order_year = 2014
Order_day = 4 | Order_year = 2011
Order_day = 4 | Order_year = 2012
Order_day = 4 | Order_year = 2013
Order_day = 4 | Order_year = 2014
Order_day = 5 | Order_year = 2011
Order_day = 5 | Order_year = 2012
Order_day = 5 | Order_year = 2013
Order_day = 5 | Order_year = 2014
Order_day = 6 | Order_year = 2011
Order_day = 6 | Order_year = 2012
Order_day = 6 | Order_year = 2013
Order_day = 6 | Order_year = 2014
Order_day = 7 | Order_year = 2011
Order_day = 7 | Order_year = 2012
Order_day = 7 | Order_year = 2013
Order_day = 7 | Order_year = 2014
Order_day = 8 | Order_year = 2011
Order_day = 8 | Order_year = 2012
Order_day = 8 | Order_year = 2013
Order_day = 8 | Order_year = 2014
Sales


Order_day = 9 | Order_year = 2011
Order_day = 9 | Order_year = 2012
Order_day = 9 | Order_year = 2013
Order_day = 9 | Order_year = 2014
Order_day = 10 | Order_year = 2011
Order_day = 10 | Order_year = 2012
Order_day = 10 | Order_year = 2013
Order_day = 10 | Order_year = 2014
Order_day = 11 | Order_year = 2011
Order_day = 11 | Order_year = 2012
Order_day = 11 | Order_year = 2013
Order_day = 11 | Order_year = 2014
Order_day = 12 | Order_year = 2011
Order_day = 12 | Order_year = 2012
Order_day = 12 | Order_year = 2013
Order_day = 12 | Order_year = 2014
Order_day = 13 | Order_year = 2011
Order_day = 13 | Order_year = 2012
Order_day = 13 | Order_year = 2013
Order_day = 13 | Order_year = 2014
Order_day = 14 | Order_year = 2011
Order_day = 14 | Order_year = 2012
Order_day = 14 | Order_year = 2013
Order_day = 14 | Order_year = 2014
Order_day = 15 | Order_year = 2011
Order_day = 15 | Order_year = 2012
Order_day = 15 | Order_year = 2013
Order_day = 15 | Order_year = 2014
Sales
Order_day
1
2
3


Order_day = 16 | Order_year = 2011
Order_day = 16 | Order_year = 2012
Order_day = 16 | Order_year = 2013
Order_day = 16 | Order_year = 2014
Order_day = 17 | Order_year = 2011
Order_day = 17 | Order_year = 2012
Order_day = 17 | Order_year = 2013
Order_day = 17 | Order_year = 2014
Order_day = 18 | Order_year = 2011
Order_day = 18 | Order_year = 2012
Order_day = 18 | Order_year = 2013
Order_day = 18 | Order_year = 2014
Order_day = 19 | Order_year = 2011
Order_day = 19 | Order_year = 2012
Order_day = 19 | Order_year = 2013
Order_day = 19 | Order_year = 2014
Order_day = 20 | Order_year = 2011
Order_day = 20 | Order_year = 2012
Order_day = 20 | Order_year = 2013
Order_day = 20 | Order_year = 2014
Order_day = 21 | Order_year = 2011
Order_day = 21 | Order_year = 2012
Order_day = 21 | Order_year = 2013
Order_day = 21 | Order_year = 2014
Order_day = 22 | Order_year = 2011
Order_day = 22 | Order_year = 2012
Order_day = 22 | Order_year = 2013
Order_day = 22 | Order_year = 2014
Sales
0
20
40
60
80
100
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31


Order_day = 23 | Order_year = 2011
Order_day = 23 | Order_year = 2012
Order_day = 23 | Order_year = 2013
Order_day = 23 | Order_year = 2014
Order_day = 24 | Order_year = 2011
Order_day = 24 | Order_year = 2012
Order_day = 24 | Order_year = 2013
Order_day = 24 | Order_year = 2014
Order_day = 25 | Order_year = 2011
Order_day = 25 | Order_year = 2012
Order_day = 25 | Order_year = 2013
Order_day = 25 | Order_year = 2014
Order_day = 26 | Order_year = 2011
Order_day = 26 | Order_year = 2012
Order_day = 26 | Order_year = 2013
Order_day = 26 | Order_year = 2014
Order_day = 27 | Order_year = 2011
Order_day = 27 | Order_year = 2012
Order_day = 27 | Order_year = 2013
Order_day = 27 | Order_year = 2014
Order_day = 28 | Order_year = 2011
Order_day = 28 | Order_year = 2012
Order_day = 28 | Order_year = 2013
Order_day = 28 | Order_year = 2014
Order_day = 29 | Order_year = 2011
Order_day = 29 | Order_year = 2012
Order_day = 29 | Order_year = 2013
Order_day = 29 | Order_year = 2014
Sales
0
20
40
60
80
100


Order_day = 30 | Order_year = 2011
Order_day = 30 | Order_year = 2012
Order_day = 30 | Order_year = 2013
Order_day = 30 | Order_year = 2014
Order_day = 31 | Order_year = 2011
Order_day = 31 | Order_year = 2012
Order_day = 31 | Order_year = 2013
Order_day = 31 | Order_year = 2014
Sales
Unit_price
0
20
40
60
80
100