Analyzing Sales Data

Date: 30 December 2022

Author: Natthaphong Siriwattanapaitoon (Team)

```
# import data
import pandas as pd
df = pd.read_csv("sample-store.csv")
```

```
# preview top 5 rows
df.head(5)
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City
0	1	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale

5 rows × 21 columns

```
# shape of dataframe
df.shape
```

```
(9994, 21)
```

```
# see data frame information using .info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Row ID          9994 non-null   int64
 1   Order ID        9994 non-null   object
 2   Order Date      9994 non-null   object
 3   Ship Date       9994 non-null   object
 4   Ship Mode       9994 non-null   object
 5   Customer ID     9994 non-null   object
 6   Customer Name   9994 non-null   object
 7   Segment         9994 non-null   object
 8   Country/Region  9994 non-null   object
 9   City            9994 non-null   object
 10  State           9994 non-null   object
 11  Postal Code     9983 non-null   float64
 12  Region          9994 non-null   object
 13  Product ID      9994 non-null   object
 14  Category        9994 non-null   object
```

We can use `pd.to_datetime()` function to convert columns 'Order Date' and 'Ship Date' to datetime.

```
# example of pd.to_datetime() function
pd.to_datetime(df['Order Date'].head(), format='%m/%d/%Y')
```

```
0   2019-11-08
1   2019-11-08
2   2019-06-12
3   2018-10-11
4   2018-10-11
Name: Order Date, dtype: datetime64[ns]
```

```
# TODO - convert order date and ship date to datetime in the original dataframe
df['Order Date'] = pd.to_datetime(df['Order Date'], format='%m/%d/%Y')
df['Ship Date'] = pd.to_datetime(df['Ship Date'], format='%m/%d/%Y')
```

```
# TODO - count nan in postal code column
df.isna().sum()
```

```
Row ID            0
Order ID          0
Order Date        0
Ship Date         0
Ship Mode         0
Customer ID       0
Customer Name     0
Segment           0
Country/Region    0
City              0
State             0
Postal Code      11
Region            0
Product ID        0
Category          0
Sub-Category      0
Product Name      0
Sales             0
Quantity          0
Discount          0
Profit            0
dtype: int64
```

```
# TODO - filter rows with missing values
df[df['Postal Code'].isna()]
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...
2234	2235	CA-2020-104066	2020-12-05	2020-12-10	Standard Class	QJ-19255	Quincy Jones	Corporate	United States	Burlington	...
5274	5275	CA-2018-162887	2018-11-07	2018-11-09	Second Class	SV-20785	Stewart Visinsky	Consumer	United States	Burlington	...
8798	8799	US-2019-150140	2019-04-06	2019-04-10	Standard Class	VM-21685	Valerie Mitchum	Home Office	United States	Burlington	...
9146	9147	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9147	9148	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9148	9149	US-2019-165505	2019-01-23	2019-01-27	Standard Class	CB-12535	Claudia Bergmann	Corporate	United States	Burlington	...
9386	9387	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9387	9388	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9388	9389	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9389	9390	US-2020-127292	2020-01-19	2020-01-23	Standard Class	RM-19375	Raymond Messe	Consumer	United States	Burlington	...
9741	9742	CA-2018-117086	2018-11-08	2018-11-12	Standard Class	QJ-19255	Quincy Jones	Corporate	United States	Burlington	...

11 rows × 21 columns

```
# TODO - Explore this dataset on your owns, ask your own questions
df[df['City']=='Burlington']
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...
683	684	US-2020-168116	2020-11-04	2020-11-04	Same Day	GT-14635	Grant Thornton	Corporate	United States	Burlington	...
684	685	US-2020-168116	2020-11-04	2020-11-04	Same Day	GT-14635	Grant Thornton	Corporate	United States	Burlington	...
1008	1009	US-2020-106705	2020-12-26	2021-01-01	Standard Class	PO-18850	Patrick O'Brill	Consumer	United States	Burlington	...
1038	1039	CA-2020-121818	2020-11-20	2020-11-21	First Class	JH-15430	Jennifer Halladay	Consumer	United States	Burlington	...
1039	1040	CA-2020-121818	2020-11-20	2020-11-21	First Class	JH-15430	Jennifer Halladay	Consumer	United States	Burlington	...
1393	1394	CA-2020-124828	2020-07-03	2020-07-04	First Class	YS-21880	Yana Sorensen	Corporate	United States	Burlington	...
2234	2235	CA-2020-104066	2020-12-05	2020-12-10	Standard Class	QJ-19255	Quincy Jones	Corporate	United States	Burlington	...
2928	2929	US-2020-120390	2020-10-19	2020-10-26	Standard Class	TH-21550	Tracy Hopkins	Home Office	United States	Burlington	...
5065	5066	CA-2020-142090	2020-11-30	2020-12-07	Standard Class	SC-20380	Shahid Collister	Consumer	United States	Burlington	...
5066	5067	CA-2020-142090	2020-11-30	2020-12-07	Standard Class	SC-20380	Shahid Collister	Consumer	United States	Burlington	...
5274	5275	CA-2018-	2018-11-07	2018-11-09	Second Class	SV-20785	Stewart Visinsky	Consumer	United States	Burlington	...

Data Analysis Part

```
# TODO 01 - how many columns, rows in this dataset
print(f"Rows: {df.shape[0]} \nColumns: {df.shape[1]}")
```

```
Rows: 9994
Columns: 21
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many nan v
print("Missing values in each column:")
print("There are 11 missing values from the 'Postal Code' column")
df.isna().sum()
```

```
Missing values in each column:
There are 11 missing values from the 'Postal Code' column


Row ID            0
Order ID          0
Order Date        0
Ship Date         0
Ship Mode         0
Customer ID       0
Customer Name     0
Segment           0
Country/Region    0
City              0
State             0
Postal Code      11
Region            0
Product ID        0
Category          0
Sub-Category      0
Product Name      0
Sales             0
Quantity          0
Discount          0
Profit            0
dtype: int64
```

```
# TODO 03 - your friend ask for `California` data, filter it and export csv for him
Califonia_data = df.query("State == 'California' ")
Califonia_data
#Califonia_data.to_csv('California_data.csv')
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
...	...	...	...	...	...	...	...	...	...	...	...
9986	9987	CA-2019-125794	2019-09-29	2019-10-03	Standard Class	ML-17410	Maris LaWare	Consumer	United States	Los Angeles	...
9990	9991	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...
9991	9992	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...
9992	9993	CA-2020-121258	2020-02-26	2020-03-03	Standard Class	DB-13060	Dave Brooks	Consumer	United States	Costa Mesa	...
9993	9994	CA-2020-119914	2020-05-04	2020-05-09	Second Class	CC-12220	Chris Cortes	Consumer	United States	Westminster	...

2001 rows × 21 columns

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 2017
cal_tex = df[df['Order Date'].dt.year == 2017]\
    .query("State == 'California' | State == 'Texas'")
cal_tex
#cal_tex.to_csv('California_Texas_2017.csv')
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...
...	...	...	...	...	...	...	...	...	...	...	...
9885	9886	CA-2017-112291	2017-04-03	2017-04-08	Standard Class	KE-16420	Katrina Edelman	Corporate	United States	Los Angeles	...
9903	9904	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United States	Carrollton	...
9904	9905	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United States	Carrollton	...
9942	9943	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United States	Anaheim	...
9943	9944	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United States	Anaheim	...

632 rows × 21 columns

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales yo
sales_2017 = df[df['Order Date'].dt.year == 2017]['Sales']\
    .agg(['sum', 'mean', 'std'])\
    .round(2)

print("Total sales, average sales, and the standard deviation of sales in 2017: ")
sales_2017
```

```
Total sales, average sales, and the standard deviation of sales in 2017:


sum     484247.50
mean       242.97
std        754.05
Name: Sales, dtype: float64
```

```
# TODO 06 - which Segment has the highest profit in 2018
highest_profit_segment_2018 = df[df['Order Date'].dt.strftime('%Y') == '2018']\
    .groupby('Segment')['Profit']\
    .agg('sum')\
    .sort_values(ascending=False)\
    .head(1)\
    .round(2)

print("The segment that had the highest profit in 2018: ")
highest_profit_segment_2018
```

```
The segment that had the highest profit in 2018:


Segment
Consumer    28460.17
Name: Profit, dtype: float64
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 - 3
lowest_sales = df[(df['Order Date'] >= '2019-04-15') & (df['Order Date'] <= '2019-1
    .groupby('State')['Sales']\
    .sum()\
    .sort_values()\
    .head(5)\
    .round(2)
print("The top 5 states that had the lowest total sales between 15 April 2019 - 31
lowest_sales
```

```
The top 5 states that had the lowest total sales between 15 April 2019 - 31 Decem
```

```
State
New Hampshire            49.05
New Mexico               64.08
District of Columbia    117.07
Louisiana               249.80
```

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e.g
sales_WestCentral_2019 = df[df['Order Date'].dt.year == 2019]\
    .query("Region == 'West' | Region == 'Central'")['Sales']\
    .sum()
sales_2019 = df[df['Order Date'].dt.year == 2019]['Sales']\
    .sum()
prop_sales_WestCentral_2019 = sales_WestCentral_2019/sales_2019
print(f"The proportion of total sales (%) in West + Central in 2019 were {(prop_sal
```

```
The proportion of total sales (%) in West + Central in 2019 were 54.97 %
```

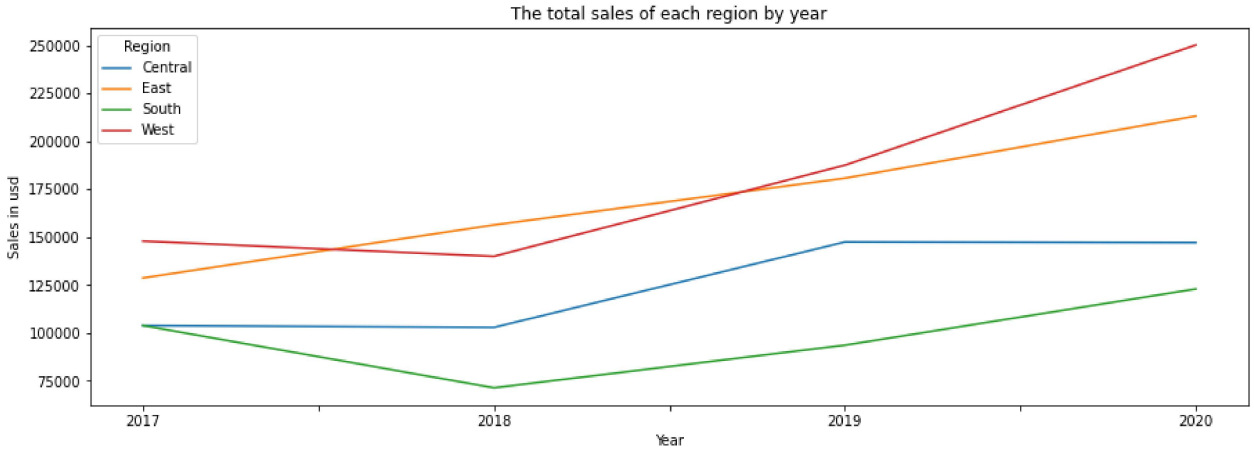
```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total sal
df_2019_2020 = df[(df['Order Date'].dt.year >= 2019) & (df['Order Date'].dt.year <=
by_orders = df_2019_2020\
    .value_counts('Product Name')\
    .sort_values(ascending=False)\
    .head(10)\
    .reset_index()
by_orders.columns = ['Top 10 Product by Orders', 'Number of Orders']
by_sales = df_2019_2020\
    .groupby('Product Name')[['Product Name', 'Sales']]\
    .agg('sum',numeric_only=True)\
    .sort_values(by='Sales', ascending=False)\
    .head(10)\
    .round(2)\
    .reset_index()
by_sales.columns = ['Top 10 Product by Sales', 'Total Sales']

by_orders_vs_sales = pd.concat([by_orders, by_sales], axis=1)
by_orders_vs_sales
```

	Top 10 Product by Orders	Number of Orders	Top 10 Product by Sales	Total Sales
0	Easy-staple paper	27	Canon imageCLASS 2200 Advanced Copier	61599.82
1	Staples	24	Hewlett Packard LaserJet 3310 Copier	16079.73
2	Staple envelope	22	3D Systems Cube Printer, 2nd Generation, Magenta	14299.89
3	Staples in misc. colors	13	GBC Ibimaster 500 Manual ProClick Binding System	13621.54
4	Staple remover	12	GBC DocuBind TL300 Electric Binding System	12737.26
5	Storex Dura Pro Binders	12	GBC DocuBind P400 Electric Binding System	12521.11
6	Chromcraft Round Conference Tables	12	Samsung Galaxy Mega 6.3	12263.71
7	Global Wood Trimmed Manager's Task Chair, Khaki	11	HON 5400 Series Task Chairs for Big and Tall	11846.56
8	Avery Non-Stick Binders	11	Martin Yale Chadless Opener Electric Letter Op...	11825.90
9	Staple-based wall hangings	10	Global Troy Executive Leather Low-Back Tilter	10169.89

```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
print("Plot 1 - The total sales of each region by year")
print("The total sales of each region have been gradually rising over time.")

df['Year'] = df['Order Date'].dt.strftime('%Y')

sales_by_region_per_year = df.groupby(['Year', 'Region'])['Sales'].agg('sum').reset

sales_by_region_per_year.pivot(columns='Region', index='Year', values='Sales').plot
```

```
Plot 1 - The total sales of each region by year
The total sales of each region have been gradually rising over time.
```

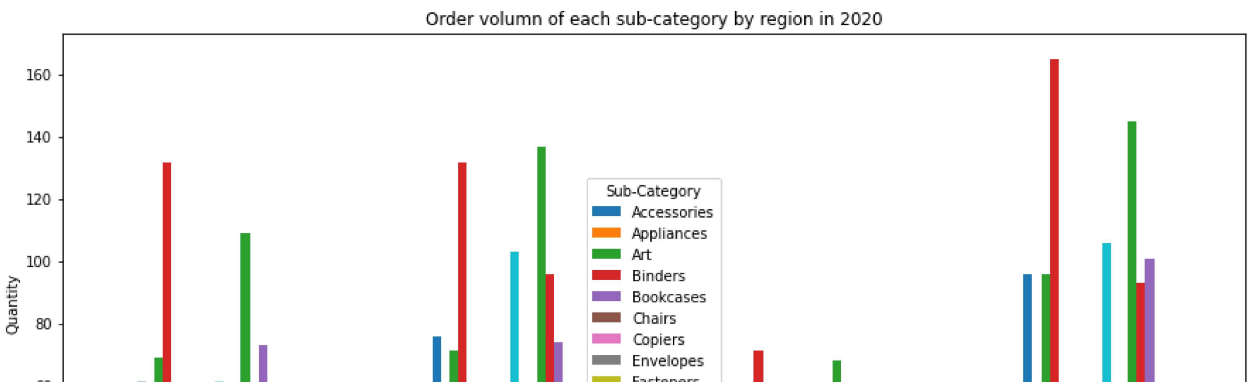
Download




```
print("Plot 2 - Order volumn of each sub-category by region in 2020")
print("In 2020, the most common items ordered in each region were binders and paper

subcat_2020 = df[df['Year'] == '2020'][['Region', 'Sub-Category']].value_counts().r

subcat_2020.columns = ['Region', 'Sub-Category', 'Quantity']

subcat_2020.pivot(columns='Sub-Category', index='Region', values='Quantity').plot(k
```

```
Plot 2 - Order volumn of each sub-category by region in 2020
In 2020, the most common items ordered in each region were binders and paper.
```

Download


Order volumn of each sub-category by region in 2020
Quantity
160
140
120
100
80
Sub-Category
Accessories
Appliances
Art
Binders
Bookcases
Chairs
Copiers
Envelopes