Final Project - Analyzing Sales Data

Date: 24 February 2023

Author: Nattakarn Lertdechachai

Course: `Pandas Foundation`

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

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

	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	Henderso
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderso
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 Lauderdal
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdal

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')
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   datetime64[ns]
 3   Ship Date       9994 non-null   datetime64[ns]
 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
```

```python
# TODO - count nan in postal code column
pos_nan = df["Postal Code"].isna().sum()

print(f"Postal Code column has {pos_nan} nan")
```

```
Postal Code column has 11 nan
```

```python
# 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
# Find Top 10 Best sales from this dataset
best_sales = df.dropna().sort_values('Sales', ascending=False).head(10)
best_sales[['Product Name', 'Sales', 'Quantity']]
```

	Product Name	Sales	Quantity
2697	Cisco TelePresence System EX90 Videoconferenci...	22638.480	6
6826	Canon imageCLASS 2200 Advanced Copier	17499.950	5
8153	Canon imageCLASS 2200 Advanced Copier	13999.960	4
2623	Canon imageCLASS 2200 Advanced Copier	11199.968	4
4190	Canon imageCLASS 2200 Advanced Copier	10499.970	3
9039	GBC Ibimaster 500 Manual ProClick Binding System	9892.740	13
4098	Ibico EPK-21 Electric Binding System	9449.950	5
4277	3D Systems Cube Printer, 2nd Generation, Magenta	9099.930	7
8488	HP Designjet T520 Inkjet Large Format Printer ...	8749.950	5
6425	Canon imageCLASS 2200 Advanced Copier	8399.976	4

Data Analysis Part

Answer 10 below questions to get credit from this course. Write `pandas` code to find answers.

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

```
rows: 9994, columns: 21
```

```
# TODO 03 - your friend ask for `California` data, filter it and export csv for h
california = df[df['State'] == 'California']
california.to_csv('california.csv')

#preview
california.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...	Post Code
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...	9003
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9003
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9003
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9003
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	9003

5 rows × 21 columns

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 201
df_2017 = df[df['Order Date'].dt.year == 2017]
cali_and_texas_2017 = df_2017.query('State == "California" | State == "Texas"')

#make csv file
cali_and_texas_2017.to_csv('cali_texas_2017.csv')

#preview
cali_and_texas_2017.head()
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...	Regi
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	West
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	West
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	West
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	West
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles	...	West

5 rows × 22 columns

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales
print('Sales Summary')
df_2017['Sales'].agg(['sum', 'mean', 'std']).round(2)

#Ans :
#Total sales in 2017 : 484247.50
#Average sales in 2017 : 242.97
#standard deviation of sales your company make in 2017 : 754.05
```

```
Sales Summary


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

```
# TODO 06 - which Segment has the highest profit in 2018
df_2018 = df[df['Order Date'].dt.year == 2018]
df_2018.groupby('Segment')['Profit'].max()

#Ans: 'Consumer' Segment has the highest profit in 2018 : 3177.48
```

```
Segment
Consumer       3177.4750
Corporate      2302.9671
Home Office    1906.4850
Name: Profit, dtype: float64
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 -
df_apr_dec_2019 = df[(df['Order Date'] >= '2019-04-15') & (df['Order Date'] <= '2
df_apr_dec_2019[['State', 'Sales']].sort_values('Sales').head()
```

	State	Sales
8658	Illinois	0.836
1112	Texas	1.192
2427	Texas	1.344
7356	Arizona	1.408
3213	Pennsylvania	1.504

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e
df_2019 = df[df['Order Date'].dt.year == 2019]
sales_2019 = df_2019['Sales'].sum()
sales_wc_2019 = df_2019.query('Region == "West" | Region == "Central"')['Sales'].

proportion_2019 = round((sales_wc_2019 / sales_2019)*100, 0)
print(proportion_2019)

#Ans: Proportion of total sales in West and Central in 2019 is 55%
```

```
55.0
```

```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
df_19_20 = df[df['Order Date'].dt.strftime('%Y').isin(['2019','2020'])]

#Number of orders
by_orders = df_19_20.value_counts('Product Name', ascending=False).head(10).reset
by_orders.columns = ['Product Name', 'Number of Orders']

#Total sales
by_sales = df_19_20.groupby('Product Name')[['Product Name','Sales']].sum()\
    .sort_values('Sales', ascending=False).head(10).round(2).reset_index()
by_sales.columns = ['Product Name', 'Total Sales']


#Merge column
orders_vs_sales = pd.concat([by_orders, by_sales], axis=1)
orders_vs_sales
```

	Product Name	Number of Orders	Product Name	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 :)
# 10.1 - The total sales of each category by year
# 10.2 - The total sales order of this company categorized by sub productcategory
```

```
# TODO Bonus - use np.where() to create new column in dataframe to help you answe
#Show products which make the most profit
import numpy as np
df_product = df.groupby('Product Name').sum('Profit')[['Sales', 'Quantity', 'Prof
    .sort_values('Profit', ascending=False)
df_product['Levels'] = np.where(df_product['Profit'] > 10000, 'High Profit',
                           np.where(df_product['Profit'] >= 4000, 'Average', 'Low
df_product
```

	Sales	Quantity	Profit	Levels
Product Name				
Canon imageCLASS 2200 Advanced Copier	61599.824	20	25199.9280	High Profit
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	27453.384	31	7753.0390	Average
Hewlett Packard LaserJet 3310 Copier	18839.686	38	6983.8836	Average
Canon PC1060 Personal Laser Copier	11619.834	19	4570.9347	Average
HP Designjet T520 Inkjet Large Format Printer - 24" Color	18374.895	12	4094.9766	Average
...	...	...	...	...
Bush Advantage Collection Racetrack Conference Table	9544.725	33	-1934.3976	Low Profit
Chromcraft Bull-Nose Wood Oval Conference Tables & Bases	9917.640	27	-2876.1156	Low Profit
Cubify CubeX 3D Printer Triple Head Print	7999.980	4	-3839.9904	Low Profit
Lexmark MX611dhe Monochrome Laser Printer	16829.901	18	-4589.9730	Low Profit
Cubify CubeX 3D Printer Double Head Print	11099.963	9	-8879.9704	Low Profit

1849 rows × 4 columns