Final Project - Analyzing Sales Data

Date: 21 January 2023

Author: Nattawat Mangmati (Yim)

Course: `Pandas Foundation`

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

```python
# preview top 5 rows
df.head(500)
```

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country/Region	City	...
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	...
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	...
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale	...
...	...	...	...	...	...	...	...	...	...	...	...
495	496	CA-2018-134782	2018-12-27	2018-12-31	Standard Class	MD-17350	Maribeth Dona	Consumer	United States	Fayetteville	...
496	497	CA-2019-126158	2019-07-25	2019-07-31	Standard Class	SC-20095	Sanjit Chand	Consumer	United States	Costa Mesa	...
497	498	CA-2019-126158	2019-07-25	2019-07-31	Standard Class	SC-20095	Sanjit Chand	Consumer	United States	Costa Mesa	...
498	499	CA-2019-126158	2019-07-25	2019-07-31	Standard Class	SC-20095	Sanjit Chand	Consumer	United States	Costa Mesa	...
499	500	CA-2019-126158	2019-07-25	2019-07-31	Standard Class	SC-20095	Sanjit Chand	Consumer	United States	Costa Mesa	...

500 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='%Y/%m/%d')
df['Ship Date'] = pd.to_datetime(df['Ship Date'], format='%Y/%m/%d')
print(df.dtypes)
```

```
Row ID                     int64
Order ID                  object
Order Date        datetime64[ns]
Ship Date         datetime64[ns]
Ship Mode                 object
Customer ID               object
Customer Name             object
Segment                   object
Country/Region            object
City                      object
State                     object
Postal Code              float64
Region                    object
Product ID                object
Category                  object
Sub-Category              object
Product Name              object
Sales                    float64
Quantity                   int64
Discount                 float64
```

```
# TODO - count nan in postal code column
nan_count = df['Postal Code'].isna().sum()
print(nan_count)
```

```
11
```

```
# TODO - filter rows with missing values
filtered_df = df[df.isna().any(axis=1)]
print(filtered_df)
```

```
      Row ID        Order ID Order Date  Ship Date       Ship Mode  \
2234    2235  CA-2020-104066 2020-12-05 2020-12-10  Standard Class
5274    5275  CA-2018-162887 2018-11-07 2018-11-09    Second Class
8798    8799  US-2019-150140 2019-04-06 2019-04-10  Standard Class
9146    9147  US-2019-165505 2019-01-23 2019-01-27  Standard Class
9147    9148  US-2019-165505 2019-01-23 2019-01-27  Standard Class
9148    9149  US-2019-165505 2019-01-23 2019-01-27  Standard Class
9386    9387  US-2020-127292 2020-01-19 2020-01-23  Standard Class
9387    9388  US-2020-127292 2020-01-19 2020-01-23  Standard Class
9388    9389  US-2020-127292 2020-01-19 2020-01-23  Standard Class
9389    9390  US-2020-127292 2020-01-19 2020-01-23  Standard Class
9741    9742  CA-2018-117086 2018-11-08 2018-11-12  Standard Class

     Customer ID     Customer Name      Segment Country/Region        City  \
2234    QJ-19255      Quincy Jones    Corporate  United States  Burlington
5274    SV-20785  Stewart Visinsky     Consumer  United States  Burlington
8798    VM-21685   Valerie Mitchum  Home Office  United States  Burlington
9146    CB-12535  Claudia Bergmann    Corporate  United States  Burlington
9147    CB-12535  Claudia Bergmann    Corporate  United States  Burlington
9148    CB-12535  Claudia Bergmann    Corporate  United States  Burlington
```

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
```

```
(9994, 21)
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many nan
missing_values = df.isna().sum()
print(missing_values)
print(missing_values[missing_values>0])
```

```
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
```

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

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 201

filtered_df = df[(df['State'] == 'California') | (df['State'] == 'Texas') & (df['
filtered_df.to_csv("filtered_orders.csv", index = False)
```

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales
df.describe([])
```

	Row ID	Postal Code	Sales	Quantity	Discount	Profit
count	9994.000000	9983.000000	9994.000000	9994.000000	9994.000000	9994.000000
mean	4997.500000	55245.233297	229.858001	3.789574	0.156203	28.656896
std	2885.163629	32038.715955	623.245101	2.225110	0.206452	234.260108
min	1.000000	1040.000000	0.444000	1.000000	0.000000	-6599.978000
25%	2499.250000	23223.000000	17.280000	2.000000	0.000000	1.728750
50%	4997.500000	57103.000000	54.490000	3.000000	0.200000	8.666500
75%	7495.750000	90008.000000	209.940000	5.000000	0.200000	29.364000
max	9994.000000	99301.000000	22638.480000	14.000000	0.800000	8399.976000

```
# TODO 06 - which Segment has the highest profit in 2018
filtered_df = df[df['Order Date'].dt.year == 2018]
segment_profit = filtered_df.groupby(['Segment'])['Profit'].sum()
highest_profit_segment = segment_profit.idxmax()
print("Segment with highest profit in 2018:", highest_profit_segment)
```

```
Segment with highest profit in 2018: Consumer
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 -
filtered_df = df[(df['Order Date'] >= '2019-04-15') & (df['Order Date'] <= '2019-
state_sales = filtered_df.groupby(['State'])['Sales'].sum()
least_sales_states = state_sales.nsmallest(5)
print("Top 5 states with the least total sales:", least_sales_states)
```

```
Top 5 states with the least total sales: State
New Hampshire            49.05
New Mexico               64.08
District of Columbia    117.07
Louisiana               249.80
South Carolina          502.48
Name: Sales, dtype: float64
```

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e
filtered_df = df[df['Order Date'].dt.year == 2019]
filtered_df = filtered_df[(filtered_df['Region'] == 'West') | (filtered_df['Regio
total_sales = filtered_df['Sales'].sum()
region_sales = filtered_df.groupby(['Region'])['Sales'].sum()
prop = (region_sales['West']+region_sales['Central'])/total_sales*100
print("Proportion of total sales in West + Central in 2019:", prop, "%")
```

```
Proportion of total sales in West + Central in 2019: 100.0 %
```

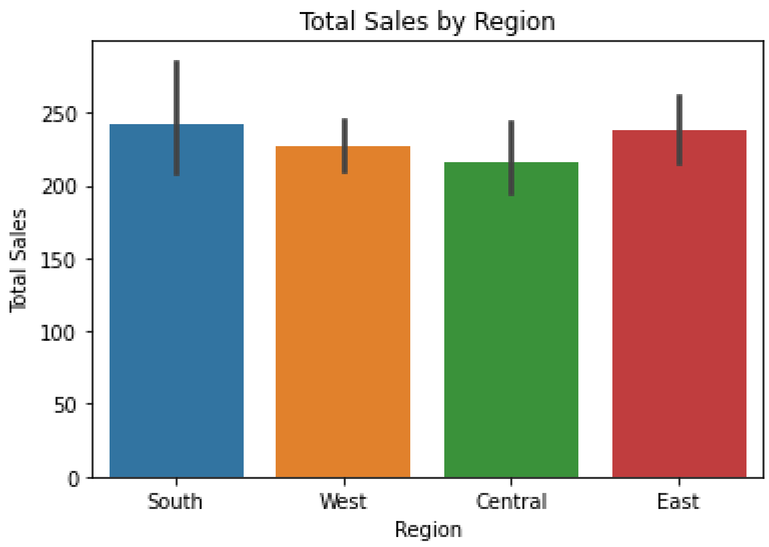
```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
filtered_df = df[(df['Order Date'] >= '2019-01-01') & (df['Order Date'] <= '2020-
product_orders_sales = filtered_df.groupby(['Product Name']).agg({'Order ID':'nun
top_10_orders = product_orders_sales.sort_values(by='Order ID', ascending=False).
top_10_sales = product_orders_sales.sort_values(by='Sales', ascending=False).head
print("Top 10 products in terms of number of orders:", top_10_orders)
print("Top 10 products in terms of total sales:", top_10_sales)
```

```
Top 10 products in terms of number of orders:
Product Name
```

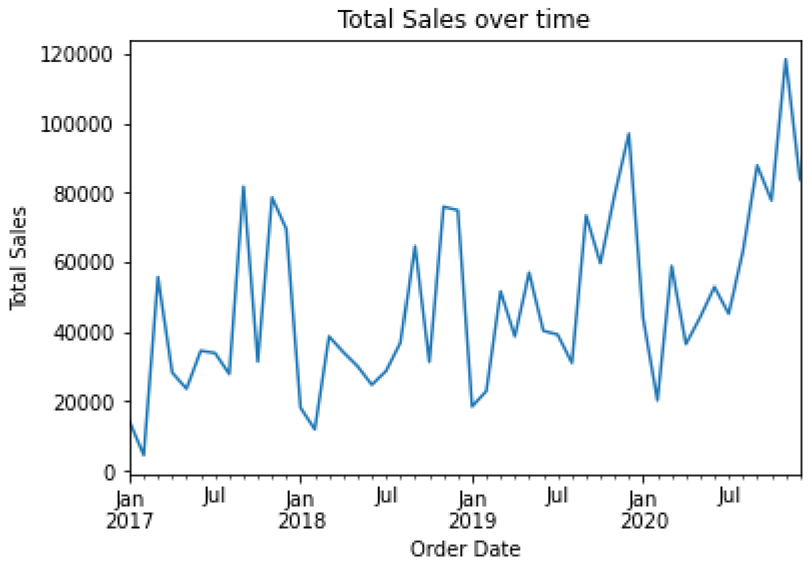
```
Easy-staple paper                                   27   1481.728
Staples                                             24    462.068
Staple envelope                                     22    644.936
Staples in misc. colors                             13    357.164
Staple remover                                      12    204.512
Storex Dura Pro Binders                             12    176.418
Chromcraft Round Conference Tables                  12   7965.053
Global Wood Trimmed Manager's Task Chair, Khaki     11   2793.086
Avery Non-Stick Binders                             11    122.128
Staple-based wall hangings                          10    233.392
Top 10 products in terms of total sales:
Product Name
Canon imageCLASS 2200 Advanced Copier                   5  61599.824
Hewlett Packard LaserJet 3310 Copier                    6  16079.732
3D Systems Cube Printer, 2nd Generation, Magenta        2  14299.890
GBC Ibimaster 500 Manual ProClick Binding System        5  13621.542
GBC DocuBind TL300 Electric Binding System              6  12737.258
```

```python
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
import matplotlib.pyplot as plt
import seaborn as sns
sns.barplot(x='Region', y='Sales', data=df)
plt.xlabel('Region')
plt.ylabel('Total Sales')
plt.title('Total Sales by Region')
plt.show()
df['Order Date'] = pd.to_datetime(df['Order Date'])
df.set_index('Order Date', inplace=True)
df.resample('M').sum()['Sales'].plot()
plt.xlabel('Order Date')
plt.ylabel('Total Sales')
plt.title('Total Sales over time')
plt.show()
```

Download



Download



```
# TODO Bonus - use np.where() to create new column in dataframe to help you answe
# I want to create a new column called "Discount" that shows whether the customer
import numpy as np

df['Discount'] = np.where(df['Sales'] > 1000, 'Yes', 'No')
discount_orders = df[df['Discount']=='Yes'].shape[0]
print("Number of orders with discounts:", discount_orders)
```

```
Number of orders with discounts: 468
```