Final Project - Analyzing Sales Data

Date: 30 November 2022

Author: Phawat.R (kane)

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')
```

```
# TODO - count nan in postal code column
```

```
df['Postal Code'].isna().sum()
```

```
11
```

```
# 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	...

```
# TODO - Explore this dataset on your owns, ask your own questions
# - Which customer has the highest order?
```

```
df.head()
df.groupby(["Customer ID",'Customer Name'])['Quantity'].sum().reset_index().sort_
```

	Customer ID	Customer Name	Quantity
349	JD-15895	Jonathan Doherty	150
787	WB-21850	William Brown	146
387	JL-15835	John Lee	143
606	PP-18955	Paul Prost	138
678	SC-20725	Steven Cartwright	133
275	EP-13915	Emily Phan	124
125	CB-12025	Cassandra Brandow	122
147	CK-12205	Chloris Kastensmidt	122
257	EH-13765	Edward Hooks	120
482	MA-17560	Matt Abelman	117

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('Ans: 21 columns and 9994 rows')
```

```
Ans: 21 columns and 9994 rows
```

```
# TODO 02 - is there any missing values?, if there is, which colunm? how many nan
```

```
df.isna().sum()
print('Ans: Yes, there is.')
print('column Postal Code: 11 nan values')
```

```
Ans: Yes, there is.
column Postal Code: 11 nan values
```

```
# TODO 03 - your friend ask for `California` data, filter it and export csv for h
```

```
df_CA = df[df['State'] =='California']
df_CA.to_csv('data_store_CA.csv')
```

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

```
# df['Year'] = pd.DatetimeIndex(df['Order Date']).year
df_CA_TX = df.query('State == "California" | State == "Texas"')
df_CA_TX = df_CA_TX[df_CA_TX['Order Date'].dt.strftime('%Y') == "2017"].reset_ind
df_CA_TX.to_csv('data_store_CA_TX_2017.csv')
```

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales
```

```
df_2017 = df[df['Order Date'].dt.strftime('%Y') == "2017"]
df_2017['Sales'].agg(['sum','mean','std']).round(2)
```

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

```
# TODO 06 - which Segment has the highest profit in 2018
```

```
df[df['Order Date'].dt.strftime('%Y') == "2018"].groupby('Segment')['Profit'].su
print('Ans: Consumer segment has the highest profit in 2018')
```

```
Ans: Consumer segment has the highest profit in 2018
```

```
# TODO 07 - which top 5 States have the least total sales between 15 April 2019 -
```

```
df_filter = df[(df['Order Date'] >= '2019-04-15') & (df['Order Date'] <= '2019-12
df_filter.groupby('State')['Sales'].sum().sort_values().head(5)
```

```
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
```

```
df_2019 = df[(df['Order Date'].dt.strftime('%Y') == '2019') & ((df['Region'] == '
df_2019_T = df[df['Order Date'].dt.strftime('%Y')== '2019']['Sales'].sum()
#df_2019_T = df[df['Order Date'].dt.year == 2019]['Sales'].sum()
Proportion = (df_2019/df_2019_T)*100
print(f"Propotion of total sale(West + Central): {Proportion.round(2)} %")
```

```
Propotion of total sale(West + Central): 54.97 %
```

```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
```

```
#df
df_19_20 = df[(df['Order Date'].dt.strftime('%Y') == '2019') | (df['Order Date'].
df_TopOrder = df_19_20.groupby(['Product ID','Product Name'])['Order ID'].count()
df_TopSale = df_19_20.groupby(['Product ID','Product Name'])['Sales'].sum().sort_
```

```
df_TopOrder
```

	Product ID	Product Name	Order ID
0	FUR-TA-10001095	Chromcraft Round Conference Tables	12
1	FUR-CH-10003774	Global Wood Trimmed Manager's Task Chair, Khaki	11
2	OFF-BI-10000301	GBC Instant Report Kit	10
3	OFF-ST-10001325	Sterilite Officeware Hinged File Box	10
4	OFF-BI-10001989	Premium Transparent Presentation Covers by GBC	9
5	FUR-TA-10003473	Bretford Rectangular Conference Table Tops	9
6	OFF-BI-10004364	Storex Dura Pro Binders	9
7	TEC-AC-10004510	Logitech Desktop MK120 Mouse and keyboard Combo	9
8	OFF-BI-10004236	XtraLife ClearVue Slant-D Ring Binder, White, 3"	9
9	FUR-CH-10000454	Hon Deluxe Fabric Upholstered Stacking Chairs,...	9

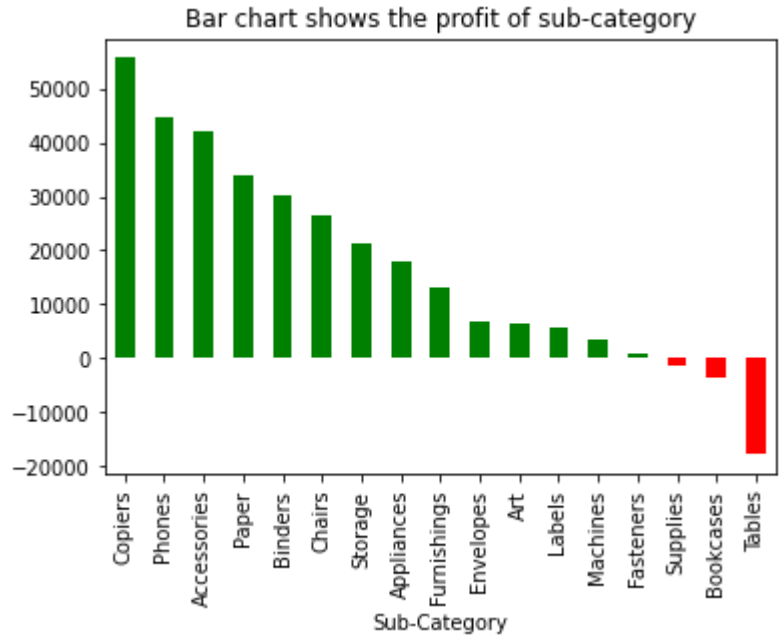
```
df_TopSale
```

	Product ID	Product Name	Sales
0	TEC-CO-10004722	Canon imageCLASS 2200 Advanced Copier	61599.824
1	TEC-CO-10001449	Hewlett Packard LaserJet 3310 Copier	16079.732
2	TEC-MA-10001047	3D Systems Cube Printer, 2nd Generation, Magenta	14299.890
3	OFF-BI-10000545	GBC Ibimaster 500 Manual ProClick Binding System	13621.542
4	OFF-BI-10001359	GBC DocuBind TL300 Electric Binding System	12737.258
5	OFF-BI-10004995	GBC DocuBind P400 Electric Binding System	12521.108
6	TEC-PH-10001459	Samsung Galaxy Mega 6.3	12263.708
7	FUR-CH-10002024	HON 5400 Series Task Chairs for Big and Tall	11846.562
8	OFF-SU-10002881	Martin Yale Chadless Opener Electric Letter Op...	11825.902
9	FUR-CH-10001215	Global Troy Executive Leather Low-Back Tilter	10169.894

```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
```

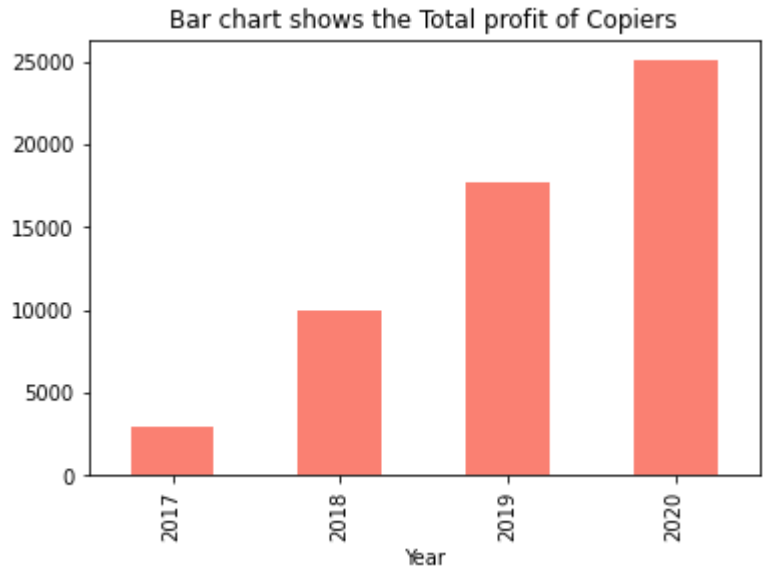
```
df_bar = df.groupby("Sub-Category")['Profit'].sum().sort_values(ascending = False
df_bar.plot(kind ="bar" , color = ( df_bar>0 ).map({True: 'g', False: 'r'})
            , title ='Bar chart shows the profit of sub-category');
```

Download



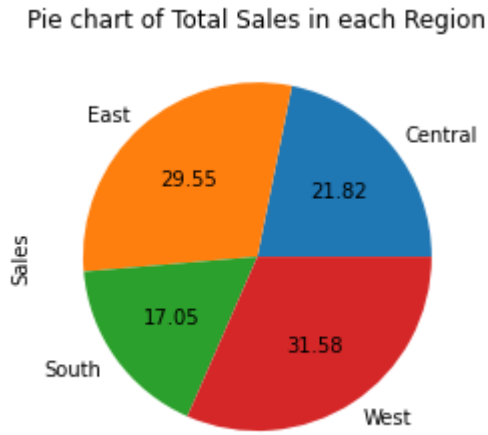

```
df['Year'] = pd.DatetimeIndex(df['Order Date']).year
#df['Month'] = pd.DatetimeIndex(df['Order Date']).month
df_prep = df[['Year','Sub-Category','Profit']]
df_pivot = df_prep.pivot_table(index = 'Year' , columns ='Sub-Category' , values
df_pivot['Copiers'].plot.bar(title = 'Bar chart shows the Total profit of Copiers
```

Download



```
df.groupby('Region')['Sales'].sum().plot(kind ='pie'
    ,title = "Pie chart of Total Sales in each Region"
    ,autopct = '%.2f');
```

Download




```
# TODO Bonus - use np.where() to create new column in dataframe to help you answe
```

```
import numpy as np
import pandas as pd

df["Gain"] = np.where(df["Profit"] >0, True, False)
df_gain = df[df['Gain'] == True].groupby('Year')['Gain'].count()
df_loss = df[df['Gain'] == False].groupby('Year')['Gain'].count()
df_merge = pd.merge(df_gain,df_loss, on = "Year")
df_merge.columns = ['Grain','Loss']
df_merge
#df_2019_T = df[df['Order Date'].dt.year == 2019]['Sales'].sum()
```

	Grain	Loss
Year		
2017	1599	394
2018	1696	406
2019	2090	497
2020	2673	639