Final Project - Analyzing Sales Data

Date: 02 December 2022

Author: Thitima Moungsesai (Bee)

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	Hend
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Hend
2	3	CA-2019-138688	6/12/2019	6/16/2019	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Ange
3	4	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Laude
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Laude

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(50), format='%m/%d/%Y')
```

```
0    2019-11-08
1    2019-11-08
2    2019-06-12
3    2018-10-11
4    2018-10-11
5    2017-06-09
6    2017-06-09
7    2017-06-09
8    2017-06-09
9    2017-06-09
10   2017-06-09
11   2017-06-09
12   2020-04-15
13   2019-12-05
14   2018-11-22
15   2018-11-22
16   2017-11-11
17   2017-05-13
18   2017-08-27
19   2017-08-27
20   2017-08-27
21   2019-12-09
22   2019-12-09
23   2020-07-16
24   2018-09-25
25   2019-01-16
26   2019-01-16
27   2018-09-17
28   2018-09-17
29   2018-09-17
30   2018-09-17
31   2018-09-17
32   2018-09-17
33   2018-09-17
34   2020-10-19
35   2019-12-08
36   2019-12-08
37   2018-12-27
38   2018-12-27
39   2018-12-27
40   2018-12-27
41   2020-09-10
42   2019-07-17
43   2020-09-19
44   2019-03-11
45   2019-03-11
46   2017-10-20
47   2019-06-20
48   2019-06-20
49   2018-04-18
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
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.head()
#df.groupby('City')['Profit'].sum().reset_index().sort_values('Profit',ascending=
df.groupby(['State','City','Segment'])['Profit'].sum().reset_index().sort_values(
#df.groupby('Sub-Category')['Profit'].sum().reset_index().sort_values('Profit',as
```

	State	City	Segment	Profit
735	New York	New York City	Consumer	30618.8441
737	New York	New York City	Home Office	17333.3985
124	California	Los Angeles	Consumer	16508.7529
1141	Washington	Seattle	Consumer	16389.6527
736	New York	New York City	Corporate	14084.7411

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

df_California.to_csv('df_California.csv')
```

```
# TODO 04 - your friend ask for all order data in `California` and `Texas` in 201
df_2017 = df[df['Order Date'].dt.strftime('%Y') == '2017'].reset_index()
df_ca_tex_2017 = df_2017[(df_2017['State'] =='California') | (df_2017['State'] ==
#df_ca_tex_2017

df_ca_tex_2017.to_csv('df_ca_tex_2017.csv')
```

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

df_2017['Sales'].agg(['sum','mean','std']).reset_index()
```

	index	Sales
0	sum	484247.498100
1	mean	242.974159
2	std	754.053357

```
# TODO 06 - which Segment has the highest profit in 2018
df_2018 = df[df['Order Date'].dt.strftime('%Y') == '2018'].reset_index()
highest_profit_of_Segment_2018 = df_2018.groupby('Segment')[['Sales']].sum().sort
highest_profit_of_Segment_2018
```

	Sales
Segment	
Consumer	266535.9333

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

sale_20190415_n_20191231 = df[(df['Order Date'].dt.strftime('%Y-%m-%d')).between(
least_state_sale_20190415_n_20191231 = sale_20190415_n_20191231\
                                .groupby('State')[['State','Sales']].sum()\
                                .sort_values('Sales',ascending = True).head(5)
least_state_sale_20190415_n_20191231
```

	Sales
State	
New Hampshire	49.05
New Mexico	64.08
District of Columbia	117.07
Louisiana	249.80
South Carolina	502.48

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e
#Region
df_2019 = df[df['Order Date'].dt.strftime('%Y') == '2019'].reset_index()
Sales_West_Central = df_2019.query("Region == 'West' or Region == 'Central'")['Sa
Sales_Total = df_2019['Sales'].sum()


# Calculate Percentage
Sales_Region_Percentage =  round((Sales_West_Central/Sales_Total*100),2)
Sales_Region_Percentage
```

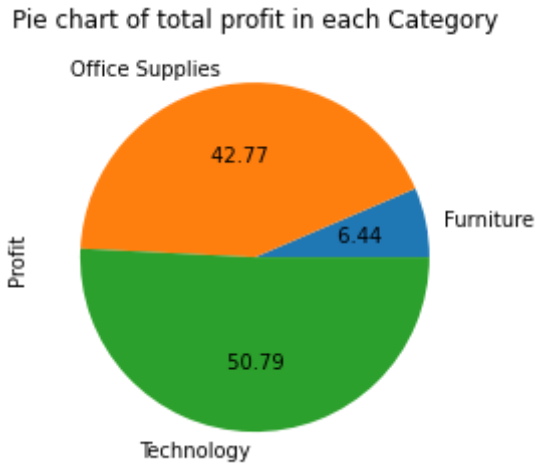
```
54.97
```

```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
df_2019_n_2020 = df[(df['Order Date'].dt.strftime('%Y')).between('2019', '2020')]
Test = df_2019_n_2020.groupby(['Product ID','Product Name'],as_index=False)['Orde

df_Top10_Order = df_2019_n_2020.groupby(['Product ID','Product Name'])['Order ID'
df_Top10_Sales = df_2019_n_2020.groupby(['Product ID','Product Name'])['Sales'].c
```

```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
df.head()
df.groupby('Category')['Profit'].sum().plot(kind = 'pie',\
              title = "Pie chart of total profit in each Category",\
              autopct =' %.2f')
```

```
<AxesSubplot:title={'center':'Pie chart of total profit in each Category'}, yla
```

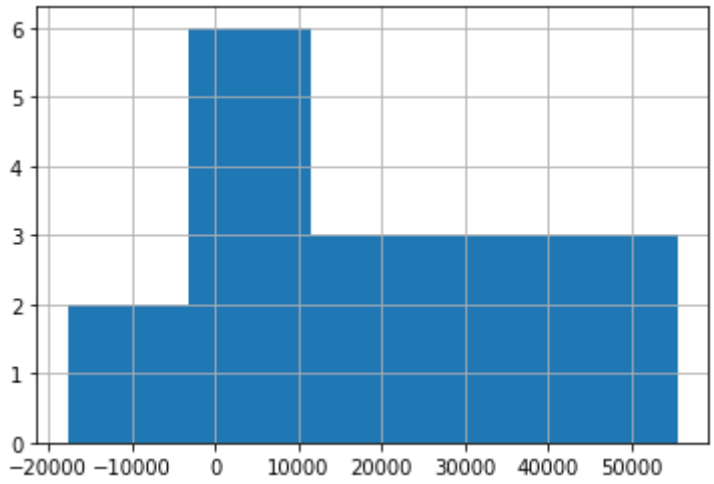
Download




```
df.head()
df.groupby('Sub-Category')['Profit'].sum().sort_values(ascending = False).hist(bi
```

```
<AxesSubplot:>
```

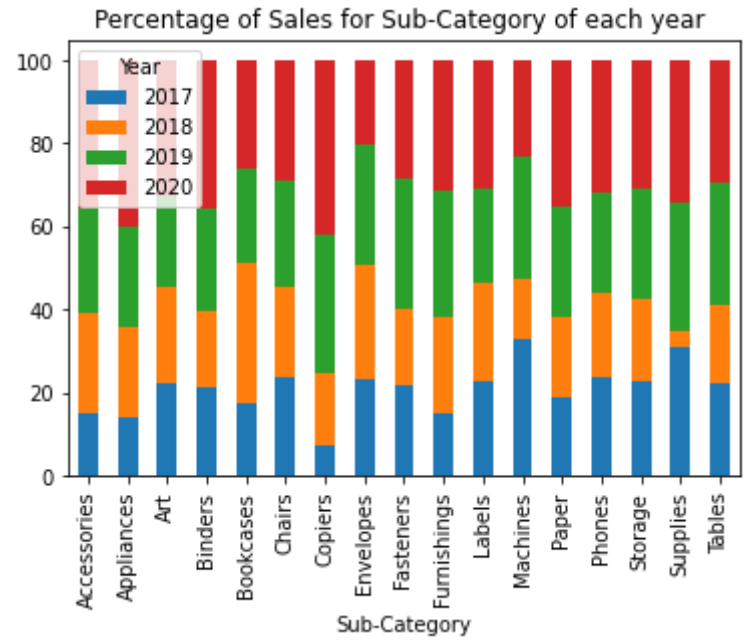
Download



```
df['Year'] = pd.DatetimeIndex(df['Order Date']).year   #Add column
#df.head()


data = df.groupby(['Sub-Category','Year'])['Sales'].sum().reset_index().sort_valu
data = data.pivot(index='Sub-Category', columns='Year', values='Sales')
data
#data.plot(kind="bar", stacked=True)
# plot the pivoted dataframe; if the column names aren't colors, remove color=df.
#data.plot()

stacked_data = data.apply(lambda x: x*100/sum(x), axis=1)
stacked_data.plot(kind="bar", stacked=True,
  title = "Percentage of Sales for Sub-Category of each year ")

#https://www.shanelynn.ie/bar-plots-in-python-using-pandas-dataframes/
```

```
<AxesSubplot:title={'center':'Percentage of Sales for Sub-Category of each year
```

Download



```
# TODO Bonus - use np.where() to create new column in dataframe to help you answe
import numpy as np
df['Staus_Profit'] = np.where(df['Profit'] < 0,"Loss","Profit")
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	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	...

5 rows × 23 columns