Final Project - Analyzing Sales Data

Date: 19 November 2022

Author: Therarat Srisaswatakul (Boss)

Course: `Pandas Foundation`

```
# import data
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
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	Hendersc
1	2	CA-2019-152156	11/8/2019	11/11/2019	Second Class	CG-12520	Claire Gute	Consumer	United States	Hendersc
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 Lauderda
4	5	US-2018-108966	10/11/2018	10/18/2018	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderda

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')
```

```
# 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.head()
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/region	c
0	1	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	H
1	2	CA-2019-152156	2019-11-08	2019-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	H
2	3	CA-2019-138688	2019-06-12	2019-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	L A
3	4	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	F L
4	5	US-2018-108966	2018-10-11	2018-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	F L

5 rows × 21 columns

```
# TODO - count nan in postal code column
cols = df.columns
clean_cols = [col.lower().replace(" ", "_").replace("-", "_") for col in cols]
df.columns = clean_cols
total_nan = df['postal_code'].isna().sum()
print(f"The total NA in postal code column is {total_nan} values")
```

```
The total NA in postal code column is 11 values
```

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

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/reg
2234	2235	CA-2020-104066	12/5/2020	12/10/2020	Standard Class	QJ-19255	Quincy Jones	Corporate	United State
5274	5275	CA-2018-162887	11/7/2018	11/9/2018	Second Class	SV-20785	Stewart Visinsky	Consumer	United State
8798	8799	US-2019-150140	4/6/2019	4/10/2019	Standard Class	VM-21685	Valerie Mitchum	Home Office	United State
9146	9147	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State
9147	9148	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State
9148	9149	US-2019-165505	1/23/2019	1/27/2019	Standard Class	CB-12535	Claudia Bergmann	Corporate	United State
9386	9387	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9387	9388	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9388	9389	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9389	9390	US-2020-127292	1/19/2020	1/23/2020	Standard Class	RM-19375	Raymond Messe	Consumer	United State
9741	9742	CA-2018-117086	11/8/2018	11/12/2018	Standard Class	QJ-19255	Quincy Jones	Corporate	United State

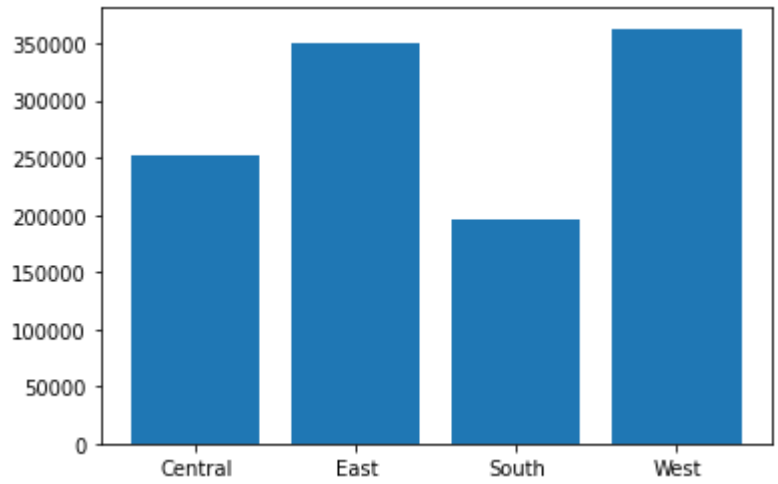
11 rows × 21 columns

```
# TODO - Explore this dataset on your owns, ask your own questions
# Calculate the total sales on each region group by segment

df2 = df.groupby(['region', 'segment'])['sales'].agg(['sum','count']).reset_index
plt.bar(df2['region'], df2['sum'])
```

```
<BarContainer object of 12 artists>
```

Download



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
total_row = df.shape[0]
total_col = df.shape[1]

print(f"Total row : {total_row} & Total column : {total_col}")
```

```
Total row : 9994 & Total column : 21
```

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

total_nan = df.isna().sum().sort_values(ascending = False).head()
print(total_nan)
```

```
postal_code    11
row_id          0
discount        0
quantity        0
sales           0
dtype: int64
```

```python
# TODO 03 - your friend ask for `California` data, filter it and export csv for h
filtered_result = df[df['state'] == 'California']

filtered_result.to_csv("california_data.csv")
```

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

filtered_result2 = df[((df['state'] == 'California') | (df['state'] == 'Texas'))

#filtered_result2.to_csv("california_taxas_2017.csv") >>> sent to your friend

filtered_result2
```

	row_id	order_id	order_date	ship_date	ship_mode	customer_id	customer_name	segment	country/reg
5	6	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United State
6	7	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United State
7	8	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United State
8	9	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United State
9	10	CA-2017-115812	2017-06-09	2017-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United State
...	...	...	...	...	...	...	...	...	...
9885	9886	CA-2017-112291	2017-04-03	2017-04-08	Standard Class	KE-16420	Katrina Edelman	Corporate	United State
9903	9904	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United State
9904	9905	CA-2017-122609	2017-11-12	2017-11-18	Standard Class	DP-13000	Darren Powers	Consumer	United State
9942	9943	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United State
9943	9944	CA-2017-143371	2017-12-28	2018-01-03	Standard Class	MD-17350	Maribeth Dona	Consumer	United State

632 rows × 22 columns

```
# TODO 05 - how much total sales, average sales, and standard deviation of sales
df5 = df[(df['order_date'] >= '2017-01-01') & (df['order_date'] <= '2017-12-31')]
        ['sales'].agg(['sum', 'mean', 'std' ]).reset_index()

df5
```

	index	sales
0	sum	484247.498100
1	mean	242.974159
2	std	754.053357

```
# TODO 06 - which Segment has the highest profit in 2018
df6 = df[(df['order_date'] >= '2018-01-01') & (df['order_date'] <= '2018-12-31')]
        .groupby('segment')['profit'].sum()\
        .head(1)

df6
```

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

df7 = df[(df['order_date'] >= '2019-04-15') & (df['order_date'] <= '2019-12-31')]
            .groupby('state')['sales'].sum()\
            .head()

df7
```

```
# TODO 08 - what is the proportion of total sales (%) in West + Central in 2019 e
df8 = df[(df['order_date'] >= '2019-04-15') & (df['order_date'] <= '2019-12-31')]
            .groupby('region')['sales'].sum().reset_index()

west = df8.iloc[3, 1]
central = df8.iloc[0, 1]

print(f"Proportion of total sales of West : Central is {round((west)/(central), 2
```

```
Proportion of total sales of West : Central is 1.28
```

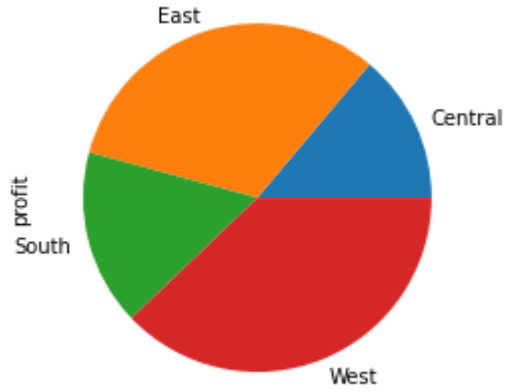
```
# TODO 09 - find top 10 popular products in terms of number of orders vs. total s
df9 = df[(df['order_date'] <= '2020-12-31') & (df['order_date'] >= '2019-01-01')]
            .groupby('product_name')['product_name'].count().head(10)

df9
```

```
# TODO 10 - plot at least 2 plots, any plot you think interesting :)
df10 = df[['region', 'profit']].groupby('region')['profit'].sum()
df10.plot(kind = 'pie')
```

```
<AxesSubplot:ylabel='profit'>
```

Download




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

df['profitable'] = np.where(df['profit'] > 0, "Profit", "Lost")
df[['order_date', 'city', 'state', 'profitable']]
```

	order_date	city	state	profitable
0	2019-11-08	Henderson	Kentucky	Profit
1	2019-11-08	Henderson	Kentucky	Profit
2	2019-06-12	Los Angeles	California	Profit
3	2018-10-11	Fort Lauderdale	Florida	Lost
4	2018-10-11	Fort Lauderdale	Florida	Profit
...	...	...	...	...
9989	2017-01-21	Miami	Florida	Profit
9990	2020-02-26	Costa Mesa	California	Profit
9991	2020-02-26	Costa Mesa	California	Profit
9992	2020-02-26	Costa Mesa	California	Profit
9993	2020-05-04	Westminster	California	Profit

9994 rows × 4 columns