Case study: Analyze Sales

By: Juan David Serna Valderrama

Some skills that we are going to use in this case study:

- Drop NaN values from DataFrame
- Removing rows based on a condition
- Change the type of columns (to_numeric, to_datetime, astype)

To explore 5 high-level business questions related to our data:

1. What was the best month for sales? How much was earned that month?
2. What city sold the most product?
3. What time should we display advertisements to maximize the likelihood of a customer’s buying a product?
4. What products are most often sold together?
5. What product sold the most? Why do you think it sold the most?

Import libraries

In [93]:
```
import pandas as pd
import os
import matplotlib.pyplot as plt
```

We have 12 files.csv. To join in a single file

In [94]:
```
df = pd.read_csv("C:/Users/juand/OneDrive/Escritorio/Sales_Data/Sales_April_2019.csv")

files = [file for file in os.listdir("C:/Users/juand/OneDrive/Escritorio/Sales_Data")]

all_months_data = pd.DataFrame()
for file in files:
    df = pd.read_csv("C:/Users/juand/OneDrive/Escritorio/Sales_Data/" + file)
    all_months_data = pd.concat([all_months_data, df])

all_months_data.to_csv("all_data.csv", index= False)
```

Read new file (updated)

In [95]:
```
all_data = pd.read_csv("C:/Users/juand/OneDrive/Escritorio/all_data.csv")
all_data.head(10)
```

Out[95]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
1	NaN	NaN	NaN	NaN	NaN	NaN
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001
6	176562	USB-C Charging Cable	1	11.95	04/29/19 13:03	381 Wilson St, San Francisco, CA 94016
7	176563	Bose SoundSport Headphones	1	99.99	04/02/19 07:46	668 Center St, Seattle, WA 98101
8	176564	USB-C Charging Cable	1	11.95	04/12/19 10:58	790 Ridge St, Atlanta, GA 30301
9	176565	Macbook Pro Laptop	1	1700	04/24/19 10:38	915 Willow St, San Francisco, CA 94016

Clean up data

Drop rows of NaN

In [96]:
```
nan_df = all_data[all_data.isna().any(axis=1)]
nan_df.head()
```

Out[96]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
1	NaN	NaN	NaN	NaN	NaN	NaN
356	NaN	NaN	NaN	NaN	NaN	NaN
735	NaN	NaN	NaN	NaN	NaN	NaN
1433	NaN	NaN	NaN	NaN	NaN	NaN
1553	NaN	NaN	NaN	NaN	NaN	NaN

In [97]:
```
all_data = all_data.dropna(how='all')
all_data.head()
```

Out[97]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001

Find 'Or' and delete it

In [98]:
```
#temp_df = all_data[all_data[condition]]
all_data = all_data[all_data['Order Date'].str[0:2] != 'Or']
```

To convert column with the correct type

In [99]:
```
# all_data['Quantity Ordered'] = Make 'int'
all_data['Quantity Ordered'] = pd.to_numeric(all_data['Quantity Ordered'])

# all_data['Price Each'] = Make 'float'
all_data['Price Each'] = pd.to_numeric(all_data['Price Each'])
```

In [ ]:

Questions 1. What was the best month for sales? How much was earned that month?

- To add new column for knowing month's number

In [100…
```
all_data['Month'] = all_data['Order Date'].str[0:2]
all_data['Month'] = all_data['Month'].astype('int32')
all_data.head()
```

Out[100…

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4

- To add a sales column

In [101…
```
all_data['Sales'] = all_data['Quantity Ordered'] * all_data['Price Each']
all_data.head()
```

Out[101…

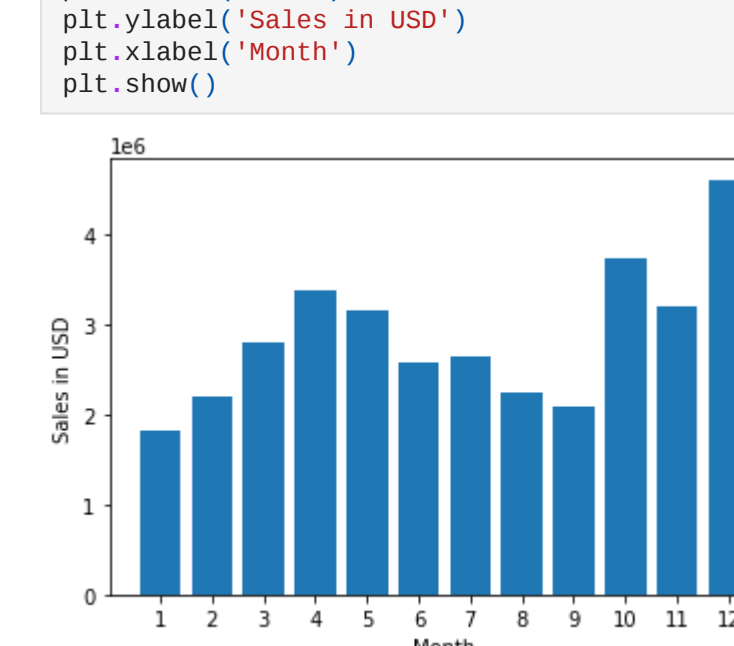
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99

- Sum per Month

In [102…
```
results = all_data.groupby('Month').sum()
```

- Show up with matployplib.pyplot

In [103…
```
months = range(1, 13)
plt.bar(months, results['Sales'])
plt.xticks(months)
plt.ylabel('Sales in USD')
plt.xlabel('Month')
plt.show()
```



Answer 1:

- The best month for sales was december.
- It earned at least 4 million dolars

Question 2. What city sold the most product?

- To add a City column

- To use .apply() for extracting caracteres

In [104…
```
def get_city(address):
    return address.split(',')[1]

def get_state(address):
    return address.split(',')[2].split(" ")[1]
all_data['City'] = all_data['Purchase Address'].apply(lambda x: f'{get_city(x)}  ({get_state(x)})')
all_data.head()
```

Out[104…

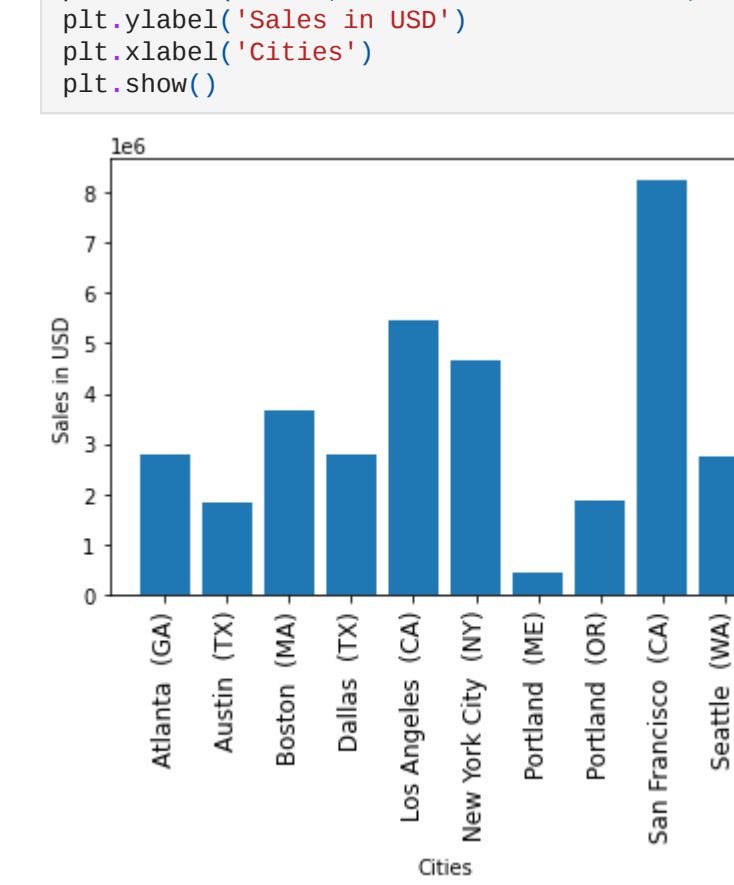
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas (TX)
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston (MA)
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)

In [105…
```
results = all_data.groupby('City').sum()
results
```

Out[105…

City	Quantity Ordered	Price Each	Month	Sales
Atlanta (GA)	16602	2.779908e+06	104794	2.795499e+06
Austin (TX)	11153	1.809874e+06	69829	1.819582e+06
Boston (MA)	22528	3.637410e+06	141112	3.661642e+06
Dallas (TX)	16730	2.752628e+06	104620	2.767975e+06
Los Angeles (CA)	33289	5.421435e+06	208325	5.452571e+06
New York City (NY)	27932	4.635371e+06	175741	4.664317e+06
Portland (ME)	2750	4.471893e+05	17144	4.497583e+05
Portland (OR)	11303	1.860558e+06	70621	1.870732e+06
San Francisco (CA)	50239	8.211462e+06	315520	8.262204e+06
Seattle (WA)	16553	2.733296e+06	104941	2.747755e+06

In [106…
```
cities = [city for city, df, in all_data.groupby('City')]

plt.bar(cities, results['Sales'])
plt.xticks(cities, rotation = 'vertical', size = 12)
plt.ylabel('Sales in USD')
plt.xlabel('Cities')
plt.show()
```



Answer 2:

- The city that most sold products was San Francisto (CA)

Question 3. What time should we display advertisements to maximize the likelihood of a customer’s buying a product?

- To convert Order Date using datetime

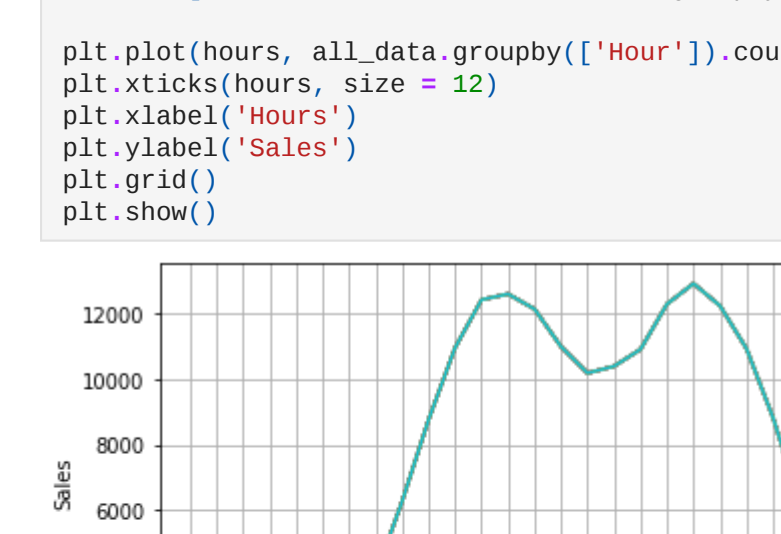
In [107…
```
all_data['Order Date'] = pd.to_datetime(all_data['Order Date'])
```

In [108…
```
all_data['Hour'] = all_data['Order Date'].dt.hour
all_data['Minute'] = all_data['Order Date'].dt.minute
all_data.head()
```

Out[108…

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Minute
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas (TX)	8	46
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston (MA)	22	30
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)	14	38
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	14	38
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	9	27

In [109…
```
hours = [hour for hour, df in all_data.groupby('Hour')]

plt.plot(hours, all_data.groupby(['Hour']).count())
plt.xticks(hours, size = 12)
plt.xlabel('Hours')
plt.ylabel('Sales')
plt.grid()
plt.show()
```



- The best hours for advertising to maximize the likelihood of a customer's buying a product are 12:00 and 19:00.

Question 4. What products are most often sold together?

In [110…
```
all_data.head()
```

Out[110…

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Minute
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas (TX)	8	46
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston (MA)	22	30
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)	14	38
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	14	38
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	9	27

- As you can see, the Order ID is repeated in some cases. For example, Google Phone (row number 3) and Wired Headphones (row number 4) have the same code. So, someone made the order with those products at the same time.

In [111…
```
# https://stackoverflow.com/questions/43348194/pandas-select-rows-if-id-appear-several-time

df = all_data[all_data['Order ID'].duplicated(keep=False)]
df.head(20)
```

Out[111…

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Minute
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)	14	38
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	14	38
18	176574	Google Phone	1	600.00	2019-04-03 19:42:00	20 Hill St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)	19	42
19	176574	USB-C Charging Cable	1	11.95	2019-04-03 19:42:00	20 Hill St, Los Angeles, CA 90001	4	11.95	Los Angeles (CA)	19	42
30	176585	Bose SoundSport Headphones	1	99.99	2019-04-07 11:31:00	823 Highland St, Boston, MA 02215	4	99.99	Boston (MA)	11	31
31	176585	Bose SoundSport Headphones	1	99.99	2019-04-07 11:31:00	823 Highland St, Boston, MA 02215	4	99.99	Boston (MA)	11	31
32	176586	AAA Batteries (4-pack)	2	2.99	2019-04-10 17:00:00	365 Center St, San Francisco, CA 94016	4	5.98	San Francisco (CA)	17	0
33	176586	Google Phone	1	600.00	2019-04-10 17:00:00	365 Center St, San Francisco, CA 94016	4	600.00	San Francisco (CA)	17	0
119	176672	Lightning Charging Cable	1	14.95	2019-04-12 11:07:00	778 Maple St, New York City, NY 10001	4	14.95	New York City (NY)	11	7
120	176672	USB-C Charging Cable	1	11.95	2019-04-12 11:07:00	778 Maple St, New York City, NY 10001	4	11.95	New York City (NY)	11	7
129	176681	Apple Airpods Headphones	1	150.00	2019-04-20 10:39:00	331 Cherry St, Seattle, WA 98101	4	150.00	Seattle (WA)	10	39
130	176681	ThinkPad Laptop	1	999.99	2019-04-20 10:39:00	331 Cherry St, Seattle, WA 98101	4	999.99	Seattle (WA)	10	39
138	176689	Bose SoundSport Headphones	1	99.99	2019-04-24 17:15:00	659 Lincoln St, New York City, NY 10001	4	99.99	New York City (NY)	17	15
139	176689	AAA Batteries (4-pack)	2	2.99	2019-04-24 17:15:00	659 Lincoln St, New York City, NY 10001	4	5.98	New York City (NY)	17	15
189	176739	34in Ultrawide Monitor	1	379.99	2019-04-05 17:38:00	730 6th St, Austin, TX 73301	4	379.99	Austin (TX)	17	38
190	176739	Google Phone	1	600.00	2019-04-05 17:38:00	730 6th St, Austin, TX 73301	4	600.00	Austin (TX)	17	38
225	176774	Lightning Charging Cable	1	14.95	2019-04-25 15:06:00	372 Church St, Los Angeles, CA 90001	4	14.95	Los Angeles (CA)	15	6
226	176774	USB-C Charging Cable	1	11.95	2019-04-25 15:06:00	372 Church St, Los Angeles, CA 90001	4	11.95	Los Angeles (CA)	15	6
233	176781	iPhone	1	700.00	2019-04-03 07:37:00	976 Hickory St, Dallas, TX 75001	4	700.00	Dallas (TX)	7	37
234	176781	Lightning Charging Cable	1	14.95	2019-04-03 07:37:00	976 Hickory St, Dallas, TX 75001	4	14.95	Dallas (TX)	7	37

- Check out the DataFrame. We have only the Order ID that is repeated.

In [112…
```
# https://stackoverflow.com/questions/27298178/concatenate-strings-from-several-rows-using-pandas-groupby

df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
df2 = df[['Order ID', 'Grouped']].drop_duplicates()
df2.head()
```

```
<ipython-input-112-5d4ac7236136>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Grouped'] = df.groupby('Order ID')['Product'].transform(lambda x: ','.join(x))
```

Out[112…

	Order ID	Grouped
3	176560	Google Phone,Wired Headphones
18	176574	Google Phone,USB-C Charging Cable
30	176585	Bose SoundSport Headphones,Bose SoundSport Hea...
32	176586	AAA Batteries (4-pack),Google Phone
119	176672	Lightning Charging Cable,USB-C Charging Cable

In [113…
```
# Referenced: https://stackoverflow.com/questions/52195887/counting-unique-pairs-of-numbers-into-a-python-dictionary
from itertools import combinations
from collections import Counter

count = Counter()

for row in df2['Grouped']:
    row_list = row.split(',')
    count.update(Counter(combinations(row_list, 2)))

for key,value in count.most_common(10):
    print(key, value)
```

```
('iPhone', 'Lightning Charging Cable') 1005
('Google Phone', 'USB-C Charging Cable') 987
('iPhone', 'Wired Headphones') 447
('Google Phone', 'Wired Headphones') 414
('Vareebadd Phone', 'USB-C Charging Cable') 361
('iPhone', 'Apple Airpods Headphones') 360
('Google Phone', 'Bose SoundSport Headphones') 220
('USB-C Charging Cable', 'Wired Headphones') 160
('Vareebadd Phone', 'Wired Headphones') 143
('Lightning Charging Cable', 'Wired Headphones') 92
```

Question 5. What product sold the most? Why do you think it sold the most?

In [114…
```
all_data.head()
```

Out[114…

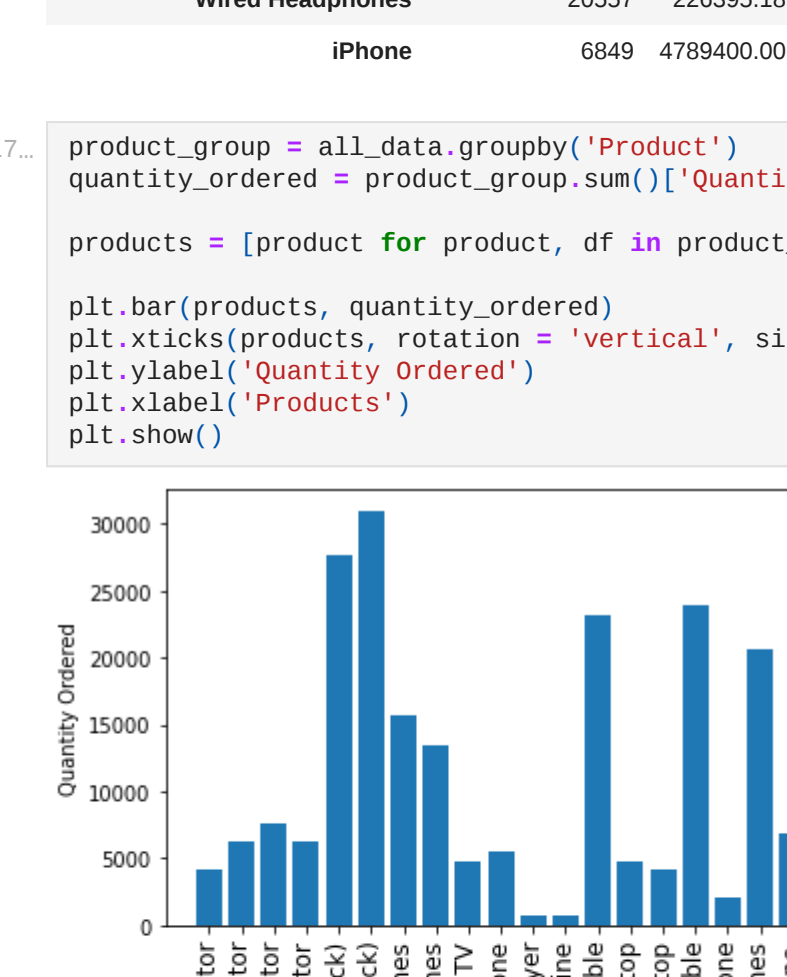
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales	City	Hour	Minute
0	176558	USB-C Charging Cable	2	11.95	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	4	23.90	Dallas (TX)	8	46
2	176559	Bose SoundSport Headphones	1	99.99	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	4	99.99	Boston (MA)	22	30
3	176560	Google Phone	1	600.00	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles (CA)	14	38
4	176560	Wired Headphones	1	11.99	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	14	38
5	176561	Wired Headphones	1	11.99	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles (CA)	9	27

In [115…
```
product_group = all_data.groupby('Product')
product_group.sum()
```

Out[115…

Product	Quantity Ordered	Price Each	Month	Sales	Hour	Minute
20in Monitor	4129	451068.99	29336	454148.71	58764	122252
27in 4K Gaming Monitor	6244	2429637.70	44440	2435097.56	90916	184331
27in FHD Monitor	7550	1125974.93	52558	1132424.50	107540	219948
34in Ultrawide Monitor	6199	2348718.19	43304	2355558.01	89076	183480
AA Batteries (4-pack)	27635	79015.68	145558	106118.40	298342	609039
AAA Batteries (4-pack)	31017	61716.59	146370	92740.83	297332	612113
Apple Airpods Headphones	15661	2332350.00	109477	2349150.00	223304	455570
Bose SoundSport Headphones	13457	1332366.75	94113	1345565.43	192445	392603
Flatscreen TV	4819	1440000.00	34224	1445700.00	68815	142789
Google Phone	5532	3315000.00	38305	3319200.00	79479	162773
LG Dryer	646	387600.00	4383	387600.00	9326	19043
LG Washing Machine	666	399600.00	4523	399600.00	9785	19462
Lightning Charging Cable	23217	323787.10	153092	347094.15	312529	634442
Macbook Pro Laptop	4728	8030800.00	33548	8037600.00	68261	137574
ThinkPad Laptop	4130	4127958.72	28950	4129958.70	59746	121508
USB-C Charging Cable	23975	261740.85	154819	286501.25	314645	647586
Vareebadd Phone	2068	826000.00	14309	827200.00	29472	61835
Wired Headphones	20557	226395.18	133397	246478.43	271720	554023
iPhone	6849	4789400.00	47941	4794300.00	98657	201688

In [117…
```
product_group = all_data.groupby('Product')
quantity_ordered = product_group.sum()['Quantity Ordered']

products = [product for product, df in product_group]

plt.bar(products, quantity_ordered)
plt.xticks(products, rotation = 'vertical', size = 12)
plt.ylabel('Quantity Ordered')
plt.xlabel('Products')
plt.show()
```

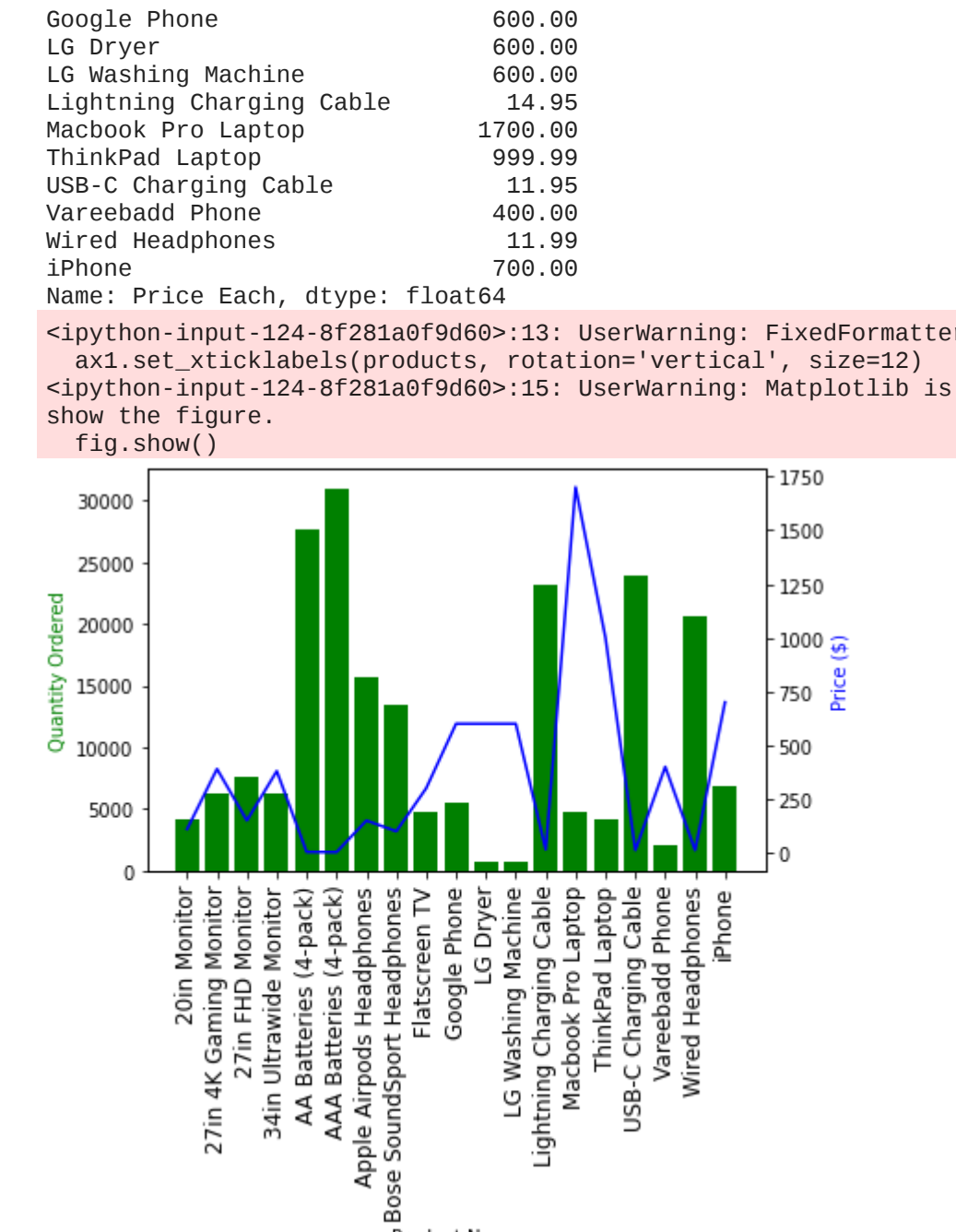


- We are going to do a deeper analysis about quantity oredered using statistics.

In [124…
```
prices = all_data.groupby('Product').mean()['Price Each']
print(prices)

fig, ax1 = plt.subplots()

ax2 = ax1.twinx()
ax1.bar(products, quantity_ordered, color='g')
ax2.plot(products, prices, color='b')

ax1.set_xlabel('Product Name')
ax1.set_ylabel('Quantity Ordered', color='g')
ax2.set_ylabel('Price ($)', color='b')
ax1.set_xticklabels(products, rotation='vertical', size=12)

fig.show()
```

```
Product
20in Monitor                  109.99
27in 4K Gaming Monitor        389.99
27in FHD Monitor              149.99
34in Ultrawide Monitor        379.99
AA Batteries (4-pack)           3.84
AAA Batteries (4-pack)          2.99
Apple Airpods Headphones      150.00
Bose SoundSport Headphones     99.99
Flatscreen TV                 300.00
Google Phone                  600.00
LG Dryer                      600.00
LG Washing Machine            600.00
Lightning Charging Cable       14.95
Macbook Pro Laptop           1700.00
ThinkPad Laptop               999.99
USB-C Charging Cable           11.95
Vareebadd Phone               400.00
Wired Headphones               11.99
iPhone                        700.00
Name: Price Each, dtype: float64
```

```
<ipython-input-124-8f281a0f9d60>:13: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax1.set_xticklabels(products, rotation='vertical', size=12)
<ipython-input-124-8f281a0f9d60>:15: UserWarning: Matplotlib is currently using module://ipykernel.pylab.backend_inline, which is a non-GUI backend, so cannot show the figure.
  fig.show()
```



- Line blue contains the price and the green bars are the quantity ordered for each product. As we can analyze the graphic, if the amount of product is less, the price increase and vice versa. For example, we can think that a lot of people are buying a MacBook Pro, in this case, the Mac is in demand, and for that reason its price increase much more than others products less demanded.

In [ ]:

In [ ]: