Title of the project:

Data Analytics Project- Sales Data Analysis using Python

Project short description:

Hello everyone!

In this tutorial, we are going to analyse the sales data generated by the e-commerce sites such as Amazon, Flipkart, Myntra, etc., to answer the following questions:

Q1) Which month is best for selling products?

Q2) Which city orders the highest number of products?

Q3) What time of the day do people order the most goods online?

Q4) Which product has the highest demand and why?

By answering these questions, we will able to increase the selling rate to make lots of profit and also to expand the business.

Requirements:

1) You need to have the dataset files with a .csv extension.
2) Install Jupyter Notebook or any similar working environment with the latest version of Python installed.
3) You must know the Python language and its libraries which includes NumPy, pandas, Matplotlib, and os.

About the dataset:

We have the sales data of each month which includes the complete details of the ordered product.

Step by step implementation:

1) First of all, import all the required libraries

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import os
```

2) Merge the sales data of each month into a single .csv file

```
In [2]: os.listdir('E:\[illegible]\DS\Sales Data Analysis\Dataset') #Here, Enter the location of folder containing the dataset files.

Out[2]: ['Sales_April_2019.csv',
         'Sales_August_2019.csv',
         'Sales_December_2019.csv',
         'Sales_February_2019.csv',
         'Sales_January_2019.csv',
         'Sales_July_2019.csv',
         'Sales_June_2019.csv',
         'Sales_March_2019.csv',
         'Sales_May_2019.csv',
         'Sales_November_2019.csv',
         'Sales_October_2019.csv',
         'Sales_September_2019.csv']
```

Here, os.listdir() returns a list containing the list of all files and directories in the specified directory.

In [3]:
```
Files=[]
for i in os.listdir('E:\\DS\Sales Data Analysis\Dataset') : #Here, Enter the location of folder containing the dataset files.
    Files.append(i)
for i in Files :
    print(i)
```

```
Sales_April_2019.csv
Sales_August_2019.csv
Sales_December_2019.csv
Sales_February_2019.csv
Sales_January_2019.csv
Sales_July_2019.csv
Sales_June_2019.csv
Sales_March_2019.csv
Sales_May_2019.csv
Sales_November_2019.csv
Sales_October_2019.csv
Sales_September_2019.csv
```

Here, we are making a list of all the dataset files.

In [4]:
```
allData=pd.DataFrame() #Empty DataFrame

for i in Files :
    current_df=pd.read_csv('E:/DS/Sales Data Analysis/Dataset/'+i )
    allData=pd.concat([allData,current_df])

print(allData.shape)
```

```
(186850, 6)
```

The pd.concat() function does all of the heavy liftings of performing concatenation operations along an axis. Here, we are concatenating all the .csv into one DataFrame i.e., allData.

In [5]:
```
allData.to_csv('E:/DS/Sales Data Analysis/Dataset/allData.csv',index=False)
```

In [6]:
```
allData.head()
```

Out[6]:

	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

Now saving the Pandas DataFrame allData as a .csv file using to_csv() method in the specified directory.
Pandas head() method is used to return the top 5 rows of allData DataFrame.

3) Code to remove missing (Null/NaN) values in the Dataset.

In [7]:
```
allData.isnull().sum()
```

Out[7]:
```
Order ID            1635
Product             1635
Quantity Ordered    1635
Price Each          1635
Order Date          1635
Purchase Address    1635
dtype: int64
```

In [8]:
```
allData=allData.dropna()
```

In [9]:
```
allData.shape
```

Out[9]:
```
(558915, 6)
```

In [10]:
```
allData.head()
```

Out[10]:

	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

4) Q1) Which month is best for selling products?

In [11]:
```
def month(s) :
    s=s.split('/')
    return s[0]

#Working of month(s)
s='04/19/19 08:46'
month(s)
```

Out[11]:
```
'04'
```

In [12]:
```
allData['Month']=allData['Order Date'].apply(month)
```

In [13]:
```
allData.head()
```

Out[13]:

	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	04
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	04
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	04

In [14]:
```
allData.dtypes
```

Out[14]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
Month               object
dtype: object
```

In [15]:
```
allData['Month'].unique()
```

Out[15]:
```
array(['04', '05', 'Order Date', '08', '09', '12', '01', '02', '03', '07',
       '06', '11', '10'], dtype=object)
```

In [16]:
```
allData=allData[allData.Month!='Order Date']
```

In [17]:
```
allData['Month'].unique()
```

Out[17]:
```
array(['04', '05', '08', '09', '12', '01', '02', '03', '07', '06', '11',
       '10'], dtype=object)
```

In [18]:
```
allData['Month']=allData['Month'].astype(int)
```

In [19]:
```
allData.dtypes
```

Out[19]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
Month                int32
dtype: object
```

In [20]:
```
allData['Quantity Ordered']=allData['Quantity Ordered'].astype(int)
allData['Price Each']=allData['Price Each'].astype(float)
```

In [21]:
```
allData.dtypes
```

Out[21]:
```
Order ID             object
Product              object
Quantity Ordered      int32
Price Each          float64
Order Date           object
Purchase Address     object
Month                 int32
dtype: object
```

In [22]:
```
allData['Sale']=allData['Quantity Ordered']*allData['Price Each']
```

In [23]:
```
allData.head()
```

Out[23]:

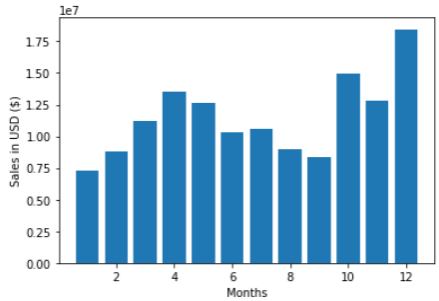
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sale
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	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

In [24]:
```
allData.groupby('Month')['Sale'].sum()
```

Out[24]:
```
Month
1     7.289027e+06
2     8.808090e+06
3     1.122840e+07
4     1.356268e+07
5     1.261043e+07
6     1.031121e+07
7     1.059110e+07
8     8.977872e+06
9     8.390241e+06
10    1.494691e+07
11    1.279841e+07
12    1.845377e+07
Name: Sale, dtype: float64
```

In [25]:
```
months=range(1,13)
plt.bar(months,allData.groupby('Month')['Sale'].sum())
plt.xlabel('Months')
plt.ylabel('Sales in USD ($)')
```

Out[25]:
```
Text(0, 0.5, 'Sales in USD ($)')
```



Therefore, the month of December is best for selling products.

5) Q2) Which city orders the highest number of products?

In [26]:
```
def city(s) :
    s=s.split(',')[1]
    return s

#Working of city(s)
s='669 Spruce St, Los Angeles, CA 90001'
city(s)
```

Out[26]:
```
' Los Angeles'
```

In [27]:
```
allData['City']=allData['Purchase Address'].apply(city)
```

In [28]:
```
allData.head()
```

Out[28]:

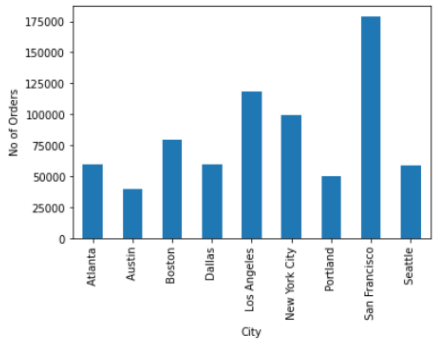
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sale	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560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

In [29]:
```
allData.groupby('City')['City'].count()
```

Out[29]:
```
City
 Atlanta          59524
 Austin           39620
 Boston           79736
 Dallas           59280
 Los Angeles     118420
 New York City    99504
 Portland         49860
 San Francisco   178928
 Seattle          58928
Name: City, dtype: int64
```

In [30]:
```
allData.groupby('City')['City'].count().plot.bar()
plt.ylabel('No of Orders')
```

Out[30]:
```
Text(0, 0.5, 'No of Orders')
```



Therefore, San Francisco orders the highest number of products.

6) Q3) What time of the day do people order the most goods online?

In [31]:
```
allData['Order Date'].dtype #String DataType
```

Out[31]:
```
dtype('O')
```

In [32]:
```
allData['Hour']=pd.to_datetime(allData['Order Date']).dt.hour
```

In [33]:
```
allData.head()
```

Out[33]:

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

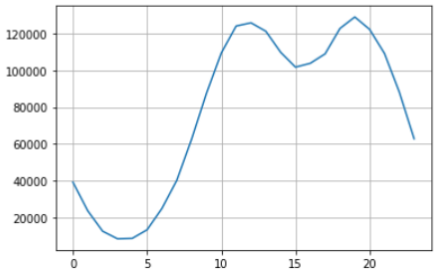
In [34]:
```
hour=[]
hour.sort()
for i in allData['Hour'] :
    hour.append(i)
hour_=list(set(hour))
print(hour_)

count_=[]
for i in range(24) :
    x=hour.count(hour_[i])
    count_.append(x)
print(count_)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23]
[39100, 23500, 12430, 8310, 8540, 13210, 24820, 40110, 62560, 87480, 109440, 124110, 125870, 121290, 109840, 101750, 103840, 10
8990, 122800, 129050, 122280, 109210, 88220, 62750]
```

In [35]:
```
plt.grid()
plt.plot(hour_,count_)
```

Out[35]:
```
[<matplotlib.lines.Line2D at 0x1f3bf39a7c0>]
```

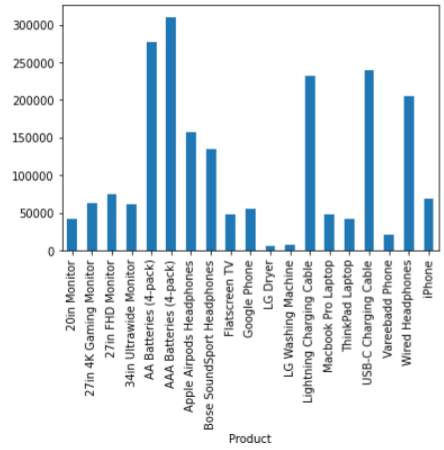


From the above plot, it can be concluded that during 10 AM to 1 PM and 6 PM to 8 PM, received a maximum number of orders and it is probably the best time to show advertisements to maximize the product selling.

7) Q4) Which product has the highest demand and why?

```
In [36]: allData.groupby('Product')['Quantity Ordered'].sum().plot.bar()
```

```
Out[36]: <AxesSubplot:xlabel='Product'>
```



It is clear from the above bar plot that AAA Batteries (4-pack) is the top-selling product.

```
In [37]: allData.groupby('Product')['Price Each'].mean()
```

```
Out[37]: 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
```

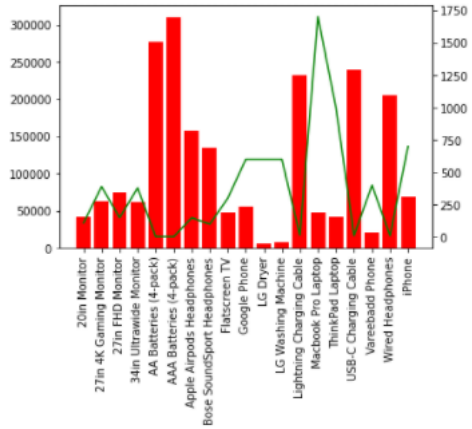
```
In [38]: products=allData.groupby('Product')['Quantity Ordered'].sum().index
         quantity=allData.groupby('Product')['Quantity Ordered'].sum()
         prices=allData.groupby('Product')['Price Each'].mean()
```

In [39]:
```
plt.figure(figsize=(50,30))
fig,ax1 = plt.subplots()
ax2=ax1.twinx()
ax1.bar(products, quantity, color='r')
ax2.plot(products, prices, 'g')
ax1.set_xticklabels(products, rotation='vertical', size=10)
```

```
<ipython-input-39-2f17c53ccfc8>:6: UserWarning: FixedFormatter should only be used together with FixedLocator
  ax1.set_xticklabels(products, rotation='vertical', size=10)
```

Out[39]:
```
[Text(0, 0, '20in Monitor'),
 Text(1, 0, '27in 4K Gaming Monitor'),
 Text(2, 0, '27in FHD Monitor'),
 Text(3, 0, '34in Ultrawide Monitor'),
 Text(4, 0, 'AA Batteries (4-pack)'),
 Text(5, 0, 'AAA Batteries (4-pack)'),
 Text(6, 0, 'Apple Airpods Headphones'),
 Text(7, 0, 'Bose SoundSport Headphones'),
 Text(8, 0, 'Flatscreen TV'),
 Text(9, 0, 'Google Phone'),
 Text(10, 0, 'LG Dryer'),
 Text(11, 0, 'LG Washing Machine'),
 Text(12, 0, 'Lightning Charging Cable'),
 Text(13, 0, 'Macbook Pro Laptop'),
 Text(14, 0, 'ThinkPad Laptop'),
 Text(15, 0, 'USB-C Charging Cable'),
 Text(16, 0, 'Vareebadd Phone'),
 Text(17, 0, 'Wired Headphones'),
 Text(18, 0, 'iPhone')]

<Figure size 3600x2160 with 0 Axes>
```



From the above plot, it can be concluded that the selling of a product depends on its price. The more expensive the product, the lower will be the quantity ordered and vice versa.