Aditi Rao EXTC 118A2088

Create a dataframe of ten rows, four columns with random values.

```
import numpy as np
import pandas as pd
```

```
df = pd.DataFrame(np.random.randn(10,4))
df
```

	0	1	2	3
0	1.437568	-0.967435	-1.134977	-0.459088
1	0.657142	-0.831060	-0.122493	-2.025464
2	-0.877594	0.885826	0.150817	0.319489
3	-0.471818	0.171775	-3.099107	0.714485
4	-0.329806	-1.224557	-2.547721	-1.450197
5	-0.846485	-0.145086	0.110895	1.159562
6	-0.884772	-1.546547	1.754197	-1.494740
7	-0.787167	0.291495	1.365775	0.874452
8	-0.383770	-1.019123	0.155924	0.443694

Create a pandas series from each of the items below: (i) a list, (ii) numpy and (iii) a dictionary

```
list = [1,2,3,4,5,6]
series=pd.Series(list)
print(series)
```

```
0    1
1    2
2    3
3    4
4    5
```

```
5    6
dtype: int64
```

```
X1 = np.array([2,4,6,8])
pd.Series(X1)
```

```
0    2
1    4
2    6
3    8
dtype: int64
```

```
X2 = {'a':1,'b':2,'c':3}
pd.Series(X2)
```

```
a    1
b    2
c    3
dtype: int64
```

Concatenate two data frames.

Create two data frames using the following two dictionaries.

GermanCars = {'Company': ['Ford', 'Mercedes', 'BMV', 'Audi'], 'Price': [23845, 171995, 135925 , 71400]}

japaneseCars = {'Company': ['Toyota', 'Honda', 'Nissan', 'Mitsubishi '], 'Price': [29995, 23600, 61500 , 58900]}

```
GermanCars = pd.DataFrame({'Company': ['Ford', 'Mercedes', 'BMV', 'Audi'], 'Price': [23845, 1

japaneseCars = pd.DataFrame({'Company': ['Toyota', 'Honda', 'Nissan', 'Mitsubishi '], 'Price'


pd.concat([GermanCars,japaneseCars], axis=0)
```

Consider the given “Automobile_data” dataset

- From the given dataset print the first and last five rows
- Clean the dataset and update the CSV file (i.e remove nan)
- Write a Pandas program to detect missing values of a given DataFrame. Display True or False.
- Write a Pandas program to drop the columns where at least one element is missing in a given DataFrame.
- Write a Pandas program to count the number of missing values in each column of a given DataFrame.

```
from google.colab import files
uploaded = files.upload()
```

Choose Files Automobile_data.csv

- **Automobile_data.csv**(application/vnd.ms-excel) - 3218 bytes, last modified: 6/22/2021 - 100% done

```
Saving Automobile_data.csv to Automobile_data.csv
```

```
import io
df2 = pd.read_csv(io.BytesIO(uploaded['Automobile_data.csv']))
print(df2)
```

```
    index      company   body-style  ...  horsepower  average-mileage    price
0       0  alfa-romero  convertible  ...       111.0               21  13495.0
1       1  alfa-romero  convertible  ...       111.0               21  16500.0
2       2  alfa-romero    hatchback  ...       154.0               19  16500.0
3       3         audi        sedan  ...       102.0               24  13950.0
4       4         audi        sedan  ...       115.0               18  17450.0
..    ...          ...          ...  ...         ...              ...      ...
56     81   volkswagen        sedan  ...        85.0               27   7975.0
57     82   volkswagen        sedan  ...        52.0               37   7995.0
58     86   volkswagen        sedan  ...       100.0               26   9995.0
59     87        volvo        sedan  ...       114.0               23  12940.0
60     88        volvo        wagon  ...       114.0               23  13415.0

[61 rows x 10 columns]
```

```
df2.head()
```

index company body- wheel- length engine- num-of- horsepower average-

```
df2.tail()
```

	index	company	body-style	wheel-base	length	engine-type	num-of-cylinders	horsepower	average-mileage
56	81	volkswagen	sedan	97.3	171.7	ohc	four	85.0	27
57	82	volkswagen	sedan	97.3	171.7	ohc	four	52.0	37
58	86	volkswagen	sedan	97.3	171.7	ohc	four	100.0	26
59	87	volvo	sedan	104.3	188.8	ohc	four	114.0	23
60	88	volvo	wagon	104.3	188.8	ohc	four	114.0	23

```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61 entries, 0 to 60
Data columns (total 10 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   index             61 non-null     int64  
 1   company           61 non-null     object 
 2   body-style        61 non-null     object 
 3   wheel-base        59 non-null     float64
 4   length            61 non-null     float64
 5   engine-type       61 non-null     object 
 6   num-of-cylinders  61 non-null     object 
 7   horsepower        59 non-null     float64
 8   average-mileage   61 non-null     int64  
 9   price             58 non-null     float64
dtypes: float64(4), int64(2), object(4)
memory usage: 4.9+ KB
```

```
df2.dropna(axis=0)
```

	index	company	body-style	wheel-base	length	engine-type	num-of-cylinders	horsepower	averag milea
0	0	alfa-romero	convertible	88.6	168.8	dohc	four	111.0	
1	1	alfa-romero	convertible	88.6	168.8	dohc	four	111.0	
2	2	alfa-romero	hatchback	94.5	171.2	ohcv	six	154.0	
3	3	audi	sedan	99.8	176.6	ohc	four	102.0	
4	4	audi	sedan	99.4	176.6	ohc	five	115.0	
5	5	audi	sedan	99.8	177.3	ohc	five	110.0	
6	6	audi	wagon	105.8	192.7	ohc	five	110.0	
7	9	bmw	sedan	101.2	176.8	ohc	four	101.0	
8	10	bmw	sedan	101.2	176.8	ohc	four	101.0	
9	11	bmw	sedan	101.2	176.8	ohc	six	121.0	
11	14	bmw	sedan	103.5	193.8	ohc	six	182.0	
12	15	bmw	sedan	110.0	197.0	ohc	six	182.0	
14	17	chevrolet	hatchback	94.5	155.9	ohc	four	70.0	
15	18	chevrolet	sedan	94.5	158.8	ohc	four	70.0	
16	19	dodge	hatchback	93.7	157.3	ohc	four	68.0	
17	20	dodge	hatchback	93.7	157.3	ohc	four	68.0	
18	27	honda	wagon	96.5	157.1	ohc	four	76.0	
19	28	honda	sedan	96.5	175.4	ohc	four	101.0	
20	29	honda	sedan	96.5	169.1	ohc	four	100.0	
24	33	jaguar	sedan	113.0	199.6	dohc	six	176.0	
25	34	jaguar	sedan	113.0	199.6	dohc	six	176.0	
26	35	jaguar	sedan	102.0	191.7	ohcv	twelve	262.0	
27	36	mazda	hatchback	93.1	159.1	ohc	four	68.0	
28	37	mazda	hatchback	93.1	159.1	ohc	four	68.0	
29	38	mazda	hatchback	93.1	159.1	ohc	four	68.0	
30	39	mazda	hatchback	95.3	169.0	rotor	two	101.0	
31	43	mazda	sedan	104.9	175.0	ohc	four	72.0	
32	44	mercedes-benz	sedan	110.0	190.9	ohc	five	123.0	
33	45	mercedes-benz	wagon	110.0	190.9	ohc	five	123.0	

34	46	mercedes-benz	sedan	120.9	208.1	ohcv	eight	184.0
35	47	mercedes-benz	hardtop	112.0	199.2	ohcv	eight	184.0
36	49	mitsubishi	hatchback	93.7	157.3	ohc	four	68.0
37	50	mitsubishi	hatchback	93.7	157.3	ohc	four	68.0
38	51	mitsubishi	sedan	96.3	172.4	ohc	four	88.0
39	52	mitsubishi	sedan	96.3	172.4	ohc	four	88.0
40	53	nissan	sedan	94.5	165.3	ohc	four	55.0
41	54	nissan	sedan	94.5	165.3	ohc	four	69.0
42	55	nissan	sedan	94.5	165.3	ohc	four	69.0
44	57	nissan	sedan	100.4	184.6	ohcv	six	152.0
45	61	porsche	hardtop	89.5	168.9	ohcf	six	207.0
46	62	porsche	convertible	89.5	168.9	ohcf	six	207.0
48	66	toyota	hatchback	95.7	158.7	ohc	four	62.0
49	67	toyota	hatchback	95.7	158.7	ohc	four	62.0

```
df2.isnull()
```

	index	company	body-style	wheel-base	length	engine-type	num-of-cylinders	horsepower	average-mileage	pri
0	False	False	False	False	False	False	False	False	False	Fa
1	False	False	False	False	False	False	False	False	False	Fa
2	False	False	False	False	False	False	False	False	False	Fa
3	False	False	False	False	False	False	False	False	False	Fa
4	False	False	False	False	False	False	False	False	False	Fa
5	False	False	False	False	False	False	False	False	False	Fa
6	False	False	False	False	False	False	False	False	False	Fa
7	False	False	False	False	False	False	False	False	False	Fa
8	False	False	False	False	False	False	False	False	False	Fa
9	False	False	False	False	False	False	False	False	False	Fa
10	False	False	False	True	False	False	False	False	False	Fa
11	False	False	False	False	False	False	False	False	False	Fa
12	False	False	False	False	False	False	False	False	False	Fa
13	False	False	False	False	False	False	False	True	False	Fa
14	False	False	False	False	False	False	False	False	False	Fa
15	False	False	False	False	False	False	False	False	False	Fa
16	False	False	False	False	False	False	False	False	False	Fa
17	False	False	False	False	False	False	False	False	False	Fa
18	False	False	False	False	False	False	False	False	False	Fa
19	False	False	False	False	False	False	False	False	False	Fa
20	False	False	False	False	False	False	False	False	False	Fa
21	False	False	False	True	False	False	False	False	False	Fa
22	False	False	False	False	False	False	False	False	False	Tı
23	False	False	False	False	False	False	False	False	False	Tı
24	False	False	False	False	False	False	False	False	False	Fa
25	False	False	False	False	False	False	False	False	False	Fa
26	False	False	False	False	False	False	False	False	False	Fa
27	False	False	False	False	False	False	False	False	False	Fa
28	False	False	False	False	False	False	False	False	False	Fa
29	False	False	False	False	False	False	False	False	False	Fa

30	False	False	False	False	False	False	False	False	False	Fa
31	False	False	False	False	False	False	False	False	False	Fa
32	False	False	False	False	False	False	False	False	False	Fa
33	False	False	False	False	False	False	False	False	False	Fa
34	False	False	False	False	False	False	False	False	False	Fa
35	False	False	False	False	False	False	False	False	False	Fa
36	False	False	False	False	False	False	False	False	False	Fa
37	False	False	False	False	False	False	False	False	False	Fa
38	False	False	False	False	False	False	False	False	False	Fa
39	False	False	False	False	False	False	False	False	False	Fa
40	False	False	False	False	False	False	False	False	False	Fa
41	False	False	False	False	False	False	False	False	False	Fa
42	False	False	False	False	False	False	False	False	False	Fa
43	False	False	False	False	False	False	False	True	False	Fa
44	False	False	False	False	False	False	False	False	False	Fa
45	False	False	False	False	False	False	False	False	False	Fa

```
df2.dropna(axis=1)
```

	index	company	body-style	length	engine-type	num-of-cylinders	average-mileage
0	0	alfa-romero	convertible	168.8	dohc	four	21
1	1	alfa-romero	convertible	168.8	dohc	four	21
2	2	alfa-romero	hatchback	171.2	ohcv	six	19
3	3	audi	sedan	176.6	ohc	four	24
4	4	audi	sedan	176.6	ohc	five	18
5	5	audi	sedan	177.3	ohc	five	19
6	6	audi	wagon	192.7	ohc	five	19
7	9	bmw	sedan	176.8	ohc	four	23
8	10	bmw	sedan	176.8	ohc	four	23
9	11	bmw	sedan	176.8	ohc	six	21
10	13	bmw	sedan	189.0	ohc	six	16
11	14	bmw	sedan	193.8	ohc	six	16
12	15	bmw	sedan	197.0	ohc	six	15
13	16	chevrolet	hatchback	141.1	ohc	three	47
14	17	chevrolet	hatchback	155.9	ohc	four	38
15	18	chevrolet	sedan	158.8	ohc	four	38
16	19	dodge	hatchback	157.3	ohc	four	31
17	20	dodge	hatchback	157.3	ohc	four	31
18	27	honda	wagon	157.1	ohc	four	30
19	28	honda	sedan	175.4	ohc	four	24
20	29	honda	sedan	169.1	ohc	four	25
21	30	isuzu	sedan	170.7	ohc	four	24
22	31	isuzu	sedan	155.9	ohc	four	38
23	32	isuzu	sedan	155.9	ohc	four	38
24	33	jaguar	sedan	199.6	dohc	six	15
25	34	jaguar	sedan	199.6	dohc	six	15
26	35	jaguar	sedan	191.7	ohcv	twelve	13
27	36	mazda	hatchback	159.1	ohc	four	30
28	37	mazda	hatchback	159.1	ohc	four	31
29	38	mazda	hatchback	159.1	ohc	four	31

30	39	mazda	hatchback	169.0	rotor	two	17
31	43	mazda	sedan	175.0	ohc	four	31
32	44	mercedes-benz	sedan	190.9	ohc	five	22
33	45	mercedes-benz	wagon	190.9	ohc	five	22
34	46	mercedes-benz	sedan	208.1	ohcv	eight	14
35	47	mercedes-benz	hardtop	199.2	ohcv	eight	14
36	49	mitsubishi	hatchback	157.3	ohc	four	37
37	50	mitsubishi	hatchback	157.3	ohc	four	31
38	51	mitsubishi	sedan	172.4	ohc	four	25
39	52	mitsubishi	sedan	172.4	ohc	four	25
40	53	nissan	sedan	165.3	ohc	four	45
41	54	nissan	sedan	165.3	ohc	four	31
42	55	nissan	sedan	165.3	ohc	four	31
43	56	nissan	wagon	170.2	ohc	four	31

For the given “company_sales_data” dataset

1. Get total profit of all months and show line plot with the following Style properties. Generated line plot must include following Style properties
    - Line Style dotted and Line-color should be red
    - Show legend at the lower right location.
    - X label name = Month Number
    - Y label name = Sold units number
    - Add a circle marker.
    - Line marker color as read
    - Line width should be 3
2. Read all product sales data and show it using a multiline plot. Display the number of units sold per month for each product using multiline plots. (i.e., Separate Plotline for each product).
3. Read toothpaste sales data of each month and show it using a scatter plot
4. Read the total profit of each month and show it using the histogram to see the most common profit ranges

5. Calculate total sale data for last year for each product and show it using a Pie chart.
6. Read Bathing soap facewash of all months and display it using the Subplot

```
from google.colab import files
uploaded = files.upload()
```

Choose Files company_sales_data.csv

- **company_sales_data.csv**(application/vnd.ms-excel) - 659 bytes, last modified: 6/22/2021 - 100% done

```
Saving company_sales_data.csv to company_sales_data (1).csv
```

```
import io
df3 = pd.read_csv(io.BytesIO(uploaded['company_sales_data.csv']))
print(df3)
```

```
    month_number  facecream  facewash  toothpaste  bathingsoap  shampoo  \
0              1       2500      1500        5200         9200     1200
1              2       2630      1200        5100         6100     2100
2              3       2140      1340        4550         9550     3550
3              4       3400      1130        5870         8870     1870
4              5       3600      1740        4560         7760     1560
5              6       2760      1555        4890         7490     1890
6              7       2980      1120        4780         8980     1780
7              8       3700      1400        5860         9960     2860
8              9       3540      1780        6100         8100     2100
9             10       1990      1890        8300        10300     2300
10            11       2340      2100        7300        13300     2400
11            12       2900      1760        7400        14400     1800

    moisturizer  total_units  total_profit
0          1500        21100        211000
1          1200        18330        183300
2          1340        22470        224700
3          1130        22270        222700
4          1740        20960        209600
5          1555        20140        201400
6          1120        29550        295500
7          1400        36140        361400
8          1780        23400        234000
9          1890        26670        266700
10         2100        41280        412800
11         1760        30020        300200
```

```
import matplotlib.pyplot as plt
month=df3['month_number']
profit=df3['total_profit']
plt.plot(df3['total_profit'],label="Profit data of last month", linestyle="--",linewidth=3, c
plt.xlabel('Month Number')      # Set the x axis label of the current axis.
plt.ylabel('Sold units number')         # Set the y axis label of the current axis.
plt.title('Company sales of last year')
plt.legend(loc='lower right')
```

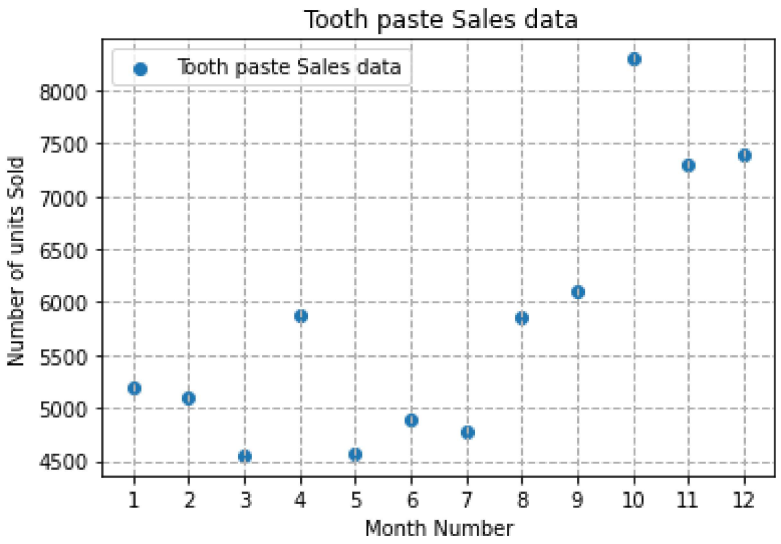
```
<matplotlib.legend.Legend at 0x7f37e70d1550>
```


```
toothpaste1 = df3['toothpaste'].tolist()
fashwash1 = df3['facewash'].tolist()
facecream1 = df3['facecream'].tolist()
bathingsoap1 = df3['bathingsoap'].tolist()
shampoo1 = df3['shampoo'].tolist()
moisturizer1 = df3['moisturizer'].tolist()

plt.plot(month,toothpaste1,label="Profit data of last month", color="red",marker="o", markerf
plt.plot(month, fashwash1,label="Profit data of last month", color="orange",marker="o", marke
plt.plot(month, facecream1,label="Profit data of last month", color="blue",marker="o", marker
plt.plot(month, bathingsoap1,label="Profit data of last month", color="purple",marker="o", ma
plt.plot(month, shampoo1,label="Profit data of last month", color="brown",marker="o", markerf
plt.plot(month, moisturizer1,label="Profit data of last month", color="yellow",marker="o", ma

plt.xlabel('Month Number')      # Set the x axis label of the current axis.
plt.ylabel('Sold units number')         # Set the y axis label of the current axis.
plt.title('Sales data')
plt.legend(loc='lower right')
```

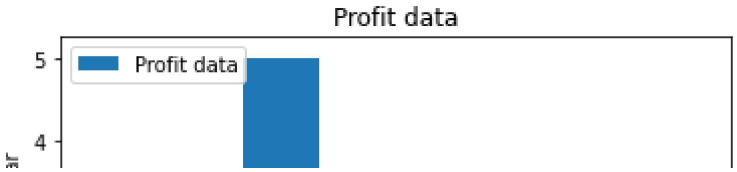
```
<matplotlib.legend.Legend at 0x7f37e6e95690>
```

Sales data

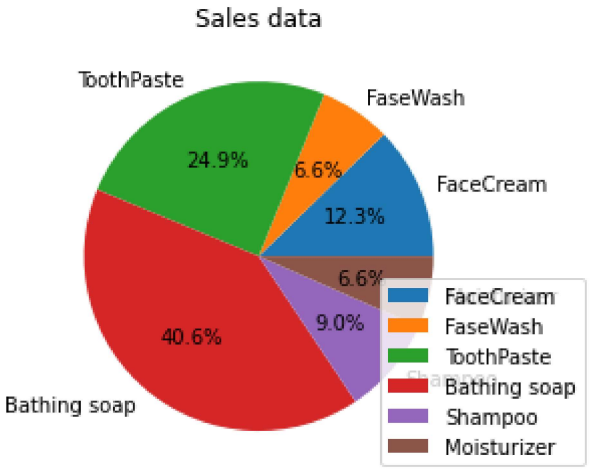
```
monthList  = df3 ['month_number'].tolist()
toothPasteSalesData = df3 ['toothpaste'].tolist()
plt.scatter(monthList, toothPasteSalesData, label = 'Tooth paste Sales data')
plt.xlabel('Month Number')
plt.ylabel('Number of units Sold')
plt.legend(loc='upper left')
plt.title(' Tooth paste Sales data')
plt.xticks(monthList)
plt.grid(True, linewidth= 1, linestyle="--")
plt.show()
```



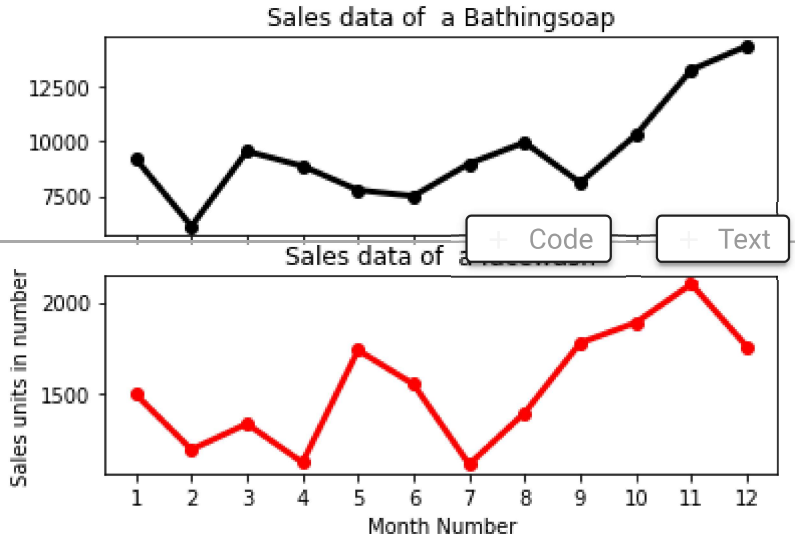

```
profitList = df3 ['total_profit'].tolist()
labels = ['low', 'average', 'Good', 'Best']
profit_range = [150000, 175000, 200000, 225000, 250000, 300000, 350000]
plt.hist(profitList, profit_range, label = 'Profit data')
plt.xlabel('profit range in dollar')
plt.ylabel('Actual Profit in dollar')
plt.legend(loc='upper left')
plt.xticks(profit_range)
plt.title('Profit data')
plt.show()
```




```
labels = ['FaceCream', 'FaseWash', 'ToothPaste', 'Bathing soap', 'Shampoo', 'Moisturizer']
salesData   = [df3 ['facecream'].sum(), df3 ['facewash'].sum(), df3 ['toothpaste'].sum(),
         df3 ['bathingsoap'].sum(), df3 ['shampoo'].sum(), df3 ['moisturizer'].sum()]
plt.axis("equal")
plt.pie(salesData, labels=labels, autopct='%1.1f%%')
plt.legend(loc='lower right')
plt.title('Sales data')
plt.show()
```




```
bathingsoap   = df3 ['bathingsoap'].tolist()
faceWashSalesData   = df3 ['facewash'].tolist()

f, axarr = plt.subplots(2, sharex=True)
axarr[0].plot(monthList, bathingsoap, label = 'Bathingsoap Sales Data', color='k', marker='o'
axarr[0].set_title('Sales data of  a Bathingsoap')
axarr[1].plot(monthList, faceWashSalesData, label = 'Face Wash Sales Data', color='r', marker
axarr[1].set_title('Sales data of  a facewash')

plt.xticks(monthList)
plt.xlabel('Month Number')
plt.ylabel('Sales units in number')
plt.show()
```



+ Code + Text

✓ 0s completed at 11:35 PM