

Aerofit Business Case Study

1. Import the dataset and do the usual data analysis steps like checking the structure & characteristics of the dataset.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy as sp
import seaborn as sns
df=pd.read_csv('/content/aerofit_treadmill.csv')
df.head()
```

The screenshot shows a Google Colab notebook titled "Aerofit Case Study_Solution A.ipynb". The code cell at the top imports necessary libraries and reads the CSV file. Below it, the output shows the first five rows of the DataFrame:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	KP281	18	Male	14	Single	3	4	29562	112
1	KP281	19	Male	15	Single	2	3	31836	76
2	KP281	19	Female	14	Partnered	4	3	30699	66
3	KP281	19	Male	12	Single	3	3	32973	85
4	KP281	20	Male	13	Partnered	4	2	35247	47

The next cell, [8], displays the DataFrame's information:

```
[8] df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Product     180 non-null    object  
 1   Age         180 non-null    int64
```

- The data type of all columns in the “customers” table.
- You can find the number of rows and columns given in the dataset
- Check for the missing values and find the number of missing values in each column

```
df.dtypes
df.shape
df.isnull().sum()
```

```

[7] #.a.The data type of all columns in the "customers" table.
df.dtypes

{x}
Product      object
Age          int64
Gender       object
Education    int64
MaritalStatus object
Usage        int64
Fitness      int64
Income       int64
Miles        int64
dtype: object

[8] #.b.You can find the number of rows and columns given in the dataset
df.shape

(180, 9)

[9] #c.Check for the missing values and find the number of missing values in each column.
df.isnull().sum()

Product      0
Age          0
Gender       0
Education    0
MaritalStatus 0
Usage        0
Fitness      0
Income       0
Miles        0
dtype: int64

```

SENSEX -1.08% 14:23 23-01-2024

2. Detect Outliers:

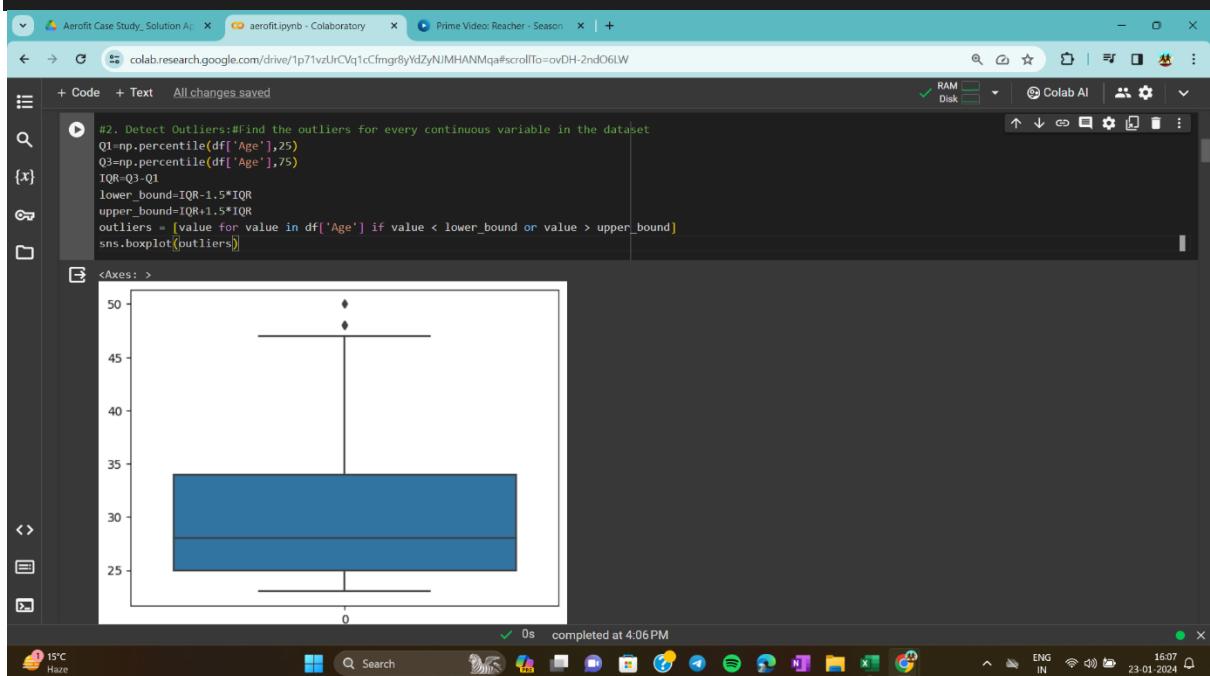
a. Find the outliers for every continuous variable in the dataset.

1. Outliers for “Age”

```

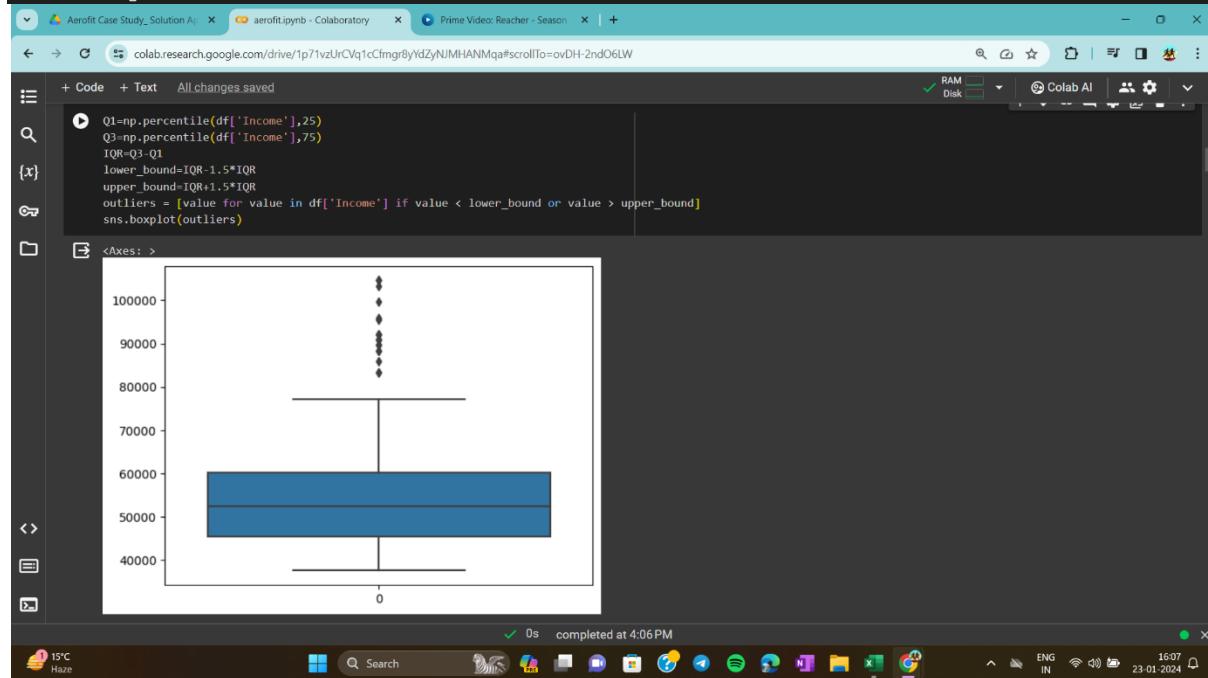
Q1=np.percentile(df['Age'],25)
Q3=np.percentile(df['Age'],75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Age'] if value < lower_bound or
value > upper_bound]
sns.boxplot(outliers)

```



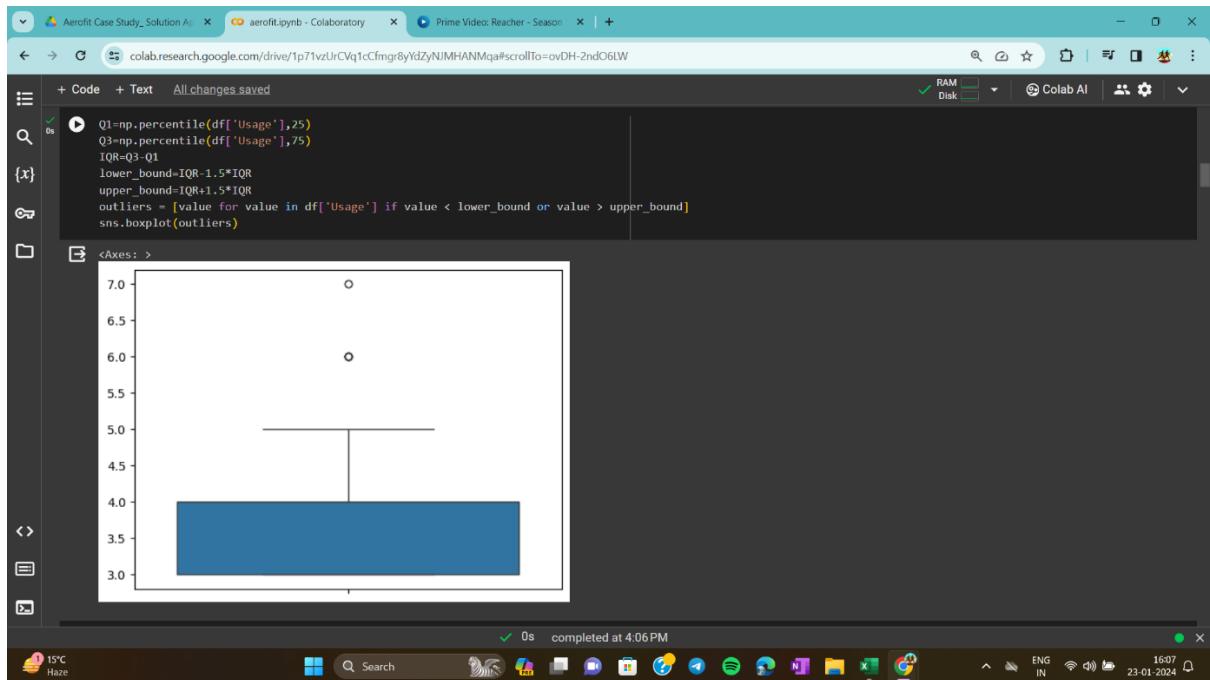
2. Outliers for “Income”

```
Q1=np.percentile(df['Income'],25)
Q3=np.percentile(df['Income'],75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Income'] if value < lower_bound or
value > upper_bound]
sns.boxplot(outliers)
```



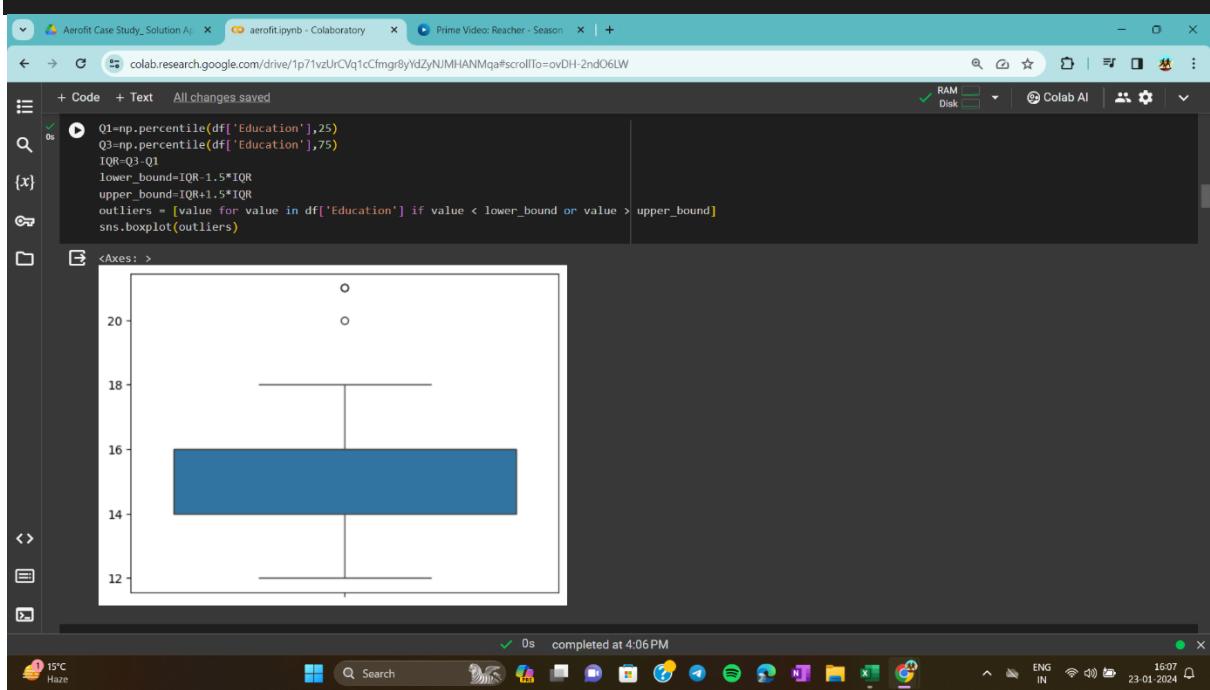
3. outliers for “Usage”

```
Q1=np.percentile(df['Usage'],25)
Q3=np.percentile(df['Usage'],75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Usage'] if value < lower_bound or
value > upper_bound]
sns.boxplot(outliers)
```



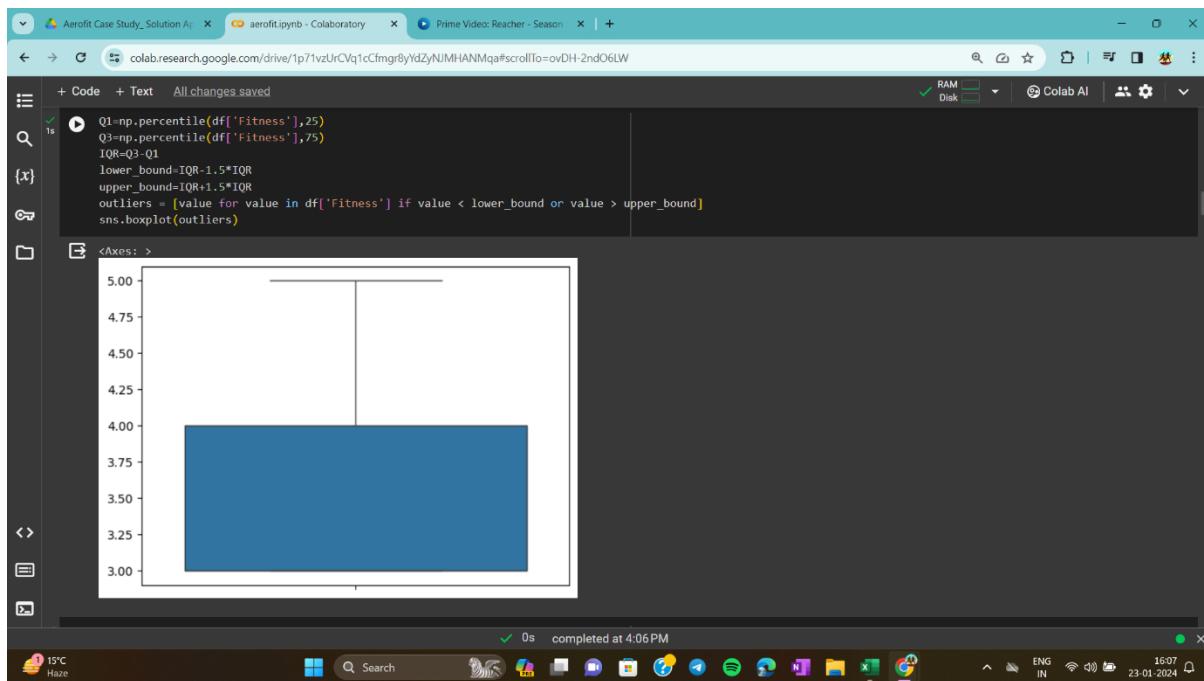
4. Outliers for "Education"

```
Q1=np.percentile(df['Education'], 25)
Q3=np.percentile(df['Education'], 75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Education'] if value < lower_bound or value > upper_bound]
sns.boxplot(outliers)
```



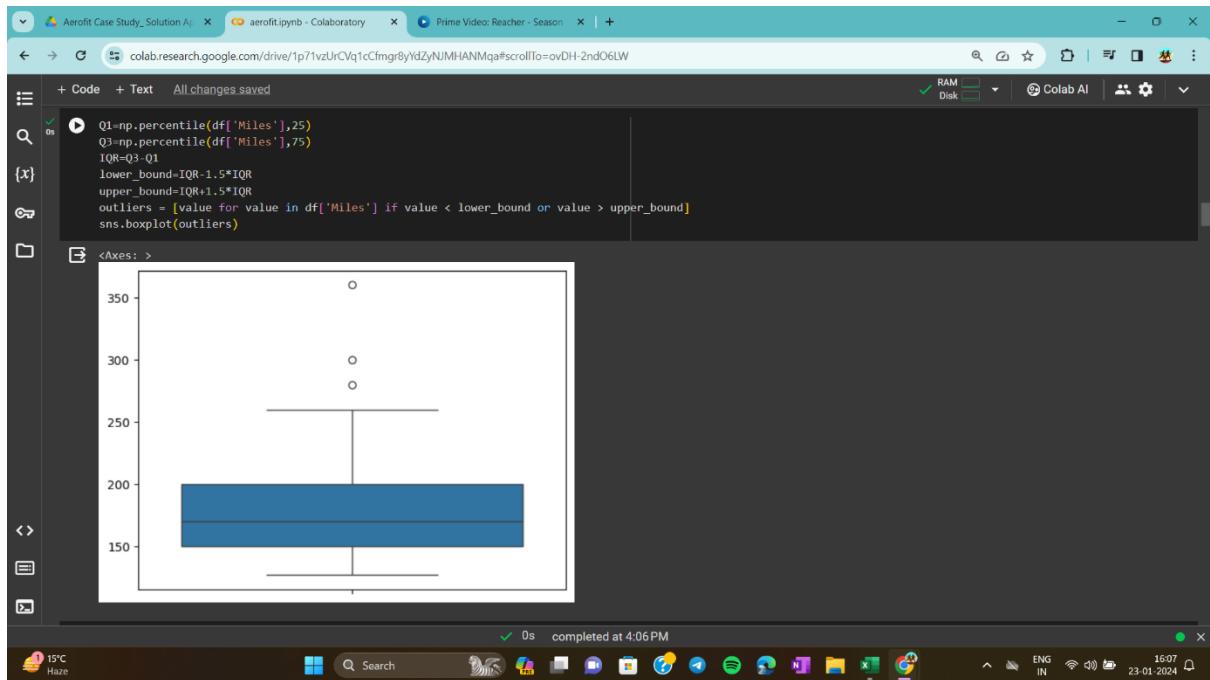
5. outliers for “Fitness”

```
Q1=np.percentile(df['Fitness'],25)
Q3=np.percentile(df['Fitness'],75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Fitness'] if value < lower_bound or
value > upper_bound]
sns.boxplot(outliers)
```



6. Outliers for “Miles”

```
Q1=np.percentile(df['Miles'],25)
Q3=np.percentile(df['Miles'],75)
IQR=Q3-Q1
lower_bound=IQR-1.5*IQR
upper_bound=IQR+1.5*IQR
outliers = [value for value in df['Miles'] if value < lower_bound or
value > upper_bound]
sns.boxplot(outliers)
```



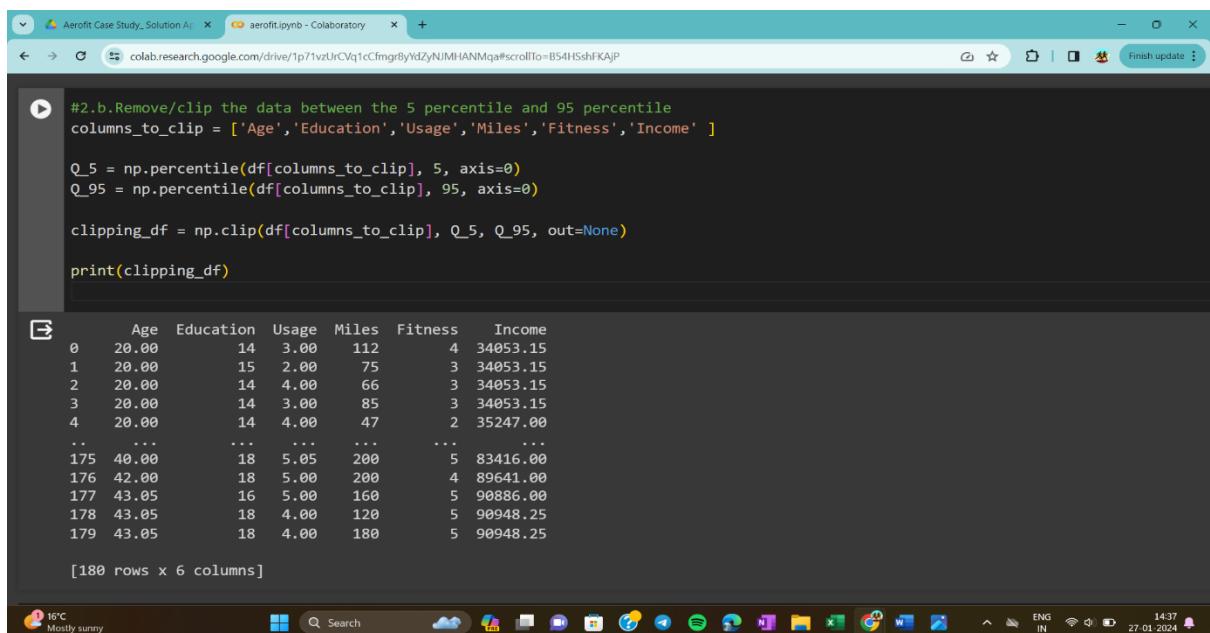
2.b.Remove/clip the data between the 5 percentile and 95 percentile.

```
#2.b.Remove/clip the data between the 5 percentile and 95 percentile
columns_to_clip = ['Age', 'Education', 'Usage', 'Miles', 'Fitness', 'Income']

Q_5 = np.percentile(df[columns_to_clip], 5, axis=0)
Q_95 = np.percentile(df[columns_to_clip], 95, axis=0)

clipping_df = np.clip(df[columns_to_clip], Q_5, Q_95, out=None)

print(clipping_df)
```



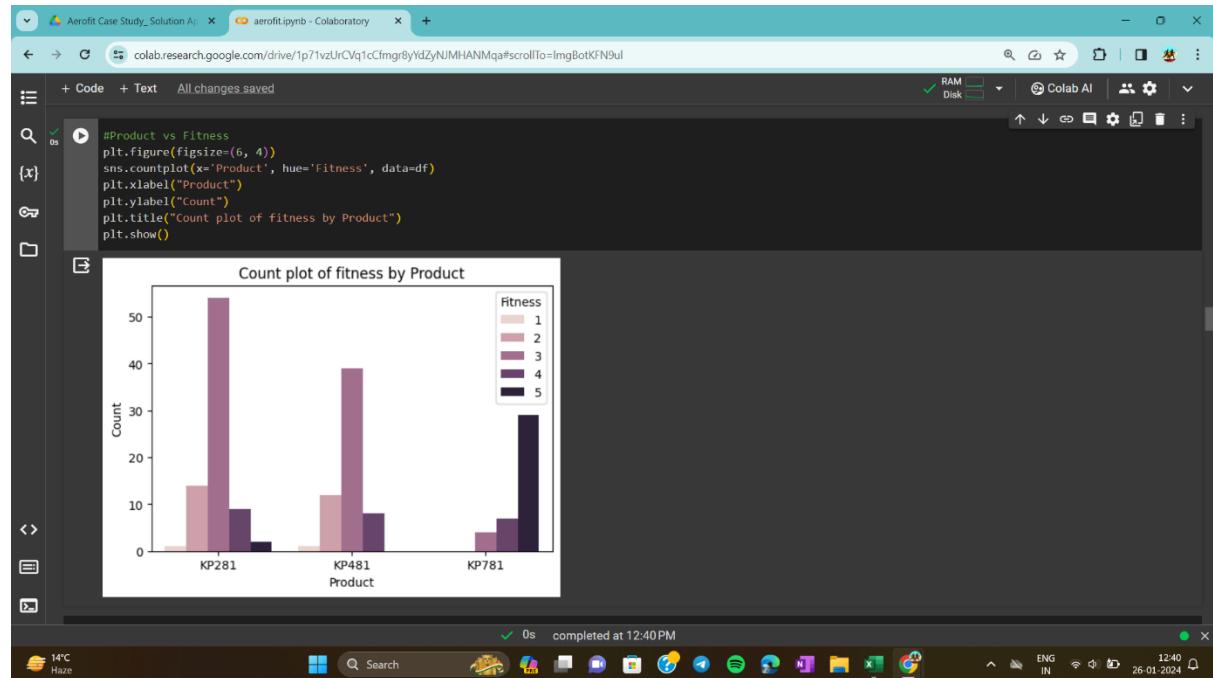
3. Check if features like marital status, Gender, and age have any effect on the product

Purchased.

a. Find if there is any relationship between the categorical variables and the output variable in the data.(count plot to find the relationship between categorical variables and output variables.)

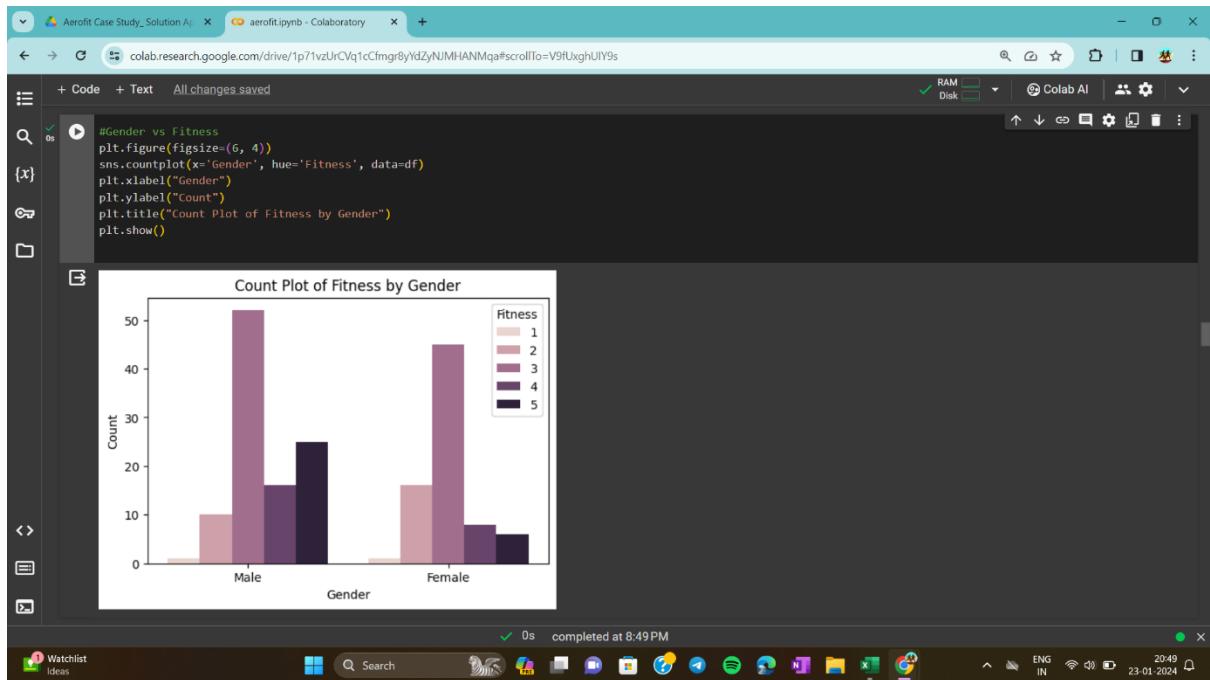
1.

```
#Product vs Fitness
plt.figure(figsize=(6, 4))
sns.countplot(x='Product', hue='Fitness', data=df)
plt.xlabel("Product")
plt.ylabel("Count")
plt.title("Count plot of fitness by Product")
plt.show()
```



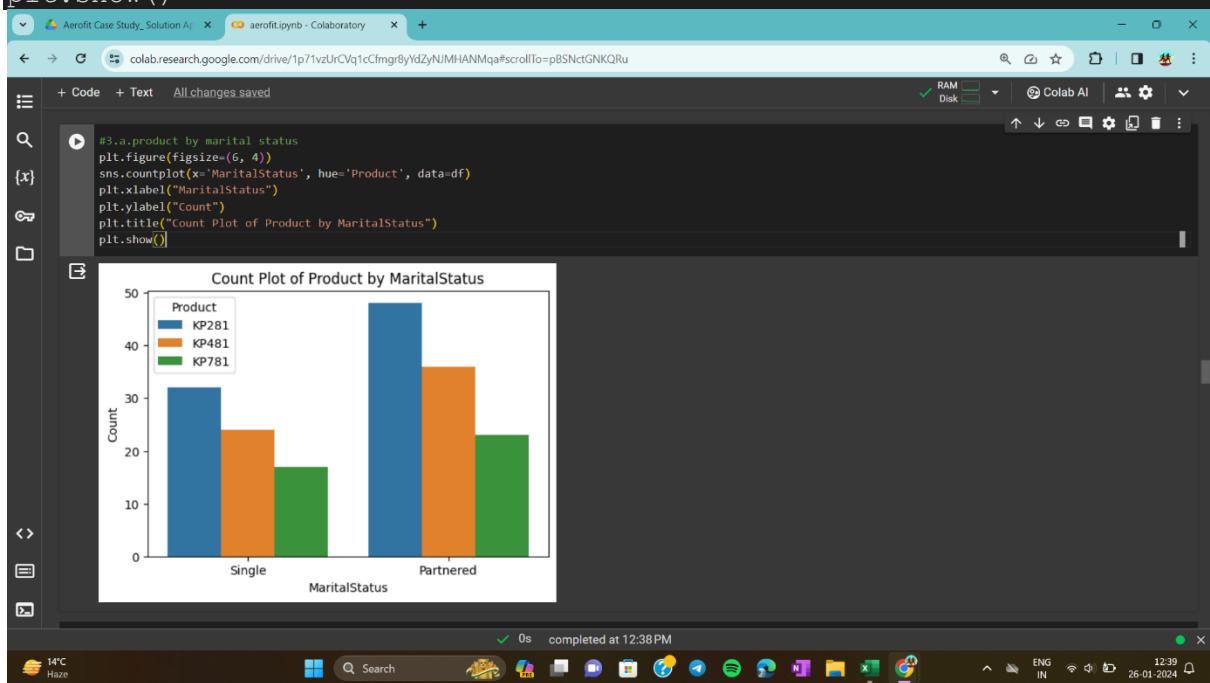
2.

```
#Gender vs Fitness
plt.figure(figsize=(6, 4))
sns.countplot(x='Gender', hue='Fitness', data=df)
plt.xlabel("Gender")
plt.ylabel("Count")
plt.title("Count Plot of Fitness by Gender")
plt.show()
```



3.

```
#3.a.product by marital status
plt.figure(figsize=(6, 4))
sns.countplot(x='MaritalStatus', hue='Product', data=df)
plt.xlabel("MaritalStatus")
plt.ylabel("Count")
plt.title("Count Plot of Product by MaritalStatus")
plt.show()
```



4. #3.a.Find if there is any relationship between the categorical variables and the output variable in the data.

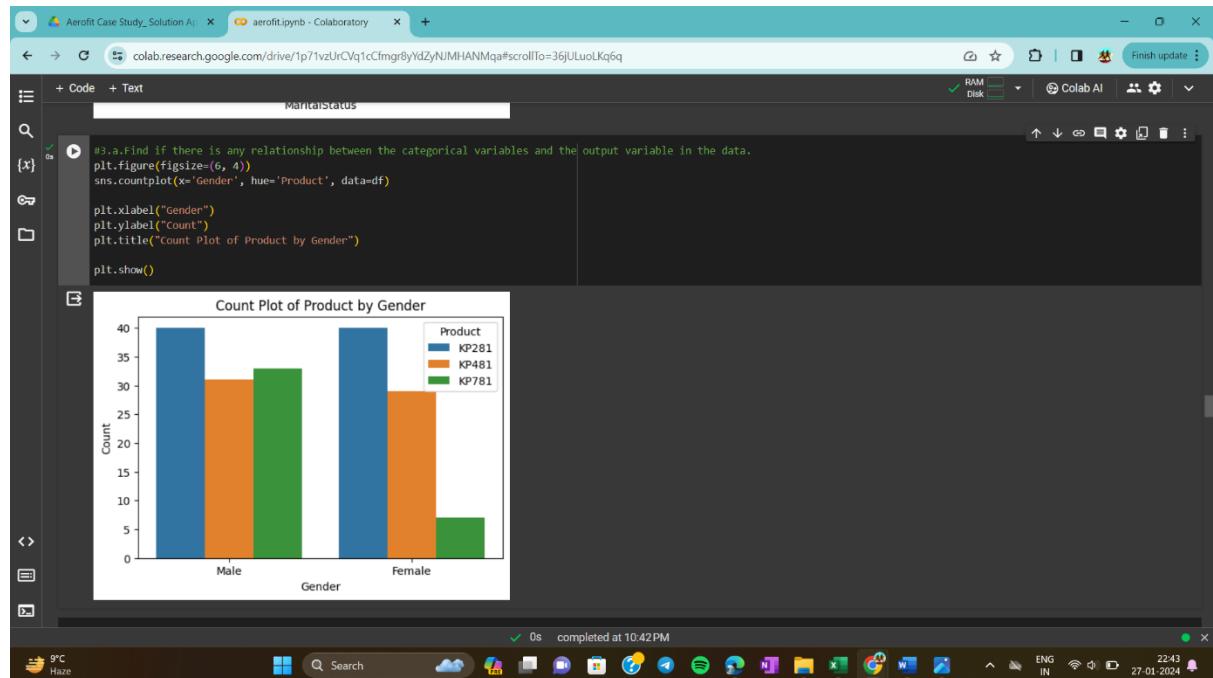
```

plt.figure(figsize=(6, 4))
sns.countplot(x='Gender', hue='Product', data=df)

plt.xlabel("Gender")
plt.ylabel("Count")
plt.title("Count Plot of Product by Gender")

plt.show()

```



5.

```

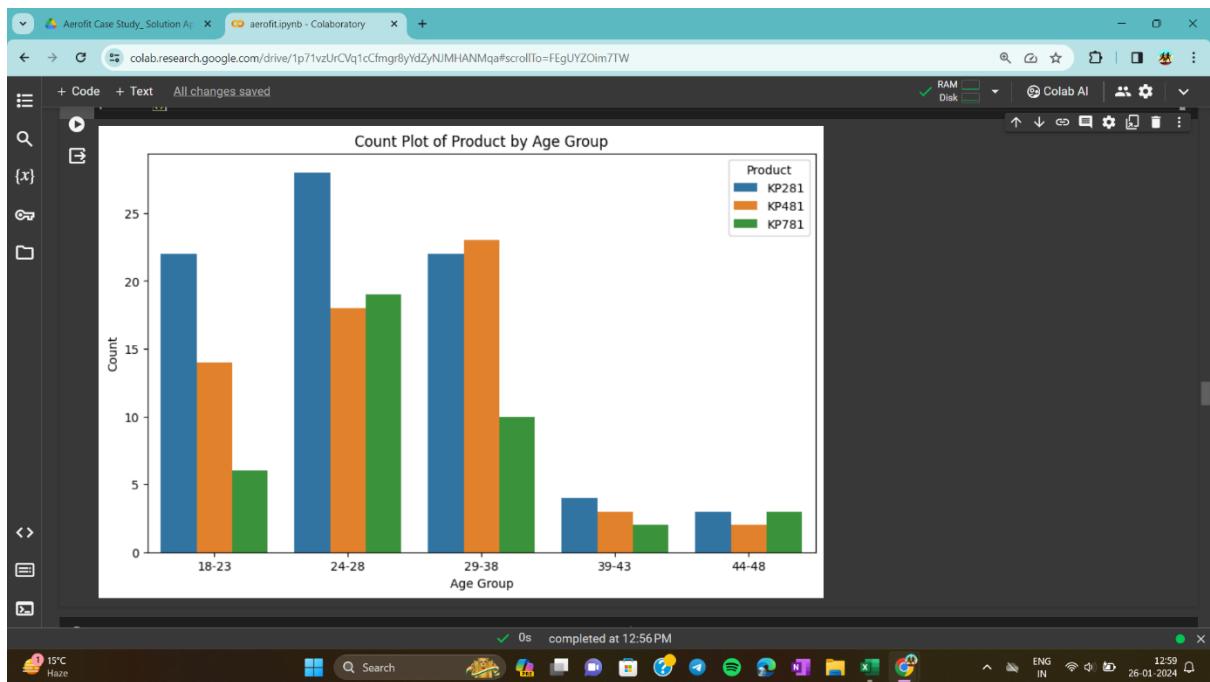
#age vs product
bins = [18, 24, 29, 39, 44, 49]
labels = ['18-23', '24-28', '29-38', '39-43', '44-48']
df['Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels,
right=False)

plt.figure(figsize=(10, 6))
sns.countplot(x='Age Group', hue='Product', data=df)

plt.xlabel("Age Group")
plt.ylabel("Count")
plt.title("Count Plot of Product by Age Group")

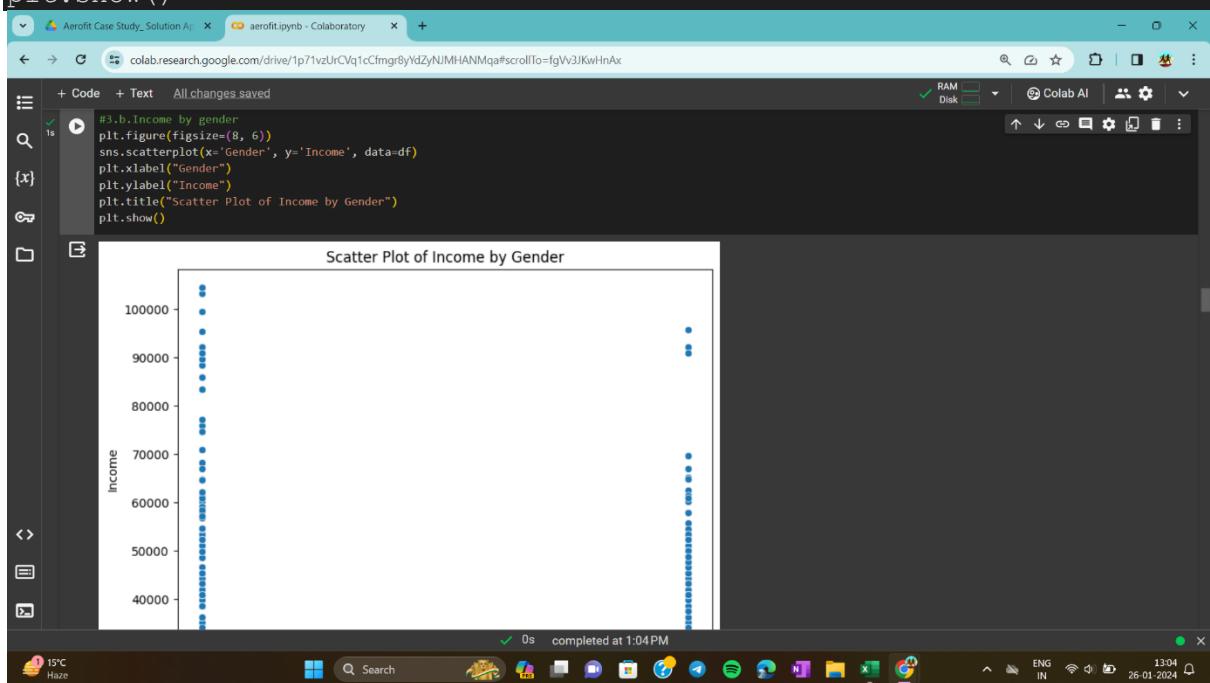
plt.show()

```

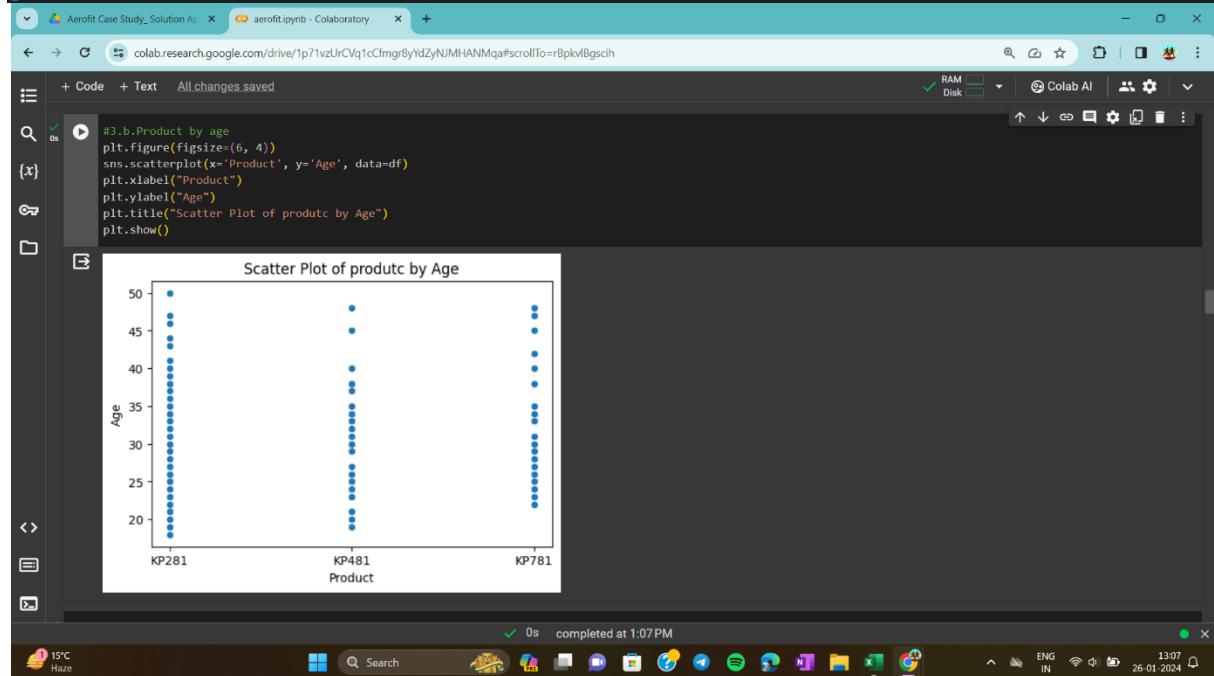


3. b. Find if there is any relationship between the continuous variables and the output variable in the data.

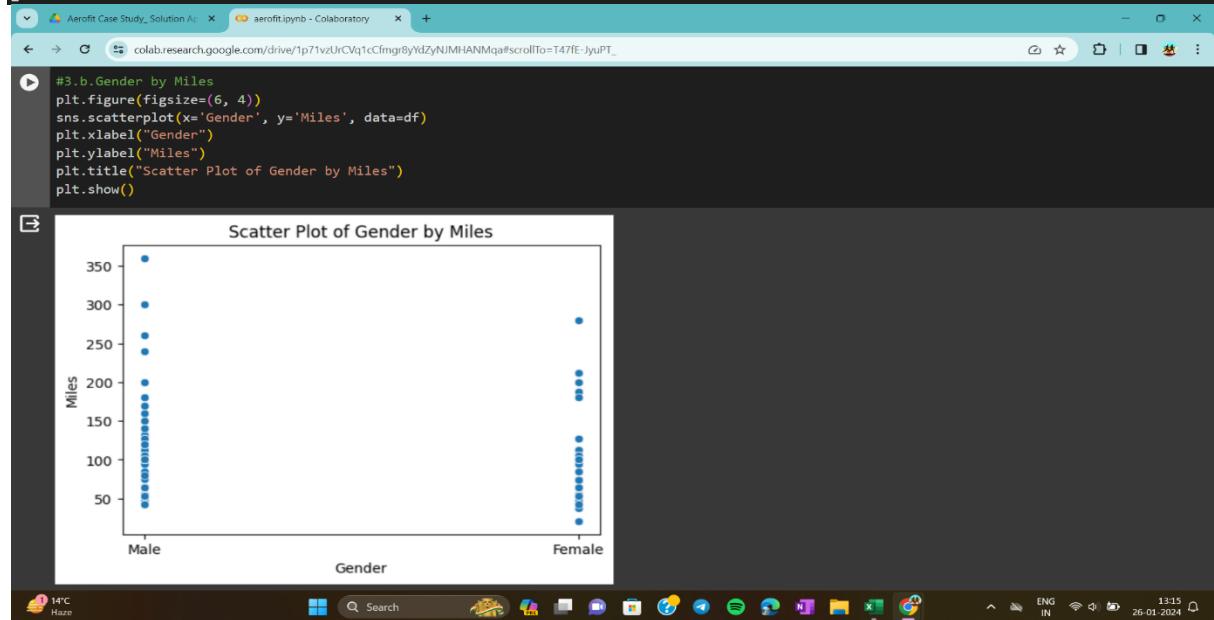
```
#3.b.Income by gender
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Gender', y='Income', data=df)
plt.xlabel("Gender")
plt.ylabel("Income")
plt.title("Scatter Plot of Income by Gender")
plt.show()
```



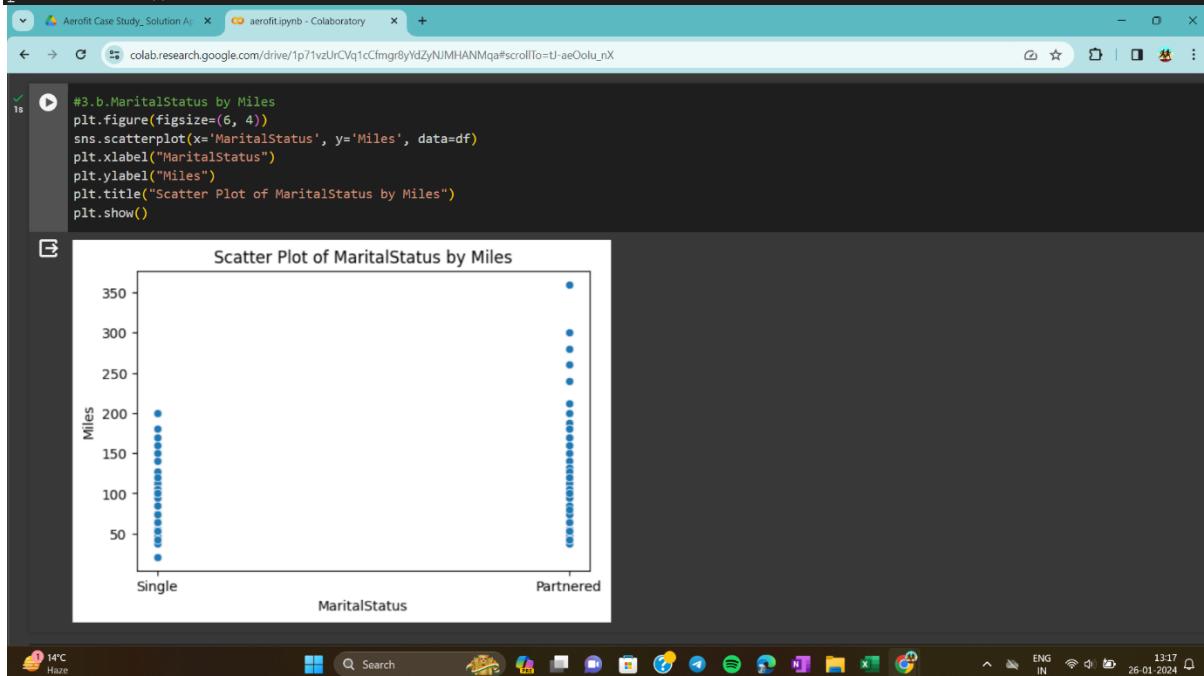
```
#3.b.Product by age
plt.figure(figsize=(6, 4))
sns.scatterplot(x='Product', y='Age', data=df)
plt.xlabel("Product")
plt.ylabel("Age")
plt.title("Scatter Plot of product by Age")
plt.show()
```



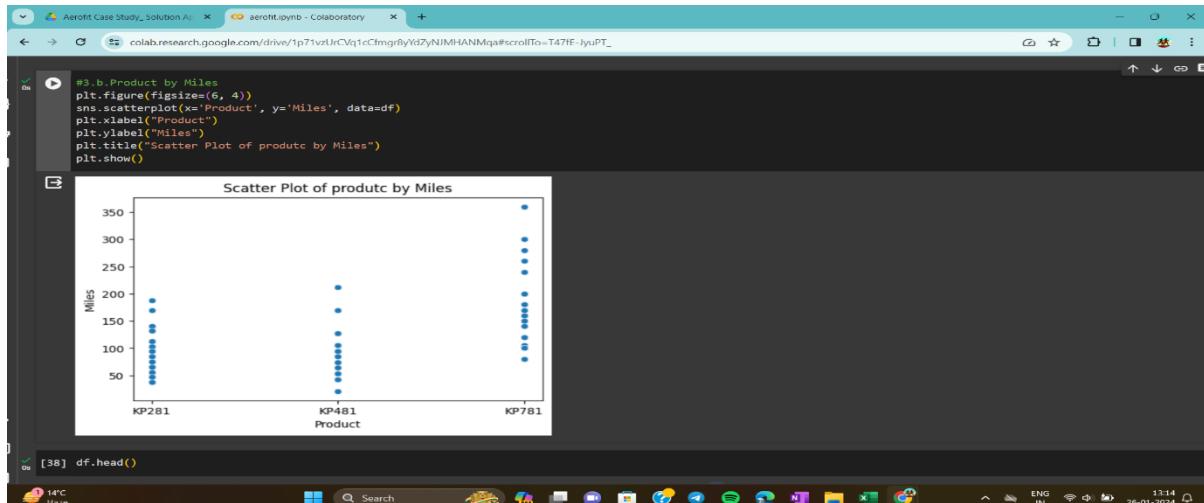
```
#3.b.Gender by Miles
plt.figure(figsize=(6, 4))
sns.scatterplot(x='Gender', y='Miles', data=df)
plt.xlabel("Gender")
plt.ylabel("Miles")
plt.title("Scatter Plot of Gender by Miles")
plt.show()
```



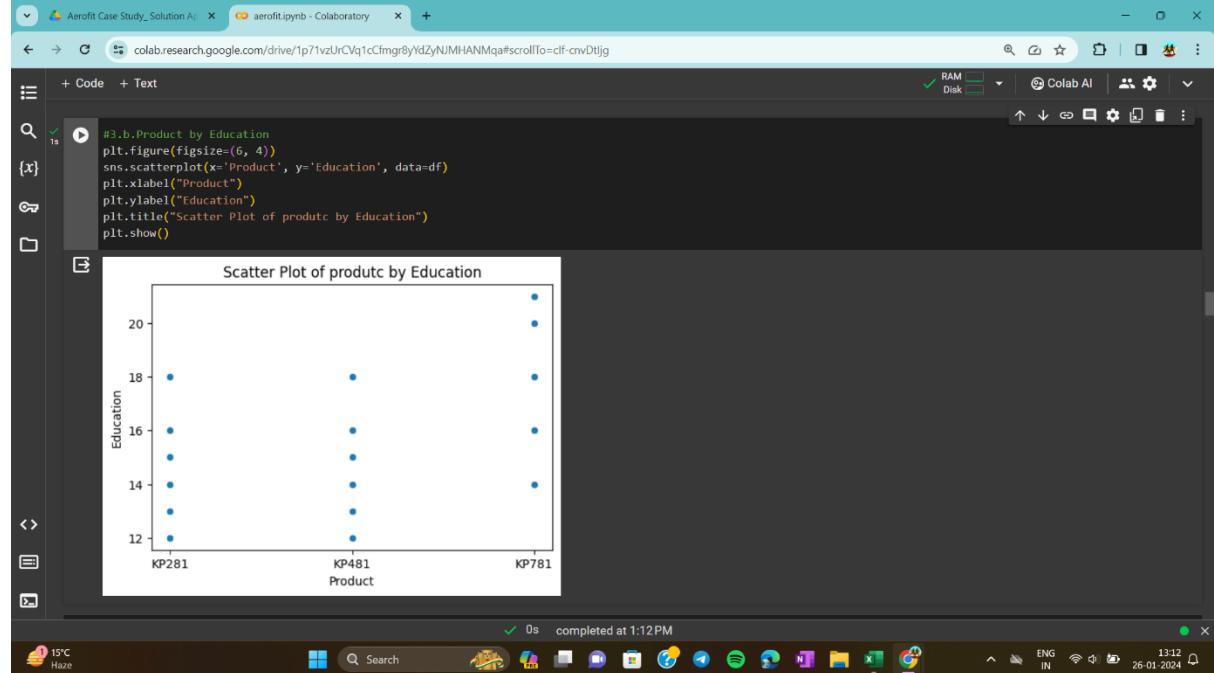
```
#3.b.MaritalStatus by Miles
plt.figure(figsize=(6, 4))
sns.scatterplot(x='MaritalStatus', y='Miles', data=df)
plt.xlabel("MaritalStatus")
plt.ylabel("Miles")
plt.title("Scatter Plot of MaritalStatus by Miles")
plt.show()
```



```
#3.b.Product by Miles
plt.figure(figsize=(6, 4))
sns.scatterplot(x='Product', y='Miles', data=df)
plt.xlabel("Product")
plt.ylabel("Miles")
plt.title("Scatter Plot of Product by Miles")
plt.show()
```



```
#3.b.Product by Education
plt.figure(figsize=(6, 4))
sns.scatterplot(x='Product', y='Education', data=df)
plt.xlabel("Product")
plt.ylabel("Education")
plt.title("Scatter Plot of product by Education")
plt.show()
```



4. Representing the Probability.

- Find the marginal probability (what per cent of customers have purchased KP281, KP481, or KP781)
- Find the probability that the customer buys a product based on each column.
- Find the conditional probability that an event occurs given that another event has occurred. Find the conditional probability that an event occurs given that another event has occurred. (Example: given that a customer is female, what is the probability she'll purchase a KP481)

```
#a.Find the marginal probability (what percent of customers have purchased KP281,KP481 or KP781)
marginal_prob_for_ages=pd.crosstab(index=df['Product'],columns=df['Age'],normalize=True,margins=True)
marginal_prob_for_ages

marginal_prob_for_Gender=pd.crosstab(index=df['Product'],columns=df['Gender'],normalize=True,margins=True)
marginal_prob_for_Gender
```

AeroFit Case Study_Solution.ipynb

```
#44. Representing the Probability  
#a. Find the marginal probability (what percent of customers have purchased KP281, KP481 or KP781)  
marginal_prob_for_ages=pd.crosstab(index=df['Product'],columns=df['Age'],normalize=True,margins=True)  
marginal_prob_for_ages
```

Age	18	19	20	21	22	23	24	25	26	27	...	41	42	43	44	45	46	47	48	
Product	KP281	0.005556	0.016667	0.011111	0.022222	0.022222	0.044444	0.027778	0.038889	0.038889	0.016667	...	0.005556	0.000000	0.005556	0.005556	0.000000	0.005556	0.005556	0.000000
	KP481	0.000000	0.005556	0.016667	0.016667	0.000000	0.038889	0.016667	0.061111	0.016667	0.005556	...	0.000000	0.000000	0.000000	0.000000	0.005556	0.000000	0.005556	0.000000
	KP781	0.000000	0.000000	0.000000	0.000000	0.016667	0.016667	0.022222	0.038889	0.011111	0.016667	...	0.000000	0.005556	0.000000	0.000000	0.005556	0.000000	0.005556	0.000000
	All	0.005556	0.022222	0.027778	0.038889	0.038889	0.100000	0.066667	0.138889	0.066667	0.038889	...	0.005556	0.005556	0.005556	0.005556	0.011111	0.005556	0.011111	0.011111

4 rows × 33 columns

```
[ ] marginal_prob_for_Gender=pd.crosstab(index=df['Product'],columns=df['Gender'],normalize=True,margins=True)  
marginal_prob_for_Gender
```

Gender	Female	Male	All	
Product	KP281	0.222222	0.222222	0.444444
	KP481	0.161111	0.172222	0.333333
	KP781	0.038889	0.183333	0.222222
	All	0.422222	0.577778	1.000000

```
[ ] marginal_prob_for_Education=pd.crosstab(index=df['Product'],columns=df['Education'],normalize=True,margins=True)
```

0s completed at 1:20PM

14°C Haze

```
marginal_prob_for_Education=pd.crosstab(index=df['Product'],columns=df['Education'],normalize=True,margins=True)  
marginal_prob_for_Education
```



```
marginal_prob_for_MaritalStatus=pd.crosstab(index=df['Product'],columns=df['MaritalStatus'],normalize=True,margins=True)  
marginal_prob_for_MaritalStatus
```

AeroFit Case Study_Solution.ipynb

```
[ ] marginal_prob_for_Education=pd.crosstab(index=df['Product'],columns=df['Education'],normalize=True,margins=True)  
marginal_prob_for_Education
```

Education	12	13	14	15	16	18	20	21	All	
Product	KP281	0.011111	0.016667	0.166667	0.022222	0.216667	0.011111	0.000000	0.000000	0.444444
	KP481	0.005556	0.011111	0.127778	0.005556	0.172222	0.011111	0.000000	0.000000	0.333333
	KP781	0.000000	0.000000	0.011111	0.000000	0.083333	0.105556	0.005556	0.016667	0.222222
	All	0.016667	0.027778	0.305556	0.027778	0.472222	0.127778	0.005556	0.016667	1.000000

```
[ ] marginal_prob_for_Maritalstatus=pd.crosstab(index=df['Product'],columns=df['Maritalstatus'],normalize=True,margins=True)  
marginal_prob_for_Maritalstatus
```

MaritalStatus	Partnered	Single	All	
Product	KP281	0.266667	0.177778	0.444444
	KP481	0.200000	0.133333	0.333333
	KP781	0.127778	0.084444	0.222222
	All	0.594444	0.405556	1.000000

```
[ ] marginal_prob_for_Usage=pd.crosstab(index=df['Product'],columns=df['Usage'],normalize=True,margins=True)  
marginal_prob_for_Usage
```

0s completed at 1:20PM

14°C Haze

```
marginal_prob_for_Usage=pd.crosstab(index=df['Product'],columns=df['Usage'],normalize=True,margins=True)  
marginal_prob_for_Usage
```

```
marginal_prob_for_Fitness=pd.crosstab(index=df['Product'],columns=df['Fitness'],normalize=True,margins=True)
marginal_prob_for_Fitness
```

The screenshot shows a Google Colab notebook titled "Aeroft Case Study_Solution A.ipynb". The code cell contains:

```
marginal_prob_for_Fitness=pd.crosstab(index=df['Product'],columns=df['Fitness'],normalize=True,margins=True)
marginal_prob_for_Fitness
```

The resulting output is a crosstabulation table:

	Fitness	1	2	3	4	5	All
Product	KP281	0.005556	0.077778	0.300000	0.050000	0.011111	0.444444
	KP481	0.005556	0.066667	0.216667	0.044444	0.000000	0.333333
	KP781	0.000000	0.005556	0.100000	0.066667	0.038889	0.111111
	All	0.011111	0.144444	0.538889	0.133333	0.172222	1.000000

```
marginal_prob_for_Income=pd.crosstab(index=df['Product'],columns=df['Income'],normalize=True,margins=True)
marginal_prob_for_Income
```

```
marginal_prob_for_Miles=pd.crosstab(index=df['Product'],columns=df['Miles'],normalize=True,margins=True)
marginal_prob_for_Miles
```

The screenshot shows a Google Colab notebook titled "Aeroft Case Study_Solution A.ipynb". The code cell contains:

```
marginal_prob_for_Income=pd.crosstab(index=df['Product'],columns=df['Income'],normalize=True,margins=True)
marginal_prob_for_Income
```

The resulting output is a crosstabulation table:

	Income	29562	38699	31836	32973	34110	35247	36384	37521	38658	39795	...	88396	89641	90886	92131	95508	95866	99601	103336
Product	KP281	0.005556	0.005556	0.016667	0.011111	0.027778	0.016667	0.011111	0.016667	0.011111	...	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
	KP481	0.000000	0.000000	0.011111	0.016667	0.000000	0.005556	0.000000	0.011111	0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
	KP781	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.011111	0.011111	0.016667	0.016667	0.005556	0.005556	0.005556	0.005556	
	All	0.005556	0.005556	0.011111	0.027778	0.027778	0.022222	0.011111	0.027778	0.011111	...	0.011111	0.011111	0.016667	0.016667	0.005556	0.005556	0.005556	0.005556	

4 rows × 33 columns

```

marginal_prob_for_Age_Group=pd.crosstab(index=df['Product'],columns=df['Age Group'],normalize=True,margins=True)
marginal_prob_for_Age_Group

```

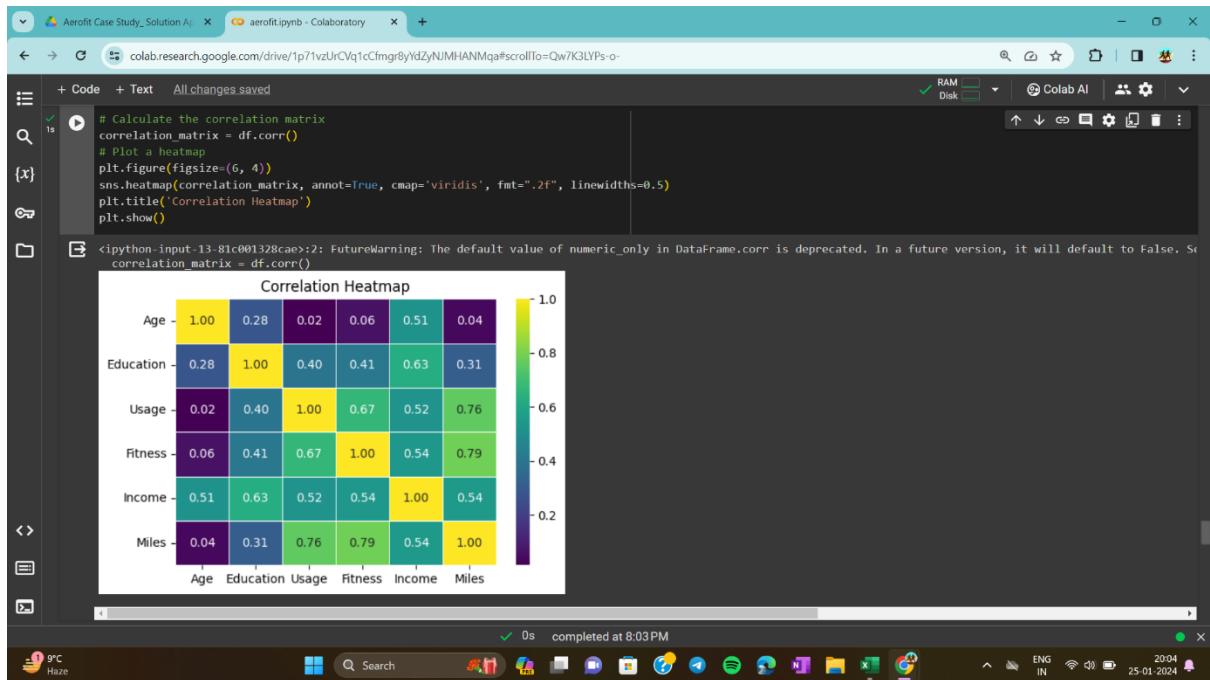
5. Check the correlation among different factors.

Find the correlation between the given features in the table. (Use of heatmap and corr function to find the correlation between the variables)

```

# Calculate the correlation matrix
correlation_matrix = df.corr()
# Plot a heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(correlation_matrix, annot=True, cmap='viridis', fmt=".2f",
            linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()

```



6. Customer profiling and recommendation

Make customer profiling for each and every product.

```
import pandas as pd

selected_product = 'KP281'

product_data = df[df['Product'] == selected_product]

product_data[['Age', 'Income', 'Gender']].head()

selected_product = 'KP481'

product_data = df[df['Product'] == selected_product]

product_data[['Age', 'Income', 'Gender']].head()
```

Aeroft Case Study_Solution A.ipynb · aerofit.ipynb - Colaboratory

[19] #Make customer profilings for each and every product.
selected_product = 'KP281'
Filter the DataFrame for the selected product
product_data = df[df['Product'] == selected_product]
product_data[['Age','Income','Gender']].head()

	Age	Income	Gender
0	18	29562	Male
1	19	31836	Male
2	19	30699	Female
3	19	32973	Male
4	20	35247	Male

[20] selected_product = 'KP481'
Filter the DataFrame for the selected product
product_data = df[df['Product'] == selected_product]
product_data[['Age','Income','Gender']].head()

	Age	Income	Gender
80	19	31836	Male
81	20	32973	Male
82	20	34110	Female
83	20	38658	Male
84	21	34110	Female

0s completed at 8:11 PM

9°C Haze

```
selected_product = 'KP781'  
# Filter the DataFrame for the selected product  
product_data = df[df['Product'] == selected_product]  
product_data[['Age','Income','Gender']].head()
```

[17] selected_product = 'KP481'
Filter the DataFrame for the selected product
product_data = df[df['Product'] == selected_product]
product_data[['Age','Income','Gender']].head()

	Age	Income	Gender
80	19	31836	Male
81	20	32973	Male
82	20	34110	Female
83	20	38658	Male
84	21	34110	Female

[18] selected_product = 'KP781'
Filter the DataFrame for the selected product
product_data = df[df['Product'] == selected_product]
product_data[['Age','Income','Gender']].head()

	Age	Income	Gender
140	22	48658	Male
141	22	54781	Male
142	22	48556	Male
143	23	58516	Male
144	23	53536	Female

0s completed at 8:11 PM

9°C Haze