Insights of Data

Import libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Import the dataset

```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749 -O aerofit_treadmill.c
```

```
--2023-08-24 05:49:36--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?163999274
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 108.157.172.10, 108.157.172.173, 108.157.172.183, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|108.157.172.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7279 (7.1K) [text/plain]
Saving to: ‘aerofit_treadmill.csv’

aerofit_treadmill.c 100%[===================>]   7.11K  --.-KB/s    in 0s

2023-08-24 05:49:36 (90.6 MB/s) - ‘aerofit_treadmill.csv’ saved [7279/7279]
```

```
df= pd.read_csv('aerofit_treadmill.csv')
```

Analysing the structure & characteristics of the dataset

```
df.head()
```

	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	75
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

```
df.shape
```

```
(180, 9)
```

```
df.columns
```

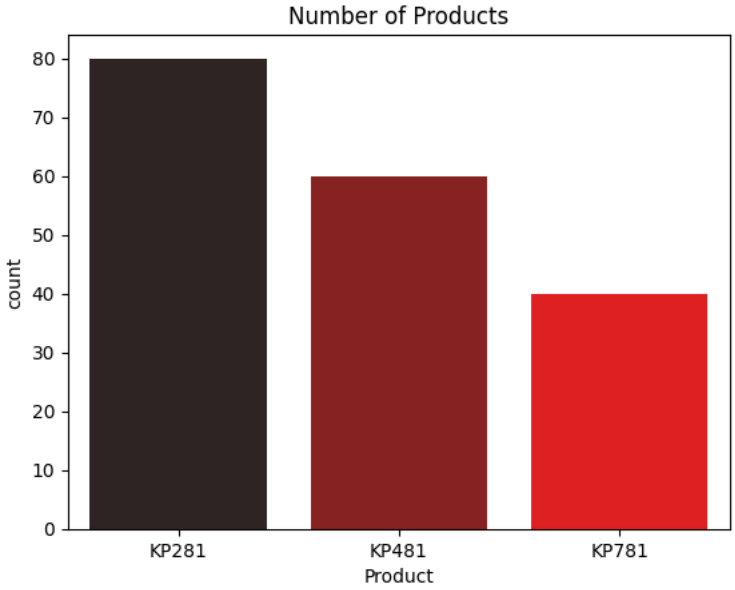
```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles'],
      dtype='object')
```

```
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 
 2   Gender         180 non-null    object
 3   Education      180 non-null    int64 
 4   MaritalStatus  180 non-null    object
 5   Usage          180 non-null    int64 
 6   Fitness        180 non-null    int64 
 7   Income         180 non-null    int64 
 8   Miles          180 non-null    int64 
```

```
dtypes: int64(6), object(3)
memory usage: 12.8+ KB
```

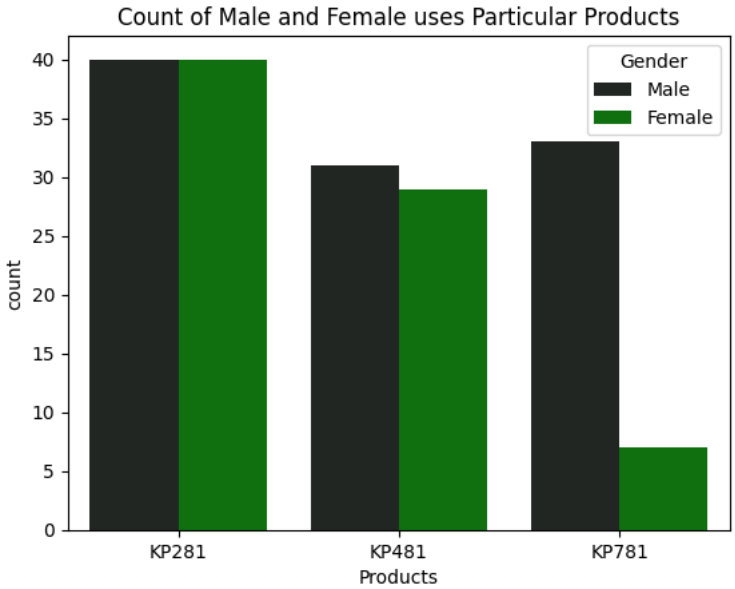
```
# Number of products
sns.countplot(data=df,x='Product',palette='dark:red')
plt.title("Number of Products")
plt.show()
```




```
#total no of Products
print('Total no of products:', df.Product.nunique())
print('Products are:', df.Product.unique())
```

```
Total no of products: 3
Products are: ['KP281' 'KP481' 'KP781']
```

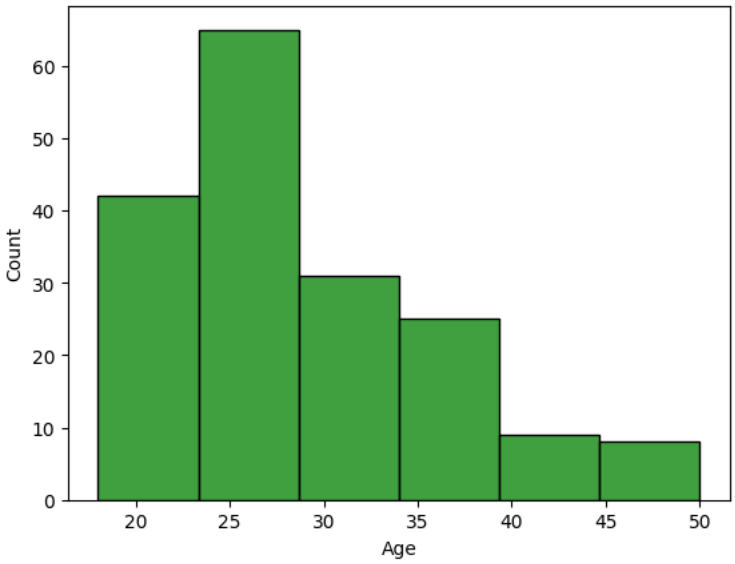
```
#Gender-wise customers for products
sns.countplot(x = "Product", data= df, hue = "Gender", palette='dark:green')
plt.xlabel("Products")
plt.title("Count of Male and Female uses Particular Products")
plt.show()
```




```
# Gender wise number of customers
df['Gender'].value_counts()
```

```
Male      104
Female     76
Name: Gender, dtype: int64
```

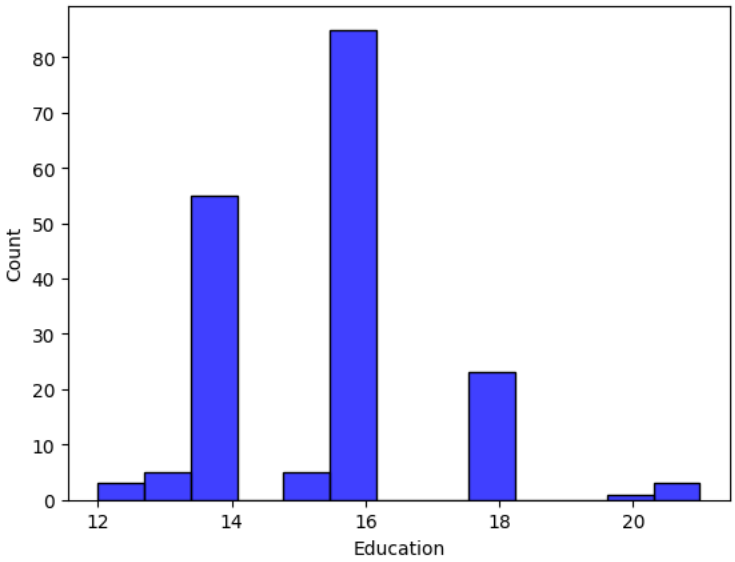
```
# Age Analysis - Histogram
sns.histplot(data=df,x='Age',bins=6, color='green')
plt.show()
```




```
# Analysis of ages
print('Total no of unique age is:', df['Age'].nunique())
print('List of unique ages are:', df.Age.unique())
print('Minimum age is:', df.Age.min())
print('Maximum age is:', df.Age.max())
```

```
Total no of unique age is: 32
List of unique ages are: [18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
 43 44 46 47 50 45 48 42]
Minimum age is: 18
Maximum age is: 50
```

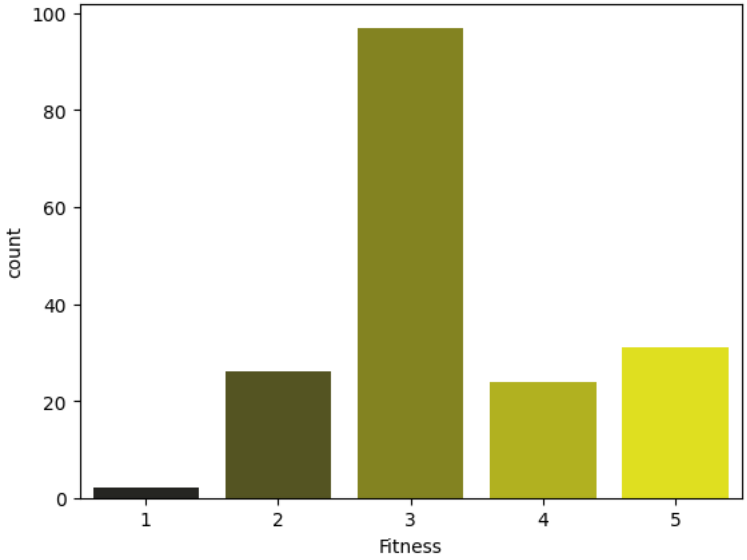
```
# Education Analysis
sns.histplot(data=df,x='Education', color='blue')
plt.show()
```




```
# list of unique Educations
print('Total no. of unique education is:', df.Education.nunique())
print('Educations are:', df.Education.unique().tolist())
```

```
Total no. of unique education is: 8
Educations are: [14, 15, 12, 13, 16, 18, 20, 21]
```

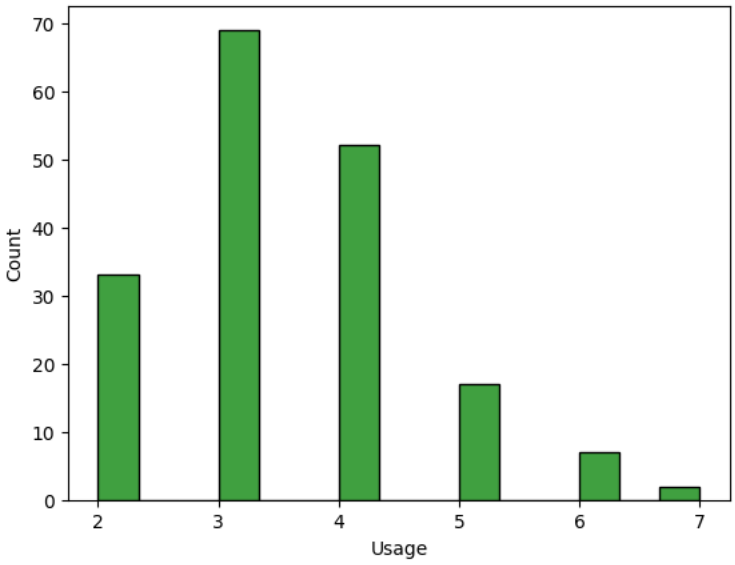
```
# Fitness rating
sns.countplot(data=df,x='Fitness',palette='dark:yellow')
plt.show()
```




```
# Number of customer on the basis of fitness rating scale 1 to 5
df.Fitness.value_counts().sort_index()
```

```
1     2
2    26
3    97
4    24
5    31
Name: Fitness, dtype: int64
```

```
# Usage Analysis
sns.histplot(data=df,x='Usage', color='green')
plt.show()
```

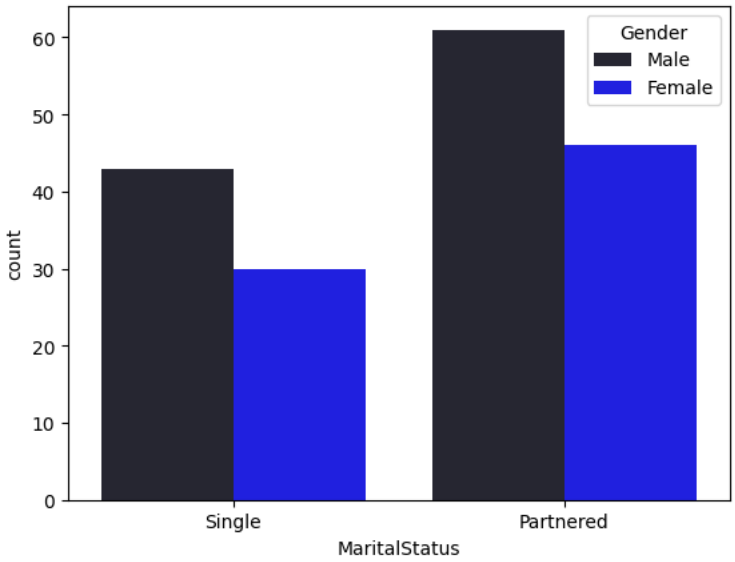



```
# Number of customers counts on Usage
df.Usage.value_counts().sort_index()
```

```
2    33
3    69
4    52
5    17
```

```
6     7
7     2
Name: Usage, dtype: int64
```

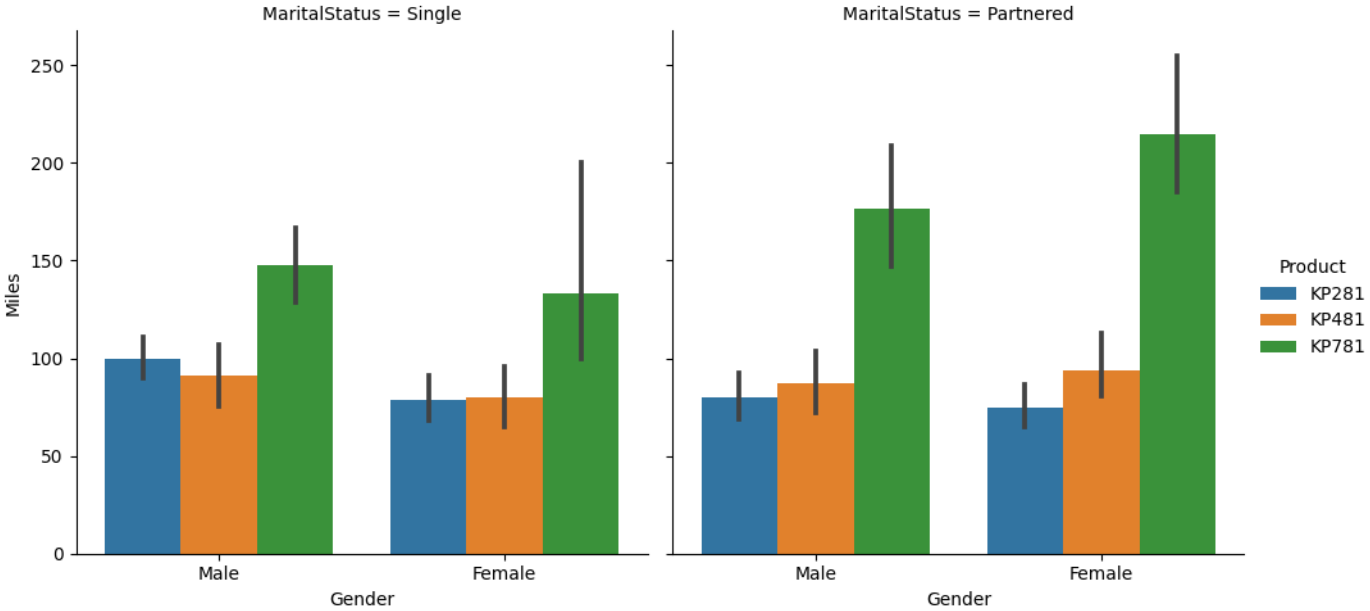
```
# Count among Gender and their Marital Status
sns.countplot(data=df,x='MaritalStatus',hue='Gender',palette='dark:blue')
plt.show()
```



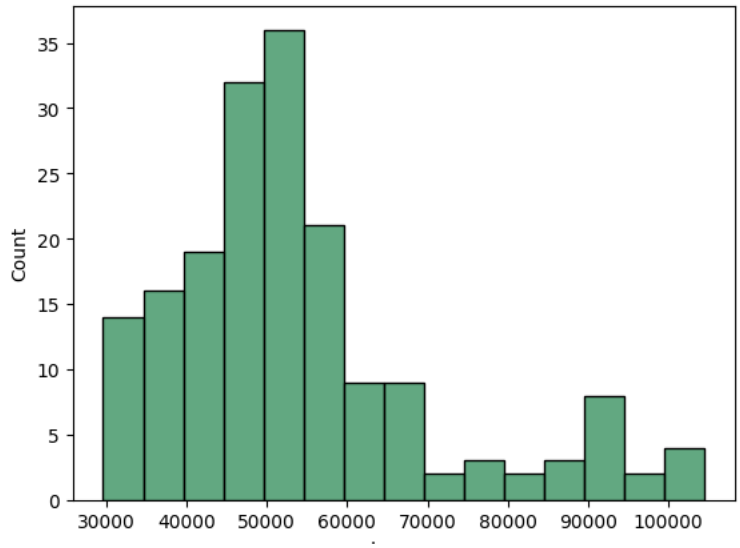

```
# Number of Single and Partnered customers
df['MaritalStatus'].value_counts()
```

```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

```
# Miles covered in each product by gender and their marital status
sns.catplot(x='Gender',y='Miles',hue='Product',col='MaritalStatus',data=df,kind='bar')
plt.show()
```




```
# Income Analysis - Histogram
sns.histplot(data=df,x='Income',bins=15, color='seagreen')
plt.show()
```




```
#Income analysis
print('Minimum Income recorded for customer is:', df.Income.min())
print('Average Income of the purchased customer is:', np.round(df.Income.mean(),2))
print('Highest Income recorded for customer is:', df.Income.max())
```

```
Minimum Income recorded for customer is: 29562
Average Income of the purchased customer is: 53719.58
Highest Income recorded for customer is: 104581
```

Conversion of Categorical attributes to Category

```
# Converting Int data type of fitness rating to object data type
df_cat = df
df_cat['Fitness_category'] = df_cat.Fitness

df_cat["Fitness_category"].replace({1:"Poor Shape",
                            2:"Bad Shape",
                            3:"Average Shape",
                            4:"Good Shape",
                            5:"Excellent Shape"},inplace=True)
df_cat.head()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_category
0	KP281	18	Male	14	Single	3	4	29562	112	Good Shape
1	KP281	19	Male	15	Single	2	3	31836	75	Average Shape
2	KP281	19	Female	14	Partnered	4	3	30699	66	Average Shape
3	KP281	19	Male	12	Single	3	3	32973	85	Average Shape
4	KP281	20	Male	13	Partnered	4	2	35247	47	Bad Shape

Missing Values & Duplicates

```
#finding missing values

df.isnull().sum().sort_values(ascending=False)
```

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

```
#finding duplicate values

df.duplicated().sum()
```

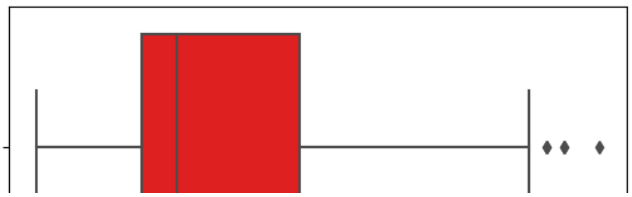
```
0
```

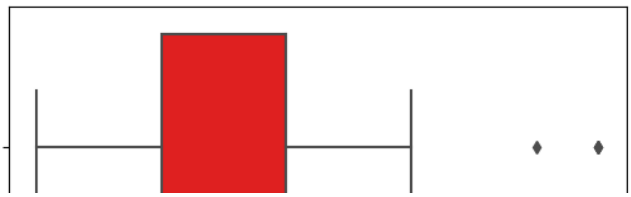
Detect outliers

```
df.describe()
```

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	5.000000	104581.000000	360.000000

```
#outliers by box-plots
fig, axis = plt.subplots(3,2, figsize=(14,7))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x="Age", orient='h', ax=axis[0,0], color= 'r')
sns.boxplot(data=df, x="Education", orient='h', ax=axis[0,1], color= 'r')
sns.boxplot(data=df, x="Usage", orient='h', ax=axis[1,0], color='b')
sns.boxplot(data=df, x="Fitness", orient='h', ax=axis[1,1], color='b')
sns.boxplot(data=df, x="Income", orient='h', ax=axis[2,0], color='g')
sns.boxplot(data=df, x="Miles", orient='h', ax=axis[2,1], color='g')
plt.show()
```





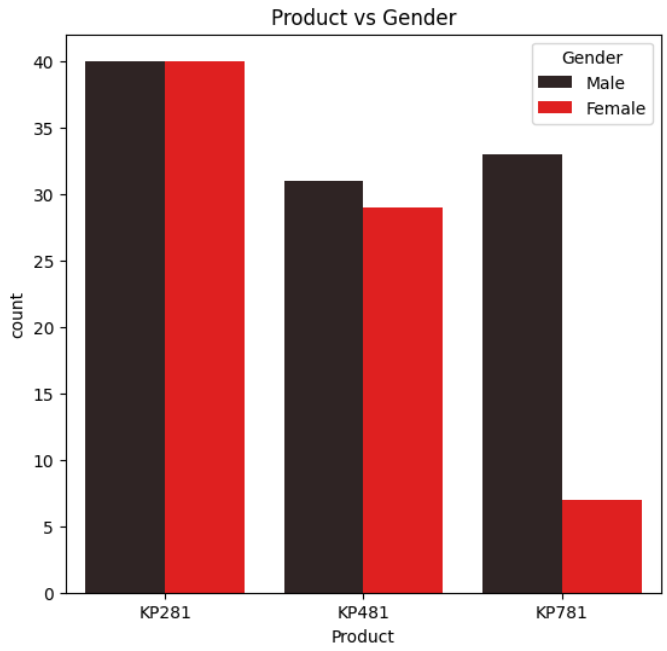
Features like gender, marital status, age, education, usage, fitness, income and miles have any effect on the product purchased

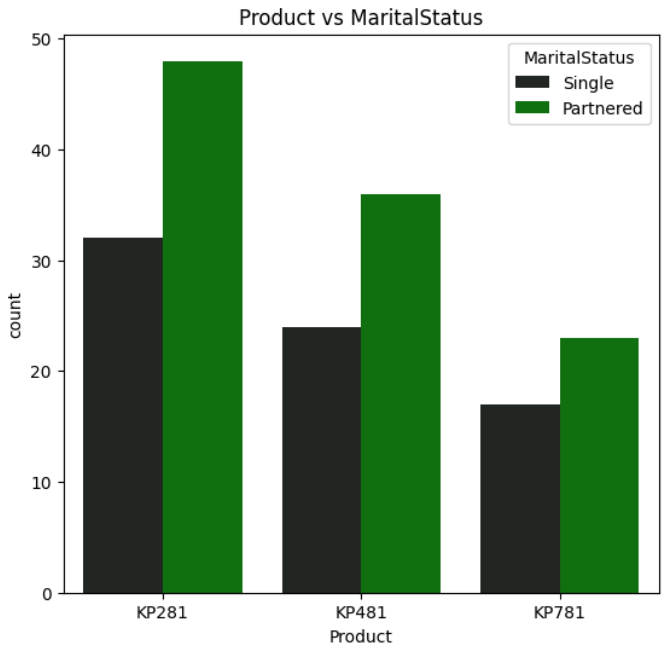
```
df.head(2)
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_category	Age_group
0	KP281	18	Male	14	Single	3	4	29562	112	Good Shape	Teen
1	KP281	19	Male	15	Single	2	3	31836	75	Average Shape	Teen

Bivariate- (Categorial-Categorial) Analysis

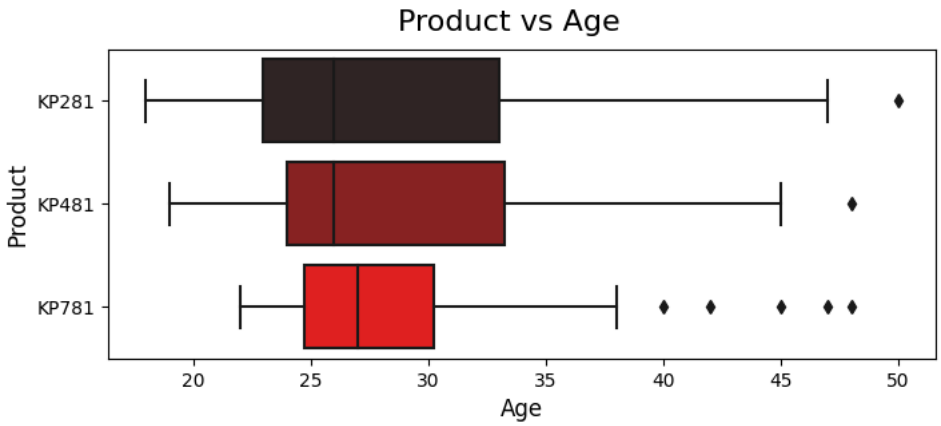
```
#Bivariate- Categorial-categorial analysis

fig, axs= plt.subplots(1,2,figsize=(14,6))

sns.countplot(data=df,x="Product",hue="Gender",palette="dark:red",ax=axs[0])
axs[0].set_title("Product vs Gender")

sns.countplot(data=df,x="Product",hue="MaritalStatus",palette="dark:green",ax=axs[1])
axs[1].set_title("Product vs MaritalStatus")
plt.show()
```



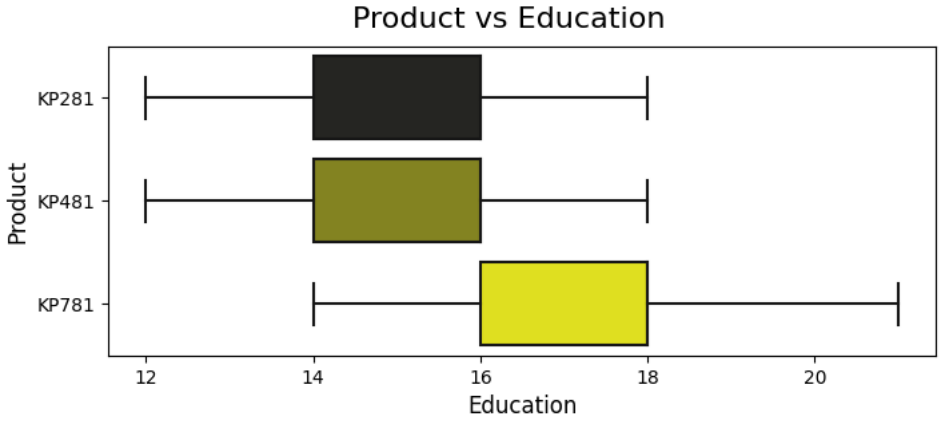


Bivariate-(Contineous-Categorial) Analysis

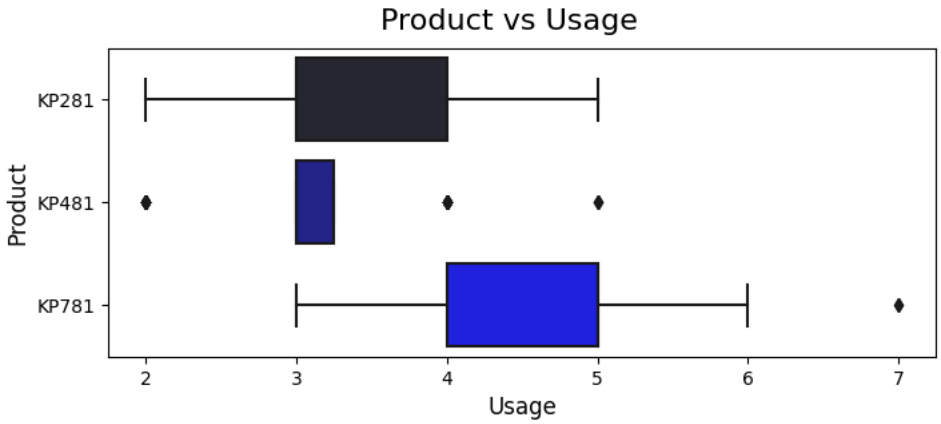
```
#Effect of Age on the product purchased

plt.figure(figsize=(8,3)).suptitle('Product vs Age', fontsize=16)

sns.boxplot(x='Age',y='Product',data=df, palette='dark:red')
plt.xlabel('Age', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```




```
#Effect of Education on the product purchased

plt.figure(figsize=(8,3)).suptitle('Product vs Education', fontsize=16)

sns.boxplot(x='Education',y='Product',data=df, palette='dark:yellow')
plt.xlabel('Education', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```

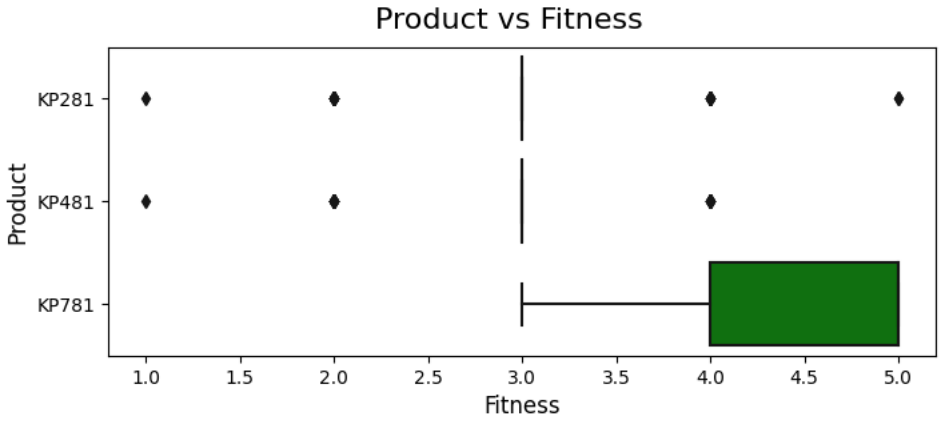



```
#Effect of Usage on the product purchased

plt.figure(figsize=(8,3)).suptitle('Product vs Usage', fontsize=16)

sns.boxplot(x='Usage',y='Product',data=df, palette='dark:blue')
plt.xlabel('Usage', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```

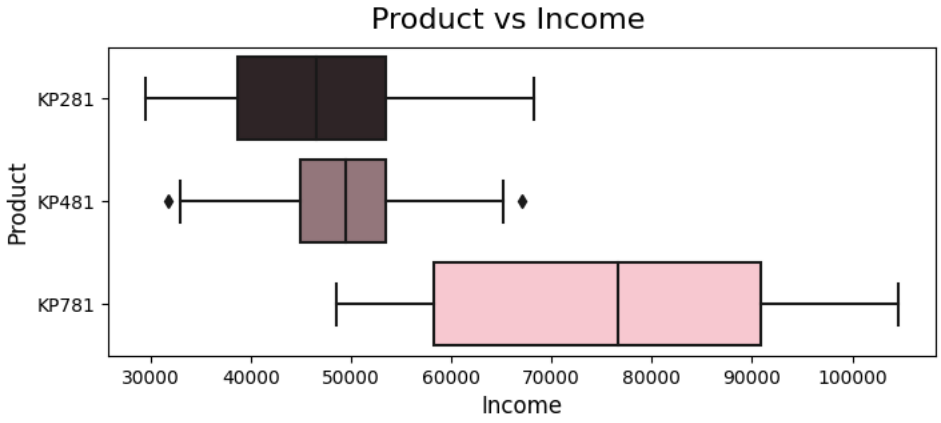



```
#Effect of Fitness on the product purchased
```

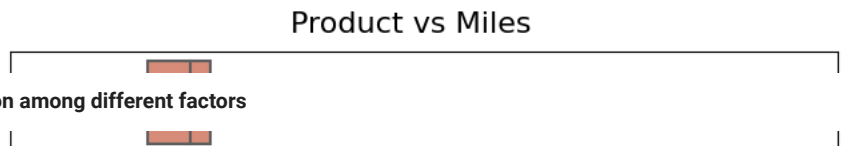
```
plt.figure(figsize=(8,3)).suptitle('Product vs Fitness', fontsize=16)

sns.boxplot(x='Fitness',y='Product',data=df, palette='dark:green')
plt.xlabel('Fitness', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```




```
#Effect of Income on the product purchased

plt.figure(figsize=(8,3)).suptitle('Product vs Income', fontsize=16)

sns.boxplot(x='Income',y='Product',data=df, palette='dark:pink')
plt.xlabel('Income', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```



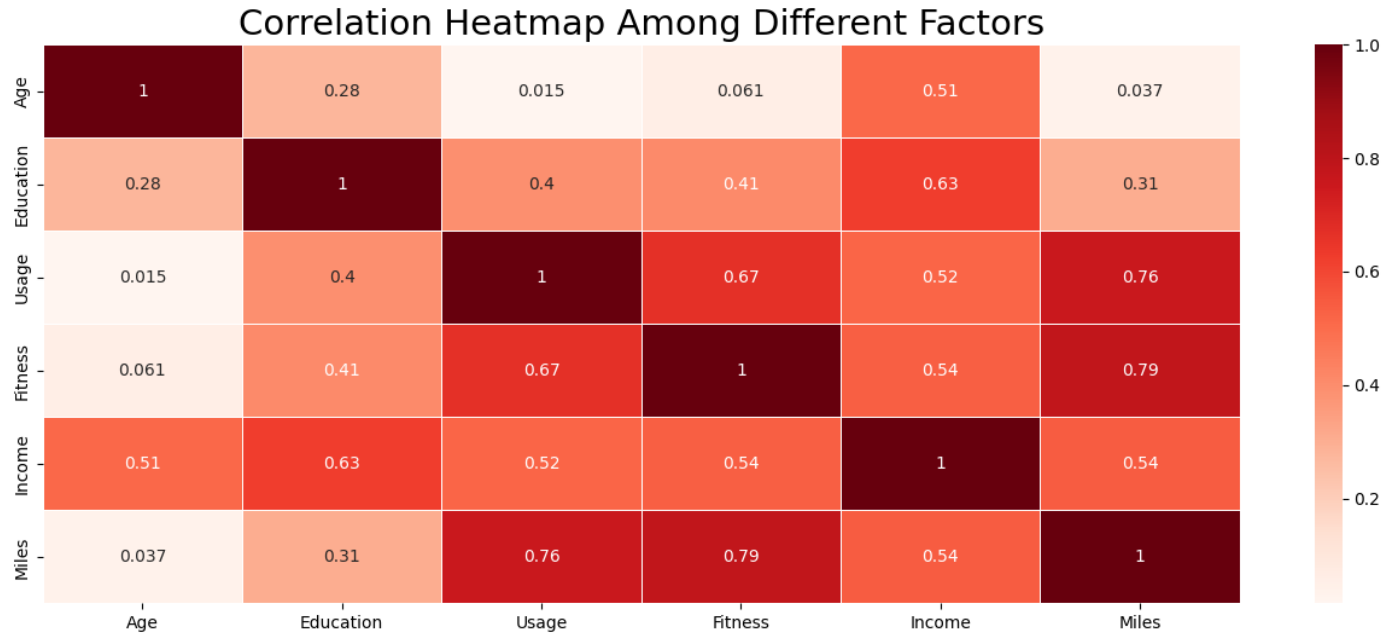

```
#Effect of Miles on the product purchased

plt.figure(figsize=(8,3)).suptitle('Product vs Miles', fontsize=16)

sns.boxplot(x='Miles',y='Product',data=df, palette='RdBu')
plt.xlabel('Miles', fontsize=12)
plt.ylabel('Product', fontsize=12)
plt.show()
```



Check correlation among different factors

```
#Correlation HeatMap

plt.figure(figsize=(16,6))

sns.heatmap(df.corr(),annot=True, cmap='Reds',linewidths=.5)
plt.title('Correlation Heatmap Among Different Factors', fontsize=21)

plt.show()
```

```
<ipython-input-38-a8a8b86bd8e7>:5: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future versior
  sns.heatmap(df.corr(),annot=True, cmap='Reds',linewidths=.5)
```



Marginal Probability

Product purchased percentage as per different Age group

```
#Grouping the age in 4 categories as per their different age range
# 0-21-->'Teen'
# 22-35-->'Adult'
# 36-45-->'Middle Aged'
# 46-60--> 'Elder'

df_cat['Age_group'] = df_cat.Age

df_cat.Age_group = pd.cut(df_cat.Age_group,bins=[0,21,35,45,60],labels=['Teen','Adult','Middle Aged','Elder'])

pd.crosstab(index=df_cat.Product,columns=df_cat.Age_group,margins=True)
```

Age_group	Teen	Adult	Middle Aged	Elder	All
Product					
KP281	10	56	11	3	80
KP481	7	45	7	1	60
KP781	0	34	4	2	40
All	17	135	22	6	180

```
# Marginal Probabilities with product type and age group
np.round(pd.crosstab(index=df_cat.Product,columns=df_cat.Age_group,normalize=True,margins=True, margins_name='Total')*100,2)
```

Age_group	Teen	Adult	Middle Aged	Elder	Total
Product					
KP281	5.56	31.11	6.11	1.67	44.44
KP481	3.89	25.00	3.89	0.56	33.33
KP781	0.00	18.89	2.22	1.11	22.22
Total	9.44	75.00	12.22	3.33	100.00

Product purchased percentage as per Gender-wise

```
#Marginal Probabilities with poroduct type and Gender
np.round((pd.crosstab(index=[df.Product],columns=[df.Gender],normalize=True,margins=True, margins_name='Total'))*100,2)
```

Gender	Female	Male	Total
Product			
KP281	22.22	22.22	44.44
KP481	16.11	17.22	33.33
KP781	3.89	18.33	22.22
Total	42.22	57.78	100.00

Product Purchased percentage as per Fitness category

```
##Marginal Probabilities with product type and fitness category
np.round(pd.crosstab(index=df_cat.Product, columns= df_cat.Fitness_category, normalize=True, margins= True, margins_name='Total')*100,2)
```

Fitness_category	Average Shape	Bad Shape	Excellent Shape	Good Shape	Poor Shape	Total
Product						
KP281	30.00	7.78	1.11	5.00	0.56	44.44
KP481	21.67	6.67	0.00	4.44	0.56	33.33
KP781	2.22	0.00	16.11	3.89	0.00	22.22
Total	53.89	14.44	17.22	13.33	1.11	100.00

Product purchased percentage as per Marital Status

```
#Marginal Probabilities with product type and marital status
round(pd.crosstab(index=df_cat.Product,columns=df_cat.MaritalStatus,normalize=True,margins= True, margins_name='Total')*100,2)
```

MaritalStatus	Partnered	Single	Total
Product			
KP281	26.67	17.78	44.44
KP481	20.00	13.33	33.33
KP781	12.78	9.44	22.22
Total	59.44	40.56	100.00

Product purchased percentage as per Gender wise- Marital Status

```
round(pd.crosstab(index=[df_cat.Product,df_cat.MaritalStatus],columns=df_cat.Gender,normalize=True,margins=True, margins_name='Total'),2)
```

Product	MaritalStatus	Female	Male	Total
KP281	Partnered	0.15	0.12	0.27
	Single	0.07	0.11	0.18
KP481	Partnered	0.08	0.12	0.20
	Single	0.08	0.06	0.13
KP781	Partnered	0.02	0.11	0.13

Conditional Probability

Among Age group, the probability of Product purchasing

```
# Conditional Probabilities with product type and age group
np.round(pd.crosstab(index=df_cat.Product,columns=df_cat.Age_group,normalize='columns',margins=True, margins_name= 'Total')*100,2)
```

Product	Teen	Adult	Middle Aged	Elder	Total
KP281	58.82	41.48	50.00	50.00	44.44
KP481	41.18	33.33	31.82	16.67	33.33
KP781	0.00	25.19	18.18	33.33	22.22

Among Gender, the probability of Product purchasing

```
# Conditional Probabilities with product type and Gender
np.round((pd.crosstab(df.Product,df.Gender,margins=True, margins_name='Total',normalize="columns"))*100,2)
```

Product	Female	Male	Total
KP281	52.63	38.46	44.44
KP481	38.16	29.81	33.33
KP781	9.21	31.73	22.22

Among Fitness category, the probability of Product purchasing

```
#Conditional Probabilities with product type and fitness category
np.round(pd.crosstab(index=df_cat.Product, columns= df_cat.Fitness_category, normalize='columns', margins= True, margins_name='Total')*100,2)
```

Product	Average Shape	Bad Shape	Excellent Shape	Good Shape	Poor Shape	Total
KP281	55.67	53.85	6.45	37.50	50.0	44.44
KP481	40.21	46.15	0.00	33.33	50.0	33.33
KP781	4.12	0.00	93.55	29.17	0.0	22.22

Among Marital Status, the probability of Product purchasing

```
#Conditional Probabilities with product type and marital status
round(pd.crosstab(index=df_cat.Product,columns=df_cat.MaritalStatus,normalize='columns',margins= True, margins_name='Total')*100,2)
```

MaritalStatus	Partnered	Single	Total
Product			
KP281	44.86	43.84	44.44
KP481	33.64	32.88	33.33
KP781	21.50	23.29	22.22

✓ 0s completed at 11:28 AM

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.