About Aerofit

Aerofit is a leading brand in the field of fitness equipment. Aerofit provides a product range including machines such as treadmills, exercise bikes, gym equipment, and fitness accessories to cater to the needs of all categories of people.

Business Problem

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to the new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

Perform descriptive analytics to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts. For each AeroFit treadmill product, construct two-way contingency tables and compute all conditional and marginal probabilities along with their insights/impact on the business.

1.***Defining Problem Statement and Analysing basic metrics***

Problem Statement: Identifing the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to the new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

1.1 Observations on shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), statistical summary

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
!wget
https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125
/original/aerofit_treadmill.csv
df = pd.read_csv('aerofit_treadmill.csv')
```

```
--2023-09-27 06:24:11--
https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125
/original/aerofit_treadmill.csv
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

2023-09-27 06:24:11 (2.01 GB/s) - ‘aerofit_treadmill.csv’ saved
[7279/7279]


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.isnull().sum()).sum()

0
```

This dataset has no missing values so no missing value treatment is required, and we can proceed with the analysis .

```
df.columns

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

df.shape

(180, 9)

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
```

```
df.describe(include='all')
```

```
       Product         Age Gender   Education MaritalStatus
Usage  \
count      180  180.000000    180  180.000000           180
180.000000
unique       3         NaN      2         NaN             2
NaN
top      KP281         NaN   Male         NaN     Partnered
NaN
freq        80         NaN    104         NaN           107
NaN
mean       NaN   28.788889    NaN   15.572222           NaN
3.455556
std        NaN    6.943498    NaN    1.617055           NaN
1.084797
min        NaN   18.000000    NaN   12.000000           NaN
2.000000
25%        NaN   24.000000    NaN   14.000000           NaN
3.000000
50%        NaN   26.000000    NaN   16.000000           NaN
3.000000
75%        NaN   33.000000    NaN   16.000000           NaN
4.000000
max        NaN   50.000000    NaN   21.000000           NaN
7.000000

           Fitness         Income       Miles
count   180.000000     180.000000  180.000000
unique         NaN            NaN         NaN
top            NaN            NaN         NaN
freq           NaN            NaN         NaN
mean      3.311111   53719.577778  103.194444
std       0.958869   16506.684226   51.863605
min       1.000000   29562.000000   21.000000
25%       3.000000   44058.750000   66.000000
50%       3.000000   50596.500000   94.000000
75%       4.000000   58668.000000  114.750000
max       5.000000  104581.000000  360.000000
```

conversion of categorical attributes to 'category'

```
df1 = df
df1['Fitness_category'] = df.Fitness
df1.head()
```

```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness
Income  \
0   KP281   18    Male         14        Single      3        4
29562
1   KP281   19    Male         15        Single      2        3
31836
2   KP281   19  Female         14     Partnered      4        3
30699
3   KP281   19    Male         12        Single      3        3
32973
4   KP281   20    Male         13     Partnered      4        2
35247

   Miles  Fitness_category
0    112                 4
1     75                 3
2     66                 3
3     85                 3
4     47                 2
```

```
df1["Fitness_category"].replace({1:"Poor Shape",
                                 2:"Bad Shape",
                                 3:"Average Shape",
                                 4:"Good Shape",
                                 5:"Excellent Shape"},inplace=True)
df1.head()
```

```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness
Income  \
0   KP281   18    Male         14        Single      3        4
29562
1   KP281   19    Male         15        Single      2        3
31836
2   KP281   19  Female         14     Partnered      4        3
30699
3   KP281   19    Male         12        Single      3        3
32973
4   KP281   20    Male         13     Partnered      4        2
35247

   Miles Fitness_category
0    112       Good Shape
1     75    Average Shape
2     66    Average Shape
3     85    Average Shape
4     47        Bad Shape
```

```
bins = [29000, 35000, 60000, 85000,105000]
labels = ['Low Income','Lower-middle income','Upper-Middle income',
'High income']
df1['IncomeSlab'] = pd.cut(df1['Income'],bins,labels = labels)
df1.head()
```

```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness
Income  \
0   KP281   18    Male         14        Single      3        4
29562
1   KP281   19    Male         15        Single      2        3
31836
2   KP281   19  Female         14     Partnered      4        3
30699
3   KP281   19    Male         12        Single      3        3
32973
4   KP281   20    Male         13     Partnered      4        2
35247

   Miles Fitness_category           IncomeSlab
0    112       Good Shape           Low Income
1     75    Average Shape           Low Income
2     66    Average Shape           Low Income
3     85    Average Shape           Low Income
4     47        Bad Shape  Lower-middle income
```

Observations:

1. This data set has 180 data points and have no missing value.
2. There are 3 unique products ,where KP281 is the most sold product.
3. The customer age range between 18 to 50 .
4. Majority of the customers are male.
5. Most of the customer have 16 years of education .
6. The standard daviation of income is high.

2.***Non-Graphical Analysis: Value counts and unique attributes***

```
df['Product'].nunique()
```

```
3
```

```
df['Income'].mean()
```

```
53719.57777777778
```

```
df['Gender'].value_counts()
```

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

```python
df['Product'].value_counts()
```

```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

```python
df['Education'].unique()
```

```
array([14, 15, 12, 13, 16, 18, 20, 21])
```

```python
df['MaritalStatus'].value_counts()
```

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

```python
# for unique list of products, listed in percentage
sr = df['Product'].value_counts(normalize=True)
stat = sr.map(lambda x: round(100*x,2))
stat
```

```
KP281    44.44
KP481    33.33
KP781    22.22
Name: Product, dtype: float64
```

```python
df.groupby('Product')['Fitness'].mean()
```

```
Product
KP281    2.9625
KP481    2.9000
KP781    4.6250
Name: Fitness, dtype: float64
```

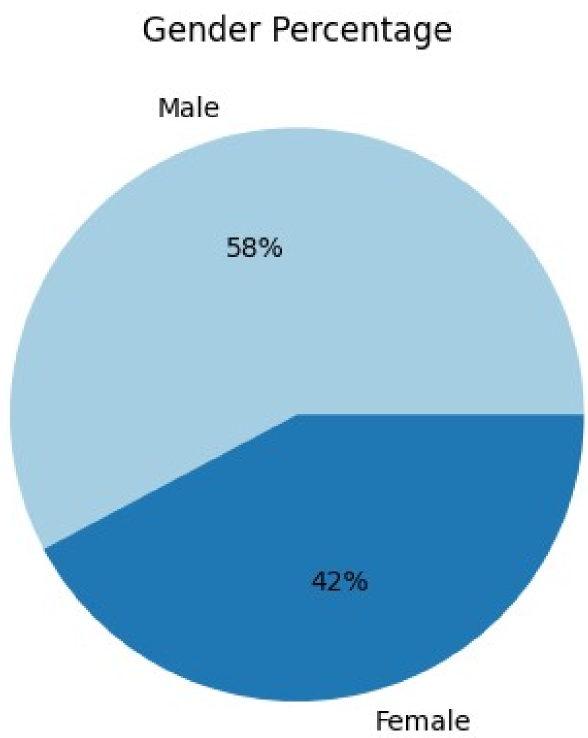
3.***Visual Analysis - Univariate & Bivariate***

1. Univariate Analysis

```python
np.round(df['Gender'].value_counts(normalize=True)*100,2)
```

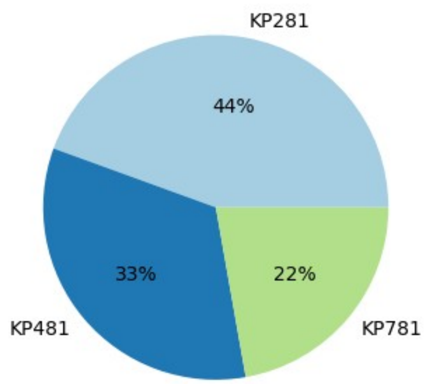
```
Male      57.78
Female    42.22
Name: Gender, dtype: float64
```

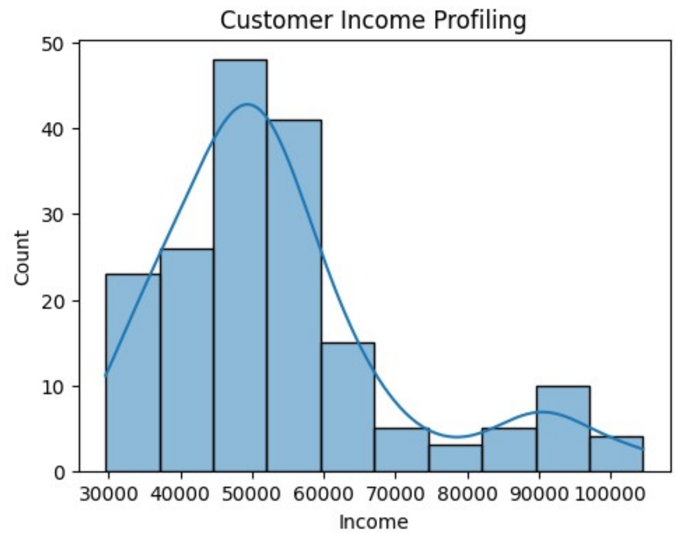
```python
gcnt=df['Gender'].value_counts()
plt.pie(gcnt, labels=df['Gender'].unique(),colors =
sns.color_palette("Paired")[0:5], autopct='%.0f%%')
plt.title('Gender Percentage')
plt.show()
```



☐**Most of the cutomers are male**

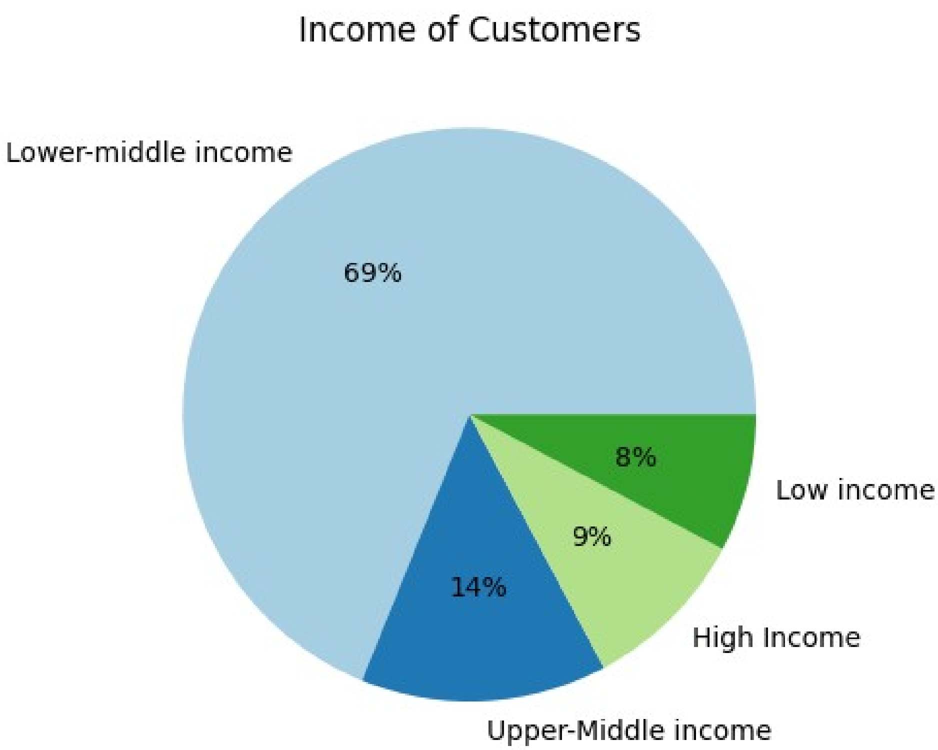
```
np.round(df['Product'].value_counts(normalize=True)*100,2)

KP281    44.44
KP481    33.33
KP781    22.22
Name: Product, dtype: float64

plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
gcnt=df['Product'].value_counts()
plt.pie(gcnt, labels=df['Product'].unique(), colors
=sns.color_palette("Paired")[0:7] ,autopct='%.0f%%')


plt.subplot(1,2,2)
sns.histplot(df["Income"], kde=True,bins=10)
plt.title('Customer Income Profiling')
plt.show()
```





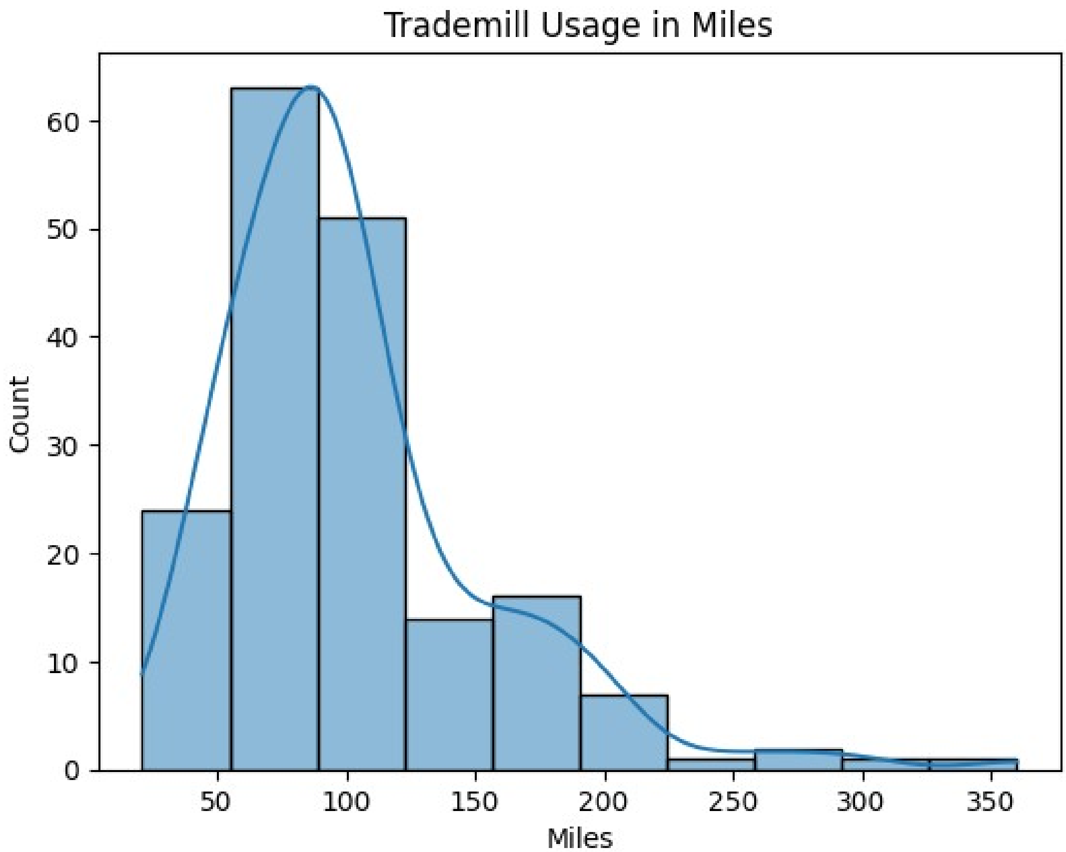
- KP281 is the most popular producrt
- Most of the customer have income between 35000-60000

```
np.round(df['IncomeSlab'].value_counts(normalize=True)*100,2)

Lower-middle income    68.89
Upper-Middle income    13.89
High income             9.44
Low Income              7.78
Name: IncomeSlab, dtype: float64

gcnt=df1['IncomeSlab'].value_counts()
plt.pie(gcnt, labels=['Lower-middle income','Upper-Middle
income','High Income' ,'Low income'], colors
=sns.color_palette("Paired")[0:7] ,autopct='%.0f%%')
plt.title('Income of Customers')
plt.show()
```



□**Most of the customer belong to lower-middle income category**

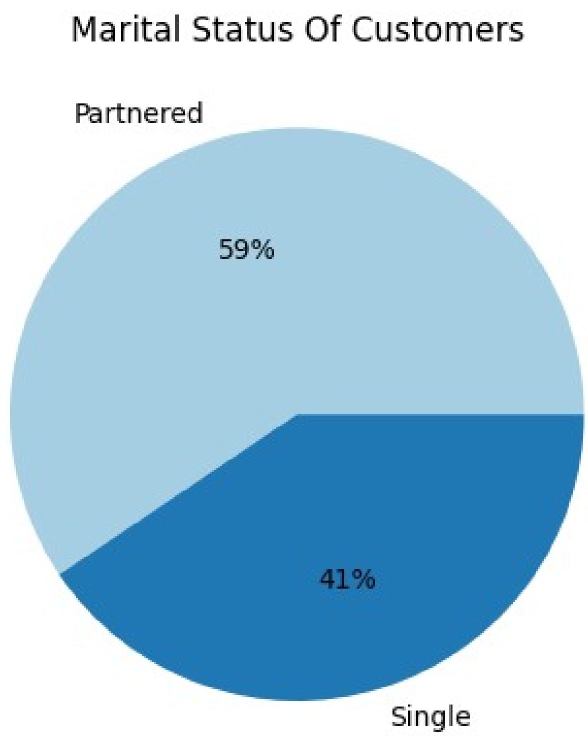
```
sns.histplot(df["Miles"], kde=True,bins=10)
plt.title('Trademill Usage in Miles')
plt.show()
```




```
np.round(df['MaritalStatus'].value_counts(normalize=True)*100,2)
```

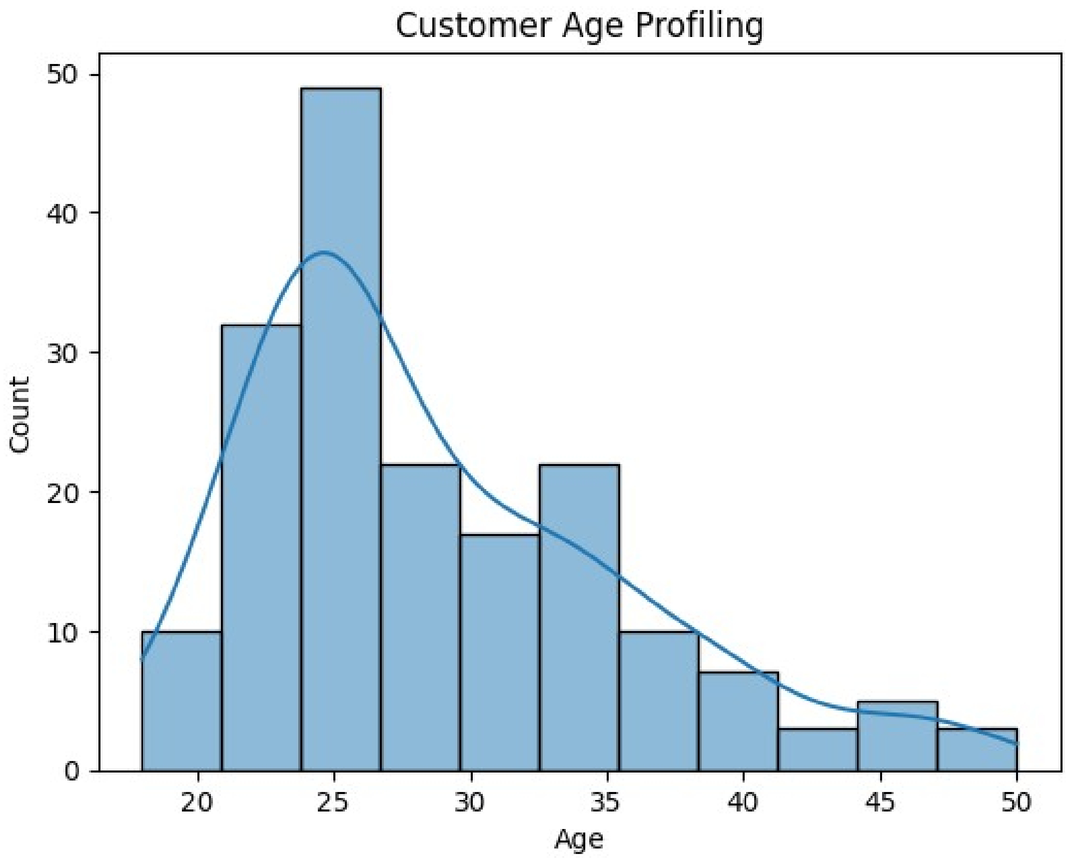
```
Partnered    59.44
Single       40.56
Name: MaritalStatus, dtype: float64
```

```
gcnt=df['MaritalStatus'].value_counts()
plt.pie(gcnt, labels=['Partnered','Single'], colors
=sns.color_palette("Paired")[0:7] ,autopct='%.0f%%')
plt.title('Marital Status Of Customers')
plt.show()
```



☐**Marital Status of most of the customers is Partnered**

```
sns.histplot(df["Age"], kde=True)
plt.title('Customer Age Profiling')
plt.show()
```



☐**most of the customers belong to the age group between 20-30**

```
#Age of the most of the customer for KP281
df1[df1['Product']=='KP281']['Age'].median()
```

```
26.0
```

```
#Age of the most of the customer for KP481
df1[df1['Product']=='KP481']['Age'].median()
```

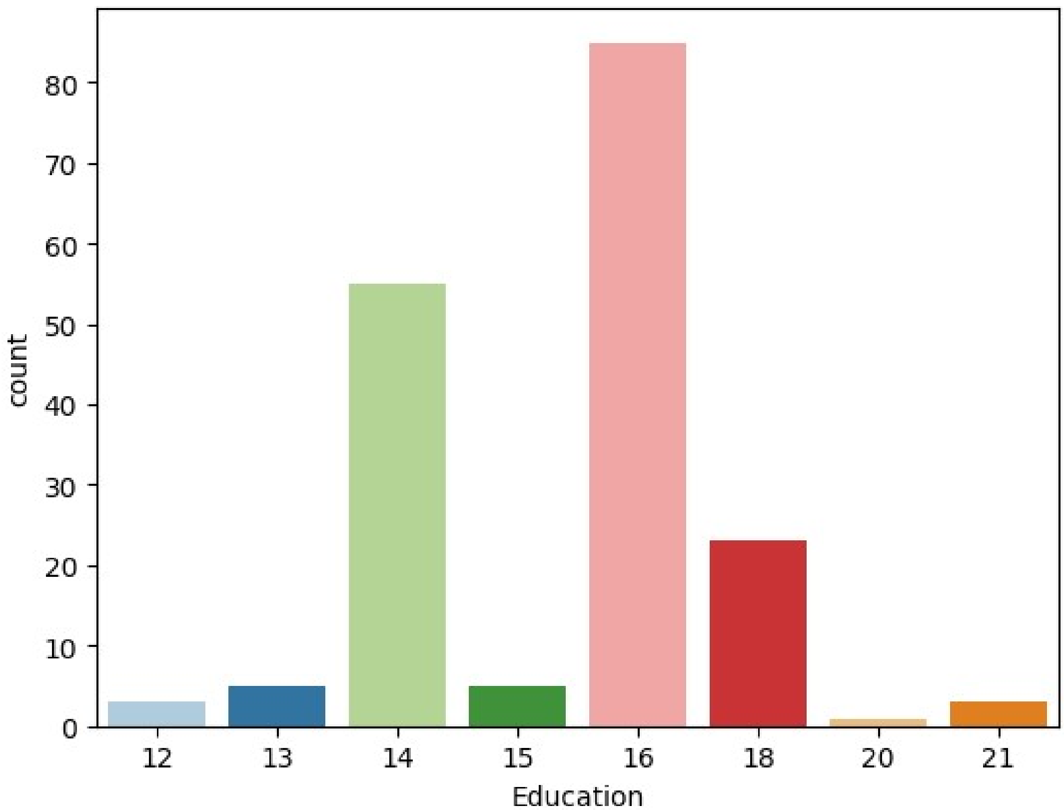
```
26.0
```

```
#Age of the most of the customer for KP281
df1[df1['Product']=='KP781']['Age'].median()
```

```
27.0
```

```
sns.countplot(df,x='Education',palette = "Paired")
```

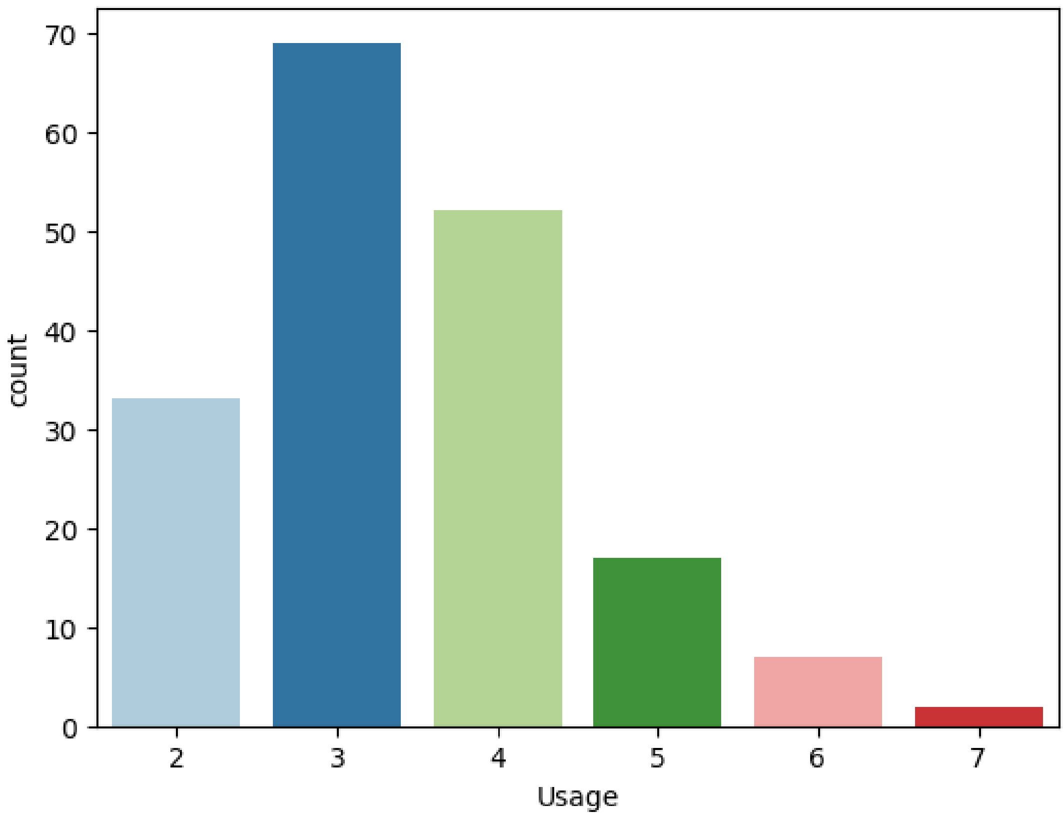
```
<Axes: xlabel='Education', ylabel='count'>
```



□**most of the customers have 16 years of education**

```
sns.countplot(df,x='Usage',palette = "Paired")
```

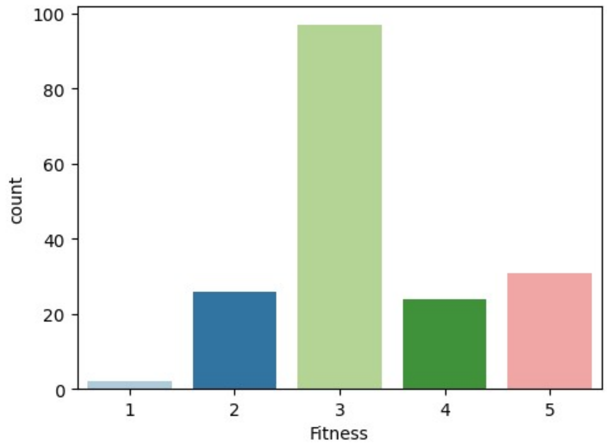
```
&lt;Axes: xlabel='Usage', ylabel='count'&gt;
```

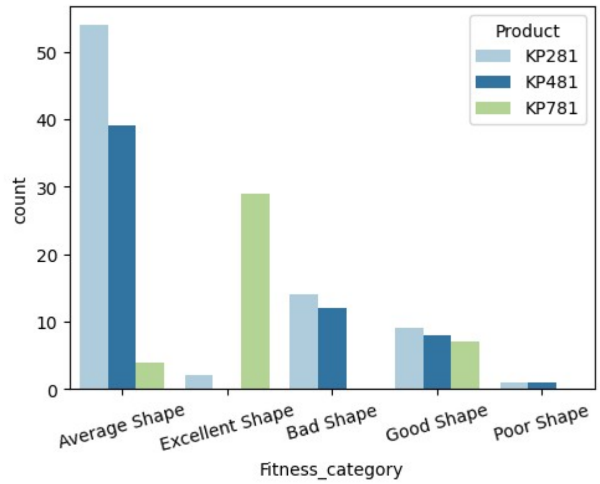


☐**most of the customers use the product 3 to 4 times in a week**

Fitness level vs Product

```
plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
sns.countplot(df,x='Fitness',palette = "Paired")



plt.subplot(1,2,2)
sns.countplot(data=df1,x='Fitness_category',hue='Product',
order=df1['Fitness_category'].value_counts(ascending=False).index,pale
tte = "Paired")
plt.xticks(rotation=15)
plt.show()
```



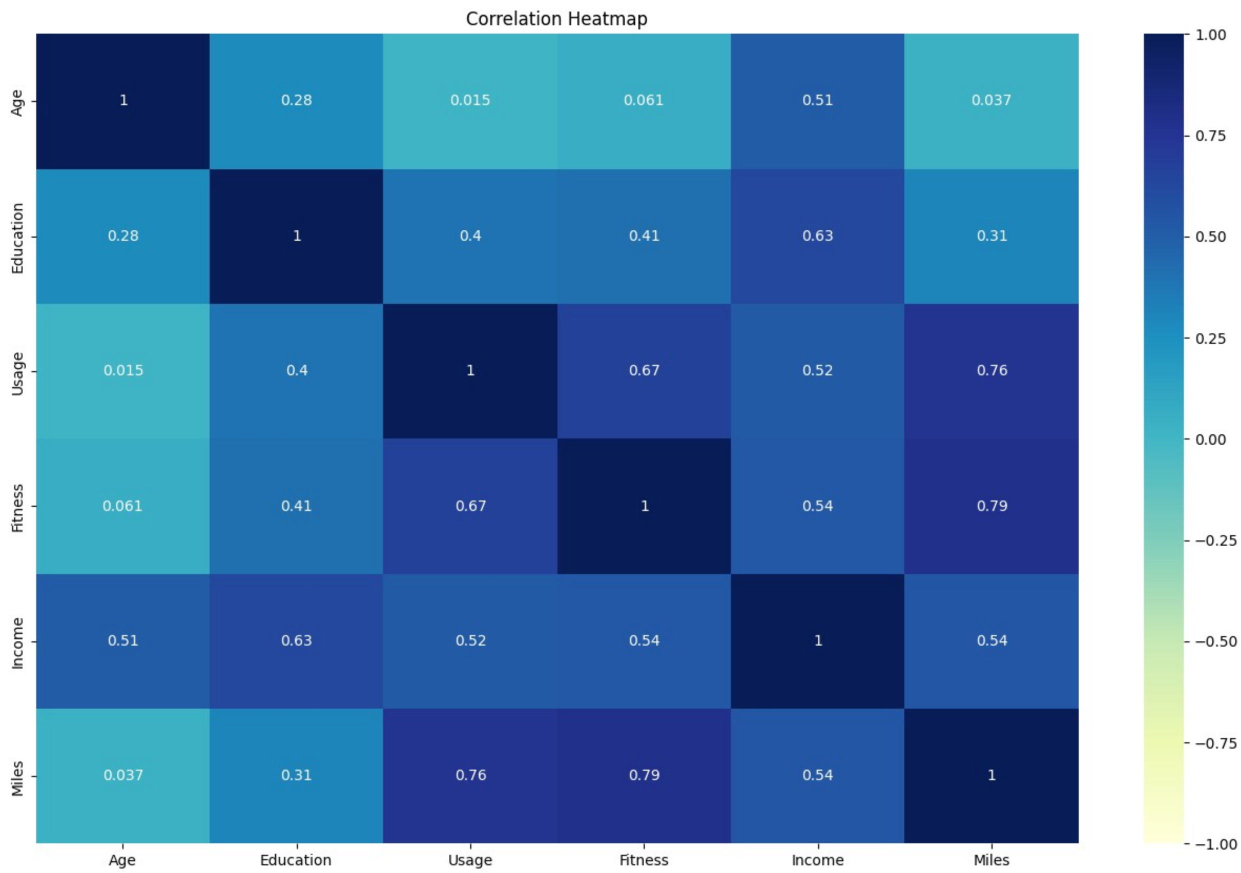


☐**Insights**

1. **most of the customers have level 3 fitness**
2. **Customer, who use KP781 are more likely to have Excellent fitness level and bodyshape**
1. **:Bi-Variant Analysis:**

```
plt.figure(figsize = (16, 10))
sns.heatmap(df1.corr(), annot=True, vmin=-1, vmax = 1,cmap="YlGnBu")
plt.title('Correlation Heatmap')
plt.show()
```

```
<ipython-input-32-88fa5004e037>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  sns.heatmap(df1.corr(), annot=True, vmin=-1, vmax = 1,cmap="YlGnBu")
```



From the above heatmap we can observe that:

Correlation between Age and Miles is 0.04

Correlation between Education and Income is 0.63

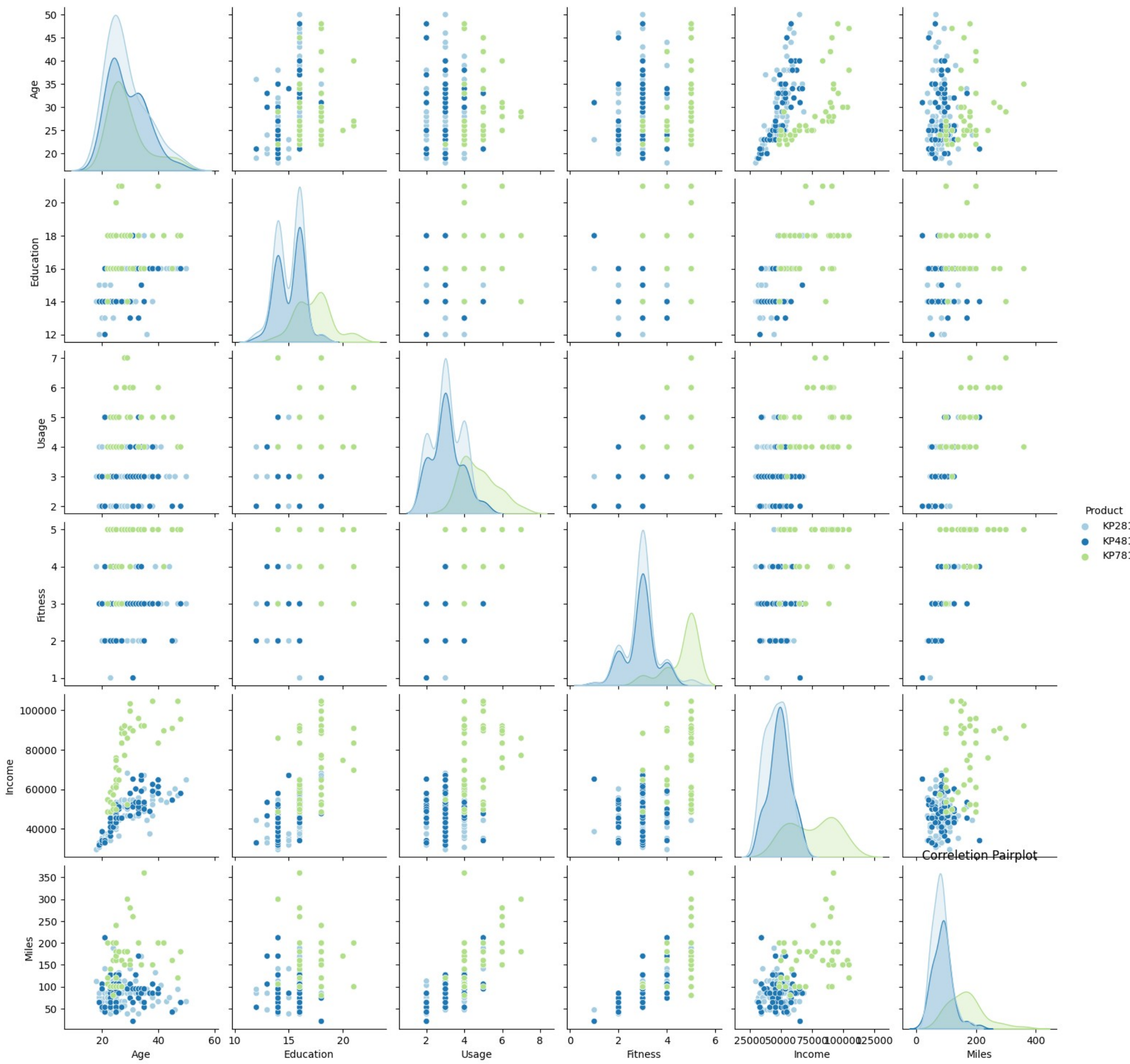
Correlation between Usage and Fitness is 0.67

Correlation between Fitness and Age is 0.06

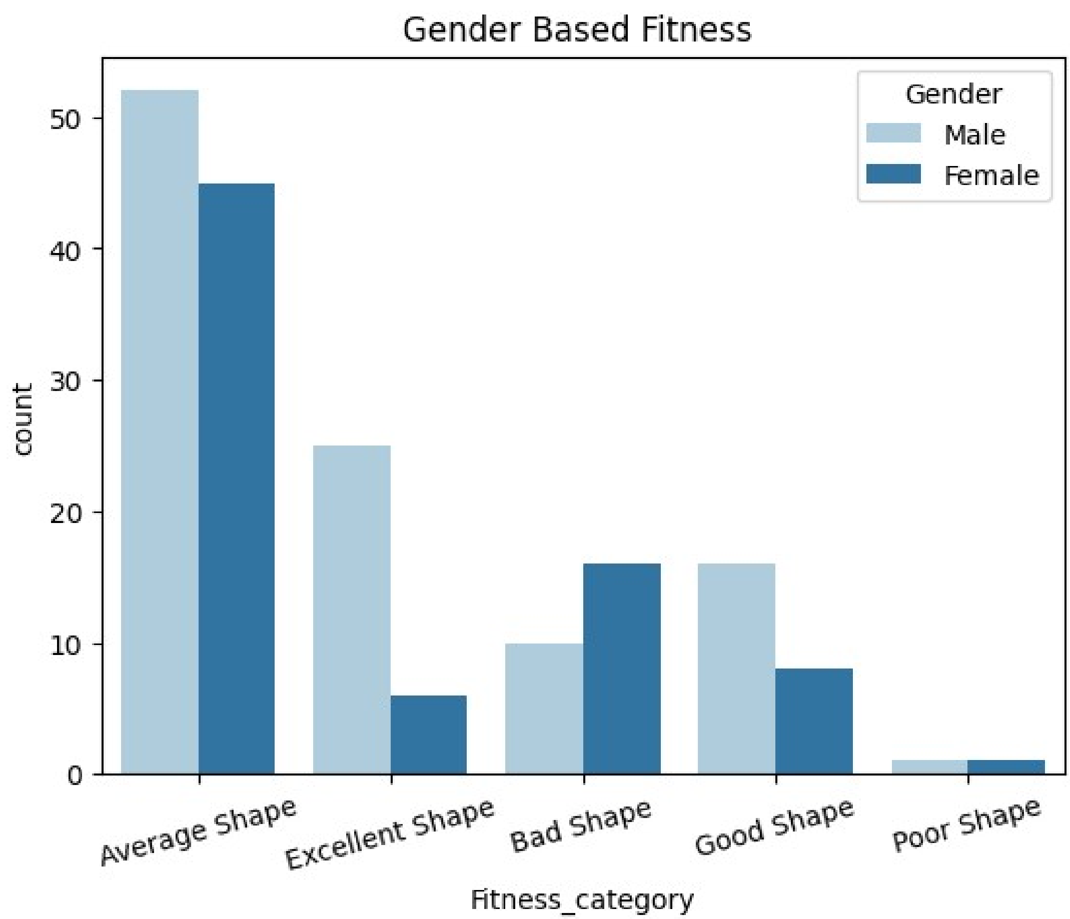
Correlation between Income and Usage is 0.52

Correlation between Miles and Age is 0.03

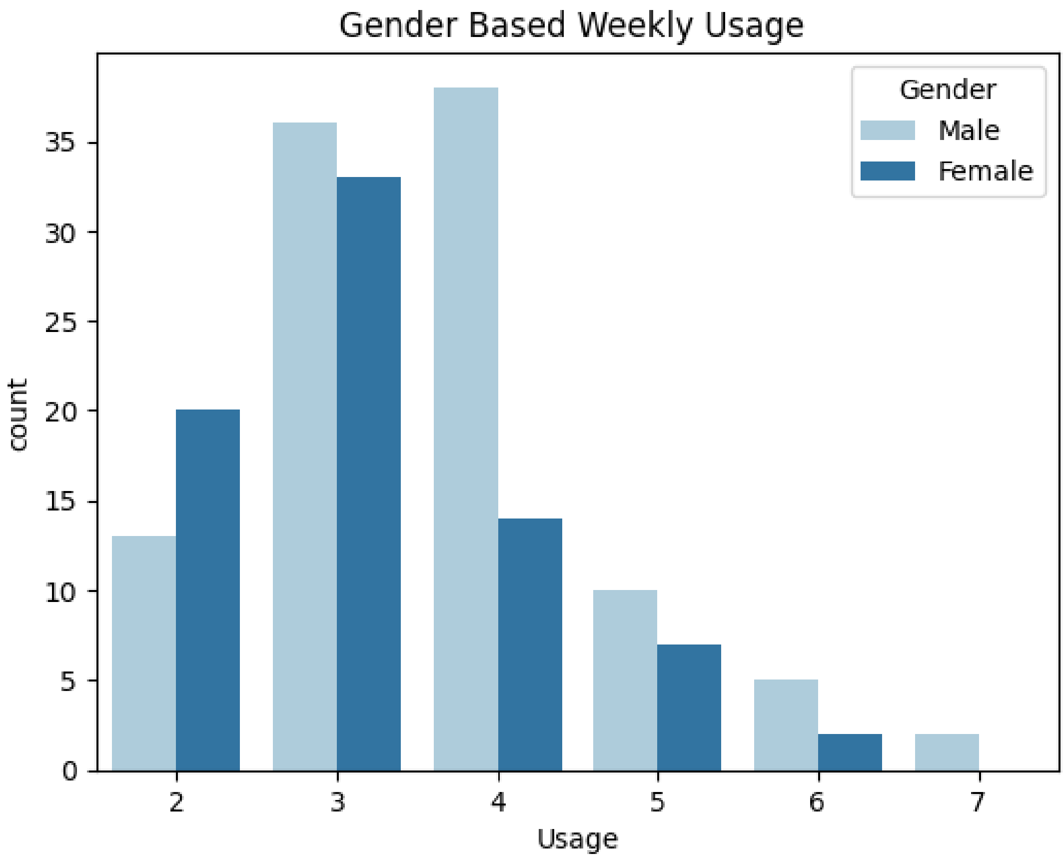
```
sns.pairplot(df1, hue='Product',palette = "Paired")
plt.title('Correletion Pairplot')
plt.show()
```




```
sns.countplot(data=df1,x='Fitness_category',hue='Gender',order=df1['Fi
tness_category'].value_counts(ascending=False).index,palette =
"Paired")
plt.title('Gender Based Fitness ')
plt.xticks(rotation=15)
plt.show

<function matplotlib.pyplot.show(close=None, block=None)>
```



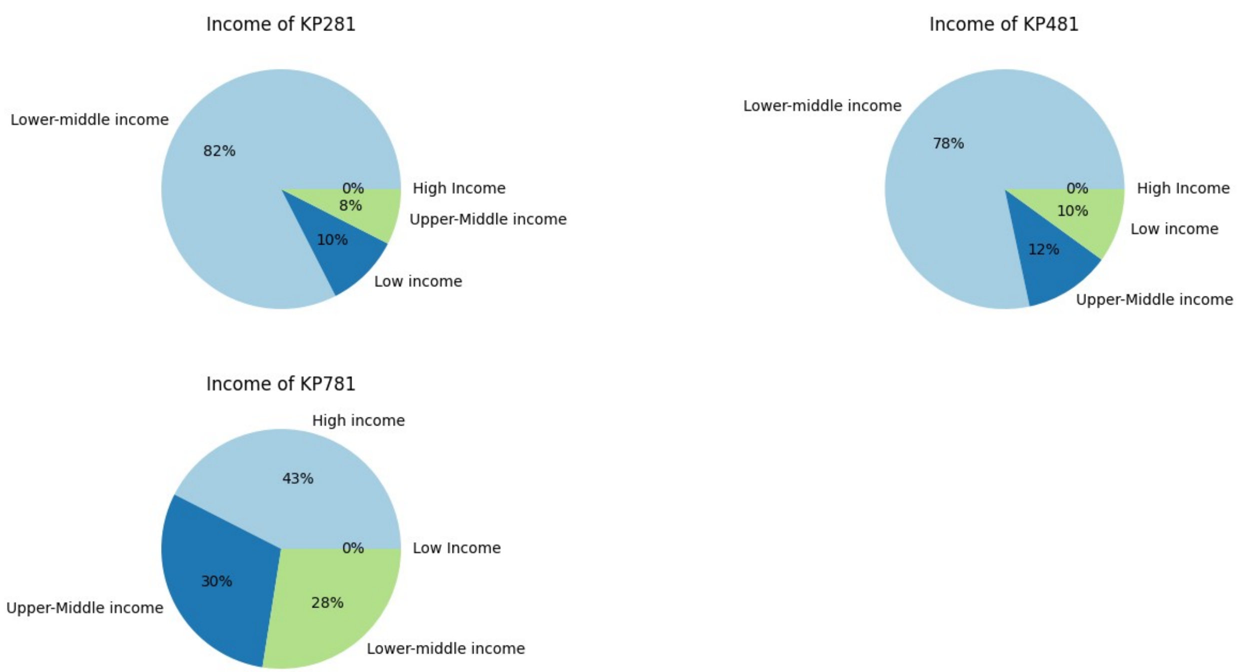

```
sns.countplot(data=df,x='Usage',hue='Gender',palette = "Paired")
plt.title('Gender Based Weekly Usage ')
plt.show()
```




```
np.round(df1[df1['Product']=="KP281"]
['IncomeSlab'].value_counts(normalize=True)*100,2)

Lower-middle income    82.5
Low Income             10.0
Upper-Middle income     7.5
High income             0.0
Name: IncomeSlab, dtype: float64

np.round(df1[df1['Product']=="KP481"]
['IncomeSlab'].value_counts(normalize=True)*100,2)

Lower-middle income    78.33
Upper-Middle income    11.67
Low Income             10.00
High income             0.00
Name: IncomeSlab, dtype: float64

np.round(df1[df1['Product']=="KP781"]
['IncomeSlab'].value_counts(normalize=True)*100,2)

High income            42.5
Upper-Middle income    30.0
Lower-middle income    27.5
```

```
Low Income              0.0
Name: IncomeSlab, dtype: float64
```

```
plt.figure(figsize=(16,8))
plt.subplot(2,2,1)
gcnt=df1[df1['Product']=="KP281"]['IncomeSlab'].value_counts()
plt.pie(gcnt, labels=['Lower-middle income','Low income','Upper-Middle 
income','High Income' ], colors =sns.color_palette("Paired")
[0:7] ,autopct='%.0f%%')
plt.title('Income of KP281')



plt.subplot(2,2,2)
gcnt=df1[df1['Product']=="KP481"]['IncomeSlab'].value_counts()
plt.pie(gcnt, labels=['Lower-middle income','Upper-Middle income','Low 
income','High Income' ], colors =sns.color_palette("Paired")
[0:7] ,autopct='%.0f%%')
plt.title('Income of KP481')


plt.subplot(2,2,3)
gcnt=df1[df1['Product']=="KP781"]['IncomeSlab'].value_counts()
plt.pie(gcnt, labels=['High income','Upper-Middle income','Lower-
middle income','Low Income' ], colors =sns.color_palette("Paired")
[0:7] ,autopct='%.0f%%')
plt.title('Income of KP781')
plt.show()
```

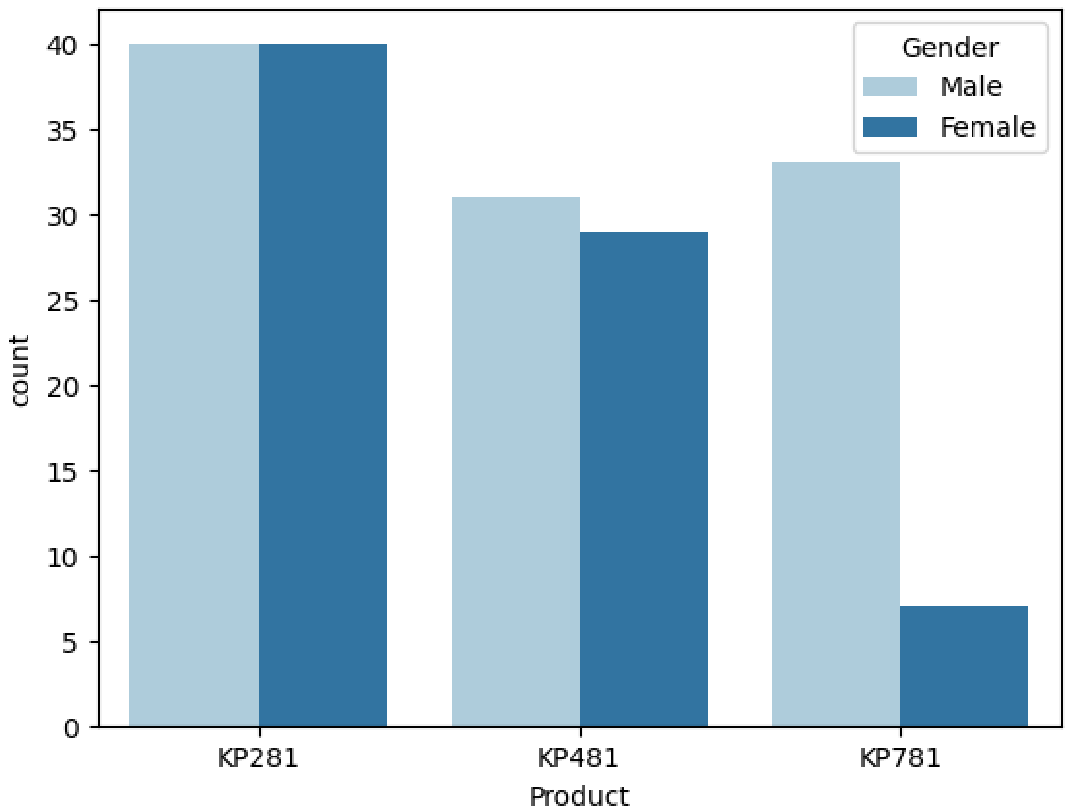


```
df1.columns
```

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

```
sns.countplot(data=df,x='Product',hue='Gender',palette = "Paired")
```

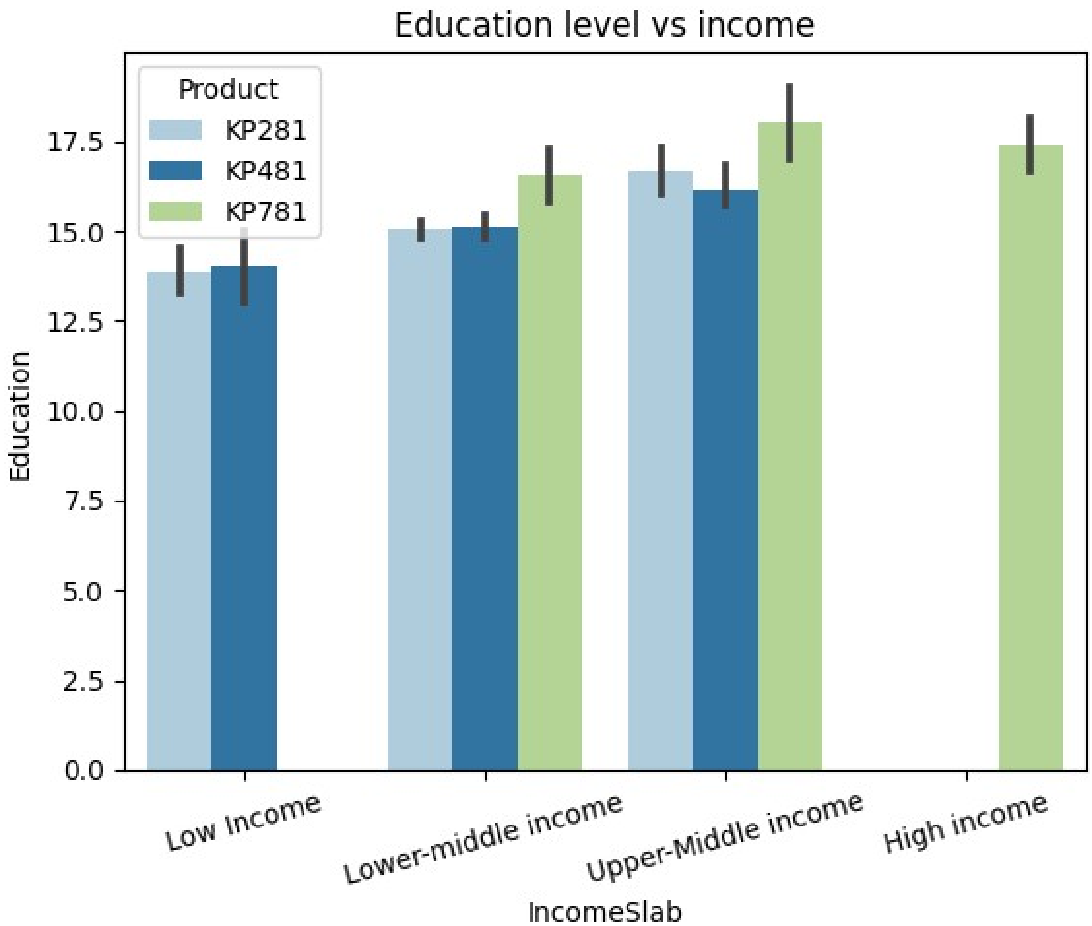
```
<Axes: xlabel='Product', ylabel='count'>
```



KP781 is more popular mong men KP281 is equally popular among both male and females

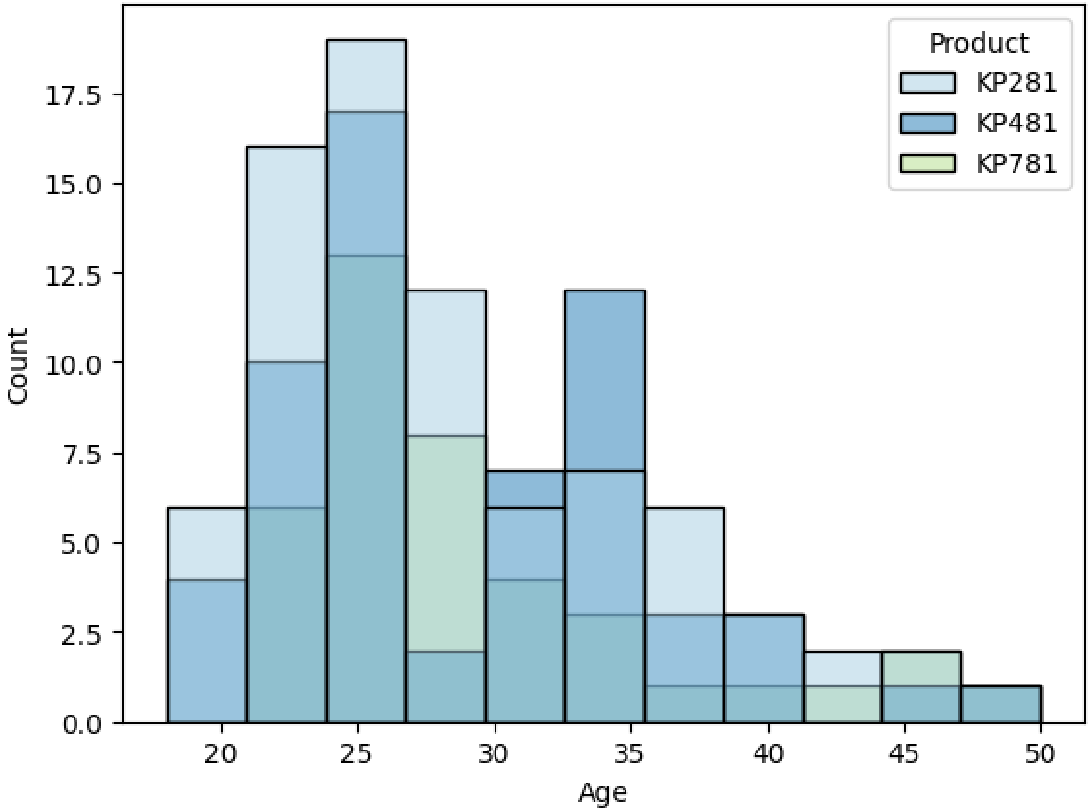
```
sns.barplot(data=df1,x='IncomeSlab',y='Education',hue='Product',palett
e = "Paired")
plt.xticks(rotation=15)
plt.title("Education level vs income")
```

```
Text(0.5, 1.0, 'Education level vs income')
```




```
sns.histplot(data=df1,x='Age',hue='Product',palette = "Paired")
```

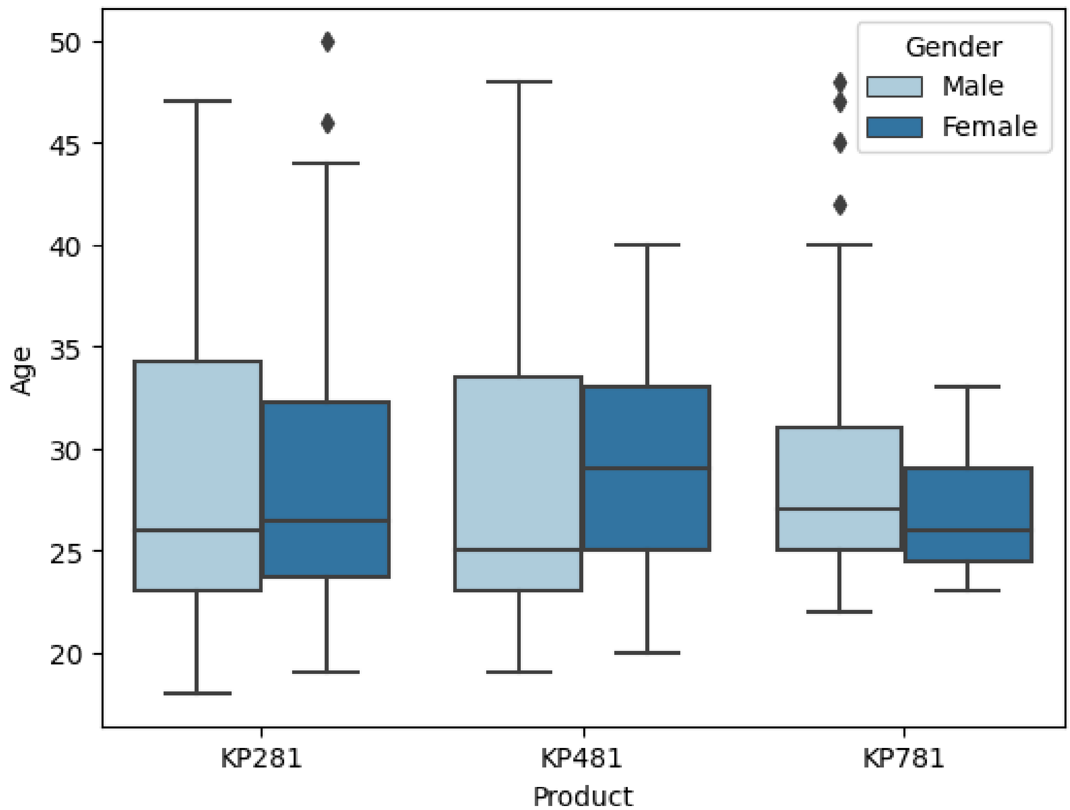
```
<Axes: xlabel='Age', ylabel='Count'>
```



Highly educated customers usually earn more

```
sns.boxplot(data=df1,y='Age',x='Product',hue='Gender',palette =
"Paired")
```

```
<Axes: xlabel='Product', ylabel='Age'>
```

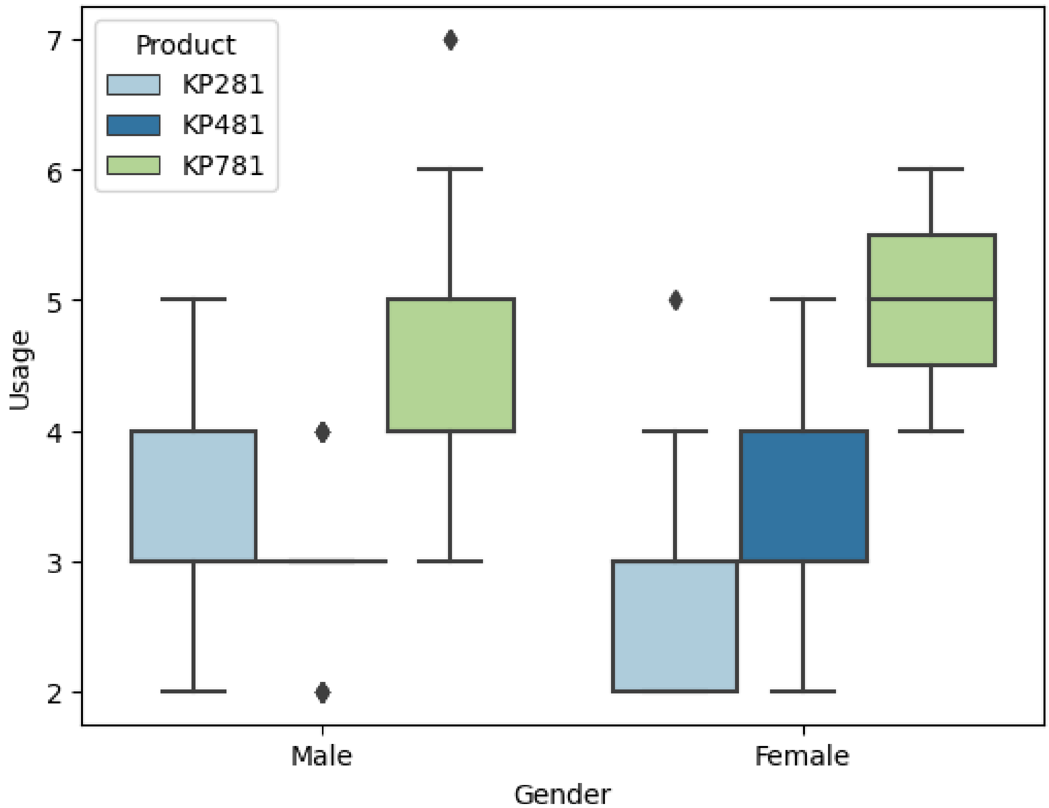


Customers purchasing products KP281 & KP481 are having same Age median value.

Customers whose age lies between 25-30, are more likely to buy KP781 product.

```
sns.boxplot(data=df1,y='Usage',hue='Product',x='Gender',palette =
"Paired")
```

```
<Axes: xlabel='Gender', ylabel='Usage'>
```

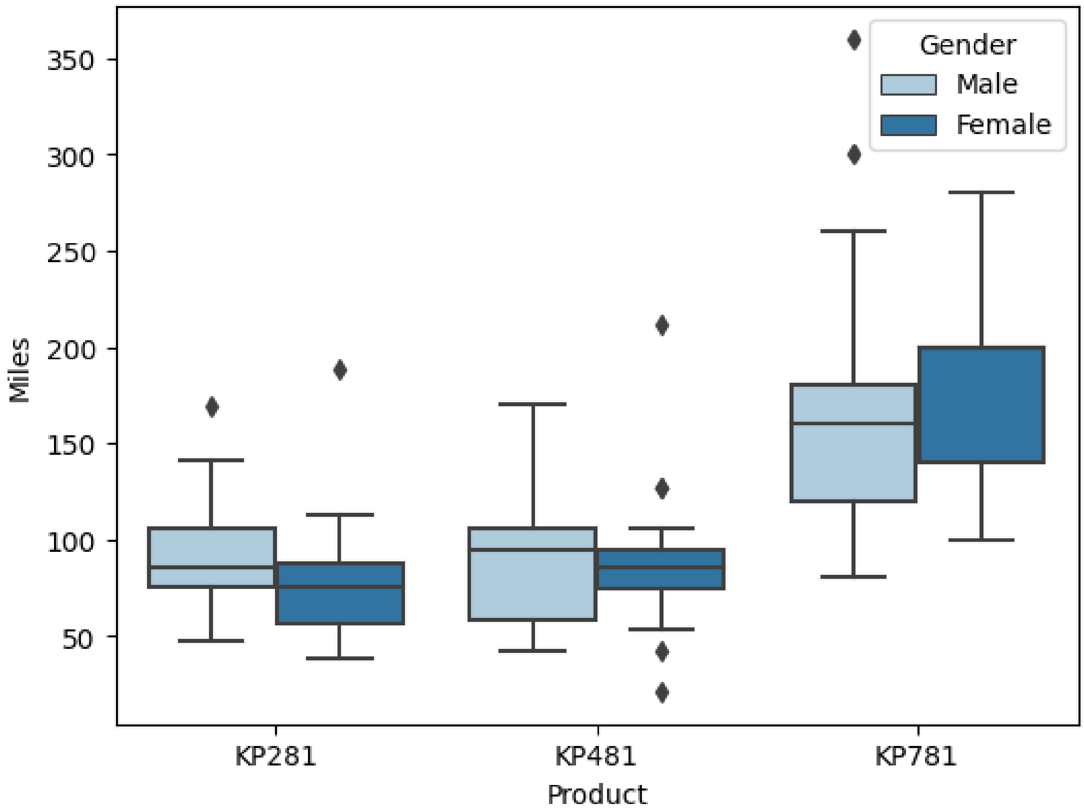


The more the customer is fit (fitness >= 3), higher the chances of the customer to purchase the KP781 product

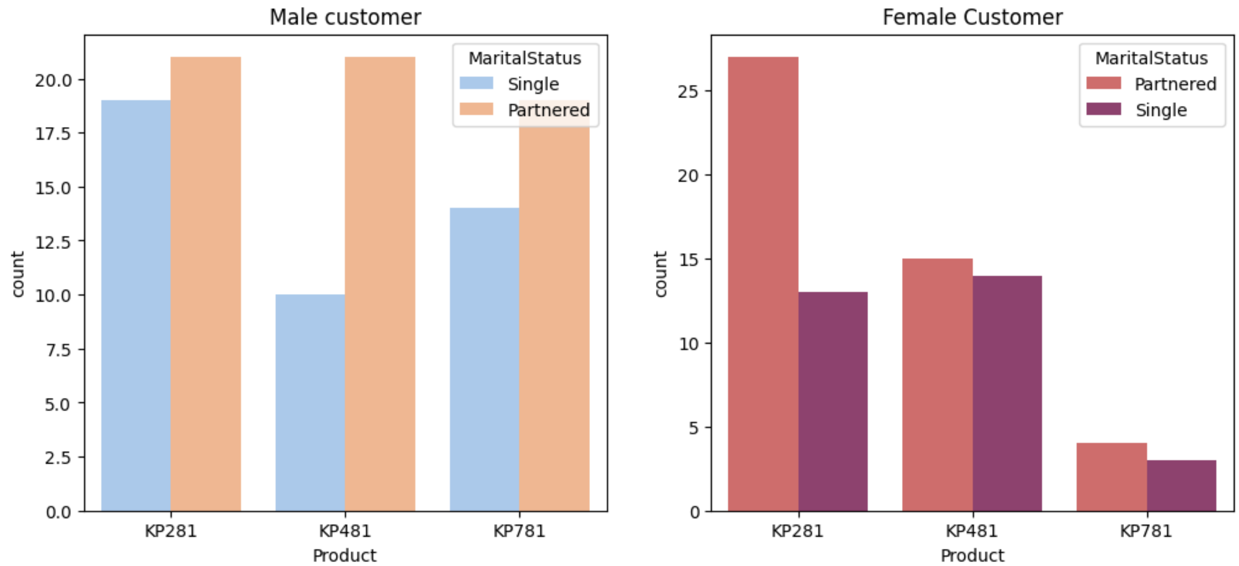
Females planning to use treadmill 3-4 times a week, are more likely to buy KP481 product

```
sns.boxplot(data=df1,y='Miles',hue='Gender',x='Product',palette =
"Paired")
```

```
<Axes: xlabel='Product', ylabel='Miles'>
```




```
dff=df[df['Gender']=='Female']
dfm=df[df['Gender']=='Male']
fig,ax=plt.subplots(nrows=1, ncols=2, figsize=(12,5))
sns.countplot(data=dfm, x='Product',
hue='MaritalStatus',ax=ax[0],palette="pastel"); ax[0].set_title('Male
customer')
sns.countplot(data=dff, x='Product',
hue='MaritalStatus',ax=ax[1],palette="flare"); ax[1].set_title('Female
Customer')

Text(0.5, 1.0, 'Female Customer')
```



If the customer expects to walk/run greater than 120 Miles per week, it is more likely that the customer will buy KP781 product.

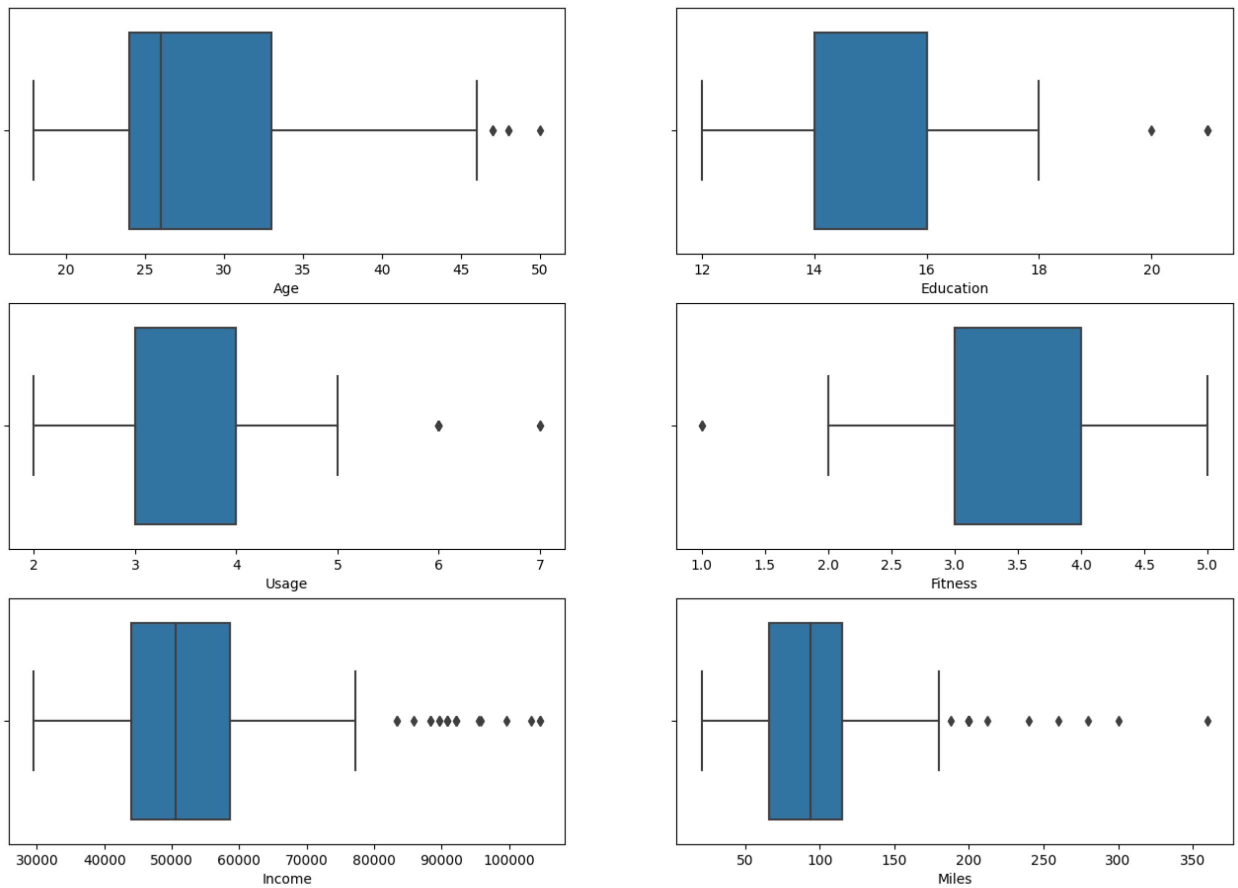
1. **Missing Value And Outlier Dectection**

```
df.isnull().sum()

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

This database has no missing values

```
plt.figure(figsize=(16,11))

plt.subplot(3,2,1)
sns.boxplot(data=df, x="Age")

plt.subplot(3,2,2)
sns.boxplot(data=df, x="Education")

plt.subplot(3,2,3)
sns.boxplot(data=df, x="Usage")
```

```
plt.subplot(3,2,4)
sns.boxplot(data=df, x="Fitness")

plt.subplot(3,2,5)
sns.boxplot(data=df, x="Income")

plt.subplot(3,2,6)
sns.boxplot(data=df, x="Miles")


plt.show()
```



Obervation

1. Age, Education and Usage are having very few outliers.
2. While Income and Miles are having more outliers.

Probability Calculations:

```
#probability of purchase of diffent products based on Income level
```

```
np.round(pd.crosstab(index=df1['Product'],
columns=[df1['IncomeSlab']],margins=True,normalize='columns') *100,2)

IncomeSlab  Low Income  Lower-middle income  Upper-Middle income  High
income  \
Product

KP281            57.14                53.23                 24.0
0.0
KP481            42.86                37.90                 28.0
0.0
KP781             0.00                 8.87                 48.0
100.0

IncomeSlab    All
Product
KP281       44.44
KP481       33.33
KP781       22.22

#probability of purchase of diffent products based on Gender
np.round((pd.crosstab([df.Product],df.Gender,margins=True,normalize="c
olumns"))*100,2)

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

#probability of purchase of diffent products based on fitness level

np.round(pd.crosstab(index=df1['Product'],
columns=df1['Fitness'],margins=True,normalize=True) *100,2)

Fitness     1      2      3      4      5     All
Product
KP281    0.56   7.78  30.00   5.00   1.11   44.44
KP481    0.56   6.67  21.67   4.44   0.00   33.33
KP781    0.00   0.00   2.22   3.89  16.11   22.22
All      1.11  14.44  53.89  13.33  17.22  100.00
```

Probability of people who has Fitness 3 purchase treadmill is 53.88%.

P(KP281|Fitness=3) = 30.00%

P(KP481|Fitness=3) = 21.66%

P(KP781|Fitness=3) = 2.22%

Probability of people who has Fitness 4 purchase treadmill is 13.33%.

P(KP281|Fitness=4) = 5.00%

P(KP481|Fitness=4) = 4.44%

P(KP781|Fitness=4) = 3.88%

Probability of people who has Fitness 5 purchase treadmill is 17.22%.

P(KP281|Fitness=5) = 1.11%

P(KP481|Fitness=5) = 0%

P(KP781|Fitness=5) = 16.11%

```
#probability of purchase based on use of trademill in days per week

np.round(pd.crosstab(index=df['Usage'],columns=df['Product'],margins=True,normalize= True)*100,2)
```

Product Usage	KP281	KP481	KP781	All
2	10.56	7.78	0.00	18.33
3	20.56	17.22	0.56	38.33
4	12.22	6.67	10.00	28.89
5	1.11	1.67	6.67	9.44
6	0.00	0.00	3.89	3.89
7	0.00	0.00	1.11	1.11
All	44.44	33.33	22.22	100.00

```
#probability of purchase for females based on marital staus

np.round(pd.crosstab(index=dfm['MaritalStatus'],columns=df['Product'],
margins=True,normalize= True)*100,2)
```

Product MaritalStatus	KP281	KP481	KP781	All
Partnered	20.19	20.19	18.27	58.65
Single	18.27	9.62	13.46	41.35
All	38.46	29.81	31.73	100.00

```
#probability of purchase for males based on marital staus

np.round(pd.crosstab(index=dff['MaritalStatus'],columns=df['Product'],
margins=True,normalize= True)*100,2)
```

Product MaritalStatus	KP281	KP481	KP781	All
Partnered	35.53	19.74	5.26	60.53
Single	17.11	18.42	3.95	39.47
All	52.63	38.16	9.21	100.00

Insight: -

Probability of purchase for the people who use treadmill 3 times a week is 38.33%.

P(KP281|Usage=3) = 20.55%

P(KP481|Usage=3) = 17.22%

P(KP781|Usage=3) = 0.55%

Probability of purchase for the people who use treadmill 3 times a week is 28.88%.

P(KP281|Usage=4) = 12.22%

P(KP481|Usage=4) = 6.66%

P(KP781|Usage=4) = 10.00%

Business Insights

Customer Profiling

KP281 customer's profile:

- KP281 is the highest selling product
- Fitness level under 3 are most likely to use KP281
- Most of the customers are low to mid income
- Females are more likely to purchase
- There is a widest range of age group of customers for this product
- These customers runs less than 120 Miles per week

KP481 customer's profile:

- This is the second highest selling product
- Customers having fitness level under 4
- Customers purchasing products KP281 & KP481 are having same Age median value.
- These customers runs less than 120 Miles per week

KP781 customer's profile:

- This is the least selling product because of the price
- most of the customers are male
- Fitness level is above 3
- Highly educated having more than 16 years of education
- These customers runs more than 120 miles per week
- Income level is high
- The median age of these customers is higher

Overall Insights

1. 58% Customers are male
2. KP281 is the most popular product

3. 69% customers earns less than 35000
4. KP781 customers are mostly male and have a good fitness level and have income more than 60000
5. Among the users, 44.44% prefer using the KP281 treadmill, while 33.33% opt for the KP481 treadmill, and only 22.22% of users favor the KP781 treadmill.
6. The trend observed among both married and single customers reflects that KP281, being an entry-level treadmill, is the most frequently purchased option, while KP781, due to its higher cost, remains the least popular choice for both customer groups.
7. The median age of customers is around 27 years

Business Recommendations

1. There is a 11% deference of sales between KP281 and KP481. The sales of KP481 can be increased by promoting this more among partnered customers with ads with famous celebrity couples as this is more perfered by partnered customers
2. EMI options can convert potential KP281 to KP481 customer
3. Based on marital status on social media platforms, more targeted ads could be pushed.
4. If the product is sold in e-commerce platform, the products can be recommended based of customers purchasing pattern and spending.
5. To increase the sales of KP781 ,promotional offer can be given during festivals and holidays. The positive reviews will increase the sales later. The EMI options can be more easy to gain more customers of less income than the avarage user of this product.
6. Fitness equipment sales are comparatively low among females, fitness marketing campaign can be launched to encourage exercise
7. KP781 treadmill should be marketed for professionals and athletes. So,ads with athletes and sponsor in sports events can give it a sales boost