AEROFIT PROJECT

In [1]:
```
#Importing the required libraries and importing data
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv
df.head()
```

Out[1]:

	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

In [3]:
```
#info of the data
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
```

In [4]:
```
#shape of the data
df.shape
```

Out[4]:
```
(180, 9)
```

- Number of rows: 180
- Number of columns: 9

In [5]:
```
#Unique no of values for each columns
df.nunique()
```

Out[5]:
```
Product           3
Age              32
Gender            2
Education         8
MaritalStatus     2
Usage             6
Fitness           5
Income           62
Miles            37
dtype: int64
```

In [6]:
```
#changing the object datatype to catagory datatype
df=df.astype({"Product":"category","Gender":"category","MaritalStatus":"category"})
```

In [7]:
```
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    category
 1   Age            180 non-null    int64   
 2   Gender         180 non-null    category
 3   Education      180 non-null    int64   
 4   MaritalStatus  180 non-null    category
 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: category(3), int64(6)
memory usage: 9.5 KB
```

In [8]:
```
#Checking the missing values in each column
df.isna().sum()
```

Out[8]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

In [9]:
```
#Statistics of all the integer datatype columns
df.describe()
```

Out[9]:

	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

Observations:

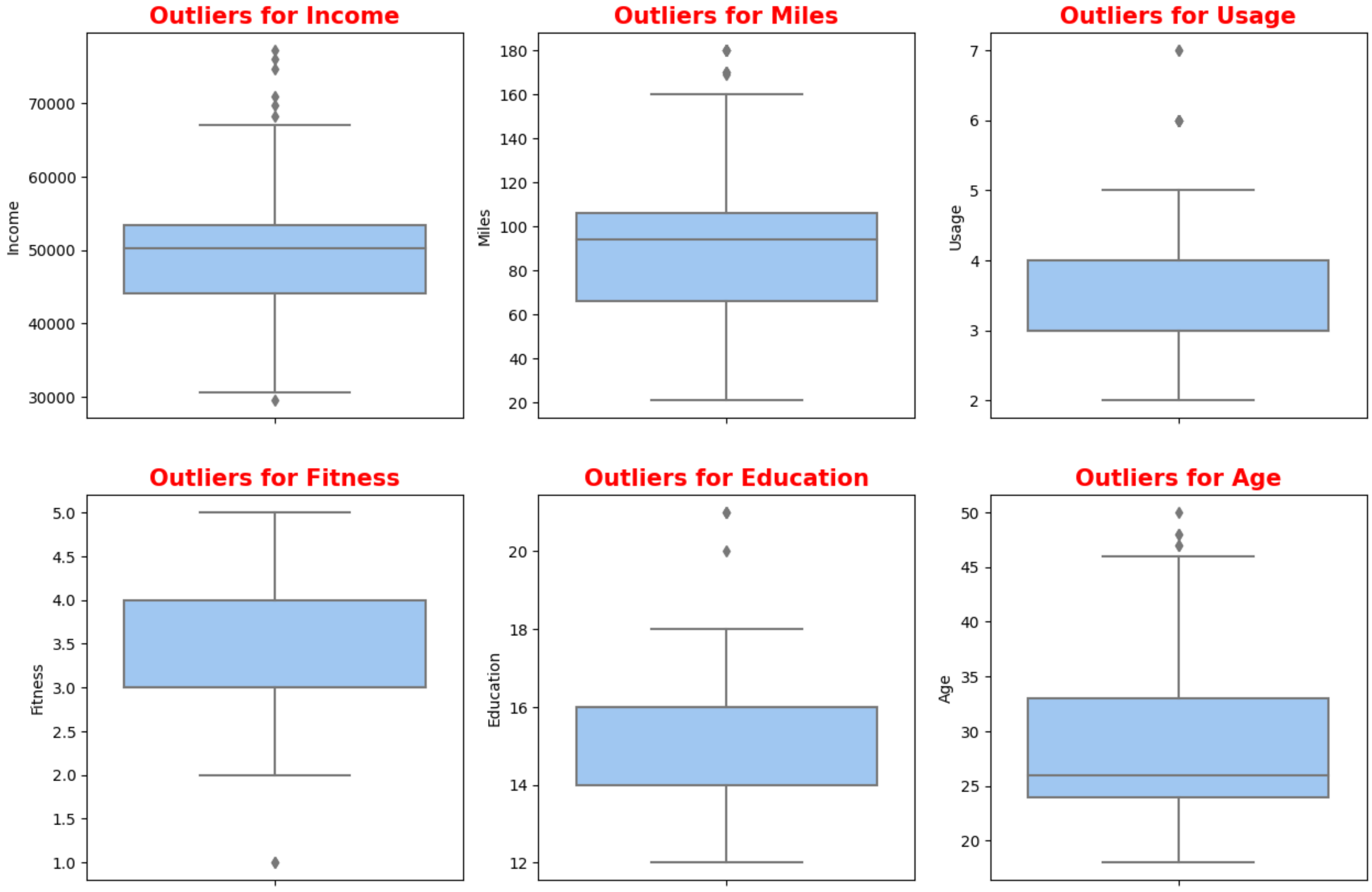
- There are no missing values in the data.
- There are 3 unique products in the dataset KP281,KP481,KP781.
- KP281 is the most frequent product.
- Minimum & Maximum age of the person is 18 & 50, mean is 28.79 and 75% of persons have age less than or equal to 33.
- Most of the people are having 16 years of education i.e. 75% of persons are having education <= 16 years.
- Out of 180 data points, 104's gender is Male and rest are the female.
- Standard deviation for Income & Miles is very high. These variables might have the outliers in it.

In [10]:
```
#Checking the difference between mean and median
print(np.percentile(df["Age"],50)-df["Age"].mean())
print(np.percentile(df["Education"],50)-df["Education"].mean())
print(np.percentile(df["Usage"],50)-df["Usage"].mean())
print(np.percentile(df["Fitness"],50)-df["Fitness"].mean())
print(np.percentile(df["Income"],50)-df["Income"].mean())
print(np.percentile(df["Miles"],50)-df["Miles"].mean())
```

```
-2.788888888888888
0.42777777777777715
-0.4555555555555557
-0.3111111111111109
-3123.0777777777766
-9.194444444444443
```

In [58]:
```
#Checking all the outliers of each column
with plt.style.context("seaborn-v0_8-pastel"):
  plt.figure(figsize=(15,10))
  plt.subplot(2,3,1)
  sns.boxplot(y="Income",data=df)
  plt.title("Outliers for Income",fontsize=15,weight="bold",color="r")
  plt.subplot(2,3,2)
  sns.boxplot(y="Miles",data=df)
  plt.title("Outliers for Miles",fontsize=15,weight="bold",color="r")
  plt.subplot(2,3,3)
  sns.boxplot(y="Usage",data=df)
  plt.title("Outliers for Usage",fontsize=15,weight="bold",color="r")
  plt.subplot(2,3,4)
  sns.boxplot(y="Fitness",data=df)
  plt.title("Outliers for Fitness",fontsize=15,weight="bold",color="r")
  plt.subplot(2,3,5)
  sns.boxplot(y="Education",data=df)
  plt.title("Outliers for Education",fontsize=15,weight="bold",color="r")
  plt.subplot(2,3,6)
  sns.boxplot(y="Age",data=df)
  plt.title("Outliers for Age",fontsize=15,weight="bold",color="r")

#plt.text(75,0.05, '53719', fontsize=50,ha="center")
#sns.heatmap(df.corr(),annot="True")
```



In [12]:
```
#Outlier treatment
df["Miles"]=np.where(df["Miles"]>(114.75+(1.5*(114.75-66))),np.percentile(df["Miles"],50),df["Miles"])
df["Income"]=np.where(df["Income"]>(58668+(1.5*(58668-44058.75))),np.percentile(df["Income"],50),df["Income"])
```

In [13]:
```
df.describe()
```

Out[13]:

	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	49275.119444	93.094444
std	6.943498	1.617055	1.084797	0.958869	9390.075400	34.228398
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	50312.250000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	53463.250000	106.000000
max	50.000000	21.000000	7.000000	5.000000	77191.000000	180.000000

Observations:

- More outliers are found in MILES and INCOME
- As the Analysis is based on the product and the data is small for outlier treatment we just replaced the outliers with median value to make the mean and median oproximately equal

Univariate Analysis

In [14]:
```
df["Product"].value_counts()
```

Out[14]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [15]:
```
df["Gender"].value_counts()
```

Out[15]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [16]:
```
df["MaritalStatus"].value_counts()
```

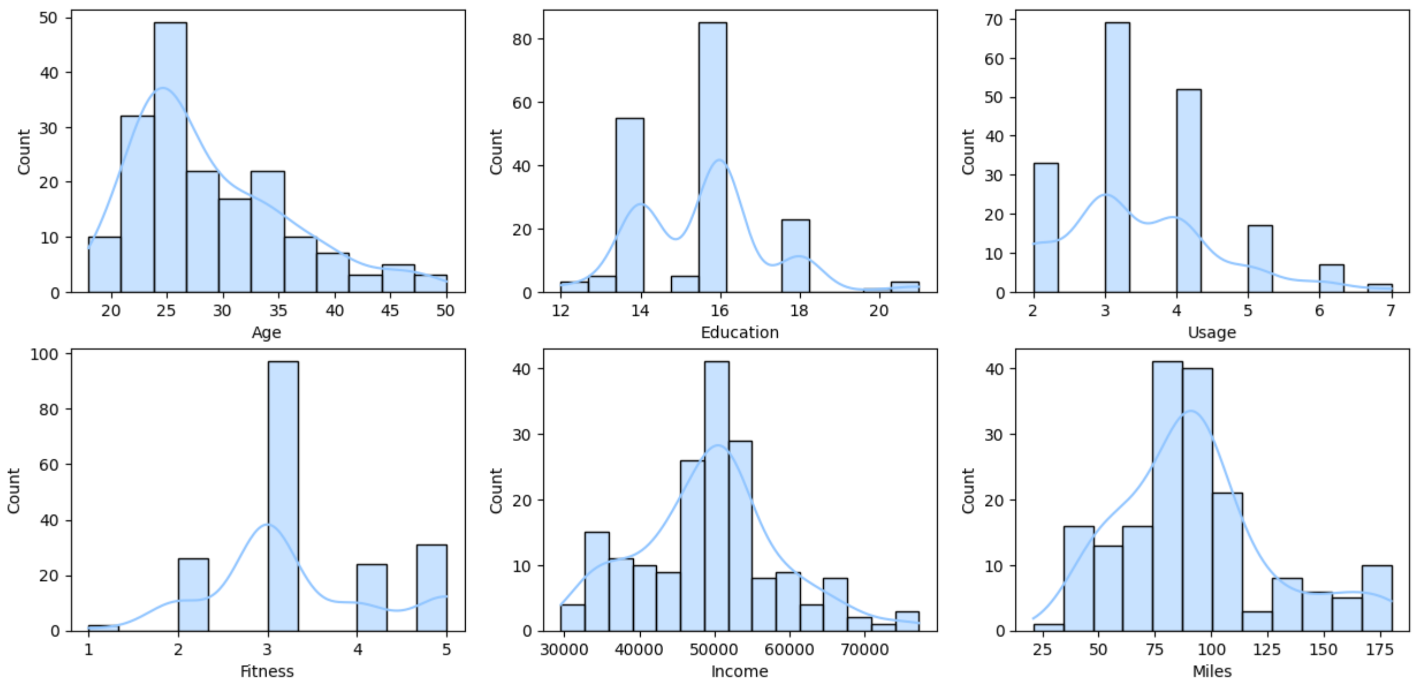
Out[16]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

Observations:

- KP281 product is the most frequent product
- The dataset contains most no of males than females

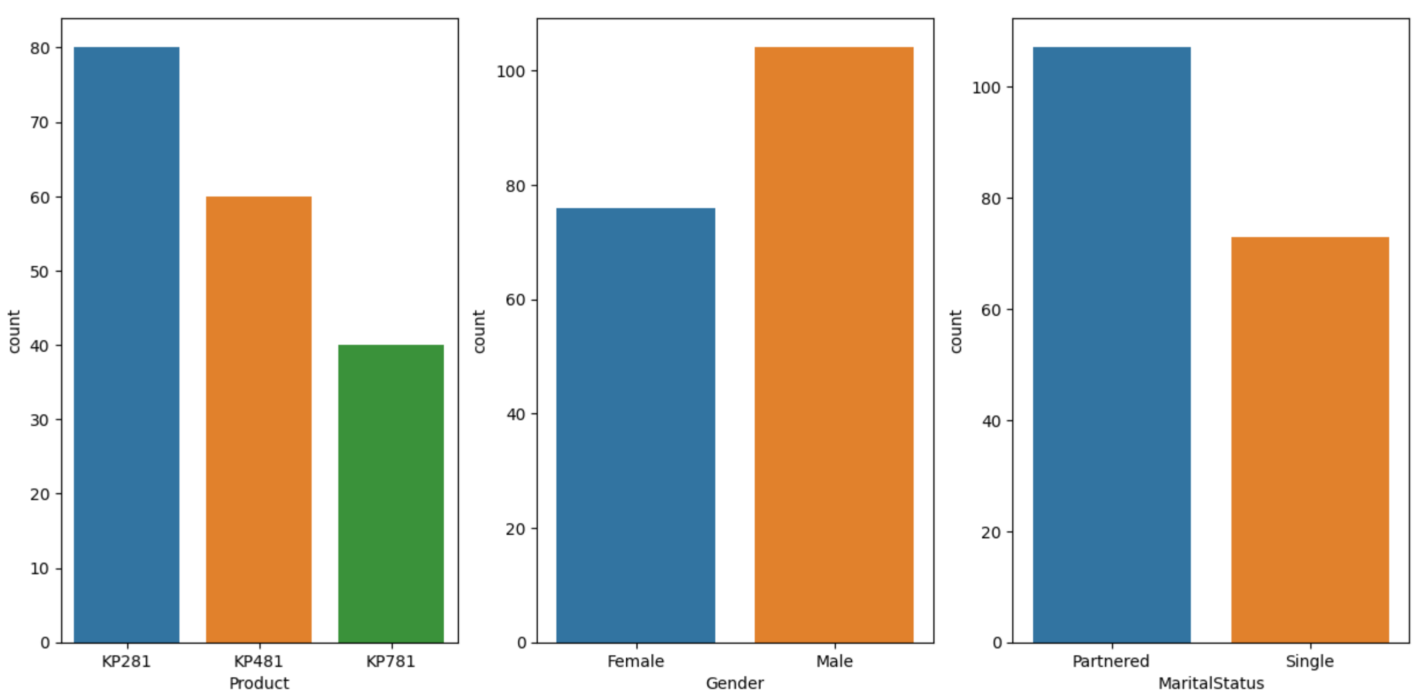
- mostly partnered people are ordering the products

In [17]:
```python
with plt.style.context("seaborn-v0_8-pastel"):
  plt.figure(figsize=(15,7))
  plt.subplot(2,3,1)
  sns.histplot(data=df, x="Age", kde=True)
  plt.subplot(2,3,2)
  sns.histplot(data=df, x="Education", kde=True)
  plt.subplot(2,3,3)
  sns.histplot(data=df, x="Usage", kde=True)
  plt.subplot(2,3,4)
  sns.histplot(data=df, x="Fitness", kde=True)
  plt.subplot(2,3,5)
  sns.histplot(data=df, x="Income", kde=True)
  plt.subplot(2,3,6)
  sns.histplot(data=df, x="Miles", kde=True)
  plt.show()
```



In [32]:
```python
#Understanding the distribution of the data for the qualitative attributes:
plt.figure(figsize=(15,7))
plt.subplot(1,3,1)
sns.countplot(data=df, x='Product')
plt.subplot(1,3,2)
sns.countplot(data=df, x='Gender')
plt.subplot(1,3,3)
sns.countplot(data=df, x='MaritalStatus')
```

Out[32]: <Axes: xlabel='MaritalStatus', ylabel='count'>



In [31]:
```python
#analysis on product based on gender
pd.crosstab(index=df['Gender'], columns=df['Product'], margins=True,normalize=True)
```

Out[31]:

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

In [19]:
```python
#Analysis of product based on Age
pd.crosstab(index=df['Age'], columns=df['Product'], margins=True)
```

Out[19]:

Product	KP281	KP481	KP781	All
Age				
18	1	0	0	1
19	3	1	0	4
20	2	3	0	5
21	4	3	0	7
22	4	0	3	7
23	8	7	3	18
24	5	3	4	12
25	7	11	7	25
26	7	3	2	12
27	3	1	3	7
28	6	0	3	9
29	3	1	2	6
30	2	2	3	7
31	2	3	1	6
32	2	2	0	4
33	2	5	1	8
34	2	3	1	6
35	3	4	1	8
36	1	0	0	1
37	1	1	0	2
38	4	2	1	7
39	1	0	0	1
40	1	3	1	5
41	1	0	0	1
42	0	0	1	1
43	1	0	0	1
44	1	0	0	1
45	0	1	1	2
46	1	0	0	1
47	1	0	1	2
48	0	1	1	2
50	1	0	0	1
All	80	60	40	180

In [30]:

```
#Analysis of product based on MaritalStatus
pd.crosstab(index=df['MaritalStatus'], columns=df['Product'], margins=True,normalize=True)
```

Out[30]:

Product	KP281	KP481	KP781	All
MaritalStatus				
Partnered	0.266667	0.200000	0.127778	0.594444
Single	0.177778	0.133333	0.094444	0.405556
All	0.444444	0.333333	0.222222	1.000000

Observations:

1. Product

- 44.44% of the customers have purchased KP2821 product.
- 33.33% of the customers have purchased KP481 product.
- 22.22% of the customers have purchased KP781 product.

2. Gender

- 57.78% of customers are male
- 42.22% of customers are female

3. MaritalStatus

- 59.44% of the customers are Partnered.
- 40.55% of customers are single

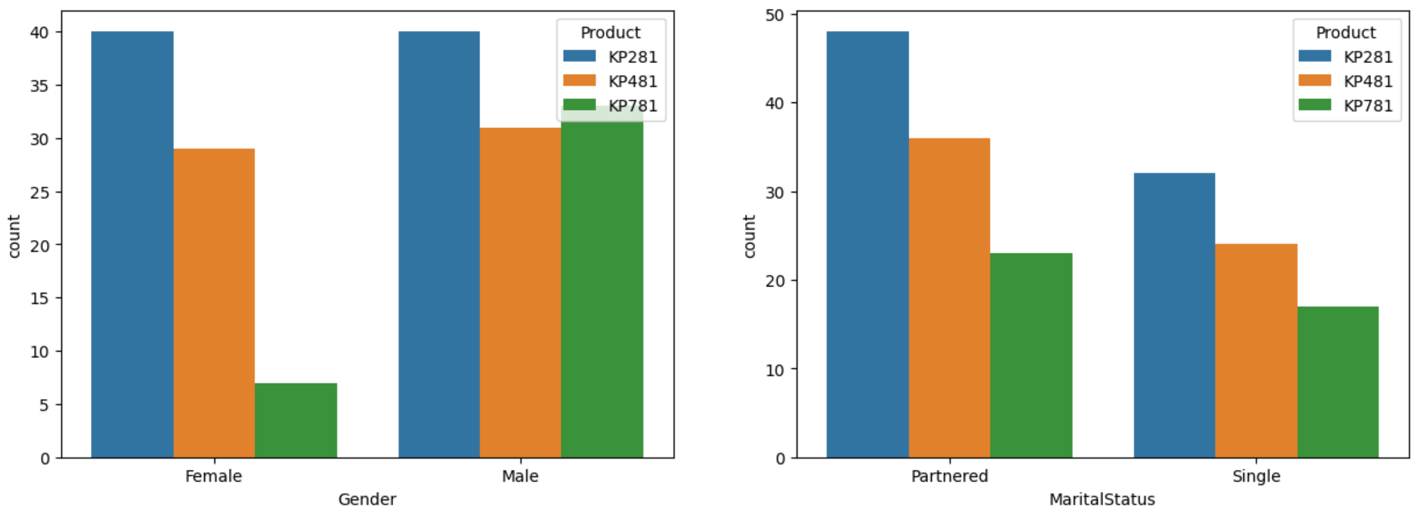
4. Age

- Mostly the products are ordered from the age range of 23-26 years old

Bivariate Analysis

In [34]:
```
plt.figure(figsize=(15,5))
plt.subplot(1,2,1)
sns.countplot(data=df, x='Gender',hue="Product")
plt.subplot(1,2,2)
sns.countplot(data=df, x='MaritalStatus',hue="Product")
```

Out[34]:
```
<Axes: xlabel='MaritalStatus', ylabel='count'>
```

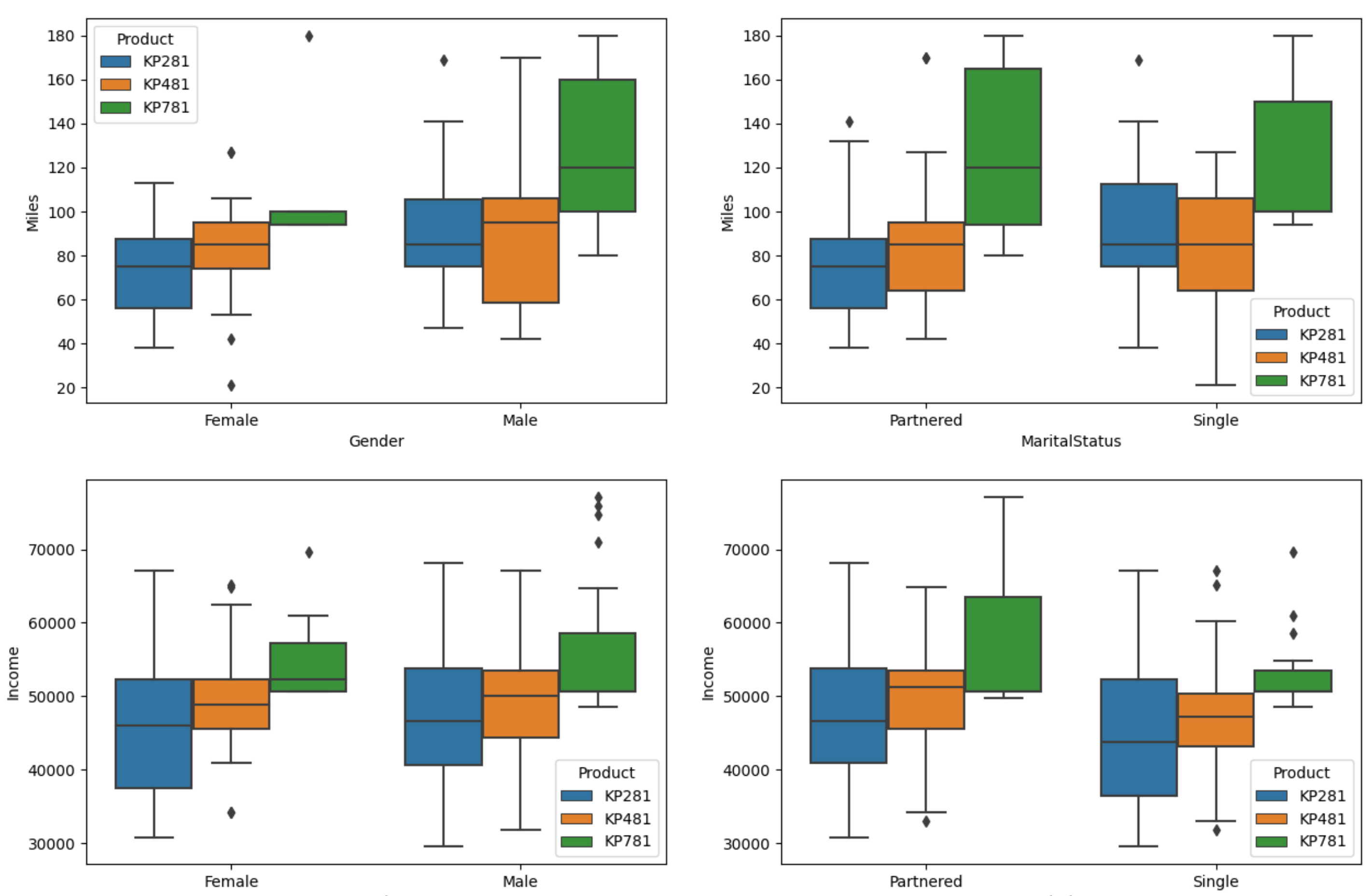


Observations:

- Males are purchasing all the products similarly but in female most of the products are KP281
- Most no of purchases are from partnered

In [40]:
```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.boxplot(x="Gender", y="Miles", hue="Product", data=df)
plt.subplot(2,2,2)
sns.boxplot(x="MaritalStatus", y="Miles", hue="Product", data=df)
plt.subplot(2,2,3)
sns.boxplot(x="Gender", y="Income", hue="Product", data=df)
plt.subplot(2,2,4)
sns.boxplot(x="MaritalStatus", y="Income", hue="Product", data=df)
```

Out[40]:
```
<Axes: xlabel='MaritalStatus', ylabel='Income'>
```



In [ ]:

Observations:

1. Gender vs Miles

- Males are mostly making use of KP781 the median of KP781 is around 120 miles which is higher.

- KP781 is less used in females but most of the miles are made in KP781 in females

1. MaritalStatus vs Miles

- Partnered people are mostly using KP781 and they are making more miles than others

1. Gender vs Income

- Mostly Males and females have approximately same Income and they are using the products in the same manner and the people having more income are mostly prefering KP781

1. MaritalStatus vs Income

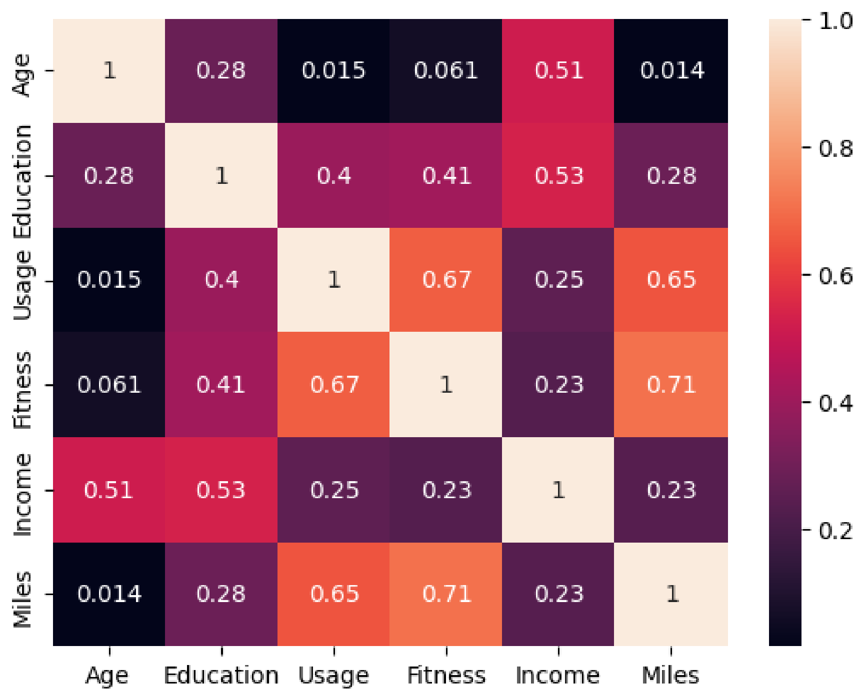
- Mostly partnered people have more income than single and they are mostly prefering KP781
- Singles are prefering all products approximately same

In [23]:

```
#Correlation based on heatmap for all integer columns
sns.heatmap(df.corr(), annot=True)
```

```
<ipython-input-23-6dc1c4c1753e>:1: 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 sil
ence this warning.
  sns.heatmap(df.corr(), annot=True)
```

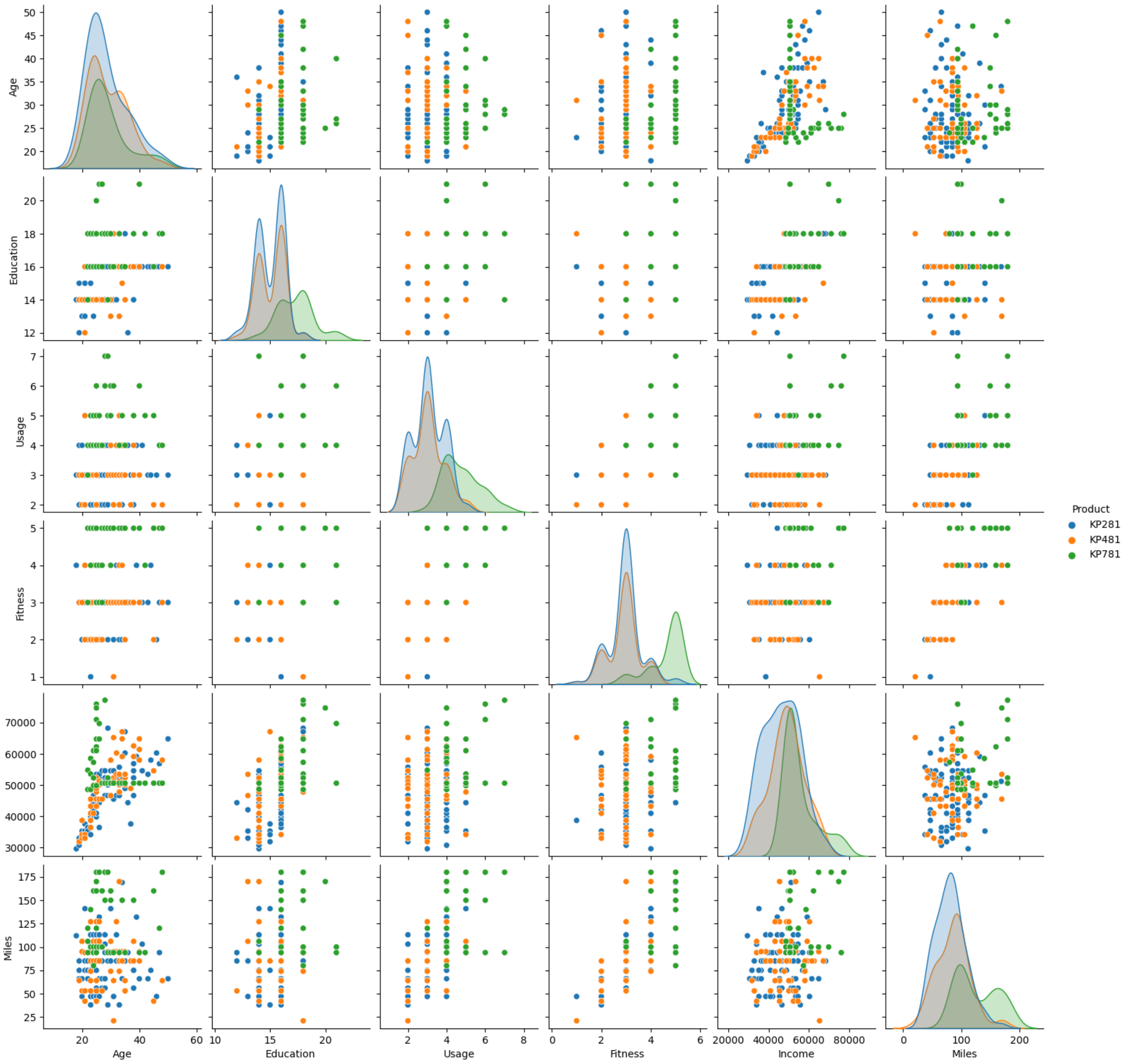
Out[23]: <Axes: >



In [44]:

```
#Correlation based on Pairplot for all integer columns
sns.pairplot(data=df,hue="Product")
```

Out[44]: <seaborn.axisgrid.PairGrid at 0x7cd1b299fd30>



Marginal Probability

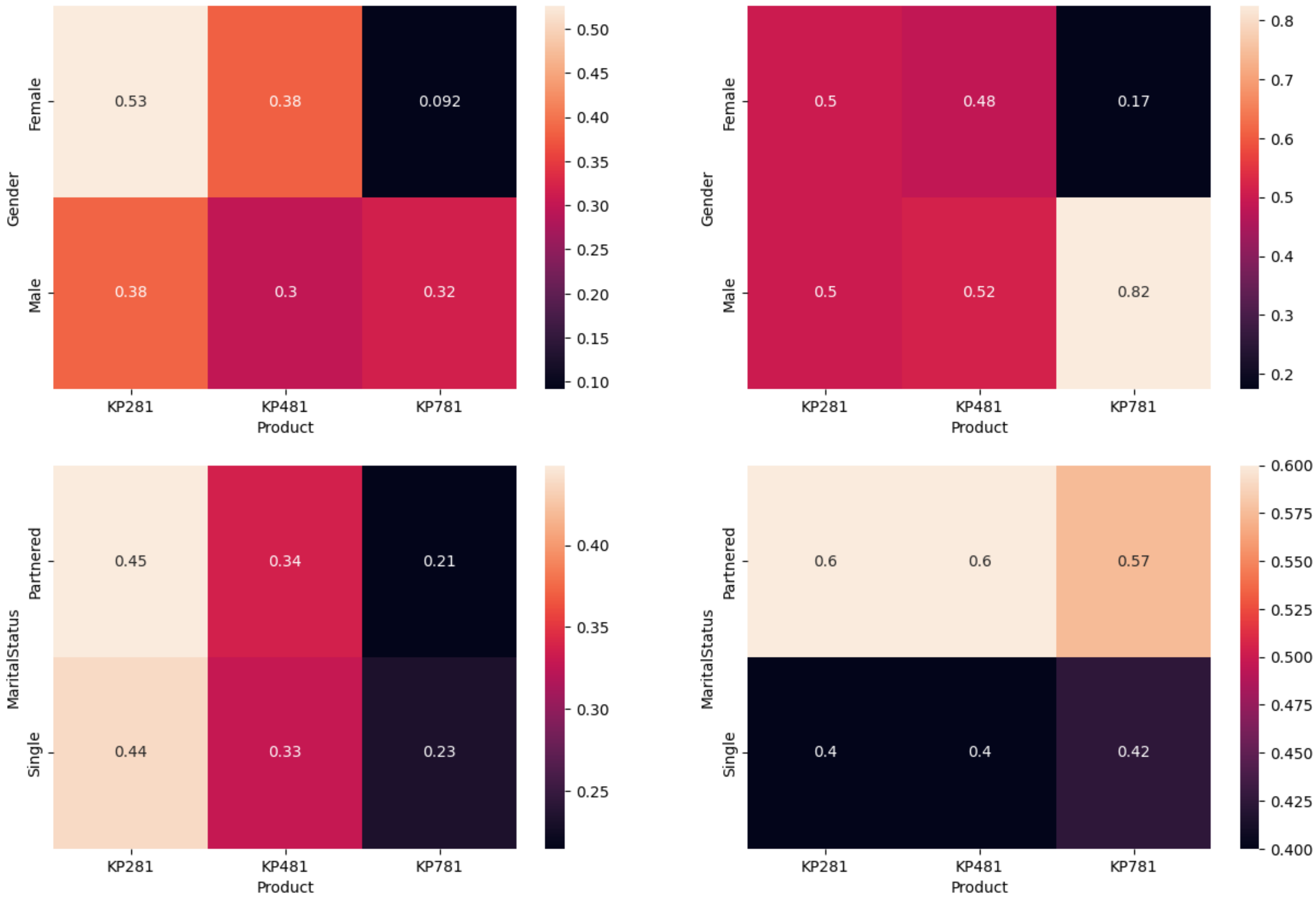
In [47]:
```python
df1=df[["Product","Gender","MaritalStatus"]].melt()
df1.groupby(["variable","value"])["value"].count()/len(df)
```

Out[47]:
```
variable       value    
Gender         Female       0.422222
               Male         0.577778
MaritalStatus  Partnered    0.594444
               Single       0.405556
Product        KP281        0.444444
               KP481        0.333333
               KP781        0.222222
Name: value, dtype: float64
```

Conditional Probability

In [57]:
```python
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.heatmap(pd.crosstab(df["Gender"],df["Product"],normalize="index"),annot=True)
plt.subplot(2,2,2)
sns.heatmap(pd.crosstab(df["Gender"],df["Product"],normalize="columns"),annot=True)
plt.subplot(2,2,3)
sns.heatmap(pd.crosstab(df["MaritalStatus"],df["Product"],normalize="index"),annot=True)
plt.subplot(2,2,4)
sns.heatmap(pd.crosstab(df["MaritalStatus"],df["Product"],normalize="columns"),annot=True)
```

Out[57]:
```
<Axes: xlabel='Product', ylabel='MaritalStatus'>
```



Observations:

1. conditional probability for each product given that in Gender
   - p(KP281|Male)=0.38
   - p(KP281|Female)=0.53
   - p(KP481|Male)=0.3
   - p(KP481|Female)=0.38
   - p(KP781|Male)=0.32
   - p(KP781|Female)=0.092
2. conditional probability for Gender given that in Product
   - p(Male|KP281)=0.5
   - p(Male|KP481)=0.52
   - p(Male|KP781)=0.82
   - P(Female|KP281)=0.5
   - p(Female|KP481)=0.48
   - p(Female|KP781)=0.17
3. conditional probability for MaritalStatus given that in Product
   - p(Single|KP281)=0.4
   - p(Single|KP481)=0.4
   - p(Single|KP781)=0.42
   - P(Partnered|KP281)=0.6
   - p(Partnered|KP481)=0.6
   - p(Partnered|KP781)=0.57
4. conditional probability for Product given that in MaritalStatus
   - p(KP281|Single)=0.44
   - p(KP281|Partnered)=0.45
   - p(KP481|Single)=0.33
   - p(KP481|Partnered)=0.34
   - p(KP781|Single)=0.23
   - p(KP781|Partnered)=0.21

Insights

1. 57.78% Customers are Male.
2. 59.44% Customers are Partnered.
3. Most sold product KP281, its 44.44% of sales out of overall Aerofit Treadmill sale.
4. KP281, KP481 products have almost similar customer's profile, except Male Partnered prefer KP481 & Female Partnered prefer KP281.
5. KP781 product is most preferred by Males, it's almost 6 times compared to Females.
6. 75% of customers are earning less than 60k, and customers who earning more than 60k prefer KP781.
7. KP781 had unique among other treadmills when it comes more usage or high fitness customer.
8. Probability of Buying KP281 increased from 44.44% to 58.7%, if customer is Female and Partnered.
9. Probability of Buying KP781 increased from 22.22% to 32.56%, if customer is Male and Single.
10. Probability of Buying KP781 decreased from 22.22% to 8.7%, if customer is Female and Partnered.

Recommendations

1. As KP781 premium product preferred by Males, more usage and high salaried people.
2. we can promote this product with similar characteristics and also we can promote upcoming premium products to them.
3. KP281 & KP481 products preferred by almost similar Characteristics and KP281 is most sold product
4. we can promote KP481 products more and can make some no cost EMI support.
5. Provide personalized Ads in E-commerce sites and in Social Media for better reach to similar characteristics of people with respective preferred products.
6. If customer is ready to by KP481 then make sure that you also provide the information of KP781 and its uses with reference to KP781.
7. Try giving discounts to KP781 so that it will be helpfull to attract the people with lower salaries.
8. Mostly partnered people are prfering the product so that try to attract single people by the advertisements.
9. When a customer come to buy a product then just give some experience to the customer by each product and let the customer know the benifits of the product.