In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [4]:
```
df=pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749")
```

In [6]:
```
df.head()
```

Out[6]:

	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 [7]:
```
df.shape
```

Out[7]:
```
(180, 9)
```

In [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
```

In [10]:
```
df.describe()
```

Out[10]:

	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

In [12]:
```
df.isnull().sum()
```

Out[12]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

In [13]:
```
df["Product"].nunique()
```

Out[13]:
```
3
```

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

Out[15]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [96]:
```
df["MaritalStatus"].nunique()
```

Out[96]:
```
2
```

In [94]:
```
df["MaritalStatus"].value_counts()
```

Out[94]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

In [97]:
```
df["Gender"].nunique()
```

Out[97]:
```
2
```

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

Out[16]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [95]:
```
df["Age"].value_counts()
```

Out[95]:
```
25    25
23    18
24    12
26    12
28     9
35     8
33     8
30     7
38     7
21     7
22     7
27     7
31     6
34     6
29     6
20     5
40     5
32     4
19     4
48     2
37     2
45     2
47     2
46     1
50     1
18     1
44     1
43     1
41     1
39     1
36     1
42     1
Name: Age, dtype: int64
```

In [ ]:
```
#Observations:
#There are no missing values in the data.
#There are 3 unique products in the dataset.
#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.
#75% of persons are having education <= 16 years.
#there are 104 Male and 76 female in the given data.
#Standard deviation for Income & Miles is very high. These variables might have the outliers in it.
```

In [ ]:
```
#univariate analysis for qualitative attributes
```

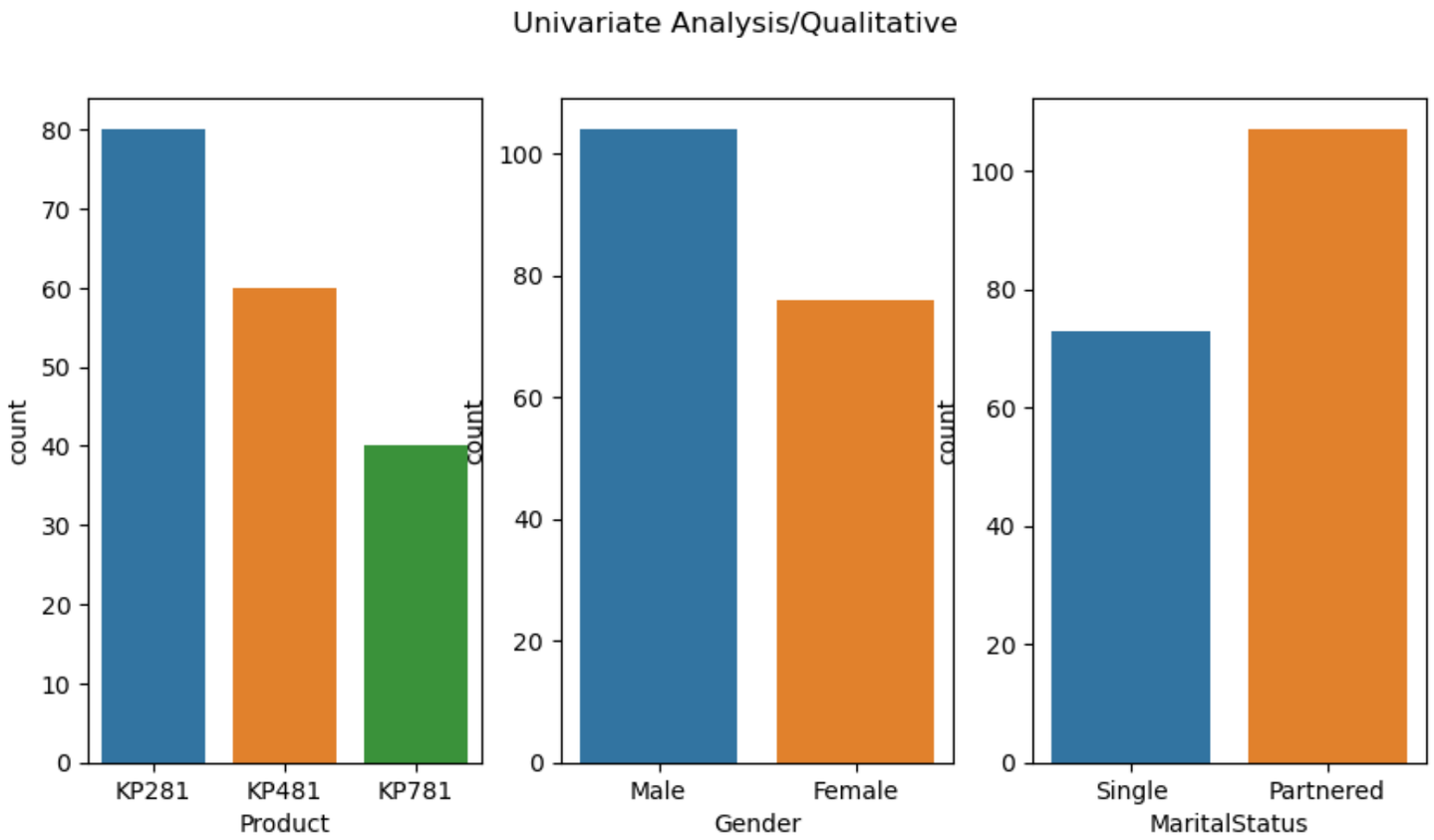
In [38]:
```
df1 = df[['Product', 'Gender', 'MaritalStatus']].melt()
df1.groupby(['variable', 'value'])[['value']].count() / len(df)*100
```

Out[38]:

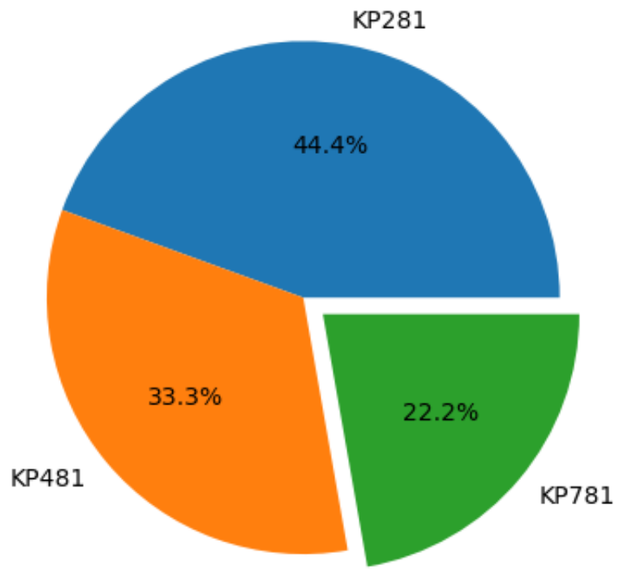
		value
variable	**value**	
Gender	**Female**	42.222222
	Male	57.777778
MaritalStatus	**Partnered**	59.444444
	Single	40.555556
Product	**KP281**	44.444444
	KP481	33.333333
	KP781	22.222222

In [ ]:
```
#gender:42.2% female & 57.77% male
#marital status:59.44% partnered & 40.55 %single
#product:44.44% KP281,33.33% KP481,22.22% KP781
```

In [84]:
```
fig = plt.figure(figsize=(10,5))

plt.subplot(1,3,1)
sns.countplot(x="Product", data=df)

plt.subplot(1,3,2)
sns.countplot(x="Gender", data=df)

plt.subplot(1,3,3)
sns.countplot(x="MaritalStatus", data=df)


fig.suptitle('Univariate Analysis/Qualitative')
plt.show()
```



In [78]:
```
plt.pie(df["Product"].value_counts(),labels=df["Product"].unique(),explode=(0,0,0.1),autopct='%1.1f%%')
plt.show()
```

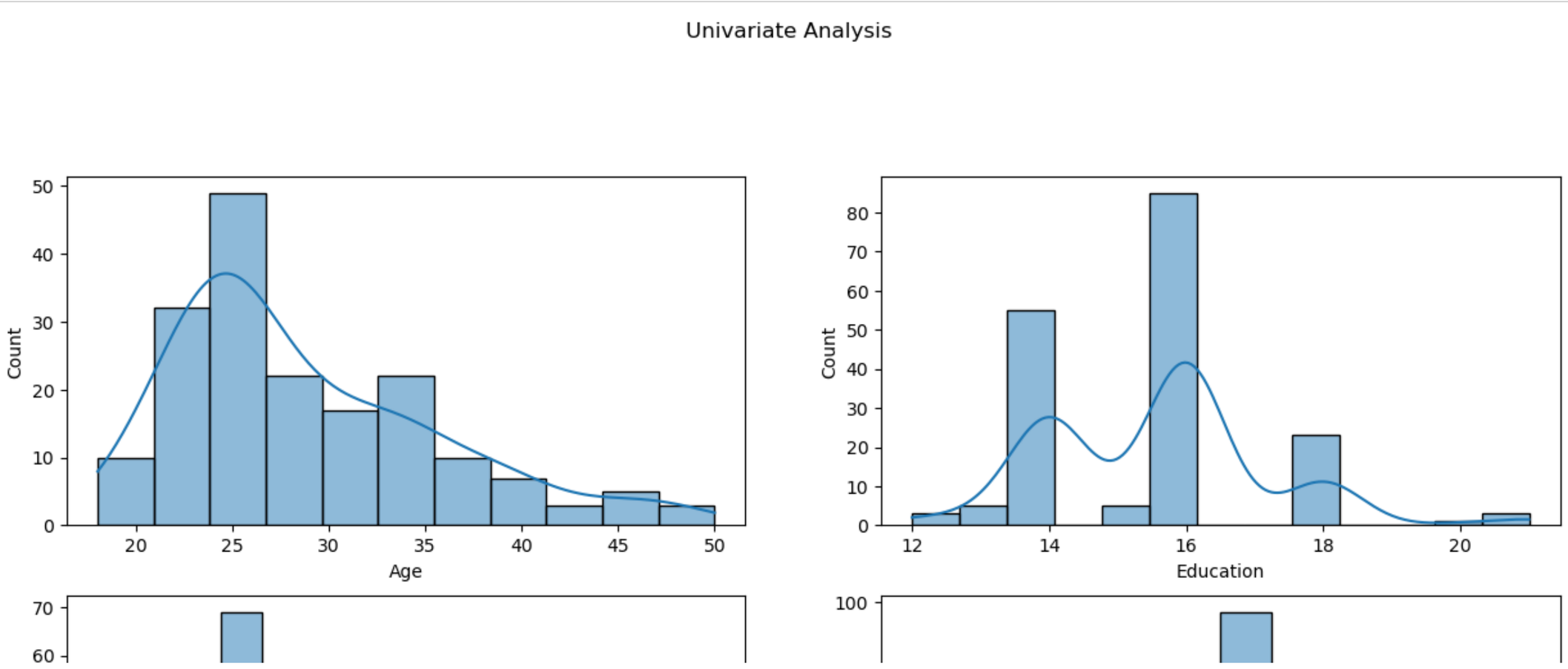


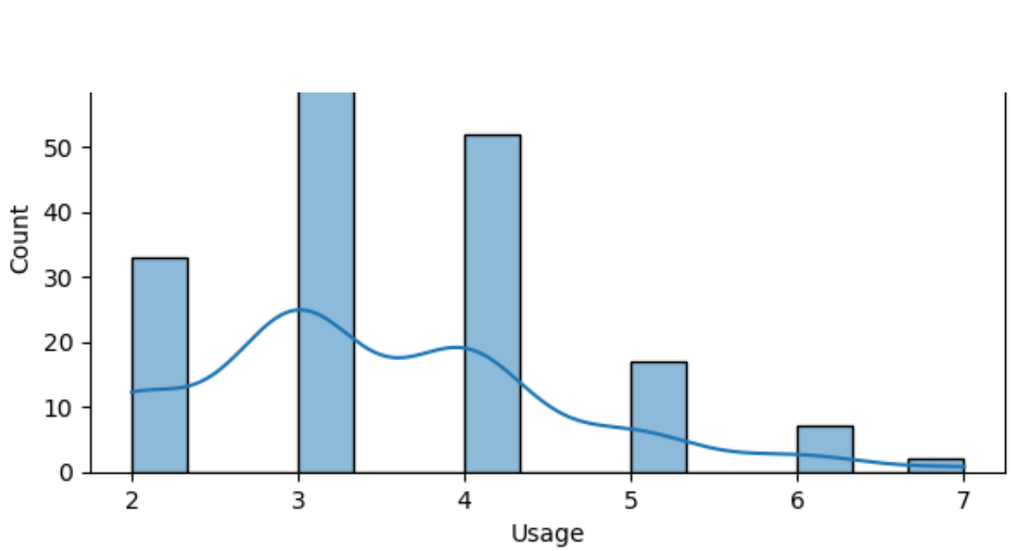
In [ ]:
```
#so from above plot we can infer that in product categeory KP281 has highest popularity among customers.
#in gender categeory male has higher count.
#in marital status categeory partnered has higher count.
```

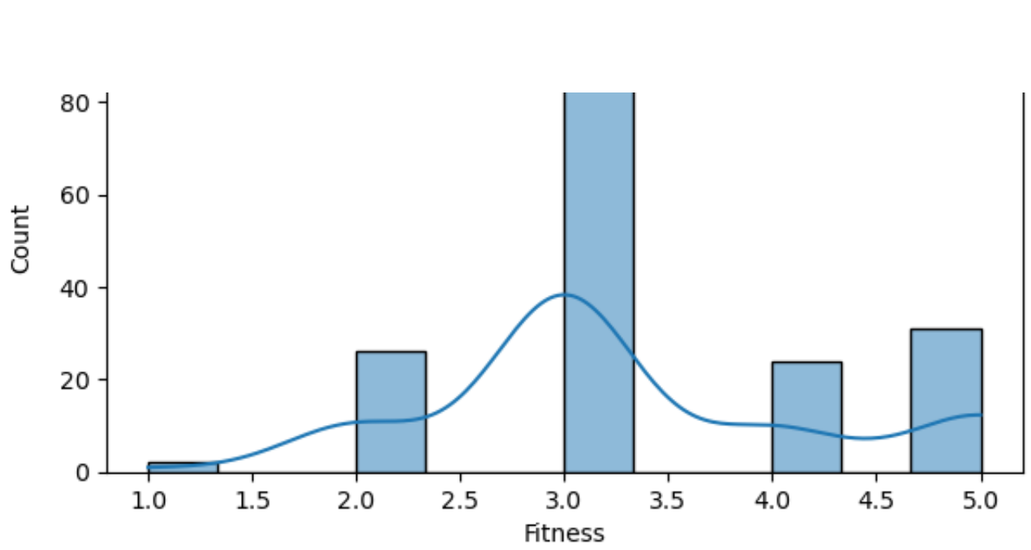
In [ ]:
```
#univariate analysis for quantitative attributes
```

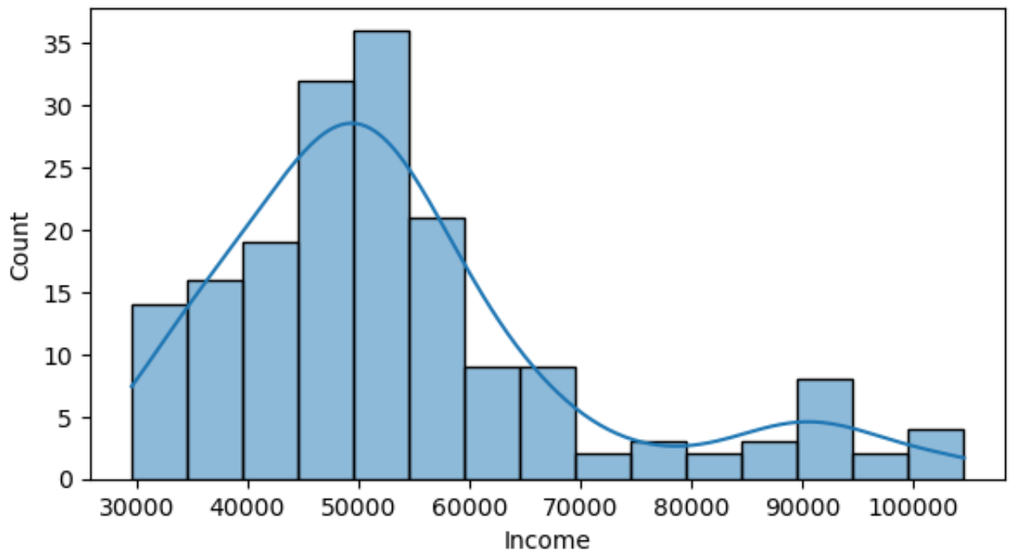
In [26]:
```

```

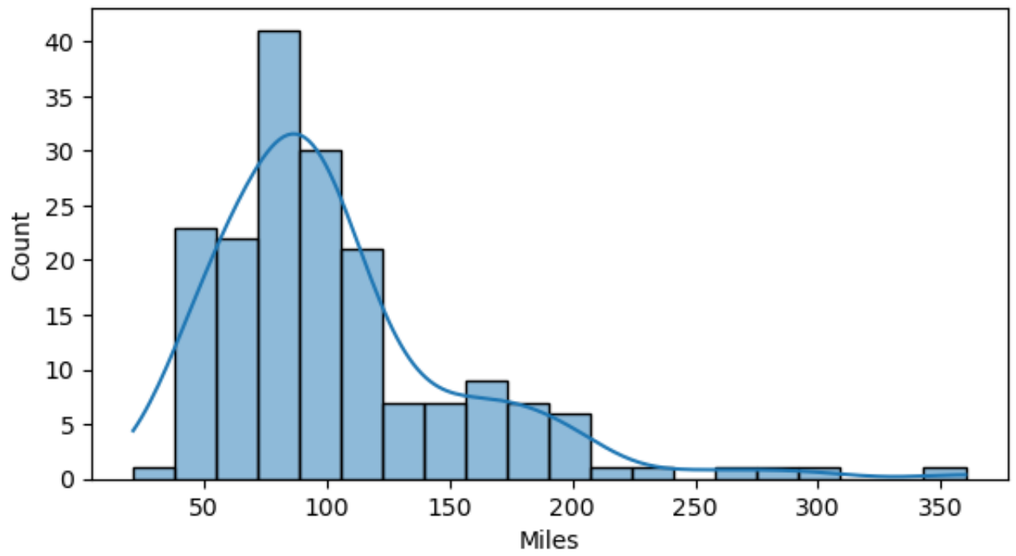
```
fig = plt.figure(figsize=(15,12))

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

plt.subplot(3,2,2)
sns.histplot(x="Education", data=df,kde=True)

plt.subplot(3,2,3)
sns.histplot(x="Usage", data=df,kde=True)

plt.subplot(3,2,4)
sns.histplot(x="Fitness", data=df,kde=True)

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

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


fig.suptitle('Univariate Analysis')
plt.show()
```







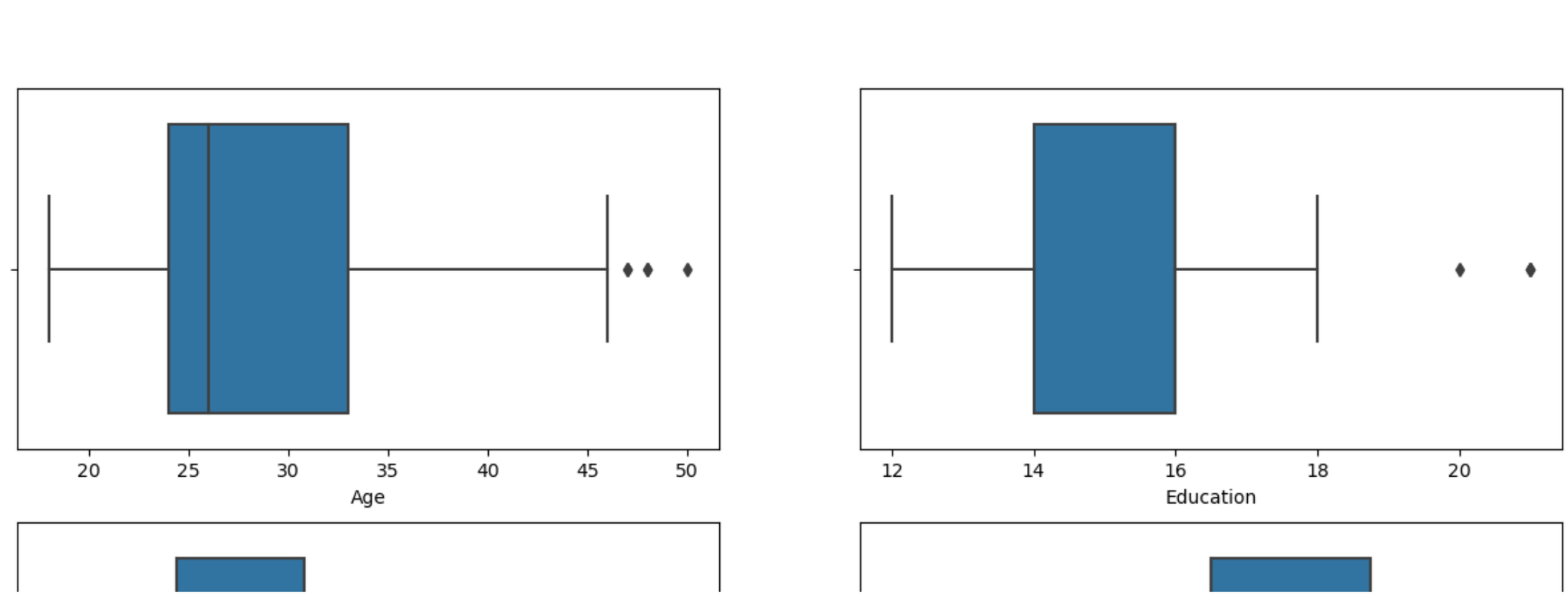


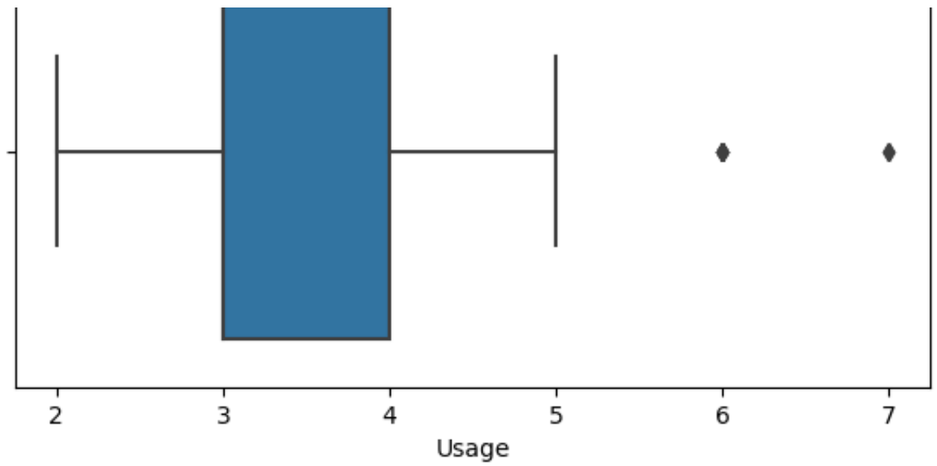


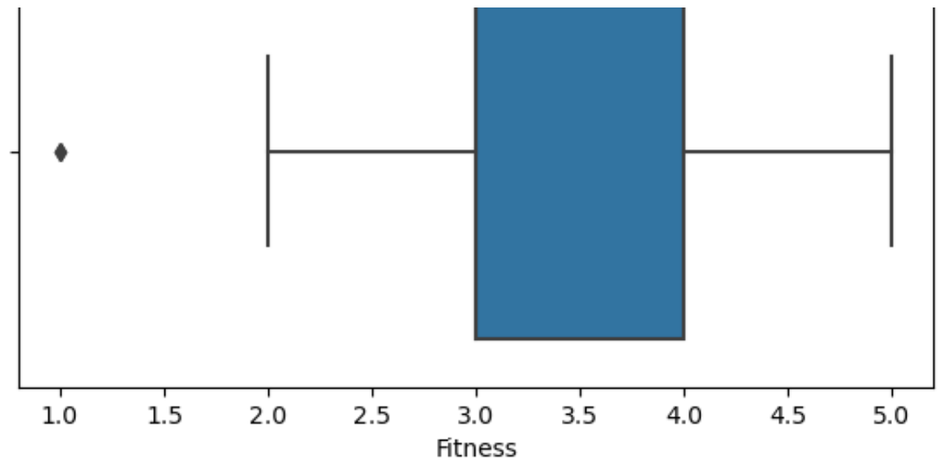
In [ ]:
```
#box plot for outlier dtetection
```

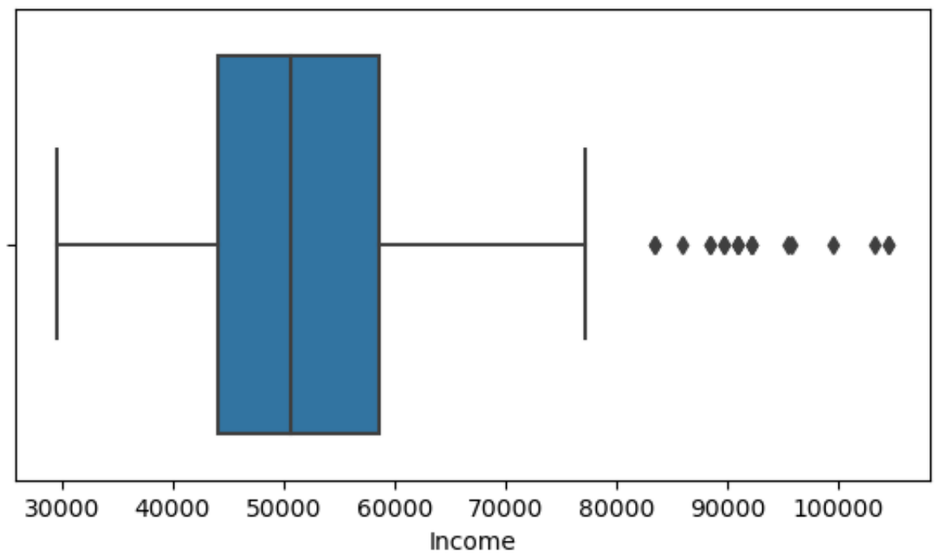
In [28]:
```

```

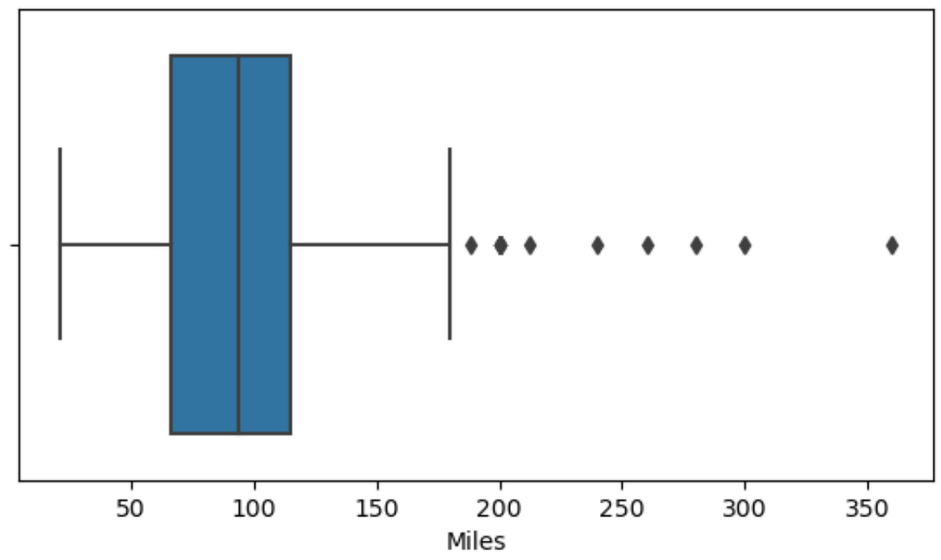
```
fig = plt.figure(figsize=(15,12))

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

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

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

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

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

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


fig.suptitle('Univariate Analysis/Box Plot')
plt.show()
```







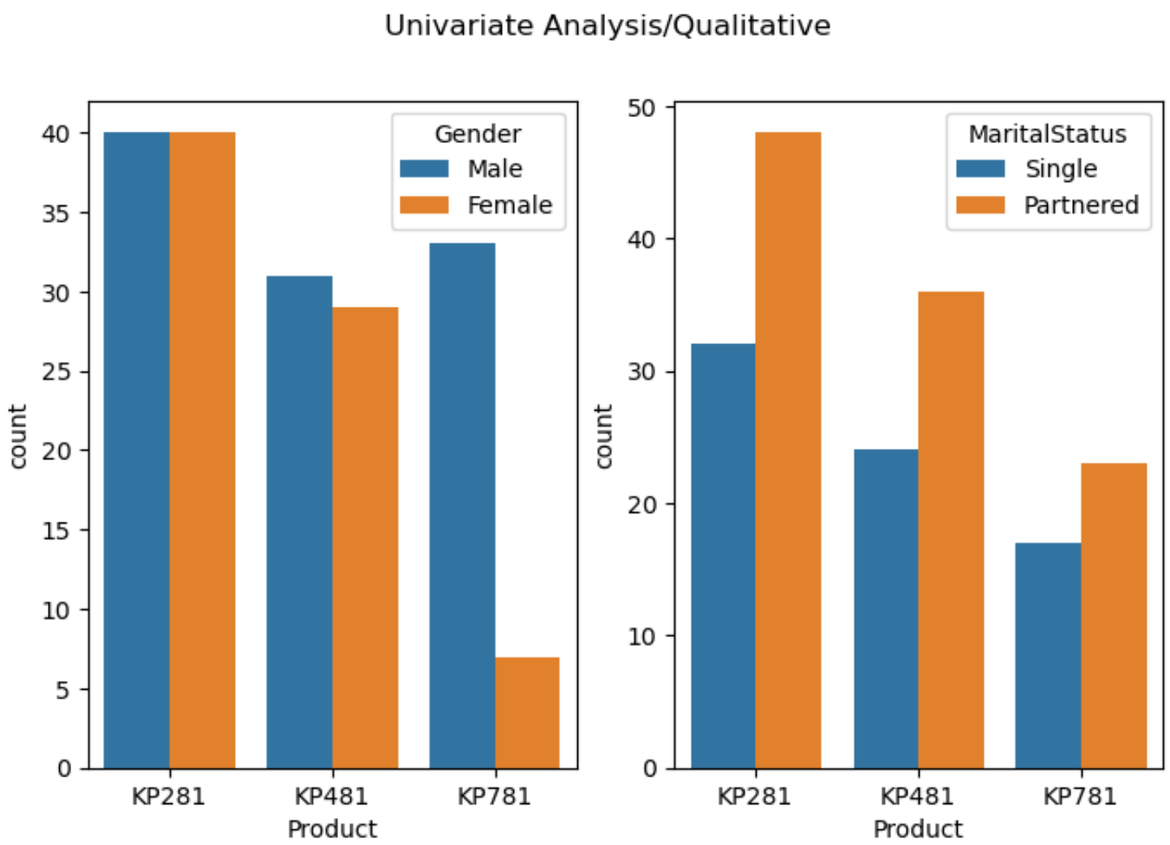




In [ ]:
```
#from the boxplot we can infer that income and miles category has very high no of outliers count.
```

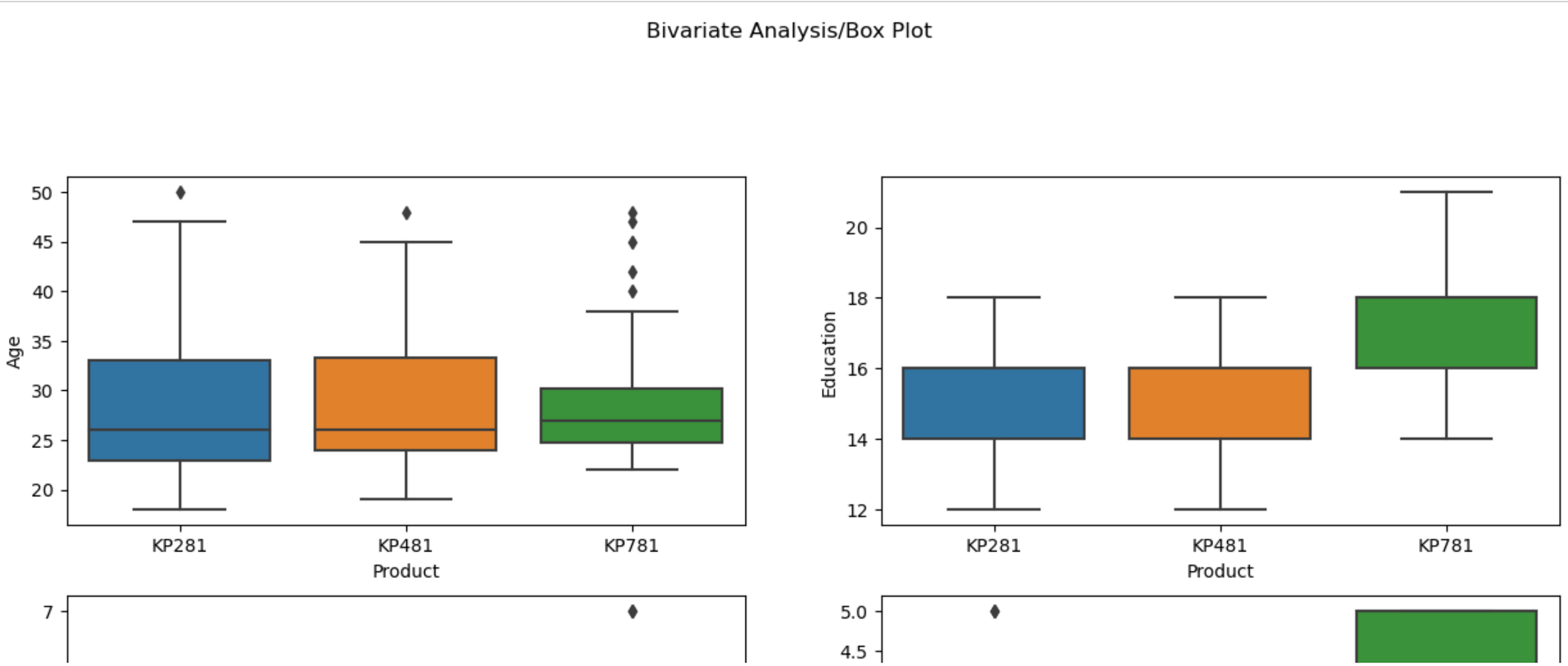
In [ ]:
```
#bivariate analysis
```

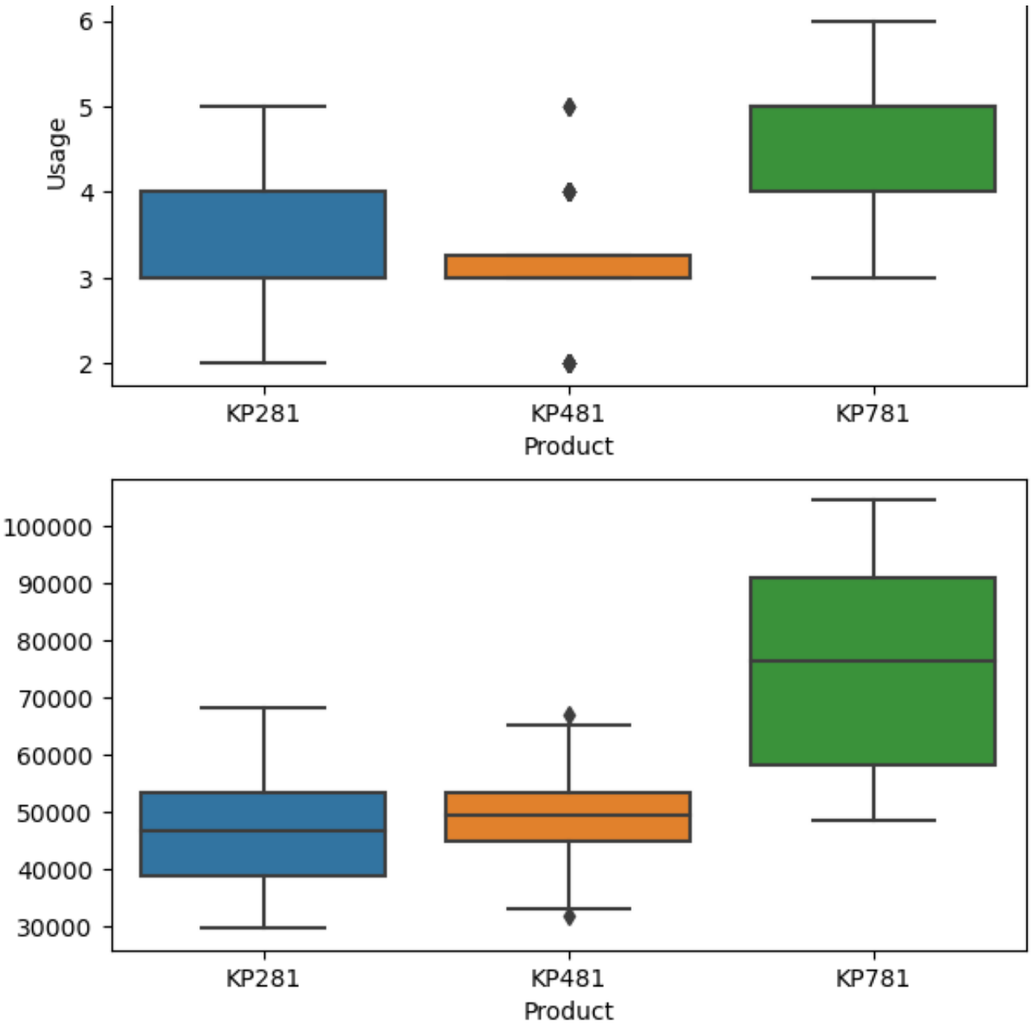
In [42]:
```
fig = plt.figure(figsize=(8,5))

plt.subplot(1,2,1)
sns.countplot(x="Product",hue="Gender", data=df)

plt.subplot(1,2,2)
sns.countplot(x="Product",hue="MaritalStatus", data=df)


fig.suptitle('Univariate Analysis/Qualitative')
plt.show()
```

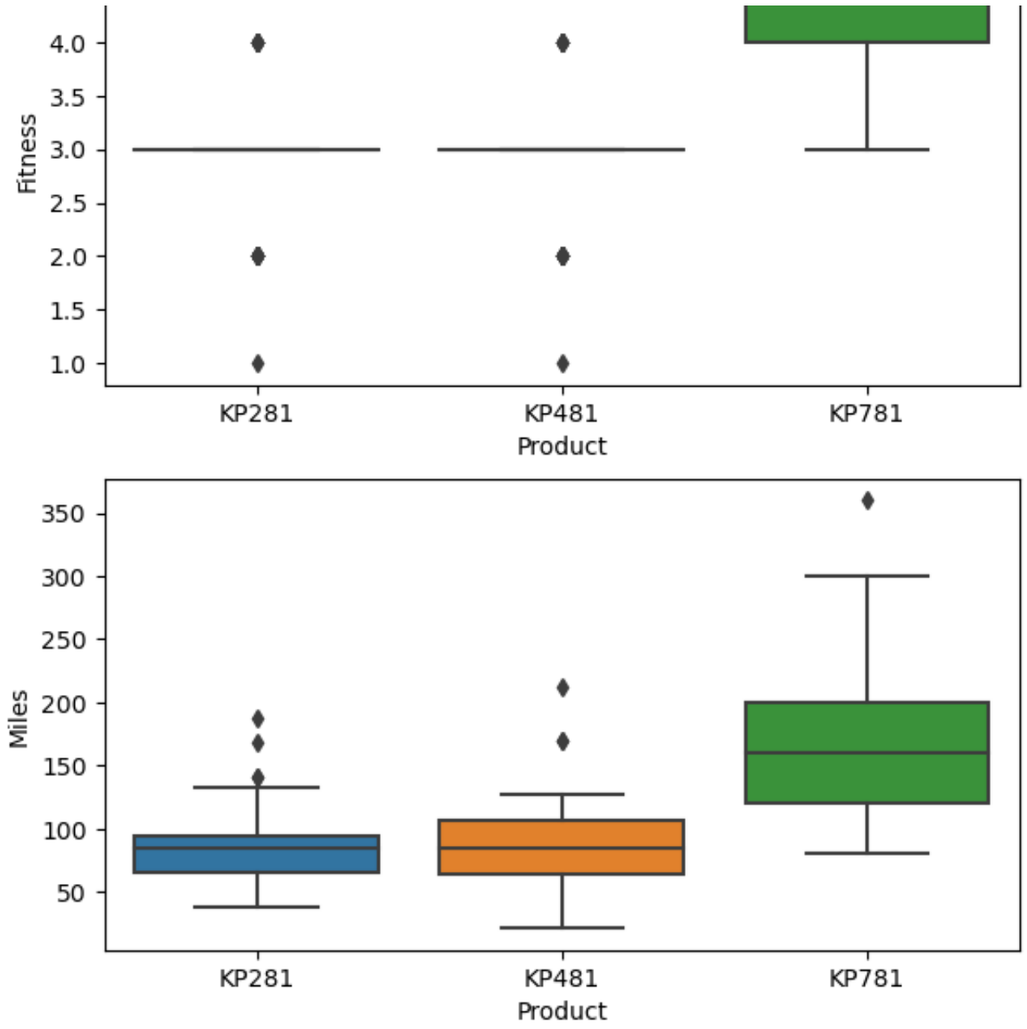


In [43]:

```
fig = plt.figure(figsize=(15,12))

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

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

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

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

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

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


fig.suptitle('Bivariate Analysis/Box Plot')
plt.show()
```




Usage
6
5
4
3
2
KP281
KP481
KP781
Product


Fitness
4.0
3.5
3.0
2.5
2.0
1.5
1.0
KP281
KP481
KP781
Product

Income
100000
90000
80000
70000
60000
50000
40000
30000
KP281
KP481
KP781
Product

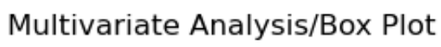
Miles
350
300
250
200
150
100
50
KP281
KP481
KP781
Product

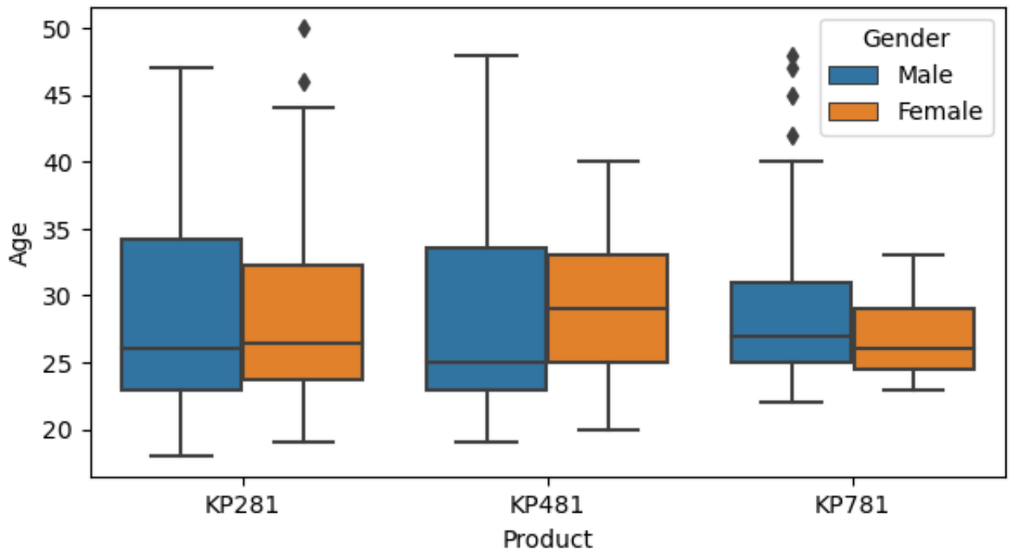
In [48]:
```
"""
observation
product vs age:
customers buying KP281 and KP481 are almost of same age group i.e.23-33 and mean/meadian around 26
while KP781 have mean around 27 but there age segment is around 25-30.
product vs education:
customers having education more than 16 are more likely to buy KP781 whereas for <16 KP281 and KP481 have almost equal probabilities.
product vs usage:
customers having more usage gonna buy KP781 more where as for less usage they will prefer KP281 and KP481.
product vs fitness:
customers having high fitness will prefer KP781.
product vs income:
customers having high income will prefer KP781 whereas for low income KP281.
product vs miles:
customer who have more running goal will prefer KP781.
"""
```

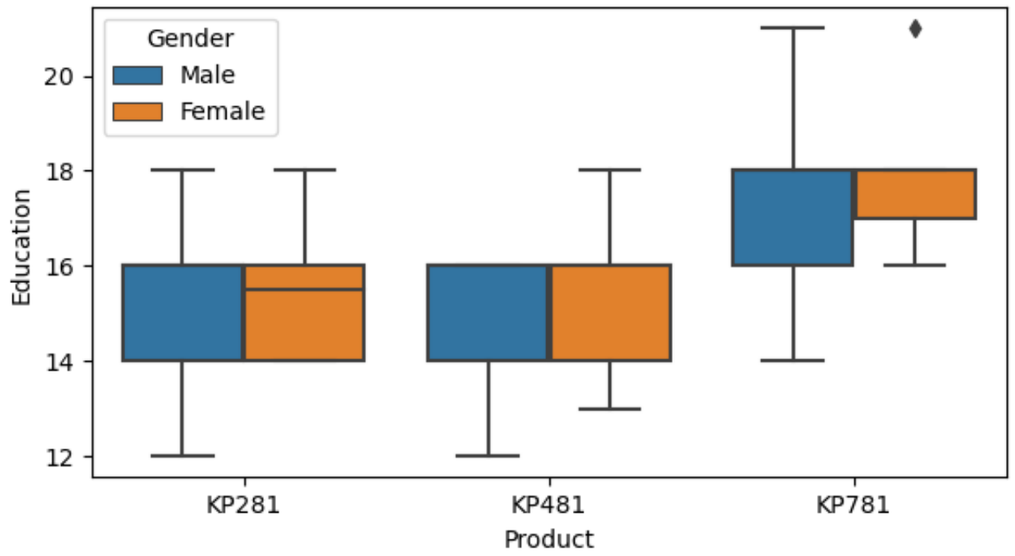
Out[48]:
```
'\nobservation\nproduct vs age:\ncustomers buying KP281 and KP481 are almost of same age group i.e.23-33 and mean/meadian around 26\nwhile KP781 have mean around 27 but there age segment is around 25-30.\nproduct vs education:\ncustomers having education more than 16 are more likely to buy KP781 whereas for <16 KP281 and KP481 have almost equal probabilities.\nproduct vs usage:\ncustomers having more usage gonna buy KP781 more where as for less usage they will prefer KP281 and KP481.\nproduct vs fitness:\ncustomers having high fitness will prefer KP781.\nproduct vs income:\ncustomers having high income will prefer KP781 whereas for low income KP281.\nproduct vs miles:\ncustomer who have more running goal will prefer KP781.\n'
```

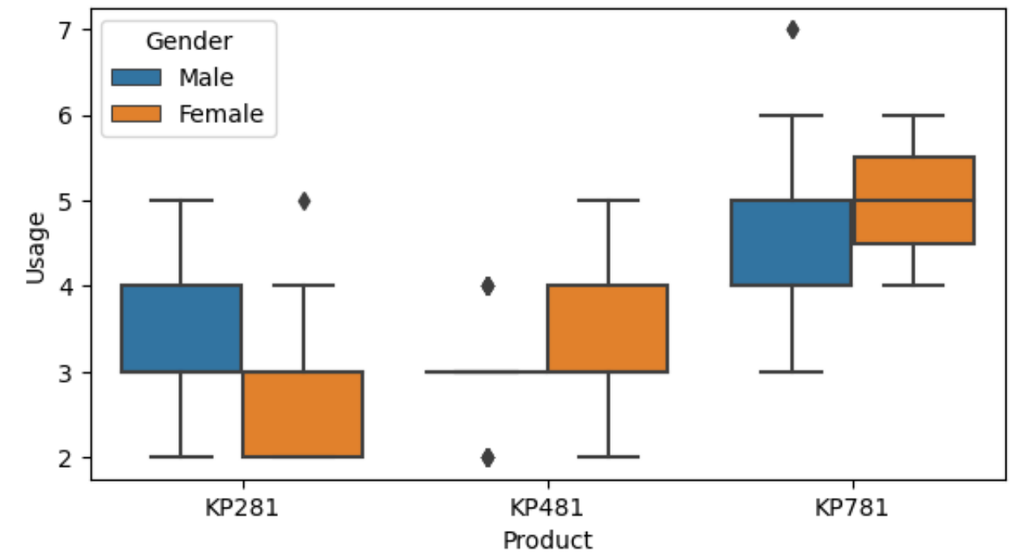
In [ ]:
```
#multi variate analysis
```

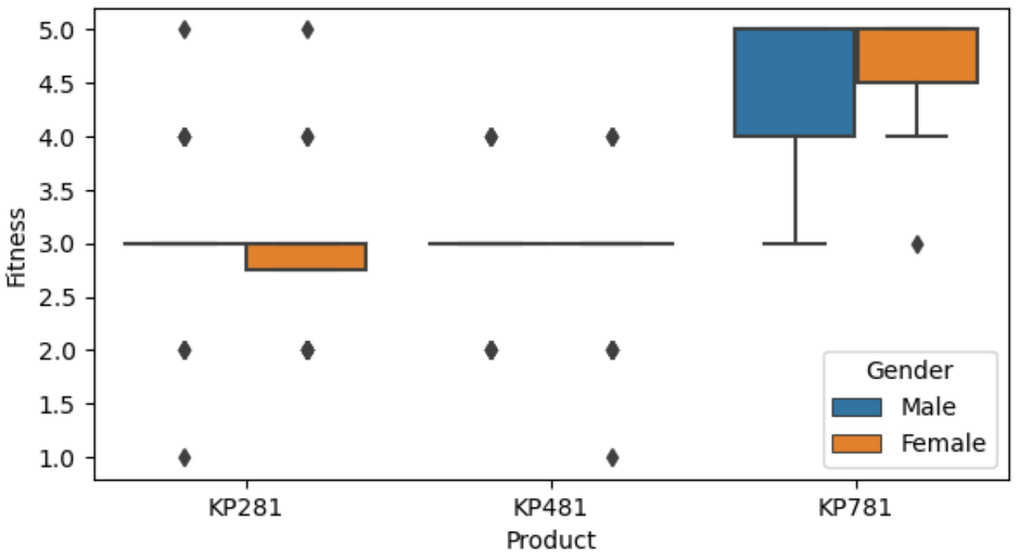
In [54]:
```
fig = plt.figure(figsize=(15,12))

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

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

plt.subplot(3,2,3)
sns.boxplot(x="Product",y="Usage",hue="Gender", data=df)

plt.subplot(3,2,4)
sns.boxplot(x="Product",y="Fitness",hue="Gender", data=df)

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

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


fig.suptitle('Multivariate Analysis/Box Plot')
plt.show()
```

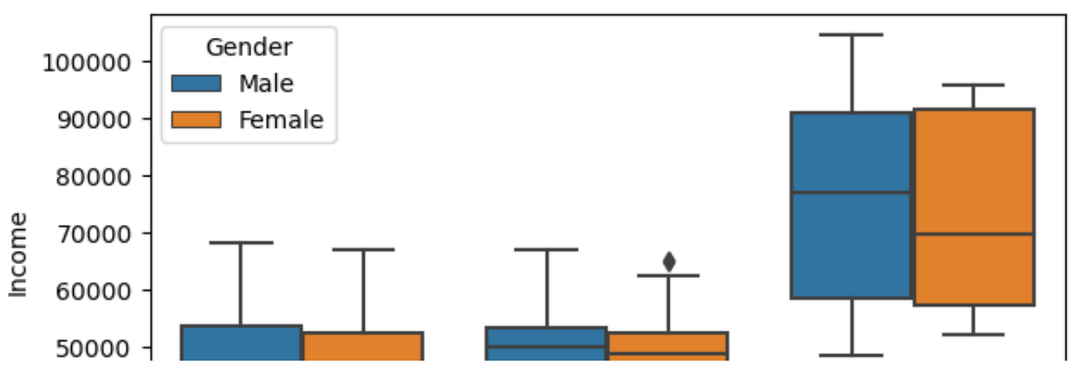


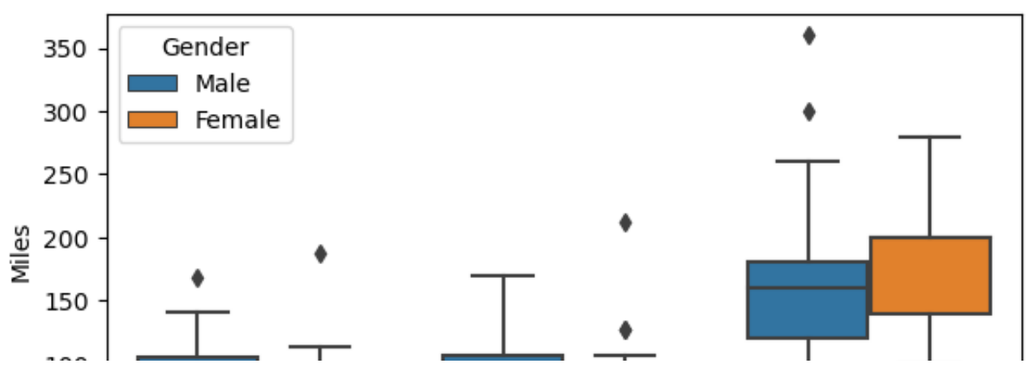

Gender
Male
Female
Age
50
45
40
35
30
25
20
KP281
KP481
KP781
Product

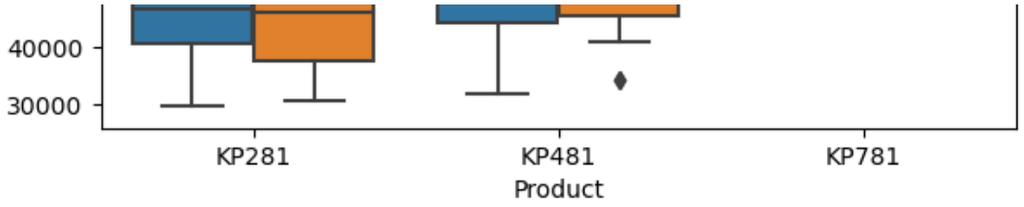

Gender
Male
Female
Education
20
18
16
14
12
KP281
KP481
KP781
Product

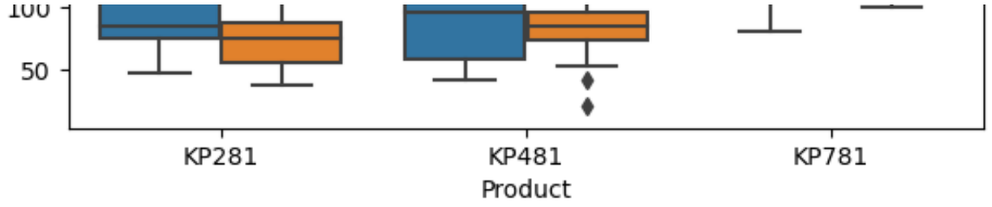

Gender
Male
Female
Usage
7
6
5
4
3
2
KP281
KP481
KP781
Product


Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Gender
Male
Female
KP281
KP481
KP781
Product


Gender
Male
Female
Income
100000
90000
80000
70000
60000
50000

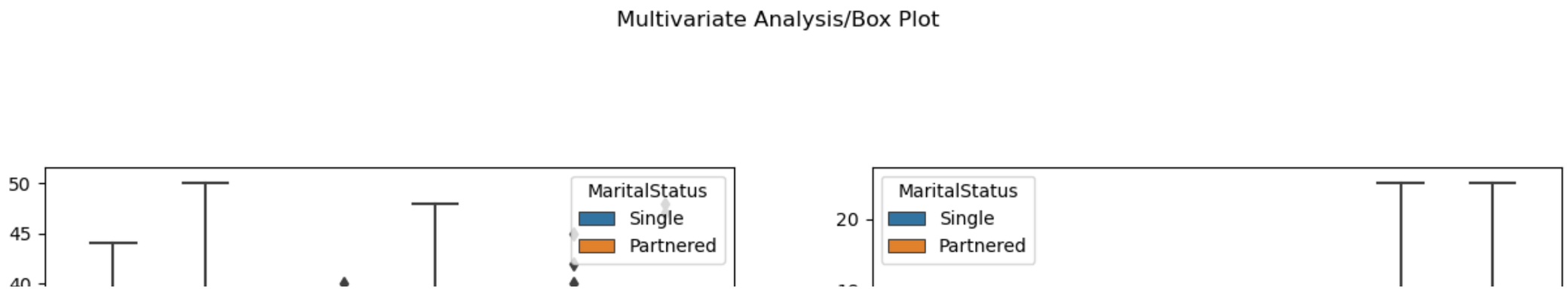

Gender
Male
Female
Miles
350
300
250
200
150

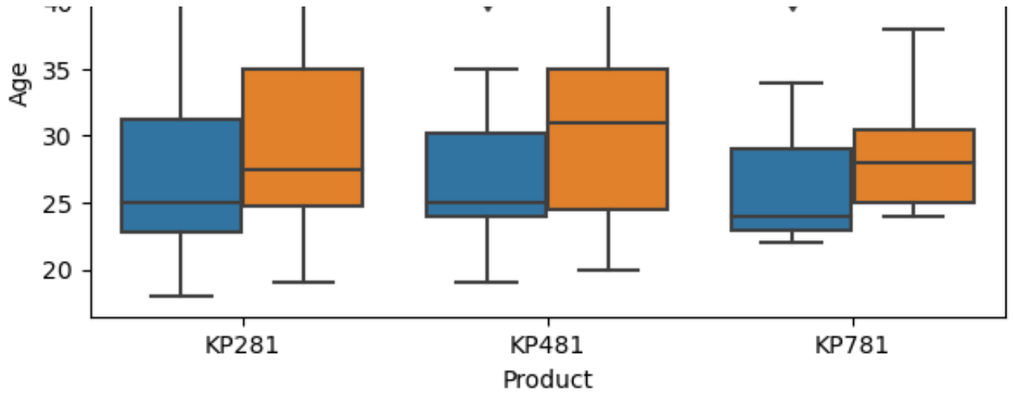


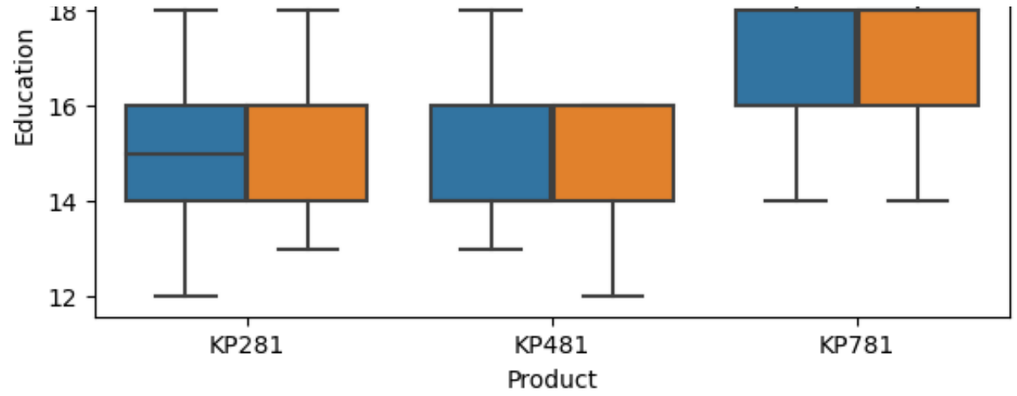


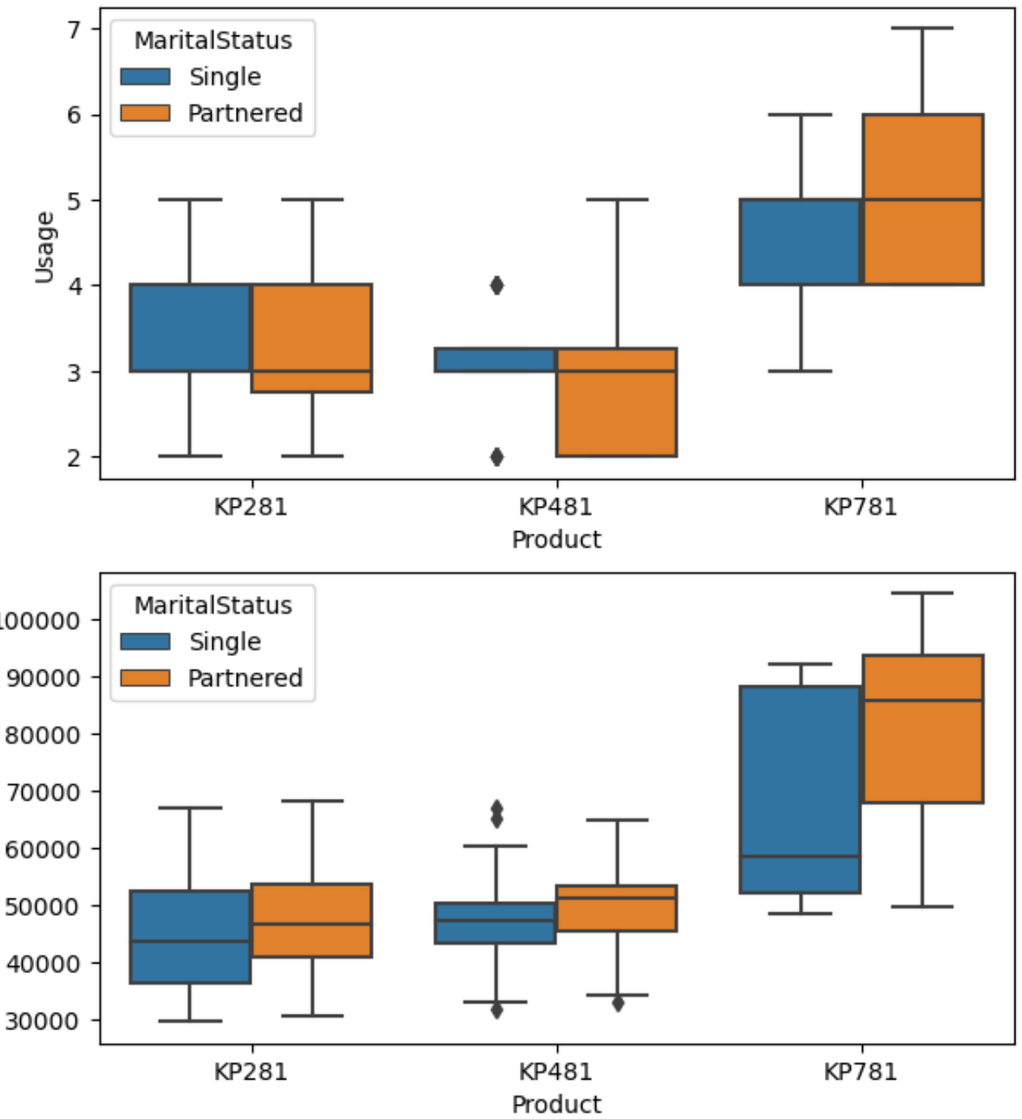
In [ ]:
```
"""
observation:
In age categeory female with 25-28 age group is preffering KP781 as compared to man with 25-30.
male & female both with high usage,high income and high running goal is preffering KP781.
for education categeory both male and female have almost equal distribution of all three product categeory.

"""
```

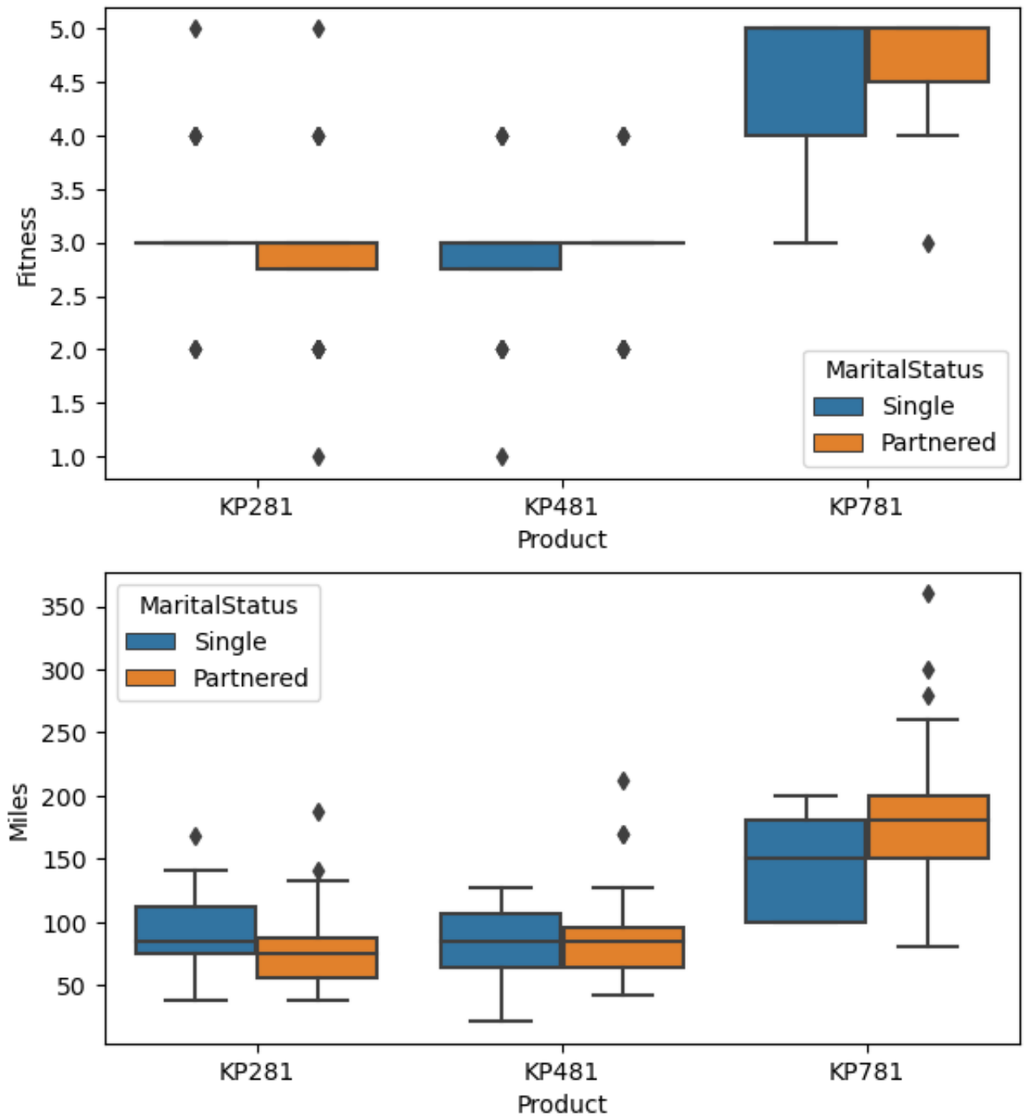
In [55]:
```
fig = plt.figure(figsize=(15,12))

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

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

plt.subplot(3,2,3)
sns.boxplot(x="Product",y="Usage",hue="MaritalStatus", data=df)

plt.subplot(3,2,4)
sns.boxplot(x="Product",y="Fitness",hue="MaritalStatus", data=df)

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

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


fig.suptitle('Multivariate Analysis/Box Plot')
plt.show()
```




Age
40
35
30
25
20
KP281
KP481
KP781
Product


Education
18
16
14
12
KP281
KP481
KP781
Product


MaritalStatus
Single
Partnered
Usage
7
6
5
4
3
2
KP281
KP481
KP781
Product


Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
MaritalStatus
Single
Partnered
KP281
KP481
KP781
Product

MaritalStatus
Single
Partnered
Income
100000
90000
80000
70000
60000
50000
40000
30000
KP281
KP481
KP781
Product

MaritalStatus
Single
Partnered
Miles
350
300
250
200
150
100
50
KP281
KP481
KP781
Product

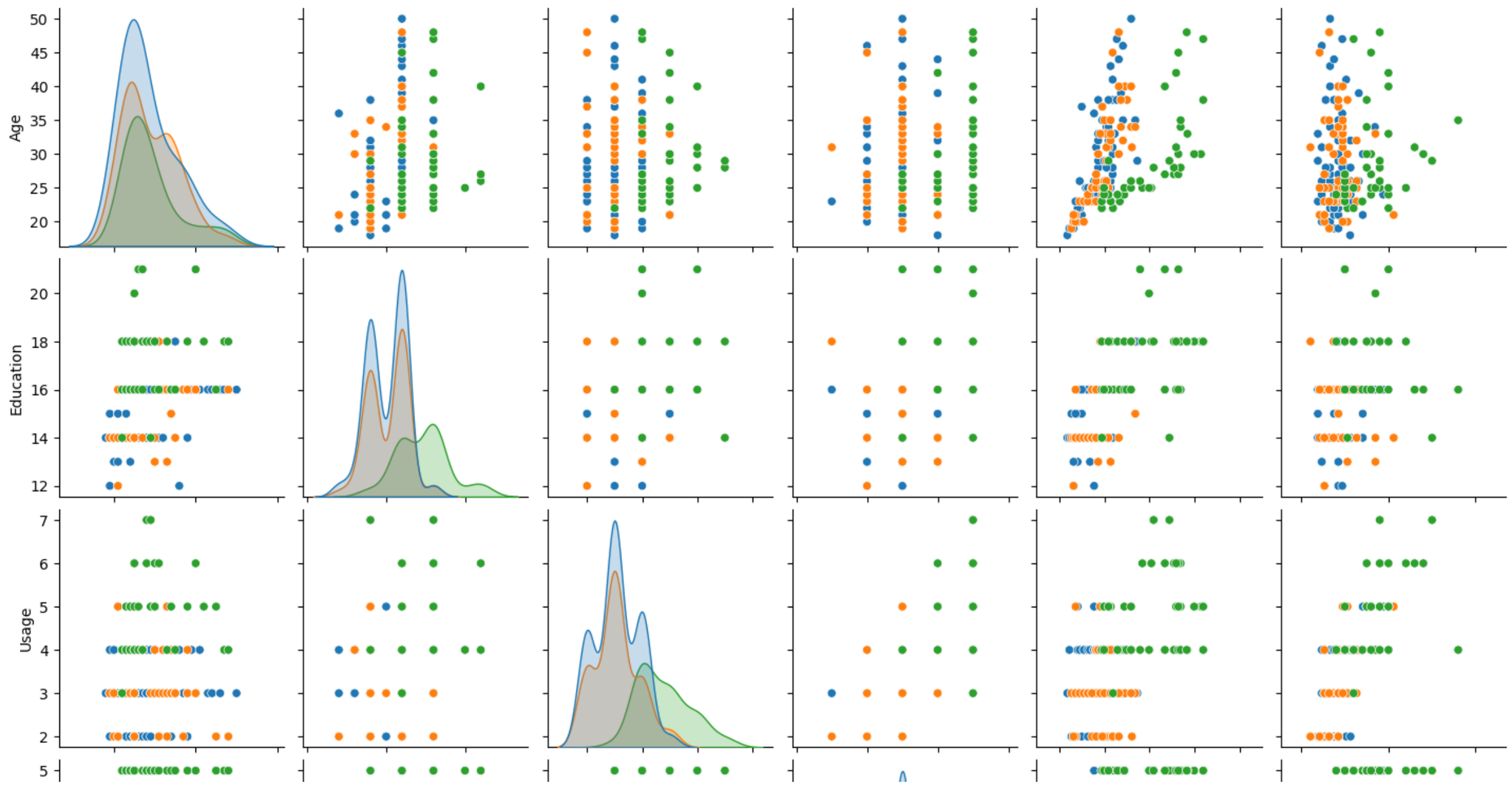
In [56]:
```
df_n=df.drop(["Product","Gender","MaritalStatus"],axis=1)
df_n.head()
```

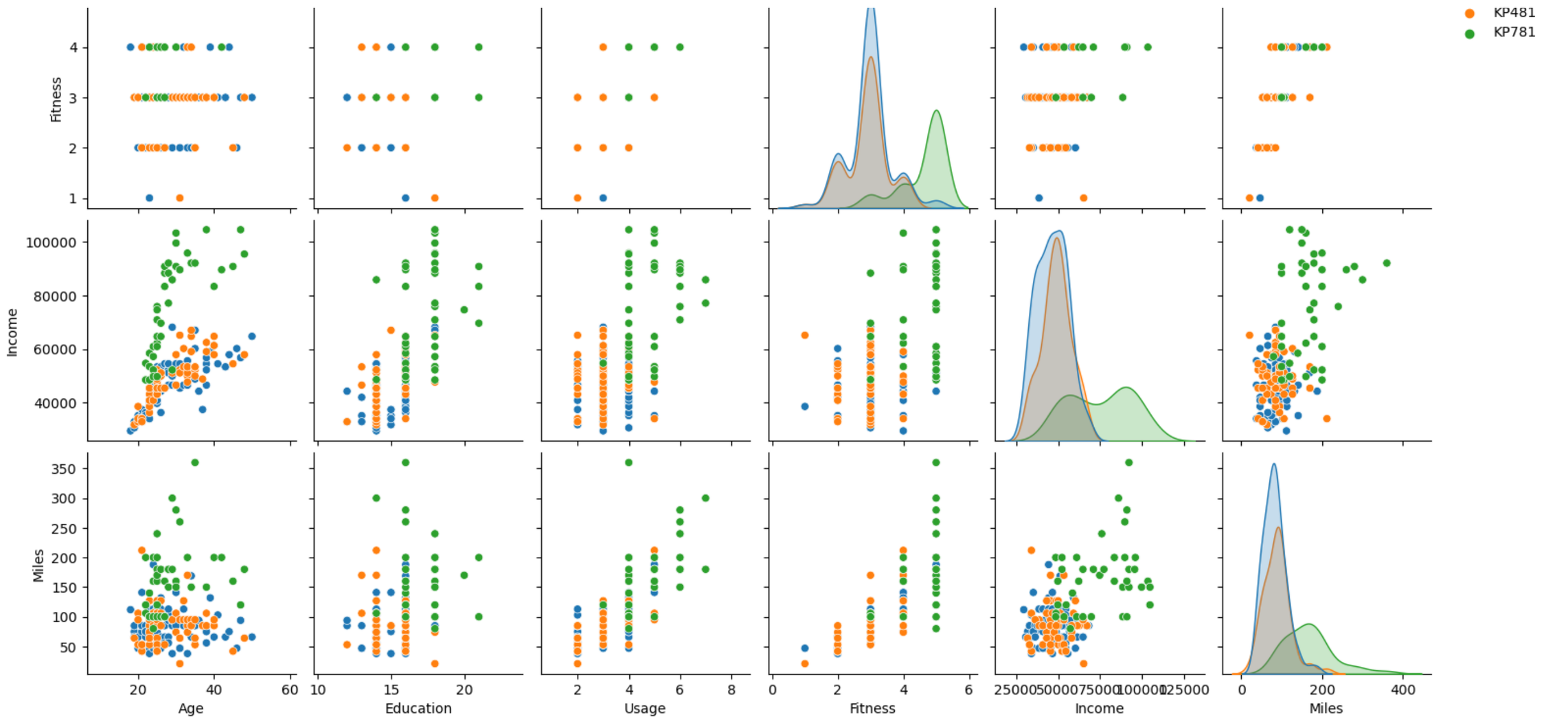
Out[56]:

	Age	Education	Usage	Fitness	Income	Miles
0	18	14	3	4	29562	112
1	19	15	2	3	31836	75
2	19	14	4	3	30699	66
3	19	12	3	3	32973	85
4	20	13	4	2	35247	47

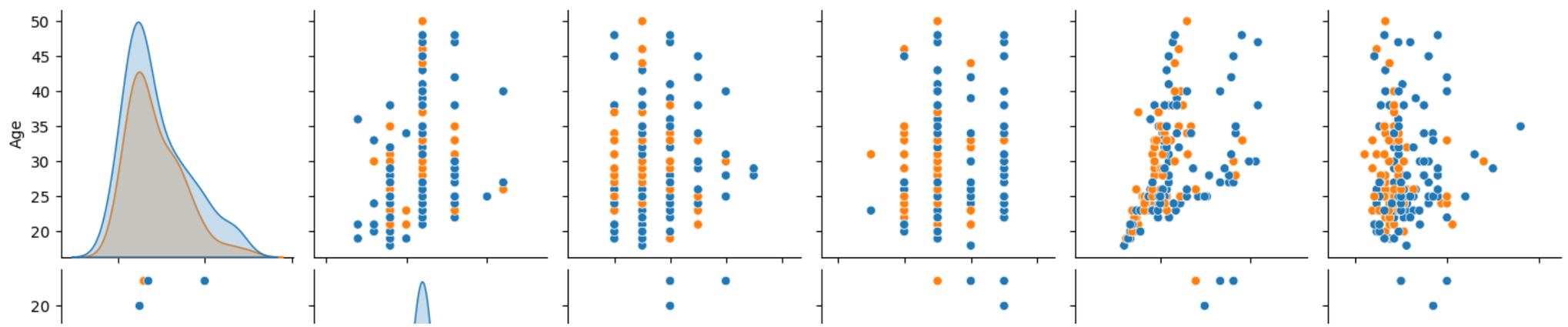
In [59]:
```
sns.pairplot(data=df,hue="Product")
```

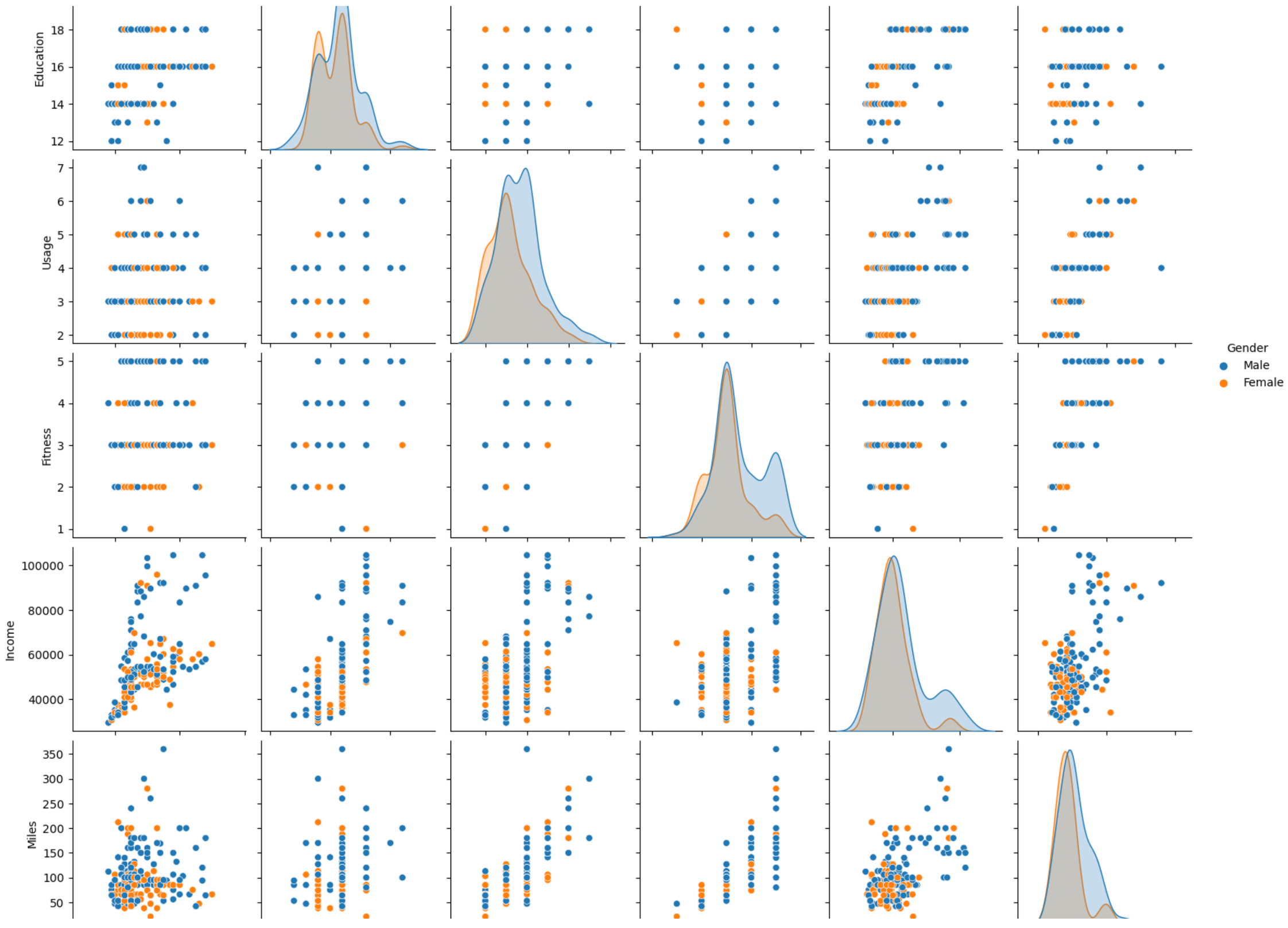
Out[59]: <seaborn.axisgrid.PairGrid at 0x7fc4cbd4bcd0>

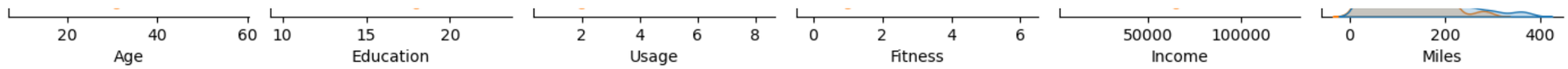




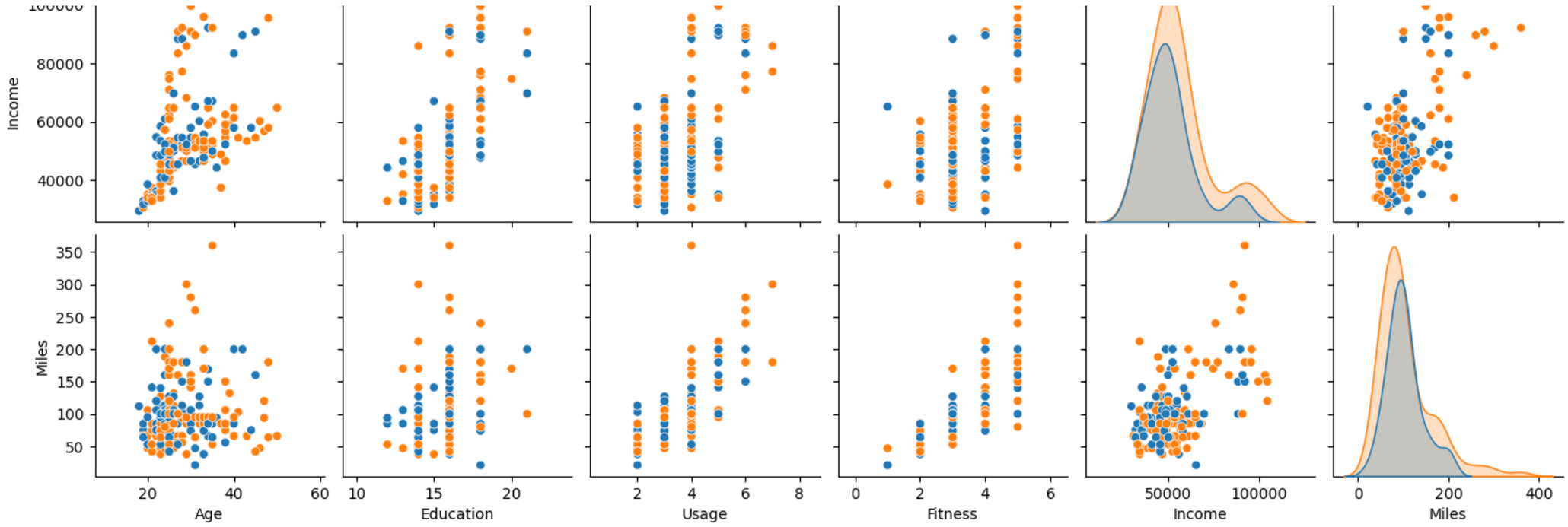
In [60]:
```
sns.pairplot(data=df,hue="Gender")
plt.show()
```




Education
Usage
Fitness
Income
Miles
Gender
Male
Female



In [61]:
```
sns.pairplot(data=df,hue="MaritalStatus")
plt.show()
```



In [62]:
```
df.corr()
```

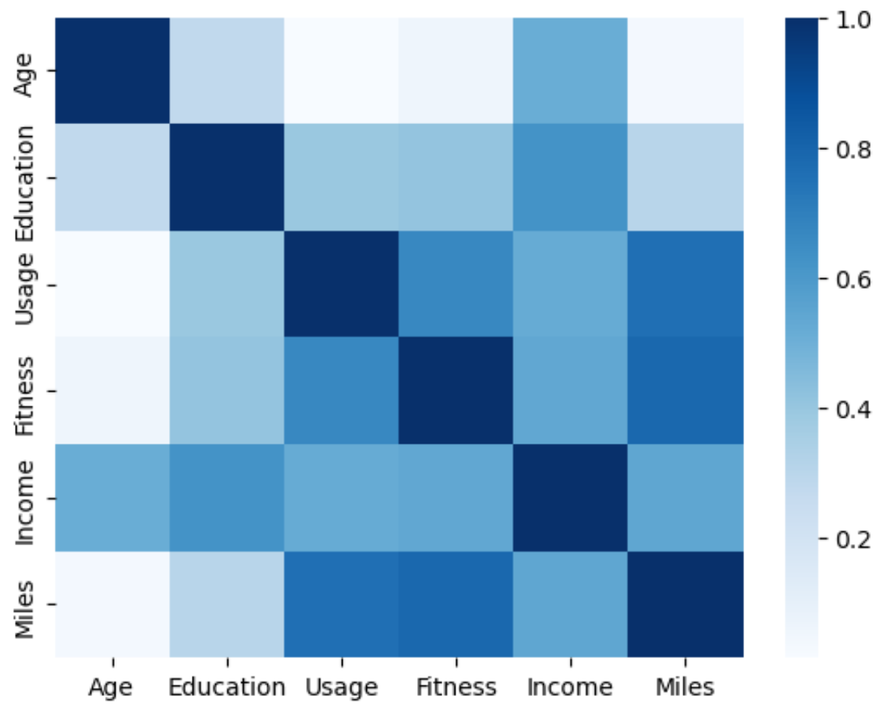
```
/var/folders/63/h28070vs2zx_1xl7yyzsblth0000gn/T/ipykernel_23406/1134722465.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is de
precated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df.corr()
```

Out[62]:

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

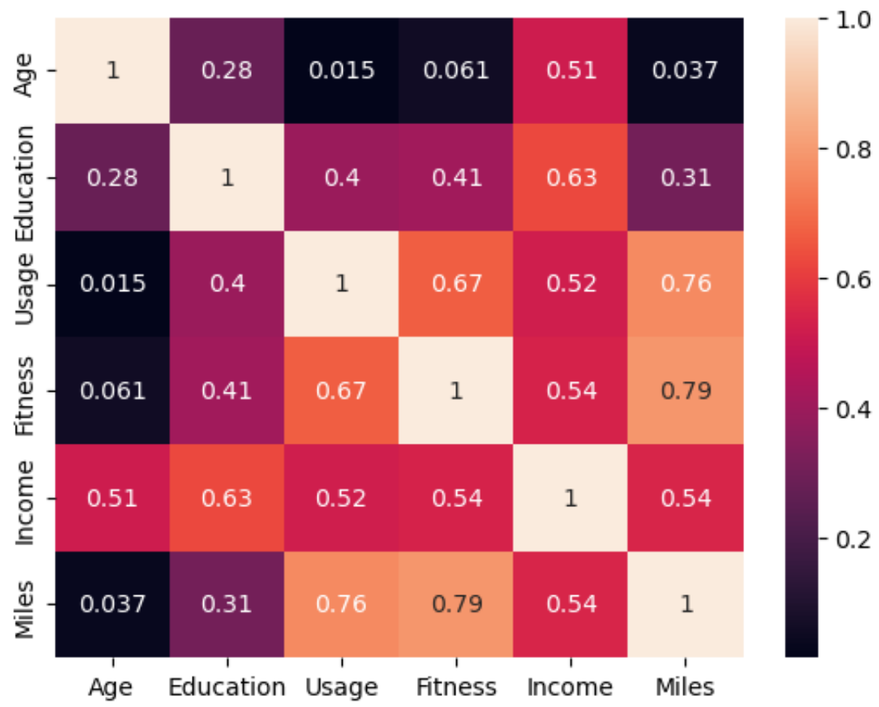
In [66]:
```
sns.heatmap(df.corr(),cmap="Blues")
plt.show()
```

```
/var/folders/63/h28070vs2zx_1xl7yyzsblth0000gn/T/ipykernel_23406/3112771539.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is de
precated. 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(df.corr(),cmap="Blues")
```



In [64]:
```
sns.heatmap(df.corr(),annot=True)
plt.show()
```

```
/var/folders/63/h28070vs2zx_1xl7yyzsblth0000gn/T/ipykernel_23406/2271070662.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is de
precated. 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(df.corr(),annot=True)
```



In [ ]:
```
#probabilities
```

In [85]:
```
df["Product"].value_counts(normalize=True)
```

Out[85]:
```
KP281    0.444444
KP481    0.333333
KP781    0.222222
Name: Product, dtype: float64
```

In [86]:
```
# Average usage of each product type by the customer
df.groupby('Product')['Usage'].mean()
```

Out[86]:
```
Product
KP281    3.087500
KP481    3.066667
KP781    4.775000
Name: Usage, dtype: float64
```

In [87]:
```
# Average Age of customer using each product
df.groupby('Product')['Age'].mean()
```

Out[87]:
```
Product
KP281    28.55
KP481    28.90
KP781    29.10
Name: Age, dtype: float64
```

In [88]:
```
# Average Education of customer using each product
df.groupby('Product')['Education'].mean()
```

Out[88]:
```
Product
KP281    15.037500
KP481    15.116667
KP781    17.325000
Name: Education, dtype: float64
```

In [89]:
```
# Average customer fitness rating for each product type purchased
df.groupby('Product')['Fitness'].mean()
```

Out[89]:
```
Product
KP281    2.9625
KP481    2.9000
KP781    4.6250
Name: Fitness, dtype: float64
```

In [ ]:
```
#conditional and marginal probability
```

In [ ]:
```
#marginal
```

In [90]:
```
pd.crosstab([df.Product],df.Gender,margins=True)
```

Out[90]:

Gender	Female	Male	All
Product			
KP281	40	40	80
KP481	29	31	60
KP781	7	33	40
All	76	104	180

In [91]:
```
np.round(((pd.crosstab(df.Product,df.Gender,margins=True))/180)*100,2)
```

Out[91]:

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

In [ ]:
```
"""
KP281 | Female = 22.22 %

KP481 | Female = 16.11 %

KP781 | Female = 3.89 %

KP281 | male = 22.22 %

KP481 | male = 17.22 %

KP781 | male = 18.33 %
"""
```

In [98]:
```
np.round(((pd.crosstab(df.Product,df.MaritalStatus,margins=True))/180)*100,2)
```

Out[98]:

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

In [ ]:
```
"""
KP281 | Partnered= 26.67 %

KP481 | Partnered = 20.00 %

KP781 | Partnered = 12.78 %

KP281 | Single = 17.78%

KP481 | Single = 13.33 %

KP781 | Single = 9.44 %
"""
```

In [ ]:
```
#conditional
```

In [92]:
```
np.round((pd.crosstab([df.Product],df.Gender,margins=True,normalize="columns"))*100,2)
```

Out[92]:

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

In [ ]:
```
"""
KP281 | Female = 52.63 %

KP481 | Female = 38.16 %

KP781 | Female = 9.21 %

KP281 | male = 38.46 %

KP481 | male = 29.81 %

KP781 | male = 31.73 %
"""
```

In [93]:
```
np.round((pd.crosstab([df.Product],df.MaritalStatus,margins=True,normalize="columns"))*100,2)
```

Out[93]:

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

In [ ]:
```
"""
KP281 | Partnered= 44.86 %

KP481 | Partnered = 33.64 %

KP781 | Partnered = 9.21 %

KP281 | Single = 43.84 %

KP481 | Single = 32.88 %

KP781 | Single = 23.29 %
"""
```

In [ ]:

In [ ]: