In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df = pd.read_csv(r'C:\Users\ssd\OneDrive\Documents\Numpy notes\11_Pandas_3\11_Pandas_3\aerofit_treadmill.csv')
```

In [4]:
```
df
```

Out[4]:

	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
...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200
177	KP781	45	Male	16	Single	5	5	90886	160
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

180 rows × 9 columns

In [7]:
```
# No null values
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 [8]:
```
df.describe()
```

Out[8]:

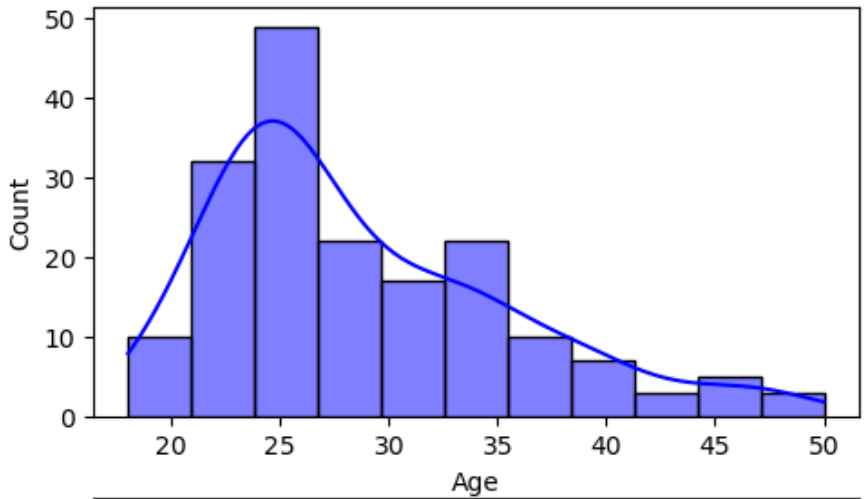
	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
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

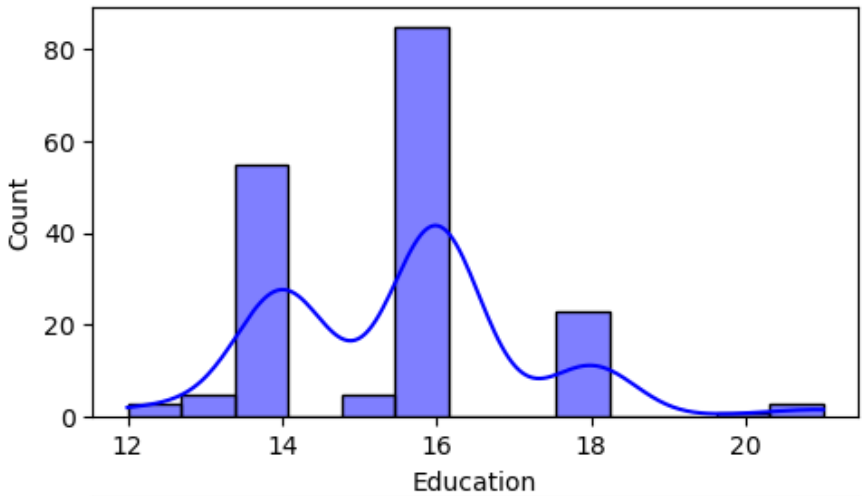
In [12]:
```
'''
Insights:

-> 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.
-> Most of the people are having 16 years of education i.e. 75% of persons are having education <= 16 years.
-> Out of 180 rows, 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.

'''
```

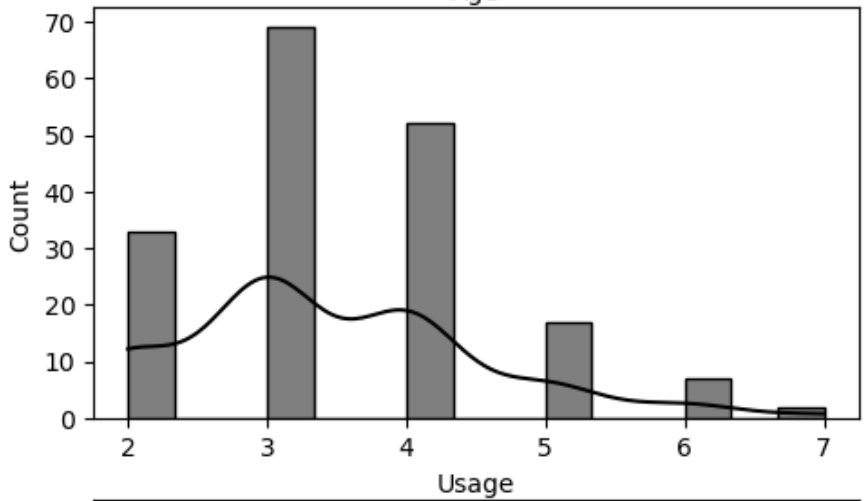
Out[12]:
```
"\nInsights:\n\n-> There are no missing values in the data.\n-> There are 3 unique products in the dataset.\n-> KP281 is the most frequent product.\n-> Minimum & Maximum age of the person is 18 & 50, mean is 28.79 and 75% of persons have age less than or equal to 33.\n-> Most of the people are having 16 years of education i.e. 75% of persons are having education <= 16 years.\n-> Out of 180 rows, 104's gender is Male and rest are the female.\n-> Standard deviation for Income & Miles is very high. These variables might have the outliers in it.\n\n"
```

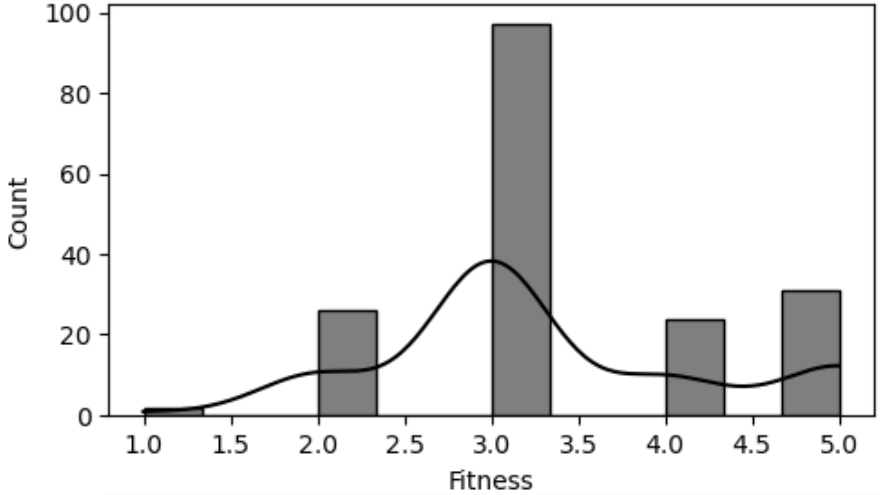
In [ ]:
```
# Univariate analysis
# Checking for the distribution of each colmmn in the data set
# Validating the factors like age, Education, usage, fitness, income, miles using a subplot
# entire univariate analysis are categorized and divided into the sub plots below
```

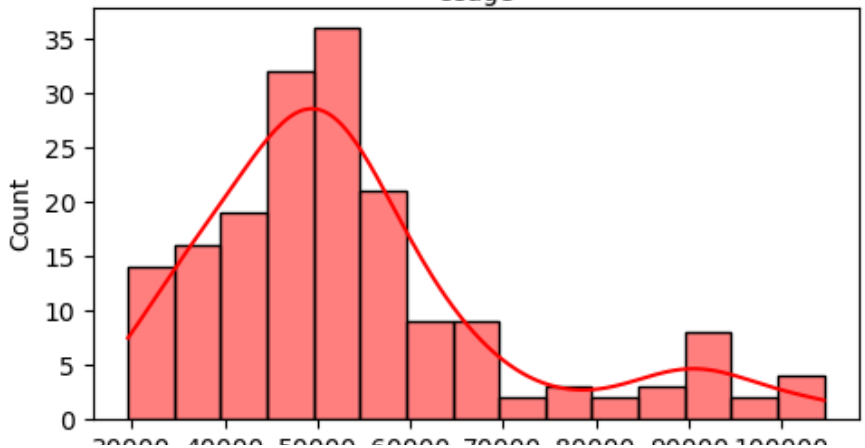
In [13]:
```
fig, axis = plt.subplots(3,2 , figsize = (12,10))
sns.histplot(data = df, x='Age',kde = True, ax = axis[0,0] ,color = 'blue')
sns.histplot(data = df, x='Education',kde = True, ax = axis[0,1] ,color = 'blue')
sns.histplot(data = df, x='Usage',kde = True, ax = axis[1,0] ,color = 'black')
sns.histplot(data = df, x='Fitness',kde = True,ax = axis[1,1] , color = 'black')
sns.histplot(data = df, x='Income',kde = True,ax = axis[2,0]  ,color = 'red')
sns.histplot(data = df, x='Miles',kde = True, ax = axis[2,1] ,color = 'red')
plt.show()
```

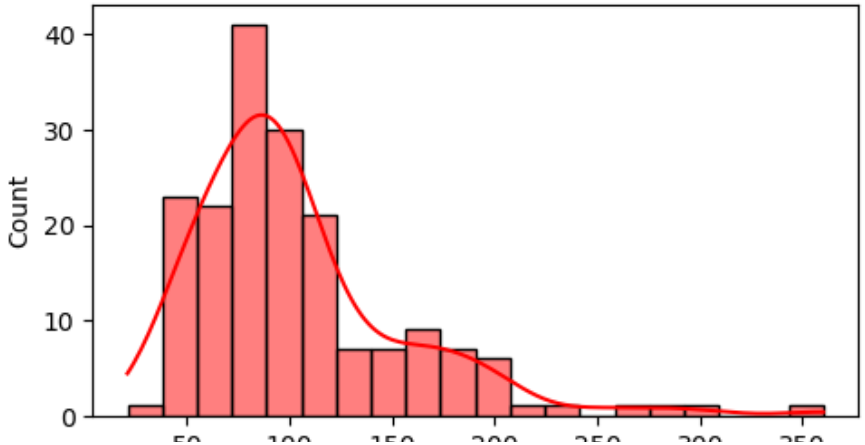

Count
Age


Count
Education


Count
Usage

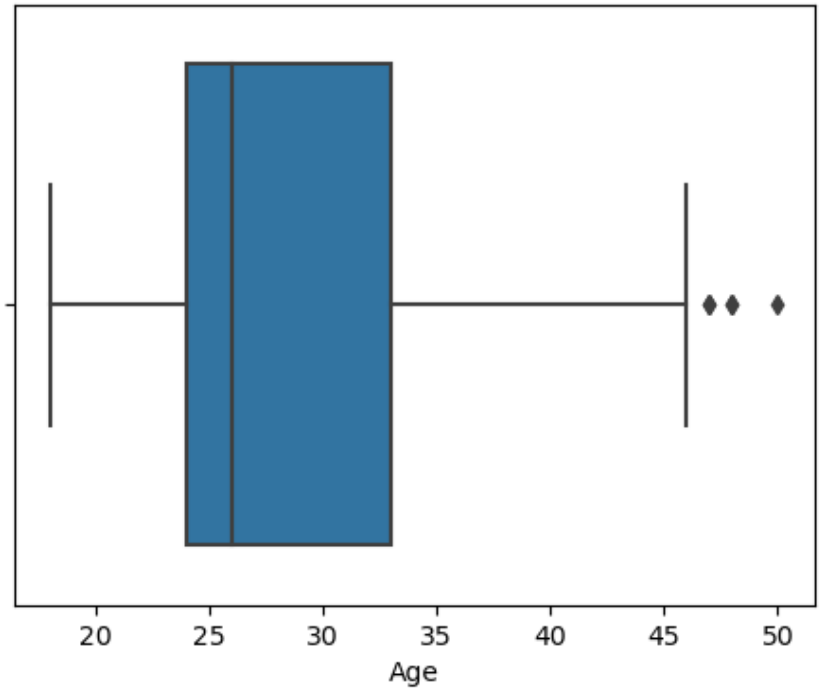

Count
Fitness

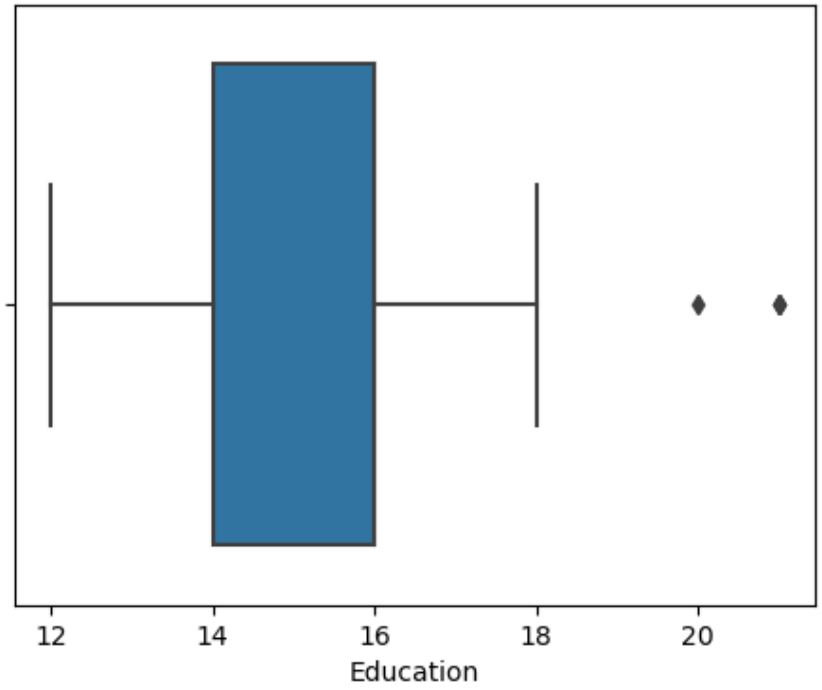

Count

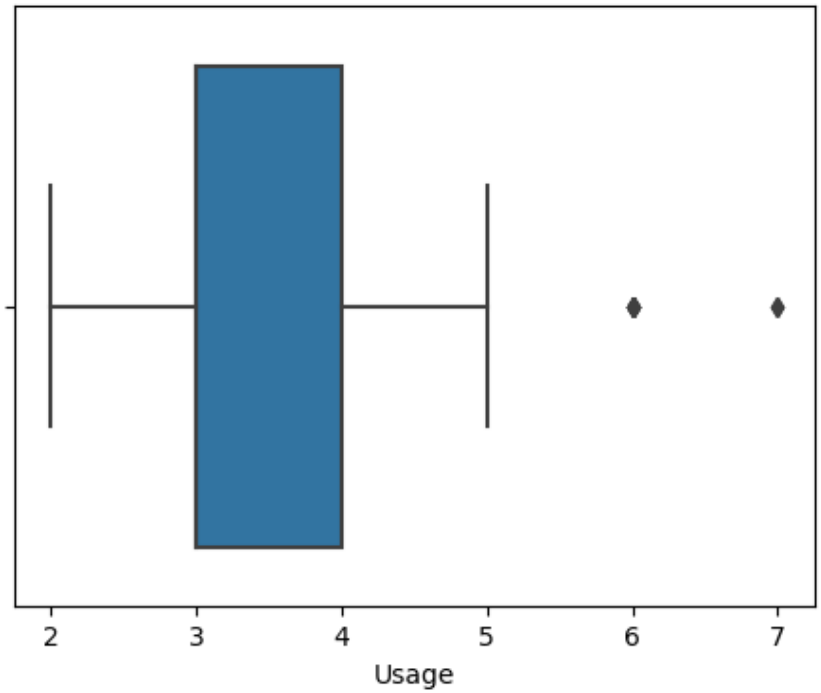

Count

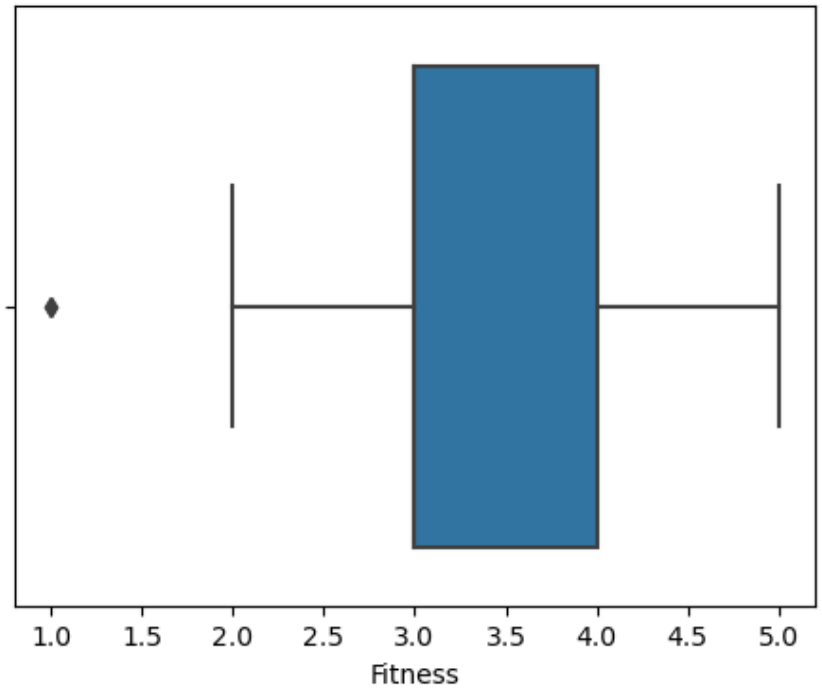
In [ ]:
```
# checking for outliers among the multiple columns in the data set
# there are some heavy outliers in miles and incomes columns
```

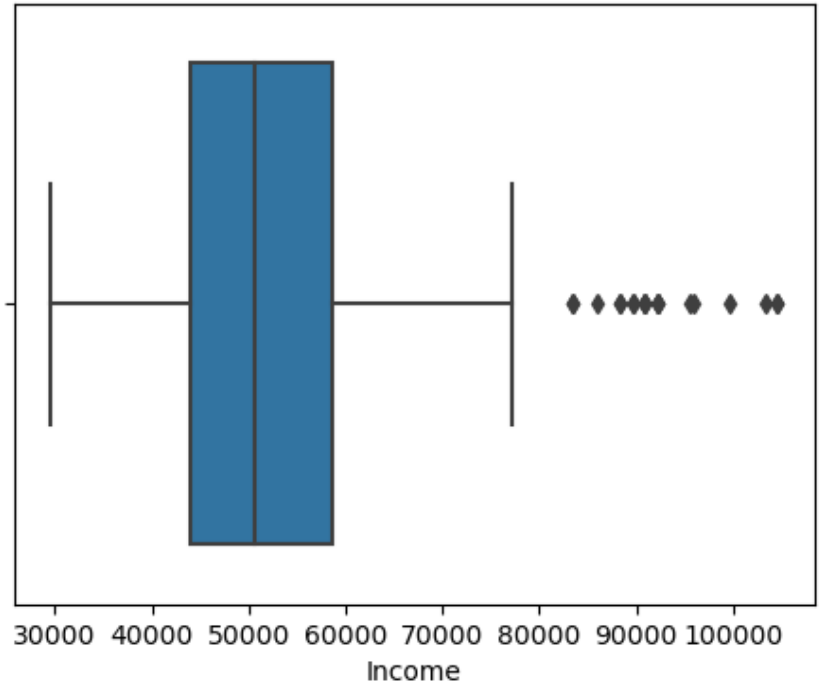
In [14]:
```
fig, axis = plt.subplots(3,2, figsize = (12,14))
sns.boxplot(data = df, x = 'Age',orient = 'h',ax = axis[0,0])
sns.boxplot(data = df, x = 'Education',orient = 'h',ax = axis[0,1])
sns.boxplot(data = df, x = 'Usage',orient = 'h',ax = axis[1,0])
sns.boxplot(data = df, x = 'Fitness',orient = 'h',ax = axis[1,1])
sns.boxplot(data = df, x = 'Income',orient = 'h',ax = axis[2,0])
sns.boxplot(data = df, x = 'Miles',orient = 'h',ax = axis[2,1])
plt.show()
```

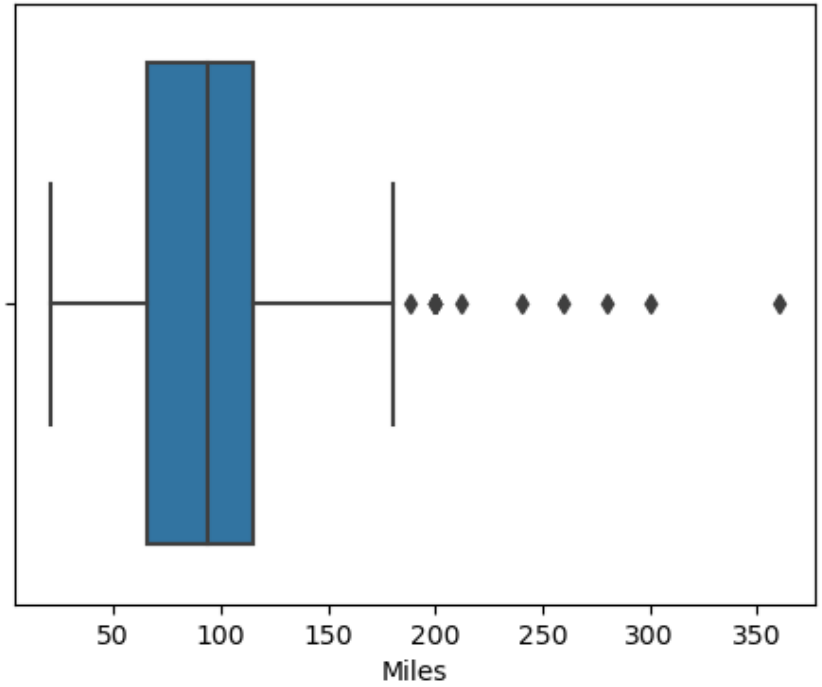

20
25
30
35
40
45
50
Age


12
14
16
18
20
Education


2
3
4
5
6
7
Usage


1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fitness

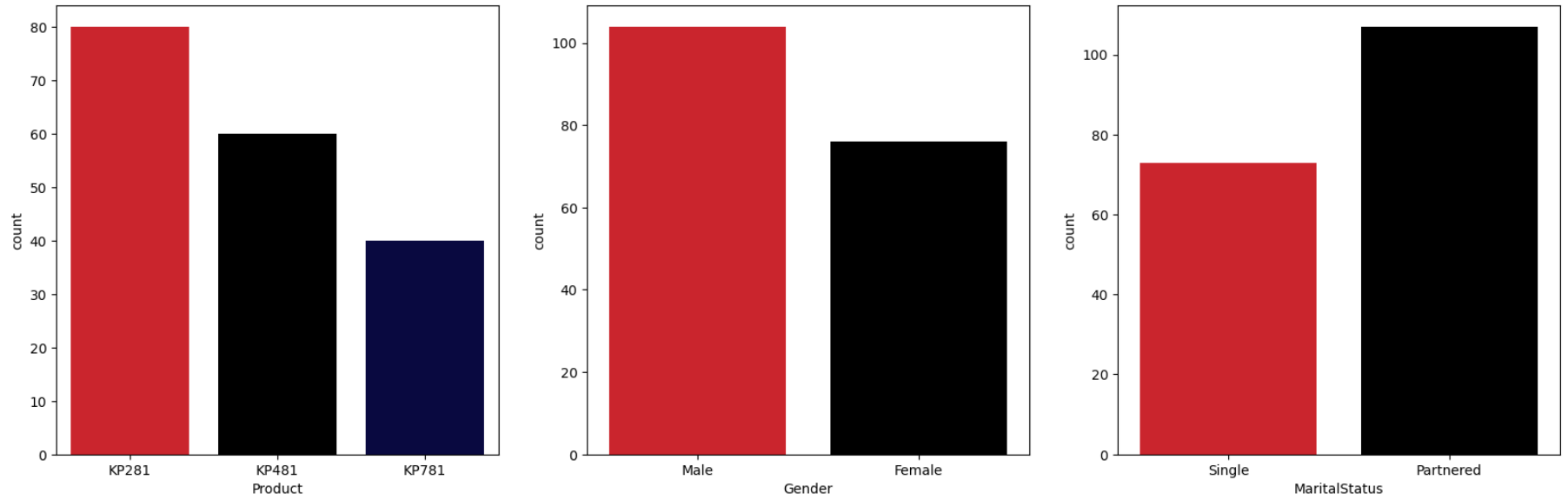




In [ ]:
```
'''
Insights:

-> from box plot outlier detection it is evident that income and miles have large number of outliers
-> Age, education and usage has less outliers compared to income and miles

'''
```

In [ ]:
```
# checking for the distribution of qualitative data in the data set
# Product, Gender, Marital Status
```

In [15]:
```
fig, axis = plt.subplots(1,3, figsize = (20,6))
sns.countplot(data = df, x = 'Product',ax = axis[0],palette=['#E50914',"#000000","#000049"])
sns.countplot(data = df, x = 'Gender',ax = axis[1],palette=['#E50914',"#000000"])
sns.countplot(data = df, x = 'MaritalStatus',ax = axis[2],palette=['#E50914',"#000000"])
plt.show()
```



In [ ]:
```
'''
Insights:

-> KP281 is the largest selling product among the tread mill cateogry
-> out of total propotion of gender data, we have dominant amount males in the gender category
-> comapared to single, there are more number of people who are married and interested in buying the tread mill equipm

'''
```

In [16]:

```
# checking for the categorized data among Gender, Marital Status and Product in the data set.

melted_data = df[['Product','Gender','MaritalStatus']].melt()

melted_data.groupby(['variable','value'])[['value']].count() / len(df)
```

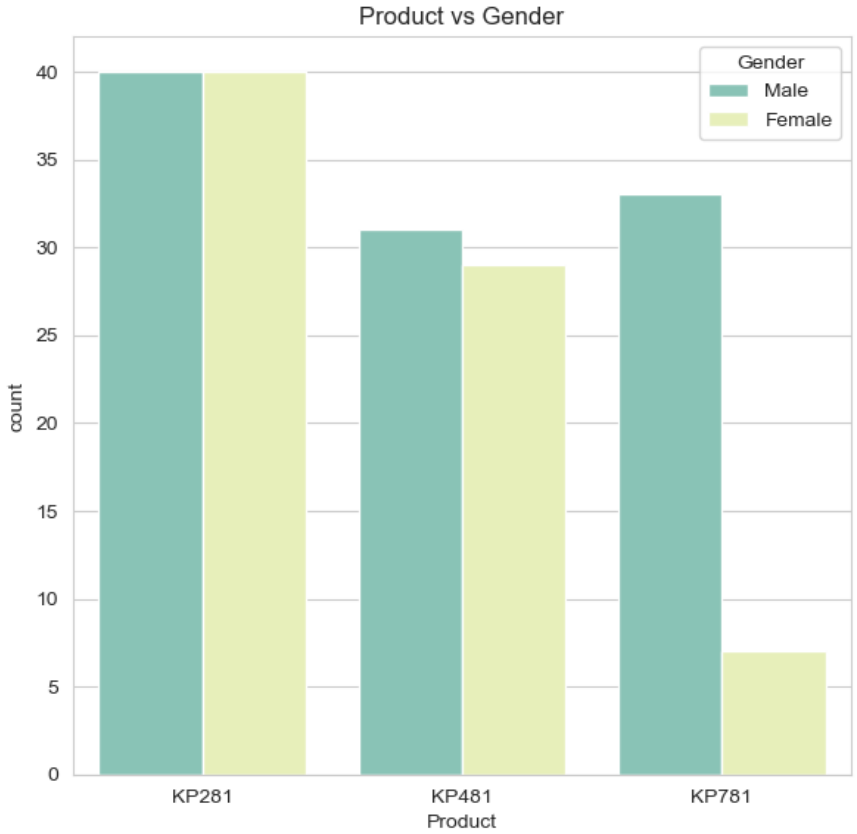
Out[16]:

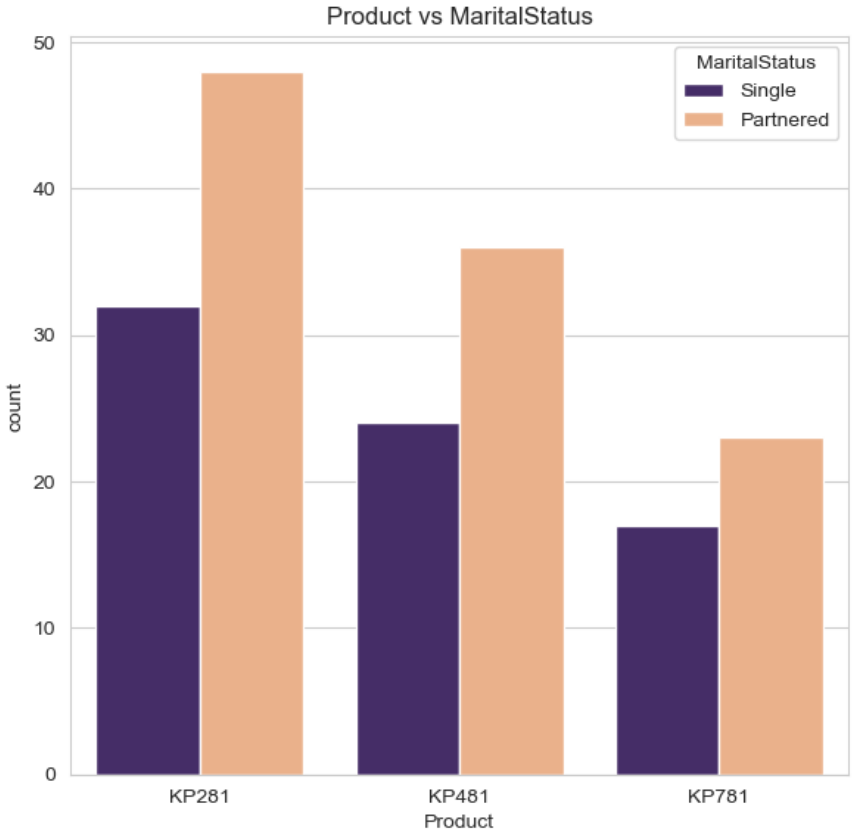
		value
variable	**value**	
Gender	**Female**	0.422222
	Male	0.577778
MaritalStatus	**Partnered**	0.594444
	Single	0.405556
Product	**KP281**	0.444444
	KP481	0.333333
	KP781	0.222222

In [ ]:
```
'''
Insights:

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.

Gender:

-> 57.78% of the customers are Male.
-> 42% of the customers are female

MaritalStatus:

-> 59.44% of the customers are Partnered.
-> 40.12% of the customers are single

'''
```

In [ ]:
```
# checking the relation between product vs age and product vs Gender and product vs Marital Status
```

In [17]:
```
sns.set_style(style = 'whitegrid')
fig, axs= plt.subplots(1,2,figsize=(15,6.5))
sns.countplot(data=df,x="Product",hue="Gender",palette=["#7fcdbb","#edf8b1"],ax=axs[0])
axs[0].set_title("Product vs Gender")
sns.countplot(data=df,x="Product",hue="MaritalStatus",palette=['#432371',"#FAAE7B"],ax=axs[1])
axs[1].set_title("Product vs MaritalStatus")
plt.show()
```





In [18]:
```
# checking for effect of age on purchasing the product

effect_of_age_on_product = pd.crosstab([df.MaritalStatus, df.Age],
                                       df.Product, margins = True)
effect_of_age_on_product
```

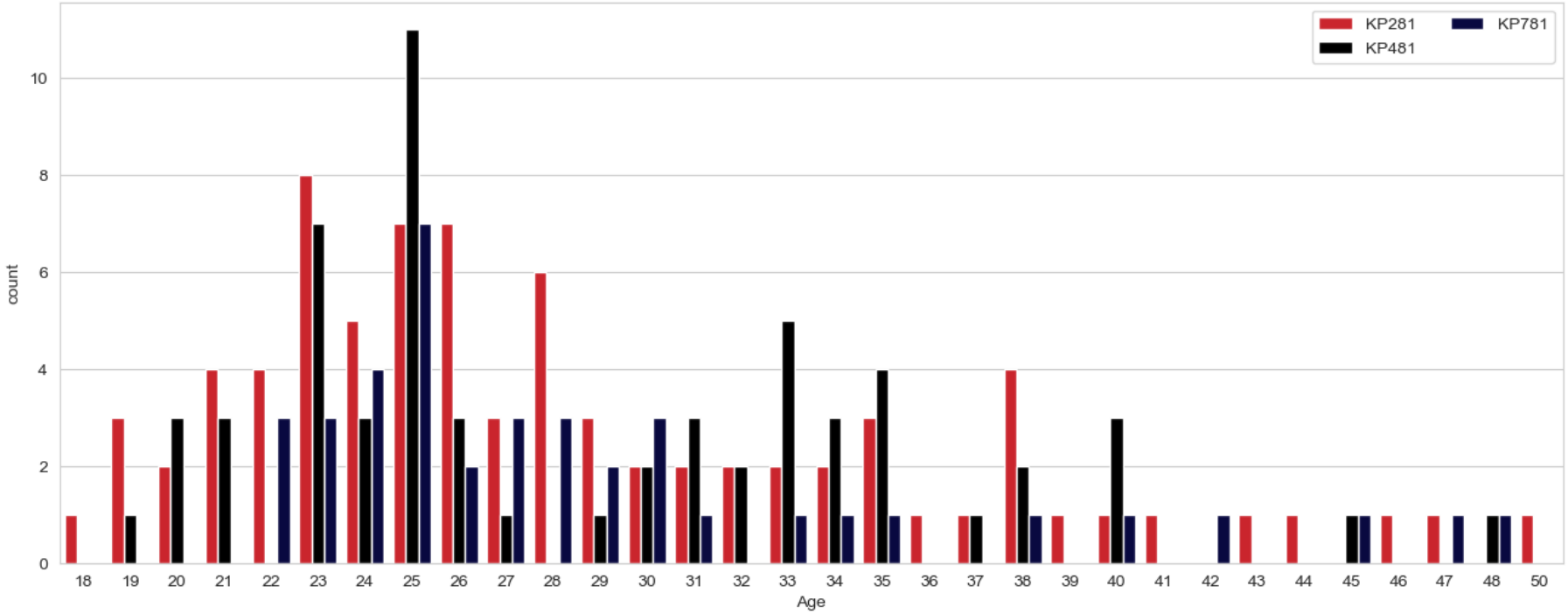
Out[18]:

Product		KP281	KP481	KP781	All
MaritalStatus	**Age**				
Partnered	**19**	1	0	0	1
	20	2	1	0	3
	21	2	3	0	5
	22	1	0	0	1
	23	4	5	0	9
	24	2	0	1	3
	25	5	6	7	18
	26	5	1	1	7
	27	2	0	2	4
	28	4	0	2	6
	29	3	1	1	5
	30	1	0	3	4
	31	1	2	1	4
	32	1	1	0	2
	33	1	4	1	6
	34	0	2	0	2
	35	2	3	1	6
	37	1	1	0	2
	38	3	2	1	6
	39	1	0	0	1
	40	1	2	0	3
	41	1	0	0	1
	43	1	0	0	1
	45	0	1	0	1
	46	1	0	0	1

Product	KP281	KP481	KP781	All
MaritalStatus Age				
47	1	0	1	2
48	0	1	1	2
50	1	0	0	1

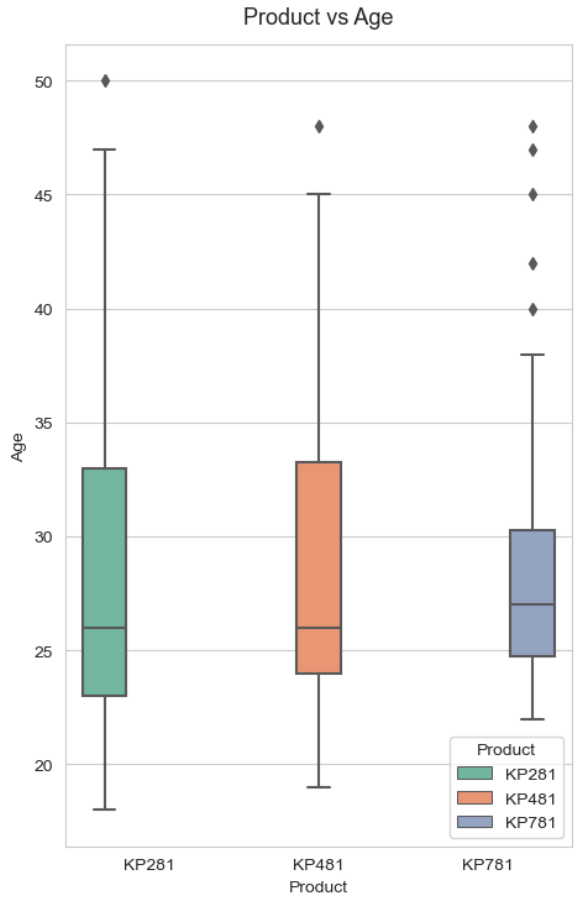
Product		KP281	KP481	KP781	All
MaritalStatus	Age				
Single	18	1	0	0	1
	19	2	1	0	3
	20	0	2	0	2
	21	2	0	0	2
	22	3	0	3	6
	23	4	2	3	9
	24	3	3	3	9
	25	2	5	0	7
	26	2	2	1	5
	27	1	1	1	3
	28	2	0	1	3
	29	0	0	1	1
	30	1	2	0	3
	31	1	1	0	2
	32	1	1	0	2
	33	1	1	0	2
	34	2	1	1	4
	35	1	1	0	2
	36	1	0	0	1
	38	1	0	0	1
	40	0	1	1	2
	42	0	0	1	1
	44	1	0	0	1
	45	0	0	1	1
All		80	60	40	180

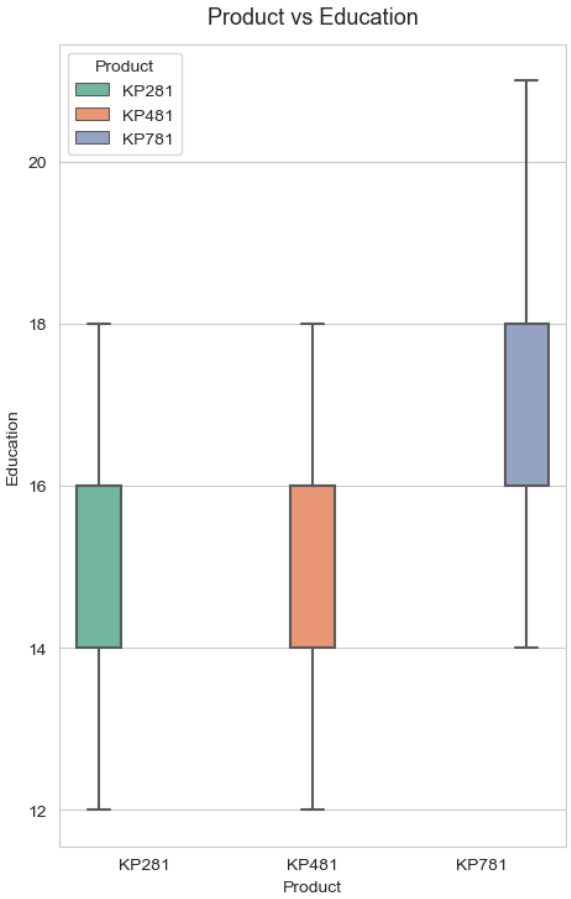
In [19]:
```
plt.figure(figsize = (16,6))
sns.countplot(data = df, x = 'Age', hue = 'Product',palette=['#E50914',"#000000",'#000049'])
plt.legend(loc = 'upper right',frameon = True, ncol = 2)
plt.show()
```

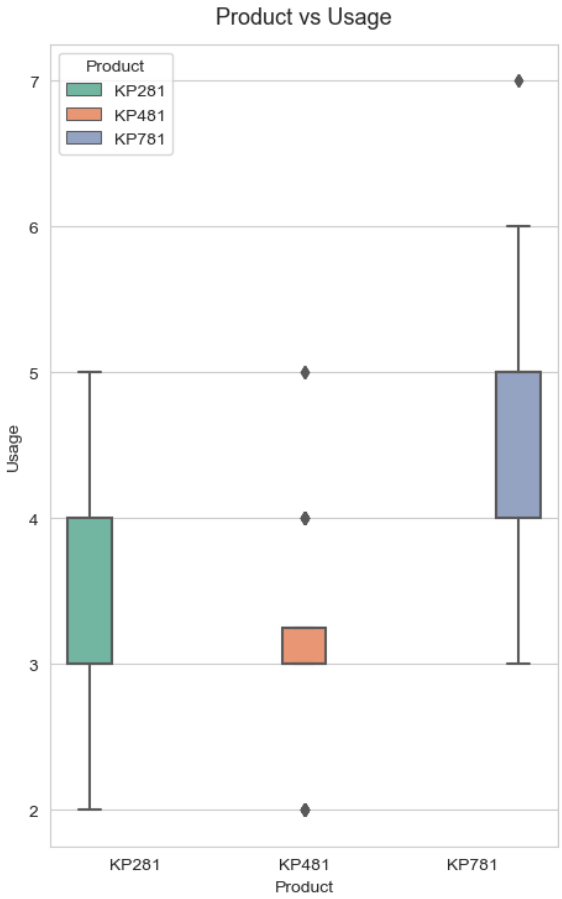


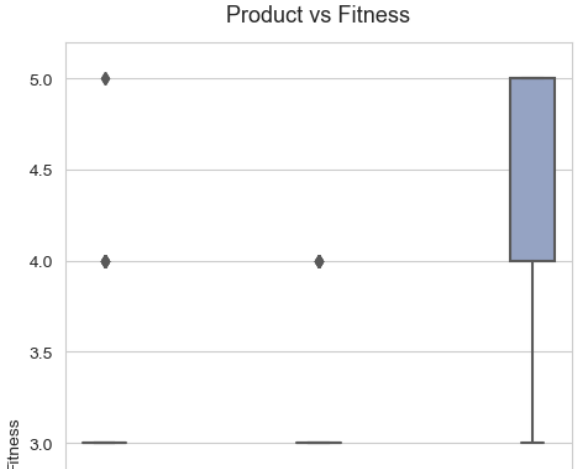
In [ ]:
```
'''
Insights:

Product vs Gender:

-> Equal number of males and females have purchased KP281 product and Almost same for the product KP481
-> Most of the Male customers have purchased the KP781 product.

Product vs MaritalStatus:

-> Customer who is Partnered, is more likely to purchase the product.

'''
```

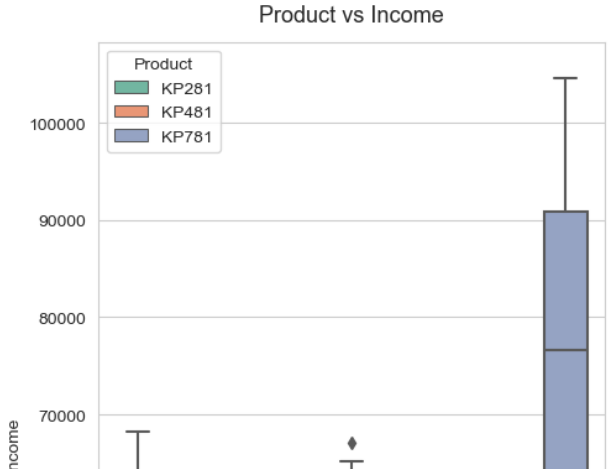
In [92]:
```
# performing bivariate analysis on multiple factors that contribute to the purchase of product

var= ['Age','Education','Usage','Fitness','Income','Miles']
sns.set_style("whitegrid")
fig,axs=plt.subplots(2,3,figsize=(18,12))
fig.subplots_adjust(top=1.3)
count=0
for i in range(2):
    for j in range(3):
        sns.boxplot(data=df,x='Product',y=var[count],ax=axs[i,j],hue='Product',palette="Set2")
        axs[i,j].set_title(f"Product vs {var[count]}",pad=12,fontsize=13)
        count+=1
```

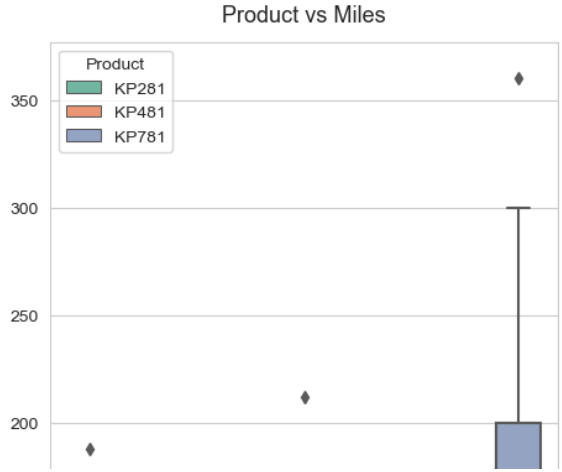

Product vs Age
Age
Product
KP281
KP481
KP781

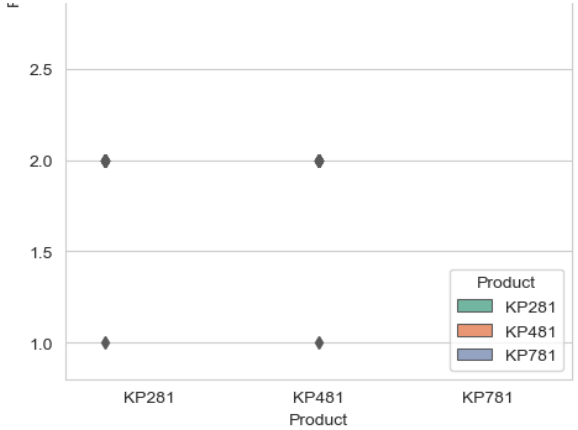

Product vs Education
Education
Product
KP281
KP481
KP781

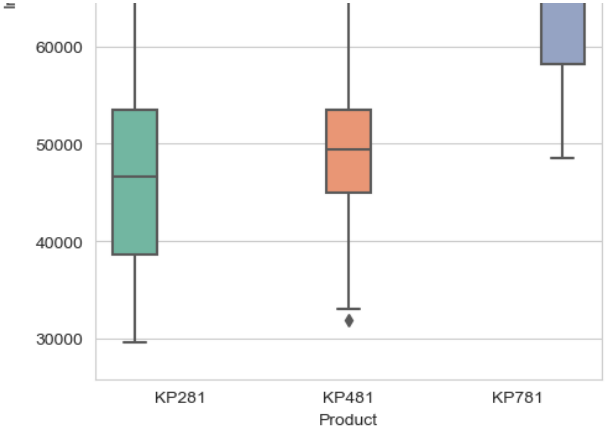

Product vs Usage
Usage
Product
KP281
KP481
KP781

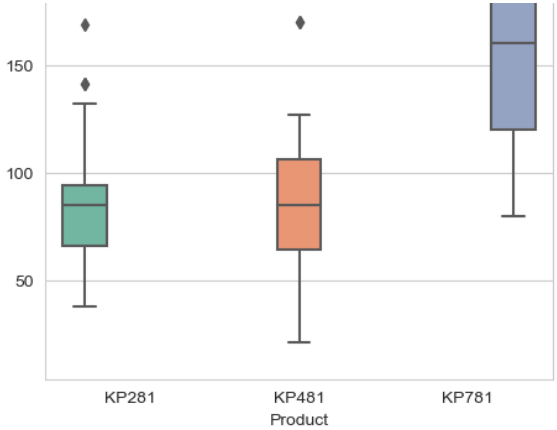

Product vs Fitness


Product vs Income
Product
KP281
KP481
KP781


Product vs Miles
Product
KP281
KP481
KP781


2.5
2.0
1.5
1.0
Product
KP281
KP481
KP781
KP281
KP481
KP781
Product


60000
50000
40000
30000
KP281
KP481
KP781
Product


150
100
50
KP281
KP481
KP781
Product

In [ ]:

```
'''

Insights:

Product vs 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

Product vs Education:

-> Customers whose Education is greater than 16, have more chances to purchase the KP781 product.
-> While the customers with Education less than 16 have equal chances of purchasing KP281 or KP481.

Product vs Usage:

-> Customers who are planning to use the treadmill greater than 4 times a week, are more likely to purchase the KP781
-> While the other customers are likely to purchasing KP281 or KP481.

Product vs Fitness:

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

Product vs Income:

-> Higher the Income of the customer (Income >= 60000), higher the chances of the customer to purchase the KP781 produ

Product vs Miles:

-> If the customer expects to walk/run greater than 120 Miles per week, it is more likely that the customer will buy K

'''
```

In [93]:
```
# performing marginal and conditional probability to ensure our analysis is number proof before we reach the business.

pd.concat(
    [ df.Product.value_counts(), df.Product.value_counts(normalize=True)
    ],
    keys=['counts', 'Marginal_Prob'],
    axis=1,
)
```

Out[93]:

	counts	Marginal_Prob
KP281	80	0.444444
KP481	60	0.333333
KP781	40	0.222222

In [94]:
```
def p_prod_given_gender(gender, print_marginal=False):
    if gender!= "Female" and gender!= "Male":
        return "Invalid gender value."
    df1= pd.crosstab(index=df['Gender'],columns=[df['Product']])
    p_781= df1['KP781'][gender] / df1.loc[gender].sum()
    p_481= df1['KP481'][gender] / df1.loc[gender].sum()
    p_281= df1['KP281'][gender] / df1.loc[gender].sum()
    if print_marginal:
        print(f"P(Male): {df1.loc['Male'].sum()/len(df):.2f}")
        print(f"P(Female): {df1.loc['Female'].sum()/len(df):.2f}")
    print(f"P(KP781/{gender}):{p_781:.2f}")
    print(f"P(KP481/{gender}):{p_481:.2f}")
    print(f"P(KP281/{gender}):{p_281:.2f}\n")
p_prod_given_gender('Male',True)
p_prod_given_gender('Female')
```

```
P(Male): 0.58
P(Female): 0.42
P(KP781/Male):0.32
P(KP481/Male):0.30
P(KP281/Male):0.38

P(KP781/Female):0.09
P(KP481/Female):0.38
P(KP281/Female):0.53
```

In [95]:
```
def p_prod_given_MaritalStatus(status, print_marginal=False):
    if status!= "Single" and status!= "Partnered":
        return " invalid MaritalStatus value."
    df1= pd.crosstab(index=df['MaritalStatus'],columns=[df['Product']])
    p_781= df1['KP781'][status] / df1.loc[status].sum()
    p_481= df1['KP481'][status] / df1.loc[status].sum()
    p_281= df1['KP281'][status] / df1.loc[status].sum()
    if print_marginal:
        print(f"P(Single): {df1.loc['Single'].sum()/len(df):.2f}")
        print(f"P(Partnered): {df1.loc['Partnered'].sum()/len(df):.2f}\n")

    print(f"P(KP781/{status}):{p_781:.2f}")
    print(f"P(KP481/{status}):{p_481:.2f}")
    print(f"P(KP281/{status}):{p_281:.2f}\n")

p_prod_given_MaritalStatus('Single',True)
p_prod_given_MaritalStatus('Partnered')
```

```
P(Single): 0.41
P(Partnered): 0.59

P(KP781/Single):0.23
P(KP481/Single):0.33
P(KP281/Single):0.44

P(KP781/Partnered):0.21
P(KP481/Partnered):0.34
P(KP281/Partnered):0.45
```

In [ ]:
```
'''
Recommendations:

-> KP781 should be a marketed as a premium model and marketing it to high income groups and educational over 20 years
segmenets could result in more sales.

-> Aerofit should conduct market research to determine if it can attract customers with income under 40000 to expand i
customer base.

-> The KP781  is a premium model, so it is ideally suited for sport people who have a high weekly average mileage

'''
```