In [2]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from scipy.stats import norm
from scipy.stats import binom
import matplotlib.pyplot as plt
```

1. Upload Data:

In [3]:

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

```
--2023-03-12 05:23:23--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 108.157.172.176, 108.157.172.10, 108.157.172.173, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|108.157.172.176|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7279 (7.1K) [text/plain]
Saving to: 'aerofit.csv'

aerofit.csv         100%[===================>]   7.11K  --.-KB/s    in 0s

2023-03-12 05:23:23 (1.30 GB/s) - 'aerofit.csv' saved [7279/7279]
```

In [4]:

```
!ls
```

```
aerofit.csv  sample_data
```

In [5]:

```
df = pd.read_csv("aerofit.csv")
```

2. Problem Statement and Evaluate Provided Data Set:

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.**

Problem Statement:

> **The Aerofit team wanted to determine which group of consumers tend to buy certain range of fitness equipment based on various variables like gender, usage, income, and so forth. This is because they want to be able to make better suggestions to new clients. On the basis of the data, we also need to offer analysis and suggestions that will aid Aerofit in making business decisions.**

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	20	Male	14	Partnered	3	3	35247	66
3	KP281	19	Male	12	Single	3	3	32973	85
4	KP281	20	Male	13	Partnered	4	2	35247	47

In [7]:

```
print(f"Aerofit dataset Basic Information:\nNumber of Records: {df.shape[0]}\nNumber of Column Data per Record: {df.shape[1]}")
```

```
Aerofit dataset Basic Information:
Number of Records: 180
Number of Column Data per Record: 9
```

In [135]:

```
df.describe().round(2)
```

Out[135]:

	Age	Education	Usage	Fitness	Income	Miles
count	180.00	180.00	180.00	180.00	180.00	180.00
mean	28.79	15.57	3.46	3.31	53719.58	103.19
std	6.94	1.62	1.08	0.96	16506.68	51.86
min	18.00	12.00	2.00	1.00	29562.00	21.00
25%	24.00	14.00	3.00	3.00	44058.75	66.00
50%	26.00	16.00	3.00	3.00	50596.50	94.00
75%	33.00	16.00	4.00	4.00	58668.00	114.75
max	50.00	21.00	7.00	5.00	104581.00	360.00

In [136]:

```
df.describe(include="all").round(2)
```

Out[136]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
count	180	180.00	180	180.00	180	180.00	180.00	180.00	180.00
unique	3	NaN	2	NaN	2	NaN	NaN	NaN	NaN
top	KP281	NaN	Male	NaN	Partnered	NaN	NaN	NaN	NaN
freq	80	NaN	104	NaN	107	NaN	NaN	NaN	NaN
mean	NaN	28.79	NaN	15.57	NaN	3.46	3.31	53719.58	103.19
std	NaN	6.94	NaN	1.62	NaN	1.08	0.96	16506.68	51.86
min	NaN	18.00	NaN	12.00	NaN	2.00	1.00	29562.00	21.00
25%	NaN	24.00	NaN	14.00	NaN	3.00	3.00	44058.75	66.00
50%	NaN	26.00	NaN	16.00	NaN	3.00	3.00	50596.50	94.00
75%	NaN	33.00	NaN	16.00	NaN	4.00	4.00	58668.00	114.75
max	NaN	50.00	NaN	21.00	NaN	7.00	5.00	104581.00	360.00

In [137]:

```
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 [138]:

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

Out[138]:

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

OBSERVATION OF PROVIDED DATASET:

1. **This data contains information of customer**
   - **Product:** This column contains the model of the threadmill purchased by a customer. There are only three products in this dataset: ***KP281, KP481, KP781*** of which the highest purchased threadmill is KP281 with 80 quantity.
   - **Age:** This column contains the age of the customer who has bought a particular. The age range in this data set is ***between 18 to 50***
   - **Gender:** This column contains the gender. ***(Male / Female)***, where male are a total of 104 and remaining are female.
   - **Education:** This column contains data of the total number of years of education of a customer who has purchased a threadmill. The education ranges from a ***minimum of 12 years to maximum of 21 years***.
   - **Maritalstatus:** This column contains data if the customer is ***Single or Partnered***. Parterened are the highest number of purchasers with almost 107 customers and the remaining are all single.
   - **Usage:** This column contains the average number of times a customer plans to use the threadmill each week. The range of the times that the customers have chosen to use is from ***2 times to 7 times a week***.
   - **Income:** This column contains the data of the income each customer makes. The range of income starts from ***$ 29562.00*** to ***$ 104581.00***
   - **Miles:** This column contains data of the average number of miles a customer expects to either walk or run on the threadmill. The range of the average miles starts from ***21.00 Miles to 360.00 Miles***
2. **There is no missing data from each record which makes this data the most clean data.**
3. **The Data Types of each columns look right and wouldn't need any changes to be done.**
4. **There are a *total of 180 records* in this AeroFit data set.**

3. Non-Graphical Analysis:

In [22]:

```
total_sales_kp281 = len(df[df["Product"]=="KP281"])*1500 #It is given that one unit of KP281 is sold at $1500
total_sales_kp481 = len(df[df["Product"]=="KP481"])*1750 #It is given that one unit of KP481 is sold at $1750
total_sales_kp781 = len(df[df["Product"]=="KP781"])*2500 #It is given that one unit of KP781 is sold at $2500

Total_sales = total_sales_kp281 + total_sales_kp481 + total_sales_kp781
print("Total Amount of Sales at Aerofit:", Total_sales)
print("Total Amount of Sales at KP281:", total_sales_kp281)
print("Total Amount of Sales at KP481:", total_sales_kp481)
print("Total Amount of Sales at KP781:", total_sales_kp781)
```

```
Total Amount of Sales at Aerofit: 325000
Total Amount of Sales at KP281: 120000
Total Amount of Sales at KP481: 105000
Total Amount of Sales at KP781: 100000
```

Mean / Mode and Unique Data Analysis

In [139]:

```python
for i in df.columns:
  print("-" * 20)
  print(f"For {i} Column")
  print("-" * 20)
  if df[i].dtypes == 'int64':
    print (f'The MEAN value of {i}: {round(df[i].mean(),2)}')
  else:
    print (f'The MODE value of {i}: {df[i].mode()[0]}')
  print (f'The total number of unique values in {i}: {df[i].nunique()}')
  print (f'The total number of unique values in {i}: {df[i].unique()}\n\n')
```

```
--------------------
For Product Column
--------------------
The MODE value of Product: KP281
The total number of unique values in Product: 3
The total number of unique values in Product: ['KP281' 'KP481' 'KP781']


--------------------
For Age Column
--------------------
The MEAN value of Age: 28.79
The total number of unique values in Age: 32
The total number of unique values in Age: [18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
 43 44 46 47 50 45 48 42]


--------------------
For Gender Column
--------------------
The MODE value of Gender: Male
The total number of unique values in Gender: 2
The total number of unique values in Gender: ['Male' 'Female']


--------------------
For Education Column
--------------------
The MEAN value of Education: 15.57
The total number of unique values in Education: 8
The total number of unique values in Education: [14 15 12 13 16 18 20 21]


--------------------
For MaritalStatus Column
--------------------
The MODE value of MaritalStatus: Partnered
The total number of unique values in MaritalStatus: 2
The total number of unique values in MaritalStatus: ['Single' 'Partnered']


--------------------
For Usage Column
--------------------
The MEAN value of Usage: 3.46
The total number of unique values in Usage: 6
The total number of unique values in Usage: [3 2 4 5 6 7]


--------------------
For Fitness Column
--------------------
The MEAN value of Fitness: 3.31
The total number of unique values in Fitness: 5
The total number of unique values in Fitness: [4 3 2 1 5]


--------------------
For Income Column
--------------------
The MEAN value of Income: 53719.58
```

```
The MEAN value of Income: 53719.58
The total number of unique values in Income: 62
The total number of unique values in Income: [ 29562  31836  30699  32973  35247  37521  36384  38658  40932  34110
  39795  42069  44343  45480  46617  48891  53439  43206  52302  51165
  50028  54576  68220  55713  60261  67083  56850  59124  61398  57987
  64809  47754  65220  62535  48658  54781  48556  58516  53536  61006
  57271  52291  49801  62251  64741  70966  75946  74701  69721  83416
  88396  90886  92131  77191  52290  85906 103336  99601  89641  95866
 104581  95508]


--------------------
For Miles Column
--------------------
The MEAN value of Miles: 103.19
The total number of unique values in Miles: 37
The total number of unique values in Miles: [112  75  66  85  47 141 103  94 113  38 188  56 132 169  64  53 106  95
 212  42 127  74 170  21 120 200 140 100  80 160 180 240 150 300 280 260
 360]
```

OBSERVATION OF MEAN / MODE VALUES & UNIQUE VALUES:

- ***Mean values for numerical columns*** **have been found.**
- ***Mode values for categorical columns*** **have been found.**
- ***Total number of unique values*** **have been found.**
- ***List of Unique values*** **have been found.**

In [140]:

```
mean_median = df.describe().loc[["50%", "mean"]].rename(index = {"50%": "Median", "mean":"Mean"}).round(2).T
mean_median["Difference_Mean_Median"] = mean_median["Mean"] - mean_median["Median"]
mean_median
```

Out[140]:

	Median	Mean	Difference_Mean_Median
Age	26.0	28.79	2.79
Education	16.0	15.57	-0.43
Usage	3.0	3.46	0.46
Fitness	3.0	3.31	0.31
Income	50596.5	53719.58	3123.08
Miles	94.0	103.19	9.19

OBSERVATION OF THE DIFFERENCE BETWEEN MEAN AND MEDIAN

- **By looking at the difference that the data in the provided dataset is not necessarily evenly distributed.**
- **The column Education is skewed more towards the left of the median value.**
- **The columns Age, Usage, Fitness, Income and Miles are skewed towards the right of their respective median values.**

Value Counts

In [141]:

```
print("Total Number of Products Sold:", df["Product"].count())
print(f'Name of products and Total of each product sold:\n{df["Product"].value_counts()}')
```

```
Total Number of Products Sold: 180
Name of products and Total of each product sold:
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

Name: Product, dtype: int64

OBSERVATION OF THREADMILLS PURCHASES BY PRODUCT MODEL:

- **We can see the *highest number of product sold is of threadmill model KP281* .**
- **We can also see the *least number of product sold is of threadmill model KP781* .**

In [142]:

```
print(f'Total Number of purcahses by customer\non the basis of their age group:\n{df["Age"].value_counts(bins = [1,17,20,30,40,50])}')
```

```
Total Number of purcahses by customer
on the basis of their age group:
(20.0, 30.0]     110
(30.0, 40.0]      48
(40.0, 50.0]      12
(17.0, 20.0]      10
(0.999, 17.0]      0
Name: Age, dtype: int64
```

OBSERVATION OF THREADMILLS PURCHASES BY AGE GROUP:

- **We can see the *highest number of total purchase is done by group of people between the Ages of 20 & 30* .**
- **We can also see the *lowest number of total purchase is done by group of people between the Ages of 18 & 20* .**

In [143]:

```
print(f'Total Number of purchases by customer \non the basis of their Gender:\n{df["Gender"].value_counts()}')
```

```
Total Number of purchases by customer
on the basis of their Gender:
Male      104
Female     76
Name: Gender, dtype: int64
```

OBSERVATION OF THREADMILLS PURCHASES BY GENDER:

- **We can see the *highest number of total purchase is done by Gender: Male* .**
- **We can also see the *lowest number of total purchases is done by Gender: Female* .**

In [144]:

```
print(f'Total Number purchases by customer\non the basis of their Education Level:\n{df["Education"].value_counts()}')
```

```
Total Number purchases by customer
on the basis of their Education Level:
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

OBSERVATION THREADMILL PURCHASES BY EDUCATION LEVEL:

- **We can see the *highest number of purchases is done by customers who have 16 years of education* .**
- **We can also see the *lowest number of purchases is done by customers with 20 years of education* .**

In [145]:

```
print(f'Total Number purchases by customers\non the basis of their MaritalStatus:\n{df["MaritalStatus"].value_counts()}')
```

```
Total Number purchases by customers
on the basis of their MaritalStatus:
```

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

OBSERVATION OF THREADMILL PURCHASES BY MARITAL STATUS:

- **We can see the *highest number of purchases is done by customers who Partnered* .**
- **We can also see the *lowest number of purchases is done by customers who are single* .**

In [146]:

```
print(f'Total Number of purchases by customers\non the basis of average weekly usage:\n{df["Usage"].value_counts()}')
```

```
Total Number of purchases by customers
on the basis of average weekly usage:
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
```

OBSERVATION OF THREADMILL PURCHASES BY USAGE:

- **We can see the *highest number of purchases is done by customers who claim to use the threadmill atleast 3 times a week* .**
- **We can also see the *lowest number of purchases is done by customers who claim to use the threadmill atleast 7 times a week* .**

In [147]:

```
print(f'Total Number of purchases by customers\non the basis of their Fitness Level:\n{df["Fitness"].value_counts()}')
```

```
Total Number of purchases by customers
on the basis of their Fitness Level:
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

OBSERVATION OF THREADMILL PURCHASES BY FITNESS:

- **We can see the *highest number of purchases is done by customers who claim to have fitness level of 3* .**
- **We can also see the *lowest number of purchases is done by customers who claim to have fitness level of 1* .**

In [148]:

```
print(f'Total Number of purchases by customers\non the basis of their Income Groups:\n{df["Income"].value_counts(bins = [29000.00, 40000.00, 50000.00, 60000.00, 70000.00, 80000.00, 90000.0
0, 100000.00, 110000.00])}')
```

```
Total Number of purchases by customers
on the basis of their Income Groups:
(50000.0, 60000.0]      55
(40000.0, 50000.0]      51
(28999.999, 40000.0]    32
(60000.0, 70000.0]      19
(90000.0, 100000.0]      9
(80000.0, 90000.0]       7
(70000.0, 80000.0]       4
(100000.0, 110000.0]     3
Name: Income, dtype: int64
```

OBSERVATION OF THREADMILL PURCHASES BY INCOME:

- **We can see the *highest number of purchases is done by customers whose income falls under the bracket of 50,000 and 60,000 dollars* .**
- **We can also see that the *second highest purchases is done by customers whose income falls under the bracket of 40,000 and 50,000 dollars* and they are not much lower than the top range.**

- We can also see the ***lowest number of purchases is done by customers whose income falls under the bracket of 100,000 and 110,000 dollars*** .

In [149]:

```
print(f'Total Number of purchases by customers\non the basis of average weekly miles:\n{df["Miles"].value_counts(bins = [20, 50, 80, 110, 140, 170, 200, 230, 260, 290, 320, 350, 360])}')
```

```
Total Number of purchases by customers
on the basis of average weekly miles:
(80.0, 110.0]     66
(50.0, 80.0]      43
(110.0, 140.0]    20
(19.999, 50.0]    17
(140.0, 170.0]    15
(170.0, 200.0]    13
(230.0, 260.0]     2
(200.0, 230.0]     1
(260.0, 290.0]     1
(290.0, 320.0]     1
(350.0, 360.0]     1
(320.0, 350.0]     0
Name: Miles, dtype: int64
```

OBSERVATION OF THREADMILL PURCHASES BY MILES WALK / RUN:

- We can see the ***highest number of purchases is done by customers who claim to walk / run for 80.00 to 110.00 Miles*** .
- We can also see the ***lowest number of purchases is done by customers who claim to walk / run for 200.00 to 360.00 Miles*** .

Marginal Probability

In [150]:

```
for i in df.columns:
  if i not in ["Income", "Age","Miles"]:
    print("\n")
    print ("-" * 30)
    print (f'Probability of Product Purchases\nBy --> {i} <--')
    print ("-" * 30)
    for j in df[i].unique():
      lenght_of_product = len(df[df[i]==j])
      length_of_total_product_purcahses = len(df)
      print (f'Probability({i}={j}): {np.round(lenght_of_product / length_of_total_product_purcahses, 2)}')
```

```


------------------------------
Probability of Product Purchases
By --> Product <--
------------------------------
Probability(Product=KP281): 0.44
Probability(Product=KP481): 0.33
Probability(Product=KP781): 0.22


------------------------------
Probability of Product Purchases
By --> Gender <--
------------------------------
Probability(Gender=Male): 0.58
Probability(Gender=Female): 0.42


------------------------------
Probability of Product Purchases
By --> Education <--
------------------------------
Probability(Education=14): 0.31
Probability(Education=15): 0.03
Probability(Education=12): 0.02
Probability(Education=13): 0.03
Probability(Education=16): 0.47
```

```
Probability(Education=18): 0.13
Probability(Education=20): 0.01
Probability(Education=21): 0.02


------------------------------
Probability of Product Purchases
By --> MaritalStatus <--
------------------------------
Probability(MaritalStatus=Single): 0.41
Probability(MaritalStatus=Partnered): 0.59


------------------------------
Probability of Product Purchases
By --> Usage <--
------------------------------
Probability(Usage=3): 0.38
Probability(Usage=2): 0.18
Probability(Usage=4): 0.29
Probability(Usage=5): 0.09
Probability(Usage=6): 0.04
Probability(Usage=7): 0.01


------------------------------
Probability of Product Purchases
By --> Fitness <--
------------------------------
Probability(Fitness=4): 0.13
Probability(Fitness=3): 0.54
Probability(Fitness=2): 0.14
Probability(Fitness=1): 0.01
Probability(Fitness=5): 0.17
```

In [151]:

```
print ("-" * 30)
print (f'Probability of Product Purchases\nBy --> Age Groups <--')
print ("-" * 30)
print (f'Probability(AgeGroup=(18 to 20): {np.round(len(df[df["Age"].between(18.00, 20.00)]) / len(df), 2)}')
print (f'Probability(AgeGroup=(21 to 30): {np.round(len(df[df["Age"].between(21.00, 30.00)]) / len(df), 2)}')
print (f'Probability(AgeGroup=(31 to 40): {np.round(len(df[df["Age"].between(31.00, 40.00)]) / len(df), 2)}')
print (f'Probability(AgeGroup=(41 to 50): {np.round(len(df[df["Age"].between(41.00, 50.00)]) / len(df), 2)}')
```

```
------------------------------
Probability of Product Purchases
By --> Age Groups <--
------------------------------
Probability(AgeGroup=(18 to 20): 0.06
Probability(AgeGroup=(21 to 30): 0.61
Probability(AgeGroup=(31 to 40): 0.27
Probability(AgeGroup=(41 to 50): 0.07
```

In [152]:

```
print ("-" * 30)
print (f'Probability of Product Purchases\nBy --> Income Groups <--')
print ("-" * 30)
print (f'Probability(Income between $29000 to $40000): {np.round(len(df[df["Income"].between(29000, 40000)]) / len(df), 2)}')
print (f'Probability(Income between $41000 to $50000): {np.round(len(df[df["Income"].between(41000, 50000)]) / len(df), 2)}')
print (f'Probability(Income between $51000 to $60000): {np.round(len(df[df["Income"].between(51000, 60000)]) / len(df), 2)}')
print (f'Probability(Income between $61000 to $70000): {np.round(len(df[df["Income"].between(61000, 70000)]) / len(df), 2)}')
print (f'Probability(Income between $71000 to $80000): {np.round(len(df[df["Income"].between(71000, 80000)]) / len(df), 2)}')
print (f'Probability(Income between $81000 to $90000): {np.round(len(df[df["Income"].between(81000, 90000)]) / len(df), 2)}')
print (f'Probability(Income between $91000 to $100000): {np.round(len(df[df["Income"].between(91000, 100000)]) / len(df), 2)}')
print (f'Probability(Income between $101000 to $110000): {np.round(len(df[df["Income"].between(101000, 110000)]) / len(df), 2)}')
```

```
------------------------------
Probability of Product Purchases
By --> Income Groups <--
------------------------------
```

```
Probability(Income between $29000 to $40000): 0.18
Probability(Income between $41000 to $50000): 0.25
Probability(Income between $51000 to $60000): 0.27
Probability(Income between $61000 to $70000): 0.09
Probability(Income between $71000 to $80000): 0.02
Probability(Income between $81000 to $90000): 0.04
Probability(Income between $91000 to $100000): 0.03
Probability(Income between $101000 to $110000): 0.02
```

In [153]:

```
print ("-" * 30)
print (f'Probability of Product Purchases\nBy --> Weekly Miles Groups <--')
print ("-" * 30)
print (f'Probability(Miles between 21 to 50): {np.round(len(df[df["Miles"].between(21, 50)]) / len(df), 2)}')
print (f'Probability(Miles between 51 to 80): {np.round(len(df[df["Miles"].between(51, 80)]) / len(df), 2)}')
print (f'Probability(Miles between 81 to 110): {np.round(len(df[df["Miles"].between(81, 110)]) / len(df), 2)}')
print (f'Probability(Miles between 111 to 140): {np.round(len(df[df["Miles"].between(111, 140)]) / len(df), 2)}')
print (f'Probability(Miles between 141 to 170): {np.round(len(df[df["Miles"].between(141, 170)]) / len(df), 2)}')
print (f'Probability(Miles between 171 to 200): {np.round(len(df[df["Miles"].between(171, 200)]) / len(df), 2)}')
print (f'Probability(Miles between 201 to 230): {np.round(len(df[df["Miles"].between(201, 230)]) / len(df), 2)}')
print (f'Probability(Miles between 231 to 260): {np.round(len(df[df["Miles"].between(231, 260)]) / len(df), 2)}')
print (f'Probability(Miles between 261 to 290): {np.round(len(df[df["Miles"].between(261, 290)]) / len(df), 2)}')
print (f'Probability(Miles between 291 to 320): {np.round(len(df[df["Miles"].between(291, 320)]) / len(df), 2)}')
print (f'Probability(Miles between 321 to 360): {np.round(len(df[df["Miles"].between(321, 360)]) / len(df), 2)}')
```

```
------------------------------
Probability of Product Purchases
By --> Weekly Miles Groups <--
------------------------------
Probability(Miles between 21 to 50): 0.09
Probability(Miles between 51 to 80): 0.24
Probability(Miles between 81 to 110): 0.37
Probability(Miles between 111 to 140): 0.11
Probability(Miles between 141 to 170): 0.08
Probability(Miles between 171 to 200): 0.07
Probability(Miles between 201 to 230): 0.01
Probability(Miles between 231 to 260): 0.01
Probability(Miles between 261 to 290): 0.01
Probability(Miles between 291 to 320): 0.01
Probability(Miles between 321 to 360): 0.01
```

Conditional Probabilty

In [154]:

```
prob_table = pd.DataFrame()
#To get the list of all columns
for i in df.columns:
  #To get the unique list of products purcahsed
  if i not in ["Product","Age","Income","Miles"]:
    print (f'\n\n--> Given {i} <--')
    for j in df["Product"].unique():
        print ("-" * 45)
        print (f'Probability of a Customer Buying Product {j}')
        print ("-" * 45)
        for k in df[i].unique():
          probabilty_of_product_intersect_column = len (df [ (df["Product"] == j) & (df[i] == k) ])
          probability_of_column_value = len (df[df[i] == k])
          prob_table.loc[j,k] = np.round(probabilty_of_product_intersect_column / probability_of_column_value, 2) * 100
          print (f'Probability(Product={j} | {i}={k}): {np.round(probabilty_of_product_intersect_column / probability_of_column_value, 2)}')
```

```
--> Given Gender <--
---------------------------------------------
Probability of a Customer Buying Product KP281
---------------------------------------------
Probability(Product=KP281 | Gender=Male): 0.38
Probability(Product=KP281 | Gender=Female): 0.53
---------------------------------------------
Probability of a Customer Buying Product KP481
```

```
Probability of a Customer Buying Product KP481
-----------------------------------------------
Probability(Product=KP481 | Gender=Male): 0.3
Probability(Product=KP481 | Gender=Female): 0.38
-----------------------------------------------
Probability of a Customer Buying Product KP781
-----------------------------------------------
Probability(Product=KP781 | Gender=Male): 0.32
Probability(Product=KP781 | Gender=Female): 0.09


--> Given Education <--
-----------------------------------------------
Probability of a Customer Buying Product KP281
-----------------------------------------------
Probability(Product=KP281 | Education=14): 0.55
Probability(Product=KP281 | Education=15): 0.8
Probability(Product=KP281 | Education=12): 0.67
Probability(Product=KP281 | Education=13): 0.6
Probability(Product=KP281 | Education=16): 0.46
Probability(Product=KP281 | Education=18): 0.09
Probability(Product=KP281 | Education=20): 0.0
Probability(Product=KP281 | Education=21): 0.0
-----------------------------------------------
Probability of a Customer Buying Product KP481
-----------------------------------------------
Probability(Product=KP481 | Education=14): 0.42
Probability(Product=KP481 | Education=15): 0.2
Probability(Product=KP481 | Education=12): 0.33
Probability(Product=KP481 | Education=13): 0.4
Probability(Product=KP481 | Education=16): 0.36
Probability(Product=KP481 | Education=18): 0.09
Probability(Product=KP481 | Education=20): 0.0
Probability(Product=KP481 | Education=21): 0.0
-----------------------------------------------
Probability of a Customer Buying Product KP781
-----------------------------------------------
Probability(Product=KP781 | Education=14): 0.04
Probability(Product=KP781 | Education=15): 0.0
Probability(Product=KP781 | Education=12): 0.0
Probability(Product=KP781 | Education=13): 0.0
Probability(Product=KP781 | Education=16): 0.18
Probability(Product=KP781 | Education=18): 0.83
Probability(Product=KP781 | Education=20): 1.0
Probability(Product=KP781 | Education=21): 1.0


--> Given MaritalStatus <--
-----------------------------------------------
Probability of a Customer Buying Product KP281
-----------------------------------------------
Probability(Product=KP281 | MaritalStatus=Single): 0.44
Probability(Product=KP281 | MaritalStatus=Partnered): 0.45
-----------------------------------------------
Probability of a Customer Buying Product KP481
-----------------------------------------------
Probability(Product=KP481 | MaritalStatus=Single): 0.33
Probability(Product=KP481 | MaritalStatus=Partnered): 0.34
-----------------------------------------------
Probability of a Customer Buying Product KP781
-----------------------------------------------
Probability(Product=KP781 | MaritalStatus=Single): 0.23
Probability(Product=KP781 | MaritalStatus=Partnered): 0.21


--> Given Usage <--
-----------------------------------------------
Probability of a Customer Buying Product KP281
-----------------------------------------------
Probability(Product=KP281 | Usage=3): 0.54
Probability(Product=KP281 | Usage=2): 0.58
Probability(Product=KP281 | Usage=4): 0.42
```

```
Probability(Product=KP281 | Usage=5): 0.12
Probability(Product=KP281 | Usage=6): 0.0
Probability(Product=KP281 | Usage=7): 0.0
--------------------------------------------
Probability of a Customer Buying Product KP481
--------------------------------------------
Probability(Product=KP481 | Usage=3): 0.45
Probability(Product=KP481 | Usage=2): 0.42
Probability(Product=KP481 | Usage=4): 0.23
Probability(Product=KP481 | Usage=5): 0.18
Probability(Product=KP481 | Usage=6): 0.0
Probability(Product=KP481 | Usage=7): 0.0
--------------------------------------------
Probability of a Customer Buying Product KP781
--------------------------------------------
Probability(Product=KP781 | Usage=3): 0.01
Probability(Product=KP781 | Usage=2): 0.0
Probability(Product=KP781 | Usage=4): 0.35
Probability(Product=KP781 | Usage=5): 0.71
Probability(Product=KP781 | Usage=6): 1.0
Probability(Product=KP781 | Usage=7): 1.0


--> Given Fitness <--
--------------------------------------------
Probability of a Customer Buying Product KP281
--------------------------------------------
Probability(Product=KP281 | Fitness=4): 0.38
Probability(Product=KP281 | Fitness=3): 0.56
Probability(Product=KP281 | Fitness=2): 0.54
Probability(Product=KP281 | Fitness=1): 0.5
Probability(Product=KP281 | Fitness=5): 0.06
--------------------------------------------
Probability of a Customer Buying Product KP481
--------------------------------------------
Probability(Product=KP481 | Fitness=4): 0.33
Probability(Product=KP481 | Fitness=3): 0.4
Probability(Product=KP481 | Fitness=2): 0.46
Probability(Product=KP481 | Fitness=1): 0.5
Probability(Product=KP481 | Fitness=5): 0.0
--------------------------------------------
Probability of a Customer Buying Product KP781
--------------------------------------------
Probability(Product=KP781 | Fitness=4): 0.29
Probability(Product=KP781 | Fitness=3): 0.04
Probability(Product=KP781 | Fitness=2): 0.0
Probability(Product=KP781 | Fitness=1): 0.0
Probability(Product=KP781 | Fitness=5): 0.94
```

In [155]:

```
print (f'--> Given Age Groups<--')
for i in df["Product"].unique():
  print ("-" * 45)
  print (f'Probability of a Customer Buying Product {i}')
  print ("-" * 45)
  print (f'Probability(Product={i} | AgeGroup=(18 to 30)): {np.round(len(df[(df["Product"] == i) & (df["Age"].between(18.00, 30.00))]) / len(df[df["Age"].between(18.00, 30.00)]), 2)}')
  print (f'Probability(Product={i} | AgeGroup=(31 to 40)): {np.round(len(df[(df["Product"] == i) & (df["Age"].between(31.00, 40.00))]) / len(df[df["Age"].between(31.00, 40.00)]), 2)}')
  print (f'Probability(Product={i} | AgeGroup=(41 to 50)): {np.round(len(df[(df["Product"] == i) & (df["Age"].between(41.00, 50.00))]) / len(df[df["Age"].between(41.00, 50.00)]), 2)}')
```

```
--> Given Age Groups<--
--------------------------------------------
Probability of a Customer Buying Product KP281
--------------------------------------------
Probability(Product=KP281 | AgeGroup=(18 to 30)): 0.46
Probability(Product=KP281 | AgeGroup=(31 to 40)): 0.4
Probability(Product=KP281 | AgeGroup=(41 to 50)): 0.5
--------------------------------------------
Probability of a Customer Buying Product KP481
--------------------------------------------
Probability(Product=KP481 | AgeGroup=(18 to 30)): 0.29
Probability(Product=KP481 | AgeGroup=(31 to 40)): 0.48
Probability(Product=KP481 | AgeGroup=(41 to 50)): 0.17
```

```
Probability(Product=KP481 | AgeGroup=(41 to 50)): 0.17
-------------------------------------------
Probability of a Customer Buying Product KP781
-------------------------------------------
Probability(Product=KP781 | AgeGroup=(18 to 30)): 0.25
Probability(Product=KP781 | AgeGroup=(31 to 40)): 0.12
Probability(Product=KP781 | AgeGroup=(41 to 50)): 0.33
```

In [156]:

```
print (f'--> Given Income Groups<--')
for i in df["Product"].unique():
  print ("-" * 45)
  print (f'Probability of a Customer Buying Product {i}')
  print ("-" * 45)
  print (f'Probability(Product={i} | Income=($29000 to $50000)): {np.round(len(df[(df["Product"] == i) & (df["Income"].between(29000, 50000))]) / len(df[df["Income"].between(29000, 50000)]), 2)}')
  print (f'Probability(Product={i} | Income=($51000 to $70000)): {np.round(len(df[(df["Product"] == i) & (df["Income"].between(51000, 70000))]) / len(df[df["Income"].between(51000, 70000)]), 2)}')
  print (f'Probability(Product={i} | Income=($71000 to $90000)): {np.round(len(df[(df["Product"] == i) & (df["Income"].between(71000, 90000))]) / len(df[df["Income"].between(71000, 90000)]), 2)}')
  print (f'Probability(Product={i} | Income=($91000 to $110000)): {np.round(len(df[(df["Product"] == i) & (df["Income"].between(91000, 110000))]) / len(df[df["Income"].between(91000, 110000)]), 2)}')
```

```
--> Given Income Groups<--
-------------------------------------------
Probability of a Customer Buying Product KP281
-------------------------------------------
Probability(Product=KP281 | Income=($29000 to $50000)): 0.58
Probability(Product=KP281 | Income=($51000 to $70000)): 0.45
Probability(Product=KP281 | Income=($71000 to $90000)): 0.0
Probability(Product=KP281 | Income=($91000 to $110000)): 0.0
-------------------------------------------
Probability of a Customer Buying Product KP481
-------------------------------------------
Probability(Product=KP481 | Income=($29000 to $50000)): 0.36
Probability(Product=KP481 | Income=($51000 to $70000)): 0.37
Probability(Product=KP481 | Income=($71000 to $90000)): 0.0
Probability(Product=KP481 | Income=($91000 to $110000)): 0.0
-------------------------------------------
Probability of a Customer Buying Product KP781
-------------------------------------------
Probability(Product=KP781 | Income=($29000 to $50000)): 0.06
Probability(Product=KP781 | Income=($51000 to $70000)): 0.18
Probability(Product=KP781 | Income=($71000 to $90000)): 1.0
Probability(Product=KP781 | Income=($91000 to $110000)): 1.0
```

In [157]:

```
print (f'--> Given Income Groups<--')
for i in df["Product"].unique():
  print ("-" * 45)
  print (f'Probability of a Customer Buying Product {i}')
  print ("-" * 45)
  print (f'Probability(Product={i} | Miles between 21 to 50): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(21, 50))]) / len(df[df["Miles"].between(21, 50)]), 2)}')
  print (f'Probability(Product={i} | Miles between 51 to 80): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(51, 80))]) / len(df[df["Miles"].between(51, 80)]), 2)}')
  print (f'Probability(Product={i} | Miles between 81 to 110): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(81, 110))]) / len(df[df["Miles"].between(81, 110)]), 2)}')
  print (f'Probability(Product={i} | Miles between 111 to 140): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(111, 140))]) / len(df[df["Miles"].between(111, 140)]), 2)}')
  print (f'Probability(Product={i} | Miles between 141 to 170): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(141, 170))]) / len(df[df["Miles"].between(141, 170)]), 2)}')
  print (f'Probability(Product={i} | Miles between 171 to 200): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(171, 200))]) / len(df[df["Miles"].between(171, 200)]), 2)}')
  print (f'Probability(Product={i} | Miles between 201 to 230): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(201, 230))]) / len(df[df["Miles"].between(201, 230)]), 2)}')
  print (f'Probability(Product={i} | Miles between 231 to 260): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(231, 260))]) / len(df[df["Miles"].between(231, 260)]), 2)}')
  print (f'Probability(Product={i} | Miles between 261 to 290): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(261, 290))]) / len(df[df["Miles"].between(261, 290)]), 2)}')
  print (f'Probability(Product={i} | Miles between 291 to 320): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(291, 320))]) / len(df[df["Miles"].between(291, 320)]), 2)}')
  print (f'Probability(Product={i} | Miles between 321 to 360): {np.round(len(df[(df["Product"] == i) & (df["Miles"].between(321, 360))]) / len(df[df["Miles"].between(321, 360)]), 2)}')
```

```
--> Given Income Groups<--
-------------------------------------------
Probability of a Customer Buying Product KP281
-------------------------------------------
Probability(Product=KP281 | Miles between 21 to 50): 0.71
Probability(Product=KP281 | Miles between 51 to 80): 0.6
Probability(Product=KP281 | Miles between 81 to 110): 0.41
```

```
Probability(Product=KP281 | Miles between 81 to 110): 0.41
Probability(Product=KP281 | Miles between 111 to 140): 0.55
Probability(Product=KP281 | Miles between 141 to 170): 0.2
Probability(Product=KP281 | Miles between 171 to 200): 0.08
Probability(Product=KP281 | Miles between 201 to 230): 0.0
Probability(Product=KP281 | Miles between 231 to 260): 0.0
Probability(Product=KP281 | Miles between 261 to 290): 0.0
Probability(Product=KP281 | Miles between 291 to 320): 0.0
Probability(Product=KP281 | Miles between 321 to 360): 0.0
--------------------------------------------
Probability of a Customer Buying Product KP481
--------------------------------------------
Probability(Product=KP481 | Miles between 21 to 50): 0.29
Probability(Product=KP481 | Miles between 51 to 80): 0.37
Probability(Product=KP481 | Miles between 81 to 110): 0.47
Probability(Product=KP481 | Miles between 111 to 140): 0.25
Probability(Product=KP481 | Miles between 141 to 170): 0.13
Probability(Product=KP481 | Miles between 171 to 200): 0.0
Probability(Product=KP481 | Miles between 201 to 230): 1.0
Probability(Product=KP481 | Miles between 231 to 260): 0.0
Probability(Product=KP481 | Miles between 261 to 290): 0.0
Probability(Product=KP481 | Miles between 291 to 320): 0.0
Probability(Product=KP481 | Miles between 321 to 360): 0.0
--------------------------------------------
Probability of a Customer Buying Product KP781
--------------------------------------------
Probability(Product=KP781 | Miles between 21 to 50): 0.0
Probability(Product=KP781 | Miles between 51 to 80): 0.02
Probability(Product=KP781 | Miles between 81 to 110): 0.12
Probability(Product=KP781 | Miles between 111 to 140): 0.2
Probability(Product=KP781 | Miles between 141 to 170): 0.67
Probability(Product=KP781 | Miles between 171 to 200): 0.92
Probability(Product=KP781 | Miles between 201 to 230): 0.0
Probability(Product=KP781 | Miles between 231 to 260): 1.0
Probability(Product=KP781 | Miles between 261 to 290): 1.0
Probability(Product=KP781 | Miles between 291 to 320): 1.0
Probability(Product=KP781 | Miles between 321 to 360): 1.0
```

Customer Profiling KP281:

1. **44% customers have purchased this threadmill.**
2. **53% female and 38% male have purchased this threadmill.**
3. **80% customers with education of 15 years have bought this threadmill.**
4. **45% of partnered and 44% of single customers have bought this threadmill.**
5. **Upto 58% of customers who claimed to use the threadmill for an average of upto 3 times a week have bought the threadmill.**
6. **Upto 56% of customers who claimed to have fitness upto 3 have bought this threadmill.**
7. **50% of customers who are aged between 41 to 50 have made a purchase of this threadmill.**
8. **58% customers who make an income between 29,000 to 50,000 dollars have made a purchase of this threadmill.**

Customer Profiling KP481:

1. **33% customers have purchased this threadmill.**
2. **38% females and 30% male have bought this threadmill**
3. **42% customers with education of 14 years have bought this threadmill.**
4. **34% Partnered and 30% Single customers have bought this threadmill.**
5. **42% to 45% customers claim to have use the threadmill average upto 3 times a week.**
6. **46% to 50% claim to be of fitness level upto 2.**
7. **48% of customers between the age of 31 to 40 have bought this threadmill.**
8. **36% to 37% customers with salary between 29,000 to 70,000 dollars have bought this threadmill.**

Customer Profiling KP781:

1. **22% customers have bought this threadmill.**
2. **32% male and only 9% of female bought this threadmill.**
3. **100% customers with education levels of 20 years and 21 years have bought this threadmill.**
4. **23% single and 21% partnered customers have bought this threadmill**

5. 71% of customers who claim to use the threadmill average of 5 times a week have bought this threadmill.
6. 94% of customer who claim to have fitness level of 5 have bought this product.
7. 33% customers who fall between the ages of 41 to 50 have bought this threadmill.
8. 100% customers whose income between 71,000 to 110,000 dollars have bought this threadmill.

Probability using Joints (Contingency Table)

In [158]:

```
df.groupby("Product")["Age"].value_counts(bins = [18,30,40,50], normalize = True).unstack().round(2)
```

Out[158]:

Age	(17.999, 30.0]	(30.0, 40.0]	(40.0, 50.0]
Product			
KP281	0.69	0.24	0.08
KP481	0.58	0.38	0.03
KP781	0.75	0.15	0.10

OBSERVATION CONTINGENCY OF PRODUCT & AGE

> 69% of customers between the ages of 18 and 30 bought KP281.
>
> 58% of customers between the ages of 18 and 30 bought KP481.
>
> 75% of customers between the ages of 18 and 30 bought KP781.

We can see that customers between the ages of 18 to 30 tend to buy the threadmills more than other age groups.

In [159]:

```
pd.crosstab(index = df["Product"], columns = [df["Gender"]], margins = True).round(2)
```

Out[159]:

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

In [160]:

```
pd.crosstab(index = df["Product"], columns = [df["Gender"]], normalize = "index", margins = True).round(2)
```

Out[160]:

Gender	Female	Male
Product		
KP281	0.50	0.50
KP481	0.48	0.52
KP781	0.18	0.82
All	0.42	0.58

OBSERVATION CONTINGENCY OF PRODUCT & Gender

> **50% male and 50% female customers bought KP281.**
>
> **52% female and 48% male customers bought KP481.**
>
> **82% male customers bought KP781.**

We can see both male and female almost equally bought both threadmills KP281 and KP481. But, the threadmill KP781 was bought most by male customers.

In [161]:

```
pd.crosstab(index = df["Product"], columns = [df["MaritalStatus"]], margins = True)
```

Out[161]:

MaritalStatus	Partnered	Single	All
Product			
KP281	48	32	80
KP481	36	24	60
KP781	23	17	40
All	107	73	180

In [162]:

```
pd.crosstab(index = df["Product"], columns = [df["MaritalStatus"]], normalize = "index", margins = True).round(2)
```

Out[162]:

MaritalStatus	Partnered	Single
Product		
KP281	0.60	0.40
KP481	0.60	0.40
KP781	0.57	0.42
All	0.59	0.41

OBSERVATION CONTINGENCY OF PRODUCT & Marital Status

> **60% partnered customers bought KP281.**
>
> **60% partnered customers bought KP481.**
>
> **57% partnered customers bought KP781.**

We can see that most customers who bought all three threadmills were majorly partnered.

In [163]:

```
pd.crosstab(index = df["Product"], columns = [df["Education"]], margins = True)
```

Out[163]:

Education	12	13	14	15	16	18	20	21	All
Product									
KP281	2	3	30	4	39	2	0	0	80
KP481	1	2	23	1	31	2	0	0	60
KP781	0	0	2	0	15	19	1	3	40
All	3	5	55	5	85	23	1	3	180

In [164]:

```
pd.crosstab(index = df["Product"], columns = [df["Education"]], normalize= "index", margins = True).round(2)
```

Out[164]:

Education	12	13	14	15	16	18	20	21
Product								
KP281	0.02	0.04	0.38	0.05	0.49	0.02	0.00	0.00
KP481	0.02	0.03	0.38	0.02	0.52	0.03	0.00	0.00
KP781	0.00	0.00	0.05	0.00	0.38	0.48	0.02	0.08
All	0.02	0.03	0.31	0.03	0.47	0.13	0.01	0.02

OBSERVATION CONTINGENCY OF PRODUCT & EDUCATION

> **49% customers with education of 16 years bought KP281.**
>
> **52% customers with education of 16 years bought KP481.**
>
> **48% customers with education of 18 years bought KP781.**

We can see that the first two threadmills was mostly bought by people who have educated for around 16 years, where as the threadmill KP781 was bought by someone who us more educated.

In [165]:

```
pd.crosstab(index = df["Product"], columns= [df["Usage"]], margins = True)
```

Out[165]:

Usage	2	3	4	5	6	7	All
Product							
KP281	19	37	22	2	0	0	80
KP481	14	31	12	3	0	0	60
KP781	0	1	18	12	7	2	40
All	33	69	52	17	7	2	180

In [166]:

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

Out[166]:

Usage	2	3	4	5	6	7
Product						
KP281	0.24	0.46	0.28	0.02	0.00	0.00
KP481	0.23	0.52	0.20	0.05	0.00	0.00
KP781	0.00	0.02	0.45	0.30	0.18	0.05
All	0.18	0.38	0.29	0.09	0.04	0.01

OBSERVATION CONTINGENCY OF PRODUCT & USAGE

> **46% customers who claim to use the threadmill an average of 3 times a week have bought KP281.**
>
> **52% customers who claim to use the threadmill an average of 3 times a week have bought KP481.**
>
> **45% customers who claim to use the threadmill an average of 4 times a week have bought KP781.**

We can see that most of the customers who bought all the threadmills tend to claim that they will use the threadmill an average of upto 4 times a week.

In [167]:

```
pd.crosstab(index = df["Product"], columns = [df["Fitness"]], margins=True)
```

Out[167]:

Fitness	1	2	3	4	5	All
Product						
KP281	1	14	54	9	2	80
KP481	1	12	39	8	0	60
KP781	0	0	4	7	29	40
All	2	26	97	24	31	180

In [168]:

```
pd.crosstab(index = df["Product"], columns = [df["Fitness"]], normalize = "index", margins = True).round(2)
```

Out[168]:

Fitness	1	2	3	4	5
Product					
KP281	0.01	0.18	0.68	0.11	0.02
KP481	0.02	0.20	0.65	0.13	0.00
KP781	0.00	0.00	0.10	0.18	0.72
All	0.01	0.14	0.54	0.13	0.17

OBSERVATION CONTINGENCY OF PRODUCT & FITNESS

> **68% customers who claim to have fitness level of 3 have bought KP281.**
>
> **65% customers who claim to have fitness level of 3 have bought KP481.**
>
> **72% customers who claim to have fitness level of 5 have bought KP781.**

Customers who claim to have fitness level of 3 have bought KP281 & KP481. Customers who have claimed to have higher fitness have bought KP781.

In [169]:

```
df.groupby("Product")["Income"].value_counts(bins=[29000,50000,70000,90000,110000], normalize = True).unstack().round(2)
```

Out[169]:

Income	(28999.999, 50000.0]	(50000.0, 70000.0]	(70000.0, 90000.0]	(90000.0, 110000.0]
Product				
KP281	0.60	0.4	0.00	0.0
KP481	0.50	0.5	0.00	0.0
KP781	0.12	0.3	0.28	0.3

In [170]:

```
pd.crosstab(index = [df["Product"],df["Gender"]], columns = [df["MaritalStatus"]], normalize = "index", margins = True).round(2)
```

Out[170]:

	MaritalStatus	Partnered	Single

Product	MaritalStatus Gender	Partnered	Single
Product	Gender		
KP281	Female	0.68	0.32
	Male	0.52	0.48
KP481	Female	0.52	0.48
	Male	0.68	0.32
KP781	Female	0.57	0.43
	Male	0.58	0.42
All		0.59	0.41

OBSERVATION OF PRODUCT AND GENDER WITH MARITAL STATUS

> **68% partnered female have mostly bought KP281.**
>
> **68% partnered male have mostly bought KP481.**
>
> **Upto 58% partnered male and female have mostly bought KP781.**

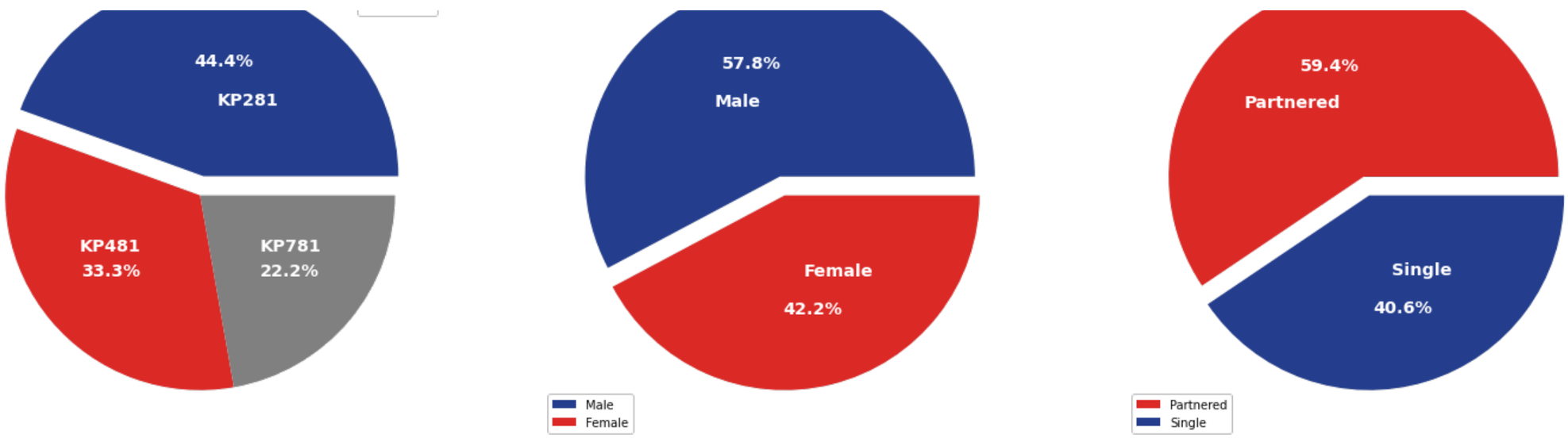
Most customers who have purcahsed the threadmills are partnered.

4. Graphical Analysis:

PIE CHARTS

In [358]:

```
plt.figure(figsize = (25,8.5))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(1,3,1)
plt.pie(df["Product"].value_counts().values, labels = df["Product"].value_counts().index, labeldistance = 0.4,
        explode = (0.1,0,0), autopct="%.1f%%", colors = [aero_blue,aero_red,aero_grey],
        textprops ={"color":"white", "fontweight":"bold","fontsize":"x-large"})
plt.legend()
plt.title("Aerofit Threadmills Contribution", fontsize = 14, fontweight = "bold")

plt.subplot(1,3,2)
plt.pie(df["Gender"].value_counts().values, labels = df["Gender"].value_counts().index, labeldistance = 0.4,
        explode = (0.1,0), autopct="%.1f%%", colors = [aero_blue,aero_red],
        textprops ={"color":"white", "fontweight":"bold","fontsize":"x-large"})
plt.legend()
plt.title("Gender Contribution", fontsize = 14, fontweight = "bold")

plt.subplot(1,3,3)
plt.pie(df["MaritalStatus"].value_counts().values, labels = df["MaritalStatus"].value_counts().index, labeldistance = 0.4,
        explode = (0.1,0), autopct="%.1f%%", colors = [aero_red, aero_blue],
        textprops ={"color":"white", "fontweight":"bold","fontsize":"x-large"})
plt.legend()
plt.title("Marital Status Contribution", fontsize = 14, fontweight = "bold")

plt.suptitle("PERCENTAGES", size = 18, fontweight = "bold")
plt.show()
```



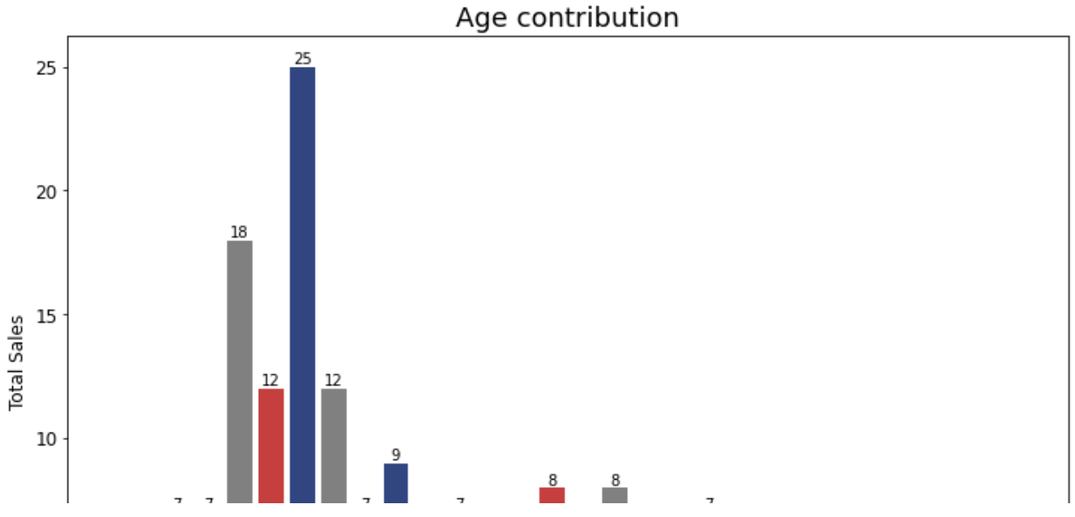
Observations:

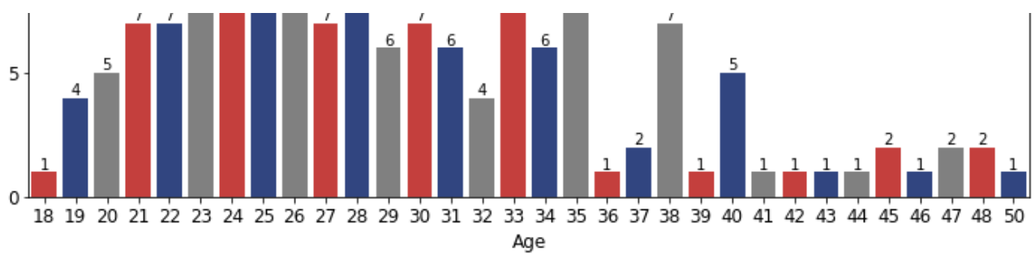
- **Aerofit Threadmils Contribution: This pie diagram shows us that product KP281 has almost 44.4% purchases of the total purchases made.**
- **Gender Contribution: This shows that of all the customers male are around 57.8%.**
- **Marital Contribution: This shows that of all the customers the most of them are partnered at 59.4%.**

Countplots (Univariate & Bivariate)

In [359]:

```
plt.figure(figsize = (12,8))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"
colors = [aero_red, aero_blue, aero_grey]
ax = sns.countplot(data = df, x = "Age", palette = colors)
for container in ax.containers:
    ax.bar_label(container)
plt.ylabel('Total Sales', fontsize=12)
plt.xlabel("Age", fontsize=12)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.title('Age contribution', fontsize=18)

plt.show()
```



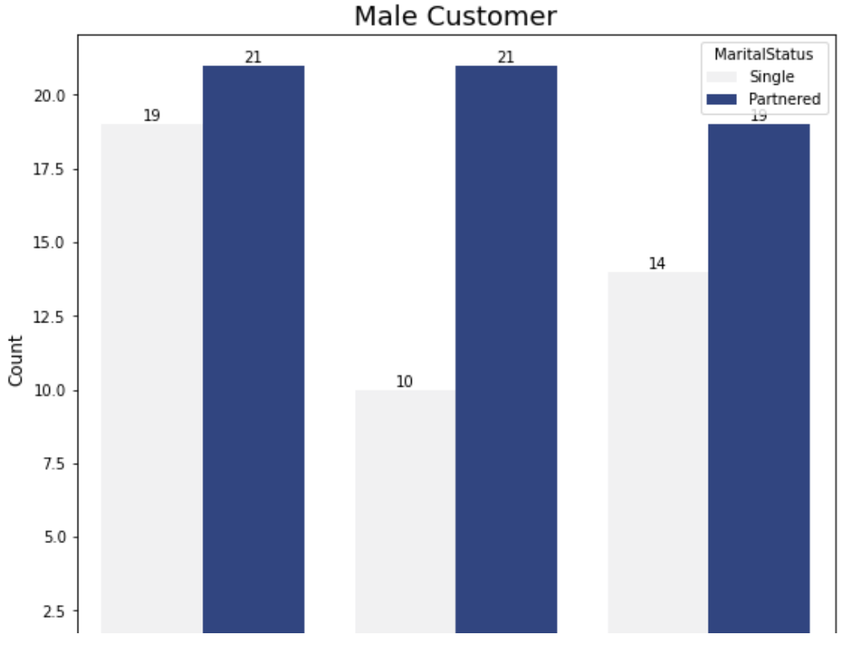


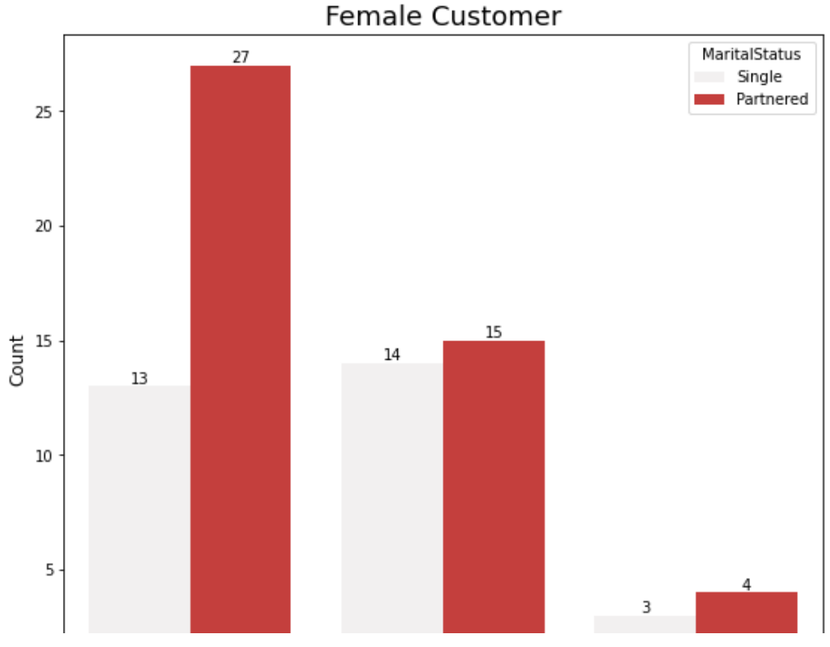
Observations:

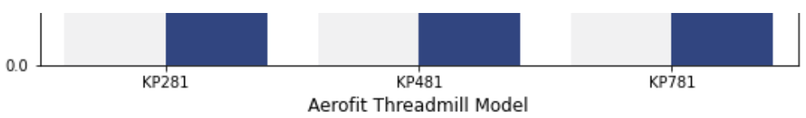
> **Out of total of 180 purchase the highest count of purchases was done by customers of Age 25.**

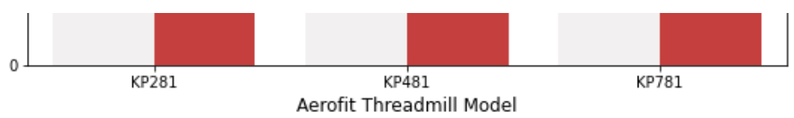
In [361]:

```
plt.figure(figsize = (20,8))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(1,2,1)
ax = sns.countplot(data = df[df["Gender"] == "Male"], x = "Product", hue = "MaritalStatus", color = aero_blue)
for container in ax.containers:
    ax.bar_label(container)
plt.ylabel('Count', fontsize=12)
plt.xlabel("Aerofit Threadmill Model", fontsize=12)
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)
plt.title('Male Customer', fontsize=18)

plt.subplot(1,2,2)
ax = sns.countplot(data = df[df["Gender"] == "Female"], x = "Product", hue = "MaritalStatus", hue_order = ['Single', 'Partnered'], color = aero_red)
for container in ax.containers:
    ax.bar_label(container)
plt.ylabel('Count', fontsize=12)
plt.xlabel("Aerofit Threadmill Model", fontsize=12)
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)
plt.title('Female Customer', fontsize=18)

plt.show()
```







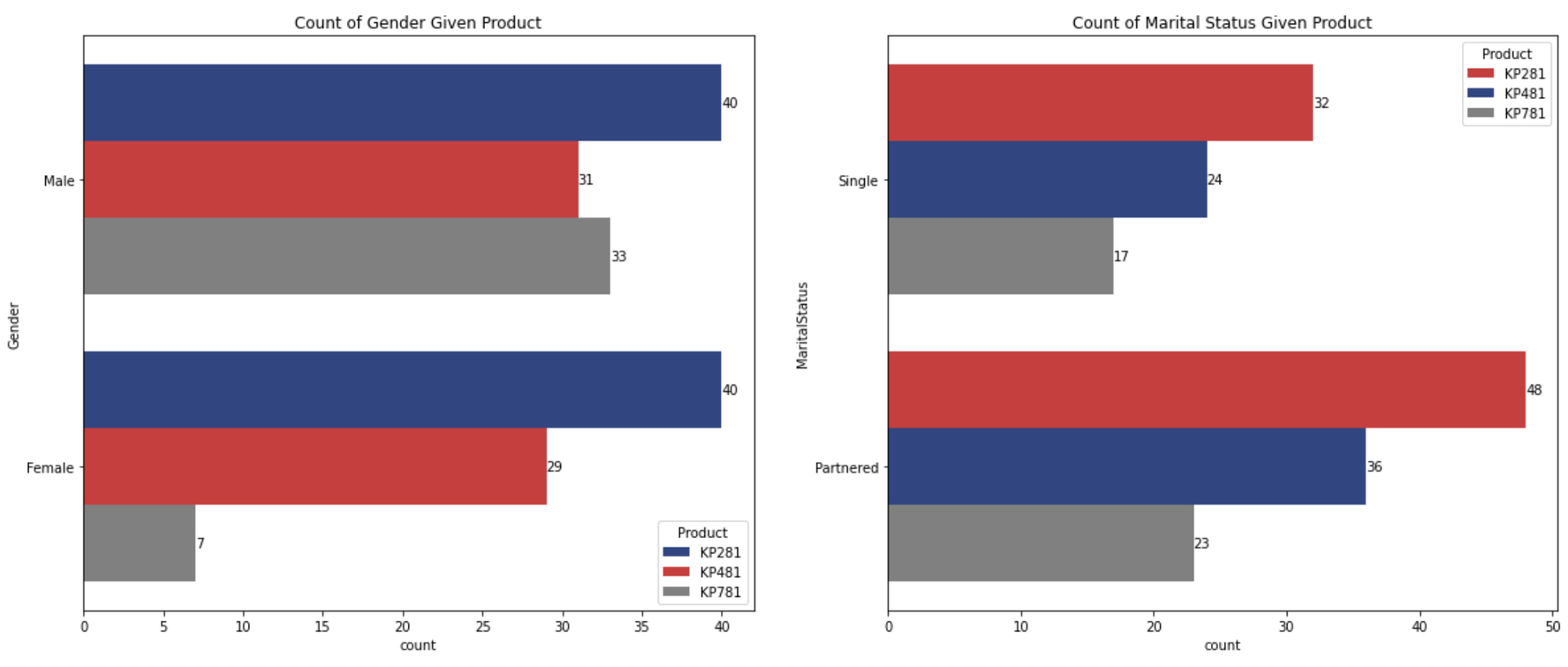


Observations:

- **Male Customer: We see that partnered and male purchased threadmills KP281, KP481 equally.**
- **Female Customer: We see that partnered and female purchase threadmill KP281 the most.**

In [362]:

```
plt.figure(figsize = (20,8))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(1,2,1)
ax = sns.countplot(data = df, y = "Gender", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
for container in ax.containers:
    ax.bar_label(container)
plt.title("Count of Gender Given Product", size = 12)

plt.subplot(1,2,2)
ax = sns.countplot(data = df, y = "MaritalStatus", hue = "Product", palette = [aero_red,aero_blue,aero_grey])
for container in ax.containers:
    ax.bar_label(container)
plt.title("Count of Marital Status Given Product", size = 12)

plt.suptitle("Categorical Count", size = 18, fontweight = "medium")
plt.show()
```



Observations:

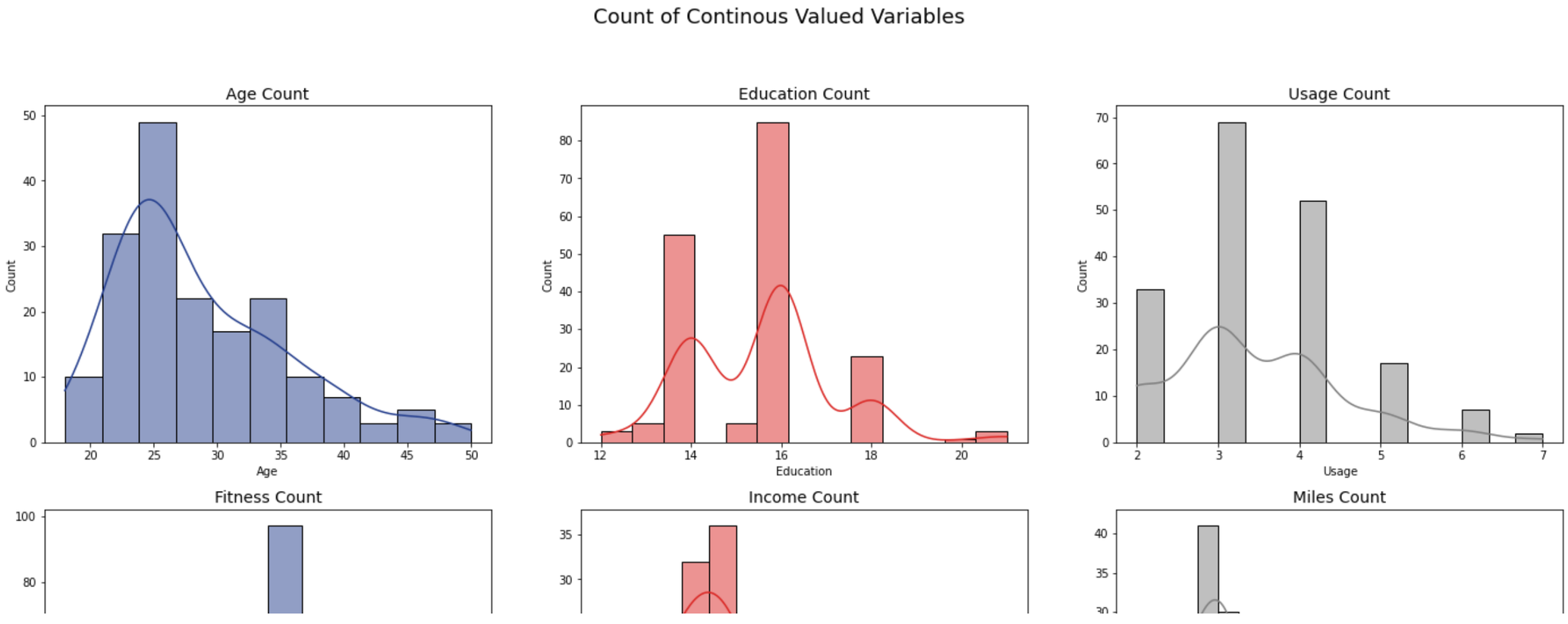
- **KP281 is the highest sold product.**
- **Both male and female have bought KP281 equally and also individually highest bought product by them.**

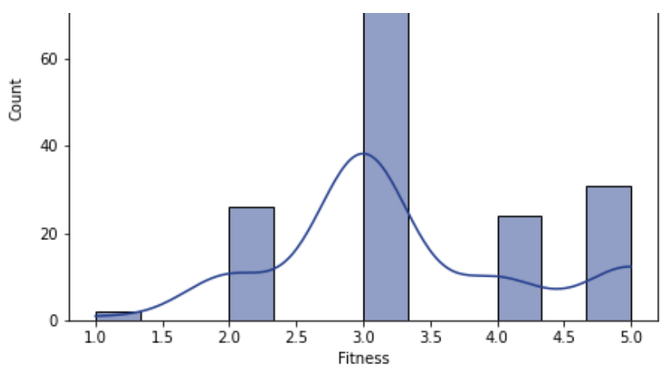
- **Both singles and partnered customers also have bought KP281 the highest threadmill.**

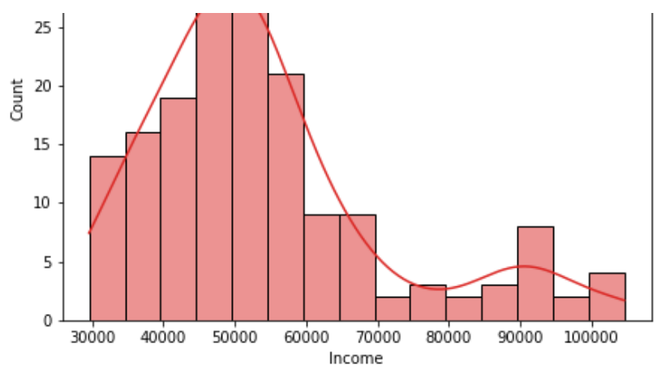
Histplots

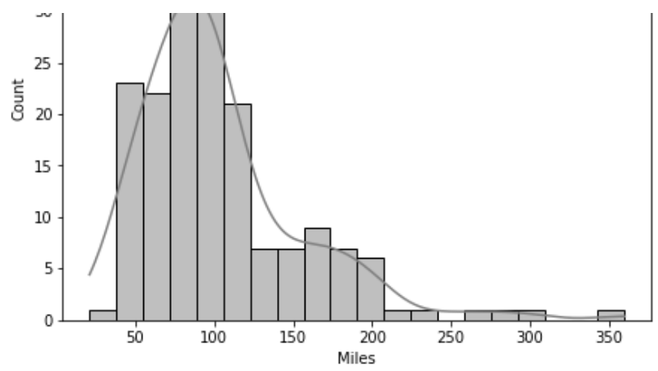
In [363]:

```
plt.figure(figsize = (24,12))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(2,3,1)
sns.histplot(data = df, x = "Age", kde = True, color = aero_blue)
plt.title("Age Count", size = 14)

plt.subplot(2,3,2)
sns.histplot(data = df, x = "Education", kde = True, color = aero_red)
plt.title("Education Count", size = 14)

plt.subplot(2,3,3)
sns.histplot(data = df, x = "Usage", kde = True, color = aero_grey)
plt.title("Usage Count", size = 14)

plt.subplot(2,3,4)
sns.histplot(data = df, x = "Fitness", kde = True, color = aero_blue)
plt.title("Fitness Count", size = 14)

plt.subplot(2,3,5)
sns.histplot(data = df, x = "Income", kde = True, color = aero_red)
plt.title("Income Count", size = 14)

plt.subplot(2,3,6)
sns.histplot(data = df, x = "Miles", kde = True, color = aero_grey)
plt.title("Miles Count", size = 14)

plt.suptitle("Count of Continous Valued Variables", size = 18)
plt.show()
```





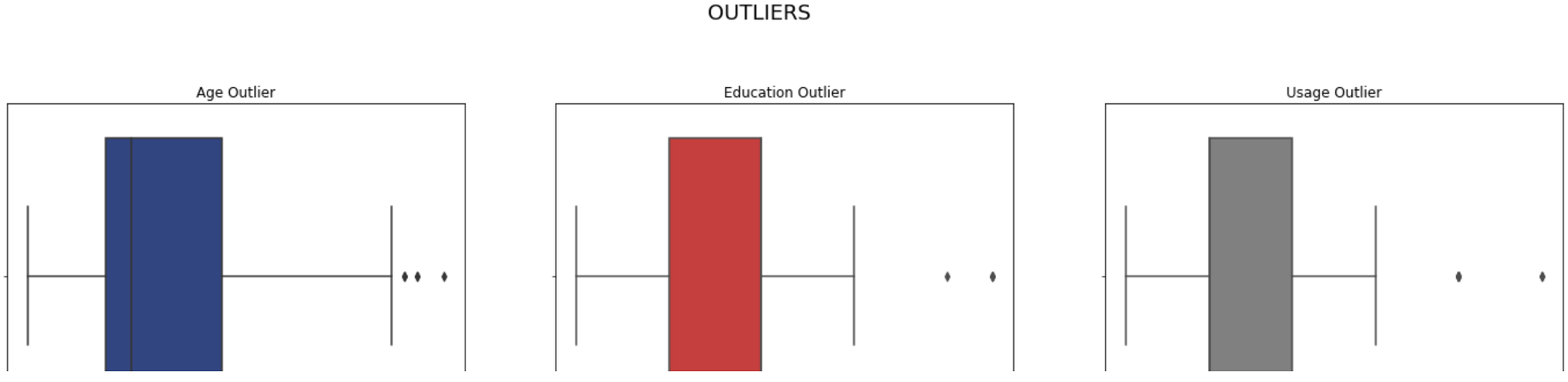


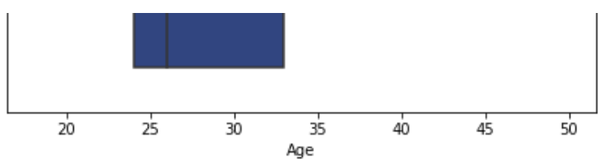


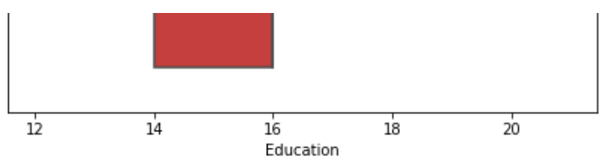
Boxplots

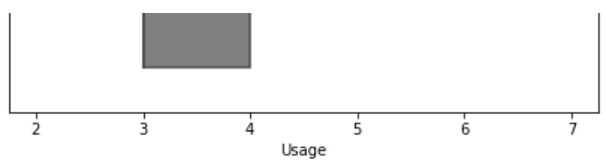
In [357]:

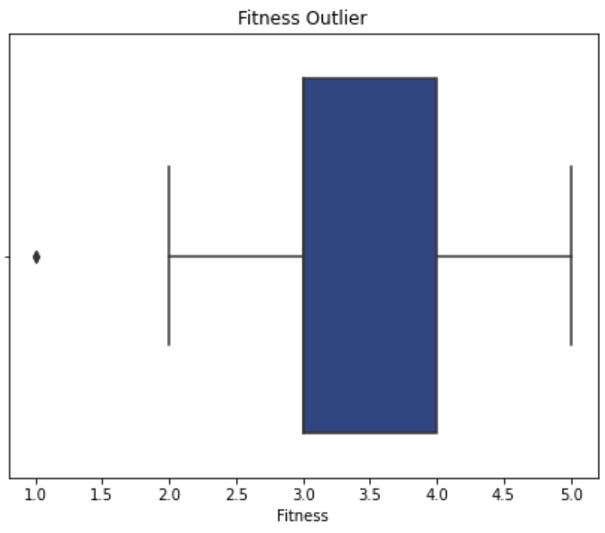
```
plt.figure(figsize = (24,12))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(2,3,1)
sns.boxplot(data = df, x = "Age", orient = "h", color = aero_blue)
plt.title("Age Outlier", size = 12)

plt.subplot(2,3,2)
sns.boxplot(data = df, x = "Education", orient = "h", color = aero_red)
plt.title("Education Outlier", size = 12)

plt.subplot(2,3,3)
sns.boxplot(data = df, x = "Usage", orient = "h", color = aero_grey)
plt.title("Usage Outlier", size = 12)

plt.subplot(2,3,4)
sns.boxplot(data = df, x = "Fitness", orient = "h", color = aero_blue)
plt.title("Fitness Outlier", size = 12)

plt.subplot(2,3,5)
sns.boxplot(data = df, x = "Income", orient = "h", color = aero_red)
plt.title("Income Outlier", size = 12)

plt.subplot(2,3,6)
sns.boxplot(data = df, x = "Miles", orient = "h", color = aero_grey)
plt.title("Miles Outlier", size = 12)

plt.suptitle("OUTLIERS", size = 18, fontweight = "medium")
plt.show()
```

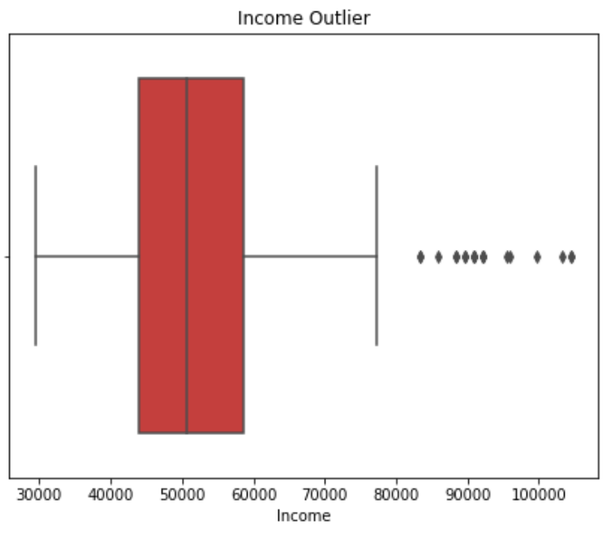


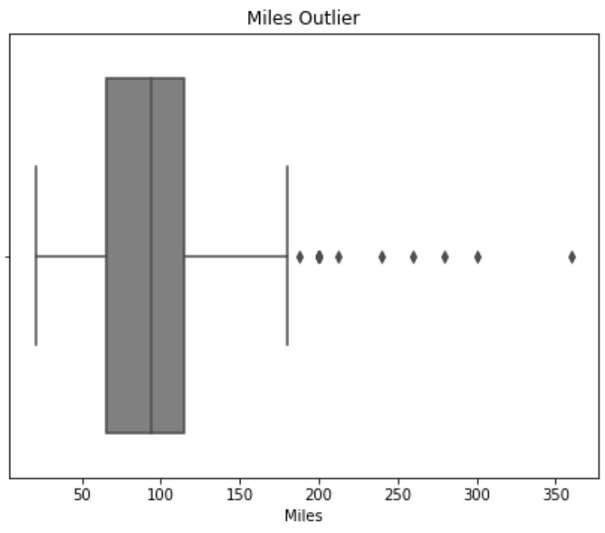










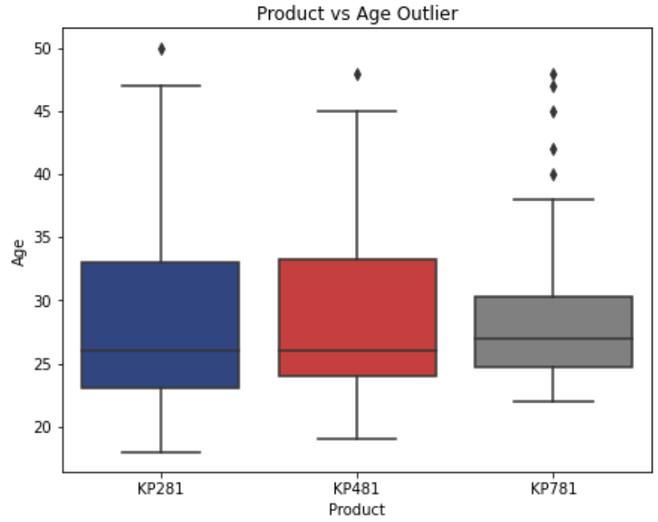


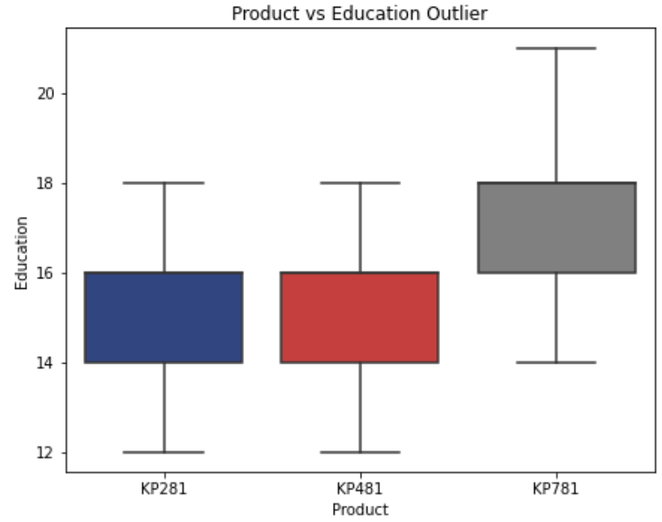
Observations:

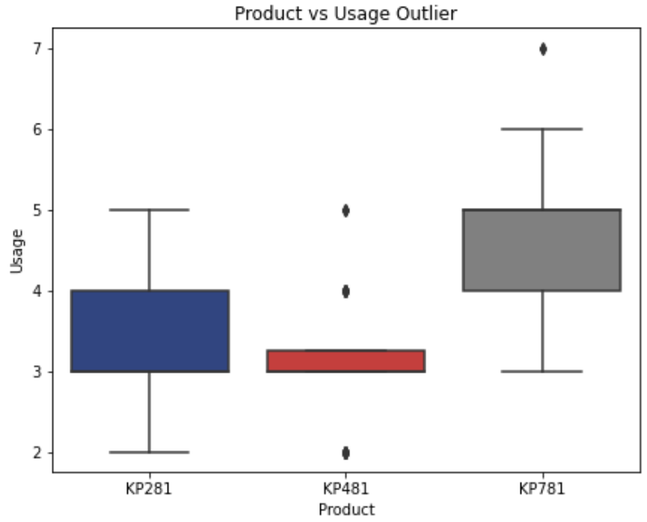
- **It is clear that both Income and Miles have got the higest outliers and the rest have minimum or lower number of outlier values.**

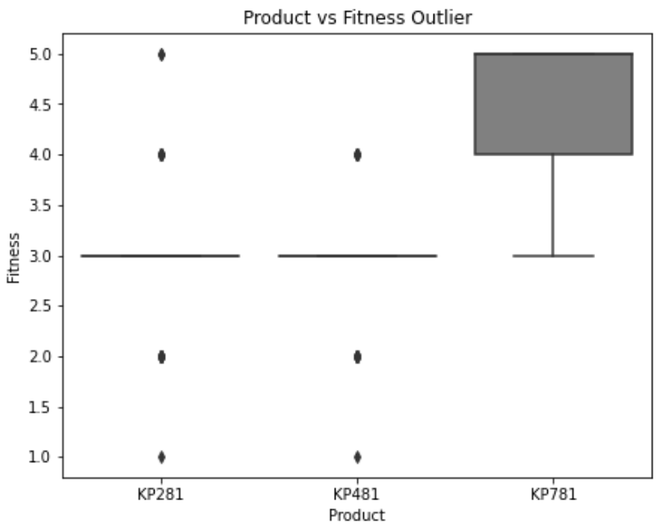
In [366]:

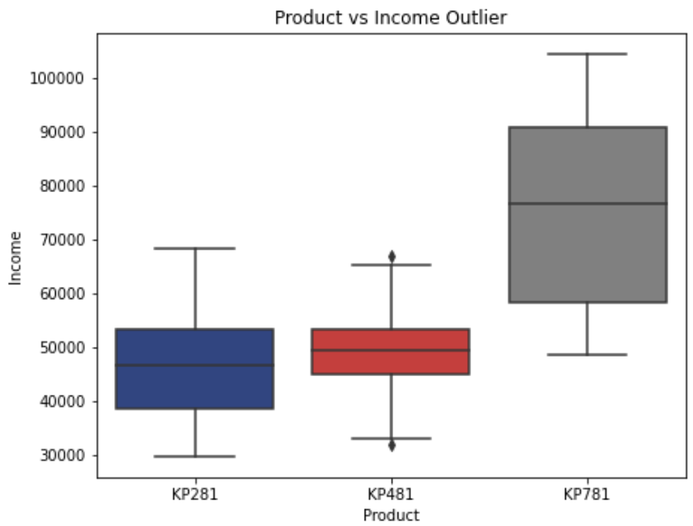
```
plt.figure(figsize = (24,12))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(2,3,1)
sns.boxplot(data = df, x = "Product", y = "Age", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Age Outlier", size = 12)

plt.subplot(2,3,2)
sns.boxplot(data = df, x = "Product", y = "Education", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Education Outlier", size = 12)

plt.subplot(2,3,3)
sns.boxplot(data = df, x = "Product", y = "Usage", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Usage Outlier", size = 12)

plt.subplot(2,3,4)
sns.boxplot(data = df, x = "Product", y = "Fitness", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Fitness Outlier", size = 12)

plt.subplot(2,3,5)
sns.boxplot(data = df, x = "Product", y = "Income", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Income Outlier", size = 12)

plt.subplot(2,3,6)
sns.boxplot(data = df, x = "Product", y = "Miles", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Miles Outlier", size = 12)

plt.suptitle("BI-VARIATE OUTLIERS", size = 18, fontweight = "medium")
plt.show()
```

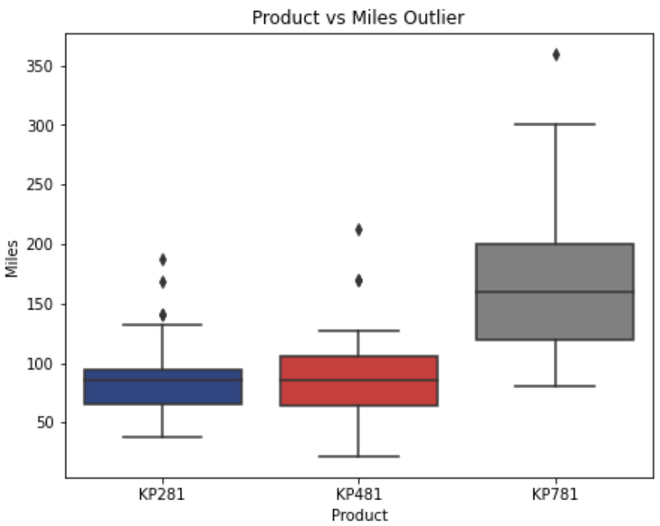












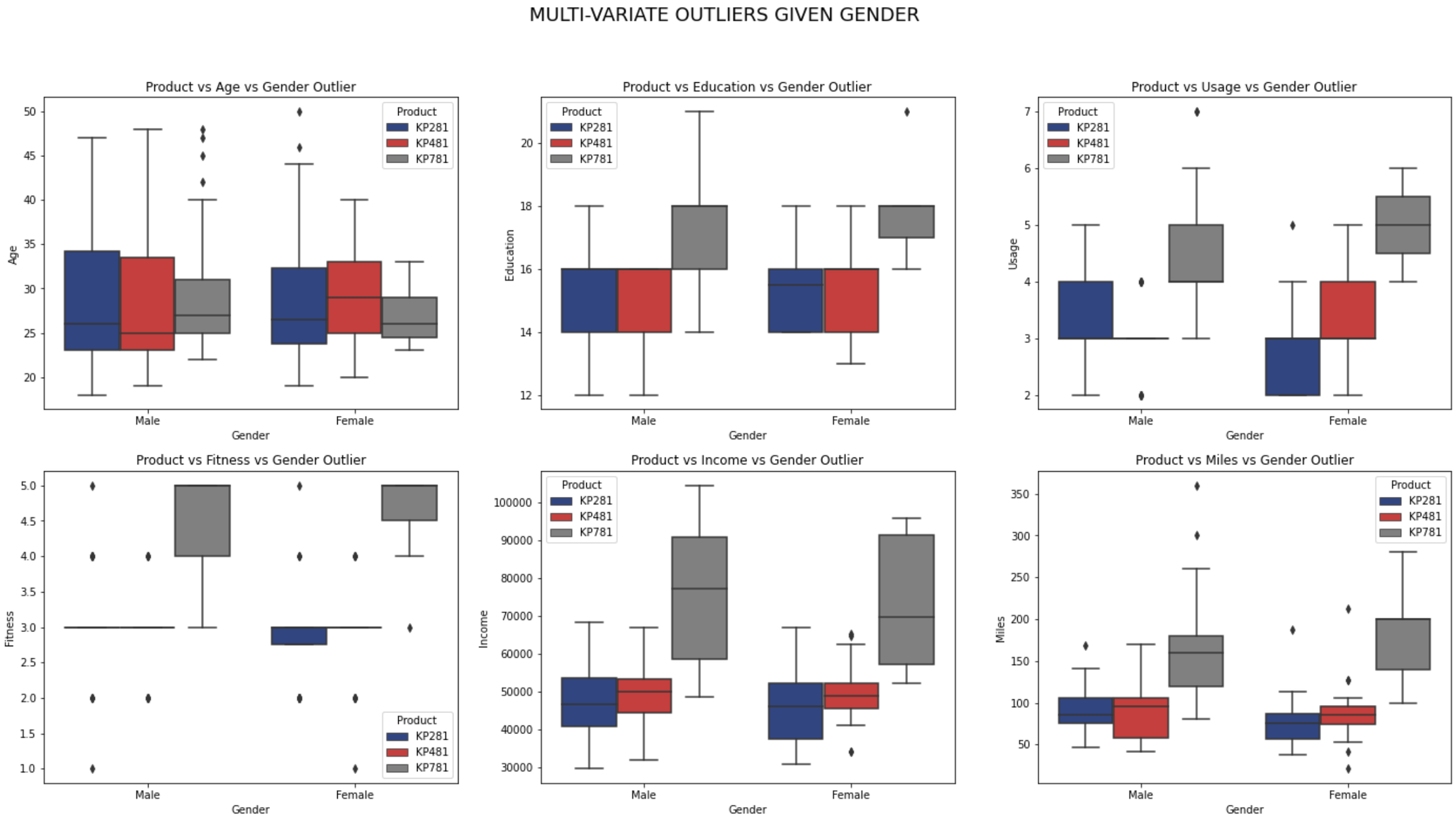
Observations:

- **Product vs Age Outlier:** We can see that the customer purchasing products KP281 and KP481 have a similar median. While the median of customer purchasing for product KP781 is slightly higher than the other two products we can say that the customers of age between 25 to 30 are more likely to buy the product KP781. We can also observe that for the product KP781 we have high outliers for customers with an older age than 35.
- **Product vs Education Outlier:** Customers with 14 years to 16 years of education have mostly purchased products KP281, KP481. Customers with 16 years to 18years of education have purchased product KP781.
- **Product vs Usage Outlier:** Customers who are planning to use threadmill 3 times a week have bought products KP281 and KP481. Customers who intent to use upto 4 times have also bought the product KP281. Customers who have intent to use more than 4 times have purchased the product KP781.
- **Product vs Fitness Outlier:** Customers who felt they have fitness of 3 have purchased the products KP281, KP481. Customers who felt they have a higher fitness level have purchased the product KP781.
- **Product vs Income Outlier:** The lower the income group of the customers they have purchased mostly KP281. The higher the income group of the customers they have purchased mostly KP781.
- **Product vs Miles Outlier:** Customers who intend to walk less than 100miles a week have purchased products KP281, KP481. While customers who intend to walk more than 100 miles a week have purchased the product KP781.

In [365]:

```
plt.figure(figsize = (24,12))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(2,3,1)
sns.boxplot(data = df, x = "Gender", y = "Age", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Age vs Gender Outlier", size = 12)

plt.subplot(2,3,2)
sns.boxplot(data = df, x = "Gender", y = "Education", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Education vs Gender Outlier", size = 12)
```

```
plt.subplot(2,3,3)
sns.boxplot(data = df, x = "Gender", y = "Usage", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Usage vs Gender Outlier", size = 12)

plt.subplot(2,3,4)
sns.boxplot(data = df, x = "Gender", y = "Fitness", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Fitness vs Gender Outlier", size = 12)

plt.subplot(2,3,5)
sns.boxplot(data = df, x = "Gender", y = "Income", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Income vs Gender Outlier", size = 12)

plt.subplot(2,3,6)
sns.boxplot(data = df, x = "Gender", y = "Miles", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Miles vs Gender Outlier", size = 12)

plt.suptitle("MULTI-VARIATE OUTLIERS GIVEN GENDER", size = 18, fontweight = "medium")
plt.show()
```



Observations:

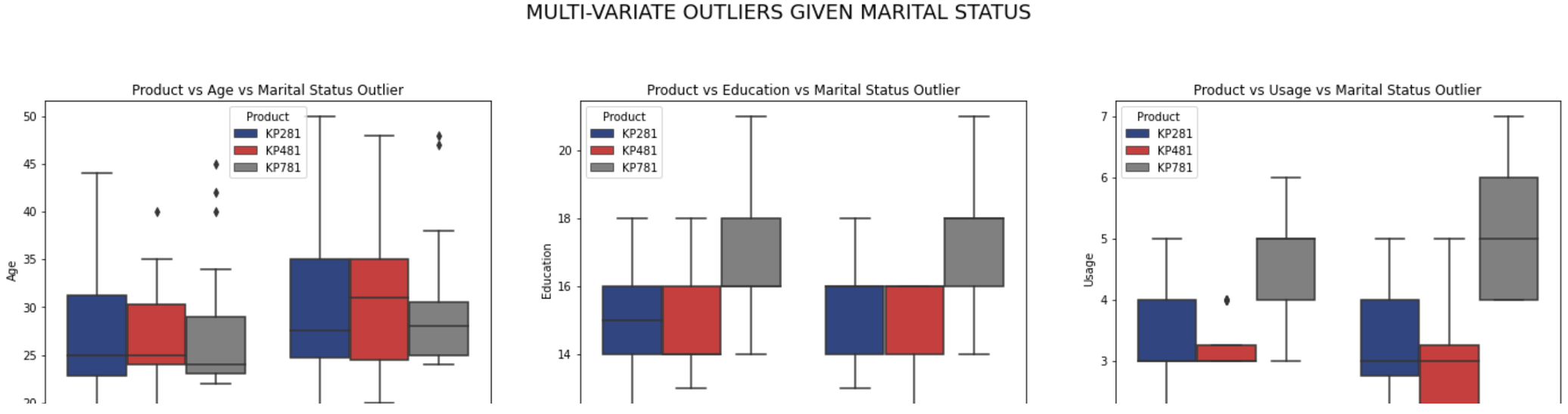
- **Gender: Male**
  - **Male between the age of 25 and 30 have mostly purchased product KP781**

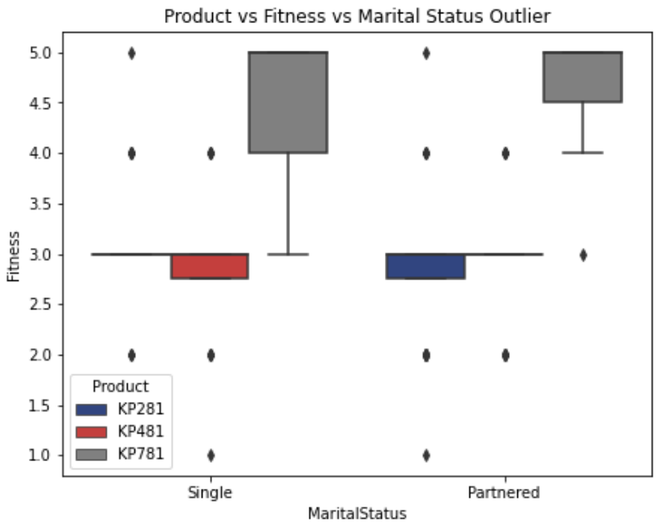
- **Male between the age of 25 and 30 have mostly purchased product KP781.**
- **Male above age of 40 have a lot of outlier who buy product KP781, so can be avoided to be targetted.**
- **Male under 16 years of education are have mostly purchased either products KP281 or KP481 and people who have education higher than 16 years have purchased product KP781.**
- **Male who claim to use the threadmill more than 4 times a week have purchased product KP781.**
- **Male who claim to have fitness more than 4 as their fitness level have purchased product KP781.**
- **Male make income more than 60,000 dollars have purchased KP781.**
- **Male who claim to walk or run higher number of miles have purchased KP781.**

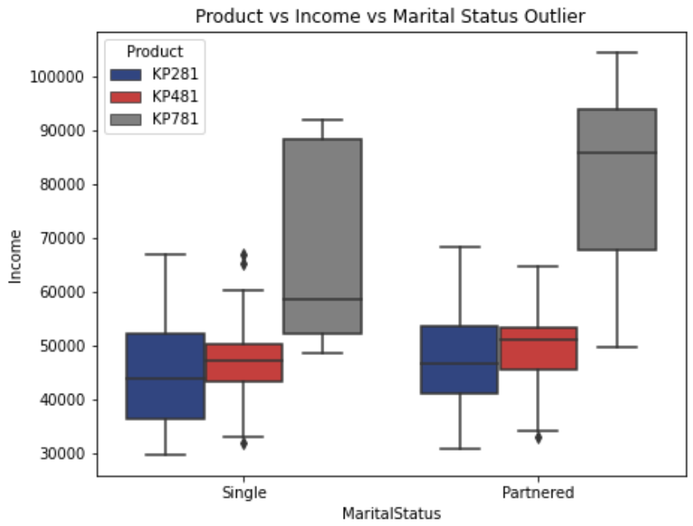
- **Gender: Female**
  - **Female with education more than 17 years have purchased the product KP781.**
  - **Female who claimed to use the threadmill more than 4 times a week have purchased the product KP781.**
  - **Female who claim to have fitness level above 4 have mostly purchased the product KP781.**
  - **Female with income more than $ 55,000 have purchased the product KP781.**
  - **Female who claimed to walk or run higher miles have purchase the product KP781.**

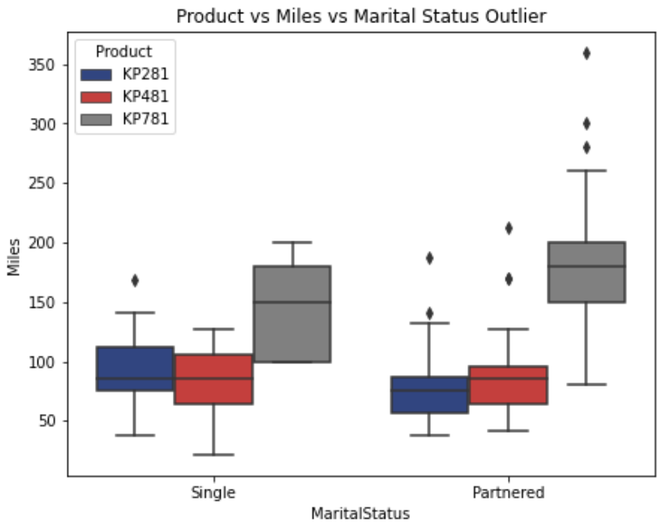
In [364]:

```
plt.figure(figsize = (24,12))
aero_blue = "#243e8d"
aero_grey = "#808080"
aero_red = "#db2926"

plt.subplot(2,3,1)
sns.boxplot(data = df, x = "MaritalStatus", y = "Age", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Age vs Marital Status Outlier", size = 12)

plt.subplot(2,3,2)
sns.boxplot(data = df, x = "MaritalStatus", y = "Education", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Education vs Marital Status Outlier", size = 12)

plt.subplot(2,3,3)
sns.boxplot(data = df, x = "MaritalStatus", y = "Usage", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Usage vs Marital Status Outlier", size = 12)

plt.subplot(2,3,4)
sns.boxplot(data = df, x = "MaritalStatus", y = "Fitness", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Fitness vs Marital Status Outlier", size = 12)

plt.subplot(2,3,5)
sns.boxplot(data = df, x = "MaritalStatus", y = "Income", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Income vs Marital Status Outlier", size = 12)

plt.subplot(2,3,6)
sns.boxplot(data = df, x = "MaritalStatus", y = "Miles", hue = "Product", palette = [aero_blue,aero_red,aero_grey])
plt.title("Product vs Miles vs Marital Status Outlier", size = 12)

plt.suptitle("MULTI-VARIATE OUTLIERS GIVEN MARITAL STATUS", size = 18, fontweight = "medium")
plt.show()
```









Observations:

- **Single:**
  - **Single customer have a similar median value of 25 for both products KP281 and KP481.**
  - **Single customers with education more than 16 years have mostly bought the product KP781.**
  - **Single who have chosen to use the threadmill for more than 4 times a week have mostly bought product KP781.**
  - **Single customers with more than 4 fitness level have purchasesed KP781.**
  - **Single customer with income over $ 50,000 have purchased KP781.**
- **Partnered:**
  - **Partnered customers with education more than 16 years have mostly bought the product KP781.**
  - **Partnered customers who have chosen to use the threadmill for more than 4 times a week have mostly bought product KP781.**
  - **Partnered customers with more than 4.5 fitness level have purchased the product KP781.**
  - **Partnered customers with income more than $ 70,000 have purchased KP781.**

Heatmap

In [342]:

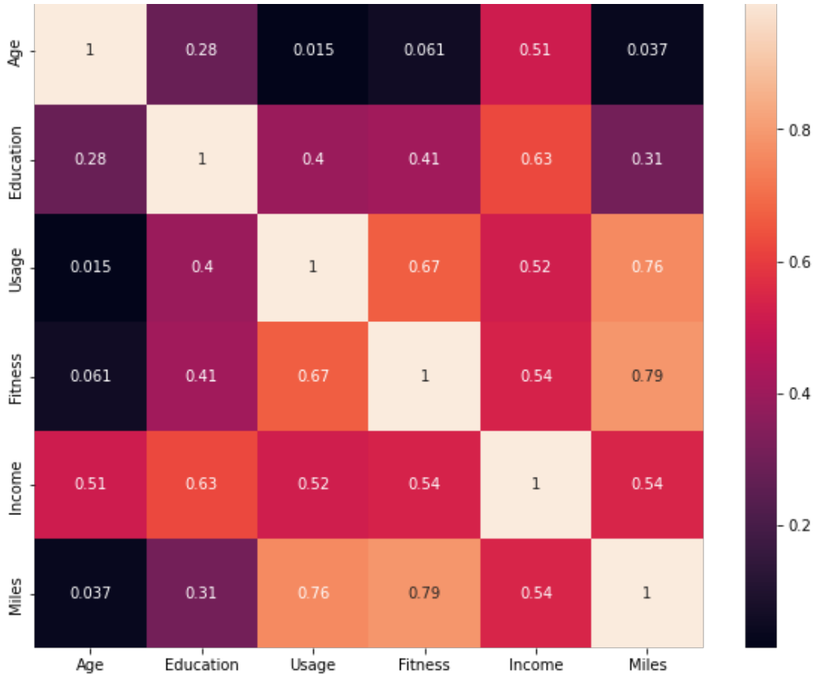
```
df.corr()
```

Out[342]:

	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 [346]:

```
plt.figure(figsize = (10,8))
sns.heatmap(df.corr(), annot = True)
plt.show()
```

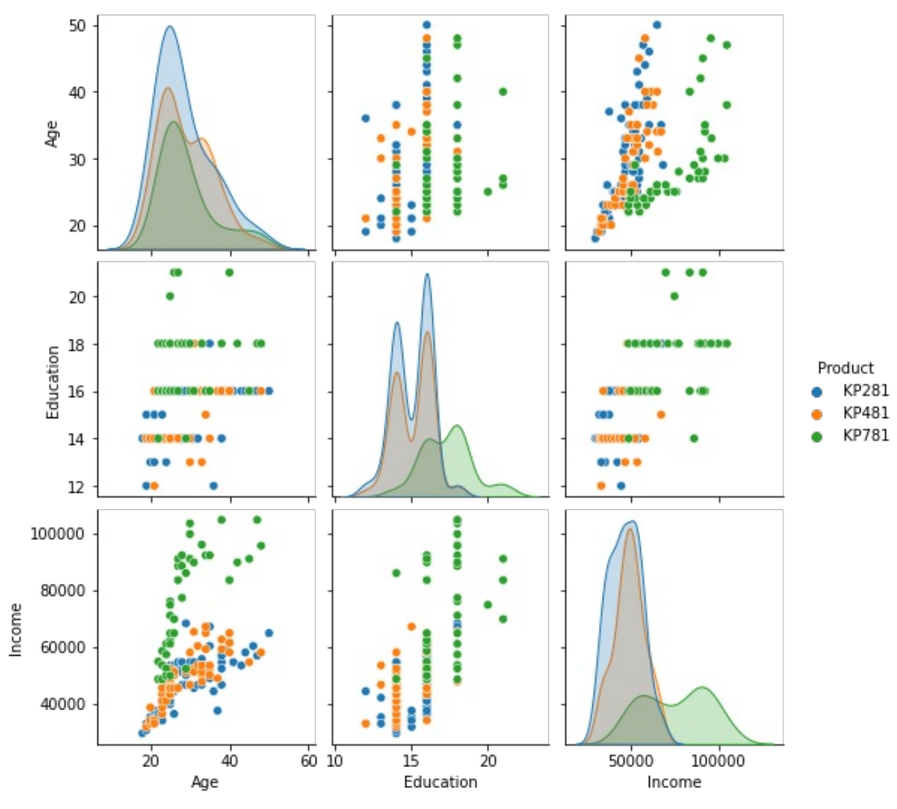


Pair Plots

In [352]:

```
plt.figure(figsize = (10,8))
sns.pairplot(data = df[["Product", "Age", "Education", "Income"]], hue = "Product")
plt.show()
```

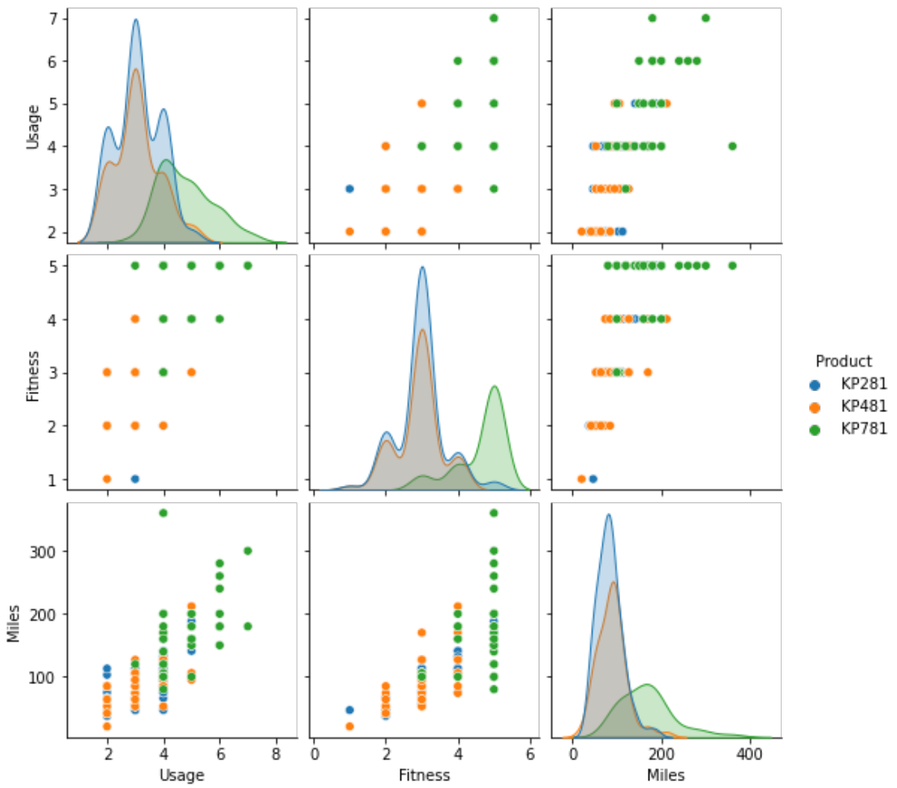
```
<Figure size 720x576 with 0 Axes>
```



In [353]:

```
plt.figure(figsize = (10,8))
sns.pairplot(data = df[["Product", "Usage", "Fitness", "Miles"]], hue = "Product")
plt.show()
```

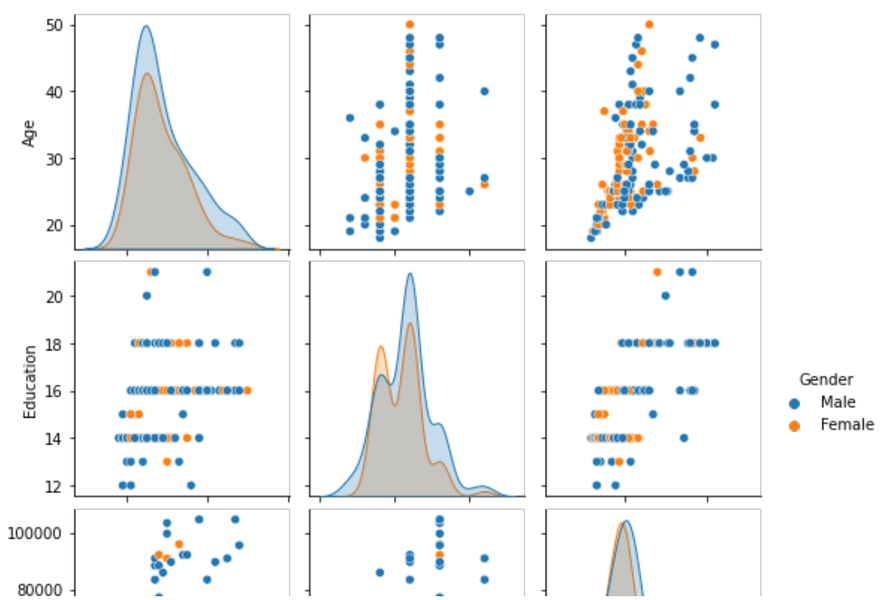
```
<Figure size 720x576 with 0 Axes>
```

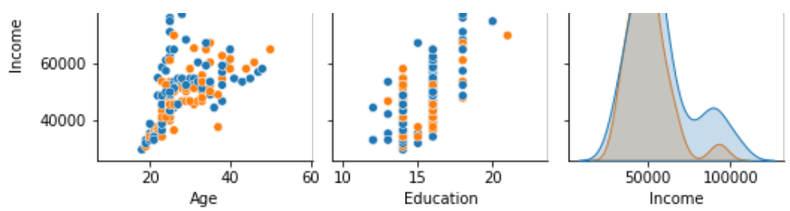


In [356]:

```
plt.figure(figsize = (10,8))
sns.pairplot(data = df[["Gender", "Age", "Education", "Income"]], hue = "Gender")
plt.show()
```

```
<Figure size 720x576 with 0 Axes>
```

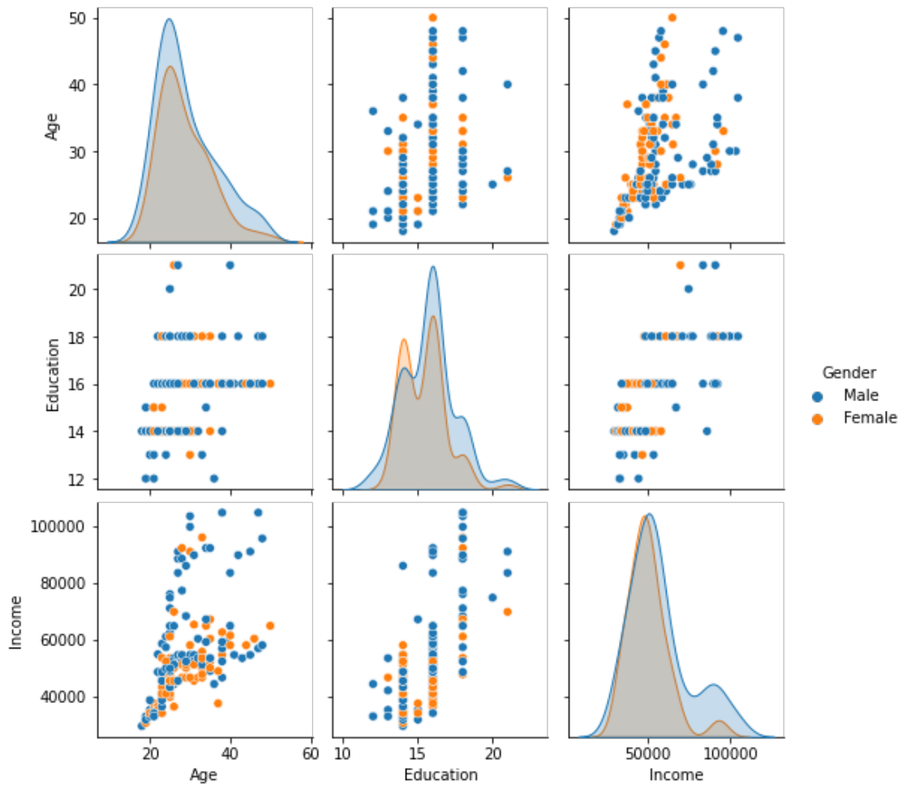




In [355]:

```
plt.figure(figsize = (10,8))
sns.pairplot(data = df[["Gender", "Age", "Education", "Income"]], hue = "Gender")
plt.show()
```

```
<Figure size 720x576 with 0 Axes>
```



Recommendations

1. **It is recommended to targeted customer who are educated above 17 years as their probability of purchasing a high end model as KP781 is more.**
2. **It is recommended to target customers who already have the sense of being fit and maintain a higher level of fitness as their probability of purchasing KP781 becomes higher.**
3. **It is receommended to target customers with higher income levels so that the sales of KP781 can be increase and in return it would increase revenue for the company.**
4. **KP281 is the most purchased product and then KP781 also get's purchased as the second highest purchased product. But it is recommended with some offers to sell the mid range product KP481.**
5. **It is recommended to target both Male and Female above the ages of 25 and income more than $ 55,000 for higher sales of KP781.**

In [ ]: