Business Case Study: AeroFit Treadmill

About Aerofit:

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

Business Problem:

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

1. Perform descriptive analytics to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts.

2. For each AeroFit treadmill product, construct two-way contingency tables and compute all conditional and marginal probabilities along with their insights/impact on the business.

Dataset:

The company collected the data on individuals who purchased a treadmill from the AeroFit stores during the prior three months. The dataset has the following features:

Product Purchased:	KP281, KP481, or KP781
Age:	In years
Gender:	Male/Female
Education:	In years
Marital Status:	Single or partnered
Usage:	The average number of times the customer plans to use the treadmill each week.
Income:	Annual income (in $)
Fitness:	Self-rated fitness on a 1-to-5 scale, where 1 is the poor shape and 5 is the excellent shape.
Miles:	The average number of miles the customer expects to walk/run each week

Product Portfolio:
The KP281 is an entry-level treadmill that sells for $1,500.
The KP481 is for mid-level runners that sell for $1,750.
The KP781 treadmill is having advanced features that sell for $2,500.

1. Import the dataset and do usual data analysis steps like checking the structure & characteristics of the dataset

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import binom as bm
```

[8]:
```
customers = pd.read_csv('Case Study/Aerofit_trademill.csv')
```

[9]:
```
customers.head()
```

[9]:

	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

a. The data type of all columns in the "customers" table.

[12]:
```
customers.dtypes
```

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

Insights:

1. From the above we can see that the 'Product', 'Gender' and 'MaritalStatus' columns are string and 'Age', 'Education', 'Usage', 'Fitness', 'Income' and 'Miles' columns are of integer data type.

2. The data types are as expected which is mentioned in the dataset description.

3. From the columns we can say 3 categorical columns and 6 numerical columns are present.

b. The number of rows and columns given in the dataset

```
[10]: customers.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
```

```
[11]: customers.shape
```

```
[11]: (180, 9)
```

Insights:

1. We have 180 rows and 9 columns present in the dataset.
2. The memory usage by the dataset is 12.8 KB
3. The rows are present with implicit indexing of 0 to 179.

c. Check for the missing values and find the number of missing values in each column

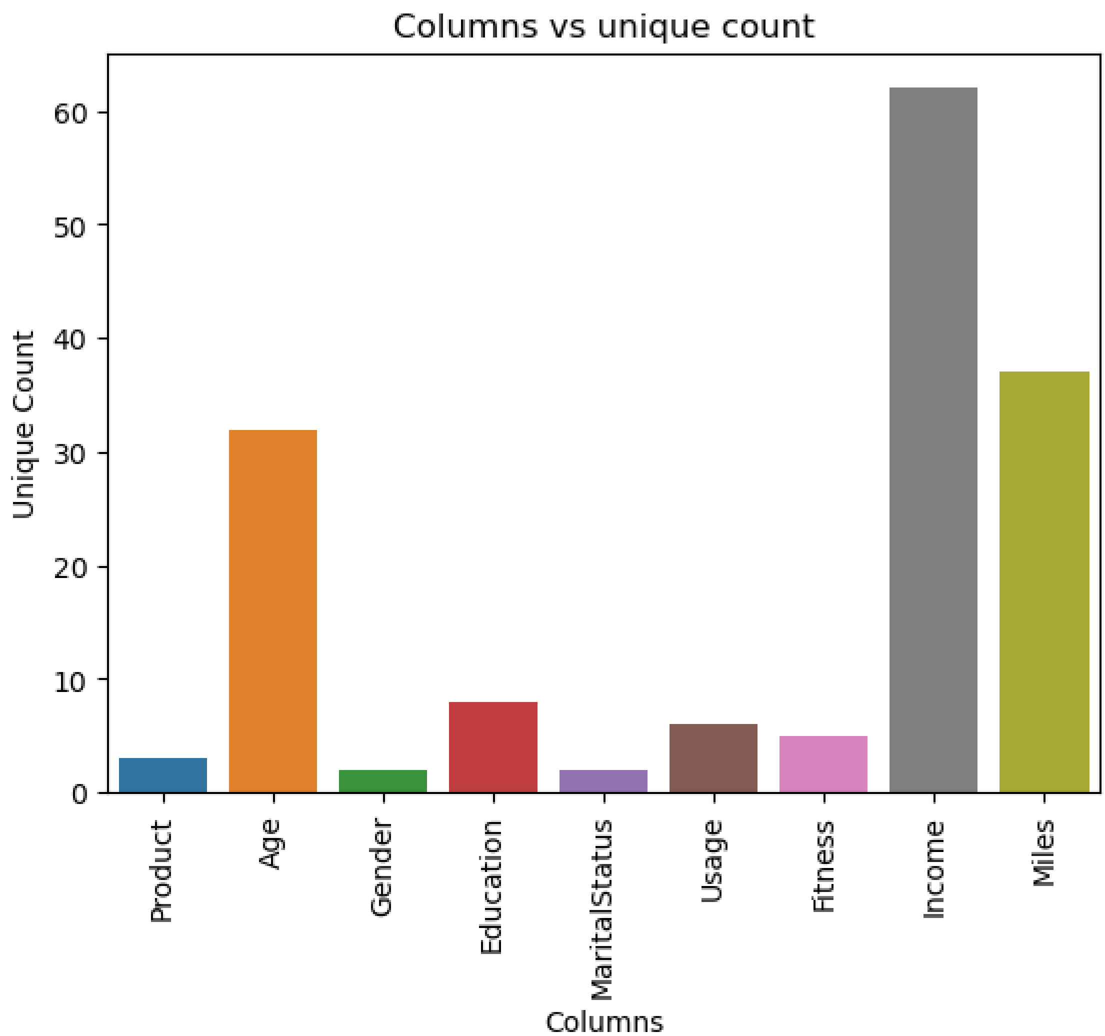
```
[13]: customers.isnull().sum()
```

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

Insights:

From above we can see for each column we don't have any null or NaN values are present.

```
[24]: sns.barplot(data = df_unique, x = 'index', y = 0)
plt.title('Columns vs unique count')
plt.xlabel('Columns')
plt.ylabel('Unique Count')
plt.xticks(rotation = 90)
plt.show()
```



The above graph depicts that though we have 108 rows but columns have different unique values.

[31]:
```python
for i in customers.columns:
    print(f'{i} column have below unique values:')
    print(customers[i].unique())
```

```
Product column have below unique values:
['KP281' 'KP481' 'KP781']
Age column have below unique values:
[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]
Gender column have below unique values:
['Male' 'Female']
Education column have below unique values:
[14 15 12 13 16 18 20 21]
MaritalStatus column have below unique values:
['Single' 'Partnered']
Usage column have below unique values:
[3 2 4 5 6 7]
Fitness column have below unique values:
[4 3 2 1 5]
Income column have below unique values:
[ 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]
Miles column have below unique values:
[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]
```

[45]:
```python
for i in customers.columns:
    if customers.dtypes[i] == 'int64':
        print('The minmum and maximum value of '+i+ ' is:'+ str(customers[i].min()) +' and '+ str(customers[i].max()))
```

```
The minmum and maximum value of Age is:18 and 50
The minmum and maximum value of Education is:12 and 21
The minmum and maximum value of Usage is:2 and 7
The minmum and maximum value of Fitness is:1 and 5
The minmum and maximum value of Income is:29562 and 104581
The minmum and maximum value of Miles is:21 and 360
```

With the above analysis lets grouping the data in multiple groups,

[279]:
```python
bin_range1 = [17,25,35,45,float('inf')]
bin_labels1 = ['Young Adults', 'Adults', 'Middle Aged Adults', 'Elder']

customers['Age_Group'] = pd.cut(customers['Age'],bins = bin_range1,labels = bin_labels1)

bin_range2 = [0,12,15,float('inf')]
bin_labels2 = ['Primary Education', 'Secondary Education', 'Higher Education']

customers['Education_Group'] = pd.cut(customers['Education'],bins = bin_range2,labels = bin_labels2)

bin_range3 = [0,40000,60000,80000,float('inf')]
bin_labels3 = ['Low Income','Moderate Income','High Income','Very High Income']

customers['Income_Group'] = pd.cut(customers['Income'],bins = bin_range3,labels = bin_labels3)

bin_range4 = [0,50,100,200,float('inf')]
bin_labels4 = ['Light Activity', 'Moderate Activity', 'Active Lifestyle', 'Fitness Enthusiast']

customers['Miles_Group'] = pd.cut(customers['Miles'],bins = bin_range4,labels = bin_labels4)
```

General Insights:

After doing the all above analysis we can observe the below points:

1. The given data set have two type of data types ‘string’ and ‘integer.
2. Dataset have 9 columns out of which 7 are numerical and 2 are categorical.
3. The dataset don’t have any null values in 108 rows.
4. The dataset don’t have any nested columns.
5. The dataset have below categorical data:
   a. 3 type of treadmill → ‘KP281’, ‘KP481’ and ‘KP781’.
   b. 2 type of gender → ‘Male’ and ‘Female’
   c. 2 types of marital status → ‘Single’ and ‘Partnered’
6. The observation from numerical columns:
   a. customers are belonging to age between 18 to 50.
   b. The education range is between 12 years to 21 years.
   c. The usage of treadmill lies between 2 to 7 time each week.
   d. The self-fitness rating is 1 to 5, from which we see we have unfit customers and fit customers data.
   e. The annual income range in the given dataset is between 29562$ to 104581$.
   f. The weekly range of miles is between 21 to 360 miles.

2. Detect Outliers:

```
for i in customers.columns:
    if customers.dtypes[i] == 'int64':
        print('The minmum and maximum value of '+i+ ' is:'+ str(customers[i].min()) +' and '+ str(customers[i].max()))
```

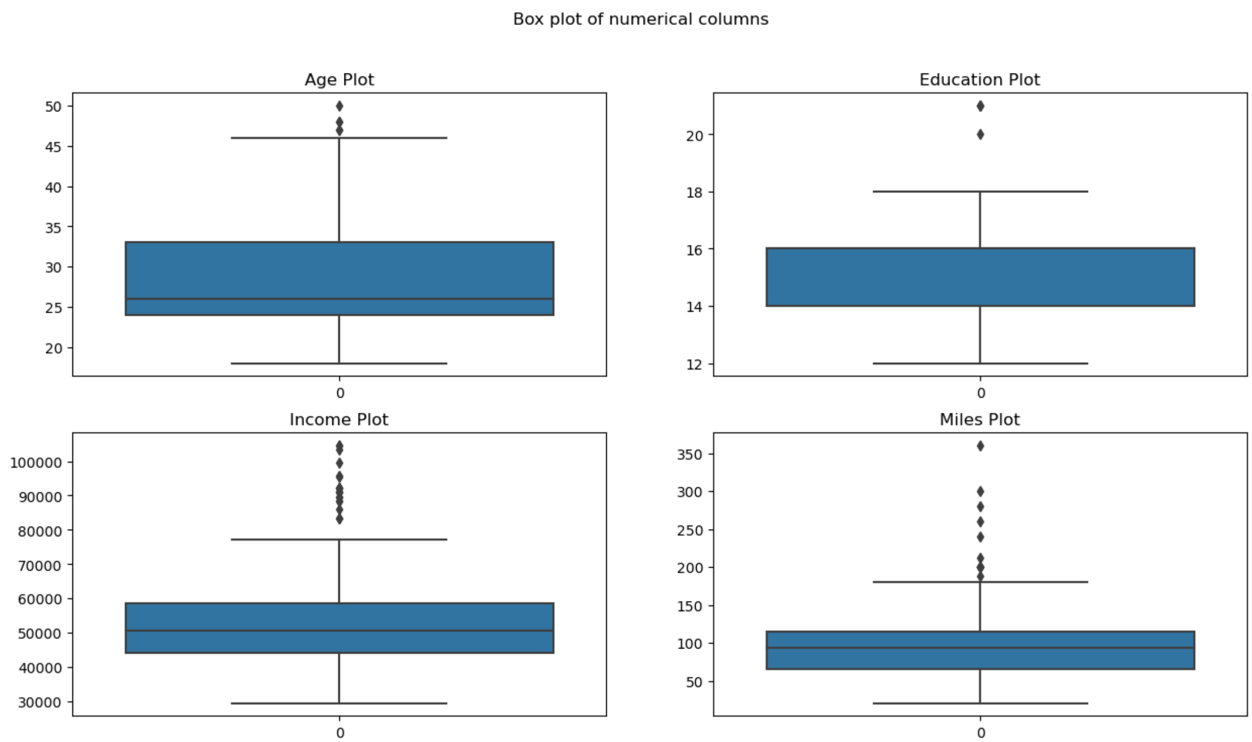
```
The minmum and maximum value of Age is:18 and 50
The minmum and maximum value of Education is:12 and 21
The minmum and maximum value of Usage is:2 and 7
The minmum and maximum value of Fitness is:1 and 5
The minmum and maximum value of Income is:29562 and 104581
The minmum and maximum value of Miles is:21 and 360
```

Above columns are numerical columns among which Age, Education, Income and Miles are continuous variables. Let’s analyze the outliers.

a. Find the outliers for every continuous variable in the dataset:

Below analysis is for outlier detection

```
[147]: plt.figure(figsize = (15,8))
plt.suptitle('Box plot of numerical columns')

plt.subplot(2,2,1)
sns.boxplot(data = customers['Age'])
plt.title('Age Plot')

plt.subplot(2,2,2)
sns.boxplot(data = customers['Education'])
plt.title('Education Plot')

plt.subplot(2,2,3)
sns.boxplot(data = customers['Income'])
plt.title('Income Plot')

plt.subplot(2,2,4)
sns.boxplot(data = customers['Miles'])
plt.title('Miles Plot')

plt.show()
```



i. Age: In Years

In the given dataset the age is lies between 18 to 50.

```
[66]: print("Mean/Average: {}".format(customers['Age'].mean()))
      print("Median: {}".format(customers['Age'].median()))
      print("Mode: {}".format(customers['Age'].mode().values[0]))
      print("Standard Deviation: {}".format(customers['Age'].std()))

      Mean/Average: 28.788888888888888
      Median: 26.0
      Mode: 25
      Standard Deviation: 6.943498135399795
```

The dataset has average age of 28.78 years, median is 26, maximum customers are 25 years old and the standard deviation on customer's age 6.94 years. The outliers are which are below 1.5IQR or above 1.5IQR.

Let's find the IQR:

```
[72]: customers['Age'].quantile(0.25)
[72]: 24.0

[73]: customers['Age'].quantile(0.5)
[73]: 26.0

[74]: customers['Age'].quantile(0.75)
[74]: 33.0

[75]: q1_s= customers['Age'].quantile(0.25)
      q3_s= customers['Age'].quantile(0.75)
      iqr_s = q3_s-q1_s
```

IQR: 9.0

[78]:
```
print(f'The IQR value is {iqr_s}')
```

```
The IQR value is 9.0
```

[79]:
```
q1_s-1.5*iqr_s, q3_s+1.5*iqr_s
```

[79]:
```
(10.5, 46.5)
```

The outliers are,

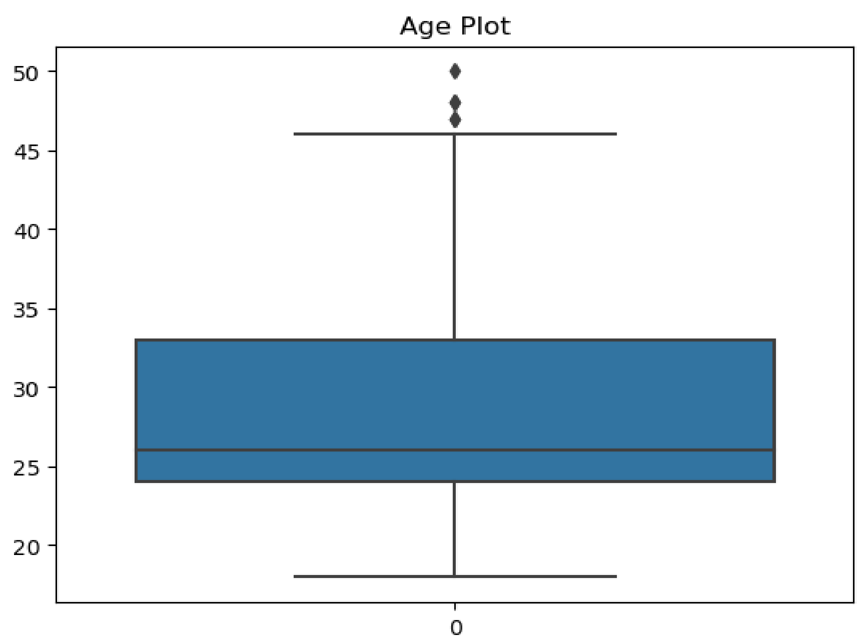
[76]:
```
customers[(customers['Age']<(q1_s-1.5*iqr_s)) | (customers['Age']>(q3_s+1.5*iqr_s)) ]
```

[76]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
78	KP281	47	Male	16	Partnered	4	3	56850	94
79	KP281	50	Female	16	Partnered	3	3	64809	66
139	KP481	48	Male	16	Partnered	2	3	57987	64
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

Let’s draw the boxplot now,

[65]:
```
sns.boxplot(data = customers['Age'])
plt.title('Age Plot')
plt.show()
```



Insights:
1. 50% age lies between 24 to 33.
2. The median value is 26.

3. The outliers are which are less than 10.5 or greater than 46.5.
4. There are 5 customers in the dataset whose age is more than 46.5.

ii. Education: In Years

In the given dataset the education is lies between 12 to 21 years.

```
[82]: print("Mean/Average: {}".format(customers['Education'].mean()))
      print("Median: {}".format(customers['Education'].median()))
      print("Mode: {}".format(customers['Education'].mode().values[0]))
      print("Standard Deviation: {}".format(customers['Education'].std()))
```

```
Mean/Average: 15.572222222222223
Median: 16.0
Mode: 16
Standard Deviation: 1.6170548978065553
```

The dataset has average education of 15.57 years, median is 16, maximum customers are 16 years old and the standard deviation on customer's education is 1.61 years. The outliers are which are below 1.5IQR or above 1.5IQR.

Let's find the IQR:

```
[84]: print('The first quantile: '+str(customers['Education'].quantile(0.25)))
      print('The second quantile: '+str(customers['Education'].quantile(0.5)))
      print('The third quantile: '+str(customers['Education'].quantile(0.75)))
```

```
The first quantile: 14.0
The second quantile: 16.0
The third quantile: 16.0
```

```
[85]: q1_s= customers['Education'].quantile(0.25)
      q3_s= customers['Education'].quantile(0.75)
      iqr_s = q3_s-q1_s
      print(f'The IQR value is {iqr_s}')
```

```
The IQR value is 2.0
```

The outliers are below or above to the given values respectively,

```
[86]: q1_s-1.5*iqr_s, q3_s+1.5*iqr_s
```

```
[86]: (11.0, 19.0)
```

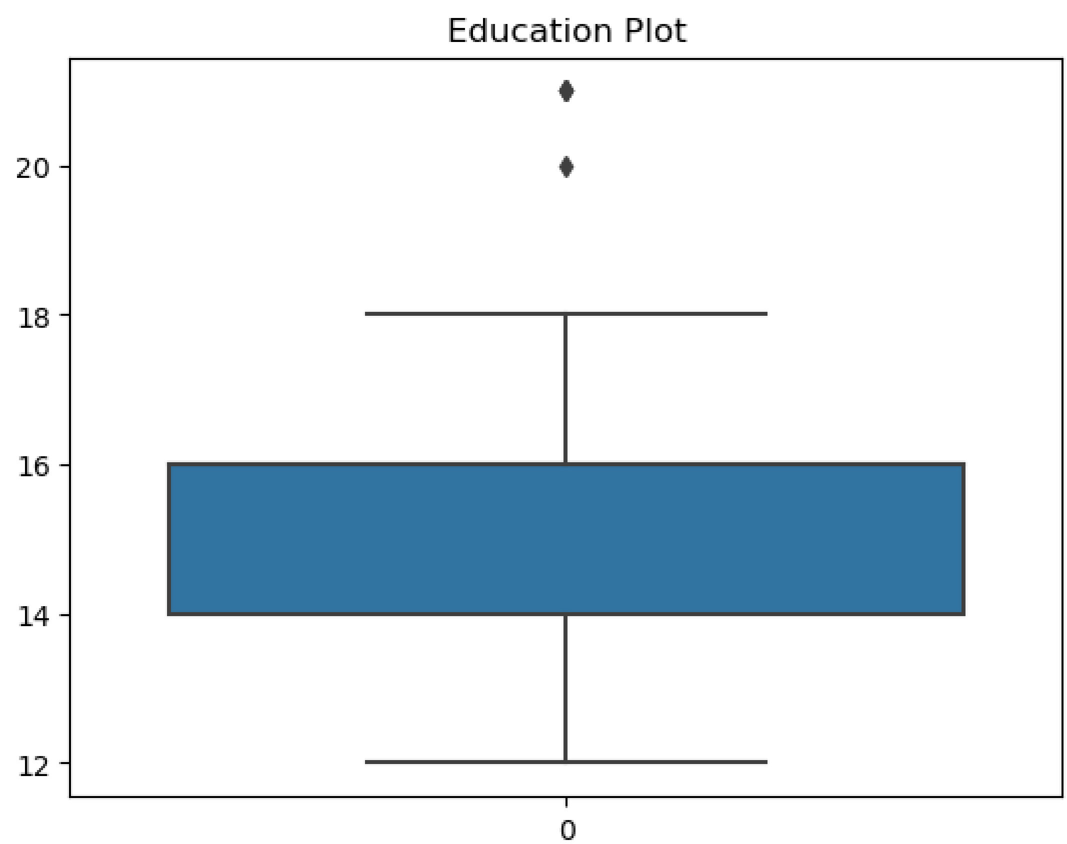
```
[87]: customers[(customers['Education']<(q1_s-1.5*iqr_s)) | (customers['Education']>(q3_s+1.5*iqr_s)) ]
```

[87]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
156	KP781	25	Male	20	Partnered	4	5	74701	170
157	KP781	26	Female	21	Single	4	3	69721	100
161	KP781	27	Male	21	Partnered	4	4	90886	100
175	KP781	40	Male	21	Single	6	5	83416	200

Let's draw the boxplot now,

```
[88]: sns.boxplot(data = customers['Education'])
plt.title('Education Plot')
plt.show()
```



Insights:
1. 50% education years lies between 14 to 16.
2. The median value is 16.
3. The outliers are which are less than 11 or greater than 19.
4. There are 4 customers in the dataset whose education year is more than 19.

iv. Income: Annual income (in $)

In the given dataset the annual income is lies between 29562 to 104581 $.

```
[104]: ## Income: Annual income (in $)
```

```
[107]: print('Minimum income: '+str(customers['Income'].min())+ ' and Maximum income: '+str(customers['Income'].max()))
       print("Mean/Average: {}".format(customers['Income'].mean()))
       print("Median: {}".format(customers['Income'].median()))
       print("Mode: {}".format(customers['Income'].mode().values[0]))
       print("Standard Deviation: {}".format(customers['Income'].std()))

       Minimum income: 29562 and Maximum income: 104581
       Mean/Average: 53719.57777777778
       Median: 50596.5
       Mode: 45480
       Standard Deviation: 16506.68422623862
```

The dataset has average income of 53719.58$, median is 50596.5$, maximum customer earns 45480$ annually and the standard deviation on customer's income is 16506.68$. The outliers are which are below 1.5IQR or above 1.5IQR.

Let's find the IQR:

```
[108]: print('The first quantile: '+str(customers['Income'].quantile(0.25)))
       print('The second quantile: '+str(customers['Income'].quantile(0.5)))
       print('The third quantile: '+str(customers['Income'].quantile(0.75)))

       The first quantile: 44058.75
       The second quantile: 50596.5
       The third quantile: 58668.0
```

```
[109]: q1_s= customers['Income'].quantile(0.25)
       q3_s= customers['Income'].quantile(0.75)
       iqr_s = q3_s-q1_s
       print(f'The IQR value is {iqr_s}')

       The IQR value is 14609.25
```

The outliers are below or above to the given values respectively,

```
[110]: q1_s-1.5*iqr_s, q3_s+1.5*iqr_s

[110]: (22144.875, 80581.875)
```

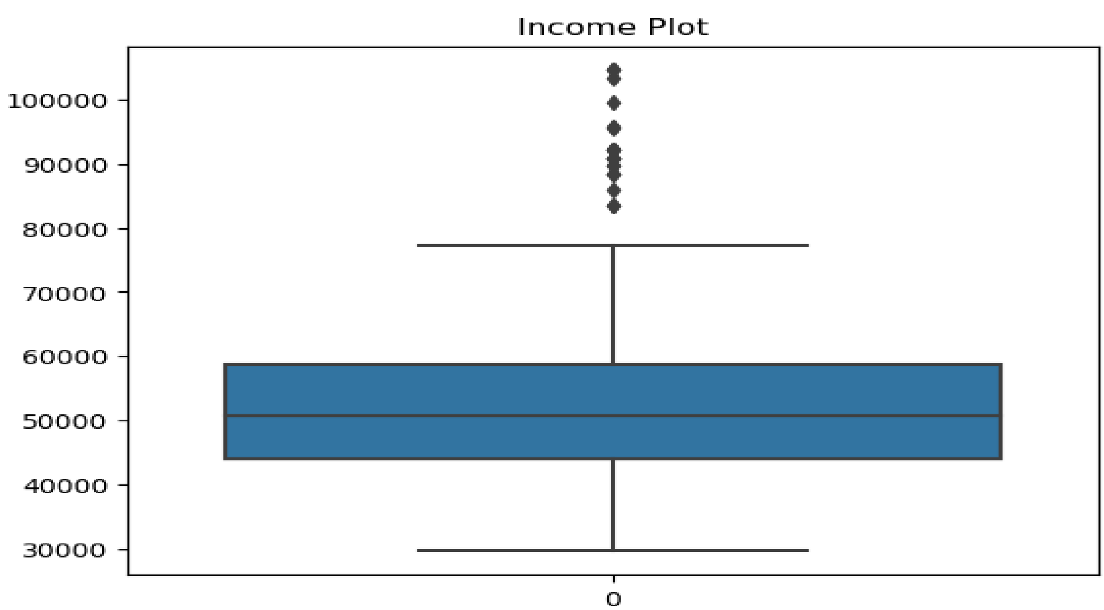
```
[111]: customers[(customers['Income']<(q1_s-1.5*iqr_s)) | (customers['Income']>(q3_s+1.5*iqr_s)) ]
```

[111]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
159	KP781	27	Male	16	Partnered	4	5	83416	160
160	KP781	27	Male	18	Single	4	3	88396	100
161	KP781	27	Male	21	Partnered	4	4	90886	100
162	KP781	28	Female	18	Partnered	6	5	92131	180
164	KP781	28	Male	18	Single	6	5	88396	150
166	KP781	29	Male	14	Partnered	7	5	85906	300
167	KP781	30	Female	16	Partnered	6	5	90886	280
168	KP781	30	Male	18	Partnered	5	4	103336	160
169	KP781	30	Male	18	Partnered	5	5	99601	150
170	KP781	31	Male	16	Partnered	6	5	89641	260
171	KP781	33	Female	18	Partnered	4	5	95866	200
172	KP781	34	Male	16	Single	5	5	92131	150
173	KP781	35	Male	16	Partnered	4	5	92131	360
174	KP781	38	Male	18	Partnered	5	5	104581	150
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

Let’s draw the boxplot now,

```
[112]: sns.boxplot(data = customers['Income'])
plt.title('Income Plot')
plt.show()
```



Insights:

1. 50% customer lies between 44058.75$ to 58668$ annually.
2. The median value is 50596.5$.
3. The outliers are which are less than 22144.875$ or greater than 80581.875$.
4. There are 19 customers in the dataset whose annual income greater than 80581.875$.

v. Miles: The average number of miles the customer expects to walk/run each week

In the given dataset the weekly miles are lies between 21 to 360.

```
[117]: ## Miles: The average number of miles the customer expects to walk/run each week
```

```
[118]: print('Minimum income: '+str(customers['Miles'].min())+ ' and Maximum income: '+str(customers['Miles'].max()))
       print("Mean/Average: {}".format(customers['Miles'].mean()))
       print("Median: {}".format(customers['Miles'].median()))
       print("Mode: {}".format(customers['Miles'].mode().values[0]))
       print("Standard Deviation: {}".format(customers['Miles'].std()))

       Minimum income: 21 and Maximum income: 360
       Mean/Average: 103.19444444444444
       Median: 94.0
       Mode: 85
       Standard Deviation: 51.86360466180934
```

The dataset has average miles of 103.19, median is 94, maximum customer achieves miles of 85 weekly and the standard deviation on customer's miles 51.86. The outliers are which are below 1.5IQR or above 1.5IQR.

Let's find the IQR:

```
[119]: print('The first quantile: '+str(customers['Miles'].quantile(0.25)))
       print('The second quantile: '+str(customers['Miles'].quantile(0.5)))
       print('The third quantile: '+str(customers['Miles'].quantile(0.75)))

       The first quantile: 66.0
       The second quantile: 94.0
       The third quantile: 114.75
```

```
[120]: q1_s= customers['Miles'].quantile(0.25)
       q3_s= customers['Miles'].quantile(0.75)
       iqr_s = q3_s-q1_s
       print(f'The IQR value is {iqr_s}')

       The IQR value is 48.75
```

The outliers are below or above to the given values respectively,

```
[121]: q1_s-1.5*iqr_s, q3_s+1.5*iqr_s

[121]: (-7.125, 187.875)
```

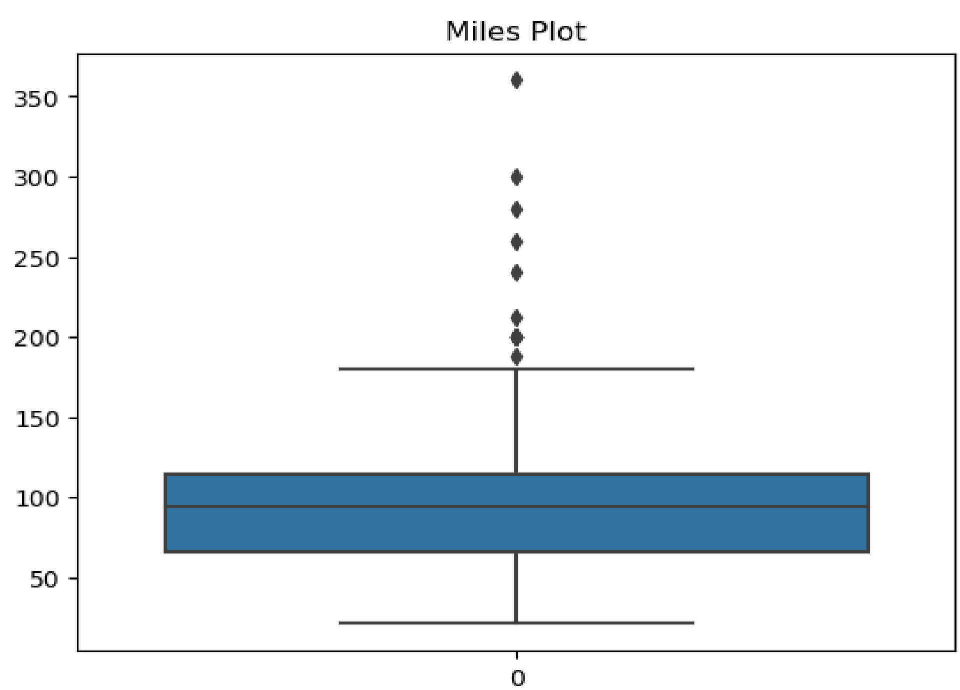
[122]:
```
customers[(customers['Miles']<(q1_s-1.5*iqr_s)) | (customers['Miles']>(q3_s+1.5*iqr_s)) ]
```

[122]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
23	KP281	24	Female	16	Partnered	5	5	44343	188
84	KP481	21	Female	14	Partnered	5	4	34110	212
142	KP781	22	Male	18	Single	4	5	48556	200
148	KP781	24	Female	16	Single	5	5	52291	200
152	KP781	25	Female	18	Partnered	5	5	61006	200
155	KP781	25	Male	18	Partnered	6	5	75946	240
166	KP781	29	Male	14	Partnered	7	5	85906	300
167	KP781	30	Female	16	Partnered	6	5	90886	280
170	KP781	31	Male	16	Partnered	6	5	89641	260
171	KP781	33	Female	18	Partnered	4	5	95866	200
173	KP781	35	Male	16	Partnered	4	5	92131	360
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200

Let's draw the boxplot now,

[123]:
```
sns.boxplot(data = customers['Miles'])
plt.title('Miles Plot')
plt.show()
```



Insights:

1. 50% customer lies between 66 to 114.75 weekly.
2. The median value is 94.
3. The outliers are which are less than -7.125 or greater than 187.875.
4. There are 13 customers in the dataset whose weekly miles is greater than 187.875.

3. Check if features like marital status, Gender, and age have any effect on the product purchased:

3.1 Univariate analysis:

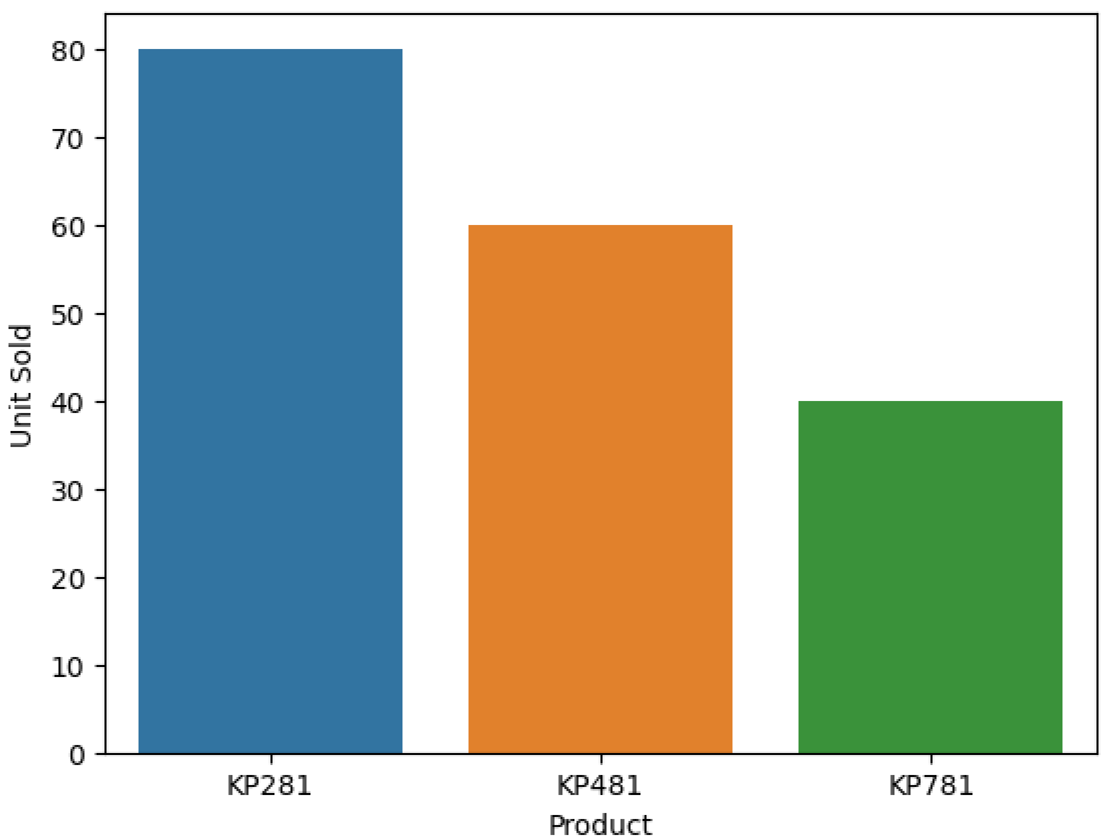
i. Categorical Variable:

a. Product Sales distribution:

Let's create a count plot between product and Unit sold,

[72]:
```
## Product Sales Distribution:
```

[78]:
```
sns.countplot(data = customers, x = 'Product')
plt.ylabel('Unit Sold')
plt.show()
```



```
[76]: customers['Product'].value_counts()
```

```
[76]: Product
KP281    80
KP481    60
KP781    40
Name: count, dtype: int64
```

The revenue generated by each type as below,

```
[82]: def price(x):
          if x['Product'] == 'KP281':
              return x['count']*1500
          elif x['Product'] == 'KP481':
              return x['count']*1750
          else:
              return x['count']*2500
```

```
[86]: sales['Total Price'] = sales.apply(price, axis = 1)
```

```
[94]: sales
```

[94]:

	Product	count	Unit Price	Total Price
0	KP281	80	1500	120000
1	KP481	60	1750	105000
2	KP781	40	2500	100000

Insights:

1. From above we can see that KP281 model is high in terms of quantity sold, next is KP481 and last is KP781.
2. By judging the unit price given we can observe that KP281 is a base model treadmill by AeroFit and KP781 is the top model AeroFit offer to their customer.
3. While observing the revenue generation all three models generates same kind of revenue though KP281 generate highest amount of revenue.

b. Gender and Marital Status distribution over product sold:

```
[101]: customers['Gender'].value_counts()
```

```
[101]: Gender
Male      104
Female     76
Name: count, dtype: int64
```

```
[102]: customers['MaritalStatus'].value_counts()
```

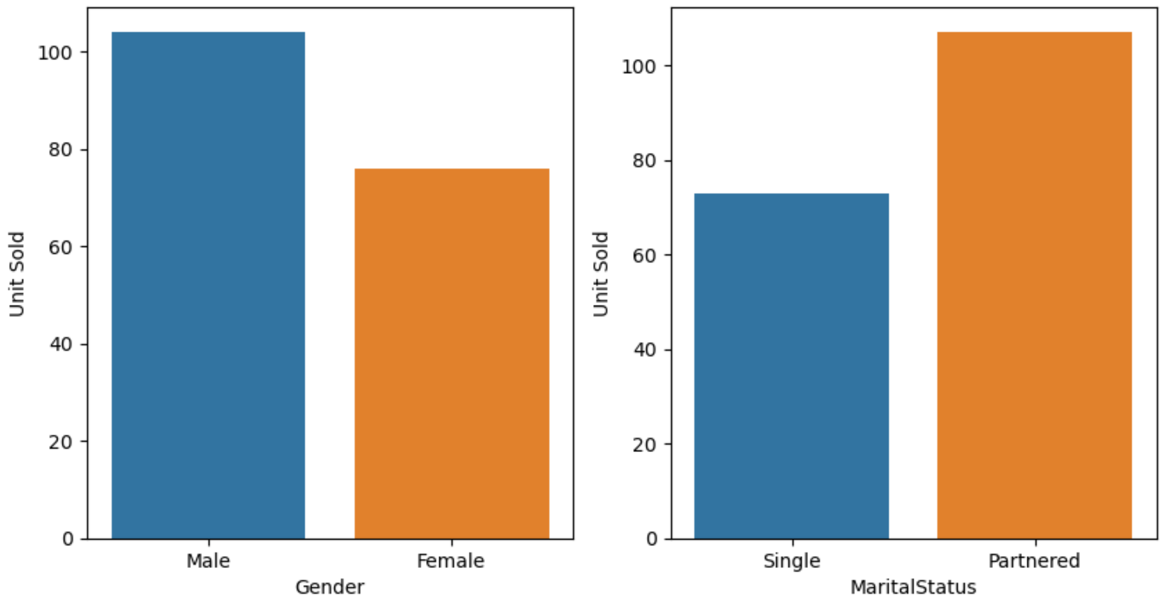
```
[102]: MaritalStatus
Partnered    107
Single        73
Name: count, dtype: int64
```

Let’s plot the graphs between gender and unit sold also for maritalstatus and unit sold.

[100]:
```
## Sales upon gender and marital status
```

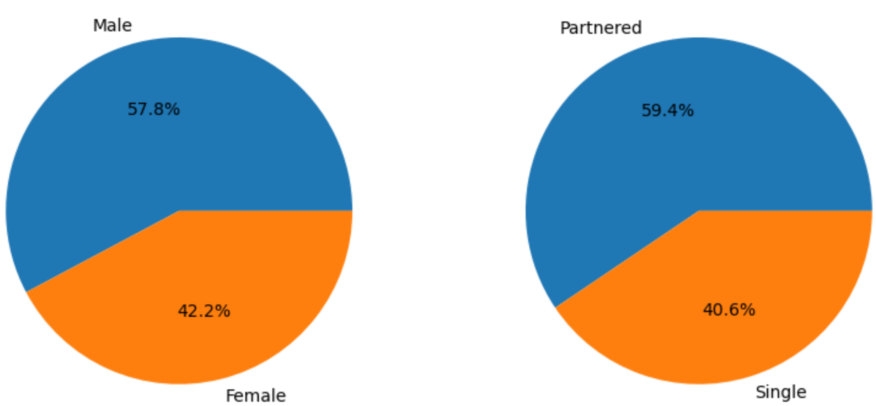
[99]:
```
plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(data = customers, x = 'Gender')
plt.ylabel('Unit Sold')

plt.subplot(1,2,2)
sns.countplot(data = customers, x = 'MaritalStatus')
plt.ylabel('Unit Sold')
```

[99]:
```
Text(0, 0.5, 'Unit Sold')
```



If we want to see the proportion then,

[107]:
```
plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
plt.pie(customers['Gender'].value_counts().values,labels = customers['Gender'].value_counts().index, autopct='%1.1f%%')

plt.subplot(1,2,2)
plt.pie(customers['MaritalStatus'].value_counts().values,labels = customers['MaritalStatus'].value_counts().index, autopct='%1.1f%%')

plt.show()
```



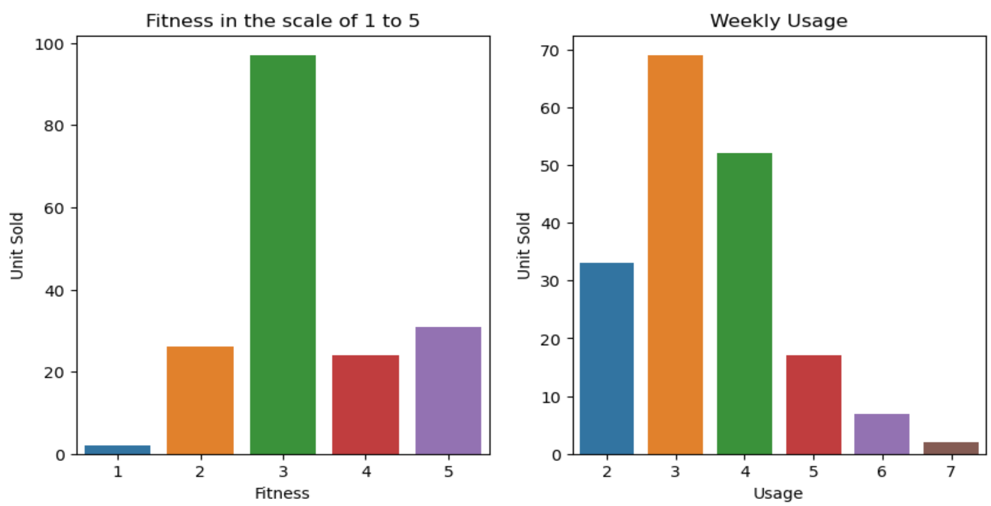
Insights:

1. We can see males are more concerned about their fitness compared to females.
2. Also we can see that couples are more concerned about their fitness compared to singles.

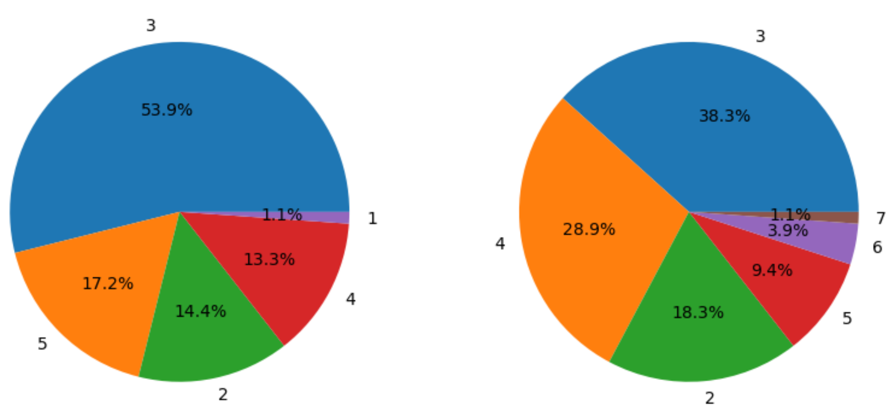
c. Buyer’s fitness and treadmill usage vs product sold:

[108]:
```
## Fitness and usage
```

[112]:
```
plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
sns.countplot(data = customers, x = 'Fitness')
plt.ylabel('Unit Sold')
plt.title('Fitness in the scale of 1 to 5')

plt.subplot(1,2,2)
sns.countplot(data = customers, x = 'Usage')
plt.ylabel('Unit Sold')
plt.title('Weekly Usage')

plt.show()
```



[113]:
```
plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
plt.pie(customers['Fitness'].value_counts().values,labels = customers['Fitness'].value_counts().index, autopct='%1.1f%%')

plt.subplot(1,2,2)
plt.pie(customers['Usage'].value_counts().values,labels = customers['Usage'].value_counts().index, autopct='%1.1f%%')

plt.show()
```



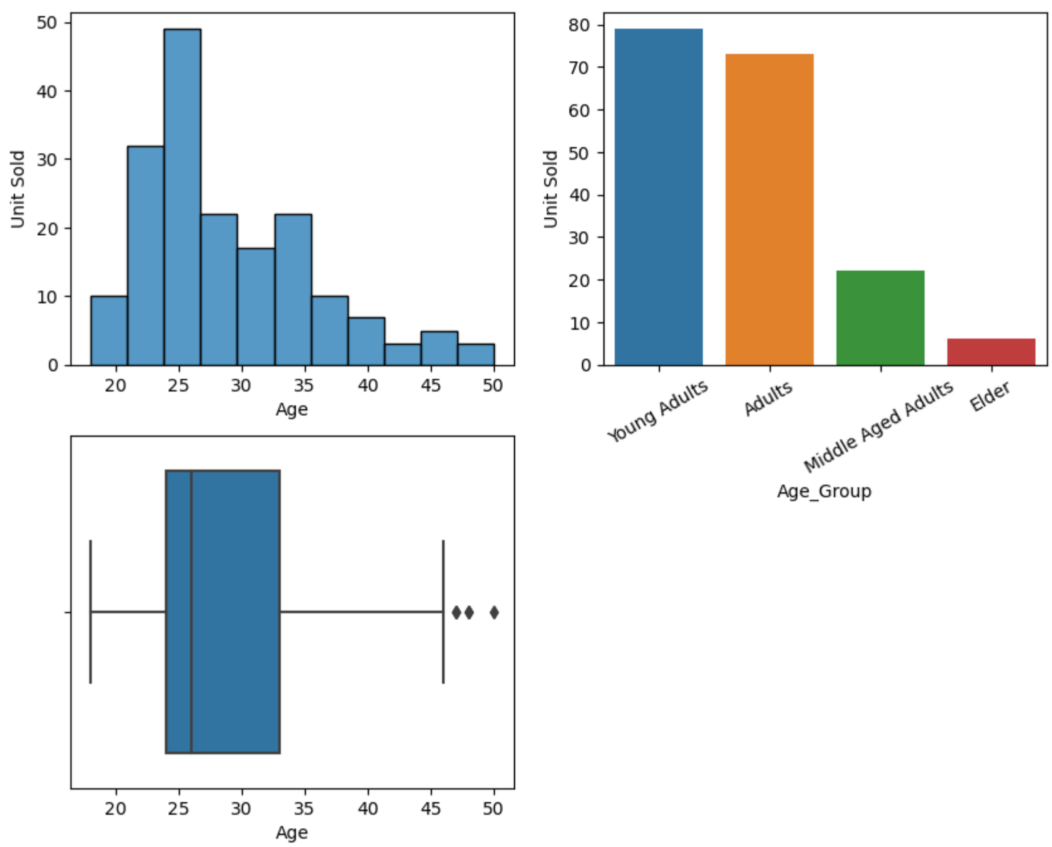
Insights:

1. 53.9% customers rated themselves as 3 for their fitness in the scale of 1 to 5. 17.2% people claimed as super fit customers. Only 1.1% people claimed themselves and under fit.
2. 18.3+38.3+28.9 = 85% people use the treadmill 2 to 4 times in a week whereas only 15% people use more than 4 times.

ii. Numerical Variable:

a. Customer age distribution:

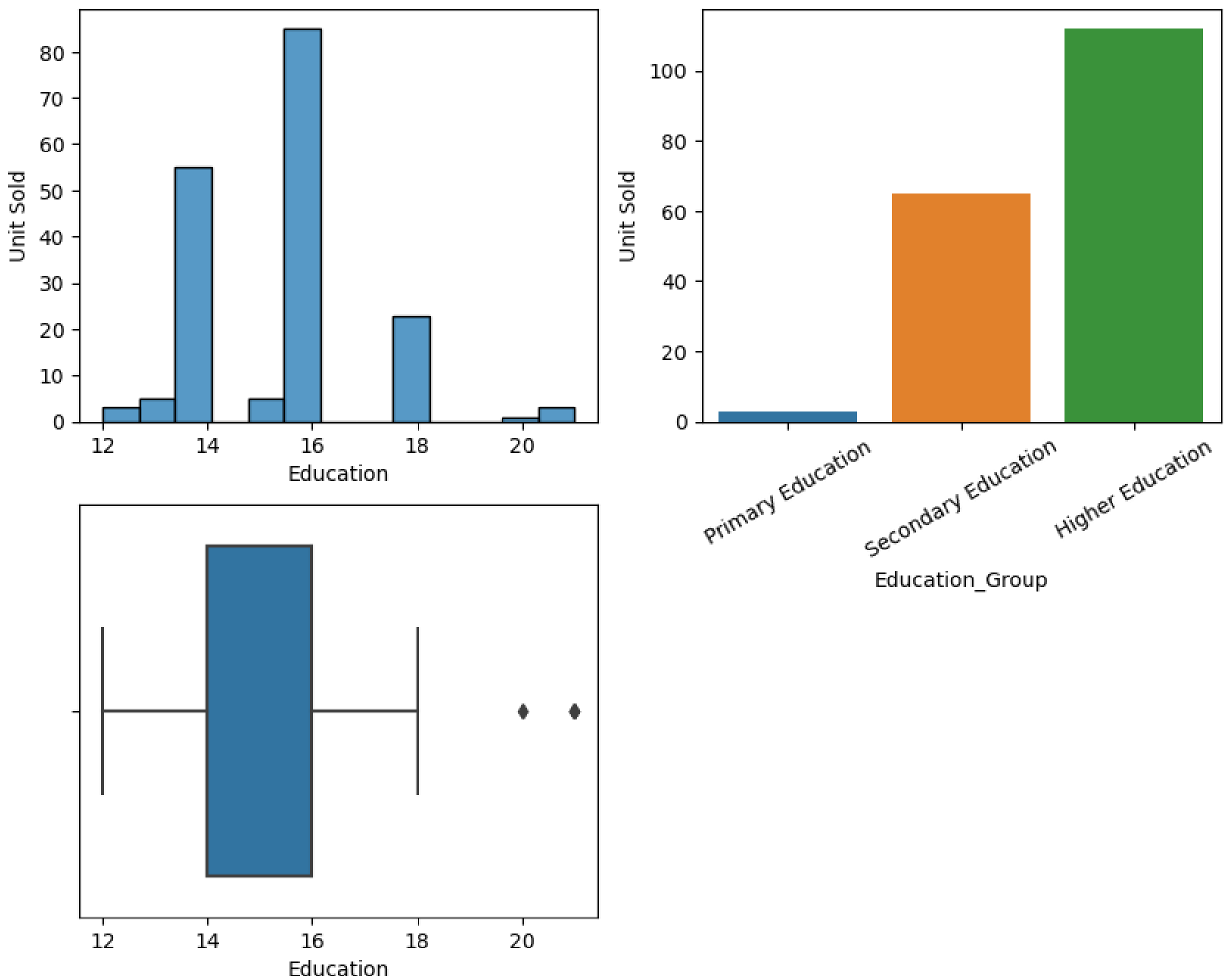
[281]:
```
plt.figure(figsize = (10,8))
plt.subplot(2,2,1)
sns.histplot(data = customers, x = 'Age')
plt.ylabel('Unit Sold')

plt.subplot(2,2,2)
sns.countplot(data = customers, x = 'Age_Group')
plt.ylabel('Unit Sold')
plt.xticks(rotation = 30)

plt.subplot(2,2,3)
sns.boxplot(data = customers, x = 'Age')

plt.show()
```



Insights:
Age group between 20 to 35 are the customers who purchased maximum treadmills.

b. Customer Education Distribution

```
[283]: plt.figure(figsize = (10,8))
plt.subplot(2,2,1)
sns.histplot(data = customers, x = 'Education')
plt.ylabel('Unit Sold')

plt.subplot(2,2,2)
sns.countplot(data = customers, x = 'Education_Group')
plt.ylabel('Unit Sold')
plt.xticks(rotation = 30)

plt.subplot(2,2,3)
sns.boxplot(data = customers, x = 'Education')

plt.show()
```

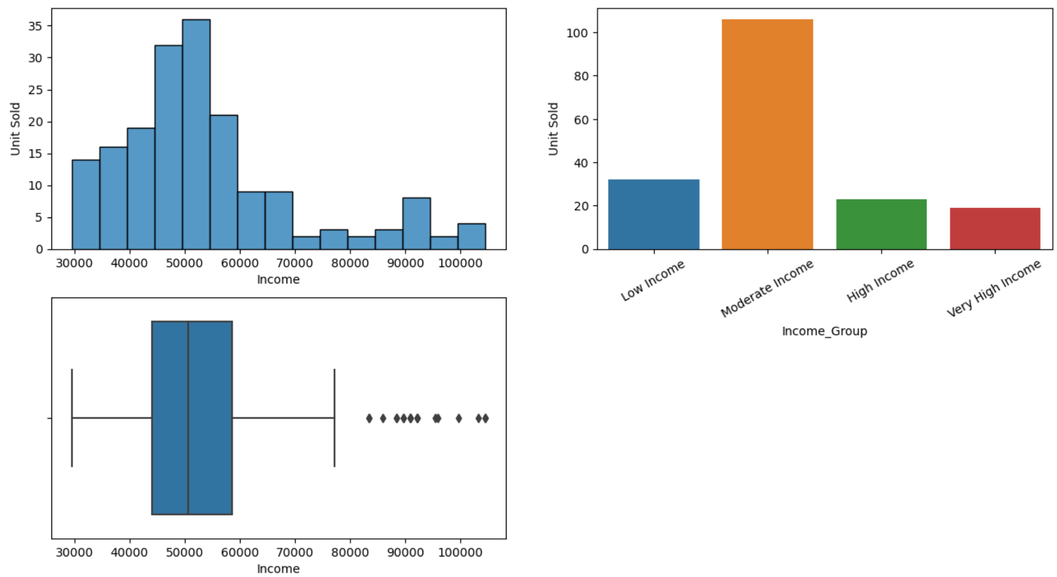


Insights:

98% of customers who have education more than 12 years are the customers who purchased maximum treadmills.

c. Income Distribution over sales:

```
[284]: plt.figure(figsize = (15,8))
plt.subplot(2,2,1)
sns.histplot(data = customers, x = 'Income')
plt.ylabel('Unit Sold')

plt.subplot(2,2,2)
sns.countplot(data = customers, x = 'Income_Group')
plt.ylabel('Unit Sold')
plt.xticks(rotation = 30)
plt.subplot(2,2,3)
sns.boxplot(data = customers, x = 'Income')


plt.show()
```

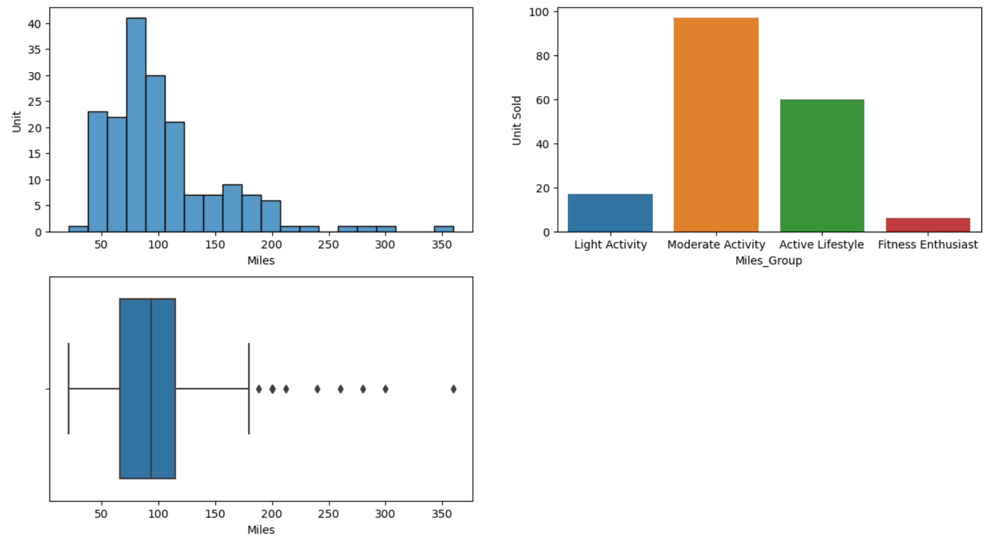


Insights:

Maximum customers who lies in the range of 40k to 60k are responsible for high sales of treadmill.

d. Customer weekly expected mileage distribution:

```
[287]: plt.figure(figsize = (15,8))
plt.subplot(2,2,1)
sns.histplot(data = customers, x = 'Miles')
plt.ylabel('Unit')

plt.subplot(2,2,2)
sns.countplot(data = customers, x = 'Miles_Group')
plt.ylabel('Unit Sold')

plt.subplot(2,2,3)
sns.boxplot(data = customers, x = 'Miles')

plt.show()
```



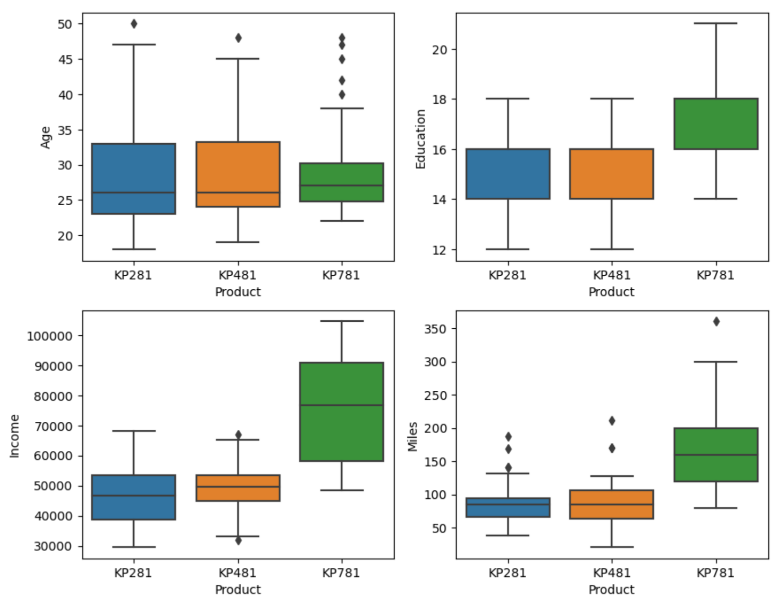
Insights:

Maximum customers bought the treadmills to use it for 50 to 200 miles weekly.

3.2 Bivariate analysis:

i. Product vs Numerical Variables:

```
plt.figure(figsize = (10, 8))
plt.subplot(2,2,1)
sns.boxplot(data = customers, x= 'Product', y='Age')

plt.subplot(2,2,2)
sns.boxplot(data = customers, x= 'Product', y='Education')

plt.subplot(2,2,3)
sns.boxplot(data = customers, x= 'Product', y='Income')

plt.subplot(2,2,4)
sns.boxplot(data = customers, x= 'Product', y='Miles')

plt.show()
```



Insights:

The above graphs depict that: KP781 the advanced treadmill is used by age group 25 to 30 also these customers are highly educated, have high income and used for high activity as they use it for 150 to 200 miles weekly.

ii. Product preference over different ages:

[294]:
```
grp_p = customers.groupby('Product')
df = grp_p['Age_Group'].value_counts(normalize = True).reset_index()
df_new = df.pivot(index = 'Product', columns = 'Age_Group', values = 'proportion')
df_final = df_new.reset_index()
```

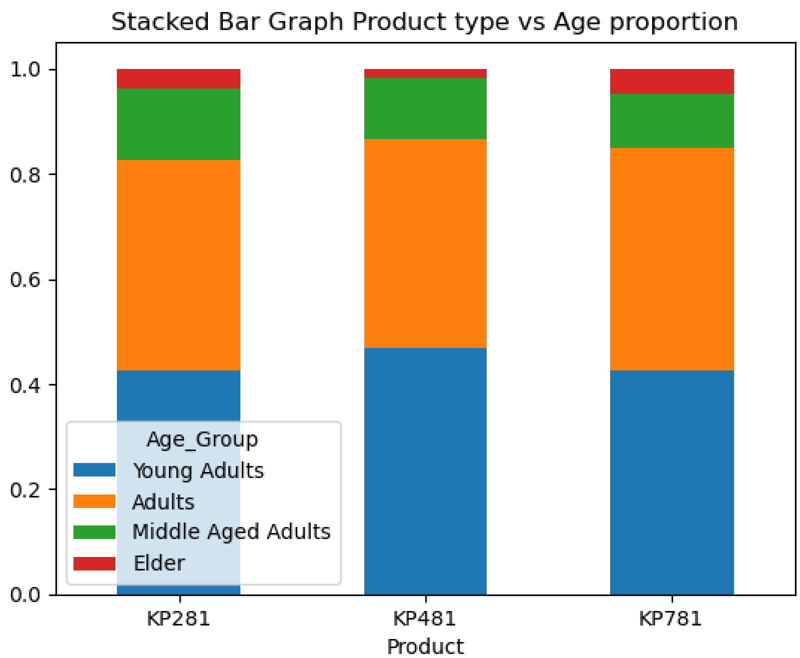
[295]:
```
df_final
```

[295]:

Age_Group	Product	Young Adults	Adults	Middle Aged Adults	Elder
0	KP281	0.425000	0.400	0.137500	0.037500
1	KP481	0.466667	0.400	0.116667	0.016667
2	KP781	0.425000	0.425	0.100000	0.050000

[297]:
```
df_final.plot(x='Product', kind='bar', stacked=True,
        title='Stacked Bar Graph Product type vs Age proportion')
plt.xticks(rotation = 0)
```

[297]:
```
(array([0, 1, 2]),
 [Text(0, 0, 'KP281'), Text(1, 0, 'KP481'), Text(2, 0, 'KP781')])
```



Insights:

From the above analysis we can conclude that there is no strong correlation between age and product type. This is evident from the nearly uniform distribution of age groups across all the product types.

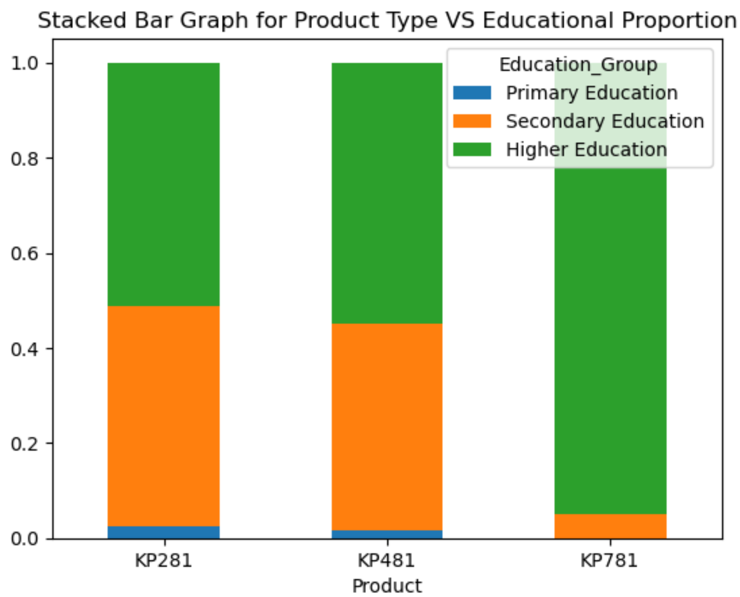
iii. Product type vs Educational group:

[298]:
```
grp_p = customers.groupby('Product')
df = grp_p['Education_Group'].value_counts(normalize = True).reset_index()
df_new = df.pivot(index = 'Product', columns = 'Education_Group', values = 'proportion')
df_final = df_new.reset_index()
df_final
```

[298]:

Education_Group	Product	Primary Education	Secondary Education	Higher Education
0	KP281	0.025000	0.462500	0.5125
1	KP481	0.016667	0.433333	0.5500
2	KP781	0.000000	0.050000	0.9500

[299]:
```
df_final.plot(x='Product', kind='bar', stacked=True,
        title='Stacked Bar Graph for Product Type VS Educational Proportion')
plt.xticks(rotation = 0)
plt.show()
```



Insights:

1. KP781 is popular among highly educated customers among educational experience of 15 to 24 years.
2. For KP281 and KP481 shows similar kind of distribution for Secondary and Higher Education.

iv. Product type vs income group:

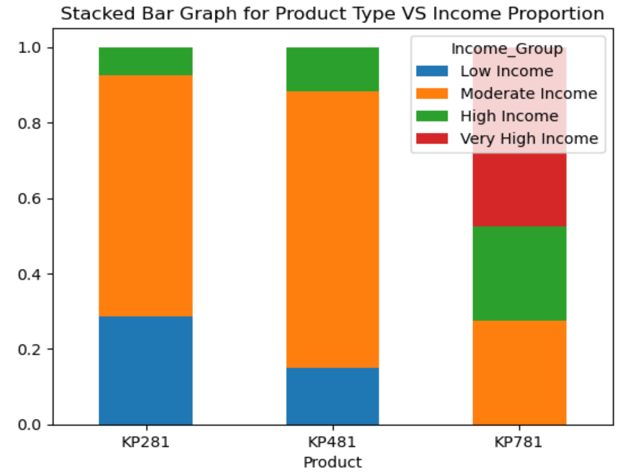
[300]:
```
## Product type vs income group
```

[301]:
```
grp_p = customers.groupby('Product')
df = grp_p['Income_Group'].value_counts(normalize = True).reset_index()
df_new = df.pivot(index = 'Product', columns = 'Income_Group', values = 'proportion')
df_final = df_new.reset_index()
df_final
```

[301]:

Income_Group	Product	Low Income	Moderate Income	High Income	Very High Income
0	KP281	0.2875	0.637500	0.075000	0.000
1	KP481	0.1500	0.733333	0.116667	0.000
2	KP781	0.0000	0.275000	0.250000	0.475

[302]:
```python
df_final.plot(x='Product', kind='bar', stacked=True,
        title='Stacked Bar Graph for Product Type VS Income Proportion')
plt.xticks(rotation = 0)
plt.show()
```



Insights:

1. From the above analysis we can say that very high income group of customers only can afford the KP781 the advance model.
2. Moderate income groups of customer prefer both KP281 and KP481.

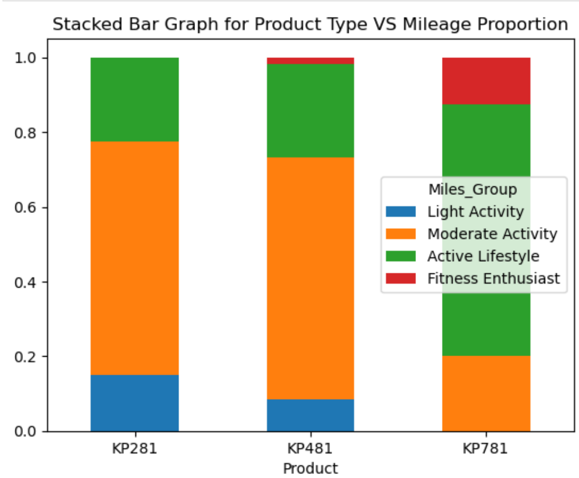
v. Product type vs customer weekly mileage:

[303]:
```python
## Product type vs mileage
grp_p = customers.groupby('Product')
df = grp_p['Miles_Group'].value_counts(normalize = True).reset_index()
df_new = df.pivot(index = 'Product', columns = 'Miles_Group', values = 'proportion')
df_final = df_new.reset_index()
df_final
```

[303]:

Miles_Group	Product	Light Activity	Moderate Activity	Active Lifestyle	Fitness Enthusiast
0	KP281	0.150000	0.625	0.225	0.000000
1	KP481	0.083333	0.650	0.250	0.016667
2	KP781	0.000000	0.200	0.675	0.125000

[304]:
```python
df_final.plot(x='Product', kind='bar', stacked=True,
        title='Stacked Bar Graph for Product Type VS Mileage Proportion')
plt.xticks(rotation = 0)
plt.show()
```

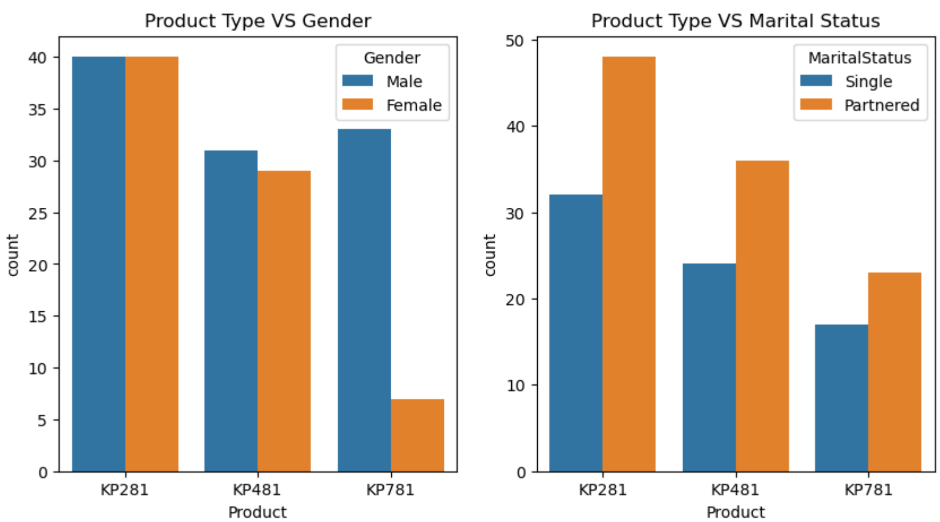


Insights:

1. KP281 and KP481 mainly used for users who moderate activity like to run 50 to 100 miles weekly.
2. KP781 is basically used for users who maintain active life style like to run 100 to 200 miles weekly.

vi. Product type vs Gender and Marital Status:

```
[312]: # Product type vs Gender and Marital Status
plt.figure(figsize = (10, 5))
plt.subplot(1,2,1)
sns.countplot(data = customers, x = 'Product', hue = 'Gender')
plt.title('Product Type VS Gender')

plt.subplot(1,2,2)
sns.countplot(data = customers, x = 'Product', hue = 'MaritalStatus')
plt.title('Product Type VS Marital Status')
plt.show()
```

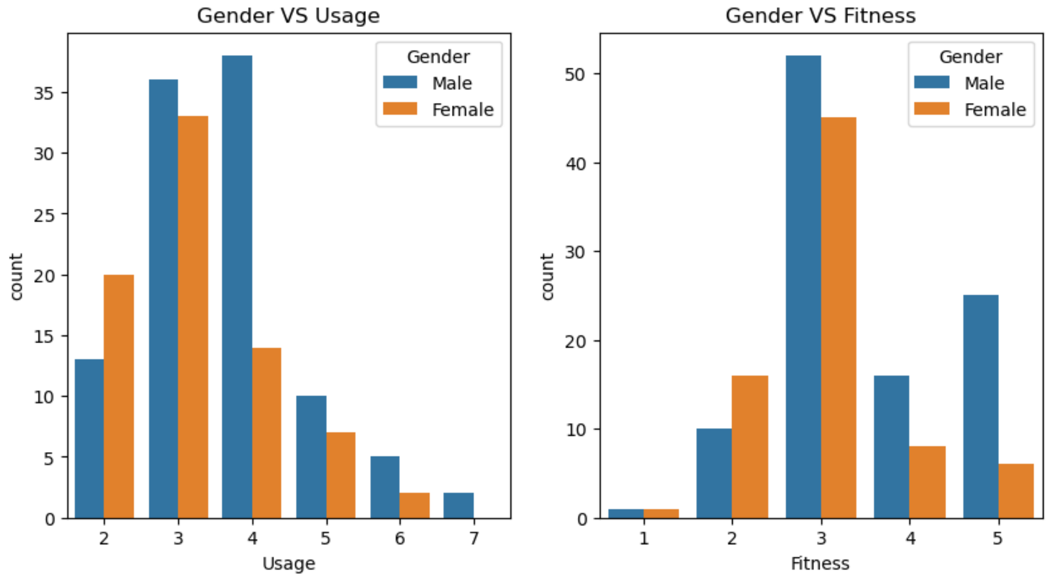


Insights:

1. For KP281 and KP481 the ratio of male and female is almost 50:50.
2. For KP781 the male customers have a higher proportion.
3. For all three type of treadmill the marital status is showing a uniform distribution, where partnered have slightly higher value than single customers.

vii. Gender VS Product Usage and Gender VS Fitness:

```
[314]: ##Gender VS Product Usage and Gender VS Fitness
plt.figure(figsize = (10, 5))
plt.subplot(1,2,1)
sns.countplot(data = customers, x = 'Usage', hue = 'Gender')
plt.title('Gender VS Usage')

plt.subplot(1,2,2)
sns.countplot(data = customers, x = 'Fitness', hue = 'Gender')
plt.title('Gender VS Fitness')
plt.show()
```



Insights:

1. Maximum male customers prefers to use treadmills 3 to 4 times weekly and female customers prefers to use the treadmill 2 to 3 time weekly.
2. 90% male customers rated themselves 3 to 5 in fitness while 80% female customers rated themselves 2 to 3 in fitness.

4. Representing the Probability:

i. Find the marginal probability (what percent of customers have purchased KP281, KP481, or KP781)

a. Probability of product purchase with respect to Gender:

```
[324]: pd.crosstab(index = customers['Product'], columns = customers['Gender'], margins = True, normalize = True).round(2)
```

[324]:

Gender Product	Female	Male	All
KP281	0.22	0.22	0.44
KP481	0.16	0.17	0.33
KP781	0.04	0.18	0.22
All	0.42	0.58	1.00

Insights:

1. The probability of a product purchased by Female is 42%
    - The conditional probability of purchasing a model given that the gender is Female
        - Product is KP281: 22%
        - Product is KP481: 16%
        - Product is KP781: 4%
2. The probability of a product purchased by male is 58%

- The conditional probability of purchasing a model given that the gender is Male
    - Product is KP281: 22%
    - Product is KP481: 17%
    - Product is KP781: 18%

b. Probability of product purchase with respect to age:

```
[326]: ## Probability of product purchase with respect to age
pd.crosstab(index = customers['Product'], columns = customers['Age_Group'], margins = True, normalize = True).round(2)
```

[326]:

Age_Group Product	Young Adults	Adults	Middle Aged Adults	Elder	All
KP281	0.19	0.18	0.06	0.02	0.44
KP481	0.16	0.13	0.04	0.01	0.33
KP781	0.09	0.09	0.02	0.01	0.22
All	0.44	0.41	0.12	0.03	1.00

Insights:

1. The probability of purchasing a product for Young Adult (18 to 25) is 44%
    - The conditional probability of purchasing a model given that the customer is Young Adult
        - For KP281: 19%
        - For KP481: 16%
        - For KP781: 9%
2. The probability of purchasing a product for Adults (26 to 35) is 41%
    - The conditional probability of purchasing a model given that the customer is adult
        - For KP281: 18%
        - For KP481: 16%
        - For KP781: 9%
3. The probability of purchasing a product for Middle Aged (36 to 45) is 12%
4. The probability of purchasing a product for Elder (Above 45) is 3%

c. Probability of purchasing a product with respect to education:

```
[327]: ## Probability of product purchase with respect to education
pd.crosstab(index = customers['Product'], columns = customers['Education_Group'], margins = True, normalize = True).round(2)
```

[327]:

Education_Group Product	Primary Education	Secondary Education	Higher Education	All
KP281	0.01	0.21	0.23	0.44
KP481	0.01	0.14	0.18	0.33
KP781	0.00	0.01	0.21	0.22
All	0.02	0.36	0.62	1.00

Insights:

1. The probability of purchasing a product for primary educated (0 to 12 years) customer is 2%.
2. The probability of purchasing a product for secondary educated (13 to 15 years) customer is 36%

- The conditional probability of purchasing a model given that the customer is secondary educated
    - For KP281: 21%
    - For KP481: 14%
    - For KP781: 1%

3. The probability of purchasing a product for highly educated (above 15 years) customer is 62%
    - The conditional probability of purchasing a model given that the customer is highly educated
        - For KP281: 23%
        - For KP481: 18%
        - For KP781: 21%

d. Probability of purchasing a product with respect to income:

```
[328]: ## Probability of product purchase with respect to income
pd.crosstab(index = customers['Product'], columns = customers['Income_Group'], margins = True, normalize = True).round(2)
```

[328]:

Income_Group Product	Low Income	Moderate Income	High Income	Very High Income	All
KP281	0.13	0.28	0.03	0.00	0.44
KP481	0.05	0.24	0.04	0.00	0.33
KP781	0.00	0.06	0.06	0.11	0.22
All	0.18	0.59	0.13	0.11	1.00

Insights:

1. The probability of purchasing a product for low income (<40k) customer is 18%.
    - The conditional probability of purchasing a model given that the customer is having low income
        - For KP281: 13%
        - For KP481: 5%
        - For KP781: 0%
2. The probability of purchasing a product for moderate income (40k – 60k) customer is 59%.
    - The conditional probability of purchasing a model given that the customer is having moderate income
        - For KP281: 28%
        - For KP481: 24%
        - For KP781: 6%
3. The probability of purchasing a product for moderate income (60k – 80k) customer is 13%.
    - The conditional probability of purchasing a model given that the customer is having high income
        - For KP281: 3%
        - For KP481: 4%
        - For KP781: 6%
4. The probability of purchasing a product for moderate income (above 80k) customer is 11%.
    - The conditional probability of purchasing a model given that the customer is having very high income
        - For KP281: 0%

- For KP481: 0%
- For KP781: 11%

e. Probability of purchasing a product with respect to marital status:

```
[329]: ## Probability of product purchase with respect to marital status
pd.crosstab(index = customers['Product'], columns = customers['MaritalStatus'], margins = True, normalize = True).round(2)
```

[329]:

MaritalStatus / Product	Partnered	Single	All
KP281	0.27	0.18	0.44
KP481	0.20	0.13	0.33
KP781	0.13	0.09	0.22
All	0.59	0.41	1.00

Insights:

1. The probability of purchasing a product for single customer is 41%.
   - The conditional probability of purchasing a model given that the customer is single
     - For KP281: 18%
     - For KP481: 13%
     - For KP781: 5%
2. The probability of purchasing a product for partnered customer is 59%.
   - The conditional probability of purchasing a model given that the customer is partnered
     - For KP281: 27%
     - For KP481: 20%
     - For KP781: 13%

f. Probability of purchasing a product with respect to weekly usage:

```
[330]: ## Probability of product purchase with respect to weekly usage
pd.crosstab(index = customers['Product'], columns = customers['Usage'], margins = True, normalize = True).round(2)
```

[330]:

Usage / Product	2	3	4	5	6	7	All
KP281	0.11	0.21	0.12	0.01	0.00	0.00	0.44
KP481	0.08	0.17	0.07	0.02	0.00	0.00	0.33
KP781	0.00	0.01	0.10	0.07	0.04	0.01	0.22
All	0.18	0.38	0.29	0.09	0.04	0.01	1.00

Insights:

1. The probability of purchasing a product for 2 times usage customer is 18%.
   - The conditional probability of purchasing a model given that the customer is 2 times user
     - For KP281: 11%
     - For KP481: 8%
     - For KP781: 0%
2. The probability of purchasing a product for 3 times usage customer is 38%.

- The conditional probability of purchasing a model given that the customer is 3-time user
    - For KP281: 21%
    - For KP481: 17%
    - For KP781: 1%

3. The probability of purchasing a product for 4 times usage customer is 29%.
    - The conditional probability of purchasing a model given that the customer is 4-time user
        - For KP281: 12%
        - For KP481: 7%
        - For KP781: 10%
4. The probability of purchasing a product for 5 times usage customer is 9%.
5. The probability of purchasing a product for 6 times usage customer is 4%.
6. The probability of purchasing a product for 7 times usage customer is 1%.

g. Probability of purchasing a product with respect to fitness:

```
[331]: # Probability of purchasing a product with respect to fitness
pd.crosstab(index = customers['Product'], columns = customers['Fitness'], margins = True, normalize = True).round(2)
```

[331]:

Fitness / Product	1	2	3	4	5	All
KP281	0.01	0.08	0.30	0.05	0.01	0.44
KP481	0.01	0.07	0.22	0.04	0.00	0.33
KP781	0.00	0.00	0.02	0.04	0.16	0.22
All	0.01	0.14	0.54	0.13	0.17	1.00

Insights:

1. The probability of purchasing a product having fitness rating 3 is 54%.
    - The conditional probability of purchasing a model given that the customer is having 3 fitness rating
        - For KP281: 30%
        - For KP481: 22%
        - For KP781: 2%
2. The probability of purchasing a product having fitness rating 4 is 13%.
    - The conditional probability of purchasing a model given that the customer is having 4 fitness rating
        - For KP281: 5%
        - For KP481: 4%
        - For KP781: 4%
3. The probability of purchasing a product having fitness rating 5 is 17%.
    - The conditional probability of purchasing a model given that the customer is having 5 fitness rating
        - For KP281: 1%
        - For KP481: 0%
        - For KP781: 16%
4. The probability of purchasing a product having fitness rating 2 is 14%.

- The conditional probability of purchasing a model given that the customer is having 2 fitness rating
  - For KP281: 8%
  - For KP481: 7%
  - For KP781: 0%

5. The probability of purchasing a product having fitness rating 1 is 1%.

h. Probability of purchasing a product with respect to weekly mileage:

```
[332]: #Probability of purchasing a product with respect to weekly mileage
pd.crosstab(index = customers['Product'], columns = customers['Miles_Group'], margins = True, normalize = True).round(2)
```

[332]:

Miles_Group Product	Light Activity	Moderate Activity	Active Lifestyle	Fitness Enthusiast	All
KP281	0.07	0.28	0.10	0.00	0.44
KP481	0.03	0.22	0.08	0.01	0.33
KP781	0.00	0.04	0.15	0.03	0.22
All	0.09	0.54	0.33	0.03	1.00

Insights:

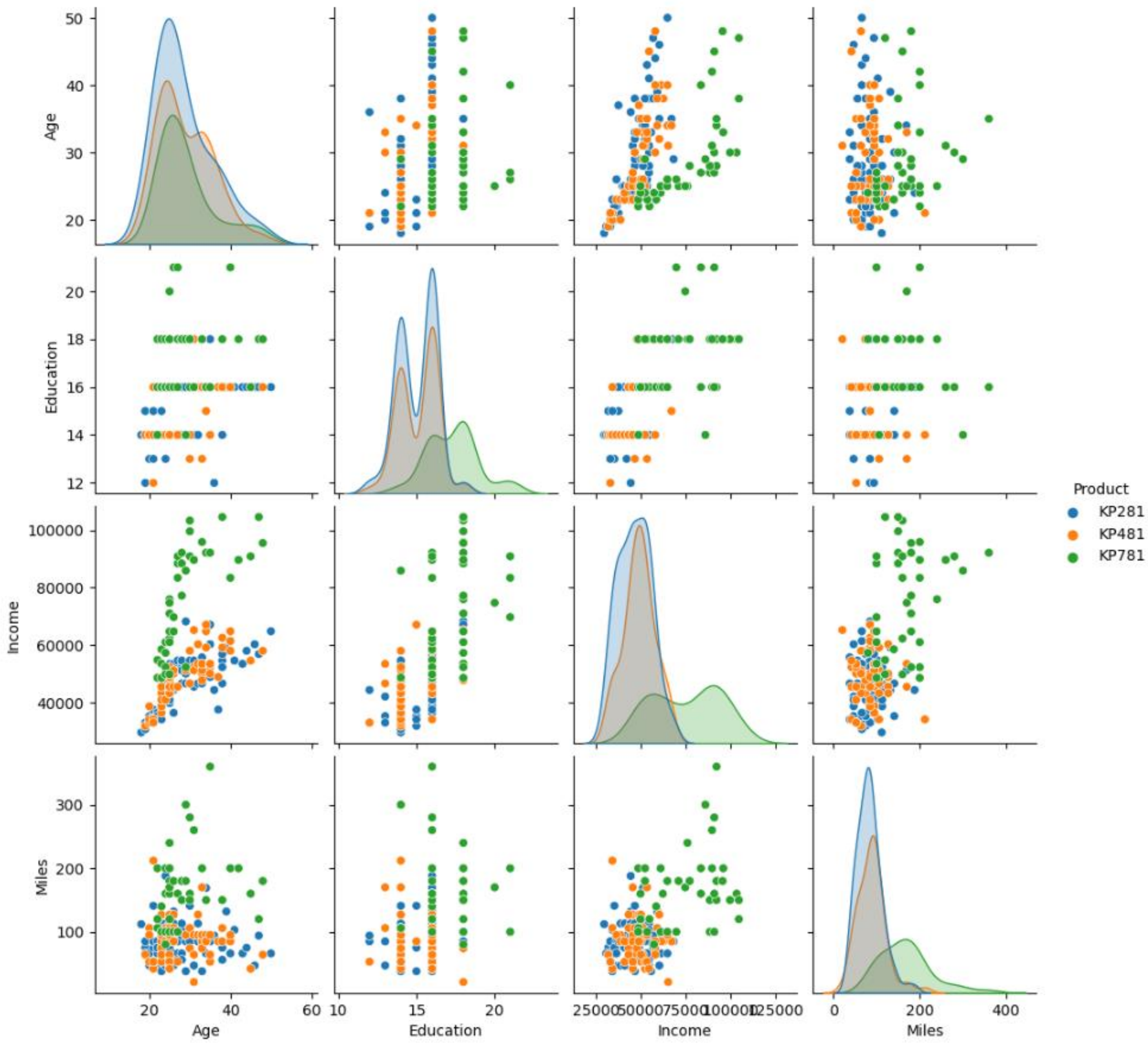
1. The probability of purchasing a product having moderate activity (51 to 100 miles/week) is 54%.
   - The conditional probability of purchasing a model given that the customer is having moderate activity
     - For KP281: 28%
     - For KP481: 22%
     - For KP781: 4%
2. The probability of purchasing a product having active lifestyle (100 to 200 miles/week) is 33%.
   - The conditional probability of purchasing a model given that the customer is having active lifestyle
     - For KP281: 10%
     - For KP481: 8%
     - For KP781: 15%
3. The probability of purchasing a product having light activity (0 to 50 miles/week) is 9%.
   - The conditional probability of purchasing a model given that the customer is having light activity
     - For KP281: 7%
     - For KP481: 3%
     - For KP781: 0%
4. The probability of purchasing a product having fitness enthusiastic (> 200 miles/week) is 3%.

5. Check the correlation among different factors

1. Pair Plot

[348]:
```
## Correlation among different factors
df = customers
df['Fitness'] = df['Fitness'].astype('str')
df['Usage'] = df['Usage'].astype('str')
```

[349]:
```
sns.pairplot(data = customers, hue = 'Product')
plt.show()
```



2. Heatmap:

[350]:
```
customers['Usage'] = customers['Usage'].astype('int64')
customers['Fitness'] = customers['Fitness'].astype('int64')
customers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 13 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   
 9   Age_Group        180 non-null    category
 10  Education_Group  180 non-null    category
 11  Income_Group     180 non-null    category
 12  Miles_Group      180 non-null    category
dtypes: category(4), int64(6), object(3)
memory usage: 14.2+ KB
```

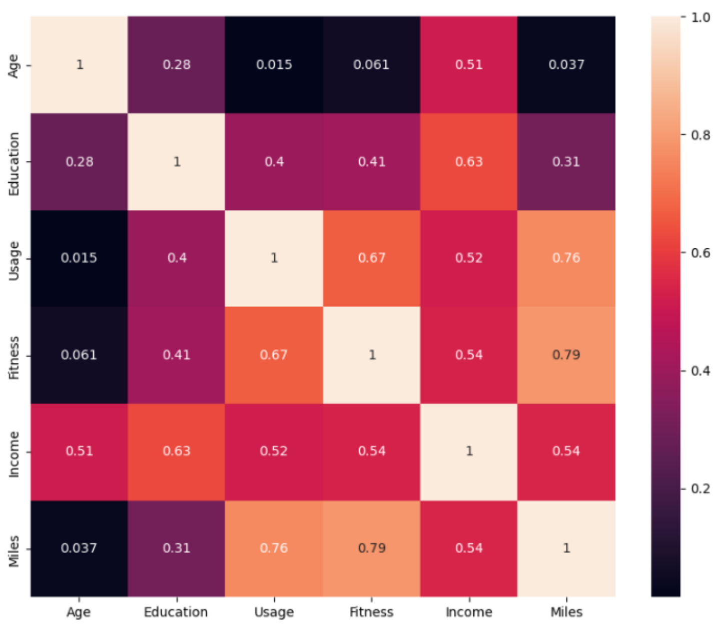
[351]:
```
df = customers[['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']]
```

[352]:
```
df.corr()
```

[352]:

	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

[354]:
```
plt.figure(figsize = (10, 8))
sns.heatmap(df.corr(), annot = True)
plt.show()
```



Insights:

1. Education and Income is positively correlated as it is obvious. Education also have a good correlation with Fitness rating and Usage of treadmills.

2. Age and Income is positively correlated.
3. Usage is highly correlated with Fitness and Miles as more the usage one can achieve more fitness and walk more miles.

6. Customer Profiling:

Based on the all above analysis we can say that,

- Probability of purchase of KP281 = 44%
- Probability of purchase of KP481 = 33%
- Probability of purchase of KP781 = 22%

a. Customer Profile for KP281 Treadmill:

- Age of customer mainly between 18 to 35 years with few between 35 to 50 years
- Education level of customer 13 years and above
- Annual Income of customer below USD 60,000
- Weekly Usage - 2 to 4 times
- Fitness Scale - 2 to 4
- Weekly Running Mileage - 50 to 100 miles

b. Customer Profile for KP481 Treadmill:

- Age of customer mainly between 18 to 35 years with few between 35 to 50 years
- Education level of customer 13 years and above
- Annual Income of customer between USD 40,000 to USD 80,000
- Weekly Usage - 2 to 4 times
- Fitness Scale - 2 to 4
- Weekly Running Mileage - 50 to 200 miles

c. Customer Profile for KP781 Treadmill:

- Gender - Male
- Age of customer between 18 to 35 years
- Education level of customer 15 years and above
- Annual Income of customer USD 80,000 and above
- Weekly Usage - 4 to 7 times
- Fitness Scale - 3 to 5
- Weekly Running Mileage - 100 miles and above

7. Recommandations:

a. The advance model treadmill KP781 exhibits a significant sales disparity in terms of gender, with only 18% of total sales attributed to female customers. To enhance this metric, it is recommended to implement targeted strategies such as offering special promotions and trials exclusively designed for the female customers.

b. Given the target customer's age, education level, and income, it's important to offer the KP281 and KP481 Treadmill at an affordable price point. Additionally, consider providing flexible payment plans that allow customers to spread the cost over several months. This can make the treadmill more accessible to customers with varying budgets. If customers received a flexible payment method one can also think to purchase the advance model KP781 also which can increase the AeroFit's revenue.

c. Create a user-friendly app that syncs with the treadmill which will convert the treadmill into a smart device. This app could track users' weekly running mileage, provide real-time feedback on their progress, and offer personalized recommendations for workouts based on their fitness scale and goals. This can enhance the overall treadmill experience and keep users engaged. Also by this AeroFit will get a good quality of data for their prediction analysis over the models.