Treadmill Customer Profile

Project Objective

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 new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

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

Product Portfolio:

- The KP281 is an entry-level treadmill that sells for $1,500;
- The KP481 is for mid-level runners and sells for $1,750;
- The KP781 treadmill is having advanced features and it sells for $2,500.

Table of Contents

- Data Exploration and Processing
- Statistical Summary
- Non-Graphical Analysis
- Graphical Analysis
- Marginal & Conditional Probabilities
- Actionable Insights & Recommendations

Data Exploration and Processing

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set(color_codes=True)
import scipy.stats as stats
import warnings
warnings.filterwarnings("ignore")
import os
```

In [2]:
```
# Importing data
os.listdir(os.getcwd())
```

Out[2]:
```
['.ipynb_checkpoints',
 'aerofit_treadmill_data.csv',
 'Overview of Python Writeup.png',
 'Treadmill Purchase Analysis.ipynb',
 'Treadmill Purchase Analysis.pdf']
```

In [3]:
```
df = pd.read_csv('aerofit_treadmill_data.csv')
# Get overview of dataset
df.head()
```

Out[3]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	KP281	18	Male	14	Single	3	4	29562	112
1	KP281	19	Male	15	Single	2	3	31836	75
2	KP281	19	Female	14	Partnered	4	3	30699	66
3	KP281	19	Male	12	Single	3	3	32973	85
4	KP281	20	Male	13	Partnered	4	2	35247	47

In [4]:
```
# Shape of dataframe
df.shape
```

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

In [5]:
```
# Identify name of all columns
df.columns
```

Out[5]:
```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles'],
      dtype='object')
```

In [6]:
```
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 [7]:
```
# Convert columns with object data type
df['Product'] = df['Product'].astype('category')
df['Gender'] = df['Gender'].astype('category')
df['MaritalStatus'] = df['MaritalStatus'].astype('category')
```

In [8]:
```
df.dtypes
```

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

In [9]:
```
df.skew()
```

Out[9]:
```
Age          0.982161
Education    0.622294
Usage        0.739494
Fitness      0.454800
Income       1.291785
Miles        1.724497
dtype: float64
```

Statistical Summary

In [10]:
```
df.describe(include='all')
```

Out[10]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
count	180	180.000000	180	180.000000	180	180.000000	180.000000	180.000000	180.000000
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	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53719.577778	103.194444
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16506.684226	51.863605
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29562.000000	21.000000
25%	NaN	24.000000	NaN	14.000000	NaN	3.000000	3.000000	44058.750000	66.000000
50%	NaN	26.000000	NaN	16.000000	NaN	3.000000	3.000000	50596.500000	94.000000
75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58668.000000	114.750000
max	NaN	50.000000	NaN	21.000000	NaN	7.000000	5.000000	104581.000000	360.000000

Observations

- there are no missing values in the data
- there are 3 unique `products`
- `KP281` is the most frequent product
- min `age` is 18, max `age` is 50, and average `age` is 28.79
- most of the people have at most `16 years of education`
- majority of the data is from `male`

In [11]:
```
# Check for missing values
df.isnull().sum()
```

Out[11]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

In [12]:
```
# Check for duplicates in dataset
df.duplicated().sum()
```

Out[12]:
```
0
```

Non-Graphical Analysis

Value Counts

In [13]:
```
df['Product'].value_counts()
```

Out[13]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [14]:
```
df['Gender'].value_counts()
```

Out[14]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [15]:
```
df['MaritalStatus'].value_counts()
```

Out[15]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

Unique Attributes

In [16]:
```
df.nunique()
```

Out[16]:
```
Product           3
Age              32
Gender            2
Education         8
MaritalStatus     2
Usage             6
Fitness           5
Income           62
Miles            37
dtype: int64
```

In [17]:
```
df['Product'].unique()
```

Out[17]:
```
['KP281', 'KP481', 'KP781']
Categories (3, object): ['KP281', 'KP481', 'KP781']
```

In [18]:
```
df['Age'].unique()
```

Out[18]:
```
array([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],
      dtype=int64)
```

In [19]:
```
df['Education'].unique()
```

Out[19]:
```
array([14, 15, 12, 13, 16, 18, 20, 21], dtype=int64)
```

In [20]:
```
df['MaritalStatus'].unique()
```

Out[20]:
```
['Single', 'Partnered']
Categories (2, object): ['Partnered', 'Single']
```

In [21]:
```
df['Usage'].unique()
```

Out[21]:
```
array([3, 2, 4, 5, 6, 7], dtype=int64)
```

In [22]:
```
df['Fitness'].unique()
```

Out[22]:
```
array([4, 3, 2, 1, 5], dtype=int64)
```

In [23]:
```
df['Income'].unique()
```

Out[23]:
```
array([ 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], dtype=int64)
```

In [24]:
```
df['Miles'].unique()
```

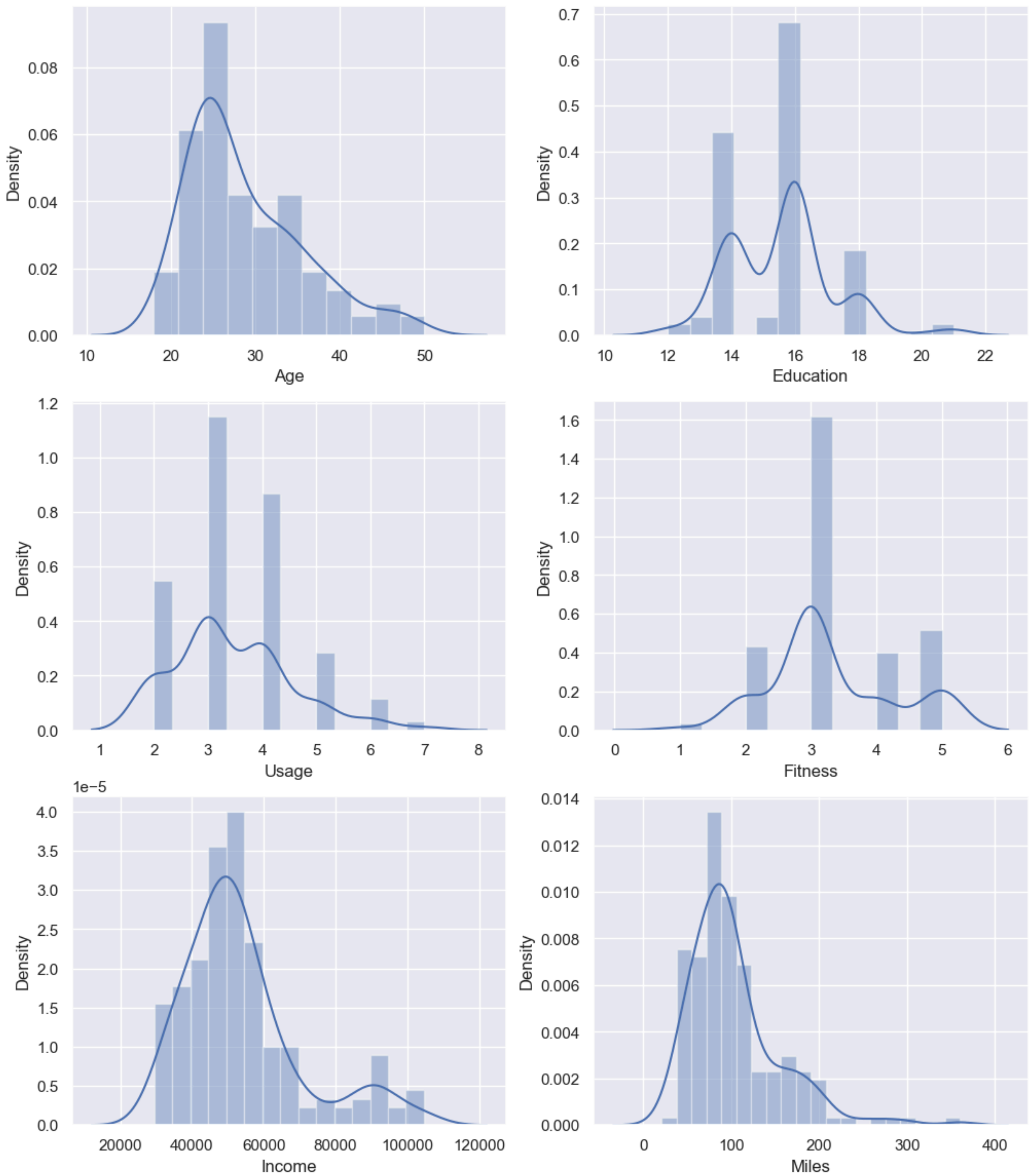
Out[24]:
```
array([112,  75,  66,  85,  47, 141, 103,  94, 113,  38, 188,  56, 132,
       169,  64,  53, 106,  95, 212,  42, 127,  74, 170,  21, 120, 200,
       140, 100,  80, 160, 180, 240, 150, 300, 280, 260, 360], dtype=int64)
```

Graphical Analysis

Univariate Analysis - Numerical Variables

Distance Plot

In [25]:
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12,10))
fig.subplots_adjust(top=1.2)

sns.distplot(df['Age'], kde=True, ax=axis[0,0])
sns.distplot(df['Education'], kde=True, ax=axis[0,1])
sns.distplot(df['Usage'], kde=True, ax=axis[1,0])
sns.distplot(df['Fitness'], kde=True, ax=axis[1,1])
sns.distplot(df['Income'], kde=True, ax=axis[2,0])
sns.distplot(df['Miles'], kde=True, ax=axis[2,1])
plt.show()
```

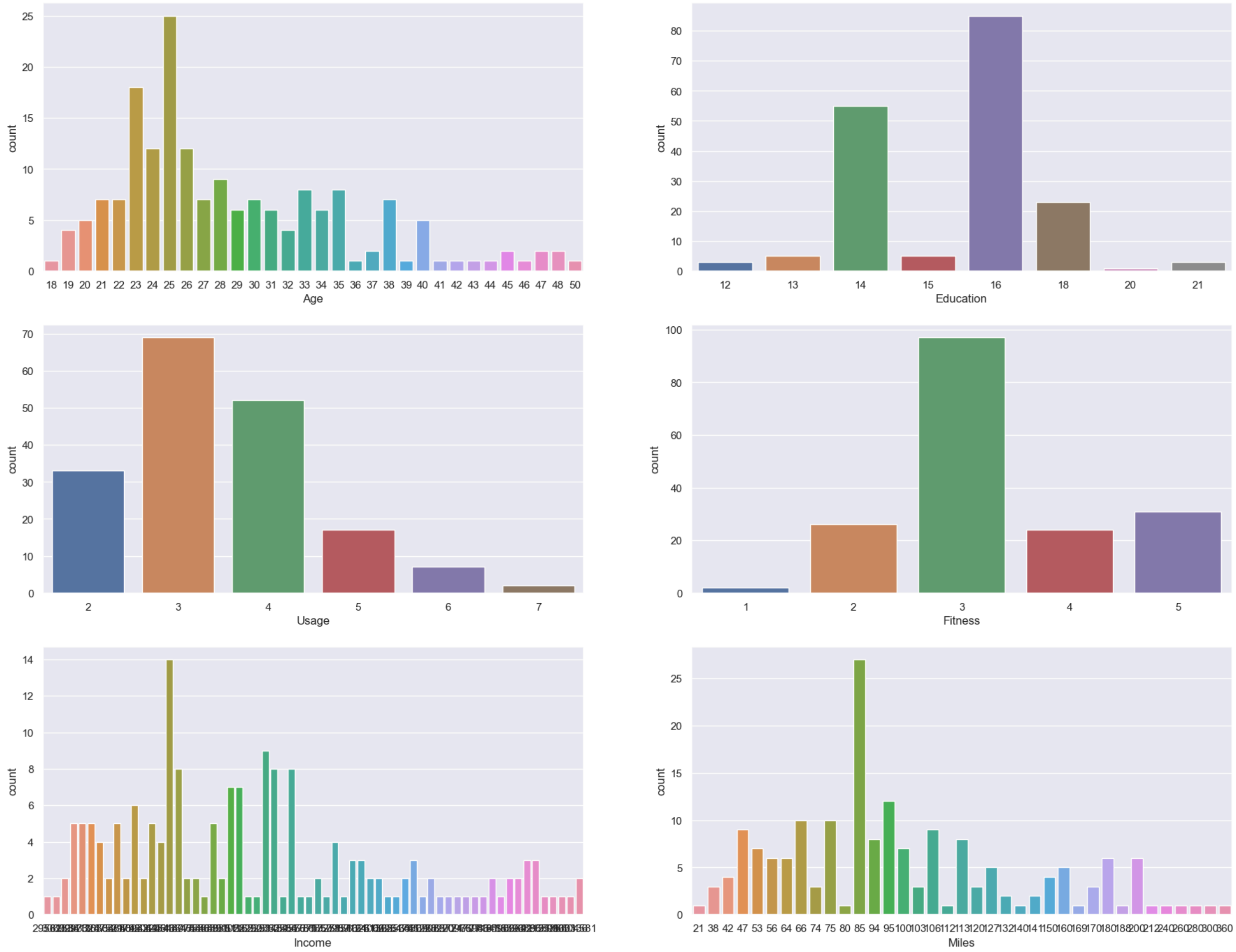


Observations

- `Income` and `Miles` are skewed to the right suggesting that they outliers
- most customers has `Fitness` level 3
- most customer has `Income` within the range of `$45,000 - $55,000`

Count Plot

In [26]:
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(22,12))
fig.subplots_adjust(top=1.2)

sns.countplot(data=df, x='Age', ax=axis[0,0])
sns.countplot(data=df, x='Education', ax=axis[0,1])
sns.countplot(data=df, x='Usage', ax=axis[1,0])
sns.countplot(data=df, x='Fitness', ax=axis[1,1])
sns.countplot(data=df, x='Income', ax=axis[2,0])
sns.countplot(data=df, x='Miles', ax=axis[2,1])
plt.show()
```



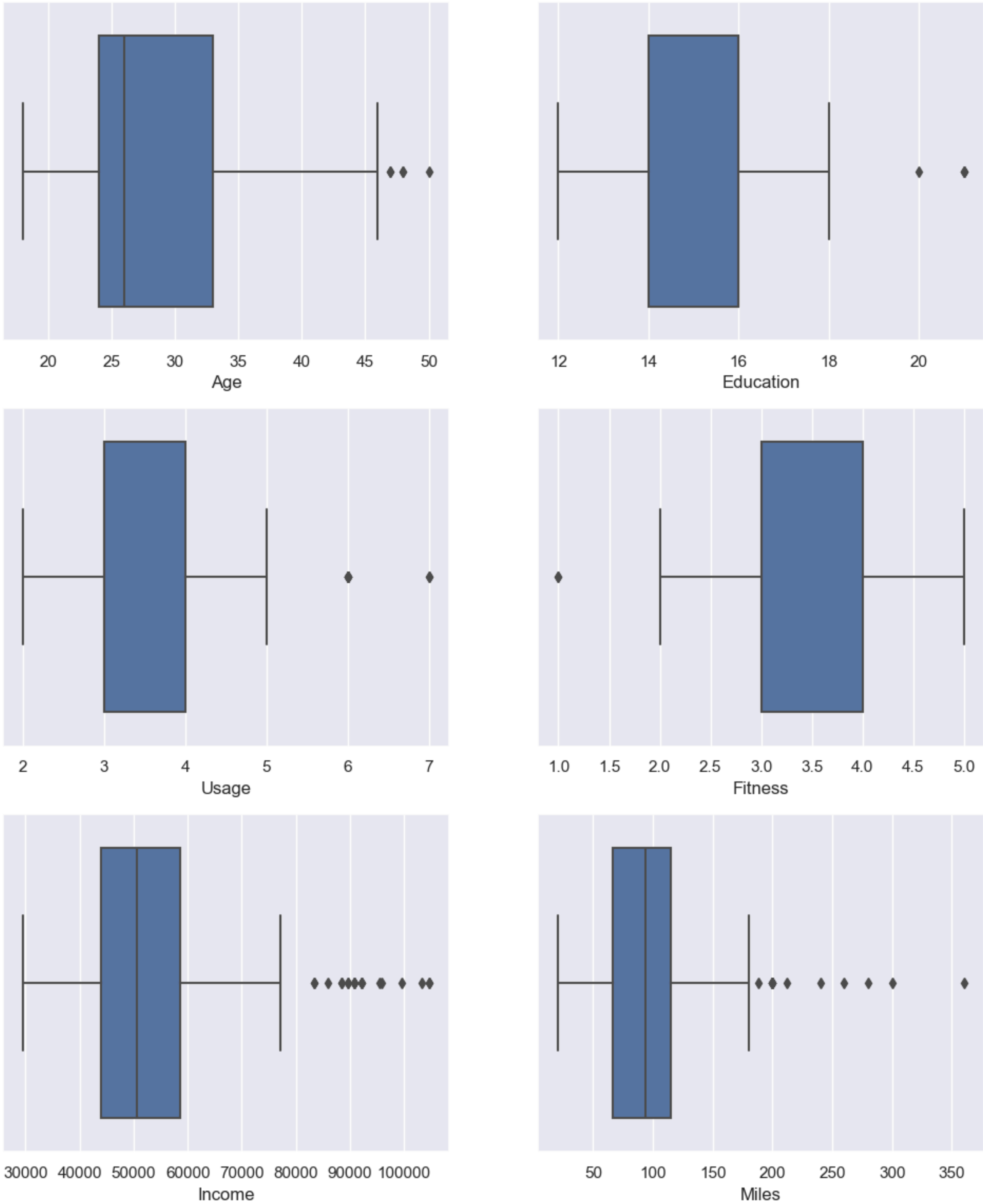
Observations

- people of `age 25` are more inclined to buy treadmills compared to older people

Box Plot

In [27]:

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12,10))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x='Age', ax=axis[0,0])
sns.boxplot(data=df, x='Education', ax=axis[0,1])
sns.boxplot(data=df, x='Usage', ax=axis[1,0])
sns.boxplot(data=df, x='Fitness', ax=axis[1,1])
sns.boxplot(data=df, x='Income', ax=axis[2,0])
sns.boxplot(data=df, x='Miles', ax=axis[2,1])
plt.show()
```



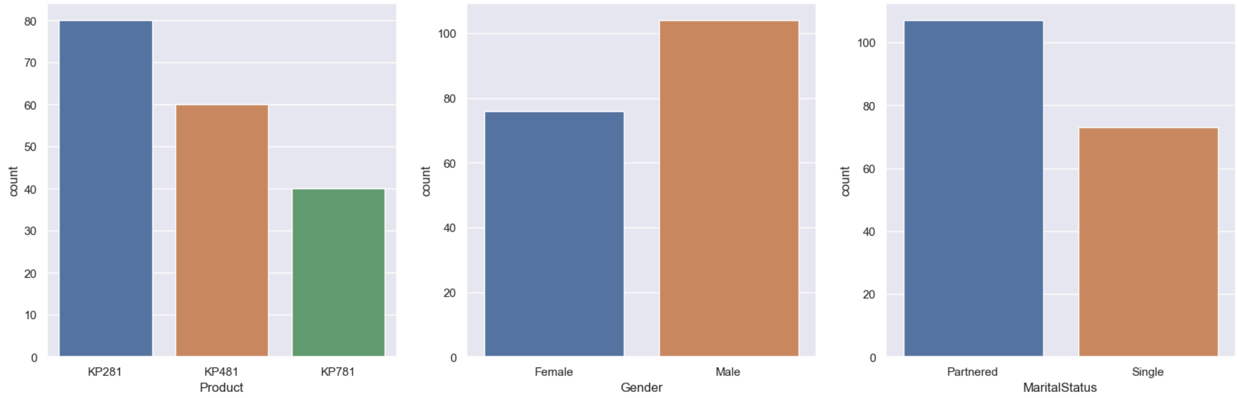
Observations

- `Age`, `Education`, `Usage`, and `Fitness` have very few outliers

Univariate Analysis - Categorical Variables

Count Plot

In [28]:
```
fig, axis = plt.subplots(nrows=1, ncols=3, figsize=(20,6))
sns.countplot(data=df, x='Product', ax=axis[0])
sns.countplot(data=df, x='Gender', ax=axis[1])
sns.countplot(data=df, x='MaritalStatus', ax=axis[2])

plt.show()
```



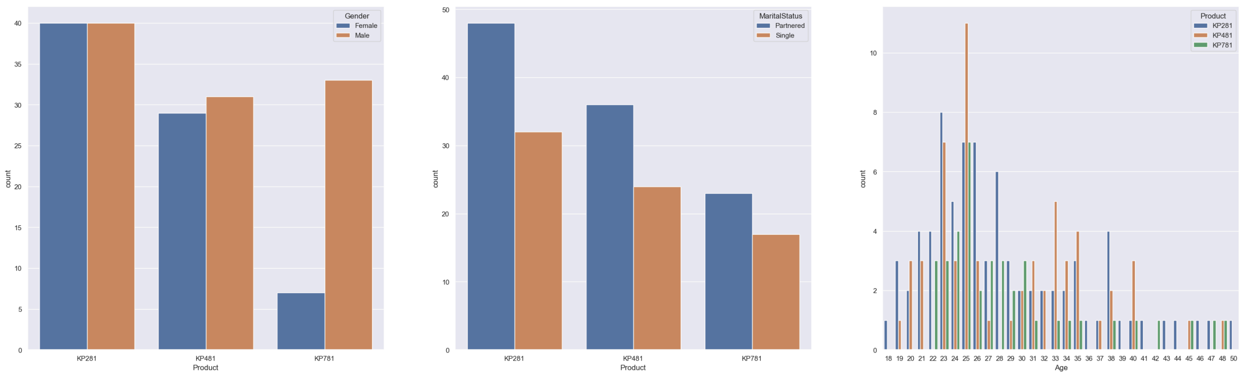
Observations

- most popular treadmill is the `KP281`
- most of the customers are `Male`
- most customers that purchase treadmills are `Partnered`

Bivariate Analysis

Checking if features have any effect on product being purchased.

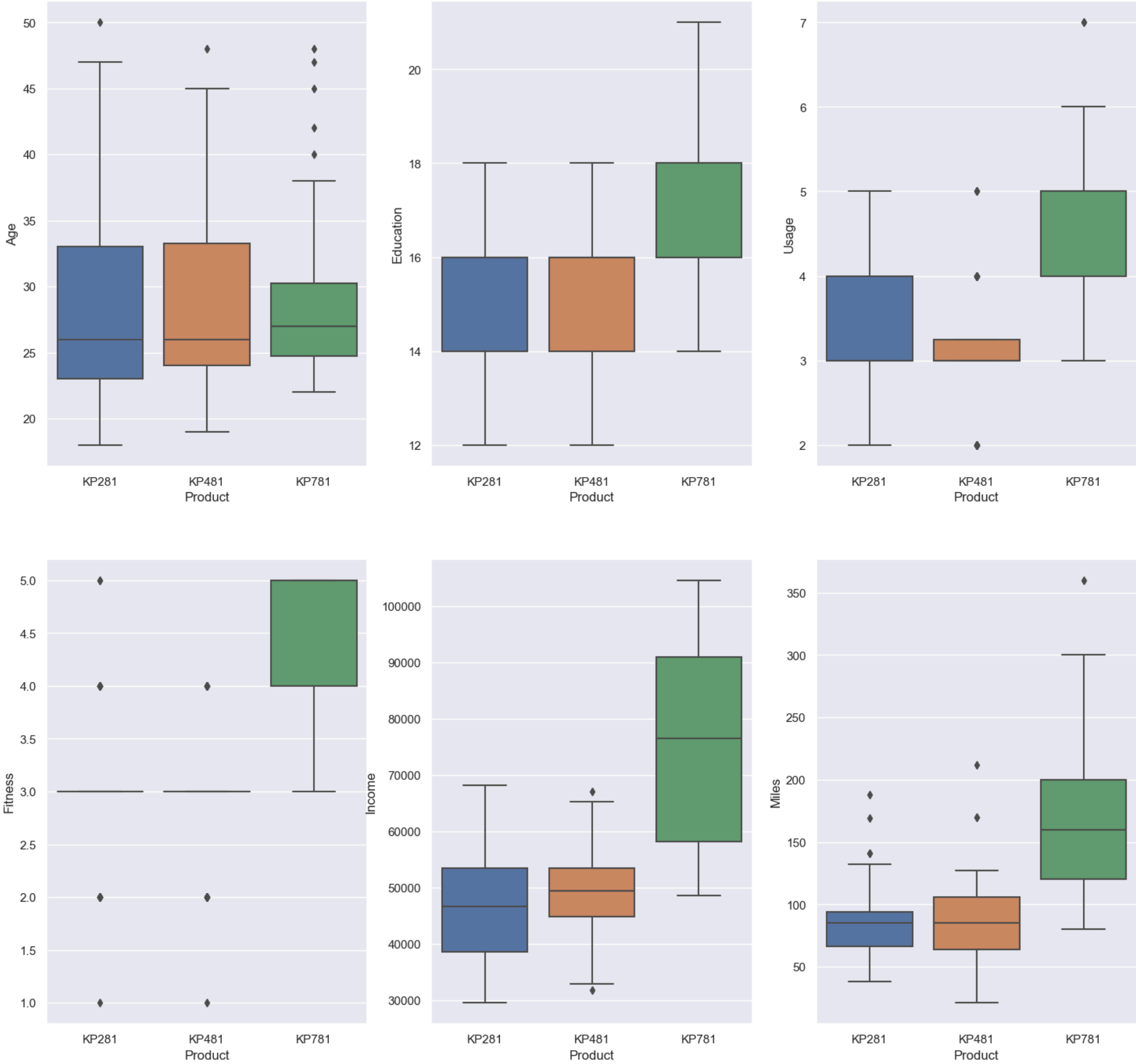
In [29]:
```
fig, axis = plt.subplots(nrows = 1, ncols = 3, figsize=(35,10))
sns.countplot(data = df, x = 'Product', hue = 'Gender', ax = axis[0])
sns.countplot(data = df, x = 'Product', hue = 'MaritalStatus', ax = axis[1])
sns.countplot(data = df, x = 'Age', hue = 'Product', ax = axis[2])

plt.show()
```



Observations

- Equal number of males and females have purchased the `KP281` which is the most desirable
- Most of the purchases were from `partnered` customers
- Customers of the `age 25` are more likely to purchase the `KP481`

In [30]:
```
attributes = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
sns.set(color_codes = True)
fig,axis = plt.subplots(nrows = 2, ncols = 3, figsize = (18,12))
fig.subplots_adjust(top = 1.2)
count = 0
for i in range(2):
    for j in range(3):
        sns.boxplot(data = df, x = 'Product', y = attributes[count], ax = axis[i,j])
        count += 1
```



Observations

Product vs Age

- `KP281` and `KP481` share the same customer's median `Age`.
- Customers between age 25 - 30 are likely to purchase the `KP781`.

Product vs Education

- Customers that has over 16 years of education are more likely to purchase `KP781`.
- Customers with less than 16 years of education have equal chance of purchasing `KP281` or `KP481`.

Product vs Usage

- Customers who purchased `KP781` are likely to use it more than 4 times a week.

Product vs Fitness

- Customers with high fitness level (fitness > 3) have a higher chance of purchasing the `KP781`.

Product vs Income

- Customers with higher income (income > 60,000) are more likely to purchase the `KP781`.

Product vs Miles

- Customers that walk/run for more than 120 miles per week are likelier to buy the `KP781`.

Correlation Analysis

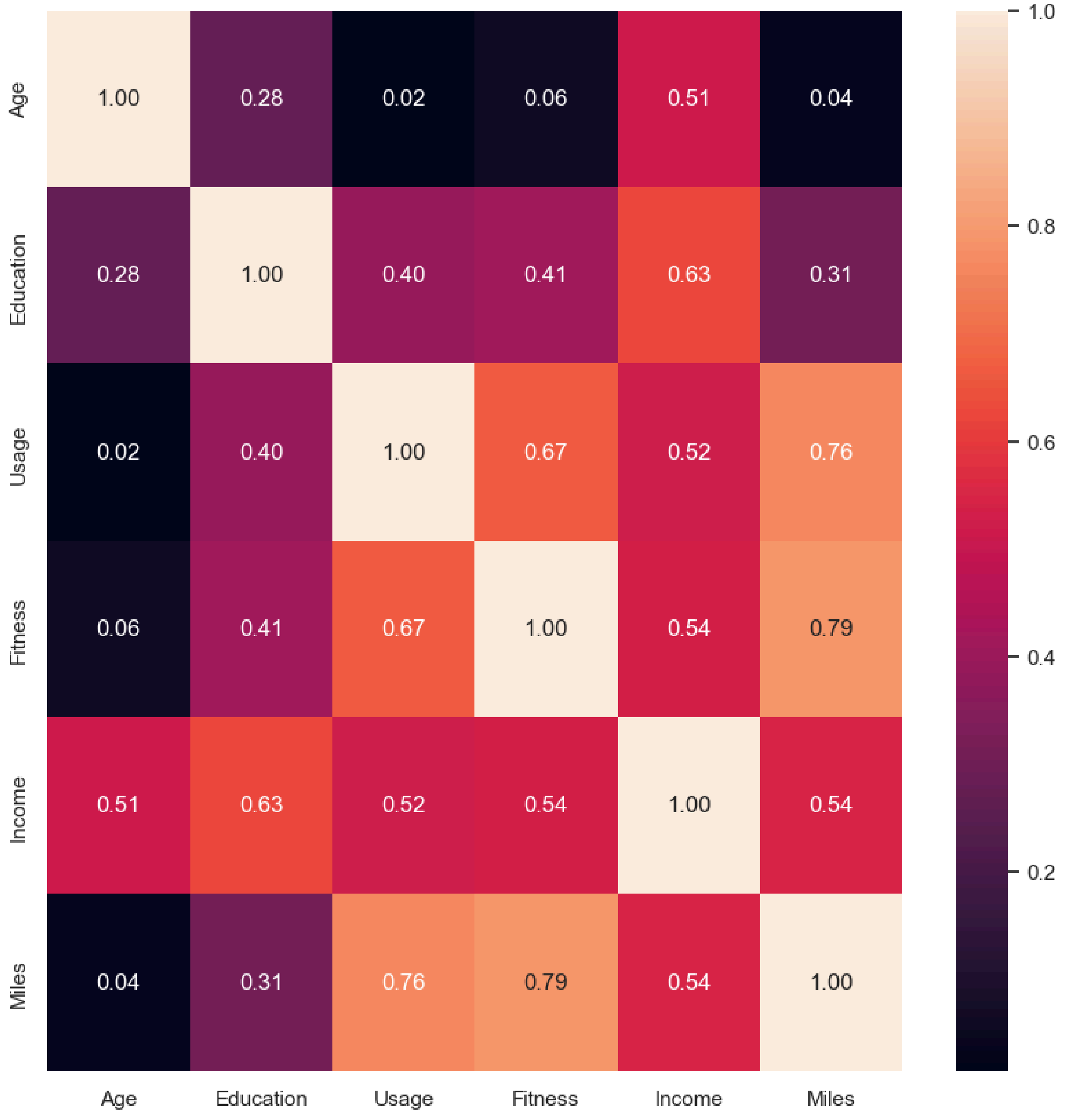
In [31]:
```
df.corr()
```

Out[31]:

	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

Heatmaps

In [32]:
```
fig, ax = plt.subplots(figsize=(10,10))
sns.set(color_codes=True)
sns.heatmap(df.corr(), ax=ax, annot=True, fmt='0.2f')
plt.show()
```



Observation

- (`Miles` & `Usage`) and (`Miles` & `Fitness`) attributes are highly correlated which means fit customers tend to use more treadmills.
- `Income` and `Education` shows a strong correlation. Customers with high income and very educated prefer the `KP781` treadmill.

Marginal & Conditional Probabilities

What percent of customers have purchased KP281, KP481, or KP781?

In [36]:
```
df1 = df[['Product', 'Gender', 'MaritalStatus']].melt()
(df1.groupby(['variable', 'value'])[['value']].count()/len(df)).mul(100).round(3).astype(str) + '%'
```

Out[36]:

		value
variable	**value**	
Gender	**Female**	42.222%
	Male	57.778%
MaritalStatus	**Partnered**	59.444%
	Single	40.556%
Product	**KP281**	44.444%
	KP481	33.333%
	KP781	22.222%

Obervations

- #### Product
    - 44.44% of the customers have purchased `KP281` product.
    - 33.33% of the customers have purchased `KP481` product.
    - 22.22% of the customers have purchased `KP781` product.
- #### Gender
    - 57.78% of the customers are `Male` .
- #### MaritalStatus
    - 59.44% of the customers are `Partnered` .

What is the probability of a customer based on `Gender` (Male or Female) buying a certain treadmill `Product` ?

In [55]:
```
def p_prod_given_gender(gender, print_marginal=False):
    if gender is not "Female" and gender is not "Male":
        return "Invalid gender value"

    df1 = pd.crosstab(index=df['Gender'], columns=df['Product'])
    p_281 = df1['KP281'][gender] / df1.loc[gender].sum()
    p_481 = df1['KP481'][gender] / df1.loc[gender].sum()
    p_781 = df1['KP781'][gender] / df1.loc[gender].sum()

    if print_marginal:
        print(f"P(Male): {df1.loc['Male'].sum()/len(df):.2f}")
        print(f"P(Female): {df1.loc['Female'].sum()/len(df):.2f}\n")

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

p_prod_given_gender('Male', True)
p_prod_given_gender('Female')
```

```
P(Male): 0.58
P(Female): 0.42

P(KP281/Male): 0.38
P(KP481/Male): 0.30
P(KP781/Male): 0.32

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

What is the probability of a customer based on `MaritalStatus` (Single or Partnered) buying a certain treadmill `Product` ?

In [58]:
```
def p_prod_given_mstatus(status, print_marginal=False):
    if status is not "Single" and status is not "Partnered":
        return "Invalid marital status value"

    df1 = pd.crosstab(index=df['MaritalStatus'], columns=df['Product'])
    p_281 = df1['KP281'][status] / df1.loc[status].sum()
    p_481 = df1['KP481'][status] / df1.loc[status].sum()
    p_781 = df1['KP781'][status] / df1.loc[status].sum()

    if print_marginal:
        print(f"P(Single): {df1.loc['Single'].sum()/len(df):.2f}")
        print(f"P(Partnered): {df1.loc['Partnered'].sum()/len(df):.2f}\n")

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

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

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

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

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

Product - Gender

In [62]:
```
product_gender = pd.crosstab(index=df['Product'], columns=df['Gender'], margins=True)
product_gender
```

Out[62]:

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

In [63]:
```
# Percentage of a Male customer purchasing a treadmill
prob = round((product_gender['Male']['All'] / product_gender['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[63]: 58.0

In [64]:
```
# Percentage of a Female customer purchasing KP781 treadmill
prob = round((product_gender['Female']['KP781'] / product_gender['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[64]: 4.0

In [66]:
```
# Percentage of a customer being a Female given that Product is KP281
prob = round((product_gender['Female']['KP281'] / product_gender['All']['KP281']),2)
pct = round(prob*100,2)
pct
```
Out[66]:
```
50.0
```

Observations:

- Female customer prefer to buy KP281 & KP481
- 50% of female tend to purchase treadmill model KP281

Product - Age

In [69]:
```
df2 = df.copy()
```

In [70]:
```
# Extracting 2 new features from Age:
# "AgeCategory" - Teens, 20s, 30s, and Above 40s
# "AgeGroup" - 14-20, 20-30, 30-40 & 40-60

bins = [14,20,30,40,60]
labels = ["Teens", "20s", "30s", "Above 40s"]
df2['AgeGroup'] = pd.cut(df2['Age'], bins)
df2['AgeCategory'] = pd.cut(df2['Age'], bins, labels=labels)
```

In [71]:
```
df2.tail()
```

Out[71]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	AgeGroup	AgeCategory
175	KP781	40	Male	21	Single	6	5	83416	200	(30, 40]	30s
176	KP781	42	Male	18	Single	5	4	89641	200	(40, 60]	Above 40s
177	KP781	45	Male	16	Single	5	5	90886	160	(40, 60]	Above 40s
178	KP781	47	Male	18	Partnered	4	5	104581	120	(40, 60]	Above 40s
179	KP781	48	Male	18	Partnered	4	5	95508	180	(40, 60]	Above 40s

In [73]:
```
product_age = pd.crosstab(index=df2['Product'], columns=df2['AgeCategory'], margins=True)
product_age
```

Out[73]:

AgeCategory	Teens	20s	30s	Above 40s	All
Product					
KP281	6	49	19	6	80
KP481	4	31	23	2	60
KP781	0	30	6	4	40
All	10	110	48	12	180

In [96]:
```
# Percentage of customers with Age between 20s and 30s among all customers
prob = round((product_age['20s']['All'] / product_age['All']['All']),2)
pct = round(prob*100,2)
pct
```
Out[96]:
```
61.0
```

Observations:

- Teens don't prefer to buy KP781
- 61% of customers are between 20 and 30 years old

Product - Income

In [75]:
```
df3 = df2.copy()
```

In [76]:
```
# Extracting 1 new categorical feature based on the Income:
# "IncomeCategory" - Low Income, Lower-Middle Income, Upper-Middle Income, High Income

bins_income = [29000, 35000, 60000, 85000, 105000]
```

```
labels_income = ["Low Income", "Lower-Middle Income", "Upper-Middle Income", "High Income"]
df3['IncomeCategory'] = pd.cut(df3['Income'], bins=bins_income, labels=labels_income)
```

In [77]:
```
df3.head()
```

Out[77]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	AgeGroup	AgeCategory	IncomeCategory
0	KP281	18	Male	14	Single	3	4	29562	112	(14, 20]	Teens	Low Income
1	KP281	19	Male	15	Single	2	3	31836	75	(14, 20]	Teens	Low Income
2	KP281	19	Female	14	Partnered	4	3	30699	66	(14, 20]	Teens	Low Income
3	KP281	19	Male	12	Single	3	3	32973	85	(14, 20]	Teens	Low Income
4	KP281	20	Male	13	Partnered	4	2	35247	47	(14, 20]	Teens	Lower-Middle Income

In [79]:
```
product_income = pd.crosstab(index=df3['Product'], columns=df3['IncomeCategory'], margins=True)
product_income
```

Out[79]:

IncomeCategory	Low Income	Lower-Middle Income	Upper-Middle Income	High Income	All
Product					
KP281	8	66	6	0	80
KP481	6	47	7	0	60
KP781	0	11	12	17	40
All	14	124	25	17	180

In [81]:
```
# Percentage of a low-income customer purchasing a treadmill
prob = round((product_income['Low Income']['All'] / product_income['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[81]: 8.0

In [83]:
```
# Percentage of a high-income customer purchasing KP781 treadmill
prob = round((product_income['High Income']['KP781'] / product_income['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[83]: 9.0

In [84]:
```
# Percentage of a high-income customer buying treadmill given that Product is KP781
prob = round((product_income['High Income']['All'] / product_income['All']['KP781']),2)
pct = round(prob*100,2)
pct
```

Out[84]: 42.0

Product - Fitness

In [85]:
```
product_fitness = pd.crosstab(index=df3['Product'], columns=df3['Fitness'], margins=True)
product_fitness
```

Out[85]:

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 [87]:
```
# Percentage of a customer having fitness level 5
prob = round((product_fitness[5]['All'] / product_fitness['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[87]:
```
17.0
```

In [88]:
```
# Percentage of a customer with fitness level 5 purchasing KP781 treadmill
prob = round((product_fitness[5]['KP781'] / product_fitness['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[88]:
```
16.0
```

In [89]:
```
# Percentage of a customer with fitness level 5 buying KP781 treadmill given that Product is KP781
prob = round((product_fitness[5]['KP781'] / product_fitness['All']['KP781']),2)
pct = round(prob*100,2)
pct
```

Out[89]:
```
72.0
```

Product - Marital Status

In [90]:
```
product_marital = pd.crosstab(index=df3['Product'], columns=df3['MaritalStatus'], margins=True)
product_marital
```

Out[90]:

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

In [92]:
```
# Percentage of customers who are partnered using treadmills
prob = round((product_marital['Partnered']['All'] / product_marital['All']['All']),2)
pct = round(prob*100,2)
pct
```

Out[92]:
```
59.0
```

Actionable Insights & Recommendations

Actionable Insights:

- Model KP281 is the best-selling product. 44.0% of all treadmill sales go to model KP281.
- The majority of treadmill customers fall within the $ 45,000 - \$ 80,000 income bracket.
  - 83% of treadmills are bought by individuals with incomes between $ 35,000 and \$ 85,000
  - There are only 8% of customers with incomes below $ 35000 who buy treadmills.
- 88% of treadmills are purchased by customers aged 20 to 40.
- Miles and Fitness & Miles and Usage are highly correlated, which means if a customer's fitness level is high they use more treadmills.
- KP781 is the only model purchased by a customer who has more than 20 years of education and an income of over $ 85,000.
- With Fitness level 4 and 5, the customers tend to use high-end treadmills and the average number of miles is above 150 per week

Recommendations:

- KP281 & KP481 are popular with customer income of $ 45,000 - \$ 60,000 and can be offered by these companies as affordable models.
- KP781 should be marketed as a Premium Model and marketing it to high income groups and educational over 20 years market segments could result in more sales.
- The KP781 is a premium model, so it is ideally suited for sporty people who have a high average weekly mileage and can be afforded by the high income customers.
- Aerofit should conduct market research to determine if it can attract customers with income under $ 35,000 to expand its customer base

In [97]:
```
df3.to_excel('treadmill_customer_data.xlsx')
```