Business Case: Aerofit - Descriptive Statistics & Probability

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:

Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading the dataset

```
df = pd.read_csv('aerofit_treadmill.csv')
df
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	KP281	18	Male	14	Single	3	4	29562	112
1	KP281	19	Male	15	Single	2	3	31836	75
2	KP281	19	Female	14	Partnered	4	3	30699	66
3	KP281	19	Male	12	Single	3	3	32973	85
4	KP281	20	Male	13	Partnered	4	2	35247	47
...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200
177	KP781	45	Male	16	Single	5	5	90886	160
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

180 rows × 9 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

Knowing the Data

Shape of the data

```
df.shape
```

```
(180, 9)
```

Columns in the dataset

```
df.columns
```

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

Data types of the columns in dataset

```
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
```

Statistical summary for numerical columns

```
df.describe()
```

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

Statistical summary of object columns

```
df.describe(include ='object')
```

	Product	Gender	MaritalStatus
count	180	180	180
unique	3	2	2
top	KP281	Male	Partnered
freq	80	104	107

Non-Graphical Analysis: Value counts and unique attributes

Unique & Nunique

```
# This loop will help me with unique data for all the columns
for i in df.columns:
  print(i, ':', df[i].unique())
```

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

```
# This loop will help me with unique data count for all the columns
for i in df.columns:
  print(i, ':', df[i].nunique())
```

```
Product : 3
Age : 32
Gender : 2
Education : 8
MaritalStatus : 2
Usage : 6
Fitness : 5
Income : 62
Miles : 37
```

Value Counts

```
# Product columns
A = (df['Product'].value_counts(normalize = True)* 100).round(2)
Product_count = A.reset_index()
Product_count
```

	Product	proportion
0	KP281	44.44
1	KP481	33.33
2	KP781	22.22

Next steps: Generate code with Product_count | View recommended plots | New interactive sheet

Insights - 44.44% customers prefer to buy/Use KP281 treadmill which can be due to its affordable cost as compared to other treadmills, 33.33% of users prefer KP481 treadmill, while 22.22% of users prefers KP781 treadmills.

```
# Gender column
(df['Gender'].value_counts(normalize = True)*100).round(2)
```

Gender	proportion
Male	57.78
Female	42.22

dtype: float64

Insights - Aerofit has 57.78% of the Male customers which is more compared to female customers that is 42.22%

```
#Marital status
(df['MaritalStatus'].value_counts(normalize = True) * 100).round(2)
```

MaritalStatus	proportion
Partnered	59.44
Single	40.56

dtype: float64

Insights - Aerofit has 59.44% of customeres who are Married and 40.56% of customers who are single

```
#Marital status
(df['Age'].value_counts(normalize = True) * 100).round(2)
```

Age	proportion
25	13.89
23	10.00
24	6.67
26	6.67
28	5.00
35	4.44
33	4.44
30	3.89
38	3.89
21	3.89
22	3.89
27	3.89
31	3.33
34	3.33
29	3.33
20	2.78
40	2.78
32	2.22
19	2.22
48	1.11
37	1.11
45	1.11
47	1.11
46	0.56
50	0.56
18	0.56
44	0.56
43	0.56
41	0.56
39	0.56
36	0.56
42	0.56

dtype: float64

Insights - By observing the value counts for age column, Most of the customers of Aerofit belongs to the age group from 23 years to to 30 years

Handling of the missing Values

```
# Check for the null values in the dataset

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

	0
Product	0
Age	0
Gender	0
Education	0
MaritalStatus	0
Usage	0
Fitness	0
Income	0
Miles	0

dtype: int64

We dont have any null values in the dataset which will help us to analyze the data more accuratly and help us give the appropriate probabilities and counts.

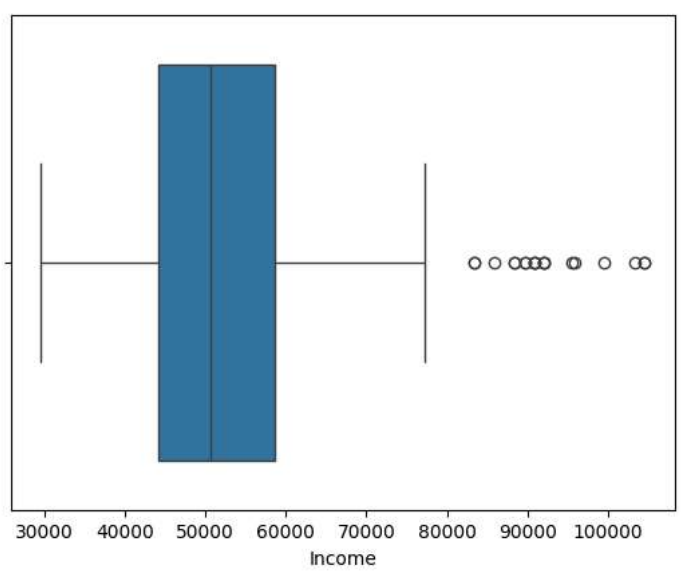
Outliers detection

```
# To find the Outliers we need to use the boxplot for the necessary columns
# lets find the 5 points first to detect the outliers q1, IQR, q3, Upper bound, lower bound

q1 = df['Income'].quantile(0.25)
q3 = df['Income'].quantile(0.75)
IQR = q3 - q1
lower_bound = q1 - 1.5 * IQR
upper_bound = q3 + 1.5 * IQR


# lets check with the boxplot if we have quartiles in the upper bound or lower bound in Income column

sns.boxplot(data = df, x = df['Income'])
plt.show()
```




```
upper_bound
```

```
80581.875
```

- We can see the outlier for Income column at the upper bound
- All the values above 80581.875 are outliers in the column

```
(len(df.loc[df['Income'] > upper_bound])/len(df) * 100)
```

```
10.555555555555555
```

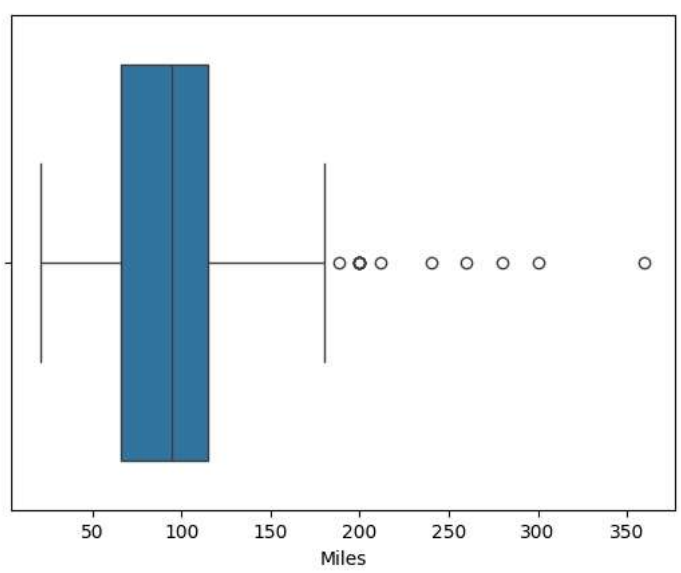
10.5% of he values in income column are outliers, but since they are less i am not dropping them, Also they might be usefull for some valuable insights further.

```
#lets check with the boxplot if we have quartiles in the upper bound or lower bound in Miles Column

q1 = df['Miles'].quantile(0.25)
q3 = df['Miles'].quantile(0.75)
IQR = q3 - q1
lower_bound = q1 - 1.5 * IQR
upper_bound = q3 + 1.5 * IQR

# lets check with the boxplot if we have quartiles in the upper bound or lower bound in Income column

sns.boxplot(data = df, x = df['Miles'])
plt.show()
```




```
upper_bound
```

```
187.875
```

- We can see the outlier for Income column at the upper bound
- All the values above 80581.875 are outliers in the column

```
(len(df.loc[df['Miles'] > upper_bound])/len(df) * 100)
```

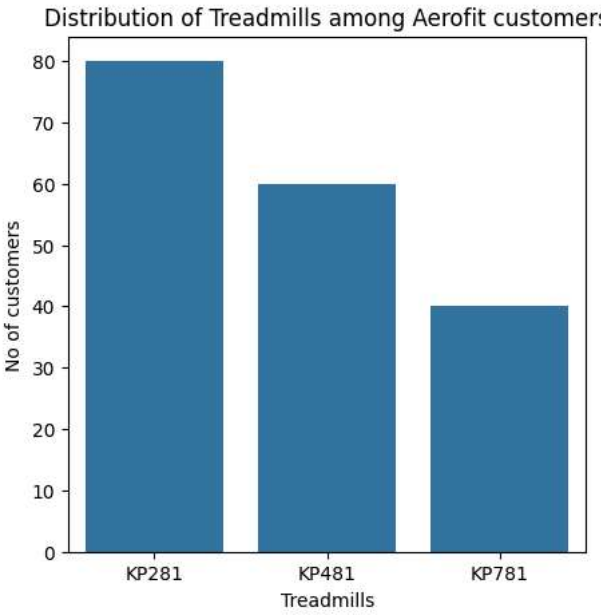
```
7.222222222222221
```

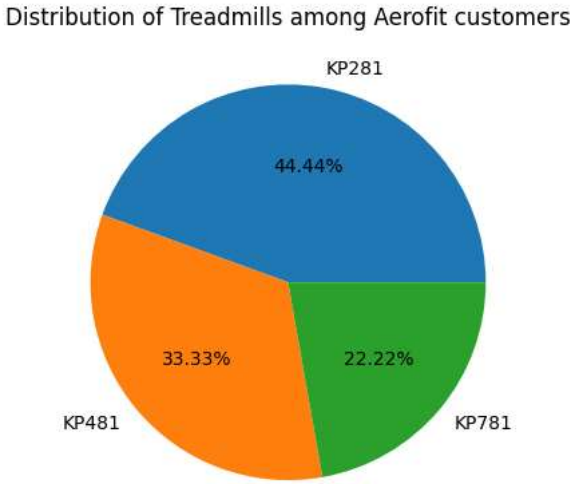
7.22% of the values in Miles column are outliers, but since they are less i am not dropping them, Also they might be usefull for some valuable insights further.

Univariate Analysis

Distribution of Treadmills among Aerofit customers

```
plt.figure(figsize = (5,5))
sns.countplot(data = df, x = 'Product')
plt.xlabel('Treadmills')
plt.ylabel('No of customers')
plt.title('Distribution of Treadmills among Aerofit customers')
plt.show()
plt.pie(df['Product'].value_counts(), labels = df['Product'].unique(), autopct = '%.2f%%')
plt.title('Distribution of Treadmills among Aerofit customers')
plt.show()
```





Insights -

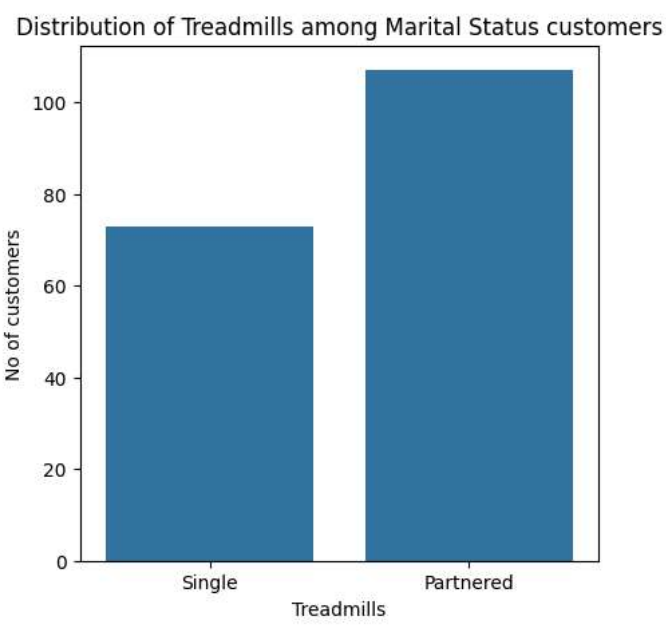
1. 44.44% customers prefer to buy/Use KP281 treadmill which can be due to its affordable cost as compared to other treadmills, 33.33% of users prefer KP481 treadmill, while 22.22% of users prefers KP781 treadmills.
2. KP281 being an entry level and the most afforadable treadmill is the most presffered choice among the customers.
3. KP781 being an advance level & expensive treadmill is used by 22.22% customers.

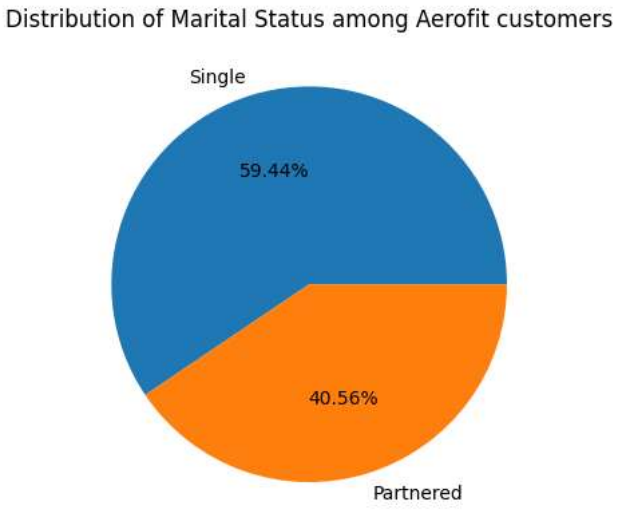
Recommendations - Continue this afforadable budget for the KP281 treadmill as it will strictly attract the more number of customers who are starting with their fitness journey.

Provide specila offers and discount on the sale of KP781 Treadmills as most users should be educated about its advance features

Distribution of Marital Status among Aerofit customers

```
plt.figure(figsize = (5,5))
sns.countplot(data = df, x = 'MaritalStatus')
plt.xlabel('Treadmills')
plt.ylabel('No of customers')
plt.title('Distribution of Treadmills among Marital Status customers')
plt.show()
plt.pie(df['MaritalStatus'].value_counts(), labels = df['MaritalStatus'].unique(), autopct = '%.2f%%')
plt.title('Distribution of Marital Status among Aerofit customers')
plt.show()
```



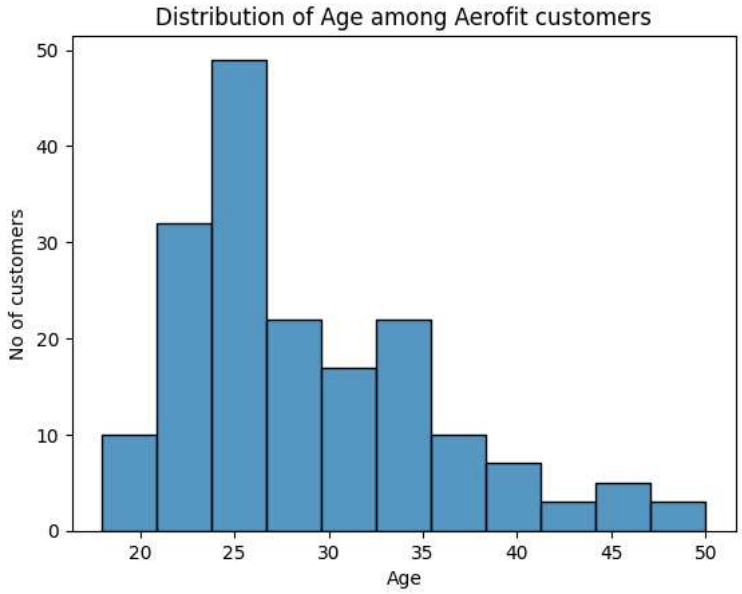


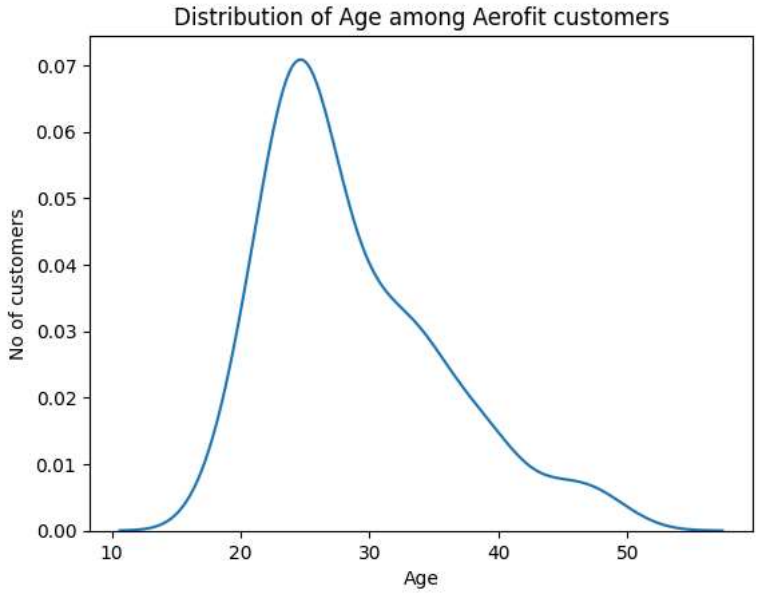
Insights - Aerofit has 59.44% of customeres who are Married and 40.56% of customers who are single

Distribution of Age

```
# Histogram
sns.histplot(data = df, x = 'Age')
plt.xlabel('Age')
plt.ylabel('No of customers')
plt.title('Distribution of Age among Aerofit customers')
plt.show()

# KDE
sns.kdeplot(data = df, x = 'Age')
plt.xlabel('Age')
plt.ylabel('No of customers')
plt.title('Distribution of Age among Aerofit customers')
plt.show()
```





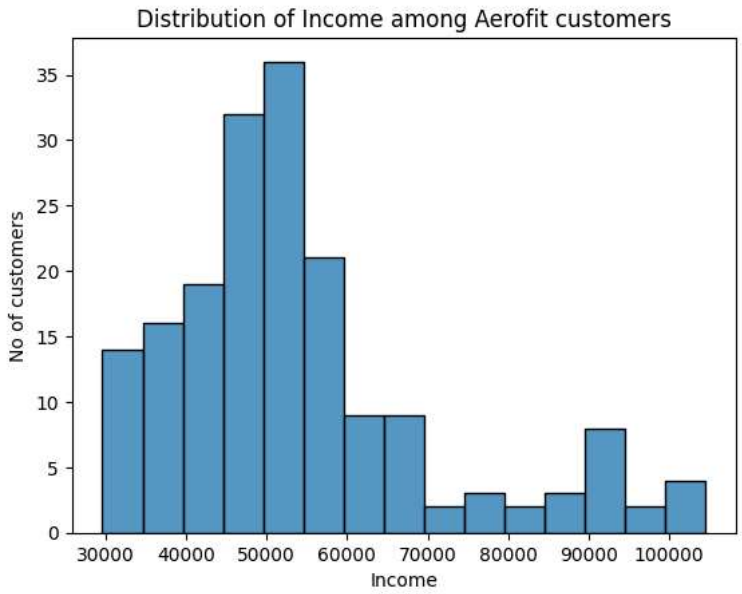
Insights - The majority of the customers of Aerofit belongs to the age group of 19 - 35 years old, and there are very few customers who are more than 35 to 40 years old.

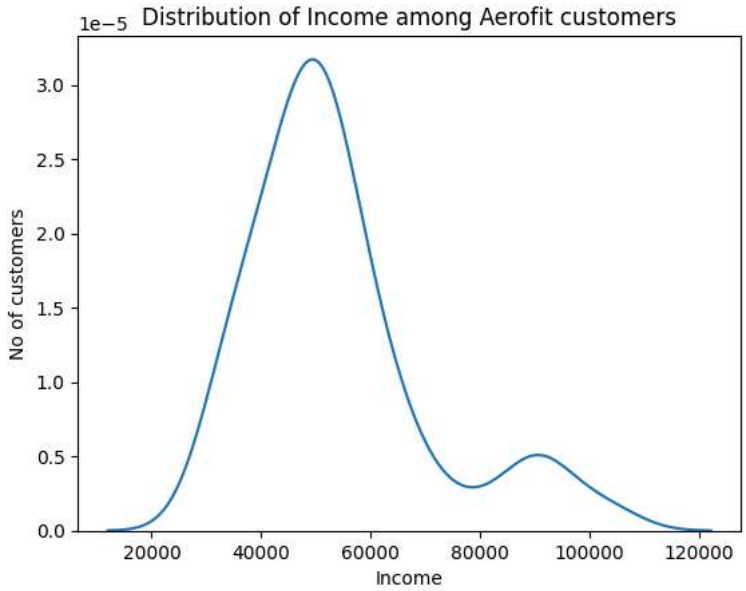
Recommendations - Aerofit should recommend more benifits of products accordingly to the older age groups and motivate them to achieve their fitness goals

Distribution of Income

```
# Histogram
sns.histplot(data = df, x = 'Income')
plt.xlabel('Income')
plt.ylabel('No of customers')
plt.title('Distribution of Income among Aerofit customers')
plt.show()

# KDE
sns.kdeplot(data = df, x = 'Income')
plt.xlabel('Income')
plt.ylabel('No of customers')
plt.title('Distribution of Income among Aerofit customers')
plt.show()
```



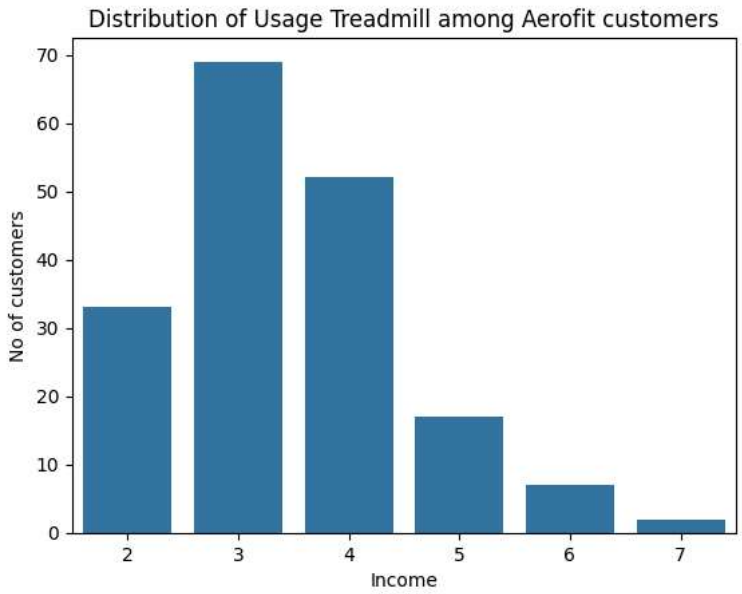


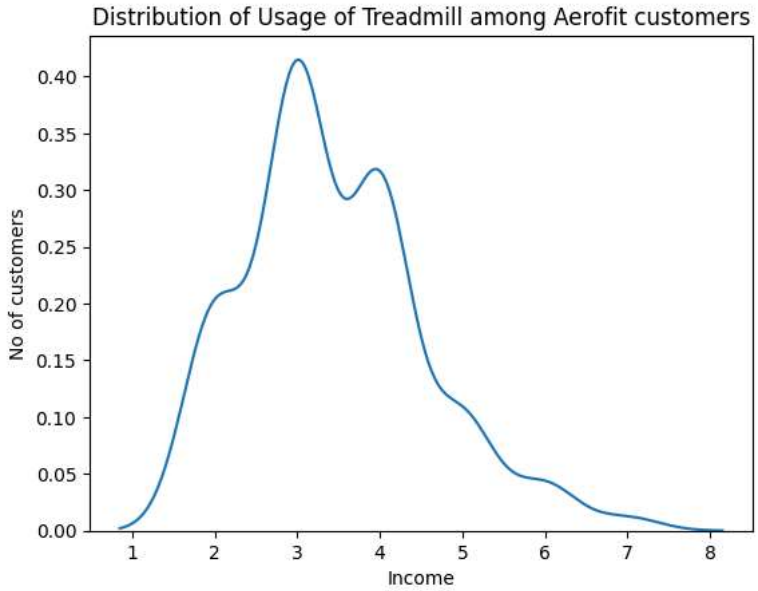
Insights- The majority of Aerofit customers are in the income range of 40000 to 60000, they have the highlest probability of purchasing the products. but we dont see the customers in the income range of more than 80000 purchasing the products, the probability is too less.

Distribution of Usage

```
# countplot
sns.countplot(data = df, x = 'Usage')
plt.xlabel('Income')
plt.ylabel('No of customers')
plt.title('Distribution of Usage Treadmill among Aerofit customers')
plt.show()

# KDE
sns.kdeplot(data = df, x = 'Usage')
plt.xlabel('Income')
plt.ylabel('No of customers')
plt.title('Distribution of Usage of Treadmill among Aerofit customers')
plt.show()
```



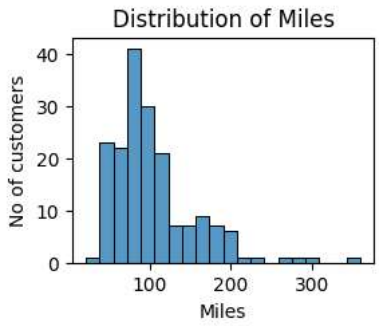


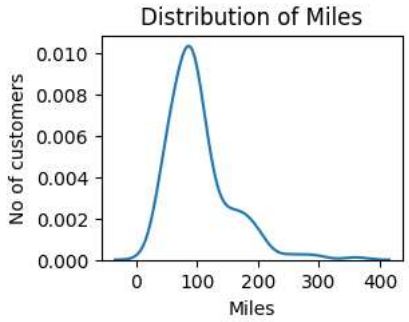
Insights - Frequency of Most of the Aerofit customers using the treadmills is 3 to 4 times a week.

Distribution of Miles traveled by Aerofit customers

```
# countplot
plt.subplot(2,2,1)
sns.histplot(data = df, x = 'Miles')
plt.xlabel('Miles')
plt.ylabel('No of customers')
plt.title('Distribution of Miles')
plt.show()

# KDE
plt.subplot(2,2,2)
sns.kdeplot(data = df, x = 'Miles')
plt.xlabel('Miles')
plt.ylabel('No of customers')
plt.title('Distribution of Miles')
plt.show()
```



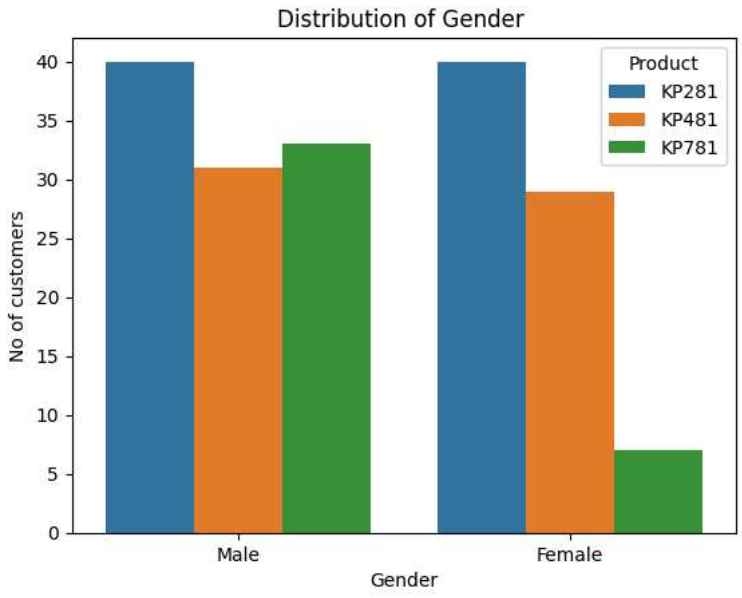


Insights - the most of the customers runs 90-100 miles on Aerofit treadmills.

Bivariate Analysis

Distribution of Gender and check whether it has any effect on the product purchased

```
sns.countplot(data = df, x = 'Gender', hue = 'Product')
plt.xlabel('Gender')
plt.ylabel('No of customers')
plt.title('Distribution of Gender ')
plt.show()
```



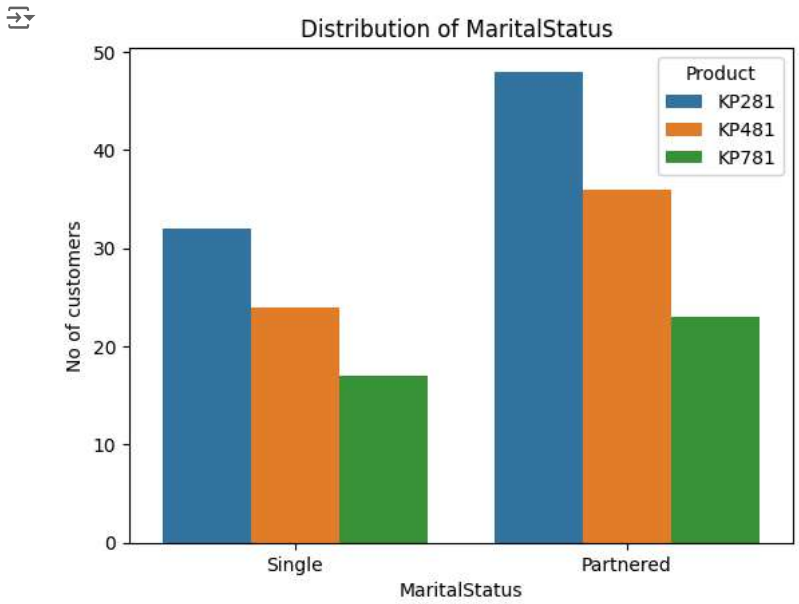
Insights -

1. KP281 is equally preferred by both Male and Female customers due to its entry level qualities and afforadable cost.
2. We even dont see much difference KP481 treadmills, though we have more number of male customers, but it still equally prefered by the femail customers.

3. For KP781 treadmills there are more female customers as compared to amle customers. the probability of female customers buying it is very less.

Distribution of Marital Status

```
sns.countplot(data = df, x = 'MaritalStatus', hue = 'Product')
plt.xlabel('MaritalStatus')
plt.ylabel('No of customers')
plt.title('Distribution of MaritalStatus ')
plt.show()
```

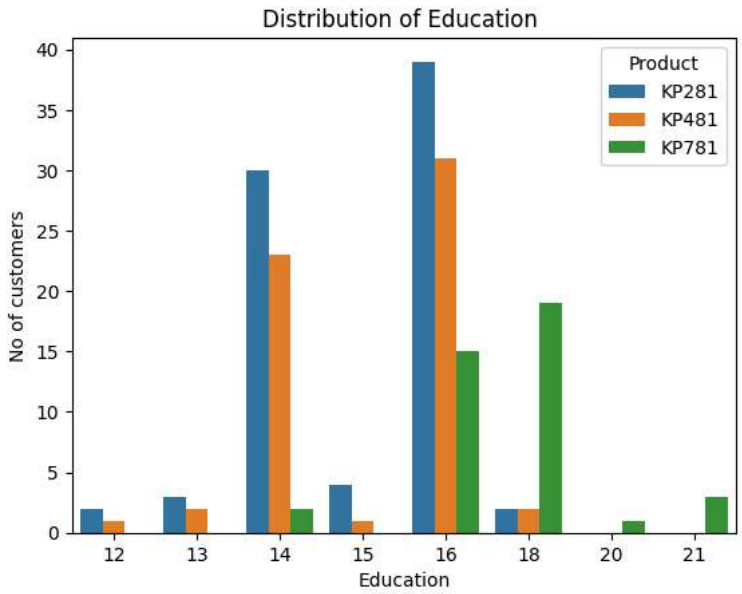


Insights -

1. Overall if we observe, for all the the three types of Treadmills - Marries customers has frequency of purchasing them.
2. Again KP281 having an entry level benifits and most affordable one it is prefeered by both marries and unmarries customers the most.
3. KP781 is the least purchased treamill amoung both marries and unmarries customers.

Distribution of Education

```
sns.countplot(data = df, x = 'Education', hue = 'Product')
plt.xlabel('Education')
plt.ylabel('No of customers')
plt.title('Distribution of Education ')
plt.show()
```



Insights -

1. Customers having education experience of 14 or 16 years mostly prefer to buy KP281 and KP481 treadmills. However again the most prefered treadmill is KP281.
2. Customers having education of 18 years also tend to buy the most expensive treadmill that is KP781 due its advance features.

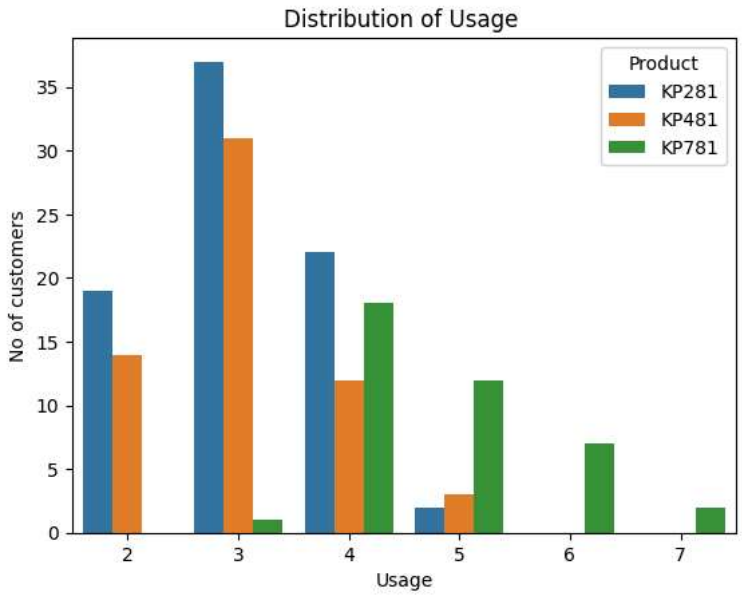
Distribution of Usage and Fitness accross each Treadmill

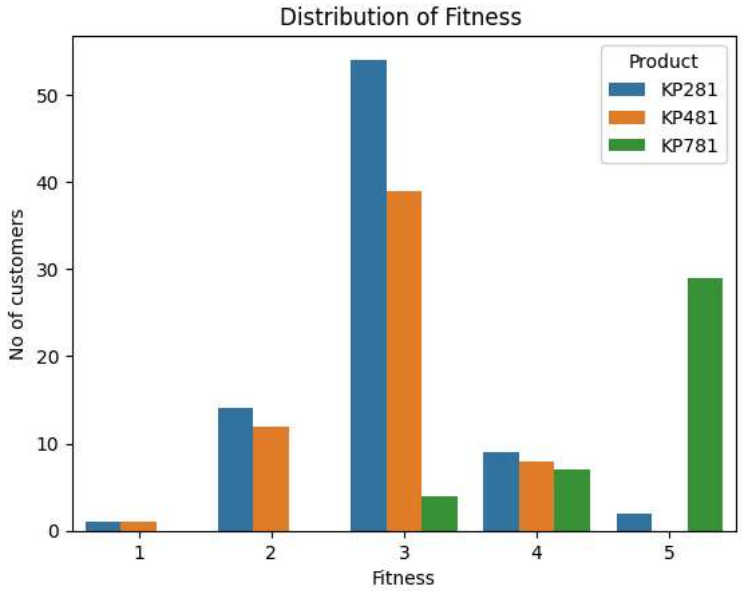
```
# Usage

sns.countplot(data = df, x = 'Usage', hue = 'Product')
plt.xlabel('Usage')
plt.ylabel('No of customers')
plt.title('Distribution of Usage ')
plt.show()

#fitness

sns.countplot(data = df, x = 'Fitness', hue = 'Product')
plt.xlabel('Fitness')
plt.ylabel('No of customers')
plt.title('Distribution of Fitness ')
plt.show()
```





Insights-

1. The customers that use the treadmills 3 times per week mostly prefer to use KP281 and KP481.
2. The customers that use the treadmills 5 times a week prefers KP781 Treadmills due to its advan facilities.

Adding Income group

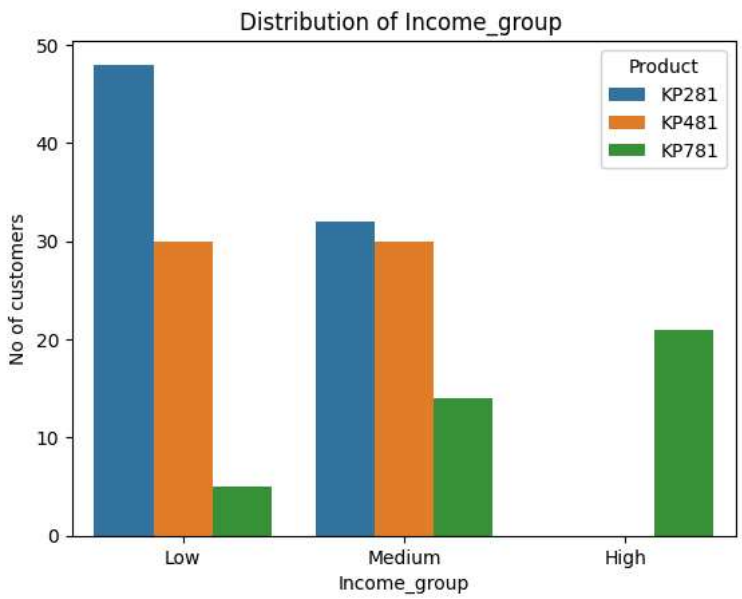
```
df['Income_group'] = pd.cut(df['Income'], bins = [0,50000,75000, 105000], labels = ['Low', 'Medium', 'High'])
df
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Income_group
0	KP281	18	Male	14	Single	3	4	29562	112	Low
1	KP281	19	Male	15	Single	2	3	31836	75	Low
2	KP281	19	Female	14	Partnered	4	3	30699	66	Low
3	KP281	19	Male	12	Single	3	3	32973	85	Low
4	KP281	20	Male	13	Partnered	4	2	35247	47	Low
...	...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200	High
176	KP781	42	Male	18	Single	5	4	89641	200	High
177	KP781	45	Male	16	Single	5	5	90886	160	High
178	KP781	47	Male	18	Partnered	4	5	104581	120	High
179	KP781	48	Male	18	Partnered	4	5	95508	180	High

180 rows × 10 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
sns.countplot(data = df, x = 'Income_group', hue = 'Product')
plt.xlabel('Income_group')
plt.ylabel('No of customers')
plt.title('Distribution of Income_group ')
plt.show()
```



Insights

1. The majority of customers that belongs to low income groups prefers to user KP281 Treadmills due to its affordable cost, Also there are few customers from this group that still prefers to use the KP481 treadmills and very low prefer(2-3 customers) KP78.
2. The customers from Medium income groups prefers to use both KP281 and KP481. and again very less (11-12) prefers the most expensive one.
3. The customers from High income range groups. all of them prefers the most expensive treadmills that is KP781.

Adding Age groups to check the impact

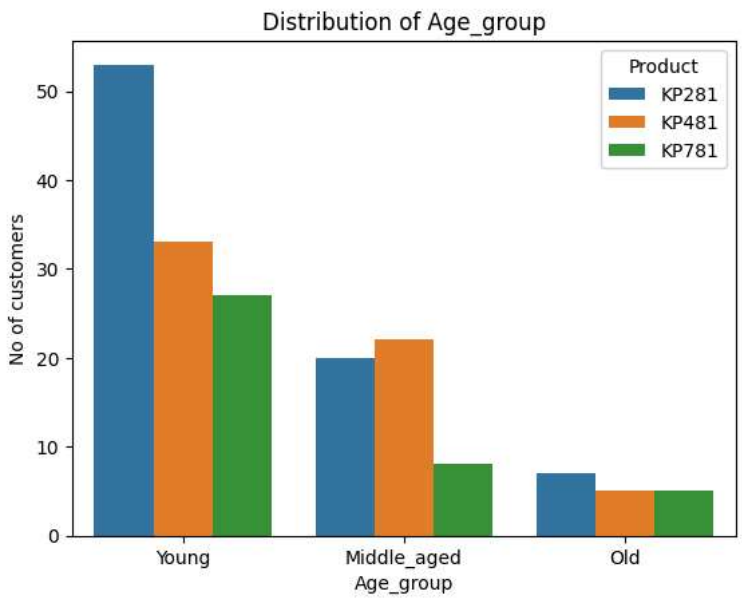
```
df['Age_group'] = pd.cut(df['Age'], bins = [0,29, 39, 50], labels = ['Young', 'Middle_aged', 'Old'])
df
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Income_group	Age_group
0	KP281	18	Male	14	Single	3	4	29562	112	Low	Young
1	KP281	19	Male	15	Single	2	3	31836	75	Low	Young
2	KP281	19	Female	14	Partnered	4	3	30699	66	Low	Young
3	KP281	19	Male	12	Single	3	3	32973	85	Low	Young
4	KP281	20	Male	13	Partnered	4	2	35247	47	Low	Young
...	...	...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200	High	Old
176	KP781	42	Male	18	Single	5	4	89641	200	High	Old
177	KP781	45	Male	16	Single	5	5	90886	160	High	Old
178	KP781	47	Male	18	Partnered	4	5	104581	120	High	Old
179	KP781	48	Male	18	Partnered	4	5	95508	180	High	Old

180 rows × 11 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
sns.countplot(data = df, x = 'Age_group', hue = 'Product')
plt.xlabel('Age_group')
plt.ylabel('No of customers')
plt.title('Distribution of Age_group')
plt.show()
```



Insights -

1. The most prefered products is KP281 among the younger age groups, but still even KP481 is prefered by many young users and also there are some users who are likely to purchase the KP781 treadmills due to its advance features.
2. Among middle aged groups the KP481 treadmills are mostly prefered as compared to KP281, due to its mid level advantages. only a few middle aged group user prefer KP781.
3. Among the old aged groups the most prefered treamill is KP281, remainig two products are equally prefered.

Conditional and Marginal probabilities

Impact of Gender on purchasing the treadmills

```
crosstab_result = pd.crosstab(index = df['Product'], columns = df['Gender'], margins = True, margins_name = 'Total')
probability_result = (crosstab_result / crosstab_result.loc['Total', 'Total']).round(2)
```

```
probability_result
```

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

Next steps: Generate code with probability_result | View recommended plots | New interactive sheet

Marginal Probabilities

- P(KP281) = 0.44
- P(KP481) = 0.33
- P(KP781) = 0.22
- P(Male) = 0.58
- P(Female) = 0.42

Conditional Probabilities

- P(KP281|Male) = 0.22
- P(KP281|Female) = 0.22
- P(KP481|Male) = 0.17
- P(KP481|Female) = 0.16
- P(KP781|Male) = 0.18
- P(KP781|Female) = 0.04

Impact of Age groups on purchase of the products

```
crosstab_result = pd.crosstab(index = df['Product'], columns = df['Age_group'], margins = True, margins_name = 'Total')
probability_result = (crosstab_result / crosstab_result.loc['Total', 'Total']).round(2)
probability_result
```

Product	Age_group	Young	Middle_aged	Old	Total
KP281		0.29	0.11	0.04	0.44
KP481		0.18	0.12	0.03	0.33
KP781		0.15	0.04	0.03	0.22
Total		0.63	0.28	0.09	1.00

Next steps: Generate code with probability_result | View recommended plots | New interactive sheet

Marginal probabilities

- P(KP281) = 0.44
- p(kp481) = 0.33
- P(KP781) = 0.22
- P(Young) = 0.63
- P(Middle_aged) = 0.28
- P(Old) = 0.09

Conditional Probabilities

1. P(KP281|Young) = 0.29
2. P(KP281|Middle_aged) = 0.11
3. P(KP281|Old) = 0.04
4. P(KP481|Young) = 0.18
5. P(KP481|Middle_aged) = 0.12
6. P(KP481|Old) = 0.03

7. P(KP781|Young) = 0.15
8. P(KP781|Middle_aged) = 0.04
9. P(KP781|Old)= 0.03

Impact of Income groups on purchase of the products

```
crosstab_result = pd.crosstab(index = df['Product'], columns = df['Income_group'], margins = True, margins_name = 'Total')
probability_result = (crosstab_result / crosstab_result.loc['Total', 'Total']).round(2)
probability_result
```

Income_group	Low	Medium	High	Total
Product				
KP281	0.27	0.18	0.00	0.44
KP481	0.17	0.17	0.00	0.33
KP781	0.03	0.08	0.12	0.22
Total	0.46	0.42	0.12	1.00

Next steps: Generate code with probability_result | View recommended plots | New interactive sheet

Marginal probabilities

1. P(KP281) = 0.44
2. p(kp481) = 0.33
3. P(KP781) = 0.22
4. P(Low) = 0.46
5. P(Medium) = 0.42
6. P(High) = 0.12

Conditional Probabilities

1. P(KP281|Low) = 0.27
2. P(KP281|Medium) = 0.18
3. P(KP281|High) = 0.00
4. P(KP481|Low) = 0.17
5. P(KP481|Medium) = 0.17
6. P(KP481|High) = 0.00
7. P(KP781|Low) = 0.03
8. P(KP781|Medium) = 0.08
9. P(KP781|High)= 0.12

Impact of Marital status on purchase of the products

```
crosstab_result = pd.crosstab(index = df['Product'], columns = df['MaritalStatus'], margins = True, margins_name = 'Total')
probability_result = (crosstab_result / crosstab_result.loc['Total', 'Total']).round(2)
probability_result
```

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

Next steps: Generate code with probability_result | View recommended plots | New interactive sheet

Marginal probabilities

1. P(KP281) = 0.44
2. p(kp481) = 0.33
3. P(KP781) = 0.22
4. P(Partnered) = 0.59
5. P(Single) = 0.41

Conditional Probabilities

1. P(KP281|Partnered) = 0.27
2. P(KP281|Single) = 0.18
3. P(KP481|Partnered) = 0.20
4. P(KP481|Single) = 0.13
5. P(KP781|Partnered) = 0.13
6. P(KP781|Single) = 0.09

What is the probability that a customer has a specific fitness level (fitness = 4) given that they purchased a particular treadmill product?

```
# Total number of customers
total=len(df)
products=['KP281','KP481','KP781']
fitness_level=4
#calculating the probablity for each product and fitness Level
probablities={}
for product in products:
  #calculating the number of customers who purchased the specific product
  total_product =len(df.loc[df['Product']==product])
  #calculating the number of customers who purchased the specific product and has fit
  total_product_fitness=len(df.loc[(df['Product']==product)&(df['Fitness']==fitness_level)])
  #calculating the conditional probablity
  conditional_probablity=total_product_fitness/total_product
  #storing the conditional probablity in the dictionary

  probablities[product]=conditional_probablity

for product,probablity in probablities.items():
  print(f'Probablity of customer having a fitness level {fitness_level} given that they have purchased a',product,'is',probablity)
```

```
Probablity of customer having a fitness level 4 given that they have purchased a KP281 is 0.1125
Probablity of customer having a fitness level 4 given that they have purchased a KP481 is 0.13333333333333333
Probablity of customer having a fitness level 4 given that they have purchased a KP781 is 0.175
```

What is the probability that a customer purchased particular treadmill product given that they run 80 miles per week?

```
# Total number of customers
total=len(df)
products=['KP281','KP481','KP781']
miles = 80
#calculating the probablity for each product and fitness Level
probablities={}
for product in products:
  #calculating the number of customers who ran 80 miles per week
  total_miles =len(df.loc[df['Miles']==miles])
  #calculating the number of customers who purchased the specific product and has ran 80 Miles
  total_product_miles=len(df.loc[(df['Product']==product)&(df['Miles']==miles)])
  #calculating the conditional probablity
  conditional_probablity=total_product_miles/total_miles
  #storing the conditional probablity in the dictionary

  probablities[product]=conditional_probablity

for product,probablity in probablities.items():
  print(f'Probablity of customer purchasing a {product} given that they run 80 miles a week is',probablity)
```

```
Probablity of customer purchasing a KP281 given that they run 80 miles a week is 0.0
Probablity of customer purchasing a KP481 given that they run 80 miles a week is 0.0
Probablity of customer purchasing a KP781 given that they run 80 miles a week is 1.0
```

Heatmap

```
mod_df = df.select_dtypes(include=['number'])
corr1 = mod_df.corr()
sns.heatmap(corr1,annot=True)
plt.show()
```

1. Age and Education:There is a positive correlation of approximately 0.28 between Age and Education. This indicates that as the customers' age increases, their education level tends to be higher.
2. Age and Income: There is a moderate positive correlation of approximately 0.51 between Age and Income. This suggests that as the customers' age increases, their income tends to be higher.
3. Education and Income: There is a relatively strong positive correlation of approximately 0.63 between Education and Income. This suggests that customers with higher levels of education tend to have higher incomes.
4. Usage and Fitness: There is a strong positive correlation of approximately 0.67 between Usage and Fitness. This indicates that customers who plan to use the treadmill more frequently tend to have higher fitness levels.
5. Fitness and Miles: There is a strong positive correlation of approximately 0.79 between Fitness and Miles. This indicates that customers with higher fitness levels also expect to walk/run more miles per week

Insights & Recommendations

1. Since most of the customers who bought KP781 are males we can say that these trademills are best suited for males and not for females, but still Aerofit has to identify why this product is not being purchased by female customers and has to introduce these features to the female customers and explain them its benifits.
2. Customers who tends to have less educational experience do not buy KP781 Treadmills. However this customers should be targeted and more accurate products according to their understanding levels should be suggested.
3. Customers with fitness level less than 3 Shouls not go KP781 treadmills
4. The majority of Aerofit customers are in the income range of 40000 to 60000, they have the highlest probability of purchasing the products. but we dont see the customers in the income range of more than 80000 purchasing the products, the probability is too less.
5. The majority of the customers of Aerofit belongs to the age group of 19 - 35 years old, and there are very few customers who are more than 35 to 40 years old.
6. Customers having education experience of 14 or 16 years mostly prefer to buy KP281 and KP481 treadmills. However again the most prefered treadmill is KP281.
7. Customers having education of 18 years also tend to buy the most expensive treadmill that is KP781 due its advance features.
8. Emphasize the budget-friendly nature of the KP281 treadmill to attract more customers.
9. Engage more with the gyms and fitness communities to showcase KP481 Features and advantages which can be most prefered choice in future and can increase the focus from KP281 to KP481.
10. Marketing campaigns should be launched to increase more interests in KP781 treadmills which focuses on more advance featured targeting the audience that is more in Technology.
11. All the justifications and points should be highlighted in more appropriate ways to justify the expensive price for KP781.

```
print('.\n\n\n\n]n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\\n\n\n\n\n\\n\n.')
```

```
]n

\n

\n
.
```

```
print('.\n\n\n\n]n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\\n\n\n\n\n\\n\n.')
```

```
]n
```

```
\n


\n
.
```

```python
print('.\n\n\n\n]n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\\n\n\n\n\n\\n\n.')
```

```
]n
```

```
\n

\n
.
```

```
print('.\n\n\n\n]n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\\n\n\n\n\n\\n\n.')
```

```
]n

\n

\n
.
```

```
print('.\n\n\n\n]n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\\n\n\n\n\n\\n\n.')
```