BUSINESS CASE:

Aerofit Business case study

- 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.
- 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.
- Aerofit Business Case Study will be performed using descriptive analytics and several python libraries like numpy,pandas,matplotlib, seaborn to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts.
- This tabular dataset consists of listings of three types of Threadmills, usage status among different customers along with details of customers such as – age, gender, education, marital status, income, self rated fitness scale, average number of miles the customer is expected to walk/run each week.
- The following data is available in a single csv file

● Product: Product Purchased KP281, KP481, or KP781

● Age: In years
● Gender: Male/Female
● Education: in years
● Marital Status: single or partnered
● Usage: 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 poor shape and 5 is the excellent shape.
● Miles: 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 has advanced features that sell for $2,500.

We will be exploring its correlation with the profile of the buyer. We have details of the buyers such as their age, gender, marital status etc. The details we will be exploring is on categorisation of Buyers, the adjoining probabilities of buying the product and make our inferences.

The questions under line of analysis, Google Colab Notebook commands along with a screenshot of the output, identifying connection between input variables and output variables are submitted below along with the valuable insights that I drew from my analysis and a few actionable recommendations are submitted below.

BASIC ANALYSIS **includes to perform descriptive analysis to create a customer profile and constructing two-way contingency tables for each AeroFit treadmill product and compute all conditional and marginal probabilities and their insights/impact on the business.**

What does 'good' look like?

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

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

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

```
data_path="https://d2beiqkhq929f0.cloudfront.net/public_assets/as
sets/000/001/125/original/aerofit_treadmill.csv"
df=pd.read_csv(data_path)
```

```
df
```

```
Number of rows: 180
Number of columns: 9
```

```
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

b) You can find the number of rows and columns given in the dataset

```
[7] 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
```

```
[8] df.shape
```

```
(180, 9)
```

```
[10] 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

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

```
[11] df.isnull().any()
```

```
Product          False
Age              False
Gender           False
Education        False
MaritalStatus    False
Usage            False
Fitness          False
Income           False
Miles            False
dtype: bool
```

```
[12] df.isna().sum()
```

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

```
[13] df.isnull().sum()/len(df)*100
```

```
Product          0.0
Age              0.0
Gender           0.0
Education        0.0
MaritalStatus    0.0
Usage            0.0
Fitness          0.0
Income           0.0
Miles            0.0
dtype: float64
```

```
df.duplicated()
```

```
0      False
1      False
2      False
3      False
4      False
       ...
175    False
176    False
177    False
178    False
179    False
Length: 180, dtype: bool
```

```
df["Product"].unique()
```

```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

Observations :

- There are no missing values in the data.
- There are no duplicate values in the data.
- Number of rows is 180 and number of columns is 90
- There are 3 unique products in the dataset.
- Minimum & Maximum age of the person is 18 & 50, mean is 28.79 and 75% of persons have age less than or equal to 33.
- Most of the people are having 16 years of education i.e. 75% of persons are having education <= 16 years.
- Standard deviation for **Income** & **Miles** is very high. These variables might have the outliers in it.

NON GRAPHICAL ANALYSIS – VALUE COUNTS :

```
[15] df.Product.value_counts()

KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

```
[16] df.MaritalStatus.value_counts()

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

```
[105] df.Education.value_counts()

16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

```
df.groupby(["Gender"])["Age"].mean()
4]
Gender
Female    28.565789
Male      28.951923
Name: Age, dtype: float64
```

```
5] df.groupby(["Gender","Product"])["Age"].mean()

Gender  Product
Female  KP281      28.450000
        KP481      29.103448
        KP781      27.000000
Male    KP281      28.650000
        KP481      28.709677
        KP781      29.545455
Name: Age, dtype: float64
```

```
6] df.groupby(["Gender","Product"])["Income"].mean()

Gender  Product
Female  KP281      46020.075000
        KP481      49336.448276
        KP781      73633.857143
Male    KP281      46815.975000
        KP481      48634.258065
        KP781      75825.030303
Name: Income, dtype: float64
```

```
[19] df.groupby(["Gender","Product"])["Product"].value_counts()
```

```
Gender  Product  Product
Female  KP281    KP281      40
        KP481    KP481      29
        KP781    KP781       7
Male    KP281    KP281      40
        KP481    KP481      31
        KP781    KP781      33
Name: Product, dtype: int64
```

```
df.groupby(["Fitness","Product"])["Product"].value_counts()
```

```
Fitness  Product  Product
1        KP281    KP281       1
         KP481    KP481       1
2        KP281    KP281      14
         KP481    KP481      12
3        KP281    KP281      54
         KP481    KP481      39
         KP781    KP781       4
4        KP281    KP281       9
         KP481    KP481       8
         KP781    KP781       7
5        KP281    KP281       2
         KP781    KP781      29
Name: Product, dtype: int64
```

```
[29] df.groupby(["MaritalStatus"])["Miles"].mean()
```

```
MaritalStatus
Partnered    104.289720
Single       101.589041
Name: Miles, dtype: float64
```

```
[30] df.groupby(["MaritalStatus","Usage"])["Miles"].mean()
```

```
MaritalStatus  Usage
Partnered      2         57.000000
               3         79.400000
               4        126.482759
               5        160.111111
               6        228.000000
               7        240.000000
Single         2         61.636364
               3         88.965517
               4        109.434783
               5        161.375000
               6        175.000000
Name: Miles, dtype: float64
```

```
[31] df.groupby(["Product"])["Usage"].mean()
```

```
Product
KP281    3.087500
KP481    3.066667
KP781    4.775000
Name: Usage, dtype: float64
```

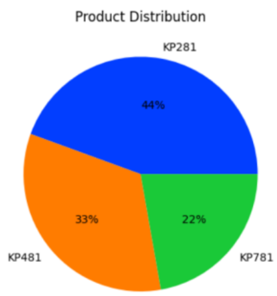
```
df.groupby(["Gender","MaritalStatus","Product"])["Miles"].aggregate([np.mean,np.median]).reset_index()
```

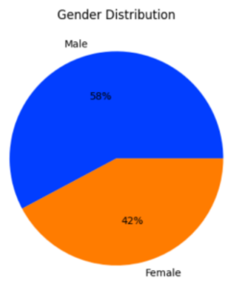
	Gender	MaritalStatus	Product	mean	median
0	Female	Partnered	KP281	74.925926	66.0
1	Female	Partnered	KP481	94.000000	85.0
2	Female	Partnered	KP781	215.000000	200.0
3	Female	Single	KP281	78.846154	75.0
4	Female	Single	KP481	80.214286	79.5
5	Female	Single	KP781	133.333333	100.0
6	Male	Partnered	KP281	80.190476	75.0
7	Male	Partnered	KP481	87.238095	95.0
8	Male	Partnered	KP781	176.315789	160.0
9	Male	Single	KP281	99.526316	94.0
10	Male	Single	KP481	91.100000	95.0
11	Male	Single	KP781	147.571429	150.0

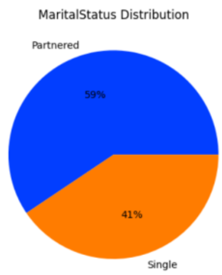
```
product_revenue =
df.groupby(['Product'])['Price'].sum().reset_index().rename(colum
ns= {'Product':"Product", "Price":'Product_revenue'})
product_revenue
```

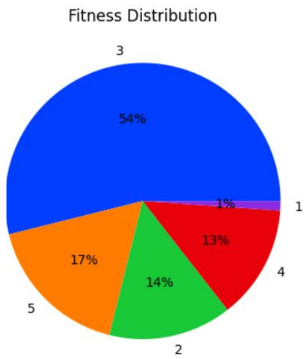
	Product	Product_revenue
0	KP281	120000
1	KP481	105000
2	KP781	100000

```
palette_color = sns.color_palette('bright')
title = ['Product Distribution', 'Gender Distribution',
'MaritalStatus Distribution', 'Fitness Distribution']
for i, col in enumerate(['Product', 'Gender', 'MaritalStatus',
'Fitness']):
product_count = df[col].value_counts().reset_index()
data = list(product_count[col])
keys = list(product_count['index'])
plt.pie(data, labels=keys, colors=palette_color,
autopct='%.0f%%')
plt.title(title[i])
plt.show()
print('-'*100)
```









Observations: The following are some observations:

1. More than 50% of the consumers possess fitness level of 3 (95 of 180), out of which, more than 50% prefer KP281 (54 of 95) followed by KP481. Meanwhile, those who have fitness level of 5 prefer the KP781 (29 of 31). All other categories prefer the cheapest treadmills over expensive ones.

2. Average income among the consumers of different treadmills have similar trends for both males and females. The obvious trend of high income people are the ones to buy most expensive items

3. While male consumers (88.4 miles) tend to run slightly more on average on KP481 treadmills than their female counterpart (87.3 miles), the difference is more significant for KP281 and KP781. Males, on an average, run 89.3 miles compared to 76.2 miles for females. The trend of males running more on average than females is not followed for KP781, male consumers run 164.1 miles on average on KP781 whereas females run 180 miles.

4. Single Females (27) are greater consumers of KP281 than their male counterparts (21). Meanwhile, single male consumers purchase more of KP481 (21 for males and 15 for females) and KP781(19 for males and 4 for females) than single females.

5. Partnered males (19) are, however, purchase more KP281 treadmills than female partnered consumers (13). Similar trend is observed in KP781 sales, where partnered male purchased 14 treadmills compared to 3 purchased by partnered females. But for KP481, female partnered (14) purchased more than male counterparts (10).

6. Partnered females, who purchased KP481 (94 miles vs 80.2 miles for singles) and KP781 (215 miles vs 133.3 miles for singles) run, on an average, more than single counterpart. But Single females who purchased KP281 (78.84 miles)have ran slightly more than partnered ones (74.92 miles).

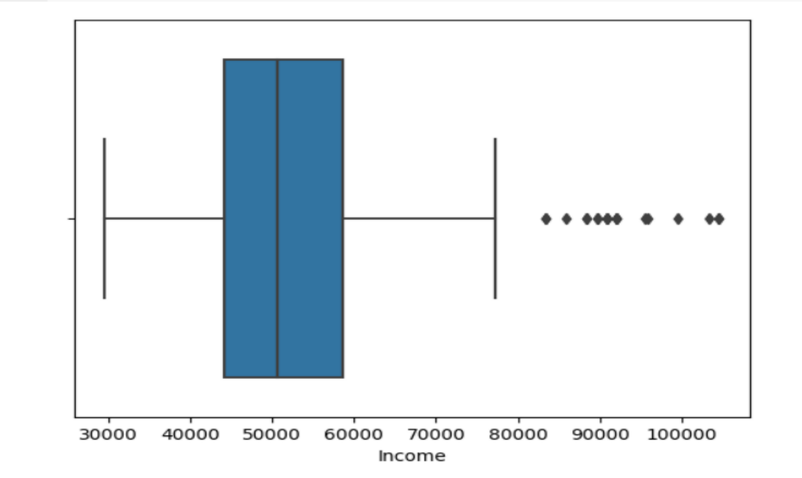
7. Single males, who purchased KP281 (99.5 miles vs 80 miles for partnered) and KP481 (91.1 miles vs 87.2 miles for partnered) run, on an average, more than partnered counterpart. But Partnered males who purchased KP781 (176.3 miles)have ran slightly more than single ones (147.8 miles).

8. The total revenue obtained on KP281 is $120000, KP481 is $105000 and KP781 is $100000.

2. Detect Outliers

a) Find the outliers for every continuous variable in the dataset

```
sns.boxplot(data=df,x="Income")
plt.show()
```



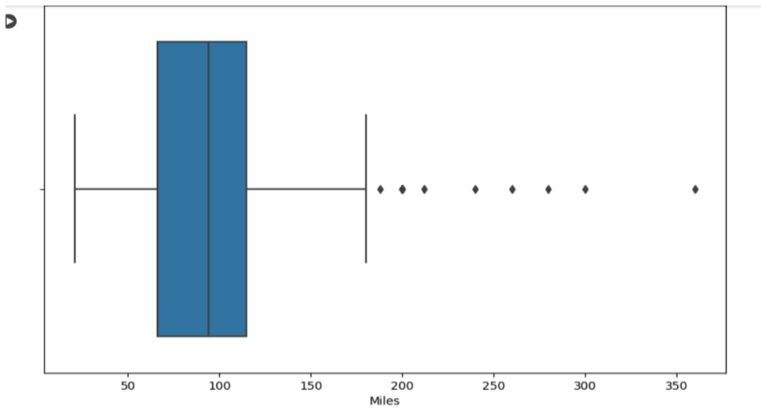

```
Q1=np.percentile(df["Income"],25)
Q2=np.percentile(df["Income"],50)
Q3=np.percentile(df["Income"],75)
IQR=Q3-Q1
lower_whisker=Q1-1.5*IQR
upper_whisker=Q3+1.5*IQR
outliers=df[df["Income"]>upper_whisker]
print(" Quartile 1: {}\n Quartile 2: {}\n Quartile 3: {}\n IQR is
: {}\n Lower_whisker(Income): {}\n Upper_whisker(Income): {}
".format(Q1,Q2,Q3,IQR,lower_whisker,upper_whisker))
```

```
Quartile 1: 44058.75
Quartile 2: 50596.5
Quartile 3: 58668.0
IQR is : 14609.25
Lower_whisker(Income): 22144.875
Upper_whisker(Income): 80581.875
```

```
[39] len(outliers)
```

```
19
```

```
fig=plt.figure(figsize=(10,6))
sns.boxplot(data=df,x="Miles")
plt.show()
```




```
Q1=np.percentile(df["Miles"],25)
Q2=np.percentile(df["Miles"],50)
Q3=np.percentile(df["Miles"],75)
IQR=Q3-Q1
low_whisker=Q1-1.5*IQR
up_whisker=Q3+1.5*IQR
outliers=df[df["Miles"]>up_whisker]
outliers1=df[df["Miles"]<low_whisker]
print(" Q1: {}\n Q2: {}\n Q3: {}\n IQR(Miles): {}\n low_whisker:
{}\n up_whisker: {}".format(Q1,Q2,Q3,IQR,low_whisker,up_whisker))
```

```
Q1: 66.0
Q2: 94.0
Q3: 114.75
IQR(Miles): 48.75
low_whisker: -7.125
up_whisker: 187.875
```

```
len(outliers)
```

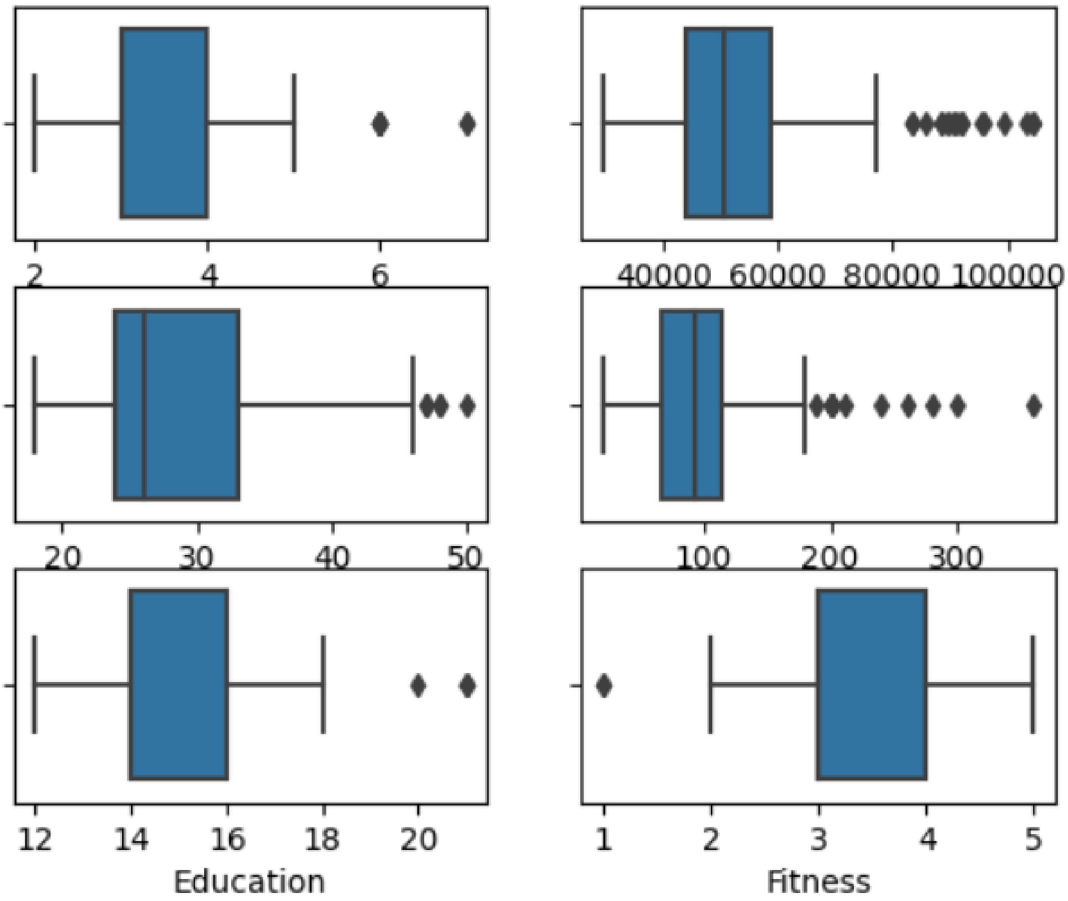
```
13
```

THE OUTLIERS IN MILES COLUMN ARE SHOWN BELOW

```
[50] outliers
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Price	age_bins
23	KP281	24	Female	16	Partnered	5	5	44343	188	1500	18-28
84	KP481	21	Female	14	Partnered	5	4	34110	212	1750	18-28
142	KP781	22	Male	18	Single	4	5	48556	200	2500	18-28
148	KP781	24	Female	16	Single	5	5	52291	200	2500	18-28
152	KP781	25	Female	18	Partnered	5	5	61006	200	2500	18-28
155	KP781	25	Male	18	Partnered	6	5	75946	240	2500	18-28
166	KP781	29	Male	14	Partnered	7	5	85906	300	2500	28-38
167	KP781	30	Female	16	Partnered	6	5	90886	280	2500	28-38
170	KP781	31	Male	16	Partnered	6	5	89641	260	2500	28-38
171	KP781	33	Female	18	Partnered	4	5	95866	200	2500	28-38
173	KP781	35	Male	16	Partnered	4	5	92131	360	2500	28-38
175	KP781	40	Male	21	Single	6	5	83416	200	2500	38-48
176	KP781	42	Male	18	Single	5	4	89641	200	2500	38-48

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(6,4))
fig.subplots_adjust(top=2.0)
sns.boxplot(data=df,x="Usage",orient='h',ax=axis[0,0])
sns.boxplot(data=df,x="Income",orient='h',ax=axis[0,1])
sns.boxplot(data=df,x="Age",orient='h',ax=axis[1,0])
sns.boxplot(data=df,x="Miles",orient='h',ax=axis[1,1])
sns.boxplot(data=df,x="Education",orient='h',ax=axis[2,0])
sns.boxplot(data=df,x="Fitness",orient='h',ax=axis[2,1])
plt.show()
```



OBSERVATION : From the boxplots it is quite clear that:

- **Age**, **Education** and **Usage** are having very few outliers.
- While **Income** and **Miles** are having more outliers. Income has 19 outliers and Miles have 13 outliers.

b) Remove/clip the data between the 5 percentile and 95 percentile

```
high=np.percentile(df["Miles"],95)
low=np.percentile(df["Miles"],5)
df1=df["Miles"].clip(high,low)
df1
```

```
[45] high=np.percentile(df["Miles"],95)
     low=np.percentile(df["Miles"],5)
     df1=df["Miles"].clip(high,low)
     df1

     0       112
     1        75
     2        66
     3        85
     4        47
            ...
     175     200
     176     200
     177     160
     178     120
     179     180
     Name: Miles, Length: 180, dtype: int64
```

```
a=np.percentile(df["Income"],5)
b=np.percentile(df["Income"],95)
c=df["Income"].clip(a,b)
c
```

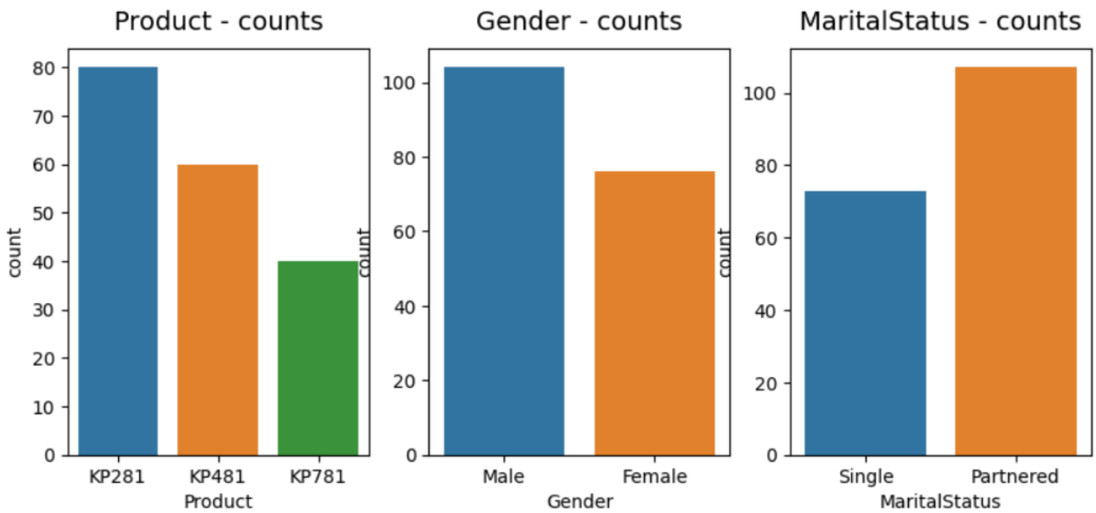
```
a=np.percentile(df["Income"],5)
b=np.percentile(df["Income"],95)
c=df["Income"].clip(a,b)
c
```

```
0      34053.15
1      34053.15
2      34053.15
3      34053.15
4      35247.00
         ...
175    83416.00
176    89641.00
177    90886.00
178    90948.25
179    90948.25
Name: Income, Length: 180, dtype: float64
```

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

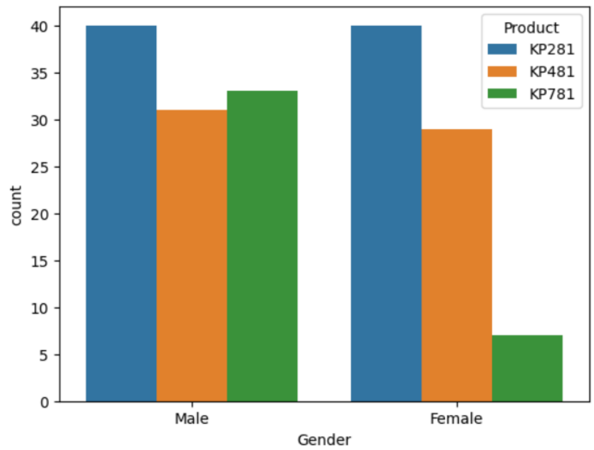
a) Find if there is any relationship between the categorical variables and the output variable in the data

```
fig, axs = plt.subplots(nrows=1, ncols=4, figsize=(10,4))
sns.countplot(data=df, x='Product', ax=axs[0])
sns.countplot(data=df, x='Gender', ax=axs[1])
sns.countplot(data=df, x='MaritalStatus', ax=axs[2])
axs[0].set_title("Product - counts", pad=10, fontsize=14)
axs[1].set_title("Gender - counts", pad=10, fontsize=14)
axs[2].set_title("MaritalStatus - counts", pad=10, fontsize=14)
plt.show()
```



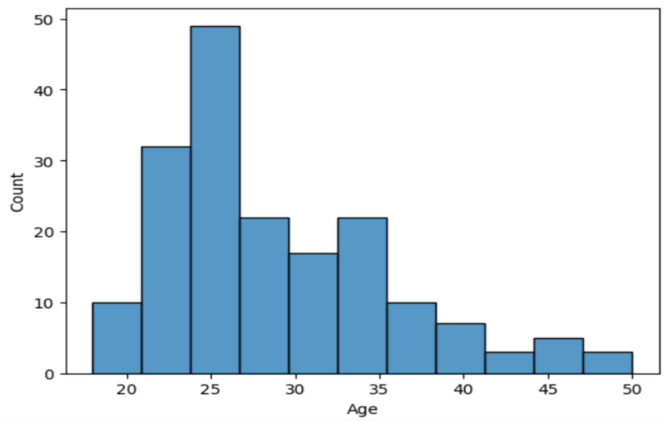
```python
sns.countplot(data=df,x="Gender",hue="Product")
```

<Axes: xlabel='Gender', ylabel='count'>



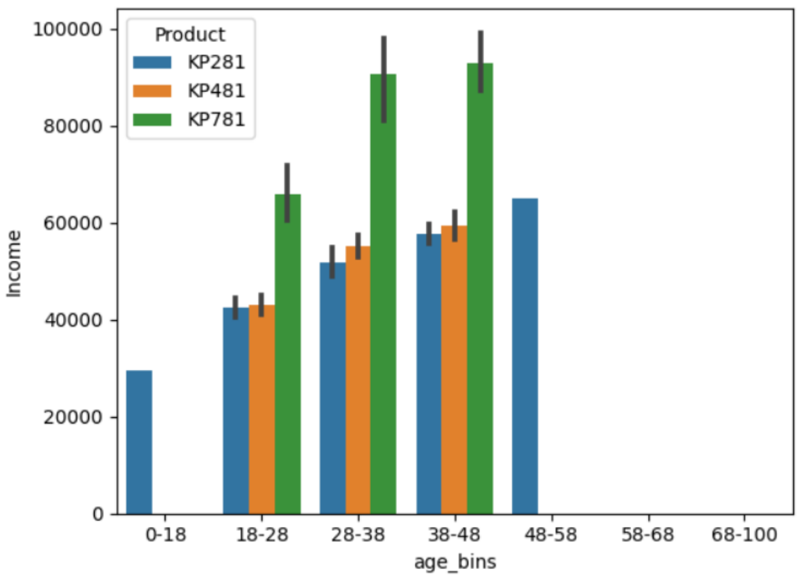

```python
sns.histplot(data=df,x="Age")
```

<Axes: xlabel='Age', ylabel='Count'>



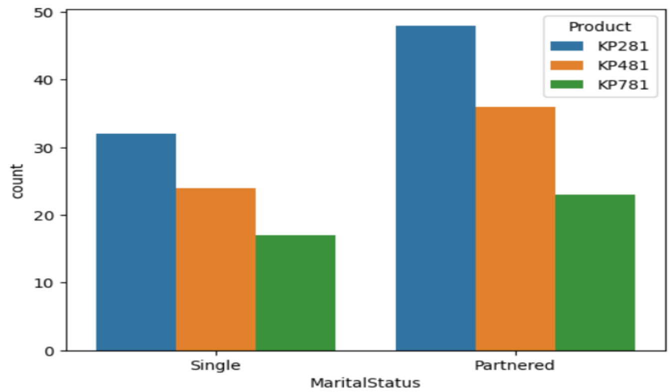

```python
df['age_bins']=pd.cut(x=df['Age'],bins=[0,18,28,38,48,58,68,100],
labels=['0-18','18-28','28-38','38-48','48-58','58-68','68-100'])
sns.barplot(data=df,x="age_bins",y="Income",hue="Product")
```

<Axes: xlabel='age_bins', ylabel='Income'>




```python
sns.countplot(data=df,x="MaritalStatus",hue="Product")
```

<Axes: xlabel='MaritalStatus', ylabel='count'>




```
pd.crosstab(df['Product'],[df['MaritalStatus'],df['Gender']],
margins=True)
```

MaritalStatus	Partnered		Single		All
Gender	Female	Male	Female	Male	
Product					
KP281	27	21	13	19	80
KP481	15	21	14	10	60
KP781	4	19	3	14	40
All	46	61	30	43	180

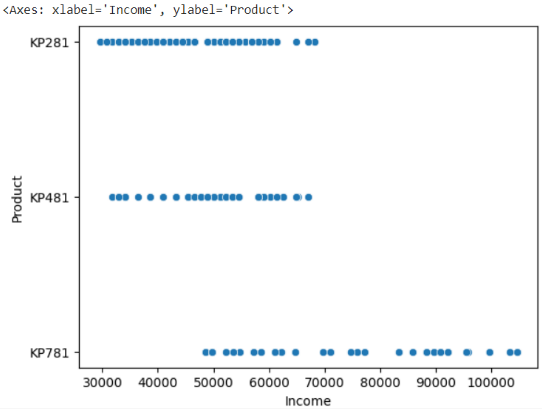
Observations :

- **KP281** is the most frequent product.
- There are more Males in the data than Females.
- More **Partnered** persons are there in the data.
- KP281 is most popular among Partnered Females
- KP481 is most popular among Partnered Males
- KP781 is most popular among Partnered Males
- KP281 is the top selling product and KP781 is the lowest selling product
- KP281 is the highest revenue generating product
- Customers whose age lies between 25-30, are more likely to buy KP781 product

b) Find if there is any relationship between the continuous variables and the output variable in the data.

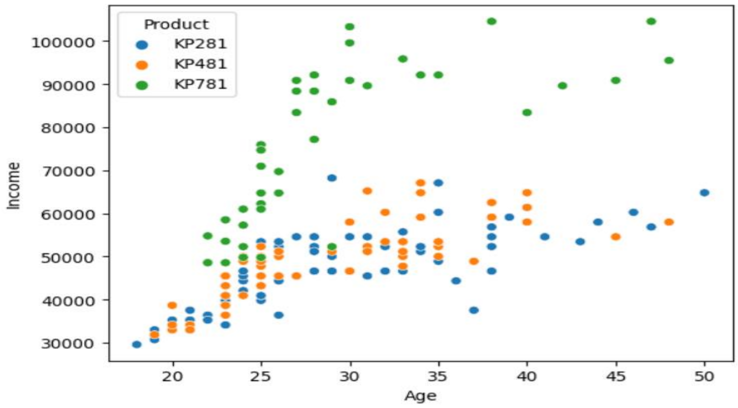
```
sns.scatterplot(x="Income",y="Product",data=df)
```

<Axes: xlabel='Income', ylabel='Product'>



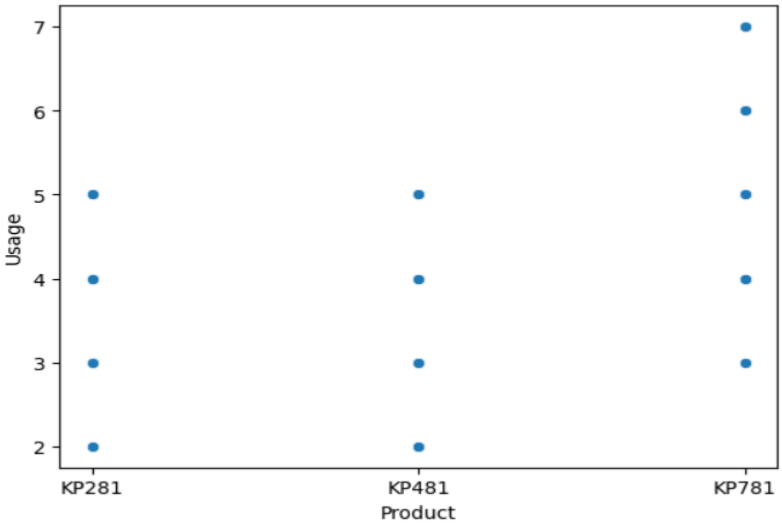
```
sns.scatterplot(x="Age",y="Income",hue="Product",data=df)
```

<Axes: xlabel='Age', ylabel='Income'>



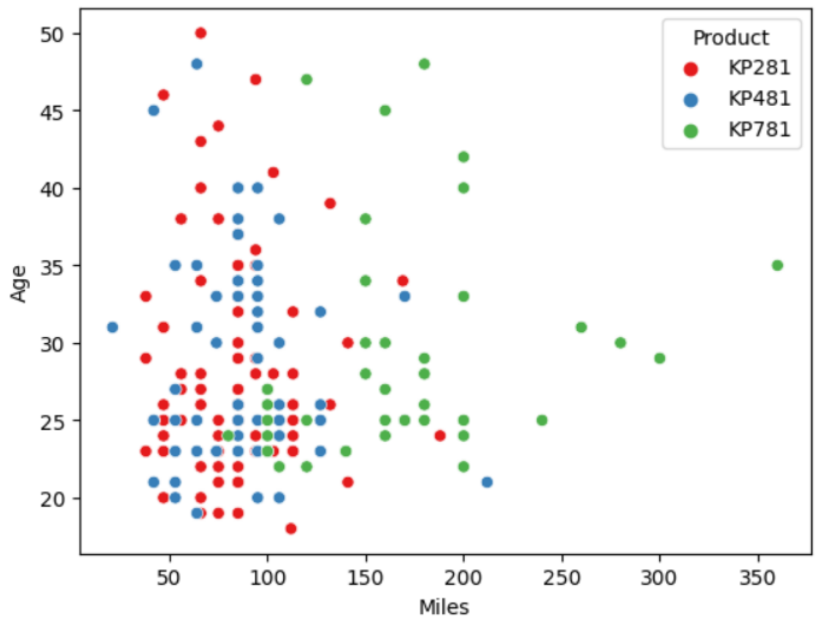

```
sns.scatterplot(x="Product",y="Usage",data=df)
```

<Axes: xlabel='Product', ylabel='Usage'>



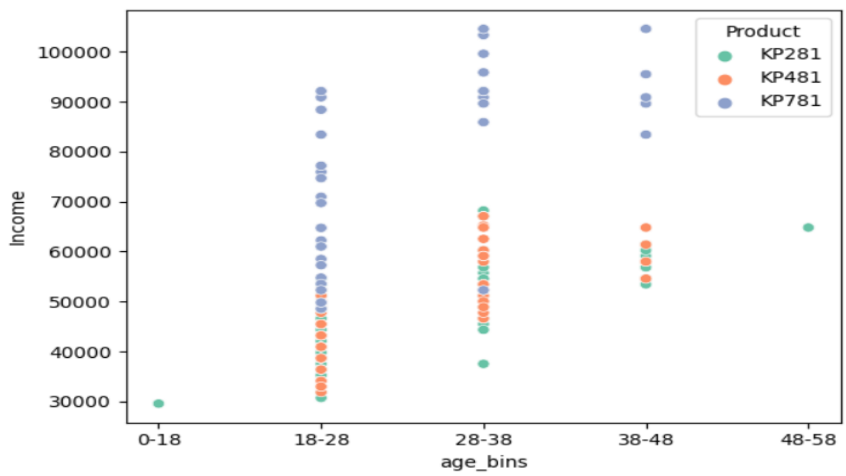

```
sns.scatterplot(x="Miles",y="Age",hue="Product",data=df,palette="
Set1")
```

```
<Axes: xlabel='Miles', ylabel='Age'>
```




```
sns.scatterplot(x="age_bins",y="Income",hue="Product",data=df,palette="Set2")
```

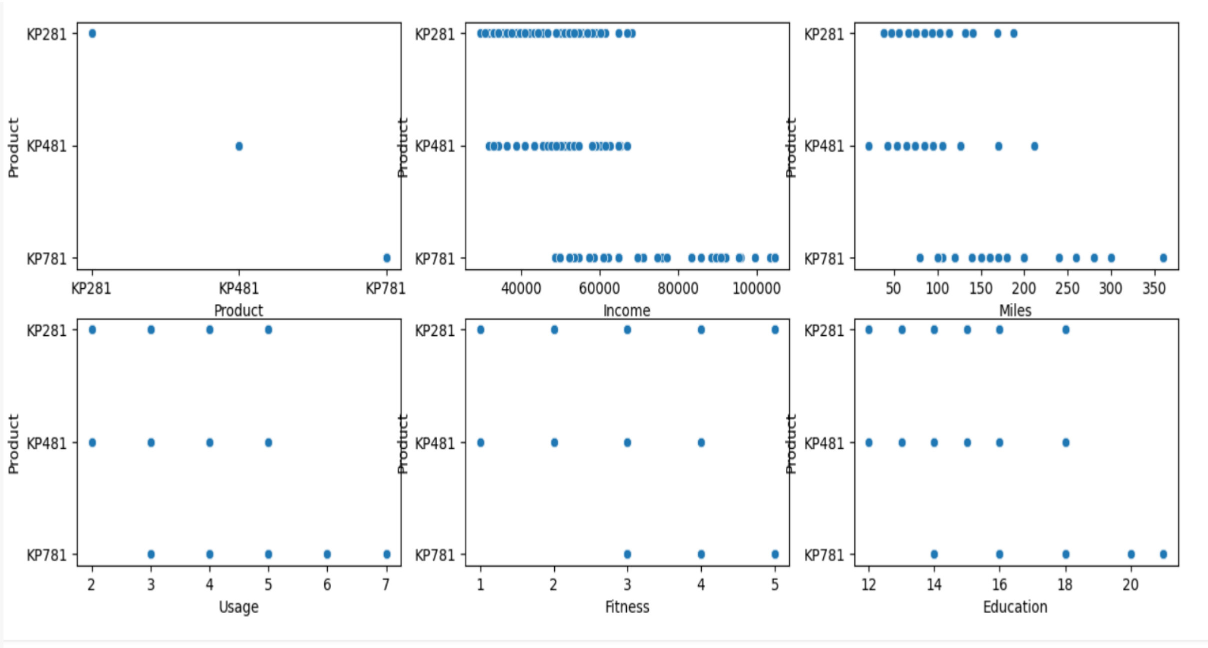
```
<Axes: xlabel='age_bins', ylabel='Income'>
```



Observations:

- High income customers are buying advanced product.
- The usage of KP781 is wide.
- Low and average income group is widely using KP281.
- 20-35 years age group is widely using this equipment.

```
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(15,6))
sns.scatterplot(data=df, x='Product',y="Product", ax=axs[0,0])
sns.scatterplot(data=df, x='Income', y="Product",ax=axs[0,1])
sns.scatterplot(data=df, x='Miles',y="Product",ax=axs[0,2])
sns.scatterplot(data=df, x='Usage',y="Product", ax=axs[1,0])
sns.scatterplot(data=df, x='Education',y="Product", ax=axs[1,2])
sns.scatterplot(data=df, x='Fitness',y="Product", ax=axs[1,1])
plt.show()
```



4. Representing the Probability

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

Marginal Distribution is as follows :

```
df["Product"].value_counts(normalize=True)
```

```
KP281    0.444444
KP481    0.333333
KP781    0.222222
Name: Product, dtype: float64
```

Observations :

Marginal probability :
--> **44.44%** of customers purchased **KP281**
--> **33.33%** of customers purchased **KP481**
--> **22.22%** of customers purchased **KP781**

b) Find the probability that the customer buys a product based on each column.

Probability of Product purchase with respect to gender:

```
pd.crosstab(df['Product'],[df['Gender']], normalize=True,
margins=True, margins_name='Total').round(2)
```

Gender	Female	Male	Total
Product			
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

Observations :

- The **Probability** of a treadmill being purchased by a **female is 42%**.
- The **Probability** of a treadmill being purchased by a **male is 58%**.

Probability of Product purchase with respect to MaritalStatus:

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

MaritalStatus	Partnered	Single	All
Product			
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

Observations :

- The **Probability** of a treadmill being purchased by a **Married/partnered Customer is 59%**.
- The **Probability** of a treadmill being purchased by a **Single Customer is 41%**.

Probability of Product purchase with respect to Usage:

```
print((pd.crosstab(index=df["Product"],columns=df["Usage"],margin
s=True,margins_name="Total",normalize=True)*100).round(2))
```

Usage	2	3	4	5	6	7	Total
Product							
KP281	10.56	20.56	12.22	1.11	0.00	0.00	44.44
KP481	7.78	17.22	6.67	1.67	0.00	0.00	33.33
KP781	0.00	0.56	10.00	6.67	3.89	1.11	22.22
Total	18.33	38.33	28.89	9.44	3.89	1.11	100.00

Observations :

- The **Probability** of a treadmill being purchased by a customer with **Usage 3 per week is 38%**.

- The **Probability** of a treadmill being purchased by a customer with **Usage 4 per week is 29%**.
- The **Probability** of a treadmill being purchased by a customer with **Usage 2 per week is 18%**
- The **Probability** of treadmill being purchase by a customer with **Usage 5,6 and 7** per week is comparatively very less i.e., **9.44%,3.89% and 1.11%** respectively

Probability of Product purchase with respect to Fitness:

```
print((pd.crosstab(index=df["Fitness"],columns=df["Product"],margins=True,margins_name="Total",normalize=True)*100).round(2))
```

```
Product   KP281  KP481  KP781   Total
Fitness
1          0.56   0.56   0.00    1.11
2          7.78   6.67   0.00   14.44
3         30.00  21.67   2.22   53.89
4          5.00   4.44   3.89   13.33
5          1.11   0.00  16.11   17.22
Total     44.44  33.33  22.22  100.00
```

Observations :

- The **Probability** of a treadmill being purchased by a customer with **Average(3) Fitness is 53.89%**.
- The **Probability** of a treadmill being purchased by a customer with **Fitness of 2,4,5 is almost 15%**.
- The **Probability** of a treadmill being purchased by a customer with **very low(1) Fitness is only 1.11%**.

Probability of Product purchase with respect to Income Range:

```
print(pd.crosstab(index=pd.cut(final_df["Income"],[20000,30000,40000,50000,60000,70000,80000],include_lowest=True,right=True),columns=final_df['Product'],margins=True, normalize=True))
```

```
Product                 KP281     KP481     KP781       All
Income
(19999.999, 30000.0]  0.006211  0.000000  0.000000  0.006211
(30000.0, 40000.0]    0.136646  0.055901  0.000000  0.192547
(40000.0, 50000.0]    0.155280  0.130435  0.031056  0.316770
(50000.0, 60000.0]    0.161491  0.142857  0.037267  0.341615
(60000.0, 70000.0]    0.037267  0.043478  0.037267  0.118012
(70000.0, 80000.0]    0.000000  0.000000  0.024845  0.024845
All                   0.496894  0.372671  0.130435  1.000000
```

Observations :

- The **Probability** of a treadmill being purchased by a customer with **income in range 50000-60000 is high i.e, 34.16% and 40000-50000 income range,probability is 31.6 %**.

- The **Probability** of a treadmill being purchased by a customer **with low average income** is very low i.e**., 0.006%.**

Probability of Product purchase with respect to Education:

```
pd.crosstab(df['Education'], df['Product'], margins=True,
normalize=True)
```

Product Education	KP281	KP481	KP781	All
12	0.011111	0.005556	0.000000	0.016667
13	0.016667	0.011111	0.000000	0.027778
14	0.166667	0.127778	0.011111	0.305556
15	0.022222	0.005556	0.000000	0.027778
16	0.216667	0.172222	0.083333	0.472222
18	0.011111	0.011111	0.105556	0.127778
20	0.000000	0.000000	0.005556	0.005556
21	0.000000	0.000000	0.016667	0.016667
All	0.444444	0.333333	0.222222	1.000000

Observations :

- Product KP281 and KP481 is more popular among customers who have education between 14 to 16. The **Probability** of customer purchasing treadmill is high for the education range of 16 i.e**., 47.22%.**
- **The Probability** of customer purchasing treadmill is low for the education range of 20 which is **0.005%.**

Probability of Product purchase with respect to Age:

```
pd.crosstab(df['Product'],[df['age_bins']], normalize=True,
margins=True, margins_name='Total').round(2)
```

age_bins Product	0-18	18-28	28-38	38-48	48-58	Total
KP281	0.01	0.27	0.12	0.04	0.01	0.44
KP481	0.00	0.18	0.13	0.03	0.00	0.33
KP781	0.00	0.14	0.06	0.03	0.00	0.22
Total	0.01	0.59	0.31	0.09	0.01	1.00

Observations :

- **The Probability of low age group** customers i.e., **18-28** buying a treadmill is high which is **59% and the Probability** of age group **28-38** purchasing is **31%**

- **The Probability** of the **other age groups** buying a treadmill is **very low.**

Probability of Product purchase with respect to Miles:

```
print(pd.crosstab(index=pd.cut(df["Miles"],[5,50,100,150,200,250,
300,350,400],include_lowest=True,right=True),
columns=df['Product'],margins=True, normalize=True))
```

```
Product             KP281     KP481     KP781       All
Miles
(4.999, 50.0]    0.066667  0.027778  0.000000  0.094444
(50.0, 100.0]    0.277778  0.216667  0.044444  0.538889
(100.0, 150.0]   0.088889  0.072222  0.050000  0.211111
(150.0, 200.0]   0.011111  0.011111  0.100000  0.122222
(200.0, 250.0]   0.000000  0.005556  0.005556  0.011111
(250.0, 300.0]   0.000000  0.000000  0.016667  0.016667
(350.0, 400.0]   0.000000  0.000000  0.005556  0.005556
All              0.444444  0.333333  0.222222  1.000000
```

Observations :

- **The Probability** of persons walking between **100-150 miles** purchasing a treadmill is high which is **21.1%.**
- **The Probability of** persons walking between **350-400 miles** purchasing a treadmill is low which is **0.005%.**

c) Find the conditional probability that an event occurs given that another event has occurred. (Example: given that a customer is female, what is the probability she'll purchase a KP481)

Conditional Probability of Product purchase with respect to Gender:

```
print("P(Product|Gender): ")
print(pd.crosstab(index=df['Gender'],columns=df['Product'],
normalize="index"))
print()
print("P(Gender|Product): ")
print(pd.crosstab(index=df['Gender'],columns=df['Product'],
normalize="columns"))
```

```
P(Product|Gender):
Product      KP281     KP481     KP781
Gender
Female    0.526316  0.381579  0.092105
Male      0.384615  0.298077  0.317308

P(Gender|Product):
Product  KP281     KP481  KP781
Gender
Female     0.5  0.483333  0.175
Male       0.5  0.516667  0.825
```

Conditional Probability of Product purchase with respect to Education :

```
print("P(Product|Education): ")
print(pd.crosstab(index=df['Education'],columns=df['Product'],
normalize="index"))
print()
print("P(Education|Product): ")
print(pd.crosstab(index=df['Education'],columns=df['Product'],
normalize="columns"))
```

```
P(Product|Education):
Product       KP281     KP481     KP781
Education
12         0.666667  0.333333  0.000000
13         0.600000  0.400000  0.000000
14         0.545455  0.418182  0.036364
15         0.800000  0.200000  0.000000
16         0.458824  0.364706  0.176471
18         0.086957  0.086957  0.826087
20         0.000000  0.000000  1.000000
21         0.000000  0.000000  1.000000

P(Education|Product):
Product     KP281     KP481  KP781
Education
12         0.0250  0.016667  0.000
13         0.0375  0.033333  0.000
14         0.3750  0.383333  0.050
15         0.0500  0.016667  0.000
16         0.4875  0.516667  0.375
18         0.0250  0.033333  0.475
20         0.0000  0.000000  0.025
21         0.0000  0.000000  0.075
```

Conditional Probability of Product purchase with respect to Usage :

```
print("P(Product|Usage): ")
print(pd.crosstab(index=df['Usage'],columns=df['Product'],
normalize="index"))
print()
print("P(Usage|Product): ")
print(pd.crosstab(index=df['Usage'],columns=df['Product'],
normalize="columns"))
```

```
P(Product|Usage):
Product     KP281     KP481     KP781
Usage
2        0.575758  0.424242  0.000000
3        0.536232  0.449275  0.014493
4        0.423077  0.230769  0.346154
5        0.117647  0.176471  0.705882
6        0.000000  0.000000  1.000000
7        0.000000  0.000000  1.000000

P(Usage|Product):
Product   KP281     KP481  KP781
Usage
2        0.2375  0.233333  0.000
3        0.4625  0.516667  0.025
4        0.2750  0.200000  0.450
5        0.0250  0.050000  0.300
6        0.0000  0.000000  0.175
7        0.0000  0.000000  0.050
```

Conditional Probability of Product purchase with respect to Fitness :

```
print("P(Product|Fitness): ")
print(pd.crosstab(index=df['Fitness'],columns=df['Product'],margi
ns=True, normalize="index"))
print()
print("P(Fitness|Product): ")
print(pd.crosstab(index=df['Fitness'],columns=df['Product'],margi
ns=True, normalize="columns"))
```

```
P(Product|Fitness):
Product     KP281     KP481     KP781
Fitness
1        0.500000  0.500000  0.000000
2        0.538462  0.461538  0.000000
3        0.556701  0.402062  0.041237
4        0.375000  0.333333  0.291667
5        0.064516  0.000000  0.935484
All      0.444444  0.333333  0.222222

P(Fitness|Product):
Product   KP281     KP481  KP781       All
Fitness
1        0.0125  0.016667  0.000  0.011111
2        0.1750  0.200000  0.000  0.144444
3        0.6750  0.650000  0.100  0.538889
4        0.1125  0.133333  0.175  0.133333
5        0.0250  0.000000  0.725  0.172222
```

Conditional Probability of Product purchase with respect to Marital Status :

```
print("P(Product|MaritalStatus): ")
print(pd.crosstab(index=df['MaritalStatus'],columns=df['Product']
,margins=True, normalize="index"))
print()
print("P(MaritalStatus|Product): ")
print(pd.crosstab(index=df['MaritalStatus'],columns=df['Product']
,margins=True, normalize="columns"))
```

```
P(Product|MaritalStatus):
Product              KP281     KP481     KP781
MaritalStatus
Partnered         0.448598  0.336449  0.214953
Single            0.438356  0.328767  0.232877
All               0.444444  0.333333  0.222222

P(MaritalStatus|Product):
Product          KP281  KP481  KP781       All
MaritalStatus
Partnered          0.6    0.6  0.575  0.594444
Single             0.4    0.4  0.425  0.405556
```

Conditional Probability of Product purchase with respect to Income Range :

```
print("P(Product|IncomeRange): ")
print(pd.crosstab(index=pd.cut(df["Income"],[20000,30000,40000,50
000,60000,70000,80000],include_lowest=True,right=True),
columns=df['Product'],margins=True, normalize="index"))
print()
print("P(IncomeRange|Product): ")
print(pd.crosstab(index=pd.cut(df["Income"],[20000,30000,40000,50
000,60000,70000,80000],include_lowest=True,right=True),
columns=df['Product'],margins=True, normalize="columns"))
```

```
P(Product|IncomeRange):
Product                  KP281     KP481     KP781
Income
(19999.999, 30000.0]  1.000000  0.000000  0.000000
(30000.0, 40000.0]    0.709677  0.290323  0.000000
(40000.0, 50000.0]    0.490196  0.411765  0.098039
(50000.0, 60000.0]    0.472727  0.418182  0.109091
(60000.0, 70000.0]    0.315789  0.368421  0.315789
(70000.0, 80000.0]    0.000000  0.000000  1.000000
All                   0.496894  0.372671  0.130435

P(IncomeRange|Product):
Product                KP281     KP481     KP781       All
Income
(19999.999, 30000.0]  0.0125  0.000000  0.000000  0.006211
(30000.0, 40000.0]    0.2750  0.150000  0.000000  0.192547
(40000.0, 50000.0]    0.3125  0.350000  0.238095  0.316770
(50000.0, 60000.0]    0.3250  0.383333  0.285714  0.341615
(60000.0, 70000.0]    0.0750  0.116667  0.285714  0.118012
(70000.0, 80000.0]    0.0000  0.000000  0.190476  0.024845
```

Conditional Probability of Product purchase with respect to Age :

```
print("P(Product|AgeRange): ")
print(pd.crosstab(index=df["age_bins"],columns=df['Product'],marg
ins=True, normalize="index"))
print()
print("P(AgeRange|Product): ")
print(pd.crosstab(index=df["age_bins"],columns=df['Product'],marg
ins=True, normalize="columns"))
```

```
P(Product|AgeRange):
Product      KP281     KP481     KP781
age_bins
0-18      1.000000  0.000000  0.000000
18-28     0.462264  0.301887  0.235849
28-38     0.400000  0.418182  0.181818
38-48     0.411765  0.294118  0.294118
48-58     1.000000  0.000000  0.000000
All       0.444444  0.333333  0.222222

P(AgeRange|Product):
Product    KP281     KP481  KP781       All
age_bins
0-18      0.0125  0.000000  0.000  0.005556
18-28     0.6125  0.533333  0.625  0.588889
28-38     0.2750  0.383333  0.250  0.305556
38-48     0.0875  0.083333  0.125  0.094444
48-58     0.0125  0.000000  0.000  0.005556
```

Conditional Probability of Product purchase with respect to Miles :

```
print("P(Product|Miles): ")
print(pd.crosstab(index=pd.cut(df["Miles"],[5,50,100,150,200,250,
300,350,400],include_lowest=True,right=True),
columns=df['Product'],margins=True, normalize="index"))
print()
print("P(Miles|Product): ")
print(pd.crosstab(index=pd.cut(df["Miles"],[5,50,100,150,200,250,
300,350,400],include_lowest=True,right=True),
columns=df['Product'],margins=True, normalize="columns"))
```

```
P(Product|IncomeRange):
Product              KP281     KP481     KP781
Miles
(4.999, 50.0]     0.705882  0.294118  0.000000
(50.0, 100.0]     0.515464  0.402062  0.082474
(100.0, 150.0]    0.421053  0.342105  0.236842
(150.0, 200.0]    0.090909  0.090909  0.818182
(200.0, 250.0]    0.000000  0.500000  0.500000
(250.0, 300.0]    0.000000  0.000000  1.000000
(350.0, 400.0]    0.000000  0.000000  1.000000
All               0.444444  0.333333  0.222222

P(IncomeRange|Product):
Product           KP281     KP481  KP781       All
Miles
(4.999, 50.0]     0.150  0.083333  0.000  0.094444
(50.0, 100.0]     0.625  0.650000  0.200  0.538889
(100.0, 150.0]    0.200  0.216667  0.225  0.211111
(150.0, 200.0]    0.025  0.033333  0.450  0.122222
(200.0, 250.0]    0.000  0.016667  0.025  0.011111
(250.0, 300.0]    0.000  0.000000  0.075  0.016667
(350.0, 400.0]    0.000  0.000000  0.025  0.005556
```

5) Check the correlation among different factors

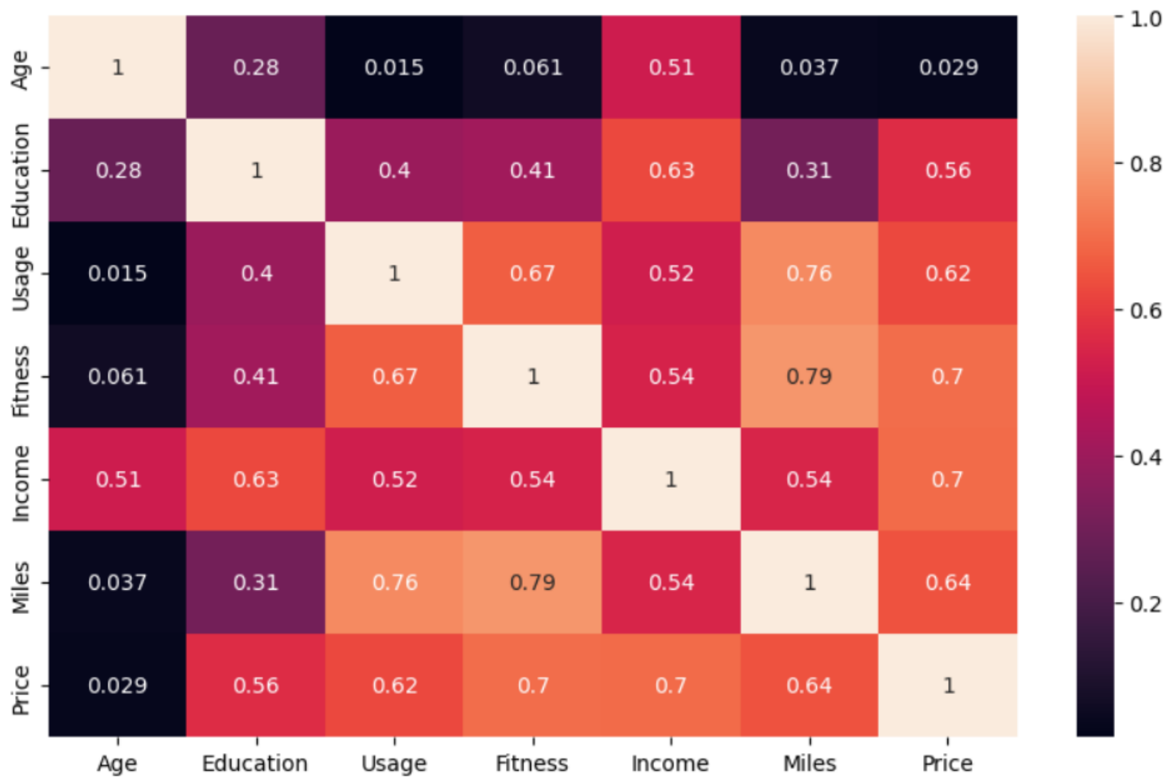
a) Find the correlation between the given features in the table.

```
df.corr()
```

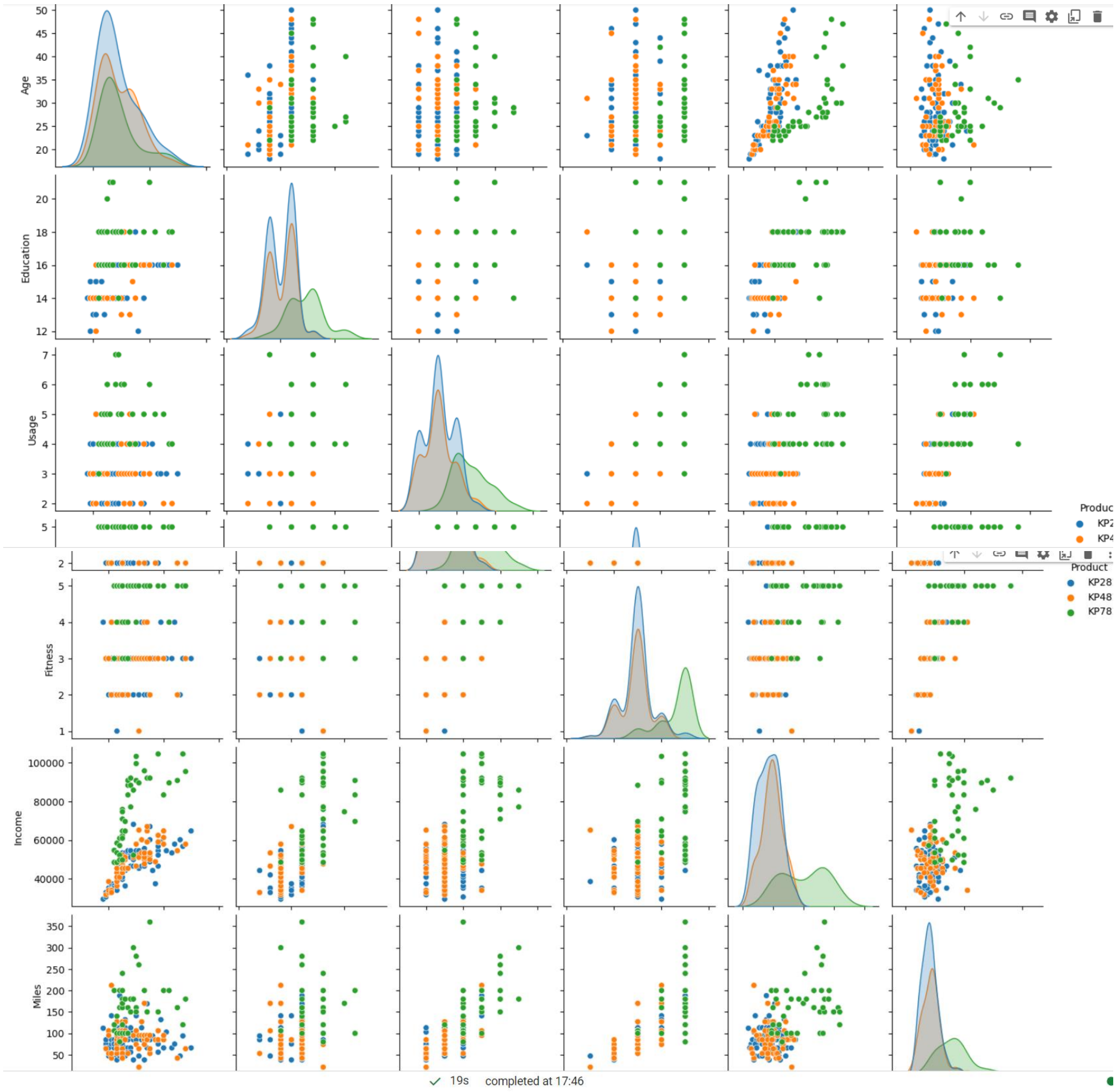
```
df.corr()
```

	Age	Education	Usage	Fitness	Income	Miles	Price
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618	0.029263
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284	0.563463
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130	0.623124
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702	0.696616
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473	0.695847
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000	0.643923
Price	0.029263	0.563463	0.623124	0.696616	0.695847	0.643923	1.000000

```
fig = plt.figure(figsize=(10,6))
fig.suptitle("Correlation Analysis")
sns.heatmap(df.corr(),annot=True)
plt.show()
```




```
sns.pairplot(df,hue="Product")
plt.show()
```



Observations :

- Fitness and Miles have a positive and a very high correlation: 0.79
- Usage and Miles have a positive and a high correlation of 0.76
- Product price and Income have a positive and a very high correlation: 0.70

This is because Education and Income have a positive and high correlation, and Product price and Income also have a positive and high correlation. That is the main reason KP781 which is a higher variant of the treadmill with higher price is so popular because of the high income of the customers

6. Customer profiling and recommendation

a) Make customer profiling's for each and every product

Customer Profiling :

- **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
  - Most affordable and entry level and Maximum Selling Product.
  - Same number of Male and Female customers.
  - More general purpose for all age group and fitness levels.

- **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 - 70 to 200 miles
  - Intermediate Price Range
  - Probability of Female customer buying KP481 is significantly higher than male.
  - KP481 product is specifically recommended for Female customers who are intermediate user.
- **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
  - least sold product.
  - high price and preferred by customers who does exercises more extensively and run more miles.
  - Female Customers who are running average 180 miles (extensive exercise) , are using product KP781, which is higher than Male average using same product.
  - KP781 can be recommended for Female customers who exercises extensively.
  - Probability of Male customer buying Product KP781(31.73%) is way more than female(9.21%).
  - Probability of a single person buying KP781 is higher than Married customers. So , KP781 is also recommended for people who are single and exercises more.
  - most of old people who are above 45 age and adult uses this product.

B) Write a detailed recommendation from the analysis that you have done.

BUSINESS INSIGHTS:

Following are the business insights we have:

- KP281, followed by KP481 and KP781, is the most popular treadmill sold from Aerofit, with having almost 50% of the market share, consequently generating highest revenue for Aerofit.
- Majority of the customers are from the age group of 20-35, most of them tend to buy KP281, followed by KP481 and KP781. It is found that customers of age greater than 30 are not very keen on buying KP781. The most probable reason is not kepping up the fitness level and under-usage of treadmills. Customers, who knows their fitness level, tend not to invest in expensive treadmills. According to probabilities, people of age between 45-35, go for KP281 more than any other age group, but for other variants it is the consumers falling age group of 25-35.
- Male consumers tend to buy treadmills more than the female counterparts, though both males and females have equal affinity towards buying KP281. Males are more dominant buyer for higher variants, this can be seen in probability scores for males and females. The main reason, probably, can be with average usage levels and miles travelled by males are more than their females, thus have more average fitness level score. This can be used to target the audience for selling upcoming products.
- Most of the consumers have completed 16 years of education. Consumers with 16 years of education or less have more affinity towards KP281, followed by KP481 and KP781. However, people with more than 16 years of education tend to go for expensive variants, KP781. The probability of consumer with 18 years of education getting KP781 is 70%, which is much more than any other education category, the nearest one to this score is 11% for consumer with 16 years of education getting KP781. This insight can be used to target audience for expensive treadmills in future, thus increase profit margin for Aerofit.
- People with partners are more probable consumers of buying a treadmill from Aerofit than single ones, partnered customers prefer KP281 and KP481, than single customers. But, Single customers are more preferable audience for pitching KP781, as per the probability. This demography can be helpful to target different category of treadmills.
- Most of the customers rate themselves as average (rating 3) in fitness. Customers with average or below-average fitness level tend to buy KP281, but people with above average fitness level prefer KP781. This implies a positive correlation between the two. The fitness concerned customers might be attracted towards extra features provided in expensive KP781, which might help them maintain and go beyond. Fitness concerned 25-35 age group males are the perfect audience for KP781.
- Average customers of Aerofit treadmills fall in the income range of 35k-60k USD. The relation between income range and product buying tendency is positive. People with higher income tend to buy KP781, the expensive variant, because of affordability. Usually, customers earning more than 60k USD, go for KP781, below which people tend to go for KP281 primarily, followed by KP481.

- As the values suggest, male consumers (88.4 miles) tend to run slightly more on average on KP481 treadmills than their female counterpart (87.3 miles), the difference is more significant for KP281 and KP781. Males, on an average, run 89.3 miles compared to 76.2 miles for females. The trend of males running more on average than females is not followed for KP781, male consumers run 164.1 miles on average on KP781 whereas females run 180 miles.
- In general, people who run/walk more miles(>130) , are more likely to use KP781 product.
- People who walk/run around 60 to 130 miles are more likely to use KP281 and KP481 products.

Customer profiles for different treadmills:

KP281: An average earning, average fitness concerned, people are usual customers. Also, people with age greater than 30, prefer KP281.

KP481: Similar to KP281, but with somewhat better earning and fitness level.

KP781: Highly educated, high income, fitness concerned with high usage and miles covered, withing the age group of 18-35 years.

Recommendations on Actionable Insights

1. A better product with better features such as advanced fitness tracking and estimator, for highly-educated, high income and active customers to increase revenue and profit margin for Aerofit.

2. Target more customers having age between 18 to 35 as more than 85% of the customers who bought treadmill lie in this range.
3. People with Education levels less than or equal to 16 are likely to purchase KP281 and KP481. And people with Education levels greater than or equal to 16 are likely to purchase KP781.
4. People with less than 16 years of education, with high fitness level, might be presented with offers for KP781, so that it encourages other group to level up their fitness by buying KP781.

5. Males are more likely to purchase a treadmill with 58% ratio than Females. Both are likely to purchase equal number of KP281 and KP481, but Males have high chances of purchasing KP781 as 82% of total sale of KP781 is purchased by Males.
6. KP781 for females, as they are falling behind in numbers for this treadmill. A campaign, encouraging women to take up fitness challenge with the treadmill, will surely make the numbers soar.

7. KP281 and KP481 bring in significant revenue and is preferred by young individuals, with age < 30 and average fitness level, adding features and discounts could help boost sales for these. Otherwise, bringing other treadmills in similar price range and maximize the market.

8. Partnered people, especially males, can be targeted with treadmills, as they are the most probable customers.

9. People with Usage less than or equal to 4 are likely to purchase KP281 and KP481. And people with Usage greater than or equal to 4 are likely to purchase KP781.
10. People with Income less than 60000 are likely to purchase KP281 and KP481. And people with Income greater than 60000 are likely to purchase KP781. Hence target higher income group people to sell KP781
11. People with Fitness Level 3 or less are likely to purchase KP281 and KP481. And with Fitness Level 5 are likely to purchase KP781. It is necessary to focus on normal people more as sales of KP281 is more and recommend KP781 to who have a high fitness level.
12. People who use the treadmill more are more likely to purchase KP781. As, buying the treadmill is directly proportional to its usage.
13. Recommend KP781 product to users who exercises/run more frequently and run more and more miles , and have high income. Since Kp781 is least selling product (22.2% share of all the products) , recommend this product some customers who exercise at intermediate to extensive level , if they are planning to go for KP481. Also the targeted Age Category is Adult and age above 45.
14. Recommend KP481 product specifically for female customers who run/walk more miles , as data shows their probability is higher. Statistical Summary about fitness level and miles for KP481 is not good as KP281 which is cheaper product. Possibly because of price, customers prefer to purchase KP281. It is recommended to make some necessary changes either to decrease the price of product or offer some discounts on the product or improve the features of the product K481 to increase customer experience.

Link to Collab Note book :

https://colab.research.google.com/drive/1otg22-aFnz2kQ8FDUpFWBgPMh1PlgeNZ?usp=drive_link