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 Information :
Product Purchased: KP281, KP481, or KP781

Age: In years

Gender: Male/Female

Education: In years

Marital Status: Single or partnered

Usage: The average number of times the customer plans to use the treadmill each week.

Income: Annual income (in $)

Fitness: Self-rated fitness on a 1-to-5 scale, where 1 is the poor shape and 5 is the excellent shape.

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

Product Portfolio:

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

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from scipy.stats import norm
import warnings
warnings.filterwarnings('ignore')
```

```
data = pd.read_csv('aerofit_treadmill.txt')
data.head()
```

Table:

	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

print (f “data description”)

data.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

print (f “Check the data types of all columns”)

data.info()

```
print(f"Total Aerofit Products present in the dataset : {data['Product'].nunique()}")
print(f"---"*40)
print(f"Count of Genders for each products :\n{data.groupby(['Product'])['Gender'].value_counts()}")
print(f"---"*40)
print(f"How did the people rated themselves : \n{data['Fitness'].value_counts().sort_values()}")
print(f"---"*40)
print(f"The average number of times the customer plans to use the treadmill each week. \
: \n{data['Usage'].value_counts().sort_values()}")
print(f"---"*40)
print(f"How many peoples are single/married :
\n{data['MaritalStatus'].value_counts().sort_values()}")
```

Total Aerofit Products present in the dataset : 3
Count of Genders for each products :

Product	Gender	

KP281	Female	40
	Male	40
KP481	Female	29
	Male	31
KP781	Female	7
	Male	33

Name: Gender, dtype: int64

How did the people rated themselves:

1		2
4		24
2		26
5		31
3		97

Name: Fitness, dtype: int64

The average number of times the customer plans to use the treadmill each week. :

7	2
6	7
5	17
2	33
4	52
3	69

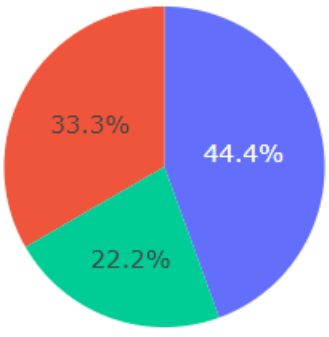
Name: Usage, dtype: int64

How many peoples are single/married : Single 73 Partnered 107

Name: MaritalStatus, dtype: int64

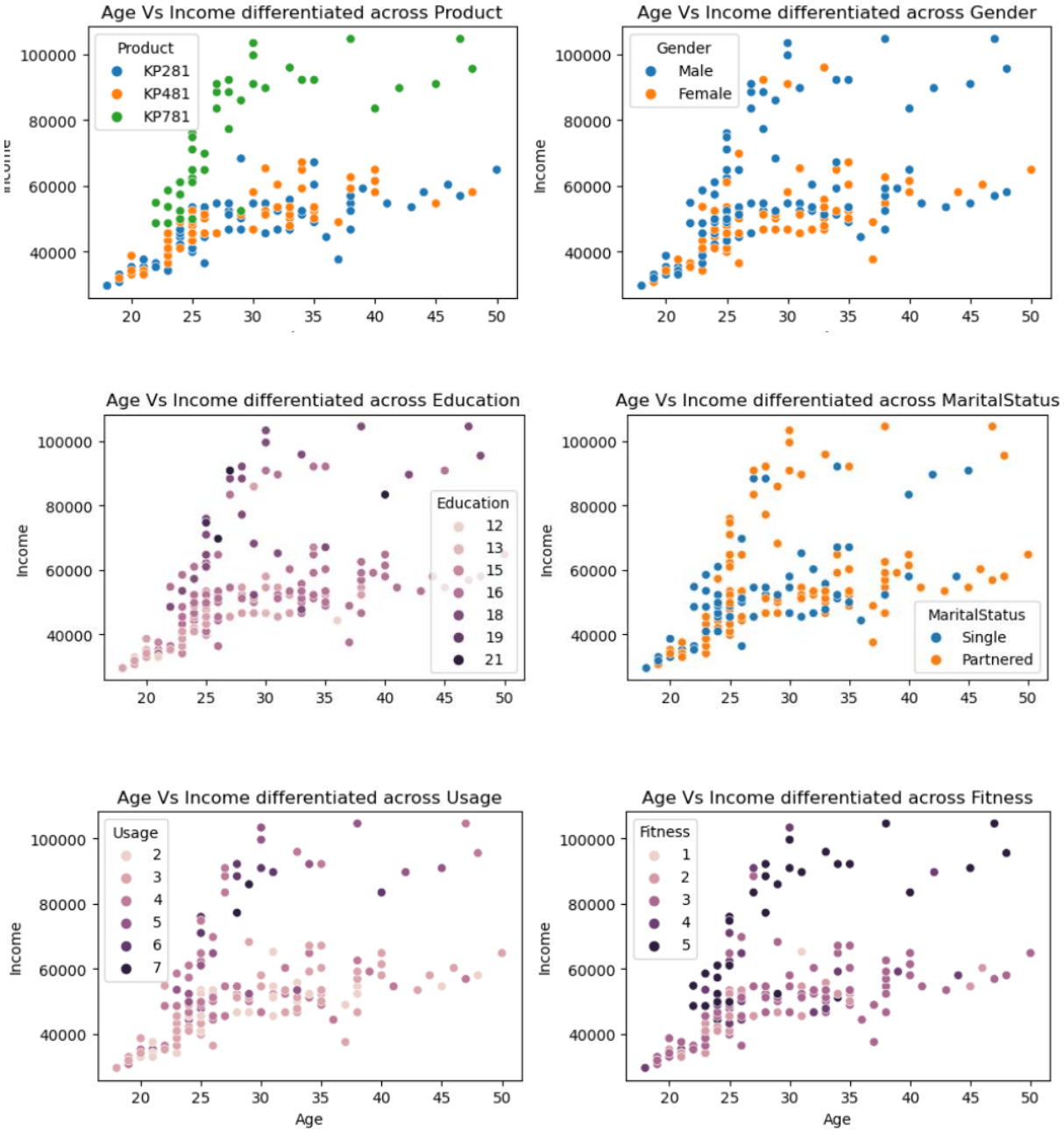
```
df = data.groupby('Product')['Gender'].count().reset_index()
df.columns = ['Product','Total_Count']
fig = px.pie(df,values='Total_Count',color='Product')
fig.update_layout(
    autosize=False,
    width=500,
    height=300
```

Product KP281 borrowed by 45% people followed by KP481 and KP781




```
col_list = ['Product','Gender','Education','MaritalStatus','Usage','Fitness']
fig,axes = plt.subplots(nrows=3,ncols=2,figsize=(10, 10))
axes = axes.flatten()
i = 0
```

```
for col in col_list:
    sns.scatterplot(data=data,x='Age',y='Income',hue=col,ax=axes[i])
    axes[i].set_title(f"Age Vs Income differentiated across {col}")
    i += 1
fig.tight_layout()
plt.show()
```



1.Income is increasing as the age increasing. But there are some exceptional people who got higher salary in their early age only.

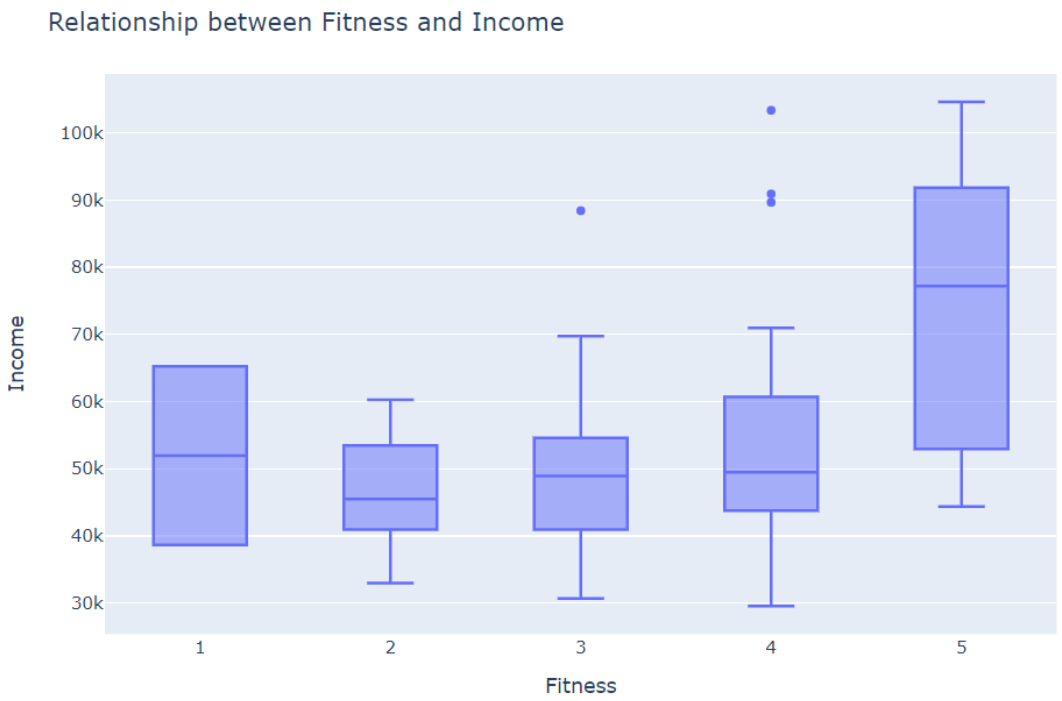
2. As per given product portfolio information KP781 is costliest treadmill --> that's why only people with highest income borrowing it.

3. Males tends to get more higher salary than Females.

```
fig = px.box(data_frame=data,x='Gender',y='Income',color='MaritalStatus');
fig.update_layout(
    title="Martial Status of customer as per there Income and Gender",
)
fig.show()
```

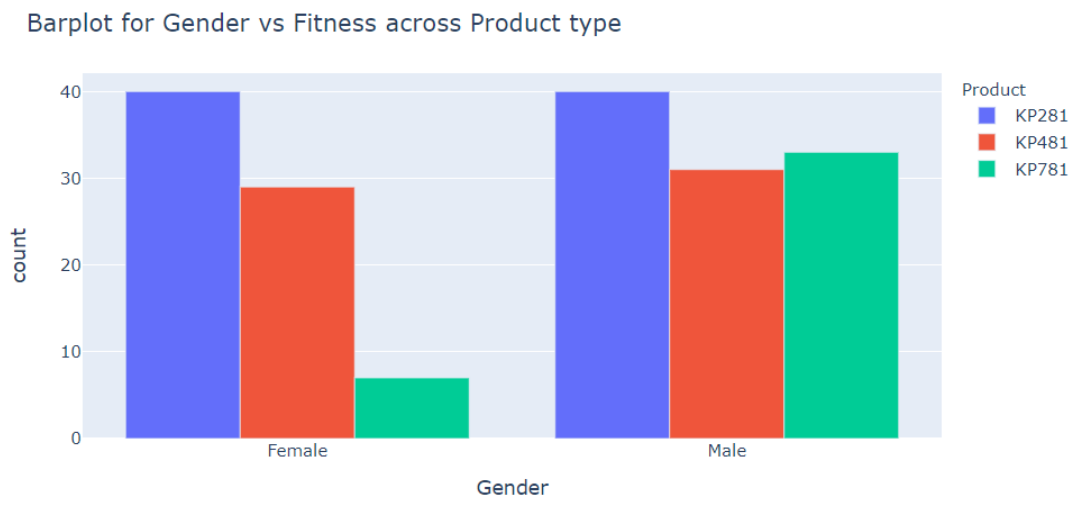
1. Male tends get more higher salary than female.
2. Married woman & man more salary than single ones.

```
fig = px.box(data_frame=data,x='Fitness',y='Income');
fig.update_layout(
    title="Relationship between Fitness and Income",
)
fig.show()
```

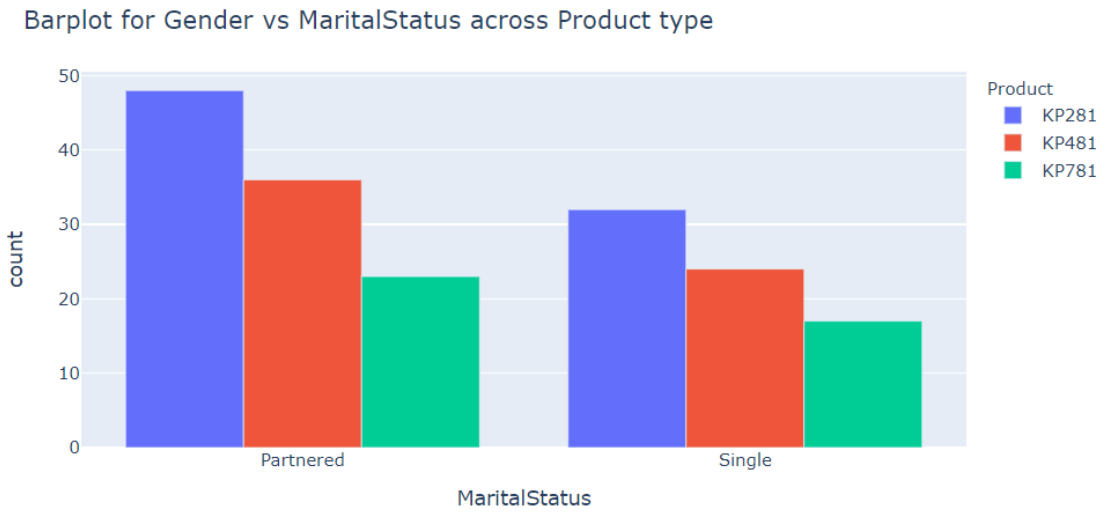


People with high salary find themselves more fit & rated 5.

```
df = data.groupby('Gender')['Product'].value_counts().reset_index(name='count')
fig = px.bar(data_frame=df,x='Gender',y='count',color='Product',barmode='group');
fig.update_layout(
   title="Barplot for Gender vs Fitness across Product type",
   width=800,
   height=400
)
fig.show()
```

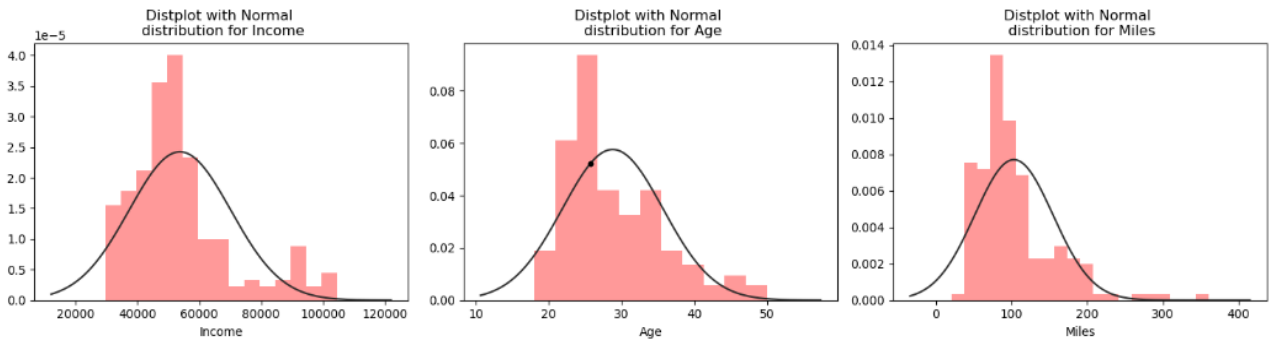


```
df = data.groupby('MaritalStatus')['Product'].value_counts().reset_index(name='count')
fig = px.bar(data_frame=df,x='MaritalStatus',y='count',color='Product',barmode='group');
fig.update_layout(
   title="Barplot for Gender vs MaritalStatus across Product type",
   width=800,
   height=400
)
fig.show()
```



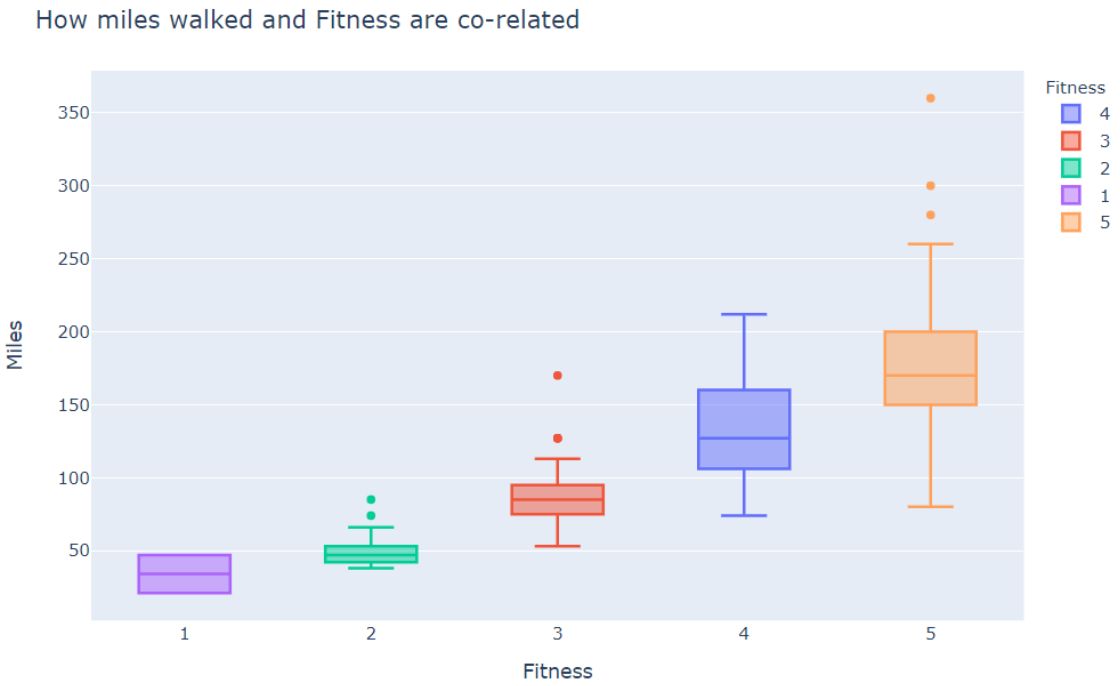
Males are more tend to buy KP781 treadmill which the costliest product among all.

```
fig,axes = plt.subplots(nrows=1,ncols=3,figsize=(15, 4))
axes = axes.flatten()
col_list = ['Income','Age','Miles']
i = 0
for col in col_list:
   sns.distplot(data[col],fit=norm,kde=False, color=['red'],ax=axes[i])
   axes[i].set_title(f"Distplot with Normal \n distribution for {col}")
   i += 1
fig.tight_layout()
plt.show()
```



Income, Age and Miles are not normally distributed.

```
fig = px.box(data_frame=data,x='Fitness',y='Miles',color='Fitness');
fig.update_layout(
   title="How miles walked and Fitness are co-related",
)
fig.show()
```



If the customer walking more no. of miles distances weekly the chances of fitness of being Fit is high.

Marginal Probability

```
def marginal_probability_cal(data,label):
   """
This method is used to calculate marginal/unconditional probabilities

   Args:
      data : dataframe
      label : columns for which we wanna calculate it
   """
   print(f"% Marginal Probability for {label} is :
\n{round(data[label].value_counts(normalize=True)*100,2)}")
   print("---"*40)

## Marginal Probability calculation for product
marginal_probability_cal(data,'Product')
# ------------------------------------------------------------------------------------------------------------
## Marginal Probability calculation for Gender
marginal_probability_cal(data,'Gender')
# ------------------------------------------------------------------------------------------------------------
## Marginal Probability calculation for Fitness
marginal_probability_cal(data,'Fitness')
```

```
## Marginal Probability calculation for Fitness
marginal_probability_cal(data,'Usage')
# ------------------------------------------------------------------------------------------------------------
## Marginal Probability calculation for Fitness
marginal_probability_cal(data,'Education')
```

% Marginal Probability for Product is: KP281: 44.44 KP481: 33.33 KP781: 22.22 Name: Product, dtype: float64 ----------

% Marginal Probability for Gender is: Male: 57.78 Female: 42.22 Name: Gender, dtype: float64 ----------

% Marginal Probability for Fitness is:

3	53.89
5	17.22
2	14.44
4	13.33
1	1.11

Name: Fitness, dtype: float64 ----------

% Marginal Probability for Usage is:

3	38.33
4	28.89
2	18.33
5	9.44
6	3.89
7	1.11

Name: Usage, dtype: float64 ----------

% Marginal Probability for Education is:

16	47.22
24	30.56
18	12.78
15	2.78
13	2.78
12	1.67
21	1.67
20	0.56

Name: Education, dtype: float64 ----------

```
def conditional_probabilities(data,label,print_marginal=False):
  """
  This method is used to calculate conditional probabilities with Product and Gender

  Args:
    data : dataframe
    label : column
  """
  print("***"*40)
  print(f"\t \t \t Conditional Probability for Gender : {label} and Products")
  print("***"*40)
  df = pd.crosstab(index=data['Gender'], columns=[data['Product']])
```

```
    print(df)
    p_781 = df['KP781'][label] / df.loc[label].sum()
    p_481 = df['KP481'][label] / df.loc[label].sum()
    p_281 = df['KP281'][label] / df.loc[label].sum()
    print('---'*40)
    print(f"P(KP781|{label}): {p_781*100:.2f}%")
    print(f"P(KP481|{label}): {p_481*100:.2f}%")
    print(f"P(KP281|{label}): {p_281*100:.2f}%\n")

conditional_probabilities(data,'Male', True)
conditional_probabilities(data,'Female',True)
```

```
Conditional Probability for Gender : Male and Products
****************************************************************************************************
Product KP281 KP481 KP781 Gender Female 40 29 7 Male 40 31 33 ----------------------------------------
P(KP781|Male): 31.73% P(KP481|Male): 29.81% P(KP281|Male): 38.46%
****************************************************************************************************
Conditional Probability for Gender : Female and Products
****************************************************************************************************
Product KP281 KP481 KP781 Gender Female 40 29 7 Male 40 31 33 ----------------------------------------
P(KP781|Female): 9.21% P(KP481|Female): 38.16% P(KP281|Female): 52.63%
```

```
def conditional_probabilities(data,label,print_marginal=False):
    """
    This method is used to calculate conditional probabilities with Product and
MaritalStatus

    Args:
        data : dataframe
        label : column
    """
    print("***"*40)
    print(f"\t \t \t Conditional Probability for Marital Status : {label} and Products")
    print("***"*40)
    df = pd.crosstab(index=data['MaritalStatus'], columns=[data['Product']])
    print(df)
    p_781 = df['KP781'][label] / df.loc[label].sum()
    p_481 = df['KP481'][label] / df.loc[label].sum()
    p_281 = df['KP281'][label] / df.loc[label].sum()
    print('---'*40)
    print(f"P(KP781|{label}): {p_781*100:.2f}%")
    print(f"P(KP481|{label}): {p_481*100:.2f}%")
    print(f"P(KP281|{label}): {p_281*100:.2f}%\n")

conditional_probabilities(data,'Single', True)
conditional_probabilities(data,'Partnered',True)
```

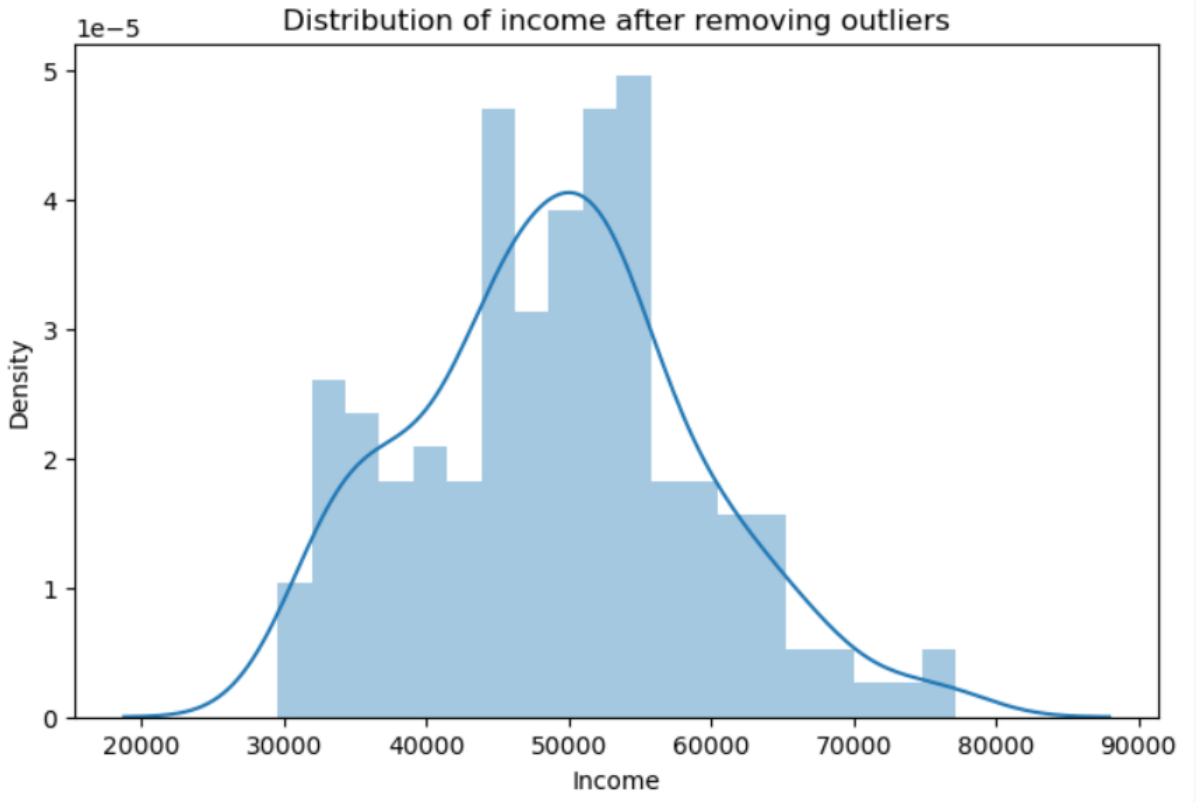
Conditional Probability for Marital Status : Single and Products
****
Product KP281 KP481 KP781 MaritalStatus Partnered 48 36 23 Single 32 24 17 ----------
P(KP781|Single): 23.29% P(KP481|Single): 32.88% P(KP281|Single): 43.84%
****
Conditional Probability for Marital Status : Partnered and Products
****
Product KP281 KP481 KP781 MaritalStatus Partnered 48 36 23 Single 32 24 17 ----------
P(KP781|Partnered): 21.50% P(KP481|Partnered): 33.64% P(KP281|Partnered): 44.86%

```
def conditional_probabilities(data,label,print_marginal=False):
  """
  This method is used to calculate conditional probabilities with Product and MaritalStatus

  Args:
    data : dataframe
    label : column
  """
  print("***"*40)
  print(f"\t \t \t Conditional Probability for Fitness rating : {label} and Products")
  print("***"*40)
  df = pd.crosstab(index=data['Fitness'], columns=[data['Product']])
  print(df)
  p_781 = df['KP781'][label] / df.loc[label].sum()
  p_481 = df['KP481'][label] / df.loc[label].sum()
  p_281 = df['KP281'][label] / df.loc[label].sum()
  print('---'*40)
  print(f"P(KP781|{label}): {p_781*100:.2f}%")
  print(f"P(KP481|{label}): {p_481*100:.2f}%")
  print(f"P(KP281|{label}): {p_281*100:.2f}%\n")

conditional_probabilities(data,1, True)
conditional_probabilities(data,2,True)
conditional_probabilities(data,3,True)
conditional_probabilities(data,4,True)
conditional_probabilities(data,5,True)
```

****
Conditional Probability for Fitness rating : 1 and Products
****
Product KP281 KP481 KP781 Fitness 1 1 1 0 2 14 12 0 3 54 39 4 4 9 8 7 5 2 0 29 ----------
---
P(KP781|1): 0.00% P(KP481|1): 50.00% P(KP281|1): 50.00%
****
Conditional Probability for Fitness rating : 2 and Products
****
Product KP281 KP481 KP781 Fitness 1 1 1 0 2 14 12 0 3 54 39 4 4 9 8 7 5 2 0 29 ----------
---
P(KP781|2): 0.00% P(KP481|2): 46.15% P(KP281|2): 53.85%
****

```
Conditional Probability for Fitness rating : 3 and Products
**********************************************************************************************
Product KP281 KP481 KP781 Fitness 1 1 1 0 2 14 12 0 3 54 39 4 4 9 8 7 5 2 0 29 ----------
-----------------------------------------------------------------------------------------------
P(KP781|3): 4.12% P(KP481|3): 40.21% P(KP281|3): 55.67%
**********************************************************************************************
Conditional Probability for Fitness rating : 4 and Products
**********************************************************************************************
Product KP281 KP481 KP781 Fitness 1 1 1 0 2 14 12 0 3 54 39 4 4 9 8 7 5 2 0 29 ----------
-----------------------------------------------------------------------------------------------
P(KP781|4): 29.17% P(KP481|4): 33.33% P(KP281|4): 37.50%
**********************************************************************************************
Conditional Probability for Fitness rating : 5 and Products
**********************************************************************************************
Product KP281 KP481 KP781 Fitness 1 1 1 0 2 14 12 0 3 54 39 4 4 9 8 7 5 2 0 29 ----------
-----------------------------------------------------------------------------------------------
P(KP781|5): 93.55% P(KP481|5): 0.00% P(KP281|5): 6.45%
```

```
3.  def remove_outliers(df, label, iqr_factor = 1.5) :
4.      """
5.      Outlier Treatment :
6.      In income columns we can see some outliers we can remove them.
7.
8.      """
9.
10.     q1 = df[label].quantile(0.25)
11.     q3 = df[label].quantile(0.75)
12.     IQR = q3-q1
13.     print(f'IQR for the label column is : {IQR}')
14.     print(df.shape)
15.     df = df[ (df[label] > (q1 - (iqr_factor * IQR))) & (df[label] < (q3 + (iqr_factor * IQR) )) ]
16.     print(df.shape)
17.
18.     plt.figure(figsize= (8,5))
19.     sns.distplot(df['Income'], bins =20)
20.     plt.title('Distribution of income after removing outliers')
21.     plt.show()
22.
23.     return df
24. df_remove_outliers = remove_outliers(data,'Income',iqr_factor=1.5)
25. IQR for the label column is : 14609.25 (180, 9) (161, 9)
```



Recommendations:

1.For selling more KP781 treadmill: company should target people with higher salaries (>70k) & also males are more tends to buy it.

2. Also KP781 treadmill's targeted customers will having age between 22-35

3. Single people with high income mostly prefer KP781 treadmill that will be most important target audience for Aerofit.