Aerofit Case Study

October 22, 2023

# 1 Business Case: Aerofit - Descriptive Statistics & Probability

# 2 1.Exploratory Data Analysis

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import copy
```

[80]:
```
df = pd.read_csv("/Users/senth/Desktop/aerofit_treadmill.csv")
```

[81]:
```
df.head()
```

[81]:
```
  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
```

[82]:
```
df.tail()
```

[82]:
```
    Product  Age Gender  Education MaritalStatus  Usage  Fitness  Income  \
175   KP781   40   Male         21        Single      6        5   83416
176   KP781   42   Male         18        Single      5        4   89641
177   KP781   45   Male         16        Single      5        5   90886
178   KP781   47   Male         18     Partnered      4        5  104581
179   KP781   48   Male         18     Partnered      4        5   95508

     Miles
175    200
176    200
177    160
178    120
```

```
179    180
```

[7]:
```python
df.shape
```

[7]:
```
(180, 9)
```

[8]:
```python
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
```

- From the above analysis, it is clear that, data has total of 9 features with mixed alpha numeric data. Also we can see that there is no missing data in the columns.
- The data type of all the columns are matching with the data present in them. But we will change the datatype of Usage and Fitness into str(object).

### 2.0.1 Changing the Datatype of Columns

- Changing the datatype of Usage and Fitness columns

[9]:
```python
df['Usage'] = df['Usage'].astype('str')
df['Fitness'] = df['Fitness'].astype('str')

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    object
 6   Fitness        180 non-null    object
 7   Income         180 non-null    int64
 8   Miles          180 non-null    int64
dtypes: int64(4), object(5)
memory usage: 12.8+ KB
```

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

[10]:
```
       Product Gender MaritalStatus Usage Fitness
count      180    180           180   180     180
unique       3      2             2     6       5
top      KP281   Male     Partnered     3       3
freq        80    104           107    69      97
```

[11]:
```
# statisctical summary of numerical data type columns

df.describe()
```

[11]:
```
              Age   Education         Income       Miles
count  180.000000  180.000000     180.000000  180.000000
mean    28.788889   15.572222   53719.577778  103.194444
std      6.943498    1.617055   16506.684226   51.863605
min     18.000000   12.000000   29562.000000   21.000000
25%     24.000000   14.000000   44058.750000   66.000000
50%     26.000000   16.000000   50596.500000   94.000000
75%     33.000000   16.000000   58668.000000  114.750000
max     50.000000   21.000000  104581.000000  360.000000
```

# 3 2.Non-Graphical Analysis

[14]:
```
df_n = df[['Product','Gender','MaritalStatus']].melt()
round(df_n.groupby(['variable','value'])['value'].count()/len(df)*100,2)
```

[14]:
```
variable       value
Gender         Female       42.22
               Male         57.78
MaritalStatus  Partnered    59.44
               Single       40.56
Product        KP281        44.44
               KP481        33.33
               KP781        22.22
Name: value, dtype: float64
```

1. Probability of Buying KP281 increased from 44.44% to 58.7%, if customer is Female and Partnered.
2. Probability of Buying KP481 increased from 33.33% to 46.7%, if customer is Female and Single.

3. Probability of Buying KP781 increased from 22.22% to 32.56%, if customer is Male and Single.
4. Probability of Buying KP781 decreased from 22.22% to 8.7%, if customer is Female and Partnered.

[84]:
```
df_ms = df[(df['Gender']=='Male')&(df['MaritalStatus']=='Single')]

v1=round(len(df_ms[df_ms['Product']=='KP281'])/len(df_ms)*100,2)
v2=round(len(df_ms[df_ms['Product']=='KP481'])/len(df_ms)*100,2)
v3=round(len(df_ms[df_ms['Product']=='KP781'])/len(df_ms)*100,2)

print("Probability of Buying KP281 product when customer is Male and Single is␣
 ↪= {}%".format(v1))
print("Probability of Buying KP481 product when customer is Male and Single is␣
 ↪= {}%".format(v2))
print("Probability of Buying KP781 product when customer is Male and Single is␣
 ↪= {}%".format(v3))
```

```
Probability of Buying KP281 product when customer is Male and Single is = 44.19%
Probability of Buying KP481 product when customer is Male and Single is = 23.26%
Probability of Buying KP781 product when customer is Male and Single is = 32.56%
```

[85]:
```
df_ms = df[(df['Gender']=='Male')&(df['MaritalStatus']=='Partnered')]

v1=round(len(df_ms[df_ms['Product']=='KP281'])/len(df_ms)*100,2)
v2=round(len(df_ms[df_ms['Product']=='KP481'])/len(df_ms)*100,2)
v3=round(len(df_ms[df_ms['Product']=='KP781'])/len(df_ms)*100,2)

print("Probability of Buying KP281 product when customer is Male and Partnered␣
 ↪is = {}%".format(v1))
print("Probability of Buying KP481 product when customer is Male and Partnered␣
 ↪is = {}%".format(v2))
print("Probability of Buying KP781 product when customer is Male and Partnered␣
 ↪is = {}%".format(v3))
```

```
Probability of Buying KP281 product when customer is Male and Partnered is =
34.43%
Probability of Buying KP481 product when customer is Male and Partnered is =
34.43%
Probability of Buying KP781 product when customer is Male and Partnered is =
31.15%
```

[86]:
```
df_ms = df[(df['Gender']=='Female')&(df['MaritalStatus']=='Single')]

v1=round(len(df_ms[df_ms['Product']=='KP281'])/len(df_ms)*100,2)
v2=round(len(df_ms[df_ms['Product']=='KP481'])/len(df_ms)*100,2)
v3=round(len(df_ms[df_ms['Product']=='KP781'])/len(df_ms)*100,2)
```

```
print("Probability of Buying KP281 product when customer is Female and Single␣
 ↪is = {}%".format(v1))
print("Probability of Buying KP481 product when customer is Female and Single␣
 ↪is = {}%".format(v2))
print("Probability of Buying KP781 product when customer is Female and Single␣
 ↪is = {}%".format(v3))
```

```
Probability of Buying KP281 product when customer is Female and Single is =
43.33%
Probability of Buying KP481 product when customer is Female and Single is =
46.67%
Probability of Buying KP781 product when customer is Female and Single is =
10.0%
```

[87]:
```
df_ms = df[(df['Gender']=='Female')&(df['MaritalStatus']=='Partnered')]

v1=round(len(df_ms[df_ms['Product']=='KP281'])/len(df_ms)*100,2)
v2=round(len(df_ms[df_ms['Product']=='KP481'])/len(df_ms)*100,2)
v3=round(len(df_ms[df_ms['Product']=='KP781'])/len(df_ms)*100,2)

print("Probability of Buying KP281 product when customer is Female and␣
 ↪Partnered is = {}%".format(v1))
print("Probability of Buying KP481 product when customer is Female and␣
 ↪Partnered is = {}%".format(v2))
print("Probability of Buying KP781 product when customer is Female and␣
 ↪Partnered is = {}%".format(v3))
```

```
Probability of Buying KP281 product when customer is Female and Partnered is =
58.7%
Probability of Buying KP481 product when customer is Female and Partnered is =
32.61%
Probability of Buying KP781 product when customer is Female and Partnered is =
8.7%
```

[15]:
```
df.nunique()
```

[15]:
```
Product           3
Age              32
Gender            2
Education         8
MaritalStatus     2
Usage             6
Fitness           5
Income           62
Miles            37
dtype: int64
```

[16]:
```
df['Gender'].value_counts()
```

[16]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

[17]:
```
df['Product'].value_counts()
```

[17]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

[18]:
```
df['MaritalStatus'].value_counts()
```

[18]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

[19]:
```
df['Fitness'].value_counts()
```

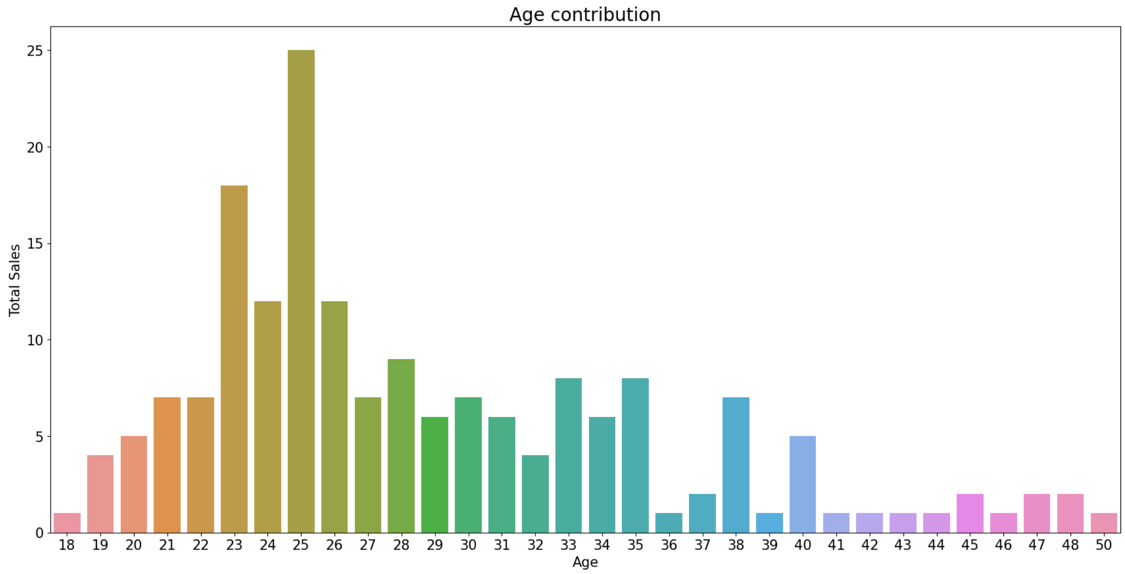
[19]:
```
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

[20]:
```
df['Usage'].value_counts()
```

[20]:
```
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
```

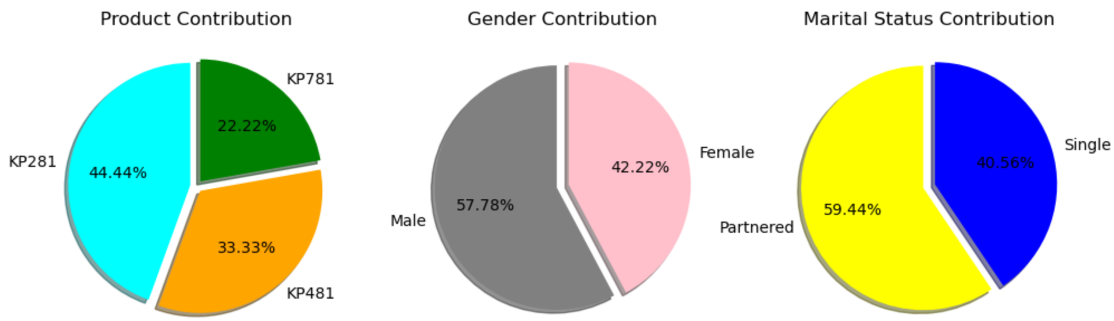
# 4 3.Visual Analysis - Univariate

[22]:
```
plt.figure(figsize=(21,10))
sns.countplot(data=df, x='Age')
plt.ylabel('Total Sales', fontsize=15); plt.title('Age contribution',␣
 ↪fontsize=20); plt.xlabel("Age", fontsize=15)
plt.xticks(fontsize=15); plt.yticks(fontsize=15); plt.show()
```



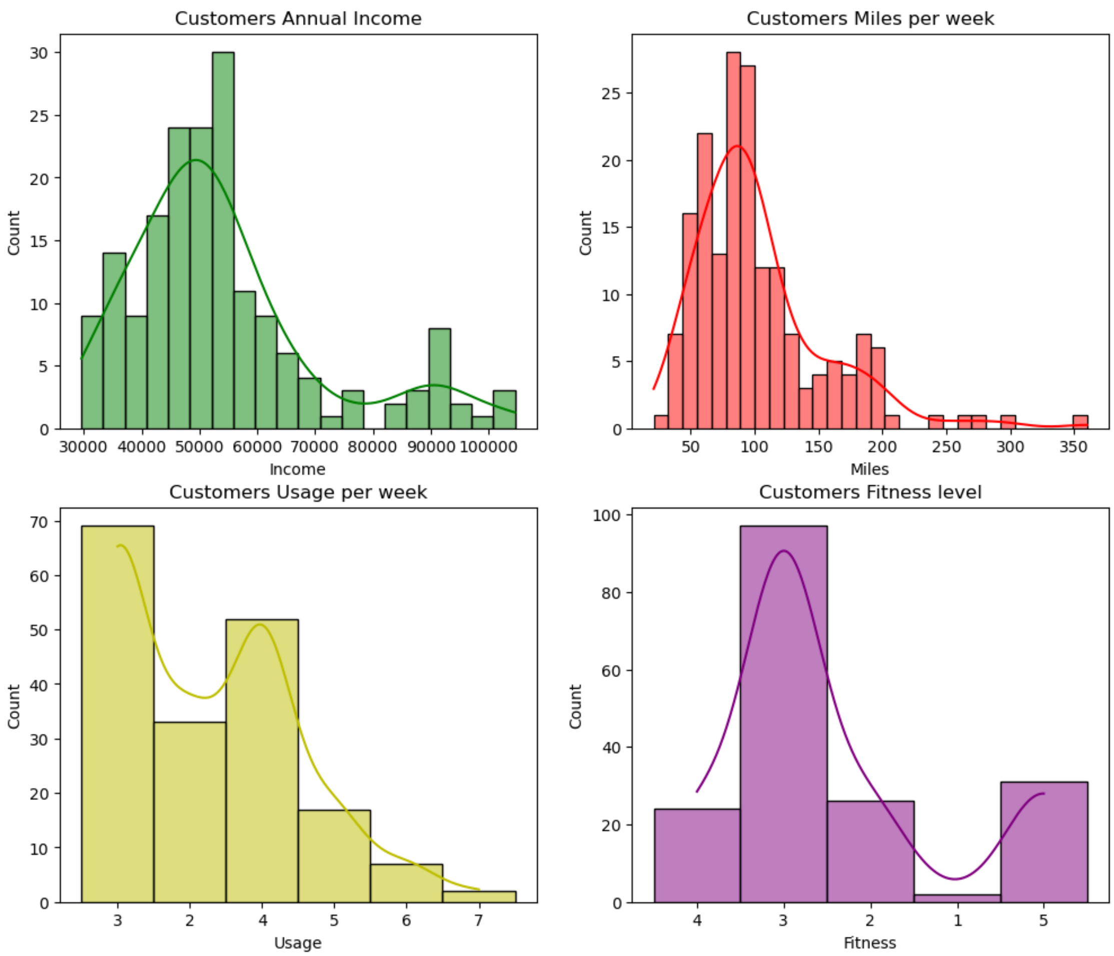

[23]:

```
fig= plt.figure(figsize=(12,5))
a1 = fig.add_subplot(131)
a1.pie(x=df['Product'].value_counts(),
        startangle=90, shadow=True, explode=[0.05,0.05,0.05],
        autopct='%1.2f%%', colors=['cyan','orange','green'],␣
↪labels=df['Product'].value_counts().index)
a1.set_title('Product Contribution')
a2= fig.add_subplot(132)
a2.pie(x=df['Gender'].value_counts(),
        startangle=90, shadow=True, explode=[0.05,0.05],
        autopct='%1.2f%%', colors=['Grey','Pink'], labels=df['Gender'].
↪value_counts().index)
a2.set_title('Gender Contribution')
a2= fig.add_subplot(133)
a2.pie(x=df['MaritalStatus'].value_counts(),
        startangle=90, shadow=True, explode=[0.05,0.05],
        autopct='%1.2f%%', colors=['yellow','blue'], labels=df['MaritalStatus'].
↪value_counts().index)
a2.set_title('Marital Status Contribution')
plt.show()
```



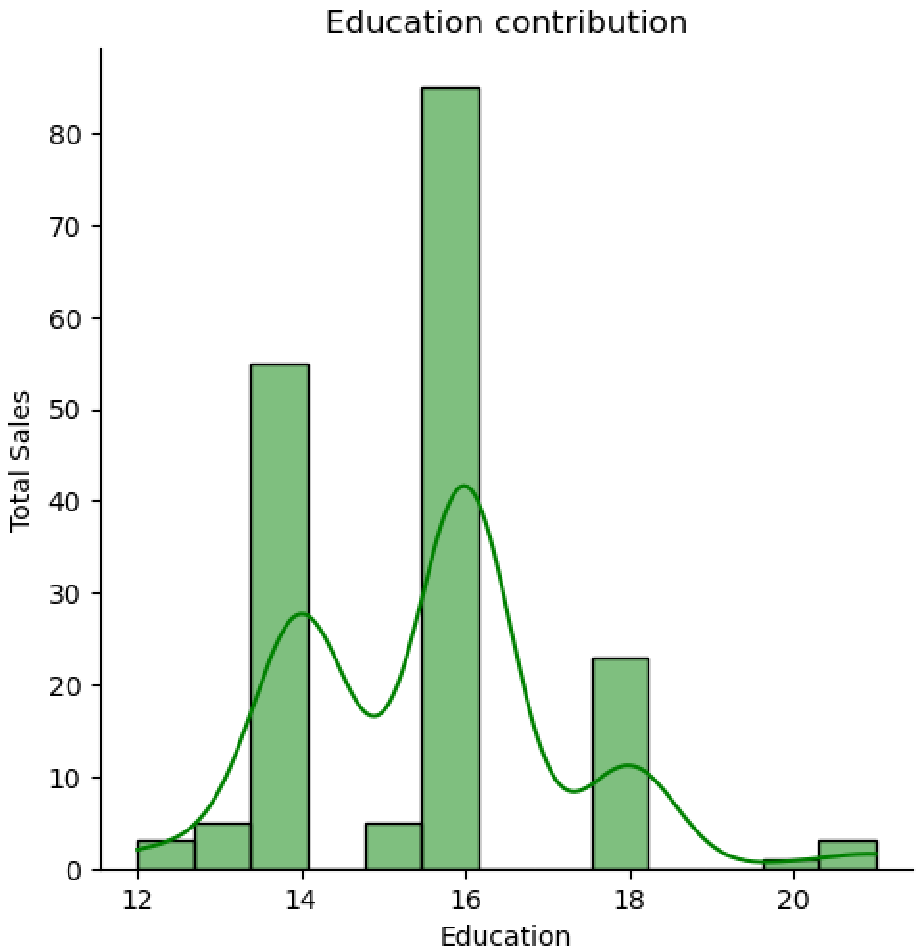

[24]:
```python
fig,ax = plt.subplots(nrows=2, ncols=2, figsize=(12,10))
sns.histplot(df['Income'], kde=True, bins=20, ax=ax[0,0], color='g'); ax[0,0].
 ↪set_title("Customers Annual Income")
sns.histplot(df['Miles'], kde=True, bins=30, ax=ax[0,1], color='r'); ax[0,1].
 ↪set_title("Customers Miles per week")
sns.histplot(df['Usage'], kde=True, ax=ax[1,0], color='y'); ax[1,0].
 ↪set_title("Customers Usage per week")
sns.histplot(df['Fitness'], kde=True, ax=ax[1,1], color='purple'); ax[1,1].
 ↪set_title("Customers Fitness level")
```

[24]:
```
Text(0.5, 1.0, 'Customers Fitness level')
```



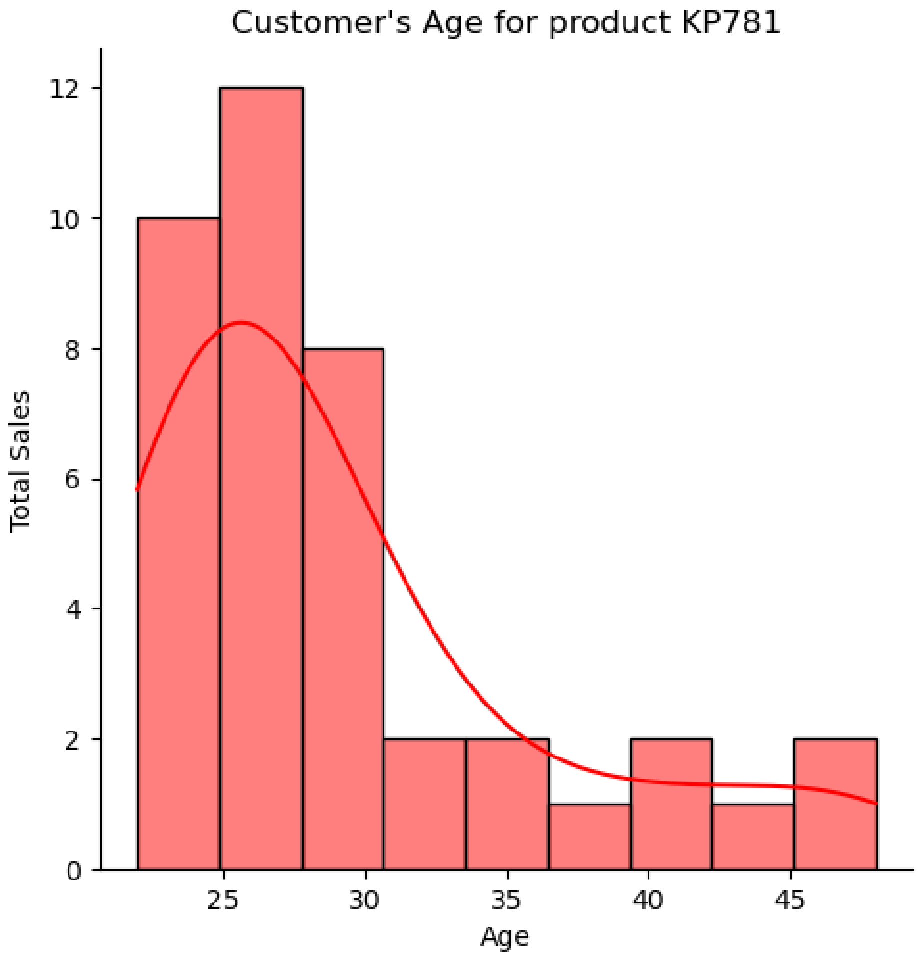

```
[25]: sns.displot(x=df['Education'], kde=True, color='g')
      plt.ylabel('Total Sales'); plt.title('Education contribution'); plt.
       ↪xlabel("Education")
```

```
[25]: Text(0.5, 9.444444444444438, 'Education')
```




[88]:
```
sns.displot(x=df[df['Product']=='KP781']['Age'], kde=True, color='r')
plt.ylabel('Total Sales'); plt.title("Customer's Age for product KP781"); plt.
 ↪xlabel("Age")
```

[88]: Text(0.5, 9.444444444444438, 'Age')




### 4.1 Each products customer preference based on customers data

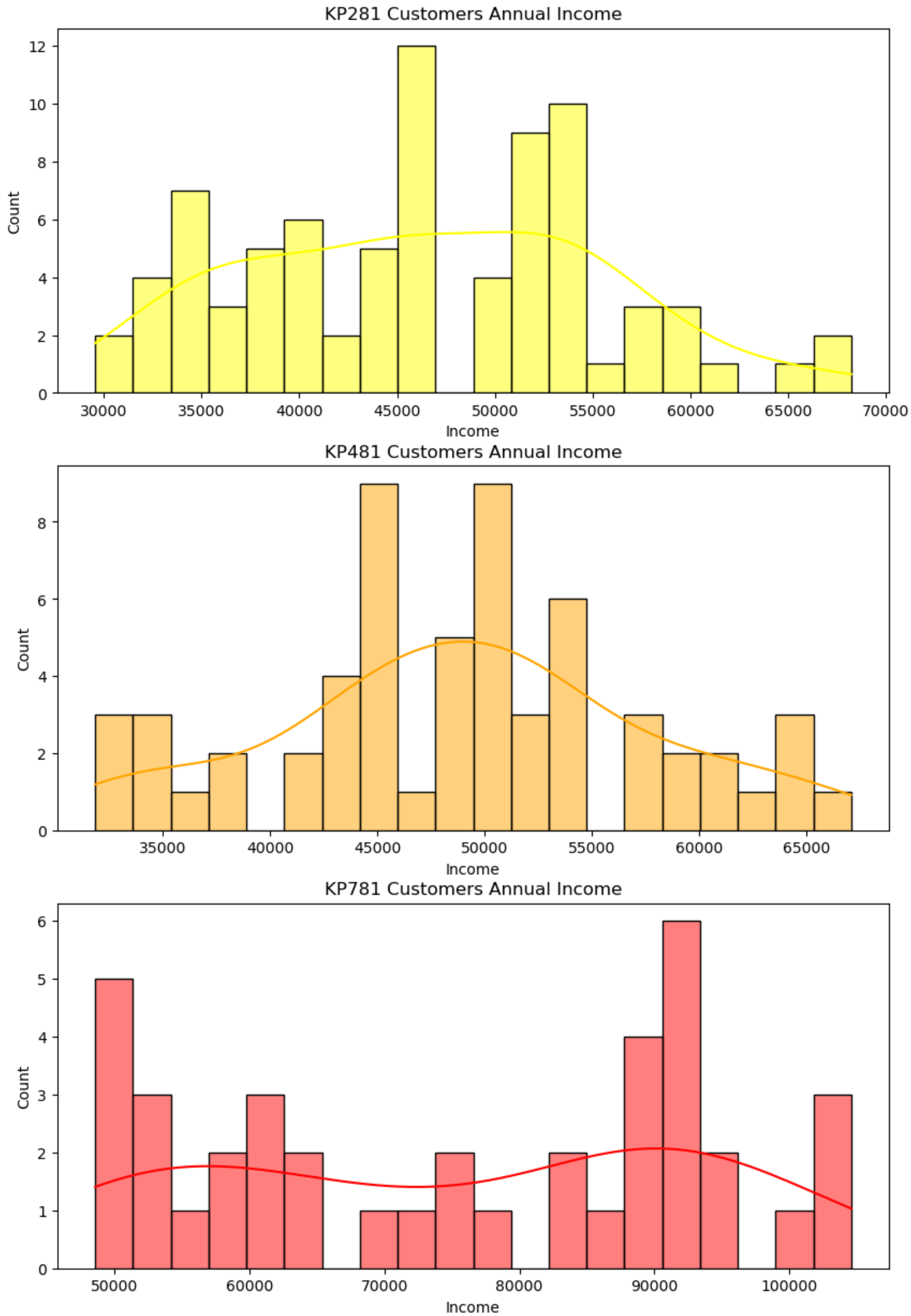
```
1.KP281 & KP481 products are bought by 30k to 70k earning customers.
2. KP781 product bought by customers who earning more than 50k.
3. KP281 & KP481 product customers mostly used to run 50 to 120 miles per week, but KP781 produ
4. Upto 4 days per week usage can prefer to KP281 & KP481.
5. More than 4 days per week usage is preferable to use KP781.
6. Beginner and Intermediate fitness people prefer KP281 and KP481.
7. Advance Fitness people prefer KP781.
```

[89]:
```
p1 = df[df['Product']=='KP281']
p2 = df[df['Product']=='KP481']
p3 = df[df['Product']=='KP781']
```

[90]:
```
fig,ax = plt.subplots(nrows=3, ncols=1, figsize=(10,15))
sns.histplot(p1['Income'], kde=True, bins=20, ax=ax[0], color='yellow'); ax[0].
↪set_title("KP281 Customers Annual Income")
```

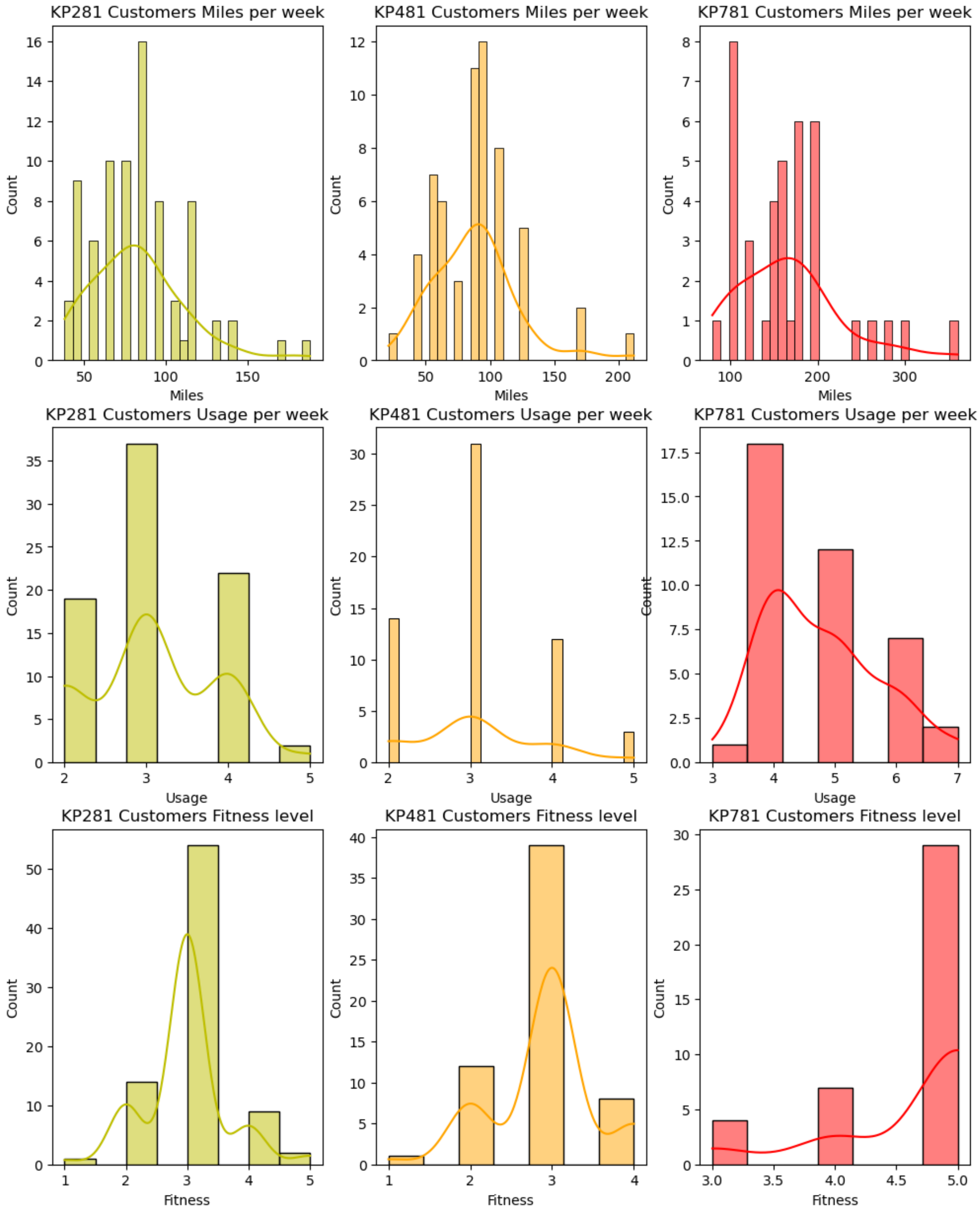
```
sns.histplot(p2['Income'], kde=True, bins=20, ax=ax[1], color='orange'); ax[1].
 ↪set_title("KP481 Customers Annual Income")
sns.histplot(p3['Income'], kde=True, bins=20, ax=ax[2], color='red'); ax[2].
 ↪set_title("KP781 Customers Annual Income")
```

[90]: Text(0.5, 1.0, 'KP781 Customers Annual Income')


KP281 Customers Annual Income
Count
Income
KP481 Customers Annual Income
Count
Income
KP781 Customers Annual Income
Count
Income

[91]:
```
fig,ax = plt.subplots(nrows=3, ncols=3, figsize=(12,15))

sns.histplot(p1['Miles'], kde=True, bins=30, ax=ax[0,0], color='y'); ax[0,0].
↪set_title("KP281 Customers Miles per week")
sns.histplot(p2['Miles'], kde=True, bins=30, ax=ax[0,1], color='orange');␣
↪ax[0,1].set_title("KP481 Customers Miles per week")
sns.histplot(p3['Miles'], kde=True, bins=30, ax=ax[0,2], color='r'); ax[0,2].
↪set_title("KP781 Customers Miles per week")

sns.histplot(p1['Usage'], kde=True, ax=ax[1,0], color='y'); ax[1,0].
↪set_title("KP281 Customers Usage per week")
sns.histplot(p2['Usage'], kde=True, ax=ax[1,1], color='orange'); ax[1,1].
↪set_title("KP481 Customers Usage per week")
sns.histplot(p3['Usage'], kde=True, ax=ax[1,2], color='r'); ax[1,2].
↪set_title("KP781 Customers Usage per week")

sns.histplot(p1['Fitness'], kde=True, ax=ax[2,0], color='y'); ax[2,0].
↪set_title("KP281 Customers Fitness level")
sns.histplot(p2['Fitness'], kde=True, ax=ax[2,1], color='orange'); ax[2,1].
↪set_title("KP481 Customers Fitness level")
sns.histplot(p3['Fitness'], kde=True, ax=ax[2,2], color='r'); ax[2,2].
↪set_title("KP781 Customers Fitness level")
```

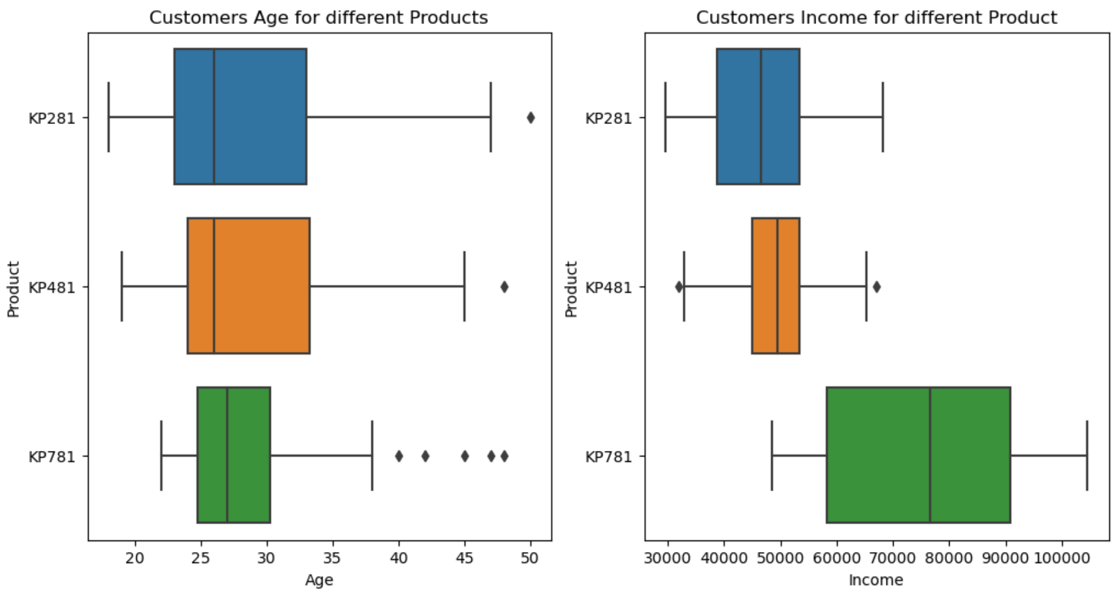
[91]:
```
Text(0.5, 1.0, 'KP781 Customers Fitness level')
```


KP281 Customers Miles per week
KP481 Customers Miles per week
KP781 Customers Miles per week
KP281 Customers Usage per week
KP481 Customers Usage per week
KP781 Customers Usage per week
KP281 Customers Fitness level
KP481 Customers Fitness level
KP781 Customers Fitness level
Count
Miles
Usage
Fitness

# 5 Visual Analysis - Bivariate

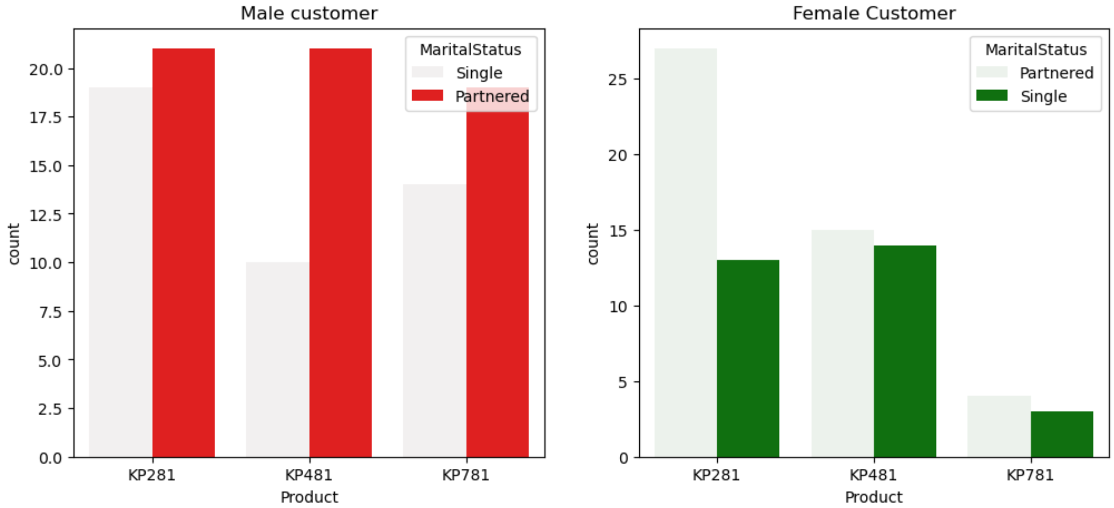
[27]:
```
fig,ax = plt.subplots(nrows=1, ncols=2, figsize=(12,6))
sns.boxplot(data=df, x='Age', y='Product', ax=ax[0]); ax[0].
↪set_title('Customers Age for different Products')
sns.boxplot(data=df, x='Income', y='Product', ax=ax[1]); ax[1].
↪set_title('Customers Income for different Product')
```

[27]: Text(0.5, 1.0, 'Customers Income for different Product')



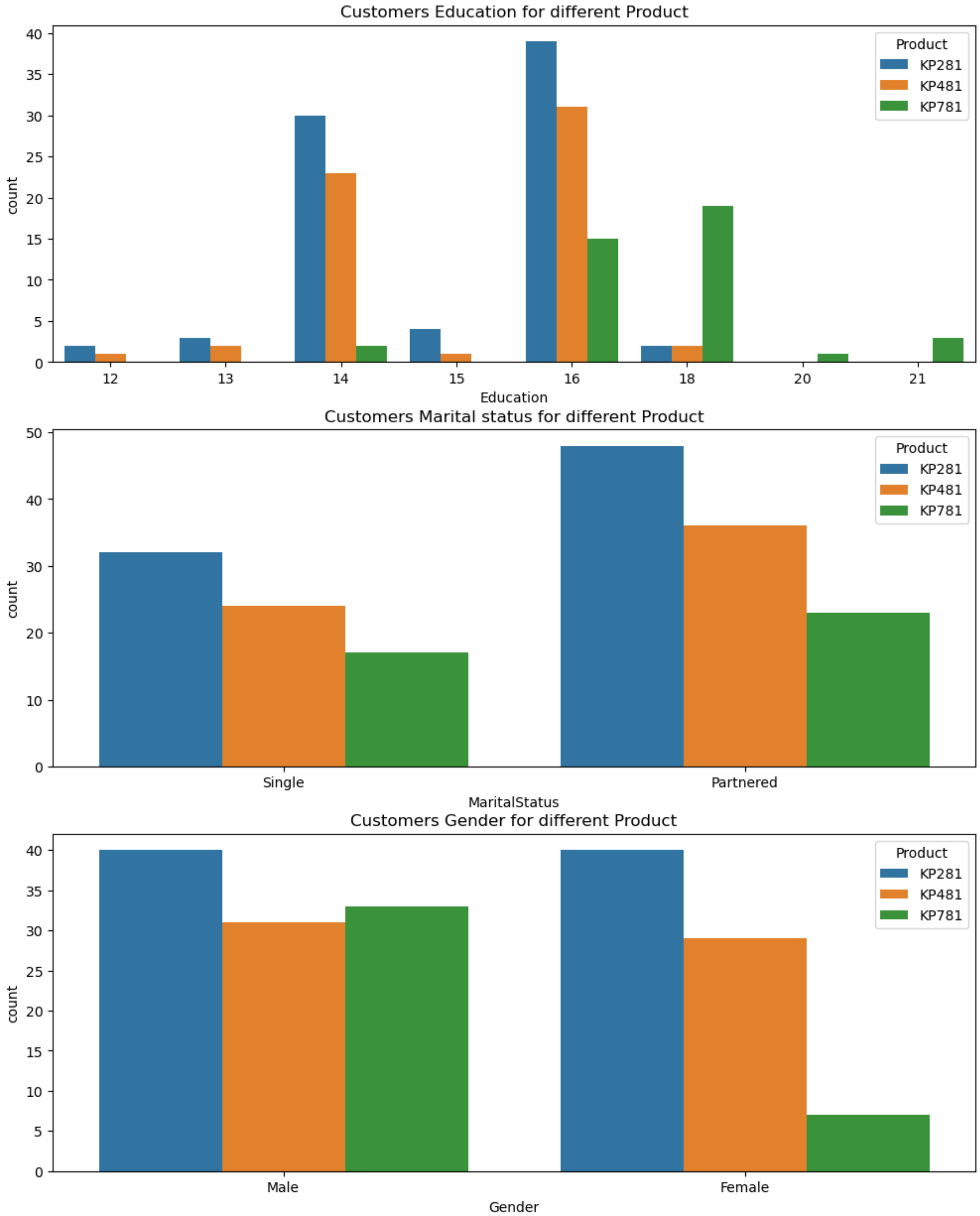

[28]:
```
dff=df[df['Gender']=='Female']
dfm=df[df['Gender']=='Male']
fig,ax=plt.subplots(nrows=1, ncols=2, figsize=(12,5))
sns.countplot(data=dfm, x='Product', hue='MaritalStatus',ax=ax[0],color='r');␣
↪ax[0].set_title('Male customer')
sns.countplot(data=dff, x='Product', hue='MaritalStatus',ax=ax[1],color='g');␣
↪ax[1].set_title('Female Customer')
```

[28]: Text(0.5, 1.0, 'Female Customer')



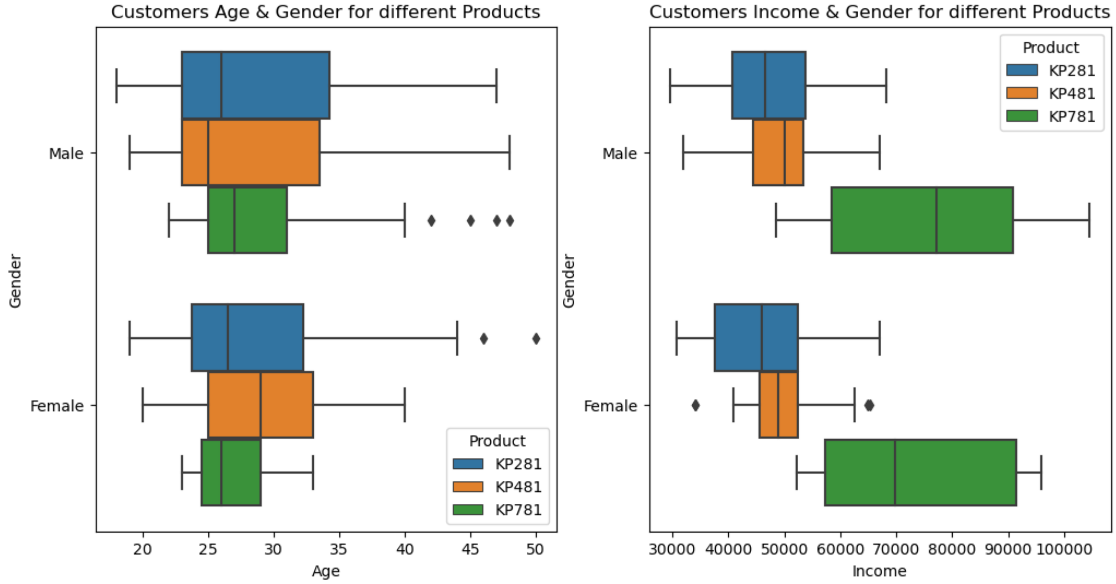

[29]:
```
fig,ax = plt.subplots(nrows=3, ncols=1, figsize=(12,15))
sns.countplot(data=df, x='Education', hue='Product', ax=ax[0]); ax[0].
 ↪set_title('Customers Education for different Product')
sns.countplot(data=df, x='MaritalStatus', hue='Product', ax=ax[1])
ax[1].set_title('Customers Marital status for different Product')
sns.countplot(data=df, x='Gender', hue='Product', ax=ax[2]); ax[2].
 ↪set_title('Customers Gender for different Product')
```

[29]:
```
Text(0.5, 1.0, 'Customers Gender for different Product')
```




[30]:
```
fig,ax = plt.subplots(nrows=1, ncols=2, figsize=(12,6))
sns.boxplot(data=df, x='Age', hue='Product', y='Gender', ax=ax[0])
ax[0].set_title('Customers Age & Gender for different Products')
sns.boxplot(data=df, x='Income', hue='Product', y='Gender', ax=ax[1])
ax[1].set_title('Customers Income & Gender for different Products')
```

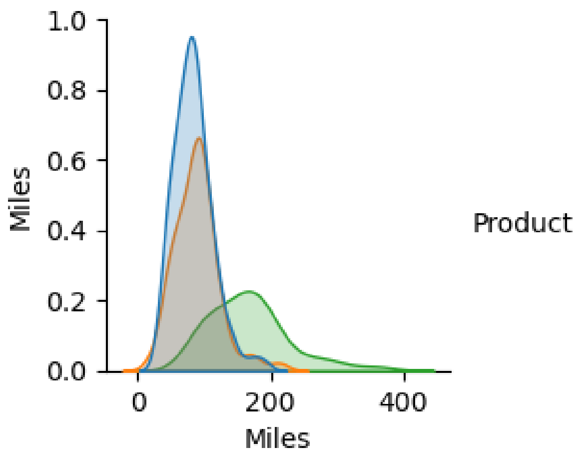
[30]: Text(0.5, 1.0, 'Customers Income & Gender for different Products')




# 6 Visual Analysis - Correlation

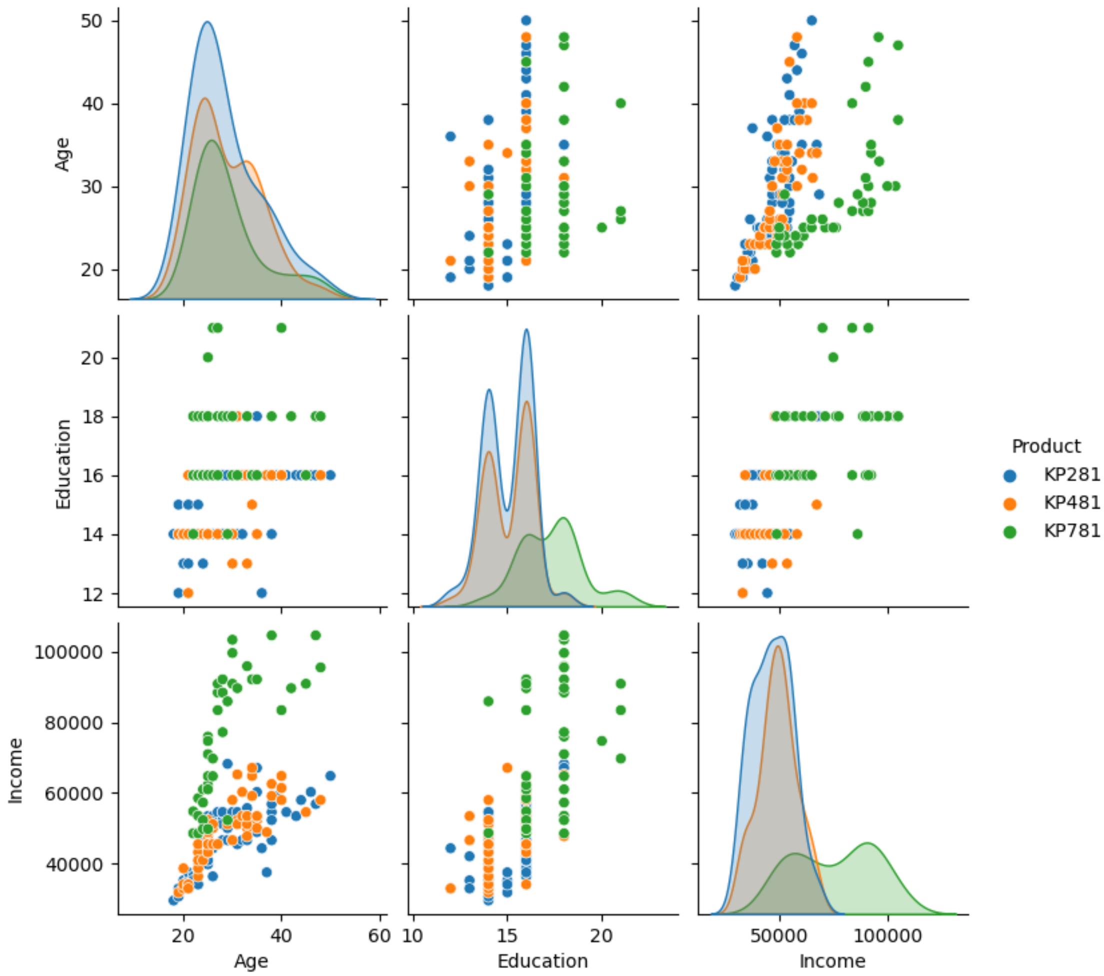
[34]:
```
sns.pairplot(data=df[['Fitness','Usage','Miles','Product']], hue='Product')
```

[34]: <seaborn.axisgrid.PairGrid at 0x2724fa99150>




[33]:
```
sns.pairplot(data=df[['Product','Age','Education','Income','Gender']],
    hue='Product')
```

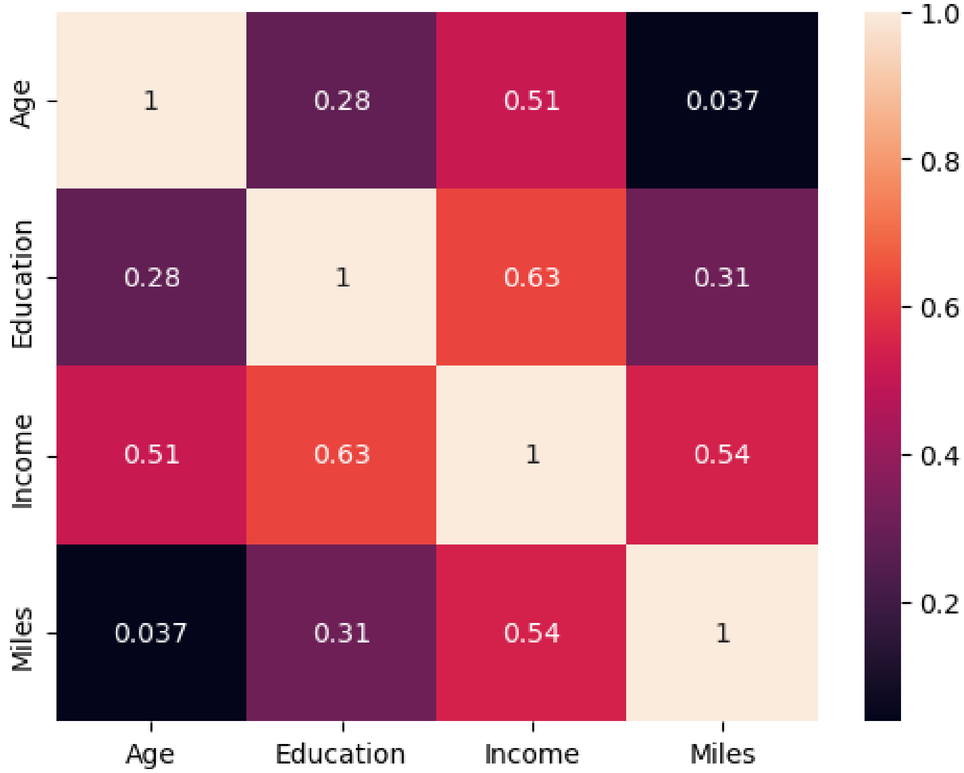
[33]: <seaborn.axisgrid.PairGrid at 0x27250f23190>




# 7 Heatmap

[35]:
```
sns.heatmap(df[['Age','Education','Usage','Fitness','Income','Miles']].corr(),
    annot=True)
```

[35]: <Axes: >




- From the pair plot we can see Age and Income are **positively correlated** and heatmap also suggests a **strong correlation** betwwen them
- Eductaion and Income are highly correlated as its obvious. Eductation also has significatnt correlation between Fitness rating and Usage of the treadmill.
- Usage is highly correlated with Fitness and Miles as more the usage more the fitness and mileage.

# 8 Two-Way Contingency Tables

## 8.1 Product VS Gender and MaritalStatus

[45]:
```
ct_p_gm = pd.crosstab(df['Product'], [df['Gender'],df['MaritalStatus']],
                      margins=True, margins_name='Total_Purchases')
ct_p_gm
```

[45]:

| Gender | Female | | Male | | Total_Purchases |
|---|---|---|---|---|---|
| MaritalStatus | Partnered | Single | Partnered | Single | |
| Product | | | | | |
| KP281 | 27 | 13 | 21 | 19 | 80 |
| KP481 | 15 | 14 | 21 | 10 | 60 |
| KP781 | 4 | 3 | 19 | 14 | 40 |

```
Total_Purchases          46      30          61      43               180
```

### 8.2 Normalized Contigency Table

[46]:
```
ct_p_gm_norm = pd.crosstab(df['Product'], [df['Gender'],df['MaritalStatus']],
                                margins=True, margins_name='Total_Purchases',␣
 ↪normalize=True)
ct_p_gm_fnorm = ct_p_gm_norm.applymap(lambda x: f"{x*100:.2f}%")
ct_p_gm_fnorm
```

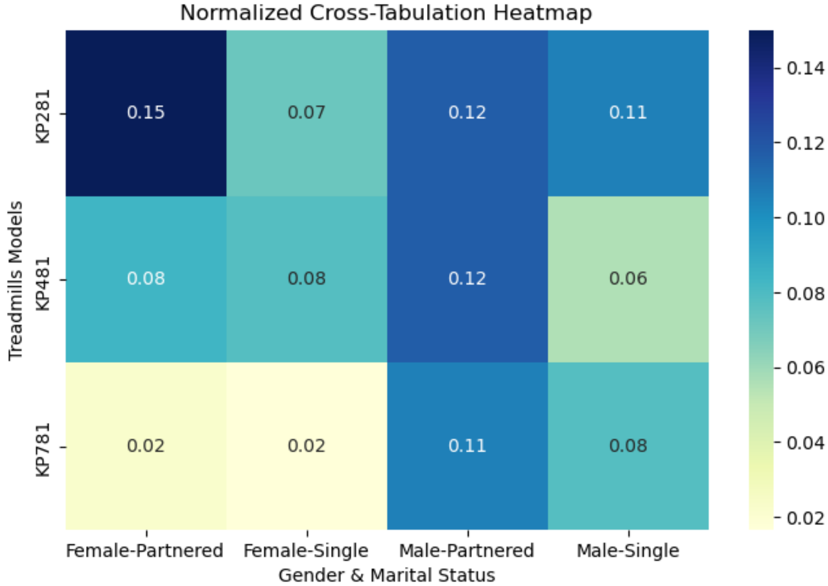
[46]:

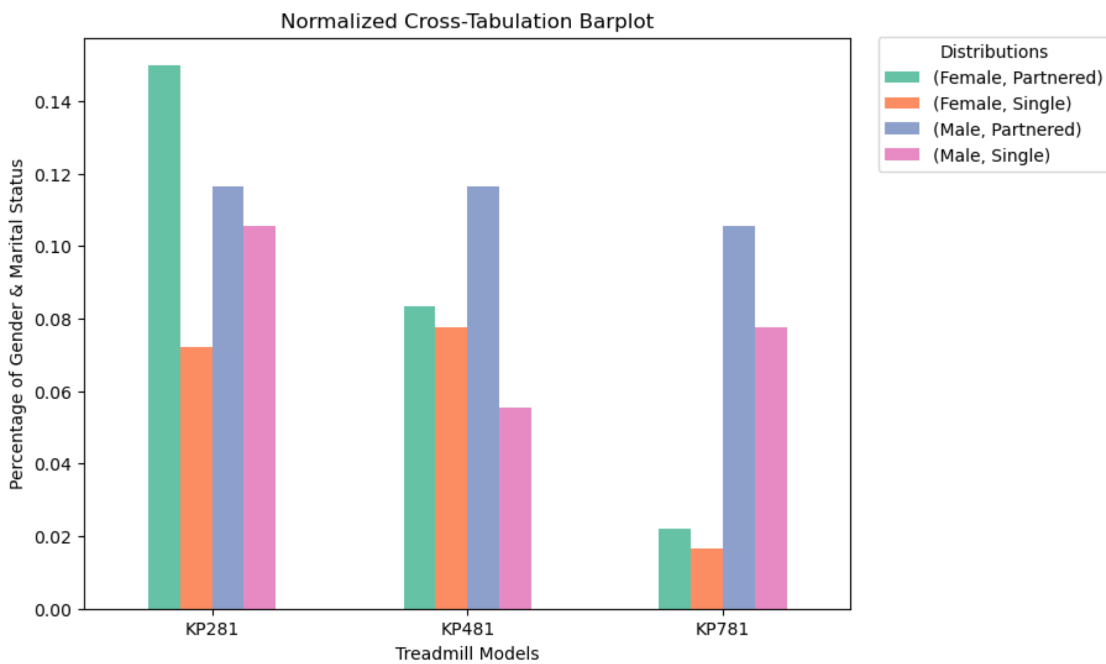
| Gender | Female | | Male | | Total_Purchases |
|---|---|---|---|---|---|
| MaritalStatus | Partnered | Single | Partnered | Single | |
| Product | | | | | |
| KP281 | 15.00% | 7.22% | 11.67% | 10.56% | 44.44% |
| KP481 | 8.33% | 7.78% | 11.67% | 5.56% | 33.33% |
| KP781 | 2.22% | 1.67% | 10.56% | 7.78% | 22.22% |
| Total_Purchases | 25.56% | 16.67% | 33.89% | 23.89% | 100.00% |

## 9 Graphical Representation of Contigency Table

[47]:
```
ct_p_gm_nomargin = pd.crosstab(df['Product'],␣
 ↪[df['Gender'],df['MaritalStatus']],
                                  normalize=True)
custom_palette = sns.color_palette("Set2")
plt.figure(figsize=(8,5))
plt.suptitle('Normalized Cross-Tabulation of Treadmill Models by Gender &␣
 ↪Marital Status', fontsize=16)

sns.heatmap(ct_p_gm_nomargin, annot=True, fmt='.2f', cmap='YlGnBu')
plt.title('Normalized Cross-Tabulation Heatmap')
plt.xlabel('Gender & Marital Status')
plt.xticks(rotation = 0)
plt.ylabel('Treadmills Models')

ax = ct_p_gm_nomargin.plot(kind='bar', figsize=(8,6),stacked=False, color=␣
 ↪custom_palette)
plt.title('Normalized Cross-Tabulation Barplot')
plt.xlabel('Treadmill Models')
plt.xticks(rotation= False)
plt.ylabel('Percentage of Gender & Marital Status')
plt.legend(title='Distributions', loc='upper right', bbox_to_anchor=(1.35, 1.
 ↪017))

plt.show()
```

Normalized Cross-Tabulation of Treadmill Models by Gender & Marital Status



### 9.1 Fitness VS Product and Gender

[48]:
```python
ct_f_gp = pd.crosstab(df['Fitness'], [df['Gender'],df['Product']],
                      margins=True, margins_name='Total_Purchases')
ct_f_gp
```

[48]:

| Gender | Female | | | Male | | | Total_Purchases |
|---|---|---|---|---|---|---|---|
| Product | KP281 | KP481 | KP781 | KP281 | KP481 | KP781 | |
| Fitness | | | | | | | |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 2 |
| 2 | 10 | 6 | 0 | 4 | 6 | 0 | 26 |
| 3 | 26 | 18 | 1 | 28 | 21 | 3 | 97 |
| 4 | 3 | 4 | 1 | 6 | 4 | 6 | 24 |
| 5 | 1 | 0 | 5 | 1 | 0 | 24 | 31 |
| Total_Purchases | 40 | 29 | 7 | 40 | 31 | 33 | 180 |

### 9.2 Normalized Contigency Table

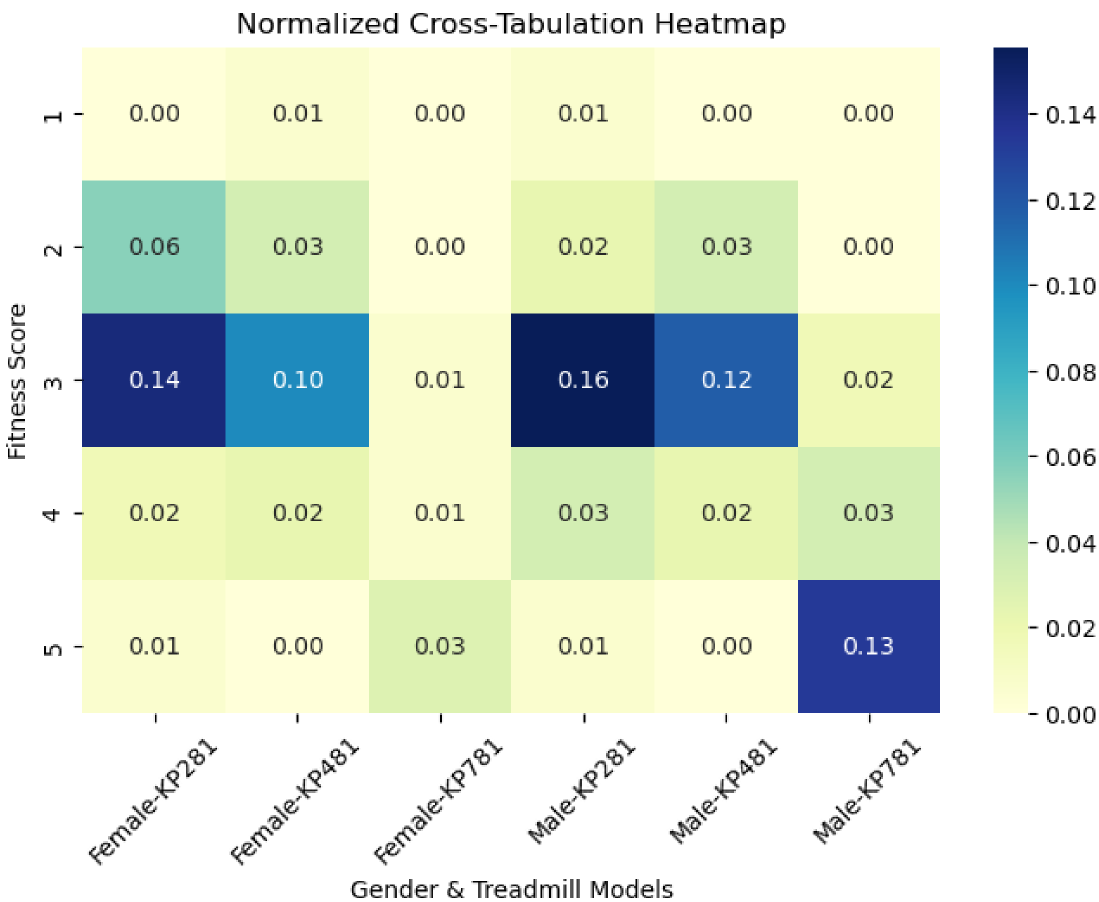
[49]:
```python
ct_f_gp_norm = pd.crosstab(df['Fitness'], [df['Gender'],df['Product']],
                           margins=True, margins_name='Total_Purchases',␣
 ↪normalize=True)
ct_f_gp_fnorm = ct_f_gp_norm.applymap(lambda x: f"{x*100:.2f}%")
ct_f_gp_fnorm
```

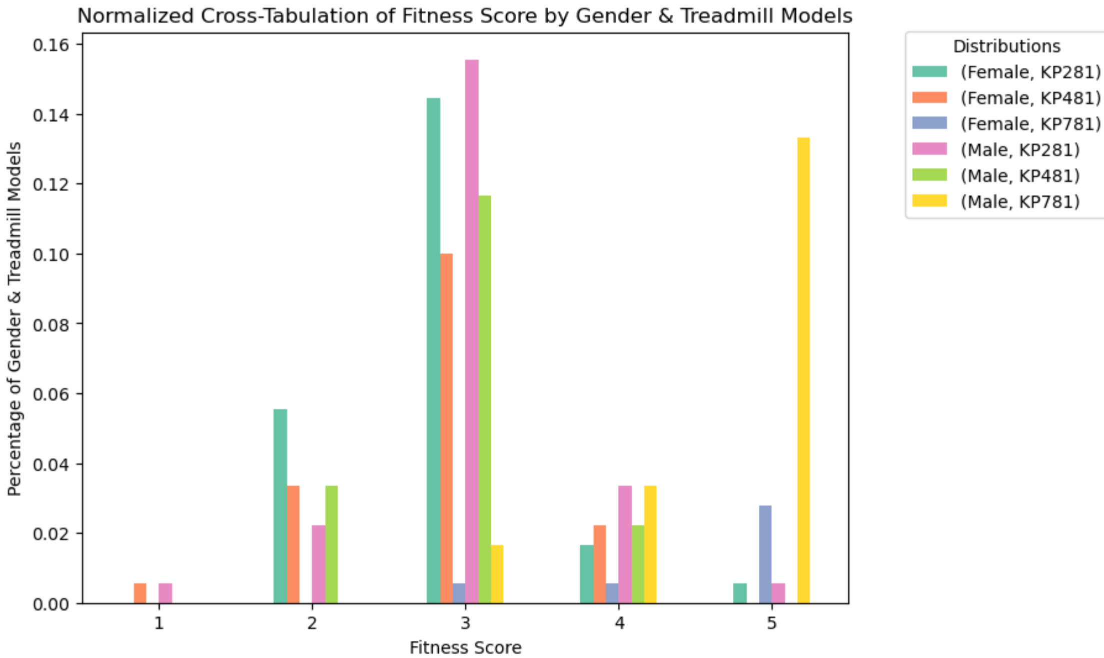
[49]:

| Gender | Female | | | Male | | | Total_Purchases |
|---|---|---|---|---|---|---|---|
| Product | KP281 | KP481 | KP781 | KP281 | KP481 | KP781 | |
| Fitness | | | | | | | |
| 1 | 0.00% | 0.56% | 0.00% | 0.56% | 0.00% | 0.00% | 1.11% |
| 2 | 5.56% | 3.33% | 0.00% | 2.22% | 3.33% | 0.00% | 14.44% |
| 3 | 14.44% | 10.00% | 0.56% | 15.56% | 11.67% | 1.67% | 53.89% |
| 4 | 1.67% | 2.22% | 0.56% | 3.33% | 2.22% | 3.33% | 13.33% |
| 5 | 0.56% | 0.00% | 2.78% | 0.56% | 0.00% | 13.33% | 17.22% |
| Total_Purchases | 22.22% | 16.11% | 3.89% | 22.22% | 17.22% | 18.33% | 100.00% |

[50]:
```python
ct_f_gp_nomargin = pd.crosstab(df['Fitness'], [df['Gender'],df['Product']],
                               normalize=True)
custom_palette = sns.color_palette("Set2")

plt.figure(figsize=(8,5))
sns.heatmap(ct_f_gp_nomargin, annot=True, fmt='.2f', cmap='YlGnBu', )
plt.title('Normalized Cross-Tabulation Heatmap')
plt.xlabel('Gender & Treadmill Models')
plt.xticks(rotation = 45)
plt.ylabel('Fitness Score')

ax = ct_f_gp_nomargin.plot(kind='bar', figsize=(8,6),stacked=False, color=␣
 ↪custom_palette)
```

```
plt.title('Normalized Cross-Tabulation of Fitness Score by Gender & Treadmill␣
↪Models')
plt.xlabel('Fitness Score')
plt.xticks(rotation= False)
plt.ylabel('Percentage of Gender & Treadmill Models')
plt.legend(title='Distributions', loc='upper right', bbox_to_anchor=(1.35, 1.
↪017))

plt.show()
```






# 10 4. Missing values & Outlier Detection

[36]:
```
df.isna().sum(axis=0)
```

[36]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

[37]:
```
df.describe().loc[['mean','50%']]
```

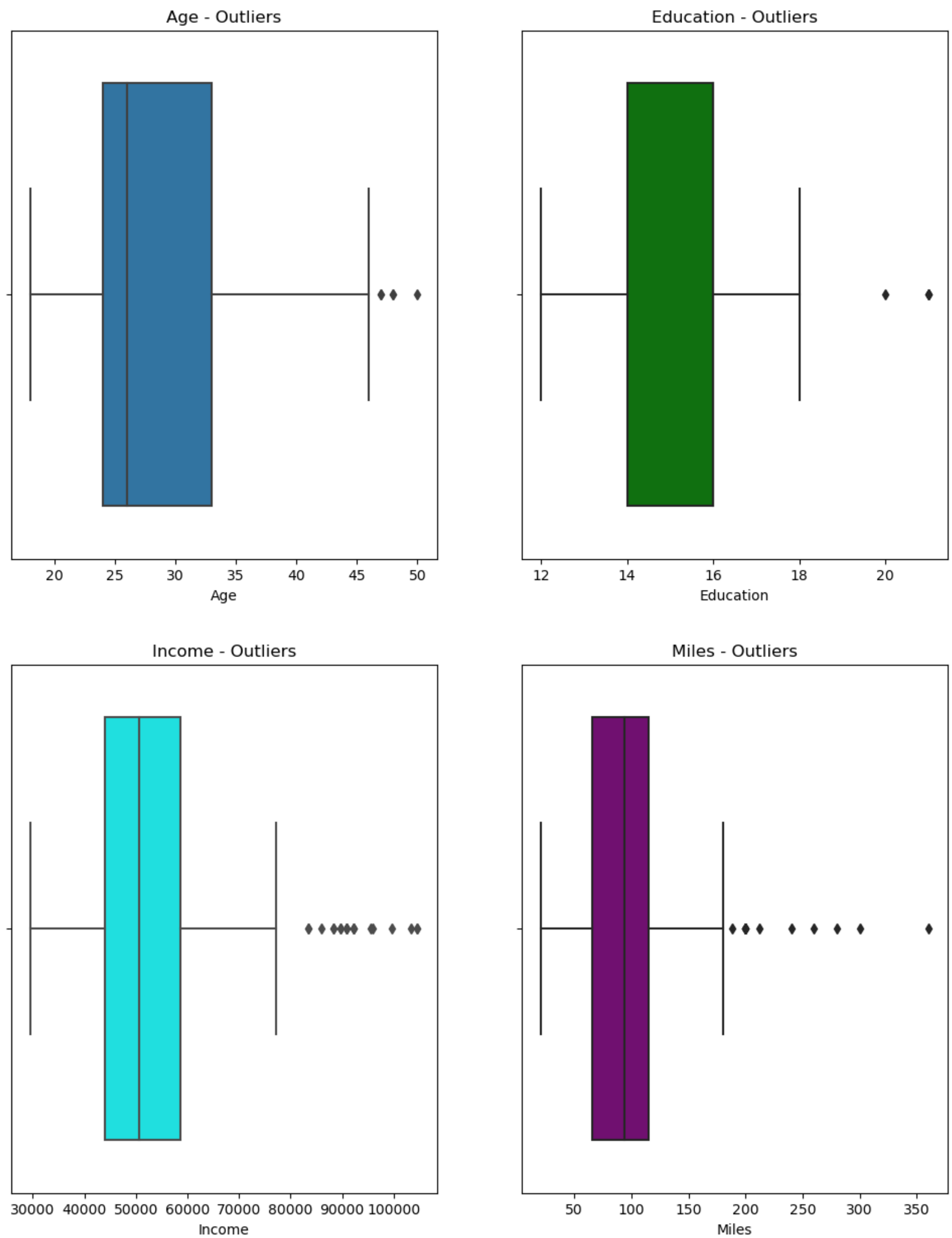
[37]:

| | Age | Education | Income | Miles |
|---|---|---|---|---|
| mean | 28.788889 | 15.572222 | 53719.577778 | 103.194444 |
| 50% | 26.000000 | 16.000000 | 50596.500000 | 94.000000 |

[40]:
```
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(12,15))
sns.boxplot(data=df, x='Age', ax=ax[0,0]);                        ax[0,0].
↪set_title('Age - Outliers')
```

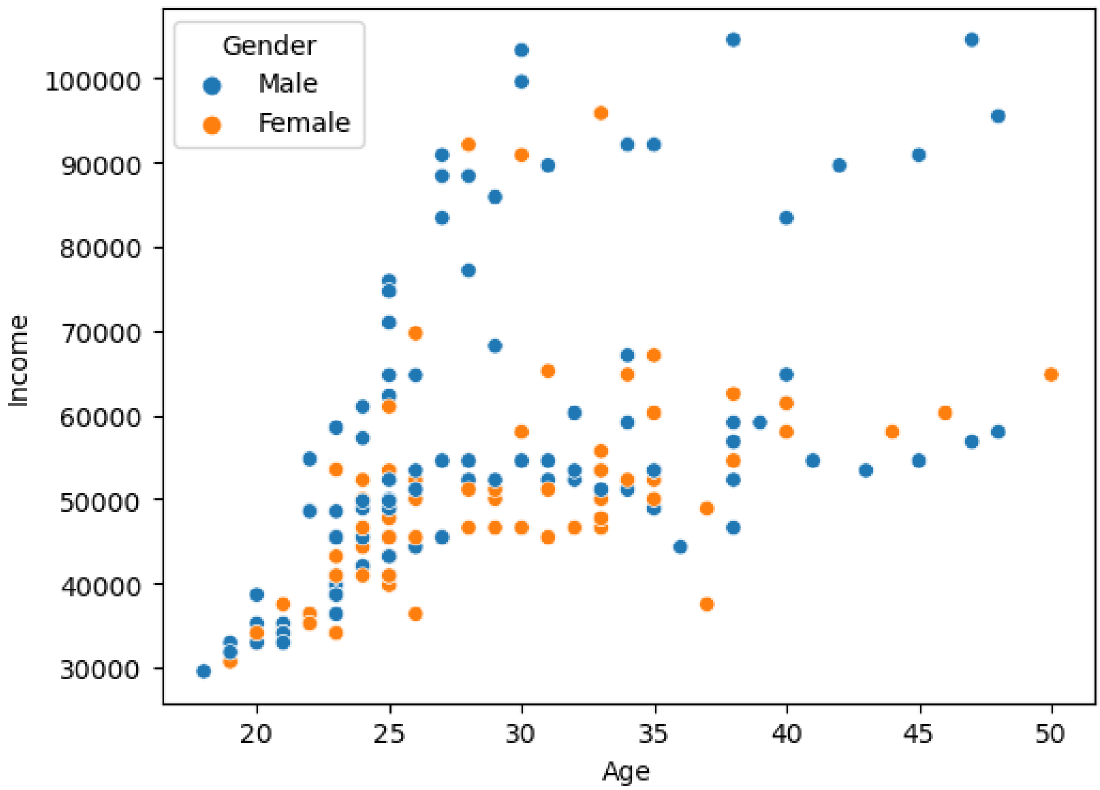
```
sns.boxplot(data=df, x='Education', ax=ax[0,1], color='green'); ax[0,1].
 ↪set_title('Education - Outliers')
sns.boxplot(data=df, x='Income', ax=ax[1,0], color='cyan');     ax[1,0].
 ↪set_title('Income - Outliers')
sns.boxplot(data=df, x='Miles', ax=ax[1,1], color='purple');    ax[1,1].
 ↪set_title('Miles - Outliers')
```

[40]: Text(0.5, 1.0, 'Miles - Outliers')




[41]:
```python
sns.scatterplot(data=df, x='Age',y='Income', hue='Gender')
```

[41]: <Axes: xlabel='Age', ylabel='Income'>




# 11 5. Computing Probability - Marginal, Conditional Probability

### 11.0.1 Creating Age group for grouping customers

[59]:
```
df['AgeGroup'] = pd.cut(df['Age'], bins=[10,20,30,40,50],
                        labels=['10-20','20-30','30-40','40-50'])
df['AgeGroup'].value_counts().reset_index()
```

[59]:

| | index | AgeGroup |
|---|---|---|
| 0 | 20-30 | 110 |
| 1 | 30-40 | 48 |
| 2 | 40-50 | 12 |
| 3 | 10-20 | 10 |

[60]:
```
# Assumptions
print('Low Income (Income < 45k) =',(df['Income']<45000).sum(), 'Customers')
print('Average Income (45k <= Income <= 60k) =',((df['Income']>=45000)
                                &(df['Income']<=60000)).sum(), 'Customers')
print('High Income (Income > 60k) =',(df['Income']>60000).sum(), 'Customers')
```

```
Low Income (Income < 45k) = 49 Customers
Average Income (45k <= Income <= 60k) = 89 Customers
```

```
High Income (Income > 60k) = 42 Customers
```

[61]:
```python
conditions = [
    df['Income'] < 45000,
    (df['Income'] >= 45000) & (df['Income'] <= 60000),
    df['Income'] > 60000
]
categories = ['Low (I<45k)', 'Average (45k>I<60k)', 'High (I>60k)']
df['IncomeCategory'] = np.select(conditions, categories)
```

### 11.0.2 Conversion of Categorical attributes to 'Category'

[62]:
```python
df['Product'] = df['Product'].astype('category')
df['Gender'] = df['Gender'].astype('category')
df['MaritalStatus'] = df['MaritalStatus'].astype('category')
df['Usage'] = df['Usage'].astype('category')
df['Fitness'] = df['Fitness'].astype('category')
df['AgeGroup'] = df['AgeGroup'].astype('category')
df['Education'] = df['Education'].astype('category')
df['IncomeCategory'] = df['IncomeCategory'].astype('category')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Product         180 non-null    category
 1   Age             180 non-null    int64
 2   Gender          180 non-null    category
 3   Education       180 non-null    category
 4   MaritalStatus   180 non-null    category
 5   Usage           180 non-null    category
 6   Fitness         180 non-null    category
 7   Income          180 non-null    int64
 8   Miles           180 non-null    int64
 9   AgeGroup        180 non-null    category
 10  IncomeCategory  180 non-null    category
dtypes: category(8), int64(3)
memory usage: 7.2 KB
```

[63]:
```python
df.describe()
```

[63]:

| | Age | Income | Miles |
|---|---|---|---|
| count | 180.000000 | 180.000000 | 180.000000 |
| mean | 28.788889 | 53719.577778 | 103.194444 |
| std | 6.943498 | 16506.684226 | 51.863605 |
| min | 18.000000 | 29562.000000 | 21.000000 |

```
25%     24.000000   44058.750000   66.000000
50%     26.000000   50596.500000   94.000000
75%     33.000000   58668.000000  114.750000
max     50.000000  104581.000000  360.000000
```

[64]:
```
df.describe(include='category')
```

[64]:
```
       Product Gender  Education MaritalStatus Usage Fitness AgeGroup  \
count      180    180        180           180   180     180      180
unique       3      2          8             2     6       5        4
top      KP281   Male         16     Partnered     3       3    20-30
freq        80    104         85           107    69      97      110

             IncomeCategory
count                   180
unique                    3
top     Average (45k>I<60k)
freq                     89
```

## 11.1 Conditional Probabilities

### 11.1.1 Product VS Gender

[54]:
```
# Value Counts of Product VS Gender
ct_p_g = pd.crosstab(df['Product'],df['Gender'], margins=True,
                     margins_name='Total_Purchases')

# Probability of Product VS Gender
ctn_p_g = pd.crosstab(df['Product'],df['Gender'], margins=True,
                      margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS Gender (Value Counts)"))
display(ct_p_g)
print()
display((f"Product VS Gender (Probability)"))
display(ctn_p_g)
```

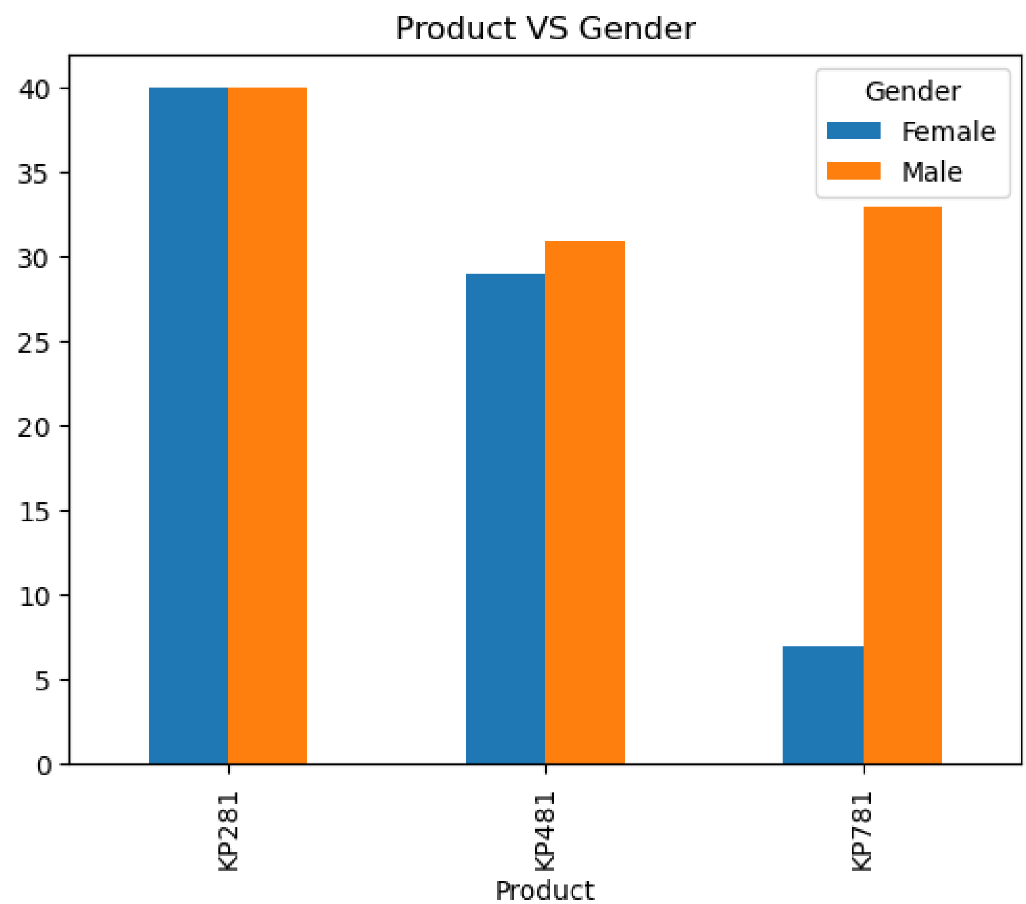
'Product VS Gender (Value Counts)'

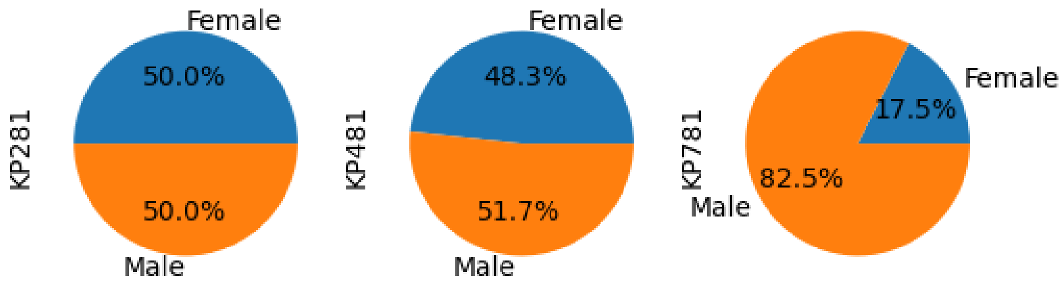
| Gender | Female | Male | Total_Purchases |
|---|---|---|---|
| Product | | | |
| KP281 | 40 | 40 | 80 |
| KP481 | 29 | 31 | 60 |
| KP781 | 7 | 33 | 40 |
| Total_Purchases | 76 | 104 | 180 |

'Product VS Gender (Probability)'

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

[55]:
```
pd.crosstab(df['Product'],df['Gender']).plot(kind='bar',title='Product VS Gender')
pd.crosstab(df['Gender'],df['Product']).plot(kind='pie', subplots=True, autopct='%1.1f%%', legend=False)
plt.show()
```






### 11.1.2 Product VS MaritalStatus

[83]:
```python
# Value Counts of Product VS MaritalStatus
ct_p_ms = pd.crosstab(df['Product'],df['MaritalStatus'], margins=True,
                      margins_name='Total_Purchases')

# Probability of Product VS MaritalStatus
ctn_p_ms = pd.crosstab(df['Product'],df['MaritalStatus'], margins=True,
                       margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS MaritalStatus (Value Counts)"))
display(ct_p_ms)
print()
display((f"Product VS MaritalStatus (Probability)"))
display(ctn_p_ms)
```

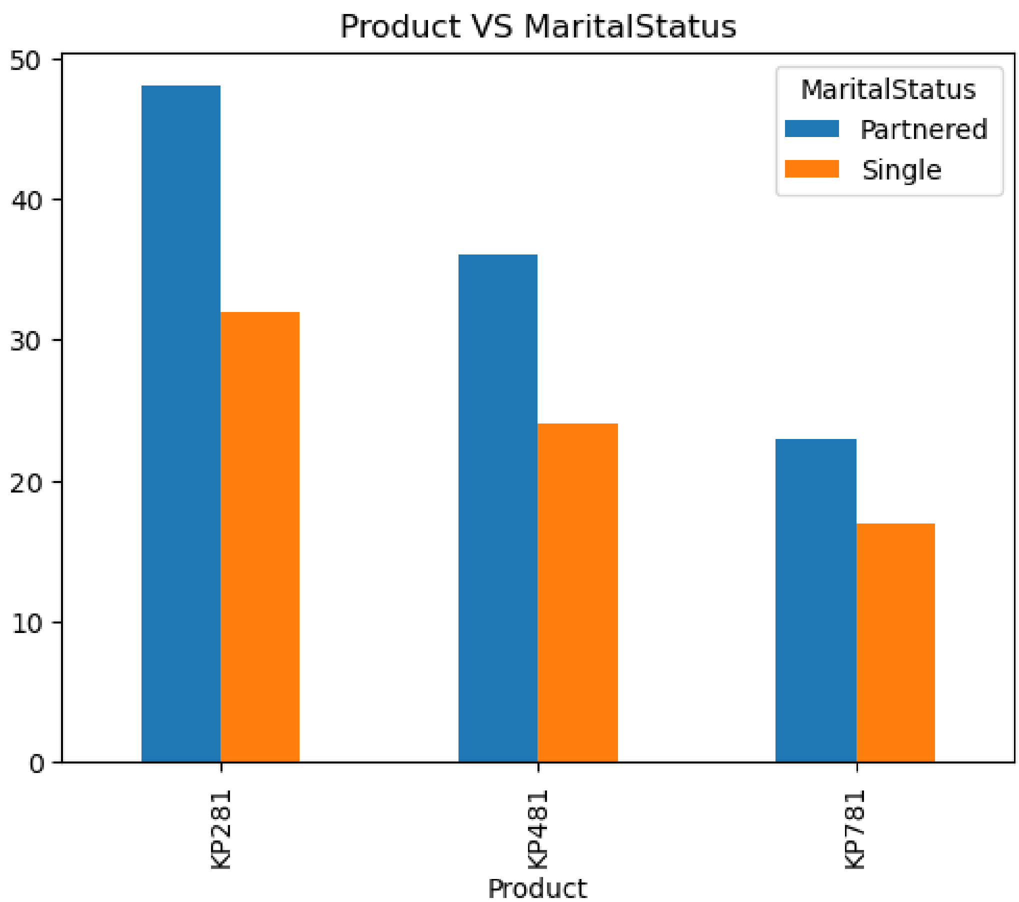
'Product VS MaritalStatus (Value Counts)'

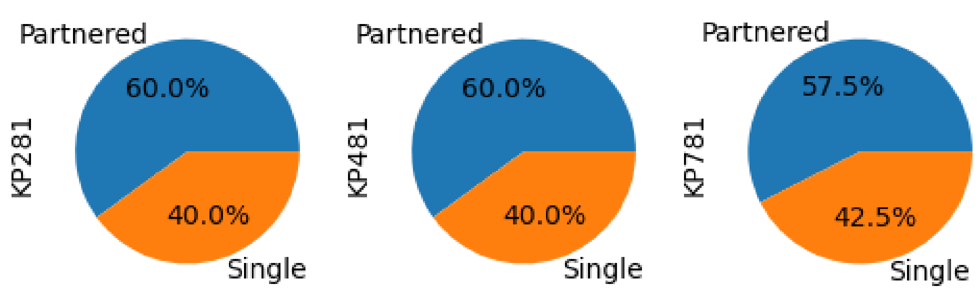
| MaritalStatus | Partnered | Single | Total_Purchases |
|---|---|---|---|
| Product | | | |
| KP281 | 48 | 32 | 80 |
| KP481 | 36 | 24 | 60 |
| KP781 | 23 | 17 | 40 |
| Total_Purchases | 107 | 73 | 180 |

'Product VS MaritalStatus (Probability)'

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

[57]:
```
pd.crosstab(df['Product'],df['MaritalStatus']).plot(kind='bar',
                              title='Product VS MaritalStatus')
pd.crosstab(df['MaritalStatus'],df['Product']).plot(kind='pie',
                              subplots=True, autopct='%1.1f%%',
                              legend=False)
plt.show()
```



### 11.1.3 Product VS Usage

[79]:
```
# Value Counts of Product VS Usage
ct_p_u = pd.crosstab(df['Product'],df['Usage'], margins=True,
                     margins_name='Total_Purchases')

# Probability of Product VS Usage
ctn_p_u = pd.crosstab(df['Product'],df['Usage'], margins=True,
                      margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS Usage (Value Counts)"))
display(ct_p_u)
print()
display((f"Product VS Usage (Probability)"))
display(ctn_p_u)
```

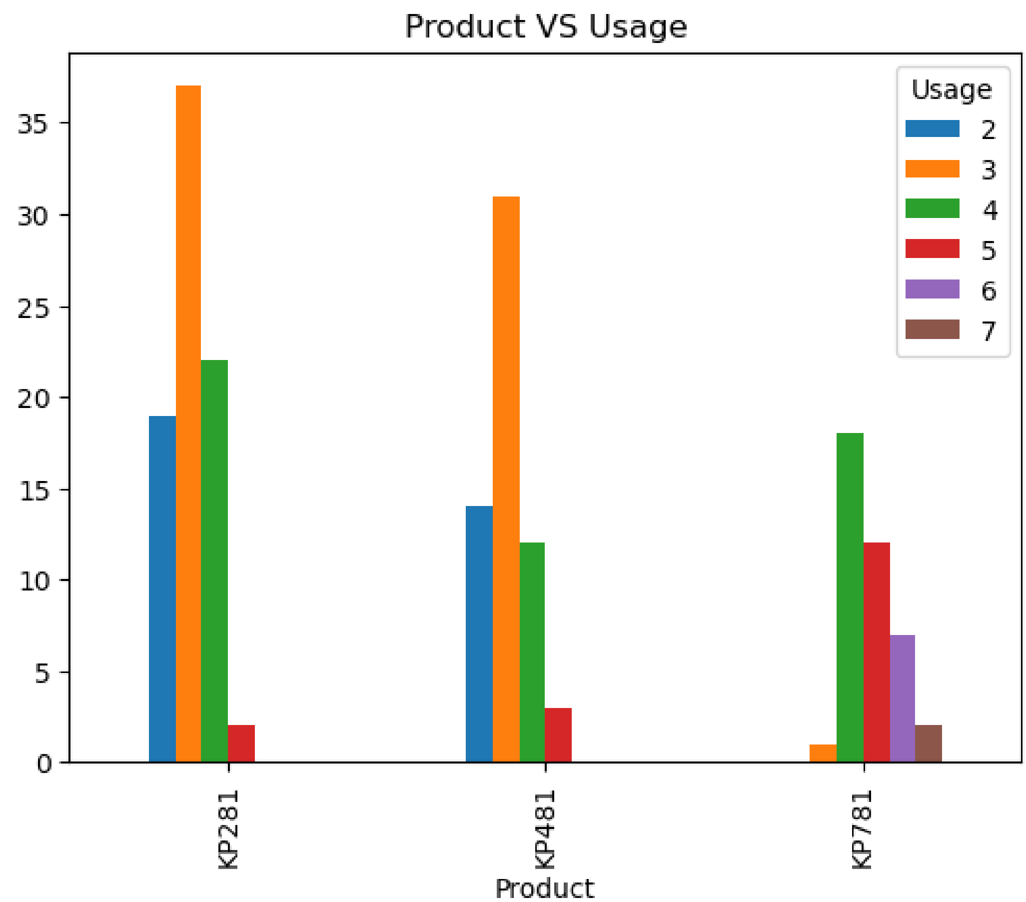
'Product VS Usage (Value Counts)'

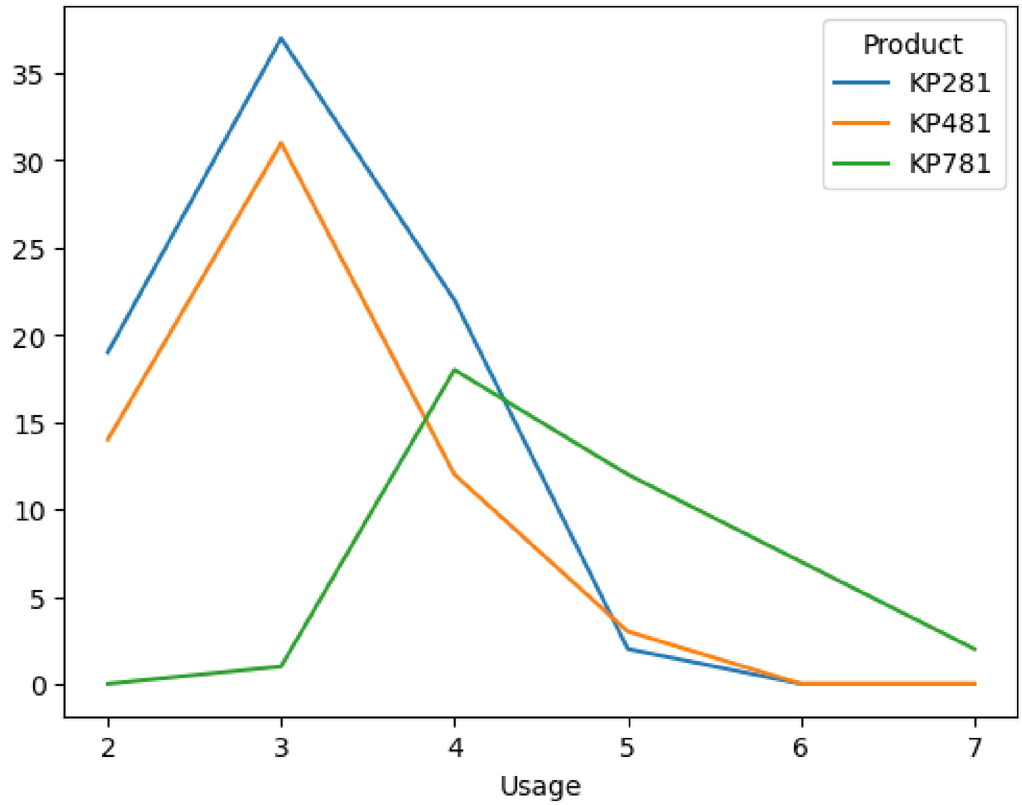
| Usage | 2 | 3 | 4 | 5 | 6 | 7 | Total_Purchases |
|---|---|---|---|---|---|---|---|
| Product | | | | | | | |
| KP281 | 19 | 37 | 22 | 2 | 0 | 0 | 80 |
| KP481 | 14 | 31 | 12 | 3 | 0 | 0 | 60 |
| KP781 | 0 | 1 | 18 | 12 | 7 | 2 | 40 |
| Total_Purchases | 33 | 69 | 52 | 17 | 7 | 2 | 180 |

'Product VS Usage (Probability)'

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

[66]:
```
pd.crosstab(df['Product'],df['Usage']).plot(kind='bar',title='Product VS Usage')
pd.crosstab(df['Usage'],df['Product']).plot(kind='line')
plt.show()
```


Product VS Usage
Usage
2
3
4
5
6
7
35
30
25
20
15
10
5
0
KP281
KP481
KP781
Product




#### 11.1.4 Product VS Fitness

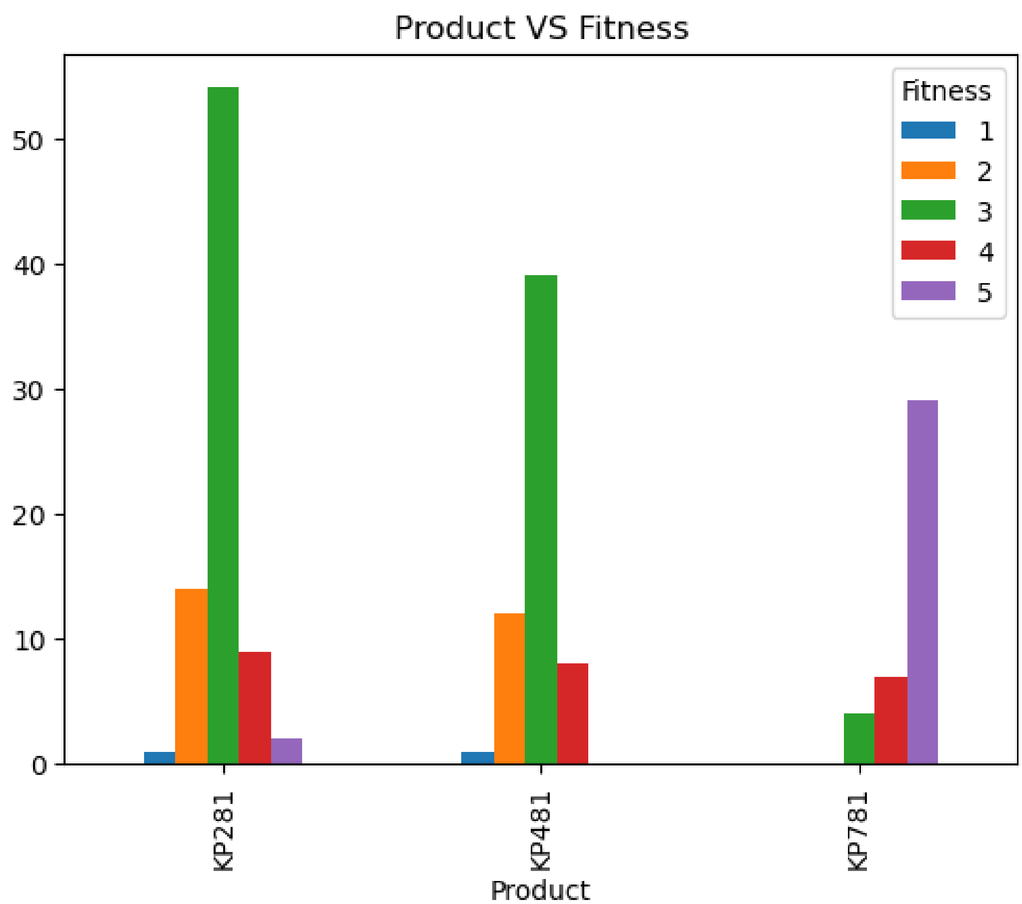
[78]:
```
# Value Counts of Product VS Fitness
ct_p_f = pd.crosstab(df['Product'],df['Fitness'], margins=True,
                     margins_name='Total_Purchases')

# Probability of Product VS Fitness
ctn_p_f = pd.crosstab(df['Product'],df['Fitness'], margins=True,
                      margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS Fitness (Value Counts)"))
display(ct_p_f)
print()
display((f"Product VS Fitness (Probability)"))
display(ctn_p_f)
```

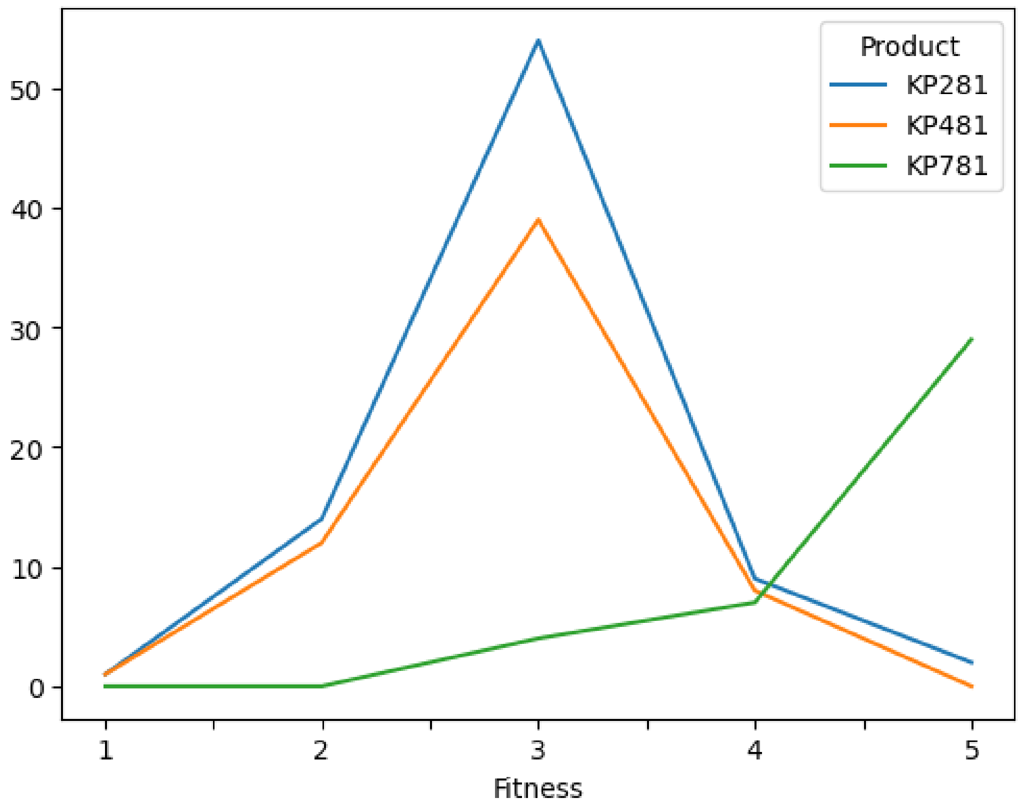
'Product VS Fitness (Value Counts)'

| Fitness | 1 | 2 | 3 | 4 | 5 | Total_Purchases |
|---|---|---|---|---|---|---|
| Product | | | | | | |
| KP281 | 1 | 14 | 54 | 9 | 2 | 80 |

```
KP481              1  12  39   8   0              60
KP781              0   0   4   7  29              40
Total_Purchases    2  26  97  24  31             180


'Product VS Fitness (Probability)'

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

[68]:
```python
pd.crosstab(df['Product'],df['Fitness']).plot(kind='bar',title='Product VS Fitness')
pd.crosstab(df['Fitness'],df['Product']).plot(kind='line')
plt.show()
```






#### 11.1.5 Product VS AgeGroup

[77]:
```
# Value Counts of Product VS AgeGroup
ct_p_ag = pd.crosstab(df['Product'],df['AgeGroup'], margins=True,
                    margins_name='Total_Purchases')

# Probability of Product VS AgeGroup
ctn_p_ag = pd.crosstab(df['Product'],df['AgeGroup'], margins=True,
                     margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS AgeGroup (Value Counts)"))
display(ct_p_ag)
print()
display((f"Product VS AgeGroup (Probability)"))
display(ctn_p_ag)
```

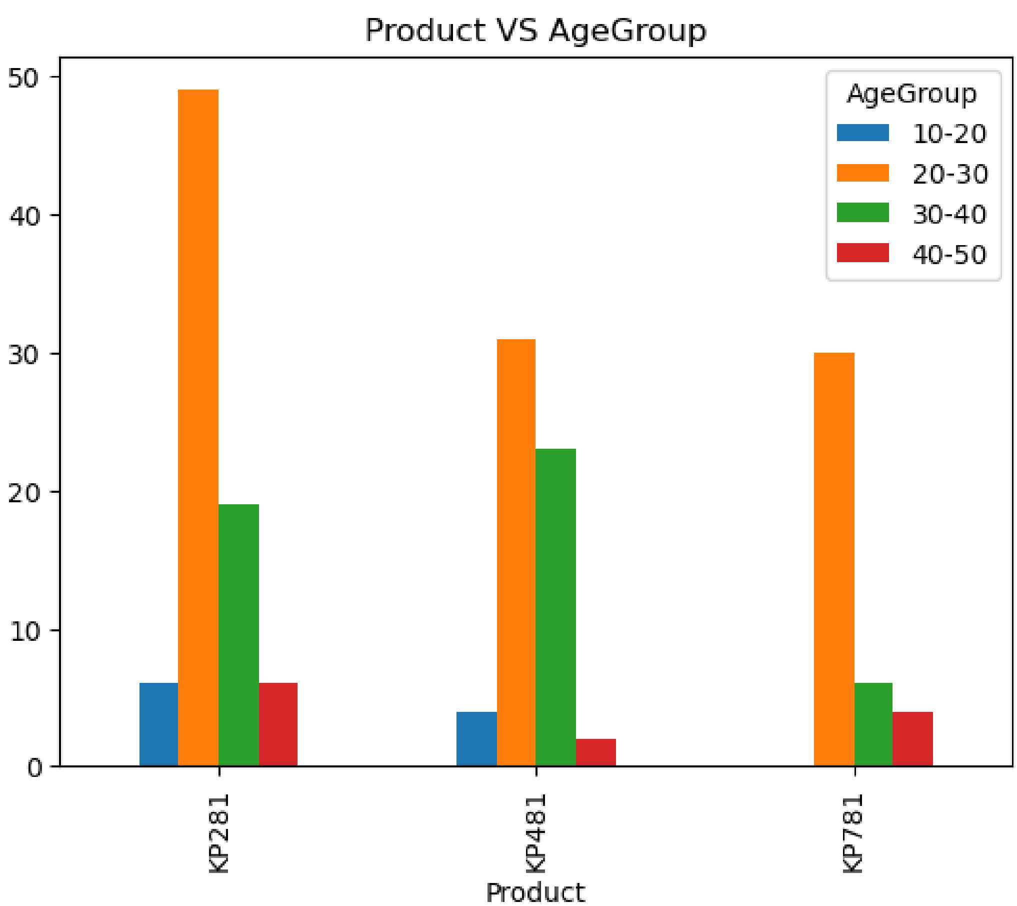
```
'Product VS AgeGroup (Value Counts)'

AgeGroup         10-20  20-30  30-40  40-50  Total_Purchases
Product
KP281                6     49     19      6               80
```

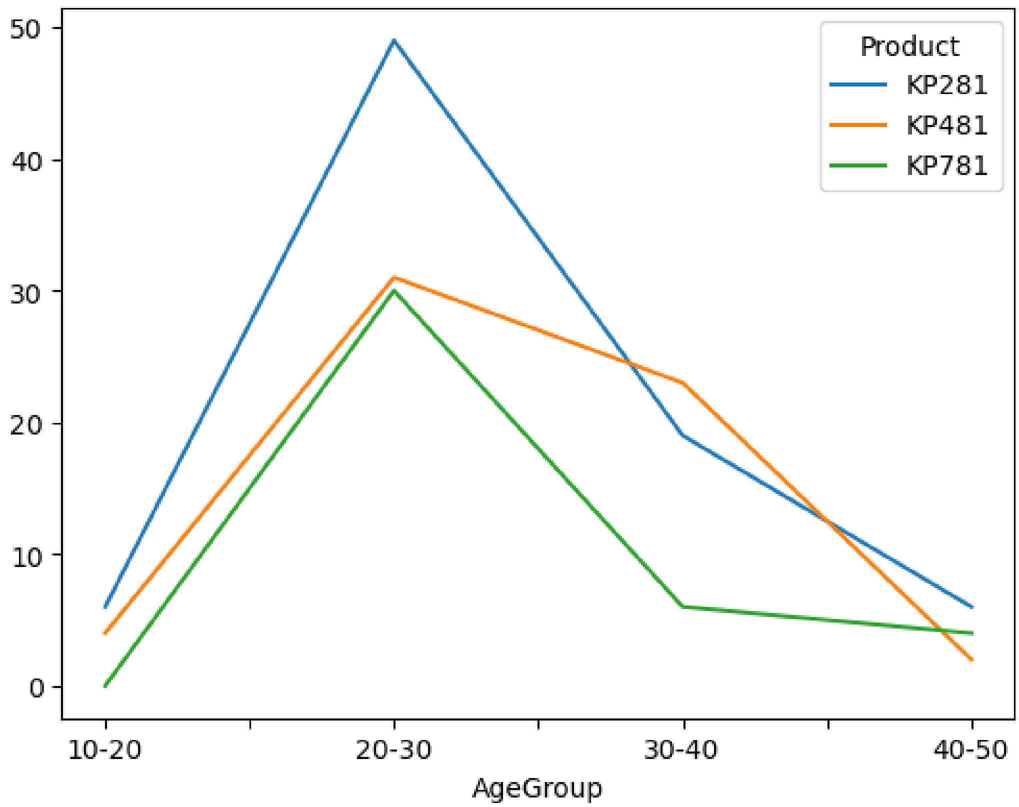
```
KP481                 4     31     23      2                60
KP781                 0     30      6      4                40
Total_Purchases      10    110     48     12               180
```

'Product VS AgeGroup (Probability)'

| AgeGroup | 10-20 | 20-30 | 30-40 | 40-50 | Total_Purchases |
|---|---|---|---|---|---|
| Product | | | | | |
| KP281 | 0.03 | 0.27 | 0.11 | 0.03 | 0.44 |
| KP481 | 0.02 | 0.17 | 0.13 | 0.01 | 0.33 |
| KP781 | 0.00 | 0.17 | 0.03 | 0.02 | 0.22 |
| Total_Purchases | 0.06 | 0.61 | 0.27 | 0.07 | 1.00 |

[70]:
```
pd.crosstab(df['Product'],df['AgeGroup']).plot(kind='bar'
                                   ,title='Product VS AgeGroup')
pd.crosstab(df['AgeGroup'],df['Product']).plot(kind='line')
plt.show()
```






#### 11.1.6 Product VS IncomeCategory

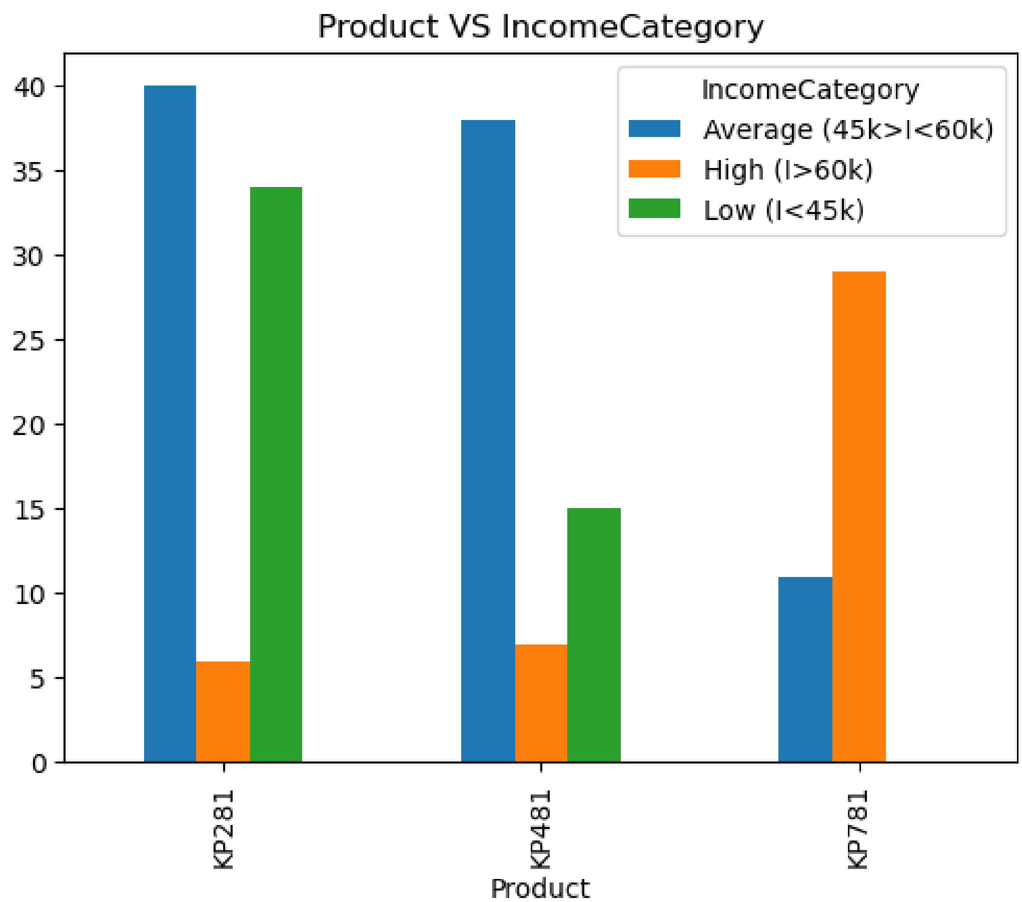
[75]:
```
ct_p_ic = pd.crosstab(df['Product'],df['IncomeCategory'], margins=True,
                    margins_name='Total_Purchases')

# Probability of Product VS IncomeCategory
ctn_p_ic = pd.crosstab(df['Product'],df['IncomeCategory'], margins=True,
                     margins_name='Total_Purchases', normalize=True).round(2)

# Display
display((f"Product VS IncomeCategory (Value Counts)"))
display(ct_p_ic)
print()
display((f"Product VS IncomeCategory (Probability)"))
display(ctn_p_ic)
```

```
'Product VS IncomeCategory (Value Counts)'

IncomeCategory   Average (45k>I<60k)  High (I>60k)  Low (I<45k)  \
Product
KP281                             40             6           34
KP481                             38             7           15
```

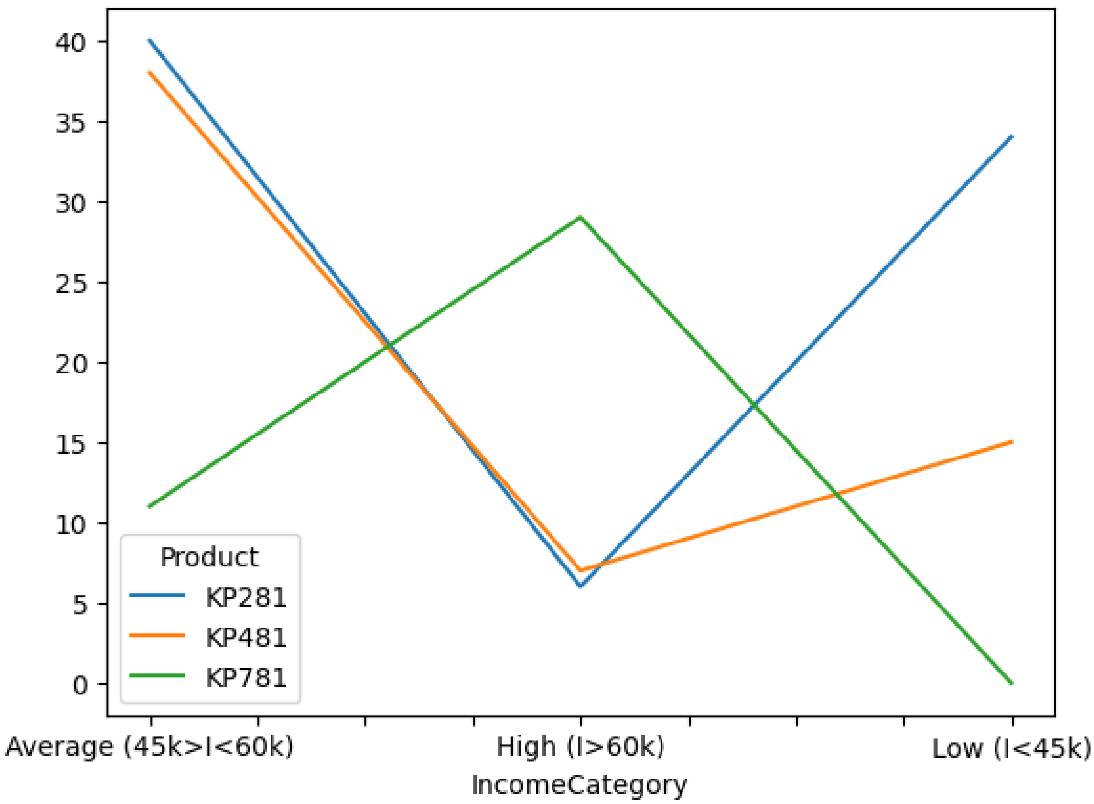
```
KP781                           11           29            0
Total_Purchases                 89           42           49

IncomeCategory   Total_Purchases
Product
KP281                         80
KP481                         60
KP781                         40
Total_Purchases              180


'Product VS IncomeCategory (Probability)'

IncomeCategory   Average (45k>I<60k)  High (I>60k)  Low (I<45k)  \
Product
KP281                           0.22          0.03         0.19
KP481                           0.21          0.04         0.08
KP781                           0.06          0.16         0.00
Total_Purchases                 0.49          0.23         0.27

IncomeCategory   Total_Purchases
Product
KP281                       0.44
KP481                       0.33
KP781                       0.22
Total_Purchases             1.00
```

[72]:

```
pd.crosstab(df['Product'],df['IncomeCategory']).plot(kind='bar',
                            title='Product VS IncomeCategory')
pd.crosstab(df['IncomeCategory'],df['Product']).plot(kind='line')
plt.show()
```



Product VS IncomeCategory
IncomeCategory
Average (45k>I<60k)
High (I>60k)
Low (I<45k)
0
5
10
15
20
25
30
35
40
KP281
KP481
KP781
Product




### 11.2 Marginal Probabilities

[76]:
```
# Calculate marginal probabilities for 'Treadmill Models'
mp_p = (df['Product'].value_counts() / len(df['Product'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'Fitness'
mp_f = (df['Fitness'].value_counts() / len(df['Fitness'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'Gender'
mp_g = (df['Gender'].value_counts() / len(df['Gender'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'MaritalStatus'
mp_ms = (df['MaritalStatus'].value_counts() / len(df['MaritalStatus'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'Usage'
mp_u = (df['Usage'].value_counts() / len(df['Usage'])
        ).reset_index(name='Probability').round(2)
```

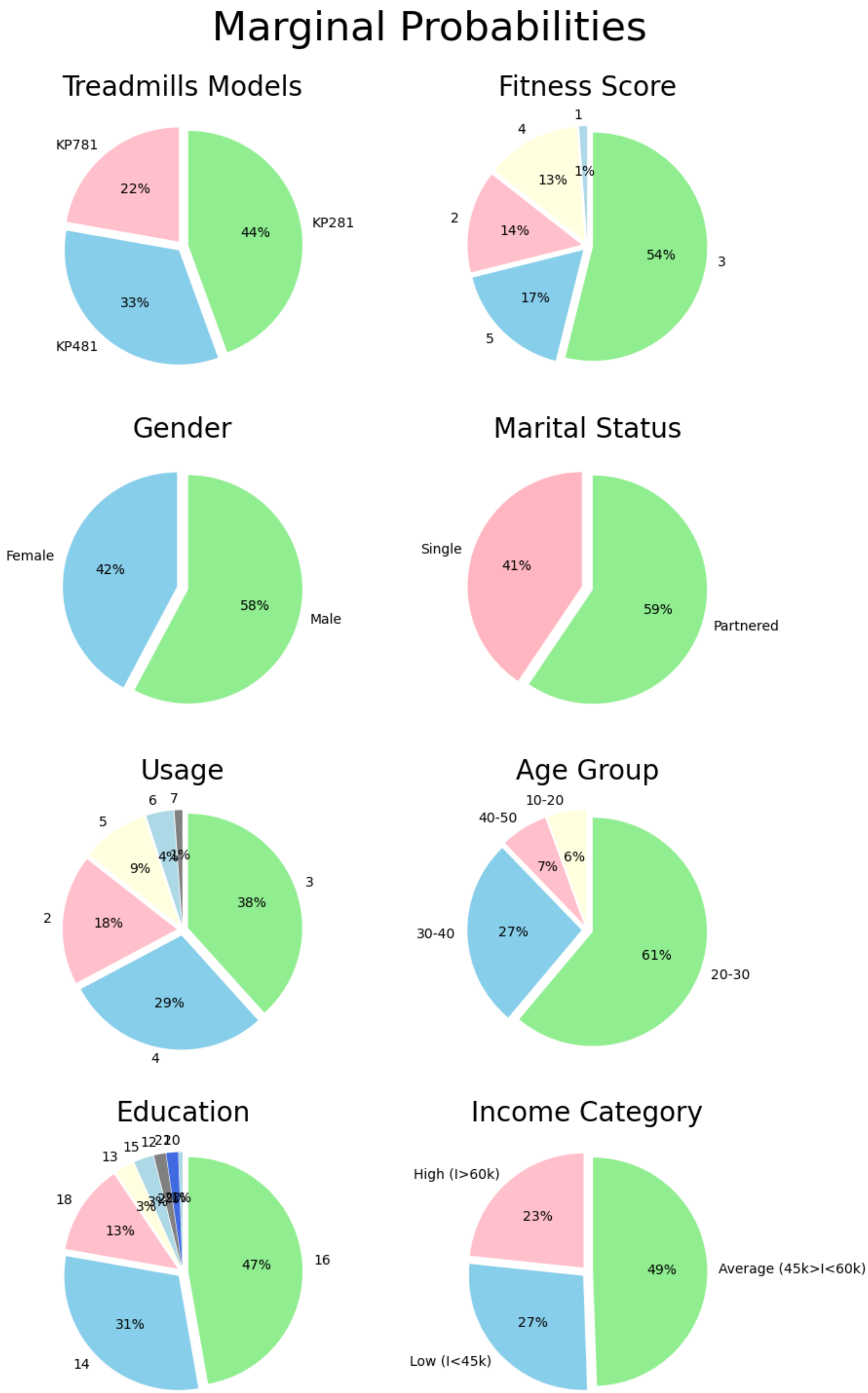
```python
# Calculate marginal probabilities for 'AgeGroup'
mp_ag = (df['AgeGroup'].value_counts() / len(df['AgeGroup'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'Usage'
mp_e = (df['Education'].value_counts() / len(df['Education'])
        ).reset_index(name='Probability').round(2)

# Calculate marginal probabilities for 'AgeGroup'
mp_ic = (df['IncomeCategory'].value_counts() / len(df['IncomeCategory'])
        ).reset_index(name='Probability').round(2)

# Display
display((f"Marginal Probability of Treadmill Models"))
display(mp_p)

display((f"Marginal Probability of Fitness"))
display(mp_f)

display((f"Marginal Probability of Gender"))
display(mp_g)

display((f"Marginal Probability of Marital Status"))
display(mp_ms)

display((f"Marginal Probability of Usage"))
display(mp_u)

display((f"Marginal Probability of Age Group"))
display(mp_ag)

display((f"Marginal Probability of Education"))
display(mp_e)

display((f"Marginal Probability of Income Category"))
display(mp_ic)
```

```
'Marginal Probability of Treadmill Models'

   index  Probability
0  KP281         0.44
1  KP481         0.33
2  KP781         0.22

'Marginal Probability of Fitness'

  index  Probability
0     3         0.54
```

```
1     5         0.17
2     2         0.14
3     4         0.13
4     1         0.01

'Marginal Probability of Gender'

    index  Probability
0    Male         0.58
1  Female         0.42

'Marginal Probability of Marital Status'

       index  Probability
0  Partnered         0.59
1     Single         0.41

'Marginal Probability of Usage'

  index  Probability
0     3         0.38
1     4         0.29
2     2         0.18
3     5         0.09
4     6         0.04
5     7         0.01

'Marginal Probability of Age Group'

   index  Probability
0  20-30         0.61
1  30-40         0.27
2  40-50         0.07
3  10-20         0.06

'Marginal Probability of Education'

  index  Probability
0    16         0.47
1    14         0.31
2    18         0.13
3    13         0.03
4    15         0.03
5    12         0.02
6    21         0.02
7    20         0.01

'Marginal Probability of Income Category'

                 index  Probability
0  Average (45k>I<60k)         0.49
1          Low (I<45k)         0.27
2         High (I>60k)         0.23
```

[74]:
```
# Distribution of Categorical Variables
plt.figure(figsize=(10,15))
plt.suptitle('Marginal Probabilities', fontsize = 30)

# Pie Chart Showing distribution of Treadmill Models
plt.subplot(4,2,1)
data_p = df['Product'].value_counts()
labels = data_p.index
plt.title('Treadmills Models', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05], autopct='%1.0f%%',
        startangle=90, counterclock=False, colors=['lightgreen', 'skyblue',␣
 ↪'pink'])

# Pie Chart Showing distribution of Fitness Score
plt.subplot(4,2,2)
data_p = df['Fitness'].value_counts()
labels = data_p.index
plt.title('Fitness Score', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05,0.05,0.05],␣
 ↪autopct='%1.0f%%',
        startangle=90, counterclock=False,
        colors=['lightgreen', 'skyblue', 'pink', 'lightyellow' ,'lightblue'])

# Pie Chart Showing distribution of Gender
plt.subplot(4,2,3)
data_p = df['Gender'].value_counts()
labels = data_p.index
plt.title('Gender', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05], autopct='%1.0f%%',
        startangle=90, counterclock=False, colors=['lightgreen', 'skyblue'])

# Pie Chart Showing distribution of Marital Status
plt.subplot(4,2,4)
data_p = df['MaritalStatus'].value_counts()
labels = data_p.index
plt.title('Marital Status', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05], autopct='%1.0f%%',
        startangle=90, counterclock=False, colors=['lightgreen', 'lightpink'])

# Pie Chart Showing distribution of Usage
plt.subplot(4,2,5)
data_p = df['Usage'].value_counts()
labels = data_p.index
plt.title('Usage', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05,0.05,0.05,0.05],
        autopct='%1.0f%%', startangle=90, counterclock=False,␣
 ↪colors=['lightgreen',
```

```
        'skyblue', 'pink', 'lightyellow' ,'lightblue', 'grey'])

# Pie Chart Showing distribution of Age Group
plt.subplot(4,2,6)
data_p = df['AgeGroup'].value_counts()
labels = data_p.index
plt.title('Age Group', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05,0.05], autopct='%1.
 ↪0f%%',
        startangle=90, counterclock=False, colors=['lightgreen', 'skyblue',␣
 ↪'pink',
                                                   'lightyellow' ,'lightblue'])

# Pie Chart Showing distribution of Education
plt.subplot(4,2,7)
data_p = df['Education'].value_counts()
labels = data_p.index
plt.title('Education', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.
 ↪05],
        autopct='%1.0f%%', startangle=90, counterclock=False,␣
 ↪colors=['lightgreen',
        'skyblue', 'pink', 'lightyellow' ,'lightblue', 'grey', 'royalblue',␣
 ↪'lightblue'])

# Pie Chart Showing distribution of Income Category
plt.subplot(4,2,8)
data_p = df['IncomeCategory'].value_counts()
labels = data_p.index
plt.title('Income Category', fontsize = 20)
plt.pie(data_p, labels=labels, explode = [0.05,0.05,0.05], autopct='%1.0f%%',
        startangle=90, counterclock=False, colors=['lightgreen', 'skyblue',␣
 ↪'pink'])

plt.tight_layout(pad= 2.0)
plt.show()
```


Marginal Probabilities
Treadmills Models
KP781
22%
KP281
44%
33%
KP481
Fitness Score
1
4
13%
1%
2
14%
54%
3
17%
5
Gender
Female
42%
58%
Male
Marital Status
Single
41%
59%
Partnered
Usage
6
7
5
3
9%
38%
2
18%
29%
4
Age Group
10-20
40-50
6%
7%
30-40
27%
61%
20-30
Education
13
15
18
3%
13%
47%
16
31%
14
Income Category
High (I>60k)
23%
49%
Average (45k>I<60k)
27%
Low (I<45k)

#### 11.2.1 KP281 customer's profile

```
1. Highest chances among other products.
2. Usage under 4days per week.
3. Fitness level mostly under 3.
4. Less to medium earning customers.
5. Females who Partnered most chance than Females who are single.
6. Customers who educated under 16 years most preferable.
7. Customers whose usage under 120 miles per week
```

### 11.3 KP481 customer's profile

```
1.Second Popular Product.
2. Usage under 4days per week.
3. Fitness level mostly under 3.
4. Less to medium earning customers.
5. Male customers who partnered prefer more than Male customers who single.
6. It has almost similar customer's profile like KP281, but KP281 is wide range of customers th
```

### 11.4 KP781 customer's profile

```
1. Mostly preferred by Male customers.
2. Usage more than 120 miles per week.
4. Fitness level more than 3.
5. Usage more than 4 days per week.
6. Customers who educated more than 16 years.
7. High salaried Customers.
```

## 12 6. Business Insights

1. 57.78% Customers are Male.
2. 59.44% Customers are Partnered.
3. Most sold product KP281, its 44.44% of sales out of overall Aerofit Treadmill sale.
4. KP281, KP481 products have almost similar customer's profile, except Male Partnered prefer KP481 & Female Partnered prefer KP281.
5. KP781 product is most preferred by Males, it's almost 6 times compared to Females.
6. 75% of customers are earning less than 60k, and customers who earning more than 60k prefer KP781.
7. KP781 had unique among other treadmills when it comes more usage or high fitness customer.
8. Probability of Buying KP281 increased from 44.44% to 58.7%, if customer is Female and Partnered.
9. Probability of Buying KP781 increased from 22.22% to 32.56%, if customer is Male and Single.
10. Probability of Buying KP781 decreased from 22.22% to 8.7%, if customer is Female and Partnered.

11. Approximately 88% of Aerofit customers belong to the low-income (29000-50000 USD) and medium-income (51000-75000 USD) groups.Remaining 11.67% belongs to High income group (above 75000 usd).
12. Due to its price of 2500 USD, the probability of customers belonging to the low-income and middle-income groups buying the KP781 treadmill is low compared to customers in the high-income group who can afford this higher-priced treadmill.
13. Customers belonging to the high-income group exclusively prefer KP781 due to its advanced features and higher cost compared to the other two treadmills.
14. Customers with 14-16 years of education prefer the KP281 and KP481 treadmills. However, among all treadmills, the majority of customers with 16-18 years of education prefer the KP781 treadmill.
15. Customers who run 60-100 miles per week prefer the KP281 treadmill, while mid runners who run 60-120 miles per week opt for the KP481. On the other hand, hardcore runners who run 120-200 miles per week prefer the KP781 treadmill due to its advanced features.
16. Customers who use treadmills 3 times a week prefer both KP281 and KP481. However, customers who use treadmills 4-5 times a week favor the KP781 treadmill.
17. Customers with fitness level 3 prefer both KP281 and KP481 treadmills, while customers with fitness level 5 predominantly use the most advanced KP781 treadmill.

# 13 7. Recommendations

## 13.1 Customer Profile For 'KP281'

```
1. Both Male and Female customers are equally likely to buy this model. Therefore company shoul
2. Company should target more customers with 16 years of education for 'KP281'.
3. Company should target more Partnered customers than Single customers for 'KP281'.
4. Company should target more customers with Usage of 3 days/week for 'KP281'.
5. Company should target more customers with Self rated Fitness Score of 3 out of 5 for 'KP281
6. Company should target more customers with Age ranges between 20 to 30 Years for 'KP281'.
7. Company should target more customers with Income ranges between $45k to $60k for 'KP281'.
```

## 13.2 Customer Profile For 'KP481'

```
1. Both Male and Female customers are almost equally likely to buy 'KP481'. so, company should
2. Company should target more customers with 16 years of education for 'KP481'.
3. Company should target more Partnered customers than Single customers for 'KP481'.
4. Company should target more customers with Usage of 3 days/week for 'KP481'.
5. Company should target more customers with Self rated Fitness Score of 3 out of 5 for 'KP481
6. Company should target more customers with Age ranges between 20 to 30 Years for 'KP481'.
7. Company should target more customers with Income ranges between $45k to $60k for 'KP481'.
```

## 13.3 Customer Profile For 'KP781'

```
1. Male customers are more likely to buy 'KP781' so company should target more Male customers
2. Company should target more customers with 18 years of education for 'KP781'.
3. Company should target more Partnered customers than Single customers for 'KP781'.
3. Company should target more customers with Usage of 4 days/week for 'KP781'.
4. Company should target more customers with Self rated Fitness Score of 5 out of 5 for 'KP781
```

```
5. Company should target more customers with Age ranges between 20 to 30 Years for 'KP781'.
6. Company should target more customers with Income greater than $60k for 'KP781'.
```

### 13.4 Actionable Insights:

The probability of female customers buying the KP781 treadmill is 4%, which is significantly lower compared to that of male customers :

1. Offer special incentives and discounts exclusively for female customers interested in purchasing the KP781 treadmill. This could include limited-time promotions, personalized offers, or package deals to make the treadmill more appealing and accessible to this customer segment. By providing targeted incentives, it can encourage more female customers to consider and invest in the KP781.

The probability of single customers purchasing each of the treadmills is lower compared to that of married customers:

1. Introduce exclusive offers and discounts for single customers as part of the collaboration with Virat Kohli. This can include special bundles, personalized packages, or limited-time promotions, providing added incentives for single customers to choose Aerofit treadmills.
2. Organize virtual fitness challenges or competitions, endorsed by Virat Kohli, to engage single customers and encourage them to participate in fitness activities with Aerofit treadmills. Prizes and recognition for participants can further boost motivation and engagement.

The probability of old customers purchasing each of the treadmills is lower compared to that of other age-group customers:

1. Offer personalized assistance to help customers aged 40-50 select the ideal treadmill model, providing them with the tools to maintain an active and healthy lifestyle. With Aerofit's expert guidance, customers can feel confident and motivated to make the most of their treadmills effective.

Due to its price of 2500 USD, the probability of customers belonging to the low-income and middle-income groups buying the KP781 treadmill is low compared to customers in the high-income group.

1. Introduce tailored discounts and incentives exclusively for customers belonging to the low and middle-income groups. These offers can include limited-time promotions, cashback rewards, or bundle deals, making the KP781 treadmill more affordable and enticing for this target audience.
2. Provide convenient EMI (Equated Monthly Installment) payment options for the KP781 treadmill. This will allow low and middle-income customers to spread the cost over several months, easing their financial burden and making the purchase more manageable.