In [56]:
```
#importing libraries

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

In [4]:
```
df = pd.read_csv('C:\\Users\\Keremane\\OneDrive\\Desktop\\Scaler Docs\\Data Sources\\Python Pandas\\aero
```

In [5]:
```
df.head()
```

Out[5]:

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

In [6]:
```
df.tail()
```

Out[6]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
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

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

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

In [8]:
```
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
```

Insights

- 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).

In [9]:
```
## changing the data type

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
```

In [10]:
```
# statisctical summary of object type columns

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

Out[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

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

df.describe()
```

Out[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

In [12]:
```
## Checking for duplicated values

df.duplicated().value_counts()
```

Out[12]:
```
False    180
dtype: int64
```

In [16]:
```
# checking the unique values for columns
for i in df.columns:
    print('Unique Values in',i,'column are :-')
    print(df[i].unique())
    print("-"*50)
```

```
Unique Values in Product column are :-
['KP281' 'KP481' 'KP781']
--------------------------------------------------
Unique Values in Age column are :-
[18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
 43 44 46 47 50 45 48 42]
--------------------------------------------------
Unique Values in Gender column are :-
['Male' 'Female']
--------------------------------------------------
Unique Values in Education column are :-
[14 15 12 13 16 18 20 21]
--------------------------------------------------
Unique Values in MaritalStatus column are :-
['Single' 'Partnered']
--------------------------------------------------
Unique Values in Usage column are :-
['3' '2' '4' '5' '6' '7']
--------------------------------------------------
Unique Values in Fitness column are :-
['4' '3' '2' '1' '5']
--------------------------------------------------
Unique Values in Income column are :-
[ 29562  31836  30699  32973  35247  37521  36384  38658  40932  34110
  39795  42069  44343  45480  46617  48891  53439  43206  52302  51165
  50028  54576  68220  55713  60261  67083  56850  59124  61398  57987
  64809  47754  65220  62535  48658  54781  48556  58516  53536  61006
  57271  52291  49801  62251  64741  70966  75946  74701  69721  83416
  88396  90886  92131  77191  52290  85906 103336  99601  89641  95866
 104581  95508]
--------------------------------------------------
Unique Values in Miles column are :-
[112  75  66  85  47 141 103  94 113  38 188  56 132 169  64  53 106  95
 212  42 127  74 170  21 120 200 140 100  80 160 180 240 150 300 280 260
 360]
--------------------------------------------------
```

Adding extra Columns with bins

In [17]:
```
#binning the age values into categories
bin_range1 = [17,25,35,45,float('inf')]
bin_labels1 = ['Young Adults', 'Adults', 'Middle Aged Adults', 'Elder']

df['age_group'] = pd.cut(df['Age'],bins = bin_range1,labels = bin_labels1)

#binning the education values into categories
bin_range2 = [0,12,15,float('inf')]
bin_labels2 = ['Primary Education', 'Secondary Education', 'Higher Education']

df['edu_group'] = pd.cut(df['Education'],bins = bin_range2,labels = bin_labels2)

#binning the income values into categories
bin_range3 = [0,40000,60000,80000,float('inf')]
bin_labels3 = ['Low Income','Moderate Income','High Income','Very High Income']

df['income_group'] = pd.cut(df['Income'],bins = bin_range3,labels = bin_labels3)

#binning the miles values into categories
bin_range4 = [0,50,100,200,float('inf')]
bin_labels4 = ['Light Activity', 'Moderate Activity', 'Active Lifestyle', 'Fitness Enthusiast ']

df['miles_group'] = pd.cut(df['Miles'],bins = bin_range4,labels = bin_labels4)
```

In [18]:
```
df.head()
```

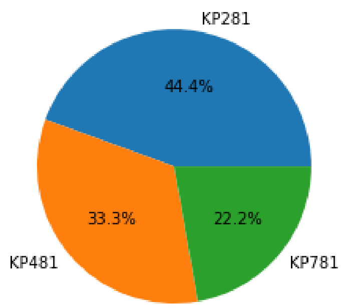
Out[18]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	age_group	edu_group	income_group	mile
0	KP281	18	Male	14	Single	3	4	29562	112	Young Adults	Secondary Education	Low Income	
1	KP281	19	Male	15	Single	2	3	31836	75	Young Adults	Secondary Education	Low Income	M
2	KP281	19	Female	14	Partnered	4	3	30699	66	Young Adults	Secondary Education	Low Income	M
3	KP281	19	Male	12	Single	3	3	32973	85	Young Adults	Primary Education	Low Income	M
4	KP281	20	Male	13	Partnered	4	2	35247	47	Young Adults	Secondary Education	Low Income	Ligh

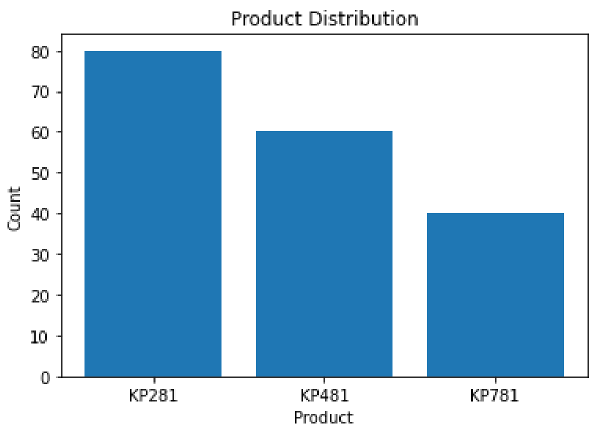
Univariate Analysis

Categorical Variables

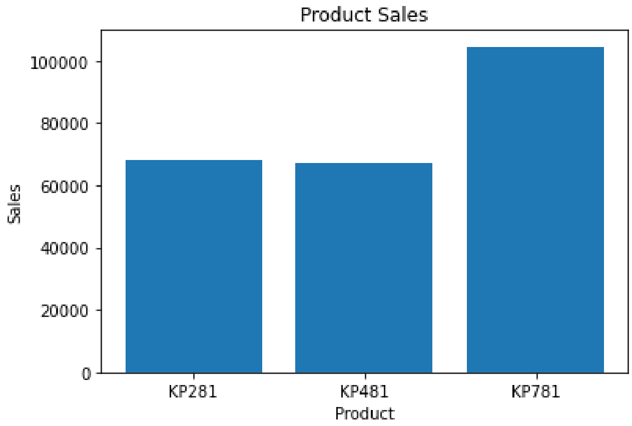
In [25]:
```
## distribution of Products

dfp=df['Product'].value_counts()
plt.pie(dfp, labels=dfp.index, autopct='%1.1f%%')
plt.show()
```



In [29]:
```
plt.bar(dfp.index, dfp)
##plt.xticks(rotation=45)
plt.xlabel('Product')
plt.ylabel('Count')
plt.title('Product Distribution')
plt.show()
```



In [30]:
```
plt.bar(df['Product'], df['Income'], data=df)
plt.xlabel('Product')
plt.ylabel('Income')
plt.title('Product v Income')
plt.show()
```

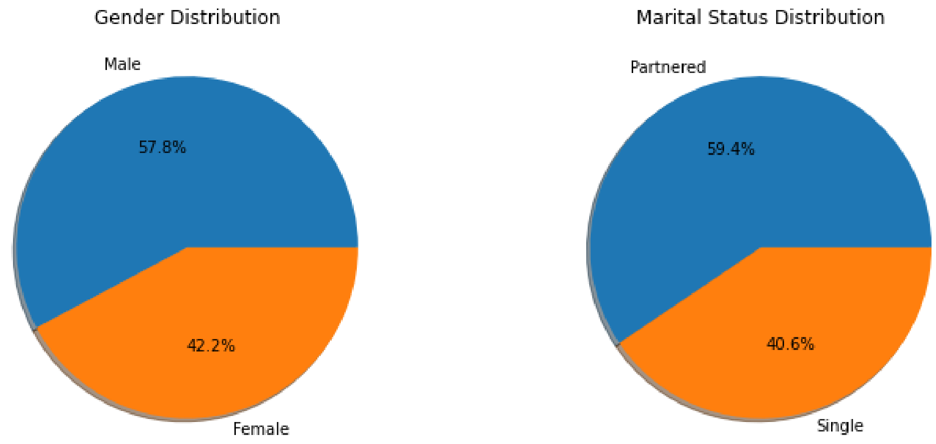


Insights

- The KP281 treadmill model, positioned as an entry-level product, has the highest number of units sold, trailed by the KP481 (mid-level) and KP781 (advanced) models.
- KP 781 having more income when compared to other 2 products. Other 2 shares almost same income

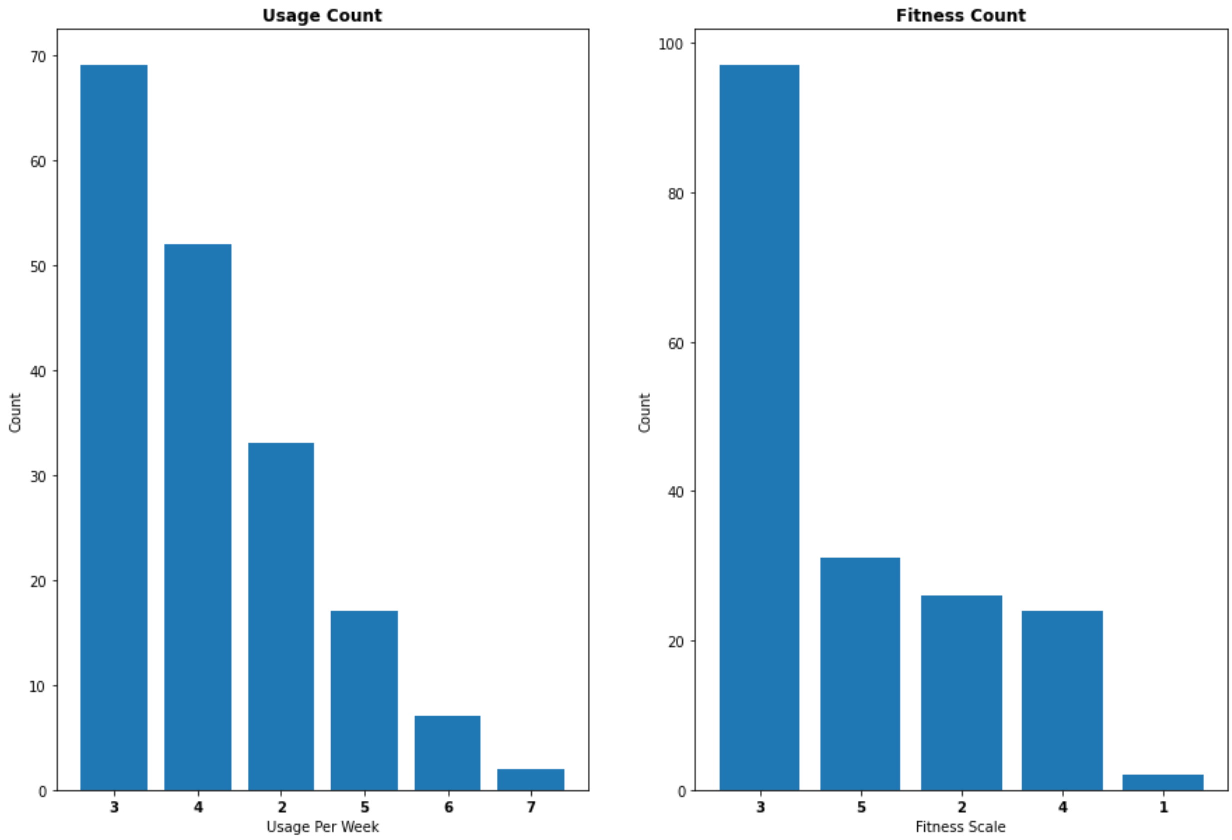
Gender and Maritial Distribution

In [33]:
```
fig = plt.figure(figsize = (12,5))
gs = fig.add_gridspec(1,2)

# creating pie chart for gender disribution
ax0 = fig.add_subplot(gs[0,0])


ax0.pie(df['Gender'].value_counts().values,labels = df['Gender'].value_counts().index,autopct = '%.1f%%
        shadow = True)

#setting title for visual
ax0.set_title('Gender Distribution')

# creating pie chart for marital status
ax1 = fig.add_subplot(gs[0,1])

color_map = ["#3A7089", "#4b4b4c"]
ax1.pie(df['MaritalStatus'].value_counts().values,labels = df['MaritalStatus'].value_counts().index,aut
        shadow = True)

#setting title for visual
ax1.set_title('Marital Status Distribution')

plt.show()
```



Buyer Fitness and Treadmill Usage

In [35]:
```
#setting the plot style
fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(1,2)

# creating bar chart for usage disribution

ax0 = fig.add_subplot(gs[0,0])
temp = df['Usage'].value_counts()
ax0.bar(x=temp.index,height = temp.values)


#adding axis label
ax0.set_ylabel('Count')
ax0.set_xlabel('Usage Per Week')
ax0.set_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax0.set_title('Usage Count',{'weight':'bold'})

#creating a info table for usage



# creating bar chart for fitness scale

ax2 = fig.add_subplot(gs[0,1])
temp = df['Fitness'].value_counts()

ax2.bar(x=temp.index,height = temp.values)


#adding axis label
ax2.set_ylabel('Count')
ax2.set_xlabel('Fitness Scale')
ax2.set_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax2.set_title('Fitness Count',{'weight':'bold'})




plt.show()
```



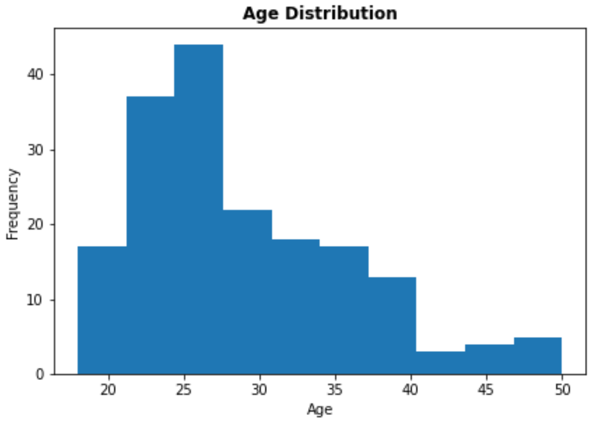
Insights

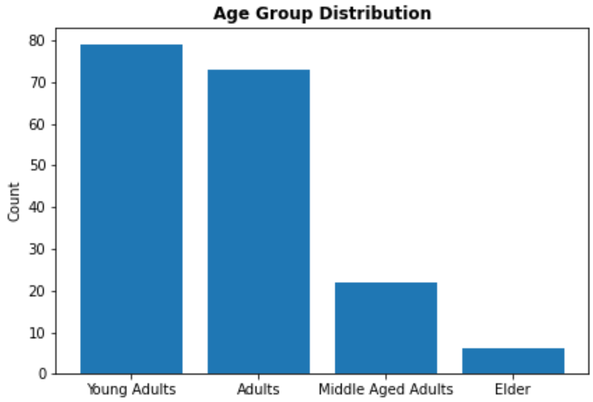
- Most of the users are using the treadmill for 3/4 times a week only.
- Most of the customers have self-evaluated their fitness at a level 3 on a scale of 1 to 5. Furthermore, a substantial amount of the total customers have rated themselves at 3 or higher, indicating commendable fitness levels.

Numerical Variables

Customer Age Distribution

In [37]:
```
#setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2)

      #creating age histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Age'])
ax0.set_xlabel('Age')
ax0.set_ylabel('Frequency')



#setting title for visual
ax0.set_title('Age Distribution',{'weight':'bold'})


   #creating age group bar chart

ax2 = fig.add_subplot(gs[0,1])
temp = df['age_group'].value_counts()
ax2.bar(x=temp.index,height = temp.values)



#adding axis label
ax2.set_ylabel('Count')
ax2.set_xticklabels(temp.index)

#setting title for visual
ax2.set_title('Age Group Distribution',{'weight':'bold'})

    #creating a table for group info

ax3 = fig.add_subplot(gs[1,0])
age_info = [['Young Adults','44%','18 to 25'],['Adults','41%','26 to 35'],['Middle Aged','12%','36 to 4
            ['Elder','3%','Above 45']]

table = ax3.table(cellText = age_info, cellLoc='center',colLabels =['Age','Probability','Group'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax3.axis('off')


plt.show()
```





Age	Probability	Group
Young Adults	44%	18 to 25
Adults	41%	26 to 35
Middle Aged	12%	36 to 45
Elder	3%	Above 45

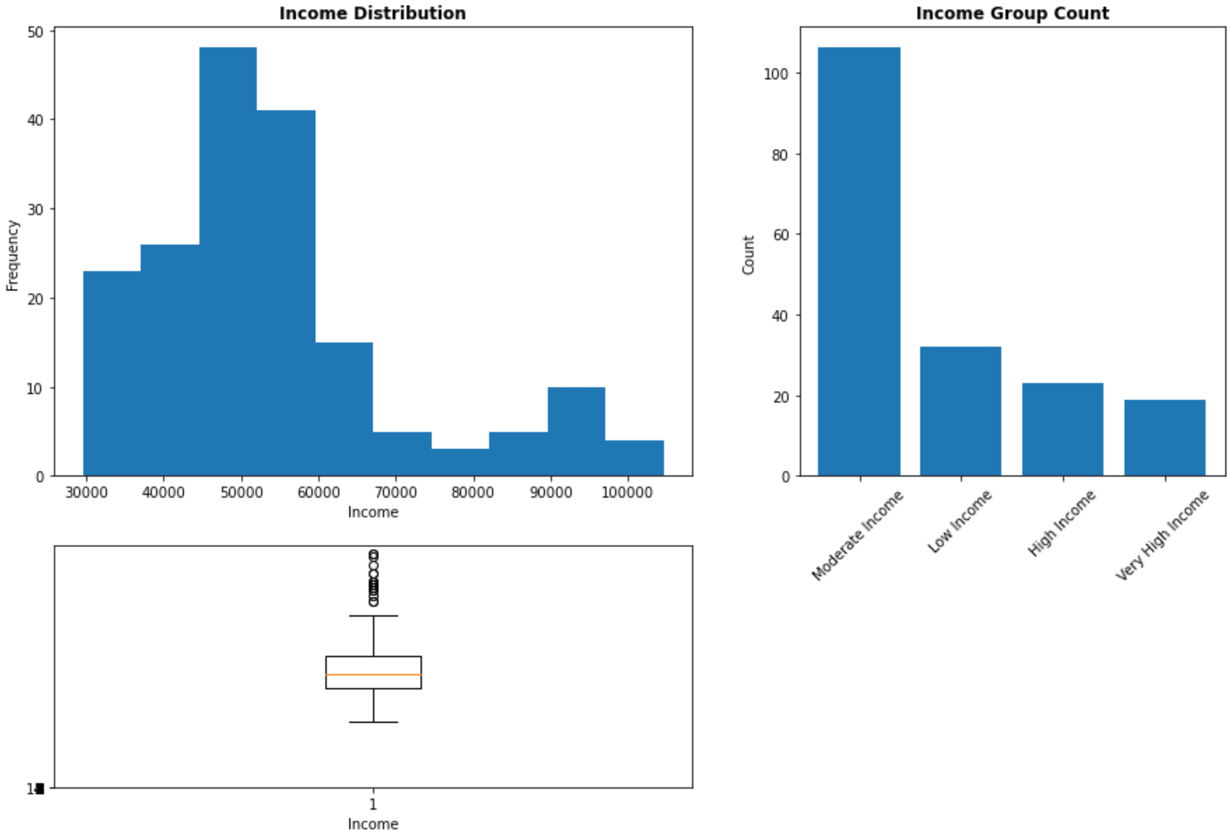
Insights

- 85% of the customers fall in the age range of 18 to 35. with a median age of 26, suggesting young people showing more interest in the companies products
- Outliers: As we can see from the box plot, there are 3 outlier's present in the age data.

Customer Income Distribution

In [43]:
```
#setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

                                    #creating Income histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Income'])
ax0.set_xlabel('Income')
ax0.set_ylabel('Frequency')



#setting title for visual
ax0.set_title('Income Distribution',{'weight':'bold'})


                                     #creating box plot for Income

ax1 = fig.add_subplot(gs[1,0])
boxplot = ax1.boxplot(x = df['Income'])



#removing y-axis ticks
ax1.set_yticks(df['Income'].value_counts())

#adding axis label
ax1.set_xlabel('Income')

                                    #creating Income group bar chart

ax2 = fig.add_subplot(gs[0,1])
temp = df['income_group'].value_counts()
ax2.bar(x=temp.index,height = temp.values)

#adding axis label
ax2.set_ylabel('Count')
ax2.set_xticklabels(temp.index)
plt.xticks(rotation=45)

#setting title for visual
ax2.set_title('Income Group Count',{'weight':'bold'})
```

Out[43]:
```
Text(0.5, 1.0, 'Income Group Count')
```



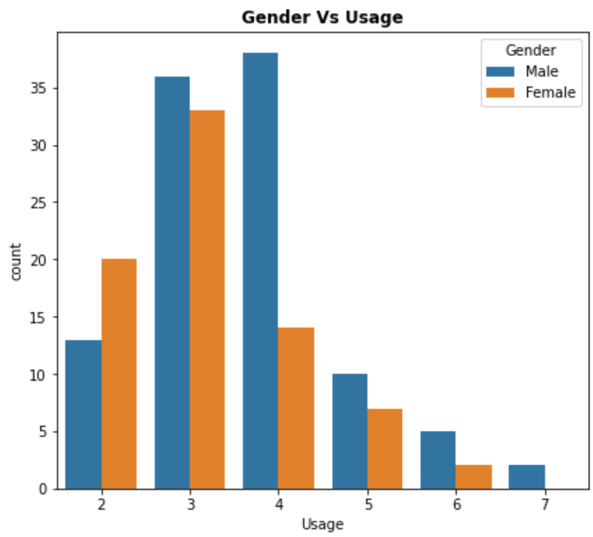
Insights

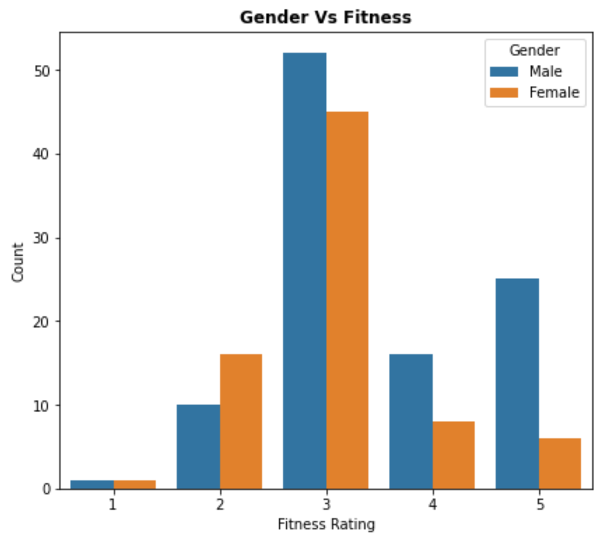
- Most of the customers fall in the income group of (40k to 60k) dollars suggesting higher inclination of this income group people towards the products.
- Suggesting Most of the total customers fall in income group of below 60k
- Outliers: As we can see from the box plot, there are many outlier's present in the income data

Bi-variate Analysis

Gender vs Product Usage And Gender Vs Fitness

In [45]:
```
#setting the plot style
fig = plt.figure(figsize = (15,6))
gs = fig.add_gridspec(1,2)

                                        # Usage Vs Gender

#creating bar plot
ax1 = fig.add_subplot(gs[0,0])

plot = sns.countplot(data = df, x = 'Usage', hue = 'Gender',order = sorted(df['Usage'].unique()),
              ax = ax1)


#setting title for visual
ax1.set_title('Gender Vs Usage',{'weight':'bold'})


                                      # Fitness Vs Gender

#creating bar plot
ax2 = fig.add_subplot(gs[0,1])

plot = sns.countplot(data = df, x = 'Fitness', hue = 'Gender',order = sorted(df['Fitness'].unique()),
              ax = ax2)


#customizing axis labels
ax2.set_xlabel('Fitness Rating')
ax2.set_ylabel('Count')

#setting title for visual
ax2.set_title('Gender Vs Fitness',{'weight':'bold'})

plt.show()
```





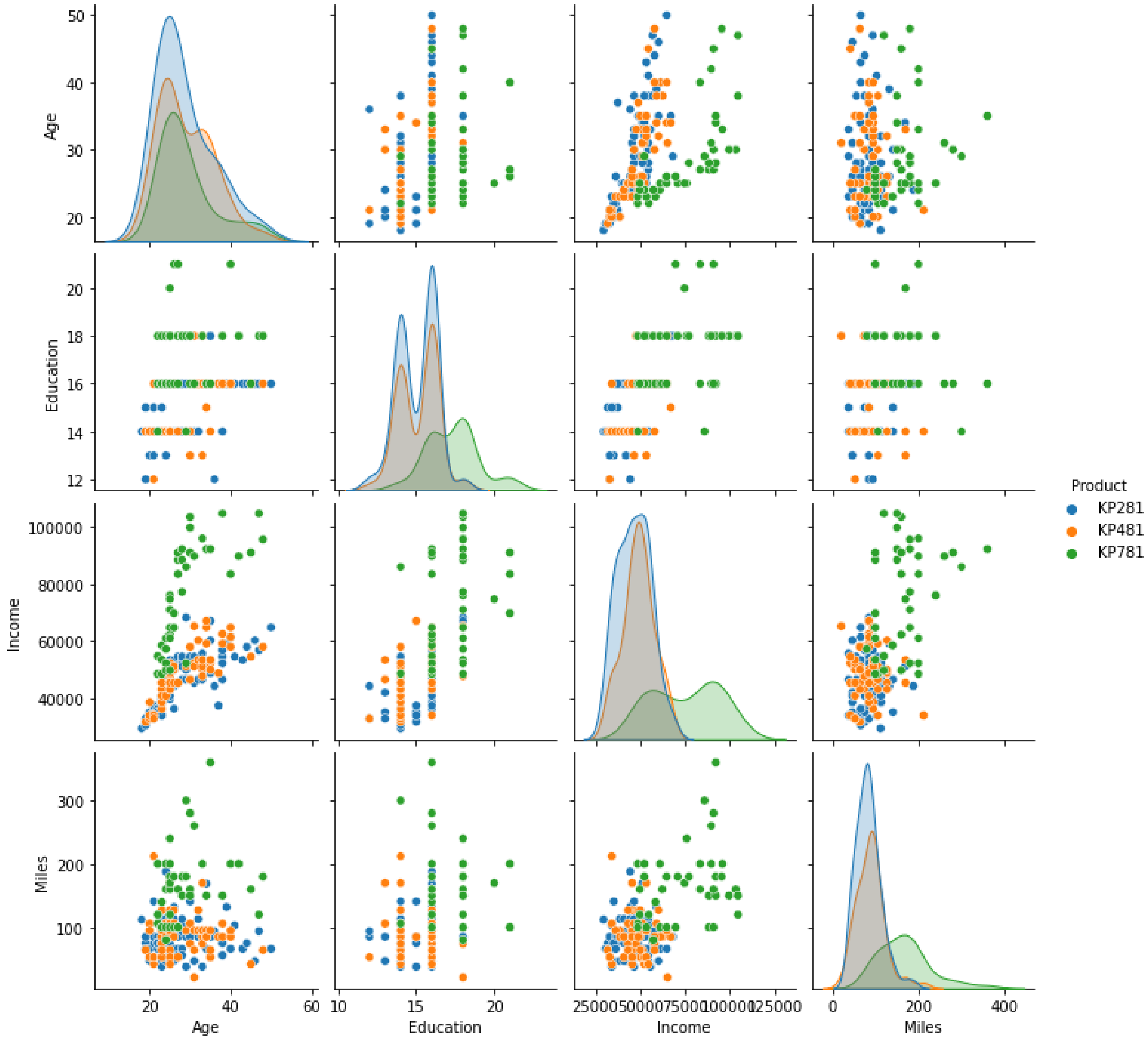
Insights

1. Gender Vs Usage
    - Most of Female customers plan to use the treadmill for 2 to 3 times a week whereas almost Most of Male customer plan to use the treadmill for 3 to 4 times a week
2. Gender Vs Fitness
    - Most of Female customers rated themselves between 2 to 3 whereas almost Most of Male customer rated themselves between 3 to 5 on the fitness scale

Correlation between Variables

Pairplot

In [47]:
```
df_copy = copy.deepcopy(df)
sns.pairplot(df_copy, hue ='Product')
plt.show()
```



Heatmap

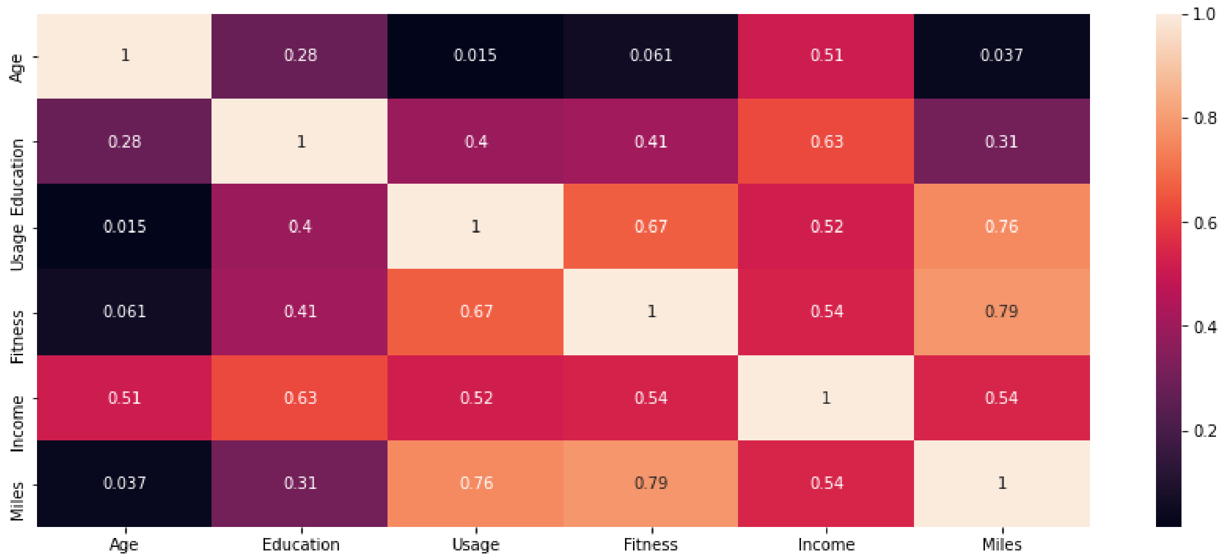
In [48]:

```
# First we need to convert object into int datatype for usage and fitness columns

df_copy['Usage'] = df_copy['Usage'].astype('int')
df_copy['Fitness'] = df_copy['Fitness'].astype('int')

df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 13 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    int32
 6   Fitness        180 non-null    int32
 7   Income         180 non-null    int64
 8   Miles          180 non-null    int64
 9   age_group      180 non-null    category
 10  edu_group      180 non-null    category
 11  income_group   180 non-null    category
 12  miles_group    180 non-null    category
dtypes: category(4), int32(2), int64(4), object(3)
memory usage: 12.8+ KB
```

In [49]:

```
corr_mat = df_copy.corr()

plt.figure(figsize=(15,6))

sns.heatmap(corr_mat,annot = True)

plt.show()
```



Insights

- 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.

Computing Probability - Marginal, Conditional Probability

Probability of product purchase w.r.t. gender

In [50]:
```
pd.crosstab(index =df['Product'],columns = df['Gender'],margins = True,normalize = True ).round(2)
```

Out[50]:

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

Insights

- The Probability of a treadmill being purchased by a female is 42%.
- The conditional probability of purchasing the treadmill model given that the customer is female is
  - For Treadmill model KP281 - 22%
  - For Treadmill model KP481 - 16%
  - For Treadmill model KP781 - 4%
- The Probability of a treadmill being purchased by a male is 58%.
- The conditional probability of purchasing the treadmill model given that the customer is male is -
  - For Treadmill model KP281 - 22%
  - For Treadmill model KP481 - 17%
  - For Treadmill model KP781 - 18%

Probability of product purchase w.r.t. Age

In [51]:
```
pd.crosstab(index =df['Product'],columns = df['age_group'],margins = True,normalize = True ).round(2)
```

Out[51]:

age_group	Young Adults	Adults	Middle Aged Adults	Elder	All
Product					
KP281	0.19	0.18	0.06	0.02	0.44
KP481	0.16	0.13	0.04	0.01	0.33
KP781	0.09	0.09	0.02	0.01	0.22
All	0.44	0.41	0.12	0.03	1.00

Insights

- The Probability of a treadmill being purchased by a Young Adult(18-25) is 44%.
- The conditional probability of purchasing the treadmill model given that the customer is Young Adult is
  - For Treadmill model KP281 - 19%
  - For Treadmill model KP481 - 16%
  - For Treadmill model KP781 - 9%
- The Probability of a treadmill being purchased by a Adult(26-35) is 41%.
- The conditional probability of purchasing the treadmill model given that the customer is Adult is -
  - For Treadmill model KP281 - 18%
  - For Treadmill model KP481 - 13%
  - For Treadmill model KP781 - 9%
- The Probability of a treadmill being purchased by a Middle Aged(36-45) is 12%.
- The Probability of a treadmill being purchased by a Elder(Above 45) is only 3%

Probability of product purchase w.r.t. Education level and Income and Fitness

In [53]:
```
pd.crosstab(index =df['Product'],columns = df['edu_group'],margins = True,normalize = True ).round(2)
```

Out[53]:

edu_group Product	Primary Education	Secondary Education	Higher Education	All
KP281	0.01	0.21	0.23	0.44
KP481	0.01	0.14	0.18	0.33
KP781	0.00	0.01	0.21	0.22
All	0.02	0.36	0.62	1.00

In [54]:
```
pd.crosstab(index =df['Product'],columns = df['income_group'],margins = True,normalize = True ).round(2
```

Out[54]:

income_group Product	Low Income	Moderate Income	High Income	Very High Income	All
KP281	0.13	0.28	0.03	0.00	0.44
KP481	0.05	0.24	0.04	0.00	0.33
KP781	0.00	0.06	0.06	0.11	0.22
All	0.18	0.59	0.13	0.11	1.00

In [55]:
```
pd.crosstab(index =df['Product'],columns = df['Fitness'],margins = True,normalize = True ).round(2)
```

Out[55]:

Fitness Product	1	2	3	4	5	All
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
All	0.01	0.14	0.54	0.13	0.17	1.00

Recommendations

Marketing Campaigns for KP781

The KP784 model exhibits a significant sales disparity in terms of gender, with only 18% of total sales attributed to female customers. To enhance this metric, it is recommended to implement targeted strategies such as offering special promotions and trials exclusively designed for the female customers.

Affordable Pricing and Payment Plans

Given the target customer's age, education level, and income, it's important to offer the KP281 and KP481 Treadmill at an affordable price point. Additionally, consider providing flexible payment plans that allow customers to spread the cost over several months. This can make the treadmill more accessible to customers with varying budgets.

User-Friendly App Integration

Create a user-friendly app that syncs with the treadmill. This app could track users' weekly running mileage, provide real-time feedback on their progress, and offer personalized recommendations for workouts based on their fitness scale and goals.This can enhance the overall treadmill experience and keep users engaged.

In [ ]: