Aerofit_Analysis

July 15, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df=pd.read_csv('/content/aerofit_treadmill.csv')
```

[3]:
```
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
```

[ ]:
```
#Import the dataset and do usual data analysis steps like checking the␣
↪structure & characteristics of the dataset
```

[4]:
```
#number of unique values in our data
for i in df.columns:
  print(i,':',df[i].nunique())
```

```
Product : 3
Age : 32
Gender : 2
Education : 8
MaritalStatus : 2
```

```
Usage : 6
Fitness : 5
Income : 62
Miles : 37
```

[5]:
```
# Check number of NULL values for each column - No Nulls detected
df.isnull().sum()
```

[5]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

[6]:
```
df['Income'].describe()
```

[6]:
```
count       180.000000
mean      53719.577778
std       16506.684226
min       29562.000000
25%       44058.750000
50%       50596.500000
75%       58668.000000
max      104581.000000
Name: Income, dtype: float64
```

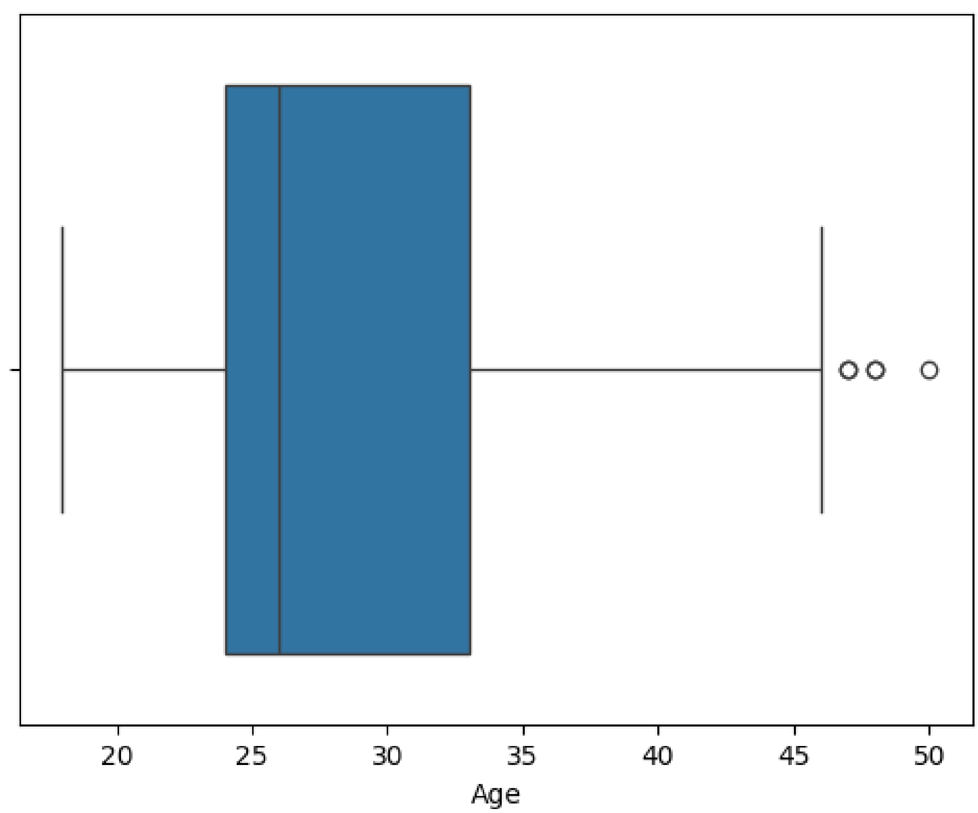
[7]:
```
#Detect Outliers (using boxplot, "describe" method by checking the difference
↪between mean and median)
for i in df.columns:
  print('Column :',i,"\n",df[i].describe())
```

```
Column : Product
 count       180
unique        3
top       KP281
freq         80
Name: Product, dtype: object
Column : Age
 count    180.000000
mean      28.788889
std        6.943498
min       18.000000
25%       24.000000
```

```
50%       26.000000
75%       33.000000
max       50.000000
Name: Age, dtype: float64
Column : Gender
 count      180
unique       2
top       Male
freq       104
Name: Gender, dtype: object
Column : Education
 count    180.000000
mean      15.572222
std        1.617055
min       12.000000
25%       14.000000
50%       16.000000
75%       16.000000
max       21.000000
Name: Education, dtype: float64
Column : MaritalStatus
 count           180
unique            2
top       Partnered
freq            107
Name: MaritalStatus, dtype: object
Column : Usage
 count    180.000000
mean       3.455556
std        1.084797
min        2.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        7.000000
Name: Usage, dtype: float64
Column : Fitness
 count    180.000000
mean       3.311111
std        0.958869
min        1.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        5.000000
Name: Fitness, dtype: float64
Column : Income
 count       180.000000
```

```
mean      53719.577778
std       16506.684226
min       29562.000000
25%       44058.750000
50%       50596.500000
75%       58668.000000
max      104581.000000
Name: Income, dtype: float64
Column : Miles
 count    180.000000
mean     103.194444
std       51.863605
min       21.000000
25%       66.000000
50%       94.000000
75%      114.750000
max      360.000000
Name: Miles, dtype: float64
```

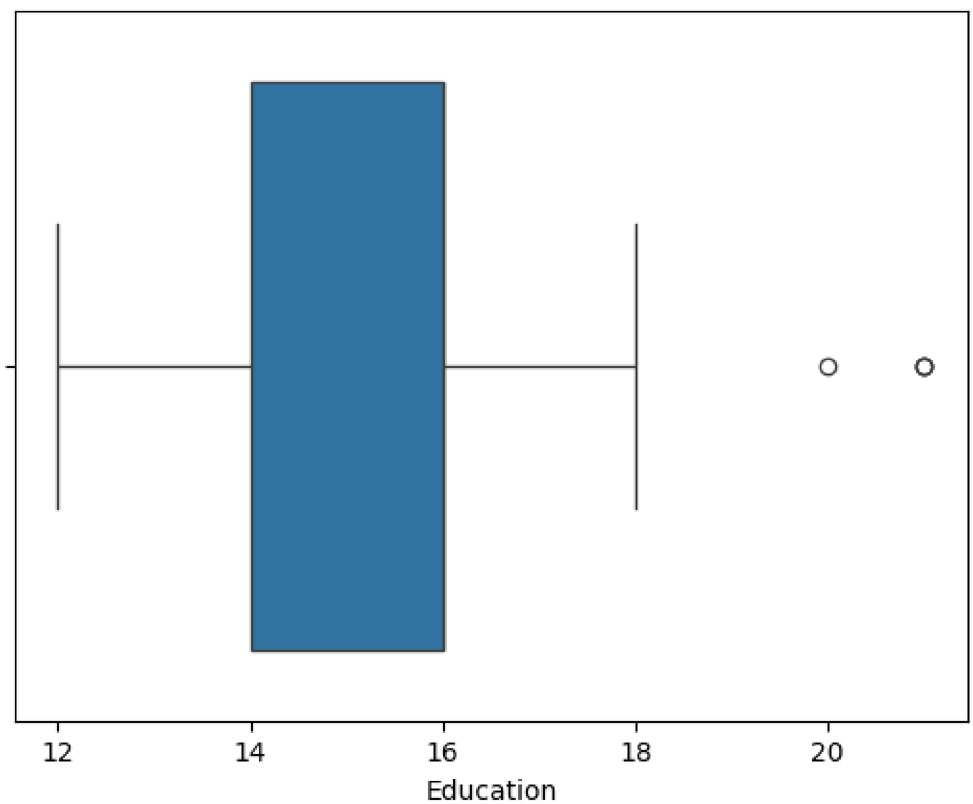
[8]:
```
# We will boxplot Product, Age, Education, Usage, Fitness, Income, Miles␣
↪numerical columns
sns.boxplot(data=df, x="Age")
plt.show()
```



[9]:
```python
# Mean for Age column is 28.788889 using describe method.
#IQR = 75%-25% = 33 - 24 = 9 - Median
#Outliers for Age > Q3(75%) + 1.5XIQR() = 33 + 1.5X9 = 46.5
Age_IQR=9
Age_75=np.percentile(df['Age'],75)
Age_outlier=Age_75+(1.5*Age_IQR)
df[df['Age']>Age_outlier]['Age'].value_counts()
# Outlier ages are 47, 48 and 50 with 2, 2 and 1 counts respectively
```

[9]:
```
Age
47    2
48    2
50    1
Name: count, dtype: int64
```

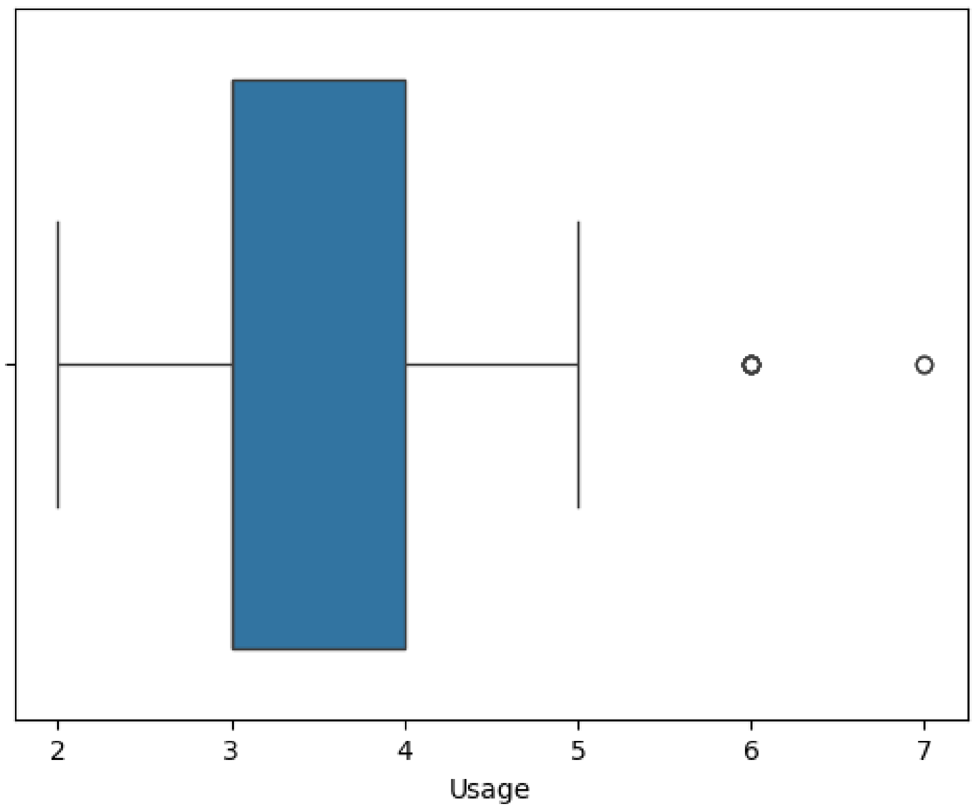
[10]:
```python
sns.boxplot(data=df, x="Education")
plt.show()
```



[11]:
```python
# Mean for Education column is 15.572222 using describe method.
#IQR = 75%-25% = 16 - 14 = 2 - Median
#Outliers for Education > Q3(75%) + 1.5XIQR() = 16 + 1.5X2 = 19
Education_IQR=2
Education_75=np.percentile(df['Education'],75)
Education_outlier=Education_75+(1.5*Education_IQR)
df[df['Education']>Education_outlier]['Education'].value_counts()
# Outlier education years are 21 and 20 with 3 and 1 counts respectively
```

[11]:
```
Education
21    3
20    1
Name: count, dtype: int64
```

[12]:
```python
sns.boxplot(data=df, x="Usage")
plt.show()
```



[13]:
```python
# Mean for Usage column is 3.455556 using describe method.
#IQR = 75%-25% = 4 - 3 = 1 - Median
#Outliers for Usage > Q3(75%) + 1.5XIQR() = 4 + 1.5X1 = 5.5
Usage_IQR=1
```

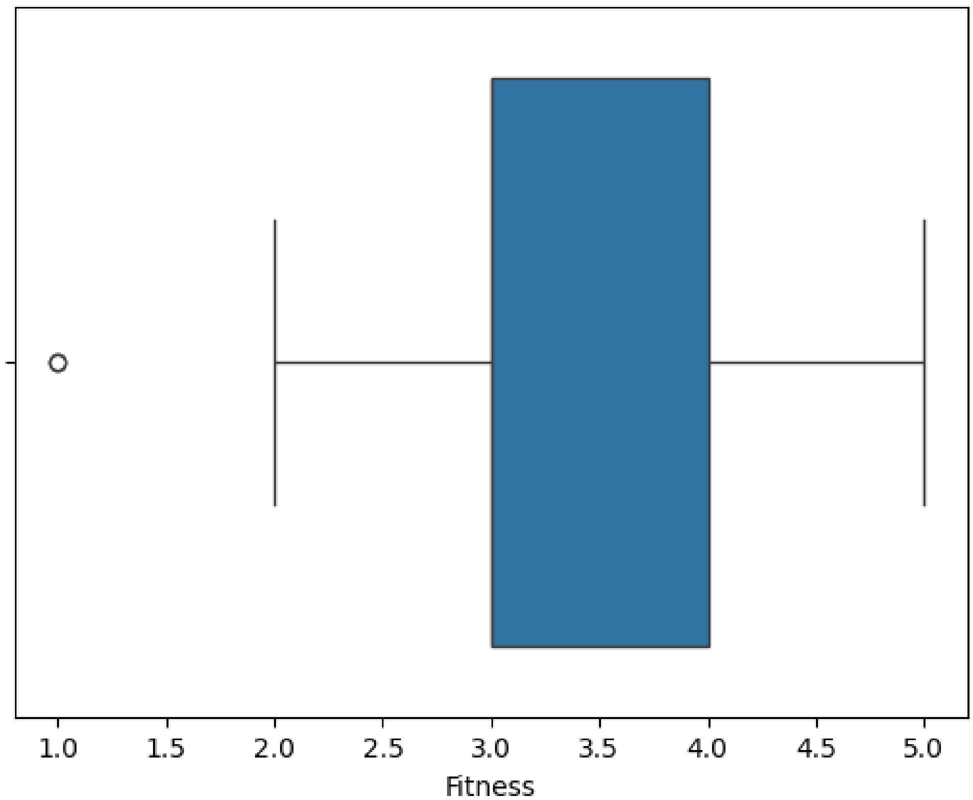
```
Usage_75=np.percentile(df['Usage'],75)
Usage_outlier=Usage_75+(1.5*Usage_IQR)
df[df['Usage']>Usage_outlier]['Usage'].value_counts()
# Outlier usage each week are 6 and 7 with 7 and 2 counts respectively
```

[13]:

```
Usage
6    7
7    2
Name: count, dtype: int64
```

[14]:

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

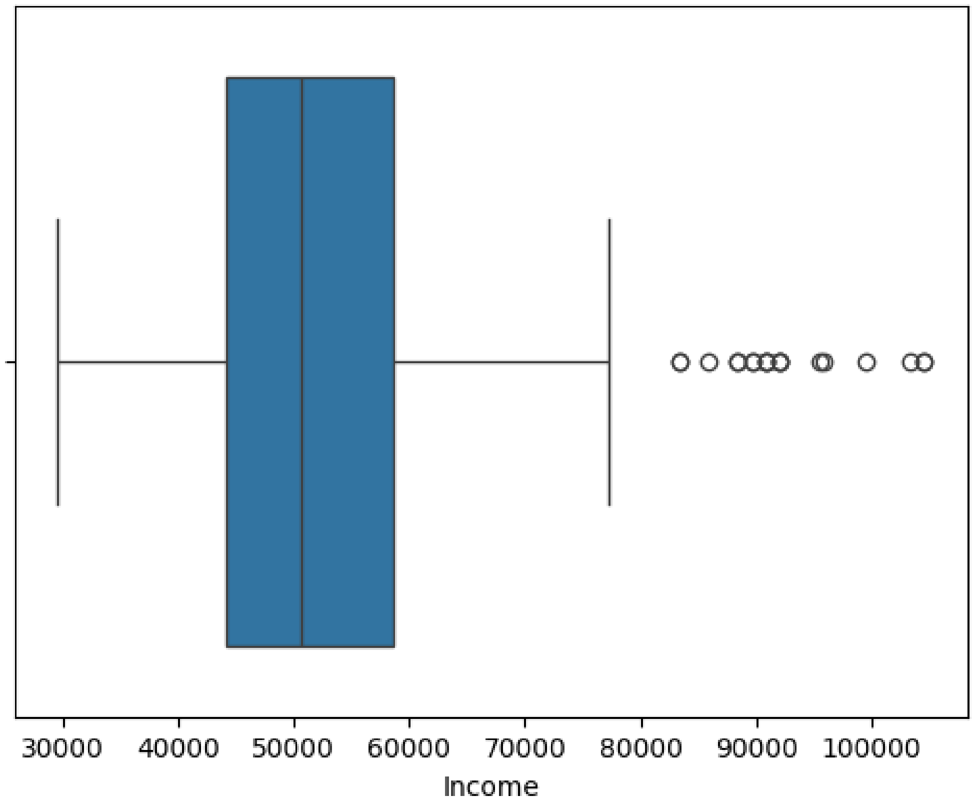


[15]:

```
# Mean for Fitness column is 3.311111 using describe method.
#IQR = 75%-25% = 4 - 3 = 1 - Median
#Outliers for Fitness > Q3(25%) - 1.5XIQR() = 4 + 1.5X1 = 2.5
Fitness_IQR=1
Fitness_25=np.percentile(df['Fitness'],25)
Fitness_outlier=Fitness_25-(1.5*Fitness_IQR)
df[df['Fitness']<Fitness_outlier]['Fitness'].value_counts()
# Outlier Fitness scale is 1 with 2 count
```

[15]: Fitness
1   2
Name: count, dtype: int64

[16]:
```
sns.boxplot(data=df, x="Income")
plt.show()
```



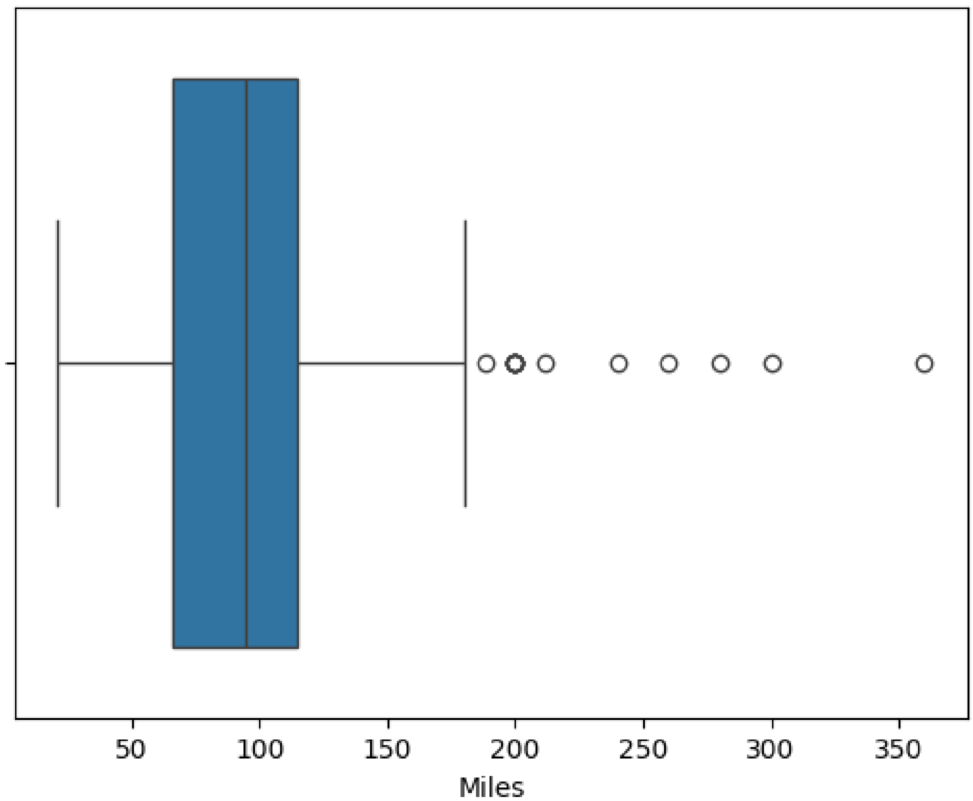
[19]:
```
# Mean for Income column is 53719.577778 using describe method.
#IQR = 75%-25% = 58668.000000 - 44058.750000 = 14609.25 - Median
#Outliers for Income > Q3(75%) + 1.5XIQR() = 58668 + (1.5 X 14609.25) = 80581.
↪875
Income_IQR=14609.25
Income_75=np.percentile(df['Income'],75)
Income_outlier=Income_75+(1.5*Income_IQR)
df[df['Income']>Income_outlier]['Income'].value_counts().sort_index()
# Income has a large number of outliers ranging between 83416 - 104581
```

[19]: Income
83416   2
85906   1
88396   2

```
89641     2
90886     3
92131     3
95508     1
95866     1
99601     1
103336    1
104581    2
Name: count, dtype: int64
```

[18]:

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

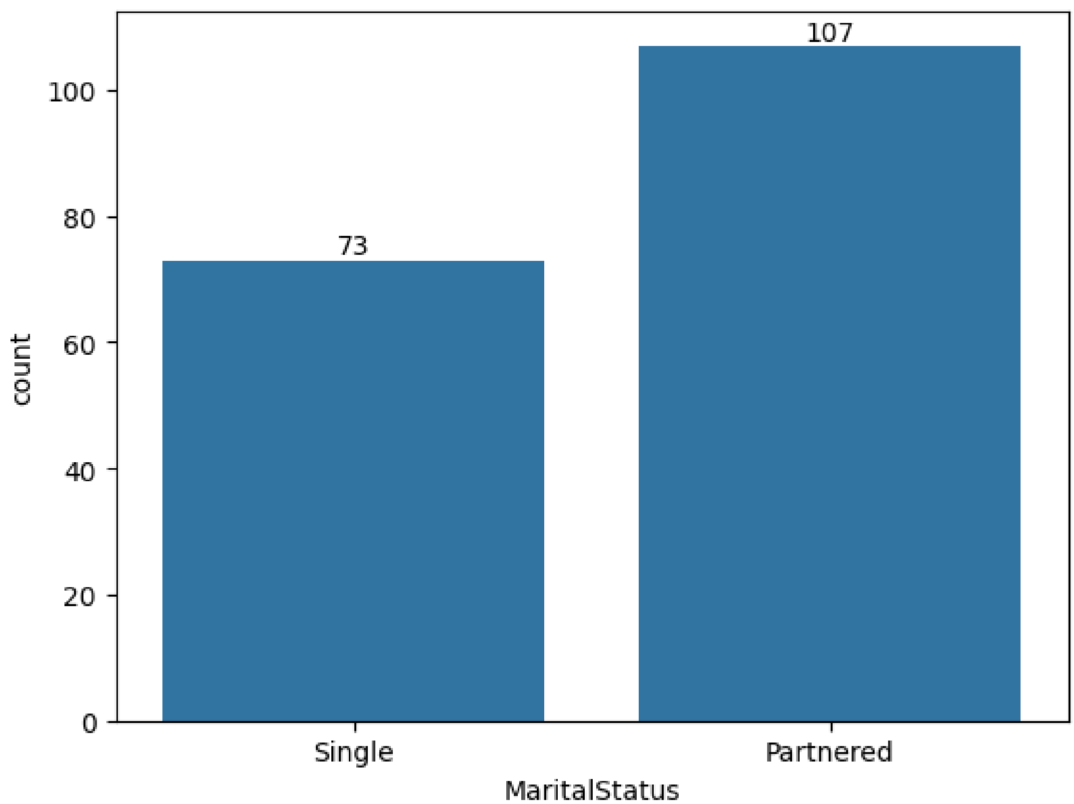


[20]:

```
# Mean for Miles column is 103.194444 using describe method.
#IQR = 75%-25% = 114.750000 - 66.000000 = 48.75 - Median
#Outliers for Miles > Q3(75%) + 1.5XIQR() = 114.75 + (1.5 X 48.75) = 212.25
Miles_IQR=48.75
Miles_75=np.percentile(df['Miles'],75)
Miles_outlier=Miles_75+(1.5*Miles_IQR)
df[df['Miles']>Miles_outlier]['Miles'].value_counts().sort_index()
# Miles has a large number of outliers ranging between 188 - 360
```

[20]:
```
Miles
188    1
200    6
212    1
240    1
260    1
280    1
300    1
360    1
Name: count, dtype: int64
```

[22]:
```
#Check if features like marital status, age have any effect on the product␣
↪purchased (using countplot, histplots, boxplots etc)
# Check effect of Marital Status on purchase
ax=sns.countplot(data=df,x='MaritalStatus')
for container in ax.containers:
    ax.bar_label(container)
plt.show()
# Out of 180 customers 107 are married and 73 are single which is roughly 107/
↪180= 59.4% and 73/180=40.5%
# Indicating Married people are more likely to buy fitness equipment
```

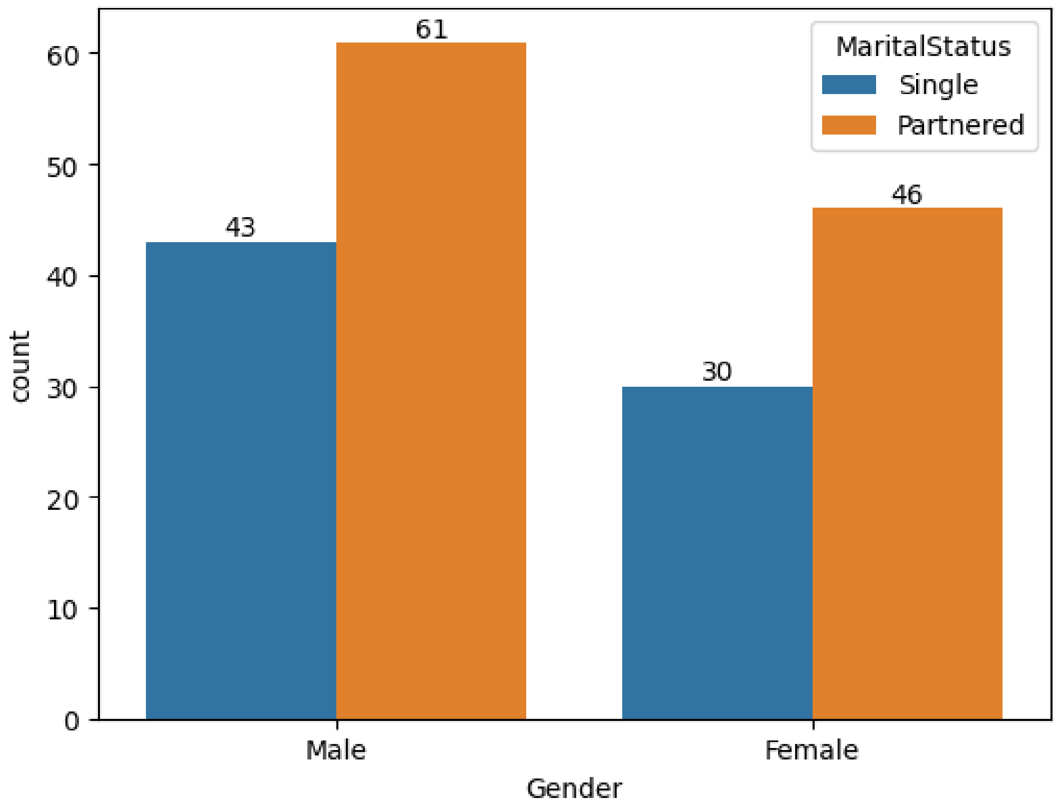


[23]:
```
# Check effect of Age Status on purchase
# For this we will first bin the age into age-groups. The min and max age using␣
↪describe method is 18 and 50 respectively.
# So we will label 18-25 as 'Young Adult', 25-40 as 'Adult' and 40-50 as␣
↪'Middle Age' and >50 as 'Old'
bins_box=[1,25,40,50,100]
bin_labels=['Young Adult','Adult','Middle Age','Old']
df['Age_group']=pd.cut(x=df['Age'],bins=bins_box,labels=bin_labels)
```

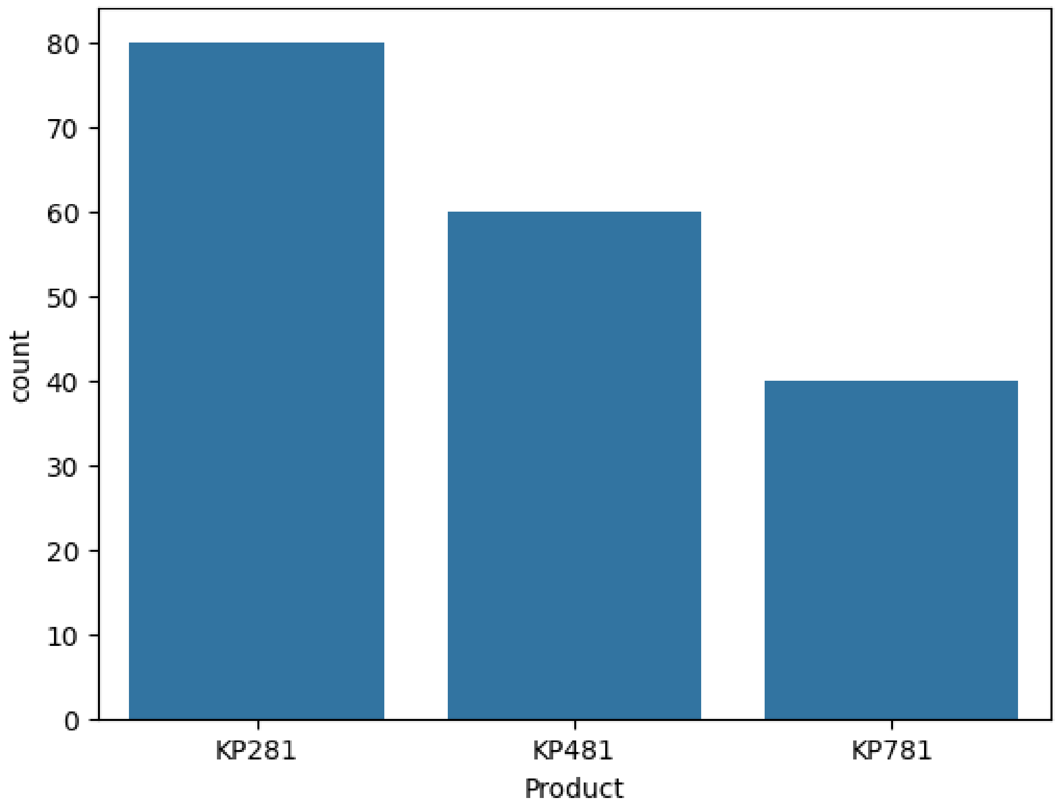
[24]:
```
df['Age_group'].value_counts(normalize=True)
# The distribution indicates that the age group 25-40 is the largest segment␣
↪that buys fitness equipment followed by
# Youg Adults.
# Middle aged people buy the least percentage of fitness equipment.
```

[24]:
```
Age_group
Adult          0.494444
Young Adult    0.438889
Middle Age     0.066667
Old            0.000000
Name: proportion, dtype: float64
```

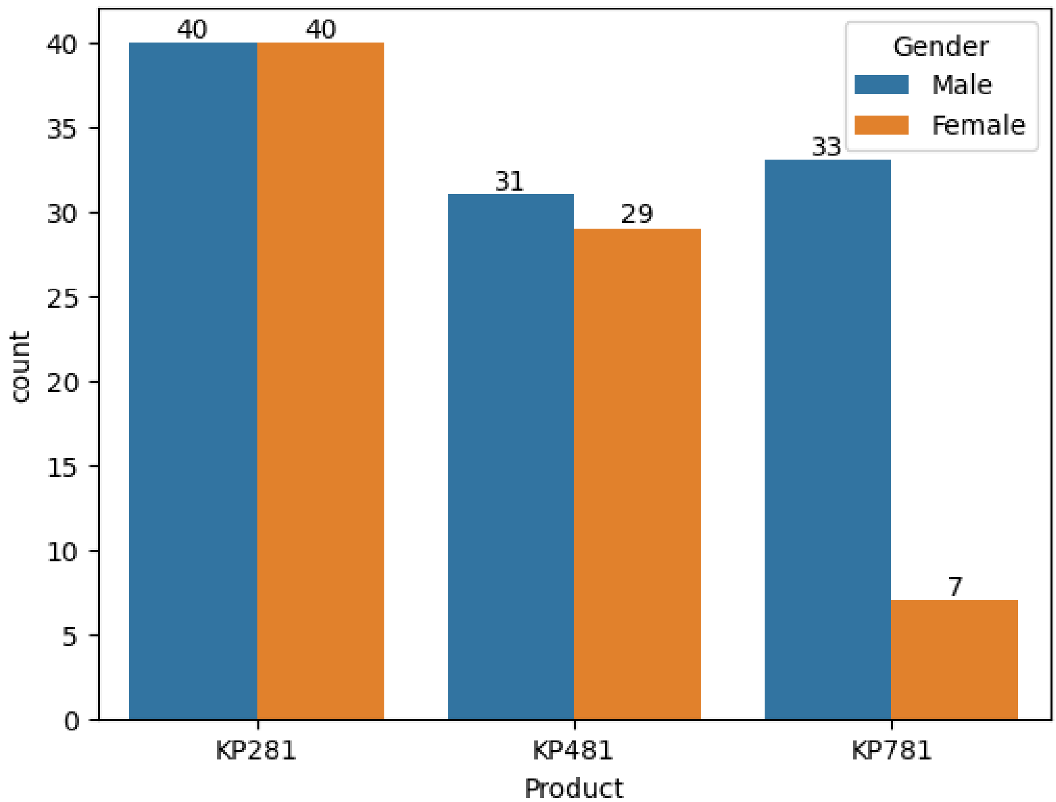
[26]:
```
# Check distribution based on gender
ax=sns.countplot(data=df,x='Gender',hue='MaritalStatus')
for container in ax.containers:
    ax.bar_label(container)
plt.show()
# 104 male members are customers and only 76 are female members out of 180␣
↪customer.
```



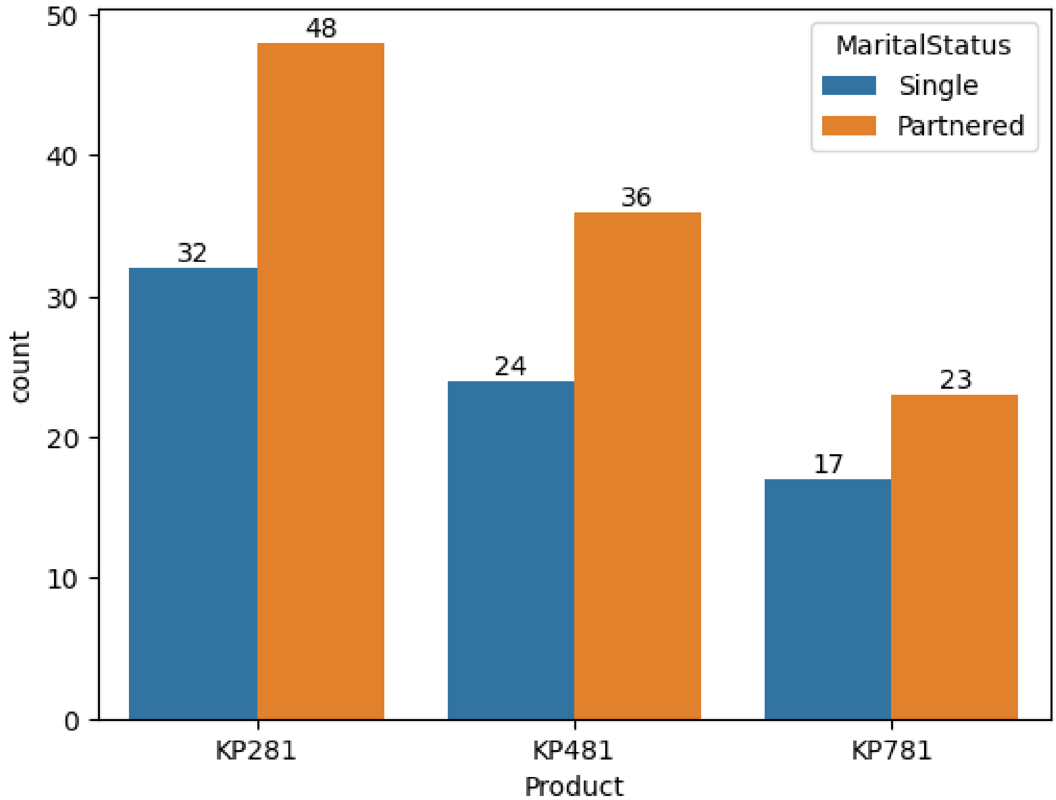
[27]:
```
# Check distribution based on product type
ax=sns.countplot(data=df,x='Product')
# Most sold product is KP281
```



[28]:
```
# Check distribution of sales of product by gender , marital status
ax=sns.countplot(data=df,x='Product',hue='Gender')
for container in ax.containers:
    ax.bar_label(container)
plt.show()
# Inference Male customers tend to buy the expensive models more than female␣
↪customers
```



[29]:
```
# Check distribution of sales of product by gender , marital status
ax=sns.countplot(data=df,x='Product',hue='MaritalStatus')
for container in ax.containers:
    ax.bar_label(container)
plt.show()
# Married customers buy fitness products in larger numbers compared to␣
↪unmarried customers.
```



[30]:
```
# Find weighted sales
def condition(x):
    if x=='KP281':
        return 1500
    elif x=='KP481':
        return 1750
    else:
        return 2500
df['Sales'] = df['Product'].apply(condition)
```

[31]:
```
# Check weighted sales
sales_df=df.groupby('Product').agg({'Sales':['sum','count']})
```

[32]:
```
sales_df=sales_df.reset_index()
```

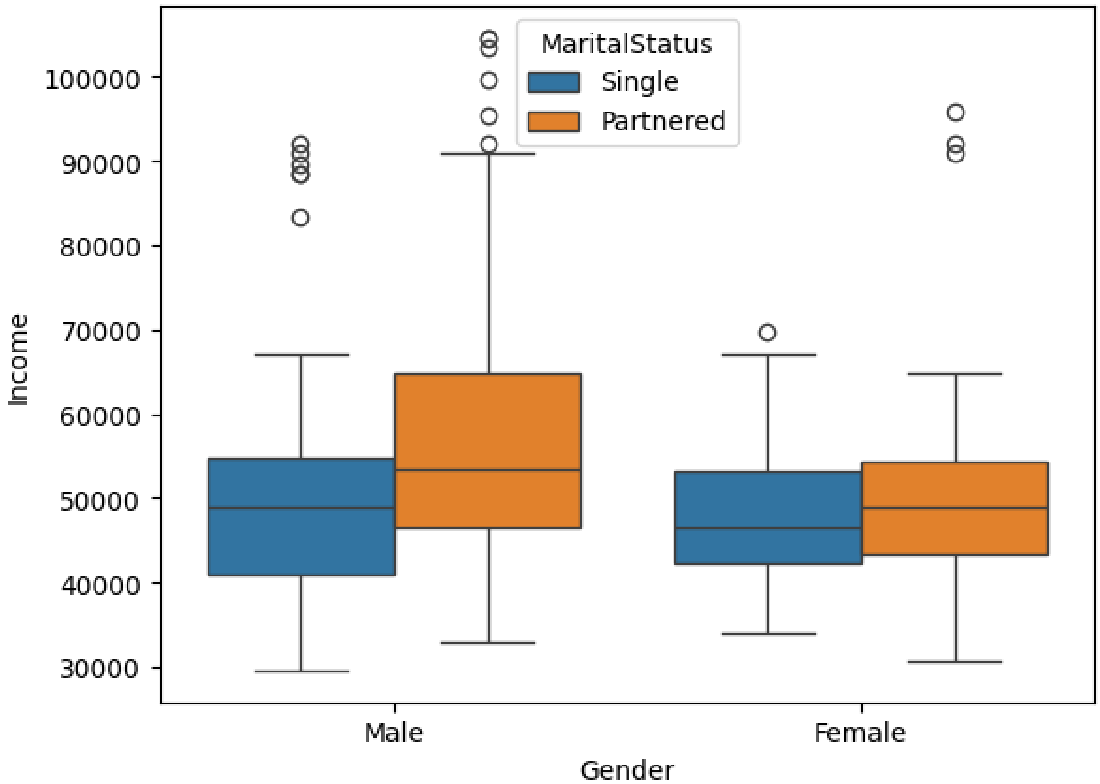
[33]:
```
sales_df.columns=['Product','Total_sales','Counts']
```

[34]:
```
sum_of_sales=sales_df['Total_sales'].sum()
sales_df['Weight']=sales_df['Total_sales']/(180*sum_of_sales)
sales_df
```

[34]:

	Product	Total_sales	Counts	Weight
0	KP281	120000	80	0.002051
1	KP481	105000	60	0.001795
2	KP781	100000	40	0.001709

[35]:
```
# Check outliers in the data
sns.boxplot(data=df,x='Gender',y='Income',hue='MaritalStatus')
plt.show()
```



[ ]:
```
# Quick inferences: Mean Income of Married men is higher than Unmarried men.
#                   Mean Income of Married Women is higher than Unmarried women.
# Number of Income outliers in Men is more than women
# All the outliers are on the higher side.
# Recommendation - Married male customers should be targeted more for sales as␣
↪probable customers group.
```

[36]:
```
# Check outliers for Male married customers
df[(df['Gender']=='Male') & (df['MaritalStatus']=='Partnered')]['Income'].
↪describe()
```

[36]:
```
count        61.000000
mean      59585.704918
std       18766.055777
min       32973.000000
25%       46617.000000
50%       53439.000000
75%       64809.000000
max      104581.000000
Name: Income, dtype: float64
```

[37]:
```
# Mathematically check if the quartiles are correct - 25%
len(df[(df['Gender']=='Male') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']<46617)])/len(df[(df['Gender']=='Male') &␣
 ↪(df['MaritalStatus']=='Partnered')])
```

[37]:
```
0.2459016393442623
```

[38]:
```
male_married_25=np.percentile(df[(df['Gender']=='Male') &␣
 ↪(df['MaritalStatus']=='Partnered')]['Income'],25)
```

[39]:
```
# Mathematically check if the quartiles are correct - 75%
len(df[(df['Gender']=='Male') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']<64809)])/len(df[(df['Gender']=='Male') &␣
 ↪(df['MaritalStatus']=='Partnered')])
```

[39]:
```
0.7377049180327869
```

[40]:
```
male_married_75=np.percentile(df[(df['Gender']=='Male') &␣
 ↪(df['MaritalStatus']=='Partnered')]['Income'],75)
```

[41]:
```
male_married_IQR=male_married_75-male_married_25
```

[42]:
```
#IQR = 75%-25% = 64809 - 46617 = 18192 - Median
#Outliers for Male married customers > Q3(75%) + 1.5XIQR() = 64809 + 1.5X18192␣
 ↪= 92097
male_married_outlier=male_married_75+(1.5*male_married_IQR)
df[(df['Gender']=='Male') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']>male_married_outlier)]['Product'].value_counts()

# All the outliers High Income married males purchased the KP781 treadmill.
```

[42]:
```
Product
KP781    6
Name: count, dtype: int64
```

[43]:
```
# Check outliers for Female married customers
```

```
df[(df['Gender']=='Female') & (df['MaritalStatus']=='Partnered')]['Income'].
 ↪describe()
```

[43]:

```
count       46.000000
mean     50693.760870
std      14343.307149
min      30699.000000
25%      43490.250000
50%      48891.000000
75%      54291.750000
max      95866.000000
Name: Income, dtype: float64
```

[44]:

```
# Mathematically check if the quartiles are correct - 25%
len(df[(df['Gender']=='Female') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']<43490.25)])/len(df[(df['Gender']=='Female') &␣
 ↪(df['MaritalStatus']=='Partnered')])
```

[44]: 0.2608695652173913

[45]:

```
female_married_25=np.percentile(df[(df['Gender']=='Female') &␣
 ↪(df['MaritalStatus']=='Partnered')]['Income'],25)
```

[46]:

```
# Mathematically check if the quartiles are correct - 75%
len(df[(df['Gender']=='Female') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']<54291.75)])/len(df[(df['Gender']=='Female') &␣
 ↪(df['MaritalStatus']=='Partnered')])
```

[46]: 0.7391304347826086

[47]:

```
female_married_75=np.percentile(df[(df['Gender']=='Female') &␣
 ↪(df['MaritalStatus']=='Partnered')]['Income'],75)
```

[48]:

```
female_married_IQR=female_married_75-female_married_25
```

[49]:

```
#IQR = 75%-25% = 54291 - 43490 = 10801 - Median
#Outliers for Female married customers > Q3(75%) + 1.5XIQR() = 54291 + 1.
 ↪5X10801 = 70492.5
female_married_outlier=female_married_75+(1.5*female_married_IQR)
df[(df['Gender']=='Female') & (df['MaritalStatus']=='Partnered') &␣
 ↪(df['Income']>female_married_outlier)]['Product'].value_counts()

# All the outliers High Income married females purchased the KP781 treadmill.
# There are 6 outliers in Married men while 3 outliers in Married Females.
```

[49]:

```
Product
KP781    3
```

```
Name: count, dtype: int64
```

[50]:
```
# Check outliers for Male unmarried customers
df[(df['Gender']=='Male') & (df['MaritalStatus']=='Single')]['Income'].
 ↪describe()
```

[50]:
```
count       43.000000
mean     52274.395349
std      17234.958809
min      29562.000000
25%      40932.000000
50%      48891.000000
75%      54678.500000
max      92131.000000
Name: Income, dtype: float64
```

[51]:
```
#IQR = 75%-25% = 54678.5 - 40932 = 13746.5 - Median
#Outliers for Male unmarried customers > Q3(75%) + 1.5XIQR() = 54678.5 + 1.
 ↪5X13746.5 = 75298.25
df[(df['Gender']=='Male') & (df['MaritalStatus']=='Single') &␣
 ↪(df['Income']>75298.25)]['Product'].value_counts()

# All the outliers High Income unmarried males purchased the KP781 treadmill.
```

[51]:
```
Product
KP781    6
Name: count, dtype: int64
```

[52]:
```
# Check outliers for Female unmarried customers
df[(df['Gender']=='Female') & (df['MaritalStatus']=='Single')]['Income'].
 ↪describe()
```

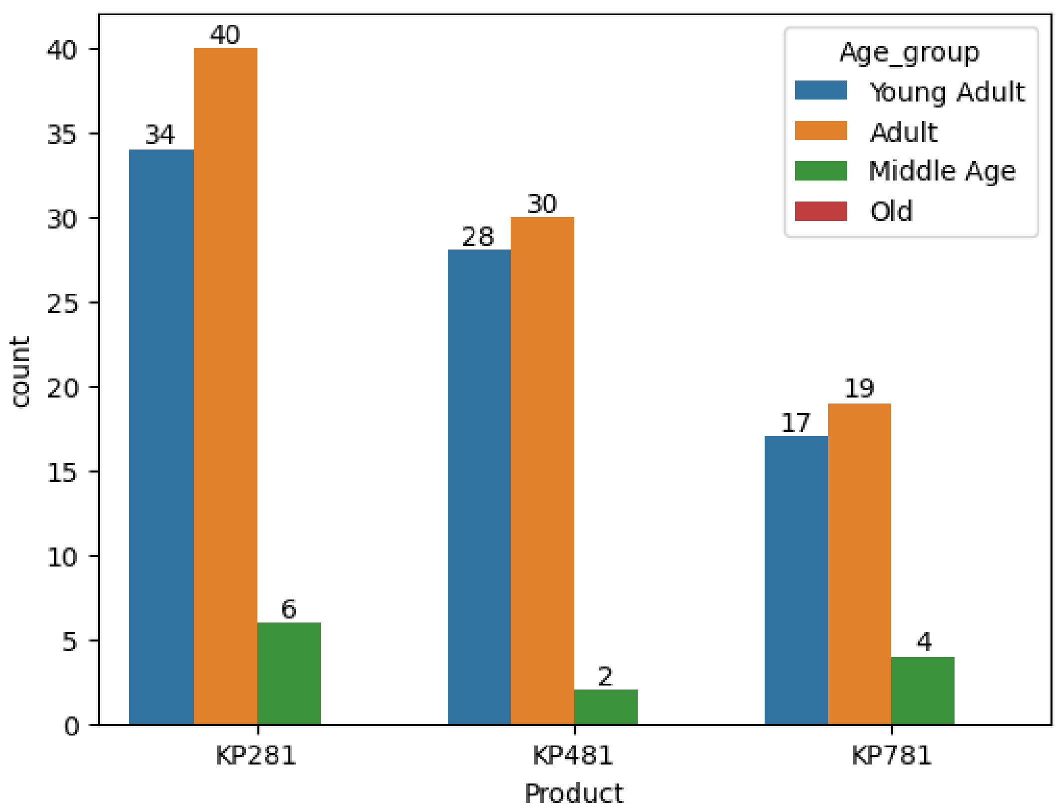
[52]:
```
count       30.00000
mean     48502.80000
std       9251.53287
min      34110.00000
25%      42353.25000
50%      46617.00000
75%      53227.50000
max      69721.00000
Name: Income, dtype: float64
```

[53]:
```
#IQR = 75%-25% = 53227.5 - 42353.25 = 10874.25 - Median
#Outliers for Female unmarried customers > Q3(75%) + 1.5XIQR() = 53227.5 + 1.
 ↪5X10874.25 = 69538.875
df[(df['Gender']=='Female') & (df['MaritalStatus']=='Single') &␣
 ↪(df['Income']>69538.875)]['Product'].value_counts()
```

```
# All the outliers High Income unmarried female purchased the KP781 treadmill.
# There is just 1 outlier in unmarried females compared to 6 outliers in␣
↪unmarried males.
```

[53]:
```
Product
KP781    1
Name: count, dtype: int64
```

[54]:
```
# Check distribution of sales of product by age
ax=sns.countplot(data=df,x='Product',hue='Age_group')
for container in ax.containers:
    ax.bar_label(container)
plt.show()
# Inference - Adults(25-40) are usually the highest buyers surpassing any other␣
↪group.
# Also price of treadmill plays a factor as we see dignificant decline in␣
↪purchasing proportions as the model price increases.
```



[55]:
```
#Representing the marginal probability like - what percent of customers have␣
↪purchased KP281, KP481, or KP781 in a table (can use pandas.crosstab here)
```

```
pd.crosstab(df['Product'],df['Gender'],margins=True,normalize=True)
# The breakup shows that ~44% customers have purchased KP281, 33% have␣
↪purchased KP481 and 22% have purchased KP781 models
# The breakup also shows that KP281 has equal puchase from Females and Males
# KP481 is purchased by 16% Females and 17.2% males
# KP781 is purchased by 3.8% Females and 18% males
```

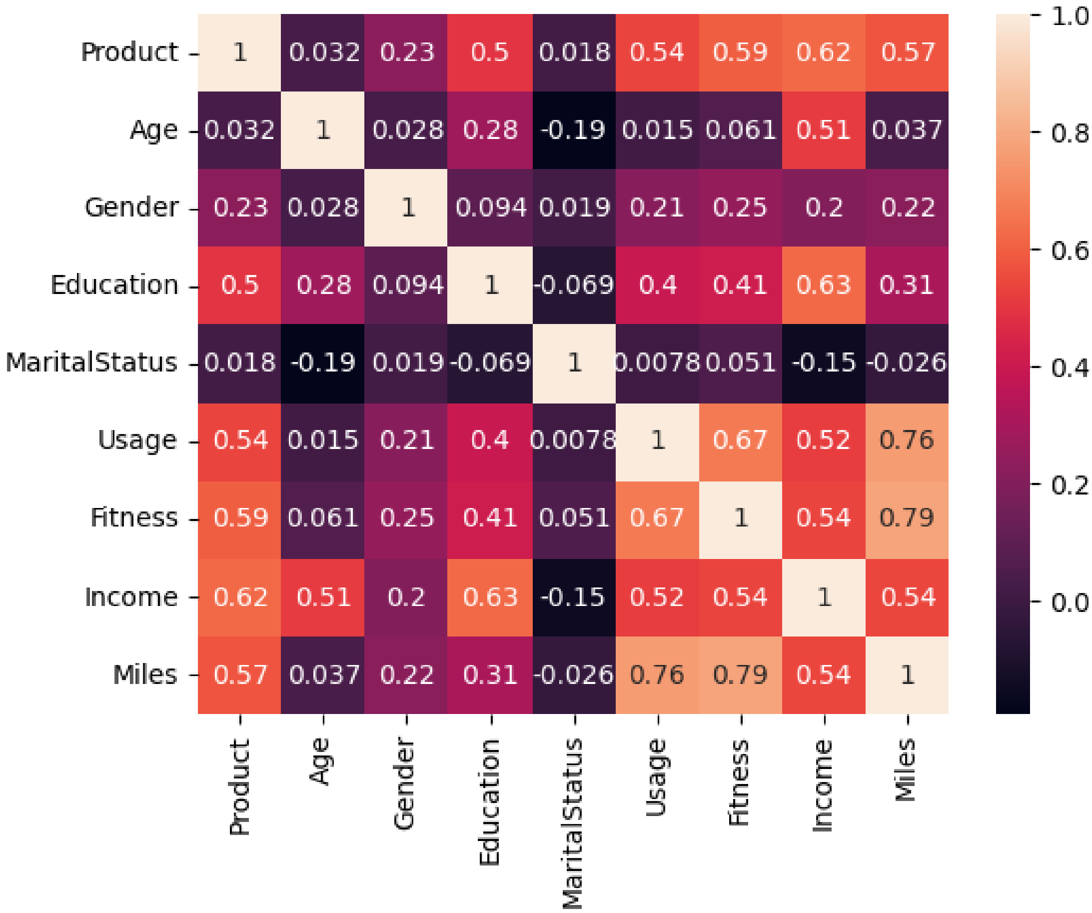
[55]:

Gender	Female	Male	All
Product			
KP281	0.222222	0.222222	0.444444
KP481	0.161111	0.172222	0.333333
KP781	0.038889	0.183333	0.222222
All	0.422222	0.577778	1.000000

[56]:

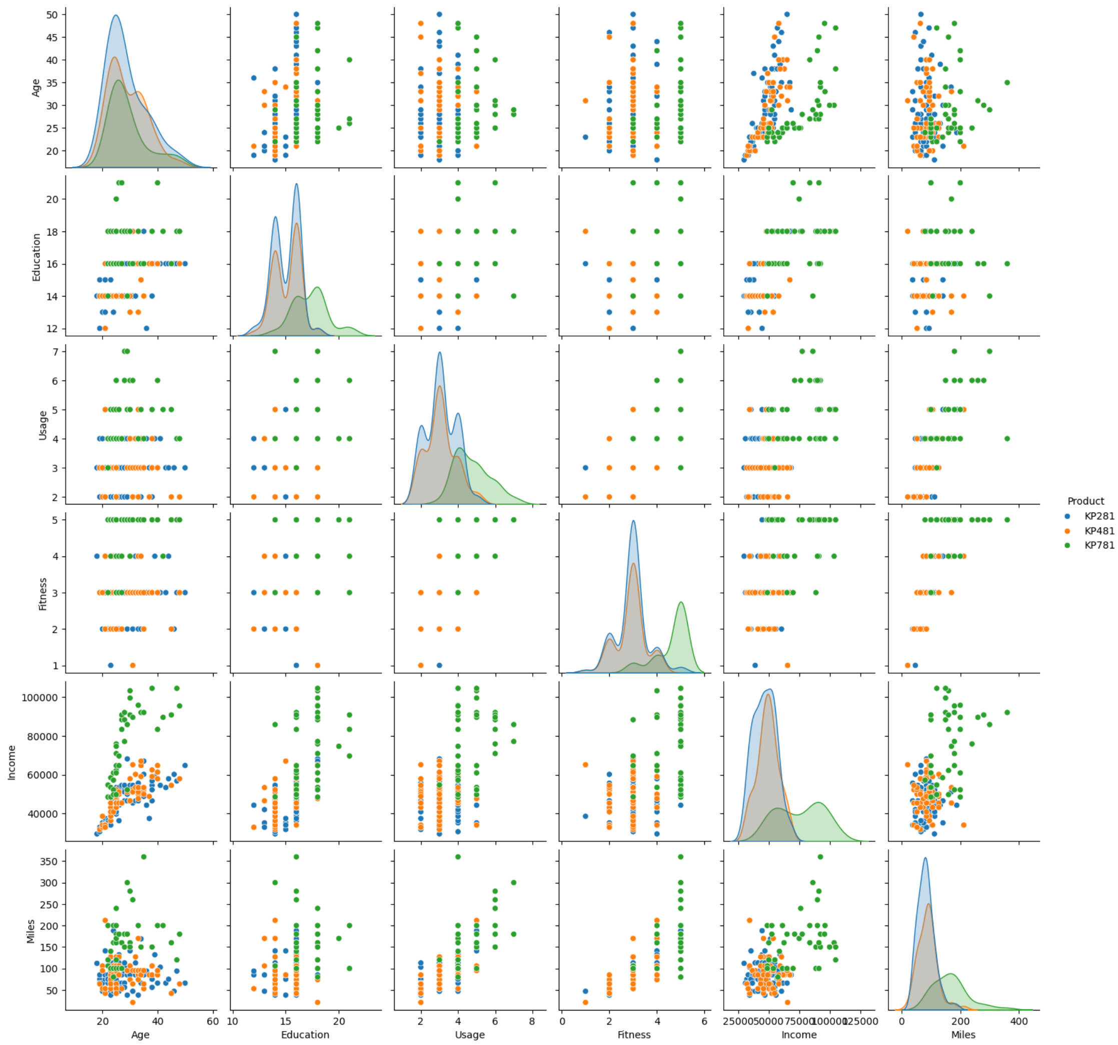
```
#Check correlation among different factors using heat maps or pair plots.
# For this we will need to convert the categorical values into numerical values
df_new=df[['Product','Age','Gender','Education','MaritalStatus','Usage','Fitness','Income','Mi
↪copy()
# We have categorical columns which have nominal data like Product, Gender ,␣
↪MaritalStatus
# Convert nominal catergorical columns
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
for col in ['Product', 'Gender' , 'MaritalStatus']:
  df_new[col] = encoder.fit_transform(df_new[col])
#df_new['Product'].value_counts()
sns.heatmap(df_new.corr(),annot=True)
```

[56]: <Axes: >




```
[ ]: # Inference - Usage, Miles and Fitness are positively correlated. This means␣
     ↪the higher the Usage and Miles per week the fitter the customer.
     # Inference - Age , Income and MaritalStatus are negatively correlated. This␣
     ↪means they are not related.
     # Inference - Usage, Fitness, Income and Miles have a positive correlation with␣
     ↪Education. Higher the education better the Income.
     # Inference - Age and Income are positively correlated.
```

```
[60]: sns.pairplot(data=df,hue='Product')
      plt.show()
```



[57]:
```
#With all the above steps you can answer questions like: What is the␣
↪probability of a male customer buying a KP781 treadmill?
pd.crosstab(df['Product'],df['Gender'],margins=True)
```

[57]:

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

[58]:
```
# Probablity of male customers buying KP781 treadmill
# Number of Males buying KP781/Number of all Male customers
print((33/104)*100) # 31.73%
```

31.73076923076923

[62]:
```
pd.crosstab(df["Product"],df['Gender'])/pd.crosstab(df["Product"],df['Gender']).
 ↪sum()
```

[62]:

Gender	Female	Male
Product		
KP281	0.526316	0.384615
KP481	0.381579	0.298077
KP781	0.092105	0.317308

[89]:
```
#Customer Profiling - Categorization of users.
# Customer profiling - Customer profiling is a marketing strategy that uses␣
 ↪data to create a picture of the
# perfect customer who will interact with your product or service.
# As per definition profiling is specific to product or service so we will go␣
 ↪by treadmill types.

# Profile for all features( We have already performed Product analysis by␣
 ↪Gender, Marital Status and Age)
# Check distribution of sales of product by Education, Usage, Fitness, Income,␣
 ↪Miles
df[df['Product']=='KP281'].describe()
```

[89]:

	Age	Education	Usage	Fitness	Income	Miles	\
count	80.000000	80.000000	80.000000	80.00000	80.00000	80.000000	
mean	28.550000	15.037500	3.087500	2.96250	46418.02500	82.787500	
std	7.221452	1.216383	0.782624	0.66454	9075.78319	28.874102	
min	18.000000	12.000000	2.000000	1.00000	29562.00000	38.000000	
25%	23.000000	14.000000	3.000000	3.00000	38658.00000	66.000000	
50%	26.000000	16.000000	3.000000	3.00000	46617.00000	85.000000	
75%	33.000000	16.000000	4.000000	3.00000	53439.00000	94.000000	
max	50.000000	18.000000	5.000000	5.00000	68220.00000	188.000000	

	Sales
count	80.0
mean	1500.0
std	0.0
min	1500.0
25%	1500.0
50%	1500.0
75%	1500.0
max	1500.0

[ ]:
```
# Customer profile for KP281
# Probablity of Male buying = 38%, Probablity of Female buying = 52%
# Total purchase numbers for Male and Female are same
```

```
# Married people are more likely to buy this type of treadmill( 32 - Single and␣
 ↪48 - Married)
# Highest number of purchases are made by 25-40 age group follwed by 18-25 age␣
 ↪group
# Education of highest purchasers has a mean of 15.03 years lies between 14-16␣
 ↪years
# Highest usage is between 3 to 4
# Highest fitness level is 3
# Income mean is ~ 46418
# Miles mean is ~ 82.78
```

[90]:
```
df[df['Product']=='KP481'].describe()
```

[90]:
```
             Age  Education      Usage   Fitness        Income       Miles  \
count  60.000000  60.000000  60.000000  60.00000     60.000000   60.000000
mean   28.900000  15.116667   3.066667   2.90000  48973.650000   87.933333
std     6.645248   1.222552   0.799717   0.62977   8653.989388   33.263135
min    19.000000  12.000000   2.000000   1.00000  31836.000000   21.000000
25%    24.000000  14.000000   3.000000   3.00000  44911.500000   64.000000
50%    26.000000  16.000000   3.000000   3.00000  49459.500000   85.000000
75%    33.250000  16.000000   3.250000   3.00000  53439.000000  106.000000
max    48.000000  18.000000   5.000000   4.00000  67083.000000  212.000000

        Sales
count    60.0
mean   1750.0
std       0.0
min    1750.0
25%    1750.0
50%    1750.0
75%    1750.0
max    1750.0
```

[ ]:
```
# Customer profile for KP481
# Probablity of Male buying = 30%, Probablity of Female buying = 38%
# Total purchase numbers for Male and Female are approximately same ( 31 - M␣
 ↪and 29 - F)
# Married people are more likely to buy this type of treadmill( 24 - Single and␣
 ↪36 - Married)
# Highest number of purchases are made by 25-40 age group follwed by 18-25 age␣
 ↪group
# Education of highest purchasers has mean of 15.116 and lies between 14-16␣
 ↪years
# Highest usage is between 3 to 3.25
# Highest fitness level is 3
# Income mean is ~ 48973.7
```

```
# Miles mean is ~ 87.933
```

[91]:
```
df[df['Product']=='KP781'].describe()
```

[91]:
```
             Age  Education      Usage    Fitness        Income       Miles  \
count  40.000000  40.000000  40.000000  40.000000      40.00000   40.000000
mean   29.100000  17.325000   4.775000   4.625000   75441.57500  166.900000
std     6.971738   1.639066   0.946993   0.667467   18505.83672   60.066544
min    22.000000  14.000000   3.000000   3.000000   48556.00000   80.000000
25%    24.750000  16.000000   4.000000   4.000000   58204.75000  120.000000
50%    27.000000  18.000000   5.000000   5.000000   76568.50000  160.000000
75%    30.250000  18.000000   5.000000   5.000000   90886.00000  200.000000
max    48.000000  21.000000   7.000000   5.000000  104581.00000  360.000000

        Sales
count    40.0
mean   2500.0
std       0.0
min    2500.0
25%    2500.0
50%    2500.0
75%    2500.0
max    2500.0
```

[ ]:
```
# Customer profile for KP781
# Probablity of Male buying = 32%, Probablity of Female buying = 9%
# Total purchase numbers for Male are higher than Females ( 33 - M and 7 - F)
# Married people are more likely to buy this type of treadmill( 17 - Single and␣
↪23 - Married)
# Highest number of purchases are made by 25-40 age group follwed by 18-25 age␣
↪group( purchases were approximately same)
# Education of highest purchasers range between 16-18 years( Outliers␣
↪indicating 20 and 21 years as well) - Highly educated
# Highest usage is 4.77 (between 4 to 5) - High Usage
# Highest fitness level is 4.62 ( between 4-5 ) - Very fit individuals
# Income mean is ~ 75441.6 - High Income
# Miles mean is ~ 166.9 - High Miles per week
```

[85]:
```
#Probability- marginal
for i in ['Product','Gender','Education','MaritalStatus','Usage','Fitness']:
  print(df[i].value_counts(normalize=True).sort_index(),"\n")
```

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

```
Gender
Female    0.422222
Male      0.577778
Name: proportion, dtype: float64

Education
12    0.016667
13    0.027778
14    0.305556
15    0.027778
16    0.472222
18    0.127778
20    0.005556
21    0.016667
Name: proportion, dtype: float64

MaritalStatus
Partnered    0.594444
Single       0.405556
Name: proportion, dtype: float64

Usage
2    0.183333
3    0.383333
4    0.288889
5    0.094444
6    0.038889
7    0.011111
Name: proportion, dtype: float64

Fitness
1    0.011111
2    0.144444
3    0.538889
4    0.133333
5    0.172222
Name: proportion, dtype: float64
```

[ ]:
```
# conditional probability.
# We have already seen the conditional probablity based on gender. Now we will
 ↪check conditional probablity for the following scenarios.
# Conditional probablity based on Age, Income and Fitness
```

[93]:
```
# Conditional probablity of purchase based on Age
```

```
pd.crosstab(df['Product'],df['Age_group'])/pd.
 ↪crosstab(df['Product'],df['Age_group']).sum()
# Results show that KP281 has normal spread across all age groups
```

[93]:

Age_group	Young Adult	Adult	Middle Age
Product			
KP281	0.43038	0.449438	0.500000
KP481	0.35443	0.337079	0.166667
KP781	0.21519	0.213483	0.333333

[95]:

```
# Conditional probablity of purchase based on Fitness
pd.crosstab(df['Product'],df['Fitness'])/pd.
 ↪crosstab(df['Product'],df['Fitness']).sum()
# Results show that a high fitness person is more likely to opt for KP781␣
 ↪treadmill.~ 93.5% probablity for a fitness level 5 person
```

[95]:

Fitness	1	2	3	4	5
Product					
KP281	0.5	0.538462	0.556701	0.375000	0.064516
KP481	0.5	0.461538	0.402062	0.333333	0.000000
KP781	0.0	0.000000	0.041237	0.291667	0.935484

[97]:

```
# Check probablity of purchase of Product types by Income
# For this we will first bin the Income into groups.
bins_box=[0,20000,40000,60000,80000,100000,120000]
bin_labels=['20k','40k','60k','80k','100k','120k']
df['Income_group']=pd.cut(x=df['Income'],bins=bins_box,labels=bin_labels)
pd.crosstab(df['Product'],df['Income_group'])/pd.
 ↪crosstab(df['Product'],df['Income_group']).sum()
# Results indicate that the maximum purchases are made by the 40k group for␣
 ↪KP281 while KP481 and KP781 were purchased by 60k and 80k Income group
# Very few purchases made for KP781 by 100k and 120k group.
```

[97]:

Income_group	40k	60k	80k	100k	120k
Product					
KP281	0.71875	0.481132	0.260870	0.0	0.0
KP481	0.28125	0.415094	0.304348	0.0	0.0
KP781	0.00000	0.103774	0.434783	1.0	1.0

[ ]:

```
#Inferences
#**********
# Men are more likely to buy fitness equipment compared to females.
# Married people are more likely to buy fitness equipment.
# Both outliers High Income married/unmarried males purchased the KP781␣
 ↪treadmill.
# Male customers tend to buy the expensive models more than female customers.
```

```
# The distribution indicates that the age group 25-40 is the largest segment␣
↪that buys fitness equipment followed by 18-25.
# Price of treadmill plays a factor as we see significant decline in purchasing␣
↪proportions as the model price increases.
# Usage, Fitness, Income and Miles have a positive correlation with Education.␣
↪Higher the education better the Income. Also higher the Income fitter the␣
↪person with more usage.



#Recommendation
# Company should target more features for Male customers.
# Lower and mid segment treadmills( KP281 and KP481 ) can be offered at␣
↪discounts to target the audience( Students/Professionals <= 16 years of␣
↪education )
# Company should bundle offers that are lucrative for married indivduals - eg.␣
↪onsite maintenance free etc.
# Company should promote the KP781 treadmill for high earning, highly educated,␣
↪highly fit customers.
# Company should include more features that would interest 25-40 age group. eg.␣
↪Ipod connectivity, Internet features, Youtube/Streaming capabilities etc.
```