In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

# Client: Fitness Brand

# Problem Statement: Brand wants to investigate whether there are differences across the product "treadmill" with respect to customer characteristics.

## Why do we want to analyze?

## To provide a better recommendation of the treadmills to the new customers

In [3]:
```
leau/project/fitness_brand/d2beiqkhq929f0.cloudfront.net_public_assets_assets_000_001_125_original_treadmill.csv_1639992749.txt")
```

In [4]:
```
df
```

Out[4]:

| | Product | Age | Gender | Education | MaritalStatus | Usage | Fitness | Income | Miles |
|---|---|---|---|---|---|---|---|---|---|
| **0** | KP281 | 18 | Male | 14 | Single | 3 | 4 | 29562 | 112 |
| **1** | KP281 | 19 | Male | 15 | Single | 2 | 3 | 31836 | 75 |
| **2** | KP281 | 19 | Female | 14 | Partnered | 4 | 3 | 30699 | 66 |
| **3** | KP281 | 19 | Male | 12 | Single | 3 | 3 | 32973 | 85 |
| **4** | KP281 | 20 | Male | 13 | Partnered | 4 | 2 | 35247 | 47 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **175** | KP781 | 40 | Male | 21 | Single | 6 | 5 | 83416 | 200 |
| **176** | KP781 | 42 | Male | 18 | Single | 5 | 4 | 89641 | 200 |
| **177** | KP781 | 45 | Male | 16 | Single | 5 | 5 | 90886 | 160 |
| **178** | KP781 | 47 | Male | 18 | Partnered | 4 | 5 | 104581 | 120 |
| **179** | KP781 | 48 | Male | 18 | Partnered | 4 | 5 | 95508 | 180 |

180 rows × 9 columns

# Data structure and characteristics

In [5]:
```
#shape of the data
df.shape
```

Out[5]:
```
(180, 9)
```

In [6]:
```
#Column names
df.columns
```

Out[6]:
```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles'],
      dtype='object')
```

### Column Information/description

1. Product Purchased: KP281, KP481, or KP781
2. Age:In years
3. Gender: Male/Female
4. Education: In years
5. MaritalStatus: Single or partnered
6. Usage: The average number of times the customer plans to use the treadmill each week.
7. Income: Annual income (in USD)
8. Fitness: Self-rated fitness on a 1-to-5 scale, where 1 is the poor shape and 5 is the excellent shape.
9. Miles: The average number of miles the customer expects to walk/run each week

### Product Portfolio:

1. The KP281 is an entry-level treadmill that sells for USD1,500.
2. The KP481 is for mid-level runners that sell for USD1,750.
3. The KP781 treadmill is having advanced features that sell for USD2,500.

In [4]:
```
# adding treadmill category in table
Level={"KP281":"Entry-level","KP481":"Mid-level","KP781":"High-level"}
Price={"KP281":1500,"KP481":1750,"KP781":2500}
df["Level"]=df["Product"].map(Level)
df["Price"]=df["Product"].map(Price)
```

In [5]:
```
df
```

Out[5]:

| | Product | Age | Gender | Education | MaritalStatus | Usage | Fitness | Income | Miles | Level | Price |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | KP281 | 18 | Male | 14 | Single | 3 | 4 | 29562 | 112 | Entry-level | 1500 |
| 1 | KP281 | 19 | Male | 15 | Single | 2 | 3 | 31836 | 75 | Entry-level | 1500 |
| 2 | KP281 | 19 | Female | 14 | Partnered | 4 | 3 | 30699 | 66 | Entry-level | 1500 |
| 3 | KP281 | 19 | Male | 12 | Single | 3 | 3 | 32973 | 85 | Entry-level | 1500 |
| 4 | KP281 | 20 | Male | 13 | Partnered | 4 | 2 | 35247 | 47 | Entry-level | 1500 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 175 | KP781 | 40 | Male | 21 | Single | 6 | 5 | 83416 | 200 | High-level | 2500 |
| 176 | KP781 | 42 | Male | 18 | Single | 5 | 4 | 89641 | 200 | High-level | 2500 |
| 177 | KP781 | 45 | Male | 16 | Single | 5 | 5 | 90886 | 160 | High-level | 2500 |
| 178 | KP781 | 47 | Male | 18 | Partnered | 4 | 5 | 104581 | 120 | High-level | 2500 |
| 179 | KP781 | 48 | Male | 18 | Partnered | 4 | 5 | 95508 | 180 | High-level | 2500 |

180 rows × 11 columns

In [46]:
```
# statistical summary
df.describe(include="all")
```

Out[46]:

| | Product | Age | Gender | Education | MaritalStatus | Usage | Fitness | Income | Miles | Level | Price |
|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 180 | 180.000000 | 180 | 180.000000 | 180 | 180.000000 | 180.000000 | 180.000000 | 180.000000 | 180 | 180.000000 |
| unique | 3 | NaN | 2 | NaN | 2 | NaN | NaN | NaN | NaN | 3 | NaN |
| top | KP281 | NaN | Male | NaN | Partnered | NaN | NaN | NaN | NaN | Entry-level | NaN |
| freq | 80 | NaN | 104 | NaN | 107 | NaN | NaN | NaN | NaN | 80 | NaN |
| mean | NaN | 28.788889 | NaN | 15.572222 | NaN | 3.455556 | 3.311111 | 53719.577778 | 103.194444 | NaN | 1805.555556 |
| std | NaN | 6.943498 | NaN | 1.617055 | NaN | 1.084797 | 0.958869 | 16506.684226 | 51.863605 | NaN | 387.978895 |
| min | NaN | 18.000000 | NaN | 12.000000 | NaN | 2.000000 | 1.000000 | 29562.000000 | 21.000000 | NaN | 1500.000000 |
| 25% | NaN | 24.000000 | NaN | 14.000000 | NaN | 3.000000 | 3.000000 | 44058.750000 | 66.000000 | NaN | 1500.000000 |
| 50% | NaN | 26.000000 | NaN | 16.000000 | NaN | 3.000000 | 3.000000 | 50596.500000 | 94.000000 | NaN | 1750.000000 |
| 75% | NaN | 33.000000 | NaN | 16.000000 | NaN | 4.000000 | 4.000000 | 58668.000000 | 114.750000 | NaN | 1750.000000 |
| max | NaN | 50.000000 | NaN | 21.000000 | NaN | 7.000000 | 5.000000 | 104581.000000 | 360.000000 | NaN | 2500.000000 |

In [47]:
```
# datatypes
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    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 
 9   Level          180 non-null    object
 10  Price          180 non-null    int64 
dtypes: int64(7), object(4)
memory usage: 15.6+ KB
```

In [48]:
```python
# to find missing values
df.isna().sum()
# conclusion: we do not have any missing value in database
```

Out[48]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
Level            0
Price            0
dtype: int64
```

In [6]:
```python
df.nunique() #to identify unique values in database
```

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

In [9]:
```python
df["Product"].unique()
#conclusion: Fitness brand has 3 types of treadmill that we want to sell to our customers
```

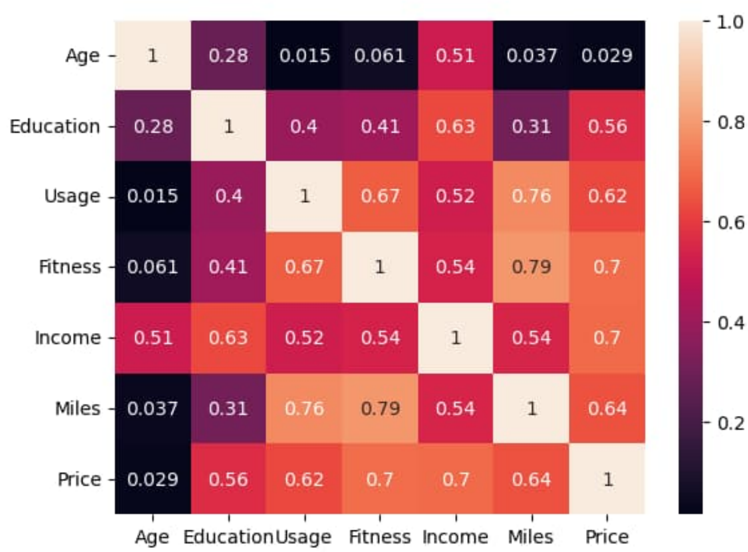
Out[9]:
```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

In [10]:
```python
#to find correlation between categories (added Price of each treadmill purposely to analyze the correclation between "treadmill"
df1=df.corr()
```

```
C:\Users\harmeet-talreja\AppData\Local\Temp\ipykernel_19708\737269955.py:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.
  df1=df.corr()
```

# Correlation between Product and other categories

In [11]:
```python
sns.heatmap(data=df1, annot=True)
plt.show()
```



In [13]:
```
sns.pairplot(data=df,hue="Product", kind="reg")
plt.show()
```


### Summary:

Price of a treadmill is highly positively correlated with "Fitness", "Income" . It shows that:

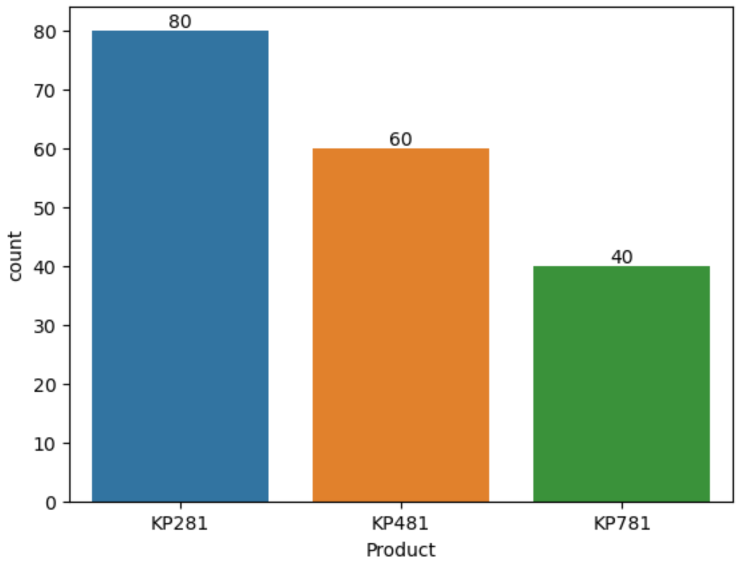
1. Conclusion 1: Best Featured treadmill "KP781" can be demanded more by: a) the person who is good in shape b) the person who has high income
2. Conclusion 2: "Price of a treadmill" is moderately positively correlated with "Education". It shows that as years of education increases advanced featured treadmill will be demanded but not in same amount.
3. Concluion 3: High positive Correlation between "fitness" and "Miles". i.e. If you want to be in a good shape, you expect on an average to walk/run more number of miles each week
4. Concluion 4: Less positive Correlation between "Age" and other categories. i.e. age doesn't matter any of the category defined above especially the type of product customer purchases.

# Product type count

In [16]:
```
df["Product"].value_counts()
#conclusion: "KP281" is most selling product
```

Out[16]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [19]:
```
x=sns.countplot(data=df,x="Product")
x.bar_label(x.containers[0])
plt.show()
#conclusion: "KP281" is most selling product
```




### Summary:

"KP281" is the most selling product

# Relationship between Age and Product

In [73]:
```
df.groupby("Product")["Age"].mean()
# On an average, 28-29 years of people purchase treadmill
```

Out[73]:
```
Product
KP281    28.55
KP481    28.90
KP781    29.10
Name: Age, dtype: float64
```

In [74]:
```
df.groupby("Product")["Age"].median()
```

Out[74]:
```
Product
KP281    26.0
KP481    26.0
KP781    27.0
Name: Age, dtype: float64
```

In [86]:
```
x=df.groupby("Product")["Age"].max()
x
```

Out[86]:
```
Product
KP281    50
KP481    48
KP781    48
Name: Age, dtype: int64
```

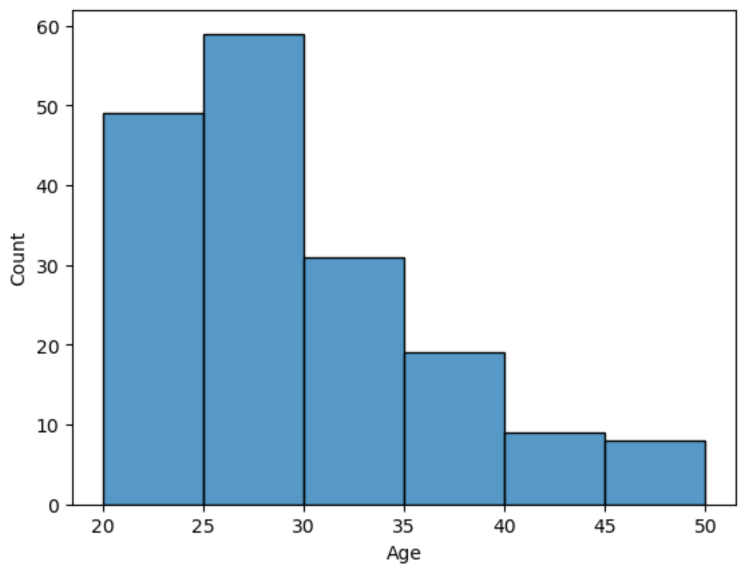
In [100]:
```
y=df.groupby("Product")["Age"].min()
y
```

Out[100]:
```
Product
KP281    18
KP481    19
KP781    22
Name: Age, dtype: int64
```

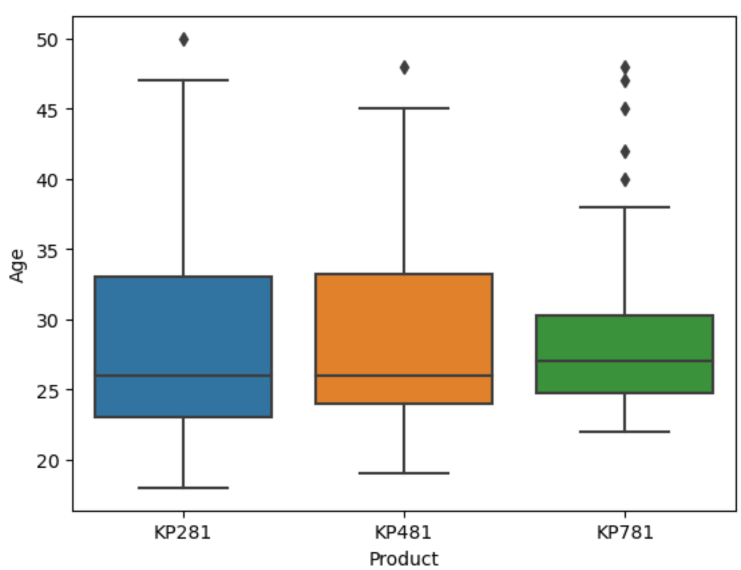
In [88]:
```
Age_range=x-y
Age_range
```

Out[88]:
```
Product
KP281    32
KP481    29
KP781    26
Name: Age, dtype: int64
```

In [55]:
```
sns.histplot(data=df,x="Age",bins=[20,25,30,35,40,45,50])
plt.show()
#Conclusion: Most of the treadmill is bought by people within age group of 25-30 without any difference in product type
```




In [39]:
```
# Outliers related to Age group
sns.boxplot(data=df, x="Product",y="Age")
plt.show()
#Conclusion: Most of the outliers can be seen for product "KP781"
```



In [153]:
```
###### Detecting Outlier age for product "KP781"
```

In [101]:
```
age_75=df.groupby("Product")["Age"].quantile(0.75)
age_75
```

Out[101]:
```
Product
KP281    33.00
KP481    33.25
KP781    30.25
Name: Age, dtype: float64
```

In [102]:
```
age_25=df.groupby("Product")["Age"].quantile(0.25)
age_25
```

Out[102]:
```
Product
KP281    23.00
KP481    24.00
KP781    24.75
Name: Age, dtype: float64
```

In [104]:
```
age_IQR=age_75-age_25
age_IQR
```

Out[104]:
```
Product
KP281    10.00
KP481     9.25
KP781     5.50
Name: Age, dtype: float64
```

In [127]:
```
age_whisker_lower=age_25-(1.5*age_IQR)
age_whisker_lower.reset_index()
```

Out[127]:

| | Product | Age |
|---|---|---|
| 0 | KP281 | 8.000 |
| 1 | KP481 | 10.125 |
| 2 | KP781 | 16.500 |

In [144]:
```
age_whisker_upper=age_25+(1.5*age_IQR)
age_whisker_upper=age_whisker_upper.reset_index()
age_whisker_upper["Age"]=age_whisker_upper["Age"].astype(int)
age_whisker_upper
```

Out[144]:

| | Product | Age |
|---|---|---|
| 0 | KP281 | 38 |
| 1 | KP481 | 37 |
| 2 | KP781 | 33 |

In [148]:
```
age_whisker_upper_new=age_whisker_upper[age_whisker_upper["Product"]=="KP781"]
age_whisker_upper_new
```

Out[148]:

| | Product | Age |
|---|---|---|
| 2 | KP781 | 33 |

In [130]:
```
df_new=df.loc[df["Product"]=="KP781"]
df_new
```

Out[130]:

| | Product | Age | Gender | Education | MaritalStatus | Usage | Fitness | Income | Miles | Level | Price |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **140** | KP781 | 22 | Male | 14 | Single | 4 | 3 | 48658 | 106 | High-level | 2500 |
| **141** | KP781 | 22 | Male | 16 | Single | 3 | 5 | 54781 | 120 | High-level | 2500 |
| **142** | KP781 | 22 | Male | 18 | Single | 4 | 5 | 48556 | 200 | High-level | 2500 |
| **143** | KP781 | 23 | Male | 16 | Single | 4 | 5 | 58516 | 140 | High-level | 2500 |
| **144** | KP781 | 23 | Female | 18 | Single | 5 | 4 | 53536 | 100 | High-level | 2500 |
| **145** | KP781 | 23 | Male | 16 | Single | 4 | 5 | 48556 | 100 | High-level | 2500 |
| **146** | KP781 | 24 | Male | 16 | Single | 4 | 5 | 61006 | 100 | High-level | 2500 |
| **147** | KP781 | 24 | Male | 18 | Partnered | 4 | 5 | 57271 | 80 | High-level | 2500 |
| **148** | KP781 | 24 | Female | 16 | Single | 5 | 5 | 52291 | 200 | High-level | 2500 |
| **149** | KP781 | 24 | Male | 16 | Single | 5 | 5 | 49801 | 160 | High-level | 2500 |
| **150** | KP781 | 25 | Male | 16 | Partnered | 4 | 5 | 49801 | 120 | High-level | 2500 |
| **151** | KP781 | 25 | Male | 16 | Partnered | 4 | 4 | 62251 | 160 | High-level | 2500 |
| **152** | KP781 | 25 | Female | 18 | Partnered | 5 | 5 | 61006 | 200 | High-level | 2500 |
| **153** | KP781 | 25 | Male | 18 | Partnered | 4 | 3 | 64741 | 100 | High-level | 2500 |
| **154** | KP781 | 25 | Male | 18 | Partnered | 6 | 4 | 70966 | 180 | High-level | 2500 |
| **155** | KP781 | 25 | Male | 18 | Partnered | 6 | 5 | 75946 | 240 | High-level | 2500 |
| **156** | KP781 | 25 | Male | 20 | Partnered | 4 | 5 | 74701 | 170 | High-level | 2500 |
| **157** | KP781 | 26 | Female | 21 | Single | 4 | 3 | 69721 | 100 | High-level | 2500 |
| **158** | KP781 | 26 | Male | 16 | Partnered | 5 | 4 | 64741 | 180 | High-level | 2500 |
| **159** | KP781 | 27 | Male | 16 | Partnered | 4 | 5 | 83416 | 160 | High-level | 2500 |
| **160** | KP781 | 27 | Male | 18 | Single | 4 | 3 | 88396 | 100 | High-level | 2500 |
| **161** | KP781 | 27 | Male | 21 | Partnered | 4 | 4 | 90886 | 100 | High-level | 2500 |
| **162** | KP781 | 28 | Female | 18 | Partnered | 6 | 5 | 92131 | 180 | High-level | 2500 |
| **163** | KP781 | 28 | Male | 18 | Partnered | 7 | 5 | 77191 | 180 | High-level | 2500 |
| **164** | KP781 | 28 | Male | 18 | Single | 6 | 5 | 88396 | 150 | High-level | 2500 |
| **165** | KP781 | 29 | Male | 18 | Single | 5 | 5 | 52290 | 180 | High-level | 2500 |
| **166** | KP781 | 29 | Male | 14 | Partnered | 7 | 5 | 85906 | 300 | High-level | 2500 |
| **167** | KP781 | 30 | Female | 16 | Partnered | 6 | 5 | 90886 | 280 | High-level | 2500 |
| **168** | KP781 | 30 | Male | 18 | Partnered | 5 | 4 | 103336 | 160 | High-level | 2500 |
| **169** | KP781 | 30 | Male | 18 | Partnered | 5 | 5 | 99601 | 150 | High-level | 2500 |
| **170** | KP781 | 31 | Male | 16 | Partnered | 6 | 5 | 89641 | 260 | High-level | 2500 |
| **171** | KP781 | 33 | Female | 18 | Partnered | 4 | 5 | 95866 | 200 | High-level | 2500 |
| **172** | KP781 | 34 | Male | 16 | Single | 5 | 5 | 92131 | 150 | High-level | 2500 |
| **173** | KP781 | 35 | Male | 16 | Partnered | 4 | 5 | 92131 | 360 | High-level | 2500 |
| **174** | KP781 | 38 | Male | 18 | Partnered | 5 | 5 | 104581 | 150 | High-level | 2500 |
| **175** | KP781 | 40 | Male | 21 | Single | 6 | 5 | 83416 | 200 | High-level | 2500 |
| **176** | KP781 | 42 | Male | 18 | Single | 5 | 4 | 89641 | 200 | High-level | 2500 |
| **177** | KP781 | 45 | Male | 16 | Single | 5 | 5 | 90886 | 160 | High-level | 2500 |
| **178** | KP781 | 47 | Male | 18 | Partnered | 4 | 5 | 104581 | 120 | High-level | 2500 |
| **179** | KP781 | 48 | Male | 18 | Partnered | 4 | 5 | 95508 | 180 | High-level | 2500 |

In [162]:
```
age_outlier=df_new[df_new["Age"]>33]["Age"].unique()
age_outlier
```

Out[162]:
```
array([34, 35, 38, 40, 42, 45, 47, 48], dtype=int64)
```

### Summary: Customer Profile related to age for each Product:

1. On an average, 28-29 years of people purchase all types of treadmill
2. "KP781" is demanded by high age group customers
3. Most of the treadmill is bought by people within age group of 25-30 without any difference in product type whereas data is majorly concentrated in 28-29 years of customers

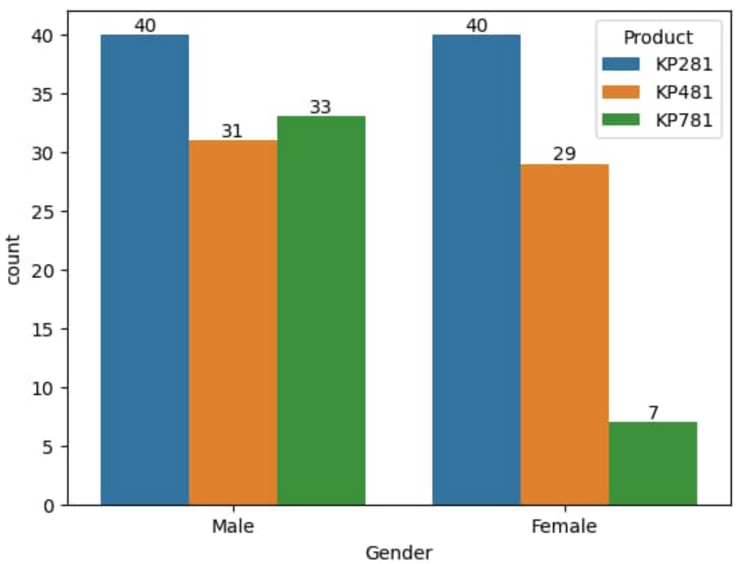
## Relationship between Gender and Product type

In [78]:
```
#contingency Table
pd.crosstab(index=df["Product"],
           columns=df["Gender"],
           margins=True)
```

Out[78]:

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

In [77]:
```
x=sns.countplot(data=df,x="Gender",hue="Product")
x.bar_label(x.containers[0])
x.bar_label(x.containers[1])
x.bar_label(x.containers[2])
plt.show()
```




In [174]:
```
marginal_prob=(df.groupby("Product")["Gender"].value_counts()/df.groupby("Product")["Gender"].count())*100
marginal_prob
# Conclusion: high featured treadmill "KP781" is mostly demanded by Male. There is no significant difference in gender from other
```

Out[174]:
```
Product  Gender
KP281    Female    50.000000
         Male      50.000000
KP481    Male      51.666667
         Female    48.333333
KP781    Male      82.500000
         Female    17.500000
Name: Gender, dtype: float64
```

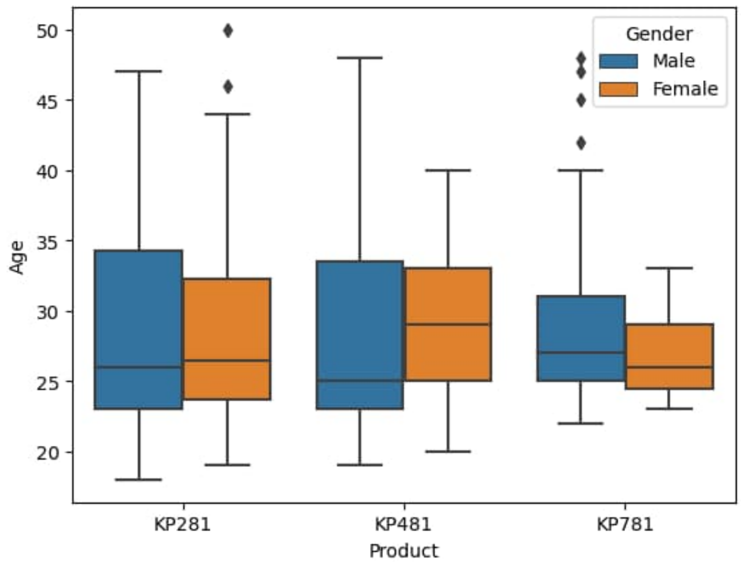
In [175]:
```
gender_prob=(df["Gender"].value_counts()/df["Gender"].count())*100
gender_prob
#Conclusion: There is high probability that male purchases treadmill than female
```

Out[175]:
```
Male      57.777778
Female    42.222222
Name: Gender, dtype: float64
```

In [176]:
```
Conditional_prob=(df.groupby("Gender")["Product"].value_counts()/df.groupby("Gender")["Product"].count())*100
Conditional_prob
#Conclusion: Females are most likely to purchase "KP281" treadmill type while males are almost equilikely to purchase all types c
```

Out[176]:
```
Gender  Product
Female  KP281      52.631579
        KP481      38.157895
        KP781       9.210526
Male    KP281      38.461538
        KP781      31.730769
        KP481      29.807692
Name: Product, dtype: float64
```

In [167]:
```
# Outliers related to Age group and Gender
sns.boxplot(data=df, x="Product",y="Age",hue="Gender")
plt.show()
#Conclusion 1: Median age of male is lower than Female except for product "KP781".
#Conclusion 2: Distribution of age group of all type of products are positively skewed for Male i.e. data is majorly concetrated
```

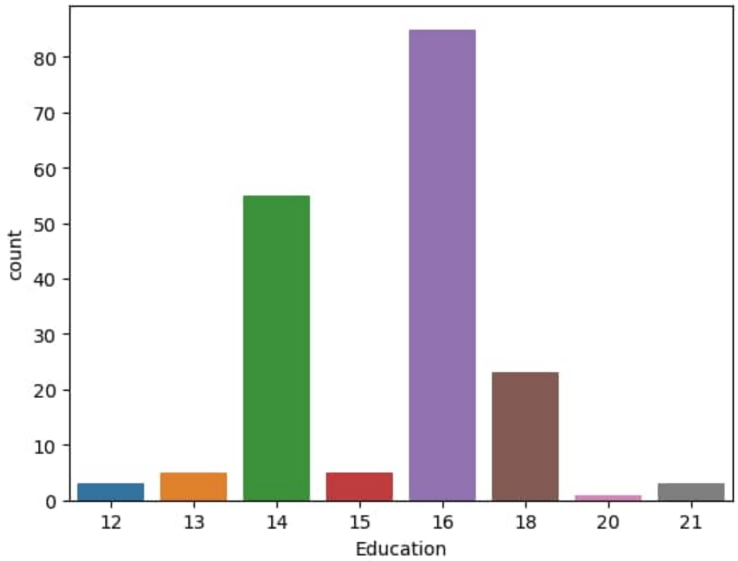



### Summary: Customer Profile related to gender for each Product:

1. high featured treadmill "KP781" is mostly demanded by Male. There is no significant difference in gender from other 2 product
2. There is high probability that male purchases treadmill than female
3. Females are most likely to purchase "KP281" treadmill type while males are almost equilikely to purchase all types of treadmill
4. Median age of male is lower than Female except for product "KP781".
5. Distribution of age group of all type of products are positively skewed for Male i.e. data is majorly concetrated in higher age group (greater than 25) while for female, data is normally or symetrically distributed i.e.there is no much variation between minimum and maximum age group for female

# Relationship between Education and Product type

In [182]:
```
sns.countplot(data=df,x="Education")
plt.show()
#conclusion: Most of the product is purchased by population having 14-16 years of education.
```



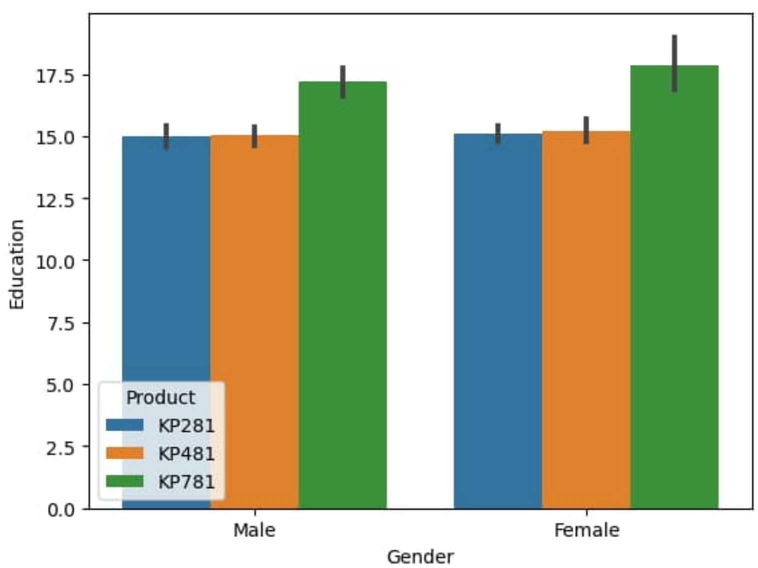

In [177]:
```
df.groupby("Product")["Education"].mean()
```

Out[177]:
```
Product
KP281    15.037500
KP481    15.116667
KP781    17.325000
Name: Education, dtype: float64
```

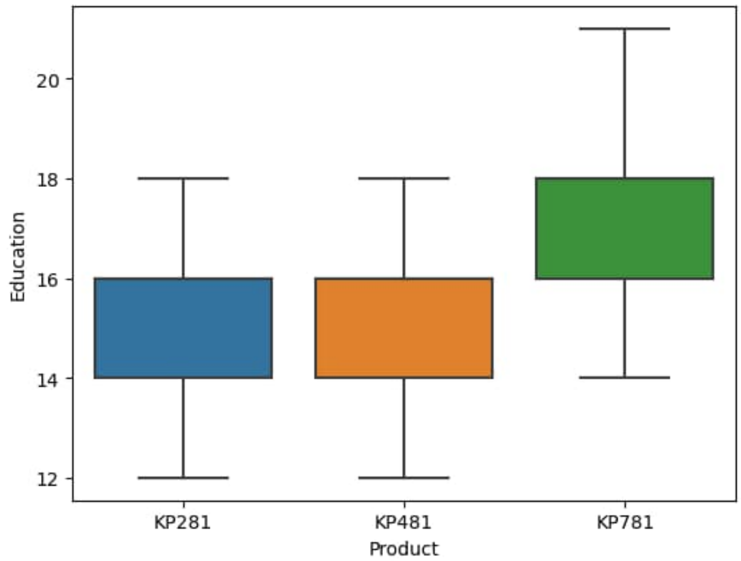
In [179]:
```
df.groupby("Product")["Education"].median()
```

Out[179]:
```
Product
KP281    16.0
KP481    16.0
KP781    18.0
Name: Education, dtype: float64
```

In [183]:
```
sns.barplot(data=df,x="Gender",y="Education",hue="Product")
plt.show()
#conclusion: On an average, advanced treadmill is purchased by population having greater years of education. there is no signific
```



In [184]:
```
sns.boxplot(data=df,x="Product",y="Education")
plt.show()
#Conclusion: there is no outlier.
```

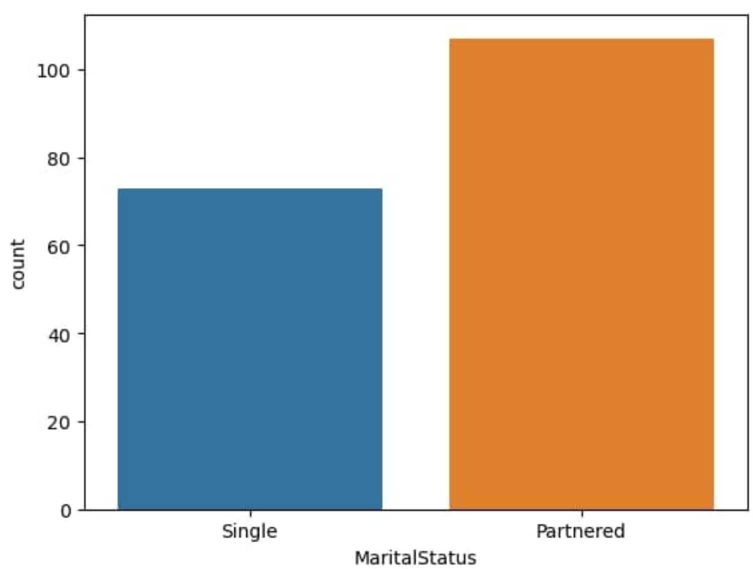



### Summary: Customer Profile related to Education years for each Product:

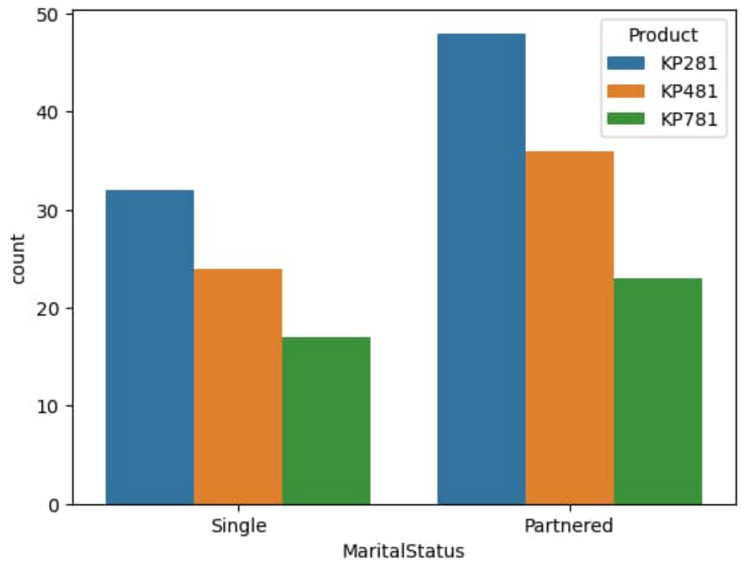
On an average, advanced treadmill is purchased by population having greater years of education. there is no significant difference between low level and mid-level treadmill customers

## Relationship between Marital status and Product type

In [187]:
```
sns.countplot(data=df,x="MaritalStatus")
plt.show()
# conclusion: Partnered population is more likely to purchase treadmill than single
```



In [188]:
```
sns.countplot(data=df,x="MaritalStatus",hue="Product")
plt.show()
#conclusion: All types of product is purchased majorly by partnered population than single
```




In [190]:
```
pd.crosstab(index=df["Product"],columns=df["MaritalStatus"],margins=True)
```

Out[190]:

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

marginal_prob_MS=df.groupby("Product")["MaritalStatus"].value_counts()/df.groupby("Product")["MaritalStatus"].count()

In [192]:
```
marginal_prob_MS=(df.groupby("Product")["MaritalStatus"].value_counts()/df.groupby("Product")["MaritalStatus"].count())*100
marginal_prob_MS
```

Out[192]:
```
Product  MaritalStatus
KP281    Partnered        60.0
         Single           40.0
KP481    Partnered        60.0
         Single           40.0
KP781    Partnered        57.5
         Single           42.5
Name: MaritalStatus, dtype: float64
```

In [193]:
```
conditional_prob_MS=(df.groupby("MaritalStatus")["Product"].value_counts()/df.groupby("MaritalStatus")["Product"].count())*100
conditional_prob_MS
#conclusion: there is no significant difference between choice of treadmill between Partnered and Single. Most of the population
```

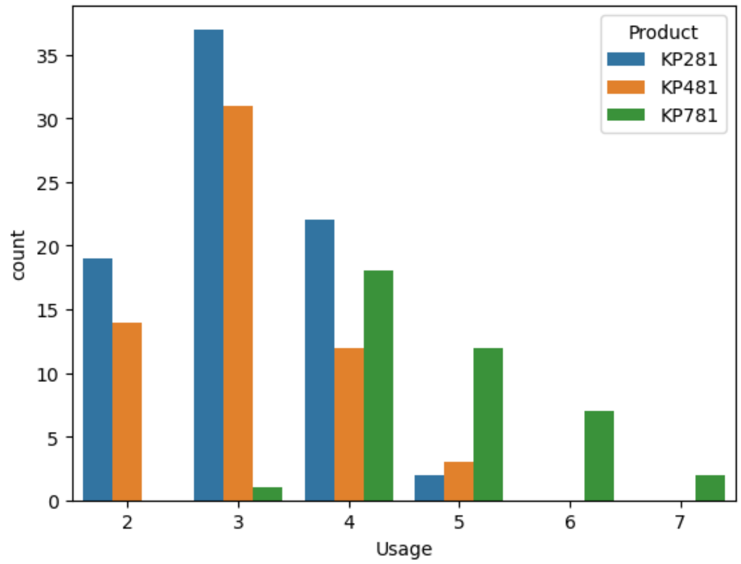
Out[193]:
```
MaritalStatus  Product
Partnered      KP281      44.859813
               KP481      33.644860
               KP781      21.495327
Single         KP281      43.835616
               KP481      32.876712
               KP781      23.287671
Name: Product, dtype: float64
```

### Summary: Customer Profile related to Marital Status for each Product:

1. There is no significant difference between choice of treadmill between Partnered and Single. Most of the population either single or partnered is most likely to purchase "low-level" treadmill.
2. Partnered population is most likely to purchase treadmill than Single

## Relationship between Usage and Product choice

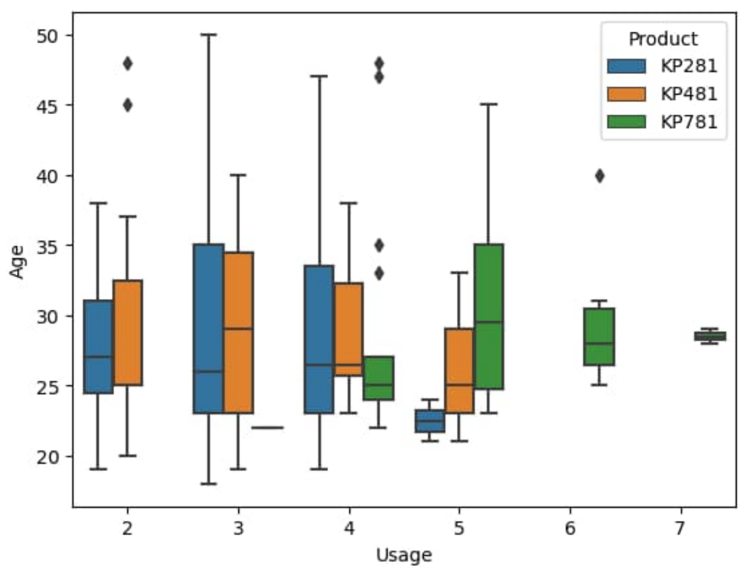
In [196]:
```
sns.countplot(data=df,x="Usage",hue="Product")
plt.show()
#conclusion: On an average, customers plan to use "low-level" and "mid-level" treadmills 3 times a week while customers plan to u
```




In [197]:
```
df.groupby("Product")["Usage"].mean()
```

Out[197]:
```
Product
KP281    3.087500
KP481    3.066667
KP781    4.775000
Name: Usage, dtype: float64
```

In [201]:
```
sns.boxplot(data=df,x="Usage",y="Age",hue="Product")
plt.show()
#conclusion: On an average, "Low-level" and "mid-level" treadmills are used 3-4 times a week in most cases but young male in age
```



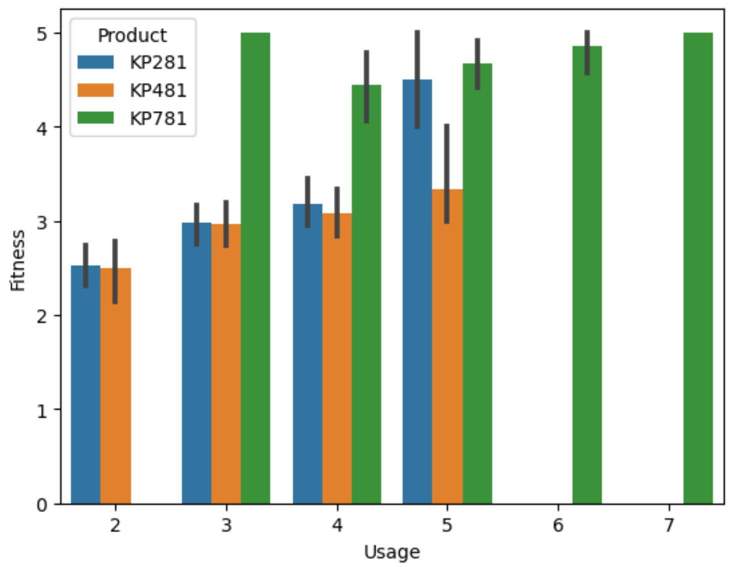
In [203]:
```
min_usage=df.groupby("Product")["Usage"].min()
min_usage
```

Out[203]:
```
Product
KP281    2
KP481    2
KP781    3
Name: Usage, dtype: int64
```

In [205]:
```
max_usage=df.groupby("Product")["Usage"].max()
max_usage
#conclusion: at max, low-level and mid-level treadmill can be used 5 times while high level can be used 7 times a week.
```

Out[205]:
```
Product
KP281    5
KP481    5
KP781    7
Name: Usage, dtype: int64
```

In [212]:
```
sns.barplot(data=df,x="Usage",y="Fitness",hue="Product")
plt.show()
#conclusion: those who used "high-level" treadmills are on an average good in shape. the striking feature of "low-level" treadmil
```




### Summary: Customer Profile related to Usage for each Product:

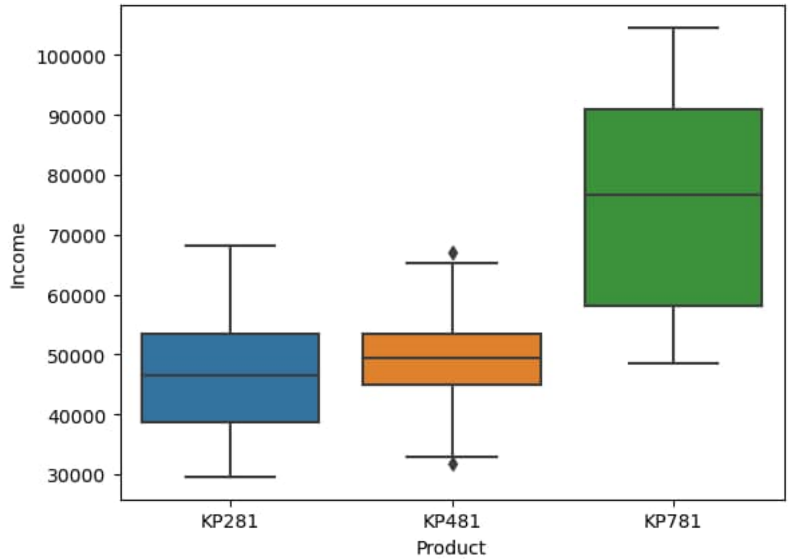
If customer plans to use the treadmill more than 3 times a week, they should go for "High-level" type else low-level and mid-level would be fine.

# Relationship between Income and Product type

In [219]:
```
df.groupby("Product")["Income"].mean()
```

Out[219]:
```
Product
KP281    46418.025
KP481    48973.650
KP781    75441.575
Name: Income, dtype: float64
```

In [218]:
```
sns.boxplot(data=df,y="Income",x="Product")
plt.show()
# High-level treadmill is purchased by high income group only. Income of the customer significantly affect the demand of differen
```




In [225]:
```
min_income=df.groupby("Product")["Income"].min()
min_income
```

Out[225]:
```
Product
KP281    29562
KP481    31836
KP781    48556
Name: Income, dtype: int64
```

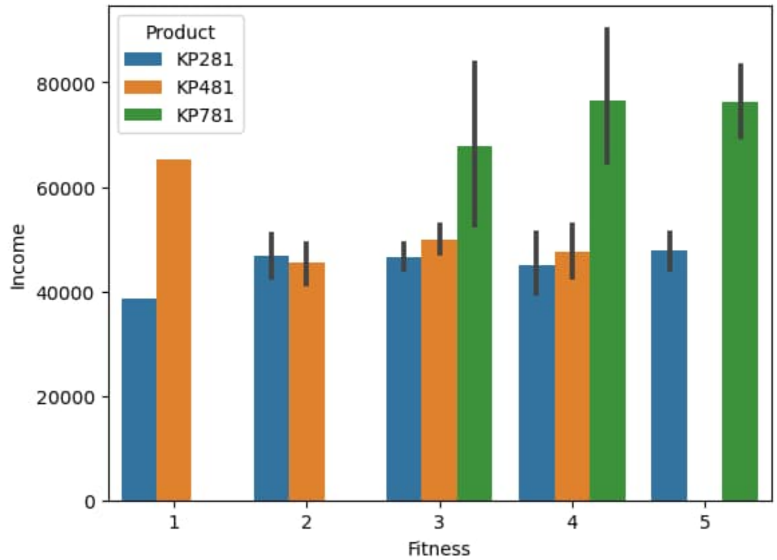
In [226]:
```
max_income=df.groupby("Product")["Income"].max()
max_income
```

Out[226]:
```
Product
KP281     68220
KP481     67083
KP781    104581
Name: Income, dtype: int64
```

In [227]:
```
range_income=max_income-min_income
range_income
# All the high income group people range between 50K and above prefer high-level treadmill while customers having income between
```

Out[227]:
```
Product
KP281    38658
KP481    35247
KP781    56025
Name: Income, dtype: int64
```

In [229]:
```
sns.barplot(data=df,x="Fitness",y="Income",hue="Product")
plt.show()
# there is a striking point we came across here that less fit people with good income prefer "Mid-level" treadmill and if custome
```




### Summary: Customer Profile related to Income for each Product:

1. "High-level" treadmill is purchased by high income group of people range between 50K and above only.
2. Income of the customer significantly affect the demand for different type of treadmill as income increases level of treadmill demanded by customers increases
3. Customers having income between 30K to 67K purchases "low-level" or "mid-level" treadmill
4. Less fit people with good income prefer "Mid-level" treadmill and if customers who are in good shape prefer "low-level" treadmill

# Relationship between fitness and product type

In [237]:
```
df.groupby("Product")["Fitness"].value_counts()
```

Out[237]:
```
Product  Fitness
KP281    3          54
         2          14
         4           9
         5           2
         1           1
KP481    3          39
         2          12
         4           8
         1           1
KP781    5          29
         4           7
         3           4
Name: Fitness, dtype: int64
```

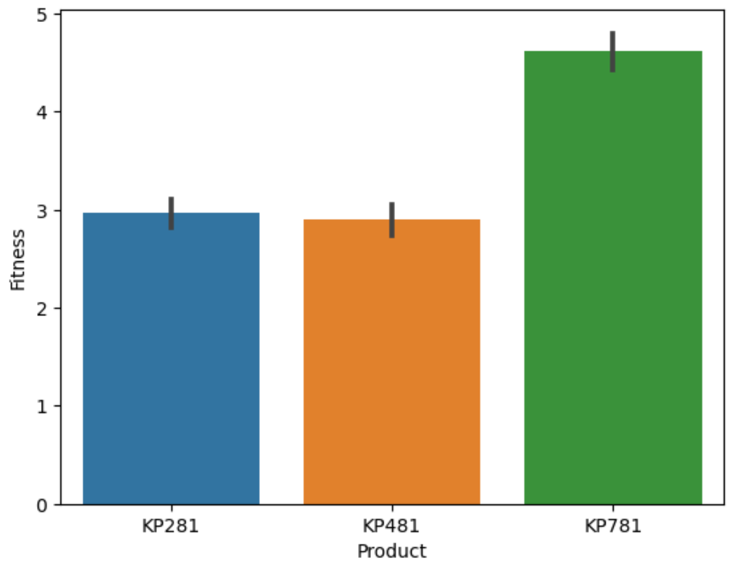
In [238]:
```
df.groupby("Product")["Fitness"].mean()
# "High-level" treadmill is associated with good fitness
```

Out[238]:
```
Product
KP281    2.9625
KP481    2.9000
KP781    4.6250
Name: Fitness, dtype: float64
```

In [239]:
```
df.groupby("Product")["Fitness"].median()
```

Out[239]:
```
Product
KP281    3.0
KP481    3.0
KP781    5.0
Name: Fitness, dtype: float64
```

In [242]:
```
sns.barplot(data=df,x="Product",y="Fitness")
plt.show()
# on an average High-level type of treadmill has 5 scale fitness
```

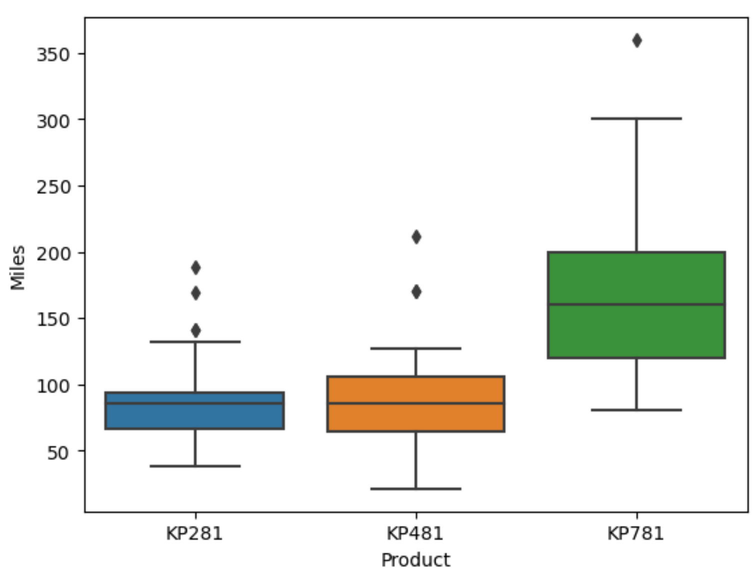



### Summary: Customer Profile related to Fitness for each Product:

1. "High-level" treadmill is associated with good fitness
2. On an average, customers who are using "Low-level" or "Mid-level" are rated between 2-3 on fitness scale of 5

## Relationship between Miles and Product type

In [259]:
```
sns.boxplot(data=df,x="Product",y="Miles")
plt.show()
```




In [257]:
```
df.groupby("Product")["Miles"].mean()
# conclusion: The average number of miles the customer expects to walk/run each week is more with "High-level" type of treadmill
```

Out[257]:
```
Product
KP281     82.787500
KP481     87.933333
KP781    166.900000
Name: Miles, dtype: float64
```

In [258]:
```
df.groupby("Product")["Miles"].median()
```

Out[258]:
```
Product
KP281     85.0
KP481     85.0
KP781    160.0
Name: Miles, dtype: float64
```

In [260]:
```
min_miles=df.groupby("Product")["Miles"].min()
min_miles
```

Out[260]:
```
Product
KP281    38
KP481    21
KP781    80
Name: Miles, dtype: int64
```

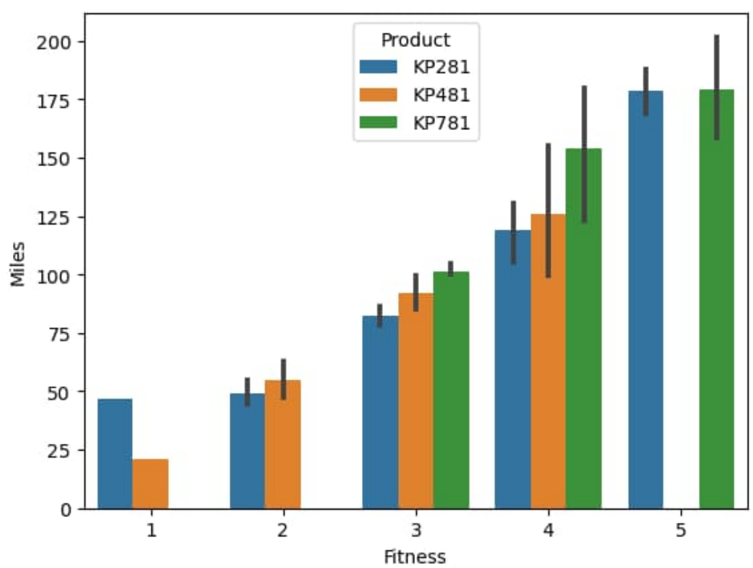
In [261]:
```
max_miles=df.groupby("Product")["Miles"].max()
max_miles
```

Out[261]:
```
Product
KP281    188
KP481    212
KP781    360
Name: Miles, dtype: int64
```

In [262]:
```
range_miles=max_miles-min_miles
range_miles
# conclusion: Better the type of treadmill, more miles are expected to run/walk each week
```

Out[262]:
```
Product
KP281    150
KP481    191
KP781    280
Name: Miles, dtype: int64
```

In [264]:
```
sns.barplot(data=df,x="Fitness",y="Miles",hue="Product")
plt.show()
# conclusion: for better fitness, more miles are expected to run/walk each week no matter which type of treadmill customer choose
```




### Summary: Customer Profile related to Miles for each Product:

1. Better the type of treadmill, more miles are expected to run/walk each week
2. The average number of miles the customer expects to walk/run each week is more with "High-level" type of treadmill

In [ ]:

## Conclusion and final insight to business for customer profile for each type of treadmill:

Target customers for "Low-Level" and "Mid-Level" type of Treadmill:

1. Customers between 25-50 years where focus should be majorly on age group of 25-35.
2. No significant difference between male and female purchasing treadmill
3. Females are most likely prefer "Low-level" type treadmill
4. Male customers above 35 years of age are more likely to purchase treadmill as compared to female customers
5. Customers who plan to use treadmill on an average 3 times a week
6. As income increase preference for better level of treadmill is demanded
7. Partnered population is most likely to purchase treadmill than Single
8. Less number of miles the customer expects to walk/run each week

Target customers for High-level of treadmill:

1. All customers above age of 35
2. Males are most likely to purchase advanced feature treadmill
3. Customers with higher education
4. Partnered population is most likely to purchase treadmill than Single
5. Customers who plan to use treadmill more than 3-4 times a week
6. High Income group people which ranges between 50K and above
7. If customer wants to be in good shape or Fitness is goal
8. Large number of miles the customer expects to walk/run each week

In [ ]:
```

```