In [ ]:

```
Problem Statement - To analyze the AeroFit dataset and come up with insights and recommenda

Objective - The purpose of this analysis is to create a customer profile for each AeroFit t

Insights:
1. Cell numbers 4,5,6,7 shows the shape of data, data types, etc...

2. Cell number 8 and 9 shows the Value counts and unique attributes.

3. Cell 14:

# This Boxplot shows:

# 1. 'KP281' is bought by both Male and Female equally and their median ages are 26.

# 2. 'KP481' is bought by both Male and Female equally. Male median age is 25 and Female me

# 3. 'KP781' is bought by Male predominantly. Male median age is 24.

4. Cell 15:

# This Boxplot shows:

# 1. 'KP281' : Male have 5 miles more as their median than Female.

# 2. 'KP481' : Male have 5 miles more as their median than Female.

# 3. 'KP781' : Male have 130 miles as their median.

5.Cell 16:

# This Boxplot shows:

# 1. 'KP281' : Mostly used by Fitness 3 and 4.

# 2. 'KP481' : Mostly used by Fitness 2,3 and 4. Those who have fitness 2 tend to go with K

# 3. 'KP781' : Mostly used by Fitness 4 and 5.

6. Cell 17:

# This Boxplot shows:

# 1. 'KP281' : Median salary 47000. People with median salary 47000 tend to prefer KP281.

# 2. 'KP481' : Median salary 50000. People with median salary 50000 tend to prefer KP481.

# 3. 'KP781' : Median salary 55000. People with median salary > 55000 tend to prefer KP781.

7. Cell 18:

# Boxplot shows:

# KP781 is majorly purchased by people who have more than 15 years of education.

8. Cell 20:
```

```
# KP281: is majorly bought by Male and Partnered people.

9. Cell 22:

# KP481: is majorly bought by Male and Partnered people.

10. Cell 24:

# KP781: is majorly bought by Male and Single people.

11. Cell 25:

************************************************

Parameters for KP281

------------------------------------------------

For KP281 Mean Age : 28.09090909090909

For KP281 Median Age : 26.0

For KP281 Variance Age : 43.188995215310996

For KP281 Mean Education : 15.0

For KP281 Median Education : 15.0

For KP281 Variance Education : 1.5

For KP281 Mean Income : 46070.64935064935

For KP281 Median Income : 46617.0

For KP281 Variance Income : 79560079.33595353

For KP281 Mean Miles : 81.49350649350649

For KP281 Median Miles : 85.0

For KP281 Variance Miles : 713.9111414900887

For KP281 Mean Usage : 3.051948051948052

For KP281 Median Usage : 3.0

For KP281 Variance Usage : 0.5762132604237868

For KP281 Mean Fitness : 2.935064935064935

For KP281 Median Fitness : 3.0

For KP281 Variance Fitness : 0.40362269309637755

************************************************

Parameters for KP481

------------------------------------------------

For KP481 Mean Age : 28.70689655172414
```

```
For KP481 Median Age : 26.0

For KP481 Variance Age : 38.17574107683001

For KP481 Mean Education : 15.120689655172415

For KP481 Median Education : 16.0

For KP481 Variance Education : 1.5114942528735626

For KP481 Mean Income : 49074.51724137931

For KP481 Median Income : 49459.5

For KP481 Variance Income : 72207749.72776766

For KP481 Mean Miles : 86.20689655172414

For KP481 Median Miles : 85.0

For KP481 Variance Miles : 862.1318814277068

For KP481 Mean Usage : 3.0517241379310347

For KP481 Median Usage : 3.0

For KP481 Variance Usage : 0.5762250453720505

For KP481 Mean Fitness : 2.8793103448275863

For KP481 Median Fitness : 3.0

For KP481 Variance Fitness : 0.38868723532970395

************************************************

Parameters for KP781

------------------------------------------------

For KP781 Mean Age : 24.533333333333335

For KP781 Median Age : 24.0

For KP781 Variance Age : 3.980952380952381

For KP781 Mean Education : 16.666666666666668

For KP781 Median Education : 16.0

For KP781 Variance Education : 1.5238095238095242

For KP781 Mean Income : 58273.73333333333

For KP781 Median Income : 57271.0

For KP781 Variance Income : 73419803.06666668

For KP781 Mean Miles : 133.73333333333332
```

```
For KP781 Median Miles : 120.0

For KP781 Variance Miles : 1326.2095238095237

For KP781 Mean Usage : 4.533333333333333

For KP781 Median Usage : 4.0

For KP781 Variance Usage : 0.9809523809523811

For KP781 Mean Fitness : 4.466666666666667

For KP781 Median Fitness : 5.0

For KP781 Variance Fitness : 0.5523809523809525

12. Cell 26:

# Count plot shows:

# 1. 'KP281' : Mostly purchased by 14 years and 16 years of experience people.

# 2. 'KP481' : Mostly purchased by 14 years and 16 years of experience people.

# 3. 'KP781' : Mostly purchased by 16 years and 18 years of experience people.

13. Cell 27:

# Count plot shows:

# 1. 'KP281' : Mostly purchased by Fitness 2,3 and 4.

# 2. 'KP481' : Mostly purchased by Fitness 2,3 and 4.

# 3. 'KP781' : Mostly purchased by Fitness 5,4.

24. Cell 28:

# Count plot shows:

# 1. 'KP281' : Mostly purchased by Usage 2,3 and 4.

# 2. 'KP481' : Mostly purchased by Usage 2,3 and 4..

# 3. 'KP781' : Mostly purchased by Usage 5,4

25. Cell 29:

# Count plot shows:

# 1. 'KP281' : Mostly purchased by Male and Female.

# 2. 'KP481' : Mostly purchased by Male and Female.

# 3. 'KP781' : Mostly purchased by Male only.

26. Cell 30:

# Count plot shows:

# 1. 'KP281' : Mostly purchased more by Partnered.
```

```
# 2. 'KP481' : Mostly purchased more by Partnered.

# 3. 'KP781' : Mostly purchased by single and partnered equally.

27. Cell 31:

# Hist plot shows: Male are more inclined to fitness than Female.

28. Cell 32:

# Female age between 33-35 buy more treadmill than male.

# Male age between 22 to 26 buy more treadmill than female.

29. Cell 33 :

# Whether customer is Single or Partnered , Male tend to buy treadmill.

30. Cell 34,35,36:

# Treadmill is bought more by age group people between 23-30 years.

# Treadmill is bought more by people with income between 35000-60000.

# Treadmill is more used for miles between 50-75 mostly.

31. Cell 37:

# Box plot shows:

# 1. People with education 12 buy KP281.

# 2. People with education 13 buy KP281 age between 20-25 and buy KP481 age between 20-35.

# 3. People with education 14 buy KP281 and KP481.

# 4. People with education 15 buy KP281.

# 5. People with education 16 buy KP281 and KP481. Age bwtween 22-26 buy KP781.

# 6. People with education 18 buy KP781 age bwetween 22-28. Age above 28 people prefer KP28

32. Cell 38:

# Table below shows the correlation.

# Age and Income have a positive correlation.

# Education and Income have a positive correlation.

# Usage and Miles have a positive correlation.

# Fitness and Miles have a positive correlation.

33. Cell 39,40:

# Heatmap below shows the value of correlation of all parameters.

# Below is the Pair plot for all the numerical columns.
```

34. Cell 41:

```
# Below heat map shows:

# 1. KP281 :  People having Education years 14 and 16 buy more.

# 2. KP481 :  People having Education years 14 and 16 buy more.
```

35. Cell 42:

```
# Very few female buy KP781.

# KP281 is more purchased by both Male and Female.
```

36. Cell 43:

```
# Below heat map shows:

# 1. KP281: Has a people usage of 2,3 and 4.

# 2. KP481: Has people usage of 2,3,4.

# 3. KP781: Has people usage of 4 and 5.
```

37. Cell 44,45:

```
# Below is the heat map of of product and Fitness.

# Below is the heat map of of product and Martial Status.
```

38. Cell 46:

```
# Below crosstab  gives the count of Male and Female in each product.
```

39. Cell 47:

```
# 51.33% people buy KP281, 38.667% people buy KP481 and 1% people buy KP781.
```

40. Cell 49,50:

```
# 56.71% female buy KP281, 41.17% female buy KP481 and 1.49% female buy KP781.

# 46.988% male buy KP281, 36.1446% male buy KP481 and 16.86% male buy KP781.

# Probability of Female and Male given they buy KP281 is 0.493 and 0.506 respectively .

# Probability of Female and Male given they buy KP481 is 0.4827 and 0.517 respectively .

# Probability of Female and Male given they buy KP781 is 0.066 and 00.9333 respectively.
```

41. Cell 52,53:

```
# 66.66 % of 12 years education buy KP281, 33.33% of 12 years education buy KP481.

# 60 % of 13 years education buy KP281, 40% of 13 years education buy KP481.

# 56.60 % of 14 years education buy KP281,of 41.5% 14 years education buy KP481.

# 80 % of 15 years education buy KP281, 20% of 15 years education buy KP481.
```

```
# 48.64 % of 16 years education buy KP281, 40.54% of 16 years education buy KP481 and 10.81
# 20 % of 18 years education buy KP281, 20%  of18 years education buy KP481 and 60%  of 18
# Probability of education years 11,12,13,14,15,16 and 18 given they buy KP281 is 0.025974,
# Probability of education years 11,12,13,14,15,16 and 18 given they buy KP481 is 0.017241,
# Probability of education years 11,12,13,14,15,16 and 18 given they buy KP781 is 0.000000,
```

42. Cell 55,56:

```
# 52.32 % Partnered buy KP281 and 50% Single buy KP281.
# 39.53 % Partnered buy KP481 and 37.5% Single buy KP481.
# 8.1395 % Partnered buy KP781 and 12.5% Single buy KP781.
# Probability of Partnered, single given that they buy KP281 is 0.584416 and 0.415584 respe
# Probability of Partnered, single given that they buy KP481 is 0.586207 and 0.41379 respec
# Probability of Partnered, single given that they buy KP781 is 0.466667 and 0.5333 respect
```

43. Cell 58,59:

```
# 52.32 % Partnered buy KP281 and 50% Single buy KP481.
# 39.53 % Partnered buy KP481 and 37.5% Single buy KP481.
# 8.1395 % Partnered buy KP781 and 12.5% Single buy KP781.
# Probability of Fitness 1,2,3,4 and 5 given that they buy KP281 is 0.012987,0.181818,0.675
# Probability of Fitness 1,2,3,4 and 5 given that they buy KP481 is 0.017241,0.206897,0.655
# Probability of Fitness 1,2,3,4 and 5 given that they buy KP781 is 0.000000,0.000000,0.133
```

44. Cell 61,62:

```
# 59.275 % usage 2 buy KP281 and 40.625% usage 2 buy KP481.
# 52.94 % usage 3 buy KP281 and 45.588% usage 3 buy KP481.
# 51.21 % usage 4 buy KP281 and 29.26% usage 4 buy KP481 and 19.512% usage 4 buy KP781.
# 14.285 % usage 5 buy KP281 and 28.57% usage 5 buy KP481 and 57.142% usage 5 buy KP781.
# 0 % usage 6 buy KP281 and 0% usage 6 buy KP481 and 100% usage 6 buy KP781.
# 0 % usage 7 buy KP281 and 0% usage 7 buy KP481 and 100% usage 7 buy KP781.
# Probability of Usage 2,3,4 and 5 given that they buy KP281 is 0.246753,0.467532,0.272727
# Probability of Usage 2,3,4 and 5 given that they buy KP481 is 0.224138,0.534483,0.206897
# Probability of Usage 3,4,5,6 and 7 given that they buy KP781 is 0.066667,0.533333,0.26666
```

45. Cell 66:

```
# Below plot shows that KP281 is purchased by people in age group between 20 to 40.

# Below plot shows that KP481 is purchased by people in age group between 20 to 40.

# Below plot shows that KP781 is purchased by people in age group between 20 to 30.
```

46. Cell 67:

```
# Below plot shows that KP281 is purchased by people in income group between 30000 to 60000

# Below plot shows that KP481 is purchased by people in age group between 40000 to 60000.

# Below plot shows that KP781 is purchased by people in age group between 40000 to 70000 mo
```

47. Cell 68:

```
# Below plot shows that KP281 is purchased by people travel miles group between 60 to 100 a

# Below plot shows that KP481 is purchased by people travel miles group between 80 to 100 a

# Below plot shows that KP781 is purchased by people travel miles group between 140 to 180
```

48. Cell 70,71:

```
# 20< Age < 30 : 51.06% of this age group buy KP281, 32.978% of this age group buy KP481 an

# 30 <Age < 40 : 47.619% of this age group buy KP281 and  52.381% of this age group buy KP4

# 40 <Age < 50 : 55.55% of this age group buy KP281, 44.44% of this age group buy KP481 .

# Age < 20 : 80% of this age group buy KP281, 20% of this age group buy KP481.

# Probability of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy K

# Probability of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy K

# Probability of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy K
```

49. Cell 73,74:

```
# 30000< Income < 40000 : 73.33 % of this income buy KP281 and 26.667  % of this income buy

# 40000 < Income < 50000 : 48.97 % of this income buy KP281, 42.857  % of this income buy K

# 50000 < Income < 60000 : 48.076 % of this income buy KP281, 42.3077  % of this income buy

# 60000< Income < 700000 :31.25 % of this income buy KP281, 43.75  % of this income buy KP4

# 70000< Income < 800000 :0 % of this income buy KP281, 0  % of this income buy KP481 and 1

# Income < 20000 :100 % of this income buy KP281.

# Probability of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 6000

# Probability of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 6000

# Probability of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 6000
```

50. Cell 76,77:

```
# 100 < Miles < 120 : 48 % of this mile range buy KP281,32  % of this mile range buy KP481
```

```
# 120< Miles < 140 : 22.222 % of this mile range buy KP281,55.556  % of this mile range buy
# 140< Miles < 160 : 66.667 % of this mile range buy KP281,0  % of this mile range buy KP48
# 160< Miles < 180 : 11.111 % of this mile range buy KP281,22.22  % of this mile range buy
# 60< Miles < 80 : 70.37 % of this mile range buy KP281,29.62  % of this mile range buy KP4
# 80 < Miles < 100 : 48.936 % of this mile range buy KP281,48.936  % of this mile range buy
# Miles < 60: 60 % of this mile range buy KP281,40  % of this mile range buy KP481
# Below table shows the conditional probability of product and Miles.
51. Cell 78,79,80:
# For KP281:
# Max purchase is done by people between age group 20 to 30.
# Max purchase is done by people between income group 50000 to 60000.
# Max purchase is done by people between miles group 80 to 100.
# For KP481:
# Max purchase is done by people between age group 20 to 30.
# Max purchase is done by people between income group 50000 to 60000.
# Max purchase is done by people between miles group 80 to 100.
# For KP781:
# Max purchase is done by people between age group 20 to 30.
# Max purchase is done by people between income group 50000 to 60000.
# Max purchase is done by people between miles group 160 to 180.
52. Cell 81-84:
# Below is the heat map between the age range and product.
# Below is the heat map between the income range and product.
# Below is the heat map between the age range and income range.
# Below is the heat map between the miles range and product.
```

In [1]:

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

In [2]:

```
df=pd.read_csv('Aerofit_treadmill.csv')
```

In [3]:

```
df
```

Out[3]:

	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

In [4]:

```
df.info()
# Data type all the columns are checked
```

```
<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
```

In [5]:

```
df.isnull().sum()
# No null values found
```

Out[5]:

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

In [6]:

```
df.describe()
```

Out[6]:

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	5.000000	104581.000000	360.000000

In [7]:

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

Out[7]:

	Product	Gender	MaritalStatus
count	180	180	180
unique	3	2	2
top	KP281	Male	Partnered
freq	80	104	107

In [8]:

```
df.nunique()
```

Out[8]:

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

In [9]:

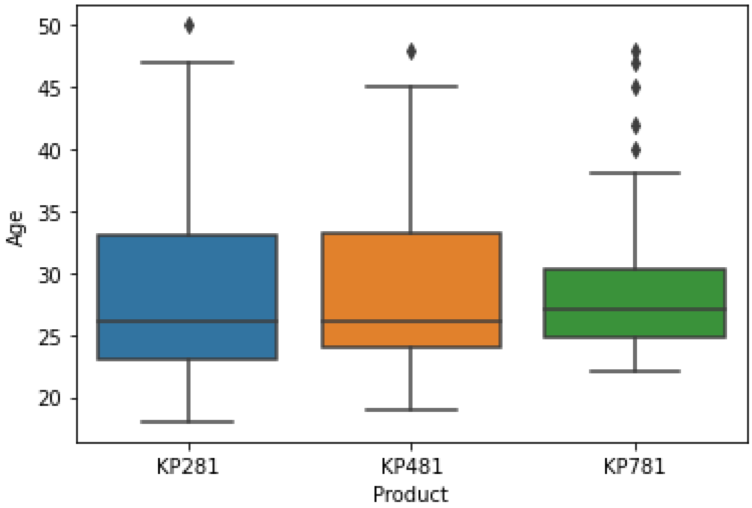
```
df['Product'].value_counts(),df['MaritalStatus'].value_counts(),df['Usage'].value_counts(),
```

Out[9]:

```
(KP281    80
 KP481    60
 KP781    40
 Name: Product, dtype: int64,
 Partnered    107
 Single        73
 Name: MaritalStatus, dtype: int64,
 3    69
 4    52
 2    33
 5    17
 6     7
 7     2
 Name: Usage, dtype: int64,
 3    97
 5    31
 2    26
 4    24
 1     2
 Name: Fitness, dtype: int64)
```

In [10]:

```
sns.boxplot(x=df['Product'],y=df['Age'])
plt.show()
# Boxplot shoes to check the outliners visually
```



In [11]:

```
def outliners(x,col):
    Q1 = np.percentile(x[col], 25)
    Q3 = np.percentile(x[col], 75)
    IQR = Q3 - Q1
    upper = Q3 +1.5*IQR
    lower = Q1 - 1.5*IQR
    #print(upper,lower)
    ls=list(x.iloc[((x[col]<lower) | (x[col]>upper)).values].index)
    return ls
```

In [12]:

```
newlist=[]
for i in ['Age','Education','Income','Miles']:
    ls=outliners(df,i)
    newlist.extend(ls)
newlist=list(set(newlist))
df.drop(newlist,axis=0,inplace=True)
# Function in cell 11 and cell 12 removes all the outliners in 'Age','Education','Income','
```

In [13]:

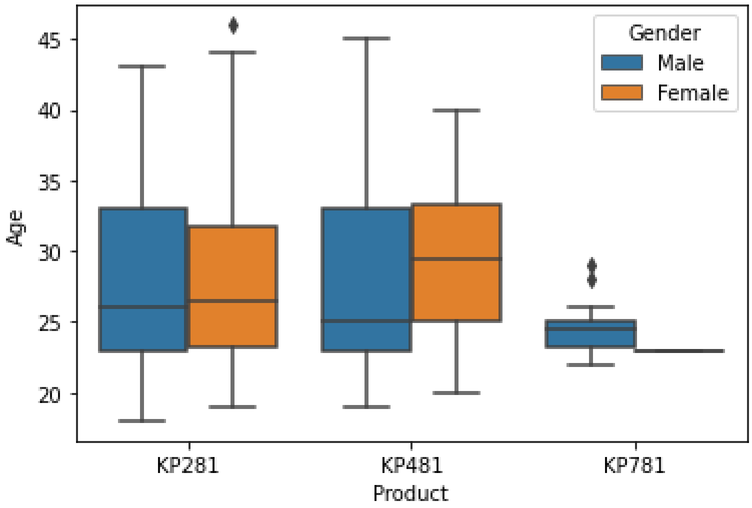
```
df.shape
# Shape of the data is 150 rows and 9 columns
```

Out[13]:

```
(150, 9)
```

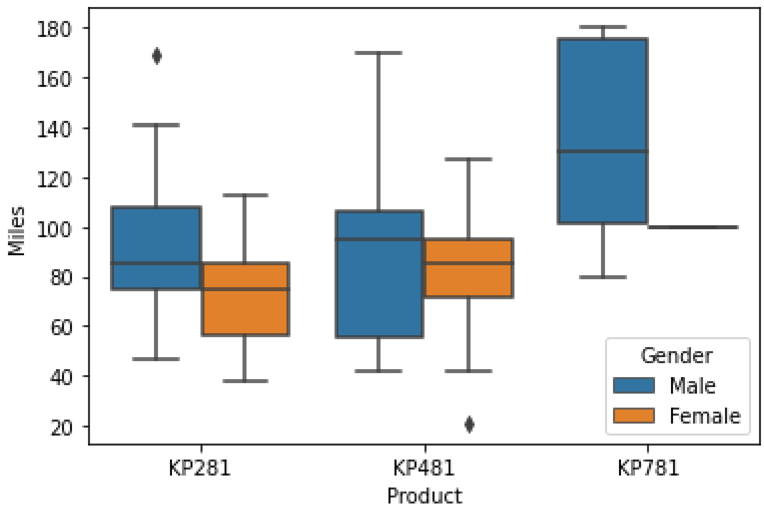
In [14]:

```
sns.boxplot(x=df['Product'],y=df['Age'],hue=df['Gender'])
plt.show()
# This Boxplot shows:
# 1. 'KP281' is bought by both Male and Female equally and their median ages are 26
# 2. 'KP481' is bought by both Male and Female equally. Male median age is 25 and Female me
# 2. 'KP781' is bought by Male predominantly. Male median age is 24
```



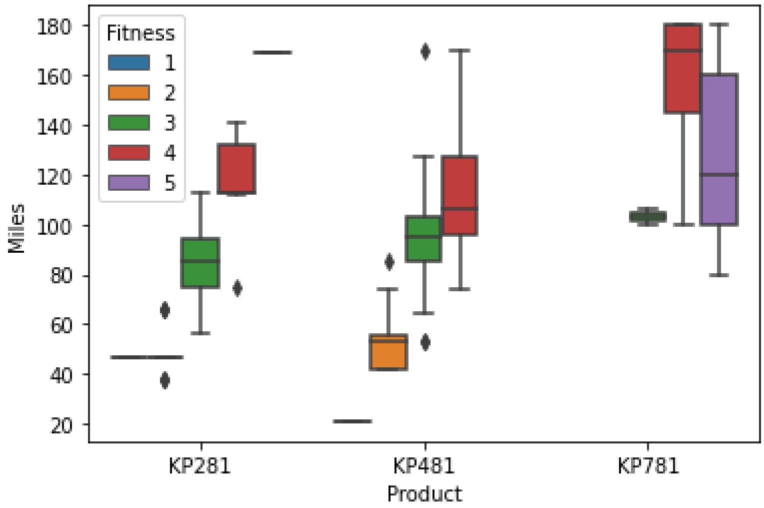
In [15]:

```
sns.boxplot(x=df['Product'],y=df['Miles'],hue=df['Gender'])
plt.show()
# This Boxplot shows:
# 1. 'KP281' : Male have 5 miles more as their median than Female
# 2. 'KP481' : Male have 5 miles more as their median than Female
# 2. 'KP781' : Male have 130 miles as their median
```



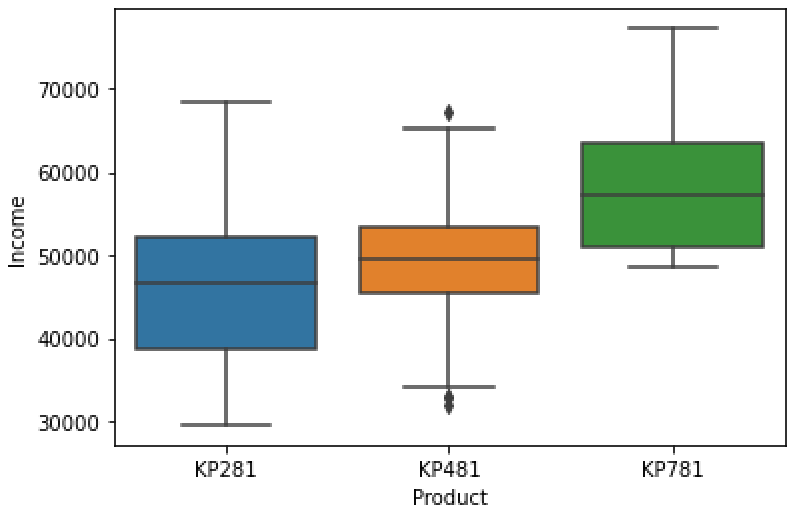
In [16]:

```
sns.boxplot(x=df['Product'],y=df['Miles'],hue=df['Fitness'])
plt.show()
# This Boxplot shows:
# 1. 'KP281' : Mostly used by Fitness 3 and 4
# 2. 'KP481' : Mostly used by Fitness 2,3 and 4. Those who have fitness 2 tend to go with K
# 2. 'KP781' : Mostly used by Fitness 4 and 5
```



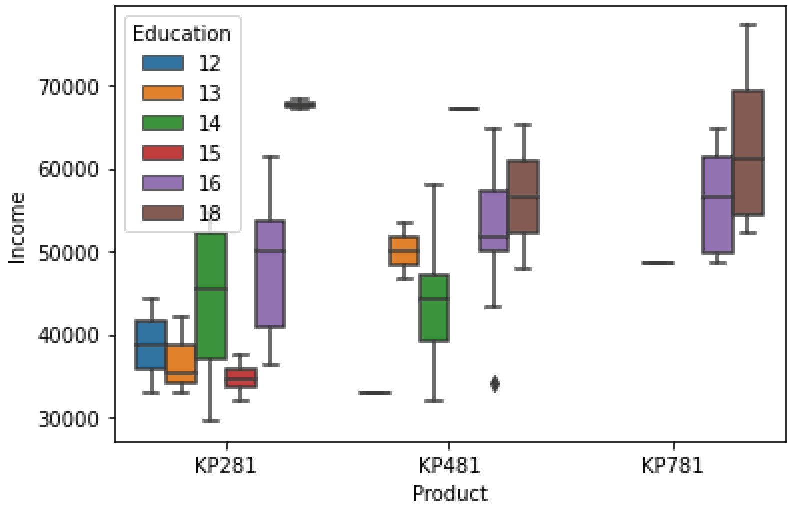
In [17]:

```
sns.boxplot(x=df['Product'],y=df['Income'])
plt.show()
# This Boxplot shows:
# 1. 'KP281' : Median salary 47000. People with median salary 47000 tend to prefer KP281
# 2. 'KP481' : Median salary 50000. People with median salary 50000 tend to prefer KP481
# 2. 'KP781' : Median salary 55000. People with median salary > 55000 tend to prefer KP781
```



In [18]:

```
sns.boxplot(x=df['Product'],y=df['Income'],hue=df['Education'])
plt.show()
# Boxplot shows:
# KP781 is majorly purchased by people who have more than 15 years of education
```



In [19]:

```
df.loc[df['Product']=='KP281'].describe()
```

Out[19]:

	Age	Education	Usage	Fitness	Income	Miles
count	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000
mean	28.090909	15.000000	3.051948	2.935065	46070.649351	81.493506
std	6.571833	1.224745	0.759087	0.635313	8919.645696	26.719116
min	18.000000	12.000000	2.000000	1.000000	29562.000000	38.000000
25%	23.000000	14.000000	3.000000	3.000000	38658.000000	66.000000
50%	26.000000	15.000000	3.000000	3.000000	46617.000000	85.000000
75%	32.000000	16.000000	4.000000	3.000000	52302.000000	94.000000
max	46.000000	18.000000	5.000000	5.000000	68220.000000	169.000000

In [20]:

```
df.loc[df['Product']=='KP281'].describe(include=object)
# KP281: is majorly bought by Male and Partnered people
```

Out[20]:

	Product	Gender	MaritalStatus
count	77	77	77
unique	1	2	2
top	KP281	Male	Partnered
freq	77	39	45

In [21]:

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

Out[21]:

	Age	Education	Usage	Fitness	Income	Miles
count	58.000000	58.000000	58.000000	58.000000	58.000000	58.000000
mean	28.706897	15.120690	3.051724	2.879310	49074.517241	86.206897
std	6.178652	1.229428	0.759095	0.623448	8497.514326	29.362082
min	19.000000	12.000000	2.000000	1.000000	31836.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	45480.000000	64.000000
50%	26.000000	16.000000	3.000000	3.000000	49459.500000	85.000000
75%	33.000000	16.000000	3.000000	3.000000	53439.000000	103.250000
max	45.000000	18.000000	5.000000	4.000000	67083.000000	170.000000

In [22]:

```
df.loc[df['Product']=='KP481'].describe(include=object)
# KP481: is majorly bought by Male and Partnered people
```

Out[22]:

	Product	Gender	MaritalStatus
count	58	58	58
unique	1	2	2
top	KP481	Male	Partnered
freq	58	30	34

In [23]:

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

Out[23]:

	Age	Education	Usage	Fitness	Income	Miles
count	15.000000	15.000000	15.000000	15.000000	15.000000	15.000000
mean	24.533333	16.666667	4.533333	4.466667	58273.733333	133.733333
std	1.995232	1.234427	0.990430	0.743223	8568.535643	36.417160
min	22.000000	14.000000	3.000000	3.000000	48556.000000	80.000000
25%	23.000000	16.000000	4.000000	4.000000	51045.500000	100.000000
50%	24.000000	16.000000	4.000000	5.000000	57271.000000	120.000000
75%	25.000000	18.000000	5.000000	5.000000	63496.000000	170.000000
max	29.000000	18.000000	7.000000	5.000000	77191.000000	180.000000

In [24]:

```
df.loc[df['Product']=='KP781'].describe(include=object)
# KP781: is majorly bought by Male and Single people
```

Out[24]:

	Product	Gender	MaritalStatus
count	15	15	15
unique	1	2	2
top	KP781	Male	Single
freq	15	14	8

In [25]:

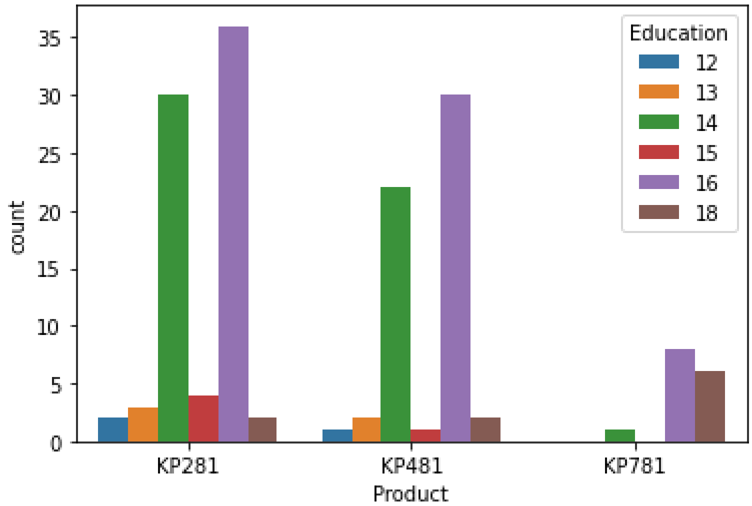
```
def parameter(deff,i,j):
    print('For',j,'Mean', i ,':',deff[i].mean())
    print('For',j,'Median', i ,':',deff[i].median())
    print('For',j,'Variance', i ,':',deff[i].var())

for j in ['KP281','KP481','KP781']:
    print('************************************************')
    print('Parameters for', j)
    print('------------------------------------------------')
    for i in ['Age','Education','Income','Miles','Usage','Fitness']:
        parameter(df.loc[df['Product']==j],i,j)
```

```
************************************************
Parameters for KP281
------------------------------------------------
For KP281 Mean Age : 28.09090909090909
For KP281 Median Age : 26.0
For KP281 Variance Age : 43.188995215310996
For KP281 Mean Education : 15.0
For KP281 Median Education : 15.0
For KP281 Variance Education : 1.5
For KP281 Mean Income : 46070.64935064935
For KP281 Median Income : 46617.0
For KP281 Variance Income : 79560079.33595353
For KP281 Mean Miles : 81.49350649350649
For KP281 Median Miles : 85.0
For KP281 Variance Miles : 713.9111414900887
For KP281 Mean Usage : 3.051948051948052
For KP281 Median Usage : 3.0
For KP281 Variance Usage : 0.5762132604237868
For KP281 Mean Fitness : 2.935064935064935
For KP281 Median Fitness : 3.0
For KP281 Variance Fitness : 0.40362269309637755
************************************************
Parameters for KP481
------------------------------------------------
For KP481 Mean Age : 28.70689655172414
For KP481 Median Age : 26.0
For KP481 Variance Age : 38.17574107683001
For KP481 Mean Education : 15.120689655172415
For KP481 Median Education : 16.0
For KP481 Variance Education : 1.5114942528735626
For KP481 Mean Income : 49074.51724137931
For KP481 Median Income : 49459.5
For KP481 Variance Income : 72207749.72776766
For KP481 Mean Miles : 86.20689655172414
For KP481 Median Miles : 85.0
For KP481 Variance Miles : 862.1318814277068
For KP481 Mean Usage : 3.0517241379310347
For KP481 Median Usage : 3.0
For KP481 Variance Usage : 0.5762250453720505
For KP481 Mean Fitness : 2.8793103448275863
For KP481 Median Fitness : 3.0
For KP481 Variance Fitness : 0.38868723532970395
************************************************
Parameters for KP781
------------------------------------------------
For KP781 Mean Age : 24.533333333333335
```

```
For KP781 Median Age : 24.0
For KP781 Variance Age : 3.980952380952381
For KP781 Mean Education : 16.666666666666668
For KP781 Median Education : 16.0
For KP781 Variance Education : 1.5238095238095242
For KP781 Mean Income : 58273.73333333333
For KP781 Median Income : 57271.0
For KP781 Variance Income : 73419803.06666668
For KP781 Mean Miles : 133.73333333333332
For KP781 Median Miles : 120.0
For KP781 Variance Miles : 1326.2095238095237
For KP781 Mean Usage : 4.533333333333333
For KP781 Median Usage : 4.0
For KP781 Variance Usage : 0.9809523809523811
For KP781 Mean Fitness : 4.466666666666667
For KP781 Median Fitness : 5.0
For KP781 Variance Fitness : 0.5523809523809525
```

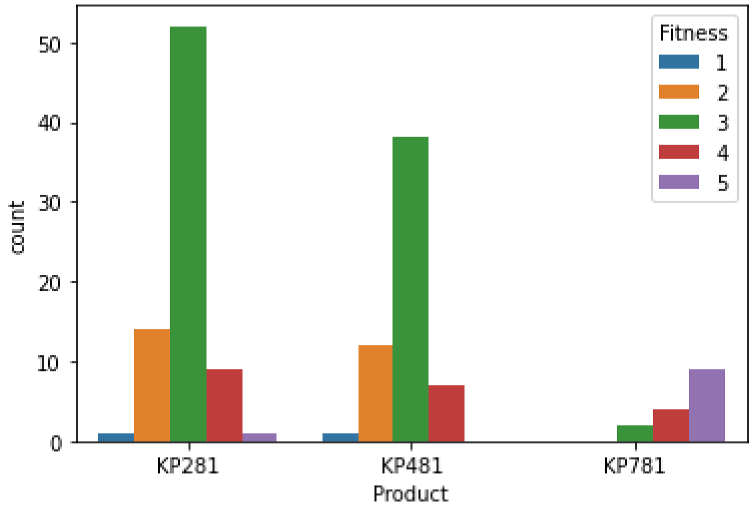
In [26]:

```
sns.countplot(x=df['Product'],hue=df['Education'])
plt.show()
# Count plot shows:
# 1. 'KP281' : Mostly purchased by 14 years and 16 years of experience people
# 2. 'KP481' : Mostly purchased by 14 years and 16 years of experience people
# 3. 'KP781' : Mostly purchased by 16 years and 18 years of experience people
```



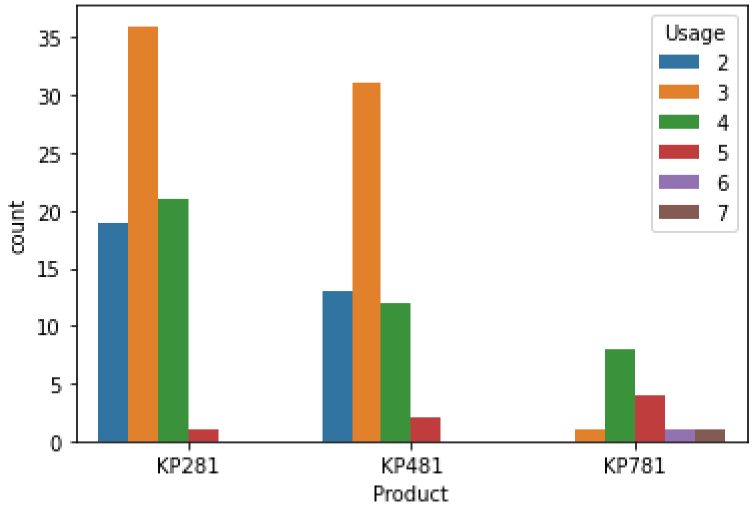
In [27]:

```
sns.countplot(x=df['Product'],hue=df['Fitness'])
plt.show()
# Count plot shows:
# 1. 'KP281' : Mostly purchased by Fitness 2,3 and 4
# 2. 'KP481' : Mostly purchased by Fitness 2,3 and 4
# 3. 'KP781' : Mostly purchased by Fitness 5,4
```



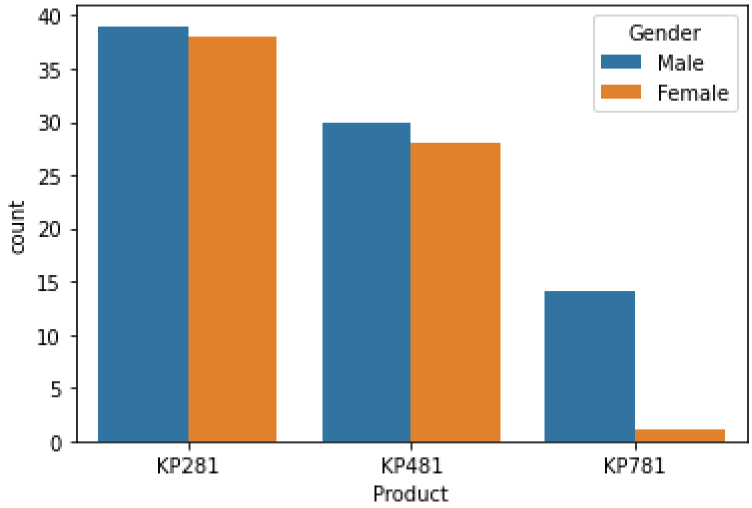
In [28]:

```
sns.countplot(x=df['Product'],hue=df['Usage'])
plt.show()
# Count plot shows:
# 1. 'KP281' : Mostly purchased by Usage 2,3 and 4
# 2. 'KP481' : Mostly purchased by Usage 2,3 and 4
# 3. 'KP781' : Mostly purchased by Usage 5,4
```



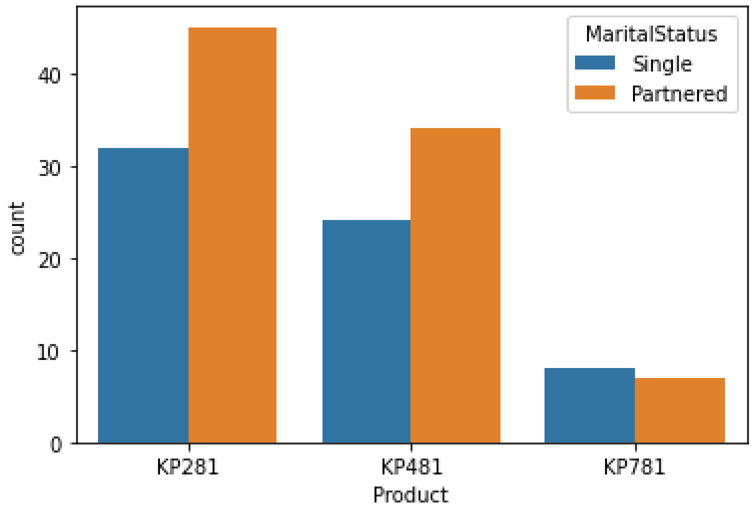
In [29]:

```
sns.countplot(x=df['Product'],hue=df['Gender'])
plt.show()
# Count plot shows:
# 1. 'KP281' : Mostly purchased by Male an Female
# 2. 'KP481' : Mostly purchased by Male an Female
# 3. 'KP781' : Mostly purchased by Male only
```



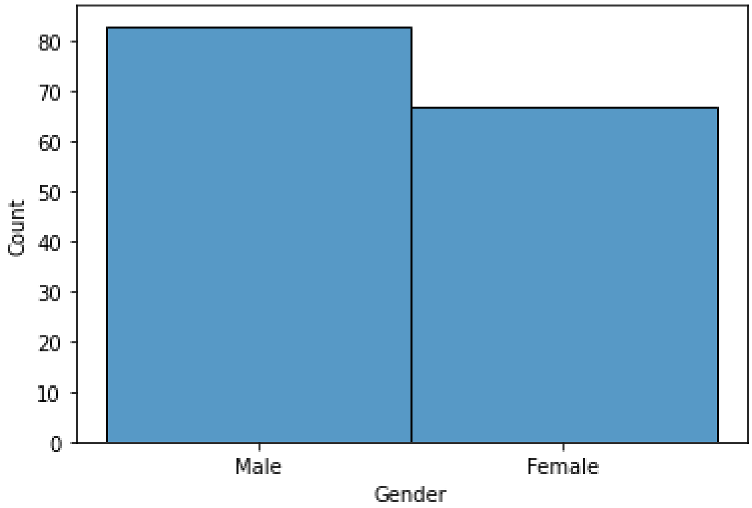
In [30]:

```
sns.countplot(x=df['Product'],hue=df['MaritalStatus'])
plt.show()
# Count plot shows:
# 1. 'KP281' : Mostly purchased more by Partnered
# 2. 'KP481' : Mostly purchased more by Partnered
# 3. 'KP781' : Mostly purchased by single and partnered equally
```



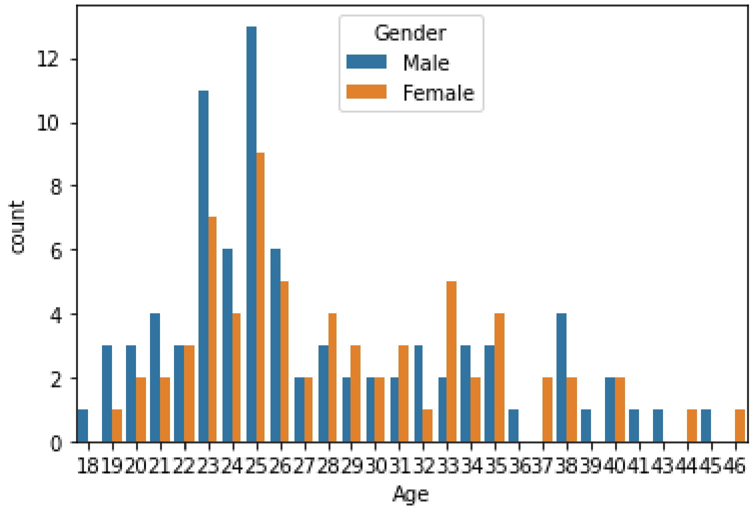
In [31]:

```
sns.histplot(x=df['Gender'])
plt.show()
# Hist plot shows: Male are more inclined to fitness than Female
```



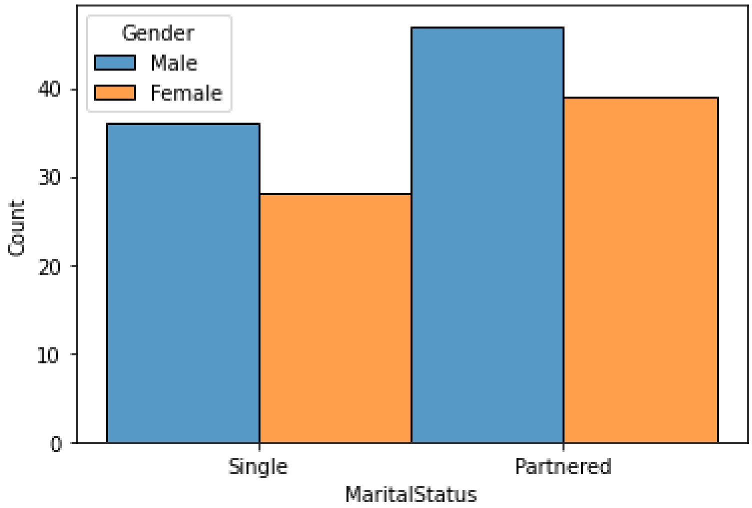
In [32]:

```
sns.countplot(x=df['Age'],hue=df['Gender'])
plt.show()
# Female age between 33-35 buy more treadmill than male
# Male age between 22 to 26 buy more treadmill than female
```



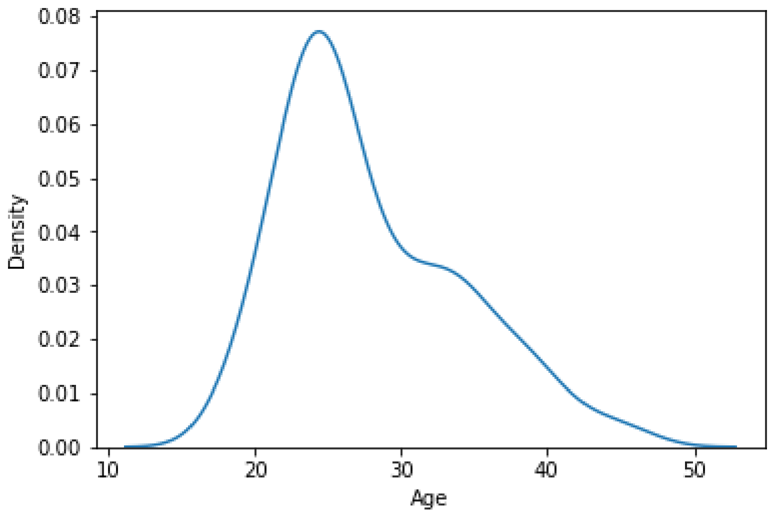
In [33]:

```
sns.histplot(x=df['MaritalStatus'],hue=df['Gender'],multiple='dodge')
plt.show()
# Whether customer is Single or Partnered , Male tend to buy treadmill
```



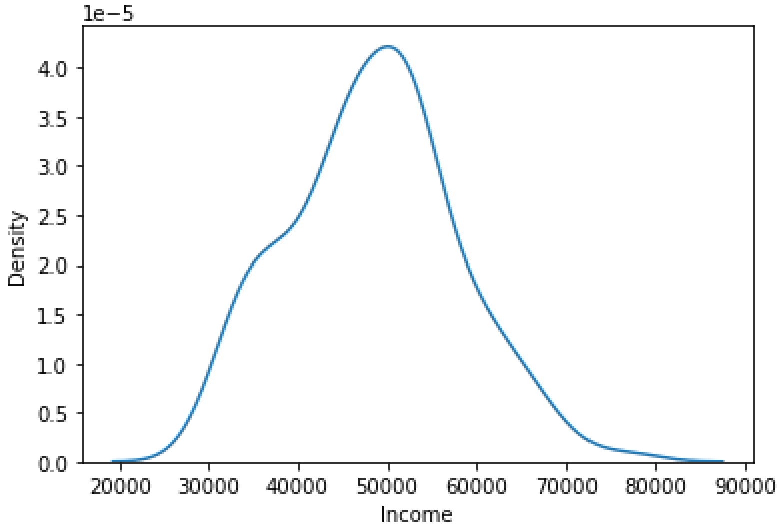
In [34]:

```
sns.kdeplot(x=df['Age'])
plt.show()
# Treadmill is bought more by age group people between 23-30 years
```



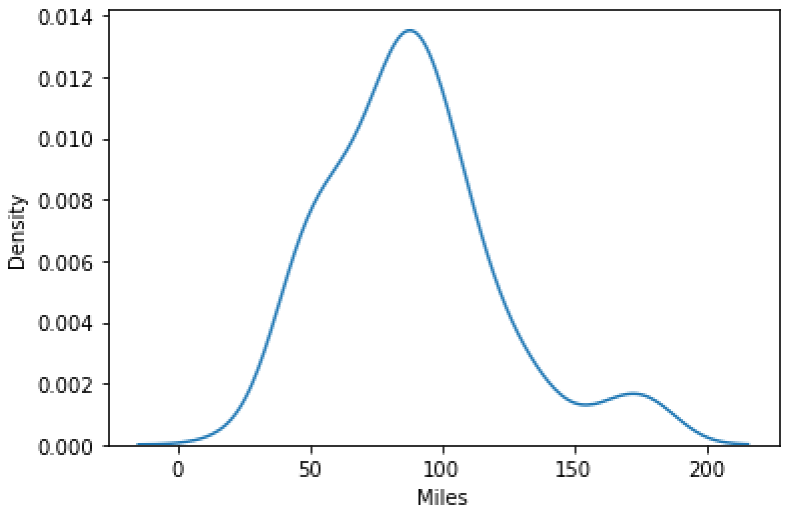
In [35]:

```
sns.kdeplot(x=df['Income'])
plt.show()
# Treadmill is bought more by people with income between 35000-60000
```



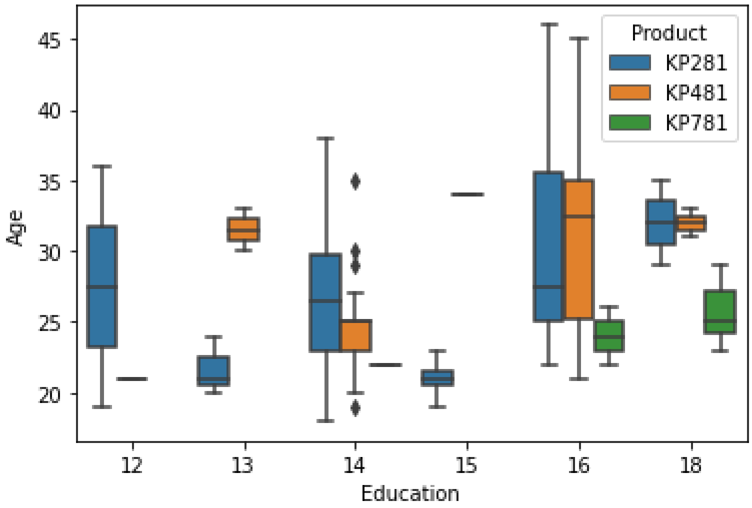
In [36]:

```
sns.kdeplot(x=df['Miles'])
plt.show()
# Treadmill is more used for miles between 50-75 mostly
```



In [37]:

```
sns.boxplot(x=df['Education'],y=df['Age'],hue=df['Product'])
plt.show()
# Box plot shows:
# 1. People with education 12 buy KP281
# 2. People with education 13 buy KP281 age between 20-25 and buy KP481 age between 20-35
# 3. People with education 14 buy KP281 and KP481
# 4. People with education 15 buy KP281
# 5. People with education 16 buy KP281 and KP481. Age bwtween 22-26 buy KP781
# 6. People with education 18 buy KP781 age bwetween 22-28. Age above 28 people prefer KP28
```



In [38]:

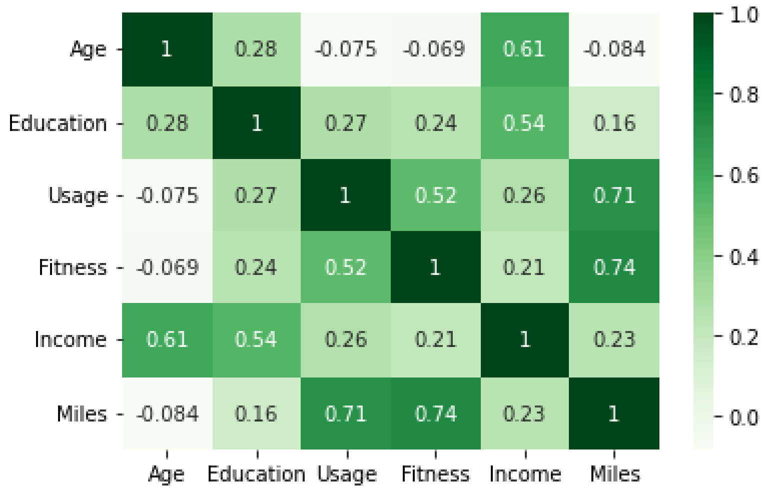
```
df.corr()
# Table below shows the correlation
# Age and Income have a positive correlation
# Education and Income have a positive correlation
# Usage and Miles have a positive correlation
# Fitness and Miles have a positive correlation
```

Out[38]:

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.277640	-0.075051	-0.069411	0.609795	-0.083593
Education	0.277640	1.000000	0.270959	0.237959	0.543339	0.157304
Usage	-0.075051	0.270959	1.000000	0.519701	0.259510	0.705753
Fitness	-0.069411	0.237959	0.519701	1.000000	0.207778	0.737187
Income	0.609795	0.543339	0.259510	0.207778	1.000000	0.233909
Miles	-0.083593	0.157304	0.705753	0.737187	0.233909	1.000000

In [39]:

```
sns.heatmap(df.corr(),cmap='Greens',annot=True)
plt.show()
# Heatmap below shows the value of correlation of all parameters
```

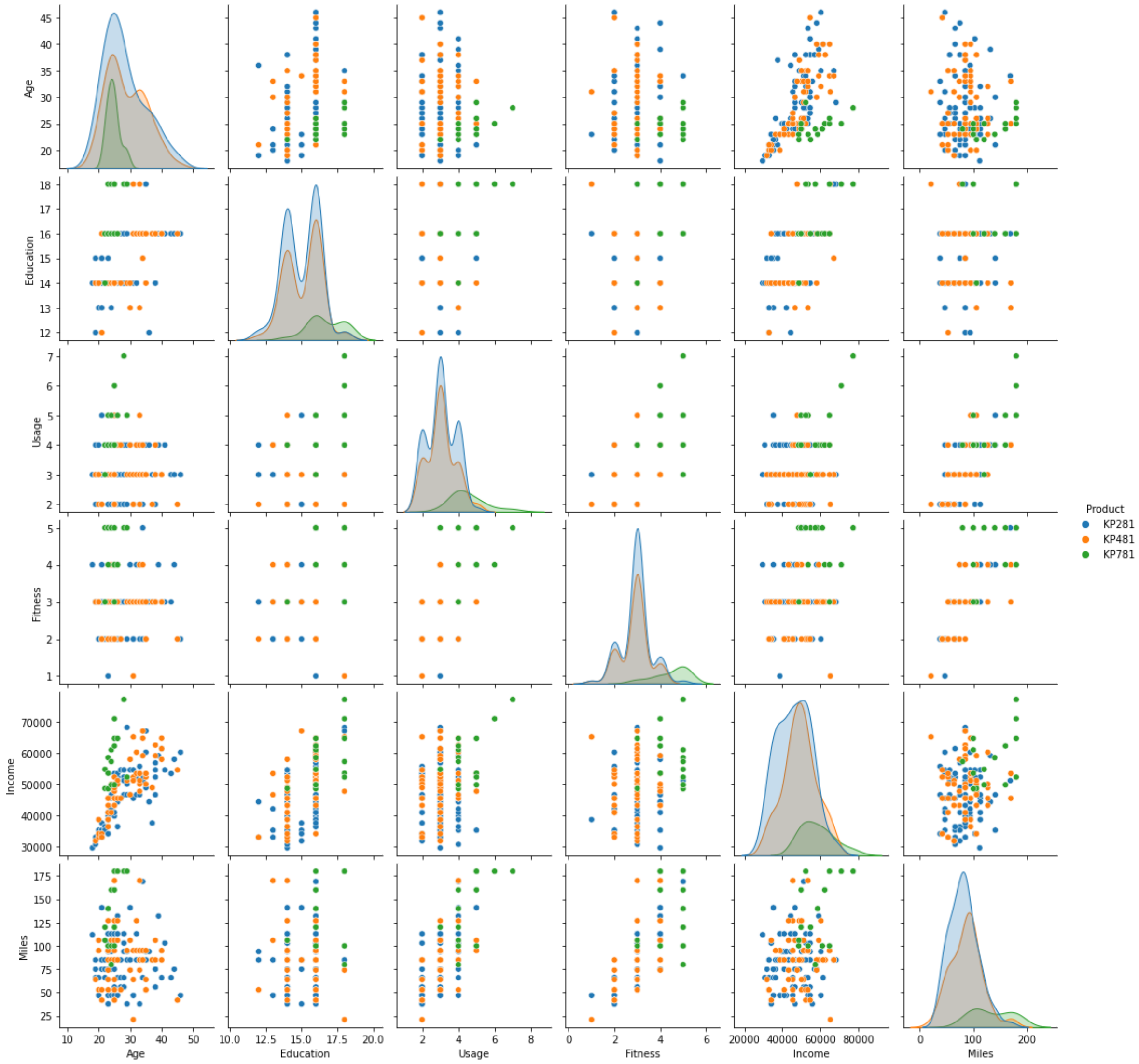


In [40]:

```
sns.pairplot(data=df,hue='Product')
# Below is the Pair plot for all the numerical columns
```

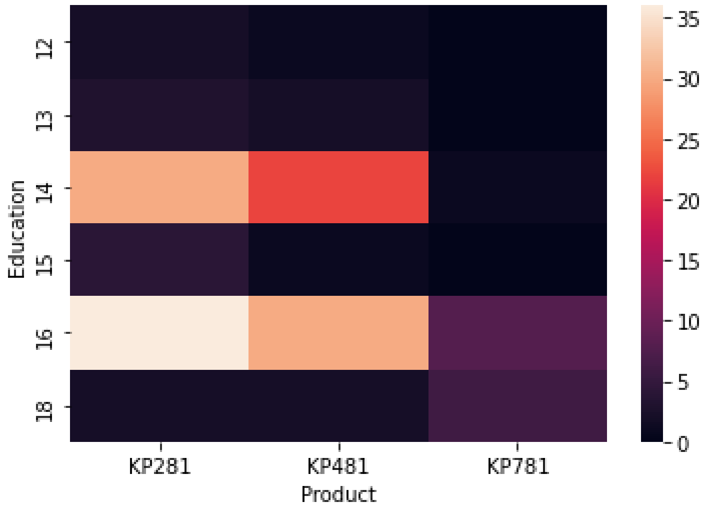
Out[40]:

```
<seaborn.axisgrid.PairGrid at 0x14e46739df0>
```



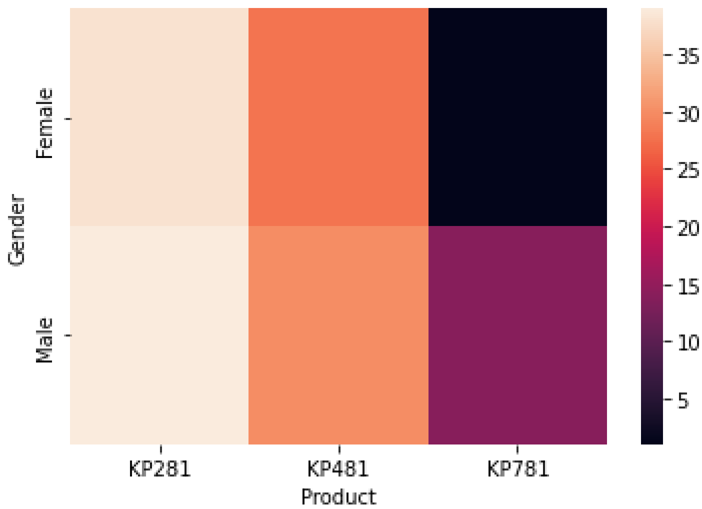
In [41]:

```
sns.heatmap(pd.crosstab(df['Education'],df['Product']))
plt.show()
# Below heat map shows:
# 1. KP281 :  People having Education years 14 and 16 buy more
# 2. KP481 :  People having Education years 14 and 16 buy more
```



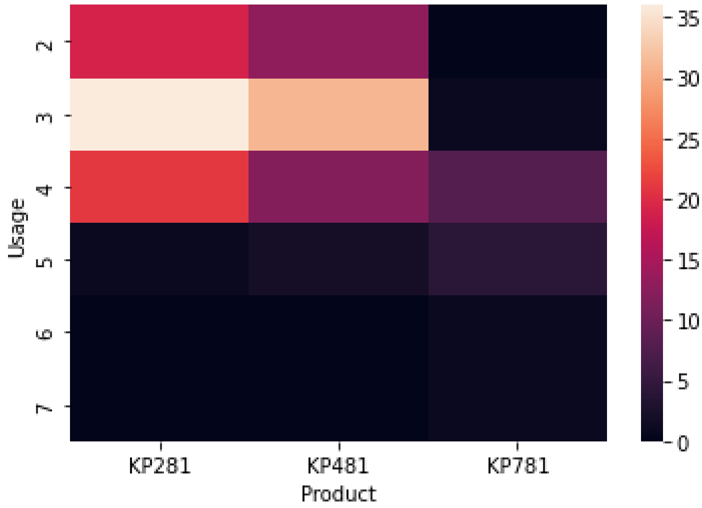
In [42]:

```
sns.heatmap(pd.crosstab(df['Gender'],df['Product']))
plt.show()
# Very few female buy KP781
# KP281 is more purchased by both Male and Female
```



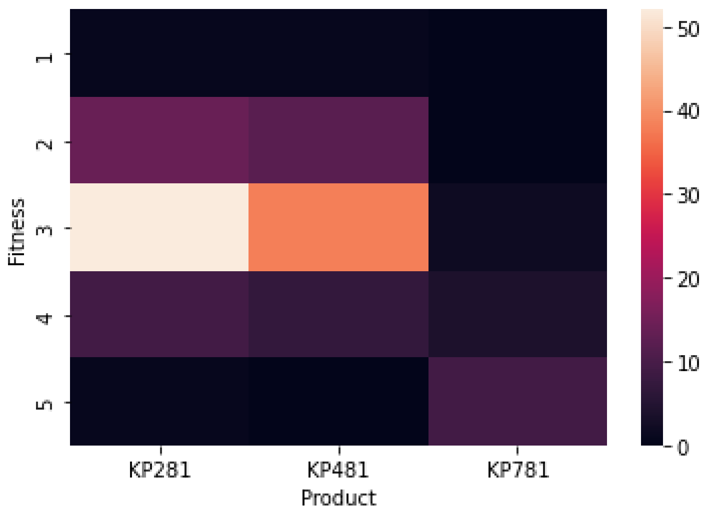
In [43]:

```
sns.heatmap(pd.crosstab(df['Usage'],df['Product']))
plt.show()
# Below heat map shows:
# 1. KP281: Has a people usage of 2,3 and 4
# 2. KP481: Has people usage of 2,3,4
# 3. KP781: Has people usage of 4 and 5
```



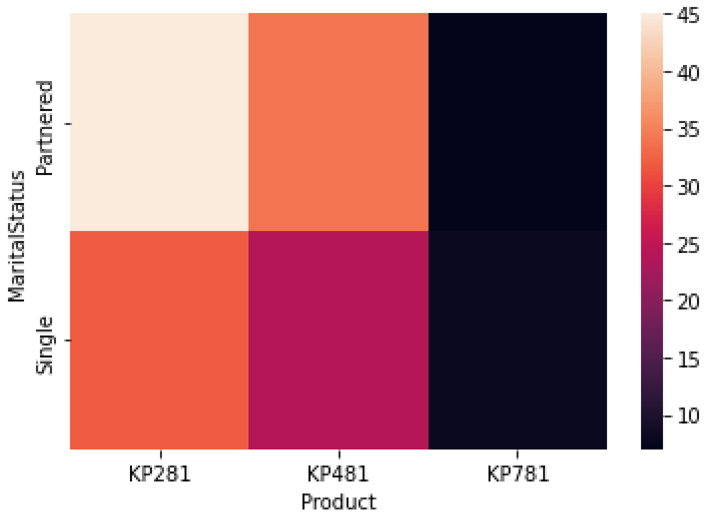
In [44]:

```
sns.heatmap(pd.crosstab(df['Fitness'],df['Product']))
plt.show()
# Below is the heat map of of product and Fitness
```



In [45]:

```
sns.heatmap(pd.crosstab(df['MaritalStatus'],df['Product']))
plt.show()
# Below is the heat map of of product and Martial Status
```



In [46]:

```
pd.crosstab(df['Product'],df['Gender'],margins=True)
# Below crosstab  gives the count of Male and Female in each product
```

Out[46]:

Gender	Female	Male	All
Product			
KP281	38	39	77
KP481	28	30	58
KP781	1	14	15
All	67	83	150

In [47]:

```
pd.crosstab(df['Product'],df['Gender'],margins=True,normalize=True)
# 51.33% people buy KP281, 38.667% people buy KP481 and 1% people buy KP781
```

Out[47]:

Gender Product	Female	Male	All
KP281	0.253333	0.260000	0.513333
KP481	0.186667	0.200000	0.386667
KP781	0.006667	0.093333	0.100000
All	0.446667	0.553333	1.000000

In [48]:

```
A=pd.crosstab(df['Product'],df['Gender'],margins=True,normalize=True)
```

In [49]:

```
A['Female']=A['Female']/A.iloc[3][0]
A['Male']=A['Male']/A.iloc[3][1]
A
# 56.71% female buy KP281, 41.17% female buy KP481 and 1.49% female buy KP781
# 46.988% male buy KP281, 36.1446% male buy KP481 and 16.86% male buy KP781
```

Out[49]:

Gender Product	Female	Male	All
KP281	0.567164	0.469880	0.513333
KP481	0.417910	0.361446	0.386667
KP781	0.014925	0.168675	0.100000
All	1.000000	1.000000	1.000000

In [50]:

```
pd.crosstab(df['Product'],df['Gender'],margins=True,normalize='index')
# Probality of Female and Male given they buy KP281 is 0.493 and 0.506 respectively
# Probality of Female and Male given they buy KP481 is 0.4827 and 0.517 respectively
# Probality of Female and Male given they buy KP781 is 0.066 and 00.9333 respectively
```

Out[50]:

Gender Product	Female	Male
KP281	0.493506	0.506494
KP481	0.482759	0.517241
KP781	0.066667	0.933333
All	0.446667	0.553333

In [51]:

```
pd.crosstab(df['Product'],df['Education'],margins=True,normalize=True)
```

Out[51]:

Education Product	12	13	14	15	16	18	All
KP281	0.013333	0.020000	0.200000	0.026667	0.240000	0.013333	0.513333
KP481	0.006667	0.013333	0.146667	0.006667	0.200000	0.013333	0.386667
KP781	0.000000	0.000000	0.006667	0.000000	0.053333	0.040000	0.100000
All	0.020000	0.033333	0.353333	0.033333	0.493333	0.066667	1.000000

In [52]:

```
A=pd.crosstab(df['Product'],df['Education'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 66.66 % of 12 years education buy KP281, 33.33% of 12 years education buy KP481
# 60 % of 13 years education buy KP281, 40% of 13 years education buy KP481
# 56.60 % of 14 years education buy KP281,of 41.5% 14 years education buy KP481
# 80 % of 15 years education buy KP281, 20% of 15 years education buy KP481
# 48.64 % of 16 years education buy KP281, 40.54% of 16 years education buy KP481 and 10.81
# 20 % of 18 years education buy KP281, 20%  of18 years education buy KP481 and 60%  of 18
```

Out[52]:

Education Product	12	13	14	15	16	18	All
KP281	0.666667	0.600000	0.566038	0.800000	0.486486	0.200000	0.513333
KP481	0.333333	0.400000	0.415094	0.200000	0.405405	0.200000	0.386667
KP781	0.000000	0.000000	0.018868	0.000000	0.108108	0.600000	0.100000
All	0.020000	0.033333	0.353333	0.033333	0.493333	0.066667	1.000000

In [53]:

```
pd.crosstab(df['Product'],df['Education'],margins=True,normalize='index')
# Probality of education years 11,12,13,14,15,16 and 18 given they buy KP281 is 0.025974,0.
# Probality of education years 11,12,13,14,15,16 and 18 given they buy KP481 is 0.017241,0.
# Probality of education years 11,12,13,14,15,16 and 18 given they buy KP781 is 0.000000,0.
```

Out[53]:

Education Product	12	13	14	15	16	18
KP281	0.025974	0.038961	0.389610	0.051948	0.467532	0.025974
KP481	0.017241	0.034483	0.379310	0.017241	0.517241	0.034483
KP781	0.000000	0.000000	0.066667	0.000000	0.533333	0.400000
All	0.020000	0.033333	0.353333	0.033333	0.493333	0.066667

In [54]:

```
pd.crosstab(df['Product'],df['MaritalStatus'],margins=True,normalize=True)
```

Out[54]:

MaritalStatus Product	Partnered	Single	All
KP281	0.300000	0.213333	0.513333
KP481	0.226667	0.160000	0.386667
KP781	0.046667	0.053333	0.100000
All	0.573333	0.426667	1.000000

In [55]:

```
A=pd.crosstab(df['Product'],df['MaritalStatus'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 52.32 % Partnered buy KP281 and 50% Single buy KP281
# 39.53 % Partnered buy KP481 and 37.5% Single buy KP481
# 8.1395 % Partnered buy KP781 and 12.5% Single buy KP781
```

Out[55]:

MaritalStatus Product	Partnered	Single	All
KP281	0.523256	0.500000	0.513333
KP481	0.395349	0.375000	0.386667
KP781	0.081395	0.125000	0.100000
All	0.573333	0.426667	1.000000

In [56]:

```
pd.crosstab(df['Product'],df['MaritalStatus'],margins=True,normalize='index')
# Probality of Partnered, single given that they buy KP281 is 0.584416 and 0.415584 respect
# Probality of Partnered, single given that they buy KP481 is 0.586207 and 0.41379 respecti
# Probality of Partnered, single given that they buy KP781 is 0.466667 and 0.5333 respectiv
```

Out[56]:

MaritalStatus	Partnered	Single
Product		
KP281	0.584416	0.415584
KP481	0.586207	0.413793
KP781	0.466667	0.533333
All	0.573333	0.426667

In [57]:

```
pd.crosstab(df['Product'],df['Fitness'],margins=True,normalize=True)
```

Out[57]:

Fitness	1	2	3	4	5	All
Product						
KP281	0.006667	0.093333	0.346667	0.060000	0.006667	0.513333
KP481	0.006667	0.080000	0.253333	0.046667	0.000000	0.386667
KP781	0.000000	0.000000	0.013333	0.026667	0.060000	0.100000
All	0.013333	0.173333	0.613333	0.133333	0.066667	1.000000

In [58]:

```
A=pd.crosstab(df['Product'],df['Fitness'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 50% of Fitness 1 buy KP281 and 50% of Fitness 1 buy KP481
# 53.84% of Fitness 2 buy KP281 and 46.15% of Fitness 2 buy KP481
# 56.52% of Fitness 3 buy KP281 and 41.30% of Fitness 3 buy KP481
# 45% of Fitness 4 buy KP281 ,35% of Fitness 4 buy KP481 and 20% of Fitness 4 buy KP781
# 10% of Fitness 5 buy KP281 ,0% of Fitness 5 buy KP481 and 90% of Fitness 5 buy KP781
```

Out[58]:

Fitness Product	1	2	3	4	5	All
KP281	0.500000	0.538462	0.565217	0.450000	0.100000	0.513333
KP481	0.500000	0.461538	0.413043	0.350000	0.000000	0.386667
KP781	0.000000	0.000000	0.021739	0.200000	0.900000	0.100000
All	0.013333	0.173333	0.613333	0.133333	0.066667	1.000000

In [59]:

```
pd.crosstab(df['Product'],df['Fitness'],margins=True,normalize='index')
# Probality of Fitness 1,2,3,4 and 5 given that they buy KP281 is 0.012987,0.181818,0.67532
# Probality of Fitness 1,2,3,4 and 5 given that they buy KP481 is 0.017241,0.206897,0.65517
# Probality of Fitness 1,2,3,4 and 5 given that they buy KP781 is 0.000000,0.000000,0.13333
```

Out[59]:

Fitness Product	1	2	3	4	5
KP281	0.012987	0.181818	0.675325	0.116883	0.012987
KP481	0.017241	0.206897	0.655172	0.120690	0.000000
KP781	0.000000	0.000000	0.133333	0.266667	0.600000
All	0.013333	0.173333	0.613333	0.133333	0.066667

In [60]:

```
pd.crosstab(df['Product'],df['Usage'],margins=True,normalize=True)
```

Out[60]:

Usage Product	2	3	4	5	6	7	All
KP281	0.126667	0.240000	0.140000	0.006667	0.000000	0.000000	0.513333
KP481	0.086667	0.206667	0.080000	0.013333	0.000000	0.000000	0.386667
KP781	0.000000	0.006667	0.053333	0.026667	0.006667	0.006667	0.100000
All	0.213333	0.453333	0.273333	0.046667	0.006667	0.006667	1.000000

In [61]:

```
A=pd.crosstab(df['Product'],df['Usage'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 59.275 % usage 2 buy KP281 and 40.625% usage 2 buy KP481
# 52.94 % usage 3 buy KP281 and 45.588% usage 3 buy KP481
# 51.21 % usage 4 buy KP281 and 29.26% usage 4 buy KP481 and 19.512% usage 4 buy KP781
# 14.285 % usage 5 buy KP281 and 28.57% usage 5 buy KP481 and 57.142% usage 5 buy KP781
# 0 % usage 6 buy KP281 and 0% usage 6 buy KP481 and 100% usage 6 buy KP781
# 0 % usage 7 buy KP281 and 0% usage 7 buy KP481 and 100% usage 7 buy KP781
```

Out[61]:

Usage	2	3	4	5	6	7	All
Product							
KP281	0.593750	0.529412	0.512195	0.142857	0.000000	0.000000	0.513333
KP481	0.406250	0.455882	0.292683	0.285714	0.000000	0.000000	0.386667
KP781	0.000000	0.014706	0.195122	0.571429	1.000000	1.000000	0.100000
All	0.213333	0.453333	0.273333	0.046667	0.006667	0.006667	1.000000

In [62]:

```
pd.crosstab(df['Product'],df['Usage'],margins=True,normalize='index')
# Probality of Usage 2,3,4 and 5 given that they buy KP281 is 0.246753,0.467532,0.272727 an
# Probality of Usage 2,3,4 and 5 given that they buy KP481 is 0.224138,0.534483,0.206897 an
# Probality of Usage 3,4,5,6 and 7 given that they buy KP781 is 0.066667,0.533333,0.266667,
```

Out[62]:

Usage	2	3	4	5	6	7
Product						
KP281	0.246753	0.467532	0.272727	0.012987	0.000000	0.000000
KP481	0.224138	0.534483	0.206897	0.034483	0.000000	0.000000
KP781	0.000000	0.066667	0.533333	0.266667	0.066667	0.066667
All	0.213333	0.453333	0.273333	0.046667	0.006667	0.006667

In [63]:

```
df.describe()
```

Out[63]:

	Age	Education	Usage	Fitness	Income	Miles
count	150.000000	150.000000	150.000000	150.000000	150.000000	150.000000
mean	27.973333	15.213333	3.200000	3.066667	48452.453333	88.540000
std	6.198015	1.313771	0.897424	0.791453	9375.253822	32.433852
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	23.000000	14.000000	3.000000	3.000000	42069.000000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	48891.000000	85.000000
75%	32.750000	16.000000	4.000000	3.000000	53511.750000	106.000000
max	46.000000	18.000000	7.000000	5.000000	77191.000000	180.000000

In [64]:

```
df['Age_split']=['0' for i in df['Age']]
for i in df['Age'].index:
    if df['Age'][i]<=19:
        df['Age_split'][i]='Age < 20'
    elif df['Age'][i]<=29:
        df['Age_split'][i]='20< Age < 30'
    elif df['Age'][i]<=39:
        df['Age_split'][i]='30 <Age < 40'
    elif df['Age'][i]<=49:
        df['Age_split'][i]='40 <Age < 50'
    elif df['Age'][i]<=60:
        df['Age_split'][i]='50< Age < 60'
df['Income_split']=['0' for i in df['Income']]
for i in df['Income'].index:
    if df['Income'][i]<=29999:
        df['Income_split'][i]='Income < 20000'
    elif df['Income'][i]<=39999:
        df['Income_split'][i]='30000< Income < 40000'
    elif df['Income'][i]<=49999:
        df['Income_split'][i]='40000 < Income < 50000'
    elif df['Income'][i]<=59999:
        df['Income_split'][i]='50000 < Income < 60000'
    elif df['Income'][i]<=69999:
        df['Income_split'][i]='60000< Income < 700000'
    elif df['Income'][i]<=80000:
        df['Income_split'][i]='70000< Income < 800000'
df['Miles_split']=['0' for i in df['Miles']]
for i in df['Miles'].index:
    if df['Miles'][i]<=59:
        df['Miles_split'][i]='Miles < 60'
    elif df['Miles'][i]<=79:
        df['Miles_split'][i]='60< Miles < 80'
    elif df['Miles'][i]<=99:
        df['Miles_split'][i]='80 < Miles < 100'
    elif df['Miles'][i]<=119:
        df['Miles_split'][i]='100 < Miles < 120'
    elif df['Miles'][i]<=139:
        df['Miles_split'][i]='120< Miles < 140'
    elif df['Miles'][i]<=159:
        df['Miles_split'][i]='140< Miles < 160'
    elif df['Miles'][i]<=180:
        df['Miles_split'][i]='160< Miles < 180'
```

```
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:4: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Age_split'][i]='Age < 20'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:6: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
```

```
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Age_split'][i]='20< Age < 30'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:8: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Age_split'][i]='30 <Age < 40'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:10: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Age_split'][i]='40 <Age < 50'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:16: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='Income < 20000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:18: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='30000< Income < 40000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:20: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='40000 < Income < 50000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:22: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='50000 < Income < 60000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:24: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='60000< Income < 700000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:26: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Income_split'][i]='70000< Income < 800000'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:36: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='100 < Miles < 120'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:32: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='60< Miles < 80'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:34: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='80 < Miles < 100'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:30: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='Miles < 60'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:40: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='140< Miles < 160'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:38: SettingWit
```

```
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='120< Miles < 140'
C:\Users\hp\AppData\Local\Temp/ipykernel_9284/1320381317.py:42: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  df['Miles_split'][i]='160< Miles < 180'
```

In [65]:

```
df
```

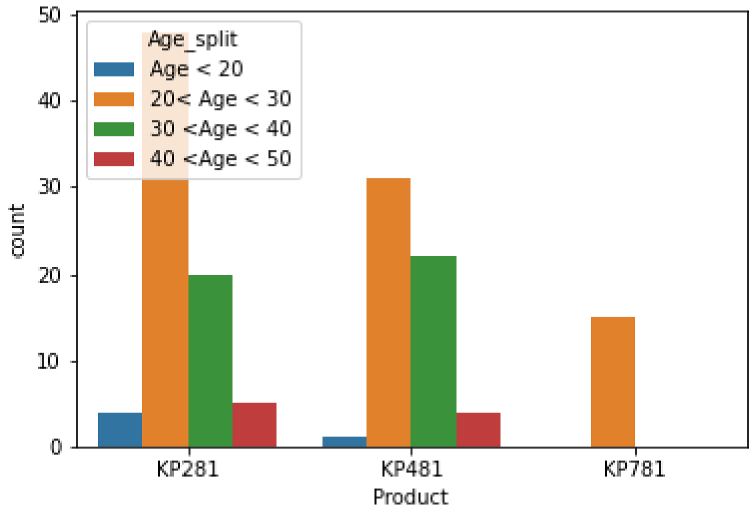
Out[65]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_spli
0	KP281	18	Male	14	Single	3	4	29562	112	Age < 2
1	KP281	19	Male	15	Single	2	3	31836	75	Age < 2
2	KP281	19	Female	14	Partnered	4	3	30699	66	Age < 2
3	KP281	19	Male	12	Single	3	3	32973	85	Age < 2
4	KP281	20	Male	13	Partnered	4	2	35247	47	20< Ag < 3
...	...	...	...	...	...	...	...	...	...	.
153	KP781	25	Male	18	Partnered	4	3	64741	100	20< Ag < 3
154	KP781	25	Male	18	Partnered	6	4	70966	180	20< Ag < 3
158	KP781	26	Male	16	Partnered	5	4	64741	180	20< Ag < 3
163	KP781	28	Male	18	Partnered	7	5	77191	180	20< Ag < 3
165	KP781	29	Male	18	Single	5	5	52290	180	20< Ag < 3

150 rows × 12 columns

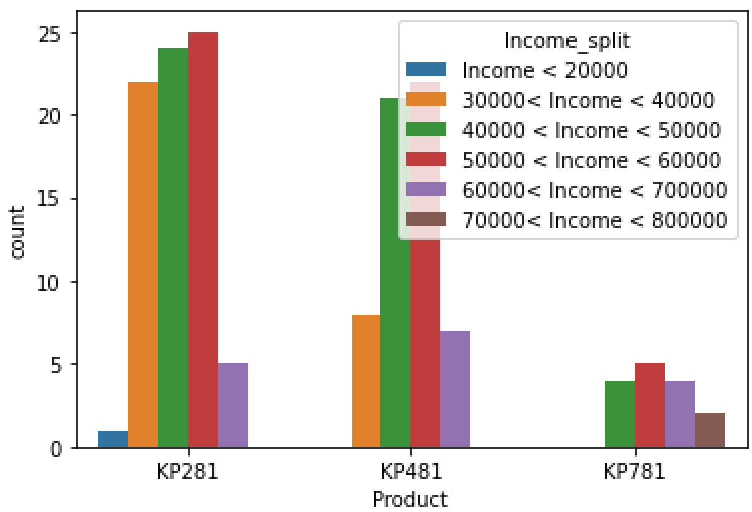
In [66]:

```
sns.countplot(x=df['Product'],hue=df['Age_split'])
plt.show()
# Below plot shows that KP281 is purchased by people in age group between 20 to 40.
# Below plot shows that KP481 is purchased by people in age group between 20 to 40.
# Below plot shows that KP781 is purchased by people in age group between 20 to 30.
```



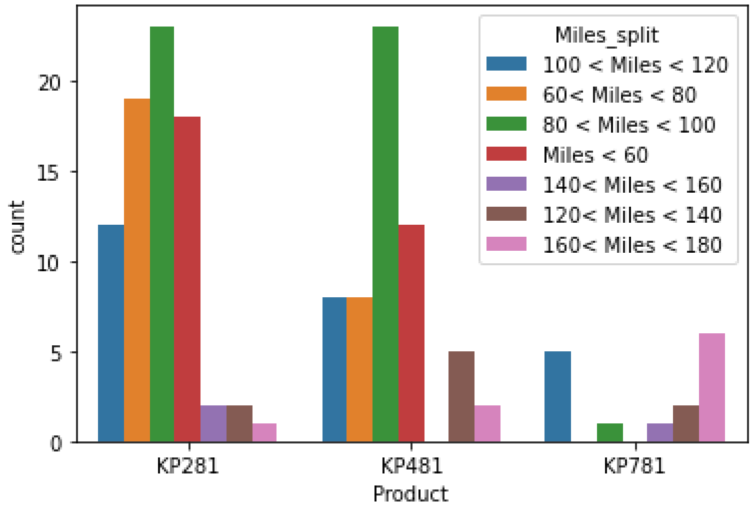
In [67]:

```
sns.countplot(x=df['Product'],hue=df['Income_split'])
plt.show()
# Below plot shows that KP281 is purchased by people in income group between 30000 to 60000
# Below plot shows that KP481 is purchased by people in age group between 40000 to 60000.
# Below plot shows that KP781 is purchased by people in age group between 40000 to 70000 mo
```



In [68]:

```
sns.countplot(x=df['Product'],hue=df['Miles_split'])
plt.show()
# Below plot shows that KP281 is purchased by people travel miles group between 60 to 100 a
# Below plot shows that KP481 is purchased by people travel miles group between 80 to 100 a
# Below plot shows that KP781 is purchased by people travel miles group between 140 to 180
```



In [69]:

```
pd.crosstab(df['Product'],df['Age_split'],margins=True,normalize=True)
```

Out[69]:

Age_split	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20	All
Product					
KP281	0.320000	0.133333	0.033333	0.026667	0.513333
KP481	0.206667	0.146667	0.026667	0.006667	0.386667
KP781	0.100000	0.000000	0.000000	0.000000	0.100000
All	0.626667	0.280000	0.060000	0.033333	1.000000

In [70]:

```
A=pd.crosstab(df['Product'],df['Age_split'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 20< Age < 30 : 51.06% of this age group buy KP281, 32.978% of this age group buy KP481 an
# 30 <Age < 40 : 47.619% of this age group buy KP281 and  52.381% of this age group buy KP4
# 40 <Age < 50 : 55.55% of this age group buy KP281, 44.44% of this age group buy KP481
# Age < 20 : 80% of this age group buy KP281, 20% of this age group buy KP481
```

Out[70]:

Age_split Product	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20	All
KP281	0.510638	0.47619	0.555556	0.800000	0.513333
KP481	0.329787	0.52381	0.444444	0.200000	0.386667
KP781	0.159574	0.00000	0.000000	0.000000	0.100000
All	0.626667	0.28000	0.060000	0.033333	1.000000

In [71]:

```
pd.crosstab(df['Product'],df['Age_split'],margins=True,normalize='index')
# Probality of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy KP2
# Probality of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy KP4
# Probality of 20< Age < 30, 30 <Age < 40,40 <Age < 50 and Age < 20 given that they buy KP7
```

Out[71]:

Age_split Product	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20
KP281	0.623377	0.25974	0.064935	0.051948
KP481	0.534483	0.37931	0.068966	0.017241
KP781	1.000000	0.00000	0.000000	0.000000
All	0.626667	0.28000	0.060000	0.033333

In [72]:

```
pd.crosstab(df['Product'],df['Income_split'],margins=True,normalize=True)
```

Out[72]:

Income_split	30000< Income < 40000	40000 < Income < 50000	50000 < Income < 60000	60000< Income < 700000	70000< Income < 800000	Income < 20000	All
Product							
KP281	0.146667	0.160000	0.166667	0.033333	0.000000	0.006667	0.513333
KP481	0.053333	0.140000	0.146667	0.046667	0.000000	0.000000	0.386667
KP781	0.000000	0.026667	0.033333	0.026667	0.013333	0.000000	0.100000
All	0.200000	0.326667	0.346667	0.106667	0.013333	0.006667	1.000000

In [73]:

```
A=pd.crosstab(df['Product'],df['Income_split'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 30000< Income < 40000 : 73.33 % of this income buy KP281 and 26.667  % of this income buy
# 40000 < Income < 50000 : 48.97 % of this income buy KP281, 42.857  % of this income buy K
# 50000 < Income < 60000 : 48.076 % of this income buy KP281, 42.3077  % of this income buy
# 60000< Income < 700000 :31.25 % of this income buy KP281, 43.75  % of this income buy KP4
# 70000< Income < 800000 :0 % of this income buy KP281, 0  % of this income buy KP481 and 1
# Income < 20000 :100 % of this income buy KP281
```

Out[73]:

Income_split	30000< Income < 40000	40000 < Income < 50000	50000 < Income < 60000	60000< Income < 700000	70000< Income < 800000	Income < 20000	All
Product							
KP281	0.733333	0.489796	0.480769	0.312500	0.000000	1.000000	0.513333
KP481	0.266667	0.428571	0.423077	0.437500	0.000000	0.000000	0.386667
KP781	0.000000	0.081633	0.096154	0.250000	1.000000	0.000000	0.100000
All	0.200000	0.326667	0.346667	0.106667	0.013333	0.006667	1.000000

In [74]:

```
pd.crosstab(df['Product'],df['Income_split'],margins=True,normalize='index')
# Probality of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 60000<
# Probality of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 60000<
# Probality of 30000< Income < 40000, 40000 < Income < 50000,50000 < Income < 60000, 60000<
```

Out[74]:

Income_split Product	30000< Income < 40000	40000 < Income < 50000	50000 < Income < 60000	60000< Income < 700000	70000< Income < 800000	Income < 20000
KP281	0.285714	0.311688	0.324675	0.064935	0.000000	0.012987
KP481	0.137931	0.362069	0.379310	0.120690	0.000000	0.000000
KP781	0.000000	0.266667	0.333333	0.266667	0.133333	0.000000
All	0.200000	0.326667	0.346667	0.106667	0.013333	0.006667

In [75]:

```
pd.crosstab(df['Product'],df['Miles_split'],margins=True,normalize=True)
```

Out[75]:

Miles_split Product	100 < Miles < 120	120< Miles < 140	140< Miles < 160	160< Miles < 180	60< Miles < 80	80 < Miles < 100	Miles < 60	All
KP281	0.080000	0.013333	0.013333	0.006667	0.126667	0.153333	0.12	0.513333
KP481	0.053333	0.033333	0.000000	0.013333	0.053333	0.153333	0.08	0.386667
KP781	0.033333	0.013333	0.006667	0.040000	0.000000	0.006667	0.00	0.100000
All	0.166667	0.060000	0.020000	0.060000	0.180000	0.313333	0.20	1.000000

In [76]:

```
A=pd.crosstab(df['Product'],df['Miles_split'],margins=True,normalize=True)
for i in range(3):
    A.iloc[i]=A.iloc[i]/A.iloc[3]
A
# 100 < Miles < 120 : 48 % of this mile range buy KP281,32  % of this mile range buy KP481
# 120< Miles < 140 : 22.222 % of this mile range buy KP281,55.556  % of this mile range buy
# 140< Miles < 160 : 66.667 % of this mile range buy KP281,0  % of this mile range buy KP48
# 160< Miles < 180 : 11.111 % of this mile range buy KP281,22.22  % of this mile range buy
# 60< Miles < 80 : 70.37 % of this mile range buy KP281,29.62  % of this mile range buy KP4
# 80 < Miles < 100 : 48.936 % of this mile range buy KP281,48.936  % of this mile range buy
# Miles < 60: 60 % of this mile range buy KP281,40  % of this mile range buy KP481
```

Out[76]:

Miles_split	100 < Miles < 120	120< Miles < 140	140< Miles < 160	160< Miles < 180	60< Miles < 80	80 < Miles < 100	Miles < 60	All
Product								
KP281	0.480000	0.222222	0.666667	0.111111	0.703704	0.489362	0.6	0.513333
KP481	0.320000	0.555556	0.000000	0.222222	0.296296	0.489362	0.4	0.386667
KP781	0.200000	0.222222	0.333333	0.666667	0.000000	0.021277	0.0	0.100000
All	0.166667	0.060000	0.020000	0.060000	0.180000	0.313333	0.2	1.000000

In [77]:

```
pd.crosstab(df['Product'],df['Miles_split'],margins=True,normalize='index')
# Below table shows the conditional probability of product and Miles
```

Out[77]:

Miles_split	100 < Miles < 120	120< Miles < 140	140< Miles < 160	160< Miles < 180	60< Miles < 80	80 < Miles < 100	Miles < 60
Product							
KP281	0.155844	0.025974	0.025974	0.012987	0.246753	0.298701	0.233766
KP481	0.137931	0.086207	0.000000	0.034483	0.137931	0.396552	0.206897
KP781	0.333333	0.133333	0.066667	0.400000	0.000000	0.066667	0.000000
All	0.166667	0.060000	0.020000	0.060000	0.180000	0.313333	0.200000

In [78]:

```
df.loc[df['Product']=='KP281'].describe(include=object)
# For KP281:
# Max purchase is done by people between age group 20 to 30
# Max purchase is done by people between income group 50000 to 60000
# Max purchase is done by people between miles group 80 to 100
```

Out[78]:

	Product	Gender	MaritalStatus	Age_split	Income_split	Miles_split
count	77	77	77	77	77	77
unique	1	2	2	4	5	7
top	KP281	Male	Partnered	20< Age < 30	50000 < Income < 60000	80 < Miles < 100
freq	77	39	45	48	25	23

In [79]:

```
df.loc[df['Product']=='KP481'].describe(include=object)
# For KP481:
# Max purchase is done by people between age group 20 to 30
# Max purchase is done by people between income group 50000 to 60000
# Max purchase is done by people between miles group 80 to 100
```

Out[79]:

	Product	Gender	MaritalStatus	Age_split	Income_split	Miles_split
count	58	58	58	58	58	58
unique	1	2	2	4	4	6
top	KP481	Male	Partnered	20< Age < 30	50000 < Income < 60000	80 < Miles < 100
freq	58	30	34	31	22	23

In [80]:

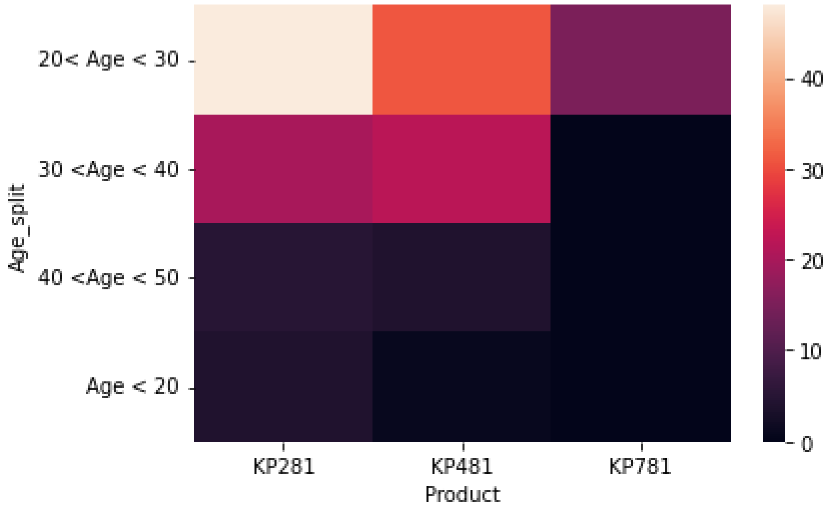
```
df.loc[df['Product']=='KP781'].describe(include=object)
# For KP781:
# Max purchase is done by people between age group 20 to 30
# Max purchase is done by people between income group 50000 to 60000
# Max purchase is done by people between miles group 160 t0 180
```

Out[80]:

	Product	Gender	MaritalStatus	Age_split	Income_split	Miles_split
count	15	15	15	15	15	15
unique	1	2	2	1	4	5
top	KP781	Male	Single	20< Age < 30	50000 < Income < 60000	160< Miles < 180
freq	15	14	8	15	5	6

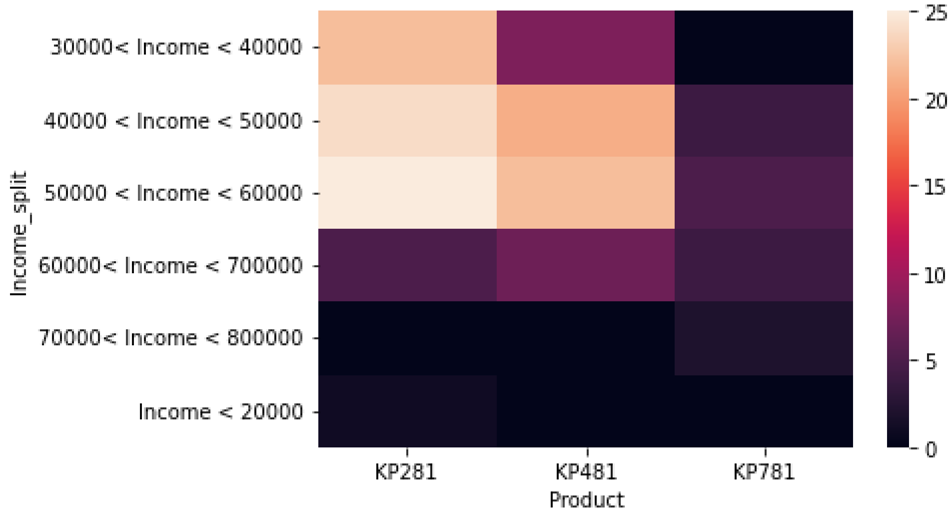
In [81]:

```
sns.heatmap(pd.crosstab(df['Age_split'],df['Product']))
plt.show()
# Below is the heat map between the age range and product
```



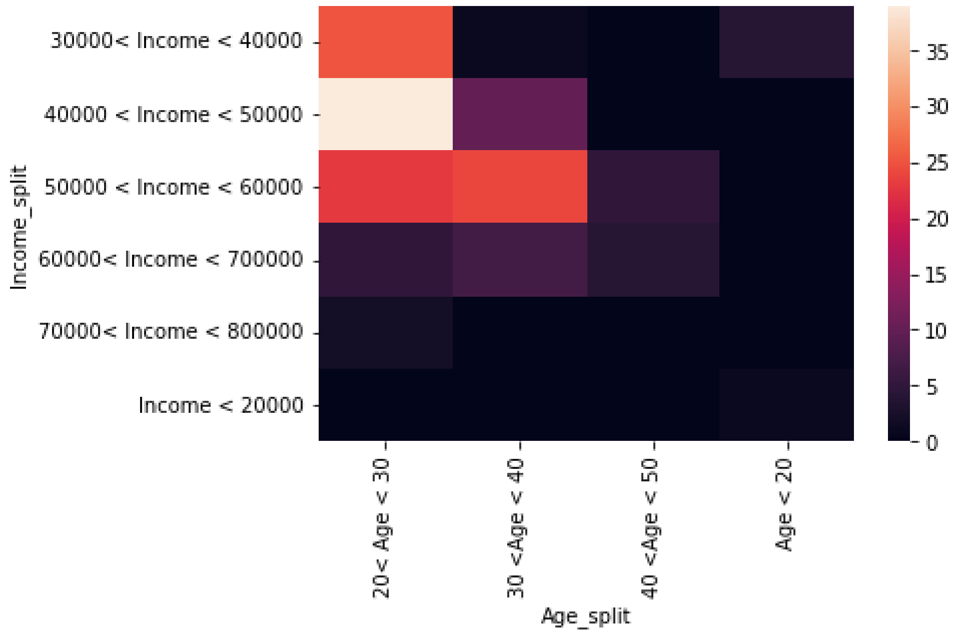
In [82]:

```
sns.heatmap(pd.crosstab(df['Income_split'],df['Product']))
plt.show()
# Below is the heat map between the income range and product
```



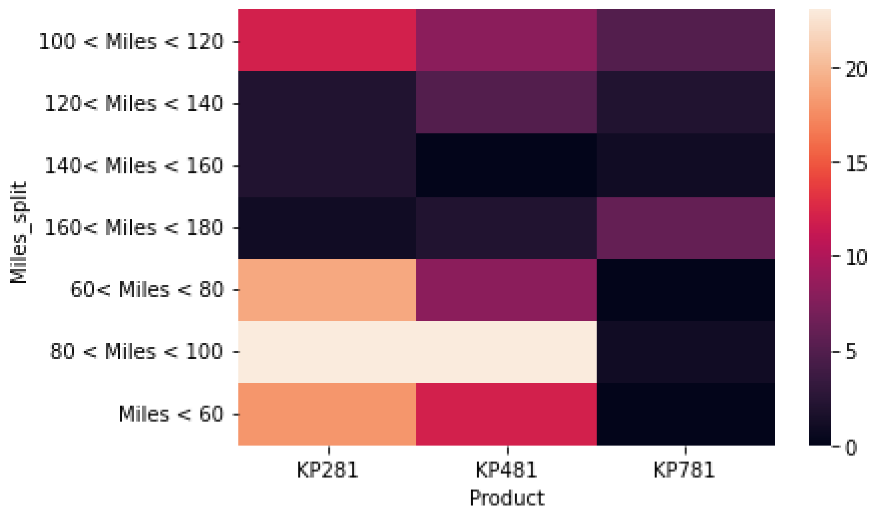
In [83]:

```
sns.heatmap(pd.crosstab(df['Income_split'],df['Age_split']))
plt.show()
# Below is the heat map between the age range and income range
```



In [84]:

```
sns.heatmap(pd.crosstab(df['Miles_split'],df['Product']))
plt.show()
```



In [85]:

```
pd.crosstab([df['Miles_split'],df['Product']],
                              df['Age_split'], margins = False)
```

Out[85]:

Miles_split	Product \ Age_split	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20
100 < Miles < 120	KP281	9	1	1	1
	KP481	6	2	0	0
	KP781	5	0	0	0
120< Miles < 140	KP281	1	1	0	0
	KP481	4	1	0	0
	KP781	2	0	0	0
140< Miles < 160	KP281	1	1	0	0
	KP781	1	0	0	0
160< Miles < 180	KP281	0	1	0	0
	KP481	1	1	0	0
	KP781	6	0	0	0
60< Miles < 80	KP281	11	3	3	2
	KP481	3	4	0	1
80 < Miles < 100	KP281	14	8	0	1
	KP481	9	11	3	0
	KP781	1	0	0	0
Miles < 60	KP281	12	5	1	0
	KP481	8	3	1	0

In [86]:

```
pd.crosstab([df['Fitness'],df['Product']],
                              df['Age_split'], margins = True)
```

Out[86]:

	Age_split	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20	All
Fitness	**Product**					
1	**KP281**	1	0	0	0	1
	KP481	0	1	0	0	1
2	**KP281**	9	4	1	0	14
	KP481	8	3	1	0	12
3	**KP281**	34	12	3	3	52
	KP481	19	15	3	1	38
	KP781	2	0	0	0	2
4	**KP281**	4	3	1	1	9
	KP481	4	3	0	0	7
	KP781	4	0	0	0	4
5	**KP281**	0	1	0	0	1
	KP781	9	0	0	0	9
All		94	42	9	5	150

In [87]:

```
pd.crosstab([df['Usage'],df['Product']],
                             df['Age_split'], margins = True)
```

Out[87]:

	Age_split	20< Age < 30	30 <Age < 40	40 <Age < 50	Age < 20	All
Usage	**Product**					
2	**KP281**	12	6	0	1	19
	KP481	8	4	1	0	13
3	**KP281**	22	8	4	2	36
	KP481	15	12	3	1	31
	KP781	1	0	0	0	1
4	**KP281**	13	6	1	1	21
	KP481	7	5	0	0	12
	KP781	8	0	0	0	8
5	**KP281**	1	0	0	0	1
	KP481	1	1	0	0	2
	KP781	4	0	0	0	4
6	**KP781**	1	0	0	0	1
7	**KP781**	1	0	0	0	1
All		94	42	9	5	150

In [ ]: