In [1]:

```
# Importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\pcs\Anaconda3\lib\site-packages\pandas\compat\_optional.py:138: UserWa
rning: Pandas requires version '2.7.0' or newer of 'numexpr' (version '2.6.8' c
urrently installed).
  warnings.warn(msg, UserWarning)
```

In [2]:

```
#Importing the dataset
df=pd.read_csv(r'F:\Scaler\Aerofit Case Study\aerofit_treadmill.txt')
```

In [3]:

```
df.head()
```

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

In [4]:

```
df.tail()
```

Out[4]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200
177	KP781	45	Male	16	Single	5	5	90886	160
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

In [5]:

```
#Checking the shape of the data
df.shape
```

Out[5]: (180, 9)

In [6]:

```
# Information about the dataset
```

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Product        180 non-null    object
 1   Age            180 non-null    int64 
 2   Gender         180 non-null    object
 3   Education      180 non-null    int64 
 4   MaritalStatus  180 non-null    object
 5   Usage          180 non-null    int64 
 6   Fitness        180 non-null    int64 
 7   Income         180 non-null    int64 
 8   Miles          180 non-null    int64 
dtypes: int64(6), object(3)
memory usage: 12.8+ KB
```

In [8]:
```
# No null values are present in the dataset as all the columns have 180 Non-
```

In [9]:
```
df.describe()
```

Out[9]:

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

Out[10]:

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

In [11]:
```
# KP281 is the most popular product
```

Checking the value count for each columns

In [12]:
```
df['Gender'].value_counts()
```

Out[12]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [13]:
```
df['MaritalStatus'].value_counts()
```

Out[13]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

In [14]:
```
df['Age'].value_counts(sort=True)
# Age group from 23 to 26 buys most of the products
```

Out[14]:
```
25    25
23    18
24    12
26    12
28     9
35     8
33     8
30     7
38     7
21     7
22     7
27     7
31     6
34     6
29     6
20     5
40     5
32     4
19     4
48     2
37     2
45     2
47     2
46     1
50     1
18     1
44     1
43     1
41     1
39     1
36     1
42     1
Name: Age, dtype: int64
```

In [15]:
```
df['Education'].value_counts()
# Most of the people have 16 or 14 years of Education
```

Out[15]:
```
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

In [16]:
```
df['Fitness'].value_counts()
#Most of the customers have Fitness"3"
```

Out[16]:
```
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

In [17]:
```
df['Usage'].value_counts()
# Most of the customers use treadmill 3 - 4 times a week
```

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

In [18]:
```
df.groupby('Product')['Gender'].value_counts()
```

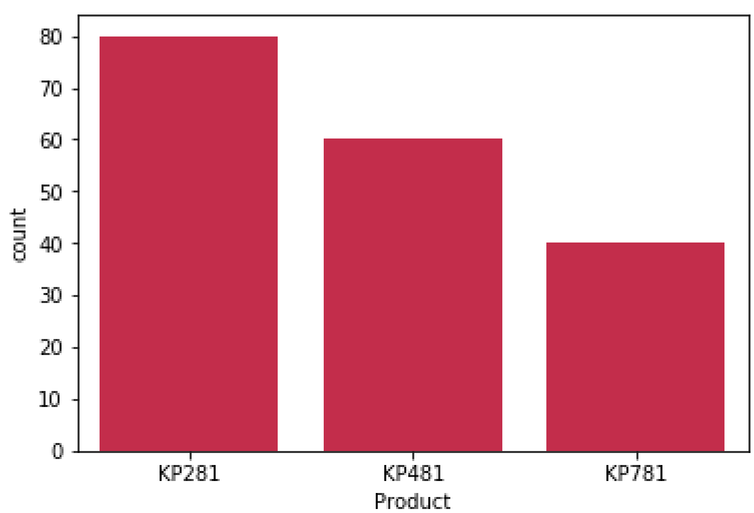
Out[18]:
```
Product  Gender
KP281    Female    40
         Male      40
KP481    Male      31
         Female    29
KP781    Male      33
         Female     7
Name: Gender, dtype: int64
```

In [19]:
```
df.groupby(['Product','Gender'])['MaritalStatus'].value_counts()
```

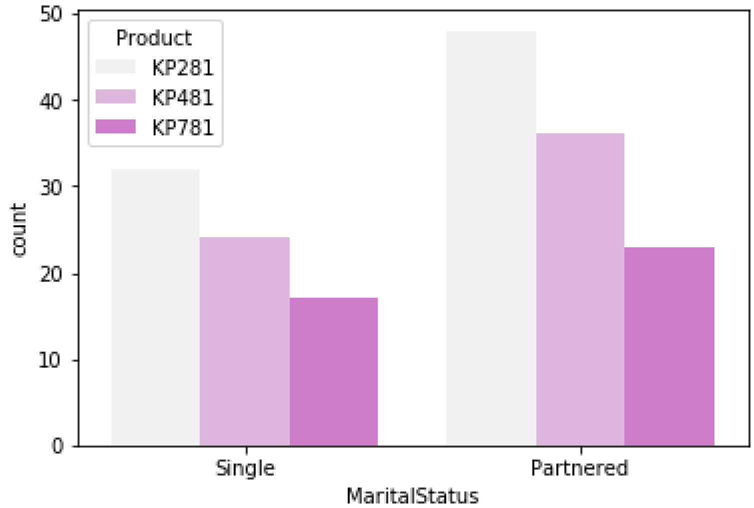
Out[19]:
```
Product  Gender  MaritalStatus
KP281    Female  Partnered        27
                 Single           13
         Male    Partnered        21
                 Single           19
KP481    Female  Partnered        15
                 Single           14
         Male    Partnered        21
                 Single           10
KP781    Female  Partnered         4
                 Single            3
         Male    Partnered        19
                 Single           14
Name: MaritalStatus, dtype: int64
```

Visual Analysis

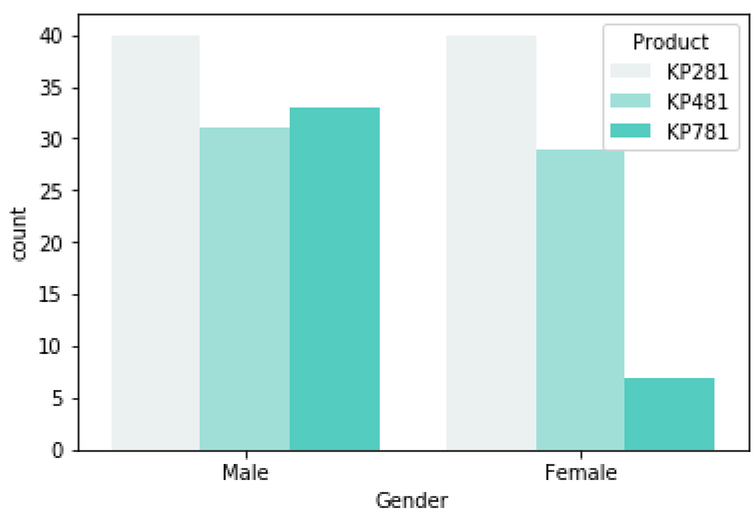
In [20]:
```
sns.countplot(x ='Product', data = df, color='crimson')
plt.show()
# KP281 is the most popular product and KP781 is the least popular product
```



In [21]:
```
sns.countplot(x = 'MaritalStatus', data = df,color='orchid',hue='Product')
plt.show()
```

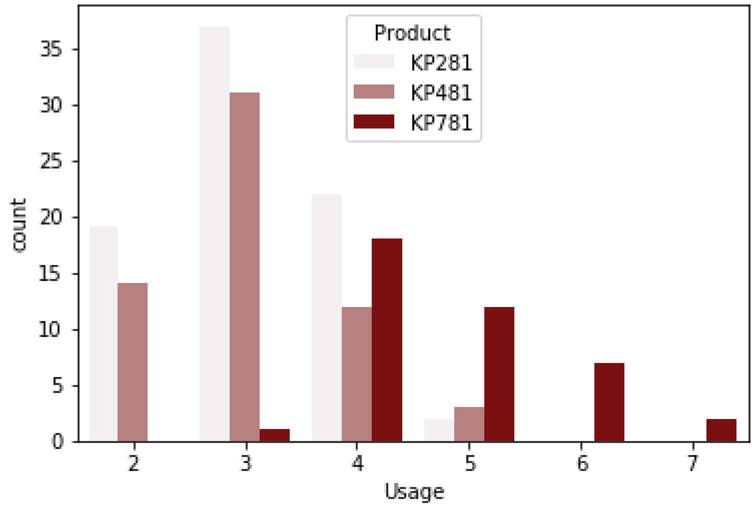


In [22]:
```
sns.countplot(x = 'Gender', data = df,hue='Product',color='turquoise')
plt.show()
```



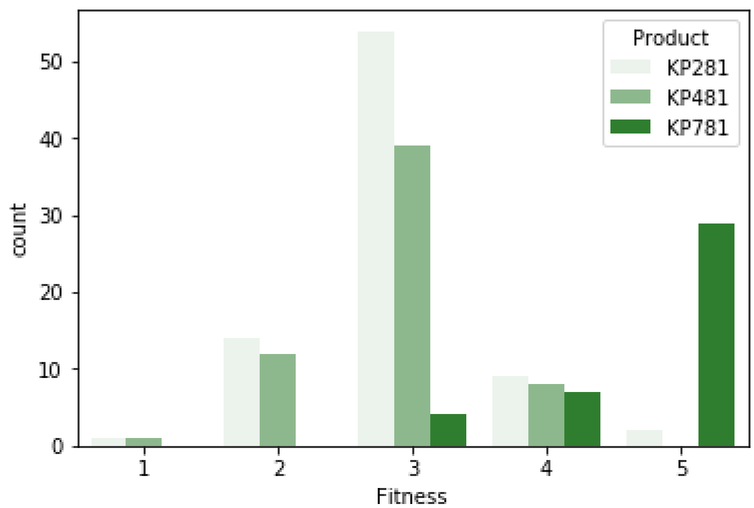
In [23]:

```
sns.countplot(x = 'Usage', data = df,hue='Product',color='darkred')
plt.show()
```

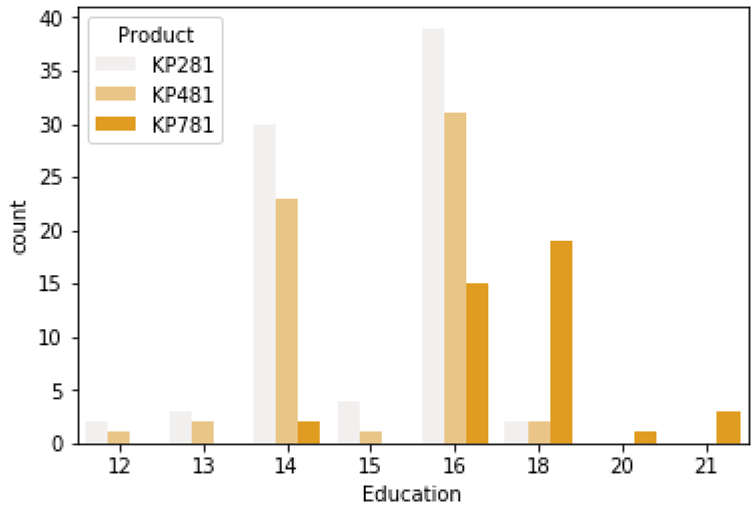


In [24]:

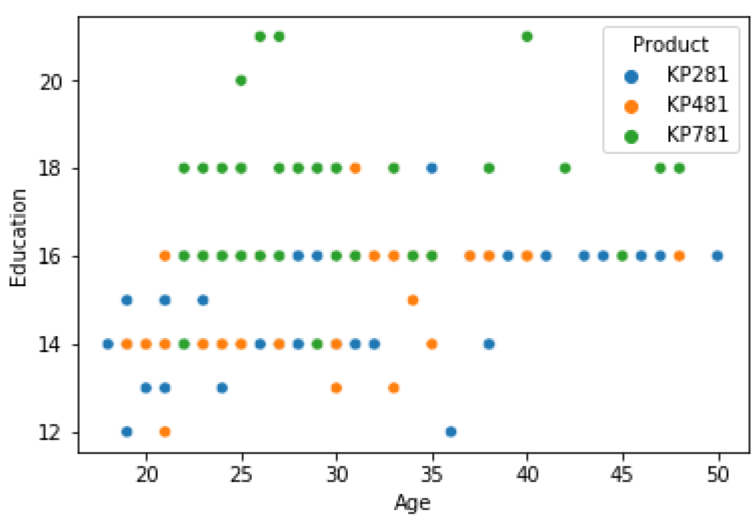
```
sns.countplot(x = 'Fitness', data = df,hue='Product',color='forestgreen')
plt.show()
```



In [25]:
```
sns.countplot(x = 'Education', data = df,hue='Product',color='orange')
plt.show()
```

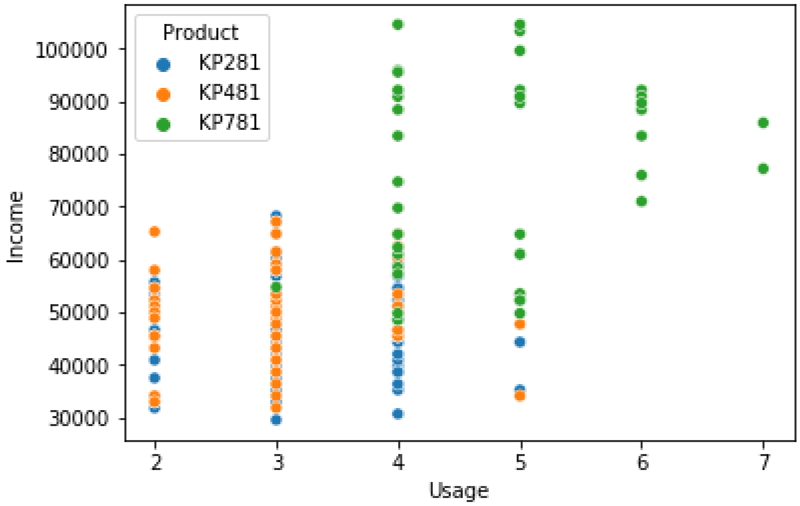


In [26]:
```
sns.scatterplot(x= df['Age'], y = df['Education'],hue='Product',data=df)
plt.show()
# Most of the person who has eduation of 18 or 18+ years opt for KP781
```



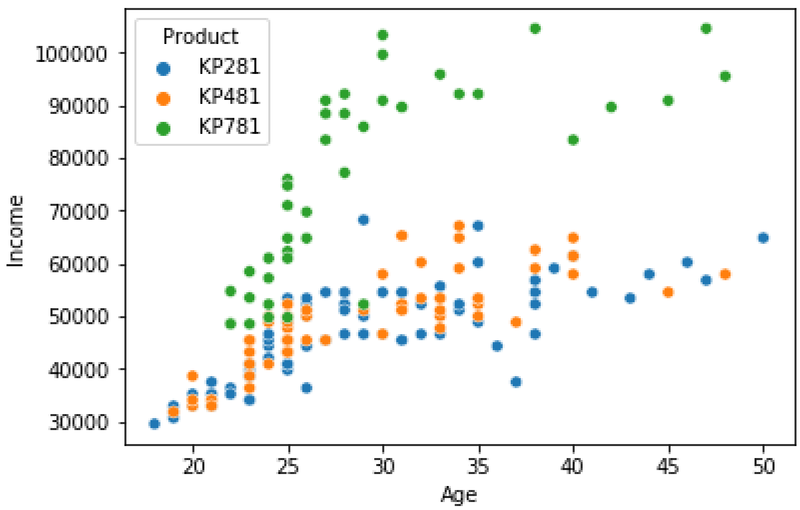
In [27]:

```
sns.scatterplot(x= df['Usage'], y = df['Income'] ,hue='Product',data=df)
plt.show()
```



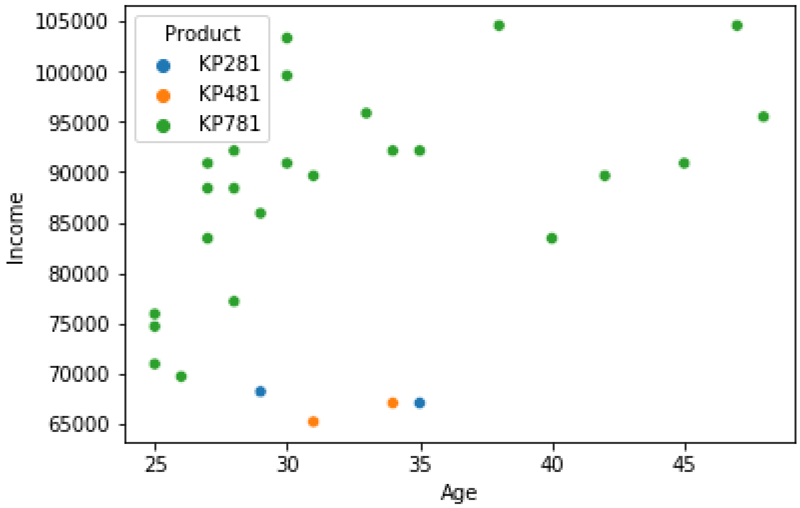
In [28]:

```
sns.scatterplot(x= df['Age'], y = df['Income'] ,hue='Product',data=df)
plt.show()
```



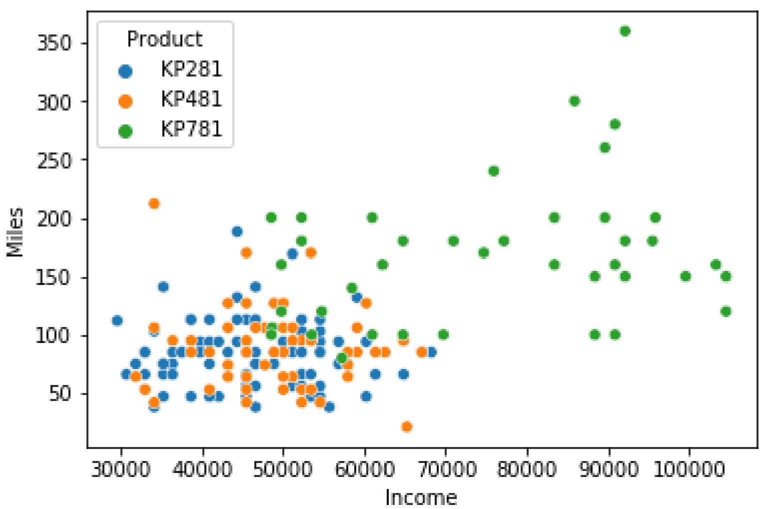
In [29]:

```
sns.scatterplot(x= 'Age', y = df.loc[df['Income']>65000]['Income'] ,hue='Pro
plt.show()
# Most of the customers who have Income more than 65000 opt for KP781
```



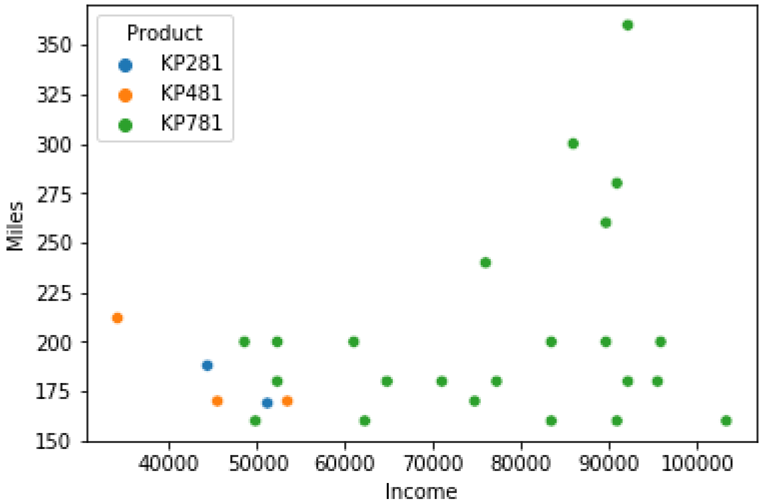
In [30]:

```
sns.scatterplot(x= df['Income'], y = df['Miles'] ,hue='Product',data=df)
plt.show()
```



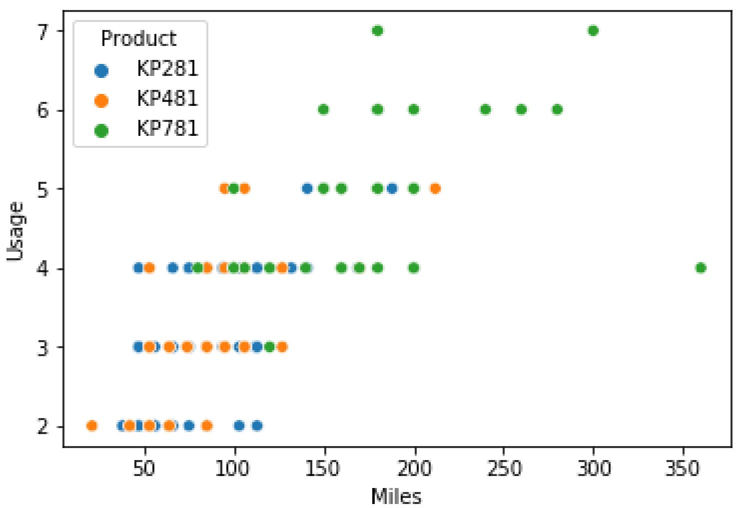
In [31]:

```
sns.scatterplot(x= df['Income'], y = df.loc[df['Miles']>150]['Miles'] ,hue='
plt.show()
```

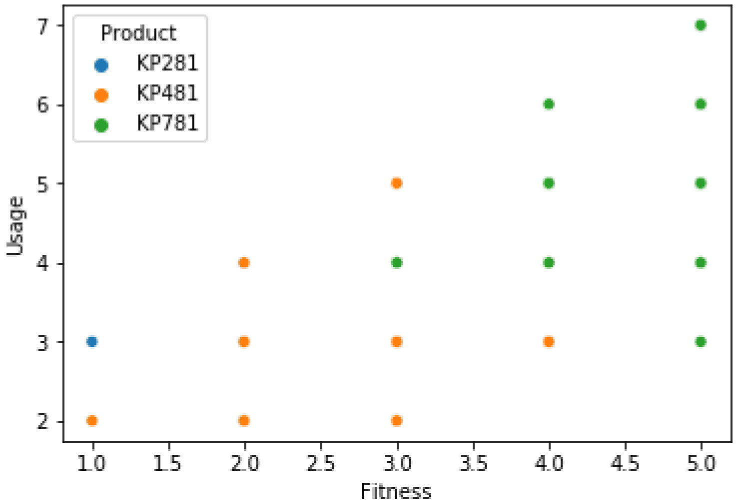


In [32]:

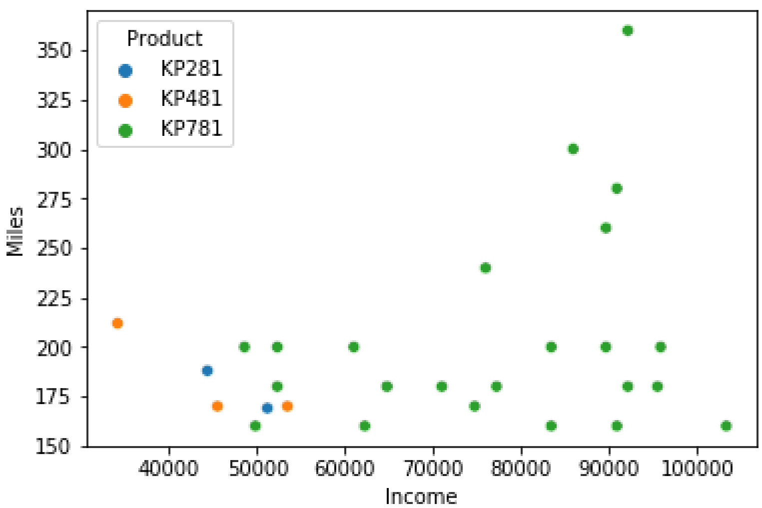
```
sns.scatterplot(x= df['Miles'], y = df['Usage'] ,hue='Product',data=df)
plt.show()
```



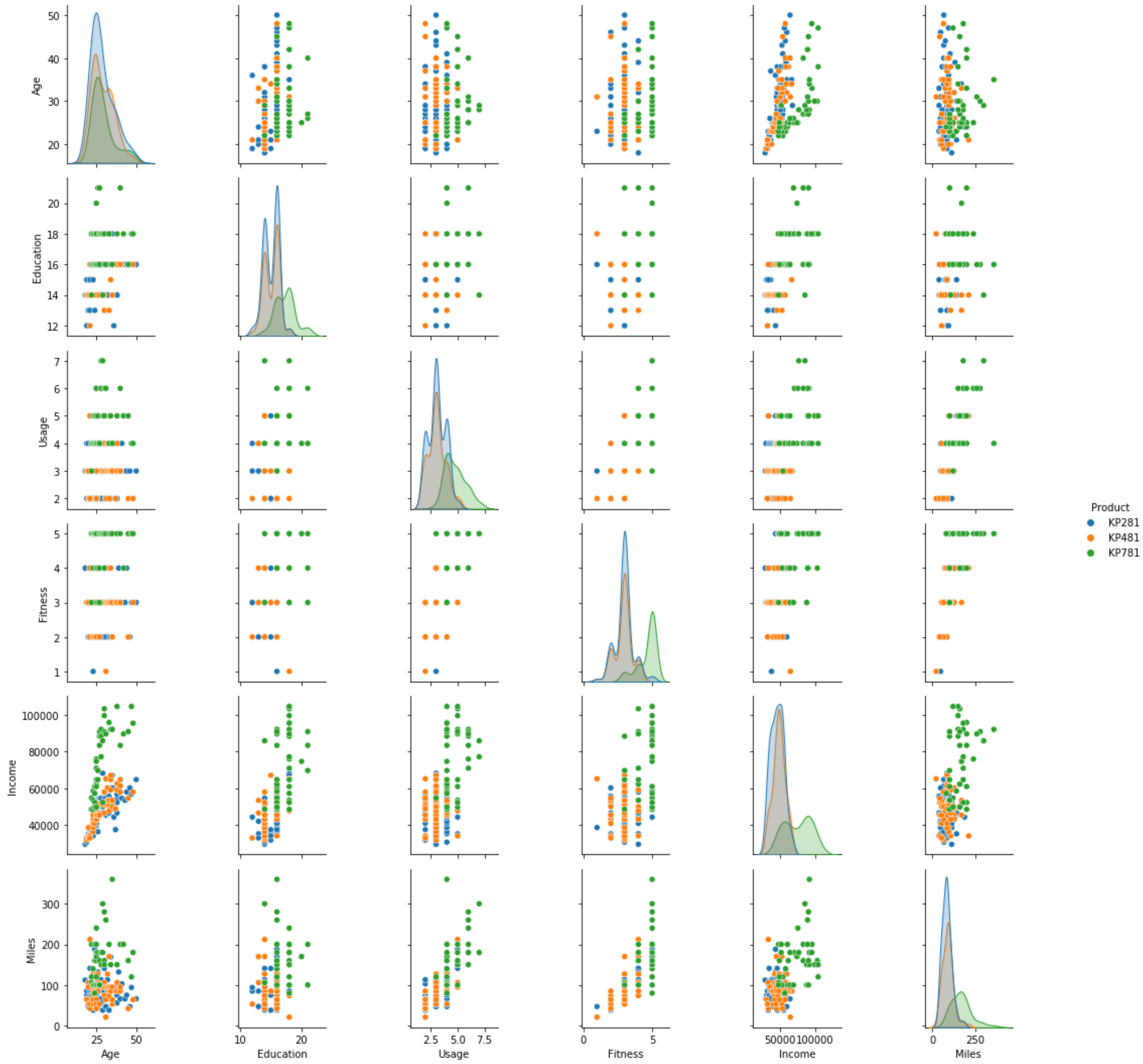
In [33]:
```
sns.scatterplot(x= df['Fitness'], y = df['Usage'] ,hue='Product',data=df,col
plt.show()
```



In [34]:
```
sns.scatterplot(x= df['Income'], y = df.loc[df['Miles']>150]['Miles'] ,hue='
plt.show()
```



In [35]:
```
sns.pairplot(data = df, hue= 'Product')
plt.show()
#Pair Plot
```

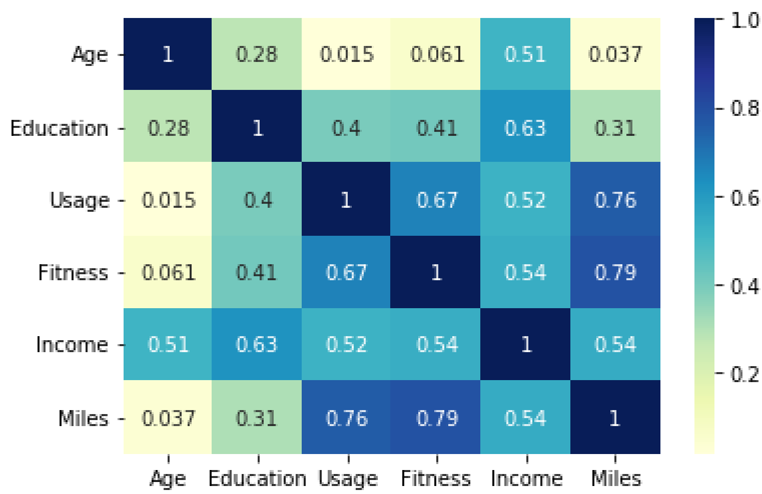


In [36]:
```
df.corr()
```

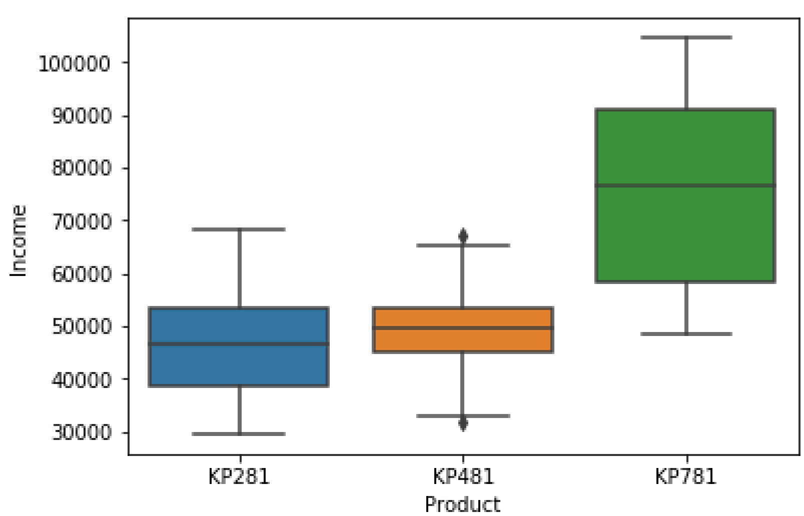
Out[36]:

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

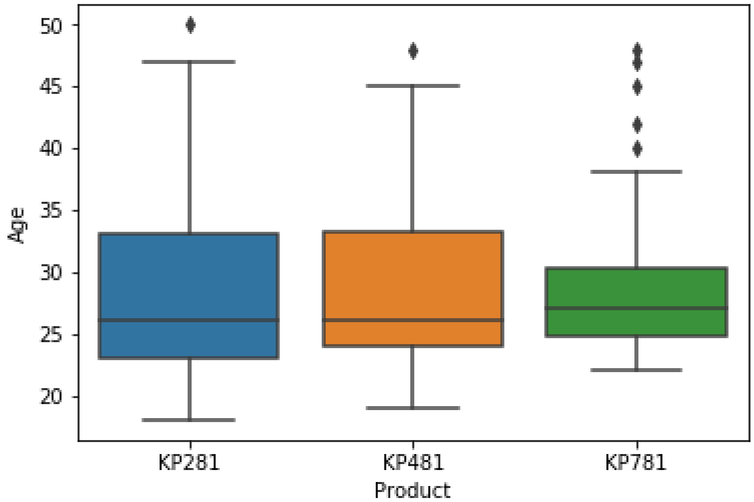
In [37]:
```
# Co-relation between different parameters
sns.heatmap(df.corr(), cmap= "YlGnBu", annot=True)
plt.show()
```



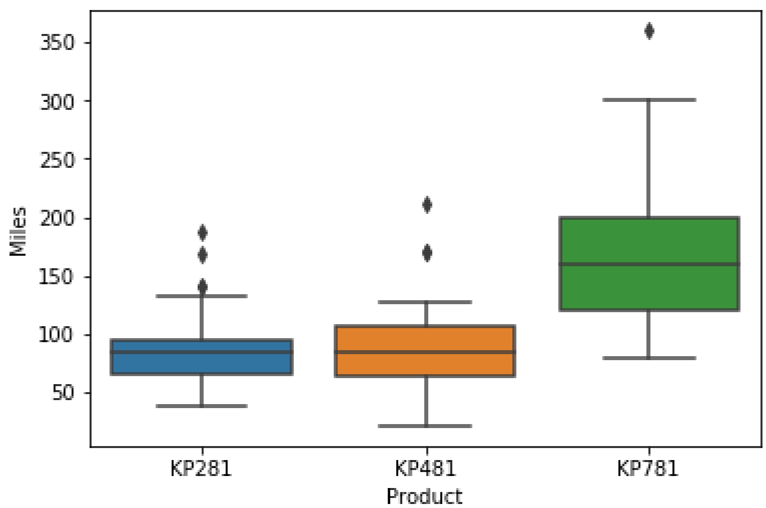
In [38]:
```
sns.boxplot(x = 'Product', y = 'Income', data = df)
plt.show()
```



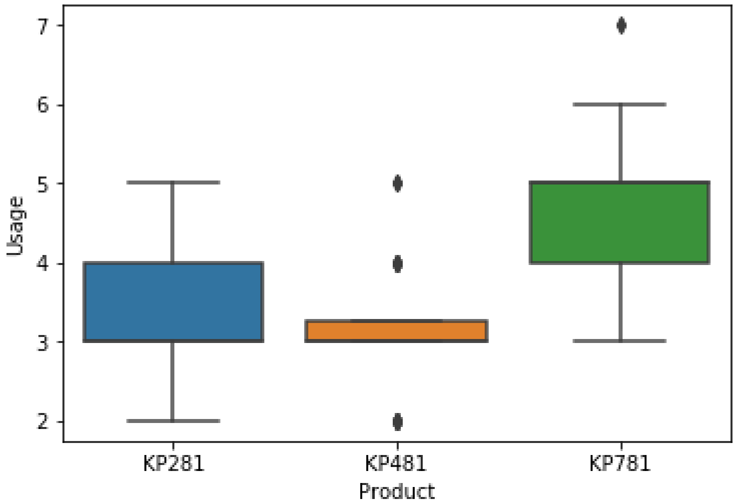
In [39]:
```
sns.boxplot(x = 'Product', y = 'Age', data = df)
plt.show()
# Most of the customers are b/w age 23 to 32
```



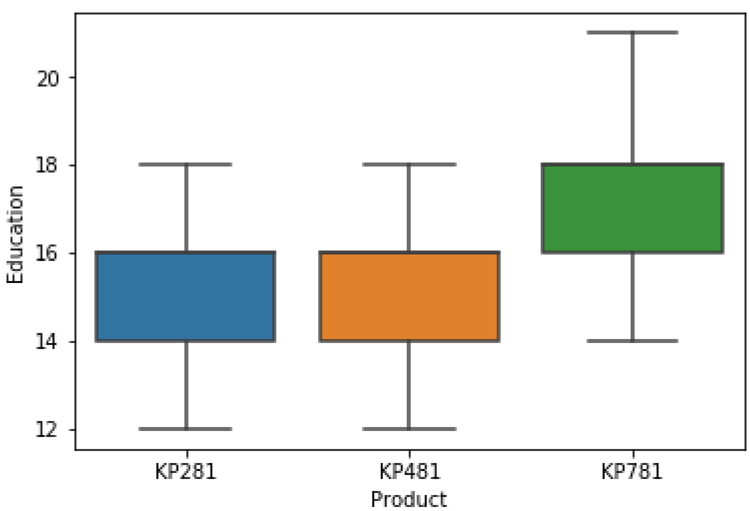
In [40]:
```
sns.boxplot(x = 'Product', y = 'Miles', data = df)
plt.show()
#Customers who run for more than 150 miles prefer KP781
```



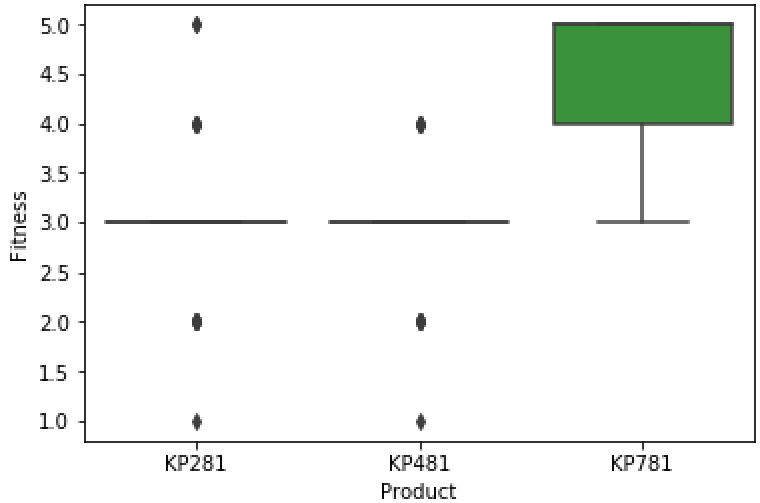
In [41]:
```
sns.boxplot(x = 'Product', y = 'Usage', data = df)
plt.show()
```



In [42]:
```
sns.boxplot(x = 'Product', y = 'Education', data = df)
plt.show()
# KP281 & KP481 has range b/w 14 to 16 but KP781 has range from 16-18
```



In [43]:
```
sns.boxplot(x = 'Product', y = 'Fitness', data = df)
plt.show()
```



Probability Tables

In [44]:
```
pd.crosstab(df.Product,df.Gender,normalize='columns',margins=True,margins_na
```

Out[44]:

Gender Product	Female	Male	Total
KP281	0.526316	0.384615	0.444444
KP481	0.381579	0.298077	0.333333
KP781	0.092105	0.317308	0.222222

In [45]:
```
pd.crosstab(df.Product, df.MaritalStatus,normalize='columns')
```

Out[45]:

MaritalStatus Product	Partnered	Single
KP281	0.448598	0.438356
KP481	0.336449	0.328767
KP781	0.214953	0.232877

In [46]:
```
pd.crosstab(df.Product, df.Age,normalize=True)
```

Out[46]:

Age Product	18	19	20	21	22	23	24	25	26
KP281	0.005556	0.016667	0.011111	0.022222	0.022222	0.044444	0.027778	0.038889	0.038889
KP481	0.000000	0.005556	0.016667	0.016667	0.000000	0.038889	0.016667	0.061111	0.016667
KP781	0.000000	0.000000	0.000000	0.000000	0.016667	0.016667	0.022222	0.038889	0.011111

3 rows × 32 columns

In [47]:
```
pd.crosstab(df.Product, df.Usage,normalize=True)
```

Out[47]:

Usage Product	2	3	4	5	6	7
KP281	0.105556	0.205556	0.122222	0.011111	0.000000	0.000000
KP481	0.077778	0.172222	0.066667	0.016667	0.000000	0.000000
KP781	0.000000	0.005556	0.100000	0.066667	0.038889	0.011111

In [48]:
```
pd.crosstab(df.Product, df.Fitness,normalize=True)
```

Out[48]:

Fitness Product	1	2	3	4	5
KP281	0.005556	0.077778	0.300000	0.050000	0.011111
KP481	0.005556	0.066667	0.216667	0.044444	0.000000
KP781	0.000000	0.000000	0.022222	0.038889	0.161111

In [49]:
```
pd.crosstab(df.Product, df.Education,normalize=True)
```

Out[49]:

Education Product	12	13	14	15	16	18	20	21
KP281	0.011111	0.016667	0.166667	0.022222	0.216667	0.011111	0.000000	0.000000
KP481	0.005556	0.011111	0.127778	0.005556	0.172222	0.011111	0.000000	0.000000
KP781	0.000000	0.000000	0.011111	0.000000	0.083333	0.105556	0.005556	0.016667

In [50]:
```
pd.crosstab(df.Product,[df.Gender, df.MaritalStatus],normalize=True)
```

Out[50]:

Gender	Female		Male	
MaritalStatus	Partnered	Single	Partnered	Single
Product				
KP281	0.150000	0.072222	0.116667	0.105556
KP481	0.083333	0.077778	0.116667	0.055556
KP781	0.022222	0.016667	0.105556	0.077778