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

In [3]:
```
df = pd.read_csv('C:/Users/91981/Desktop/Mansi/scaler/Prob and Stats/aerofi
```

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 [5]:
```
df.describe(include = 'all')
```

Out[5]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	
count	180	180.000000	180	180.000000	180	180.000000	180.000000	
unique	3	NaN	2	NaN	2	NaN	NaN	
top	KP281	NaN	Male	NaN	Partnered	NaN	NaN	
freq	80	NaN	104	NaN	107	NaN	NaN	
mean	NaN	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29
25%	NaN	24.000000	NaN	14.000000	NaN	3.000000	3.000000	44
50%	NaN	26.000000	NaN	16.000000	NaN	3.000000	3.000000	50
75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58
max	NaN	50.000000	NaN	21.000000	NaN	7.000000	5.000000	104

In [6]:
```
df.shape
```

Out[6]:
```
(180, 9)
```

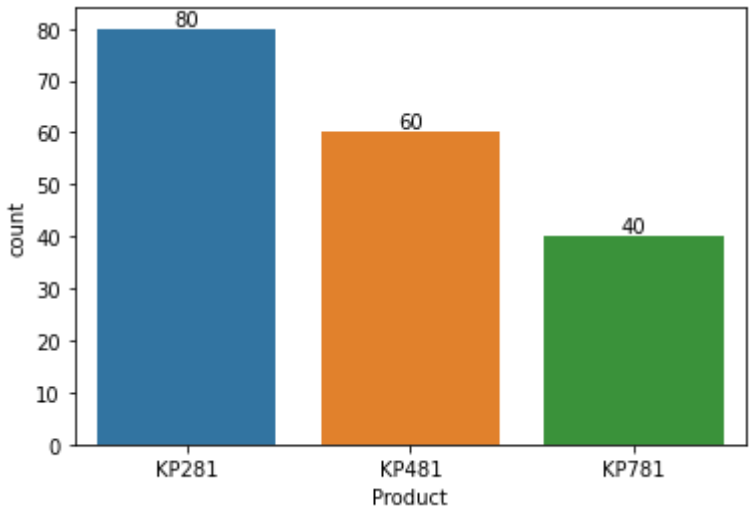
In [8]:
```
df.info()
```

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

In [16]:
```
Pdt_counts = df['Product'].value_counts()
Pdt_counts
```

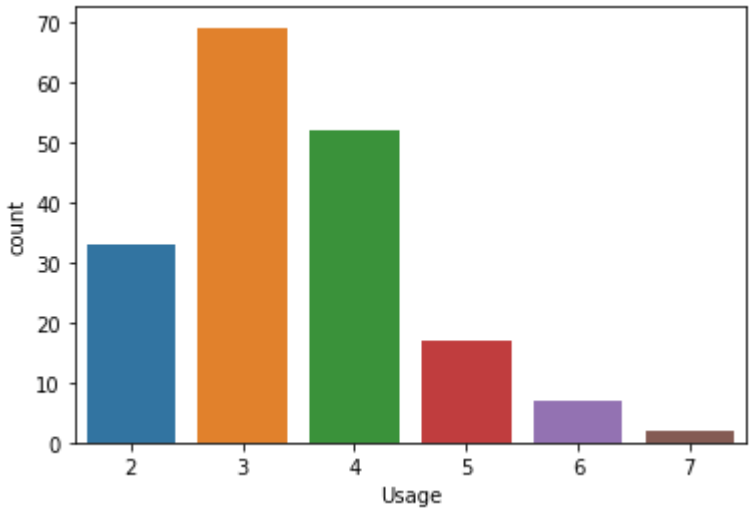
Out[16]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [3]:
```
ax = sns.countplot(x = 'Product', data = df)
for i in ax.containers:
    ax.bar_label(i,)
plt.show()
```



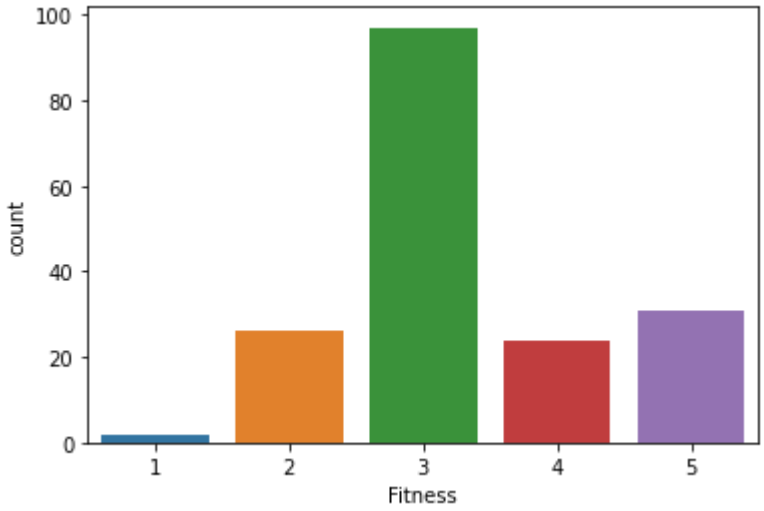
In [5]:
```
df['Usage'].value_counts()
sns.countplot(x = 'Usage', data = df)
```

Out[5]: <AxesSubplot:xlabel='Usage', ylabel='count'>

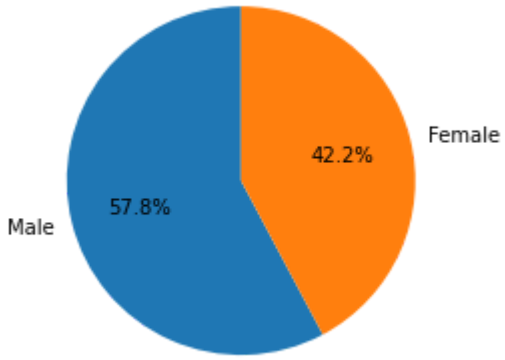


In [6]:
```
df['Fitness'].value_counts()
sns.countplot(x = 'Fitness', data = df)
```

Out[6]: <AxesSubplot:xlabel='Fitness', ylabel='count'>

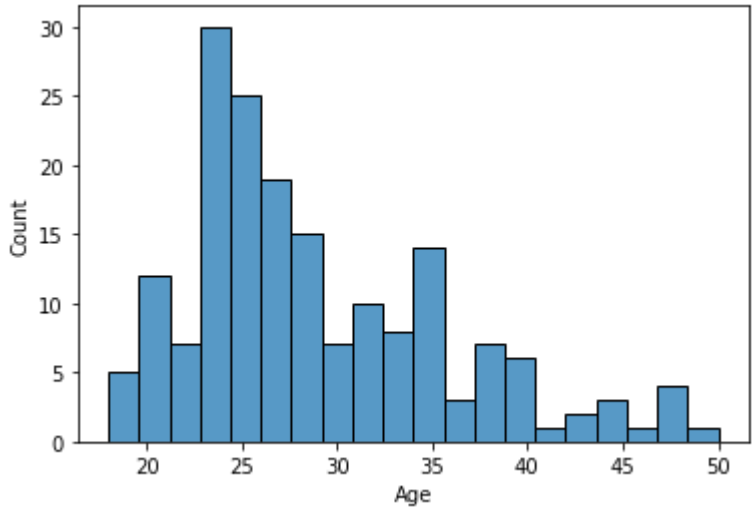


In [14]:
```
gen_counts = df['Gender'].value_counts()
plt.pie(gen_counts, labels= gen_counts.index, autopct='%1.1f%%', startangle
plt.show()
```



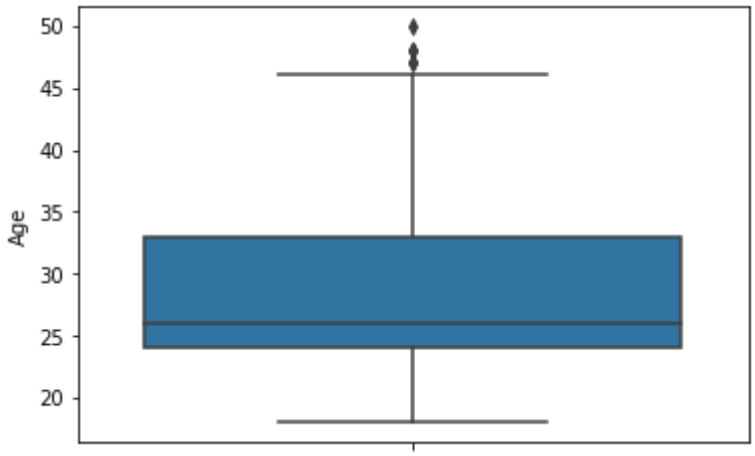
In [3]:
```
sns.histplot(x= 'Age', data = df, bins = 20)
```

Out[3]: <AxesSubplot:xlabel='Age', ylabel='Count'>



In [20]:
```
sns.boxplot(y= 'Age', data = df)
```

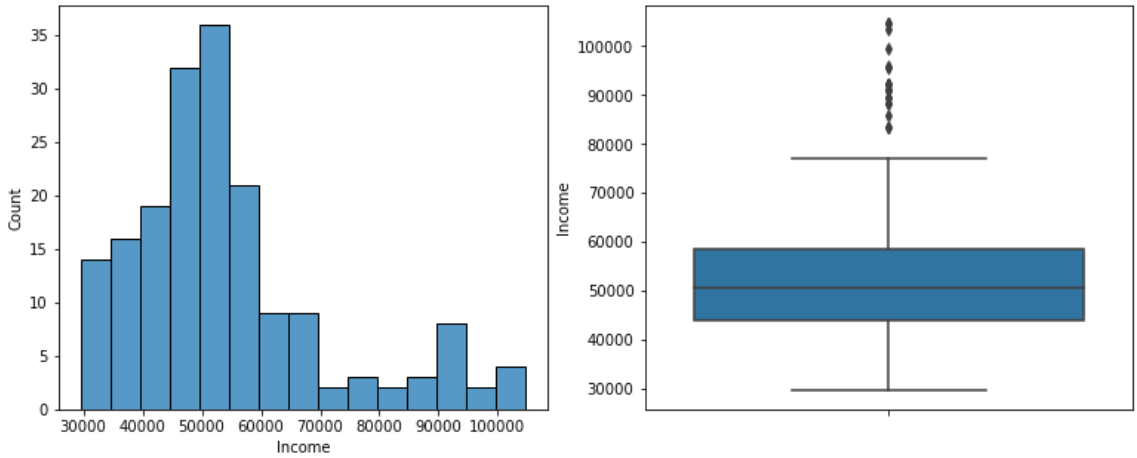
Out[20]: <AxesSubplot:ylabel='Age'>



In [14]:
```
fig = plt.figure(figsize = (13,5))
plt.subplot(1,2,1)
sns.histplot(x= 'Income', data = df)

plt.subplot(1,2,2)
sns.boxplot(y = 'Income', data = df)
```

Out[14]: <AxesSubplot:ylabel='Income'>



In [15]:
```
df['Income'].mean()
```

Out[15]: 53719.57777777778

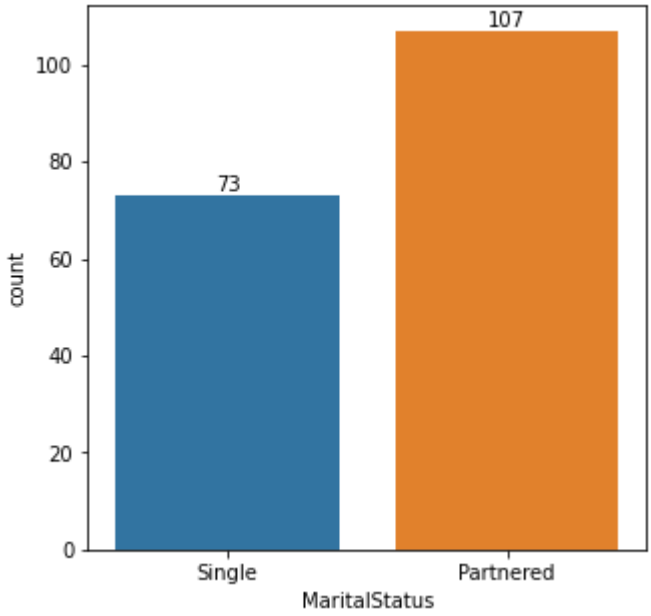
In [18]:
```
df['Income'].median()
```

Out[18]: 50596.5

In [17]:
```
df['Income'].max()
```

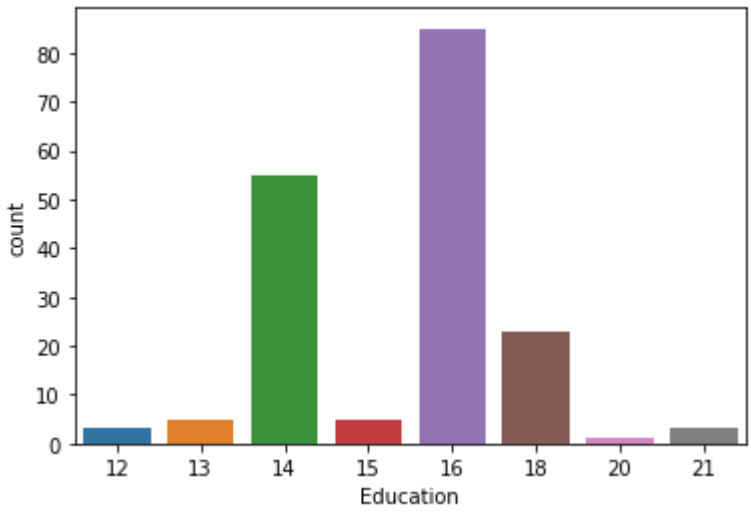
Out[17]: 104581

In [4]:
```
fig = plt.figure(figsize = (5,5))
ax = sns.countplot(x = 'MaritalStatus', data = df)
for i in ax.containers:
    ax.bar_label(i,)
plt.show()
```



In [27]:
```
sns.countplot(x = 'Education', data = df)
```

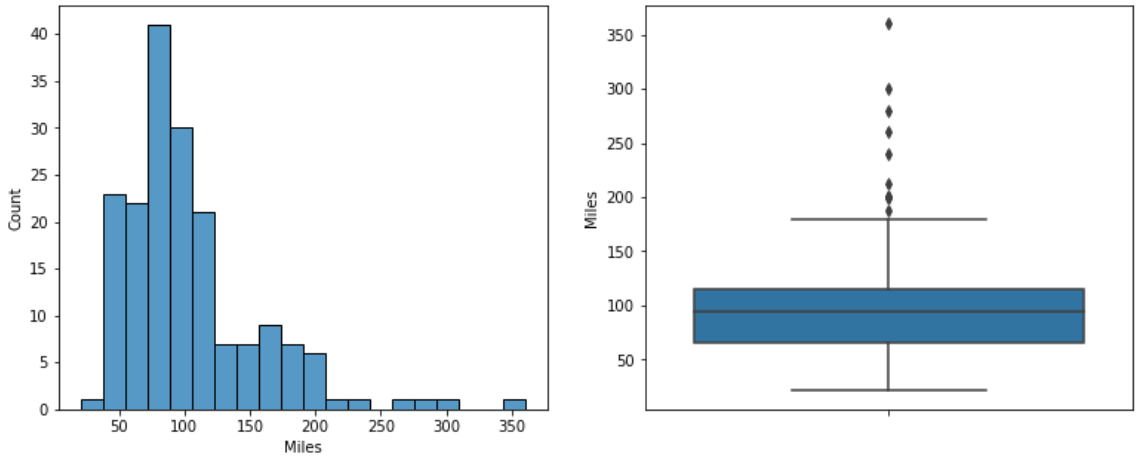
Out[27]:
```
<AxesSubplot:xlabel='Education', ylabel='count'>
```



In [20]:
```
fig = plt.figure(figsize  = (13,5))
plt.subplot(1,2,1)
sns.histplot(x = 'Miles', data = df)

plt.subplot(1,2,2)
sns.boxplot(y = 'Miles', data = df)
```

Out[20]: <AxesSubplot:ylabel='Miles'>



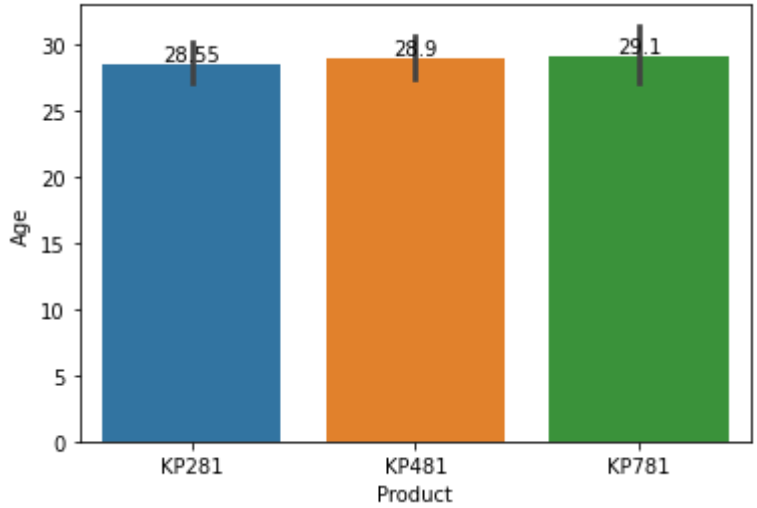
In [25]:
```
df['Miles'].median()
```

Out[25]: 94.0

In [30]:
```
# Bivariate Analysis
```

In [4]:
```
# Product and Age - Categorical Continous
```

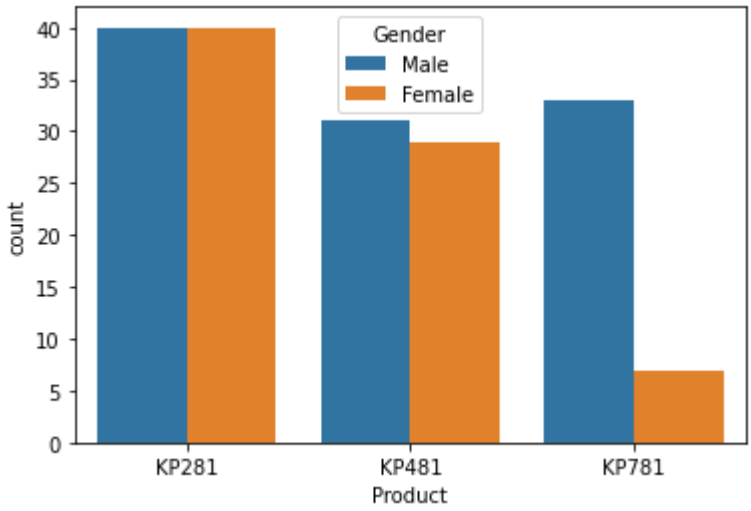
In [10]:
```
ax = sns.barplot(y = 'Age', x = 'Product', data = df)
ax.bar_label(ax.containers[0])
plt.show()
```



In [11]:
```
# Product and Gender - categorical, categorical
```

In [13]:
```
sns.countplot(x = 'Product', data = df, hue = 'Gender')
```

Out[13]:
```
<AxesSubplot:xlabel='Product', ylabel='count'>
```



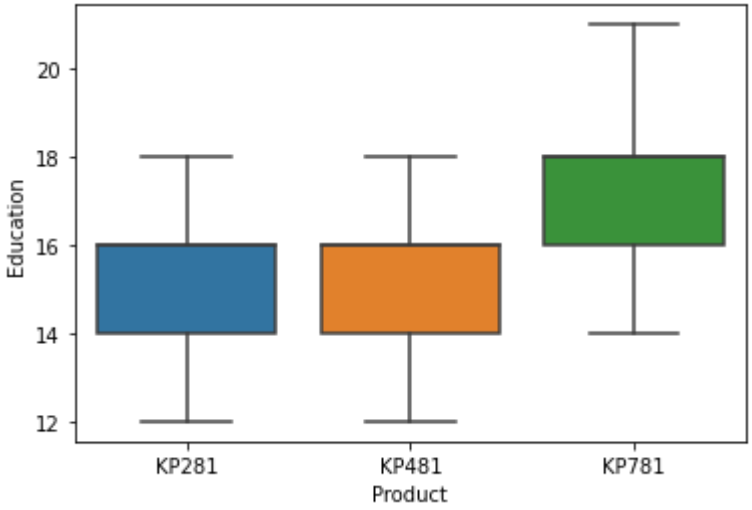
In [14]:
```
# Product and Education - categorical, continous
```

In [5]:
```
df['Education'].value_counts()
```

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

In [15]:
```
sns.boxplot(x = 'Product', y = 'Education', data = df)
```

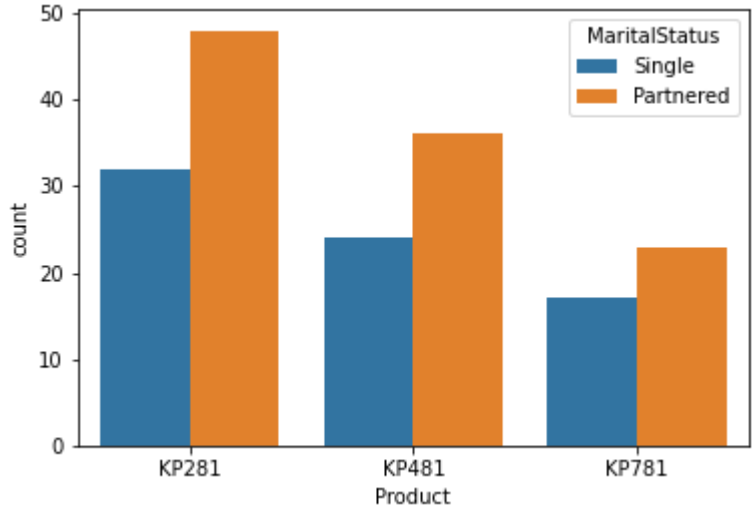
Out[15]:
```
<AxesSubplot:xlabel='Product', ylabel='Education'>
```



In [16]:
```
# Product and marital status - categorical categorical
```

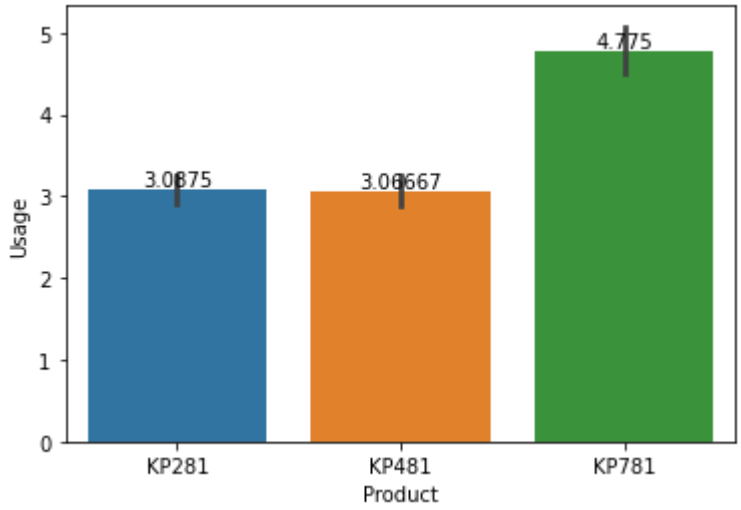
In [18]:
```
sns.countplot(x= 'Product', hue = 'MaritalStatus', data = df)
```

Out[18]:
```
<AxesSubplot:xlabel='Product', ylabel='count'>
```



In [19]:
```
# Product and Usage
```

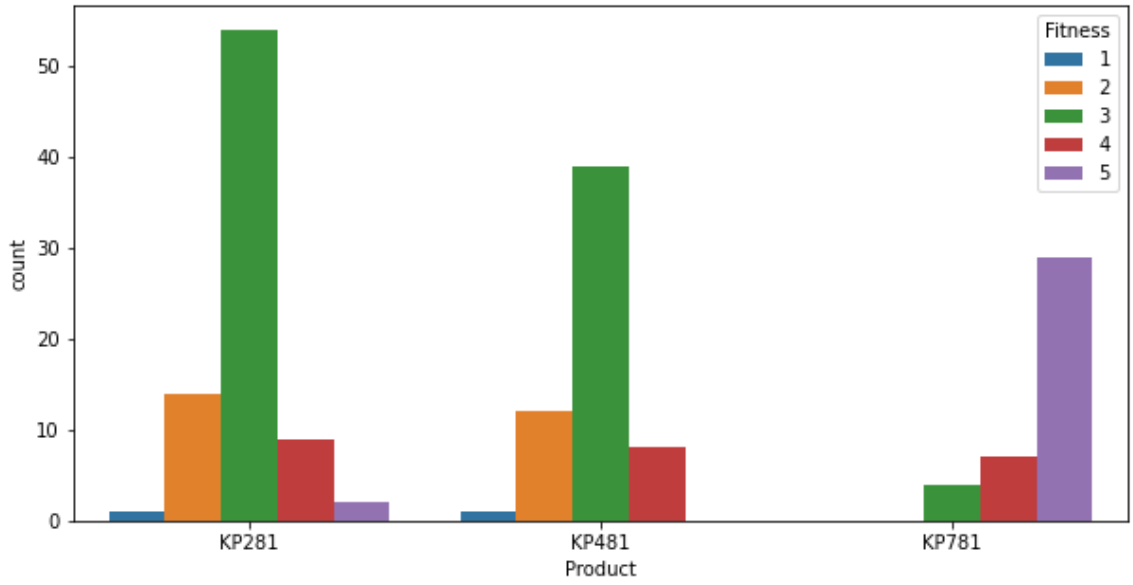
In [22]:
```
ax = sns.barplot(x= 'Product', y = 'Usage', data = df)
ax.bar_label(ax.containers[0])
plt.show()
```



In [23]:
```
# Product and Fitness - Categorical categorical
```

In [26]:
```
fig = plt.figure( figsize = (10,5))
sns.countplot(x= 'Product', hue = 'Fitness', data = df)
```

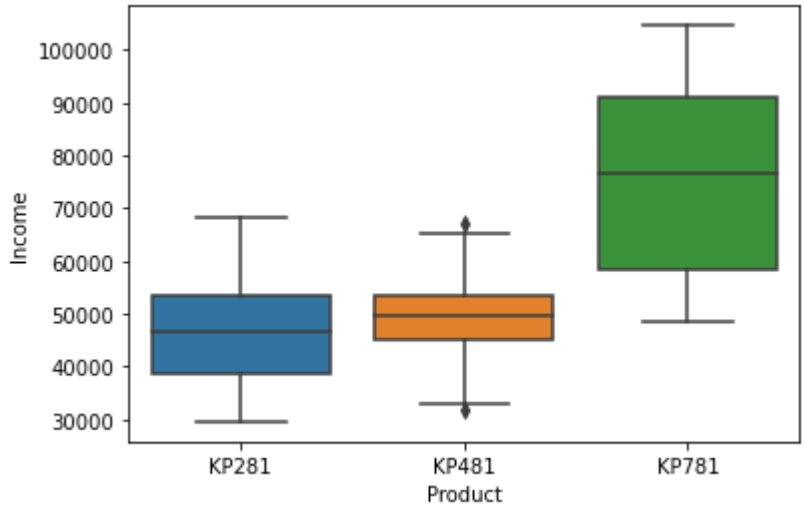
Out[26]:
```
<AxesSubplot:xlabel='Product', ylabel='count'>
```



In [26]:
```
# Product and Income
```

In [28]:
```
sns.boxplot(x = 'Product', y = 'Income', data = df)
```

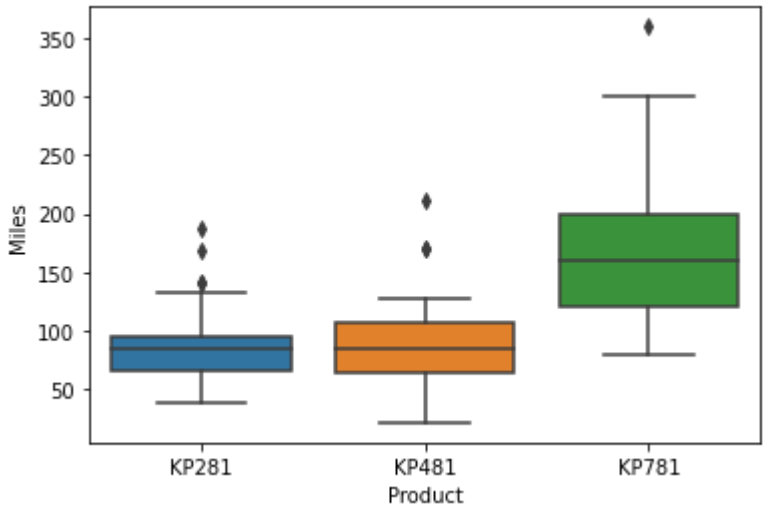
Out[28]:
```
<AxesSubplot:xlabel='Product', ylabel='Income'>
```



In [29]:
```
# product and miles
```

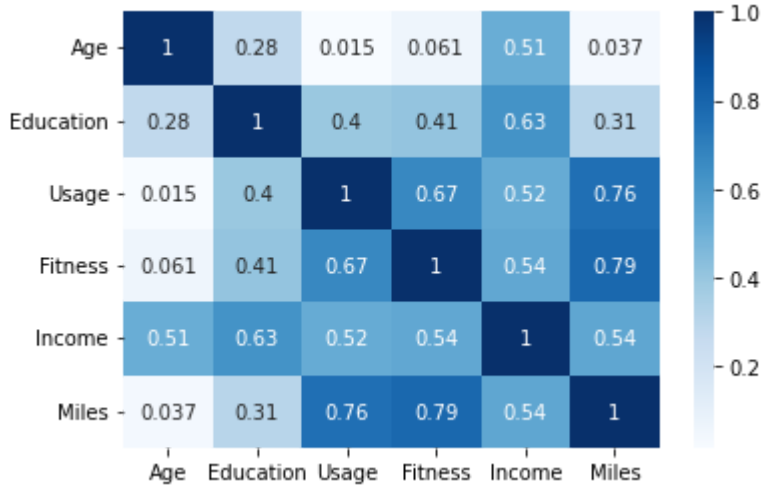
In [30]:
```
sns.boxplot(x = 'Product', y = 'Miles', data = df)
```

Out[30]: <AxesSubplot:xlabel='Product', ylabel='Miles'>



In [31]:
```
sns.heatmap(df.corr(), cmap ='Blues', annot =True)
```

Out[31]: <AxesSubplot:>

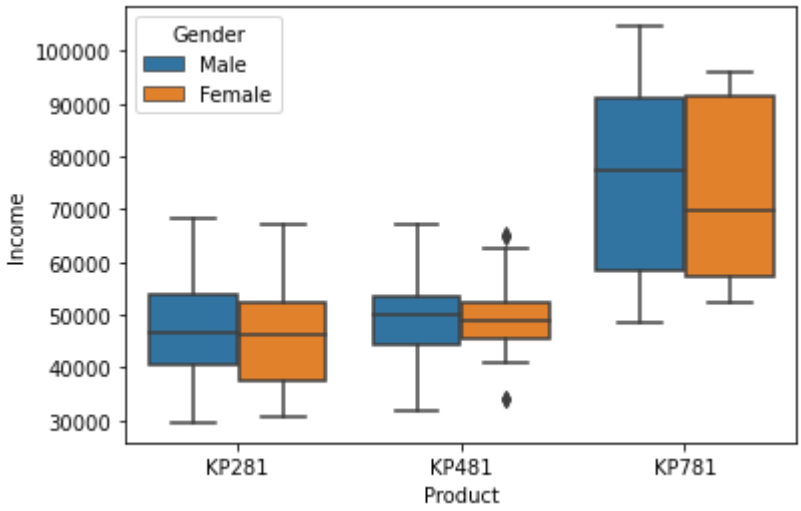


In [32]:
```
# Multivariate Analysis
```

In [34]:
```
# Product, gender and income
```

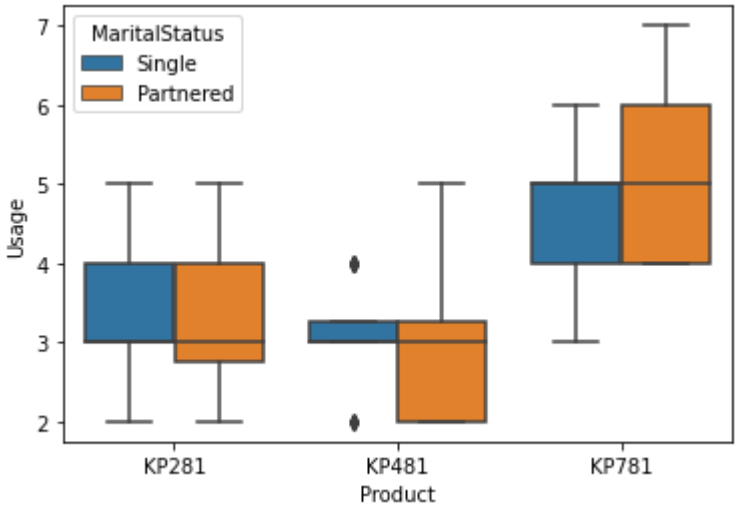
In [35]:
```
sns.boxplot(x='Product', y = 'Income', data = df, hue = 'Gender')
```

Out[35]:
```
<AxesSubplot:xlabel='Product', ylabel='Income'>
```



In [3]:
```
sns.boxplot(x='Product', y = 'Usage', data = df, hue = 'MaritalStatus')
```

Out[3]:
```
<AxesSubplot:xlabel='Product', ylabel='Usage'>
```



In [6]:
```
# Binning of Numerical data to partly treat outliers and for easier analysi
```

In [4]:
```
df['Age_bins']= pd.cut(df['Age'], bins = [10,20,30,40,50], labels = ['Till
df
```

Out[4]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_bi
0	KP281	18	Male	14	Single	3	4	29562	112	Till
1	KP281	19	Male	15	Single	2	3	31836	75	Till
2	KP281	19	Female	14	Partnered	4	3	30699	66	Till
3	KP281	19	Male	12	Single	3	3	32973	85	Till
4	KP281	20	Male	13	Partnered	4	2	35247	47	Till
...	...	...	...	...	...	...	...	...	...	
175	KP781	40	Male	21	Single	6	5	83416	200	30-40
176	KP781	42	Male	18	Single	5	4	89641	200	40-50
177	KP781	45	Male	16	Single	5	5	90886	160	40-50
178	KP781	47	Male	18	Partnered	4	5	104581	120	40-50
179	KP781	48	Male	18	Partnered	4	5	95508	180	40-50

180 rows × 10 columns

In [14]:
```
df['Income'].value_counts().sort_index()
```

Out[14]:
```
29562     1
30699     1
31836     2
32973     5
34110     5
         ..
95508     1
95866     1
99601     1
103336    1
104581    2
Name: Income, Length: 62, dtype: int64
```

In [5]:
```
df['Income_bins'] = pd.cut(df['Income'], bins = [25000,40000,60000,80000,20
                           labels = ['Low','Medium','High','Very High'])
```

In [6]:
```
df['Mile_bins']= pd.cut(df['Miles'], bins = [10,50,100,150,200,500], labels
df
```

Out[6]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_bi
0	KP281	18	Male	14	Single	3	4	29562	112	Till
1	KP281	19	Male	15	Single	2	3	31836	75	Till
2	KP281	19	Female	14	Partnered	4	3	30699	66	Till
3	KP281	19	Male	12	Single	3	3	32973	85	Till
4	KP281	20	Male	13	Partnered	4	2	35247	47	Till
...	...	...	...	...	...	...	...	...	...	
175	KP781	40	Male	21	Single	6	5	83416	200	30-40
176	KP781	42	Male	18	Single	5	4	89641	200	40-50
177	KP781	45	Male	16	Single	5	5	90886	160	40-50
178	KP781	47	Male	18	Partnered	4	5	104581	120	40-50
179	KP781	48	Male	18	Partnered	4	5	95508	180	40-50

180 rows × 12 columns

In [24]:
```
df['Education'].value_counts()
```

Out[24]:
```
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

In [7]:
```
df['Education_bins'] = pd.cut(df['Education'], bins = [10,15,18,22], labels
df
```

Out[7]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_bi
0	KP281	18	Male	14	Single	3	4	29562	112	Till
1	KP281	19	Male	15	Single	2	3	31836	75	Till
2	KP281	19	Female	14	Partnered	4	3	30699	66	Till
3	KP281	19	Male	12	Single	3	3	32973	85	Till
4	KP281	20	Male	13	Partnered	4	2	35247	47	Till
...	...	...	...	...	...	...	...	...	...	
175	KP781	40	Male	21	Single	6	5	83416	200	30-40
176	KP781	42	Male	18	Single	5	4	89641	200	40-50
177	KP781	45	Male	16	Single	5	5	90886	160	40-50
178	KP781	47	Male	18	Partnered	4	5	104581	120	40-50
179	KP781	48	Male	18	Partnered	4	5	95508	180	40-50

180 rows × 13 columns

In [27]:
```
from IPython.display import display
col_name = ['Gender','MaritalStatus','Education_bins','Age_bins','Income_bi
for i in col_name:
    print('Table for ',str(i),'vs Treadmill Product')
    display(pd.crosstab(index = df[i], columns = df['Product'], margins = T
    print('\n')
```

```
Table for  Gender vs Treadmill Product
```

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

```
Table for  MaritalStatus vs Treadmill Product
```

Product	KP281	KP481	KP781	All
MaritalStatus				
Partnered	48	36	23	107
Single	32	24	17	73

In [26]:
```
from IPython.display import display
col_name = ['Gender','MaritalStatus','Education_bins','Age_bins','Income_bi
for i in col_name:
    print('Table for ',str(i),'vs Treadmill Product')
    display(pd.crosstab(index = df[i], columns = df['Product'], margins = T
    print('\n')
```

```
Table for  Gender vs Treadmill Product
```

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

```
Table for  MaritalStatus vs Treadmill Product
```

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

In [8]:
```
# Assessing multivariate probablities
# Age and gender
```

In [13]:
```
pd.crosstab(index = [df.Gender,df.Age_bins], columns = df['Product'], margi
```

Out[13]:

Gender	Age_bins	KP281	KP481	KP781
Female	Till 20	0.666667	0.333333	0.000000
	20-30yrs	0.541667	0.333333	0.125000
	30-40yrs	0.409091	0.545455	0.045455
	40-50yrs	1.000000	0.000000	0.000000
Male	Till 20	0.571429	0.428571	0.000000
	20-30yrs	0.370968	0.241935	0.387097
	30-40yrs	0.384615	0.423077	0.192308
	40-50yrs	0.333333	0.222222	0.444444
All		0.444444	0.333333	0.222222

In [14]:
```
# Age, gender and usage
```

In [8]:
```
pd.crosstab(index = [df.Gender, df.Fitness], columns = df['Product'], margi
```

Out[8]:

Gender	Fitness	KP281	KP481	KP781
	1	0.000000	1.000000	0.000000
	2	0.625000	0.375000	0.000000
Female	3	0.577778	0.400000	0.022222
	4	0.375000	0.500000	0.125000
	5	0.166667	0.000000	0.833333
	1	1.000000	0.000000	0.000000
	2	0.400000	0.600000	0.000000
Male	3	0.538462	0.403846	0.057692
	4	0.375000	0.250000	0.375000
	5	0.040000	0.000000	0.960000
All		0.444444	0.333333	0.222222

In [16]:
```
pd.crosstab(index = [df.Gender,df.Age_bins, df.Income_bins], columns = df['
```

Out[16]:

Gender	Age_bins	Income_bins	KP281	KP481	KP781
Female	Till 20	Low	0.666667	0.333333	0.000000
	20-30yrs	Low	0.909091	0.090909	0.000000
		Medium	0.484848	0.454545	0.060606
		High	0.000000	0.000000	1.000000
		Very High	0.000000	0.000000	1.000000
	30-40yrs	Low	1.000000	0.000000	0.000000
		Medium	0.428571	0.571429	0.000000
		High	0.333333	0.666667	0.000000
		Very High	0.000000	0.000000	1.000000
	40-50yrs	Medium	1.000000	0.000000	0.000000
		High	1.000000	0.000000	0.000000
Male	Till 20	Low	0.571429	0.428571	0.000000
	20-30yrs	Low	0.600000	0.400000	0.000000
		Medium	0.444444	0.305556	0.250000
		High	0.111111	0.000000	0.888889
		Very High	0.000000	0.000000	1.000000
	30-40yrs	Medium	0.529412	0.470588	0.000000
		High	0.250000	0.750000	0.000000
		Very High	0.000000	0.000000	1.000000
	40-50yrs	Medium	0.600000	0.400000	0.000000
		Very High	0.000000	0.000000	1.000000
All			0.444444	0.333333	0.222222

In [9]:
```
pd.crosstab(index = [df.Gender,df.Education_bins], columns = df['Product'],
```

Out[9]:

Gender	Education_bins	KP281	KP481	KP781
Female	Less Edu	0.606061	0.393939	0.000000
	Avg Edu	0.476190	0.380952	0.142857
	High Edu	0.000000	0.000000	1.000000
Male	Less Edu	0.542857	0.400000	0.057143
	Avg Edu	0.318182	0.257576	0.424242
	High Edu	0.000000	0.000000	1.000000
All		0.444444	0.333333	0.222222

In [10]:
```
pd.crosstab(index = [df.Gender,df.MaritalStatus], columns = df['Product'],
```

Out[10]:

Gender	MaritalStatus	KP281	KP481	KP781
Female	Partnered	0.586957	0.326087	0.086957
	Single	0.433333	0.466667	0.100000
Male	Partnered	0.344262	0.344262	0.311475
	Single	0.441860	0.232558	0.325581
All		0.444444	0.333333	0.222222

In [12]:
```
pd.crosstab(index = [df.Education_bins,df.Income_bins], columns = df['Produ
```

Out[12]:

Education_bins	Income_bins	KP281	KP481	KP781
Less Edu	Low	0.680000	0.320000	0.000000
	Medium	0.536585	0.439024	0.024390
	High	0.000000	1.000000	0.000000
	Very High	0.000000	0.000000	1.000000
Avg Edu	Low	0.857143	0.142857	0.000000
	Medium	0.446154	0.400000	0.153846
	High	0.300000	0.300000	0.400000
	Very High	0.000000	0.000000	1.000000
High Edu	High	0.000000	0.000000	1.000000
	Very High	0.000000	0.000000	1.000000
All		0.444444	0.333333	0.222222

In [13]:
```
pd.crosstab(index = [df.Education_bins,df.Mile_bins], columns = df['Product
```

Out[13]:

	Product	KP281	KP481	KP781
Education_bins	Mile_bins			
Less Edu	0-50	0.875000	0.125000	0.000000
	50-100	0.578947	0.421053	0.000000
	100-150	0.555556	0.388889	0.055556
	150-200	0.000000	1.000000	0.000000
	200+	0.000000	0.500000	0.500000
Avg Edu	0-50	0.555556	0.444444	0.000000
	50-100	0.491228	0.403509	0.105263
	100-150	0.300000	0.300000	0.400000
	150-200	0.111111	0.000000	0.888889
	200+	0.000000	0.000000	1.000000
High Edu	50-100	0.000000	0.000000	1.000000
	150-200	0.000000	0.000000	1.000000
All		0.444444	0.333333	0.222222

In [14]:
```
pd.crosstab(index = [df.Education_bins,df.Income_bins,df.Mile_bins], column
```

Out[14]:

Education_bins	Income_bins	Mile_bins	KP281	KP481	KP781
Less Edu	Low	0-50	1.000000	0.000000	0.000000
		50-100	0.647059	0.352941	0.000000
		100-150	0.800000	0.200000	0.000000
		200+	0.000000	1.000000	0.000000
	Medium	0-50	0.833333	0.166667	0.000000
		50-100	0.550000	0.450000	0.000000
		100-150	0.461538	0.461538	0.076923
		150-200	0.000000	1.000000	0.000000
	High	50-100	0.000000	1.000000	0.000000
	Very High	200+	0.000000	0.000000	1.000000
Avg Edu	Low	0-50	0.500000	0.500000	0.000000
		50-100	1.000000	0.000000	0.000000
		100-150	1.000000	0.000000	0.000000
	Medium	0-50	0.600000	0.400000	0.000000
		50-100	0.463415	0.463415	0.073171
		100-150	0.384615	0.384615	0.230769
		150-200	0.333333	0.000000	0.666667
	High	0-50	0.500000	0.500000	0.000000
		50-100	0.454545	0.363636	0.181818
		100-150	0.000000	1.000000	0.000000
		150-200	0.000000	0.000000	1.000000
		200+	0.000000	0.000000	1.000000
	Very High	50-100	0.000000	0.000000	1.000000
		100-150	0.000000	0.000000	1.000000
		150-200	0.000000	0.000000	1.000000
		200+	0.000000	0.000000	1.000000
High Edu	High	50-100	0.000000	0.000000	1.000000
		150-200	0.000000	0.000000	1.000000
	Very High	50-100	0.000000	0.000000	1.000000
		150-200	0.000000	0.000000	1.000000
All			0.444444	0.333333	0.222222

(Column header "Product" spans KP281, KP481, KP781.)

In [ ]: