Aerofit Business Case - Descriptive Statistics

About Aerofit:

Aerofit is a leading brand in the field of fitness equipment. Aerofit provides a product range including machines such as treadmills, exercise bikes, gym equipment, and fitness accessories to cater to the needs of all categories of people.

Problem Statement:

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to the new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

- Perform descriptive analytics to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts.
- For each AeroFit treadmill product, construct two-way contingency tables and compute all conditional and marginal probabilities along with their insights/impact on the business.

Column Profiling:

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

Importing Libraries and Dataset

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
data=pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749")
```

```
data.head()
```

	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

Checking the structure & characteristics of the dataset

```
#finding the shape of the dataset
data.shape
```

```
(180, 9)
```

```
#Finding the datatypes of the columns
data.dtypes
```

	0
Product	object
Age	int64
Gender	object
Education	int64
MaritalStatus	object
Usage	int64
Fitness	int64
Income	int64
Miles	int64

dtype: object

```
#Getting the information regarding the count of non-null values
data.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
```

```
#Checking the null values for each column
data.isna().sum()
```

	0
Product	0
Age	0
Gender	0
Education	0
MaritalStatus	0
Usage	0
Fitness	0
Income	0
Miles	0

dtype: int64

```
#Getting the count unique values in each column
data.nunique()
```

	0
Product	3
Age	32
Gender	2
Education	8
MaritalStatus	2
Usage	6
Fitness	5
Income	62
Miles	37

dtype: int64

Analyzing value counts

```
data['Gender'].value_counts()
```

Gender	count
Male	104
Female	76

dtype: int64

```
data['Product'].value_counts()
```

Product	count
KP281	80
KP481	60
KP781	40

dtype: int64

```
data['Education'].value_counts()
```

Education	count
16	85
14	55
18	23
15	5
13	5
12	3
21	3
20	1

dtype: int64

```
data['MaritalStatus'].value_counts()
```

MaritalStatus	count
Partnered	107
Single	73

dtype: int64

```
data['Usage'].value_counts()
```

Usage	count
3	69
4	52
2	33
5	17
6	7
7	2

dtype: int64

```
data['Fitness'].value_counts()
```

Fitness	count
3	97
5	31
2	26
4	24
1	2

dtype: int64

```
data['Miles'].value_counts()
```

	count
Miles	
85	27
95	12
66	10
75	10
47	9
106	9
94	8
113	8
53	7
100	7
180	6
200	6
56	6
64	6
127	5
160	5
42	4
150	4
38	3
74	3
170	3
120	3
103	3
132	2
141	2
280	1
260	1
300	1
240	1
112	1
212	1
80	1
140	1
21	1
169	1
188	1
360	1

dtype: int64

Outlier Detection

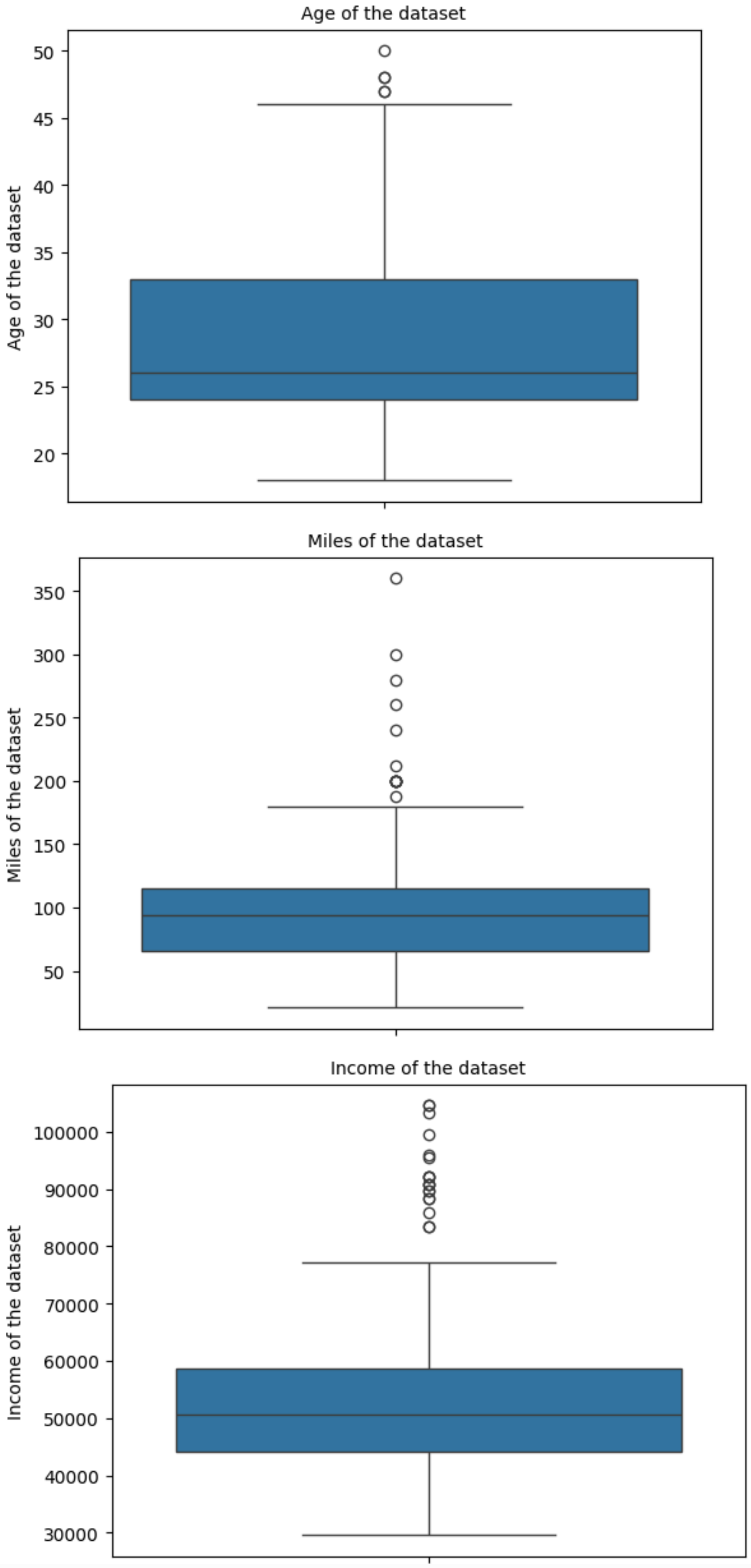
Outliers can be resolved using binning which is used in the following discussion

```
data.describe()
```

	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

```
Num_cols=['Age','Miles','Income']
```

```
for i in Num_cols:
  sns.boxplot(y = data[i])
  plt.yticks(fontsize=10)
  plt.ylabel(f"{i} of the dataset", fontsize=10)
  plt.title(f"{i} of the dataset", fontsize=10)
  plt.show()
```

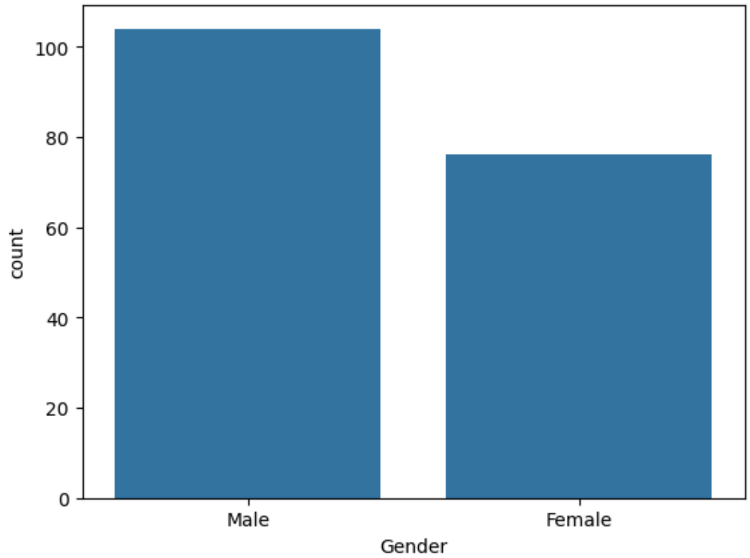



```
data.describe()
```

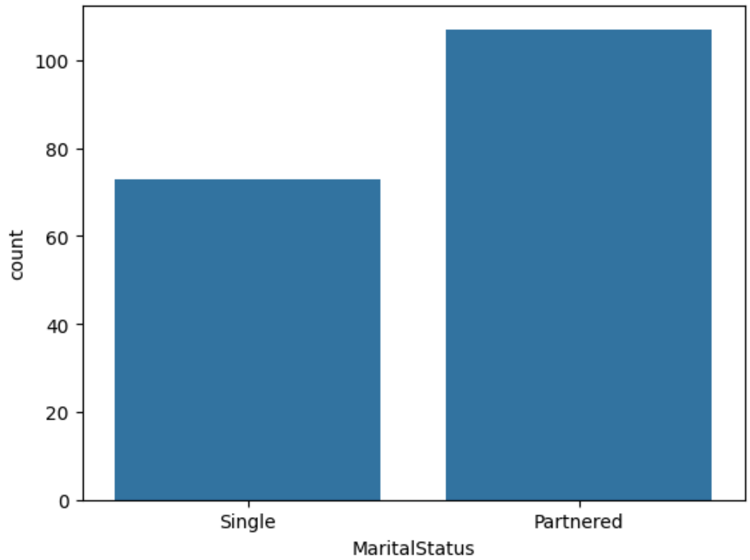
	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
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

Univariate Analysis

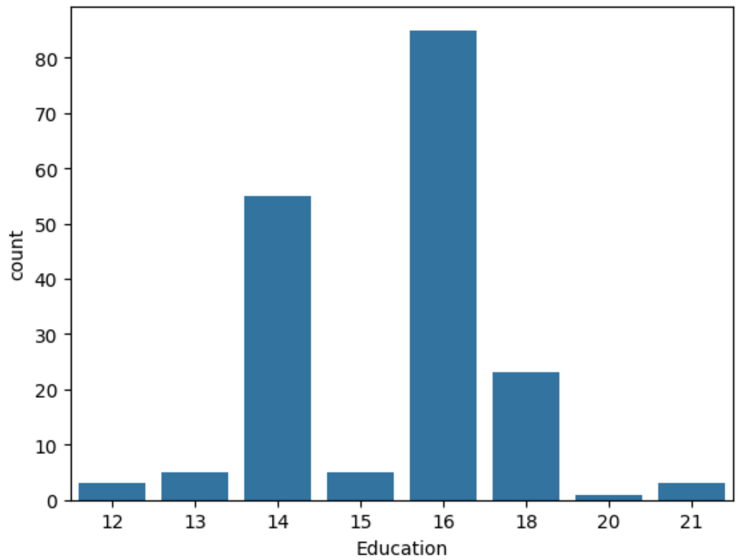
```
# For categorical variables
cat_cols=['Gender','MaritalStatus','Education','Usage','Fitness','Product']
for i in cat_cols:
  sns.countplot(x=i,data=data)
  plt.show()
```



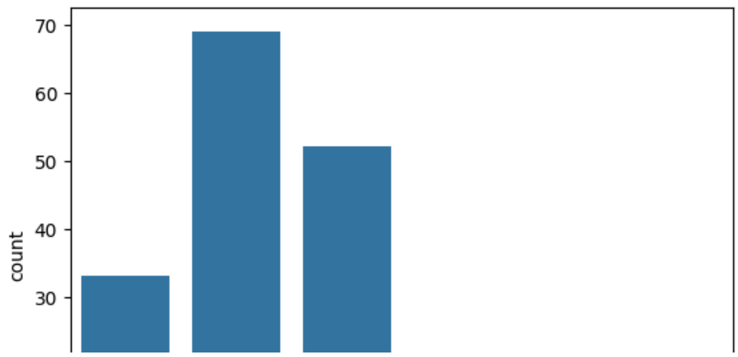
count
Male
Female
Gender



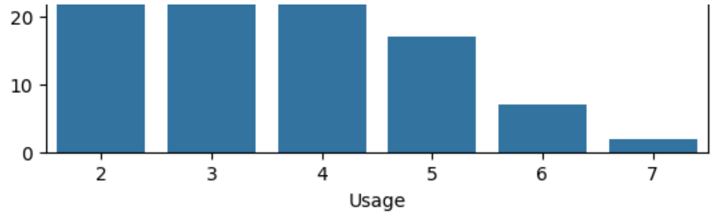
count
Single
Partnered
MaritalStatus

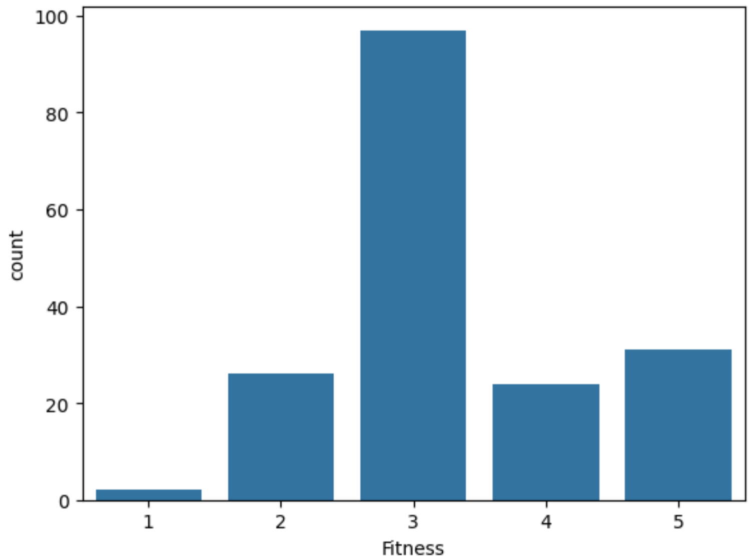


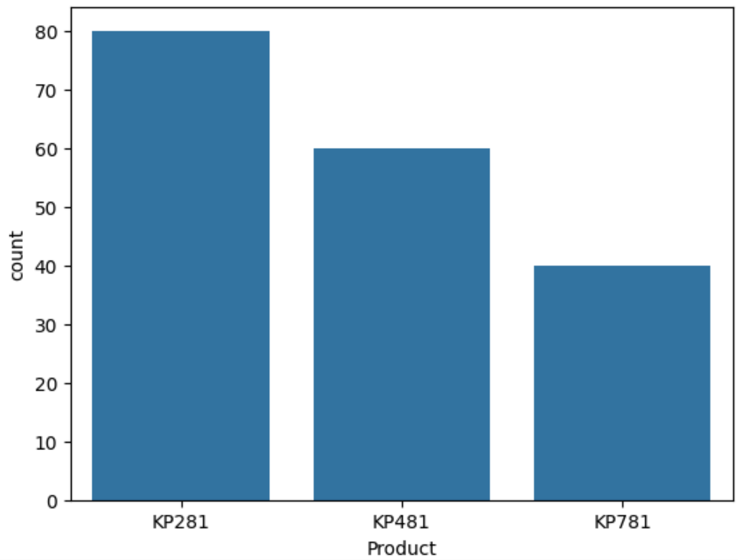
count
Education



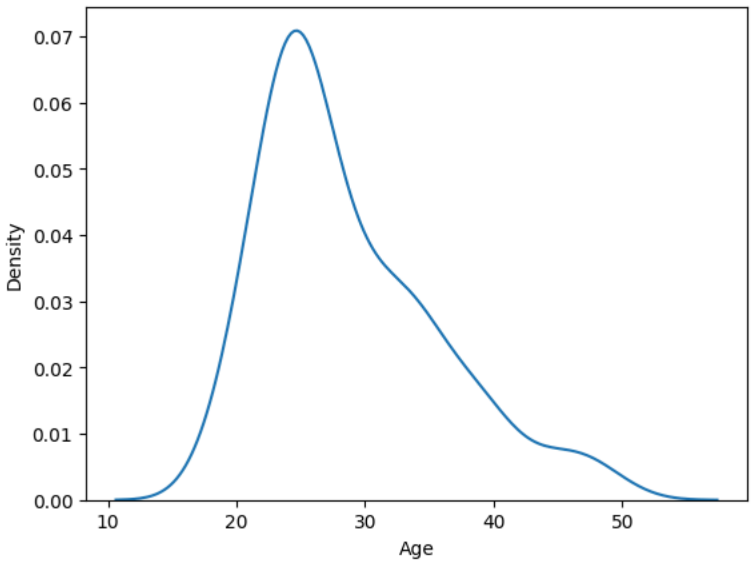
count

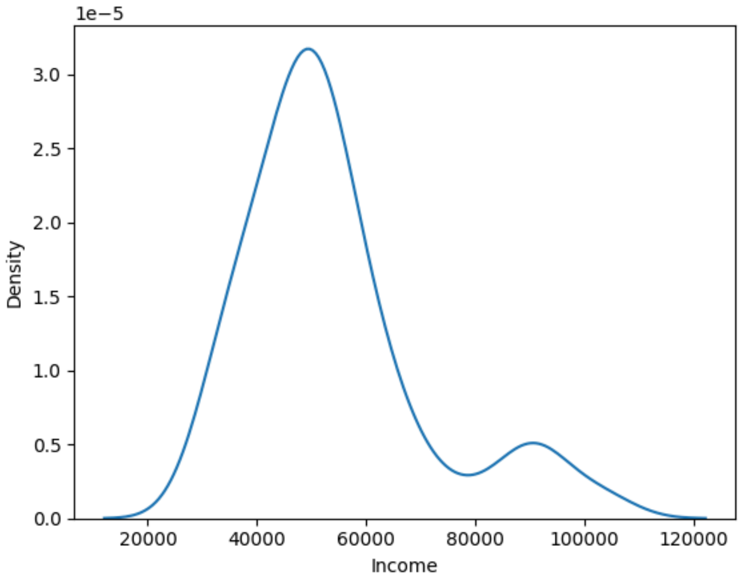

20
10
0
2
3
4
5
6
7
Usage

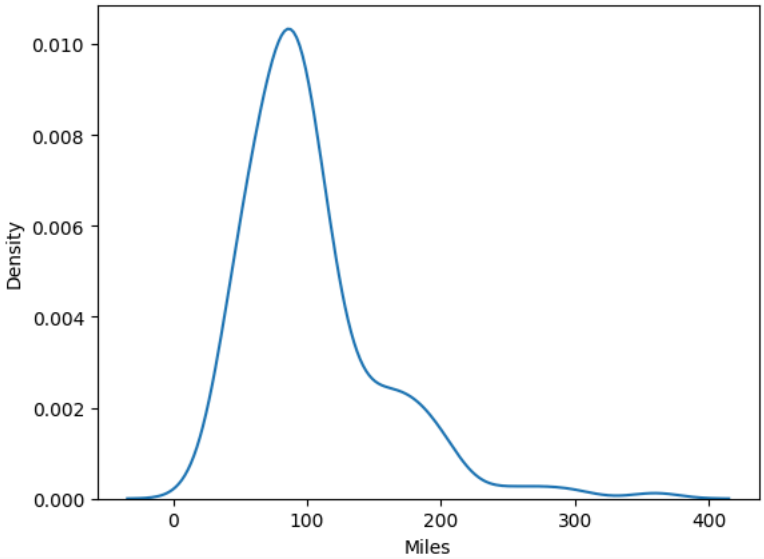

count
100
80
60
40
20
0
1
2
3
4
5
Fitness


count
80
70
60
50
40
30
20
10
0
KP281
KP481
KP781
Product

```
# For Numerical variables
num_cols=['Age','Income','Miles']
for i in num_cols:
  sns.kdeplot(data[i])
  plt.show()
```

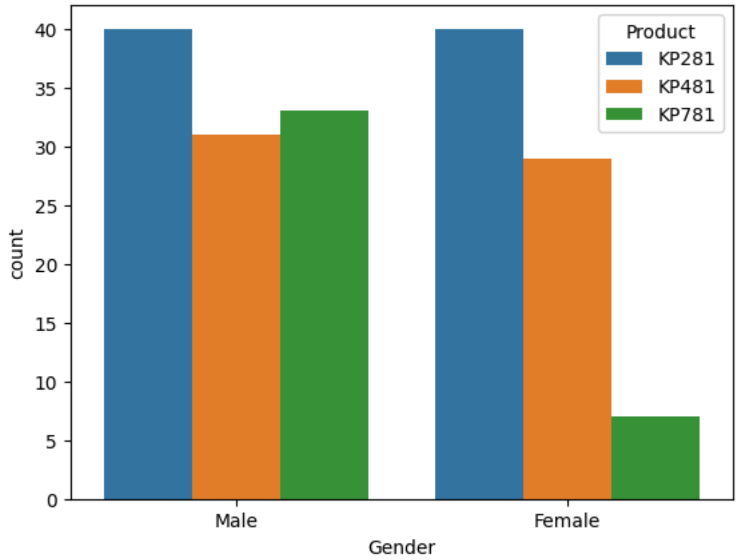




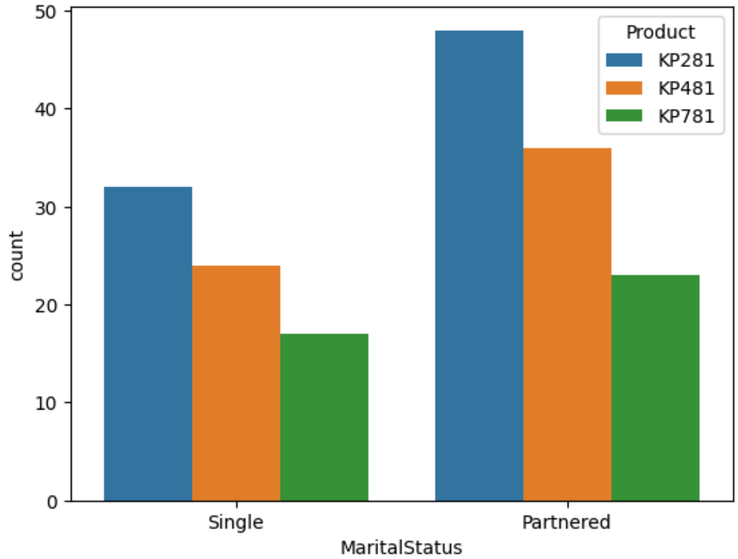


Bivariate Analysis

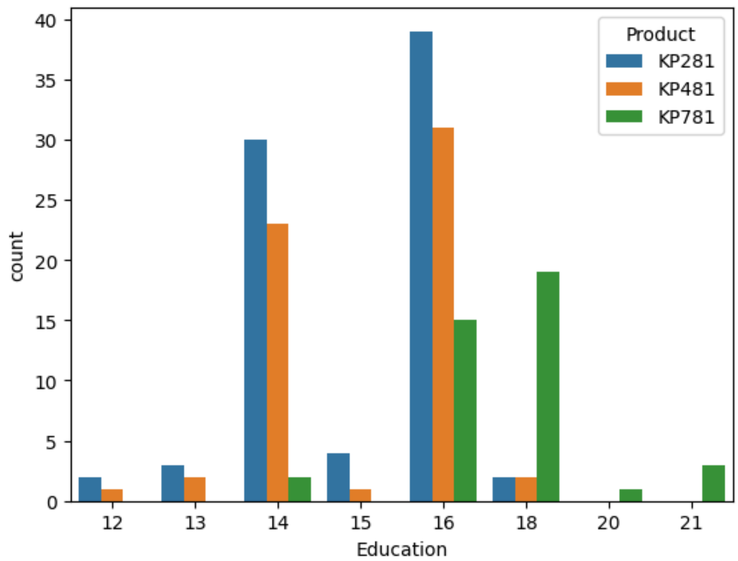
```
#Bivariate Analysis fo categorical using values
cat_cols=['Gender','MaritalStatus','Education','Usage','Fitness']
for i in cat_cols:
  sns.countplot(x=i,hue='Product',data=data)
  plt.show()
```



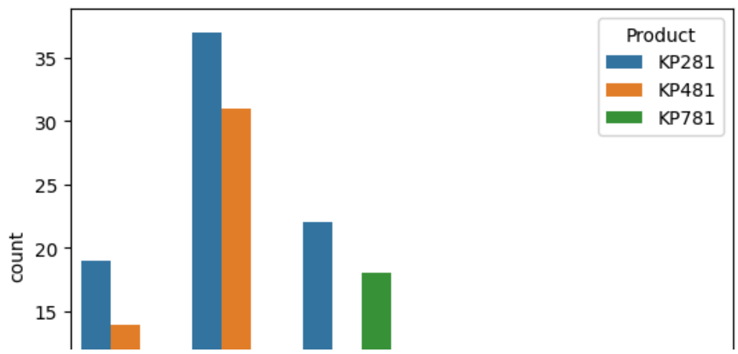
Product
KP281
KP481
KP781
count
Male
Female
Gender



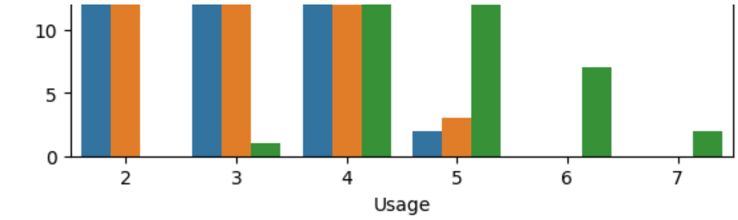
Product
KP281
KP481
KP781
count
Single
Partnered
MaritalStatus

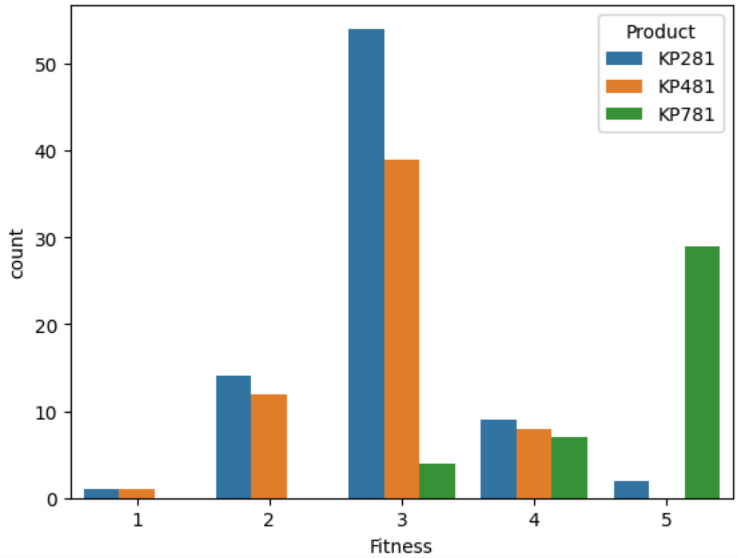


Product
KP281
KP481
KP781
count
Education

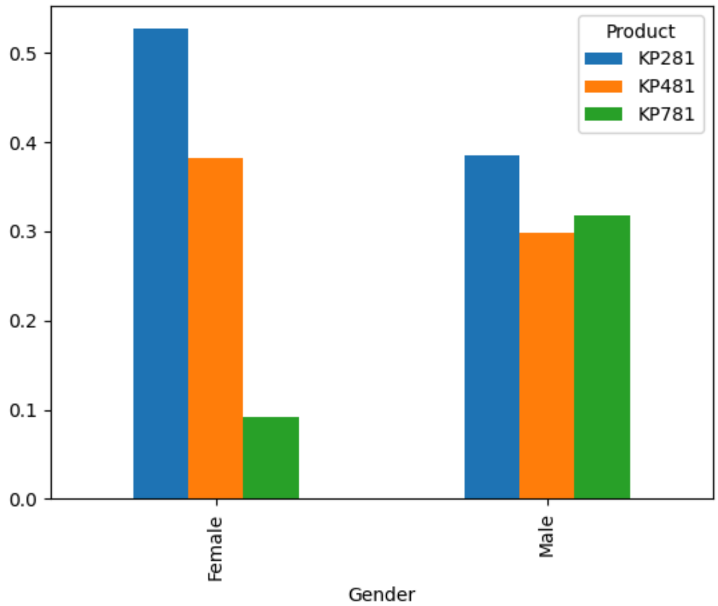


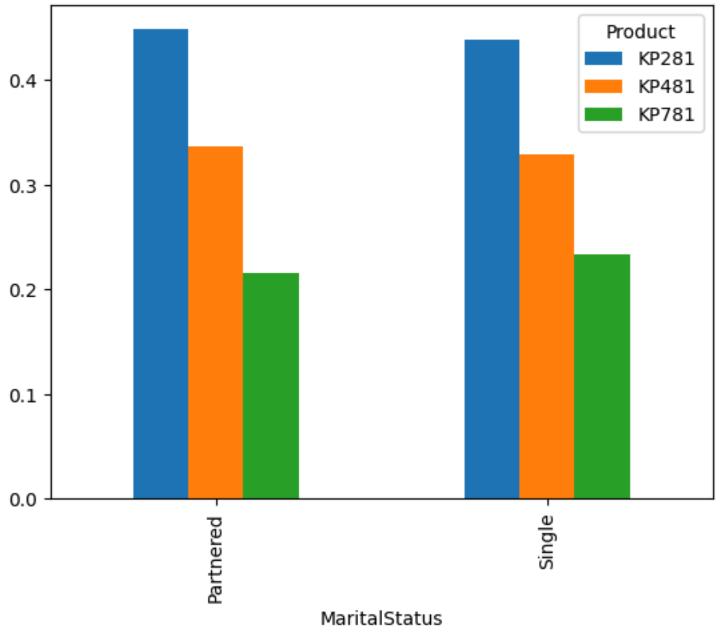
Product
KP281
KP481
KP781
count

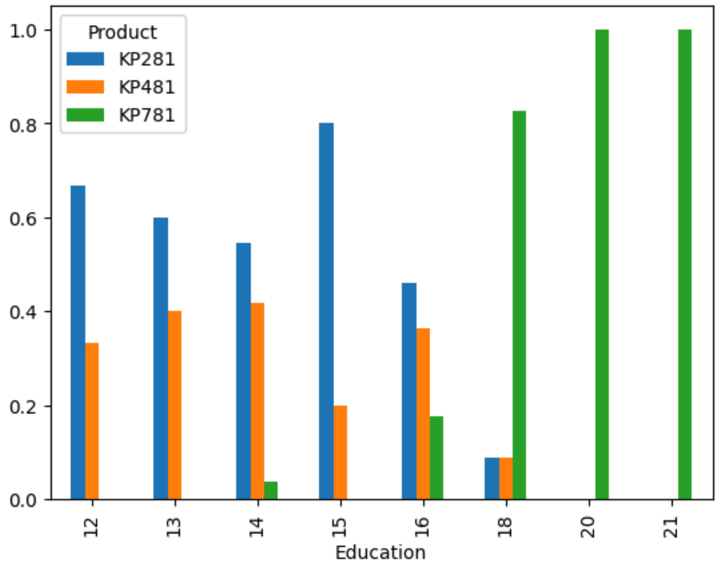


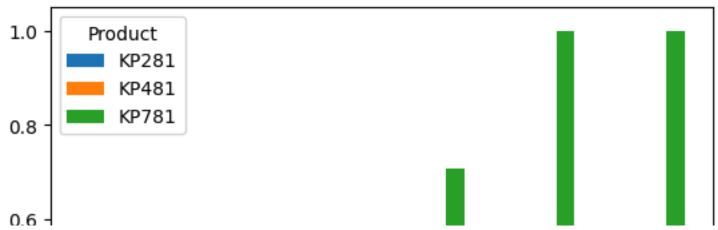


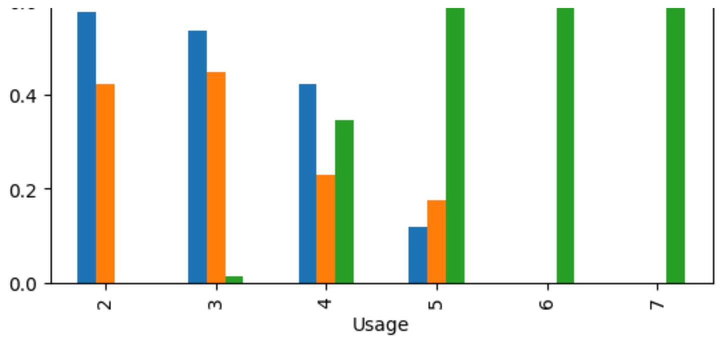

```
#Bivariate Analysis fo categorical using Propotions/Percentage
for i in cat_cols:
   i=pd.crosstab(data[i],data['Product'],normalize='index')
   i.plot(kind='bar')
   plt.show()
```

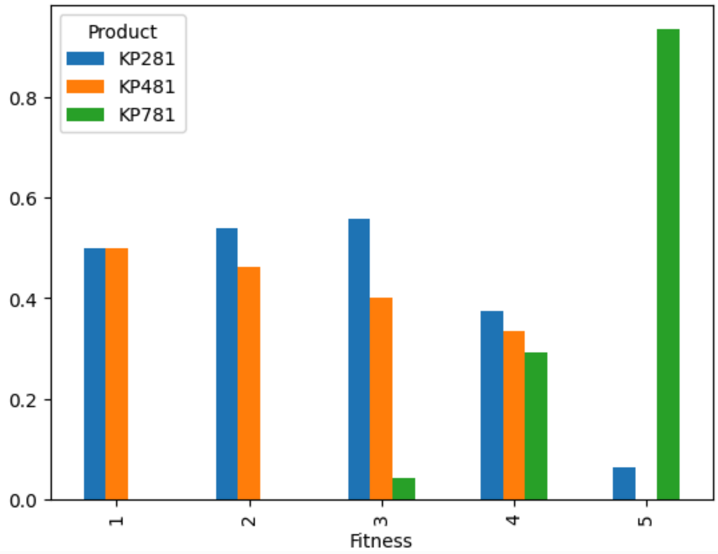

Product
KP281
KP481
KP781
0.0
0.1
0.2
0.3
0.4
0.5
Female
Male
Gender


Product
KP281
KP481
KP781
0.0
0.1
0.2
0.3
0.4
Partnered
Single
MaritalStatus

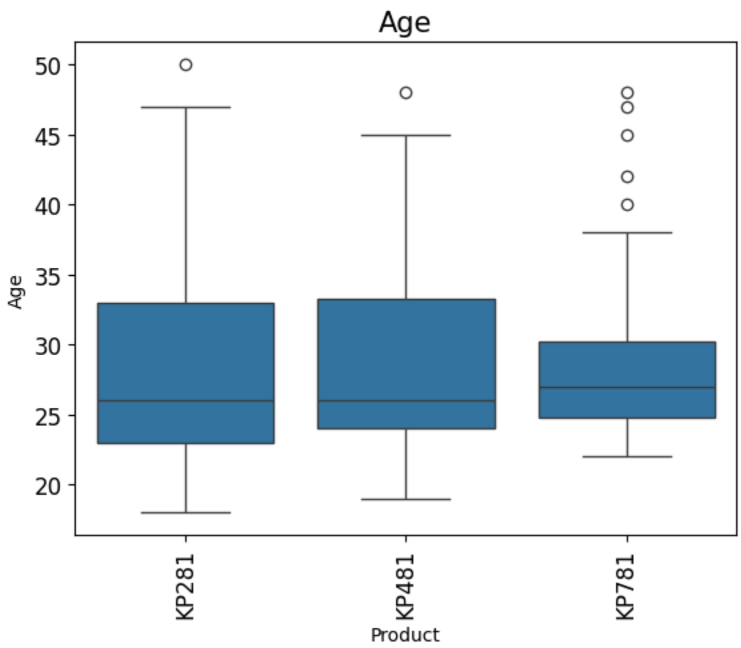

Product
KP281
KP481
KP781
0.0
0.2
0.4
0.6
0.8
1.0
12
13
14
15
16
18
20
21
Education

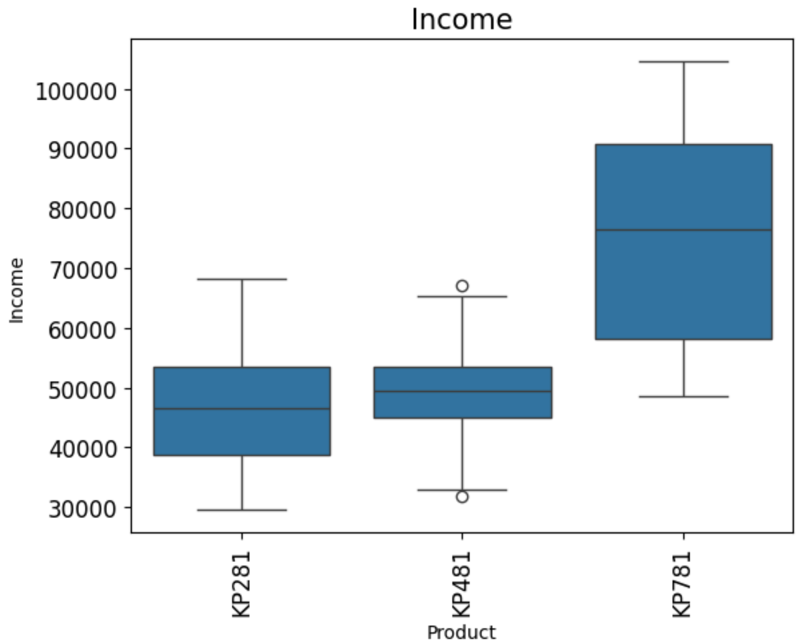

Product
KP281
KP481
KP781
0.6
0.8
1.0

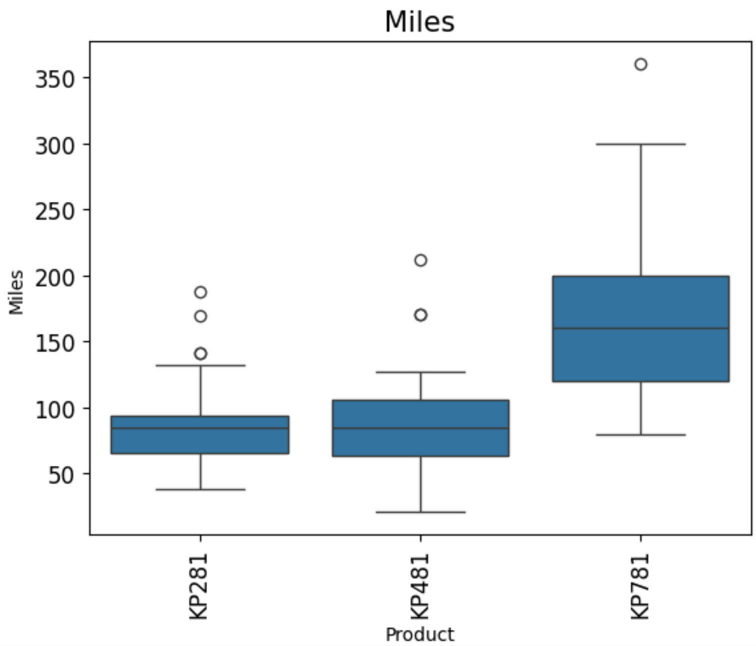





```
#Bivariate Analysis fo Numerical variables using values
num_cols=['Age','Income','Miles']
for i in num_cols:
  sns.boxplot(x='Product', y=i, data=data)
  plt.xticks(rotation=90,fontsize=12)
  plt.yticks(fontsize=12)
  plt.title(i, fontsize=15)
  plt.show()
```





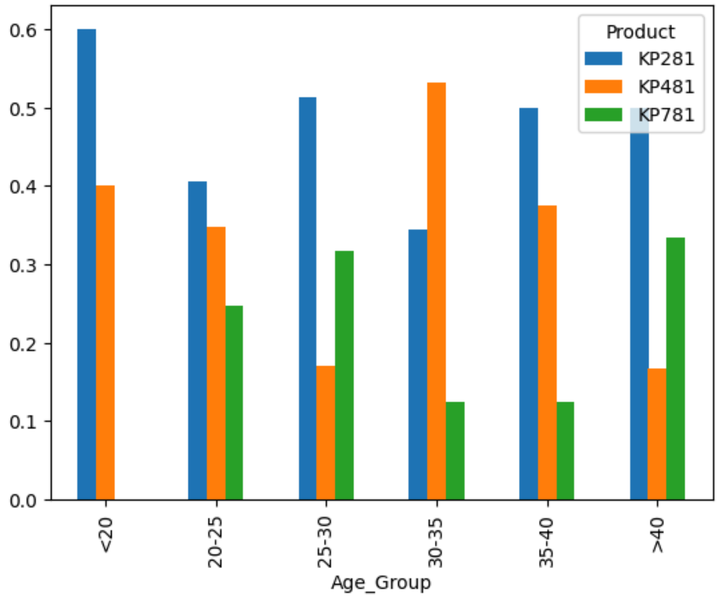


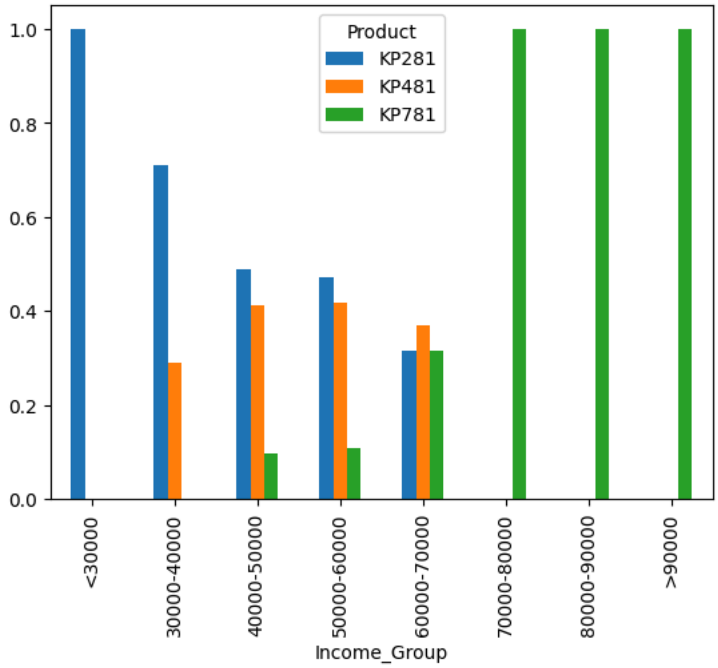

```
data_copy=data.copy()
```

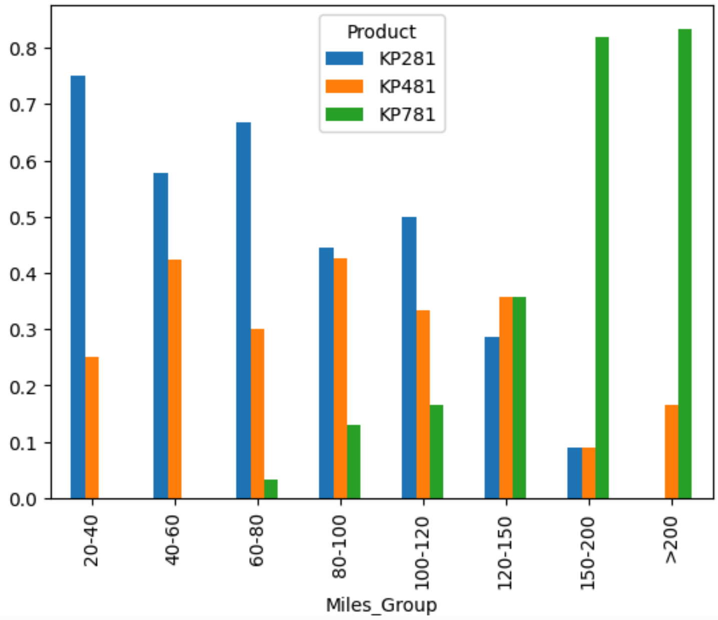
```
# Binning numerical columns to categorical
bins=[0,30000,40000,50000,60000,70000,80000,90000,120000]
labels=['<30000','30000-40000','40000-50000','50000-60000','60000-70000','70000-80000','80000-90000','>90000']
data['Income_Group']=pd.cut(data['Income'],bins=bins,labels=labels)
data.head()
bins=[0,20,25,30,35,40,51]
labels=['<20','20-25','25-30','30-35','35-40','>40']
data['Age_Group']=pd.cut(data['Age'],bins=bins,labels=labels)
data.head()
bins=[0,20,40,60,80,100,120,150,200,360]
labels=['<20','20-40','40-60','60-80','80-100','100-120','120-150','150-200','>200']
data['Miles_Group']=pd.cut(data['Miles'],bins=bins,labels=labels)
data.head()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Income_Group	Age_Group	Miles_Group
0	KP281	18	Male	14	Single	3	4	29562	112	<30000	<20	100-120
1	KP281	19	Male	15	Single	2	3	31836	75	30000-40000	<20	60-80
2	KP281	19	Female	14	Partnered	4	3	30699	66	30000-40000	<20	60-80
3	KP281	19	Male	12	Single	3	3	32973	85	30000-40000	<20	80-100
4	KP281	20	Male	13	Partnered	4	2	35247	47	30000-40000	<20	40-60

```
#Bivariate Analysis fo Numerical variables using its propotion/percentage
num_cols=['Age_Group','Income_Group','Miles_Group']
for i in num_cols:
   i=pd.crosstab(data[i],data['Product'],normalize='index')
   i.plot(kind='bar')
   plt.show()
```

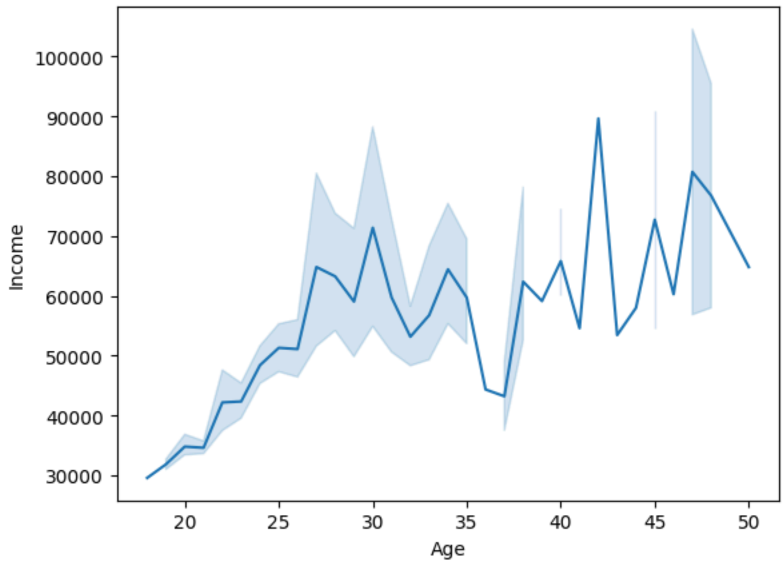


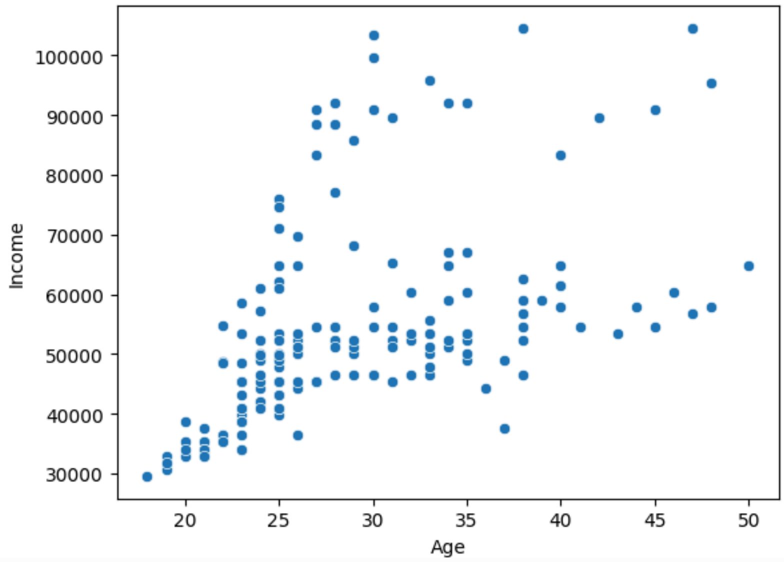




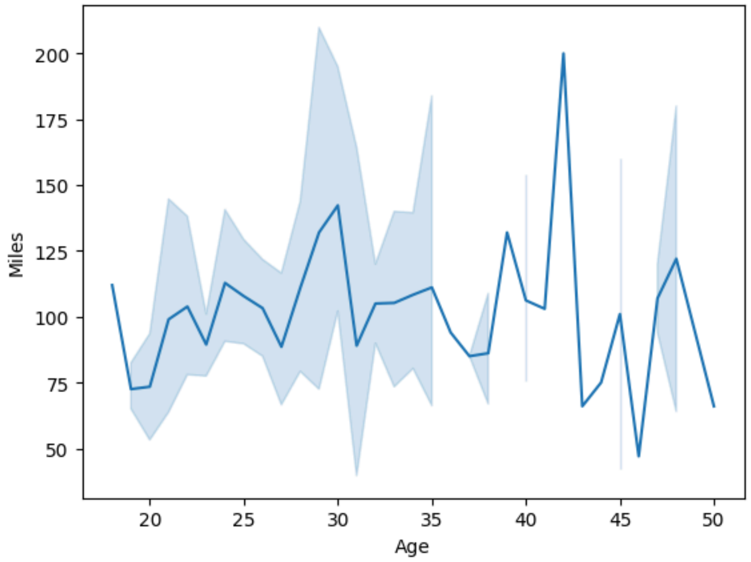

```
sns.lineplot(x='Age', y='Income', data=data)
plt.show()
```

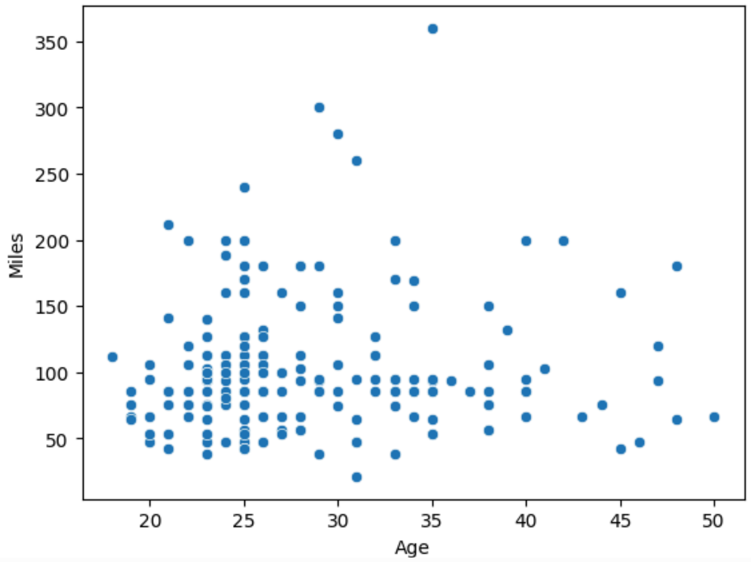
```
sns.scatterplot(x='Age', y='Income', data=data)
plt.show()
```



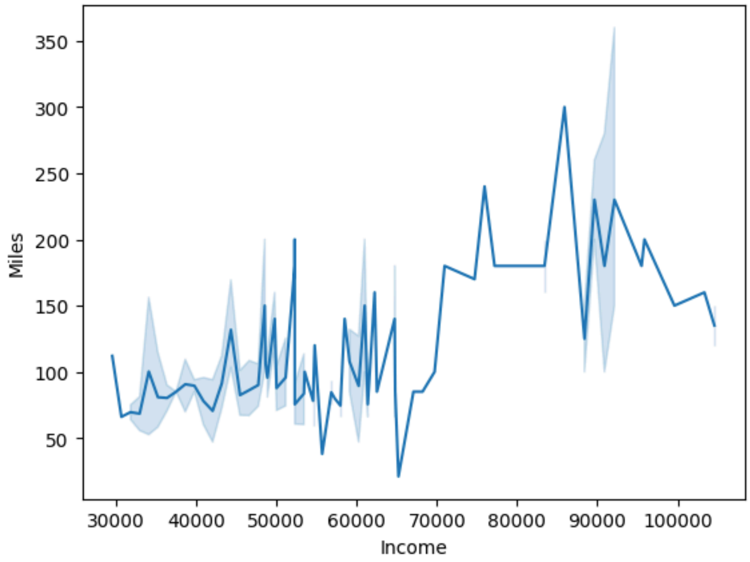


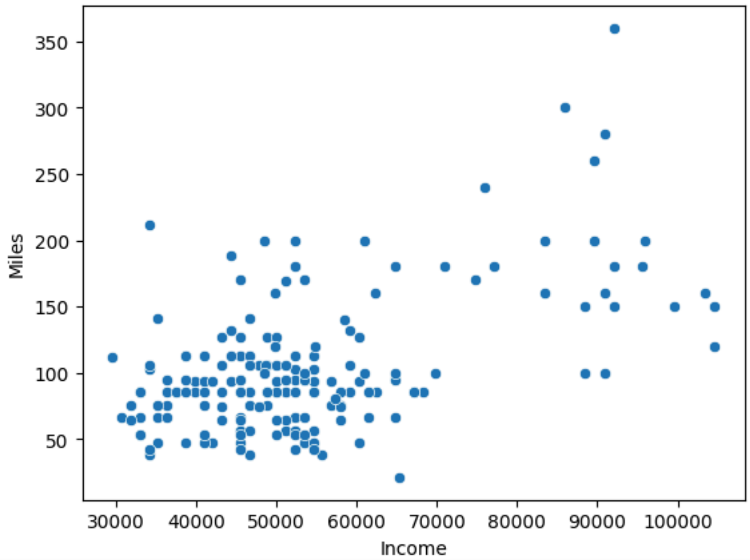

```
sns.lineplot(x='Age', y='Miles', data=data)
plt.show()

sns.scatterplot(x='Age', y='Miles', data=data)
plt.show()
```






```
sns.lineplot(x='Income', y='Miles', data=data)
plt.show()

sns.scatterplot(x='Income', y='Miles', data=data)
plt.show()
```





Multivariate

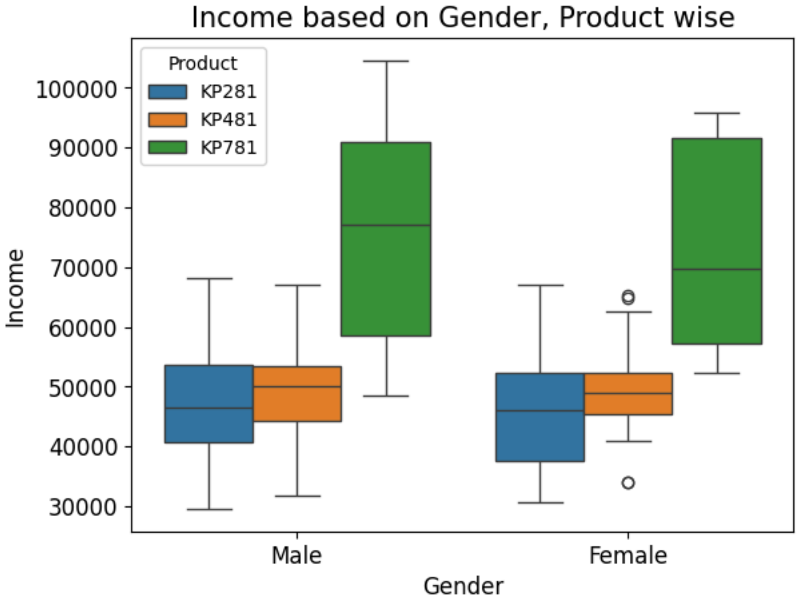
Multivariate analysis of Product on the basis of income.

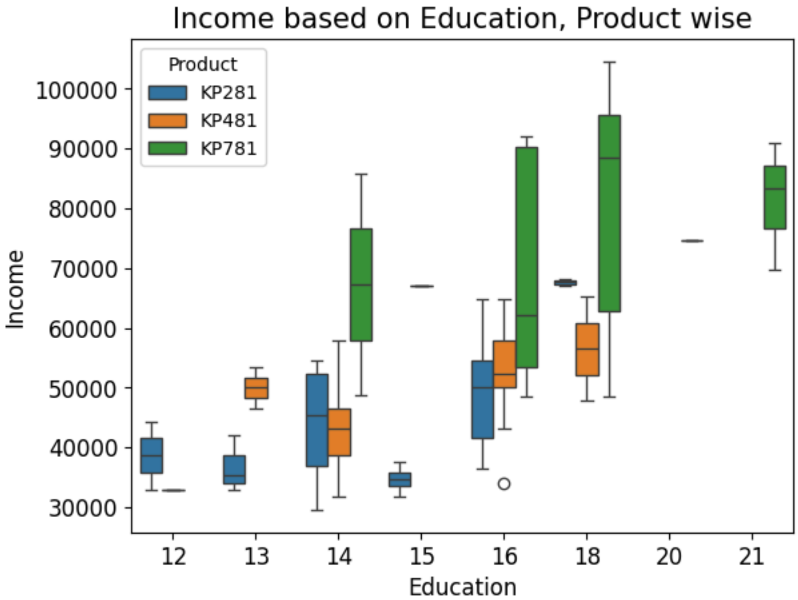
```
data.columns
```

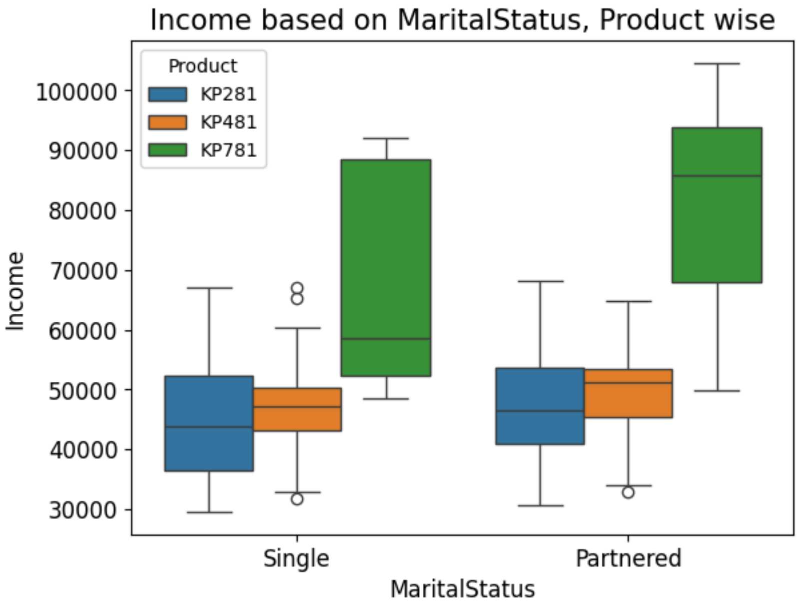
```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles', 'Income_Group', 'Age_Group',
       'Miles_Group'],
      dtype='object')
```

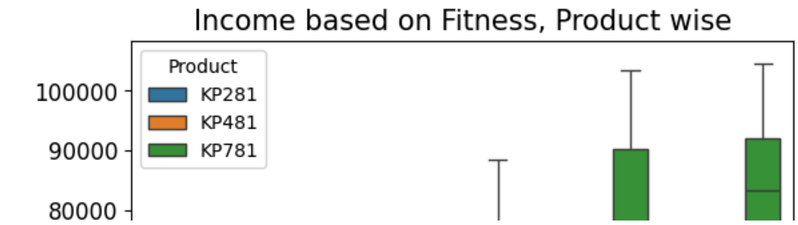
```
catcols=['Gender','Education','MaritalStatus','Fitness','Usage']
```

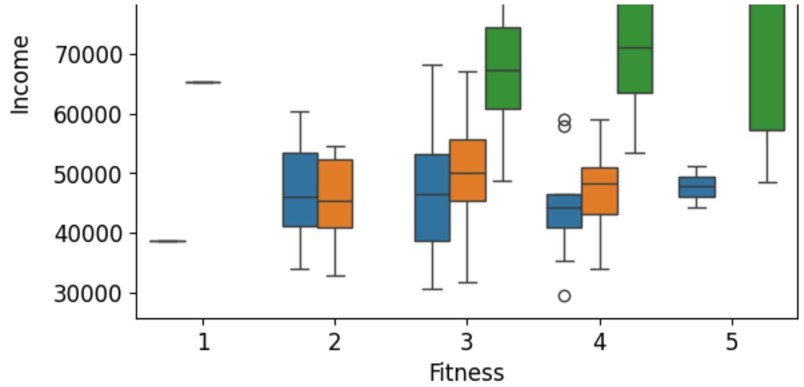
```
for i in catcols:
  sns.boxplot(x=i,y='Income',hue='Product',data=data)
  plt.xlabel(i, fontsize=12)
  plt.ylabel('Income', fontsize=12)
  plt.xticks(fontsize=12)
  plt.yticks(fontsize=12)
  plt.title(f'Income based on {i}, Product wise', fontsize=15)
  plt.show()
```

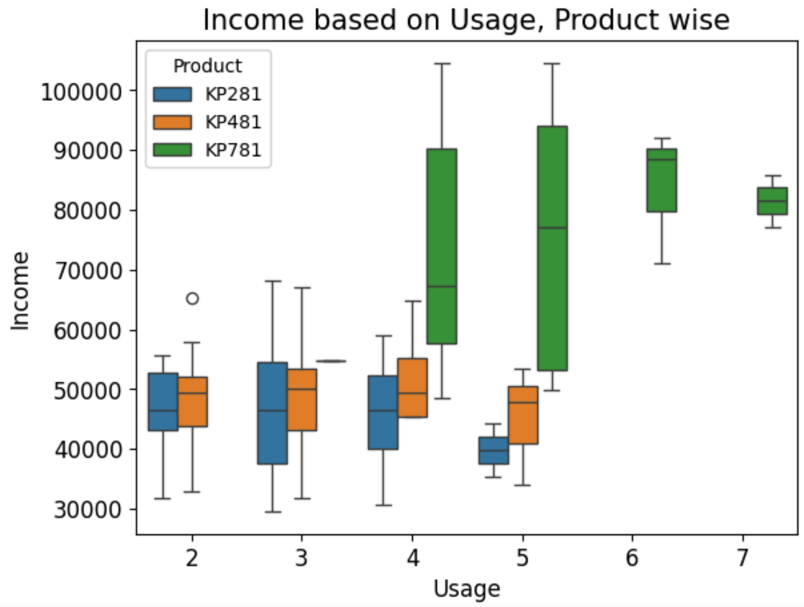

Income based on Gender, Product wise
Product
KP281
KP481
KP781
Income
100000
90000
80000
70000
60000
50000
40000
30000
Male
Female
Gender


Income based on Education, Product wise
Product
KP281
KP481
KP781
Income
100000
90000
80000
70000
60000
50000
40000
30000
12
13
14
15
16
18
20
21
Education


Income based on MaritalStatus, Product wise
Product
KP281
KP481
KP781
Income
100000
90000
80000
70000
60000
50000
40000
30000
Single
Partnered
MaritalStatus


Income based on Fitness, Product wise
Product
KP281
KP481
KP781
100000
90000
80000

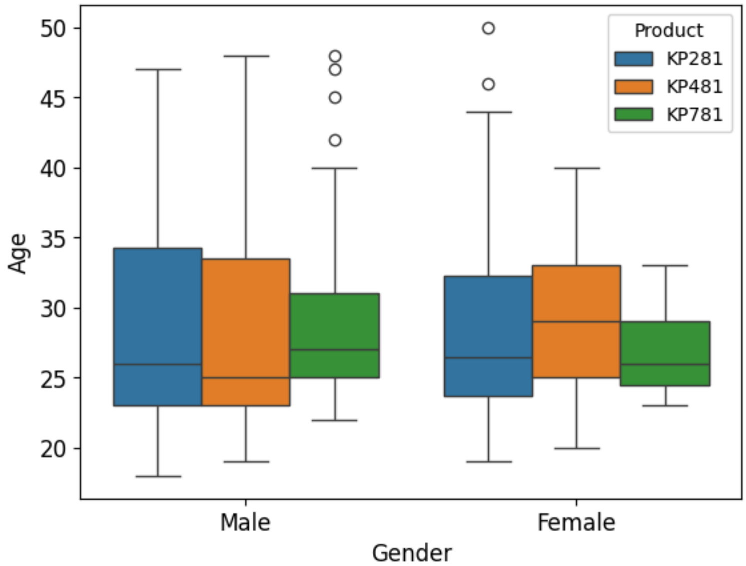




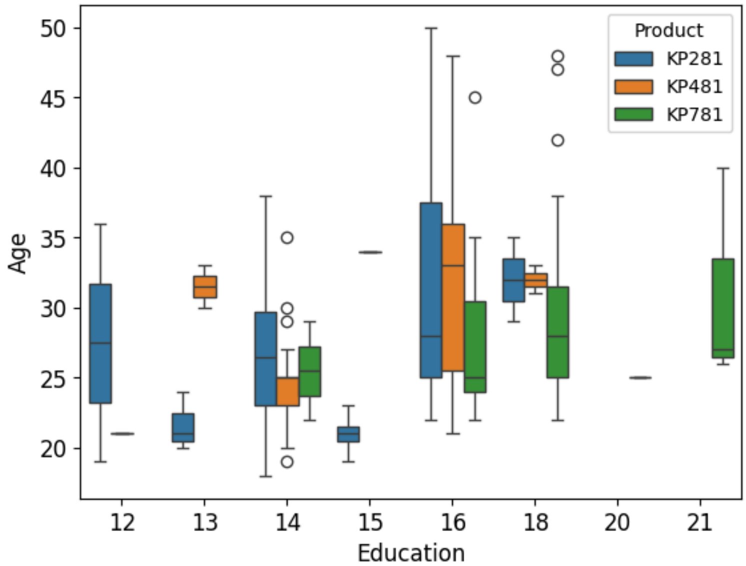
Multivariate analysis of Product on the basis of age:

```
for i in catcols:
  sns.boxplot(x=i,y='Age',hue='Product',data=data)
  plt.xlabel(i, fontsize=12)
  plt.ylabel('Age', fontsize=12)
  plt.xticks(fontsize=12)
  plt.yticks(fontsize=12)
  plt.title(f'Age based on {i}, Product wise', fontsize=15)
  plt.show()
```

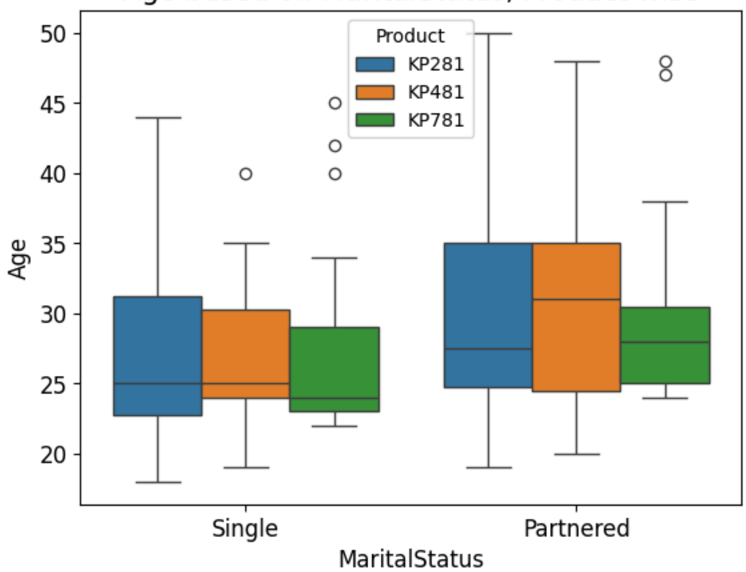
Age based on Gender, Product wise



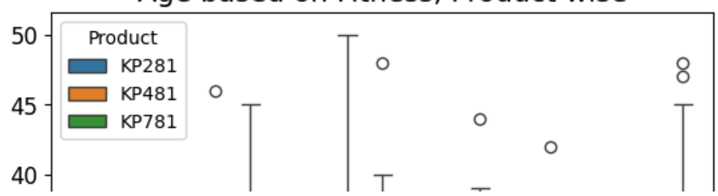
Age based on Education, Product wise

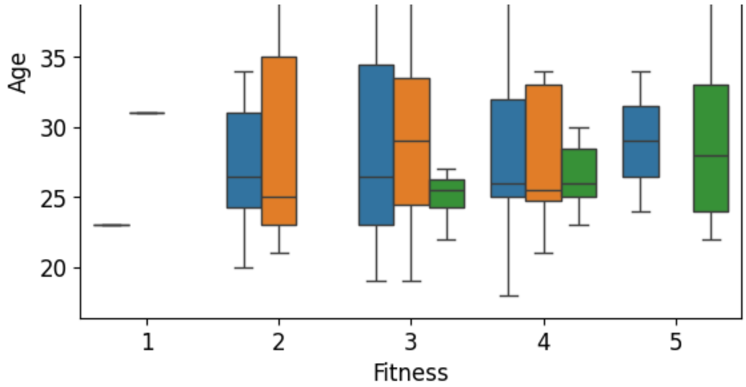


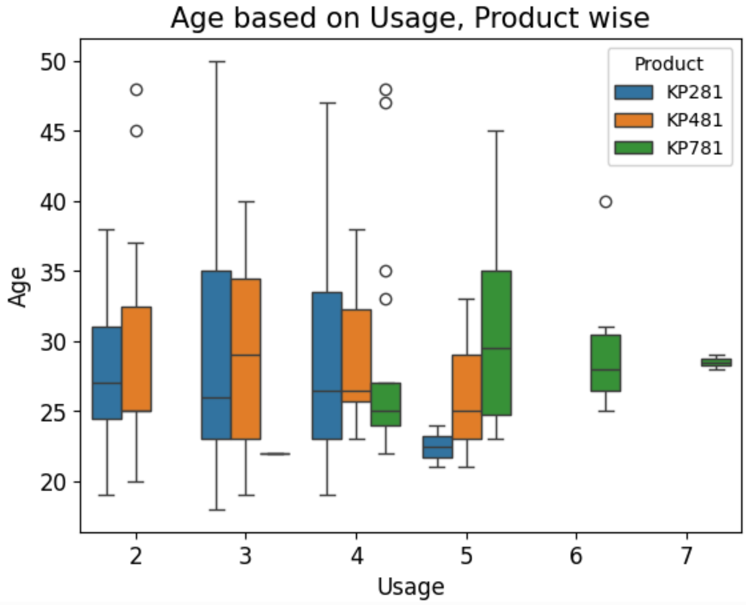
Age based on MaritalStatus, Product wise



Age based on Fitness, Product wise







Multivariate analysis of Product on the basis of Miles:

```
for i in catcols:
  sns.boxplot(x=i,y='Miles',hue='Product',data=data)
  plt.xlabel(i, fontsize=12)
  plt.ylabel('Miles', fontsize=12)
  plt.xticks(fontsize=12)
  plt.yticks(fontsize=12)
  plt.title(f'Miles based on {i}, Product wise', fontsize=15)
  plt.show()
```