General Analysis

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

```
data_path = "aerofit_treadmill.csv"
df = pd.read_csv(data_path)
df
```

	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

Next steps: Generate code with df View recommended plots

```
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
```

```
print(f"Number of rows: {df.shape[0]}\nNumber of columns: {df.shape[1]}")
```

```
Number of rows: 180
Number of columns: 9
```

```
df.describe(include= "all")
```

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

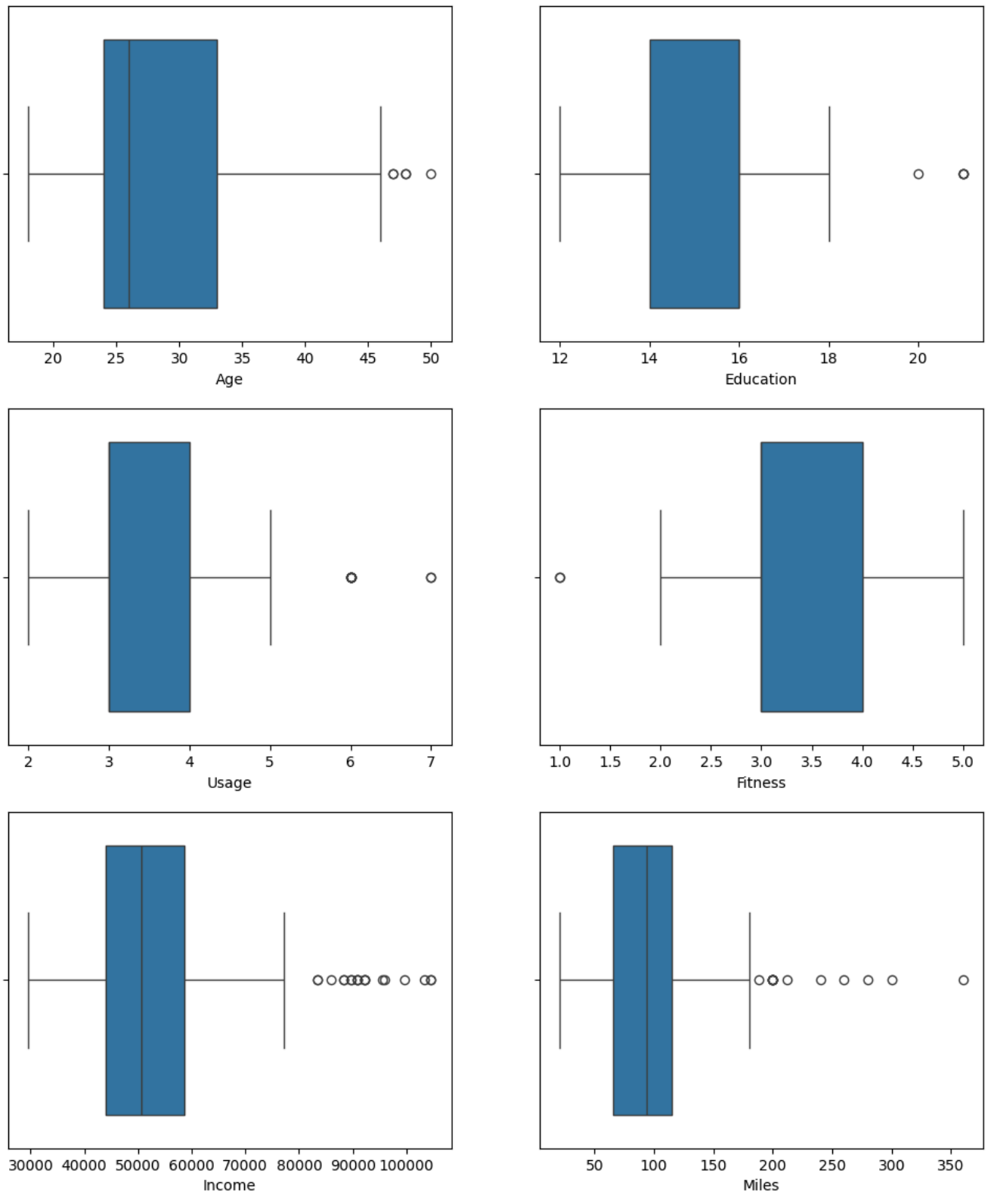
```
df.isna().sum()
```

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

There are no missing values in the data. There are 3 unique products in the dataset. KP281 is the most frequent product.

Detecting Outliers

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x="Age", orient='h', ax=axis[0,0])
sns.boxplot(data=df, x="Education", orient='h', ax=axis[0,1])
sns.boxplot(data=df, x="Usage", orient='h', ax=axis[1,0])
sns.boxplot(data=df, x="Fitness", orient='h', ax=axis[1,1])
sns.boxplot(data=df, x="Income", orient='h', ax=axis[2,0])
sns.boxplot(data=df, x="Miles", orient='h', ax=axis[2,1])
plt.show()
```



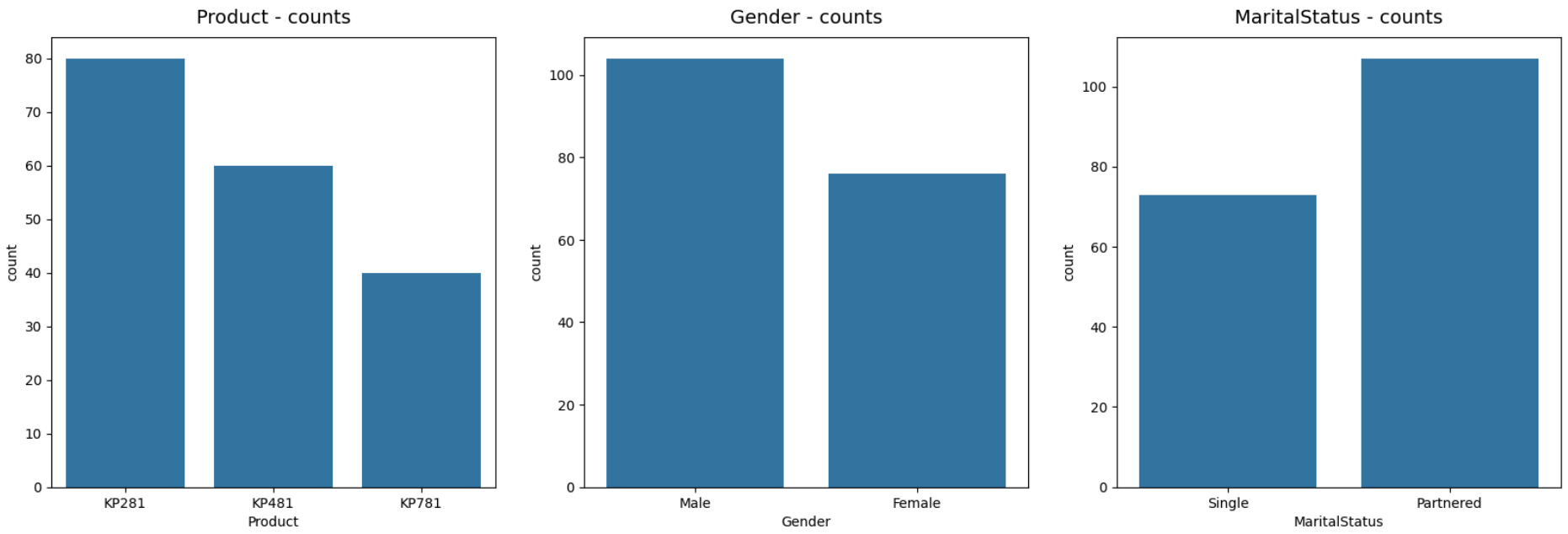
Observation

From the boxplot it is observed that:

- Age, Education and Usage have few outliers.
- Income and Miles have more outliers

Understanding the Distribution of Data

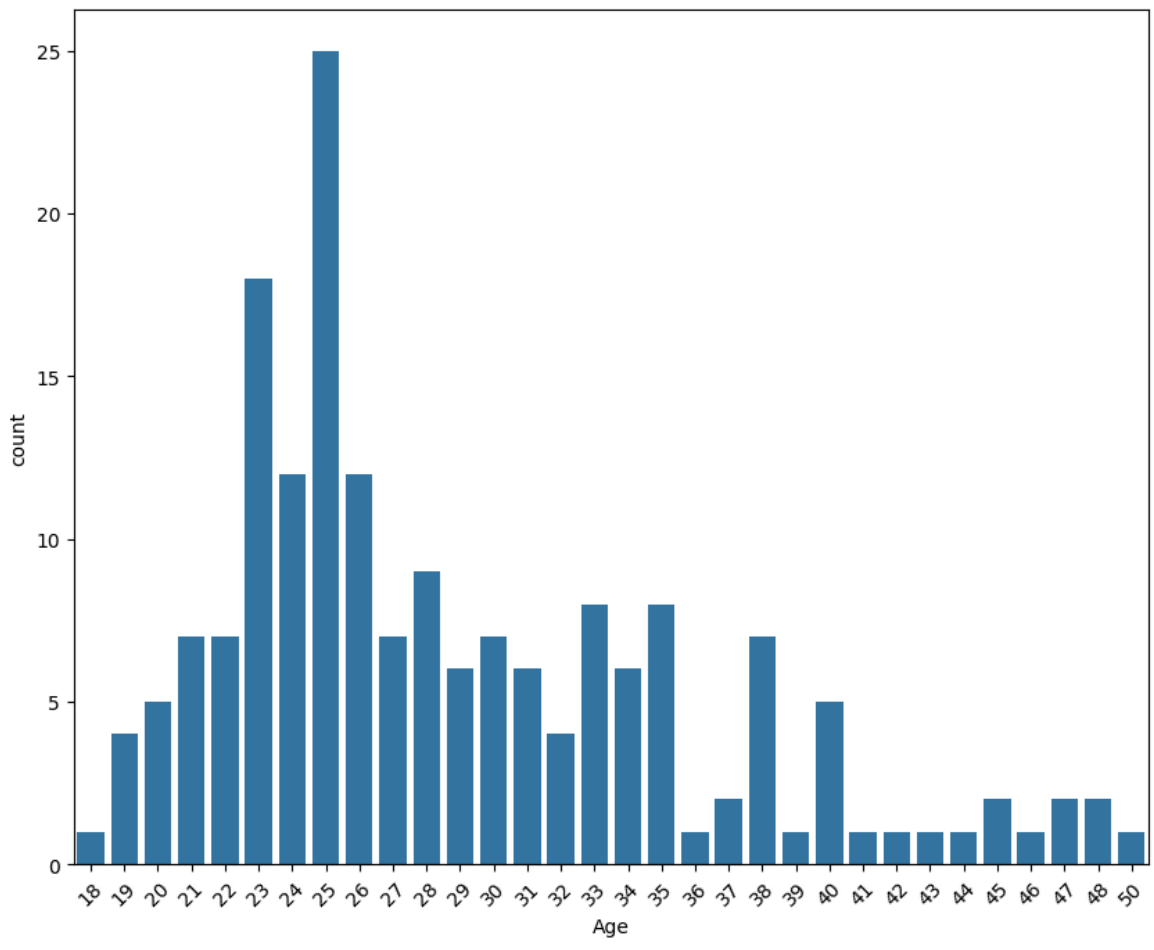
```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(20, 6))
sns.countplot(data=df, x='Product', ax=axs[0])
sns.countplot(data=df, x='Gender', ax=axs[1])
sns.countplot(data=df, x='MaritalStatus', ax=axs[2])

axs[0].set_title("Product - counts", pad=10, fontsize=14)
axs[1].set_title("Gender - counts", pad=10, fontsize=14)
axs[2].set_title("MaritalStatus - counts", pad=10, fontsize=14)
plt.show()
```



Observation:

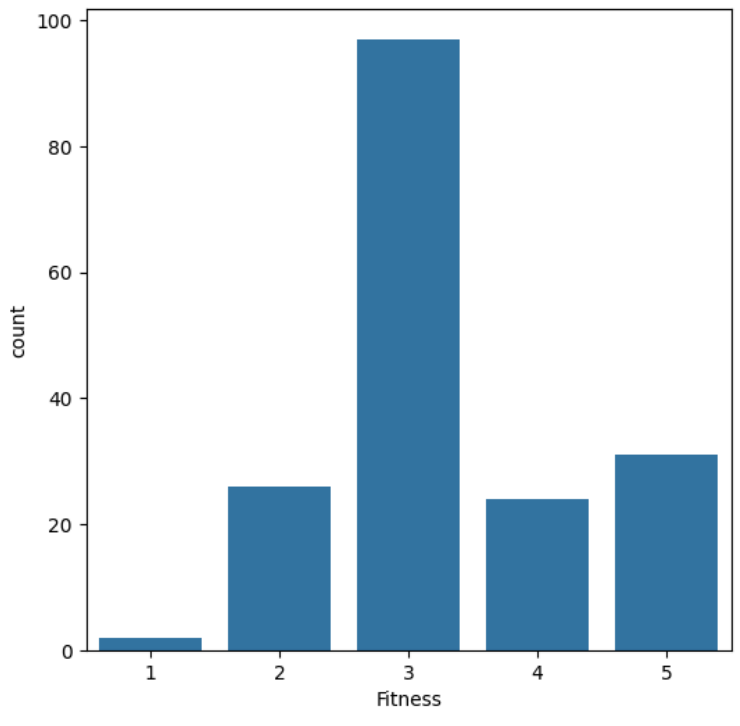
1. KP281 is the most frequent treadmill product.
2. There are more Males in the data than Females.
3. Partnered customers are more as compared to single customers.

```
plt.figure(figsize=(10, 8))
sns.countplot(data=df, x="Age")
plt.xticks(rotation = 45)
plt.show()
```



- Most customers belong to the range of 23-26 years of age.
- Highest number of customers belong to 25 years range.

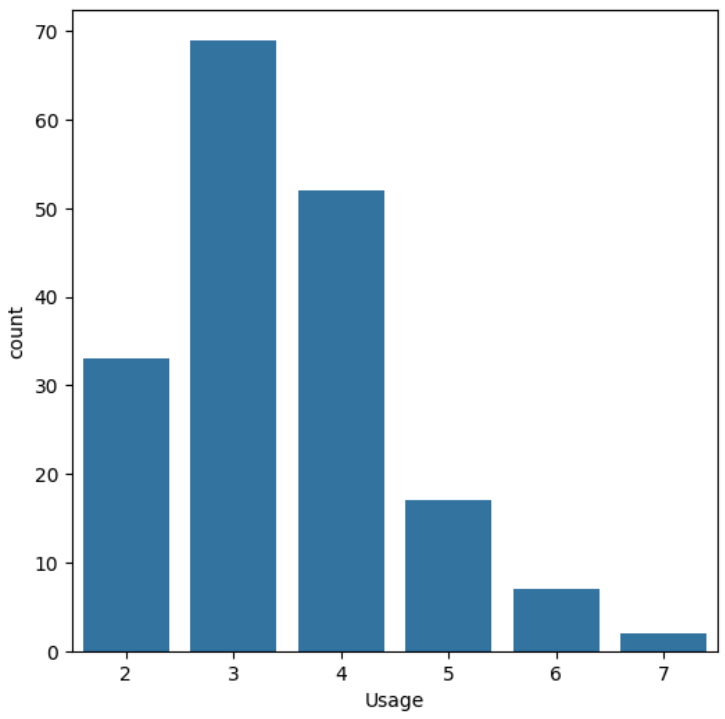
```
plt.figure(figsize=(6, 6))
sns.countplot(data=df, x="Fitness")
plt.show()
```



1. Most customer rate themselves as level 3 fitness.

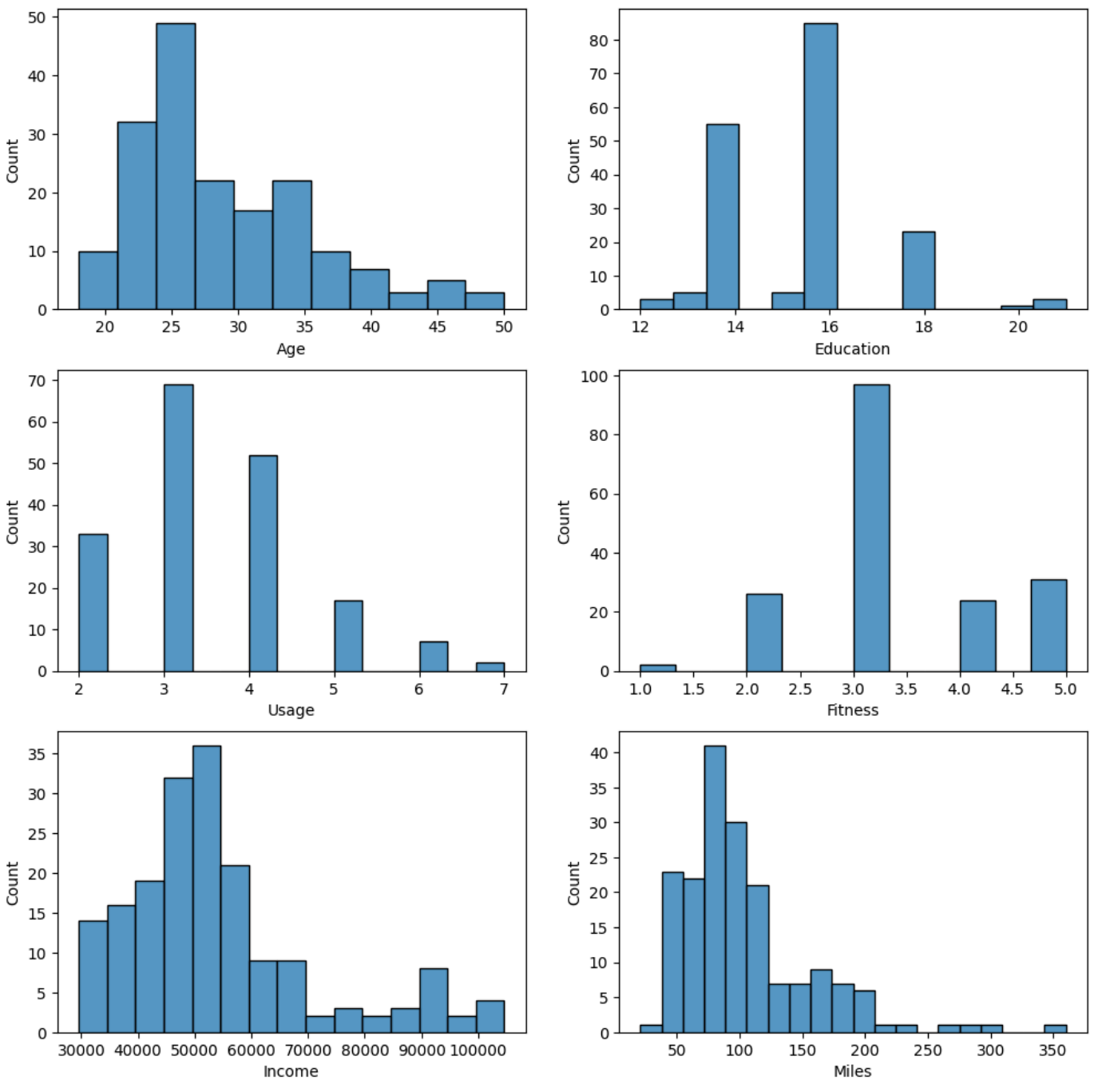
2. There are significant number of users for level 4 & 5 as well.

```
plt.figure(figsize=(6, 6))
sns.countplot(data=df, x="Usage")
plt.show()
```

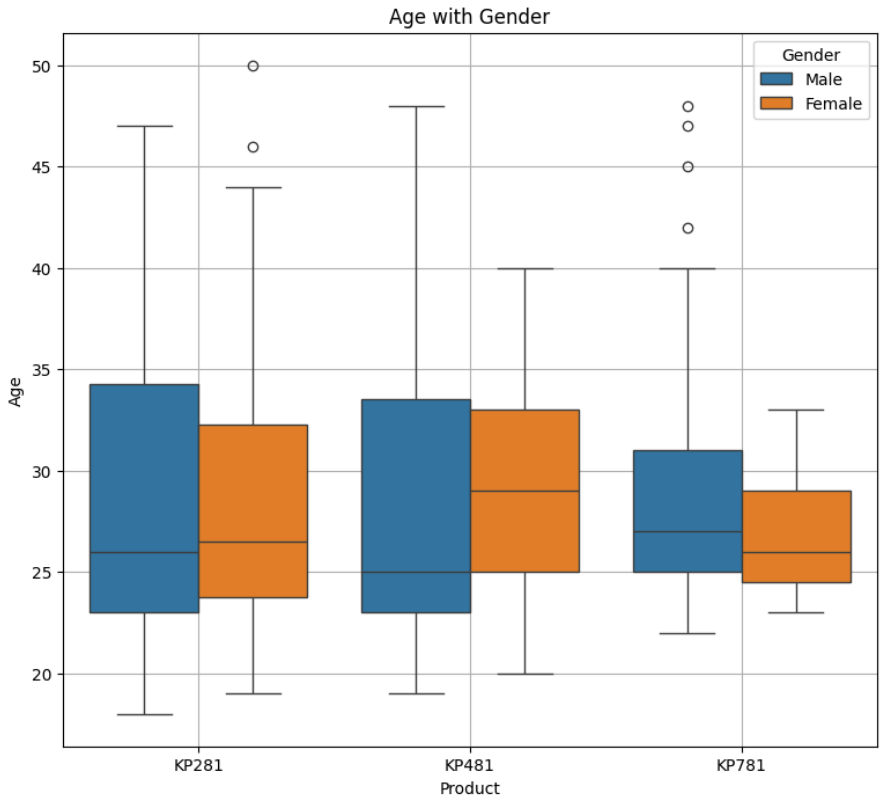


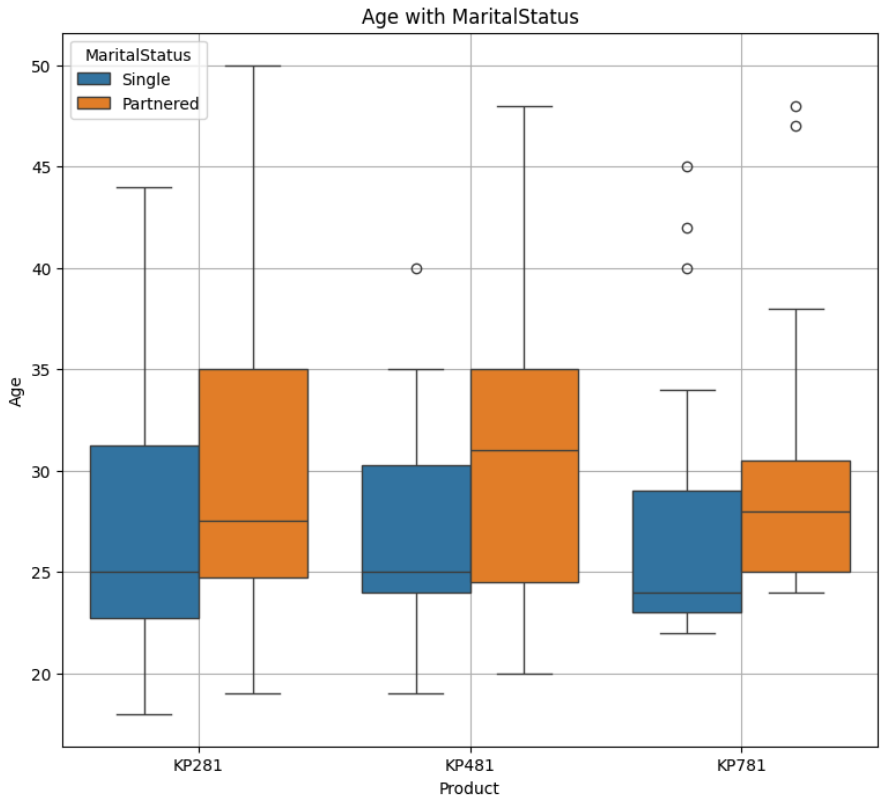
1. For most of the customers the weekly average usage of treadmill tends to be between 2-4 times.
2. There is high probability of using treadmill thrice a week.

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 12))

sns.histplot(data=df, x="Age", ax=axis[0,0])
sns.histplot(data=df, x="Education", ax=axis[0,1])
sns.histplot(data=df, x="Usage", ax=axis[1,0])
sns.histplot(data=df, x="Fitness", ax=axis[1,1])
sns.histplot(data=df, x="Income", ax=axis[2,0])
sns.histplot(data=df, x="Miles", ax=axis[2,1])
plt.show()
```




```
fig = plt.figure(figsize=(20,8))

plt.subplot(1, 2, 1)
sns.boxplot(data=df, y="Age", x = "Product", hue = "Gender")
plt.title("Age with Gender")
plt.grid()

plt.subplot(1, 2, 2)
sns.boxplot(data=df, y="Age", x = "Product", hue = "MaritalStatus")
plt.title("Age with MaritalStatus")
plt.grid()

plt.show()
```

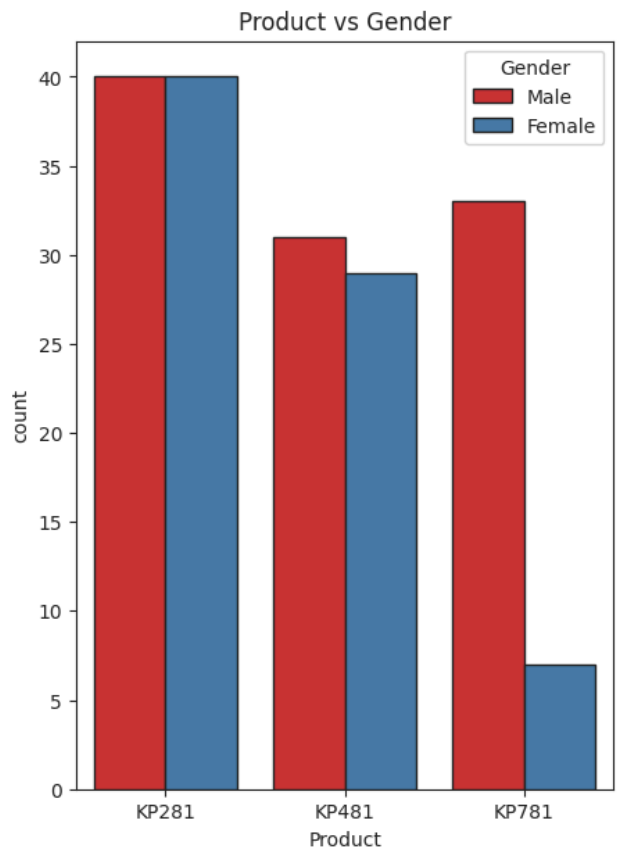




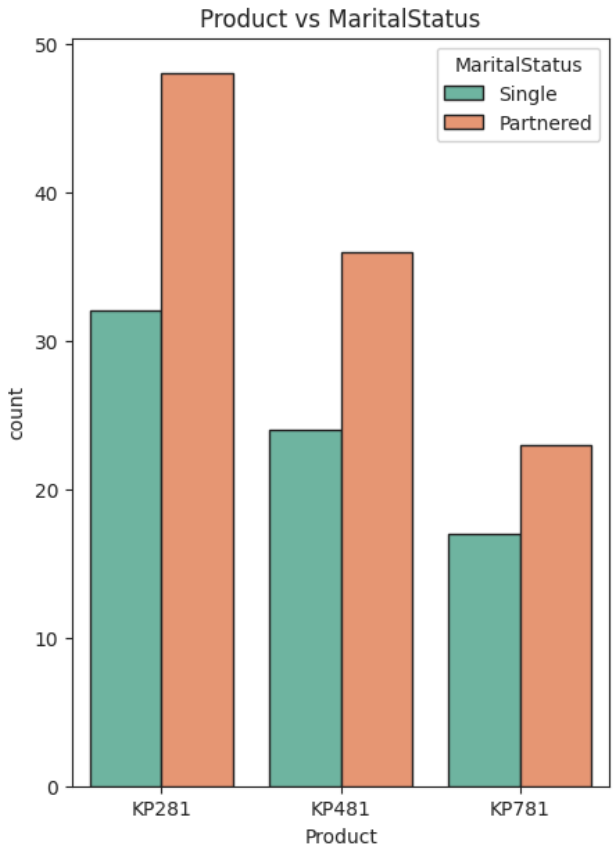
Observations :-

1. 75 % of customers below the age of 34 tend to buy the products.
2. The age of married customers buying products is more than the age of single customers.
3. Single male customers above the age of 40 are more likely to buy the KP781 model of treadmil.
4. Partnered customers below the age of 32 have shown more interest in buying the KP481 product.

```
sns.set_style(style='ticks')
fig = plt.figure(figsize=(5, 7))
sns.countplot(data=df, x='Product', hue='Gender',edgecolor="0.15", palette='Set1')
plt.title("Product vs Gender")
plt.show()
```



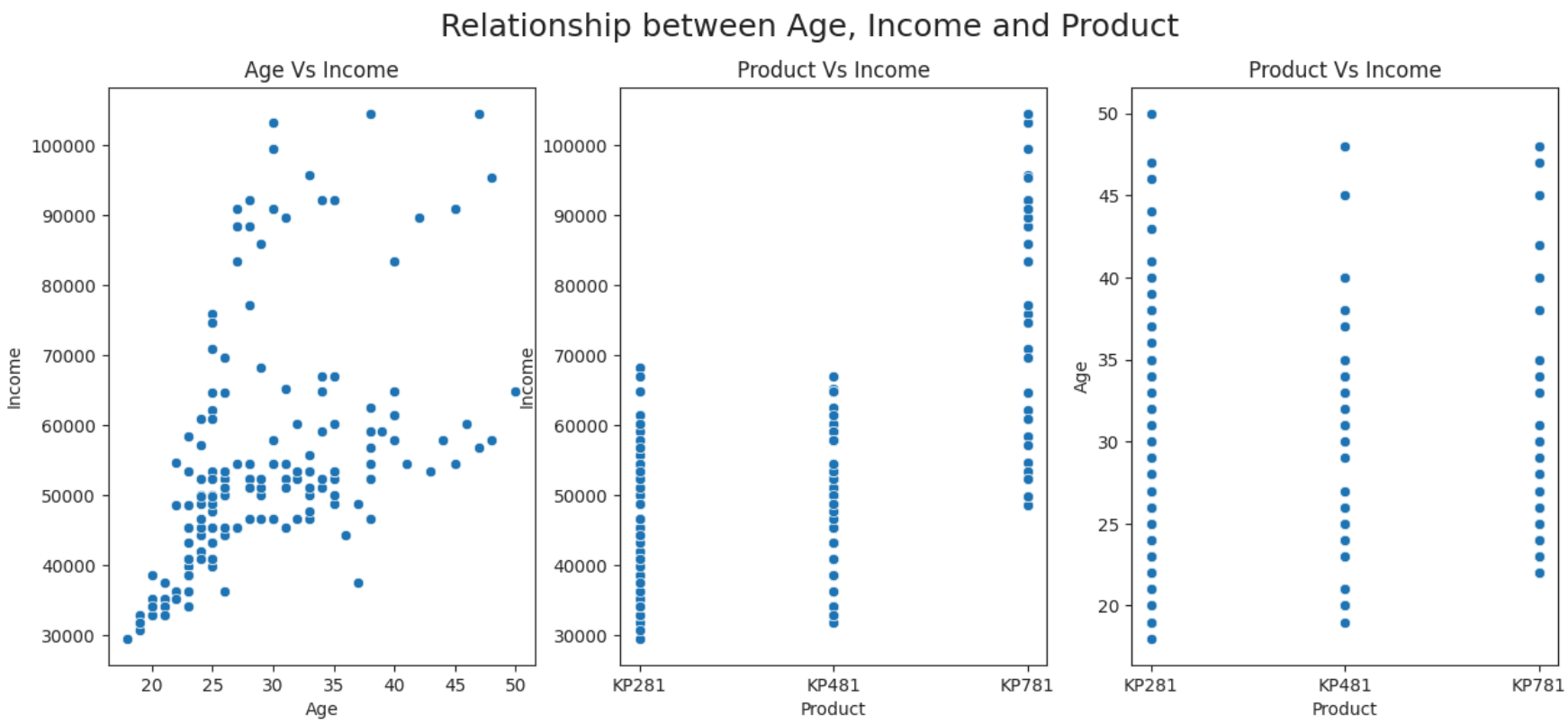

```
sns.set_style(style='ticks')
fig = plt.figure(figsize=(5, 7))
sns.countplot(data=df, x='Product', hue='MaritalStatus', edgecolor="0.15", palette='Set2')
plt.title("Product vs MaritalStatus")
plt.show()
```



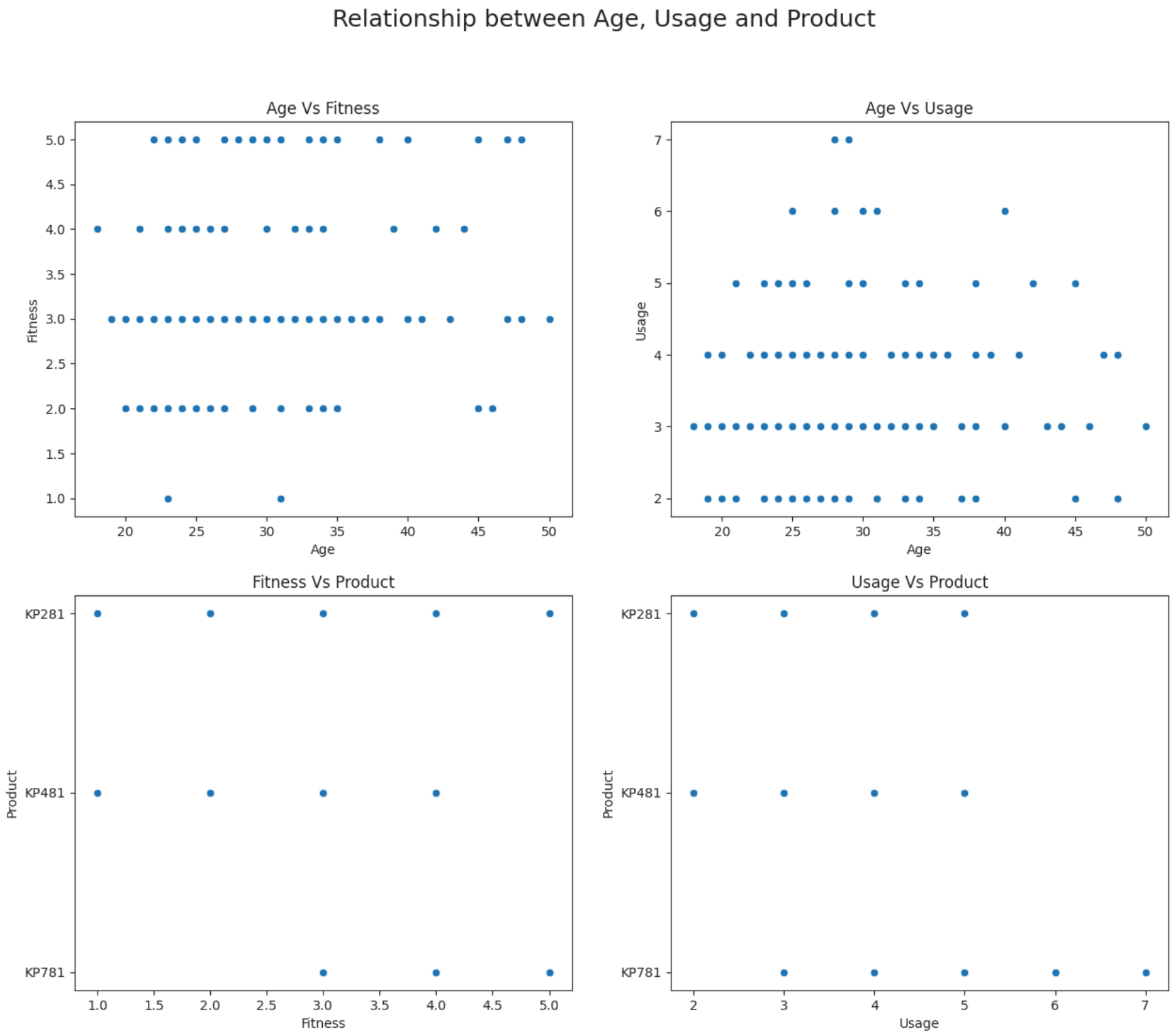
Observations

1. Product Vs Gender
   - An equal number of males and females have purchased the KP281 product.
   - Most of the Male customers have purchased the KP781 product.
2. Product Vs MaritalStatus
   - Customers with the marital status as partnered are more likely to purchase the product.

```
fig = plt.figure(figsize=(15, 6)).suptitle('Relationship between Age, Income and Product', fontsize=18)

plt.subplot(1, 3, 1)
sns.scatterplot(data=df, x='Age', y='Income')
plt.title("Age Vs Income")

plt.subplot(1, 3, 2)
sns.scatterplot(data=df, x='Product', y='Income')
plt.title("Product Vs Income")

plt.subplot(1, 3, 3)
sns.scatterplot(data=df, x='Product', y='Age')
plt.title("Product Vs Income")

plt.show()
```




```
fig = plt.figure(figsize=(15, 12)).suptitle('Relationship between Age, Usage and Product', fontsize=18)

plt.subplot(2, 2, 1)
sns.scatterplot(data=df, x='Age', y='Fitness')
plt.title("Age Vs Fitness")

plt.subplot(2,2,2)
sns.scatterplot(data=df, x='Age', y='Usage')
plt.title("Age Vs Usage")

plt.subplot(2,2,3)
sns.scatterplot(data=df, x='Fitness', y='Product')
plt.title("Fitness Vs Product")

plt.subplot(2,2,4)
sns.scatterplot(data=df, x='Usage', y='Product')
plt.title("Usage Vs Product")


plt.show()
```



Observations :

KP281:

1. This is a most purchased product among customers.
2. This is a desirable product for customers whose income ranges between 45K- 50K.

KP481 :

1. This product is popular among users of age 28-31.
2. Customers with income of 50K - 60K are more likely to purchase it.

KP781:

1. Higher the age, higher is the income of the customers.
2. Higher the income range of customer more are the chances of buying this model which is most advanced model.

```
df['Product'].value_counts(normalize=True)
```

```
KP281    0.444444
KP481    0.333333
KP781    0.222222
Name: Product, dtype: float64
```

```
df1 = df[['Product', 'Gender', 'MaritalStatus']].melt()
df1.groupby(['variable', 'value'])[['value']].count() / len(df)
```

		value
variable	**value**	
Gender	**Female**	0.422222
	Male	0.577778
MaritalStatus	**Partnered**	0.594444
	Single	0.405556
Product	**KP281**	0.444444
	KP481	0.333333
	KP781	0.222222

Representing the Probability

Observations

1. Product

- 44.44% of the customers have purchased KP281 product.
- 33.33% of the customers have purchased KP481 product.
- 22.22% of the customers have purchased KP781 product.

2. Gender

- 57.78% of the customers are Male.
- 43.22% of the customers are Female.

3. MaritalStatus

- 59.44% of the customers are Partnered.
- 40.55% of customers are Single.

```
df1 = pd.crosstab(index=df['Gender'], columns=[df['Product']])
df1
```

Product	KP281	KP481	KP781
Gender			
Female	40	29	7
Male	40	31	33

```
def p_prod_given_gender(gender, print_marginal=False):
  df1 = pd.crosstab(index=df['Gender'], columns=[df['Product']])
  p_781 = df1['KP781'][gender] / df1.loc[gender].sum()
  p_481 = df1['KP481'][gender] / df1.loc[gender].sum()
  p_281 = df1['KP281'][gender] / df1.loc[gender].sum()
  if print_marginal:
    print(f"P(Male): {df1.loc['Male'].sum()/len(df):.2f}")
    print(f"P(Female): {df1.loc['Female'].sum()/len(df):.2f}\n")
  print(f"P(KP781/{gender}): {p_781:.2f}")
  print(f"P(KP481/{gender}): {p_481:.2f}")
  print(f"P(KP281/{gender}): {p_281:.2f}\n")
p_prod_given_gender('Male', True)
p_prod_given_gender('Female')
```

```
P(Male): 0.58
P(Female): 0.42

P(KP781/Male): 0.32
P(KP481/Male): 0.30
P(KP281/Male): 0.38

P(KP781/Female): 0.09
P(KP481/Female): 0.38
P(KP281/Female): 0.53
```

Probability of male buying a treadmill is :

- P(Male): 0.58

Probability of female buying a treadmill is :

- P(Female): 0.42

Probability of customer buying a treadmill given the customer is male :

- KP781: 0.32
- KP481: 0.30
- KP281: 0.38

Probability of customer buying a treadmill given the customer is female :

- KP781: 0.09
- KP481: 0.38
- KP281: 0.53

```
df2 = pd.crosstab(index=df['MaritalStatus'], columns=[df['Product']])
df2
```

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

```
def p_prod_given_ms(maristatus, print_marginal=False):
  df2 = pd.crosstab(index=df['MaritalStatus'], columns=[df['Product']])
  p_781 = df2['KP781'][maristatus] / df2.loc[maristatus].sum()
  p_481 = df2['KP481'][maristatus] / df2.loc[maristatus].sum()
  p_281 = df2['KP281'][maristatus] / df2.loc[maristatus].sum()
  if print_marginal:
    print(f"P(Partnered): {df2.loc['Partnered'].sum()/len(df):.2f}")
    print(f"P(Single): {df2.loc['Single'].sum()/len(df):.2f}\n")
  print(f"P(KP781/{maristatus}): {p_781:.2f}")
  print(f"P(KP481/{maristatus}): {p_481:.2f}")
  print(f"P(KP281/{maristatus}): {p_281:.2f}\n")
p_prod_given_ms('Partnered', True)
p_prod_given_ms('Single')
```

```
P(Partnered): 0.59
P(Single): 0.41

P(KP781/Partnered): 0.21
P(KP481/Partnered): 0.34
P(KP281/Partnered): 0.45

P(KP781/Single): 0.23
P(KP481/Single): 0.33
P(KP281/Single): 0.44
```

Probability of partnered customer buying a treadmill is :

- P(Partnered): 0.59

Probability of single customer buying a treadmill is :

- P(Single): 0.41

Probability of customer buying a treadmill given marital status is partnered :

- KP781: 0.21
- KP481: 0.34

- KP281: 0.45

Probability of customer buying a treadmill given marital status is single : -

- KP781: 0.23
- KP481: 0.33
- KP281: 0.44

```
p_781 = df2['KP781']["Partnered"] / df2.loc["Partnered"].sum()
p_781
```

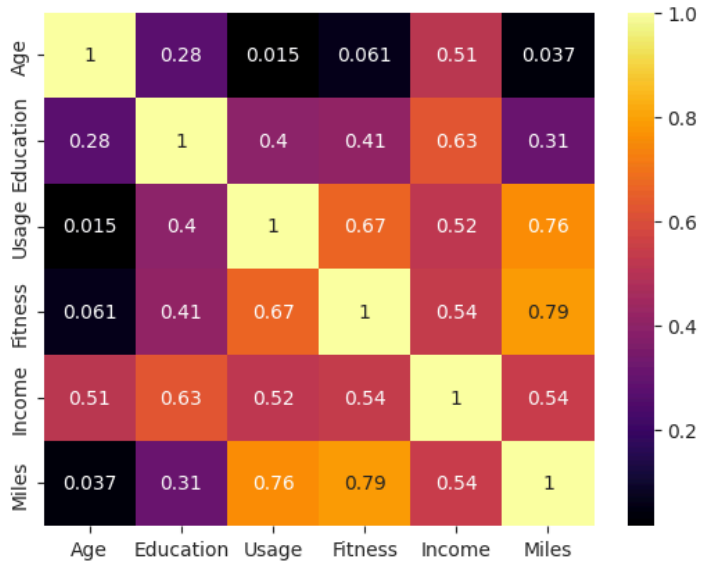
```
0.21495327102803738
```

Checking the Correlation among different factors

```
df.corr(numeric_only = True)
```

	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

```
sns.heatmap(df.corr(numeric_only = True), cmap='inferno', annot=True)
plt.show()
```



Observations :

We see a strong correlation between the following attributes:

- Fitness and miles.
- Usage and miles.
- Usage and fitness.