Business Case- Aerofit - Descriptive Statistics & Probability

In [1]:

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

Import the dataset and do usual data analysis steps like checking the structure & characteristics of the dataset

In [2]:

```
#Importing the Dataset of Aerofit
df = pd.read_csv('C://Users//dell//OneDrive//Desktop//Personal Doc//Aerofit_Tredmill.csv
df
```

Out[2]:

	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 [3]:

```
df.describe()
```

Out[3]:

	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 [4]:

```
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
```

As we can see in the above code it shows-

1. Product, Gender, MaritalStatus is in object Dtype.
2. Age, Education, Usage, Fitness, Income, Miles is in int64 Dtype.

In [5]:

```
#Checking the rows and columns of the dataset
df.shape
```

Out[5]:

```
(180, 9)
```

In [6]:

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

Out[6]:

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

In [7]:

```
mean = df.mean()
mean
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_33676\2523297653.py:1: FutureWa
rning: The default value of numeric_only in DataFrame.mean is deprecated.
In a future version, it will default to False. In addition, specifying 'nu
meric_only=None' is deprecated. Select only valid columns or specify the v
alue of numeric_only to silence this warning.
  mean = df.mean()
```

Out[7]:

```
Age             28.788889
Education       15.572222
Usage            3.455556
Fitness          3.311111
Income       53719.577778
Miles          103.194444
dtype: float64
```

In [8]:

```
median = df.median()
median
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_33676\1236989899.py:1: FutureWa
rning: The default value of numeric_only in DataFrame.median is deprecate
d. In a future version, it will default to False. In addition, specifying
'numeric_only=None' is deprecated. Select only valid columns or specify th
e value of numeric_only to silence this warning.
  median = df.median()
```

Out[8]:

```
Age             26.0
Education       16.0
Usage            3.0
Fitness          3.0
Income       50596.5
Miles           94.0
dtype: float64
```

What we have find in this dataset so far is-

1. The total number of rows is 180 and columns is 9 in this dataset.
2. The mean age is 28.7 as well as median is 26, also minimum age is 18 and maximum age is 50 in this dataset.
3. In Education column, the mean of education is 15.57 ans weel as median is 16, also minimum Education is 15 and maximum Education is 21.
4. In Usage column, the mean of Usage per week is 3.4 and median is 3, also minimum Usage is 2 and maximum Usage is 7.
5. In Fitness column, the mean of Fitness is 3.3 and median is 3, also minimum Fitness is 1 and maximum Fitness is 5.
6. In Income column (in $), the mean Income is 53719.57 and median is 50596.5, also minimum Income is 29562 and Maximum Income is 104581.
7. In Miles column, the mean is 103.19 and median is 94, also minimum Miles is 21 and Maximum miles is 360.

Non Graphical Analysis: Value Counts and Unique Characters

In [9]:

```
# number of unique product id'd
df['Product'].unique()
```

Out[9]:

```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

In [10]:

```
# number of unique ages
df['Age'].unique()
```

Out[10]:

```
array([18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 43, 44, 46, 47, 50, 45, 48, 42],
      dtype=int64)
```

In [11]:

```
# List of number of male and female customer
df['Gender'].value_counts()
```

Out[11]:

```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [12]:

```
# list of unique Educations
df['Education'].unique().tolist()
```

Out[12]:

```
[14, 15, 12, 13, 16, 18, 20, 21]
```

In [13]:

```
# Number of customer againts the rating scale 1 to 5
df['Fitness'].value_counts().sort_index()
```

Out[13]:

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

In [14]:

```
## does income have any effect on the choice of the product
df.groupby('Product')["Income"].describe()
```

Out[14]:

	count	mean	std	min	25%	50%	75%	max
Product								
KP281	80.0	46418.025	9075.783190	29562.0	38658.00	46617.0	53439.0	68220.0
KP481	60.0	48973.650	8653.989388	31836.0	44911.50	49459.5	53439.0	67083.0
KP781	40.0	75441.575	18505.836720	48556.0	58204.75	76568.5	90886.0	104581.0

In [15]:

```
# Number of customers with 3 different product types
df['Product'].value_counts().sort_index()
```

Out[15]:

```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [16]:

```
# Number of customers counts on Usage
df['Usage'].value_counts().sort_index()
```

Out[16]:

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

In [17]:

```
# Number of Single and Partnered customers
df['MaritalStatus'].value_counts()
```

Out[17]:

```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

Categorize the fitness level into a different categories by adding a new column

In [18]:

```
# Converting Int data type of fitness rating to object data type in new column
df_category = df
df_category['Fitness_Category'] = df.Fitness
df_category.head()
```

Out[18]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_(
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 [26]:

```
df_category['Fitness_Category'].replace({1:"Poor",
                                          2:"Bad",
                                          3:"Average",
                                          4:"Good",
                                          5:"Excellent"},inplace=True)
df_category.head()
```

Out[26]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_(
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	

Categorization of Fitness Rating to following descriptive categories

1. Poor (1)
2. Bad (2)
3. Average (3)
4. Good (4)
5. Excellent (5)

Probabilities

Marginal Probabilities

In [27]:

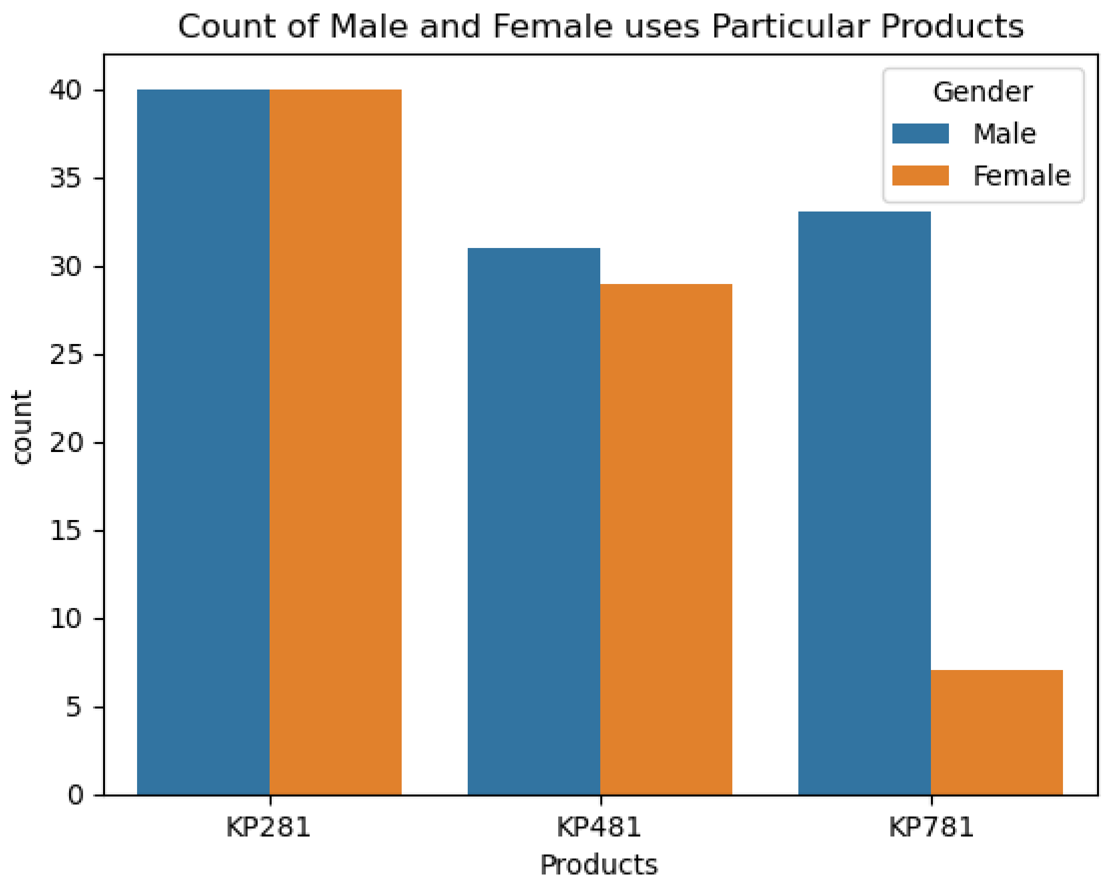
```
pd.crosstab([df.Product],df.Gender,margins=True)
```

Out[27]:

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

In [29]:

```
sns.countplot(x = "Product", data= df, hue = "Gender")
plt.xlabel("Products")
plt.title("Count of Male and Female uses Particular Products")
plt.show()
```



In [28]:

```
np.round(((pd.crosstab(df.Product,df.Gender,margins=True))/180)*100,2)
```

Out[28]:

Gender	Female	Male	All
Product			
KP281	22.22	22.22	44.44
KP481	16.11	17.22	33.33
KP781	3.89	18.33	22.22
All	42.22	57.78	100.00

Marginal Probability

=> Probability of Male Customer Purchasing any product is : 57.77 %

=> Probability of Female Customer Purchasing any product is : 42.22 %

Marginal Probability of any customer buying

=> product KP281 is : 44.44 % (cheapest / entry level product)

=> product KP481 is : 33.33 % (intermediate user level product)

Conditional Probabilities

In [31]:

```
np.round((pd.crosstab([df.Product],df.Gender,margins=True,normalize="columns"))*100,2)
```

Out[31]:

Gender	Female	Male	All
Product			
KP281	52.63	38.46	44.44
KP481	38.16	29.81	33.33
KP781	9.21	31.73	22.22

Probability of Selling Product

KP281 | Female = 52 %

KP481 | Female = 38 %

KP781 | Female = 10 %

KP281 | male = 38 %

KP481 | male = 30 %

KP781 | male = 32 %

Probability of Female customer buying KP281(52.63%) is more than male(38.46%).

KP281 is more recommended for female customers.

Probability of Male customer buying Product KP781(31.73%) is way more than female(9.21%).

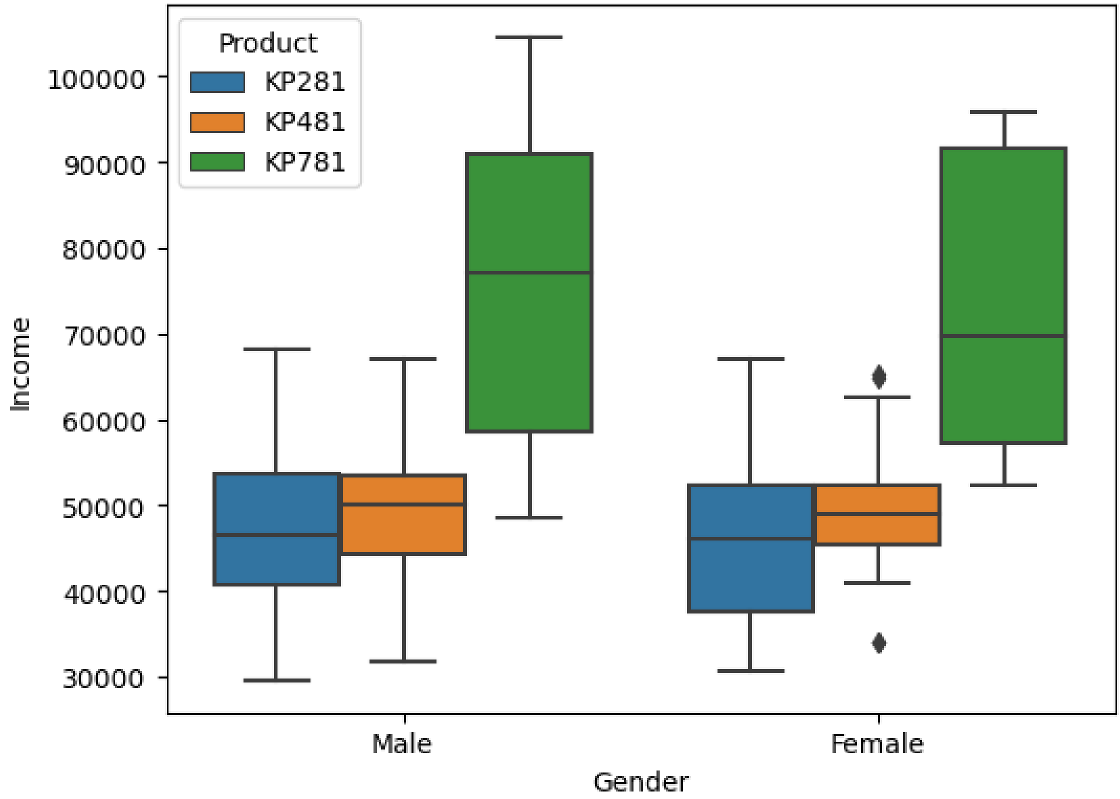
Probability of Female customer buying Product KP481(38.15%) is significantly higher than male (29.80%.)

KP481 product is specifically recommended for Female customers who are intermediate user.

Visualizations

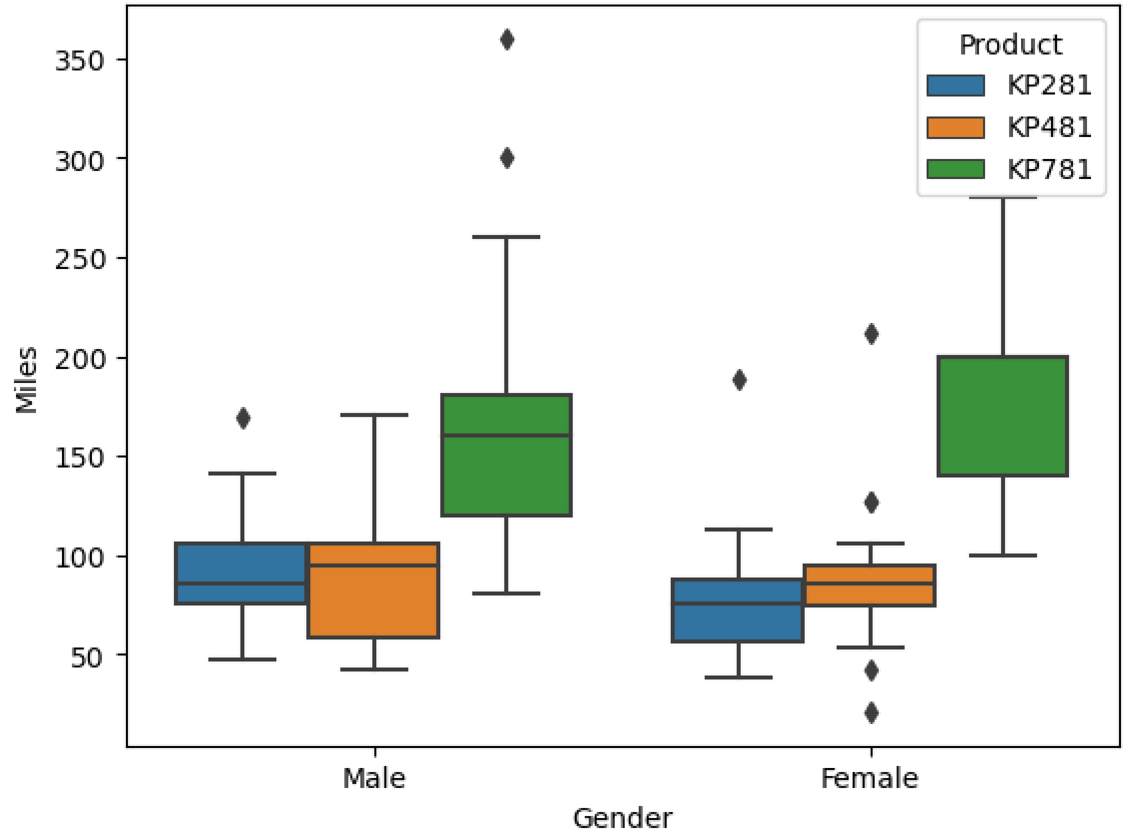
In [33]:

```
sns.boxplot(x="Gender", y="Income", hue="Product", data=df)
plt.show()
```



In [34]:

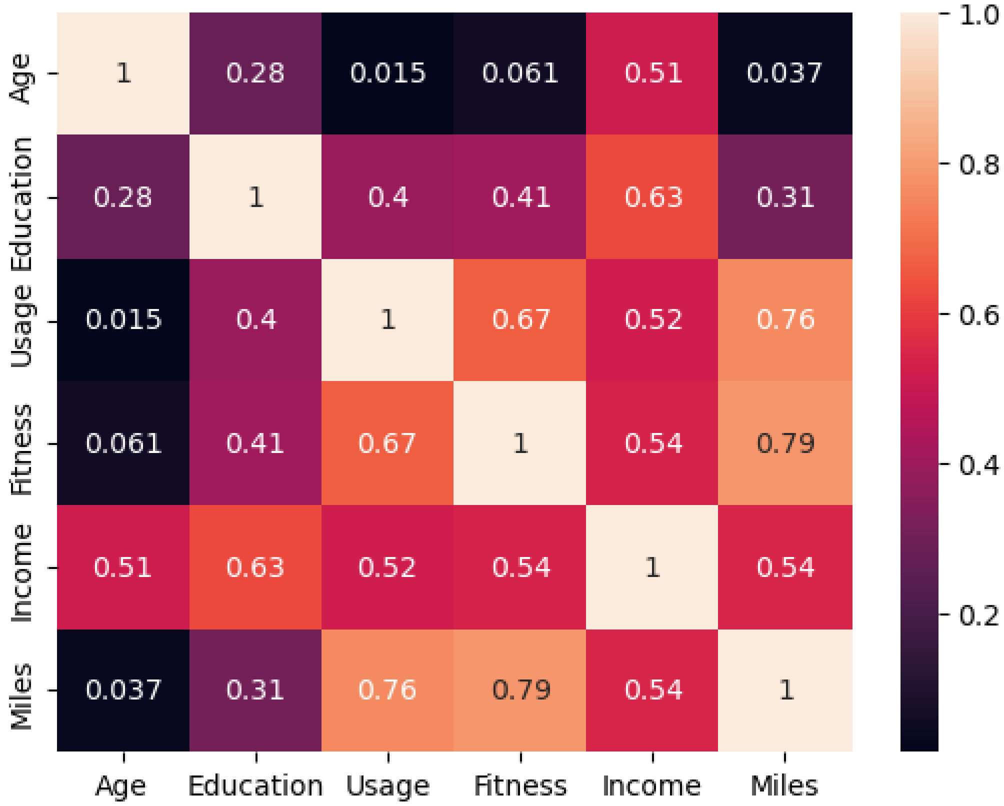
```
# impact of miles on the choice of the product
sns.boxplot(x="Gender", y="Miles", hue="Product", data=df)
plt.show()
```



In [36]:

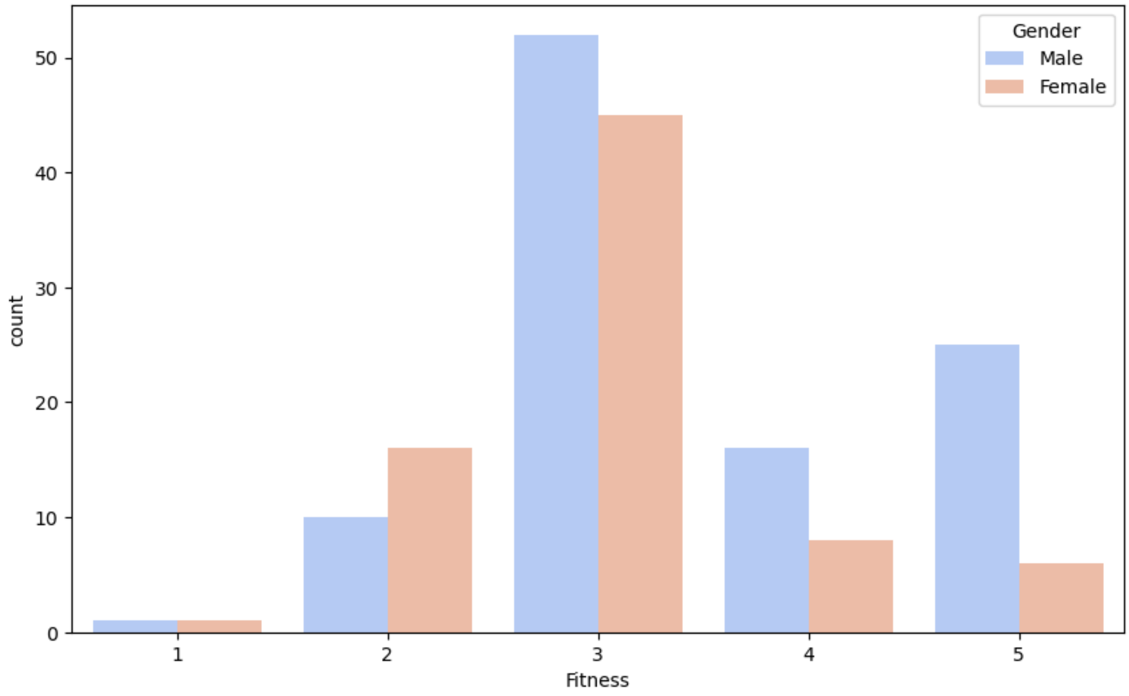
```
# Check correlation among different factors using heat maps
sns.heatmap(df.corr(), annot=True)
plt.show()
```

```
C:\Users\dell\AppData\Local\Temp\ipykernel_33676\3156606364.py:2: FutureWa
rning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  sns.heatmap(df.corr(), annot=True)
```



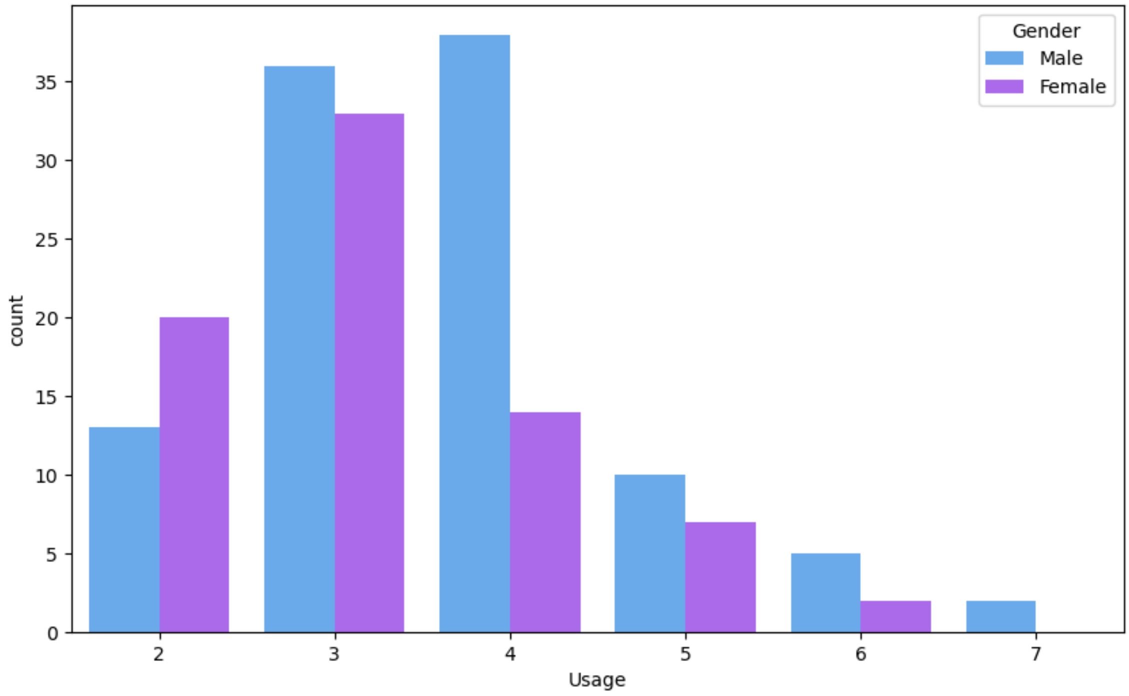
In [37]:

```
# Fitness rating among the customers categorised by Gender
plt.figure(figsize=(10,6))
sns.countplot(data=df,x='Fitness',hue='Gender',palette='coolwarm')
plt.show()
```



In [38]:

```
# Purchased product usage among Gender
plt.figure(figsize=(10,6))
sns.countplot(data=df,x='Usage',hue='Gender',palette='cool')
plt.show()
```



Recommendations

- In the above analysis we see that females are less as compared to males in using the above treadmill range. The company should advertise their product and run marketing campaign for females so that females get encouraged and believe that it is equally important for them and buy these products.
- In the above analysis we can see that KP281 and KP481 have sold more than KP781 & it has been less used as compared to others.The company should do promotions for KP781 tredmill,they should do promotion by-

1. Doing advertising in Social media websites and other websites also.
2. Collaborating with influencers in social media platform (Youtube,Instagram,pinterest) so that it should be more visible to people and they buy the product.
3. Promoting the product through mass media like television,newspapers etc.
4. Company should provide discount on KP781 tredmill (specially during festival season) so that people should attract towards the product and buy it

- COmpany should create new marketing team with young minds along with experienced employees so that their product could reach masses.
- According to the need , company should change their strategies from time to time.
- The company should sell their product(KP781) in competitive prices to attract more customers.