business-case-aerofit

December 30, 2023

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

[4]:
```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/
↪original/aerofit_treadmill.csv?1639992749
```

```
Downloading…
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/ori
ginal/aerofit_treadmill.csv?1639992749
To: /content/aerofit_treadmill.csv?1639992749
100% 7.28k/7.28k [00:00<00:00, 25.8MB/s]
```

1) ***Defining Problem Statement and Analysing basic metrics***

1. Observations on shape of data, data types of all the attributes, conversion of categorical attributes to 'category' (If required), statistical summary

[5]:
```
df = pd.read_csv('aerofit_treadmill.csv?1639992749')
df.head()
```

[5]:

	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

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

[ ]:
```
df.shape
```

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

[ ]:
```
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
```

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

[ ]:
```
              Age   Education       Usage     Fitness         Income  \
count  180.000000  180.000000  180.000000  180.000000     180.000000
mean    28.788889   15.572222    3.455556    3.311111   53719.577778
std      6.943498    1.617055    1.084797    0.958869   16506.684226
min     18.000000   12.000000    2.000000    1.000000   29562.000000
25%     24.000000   14.000000    3.000000    3.000000   44058.750000
50%     26.000000   16.000000    3.000000    3.000000   50596.500000
75%     33.000000   16.000000    4.000000    4.000000   58668.000000
max     50.000000   21.000000    7.000000    5.000000  104581.000000

            Miles
count  180.000000
mean   103.194444
std     51.863605
min     21.000000
25%     66.000000
50%     94.000000
75%    114.750000
max    360.000000
```

Descriptive Analysis

- Total count of all columns is **180**
- **Age**: **Mean** age of the customer is **28** years, half of the customer's **mean** age is **26**.
- **Education**: Mean **Education** is **15** with maximum as **21** and minimum as **12**.
- **Usage**: Mean **Usage** per week is **3.4**, with maximum as **7** and minimum as **2**.
- **Fitness**: Average **rating** is **3.3** on a scale of **1 to 5**.

- **Miles**: Average number of **miles** the customer walks is **103** with maximum distance travelled by most people is almost **115** and minimum is **21**
- **Income** (in $): Most customer earns around **58K** annually, with maximum of **104K** and minimum almost **30K**

2) **Non-Graphical Analysis: Value counts and unique attributes**

[ ]:
```
df['Product'].nunique()
```

[ ]:
```
3
```

[ ]:
```
df['Product'].unique()
```

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

[ ]:
```
df['Product'].unique().tolist()
```

[ ]:
```
['KP281', 'KP481', 'KP781']
```

[ ]:
```
df['Age'].nunique()
```

[ ]:
```
32
```

[ ]:
```
df['Age'].unique()
```

[ ]:
```
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])
```

[ ]:
```
df['Age'].unique().tolist()
```

[ ]:
```
[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]
```

[ ]:
```
df['Age'].value_counts()
```

[ ]:
```
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
```

[ ]:
```
df['Gender'].nunique()
```

[ ]:
```
2
```

[ ]:
```
df['Gender'].unique()
```

[ ]:
```
array(['Male', 'Female'], dtype=object)
```

[ ]:
```
df['Gender'].unique().tolist()
```

[ ]:
```
['Male', 'Female']
```

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

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

[6]:
```
df['Education'].nunique()
```

[6]:
```
8
```

[7]:
```
df['Education'].unique()
```

[7]:
```
array([14, 15, 12, 13, 16, 18, 20, 21])
```

[8]:
```
df['Education'].value_counts()
```

[8]:
```
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

[ ]:
```
df['Education'].unique().tolist()
```

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

[9]:
```
df['Education'].value_counts().sort_index()
```

[9]:
```
12     3
13     5
14    55
15     5
16    85
18    23
20     1
21     3
Name: Education, dtype: int64
```

[10]:
```
df['Fitness'].nunique()
```

[10]:
```
5
```

[11]:
```
df['Fitness'].unique()
```

[11]:
```
array([4, 3, 2, 1, 5])
```

[12]:
```
df['Fitness'].value_counts()
```

[12]:
```
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

[ ]:
```
df['Fitness'].value_counts().sort_index()
```

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

[13]:
```
df['Fitness'].unique().tolist()
```

[13]:
```
[4, 3, 2, 1, 5]
```

[14]:
```
df['Product'].nunique()
```

[14]:
```
3
```

[15]:
```
df['Product'].unique()
```

[15]:
```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

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

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

[17]:
```
df['Product'].unique().tolist()
```

[17]:
```
['KP281', 'KP481', 'KP781']
```

[ ]:
```
df['Product'].value_counts().sort_index()
```

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

[18]:
```
df['Usage'].nunique()
```

[18]:
```
6
```

[19]:
```
df['Usage'].unique()
```

[19]:
```
array([3, 2, 4, 5, 6, 7])
```

[20]:
```
df['Usage'].unique().tolist()
```

[20]:
```
[3, 2, 4, 5, 6, 7]
```

[21]:
```
df['Usage'].value_counts()
```

[21]:
```
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
```

[ ]:
```
df['Usage'].value_counts().sort_index()
```

[ ]:
```
2    33
3    69
4    52
5    17
6     7
```

```
7     2
Name: Usage, dtype: int64
```

[23]:
```
df['MaritalStatus'].nunique()
```

[23]:
```
2
```

[24]:
```
df['MaritalStatus'].unique()
```

[24]:
```
array(['Single', 'Partnered'], dtype=object)
```

[25]:
```
df['MaritalStatus'].unique().tolist()
```

[25]:
```
['Single', 'Partnered']
```

[26]:
```
df['MaritalStatus'].value_counts().sort_index()
```

[26]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

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

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

Summary * **KP281, KP481, KP781** are the 3 different products * There are **32** unique ages * Majority of the customers who have purchased are **Married/Partnered** * **104 Males** and **76 Females** are in the customers list

conversion of categorical attributes to 'category'

[27]:
```
category = df
category['Fitness_category'] = df.Fitness
category.head()
```

[27]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_category
0	KP281	18	Male	14	Single	3	4	29562	112	4
1	KP281	19	Male	15	Single	2	3	31836	75	3
2	KP281	19	Female	14	Partnered	4	3	30699	66	3
3	KP281	19	Male	12	Single	3	3	32973		
4	KP281	20	Male	13	Partnered	4	2	35247		

```
3     85                3
4     47                2
```

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

```
[ ]:   Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles Fitness_category
0    112              Bad
1     75          Average
2     66          Average
3     85          Average
4     47             Good
```

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

```
[ ]:              Age   Education       Usage     Fitness         Income  \
count  180.000000  180.000000  180.000000  180.000000     180.000000
mean    28.788889   15.572222    3.455556    3.311111   53719.577778
std      6.943498    1.617055    1.084797    0.958869   16506.684226
min     18.000000   12.000000    2.000000    1.000000   29562.000000
25%     24.000000   14.000000    3.000000    3.000000   44058.750000
50%     26.000000   16.000000    3.000000    3.000000   50596.500000
75%     33.000000   16.000000    4.000000    4.000000   58668.000000
max     50.000000   21.000000    7.000000    5.000000  104581.000000

            Miles
count  180.000000
mean   103.194444
std     51.863605
min     21.000000
25%     66.000000
50%     94.000000
75%    114.750000
max    360.000000
```

Mean Age of the given customer dataset is **28.78**

Minimum Age of the customer starts from **18** and **maximum age** is **50**

Average usage per week for a customer is **3** days

Average Fitness rating is 3 with most common fitness rating is **4**

[28]:
```
statistical = df['Product'].value_counts(normalize=True)
stat = statistical.map(lambda calc: round(100*calc,2))
stat
```

[28]:
```
KP281    44.44
KP481    33.33
KP781    22.22
Name: Product, dtype: float64
```

[29]:
```
gender = df['Gender'].value_counts(normalize=True)
gender_result = gender.map(lambda calc: round(100*calc,2))
gender_result
```

[29]:
```
Male      57.78
Female    42.22
Name: Gender, dtype: float64
```

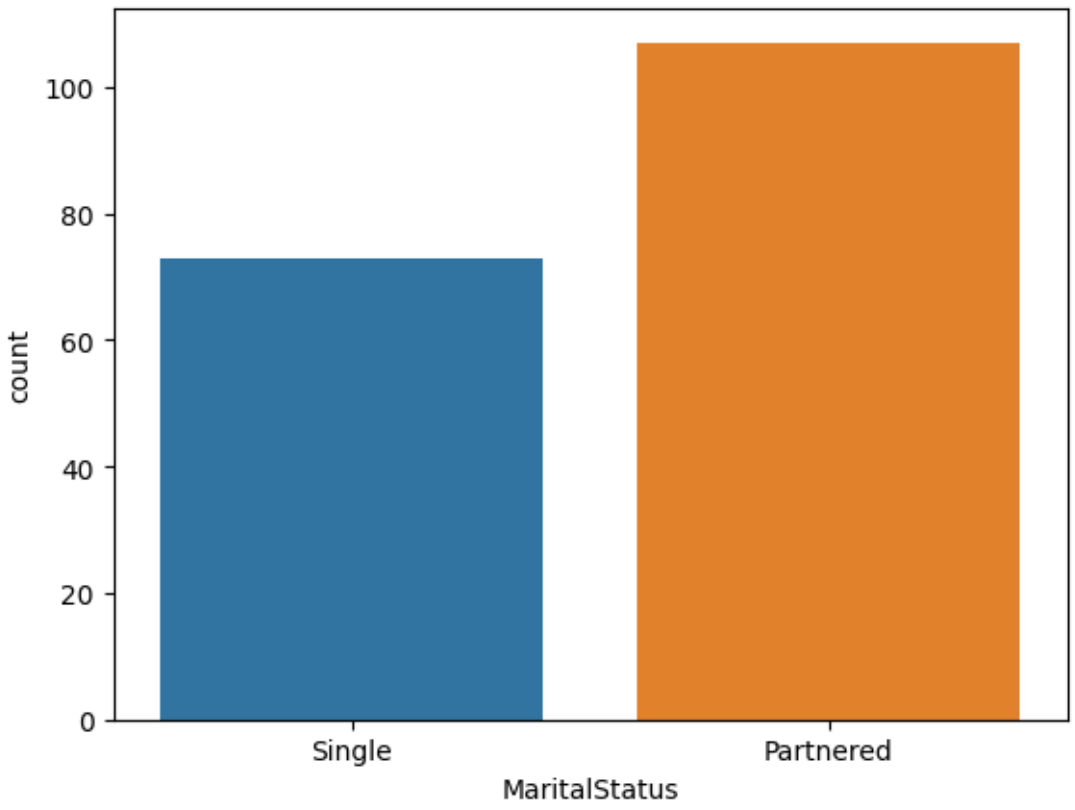
[30]:
```
marital_status = df['MaritalStatus'].value_counts(normalize=True)
marital_status_result = marital_status.map(lambda calc:round(100*calc,2))
marital_status_result
```

[30]:
```
Partnered    59.44
Single       40.56
Name: MaritalStatus, dtype: float64
```

3) Visual Analysis - Univariate & Bivariate

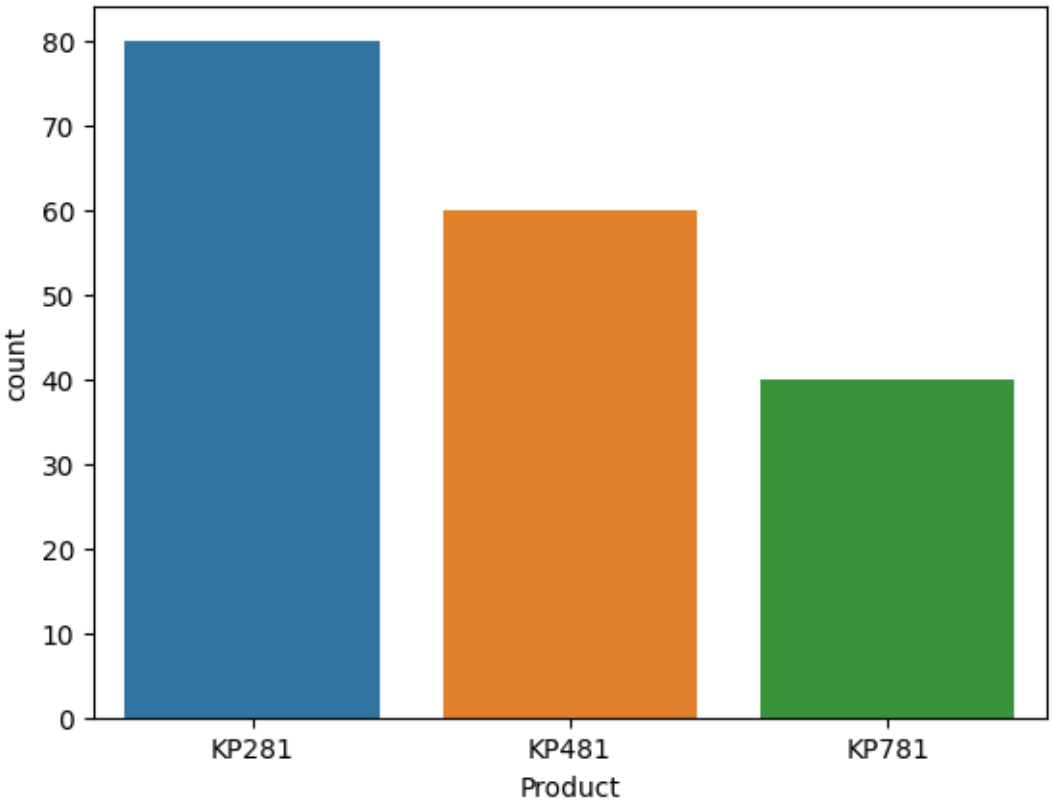
1. For continuous variable(s): Distplot, countplot, histogram for univariate analysis
2. For categorical variable(s): Boxplot
3. For correlation: Heatmaps, Pairplots

[ ]:
```
sns.countplot(data=df,x='MaritalStatus')
plt.show()
```



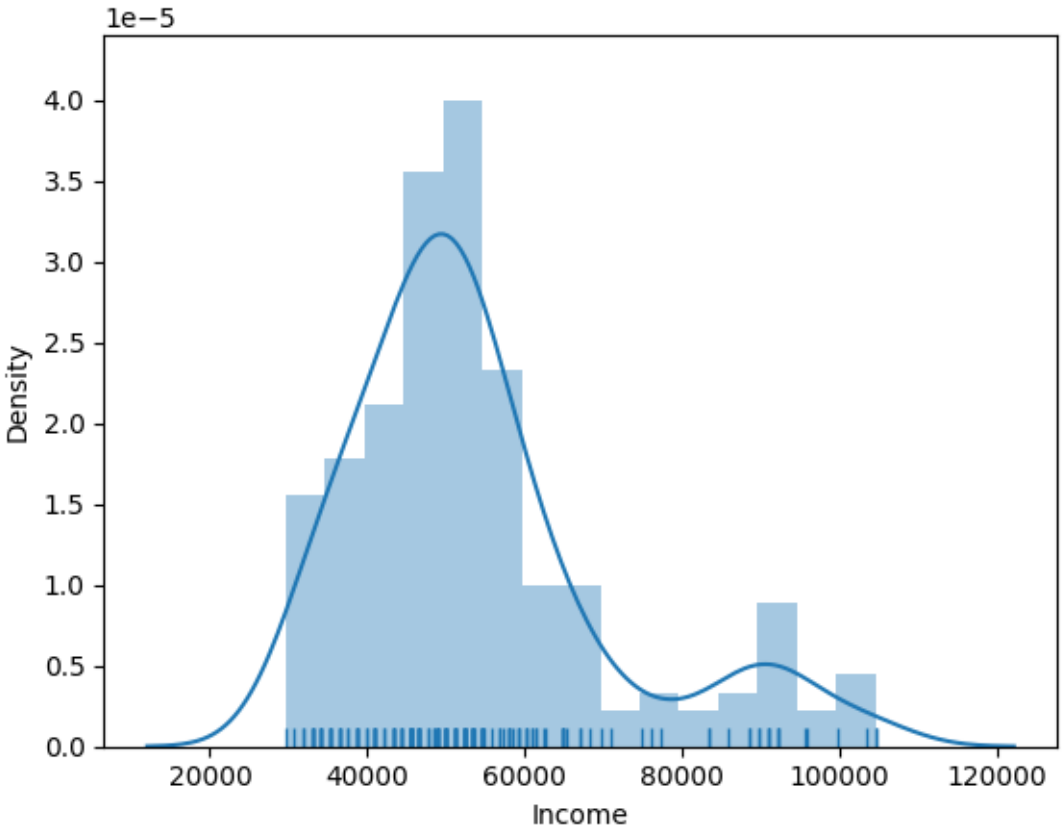
[ ]:
```
sns.countplot(data=df,x='Product')
plt.show
```

[ ]: <function matplotlib.pyplot.show(close=None, block=None)>



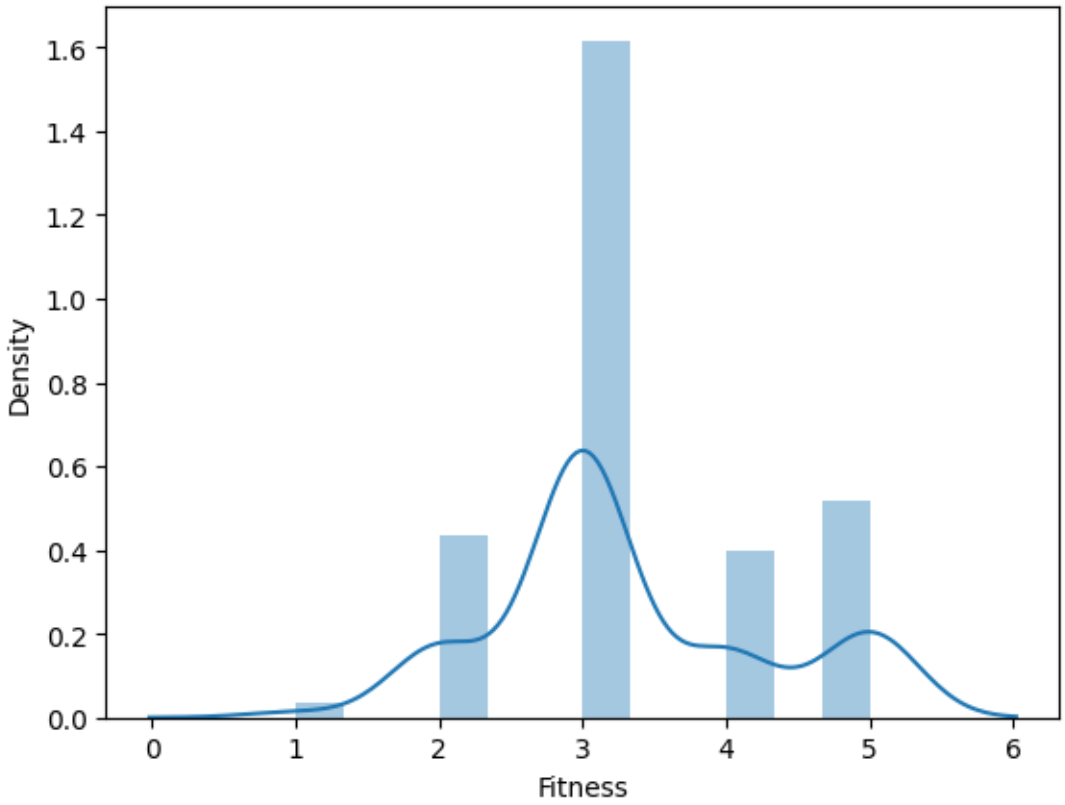
[ ]:
```
sns.distplot(df.Income,rug=True)
plt.show()
```

```
<ipython-input-36-750d3bf61763>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Income,rug=True)
```



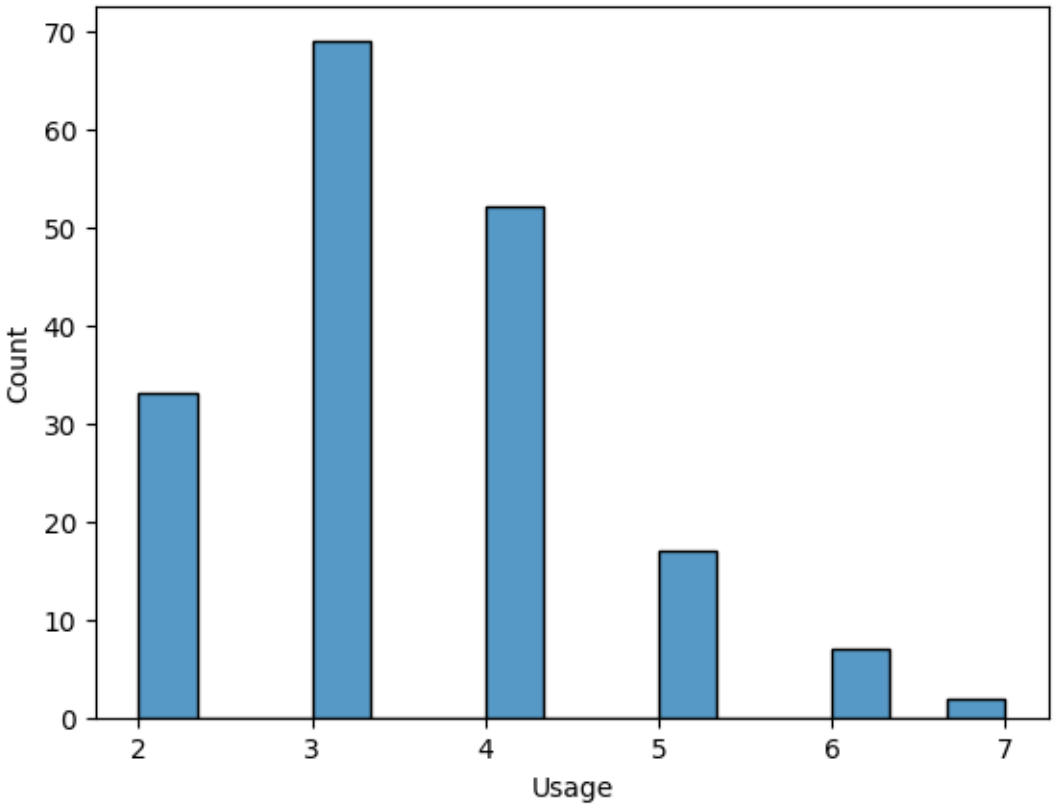
[ ]:
```
sns.distplot(df.Fitness)
plt.show()
```

```
<ipython-input-37-d101120d52c8>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Fitness)
```

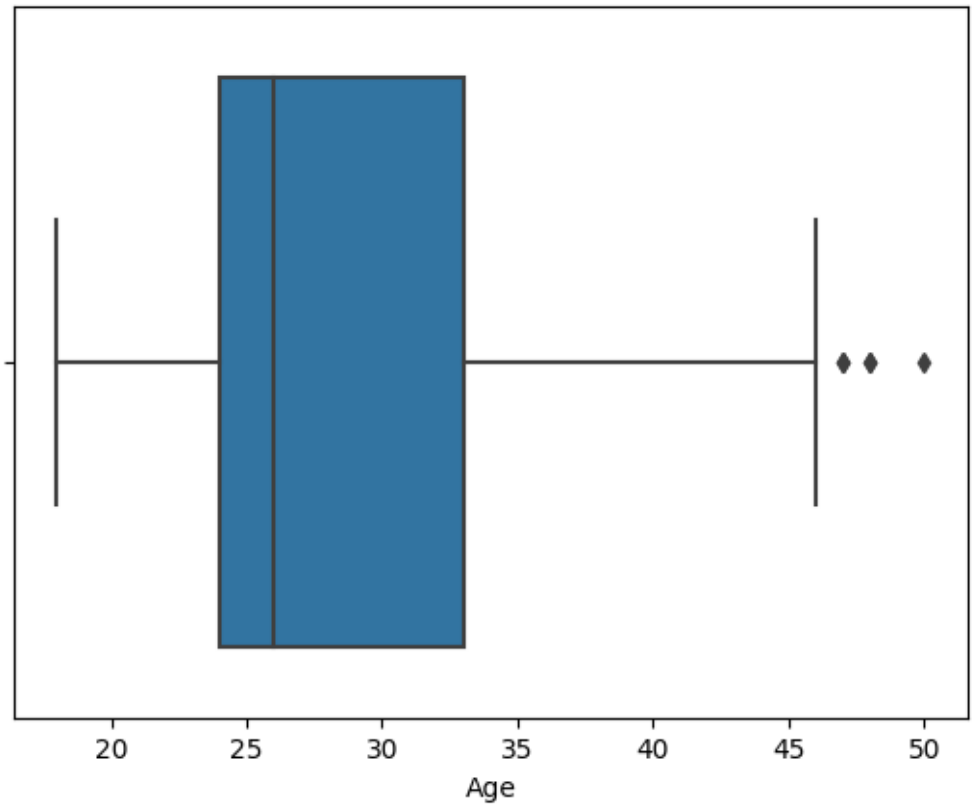


[ ]:
```
sns.histplot(data=df,x='Usage')
```

[ ]:
```
<Axes: xlabel='Usage', ylabel='Count'>
```

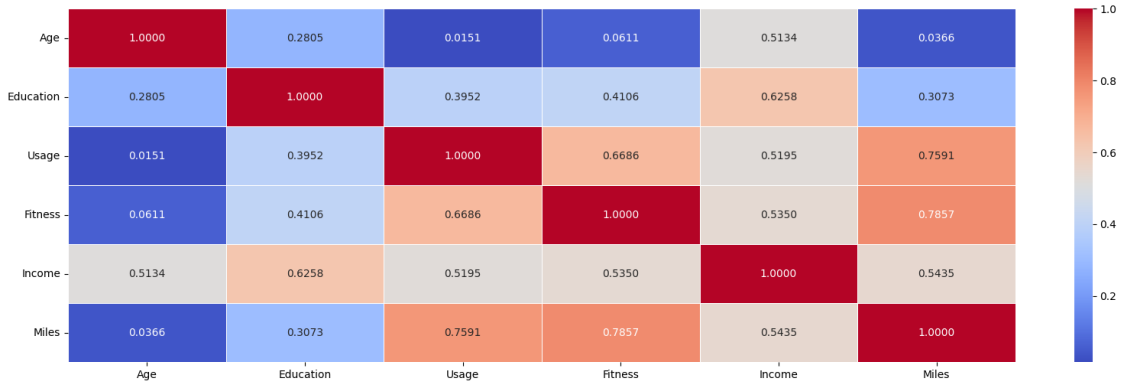


[ ]:
```
sns.boxplot(data=df,x='Age')
plt.show()
```



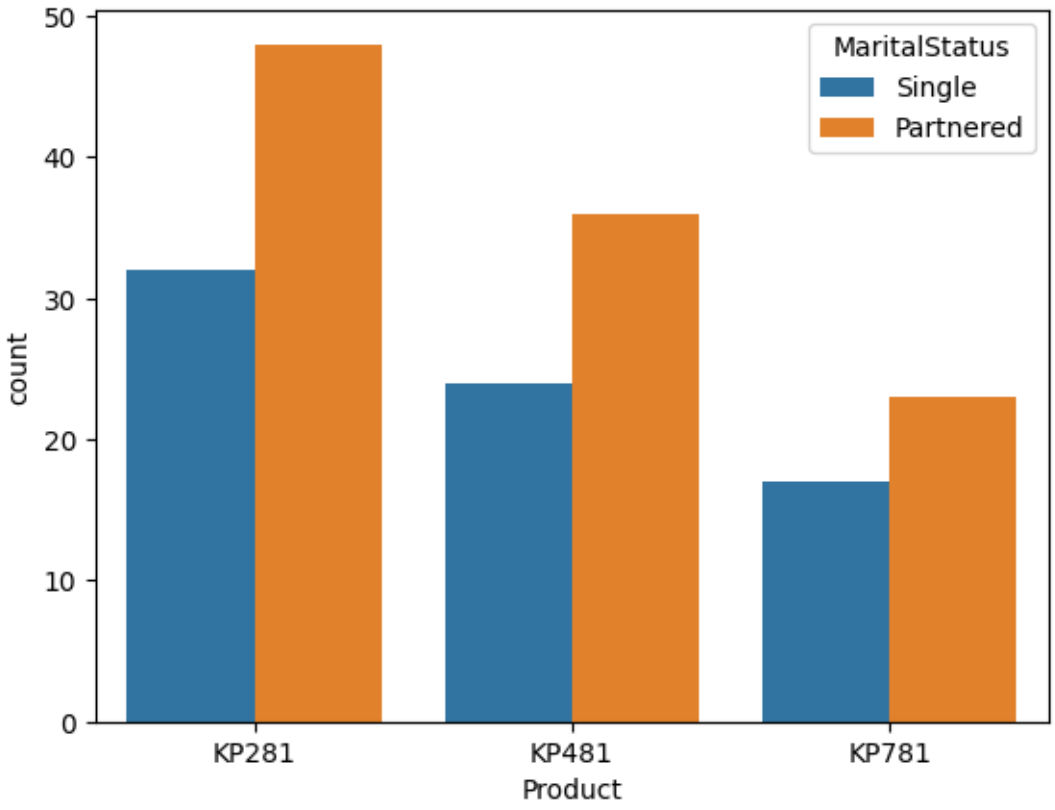
[ ]:
```
plt.figure(figsize=(20,6))
ax = sns.heatmap(df.corr(),annot=True,fmt='.4f',linewidths=.5,cmap='coolwarm')
plt.yticks(rotation=0)
plt.show()
```

```
<ipython-input-40-aabeea718d34>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  ax = sns.heatmap(df.corr(),annot=True,fmt='.4f',linewidths=.5,cmap='coolwarm')
```

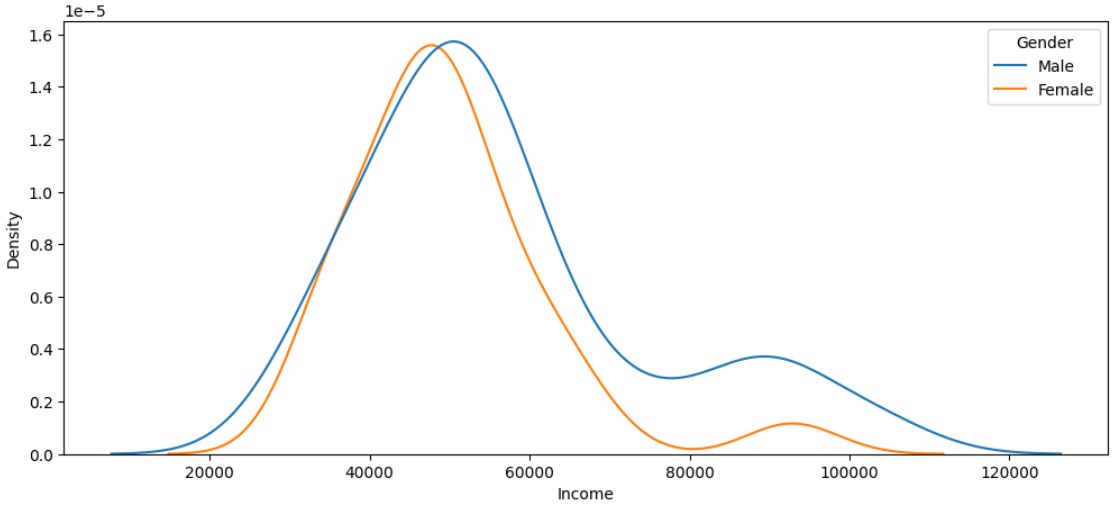


[ ]:
```
sns.pairplot(df,hue='MaritalStatus',kind='reg')
plt.show()
```

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



[ ]:
```
plt.figure(figsize=(12,5))
sns.kdeplot(data=df,x='Income',hue='Gender')
plt.show()
```



4) Missing Value & Outlier Detection

[ ]:
```
df.isna().sum()
```

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

[ ]:
```
df.duplicated().sum()
```

[ ]:
```
0
```

5) Business Insights based on Non-Graphical and Visual Analysis

1. Comments on the range of attributes
2. Comments on the distribution of the variables and relationship between them
3. Comments for each univariate and bivariate plot

[ ]:
```
category.head()
```

[ ]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles Fitness_category
0    112              Bad
1     75          Average
2     66          Average
3     85          Average
4     47             Good
```

[ ]:
```
category['age_group'] = category.Age
category.head()
```

[ ]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
```

```
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles Fitness_category  age_group
0    112              Bad         18
1     75          Average         19
2     66          Average         19
3     85          Average         19
4     47             Good         20
```

[ ]:
```
category.age_group = pd.cut(category.
↪age_group,bins=[0,21,35,45,60],labels=['Teen','Adult','Middle Aged','Elder'])
```

[ ]:
```
category.head()
```

[ ]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles Fitness_category age_group
0    112              Bad      Teen
1     75          Average      Teen
2     66          Average      Teen
3     85          Average      Teen
4     47             Good      Teen
```

1)Comments on the range of attributes

[ ]:
```
'''#we use attributes such as fitness category, age category
by using the newly created variable: category['age_group'] = category.Age'''
'''#also we are used different types of plots
such as box plot, pair plot, test plot, countplot, etc.,
for analysing the data'''
```

2) Comments on the distribution of the variables and relationship between them

[ ]:
```
'''#The given dataset aerofit for distribution
of the varibles, I used different plots
to get understand of the relationship
between different variables by creating
new variables called age category,
fitness category and so on
loaded in the dataset
to find out the customers
```

```
who are all comes under which category and analyse
their behaviour'''
```

3) Comments for each univariate and bivariate plot

[ ]:
```
'''#The univariate plots different observations/values of the
same variable corresponding to the index/observation number.
for univariate analysis, I used countplot for
marital status & product
for boxplot I used age
same for distplot fitness and income
and for histplot used the usage variable
to analyse the data and customer behaviours'''

'''#Bivariate analysis is the simultaneous analysis of two variables.
It explores the concept of the relationship between two variable
whether there exists an association
and the strength of this association or
whether there are differences between two variables
and the significance of these differenc
for correaltion we used headtmap
and used the pairplot to
analyse the marital status
using the countplot to
analyse product & marital status variation
and kedplot to analyse the income and gender'''
```

6) **Recommendations**

Actionable items for business. No technical jargon. No complications. Simple action items that everyone can understand

[ ]:
```
'''#KP281 & KP481 are the products
most liked by the customers where
their income lies in the range of 39K - 53K Dollars.
recommending to target the Age above 40 years to get the good sales
in this product KP781 because mostly those age people
have some things to maintain their body health
but if you see in th eage range of youngsters
they simply maintain their health good
and also fit, so we need to target only those people
and recommend to target the female customers also
mainly provide the good customer support
to reach a good height in business'''
```