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

print('Import done')
```

```
Import done
```

```python
# Link to the CSV file

url =
'https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/12
5/original/aerofit_treadmill.csv?1639992749'
df = pd.read_csv(url)
df
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 180,\n  \"fields\": [\
n    {\n      \"column\": \"Product\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"KP281\",\n          \"KP481\",\n
\"KP781\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Age\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 6,\n        \"min\": 18,\n        \"max\": 50,\n
\"num_unique_values\": 32,\n        \"samples\": [\n          45,\n
33,\n          43\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Gender\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"Female\",\n          \"Male\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Education\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1,\n        \"min\": 12,\n        \"max\": 21,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          15,\n
18\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"MaritalStatus\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"Partnered\",\n          \"Single\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Usage\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 1,\n        \"min\":
2,\n        \"max\": 7,\n        \"num_unique_values\": 6,\n
\"samples\": [\n          3,\n          2\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Fitness\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 0,\n
\"min\": 1,\n        \"max\": 5,\n        \"num_unique_values\": 5,\n
```

```
\"samples\": [\n          3,\n          5\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Income\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\": 16506,\n
\"min\": 29562,\n        \"max\": 104581,\n
\"num_unique_values\": 62,\n        \"samples\": [\n          88396,\n
103336\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Miles\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 51,\n        \"min\": 21,\n        \"max\": 360,\n
\"num_unique_values\": 37,\n        \"samples\": [\n          95,\n
169\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```


```python
# Calculate the mean of each numerical column
mean_income = df['Income'].mean()
mean_steps = df['Miles'].mean()
mean_age = df['Age'].mean()
mean_usage = df['Usage'].mean()
mean_fitness = df['Fitness'].mean()
mean_education = df['Education'].mean()

# Print the results
print("Mean Income:", mean_income)
print("Mean Steps Walked (Miles):", mean_steps)
print("Mean Age:", mean_age)
print("Mean Usage:", mean_usage)
print("Mean Fitness:", mean_fitness)
print("Mean Education:", mean_education)
```

```
Mean Income: 53719.57777777778
Mean Steps Walked (Miles): 103.19444444444444
Mean Age: 28.788888888888888
Mean Usage: 3.4555555555555557
Mean Fitness: 3.311111111111111
Mean Education: 15.572222222222223
```

Observations After looking at the Dataset and calculating Mean

1. Mean Income: 53,719 USD
2. Mean Steps Walked (Miles): 103
3. Mean Age: 28.7 years
4. Mean Usage: 3.4 per week

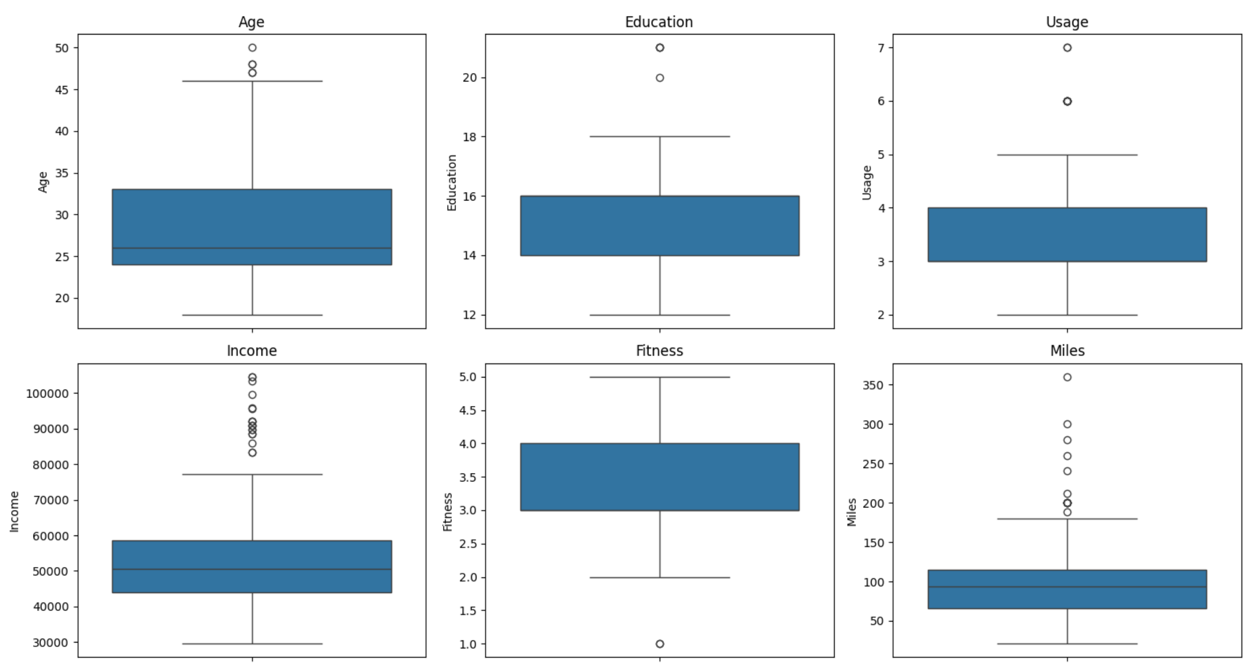
5. Mean Fitness: 3.3/5.0
6. Mean Education: 15.6 years

Checking for Missing Values and Outliers

```
# Checking for missing values
missing_values = df.isnull().sum()
print("Missing Values:")
print(missing_values)

# Visualizing missing values
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False)
plt.title('Missing Values Heatmap')
plt.show()

# Detecting outlier for continuous variables
continuous_vars = ['Age', 'Education', 'Usage', 'Income', 'Fitness',
'Miles']

plt.figure(figsize=(15, 8))
for i, col in enumerate(continuous_vars, 1):
    plt.subplot(2, 3, i)
    sns.boxplot(y=df[col])
    plt.title(col)
plt.tight_layout()
plt.show()
```

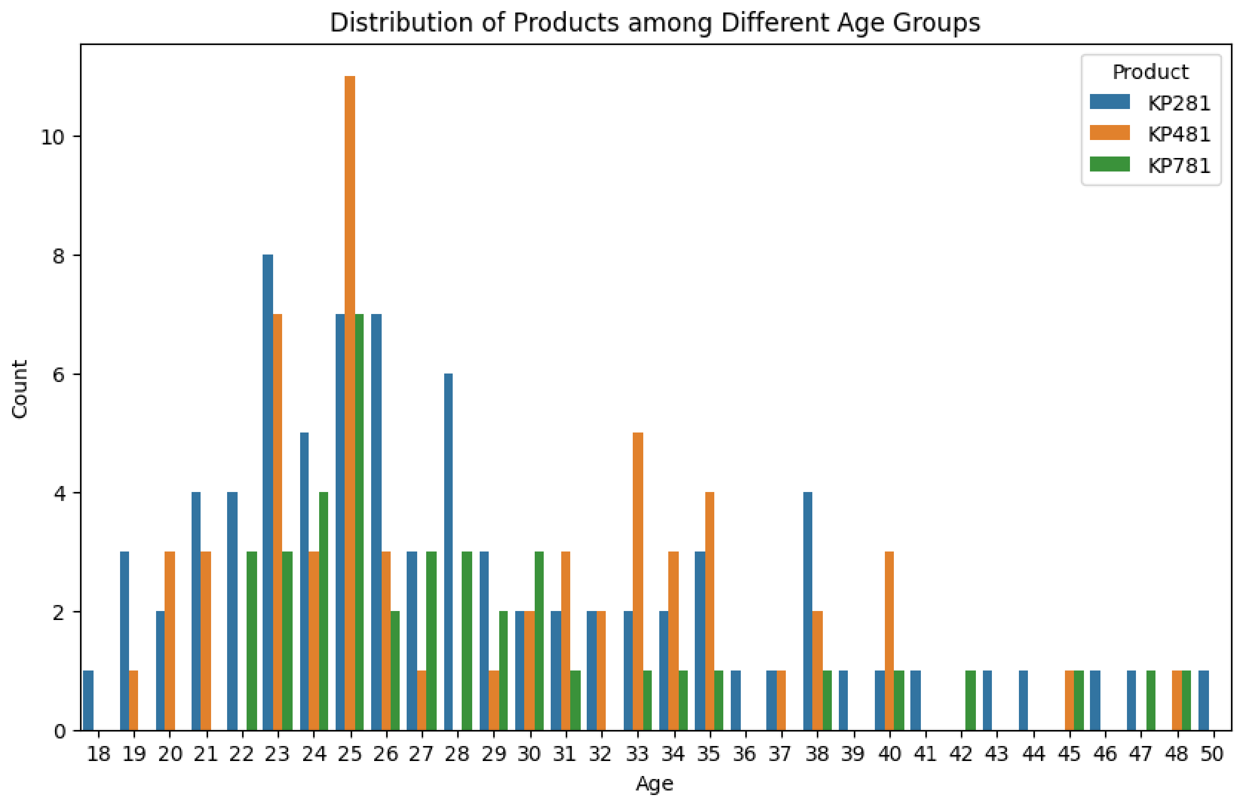
```
Missing Values:
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```


Missing Values Heatmap
0
6
12
18
24
30
36
42
48
54
60
66
72
78
84
90
96
102
108
114
120
126
132
138
144
150
156
162
168
174
Product
Age
Gender
Education
MaritalStatus
Usage
Fitness
Income
Miles


Age
Education
Usage
Income
Fitness
Miles

Distrubution of products among Different Age Groups(Bar Plot)

```
# Distrubution of products among Different Age Groups(Bar Plot)

plt.figure(figsize=(10, 6))
sns.countplot(data = df, x = 'Age', hue = 'Product')
plt.title('Distribution of Products among Different Age Groups')
plt.xlabel('Age')
plt.ylabel('Count')
plt.legend(title='Product')
plt.show()
```



No.of Unique Products in the Dataset

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

# Therefore, there are 3 unique products in the DataFrame
```

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

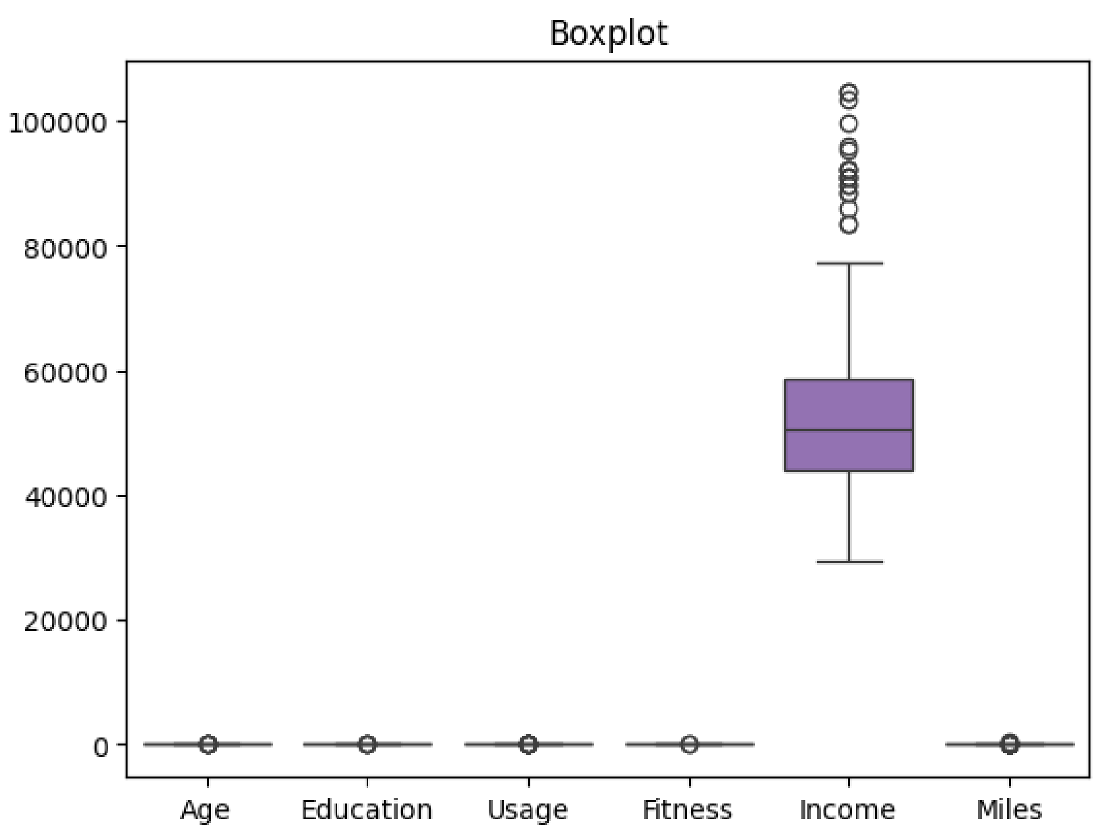
```
df.describe(include = "all")
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 11,\n  \"fields\": [\n    {\n      \"column\": \"Product\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 4,\n        \"samples\": [\n          3,\n          \"80\",\n          \"180\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 55.58832332198464,\n        \"min\": 6.943498135399795,\n        \"max\": 180.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n          28.788888888888888,\n          26.0,\n          180.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Gender\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 4,\n        \"samples\": [\n          2,\n          \"104\",\n          \"180\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Education\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 59.04362112875324,\n        \"min\": 1.6170548978065569,\n        \"max\": 180.0,\n        \"num_unique_values\": 7,\n        \"samples\": [\n          180.0,\n          15.572222222222223,\n          16.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"MaritalStatus\",\n      \"properties\": {\n        \"dtype\": \"category\",\n        \"num_unique_values\": 4,\n        \"samples\": [\n          2,\n          \"107\",\n          \"180\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Usage\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 62.474604277313155,\n        \"min\": 1.0847970343962436,\n        \"max\": 180.0,\n        \"num_unique_values\": 7,\n        \"samples\": [\n          180.0,\n          3.4555555555555557,\n          4.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Fitness\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 62.63086276036247,\n        \"min\": 0.958868565619312,\n        \"max\": 180.0,\n        \"num_unique_values\": 7,\n        \"samples\": [\n          180.0,\n          3.311111111111111,\n          4.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Income\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 31403.855763201762,\n        \"min\": 180.0,\n        \"max\": 104581.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n          53719.57777777778,\n          50596.5,\n          180.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Miles\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 106.52090041797726,\n        \"min\": 21.0,\n        \"max\": 360.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n          103.19444444444444,\n          94.0,\n          180.0\n        ],\n
```

\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

BoxPlot(Pair Plot) and the Difference in Mean and Median for each column

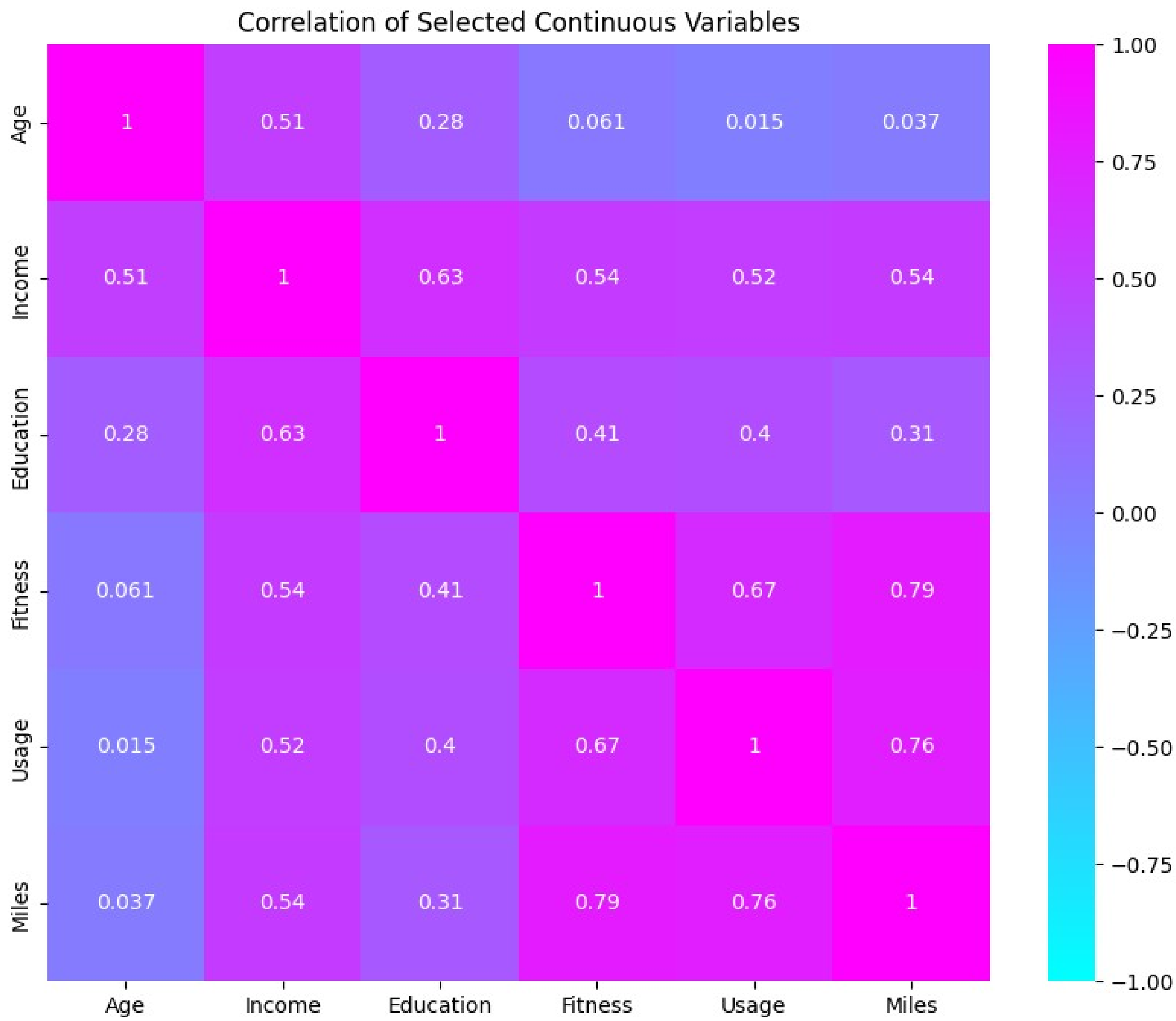
```
# BoxPlot(Pair Plot to check correlation among different factors)
sns.boxplot(data = df)
plt.title('Boxplot')
plt.show()

# Difference between MEAN and MEDIAN for each numerical column
summary_stats = df.describe()

difference = summary_stats.loc['mean'] - summary_stats.loc['50%']
print("Difference between the MEAN and MEDIAN : ")
print(difference)
```



```
Difference between the MEAN and MEDIAN :
Age             2.788889
Education      -0.427778
Usage           0.455556
Fitness         0.311111
Income       3123.077778
Miles           9.194444
dtype: float64
```

Heat Map to check correlation among different factors

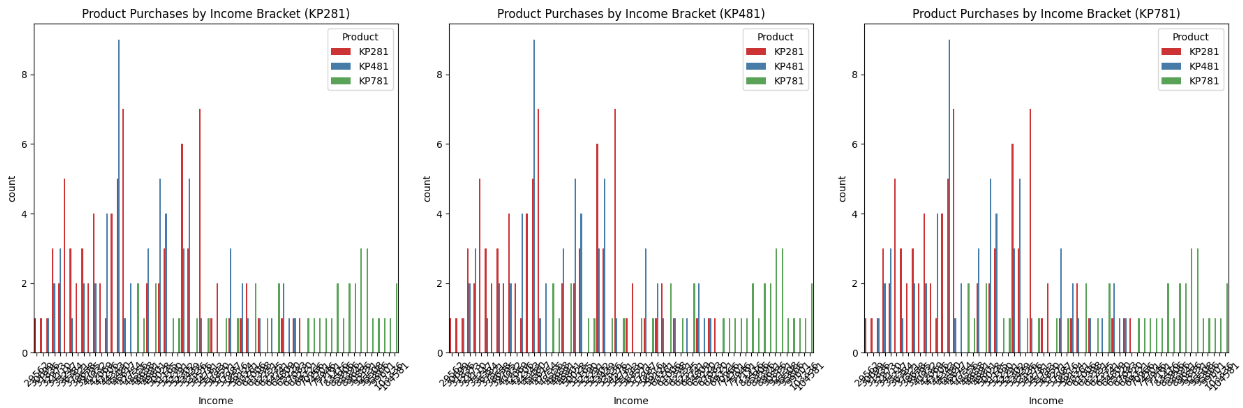
```
continuous_columns = ["Age", "Income", "Education", "Fitness",
"Usage",  "Miles"]

# Calculating correlation matrix
correlation_matrix = df[continuous_columns].corr()

# Plotting the heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(data = correlation_matrix, annot = True, cmap = 'cool',
vmin = -1, vmax = 1)
plt.title("Correlation of Selected Continuous Variables")
plt.show()

# vmin and vmax are used to set the MINIMUM and MAXIMUM values of the
color scale
```



Individual Bar Plots for each product

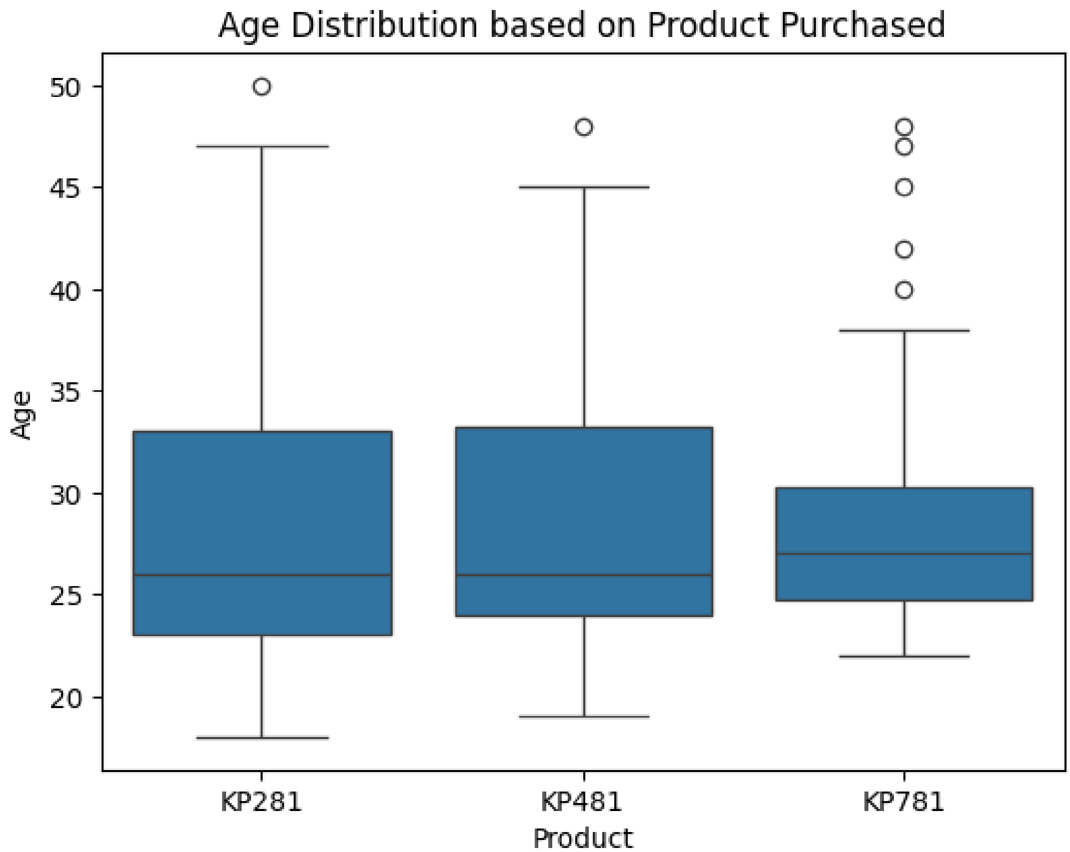
```
# Individual Bar Plots for each product
plt.figure(figsize=(18, 6))

# Bar Plot for KP281
plt.subplot(1, 3, 1)
sns.countplot(data = df, x = 'Income', hue = 'Product', palette =
'Set1')
plt.title('Product Purchases by Income Bracket (KP281)')
plt.xticks(rotation=45)

# Bar Plot for KP481
plt.subplot(1, 3, 2)
sns.countplot(data = df, x = 'Income', hue = 'Product', palette =
'Set1')
```

```
plt.title('Product Purchases by Income Bracket (KP481)')
plt.xticks(rotation=45)

# Bar Plot for KP781
plt.subplot(1, 3, 3)
sns.countplot(data = df, x = 'Income', hue = 'Product', palette =
'Set1')
plt.title('Product Purchases by Income Bracket (KP781)')
plt.xticks(rotation=45)

# Display
plt.tight_layout()
plt.show()
```



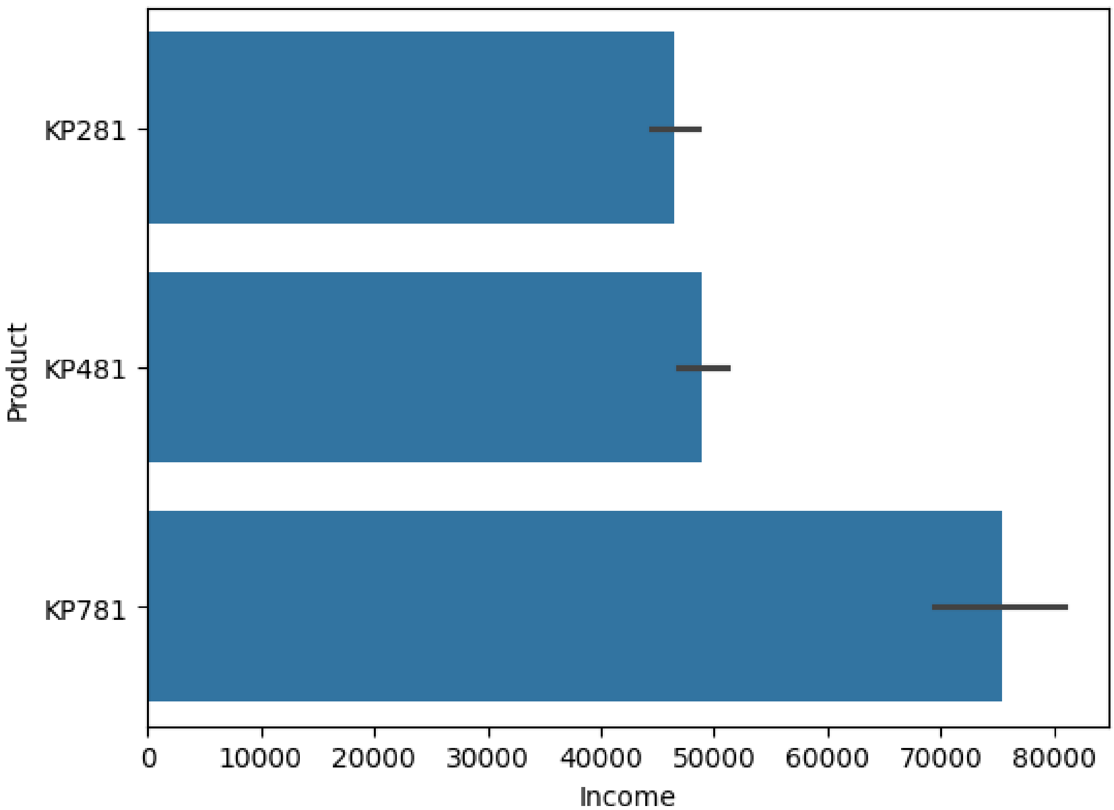
BoxPlot for Age

```
# BoxPlot for Age

sns.boxplot(x = 'Product', y = 'Age', data = df)
plt.title('Age Distribution based on Product Purchased')
plt.show()
```



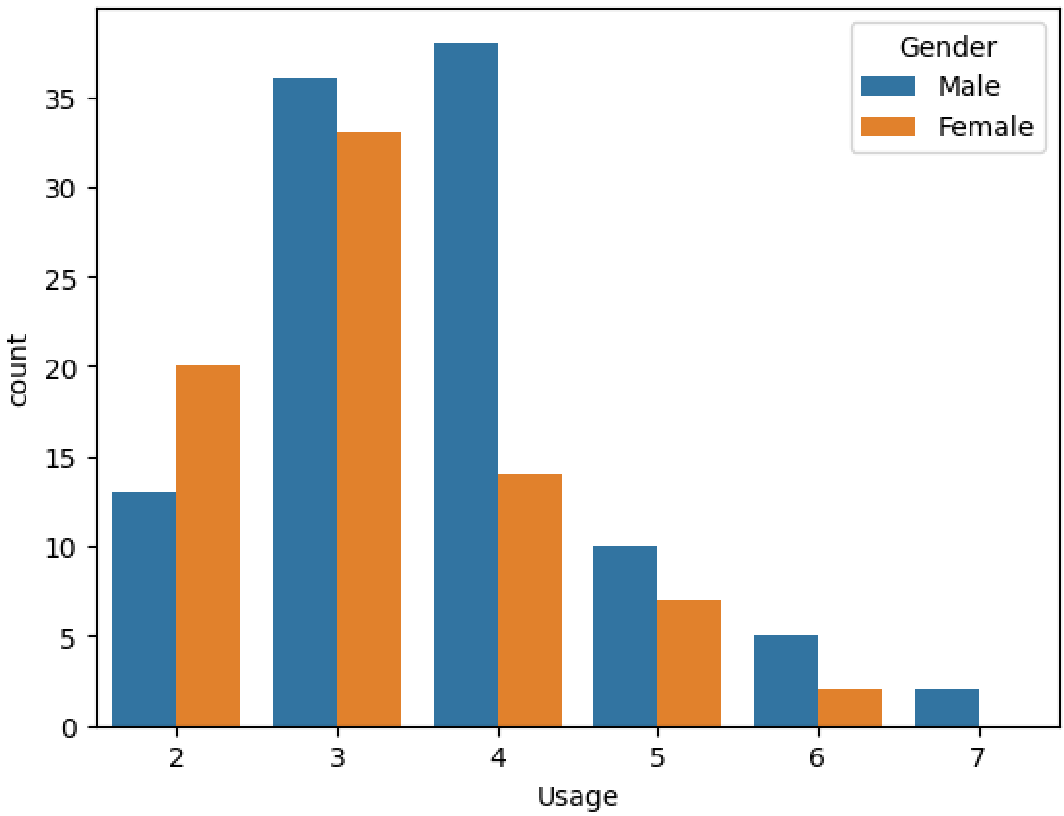
Income-Product Plot

```
# Different Plots


# 1.) Income-Product Plot
sns.barplot(data = df, x='Income', y='Product')
plt.show()
```



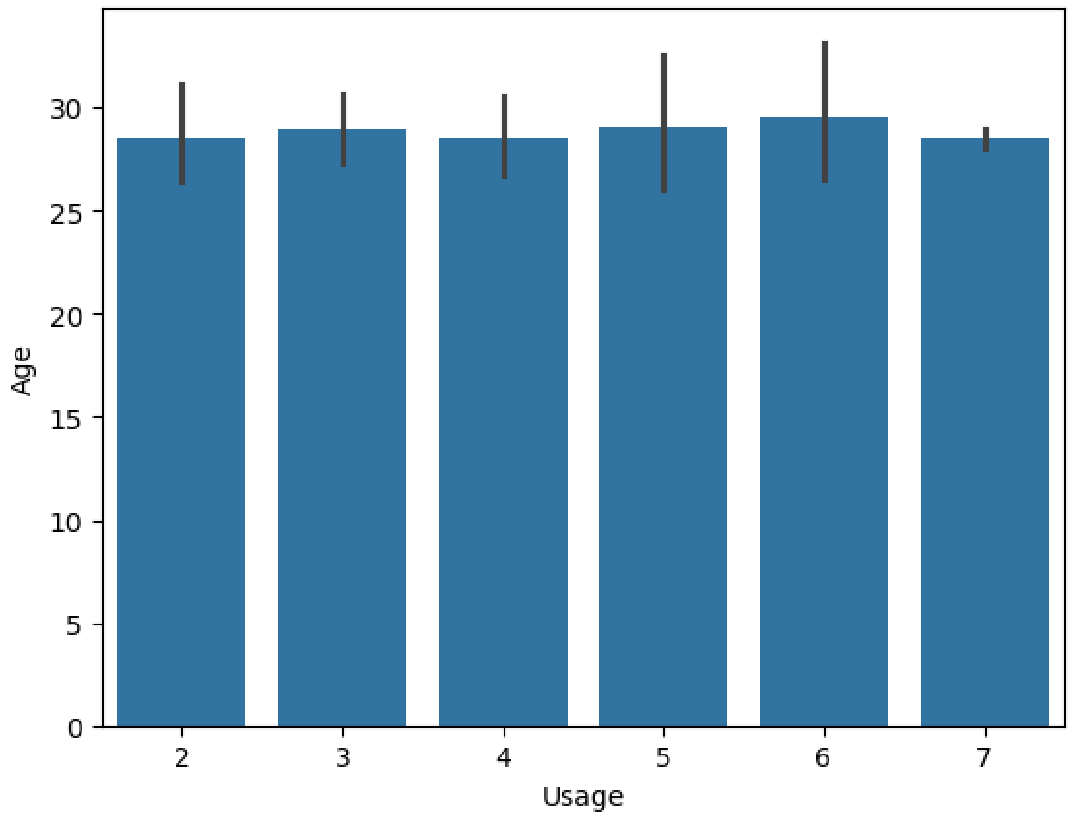
Product-Gender

```
# 2.) Product-Gender Plot
sns.countplot(data = df, x = 'Usage', hue = 'Gender')
plt.show()
```



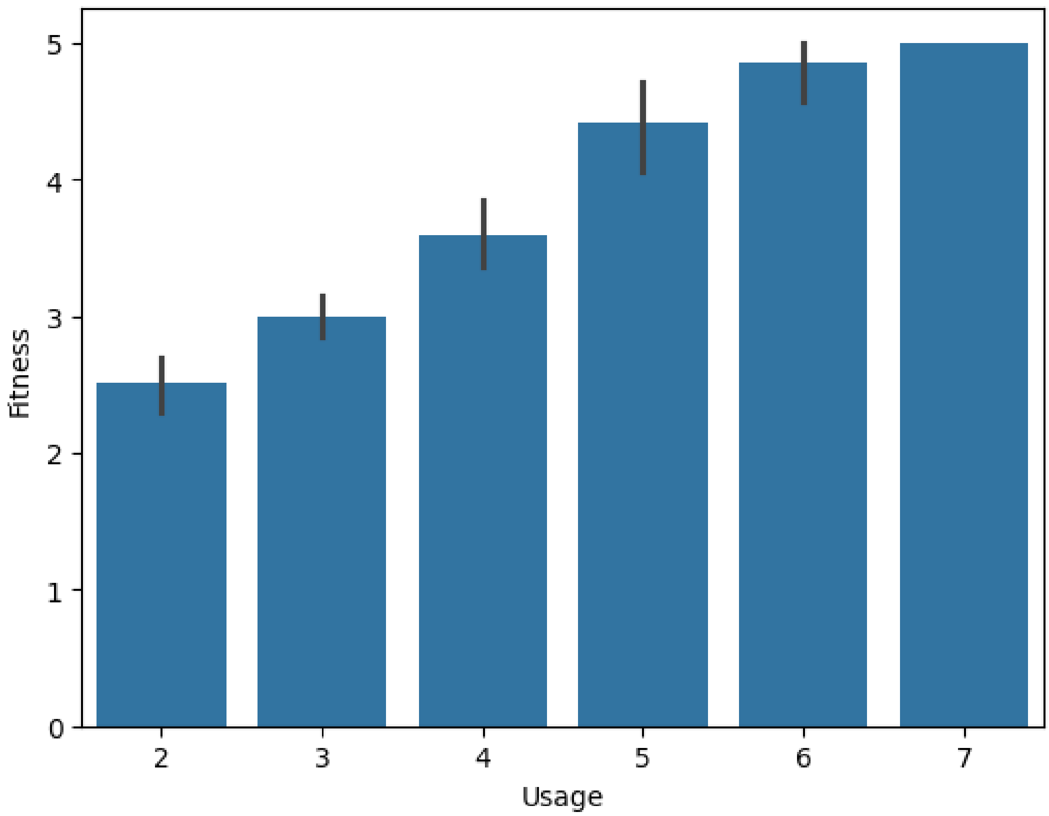
Usage-Age Plot

```
# 3.) Usage-Age Plot
sns.barplot(data = df, x = 'Usage', y = 'Age')
plt.show()
```



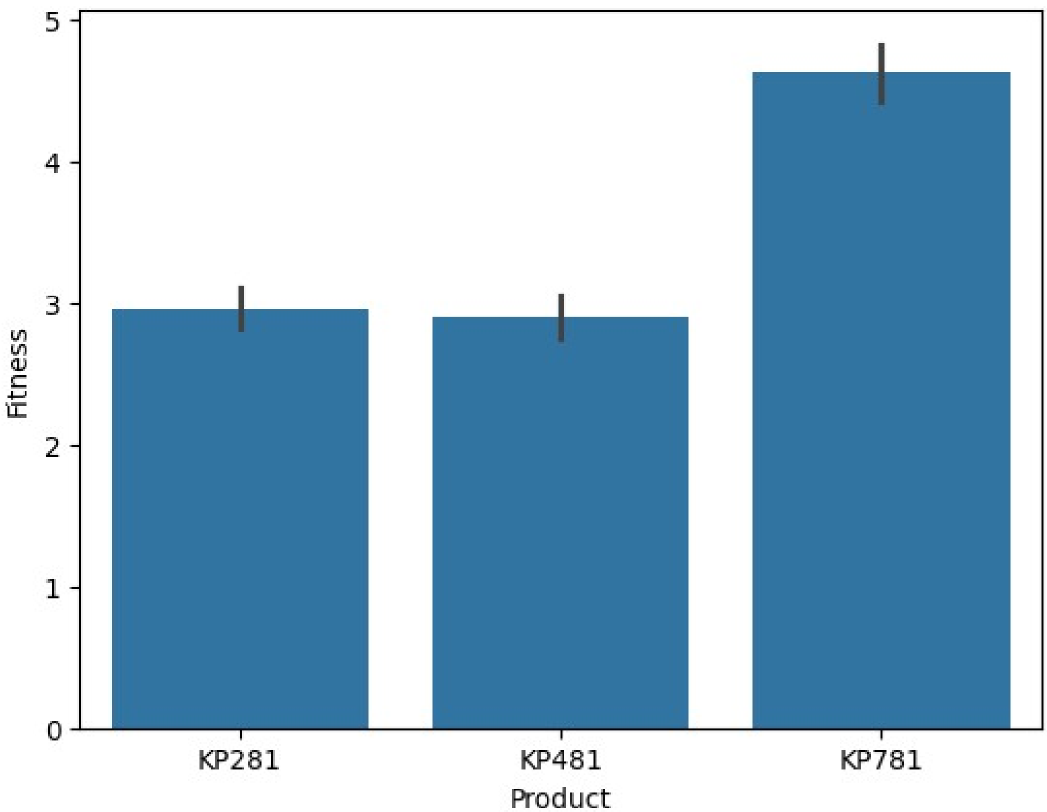
Usage-Fitness Plot

```
# 4.) Usage-Fitness Plot
sns.barplot(data = df, x = 'Usage', y = 'Fitness')
plt.show()
```



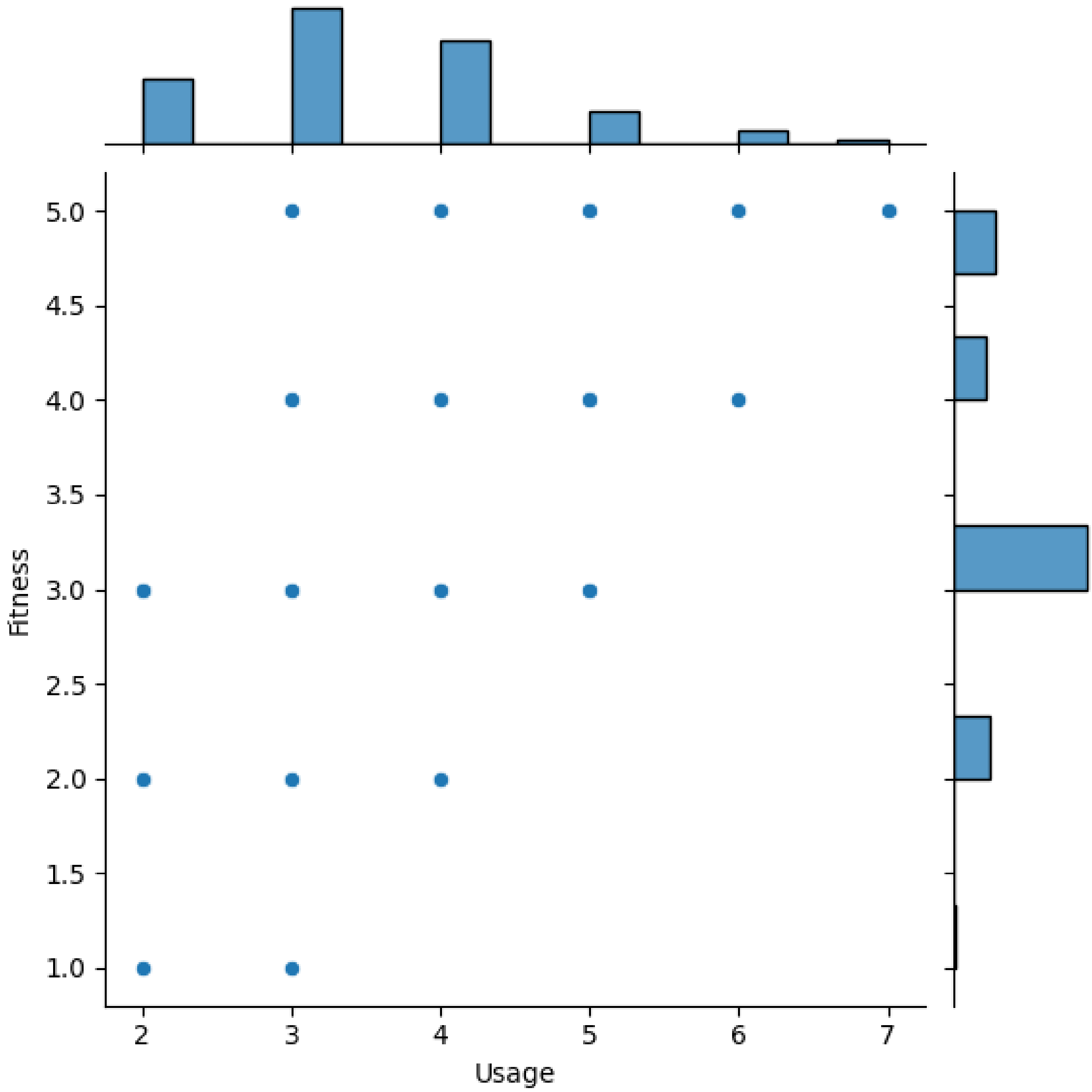
Product-Fitness Plot

```
# 5.) Product-Fitness Plot
sns.barplot(data = df, x = 'Product', y = 'Fitness')
plt.show()
```



Usage-Fitness Scatter Plot

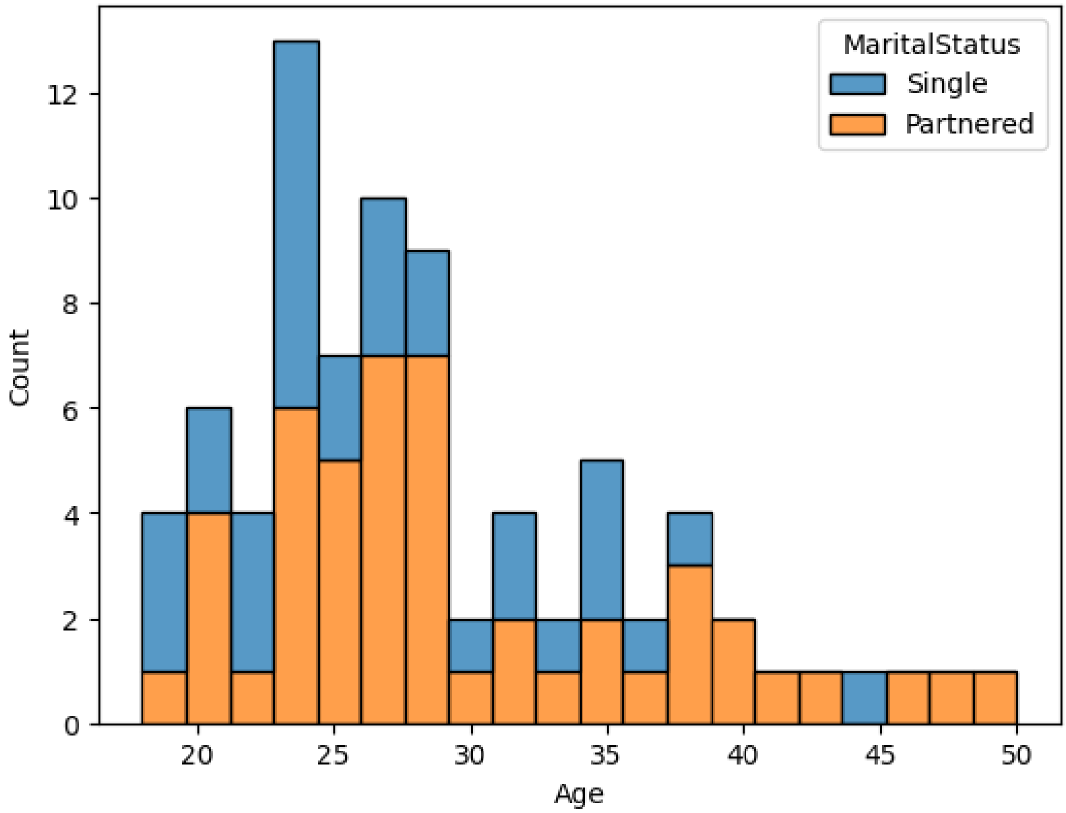
```
# 6.) Usage-Fitness Scatter Plot
sns.jointplot(data = df, x = 'Usage', y = 'Fitness', kind = 'scatter')
plt.show()
```

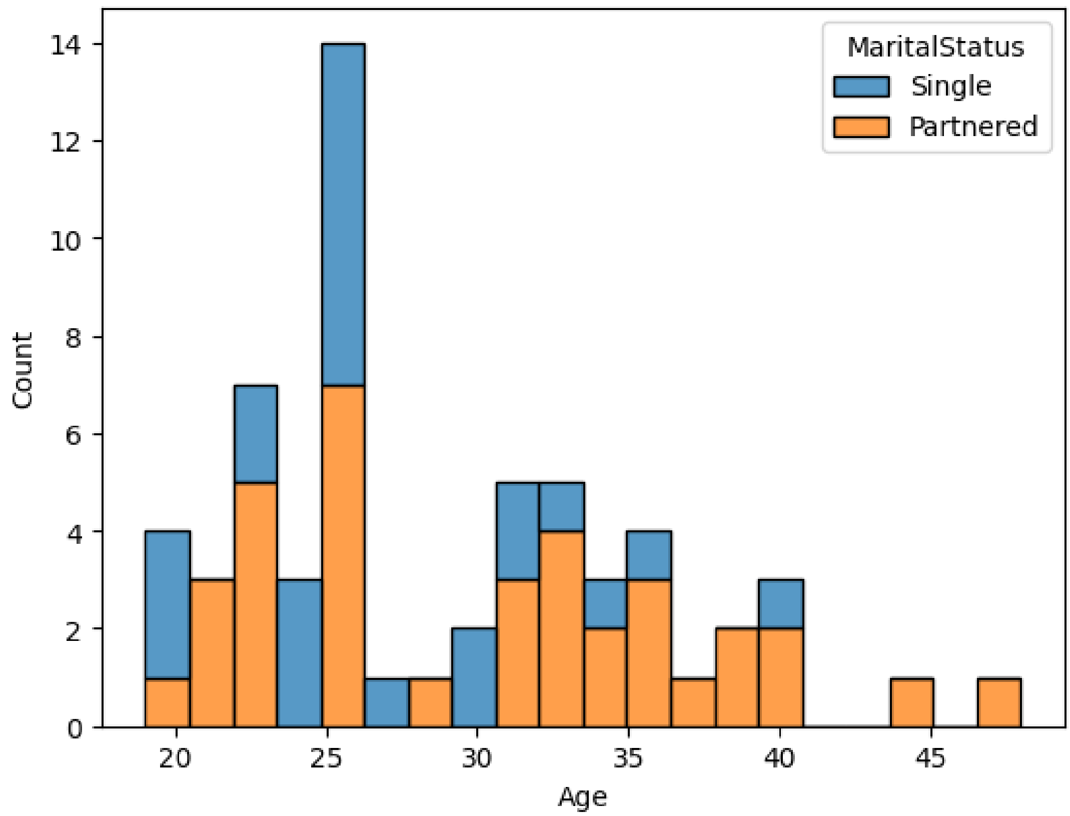


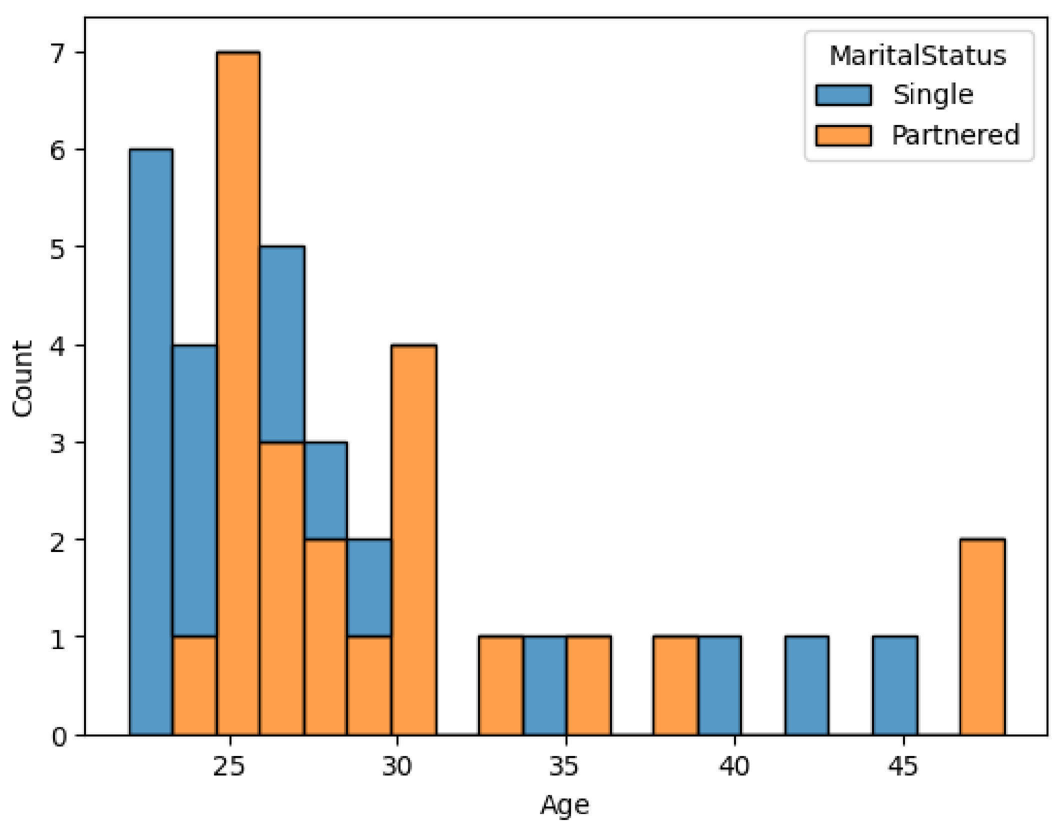
HistPlots for different types of Products(Treadmills - KP218, KP418, KP718)

```
# HistPlots for different Treadmills

# KP-281
sns.histplot(data = df[df['Product'] == 'KP281'], x = 'Age', hue =
'MaritalStatus', multiple = 'stack', bins=20)
plt.show()

# KP-481
sns.histplot(data = df[df['Product'] == 'KP481'], x = 'Age', hue =
'MaritalStatus', multiple = 'stack', bins=20)
```

```
plt.show()

# KP-781
sns.histplot(data = df[df['Product'] == 'KP781'], x = 'Age', hue =
'MaritalStatus', multiple = 'stack', bins=20)
plt.show()
```




MaritalStatus
Single
Partnered
Count
Age
0
2
4
6
8
10
12
14
20
25
30
35
40
45


MaritalStatus
Single
Partnered
Count
Age
0
1
2
3
4
5
6
7
25
30
35
40
45

2-Way Contingency Table

```
# 2-Way Contingency Table
genderProduct_table = pd.crosstab(index=df['Gender'],
columns=df['Product'], margins=True)

print("2-Way Contingency Gender-Product Table :")
print()
print(genderProduct_table)
```

```
2-Way Contingency Gender-Product Table :

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

Marginal Probabilities

```
# Marginal Probabilities(using pandas.crosstab)

marginal_probability = pd.crosstab(index=df['Product'],
columns='Count(in %)', normalize=True) * 100

print("Marginal Probability of Each Product :")
print()
print(marginal_probability)
```

```
Marginal Probability of Each Product :

col_0    Count(in %)
Product
KP281      44.444444
KP481      33.333333
KP781      22.222222
```

Probability of a Male customer buying a KP-718 Treadmill

```
# Finding the probability of a male customer buying a KP718 Treadmill


# No.of Male customers who bought the KP781 Treadmills
male_kp781_count = df[(df['Gender'] == 'Male') & (df['Product'] ==
'KP781')].shape[0]
```

```
print("No.of Males who bought KP781 Treadmill :", male_kp781_count)

# Total No.of Male customers who bought ANY Treadmills
total_male_count = df[df['Gender'] == 'Male'].shape[0]
print("Total No.of Males :", total_male_count)

# Probability of a male customer buying a KP781 treadmill
probability_male_kp781 = male_kp781_count / total_male_count
probability_male_kp781 = probability_male_kp781 * 100 # In Percentage
print()

print("Probability(in %) of a Male customer buying a KP781
Treadmill:", probability_male_kp781, "%")
```

```
No.of Males who bought KP781 Treadmill : 33
Total No.of Males : 104

Probability(in %) of a Male customer buying a KP781 Treadmill:
31.73076923076923 %
```

Non-Graphical Analysis

```
# Non-Graphical Analysis
# Iterates over all columns and prints all unique values and lists
them

for col in df.columns:
    print(f"Column: {col}")
    print(df[col].value_counts())
    print("Unique values:", df[col].unique())
    print()

# Unique values and value counts for each column
unique_values_counts = {col: {'unique_values': df[col].unique(),
'value_counts': df[col].value_counts()} for col in df.columns}
print("Unique Values and Value Counts for Each Column:")
print(unique_values_counts)
```

```
Column: Product
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
Unique values: ['KP281' 'KP481' 'KP781']

Column: Age
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
Unique values: [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]

Column: Gender
Male      104
Female     76
Name: Gender, dtype: int64
Unique values: ['Male' 'Female']

Column: Education
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
```

```
Name: Education, dtype: int64
Unique values: [14 15 12 13 16 18 20 21]

Column: MaritalStatus
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
Unique values: ['Single' 'Partnered']

Column: Usage
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
Unique values: [3 2 4 5 6 7]

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

Column: Income
45480    14
52302     9
46617     8
54576     8
53439     8
         ..
65220     1
55713     1
68220     1
30699     1
95508     1
Name: Income, Length: 62, dtype: int64
Unique values: [ 29562  31836  30699  32973  35247  37521  36384
38658  40932  34110
  39795  42069  44343  45480  46617  48891  53439  43206  52302  51165
  50028  54576  68220  55713  60261  67083  56850  59124  61398  57987
  64809  47754  65220  62535  48658  54781  48556  58516  53536  61006
  57271  52291  49801  62251  64741  70966  75946  74701  69721  83416
  88396  90886  92131  77191  52290  85906 103336  99601  89641  95866
 104581  95508]
```

```
Column: 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
Name: Miles, dtype: int64
Unique values: [112  75  66  85  47 141 103  94 113  38 188  56 132
169  64  53 106  95
 212  42 127  74 170  21 120 200 140 100  80 160 180 240 150 300 280
260
 360]

Unique Values and Value Counts for Each Column:
{'Product': {'unique_values': array(['KP281', 'KP481', 'KP781'],
dtype=object), 'value_counts': KP281    80
KP481    60
```

```
KP781    40
Name: Product, dtype: int64}, 'Age': {'unique_values': 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]),
'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}, 'Gender': {'unique_values': array(['Male',
'Female'], dtype=object), 'value_counts': Male      104
Female     76
Name: Gender, dtype: int64}, 'Education': {'unique_values': array([14,
15, 12, 13, 16, 18, 20, 21]), 'value_counts': 16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64}, 'MaritalStatus': {'unique_values':
```

```
array(['Single', 'Partnered'], dtype=object), 'value_counts':
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64}, 'Usage': {'unique_values':
array([3, 2, 4, 5, 6, 7]), 'value_counts': 3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64}, 'Fitness': {'unique_values': array([4, 3,
2, 1, 5]), 'value_counts': 3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64}, 'Income': {'unique_values':
array([ 29562,  31836,  30699,  32973,  35247,  37521,  36384,  38658,
        40932,  34110,  39795,  42069,  44343,  45480,  46617,  48891,
        53439,  43206,  52302,  51165,  50028,  54576,  68220,  55713,
        60261,  67083,  56850,  59124,  61398,  57987,  64809,  47754,
        65220,  62535,  48658,  54781,  48556,  58516,  53536,  61006,
        57271,  52291,  49801,  62251,  64741,  70966,  75946,  74701,
        69721,  83416,  88396,  90886,  92131,  77191,  52290,  85906,
       103336,  99601,  89641,  95866, 104581,  95508]),
'value_counts': 45480    14
52302     9
46617     8
54576     8
53439     8
         ..
65220     1
55713     1
68220     1
30699     1
95508     1
Name: Income, Length: 62, dtype: int64}, 'Miles': {'unique_values':
array([112,  75,  66,  85,  47, 141, 103,  94, 113,  38, 188,  56,
132,
       169,  64,  53, 106,  95, 212,  42, 127,  74, 170,  21, 120,
200,
       140, 100,  80, 160, 180, 240, 150, 300, 280, 260, 360]),
'value_counts': 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
Name: Miles, dtype: int64}}
```

General Comments/Observations

Range of Attributes:

1. Has different attributes like age, gender, income, education, usage(per week), marital status, etc.
2. Age ranges from 18 to 48 yrs
3. Education ranges from 12 to 21 yrs
4. Usage(per week) ranges from 2 to 6
5. Income(in USD) ranges from 29,562 to 104,581
6. Fitness(level on a scale of 5) ranges from 2 to 5
7. No.of miles(per week) ranges from 47 to 200

Distribution of Variables and Relationships:

1. There's a wide distribution across age, education and income ranges
2. There's direct correlation between income and no.of miles run per week.Greater the income, the more no.of miles were run in a week. Thus, it can be said that the wealthy focus a lot on physical health
3. Gender distribution is relatively balanced
4. Different people prefer different products

Recommendations

- KP218

1. Target both genders equally.
2. Prioritize customers who use the product 3 days/week.
3. Focus on Partnered customers.
4. Target customers with 16 years of education.

- KP418

1. Target both genders equally.
2. Prioritize customers with 14-16 years of education.
3. Focus on Partnered customers.
4. Target customers with 16 years of education.

- KP718

1. Focus on male customers.
2. Target customers with 18 years of education.
3. Prioritize customers who use the product 4 days/week.
4. Focus on Partnered customers.