


```
!wget "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749" -O aerofit_treadmil
!gdown "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749" -O aerofit_treadmi
!pip install numpy
!pip install pandas_summary
```

 [Show hidden output](#)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random
import seaborn as sns
from pandas_summary import DataFrameSummary as dfs
from scipy.stats.mstats import winsorize
```

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

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

```
df.info()
```

 [Show hidden output](#)

```
df.head(10)
```



	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
5	KP281	20	Female	14	Partnered	3	3	32973	66
6	KP281	21	Female	14	Partnered	3	3	35247	75
7	KP281	21	Male	13	Single	3	3	32973	85
8	KP281	21	Male	15	Single	5	4	35247	141
9	KP281	21	Female	15	Partnered	2	3	37521	85

```
df.tail(10)
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
170	KP781	31	Male	16	Partnered	6	5	89641	260
171	KP781	33	Female	18	Partnered	4	5	95866	200
172	KP781	34	Male	16	Single	5	5	92131	150
173	KP781	35	Male	16	Partnered	4	5	92131	360
174	KP781	38	Male	18	Partnered	5	5	104581	150
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

```
df.shape
```

(180, 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.isnull().any()
```

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

dtvne: bool

```
df.nunique()
```



```

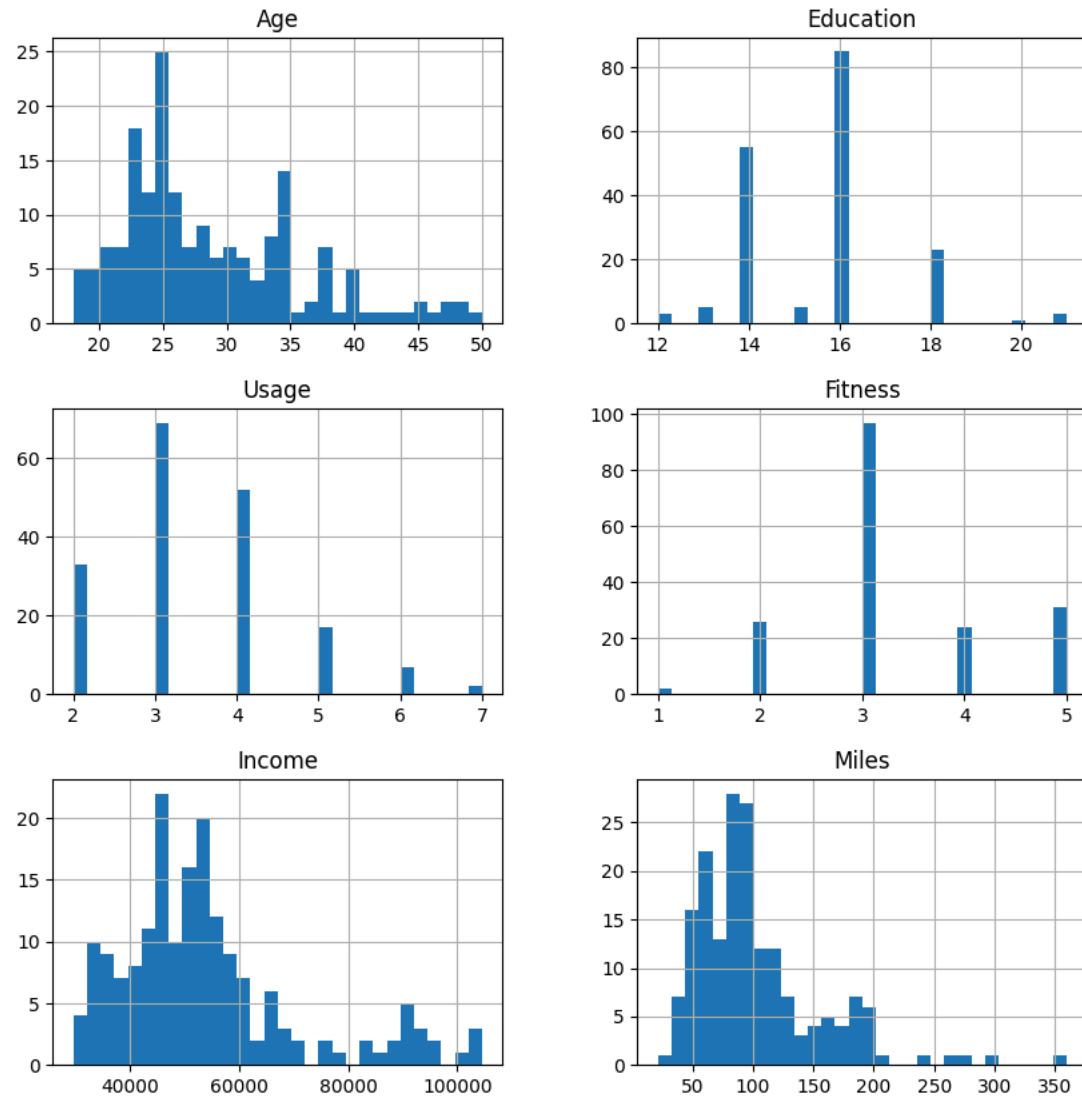
0
Product      3
Age          32
Gender       2
Education    8
MaritalStatus 2
Usage        6
Fitness      5
Income       62
Miles        37

```

dtype: int64

```
df.drop_duplicates(inplace=True)
```

```
df.hist(figsize=(10,10), bins=30)
plt.show()
```



```
# Detect Outliers (using boxplot, "describe" method by checking the difference between mean and median)
```

```

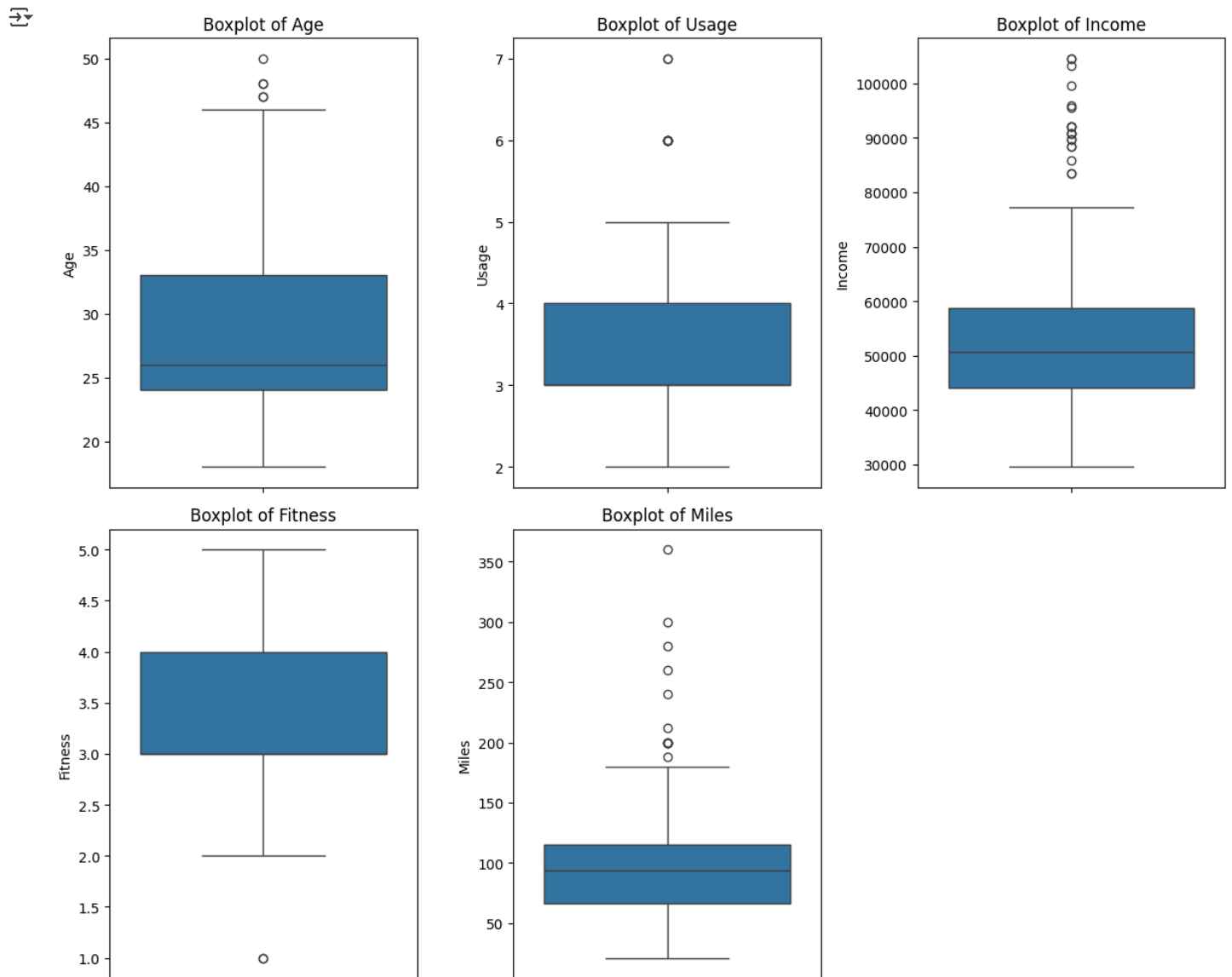
#BOXPLOT OUTLIERS
numeric_columns = ['Age', 'Usage', 'Income', 'Fitness', 'Miles']
plt.figure(figsize=(12, 10))

```

```

for i, col in enumerate(numeric_columns):
    plt.subplot(2, 3, i + 1)
    sns.boxplot(y=df[col])
    plt.title(f'Boxplot of {col}')
plt.tight_layout()
plt.show()

```



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

```
df[numeric_columns].median()
```

	0
Age	26.0
Usage	3.0
Income	50596.5
Fitness	3.0
Miles	94.0

dtype: float64

```
df['Age'] = winsorize(df['Age'], limits=[0.05, 0.05])
df['Usage'] = winsorize(df['Usage'], limits=[0.05, 0.05])
df['Income'] = winsorize(df['Income'], limits=[0.05, 0.05])
df['Miles'] = winsorize(df['Miles'], limits=[0.05, 0.05])

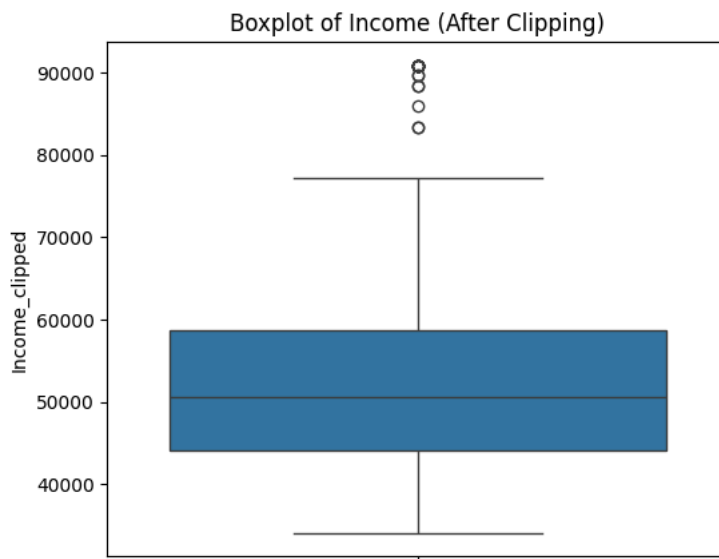
df['Income_log'] = np.log(df['Income'] + 1)
df['Miles_sqrt'] = np.sqrt(df['Miles'])

# Define percentile limits as per income
lower_bound = df['Income'].quantile(0.05) # 5th percentile
upper_bound = df['Income'].quantile(0.95) # 95th percentile

# Apply np.clip() to cap extreme values
df['Income_clipped'] = np.clip(df['Income'], lower_bound, upper_bound)

# Boxplot after clipping
plt.figure(figsize=(6, 5))
sns.boxplot(y=df['Income_clipped'])
plt.title("Boxplot of Income (After Clipping)")
plt.show()
```

```
/usr/local/lib/python3.11/dist-packages/numpy/lib/_function_base_impl.py:4968: UserWarning: Warning: 'partition' will ignore the 'mask'
arr.partition(
/usr/local/lib/python3.11/dist-packages/numpy/lib/_function_base_impl.py:4968: UserWarning: Warning: 'partition' will ignore the 'mask'
arr.partition(
```



```
# Define percentile limits as per miles
lower_bound = df['Miles'].quantile(0.05) # 5th percentile
upper_bound = df['Miles'].quantile(0.95) # 95th percentile

# Apply np.clip() to cap extreme values
df['Miles_clipped'] = np.clip(df['Miles'], lower_bound, upper_bound)

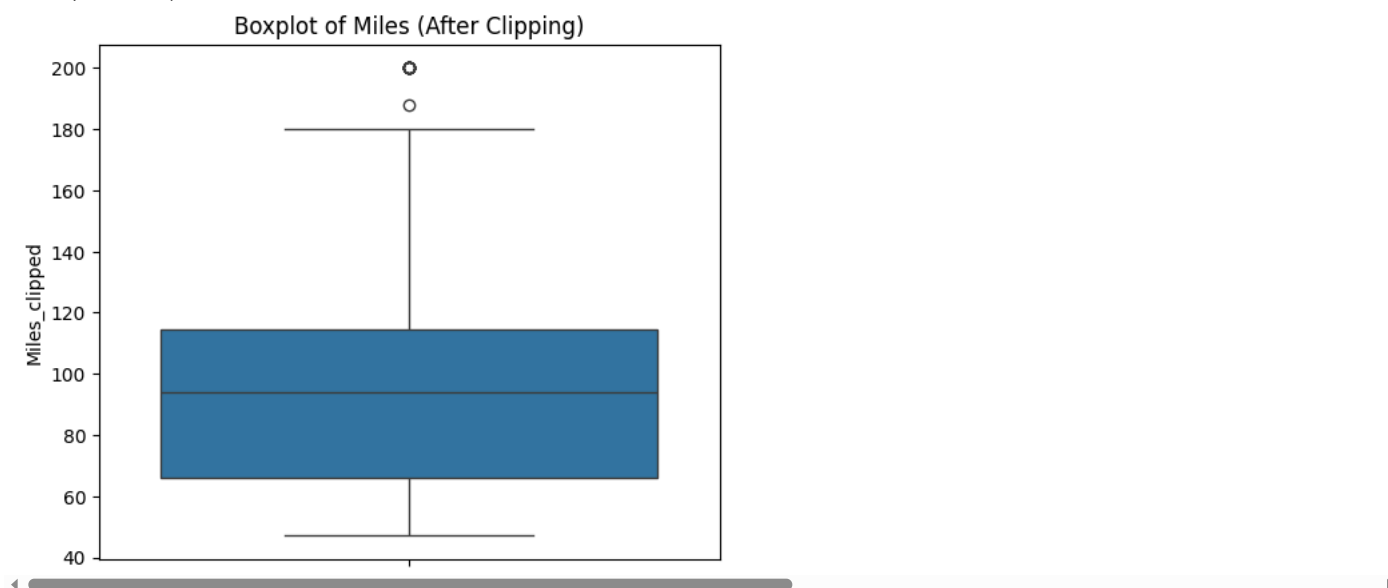
# Boxplot after clipping
plt.figure(figsize=(6, 5))
sns.boxplot(y=df['Miles_clipped'])
plt.title("Boxplot of Miles (After Clipping)")
```

```
plt.show()
```

```

/usr/local/lib/python3.11/dist-packages/numpy/lib/_function_base_impl.py:4968: UserWarning: Warning: 'partition' will ignore the 'mask'
arr.partition(
/usr/local/lib/python3.11/dist-packages/numpy/lib/_function_base_impl.py:4968: UserWarning: Warning: 'partition' will ignore the 'mask'
arr.partition(

```



Check if features like marital status, age have any effect on the product purchased (using countplot, histplots, boxplots etc)

```

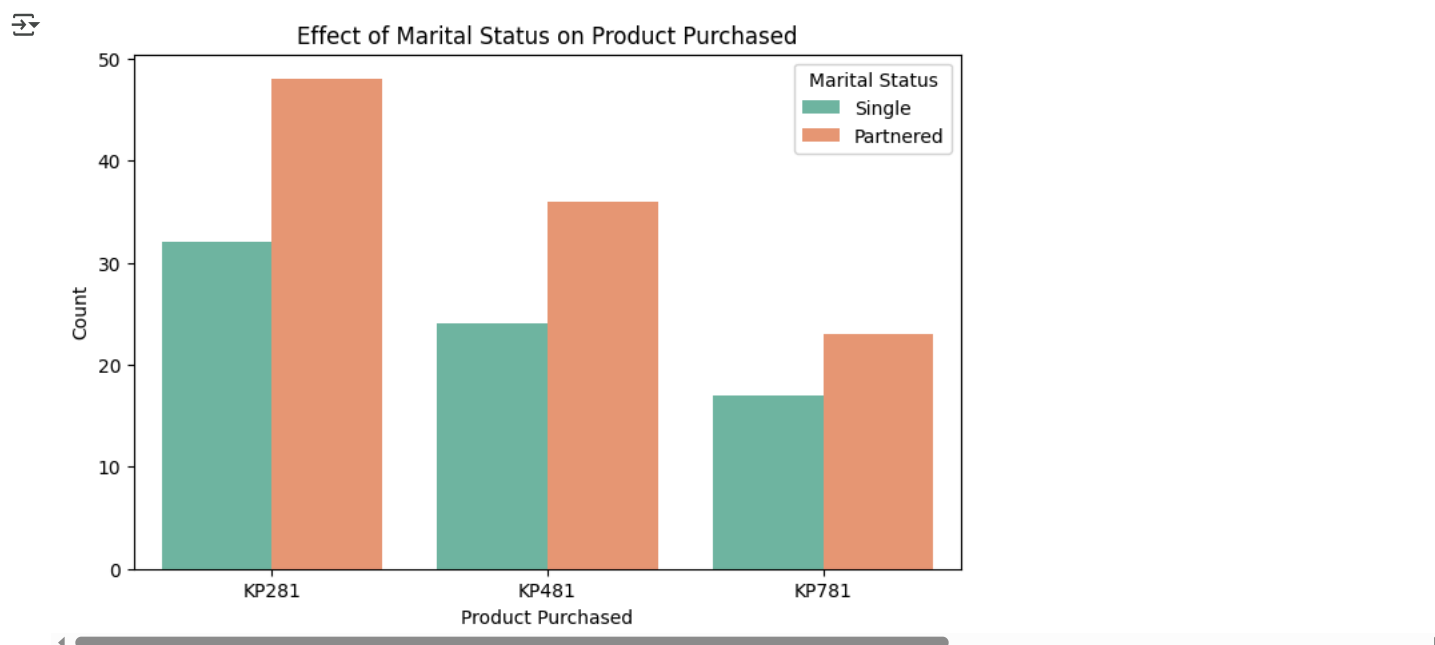
plt.figure(figsize=(8, 5))
sns.countplot(x='Product', hue='MaritalStatus', data=df, palette='Set2')

```

```

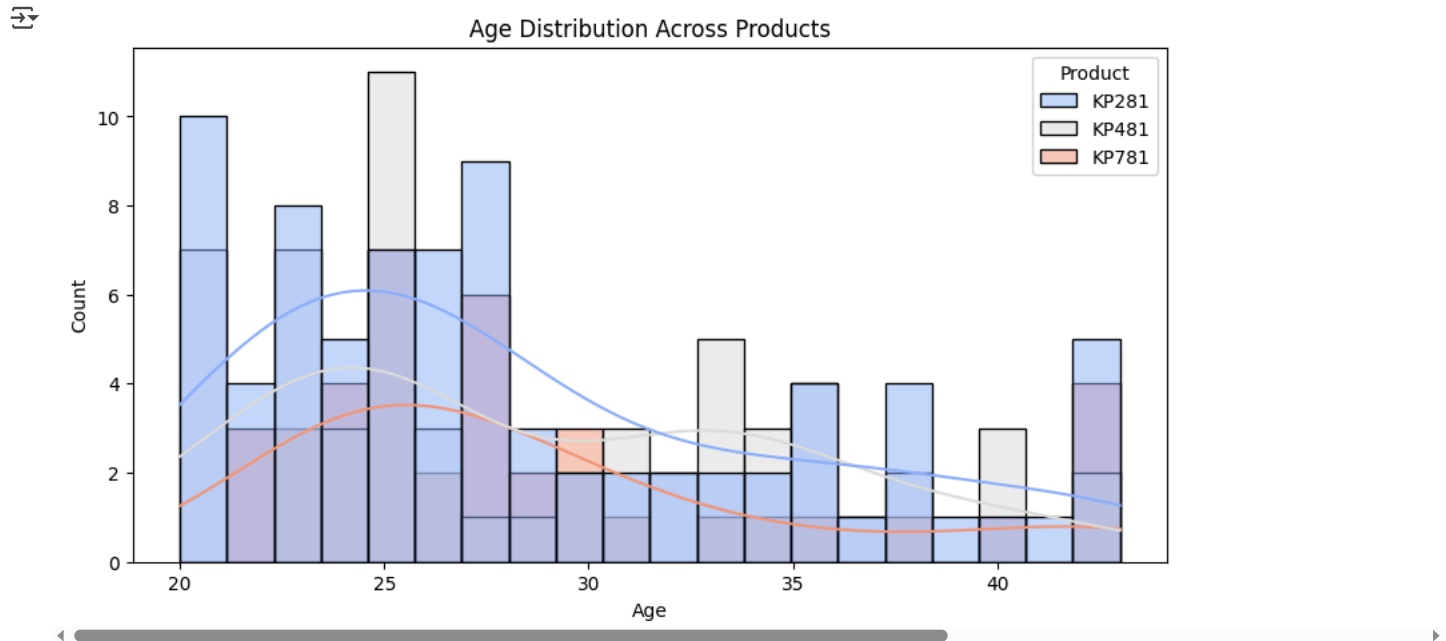
plt.title('Effect of Marital Status on Product Purchased')
plt.xlabel('Product Purchased')
plt.ylabel('Count')
plt.legend(title='Marital Status', labels=['Single', 'Partnered'])
plt.show()

```



```
plt.figure(figsize=(10, 5))
sns.histplot(data=df, x='Age', hue='Product', bins=20, kde=True, palette='coolwarm')

plt.title('Age Distribution Across Products')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



```
# Representing the marginal probability like - what percent of customers have purchased KP281, KP481, or KP781 in a table (can use pandas.cr
```

```
# Calculate the percentage of customers purchasing each product
marginal_prob = pd.crosstab(df['Product'], columns='Count', normalize=True) * 100
```

```
# Rename columns for clarity
marginal_prob.columns = ['Percentage of Customers (%)']
```

```
# Display the table
print(marginal_prob)
```

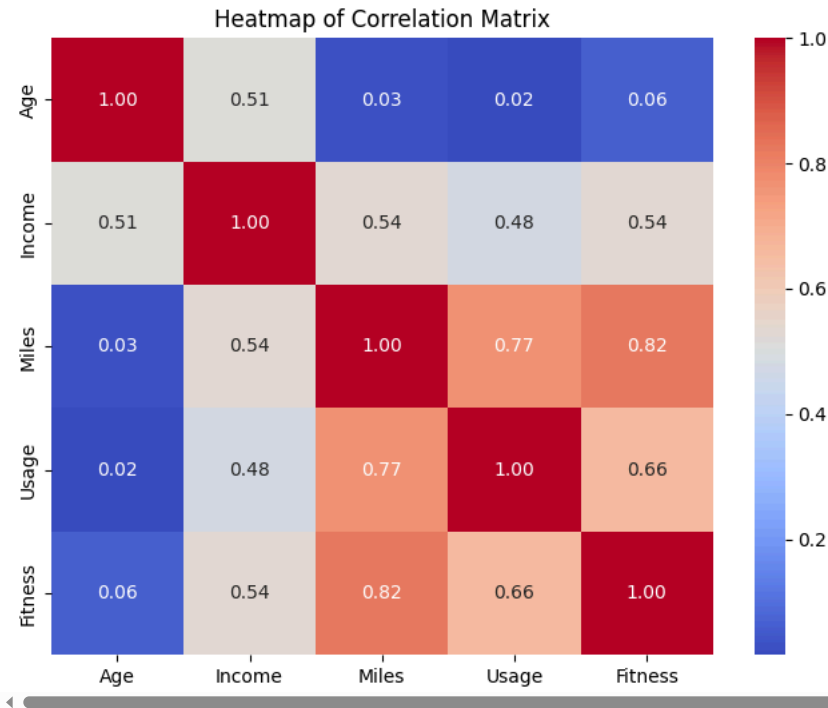
```
Product
KP281      44.444444
KP481      33.333333
KP781      22.222222
```

```
df1 = df[['Age', 'Income', 'Miles', 'Usage', 'Fitness']]
```

```
# Print the title separately
print("CORRELATION MATRIX")
```

```
# Create the heatmap
plt.figure(figsize=(8,6)) # Adjust figure size if needed
sns.heatmap(df1.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Heatmap of Correlation Matrix")
plt.show()
```

↗ CORRELATION MATRIX



```
# Create a crosstab to count occurrences
gender_product_counts = pd.crosstab(df['Gender'], df['Product'], normalize='index') * 100
```

```
# Extract the probability of a male buying KP781
prob_male_KP781 = gender_product_counts.loc['Male', 'KP781']
```

```
print(f"Probability of a male customer buying KP781: {prob_male_KP781:.2f}%")
```

↗ Probability of a male customer buying KP781: 31.73%

#Customer Profiling - Categorization of users

```
# Define function to categorize customers
def categorize_customer(row):
    if row['Age'] < 30 and row['Income'] < 50000 and row['Product'] == 'KP281':
        return 'Young Budget Buyers'
    elif 30 <= row['Age'] <= 45 and 50000 <= row['Income'] <= 80000 and row['Product'] == 'KP481':
        return 'Mid-Level Users'
    elif row['Age'] > 45 and row['Income'] > 80000 and row['Product'] == 'KP781':
        return 'High-End Buyers'
    elif row['Fitness'] >= 4 and row['Miles'] > 15:
        return 'Fitness Enthusiasts'
    elif row['Fitness'] <= 2 and row['Miles'] < 5:
        return 'Casual Users'
    else:
        return 'Other'
```

```
# Apply function to categorize customers
df['Customer_Segment'] = df.apply(categorize_customer, axis=1)
```

```
# Display count of each segment
segment_counts = df['Customer_Segment'].value_counts()
print(segment_counts)
```


↗ Customer_Segment

Other	69
Fitness Enthusiasts	47
Young Budget Buyers	40
Mid-Level Users	24
Name: count, dtype: int64	

```
plt.figure(figsize=(5, 5))
sns.countplot(y=df['Customer_Segment'], palette='coolwarm', order=df['Customer_Segment'].value_counts().index)
```

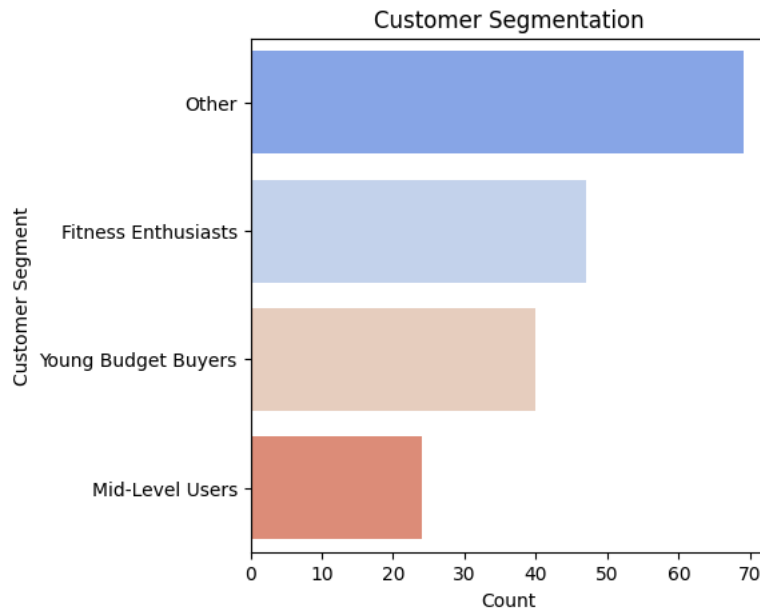


```
plt.title("Customer Segmentation")
plt.xlabel("Count")
plt.ylabel("Customer Segment")
plt.show()
```

 <ipython-input-32-83c18e14a211>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `v` variable to `hue` and set `legend`

```
sns.countplot(y=df['Customer_Segment'], palette='coolwarm', order=df['Customer_Segment'].value_counts().index)
```



Start coding or [generate](#) with AI.

INSIGHTS FOR AEROFIT_TREADMILL

#1. Customer Demographics

#####Age Distribution: Are younger or older customers buying AeroFit treadmills more?

#####Gender Split: Do males or females buy the treadmill more?

#####Income Levels: Are higher-income groups the primary buyers?

#2. Product Preferences

#####Treadmill Types: Which models (e.g., standard, incline, smart) are more popular?

#####Usage Patterns: Are customers using treadmills for weight loss, endurance, or casual walking?

#3. Purchase Behavior

#####Price Sensitivity: Do most customers buy budget, mid-range, or high-end models?

#####Seasonal Trends: Are there peak sales in specific months?

#####Sales Channels: Do customers prefer online purchases or in-store?