AeroFit Treadmill Customer Analysis

1. Problem Statement and Defining Business Objectives

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

2. Data Loading and Initial Exploration

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

# Load the dataset
df = pd.read_csv('aerofit_treadmill.csv')

# View the first few rows
print('--- First 5 Rows ---')
print(df.head())

```

```
--- First 5 Rows ---
  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
```

```
 # Get information about the dataset
 print('\n--- Data Info ---')
 df.info()
```

```

--- Data 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
```

```
 # Get statistical summary
 print('\n--- Statistical Summary ---')
 print(df.describe())
```

```

--- Statistical Summary ---
              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
```

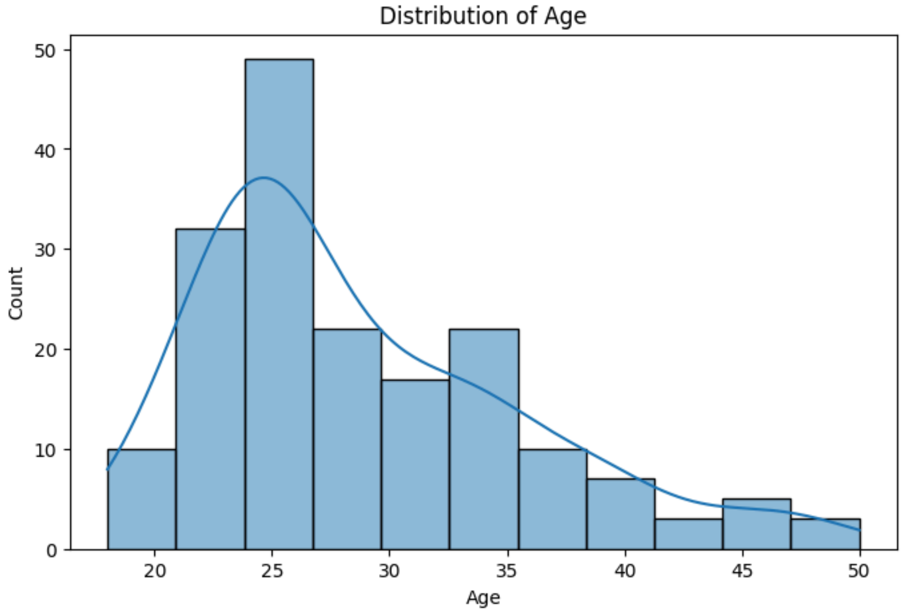
```
# Check for missing values
print('\n--- Missing Values ---')
print(df.isnull().sum())
```

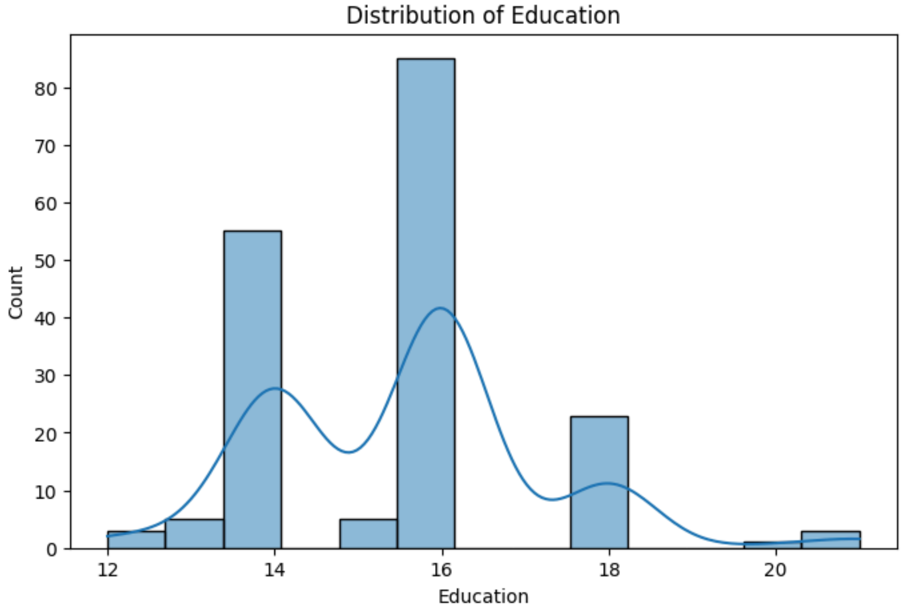
```
--- Missing Values ---
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

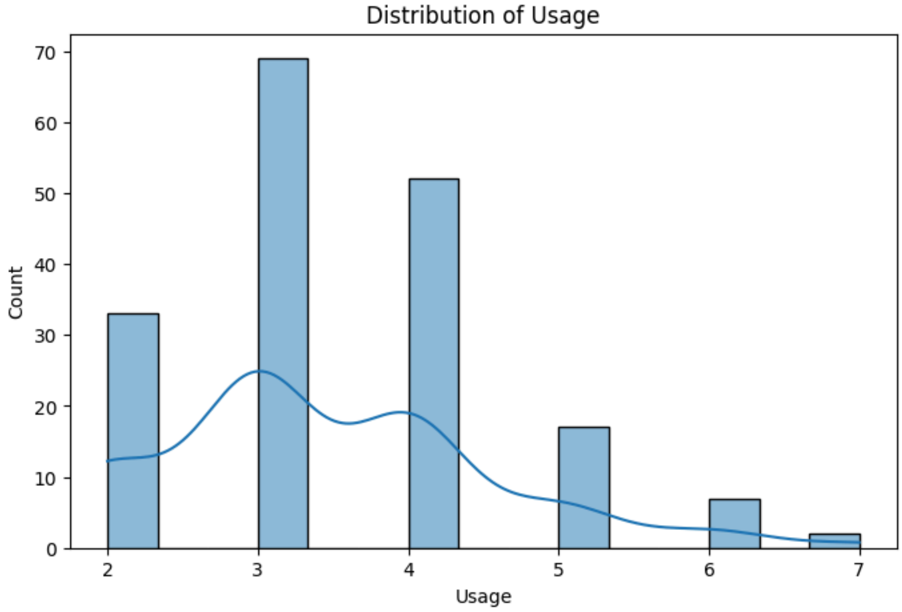
3. Visual Analysis (EDA)

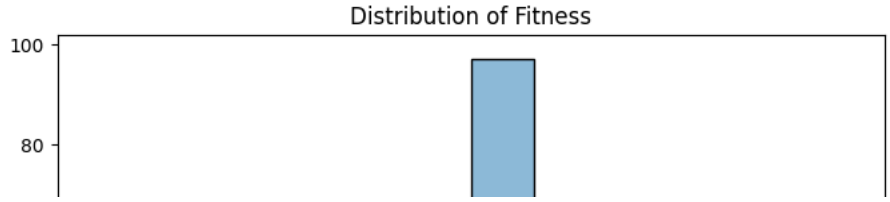
Univariate Analysis

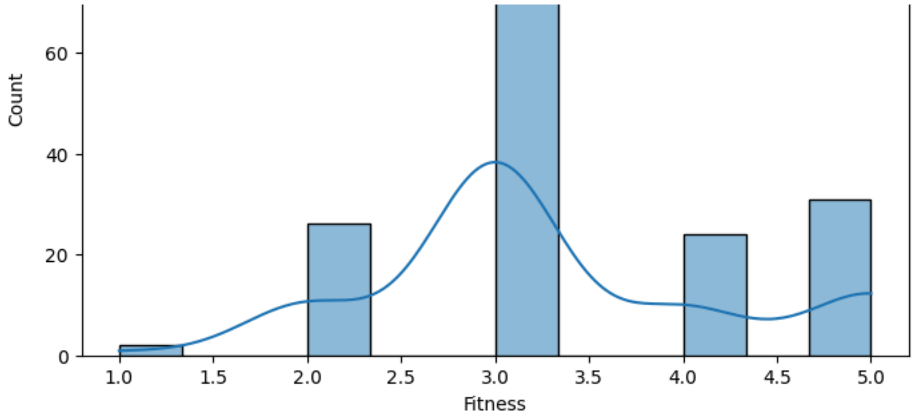
```
# Histograms for numerical features
numerical_features = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
for feature in numerical_features:
    plt.figure(figsize=(8, 5))
    sns.histplot(df[feature], kde=True)
    plt.title(f'Distribution of {feature}')
    plt.show()
```

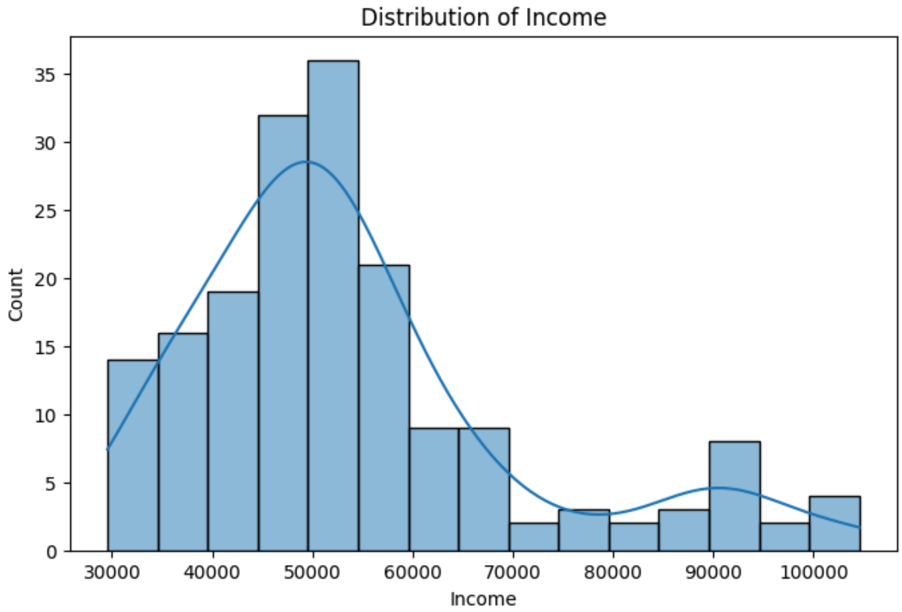

Distribution of Age
Count
0
10
20
30
40
50
20
25
30
35
40
45
50
Age

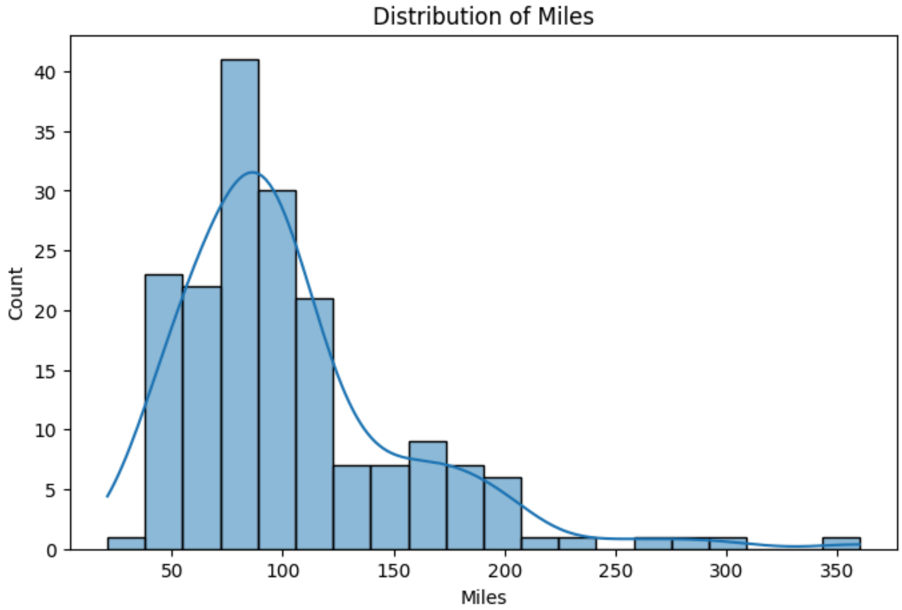

Distribution of Education
Count
0
10
20
30
40
50
60
70
80
12
14
16
18
20
Education


Distribution of Usage
Count
0
10
20
30
40
50
60
70
2
3
4
5
6
7
Usage

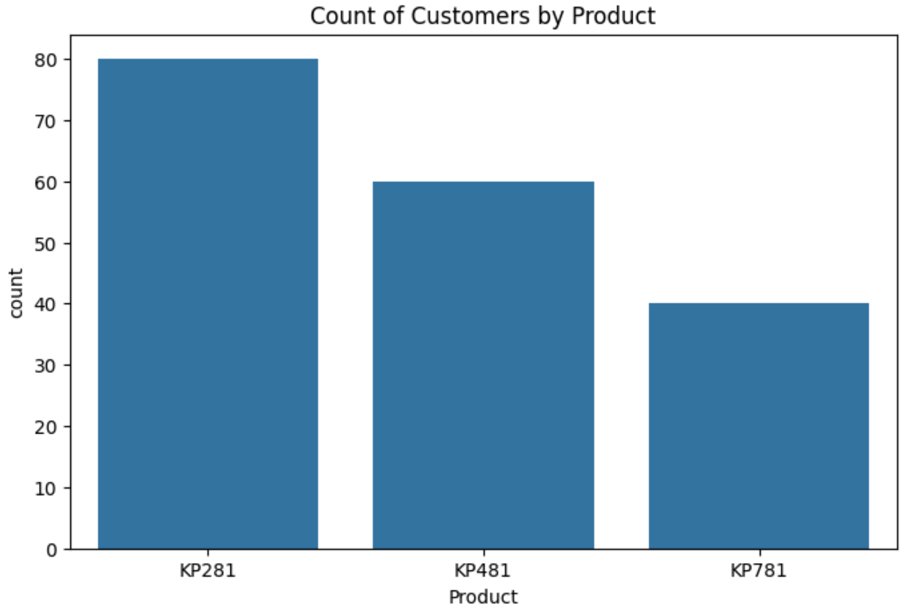

Distribution of Fitness
100
80

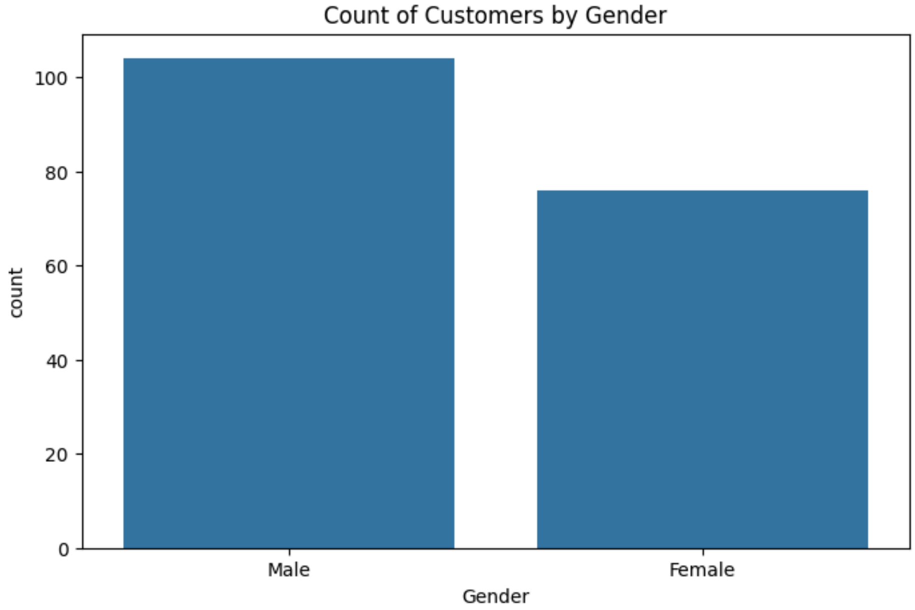

Count
0
20
40
60
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Fitness

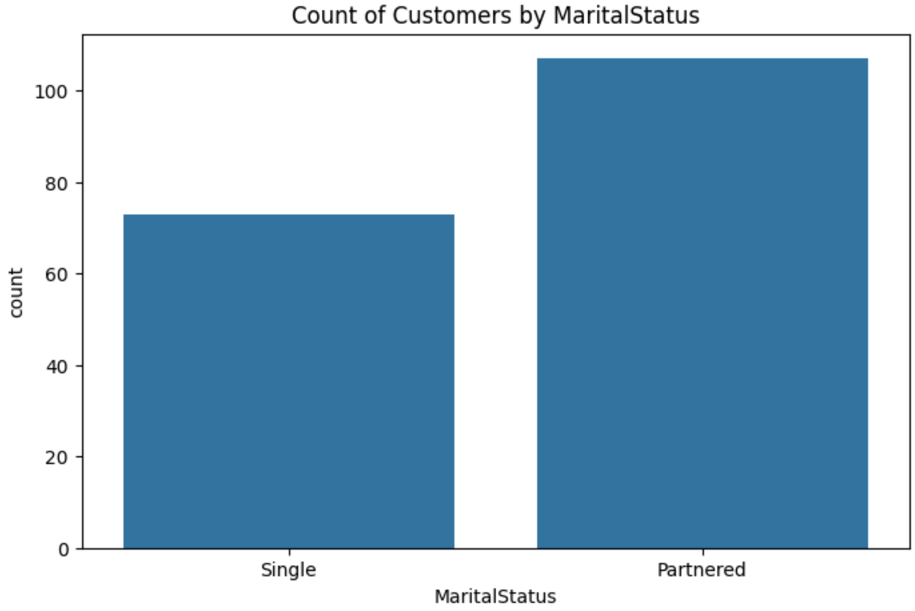

Distribution of Income
Count
0
5
10
15
20
25
30
35
30000
40000
50000
60000
70000
80000
90000
100000
Income


Distribution of Miles
Count
0
5
10
15
20
25
30
35
40
50
100
150
200
250
300
350
Miles

```
# Countplots for categorical features
categorical_features = ['Product', 'Gender', 'MaritalStatus']
for feature in categorical_features:
    plt.figure(figsize=(8, 5))
    sns.countplot(x=feature, data=df)
    plt.title(f'Count of Customers by {feature}')
    plt.show()
```



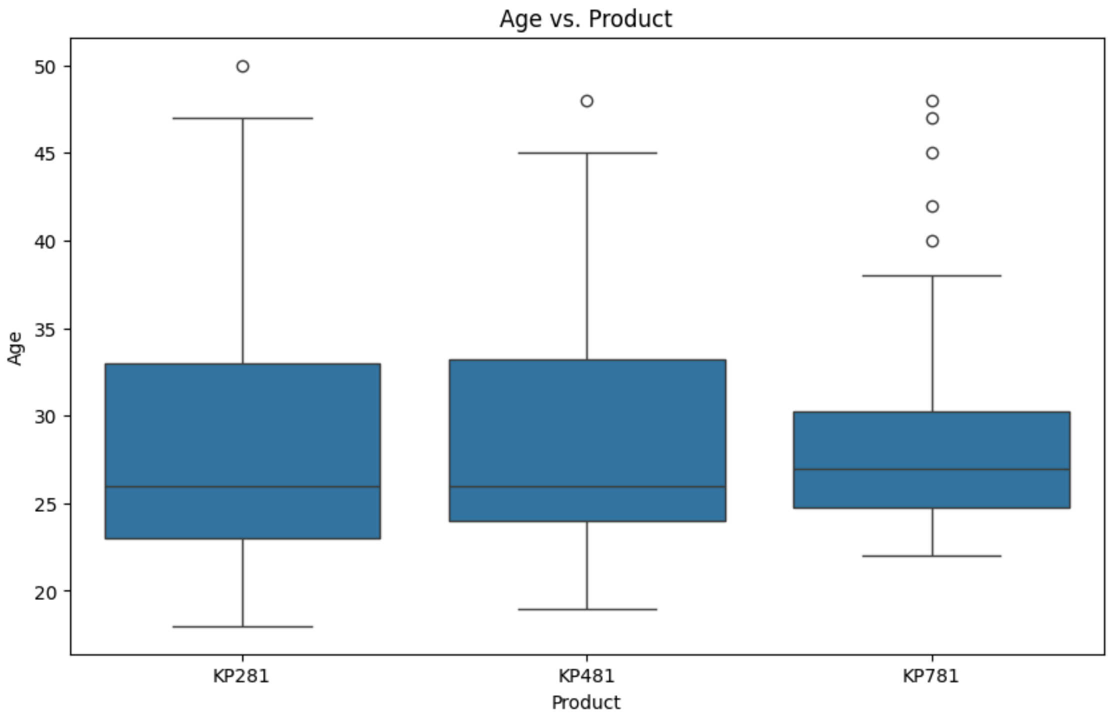


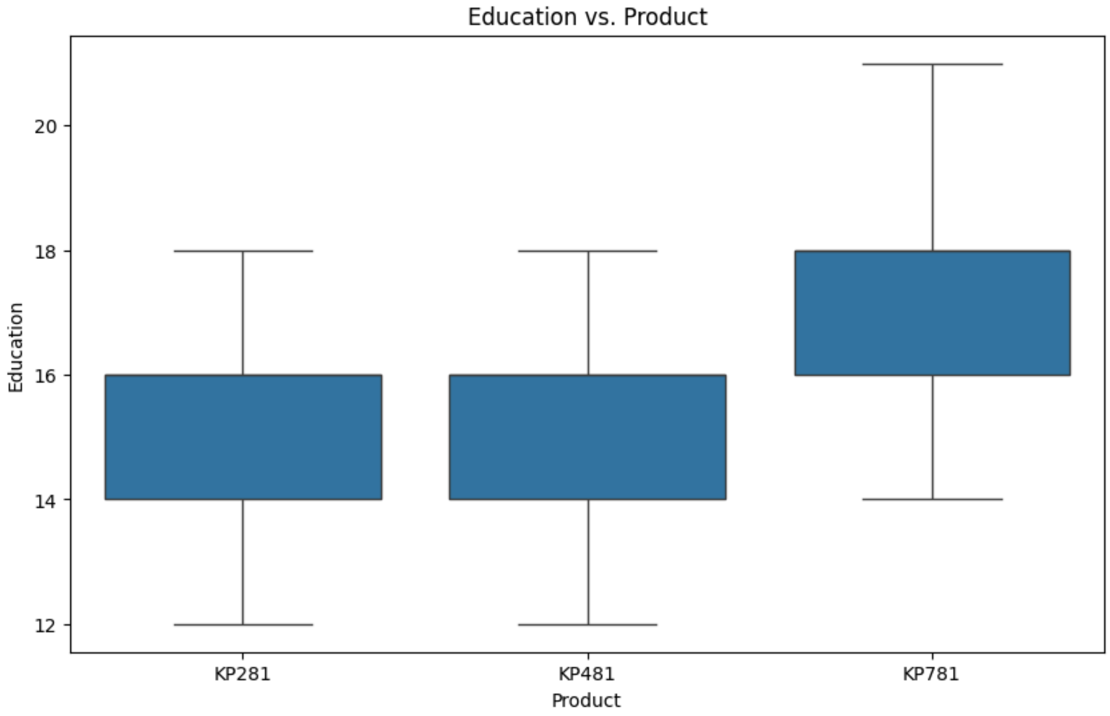


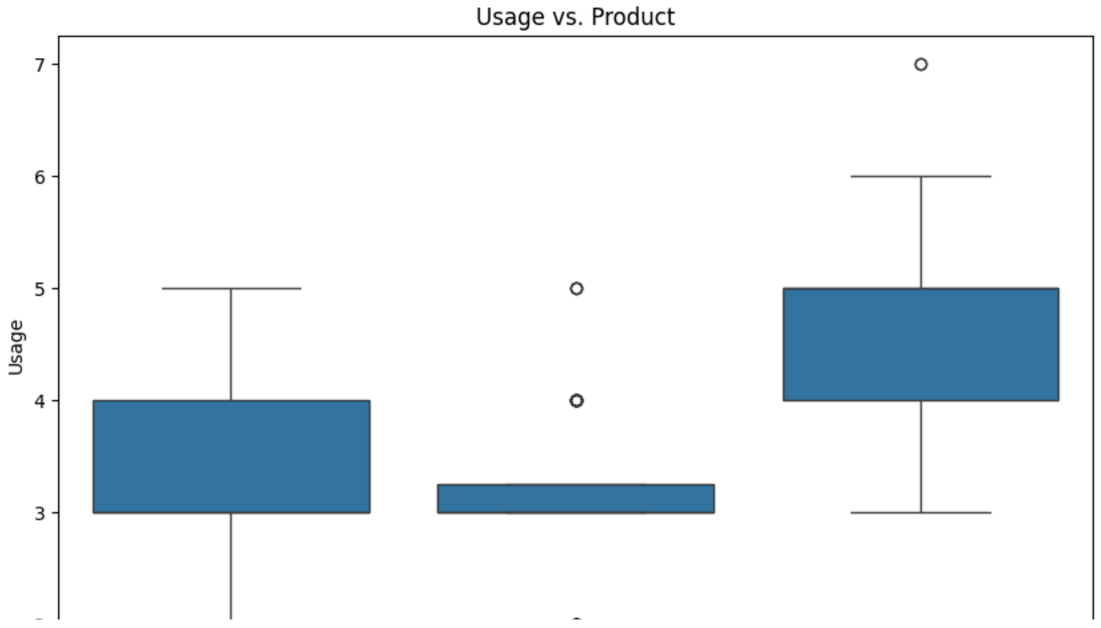
Bivariate Analysis

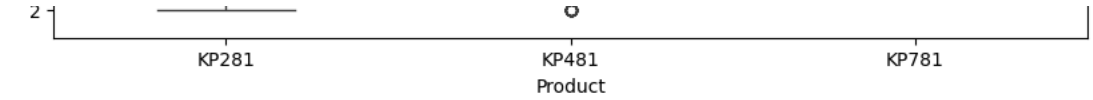
```
# Boxplots for numerical features vs. Product
for feature in numerical_features:
    plt.figure(figsize=(10, 6))
```

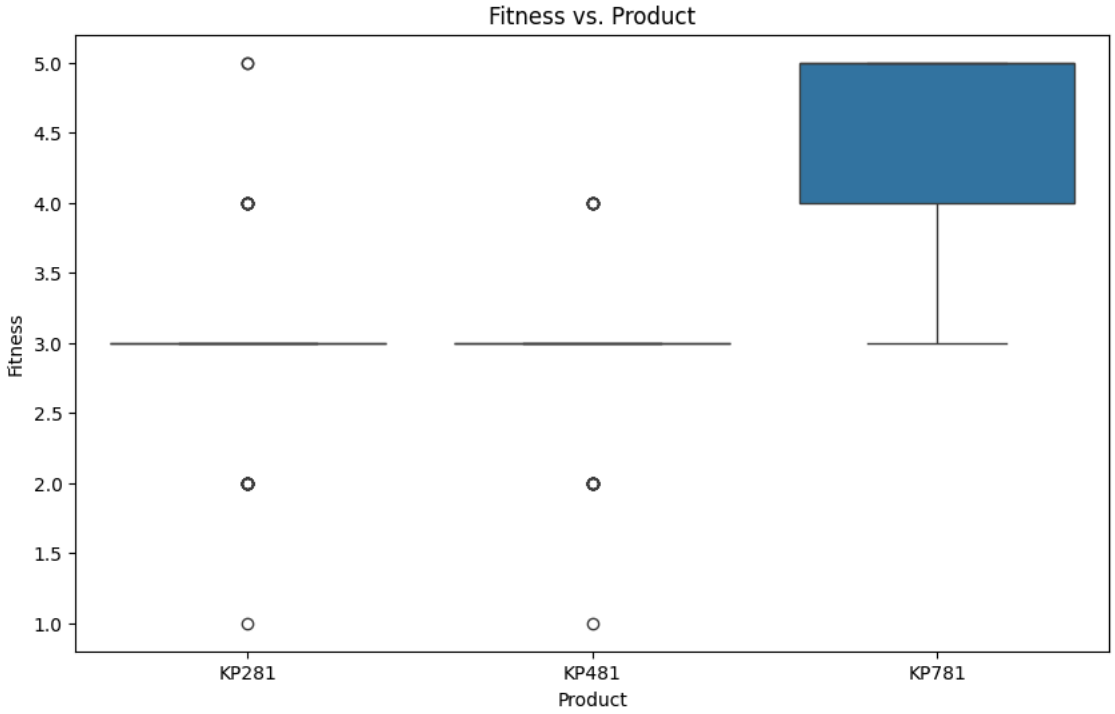
```
sns.boxplot(x='Product', y=feature, data=df)
plt.title(f'{feature} vs. Product')
plt.show()
```

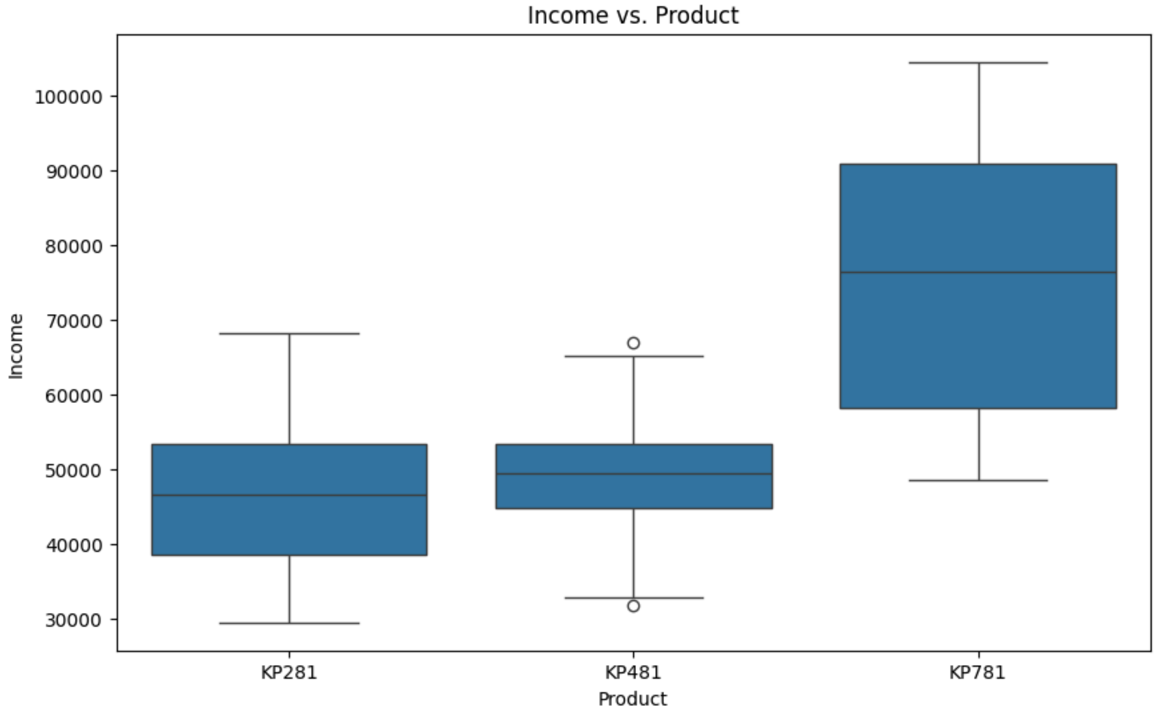

Age vs. Product
Age
50
45
40
35
30
25
20
KP281
KP481
KP781
Product

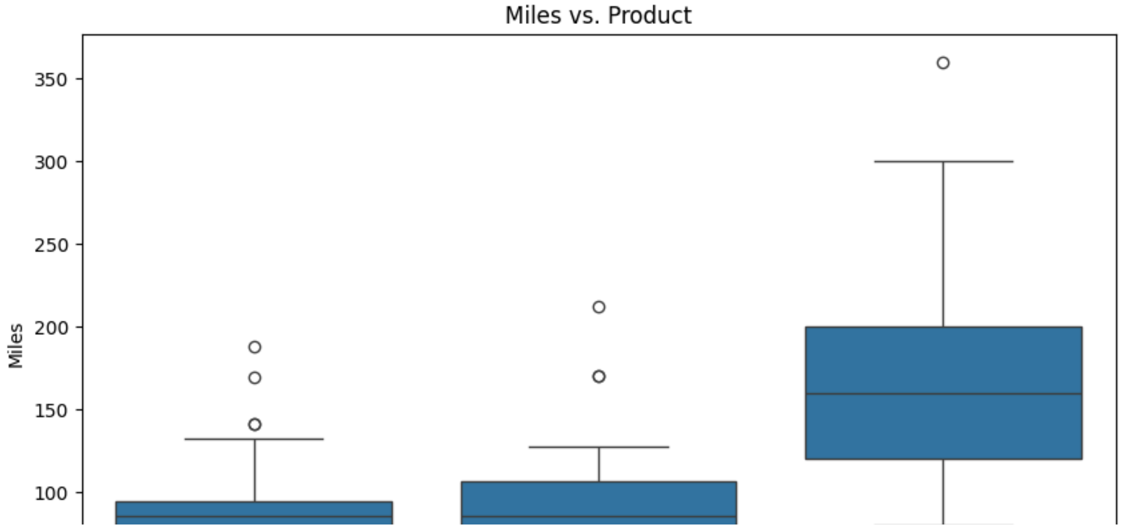

Education vs. Product
Education
20
18
16
14
12
KP281
KP481
KP781
Product

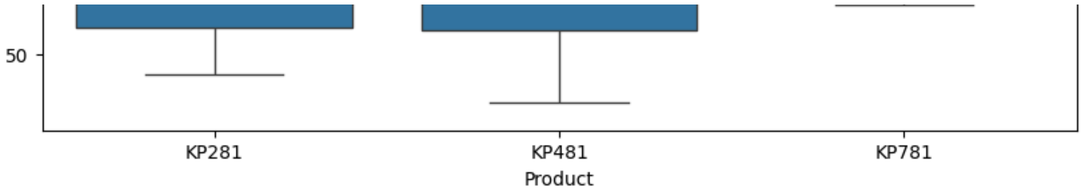

Usage vs. Product
Usage
7
6
5
4
3


2
KP281
KP481
KP781
Product

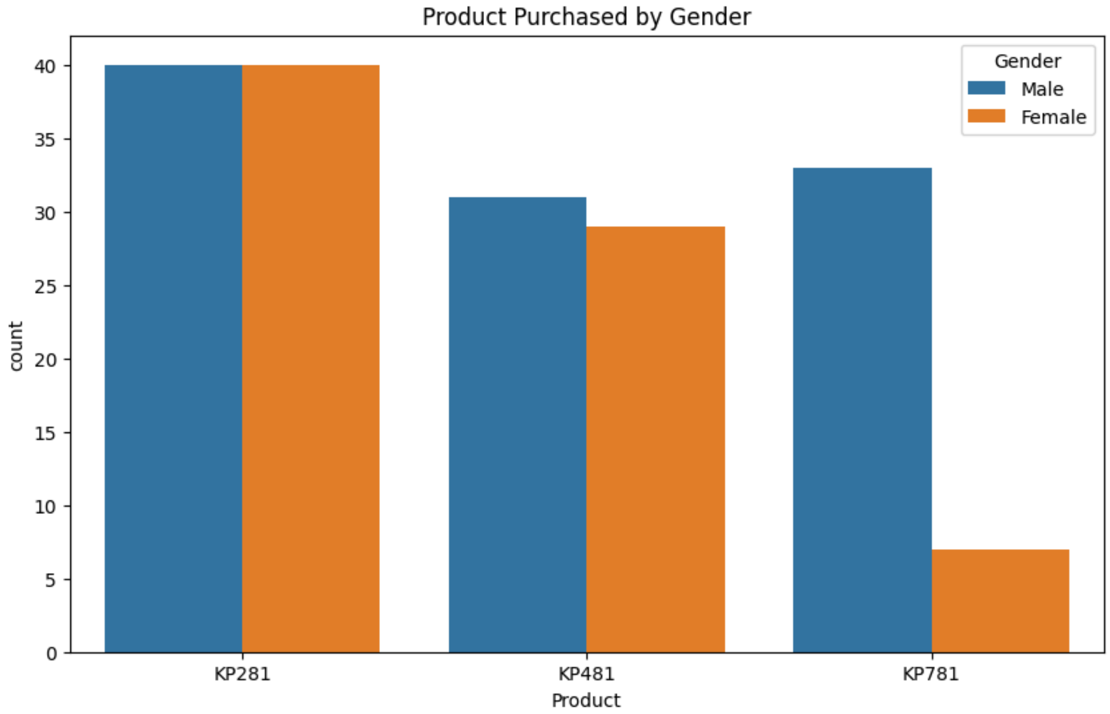

Fitness vs. Product
Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
KP281
KP481
KP781
Product

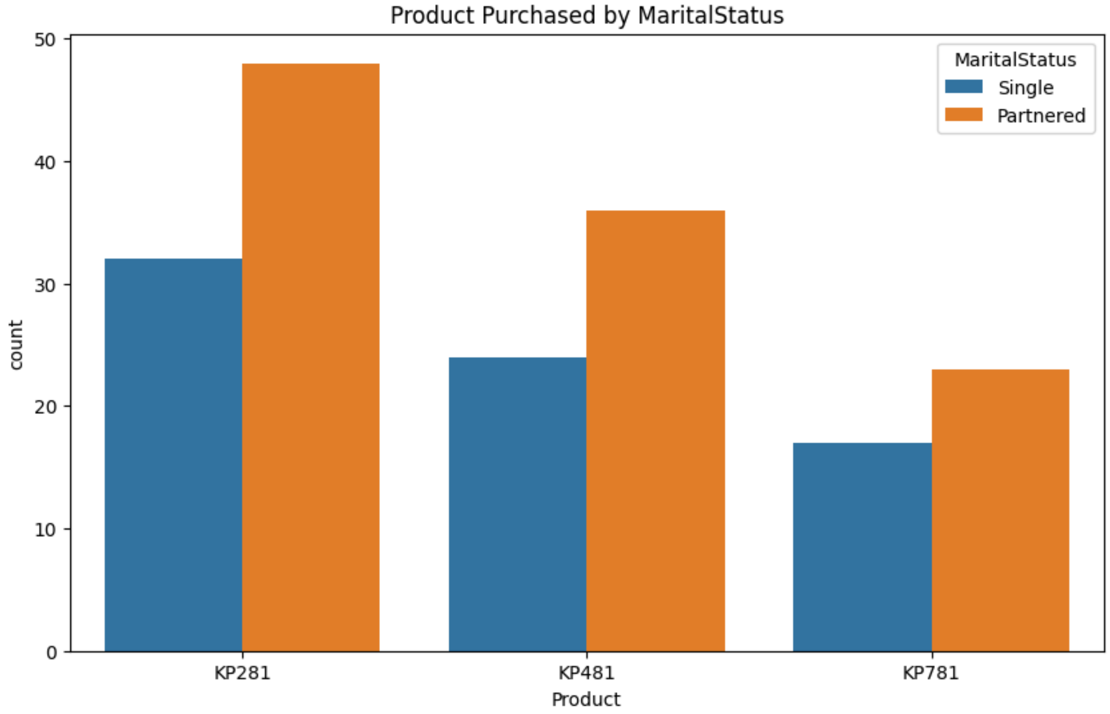

Income vs. Product
Income
100000
90000
80000
70000
60000
50000
40000
30000
KP281
KP481
KP781
Product


Miles vs. Product
Miles
350
300
250
200
150
100



```
# Countplots for categorical features vs. Product
for feature in ['Gender', 'MaritalStatus']:
    plt.figure(figsize=(10, 6))
    sns.countplot(x='Product', hue=feature, data=df)
    plt.title(f'Product Purchased by {feature}')
    plt.legend(title=feature)
    plt.show()
```





Correlation Analysis

```
# Correlation Heatmap
plt.figure(figsize=(12, 8))
correlation = df.corr(numeric_only=True)
```