Aerofit Treadmill Buyer Profile and Data Analysis Report

Khola Shams 10/14/24 Business Analysis / EDA

Table of Contents

1 Introduction

This report presents a thorough analysis of customer demographics, purchasing behavior, and product performance related to Aerofit treadmills. The primary objective is to derive actionable insights and recommendations that can guide marketing strategies, product development, and customer engagement initiatives. By investigating the characteristics of the target audience for each type of treadmill, Aerofit aims to enhance its recommendations to new customers.

2 Project Details

The market research team at Aerofit seeks to identify the characteristics of the target audience for each type of treadmill offered by the company. The insights gathered will inform marketing strategies and product development.

2.1 Product Portfolio

- **KP281**: Entry-level treadmill, priced at **$1,500**.
- **KP481**: Mid-level treadmill designed for runners, priced at **$1,750**.
- **KP781**: Advanced treadmill with premium features, priced at **$2,500**.

3 Data Description

The dataset, **aerofit_treadmill_data.csv**, contains information on individuals who purchased a treadmill from Aerofit stores over the past three months. The features included are:

- **Product**: Type of product purchased (KP281, KP481, or KP781).
- **Age**: Customer age in years.
- **Gender**: Customer gender (male/female).
- **Education**: Education level in years.
- **MaritalStatus**: Customer marital status (single or partnered).
- **Usage**: Average number of times the customer plans to use the treadmill each week.
- **Fitness**: Self-rated fitness level on a scale of 1-5.
- **Income**: Annual income in US dollars.
- **Miles**: Average number of miles the customer expects to walk/run each week.

4 Data Exploration and Processing

4.1 Data Import and Overview

- The data is imported using pandas for analysis.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from warnings import filterwarnings
filterwarnings("ignore")
```

```
df = pd.read_csv("/content/aerofit_treadmill_data.csv")
```

```
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

- The shape of the DataFrame is checked to understand the number of rows and columns.

```
df.shape
```

```
(180, 9)
```

```
df.columns
```

```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles'],
      dtype='object')
```

- Data types of each column are verified to ensure proper analysis.

```
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
```

- A check for missing values is performed to assess data quality.

```
1 df.isnull().sum()
```

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

dtype: int64

- The dataset is scanned for duplicate entries.

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

0

4.2 Statistical Summary

A statistical summary is generated for both categorical and numerical features. Key observations include:

4.2.1 Categorical Features:

Distribution of gender, marital status, and product types.

```
1 df.describe(include = "object")
```

	Product	Gender	MaritalStatus
count	180	180	180
unique	3	2	2
top	KP281	Male	Partnered
freq	80	104	107

4.2.2 Numerical Features:

Summary statistics including mean, median, and standard deviation for age, income, usage, fitness, and miles.

```
1 df.describe()
```

	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

4.3 Non-Graphical Analysis

4.3.1 Value Counts:

Counts of unique values for all categorical features are documented.

```
1 df["Product"].value_counts()
```

Product	count
KP281	80
KP481	60
KP781	40

dtype: int64

```
1 df["Gender"].value_counts()
```

Gender	count
Male	104
Female	76

dtype: int64

```
1 df["MaritalStatus"].value_counts()
```

MaritalStatus	count
Partnered	107
Single	73

dtype: int64

4.3.2 Unique Attributes:

The unique attributes for each categorical feature are listed.

```
for column in df.columns:
  if df[column].dtype == object:
    print(f"Unique values for {column}: {df[column].unique()}")

```

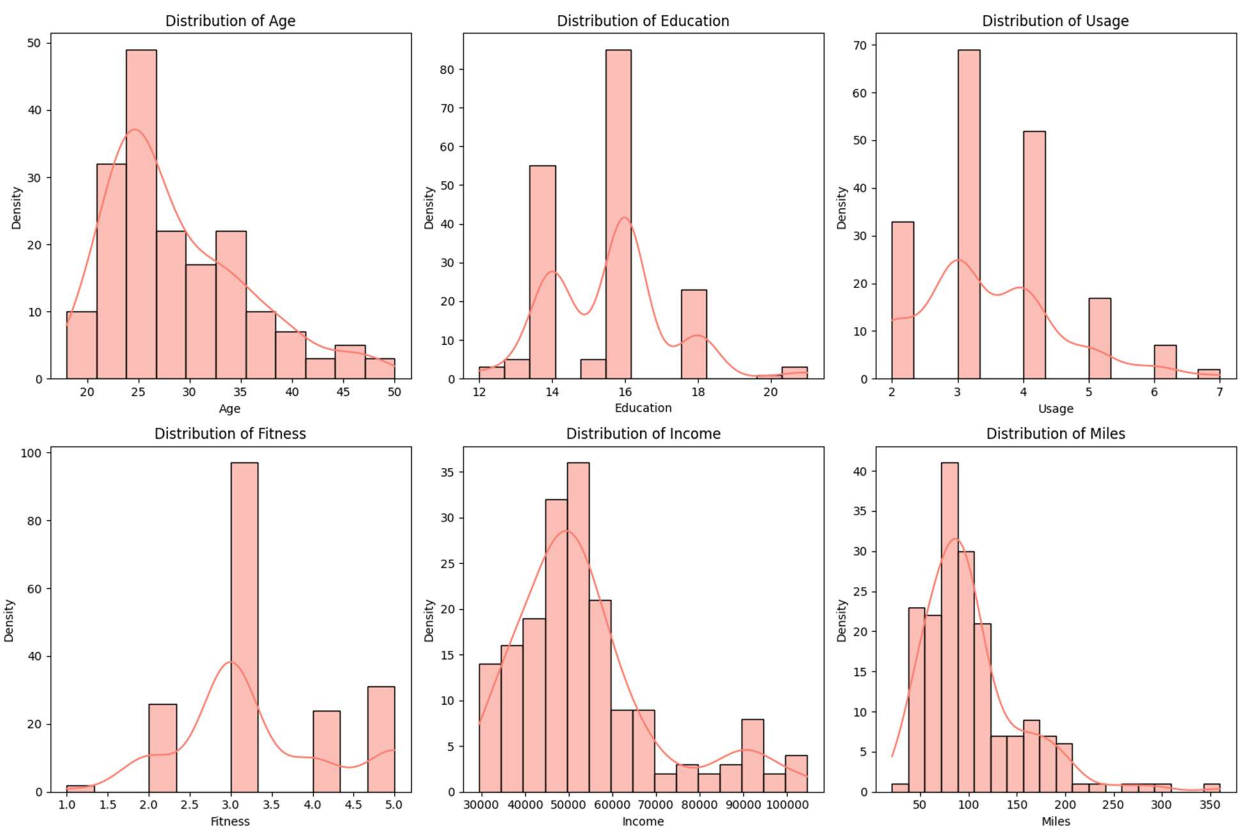
```
Unique values for Product: ['KP281' 'KP481' 'KP781']
Unique values for Gender: ['Male' 'Female']
Unique values for MaritalStatus: ['Single' 'Partnered']
```

4.4 Graphical Analysis

4.4.1 Univariate Analysis - Numerical Features

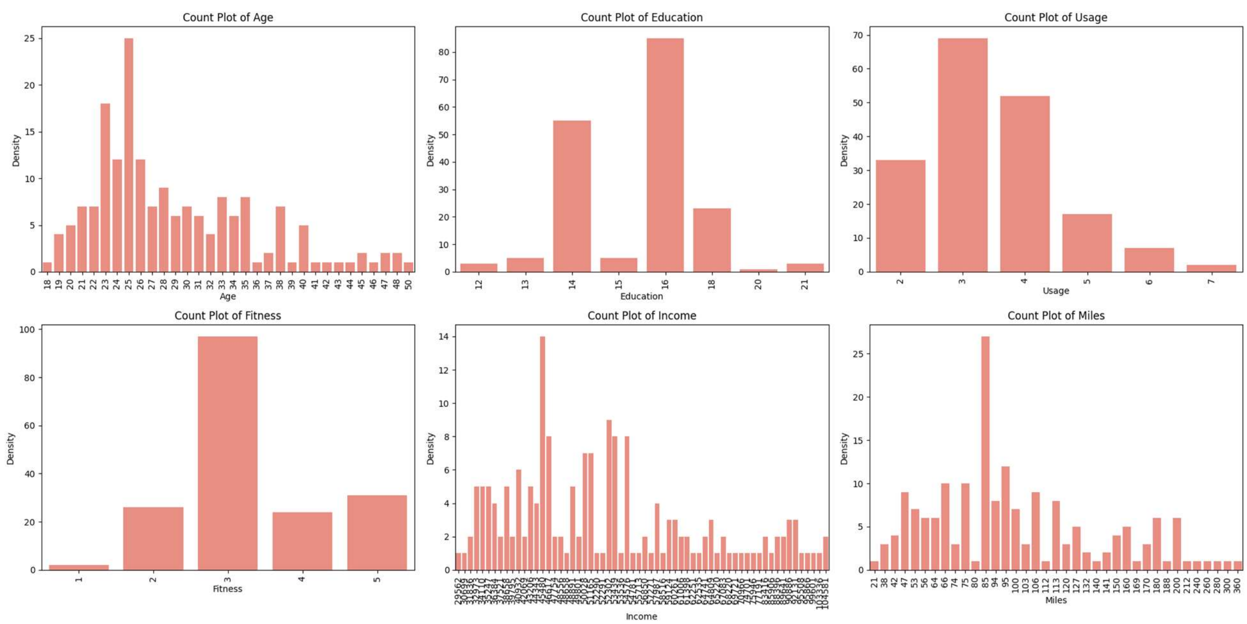
- **Distribution Plot**: Plots to visualize the distribution of numerical features such as age, income, and usage.

```
#D I S T R I B U T I O N    P L O T
numerical_features = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']

fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(15, 10))

for i, feature in enumerate(numerical_features):
  row = i // 3
  col = i % 3
  sns.histplot(df[feature], kde=True, ax=axes[row, col], color = "salmon")
  axes[row, col].set_title(f'Distribution of {feature}')
  axes[row, col].set_xlabel(feature)
  axes[row, col].set_ylabel('Density')

plt.tight_layout()
plt.show()

```



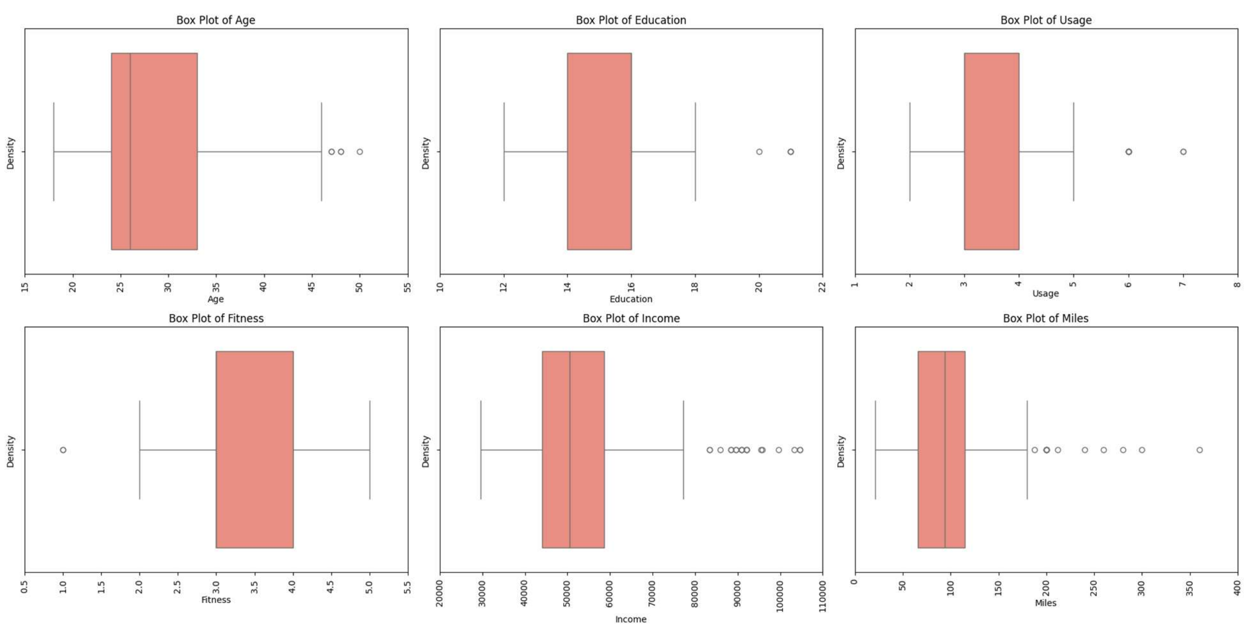
- **Count Plot**: Count plots for the frequency of different fitness levels and product types.

```python
# C O U N T   P L O T
numerical_features = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']

fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(20, 10))

for i, feature in enumerate(numerical_features):
  row = i // 3
  col = i % 3
  sns.countplot(x = df[feature], ax=axes[row, col], color = "salmon")
  axes[row, col].set_title(f'Count Plot of {feature}')
  axes[row, col].set_xlabel(feature)
  ticks = axes[row, col].get_xticks()
  # Set the x-axis ticks and rotate the labels
  axes[row, col].set_xticks(ticks, labels=axes[row, col].get_xticklabels(), rotation=90)
  axes[row, col].set_ylabel('Density')

plt.tight_layout()
plt.show()

```



- **Box Plot**: Box plots to identify the spread and potential outliers in numerical features.

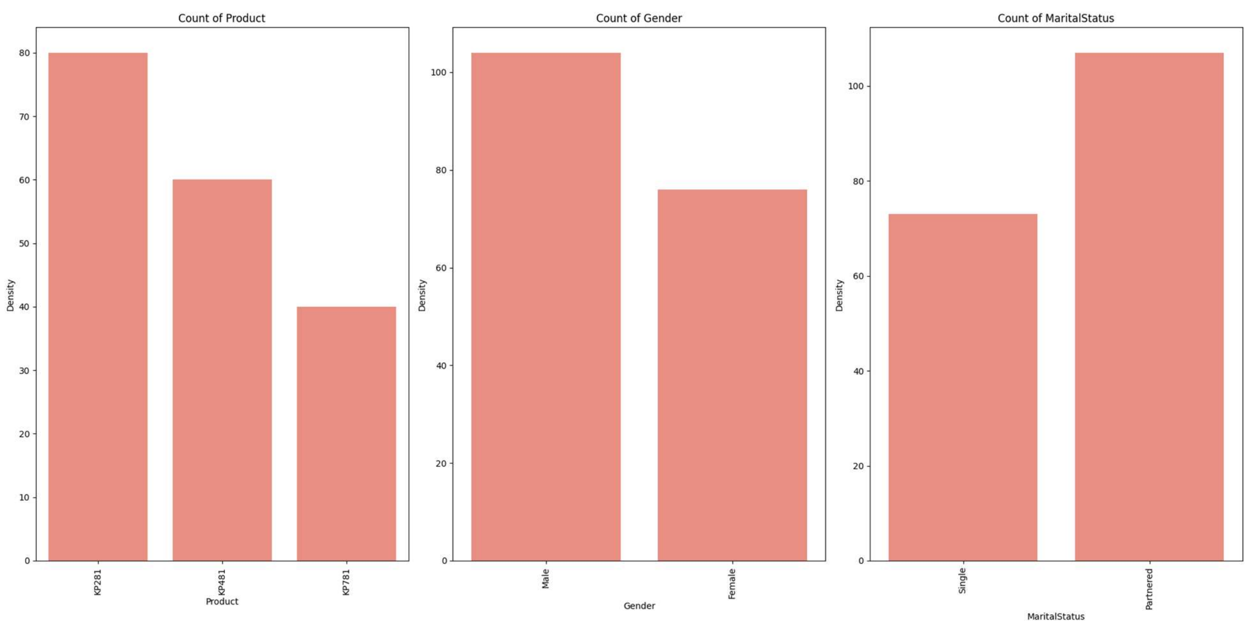
```python
# B O X   P L O T
numerical_features = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']

fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(20, 10))

for i, feature in enumerate(numerical_features):
  row = i // 3
  col = i % 3
  sns.boxplot(x = df[feature], ax=axes[row, col], color = "salmon")
  axes[row, col].set_title(f'Box Plot of {feature}')
  axes[row, col].set_xlabel(feature)
  ticks = axes[row, col].get_xticks()
  # Set the x-axis ticks and rotate the labels
  axes[row, col].set_xticks(ticks, labels=axes[row, col].get_xticklabels(), rotation=90)
  axes[row, col].set_ylabel('Density')

plt.tight_layout()
plt.show()

```



4.4.2 Univariate Analysis - Categorical Features

- **Count Plot**: Count plots to visualize the distribution of categorical features like gender and marital status.

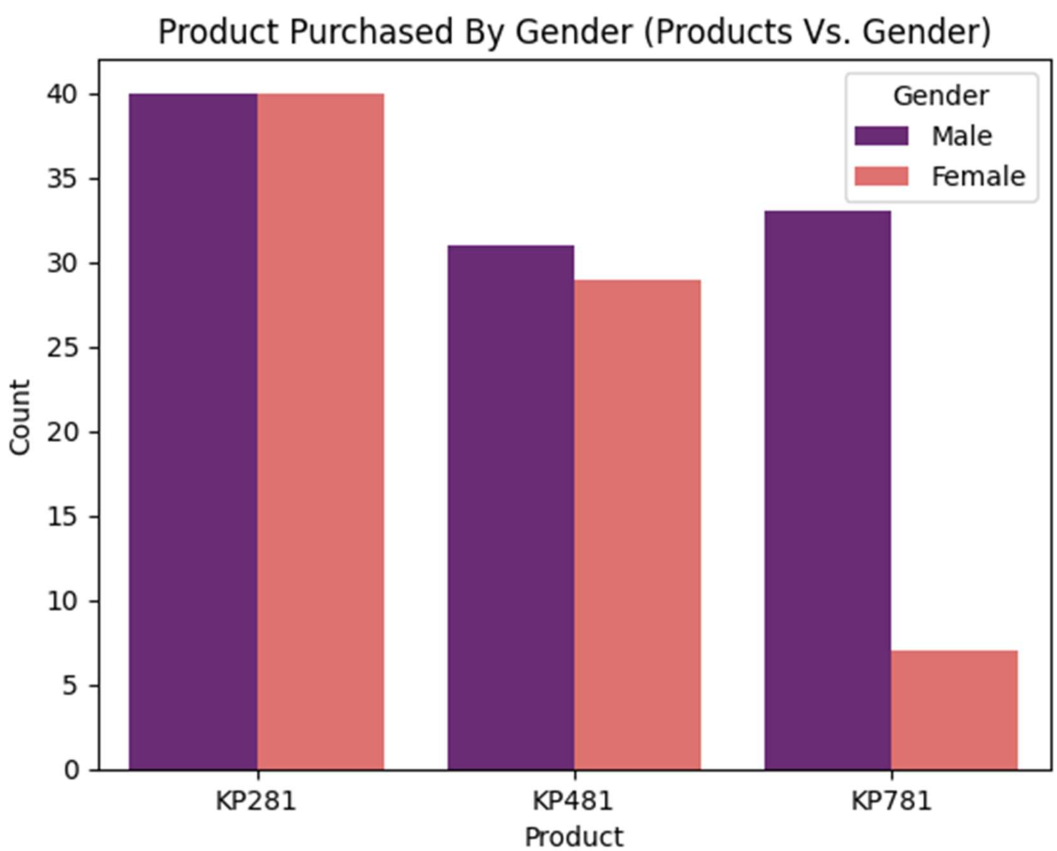
```python
# Count plot for Categorical Features

categorical_features = ["Product", "Gender", "MaritalStatus"]

fig, axes = plt.subplots(nrows=1, ncols=len(categorical_features), figsize=(20, 10))

for i, feature in enumerate(categorical_features):
  sns.countplot(x = df[feature], ax=axes[i], color = "salmon")
  axes[i].set_title(f'Count of {feature}')
  axes[i].set_xlabel(feature)
  ticks = axes[i].get_xticks()
  # Set the x-axis ticks and rotate the labels
  axes[i].set_xticks(ticks, labels=axes[i].get_xticklabels(), rotation=90)
  axes[i].set_ylabel('Density')

plt.tight_layout()
plt.show()

```



4.4.3 Bivariate Analysis

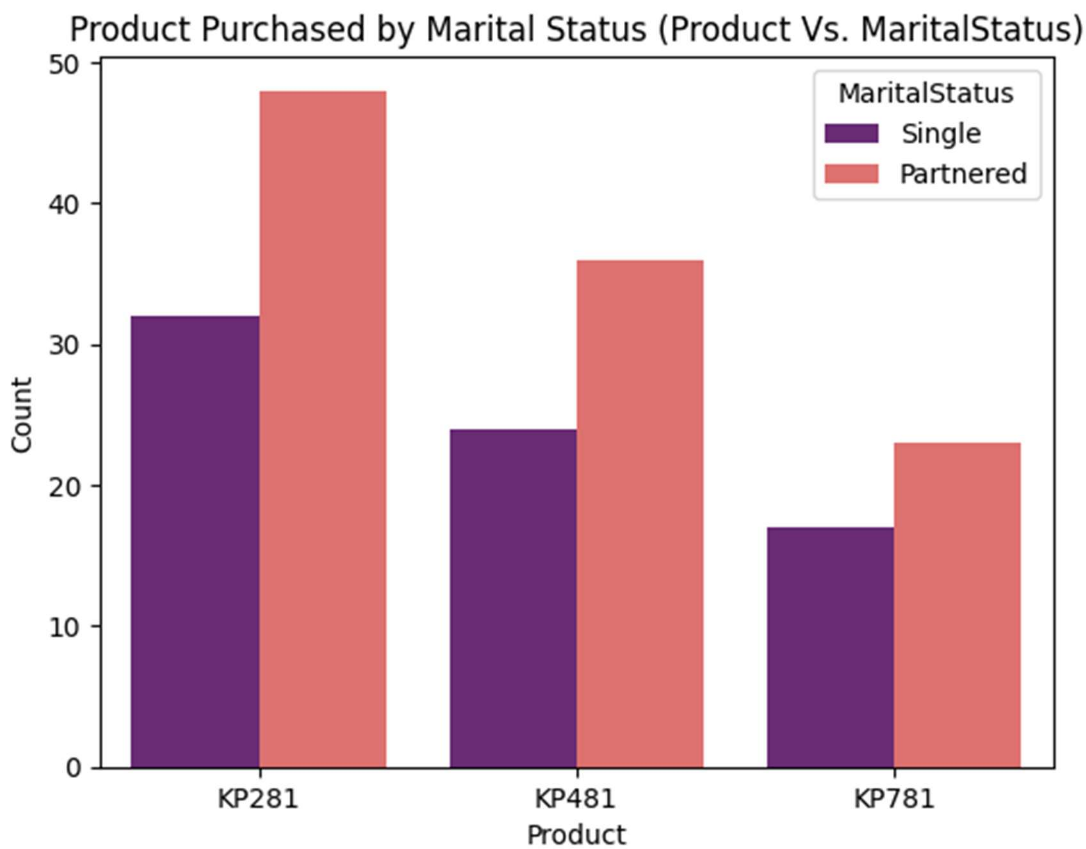
- **Product vs. Gender**: Analyzing the relationship between the product purchased and the customer's gender.

```
sns.countplot(x = "Product", hue = "Gender", data = df,  palette = "magma")
plt.xlabel("Product")
plt.ylabel("Count")
plt.title("Product Purchased By Gender (Products Vs. Gender)")
plt.show()
```



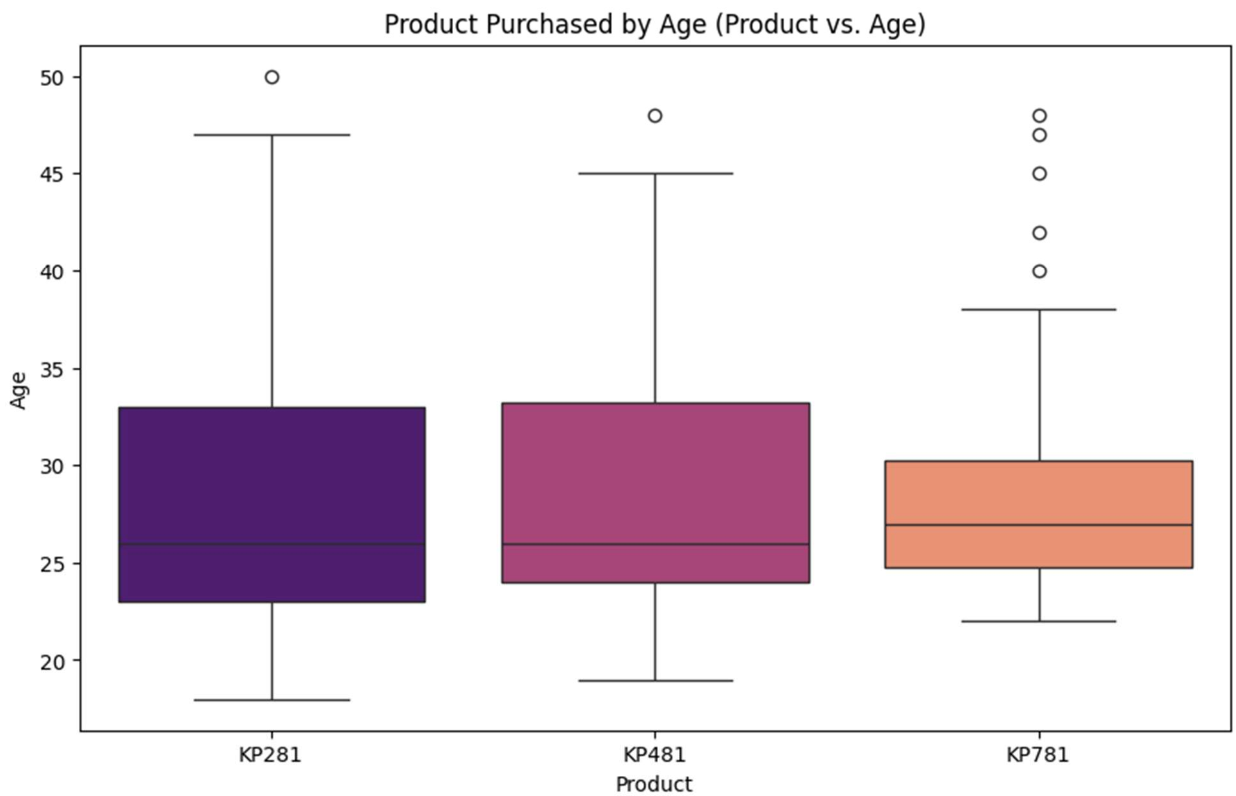
- **Product vs. Marital Status**: Examining how marital status influences product choice.

```
sns.countplot(x= "Product", hue = "MaritalStatus", data = df, palette = "magma")
plt.xlabel("Product")
plt.ylabel("Count")
plt.title("Product Purchased by Marital Status (Product Vs. MaritalStatus)")
plt.show()
```



- **Product vs. Age**: Exploring the relationship between product choice and customer age.

```
# Product vs Age
plt.figure(figsize=(10, 6))
sns.boxplot(x='Product', y='Age', data=df, palette='magma')
plt.xlabel('Product')
plt.ylabel('Age')
plt.title('Product Purchased by Age (Product vs. Age)')
plt.show()
```



4.4.4 Multivariate Analysis

- **Pair Plots**: Creating pair plots to show relationships among multiple features simultaneously.

```
plt.figure(figsize = (10,5))
sns.pairplot(df)
plt.show()
```

4.5 Correlation Analysis

4.5.1 Graph

```
numerical_columns = df.select_dtypes(include = ["number"])
numerical_columns
```

	Age	Education	Usage	Fitness	Income	Miles
0	18	14	3	4	29562	112
1	19	15	2	3	31836	75
2	19	14	4	3	30699	66
3	19	12	3	3	32973	85
4	20	13	4	2	35247	47
...	...	...	...	...	...	...
175	40	21	6	5	83416	200
176	42	18	5	4	89641	200
177	45	16	5	5	90886	160
178	47	18	4	5	104581	120
179	48	18	4	5	95508	180

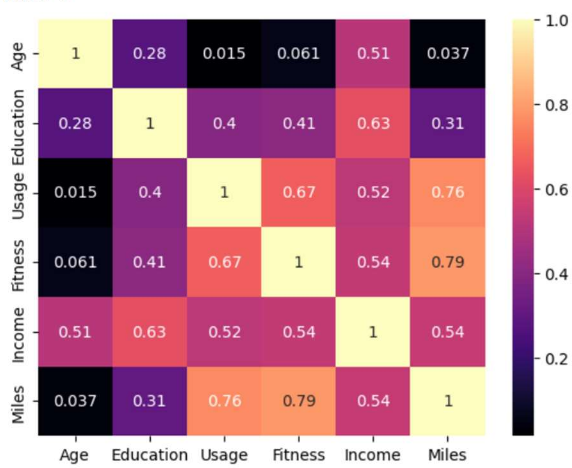
180 rows × 6 columns

```
correlation = numerical_columns.corr()
correlation
```

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

```
sns.heatmap(correlation, annot = True, cmap = "magma")
```

<Axes: >



4.5.2 Observations From the Heatmap

4.5.2.1 Strong Positive Correlation:

- 'Miles' and 'Usage' have a strong positive correlation (0.76), which is expected as people who plan to use the treadmill more often are likely to cover more miles.
- 'Income' and 'Miles' have a moderate positive correlation (0.54), indicating that individuals with higher incomes tend to cover more miles.
- 'Income' and 'Usage' show a moderate positive correlation (0.52), suggesting that those with higher incomes tend to use the treadmill more frequently.
- 'Age' and 'Fitness' have a moderate positive correlation (0.61), which implies that older individuals tend to rate their fitness higher.
- 'Education' and 'Income' show a moderate positive correlation (0.63), which implies that users with higher education have higher income.

4.5.2.2 Moderate Correlation:

- 'Age' and 'Income' have a moderate positive correlation (0.51), suggesting a potential trend where older individuals tend to have higher incomes.
- 'Education' and 'Age' have a moderate positive correlation (0.63), showing that higher education individuals may be older in age.

4.5.2.3 Weak or No Correlation:

- 'Fitness' and 'Miles' have a very weak correlation (0.06), indicating that self-rated fitness doesn't strongly influence how many miles users plan to cover.
- 'Fitness' and 'Usage' have a very weak correlation (0.05), indicating that self-rated fitness doesn't strongly influence how many times users plan to use the treadmill per week.

4.5.2.4 Other Observations

- Based on the correlation matrix, it's evident that 'Usage' and 'Miles' are strongly positively correlated, and 'Income' is moderately correlated with both of them.
- This indicates that users who plan to use the treadmill more and cover more miles tend to have higher incomes.
- Additionally, 'Age' and 'Fitness' are moderately positively correlated, suggesting that older individuals may rate their fitness higher.

4.6 Outlier Detection

Outliers are detected using the Interquartile Range (IQR) method, allowing identification of extreme values that may affect analysis.

```
def iqr_outliers(df):
  Q1 = np.percentile(df, 25)
  Q3 = np.percentile(df, 75)
  iqr = Q3 - Q1
  lower_bound = Q1 - 1.5 *iqr
  upper_bound = Q3 + 1.5*iqr
  outliers = [x for x in df if x < lower_bound or x > upper_bound]
  return outliers
```

```
print(numerical_features)
```

```
['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
```

```
for features in numerical_features:
  outliers = iqr_outliers(df[features])
  print(f"Outliers in {features}: {outliers}")
```

```
Outliers in Age: [47, 50, 48, 47, 48]
Outliers in Education: [20, 21, 21, 21]
Outliers in Usage: [6, 6, 6, 7, 6, 7, 6, 6, 6]
Outliers in Fitness: [1, 1]
Outliers in Income: [83416, 88396, 90886, 92131, 88396, 85906, 90886, 103336, 99601, 89641, 95866, 92131, 92131, 104581, 83416, 89641, 90886, 104581, 95508]
Outliers in Miles: [188, 212, 200, 200, 200, 240, 300, 280, 260, 200, 360, 200, 200]
```

4.7 Conditional Probabilities

4.7.1 Product Purchases:

The percentage of customers purchasing KP281, KP481, or KP781 is calculated.

What percent of customers have purchased KP281, KP481, or KP781?

```
percentage_cust = round(df["Product"].value_counts(normalize = True) * 100, 2)
# percentage_cust
customer_product_purchased = pd.DataFrame({
    "Product" : percentage_cust.index,
    "Percentage (%)" : percentage_cust.values
})
customer_product_purchased
```

	Product	Percentage (%)
0	KP281	44.44
1	KP481	33.33
2	KP781	22.22

4.7.2 Product – Gender:

- Percentage of male customers purchasing a treadmill.

```
# percentage of male customers purchasing a treadmill
male_count = round((df["Gender"].value_counts(normalize = True)["Male"]) * 100, 2)
# male_count
male_customers_purchased = pd.DataFrame({
    "Gender" : ["Males"],
    "percentage (%)" : male_count
})
male_customers_purchased
```

	Gender	percentage (%)
0	Males	57.78

- Percentage of female customers purchasing KP781.

```python
# extracting just the female count
df[(df["Product"] == "KP781")]["Gender"].value_counts()["Female"]
```

7

```python
# Percentage of female customers who bought KP781 treadmill.
female_KP781 = round((df[(df["Product"] == "KP781")]["Gender"].value_counts(normalize = True)["Female"]) * 100, 2)
female_KP781_percentage = pd.DataFrame({
    "Gender" : ["Females"],
    "Percentage (%) of females who bought KP781" : female_KP781
})

female_KP781_percentage
```

	Gender	Percentage (%) of females who bought KP781
0	Females	17.5

- Probability of a customer being female given that the product is KP281.

```python
# count of females who bought KP281
female_count_KP281 = df[df["Product"] == "KP281"]["Gender"].value_counts()["Female"]
female_count_KP281
```

40

```python
# total count of KP281 purchased
KP281_count = df["Product"].value_counts()["KP281"]
KP281_count
```

80

```python
# finding probablity of customers being female given that product is KP281
probability_female_KP281 = female_count_KP281 / KP281_count
probability_female_KP281
```

0.5

```python
# fact table for the probablity of customers being female given that product is KP281
probability_fact = pd.DataFrame({
    "Gender": ["Female"],
    "Female_Kp281_count" : female_count_KP281,
    "KP281_count" : KP281_count,
    "Probability of customer being female and product is KP281" : probability_female_KP281
})

probability_fact
```

	Gender	Female_Kp281_count	KP281_count	Probability of customer being female and product is KP281
0	Female	40	80	0.5

4.7.3 Product – Age:

- Percentage of customers aged between 20 and 30 among all customers.

- Product – Age

1. Percentage of customers with Age between 20s and 30s among all customers

+ Code + Text

```
[ ]
# count of customers with age between 20s and 30s among all customers
((df["Age"] >=20) & (df["Age"]>=30)).value_counts()[True]

```

```
67
```

```
[ ]
# percentage of customers with age between 20s and 30s among all customers
age_percentage = round((((df["Age"] >=20) & (df["Age"]>=30)).value_counts(normalize = True)[True]) * 100, 2)
age_percentage
```

```
37.22
```

```
[ ]
# fact table for percentage of customers with age between 20s and 30s among all customers
age_percentage_fact = pd.DataFrame({
    "Age" : ["20s to 30s"],
    "Percentage (%)" : age_percentage
})

age_percentage_fact
```

	Age	Percentage (%)
0	20s to 30s	37.22

4.7.4 Product – Income:

- Percentage of low-income customers purchasing a treadmill.

```
[ ]
# low_income threshold
low_income_threshold = 50000
```

+ Code + Text

```
[ ]
# low income customers count
low_income_customers = df[df["Income"] <= low_income_threshold].shape[0]
low_income_customers
```

```
83
```

+ Code + Text

```
# total_customers
total_customers = df.shape[0]
total_customers
```

```
180
```

```
[ ]
# percentage of low income customers purchasing a treadmill

percentage_low_income = round((low_income_customers / total_customers) * 100, 2)

# percentage_low_income

percentage_low_income_fact = pd.DataFrame({
    "Low Income Customers" : [low_income_customers],
    "total_customers" : [total_customers],
    "Percentage of low income customers" : [percentage_low_income]
})
percentage_low_income_fact
```

	Low Income Customers	total_customers	Percentage of low income customers
0	83	180	46.11

- Percentage of high-income customers purchasing KP781.

```python

# High-income threshold
high_income_threshold = 80000

# Customers with high income who purchased KP781
high_income_kp781_customers = df[(df['Income'] >= high_income_threshold) & (df['Product'] == 'KP781')].shape[0]

# Total number of customers who purchased KP781
total_kp781_customers = df[df['Product'] == 'KP781'].shape[0]

# Percentage of high-income customers purchasing KP781
percentage_high_income_kp781 = (high_income_kp781_customers / total_kp781_customers) * 100 if total_kp781_customers > 0 else 0

percentage_high_income_kp78_fact = pd.DataFrame({
    "High Income Customers" : [high_income_kp781_customers],
    "Total KP781 Customers" : [total_kp781_customers],
    "Percentage of high income customers purchasing KP781" : [percentage_high_income_kp781]
})

percentage_high_income_kp78_fact

```

	High Income Customers	Total KP781 Customers	Percentage of high income customers purchasing KP781
0	19	40	47.5

- Percentage of high-income customers buying a treadmill given that the product is KP781.

```python
# Customers with high income who purchased KP781
high_income_kp781_customers = df[(df['Income'] >= high_income_threshold) & (df['Product'] == 'KP781')].shape[0]

# Total number of customers who purchased KP781
total_kp781_customers = df[df['Product'] == 'KP781'].shape[0]

# Percentage of high-income customers purchasing KP781 given that the product is KP781
percentage_high_income_kp781_given_kp781 = (high_income_kp781_customers / total_kp781_customers) * 100 if total_kp781_customers > 0 else 0

percentage_high_income_kp781_given_kp781_fact = pd.DataFrame({
    "High Income Customers who bought KP781" : [high_income_kp781_customers],
    "Total KP781 Customers" : [total_kp781_customers],
    "Percentage of high income customers purchasing KP781 given KP781" : [percentage_high_income_kp781_given_kp781]
})

percentage_high_income_kp781_given_kp781_fact

```

	High Income Customers who bought KP781	Total KP781 Customers	Percentage of high income customers purchasing KP781 given KP781
0	19	40	47.5

4.7.5 Product – Fitness:

- Percentage of customers with a fitness level of 5.

```python
percentage_fitness_5 = round((df["Fitness"].value_counts(normalize = True)[5]) * 100, 2)
# print(f"The Percentage of customers that have fitness level 5 is : {percentage_fitness_5}%")

percentage_fitness_5_fact = pd.DataFrame({
    "Fitness Level" : [5],
    "Percentage (%)" : [percentage_fitness_5]
})

percentage_fitness_5_fact

```

	Fitness Level	Percentage (%)
0	5	17.22

- Percentage of fitness level 5 customers purchasing KP781.

```python
percentage_fitness_5_product_kp781 = round((df[(df["Fitness"] ==5)]["Product"].value_counts(normalize = True)["KP781"]) * 100, 2)

percentage_fitness_5_product_kp781_fact = pd.DataFrame({
    "Fitness Level" : [5],
    "Percentage (%)" : [percentage_fitness_5_product_kp781]
})

percentage_fitness_5_product_kp781_fact

```

	Fitness Level	Percentage (%)
0	5	93.55

- Percentage of customer with fitness level 5 buying KP781 treadmill

```python
fitness_5_kp781_customers = df[(df['Fitness'] == 5) & (df['Product'] == 'KP781')]
num_fitness_5_kp781_customers = len(fitness_5_kp781_customers)
num_kp781_customers = len(df[df['Product'] == 'KP781'])

percentage_fitness_5_kp781 = round((num_fitness_5_kp781_customers / num_kp781_customers) * 100, 2)
percentage_fitness_5_kp781_buying = pd.DataFrame({
    "Fitness Level" : [5],
    "Percentage (%)" : [percentage_fitness_5_kp781]
})

percentage_fitness_5_kp781_buying

```

	Fitness Level	Percentage (%)
0	5	72.5

4.7.6 Product - Marital Status:

- Percentage of partnered customers using treadmills.

```python
partnered_customers = df[df['MaritalStatus'] == 'Partnered'].shape[0]
partnered_customers

```

```
107
```

```python
total_customers
```

```
180
```

```python
percentage_partnered = round((partnered_customers / total_customers) * 100, 2)
percentage_partnered
```

```
59.44
```

```python

partnered_percentage_fact = pd.DataFrame({
    'Marital Status': ['Partnered'],
    'Percentage (%)': [percentage_partnered]
})

partnered_percentage_fact
```

	Marital Status	Percentage (%)
0	Partnered	59.44

5 Customer Demographics

5.1 Gender Distribution

- Total Females Who Bought KP781: **7**
- Percentage of Females Buying KP781: **17.5%**
- Total Females Who Bought KP281: **40**
- Total Purchases of KP281: **80**
- Probability of Customer Being Female for KP281: **0.5**

5.2 Age Distribution

- Customers Aged 20s to 30s: **67**
- Percentage of Customers Aged 20s to 30s: **37.22%**

5.3 Income Analysis

5.3.1 Low-Income Customer Insights:

- Low-Income Customers (Income ≤ $50,000): **83**
- Total Customers: **180**
- Percentage of Low-Income Customers Purchasing a Treadmill: **46.11%**

5.3.2 High-Income Customer Insights:

- High-Income Customers (Income >= $80,000) : **19**
- Total KP781 Customers : **40**
- Percentage of High-income Customers Purchasing a Treadmill KP781: **47.5**

5.3.3 High-Income customer who Purchased KP781

- High-Income customer who bought KP781: **19**
- Total KP781 customers : **40**
- Percentage of High-income customers purchasing KP781 given that the product is KP781: **47.5**

5.4 Fitness Level Insights

5.4.1 Fitness Level 5 Analysis:

- Percentage of Customers with Fitness Level 5: **17.22%**
- Percentage of Fitness Level 5 Customers Buying KP781: **93.55%**
- Percentage of KP781 Customers with Fitness Level 5: **72.5%**

5.5 Marital Status Analysis

5.5.1 Partnered Customers:

- Total Partnered Customers: **107**
- Total Customers: **180**
- Percentage of Partnered Customers: **59.44%**

6 Insights and Recommendations

Based on the comprehensive analysis of the dataset, the following actionable insights and recommendations are provided to improve product offerings, customer targeting, and business strategies for Aerofit treadmills:

6.1 Target Audience

6.1.1 Insight:

A significant portion of customers purchasing treadmills are males, particularly those who are partnered/married.

6.1.2 Recommendation:

- Focus marketing efforts on these segments, highlighting the benefits and features that cater to their specific needs.
- Consider tailoring marketing campaigns and promotional offers to attract and retain male customers.

6.2 Product Portfolio

6.2.1 Insight:

KP281 and KP481 are the most popular products, appealing to customers who prefer mid-range and entry-level treadmills.

6.2.2 Recommendation:

- Continue investing in developing and promoting these products to capitalize on their market demand.
- Consider introducing new models that fall within these price ranges and cater to specific customer needs, such as features related to fitness tracking, entertainment options, and customization.

6.3 Pricing Strategy

6.3.1 Insight:

The moderate correlation between income levels and treadmill usage indicates some price sensitivity among customers.

6.3.2 Recommendation:

- Analyze the relationship between price points and sales volumes for each product to optimize pricing strategies.
- Explore options like tiered pricing or bundled offers to attract customers across different income levels while maximizing revenue.

6.4 Marketing Strategies

6.4.1 Insight:

High-income customers and fitness enthusiasts are likely to be drawn to treadmills with advanced features.

6.4.2 Recommendation:

- Highlight the health and fitness benefits of owning a treadmill, particularly targeting those with higher incomes or greater interest in exercise.
- Use targeted advertising and promotions to reach potential customers based on demographics (age, gender, marital status) and interests (fitness, sports, health).
- Develop content marketing strategies that focus on providing valuable information about the benefits of regular exercise, fitness tips, and how treadmills can help achieve specific goals.

6.5 Customer Retention

6.5.1 Insight:

Customer engagement can be enhanced through personalized experiences and exclusive offers.

6.5.2 Recommendation:

- Encourage customer loyalty by building a strong relationship with users, providing excellent after-sales support, and offering exclusive benefits.
- Consider launching programs like fitness challenges, user communities, or educational resources to increase engagement and retain existing customers.
- Collect customer feedback through surveys or online reviews to understand their experiences and address any pain points.

6.6 Product Improvements

6.6.1 Insight:

Customers with higher fitness levels tend to purchase the KP781 treadmill, suggesting a demand for advanced features.

6.6.2 Recommendation:

- Research potential improvements to product features, taking into consideration factors like age and fitness levels.
- Analyze data from customers with a fitness level of 5 and those who purchase KP781 to assess the need for additional advanced features.
- Introduce features that cater to users' preferences, such as improved console interfaces, personalized workout programs, and advanced tracking capabilities.

6.7 Geographic Targeting

6.7.1 Insight:

Certain geographical areas may exhibit a higher demand for treadmills based on regional fitness trends and preferences.

6.7.2 Recommendation:

- Consider conducting market research to understand the specific geographical areas where there is a high demand for treadmills.

- Tailor marketing campaigns and product availability based on regional preferences and needs.
- Consider partnering with fitness centers or gyms in specific locations to increase product awareness and sales.

6.8 Data-Driven Decisions

6.8.1 Insight:

Data analytics offer valuable insights into customer behavior and preferences.

6.8.2 Recommendation:

- Continue utilizing data analysis to monitor sales trends, customer behaviors, and marketing campaign effectiveness.
- Use insights derived from the analysis to inform decision-making, optimize pricing, and refine marketing strategies.
- Invest in building a robust data analytics infrastructure to enable efficient data collection, analysis, and reporting.

7 Conclusion

This comprehensive analysis of Aerofit treadmill purchases provides valuable insights into customer demographics, preferences, and behaviors. By implementing the recommended strategies and continuously monitoring market trends, Aerofit can strengthen its market position, enhance customer satisfaction, and drive sales growth.