🏋️ Aerofit - Descriptive Statistics & Probability 📊

Business Problem

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company. This will help them provide better recommendations to new customers. The team plans to investigate whether there are differences in customer characteristics across the products.

1. Perform Descriptive Analytics:

- Create a customer profile for each AeroFit treadmill product.
- Develop appropriate tables and charts to summarize the data and identify patterns.

2. Construct Two-Way Contingency Tables:

- For each AeroFit treadmill product:
  - Build two-way contingency tables to explore relationships between variables (e.g., treadmill type vs. age group).
  - Compute:
    - **Conditional Probabilities**: Probability of one event given another.
    - **Marginal Probabilities**: Overall probability of events.
- Provide insights and discuss their impact on the business.

Let's analyze the data to uncover trends and actionable insights for AeroFit! 🚀

About Aerofit 🏋️

Aerofit is a leading brand in the field of fitness equipment. The company offers a wide range of products, including treadmills, exercise bikes, gym equipment, and fitness accessories, to cater to the needs of people from all walks of life.

🏋️ AeroFit - Customer Data Overview 📊

Dataset Description

The company collected data on individuals who purchased a treadmill from AeroFit stores during the past three months. The dataset includes the following features:

Features:

1. **Product Purchased**: The type of treadmill purchased (KP281, KP481, or KP781).

2. **Age**: The age of the customer (in years).
3. **Gender**: Gender of the customer (Male/Female).
4. **Education**: Years of education completed by the customer.
5. **Marital Status**: Whether the customer is single or partnered.
6. **Usage**: The average number of times the customer plans to use the treadmill each week.
7. **Income**: The annual income of the customer (in $).
8. **Fitness**: Self-rated fitness level on a scale of 1 to 5:
    - 1: Poor shape
    - 5: Excellent shape
9. **Miles**: The average number of miles the customer expects to walk/run each week.

1. Defining Problem Statement and Analysing basic metrics

Problem Statement

AeroFit aims to understand the customer profiles for each treadmill type to offer better recommendations to new buyers.

Customer Profiling - Analyze and Summarize:

- Explore customer characteristics for each treadmill product.
- Highlight key patterns and trends in the data using tables and charts.

Exploring Relationships - Investigate Relationships:

- Examine how different customer characteristics (e.g., age, gender, income) relate to the choice of treadmill.
- Use **two-way contingency tables** to calculate probabilities, showing:
    - How likely one event is when another occurs (e.g., which age group prefers a specific treadmill type).

Outcome:

This analysis will provide valuable insights to:

- Help AeroFit tailor its marketing strategies.
- Enhance customer recommendations.
- Better align products with target customer groups.

In [103…]
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [104…]
```
!gdown 1Uwm8UVs-5qqz3oElA95t0x8FNAwkjdNq
```

```
Downloading...
From: https://drive.google.com/uc?id=1Uwm8UVs-5qqz3oElA95t0x8FNAwkjdNq
To: /content/aerofit_treadmill_dataset.txt

  0% 0.00/7.28k [00:00<?, ?B/s]
100% 7.28k/7.28k [00:00<00:00, 17.7MB/s]
```

In [105…]
```
df = pd.read_csv('aerofit_treadmill_dataset.txt', on_bad_lines='warn')
```

In [106…]
```
print("First 5 rows of the dataset:\n")
df.head()
```

```
First 5 rows of the dataset:
```

Out[106…]

	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

Basic metrics

Basic metrics include:

- **Number of records and attributes** in the dataset.
- **Data types** of columns.
- **Count of unique values**.
- **Presence of missing values**.

In [107…
```
# Basic information about the dataset
```

In [108…
```
print("Basic Information about the dataset:\n")
df.info()
```

```
Basic Information about the dataset:

<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
```

In [109…
```
# Shape of the dataset
```

In [110…
```
print("Shape of the dataset:")
print(f"Number of rows: {df.shape[0]}, Number of columns: {df.shape[1]}")
```

```
Shape of the dataset:
Number of rows: 180, Number of columns: 9
```

In [111…
```
# Checking for missing values
```

In [112…
```
print("Missing Values Count:\n")
print(df.isnull().sum())
```

```
Missing Values Count:

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

In [113…
```python
# Count of unique values in each column
```

In [114…
```python
print("Unique Value Count in Each Column:\n")
for col in df.columns:
    print(f"{col}: {df[col].nunique()} unique values")
```

```
Unique Value Count in Each Column:

Product: 3 unique values
Age: 32 unique values
Gender: 2 unique values
Education: 8 unique values
MaritalStatus: 2 unique values
Usage: 6 unique values
Fitness: 5 unique values
Income: 62 unique values
Miles: 37 unique values
```

Insights from Basic Metrics

Shape & Size:

- **180 rows**, **9 columns** – a manageable dataset for analysis.

Data Types:

- **Numeric**: Age, Education, Usage, Fitness, Income, Miles.
- **Categorical**: Product, Gender, MaritalStatus.

Missing Data:

- **None**, all fields are complete.

Unique Values:

- **Product**: 3 types.

- **Gender & MaritalStatus**: 2 each.
- **Education**: 8 levels.
- **Age, Usage, Fitness, Income, Miles**: Multiple distinct values.

Key Observations:

- The dataset is **clean** and has **no missing information**, making it easy to analyze right away.
- It has a good balance of **demographic details** (like Age, Gender, Education, MaritalStatus), **product usage and fitness measures** (Usage, Fitness, Miles), and an **economic indicator** (Income).
- Because the **numbers** (like Age, Income, and Miles) cover a good range, it's possible to compare different groups (like different Product categories or Gender) easily.

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

1.1.1 Observations on the Shape of Data

In [115…
```
# Observing the shape of the dataset
```

In [116…
```
print("Number of records (rows):", df.shape[0])
print("Number of attributes (columns):", df.shape[1])
```

```
Number of records (rows): 180
Number of attributes (columns): 9
```

1.1.2 Data Types of All Attributes

In [117…
```
# Displaying data types
```

In [118…
```
print("Data types of each column before conversion:")
print(df.dtypes)
```

```
Data types of each column before conversion:
Product          object
Age               int64
Gender           object
Education         int64
MaritalStatus    object
Usage             int64
Fitness           int64
Income            int64
Miles             int64
dtype: object
```

In [119…
```
# Count of unique data types
```

In [120…
```
print("Unique Data Types Count: \n")
print(df.dtypes.value_counts())
```

```
Unique Data Types Count:

int64     6
object    3
Name: count, dtype: int64
```

1.1.3 Conversion of Categorical Attributes to 'Category'

In [121…
```
# Columns that we want to treat as categorical
categorical_cols = ['Product', 'Gender', 'MaritalStatus']

# Convert the identified columns to the 'category' data type
df[categorical_cols] = df[categorical_cols].astype('category')

print("Data types of each selected categorical column after conversion:\n")
print(df.dtypes[categorical_cols])
```

```
Data types of each selected categorical column after conversion:

Product          category
Gender           category
MaritalStatus    category
dtype: object
```

In [122…
```
print("Data types of all columns after conversion:\n")
print(df.dtypes)
```

```
Data types of all columns after conversion:

Product          category
Age                 int64
Gender           category
Education           int64
MaritalStatus    category
Usage               int64
Fitness             int64
Income              int64
Miles               int64
dtype: object
```

1.1.4 Statistical Summary

In [123…
```
# For numeric columns
print("Statistical summary for numeric attributes:\n")
print(df.describe())
```

```
Statistical summary for numeric attributes:

              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
```

In [124…
```
# For categorical columns, include='category' helps get the count, unique, top, and freq of categorical data
print("Statistical summary for categorical attributes:\n")
print(df.describe(include='category'))
```

```
Statistical summary for categorical attributes:

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

Insights

Dataset Overview:

- **Size**: The dataset consists of **180 rows** and **9 columns**.
- **Key Attributes**: Includes **3 categorical variables** (Product, Gender, MaritalStatus) and **6 numeric variables** (Age, Education, Usage, Fitness, Income, Miles).

Missing Data:

- **No significant missing values detected**; all attributes are well-represented.

Key Observations:

- **Product Preferences**: **KP281** is the most frequently purchased product.
- **Demographics**: Slightly more males, with an **average age of about 29** and an **education level typically around 15–16 years**.
- **Usage & Fitness**: Average weekly usage is **around 3–4 times**, and fitness levels generally hover around a **moderate rating of 3** (on a 1–5 scale).
- **Income & Miles**: Typical income is **approximately $53,720**, and the **average mileage covered is around 103 miles**.

---

2. Non-Graphical Analysis: Value counts and unique attributes

2.1 Value Counts

Value counts show the frequency of each unique value in categorical columns, helping to understand the distribution of categories in the dataset.

In [125…
```
# Print value counts for 'Product'
print("Value counts for 'Product':\n")
print(df['Product'].value_counts())
```

```
Value counts for 'Product':

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

In [126…
```
# Print value counts for 'Gender'
print("Value counts for 'Gender':\n")
print(df['Gender'].value_counts())
```

```
Value counts for 'Gender':

Gender
Male      104
Female     76
Name: count, dtype: int64
```

In [127…
```
# Print value counts for 'MaritalStatus'
print("Value counts for 'MaritalStatus':\n")
```

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

```
Value counts for 'MaritalStatus':

MaritalStatus
Partnered    107
Single        73
Name: count, dtype: int64
```

Insights From Value Counts

Product:

- **KP281**: The most purchased product with **80 buyers**, indicating a clear preference.
- **KP481**: The second most popular product, purchased by **60 buyers**.
- **KP781**: The least purchased product, with **40 buyers**.

Gender:

- **Male customers**: **104 buyers**, slightly more than females.
- **Female customers**: **76 buyers**, showing a small difference in favor of males.

Marital Status:

- **Partnered individuals**: **107 buyers**, forming the larger portion of the market.
- **Single customers**: **73 buyers**, contributing a smaller but significant share.

2.2 Unique Attributes

Let's examine the unique values in each column to understand the diversity and variety within the dataset.

In [128…
```
# Count of unique values in each column using nunique()
```

In [129…
```
print("Count of unique values in each column:")
print(df.nunique())
```

```
Count of unique values in each column:
Product           3
Age              32
Gender            2
Education         8
MaritalStatus     2
Usage             6
Fitness           5
Income           62
Miles            37
dtype: int64
```

In [130…
```
# Actual unique values for each categorical column using unique()
```

In [131…
```
for i, col in enumerate(categorical_cols):
    if i > 0:
        print()
    print(f"Unique values in '{col}':")
    print(df[col].unique())
```

```
Unique values in 'Product':
['KP281', 'KP481', 'KP781']
Categories (3, object): ['KP281', 'KP481', 'KP781']

Unique values in 'Gender':
['Male', 'Female']
Categories (2, object): ['Female', 'Male']

Unique values in 'MaritalStatus':
['Single', 'Partnered']
Categories (2, object): ['Partnered', 'Single']
```

In [132…
```
# Here we use 'Fitness' as an example numeric column.
```

In [133…
```
numeric_col = 'Fitness'
print(f"Number of unique values in '{numeric_col}': {df[numeric_col].nunique()}")
print(f"Actual unique values in '{numeric_col}': {df[numeric_col].unique()}")
```

```
Number of unique values in 'Fitness': 5
Actual unique values in 'Fitness': [4 3 2 1 5]
```

Insights From Unique Values

Product:

- **3 unique products**: KP281, KP481, and KP781.
- Falls under **least diversity** with limited unique categories.

Age:

- **32 unique age groups**, indicating a wide age range among customers.
- Part of **highest diversity**, reflecting significant variation.

Gender:

- **Two genders**: Male and Female.
- Belongs to **least diversity** with minimal variation.

Education:

- **8 unique education levels**, reflecting diverse educational backgrounds.
- Falls under **moderate diversity**, capturing a balanced range.

Marital Status:

- **Two categories**: Single and Partnered, showing simple marital demographics.
- Part of **least diversity** with very few unique categories.

Usage:

- **6 unique usage levels**, highlighting varying product utilization patterns.
- Falls into **moderate diversity**, indicating a reasonable spread.

Fitness:

- **5 unique fitness levels** (1 to 5), indicating differences in fitness priorities.
- Part of **moderate diversity**, with some variety in fitness habits.

Income:

- **62 unique income levels**, showing substantial income diversity.
- Falls under **highest diversity**, reflecting a wide range of incomes.

Miles:

- **37 unique distance categories**, indicating varied travel habits among customers.
- Included in **highest diversity**, demonstrating significant variation.

---

3. Visual Analysis - Univariate & Bivariate

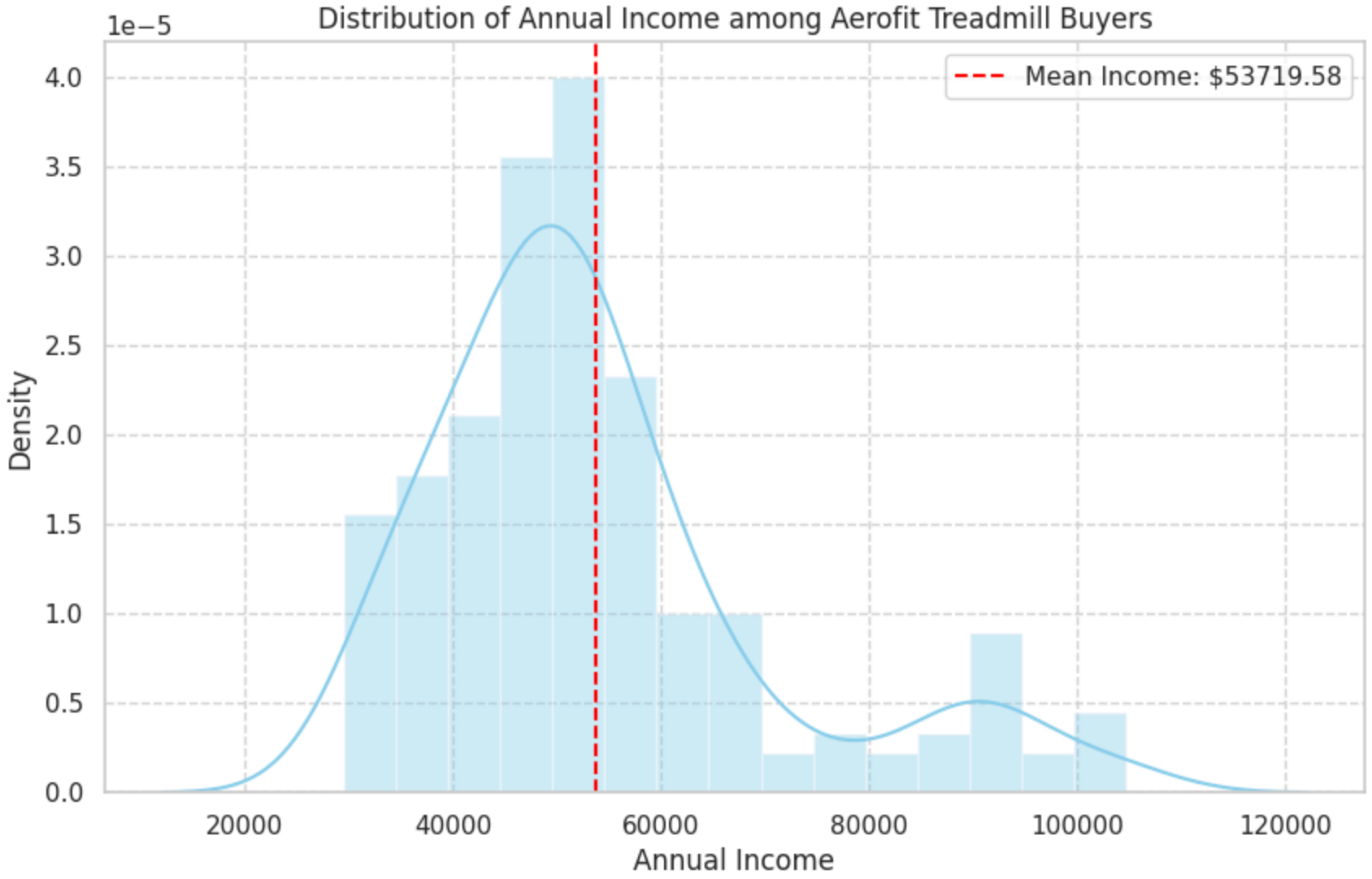
In [134…
```
# Save the original dataset
df_raw = df.copy()
```

A) Descriptive Analysis

3.1 Univariate Distribution Analysis: Income and Usage Patterns

3.1.1 What is the distribution of annual income among Aerofit treadmill buyers?

In [135…
```
# Plot the distribution of Annual Income
plt.figure(figsize=(10, 6))  # Set the figure size
sns.distplot(df['Income'], kde=True, hist=True, color='skyblue')

# Calculate the mean income
mean_income = df['Income'].mean()  # Compute the mean of the 'Income' column

# Add a vertical dashed red line to indicate the mean income
plt.axvline(mean_income, color='red', linestyle='--', label=f'Mean Income: ${mean_income:.2f}')  # Dashed red line at mean

# Set plot title and axis labels
plt.title('Distribution of Annual Income among Aerofit Treadmill Buyers')  # Title of the plot
plt.xlabel('Annual Income')  # X-axis label
plt.ylabel('Density')  # Y-axis label

plt.grid(True)  # Display grid lines
plt.legend()  # Display the legend with the mean income label
plt.show()  # Render and display the plot
```



Insights from Distribution of Annual Income among Aerofit Treadmill Buyers

- **Average Buyer's Income:**
  On average, individuals purchasing Aerofit treadmills have an annual income of about **$53,719**.

- **Most Common Income Range:**
  Most buyers earn between $40,000 and **60,000** annually.

- **Income Distribution:**
  The graph reveals that more buyers earn **below the average income** level than above it.

- **Fewer High Earners:**
  A smaller proportion of buyers have incomes around $80,000 to$**90,000**.

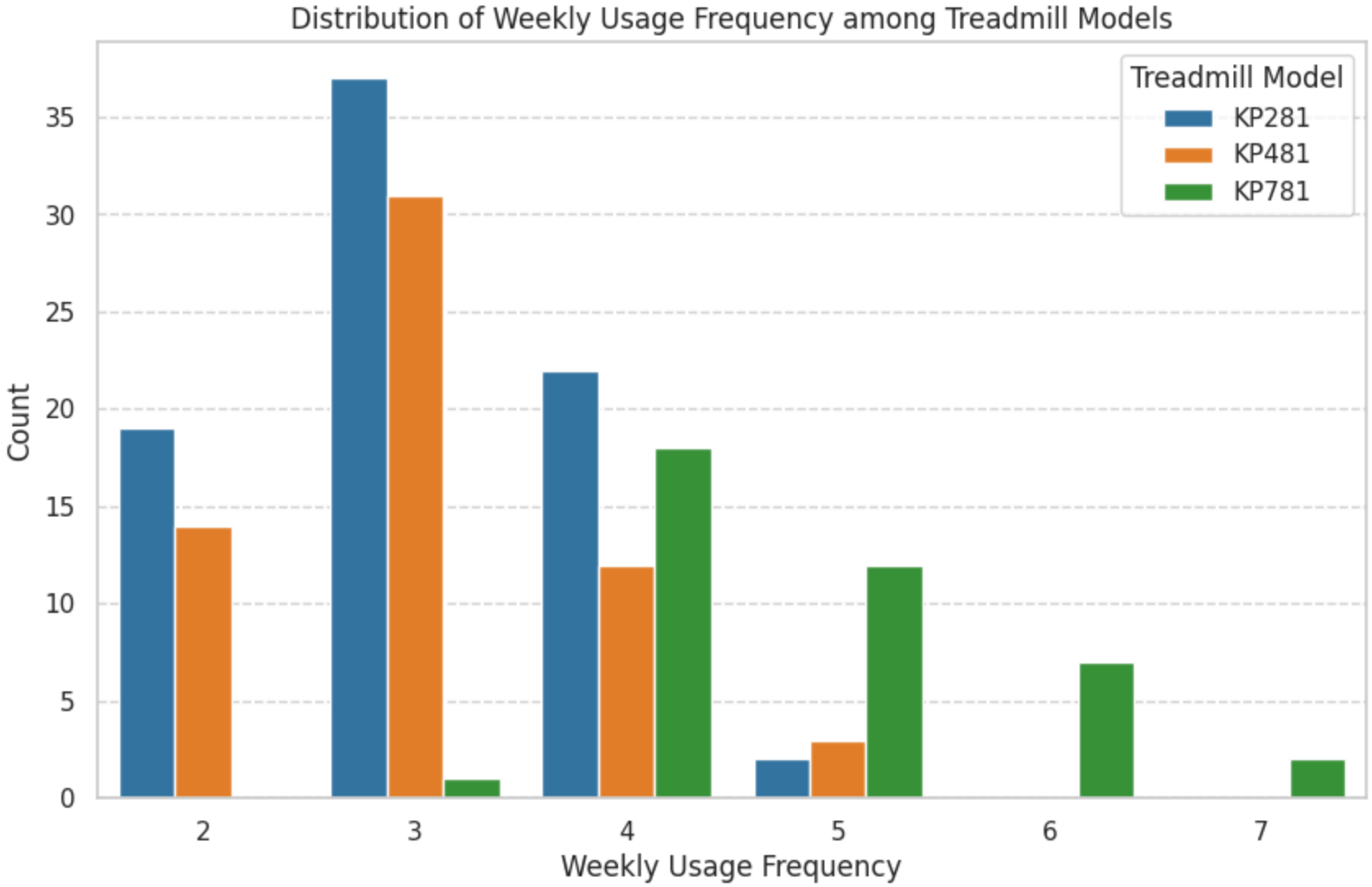
- **Very Low or High Incomes Are Uncommon:**
  It is rare for buyers to earn **less than $20,000** or **more than $100,000**.

This analysis suggests that Aerofit treadmills are most popular among **middle-income earners**, with fewer purchases from individuals at the very low or high ends of the income spectrum.

3.1.2 What is the distribution of weekly usage frequency among the different treadmill models?

In [136…

```
# Plotting the distribution of weekly usage frequency among treadmill models
plt.figure(figsize=(10, 6))  # Set the size of the figure
sns.countplot(data=df, x='Usage', hue='Product', palette=sns.color_palette("tab10"))  # Countplot with 'Usage' on x-axis and 'Prod

# Set plot title and axis labels
plt.title('Distribution of Weekly Usage Frequency among Treadmill Models')  # Add a title
plt.xlabel('Weekly Usage Frequency')  # Label for the x-axis
plt.ylabel('Count')  # Label for the y-axis

# Add a legend for the treadmill models
plt.legend(title='Treadmill Model')  # Add a legend with title

# Show the plot
plt.show()
```



Insights of Distribution of Weekly Usage Frequency among Treadmill Models

- **Most Popular Usage Frequency:**
  The majority of users across all models use their treadmills **3 times a week**, with the **KP281** model being the most frequently used at this frequency.

- **Model KP281:**
  This model is notably popular among users who use their treadmill **2 to 4 times per week**, suggesting it might be preferred by moderate users.

- **Model KP481:**
  Shows a relatively balanced usage across different frequencies, but it is particularly notable at **4 times per week**, indicating a preference among users who are slightly more active.

- **Model KP781:**
  This model stands out for **higher frequency usage (5-7 times per week)**, indicating it may be favored by the most active users or those in more intensive training regimes.

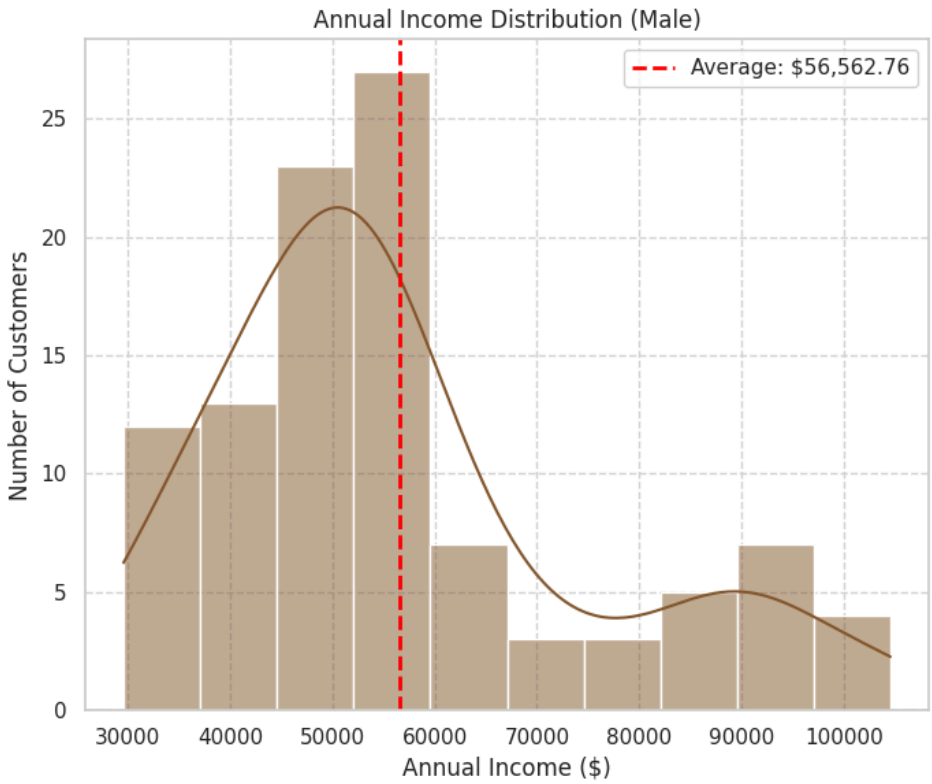
- **Least Frequent Usage:**
  The data shows very few users across all models using the treadmill **6 or 7 times per week**, with the **KP781** model being an exception at **6 times per week**, where it's relatively more common compared to the other models.

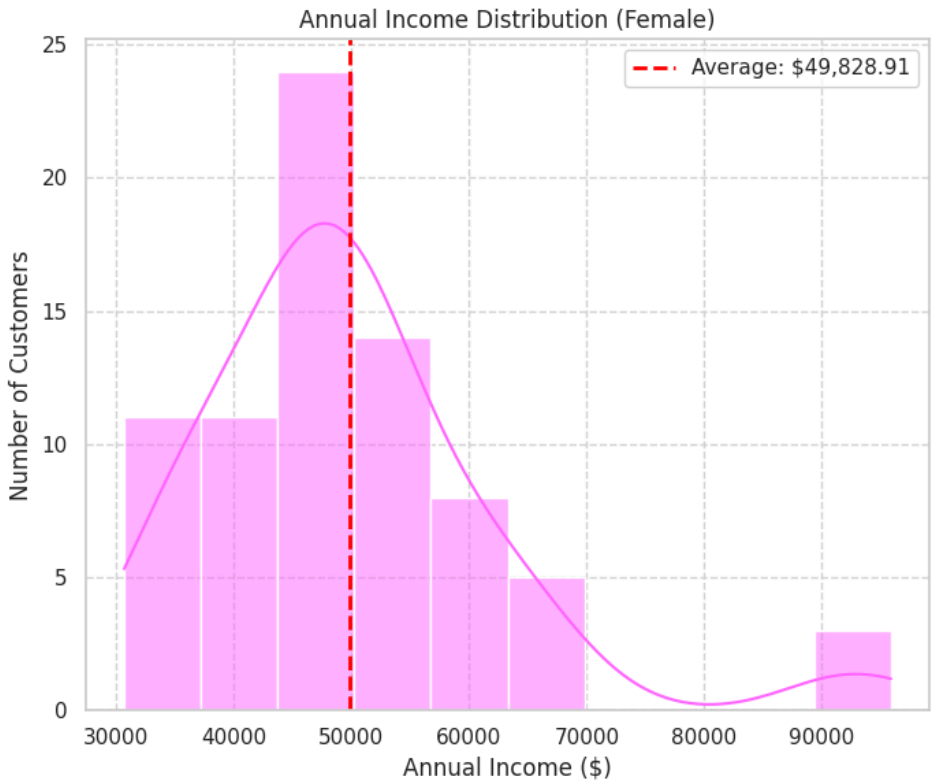
This analysis reveals distinct user preferences for treadmill models based on exercise frequency, guiding manufacturers in targeting specific market segments. The **KP781** attracts high-frequency users, while the **KP281** and **KP481** are preferred for moderate and regular routines.

3.1.3 What is the distribution of annual income among male and female customers separately?

In [137…

```
# Separate income data for male and female customers
male_data = df[df['Gender'] == 'Male']['Income']
female_data = df[df['Gender'] == 'Female']['Income']

# Create a figure with two subplots
fig, axes = plt.subplots(1, 2, figsize=(14, 6))

# Plot histogram for male customers using the first color from the palette
sns.histplot(male_data, bins=10, color="#86592d", kde=True, ax=axes[0])
axes[0].set_title('Annual Income Distribution (Male)')
axes[0].set_xlabel('Annual Income ($)')
axes[0].set_ylabel('Number of Customers')
axes[0].axvline(male_data.mean(), color='red', linestyle='dashed', linewidth=2, label=f'Average: ${male_data.mean():,.2f}')
axes[0].legend()

# Plot histogram for female customers using the second color from the palette
sns.histplot(female_data, bins=10, color="#ff66ff", kde=True, ax=axes[1])
axes[1].set_title('Annual Income Distribution (Female)')
axes[1].set_xlabel('Annual Income ($)')
axes[1].set_ylabel('Number of Customers')
axes[1].axvline(female_data.mean(), color='red', linestyle='dashed', linewidth=2, label=f'Average: ${female_data.mean():,.2f}')
axes[1].legend()

# Adjust layout and display the plots
plt.tight_layout()
plt.show()
```





Insights of Annual Income Distribution of Both Male and Female

- **Income Range:**
  Both male and female customers have a broad income range, spanning from approximately $30,000 to$ **90,000** annually.

- **Distribution Peaks:**
  Male income distribution peaks around **$50,000**, while females show the highest concentration slightly lower, at **$40,000**.

- **Skewness:**
  Male income distribution is **slightly right-skewed**, with a few males earning significantly above average. The female distribution also shows **right-skewness**, but with a more pronounced tail in the higher income range.

- **Average Income:**
  The mean income for males is **$56,562.76**, higher than females at **$49,828.91**, highlighting a **gender gap in earnings**.

- **Commonality and Differences:**
  Both genders cluster in the **middle-income bracket**, but males are more likely to reach **upper-income levels** than females.

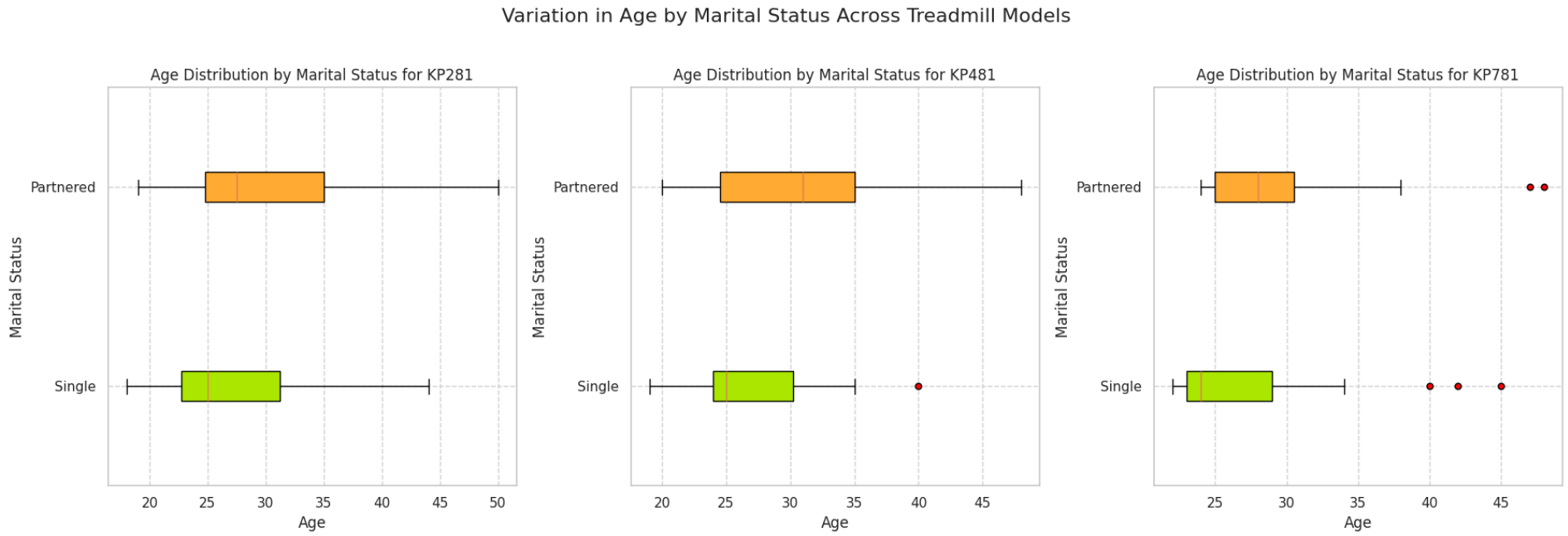
This graph highlights an **income disparity** between male and female customers, with males generally earning more and reaching higher income brackets. These insights can inform targeted marketing strategies and service offerings tailored to distinct financial profiles.

3.2 Multivariate Distribution Analysis: Age and Marital Status Across Models

3.2.1 What is the variation in the age of customers by their marital status across different treadmill models?

In [138…

```
# Define colors for each marital status
color_map = {
    'Single': '#ace600',
    'Partnered': '#ffad33',
}

# Define properties for the outlier points
flierprops = dict(marker='o', markerfacecolor='red', markersize=5, linestyle='none')

# Unique products
products = df['Product'].unique()

# Set up the figure and axes
fig, axes = plt.subplots(nrows=1, ncols=3, figsize=(18, 6))

# Generate boxplots for each product
for ax, product in zip(axes, products):
    # Prepare data for the boxplot
    subset = df[df['Product'] == product]
    grouped_data = subset.groupby('MaritalStatus')['Age'].apply(list).reindex(['Single', 'Partnered'])

    # Create the boxplot horizontally with outlier properties
    bplot = ax.boxplot(grouped_data, vert=False, patch_artist=True, labels=grouped_data.index,
                       showfliers=True, flierprops=flierprops)

    # Set properties for boxplots
    for patch, status in zip(bplot['boxes'], grouped_data.index):
        patch.set_facecolor(color_map[status])  # Specific color for each status

    # Add individual titles and set common Y-label
    ax.set_title(f'Age Distribution by Marital Status for {product}')
    ax.set_xlabel('Age')  # Set x-axis label to Age
    ax.set_ylabel('Marital Status')  # Set y-label to Marital Status

# Add a suptitle
fig.suptitle('Variation in Age by Marital Status Across Treadmill Models', fontsize=16, y=1.02)
print("\n")

plt.tight_layout()
plt.show()
```



Insights of Variation in Age by Marital Status Across Treadmill Models

- **Age Range Variation:**
  Each marital status group shows a distinct age range across treadmill models. For example, 'Single' users typically exhibit a **narrower age range**, indicating a more uniform age group preferring specific models.

- **Consistency Across Models:**
  The age distribution for 'Partnered' and 'Single' categories appears **consistent across the three models**, suggesting similar treadmill usage preferences among these groups regardless of the model.

- **Higher Age Median for Partnered:**
  In all models, the **'Partnered' group** has a slightly **higher median age** compared to the 'Single' group. This implies that partnered individuals may be more invested in maintaining fitness at a later stage in life.

- **Presence of Outliers:**
  The 'Single' category for **KP481** and **KP781** shows outliers, with a few significantly older individuals preferring these models. This suggests these treadmills have an appeal to a **broader age range**.

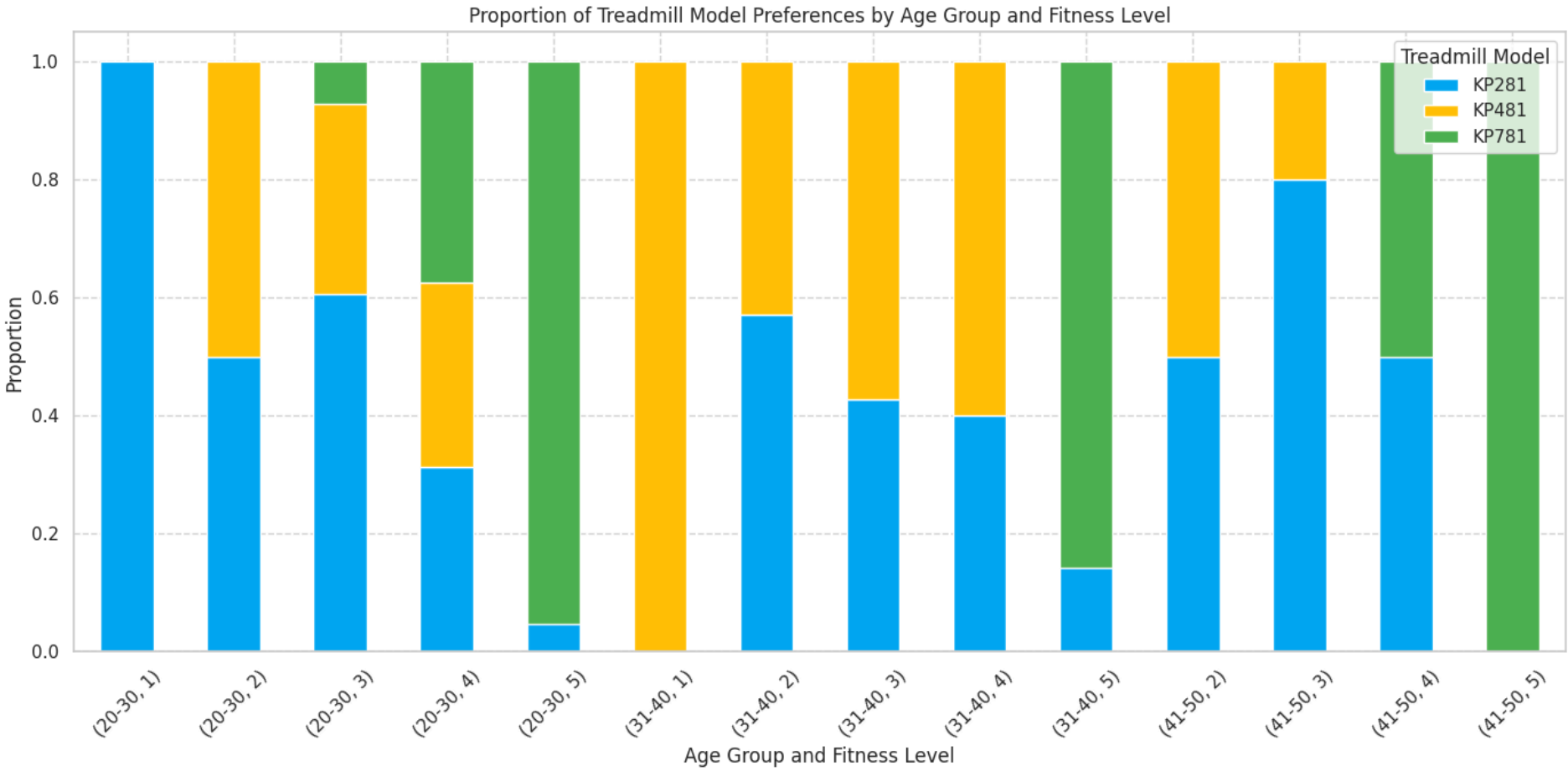
- **Age and Marital Status Correlation:**
  The consistent positioning of boxplots across treadmill models indicates a possible **correlation between marital status and age** in fitness equipment preferences. Partnered individuals may prefer **more robust or versatile models**, suitable for shared usage or more mature fitness needs.

These graphs reveal that **partnered users** are generally older, indicating a preference for treadmills suitable for mature and shared usage. The presence of outliers among **single users** in some models suggests these treadmills also attract a **broader age range**.

3.2.2 What proportion of each age group prefers each treadmill model, and how does this vary by fitness level?

In [139…

```
# Categorize ages into groups
df['Age Group'] = pd.cut(df['Age'], bins=[20, 30, 40, 50], labels=['20-30', '31-40', '41-50'])

# Crosstab to get counts of each model by age group and fitness
counts = pd.crosstab(index=[df['Age Group'], df['Fitness']], columns=df['Product'], margins=False)

# Normalize the counts to get proportions within each age and fitness group
proportions = counts.div(counts.sum(axis=1), axis=0)

# Plotting
plt.figure(figsize=(14, 7))
proportions.plot(kind='bar', stacked=True, color=['#03A9F4', '#FFC107', '#4CAF50'], ax=plt.gca())
plt.title('Proportion of Treadmill Model Preferences by Age Group and Fitness Level')
plt.xlabel('Age Group and Fitness Level')
plt.ylabel('Proportion')
plt.legend(title='Treadmill Model')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Insights of Proportion of Treadmill Model Preferences by Age Group and Fitness Level

- **Younger, Fitter Choices:**
  The **20-30 age group** with higher fitness levels (4 and 5) shows a strong preference for the **KP781 model**, indicating that younger, fitness-conscious users lean towards high-end options.

- **Broad Appeal of KP281:**
  The **KP281 model** remains consistently popular across all age groups, suggesting it is a versatile choice with wide demographic appeal.

- **Shift in Preferences with Age:**
  Older users, particularly those in the **41-50 age group** with fitness levels 3 to 5, show a shift towards the **KP481 and KP781 models**, highlighting a preference for advanced features or greater durability.

- **High Fitness, High-End Models:**
  In every age group, individuals with the **highest fitness ratings (level 5)** predominantly choose the **KP781**, reinforcing its appeal among serious exercisers.

- **Fitness Level Impact:**
  Within the same age group (e.g., **31-40**), variations in treadmill preferences by fitness level underline the **significant role of fitness** in influencing choice, beyond age alone.

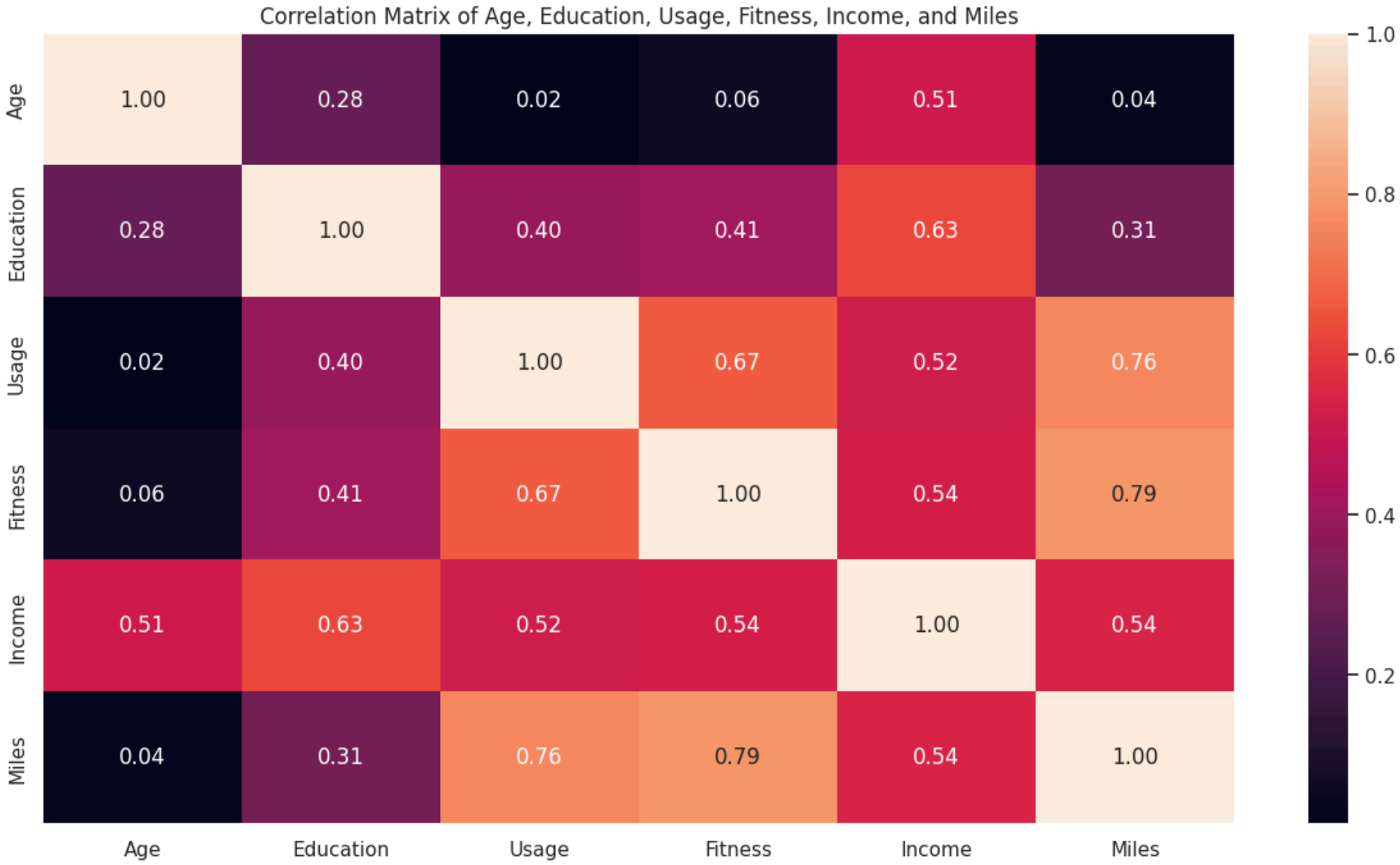
This graph shows that treadmill preferences differ by **age and fitness levels**, with younger, fitter individuals preferring high-end models. Observations like these can guide **marketing strategies** and **product development** tailored to customer needs.

3.3 Correlation and Interaction Insights: Relationships Across Metrics

3.3.1 What are the relationships between 'Age,' 'Education,' 'Usage,' 'Fitness,' 'Income,' and 'Miles' in terms of correlation?

In [140…

```
# Calculating the correlation matrix for selected columns
correlation_matrix = df[['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']].corr()

# Plotting the correlation matrix using a heatmap
plt.figure(figsize=(15, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='rocket', fmt=".2f")
plt.title('Correlation Matrix of Age, Education, Usage, Fitness, Income, and Miles')
plt.show()
```



Insights of Correlation Matrix of Age, Education, Usage, Fitness, Income, and Miles

- **Income and Education:**
  There's a **strong positive correlation (0.63)** between income and education, indicating that higher educational attainment is associated with higher income levels.

- **Usage and Miles:**
  A **very strong correlation (0.76)** exists between usage and miles, showing that customers who use their equipment more frequently tend to log more miles.

- **Fitness and Usage:**
  Fitness and usage have a **significant positive correlation (0.67)**, suggesting that frequent use of fitness equipment contributes to higher fitness levels.

- **Income and Age:**
  There's a **moderate positive correlation (0.51)** between income and age, implying that income tends to increase as customers grow older.

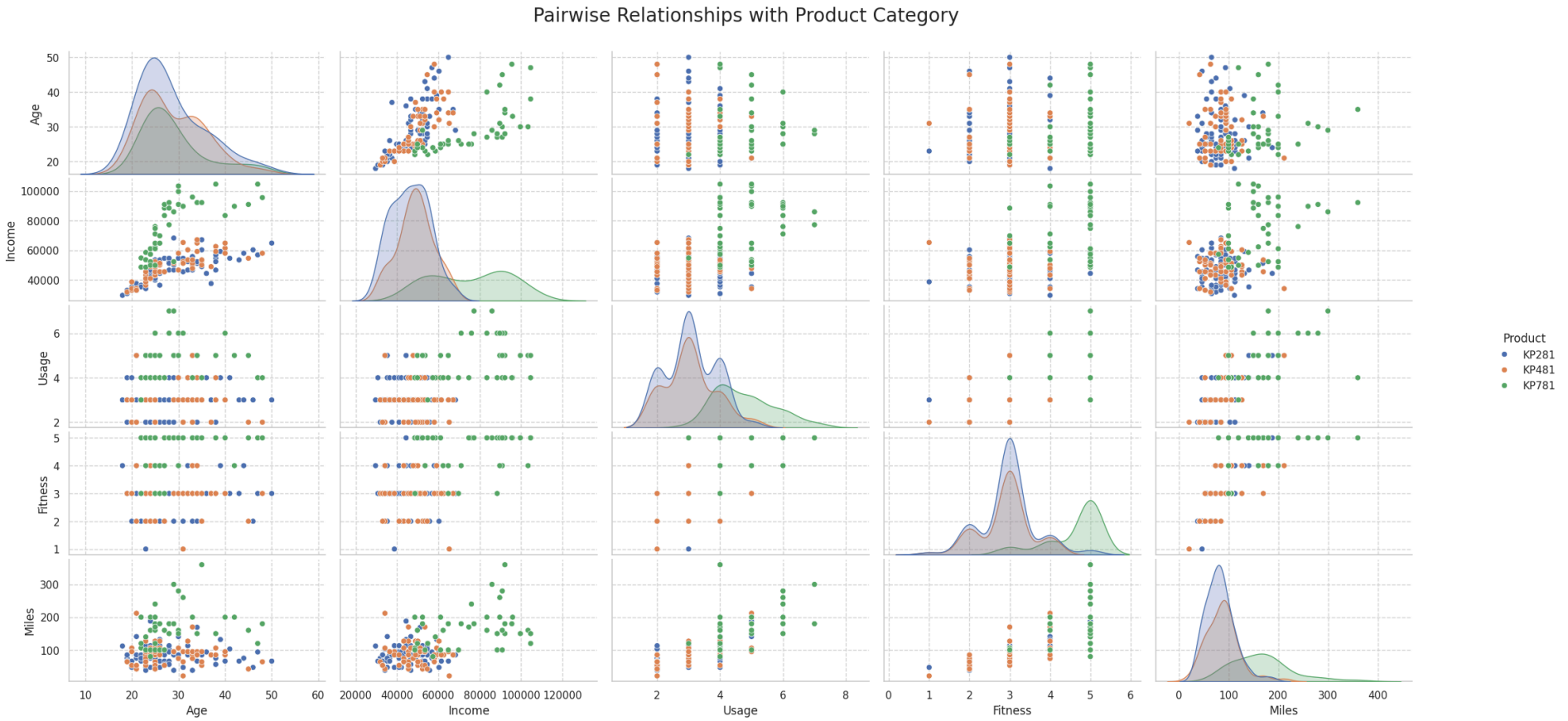
- **Fitness and Miles:**
  A **strong correlation (0.79)** is observed between fitness and miles, highlighting that customers with higher fitness levels generally cover more miles.

The correlation analysis shows that higher education is linked to higher income, while frequent fitness equipment usage correlates with improved fitness levels and greater miles covered, offering key insights for targeted marketing and product improvements.

3.3.2 How do the variables 'Age', 'Income', 'Usage', 'Fitness', and 'Miles' interact across different products, and what patterns can be observed in their relationships when segmented by product type?

In [141…

```
# Generate a pairplot to visualize pairwise relationships among the variables.
pairplot = sns.pairplot(df, hue='Product', diag_kind='kde', vars=['Age', 'Income', 'Usage', 'Fitness', 'Miles'])

# Set the title of the pairplot
pairplot.fig.suptitle('Pairwise Relationships with Product Category', size=20, y=1.05);
print("\n")

# Overall figure size
pairplot.fig.set_size_inches(24, 10)
# Show the plot
plt.show()
```



Insights of Pairwise Relationships with Product Category

- **Age Distribution:**
  KP781 attracts slightly older users, while KP281 and KP481 are more popular among younger demographics, showing varied age preferences across models.

- **Income Trends:**
  KP481 users tend to have higher income levels, positioning it as a premium product, while KP281 and KP781 have broader income spreads, appealing to a wider audience.

- **Usage and Fitness Correlation:**
  Higher usage rates are correlated with better fitness levels across all models, with KP781 showing the strongest link, making it ideal for fitness enthusiasts.

- **Miles Coverage:**
  KP781 supports higher mileage, favored by serious fitness users, while KP281 and KP481 attract users with moderate mileage, suggesting casual use.

- **Product-Specific Patterns:**
  KP481 attracts higher-income users with moderate fitness goals, while KP281 appeals broadly but to users with lower fitness levels, showing distinct user clusters.

The analysis highlights **KP781** as preferred by active, older users, while **KP481** attracts wealthier, moderately active individuals. These patterns can guide targeted marketing and product enhancements for each demographic.

B) Probability Analysis

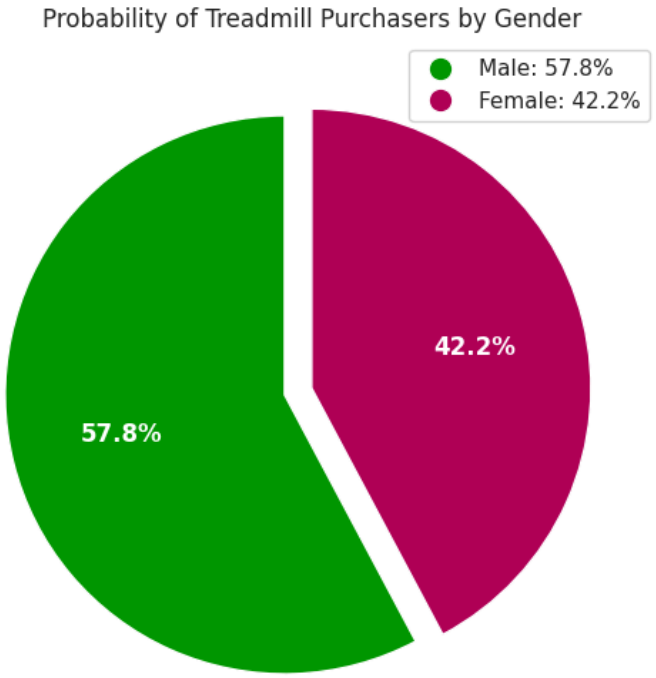
3.4 Representing Marginal Probability

3.4.1 What is the overall probability that a treadmill purchaser is male?

In [142…

```
# Count the number of males and females
gender_counts = df['Gender'].value_counts()

# Calculate the proportion of each gender for the pie chart
gender_proportion = gender_counts / gender_counts.sum()

# Set up the figure and axes for two subplots
fig, axes = plt.subplots(1, 2, figsize=(14, 6))

# Bar chart for counts on the left
axes[0].bar(gender_counts.index, gender_counts.values, color=['#009900', '#b30059'])
axes[0].set_title("Count of Male vs. Female Treadmill Purchasers")
axes[0].set_ylabel('Count')
axes[0].set_xlabel('Gender')

# Pie chart for probability on the right
colors = ['#009900', '#b30059']
explode = [0.1 if label == 'Male' else 0 for label in gender_counts.index]  # Explode the 'Male' segment
wedges, texts, autotexts = axes[1].pie(gender_proportion, labels=gender_proportion.index, autopct='%1.1f%%', startangle=90, colors
axes[1].set_title("Probability of Treadmill Purchasers by Gender")

# Highlight the 'Male' segment with a custom legend
custom_legend = [plt.Line2D([0], [0], marker='o', color='w', label='Male: {:.1f}%'.format(gender_proportion['Male']*100),
                            markerfacecolor='#009900', markersize=12),
                 plt.Line2D([0], [0], marker='o', color='w', label='Female: {:.1f}%'.format(gender_proportion['Female']*100),
                            markerfacecolor='#b30059', markersize=12)]
axes[1].legend(handles=custom_legend, loc='upper right')

# Show the plots
```

```
plt.tight_layout()
plt.show()
```



Insights from the Treadmill Purchasers by Gender

- **Gender Distribution in Counts:**
  The bar chart shows that more **males** purchase treadmills than **females**, with males comprising a larger share of total purchases.
- **Proportional Representation:**
  The pie chart reveals that **57.8% of treadmill purchasers are male**, indicating a majority presence without excessive dominance.
- **Female Representation:**
  Females account for **42.2% of purchasers**, highlighting a significant portion of the market despite males buying more treadmills.
- **Color Coding:**
  The use of **green for males** and **pink for females** in the charts allows for quick and easy visual interpretation of the data.
- **Market Potential:**
  The balanced representation of male and female purchasers suggests strong market potential for **gender-specific treadmill marketing**

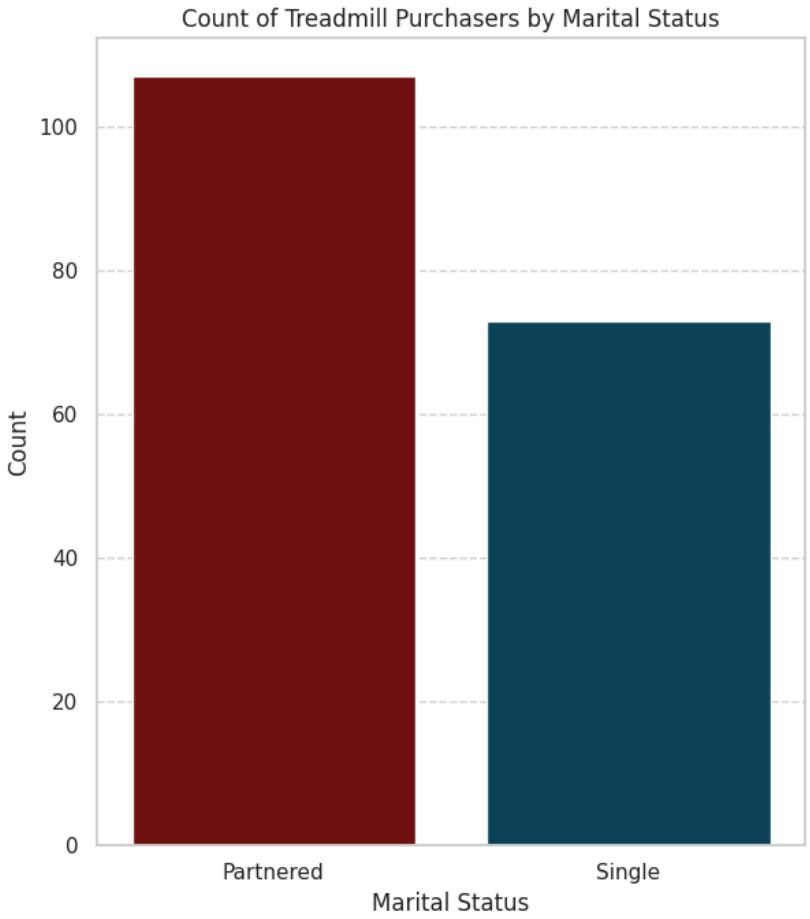
strategies.

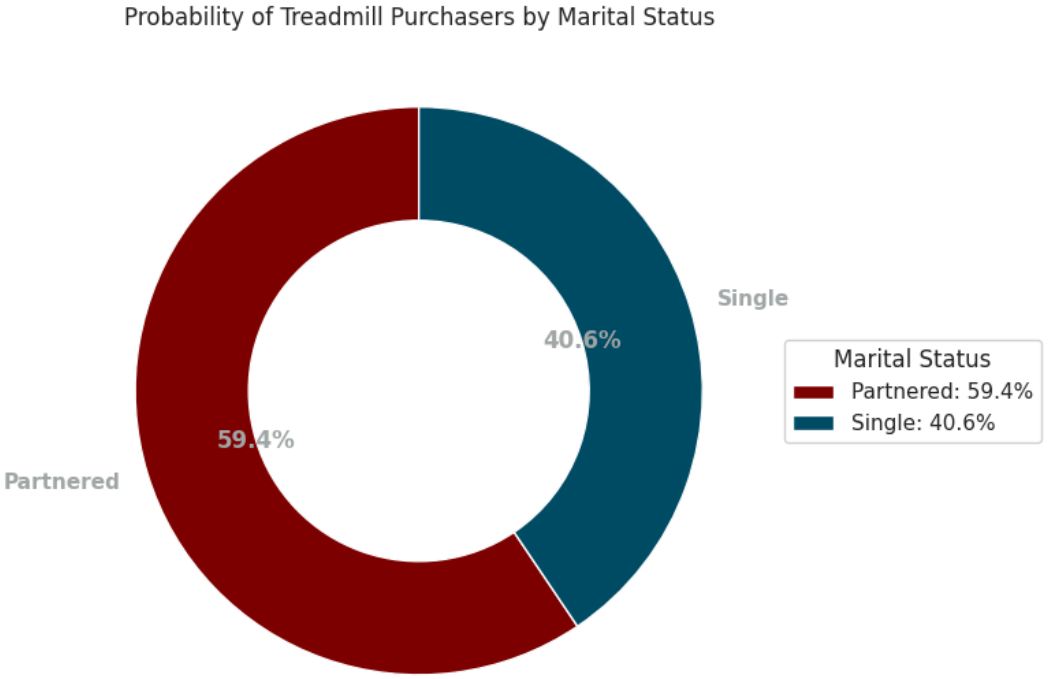
The charts reveal that males dominate treadmill purchases at **57.8%**, while females represent a substantial 42.2%. This insight emphasizes the importance of tailored marketing strategies to appeal effectively to both genders.

3.4.2 What is the probability that a customer purchasing a treadmill is married?

In [143…

```
# Count the occurrences of each marital status
marital_counts = df['MaritalStatus'].value_counts()

# Calculate the proportion of each marital status
marital_proportion = marital_counts / marital_counts.sum()

# Set up the figure and axes for two subplots
fig, axes = plt.subplots(1, 2, figsize=(14, 7))

# Create a bar chart for counts using Seaborn on the left
sns.barplot(x=marital_counts.index, y=marital_counts.values, ax=axes[0], palette=["#800000", "#004b66"])
axes[0].set_title("Count of Treadmill Purchasers by Marital Status")
axes[0].set_ylabel('Count')
axes[0].set_xlabel('Marital Status')

# Create a donut chart for probability on the right using Matplotlib
colors = sns.color_palette(["#800000", "#004b66"])[0:len(marital_counts)]
wedges, texts, autotexts = axes[1].pie(marital_proportion, labels=marital_proportion.index, autopct='%1.1f%%', startangle=90, colo
axes[1].set_title("Probability of Treadmill Purchasers by Marital Status")

# Prepare legend labels with proportion values
legend_labels = ['{}: {:.1f}%'.format(i, p*100) for i, p in zip(marital_proportion.index, marital_proportion)]

# Add a legend with the labels outside the donut chart
axes[1].legend(wedges, legend_labels,
               title="Marital Status",
               loc="center left",
               bbox_to_anchor=(1, 0, 0.5, 1))

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





Insights from the Treadmill Purchasers by Marital Status

- **Majority Married:**
  The donut chart shows that **59.4% of treadmill purchasers are partnered or married**, making this group the majority.
- **Significant Single Representation:**
  A substantial **40.6% of purchasers** are single, highlighting diversity in marital status among treadmill buyers.
- **Visual Comparison:**
  The bar chart complements the donut chart by visually showing that partnered purchasers outnumber single purchasers.
- **Color Coding:**
  Red for partnered and **blue for single** ensures clear differentiation between the categories across both charts.
- **Distribution Clarity:**
  Together, the charts provide a **clear and quick understanding** of marital status distribution among treadmill buyers.

The charts reveal that **60% of treadmill buyers are married or partnered**, suggesting they are more likely to buy treadmills, possibly for shared or family health goals.

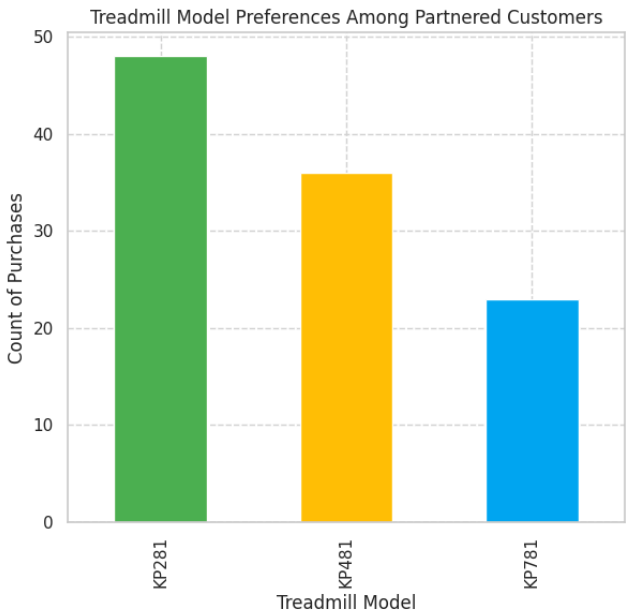
3.5 Constructing Two-Way Contingency Tables

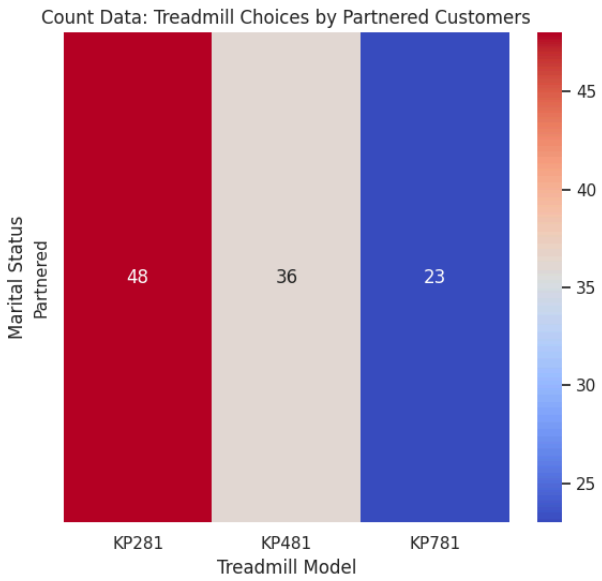
3.5.1 How frequently do partnered customers choose each model of treadmill, and what are the probabilities associated with these choices?

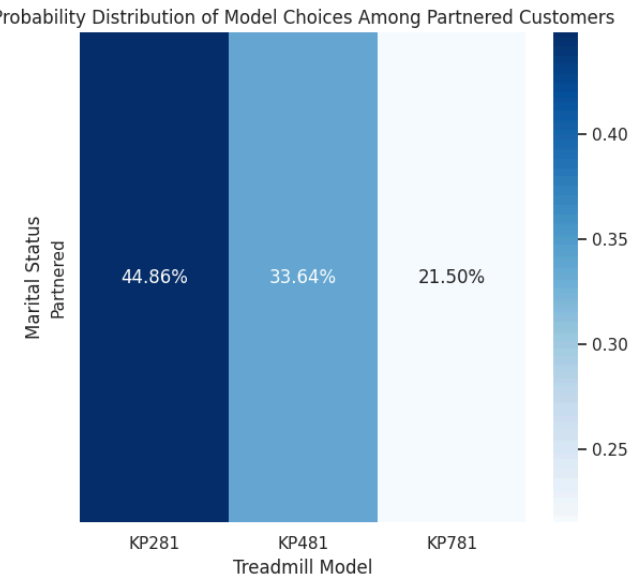
In [144…

```
# Filter data to focus only on partnered customers for this analysis
partnered_df = df[df['MaritalStatus'] == 'Partnered']

# Creating a contingency table of counts for how partnered customers choose each treadmill model
contingency_counts = pd.crosstab(partnered_df['MaritalStatus'], partnered_df['Product'])

# Creating a normalized contingency table to understand the probability of each model choice among partnered customers
contingency_probabilities = pd.crosstab(partnered_df['MaritalStatus'], partnered_df['Product'], normalize='index')

# Print the contingency tables for quick reference and validation
print("Contingency Table - Counts:\n", contingency_counts)
print("\nContingency Table - Probabilities:\n", contingency_probabilities.applymap(lambda x: f"{x:.2%}"))

# Set up the figure and axes for visualizations
fig, ax = plt.subplots(1, 3, figsize=(18, 6))

# Bar chart visualizing the count of purchases per model by partnered customers
contingency_counts.loc['Partnered'].plot(kind='bar', ax=ax[0], color=['#4CAF50', '#FFC107', '#03A9F4'])
ax[0].set_title('Treadmill Model Preferences Among Partnered Customers')
ax[0].set_ylabel('Count of Purchases')
ax[0].set_xlabel('Treadmill Model')

# Heatmap displaying the raw count data, providing a visual representation of the table data
sns.heatmap(contingency_counts, annot=True, cmap='coolwarm', fmt="d", ax=ax[1])
ax[1].set_title('Count Data: Treadmill Choices by Partnered Customers')
ax[1].set_xlabel('Treadmill Model')
ax[1].set_ylabel('Marital Status')

# Heatmap displaying the probabilities, which helps to visualize the relative preferences for treadmill models
sns.heatmap(contingency_probabilities, annot=True, cmap='Blues', fmt=".2%", ax=ax[2])
ax[2].set_title('Probability Distribution of Model Choices Among Partnered Customers')
ax[2].set_xlabel('Treadmill Model')
ax[2].set_ylabel('Marital Status')

plt.tight_layout()
plt.show()
```

```
Contingency Table - Counts:
 Product        KP281  KP481  KP781
MaritalStatus
Partnered         48     36     23

Contingency Table - Probabilities:
 Product         KP281    KP481    KP781
MaritalStatus
Partnered      44.86%  33.64%  21.50%
```







Insights of Treadmill Model Preferences Among Partnered Customers

- **Model KP281 Preference:**
  Partnered customers show a **strong preference for the KP281 model**, with nearly half of the purchases (48 times), highlighting its popularity among this demographic.

- **Declining Preference for Higher Models:**
  Preferences decline as model numbers increase, with KP481 being less favored (**36 times**) and KP781 the least preferred (**23 times**) among partnered customers.

- **Quantified Probabilities:**
  KP281 is chosen by **44.86%** of partnered customers, followed by KP481 at **33.64%**, and KP781 at **21.50%**, showing a clear preference distribution.

- **Visual Appeal and Clarity:**
  The **bar chart and heatmap** together provide a clear comparison of raw counts and probabilities, aiding quick and easy understanding of the

data.

- **Detailed Counts and Proportions:**
  The count data underscores the actual demand distribution: KP281 (**48 times**), KP481 (**36 times**), and KP781 (**23 times**) among partnered customers.

The analysis reveals a **strong preference for KP281** among partnered customers, with **44.86% choosing it**. This insight suggests that targeted marketing efforts for KP281 could better serve this demographic and optimize sales.

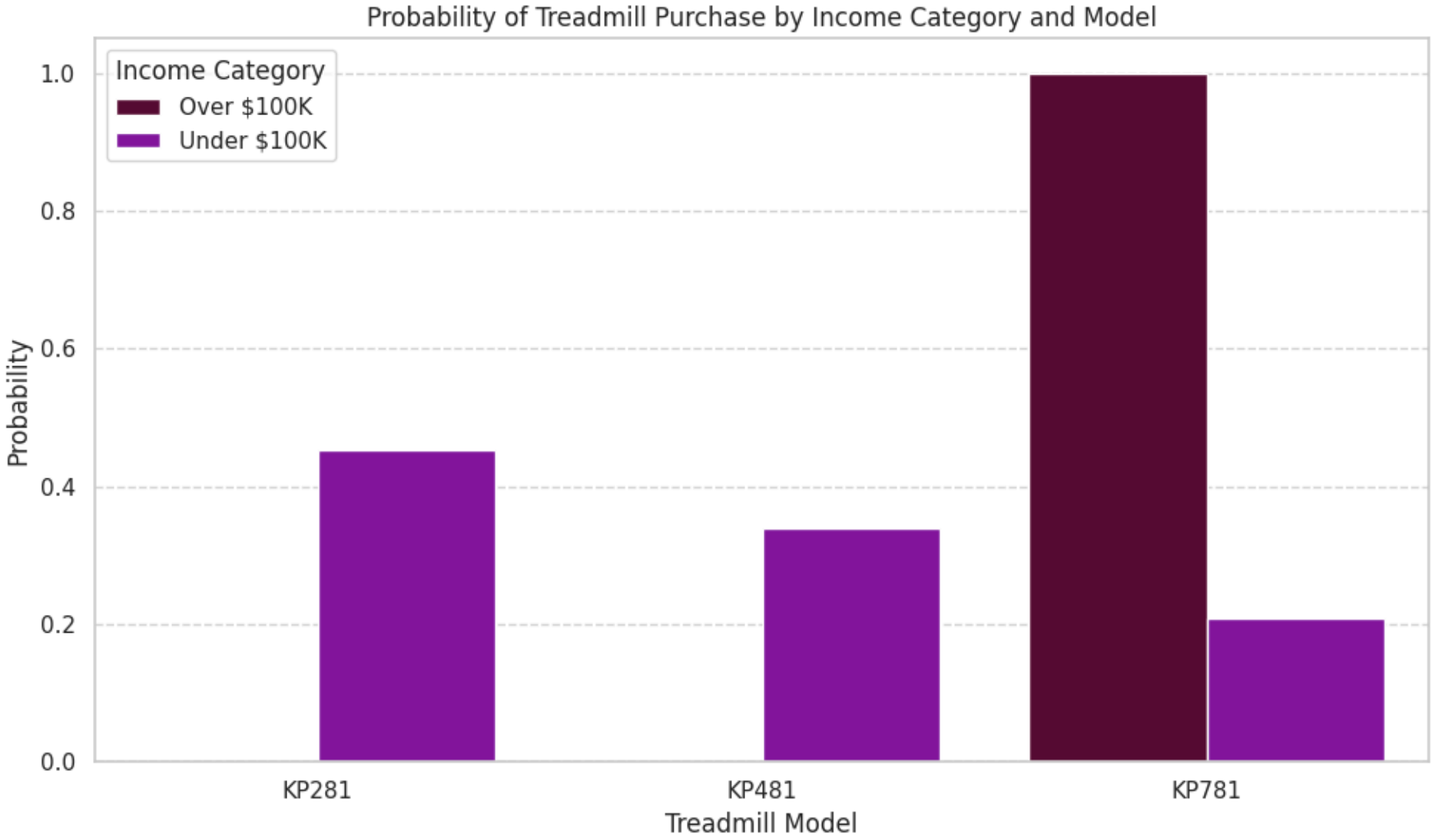
3.6 Computing Conditional Probabilities

3.6.1 If a customer has an annual income of over $100,000, what is the probability they will purchase the high-end KP781 treadmill?

In [145…

```
# Creating an income category based on the threshold of $100,000
df['Income Category'] = df['Income'].apply(lambda x: 'Over $100K' if x > 100000 else 'Under $100K')

# Crosstab to calculate the counts of each model by income category
model_income_crosstab = pd.crosstab(df['Income Category'], df['Product'])

# Convert counts to probabilities within each income category
model_income_prob = model_income_crosstab.div(model_income_crosstab.sum(axis=1), axis=0)

# Print the probability table
print("Conditional Probability Table:")
print(model_income_prob.applymap(lambda x: f"{x:.2%}"))

# Plotting the data
plt.figure(figsize=(10, 6))
custom_palette = ['#660033', '#8f00b3']
sns.barplot(data=model_income_prob.reset_index().melt(id_vars='Income Category'), x='Product', y='value', hue='Income Category', p
plt.title('Probability of Treadmill Purchase by Income Category and Model')
plt.ylabel('Probability')
plt.xlabel('Treadmill Model')
plt.legend(title='Income Category')
plt.tight_layout()
plt.show()
```

```
Conditional Probability Table:
Product           KP281   KP481    KP781
Income Category
Over $100K         0.00%   0.00%  100.00%
Under $100K       45.20%  33.90%   20.90%
```



Insights of Treadmill Preferences by Income Category

- **Exclusive Preference for Luxury:**
  Customers earning over **$100,000** exclusively purchase the KP781 treadmill, making it the preferred choice for **100%** of the high-income category.

- **Diverse Preferences for Lower Incomes:**
  Customers with incomes under **$100,000** exhibit varied preferences, choosing KP281 (**45.2%**), KP481 (**33.9%**), and KP781 (**20.9%**).

- **High-End Appeal:**
  The KP781's popularity among wealthier customers underscores its **luxury status**, likely due to advanced features or the prestige associated

with its price.

- **Accessible Options:**
  KP281 and KP481 are more accessible to lower-income customers, indicating their positioning as **budget-friendly models**.

- **Marketing Focus:**
  There is a clear opportunity for **targeted marketing strategies**, promoting KP781 to affluent buyers while positioning KP281 and KP481 for a broader, cost-conscious audience.

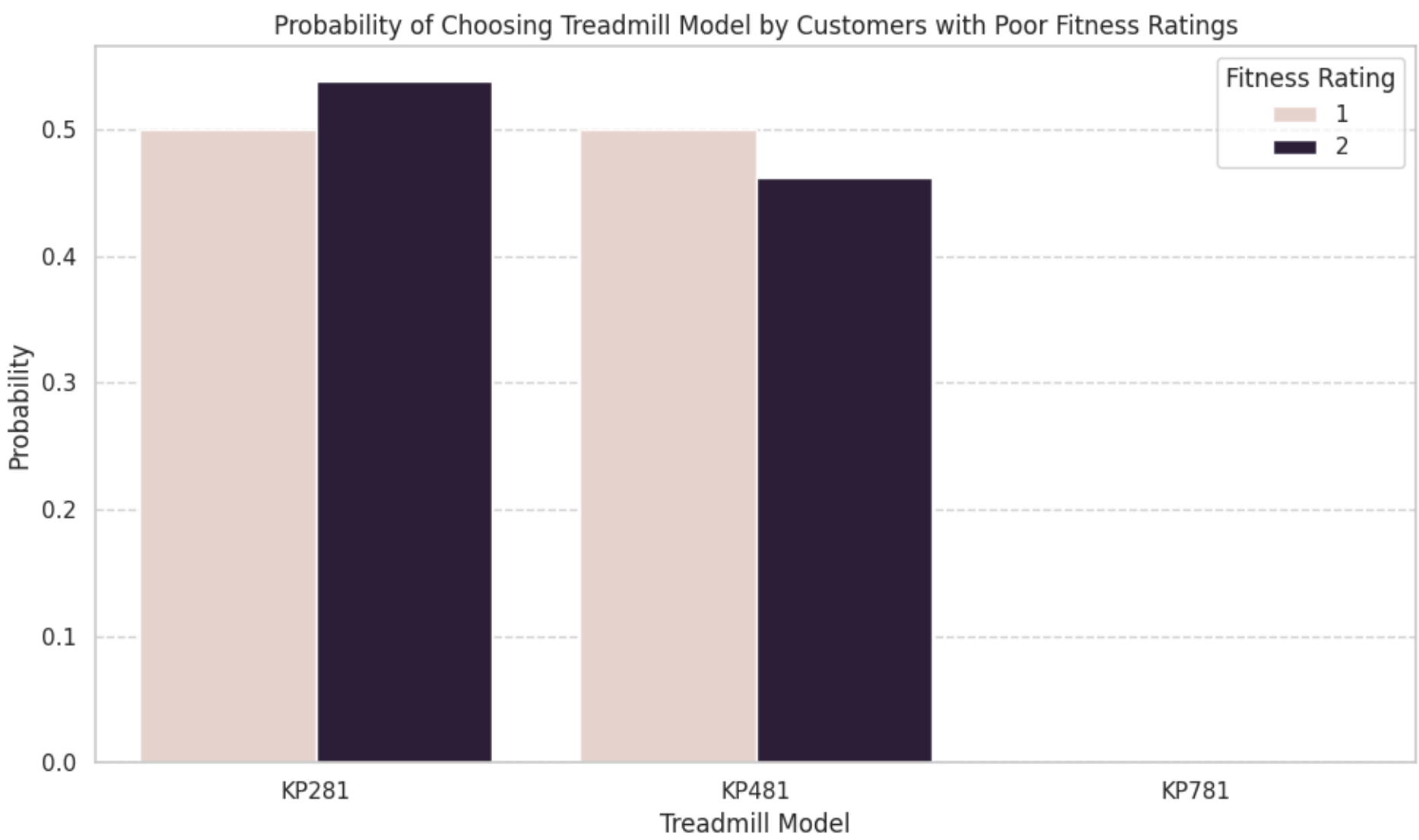
This graph indicates that the exclusive preference for KP781 among customers earning over **$100,000** highlights its status as a **premium option**. Focused marketing efforts could effectively capture this affluent segment, while KP281 and KP481 cater to the majority of the market.

3.6.2 If a customer rates their overall fitness as poor, what is the probability they will choose the KP281 treadmill, considering it's more budget-friendly?

In [146…

```
# Filter data for poor fitness ratings (assuming 'poor' is represented by 1 and 2)
poor_fitness_df = df[df['Fitness'] <= 2]

# Creating a contingency table of counts for poor fitness ratings
contingency_counts = pd.crosstab(poor_fitness_df['Product'], poor_fitness_df['Fitness'])

# Convert counts to probabilities within the 'poor' fitness category
contingency_probabilities = contingency_counts.div(contingency_counts.sum(axis=0), axis=1)

# Print the probability table
print("Conditional Probability Table for Customers with Poor Fitness Ratings:\n", contingency_probabilities.applymap(lambda x: f"{

# Plotting the probabilities using Seaborn
plt.figure(figsize=(10, 6))
sns.barplot(data=contingency_probabilities.reset_index().melt(id_vars='Product'), x='Product', y='value', hue='Fitness')
plt.title('Probability of Choosing Treadmill Model by Customers with Poor Fitness Ratings')
plt.ylabel('Probability')
plt.xlabel('Treadmill Model')
plt.legend(title='Fitness Rating')
plt.tight_layout()
plt.show()
```

```
Conditional Probability Table for Customers with Poor Fitness Ratings:
 Fitness       1       2
Product
KP281    50.00%  53.85%
KP481    50.00%  46.15%
```



Insights of Probability of Choosing Treadmill Model by Customers with Poor Fitness Ratings

- **Equal Preference at Lower Fitness:**
  Customers with a fitness rating of '1' (Very Poor) show an equal probability (**50%**) of choosing either the KP281 or KP481, indicating both models are equally accessible for this group.

- **Slight Preference Increase with KP281:**
  For customers with a fitness rating of '2' (Poor), the probability of choosing KP281 increases slightly to **53.85%**, reflecting a marginal preference for this budget-friendly model.

- **Lower Interest in KP481 for Poor Fitness:**
  The probability of choosing KP481 decreases to **46.15%** for fitness rating '2', suggesting a shift toward KP281 as fitness concerns slightly improve.

- **Absence of KP781 Choices:**
  No customers with poor fitness ratings choose the KP781 treadmill, likely due to its **advanced features or higher price**, making it unsuitable for this group.

- **Balanced Decision Making:**
  The data indicates that customers with poor fitness make **balanced decisions**, prioritizing cost and suitability in their treadmill choices.

The data underscores the **KP281 treadmill** as a favored choice among customers with poor fitness due to its **affordability and fundamental features**. Marketing efforts should focus on highlighting its **cost-effectiveness and basic functionality** to better target and serve this demographic.

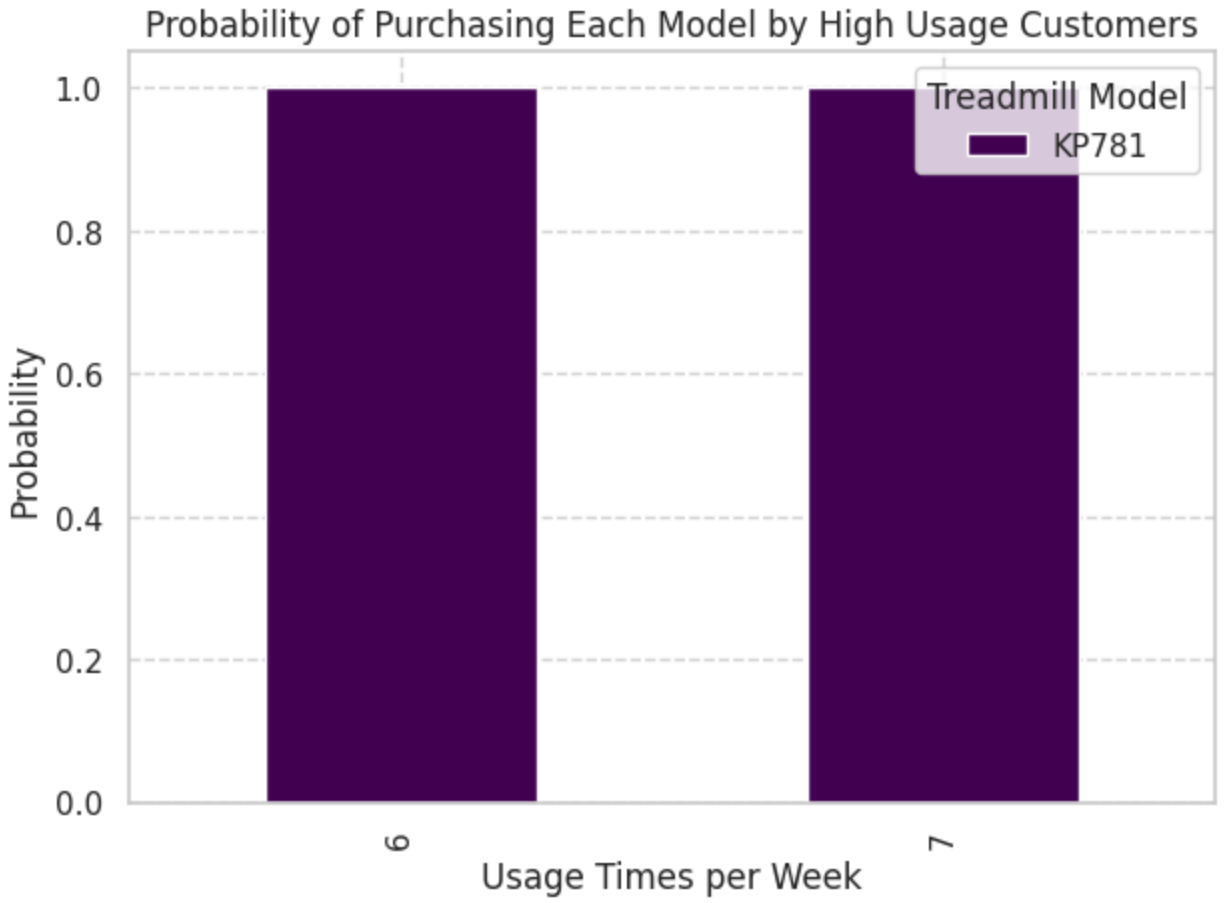
3.7 Customer Profiling - Categorization of Users

3.7.1 Among customers who use the treadmill more than five times a week, what is the probability that they purchased the high-end KP781 model?

In [147…

```
# Filter for high usage (more than 5 times a week)
high_usage_df = df[df['Usage'] > 5]

# Crosstab to calculate the counts of treadmill model choices among high usage customers
high_usage_crosstab = pd.crosstab(high_usage_df['Usage'], high_usage_df['Product'])

# Convert counts to probabilities within the high usage category
high_usage_probabilities = high_usage_crosstab.div(high_usage_crosstab.sum(axis=1), axis=0)

# Print the probability table
print("Probability Table for High Usage Customers:\n", high_usage_probabilities.applymap(lambda x: f"{x:.2%}"))

# Plotting the probabilities using a bar chart
high_usage_probabilities.plot(kind='bar', stacked=True, colormap='viridis')
plt.title('Probability of Purchasing Each Model by High Usage Customers')
plt.ylabel('Probability')
plt.xlabel('Usage Times per Week')
plt.legend(title='Treadmill Model')
plt.tight_layout()
plt.show()
```

```
Probability Table for High Usage Customers:
 Product    KP781
Usage
6         100.00%
7         100.00%
```



Insights of Probability of Purchasing Each Model by High Usage Customers

- **Exclusive Preference:**
  High-usage customers exclusively purchase the **KP781 model**, with no selections of other models, demonstrating strong loyalty to this high-end option.

- **Uniformity Across Frequencies:**
  Both usage categories, **6 and 7 times per week**, show a **100% probability** of choosing KP781, highlighting its consistent appeal to frequent users.

- **Indicative of Quality:**
  The strong preference for KP781 among high-frequency users suggests it offers **features and durability** highly valued by those who use

treadmills intensively.

- **Target Audience:**
  Marketing strategies could focus on this segment, emphasizing KP781's **advanced features and durability** to attract more high-usage customers.

- **No Diversification:**
  The lack of other model selections by this group suggests a **niche market** highly satisfied with the KP781.

This graph indicates that all **high-usage treadmill customers** exclusively select the KP781 model, showcasing its appeal to frequent users. This insight is crucial for developing **marketing strategies** and **product offerings** tailored to active users.

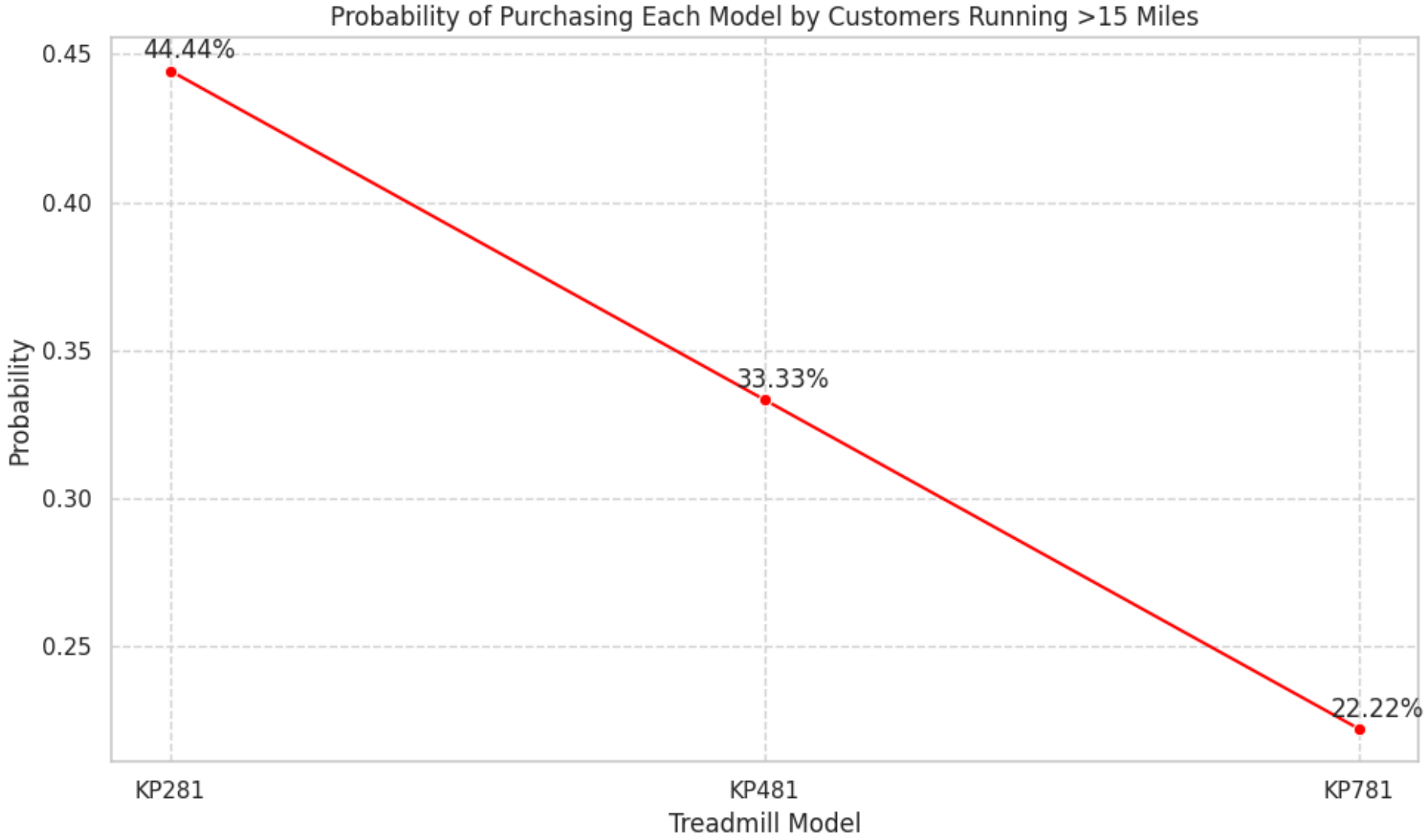
3.7.2 For customers expecting to run more than 15 miles per week, what treadmill model do they most likely purchase?

In [148…

```
# Filter data for those expecting to run more than 15 miles per week
high_mileage_df = df[df['Miles'] > 15]

# Create a crosstab of the counts of treadmill models purchased by these high-mileage customers
model_counts = pd.crosstab(index=high_mileage_df['Product'], columns='Count')

# Convert counts to probability
model_probability = model_counts.div(model_counts.sum())

# Print results
print("Counts of Treadmill Models for High Mileage Runners:")
print(model_counts)
print("\nProbability of Choosing Each Model for High Mileage Runners:")
print(model_probability.applymap(lambda x: f"{x:.2%}"))

# Reset index for plotting
model_probability_reset = model_probability.reset_index().melt(id_vars='Product', var_name='Variable', value_name='Probability')

# Visualize the probability with a line chart using seaborn
plt.figure(figsize=(10, 6))
lineplot = sns.lineplot(data=model_probability_reset, x='Product', y='Probability', marker='o', color='red')
plt.title('Probability of Purchasing Each Model by Customers Running >15 Miles')
plt.xlabel('Treadmill Model')
plt.ylabel('Probability')
plt.grid(True)
plt.xticks(rotation=0)

# Adding text labels for probability percentages
for x, y in zip(model_probability_reset['Product'], model_probability_reset['Probability']):
    plt.annotate(f'{y:.2%}', xy=(x, y), textcoords="offset points", xytext=(9,6), ha='center')
```

```
plt.tight_layout()
plt.show()
```

```
Counts of Treadmill Models for High Mileage Runners:
col_0    Count
Product
KP281       80
KP481       60
KP781       40

Probability of Choosing Each Model for High Mileage Runners:
col_0     Count
Product
KP281    44.44%
KP481    33.33%
KP781    22.22%
```



Insights of Probability of Purchasing Each Model by Customers Running >15 Miles

- **Model Preference Gradient:**
  The **KP281 model** is the most preferred treadmill, with **44.44% of high-mileage runners** choosing it, indicating its suitability for regular runners.

- **Declining Preference with Price:**
  A clear trend shows that as the treadmill model becomes more advanced and likely more expensive (KP481 to KP781), its probability of being chosen decreases.

- **Lowest Preference for High-End Model:**
  The **KP781 model**, despite being the high-end option, is the least preferred among high-mileage runners, chosen by only **22.22%**.

- **Balanced Needs and Costs:**
  High-mileage runners tend to prefer treadmills that **balance functionality and cost**, avoiding the most expensive models.

- **Gradual Decrease:**
  The decrease in preference from KP281 to KP781 reflects how factors like **cost or tailored features** for less intensive use may deter high-mileage runners from higher-end models.

This graph indicates that **high-mileage treadmill users** predominantly choose the KP281 model, favoring its balance of essential features and affordability. This reflects a preference for **practicality over luxury** in high-usage scenarios.

3.8 Probability - Marginal and Conditional Probability

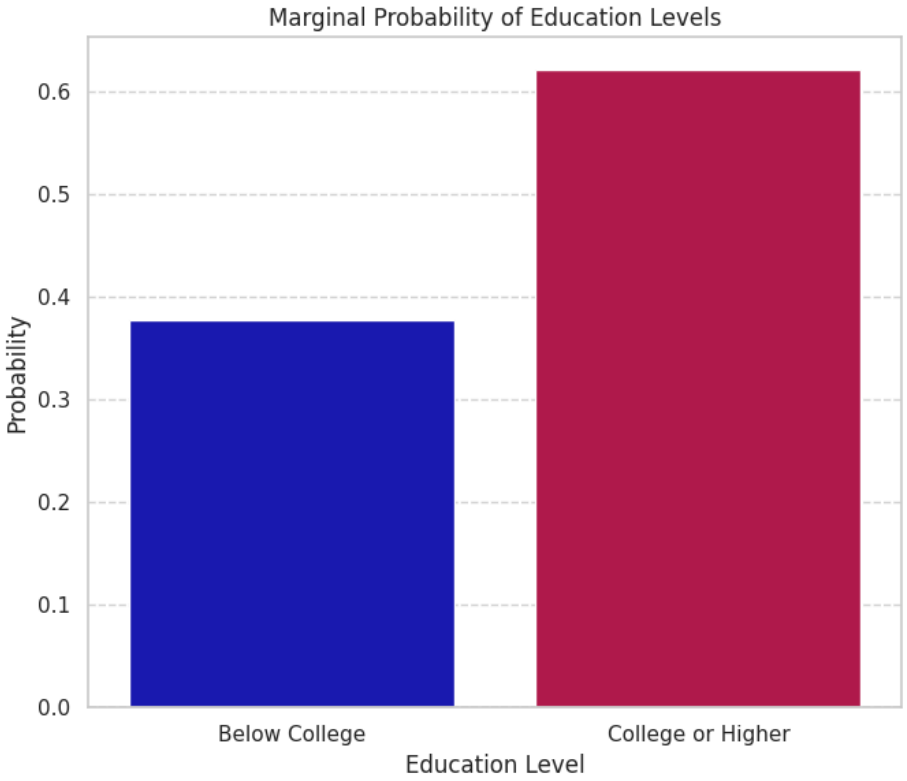
3.8.1 What is the marginal probability of customers having a college degree, and how does this educational level condition their choice of treadmill, particularly in relation to their income category?

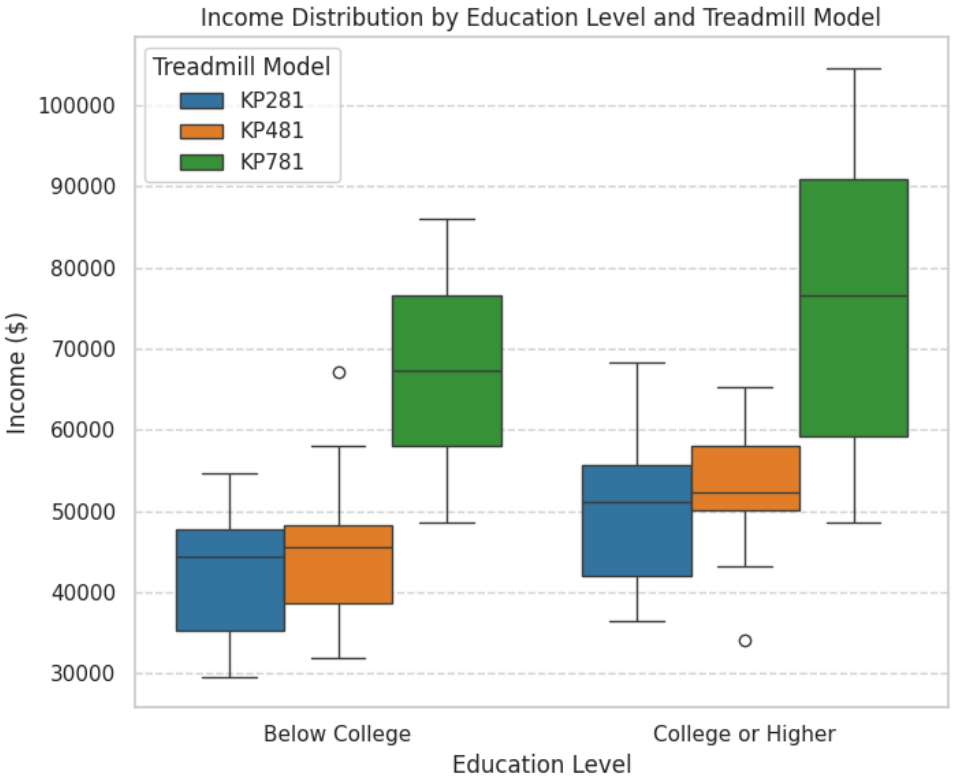
In [149…

```
# Check if the 'Education Level' column exists in the DataFrame and create it if it does not.
# This is based on the 'Education' column which contains numeric years of education.
if 'Education Level' not in df.columns:
    # Transform 'Education' into a categorical variable 'Education Level'
    df['Education Level'] = df['Education'].apply(lambda x: 'College or Higher' if x >= 16 else 'Below College')

# Crosstab for marginal probability of education levels
education_marginal = pd.crosstab(index=df['Education Level'], columns='Total', normalize=True)

# Crosstab for conditional probabilities of treadmill purchases by education level
conditional_probability = pd.crosstab(index=df['Education Level'], columns=df['Product'], normalize='index')

# Print the probability tables
print("Marginal Probability of Education Levels:")
print(education_marginal)
print("\nConditional Probability of Treadmill Purchase by Education Level:")
print(conditional_probability.applymap(lambda x: f"{x:.2%}"))

# Set up the figure for side-by-side plots
fig, ax = plt.subplots(1, 2, figsize=(14, 6))

# Plotting the marginal probability of having a college degree
custom_palette = ["#0000cc","#cc0044"]
sns.barplot(x=education_marginal.index, y=education_marginal['Total'], ax=ax[0], palette=custom_palette)
ax[0].set_title('Marginal Probability of Education Levels')
ax[0].set_ylabel('Probability')
```

```
ax[0].set_xlabel('Education Level')

# Box plot for income distribution by education level and treadmill model
sns.boxplot(x='Education Level', y='Income', hue='Product', data=df, palette='tab10', ax=ax[1])
ax[1].set_title('Income Distribution by Education Level and Treadmill Model')
ax[1].set_xlabel('Education Level')
ax[1].set_ylabel('Income ($)')
ax[1].legend(title='Treadmill Model', loc='upper left')

plt.tight_layout()
plt.show()
```

```
Marginal Probability of Education Levels:
col_0                 Total
Education Level
Below College      0.377778
College or Higher  0.622222

Conditional Probability of Treadmill Purchase by Education Level:
Product             KP281   KP481   KP781
Education Level
Below College      57.35%  39.71%   2.94%
College or Higher  36.61%  29.46%  33.93%
```





Insights of Marginal Probability of Education Levels and Income Distribution by Education Level and Treadmill Model

- **Marginal Probability:**
  The majority of customers (**62.22%**) have a college or higher education, highlighting a predominantly well-educated customer base.
- **Education and Model Choice:**
  Customers with lower education levels (Below College) show a strong preference for the **KP281 model**, with **57.35%** choosing it.
- **High-End Treadmill Preference:**
  Customers with higher education levels have a more **balanced distribution** across treadmill models, with **33.93%** opting for the high-end **KP781**.
- **Income Distribution by Education:**
  The box plot shows that **income increases with education**, with college-educated customers tending to have higher incomes, enabling them to afford more expensive models.
- **Income and Treadmill Model:**
  Among the college-educated group, there is an **upward trend in income** from KP281 to KP781, indicating income significantly influences treadmill model choice.

This graph illustrates that **higher education levels** are associated with **higher incomes** and a greater preference for high-end treadmill models. Marketing strategies should consider educational background as a key factor in influencing customer purchasing behavior.

---

4. Missing Value & Outlier Detection

Using df_raw for missing value & outlier detection

4.1 Identifying Missing Values

In [150…

```
# Count missing values per column
missing_values = df_raw.isnull().sum()
missing_percentage = (df_raw.isnull().sum() / len(df)) * 100

# Combine the results into a single DataFrame for clarity
missing_data_df = pd.DataFrame({
    'Missing Values': missing_values,
    'Percentage (%)': missing_percentage
```

```
}).sort_values(by='Percentage (%)', ascending=False)

print("Missing Values Summary:")
print(missing_data_df)
```

```
Missing Values Summary:
              Missing Values  Percentage (%)
Product                    0             0.0
Age                        0             0.0
Gender                     0             0.0
Education                  0             0.0
MaritalStatus              0             0.0
Usage                      0             0.0
Fitness                    0             0.0
Income                     0             0.0
Miles                      0             0.0
```

Insights from Missing Value Summary

- **Complete Records:**
  All customer data fields (Product, Age, Gender, etc.) are fully populated, indicating comprehensive and thorough data collection. This completeness ensures that any analysis or customer profiling is based on full information.

- **Ready for Immediate Use:**
  With no missing values, the dataset is ready for immediate analysis without the need for any preliminary cleaning steps. This means faster turnaround times for generating customer insights and profiles.

- **High Data Reliability:**
  The absence of missing data enhances the reliability of the analytics performed, ensuring that the insights generated are accurate and reflective of the entire customer base.

The dataset's complete absence of missing values ensures that AeroFit can perform uninterrupted and reliable analyses. This foundation enables precise customer segmentation and profiling, crucial for developing targeted marketing campaigns and enhancing customer engagement strategies.

4.2 Detecting Outliers

In [151…

```
# Function to count and return the number of outliers in each column of a DataFrame
def count_outliers(df, columns):
    outlier_counts = {}
    for column in columns:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
```

```
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        # Count outliers
        column_outliers = df[(df[column] < lower_bound) | (df[column] > upper_bound)]
        outlier_counts[column] = column_outliers.shape[0]

    return outlier_counts

# Columns to check for outliers
columns = df_raw.select_dtypes(include='int64').columns

# Count outliers in specified columns
outlier_counts = count_outliers(df_raw, columns)

# Create a DataFrame to display the counts
outlier_counts_df = pd.DataFrame(list(outlier_counts.items()), columns=['Column', 'Outlier Count'])
print("Number of outliers detected in each column:")
print(outlier_counts_df)
```

```
Number of outliers detected in each column:
      Column  Outlier Count
0        Age              5
1  Education              4
2      Usage              9
3    Fitness              2
4     Income             19
5      Miles             13
```

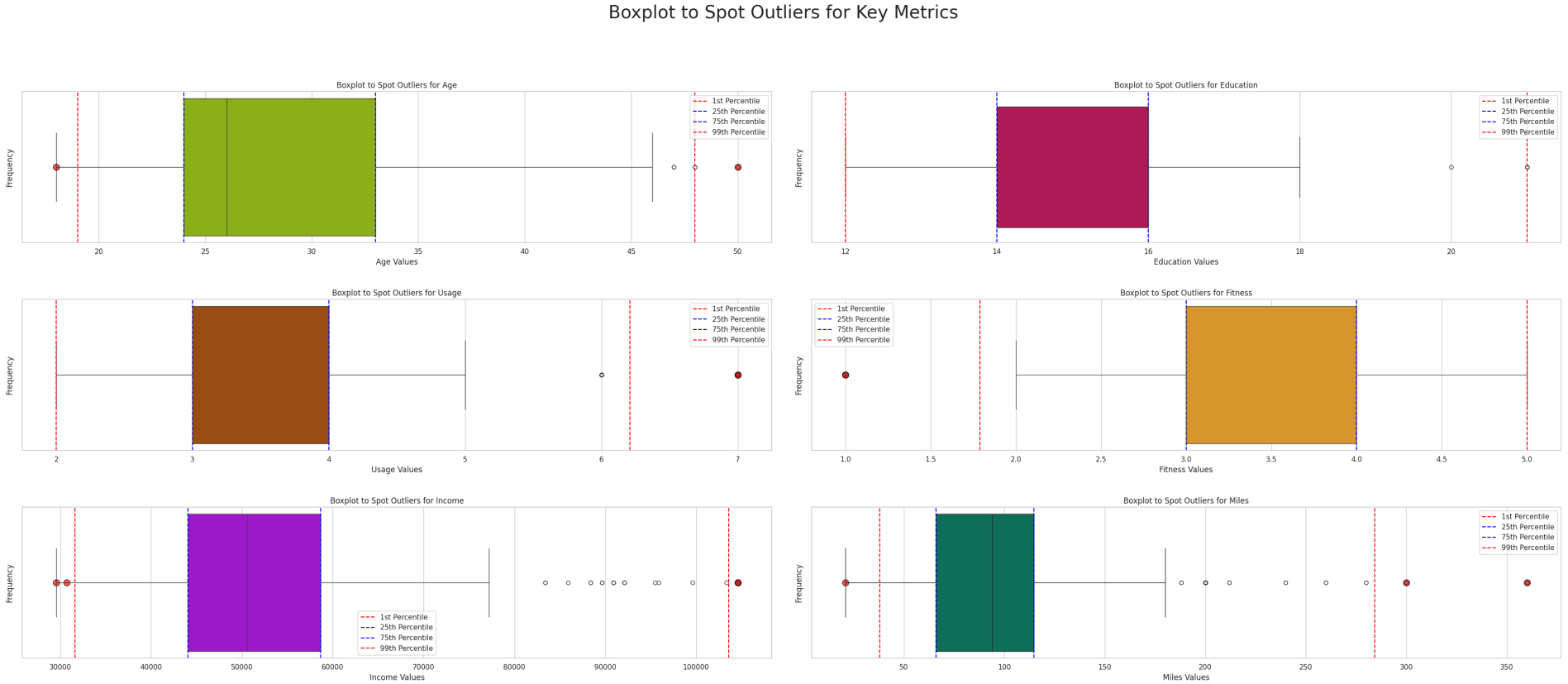
4.3 Visualizing Outliers

Boxplot to Spot Outliers:

In [152…

```
# Setup for plots
sns.set(style="whitegrid")
fig, axes = plt.subplots(3, 2, figsize=(35, 15))
axes = axes.flatten()

# Define a custom color palette for each boxplot for visual distinction
custom_palette = ["#99cc00", "#cc0052", "#b34700", "#f39c12", "#ac00e6", "#008066"]

# Loop through each column to create a boxplot and highlight percentiles and outliers
for i, col in enumerate(columns):
    sns.boxplot(x=df_raw[col], ax=axes[i], color=custom_palette[i])  # Boxplot for each column with specified color
    # Calculate quantiles to place percentile lines
    P1, Q1, Q3, P99 = df_raw[col].quantile([0.01, 0.25, 0.75, 0.99])
    # Adding vertical lines to mark the 1st, 25th, 75th, and 99th percentiles
```

```
    axes[i].axvline(x=P1, color='red', linestyle='--', label='1st Percentile')
    axes[i].axvline(x=Q1, color='blue', linestyle='--', label='25th Percentile')
    axes[i].axvline(x=Q3, color='blue', linestyle='--', label='75th Percentile')
    axes[i].axvline(x=P99, color='red', linestyle='--', label='99th Percentile')
    axes[i].set_title(f'Boxplot to Spot Outliers for {col}')  # Set title for each subplot
    axes[i].set_xlabel(f'{col} Values')  # Label for x-axis
    axes[i].set_ylabel('Frequency')  # Label for y-axis

    # Marking outliers with a red dot
    outliers = df_raw[col][(df_raw[col] < P1) | (df_raw[col] > P99)]
    for outlier in outliers:
        axes[i].scatter(outlier, 0, color='red', s=100, edgecolor='black', alpha=0.7)

    axes[i].legend()

# Add a descriptive suptitle for the entire figure
plt.suptitle('Boxplot to Spot Outliers for Key Metrics', fontsize=28, y=1.02)
print("\n")
plt.tight_layout(pad=3.0)
plt.show()  # Display the plot
```



Insights of Boxplot to Spot Outliers for Key Metrics

- **Age Variability:**
  The age group largely centers around **midlife**, yet outliers present **under 25** and **over 45** suggest diverse treadmill usage across generations.

- **Usage Patterns:**
  Regular usage peaks around the **median** but displays outliers at **extremely low and high frequencies**, highlighting varying commitment levels among users.

- **Educational Distribution:**
  Most users are **well-educated**, with outliers indicating a few users have either **minimal or exceptionally high education levels**.

- **Fitness Range:**
  Users generally report **moderate to high fitness levels**; however, very high fitness ratings indicate some users are potentially **elite athletes or fitness enthusiasts**.

- **Income Disparity:**
  The income distribution shows **significant diversity**, with outliers in the **upper income brackets**, indicating that some **high earners** are also treadmill users.

- **Mileage Extremes:**
  Most users log **moderate miles**, with outliers at both ends suggesting some **barely use their treadmills** while others are **exceptionally active**.

- **Focused Marketing Strategy:**
  The presence of outliers across the metrics can help in tailoring **marketing strategies** to appeal to both **typical users** and **outliers** who may require specialized treadmill features.

- **Product Development Insight:**
  Outliers in usage and fitness levels could influence new features or adaptations in **treadmill designs** to accommodate **extreme usage scenarios**.

- **Financial Strategy Consideration:**
  Understanding the **wide income range** among users can guide **pricing strategies** to make treadmills accessible to more people while also catering to **affluent customers**.

- **Red Circle Indicators:**
  The **red circles** mark the outliers, making it visually straightforward to identify **values that deviate significantly from the norm**, which are important for addressing specific customer needs.

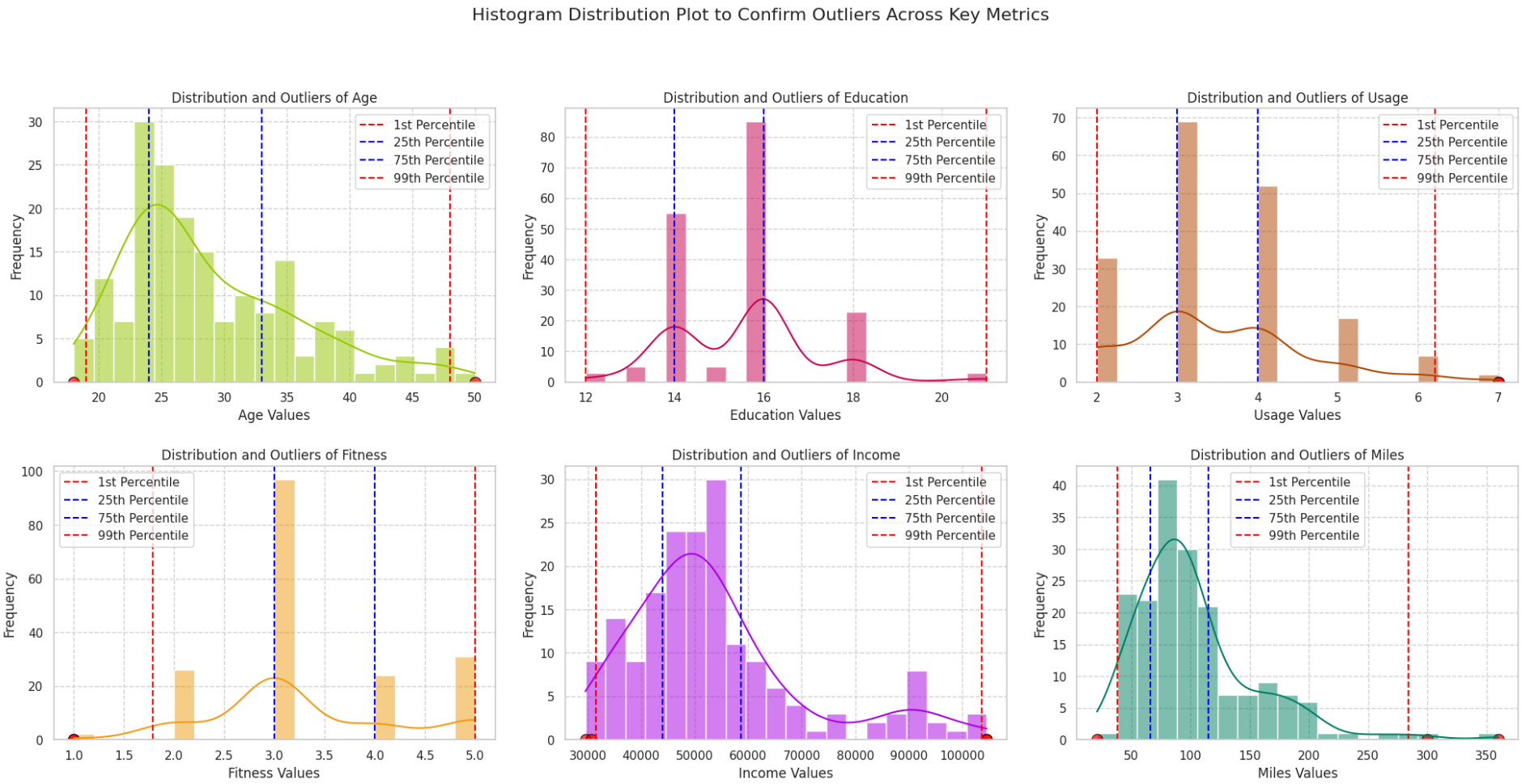
This graph indicates that the range and distribution of values in key metrics like **age, income, and usage** include notable **outliers**, especially in **income and miles**. Acknowledging these outliers can guide **tailored marketing strategies** and **product developments** to address **diverse customer needs** effectively.

Distribution Plot to Confirm Outliers:

In [153…

```
# Setup for plots
sns.set_style("whitegrid", {
    'grid.color': 'lightgray',
    'grid.linestyle': '--',
    'grid.linewidth': 0.5,
})

# Creating a figure with a grid of subplots for histogram plots
fig, axes = plt.subplots(2, 3, figsize=(20, 10))
axes = axes.flatten()

# Define a custom color palette for each histogram for visual distinction
custom_palette = ["#99cc00", "#cc0052", "#b34700", "#f39c12", "#ac00e6", "#008066"]

# Loop through each column to create a histogram for outliers
for i, col in enumerate(columns):
    # Create a histogram for each column
    sns.histplot(df_raw[col], kde=True, color=custom_palette[i], ax=axes[i], bins=20)
    # Marking the 1st, 25th, 75th, and 99th percentiles
    P1, Q1, Q3, P99 = df_raw[col].quantile([0.01, 0.25, 0.75, 0.99])
    axes[i].axvline(x=P1, color='red', linestyle='--', label='1st Percentile')
    axes[i].axvline(x=Q1, color='blue', linestyle='--', label='25th Percentile')
    axes[i].axvline(x=Q3, color='blue', linestyle='--', label='75th Percentile')
    axes[i].axvline(x=P99, color='red', linestyle='--', label='99th Percentile')
    axes[i].set_title(f'Distribution and Outliers of {col}')  # Set title for each subplot
    axes[i].set_xlabel(f'{col} Values')  # Label for x-axis
    axes[i].set_ylabel('Frequency')  # Label for y-axis

    # Highlighting outliers directly on the plot
    outliers_below = df_raw[col][df_raw[col] < P1]
    outliers_above = df_raw[col][df_raw[col] > P99]
    for outlier in outliers_below:
        axes[i].scatter(outlier, 0, color='red', s=100, edgecolor='black', alpha=0.7)
    for outlier in outliers_above:
        axes[i].scatter(outlier, 0, color='red', s=100, edgecolor='black', alpha=0.7)

    axes[i].legend()

# Adjust layout to prevent overlap and ensure clarity
plt.tight_layout()
```

```
# Add a descriptive suptitle for the entire figure
plt.suptitle('Histogram Distribution Plot to Confirm Outliers Across Key Metrics', fontsize=16, y=1.02)
print("\n")
plt.tight_layout(pad=2.0)
plt.show()
```



Insights from Distribution Plot to Confirm Outliers Across Key Metrics

- **Age Distribution:**
  The majority of data points for age are clustered around **30-40 years**, with outliers beyond the **50-year mark**, indicating fewer older customers.

- **Education Distribution:**
  Most customers have around **16 years of education**, typical for those with a **college degree**, with few outliers suggesting **higher education levels**.

- **Usage Distribution:**
  Usage patterns are heavily concentrated around **3 to 5 times per week**, with minimal activity noted as extreme outliers at **2 and 7 times per week**.

- **Fitness Distribution:**
  Fitness levels show a clear peak at **level 3**, with levels beyond **4.5** being rare and treated as outliers, reflecting a **lesser number of highly fit individuals**.

- **Income Distribution:**
  The income distribution is skewed towards the **lower end** but has a long tail extending to **$100,000**, with significant outliers beyond this range, suggesting a **few high earners**.

- **Miles Distribution:**
  Most customers run between **50 to 150 miles**, with a steep drop-off and notable outliers above **250 miles**, indicating **exceptional usage**.

- **Outlier Marking:**
  Outliers, marked in **red**, clearly indicate values that deviate significantly from the common range, essential for assessing **data quality and accuracy**.

- **Percentile Lines:**
  The incorporation of **1st, 25th, 75th, and 99th percentile lines** helps in visually assessing the spread and concentration of data, facilitating quick **outlier identification**.

- **Visual Clarity:**
  The use of **distinct colors** for different percentiles and histograms enhances the **clarity and readability** of the distribution of each variable.

- **Data Consistency:**
  The patterns observed across multiple metrics show **consistent recording practices**, which is crucial for **reliable analysis**.

This graph indicates the effectiveness of using **histogram overlays with percentile markers** to identify and confirm outliers in key metrics, revealing both **typical trends** and **extreme values** in customer data. This approach is invaluable for ensuring **robust data analysis** and **informed decision-making**.

---

5. Business Insights Based on Non-Graphical and Visual Analysis

A) Comments on the Range of Attributes

i. Age

- **Range:** Aerofit's treadmills are used by people from their **early 20s up to their mid-40s**.
- **Implication:** The treadmills should work well for both **young adults and older individuals**.
- **Marketing Tip:** Show how the treadmills are great for both **beginners and experienced users** in ads.

ii. Income

- **Range:** Most buyers earn a **middle income**, but there are also some **wealthier buyers**.
- **Implication:** Aerofit's treadmills are priced right for most people, and there's also a chance to sell more **expensive models**.
- **Strategy:** Keep promoting **good value** and also talk up the fancier treadmills to people who might spend more.

iii. Usage and Fitness Levels

- **Range:** Some people use the treadmills just a few times a month, others use them **every day**. Fitness levels vary a lot too.
- **Implication:** Aerofit needs treadmills that are good for both **casual and very active users**.
- **Product Line Strategy:** Offer simple treadmills for **light users** and more advanced ones for those who **train hard**.

B) Comments on the Distribution of the Variables and Relationship Between Them

i. Age and Marital Status

- **Observation:** Older users are often **married or have a partner**.
- **Marketing Strategy:** Suggest that treadmills can help **couples or families stay healthy together**.

ii. Income and Education

- **Observation:** People who earn more usually have **higher education**.
- **Product Positioning:** Target the more expensive treadmills at people with higher education, highlighting **smart features** and better performance.

iii. Usage and Miles

- **Observation:** The more often people use their treadmills, the more **miles they log**.
- **Product Reliability and Endurance:** Talk about how **durable** the treadmills are, good for those who use them a lot.

C) Comments for Each Univariate and Bivariate Plot

i. Univariate Insights

- **Income Distribution:**

    - **Observation:** Most treadmill buyers make a **middle-level income**.
    - **Marketing Focus:** Focus ads on how **affordable** the treadmills are.
- **Age Distribution:**
    - **Observation:** Most buyers are between **25 and 45 years old**.
    - **Target Marketing:** Tailor ads to fit the **lifestyle and fitness needs** of this age group.

ii. Bivariate Insights

- **Income vs. Treadmill Model:**
    - **Observation:** Richer people tend to buy the most expensive model, **KP781**.
    - **Action:** Market KP781 as a **premium choice**, perfect for those who want the best.
- **Usage Frequency vs. Model Type:**
    - **Observation:** People who use treadmills a lot prefer the **KP781**.
    - **Marketing Tip:** Highlight how **sturdy and suitable** KP781 is for frequent use.
- **Age vs. Model Preference:**
    - **Observation:** Younger folks often go for the cheaper **KP281**; older people like the pricier **KP781**.
    - **Strategy:** Advertise KP281 as **budget-friendly** for younger users and KP781 as **packed with features** for older users.

Aerofit can boost its market presence by strategically marketing its treadmills' affordability and versatility to a diverse range of customers, from young adults to those in their mid-40s, ensuring it meets varying fitness and income levels.

---

6. Business Insights for AeroFit: Strategies Based on Customer Data and Market Analysis

Age Diversity:

- **Wide User Base:** AeroFit treadmills appeal to a **diverse age range** from **young adults to seniors**.
- **Product Appeal:** This broad appeal suggests the treadmills are **versatile**, suitable for **various fitness levels and needs**.

Income Levels:

- **Middle-Income Majority:** The core customer base consists of **middle-income earners**, indicating the treadmills are seen as **affordable and value-for-money**.
- **High-Income Interest:** There's growing interest from **high-income consumers**, possibly attracted by **premium features**.

Usage Patterns:

- **High User Satisfaction:** Regular users report **high satisfaction**, indicating the treadmills meet or exceed their **fitness expectations**.
- **Correlation with Loyalty:** Frequent use is likely linked to **customer loyalty**, suggesting satisfaction drives **repeat usage**.

Demographic Variations:

- **Young Adults:** Younger users tend to prefer more **technologically advanced** or **budget-friendly models**.
- **Older Adults:** Older adults often opt for treadmills with enhanced **comfort and usability features**.
- **Families vs. Singles:** Families and partnered individuals may prefer treadmills that support **multiple user profiles** or have features that are conducive to **shared use**.

Product Usage Insights:

- **Diverse Application:** Treadmills are used for a range of activities, from **light walking** to **intense running sessions**.
- **Feature Utilization:** Different features appeal to different segments, such as **robust build quality** for intense users and **interactive programs** for technologically adept users.

These expanded insights provide a **deeper understanding** of AeroFit's **market dynamics** and **customer preferences**, highlighting areas of **strength** and potential **growth opportunities** within different consumer segments.

---

7. Recommendations

Develop Age-Specific Models:

- Create treadmills for different age groups.
- Focus on strong and feature-rich models for younger adults.
- For older adults, make treadmills easy to use and gentle on joints.

Expand High-End Offerings:

- Launch advanced treadmills with features like fitness tracking and personalized programs.
- Target customers with higher incomes who want top-quality products.

Enhance Customer Loyalty Programs:

- Start reward programs that give benefits for regular use, like discounts or special content.
- Keep customers coming back by offering them valuable rewards.

Promote Family and Couple Discounts:

- Give special prices for couples or families, which might lead to buying more than one treadmill.
- Make these offers to attract older customers and those with partners.

Targeted Marketing Campaigns:

- Use information from data to create specific ads and promotions.
- Focus on what customers aged 25-45 like and need to increase sales.

Increase Durability Focus:

- Talk about how durable and reliable the treadmills are in advertisements.
- Think about offering longer warranties to show the treadmills last a long time.

Adjust Inventory Based on Demand:

- Keep an eye on sales trends to make sure the number of treadmills matches customer demand.
- This helps avoid having too many or too few treadmills in stock.

Regular Customer Feedback Surveys:

- Regularly ask customers what they think about the treadmills and service.
- Use their responses to make the treadmills and service better.

Introduce Interactive Features:

- Add fun tech features like virtual reality workouts or online competitions for tech-savvy users.
- These features make workouts more interesting and help stand out in the market.

Streamline Customer Support:

- Make sure help is easy to get, knows a lot, and responds fast.
- Good support increases customer happiness and trust in AeroFit.

These recommendations focus on meeting specific needs based on what we know about customers. This helps AeroFit improve products, keep customers happy, and grow the business effectively.

---

*Created by **Rishabh Dev Sahu*** ✨