In [ ]:
```
Problem Statement:
Create comprehensive customer profiles for each AeroFit treadmill product through d
Develop two-way contingency tables and analyze conditional and marginal probabiliti

About data:
The company collected the data on individuals who purchased a treadmill from the Ae
 Product Portfolio
 The KP281 is an entry-level treadmill that sells for USD 1,500.
 The KP481 is for mid-level runners that sell for USD 1,750.
 The KP781 treadmill is having advanced features that sell for USD 2,500.

Data Set:
Feature              Description
Product              Product Purchased: KP281, KP481, or KP781
 Age                 Age of buyer in years
 Gender              Gender of the buyer (Male/Female)
 Education           Education of buyer in years
 MaritalStatus       MaritalStatus of buyer (Single or partnered)
 Usage               The average number of times the buyer plans to use the treadmi
 Income              Annual income of the buyer (in $)
 Fitness             Self-rated fitness on a 1-to-5 scale, where 1 is the poor shap
 Miles               The average number of miles the buyer expects to walk/run each



and Analysing basic metrics (10 Points)
Observations on shape of data, data types of all the attributes, conversion of cate
```

In [7]:
```
#importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import copy
```

In [15]:
```
# loading the dataset
df = pd.read_csv('C:\\Users\\deepa\\Documents\\Scalar\\Aerofit\\aerofit_treadmill.
```

In [17]:
```
df.head()
```

Out[17]:

	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

In [19]:
```
df.tail()
```

Out[19]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200
177	KP781	45	Male	16	Single	5	5	90886	160
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

In [23]:
```
df.shape
```

Out[23]:
```
(180, 9)
```

In [25]:
```
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
```

In [ ]:
```
Changing the Datatype of Columns
#Changing the datatype of Usage and Fitness columns
```

In [33]:
```
df['Fitness'] = df['Fitness'].astype('str')
```

In [35]:
```
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    object
 7   Income         180 non-null    int64 
 8   Miles          180 non-null    int64 
dtypes: int64(5), object(4)
memory usage: 12.8+ KB
```

In [ ]:
```
Statistical Summary
```

In [37]:
```
df.describe(include = 'object')
```

Out[37]:

	Product	Gender	MaritalStatus	Fitness
count	180	180	180	180
unique	3	2	2	5
top	KP281	Male	Partnered	3
freq	80	104	107	97

In [ ]:
```
#1. Product - Over the past three months, the KP281 product demonstrated the highes
 #44% of total sales.
 #2. Gender - Based on the data of last 3 months, around 58% of the buyers were Mal
# 3. Marital Status - Based on the data of last 3 months, around 60% of the buyers
```

In [39]:
```
 # statisctical summary of numerical data type columns
 df.describe()
```

Out[39]:

	Age	Education	Usage	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	53719.577778	103.194444
std	6.943498	1.617055	1.084797	16506.684226	51.863605
min	18.000000	12.000000	2.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	104581.000000	360.000000

In [ ]:
```
#1. Age - The age range of customers spans from 18 to 50 year, with an average age
 #2. Education - Customer education levels vary between 12 and 21 years, with an av
 #3. Usage - Customers intend to utilize the product anywhere from 2 to 7 times per
 #4. Fitness - On average, customers have rated their fitness at 3 on a 5-point sca
 #5. Income - The annual income of customers falls within the range of USD 30,000 t
 #6. Miles - Customers' weekly running goals range from 21 to 360 miles, with an av
```

In [ ]:
```
Duplicate Detection
```

In [41]:
```
df.duplicated().value_counts()
```

Out[41]:
```
False    180
Name: count, dtype: int64
```

In [ ]:
```
#There are no duplicate entries in the dataset
```

In [ ]:
```
 Sanity Check for columns
```

In [43]:
```
 # checking the unique values for columns
 for i in df.columns:
    print('Unique Values in',i,'column are :-')
    print(df[i].unique())
    print('-'*70)
```

```
Unique Values in Product column are :-
['KP281' 'KP481' 'KP781']
----------------------------------------------------------------------
Unique Values in Age column are :-
[18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
 43 44 46 47 50 45 48 42]
----------------------------------------------------------------------
Unique Values in Gender column are :-
['Male' 'Female']
----------------------------------------------------------------------
Unique Values in Education column are :-
[14 15 12 13 16 18 20 21]
----------------------------------------------------------------------
Unique Values in MaritalStatus column are :-
['Single' 'Partnered']
----------------------------------------------------------------------
Unique Values in Usage column are :-
[3 2 4 5 6 7]
----------------------------------------------------------------------
Unique Values in Fitness column are :-
['4' '3' '2' '1' '5']
----------------------------------------------------------------------
Unique Values in Income column are :-
[ 29562  31836  30699  32973  35247  37521  36384  38658  40932  34110
  39795  42069  44343  45480  46617  48891  53439  43206  52302  51165
  50028  54576  68220  55713  60261  67083  56850  59124  61398  57987
  64809  47754  65220  62535  48658  54781  48556  58516  53536  61006
  57271  52291  49801  62251  64741  70966  75946  74701  69721  83416
  88396  90886  92131  77191  52290  85906 103336  99601  89641  95866
 104581  95508]
----------------------------------------------------------------------
Unique Values in Miles column are :-
[112  75  66  85  47 141 103  94 113  38 188  56 132 169  64  53 106  95
 212  42 127  74 170  21 120 200 140 100  80 160 180 240 150 300 280 260
 360]
----------------------------------------------------------------------
```

In [ ]:
```
#The dataset does not contain any abnormal values.
 Adding new columns for better analysis

 Creating New Column and Categorizing values in Age,Education,Income and Miles to d
 Age Column
 Categorizing the values in age column in 4 different buckets:
 1. Young Adult: from 18 - 25
 2. Adults: from 26 - 35
 3. Middle Aged Adults: 36-45
 4. Elder :46 and above
 Education Column
 Categorizing the values in education column in 3 different buckets:
 1. Primary Education: upto 12
 2. Secondary Education: 13 to 15
 3. Higher Education: 16 and above
 Income Column
 Categorizing the values in Income column in 4 different buckets:
 1. Low Income - Upto 40,000
 2. Moderate Income - 40,000 to 60,000
```

```
3. High Income - 60,000 to 80,000
4. Very High Income - Above 80,000
Miles column
Categorizing the values in miles column in 4 different buckets:
1. Light Activity - Upto 50 miles
2. Moderate Activity - 51 to 100 miles
3. Active Lifestyle - 101 to 200 miles
4. Fitness Enthusiast - Above 200 miles
```

In [139…

```
# Binning the Age values into categories
bin_range1 = [17, 25, 35, 45, float('inf')]
bin_labels1 = ['Young Adults', 'Adults', 'Middle Aged Adults', 'Elder']
df['age_group'] = pd.cut(df['Age'], bins=bin_range1, labels=bin_labels1)
# Binning the education values into categories
bin_range2 = [0, 12, 15, float('inf')]
bin_labels2 = ['Primary Education', 'Secondary Education', 'Higher Education']
df['edu_group'] = pd.cut(df['Education'], bins=bin_range2, labels=bin_labels2)
# Binning the income values into categories
bin_range3 = [0, 40000, 60000, 80000, float('inf')]
bin_labels3 = ['Low Income', 'Moderate Income', 'High Income', 'Very High Income']
df['income_group'] = pd.cut(df['Income'], bins=bin_range3, labels=bin_labels3)
# Bin the 'Miles' values into categories
bin_range4 = [0, 50, 100, 200, float('inf')]
bin_labels4 = ['Light Activity', 'Moderate Activity', 'Active Lifestyle', 'Fitness
df['miles_group'] = pd.cut(df['Miles'], bins=bin_range4, labels=bin_labels4)
```

In [141…

```
df.head()
```

Out[141…

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	incom
0	KP281	18	Male	14	Single	3	4	29562	112	Low
1	KP281	19	Male	15	Single	2	3	31836	75	Low
2	KP281	19	Female	14	Partnered	4	3	30699	66	Low
3	KP281	19	Male	12	Single	3	3	32973	85	Low
4	KP281	20	Male	13	Partnered	4	2	35247	47	Low

In [ ]:

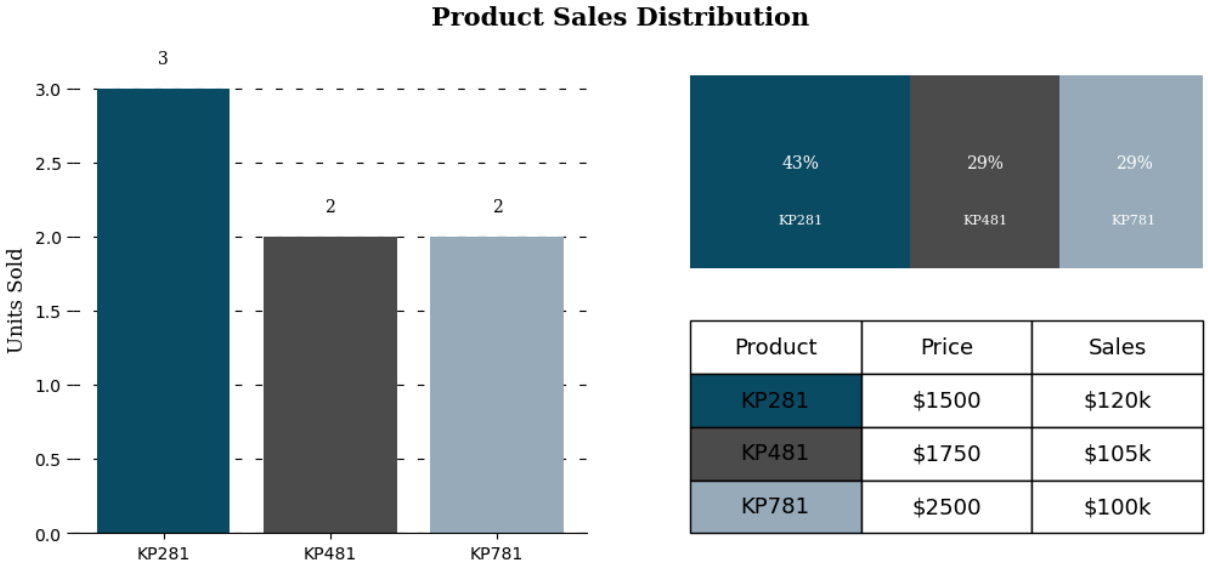
```
3.Univariate Analysis
 3.1 Categorical Variables
 3.1.1  Product Sales Distribution
```

In [77]:

```
import matplotlib.pyplot as plt
import pandas as pd

# Sample DataFrame for demonstration
data = {'Product': ['KP281', 'KP481', 'KP781', 'KP281', 'KP481', 'KP781', 'KP281']}
```

```
df = pd.DataFrame(data)

# Setting the plot style
fig = plt.figure(figsize=(12, 5))
gs = fig.add_gridspec(2, 2)

# Creating plot for the product column
ax0 = fig.add_subplot(gs[:, 0])
product_count = df['Product'].value_counts()
color_map = ["#0e4f66", "#4b4b4c", '#99AEBB']
ax0.bar(product_count.index, product_count.values, color=color_map, zorder=2)

# Adding the value counts
for i in range(len(product_count)):
    ax0.text(i, product_count[i] + 0.2, product_count[i],
             {'font': 'serif', 'size': 10}, ha='center', va='center')

# Adding grid lines
ax0.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax0.spines[s].set_visible(False)

# Adding axis label
ax0.set_ylabel('Units Sold', fontfamily='serif', fontsize=12)

# Creating a plot for product % sale
ax1 = fig.add_subplot(gs[0, 1])
product_count_percent = ((product_count.values / df.shape[0]) * 100).round()
ax1.barh(product_count.index[0], product_count_percent[0], color="#0e4f66")
ax1.barh(product_count.index[0], product_count_percent[1],
         left=product_count_percent[0], color='#4b4b4c')
ax1.barh(product_count.index[0], product_count_percent[2],
         left=product_count_percent[0] + product_count_percent[1], color='#99AEBB')
ax1.set(xlim=(0, 100))

# Adding info to each bar
info_percent = [
    product_count_percent[0] / 2,
    product_count_percent[0] + product_count_percent[1] / 2,
    product_count_percent[0] + product_count_percent[1] + product_count_percent[2]
]
for i in range(3):
    ax1.text(info_percent[i], 0.04, f"{product_count_percent[i]:.0f}%",
             va='center', ha='center', fontsize=10, fontweight='light', fontfamily=
    ax1.text(info_percent[i], -0.2, product_count.index[i],
             va='center', ha='center', fontsize=8, fontweight='light', fontfamily='

# Removing the axis lines
ax1.axis('off')

# Creating a plot for product portfolio
ax2 = fig.add_subplot(gs[1, 1])
product_portfolio = [['KP281', '$1500', '$120k'], ['KP481', '$1750', '$105k'], ['KP
color_2d = [['#0e4f66', '#FFFFFF', '#FFFFFF'], ['#4b4b4c', '#FFFFFF', '#FFFFFF'], [
```

```
table = ax2.table(cellText=product_portfolio, cellColours=color_2d, cellLoc='center
                  colLabels=['Product', 'Price', 'Sales'], colLoc='center', bbox=[0
table.set_fontsize(13)

# Removing axis
ax2.axis('off')

# Adding title to the visual
fig.suptitle('Product Sales Distribution',
             fontproperties={'family': 'serif', 'size': 15, 'weight': 'bold'})
plt.show()
```

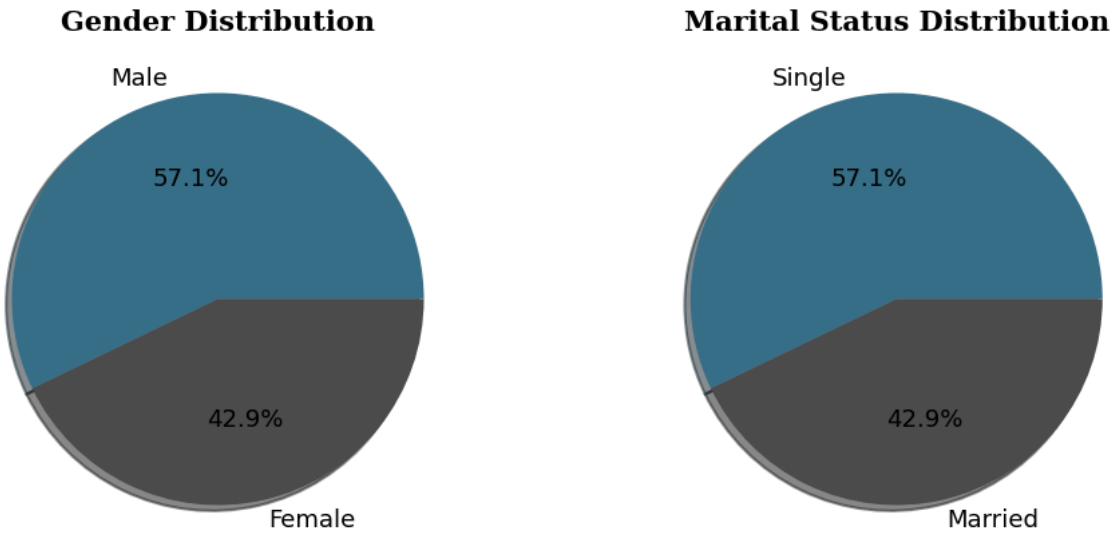


In [ ]:
```
 #The KP281 treadmill model, positioned as an entry-level product, has the highest
# All three models have nearly equal contributions in terms of generating sales rev
```

In [ ]:
```
Gender and Martial Status Distribution
```

In [85]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Sample DataFrame for demonstration
data = {'Gender': ['Male', 'Female', 'Male', 'Female', 'Female', 'Male', 'Male'],
        'MaritalStatus': ['Single', 'Married', 'Single', 'Single', 'Married', 'Marr
df = pd.DataFrame(data)

# Setting the plot style
fig = plt.figure(figsize=(12, 5))
gs = fig.add_gridspec(1, 2)

# Creating pie chart for gender distribution
ax0 = fig.add_subplot(gs[0, 0])
color_map_gender = ["#3A7089", "#4b4b4c"]
ax0.pie(df['Gender'].value_counts().values,
        labels=df['Gender'].value_counts().index,
        autopct='%.1f%%',
        shadow=True,
        colors=color_map_gender,
```

```
        wedgeprops={'linewidth': 1.5},
        textprops={'fontsize': 13, 'color': 'black'})
# Setting title for visual
ax0.set_title('Gender Distribution', fontdict={'font': 'serif', 'size': 15, 'weight

# Creating pie chart for marital status
ax1 = fig.add_subplot(gs[0, 1])
color_map_marital = ["#3A7089", "#4b4b4c"]
ax1.pie(df['MaritalStatus'].value_counts().values,
        labels=df['MaritalStatus'].value_counts().index,
        autopct='%.1f%%',
        shadow=True,
        colors=color_map_marital,
        wedgeprops={'linewidth': 1.5},
        textprops={'fontsize': 13, 'color': 'black'})
# Setting title for visual
ax1.set_title('Marital Status Distribution', fontdict={'font': 'serif', 'size': 15,

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

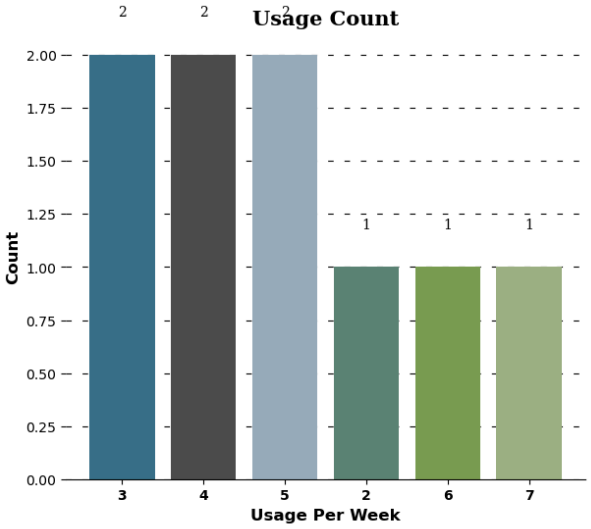


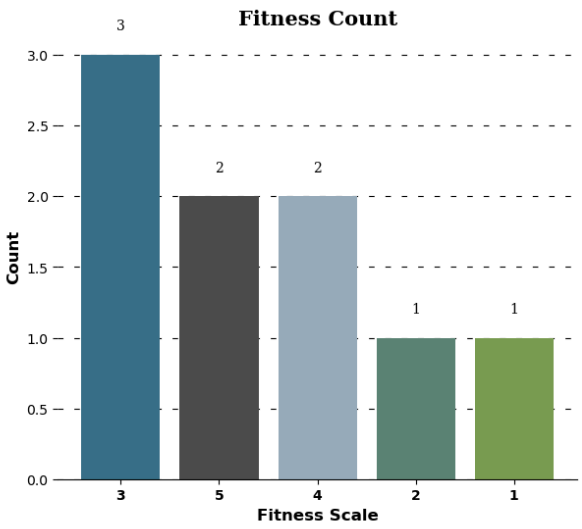
In [ ]:
```
Buyer Fitness and Thread mill usage
```

In [89]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Example DataFrame
data = {'Usage': ['3', '4', '2', '5', '6', '7', '3', '4', '5'],
        'Fitness': ['3', '5', '2', '4', '1', '3', '5', '3', '4']}
df = pd.DataFrame(data)

# Setting the plot style
fig = plt.figure(figsize=(15, 10))
gs = fig.add_gridspec(2, 2, height_ratios=[0.65, 0.35])

# Creating bar chart for usage distribution
ax0 = fig.add_subplot(gs[0, 0])
temp = df['Usage'].value_counts()
```

```
color_map = ["#3A7089", "#4b4b4c", '#99AEBB', '#5C8374', '#7A9D54', '#9EB384']
ax0.bar(x=temp.index, height=temp.values, color=color_map, zorder=2)

# Adding the value_counts
for i in range(len(temp)):
    ax0.text(i, temp[i] + 0.2, temp[i], {'font': 'serif', 'size': 10}, ha='center',

# Adding grid lines
ax0.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax0.spines[s].set_visible(False)

# Adding axis label
ax0.set_ylabel('Count', fontweight='bold', fontsize=12)
ax0.set_xlabel('Usage Per Week', fontweight='bold', fontsize=12)
ax0.set_xticks(range(len(temp.index)))
ax0.set_xticklabels(temp.index, fontweight='bold')
ax0.set_title('Usage Count', {'font': 'serif', 'size': 15, 'weight': 'bold'})

# Creating a table for usage information
ax1 = fig.add_subplot(gs[1, 0])
usage_info = [['3', '38%'], ['4', '29%'], ['2', '19%'], ['5', '9%'], ['6', '4%'], [
color_2d = [["#3A7089", '#FFFFFF'], ["#4b4b4c", '#FFFFFF'], ['#99AEBB', '#FFFFFF'],
            ['#5C8374', '#FFFFFF'], ['#7A9D54', '#FFFFFF'], ['#9EB384', '#FFFFFF']]
table = ax1.table(cellText=usage_info, cellColours=color_2d, cellLoc='center',
                  colLabels=['Usage Per Week', 'Percent'], colLoc='center', bbox=[0
table.set_fontsize(13)
ax1.axis('off')

# Creating bar chart for fitness scale
ax2 = fig.add_subplot(gs[0, 1])
temp = df['Fitness'].value_counts()
ax2.bar(x=temp.index, height=temp.values, color=color_map, zorder=2)

# Adding the value_counts
for i in range(len(temp)):
    ax2.text(i, temp[i] + 0.2, temp[i], {'font': 'serif', 'size': 10}, ha='center',

# Adding grid lines
ax2.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax2.spines[s].set_visible(False)

# Adding axis label
ax2.set_ylabel('Count', fontweight='bold', fontsize=12)
ax2.set_xlabel('Fitness Scale', fontweight='bold', fontsize=12)
ax2.set_xticks(range(len(temp.index)))
ax2.set_xticklabels(temp.index, fontweight='bold')
ax2.set_title('Fitness Count', {'font': 'serif', 'size': 15, 'weight': 'bold'})

# Creating a table for fitness information
ax3 = fig.add_subplot(gs[1, 1])
```

```
fitness_info = [['3', '54%'], ['5', '17%'], ['2', '15%'], ['4', '13%'], ['1', '1%']
color_2d = [["#3A7089", '#FFFFFF'], ["#4b4b4c", '#FFFFFF'], ['#99AEBB', '#FFFFFF'],
            ['#5C8374', '#FFFFFF'], ['#7A9D54', '#FFFFFF']]
table = ax3.table(cellText=fitness_info, cellColours=color_2d, cellLoc='center',
                  colLabels=['Fitness', 'Percent'], colLoc='center', bbox=[0, 0, 1,
table.set_fontsize(13)
ax3.axis('off')

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





Usage Per Week	Percent
3	38%
4	29%
2	19%
5	9%
6	4%
7	1%

Fitness	Percent
3	54%
5	17%
2	15%
4	13%
1	1%

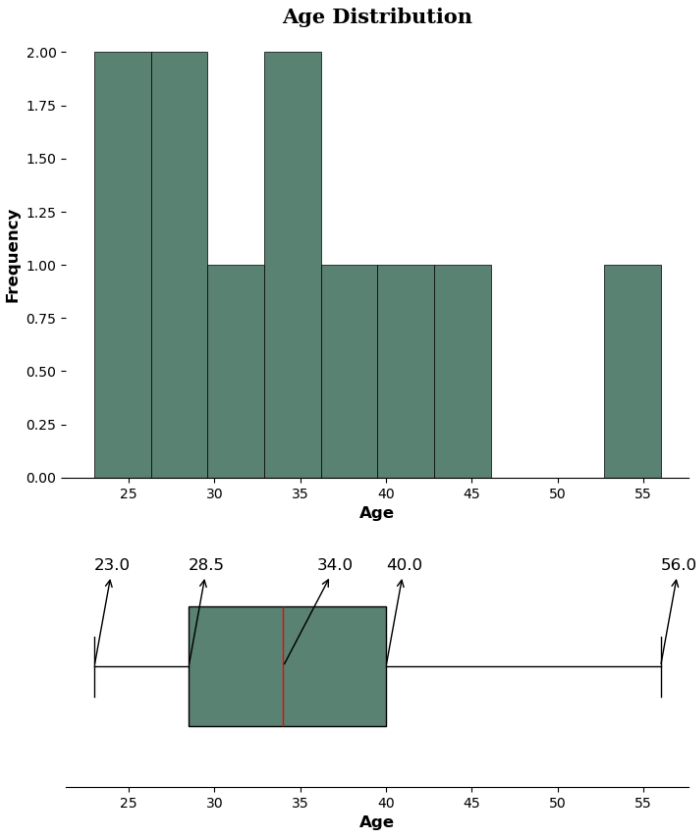
In [ ]:
```
#Almost 85% of the customers plan to use the treadmill for 2 to 4 times a week and
# 54% of the customers have self-evaluated their fitness at a level 3 on a scale of
# or higher, indicating commendable fitness levels.
```

In [ ]:
```
3.2-Numerical Variables
3.2.1-Customer Age Distribution
```

In [95]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Example DataFrame
data = {'Age': [23, 35, 45, 29, 56, 30, 23, 41, 34, 28, 39],
        'age_group': ['Young Adults', 'Adults', 'Middle Aged', 'Adults', 'Elder', '
df = pd.DataFrame(data)

# Setting the plot style
fig = plt.figure(figsize=(15, 10))
gs = fig.add_gridspec(2, 2, height_ratios=[0.65, 0.35], width_ratios=[0.6, 0.4])

# Creating age histogram
```

```
ax0 = fig.add_subplot(gs[0, 0])
ax0.hist(df['Age'], color='#5C8374', linewidth=0.5, edgecolor='black')
ax0.set_xlabel('Age', fontsize=12, fontweight='bold')
ax0.set_ylabel('Frequency', fontsize=12, fontweight='bold')

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax0.spines[s].set_visible(False)

# Setting title for visual
ax0.set_title('Age Distribution', {'font': 'serif', 'size': 15, 'weight': 'bold'})

# Creating box plot for age
ax1 = fig.add_subplot(gs[1, 0])
boxplot = ax1.boxplot(x=df['Age'], vert=False, patch_artist=True, widths=0.5)

# Customize box and whisker colors
boxplot['boxes'][0].set(facecolor='#5C8374')

# Customize median line
boxplot['medians'][0].set(color='red')

# Customize outlier markers
for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor="#4b4b4c")

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax1.spines[s].set_visible(False)

# Adding 5 point summary annotations
info = [i.get_xdata() for i in boxplot['whiskers']]  # Getting the upperlimit, Q1,
median = df['Age'].quantile(0.5)  # Getting Q2
for i, j in info:  # Using i, j here because of the output type of info list compre
    ax1.annotate(text=f"{i:.1f}", xy=(i, 1), xytext=(i, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"
    ax1.annotate(text=f"{j:.1f}", xy=(j, 1), xytext=(j, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"

# Adding the median separately because it was included in info list
ax1.annotate(text=f"{median:.1f}", xy=(median, 1), xytext=(median + 2, 1.4), fontsi
             arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"))

# Removing y-axis ticks
ax1.set_yticks([])

# Adding axis label
ax1.set_xlabel('Age', fontweight='bold', fontsize=12)

# Creating age group bar chart
ax2 = fig.add_subplot(gs[0, 1])
temp = df['age_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c", '#99AEBB', '#5C8374']
ax2.bar(x=temp.index, height=temp.values, color=color_map, zorder=2)

# Adding the value_counts
```

```
for i in temp.index:
    ax2.text(i, temp[i] + 2, temp[i], {'font': 'serif', 'size': 10}, ha='center', v

# Adding grid lines
ax2.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax2.spines[s].set_visible(False)

# Adding axis label
ax2.set_ylabel('Count', fontweight='bold', fontsize=12)
ax2.set_xticklabels(temp.index, fontweight='bold')

# Setting title for visual
ax2.set_title('Age Group Distribution', {'font': 'serif', 'size': 15, 'weight': 'bo

# Creating a table for group info
ax3 = fig.add_subplot(gs[1, 1])
age_info = [['Young Adults', '44%', '18 to 25'], ['Adults', '41%', '26 to 35'], ['M
            ['Elder', '3%', 'Above 45']]
color_2d = [["#3A7089", '#FFFFFF', '#FFFFFF'], ["#4b4b4c", '#FFFFFF', '#FFFFFF'], [
            ['#5C8374', '#FFFFFF', '#FFFFFF']]
table = ax3.table(cellText=age_info, cellColours=color_2d, cellLoc='center',
                  colLabels=['Age', 'Probability', 'Group'], colLoc='center', bbox=
table.set_fontsize(13)

# Removing axis
ax3.axis('off')

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



Age	Probability	Group
Young Adults	44%	18 to 25
Adults	41%	26 to 35
Middle Aged	12%	36 to 45
Elder	3%	Above 45

In [ ]:
```
#Insights
# 85% of the customers fall in the age range of 18 to 35. with a median age of 26,
 #Outliers
 # As we can see from the box plot, there are 3 outlier's present in the age data.
```

In [ ]:
```
3.2.2 Customer Education Distribution
```

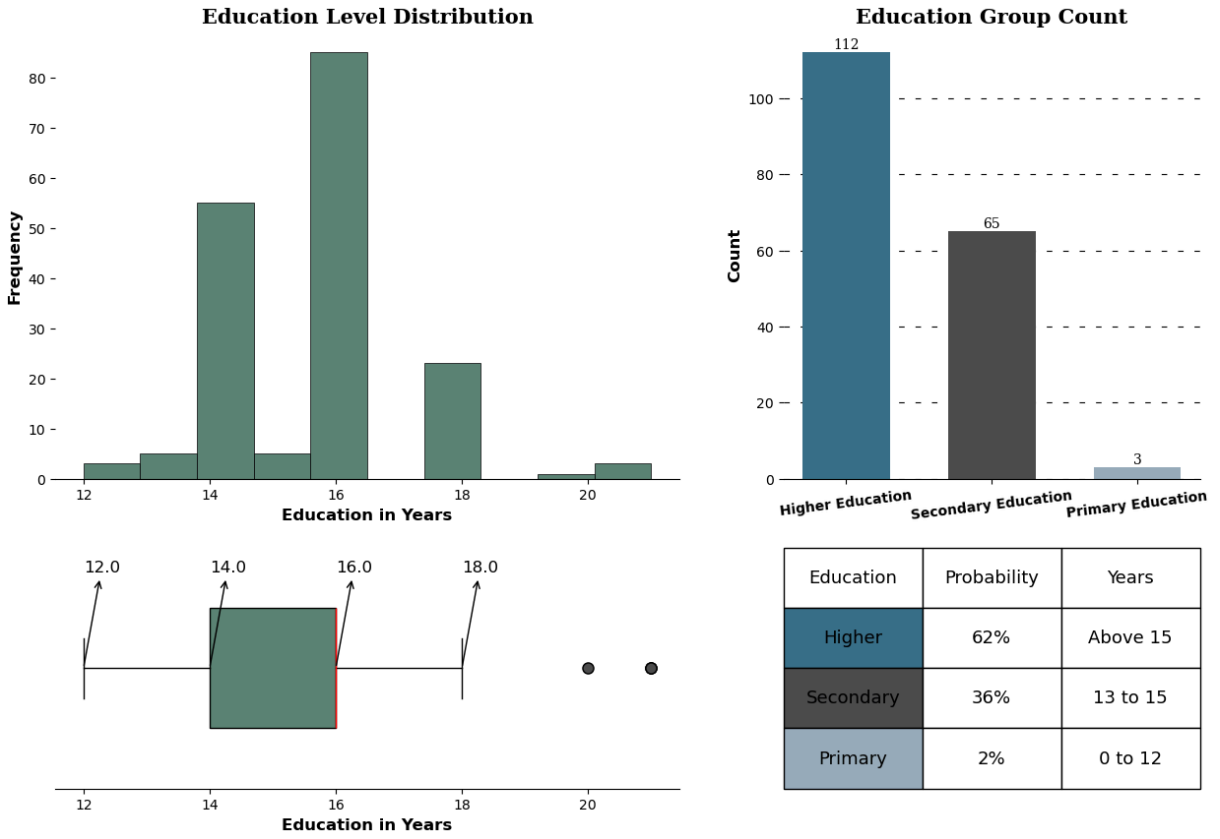
In [143…
```
print(df.columns)
```

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

In [145…
```
# Setting the plot style
fig = plt.figure(figsize=(15, 10))
gs = fig.add_gridspec(2, 2, height_ratios=[0.65, 0.35], width_ratios=[0.6, 0.4])

# Creating education histogram
ax0 = fig.add_subplot(gs[0, 0])
ax0.hist(df['Education'], color='#5C8374', linewidth=0.5, edgecolor='black')
ax0.set_xlabel('Education in Years', fontsize=12, fontweight='bold')
ax0.set_ylabel('Frequency', fontsize=12, fontweight='bold')

# Removing the axis lines
```

```
for s in ['top', 'left', 'right']:
    ax0.spines[s].set_visible(False)

# Setting title for visual
ax0.set_title('Education Level Distribution', {'font': 'serif', 'size': 15, 'weight

# Creating box plot for education
ax1 = fig.add_subplot(gs[1, 0])
boxplot = ax1.boxplot(x=df['Education'], vert=False, patch_artist=True, widths=0.5)

# Customize box and whisker colors
boxplot['boxes'][0].set(facecolor='#5C8374')

# Customize median line
boxplot['medians'][0].set(color='red')

# Customize outlier markers
for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor="#4b4b4c")

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax1.spines[s].set_visible(False)

# Adding 5 point summary annotations
info = [i.get_xdata() for i in boxplot['whiskers']]  # Getting the upperlimit, Q1,
median = df['Education'].quantile(0.5)  # Getting Q2
for i, j in info:  # Using i, j here because of the output type of info list compre
    ax1.annotate(text=f"{i:.1f}", xy=(i, 1), xytext=(i, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"
    ax1.annotate(text=f"{j:.1f}", xy=(j, 1), xytext=(j, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"

# Removing y-axis ticks
ax1.set_yticks([])

# Adding axis label
ax1.set_xlabel('Education in Years', fontweight='bold', fontsize=12)

# Creating education group bar chart
ax2 = fig.add_subplot(gs[0, 1])
temp = df['edu_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c", '#99AEBB']
ax2.bar(x=temp.index, height=temp.values, color=color_map, zorder=2, width=0.6)

# Adding the value_counts
for i in temp.index:
    ax2.text(i, temp[i] + 2, temp[i], {'font': 'serif', 'size': 10}, ha='center', v

# Adding grid lines
ax2.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax2.spines[s].set_visible(False)
```

```
# Adding axis label
ax2.set_ylabel('Count', fontweight='bold', fontsize=12)
ax2.set_xticklabels(temp.index, fontweight='bold', rotation=7)

# Setting title for visual
ax2.set_title('Education Group Count', {'font': 'serif', 'size': 15, 'weight': 'bol

# Creating a table for group info
ax3 = fig.add_subplot(gs[1, 1])
edu_info = [['Higher', '62%', 'Above 15'], ['Secondary', '36%', '13 to 15'], ['Prim
color_2d = [["#3A7089", '#FFFFFF', '#FFFFFF'], ["#4b4b4c", '#FFFFFF', '#FFFFFF'], [
table = ax3.table(cellText=edu_info, cellColours=color_2d, cellLoc='center',
                  colLabels=['Education', 'Probability', 'Years'], colLoc='center',
table.set_fontsize(13)

# Removing axis
ax3.axis('off')

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



Education	Probability	Years
Higher	62%	Above 15
Secondary	36%	13 to 15
Primary	2%	0 to 12

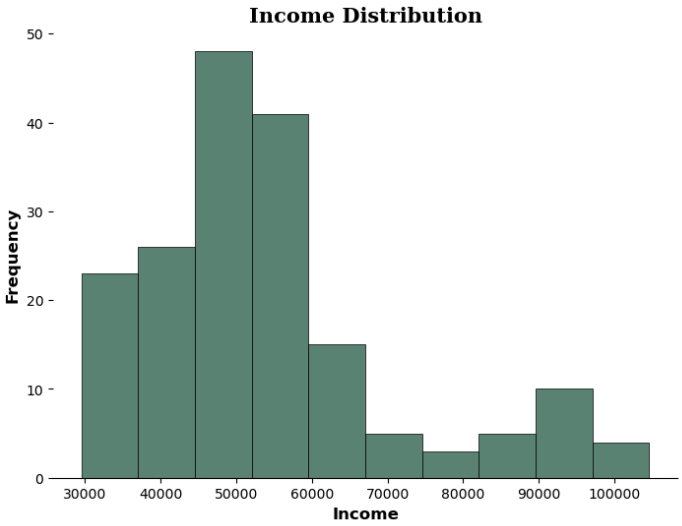
In [ ]:
```
#98% of the customers have education more than 13 years highlighting a strong incli
 #health awareness driven by education could play a pivotal role in this trend.
 #Outliers
 #As we can see from the box plot, there are 2 outlier's present in the education d
```

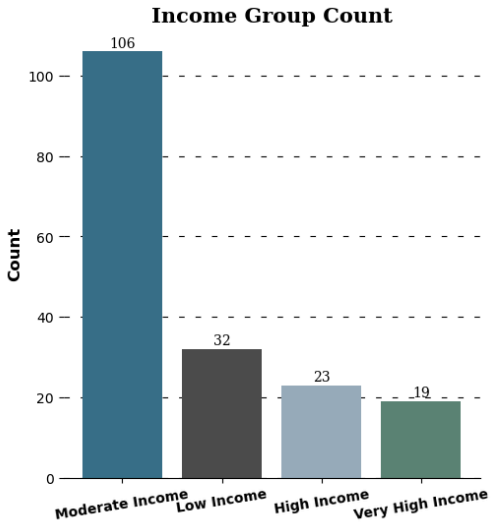
In [ ]:
```
3.2.3-- Customer Income Distribution
```

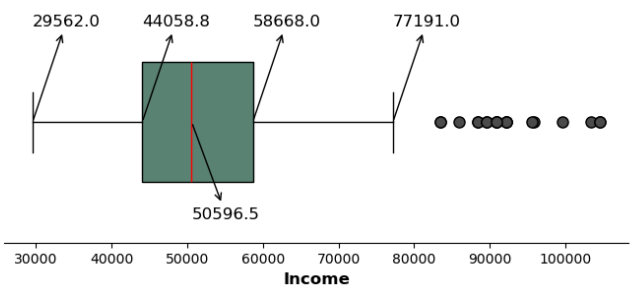
In [147…
```
 #setting the plot style
 fig = plt.figure(figsize = (15,10))
```

```
 gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])
                                    #creating Income histogram

ax0 = fig.add_subplot(gs[0,0])
 ax0.hist(df['Income'],color= '#5C8374',linewidth=0.5,edgecolor='black')
 ax0.set_xlabel('Income',fontsize = 12,fontweight = 'bold')
 ax0.set_ylabel('Frequency',fontsize = 12,fontweight = 'bold')
 #removing the axis lines
 for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#setting title for visual
 ax0.set_title('Income Distribution',{'font':'serif', 'size':15,'weight':'bold'})

                                     #creating box plot for Income

ax1 = fig.add_subplot(gs[1,0])
 boxplot = ax1.boxplot(x = df['Income'],vert = False,patch_artist = True,widths = 0
 # Customize box and whisker colors
 boxplot['boxes'][0].set(facecolor='#5C8374')
 # Customize median line
 boxplot['medians'][0].set(color='red')
 # Customize outlier markers
 for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor= "#4b4b4c")

#removing the axis lines
 for s in ['top','left','right']:
    ax1.spines[s].set_visible(False)
 #adding 5 point summary annotations
 info = [i.get_xdata() for i in boxplot['whiskers']] #getting the upperlimit,Q1,Q3
 median = df['Income'].quantile(0.5) #getting Q2
 for i,j in info: #using i,j here because of the output type of info list comprehen

    ax1.annotate(text = f"{i:.1f}", xy = (i,1), xytext = (i,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0

    ax1.annotate(text = f"{j:.1f}", xy = (j,1), xytext = (j,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0
#adding the median separately because it was included in info list
 ax1.annotate(text = f"{median:.1f}",xy = (median,1),xytext = (median,0.6),fontsize
            arrowprops= dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"))
 #removing y-axis ticks
 ax1.set_yticks([])
 #adding axis label
 ax1.set_xlabel('Income',fontweight = 'bold',fontsize = 12)
                                    #creating Income group bar chart

ax2 = fig.add_subplot(gs[0,1])
 temp = df['income_group'].value_counts()
 color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
 ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)
 #adding the value_counts
 for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va = 'c
 #adding grid lines
```

```
 ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes = (5,10)
 #removing the axis lines
 for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
 ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
 ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 9)
 #setting title for visual
 ax2.set_title('Income Group Count',{'font':'serif', 'size':15,'weight':'bold'})
                                        #creating a table group info

ax3 = fig.add_subplot(gs[1,1])
 inc_info = [['Low','18%','Below 40k'],['Moderate','59%','40k to 60k'],['High','13%
            ['Vey High','10%','Above 80k']]
 color_2d = [["#4b4b4c",'#FFFFFF','#FFFFFF'],["#3A7089",'#FFFFFF','#FFFFFF'],['#99A
            ['#5C8374','#FFFFFF','#FFFFFF']]
 table = ax3.table(cellText = inc_info, cellColours=color_2d, cellLoc='center',
                   colLabels =['Income Grp','Probability','Income($)'],
                   colLoc = 'center',bbox =[0, 0, 1, 1])
 table.set_fontsize(13)
 #removing axis
ax3.axis('off')
 bin_range3 = [0,40000,60000,80000,float('inf')]
 bin_labels3 = ['Low Income','Moderate Income','High Income','Very High Income']
 plt.show()
```







Income Grp	Probability	Income($)
Low	18%	Below 40k
Moderate	59%	40k to 60k
High	13%	60k to 80k
Vey High	10%	Above 80k

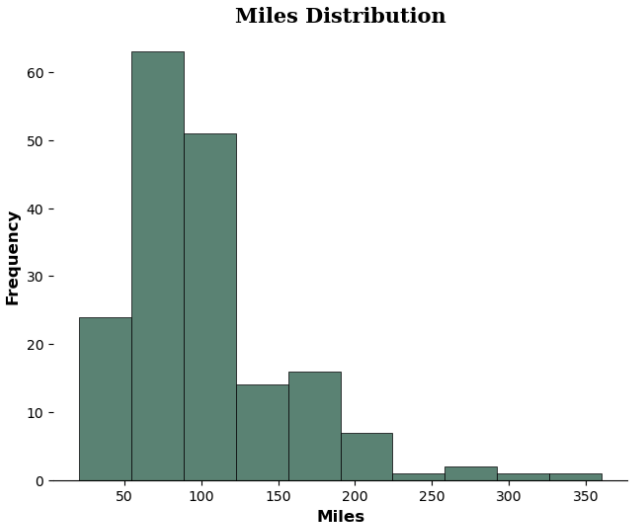
In [ ]:
```
#Almost 60% of the customers fall in the income group of (40k to 60k) dollars sugge
 #Surprisingly 18% of the customers fall in the income group of (<40) suggesting al
 #them falling in 60k and above income group
```

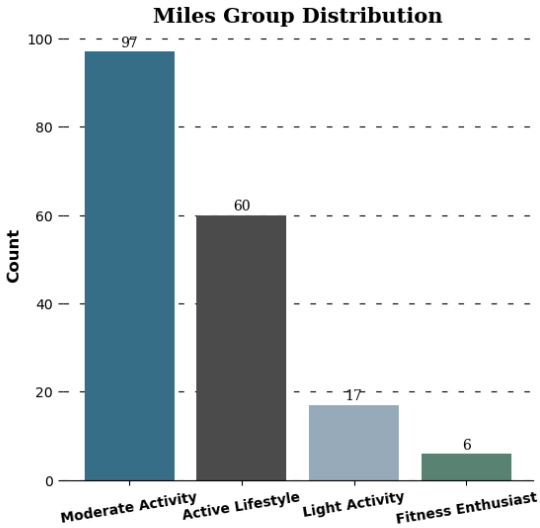
```
# Outliers
 #As we can see from the box plot, there are many outlier's present in the income d
```

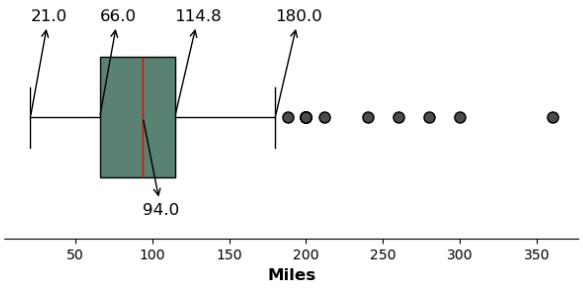
In [ ]:
```
3.2.4  Customers Expected Weekly Mileage
```

In [151…
```
# Setting the plot style
fig = plt.figure(figsize=(15, 10))
gs = fig.add_gridspec(2, 2, height_ratios=[0.65, 0.35], width_ratios=[0.55, 0.45])

# Creating miles histogram
ax0 = fig.add_subplot(gs[0, 0])
ax0.hist(df['Miles'], color='#5C8374', linewidth=0.5, edgecolor='black')
ax0.set_xlabel('Miles', fontsize=12, fontweight='bold')
ax0.set_ylabel('Frequency', fontsize=12, fontweight='bold')

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax0.spines[s].set_visible(False)

# Setting title for visual
ax0.set_title('Miles Distribution', {'font': 'serif', 'size': 15, 'weight': 'bold'}

# Creating box plot for miles
ax1 = fig.add_subplot(gs[1, 0])
boxplot = ax1.boxplot(x=df['Miles'], vert=False, patch_artist=True, widths=0.5)

# Customize box and whisker colors
boxplot['boxes'][0].set(facecolor='#5C8374')

# Customize median line
boxplot['medians'][0].set(color='red')

# Customize outlier markers
for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor="#4b4b4c")

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax1.spines[s].set_visible(False)

# Adding 5-point summary annotations
info = [i.get_xdata() for i in boxplot['whiskers']]  # Getting the upper limit, Q1,
median = df['Miles'].quantile(0.5)  # Getting Q2
for i, j in info:  # Using i, j here because of the output type of info list compre
    ax1.annotate(text=f"{i:.1f}", xy=(i, 1), xytext=(i, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"

    ax1.annotate(text=f"{j:.1f}", xy=(j, 1), xytext=(j, 1.4), fontsize=12,
                 arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"

# Adding the median separately
ax1.annotate(text=f"{median:.1f}", xy=(median, 1), xytext=(median, 0.6), fontsize=1
             arrowprops=dict(arrowstyle="<-", lw=1, connectionstyle="arc,rad=0"))

# Removing y-axis ticks
```

```
ax1.set_yticks([])

# Adding axis label
ax1.set_xlabel('Miles', fontweight='bold', fontsize=12)

# Creating miles group bar chart
ax2 = fig.add_subplot(gs[0, 1])
temp = df['miles_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c", '#99AEBB', '#5C8374']
ax2.bar(x=temp.index, height=temp.values, color=color_map, zorder=2)

# Adding the value counts
for i in temp.index:
    ax2.text(i, temp[i] + 2, temp[i], {'font': 'serif', 'size': 10}, ha='center', v

# Adding grid lines
ax2.grid(color='black', linestyle='--', axis='y', zorder=0, dashes=(5, 10))

# Removing the axis lines
for s in ['top', 'left', 'right']:
    ax2.spines[s].set_visible(False)

# Adding axis label
ax2.set_ylabel('Count', fontweight='bold', fontsize=12)
ax2.set_xticklabels(temp.index, fontweight='bold', rotation=9)

# Setting title for visual
ax2.set_title('Miles Group Distribution', {'font': 'serif', 'size': 15, 'weight': '

# Creating a table for group info
ax3 = fig.add_subplot(gs[1, 1])
miles_info = [['Light Activity', '9%', '0 to 50'], ['Moderate Activity', '54%', '51
              ['Fitness Enthusiast', '3%', 'Above 200']]
color_2d = [['#99AEBB', '#FFFFFF', '#FFFFFF'], ["#3A7089", '#FFFFFF', '#FFFFFF'], [
            ['#5C8374', '#FFFFFF', '#FFFFFF']]
table = ax3.table(cellText=miles_info, cellColours=color_2d, cellLoc='center', colL
                  colLoc='center', bbox=[0, 0, 1, 1])
table.set_fontsize(11)

# Removing axis
ax3.axis('off')

plt.show()
```





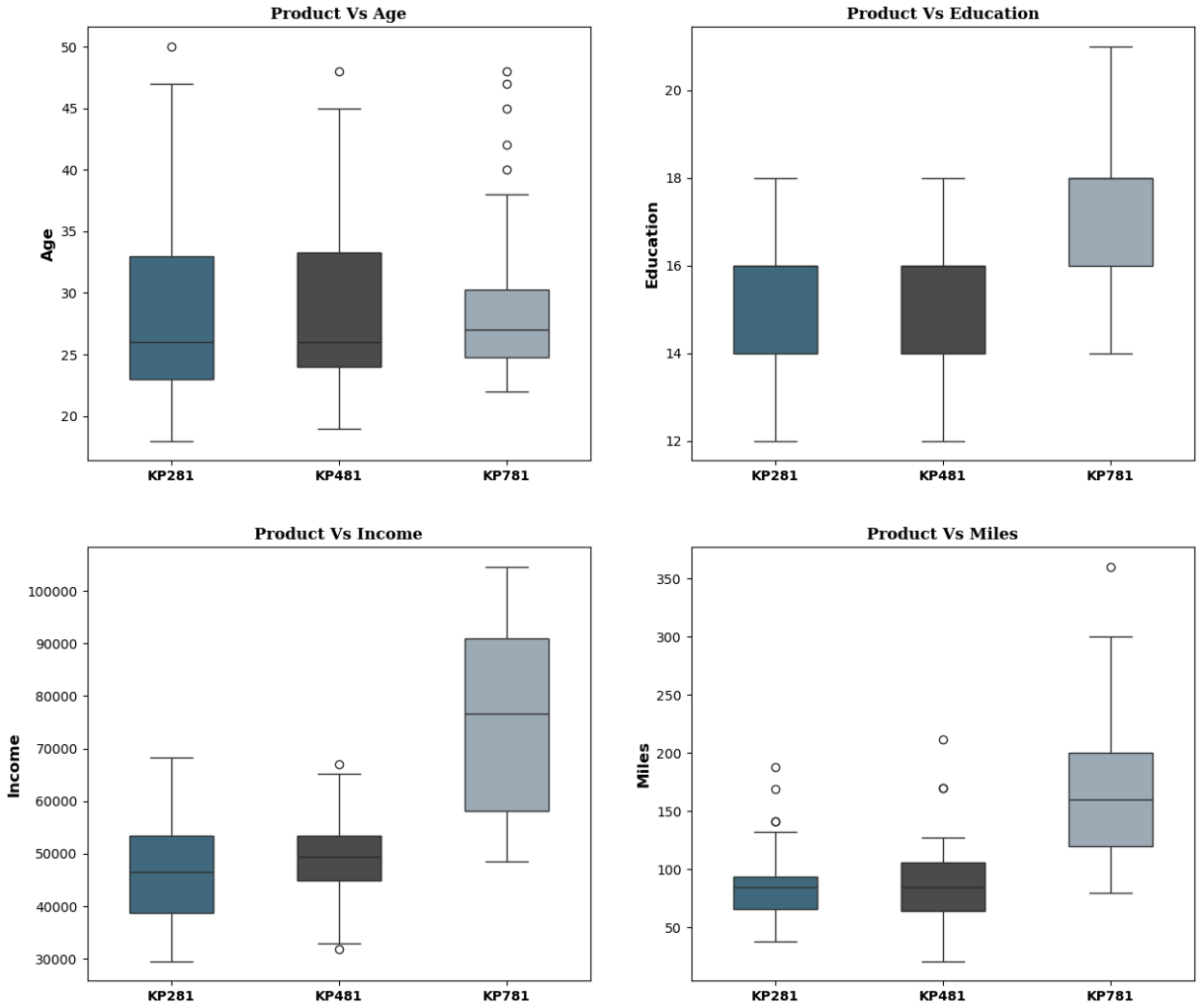


Activity	Probability	Miles
Light Activity	9%	0 to 50
Moderate Activity	54%	51 to 100
Active Lifestyle	34%	101 to 200
Fitness Enthusiast	3%	Above 200

In [ ]:
```
Almost 88% of the customers plans to use the treadmill for 50 to 200 miles per week
 Outliers
 As we can see from the box plot, there are 8 outlier's present in the miles data.
```

In [ ]:
```
4. Bivariate Analysis
4.1- Analysis of Product Type
```

In [153…
```
 #setting the plot style
 fig = plt.figure(figsize = (15,13))
 gs = fig.add_gridspec(2,2)
 for i,j,k in [(0,0,'Age'),(0,1,'Education'),(1,0,'Income'),(1,1,'Miles')]:

    #plot position
    ax0 = fig.add_subplot(gs[i,j])
    #plot
    sns.boxplot(data = df, x = 'Product', y  = k ,ax = ax0,width = 0.5, palette =["
    #plot title
    ax0.set_title(f'Product Vs {k}',{'font':'serif', 'size':12,'weight':'bold'})

    #customizing axis
    ax0.set_xticklabels(df['Product'].unique(),fontweight = 'bold')
    ax0.set_ylabel(f'{k}',fontweight = 'bold',fontsize = 12)
    ax0.set_xlabel('')

plt.show()
```

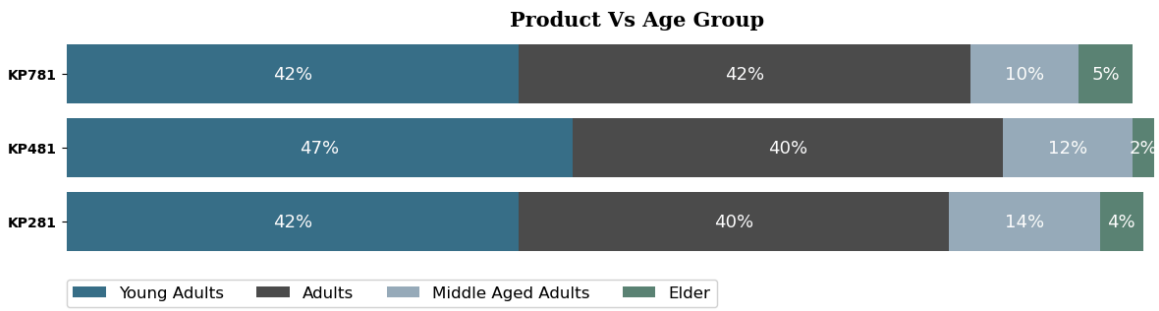


In [ ]:
```
The analysis presented above clearly indicates a strong preference for the treadmi
levels, and intend to engage in running activities exceeding 150 miles per week.
```

In [ ]:
```
4.2 Product Preferences Across Age
```

In [195…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Example DataFrame (df) setup (if needed to recreate)
# df = pd.DataFrame({
#     'Product': ['A', 'A', 'B', 'B', 'C', 'C'],
#     'age_group': ['20-30', '30-40', '20-30', '40-50', '30-40', '40-50']
# })

# Setting the plot style
fig, ax0 = plt.subplots(figsize=(15, 3))

# Product vs age group
val = 'age_group'

# Creating required df
df_grp = df.groupby('Product')[val].value_counts(normalize=True).round(2).reset_ind
```

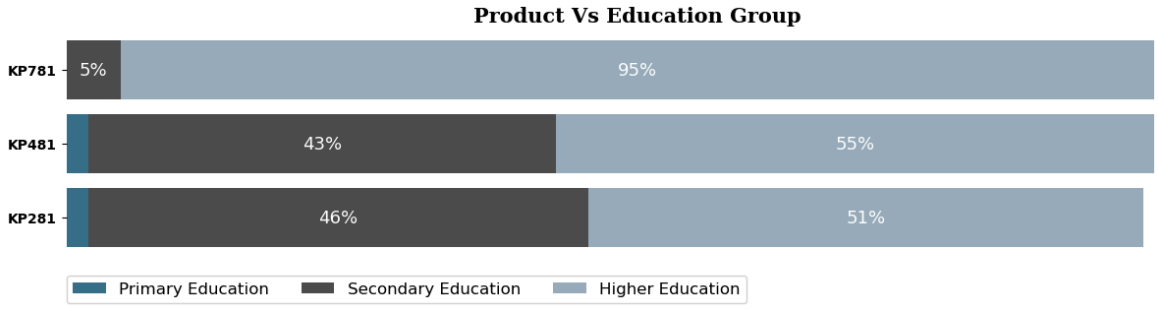
```
# Pivoting the DataFrame correctly
df_grp = df_grp.pivot(index='Product', columns=val, values='percentage').fillna(0)

# For left parameter in ax.barh
temp = np.zeros(len(df_grp), dtype=float)
color_map = ["#3A7089", "#4b4b4c", '#99AEBB', '#5C8374']

# Plotting the visual
for i, j in zip(df_grp.columns, color_map):
    ax0.barh(df_grp.index, width=df_grp[i], left=temp, label=i, color=j)
    temp += df_grp[i].values

# Inserting text
temp = np.zeros(len(df_grp), dtype=float)
for i in df_grp.columns:
    for j, k in enumerate(df_grp[i]):
        if k == 0:
            continue
        # Calculate the position for text
        posx = k / 2 + temp[j]

        # Check if posx and posy are finite values
        if np.isfinite(posx):
            ax0.text(posx, j, f"{k:.0%}", va='center', ha='center', fontsize=13, co

    temp += df_grp[i].values

# Removing the axis lines
for s in ['top', 'left', 'right', 'bottom']:
    ax0.spines[s].set_visible(False)

# Customizing ticks
ax0.set_xticks([])
ax0.set_yticklabels(df_grp.index, fontweight='bold')

# Plot title
ax0.set_title('Product Vs Age Group', {'font': 'serif', 'size': 15, 'weight': 'bold

# Adding legend
ax0.legend(loc=(0, -0.2), ncol=4, fontsize=12)

plt.show()
```



In [ ]:
```
The analysis provided above distinctly demonstrates that there exists no strong cor
 uniform distribution of age groups across all the products.
```

In [ ]:
```
4.3 Product Preferences Across Education Levels
```

In [201…
```
# Setting the plot style
fig, ax0 = plt.subplots(figsize=(15, 3))

# Product vs edu group
val = 'edu_group'

# Creating required df
df_grp = df.groupby('Product')[val].value_counts(normalize=True).round(2).reset_ind

# Pivoting the DataFrame correctly
df_grp = df_grp.pivot(index='Product', columns=val, values='percentage').fillna(0)

# For left parameter in ax.barh
temp = np.zeros(len(df_grp), dtype=float)
color_map = ["#3A7089", "#4b4b4c", '#99AEBB']

# Plotting the visual
for i, j in zip(df_grp.columns, color_map):
    ax0.barh(df_grp.index, width=df_grp[i], left=temp, label=i, color=j)
    temp += df_grp[i].values

# Inserting text
temp = np.zeros(len(df_grp), dtype=float)
for i in df_grp.columns:
    for j, k in enumerate(df_grp[i]):
        if k < 0.05:  # Skip small values
            continue
        posx = k / 2 + temp[j]  # Calculate text position
        if np.isfinite(posx):  # Check if posx is finite
            ax0.text(posx, j, f"{k:.0%}", va='center', ha='center', fontsize=13, co
    temp += df_grp[i].values

# Removing the axis lines
for s in ['top', 'left', 'right', 'bottom']:
    ax0.spines[s].set_visible(False)

# Customizing ticks
ax0.set_xticks([])
ax0.set_yticklabels(df_grp.index, fontweight='bold')

# Plot title
ax0.set_title('Product Vs Education Group', {'font': 'serif', 'size': 15, 'weight':

# Adding legend
ax0.legend(loc=(0, -0.2), ncol=4, fontsize=12)

plt.show()
```

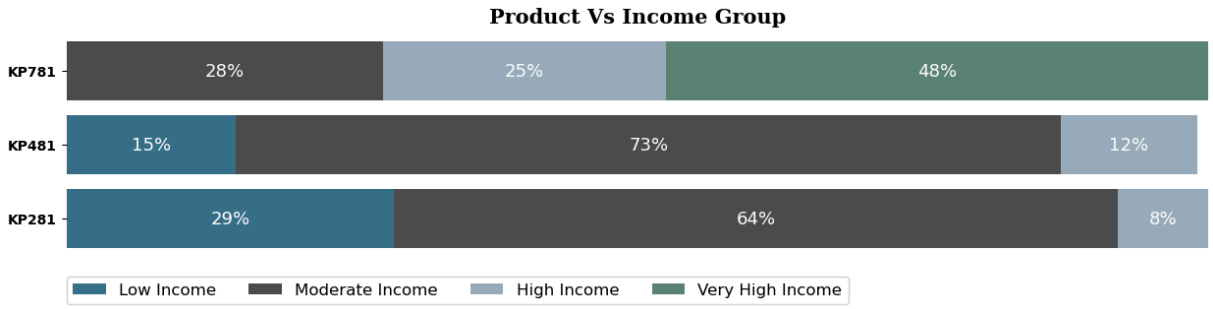


In [ ]:
```
The analysis provided above cleary demonstrates the preference of Highly Educated p
 For treadmill models KP481 and KP281, the distribution of customer with Secondary
```

In [ ]:
```
4.4 Product Preference Across Income Group
```

In [205…
```
# Setting the plot style
fig, ax0 = plt.subplots(figsize=(15, 3))

# Product vs income group
val = 'income_group'

# Creating required df
df_grp = df.groupby('Product')[val].value_counts(normalize=True).round(2).reset_ind

# Pivoting the DataFrame correctly
df_grp = df_grp.pivot(index='Product', columns=val, values='percentage').fillna(0)

# For left parameter in ax.barh
temp = np.zeros(len(df_grp), dtype=float)
color_map = ["#3A7089", "#4b4b4c", '#99AEBB', '#5C8374']

# Plotting the visual
for i, j in zip(df_grp.columns, color_map):
    ax0.barh(df_grp.index, width=df_grp[i], left=temp, label=i, color=j)
    temp += df_grp[i].values

# Inserting text
temp = np.zeros(len(df_grp), dtype=float)
for i in df_grp.columns:
    for j, k in enumerate(df_grp[i]):
        if k < 0.05:
            continue
        posx = k / 2 + temp[j]  # Calculate text position
        if np.isfinite(posx):  # Check if posx is finite
            ax0.text(posx, j, f"{k:.0%}", va='center', ha='center', fontsize=13, co
    temp += df_grp[i].values

# Removing the axis lines
for s in ['top', 'left', 'right', 'bottom']:
    ax0.spines[s].set_visible(False)

# Customizing ticks
ax0.set_xticks([])
ax0.set_yticklabels(df_grp.index, fontweight='bold')
```

```
# Plot title
ax0.set_title('Product Vs Income Group', {'font': 'serif', 'size': 15, 'weight': 'b

# Adding legend
ax0.legend(loc=(0, -0.2), ncol=4, fontsize=12)

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

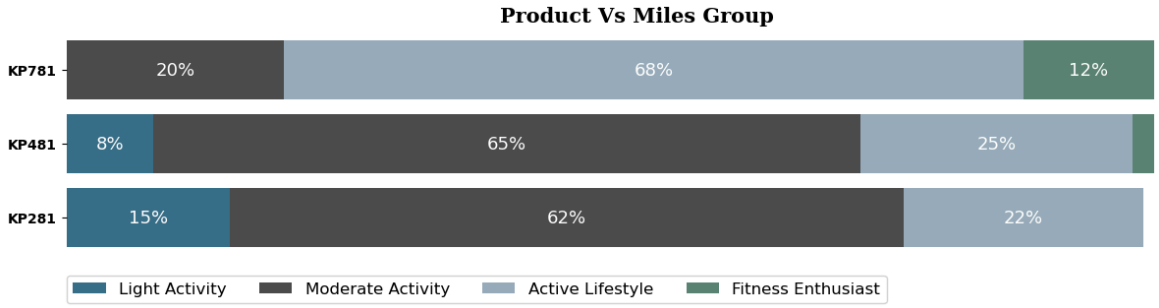


In [ ]:
```
Treadmill model KP781 is preferred more by customers with Very High Income
 Both treadmill models, KP481 and KP281, are preferred more by customers with Moder
```

In [ ]:
```
4.5 Product preference across customer weekly mileage
```

In [207…
```
 #setting the plot style
 fig,ax0 = plt.subplots(figsize = (15,3))
                                                    #product vs miles group
 val = 'miles_group'
 # Creating required df
df_grp = df.groupby('Product')[val].value_counts(normalize=True).round(2).reset_ind

# Pivoting the DataFrame correctly
df_grp = df_grp.pivot(index='Product', columns=val, values='percentage').fillna(0)
 #for left parameter in ax.barh
 temp = np.zeros(len(df_grp),dtype = float)
 color_map =  ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
 #plotting the visual
 for i,j in zip(df_grp.columns,color_map):
    ax0.barh(df_grp.index,width = df_grp[i],left = temp, label = i,color = j)
    temp += df_grp[i].values
 #inserting text
 temp = np.zeros(len(df_grp),dtype = float)
 for i in df_grp.columns:
    for j,k in enumerate(df_grp[i]):
        if k < 0.05:
             continue
        posx = k / 2 + temp[j]  # Calculate text position
        if np.isfinite(posx):  # Check if posx is finite
            ax0.text(posx, j, f"{k:.0%}", va='center', ha='center', fontsize=13, co
    temp += df_grp[i].values
 #removing the axis lines
 for s in ['top','left','right','bottom']:
    ax0.spines[s].set_visible(False)

#customizing ticks
```

```
ax0.set_xticks([])
ax0.set_yticklabels(df_grp.index,fontweight = 'bold')
#plot title
ax0.set_title('Product Vs Miles Group',{'font':'serif', 'size':15,'weight':'bold'}
#adding legend
ax0.legend(loc = (0,-0.2),ncol = 4,fontsize = 12)
plt.show()
```

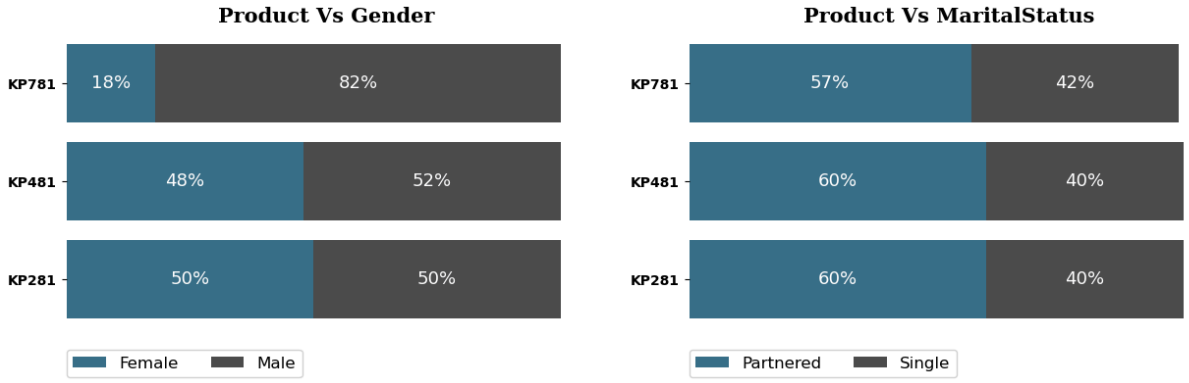


In [ ]:
```
Treadmill model KP781 is preferred more by customers planning to run 100 to 200 mil
 Both treadmill models, KP481 and KP281, are preferred more by customers planning t
```

In [ ]:
```
4.6 Product Preference across Gender and Martial Status
```

In [211…
```
# Setting the plot style
fig = plt.figure(figsize=(15, 4))
gs = fig.add_gridspec(1, 2)

for r, c, val in [(0, 0, 'Gender'), (0, 1, 'MaritalStatus')]:
    ax0 = fig.add_subplot(gs[r, c])

    # Creating required df
    df_grp = df.groupby('Product')[val].value_counts(normalize=True).round(2)
    df_grp.name = 'count'
    df_grp = df_grp.reset_index()
    df_grp = df_grp.pivot(columns=val, index='Product', values='count')

    # For left parameter in ax.barh
    temp = np.zeros(len(df_grp), dtype=float)
    color_map = ["#3A7089", "#4b4b4c"]

    # Plotting the visual
    for i, j in zip(df_grp.columns, color_map):
        ax0.barh(df_grp.index, width=df_grp[i], left=temp, label=i, color=j)
        temp += df_grp[i].values

    # Inserting text
    temp = np.zeros(len(df_grp), dtype=float)
    for i in df_grp.columns:
        for j, k in enumerate(df_grp[i]):
            if k < 0.05:
                continue
            ax0.text(k / 2 + temp[j], df_grp.index[j], f"{k:.0%}",
                     va='center', ha='center', fontsize=13, color='white')
        temp += df_grp[i].values
```

```
    # Removing the axis lines
    for s in ['top', 'left', 'right', 'bottom']:
        ax0.spines[s].set_visible(False)

    # Customizing ticks
    ax0.set_xticks([])
    ax0.set_yticklabels(df_grp.index, fontweight='bold')

    # Plot title
    ax0.set_title(f'Product Vs {val}', {'font': 'serif', 'size': 15, 'weight': 'bol

    # Adding legend
    ax0.legend(loc=(0, -0.15), ncol=2, fontsize=12)

plt.show()
```

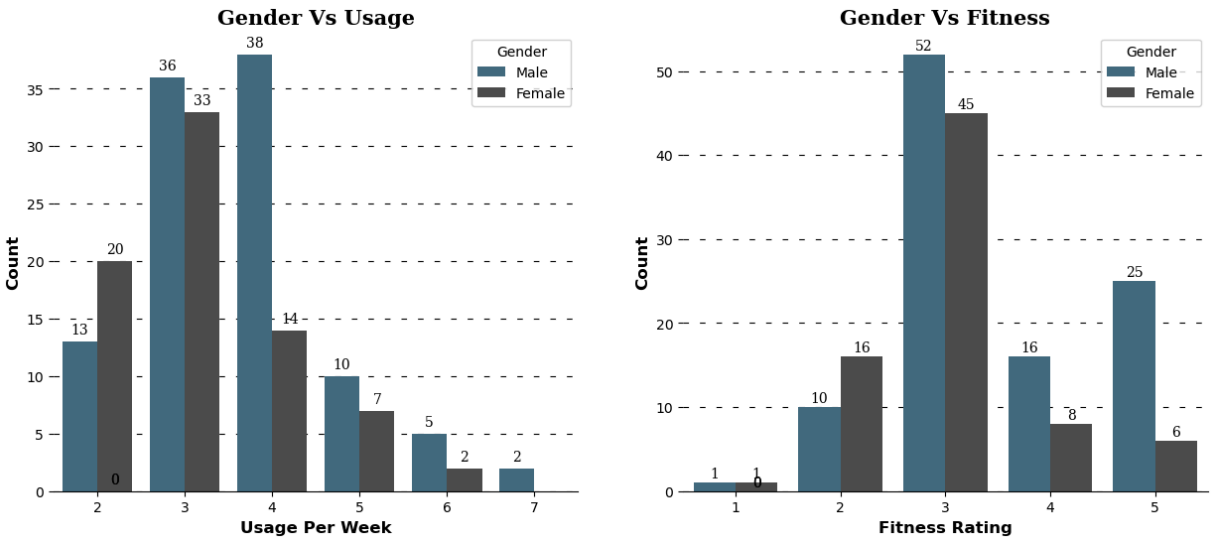


In [ ]:
```
1. Gender
 Treadmill model KP781 is preferred more by male customers.
 Both treadmill models, KP481 and KP281, show equal distribution of both the gender
 2. Marital Status
 For all three treadmill models, there is a uniform distribution of Married and Sin
preference
```

In [ ]:
```
4.7 Gender vs Product Usage And Gender Vs Fitness
```

In [213…
```
 #setting the plot style
 fig = plt.figure(figsize = (15,6))
 gs = fig.add_gridspec(1,2)
                                        # Usage Vs Gender
#creating bar plot
 ax1 = fig.add_subplot(gs[0,0])
 plot = sns.countplot(data = df, x = 'Usage', hue = 'Gender',order = sorted(df['Usa
              ax = ax1,palette = ["#3A7089","#4b4b4c"],zorder = 2)
 #adding the value_counts
 for i in plot.patches:
    ax1.text(i.get_x()+0.2,i.get_height()+1,f'{i.get_height():.0f}',{'font':'serif'
 #adding grid lines
 ax1.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes = (5,10)
 #removing the axis lines
 for s in ['top','left','right']:
    ax1.spines[s].set_visible(False)

#adding axis label
```

```
 ax1.set_xlabel('Usage Per Week',fontweight = 'bold',fontsize = 12)
 ax1.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
 #setting title for visual
 ax1.set_title('Gender Vs Usage',{'font':'serif', 'size':15,'weight':'bold'})
                                        # Fitness Vs Gender
 #creating bar plot
 ax2 = fig.add_subplot(gs[0,1])
 plot = sns.countplot(data = df, x = 'Fitness', hue = 'Gender',order = sorted(df['F
             ax = ax2,palette = ["#3A7089","#4b4b4c"],zorder = 2)
 #adding the value_counts
 for i in plot.patches:
    ax2.text(i.get_x()+0.2,i.get_height()+1,f'{i.get_height():.0f}',{'font':'serif'
 #adding grid lines
 ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes = (5,10)
 #removing the axis lines
 for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#customizing axis labels
 ax2.set_xlabel('Fitness Rating',fontweight = 'bold',fontsize = 12)
 ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
 #setting title for visual
 ax2.set_title('Gender Vs Fitness',{'font':'serif', 'size':15,'weight':'bold'})
plt.show()
```

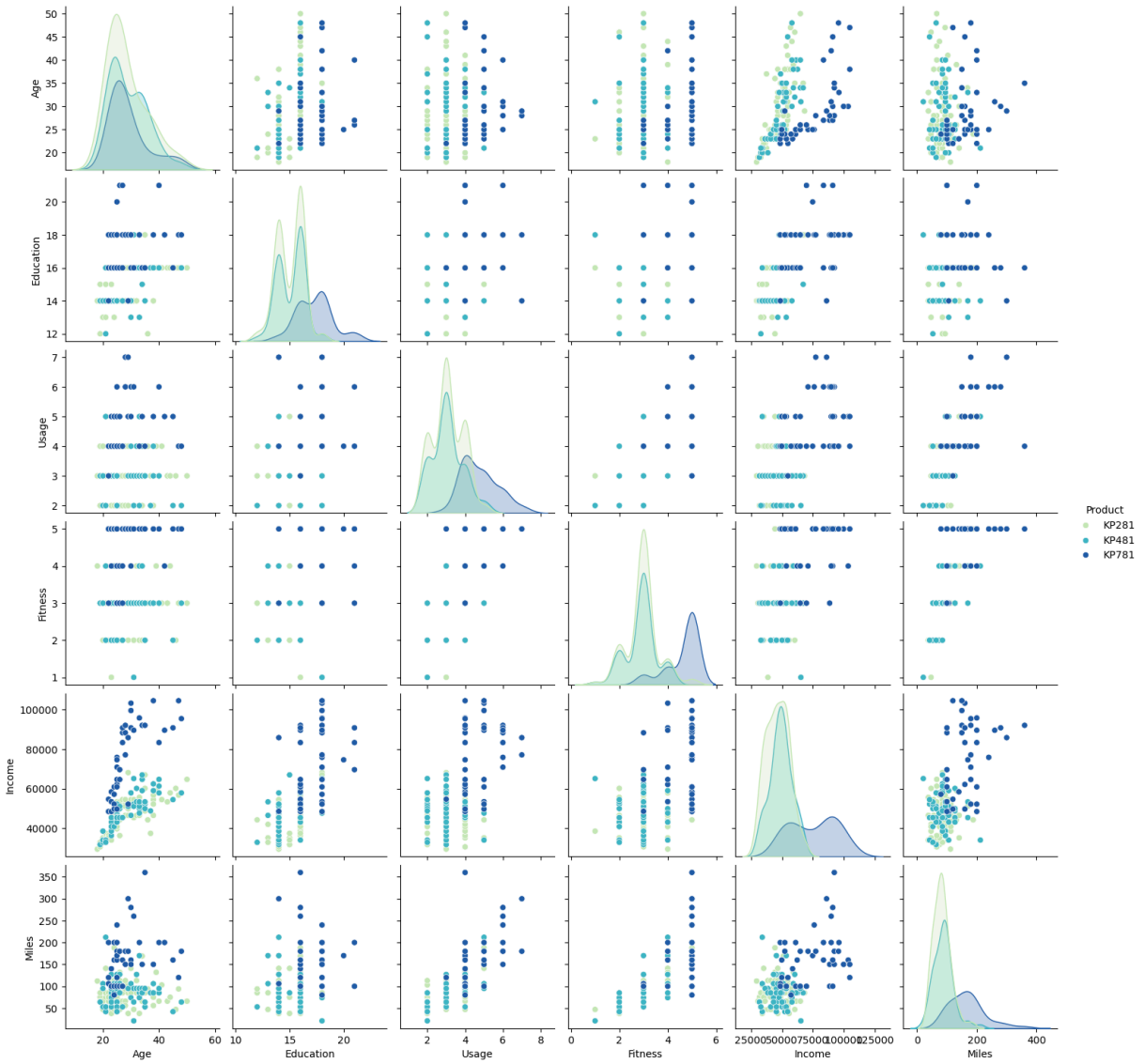


In [ ]:
```
1. Gender Vs Usage
 Almost 70% of Female customers plan to use the treadmill for 2 to 3 times a week w
 3 to 4 times a week
 2. Gender Vs Fitness
 Almost 80% of Female customers rated themselves between 2 to 3 whereas almost 90%
 fitness scale
```

In [ ]:
```
5. Correlation between Variables
5.1 Pairplot
```

In [221…
```
import copy
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Create a deep copy of the DataFrame
df_copy = copy.deepcopy(df)

# Create a pairplot
sns.pairplot(df_copy, hue='Product', palette='YlGnBu')

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



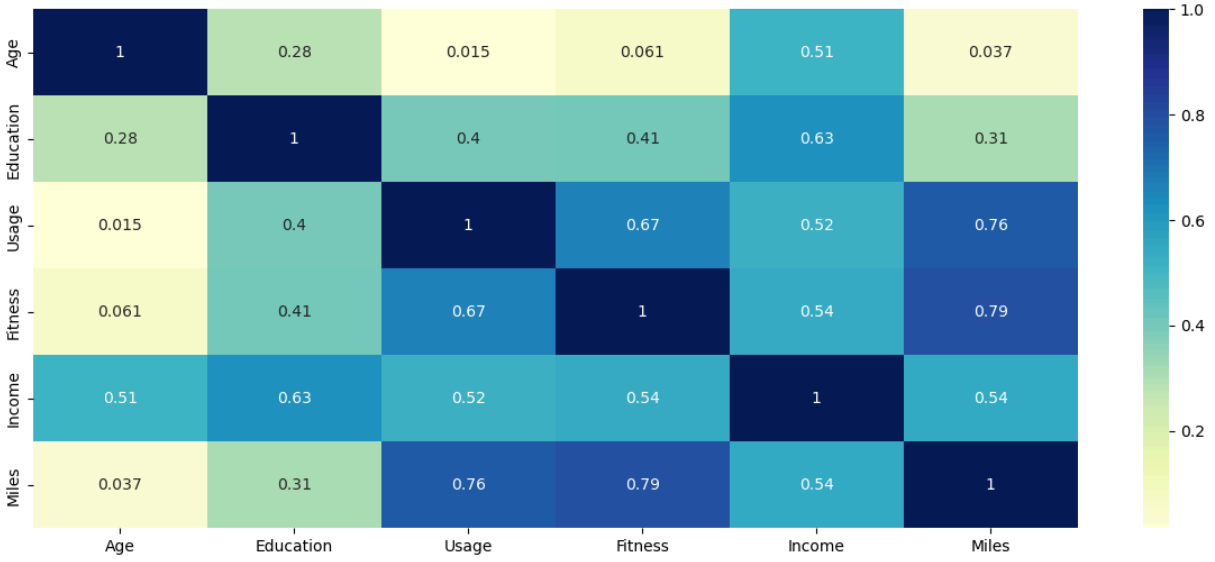
In [ ]:
```
5.2 Heatmap
```

In [223…
```
 # First we need to convert object into int datatype for usage and fitness columns
df_copy['Usage'] = df_copy['Usage'].astype('int')
 df_copy['Fitness'] = df_copy['Fitness'].astype('int')
 df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 13 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    int32   
 6   Fitness        180 non-null    int32   
 7   Income         180 non-null    int64   
 8   Miles          180 non-null    int64   
 9   income_group   180 non-null    category
 10  miles_group    180 non-null    category
 11  age_group      180 non-null    category
 12  edu_group      180 non-null    category
dtypes: category(4), int32(2), int64(4), object(3)
memory usage: 12.8+ KB
```

In [233…

```
# Select only numeric columns for correlation
numeric_df = df_copy.select_dtypes(include=[np.number])

# Compute the correlation matrix
corr_mat = numeric_df.corr()

# Create a heatmap to visualize correlations
plt.figure(figsize=(15, 6))
sns.heatmap(corr_mat, annot=True, cmap="YlGnBu")
plt.show()
```



In [ ]:

```
From the pair plot we can see Age and Income are positively correlated and heatmap
 Eductaion and Income are highly correlated as its obvious. Eductation also has sig
treadmill.
 Usage is highly correlated with Fitness and Miles as more the usage more the fitne
```

In [ ]:
```
6. Computing Probability - Marginal, Conditional Probability
6.1 Probability of product purchase w.r.t. gender
```

In [235…
```
pd.crosstab(index =df['Product'],columns = df['Gender'],margins = True,normalize =
```

Out[235…

Gender	Female	Male	All
Product			
KP281	0.22	0.22	0.44
KP481	0.16	0.17	0.33
KP781	0.04	0.18	0.22
All	0.42	0.58	1.00

In [ ]:
```
1. The Probability of a treadmill being purchased by a female is 42%.
 The conditional probability of purchasing the treadmill model given that the custo
 For Treadmill model KP281 - 22%
 For Treadmill model KP481 - 16%
 For Treadmill model KP781 - 4%
 1. The Probability of a treadmill being purchased by a male is 58%.
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 22%
 For Treadmill model KP481 - 17%
 For Treadmill model KP781 - 18%
```

In [ ]:
```
 6.2 Probability of product purchase w.r.t. Age
```

In [237…
```
pd.crosstab(index =df['Product'],columns = df['age_group'],margins = True,normalize
```

Out[237…

age_group	Young Adults	Adults	Middle Aged Adults	Elder	All
Product					
KP281	0.19	0.18	0.06	0.02	0.44
KP481	0.16	0.13	0.04	0.01	0.33
KP781	0.09	0.09	0.02	0.01	0.22
All	0.44	0.41	0.12	0.03	1.00

In [ ]:
```
 1.The Probability of a treadmill being purchased by a Young Adult(18-25) is 44%.
 The conditional probability of purchasing the treadmill model given that the custo
 For Treadmill model KP281 - 19%
 For Treadmill model KP481 - 16%
 For Treadmill model KP781 - 9%
 2. The Probability of a treadmill being purchased by an Adult(26-35) is 41%.
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 18%
 For Treadmill model KP481 - 13%
 For Treadmill model KP781 - 9%
```

```
 3. The probability of a treadmill being purchased by a middle age (36-45) is 12%.
 4. The Probability of a treadmill being purchased by an Elder(Above 45) is only 3%
```

In [ ]:
```
 6.3 Probability of product purchase w.r.t. Education level
```

In [239…
```
pd.crosstab(index =df['Product'],columns = df['edu_group'],margins = True,normalize
```

Out[239…

Product	Primary Education	Secondary Education	Higher Education	All
KP281	0.01	0.21	0.23	0.44
KP481	0.01	0.14	0.18	0.33
KP781	0.00	0.01	0.21	0.22
All	0.02	0.36	0.62	1.00

In [ ]:
```
1. The Probability of a treadmill being purchased by a customer with Higher Educati
 The conditional probability of purchasing the treadmill model given that the custo
 For Treadmill model KP281 - 23%
 For Treadmill model KP481 - 18%
 For Treadmill model KP781 - 21%
 2. The Probability of a treadmill being purchased by a customer with Secondary Edu
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 21%
 For Treadmill model KP481 - 14%
 For Treadmill model KP781 - 1%
 3. The Probability of a treadmill being purchased by a customer with Primary Educa
```

In [ ]:
```
6.4  Probability of product purchase w.r.t. Income
```

In [241…
```
pd.crosstab(index =df['Product'],columns = df['income_group'],margins = True,normal
```

Out[241…

Product	Low Income	Moderate Income	High Income	Very High Income	All
KP281	0.13	0.28	0.03	0.00	0.44
KP481	0.05	0.24	0.04	0.00	0.33
KP781	0.00	0.06	0.06	0.11	0.22
All	0.18	0.59	0.13	0.11	1.00

In [ ]:
```
 1.The Probability of a treadmill being purchased by a customer with Low Income(<40
 2.The conditional probability of purchasing the treadmill model given that the cus
For Treadmill model KP281 - 13%
 For Treadmill model KP481 - 5%
 For Treadmill model KP781 - 0%
 3. The Probability of a treadmill being purchased by a customer with Moderate Inco
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 28%
```

```
 For Treadmill model KP481 - 24%
 For Treadmill model KP781 - 6%
 4. The Probability of a treadmill being purchased by a customer with High Income(6
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 3%
 For Treadmill model KP481 - 4%
 For Treadmill model KP781 - 6%
 5. The Probability of a treadmill being purchased by a customer with Very High Inc
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 0%
 For Treadmill model KP481 - 0%
 For Treadmill model KP781 - 11%
```

In [ ]:
```
6.5  Probability of product purchase w.r.t. Marital Status
```

In [243…
```
pd.crosstab(index =df['Product'],columns = df['MaritalStatus'],margins = True,norma
```

Out[243…

Product	Partnered	Single	All
KP281	0.27	0.18	0.44
KP481	0.20	0.13	0.33
KP781	0.13	0.09	0.22
All	0.59	0.41	1.00

In [ ]:
```
The Probability of a treadmill being purchased by a Married Customer is 59%.
 The conditional probability of purchasing the treadmill model given that the custo
 For Treadmill model KP281 - 27%
 For Treadmill model KP481 - 20%
 For Treadmill model KP781 - 13%
 1. The Probability of a treadmill being purchased by a Unmarried Customer is 41%.
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 18%
 For Treadmill model KP481 - 13%
 For Treadmill model KP781 - 9%
```

In [ ]:
```
6.6 Probability of product purchase w.r.t. Weekly Usage
```

In [245…
```
pd.crosstab(index =df['Product'],columns = df['Usage'],margins = True,normalize = T
```

Out[245…

Usage	2	3	4	5	6	7	All
Product							
KP281	0.11	0.21	0.12	0.01	0.00	0.00	0.44
KP481	0.08	0.17	0.07	0.02	0.00	0.00	0.33
KP781	0.00	0.01	0.10	0.07	0.04	0.01	0.22
All	0.18	0.38	0.29	0.09	0.04	0.01	1.00

In [ ]:
```
 1.The Probability of a treadmill being purchased by a customer with Usage 3 per we
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 21%
 For Treadmill model KP481 - 17%
 For Treadmill model KP781 - 1%
 2. The Probability of a treadmill being purchased by a customer with Usage 4 per w
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 12%
 For Treadmill model KP481 - 7%
 For Treadmill model KP781 - 10%
 3. The Probability of a treadmill being purchased by a customer with Usage 2 per w
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 11%
 For Treadmill model KP481 - 8%
 For Treadmill model KP781 - 0%
```

In [ ]:
```
6.7 Probability of product purchase w.r.t. Weekly Usage
```

In [249…
```
pd.crosstab(index =df['Product'],columns = df['Fitness'],margins = True,normalize =
```

Out[249…

Fitness	1	2	3	4	5	All
Product						
KP281	0.01	0.08	0.30	0.05	0.01	0.44
KP481	0.01	0.07	0.22	0.04	0.00	0.33
KP781	0.00	0.00	0.02	0.04	0.16	0.22
All	0.01	0.14	0.54	0.13	0.17	1.00

In [ ]:
```
1.The Probability of a treadmill being purchased by a customer with Average(3) Fitn
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 30%
 For Treadmill model KP481 - 22%
 For Treadmill model KP781 - 2%
 2. The Probability of a treadmill being purchased by a customer with Fitness of 2,
 3. The Probability of a treadmill being purchased by a customer with very low(1) F
```

In [ ]:
```
6.8 Probability of product purchase w.r.t. weekly mileage
```

In [251…
```
pd.crosstab(index =df['Product'],columns = df['miles_group'],margins = True,normal
```

Out[251…

Product	miles_group	Light Activity	Moderate Activity	Active Lifestyle	Fitness Enthusiast	All
KP281		0.07	0.28	0.10	0.00	0.44
KP481		0.03	0.22	0.08	0.01	0.33
KP781		0.00	0.04	0.15	0.03	0.22
All		0.09	0.54	0.33	0.03	1.00

In [ ]:
```
1. The Probability of a treadmill being purchased by a customer with lifestyle of L
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 7%
 For Treadmill model KP481 - 3%
 For Treadmill model KP781 - 0%
 2. The Probability of a treadmill being purchased by a customer with lifestyle of
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 28%
 For Treadmill model KP481 - 22%
 For Treadmill model KP781 - 4%
 3. The Probability of a treadmill being purchased by a customer has Active Lifesty
 The conditional probability of purchasing the treadmill model given that the custo
For Treadmill model KP281 - 10%
 For Treadmill model KP481 - 8%
 For Treadmill model KP781 - 15%
 4. The Probability of a treadmill being purchased by a customer who is Fitness Ent
```

In [ ]:
```
7 . Customer Profiling
Based on above analysis
 Probability of purchase of KP281 = 44%
 Probability of purchase of KP481 = 33%
 Probability of purchase of KP781 = 22%
 Customer Profile for KP281 Treadmill:
 Age of customer mainly between 18 to 35 years with few between 35 to 50 years
 Education level of customer 13 years and above
 Annual Income of customer below USD 60,000
 Weekly Usage - 2 to 4 times
 Fitness Scale - 2 to 4
 Weekly Running Mileage - 50 to 100 miles
 Customer Profile for KP481 Treadmill:
 Age of customer mainly between 18 to 35 years with few between 35 to 50 years
 Education level of customer 13 years and above
 Annual Income of customer between USD 40,000 to USD 80,000
 Weekly Usage - 2 to 4 times
 Fitness Scale - 2 to 4
 Weekly Running Mileage - 50 to 200 miles
 Customer Profile for KP781 Treadmill:
 Gender - Male
 Age of customer between 18 to 35 years
 Education level of customer 15 years and above
 Annual Income of customer USD 80,000 and above
```

```
 Weekly Usage - 4 to 7 times
 Fitness Scale - 3 to 5
 Weekly Running Mileage - 100 miles and above
```

In [ ]:
```
8. Recommendations
 Marketing Campaigns for KP781
 The KP784 model exhibits a significant sales disparity in terms of gender, with on
To enhance this metric, it is  recommended to implement targeted strategies such as
 Affordable Pricing and Payment Plans
 Given the target customer's age, education level, and income, it's important to of
Additionally, consider providing flexible payment plans that allow customers to spr
 budgets.
 User-Friendly App Integration
 Create a user-friendly app that syncs with the treadmill. This app could track use
 personalized recommendations for workouts based on their fitness scale and goals.
This can enhance the overall treadmill experience and keep users engaged.
```

In [ ]:
```

```