🎯 Objective

Create comprehensive customer profiles for each AeroFit treadmill product through descriptive analytics. Develop two-way contingency tables and analyze conditional and marginal probabilities to discern customer characteristics, facilitating improved product recommendations and informed business decisions.

About Data

The company collected the data on individuals who purchased a treadmill from the AeroFit stores during three months.The data is available in a single csv file

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.

```
#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
```

```
df = pd.read_csv('aerofit_treadmill.csv?1639992749')
```

```
df.head()
```

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

```
df.tail()
```

	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

```
df.shape
```

```
(180, 9)
```

```
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
```

Insights

- From the above analysis, it is clear that, data has total of 9 features with mixed alpha numeric data. Also we can see that there is no missing data in the columns.
- The data type of all the columns are matching with the data present in them. But we will change the datatype of Usage and Fitness into str(object).

🔁 Changing the Datatype of Columns

- Changing the datatype of Usage and Fitness columns

```
df['Usage'] = df['Usage'].astype('str')
df['Fitness'] = df['Fitness'].astype('str')

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    object
 6   Fitness        180 non-null    object
 7   Income         180 non-null    int64 
 8   Miles          180 non-null    int64 
dtypes: int64(4), object(5)
memory usage: 12.8+ KB
```

📝 Statistical Summary

```
df.describe(include = 'object')
```

	Product	Gender	MaritalStatus	Usage	Fitness
count	180	180	180	180	180
unique	3	2	2	6	5
top	KP281	Male	Partnered	3	3
freq	80	104	107	69	97

🔍 Insights

1. Product - Over the past three months, the KP281 product demonstrated the highest sales performance among the three products, accounting for approximately 44% of total sales.
2. Gender - Based on the data of last 3 months, around 58% of the buyers were Male and 42% were female
3. Marital Status - Based on the data of last 3 months, around 60% of the buyers were Married and 40% were single

```
df.describe()
```

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

🔍 Insights

1. Age - The age range of customers spans from 18 to 50 year, with an average age of 29 years.
2. Education - Customer education levels vary between 12 and 21 years, with an average education duration of 16 years.
3. Usage - Customers intend to utilize the product anywhere from 2 to 7 times per week, with an average usage frequency of 3 times per week.

4. Fitness - On average, customers have rated their fitness at 3 on a 5-point scale, reflecting a moderate level of fitness.
5. Income - The annual income of customers falls within the range of USD 30,000 to USD 100,000, with an average income of approximately USD 54,000.
6. Miles - Customers' weekly running goals range from 21 to 360 miles, with an average target of 103 miles per week.

👥 Duplicate Detection

```
df.duplicated().value_counts()
```

```
False    180
dtype: int64
```

🔍 Insights

- There are no duplicate entries in the dataset

➕ Adding new columns for better analysis

- Creating New Column and Categorizing values in Age,Education,Income and Miles to different classes for better visualization

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

```
#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)

#binning the miles values into categories
bin_range4 = [0,50,100,200,float('inf')]
bin_labels4 = ['Light Activity', 'Moderate Activity', 'Active Lifestyle', 'Fitness Enthusiast ']

df['miles_group'] = pd.cut(df['Miles'],bins = bin_range4,labels = bin_labels4)
```

```
df.head()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	age_group	edu
0	KP281	18	Male	14	Single	3	4	29562	112	Young Adults	Sec Ed
1	KP281	19	Male	15	Single	2	3	31836	75	Young Adults	Sec Ed
2	KP281	19	Female	14	Partnered	4	3	30699	66	Young Adults	Sec Ed

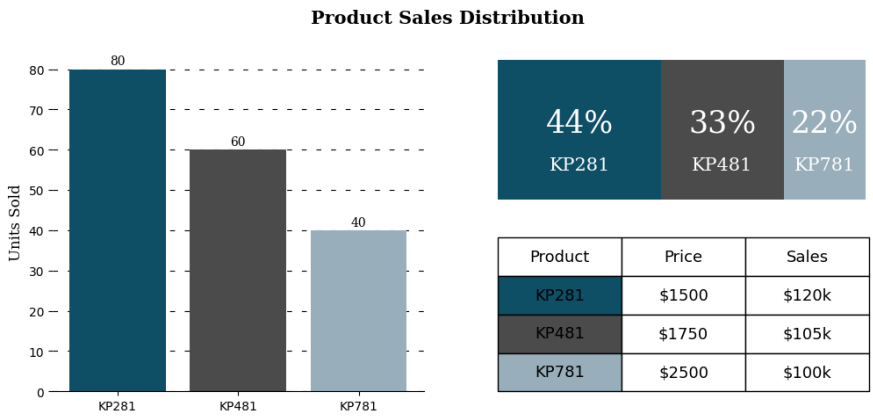
Univariate Analysis

Categorical Variables

💰 Product Sales Distribution

```
#setting the plot style

fig = plt.figure(figsize = (12,5))
gs = fig.add_gridspec(2,2)

#creating plot for 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 product_count.index:
    ax0.text(i,product_count[i]+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.loc['percent'][0],color = "#0e4f66")
ax1.barh(product_count.index[0],product_count.loc['percent'][1],left = product_count.loc['percent'][0],color = '#4b4b4c')
ax1.barh(product_count.index[0],product_count.loc['percent'][2],
         left = product_count.loc['percent'][0] + product_count.loc['percent'][1], color = '#99AEBB')
ax1.set(xlim=(0,100))


# adding info to the each bar
product_count['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]/2]
for i in range(3):
    ax1.text(product_count['info_percent'][i],0.04,f"{product_count['percent'][i]:.0f}%",
             va = 'center', ha='center',fontsize=25, fontweight='light', fontfamily='serif',color='white')

    ax1.text(product_count['info_percent'][i],-0.2,product_count.index[i],
             va = 'center', ha='center',fontsize=15, fontweight='light', fontfamily='serif',color='white')

#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'],['KP781','$2500','$100k']]
color_2d = [['#0e4f66','#FFFFFF','#FFFFFF'],['#4b4b4c','#FFFFFF','#FFFFFF'],['#99AEBB','#FFFFFF','#FFFFFF']]

table = ax2.table(cellText = product_portfolio, cellColours=color_2d, cellLoc='center',colLabels =['Product','Price','Sales'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

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()
```



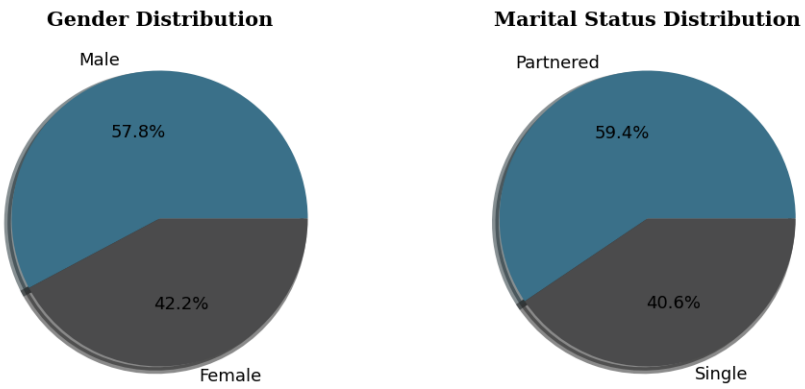
🔍 Insights

- The KP281 treadmill model, positioned as an entry-level product, has the highest number of units sold, trailed by the KP481 (mid-level) and KP781 (advanced) models.
- All three models have nearly equal contributions in terms of generating sales revenue.

👩 👨 Gender and 👫 Marital Status Disribution

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

                                    # creating pie chart for gender disribution
ax0 = fig.add_subplot(gs[0,0])

color_map = ["#3A7089", "#4b4b4c"]
ax0.pie(df['Gender'].value_counts().values,labels = df['Gender'].value_counts().index,autopct = '%.1f%%',
        shadow = True,colors = color_map,wedgeprops = {'linewidth': 5},textprops={'fontsize': 13, 'color': 'black'})

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

                                    # creating pie chart for marital status
ax1 = fig.add_subplot(gs[0,1])

color_map = ["#3A7089", "#4b4b4c"]
ax1.pie(df['MaritalStatus'].value_counts().values,labels = df['MaritalStatus'].value_counts().index,autopct = '%.1f%%',
        shadow = True,colors = color_map,wedgeprops = {'linewidth': 5},textprops={'fontsize': 13, 'color': 'black'})
```

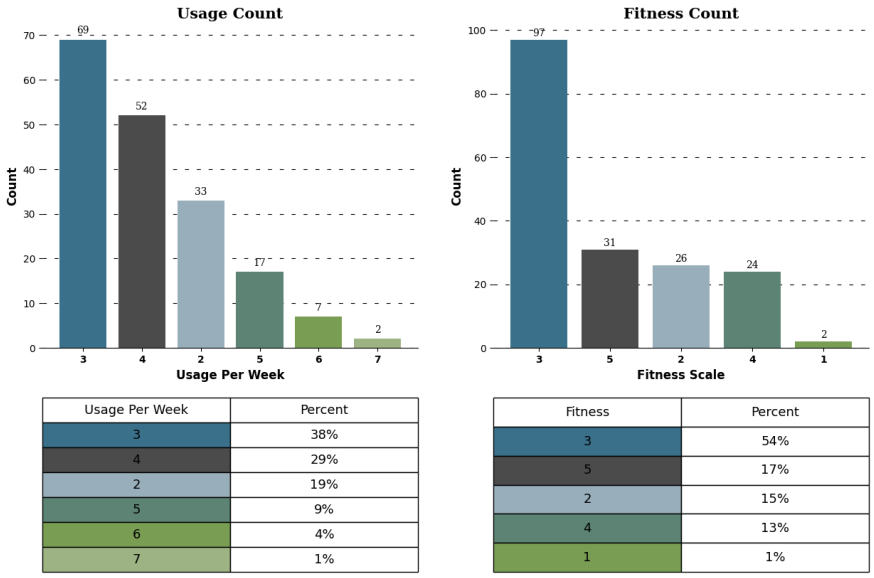
```
#setting title for visual
ax1.set_title('Marital Status Distribution',{'font':'serif', 'size':15,'weight':'bold'})

plt.show()
```



Buyer 🏋️‍♀️ Fitness and 🏃‍♂️ Treadmill Usage

```
#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 disribution
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 temp.index:
    ax0.text(i,temp[i]+2,temp[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('Count',fontweight = 'bold',fontsize = 12)
ax0.set_xlabel('Usage Per Week',fontweight = 'bold',fontsize = 12)
ax0.set_xticklabels(temp.index,fontweight = 'bold')

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

                                    #creating a info table for usage
ax1 = fig.add_subplot(gs[1,0])
usage_info = [['3','38%'],['4','29%'],['2','19%'],['5','9%'],['6','4%'],['7','1%']]
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, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax1.axis('off')

                                    # creating bar chart for fitness scale
ax2 = fig.add_subplot(gs[0,1])
temp = df['Fitness'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374','#7A9D54','#9EB384']
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 = '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_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax2.set_title('Fitness Count',{'font':'serif', 'size':15,'weight':'bold'})

                                    #creating a info table for usage

ax1 = 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 = ax1.table(cellText = fitness_info, cellColours=color_2d, cellLoc='center',colLabels =['Fitness','Percent'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax1.axis('off')


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%

🔍 Insights

- Almost 85% of the customers plan to use the treadmill for 2 to 4 times a week and only 15% using 5 times and above each week
- 54% of the customers have self-evaluated their fitness at a level 3 on a scale of 1 to 5. Furthermore, a substantial 84% of the total customers have rated themselves at 3 or higher, indicating commendable fitness levels.

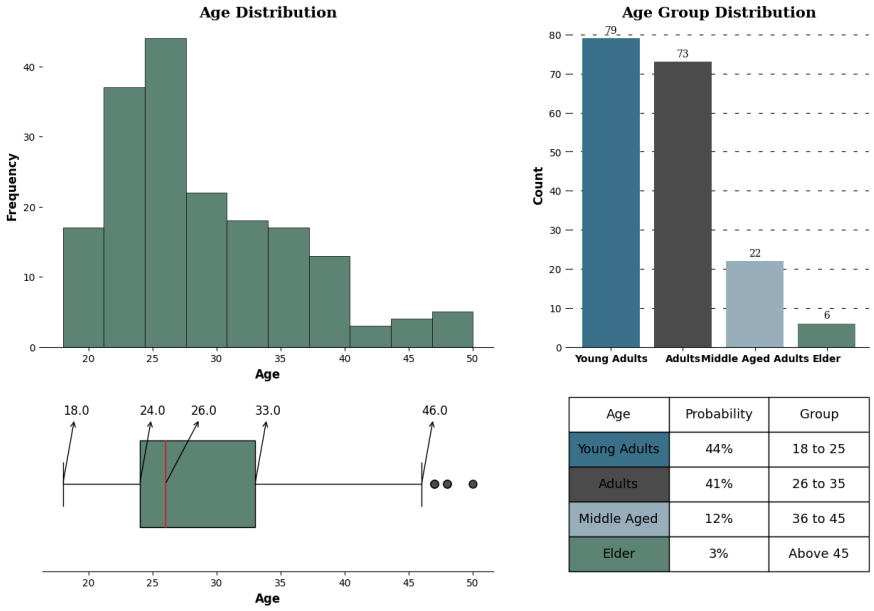
Numerical Variables

📅 Customer Age Distribution

```
#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,Q3 and lowerlimit

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 comprehension

    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),fontsize = 12,
             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',va = '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_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax2.set_title('Age Group Distribution',{'font':'serif', 'size':15,'weight':'bold'})

                                    #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'],['Middle Aged','12%','36 to 45'],
            ['Elder','3%','Above 45']]
```

```
color_2d = [['#3A7089','#FFFFFF','#FFFFFF'],['#4D4D4C','#FFFFFF','#FFFFFF'],['#99AEBB','#FFFFFF','#FFFFFF'],
            ['#5C8374','#FFFFFF','#FFFFFF']]

table = ax3.table(cellText = age_info, cellColours=color_2d, cellLoc='center',colLabels =['Age','Probability','Group'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax3.axis('off')


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

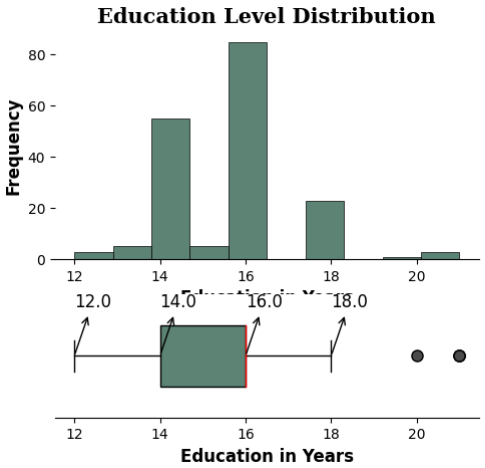
🔍 Insights

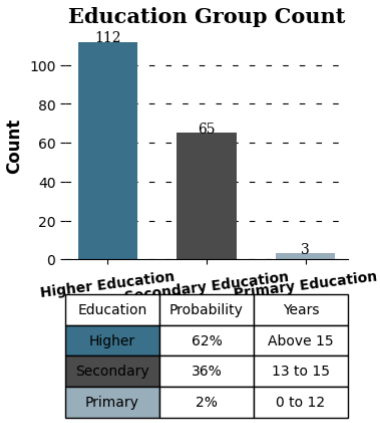
- 85% of the customers fall in the age range of 18 to 35. with a median age of 26, suggesting young people showing more interest in the companies products
- **Outliers**
- As we can see from the box plot, there are 3 outlier's present in the age data.

👨‍🎓👩‍🎓 Customer Education Distribution

```
#setting the plot style

fig = plt.figure(figsize = (10,5))
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)
```

```
    ax0.spines[s].set_visible(False)

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


                                    #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,Q3 and lowerlimit

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 comprehension

    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',va = '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_xticklabels(temp.index,fontweight = 'bold',rotation = 7)

#setting title for visual
ax2.set_title('Education Group Count',{'font':'serif', 'size':15,'weight':'bold'})


                                       #creating a table for group info

ax3 = fig.add_subplot(gs[1,1])
edu_info = [['Higher','62%','Above 15'],['Secondary','36%','13 to 15'],['Primary','2%','0 to 12']]
color_2d = [["#3A7089",'#FFFFFF','#FFFFFF'],["#4b4b4c",'#FFFFFF','#FFFFFF'],['#99AEBB','#FFFFFF','#FFFFFF']]

table = ax3.table(cellText = edu_info, cellColours=color_2d, cellLoc='center',colLabels =['Education','Probability','Years'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax3.axis('off')


plt.show()
```





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

🔍 Insights

- 98% of the customers have education more than 13 years highlighting a strong inclination among well-educated individuals to purchase the products. It's plausible that health awareness driven by education could play a pivotal role in this trend.

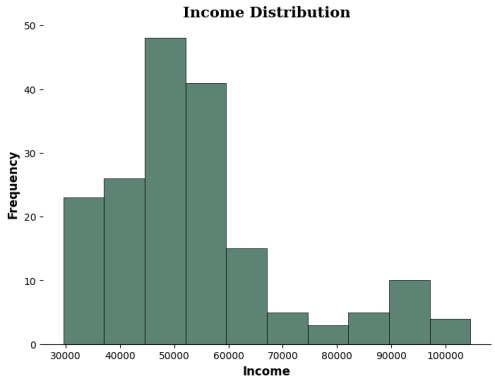
* Outliers

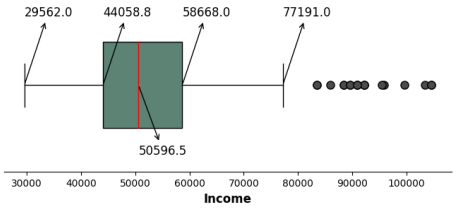
As we can see from the box plot, there are 2 outlier's present in the education data.

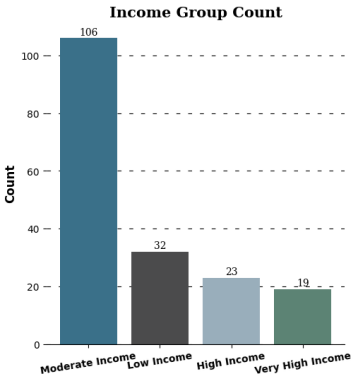
💲 Customer Income Distribution

```
#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 = 0.5)

# 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,Q3 and lowerlimit

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 comprehension
```

```
    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 = 12,
             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 = '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_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%','60k to 80k'],
            ['Vey High','10%','Above 80k']]
color_2d = [["#4b4b4c",'#FFFFFF','#FFFFFF'],["#3A7089",'#FFFFFF','#FFFFFF'],['#99AEBB','#FFFFFF','#FFFFFF'],
            ['#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

🔍 Insights

- Almost 60% of the customers fall in the income group of (40k to 60k) dollars suggesting higher inclination of this income group people towards the products.
- Surprisingly 18% of the customers fall in the income group of (<40) suggesting almost 77% of the total customers fall in income group of below 60k and only 23% of them falling in 60k and above income group

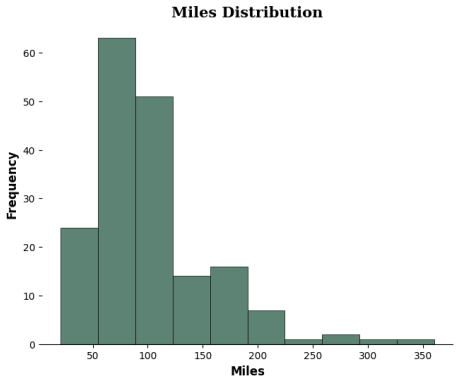
* Outliers

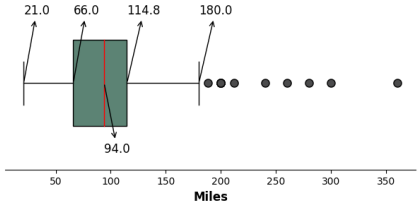
As we can see from the box plot, there are many outlier's present in the income data.

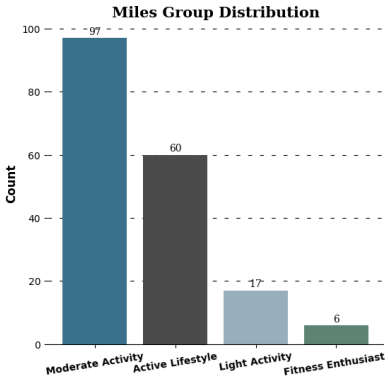
🏃‍♀️🏃‍♀️ Customers Expected Weekly Mileage

```
#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 upperlimit,Q1,Q3 and lowerlimit

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 comprehension

    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 = 12,
             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',va = '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_xticklabels(temp.index,fontweight = 'bold',rotation = 9)

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


                                      #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 to 100'],['Active Lifestyle','34%','101 to 200'],
              ['Fitness Enthusiast','3%','Above 200']]
color_2d = [['#99AEBB','#FFFFFF','#FFFFFF'],["#3A7089",'#FFFFFF','#FFFFFF'],["#4b4b4c",'#FFFFFF','#FFFFFF'],
              ['#5C8374','#FFFFFF','#FFFFFF']]

table = ax3.table(cellText = miles_info, cellColours=color_2d, cellLoc='center',colLabels =['Activity','Probability','Miles'],
                  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

🔍 Insights

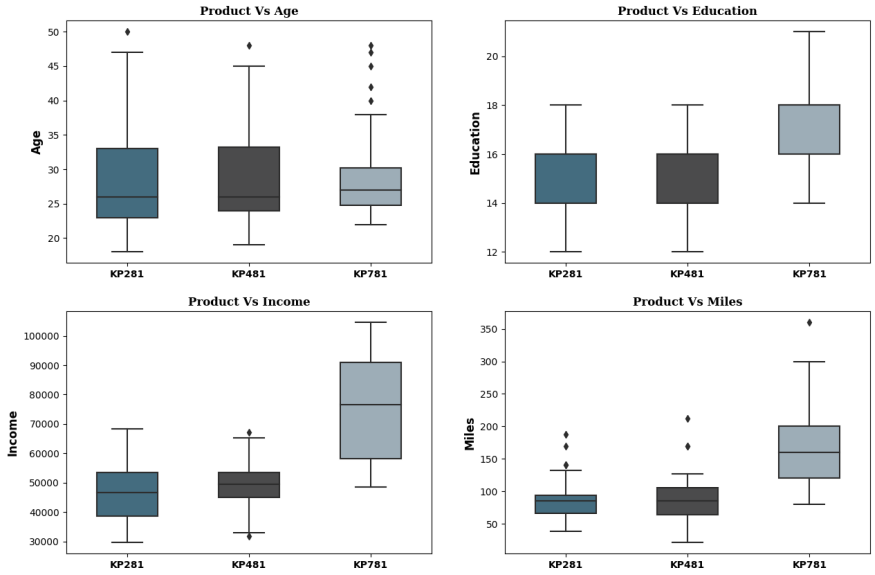
- Almost 88% of the customers plans to use the treadmill for 50 to 200 miles per week with a median of 94 miles per week.

* Outliers

As we can see from the box plot, there are 8 outlier's present in the miles data.

👨‍🏫 Analysis of Product Type

```
#setting the plot style
fig = plt.figure(figsize = (15,10))
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 =["#3A7089", "#4b4b4c",'#99AEBB'])

    #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()
```



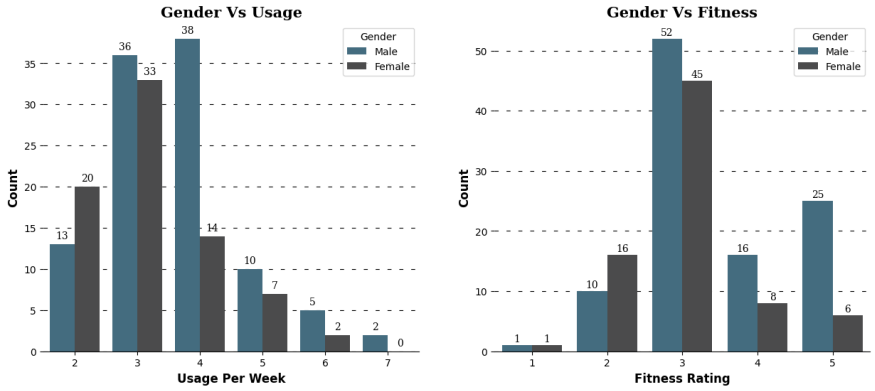
🔍 Insights

- The analysis presented above clearly indicates a strong preference for the treadmill model KP781 among customers who possess higher education, higher income levels, and intend to engage in running activities exceeding 150 miles per week.

👩 👨 Gender vs Product Usage And Gender Vs Fitness

```
#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['Usage'].unique()),
                     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','size' : 10},ha = 'center',va = 'center')

#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['Fitness'].unique()),
                     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','size' : 10},ha = 'center',va = '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)

#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()
```



🔍 Insights

1. Gender Vs Usage

- Almost 70% of Female customers plan to use the treadmill for 2 to 3 times a week whereas almost 70% of Male customer plan to use the treadmill for 3 to 4 times a week

2. Gender Vs Fitness

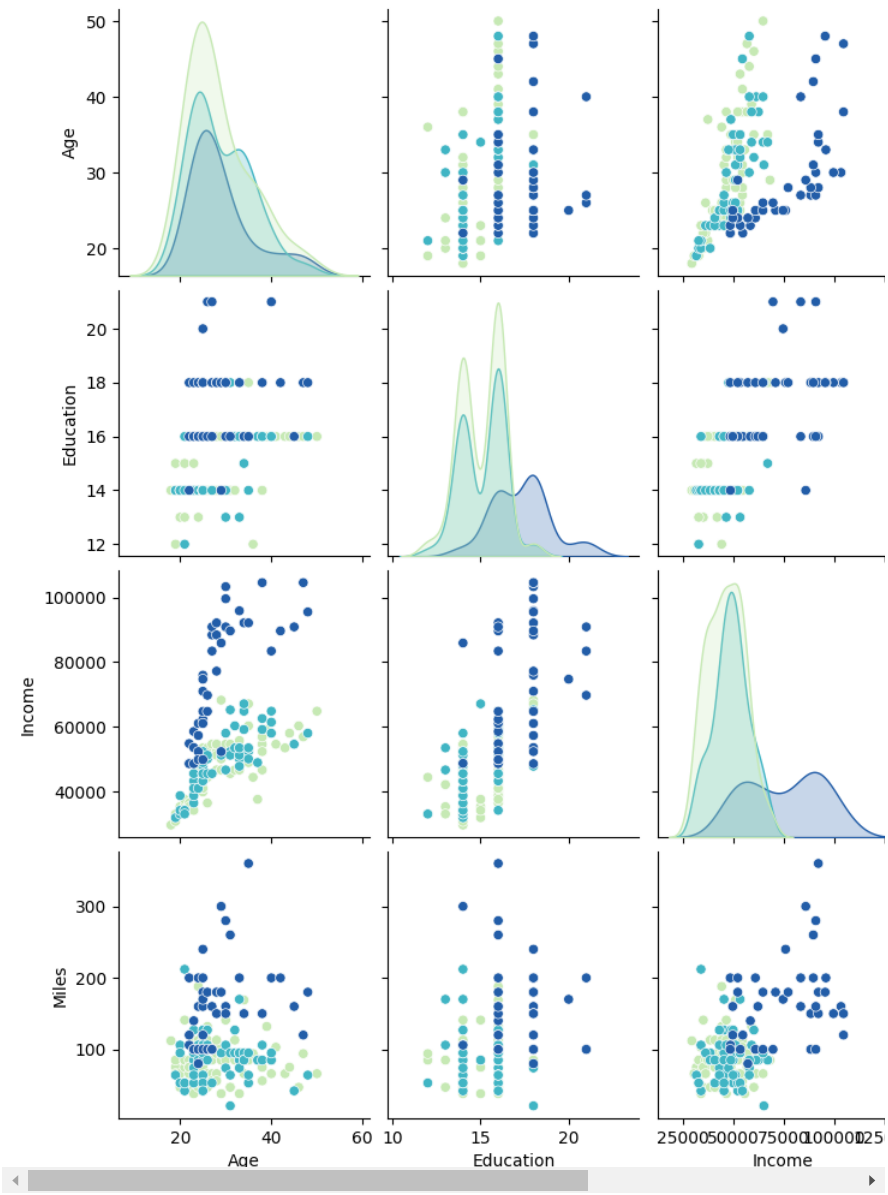
- Almost 80% of Female customers rated themselves between 2 to 3 whereas almost 90% of Male customer rated themselves between 3 to 5 on the fitness scale

Correlation between Variables

Pairplot

```
df_copy = copy.deepcopy(df)
```

```
sns.pairplot(df_copy, hue ='Product', palette= 'YlGnBu')
plt.show()
```

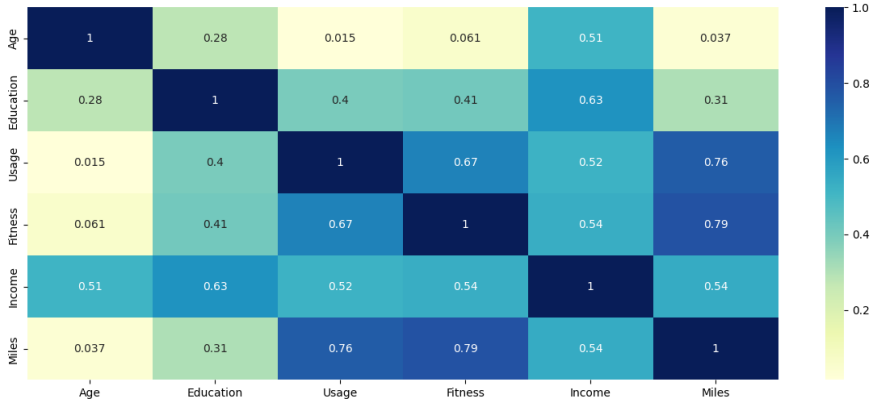


Heatmap

```
# 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    int64
 6   Fitness        180 non-null    int64
 7   Income         180 non-null    int64
 8   Miles          180 non-null    int64
 9   age_group      180 non-null    category
 10  edu_group      180 non-null    category
 11  income_group   180 non-null    category
 12  miles_group    180 non-null    category
dtypes: category(4), int64(6), object(3)
memory usage: 14.2+ KB
```

```
corr_mat = df_copy.corr()

plt.figure(figsize=(15,6))

sns.heatmap(corr_mat,annot = True, cmap="YlGnBu")

plt.show()
```



🔍 Insights

- From the pair plot we can see Age and Income are positively correlated and heatmap also suggests a strong correlation between them
- Eductaion and Income are highly correlated as its obvious. Eductation also has significatnt correlation between Fitness rating and Usage of the treadmill.
- Usage is highly correlated with Fitness and Miles as more the usage more the fitness and mileage.

Computing Probability - Marginal, Conditional Probability

Probability of product purchase w.r.t. gender

```
pd.crosstab(index =df['Product'],columns = df['Gender'],margins = True,normalize = True ).round(2)
```

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

🔍 Insights

1. The Probability of a treadmill being purchased by a female is 42%.

   The conditional probability of purchasing the treadmill model given that the customer is female is

- For Treadmill model KP281 - 22%
- For Treadmill model KP481 - 16%
- For Treadmill model KP781 - 4%

2. The Probability of a treadmill being purchased by a male is 58%. The conditional probability of purchasing the treadmill model given that the customer is male is

- For Treadmill model KP281 - 22%
- For Treadmill model KP481 - 17%
- For Treadmill model KP781 - 18%

Probability of product purchase w.r.t. Age

```
pd.crosstab(index =df['Product'],columns = df['age_group'],margins = True,normalize = True ).round(2)
```

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

Insights

1. The Probability of a treadmill being purchased by a Young Adult(18-25) is

- List item
- List item

44%. The conditional probability of purchasing the treadmill model given that the customer is Young Adult is

- For Treadmill model KP281 - 19%
- For Treadmill model KP481 - 16%
- For Treadmill model KP781 - 9%

2. The Probability of a treadmill being purchased by a Adult(26-35) is 41%.

The conditional probability of purchasing the treadmill model given that the customer is Adult is

- 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 Aged(36-45) is 12%.
4. The Probability of a treadmill being purchased by a Elder(Above 45) is only 3%.

Probability of product purchase w.r.t. Education level

```
pd.crosstab(index =df['Product'],columns = df['edu_group'],margins = True,normalize = True ).round(2)
```

edu_group	Primary Education	Secondary Education	Higher Education	All
Product				
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

Insights

1. The Probability of a treadmill being purchased by a customer with Higher Education(Above 15 Years) is 62%. The conditional probability of purchasing the treadmill model given that the customer has Higher Education is

- 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 Education(13-15 yrs) is 36%.

The conditional probability of purchasing the treadmill model given that the customer has Secondary Education is

- 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 Education(0 to 12 yrs) is only 2%.

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:

    1. Age of customer mainly between 18 to 35 years with few between 35 to 50 years
    2. Education level of customer 13 years and above
    3. Annual Income of customer below USD 60,000
    4. Weekly Usage - 2 to 4 times
    5. Fitness Scale - 2 to 4
    6. Weekly Running Mileage - 50 to 100 miles