Business Case : Aerofit

1. Defining Problem Statement and Analysing basic metrics

- ➔ Aerofit is a leading brand in the field of fitness equipment. Aerofit provides a product range including machines such as treadmills, exercise bikes, gym equipment, and fitness accessories to cater to the needs of all categories of people.
- ➔ In this case study we aim to identify and profile the target audience for each type of treadmill offered by AeroFit. This will help AeroFit provide better product recommendations to new customers and tailor marketing strategies to specific customer segments. To achieve this, we will perform descriptive analytics to develop customer profiles for each treadmill product and investigate differences in customer characteristics across the products.

Let's import essential libraries and our data

```
# import dependancies
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# read data
df = pd.read_csv('/content/drive/MyDrive/Data sets/aerofit_treadmill.csv')
```

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

```
df
```

	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

Columns list

```
df.columns
```

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

Shape of data

```
df.shape
```

```
(180, 12)
```

Data types of all columns

```
df.dtypes
```

	0
Product	object
Age	int64
Gender	object
Education	int64
MaritalStatus	object
Usage	int64
Fitness	int64
Income	int64
Miles	int64
AgeGroup	category
MileGroup	category
IncomeGroup	category

dtype: object

Statistical summary of the data

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

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	AgeGroup	MileGroup	IncomeGroup
count	180	180.000000	180	180.000000	180	180.000000	180.000000	180.000000	180.000000	180	180	179
unique	3	NaN	2	NaN	2	NaN	NaN	NaN	NaN	4	4	8
top	KP281	NaN	Male	NaN	Partnered	NaN	NaN	NaN	NaN	18-25	0-100	50000-60000
freq	80	NaN	104	NaN	107	NaN	NaN	NaN	NaN	79	114	55
mean	NaN	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53719.577778	103.194444	NaN	NaN	NaN
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16506.684226	51.863605	NaN	NaN	NaN
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29562.000000	21.000000	NaN	NaN	NaN
25%	NaN	24.000000	NaN	14.000000	NaN	3.000000	3.000000	44058.750000	66.000000	NaN	NaN	NaN
50%	NaN	26.000000	NaN	16.000000	NaN	3.000000	3.000000	50596.500000	94.000000	NaN	NaN	NaN
75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58668.000000	114.750000	NaN	NaN	NaN
max	NaN	50.000000	NaN	21.000000	NaN	7.000000	5.000000	104581.000000	360.000000	NaN	NaN	NaN

Count of rows and columns and its info

```
# lets understand number of rows and columns
print(f"Number of rows: {df.shape[0]}")
print(f"Number of columns: {df.shape[1]}")
```

```
Number of rows: 180
Number of columns: 9
```

```
# lets do some basic analysis about df
df.info() # 9 columns with 180 rows of records / 6 - integers coulumns / 3 object type columns
```

```
<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
```

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

```
# number of unique values present in each column
df.nunique()
```

	0
Product	3
Age	32
Gender	2
Education	8
MaritalStatus	2
Usage	6
Fitness	5
Income	62
Miles	37

dtype: int64

```
# unique values in each columns
df.apply(lambda col: col.unique())
```

	0
Product	[KP281, KP481, KP781]
Age	[18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 2...
Gender	[Male, Female]
Education	[14, 15, 12, 13, 16, 18, 20, 21]
MaritalStatus	[Single, Partnered]
Usage	[3, 2, 4, 5, 6, 7]
Fitness	[4, 3, 2, 1, 5]
Income	[29562, 31836, 30699, 32973, 35247, 37521, 363...
Miles	[112, 75, 66, 85, 47, 141, 103, 94, 113, 38, 1...

dtype: object

```
# product value counts
df['Product'].value_counts()
```

Product	count
KP281	80
KP481	60
KP781	40

dtype: int64

```
df['Gender'].value_counts()
```

Gender	count
Male	104
Female	76

dtype: int64

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

MaritalStatus	count
Partnered	107
Single	73

dtype: int64

```
df['Age'].value_counts() # most purchased age is 25, and most purchased range is 25-35
```

Age	count
25	25
23	18
24	12
26	12

Let’s check is there any null values in our data

```
df.isnull().any()
```

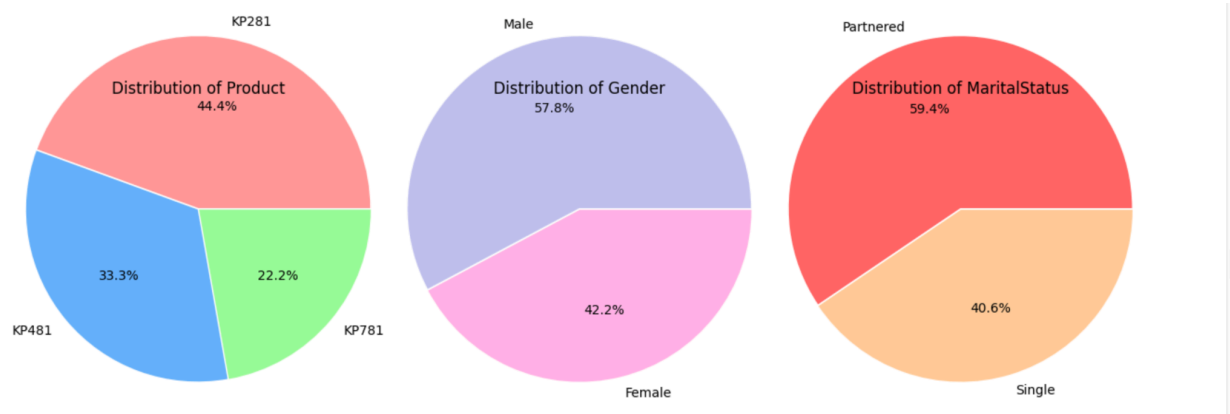
	0
Product	False
Age	False
Gender	False
Education	False
MaritalStatus	False
Usage	False
Fitness	False
Income	False
Miles	False
AgeGroup	False
MileGroup	False
IncomeGroup	True

dtype: bool

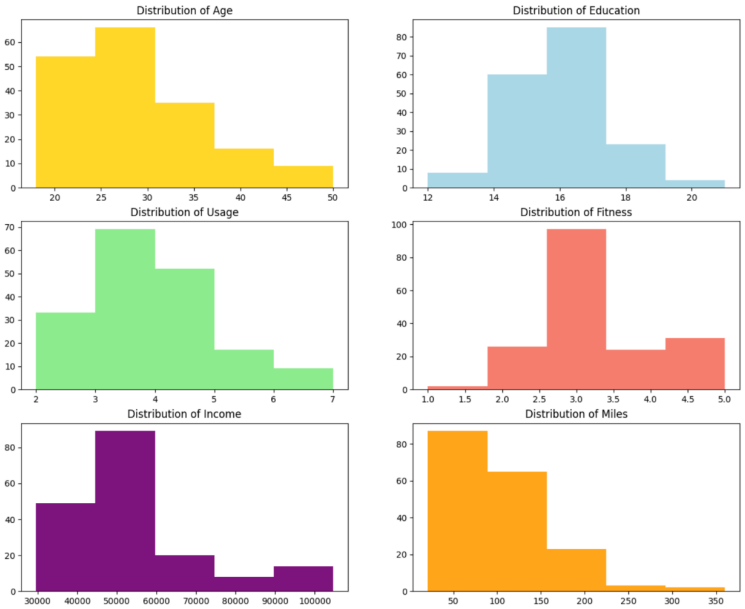
3. Visual Analysis - Univariate & Bivariate

```
# uni variate visual analysis --> Let's understand the distribution of Product, Gender and Marital status
```

```
plt.figure(figsize=(15,3))
plot_cnt = 1

colors_list = [['#ff9999','#66b3ff','#99ff99'],  # for 'Product'
               ['#c2c2f0', '#ffb3e6'],  # for 'Gender'
               ['#ff6666', '#ffcc99']]  # for 'MaritalStatus'

# distribution of pie charts for below columns

for i, col in enumerate(['Product', 'Gender', 'MaritalStatus']):
    plt.subplot(1, 3, plot_cnt)
    explode_tuple = tuple([0.01] * df[col].nunique())
    plt.pie(df[col].value_counts(),
            autopct='%1.1f%%',
            labels=df[col].value_counts().index,
            radius=2,
            explode=explode_tuple,
            colors=colors_list[i])  # pass custom colors
    plt.title(f'Distribution of {col}')
    plot_cnt += 1
plt.show()
```

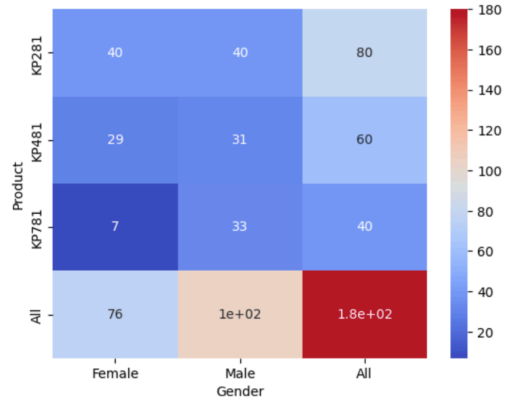


```
plt.figure(figsize=(15,12))
fig.subplots_adjust(top=2)

# histogram plots for below columns

plot_1 = 1 # initializer for loop

# Define a list of colors for the histograms
colors = ['gold', 'lightblue', 'lightgreen', 'salmon', 'purple', 'orange']

for i, col in enumerate(['Age','Education','Usage','Fitness','Income','Miles']):
    plt.subplot(3, 2, plot_1)

    # Use different colors for each histogram
    plt.hist(df[col], bins=5, color=colors[i])

    plt.title(f'Distribution of {col}')
    plot_1 += 1

plt.show()
```




```
PG = pd.crosstab(df['Product'], df['Gender'], margins=True)
sns.heatmap(PG, cmap='coolwarm', annot=True)
print(PG)
```

```
Gender   Female  Male  All
Product
KP281        40    40   80
KP481        29    31   60
KP781         7    33   40
All          76   104  180
```



Let’s create some columns and groups for better analysis (categorization of users):

```
# Let's categorize the columns and create groups

age_bins=[17, 25, 35, 45, 150]
age_groups = ['18-25', '26-35', '36-45', '>45']

# Categorize the ages into age groups
df['AgeGroup'] = pd.cut(df['Age'], bins=age_bins, labels=age_groups)

#Miles
miles_bins=[0, 100, 200, 300, 1000]
miles_groups = ['0-100', '101-200', '201-300', '>300']

# Categorize the miles into age groups
df['MileGroup'] = pd.cut(df['Miles'], bins=miles_bins, labels=miles_groups)

#Income
income_bins=[30000, 40000, 50000, 60000, 70000, 80000, 90000, 100000, 110000]
income_groups = ['30000-40000', '40000-50000', '50000-60000', '60000-70000', '70000-80000', '80000-90000', '90000-100000', '>100000']

# Categorize the income into age groups
df['IncomeGroup'] = pd.cut(df['Income'], bins=income_bins, labels=income_groups)

df
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	AgeGroup	MileGroup	IncomeGroup
0	KP281	18	Male	14	Single	3	4	29562	112	18-25	101-200	NaN
1	KP281	19	Male	15	Single	2	3	31836	75	18-25	0-100	30000-40000
2	KP281	19	Female	14	Partnered	4	3	30699	66	18-25	0-100	30000-40000
3	KP281	19	Male	12	Single	3	3	32973	85	18-25	0-100	30000-40000
4	KP281	20	Male	13	Partnered	4	2	35247	47	18-25	0-100	30000-40000
...	...	...	...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200	36-45	101-200	80000-90000
176	KP781	42	Male	18	Single	5	4	89641	200	36-45	101-200	80000-90000
177	KP781	45	Male	16	Single	5	5	90886	160	36-45	101-200	90000-100000
178	KP781	47	Male	18	Partnered	4	5	104581	120	>45	101-200	>100000
179	KP781	48	Male	18	Partnered	4	5	95508	180	>45	101-200	90000-100000

180 rows × 12 columns

Probability:

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

Gender	Female	Male	All
Product			
KP281	0.222222	0.222222	0.444444
KP481	0.161111	0.172222	0.333333
KP781	0.038889	0.183333	0.222222
All	0.422222	0.577778	1.000000

- ➔ Probability of Female to purchase, P(F) = 0.42
- ➔ Probability of Male to purchase, P(M) = 0.57
- ➔ Probability to purchase KP281, P(KP1) = 0.44
- ➔ Probability for a Female to purchase KP781, P(F n KP781) = 0.038
- ➔ Probability for a Male to purchase KP781, P(M n KP781) = 0.18
- ➔ In our above insights we have observed that males purchase KP781 more than females. The probability stats also indicate the same.

```
pd.crosstab(df['Product'], df['Usage'], margins = True, normalize = True)
```

Usage	2	3	4	5	6	7	All
Product							
KP281	0.105556	0.205556	0.122222	0.011111	0.000000	0.000000	0.444444
KP481	0.077778	0.172222	0.066667	0.016667	0.000000	0.000000	0.333333
KP781	0.000000	0.005556	0.100000	0.066667	0.038889	0.011111	0.222222
All	0.183333	0.383333	0.288889	0.094444	0.038889	0.011111	1.000000

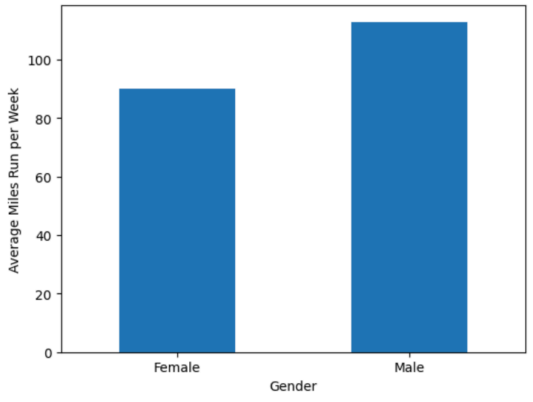
- ➔ With usage more, customers are definitely buying advanced models.
- ➔ Probability of purchasing KP3 for customer who runs for 5 days a week, P(KP781 n U5) = 0.067

```
# Let's find conditional probability
cp = pd.crosstab(df['Product'], [df['AgeGroup']], margins = True)
cp
```

AgeGroup	18–25	26–35	36–45	>45	All
Product					
KP281	34	32	11	3	80
KP481	28	24	7	1	60
KP781	17	17	4	2	40
All	79	73	22	6	180

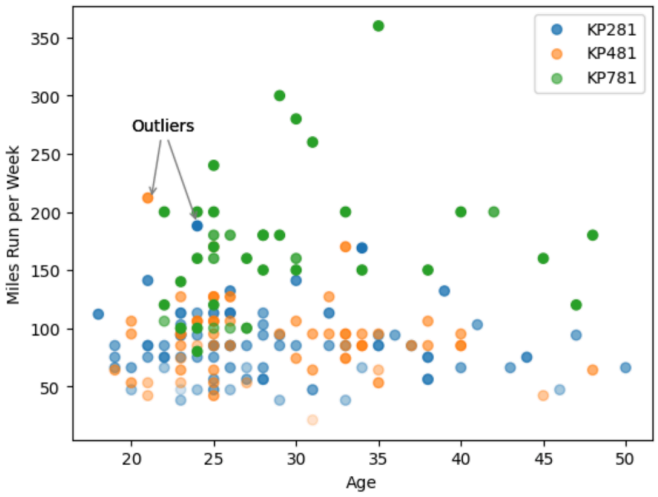
Gender-Miles relation

```
# Bar chart to visualize the gender analysis
gender_data = df.groupby('Gender')['Miles'].mean()
gender_data.plot(kind='bar', xlabel='Gender', ylabel='Average Miles Run per Week')
plt.xticks(rotation=0)
plt.show()
```



```
# Product purchase by Age vs Miles vs Fitness Level
```

```
df["fitness_alpha"] = df['Fitness']/5
products = df['Product'].unique()

for product in products:
    df_product = df[df['Product'] == product]
    plt.scatter(df_product['Age'], df_product['Miles'], label=product, alpha=df_product['fitness_alpha'])

# Add axis labels and legend
plt.xlabel('Age')
plt.ylabel('Miles Run per Week')
plt.legend()

plt.annotate("Outliers",xy=(24, 190), xytext=(20,270), arrowprops=dict(color="grey", arrowstyle="->"))
plt.annotate("Outliers", xy=(21.2, 211), xytext=(20,270),arrowprops=dict(color="grey", arrowstyle="->"))

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

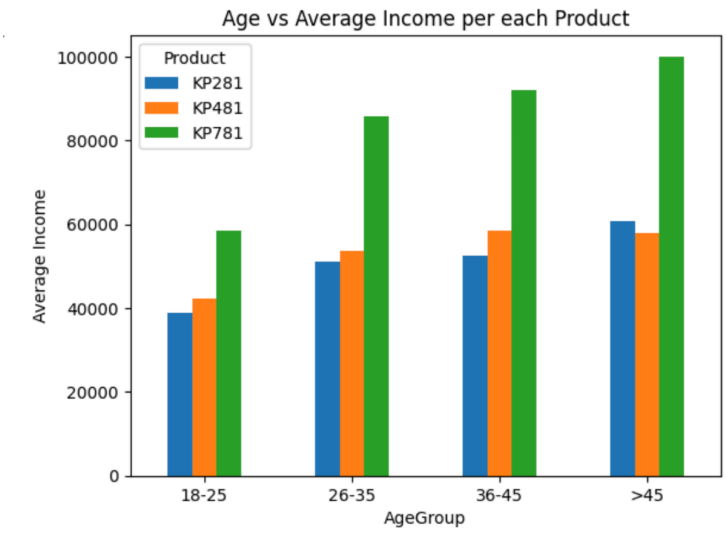



```
# Age vs average income per each product
```

```
# calculate the average income for each age group per each product
age_data = df.groupby(['AgeGroup', 'Product'])['Income'].mean().unstack()
print(age_data)
```

```
Product          KP281         KP481          KP781
AgeGroup
18-25     38724.882353  42312.642857   58375.529412
26-35     51129.468750  53598.250000   85686.235294
36-45     52612.090909  58474.285714   92131.000000
>45       60640.000000  57987.000000  100044.500000
```

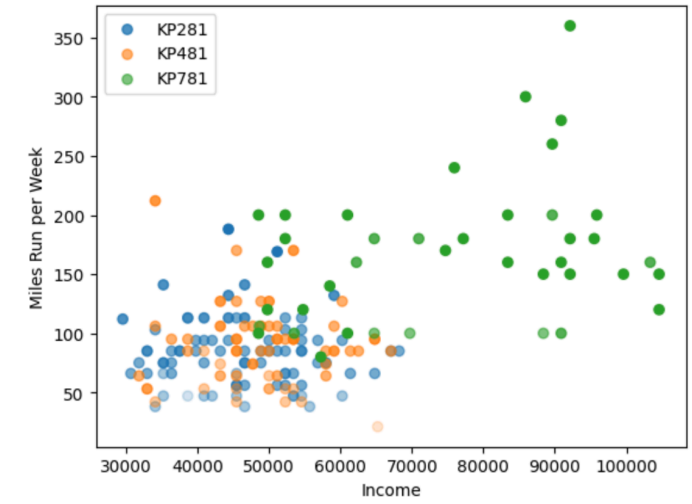
```
# create a bar chart to visualize the age analysis

age_data.plot(kind='bar', ylabel='Average Income')

plt.xticks(rotation = 0)
plt.title('Age vs Average Income per each Product')
plt.legend(title='Product')
plt.show()
```




```
# Income vs Miles per week vs Fitness level
```

```
products = df['Product'].unique()

for product in products:
  df_product = df[df['Product'] == product]
  plt.scatter( df_product['Income'], df_product['Miles'], label=product, alpha= df_product['fitness_alpha'])

# Add axis labels and legend
plt.xlabel('Income')
plt.ylabel('Miles Run per Week')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7ee2118857e0>
```



4. Missing Value & Outlier Detection

Let’s find out missing values in the data

```
df.isnull().any()
```

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

dtype: bool

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

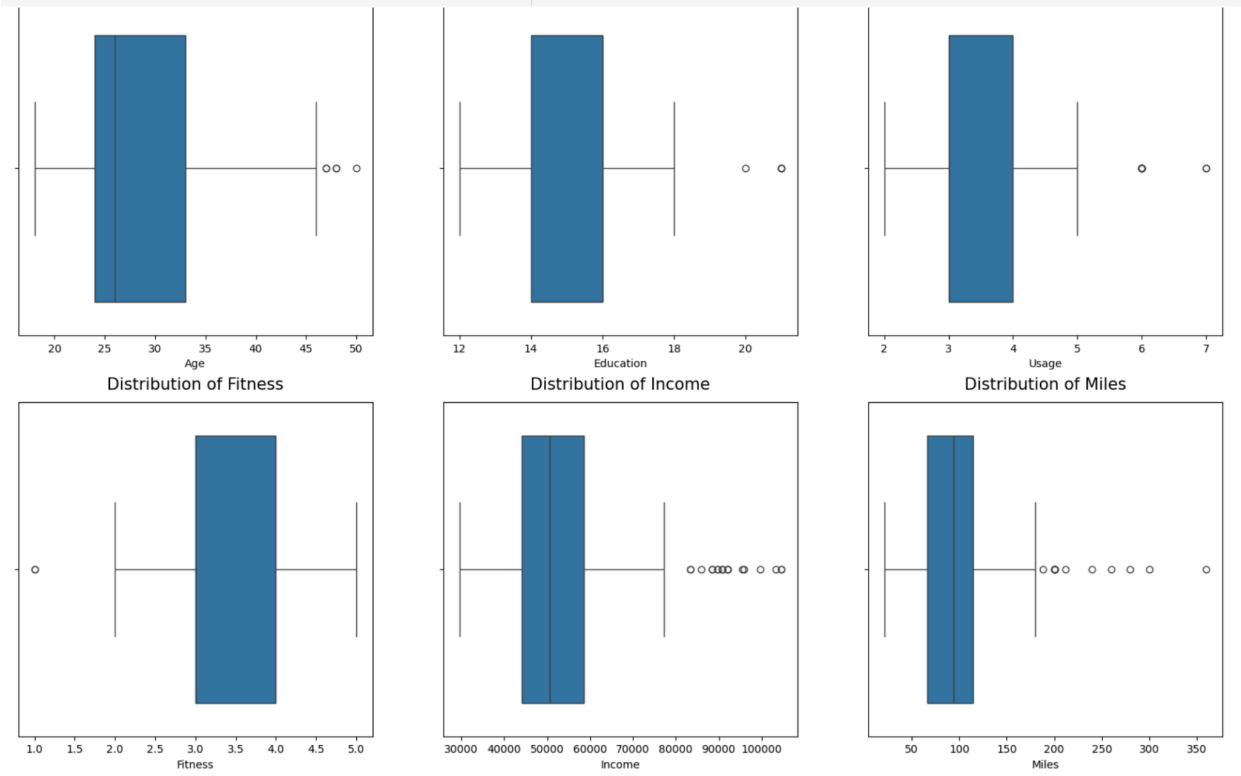
	0
Product	0
Age	0
Gender	0
Education	0
MaritalStatus	0
Usage	0
Fitness	0
Income	0
Miles	0

dtype: int64

➔ This dataset has no null records/values. So we do not need missing value treatment.

Let's detect outliers in the data

```
# Distribution of data among columns
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(20, 5))
fig.subplots_adjust(top=2)

cols = ['Age','Education','Usage','Fitness','Income','Miles']
count = 0

for row in range(2):
    for col in range(3):
        sns.boxplot(data=df, x=cols[count], ax=axs[row][col])
        axs[row,col].set_title(f"Distribution of {cols[count]}", pad=12, fontsize=15)
        count += 1
plt.show()
```

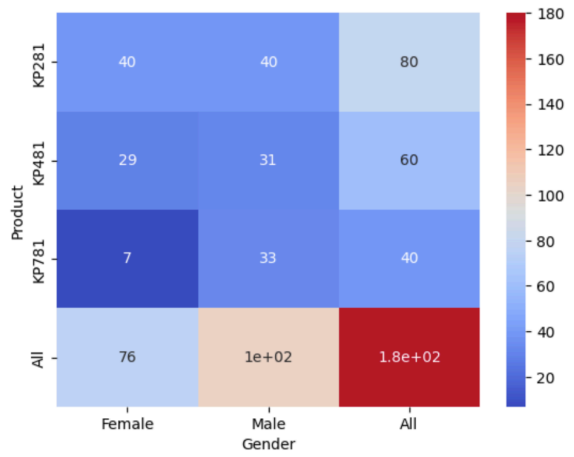


- ➔ **Age:** Age values range from 18 to 50. But all the descriptive statistics lie around 25.
- ➔ **Education:** Considering the 50% and 60% percentiles are 16, most customers have 16 years of education. Mode also gives the same info.
- ➔ **Usage:** Many claim to use the machine between 3 and 4.
- ➔ **Fitness:** Many claim to have fitness between 3 and 4.
- ➔ **Income:** Income values range from 30K to 10K. Average is 53719.
- ➔ **Miles:** Miles values range from 21 to 360. Average is 103, median is 94 and mode is 85.

Let's identify correlation with heat map:

```
PG = pd.crosstab(df['Product'], df['Gender'], margins=True)
sns.heatmap(PG, cmap='coolwarm', annot=True)
print(PG)
```

```
Gender   Female  Male  All
Product
KP281        40    40   80
KP481        29    31   60
KP781         7    33   40
All          76   104  180
```



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

1. Among the 180 records of the 3 products, 80 records are for KP281, 60 are for KP481 and 40 are for KP781.
2. KP281 and KP481 are preferred by male and females equally. But KP781 is preferred by more males than females.
3. Purchases are done by more partners than single customers.

Insights from pie chart :

1. 44.4% have bought KP281, 33.3% have bought KP481 and 22.2% have bought KP781.
2. 57.8% are male and 42.2% are female.
3. 60% are partnered and 40% are single

Insights from the histogram plot :

1. Age: 25 to 30 have more records.
2. Education: >19 years of education is the minimum.
3. Fitness: Moderate fitness people more prominent to purchase.
4. Miles: A consistent decrease in the number of miles run per week is observed.

Insights from bar plot :

1. Average miles run by male is greater than average miles run by females.

Insights from scatter plot : (Product purchase by Age vs Miles vs Fitness level)

1. If the expected miles per week is less than 130, it's more likely for the customer to buy a KP281 or KP481 model.
2. If the number of miles per week is higher, the customer is purchasing the KP781 model.
3. Few records with miles per week is higher but purchased basic models can be considered outliers.

Insights from scatter plot : (Income vs Miles per week vs Fitness Level)

1. Most entry-level model purchasers have a low fitness level.
2. It's interesting to see that customers with high income run more miles and their fitness levels are high.

Insights from Bar chart : (Average Income vs Age Group)

1. In every age group, the average income of advanced model purchasers is high.

6. Recommendations

1. Most of the customers running for 2 to 4 days a week, consider themselves moderate fitness and purchase entry-level(KP281) and mid-level(KP481) models.
2. And most of the customers running for more than 4 days a week are purchasing advanced models.
3. Most entry-level model purchasers have a low fitness level.
4. It's interesting to see that customers with high income run more miles and their fitness levels are high.
5. When the customer has high incomes, recommend advanced model
6. When a customer is more into fitness, recommend an advanced model.
7. When the usage per week is less than 4 days, customers are more likely to prefer entry-level(KP281) and mid-level(KP481) models.
8. More Advanced models were pushed for males, as they tend to have higher fitness goals and use treadmill more weekly.
9. 18 to 30 year old customers tend to purchase entry-level(KP281) and mid-level model(KP481) treadmills.
10. Customers who use treadmills more than 5 times a week are definitely buying advanced model(KP781)