Business Case: Aerofit - Descriptive Statistics & Probability

---

About Aerofit

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.

Business Problem

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to the new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics.

1. Perform descriptive analytics to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts.
2. For each AeroFit treadmill product, construct two-way contingency tables and compute all conditional and marginal probabilities along with their insights/impact on the business.

1. Defining Problem Statement and Analysing basic metrics

Import Libraries

Importing the libraries we need

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

Loading The Dataset

```
df = pd.read_csv("aerofit_treadmill.csv")
df
```

```
    Product  Age  Gender  Education MaritalStatus  Usage  Fitness
Income  \
0     KP281   18    Male         14        Single      3        4
29562
1     KP281   19    Male         15        Single      2        3
31836
2     KP281   19  Female         14     Partnered      4        3
30699
3     KP281   19    Male         12        Single      3        3
32973
4     KP281   20    Male         13     Partnered      4        2
35247
..      ...  ...     ...        ...           ...    ...      ...
...
175   KP781   40    Male         21        Single      6        5
83416
176   KP781   42    Male         18        Single      5        4
89641
177   KP781   45    Male         16        Single      5        5
90886
178   KP781   47    Male         18     Partnered      4        5
104581
179   KP781   48    Male         18     Partnered      4        5
95508

     Miles
0      112
1       75
2       66
3       85
4       47
..     ...
175    200
176    200
177    160
178    120
179    180

[180 rows x 9 columns]
```

Let's check the first 5 data

```
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
```

Let's check the last 5 data

```
df.tail()
```

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

     Miles
175    200
176    200
177    160
178    120
179    180
```

```
#computing no if rows and columns
print(f"Number of rows: {df.shape[0]}\nNumber of columns:
{df.shape[1]}")
```

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

Statistical Summary

```
# statisctical summary of object type columns

df.describe(include ='object')

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

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

```
# statisctical summary of numerical data type columns

df.describe()

              Age   Education       Usage     Fitness
Income  \
count  180.000000  180.000000  180.000000  180.000000     180.000000

mean    28.788889   15.572222    3.455556    3.311111   53719.577778

std      6.943498    1.617055    1.084797    0.958869   16506.684226
```

min	18.000000	12.000000	2.000000	1.000000	29562.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000
max	50.000000	21.000000	7.000000	5.000000	104581.000000

	Miles
count	180.000000
mean	103.194444
std	51.863605
min	21.000000
25%	66.000000
50%	94.000000
75%	114.750000
max	360.000000

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.
7. Minimum & Maximum age of the person is 18 & 50, mean is 28.79 and 75% of persons have age less than or equal to 33.
8. Most of the people are having 16 years of education i.e. 75% of persons are having education <= 16 years.
9. Out of 180 data points, 104's gender is Male and rest are the female.

10. Standard deviation for Income & Miles is very high. These variables might have the outliers in it.

11. There are 180 rows and 9 columns.

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

```python
#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  \
0   KP281   18    Male         14        Single      3        4
29562
1   KP281   19    Male         15        Single      2        3
31836
2   KP281   19  Female         14     Partnered      4        3
30699
3   KP281   19    Male         12        Single      3        3
32973
4   KP281   20    Male         13     Partnered      4        2
35247

   Miles     age_group            edu_group income_group
miles_group
0    112  Young Adults  Secondary Education   Low Income   Active
```

```
Lifestyle
1      75   Young Adults   Secondary Education   Low Income   Moderate
Activity
2      66   Young Adults   Secondary Education   Low Income   Moderate
Activity
3      85   Young Adults     Primary Education   Low Income   Moderate
Activity
4      47   Young Adults   Secondary Education   Low Income      Light
Activity
```

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

Duplicate Detection

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

False    180
dtype: int64
```

Insights

1. There are no duplicate entries in the dataset.

Value Count check for Columns

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

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

1. There are 3 unique products available in the dataset.

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

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

1. There are 104 male and 76 female available in the dataset.

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

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

1. There are 107 Partnered and 73 single Male and Female available in the dataset.

Unique Values check for all columns

```
# 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]
======================================================================
Unique Values in age_group column are :-
```

```
['Young Adults', 'Adults', 'Middle Aged Adults', 'Elder']
Categories (4, object): ['Young Adults' < 'Adults' < 'Middle Aged
Adults' < 'Elder']
======================================================================
Unique Values in edu_group column are :-
['Secondary Education', 'Primary Education', 'Higher Education']
Categories (3, object): ['Primary Education' < 'Secondary Education' <
'Higher Education']
======================================================================
Unique Values in income_group column are :-
['Low Income', 'Moderate Income', 'High Income', 'Very High Income']
Categories (4, object): ['Low Income' < 'Moderate Income' < 'High
Income' < 'Very High Income']
======================================================================
Unique Values in miles_group column are :-
['Active Lifestyle', 'Moderate Activity', 'Light Activity', 'Fitness
Enthusiast ']
Categories (4, object): ['Light Activity' < 'Moderate Activity' <
'Active Lifestyle' <
                         'Fitness Enthusiast ']
======================================================================
```

Insights

1. The dataset does not contain any abnormal values.

3. Visual Analysis - Univariate & Bivariate

Univariate Analysis

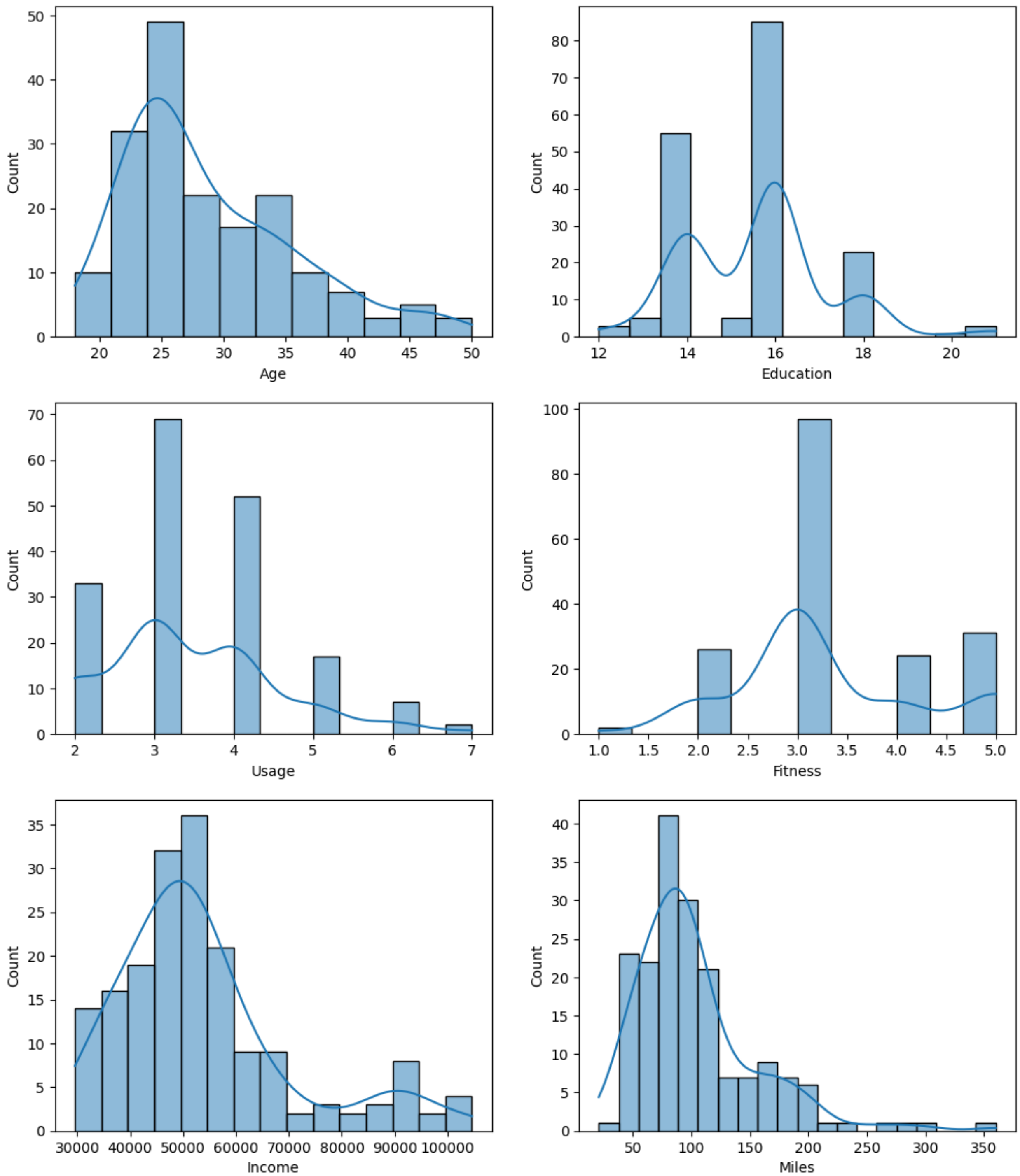
For continuous variable(s):

Understanding the distribution of the data for the quantitative attributes:

1. Age
2. Education
3. Usage
4. Fitness
5. Income
6. Miles

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)
```

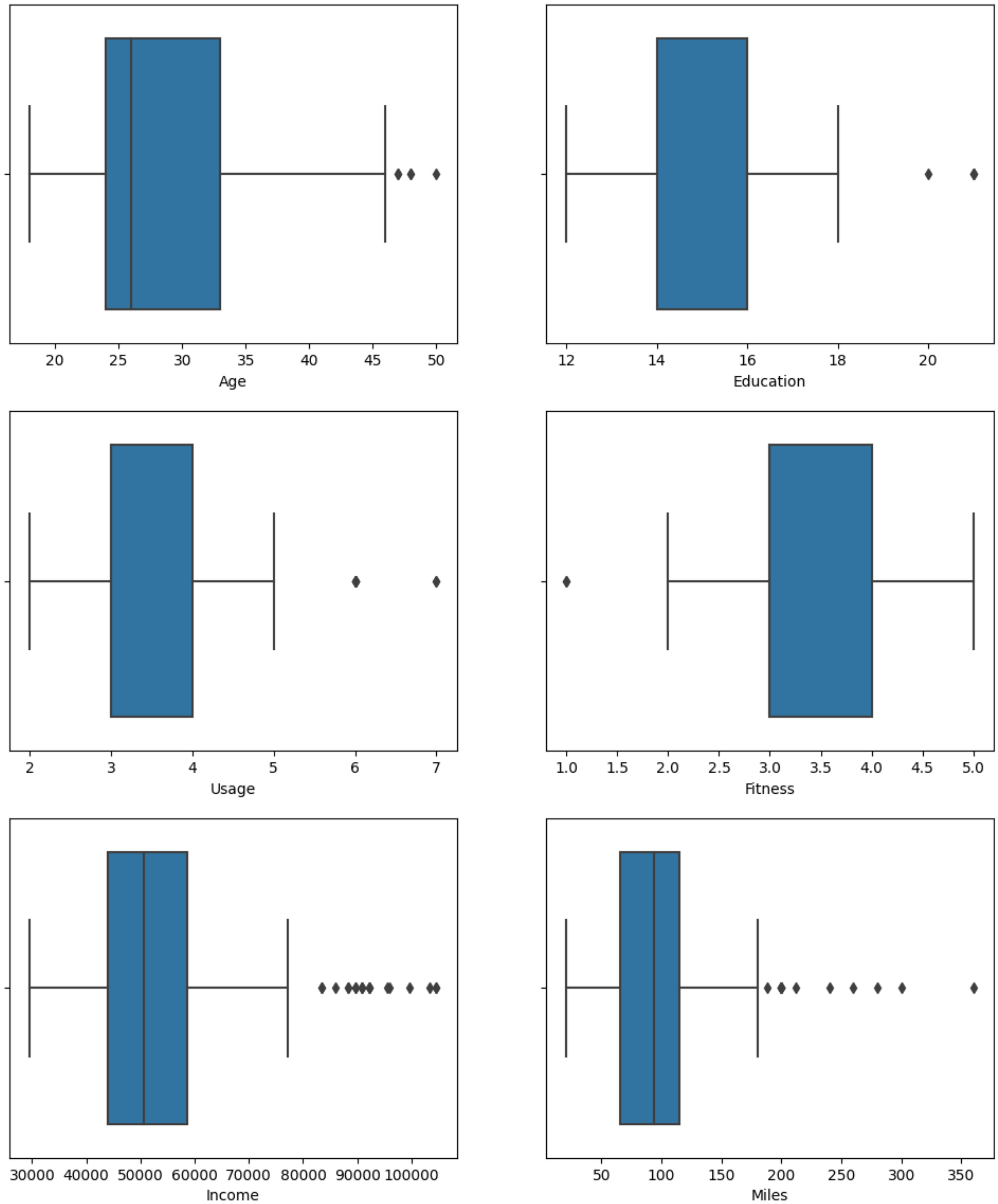
```python
sns.histplot(data=df, x="Age", kde=True, ax=axis[0,0])
sns.histplot(data=df, x="Education", kde=True, ax=axis[0,1])
sns.histplot(data=df, x="Usage", kde=True, ax=axis[1,0])
sns.histplot(data=df, x="Fitness", kde=True, ax=axis[1,1])
sns.histplot(data=df, x="Income", kde=True, ax=axis[2,0])
sns.histplot(data=df, x="Miles", kde=True, ax=axis[2,1])
plt.show()
```



Outliers detection using BoxPlots

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x="Age", ax=axis[0,0])
sns.boxplot(data=df, x="Education", ax=axis[0,1])
```

```
sns.boxplot(data=df, x="Usage", ax=axis[1,0])
sns.boxplot(data=df, x="Fitness", ax=axis[1,1])
sns.boxplot(data=df, x="Income", ax=axis[2,0])
sns.boxplot(data=df, x="Miles" , ax=axis[2,1])
plt.show()
```



Insight

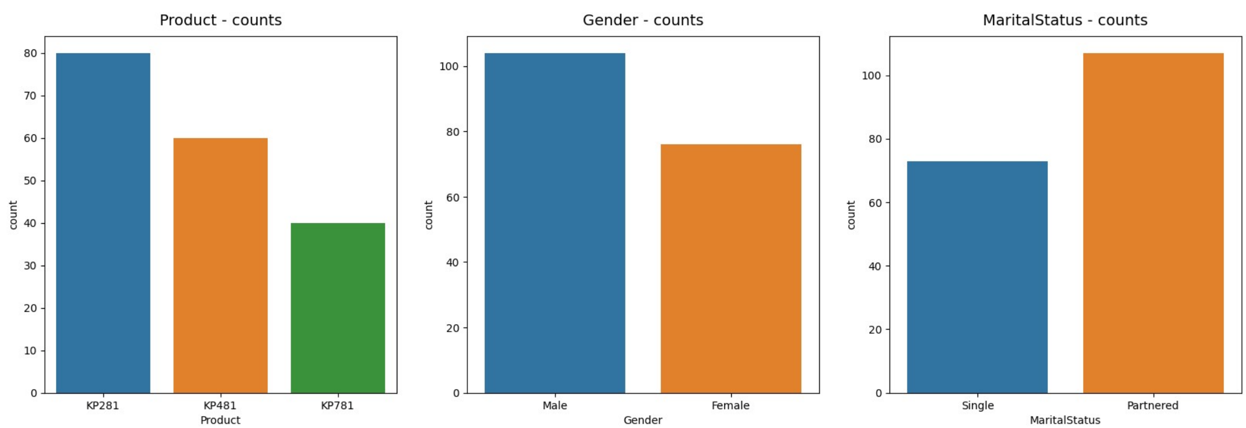
Even from the boxplots it is quite clear that:

1. Age, Education and Usage are having very few outliers.
2. While Income and Miles are having more outliers.

Understanding the distribution of the data for the qualitative attributes:

1. Product
2. Gender
3. Marital Status

```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(20, 6))
sns.countplot(data=df, x='Product', ax=axs[0])
sns.countplot(data=df, x='Gender', ax=axs[1])
sns.countplot(data=df, x='MaritalStatus', ax=axs[2])

axs[0].set_title("Product - counts", pad=10, fontsize=14)
axs[1].set_title("Gender - counts", pad=10, fontsize=14)
axs[2].set_title("MaritalStatus - counts", pad=10, fontsize=14)
plt.show()
```



Insight

1. KP281 is the most frequent product.
2. Thare are more Males in the data than Females.
3. More Partnered persons are there in the data.

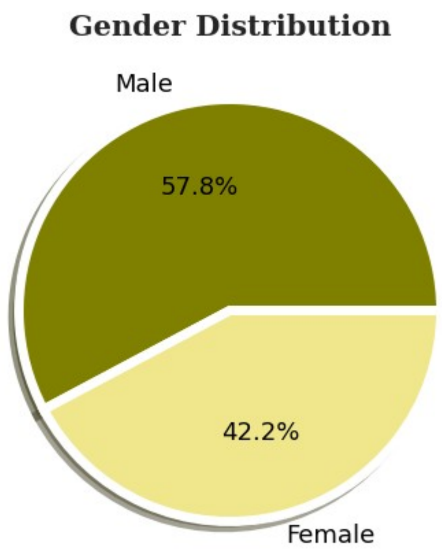
To be precise - normalized count for each variable is shown below

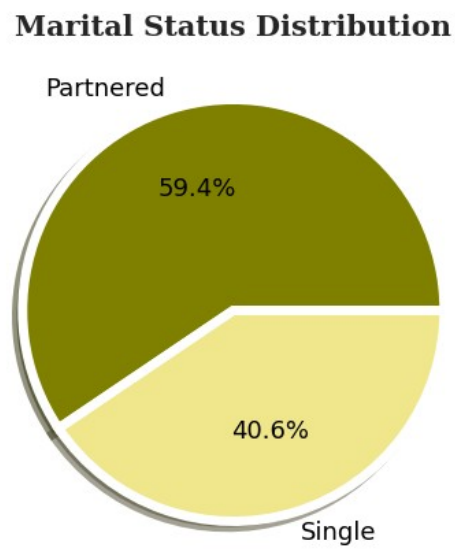
```
df1 = df[['Product', 'Gender', 'MaritalStatus']].melt()
df1.groupby(['variable', 'value'])[['value']].count() / len(df)
```

variable	value	value
Gender	Female	0.422222
	Male	0.577778
MaritalStatus	Partnered	0.594444
	Single	0.405556
Product	KP281	0.444444
	KP481	0.333333
	KP781	0.222222

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 = ['#808000',"#F0E68C"]
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 = ['#808000',"#F0E68C"]
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()
```





Insight

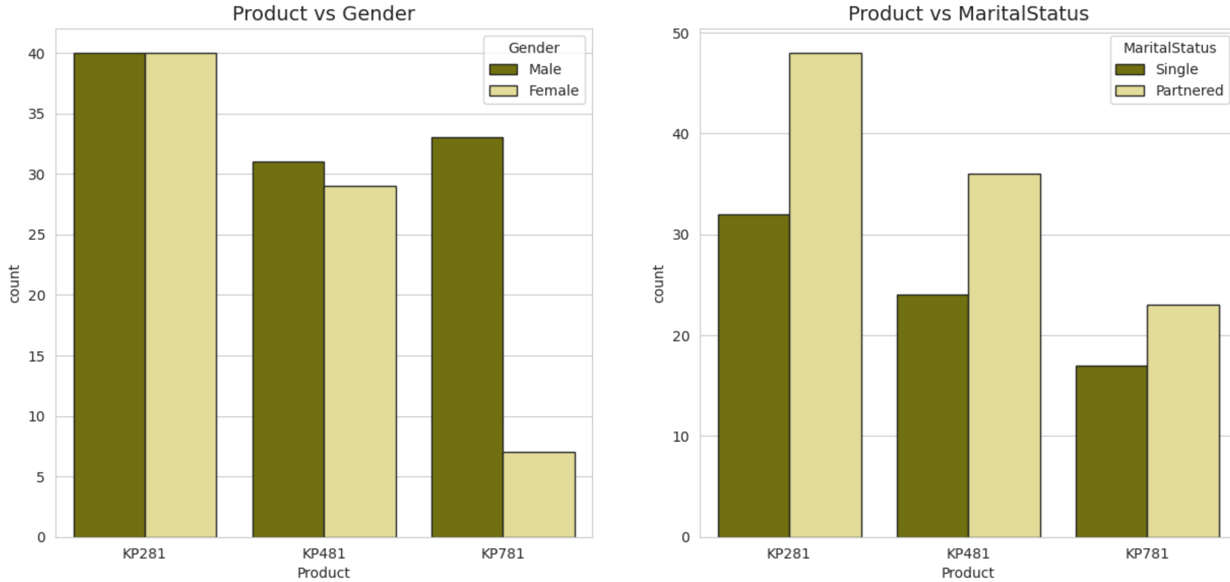
- **Product**
1. 44.44% of the customers have purchased KP2821 product.
1. 33.33% of the customers have purchased KP481 product.
1. 22.22% of the customers have purchased KP781 product.
- **Gender**
1. 57.78% of the customers are Male.
- **MaritalStatus**
1. 59.44% of the customers are Partnered.

Bivariate Analysis

For categorical variable(s):

Checking if features - Gender or MaritalStatus have any effect on the product purchased.

```
sns.set_style(style='whitegrid')
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(15, 6.5))
sns.countplot(data=df, x='Product', hue='Gender', edgecolor="0.15",
palette=['#808000',"#F0E68C"], ax=axs[0])
sns.countplot(data=df, x='Product', hue='MaritalStatus',
edgecolor="0.15", palette=['#808000',"#F0E68C"], ax=axs[1])
axs[0].set_title("Product vs Gender", fontsize=14)
axs[1].set_title("Product vs MaritalStatus", fontsize=14)
plt.show()
```



Insight

Product vs Gender

- Equal number of males and females have purchased KP281 product and Almost same for the product KP481
- Most of the Male customers have purchased the KP781 product.

Product vs MaritalStatus

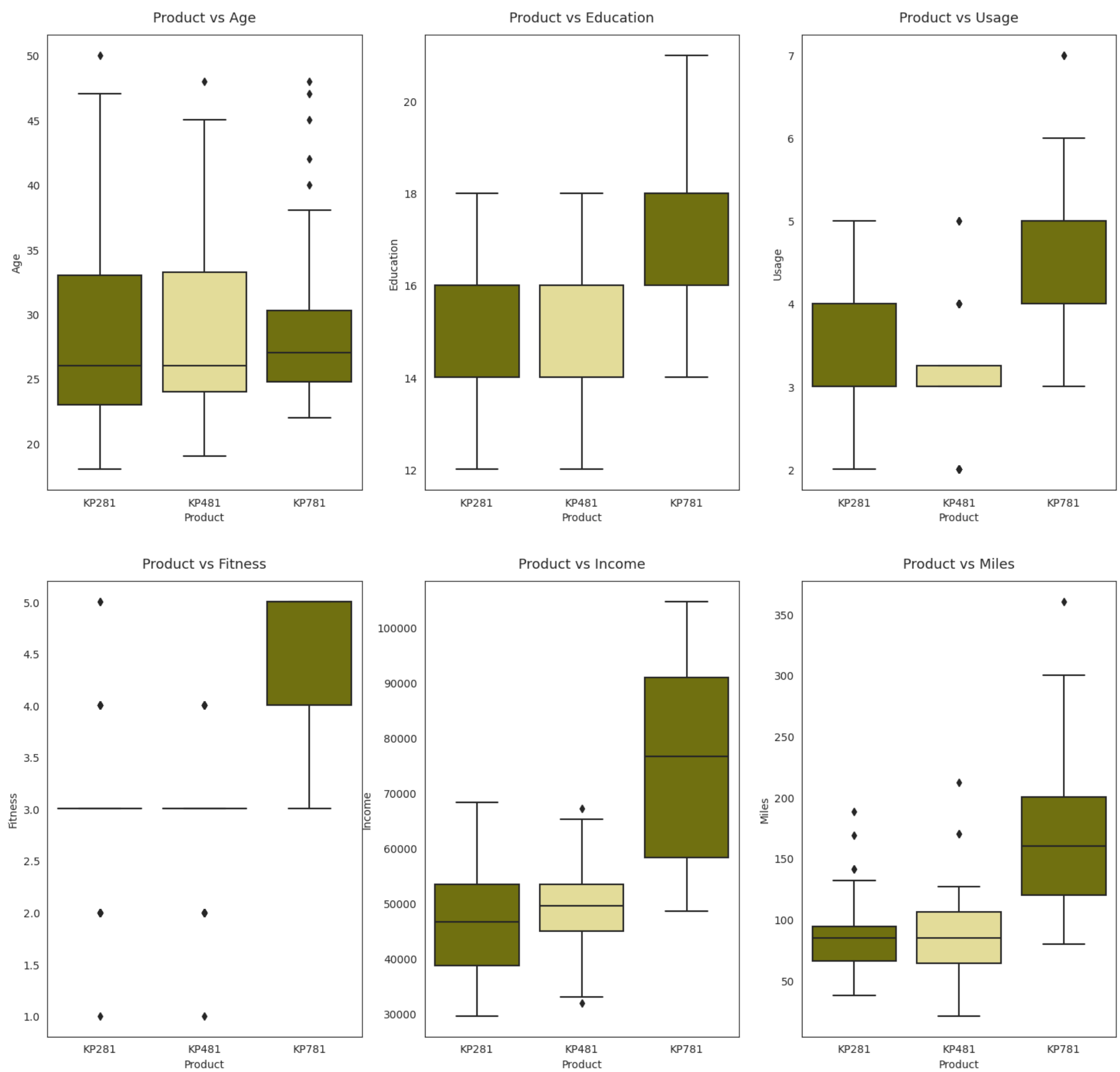
- Customer who is Partnered, is more likely to purchase the product.

Checking if following features have any effect on the product purchased:

1. Age
2. Education
3. Usage
4. Fitness
5. Income
6. Miles

```
attrs = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
sns.set_style("white")
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(18, 12))
fig.subplots_adjust(top=1.2)
count = 0
for i in range(2):
    for j in range(3):
        sns.boxplot(data=df, x='Product', y=attrs[count], ax=axs[i,j],
```

```
palette=['#808000',"#F0E68C"])
        axs[i,j].set_title(f"Product vs {attrs[count]}", pad=12,
fontsize=13)
        count += 1
```



Insights

1. Product vs Age

- Customers purchasing products KP281 & KP481 are having same Age median value.
- Customers whose age lies between 25-30, are more likely to buy KP781 product

2. Product vs Education

- Customers whose Education is greater than 16, have more chances to purchase the KP781 product.

- While the customers with Education less than 16 have equal chances of purchasing KP281 or KP481.

3. Product vs Usage

- Customers who are planning to use the treadmill greater than 4 times a week, are more likely to purchase the KP781 product.
- While the other customers are likely to purchasing KP281 or KP481.

4. Product vs Fitness

- The more the customer is fit (fitness >= 3), higher the chances of the customer to purchase the KP781 product.

5. Product vs Income

- Higher the Income of the customer (Income >= 60000), higher the chances of the customer to purchase the KP781 product.

6. Product vs Miles

- If the customer expects to walk/run greater than 120 Miles per week, it is more likely that the customer will buy KP781 product.

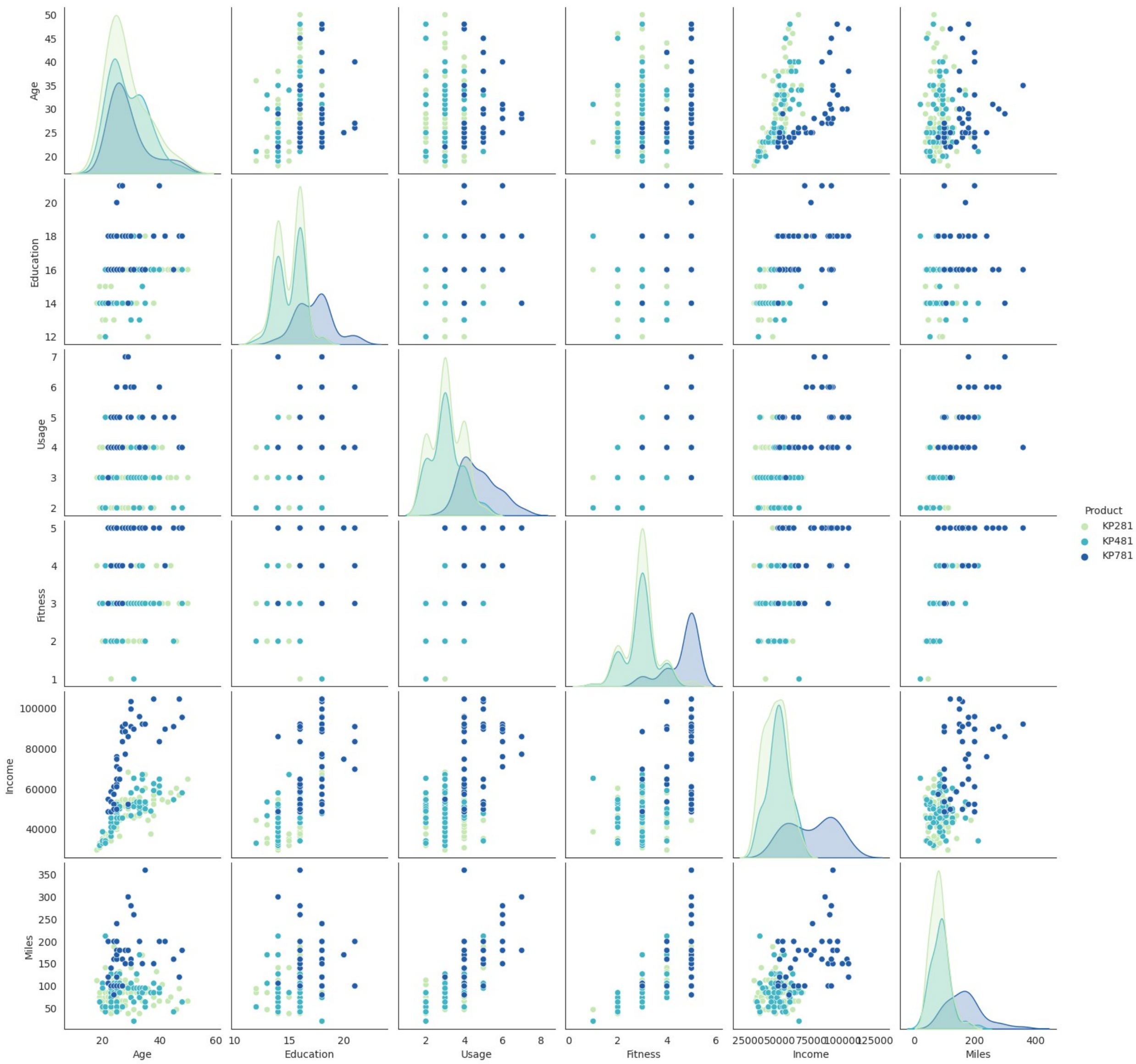
3.3 For Correlation: Heatmaps, Pairplots

Correlation between Variables

3.3.1 Pairplot

A pairplot plot a pairwise relationships in a dataset. The pairplot function creates a grid of Axes such that each variable in data will by shared in the y-axis across a single row and in the x-axis across a single column.

```
df_copy = df

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



3.3.2 Heatmap

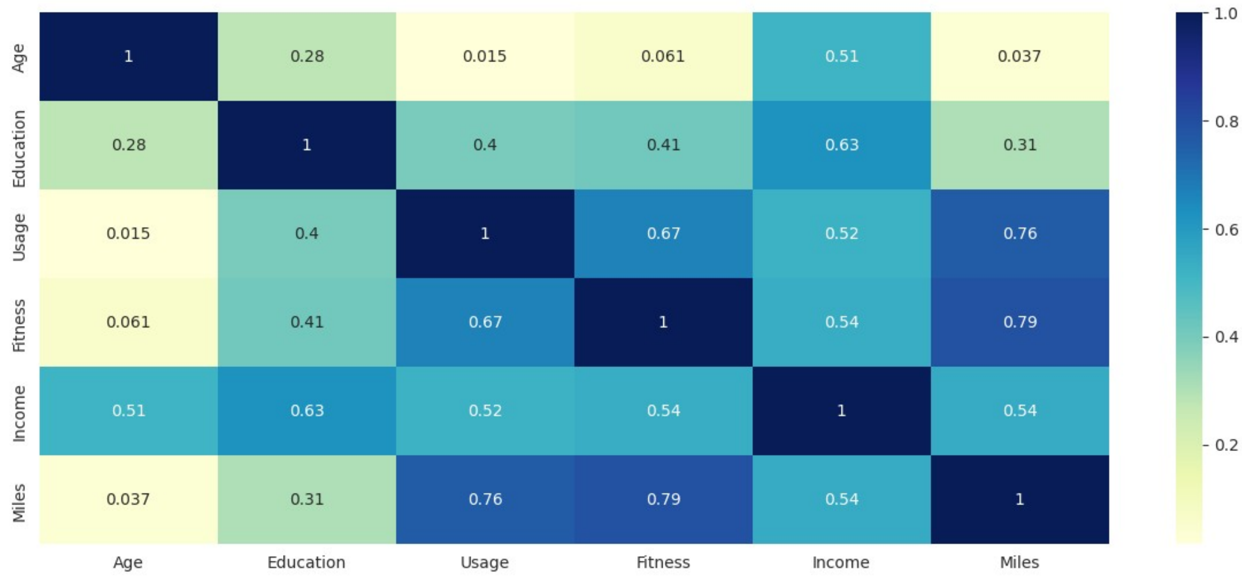
A heatmap is a plot of rectangular data as a color-encoded matrix. As parameter it takes a 2D dataset. That dataset can be coerced into an ndarray. This is a great way to visualize data, because it can show the relation between variabels including time.

```
# 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()
```

```
<ipython-input-35-ba0b4211d231>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  corr_mat = df_copy.corr()
```



Insights

1. From the pair plot we can see Age and Income are positively correlated and heatmap also suggests a strong correlation betwwen them

2. Eductaion and Income are highly correlated as its obvious. Eductation also has significatnt correlation between Fitness rating and Usage of the treadmill.

3. Usage is highly correlated with Fitness and Miles as more the usage more the fitness and mileage.