```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/drive/MyDrive/Probs_file.csv')
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
...	...	...	...	...	...	...	...	...	...
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

180 rows × 9 columns

```
df.info
```

```
<bound method DataFrame.info of     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]>
```

```
print(f'Number of rows :{df.shape[0]} \nNumber of columns: {df.shape[1]}')
```

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

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

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	
count	180	180.000000	180	180.000000	180	180.000000	180.000000	180.000000	18
unique	3	NaN	2	NaN	2	NaN	NaN	NaN	
top	KP281	NaN	Male	NaN	Partnered	NaN	NaN	NaN	
freq	80	NaN	104	NaN	107	NaN	NaN	NaN	
mean	NaN	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53719.577778	10
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16506.684226	5
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29562.000000	2

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

```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58668.000000	11

Observations

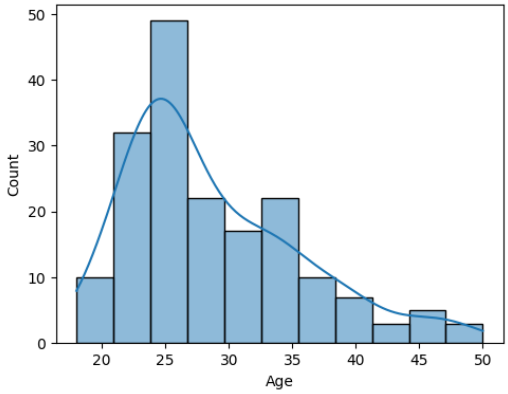
- There are 3 unique items in the dataset
- No missing values are present
- Max and min ages are 50 and 18 respectively. Mean age is about 28.78
- Most of the machines are bought by males (i.e 104) and the rest are female
- The top item of purchase is KP281 with a frequency of 80
- The majority of people are partnered (freq= 107)
- The education mean is about 16 years
- The std deviation of Income and Miles is quite high compared to the rest.

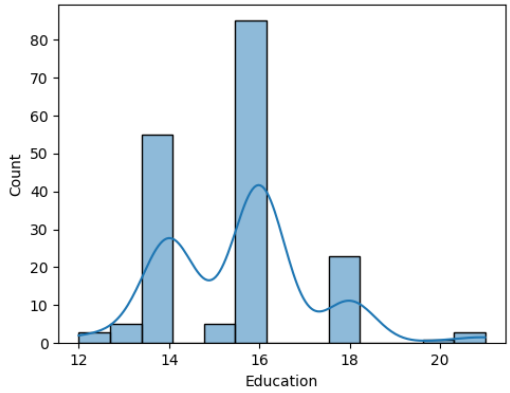
Analysis of data - Univariate

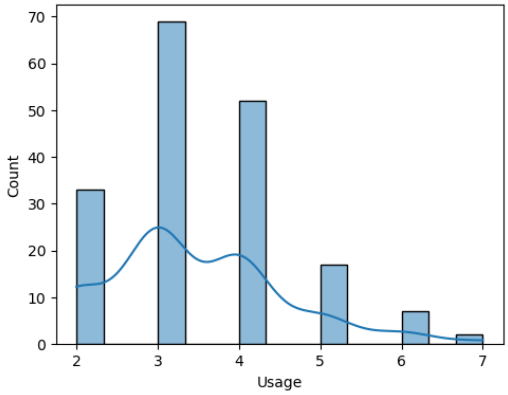
Understanding data distribution for different numerical attributes

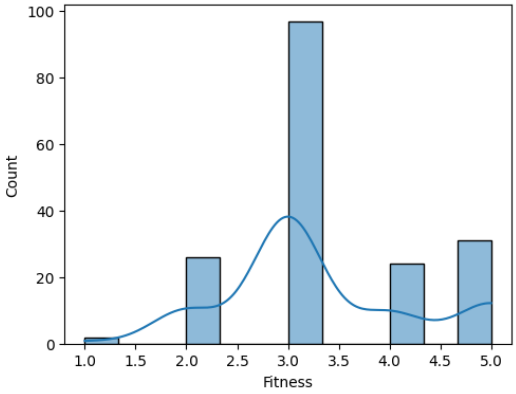
- Age
- Education
- Usage
- Fitness
- Income
- Miles

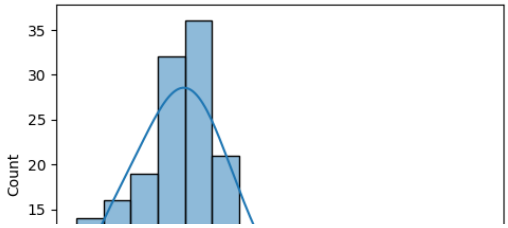
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

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()
```

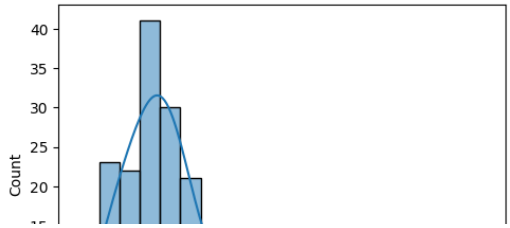






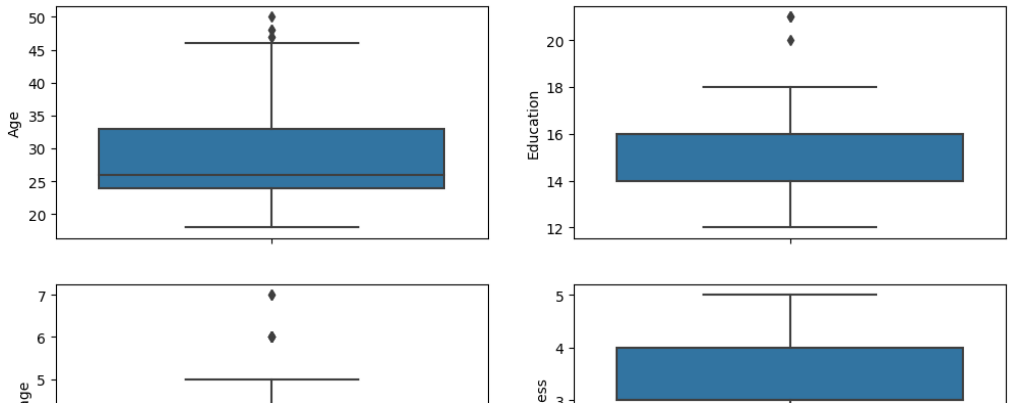






Outlier Detection using BoxPlots

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

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

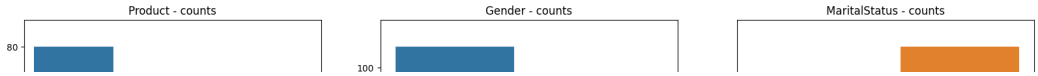


From the above plot it can be inferred that the presence of Outliers is close to negligible in all colums except the Income and Miles columns.

Understanding Data Distribution for Categorical Attributes

- Gender
- Product
- Marital Status

```
fig,axis= plt.subplots(nrows=1, ncols=3,figsize=(20,8))
fig.subplots_adjust(top=1.2)

sns.countplot(data=df, x='Product', ax=axis[0])
sns.countplot(data=df, x='Gender', ax=axis[1])
sns.countplot(data=df, x='MaritalStatus', ax=axis[2])

axis[0].set_title("Product - counts", )
axis[1].set_title("Gender - counts", )
axis[2].set_title("MaritalStatus - counts")
plt.show()
```



- The highest count is for the product KP281
- More males buy the machines compared to females
- Majority of the purchases are done by people with partners

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

		value
variable	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

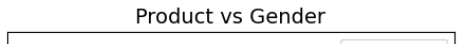
Product

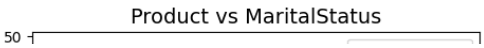
- The product split percentages are 44.4% for KP281, 33.3% for KP481 , 22.2% for KP781
- The percentage of partnered people is 59.4% compared to the 40.55% which is single
- The male to female ratio is 0.577:0.4222

Bivariate Analysis

Checking if features - Gender or Marital Status have any effect on the product purchase

```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(12, 7))
sns.countplot(data=df, x='Product', hue='Gender', ax=axs[0])
sns.countplot(data=df, x='Product', hue='MaritalStatus', ax=axs[1])
axs[0].set_title("Product vs Gender", fontsize=14)
axs[1].set_title("Product vs MaritalStatus", fontsize=14)
plt.show()
```

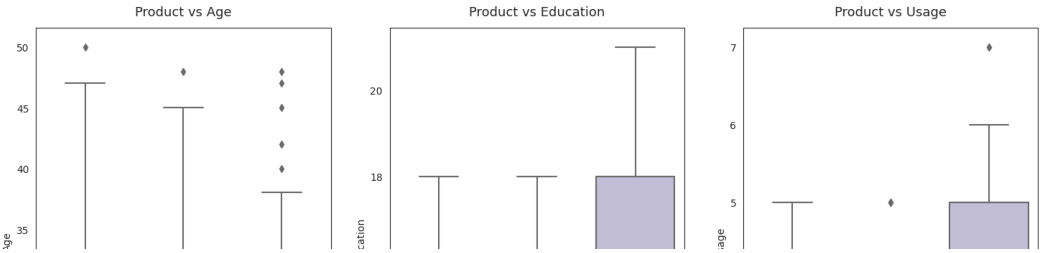




- The male to female product purchases are similar across all the products except for KP781
- Partnered customers have more purchases compared to singles across all products

Checking if numerical attributes have an effect on product purchase.

```
att = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
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=att[count], ax=axs[i,j], palette='Set3')
        axs[i,j].set_title(f"Product vs {att[count]}", pad=12, fontsize=13)
        count += 1
```



- Product vs Age

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

- Product vs Education

1. Customers whose Education is greater than 16, have more chances to purchase the KP781 product.
2. While the customers with Education less than 16 have equal chances of purchasing KP281 or KP481.

- Product vs Usage

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

- Product vs Fitness

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

- Product vs Income

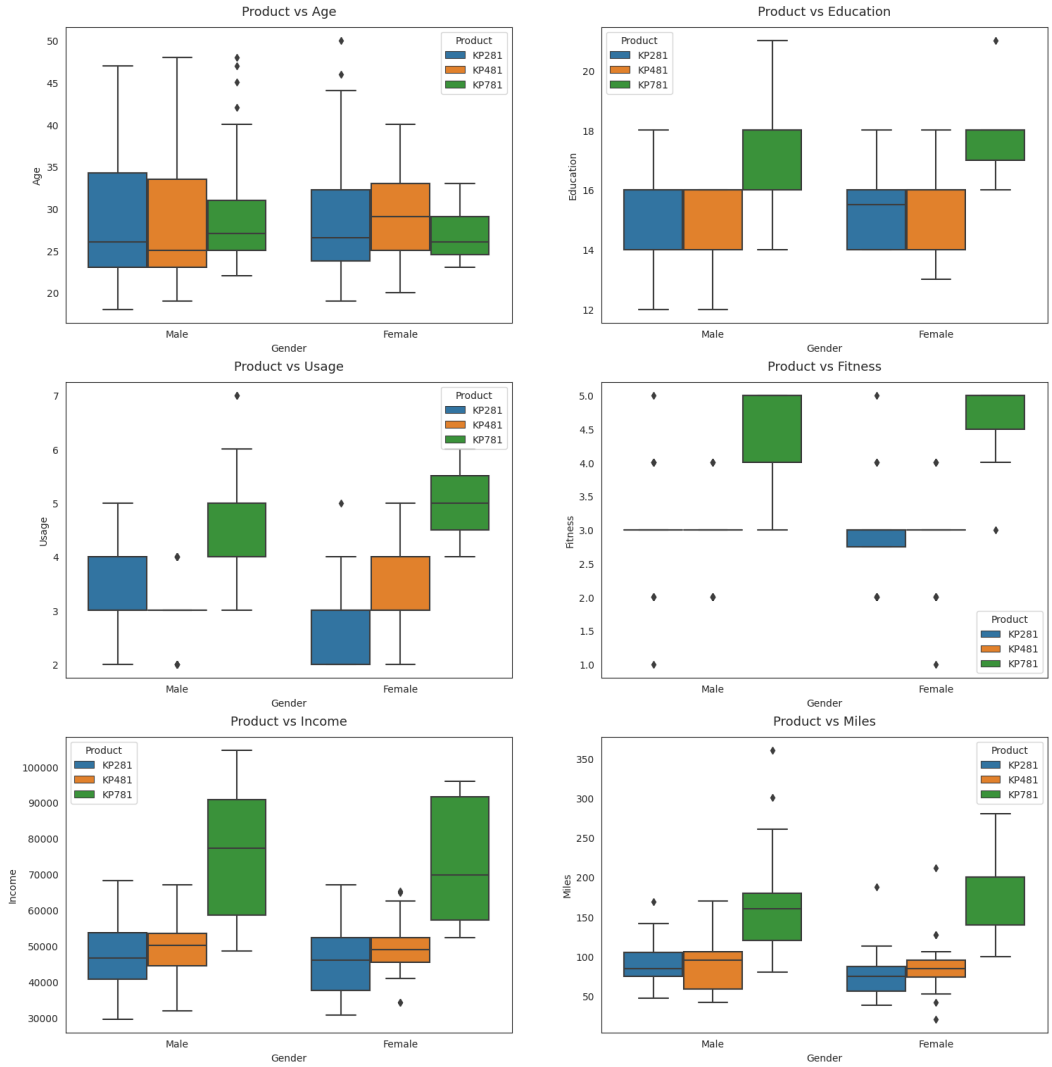
1. Higher the Income of the customer (Income >= 60000), higher the chances of the customer to purchase the KP781 product.

- Product vs Miles

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

Multivariate Analysis

```
att = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(18, 12))
fig.subplots_adjust(top=1.3)
count = 0
for i in range(3):
    for j in range(2):
        sns.boxplot(data=df, x='Gender', y=att[count], hue='Product', ax=axs[i,j])
        axs[i,j].set_title(f"Product vs {att[count]}", pad=12, fontsize=13)
        count += 1
```



Females planning to use treadmill 3-4 times a week, are more likely to buy KP481 product

Marginal Probabilty

```
df['Product'].value_counts(normalize=True)
```

```
KP281    0.444444
KP481    0.333333
KP781    0.222222
Name: Product, dtype: float64
```

Conditional Probabilities

Probability of product given gender

```
def p_prod_given_gender(gender, print_marginal=False):
    if gender != "Female" and gender != "Male":
        return "Invalid gender value."

    df1 = pd.crosstab(index=df['Gender'], columns=[df['Product']])
    p_781 = df1['KP781'][gender] / df1.loc[gender].sum()
    p_481 = df1['KP481'][gender] / df1.loc[gender].sum()
    p_281 = df1['KP281'][gender] / df1.loc[gender].sum()

    if print_marginal:
        print(f"P(Male): {df1.loc['Male'].sum()/len(df):.2f}")
        print(f"P(Female): {df1.loc['Female'].sum()/len(df):.2f}\n")
```

```
    print(f"P(KP781/{gender}): {p_781:.2f}")
    print(f"P(KP481/{gender}): {p_481:.2f}")
    print(f"P(KP281/{gender}): {p_281:.2f}\n")

p_prod_given_gender('Male', True)
p_prod_given_gender('Female')
```

```
 P(Male): 0.58
 P(Female): 0.42

 P(KP781/Male): 0.32
 P(KP481/Male): 0.30
 P(KP281/Male): 0.38

 P(KP781/Female): 0.09
 P(KP481/Female): 0.38
 P(KP281/Female): 0.53
```

Probability of product given Marital Status

```
def p_prod_given_mstatus(status, print_marginal=False):
    if status != "Single" and status != "Partnered":
        return "Invalid marital status value."

    df1 = pd.crosstab(index=df['MaritalStatus'], columns=[df['Product']])
    p_781 = df1['KP781'][status] / df1.loc[status].sum()
    p_481 = df1['KP481'][status] / df1.loc[status].sum()
    p_281 = df1['KP281'][status] / df1.loc[status].sum()

    if print_marginal:
        print(f"P(Single): {df1.loc['Single'].sum()/len(df):.2f}")
        print(f"P(Partnered): {df1.loc['Partnered'].sum()/len(df):.2f}\n")

    print(f"P(KP781/{status}): {p_781:.2f}")
    print(f"P(KP481/{status}): {p_481:.2f}")
    print(f"P(KP281/{status}): {p_281:.2f}\n")

p_prod_given_mstatus('Single', True)
p_prod_given_mstatus('Partnered')
```

```
 P(Single): 0.41
 P(Partnered): 0.59

 P(KP781/Single): 0.23
 P(KP481/Single): 0.33
 P(KP281/Single): 0.44

 P(KP781/Partnered): 0.21
 P(KP481/Partnered): 0.34
 P(KP281/Partnered): 0.45
```

Business Recommendations

- Improve Product Descriptions: Provide clear and concise product descriptions for each treadmill model, highlighting their unique features and benefits. This will assist customers in making informed decisions and better understanding the differences between the products.
- Target Marketing Efforts: Utilize customer data to target marketing efforts more effectively. For example, promote the KP281 model to entry-level fitness enthusiasts, KP481 to mid-level runners, and KP781 to those seeking advanced features and higher performance.
- In-Store Experience: Enhance the in-store experience by providing interactive displays and demos of the different treadmill models. This will engage customers and help them make well-informed decisions.
- Digital Marketing: Leverage digital marketing channels such as social media, online advertisements, and email campaigns to reach a broader audience. Engage with potential customers online and showcase the product range effectively.
- Customer Segmentation: Categorize customers into distinct segments based on their preferences, such as fitness level, age group, and usage patterns. This will help tailor marketing strategies and product recommendations for each segment.
- Focus on Mid-Level Treadmill: Since the KP481 treadmill has a lower price point compared to the advanced KP781 model, it may attract a larger customer base. Consider promoting the mid-level treadmill as an attractive option for cost-conscious customers.