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

```
aero = pd.read_csv("Aerofit_treadmill.csv")
```

```
aero.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

```
aero.shape
```

```
(180, 9)
```

```
aero.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
```

```
aero.duplicated().sum()
```

```
0
```

```
aero.isna().sum()
```

```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

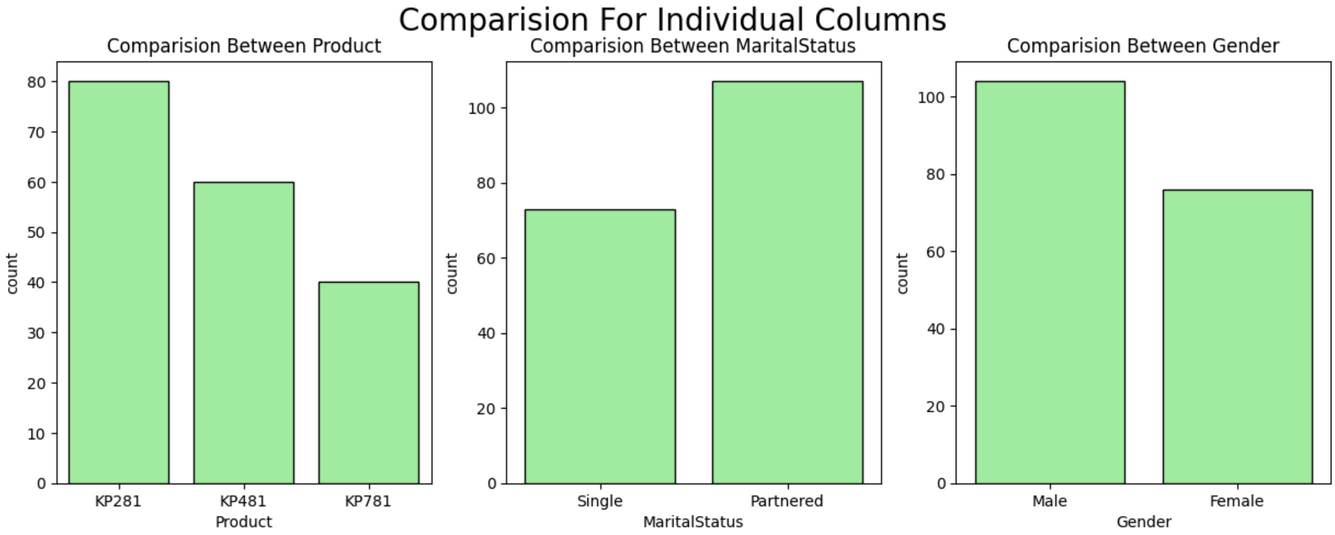
```
aero.describe()
```

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	5.000000	104581.000000	360.000000

Univariate Analysis

Comparision For Individual Columns

```
plt.figure(figsize = (15,5))

plt.subplot(1,3,1)
sns.countplot(data = aero,x = "Product",color = "palegreen",edgecolor = "black")
plt.title("Comparision Between Product")


plt.subplot(1,3,2)
sns.countplot(data = aero,x="MaritalStatus",color = "palegreen",edgecolor = "black")
plt.title("Comparision Between MaritalStatus")

plt.subplot(1,3,3)
sns.countplot(data = aero,x="Gender",color = "palegreen",edgecolor = "black")
plt.title("Comparision Between Gender")

plt.suptitle("Comparision For Individual Columns",fontsize=20)
plt.show()
```



Insights and Recommondation

Comparision Between Product

```
Insights : The KP281 product  is highest sales and KP481 is second highest sales and third highest sales is KP781 product.

Recommondation : Focus On the Low Number of product is Sales to give any offers of discount to the customer to buy this product.
```

Comparision Between MaritalStatus

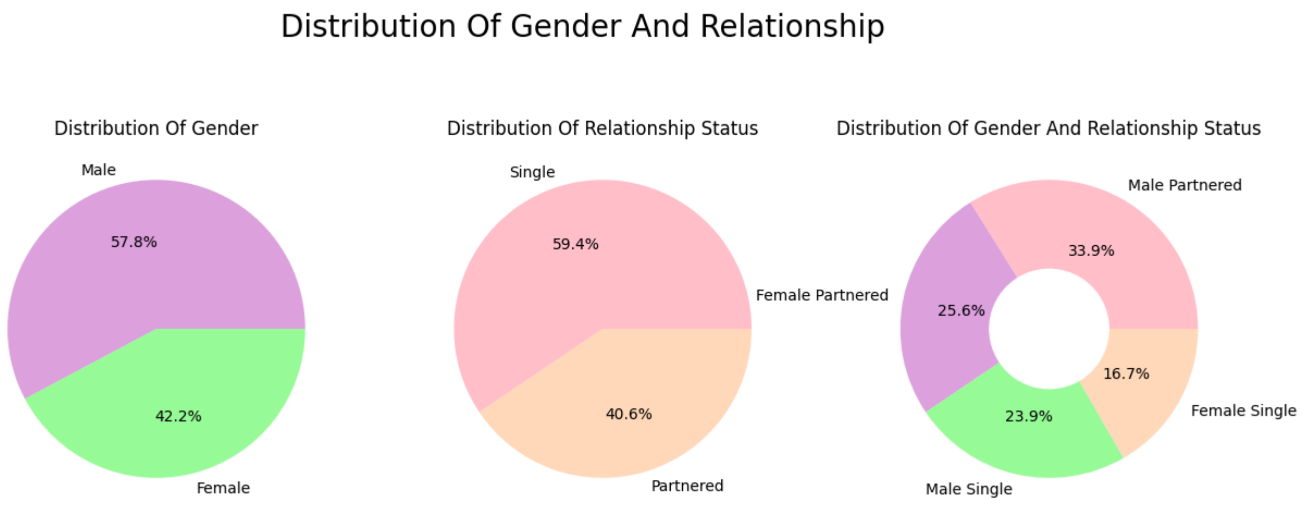
```
Insights : The Partnered Relationship Status is high number of  product is purchased and the Singke Relationship Status is compared
to low number of  product is purchased.

Recommondation : To produced the advertisement Singles related there more possiablity to buy a product singles
```

Comparision Between Gender

```
Insights : The Male is High number of product is Purchased and Female is low number of  product is purchsed compared to Male Gender.

Recommondation : Give me the More discount for Females that become more probability to buying product of females.
```

Distribution Of Gender & Relationship

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

plt.subplot(1,3,1)
plt.pie(aero["Gender"].value_counts(),labels = aero["Gender"].unique(),autopct='%1.1f%%',colors = ["plum","palegreen"])
plt.title("Distribution Of Gender")

plt.subplot(1,3,2)
plt.pie(aero["MaritalStatus"].value_counts(),labels = aero["MaritalStatus"].unique(),autopct='%1.1f%%',colors = ["pink","peachpuff"])
plt.title("Distribution Of Relationship Status")


plt.subplot(1,3,3)
data = aero[["Gender","MaritalStatus"]].value_counts()
data.index = [i[0]+" "+i[1] for i in data.index]
plt.pie(data,labels = data.index,autopct='%1.1f%%',colors = ["pink","plum","palegreen","peachpuff"])
circle = plt.Circle(xy = (0,0),radius = 0.4,color = "white")
plt.gcf()
plt.gca().add_artist(circle)
plt.title("Distribution Of Gender And Relationship Status")

plt.suptitle("Distribution Of Gender And Relationship",fontsize=20)
plt.show()
```



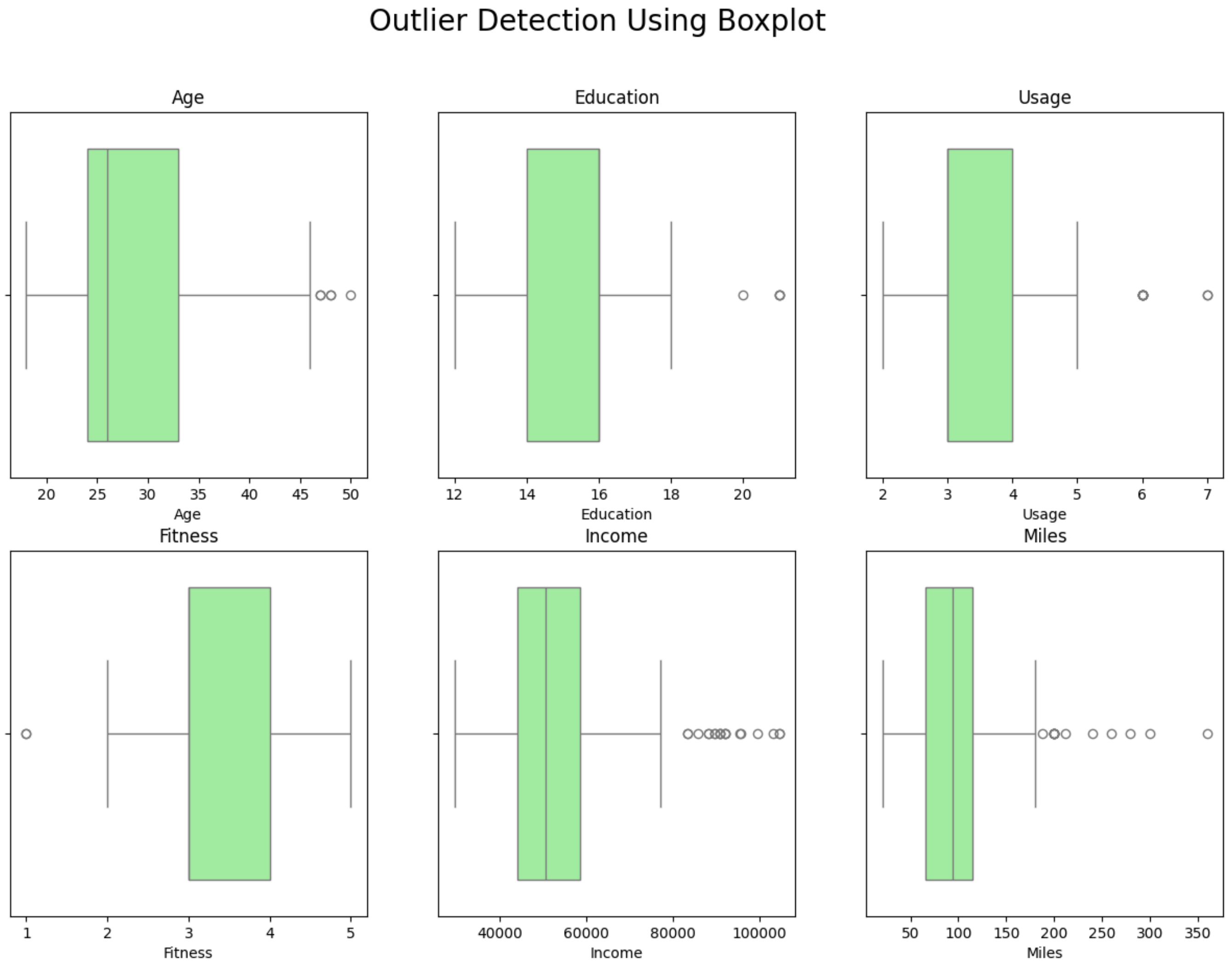
Insights and Recommondation

Distribution Of Gender And Relationship

```
Insights : The more contribution is Male Partnered(33.9%) to purchase the high number of product and less contribution is
           Female singles(16.7%)

Recommondation: To focus on the Female Singles to this particular area Give me offer and complimentary gifts.
```

Outlier Detection Using Boxplot in All Columns

```
plt.figure(figsize = (15,10))

plt.subplot(2,3,1)
sns.boxplot(x = aero["Age"],color = "palegreen")
plt.title("Age")

plt.subplot(2,3,2)
sns.boxplot(x = aero["Education"],color = "palegreen")
plt.title("Education")

plt.subplot(2,3,3)
sns.boxplot(x = aero["Usage"],color = "palegreen")
plt.title("Usage")

plt.subplot(2,3,4)
sns.boxplot(x = aero["Fitness"],color = "palegreen")
plt.title("Fitness")

plt.subplot(2,3,5)
sns.boxplot(x = aero["Income"],color = "palegreen")
plt.title("Income")

plt.subplot(2,3,6)
sns.boxplot(x = aero["Miles"],color = "palegreen")
plt.title("Miles")

plt.suptitle(' Outlier Detection Using Boxplot ', fontsize=20)
plt.show()
```



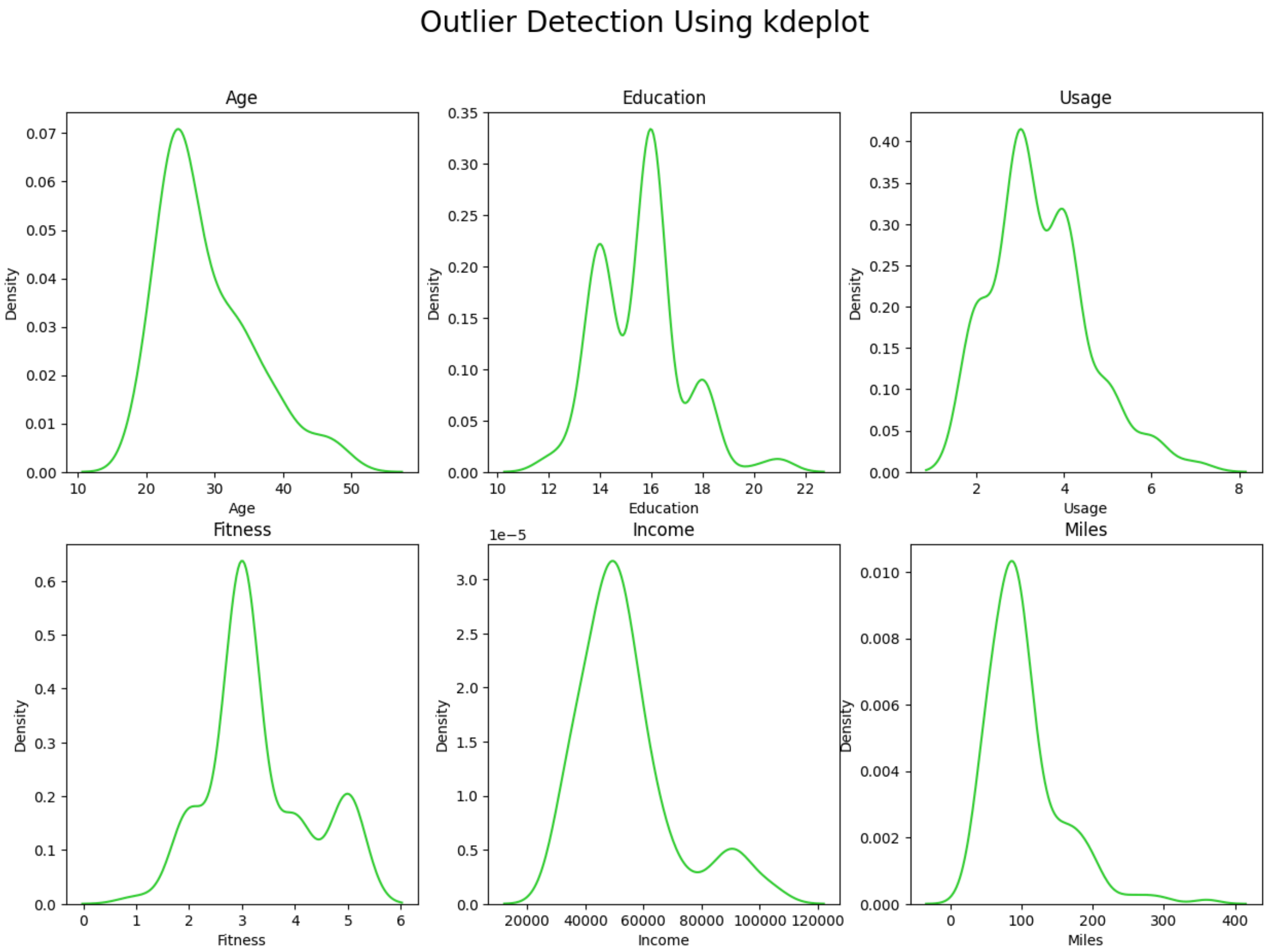
Insights and Recommondation

Outlier Detection Using Boxplot

```
Insights : The Boxplot shows each every columns outliers.
           The more outliers is detecting " Income,Miles " columns
```

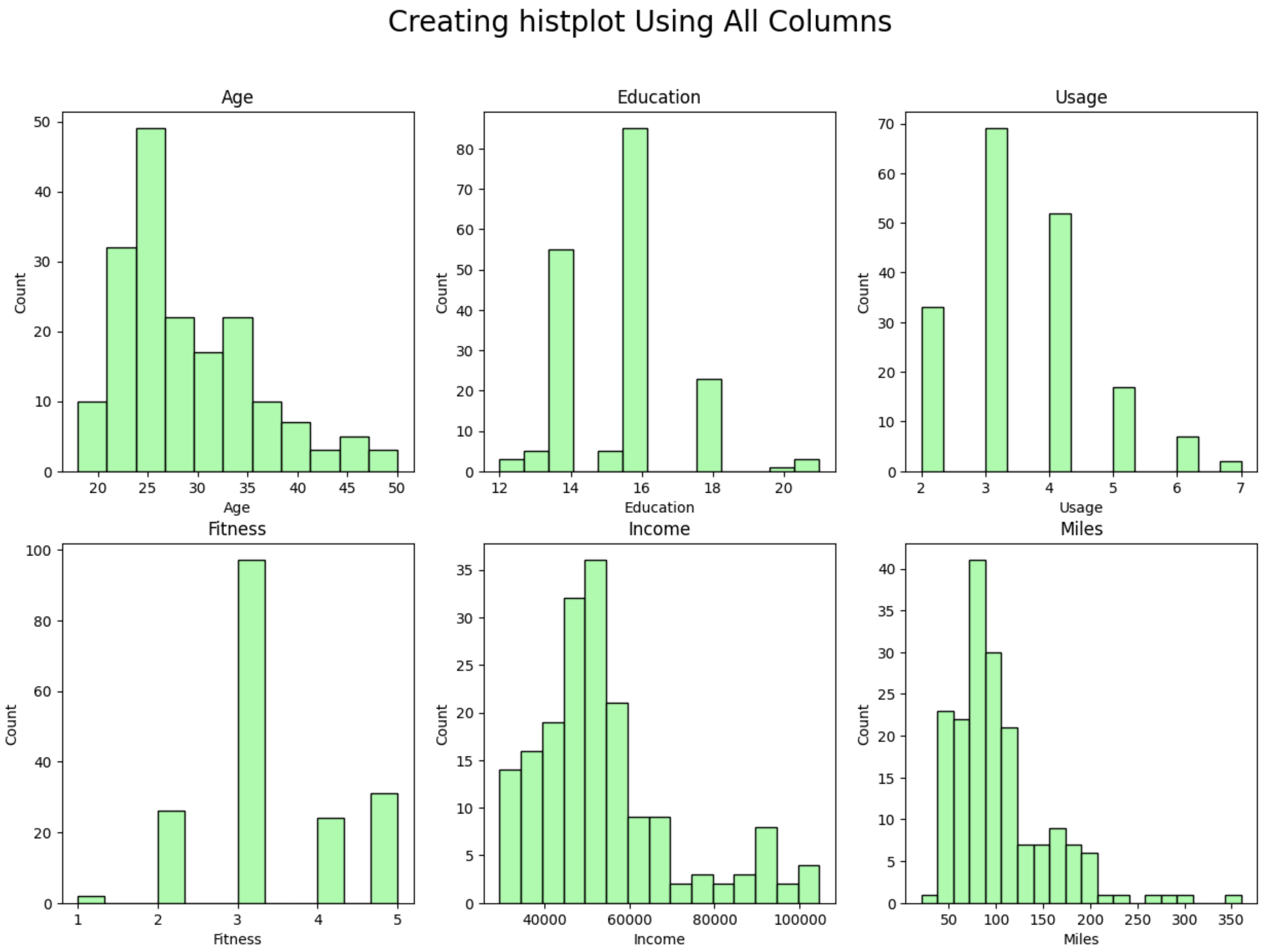
Outlier Detection Using kdeplot in All Columns

```
plt.figure(figsize = (15,10))

plt.subplot(2,3,1)
sns.kdeplot(aero["Age"],color = "limegreen")
plt.title("Age")

plt.subplot(2,3,2)
sns.kdeplot(aero["Education"],color = "limegreen")
plt.title("Education")

plt.subplot(2,3,3)
sns.kdeplot(aero["Usage"],color = "limegreen")
plt.title("Usage")

plt.subplot(2,3,4)
sns.kdeplot(aero["Fitness"],color = "limegreen")
plt.title("Fitness")

plt.subplot(2,3,5)
sns.kdeplot(aero["Income"],color = "limegreen")
plt.title("Income")

plt.subplot(2,3,6)
sns.kdeplot(aero["Miles"],color = "limegreen")
plt.title("Miles")

plt.suptitle(' Outlier Detection Using kdeplot ', fontsize=20)
plt.show()
```



Creating histplot Using All Columns

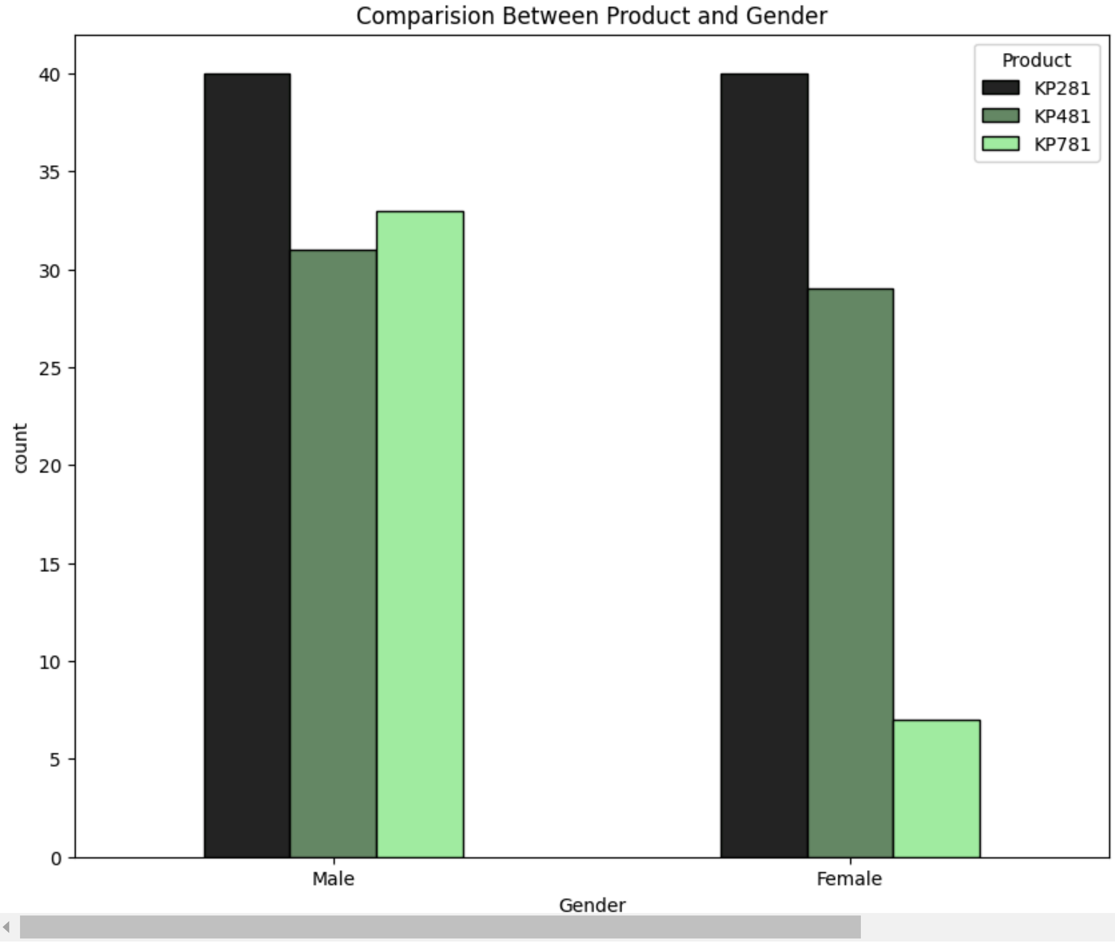
```
plt.figure(figsize = (15,10))

plt.subplot(2,3,1)
sns.histplot(aero["Age"],color = "palegreen",edgecolor = "black")
plt.title("Age")

plt.subplot(2,3,2)
sns.histplot(aero["Education"],color = "palegreen",edgecolor = "black")
plt.title("Education")

plt.subplot(2,3,3)
sns.histplot(aero["Usage"],color = "palegreen",edgecolor = "black")
plt.title("Usage")

plt.subplot(2,3,4)
sns.histplot(aero["Fitness"],color = "palegreen",edgecolor = "black")
plt.title("Fitness")

plt.subplot(2,3,5)
sns.histplot(aero["Income"],color = "palegreen",edgecolor = "black")
plt.title("Income")

plt.subplot(2,3,6)
sns.histplot(aero["Miles"],color = "palegreen",edgecolor = "black")
plt.title("Miles")

plt.suptitle(' Creating histplot Using All Columns ', fontsize=20)
plt.show()
```



Bi-Variate Analysis

Comparision Between Product and Gender

```
plt.figure(figsize = (10,8))
sns.countplot(data = aero,x = "Gender",hue = "Product",dodge = True,width = 0.5,legend = True,palette="dark:palegreen",edgecolor = "blac
plt.title("Comparision Between Product and Gender")
plt.show()
```



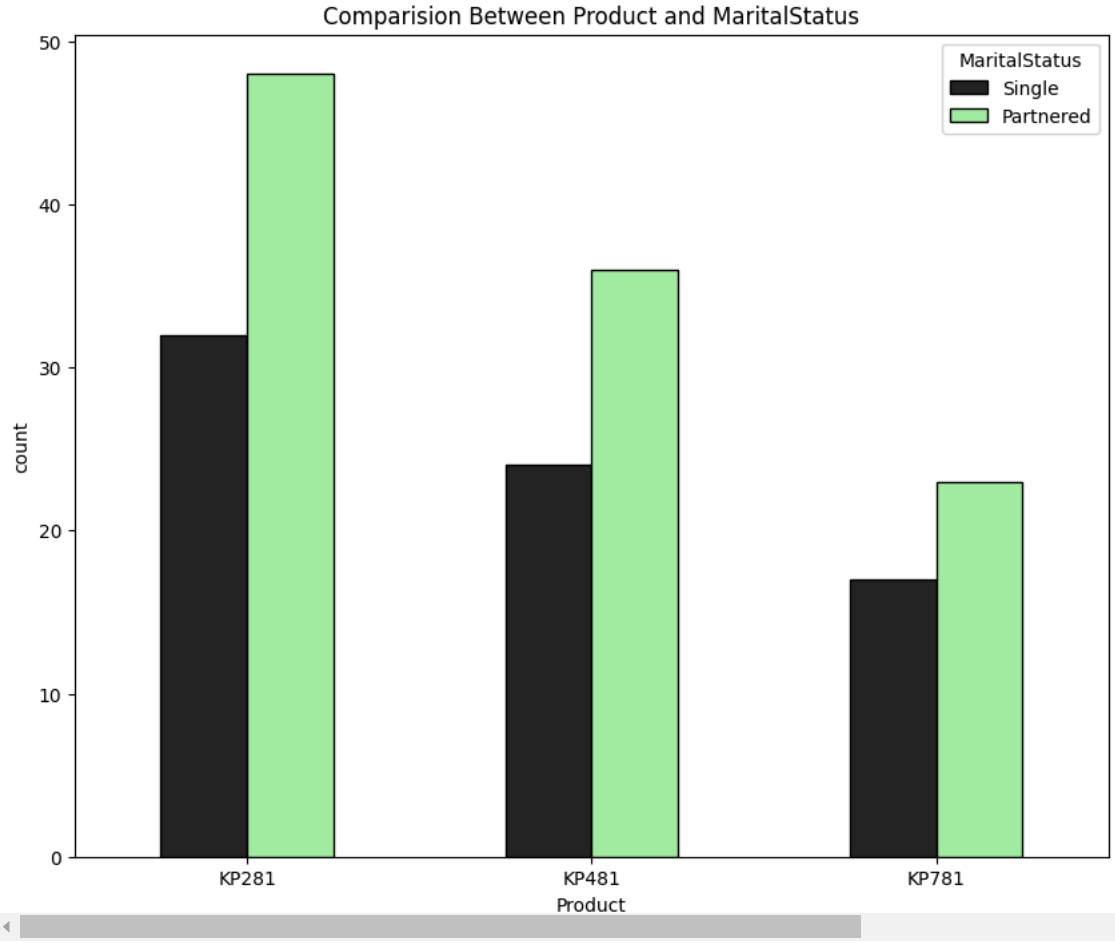
Insights and Recommondation

Comparision Between Product and Gender

```
Insights : The KP 281  is high Number of product is purchased both Male & Female.
           The KP781 Product is less number of Purchased in the females because the KP781 product is more expensive one
               but the male has purchased high number of KP781 product so mens are intrest in fitness more then Women
               so high number of possibility to purchase premium brand machine.

Recommondation: To focus on the Female Side give me trails for the premium brand machine it would become more possibility
                    to purchase premium brand machines.
```

Comparision Between Product and MaritalStatus

```
plt.figure(figsize = (10,8))
sns.countplot(data = aero,hue = "MaritalStatus",x = "Product",dodge = True,width = 0.5,palette="dark:palegreen",edgecolor = "black")
plt.title("Comparision Between Product and MaritalStatus")
plt.show()
```

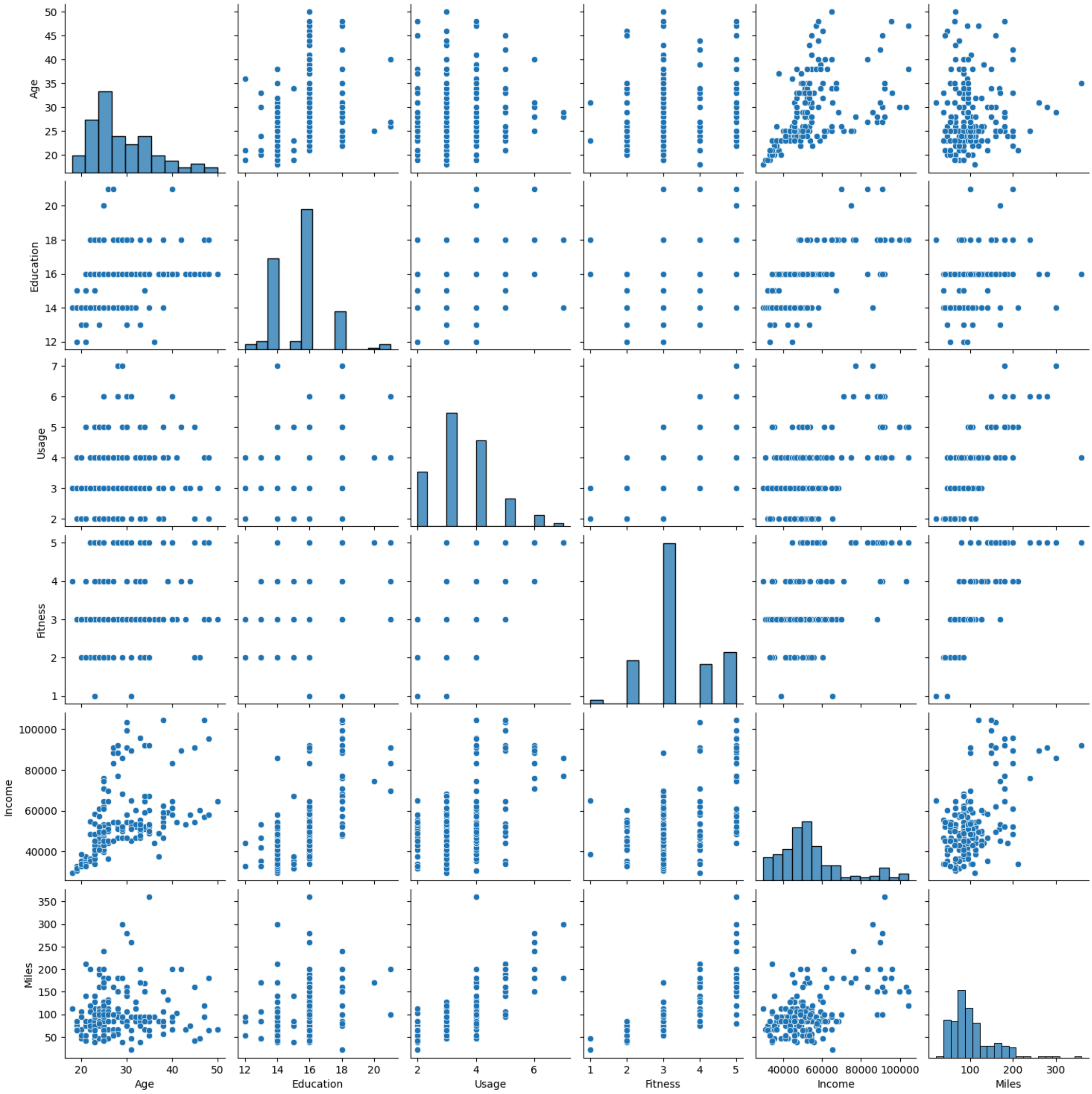


Insights and Recommondation

Comparision Between Product and MaritalStatus

```
Insights :

    Product KP281: For the “Single” marital status, the count is approximately 20.
                   For the “Partnered” marital status, the count is nearly 45.

    Product KP481: Both “Single” and “Partnered” have a count close to 30.

    Product KP781: The count for “Single” is around 10.
                   For “Partnered,” it’s slightly above 5.

Recommondation: TUse this data to analyze how marital status influences product choices.
               Identify potential trends or correlations.
```

```
sns.pairplot(aero)
plt.show()
```



Age
Education
Usage
Fitness
Income
Miles
50
45
40
35
30
25
20
12
14
16
18
20
2
3
4
5
6
7
1
100000
80000
60000
40000
350
300
250
200
150
100

Table

Average Salary of Male and Female of Single and Partnered

```
data = round(aero.pivot_table(values = "Income",index = "MaritalStatus",columns = "Gender",aggfunc = "mean"),2)
data.columns.name = None
data.index.name = None
print(data)
```

```
             Female     Male
Partnered  50693.76  59585.7
Single     48502.80  52274.4
```

Average Income Of People Their Buying Product

```
price_data = pd.DataFrame({"Product":["KP281","KP481","KP781"],"Price":[1500,1750,2500]})

aero_new = pd.merge(aero,price_data,on = "Product")
aero_new.groupby("Product")["Income"].mean().reset_index()
```

	Product	Income
0	KP281	46418.025
1	KP481	48973.650
2	KP781	75441.575

No of Product Buying Age-Wise Category

```
bin_ = [15,20,31,41,51]
label = ["Teenage","Adult","Middle Age","Old Age"]
aero_new["Age-Wise Category"] = pd.cut(aero_new["Age"],bins = bin_,labels = label)

aero_new["Age-Wise Category"].value_counts().reset_index().rename(columns = {"count":"No Of Count"})
```

	Age-Wise Category	No Of Count
0	Adult	116
1	Middle Age	43
2	Old Age	11
3	Teenage	10

No of Product Buying Gender and Age-Wise Category

```
aero_new[["Gender","Age-Wise Category"]].value_counts().reset_index().rename(columns = {"count":"No Of Count"})
```

	Gender	Age-Wise Category	No Of Count
0	Male	Adult	65
1	Female	Adult	51
2	Male	Middle Age	24
3	Female	Middle Age	19
4	Male	Old Age	8
5	Male	Teenage	7
6	Female	Teenage	3
7	Female	Old Age	3

No of Product Buying MaritalStatus and Age-Wise Category

```
aero_new[["MaritalStatus","Age-Wise Category"]].value_counts().reset_index().rename(columns = {"count":"No Of Count"})
```

	MaritalStatus	Age-Wise Category	No Of Count
0	Partnered	Adult	66
1	Single	Adult	50
2	Partnered	Middle Age	29
3	Single	Middle Age	14
4	Partnered	Old Age	8
5	Single	Teenage	6
6	Partnered	Teenage	4
7	Single	Old Age	3

Age-Wise Category people average Income

```
aero_new.groupby("Age-Wise Category")["Income"].mean().reset_index().rename(columns = {"Income":"Avg Income"})
```

```
C:\Users\mssou\AppData\Local\Temp\ipykernel_5396\4176415744.py:1: FutureWarning: The default of observed=False is deprecated and wil
  aero_new.groupby("Age-Wise Category")["Income"].mean().reset_index().rename(columns = {"Income":"Avg Income"})
```

	Age-Wise Category	Avg Income
0	Teenage	33086.700000
1	Adult	51821.172414
2	Middle Age	59090.139535
3	Old Age	71502.272727

Probability and statistics

```
Start coding or generate with AI.
```

```
value = pd.crosstab(aero["Gender"],aero["MaritalStatus"],margins = True)
value.index.name = None
value.columns.name = None
value
```

	Partnered	Single	All
Female	46	30	76
Male	61	43	104
All	107	73	180

Marginal Probabllity

```
# Probability Of P(Male)
print("P(Male) = ",round(value.loc["Male"]["All"]/value.loc["All"]["All"]*100,2))
# Probability Of P(Female)
print("P(Female) = ",round(value.loc["Female"]["All"]/value.loc["All"]["All"]*100,2))
# Total Probability
print("Total Probability = ",round(value.loc["All"]["All"]/value.loc["All"]["All"]*100,2))
```

```
P(Male) =  57.78
P(Female) =  42.22
Total Probability =  100.0
```

```
# Probability Of P(Single)
print("P(Single) = ",round(value["Single"]["All"]/value.loc["All"]["All"]*100,2))
# Probability Of P(Partnered)
print("P(Partnered) = ",round(value["Partnered"]["All"]/value.loc["All"]["All"]*100,2))
# Total Probability
print("Total Probability = ",round(value.loc["All"]["All"]/value.loc["All"]["All"]*100,2))
```

```
P(Single) =  40.56
P(Partnered) =  59.44
Total Probability =  100.0
```

Joint Probability

```
# Probability Of P(Partnered ∩ Male)
print("P(Partnered ∩ Male) = ",round(value.loc["Male"]["Partnered"]/value.loc["All"]["All"]*100,2))
# Probability Of P(Partnered ∩ Female)
print("P(Partnered ∩ Female) = ",round(value.loc["Female"]["Partnered"]/value.loc["All"]["All"]*100,2))
# Total Probability
print("Total Probability Partnered = ",round(value["Partnered"]["All"]/value.loc["All"]["All"]*100,2))
```

```
P(Partnered ∩ Male) =  33.89
P(Partnered ∩ Female) =  25.56
Total Probability Partnered =  59.44
```

```
# Probability Of P(Single ∩ Male)
print("P(Partnered ∩ Male) = ",round(value.loc["Male"]["Single"]/value.loc["All"]["All"]*100,2))
# Probability Of P(Single ∩ Female)
print("P(Partnered ∩ Female) = ",round(value.loc["Female"]["Single"]/value.loc["All"]["All"]*100,2))
# Total Probability Single
print("Total Probability Single = ",round(value["Single"]["All"]/value.loc["All"]["All"]*100,2))
```

```
P(Partnered ∩ Male) =  23.89
P(Partnered ∩ Female) =  16.67
Total Probability Single =  40.56
```

Conditional Probability

```
# Probability Of P(Partnered|Female)
print("P(Partnered|Female) = ",round(value.loc["Female"]["Single"]/value.loc["Female"]["All"]*100,2))
print("P(Partnered|Female) = ",round(value.loc["Female"]["Partnered"]/value.loc["Female"]["All"]*100,2))
```

```
P(Partnered|Female) =  39.47
P(Partnered|Female) =  60.53
```

```
# Probability Of P(Partnered|Male)
print("P(Partnered|Male) = ",round(value.loc["Male"]["Single"]/value.loc["Male"]["All"]*100,2))
# Probability Of P(Partnered|Male)
print("P(Partnered|Male) = ",round(value.loc["Male"]["Partnered"]/value.loc["Male"]["All"]*100,2))
```