Problem Statement

1. Identify the characteristics of the target audience for each type of treadmill offered by the company and provide a better recommendation of the treadmills to the new customers.
2. Investigate 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.

Importing Libraries

In [474]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Analysis of basic data matrix

In [475]:
```
data = pd.read_csv("aerofit_treadmill.csv")
```

In [476]:
```
data.shape
```

Out[476]:
```
(180, 9)
```

In [477]:
```
data.head()
```

Out[477]:

	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

In [478]:
```
data.columns
```

Out[478]:
```
Index(['Product', 'Age', 'Gender', 'Education', 'MaritalStatus', 'Usage',
       'Fitness', 'Income', 'Miles'],
      dtype='object')
```

In [479]:
```
data.dtypes
```

Out[479]:
```
Product          object
Age               int64
Gender           object
Education         int64
MaritalStatus    object
Usage             int64
Fitness           int64
Income            int64
Miles             int64
dtype: object
```

In [480]:
```
data.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
```

In [481]:
```
columnsCat = ['Product', 'Gender', 'MaritalStatus']
for x in columnsCat:
    data[x] = data[x].astype("category")
data.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    category
 1   Age            180 non-null    int64   
 2   Gender         180 non-null    category
 3   Education      180 non-null    int64   
 4   MaritalStatus  180 non-null    category
 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: category(3), int64(6)
memory usage: 9.5 KB
```

In [482]:
```
data.describe()
```

Out[482]:

	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

In [483]:
```
data.describe(include='all')
```

Out[483]:

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

Analysis of Basic data matrix:

1. The shape of given data is (180, 9) which indicates there are total 180 customers data for 9 columns/ characteristics
2. For all the attributes Data types were respectively: Product, Gender and Marital Status -> object/ string data type Age, Education, Usage, Fitness, Income, Miles -> Integer data type
3. Converted Product, Gender and Marital Status attribute to Category type
4. For statistical Summary
   i. KP281 is mostly used product with frequency 80
   ii. Mean Age is 28.79 while Median Age is 26
   iii. Aerofit Products used by Male customers more as compared to others with frequency 104
   iv. Mean Education for customers using products is 15.57 while Median Education is 16 years
   v. Most of the Customers are married with frequency 107
   vi. Mean usage of this products is 3.45 days per week and median is 3 days/week
   vii. Mean fitness for the customers is 3.45 and Median is 3 Which indicates moderate fitness
   viii. Standard deviation of Income is very high so it may contain outliers and Mean Income is 53719.577778
   ix. Miles per week has mean value 103.19 and median value is 94

In [ ]:

```
```

Non Graphical Analysis

In [484]:

```
data.isna().sum()
```

Out[484]:

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

In [485]:
```
data.nunique()
```

Out[485]:
```
Product           3
Age              32
Gender            2
Education         8
MaritalStatus     2
Usage             6
Fitness           5
Income           62
Miles            37
dtype: int64
```

In [486]:
```
for x in data.columns:
    if data[x].dtype.name == "category":
        print(x, "-----" ,data[x].unique())
```

```
Product ----- ['KP281', 'KP481', 'KP781']
Categories (3, object): ['KP281', 'KP481', 'KP781']
Gender ----- ['Male', 'Female']
Categories (2, object): ['Female', 'Male']
MaritalStatus ----- ['Single', 'Partnered']
Categories (2, object): ['Partnered', 'Single']
```

In [487]:
```
values = data['Product'].value_counts()
normalized = data['Product'].value_counts(normalize= True).round(2)
print("Value counts for Products")
print(values)
print("-----------------------------------------------------------------------------------------------------------------------
print("Normalized counts for Products")
print(normalized)
```

```
Value counts for Products
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
------------------------------------------------------------------------------------------------------------------------
Normalized counts for Products
KP281    0.44
KP481    0.33
KP781    0.22
Name: Product, dtype: float64
```

In [488]:
```
values = data['Gender'].value_counts()
normalized = data['Gender'].value_counts(normalize= True).round(2)
print("Value counts for Gender")
print(values)
print("-----------------------------------------------------------------------------------------------------------------------
print("Normalized counts for Gender")
print(normalized)
```

```
Value counts for Gender
Male      104
Female     76
Name: Gender, dtype: int64
------------------------------------------------------------------------------------------------------------------------
Normalized counts for Gender
Male      0.58
Female    0.42
Name: Gender, dtype: float64
```

In [489]:
```
values = data['MaritalStatus'].value_counts()
normalized = data['MaritalStatus'].value_counts(normalize= True).round(2)
print("Value counts for Marital status")
print(values)
print("--------------------------------------------------------------------------------------------------------------
print("Normalized counts for Marital status")
print(normalized)
```

```
Value counts for Marital status
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
-----------------------------------------------------------------------------------------------------------------------------
Normalized counts for Marital status
Partnered    0.59
Single       0.41
Name: MaritalStatus, dtype: float64
```

In [490]:
```
values = data['Fitness'].value_counts()
normalized = data['Fitness'].value_counts(normalize= True).round(2)
print("Value counts for Fitness")
print(values)
print("--------------------------------------------------------------------------------------------------------------
print("Normalized counts for Fitness")
print(normalized)
```

```
Value counts for Fitness
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
-----------------------------------------------------------------------------------------------------------------------------
Normalized counts for Fitness
3    0.54
5    0.17
2    0.14
4    0.13
1    0.01
Name: Fitness, dtype: float64
```

In [491]:
```
values = data['Usage'].value_counts()
normalized = data['Usage'].value_counts(normalize= True).round(2)
print("Value counts for Usage")
print(values)
print("------------------------------------------------------------------------------------------------------------------------
print("Normalized counts for Usage")
print(normalized)
```

```
Value counts for Usage
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
------------------------------------------------------------------------------------------------------------------------
Normalized counts for Usage
3    0.38
4    0.29
2    0.18
5    0.09
6    0.04
7    0.01
Name: Usage, dtype: float64
```

Non Graphical Analysis:

1. There is no null or empty data present in the dataset.
2. Product column has 3 Unique valus which are KP281, KP481 and KP781 out of them KP281 is mostly buyed product with 44% then KP281 with 33 % and KP781 with 22%
3. There are 2 values for Gender column Male and Female which indicates Male customers are greater than female customers from the data 58 % customers are male while 42% are females
4. 59 % of customers are partnered and 41 % are Single
5. From the fitness data analysis around 54% of customers are moderately fit
6. From usage analaysis 38 % of customers uses product 3 days per week and 29 % customers uses 4 days/week

In [ ]:

Graphical Analysis

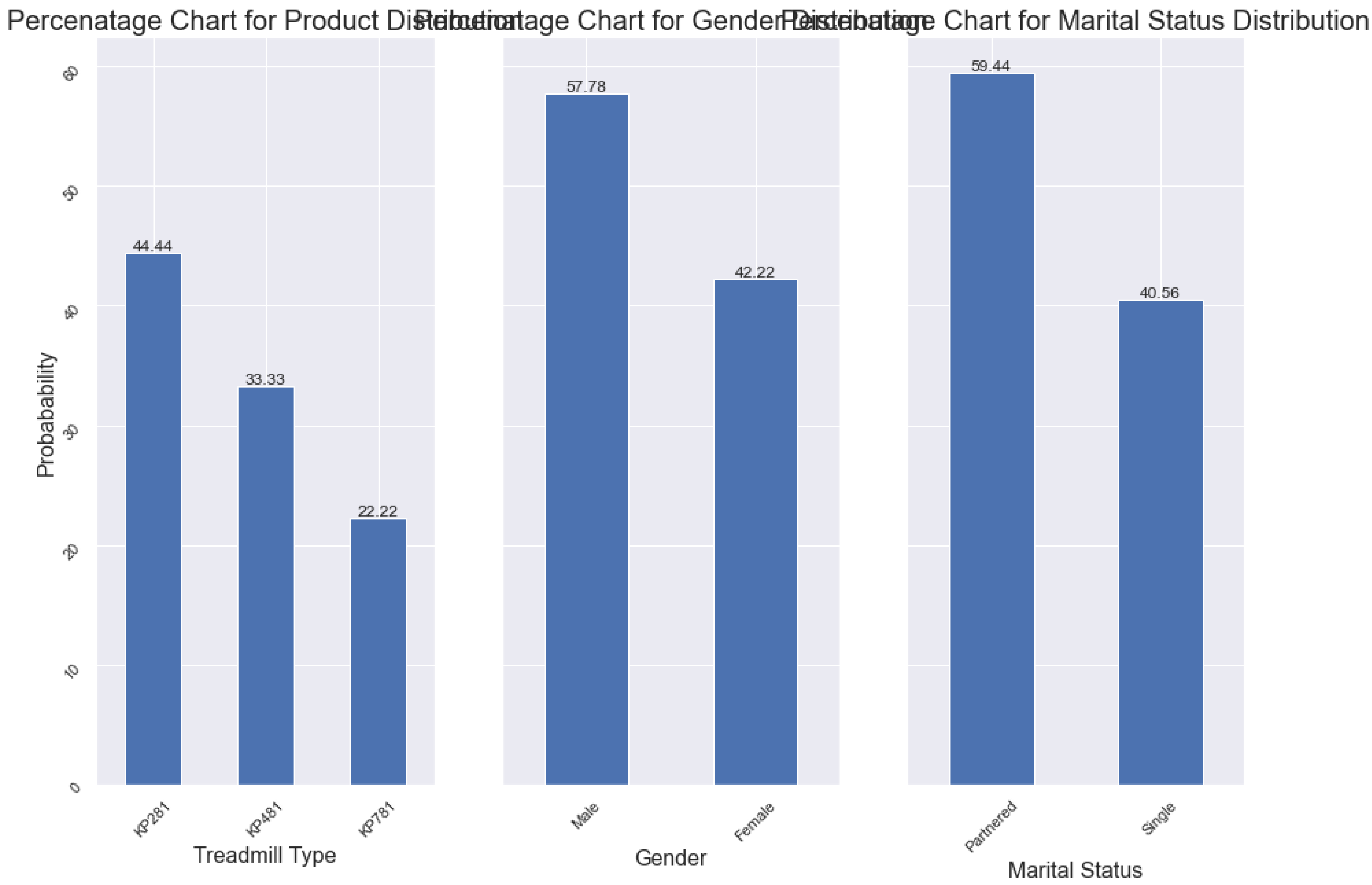
Univariate Analysis

In [492]:

```
fig, axs = plt.subplots(1, 3, figsize=(15, 10), sharey=False)
graph = sns.countplot(x = 'Product', hue= 'Product', data= data, dodge = False , ax = axs[0])
for i in graph.containers:
    graph.bar_label(i,)
graph.set_title("Product Distribution", fontsize = 12)
graph2 = sns.countplot(x = 'Gender', hue = 'Gender', data = data , dodge=False, ax = axs[1])
for i in graph2.containers:
    graph2.bar_label(i,)
graph2.set_title("Gender Distribution", fontsize = 12)
graph3 = sns.countplot(x = 'MaritalStatus', hue = 'MaritalStatus' ,data = data , dodge=False, ax = axs[2])
for i in graph3.containers:
    graph3.bar_label(i,)
graph3.set_title("Marital Status Distribution", fontsize = 12)
plt.show()
```

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

```
In [493]: fig, axs = plt.subplots(1, 3, figsize=(15, 10), sharey=True)
graph = (data['Product'].value_counts(normalize= True).round(4)*100).plot(kind = "bar", ax = axs[0], xlabel ="Treadmill Type", yla
for i in graph.containers:
    graph.bar_label(i,)
graph.set_title("Percenatage Chart for Product Distribution")
graph.tick_params(labelrotation=45)
graph2 = (data['Gender'].value_counts(normalize= True).round(4)*100).plot(kind = "bar", ax = axs[1], xlabel ="Gender", ylabel = "P
for i in graph2.containers:
    graph2.bar_label(i,)
graph2.set_title("Percenatage Chart for Gender Distribution")
graph2.tick_params(labelrotation=45)
graph3 = (data['MaritalStatus'].value_counts(normalize= True).round(4)*100).plot(kind = "bar", ax = axs[2], xlabel ="Marital Statu
for i in graph3.containers:
    graph3.bar_label(i,)
graph3.set_title("Percenatage Chart for Marital Status Distribution")
graph3.tick_params(labelrotation=45)
plt.show()
```



Product

44.44% of the customers have purchased KP2821 product.
33.33% of the customers have purchased KP481 product.
22.22% of the customers have purchased KP781 product.

Gender

57.78% of the customers are Male and rest 42.22% are females.

MaritalStatus

59.44% of the customers are Partnered.

In [ ]:

```

```

In [494]:

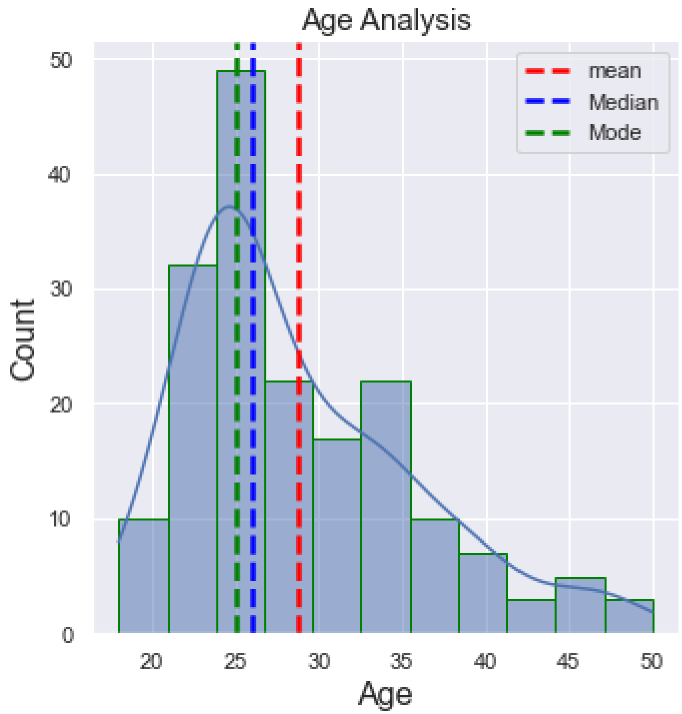
```
data['Age'].describe()
```

Out[494]:

```
count    180.000000
mean      28.788889
std        6.943498
min       18.000000
25%       24.000000
50%       26.000000
75%       33.000000
max       50.000000
Name: Age, dtype: float64
```

In [495]:

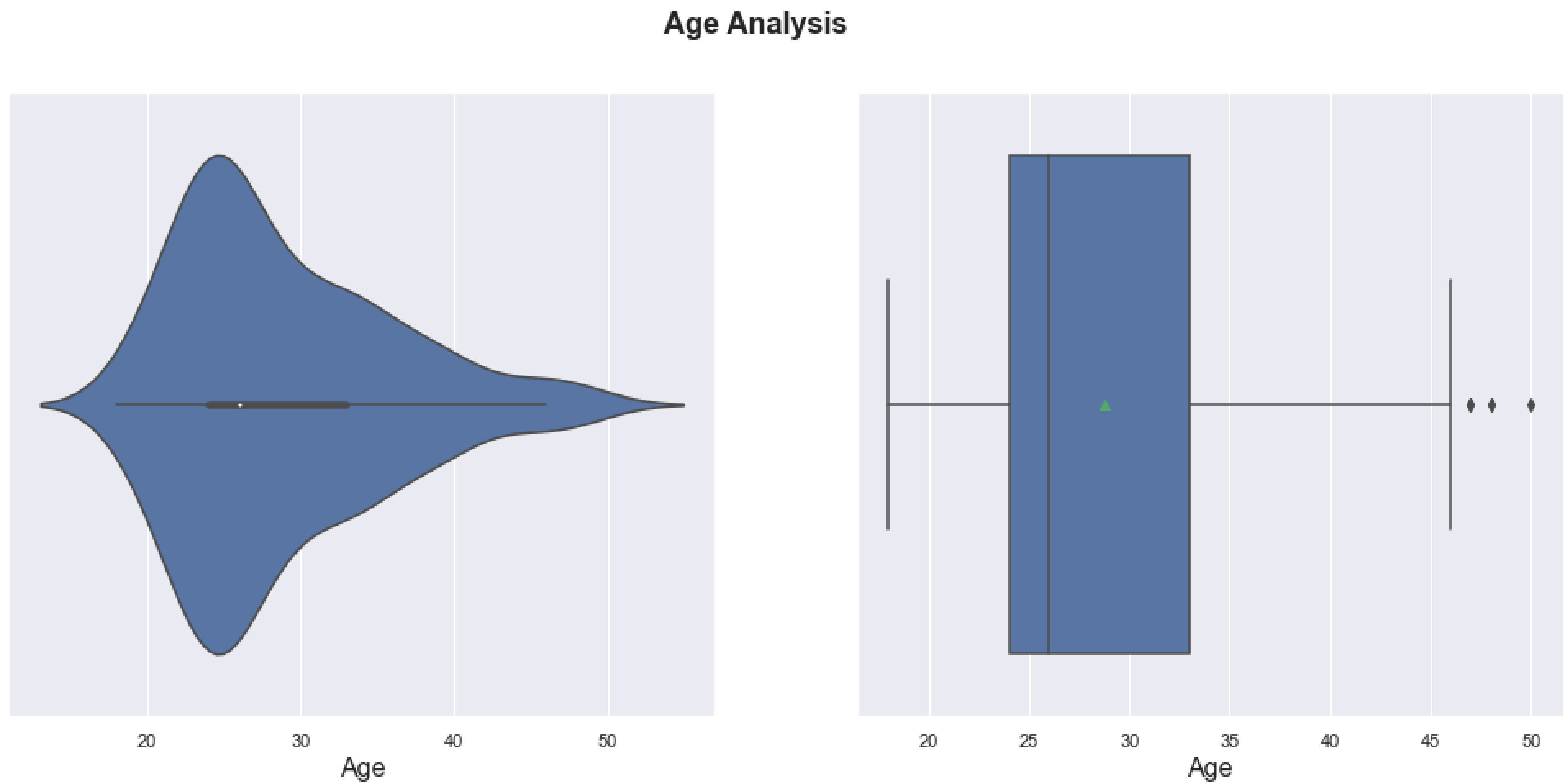
```
ax = sns.displot(data['Age'], kde = True, edgecolor='green')
plt.axvline(data['Age'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Age'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Age'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("Age Analysis", fontsize = 15)
plt.show()
```




```
Here we can see that mean for age is greater than median and mode so it is positively skewed in terms of age
```

In [496]:

```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("Age Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Age', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Age', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Age")
plt.show()
```



In [497]:
```
Q3, Q1 = np.percentile(data['Age'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Age'].mean(),2))
print("Median: ", data['Age'].median())
print("Mode: " , round(data['Age'].mode(), 2))
print("IQR: " , IQR)
print("Maximum Age Excluding Outlier: " , maxExcludingOutlier)
print("Minimum Age Excluding Outlier: " , minExcludingOutlier)
```

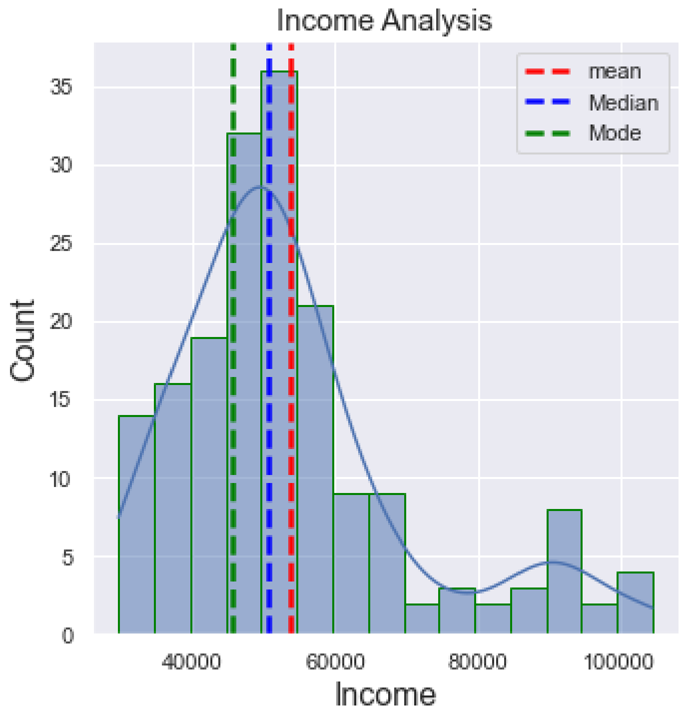
```
Q1:  24.0
Q3:  33.0
Mean:  28.79
Median:  26.0
Mode:  0    25
dtype: int64
IQR:  9.0
Maximum Age Excluding Outlier:  46.5
Minimum Age Excluding Outlier:  10.5
```

1. Here we can say that data is right skewed or positively skewed
2. Mean Age of customers is 28.79, Median age is 26 and mode age is 25
3. Most of the customers buying teadmills lies between age 24 to 33
4. Customers buying treadmill after age of 40 and before 20 are very less
5. Here we can see few outliers for Age

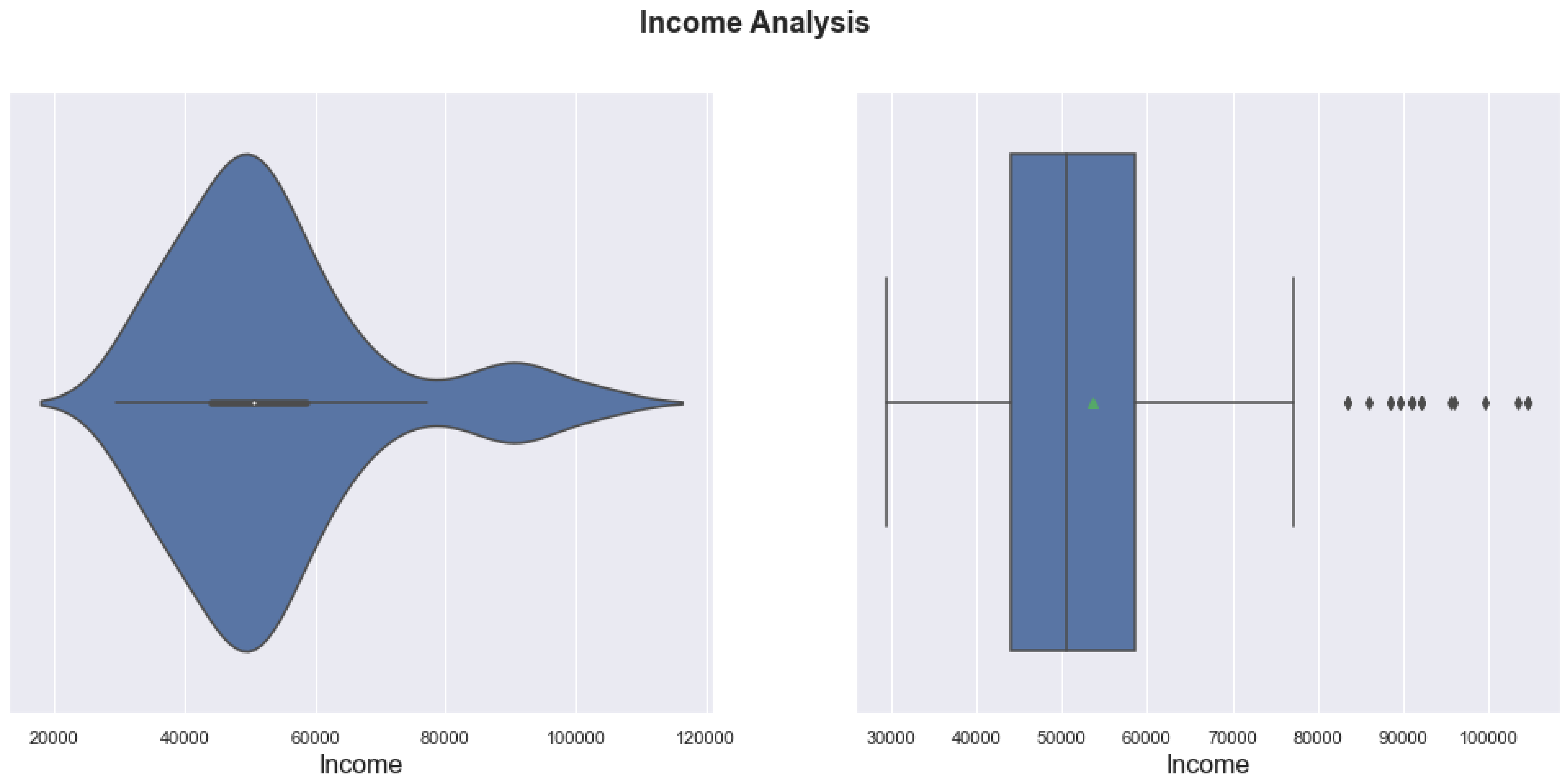
In [498]:
```
data['Income'].describe()
```

Out[498]:
```
count       180.000000
mean      53719.577778
std       16506.684226
min       29562.000000
25%       44058.750000
50%       50596.500000
75%       58668.000000
max      104581.000000
Name: Income, dtype: float64
```

In [499]:
```
ax = sns.displot(data['Income'], kde = True, edgecolor='green')
plt.axvline(data['Income'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Income'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Income'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("Income Analysis", fontsize = 15)
plt.show()
```



In [500]:
```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("Income Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Income', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Income', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Income")
plt.show()
```



In [501]:
```
Q3, Q1 = np.percentile(data['Income'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Income'].mean(),2))
print("Median: ", data['Income'].median())
print("Mode: ", round(data['Income'].mode(), 2))
print("IQR: " , IQR)
print("Maximum Income Excluding Outlier: " , maxExcludingOutlier)
print("Minimum Income Excluding Outlier: " , minExcludingOutlier)
```

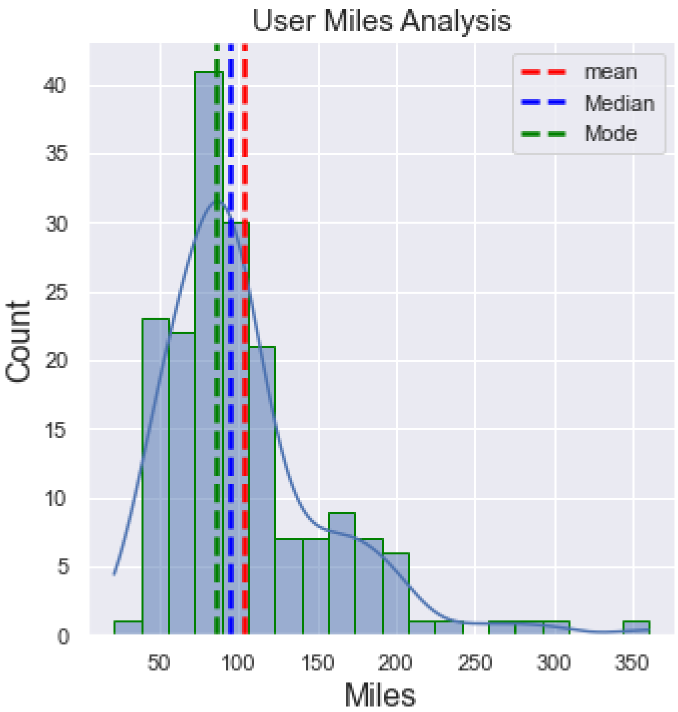
```
Q1:  44058.75
Q3:  58668.0
Mean:  53719.58
Median:  50596.5
Mode:  0    45480
dtype: int64
IQR:  14609.25
Maximum Income Excluding Outlier:  80581.875
Minimum Income Excluding Outlier:  22144.875
```

1. Here we can say that data is right skewed or positively skewed
2. Mean Income of customers is 53719 dollars , Median Income is 50596 dollars and mode Income is 45480 dollars.
3. Most of the customers Income buying teadmills lies between 44000 dollars to 658668 dollars.
4. Here we can see outliers for Income for customers whose income greater than 80000 dollars.

In [502]:
```
data['Miles'].describe()
```

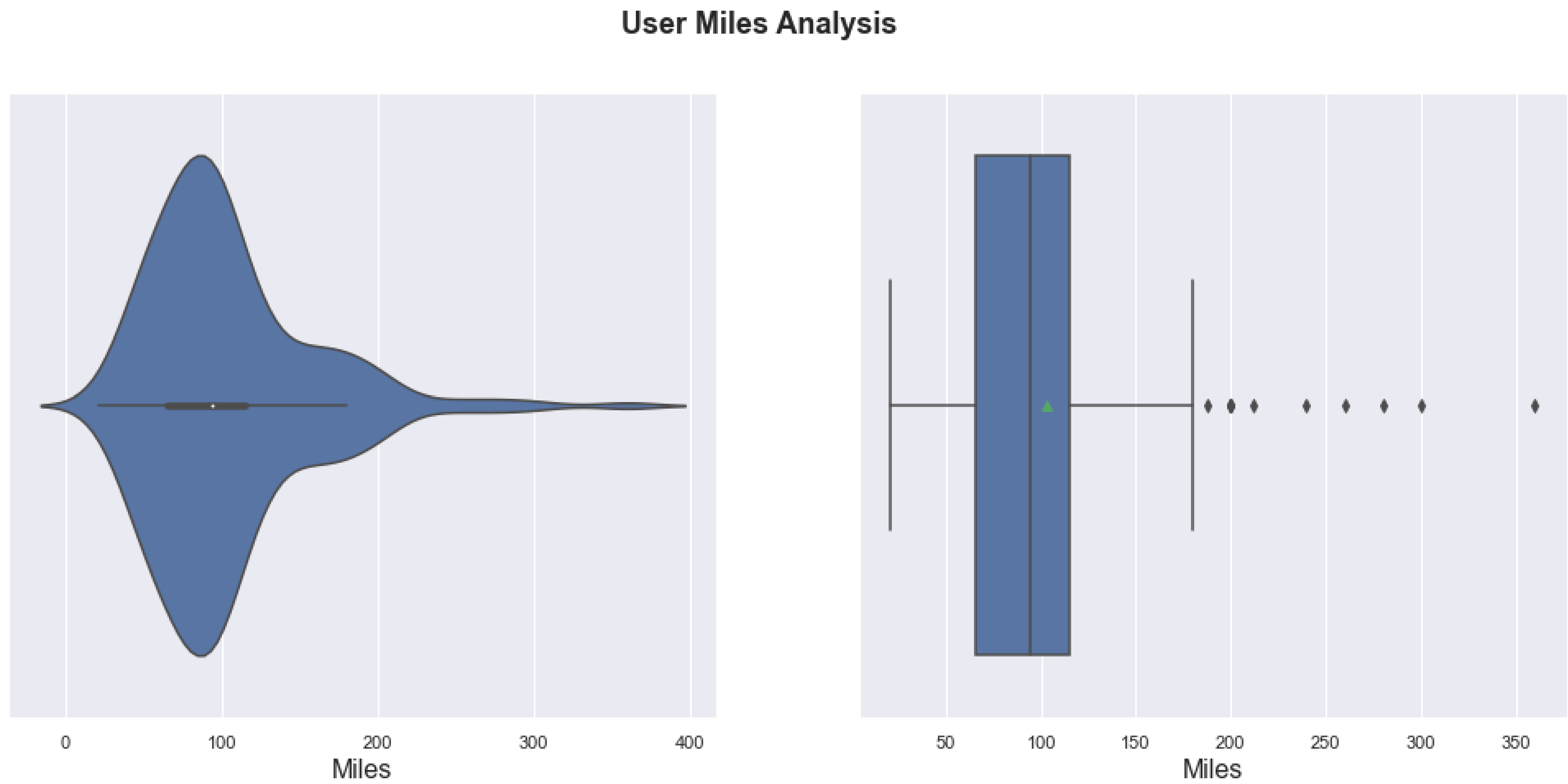
Out[502]:
```
count    180.000000
mean     103.194444
std       51.863605
min       21.000000
25%       66.000000
50%       94.000000
75%      114.750000
max      360.000000
Name: Miles, dtype: float64
```

In [503]:
```
ax = sns.displot(data['Miles'], kde = True, edgecolor='green')
plt.axvline(data['Miles'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Miles'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Miles'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("User Miles Analysis", fontsize = 15)
plt.show()
```



In [504]:

```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("User Miles Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Miles', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Miles', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Miles")
plt.show()
```



In [505]:
```
Q3, Q1 = np.percentile(data['Miles'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Miles'].mean(),2))
print("Median: ", data['Miles'].median())
print("Mode: ", round(data['Miles'].mode(), 2))
print("IQR: " , IQR)
print("Maximum Miles Excluding Outlier: " , maxExcludingOutlier)
print("Minimum Miles Excluding Outlier: " , minExcludingOutlier)
```

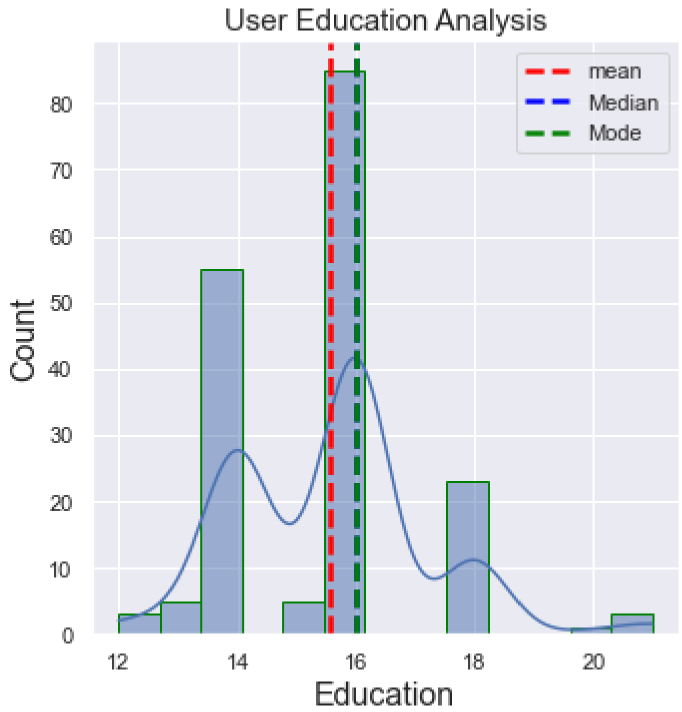
```
Q1:  66.0
Q3:  114.75
Mean:  103.19
Median:  94.0
Mode:  0    85
dtype: int64
IQR:  48.75
Maximum Miles Excluding Outlier:  187.875
Minimum Miles Excluding Outlier:  -7.125
```

1. Here we can say that data is right skewed or positively skewed
2. Mean Miles per week of customers is 103 miles per week
3. Miles having more Outliers which are greater than 188 miles per week

In [506]:
```
data['Education'].describe()
```

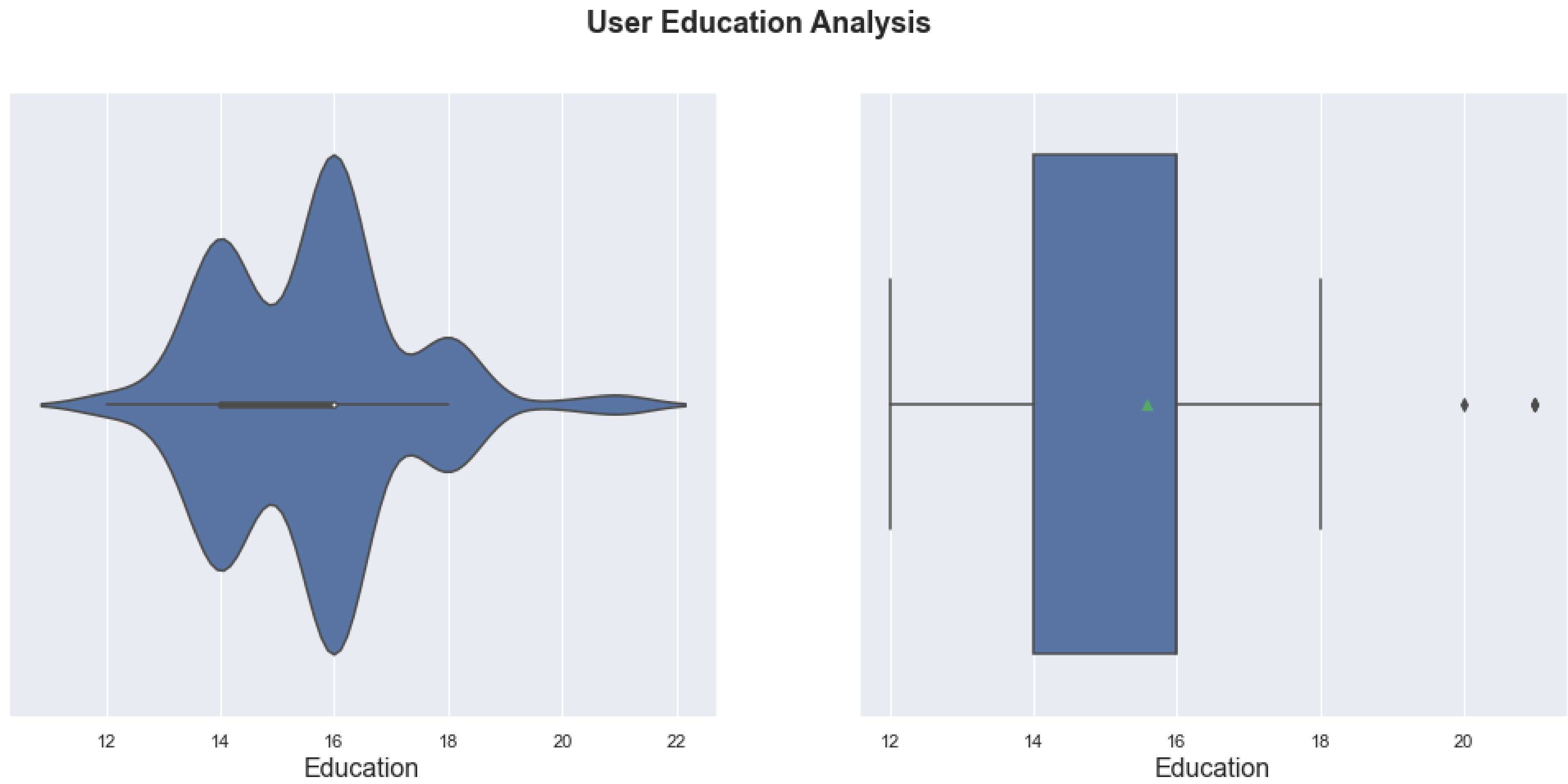
Out[506]:
```
count    180.000000
mean      15.572222
std        1.617055
min       12.000000
25%       14.000000
50%       16.000000
75%       16.000000
max       21.000000
Name: Education, dtype: float64
```

In [507]:
```
ax = sns.displot(data['Education'], kde = True, edgecolor='green')
plt.axvline(data['Education'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Education'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Education'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("User Education Analysis", fontsize = 15)
plt.show()
```



In [508]:

```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("User Education Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Education', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Education', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Education")
plt.show()
```



In [509]:
```python
Q3, Q1 = np.percentile(data['Education'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Education'].mean(),2))
print("Median: ", data['Education'].median())
print("IQR: " , IQR)
print("Maximum years of Education Excluding Outlier: " , maxExcludingOutlier)
print("Minimum years of Education Excluding Outlier: " , minExcludingOutlier)
```

```
Q1:  14.0
Q3:  16.0
Mean:  15.57
Median:  16.0
IQR:  2.0
Maximum years of Education Excluding Outlier:  19.0
Minimum years of Education Excluding Outlier:  11.0
```

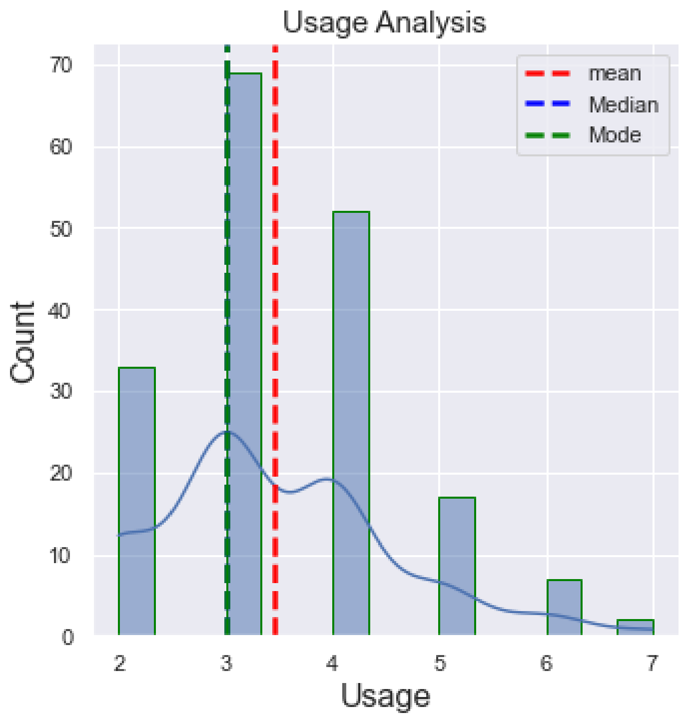
1. Here we can say that data is right skewed or positively skewed
2. Median Education of customers is 16 years
3. Education is having very few Outliers with greater than 18 years of education

In [ ]:

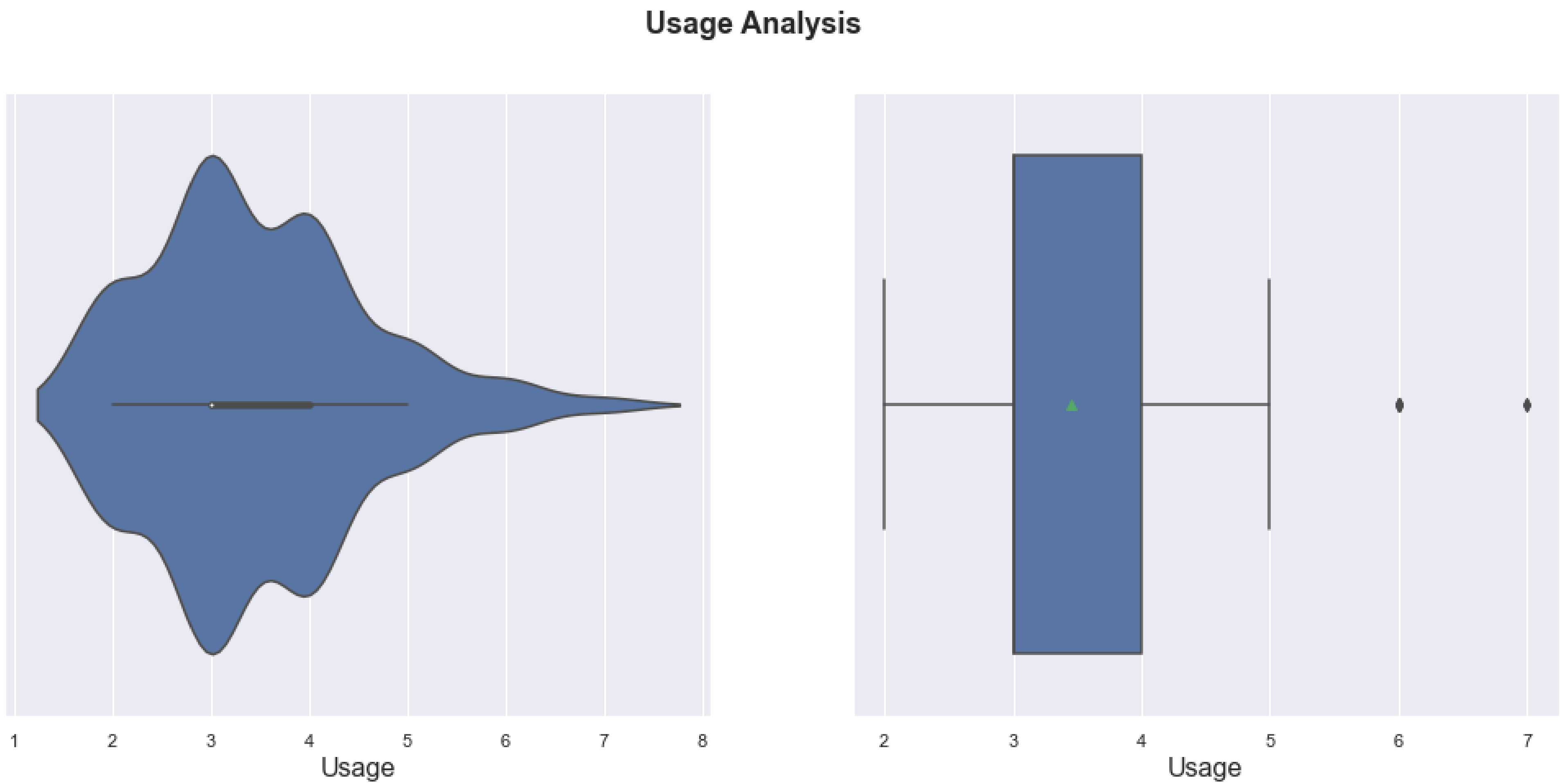
In [510]:
```python
data['Usage'].describe()
```

Out[510]:
```
count    180.000000
mean       3.455556
std        1.084797
min        2.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        7.000000
Name: Usage, dtype: float64
```

In [511]:
```
ax = sns.displot(data['Usage'], kde = True, edgecolor='green')
plt.axvline(data['Usage'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Usage'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Usage'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("Usage Analysis", fontsize = 15)
plt.show()
```



In [512]:
```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("Usage Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Usage', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Usage', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Usage")
plt.show()
```



In [513]:
```
Q3, Q1 = np.percentile(data['Usage'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Usage'].mean(),2))
print("Median: ", data['Usage'].median())
print("IQR: " , IQR)
print("Maximum Usage Excluding Outlier: " , maxExcludingOutlier)
print("Minimum Usage Excluding Outlier: " , minExcludingOutlier)
```

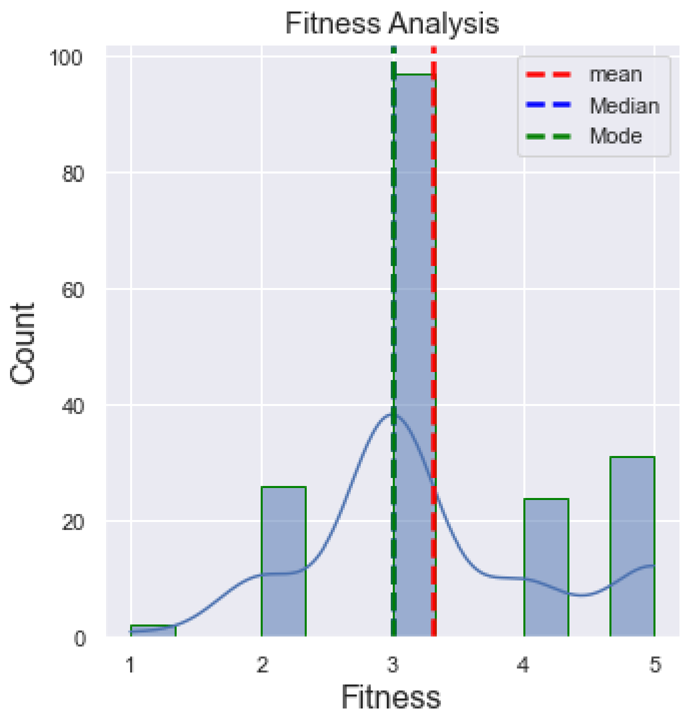
```
Q1:  3.0
Q3:  4.0
Mean:  3.46
Median:  3.0
IQR:  1.0
Maximum Usage Excluding Outlier:  5.5
Minimum Usage Excluding Outlier:  1.5
```

1. Most of customers expect they will be using the treadmill 3-4 days per week.
2. There are few outliers where customer are usage of treadmill for 6 or 7 times per week

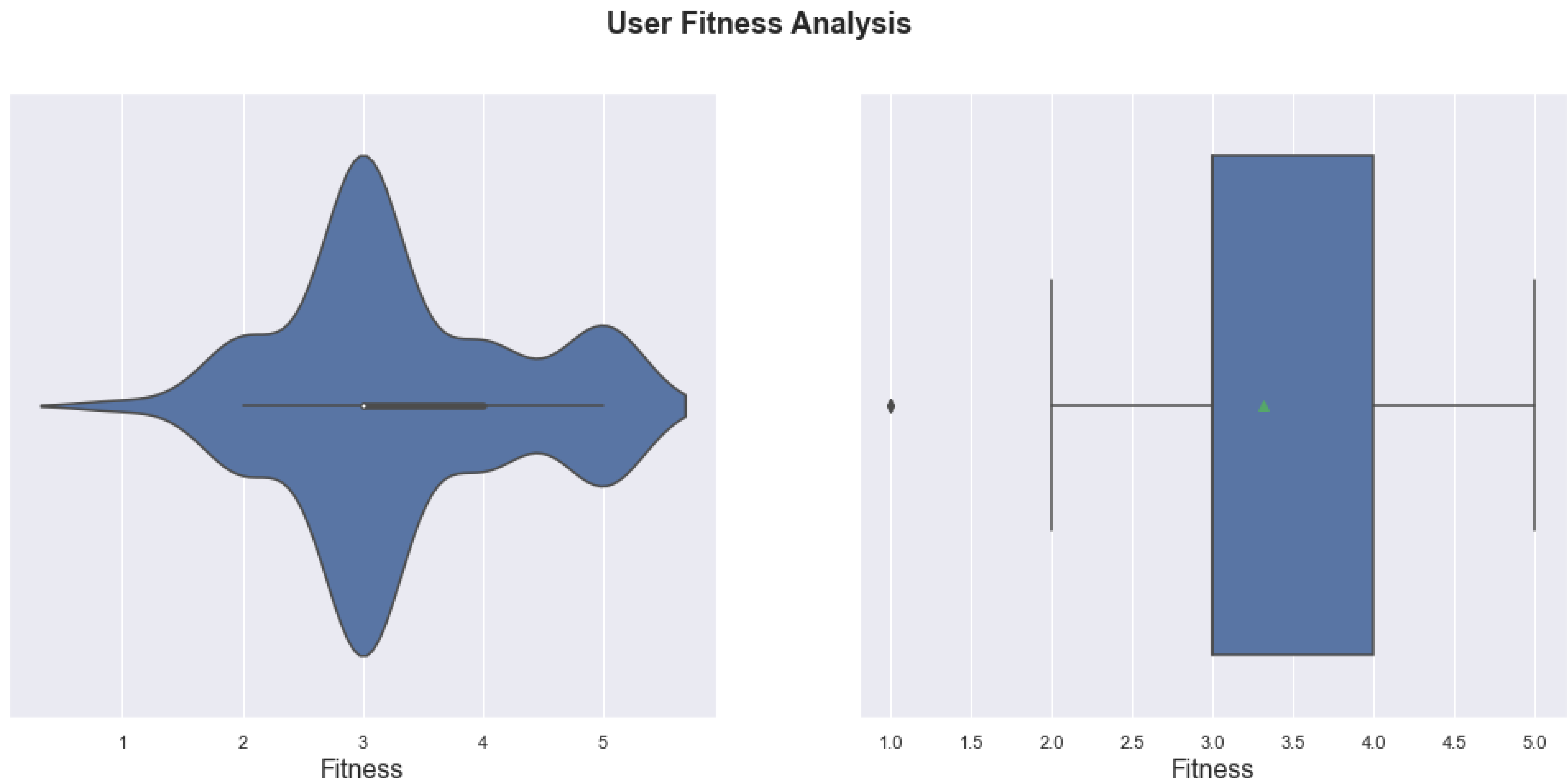
In [514]:
```
data['Fitness'].describe()
```

Out[514]:
```
count    180.000000
mean       3.311111
std        0.958869
min        1.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        5.000000
Name: Fitness, dtype: float64
```

In [515]:
```
ax = sns.displot(data['Fitness'], kde = True, edgecolor='green')
plt.axvline(data['Fitness'].mean(), ls = '--', color = "red", lw = 2.5, label = "mean")
plt.axvline(data['Fitness'].median(), ls = '--' ,color = 'blue', lw = 2.5, label = 'Median')
plt.axvline(data['Fitness'].mode()[0], ls = '--', color = 'green', lw = 2.5, label = 'Mode')
plt.legend()
plt.title("Fitness Analysis", fontsize = 15)
plt.show()
```



In [516]:
```
fig, axes =plt.subplots(1,2,figsize=(17, 7))
fig.suptitle("User Fitness Analysis "  , fontsize=18, fontweight='bold')
sns.violinplot(x = 'Fitness', data= data, showmeans=True , ax = axes[0])
sns.boxplot(x = 'Fitness', data= data, showmeans=True, ax = axes[1])
plt.xlabel("Fitness")
plt.show()
```



In [517]:
```
Q3, Q1 = np.percentile(data['Fitness'], [75 ,25])
IQR = Q3 - Q1
maxExcludingOutlier = Q3 + 1.5 * IQR
minExcludingOutlier = Q1 - 1.5 * IQR

print("Q1: ", Q1)
print("Q3: ", Q3)
print("Mean: ", round(data['Fitness'].mean(),2))
print("Median: ", data['Fitness'].median())
print("Mode: ", round(data['Fitness'].mode(),2))
print("IQR: " , IQR)
print("Maximum User Fitness Excluding Outlier: " , maxExcludingOutlier)
print("Minimum User Fitness Excluding Outlier: " , minExcludingOutlier)
```

```
Q1:  3.0
Q3:  4.0
Mean:  3.31
Median:  3.0
Mode:  0    3
dtype: int64
IQR:  1.0
Maximum User Fitness Excluding Outlier:  5.5
Minimum User Fitness Excluding Outlier:  1.5
```

Most of the customers have self-rated their fitness as 3 i.e., Moderate

In [ ]:

In [518]:
```
# Sales Analysis
ProductCounts = pd.DataFrame(data.groupby('Product').size())
ProductCounts
ProductCounts.reset_index(inplace = True)
ProductCounts.columns = ['Product', 'Quantity']
ProductCounts
```

Out[518]:

	Product	Quantity
0	KP281	80
1	KP481	60
2	KP781	40

In [519]:
```python
prices = [1500, 1750, 2500]
ProductCounts['TotalPrice'] = ProductCounts['Quantity'] * prices
ProductCounts
```

Out[519]:

	Product	Quantity	TotalPrice
0	KP281	80	120000
1	KP481	60	105000
2	KP781	40	100000

In [520]:
```python
totalSales = ProductCounts['TotalPrice'].sum()
totalSales
```

Out[520]:
```
325000
```

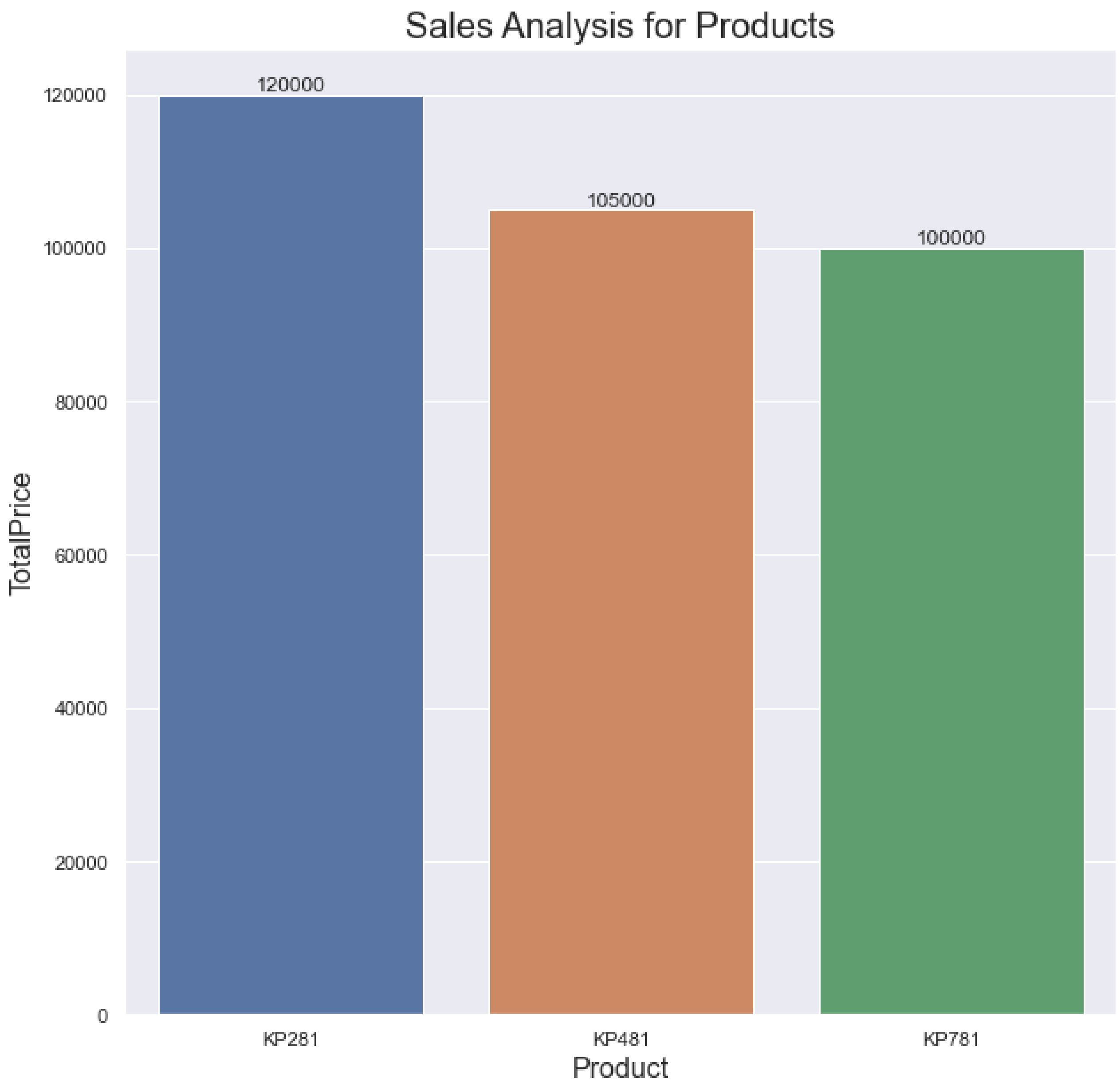
In [521]:
```python
ProductCounts['Percentage'] = ProductCounts['TotalPrice'] * 100 / totalSales
```

In [522]:
```python
ProductCounts
```

Out[522]:

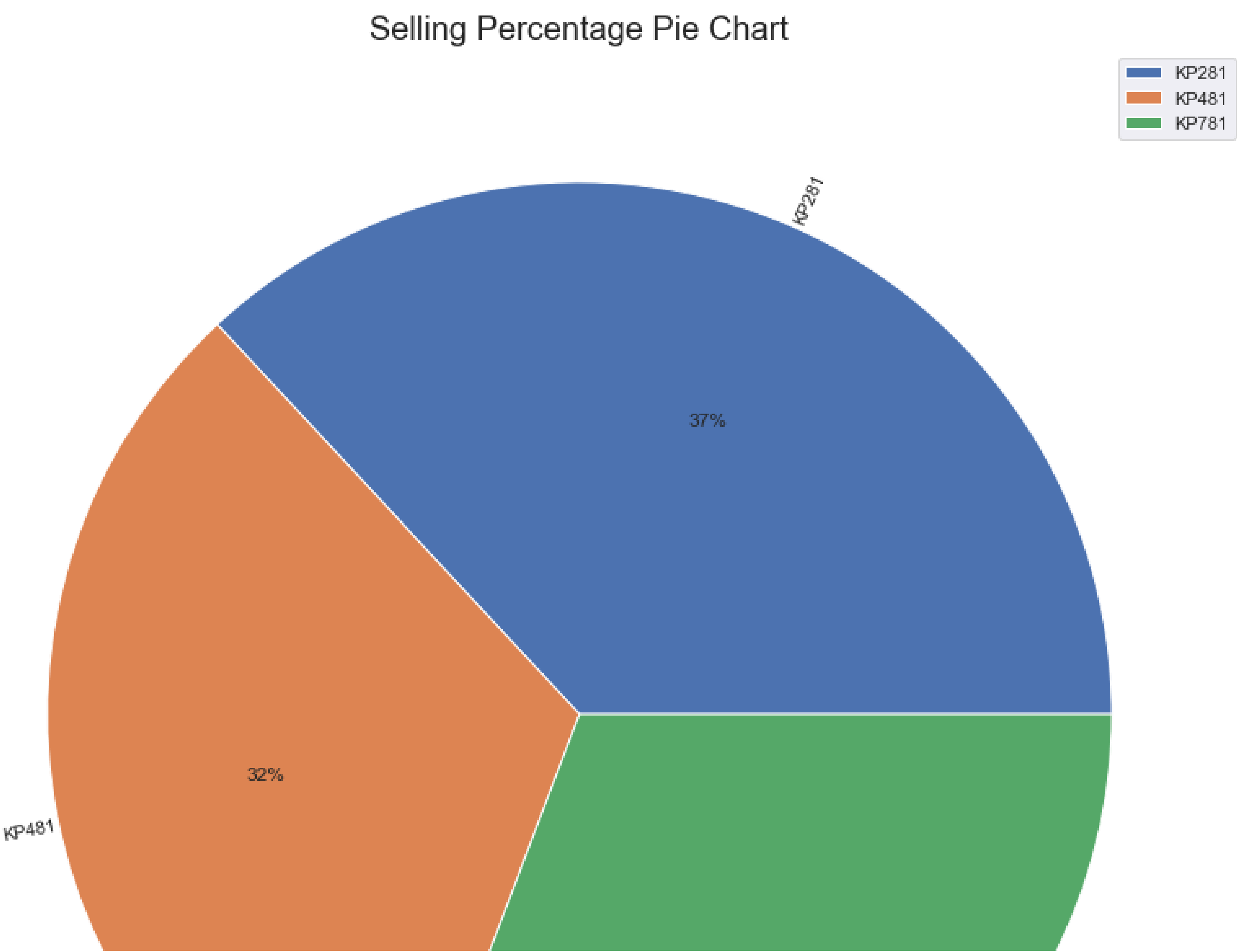
	Product	Quantity	TotalPrice	Percentage
0	KP281	80	120000	36.923077
1	KP481	60	105000	32.307692
2	KP781	40	100000	30.769231

In [523]:
```
plt.figure(figsize=(10,10))
graph = sns.barplot(x = 'Product', y = 'TotalPrice',data = ProductCounts)
for i in graph.containers:
    graph.bar_label(i,)
plt.title("Sales Analysis for Products")
plt.show()
```



In [524]:
```
plt.figure(figsize=(5,5))
values = list(ProductCounts['TotalPrice'])
labels = list(ProductCounts['Product'])
plt.figure(figsize=(15,15))
plt.pie(values ,labels = labels, autopct='%.0f%%', labeldistance=1 , rotatelabels = 270) # To show the portions in %ages
plt.title("Selling Percentage Pie Chart" , fontsize = 20)
plt.legend(labels, loc = 'best' )
plt.show()
```

```
<Figure size 360x360 with 0 Axes>
```



From the above sales analysis
Out of the total sales, KP281 Product is responsible for around 37% Income from buying, KP481 32% and KP781 31%

In [ ]:

Bivariate Analysis

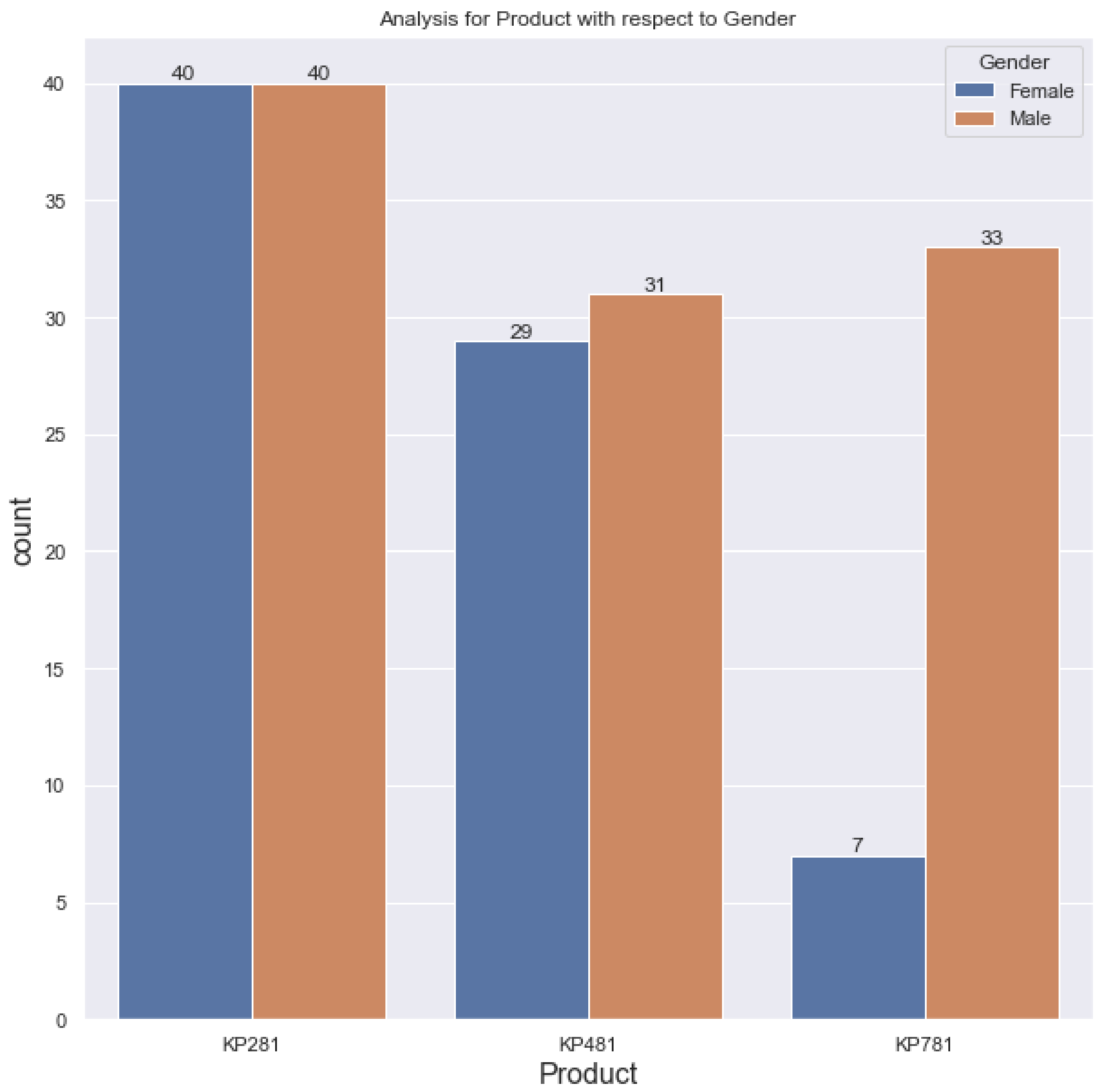
In [525]:
```
# Bivariate Analysis for gender and product
```

In [526]:
```
data.groupby([data['Product'], data['Gender']]).size()
```

Out[526]:
```
Product  Gender
KP281    Female    40
         Male      40
KP481    Female    29
         Male      31
KP781    Female     7
         Male      33
dtype: int64
```

In [527]:

```
plt.figure(figsize=(10, 10))
graph = sns.countplot(x = 'Product', data= data, hue = 'Gender')
for i in graph.containers:
    graph.bar_label(i,)
plt.title("Analysis for Product with respect to Gender", fontsize = 12)
plt.show()
```



KP281 -> Male and female bought equal number of products

KP481 -> Slightly less number of females bought KP481 Treadmill as compared to male customers

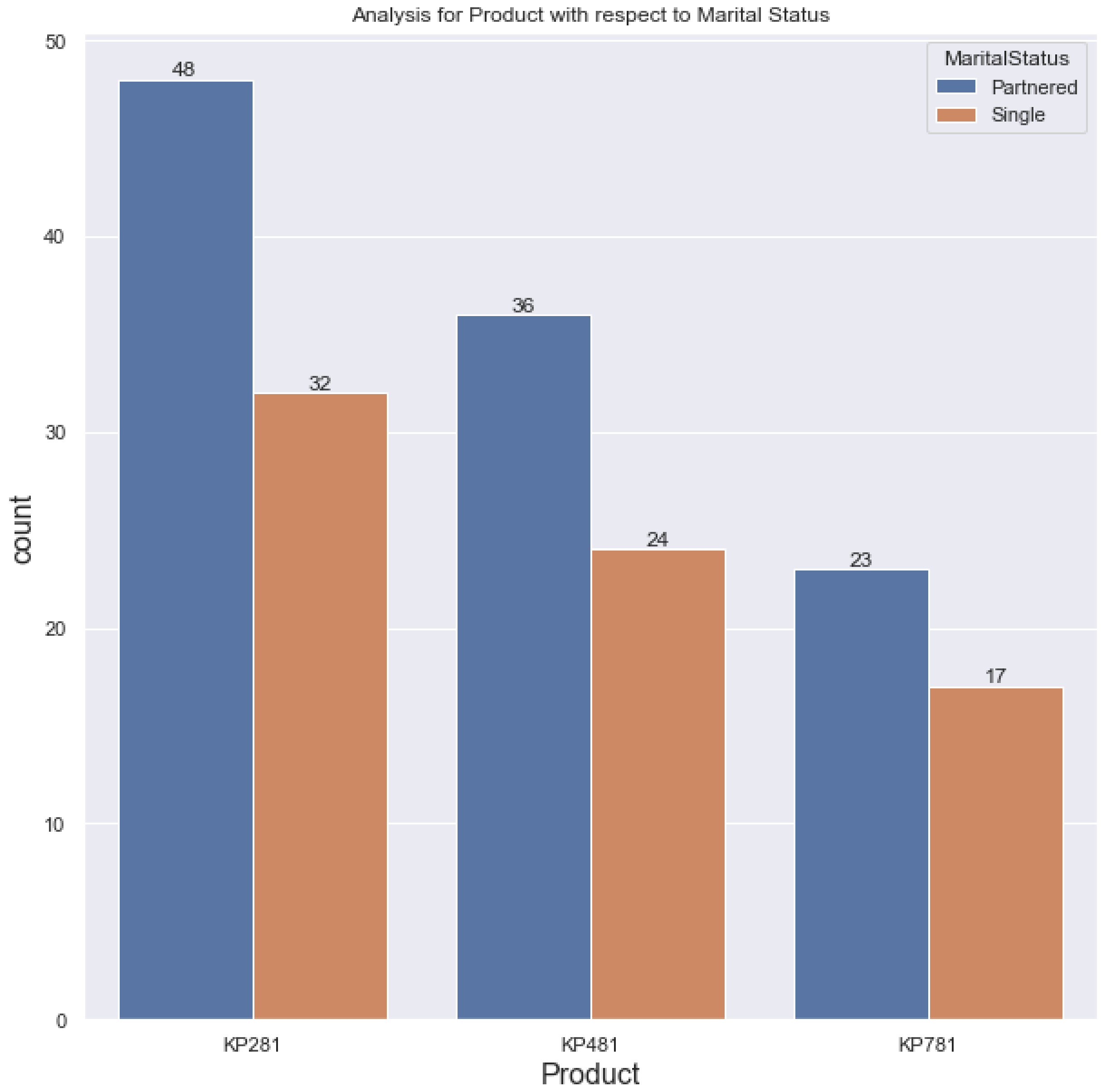
KP781 -> There is huge difference between Male and female customers who bought KP781 Treadmill

only 7 females bought KP781 while 33 males customers bought the same

In [528]:
```
data.groupby([data['Product'], data['MaritalStatus']]).size()
```

Out[528]:
```
Product  MaritalStatus
KP281    Partnered        48
         Single           32
KP481    Partnered        36
         Single           24
KP781    Partnered        23
         Single           17
dtype: int64
```

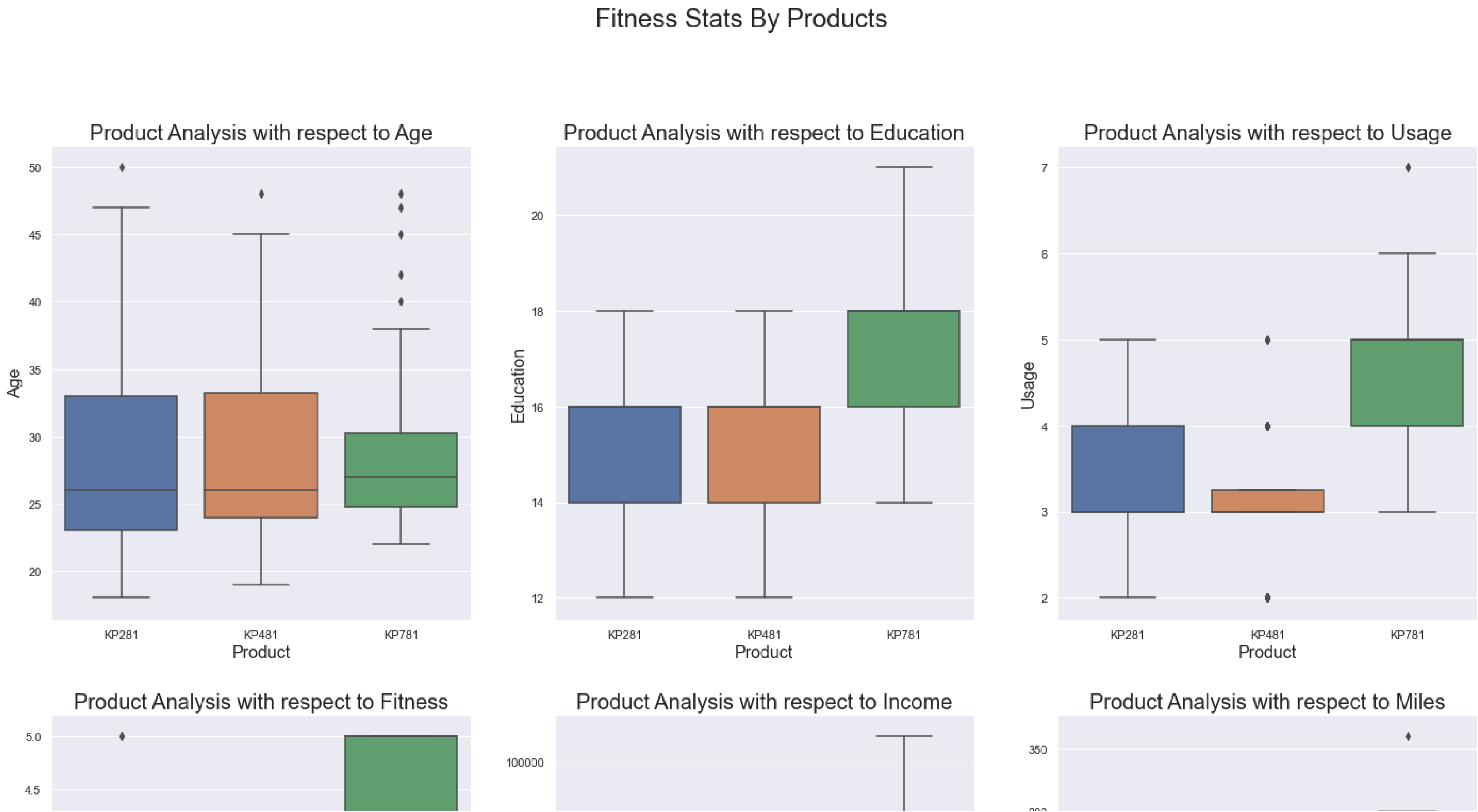
In [529]:
```
plt.figure(figsize=(10, 10))
graph = sns.countplot(x = 'Product', data= data, hue = 'MaritalStatus')
for i in graph.containers:
    graph.bar_label(i,)
plt.title("Analysis for Product with respect to Marital Status", fontsize = 12)
plt.show()
```

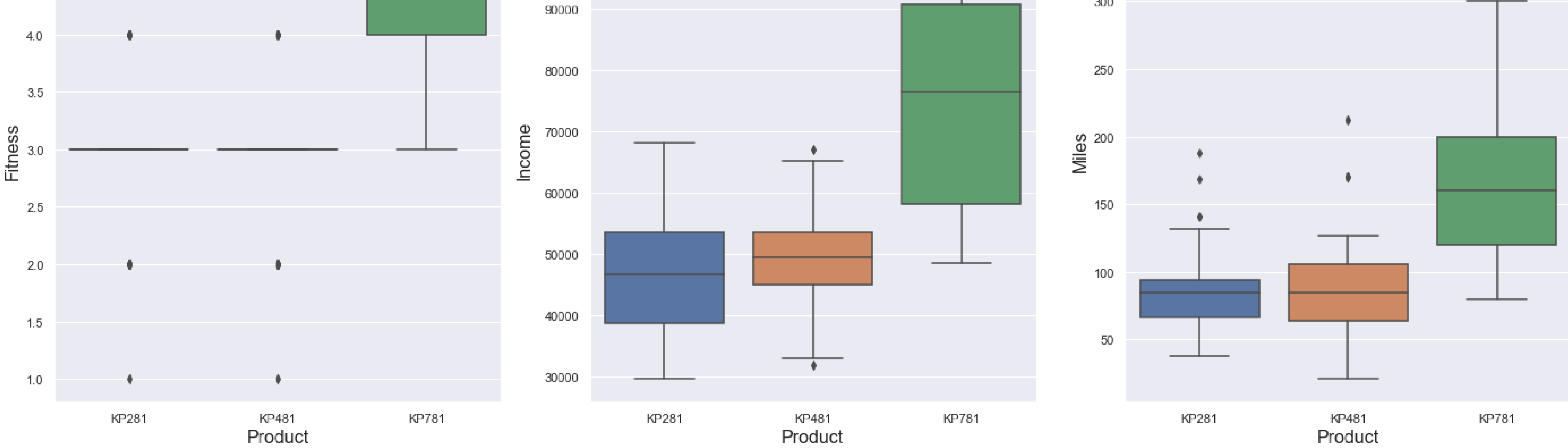


From the above chart we can say that customers who are married are more likely to bought the product

In [530]:
```
fig, axes = plt.subplots(2, 3, figsize=(24, 18), sharey=False)
sns.set( rc = {'figure.figsize' : ( 20, 20 ), 'axes.labelsize' : 20 , 'axes.titlesize' : 20})
fig.suptitle("Fitness Stats By Products", fontsize=24)
sns.boxplot(ax=axes[0, 0], data=data, x='Product', y='Age', saturation=0.75,width=0.8,dodge=False ).set(title='Product Analysis wi
sns.boxplot(ax=axes[0, 1], data=data, x='Product', y='Education',  saturation=0.75,width=0.8,dodge=False).set(title='Product Analy
sns.boxplot(ax=axes[0, 2], data=data, x='Product', y='Usage',  saturation=0.75,width=0.8,dodge=False).set(title='Product Analysis 
sns.boxplot(ax=axes[1, 0], data=data, x='Product', y='Fitness',  saturation=0.75,width=0.8,dodge=False).set(title='Product Analysi
sns.boxplot(ax=axes[1, 1], data=data, x='Product', y='Income',  saturation=0.75,width=0.8,dodge=False).set(title='Product Analysis
sns.boxplot(ax=axes[1, 2], data=data, x='Product', y='Miles',  saturation=0.75,width=0.8,dodge=False).set(title='Product Analysis 


plt.show()
```





1. Product Analysis with respect to Age:
   - Customers who are purchasing KP281 and KP481 have same median age value which is 26
   - Customers from age range 25 to 30 most likely to purchase treadmill KP781
2. Product Analysis with respect to Education:
   - For Treadmills having 14 to 16 years of education are more likely to purchase KP281 or KP481 treadmills with equal chances
   - Customers who are having 16 plus years of education are more likely to buy KP781 treadmill
3. Product Analysis with respct to Usage:
   - Customers who wants to use treadmill 3 to 4 days a week having more chances to buy KP281 or KP481 treadmill
   - While customers who tends to use teadmill more than 4 times a week having chances to purchase KP781 treadmill
4. Product Analysis with respect to fitness:
   - For people with moderate fitness they are having high cahnce to buy KP281 or KP481
   - But customers who are more fit whose fitness level is greater than 3 are more likely to purchase KP781 treadmill
5. Product Analysis with respect to Income:
   - For people with income greater than 58 to 59 K dollars are more likely to buy KP781 teradmill while others having more chances to go for KP281 or KP481 treadmill
6. Product Analysis with respct to Miles:
   - If customer expects to walk/ run more than 120 miles a week they he more likely purchases KP781 Treadmill.

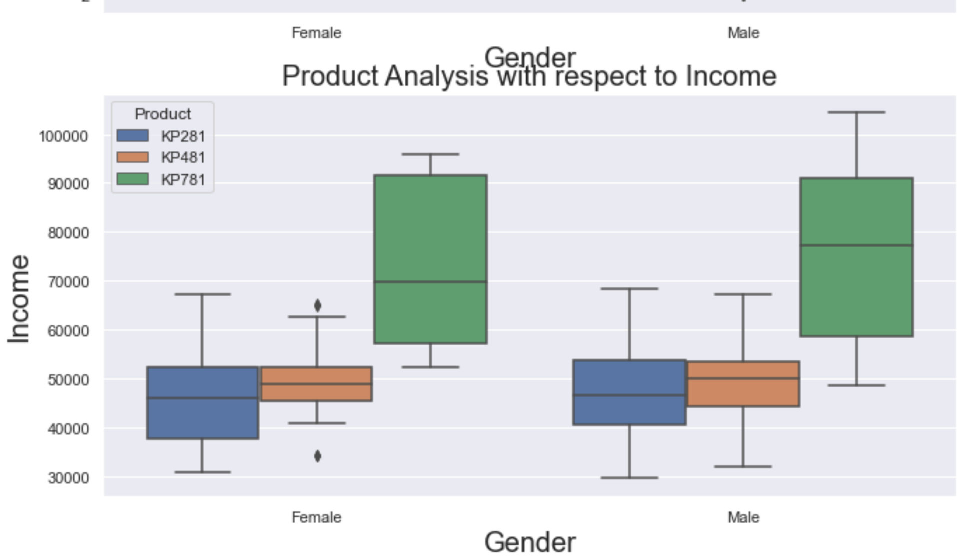
```
In [ ]:
```

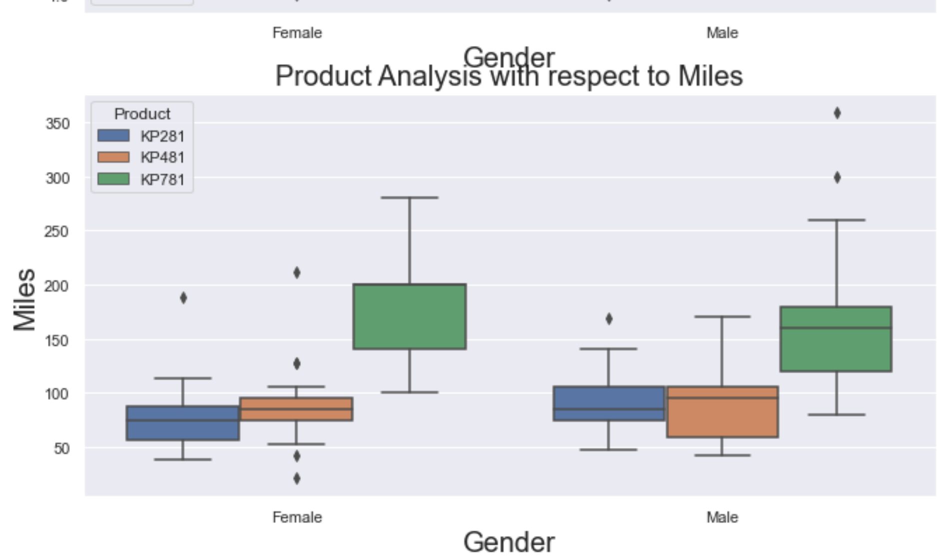
Maultivariate Analysis

In [531]:

```
fig, axes = plt.subplots(3, 2, figsize=(24, 18), sharey=False)
sns.set( rc = {'figure.figsize' : ( 20, 20 ), 'axes.labelsize' : 16 , 'axes.titlesize' : 20})
fig.suptitle("Multivariate Analysis for Products wrt Gender", fontsize=24)
sns.boxplot(ax=axes[0, 0], data=data, x='Gender', y='Age', hue = 'Product').set(title='Product Analysis with respect to Age')
sns.boxplot(ax=axes[0, 1], data=data, x='Gender', y='Education', hue = 'Product').set(title='Product Analysis with respect to Educ
sns.boxplot(ax=axes[1, 0], data=data, x='Gender', y='Usage', hue = 'Product').set(title='Product Analysis with respect to Usage')
sns.boxplot(ax=axes[1, 1], data=data, x='Gender', y='Fitness', hue = 'Product').set(title='Product Analysis with respect to Fitnes
sns.boxplot(ax=axes[2, 0], data=data, x='Gender', y='Income', hue = 'Product').set(title='Product Analysis with respect to Income'
sns.boxplot(ax=axes[2, 1], data=data, x='Gender', y='Miles', hue = 'Product').set(title='Product Analysis with respect to Miles')


plt.show()
```



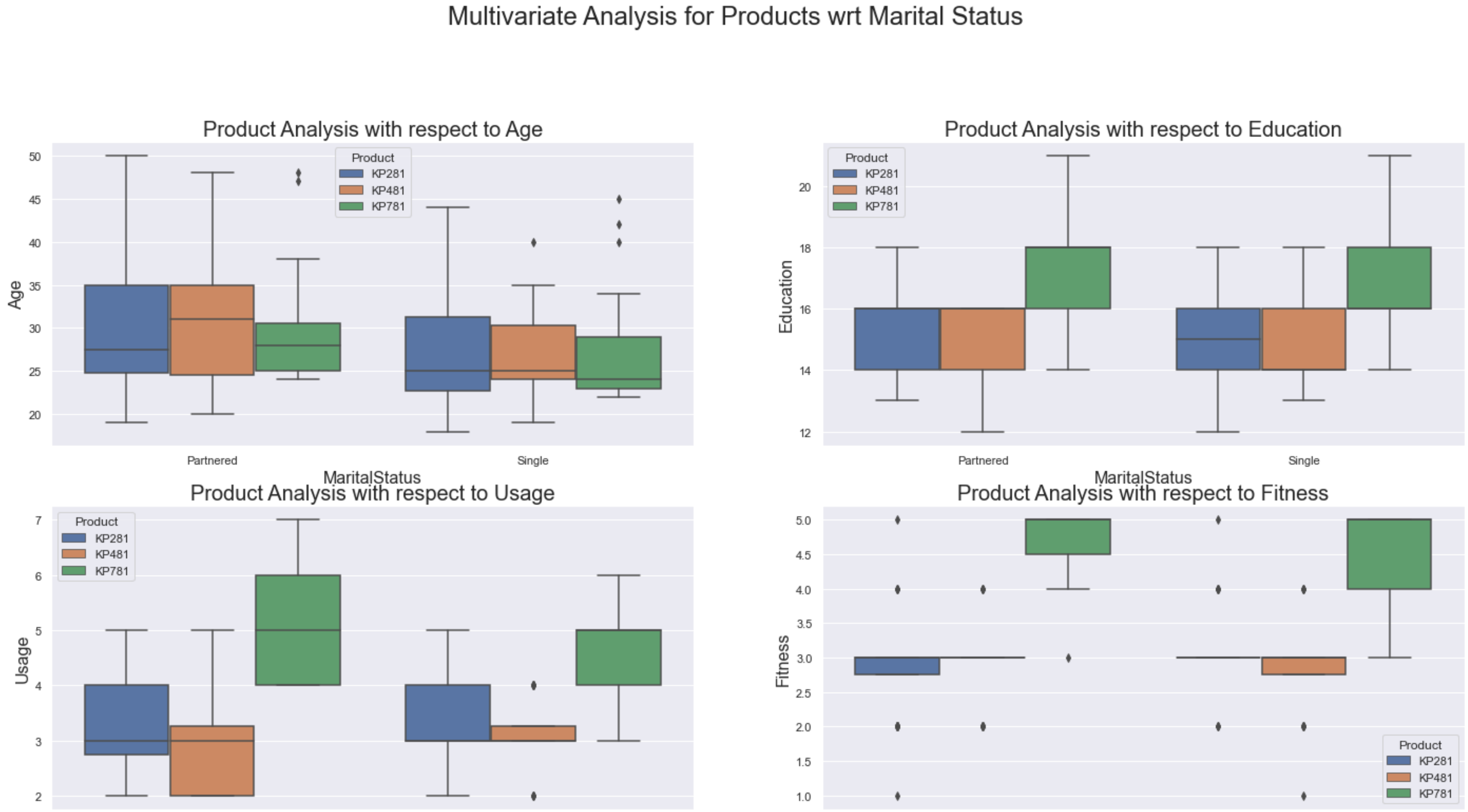


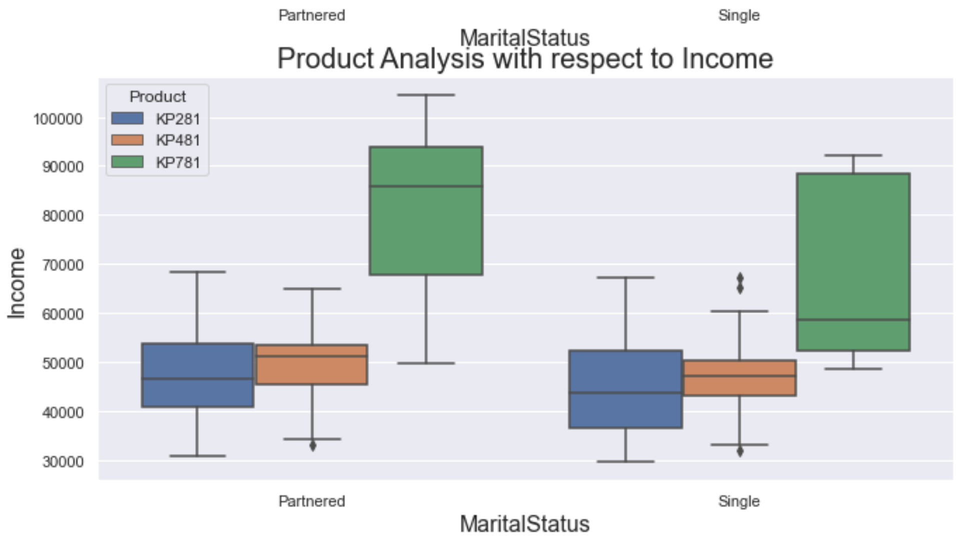

```
1. Females from Age range 25 to 28 are more likely to buy KP781 treadmill
2. Males from age range 25 to 31 have haigh chances of buying KP781 teardmill
3. Females with 14 to 16 years of education have equal chances of buying KP281 and KP481 treadmills
4. Females with 17 to 18 years of education are more likely to buy KP781 treadmill
5. males with 14 to 16 years of education have equal chances of buying KP281 and KP481 treadmills
6. males with 16 to 18 years of education are more likely to buy KP781 treadmill
7. Females who tends to use treadmill 2 to 3 days per week have higher chances to go for KP281 treadmill
8. Females who tends to use treadmill 3 to 4 days per week have higher chances to go for KP481 treadmill
9. Females who tends to use treadmill 4 to 6 days per week have higher chances to go for KP781 treadmill
10. Females who tends to use treadmill 3 to 4 days per week have higher chances to go for KP281 treadmill
11. Females who tends to use treadmill 4 to 5 days per week have higher chances to go for KP781 treadmill
12. Females and males with fitness level 4 to 5 have high chnaces of buying KP781 treadmill
13. Females and males with income more than 5500 Dollar have high chances to buy KP781 treadmill
14. Customers with more than 120 miles tends to buy KP781 treadmill
```

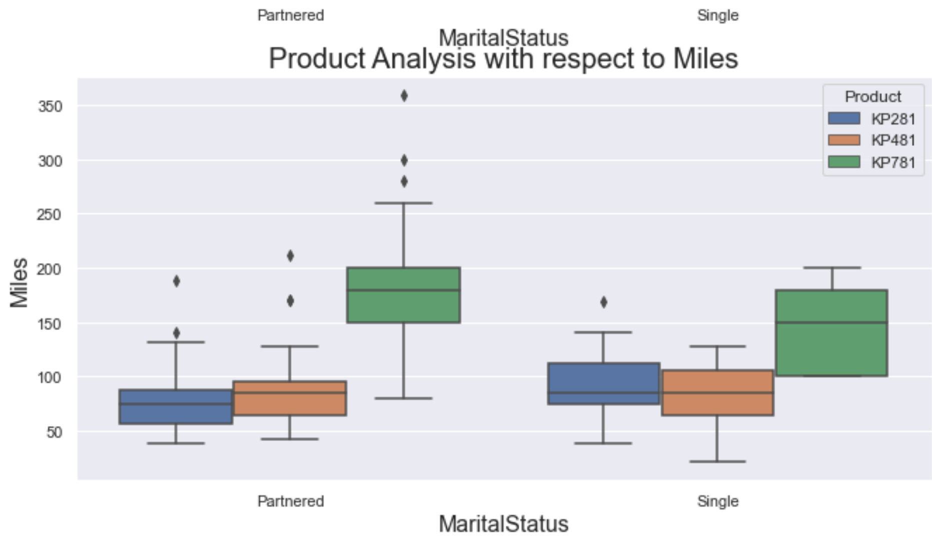
In [532]:

```
fig, axes = plt.subplots(3, 2, figsize=(24, 18), sharey=False)
sns.set( rc = {'figure.figsize' : ( 20, 20 ), 'axes.labelsize' : 16 , 'axes.titlesize' : 20})
fig.suptitle("Multivariate Analysis for Products wrt Marital Status", fontsize=24)
sns.boxplot(ax=axes[0, 0], data=data, x='MaritalStatus', y='Age', hue = 'Product').set(title='Product Analysis with respect to Age
sns.boxplot(ax=axes[0, 1], data=data, x='MaritalStatus', y='Education', hue = 'Product').set(title='Product Analysis with respect 
sns.boxplot(ax=axes[1, 0], data=data, x='MaritalStatus', y='Usage', hue = 'Product').set(title='Product Analysis with respect to U
sns.boxplot(ax=axes[1, 1], data=data, x='MaritalStatus', y='Fitness', hue = 'Product').set(title='Product Analysis with respect to
sns.boxplot(ax=axes[2, 0], data=data, x='MaritalStatus', y='Income', hue = 'Product').set(title='Product Analysis with respect to 
sns.boxplot(ax=axes[2, 1], data=data, x='MaritalStatus', y='Miles', hue = 'Product').set(title='Product Analysis with respect to M


plt.show()
```

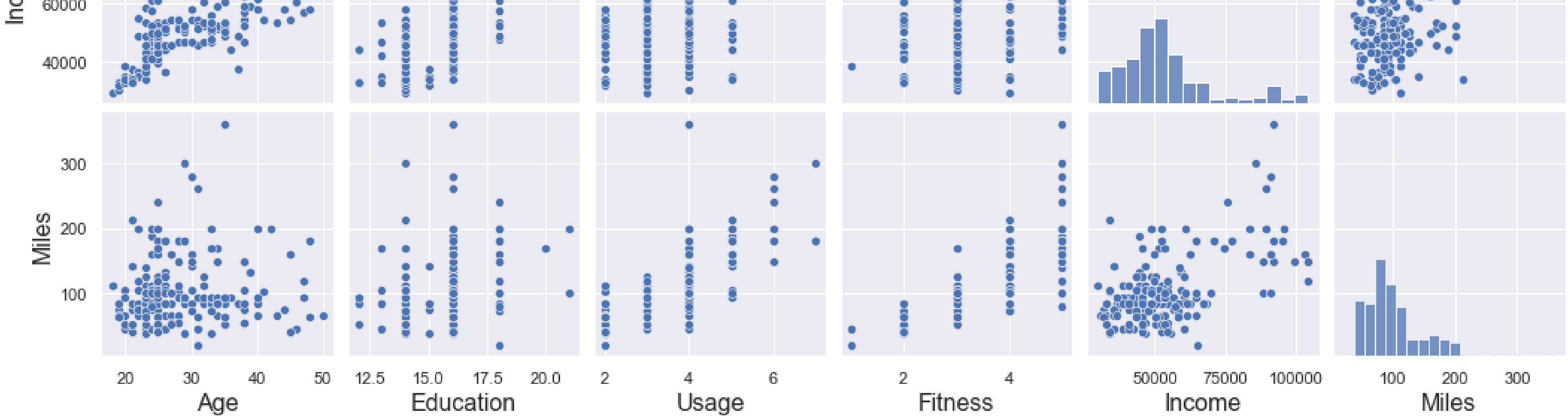



Partnered
Single
MaritalStatus
Product Analysis with respect to Income
Product
KP281
KP481
KP781
Income
100000
90000
80000
70000
60000
50000
40000
30000
Partnered
Single
MaritalStatus


Partnered
Single
MaritalStatus
Product Analysis with respect to Miles
Product
KP281
KP481
KP781
Miles
350
300
250
200
150
100
50
Partnered
Single
MaritalStatus

Pairplot

In [533]:
```
sns.pairplot(data)
plt.show()
```



Heatmap

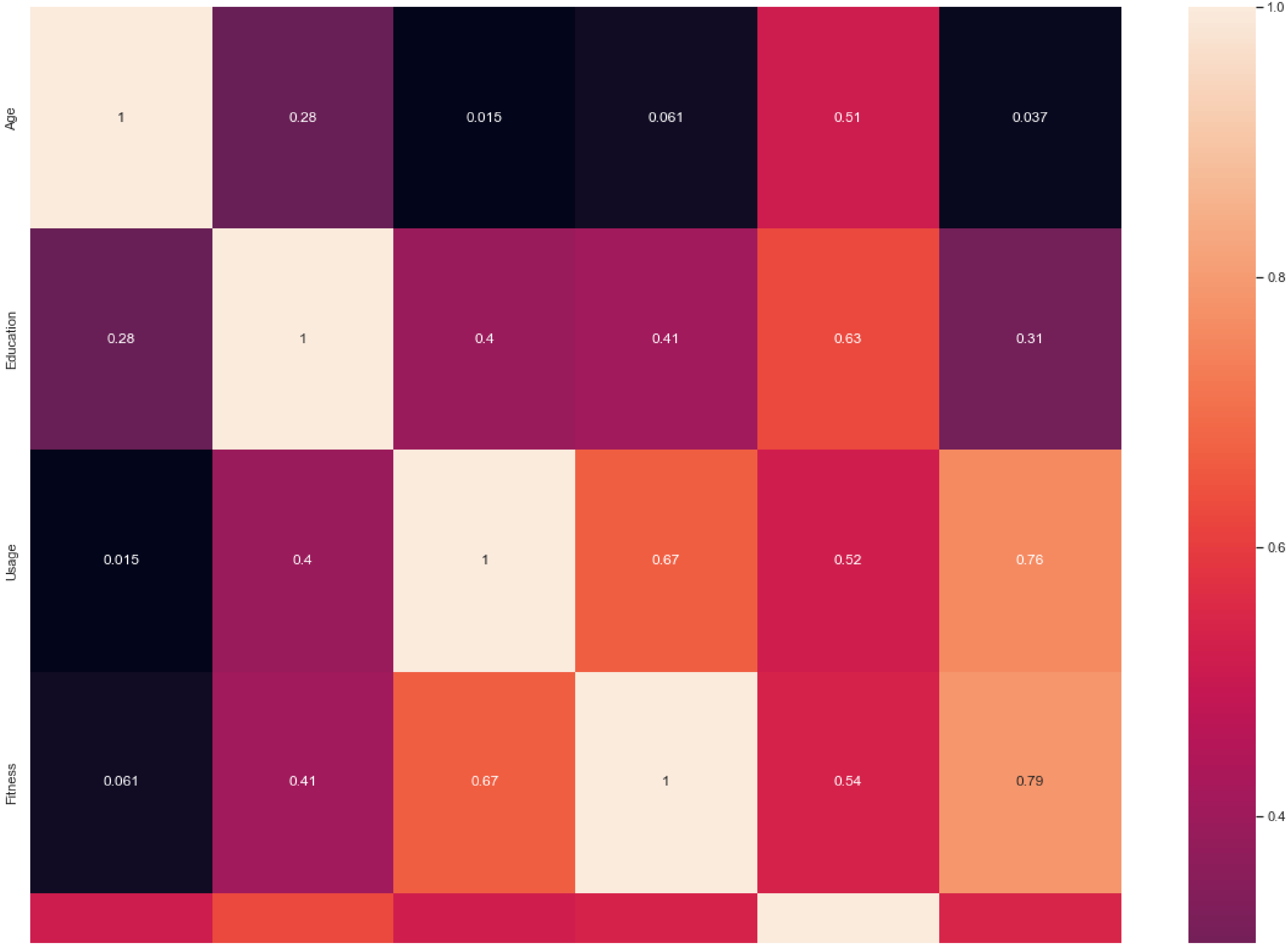
In [534]:
```
corr = data.corr()
corr
```

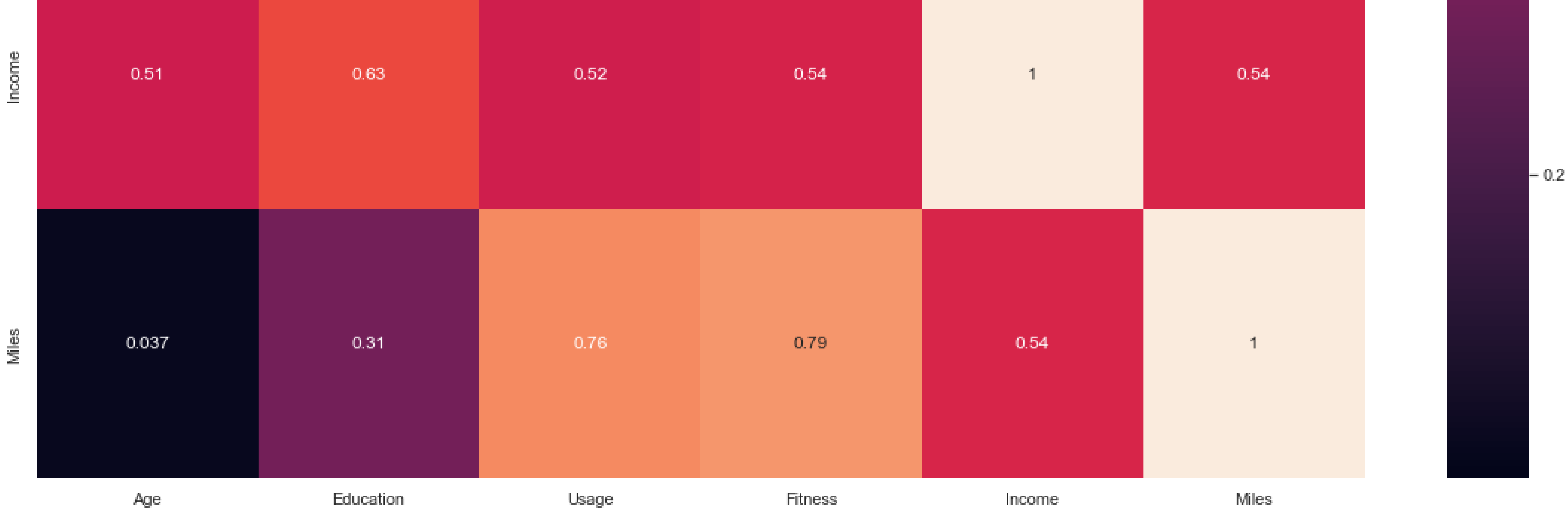
Out[534]:

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

```
Age and Income highly correlated
(Education and Fitness) and (Education and Income) highly correlated
Usage is highly correlated with Miles, Fitness and income
Fitness is highly correlated with Miles, usage and Income
Income is Highly correlated with all the factors Education, Age, Usage, Fitness and miles
Miles are highly correlated with Usage and fitness
```

In [535]:
```
sns.heatmap(corr, annot= True)
plt.show()
```





In [ ]:

Marginal Probabilities for all type of Products

In [536]:

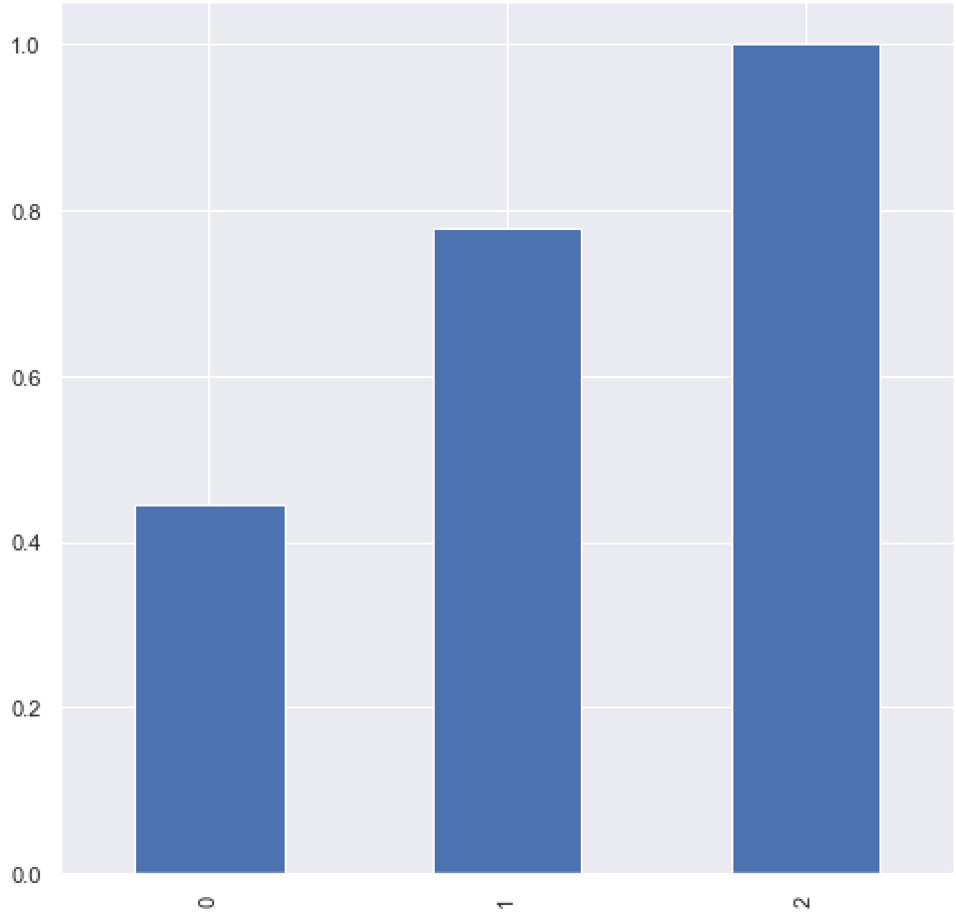
```
marginal = pd.DataFrame(data.groupby('Product').size().div(len(data)))
marginal.reset_index(inplace= True)
marginal.columns = ['Product' , 'Marginal Prob']
marginal
```

Out[536]:

	Product	Marginal Prob
0	KP281	0.444444
1	KP481	0.333333
2	KP781	0.222222

In [537]:

```
plt.figure(figsize=(8,8))
pdf = pd.Series((marginal["Marginal Prob"]))
cdf = pdf.cumsum()
cdf.plot(kind='bar')
plt.show()
```




```
Marginal Probabilities for Different Products:
    * KP281 -> 0.44
    * KP481 -> 0.33
    * KP781 -> 0.22
```

Conditional Probabilities

Conditional Probability of each product given Gender

In [538]:
```
pd.crosstab(index = data[ 'Gender' ], columns = data[ 'Product' ], margins = True , normalize = 'index' ).round(3)
```

Out[538]:

Product Gender	KP281	KP481	KP781
Female	0.526	0.382	0.092
Male	0.385	0.298	0.317
All	0.444	0.333	0.222

```
P(KP281/Female): 0.526
P(KP481/Female): 0.382
P(KP781/Female): 0.092

P(KP281/Male): 0.384
P(KP481/Male): 0.298
P(KP781/Male): 0.317
```

Conditional Probability of each product given Marital Status

In [539]:
```
pd.crosstab(index = data[ 'MaritalStatus' ], columns = data[ 'Product' ], margins = True , normalize = 'index' ).round(3)
```

Out[539]:

Product MaritalStatus	KP281	KP481	KP781
Partnered	0.449	0.336	0.215
Single	0.438	0.329	0.233
All	0.444	0.333	0.222

```
P(KP281/Single): 0.438
P(KP481/Single): 0.329
```

```
P(KP781/Single): 0.233

P(KP281/Partnered): 0.449
P(KP481/Partnered): 0.336
P(KP781/Partnered): 0.215
```

Conditional Probability of each product given Usage

In [540]:
```python
pd.crosstab(index = data[ 'Usage' ], columns = data[ 'Product' ], margins = True , normalize = 'index' ).round(3)
```

Out[540]:

Product Usage	KP281	KP481	KP781
2	0.576	0.424	0.000
3	0.536	0.449	0.014
4	0.423	0.231	0.346
5	0.118	0.176	0.706
6	0.000	0.000	1.000
7	0.000	0.000	1.000
All	0.444	0.333	0.222

```
P(KP281/2): 0.576
P(KP481/2): 0.424
P(KP781/2): 0.000

P(KP281/3): 0.536
P(KP481/3): 0.449
P(KP781/3): 0.014

P(KP281/4): 0.423
P(KP481/4): 0.231
P(KP781/4): 0.346

P(KP281/5): 0.118
P(KP481/5): 0.176
P(KP781/5): 0.706

P(KP281/6): 0.000
```

```
P(KP481/6): 0.000
P(KP781/6): 1.000

P(KP281/7): 0.000
P(KP481/7): 0.000
P(KP781/7): 1.00
```

In [ ]:

Conditional Probability of each product given Fitness

In [541]:

```
pd.crosstab(index = data[ 'Fitness' ], columns = data[ 'Product' ], margins = True , normalize = 'index' ).round(3)
```

Out[541]:

Product / Fitness	KP281	KP481	KP781
1	0.500	0.500	0.000
2	0.538	0.462	0.000
3	0.557	0.402	0.041
4	0.375	0.333	0.292
5	0.065	0.000	0.935
All	0.444	0.333	0.222

```
P(KP281/1): 0.500
P(KP481/1): 0.500
P(KP781/1): 0.000

P(KP281/2): 0.538
P(KP481/2): 0.462
P(KP781/2): 0.000

P(KP281/3): 0.557
P(KP481/3): 0.402
P(KP781/3): 0.041

P(KP281/4): 0.375
P(KP481/4): 0.333
P(KP781/4): 0.292
```

```
P(KP281/5): 0.065
P(KP481/5): 0.000
P(KP781/5): 0.935
```

In [ ]:

In [ ]:

Product wise Customer Profile

1. KP281

1. Around 44.44 % of customers bought KP281 Product
2. Male and Females equally bought this product
3. Most of the customers who bought KP281 are Partnered
4. Median age of customer buying KP281 is 26
5. Education of customers who bought KP281 is between 14 to 16 years
6. Most of the customers expects to use treadmill 3 to 4 days per week.
7. Fitness level for the customers using KP281 is 3 i.e., Moderate
8. Customers who bought this treadmill have income less than 60k with an average of 55K.
9. Around 37% of total Income comes from this model which is highest of all treadmills

2. KP481

1. Around 33.33 % of customers bought KP481 Product
2. Male customers bought this product slightly more than Females.
3. Most of the customers who bought KP481 are Partnered
4. Median age of customer buying KP481 is 26
5. Education of customers who bought KP281 is between 14 to 16 years
6. Most of the customers expects to use treadmill 3 days per week.
7. Fitness level for the customers using KP481 is 3 i.e., Moderate
8. Around 32% of total Income comes from this model which is second highest of all treadmills

3. KP781

1. Around 22.22 % of customers bought KP781 Product
2. Male bought this product most likely.
3. Most of the customers who bought KP781 are Partnered
4. Median age of customer buying KP781 is 27
5. Education of customers who bought KP281 is between 16 to 18 years
6. Most of the customers expects to use treadmill 4 to 5 days per week.
7. Fitness level for the customers using KP781 is 4 to 5 i.e., High
8. Customers who bought this treadmill have income More than 60k with an average of 55K.
9. Around 31% of total Income comes from this model.

Recommandations

```
1. KP281 & KP481 attracts people with income less than 60k , may be because of cost of both models, so we can market it as budget friendly treadmills as they are popular in bothe men and women.
2. KP781 Product seems attracts to youth and people with high fitness so we can advertise it for athletes.
3. We should advertise KP781 Product with extra feature and benefits to attracts customers with long term benefits, by providing services after purchase as well
4. We can see sales from female customers for KP781 are very less to increase this sells we can provide offers on women special days etc.
5. From the analysis 18 to 35 age group people tends to buy treadmills so we nned to see how we can increase sells for other age groups as well.
6. Partnered people seems to buy KP781 Product more compared to others, so we should focus on sales for singles as well by running campaigns in colleges.
```