```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749
```

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv?1639992749
To: /content/aerofit_treadmill.csv?1639992749
100% 7.28k/7.28k [00:00<00:00, 10.2MB/s]
```

Introduction:

This dataset is all about individuals who purchased a treadmill from the AeroFit stores during the prior three months. 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.

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 products that are to be analysed using this dataset is as follows:

1. The KP281 is an entrylevel treadmill that sells for $1500
2. The KP481 is an midlevel treadmill that sells for $1,7500
3. The KP781 is having advanced features that sell for $2500

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

Basic Checks/ Basic Data analysis

```
#fetching the data
df = pd.read_csv('/content/aerofit_treadmill.csv?1639992749')
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

```
#knowing the basic details of data
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
```

```
#getting the numerical and statistical data description
df.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

```
#to know about the non numerical(categorical) data description
df.describe(include = 'object')
```

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

```
# to get the number of rows and columns
df.shape
```

```
(180, 9)
```

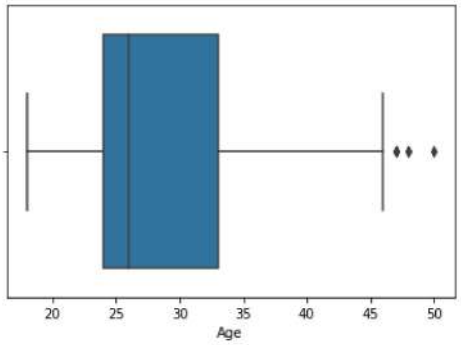
Inference:

1. The data has total 180 rows and 9 columns will no null values. This a very clean dataset.
2. The basic analysis also tells that the KP281 is the most bought with male customers and partnered customers getting the aerofit treadmills the most.
3. The minimum age of the customer who bought the treadmill is 18 and the maximum is 50.
4. The customers who uses the treadmill has got a minimum education for 12 years and maximum for 21 years.
5. The customers who bought it use it minimum 2 times a week and maximum 7 times a week. On an average it is used 3.5 times a week
6. The minimum fitness scale rated is 1 (poor) and maximum is 5(excellent) with 3.3 being the average rating overall.
7. The customers who purchased this has got a minimum salary of $29562 and the maximum salary being 104581$.
8. The minimum miles covered in a week is 21 while the maximum being 360. However, the average miles covered using the treadmill per week is 103.2.

Outlier Analysis using box plot

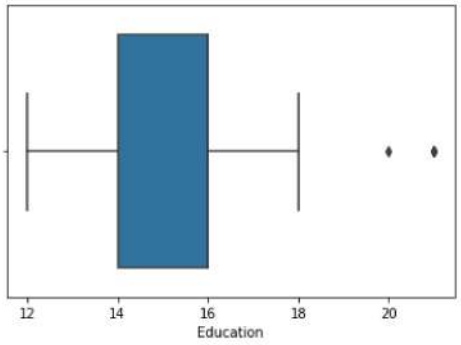
```
#to find outliers in age
sns.boxplot(df['Age'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89db73310>
```



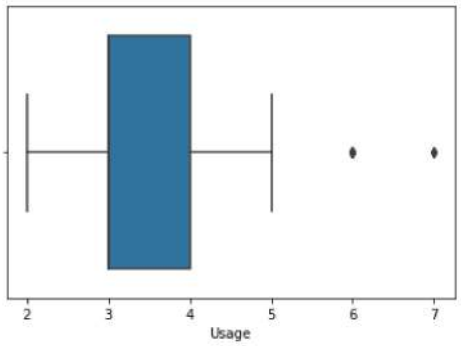
```
#to find outliers in education
sns.boxplot(df['Education'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89dab7e80>
```



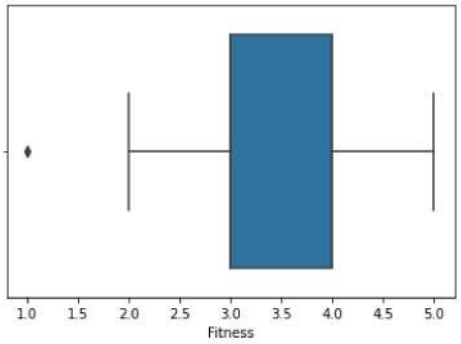

```
# to find outliers in usage
sns.boxplot(df['Usage'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d659910>
```



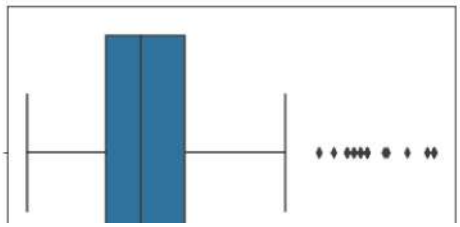

```
# to find outliers in fitness
sns.boxplot(df['Fitness'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d5b8040>
```



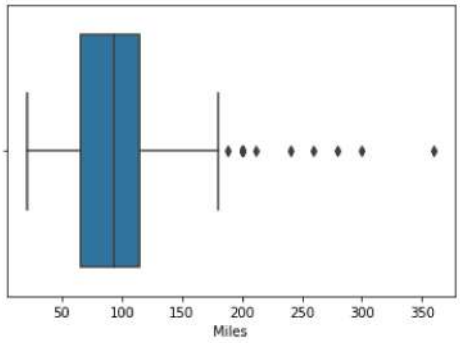

```
#to find outliers in income
sns.boxplot(df['Income'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d59b760>
```



```
# to find outliers in miles
sns.boxplot(df['Miles'])
```

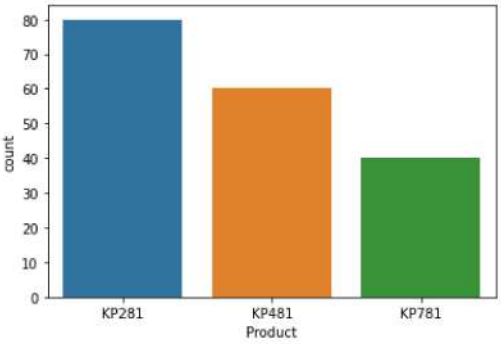
```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d59b640>
```



Univariate analysis(categorical data distribution/ description)

```
#to get distribution of product
sns.countplot(df['Product'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d4c9df0>
```

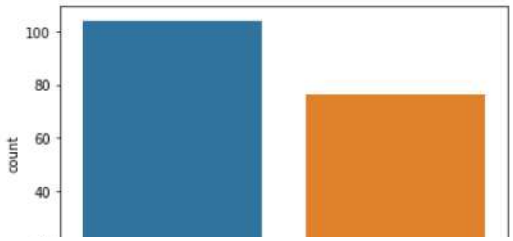



```
df['Product'].value_counts(normalize = True)
```

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

```
#to get distribution of gender
sns.countplot(df['Gender'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d4806a0>
```

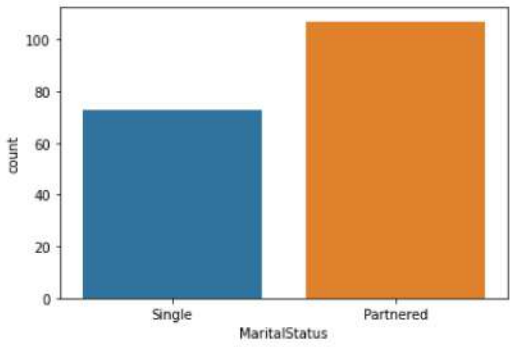



```
df['Gender'].value_counts(normalize = True)
```

```
Male      0.577778
Female    0.422222
Name: Gender, dtype: float64
```

```
#to get distribution of marital status
sns.countplot(df['MaritalStatus'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d3f3250>
```




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

```
Partnered    0.594444
Single       0.405556
Name: MaritalStatus, dtype: float64
```

Inference:

1. The probability of customers buying KP281 is 0.44, KP481 is 0.33 and KP781 is 0.22
2. The probability of male customers buying treadmill is 0.58 and female is 0.42
3. The probability of partnered customers getting the treadmill is 0.6 and single is 0.4

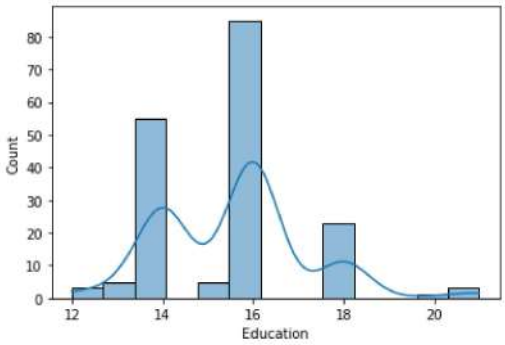
Univariate Analysis(numerical data distribution)

```
sns.histplot(df['Age'], kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d3b7a00>
```

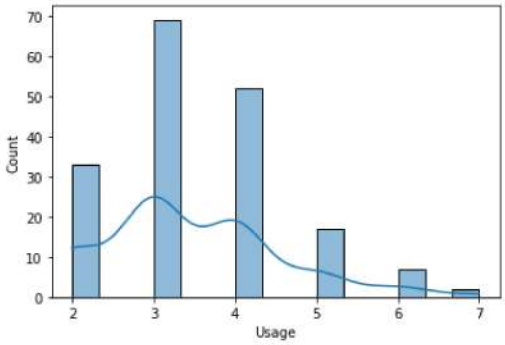
```
sns.histplot(df['Education'], kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b2e6070>
```



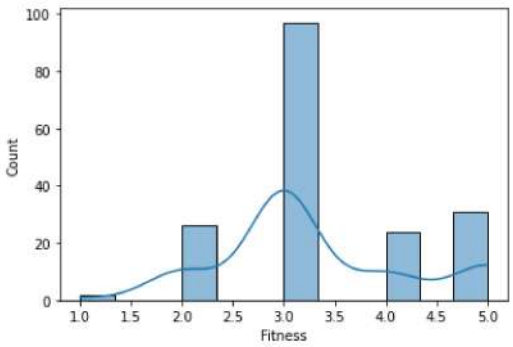

```
sns.histplot(df['Usage'], kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b2c6910>
```



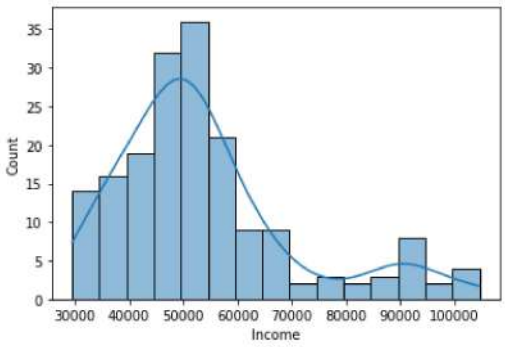

```
sns.histplot(df['Fitness'], kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b309520>
```



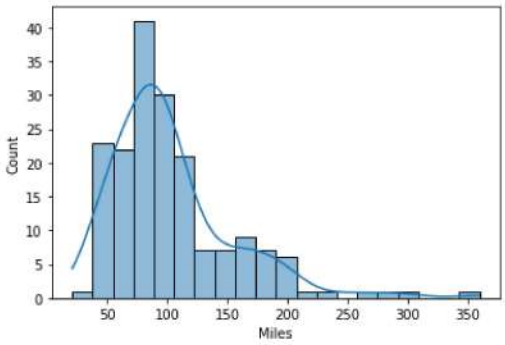

```
sns.histplot(df['Income'], kde = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b1645e0>
```




```
sns.histplot(df['Miles'], kde = True)
```

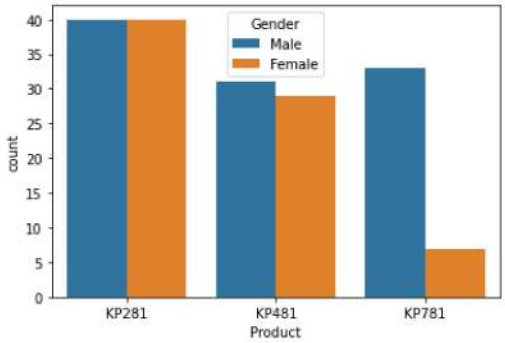
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b0e99a0>
```



Bivariate Analysis

```
sns.countplot(df['Product'], hue = df['Gender'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b082ac0>
```

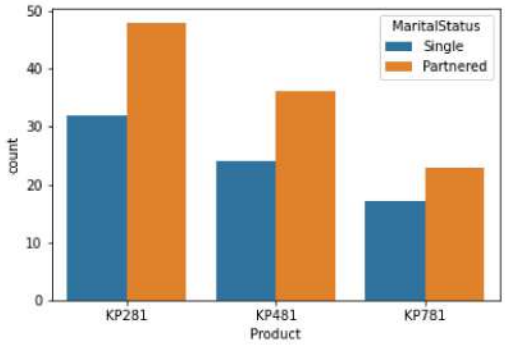



```
df.groupby(['Product'])['Gender'].value_counts(normalize = True)
```

```
Product  Gender
KP281    Female    0.500000
         Male      0.500000
KP481    Male      0.516667
         Female    0.483333
KP781    Male      0.825000
         Female    0.175000
Name: Gender, dtype: float64
```

```
sns.countplot(df['Product'], hue = df['MaritalStatus'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b1e40d0>
```



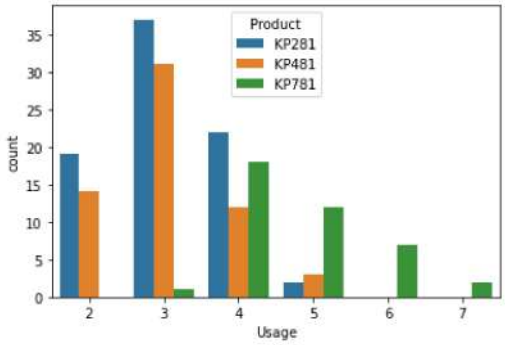

```
df.groupby(['Product'])['MaritalStatus'].value_counts(normalize = True)
```

```
Product  MaritalStatus
KP281    Partnered        0.600
```

```
            Single          0.400
KP481       Partnered       0.600
            Single          0.400
KP781       Partnered       0.575
            Single          0.425
Name: MaritalStatus, dtype: float64
```

```
sns.countplot(df['Usage'], hue = df['Product'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From \
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89d3ae310>
```




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

```
Product  Usage
KP281    3        0.462500
         4        0.275000
         2        0.237500
         5        0.025000
KP481    3        0.516667
         2        0.233333
         4        0.200000
         5        0.050000
KP781    4        0.450000
         5        0.300000
         6        0.175000
         7        0.050000
         3        0.025000
Name: Usage, dtype: float64
```

```
plt.figure(figsize = (15,8))
sns.countplot(df['Miles'], hue = df['Product'])
```

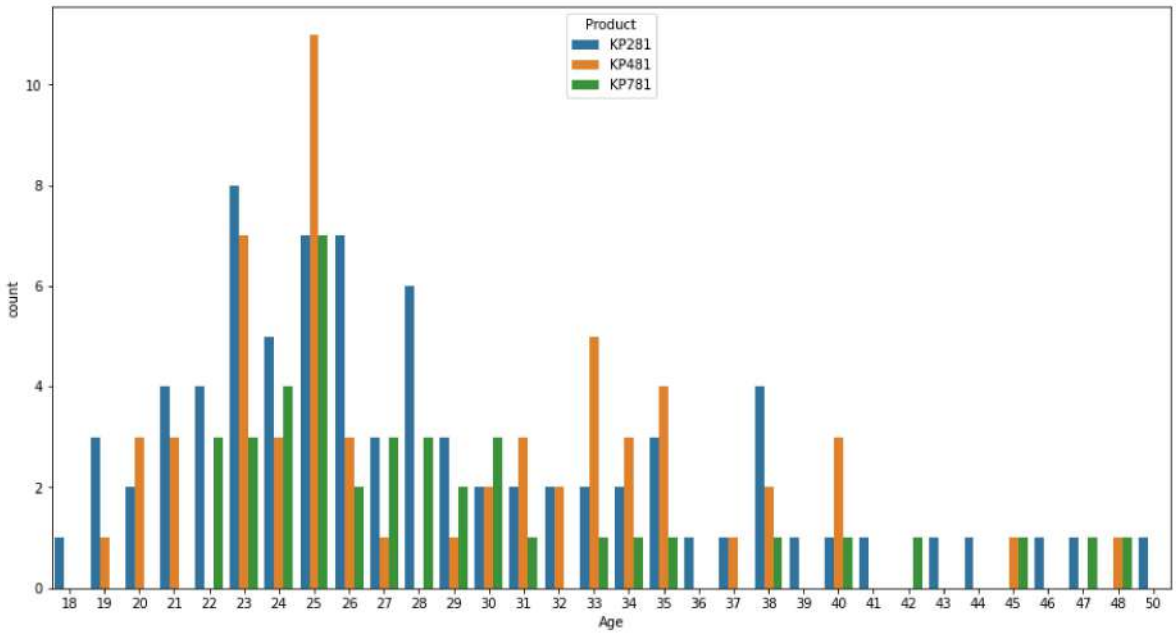
```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
```

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

```
Product  Miles
KP281    85       0.200000
         66       0.125000
         75       0.125000
         47       0.112500
         94       0.100000
         113      0.100000
         56       0.075000
         38       0.037500
         103      0.037500
         132      0.025000
         141      0.025000
         112      0.012500
         169      0.012500
         188      0.012500
KP481    95       0.200000
         85       0.183333
         106      0.133333
         53       0.116667
         64       0.100000
         127      0.083333
         42       0.066667
         74       0.050000
         170      0.033333
         21       0.016667
         212      0.016667
KP781    100      0.175000
         180      0.150000
         200      0.150000
         160      0.125000
         150      0.100000
         120      0.075000
         80       0.025000
         106      0.025000
         140      0.025000
         170      0.025000
         240      0.025000
         260      0.025000
         280      0.025000
         300      0.025000
         360      0.025000
Name: Miles, dtype: float64
```

```
plt.figure(figsize = (15,8))
sns.countplot(df['Age'], hue = df['Product'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89b072a30>
```

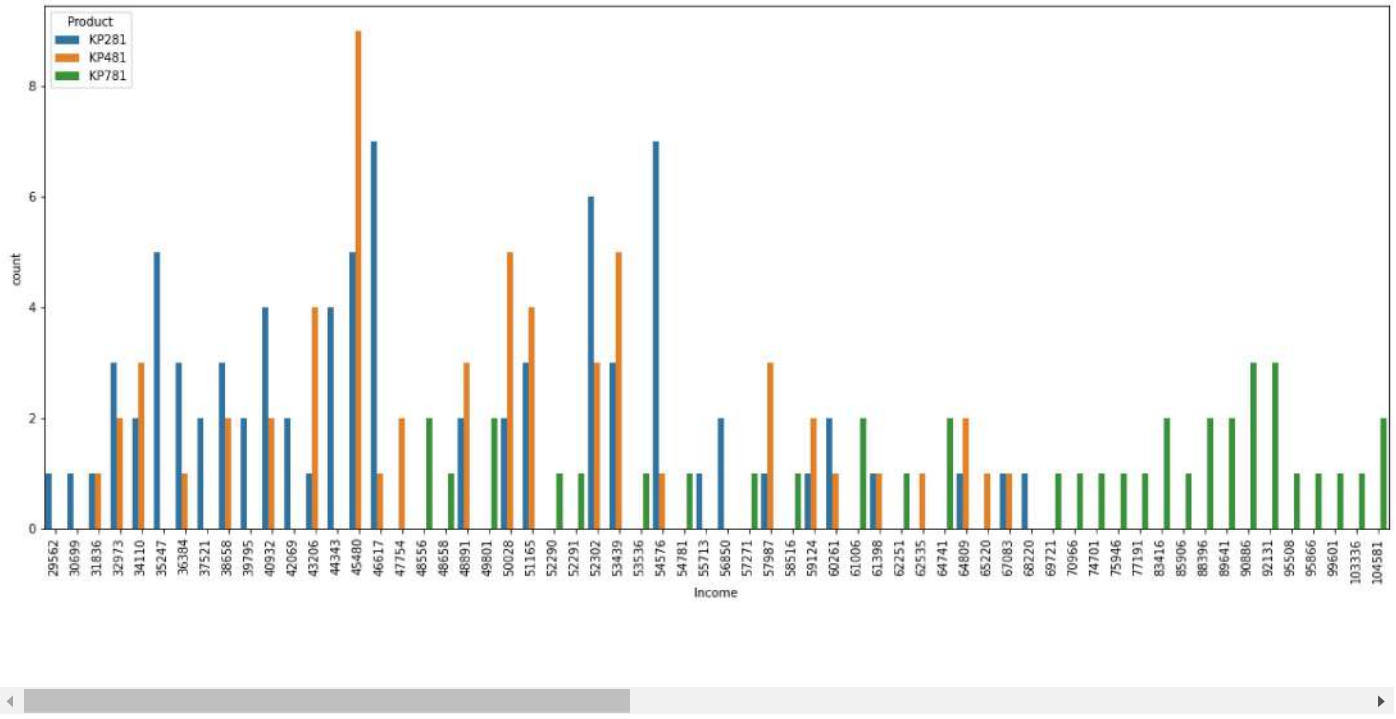


```
df.groupby(['Product'])['Age'].value_counts(normalize = True)
```

```
Product  Age
KP281    23     0.1000
         25     0.0875
         26     0.0875
         28     0.0750
         24     0.0625
                 ...
KP781    40     0.0250
         42     0.0250
         45     0.0250
         47     0.0250
         48     0.0250
Name: Age, Length: 68, dtype: float64
```

```
plt.figure(figsize = (20,8))
sns.countplot(df['Income'], hue = df['Product'])
plt.xticks(Rotation= 90)
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
        51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61]),
 <a list of 62 Text major ticklabel objects>)
```

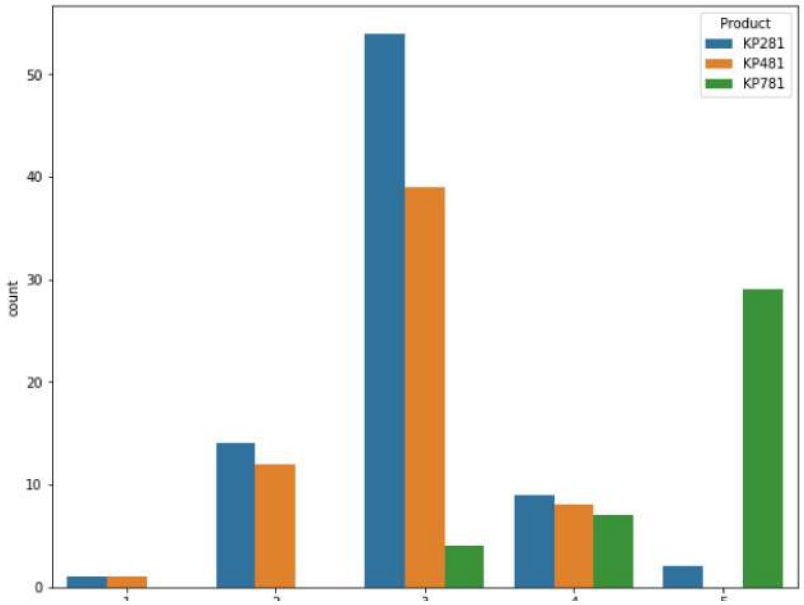



```
df.groupby(['Product'])['Income'].value_counts(normalize = True)
```

```
Product  Income
KP281    46617     0.0875
         54576     0.0875
         52302     0.0750
         35247     0.0625
         45480     0.0625
                    ...
KP781    85906     0.0250
         95508     0.0250
         95866     0.0250
         99601     0.0250
         103336    0.0250
Name: Income, Length: 83, dtype: float64
```

```
plt.figure(figsize = (10,8))
sns.countplot(df['Fitness'], hue = df['Product'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89a9c4ac0>
```

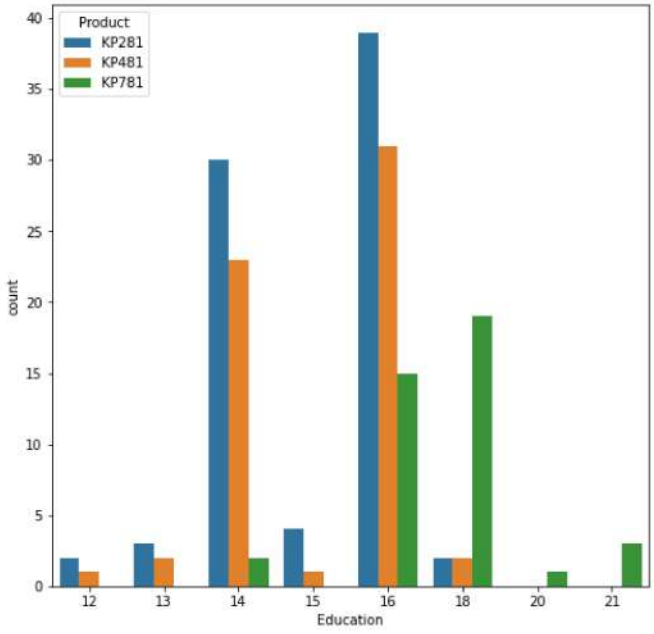



```
df.groupby(['Product'])['Fitness'].value_counts(normalize = True)
```

```
Product  Fitness
KP281    3          0.675000
         2          0.175000
         4          0.112500
         5          0.025000
         1          0.012500
KP481    3          0.650000
         2          0.200000
         4          0.133333
         1          0.016667
KP781    5          0.725000
         4          0.175000
         3          0.100000
Name: Fitness, dtype: float64
```

```
plt.figure(figsize = (8,8))
sns.countplot(df['Education'], hue = df['Product'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From v
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89abbed30>
```



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

```
Product  Education
KP281    16           0.487500
         14           0.375000
         15           0.050000
         13           0.037500
         12           0.025000
         18           0.025000
KP481    16           0.516667
         14           0.383333
         13           0.033333
         18           0.033333
         12           0.016667
         15           0.016667
KP781    18           0.475000
         16           0.375000
         21           0.075000
         14           0.050000
         20           0.025000
Name: Education, dtype: float64
```

Inference:

1. The probability of male and female customers buying KP281 is 0.5 each
2. The probability of male customers getting KP481 is 0.52 and female is 0.48
3. The probabilit of male customers getting KP781 is 0.83 and female is 0.17
4. The probability of partnered getting KP281 and KP481 is 0.6 each and 0.58 incase of KP781
5. The KP281 is used 3 times a week maximum with a probability 0.46 and 5 times the least with probability being 0.02, KP481 is used 3 times most with probability 0.52 and 5 times least with 0.05 probability. KP781 is used most for 4 times per week with 0.45 probability and 7 or 3 times least per week with probability 0.05 or 0.02
6. The people who use KP281 cover a minimum of 38 miles per week upto a maximum of 188 miles per week.
7. The people who use KP481 cover a minimum of 21 miles per week upto a maximum of 212 miles per week.
8. The people who use KP781 cover a minimum of 80 miles per week upto a maximum of 360 miles per week.
9. People with salary greater than $48000 only purchase KP781
10. People who use KP281 and KP481 rate them on fitness scale 3 mostly with the probability 0.68 and 0.65 respectively
11. People who use KP781 mostly rate them to be 5 with the probability 0.73
12. Most people who purchase KP781 has got 14 years of education atleast

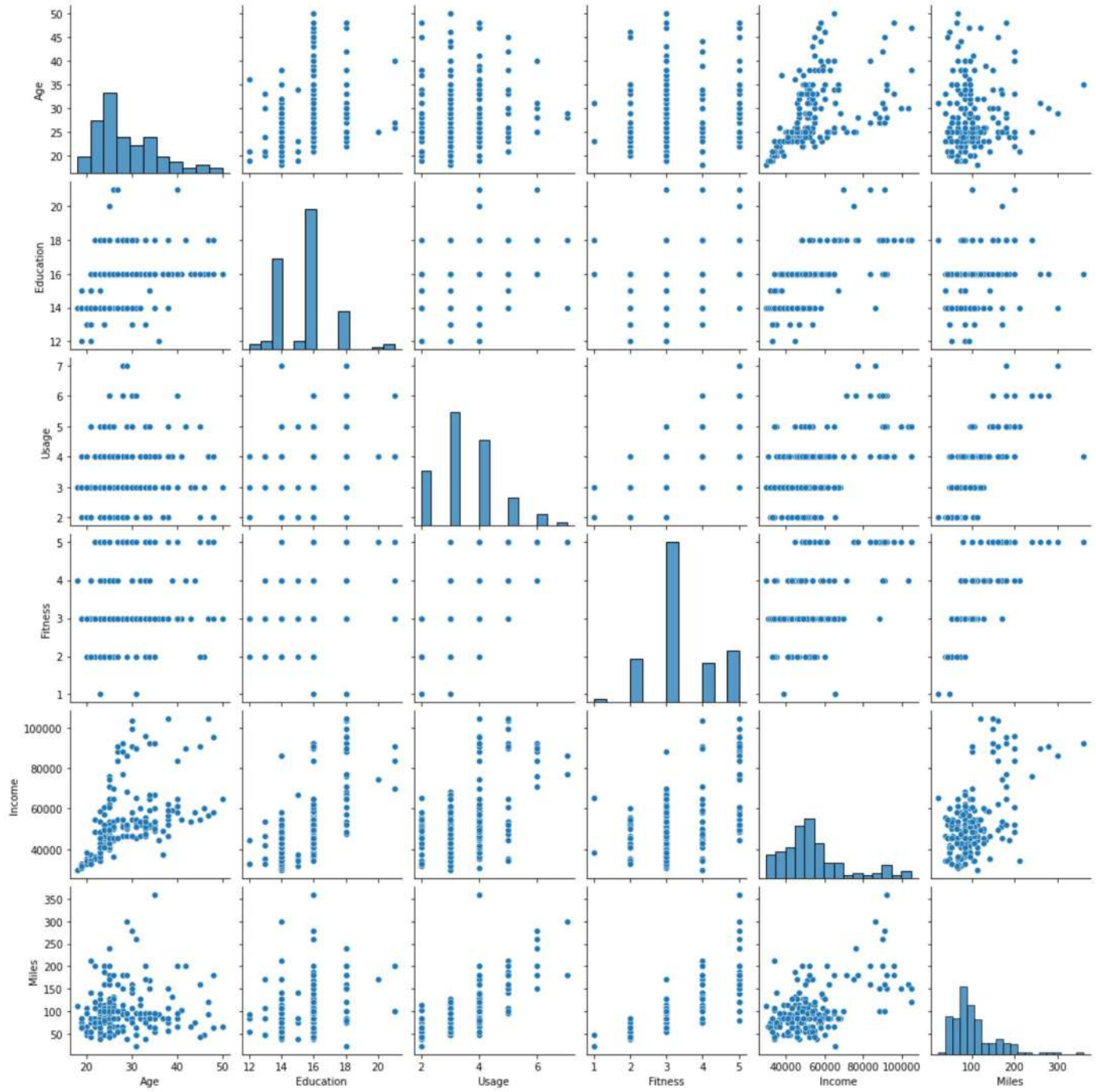
Correlation and pairplot

```
plt.figure(figsize = (10,10))
sns.heatmap(df.corr(), annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb89a810940>
```


```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7fb89ad6d250>
```



Inference:

1. Age and income is highly correlated
2. Education and income is highly correlated aswell
3. Usage and fitness aswell as usage and miles are highly correlated
4. Miles and fitness are highly correlated aswell

General Analysis

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

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

```
#correlation
df.corr()
```

	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

```
#to check on conditional probability
df1 = pd.crosstab(index= df['Gender'], columns = df['Product'], margins = True)
df1
```

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

```
#probability of male using KP281 given KP281 users
d_281m = df1['KP281']['Male']/df1['KP281']['All']
d_281m
```

```
0.5
```

```
#probability of female using KP281 given KP281 users
d_281f = df1['KP281']['Female']/df1['KP281']['All']
d_281f
```

```
0.5
```

```
#probability of male using KP281 given male users
d_281_m = df1['KP281']['Male']/df1['All']['Male']
d_281_m
```

```
0.38461538461538464
```

```
#probability of female using KP281 given female users
d_281_f = df1['KP281']['Female']/df1['All']['Female']
d_281_f
```

```
0.5263157894736842
```

```
#probaility of female using aerofit products
f = df1['All']['Female']/len(df)
f
```

```
0.4222222222222222
```

```
#probability of male using aerofit products
m = df1['All']['Male']/len(df)
m
```

```
0.5777777777777777
```

```
#probability of male using KP481 given KP481 users
d_481m = df1['KP481']['Male']/df1['KP481']['All']
d_481m
```

```
0.5166666666666667
```

```
#probability of female using KP481 given KP481 users
d_481f = df1['KP481']['Female']/df1['KP481']['All']
d_481f
```

```
0.48333333333333334
```

```
#probability of male using KP481 given male users
d_481_m = df1['KP481']['Male']/df1['All']['Male']
d_481_m
```

```
0.2980769230769231
```

```
#probability of female using KP481 given female users
d_481_f = df1['KP481']['Female']/df1['All']['Female']
d_481_f
```

```
0.3815789473684211
```

```
#probability of male using KP781 given KP781 users
d_781m = df1['KP781']['Male']/df1['KP781']['All']
d_781m
```

```
0.825
```

```
#probability of female using KP781 given KP781 users
d_781f = df1['KP781']['Female']/df1['KP781']['All']
d_781f
```

```
0.175
```

```
#probability of male using KP781 given male users
d_781_m = df1['KP781']['Male']/df1['All']['Male']
d_781_m
```

```
0.3173076923076923
```

```
#probability of female using KP781 given female users
d_781_f = df1['KP781']['Female']/df1['All']['Female']
d_781_f
```

```
0.09210526315789473
```

```
#to check on conditional probability
df2 = pd.crosstab(index= df['MaritalStatus'], columns = df['Product'], margins = True)
df2
```

Product	KP281	KP481	KP781	All
MaritalStatus				
Partnered	48	36	23	107
Single	32	24	17	73
All	80	60	40	180

```
#probaility of single using aerofit products
s = df2['All']['Single']/len(df)
s
```

```
0.40555555555555556
```

```
#probaility of partnered using aerofit products
mr = df2['All']['Partnered']/len(df)
mr
```

```
0.5944444444444444
```

```
#probability of single using KP781 given single users
d_781_s = df2['KP781']['Single']/df2['All']['Single']
d_781_s
```

```
0.2328767123287671
```

```
#probability of Partenered using KP781 given Partenered users
d_781_p = df2['KP781']['Partnered']/df2['All']['Partnered']
d_781_p
```

```
0.21495327102803738
```

```
#probability of single using KP781 given KP781 users
d_781s = df2['KP781']['Single']/df2['KP781']['All']
d_781s
```

```
0.425
```

```
#probability of partnered using KP781 given KP781 users
d_781p = df2['KP781']['Partnered']/df2['KP781']['All']
d_781p
```

```
0.575
```

```
#probability of single using KP481 given single users
d_481_s = df2['KP481']['Single']/df2['All']['Single']
d_481_s
```

```
0.3287671232876712
```

```
#probability of partnered using KP481 given partnered users
d_481_p = df2['KP481']['Partnered']/df2['All']['Partnered']
d_481_p
```

```
0.3364485981308411
```

```
#probability of single using KP481 given KP481 users
d_481s = df2['KP481']['Single']/df2['KP481']['All']
d_481s
```

```
0.4
```

```
#probability of partnered using KP481 given KP481 users
d_481p = df2['KP481']['Partnered']/df2['KP481']['All']
d_481p
```

```
0.6
```

```
#probability of single using KP281 given single users
d_281_s = df2['KP281']['Single']/df2['All']['Single']
d_281_s
```

```
0.4383561643835616
```

```
#probability of partnered using KP281 given partnered users
d_281_p = df2['KP281']['Partnered']/df2['All']['Partnered']
d_281_p
```

```
0.4485981308411215
```

```
#probability of single using KP281 given KP281 users
d_281s = df2['KP281']['Single']/df2['KP281']['All']
```

```
d_281s
```

```
0.4
```

```
#probability of partnered using KP281 given KP281 users
d_281p = df2['KP281']['Partnered']/df2['KP281']['All']
d_281p
```

```
0.6
```

```
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

```
df[df['Age']<20]
```

	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
80	KP481	19	Male	14	Single	3	3	31836	64

```
#probability of teenagers (18 to 20(excl)) using the aerofit product
len(df[df['Age']<20])/len(df)
```

```
0.027777777777777776
```

```
df[df['Age']<20]['Gender'].value_counts(normalize = True)
```

```
Male      0.8
Female    0.2
Name: Gender, dtype: float64
```

```
df[df['Age']<20]['Education'].value_counts(normalize = True)
```

```
14    0.6
15    0.2
12    0.2
Name: Education, dtype: float64
```

```
df[df['Age']<20]['MaritalStatus'].value_counts(normalize = True)
```

```
Single       0.8
Partnered    0.2
Name: MaritalStatus, dtype: float64
```

```
df[df['Age']<20]['Usage'].value_counts(normalize = True)
```

```
3    0.6
2    0.2
4    0.2
Name: Usage, dtype: float64
```

```
df[df['Age']<20]['Usage'].mean()
```

```
3.0
```

```
df[df['Age']<20]['Fitness'].value_counts(normalize = True)
```

```
3    0.8
4    0.2
Name: Fitness, dtype: float64
```

```
df[df['Age']<20]['Fitness'].mean()
```

```
3.2
```

```
df[df['Age']<20]['Income'].value_counts(normalize = True)
```

```
31836    0.4
29562    0.2
30699    0.2
32973    0.2
Name: Income, dtype: float64
```

```
df[df['Age']<20]['Income'].mean()
```

```
31381.2
```

```
df[df['Age']<20]['Miles'].value_counts(normalize = True)
```

```
112    0.2
75     0.2
66     0.2
85     0.2
64     0.2
Name: Miles, dtype: float64
```

```
df[df['Age']<20]['Miles'].mean()
```

```
80.4
```

Inference:

1. 0.027 person in the age range 18 to 20(excl) use KP281 treadmill of which 0.8 i.e 80% in that age group who use this is male and remaining 0.2 i.e 20% is female.
2. The average education in this age group is 13.8 years of which 0.6 have 14years, 0.2 has 15years and 0.2 has 12years. 80% are single(0.8) and 20%(0.2) are partnered.
3. The average number of times it is being used in a week is 3 of which 0.6 use 3 times a week, 0.2 use 2 times a week and 0.2 use 4 times a week.
4. The average fitness rate scale is 3.2 of which 0.8 rate themselves t be 3 and 0.2 to be 4.
5. The average income is 31381.2 of which 0.4 get $31836 and the average miles covered in a week is 80.4.

```
df_103 = df[(df['Age']>=20) & (df['Age'] < 30)]
df_103
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
4	KP281	20	Male	13	Partnered	4	2	35247	47
5	KP281	20	Female	14	Partnered	3	3	32973	66
6	KP281	21	Female	14	Partnered	3	3	35247	75
7	KP281	21	Male	13	Single	3	3	32973	85
8	KP281	21	Male	15	Single	5	4	35247	141
...	...	...	...	...	...	...	...	...	...
162	KP781	28	Female	18	Partnered	6	5	92131	180
163	KP781	28	Male	18	Partnered	7	5	77191	180
164	KP781	28	Male	18	Single	6	5	88396	150
165	KP781	29	Male	18	Single	5	5	52290	180
166	KP781	29	Male	14	Partnered	7	5	85906	300

108 rows × 9 columns

```
len(df[(df['Age']>=20) & (df['Age'] < 30)])/len(df)
```

```
0.6
```

```
df_103['Product'].value_counts(normalize = True)
```

```
KP281    0.453704
KP481    0.296296
KP781    0.250000
Name: Product, dtype: float64
```

```
df_103['Age'].value_counts(normalize = True)
```

```
25    0.231481
23    0.166667
24    0.111111
26    0.111111
28    0.083333
21    0.064815
22    0.064815
27    0.064815
29    0.055556
20    0.046296
Name: Age, dtype: float64
```

```
df_103['Gender'].value_counts(normalize = True)
```

```
Male      0.564815
Female    0.435185
Name: Gender, dtype: float64
```

```
df_103['Education'].value_counts(normalize = True)
```

```
16    0.398148
14    0.388889
18    0.120370
13    0.027778
15    0.027778
21    0.018519
12    0.009259
20    0.009259
Name: Education, dtype: float64
```

```
df_103['MaritalStatus'].value_counts(normalize = True)
```

```
Partnered    0.564815
Single       0.435185
Name: MaritalStatus, dtype: float64
```

```
df_103['Usage'].value_counts(normalize = True)
```

```
3    0.351852
4    0.314815
2    0.185185
5    0.092593
6    0.037037
7    0.018519
Name: Usage, dtype: float64
```

```
df_103['Usage'].mean()
```

```
3.5
```

```
df_103['Fitness'].value_counts(normalize = True)
```

```
3    0.527778
5    0.175926
2    0.157407
4    0.129630
1    0.009259
Name: Fitness, dtype: float64
```

```
df_103['Fitness'].mean()
```

```
3.3055555555555554
```

```
df_103['Income'].value_counts(normalize = True)
```

```
45480    0.120370
40932    0.055556
35247    0.046296
50028    0.046296
38658    0.046296
34110    0.046296
43206    0.046296
32973    0.037037
52302    0.037037
51165    0.037037
36384    0.037037
44343    0.027778
46617    0.027778
48891    0.027778
54576    0.027778
49801    0.018519
61006    0.018519
88396    0.018519
48556    0.018519
64741    0.018519
39795    0.018519
42069    0.018519
53439    0.018519
92131    0.009259
77191    0.009259
90886    0.009259
70966    0.009259
83416    0.009259
69721    0.009259
74701    0.009259
75946    0.009259
52290    0.009259
48658    0.009259
62251    0.009259
52291    0.009259
57271    0.009259
53536    0.009259
58516    0.009259
54781    0.009259
47754    0.009259
68220    0.009259
37521    0.009259
85906    0.009259
Name: Income, dtype: float64
```

```
df_103['Income'].mean()
```

```
49326.0462962963
```

```
df_103['Miles'].value_counts(normalize = True)
```

```
85     0.120370
106    0.064815
113    0.064815
100    0.064815
47     0.055556
75     0.055556
180    0.046296
66     0.046296
95     0.046296
53     0.046296
94     0.046296
56     0.037037
127    0.037037
42     0.027778
160    0.027778
200    0.027778
38     0.018519
103    0.018519
64     0.018519
170    0.018519
120    0.018519
240    0.009259
150    0.009259
80     0.009259
212    0.009259
140    0.009259
74     0.009259
132    0.009259
```

```
188    0.009259
141    0.009259
300    0.009259
Name: Miles, dtype: float64
```

```
df_103['Miles'].mean()
```

```
102.73148148148148
```

Inference:

1. 0.6 in the age group 20 to 30(excl) use aerofit treadmill of which 0.45 use KP281, 0.3 use KP481 and 0.25 use KP781.
2. 0.56 are male and 0.44 are female in this age group of which 0.56 are partnered and 0.44 are single.
3. The average usage per week is 3.5 times with a fitness scale average of 3.3.
4. The average income who buy the treadmill in this age group is $49326.
5. The average miles it is used is 102.7

```
df_104 = df[(df['Age']>=30) & (df['Age'] < 40)]
df_104
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
53	KP281	30	Male	14	Partnered	4	4	46617	141
54	KP281	30	Male	14	Single	3	3	54576	85
55	KP281	31	Male	14	Partnered	2	2	54576	47
56	KP281	31	Female	14	Single	2	2	45480	47
57	KP281	32	Female	14	Single	3	4	46617	113
58	KP281	32	Male	14	Partnered	4	3	52302	85
59	KP281	33	Female	16	Single	2	2	55713	38
60	KP281	33	Female	16	Partnered	3	3	46617	85
61	KP281	34	Male	16	Single	4	5	51165	169
62	KP281	34	Female	16	Single	2	2	52302	66
63	KP281	35	Male	16	Partnered	4	3	48891	85
64	KP281	35	Female	16	Partnered	3	3	60261	94
65	KP281	35	Female	18	Single	3	3	67083	85
66	KP281	36	Male	12	Single	4	3	44343	94
67	KP281	37	Female	16	Partnered	3	3	37521	85
68	KP281	38	Male	16	Partnered	3	3	46617	75
69	KP281	38	Female	14	Partnered	2	3	54576	56
70	KP281	38	Male	14	Single	2	3	52302	56

```
len(df_104)/len(df)
```

```
0.2777777777777778
```

```
df_104['Product'].value_counts(normalize = True)
```

```
KP481    0.44
KP281    0.40
KP781    0.16
Name: Product, dtype: float64
```

118	KP481	32	Male	16	Single	4	3	60261	127

```
df_104['Age'].value_counts(normalize = True)
```

```
33    0.16
35    0.16
30    0.14
38    0.14
31    0.12
34    0.12
32    0.08
37    0.04
36    0.02
39    0.02
Name: Age, dtype: float64
```

126	KP481	34	Male	16	Partnered	3	4	59124	85

```
df_104['Gender'].value_counts(normalize = True)
```

```
Male      0.54
Female    0.46
Name: Gender, dtype: float64
```

130	KP481	35	Female	16	Single	3	2	50028	64

```
df_104['Education'].value_counts(normalize = True)
```

```
16    0.58
14    0.20
18    0.14
13    0.04
12    0.02
15    0.02
Name: Education, dtype: float64
```

168	KP781	30	Male	18	Partnered	5	4	103336	160

```
df_104['MaritalStatus'].value_counts(normalize = True)
```

```
Partnered    0.66
Single       0.34
Name: MaritalStatus, dtype: float64
```

```
df_104['Usage'].value_counts(normalize = True)
```

```
3    0.40
4    0.26
2    0.20
5    0.10
6    0.04
Name: Usage, dtype: float64
```

```
df_104['Usage'].mean()
```

```
3.38
```

```
df_104['Fitness'].value_counts(normalize = True)
```

```
3    0.54
5    0.16
4    0.14
2    0.14
1    0.02
Name: Fitness, dtype: float64
```

```
df_104['Fitness'].mean()
```

```
3.28
```

```
df_104['Income'].value_counts(normalize = True)
```

```
46617     0.10
52302     0.10
53439     0.10
59124     0.06
51165     0.06
54576     0.06
92131     0.04
50028     0.04
48891     0.04
60261     0.04
67083     0.04
90886     0.02
89641     0.02
103336    0.02
99601     0.02
64809     0.02
95866     0.02
62535     0.02
57987     0.02
47754     0.02
65220     0.02
56850     0.02
37521     0.02
44343     0.02
55713     0.02
45480     0.02
104581    0.02
Name: Income, dtype: float64
```

```
df_104['Income'].mean()
```

```
60305.92
```

```
df_104['Miles'].value_counts(normalize = True)
```

```
85     0.22
95     0.12
150    0.06
106    0.04
47     0.04
94     0.04
75     0.04
56     0.04
74     0.04
64     0.04
```

```
53     0.04
170    0.02
200    0.02
260    0.02
160    0.02
280    0.02
141    0.02
127    0.02
21     0.02
132    0.02
66     0.02
169    0.02
38     0.02
113    0.02
360    0.02
Name: Miles, dtype: float64
```

```
df_104['Miles'].mean()
```

```
106.6
```

Inference:

1. 0.28 in this age group (30 to 40 (excl)) use treadmill of which 0.44 use KP481, 0.40 use KP281 and 0.16 use KP781.
2. 0.54 who use treadmill is male and 0.46 is female.
3. The average usage perweek is 3.38 and average fitness scale is 3.28.
4. 0.66 are partnered and 0.34 are single.
5. The average income for this age group is $60305.92 and average miles covered in a week is 106.6 by this age group

```
df_105 = df[(df['Age']>=40) & (df['Age'] < 50)]
df_105
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
73	KP281	40	Male	16	Partnered	3	3	61398	66
74	KP281	41	Male	16	Partnered	4	3	54576	103
75	KP281	43	Male	16	Partnered	3	3	53439	66
76	KP281	44	Female	16	Single	3	4	57987	75
77	KP281	46	Female	16	Partnered	3	2	60261	47
78	KP281	47	Male	16	Partnered	4	3	56850	94
135	KP481	40	Female	16	Partnered	3	3	61398	85
136	KP481	40	Female	16	Single	3	3	57987	85
137	KP481	40	Male	16	Partnered	3	3	64809	95
138	KP481	45	Male	16	Partnered	2	2	54576	42
139	KP481	48	Male	16	Partnered	2	3	57987	64
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

```
len(df_105)/len(df)
```

```
0.08888888888888889
```

```
df_105['Product'].value_counts(normalize = True)
```

```
KP281    0.3750
KP481    0.3125
KP781    0.3125
Name: Product, dtype: float64
```

```
df_105['Age'].value_counts(normalize = True)
```

```
40    0.3125
47    0.1250
45    0.1250
48    0.1250
41    0.0625
43    0.0625
44    0.0625
46    0.0625
42    0.0625
Name: Age, dtype: float64
```

```
df_105['Gender'].value_counts(normalize = True)
```

```
Male      0.75
Female    0.25
Name: Gender, dtype: float64
```

```
df_105['MaritalStatus'].value_counts(normalize = True)
```

```
Partnered    0.6875
Single       0.3125
Name: MaritalStatus, dtype: float64
```

```
df_105['Education'].value_counts(normalize = True)
```

```
16    0.7500
18    0.1875
21    0.0625
Name: Education, dtype: float64
```

```
df_105['Usage'].value_counts(normalize = True)
```

```
3    0.4375
4    0.2500
2    0.1250
5    0.1250
6    0.0625
Name: Usage, dtype: float64
```

```
df_105['Usage'].mean()
```

```
3.5625
```

```
df_105['Fitness'].value_counts(normalize = True)
```

```
3    0.500
5    0.250
4    0.125
2    0.125
Name: Fitness, dtype: float64
```

```
df_105['Fitness'].mean()
```

```
3.5
```

```
df_105['Income'].value_counts(normalize = True)
```

```
57987     0.1875
61398     0.1250
54576     0.1250
53439     0.0625
60261     0.0625
56850     0.0625
64809     0.0625
83416     0.0625
89641     0.0625
90886     0.0625
104581    0.0625
95508     0.0625
Name: Income, dtype: float64
```

```
df_105['Income'].mean()
```

```
69081.25
```

```
df_105['Miles'].value_counts(normalize = True)
```

```
66     0.1250
85     0.1250
200    0.1250
103    0.0625
75     0.0625
47     0.0625
94     0.0625
95     0.0625
42     0.0625
64     0.0625
160    0.0625
120    0.0625
180    0.0625
Name: Miles, dtype: float64
```

```
df_105['Miles'].mean()
```

```
105.125
```

Inference:

1. 0.09 in this age group (40 to 50 (excl)) use treadmill of which 0.38 use KP281, 0.31 use KP481 and 0.31 use KP781.
2. 0.75 who use are male and 0.25 are female.
3. 0.69 who use in this age is partnered and 0.31 are single.
4. The average usage per week is 3.6 times and fitness scale average would be 3.5.
5. The average income of this age group is $69081 and 105 miles are covered per week in average

```
df_106 = df[(df['Age']>=50)]
df_106
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
79	KP281	50	Female	16	Partnered	3	3	64809	66

Inference:

A single partnered person (age = 50) with 16 years of education uses this basic treadmill 3 times a week and covers 66 miles per week and she rates her to be 3 with respect to fitness scale and her income is about $64809

Recommendations

1. People who would like to exercise more to be fit with a income more than $48000 and is in the age group of 40 to 50 could be given KP781
2. People in the age group 50 and above could be given basic KP281 treadmill just too make them stay active and to maintain their health
3. New users and teen (people in age group less than 20 years) can be given KP281
4. Male and female customers in the age group 20 to 45 can be given KP481 since the provided data shows that many female customers prefer KP481
5. Partnered customers who are not new users could be given KP481