ita4cq3bg

March 20, 2024

[ ]:

AEROFIT CASE STUDY

Aerofit is a top company in the field of exercise equipment and it provides various fitness equipment.

PROBLEM STATEMENT

The market research team at AeroFit wants to identify the characteristics of the target audience for each type of treadmill offered by the company, to provide a better recommendation of the treadmills to the new customers. The team decides to investigate whether there are differences across the product with respect to customer characteristics. 1. Perform descriptive analytics to create a customer profile for each AeroFit treadmill product by developing appropriate tables and charts. 2. For each AeroFit treadmill product, construct two-way contingency tables and compute all conditional and marginal probabilities along with their insights/impact on the business.

OBJECTIVE

1. Import the dataset and do usual data analysis steps like checking the structure & characteristics of the dataset
2. Detect Outliers (using boxplot, "describe" method by checking the difference between mean and median)
3. Check if features like marital status, age have any effect on the product purchased (using countplot, histplots, boxplots etc)
4. Representing the marginal probability like - what percent of customers have purchased KP281, KP481, or KP781 in a table (can use pandas.crosstab here)
5. Check correlation among different factors using heat maps or pair plots.
6. With all the above steps you can answer questions like: What is the probability of a male customer buying a KP781 treadmill?
7. Customer Profiling - Categorization of users.
8. Probability- marginal, conditional probability.
9. Some recommendations and actionable insights, based on the inferences.

BASIC METRICS

[ ]:
```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/
  ↪original/aerofit_treadmill.csv?1639992749

import pandas as pd
import numpy as np
import seaborn as sns
```

```
import matplotlib.pyplot as plt


new_filename = "aerofit_treadmill.csv?1639992749"


df = pd.read_csv(new_filename)


print(df.head())
```

```
--2024-03-20 17:25:43--  https://d2beiqkhq929f0.cloudfront.net/public_assets/ass
ets/000/001/125/original/aerofit_treadmill.csv?1639992749
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)…
108.157.172.183, 108.157.172.10, 108.157.172.173, …
Connecting to d2beiqkhq929f0.cloudfront.net
(d2beiqkhq929f0.cloudfront.net)|108.157.172.183|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 7279 (7.1K) [text/plain]
Saving to: ‘aerofit_treadmill.csv?1639992749.1’

aerofit_treadmill.c 100%[===================>]   7.11K  --.-KB/s    in 0s

2024-03-20 17:25:43 (2.02 GB/s) - ‘aerofit_treadmill.csv?1639992749.1’ saved
[7279/7279]

  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
```

This code is giving the first five rows of the data.signifying the age,gender,education,Marital status,usage,fitness level, income and miles crossed by the user in respect to different models of Treadmill.

[ ]:
```
print(f"Number of rows: {df.shape[0]}")
```

```
Number of rows: 180
```

this line gives the number of rows ih tne given dataset.

[ ]:
```
print(f"Number of columns: {df.shape[1]}")
```

```
Number of columns: 9
```

this line gives the number of columns ih tne given dataset.

[ ]:
```
print(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
None
```

this line gives about he nullity,count,data type of various elements in first row of the dataset.

[ ]:
```
print(df.describe())
```

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

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

Analysis

columns number = 180;

age = meanAge is 28,half of the customer's mean age is 26.

Education : mean Education age is 15 with maximum 21 and minimum as 12.

fitness: average rating is 3.3 on scale of 1 to 5.

Miles : average miles covered by people is 103 with maximum being 115 and minimum is 25.

usage : mean usage is 3.4 with maximum being 7 and minimum 2.

income : mostly earn 58,000 with maximum 1,04,000 and minimum 30,000.

count: The number of non-null values present in the column.

mean: The average of the values in the column.

std: The standard deviation of the values in the column.

min: The minimum value present in the column.

max: The maximum value present in the column.

percentiles: By default, it also calculates the 25th percentile (1st quartile), 50th percentile (median), and 75th percentile (3rd quartile) of the data distribution. You can customize which percentiles are calculated using the percentiles parameter.

these all information will be displayed with respect to dataset.

MORE INFORMATION ABPUT DATA

number of male and female in data

[ ]:
```
df['Gender'].value_counts()
```

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

customers against the rating scale 1 to 5

[ ]:
```
df['Fitness'].value_counts().sort_index()
```

[ ]:
```
1     2
2    26
3    97
4    24
5    31
Name: Fitness, dtype: int64
```

customers with 3 different product types

[ ]:
```
df['Product'].value_counts().sort_index()
```

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

customers counts on Usage

[ ]:
```
df['Usage'].value_counts().sort_index()
```

[ ]:
```
2    33
3    69
4    52
5    17
6     7
7     2
Name: Usage, dtype: int64
```

single and partners customer number

[ ]:
```
df['MaritalStatus'].value_counts()
```

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

From this data we can infer following things :

KP281, KP481, KP781 are the 3 different products

104 males and 76 females are in the customer list.

Highest fitness rating is 3

Most customers use treadmill atleast 3 days per week.

Most customers who has bought it are married / partnered.

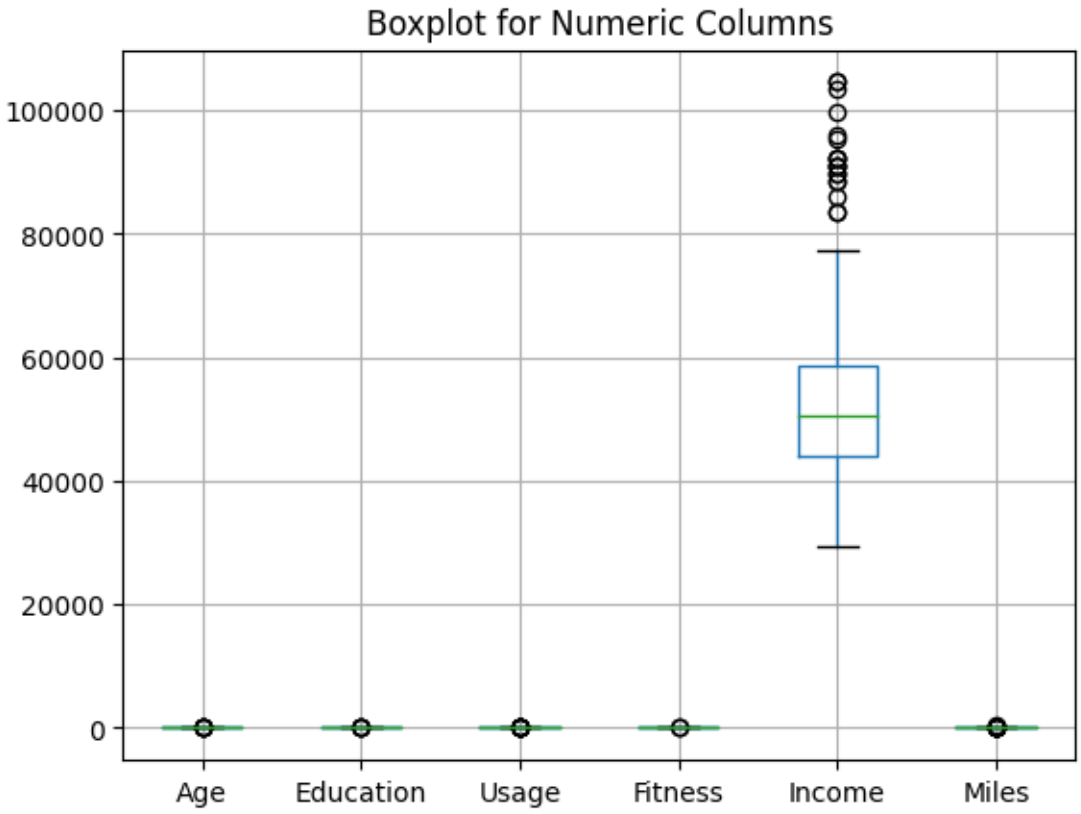
[ ]:
```
df_category = df
df_category['Fitness_category'] = df.Fitness
df_category.head()
```

[ ]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Fitness_category
0	KP281	18	Male	14	Single	3	4	29562	112	4
1	KP281	19	Male	15	Single	2	3	31836	75	3
2	KP281	19	Female	14	Partnered	4	3	30699	66	3
3	KP281	19	Male	12	Single	3	3	32973	85	3
4	KP281	20	Male	13	Partnered	4	2	35247	47	2

[ ]:
```
df.boxplot()
plt.title("Boxplot for Numeric Columns")
```

```
plt.show()
```



this code shows the boxplot graph of the dataset with respect to all the parameters shown in the dataset.

[ ]:
```
summary = df.describe()
summary.loc['mean_median_diff'] = summary.loc['mean'] - summary.loc['50%']
print(summary)
```

```
                       Age   Education       Usage     Fitness  \
count           180.000000  180.000000  180.000000  180.000000
mean             28.788889   15.572222    3.455556    3.311111
std               6.943498    1.617055    1.084797    0.958869
min              18.000000   12.000000    2.000000    1.000000
25%              24.000000   14.000000    3.000000    3.000000
50%              26.000000   16.000000    3.000000    3.000000
75%              33.000000   16.000000    4.000000    4.000000
max              50.000000   21.000000    7.000000    5.000000
mean_median_diff  2.788889   -0.427778    0.455556    0.311111

                      Income       Miles
count             180.000000  180.000000
```

```
mean               53719.577778  103.194444
std                16506.684226   51.863605
min                29562.000000   21.000000
25%                44058.750000   66.000000
50%                50596.500000   94.000000
75%                58668.000000  114.750000
max               104581.000000  360.000000
mean_median_diff    3123.077778    9.194444
```

Mean age of dataset is 28.78

Minimum Age: 18 and maximum age : 15

25% of the customers age is 24

75% of the customer age is 33

Average usage per week bya person is 3 day.

Average Fitness rating is 3

Average Income of a customer is 54k per year.

Highest salary of a customer is 104k.

Maximum distance covered by a customer is 360 miles.

Maximum Education qualification is 21, with most frequent education as 16

Around 25% of person's covered an average of 66 miles,

these lines shows the mean and median difference in the dataset.

[ ]:
```
sr = df['Product'].value_counts(normalize=True)
stats = sr.map(lambda calc: round(100*calc,2))
stats
```

[ ]:
```
KP281    44.44
KP481    33.33
KP781    22.22
Name: Product, dtype: float64
```

44.44 customer bought KP281 treadmill.

33.33 customer bought KP481 treadmill.

22.22 customer bought KP781 treadmill.

[ ]:
```
gender = df['Gender'].value_counts(normalize=True)
gender_reso = gender.map(lambda calc: round(100*calc,2))
gender_reso
```

[ ]:
```
Male      57.78
Female    42.22
Name: Gender, dtype: float64
```

57.78% of customers are Male and 42.22% customers are Female

[ ]:
```
married_Status = df['MaritalStatus'].value_counts(normalize=True)
married_Status_resolve = married_Status.map(lambda calc:round(100*calc,2))
married_Status_resolve
```

[ ]:
```
Partnered    59.44
Single       40.56
Name: MaritalStatus, dtype: float64
```

59.44% of customers are Married. 40.56% of customers are Single

[ ]:
```
use = df['Usage'].value_counts(normalize=True).map(lambda calc:
 ↪round(100*calc,2)).reset_index()
use.rename(columns={'index':'DaysPerWeek'},inplace=True)
use
```

[ ]:

	DaysPerWeek	Usage
0	3	38.33
1	4	28.89
2	2	18.33
3	5	9.44
4	6	3.89
5	7	1.11

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517d7b8790>
```

```
from matplotlib import pyplot as plt
use['DaysPerWeek'].plot(kind='hist', bins=20, title='DaysPerWeek')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
use['Usage'].plot(kind='hist', bins=20, title='Usage')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517d60e0e0>
```

```
from matplotlib import pyplot as plt
use.plot(kind='scatter', x='DaysPerWeek', y='Usage', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517d7b95d0>
```

```
from matplotlib import pyplot as plt
use['DaysPerWeek'].plot(kind='line', figsize=(8, 4), title='DaysPerWeek')
plt.gca().spines[['top', 'right']].set_visible(False)
```

```
from matplotlib import pyplot as plt
use['Usage'].plot(kind='line', figsize=(8, 4), title='Usage')
plt.gca().spines[['top', 'right']].set_visible(False)
```

39% customers use it for 3 days per week

2% customers us it for 27 days a week.

[ ]:
```
rating = df['Fitness'].value_counts(normalize=True).map(lambda calc:
  ↪round(100*calc,2)).reset_index()
rating.rename(columns={'index':'Rating'},inplace=True)
rating
```

[ ]:

	Rating	Fitness
0	3	53.89
1	5	17.22
2	2	14.44
3	4	13.33
4	1	1.11

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517da34070>
```

```
from matplotlib import pyplot as plt
rating['Rating'].plot(kind='hist', bins=20, title='Rating')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
from matplotlib import pyplot as plt
rating['Fitness'].plot(kind='hist', bins=20, title='Fitness')
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517da35ba0>
```

```
from matplotlib import pyplot as plt
rating.plot(kind='scatter', x='Rating', y='Fitness', s=32, alpha=.8)
plt.gca().spines[['top', 'right',]].set_visible(False)
```

```
<google.colab._quickchart_helpers.SectionTitle at 0x7e517da36770>
```

```
from matplotlib import pyplot as plt
rating['Rating'].plot(kind='line', figsize=(8, 4), title='Rating')
plt.gca().spines[['top', 'right']].set_visible(False)
```

```
from matplotlib import pyplot as plt
rating['Fitness'].plot(kind='line', figsize=(8, 4), title='Fitness')
plt.gca().spines[['top', 'right']].set_visible(False)
```

53 % of customers has rated themselves as 3.

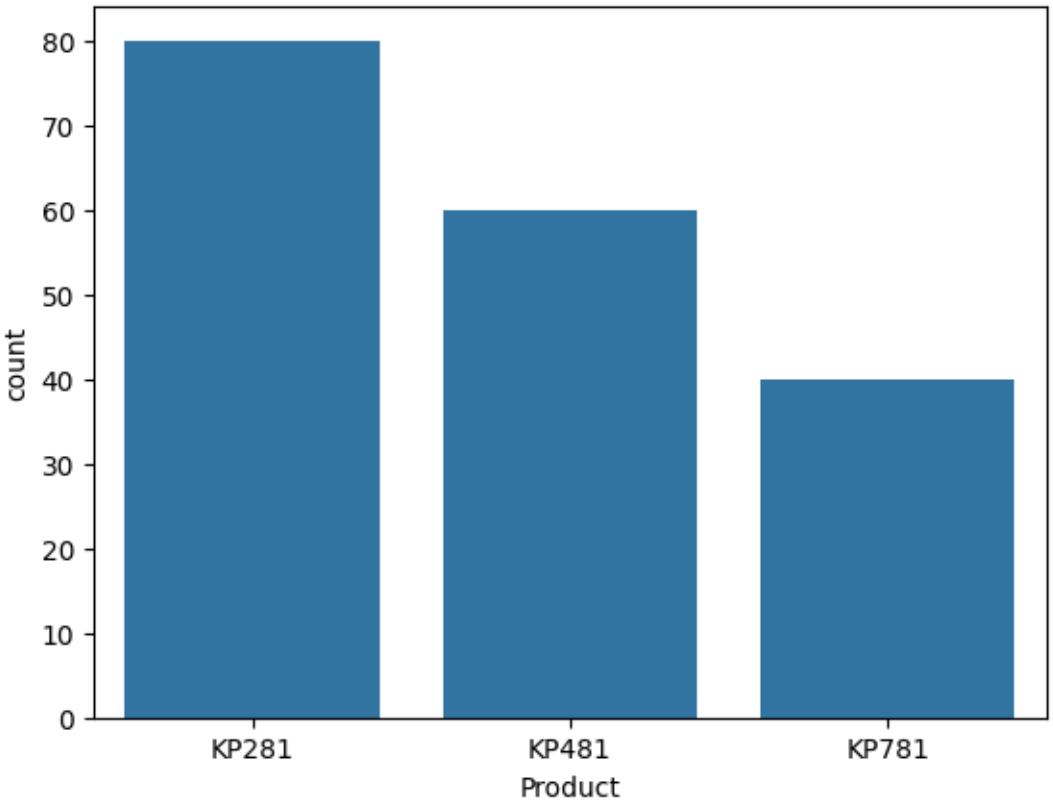
14 % of customers has rated themselves as average.

17 % of customers has rated themselves as good fitness.

VISUAL ANAYSIS

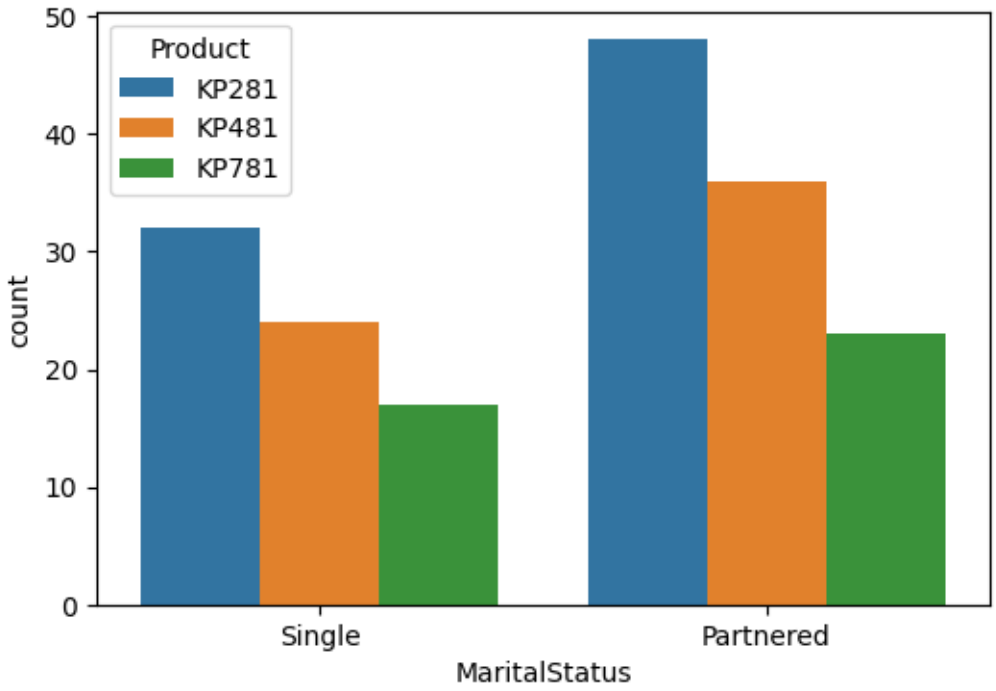
[ ]:
```
sns.countplot(data=df,x='Product')
plt.show
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



KP281 is most commonly used. KP481 is second most commonly used. KP781 is least commonly used.

[ ]:
```
plt.figure(figsize=(6, 4))
sns.countplot(data=df,x='MaritalStatus', hue='Product', dodge='False')
plt.show()
```

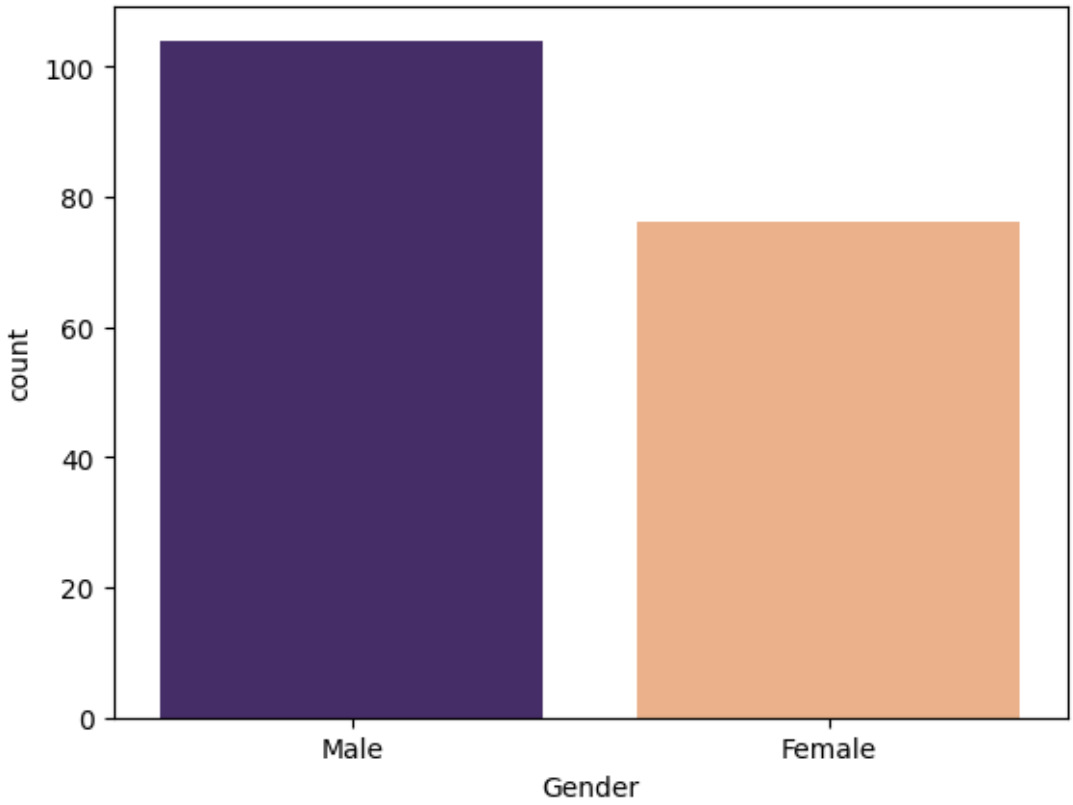


this graph shows the the number of type of treadmill which single and partnered users buy.

[ ]:
```
sns.countplot(data=df,x='Gender',palette=['#432371',"#FAAE7B"])
plt.show
```

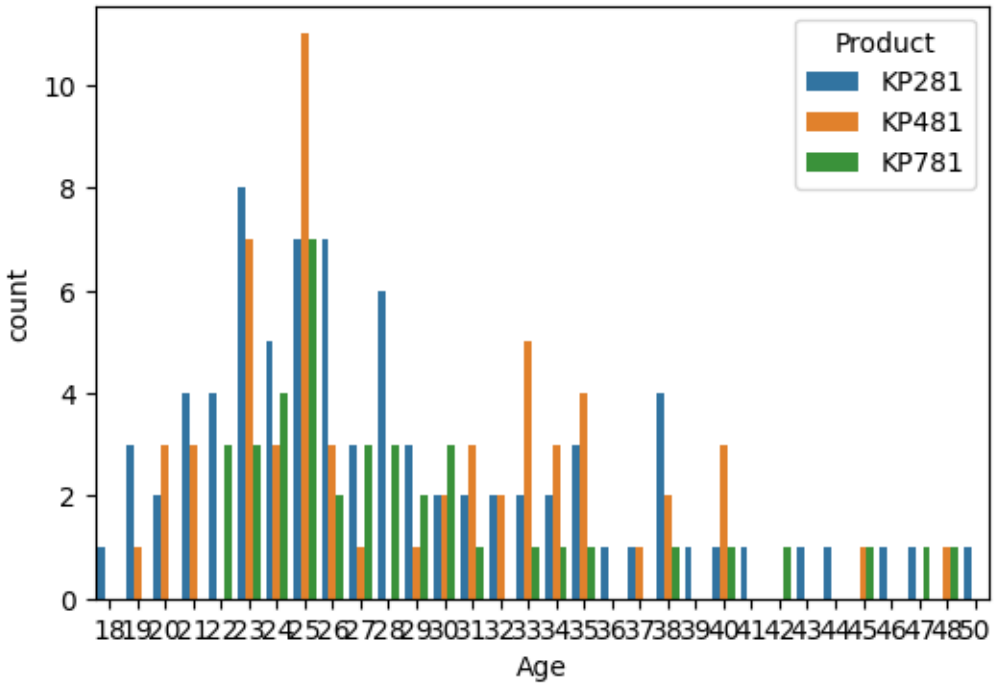
```
<ipython-input-37-67c237829c26>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(data=df,x='Gender',palette=['#432371',"#FAAE7B"])
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Males are more product than females.

[ ]:
```
plt.figure(figsize=(6, 4))
sns.countplot(data=df,x='Age', hue='Product', dodge='False')
plt.show()
```

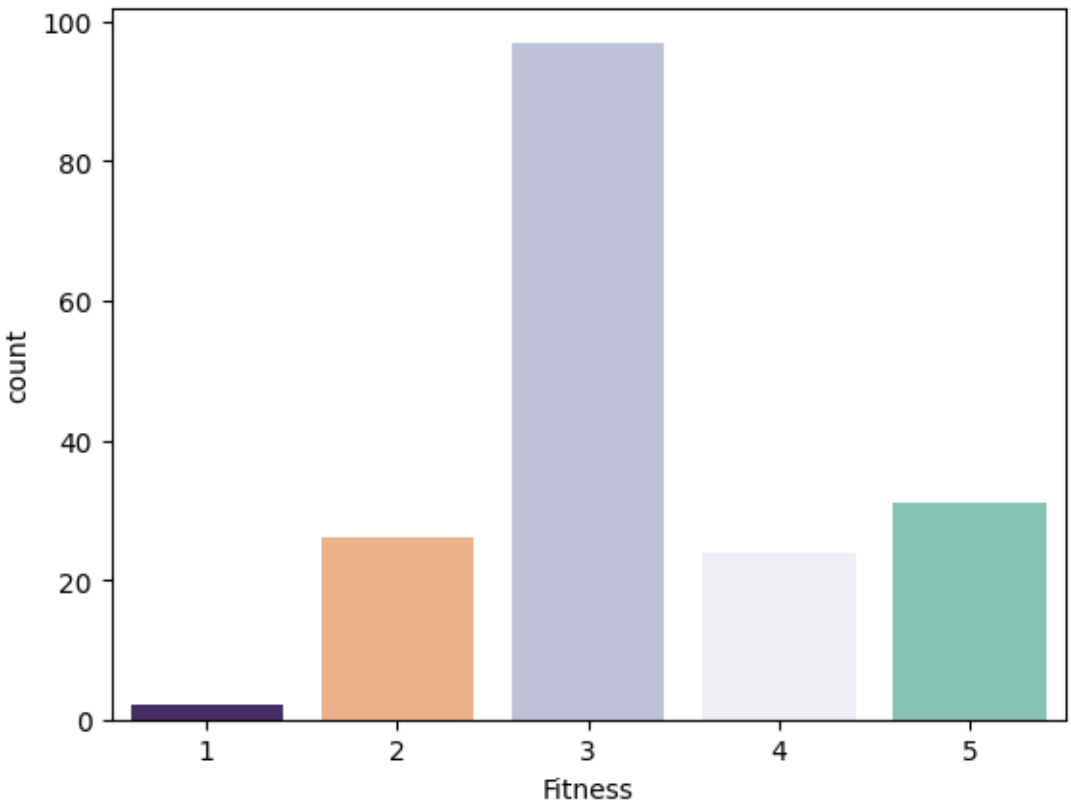


this graph shows the variation of buying of different types of treadmill with respect to age group.

[ ]:
```
sns.countplot(data=df,x='Fitness',palette=['#432371',"#FAAE7B","#bcbddc",␣
↪"#efedf5",'#7fcdbb'])
plt.show
```

```
<ipython-input-38-7009b14c6cdd>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(data=df,x='Fitness',palette=['#432371',"#FAAE7B","#bcbddc",
"#efedf5",'#7fcdbb'])
```

[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

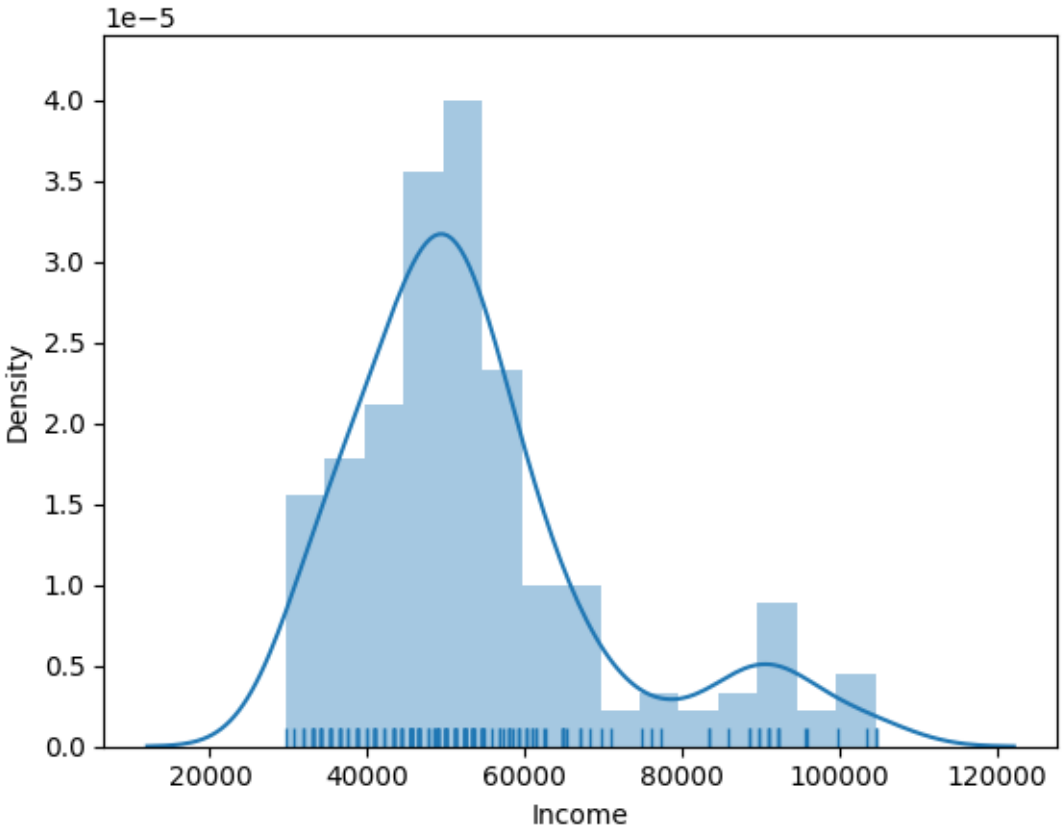


more than 90% people has rated themselves as average

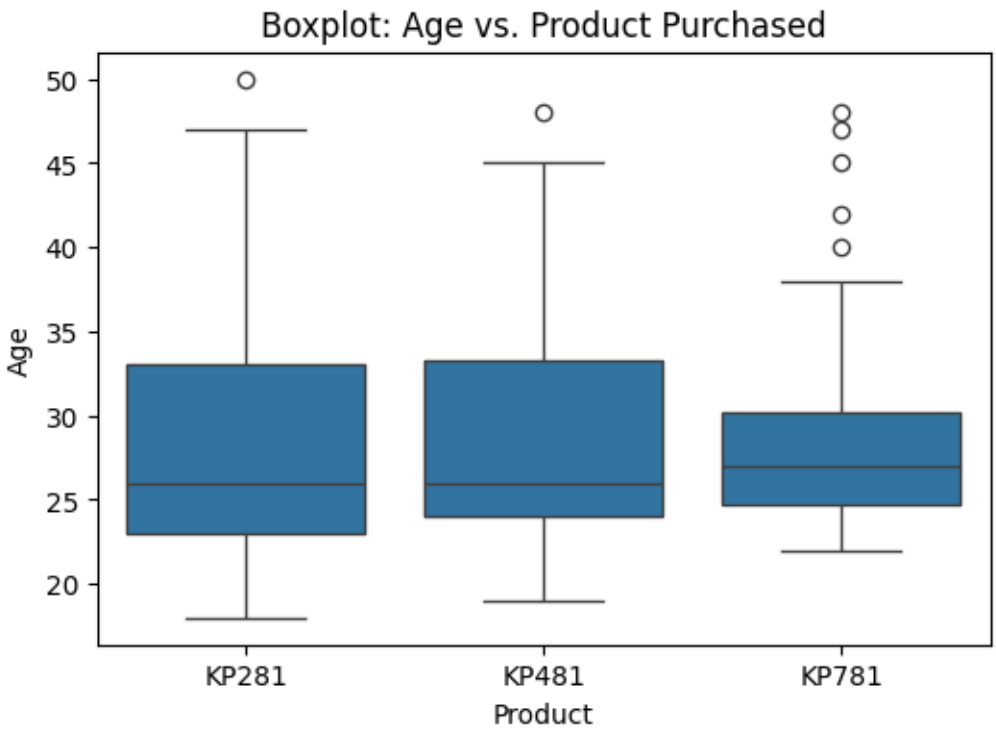
Excelklent shape is seond highest provided by the customers.

[ ]:
```
sns.distplot(df.Income,rug=True)
plt.show()
```

```
<ipython-input-39-750d3bf61763>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.Income,rug=True)
```

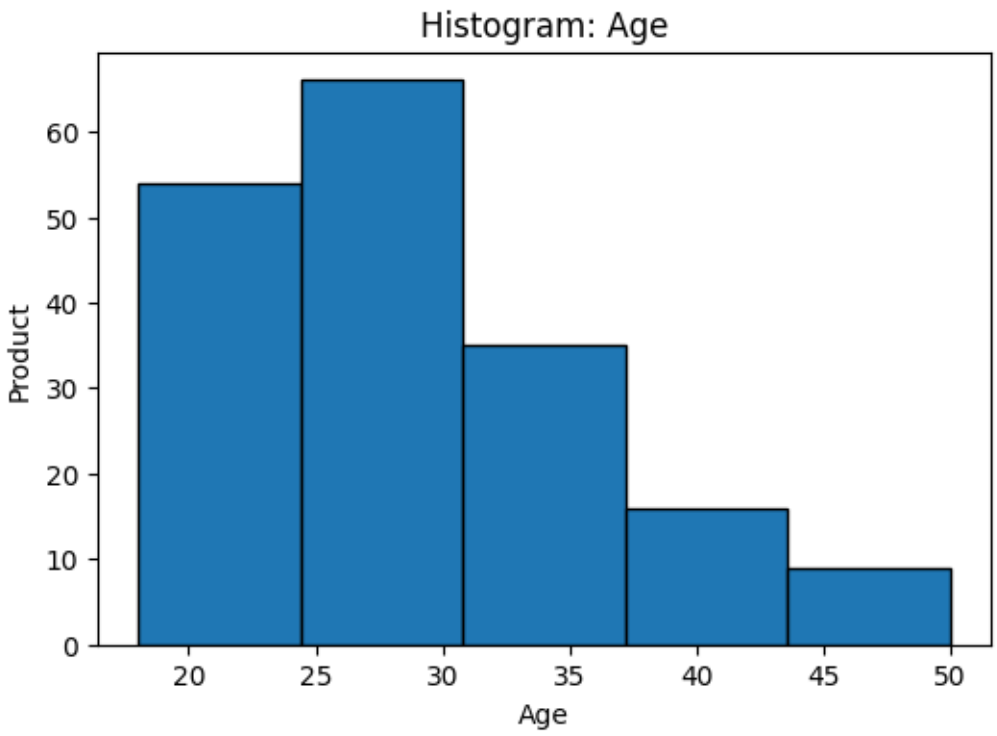


[ ]:
```
plt.figure(figsize=(6, 4))
plt.title("Boxplot: Age vs. Product Purchased")
sns.boxplot(x='Product', y='Age', data=df)
plt.show()
```



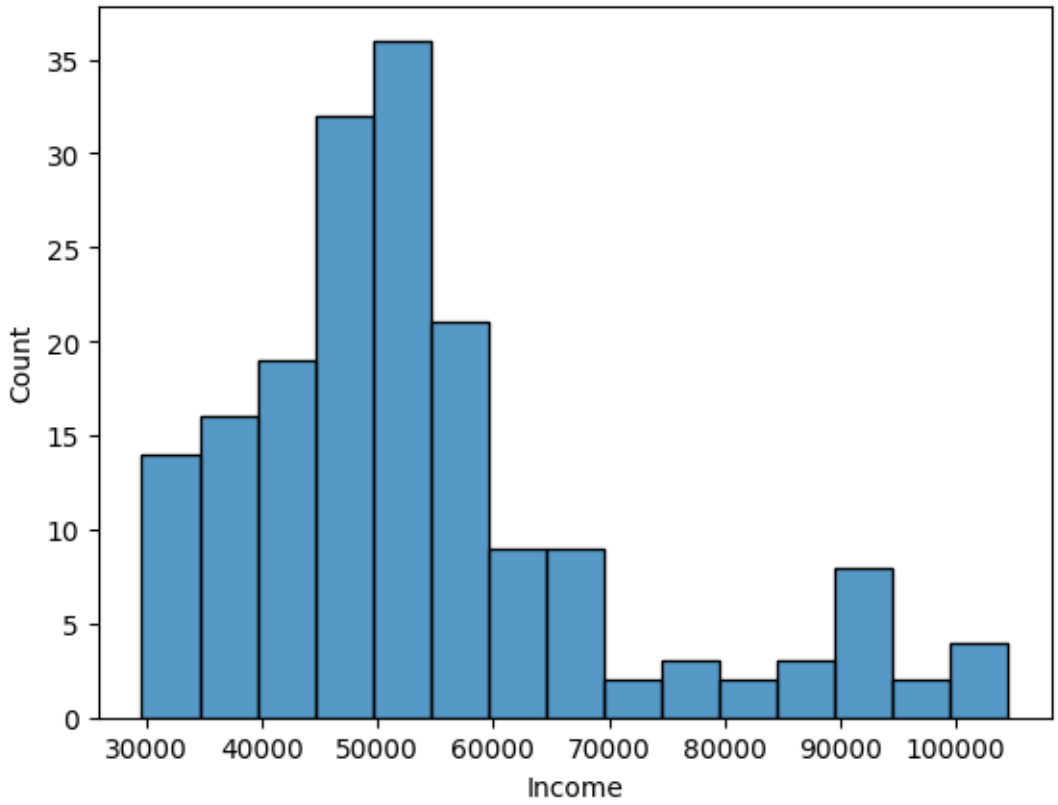
this graph shows the variation of buying of different types of treadmill with respect to age group.

[ ]:
```
plt.figure(figsize=(6, 4))
plt.title("Histogram: Age")
plt.hist(df['Age'], bins=5, edgecolor='black')
plt.xlabel("Age")
plt.ylabel("Product")
plt.show()
```



[ ]:
```
sns.histplot(data=df,x='Income')
```

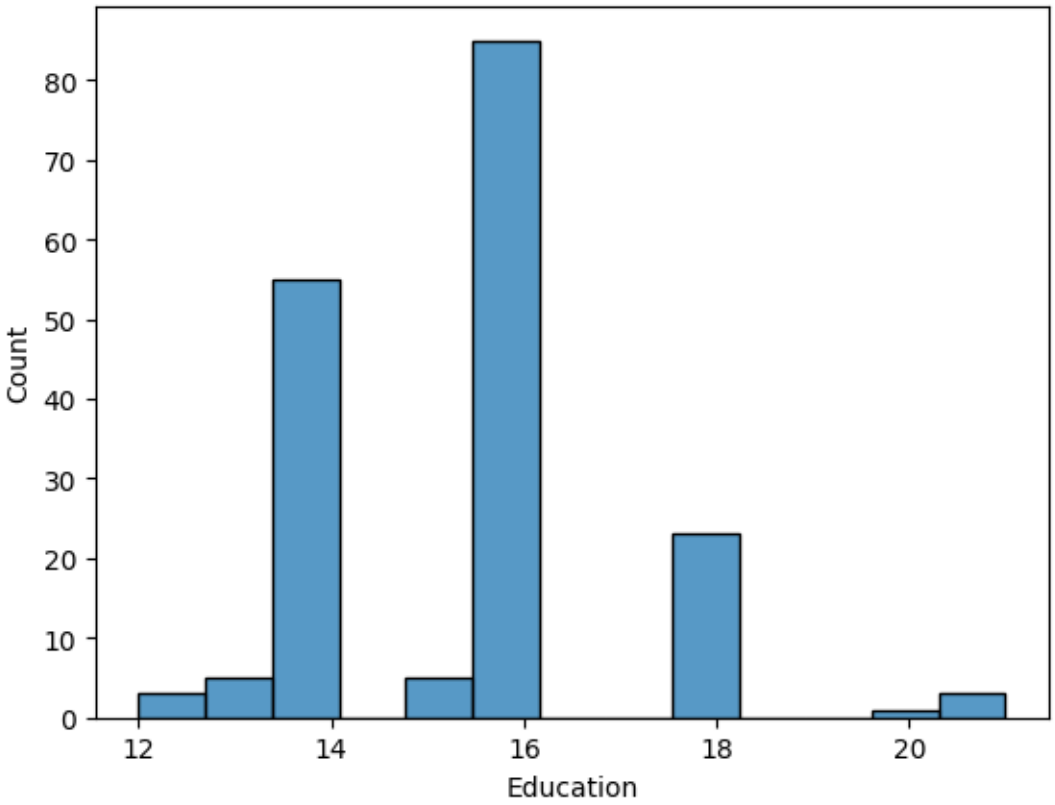
[ ]:
```
<Axes: xlabel='Income', ylabel='Count'>
```



more than 35% customers earn 50-55k more than 30% customers earn 45-50k more than 20% customers earn 55-60k

[ ]:
```
sns.histplot(data=df,x='Education')
```

[ ]:
```
<Axes: xlabel='Education', ylabel='Count'>
```



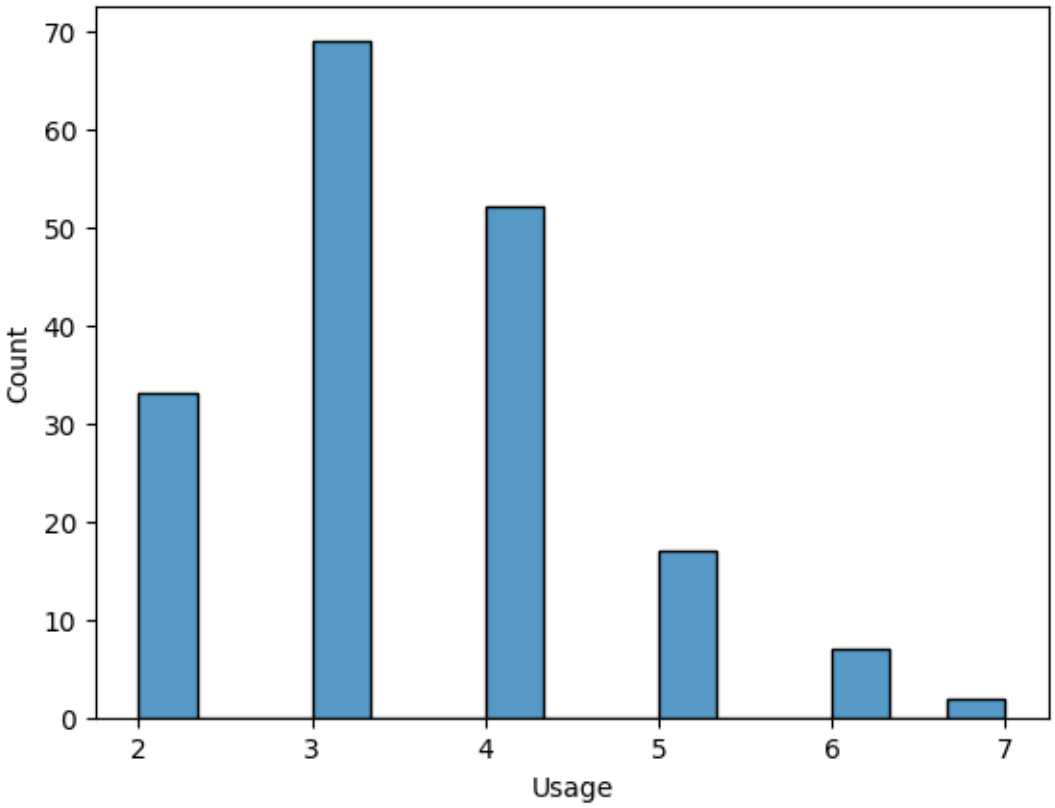
16 is the most common education here

14 is the highest common education here

20 is the leat common education here

[ ]:
```
sns.histplot(data=df,x='Usage')
```

[ ]:
```
<Axes: xlabel='Usage', ylabel='Count'>
```



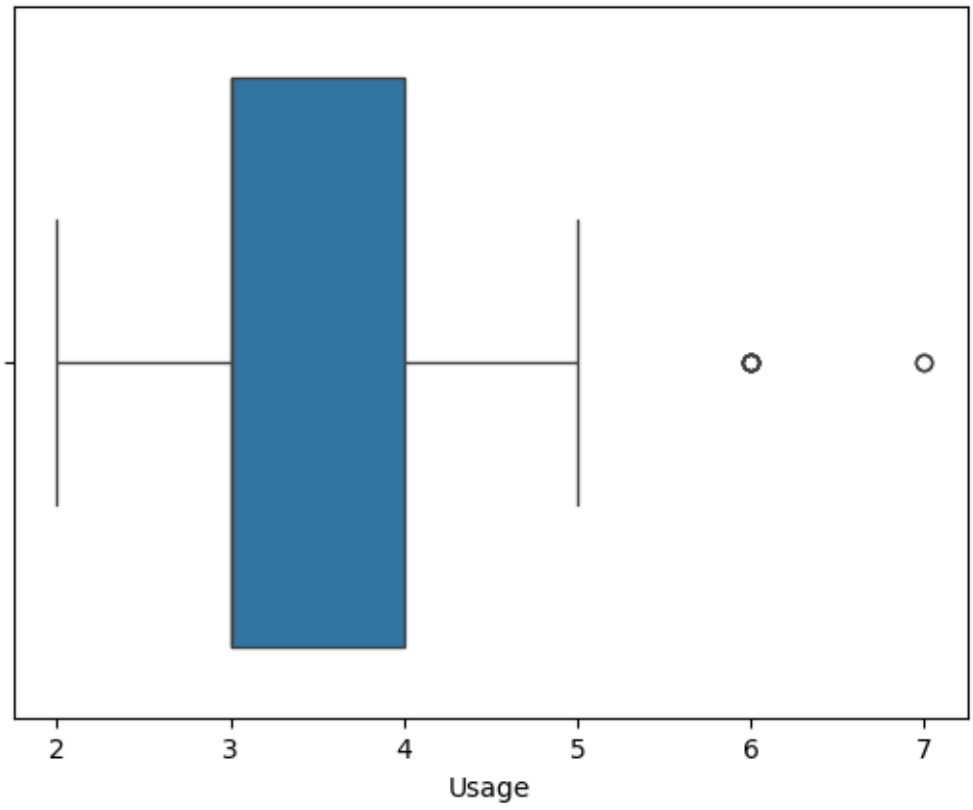
most common usage : 3 day

second: 4 day

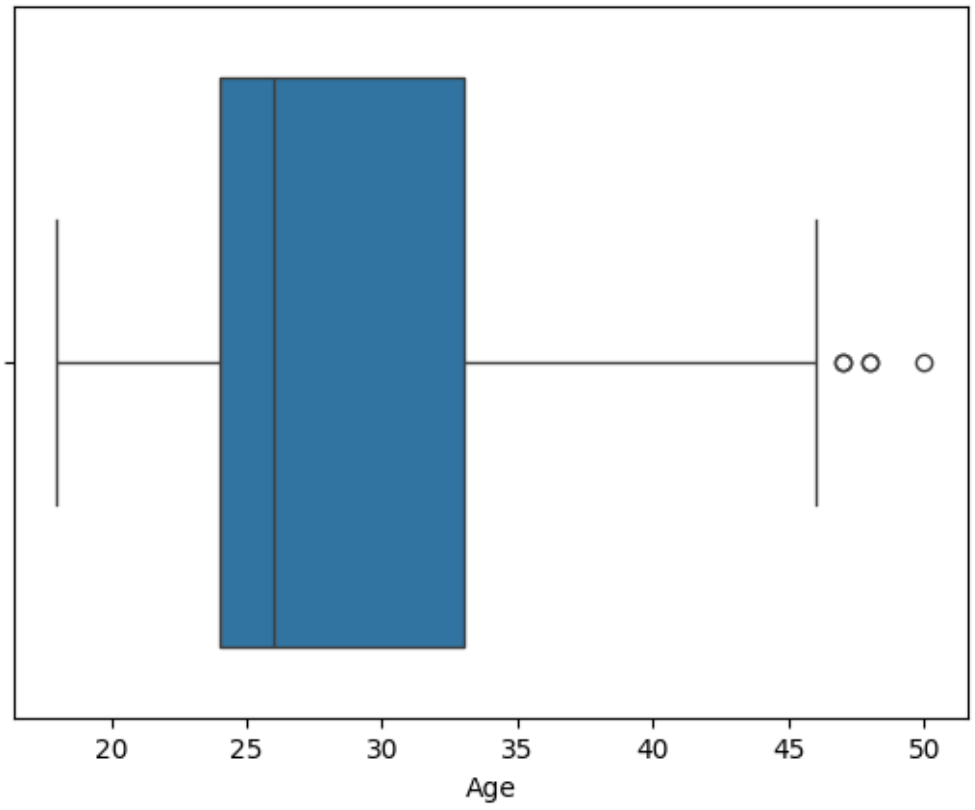
third : 2 day

very few : 7 days

[ ]:
```
sns.boxplot(data=df,x='Usage')
plt.show()
```



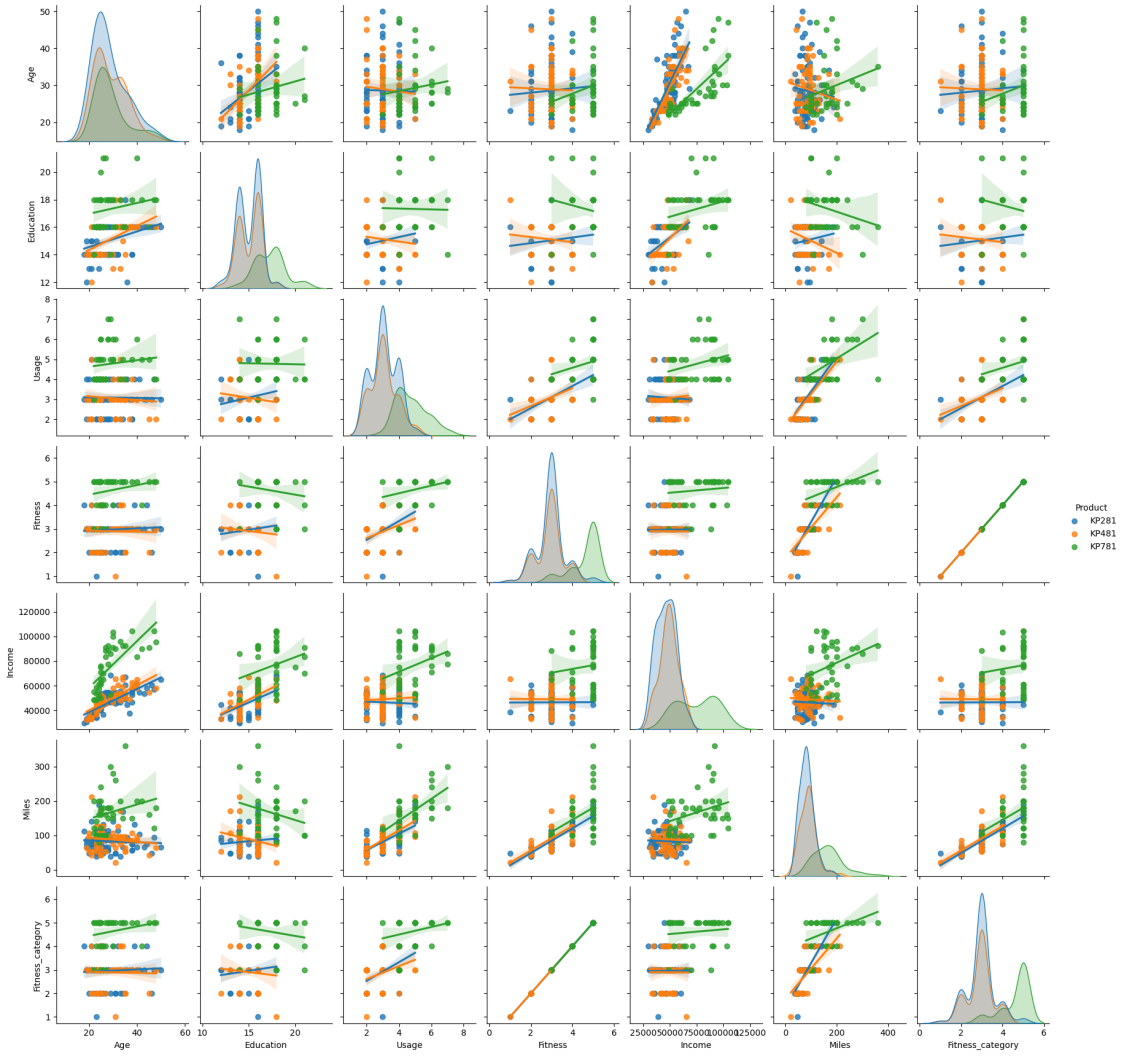
[ ]:
```
sns.boxplot(data=df,x='Age')
plt.show()
```



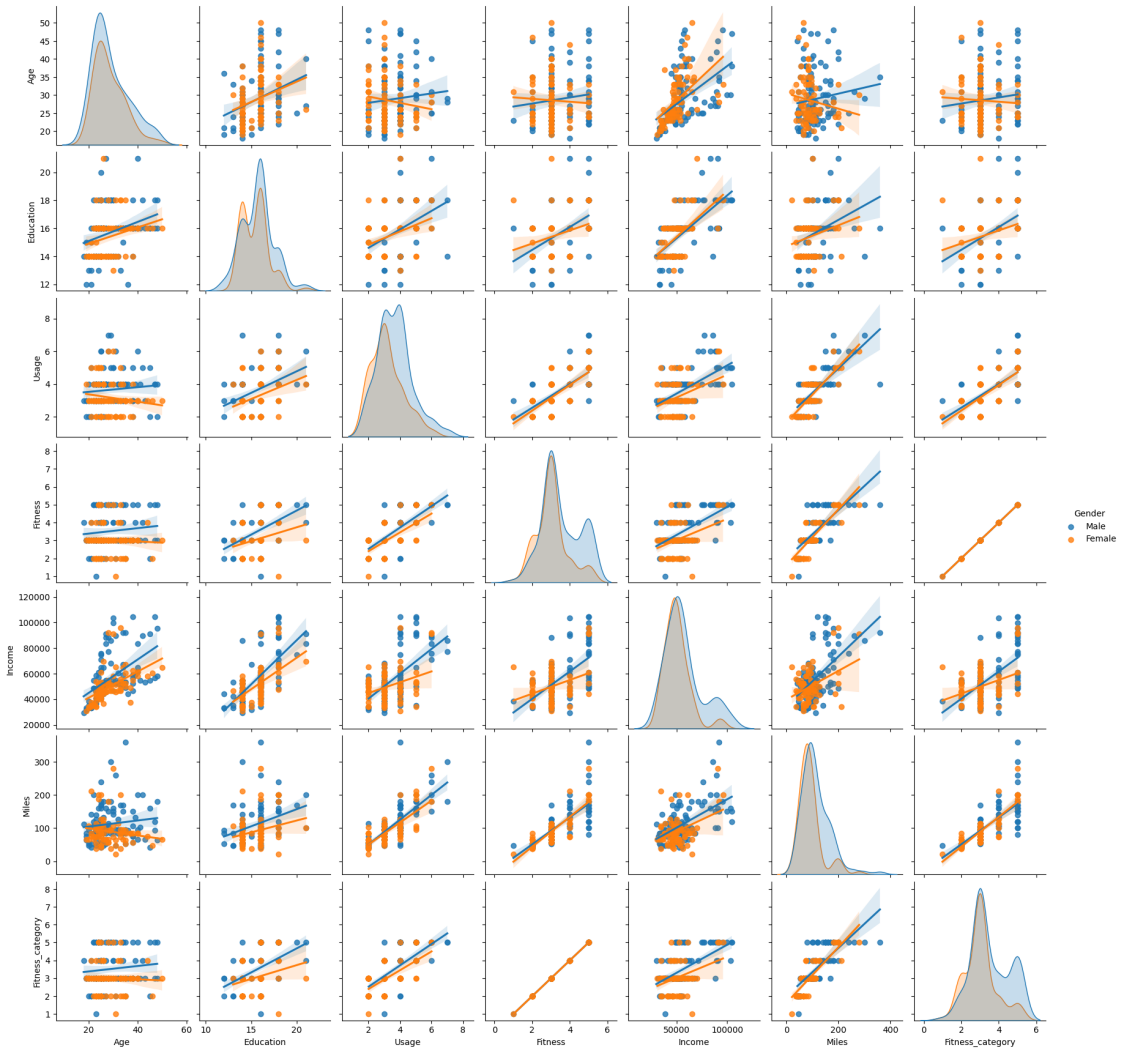
most common age when customers are purchasing the product : 23 to 34

very few customers above age 45 are purchsing the product

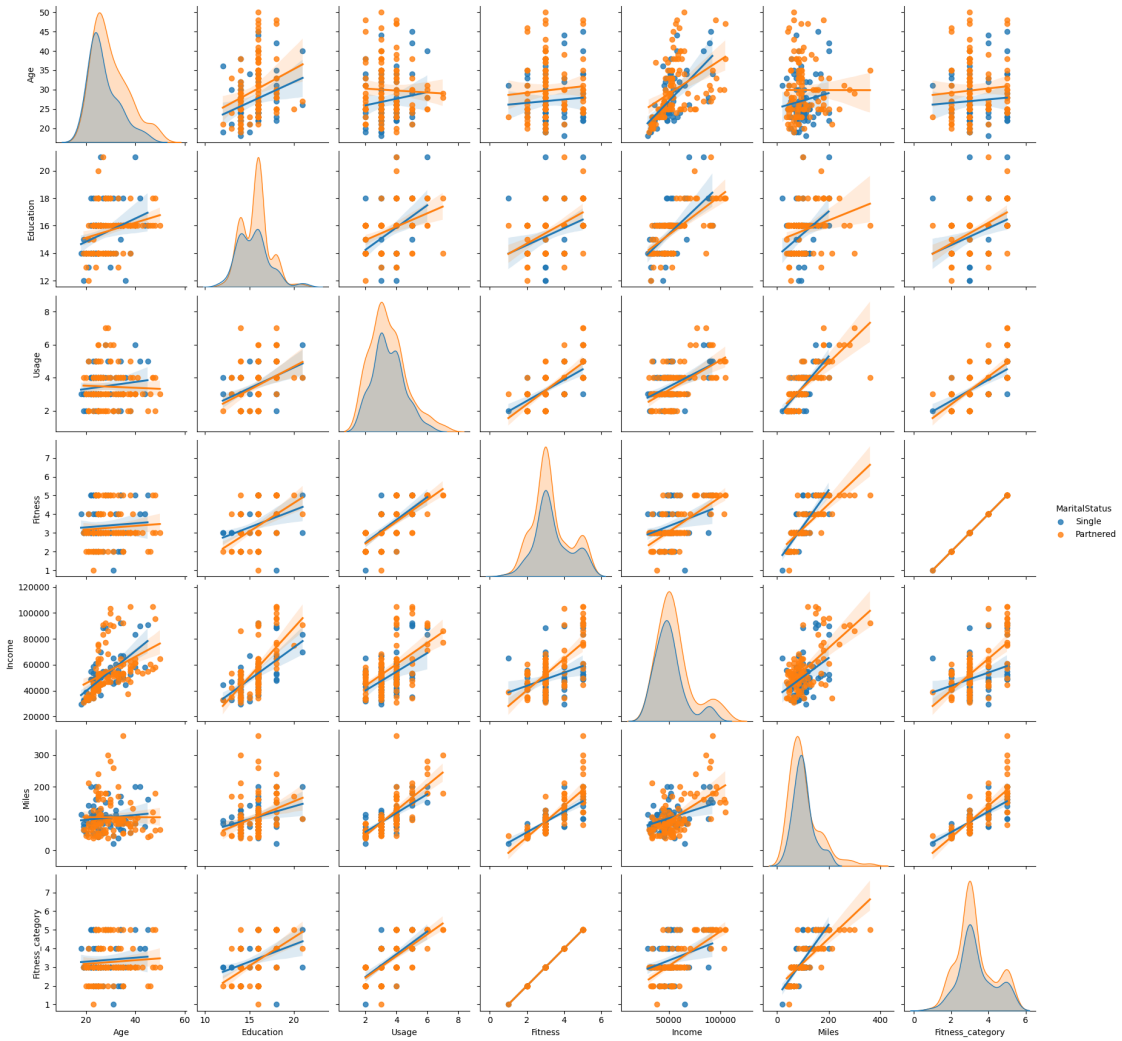
[ ]:
```
sns.pairplot(df,hue='Product',kind='reg')
plt.show()
```



[ ]:
```
sns.pairplot(df,hue='Gender',kind='reg')
plt.show()
```



[ ]:
```
sns.pairplot(df,hue='MaritalStatus',kind='reg')
plt.show()
```



BIVARIATE ANALYSIS

[ ]:
```
df.groupby('Product')['Age'].mean()
```

[ ]:
```
Product
KP281    28.55
KP481    28.90
KP781    29.10
Name: Age, dtype: float64
```

MEAN AGE :

purchasing KP281 : 28.55

purchasing KP281 : 28.90

purchasing KP281 : 29.10

[ ]:
```
df.groupby('Product')['Usage'].mean()
```

[ ]:
```
Product
KP281    3.087500
KP481    3.066667
KP781    4.775000
Name: Usage, dtype: float64
```

MEAN USGAE OF PRODUCT

KP281 3.08

KP481 3.06

KP781 4.77

[ ]:
```
df.groupby('Product')['Education'].mean()
```

[ ]:
```
Product
KP281    15.037500
KP481    15.116667
KP781    17.325000
Name: Education, dtype: float64
```

MEAN EDUCATION QUALIFICATION OF CUSTOMER WHO PURCHSED THIS PRODUCT

KP281 15.03

KP481 15.11

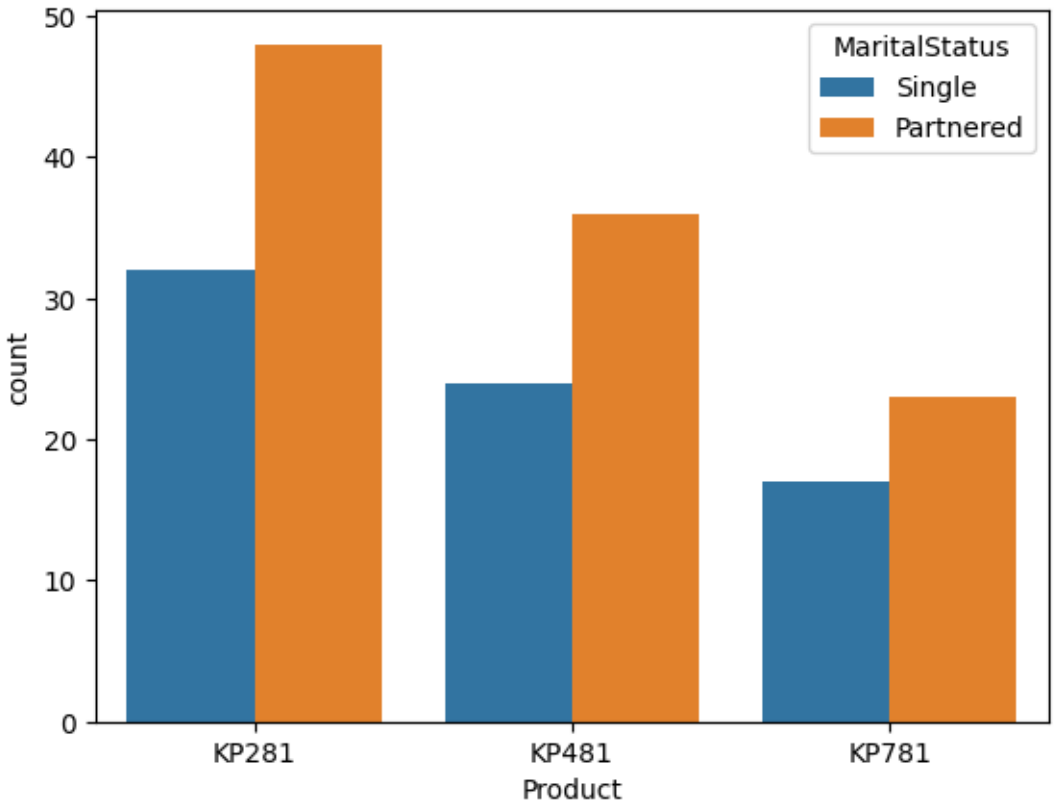
KP781 17.32

[ ]:
```
df.groupby('Product')['Fitness'].mean()
```

[ ]:
```
Product
KP281    2.9625
KP481    2.9000
KP781    4.6250
Name: Fitness, dtype: float64
```

Customer fitness mean KP281 2.96 KP481 2.90 KP781 4.62

[ ]:
```
sns.countplot(data=df,x='Product',hue='MaritalStatus')
plt.show()
```

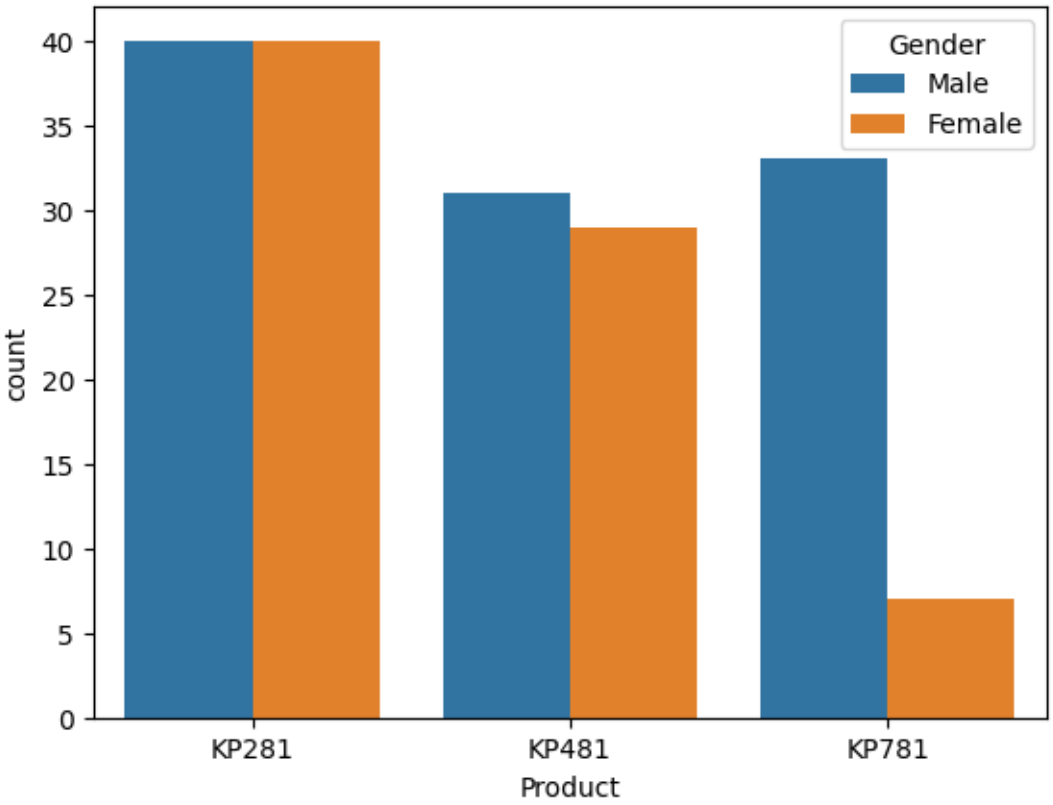


KP281 is most preferred.

KP481 is second most preferred.

Partnered customers are the major product purchasers

[ ]:
```
sns.countplot(data=df,x='Product',hue='Gender')
plt.show()
```

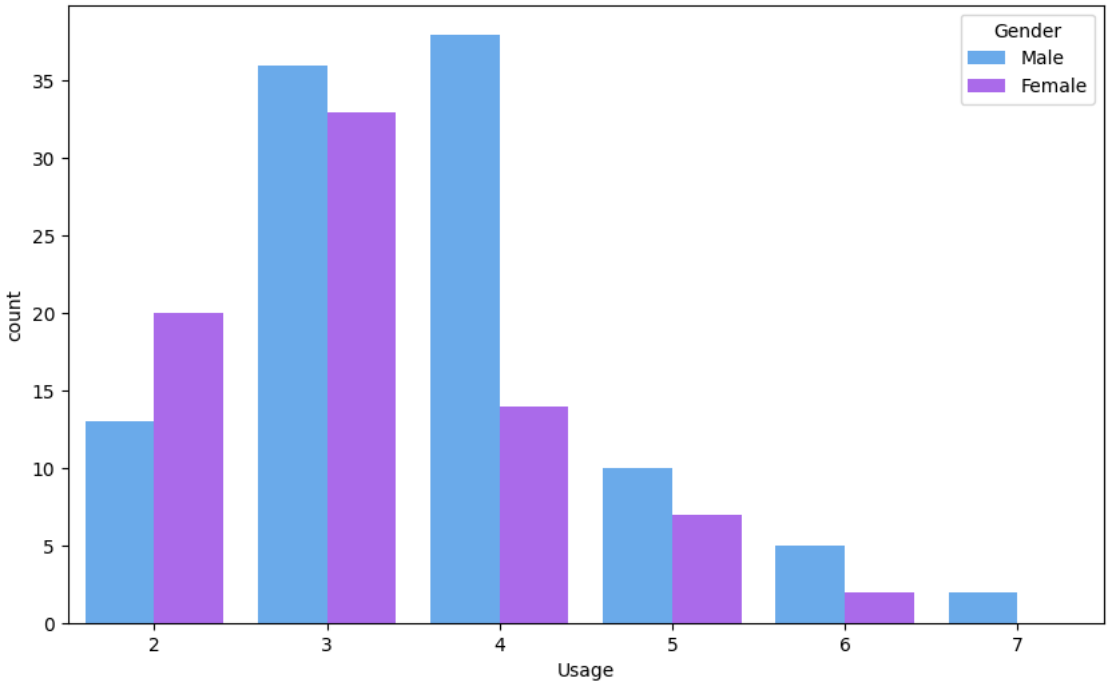


KP281 IS PREFERRD BY BOTH MALE AND FEMALE

KP481 IS MORE PREFFERED BY MALE

KP781 IS AGAIN MOSTLY PREFERRED BY MALES THAM FEMALES.

[ ]:
```
plt.figure(figsize=(10,6))
sns.countplot(data=df,x='Usage',hue='Gender',palette='cool')
plt.show()
```

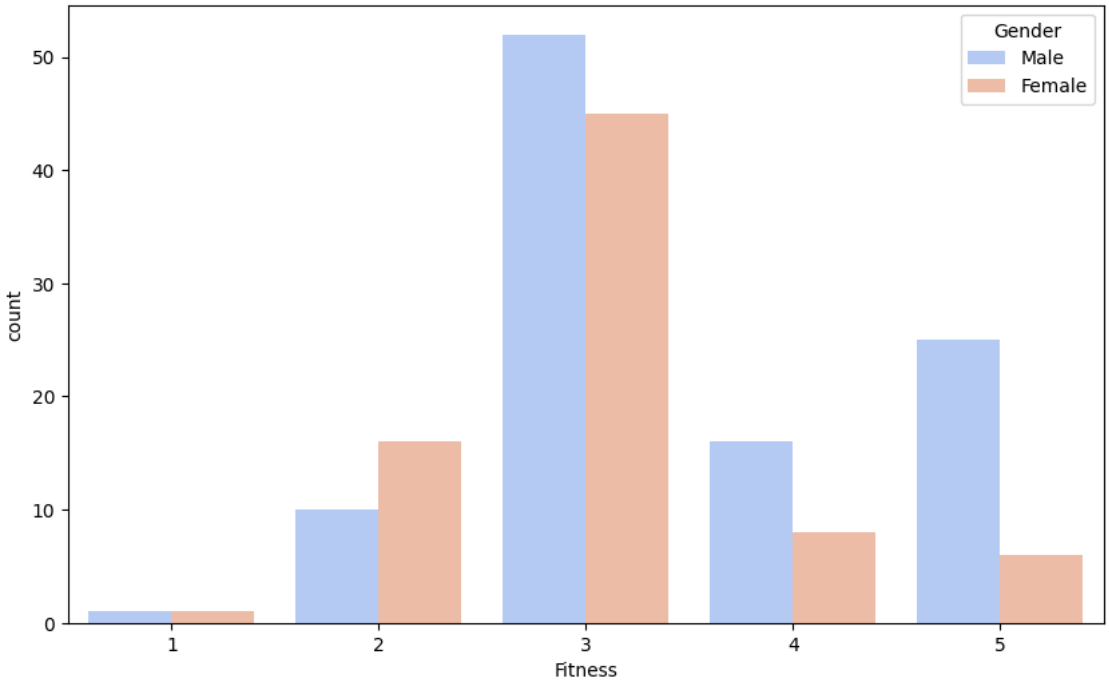


male usgae is 4 day per week

female uses it 3 days per week

few males use it 7 days per week,females maximum usage is 6 days per week.

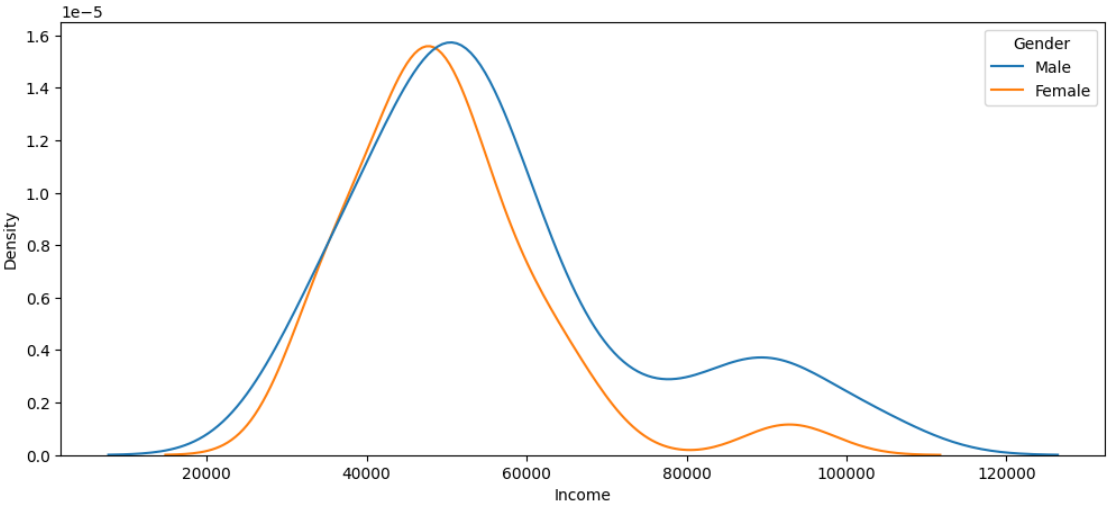
[ ]:
```
plt.figure(figsize=(10,6))
sns.countplot(data=df,x='Fitness',hue='Gender',palette='coolwarm')
plt.show()
```



Among the fitness rating both Male and Female most have rated as average

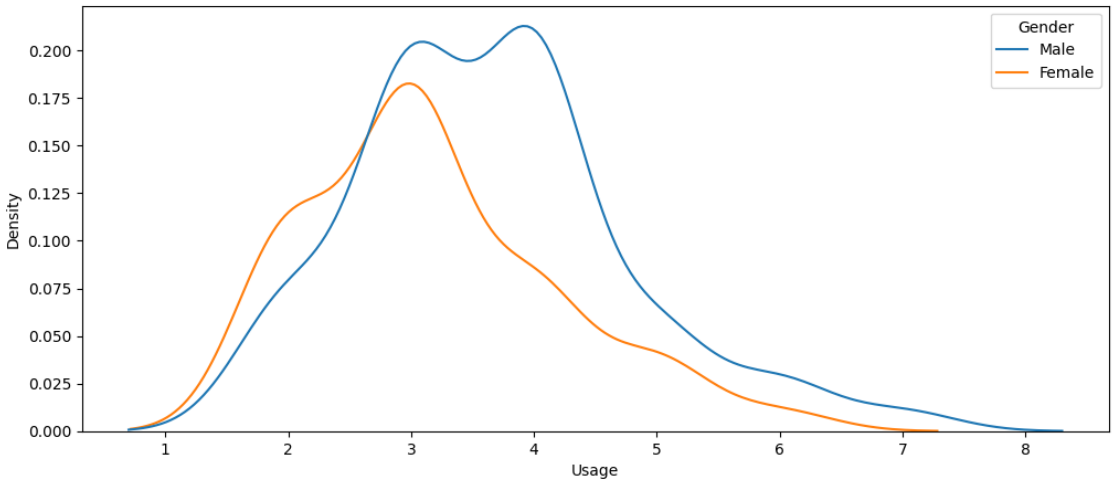
Significant number of Male customers are at Excellent shape compared to Female customers

[ ]:
```
plt.figure(figsize=(12,5))
sns.kdeplot(data=df,x='Income',hue='Gender')
plt.show()
```



we can conclude the spike from 40K to around 80K is the most common income per annum of the customers

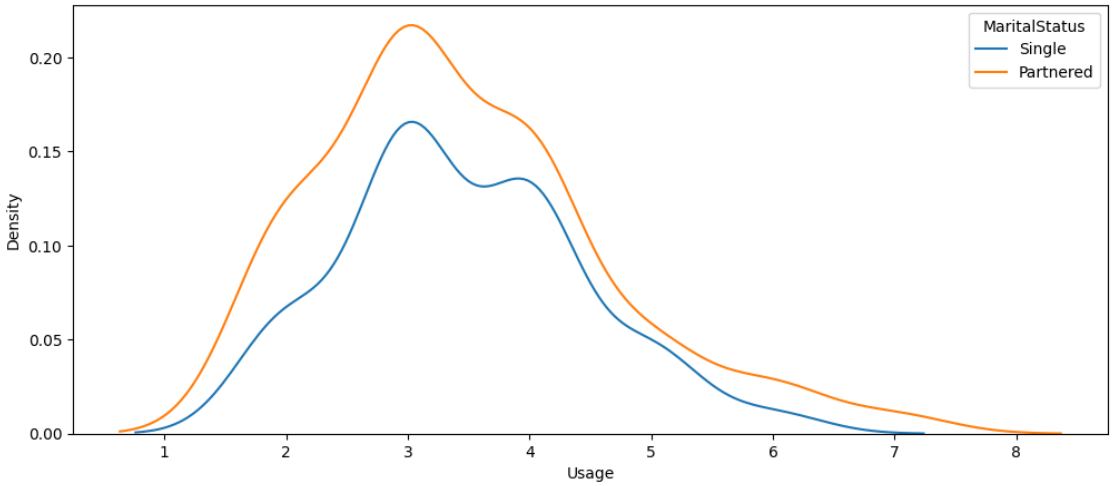
[ ]:
```
plt.figure(figsize=(12,5))
sns.kdeplot(data=df,x='Usage',hue='Gender')
plt.show()
```



male customer usage is significantly more than the female customers.

Female customer loses consistently after 3 days.

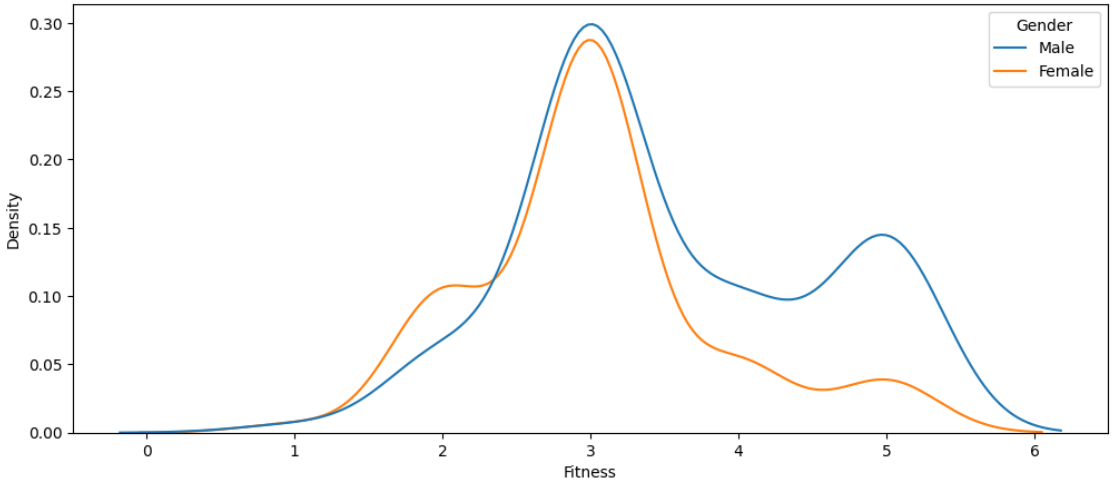
[ ]:
```
plt.figure(figsize=(12,5))
sns.kdeplot(data=df,x='Usage',hue='MaritalStatus')
plt.show()
```



Partnered customers usage is higher than single customers

Partnered customers also have greater consistency per week of 7 days per week than single customers

[ ]:
```
plt.figure(figsize=(12,5))
sns.kdeplot(data=df,x='Fitness',hue='Gender')
plt.show()
```

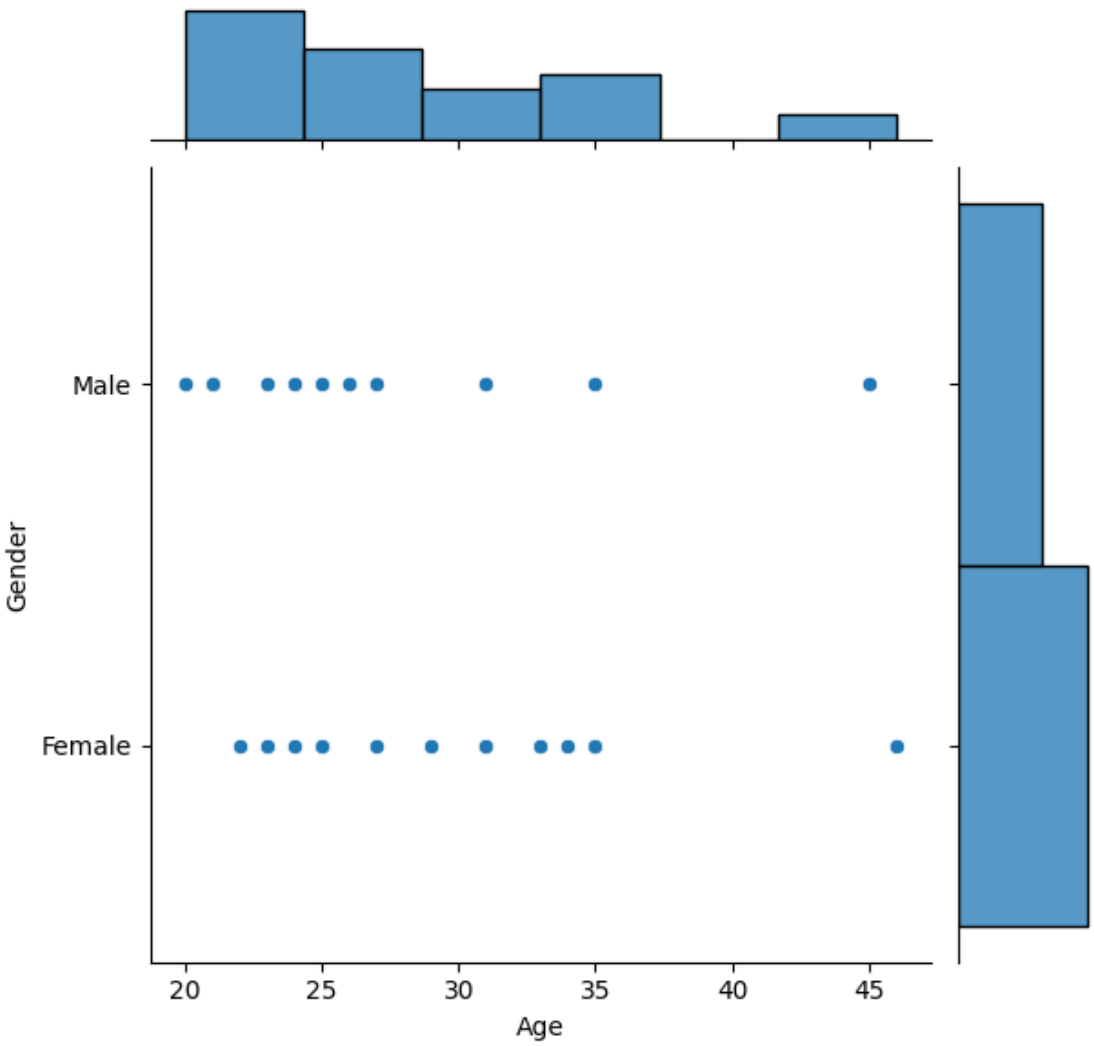


Male customers are in better shape the female customers

Though Female customers do not have poor shape, they are also not in excellent shape

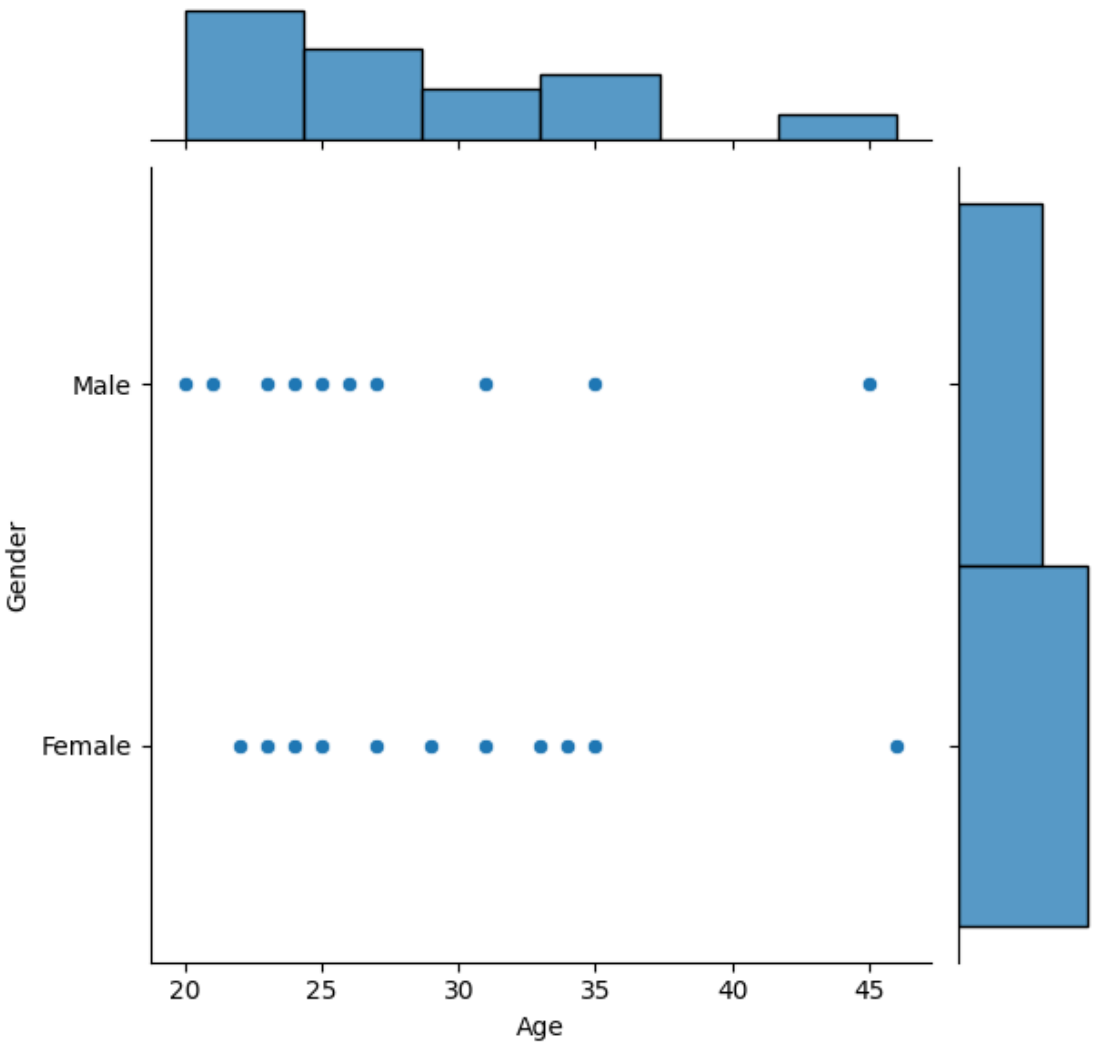
Some Male customers excellent body shape and few customers have poor shape as well

[ ]:
```
sns.jointplot(x='Age',y='Gender',data=df[df.Fitness<3])
plt.show()
```



Product is not familiar with womens.

[ ]:
```
sns.jointplot(x='Age',y='Gender',data=df[df.Fitness<3])
plt.show()
```



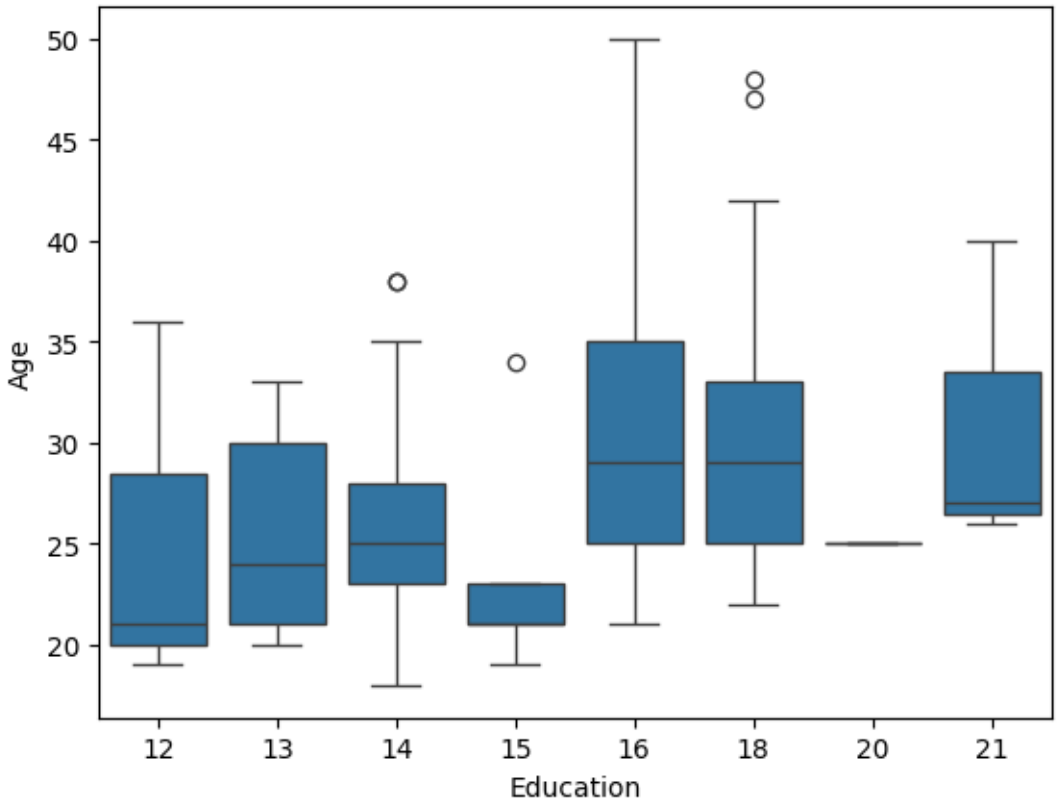
above plot shows the salary distribution of male and female.

avaerage fitness level is 3 to 4.

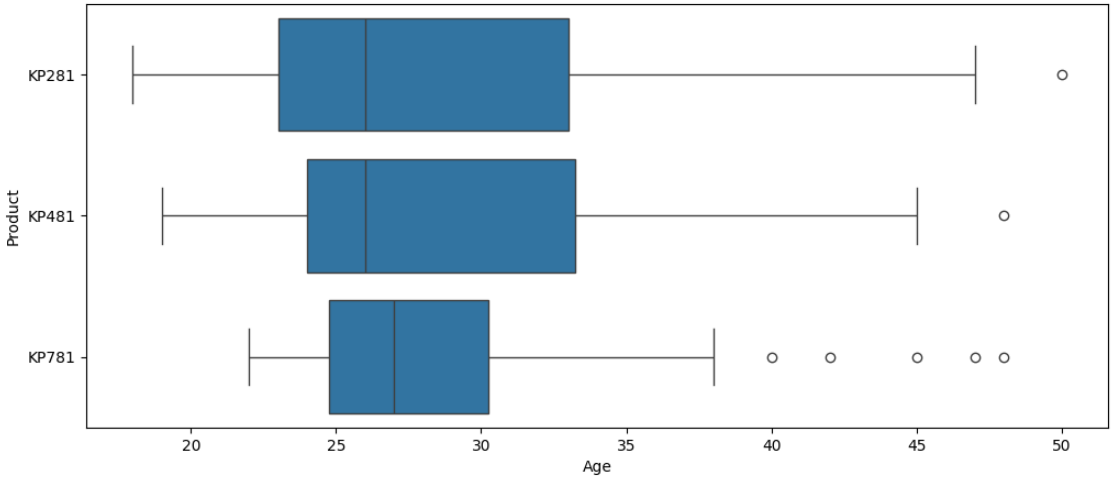
ther is very few custometrs who earn a lot and run miles.

[ ]:
```
sns.boxplot(x='Education',y='Age',data=df)
```

[ ]:
```
<Axes: xlabel='Education', ylabel='Age'>
```



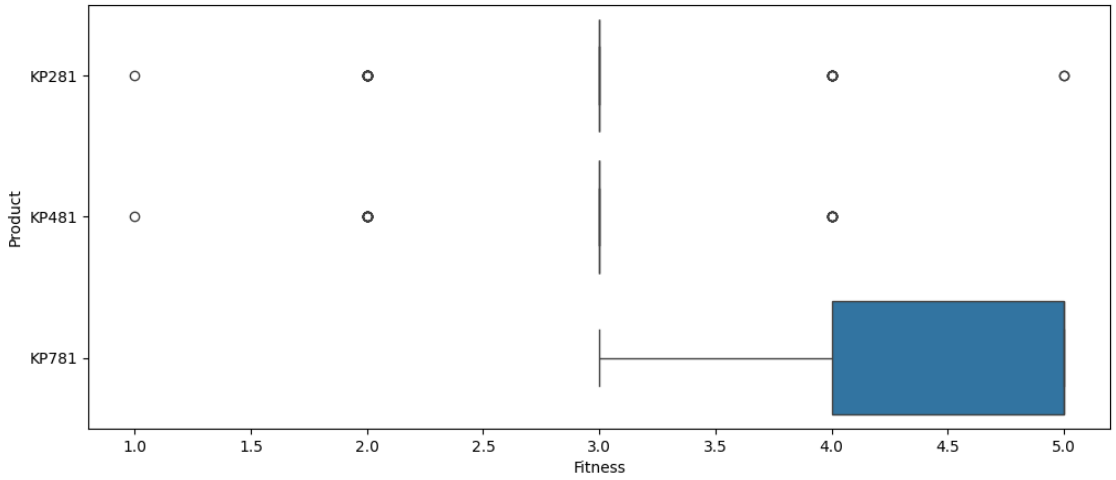
[ ]:
```
plt.figure(figsize=(12,5))
sns.boxplot(x='Age',y='Product',data=df)
plt.show()
```



Customers with Higher education of 16 to 18 have preferred KP781

Customers with education between 14 to 16 prefer KP281 and KP481 equally

[ ]:
```
plt.figure(figsize=(12,5))
sns.boxplot(x='Fitness',y='Product',data=df)
plt.show()
```



Customers with excellent shape are using KP781

single customers have higher proportion than partnered customers.

partnered customers are more than sngle customers.

REFERENCES FROM ALL THESE PLOTS

KP781 is more popular amon single nad partnered customers.

single females does nolt prefer much ofthe product.

ingle Female customers bought KP281 treadmill slightly more compared to Single Male customers.

single male customers bought KP781 more than single female customers.

single males are buying treadmill than single feamles.

ingle Female customers bought KP281 treadmill slightly more compared to Single Male customers.

[ ]:
```
df.isna().sum()
```

[ ]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
```

```
Miles               0
Fitness_category    0
dtype: int64
```

[ ]:
```
df.duplicated().sum()
```

[ ]:
```
0
```

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

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

Probability of buying

KP281 0.44

KP481 0.33

KP781 0.22

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

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

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

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

PROBABILITY OF BOTH PRODUCT FOR BOTH GENDER

[ ]:
```
def gender_Probability(gender,df):
    print(f"Prob P(KP781) for {gender}: {round(df['KP781'][gender]/df.
 ↪loc[gender].sum(),3)}")
    print(f"Prob P(KP481) for {gender}: {round(df['KP481'][gender]/df.
 ↪loc[gender].sum(),3)}")
    print(f"Prob P(KP281) for {gender}: {round(df['KP281'][gender]/df.
 ↪loc[gender].sum(),3)}")

df_temp = pd.crosstab(index=df['Gender'],columns=[df['Product']])
print("Prob of Male: ",round(df_temp.loc['Male'].sum()/len(df),3))
print("Prob of Female: ",round(df_temp.loc['Female'].sum()/len(df),3))
print()
```

```
gender_Probability('Male',df_temp)
print()
gender_Probability('Female',df_temp)
```

```
Prob of Male:  0.578
Prob of Female:  0.422

Prob P(KP781) for Male: 0.317
Prob P(KP481) for Male: 0.298
Prob P(KP281) for Male: 0.385

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

PROBABILITY OF EACH PRODUCT FOR MARITAL STATUS

[ ]:
```
def MS_Probability(ms_status,df):
    print(f"Prob P(KP781) for {ms_status}: {round(df['KP781'][ms_status]/df.
 ↪loc[ms_status].sum(),3)}")
    print(f"Prob P(KP481) for {ms_status}: {round(df['KP481'][ms_status]/df.
 ↪loc[ms_status].sum(),3)}")
    print(f"Prob P(KP281) for {ms_status}: {round(df['KP281'][ms_status]/df.
 ↪loc[ms_status].sum(),3)}")

df_temp = pd.crosstab(index=df['MaritalStatus'],columns=[df['Product']])
print("Prob of P(Single): ",round(df_temp.loc['Single'].sum()/len(df),3))
print("Prob of P(Married/Partnered): ",round(df_temp.loc['Partnered'].sum()/
 ↪len(df),3))
print()
MS_Probability('Single',df_temp)
print()
MS_Probability('Partnered',df_temp)
```

```
Prob of P(Single):  0.406
Prob of P(Married/Partnered):  0.594

Prob P(KP781) for Single: 0.233
Prob P(KP481) for Single: 0.329
Prob P(KP281) for Single: 0.438

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

[ ]:
```
df_category.head()
```

[ ]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	\
0	KP281	18	Male	14	Single	3	4	29562	
1	KP281	19	Male	15	Single	2	3	31836	
2	KP281	19	Female	14	Partnered	4	3	30699	
3	KP281	19	Male	12	Single	3	3	32973	
4	KP281	20	Male	13	Partnered	4	2	35247	

	Miles	Fitness_category
0	112	4
1	75	3
2	66	3
3	85	3
4	47	2

[ ]:
```
df_category['age_group'] = df_category.Age
df_category.head()
```

[ ]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	\
0	KP281	18	Male	14	Single	3	4	29562	
1	KP281	19	Male	15	Single	2	3	31836	
2	KP281	19	Female	14	Partnered	4	3	30699	
3	KP281	19	Male	12	Single	3	3	32973	
4	KP281	20	Male	13	Partnered	4	2	35247	

	Miles	Fitness_category	age_group
0	112	4	18
1	75	3	19
2	66	3	19
3	85	3	19
4	47	2	20

[ ]:
```
df_category.age_group.value_counts()
```

[ ]:

25	25
23	18
24	12
26	12
28	9
35	8
33	8
30	7
38	7
21	7
22	7
27	7
31	6
34	6

```
29     6
20     5
40     5
32     4
19     4
48     2
37     2
45     2
47     2
46     1
50     1
18     1
44     1
43     1
41     1
39     1
36     1
42     1
Name: age_group, dtype: int64
```

[ ]:
```
df_category.loc[df_category.Product=='KP281']["age_group"].value_counts()
```

[ ]:
```
23    8
25    7
26    7
28    6
24    5
38    4
21    4
22    4
29    3
19    3
27    3
35    3
34    2
33    2
32    2
31    2
30    2
20    2
41    1
47    1
46    1
44    1
43    1
18    1
40    1
```

```
39    1
37    1
36    1
50    1
Name: age_group, dtype: int64
```

[ ]:
```
df_category.loc[df_category.Product=='KP481']["age_group"].value_counts()
```

[ ]:
```
25    11
23     7
33     5
35     4
31     3
21     3
24     3
26     3
40     3
20     3
34     3
38     2
30     2
32     2
45     1
19     1
37     1
29     1
27     1
48     1
Name: age_group, dtype: int64
```

[ ]:
```
df_category.loc[df_category.Product=='KP781']["age_group"].value_counts()
```

[ ]:
```
25    7
24    4
22    3
27    3
28    3
30    3
23    3
26    2
29    2
40    1
47    1
45    1
42    1
31    1
38    1
```

```
35    1
34    1
33    1
48    1
Name: age_group, dtype: int64
```

[ ]:
```
pd.crosstab(index=df_category.Product,columns=df_category.
 ↪age_group,margins=True)
```

[ ]:
```
age_group  18  19  20  21  22  23  24  25  26  27  …  41  42  43  44  45  \
Product                                            …
KP281       1   3   2   4   4   8   5   7   7   3  …   1   0   1   1   0
KP481       0   1   3   3   0   7   3  11   3   1  …   0   0   0   0   1
KP781       0   0   0   0   3   3   4   7   2   3  …   0   1   0   0   1
All         1   4   5   7   7  18  12  25  12   7  …   1   1   1   1   2

age_group  46  47  48  50  All
Product
KP281       1   1   0   1   80
KP481       0   0   1   0   60
KP781       0   1   1   0   40
All         1   2   2   1  180

[4 rows x 33 columns]
```

[ ]:
```
np.round(pd.crosstab(index=df_category.Product,columns=df_category.
 ↪age_group,normalize=True,margins=True)*100,2)
```

[ ]:
```
age_group    18    19    20    21    22     23    24     25    26    27  … \
Product                                                                  …
KP281      0.56  1.67  1.11  2.22  2.22   4.44  2.78   3.89  3.89  1.67  …
KP481      0.00  0.56  1.67  1.67  0.00   3.89  1.67   6.11  1.67  0.56  …
KP781      0.00  0.00  0.00  0.00  1.67   1.67  2.22   3.89  1.11  1.67  …
All        0.56  2.22  2.78  3.89  3.89  10.00  6.67  13.89  6.67  3.89  …

age_group    41    42    43    44    45    46    47    48    50     All
Product
KP281      0.56  0.00  0.56  0.56  0.00  0.56  0.56  0.00  0.56   44.44
KP481      0.00  0.00  0.00  0.00  0.56  0.00  0.00  0.56  0.00   33.33
KP781      0.00  0.56  0.00  0.00  0.56  0.00  0.56  0.56  0.00   22.22
All        0.56  0.56  0.56  0.56  1.11  0.56  1.11  1.11  0.56  100.00

[4 rows x 33 columns]
```

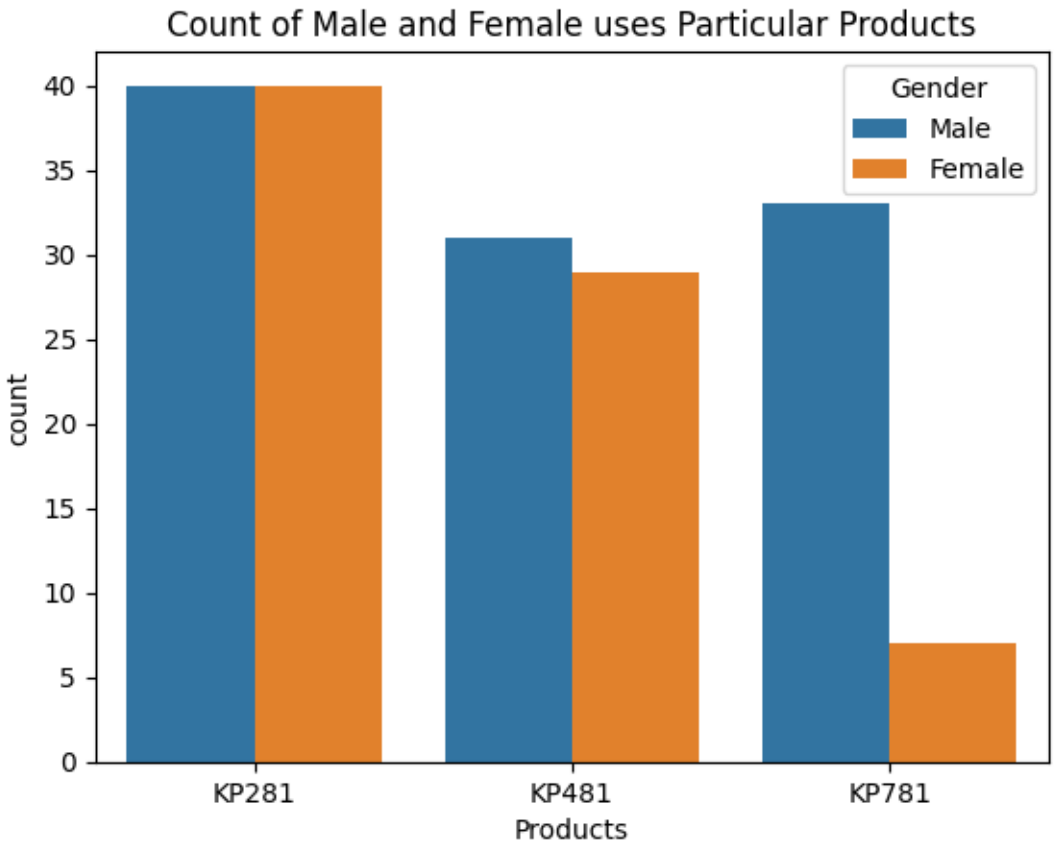
[ ]:
```
pd.
 ↪crosstab(columns=df_category["Fitness_category"],index=df_category["Product"])
```

[ ]:
```
Fitness_category  1   2   3  4   5
Product
KP281             1  14  54  9   2
KP481             1  12  39  8   0
KP781             0   0   4  7  29
```

[ ]:
```
round(pd.
  ↪crosstab(index=df_category["Product"],columns=df_category["Fitness_category"],normalize="co
```

[ ]:
```
Fitness_category     1      2      3      4      5
Product
KP281             50.0  53.85  55.67  37.50   6.45
KP481             50.0  46.15  40.21  33.33   0.00
KP781              0.0   0.00   4.12  29.17  93.55
```

[ ]:
```
sns.countplot(x = "Product", data= df, hue = "Gender")
plt.xlabel("Products")
plt.title("Count of Male and Female uses Particular Products")
plt.show()
```



[ ]:
```
pd.crosstab([df.Product],df.Gender,margins=True)
```

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

[ ]:
```
np.round(((pd.crosstab(df.Product,df.Gender,margins=True))/180)*100,2)
```

[ ]:
```
Gender   Female   Male     All
Product
KP281     22.22  22.22   44.44
KP481     16.11  17.22   33.33
KP781      3.89  18.33   22.22
All       42.22  57.78  100.00
```

MARGINAL PROBABILITY

PROBABILITY OF CUSTOMER PURCHASING ANY PRODUCT

MALE 57.77

FEMALE 42.22

PROBABLITY OF ANY CUSTOMER BUYIMG

KP281 44.44%

KP481 33.33%

KP781 22.22%

CONDITIONAL PROBABILITY

[ ]:
```
np.round((pd.crosstab([df.Product],df.
  ↪Gender,margins=True,normalize="columns"))*100,2)
```

[ ]:
```
Gender   Female   Male    All
Product
KP281     52.63  38.46  44.44
KP481     38.16  29.81  33.33
KP781      9.21  31.73  22.22
```

CUSTOMETR PROFILLING Probability of Selling Product

KP281 | Female = 52 %

KP481 | Female = 38 %

KP781 | Female = 10 %

KP281 | male = 38 %

KP481 | male = 30 %

KP781 | male = 32 %

Probability of Female customer buying KP281(52.63%) is more than male(38.46%).

KP281 is more recommended for female customers.

Probability of Male customer buying Product KP781(31.73%) is more than female(9.21%).

Probability of Female customer buying Product KP481(38.15%) is higher than male (29.80%.)

KP481 product is recommended for Female customers (intermediate user).

Customer Profile for KP481 Treadmill:

customer age between 18 to 35 years with few between 35 to 50 years Education level of customer 13 years and above

Annual Income of customer between USD 40,000 to USD 80,000

Weekly Usage - 2 to 4 times

Fitness Scale - 2 to 4

Weekly Running Mileage - 50 to 200 miles

Customer Profile for KP781 Treadmill:

Gender - Male

Age of customer between 18 to 35 years

Education level of customer 15 years and above

Annual Income of customer USD 80,000 and above

Weekly Usage - 4 to 7 times

Fitness Scale - 3 to 5

Weekly Running Mileage - 100 miles and above

RECOMMENDATION

MARKETTING(KP781) The KP784 model has sales change in terms of gender, with only 18% of total sales is to female customers. To enhance this metric, trials can be arranged for female customers.

User App development

Create app that aligns with the treadmill. add tracking features' weekly running mileage, provide feedback and offer recommendations for workouts for individuals.

Research is required for expanding market beyond 50 years of age.

Female prefer exercising equipments low.