```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/or
import pandas as pd
import numpy as np
df_original=pd.read_csv('aerofit_treadmill.csv?1639992749')
```

```
--2025-01-14 06:13:47--  https://d2beiqkhq929f0.cloudfront.net/public_asset
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)
HTTP request sent, awaiting response... 200 OK
Length: 7279 (7.1K) [text/plain]
Saving to: 'aerofit_treadmill.csv?1639992749'

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

2025-01-14 06:13:47 (139 MB/s) - 'aerofit_treadmill.csv?1639992749' saved [
```

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df=df_original
df.head()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
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	

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
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

```
#1. Import the dataset and do usual data analysis steps like checking the struc
# a. The data type of all columns in the “customers” table.

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
```

```
#b.  You can find the number of rows and columns given in the dataset

df.shape
```

```
(180, 9)
```

```
#c.  Check for the missing values and find the number of missing values in eacl

df.isna().sum()
```

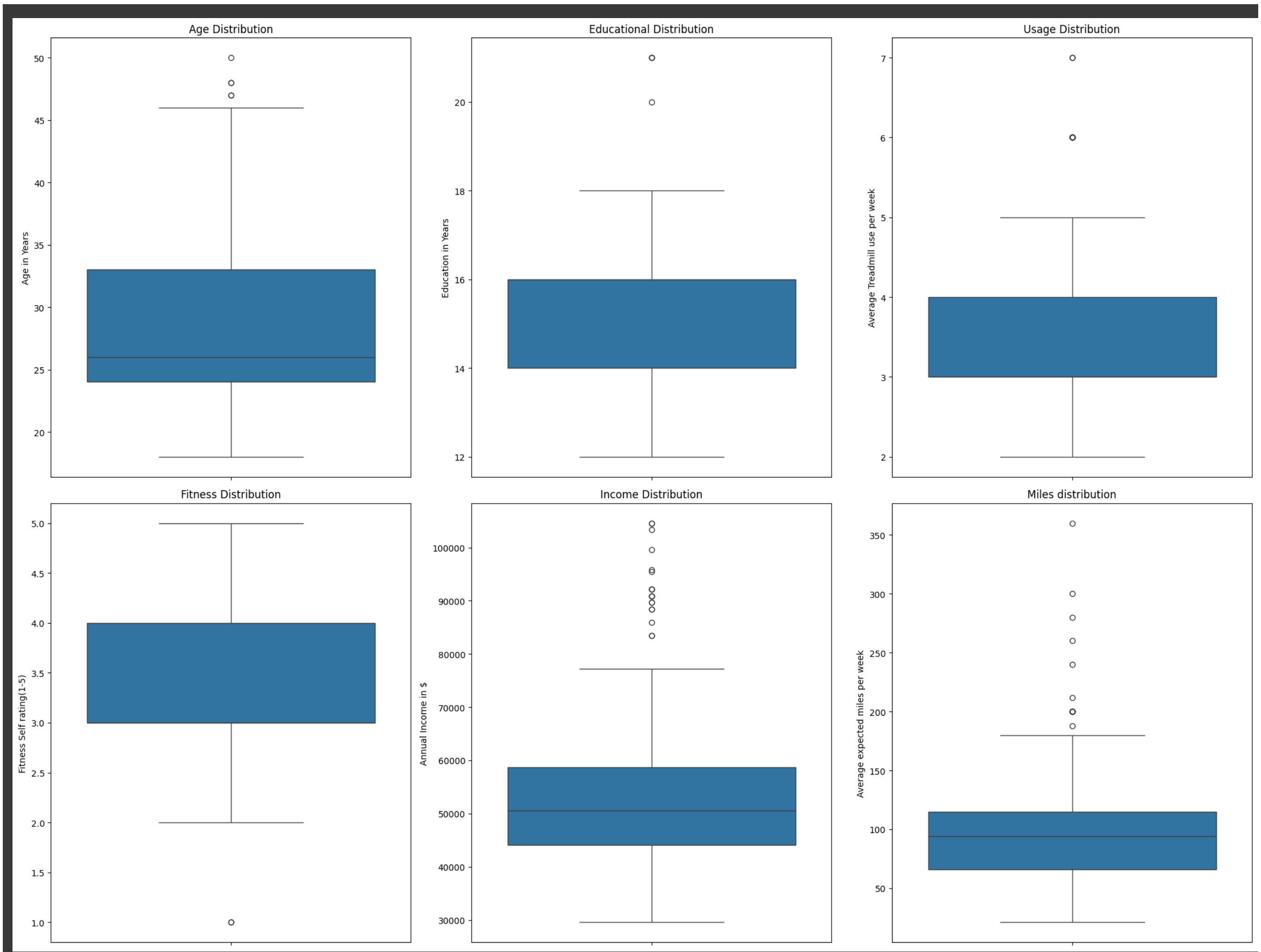
	0
Product	0
Age	0
Gender	0
Education	0
MaritalStatus	0
Usage	0
Fitness	0
Income	0
Miles	0

dtype: int64

As seen above, there are no columns with missing values.

```
#2 Detect Outliers
#a. Find the outliers for every continuous variable in the dataset using boxplc

plt.figure(figsize=[20,15])

#Age plot
plt.subplot(2,3,1)
sns.boxplot(data=df['Age'])
plt.title('Age Distribution')
plt.ylabel('Age in Years')

#Education plot
plt.subplot(2,3,2)
sns.boxplot(df['Education'])
plt.title('Educational Distribution')
plt.ylabel('Education in Years')


#Usage
plt.subplot(2,3,3)
sns.boxplot(df['Usage'])
```

```
plt.title('Usage Distribution')
plt.ylabel('Average Treadmill use per week')



#Fitness
plt.subplot(2,3,4)
sns.boxplot(df['Fitness'])
plt.title('Fitness Distribution')
plt.ylabel('Fitness Self rating(1-5)')

#Income
plt.subplot(2,3,5)
sns.boxplot(df['Income'])
plt.title('Income Distribution')
plt.ylabel('Annual Income in $')


#Miles
plt.subplot(2,3,6)
sns.boxplot(df['Miles'])
plt.title('Miles distribution')
plt.ylabel('Average expected miles per week')

plt.tight_layout()
plt.show()
```


Age Distribution
Age in Years
Educational Distribution
Education in Years
Usage Distribution
Average Treadmill use per week
Fitness Distribution
Fitness Self rating(1-5)
Income Distribution
Annual Income in $
Miles distribution
Average expected miles per week

We can clearly see the outliers with above box plots. For Age , we have outliers for those age greater than 46 Educational Years greater than 18 are the outliers. Treadmill usage of more than 5 per week are the outliers. Fitness Rating of less than 2 are the outliers. Annual Income greater than 80k are the outliers. Average miles more than 180 are the outliers.

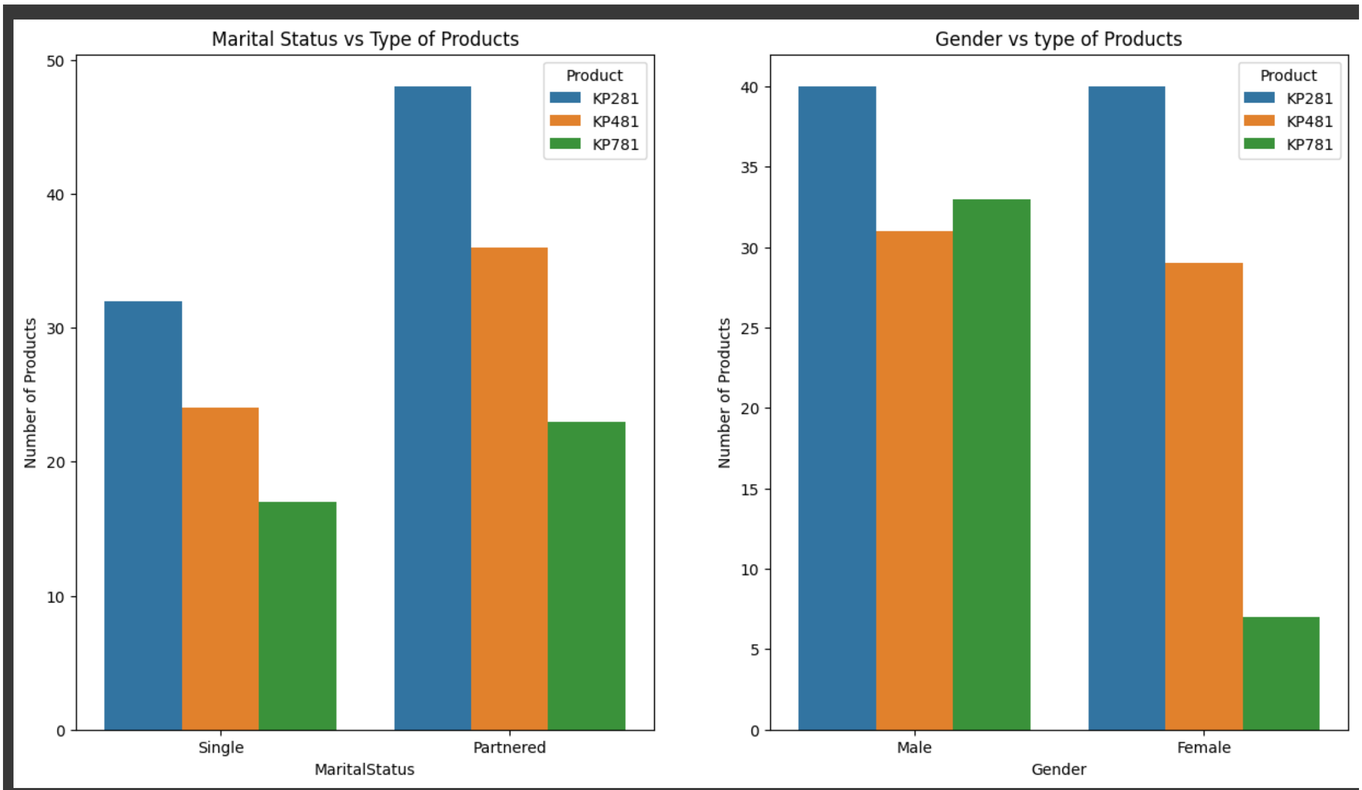
```
#b. Remove/clip the data between the 5 percentile and 95 percentile

#Age
age_p5=np.percentile(df['Age'],5)
age_p95=np.percentile(df['Age'],95)
df['Age']=np.clip(df['Age'],age_p5,age_p95)

#Education
edu_p5=np.percentile(df['Education'],5)
edu_p95=np.percentile(df['Education'],95)
df['Education']=np.clip(df['Education'],edu_p5,edu_p95)

#Usage
u_p5 = np.percentile(df['Usage'],5)
u_p95=np.percentile(df['Usage'],95)
df['Usage']=np.clip(df['Usage'],u_p5,u_p95)

#Fitness
f_p5=np.percentile(df['Fitness'],5)
f_p95=np.percentile(df['Fitness'],95)
df['Fitness']=np.clip(df['Fitness'],f_p5,f_p95)

#Income
i_p5=np.percentile(df['Income'],5)
i_p95=np.percentile(df['Income'],95)
df['Income']=np.clip(df['Income'],i_p5,i_p95)



#Miles
m_p5=np.percentile(df['Miles'],5)
m_p95=np.percentile(df['Miles'],95)
df['Miles']=np.clip(df['Miles'],m_p5,m_p95)
```

In Above code , we have clipped the column values between their 5 and 95 percentile. Below we can clearly see their min and max values being changed.

```
df.describe()
```

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.641389	15.572222	3.396944	3.322222	53477.070000	101.088889
std	6.446373	1.362017	0.952682	0.937461	15463.662523	43.364286
min	20.000000	14.000000	2.000000	2.000000	34053.150000	47.000000
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	43.050000	18.000000	5.050000	5.000000	90948.250000	200.000000

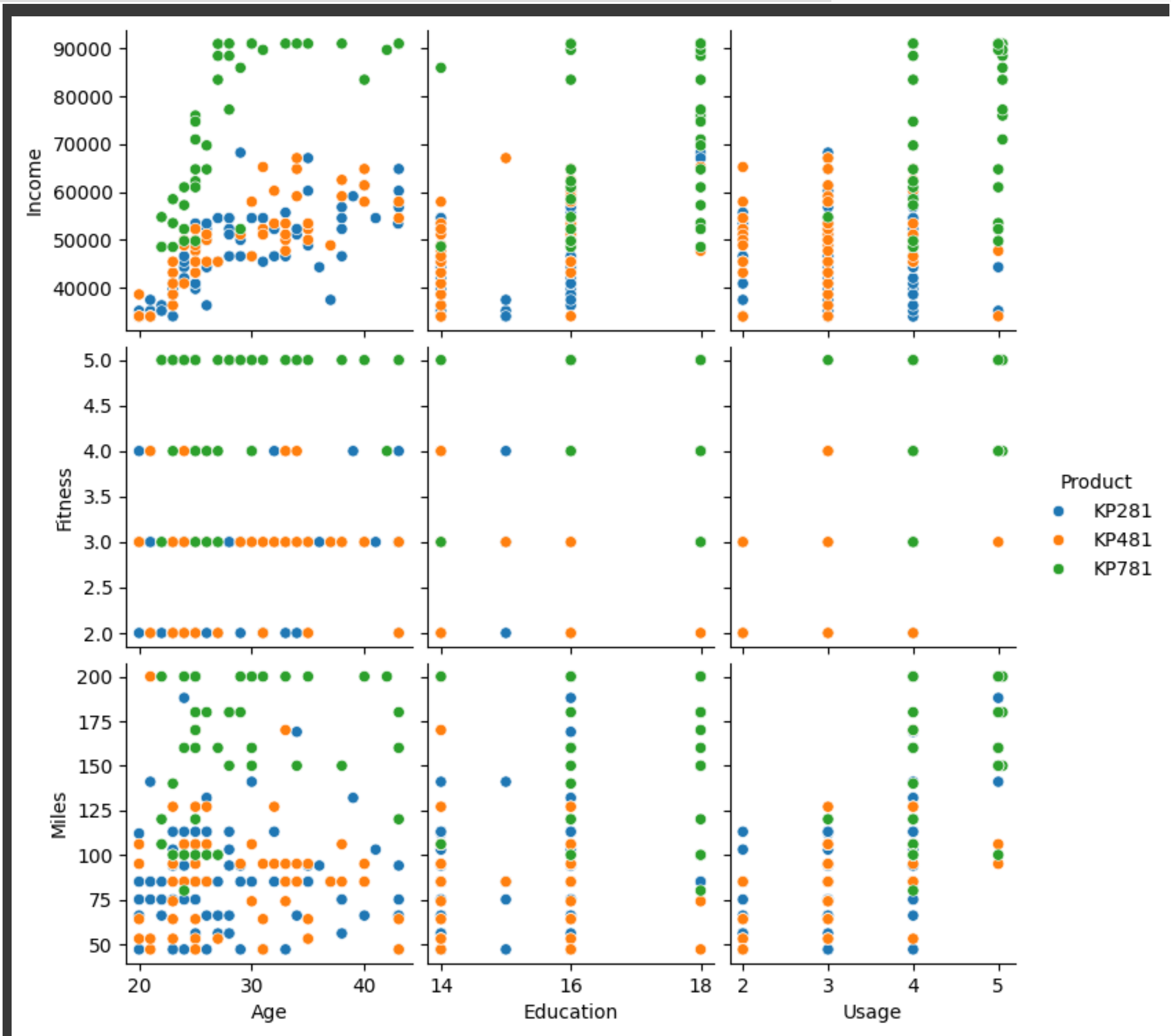
```
#3. Check if features like marital status, Gender, and age have any effect on t
#Find if there is any relationship between the categorical variables and the ou

plt.figure(figsize=[15,8])

#MaritalStatus
plt.subplot(1,2,1)
sns.countplot(x='MaritalStatus',hue='Product',data=df)
plt.ylabel('Number of Products')
plt.title('Marital Status vs Type of Products')


#Gender
plt.subplot(1,2,2)
sns.countplot(x='Gender',hue='Product',data=df)
plt.ylabel('Number of Products')
plt.title('Gender vs type of Products')


plt.show()
```



For Marital Status , we see there is significant difference in number of products where partnered ones have more products than the Single ones. Also , KP281 has the maximum count followed by KP481 and then KP781 irrespective of marital status.

For Gender,Both males and females bought same number of KP281 product and almost similar number of KP481 but KP781 were bought in more numbers by Males than Females

```
#b. Find if there is any relationship between the continuous variables and the

sns.pairplot(hue='Product',x_vars=['Age','Education','Usage'],y_vars=['Income',

plt.show()
```



Insights: For the above scatter plots , we can clearly see the following:

1. People with less Education have less income and using more KP481 product. People with mid education have mixed product usage and slighly greater income. Most of the KP781 users were people with highest education(18 years) and also highest income.
2. Fitness score with Age , it is quite visible that regardless of age most people using KP781 have rated themselves as highest (5) in fitness rating
3. People with low to medium income are using KP281 and KP481. However,KP781 usage is used mostly with young individuals(age < 30) irrespective of income but as the age increases only the high income people are using it.

```
# 4. Representing the Probability

#Find the marginal probability (what percent of customers have purchased KP281,

table=pd.crosstab(df['Product'],df['Gender'])
```

Gender Product	Female	Male
KP281	40	40
KP481	29	31
KP781	7	33

Next steps: Generate code with `table` View recommended plots New interactive sheet

```
#Marginal Probability
sum=table.sum().sum()

joint_probability=table/sum
marginal_probability=joint_probability.sum(axis=1)

marginal_probability
```

	0
Product	
KP281	0.444444
KP481	0.333333
KP781	0.222222

dtype: float64

Above is the Marginal probability of each product with respect to gender

```
#Find the probability that the customer buys a product based on each column.

marginal_probability_gender=joint_probability.sum(axis=0)

marginal_probability_gender
```

	0
Gender	
Female	0.422222
Male	0.577778

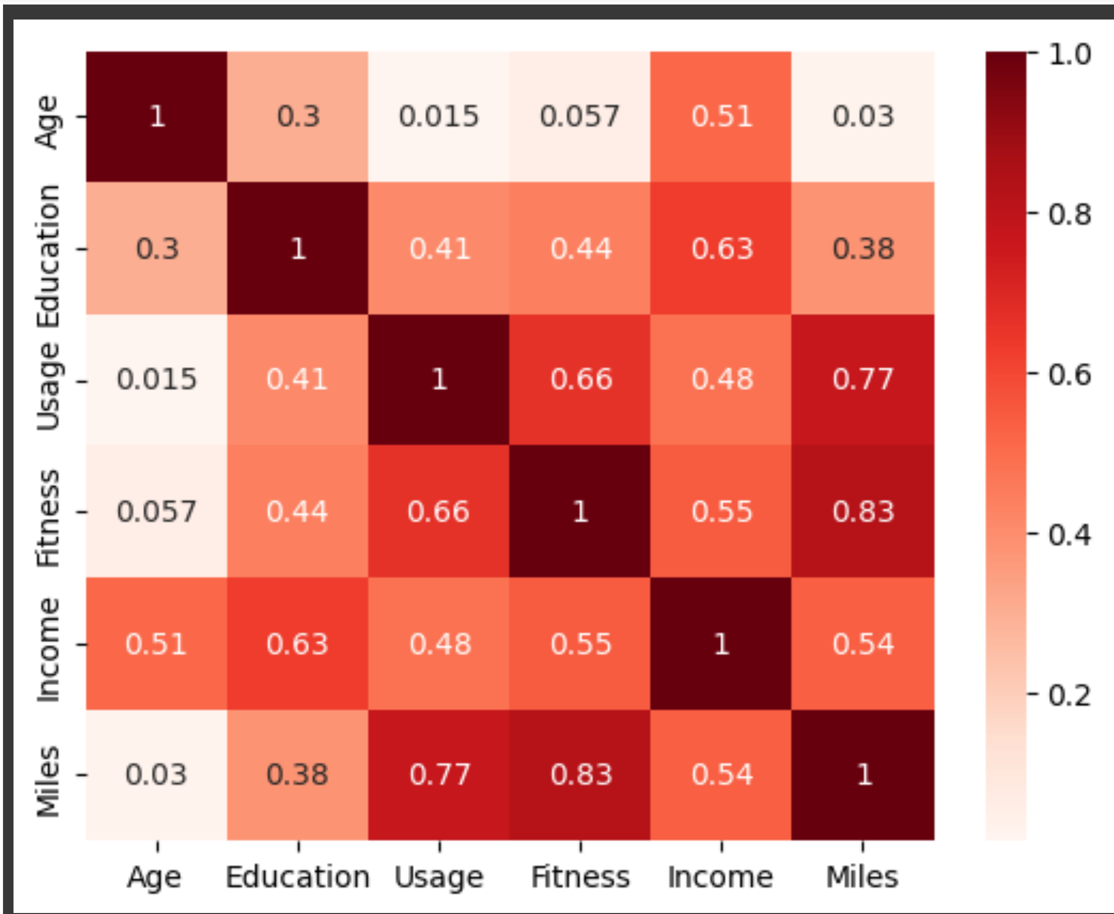
dtype: float64

Above is the marginal probability of type of customer buying the product.

```
#Find the conditional probability that an event occurs given that another event
# (Example: given that a customer is female, what is the probability
#she'll purchase a KP481)

P_KP281_F = table.loc['KP281','Female']/table['Female'].sum()
P_KP281_M = table.loc['KP281','Male']/table['Male'].sum()

P_KP481_F = table.loc['KP481','Female']/table['Female'].sum()
P_KP481_M = table.loc['KP481','Male']/table['Male'].sum()

P_KP781_F = table.loc['KP781','Female']/table['Female'].sum()
P_KP781_M = table.loc['KP781','Male']/table['Male'].sum()

print('Probability that customer is Male and products purchased KP281 , KP481 a

print('Probability that customer is Female and products purchased KP281 , KP481
```

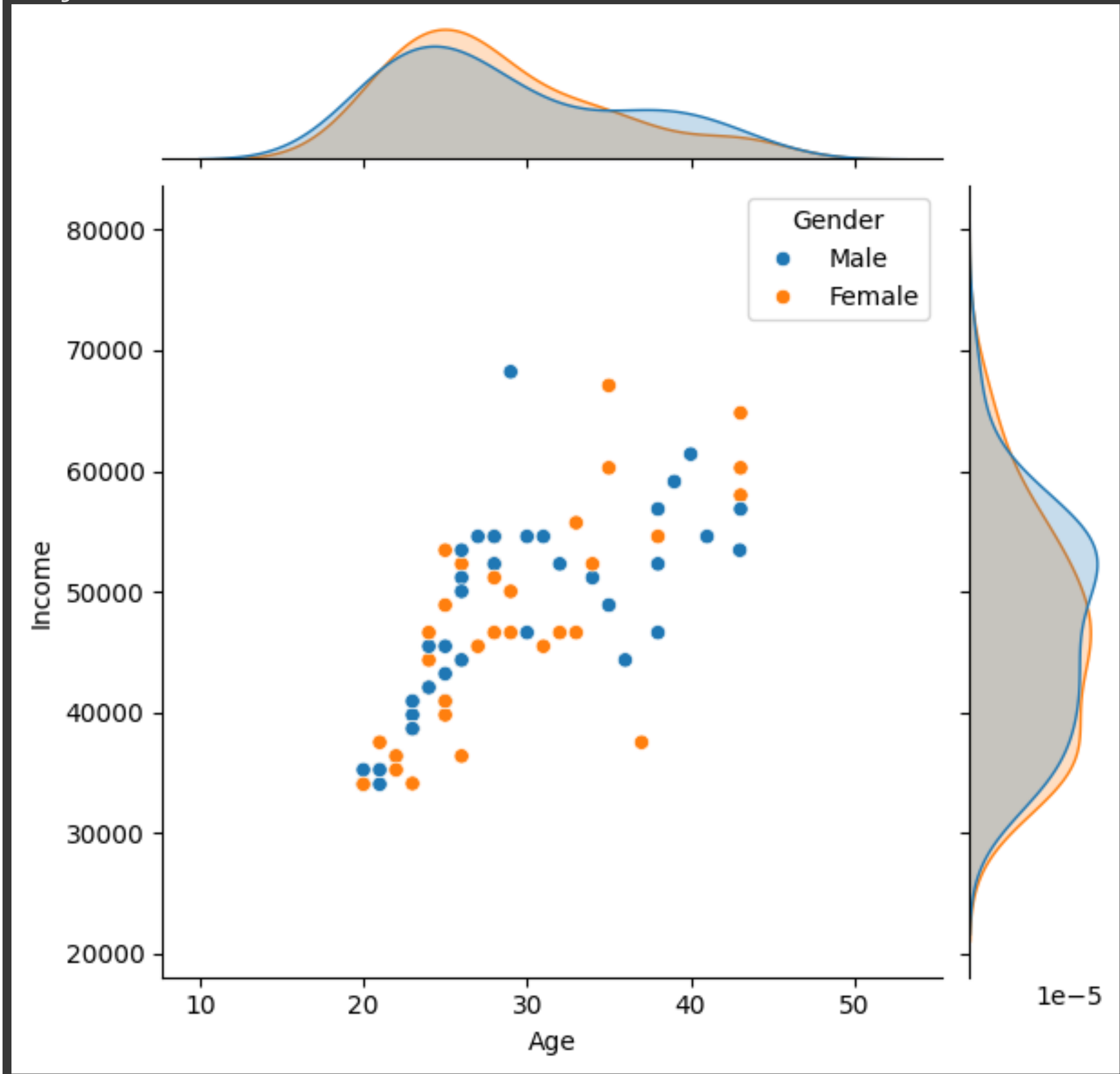
```
Probability that customer is Male and products purchased KP281 , KP481 and
Probability that customer is Female and products purchased KP281 , KP481 an
```

```
# 5.Check the correlation among different factors

#a.Find the correlation between the given features in the

num_df=df.select_dtypes(include=[int,float])
num_df.corr()
sns.heatmap(num_df.corr(),cmap='Reds',annot=True)
plt.show()
```



```
#6  Customer profiling and recommendation

# a. Make customer profilings for each and every product : gender , age and inc

#sns.scatterplot(x='Age',y='Income',hue='Gender',data=(df[df['Product']=='KP281


plt.figure(figsize=[20,20])
#KP281
sns.jointplot(x='Age',y='Income',hue='Gender',data=(df[df['Product']=='KP281'])

plt.show()
```

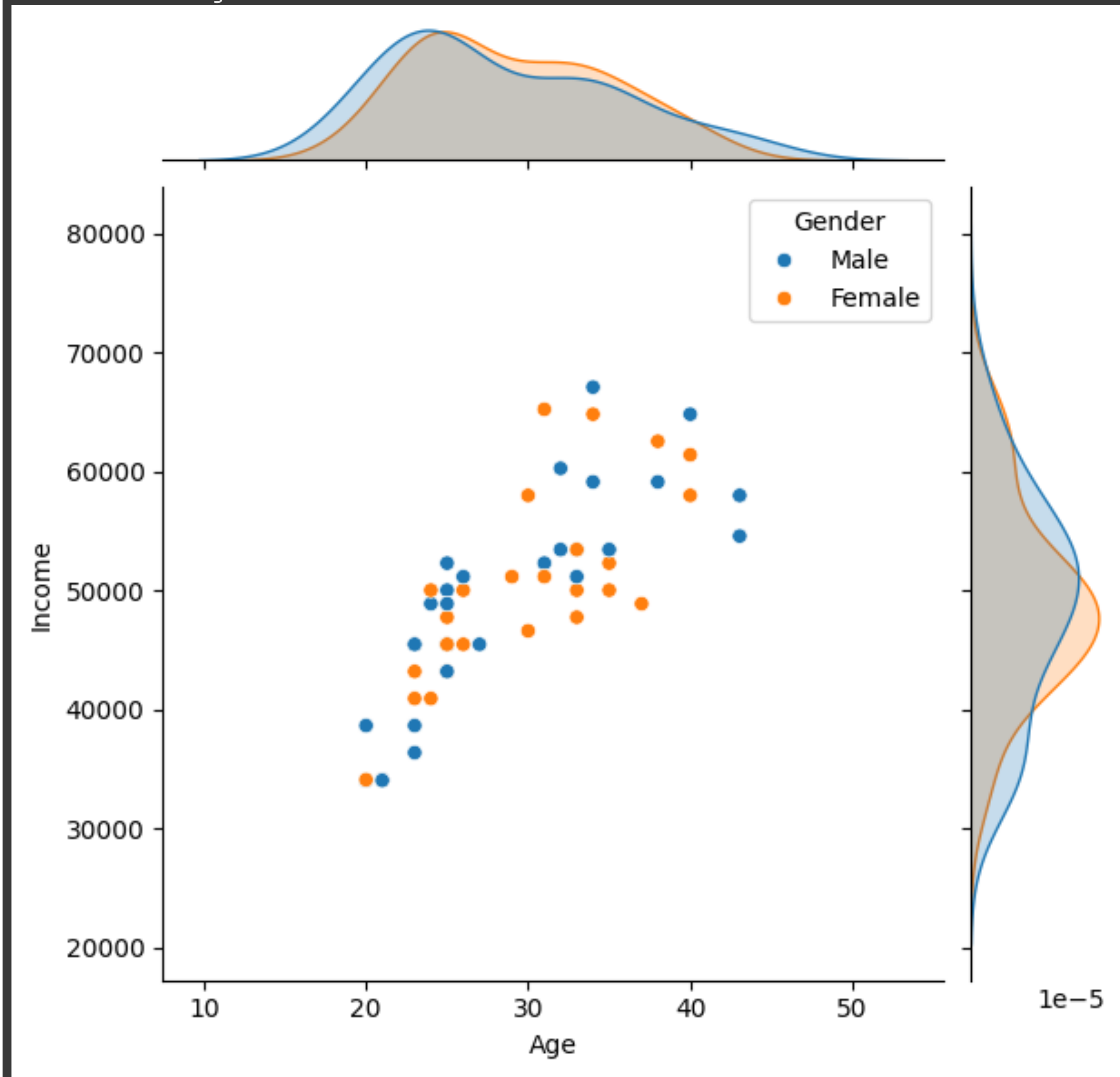
```
<Figure size 2000x2000 with 0 Axes>
```



For KP281 , people with age group 20-30 and annual income 50k-60k bought it most.

```
#KP481
sns.jointplot(x='Age',y='Income',hue='Gender',data=(df[df['Product']=='KP481'])
```

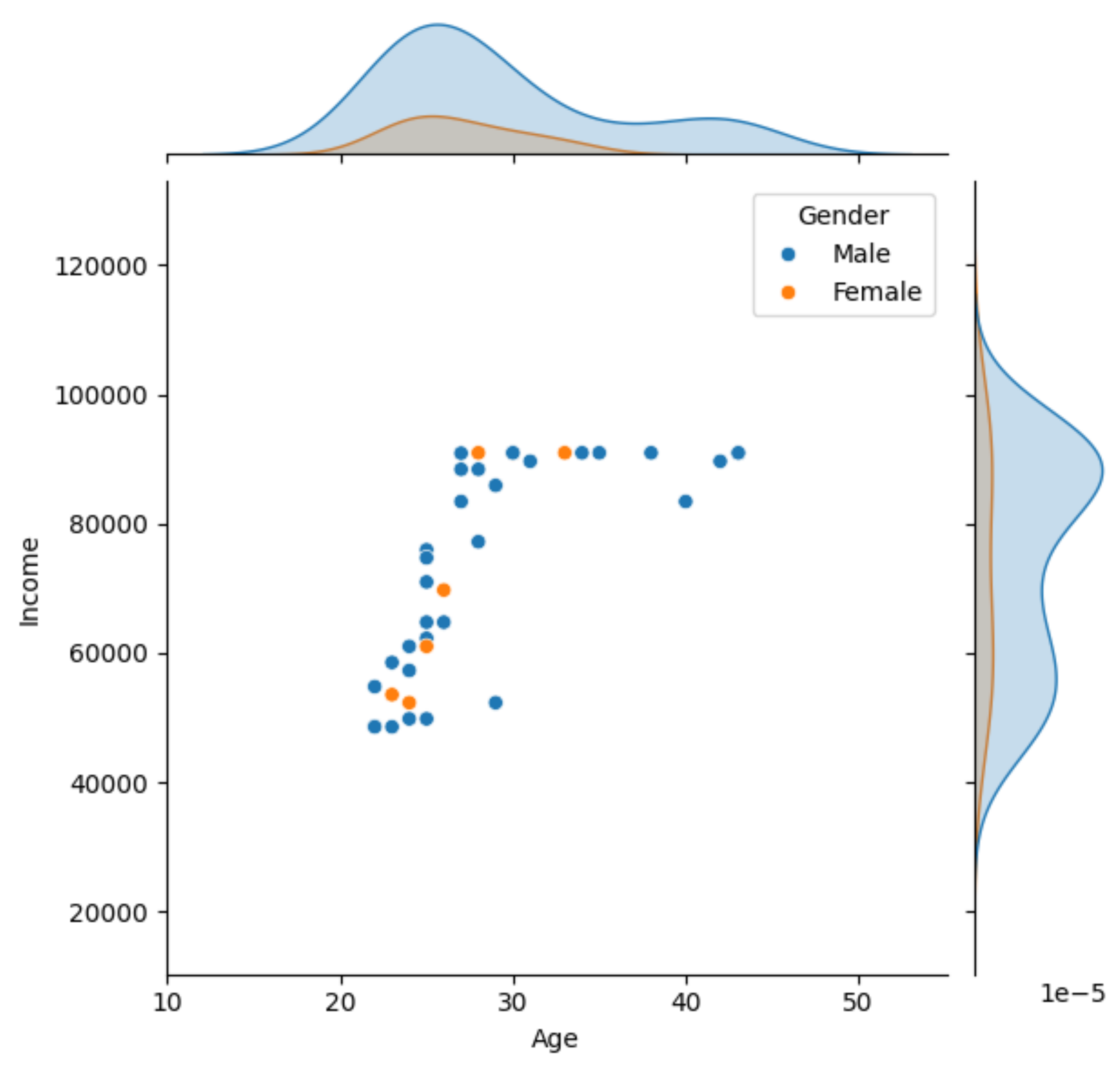
<seaborn.axisgrid.JointGrid at 0x7cb754d594b0>



For KP481 , Males and Females with age group 20-30 , Females with annual income 45k-55k and males with annual income 40k-60k bought it most.

```
#KP781
sns.jointplot(x='Age',y='Income',hue='Gender',data=(df[df['Product']=='KP781'])
```

```
<seaborn.axisgrid.JointGrid at 0x7cb755538070>
```



For KP781 , Males with age group 20-30 and annual income 80k-100k bought it most.

```
#b Write a detailed recommendation from the analysis that you have done.
```

Recommendations:

1. Product KP781 should be sold to Males with high income as a Target customers.
2. KP281 and KP481 can be focussed on selling it to young age group of 20-30
3. KP781 should also be sold to people with fitness rating 5 and also to people who's usage is high.