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

Part1

```
# 1.Observations on shape of data, data types of all the attributes, conversion of categorical attributes to 'category'
# 2.Non-Graphical Analysis: Value counts and unique attributes (10 Points)
```

```
df=pd.read_csv("/content/aerofit.csv")
```

```
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.tail()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
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

```
df.shape #cheking shape
```

```
(180, 9)
```

```
df.info() #information of col
```

```
<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
```

--> **Insight** 1.there is no null value

```
# unique value
for i in df.columns:
  print(i ,":",df[i].nunique())
```

```
Product : 3
Age : 32
Gender : 2
Education : 8
MaritalStatus : 2
Usage : 6
Fitness : 5
Income : 62
Miles : 37
```

```
# columns of datframe
df.columns
```

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

unique

```
df['Product'].unique()
```

```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

```
df.nunique()
```

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

```
# ckecking nulls
df.isnull().sum()
```

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

value counts

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

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

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

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

```
df['Education'].value_counts()
```

```
16    85
14    55
18    23
15     5
13     5
12     3
21     3
20     1
Name: Education, dtype: int64
```

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

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

```
df['Usage'].value_counts() #in each week
```

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

converting numerical to category type

```
df2=df.copy()
df2['Gender'].replace(['Male','Female'],[1,0],inplace=True)
df2['MaritalStatus'].replace(['Single','Partnered'],[0,1],inplace=True)
df2['Product'].replace(['KP281','KP481','KP781'],[0,1,2],inplace=True)
```

```
df2
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	0	18	1	14	0	3	4	29562	112
1	0	19	1	15	0	2	3	31836	75
2	0	19	0	14	1	4	3	30699	66
3	0	19	1	12	0	3	3	32973	85
4	0	20	1	13	1	4	2	35247	47
...	...	...	...	...	...	...	...	...	...
175	2	40	1	21	0	6	5	83416	200
176	2	42	1	18	0	5	4	89641	200
177	2	45	1	16	0	5	5	90886	160
178	2	47	1	18	1	4	5	104581	120
179	2	48	1	18	1	4	5	95508	180

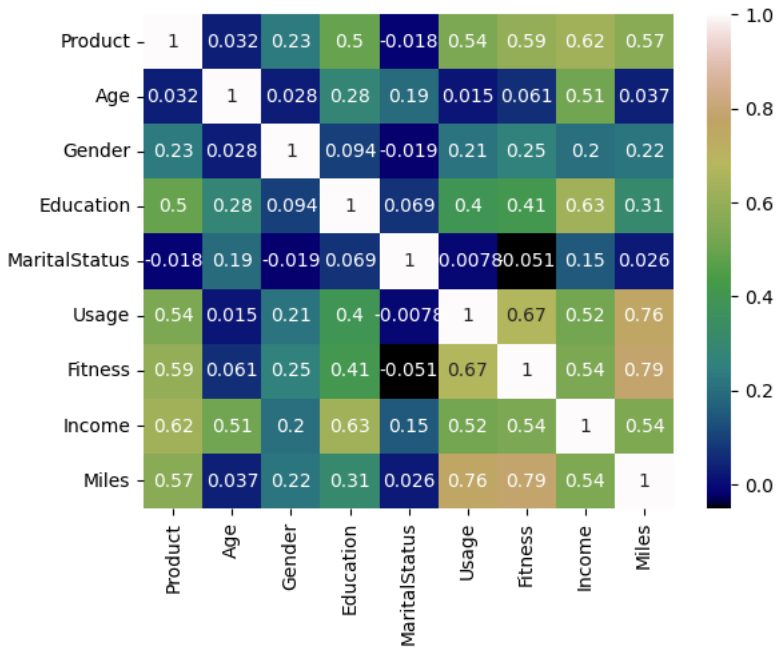
180 rows × 9 columns

```
df2.corr()
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
Product	1.000000	0.032225	0.230653	0.495018	-0.017602	0.537447	0.594883	0.624168	0.571596
Age	0.032225	1.000000	0.027544	0.280496	0.192152	0.015064	0.061105	0.513414	0.036618
Gender	0.230653	0.027544	1.000000	0.094089	-0.018836	0.214424	0.254609	0.202053	0.217869
Education	0.495018	0.280496	0.094089	1.000000	0.068569	0.395155	0.410581	0.625827	0.307284
MaritalStatus	-0.017602	0.192152	-0.018836	0.068569	1.000000	-0.007786	-0.050751	0.150293	0.025639
Usage	0.537447	0.015064	0.214424	0.395155	-0.007786	1.000000	0.668606	0.519537	0.759130
Fitness	0.594883	0.061105	0.254609	0.410581	-0.050751	0.668606	1.000000	0.535005	0.785702
Income	0.624168	0.513414	0.202053	0.625827	0.150293	0.519537	0.535005	1.000000	0.543473
Miles	0.571596	0.036618	0.217869	0.307284	0.025639	0.759130	0.785702	0.543473	1.000000

```
sns.heatmap(df2.corr(),cmap='gist_earth',annot=True)
```

```
<Axes: >
```



insight

1.product is (+ve)ly corr with related to (Usage,Fitness,Income,Miles)

2.gender(-ve) corr with Age,Education,MaritalStatus.....and(+ve) corr with Fitness,Miles

3.Age is (+ve)ly corr with income.

4.Education is(+ve)corr with (Usage,Fitness,Income,Miles)...and(-ve)corr with (MaritalStatus,Gender)

5.Usage is(-ve)corr with (Age,MartialStatus)....and (+ve)corr with (Fitness,Income,Miles)

6.Fitness is (+ve)corr with (Usage,Income,Milesm)

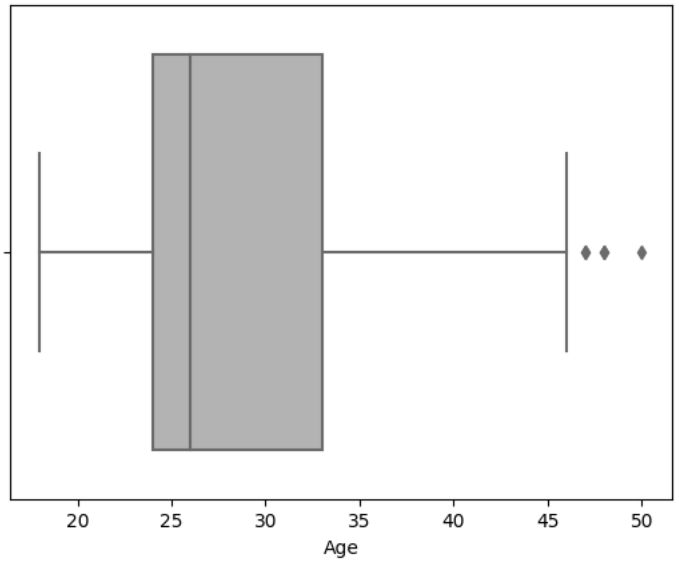
7.miles is(+ve)corr with ((Usage,Fitness,Income) .....and (-ve)corr with (Age,Martialstatus)

```
#outliers
```

Missing Value & Outlier Detection (10 Points)

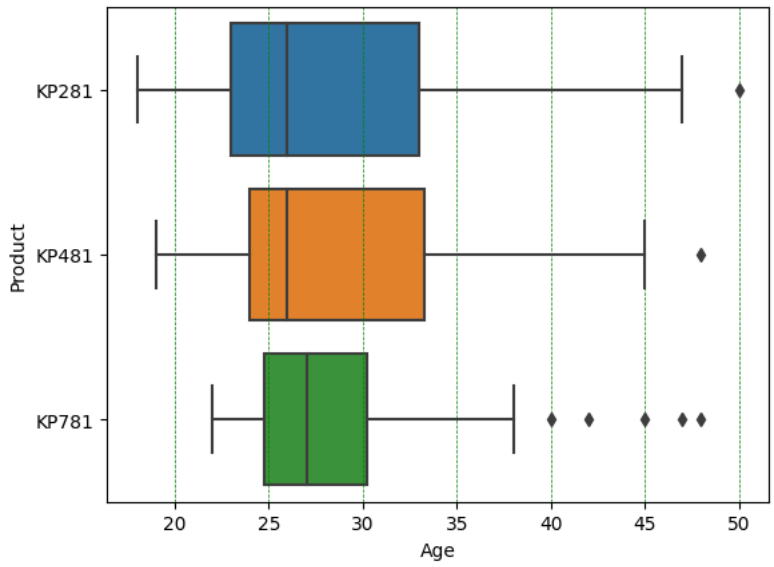
```
sns.boxplot(x=df['Age'],color=".7",)
```

```
<Axes: xlabel='Age'>
```



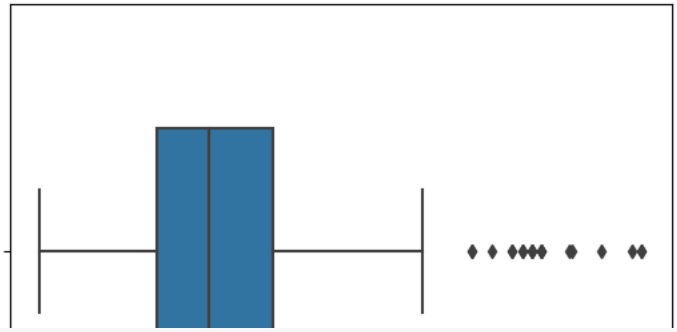

```
plt.grid(color = 'green', linestyle = '--', linewidth = 0.5)
sns.boxplot(x=df['Age'],y=df['Product'])
```

```
<Axes: xlabel='Age', ylabel='Product'>
```



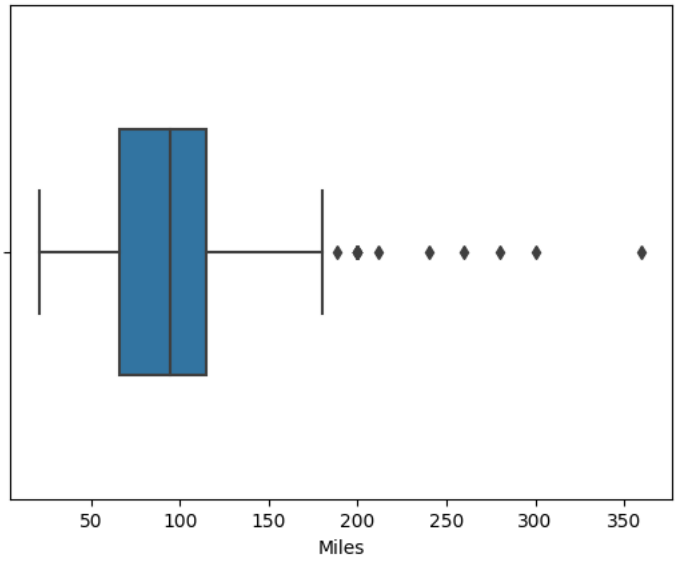

```
sns.boxplot(x=df["Income"],width=.5)
```

```
<Axes: xlabel='Income'>
```



```
sns.boxplot(x=df["Miles"],width=.5)
```

```
<Axes: xlabel='Miles'>
```

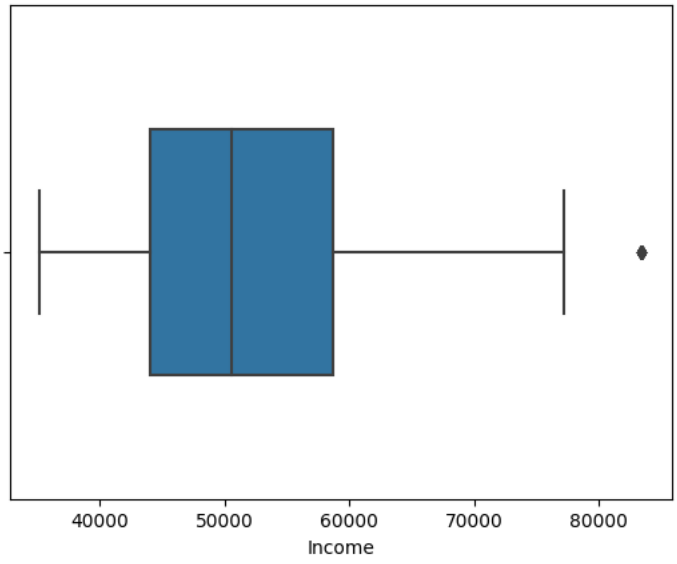


Note:As there are only few data sample we can us clliping

```
list_=['Miles','Age','Income']
for i in list_:
  df2[i]=np.clip(df2[i],a_min=np.percentile(df2[i],10),a_max=np.percentile(df2[i],90))
```

```
#checking outliers
sns.boxplot(x=df2["Income"],width=.5)
```

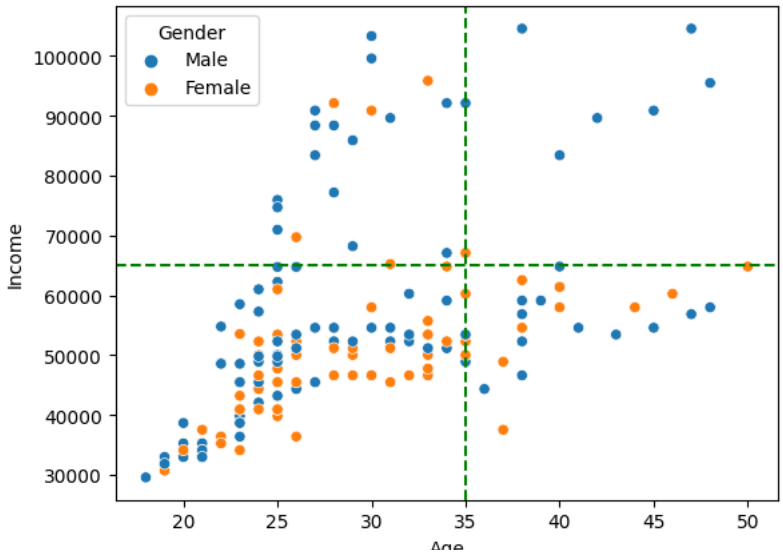
```
<Axes: xlabel='Income'>
```



Note:from above outliers decrease

```
# scattere plot to know
# diff age vs income
sns.scatterplot(data=df,x=df['Age'],y=df['Income'],hue="Gender")
plt.axvline(35, color="g", linestyle="--")
plt.axhline(65000, color="g", linestyle="--")
```

```
<matplotlib.lines.Line2D at 0x7c0e9864ff10>
```

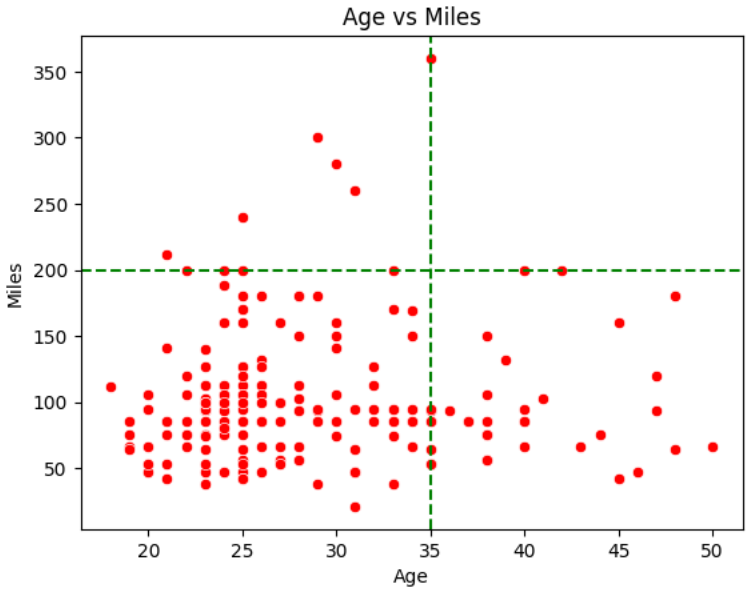


Double-click (or enter) to edit

```
# diff age vs Miles

sns.scatterplot(data=df,x=df['Age'],y=df['Miles'],color='red')
plt.axvline(35, color="g", linestyle="--")
plt.axhline(200, color="g", linestyle="--")
plt.title('Age vs Miles')
```

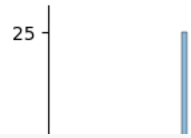
```
Text(0.5, 1.0, 'Age vs Miles')
```



Note-->3rd quardent has more scatter plots 2.less age group run more miles

```
sns.displot(df['Age'],kde=True,bins=100)
```

```
<seaborn.axisgrid.FacetGrid at 0x7ddc70587670>
```



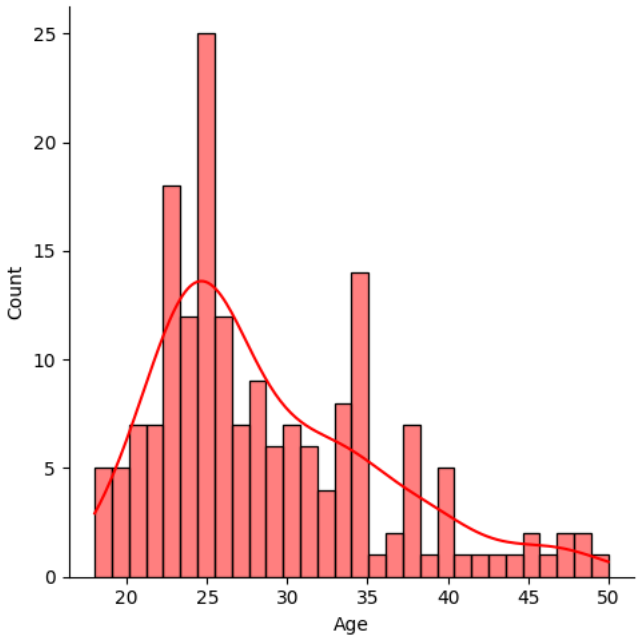

```
df['Age'].describe()
```

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



```
#obsorving the age to creat bins
plt.figure(figsize=(4,4))
sns.displot(df['Age'], kde=True,bins=30,color='red')
plt.show()
```

```
<Figure size 400x400 with 0 Axes>
```




```
#creat 5 group of age
bins = [-1,20,25,30,35,40,55]
label=['<20','20-25','25-30','30-35','35-40','40+']
df['Age_bins']=pd.cut(df['Age'],bins=bins,labels=label)
df.head()
```

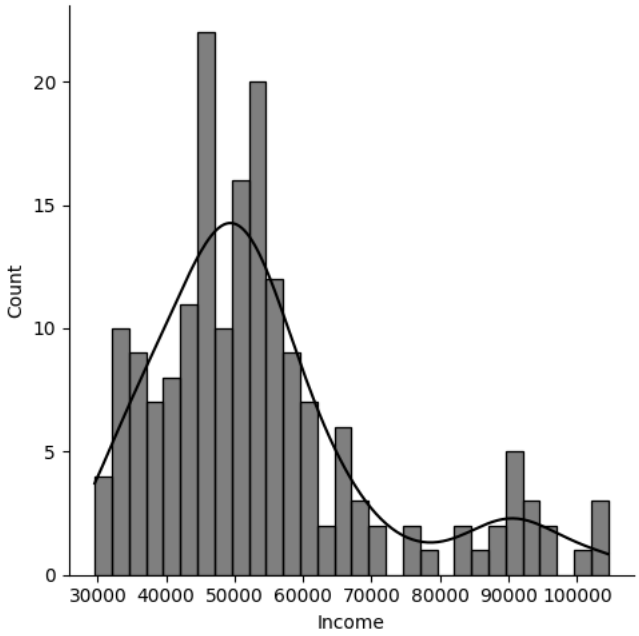
	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_bins
0	KP281	18	Male	14	Single	3	4	29562	112	<20
1	KP281	19	Male	15	Single	2	3	31836	75	<20
2	KP281	19	Female	14	Partnered	4	3	30699	66	<20
3	KP281	19	Male	12	Single	3	3	32973	85	<20
4	KP281	20	Male	13	Partnered	4	2	35247	47	<20

```
df['Income'].describe()
```

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

```
plt.figure(figsize=(4,4))
sns.displot(df['Income'], kde=True,bins=30,color='black')
plt.show()
```

<Figure size 400x400 with 0 Axes>




```
bins = [0,30000,40000,50000,60000,70000,80000,90000,110000]
label=['0-30k','30k-40k','40k-50k','50k-60k','60k-70k','70k-80k','80k-90k','90k-110k']
df['income_bins']=pd.cut(df['Income'],bins=bins,labels=label)
df.head(20)
```

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Age_bins	in
0	KP281	18	Male	14	Single	3	4	29562	112	<20	
1	KP281	19	Male	15	Single	2	3	31836	75	<20	
2	KP281	19	Female	14	Partnered	4	3	30699	66	<20	
3	KP281	19	Male	12	Single	3	3	32973	85	<20	
4	KP281	20	Male	13	Partnered	4	2	35247	47	<20	
5	KP281	20	Female	14	Partnered	3	3	32973	66	<20	
6	KP281	21	Female	14	Partnered	3	3	35247	75	20-25	
7	KP281	21	Male	13	Single	3	3	32973	85	20-25	
8	KP281	21	Male	15	Single	5	4	35247	141	20-25	
9	KP281	21	Female	15	Partnered	2	3	37521	85	20-25	
10	KP281	22	Male	14	Single	3	3	36384	85	20-25	
11	KP281	22	Female	14	Partnered	3	2	35247	66	20-25	
12	KP281	22	Female	16	Single	4	3	36384	75	20-25	
13	KP281	22	Female	14	Single	3	3	35247	75	20-25	
14	KP281	23	Male	16	Partnered	3	1	38658	47	20-25	
15	KP281	23	Male	16	Partnered	3	3	40932	75	20-25	
16	KP281	23	Female	14	Single	2	3	34110	103	20-25	
17	KP281	23	Male	16	Partnered	4	3	39795	94	20-25	
18	KP281	23	Female	16	Single	4	3	38658	113	20-25	
19	KP281	23	Female	15	Partnered	2	2	34110	38	20-25	

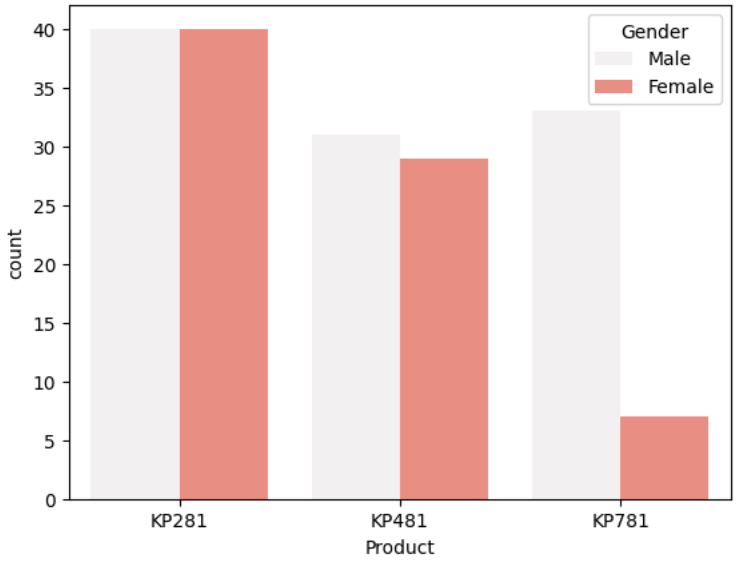
Check if features like marital status, age have any effect on the product purchased

```
#crosstab
#Gender effect on product purchase
A=pd.crosstab(index=df['Gender'],columns=df['Product'],margins=True,normalize=True)
A
```

Product	KP281	KP481	KP781	All
Gender				
Female	0.222222	0.161111	0.038889	0.422222

```
sns.countplot(data=df,x='Product',hue='Gender',color="salmon")
```

```
<Axes: xlabel='Product', ylabel='count'>
```

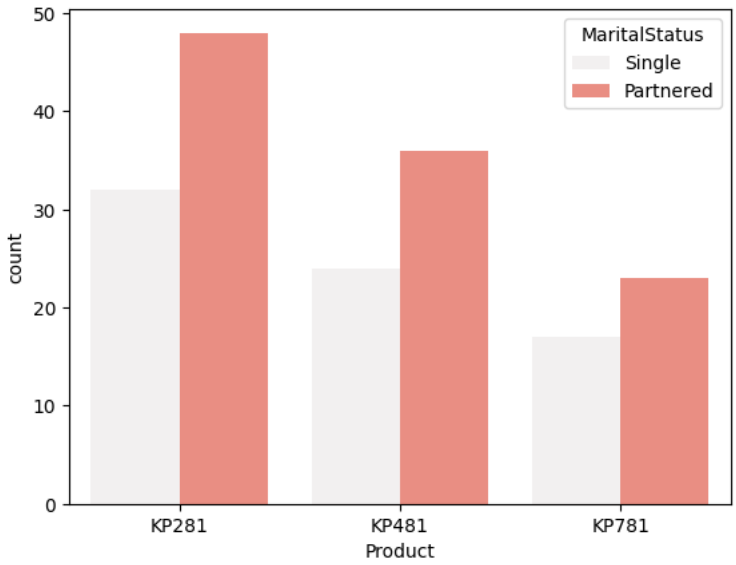



```
#Martialstatus effect on product purchase
B=pd.crosstab(index=df['MaritalStatus'],columns=df['Product'],margins=True,normalize=True)
B
```

Product	KP281	KP481	KP781	All
MaritalStatus				
Partnered	0.266667	0.200000	0.127778	0.594444
Single	0.177778	0.133333	0.094444	0.405556
All	0.444444	0.333333	0.222222	1.000000

```
sns.countplot(data=df,x='Product',hue='MaritalStatus',color="salmon")
```

```
<Axes: xlabel='Product', ylabel='count'>
```



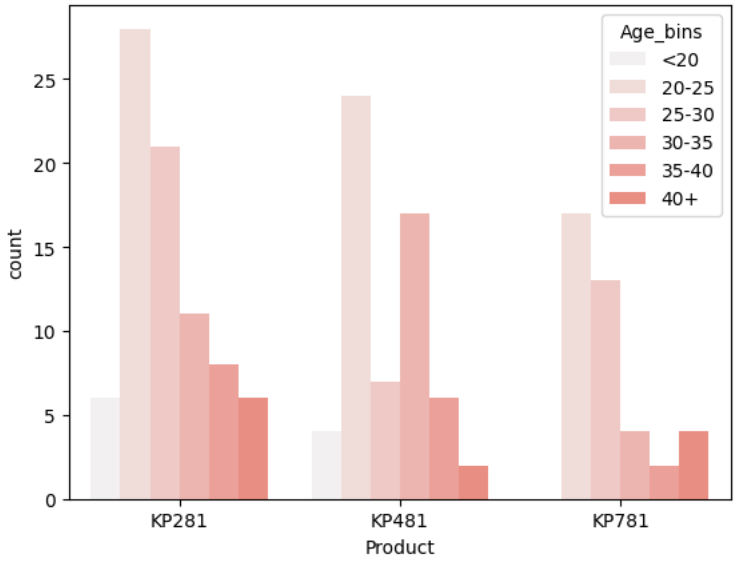

```
#Age effect on product purchase
```

```
C=pd.crosstab(index=df['Age_bins'],columns=df['Product'],margins=True,normalize=True)
C
```

Product	KP281	KP481	KP781	All
Age_bins				
<20	0.033333	0.022222	0.000000	0.055556
20-25	0.155556	0.133333	0.094444	0.383333
25-30	0.116667	0.038889	0.072222	0.227778
30-35	0.061111	0.094444	0.022222	0.177778
35-40	0.044444	0.033333	0.011111	0.088889

```
sns.countplot(data=df,x='Product',hue='Age_bins',color="salmon")
```

```
<Axes: xlabel='Product', ylabel='count'>
```



insight..,

1.product KP281 and kp481 has equal sales between males and female .....but kp781 has more male purchase

2.Partnerd has more sales in every product which is 59.4%

3.Age group between 20-25 has heightest sales of all prodict...,which is 38.4%

4.so we can give more offer to Partnered customers...we can show more benificial of product as a marketing to attract young people....we can also use young fitness traners ....revies of beficial of tradmils

5.1.3rd quardent has more scatter plots..... 2.less age group run more miles in age vs Miles graph

```
(69/180)*100
```

```
38.333333333333336
```