In [1]:

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

In [2]:

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

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv (http
s://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aerofit_treadmill.csv)
To: C:\Users\anusha\Desktop\Numpy\aerofit_treadmill.csv

  0%|          | 0.00/7.28k [00:00<?, ?B/s]
100%|##########| 7.28k/7.28k [00:00<?, ?B/s]
```

In [2]:

```
df = pd.read_csv("aerofit_treadmill.csv")
```

In [3]:

```
df
```

Out[3]:

	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
...	...	...	...	...	...	...	...	...	...
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

180 rows × 9 columns

1b.Observations on the shape of data, data types of all the attributes,

#conversion of categorical attributes to 'category' (If required), missing value detection, statistical summary

In [4]:

```
df.shape
```

Out[4]:

```
(180, 9)
```

In [5]:

```
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
```

In [6]:

```
df.describe(include="all")
```

Out[6]:

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

In [5]:

```
df.describe(include="object")
```

Out[5]:

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

In [8]:

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

Out[8]:

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

2 QUestion value counts and unique attributes

In [7]:

```
df.columns
```

Out[7]:

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

In [6]:

```
product_count = df['Product'].value_counts().reset_index()
product_count.columns = ['Product_Name', 'Count']
product_count
```

Out[6]:

	Product_Name	Count
0	KP281	80
1	KP481	60
2	KP781	40

In [8]:

```
gender_count = df['Gender'].value_counts().reset_index()
gender_count.columns = ['Gender', 'Count']
gender_count
```

Out[8]:

	Gender	Count
0	Male	104
1	Female	76

In [9]:

```
marital_status_count = df['MaritalStatus'].value_counts().reset_index()
marital_status_count.columns = ['MaritalStatus', 'Count']
marital_status_count
```

Out[9]:

	MaritalStatus	Count
0	Partnered	107
1	Single	73

In [10]:

```
Fitness_count = df['Fitness'].value_counts().reset_index()
Fitness_count.columns = ['Fitness', 'Count']
Fitness_count
```

Out[10]:

	Fitness	Count
0	3	97
1	5	31
2	2	26
3	4	24
4	1	2

In [11]:

```
df.describe()[['Age', 'Income', 'Miles']].loc[['min', 'max']]
```

Out[11]:

	Age	Income	Miles
min	18.0	29562.0	21.0
max	50.0	104581.0	360.0

In [12]:

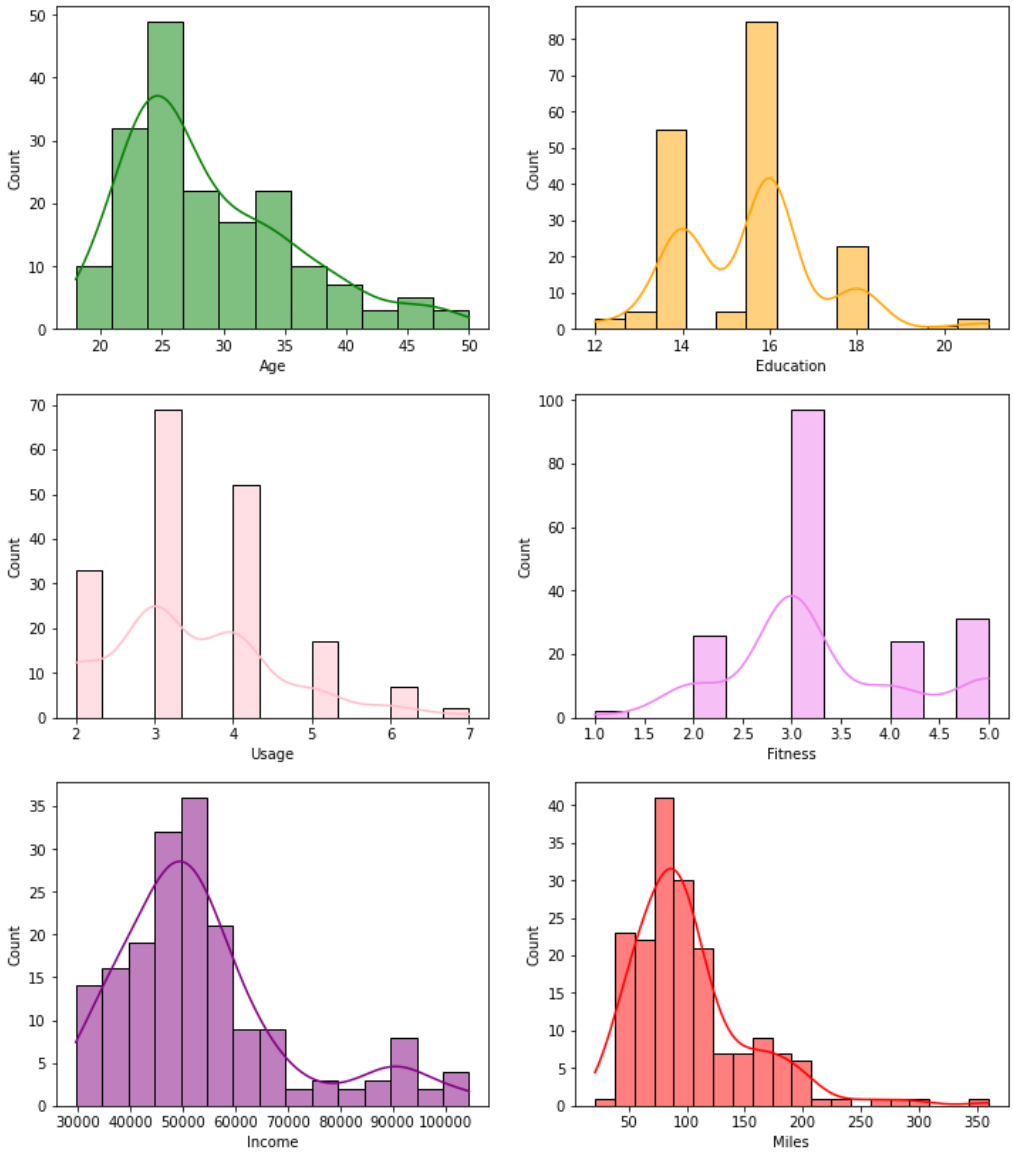
```
for i in df.columns:
    print(i,':',df[i].nunique())
```

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

3 Visual Analysis - Univariate & Bivariate

In [10]:

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

sns.histplot(data=df, x="Age", kde=True, ax=axis[0,0],color ='Green')
sns.histplot(data=df, x="Education", kde=True, ax=axis[0,1],color = 'Orange')
sns.histplot(data=df, x="Usage", kde=True, ax=axis[1,0],color = 'Pink')
sns.histplot(data=df, x="Fitness", kde=True, ax=axis[1,1],color = 'Violet')
sns.histplot(data=df, x="Income", kde=True, ax=axis[2,0],color = 'Purple')
sns.histplot(data=df, x="Miles", kde=True, ax=axis[2,1],color = 'Red')
plt.show()
```



In [10]:

```
df["Income"].max()
```

Out[10]:

```
104581
```

In [11]:

```
df["Income"].min()
```

Out[11]:

```
29562
```

In [4]:

```
Income_bins = [25000,45000,65000,85000,105000]
Income_labels=["Low","Medium","High","very_High"]
df["Income_new"]=pd.cut(df["Income"],
                        bins=Income_bins,
                        labels=Income_labels,
                        right=True)
```

In [5]:

```
df
```

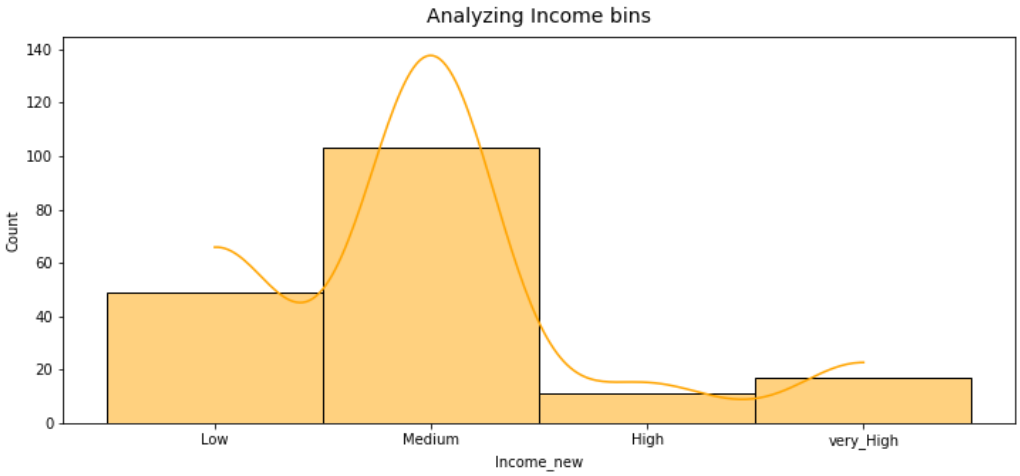
Out[5]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Income_new
0	KP281	18	Male	14	Single	3	4	29562	112	Low
1	KP281	19	Male	15	Single	2	3	31836	75	Low
2	KP281	19	Female	14	Partnered	4	3	30699	66	Low
3	KP281	19	Male	12	Single	3	3	32973	85	Low
4	KP281	20	Male	13	Partnered	4	2	35247	47	Low
...	...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200	High
176	KP781	42	Male	18	Single	5	4	89641	200	very_High
177	KP781	45	Male	16	Single	5	5	90886	160	very_High
178	KP781	47	Male	18	Partnered	4	5	104581	120	very_High
179	KP781	48	Male	18	Partnered	4	5	95508	180	very_High

180 rows × 10 columns

In [11]:

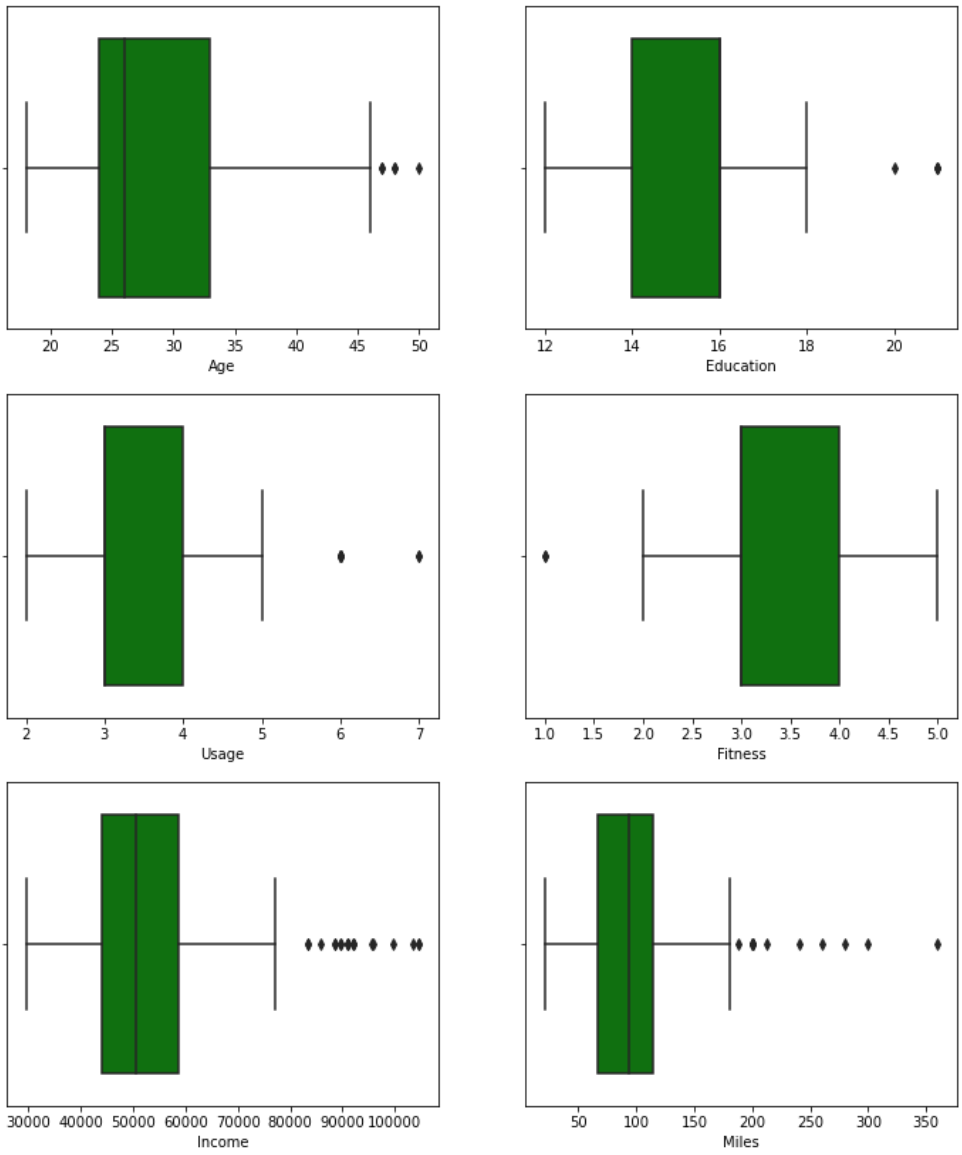
```
plt.figure(figsize=(12,5))
sns.histplot(data=df, x="Income_new", kde=True,color ='Orange')
plt.title("Analyzing Income bins", pad=10, fontsize=14)
plt.show()
```



Boxplot usage for outliners detection

In [11]:

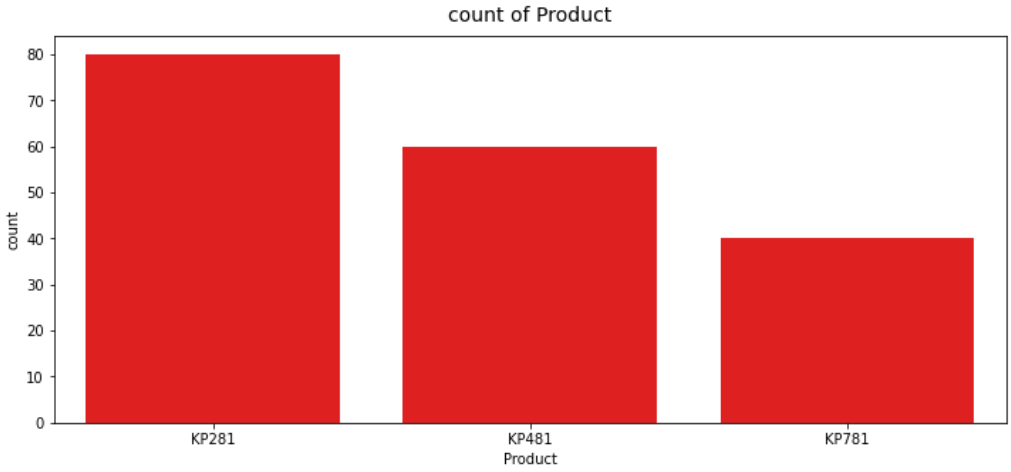
```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x="Age", orient='h', ax=axis[0,0],color ='Green')
sns.boxplot(data=df, x="Education", orient='h', ax=axis[0,1],color ='Green')
sns.boxplot(data=df, x="Usage", orient='h', ax=axis[1,0],color ='Green')
sns.boxplot(data=df, x="Fitness", orient='h', ax=axis[1,1],color ='Green')
sns.boxplot(data=df, x="Income", orient='h', ax=axis[2,0],color ='Green')
sns.boxplot(data=df, x="Miles", orient='h', ax=axis[2,1],color ='Green')
plt.show()
```



categorical columns analyzation

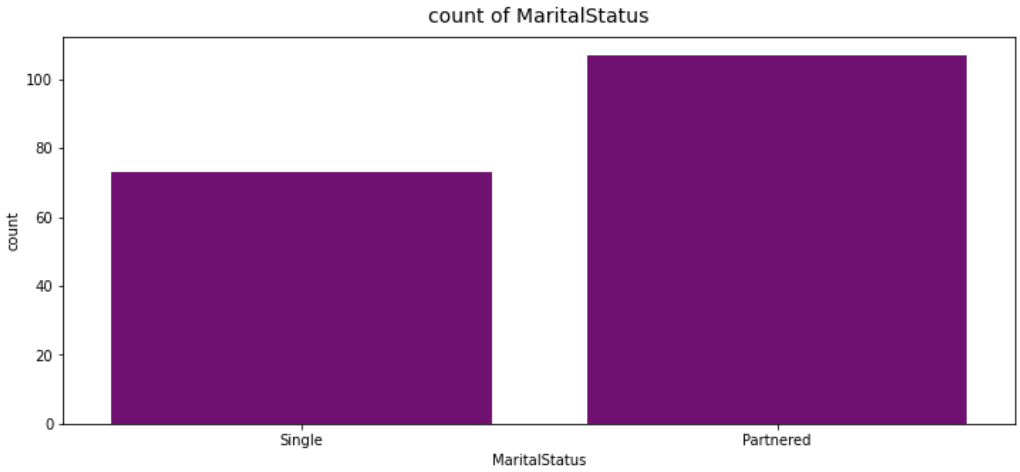
In [12]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Product',color = 'Red')
plt.title("count of Product", pad=10, fontsize=14)
plt.show()
```



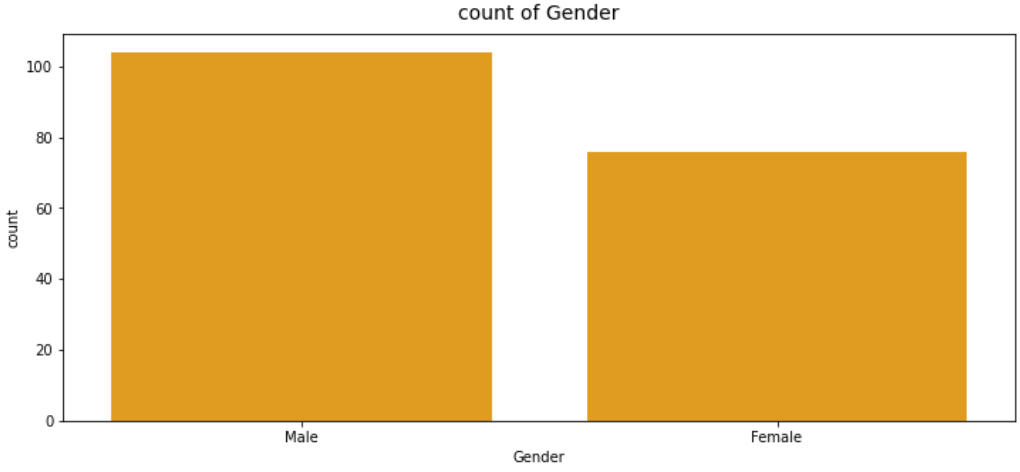
In [18]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='MaritalStatus',color = 'purple')
plt.title("count of MaritalStatus", pad=10, fontsize=14)
plt.show()
```



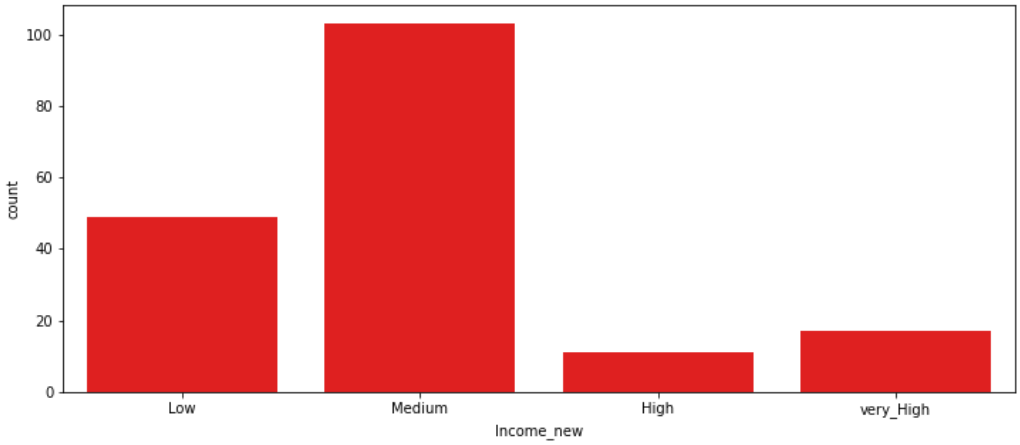
In [13]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Gender',color = 'Orange')
plt.title("count of Gender", pad=10, fontsize=14)
plt.show()
```



In [13]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Income_new',color = 'Red')
plt.show()
```



In [14]:

```
df1 = df[['Product', 'Gender', 'MaritalStatus','Income_new']].melt()
df1.groupby(['variable', 'value'])[['value']].count() / len(df)
```

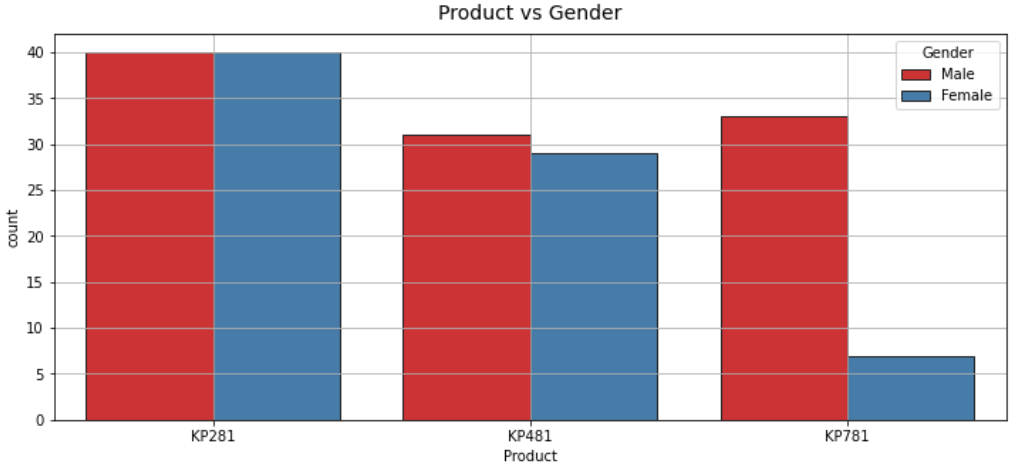
Out[14]:

variable	value	value
Gender	Female	0.422222
	Male	0.577778
Income_new	High	0.061111
	Low	0.272222
	Medium	0.572222
	very_High	0.094444
MaritalStatus	Partnered	0.594444
	Single	0.405556
Product	KP281	0.444444
	KP481	0.333333
	KP781	0.222222

Bivaraite Analysis

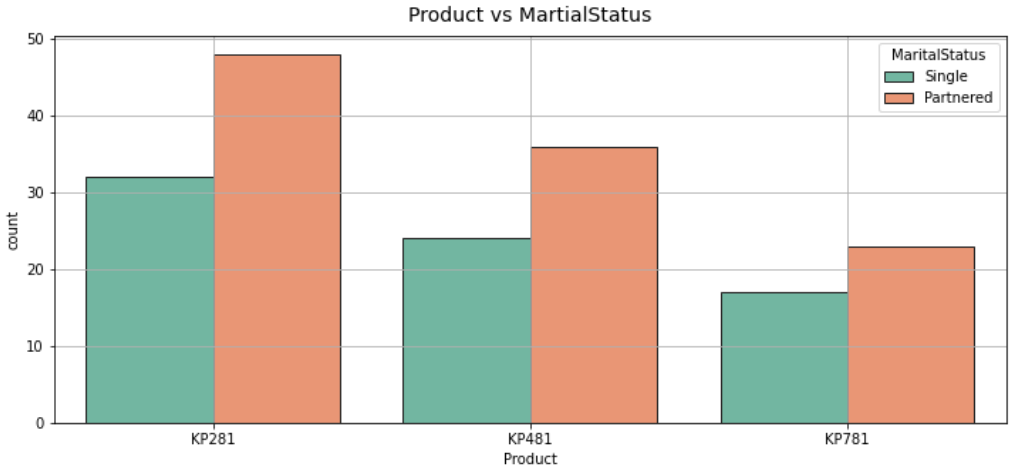
In [21]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Product', hue='Gender', edgecolor="0.15",palette='Set1')
plt.title("Product vs Gender", pad=10, fontsize=14)
plt.grid()
plt.show()
```



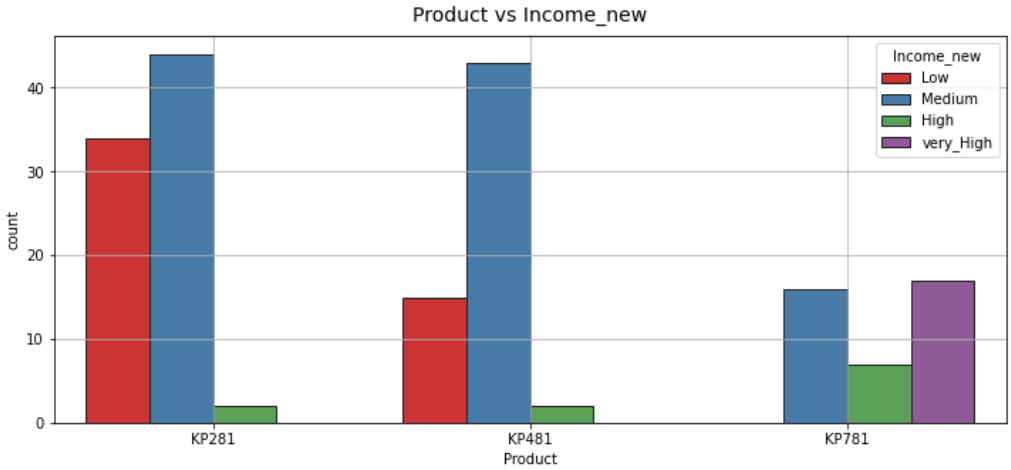
In [25]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Product', hue='MaritalStatus', edgecolor="0.15", palette='Set2')
plt.title("Product vs MartialStatus", pad=10, fontsize=14)
plt.grid()
plt.show()
```



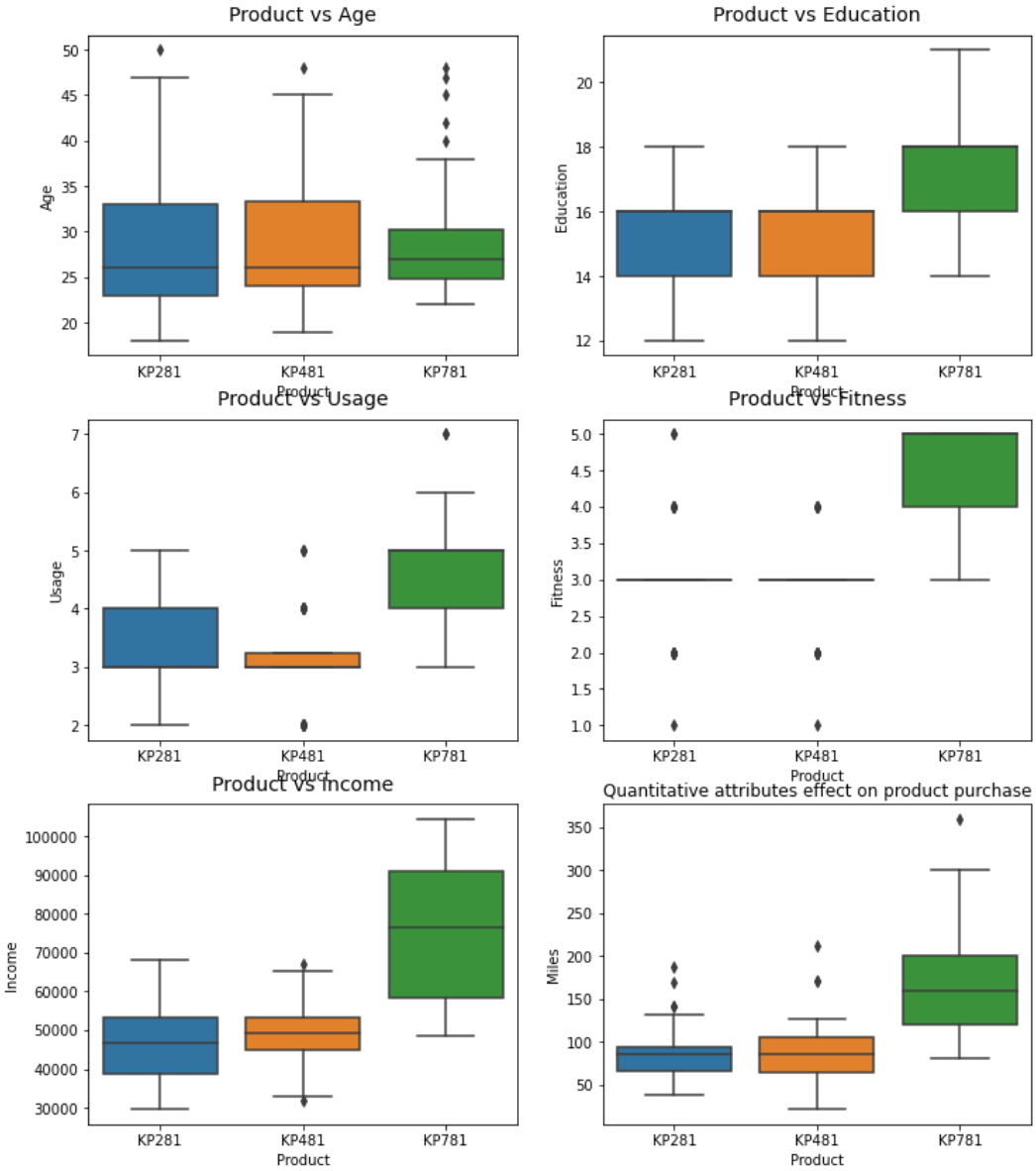
In [15]:

```
plt.figure(figsize=(12,5))
sns.countplot(data=df, x='Product', hue='Income_new', edgecolor="0.15",palette='Set1')
plt.title("Product vs Income_new", pad=10, fontsize=14)
plt.grid()
plt.show()
```



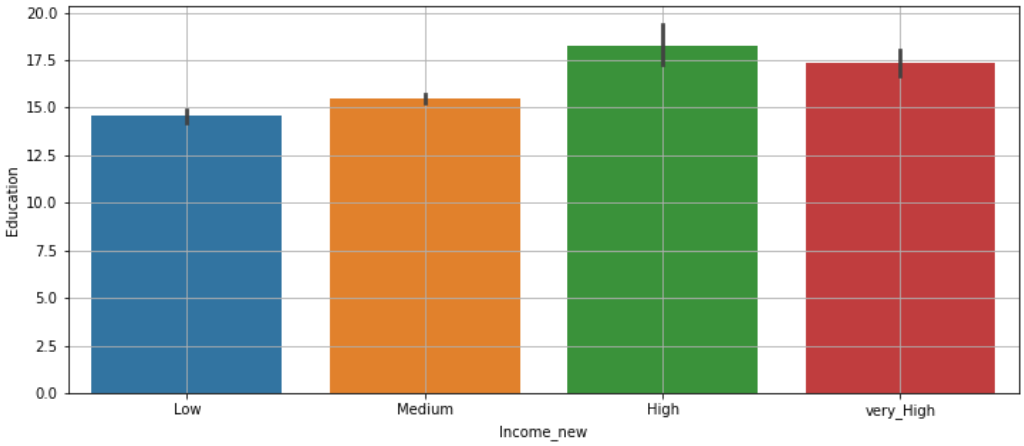
In [4]:

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.2)

sns.boxplot(data=df, x="Product",y = "Age",  ax=axis[0,0])
sns.boxplot(data=df, x="Product",y="Education",  ax=axis[0,1])
sns.boxplot(data=df, x="Product",y="Usage",  ax=axis[1,0])
sns.boxplot(data=df, x="Product",y="Fitness",  ax=axis[1,1])
sns.boxplot(data=df, x="Product",y="Income",  ax=axis[2,0])
sns.boxplot(data=df, x="Product",y="Miles",  ax=axis[2,1])
axis[0,0].set_title("Product vs Age", pad=10, fontsize=14)
axis[0,1].set_title("Product vs Education", pad=10, fontsize=14)
axis[1,0].set_title("Product vs Usage", pad=10, fontsize=14)
axis[1,1].set_title("Product vs Fitness", pad=10, fontsize=14)
axis[2,0].set_title("Product vs Income", pad=10, fontsize=14)
axis[2,1].set_title("Product vs Miles", pad=10, fontsize=14)
plt.title("Quantitative attributes effect on product purchase")
plt.show()
```



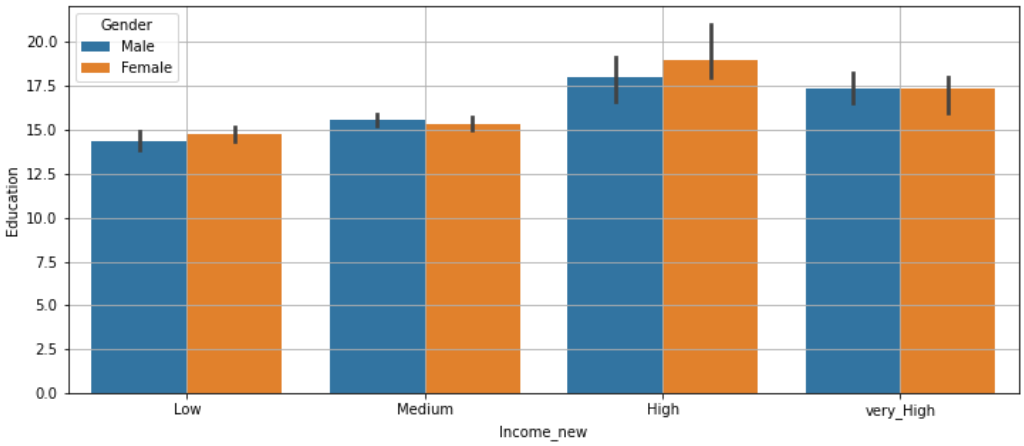
In [9]:

```
plt.figure(figsize=(12,5))
sns.barplot(data=df, x="Income_new",y = "Education")

plt.grid()
plt.show()
```



In [7]:

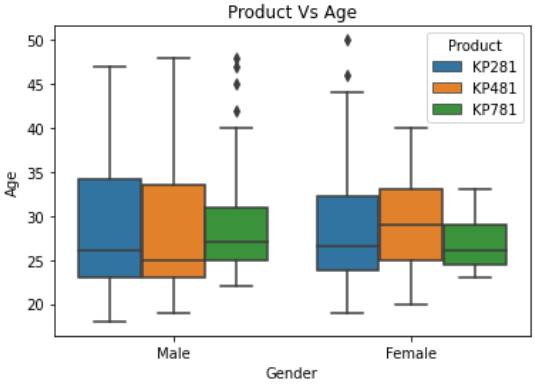
```
plt.figure(figsize=(12,5))
sns.barplot(data=df, x="Income_new",y = "Education",hue='Gender')

plt.grid()
plt.show()
```



Multivariate Analysis

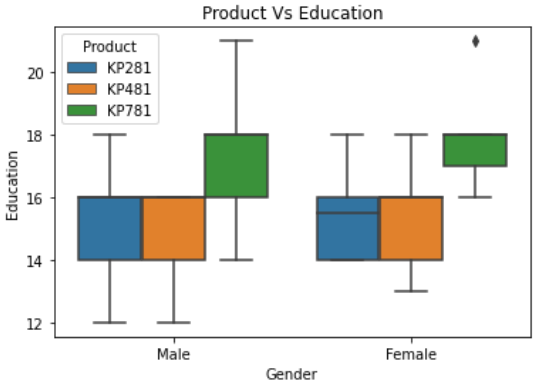
In [38]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Age",
            hue="Product")
plt.title("Product Vs Age")
plt.show()
```



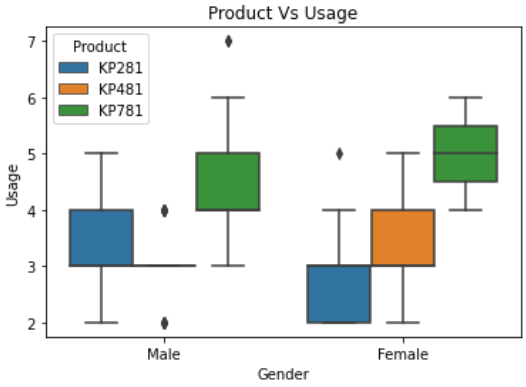
In [39]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Education",
            hue="Product")
plt.title("Product Vs Education")
plt.show()
```



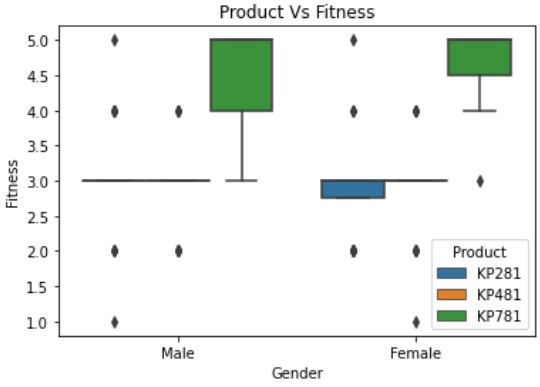
In [40]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Usage",
            hue="Product")
plt.title("Product Vs Usage")
plt.show()
```



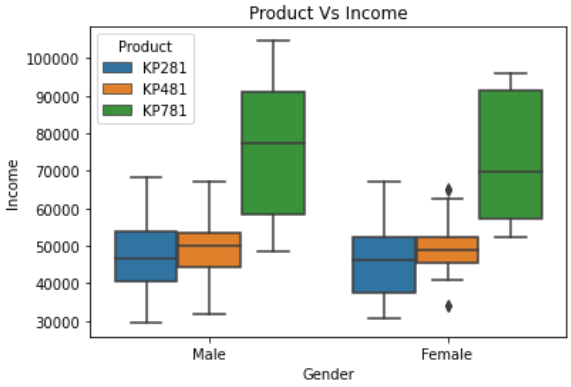
In [41]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Fitness",
            hue="Product")
plt.title("Product Vs Fitness")
plt.show()
```



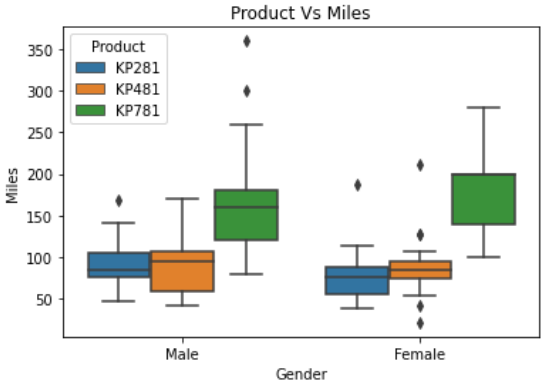
In [42]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Income",
            hue="Product")
plt.title("Product Vs Income")
plt.show()
```



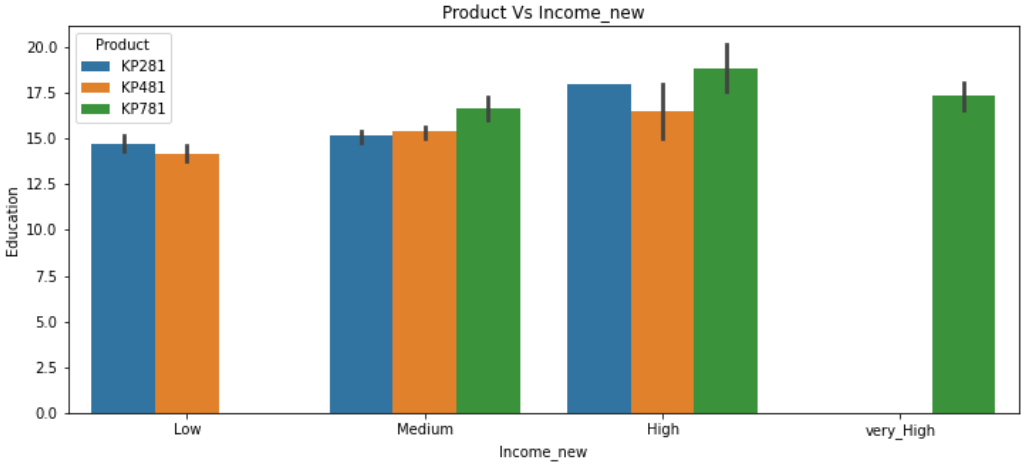
In [43]:

```
sns.boxplot(data=df,
            x="Gender",
            y="Miles",
            hue="Product")
plt.title("Product Vs Miles")
plt.show()
```



In [25]:

```
plt.figure(figsize=(12,5))
sns.barplot(data=df,
            x="Income_new",
            y="Education",
            hue="Product")
plt.title("Product Vs Income_new")
plt.show()
```



For correlation: Heatmaps, Pairplots

In [26]:

```
df.corr()
```

Out[26]:

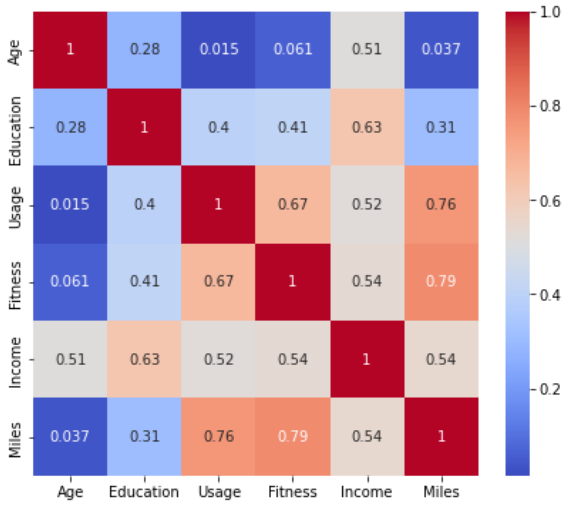
	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414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

In [5]:

```
plt.figure(figsize=(7,6))
sns.heatmap(df.corr(), cmap="coolwarm", annot=True)
```

Out[5]:

```
<AxesSubplot:>
```

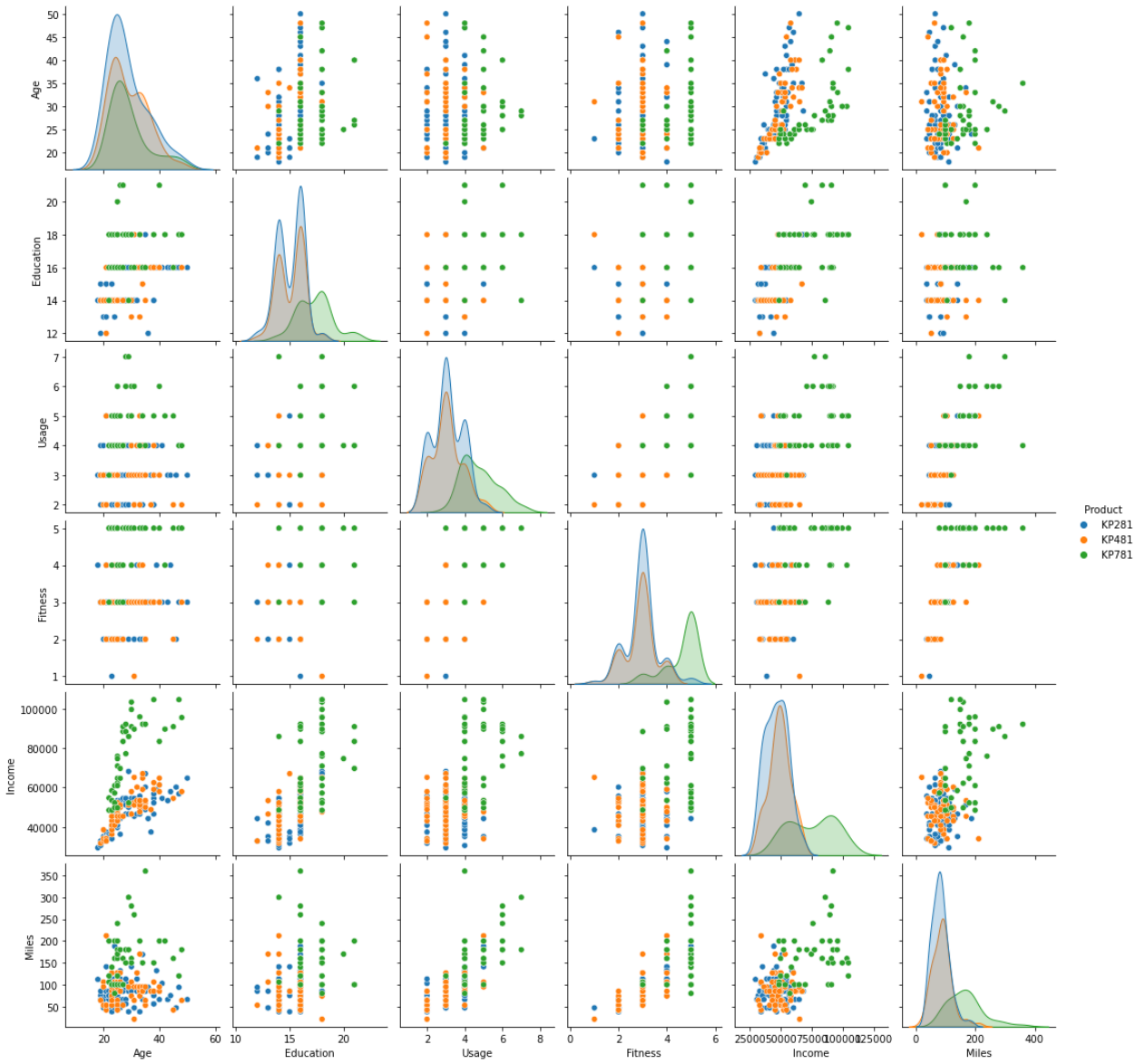


In [17]:

```
sns.pairplot(df, hue ='Product')
```

Out[17]:

```
<seaborn.axisgrid.PairGrid at 0xe65bd33f40>
```



Computing Marginal & Conditional Probabilitie

In [25]:

```
pd.crosstab(df['Product'],[df['Gender']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[25]:

Gender	Female	Male	Total
Product			
KP281	0.22	0.22	0.44
KP481	0.16	0.17	0.33
KP781	0.04	0.18	0.22
Total	0.42	0.58	1.00

In [4]:

```
pd.crosstab(index=df["Product"],
            columns=df["Gender"],
            margins=True)
```

Out[4]:

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

In [28]:

```
pd.crosstab(df['Product'],[df['MaritalStatus']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[28]:

MaritalStatus	Partnered	Single	Total
Product			
KP281	0.27	0.18	0.44
KP481	0.20	0.13	0.33
KP781	0.13	0.09	0.22
Total	0.59	0.41	1.00

In [29]:

```
pd.crosstab(df['Product'],[df['Fitness']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[29]:

Fitness	1	2	3	4	5	Total
Product						
KP281	0.01	0.08	0.30	0.05	0.01	0.44
KP481	0.01	0.07	0.22	0.04	0.00	0.33
KP781	0.00	0.00	0.02	0.04	0.16	0.22
Total	0.01	0.14	0.54	0.13	0.17	1.00

In [30]:

```
pd.crosstab(df['Product'],[df['Education']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[30]:

Education	12	13	14	15	16	18	20	21	Total
Product									
KP281	0.01	0.02	0.17	0.02	0.22	0.01	0.00	0.00	0.44
KP481	0.01	0.01	0.13	0.01	0.17	0.01	0.00	0.00	0.33
KP781	0.00	0.00	0.01	0.00	0.08	0.11	0.01	0.02	0.22
Total	0.02	0.03	0.31	0.03	0.47	0.13	0.01	0.02	1.00

In [27]:

```
pd.crosstab(df['Income_new'],[df['Education']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[27]:

Education	12	13	14	15	16	18	20	21	Total
Income_new									
Low	0.02	0.02	0.13	0.02	0.09	0.00	0.00	0.00	0.27
Medium	0.00	0.01	0.17	0.00	0.35	0.04	0.00	0.00	0.57
High	0.00	0.00	0.00	0.01	0.01	0.03	0.01	0.01	0.06
very_High	0.00	0.00	0.01	0.00	0.03	0.06	0.00	0.01	0.09
Total	0.02	0.03	0.31	0.03	0.47	0.13	0.01	0.02	1.00

In [9]:

```
pd.crosstab(index=df["Income_new"],
            columns=df["Education"],
            margins=True)
```

Out[9]:

Income_new / Education	12	13	14	15	16	18	20	21	All
Low	3	3	23	4	16	0	0	0	49
Medium	0	2	31	0	63	7	0	0	103
High	0	0	0	1	1	6	1	2	11
very_High	0	0	1	0	5	10	0	1	17
All	3	5	55	5	85	23	1	3	180

In [6]:

```
pd.crosstab(df['Income_new'],[df['Product']], normalize=True, margins=True, margins_name='Total').round(2)
```

Out[6]:

Income_new / Product	KP281	KP481	KP781	Total
Low	0.19	0.08	0.00	0.27
Medium	0.24	0.24	0.09	0.57
High	0.01	0.01	0.04	0.06
very_High	0.00	0.00	0.09	0.09
Total	0.44	0.33	0.22	1.00

In [7]:

```
pd.crosstab(index=df["Income_new"],
            columns=df["Product"],
            margins=True)
```

Out[7]:

Income_new / Product	KP281	KP481	KP781	All
Low	34	15	0	49
Medium	44	43	16	103
High	2	2	7	11
very_High	0	0	17	17
All	80	60	40	180

Marginal Probability

In [24]:

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

Out[24]:

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

In [ ]:

```

```