

```
!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, 16.1MB/s]
```

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

```
df = pd.read_csv('aerofit_treadmill.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

Next steps:

Generate code with df

View recommended plots

New interactive sheet

```
df.shape
```

```
(180, 9)
```


```
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
```

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

	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


```
df.dtypes
```





		0
<b>Product</b>		object
<b>Age</b>		int64
<b>Gender</b>		object
<b>Education</b>		int64
<b>MaritalStatus</b>		object
<b>Usage</b>		int64
<b>Fitness</b>		int64
<b>Income</b>		int64
<b>Miles</b>		int64

**dtype:** object


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



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




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



		count
<b>Gender</b>		
<b>Male</b>		104
<b>Female</b>		76

**dtype:** int64


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



		count
<b>Product</b>		
<b>KP281</b>		80
<b>KP481</b>		60
<b>KP781</b>		40

**dtype:** int64

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



		count
<b>MaritalStatus</b>		
<b>Partnered</b>		107
<b>Single</b>		73

**dtype:** int64

```
df.nunique()
```

	0
<b>Product</b>	3
<b>Age</b>	32
<b>Gender</b>	2
<b>Education</b>	8
<b>MaritalStatus</b>	2
<b>Usage</b>	6
<b>Fitness</b>	5
<b>Income</b>	62
<b>Miles</b>	37

dtype: int64

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

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

```
df["Gender"].unique()
```

```
array(['Male', 'Female'], dtype=object)
```

```
df["MaritalStatus"].unique()
```

```
array(['Single', 'Partnered'], dtype=object)
```

```
sns.distplot(df['Age'])
```

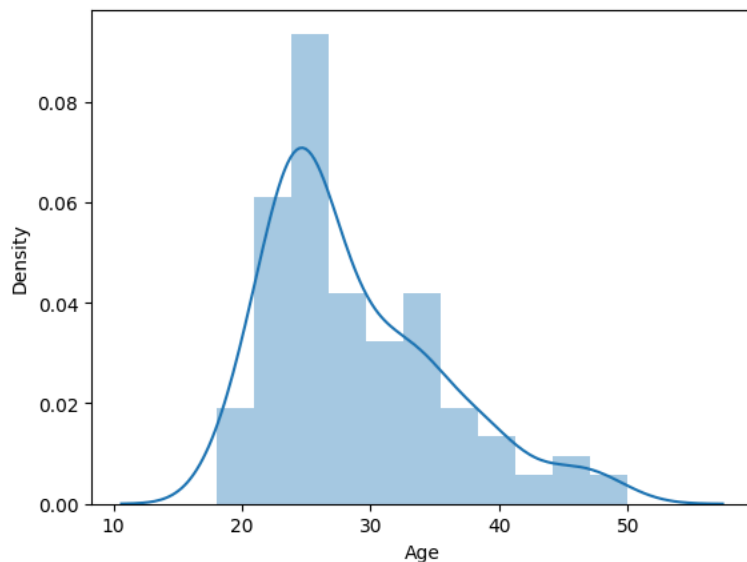
```
/tmp/ipython-input-3255828239.py:1: UserWarning:
```

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



```
sns.distplot(df['Income'])
```

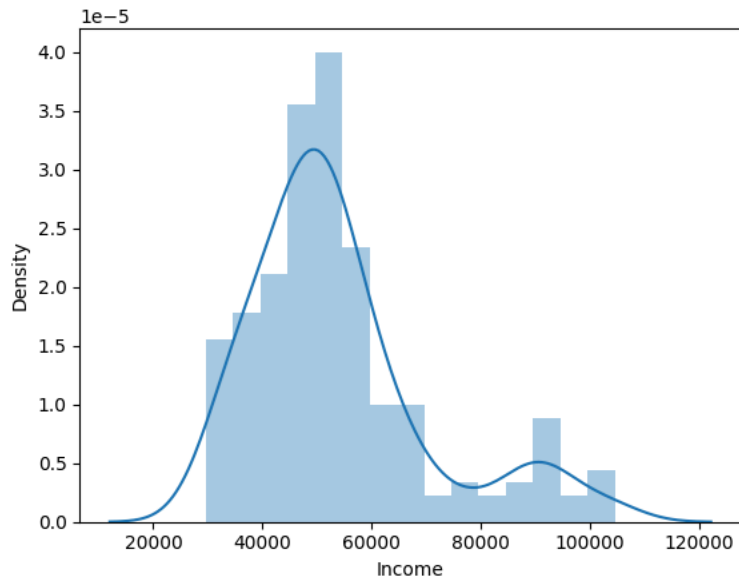
 /tmp/ipython-input-1426022472.py:1: UserWarning:

`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).

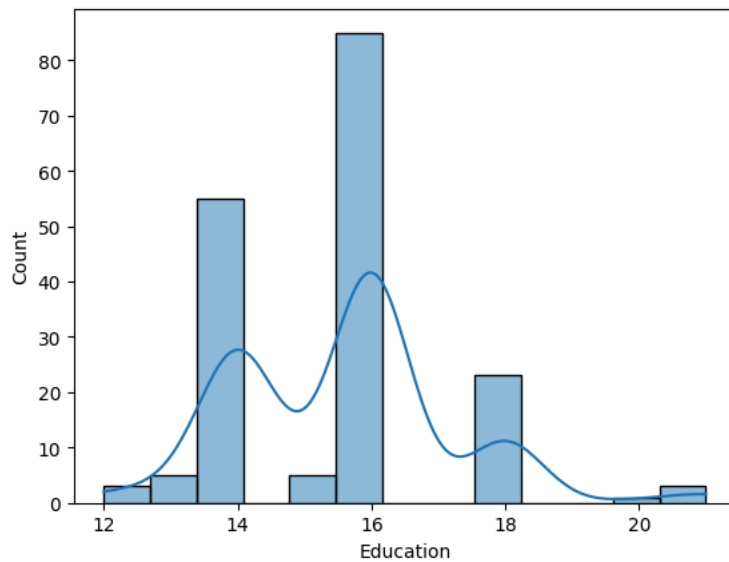
For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot(df['Income'])  
<Axes: xlabel='Income', ylabel='Density'>
```



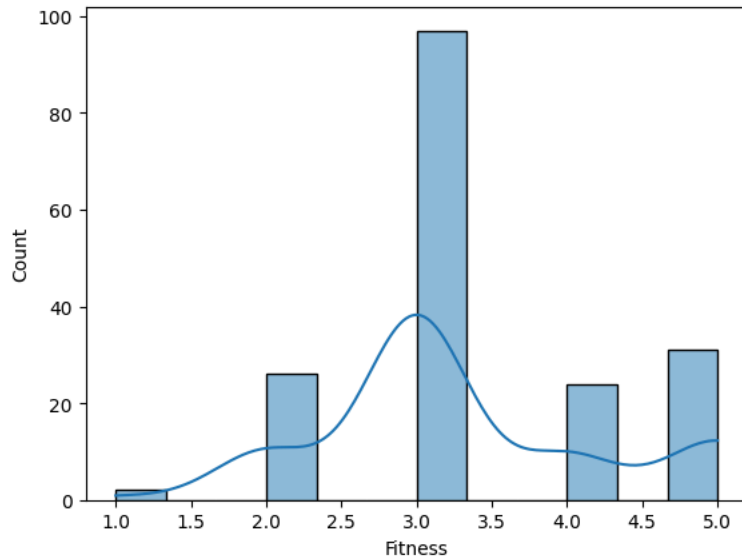
```
sns.histplot(df['Education'], kde=True)
```

 <Axes: xlabel='Education', ylabel='Count'>



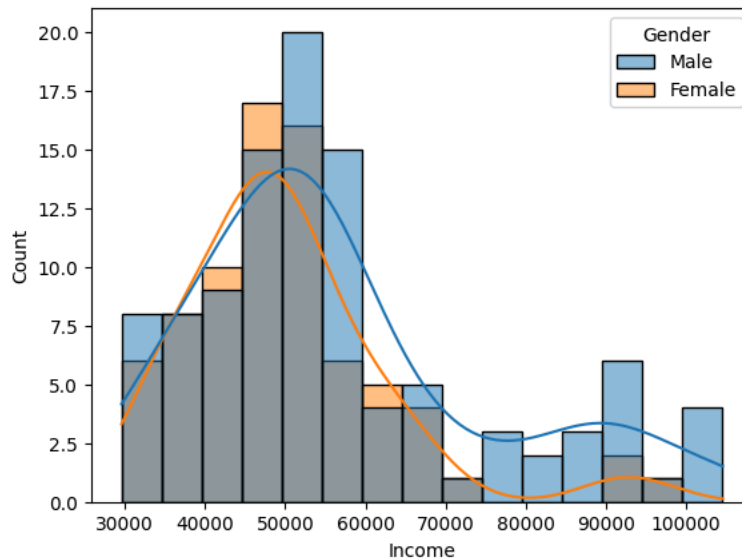
```
sns.histplot(df['Fitness'], kde=True)
```

<Axes: xlabel='Fitness', ylabel='Count'>



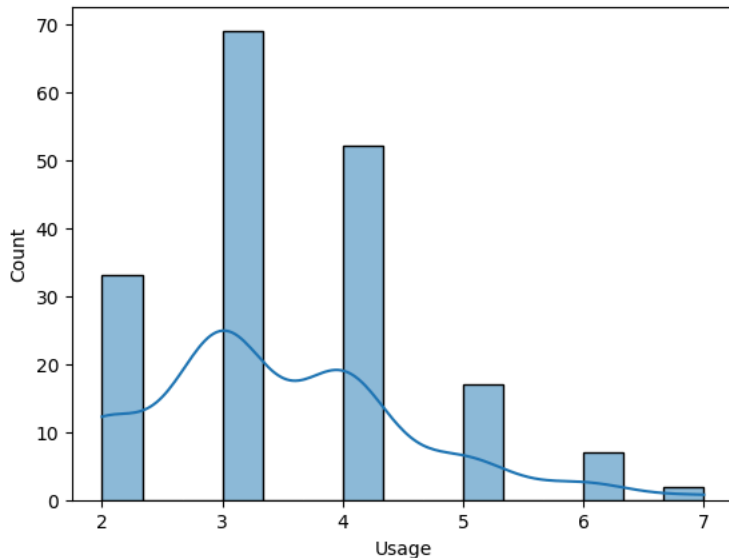
```
sns.histplot(x="Income", kde=True, data=df, hue="Gender")
```

<Axes: xlabel='Income', ylabel='Count'>



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

<Axes: xlabel='Usage', ylabel='Count'>



```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(15, 5))
sns.countplot(data=df, x='Product', ax=axs[0], palette="coolwarm")
sns.countplot(data=df, x='Gender', ax=axs[1], palette="Set1")
sns.countplot(data=df, x='MaritalStatus', ax=axs[2], palette="Set2")
```

```
axs[0].set_title('Product-Counts')
axs[1].set_title('Gender-Counts')
axs[2].set_title('MaritalStatus-Counts')
```

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

/tmp/ipython-input-2520653886.py:2: 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`

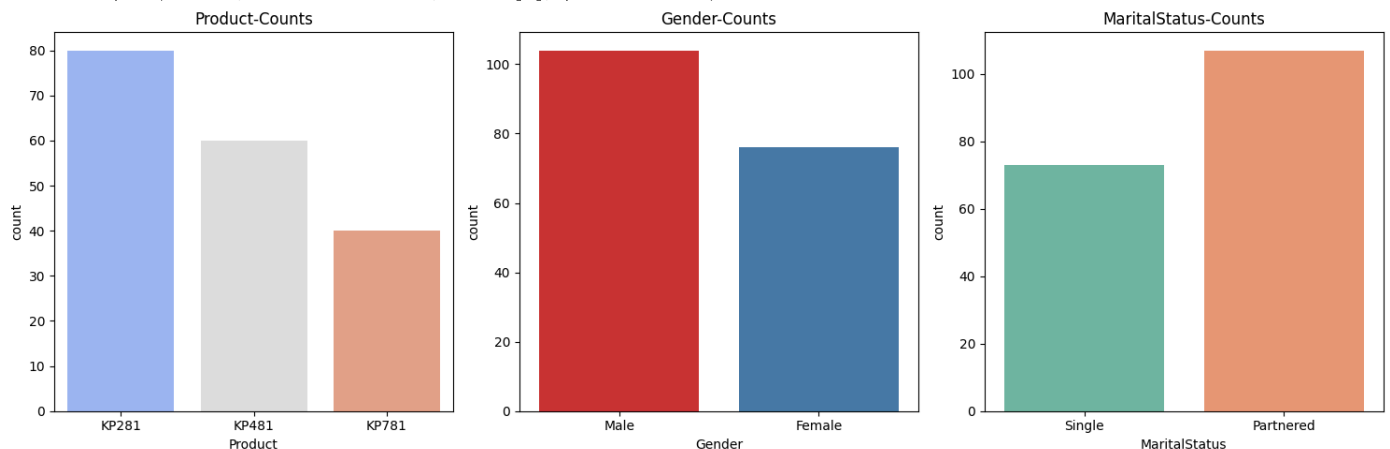
```
sns.countplot(data=df, x='Product', ax=axs[0], palette="coolwarm")
/tmp/ipython-input-2520653886.py:3: FutureWarning:
```

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

```
sns.countplot(data=df, x='Gender', ax=axs[1], palette="Set1")
/tmp/ipython-input-2520653886.py:4: 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`

```
sns.countplot(data=df, x='MaritalStatus', ax=axs[2], palette="Set2")
```

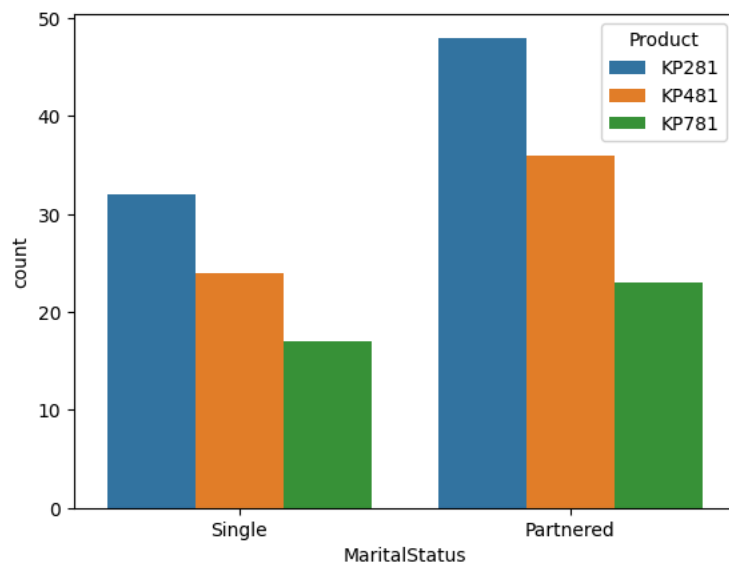


## ✓ Observations:

1. KP281 is the most frequent product.
2. There are more Males in the data than Females.
3. More Partnered persons are there in the data.

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

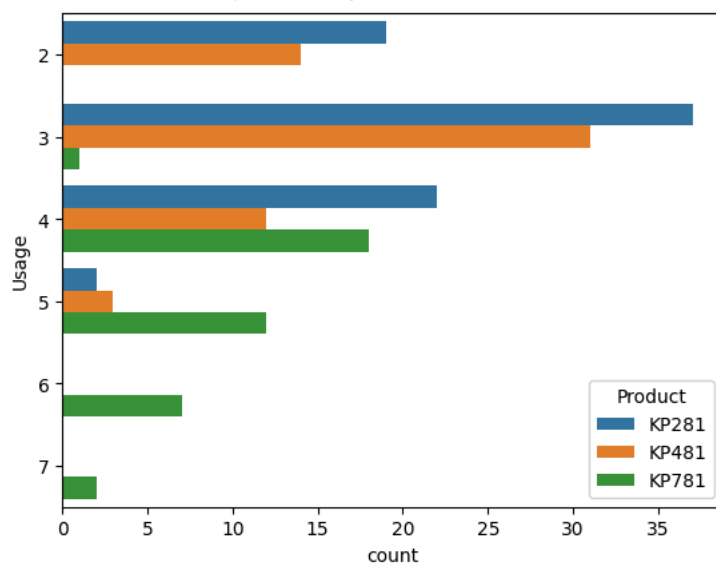
<Axes: xlabel='MaritalStatus', ylabel='count'>



- ✓ 1. KP281 is the best pick for both Partnered and Singles.

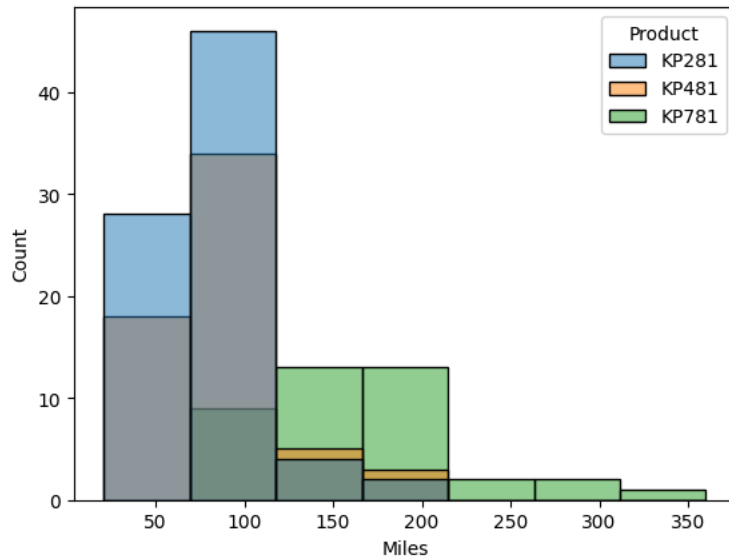
```
sns.countplot(data=df, y='Usage', hue="Product" )
```

<Axes: xlabel='count', ylabel='Usage'>



```
sns.histplot(data=df, x='Miles', hue="Product", bins=7)
```

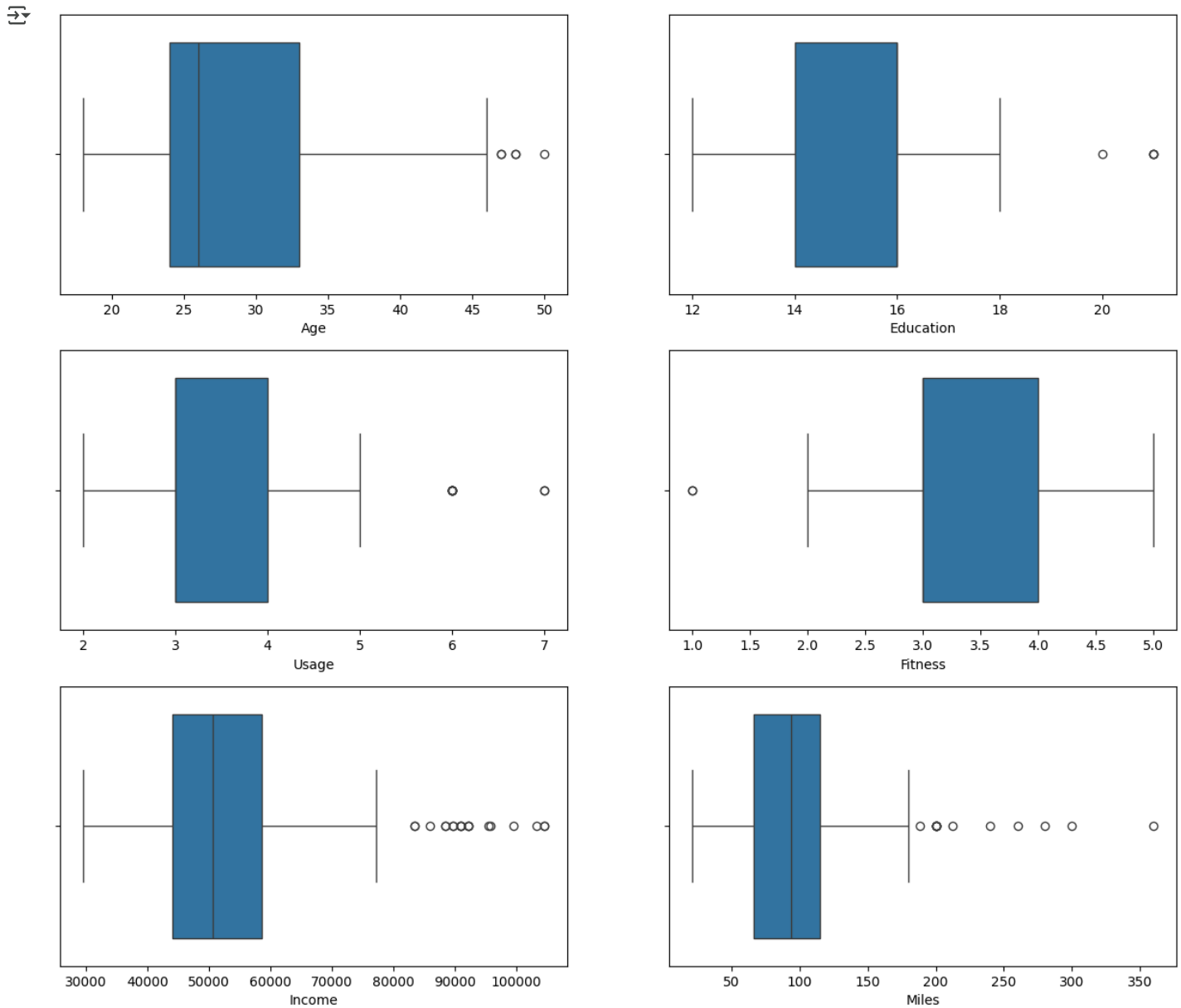
<Axes: xlabel='Miles', ylabel='Count'>



```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(15, 10))
fig.subplots_adjust(top=1.1)
```

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






✓ 1. Even from the boxplot it is quit clear that:

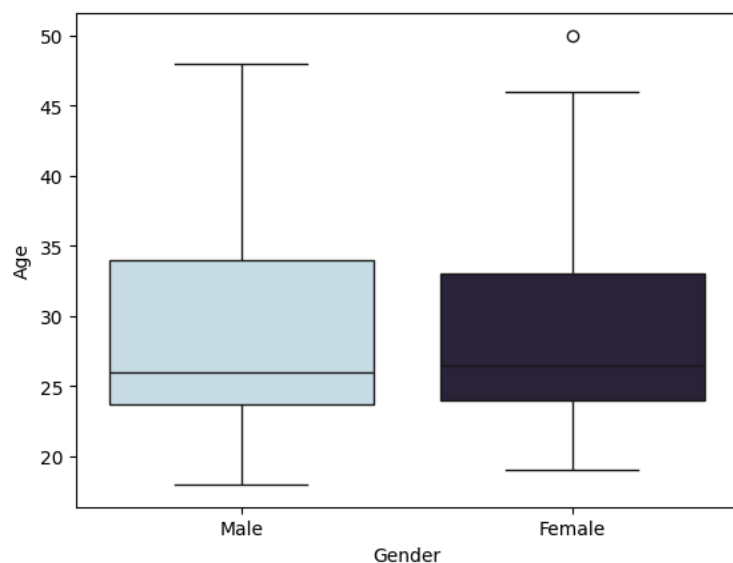
Age, Education and Usage are having very few outliers while Income and Miles are having more outliers.

```
sns.boxplot(data=df, y="Age", x="Gender", palette='ch:s=.25,rot=-.25', legend=False)
```


 /tmp/ipython-input-3764614881.py: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`

```
sns.boxplot(data=df, y="Age", x="Gender", palette='ch:s=.25,rot=-.25', legend=False)
<Axes: xlabel='Gender', ylabel='Age'>
```

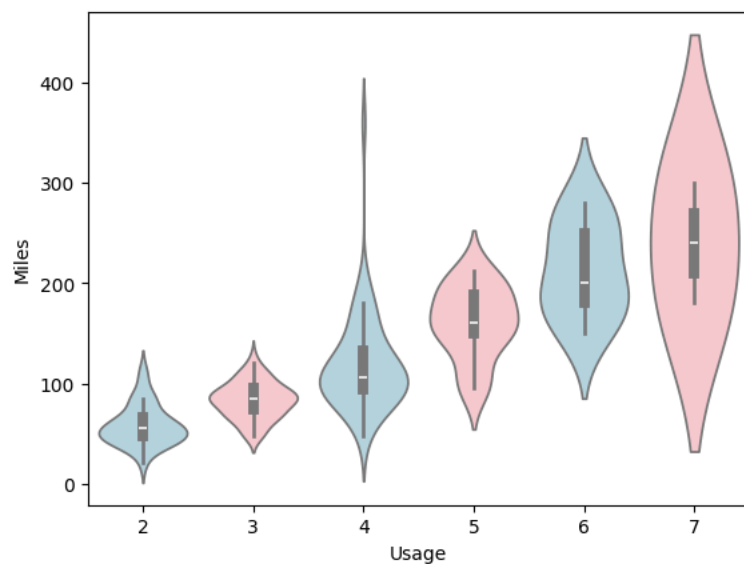


```
sns.violinplot(data=df, x="Usage", y="Miles", palette=["lightblue","pink"])
```


 /tmp/ipython-input-3477126622.py: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`

```
sns.violinplot(data=df, x="Usage", y="Miles", palette=["lightblue","pink"])
/tmp/ipython-input-3477126622.py:1: UserWarning:
The palette list has fewer values (2) than needed (6) and will cycle, which may produce an uninterpretable plot.
sns.violinplot(data=df, x="Usage", y="Miles", palette=["lightblue","pink"])
<Axes: xlabel='Usage', ylabel='Miles'>
```

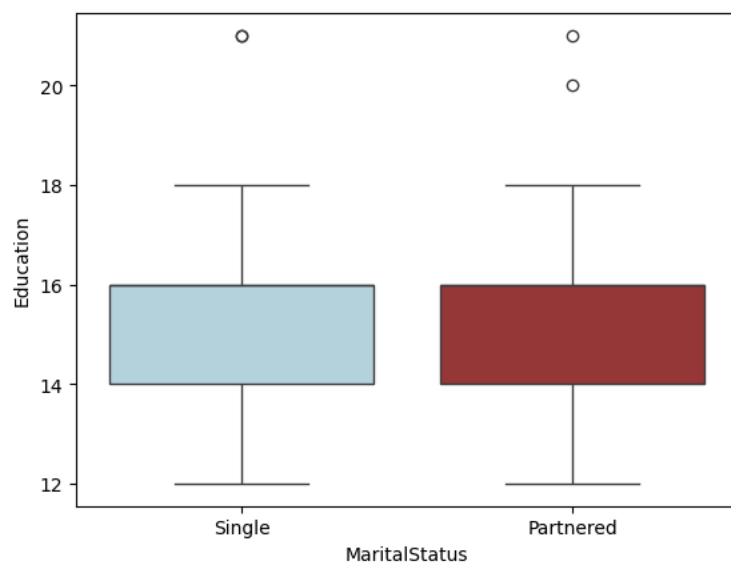


```
sns.boxplot(data=df, x="MaritalStatus", y="Education", palette=["lightblue","brown"])
```

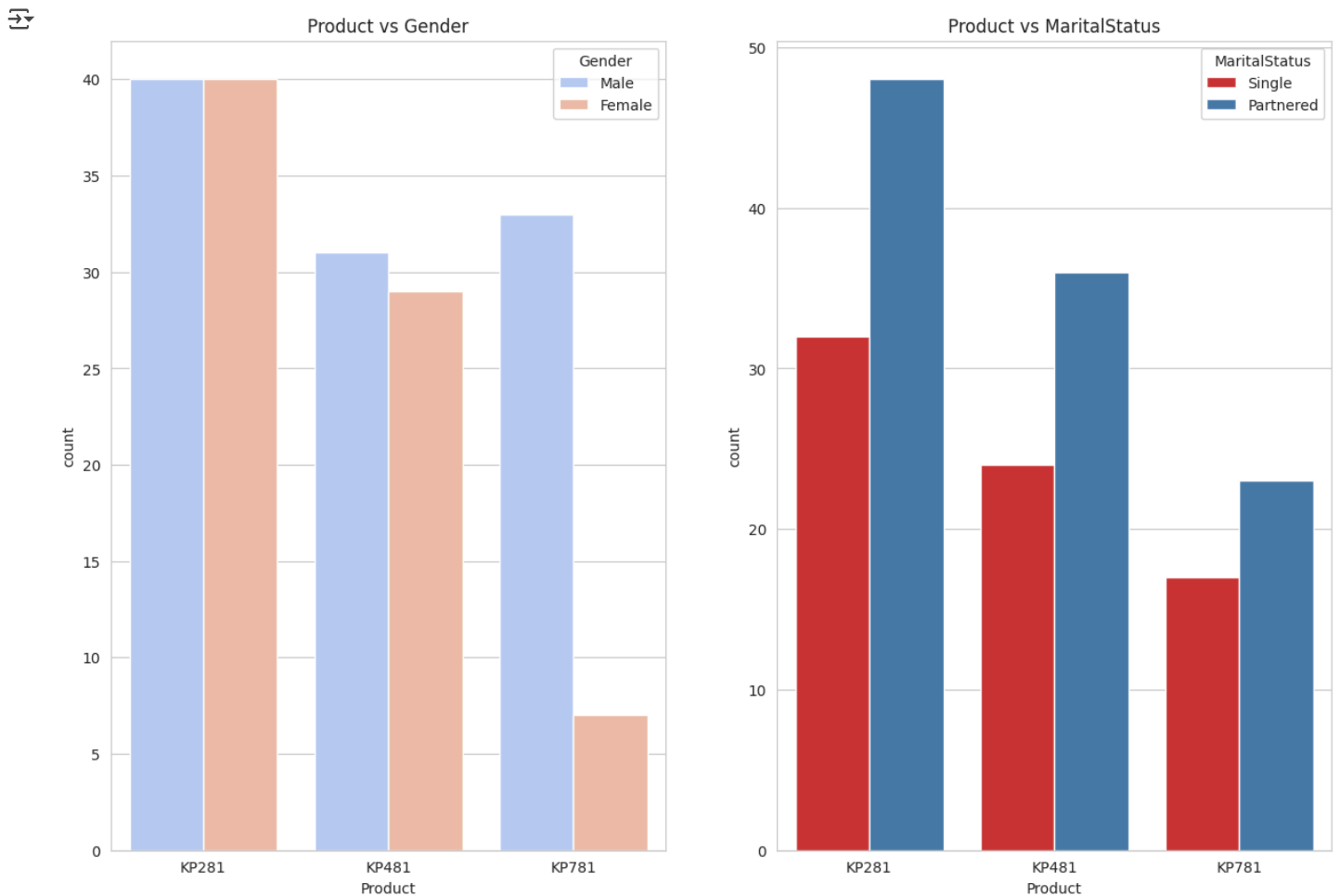
 /tmp/ipython-input-2656843954.py:1: FutureWarning:

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

```
sns.boxplot(data=df, x="MaritalStatus", y="Education", palette=["lightblue", "brown"])
<Axes: xlabel='MaritalStatus', ylabel='Education'>
```



```
sns.set_style(style="whitegrid")
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(15, 10))
sns.countplot(data=df, x='Product', hue="Gender", ax=axs[0], palette="coolwarm")
sns.countplot(data=df, x='Product', hue="MaritalStatus", ax=axs[1], palette="Set1")
axs[0].set_title("Product vs Gender")
axs[1].set_title("Product vs MaritalStatus")
plt.show()
```



## Observations

### Product vs Gender

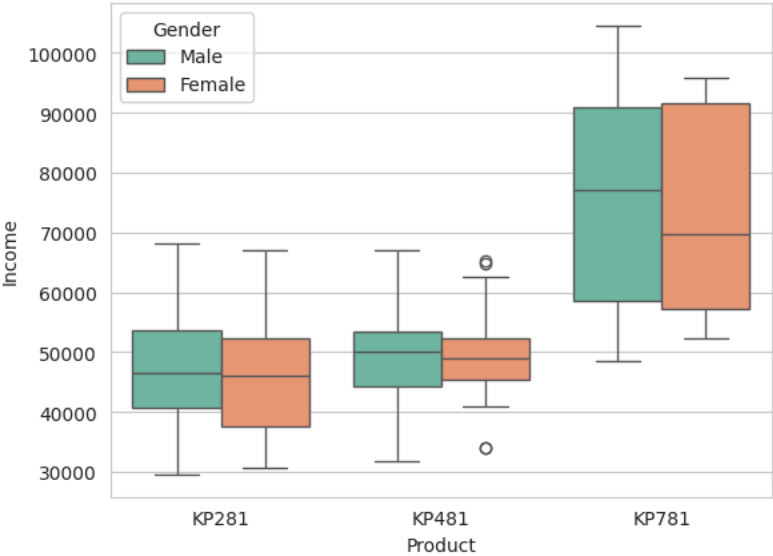
Equal number if males and females have purchased KP281 product and almost same for the product KP481. Most of the males have purchased the product KP781.

### Product vs MaritalStatus:



Customers who is Partnered, is more likely to purchase the product.

```
sns.boxplot(data=df, x="Product", y="Income", hue="Gender", palette="Set2")
```


 <Axes: xlabel='Product', ylabel='Income'>





```
pd.crosstab(index=df.Product,columns=df.Gender,margins=True,normalize='index')
```


Gender	Female	Male
Product		
KP281	0.500000	0.500000
KP481	0.483333	0.516667
KP781	0.175000	0.825000
All	0.422222	0.577778



```
pd.crosstab(index=df.Product,columns=df.MaritalStatus,margins=True,normalize='index')
```

MaritalStatus	Partnered	Single
Product		
KP281	0.600000	0.400000
KP481	0.600000	0.400000
KP781	0.575000	0.425000
All	0.594444	0.405556

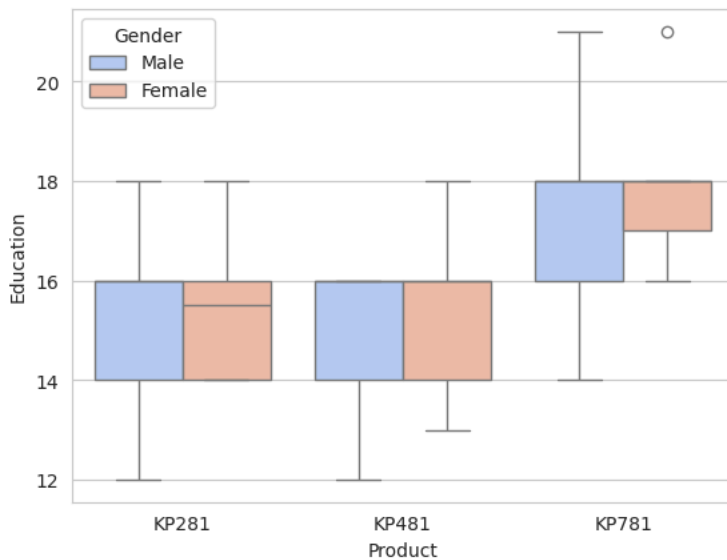


```
sns.boxplot(data=df, x="MaritalStatus", y="Miles", hue="Fitness", palette="coolwarm")
```

```
<Axes: xlabel='MaritalStatus', ylabel='Miles'>
```

```
sns.boxplot(data=df, y="Education", x="Product", hue="Gender", palette="coolwarm")
```

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



```
attrs = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
sns.set_style("white")
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(15, 10))
fig.subplots_adjust(top=1.2)
count = 0
for i in range(2):
    for j in range(3):
        sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
        count += 1
plt.show()
```

```
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
/tmp/ipython-input-2518129669.py:8: FutureWarning:
```

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

```
sns.boxplot(data=df, x="Product", y=attrs[count], ax=axs[i,j], palette="coolwarm")
```

