```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data_path = "sample_data/aerofit_treadmill.csv"
df = pd.read_csv(data_path)
display(df)
```

	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

Next steps: Generate code with df View recommended plots

```
print(f" Number of rows: {df.shape[0]}\nNumber of columns: {df.shape[1]} " )
```

```
 Number of rows: 180
Number of columns: 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

```
print('\nColumns with missing value:')
print(df.isnull().any())
```

```
Columns with missing value:
Product          False
Age              False
Gender           False
Education        False
MaritalStatus    False
Usage            False
Fitness          False
Income           False
Miles            False
dtype: bool
```

Observations:

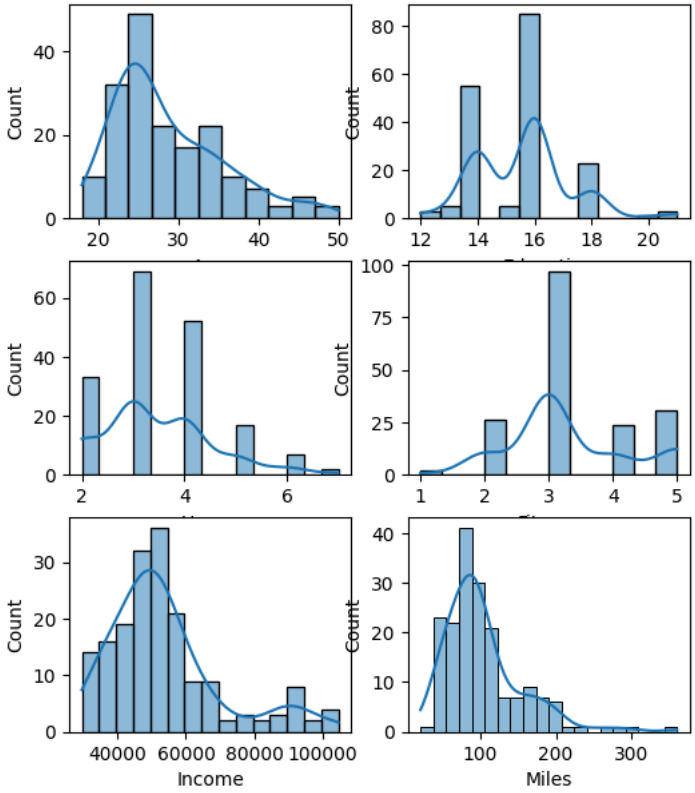
- There are no missing values within the dataset.
- The dataset contains 3 distinct products.
- KP281 stands out as the most frequently occurring product.
- The age range spans from 18 to 50 years, with an average age of 28.79. Additionally, 75% of individuals in the dataset are aged 33 or younger.
- A significant portion of the population, constituting 75%, possesses 16 years of education or less.
- Among the 180 data points, 104 individuals are identified as Male, while the remaining are Female.
- Notably, both Income and Miles exhibit considerably high standard deviations, suggesting the presence of potential outliers within these variables.

Univariate Analysis:

Understanding the distribution of the data for the quantitative atributes:

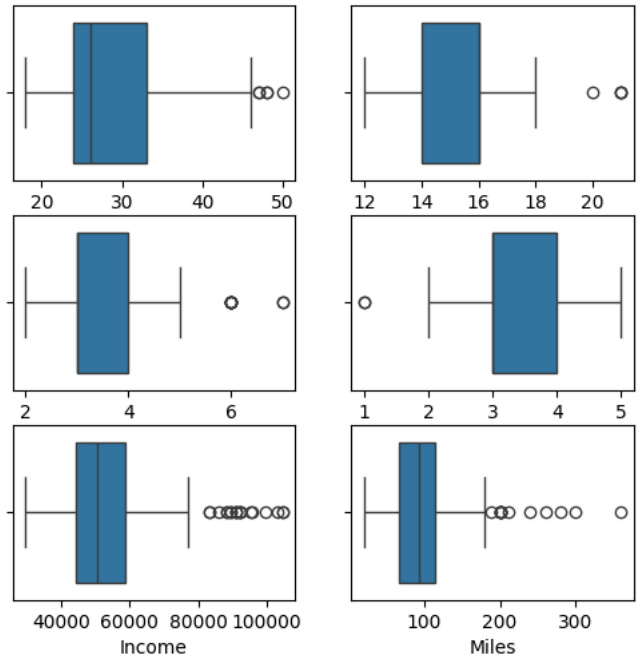
1. Age
2. Education
3. Usage
4. Fitness
5. Income
6. Miles

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(6, 5))
fig.subplots_adjust(top=1.2)
sns.histplot(data=df, x="Age", kde=True, ax=axis[0,0])
sns.histplot(data=df, x="Education", kde=True, ax=axis[0,1])
sns.histplot(data=df, x="Usage", kde=True, ax=axis[1,0])
sns.histplot(data=df, x="Fitness", kde=True, ax=axis[1,1])
sns.histplot(data=df, x="Income", kde=True, ax=axis[2,0])
sns.histplot(data=df, x="Miles", kde=True, ax=axis[2,1])
plt.show()
```



Outliers detection using BoxPlots

```
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(6, 5))
fig.subplots_adjust(top=1.0)
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()
```



Observations:

From the boxplots, it is evident that:

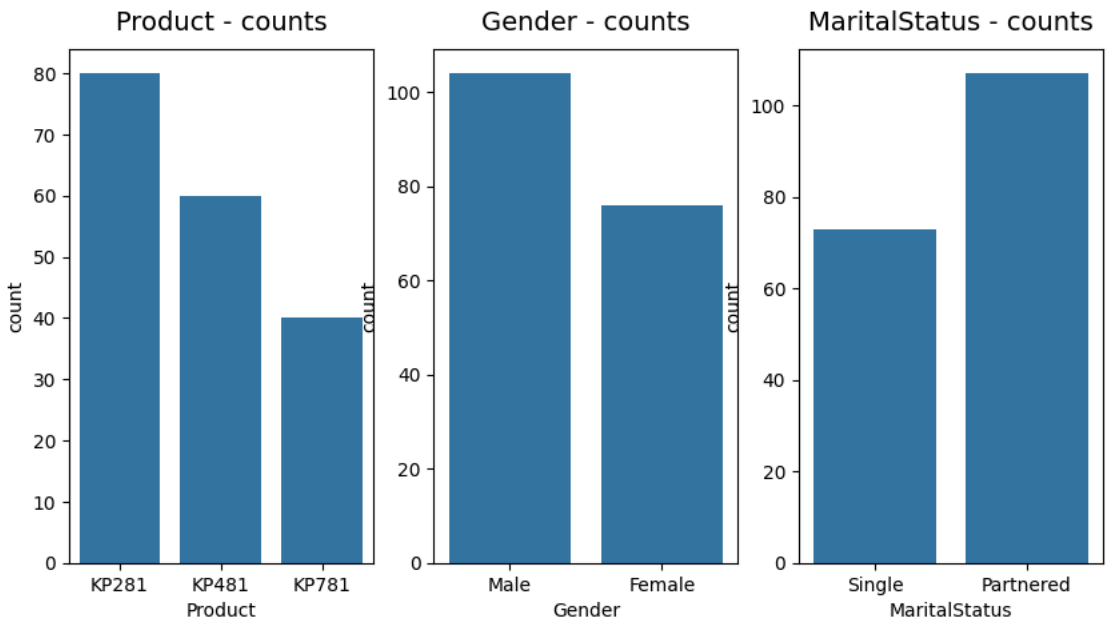
- Age, Education, and Usage have very few outliers.

- Income and Miles exhibit more outliers.

Understanding the distribution of the data for the qualitative attributes:

1. Product
2. Gender
3. MaritalStatus

```
fig, axs = plt.subplots(nrows=1, ncols=3, figsize=(10,5))
sns.countplot(data=df, x='Product', ax=axs[0])
sns.countplot(data=df, x='Gender', ax=axs[1])
sns.countplot(data=df, x='MaritalStatus', ax=axs[2])
axs[0].set_title("Product - counts", pad=10, fontsize=14)
axs[1].set_title("Gender - counts", pad=10, fontsize=14)
axs[2].set_title("MaritalStatus - counts", pad=10,fontsize=14)
plt.show()
```



Observations:

- KP281 is the most frequent product in the dataset.
- There are more males than females in the data.
- There is a higher number of partnered individuals in the data.

normalized count for each variable is shown below:

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

		value
variable	**value**	
Gender	**Female**	0.422222
	Male	0.577778
MaritalStatus	**Partnered**	0.594444
	Single	0.405556
Product	**KP281**	0.444444
	KP481	0.333333
	KP781	0.222222

Observations

Product

- 44.44% of the customers have purchased the KP2821 product.

- 33.33% of the customers have purchased the KP481 product.
- 22.22% of the customers have purchased the KP781 product.

Gender

- 57.78% of the customers are Male.

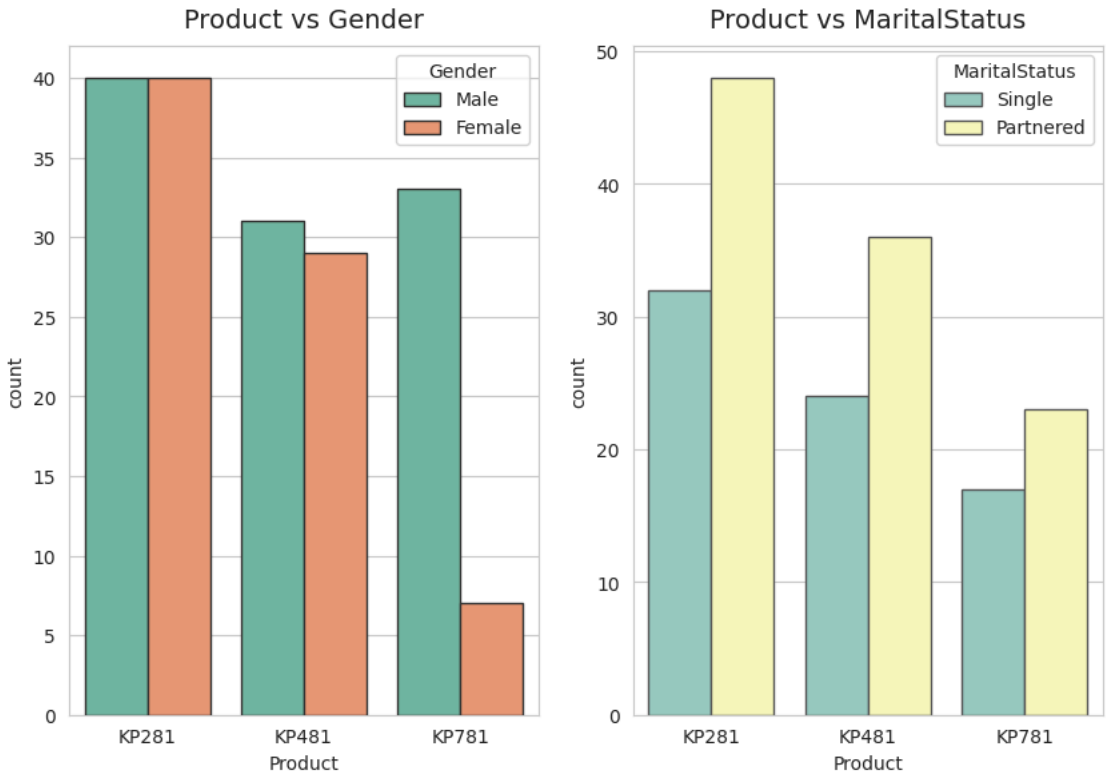
MaritalStatus

- 59.44% of the customers are Partnered.

Bivariate Analysis:

Checking if features - Gender or MaritalStatus have any effect on the product purchased.

```
sns.set_style(style='whitegrid')
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(10, 6.5))
sns.countplot(data=df, x='Product', hue='Gender', edgecolor="0.20",
palette='Set2', ax=axs[0])
sns.countplot(data=df, x='Product', hue='MaritalStatus',
edgecolor="0.35", palette='Set3', ax=axs[1])
axs[0].set_title("Product vs Gender", pad=10, fontsize=14)
axs[1].set_title("Product vs MaritalStatus", pad=10, fontsize=14)
plt.show()
```



Observations:

Product vs Gender

- An equal number of males and females have purchased the KP281 product, and this pattern is almost the same for the KP481 product.
- The majority of male customers have purchased the KP781 product.

Product vs MaritalStatus

- Customers who are partnered are more likely to purchase the product.

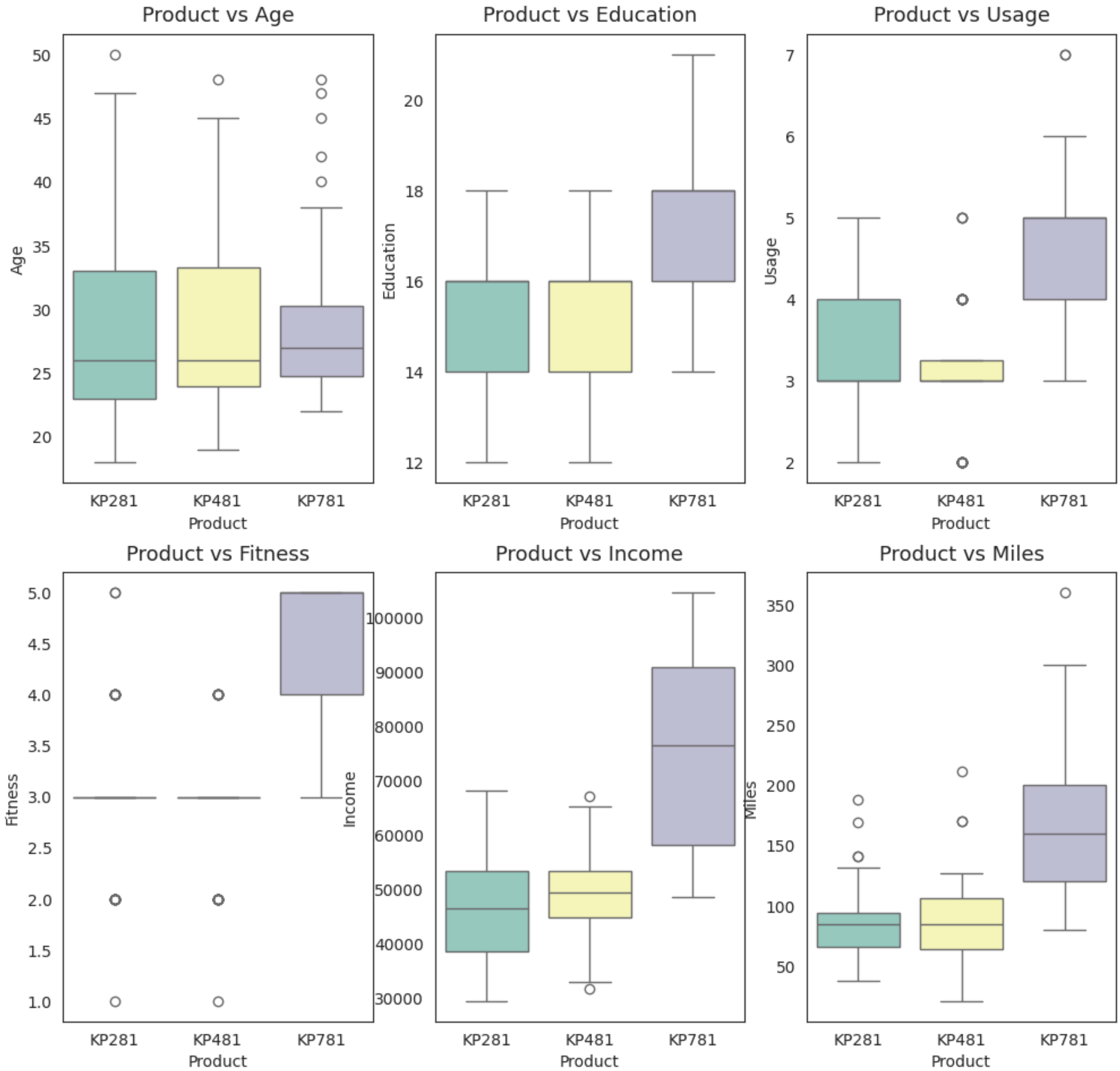
Effect on Product Purchased:

1. Age
2. Education
3. Usage
4. Fitness

5. Income
6. Miles

```
attrs = ['Age', 'Education', 'Usage', 'Fitness', 'Income',
'Miles']
sns.set_style("white")
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(12, 8))
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='Set3')
    axs[i,j].set_title(f"Product vs {attrs[count]}",pad=8, fontsize=13)
    count += 1
```

```
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
<ipython-input-16-124a5fa550be>:9: FutureWarning:

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

  sns.boxplot(data=df, x='Product', y=attrs[count],ax=axs[i,j], palette='Set3')
```



Observations

Product vs Age

- KP281 & KP481 customers have similar median age.
- Age 25-30 favors KP781 purchase.

Product vs Education

- Education > 16 favors KP781.
- Education <= 16 shows equal KP281/KP481 chances.

Product vs Usage

- Usage > 4 times/week favors KP781.
- Otherwise, KP281/KP481 likely.

Product vs Fitness

- Fitness >= 3 favors KP781.

Product vs Income

- Income >= 60000 favors KP781.

Product vs Miles

- Miles > 120/week favors KP781.

Multivariate Analysis:

```
attrs = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
sns.set_style("white")
fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(12, 8))
fig.subplots_adjust(top=1)
count = 0
for i in range(3):
  for j in range(2):
    sns.boxplot(data=df, x='Gender', y=attrs[count], hue='Product', ax=axs[i,j], palette='Set3')
    axs[i,j].set_title(f"Product vs {attrs[count]}", pad=8, fontsize=13)
    count += 1
```