aerofit

April 1, 2024

[ ]:
```
Aerofit Bussiness case
```

[31]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math
import warnings
warnings.filterwarnings("ignore")
```

[53]:
```python
df = pd.read_csv('aerofit_treadmill.txt')
```

[55]:
```python
df.head()
```

[55]:

	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

[34]:
```python
#1
df.dtypes
```

[34]:
```
Product          object
Age               int64
Gender           object
Education         int64
MaritalStatus    object
Usage             int64
Fitness           int64
Income            int64
Miles             int64
dtype: object
```

Insights: There are 9 columns in the dataframe, 3 columns are strings. other columns are integer datatype

[35]:
```
df.shape
```

[35]:
```
(180, 9)
```

Insights: There are 180 rows and 9 columns

[36]:
```
print(df.isnull().any())
```

```
Product          False
Age              False
Gender           False
Education        False
MaritalStatus    False
Usage            False
Fitness          False
Income           False
Miles            False
dtype: bool
```

Insights: No null values are present in entire dataframe

[37]:
```
#4
contingency_table = pd.crosstab(index=df['Product'], columns='count')
marginal_probabilities = contingency_table / len(df)
print("\nMarginal Probabilities:")
print(marginal_probabilities*100)
```

```

Marginal Probabilities:
col_0        count
Product
KP281    44.444444
KP481    33.333333
KP781    22.222222
```

[38]:
```
#4
df1 = df[['Product', 'Gender', 'MaritalStatus']].melt()
df1.groupby(['variable', 'value'])[['value']].count() / len(df)*100
```

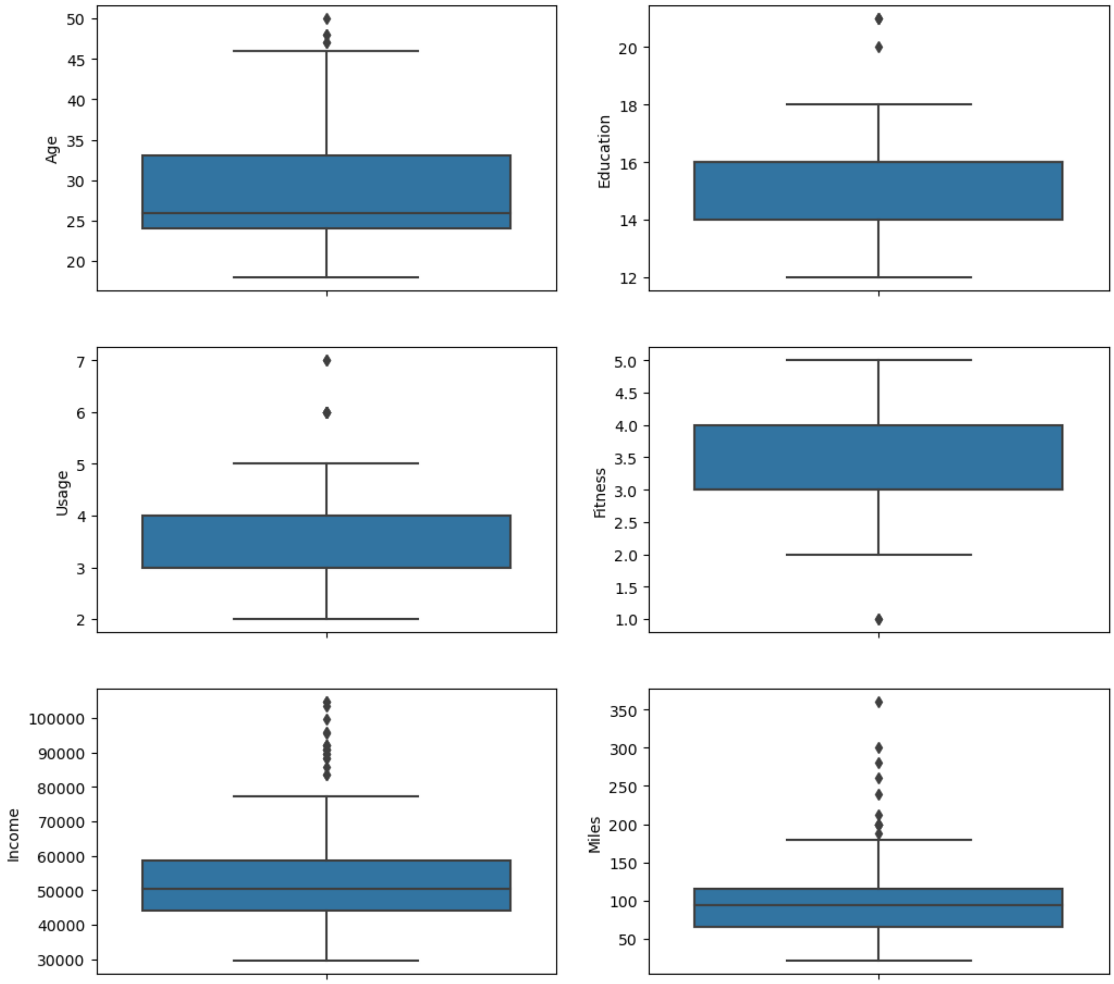
[38]:
```
                              value
variable      value
Gender        Female      42.222222
              Male        57.777778
MaritalStatus Partnered   59.444444
              Single      40.555556
Product       KP281       44.444444
              KP481       33.333333
              KP781       22.222222
```

[39]:
```
#4
conditional_prob = df[df['Product'] == 'KP481']['Gender'].
 ↪value_counts(normalize=True)
print("\nConditional Probability for customer is female, probability of␣
 ↪purchasing KP481:")
print(conditional_prob['Female'])
conditional_prob = df[df['Product'] == 'KP281']['Gender'].
 ↪value_counts(normalize=True)
print("\nConditional Probability for customer is female, probability of␣
 ↪purchasing KP281:")
print(conditional_prob['Female'])
conditional_prob = df[df['Product'] == 'KP781']['Gender'].
 ↪value_counts(normalize=True)
print("\nConditional Probability for customer is female, probability of␣
 ↪purchasing KP781:")
print(conditional_prob['Female'])
```

```
Conditional Probability for customer is female, probability of purchasing KP481:
0.48333333333333334

Conditional Probability for customer is female, probability of purchasing KP281:
0.5

Conditional Probability for customer is female, probability of purchasing KP781:
0.175
```

Insights: Product: 44.44% of the customers have purchased KP2821 product 33.33% of the customers have purchased KP481 product 22.22% of the customers have purchased KP781 product Gender: 57.78% of the customers are Male 42.2% of the customers are Female Marital Status: 59.44% of the customers are Partnered Conditional Probability for customer is female, probability of purchasing KP481: 0.48 Conditional Probability for customer is female, probability of purchasing KP281: 0.5 Conditional Probability for customer is female, probability of purchasing KP781: 0.175

[40]:
```
#2
fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(12, 10))
fig.subplots_adjust(top=1.0)
sns.boxplot(data=df, y="Usage", orient='Vertical', ax=axis[1,0])
sns.boxplot(data=df, y="Fitness", orient='Vertical', ax=axis[1,1])
sns.boxplot(data=df, y="Income", orient='Vertical', ax=axis[2,0])
sns.boxplot(data=df, y="Miles", orient='Vertical', ax=axis[2,1])
sns.boxplot(data=df, y="Age", orient='Vertical', ax=axis[0,0])
sns.boxplot(data=df, y="Education", orient='Vertical', ax=axis[0,1])
plt.show()
```



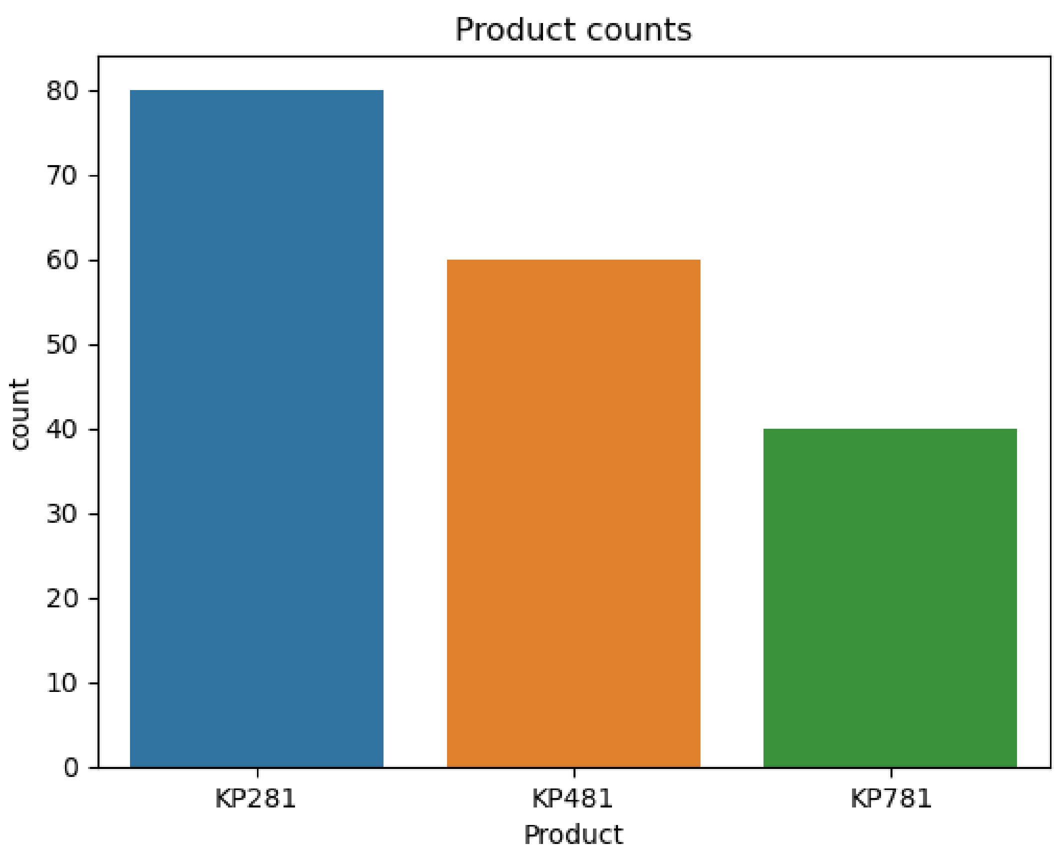
[50]:
```
percentiles = df.quantile([0.05, 0.95])
df_clipped = df.apply(lambda x: np.clip(x, percentiles.loc[0.05, x.name],
 ↪percentiles.loc[0.95, x.name]))
print("Summary Statistics Before Clipping:")
print(df.describe())
print("\nSummary Statistics After Clipping:")
print(df_clipped.describe())
```

```
Summary Statistics Before Clipping:
        Variable1   Variable2   Variable3
count  100.000000  100.000000  100.000000
mean     0.059808    0.082013   -0.059232
std      1.012960    1.039879    0.956799
min     -2.552990   -2.223403   -2.772593
25%     -0.643857   -0.745430   -0.596565
50%      0.094096    0.024655   -0.075359
75%      0.737077    0.847480    0.538657
```

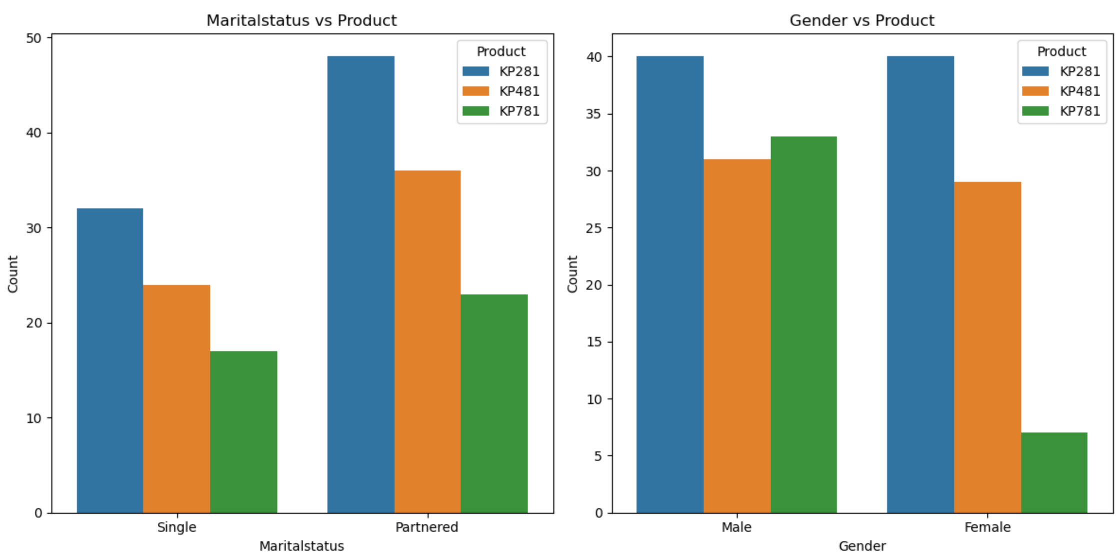
```
max      2.269755    2.383145    2.303917

Summary Statistics After Clipping:
        Variable1   Variable2   Variable3
count  100.000000  100.000000  100.000000
mean     0.062287    0.089032   -0.061743
std      0.956410    0.998585    0.849273
min     -1.614713   -1.349119   -1.619908
25%     -0.643857   -0.745430   -0.596565
50%      0.094096    0.024655   -0.075359
75%      0.737077    0.847480    0.538657
max      1.789955    1.910709    1.496579
```

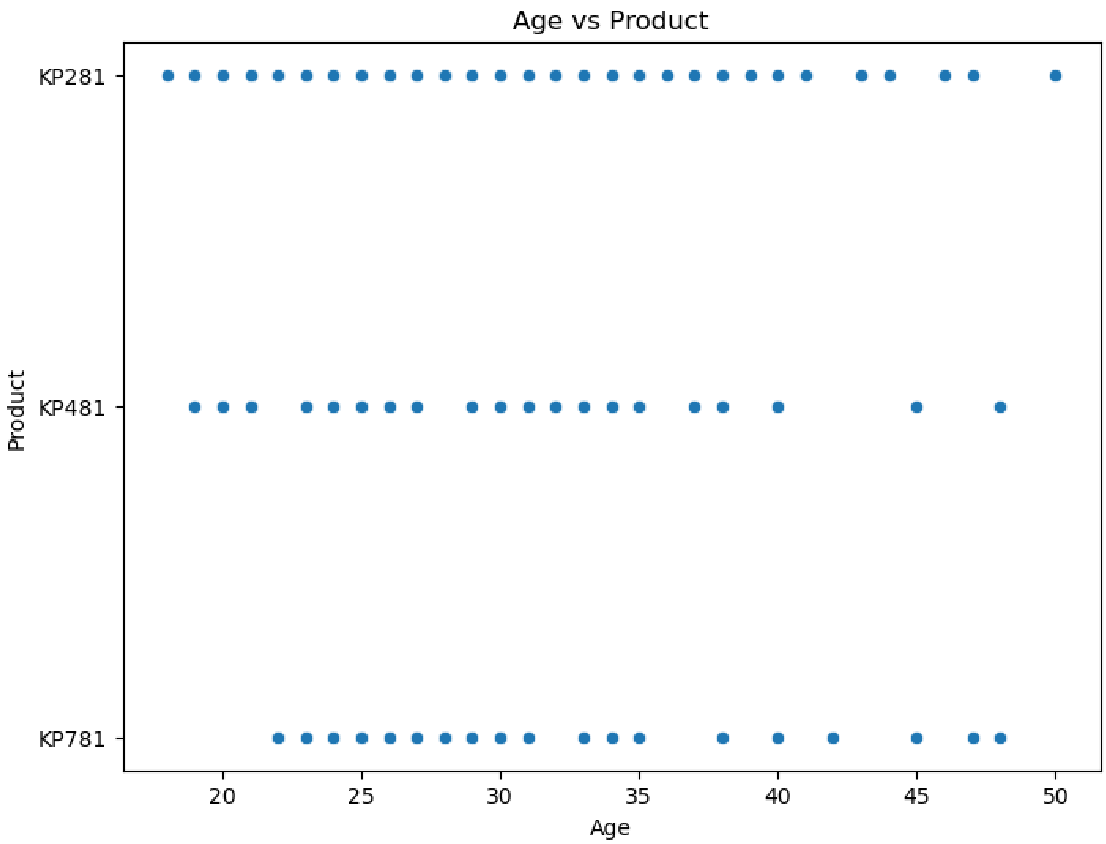
[44]:
```
#3
sns.countplot(data=df, x='Product')
plt.title('Product counts')
```

[44]: Text(0.5, 1.0, 'Product counts')



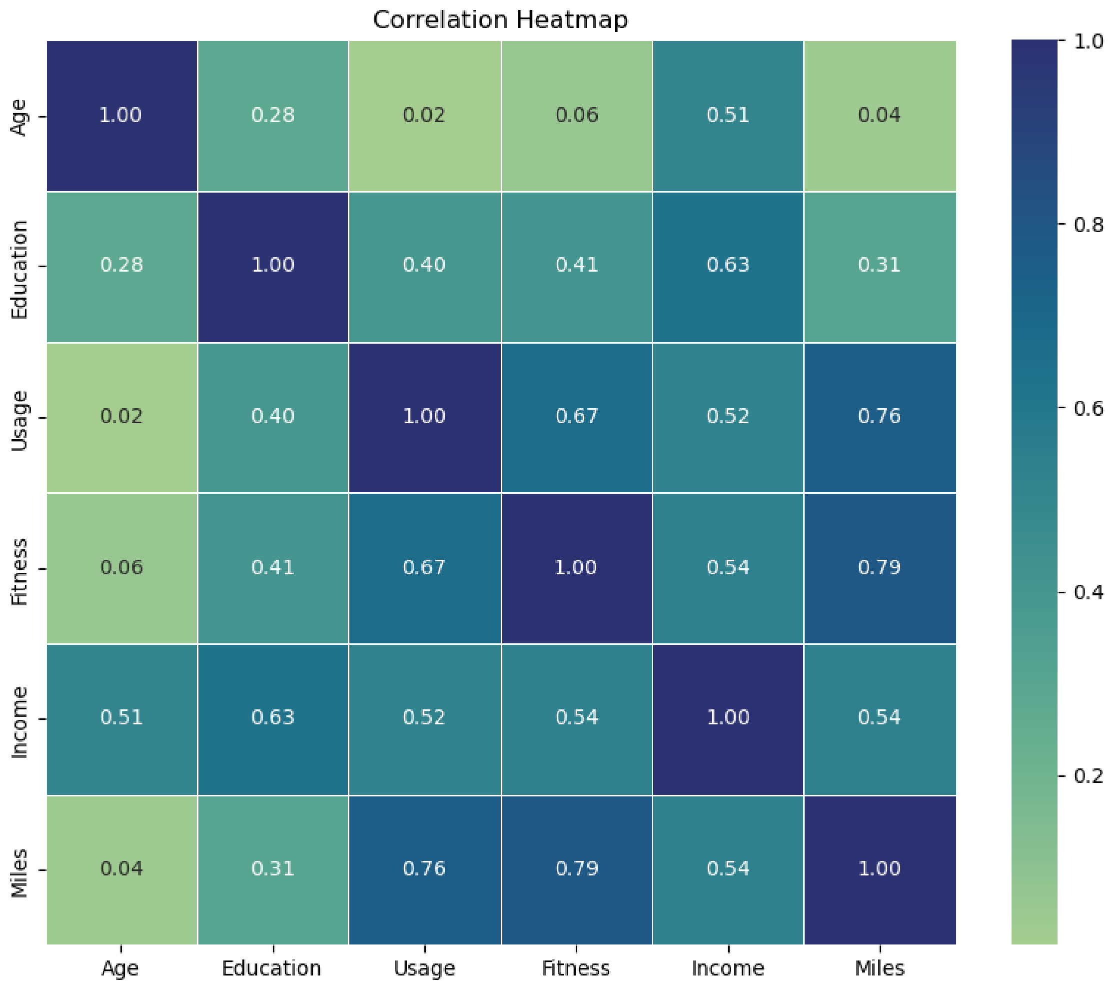
[41]:
```
output_variable = 'Product'
categorical_vars = ['MaritalStatus', 'Gender']
continuous_var = 'Age'
plt.figure(figsize=(12, 6))
for i, cat_var in enumerate(categorical_vars, start=1):
    plt.subplot(1, len(categorical_vars), i)
    sns.countplot(x=cat_var, hue=output_variable, data=df)
    plt.title(f'{cat_var.capitalize()} vs {output_variable.capitalize()}')
    plt.xlabel(cat_var.capitalize())
    plt.ylabel('Count')
plt.tight_layout()
plt.show()
plt.figure(figsize=(8, 6))
sns.scatterplot(x=continuous_var, y=output_variable, data=df)
plt.title(f'{continuous_var.capitalize()} vs {output_variable.capitalize()}')
plt.xlabel(continuous_var.capitalize())
plt.ylabel(output_variable.capitalize())
plt.show()
```





Insights: KP281 is the most frequent product There are more Males in the data than Females More Partnered persons are there in the data Product vs Gender: Equal number of males and females have purchased KP281 product and Almost same for the product KP481 Most of the Male customers have purchased the KP781 product. Product vs MaritalStatus: Customer who is Partnered, is more likely to purchase the product.

[42]:
```
#5
corr_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='crest', fmt=".2f", linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



[63]:

```python
#6
product_KP281 = df[df['Product'] == 'KP281']
average_age_KP281 = product_KP281['Age'].mean()
gender_distribution_KP281 = product_KP281['Gender'].value_counts(normalize=True)
income_group_distribution_KP281 = product_KP281['Income'].
  ↪value_counts(normalize=True)
print(f'Customer Profiling for Product KP281:\n')
print(f'Average Age: {average_age_KP281:.2f} years\n')
print('Gender Distribution:')
print(gender_distribution_KP281)
print('\nIncome Group Distribution:')
print(income_group_distribution_KP281.head())
print('\nRecommendation based on Customer Profiling for KP281:')
print('------------------------------------------------')

# Example recommendations (you can customize these based on your analysis):
```

```
print('- Target marketing campaigns towards the age group with the highest␣
 ↪representation among KP281 customers.')
print('- Tailor product features or promotions to appeal to the predominant␣
 ↪gender group purchasing KP281.')
print('- Adjust pricing or offer discounts targeting income groups that␣
 ↪contribute significantly to KP281 sales.')
print('- Consider introducing product variants or customization options based␣
 ↪on customer preferences identified in the analysis.')
print('- Monitor customer feedback and market trends regularly to adapt␣
 ↪strategies and offerings accordingly.')
```

```
Customer Profiling for Product KP281:

Average Age: 28.55 years

Gender Distribution:
Male      0.5
Female    0.5
Name: Gender, dtype: float64

Income Group Distribution:
46617    0.0875
54576    0.0875
52302    0.0750
35247    0.0625
45480    0.0625
Name: Income, dtype: float64

Recommendation based on Customer Profiling for KP281:
-----------------------------------------------
- Target marketing campaigns towards the age group with the highest
representation among KP281 customers.
- Tailor product features or promotions to appeal to the predominant gender
group purchasing KP281.
- Adjust pricing or offer discounts targeting income groups that contribute
significantly to KP281 sales.
- Consider introducing product variants or customization options based on
customer preferences identified in the analysis.
- Monitor customer feedback and market trends regularly to adapt strategies and
offerings accordingly.
```

[62]:
```
product_KP481 = df[df['Product'] == 'KP481']

# Calculate average age, gender distribution, and income group for customers␣
 ↪purchasing KP281
average_age_KP481 = product_KP481['Age'].mean()
gender_distribution_KP481 = product_KP481['Gender'].value_counts(normalize=True)
```

```
income_group_distribution_KP481 = product_KP481['Income'].
 ↪value_counts(normalize=True)

# Print customer profiling information
print(f'Customer Profiling for Product KP481:\n')
print(f'Average Age: {average_age_KP481:.2f} years\n')
print('Gender Distribution:')
print(gender_distribution_KP481)
print('\nIncome Group Distribution:')
print(income_group_distribution_KP481.head())
print('\nRecommendation based on Customer Profiling for KP481:')
print('-----------------------------------------------')

# Example recommendations (you can customize these based on your analysis):
print('- Target marketing campaigns towards the age group with the highest␣
 ↪representation among KP481 customers.')
print('- Tailor product features or promotions to appeal to the predominant␣
 ↪gender group purchasing KP481.')
print('- Adjust pricing or offer discounts targeting income groups that␣
 ↪contribute significantly to KP481 sales.')
print('- Consider introducing product variants or customization options based␣
 ↪on customer preferences identified in the analysis.')
print('- Monitor customer feedback and market trends regularly to adapt␣
 ↪strategies and offerings accordingly.')
```

```
Customer Profiling for Product KP481:

Average Age: 28.90 years

Gender Distribution:
Male      0.516667
Female    0.483333
Name: Gender, dtype: float64

Income Group Distribution:
45480    0.150000
50028    0.083333
53439    0.083333
43206    0.066667
51165    0.066667
Name: Income, dtype: float64

Recommendation based on Customer Profiling for KP481:
-----------------------------------------------
- Target marketing campaigns towards the age group with the highest
representation among KP481 customers.
- Tailor product features or promotions to appeal to the predominant gender
```

```
group purchasing KP481.
- Adjust pricing or offer discounts targeting income groups that contribute
significantly to KP481 sales.
- Consider introducing product variants or customization options based on
customer preferences identified in the analysis.
- Monitor customer feedback and market trends regularly to adapt strategies and
offerings accordingly.
```

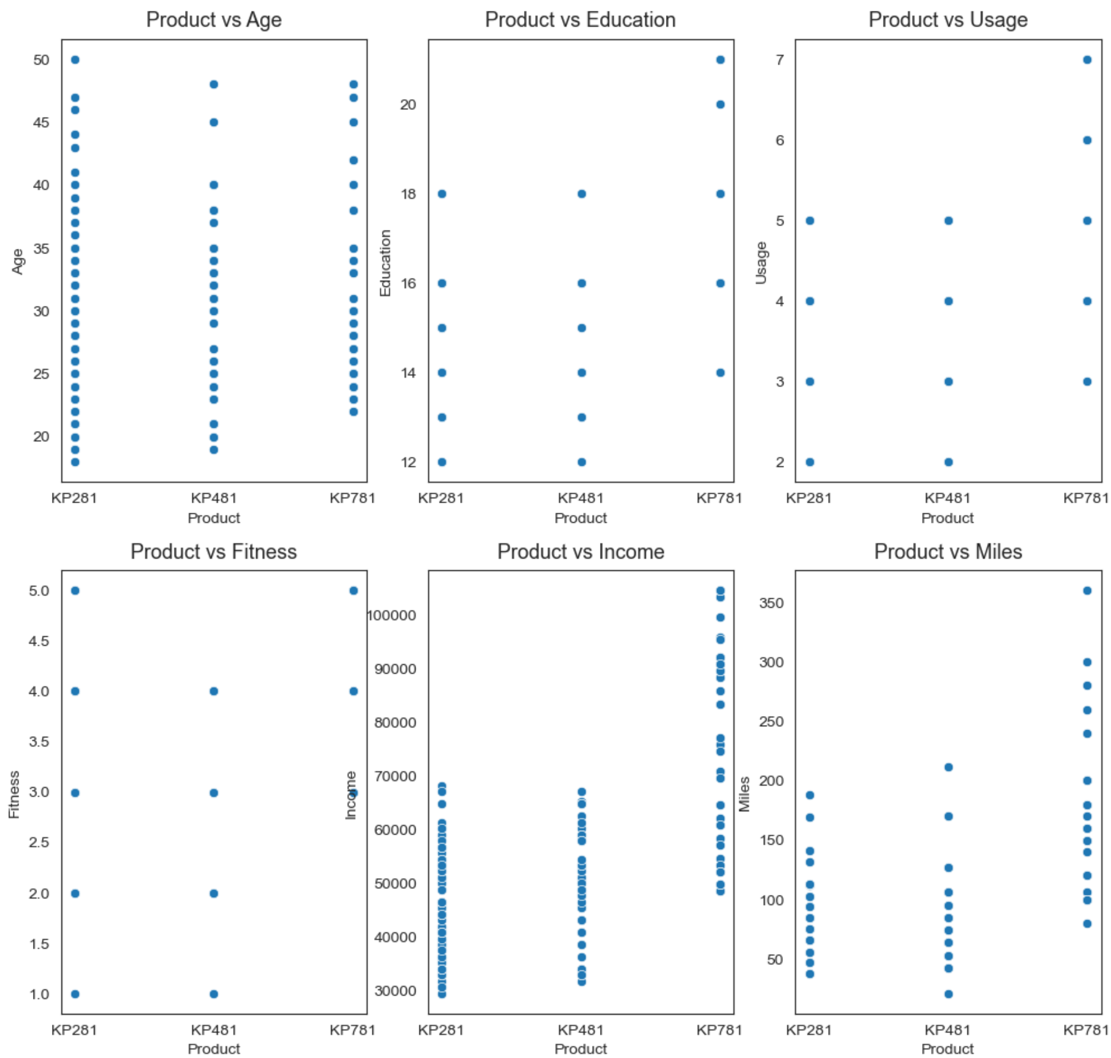
[60]:
```python
product_KP781 = df[df['Product'] == 'KP781']
average_age_KP781 = product_KP781['Age'].mean()
gender_distribution_KP781 = product_KP781['Gender'].value_counts(normalize=True)
income_group_distribution_KP781 = product_KP781['Income'].
 ↪value_counts(normalize=True)

# Print customer profiling information
print(f'Customer Profiling for Product KP781:\n')
print(f'Average Age: {average_age_KP781:.2f} years\n')
print('Gender Distribution:')
print(gender_distribution_KP781)
print('\nIncome Group Distribution:')
print(income_group_distribution_KP781.head())
print('\nRecommendation based on Customer Profiling for KP781:')
print('----------------------------------------------')

# Example recommendations (you can customize these based on your analysis):
print('- Target marketing campaigns towards the age group with the highest␣
 ↪representation among KP781 customers.')
print('- Tailor product features or promotions to appeal to the predominant␣
 ↪gender group purchasing KP781.')
print('- Adjust pricing or offer discounts targeting income groups that␣
 ↪contribute significantly to KP781 sales.')
print('- Consider introducing product variants or customization options based␣
 ↪on customer preferences identified in the analysis.')
print('- Monitor customer feedback and market trends regularly to adapt␣
 ↪strategies and offerings accordingly.')
```

```
Customer Profiling for Product KP781:

Average Age: 29.10 years

Gender Distribution:
Male      0.825
Female    0.175
Name: Gender, dtype: float64

Income Group Distribution:
92131    0.075
```

```
90886    0.075
49801    0.050
89641    0.050
83416    0.050
Name: Income, dtype: float64

Recommendation based on Customer Profiling for KP781:
-----------------------------------------------
- Target marketing campaigns towards the age group with the highest
representation among KP781 customers.
- Tailor product features or promotions to appeal to the predominant gender
group purchasing KP781.
- Adjust pricing or offer discounts targeting income groups that contribute
significantly to KP781 sales.
- Consider introducing product variants or customization options based on
customer preferences identified in the analysis.
- Monitor customer feedback and market trends regularly to adapt strategies and
offerings accordingly.
```

[61]:

```
#3
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.scatterplot(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
```



Insights: Product vs Age: Customers purchasing products KP281 & KP481 are having same Age median value Customers whose age lies between 25-30, are more likely to buy KP781 product Product vs Education Customers whose Education is greater than 16, have more chances to purchase the KP781 product. While the customers with Education less than 16 have equal chances of purchasing KP281 or KP481. Product vs Usage: Customers who are planning to use the treadmill greater than 4 times a week, are more likely to purchase the KP781 product. While the other customers are likely to purchasing KP281 or KP481. Product vs Fitness: The more the customer is fit (fitness >= 3), higher the chances of the customer to purchase the KP781 product. Product vs Income: Higher the Income of the customer (Income >= 60000), higher the chances of the customer to purchase the KP781 product. Product vs Miles: If the customer expects to walk/run greater than 120 Miles per week, it is more likely that the customer will buy KP781 product.

```
[ ]:
```