```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/125/original/aero
```

```
--2022-12-18 17:39:37--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/(
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 13.224.9.:
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|13.224.9
HTTP request sent, awaiting response... 200 OK
Length: 7279 (7.1K) [text/plain]
Saving to: ‘aerofit_treadmill.csv?1639992749.1’

aerofit_treadmill.c 100%[===================>]   7.11K  --.-KB/s    in 0s

2022-12-18 17:39:37 (950 MB/s) - ‘aerofit_treadmill.csv?1639992749.1’ saved [7279/72:
```

```
import warnings
warnings.filterwarnings("ignore")
```

```
import pandas as pd
import numpy as np
df=pd.read_csv('aerofit_treadmill.csv?1639992749')
```

```
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.shape
```

```
(180, 9)
```

```
df.isna().sum() #no null values present
```

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

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

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

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

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

```
print('Maximum and Minimum age of user is {} and {} respectively' .format(max(df['Age']),
```

```
Maximum and Minimum age of user is 50 and 18 respectively
```

- There are a total of 180 rows with 9 columns
- Out of 180 users, 60 use the midlevel runners that sell for 1,750 dollars.
- The data covers Single or Partened Males and Females from age 18 to 50
- Out of 180 users, 80 use the entrylevel treadmill that sell for 1,500 dollars.
- Out of 180 users, 40 use the treadmill having advanced features that sell for 2,500 dollars.

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

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

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

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

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

```
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

From the data above,

1. Continuous variables are - Age, Income and Miles
2. Categorical variables are - Product, Gender, Education, Usage, MaritalStatus and Fitness

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

	Age	Income	Miles
count	180.000000	180.000000	180.000000
mean	28.788889	53719.577778	103.194444
std	6.943498	16506.684226	51.863605
min	18.000000	29562.000000	21.000000
25%	24.000000	44058.750000	66.000000
50%	26.000000	50596.500000	94.000000
75%	33.000000	58668.000000	114.750000
max	50.000000	104581.000000	360.000000

Looking at the descriptive summary of columns above, one can conclude that there are some outliers in the data. We can also make boxplots to be double sure.

```
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['Education'].unique()
```

```
array([14, 15, 12, 13, 16, 18, 20, 21])
```

```
def find_outliers_IQR(df):

   q1=df.quantile(0.25)

   q3=df.quantile(0.75)

   IQR=q3-q1

   outliers = df[((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]

   return outliers


outliers_income = find_outliers_IQR(df["Income"])
print(outliers_income)
print(len(outliers_income))
```

```
159     83416
160     88396
161     90886
162     92131
164     88396
166     85906
167     90886
168    103336
169     99601
170     89641
171     95866
172     92131
173     92131
174    104581
175     83416
176     89641
177     90886
178    104581
179     95508
Name: Income, dtype: int64
19
```

```
outliers_age = find_outliers_IQR(df["Age"])
print(outliers_age)
print(len(outliers_age))
```

```
78     47
79     50
139    48
178    47
179    48
Name: Age, dtype: int64
5
```

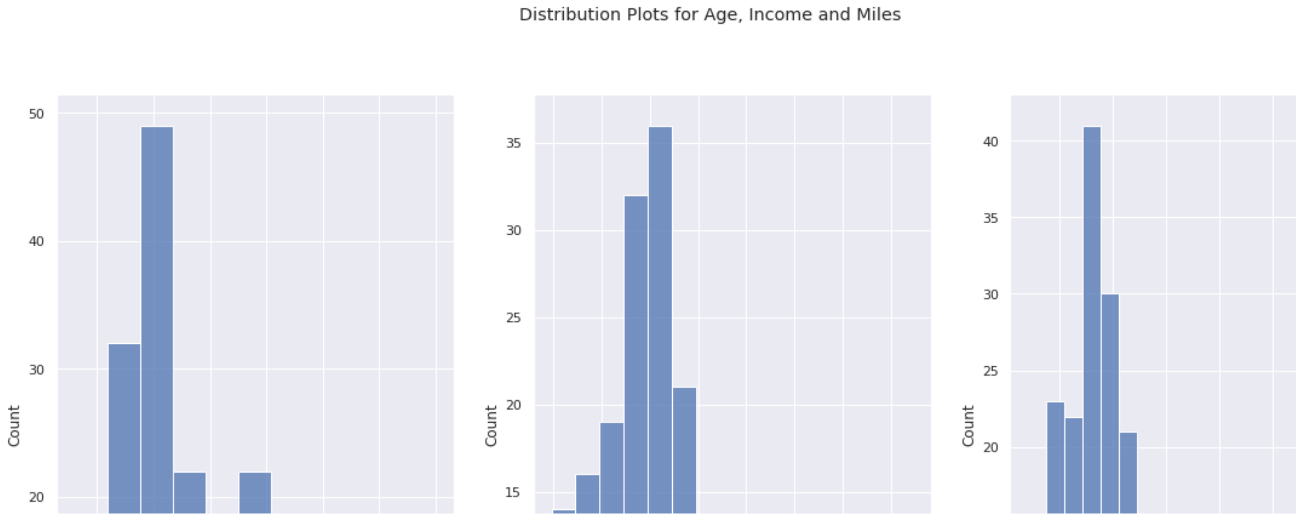
```
outliers_miles = find_outliers_IQR(df["Miles"])
print(outliers_miles)
print(len(outliers_miles))
```

```
23     188
84     212
142    200
148    200
152    200
155    240
166    300
167    280
170    260
171    200
173    360
175    200
176    200
Name: Miles, dtype: int64
13
```

from the above function, there are almost 20 outliers in the income column which accounts to almost 11% of data, and 13 entries from miles column. I am going to go ahead with my analyses without removing the outliers. Also a general observation is people using the KP781 model have a higher salary range.

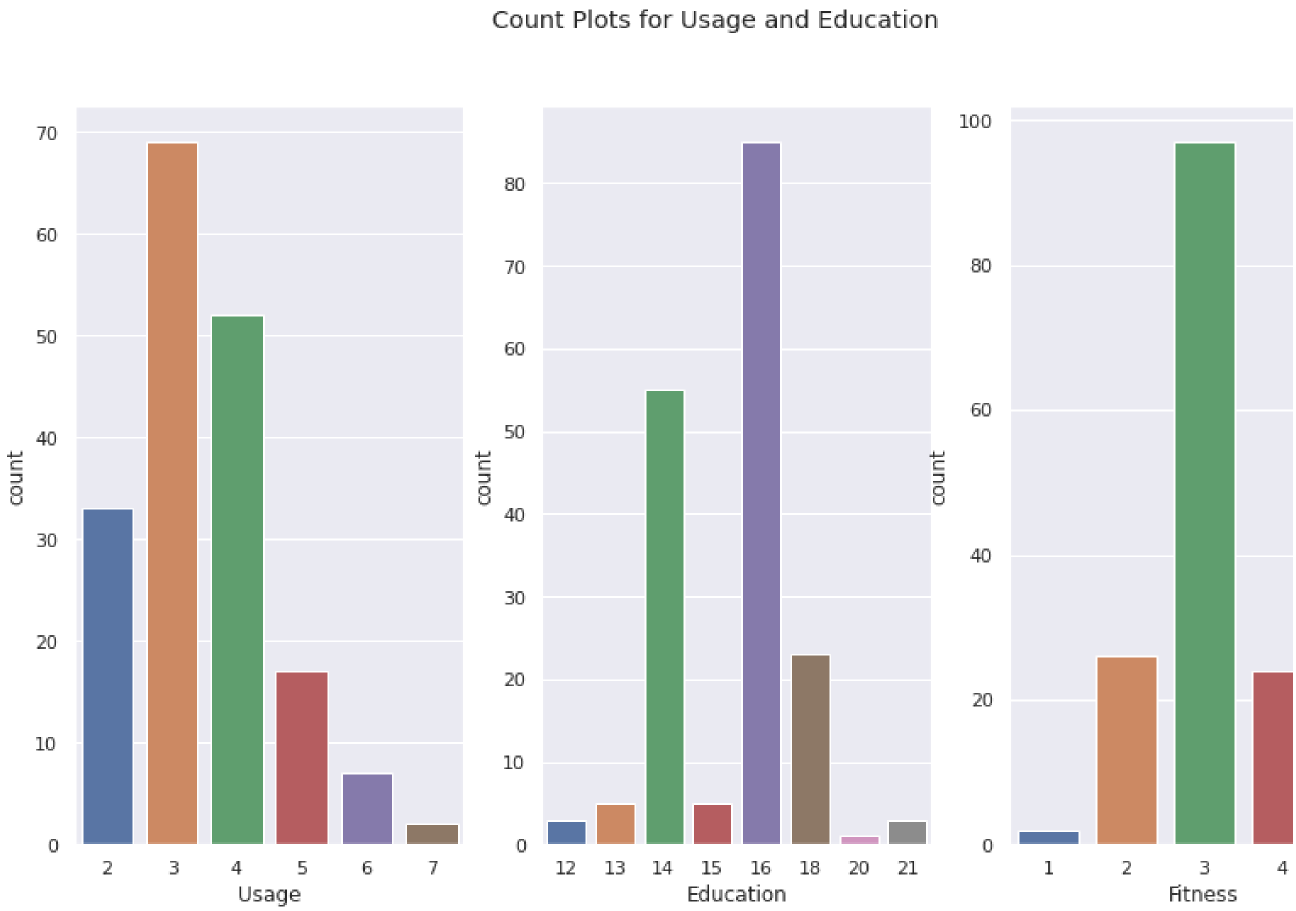
```
#https://dev.to/thalesbruno/subplotting-with-matplotlib-and-seaborn-5ei8
import seaborn as sns
from matplotlib import pyplot as plt
sns.set()
fig, axes = plt.subplots(1, 3, figsize=(20, 10))
fig.suptitle('Distribution Plots for Age, Income and Miles')
sns.histplot(ax=axes[0], data=df, x='Age')
sns.histplot(ax=axes[1], data=df, x='Income')
sns.histplot(ax=axes[2], data=df, x='Miles')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5c46c35b80>
```



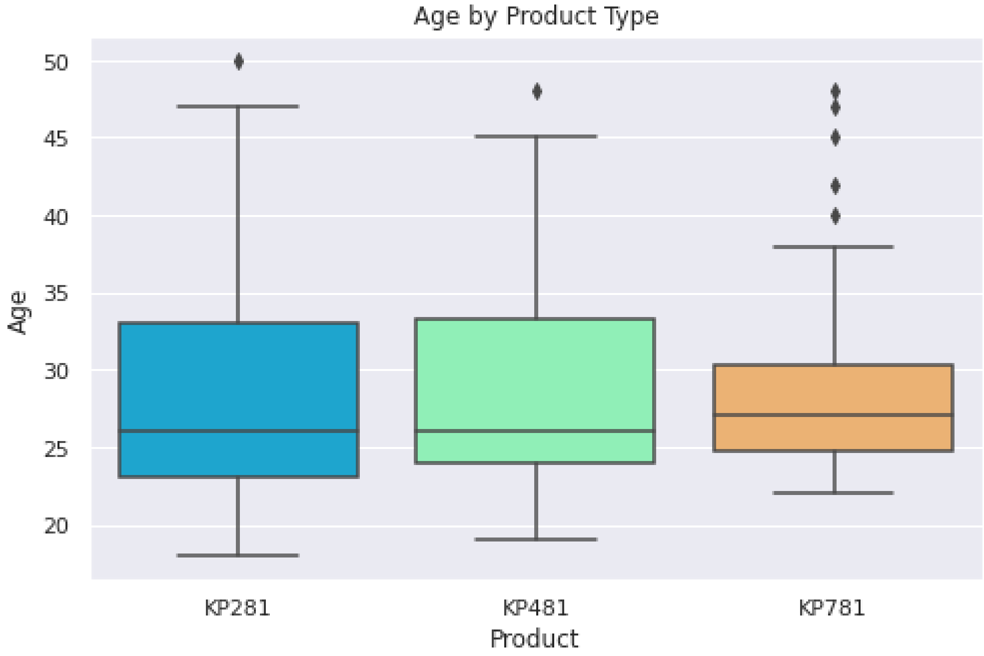

```
#for usage and education
fig, axes = plt.subplots(1, 3, figsize=(14, 8))
fig.suptitle('Count Plots for Usage and Education')
sns.countplot(ax=axes[0], data=df, x='Usage')
sns.countplot(ax=axes[1], data=df, x='Education')
sns.countplot(ax=axes[2], data=df, x='Fitness')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5c46a8fc10>
```



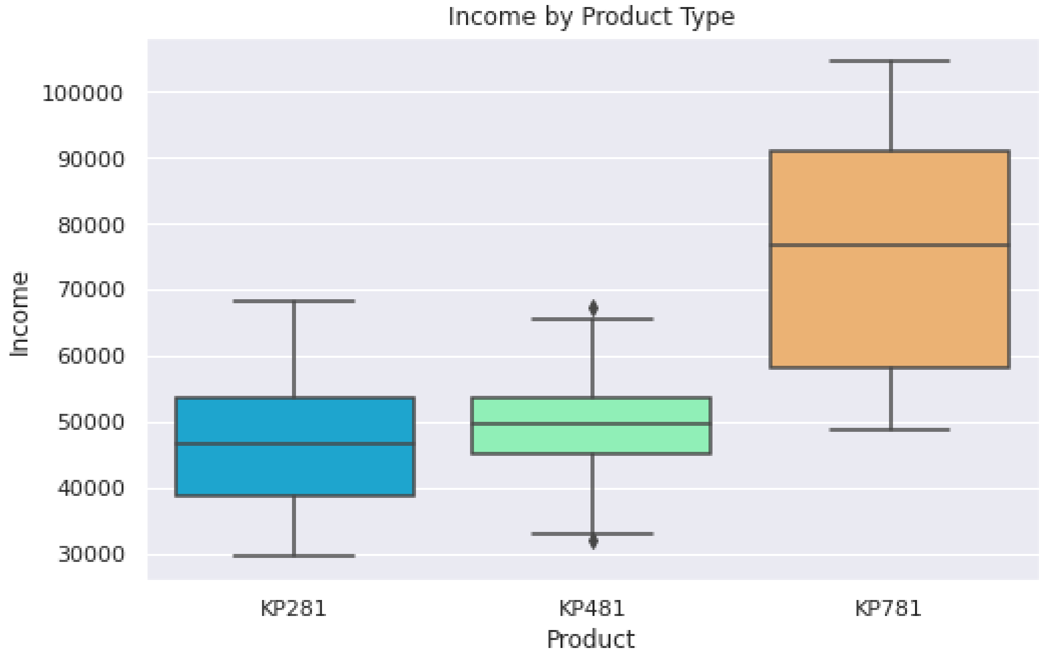
```
plt.figure(figsize=(8,5))
sns.boxplot(x='Product',y='Age',data=df, palette='rainbow')
plt.title("Age by Product Type")
```

```
Text(0.5, 1.0, 'Age by Product Type')
```



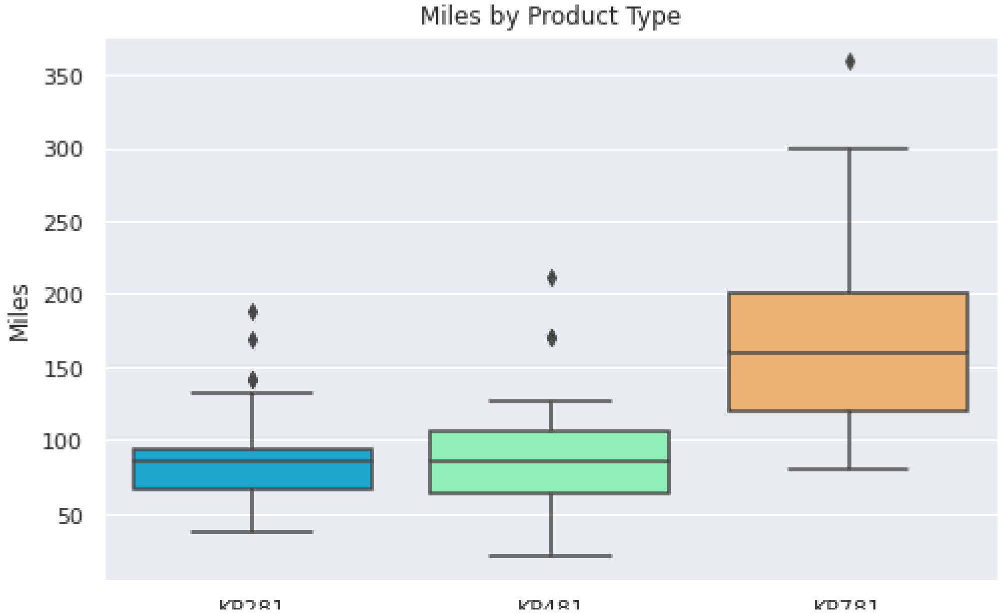

```
plt.figure(figsize=(8,5))
sns.boxplot(x='Product',y='Income',data=df, palette='rainbow')
plt.title("Income by Product Type")
```

```
Text(0.5, 1.0, 'Income by Product Type')
```



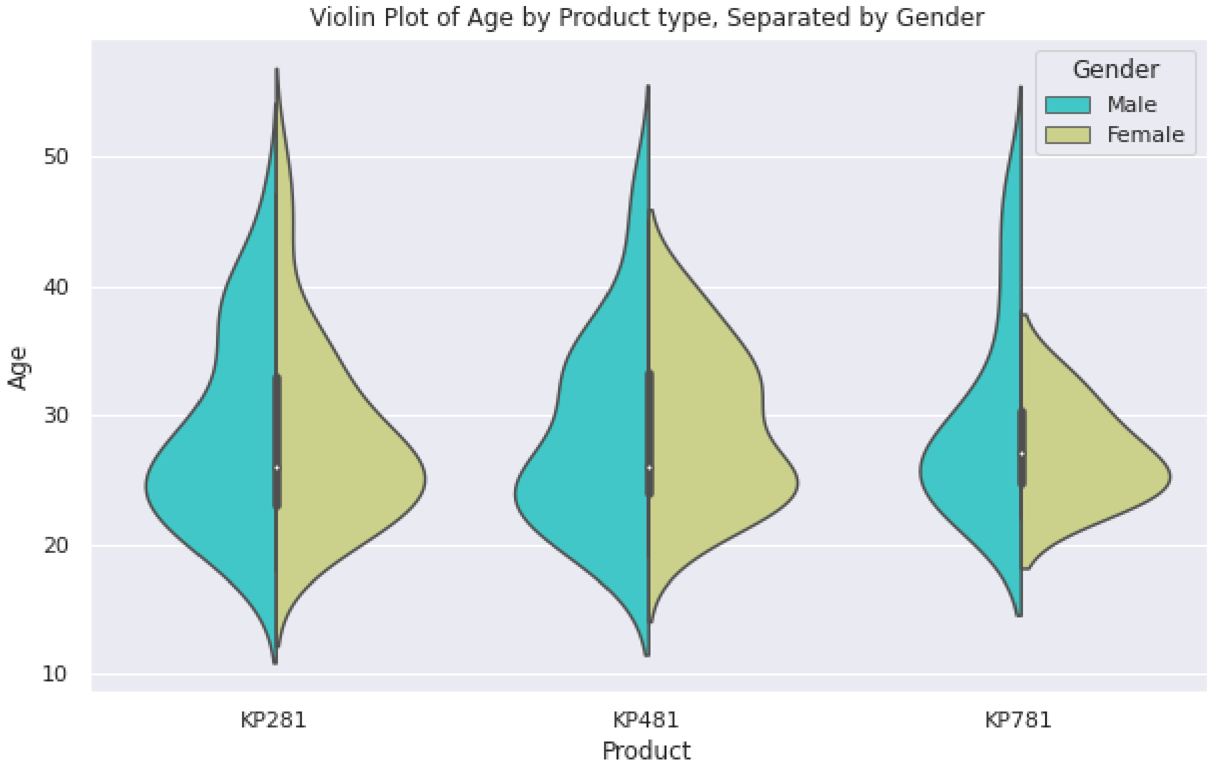

```
plt.figure(figsize=(8,5))
sns.boxplot(x='Product',y='Miles',data=df, palette='rainbow')
plt.title("Miles by Product Type")
```

```
Text(0.5, 1.0, 'Miles by Product Type')
```




```
plt.figure(figsize=(10,6))
sns.violinplot(x='Product',y='Age',data=df, hue='Gender', split='True', palette='rainbow')
plt.title("Violin Plot of Age by Product type, Separated by Gender")
```

```
Text(0.5, 1.0, 'Violin Plot of Age by Product type, Separated by Gender')
```

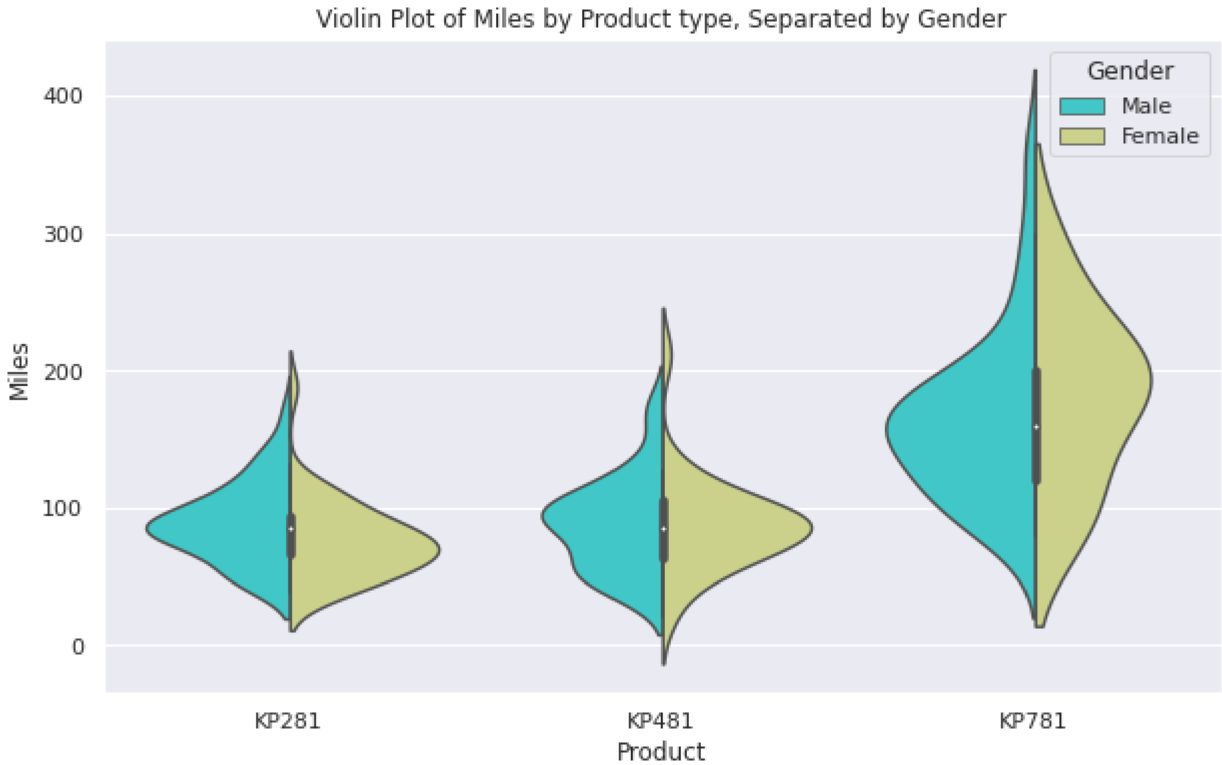



```
plt.figure(figsize=(10,6))
sns.violinplot(x='Product',y='Income',data=df, hue='Gender', split='True', palette='rainbo
plt.title("Violin Plot of Income by Product type, Separated by Gender")
```

```
Text(0.5, 1.0, 'Violin Plot of Income by Product type, Separated by Gender')
```

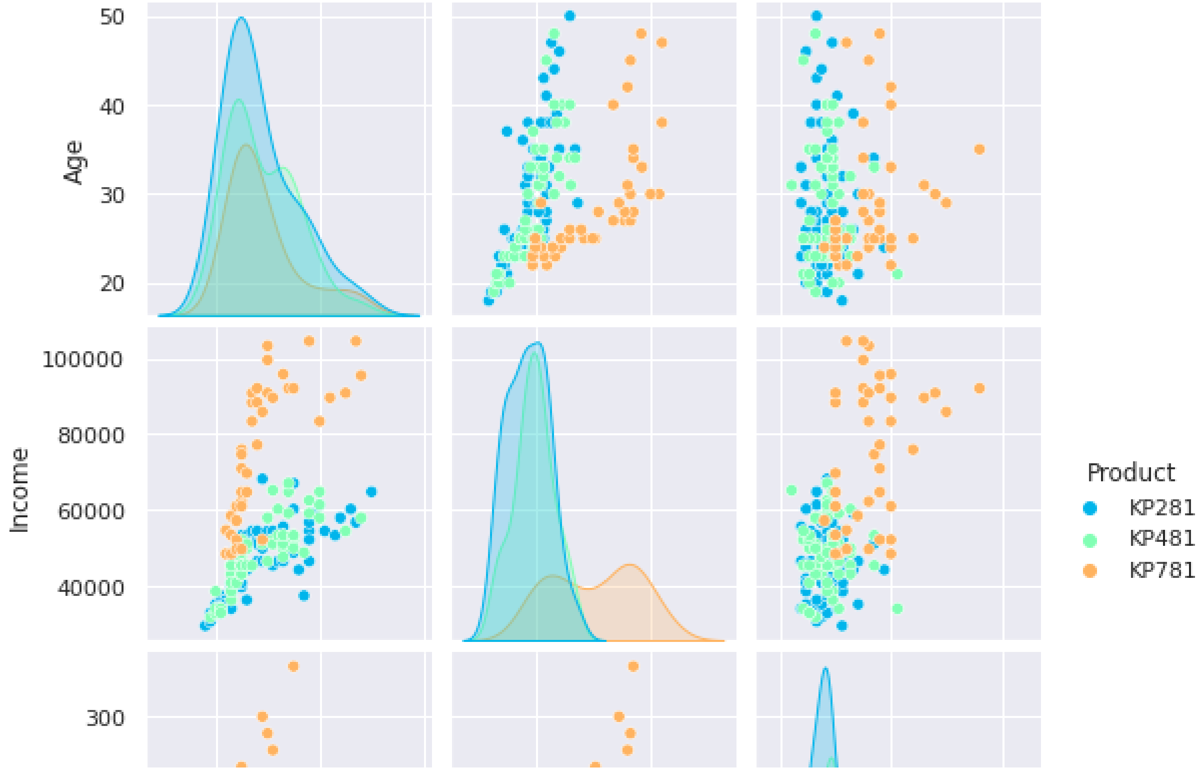

```
plt.figure(figsize=(10,6))
sns.violinplot(x='Product',y='Miles',data=df, hue='Gender', split='True', palette='rainbow
plt.title("Violin Plot of Miles by Product type, Separated by Gender")
```

```
Text(0.5, 1.0, 'Violin Plot of Miles by Product type, Separated by Gender')
```




```
sns.pairplot(data=df[['Age', 'Income', 'Miles','Product']], hue='Product', palette='rainbo
```

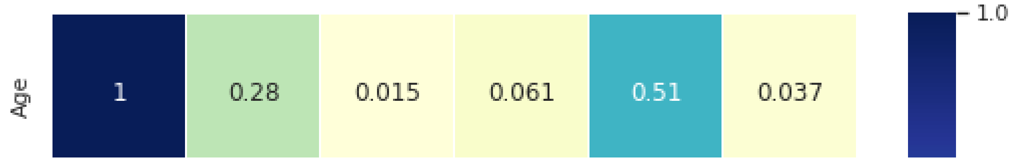
<seaborn.axisgrid.PairGrid at 0x7f5c4672bc70>



User of model KP281 and KP481 tend to have a similar range of income and tend to run in a similar pattern, whereas users of model KP781 have a higher range of income and tend to run more than the users of above two product category.

```
fig, ax = plt.subplots(figsize=(9,9))
sns.heatmap(data=df.corr(),cmap="YlGnBu", annot=True, ax=ax, linewidths=.5)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5c463e6c10>
```




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

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

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

MaritalStatus	Partnered	Single	All
Product			
KP281	48	32	80
KP481	36	24	60
KP781	23	17	40
All	107	73	180

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

MaritalStatus	Partnered		Single		All
Gender	Female	Male	Female	Male	
Product					
KP281	27	21	13	19	80
KP481	15	21	14	10	60
KP781	4	19	3	14	40
All	46	61	30	43	180

Recommendations:

1. Model KP281 is the most prefered model among the users and should be stocked more followed by KP481 and KP781.

2. Customers with a median income between 70k to 80k are the potential customers for model KP781. Customers with a median salary between 45k-50k tend to buy KP281 and KP481.
3. Customers who have a target of running between 50-120 miles/week look for model KP281 and KP481 whereas customers who target to run 125-200 miles/week look for the advanced product.
4. Age bracket for the advanced model KP781 is 23-38 years. As a customer gets older, they prefer the lighter models such as KP281 and KP481 with age range of 18-48
5. Models KP281 and KP481 are equally prefered by both males and females. Out of 76 female customers, only 7 preferred to buy model KP781 which helps us understand that almost 90% female prefer KP281 or KP481.
6. Out of 104 males, data is almost equally distributed for the product/model type they preferred.
7. Out of 180 customers, almost 60% have a partner.
8. 60% of customers who either bought KP281 or KP481 are having a partner. The number drops to 57.5% for model KP781.
9. Females either married or partnered dont prefer to buy the heavy model KP781.

✓ 0s completed at 11:09 PM