In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files
```

Uploading the Dataset

In [2]:
```
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving aerofit_treadmill.csv to aerofit_treadmill.csv
```

Reading the Dataset

In [3]:
```
df_afit = pd.read_csv("aerofit_treadmill.csv")
```

Analysing Basic Metrics - Exploratory Data Analysis(EDA)

In [4]:
```
df_afit.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
```

There are 9 Columns and 180 rows.

In [5]:
```
# Printing first 5 rows of dataframe
df_afit.head()
```

Out[5]:

	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

In [6]:

```
# Checking type of Dataset
type(df_afit)
```

Out[6]:

```
pandas.core.frame.DataFrame
```

In [7]:

```
# Datatypes of columns
df_afit.dtypes
```

Out[7]:

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

There are 3 columns of type - Object 6 columns of tyoe - int

In [8]:

```
df_afit.shape
```

Out[8]:

```
(180, 9)
```

In [9]:

```
df_afit.columns
```

Out[9]:

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

In [10]:

```
df_afit.describe()
```

Out[10]:

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	5.000000	104581.000000	360.000000

In [32]:
```
df_afit.describe(include="all")
```

Out[32]:

	Product	Age	Category_Age	Gender	Education	MaritalStatus	Usage	Fitnes
count	180	180.000000	180	180	180.000000	180	180.000000	180.00000
unique	3	NaN	3	2	NaN	2	NaN	NaI
top	KP281	NaN	Young	Male	NaN	Partnered	NaN	NaI
freq	80	NaN	113	104	NaN	107	NaN	NaI
mean	NaN	28.788889	NaN	NaN	15.572222	NaN	3.455556	3.31111
std	NaN	6.943498	NaN	NaN	1.617055	NaN	1.084797	0.95886
min	NaN	18.000000	NaN	NaN	12.000000	NaN	2.000000	1.00000
25%	NaN	24.000000	NaN	NaN	14.000000	NaN	3.000000	3.00000
50%	NaN	26.000000	NaN	NaN	16.000000	NaN	3.000000	3.00000
75%	NaN	33.000000	NaN	NaN	16.000000	NaN	4.000000	4.00000
max	NaN	50.000000	NaN	NaN	21.000000	NaN	7.000000	5.00000

Checking for NA values

In [11]:
```
df_afit.isna().sum()
```

Out[11]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

There are no NA/missing values in given dataset

Conversion of categorical attributes to 'category'

Categorizing income column

In [22]:
```
print("Min_income =", min(df_afit.Income)," ","Max_income =", max(df_afit.Income))
```

```
Min_income = 29562   Max_income = 104581
```

In [23]:
```
def income_bins(income):
  if (income <= 40000): return 'low'
  if (income > 40000 and income <= 80000): return 'medium'
  if (income > 80000): return 'high'

df_afit['Category_income'] = df_afit['Income'].apply(income_bins)
df_afit.head()
```

Out[23]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Category_income
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

Categorizing Miles column

In [24]:
```
print("Min_miles_covered =",min(df_afit.Miles)," ","Max_miles_covered =",max(df_afit.M
```

```
Min_miles_covered = 21   Max_miles_covered = 360
```

In [25]:
```
def miles_bins(miles):
  if (miles <= 100): return 'Basic_workout'
  if (miles > 100 and miles <= 200): return 'Medium_workout'
  if (miles > 200): return 'Intense_workout'

df_afit['Category_miles'] = df_afit['Miles'].apply(miles_bins)
df_afit.head()
```

Out[25]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Category_income
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

Categorizing Age column

In [27]:
```
print("Min_age =",min(df_afit.Age)," ","Max_age =",max(df_afit.Age))
```

```
Min_age = 18   Max_age = 50
```

In [28]:
```
def age_bins(age):
  if (age <= 29): return 'Young'
  if (age > 29 and age <= 40): return 'Adult'
  if (age > 40): return 'Aged'

df_afit['Category_Age'] = df_afit['Age'].apply(age_bins)
df_afit
```

Out[28]:

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

180 rows × 12 columns

Let us re-arrange the columns

In [29]:
```
df_afit = df_afit[["Product","Age","Category_Age","Gender","Education","MaritalStatus'
df_afit.head()
```

Out[29]:

	Product	Age	Category_Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Category
0	KP281	18	Young	Male	14	Single	3	4	29562	
1	KP281	19	Young	Male	15	Single	2	3	31836	
2	KP281	19	Young	Female	14	Partnered	4	3	30699	
3	KP281	19	Young	Male	12	Single	3	3	32973	
4	KP281	20	Young	Male	13	Partnered	4	2	35247	

In [31]:

```
df_afit.shape
```

Out[31]:

```
(180, 12)
```

Observations

==> Given Dataset is about Areofit company, which is into selling the fintess products.

==> 3 Unique product details given - (KP281,KP481,Kp781)

==> There are 9 columns and 180 rows - before categorization. After categorization - 12 columns and 180 rows

==> No Null values found in the given dataset

==> KP281 product is most sold

==> Minimum age of person = 18, Maximum = 50. Mean = 28.79

==> People have mininum of 12 years Education and maximum of 21yrs Education

==> There are 104 Male customers and 76 Female customers

==> Standard deviation for Income & Miles is very high. These variables might have the outliers in it.

Non-Graphical Analysis - Value counts and unique attributes

In [33]:

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

Out[33]:

```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [48]:

```
df_afit["Fitness"].value_counts(sort = False)
```

Out[48]:
```
4    24
3    97
2    26
1     2
5    31
Name: Fitness, dtype: int64
```

In [36]:
```
df_afit[["Fitness" , "Gender"]].value_counts(sort = False)
```

Out[36]:
```
Fitness  Gender
1        Female     1
         Male       1
2        Female    16
         Male      10
3        Female    45
         Male      52
4        Female     8
         Male      16
5        Female     6
         Male      25
dtype: int64
```

In [37]:
```
df_afit["Product"].value_counts()
```

Out[37]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [38]:
```
df_afit[["Product" , "Gender"]].value_counts(sort = False)
```

Out[38]:
```
Product  Gender
KP281    Female    40
         Male      40
KP481    Female    29
         Male      31
KP781    Female     7
         Male      33
dtype: int64
```

In [39]:
```
df_afit[["Product" , "Gender","Fitness"]].value_counts(sort = False)
```

Out[39]:
```
Product  Gender  Fitness
KP281    Female  2          10
                 3          26
                 4           3
                 5           1
         Male    1           1
                 2           4
                 3          28
                 4           6
                 5           1
KP481    Female  1           1
                 2           6
                 3          18
                 4           4
         Male    2           6
                 3          21
                 4           4
KP781    Female  3           1
                 4           1
                 5           5
         Male    3           3
                 4           6
                 5          24
dtype: int64
```

In [40]:
```
df_afit[["Product","Fitness"]].value_counts(sort = False)
```

Out[40]:
```
Product  Fitness
KP281    1           1
         2          14
         3          54
         4           9
         5           2
KP481    1           1
         2          12
         3          39
         4           8
KP781    3           4
         4           7
         5          29
dtype: int64
```

In [41]:
```
df_afit["Category_Age"].value_counts()
```

Out[41]:
```
Young    113
Adult     55
Aged      12
Name: Category_Age, dtype: int64
```

In [42]:
```
df_afit["Category_income"].value_counts()
```

Out[42]:
```
medium    129
low        32
high       19
Name: Category_income, dtype: int64
```

In [43]:
```
df_afit["Category_miles"].value_counts()
```

Out[43]:
```
Basic_workout      114
Medium_workout      60
Intense_workout      6
Name: Category_miles, dtype: int64
```

In [44]:
```
df_afit["Product"].unique()
```

Out[44]:
```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

In [45]:
```
df_afit["Product"].nunique()
```

Out[45]:
```
3
```

In [46]:
```
df_afit["Age"].nunique()
```

Out[46]:
```
32
```

In [47]:
```
df_afit["Education"].nunique()
```

Out[47]:
```
8
```

Observations

==> There are 104 Male customers and 76 Female customers

==> There are 5 categories of Fitness level - Most of the customers fall into Fitness3 category

==> In Fitness 3 category - there are 53 male and 45 female customers

==> KP281 is the most sold product, followed by KP481 and KP781

==> Compared to female, male customers have more fitness levels

==> Customers have achieved Fitness level 3 by using the product KP281 widely.

==> Customer age groups fall into below

```
113 - young age customers ( <=29 )
55 - Adult age customers ( > 29 and  <= 40 )
12 - old age customers ( >40 )
```

==> out of 180 customers, 129 have medium income, 32 have low income, 19 have very high income

==> out of 180 customers, 114 do basic workout using treadmill, 60 do medium workout and 6 do intense workout

==> There are totally 32 age groups of customers and they have minimum of 12 years of Education

==> There are 8 Education groups in total.

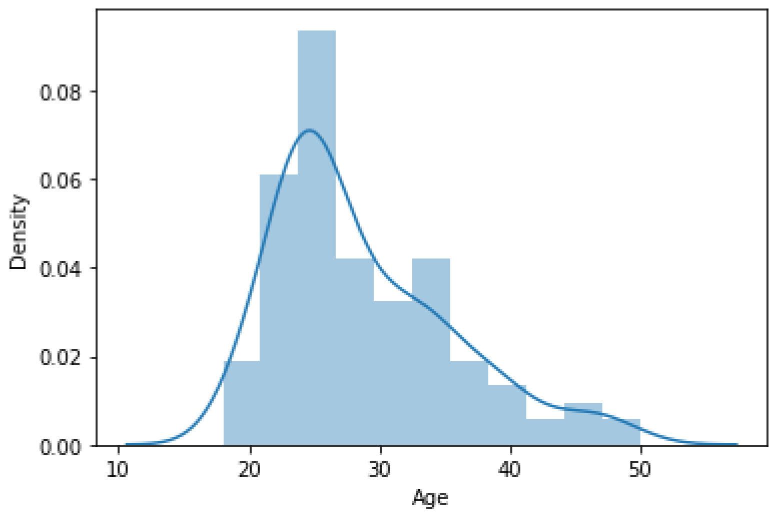
Visual Analysis

Univariate Analysis

Distplot

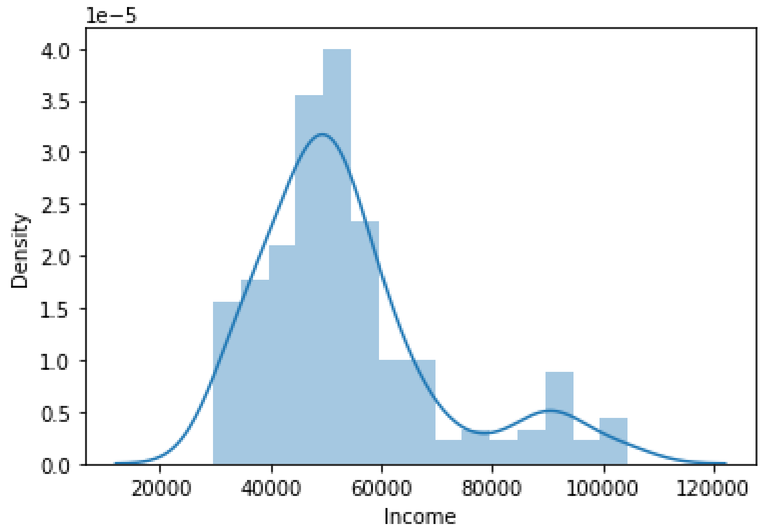
In [51]:
```
sns.distplot(df_afit.Age)
plt.show()
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please a
dapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



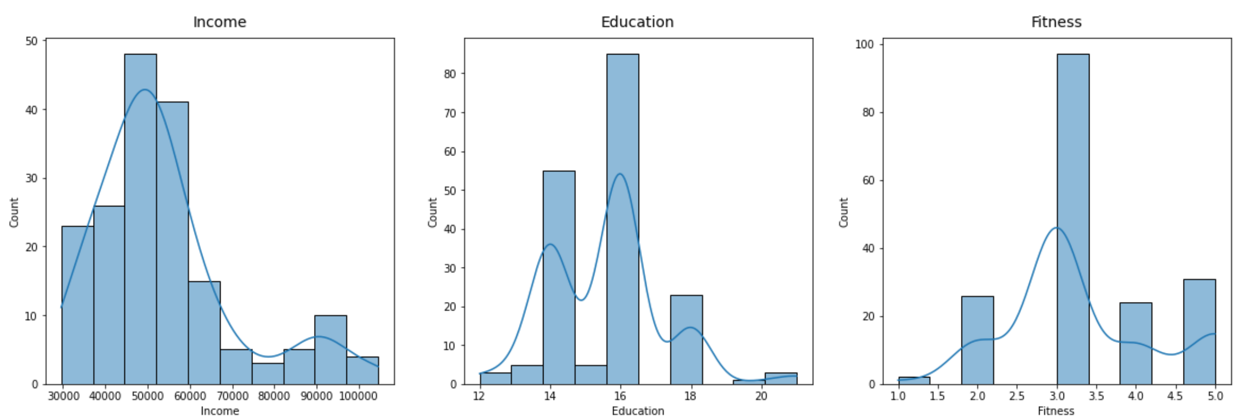
In [52]:
```
sns.distplot(df_afit.Income)
plt.show()
```

```
/usr/local/lib/python3.9/dist-packages/seaborn/distributions.py:2619: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please a
dapt your code to use either `displot` (a figure-level function with similar flexibil
ity) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



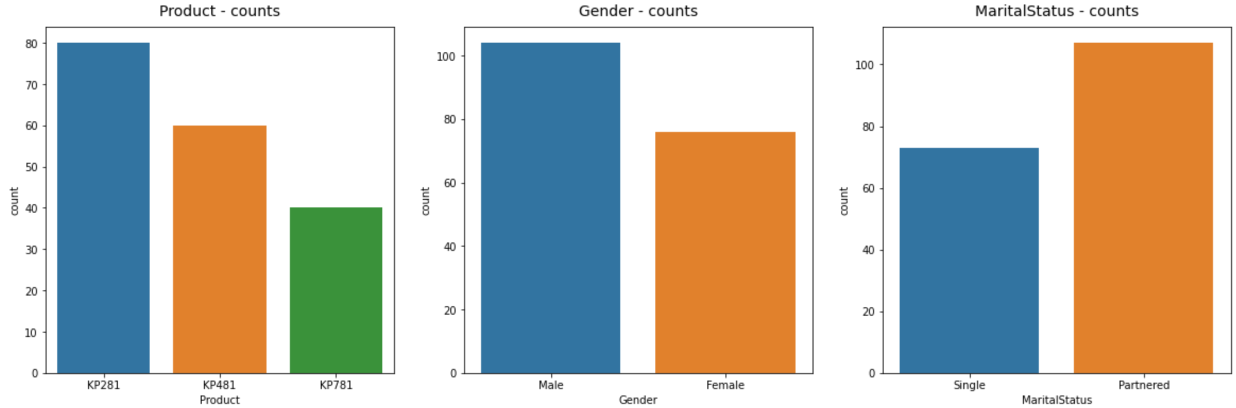
Histplot

In [55]:
```
fig, axs = plt.subplots(nrows=1, ncols = 3, figsize=(20,6))
sns.histplot(df_afit['Income'], kde = True, bins = 10 ,ax = axs[0])
sns.histplot(df_afit['Education'], kde = True, bins = 10 ,ax = axs[1])
sns.histplot(df_afit['Fitness'], kde = True, bins = 10 ,ax = axs[2])

axs[0].set_title("Income", pad=10, fontsize=14)
axs[1].set_title("Education", pad=10, fontsize=14)
axs[2].set_title("Fitness", pad=10, fontsize=14)

plt.show()
```



Countplot

In [56]:
```
fig, axs = plt.subplots(nrows=1, ncols = 3, figsize=(20,6))

sns.countplot(data=df_afit, x='Product', ax=axs[0])
sns.countplot(data=df_afit, x='Gender', ax=axs[1])
sns.countplot(data=df_afit, 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()
```



Univariate Analysis Observations

Distplot

==> More customers fall into 20-30 age group

==> More customers have income between - 40,000 - 60,000

Histplot

==> More customers have 16yrs Education

==> More customers have fitness level 3

Countplot

==> KP281 is most sold product

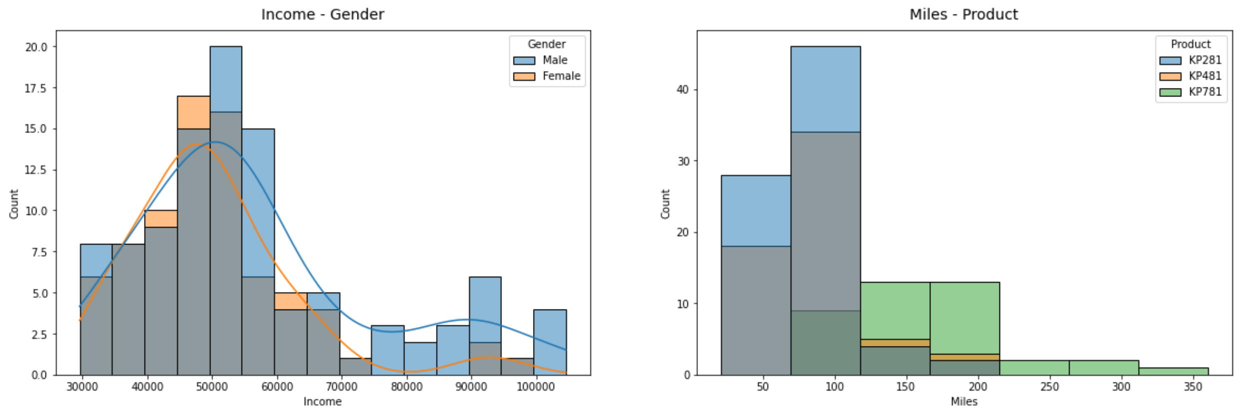
==> Male customers are more than female.

==> More partnered customers are there than single customers in data

Bivariate Analysis

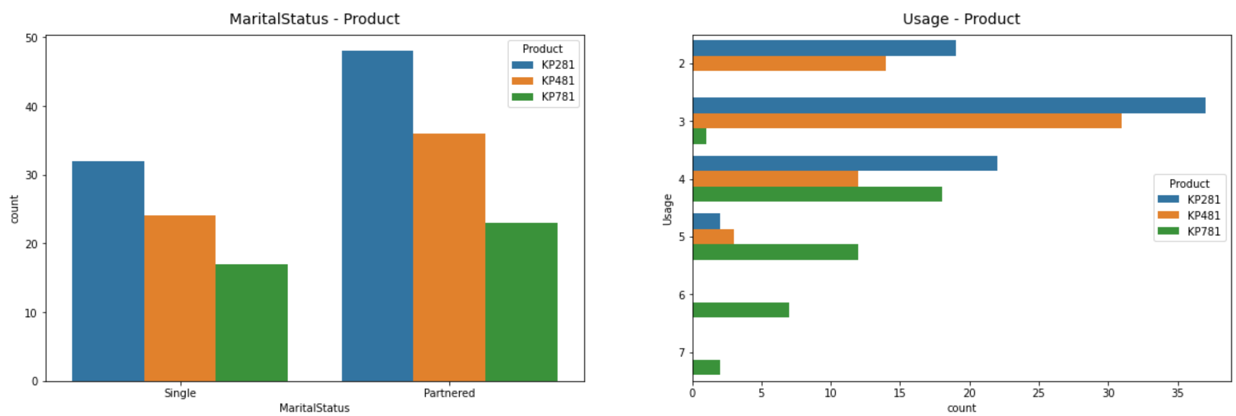
Histplot

In [65]:
```
fig, axs = plt.subplots(nrows=1, ncols = 2, figsize=(20,6))

sns.histplot(x="Income", data = df_afit, kde = True, hue="Gender",ax=axs[0])
sns.histplot(data = df_afit, x='Miles', hue='Product',bins= 7,ax=axs[1])

axs[0].set_title("Income - Gender", pad=10, fontsize=14)
axs[1].set_title("Miles - Product", pad=10, fontsize=14)

plt.show()
```



Countplot

In [67]:
```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

sns.countplot(data=df_afit, x = df_afit['MaritalStatus'], hue='Product', ax =axs[0])
sns.countplot(data=df_afit, y='Usage',hue='Product',ax=axs[1])

axs[0].set_title("MaritalStatus - Product", pad=10, fontsize=14)
axs[1].set_title("Usage - Product", pad=10, fontsize=14)

plt.show()
```



Bivariate Analysis - Observations

Histplot

==> Male customers have more income compared to female

==> More customers have income between - 40,000 - 60,000

==> KP281 is widely used treadmill than other 2.

Countplot

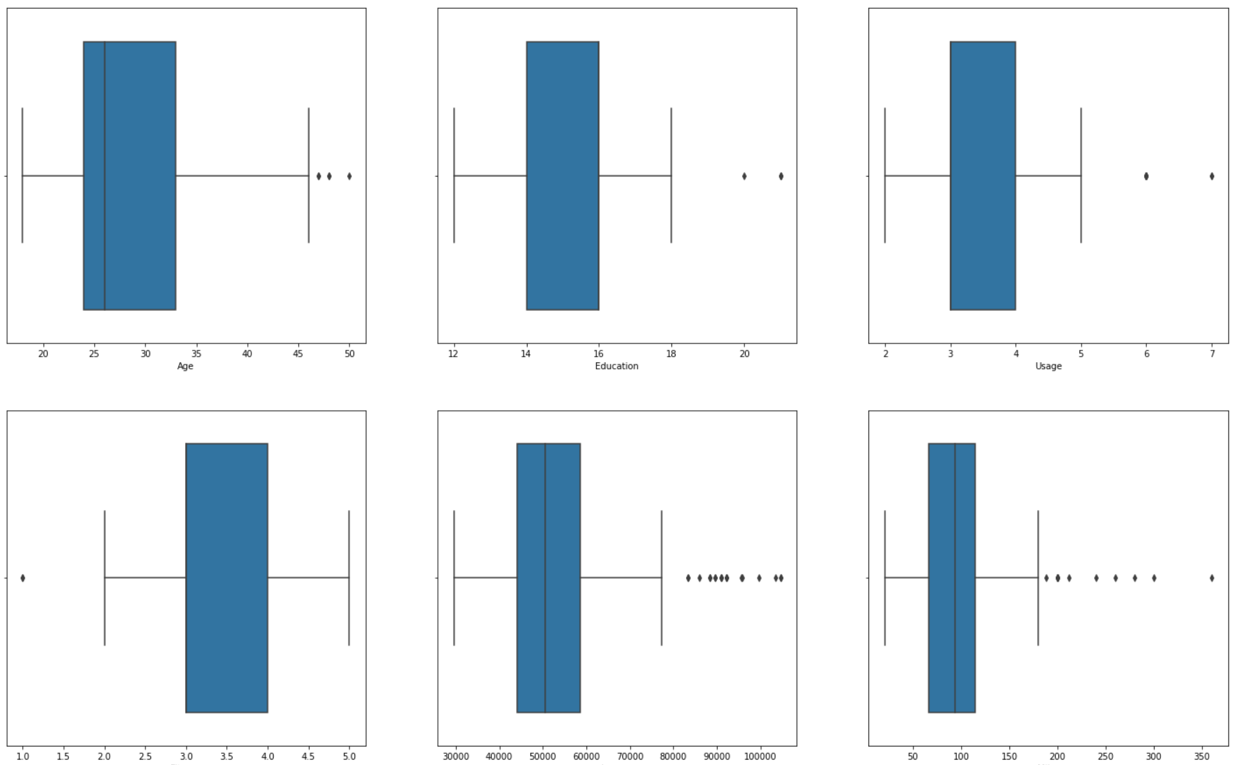
==> Partnered customers are more than single customers

==> Again KP281 is widely used treadmill by customers

==> KP281 is used 3 times a week more followed by KP481 and KP781

Boxplot

In [77]:
```
fig, axis = plt.subplots(nrows=2, ncols=3, figsize=(25, 12))
fig.subplots_adjust(top=1.1)

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

plt.show()
```



Observations

Income and miles have more outliers, lets try to remove them.

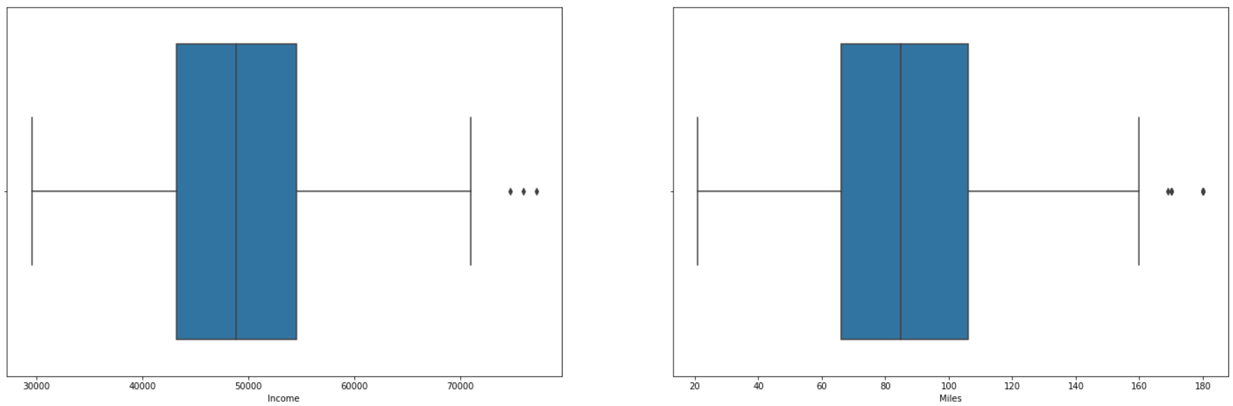
Outliers Removal

In [78]:
```
def remove_outliers(df,col):
  q1 = df[col].quantile(0.25)
  q3 = df[col].quantile(0.75)
  iqr = q3 - q1
  ul =  q3 + 1.5*iqr
```

```
  ll = q1 - 1.5*iqr

  return df[(df[col] > ll) & (df[col] < ul)]
```

In [79]:
```
df_miles = remove_outliers(df_afit,"Miles")
df_income = remove_outliers(df_afit,"Income")
```

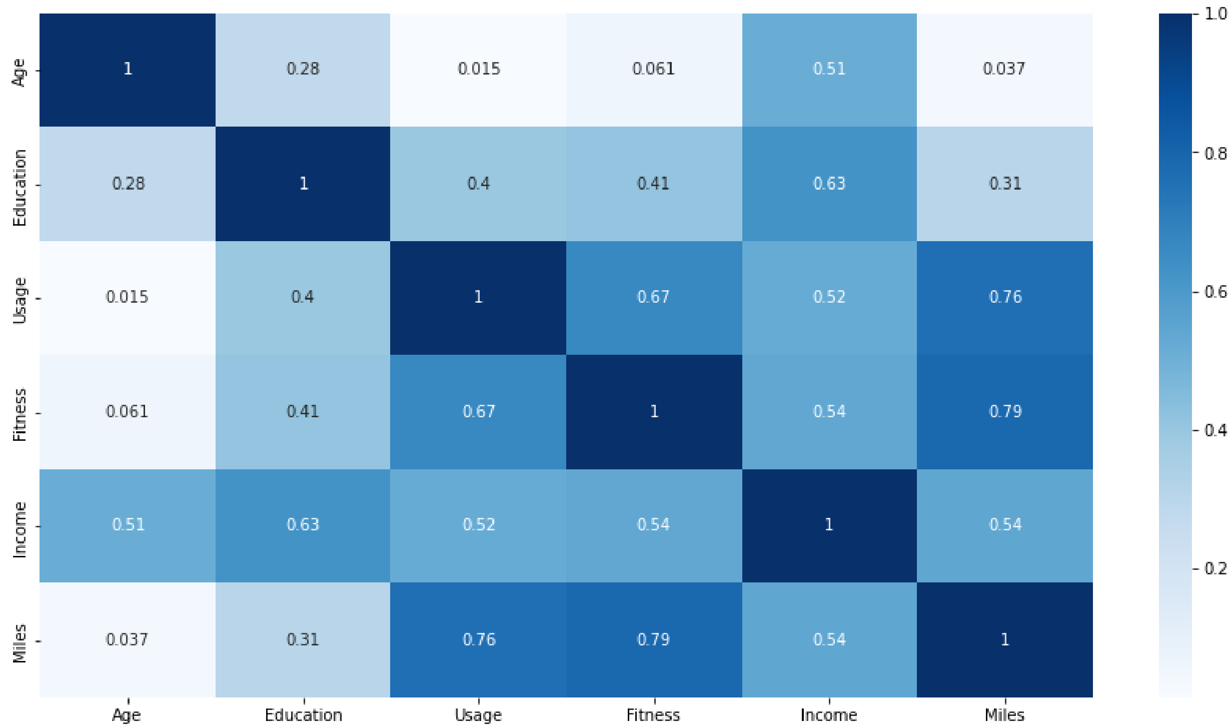
In [87]:
```
fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(25, 6))
fig.subplots_adjust(top=1.1)

sns.boxplot(data=df_income, x="Income", orient='h', ax=axis[0])
sns.boxplot(data=df_miles, x="Miles", orient='h', ax=axis[1])

plt.show()
```



Outliers are removed now

Correlation using heat map

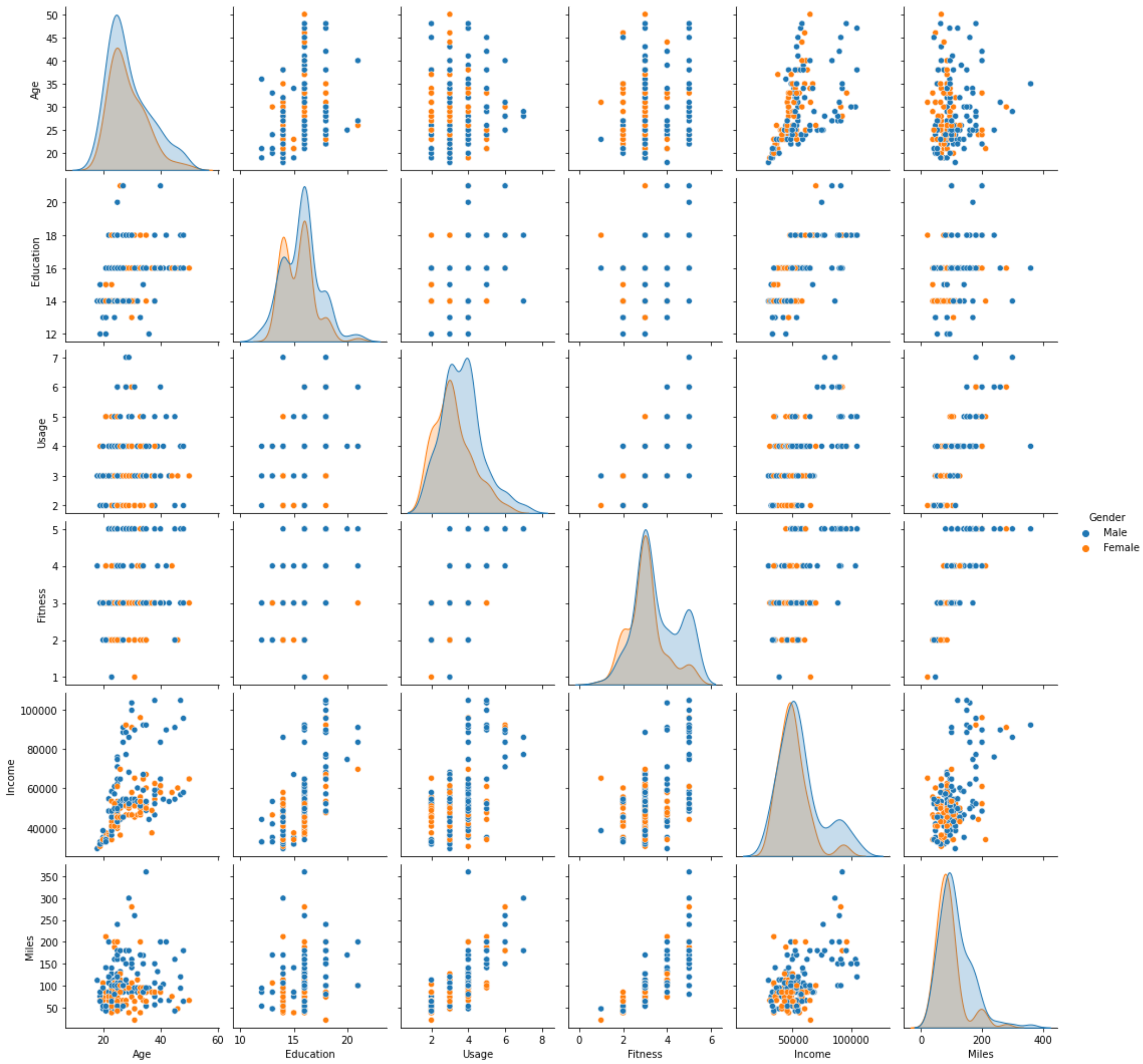
In [88]:
```
plt.figure(figsize =(15,8))
sns.heatmap(df_afit.corr(),cmap="Blues",annot=True)
plt.show()
```



Correlation using pairplot

In [89]:
```
plt.figure(figsize =(15,8))
sns.pairplot(data = df_afit, hue = 'Gender')
plt.show()
```

```
<Figure size 1080x576 with 0 Axes>
```

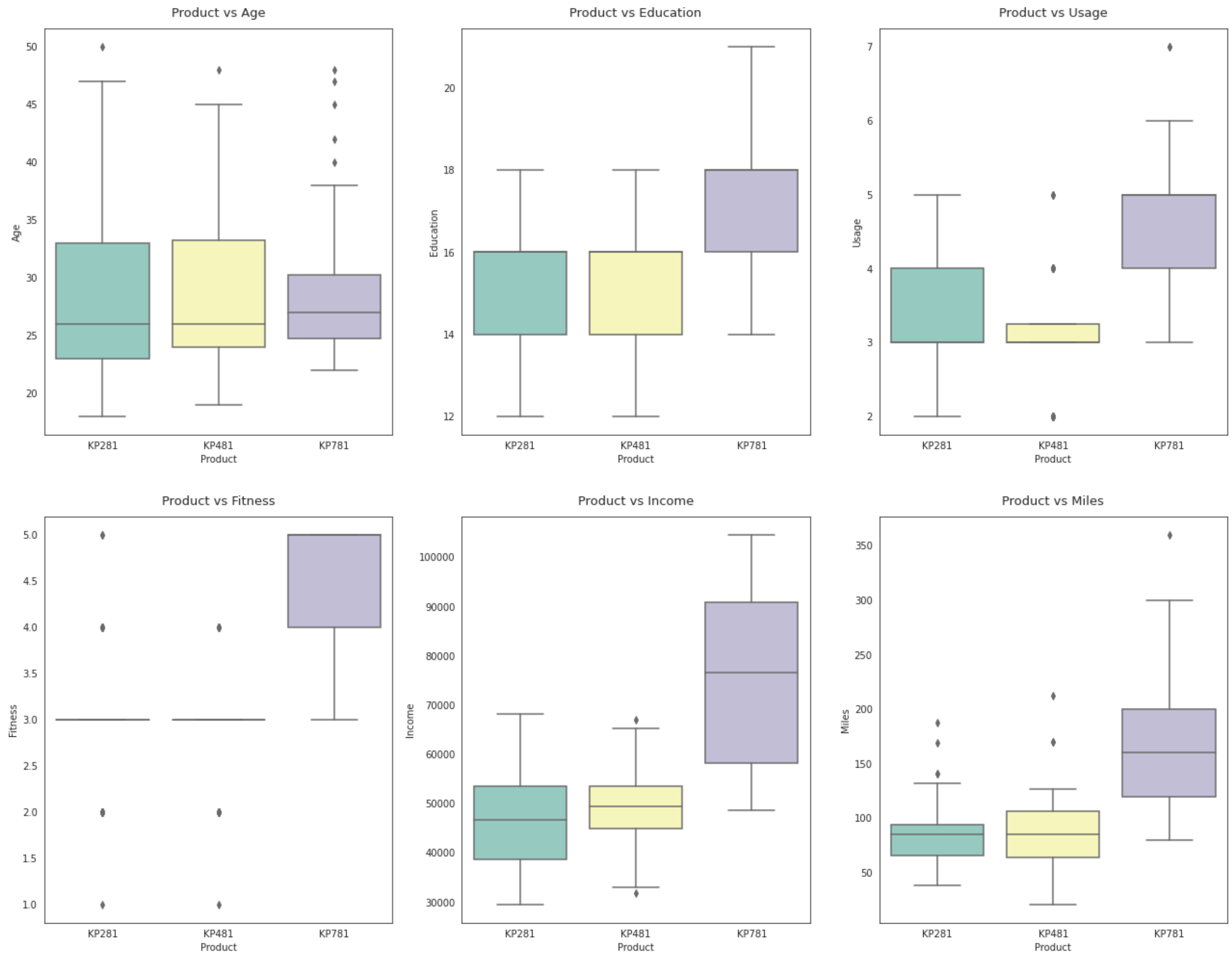


Checking if following features have any effect on the product purchased

Age, Education, Usage, Fitness, Income, Miles

In [91]:

```
attrs = ['Age', 'Education', 'Usage', 'Fitness', 'Income', 'Miles']
sns.set_style("white")
fig, axs = plt.subplots(nrows=2, ncols=3, figsize=(22, 12))
fig.subplots_adjust(top=1.2)
count = 0
for i in range(2):
  for j in range(3):
    sns.boxplot(data=df_afit, x='Product', y=attrs[count], ax=axs[i,j], palette='Set3'
    axs[i,j].set_title(f"Product vs {attrs[count]}", pad=12, fontsize=13)
    count += 1
```



Observations

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.

Marginal Probability with respect to Products

In [92]:
```
pd.crosstab(index = df_afit["Product"],columns = df_afit["Gender"],margins = True)
```

Out[92]:

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

In [93]:
```
# Probability of female customers buying products
df_afit_female = df_afit[df_afit["Gender"] == 'Female']
len(df_afit_female) / len(df_afit)
```

Out[93]:
```
0.4222222222222222
```

In [94]:
```
# Probability of Male customers buying products
df_afit_male = df_afit[df_afit["Gender"] == 'Male']
len(df_afit_male) / len(df_afit)
```

Out[94]:
```
0.5777777777777777
```

In [95]:
```
pd.crosstab(index = df_afit["Product"],columns = df_afit["MaritalStatus"],margins = Tr
```

Out[95]:

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

In [96]:

```
# Prob of partnered customers buying the products
df_afit_partnered = df_afit[df_afit["MaritalStatus"] == 'Partnered']
len(df_afit_partnered) / len(df_afit)
```

Out[96]:

```
0.5944444444444444
```

In [98]:

```
# Prob of buying products by customers who are single
df_afit_single = df_afit[df_afit["MaritalStatus"] == 'Single']
len(df_afit_single) / len(df_afit)
```

Out[98]:

```
0.40555555555555556
```

In [99]:

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

Out[99]:

Usage	2	3	4	5	6	7	All
Product							
KP281	19	37	22	2	0	0	80
KP481	14	31	12	3	0	0	60
KP781	0	1	18	12	7	2	40
All	33	69	52	17	7	2	180

In [100...

```
# Probability of customers using any threadmil 2 times a week
df_afit_usage2 =  df_afit[df_afit["Usage"] == 2]
len(df_afit_usage2) / len(df_afit)
```

Out[100]:

```
0.18333333333333332
```

In [102...

```
# Probability of customers using any threadmil 3 times a week
df_afit_usage3 =  df_afit[df_afit["Usage"] == 3]
len(df_afit_usage3) / len(df_afit)
```

Out[102]:

```
0.38333333333333336
```

In [103...

```
# Probability of customers using any threadmil 4 times a week
df_afit_usage4 =  df_afit[df_afit["Usage"] == 4]
len(df_afit_usage4) / len(df_afit)
```

Out[103]:

```
0.28888888888888886
```

Observations

==> Probability of female customers buying products = 0.422

==> Probability of Male customers buying products = 0.577

==> Probability of partnered customers buying the products = 0.594

==> Probability of buying products by customers who are single = 0.405

==> Probability of customers using any treadmil 2 times a week = 0.183

==> Probability of customers using any treadmil 3 times a week = 0.383

==> Probability of customers using any treadmil 4 times a week = 0.288

Conditional Probability

In [ ]:
```
pd.crosstab(index = df_afit["Product"],columns = df_afit["Fitness"],margins = True)
```

In [104…
```
# Lets focus on fitness level 3
df_afit_fitness3 = df_afit[df_afit["Fitness"] == 3]
```

Probability of customers buying treadmill, given fitness level 3

In [105…
```
#Prob of customers having fitness level 3 with threadmill KP281
df_afit_kp281 = df_afit_fitness3[df_afit_fitness3["Product"] == 'KP281']
len(df_afit_kp281) / len(df_afit_fitness3)
```

Out[105]:
```
0.5567010309278351
```

In [106…
```
#Prob of customers having fitness level 3 with threadmill KP481
df_afit_kp481 = df_afit_fitness3[df_afit_fitness3["Product"] == 'KP481']
len(df_afit_kp481) / len(df_afit_fitness3)
```

Out[106]:
```
0.4020618556701031
```

In [107…
```
#Prob of customers having fitness level 3 with threadmill KP781
df_afit_kp781 = df_afit_fitness3[df_afit_fitness3["Product"] == 'KP781']
len(df_afit_kp781) / len(df_afit_fitness3)
```

Out[107]:
```
0.041237113402061855
```

Probability of buying any threadmil given male customers

In [108…
```
df_afit_male.head()
```

Out[108]:

	Product	Age	Category_Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Category
0	KP281	18	Young	Male	14	Single	3	4	29562	
1	KP281	19	Young	Male	15	Single	2	3	31836	
3	KP281	19	Young	Male	12	Single	3	3	32973	
4	KP281	20	Young	Male	13	Partnered	4	2	35247	
7	KP281	21	Young	Male	13	Single	3	3	32973	

In [109…
```
pd.crosstab(index = df_afit["Product"],columns = df_afit["Gender"],margins = True)
```

Out[109]:

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

In [110…
```
#Probability of buying KP281 given male customers
df_afit_male_kp281 = df_afit_male[df_afit_male["Product"] == 'KP281']
len(df_afit_male_kp281) / len(df_afit_male)
```

Out[110]:
```
0.38461538461538464
```

In [111…
```
#Probability of buying KP481 given male customers
df_afit_male_kp481 = df_afit_male[df_afit_male["Product"] == 'KP481']
len(df_afit_male_kp481) / len(df_afit_male)
```

Out[111]:
```
0.2980769230769231
```

In [112…
```
#Probability of buying KP781 given male customers
df_afit_male_kp781 = df_afit_male[df_afit_male["Product"] == 'KP781']
len(df_afit_male_kp781) / len(df_afit_male)
```

Out[112]:
```
0.3173076923076923
```

Probability of buying any threadmil given female customers

In [113…
```
df_afit_female.head()
```

Out[113]:

	Product	Age	Category_Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Categor
2	KP281	19	Young	Female	14	Partnered	4	3	30699	
5	KP281	20	Young	Female	14	Partnered	3	3	32973	
6	KP281	21	Young	Female	14	Partnered	3	3	35247	
9	KP281	21	Young	Female	15	Partnered	2	3	37521	
11	KP281	22	Young	Female	14	Partnered	3	2	35247	

In [114…
```
#Probability of buying KP281 given female customers
df_afit_female_kp281 = df_afit_female[df_afit_female["Product"] == 'KP281']
len(df_afit_female_kp281) / len(df_afit_female)
```

Out[114]: 0.5263157894736842

In [115…
```
#Probability of buying KP481 given female customers
df_afit_female_kp481 = df_afit_female[df_afit_female["Product"] == 'KP481']
len(df_afit_female_kp481) / len(df_afit_female)
```

Out[115]: 0.3815789473684211

In [116…
```
#Probability of buying KP781 given female customers
df_afit_female_kp781 = df_afit_female[df_afit_female["Product"] == 'KP781']
len(df_afit_female_kp781) / len(df_afit_female)
```

Out[116]: 0.09210526315789473

Probability of buying any threadmil given Marital status

In [117…
```
pd.crosstab(index = df_afit["Product"],columns = df_afit["MaritalStatus"],margins = Tr
```

Out[117]:

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

In [118…
```
df_afit_partnered.head()
```

Out[118]:

	Product	Age	Category_Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Category
2	KP281	19	Young	Female	14	Partnered	4	3	30699	
4	KP281	20	Young	Male	13	Partnered	4	2	35247	
5	KP281	20	Young	Female	14	Partnered	3	3	32973	
6	KP281	21	Young	Female	14	Partnered	3	3	35247	
9	KP281	21	Young	Female	15	Partnered	2	3	37521	

In [119…

```
#Probability of buying KP281 given married customers
df_afit_partnered_kp281 = df_afit_partnered[df_afit_partnered["Product"] == 'KP281']
len(df_afit_partnered_kp281) / len(df_afit_partnered)
```

Out[119]:

```
0.4485981308411215
```

In [120…

```
#Probability of buying KP481 given married customers
df_afit_partnered_kp481 = df_afit_partnered[df_afit_partnered["Product"] == 'KP481']
len(df_afit_partnered_kp481) / len(df_afit_partnered)
```

Out[120]:

```
0.3364485981308411
```

In [122…

```
#Probability of buying KP781 given married customers
df_afit_partnered_kp781 = df_afit_partnered[df_afit_partnered["Product"] == 'KP781']
len(df_afit_partnered_kp781) / len(df_afit_partnered)
```

Out[122]:

```
0.21495327102803738
```

Observations

Probability of customers buying treadmill, given fitness level 3

p[kp281|fitness3] = 0.556

p[kp481|fitness3] = 0.402

p[kp781|fitness3] = 0.041

Probability of buying any threadmil given male customers

p[kp281|male] = 0.384

p[kp481|male] = 0.298

p[kp781|male] = 0.317

Probability of buying any threadmil given female customers

p[kp281|female] = 0.526

p[kp481|female] = 0.381

p[kp781|female] = 0.092

Probability of buying any threadmil given Marital status

p[kp281|married] = 0.448

p[kp481|married] = 0.336

p[kp781|married] = 0.214

Customer Profiling

In [123…

```
df_afit["Category_Age"].value_counts()
```

Out[123]:

```
Young     113
Adult      55
Aged       12
Name: Category_Age, dtype: int64
```

In [124…

```
df_afit["Category_income"].value_counts()
```

Out[124]:

```
medium    129
low        32
high       19
Name: Category_income, dtype: int64
```

In [125…

```
df_afit["Category_miles"].value_counts()
```

Out[125]:

```
Basic_workout      114
Medium_workout      60
Intense_workout      6
Name: Category_miles, dtype: int64
```

Observations

Age -

==> There are 113 - young, 55 - adult, 12 - old aged customers in given dataset.

Income -

==> There are 129 customers with medium income, 32 customers with low income and 19 customers with very high income

Miles -

==> There are 114 customers who do basic workout, 60 customers who do medium workout and 6 customers who do intense workout

Business Insights

Summary Profiles: Market Audience: Young to Middle-Aged Adults (Ages 20-40) for all models

KP281: Best Valued; Most affordable model. Motivating Cardio. Recommended for the average consumer, Sedentary to Moderate activity

KP481: Mid-grade model. Moderate to High Activity.

KP781:Luxury Grade Model; Full body workout with immersive technology, Recommended forfitness fanatics or seasoned runners, Consumers with Higher Income.

Market toward young to middle-aged males with higher incomes

Actionable Insights

Younger people with 16 and above years of education go for 'KP781' model and less people with 16 years of education tend to go for the other two models.

People with less age and aiming for higher miles goes for the KP781 product. People with all categories of ages aiming for lower miles will go for the KP281 product

KP781 customers are fit, more often men than women, and also have an income range that stretches higher (~50k-100k+), which matches an earlier observation we had that income and fitness have a positive relationship.

KP281 appears to be a mass-appeal product, with the highest number of overall customers and an equal distribution of male and female users.

KP481 sits in the middle of the three products, with not as many overall customers, but appealing to the some of the same types of consumers as those who buy KP281. We know that customers of this product tend to use it slighly less often per week.

KP281 and KP481 once again show similarity in that the majority of their customers are within similar income brackets (~35k-60k) and are about the same fitness level.

Marital Status does not appear to affect product choice, though when looking at KP781, thosewho are partnered have higher fitness levels than those who are single

KP781 customers are fit, more often men than women, and also have an income range that stretches higher (~50k-100k+), which matches an earlier observation we had that income and fitness have a positive relationship.

KP281 appears to be a mass-appeal product, with the highest number of overall customers and an equal distribution of male and female users.

KP481 sits in the middle of the three products, with not as many overall customers, but appealing to the some of the same types of consumers as those who buy KP281. We know that customers of this product tend to use it slighly less often per week.

KP281 and KP481 once again show similarity in that the majority of their customers are within similar income brackets (~35k-60k) and are about the same fitness level

Marital Status does not appear to affect product choice, though when looking at KP781, those who are partnered have higher fitness levels than those who are single

Recommendations

KP281 is most frequent product and mass appealing product so always have it in the stock

Customer who is Partnered, is more likely to purchase the product. We need to do more survey why singles are not purchasing more KP481 is tend to be purcahsed mostly by the medium income users. so if income between 40 to 80K they tend to go for KP481.

As per the data most of the customers are having 16 years of education and we need to enquiry why more than 16 education peoples are not purchasing.

Males with higer income tend to buy KP781 treadmill than the females with higher income so availability of kp781 is advisable for male and higher income customers

People aimimg for more than 100 miles are going for KP781 only so finding the reason will boost the other model sales