Cardio Good Fitness Case Study - Descriptive Statistics

The market research team at AdRight is assigned the task to identify the profile of the typical customer for each treadmill product offered by CardioGood Fitness. The market research team decides to investigate whether there are differences across the product lines with respect to customer characteristics. The team decides to collect data on individuals who purchased a treadmill at a CardioGoodFitness retail store during the prior three months. The data are stored in the CardioGoodFitness.csv file.

The team identifies the following customer variables to study:

*product purchased, TM195, TM498, or TM798;

*gender;

*age, in years;

*education, in years;

*relationship status, single or partnered;

*annual household income ;

*average number of times the customer plans to use the treadmill each week;

*average number of miles the customer expects to walk/run each week;

*and self-rated fitness on an 1-to-5 scale, where 1 is poor shape and 5 is excellent shape.

Perform descriptive analytics to create a customer profile for each CardioGood Fitness treadmill product line.

In [1]:
```
%pwd
```

Out[1]:
```
'C:\\Users\\PC'
```

In [2]:
```
import os

# Get the current working directory
current_directory = os.getcwd()
print(f"Current Working Directory: {current_directory}")

# Change the current working directory to a new path
```

```
new_directory = "C:\\Users\\PC\\Downloads\\GL_P"
os.chdir(new_directory)

# Verify the change
updated_directory = os.getcwd()
print(f"Updated Working Directory: {updated_directory}")
```

```
Current Working Directory: C:\Users\PC
Updated Working Directory: C:\Users\PC\Downloads\GL_P
```

In [4]:
```
# Load the necessary packages

import numpy as np
import pandas as pd
```

In [5]:
```
# Load the Cardio Dataset

mydata = pd.read_csv('CardioGoodFitness.csv')
```

In [6]:
```
mydata.head()
```

Out[6]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	TM195	18	Male	14	Single	3	4	29562	112
1	TM195	19	Male	15	Single	2	3	31836	75
2	TM195	19	Female	14	Partnered	4	3	30699	66
3	TM195	19	Male	12	Single	3	3	32973	85
4	TM195	20	Male	13	Partnered	4	2	35247	47

In [7]:
```
mydata.describe(include="all")
```

Out[7]:

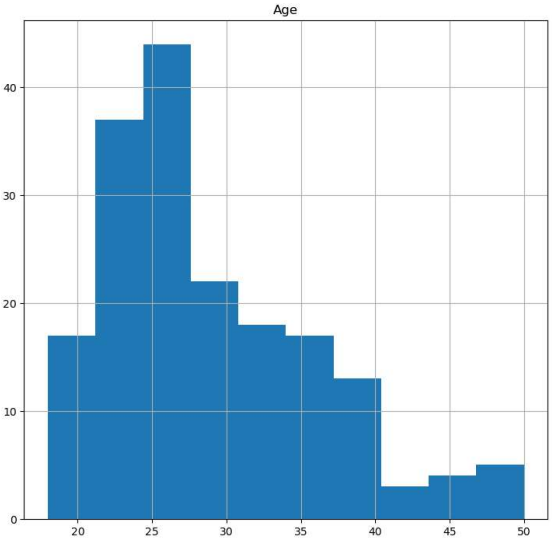
	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Incom
count	180	180.000000	180	180.000000	180	180.000000	180.000000	180.00000
unique	3	NaN	2	NaN	2	NaN	NaN	NaI
top	TM195	NaN	Male	NaN	Partnered	NaN	NaN	NaI
freq	80	NaN	104	NaN	107	NaN	NaN	NaI
mean	NaN	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53719.57777
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16506.68422
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29562.00000
25%	NaN	24.000000	NaN	14.000000	NaN	3.000000	3.000000	44058.75000
50%	NaN	26.000000	NaN	16.000000	NaN	3.000000	3.000000	50596.50000
75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58668.00000
max	NaN	50.000000	NaN	21.000000	NaN	7.000000	5.000000	104581.00000

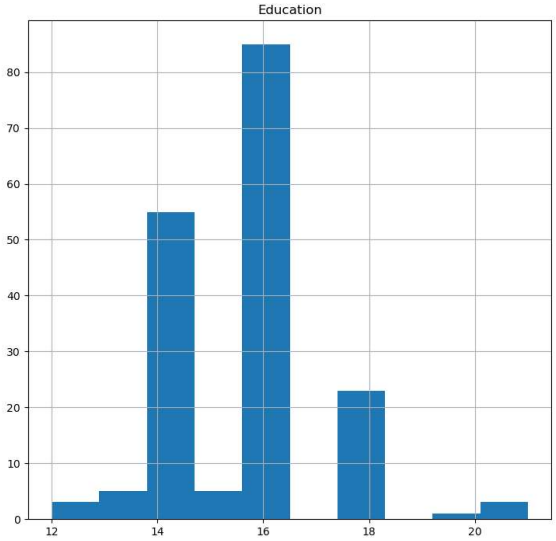
In [8]:
```
mydata.info()
```

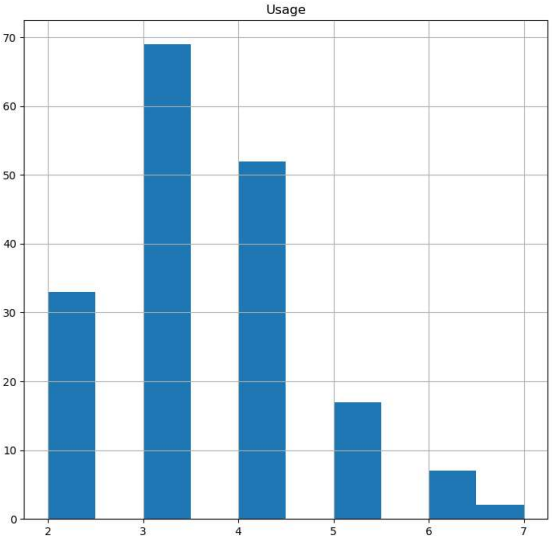
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Product        180 non-null    object
 1   Age            180 non-null    int64 
 2   Gender         180 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
```

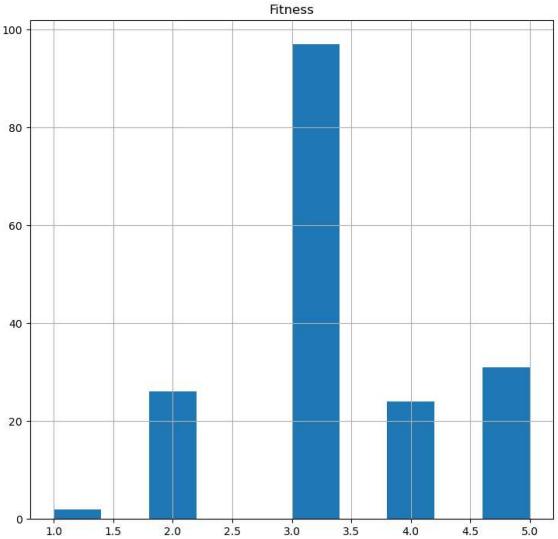
In [9]:
```
import matplotlib.pyplot as plt
%matplotlib inline

mydata.hist(figsize=(20,30))
```

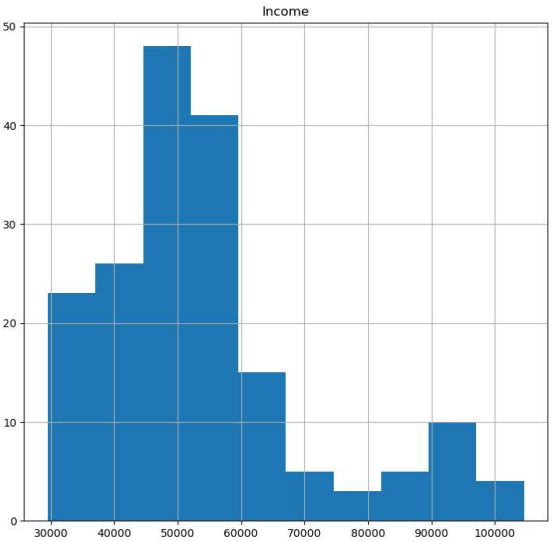
Out[9]:
```
array([[<Axes: title={'center': 'Age'}>,
        <Axes: title={'center': 'Education'}>],
       [<Axes: title={'center': 'Usage'}>,
        <Axes: title={'center': 'Fitness'}>],
       [<Axes: title={'center': 'Income'}>,
        <Axes: title={'center': 'Miles'}>]], dtype=object)
```

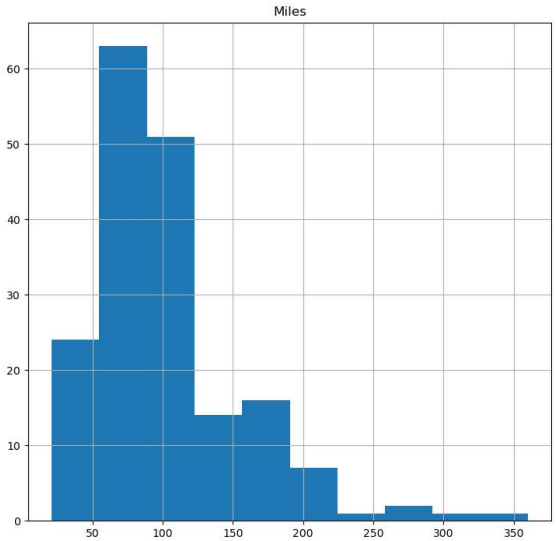








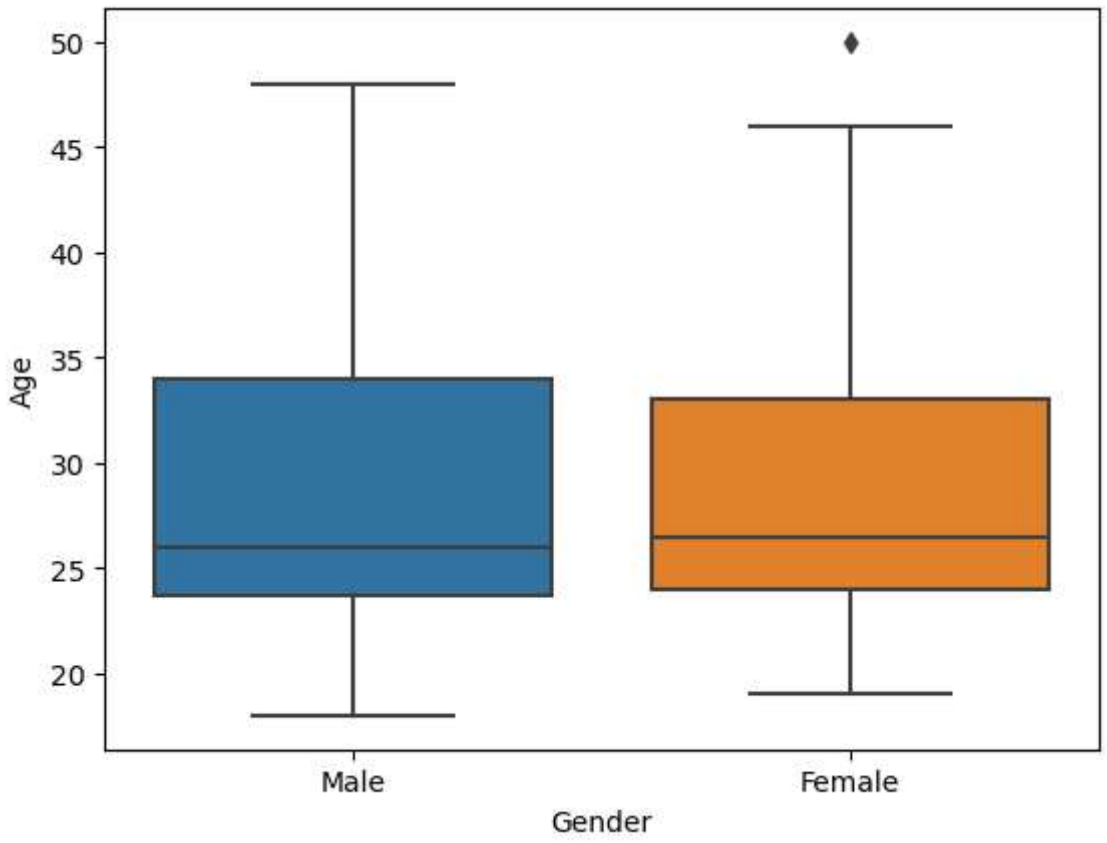




In [10]:
```
import seaborn as sns
```

```
sns.boxplot(x="Gender", y="Age", data=mydata)
```

Out[10]:
```
<Axes: xlabel='Gender', ylabel='Age'>
```



In [11]:
```
pd.crosstab(mydata['Product'], mydata['Gender'] )
```

Out[11]:

Product	Female	Male
TM195	40	40
TM498	29	31
TM798	7	33

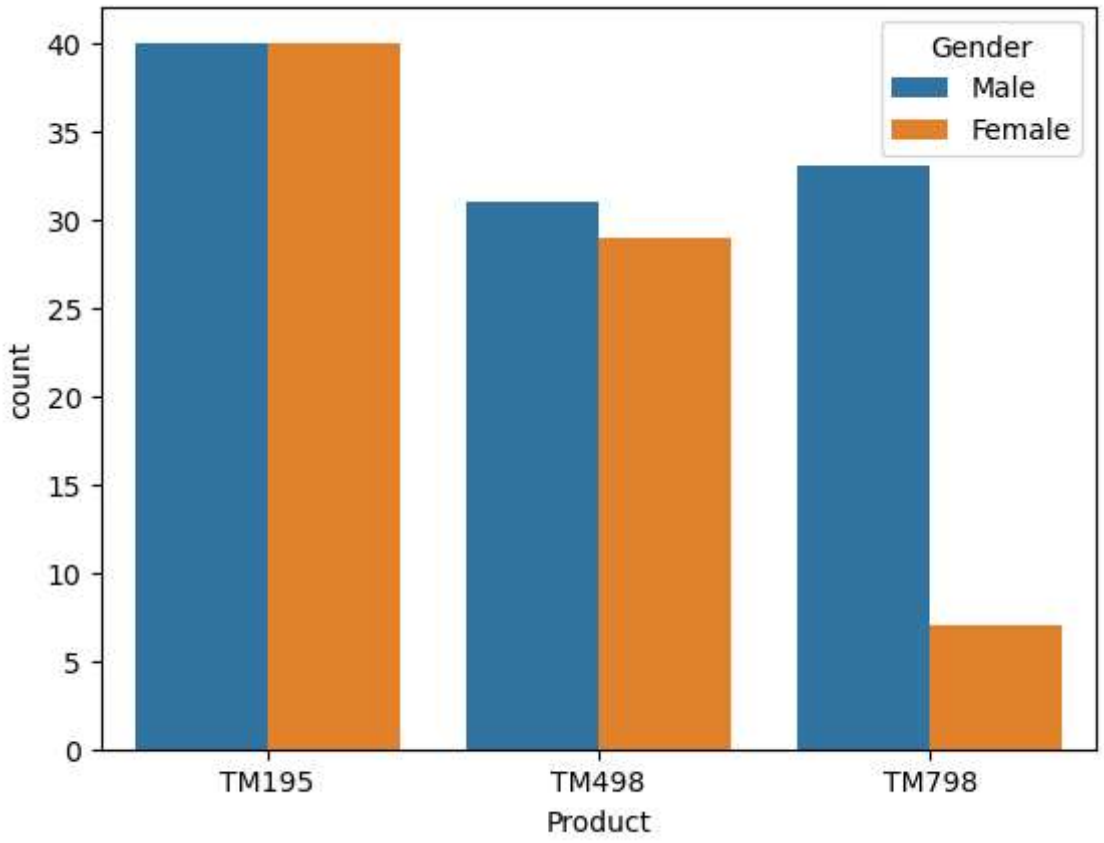
In [12]:
```
pd.crosstab(mydata['Product'],mydata['MaritalStatus'] )
```

Out[12]:

Product	Partnered	Single
TM195	48	32
TM498	36	24
TM798	23	17

In [13]:
```
sns.countplot(x="Product", hue="Gender", data=mydata)
```

Out[13]:
```
<Axes: xlabel='Product', ylabel='count'>
```



In [14]:
```
pd.pivot_table(mydata, index=['Product', 'Gender'],
               columns=[ 'MaritalStatus'])
```

Out[14]:

		Age		Education		Fitness		
	MaritalStatus	Partnered	Single	Partnered	Single	Partnered	Single	Partnered
Product	Gender							
TM195	Female	28.333333	28.692308	14.888889	15.538462	2.851852	2.923077	46153.777778
	Male	31.380952	25.631579	15.428571	14.473684	2.857143	3.263158	50028.000000
TM498	Female	30.000000	28.142857	15.200000	15.214286	2.933333	2.785714	49724.800000
	Male	30.380952	25.200000	15.285714	14.500000	2.904762	3.000000	49378.285714
TM798	Female	29.000000	24.333333	17.500000	18.333333	5.000000	4.000000	84972.250000
	Male	30.000000	28.928571	17.421053	16.928571	4.631579	4.642857	81431.368421

In [15]:
```
sns.pairplot(mydata)
```

```
C:\Users\PC\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The fig
ure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[15]:
```
<seaborn.axisgrid.PairGrid at 0x1a451df93d0>
```

In [16]:
```
mydata['Age'].std()
```

Out[16]:
```
6.943498135399795
```

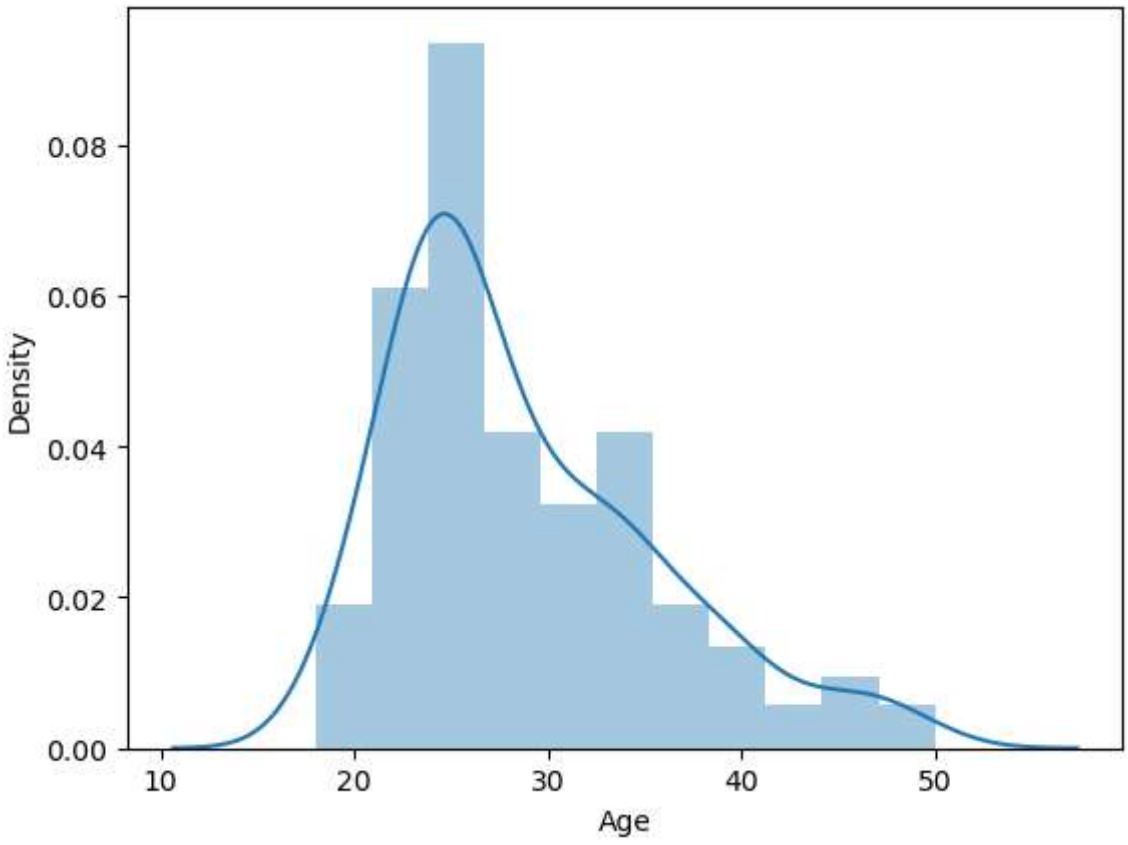
In [17]:
```
mydata['Age'].mean()
```

Out[17]:
```
28.788888888888888
```

In [18]:
```
sns.distplot(mydata['Age'])
```

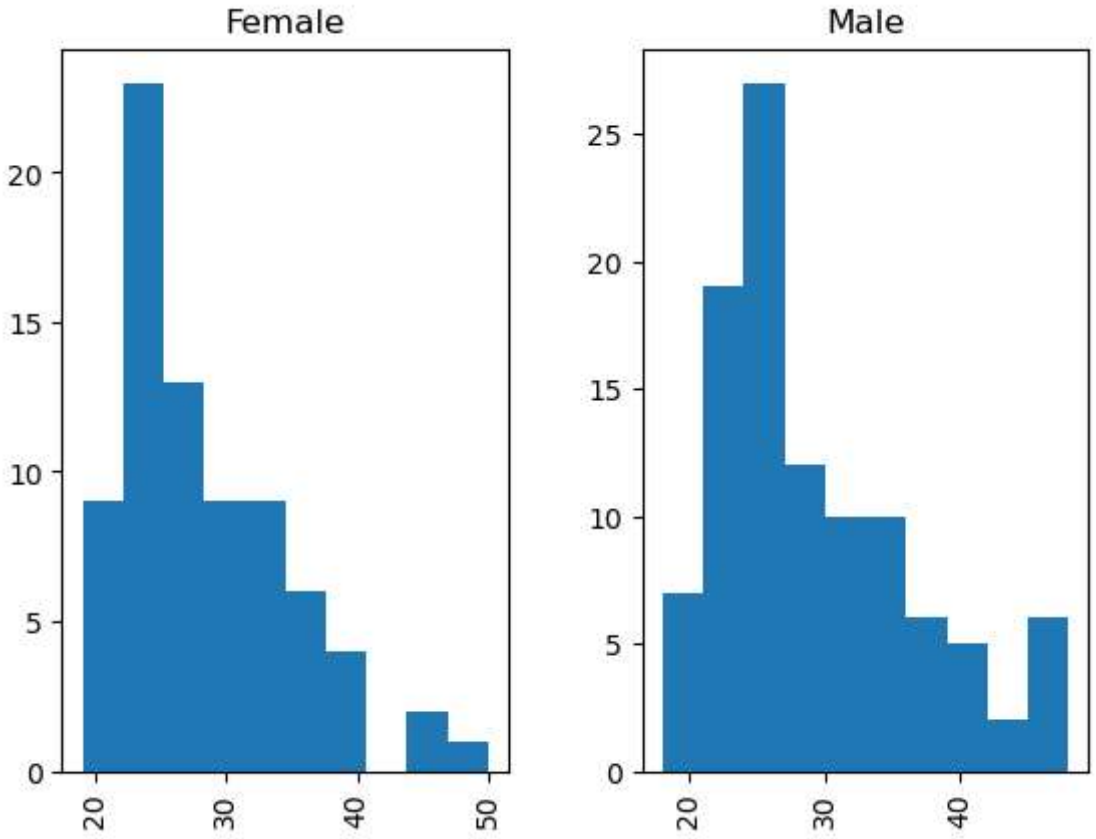
```
C:\Users\PC\AppData\Local\Temp\ipykernel_16464\3718817797.py:1: UserWarning:

`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(mydata['Age'])
```

Out[18]: <Axes: xlabel='Age', ylabel='Density'>



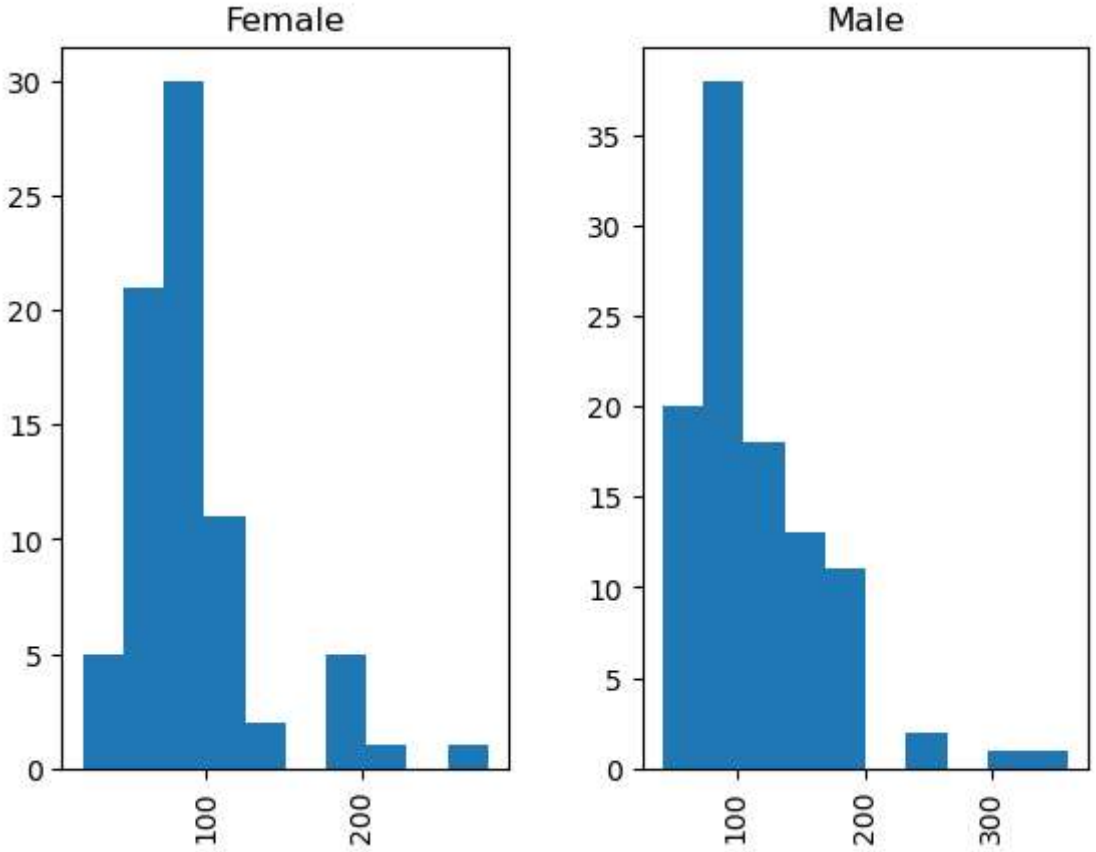
In [19]:
```
mydata.hist(by='Gender', column = 'Age')
```

Out[19]:
```
array([<Axes: title={'center': 'Female'}>,
       <Axes: title={'center': 'Male'}>], dtype=object)
```



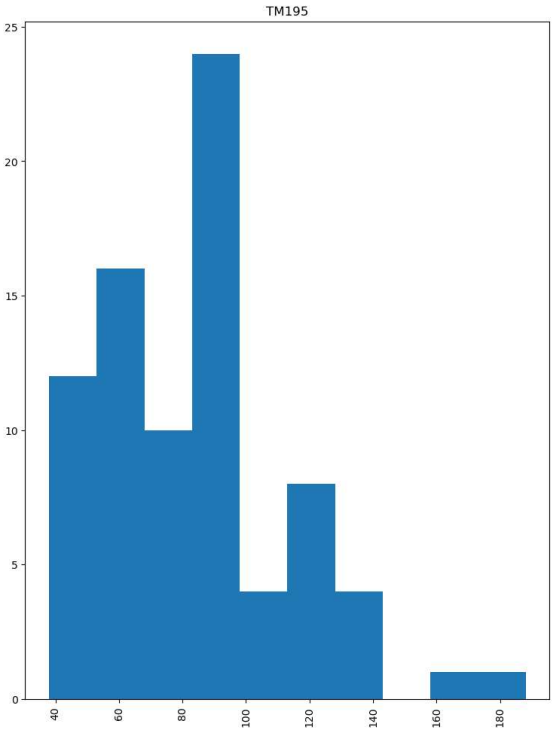
In [20]:
```
mydata.hist(by='Gender',column = 'Miles')
```

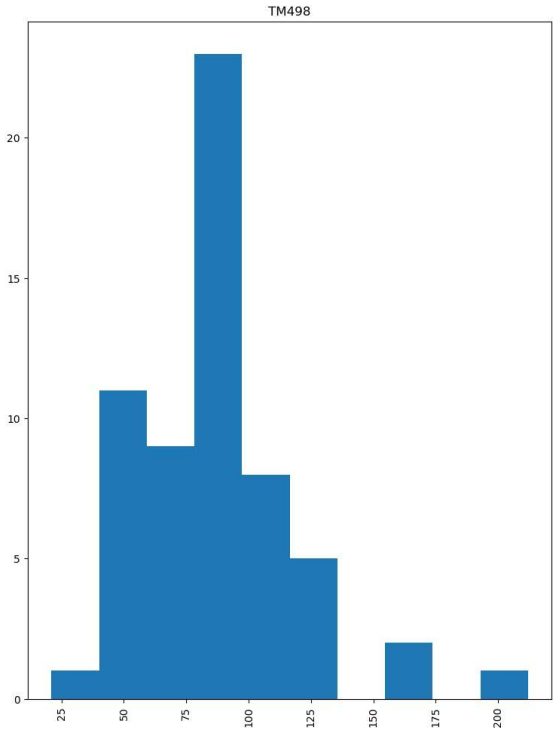
Out[20]:
```
array([<Axes: title={'center': 'Female'}>,
       <Axes: title={'center': 'Male'}>], dtype=object)
```

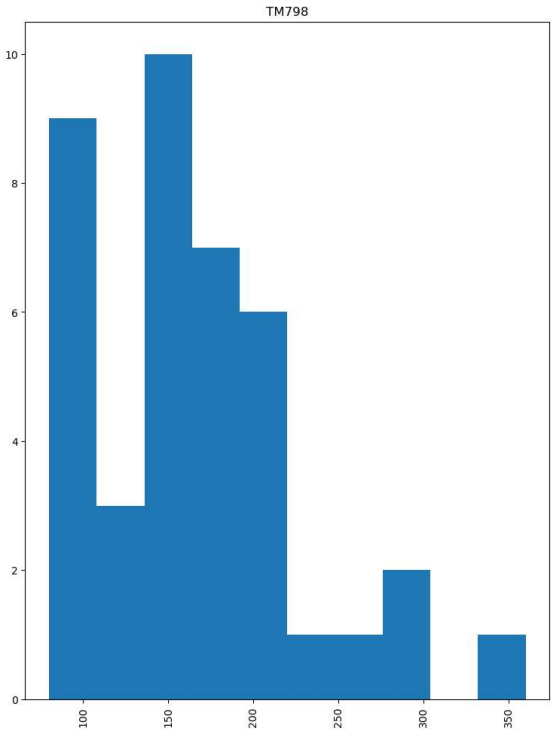


In [21]:
```
mydata.hist(by='Product',column = 'Miles', figsize=(20,30))
```

Out[21]:
```
array([[<Axes: title={'center': 'TM195'}>,
        <Axes: title={'center': 'TM498'}>],
       [<Axes: title={'center': 'TM798'}>, <Axes: >]], dtype=object)
```







In [29]:
```
numeric_columns = mydata.select_dtypes(include=['number'])
```

In [30]:
```
corr = numeric_columns.corr()
corr
```

Out[30]:

	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

In [31]:
```
sns.heatmap(corr, annot=True)
```

Out[31]:
```
<Axes: >
```

In [ ]:
```

```