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]:
```
# Load the necessary packages

import numpy as np
import pandas as pd
```

In [9]:
```
# Load the Cardio Dataset

mydata = pd.read_csv('C:\\Users\\hp\\Downloads\\CardioGoodFitness.csv')
```

In [7]:
```
mydata.head()
```

Out[7]:

	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 [8]:
```
mydata.describe(include="all")
```

Out[8]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mile
count	180	180.000000	180	180.000000	180	180.000000	180.000000	180.000000	180.00000

unique	3	NaN	2	NaN	2	NaN	NaN	NaN	Na
top	TM195	NaN	Male	NaN	Partnered	NaN	NaN	NaN	Na
freq	80	NaN	104	NaN	107	NaN	NaN	NaN	Na
mean	NaN	28.788889	NaN	15.572222	NaN	3.455556	3.311111	53719.577778	103.19444
std	NaN	6.943498	NaN	1.617055	NaN	1.084797	0.958869	16506.684226	51.86360
min	NaN	18.000000	NaN	12.000000	NaN	2.000000	1.000000	29562.000000	21.00000
25%	NaN	24.000000	NaN	14.000000	NaN	3.000000	3.000000	44058.750000	66.00000
50%	NaN	26.000000	NaN	16.000000	NaN	3.000000	3.000000	50596.500000	94.00000
75%	NaN	33.000000	NaN	16.000000	NaN	4.000000	4.000000	58668.000000	114.75000
max	NaN	50.000000	NaN	21.000000	NaN	7.000000	5.000000	104581.000000	360.00000

In [7]:

```
mydata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 9 columns):
Product          180 non-null object
Age              180 non-null int64
Gender           180 non-null object
Education        180 non-null int64
MaritalStatus    180 non-null object
Usage            180 non-null int64
Fitness          180 non-null int64
Income           180 non-null int64
Miles            180 non-null int64
dtypes: int64(6), object(3)
memory usage: 12.7+ KB
```

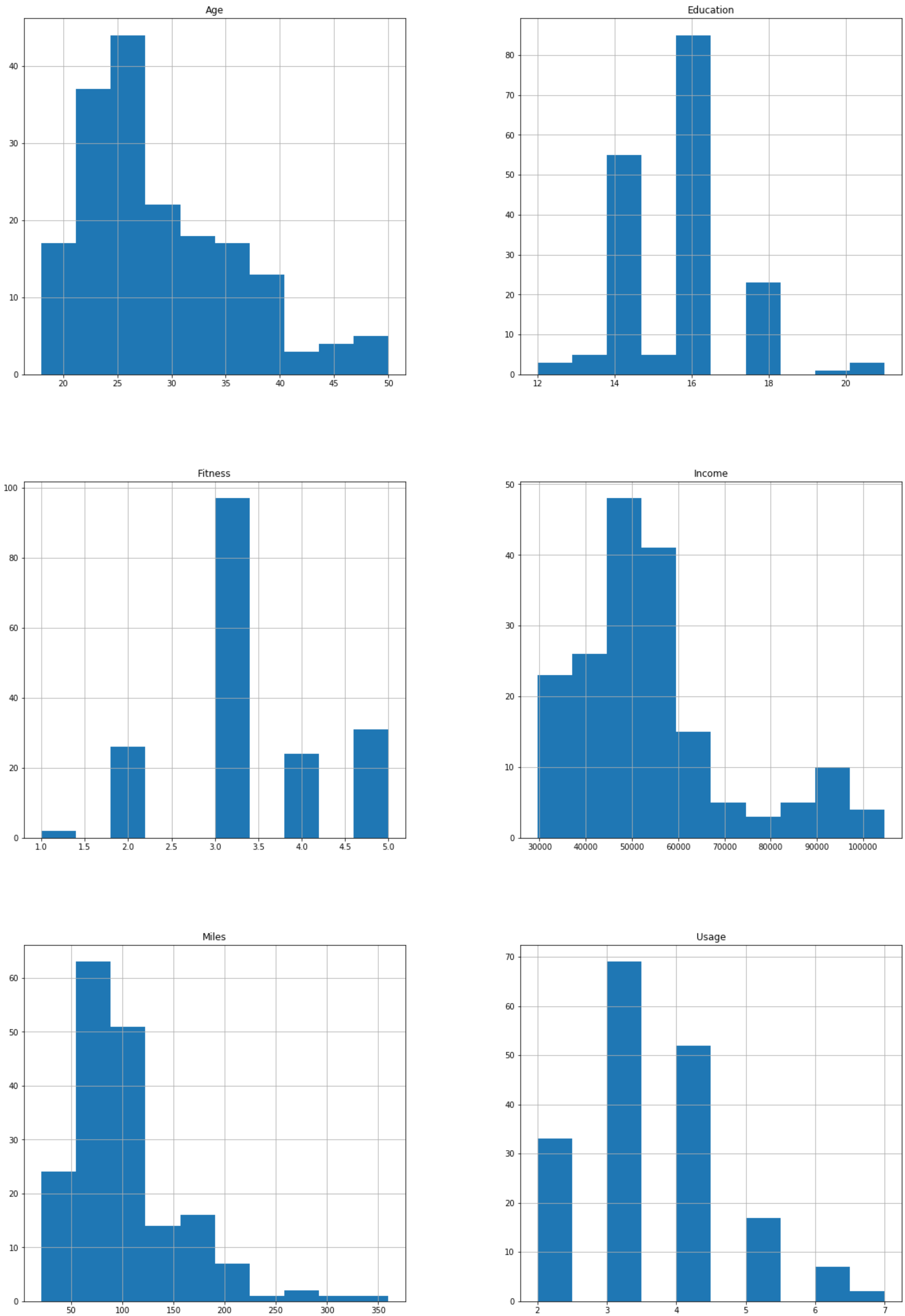
In [14]:

```
import matplotlib.pyplot as plt
%matplotlib inline

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

Out[14]:

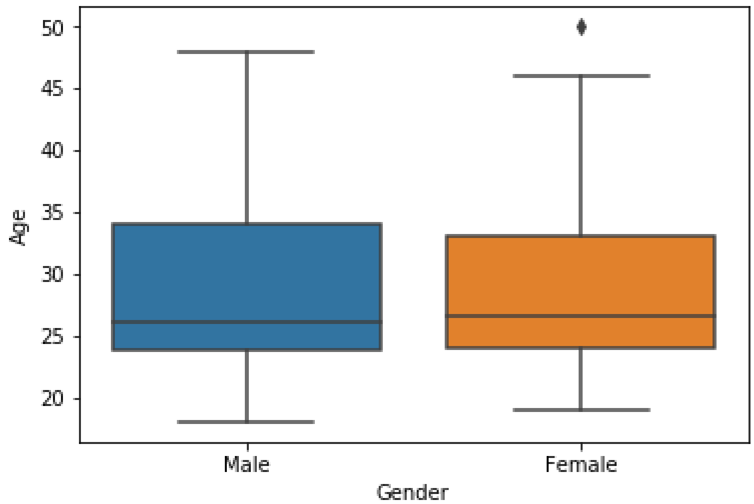
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x11501bac8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x1151e5b70>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x1156e1080>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x11580a0f0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x11585efd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x11587e048>]], dtype=object)
```



In [20]:
```
import seaborn as sns
```

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

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x1a199ac5c0>



In [21]:
```
pd.crosstab(mydata['Product'],mydata['Gender'] )
```

Out[21]:

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

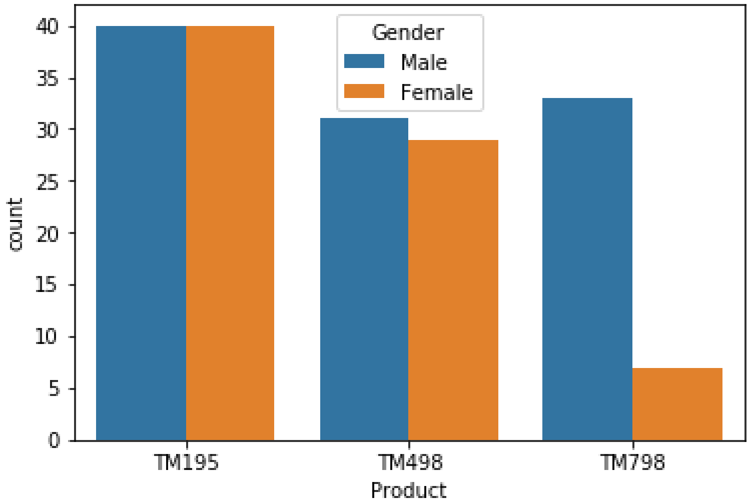
In [22]:
```
pd.crosstab(mydata['Product'],mydata['MaritalStatus'] )
```

Out[22]:

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

In [24]:
```
sns.countplot(x="Product", hue="Gender", data=mydata)
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x1a19c83390>



In [41]:
```
pd.pivot_table(mydata, index=['Product', 'Gender'],
                     columns=[ 'MaritalStatus'], aggfunc=len)
```

Out[41]:

		Age		Education		Fitness		Income		M
	MaritalStatus	**Partnered**	**Single**	**Partnered**	**Single**	**Partnered**	**Single**	**Partnered**	**Single**	**Partnered**
Product	**Gender**									
TM195	**Female**	27	13	27	13	27	13	27	13	27
	Male	21	19	21	19	21	19	21	19	21
TM498	**Female**	15	14	15	14	15	14	15	14	15
	Male	21	10	21	10	21	10	21	10	21
TM798	**Female**	4	3	4	3	4	3	4	3	4
	Male	19	14	19	14	19	14	19	14	19

In [42]:
```
pd.pivot_table(mydata,'Income', index=['Product', 'Gender'],
                     columns=[ 'MaritalStatus'])
```

Out[42]:

	MaritalStatus	**Partnered**	**Single**
Product	**Gender**		
TM195	**Female**	46153.777778	45742.384615
	Male	50028.000000	43265.842105
TM498	**Female**	49724.800000	48920.357143
	Male	49378.285714	47071.800000
TM798	**Female**	84972.250000	58516.000000
	Male	81431.368421	68216.428571

In [43]:
```
pd.pivot_table(mydata,'Miles', index=['Product', 'Gender'],
                     columns=[ 'MaritalStatus'])
```

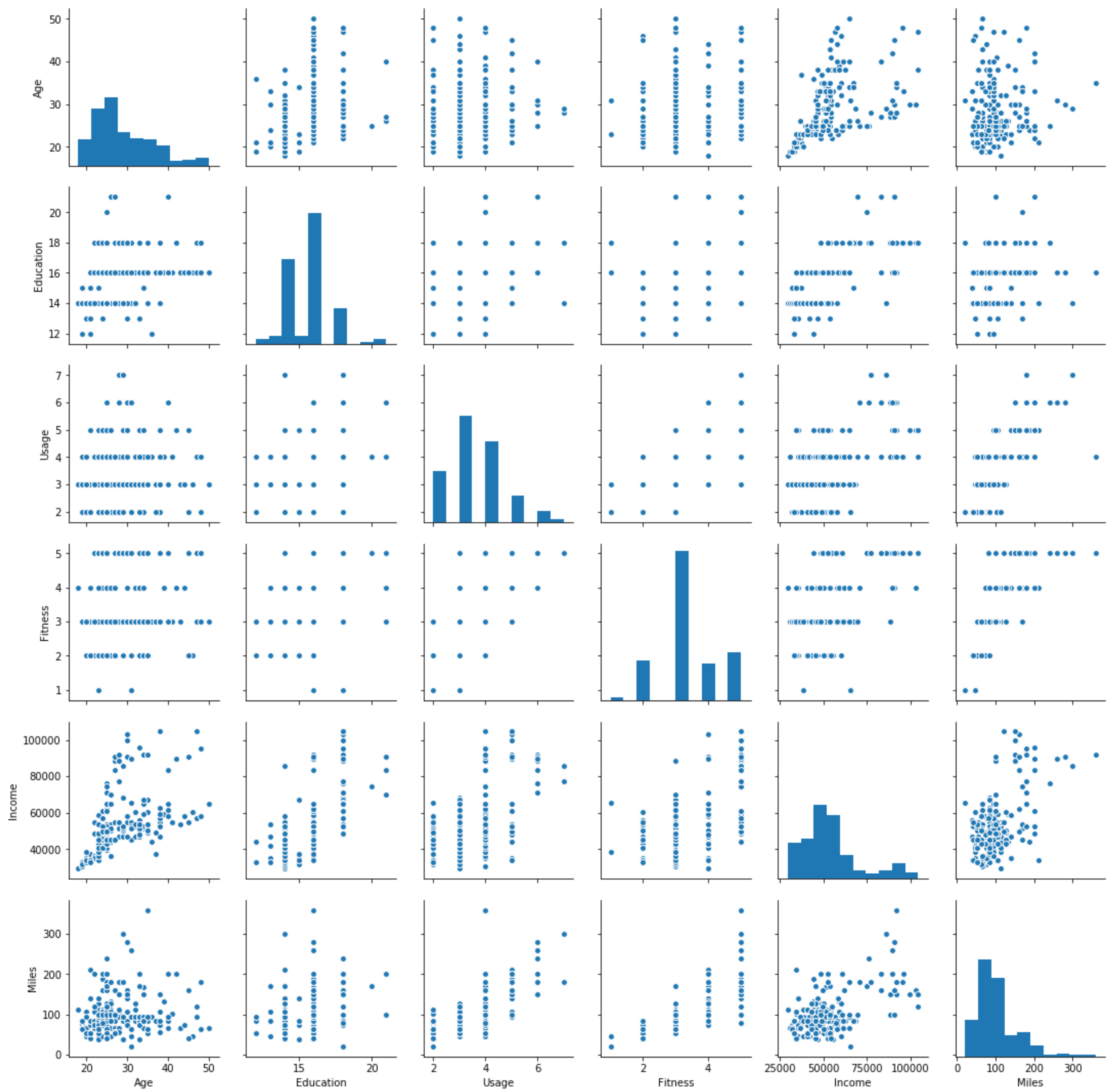
Out[43]:

	MaritalStatus	**Partnered**	**Single**
Product	**Gender**		
TM195	**Female**	74.925926	78.846154

	Male	80.190476	99.526316
TM498	**Female**	94.000000	80.214286
	Male	87.238095	91.100000
TM798	**Female**	215.000000	133.333333
	Male	176.315789	147.571429

In [44]:
```
sns.pairplot(mydata)
```

Out[44]: <seaborn.axisgrid.PairGrid at 0x1a19ed4898>



In [45]:
```
mydata['Age'].std()
```

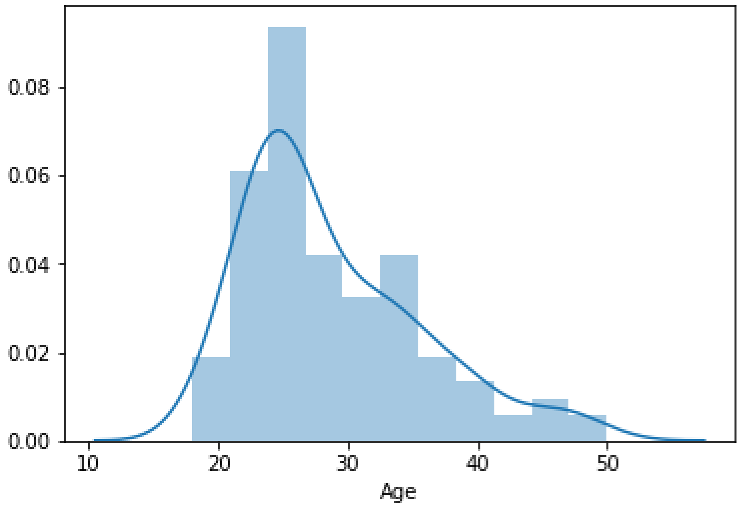
Out[45]: 6.9434981353997953

In [46]:
```
mydata['Age'].mean()
```

Out[46]: 28.788888888888888

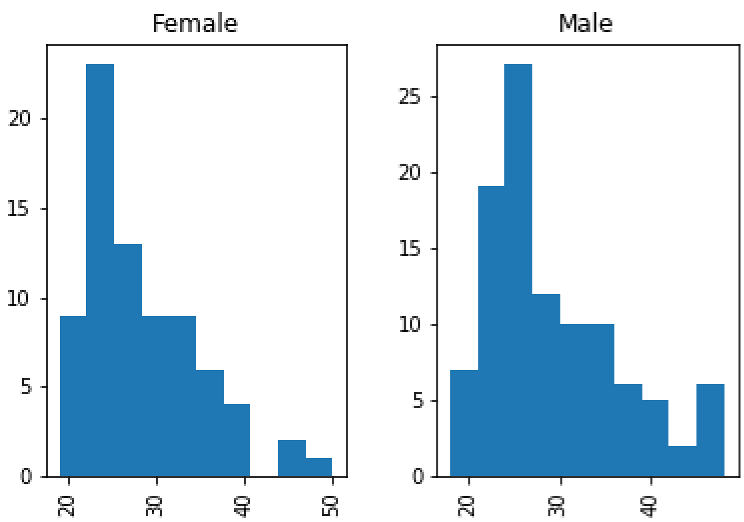
In [50]:
```
sns.distplot(mydata['Age'])
```

Out[50]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1b1bffd0>
```



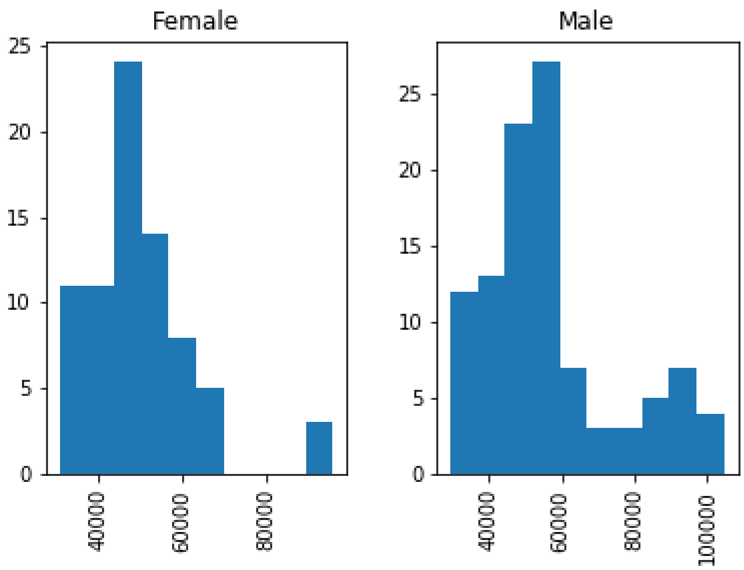
In [58]:
```
mydata.hist(by='Gender',column = 'Age')
```

Out[58]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x1a1bade860>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x1a1bc68518>], dtype=object)
```



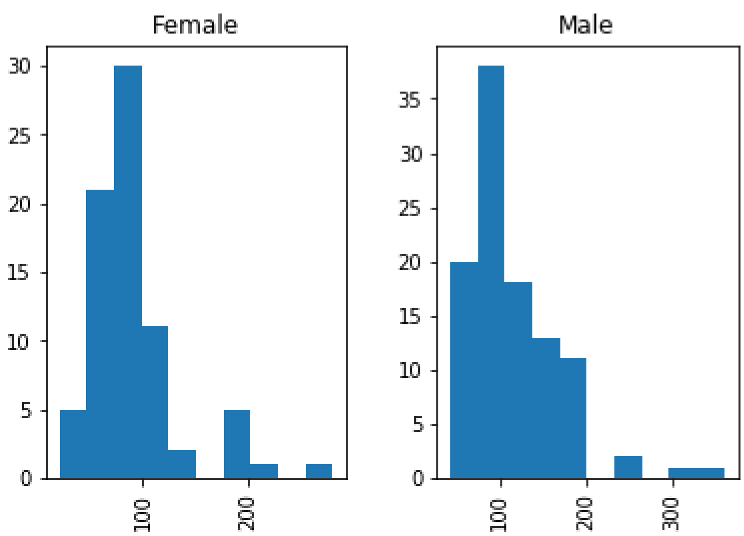
In [59]:
```
mydata.hist(by='Gender',column = 'Income')
```

Out[59]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x1a1bba48d0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x1a1b9cedd8>], dtype=object)
```



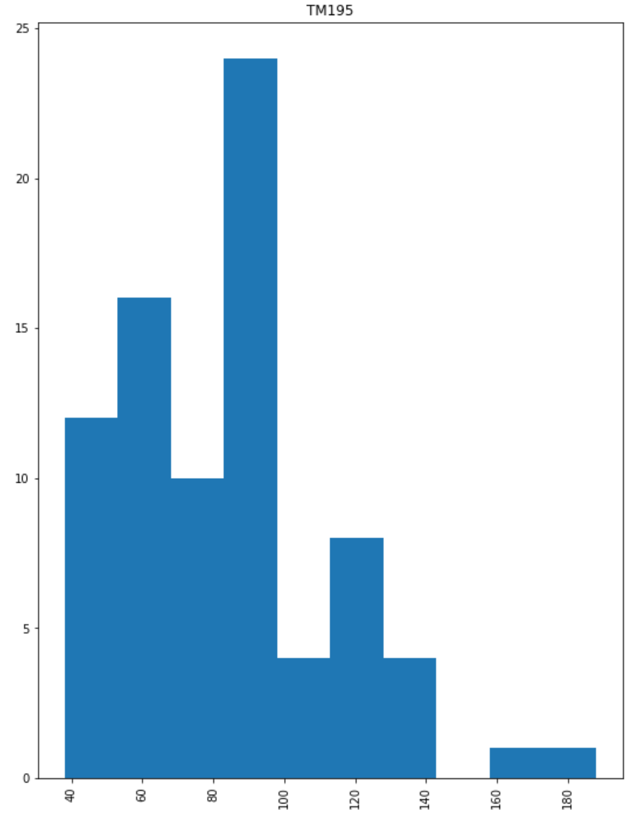
In [60]:
```
mydata.hist(by='Gender',column = 'Miles')
```

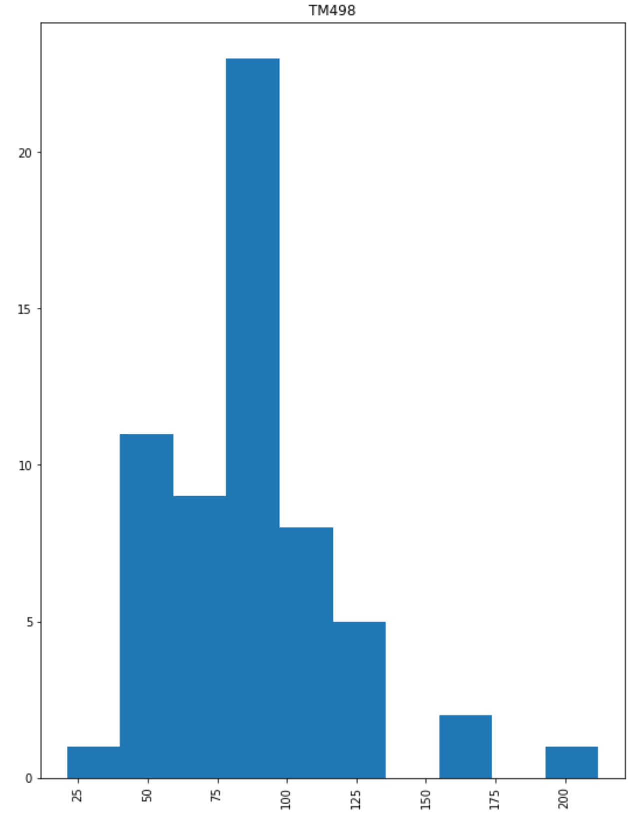
Out[60]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x1a1a5c3cc0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x1a19e4b828>], dtype=object)
```

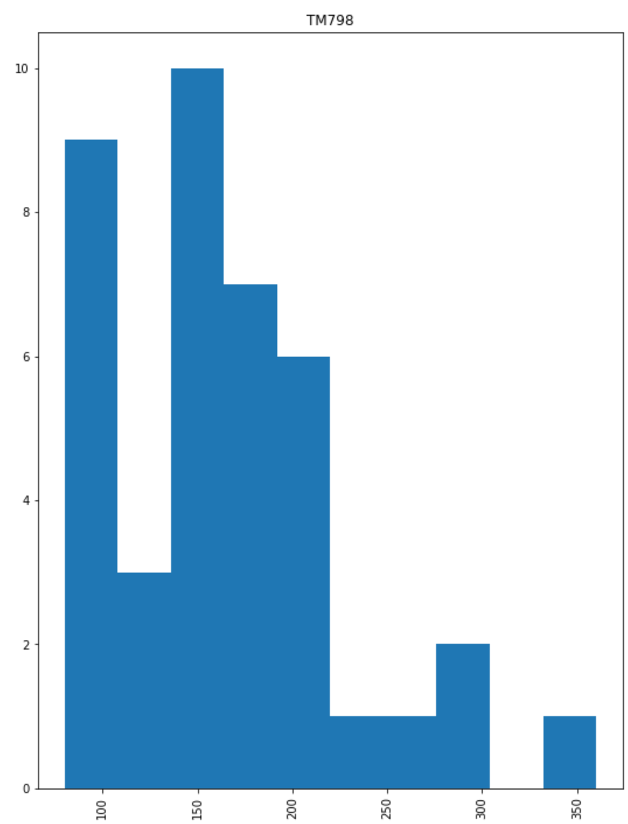


In [62]:
```
mydata.hist(by='Product',column = 'Miles', figsize=(20,30))
```

Out[62]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x1a1bfdd668>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x1a1c04e1d0>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x1a1c1271d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x1a1c15e710>]], dtype=object)
```







In [67]:

```
corr = mydata.corr()
corr
```

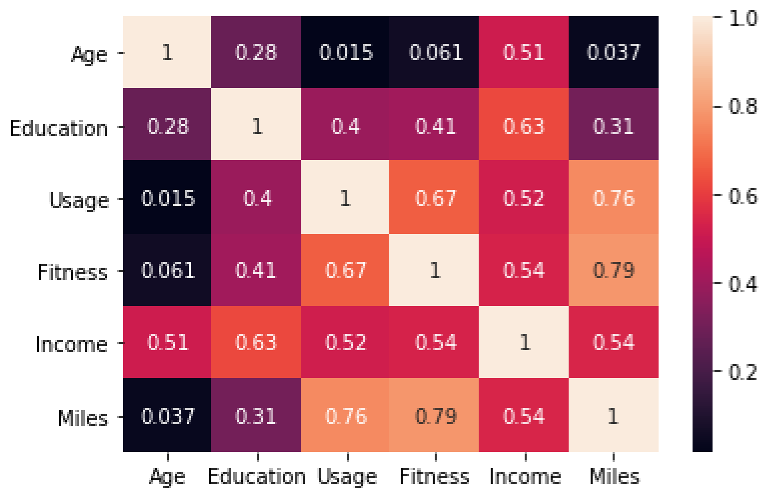
Out[67]:

	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 [66]:

```
sns.heatmap(corr, annot=True)
```

Out[66]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1cb58a20>



In [96]:

```
# Simple Linear Regression


#Load function from sklearn
from sklearn import linear_model

# Create linear regression object
regr = linear_model.LinearRegression()

y = mydata['Miles']
x = mydata[['Usage','Fitness']]

# Train the model using the training sets
regr.fit(x,y)
```

Out[96]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)

In [97]:

```
regr.coef_
```

Out[97]: array([20.21486334, 27.20649954])

In [98]:

```
regr.intercept_
```

Out[98]: -56.742881784648617

In [ ]:

```
# MilesPredicted = -56.74 + 20.21*Usage + 27.20*Fitness
```