Aerofit Analysis - Case Study

June 6, 2023

0.1 Bandi Saideva

0.2 Aerofit Case Study

0.3 Descriptive Statistics and Probability

0.4 Date: June 04, 2023

0.4.1 Imports

[1]:
```
# pip install -q seaborn
```

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import norm
import seaborn as sns
```

0.4.2 1. Import the dataset and do usual data analysis steps like checking the structure & characteristics of the dataset

[3]:
```
aerofit_df = pd.read_csv("aerofit.csv")
aerofit_df.head()
```

[3]:
```
  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
```

[4]:
```
aerofit_df.shape
```

[4]:
```
(180, 9)
```

[5]:
```
aerofit_df.describe()
```

[5]:
```
              Age   Education       Usage     Fitness         Income  \
count  180.000000  180.000000  180.000000  180.000000     180.000000
mean    28.788889   15.572222    3.455556    3.311111   53719.577778
```

```
std      6.943498   1.617055   1.084797   0.958869   16506.684226
min     18.000000  12.000000   2.000000   1.000000   29562.000000
25%     24.000000  14.000000   3.000000   3.000000   44058.750000
50%     26.000000  16.000000   3.000000   3.000000   50596.500000
75%     33.000000  16.000000   4.000000   4.000000   58668.000000
max     50.000000  21.000000   7.000000   5.000000  104581.000000

            Miles
count  180.000000
mean   103.194444
std     51.863605
min     21.000000
25%     66.000000
50%     94.000000
75%    114.750000
max    360.000000
```

- we can say if the mean is more far from median then the data has more outliers
- From the data we can observe that col Miles has more outliers

[6]:
```
aerofit_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
```

- Looks like no need to preprocess the data, the data looks cleaned.

Unique Values Detection

[7]:
```
aerofit_df['Product'].unique()
```

[7]:
```
array(['KP281', 'KP481', 'KP781'], dtype=object)
```

[8]:
```
aerofit_df['Age'].unique()
# we can categorize age column
```

[8]: array([18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 43, 44, 46, 47, 50, 45, 48, 42],
      dtype=int64)

```
[9]: aerofit_df['Gender'].unique()
```

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

```
[10]: aerofit_df['Education'].unique()
# we can categorize Education col
```

[10]: array([14, 15, 12, 13, 16, 18, 20, 21], dtype=int64)

```
[11]: aerofit_df['MaritalStatus'].unique()
```

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

```
[12]: aerofit_df['Usage'].unique()
```

[12]: array([3, 2, 4, 5, 6, 7], dtype=int64)

```
[13]: aerofit_df['Fitness'].unique()
```

[13]: array([4, 3, 2, 1, 5], dtype=int64)

```
[14]: aerofit_df['Income'].unique()
# we can categorize income col
```

```
[14]: array([ 29562,  31836,  30699,  32973,  35247,  37521,  36384,  38658,
        40932,  34110,  39795,  42069,  44343,  45480,  46617,  48891,
        53439,  43206,  52302,  51165,  50028,  54576,  68220,  55713,
        60261,  67083,  56850,  59124,  61398,  57987,  64809,  47754,
        65220,  62535,  48658,  54781,  48556,  58516,  53536,  61006,
        57271,  52291,  49801,  62251,  64741,  70966,  75946,  74701,
        69721,  83416,  88396,  90886,  92131,  77191,  52290,  85906,
       103336,  99601,  89641,  95866, 104581,  95508], dtype=int64)
```

```
[15]: aerofit_df['Miles'].unique()
# we can categorize miles col
```

```
[15]: array([112,  75,  66,  85,  47, 141, 103,  94, 113,  38, 188,  56, 132,
       169,  64,  53, 106,  95, 212,  42, 127,  74, 170,  21, 120, 200,
       140, 100,  80, 160, 180, 240, 150, 300, 280, 260, 360], dtype=int64)
```

```
[16]: # Categorizing Age col
np.min(aerofit_df['Age']), np.max(aerofit_df['Age'])
```

[16]: (18, 50)

[17]:
```python
# ages --> {18 - 25 : young, 26- 35: middle age, 36 - 50: old}

def age_group(x):
    if x >= 18 and x <= 25:
        return "Young"
    elif x >= 26 and x <= 35:
        return "Middle Age"
    else:
        return "Old"

aerofit_df["AgeGroup"] = aerofit_df['Age'].apply(age_group)
aerofit_df.head()
```

[17]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles AgeGroup
0    112    Young
1     75    Young
2     66    Young
3     85    Young
4     47    Young
```

[18]:
```python
aerofit_df.tail()
```

[18]:
```
    Product  Age Gender  Education MaritalStatus  Usage  Fitness  Income  \
175   KP781   40   Male         21        Single      6        5   83416
176   KP781   42   Male         18        Single      5        4   89641
177   KP781   45   Male         16        Single      5        5   90886
178   KP781   47   Male         18     Partnered      4        5  104581
179   KP781   48   Male         18     Partnered      4        5   95508

     Miles AgeGroup
175    200      Old
176    200      Old
177    160      Old
178    120      Old
179    180      Old
```

[19]:
```python
aerofit_df['AgeGroup'].value_counts()
```

[19]:
```
Young         79
Middle Age    73
```

```
Old           28
Name: AgeGroup, dtype: int64
```

[20]:
```
# Most of the customers are belong to young & Middle Age age group, so our ads
↪should must target young and Middle Age people
```

[21]:
```
# Categorizing Education col
np.min(aerofit_df['Education']), np.max(aerofit_df['Education'])
```

[21]:
```
(12, 21)
```

[22]:
```
# Education --> {12 : Schooling, 13 - 16: Graduation, > 16: PostGraduation}

def edu_group(x):
    if x == 12:
        return "Schooling"
    elif x >= 13 and x <= 16:
        return "Graduation"
    else:
        return "PostGraduation"

aerofit_df["Qualification"] = aerofit_df['Education'].apply(edu_group)
aerofit_df.head()
```

[22]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles AgeGroup Qualification
0    112    Young    Graduation
1     75    Young    Graduation
2     66    Young    Graduation
3     85    Young     Schooling
4     47    Young    Graduation
```

[23]:
```
aerofit_df["Qualification"].value_counts()
```

[23]:
```
Graduation        150
PostGraduation     27
Schooling           3
Name: Qualification, dtype: int64
```

[24]:
```
# most of the graudates are tend to buy our products.
```

[25]:
```python
# Categorizing Income col
np.min(aerofit_df['Income']), np.max(aerofit_df['Income'])
```

[25]: (29562, 104581)

[26]:
```python
# Income --> {29_562 - 40_000 : Low, 40_000 - 80_000: Medium, > 80_000: High}

def income_group(x):
    if x >= 29_562 and  x <= 40_000:
        return "Low"
    elif x > 40_000 and x <= 80_000:
        return "Medium"
    else:
        return "High"

aerofit_df["Status"] = aerofit_df['Income'].apply(income_group)
aerofit_df.head()
```

[26]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles AgeGroup Qualification Status
0    112    Young    Graduation    Low
1     75    Young    Graduation    Low
2     66    Young    Graduation    Low
3     85    Young     Schooling    Low
4     47    Young    Graduation    Low
```

[27]:
```python
aerofit_df["Status"].value_counts()
```

[27]:
```
Medium    129
Low        32
High       19
Name: Status, dtype: int64
```

[28]:
```python
# Most of the customers of medium income are interested in our products.
```

[29]:
```python
# Categorizing Miles col
np.min(aerofit_df['Miles']), np.max(aerofit_df['Miles'])
```

[29]: (21, 360)

[30]:
```python
# Miles --> {21 - 100 : Low, 100 - 200: Medium, > 200: High}
```

```python
def miles_group(x):
    if x >= 21 and  x <= 100:
        return "Low"
    elif x > 100 and x <= 200:
        return "Medium"
    else:
        return "High"

aerofit_df["MilesCat"] = aerofit_df['Miles'].apply(miles_group)
aerofit_df.head()
```

[30]:

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

   Miles AgeGroup Qualification Status MilesCat
0    112    Young    Graduation    Low   Medium
1     75    Young    Graduation    Low      Low
2     66    Young    Graduation    Low      Low
3     85    Young     Schooling    Low      Low
4     47    Young    Graduation    Low      Low
```

[31]:

```python
aerofit_df["MilesCat"].value_counts()
```

[31]:

```
Low       114
Medium     60
High        6
Name: MilesCat, dtype: int64
```

[32]:

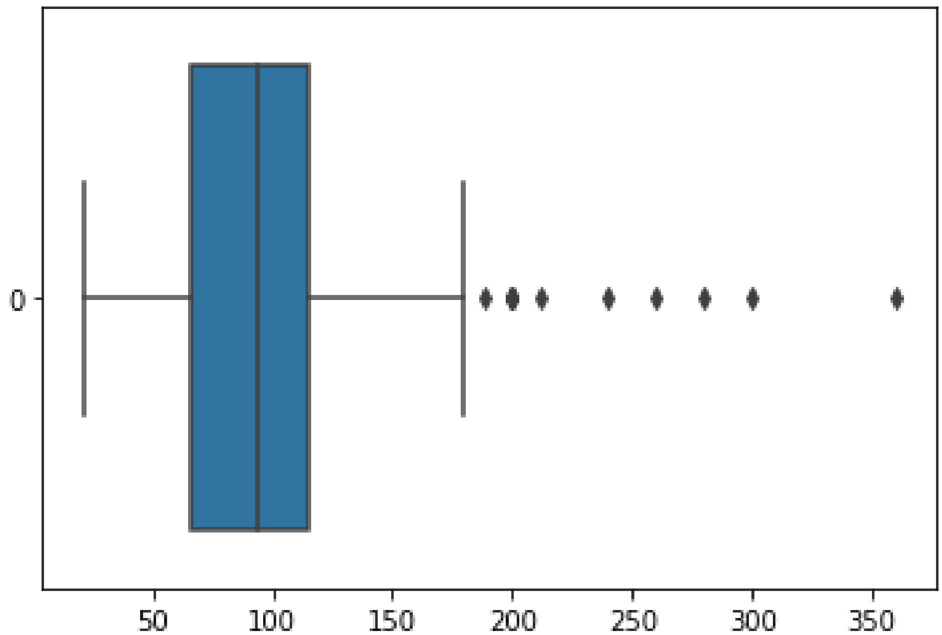
```python
# Most of the customers are using our product less.
```

Now the Data looks more Descriptive

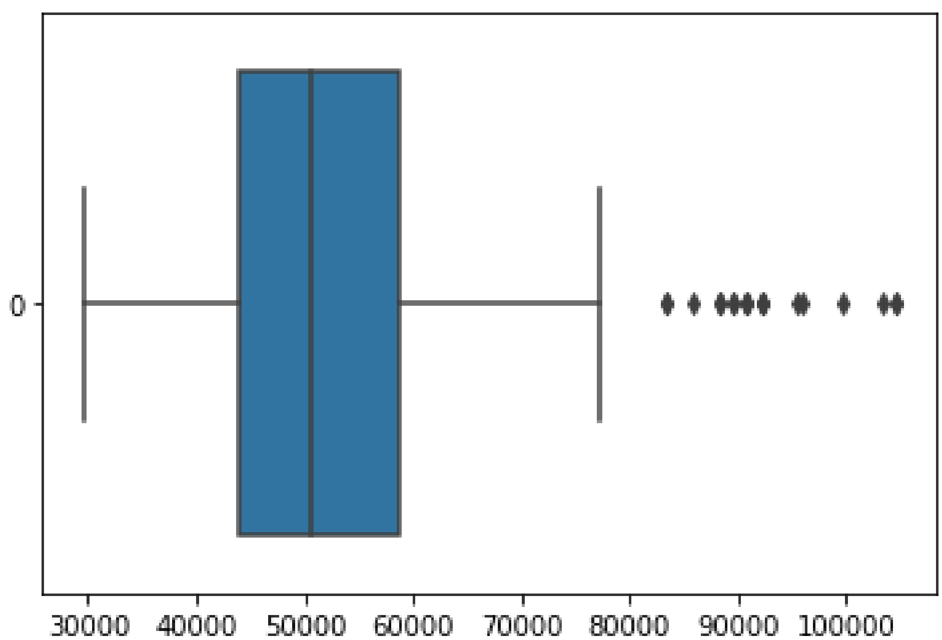
0.4.3 2. Detect Outliers (using boxplot, "describe" method by checking the difference between mean and median)

[33]:

```python
sns.boxplot(aerofit_df['Miles'], orient = 'h');
# most of the outliers in terms of miles are above 175
# I think these people are fitness freaks
```




```
[34]: sns.boxplot(aerofit_df['Income'], orient = 'h');
# income has also some outliers, where the customers income's are more than␣
↪80_0000
# these are the rich people, who are very likely to buy our products.
```



[35]:
```
aerofit_df.head()
```

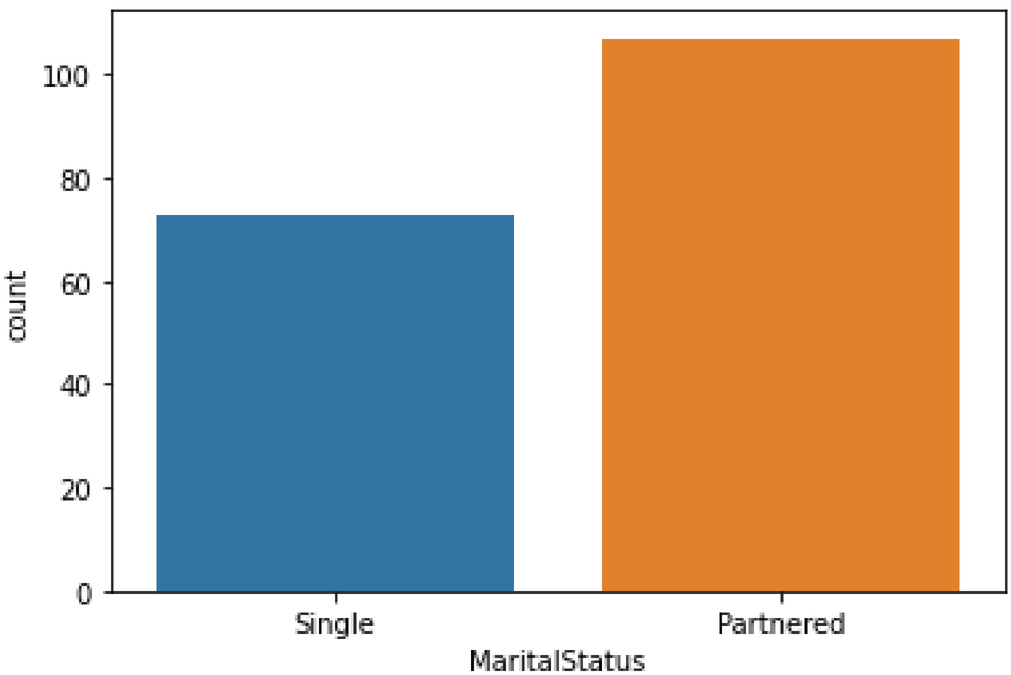
[35]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles AgeGroup Qualification Status MilesCat
0    112    Young    Graduation    Low   Medium
1     75    Young    Graduation    Low      Low
2     66    Young    Graduation    Low      Low
3     85    Young     Schooling    Low      Low
4     47    Young    Graduation    Low      Low
```

0.4.4 3. Check if features like marital status, age have any effect on the product purchased (using countplot, histplots, boxplots etc)

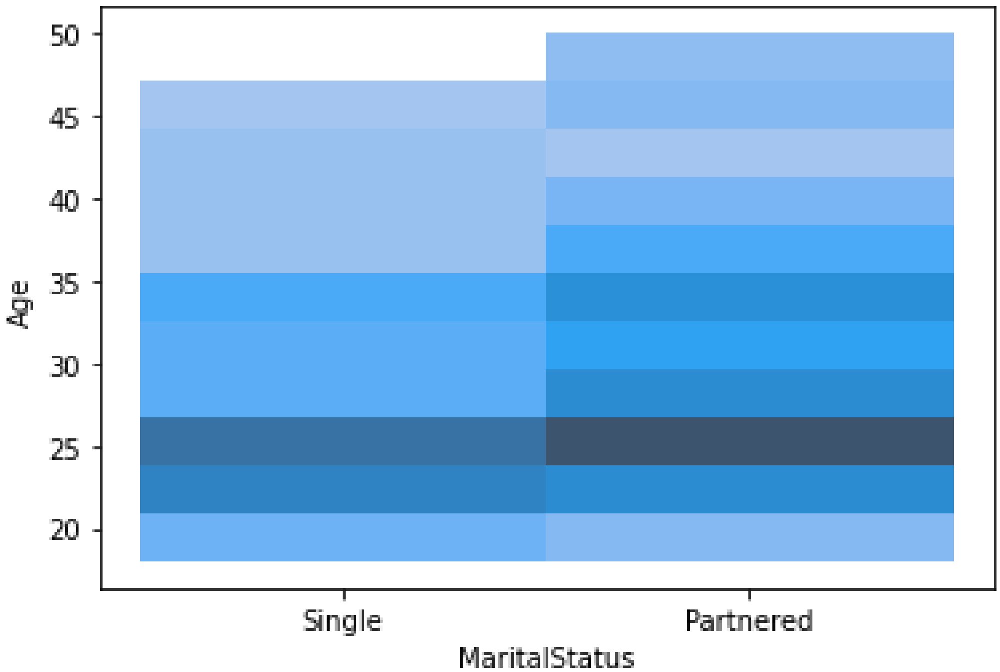
[36]:
```
sns.countplot(x  = aerofit_df['MaritalStatus']);

# most of the customers are partnered
```



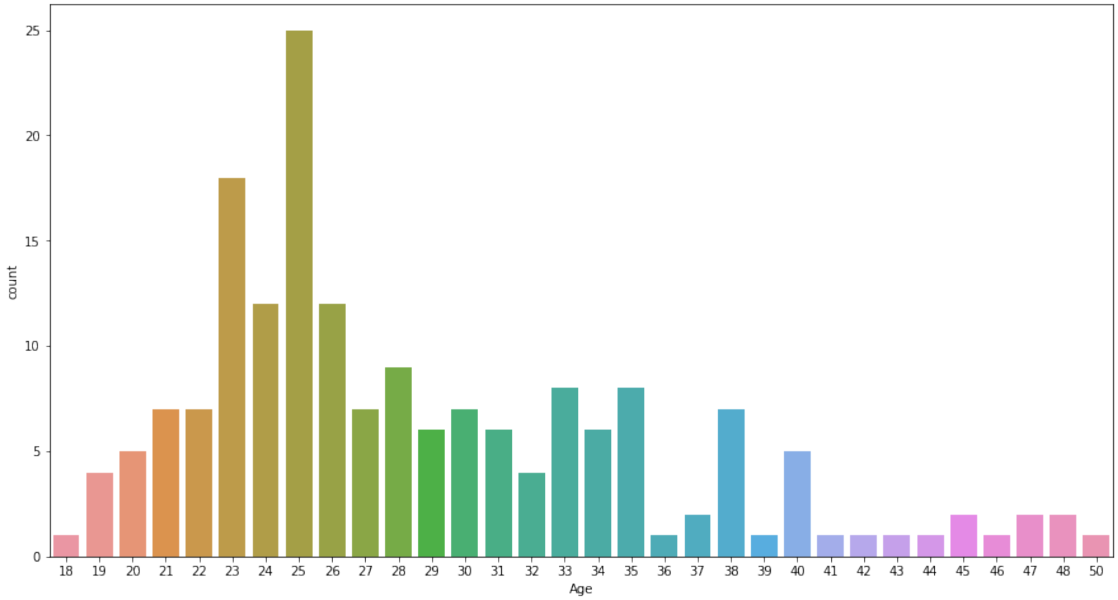
[37]:
```
sns.histplot(x  = aerofit_df['MaritalStatus'], y = aerofit_df['Age']);

# from the plot we can observe the most of the customers with age 25 to 27 are␣
↪partenered.
# maximum number of customers are seen in the range
#       21 to 26 single and 21 to 29 partenered.
```

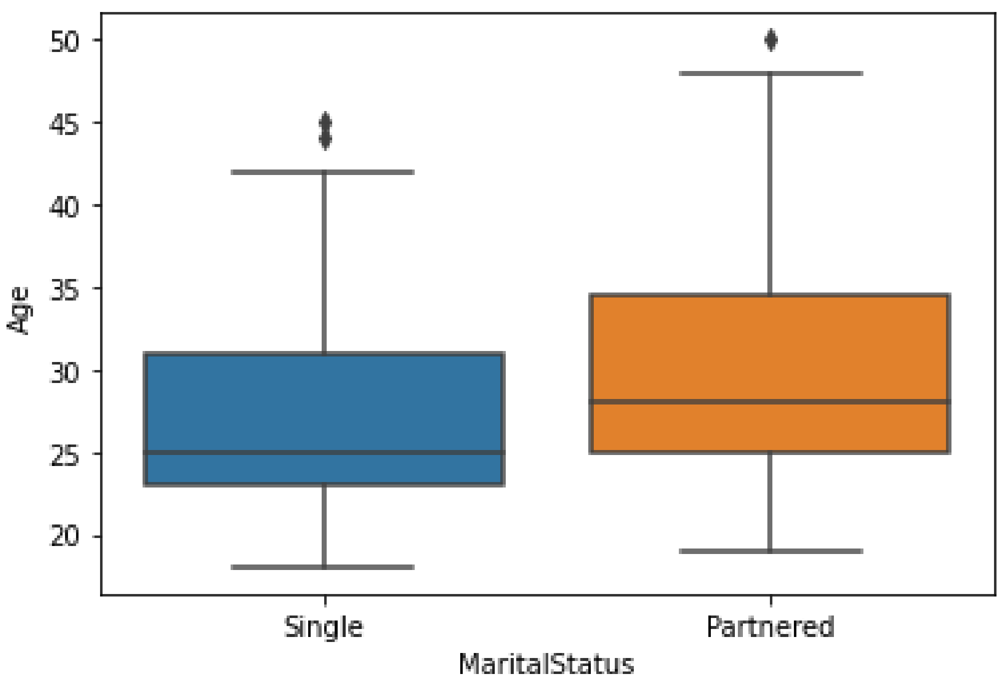


[38]:
```
fig, axes = plt.subplots(figsize = (15, 8))

sns.countplot(x  = aerofit_df['Age']);
# most of the customers in the range 23 to 27
```

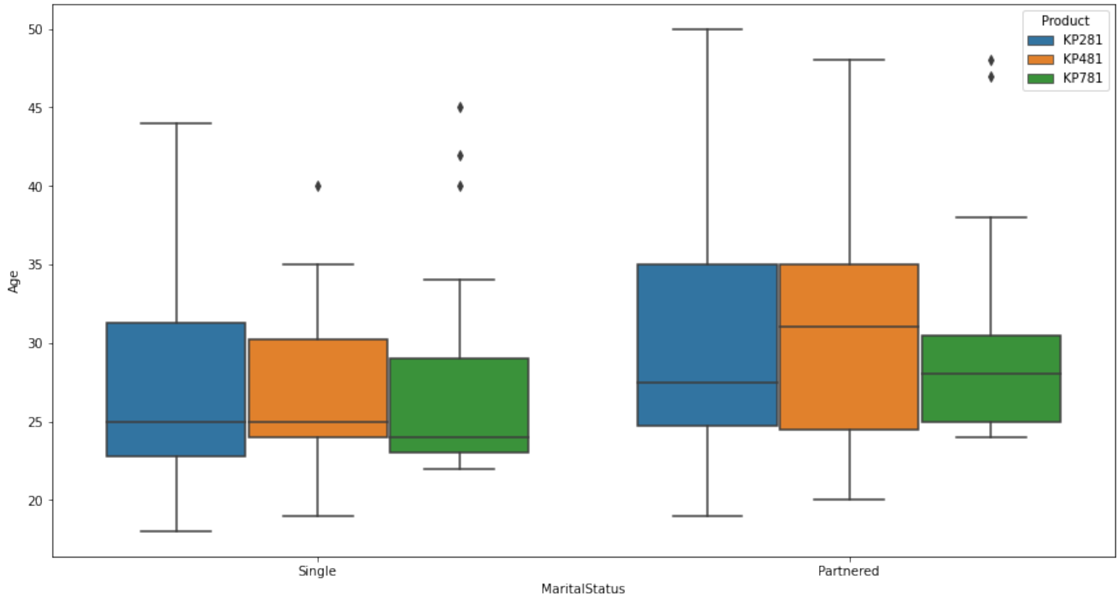


[39]:
```
sns.boxplot(data = aerofit_df, x = 'MaritalStatus', y = 'Age');
```

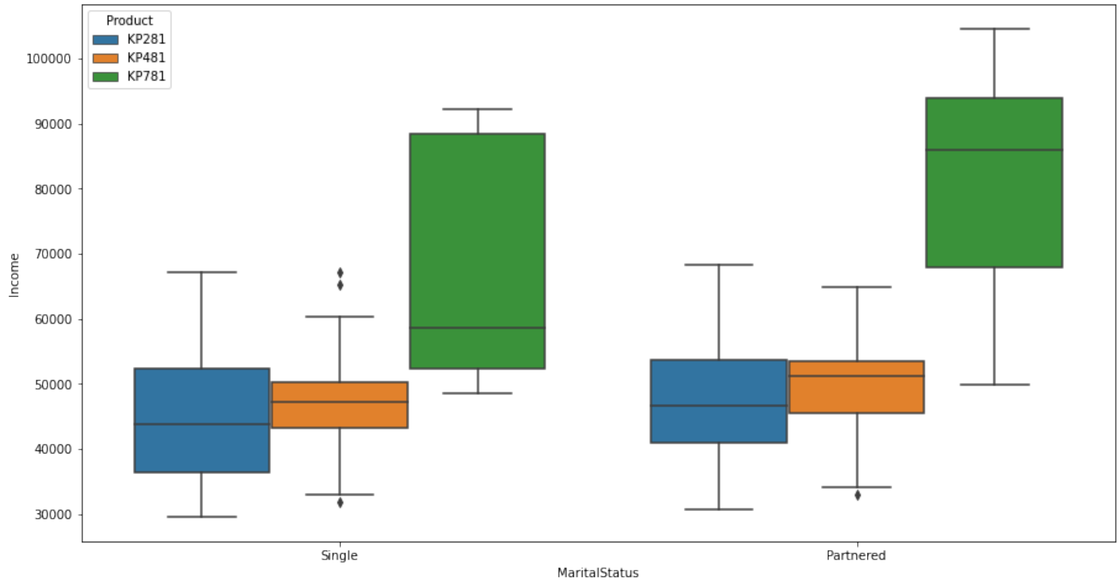


[40]:
```
fig, axes = plt.subplots(figsize = (15, 8))
sns.boxplot(data = aerofit_df, x = 'MaritalStatus', y = 'Age', hue = 'Product');
```

```
# from the plots we can see more single customers are buying KP281
# most parternered customers are buying KP281, KP481
# better to recommend KP281 first to the customers.
```



[41]:
```
fig, axes = plt.subplots(figsize = (15, 8))
sns.boxplot(data = aerofit_df, x = 'MaritalStatus', y = 'Income', hue =␣
↪'Product');
# high income are more tentive to buy KP781
```

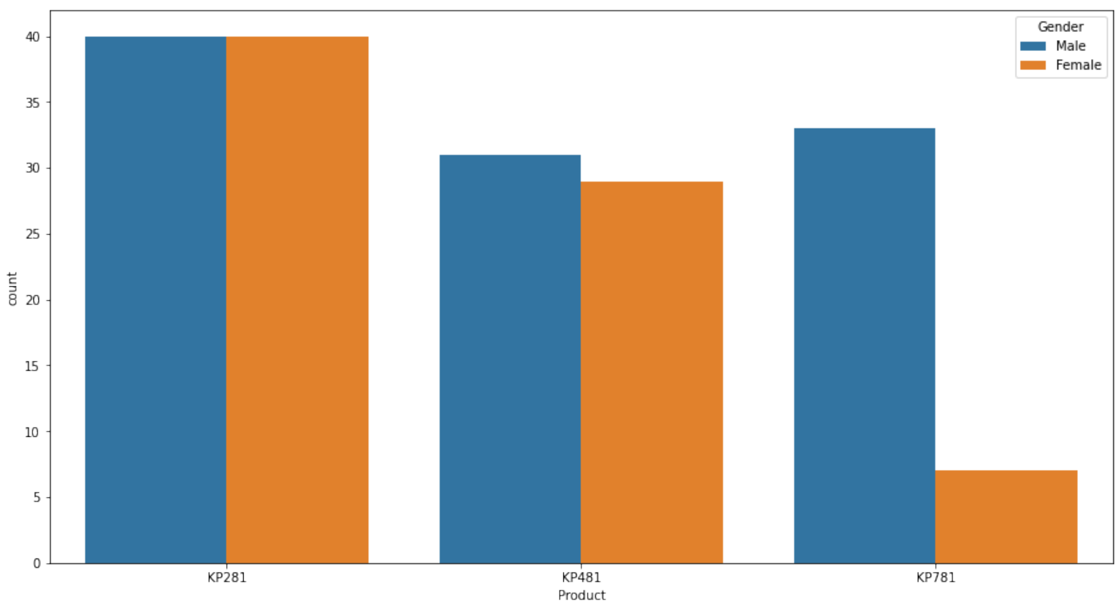


[42]:
```
gender_product_type_counts = pd.crosstab(aerofit_df['Product'],␣
    ↪aerofit_df['Gender'], margins = True)
gender_product_type_counts
```

[42]:

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

[43]:
```
fig, axes = plt.subplots(figsize = (15, 8))
sns.countplot(data = aerofit_df, x = 'Product', hue = 'Gender');
# males are more tentative to buy KP781 than Female
# Female are more tentative to buy KP481 than male
# male and female are equally likely to buy KP281
```



0.4.5 4. Representing the marginal probability like -

0.4.6 what percent of customers have purchased KP281, KP481, or KP781 in a table (can use pandas.crosstab here)

[44]:
```
products_counts = aerofit_df['Product'].value_counts().to_frame()
products_counts
```

[44]:

	Product
KP281	80
KP481	60
KP781	40

[45]:
```
# probability of customer buying KP281
KP281_prob = products_counts.loc['KP281'] / np.sum(products_counts['Product'])
KP281_prob
```

[45]:
```
Product    0.444444
Name: KP281, dtype: float64
```

[46]:
```
# probability of customer buying KP481
KP481_prob = products_counts.loc['KP481'] / np.sum(products_counts['Product'])
KP481_prob
```

[46]:
```
Product    0.333333
Name: KP481, dtype: float64
```

[47]:
```
# probability of customer buying KP781
KP781_prob = products_counts.loc['KP781'] / np.sum(products_counts['Product'])
KP781_prob
```

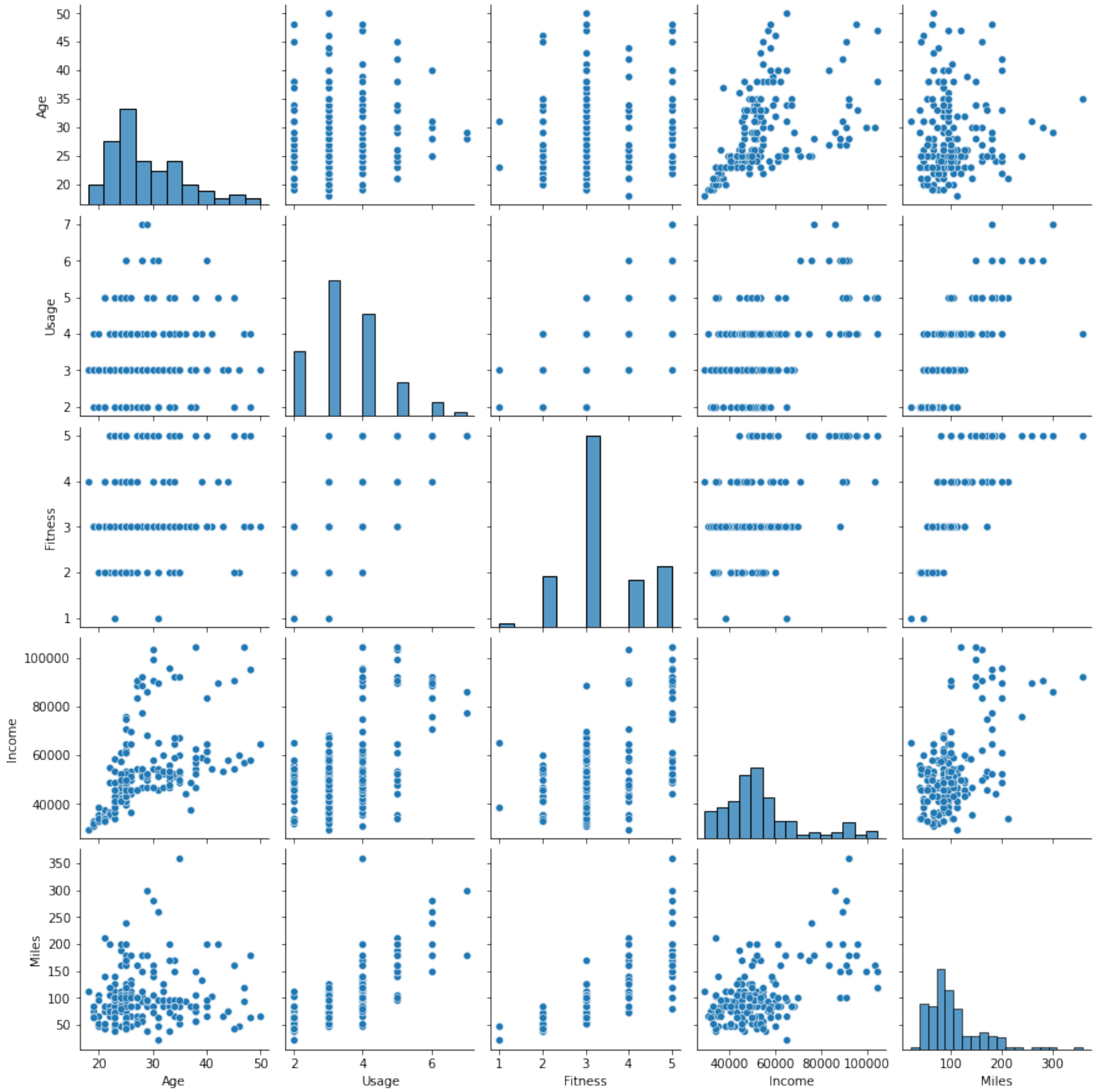
[47]:
```
Product    0.222222
Name: KP781, dtype: float64
```

[48]:
```
# probability of buying aerofit product by a customer are
# KP281_prob > KP481_prob > KP781_prob
```

0.4.7 5. Check correlation among different factors using heat maps or pair plots.

[49]:
```
sns.pairplot(aerofit_df[['Age', 'Usage', 'Fitness', 'Income', 'Miles']]);

# from the plots we can see more correlations between
# Age, Income
# Usage, Fitness
# less correlation between Income, Miles
```



0.4.8 6. With all the above steps you can answer questions like: What is the probability of a male customer buying a KP781 treadmill?

[50]:
```
pd.crosstab(aerofit_df['Product'], aerofit_df['Gender'], margins = True)
```

[50]:

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

[51]:
```
gender_product_type_counts = pd.crosstab(aerofit_df['Product'],␣
 ↪aerofit_df['Gender'])
gender_product_type_counts
```

[51]:
```
Gender   Female  Male
Product
KP281        40    40
KP481        29    31
KP781         7    33
```

[52]:
```
# What is the probability of a male customer buying a KP781 treadmill?
# given a male customer what is the prob of buying a KP781 treadmill?

count_male_KP781 =  gender_product_type_counts.loc["KP781"].loc['Male']
count_male_KP781
```

[52]: 33

[53]:
```
total_male_customers = np.sum(gender_product_type_counts["Male"])
total_male_customers
```

[53]: 104

[54]:
```
male_KP781_prob = count_male_KP781 / total_male_customers
male_KP781_prob

# there is 0.3 prob that if a customer is male that he will buy KP781 treadmill
```

[54]: 0.3173076923076923

0.4.9 7. Customer Profiling - Categorization of users.

[55]:
```
aerofit_df.head()
```

[55]:
```
  Product  Age  Gender  Education MaritalStatus  Usage  Fitness  Income  \
0   KP281   18    Male         14        Single      3        4   29562
1   KP281   19    Male         15        Single      2        3   31836
2   KP281   19  Female         14     Partnered      4        3   30699
3   KP281   19    Male         12        Single      3        3   32973
4   KP281   20    Male         13     Partnered      4        2   35247

   Miles AgeGroup Qualification Status MilesCat
0    112    Young    Graduation    Low   Medium
1     75    Young    Graduation    Low      Low
2     66    Young    Graduation    Low      Low
3     85    Young     Schooling    Low      Low
4     47    Young    Graduation    Low      Low
```

[56]:
```python
# We have already classified customers based on age, education, income, miles
```

0.4.10 8. Probability- marginal, conditional probability.

[57]:
```python
# marginal probability of customer usage is 3.

len(aerofit_df.loc[aerofit_df['Usage'] == 3]) / len(aerofit_df)
```

[57]: 0.38333333333333336

[58]:
```python
np.mean(aerofit_df['Usage'])
```

[58]: 3.4555555555555557

[59]:
```python
np.mean(aerofit_df['Fitness'])
```

[59]: 3.311111111111111

[60]:
```python
# conditional probability- probability of customer buying KP281 given Status as␣
↪low

status_product_type_counts = pd.crosstab(aerofit_df['Product'],␣
↪aerofit_df['Status'])
status_product_type_counts
```

[60]:

Status Product	High	Low	Medium
KP281	0	23	57
KP481	0	9	51
KP781	19	0	21

[61]:
```python
status_product_type_counts.loc['KP281']['Low'] / np.
↪sum(status_product_type_counts['Low'])
# low status people are having high prob of buying KP281
```

[61]: 0.71875

[62]:
```python
# conditional probability- probability of customer buying KP781 given Status as␣
↪high

status_product_type_counts.loc['KP781']['High'] / np.
↪sum(status_product_type_counts['High'])
# a high salaried man always tending to buy KP781, so better recommend them␣
↪KP781 at first
```

[62]: 1.0

[63]:
```
age_product_type_counts = pd.crosstab(aerofit_df['Product'],␣
 ↪aerofit_df['AgeGroup'])
age_product_type_counts
```

[63]:
```
AgeGroup  Middle Age  Old  Young
Product
KP281             32   14     34
KP481             24    8     28
KP781             17    6     17
```

[64]:
```
# Most of them based on age classification are willing to KP281
```

[65]:
```
fit_product_type_counts = pd.crosstab(aerofit_df['Product'],␣
 ↪aerofit_df['Fitness'])
fit_product_type_counts
```

[65]:
```
Fitness  1   2   3  4   5
Product
KP281    1  14  54  9   2
KP481    1  12  39  8   0
KP781    0   0   4  7  29
```

[66]:
```
# most fitness people willing to buy KP781
# less fitness people are willing to buy KP281, KP481# less fitness people are␣
 ↪willing to buy KP281, KP481
```

0.4.11 9. Some recommendations and actionable insights, based on the inferences.

- rich people and fitness people are more willing to buy KP781.
- most of the customers are medium status people, so our ads should target them.
- better to produce high KP281 products compared to KP481 which should be greater in number than KP781.
- Males customers are more likely to buy KP781, female customers are less likely to buy KP781
- Our targeted audiences can be between age 23 to 28.
- we can also target couples, so our product can be used by male, female and children, as most of the customers are parterned.

[ ]: