Business Case: Aerofit - Descriptive Statistics & Probability

In [1]:
```
#Importing the dataset
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

DF = pd.read_csv(r"H:\Scaler\Pandas\Aerofit Project\aerofit_treadmill.csv")
```

In [3]:
```
#Checking the imported dataset
DF.head()
```

Out[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

In [4]:
```
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
```

In [5]:
```
DF.describe()
```

Out[5]:

	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 [6]:
```
DF.shape
```

Out[6]:
```
(180, 9)
```

In [8]:
```
9 columns
tled 'Product','Gender'& 'MaritalStatus' is object whereas all the other coluns contain integer datat type
mean of education seems to be 16,mean of usage is 3 and mean of fitness is also 3 , mean income is 53720 and mean of miles is 103
```

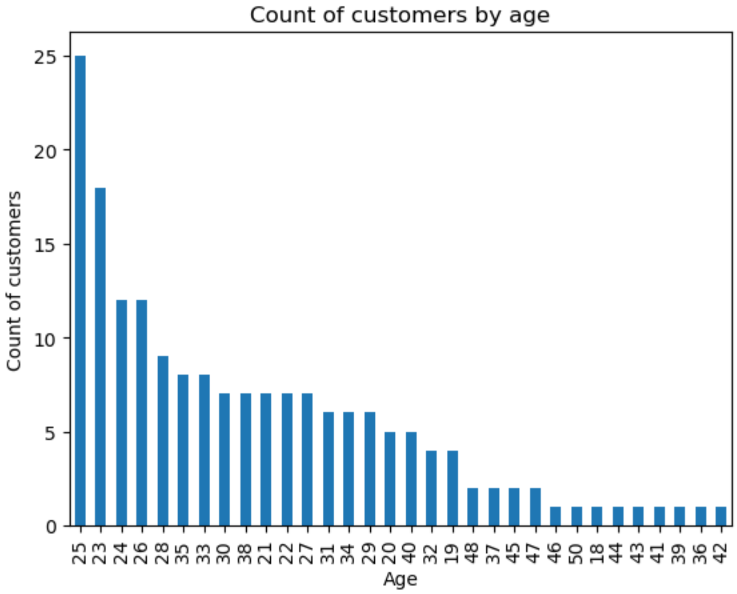
In [9]:
```
DF['Age'].nunique()
```

Out[9]:
```
32
```

In [10]:
```
DF['Age'].value_counts()
```

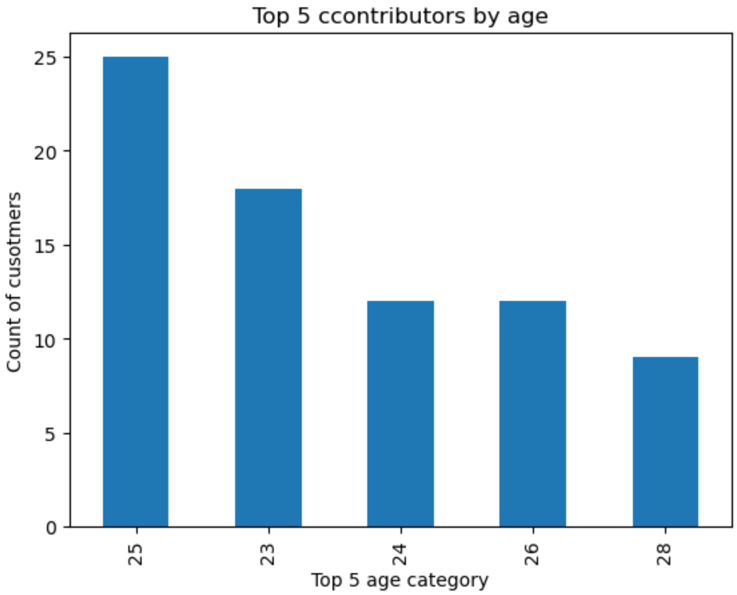
Out[10]:
```
25    25
23    18
24    12
26    12
28     9
35     8
33     8
30     7
38     7
21     7
22     7
27     7
31     6
34     6
29     6
20     5
40     5
32     4
19     4
48     2
37     2
45     2
47     2
46     1
50     1
18     1
44     1
43     1
41     1
39     1
36     1
42     1
Name: Age, dtype: int64
```

In [51]:
```
DF['Age'].value_counts().plot(kind='bar')
plt.xlabel('Age')
plt.ylabel('Count of customers')
plt.title('Count of customers by age')
plt.show()
```

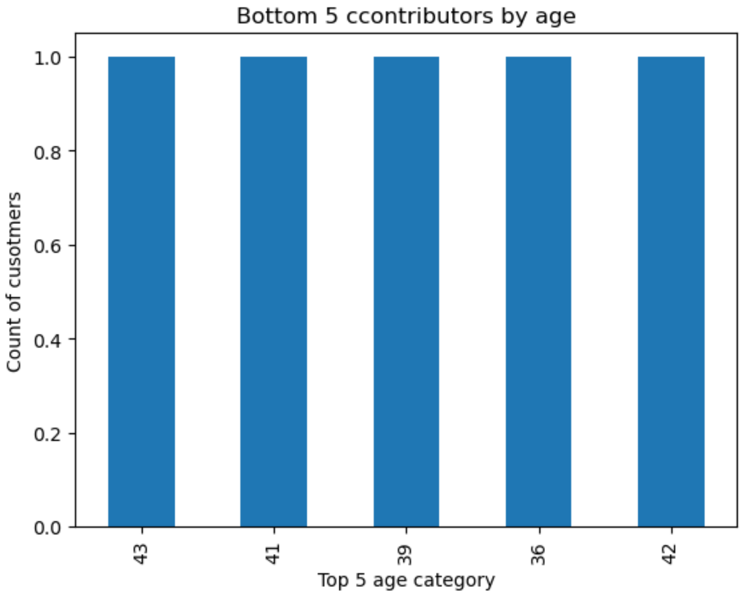


In [13]:
```
#The highest count of customers is of the age 25 followed by customers of age 23 and 24
```

In [52]:
```
DF['Age'].value_counts().head().plot(kind='bar')
plt.xlabel('Top 5 age category')
plt.ylabel('Count of cusotmers')
plt.title('Top 5 ccontributors by age')
plt.show()
```



In [53]:
```
DF['Age'].value_counts().tail().plot(kind='bar')
plt.xlabel('Top 5 age category')
plt.ylabel('Count of cusotmers')
plt.title('Bottom 5 ccontributors by age')
plt.show()
```



In [16]:
```
#The top 5 categories in age contributors are 25,23,24,26,28 and the bottom 5 contributors are of age 43,41,39,36 & 42 .
```

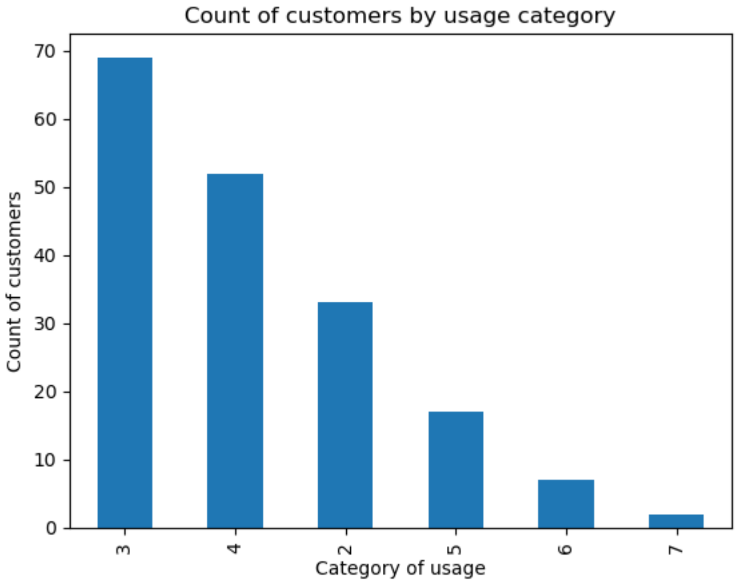
In [19]:
```
DF['Usage'].nunique()
```

Out[19]:
```
6
```

In [20]:
```
DF['Usage'].value_counts()
```

Out[20]:
```
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
```

In [50]:
```
DF['Usage'].value_counts().plot(kind='bar')
plt.xlabel('Category of usage')
plt.ylabel('Count of customers')
plt.title('Count of customers by usage category')
plt.show()
```



In [23]:
```
#The highest contributor to the usage category is 3 and the lowest contributor is 7
```

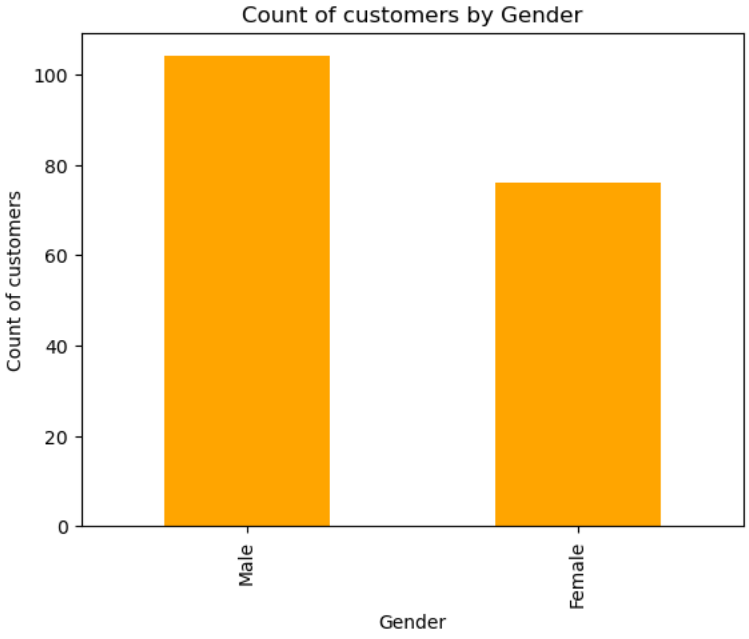
In [26]:
```
DF['Gender'].nunique()
```

Out[26]:
```
2
```

In [27]:
```
DF['Gender'].value_counts()
```

Out[27]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

In [49]:
```
DF['Gender'].value_counts().plot(kind='bar',color='orange')
plt.xlabel('Gender')
plt.ylabel('Count of customers')
plt.title('Count of customers by Gender')
plt.show()
```



In [30]:
```
DF['Gender'].value_counts(normalize=True)
```

Out[30]:
```
Male      0.577778
Female    0.422222
Name: Gender, dtype: float64
```

In [31]:
```
#The contribution of male customers is higher than the female customers with the male customers contributing to 58%
#whereas the female customers contribute to 42% of the population
```

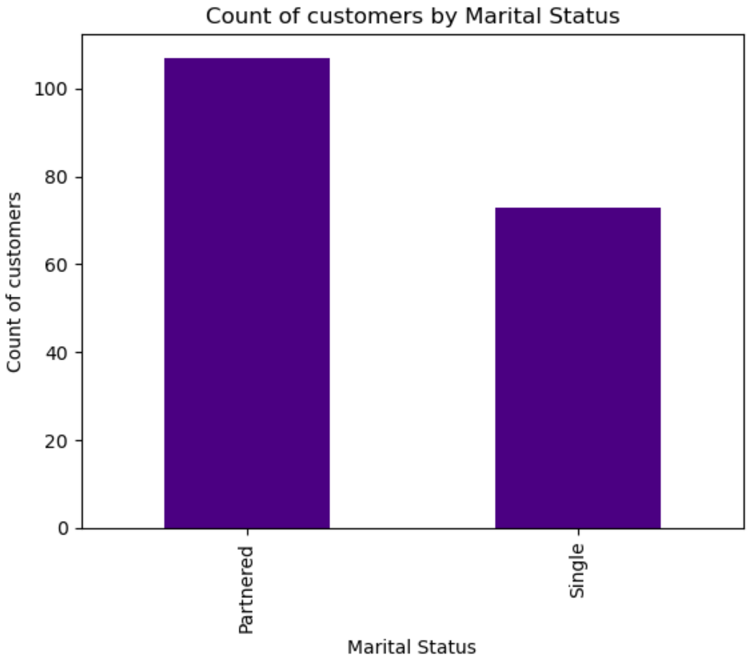
In [33]:
```
DF['MaritalStatus'].nunique()
```

Out[33]:
```
2
```

In [34]:
```
DF['MaritalStatus'].value_counts()
```

Out[34]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

In [48]:
```
DF['MaritalStatus'].value_counts().plot(kind='bar',color='indigo')
plt.xlabel('Marital Status')
plt.ylabel('Count of customers')
plt.title('Count of customers by Marital Status')
plt.show()
```



In [37]:
```
DF['MaritalStatus'].value_counts(normalize=True)
```

Out[37]:
```
Partnered    0.594444
Single       0.405556
Name: MaritalStatus, dtype: float64
```

In [38]:
```
#The count of partnered cusotmers is higher than the single customers with the ratio of partnered customers being 59%
#and single  customers being 40%
```

In [40]:
```
DF['Fitness'].nunique()
```

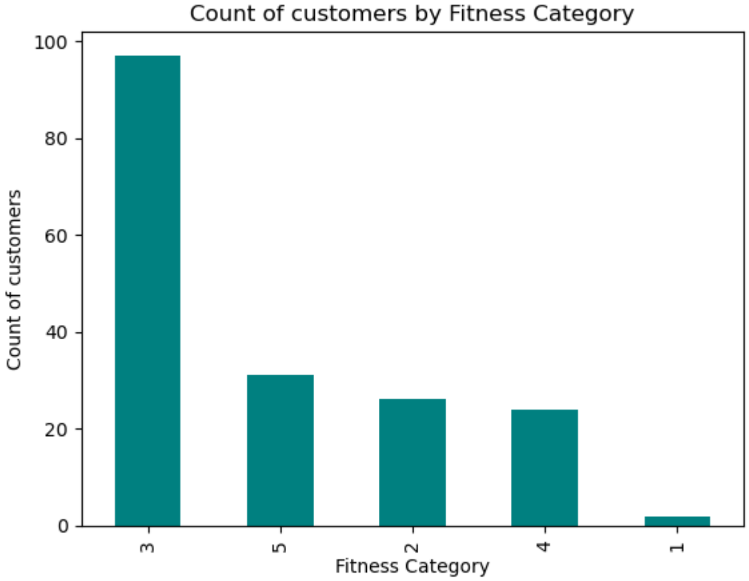
Out[40]:
```
5
```

In [41]:
```
DF['Fitness'].value_counts()
```

Out[41]:
```
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
```

In [47]:

```
DF['Fitness'].value_counts().plot(kind='bar',color='teal')
plt.xlabel('Fitness Category')
plt.ylabel('Count of customers')
plt.title('Count of customers by Fitness Category')
plt.show()
```



In [45]:

```
DF['Fitness'].value_counts(normalize=True)
```

Out[45]:

```
3    0.538889
5    0.172222
2    0.144444
4    0.133333
1    0.011111
Name: Fitness, dtype: float64
```

In [46]:

```
#Category 3 is the highest contributor in the fitness column with a total count of 97 customers followed by category 5
```

In [56]:

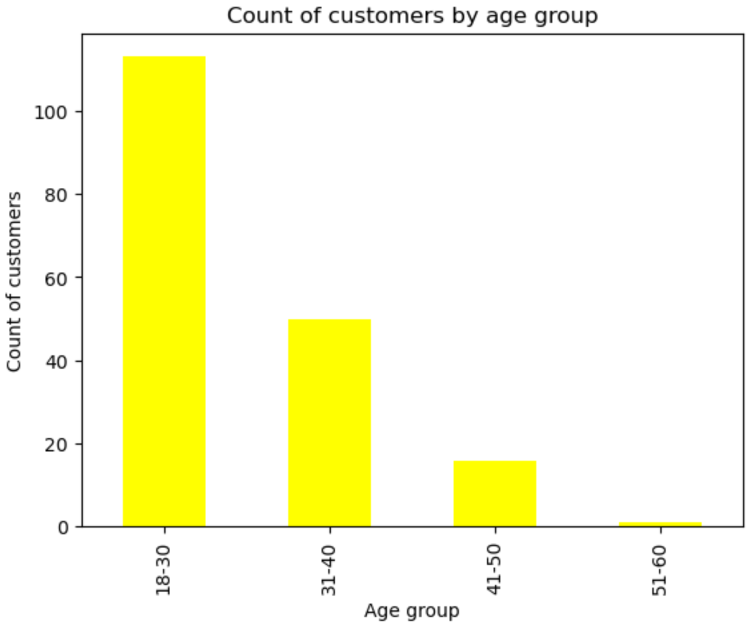
```
#Creating age categories by age range
DF1 = pd.read_csv(r"H:\Scaler\Pandas\Aerofit Project\aerofit_treadmill.csv")
bins = [17,30,40,50,61]
labels = ['18-30','31-40','41-50','51-60']
DF1['Age']=pd.cut(DF1['Age'],bins=bins,labels=labels,right=False)
DF1['Age'].value_counts()
```

Out[56]:

```
18-30    113
31-40     50
41-50     16
51-60      1
Name: Age, dtype: int64
```

In [57]:

```
DF1['Age'].value_counts().plot(kind='bar',color='yellow')
plt.xlabel('Age group')
plt.ylabel('Count of customers')
plt.title('Count of customers by age group')
plt.show()
```



In [58]:

```
DF1['Age'].value_counts(normalize=True)
```

Out[58]:

```
18-30    0.627778
31-40    0.277778
41-50    0.088889
51-60    0.005556
Name: Age, dtype: float64
```

In [59]:

```
#The highest count of customers is in the age range of 18-30 with a overall count of 113 making up 63% of  the overall customers
```

In [64]:

```
pd.crosstab(DF1['Age'],DF1['Product'],margins=True,margins_name='Total')
```

Out[64]:

Product	KP281	KP481	KP781	Total
Age				
18-30	53	33	27	113
31-40	20	22	8	50
41-50	6	5	5	16
51-60	1	0	0	1
Total	80	60	40	180

In [114]:

```
pd.crosstab(DF1['Age'],DF1['Product'],normalize=True,margins=True,margins_name='Total')
```

Out[114]:

Product	KP281	KP481	KP781	Total
Age				
18-30	0.294444	0.183333	0.150000	0.627778
31-40	0.111111	0.122222	0.044444	0.277778
41-50	0.033333	0.027778	0.027778	0.088889
51-60	0.005556	0.000000	0.000000	0.005556
Total	0.444444	0.333333	0.222222	1.000000

In [65]:
```
pd.crosstab(DF['Gender'],DF1['Product'],margins=True,margins_name='Total')
```

Out[65]:

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

In [115]:
```
pd.crosstab(DF['Gender'],DF1['Product'],normalize=True,margins=True,margins_name='Total')
```

Out[115]:

Product	KP281	KP481	KP781	Total
Gender				
Female	0.222222	0.161111	0.038889	0.422222
Male	0.222222	0.172222	0.183333	0.577778
Total	0.444444	0.333333	0.222222	1.000000

In [66]:
```
pd.crosstab(DF['Gender'],DF1['MaritalStatus'],margins=True,margins_name='Total')
```

Out[66]:

MaritalStatus	Partnered	Single	Total
Gender			
Female	46	30	76
Male	61	43	104
Total	107	73	180

In [118]:
```
pd.crosstab(DF['Gender'],DF1['MaritalStatus'],normalize=True,margins=True,margins_name='Total')
```

Out[118]:

MaritalStatus	Partnered	Single	Total
Gender			
Female	0.255556	0.166667	0.422222
Male	0.338889	0.238889	0.577778
Total	0.594444	0.405556	1.000000

In [69]:
```
pd.crosstab(DF['Product'],DF1['MaritalStatus'],margins=True,margins_name='Total')
```

Out[69]:

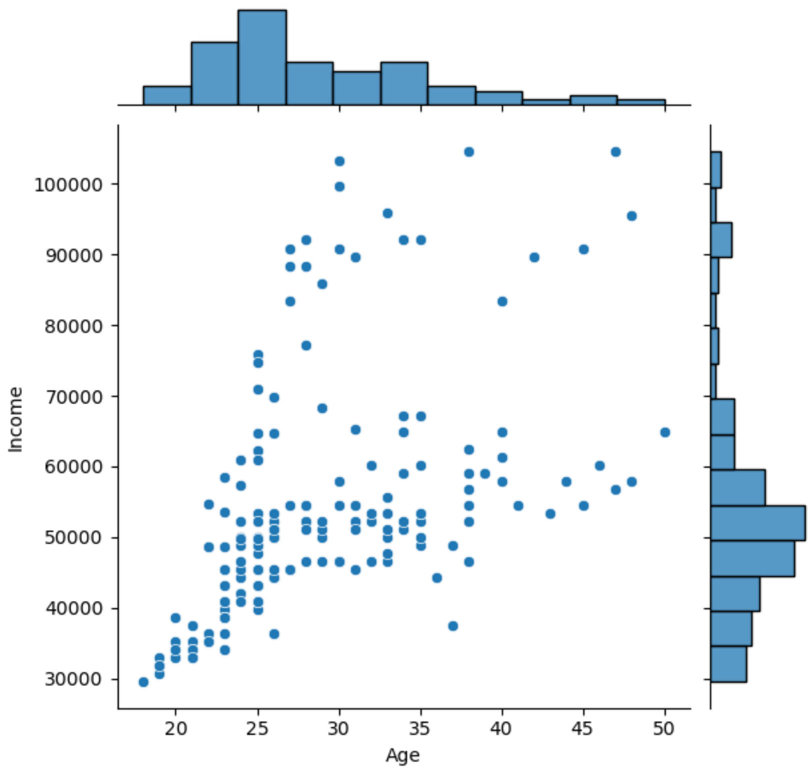
MaritalStatus	Partnered	Single	Total
Product			
KP281	48	32	80
KP481	36	24	60
KP781	23	17	40
Total	107	73	180

In [116]:
```
pd.crosstab(DF['Product'],DF1['MaritalStatus'],normalize=True,margins=True,margins_name='Total')
```

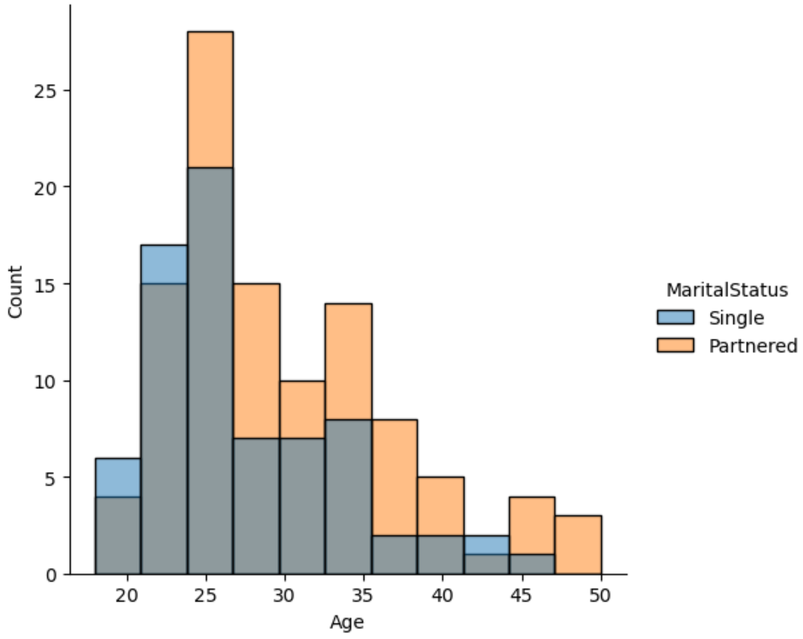
Out[116]:

MaritalStatus	Partnered	Single	Total
Product			
KP281	0.266667	0.177778	0.444444
KP481	0.200000	0.133333	0.333333
KP781	0.127778	0.094444	0.222222
Total	0.594444	0.405556	1.000000

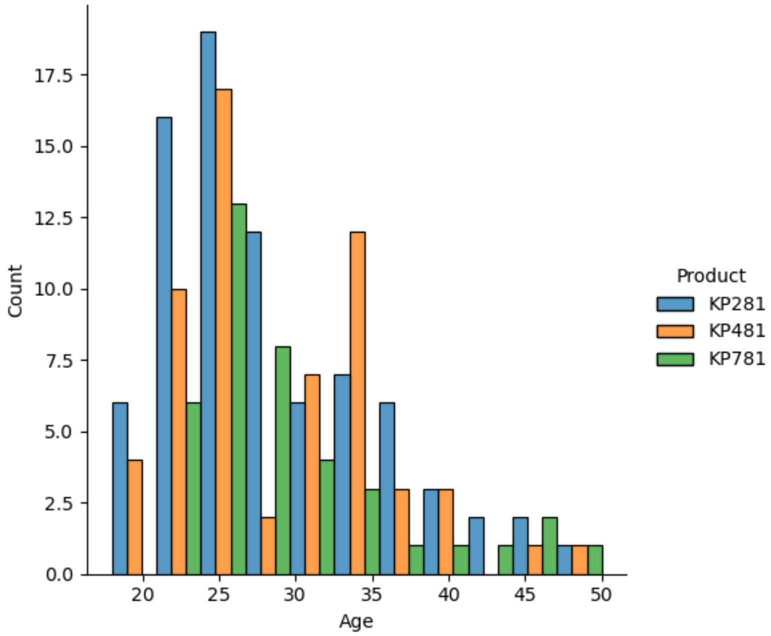
In [70]:
```
sns.jointplot(x='Age',y='Income',data=DF)
plt.show()
```



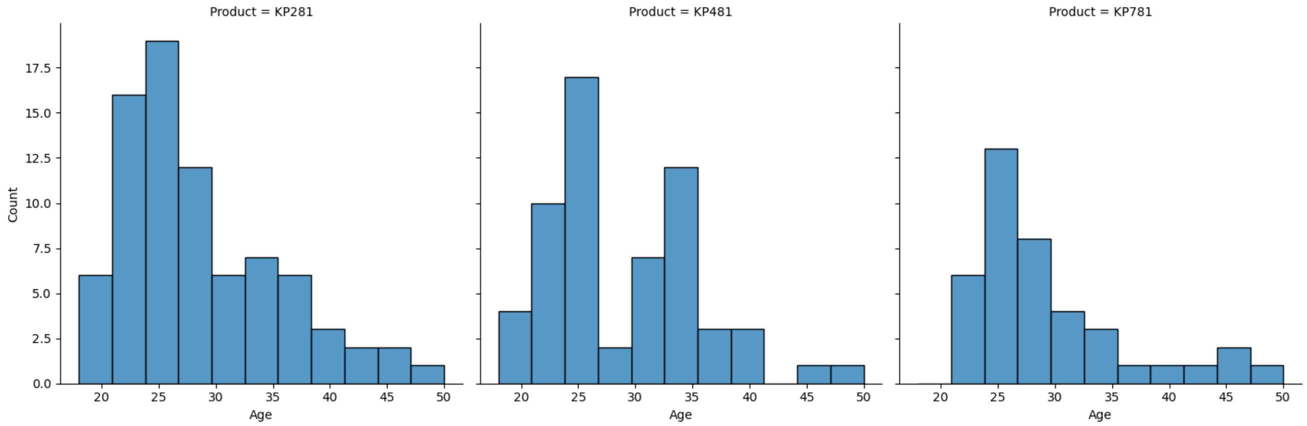
In [71]:
```
sns.displot(DF,x='Age',hue='MaritalStatus')
plt.show()
```



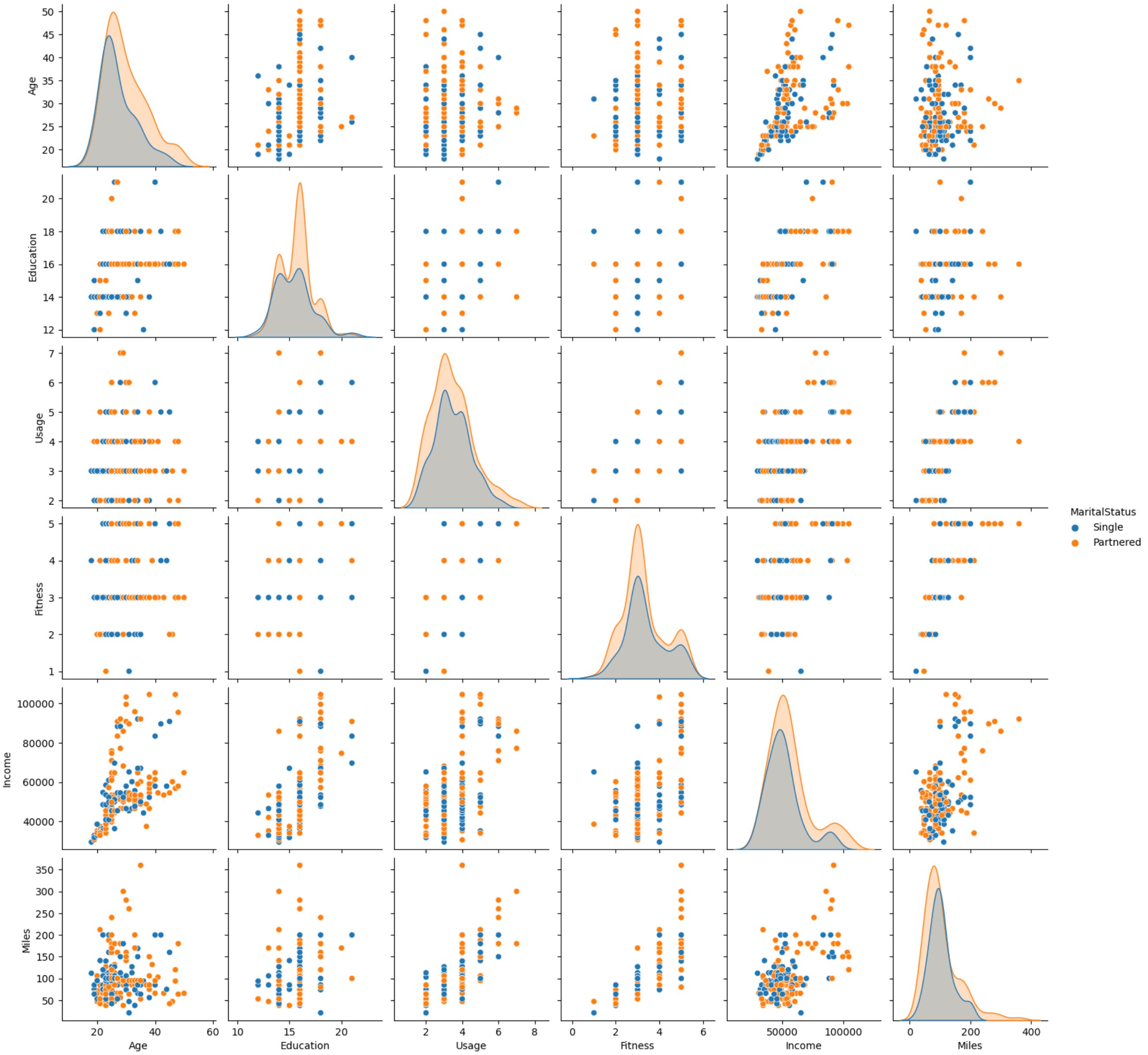
In [72]:
```
sns.displot(DF,x='Age',hue='Product',multiple='dodge')
plt.show()
```



In [73]:
```
sns.displot(DF, x="Age", col="Product")
plt.show()
```



In [74]:
```
sns.pairplot(DF,hue='MaritalStatus')
plt.show()
```



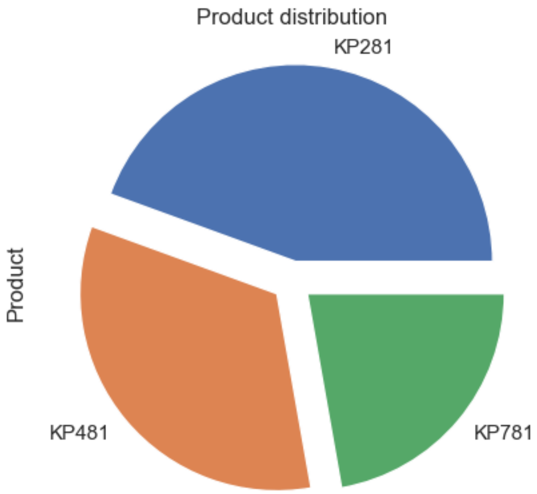
In [105]:
```
DF['Product'].value_counts()
```

Out[105]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

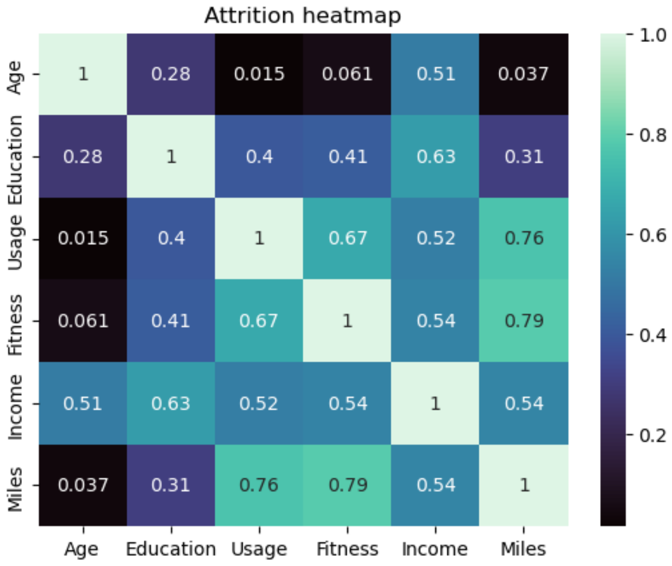
In [106]:
```
DF['Product'].value_counts(normalize=True)
```

Out[106]:
```
KP281    0.444444
KP481    0.333333
KP781    0.222222
Name: Product, dtype: float64
```

```
In [123]: DF['Product'].value_counts(normalize=True).plot(kind='pie',explode=[0.1,0.1,0.1])
          plt.title("Product distribution")
          plt.show()
```



```
In [76]: ax = sns.heatmap(DF.corr(),annot=True,cmap='mako')
         plt.title('Attrition heatmap')
         plt.show()
```



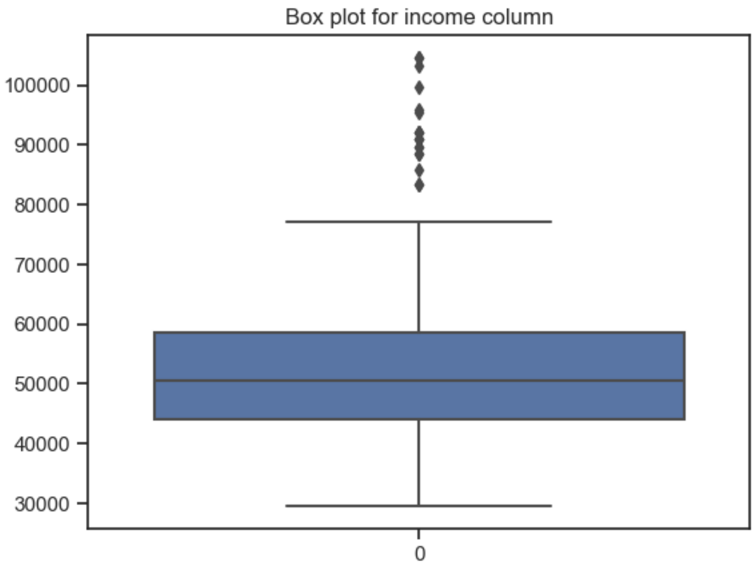
```
In [77]: def find_outliers_IQR(df):
             q1 = df.quantile(0.25)
             q3 = df.quantile(0.75)
             IQR = q3-q1
             outliers = df[((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]
             return outliers
```

```
In [78]: Income_outliers = find_outliers_IQR(DF['Income'])

         print('number of outliers: '+ str(len(Income_outliers)))

         print('max outlier value: '+ str(Income_outliers.max()))

         print('min outlier values: '+ str(Income_outliers.min()))
```

```
number of outliers: 19
max outlier value: 104581
min outlier values: 83416
```

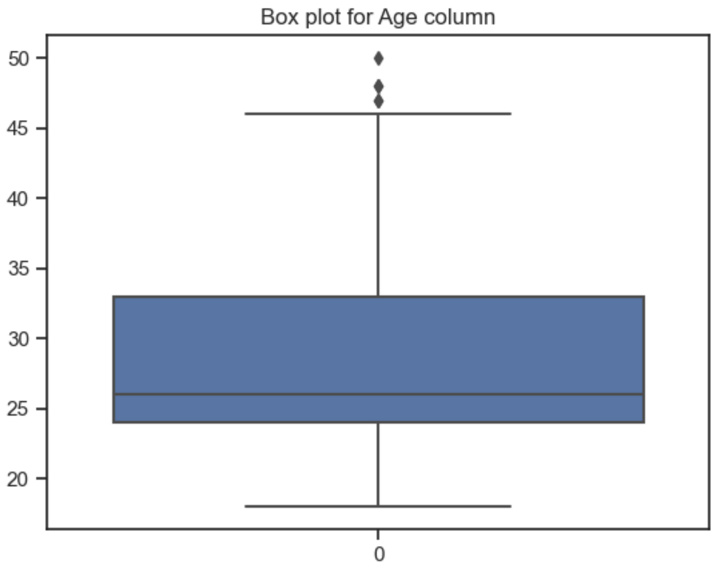
In [124]:

```
sns.boxplot(data=DF['Income'])
plt.title("Box plot for income column")
plt.show()
```



In [125]:

```
sns.boxplot(data=DF['Age'])
plt.title("Box plot for Age column")
plt.show()
```



In [81]:

```
Age_outliers = find_outliers_IQR(DF['Age'])

print('number of outliers: '+ str(len(Age_outliers)))

print('max outlier value: '+ str(Age_outliers.max()))

print('min outlier values: '+ str(Age_outliers.min()))
```

```
number of outliers: 5
max outlier value: 50
min outlier values: 47
```

In [109]:

```
DF['Age'].mean()
```

Out[109]:

```
28.788888888888888
```

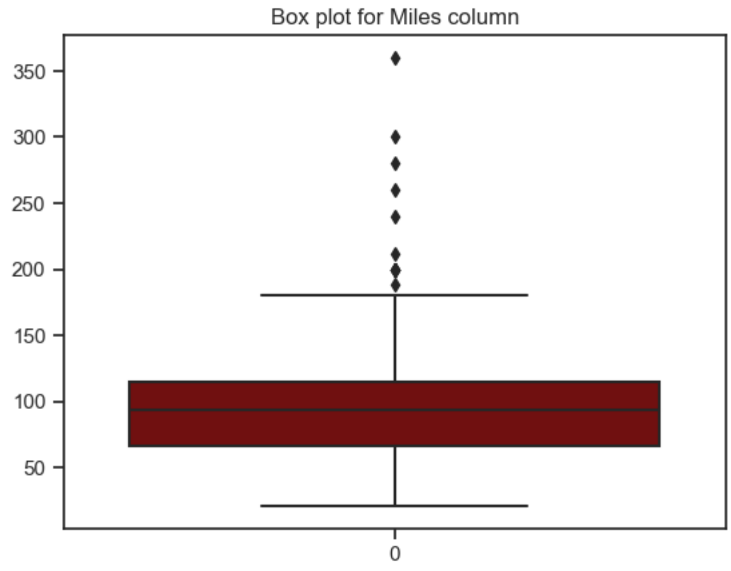
In [110]:

```
DF['Age'].mode()
```

Out[110]:

```
0    25
Name: Age, dtype: int64
```

In [126]:
```
sns.boxplot(data=DF['Miles'],color='maroon')
plt.title("Box plot for Miles column")
plt.show()
```



In [83]:
```
Miles_outliers = find_outliers_IQR(DF['Miles'])

print('number of outliers: '+ str(len(Miles_outliers)))

print('max outlier value: '+ str(Miles_outliers.max()))

print('min outlier values: '+ str(Miles_outliers.min()))
```

```
number of outliers: 13
max outlier value: 360
min outlier values: 188
```

In [111]:
```
DF['Miles'].mean()
```

Out[111]:
```
103.19444444444444
```

In [112]:
```
DF['Miles'].mode()
```

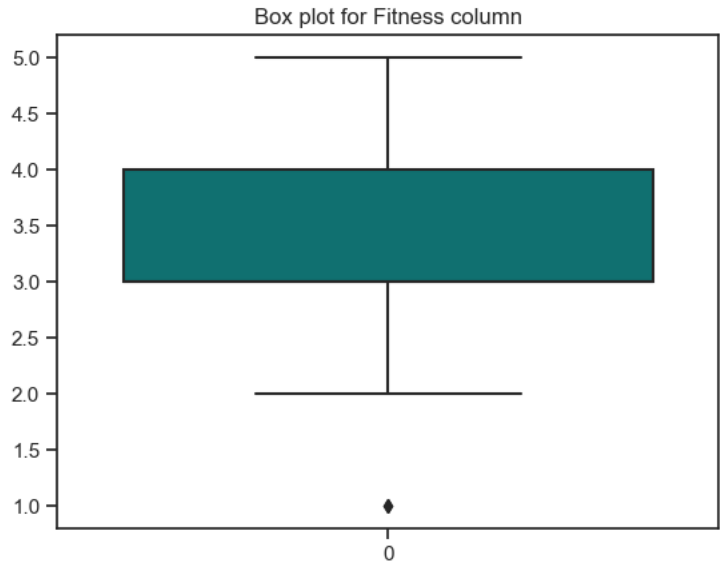
Out[112]:
```
0    85
Name: Miles, dtype: int64
```

In [113]:
```
DF['Miles'].median()
```

Out[113]:
```
94.0
```

In [128]:

```
sns.boxplot(data=DF['Fitness'],color='teal')
plt.title("Box plot for Fitness column")
plt.show()
```



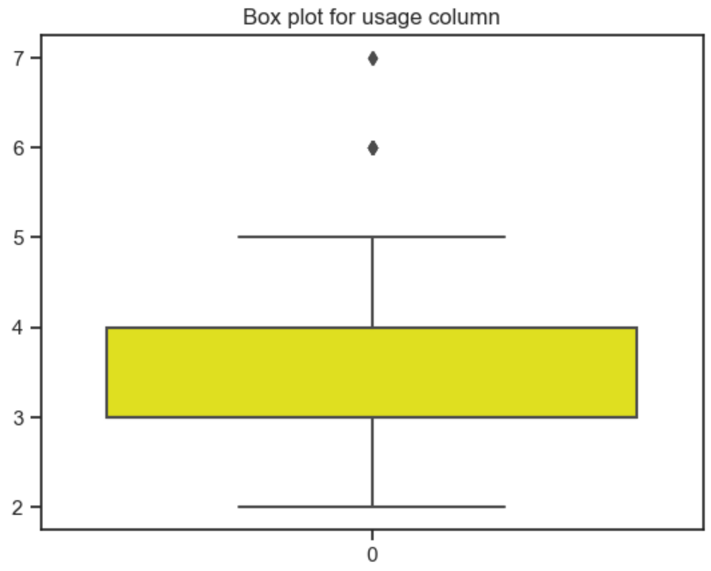
In [85]:

```
Fitness_outliers = find_outliers_IQR(DF['Fitness'])

print('number of outliers: '+ str(len(Fitness_outliers)))

print('max outlier value: '+ str(Fitness_outliers.max()))

print('min outlier values: '+ str(Fitness_outliers.min()))
```

```
number of outliers: 2
max outlier value: 1
min outlier values: 1
```

In [130]:

```
sns.boxplot(data=DF['Usage'],color='yellow')
plt.title("Box plot for usage column")
plt.show()
```



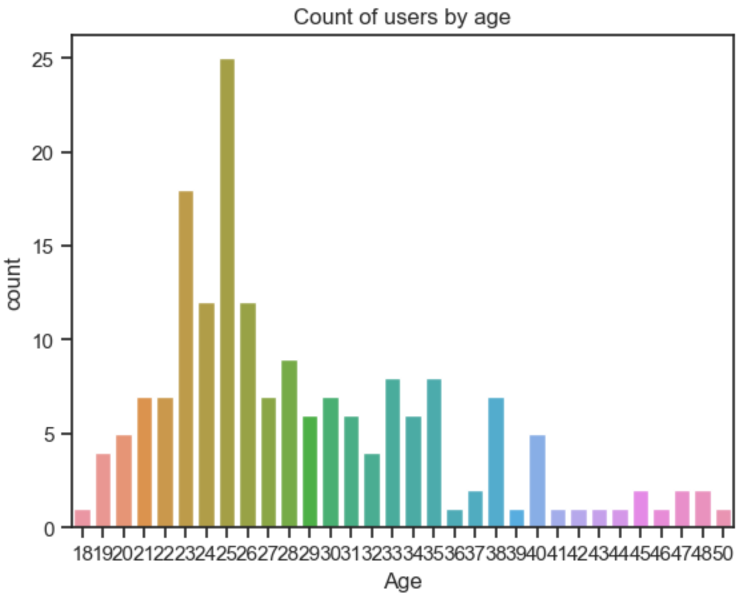
In [87]:

```
Usage_outliers = find_outliers_IQR(DF['Usage'])

print('number of outliers: '+ str(len(Usage_outliers)))

print('max outlier value: '+ str(Usage_outliers.max()))

print('min outlier values: '+ str(Usage_outliers.min()))
```

```
number of outliers: 9
max outlier value: 7
min outlier values: 6
```

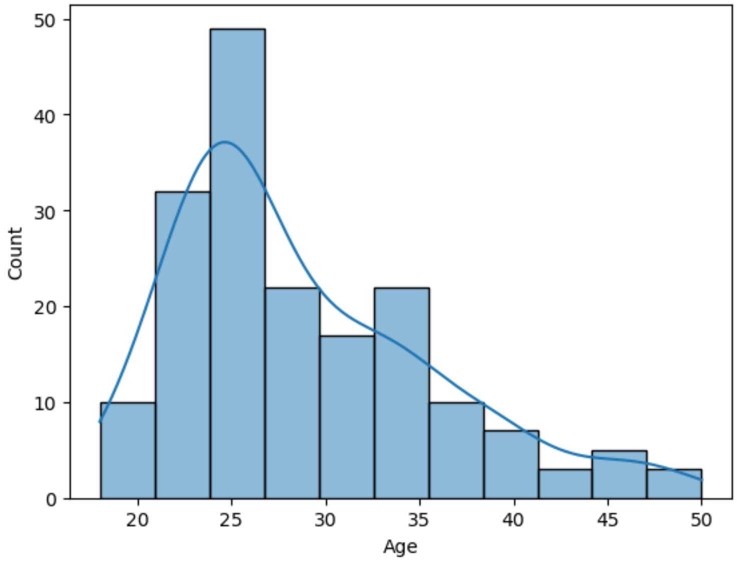
In [121]:

```
sns.countplot(x=DF['Age'])
plt.title ('Count of users by age')
plt.show()
```

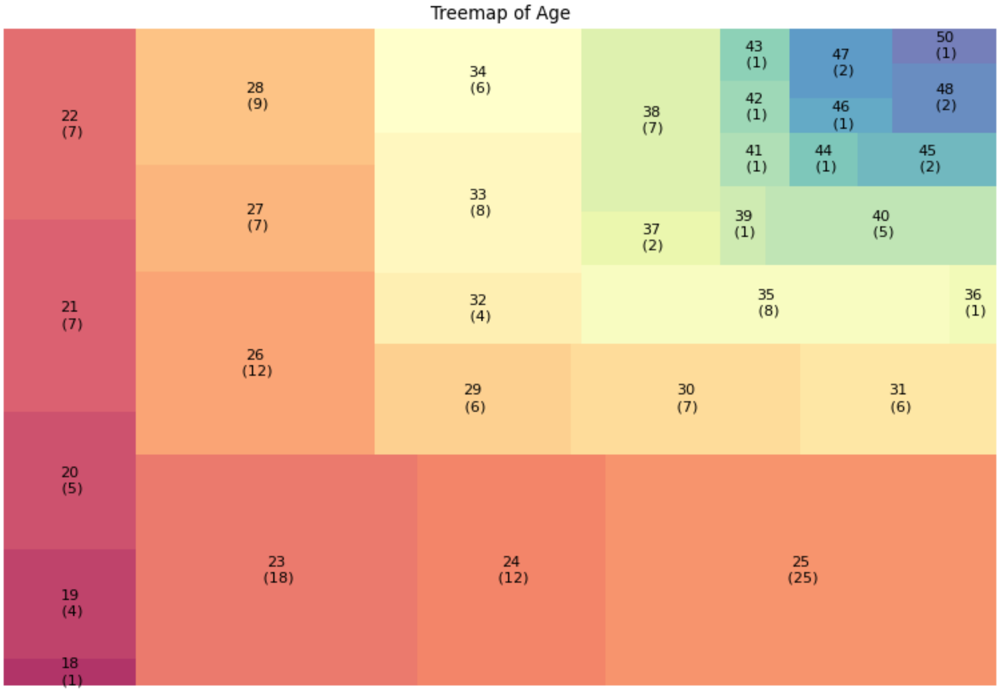


In [89]:

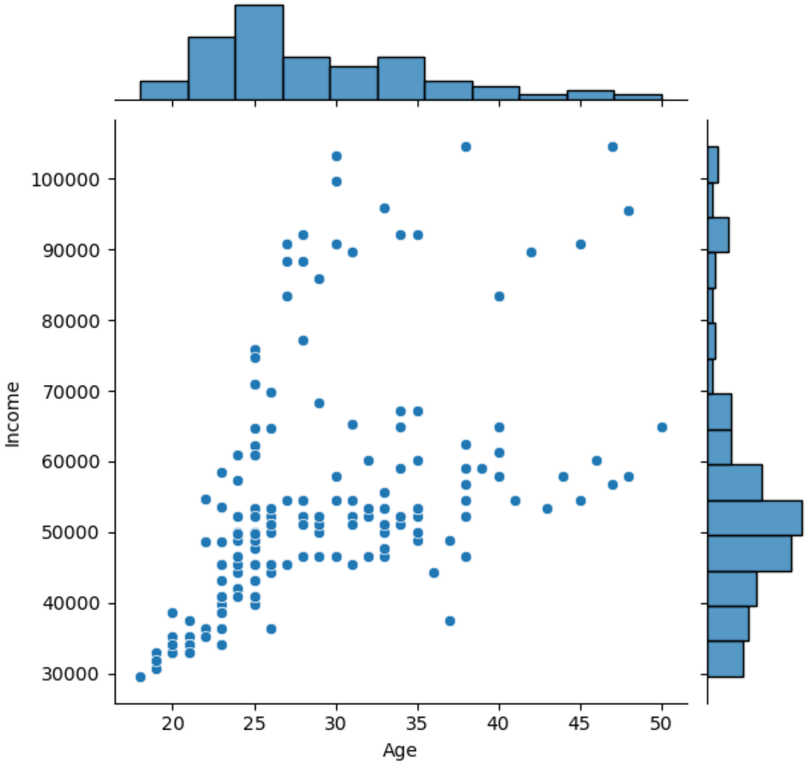
```
sns.histplot(DF['Age'],kde=True)
plt.show()
```



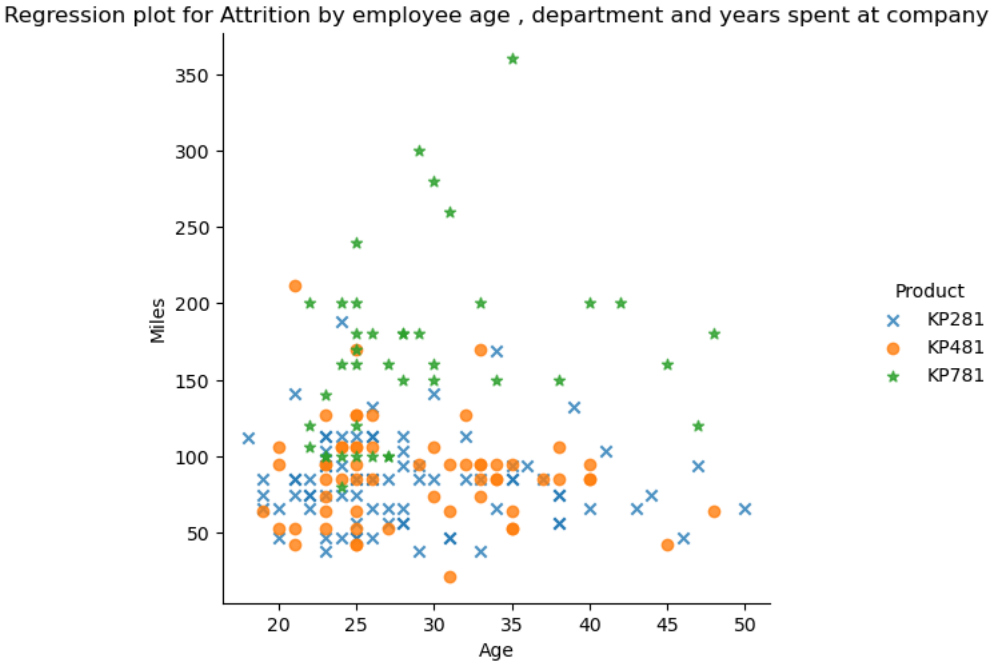
In [90]:
```
import squarify
df = DF.groupby('Age').size().reset_index(name='Agecounts')
labels = df.apply(lambda x: str(x[0]) + "\n (" + str(x[1]) + ")", axis=1)
sizes = df['Agecounts'].values.tolist()
colors = [plt.cm.Spectral(i/float(len(labels))) for i in range(len(labels))]

plt.figure(figsize=(12,8), dpi= 80)
squarify.plot(sizes=sizes, label=labels, color=colors, alpha=.8)
plt.title('Treemap of Age')
plt.axis('off')
plt.show()
```



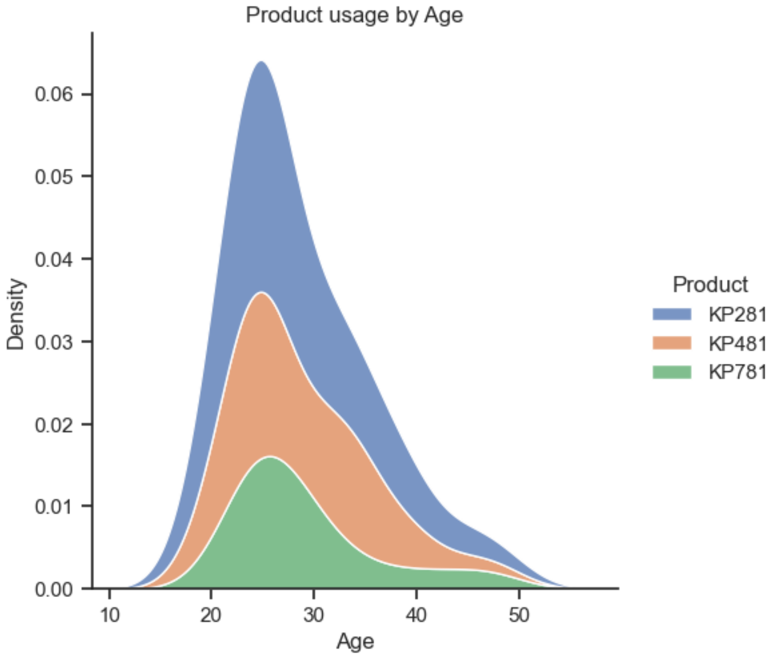
In [91]:
```
sns.jointplot(x='Age',y='Income',data=DF)
plt.show()
```



```
In [92]: sns.lmplot(x='Age',y='Miles',hue='Product',data=DF,markers=['x','o','*'],fit_reg=False)
         plt.title('Regression plot for Attrition by employee age , department and years spent at company')
         plt.grid(False)
         plt.show()
```

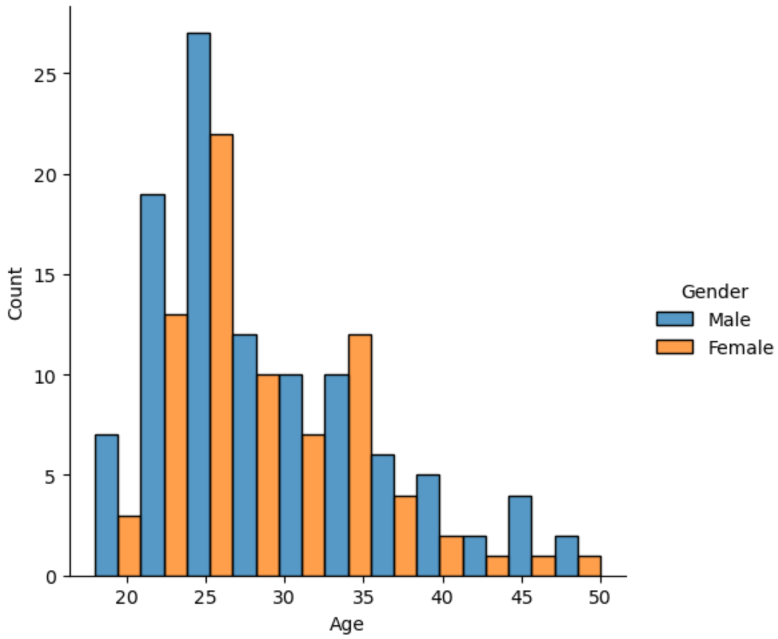


```
In [119]: sns.displot(DF, x="Age", hue="Product", kind="kde", multiple="stack")
          plt.title('Product usage by Age')
          plt.show()
```

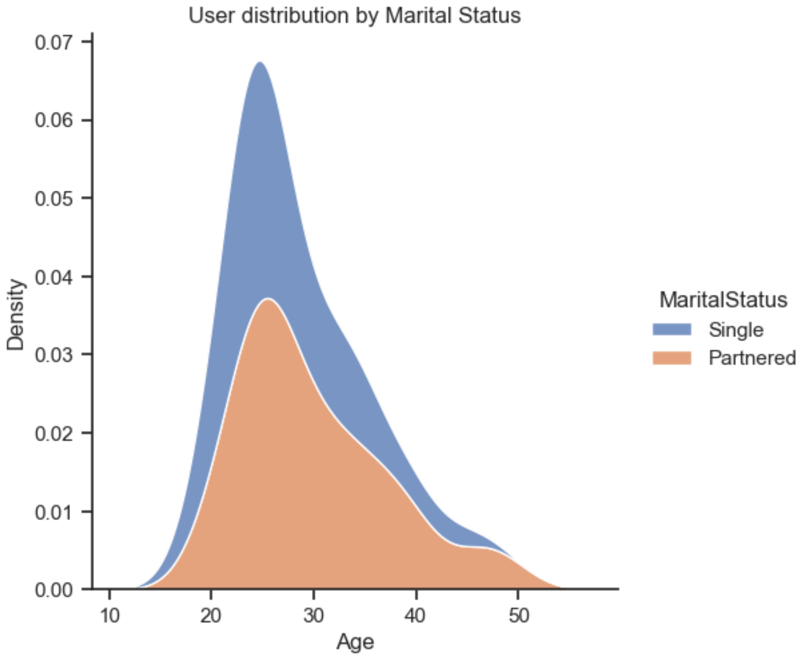


In [94]:
```
sns.displot(DF, x="Age", hue="Gender",multiple="dodge")
```

Out[94]:
```
<seaborn.axisgrid.FacetGrid at 0x1c97bf3f0a0>
```

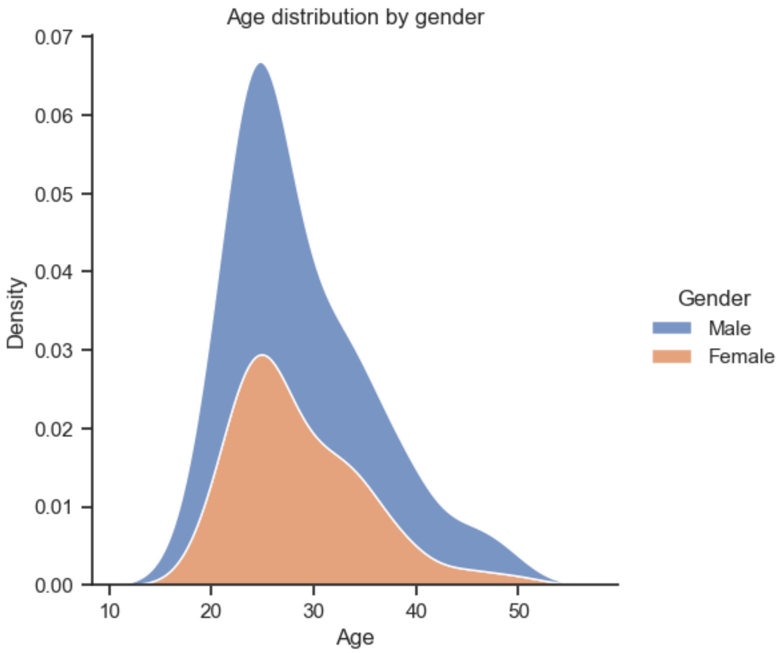


In [122]:
```
sns.displot(DF, x="Age", hue="MaritalStatus", kind="kde", multiple="stack")
plt.title("User distribution by Marital Status")
plt.show()
```


User distribution by Marital Status

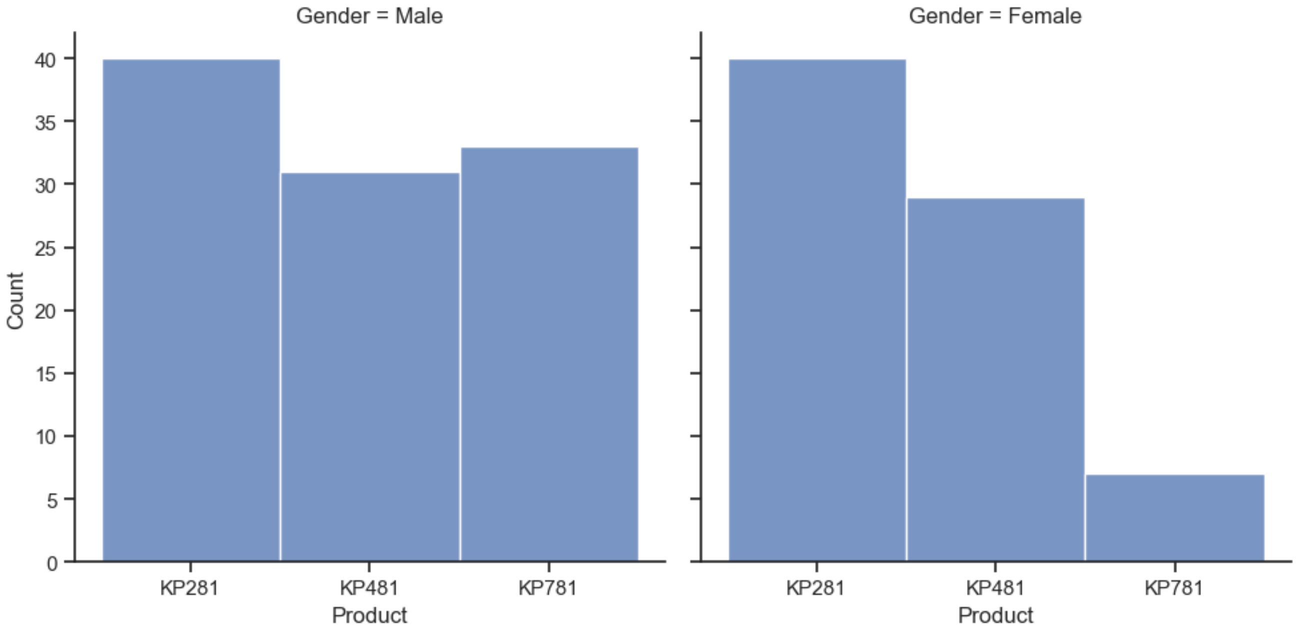
In [120]:

```
sns.displot(DF, x="Age", hue="Gender", kind="kde", multiple="stack")
plt.title("Age distribution by gender")
plt.show()
```

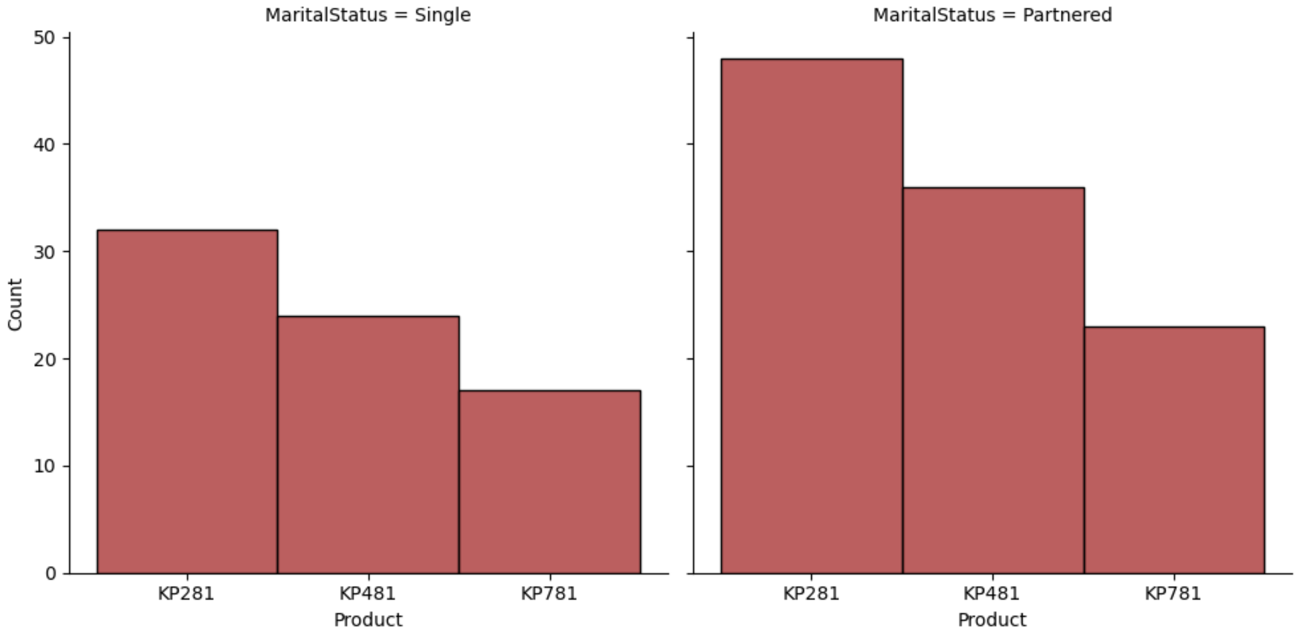


In [132]:

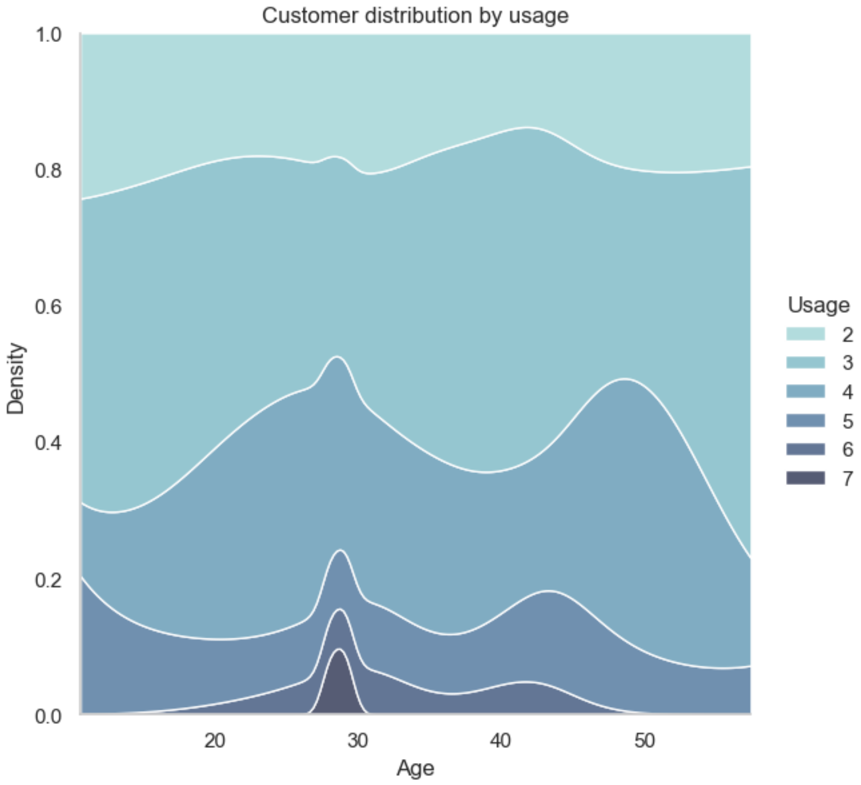
```
sns.displot(DF, x="Product", col="Gender")
#plt.title('Product purchased by Gender')
plt.show()
```



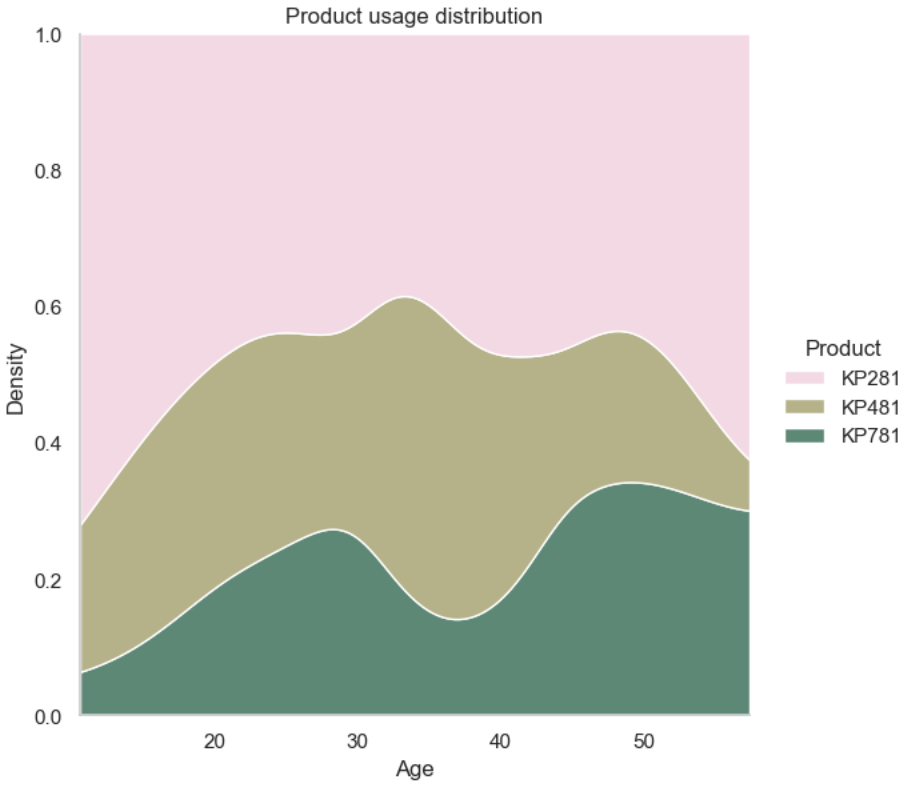
In [98]:
```
sns.displot(DF, x="Product", col="MaritalStatus",color='brown')
plt.show()
```



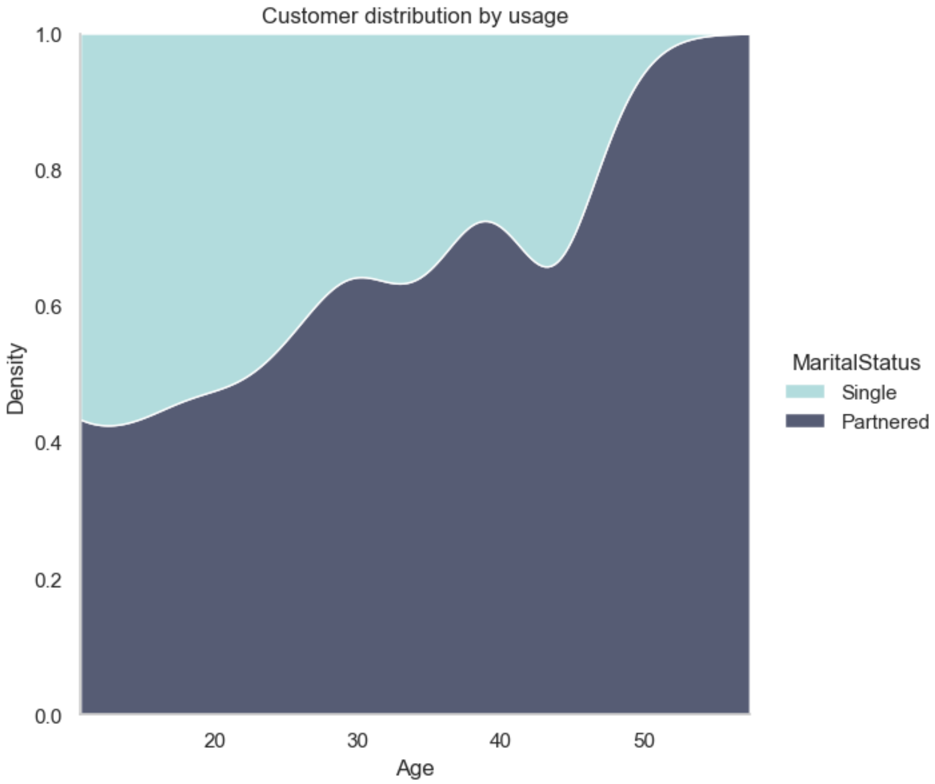
In [99]:
```
sns.set_theme(style="whitegrid")

sns.displot(
    data=DF,
    x="Age", hue="Usage",
    kind="kde", height=6,
    multiple="fill", clip=(0, None),
    palette="ch:rot=-.25,hue=1,light=.75",
)
plt.title('Customer distribution by usage')
plt.grid(False)
plt.show()
```



In [138]:
```
sns.set_theme(style="whitegrid")

sns.displot(
    data=DF,
    x="Age", hue="Product",
    kind="kde", height=6,
    multiple="fill", clip=(0, None),
    palette="ch:rot=-0.9,dark=0.3",
)
plt.title('Product usage distribution')
plt.grid(False)
plt.show()
```

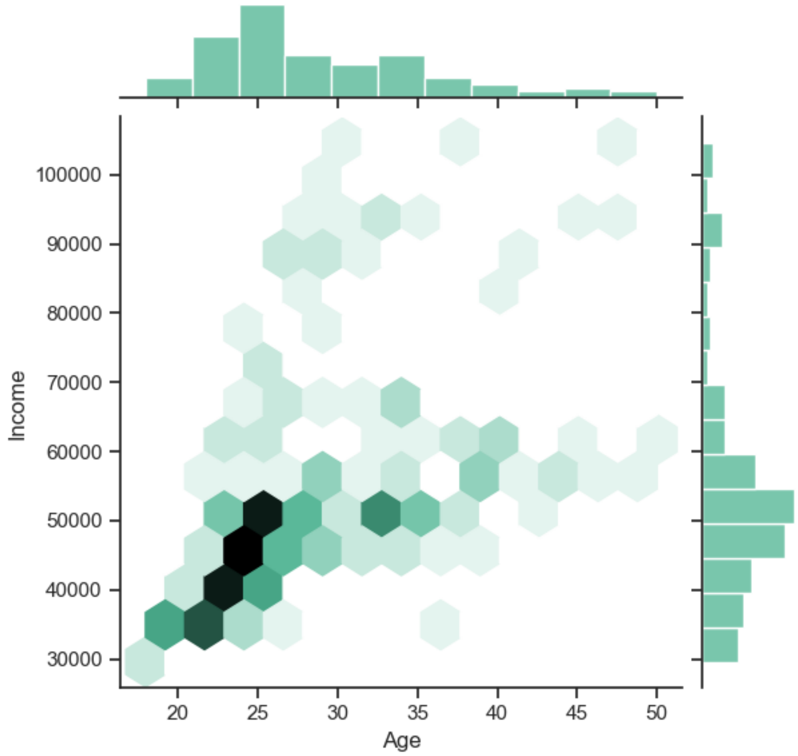


In [102]:
```
sns.set_theme(style="whitegrid")

sns.displot(
    data=DF,
    x="Age", hue="MaritalStatus",
    kind="kde", height=6,
    multiple="fill", clip=(0, None),
    palette="ch:rot=-.25,hue=1,light=.75",
)
plt.title('Customer distribution by usage')
plt.grid(False)
plt.show()
```



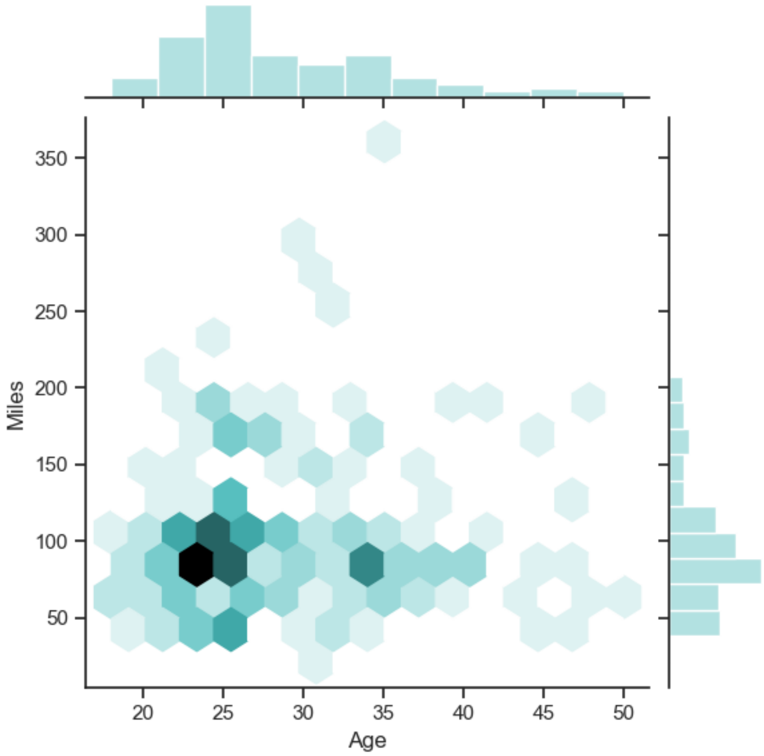
In [103]:

```
sns.set_theme(style="ticks")

sns.jointplot(DF,x='Age', y='Income', kind="hex", color="#4CB391")

plt.show()
```



In [104]:

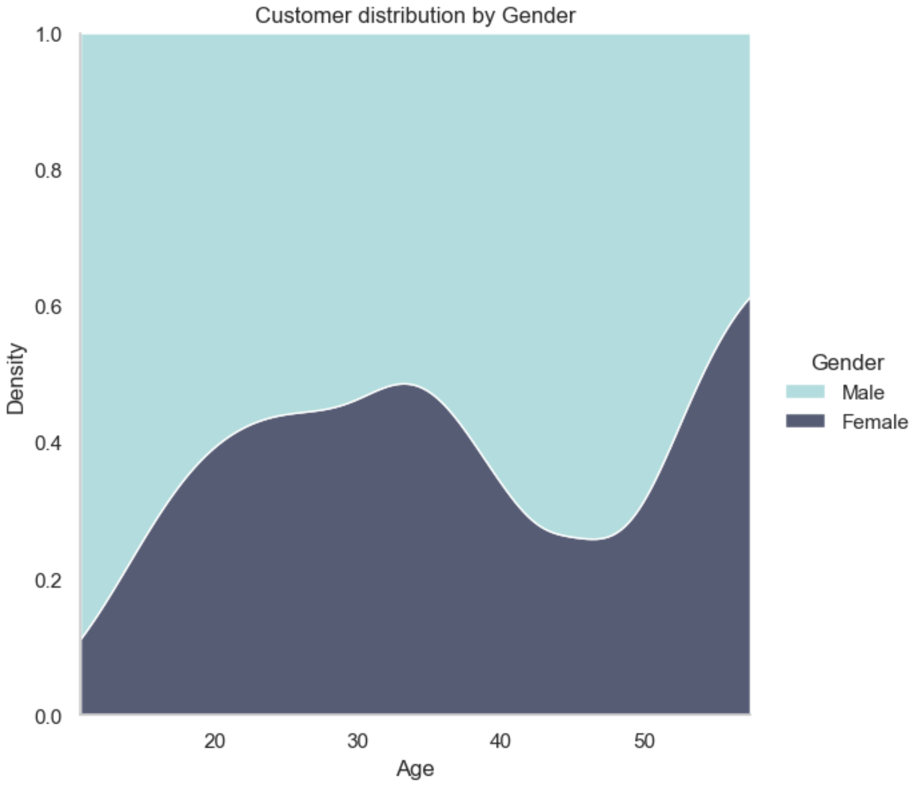
```
sns.set_theme(style="ticks")

sns.jointplot(DF,x='Age', y='Miles', kind="hex", color='#98D8D8')

plt.show()
```



In [107]:
```
DF['Income'].mode()
```

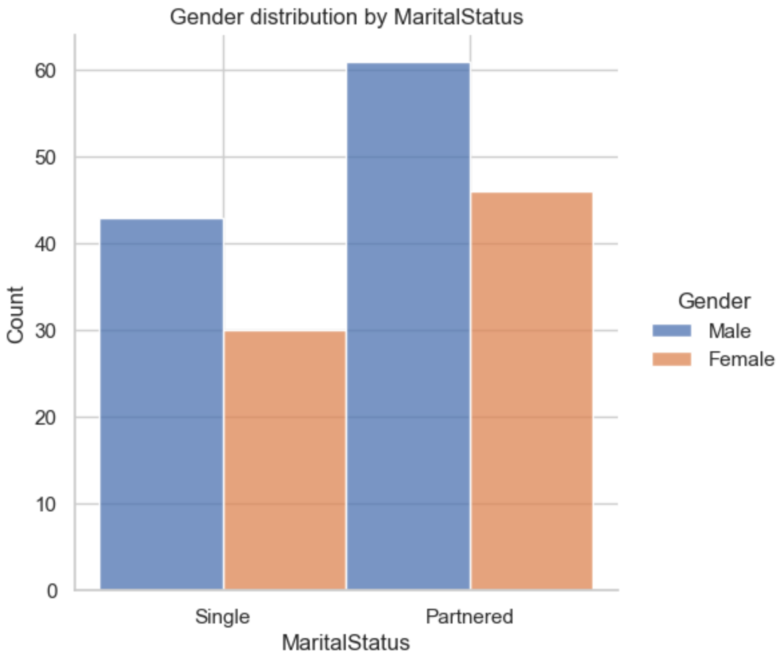
Out[107]:
```
0    45480
Name: Income, dtype: int64
```

In [134]:
```
sns.set_theme(style="whitegrid")

sns.displot(
    data=DF,
    x="Age", hue="Gender",
    kind="kde", height=6,
    multiple="fill", clip=(0, None),
    palette="ch:rot=-.25,hue=1,light=.75",
)
plt.title('Customer distribution by Gender')
plt.grid(False)
plt.show()
```



In [137]:

```
sns.displot(DF, x="MaritalStatus", hue="Gender",multiple="dodge")
plt.title('Gender distribution by MaritalStatus')
plt.show()
```



In [ ]: