AER

May 13, 2024

```
[ ]: #importing libraries
     import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     import warnings
     warnings.filterwarnings('ignore')
     import copy
```

```
[ ]: # Downloading The Data From GDrive
     !gdown 1KChvkol7HaEo0i2CUrU_2_WV5ZLd8x9-2msoB8n4_8E
     df = pd.read_excel("aerofit_treadmill.xlsx")
```

```
Downloading…
From (original):
https://drive.google.com/uc?id=1KChvkol7HaEo0i2CUrU_2_WV5ZLd8x9-2msoB8n4_8E
From (redirected): https://docs.google.com/spreadsheets/d/1KChvkol7HaEo0i2CUrU_2
_WV5ZLd8x9-2msoB8n4_8E/export?format=xlsx
To: /content/aerofit_treadmill.xlsx
11.4kB [00:00, 26.9MB/s]
```

```
[ ]: df
```

```
[ ]:     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
     ..      …   …     …         …            …     …      …      …
     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
     0      112
```

```
1       75
2       66
3       85
4       47
..      …
175    200
176    200
177    160
178    120
179    180

[180 rows x 9 columns]
```

#1- Defining Problem Statement and Analysing basic metrics

[ ]:
```
#finding the shape of the data

df.shape
```

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

[ ]:
```
#Finding the dimension of the data
df.ndim
```

[ ]:
```
2
```

[ ]:
```
#Finding the column names of the data
df.columns
```

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

[ ]:
```
#data type of all attributes
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
```

[ ]:
```
#statistical summary of the data for quick data analysis
df.describe()
```

[ ]:
```
              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
```

[ ]:
```
# finding out if any column contains null value
df.any().isnull()
```

[ ]:
```
Product          False
Age              False
Gender           False
Education        False
MaritalStatus    False
Usage            False
Fitness          False
Income           False
Miles            False
dtype: bool
```

[ ]:
```
df.isnull().sum()
```

[ ]:
```
Product          0
Age              0
Gender           0
```

```
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

It seems that our data does not have any of the columns that has any null value.

```
[ ]:
```

#2- Non-Graphical Analysis: Value counts and unique attributes.

```
[ ]: # checking for the value counts in all the columns
     num=1
     for i in df.columns:
         print('Value Counts in column number',num,"which is",i,'are :-')
         print(df[i].value_counts())
         print('-'*70)
         print(' '*70)
         print(' '*70)
         print(' '*70)
         print(' '*70)
         print('-'*70)
         num=num+1
```

```
Value Counts in column number 1 which is Product are :-
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 2 which is Age are :-
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
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 3 which is Gender are :-
Male      104
Female     76
Name: Gender, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 4 which is Education are :-
16    85
14    55
18    23
15     5
13     5
12     3
21     3
```

```
20     1
Name: Education, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 5 which is MaritalStatus are :-
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 6 which is Usage are :-
3    69
4    52
2    33
5    17
6     7
7     2
Name: Usage, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 7 which is Fitness are :-
3    97
5    31
2    26
4    24
1     2
Name: Fitness, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
Value Counts in column number 8 which is Income are :-
45480    14
52302     9
```

```
46617     8
54576     8
53439     8
         ..
65220     1
55713     1
68220     1
30699     1
95508     1
Name: Income, Length: 62, dtype: int64
----------------------------------------------------------------------
```

```
----------------------------------------------------------------------
Value Counts in column number 9 which is Miles are :-
85     27
95     12
66     10
75     10
47      9
106     9
94      8
113     8
53      7
100     7
180     6
200     6
56      6
64      6
127     5
160     5
42      4
150     4
38      3
74      3
170     3
120     3
103     3
132     2
141     2
280     1
260     1
300     1
240     1
112     1
212     1
```

```
80       1
140      1
21       1
169      1
188      1
360      1
Name: Miles, dtype: int64
----------------------------------------------------------------------




----------------------------------------------------------------------
```

[ ]:
```
# checking the unique attributes
num=1
for i in df.columns:
    print('Unique Values in column number',num,"which is",i,'are :-')
    print(df[i].unique())
    print('-'*70)
    print(' '*70)
    print(' '*70)
    print(' '*70)
    print(' '*70)
    print('-'*70)
    num=num+1
# for i in df.columns:
#     print('Unique Values in',i,'column are :-')
#     print(df[i].unique())
#     print('-'*70)
```

```
Unique Values in column number 1 which is Product are :-
['KP281' 'KP481' 'KP781']
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 2 which is Age are :-
[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]
----------------------------------------------------------------------




----------------------------------------------------------------------
```

```
Unique Values in column number 3 which is Gender are :-
['Male' 'Female']
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 4 which is Education are :-
[14 15 12 13 16 18 20 21]
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 5 which is MaritalStatus are :-
['Single' 'Partnered']
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 6 which is Usage are :-
[3 2 4 5 6 7]
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 7 which is Fitness are :-
[4 3 2 1 5]
----------------------------------------------------------------------




----------------------------------------------------------------------
Unique Values in column number 8 which is Income are :-
[ 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]
```

```
----------------------------------------------------------------------
```

```
----------------------------------------------------------------------
Unique Values in column number 9 which is Miles are :-
[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]
----------------------------------------------------------------------
```

```
----------------------------------------------------------------------
```

#3- Visual Analysis - Univariate & Bivariate

0.1 3.1 Univariate Analysis

Adding additional columns to create a better understanding about data and for better visualization.

Age Column

Categorizing the values in age column in 4 different buckets:

1. Young Adult: from 18 - 25
2. Adults: from 26 - 35
3. Middle Aged Adults: 36 - 45
4. Elder : 46 and above

Education Column

Categorizing the values in education column in 3 different buckets:

1. Primary Education: upto 12
2. Secondary Education: 13 - 15
3. Higher Education: 16 and above

Income Column

Categorizing the values in Income column in 4 different buckets:

1. Low Income - upto 40,000
2. Moderate Income - 40,000 to 60,000
3. High Income - 60,000 to 80,000
4. Very High Income - 80,000 and above

Miles column

Categorizing the values in miles column in 4 different buckets:

1. Light Activity - Upto 50 miles

2. Moderate Activity - 51 to 100 miles
3. Active Lifestyle - 101 to 200 miles
4. Fitness Enthusiast - Above 200 miles

[ ]:
```
#binning the age values into categories
bin_range1 = [17,25,35,45,float('inf')]
bin_labels1 = ['Young Adults', 'Adults', 'Middle Aged Adults', 'Elder']

df['age_group'] = pd.cut(df['Age'],bins = bin_range1,labels = bin_labels1)

#binning the education values into categories
bin_range2 = [0,12,15,float('inf')]
bin_labels2 = ['Primary Education', 'Secondary Education', 'Higher Education']

df['edu_group'] = pd.cut(df['Education'],bins = bin_range2,labels = bin_labels2)

#binning the income values into categories
bin_range3 = [0,40000,60000,80000,float('inf')]
bin_labels3 = ['Low Income','Moderate Income','High Income','Very High Income']

df['income_group'] = pd.cut(df['Income'],bins = bin_range3,labels = bin_labels3)

#binning the miles values into categories
bin_range4 = [0,50,100,200,float('inf')]
bin_labels4 = ['Light Activity', 'Moderate Activity', 'Active Lifestyle',␣
 ↪'Fitness Enthusiast ']

df['miles_group'] = pd.cut(df['Miles'],bins = bin_range4,labels = bin_labels4)
df
```

[ ]:
```
    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
..      …   …     …          …           …     …       …     …
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          age_group            edu_group      income_group  \
0      112       Young Adults  Secondary Education        Low Income
1       75       Young Adults  Secondary Education        Low Income
2       66       Young Adults  Secondary Education        Low Income
3       85       Young Adults    Primary Education        Low Income
```

```
4       47          Young Adults  Secondary Education        Low Income
..      …                      …                    …                …
175    200  Middle Aged Adults     Higher Education  Very High Income
176    200  Middle Aged Adults     Higher Education  Very High Income
177    160  Middle Aged Adults     Higher Education  Very High Income
178    120               Elder     Higher Education  Very High Income
179    180               Elder     Higher Education  Very High Income

            miles_group
0     Active Lifestyle
1    Moderate Activity
2    Moderate Activity
3    Moderate Activity
4       Light Activity
..                 …
175   Active Lifestyle
176   Active Lifestyle
177   Active Lifestyle
178   Active Lifestyle
179   Active Lifestyle

[180 rows x 13 columns]
```

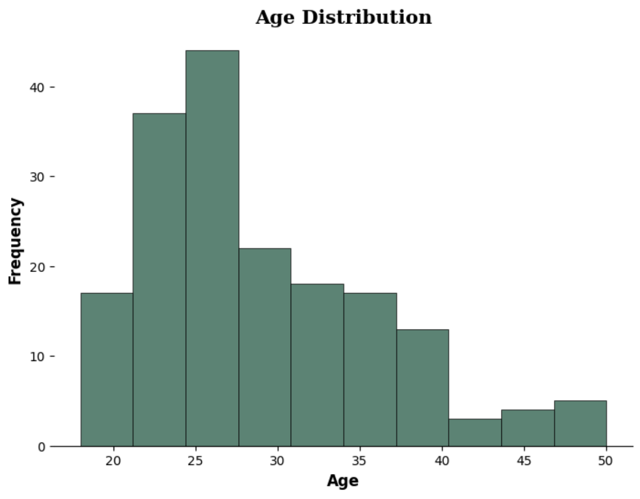
###3.1.1 For Continous variable

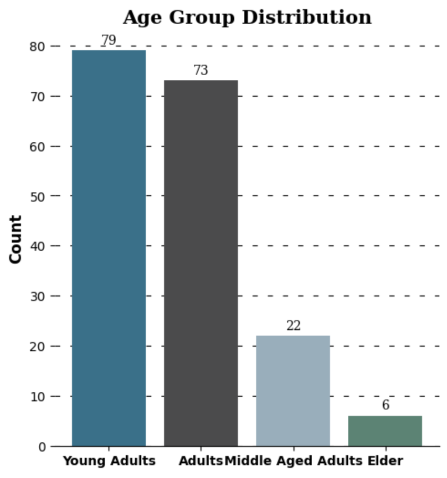
3.1.1.1 Customer Age Distribution

[ ]:
```
#setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

                                    #creating age histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Age'],color= '#5C8374',linewidth=0.5,edgecolor='black')
ax0.set_xlabel('Age',fontsize = 12,fontweight = 'bold')
ax0.set_ylabel('Frequency',fontsize = 12,fontweight = 'bold')

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#setting title for visual
ax0.set_title('Age Distribution',{'font':'serif', 'size':15,'weight':'bold'})

                                    #creating age group bar chart
```

```
ax2 = fig.add_subplot(gs[0,1])
temp = df['age_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax2.set_title('Age Group Distribution',{'font':'serif', 'size':15,'weight':
 ↪'bold'})
```

```
[ ]: Text(0.5, 1.0, 'Age Group Distribution')
```





Insights

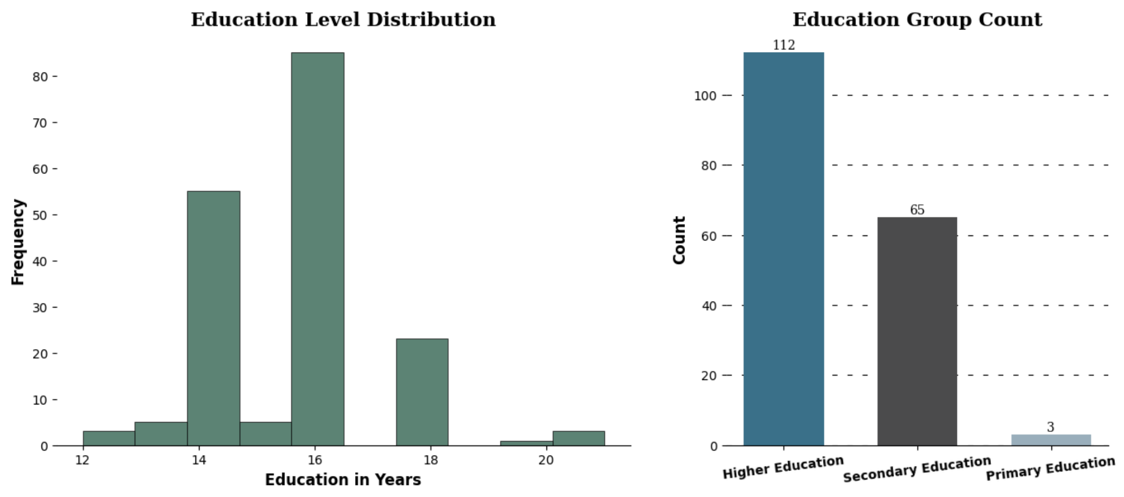
- 85% of the customers fall in the age range of 18 to 35. with a median age of 26, suggesting young people showing more interest in the companies products

3.1.1.2 Customer Education Distribution

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

                                    #creating education histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Education'],color= '#5C8374',linewidth=0.5,edgecolor='black')
ax0.set_xlabel('Education in Years',fontsize = 12,fontweight = 'bold')
ax0.set_ylabel('Frequency',fontsize = 12,fontweight = 'bold')

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#setting title for visual
ax0.set_title('Education Level Distribution',{'font':'serif', 'size':
 ↪15,'weight':'bold'})

ax2 = fig.add_subplot(gs[0,1])
temp = df['edu_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2,width =␣
 ↪0.6)

#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 7)
```

```
#setting title for visual
ax2.set_title('Education Group Count',{'font':'serif', 'size':15,'weight':
 ↪'bold'})
```

```
[ ]: Text(0.5, 1.0, 'Education Group Count')
```



Insights

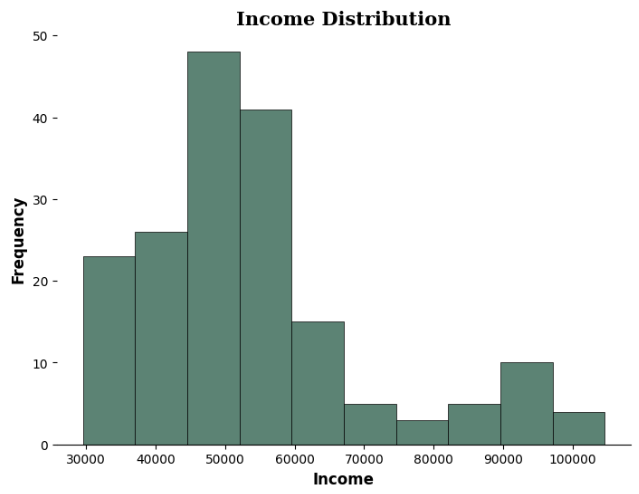
- 98% of the customers have education more than 13 years highlighting a strong inclination among well-educated individuals to purchase the products. It's plausible that health awareness driven by education could play a pivotal role in this trend.

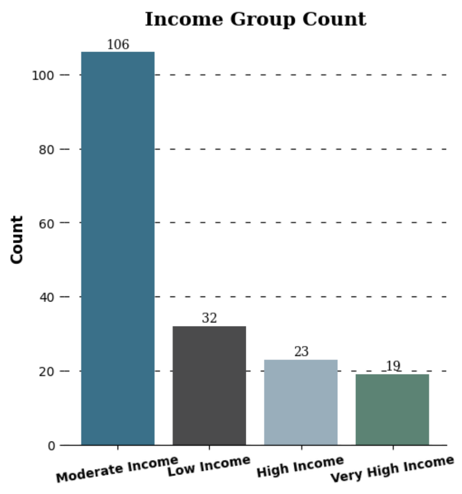
####3.1.1.3 Customer Income Distribution

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

                                    #creating Income histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Income'],color= '#5C8374',linewidth=0.5,edgecolor='black')
ax0.set_xlabel('Income',fontsize = 12,fontweight = 'bold')
ax0.set_ylabel('Frequency',fontsize = 12,fontweight = 'bold')

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)
```

```
#setting title for visual
ax0.set_title('Income Distribution',{'font':'serif', 'size':15,'weight':'bold'})

#creating Income group bar chart

ax2 = fig.add_subplot(gs[0,1])
temp = df['income_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 9)

#setting title for visual
ax2.set_title('Income Group Count',{'font':'serif', 'size':15,'weight':'bold'})
```

[ ]:
```
Text(0.5, 1.0, 'Income Group Count')
```





Insights

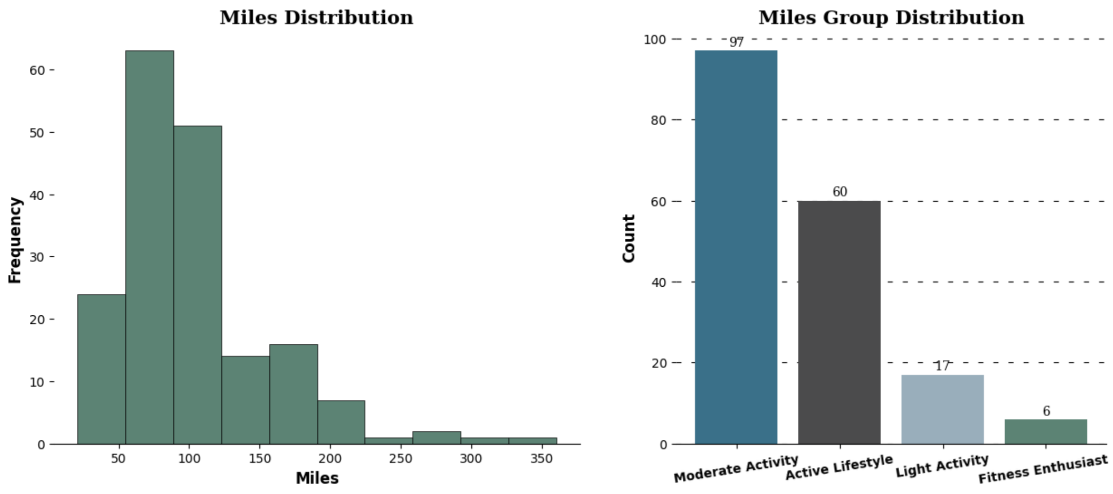
- Almost 60% of the customers fall in the income group of (40k to 60k) dollars suggesting higher inclination of this income group people towards the products.
- Surprisingly 18% of the customers fall in the income group of (<40) suggesting almost 77% of the total customers fall in income group of below 60k and only 23% of them falling in 60k and above income group

####3.1.1.4 Customer Expected Weekely Milage

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.55,0.45])

                                    #creating miles histogram

ax0 = fig.add_subplot(gs[0,0])

ax0.hist(df['Miles'],color= '#5C8374',linewidth=0.5,edgecolor='black')
ax0.set_xlabel('Miles',fontsize = 12,fontweight = 'bold')
ax0.set_ylabel('Frequency',fontsize = 12,fontweight = 'bold')

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#setting title for visual
ax0.set_title('Miles Distribution',{'font':'serif', 'size':15,'weight':'bold'})

#creating Miles group bar chart

ax2 = fig.add_subplot(gs[0,1])
temp = df['miles_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))
```

```
#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 9)

#setting title for visual
ax2.set_title('Miles Group Distribution',{'font':'serif', 'size':15,'weight':
 ↪'bold'})
```

```
[ ]: Text(0.5, 1.0, 'Miles Group Distribution')
```



Insights

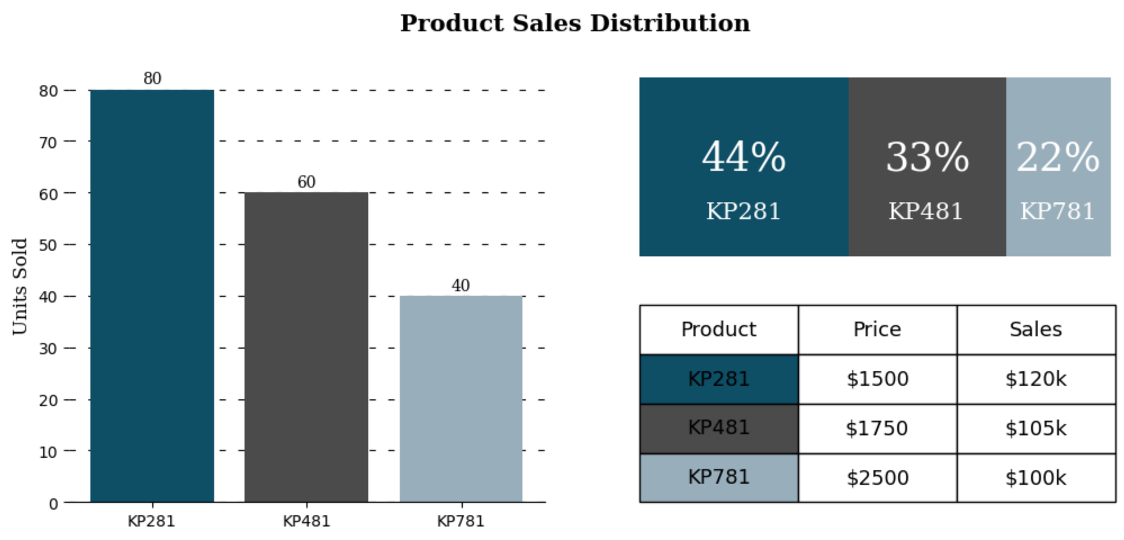
- Almost 88% of the customers plans to use the treadmill for `50 to 200 miles` per week with a median of `94 miles per week`.

###3.1.2 For categorical variable

3.1.2.1 Product Sales Distribution

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (12,5))
gs = fig.add_gridspec(2,2)

                                            #creating plot for product column

ax0 = fig.add_subplot(gs[:,0])
```

```
product_count = df['Product'].value_counts()

color_map = ["#0e4f66", "#4b4b4c",'#99AEBB']

ax0.bar(product_count.index,product_count.values,color = color_map,zorder = 2)

#adding the value_counts
for i in product_count.index:
    ax0.text(i,product_count[i]+2,product_count[i],{'font':'serif','size' :␣
 ↪10},ha = 'center',va = 'center')

#adding grid lines
ax0.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#adding axis label
ax0.set_ylabel('Units Sold',fontfamily='serif',fontsize = 12)





                                                        # creating a plot for product␣
 ↪% sale
ax1 = fig.add_subplot(gs[0,1])

product_count['percent'] = ((product_count.values/df.shape[0])* 100).round()

ax1.barh(product_count.index[0],product_count.loc['percent'][0],color =␣
 ↪"#0e4f66")
ax1.barh(product_count.index[0],product_count.loc['percent'][1],left =␣
 ↪product_count.loc['percent'][0],color = '#4b4b4c')
ax1.barh(product_count.index[0],product_count.loc['percent'][2],
         left = product_count.loc['percent'][0] + product_count.
 ↪loc['percent'][1], color = '#99AEBB')
ax1.set(xlim=(0,100))


# adding info to the each bar
product_count['info_percent'] =[product_count['percent'][0]/
 ↪2,product_count['percent'][0] + product_count['percent'][1]/2,
                                  product_count['percent'][0] +␣
 ↪product_count['percent'][1] + product_count['percent'][2]/2]
for i in range(3):
    ax1.text(product_count['info_percent'][i],0.
 ↪04,f"{product_count['percent'][i]:.0f}%",
             va = 'center', ha='center',fontsize=25, fontweight='light',␣
 ↪fontfamily='serif',color='white')

    ax1.text(product_count['info_percent'][i],-0.2,product_count.index[i],
             va = 'center', ha='center',fontsize=15, fontweight='light',␣
 ↪fontfamily='serif',color='white')

#removing the axis lines
ax1.axis('off')





                                                             #creating a plot for product␣
 ↪portfolio
ax2 = fig.add_subplot(gs[1,1])

product_portfolio =␣
 ↪[['KP281','$1500','$120k'],['KP481','$1750','$105k'],['KP781','$2500','$100k']]
color_2d =␣
 ↪[['#0e4f66','#FFFFFF','#FFFFFF'],['#4b4b4c','#FFFFFF','#FFFFFF'],['#99AEBB','#FFFFFF','#FFF

table = ax2.table(cellText = product_portfolio, cellColours=color_2d,␣
 ↪cellLoc='center',colLabels =['Product','Price','Sales'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax2.axis('off')

#adding title to the visual
fig.suptitle('Product Sales Distribution',fontproperties = {'family':'serif',␣
 ↪'size':15,'weight':'bold'})

plt.show()
```



Product	Price	Sales
KP281	$1500	$120k
KP481	$1750	$105k
KP781	$2500	$100k

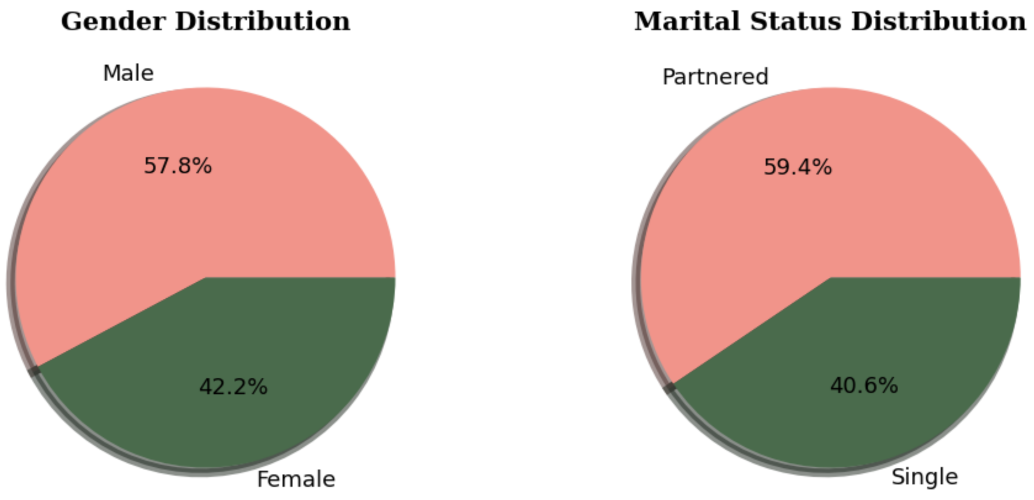
Insights

1. According to this data, KP281 treadmill model, positioned as an entry-level product, we can conclude that, it has the highest number of units sold, trailed by the KP481 (mid-level) and KP781 (advanced) models.
2. On the other hand, we can also say that all three models have nearly equal contributions in terms of generating sales revenue.

3.1.2.2 Gender and Marital Status Disribution

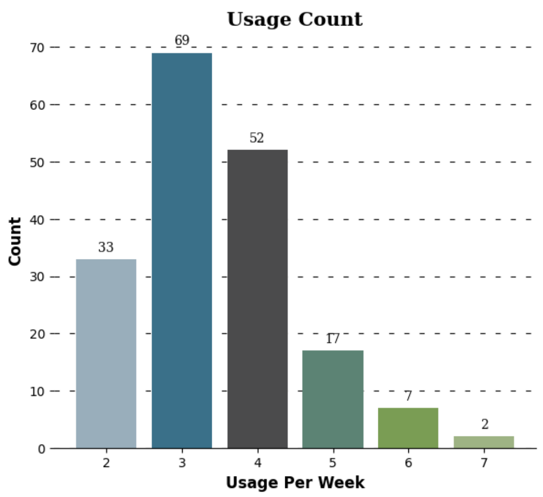
[ ]:
```
#setting the plot style
fig = plt.figure(figsize = (12,5))
gs = fig.add_gridspec(1,2)

                                        # creating pie chart for gender␣
 ↪disribution
ax0 = fig.add_subplot(gs[0,0])

color_map = ["#F1948A", "#4a6b4c"]
ax0.pie(df['Gender'].value_counts().values,labels = df['Gender'].value_counts().
 ↪index,autopct = '%.1f%%',
        shadow = True,colors = color_map,wedgeprops = {'linewidth':␣
 ↪5},textprops={'fontsize': 13, 'color': 'black'})

#setting title for visual
ax0.set_title('Gender Distribution',{'font':'serif', 'size':15,'weight':'bold'})

                                        # creating pie chart for marital status
ax1 = fig.add_subplot(gs[0,1])
```

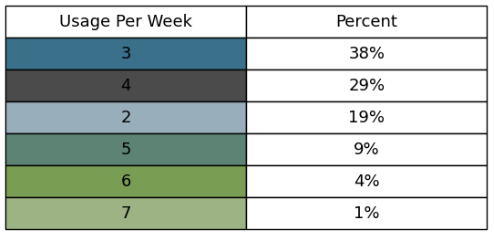
```
color_map = ["#F1948A", "#4a6b4c"]
ax1.pie(df['MaritalStatus'].value_counts().values,labels = df['MaritalStatus'].
 ↪value_counts().index,autopct = '%.1f%%',
        shadow = True,colors = color_map,wedgeprops = {'linewidth':␣
 ↪5},textprops={'fontsize': 13, 'color': 'black'})

#setting title for visual
ax1.set_title('Marital Status Distribution',{'font':'serif', 'size':15,'weight':
 ↪'bold'})

plt.show()
```



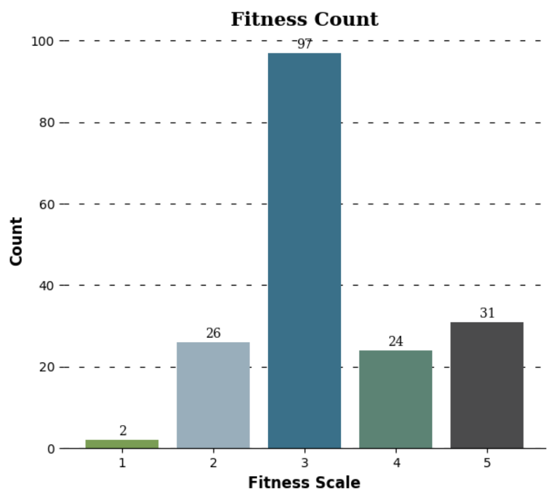
3.1.2.3 Buyer Fitness and Treadmill Usage

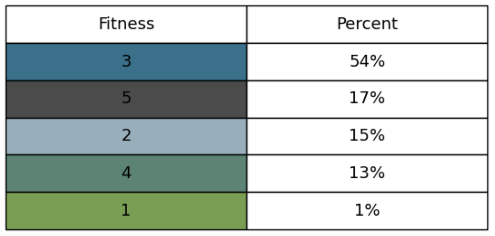
[ ]:
```
#setting the plot style
fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35])

                                        # creating bar chart for usage␣
 ↪disribution

ax0 = fig.add_subplot(gs[0,0])
temp = df['Usage'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374','#7A9D54','#9EB384']
ax0.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
for i in temp.index:
    ax0.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax0.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax0.spines[s].set_visible(False)

#adding axis label
ax0.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax0.set_xlabel('Usage Per Week',fontweight = 'bold',fontsize = 12)
#ax0.set_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax0.set_title('Usage Count',{'font':'serif', 'size':15,'weight':'bold'})

                                        #creating a info table for usage

ax1 = fig.add_subplot(gs[1,0])
usage_info =␣
 ↪[['3','38%'],['4','29%'],['2','19%'],['5','9%'],['6','4%'],['7','1%']]
color_2d =␣
 ↪[["#3A7089",'#FFFFFF'],["#4b4b4c",'#FFFFFF'],['#99AEBB','#FFFFFF'],['#5C8374','#FFFFFF'],['
           ['#9EB384','#FFFFFF']]

table = ax1.table(cellText = usage_info, cellColours=color_2d,␣
 ↪cellLoc='center',colLabels =['Usage Per Week','Percent'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax1.axis('off')

                                        # creating bar chart for fitness scale

ax2 = fig.add_subplot(gs[0,1])
temp = df['Fitness'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374','#7A9D54','#9EB384']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xlabel('Fitness Scale',fontweight = 'bold',fontsize = 12)
#ax2.set_xticklabels(temp.index,fontweight = 'bold')

#setting title for visual
ax2.set_title('Fitness Count',{'font':'serif', 'size':15,'weight':'bold'})

                                        #creating a info table for usage

ax1 = fig.add_subplot(gs[1,1])
fitness_info = [['3','54%'],['5','17%'],['2','15%'],['4','13%'],['1','1%']]
color_2d =␣
 ↪[["#3A7089",'#FFFFFF'],["#4b4b4c",'#FFFFFF'],['#99AEBB','#FFFFFF'],['#5C8374','#FFFFFF'],['

table = ax1.table(cellText = fitness_info, cellColours=color_2d,␣
 ↪cellLoc='center',colLabels =['Fitness','Percent'],
                  colLoc = 'center',bbox =[0, 0, 1, 1])

table.set_fontsize(13)

#removing axis
ax1.axis('off')


plt.show()
```





Usage Per Week	Percent
3	38%
4	29%
2	19%
5	9%
6	4%
7	1%





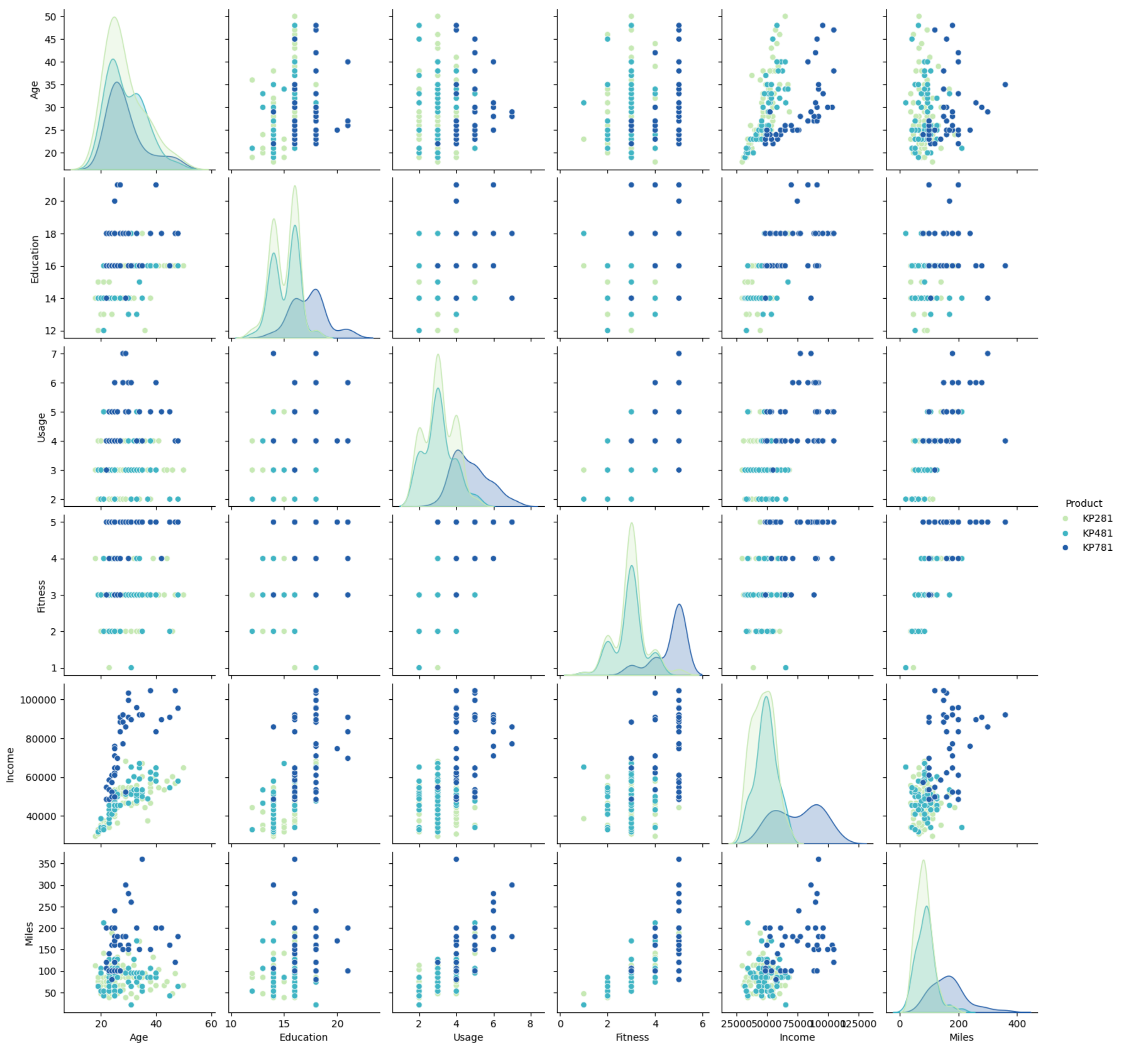
Fitness	Percent
3	54%
5	17%
2	15%
4	13%
1	1%

###3.3 For Correlation

3.3.1 Pairplot

```
[ ]: df_copy = copy.deepcopy(df)
```

```
[ ]: sns.pairplot(df_copy, hue ='Product', palette= 'YlGnBu')
     plt.show()
```

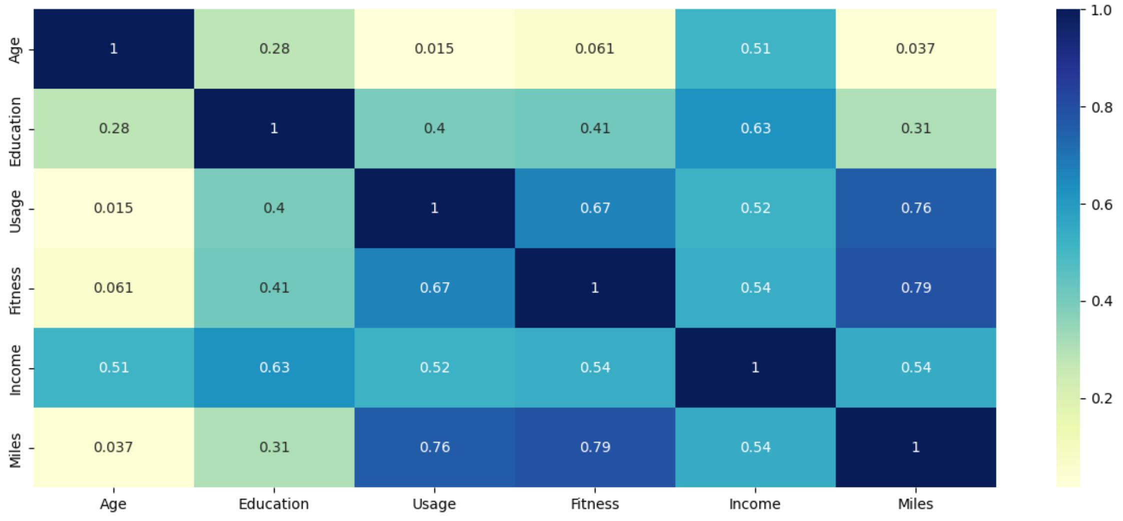


3.3.2 Heatmap

```
[ ]: # First we need to convert object into int datatype for usage and fitness␣
      ↪columns

     df_copy['Usage'] = df_copy['Usage'].astype('int')
     df_copy['Fitness'] = df_copy['Fitness'].astype('int')

     df_copy.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 13 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
 9   age_group      180 non-null    category
 10  edu_group      180 non-null    category
 11  income_group   180 non-null    category
 12  miles_group    180 non-null    category
dtypes: category(4), int64(6), object(3)
memory usage: 14.2+ KB
```

[ ]:
```
corr_mat = df_copy.corr()

plt.figure(figsize=(15,6))

sns.heatmap(corr_mat,annot = True, cmap="YlGnBu")

plt.show()
```



Insights

- From the pair plot we can see `Age and Income` are **positively correlated** and heatmap also suggests a **strong correlation** betwwen them
- `Eductaion` and `Income` are highly correlated as its obvious. Eductation also has significatnt correlation between `Fitness rating and Usage of the treadmill.`

- `Usage` is highly correlated with `Fitness and Miles` as more the usage more the fitness and mileage.

##3.4 Probability

Gender

```
[ ]: pd.crosstab(index =df['Product'],columns = df['Gender'],margins =␣
      ↪True,normalize = True ).round(2)
```

```
[ ]: Gender   Female  Male   All
     Product
     KP281      0.22  0.22  0.44
     KP481      0.16  0.17  0.33
     KP781      0.04  0.18  0.22
     All        0.42  0.58  1.00
```

Insights

1. The **`Probability`** of a treadmill being purchased by a **`female is 42%`**.
   - **`The conditional probability`** of purchasing the treadmill model given that the customer is **`female`** is
     - For Treadmill model KP281 - **22%**
     - For Treadmill model KP481 - **16%**
     - For Treadmill model KP781 - **4%**
2. The **`Probability`** of a treadmill being purchased by a **`male is 58%`**.
   - **`The conditional probability`** of purchasing the treadmill model given that the customer is **`male`** is -
     - For Treadmill model KP281 - **22%**
     - For Treadmill model KP481 - **17%**
     - For Treadmill model KP781 - **18%**

Age

```
[ ]: pd.crosstab(index =df['Product'],columns = df['age_group'],margins =␣
      ↪True,normalize = True ).round(2)
```

```
[ ]: age_group  Young Adults  Adults  Middle Aged Adults  Elder   All
     Product
     KP281              0.19    0.18                0.06   0.02  0.44
     KP481              0.16    0.13                0.04   0.01  0.33
     KP781              0.09    0.09                0.02   0.01  0.22
     All                0.44    0.41                0.12   0.03  1.00
```

*** Insights***

1. The **Probability** of a treadmill being purchased by a **Young Adult(18-25) is 44%**.
   - **The conditional probability** of purchasing the treadmill model given that the customer is **Young Adult** is
     - For Treadmill model KP281 - **19%**
     - For Treadmill model KP481 - **16%**
     - For Treadmill model KP781 - **9%**
2. The **Probability** of a treadmill being purchased by a **Adult(26-35) is 41%**.
   - **The conditional probability** of purchasing the treadmill model given that the customer is **Adult** is -
     - For Treadmill model KP281 - **18%**
     - For Treadmill model KP481 - **13%**
     - For Treadmill model KP781 - **9%**
3. The **Probability** of a treadmill being purchased by a **Middle Aged(36-45) is 12%**.
4. The **Probability** of a treadmill being purchased by a **Elder(Above 45) is only 3%**.

Income

```
[ ]: pd.crosstab(index =df['Product'],columns = df['income_group'],margins =␣
      ↪True,normalize = True ).round(2)
```

```
[ ]: income_group  Low Income  Moderate Income  High Income  Very High Income   All
     Product
     KP281               0.13             0.28         0.03              0.00  0.44
     KP481               0.05             0.24         0.04              0.00  0.33
     KP781               0.00             0.06         0.06              0.11  0.22
     All                 0.18             0.59         0.13              0.11  1.00
```

Insights

1. The **Probability** of a treadmill being purchased by a customer with **Low Income(<40k) is 18%**.
   - **The conditional probability** of purchasing the treadmill model given that the customer has **Low Income** is -
     - For Treadmill model KP281 - **13%**
     - For Treadmill model KP481 - **5%**
     - For Treadmill model KP781 - **0%**
2. The **Probability** of a treadmill being purchased by a customer with **Moderate Income(40k - 60k) is 59%**.
   - **The conditional probability** of purchasing the treadmill model given that the customer has **Moderate Income** is -
     - For Treadmill model KP281 - **28%**
     - For Treadmill model KP481 - **24%**

    - For Treadmill model KP781 - **6%**

3. The **Probability** of a treadmill being purchased by a customer with **High Income(60k - 80k) is 13%**

   - **The conditional probability** of purchasing the treadmill model given that the customer has **High Income** is -
     - For Treadmill model KP281 - **3%**
     - For Treadmill model KP481 - **4%**
     - For Treadmill model KP781 - **6%**

4. The **Probability** of a treadmill being purchased by a customer with **Very High Income(>80k) is 11%**

   - **The conditional probability** of purchasing the treadmill model given that the customer has **High Income** is -
     - For Treadmill model KP281 - **0%**
     - For Treadmill model KP481 - **0%**
     - For Treadmill model KP781 - **11%**

Marital Status

```
[ ]: pd.crosstab(index =df['Product'],columns = df['MaritalStatus'],margins =␣
      ↪True,normalize = True ).round(2)
```

```
[ ]: MaritalStatus  Partnered  Single   All
     Product
     KP281               0.27    0.18  0.44
     KP481               0.20    0.13  0.33
     KP781               0.13    0.09  0.22
     All                 0.59    0.41  1.00
```

Insights

1. The **Probability** of a treadmill being purchased by a **Married Customer is 59%**.

   - **The conditional probability** of purchasing the treadmill model given that the customer is **Married** is
     - For Treadmill model KP281 - **27%**
     - For Treadmill model KP481 - **20%**
     - For Treadmill model KP781 - **13%**

2. The **Probability** of a treadmill being purchased by a **Unmarried Customer is 41%**.

   - **The conditional probability** of purchasing the treadmill model given that the customer is **Unmarried** is -
     - For Treadmill model KP281 - **18%**

– For Treadmill model KP481 - **13%**

– For Treadmill model KP781 - **9%**

1 4 - Missing Value & Outlier Detection

##4.1 Missing Value Detection

```
[ ]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 13 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
 9   age_group      180 non-null    category
 10  edu_group      180 non-null    category
 11  income_group   180 non-null    category
 12  miles_group    180 non-null    category
dtypes: category(4), int64(6), object(3)
memory usage: 14.2+ KB
```

```
[ ]: # checking for the value counts in all the columns
     for i in df.columns:
         print('Missing column number',num,"which is",i,'are :-')
         print(df[i].isnull().sum())
```

```
Missing column number 1 which is Product are :-
0
Missing column number 1 which is Age are :-
0
Missing column number 1 which is Gender are :-
0
Missing column number 1 which is Education are :-
0
Missing column number 1 which is MaritalStatus are :-
0
Missing column number 1 which is Usage are :-
0
```

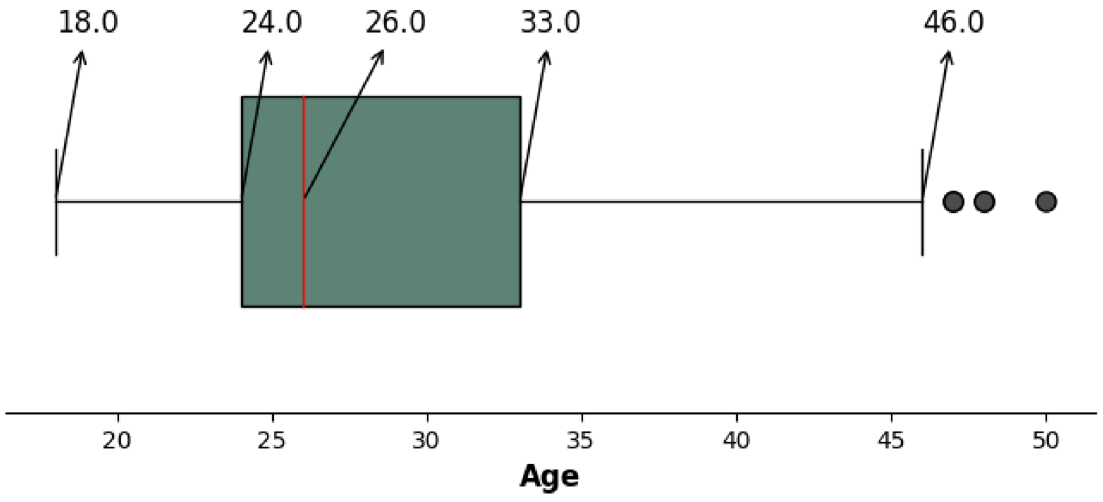
```
Missing column number 1 which is Fitness are :-
0
Missing column number 1 which is Income are :-
0
Missing column number 1 which is Miles are :-
0
Missing column number 1 which is age_group are :-
0
Missing column number 1 which is edu_group are :-
0
Missing column number 1 which is income_group are :-
0
Missing column number 1 which is miles_group are :-
0
```

We can clearly see that there are no missing values in any of the columns.

1.1 4.2 Outliers Detection

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

#creating box plot for age


ax1 = fig.add_subplot(gs[1,0])
boxplot = ax1.boxplot(x = df['Age'],vert = False,patch_artist = True,widths = 0.
 ↪5)

# Customize box and whisker colors
boxplot['boxes'][0].set(facecolor='#5C8374')

# Customize median line
boxplot['medians'][0].set(color='red')

# Customize outlier markers
for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor= "#4b4b4c")

#removing the axis lines
for s in ['top','left','right']:
    ax1.spines[s].set_visible(False)

#adding 5 point summary annotations
info = [i.get_xdata() for i in boxplot['whiskers']] #getting the␣
 ↪upperlimit,Q1,Q3 and lowerlimit
```

```
median = df['Age'].quantile(0.5) #getting Q2

for i,j in info: #using i,j here because of the output type of info list␣
  ↪comprehension

    ax1.annotate(text = f"{i:.1f}", xy = (i,1), xytext = (i,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1,␣
  ↪connectionstyle="arc,rad=0"))

    ax1.annotate(text = f"{j:.1f}", xy = (j,1), xytext = (j,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1,␣
  ↪connectionstyle="arc,rad=0"))

#adding the median separately because it was included in info list
ax1.annotate(text = f"{median:.1f}",xy = (median,1),xytext = (median + 2,1.
  ↪4),fontsize = 12,
            arrowprops= dict(arrowstyle="<-", lw=1,␣
  ↪connectionstyle="arc,rad=0"))

#removing y-axis ticks
ax1.set_yticks([])

#adding axis label
ax1.set_xlabel('Age',fontweight = 'bold',fontsize = 12)
plt.show()
```

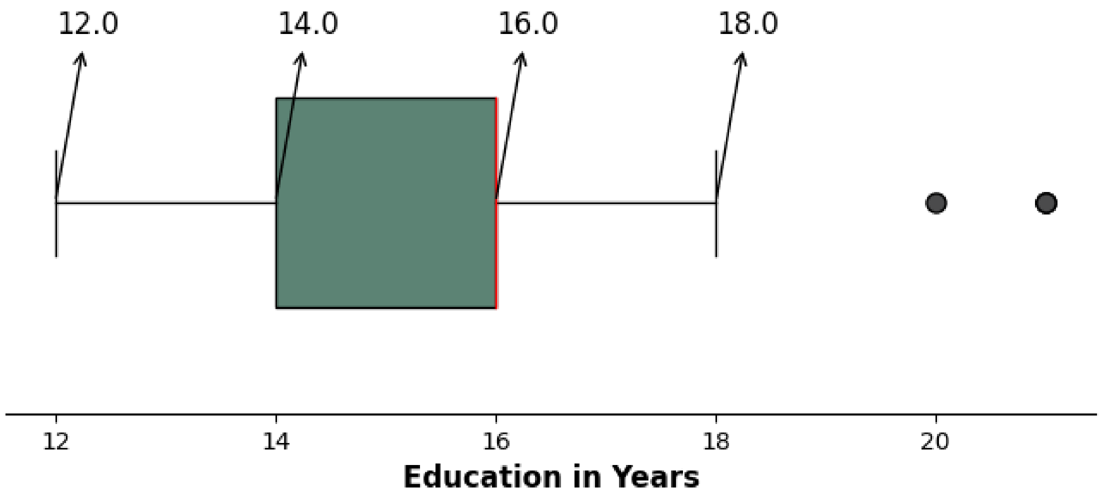


- **Outliers**
  - As we can see from the box plot, there are `3 outlier's` present in the age data.

Customer Education

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

 #creating box plot for education

ax1 = fig.add_subplot(gs[1,0])
boxplot = ax1.boxplot(x = df['Education'],vert = False,patch_artist =␣
 ↪True,widths = 0.5)

# Customize box and whisker colors
boxplot['boxes'][0].set(facecolor='#5C8374')

# Customize median line
boxplot['medians'][0].set(color='red')

# Customize outlier markers
for flier in boxplot['fliers']:
    flier.set(marker='o', markersize=8, markerfacecolor= "#4b4b4c")

#removing the axis lines
for s in ['top','left','right']:
    ax1.spines[s].set_visible(False)

#adding 5 point summary annotations
info = [i.get_xdata() for i in boxplot['whiskers']] #getting the␣
 ↪upperlimit,Q1,Q3 and lowerlimit

median = df['Education'].quantile(0.5) #getting Q2

for i,j in info: #using i,j here because of the output type of info list␣
 ↪comprehension

    ax1.annotate(text = f"{i:.1f}", xy = (i,1), xytext = (i,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1,␣
 ↪connectionstyle="arc,rad=0"))

    ax1.annotate(text = f"{j:.1f}", xy = (j,1), xytext = (j,1.4),fontsize = 12,
                 arrowprops= dict(arrowstyle="<-", lw=1,␣
 ↪connectionstyle="arc,rad=0"))


#removing y-axis ticks
ax1.set_yticks([])
```

```
#adding axis label
ax1.set_xlabel('Education in Years',fontweight = 'bold',fontsize = 12)

plt.show()
```



- **Outliers**
  - As we can see from the box plot, there are `2 outlier's` present in the education data.

2 5 - Business Insights based on Non-Graphical and Visual Analysis

##5.1 Comments on the range of attributes

Insights

1. Product - Over the past three months, the `KP281` product demonstrated the `highest sales performance` among the three products, accounting for approximately `44%` of total sales.

2. Gender - Based on the data of last 3 months, around `58%` of the buyers were `Male` and `42%` were `female`

3. Marital Status - Based on the data of last 3 months, around `60%` of the buyers were `Married` and `40%` were `single`

```
[ ]: # statisctical summary of numerical data type columns

df.describe()
```

```
[ ]:               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
```

Insights

1. Age - The age range of customers spans from `18 to 50 year`, with an average age of `29 years`.

2. Usage - Customers intend to utilize the product anywhere from `2 to 7 times per week`, with an average usage frequency of `3 times per week`.

3. Fitness - On average, customers have rated their fitness at `3` on a 5-point scale, reflecting a `moderate level of fitness`.

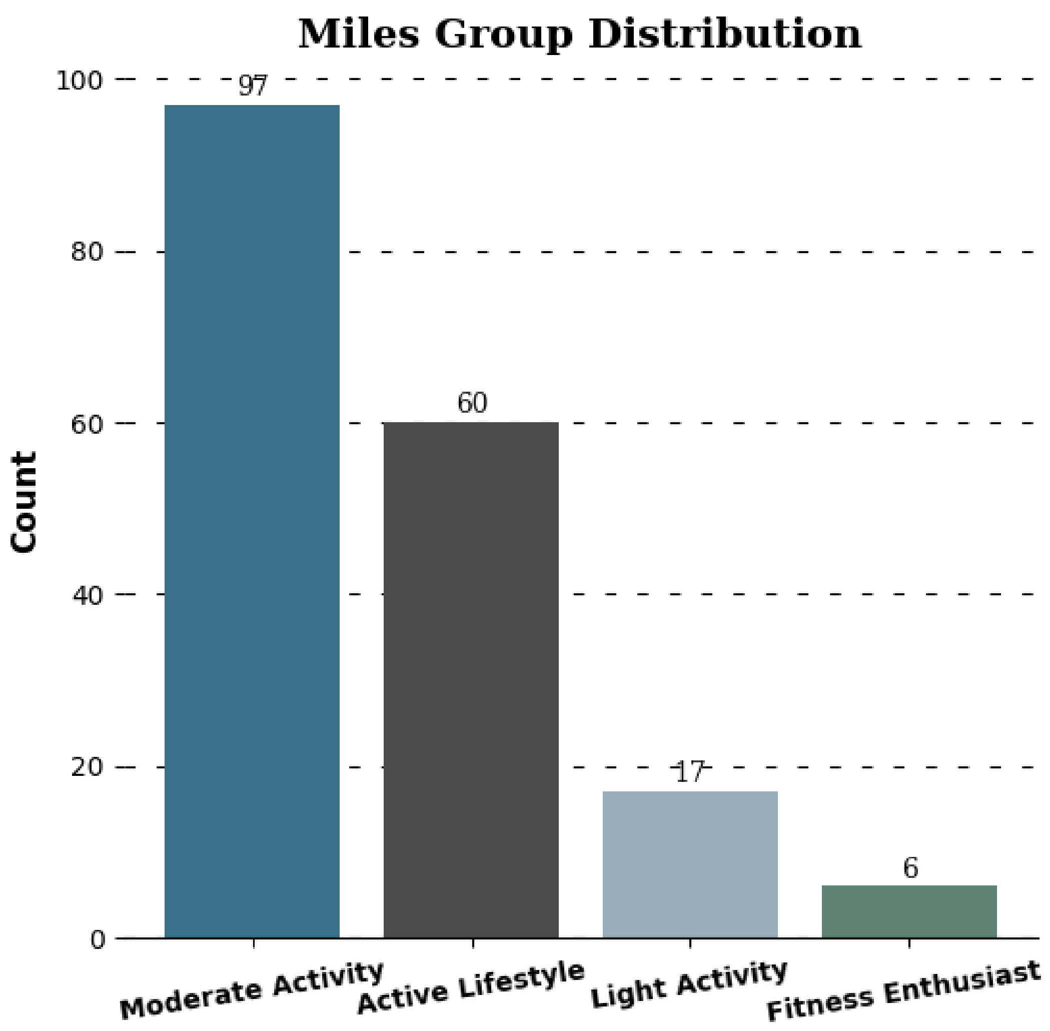
4. Income - The annual income of customers falls within the range of `USD 30,000 to USD 100,000`, with an average income of approximately `USD 54,000`.

##5.2 Comments on the distribution of the variables and relationship between them

** Customer Expected Weekley Milage**

```
[ ]: #setting the plot style

     fig = plt.figure(figsize = (15,10))
     gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.55,0.45])


     #creating Miles group bar chart

     ax2 = fig.add_subplot(gs[0,1])
     temp = df['miles_group'].value_counts()
     color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
     ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)
```

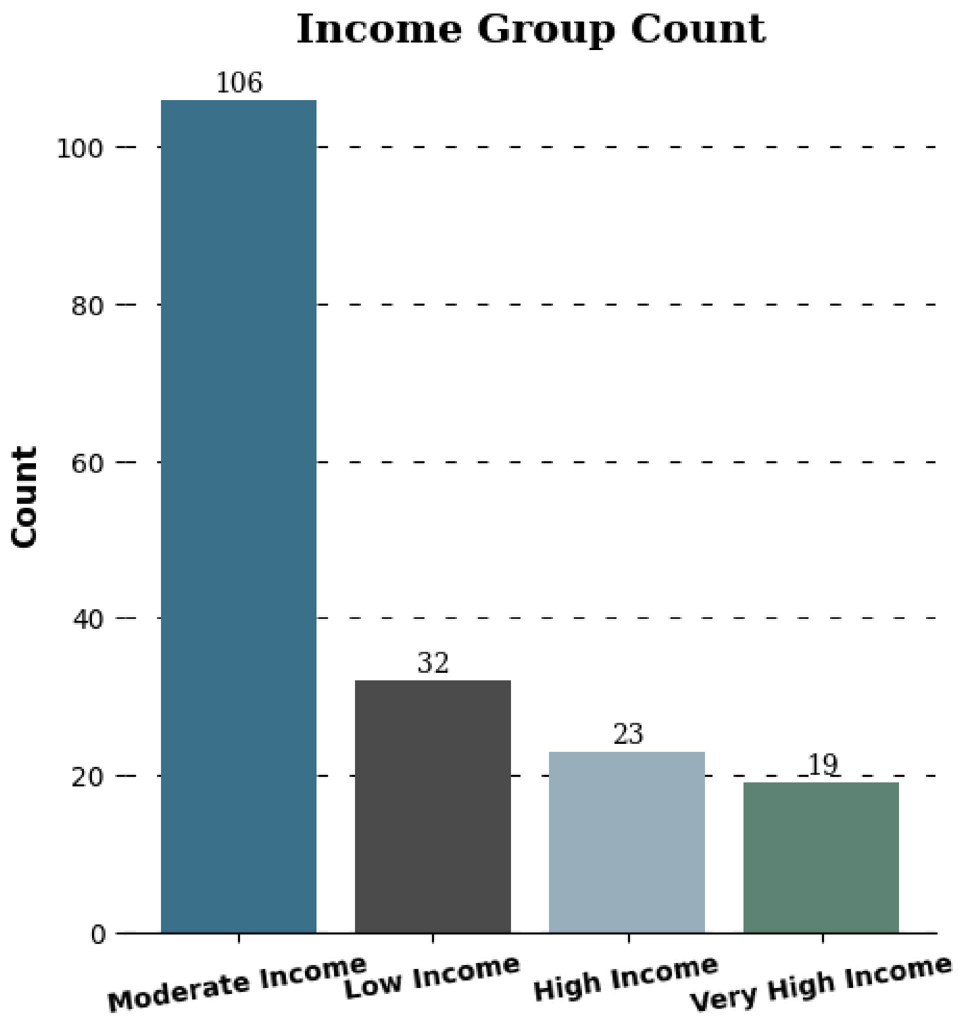
```
#adding the value_counts
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 9)

#setting title for visual
ax2.set_title('Miles Group Distribution',{'font':'serif', 'size':15,'weight':
 ↪'bold'})

plt.show()
```



Income Group Chart

```
[ ]: #setting the plot style

fig = plt.figure(figsize = (15,10))
gs = fig.add_gridspec(2,2,height_ratios=[0.65, 0.35],width_ratios = [0.6,0.4])

                                    #creating Income group bar chart

ax2 = fig.add_subplot(gs[0,1])
temp = df['income_group'].value_counts()
color_map = ["#3A7089", "#4b4b4c",'#99AEBB','#5C8374']
ax2.bar(x=temp.index,height = temp.values,color = color_map,zorder = 2)

#adding the value_counts
```

```
for i in temp.index:
    ax2.text(i,temp[i]+2,temp[i],{'font':'serif','size' : 10},ha = 'center',va␣
 ↪= 'center')

#adding grid lines
ax2.grid(color = 'black',linestyle = '--', axis = 'y', zorder = 0, dashes =␣
 ↪(5,10))

#removing the axis lines
for s in ['top','left','right']:
    ax2.spines[s].set_visible(False)

#adding axis label
ax2.set_ylabel('Count',fontweight = 'bold',fontsize = 12)
ax2.set_xticklabels(temp.index,fontweight = 'bold',rotation = 9)

#setting title for visual
ax2.set_title('Income Group Count',{'font':'serif', 'size':15,'weight':'bold'})
plt.show()
```



##5.3 Comments for each univariate and bivariate plot

Insight On Product Sales - The KP281 treadmill model, positioned as an entry-level product, has the highest number of units sold, trailed by the KP481 (mid-level) and KP781 (advanced) models.

- All three models have nearly equal contributions in terms of generating sales revenue.

Insight On Fitness And Treadmill usage

- Almost `85%` of the customers plan to use the treadmill for `2 to 4 times a week` and only 15% using 5 times and above each week
- 54% of the customers have self-evaluated their fitness at a level 3 on a scale of 1 to 5. Furthermore, a substantial 84% of the total customers have rated themselves at 3 or higher, indicating commendable fitness levels.

Insights On Age

- `85%` of the customers fall in the age range of `18 to 35`. with a median age of `26`, suggesting young people showing more interest in the companies products

Insignt On Customer Education

- `98%` of the customers have education more than 13 years highlighting a strong inclination among well-educated individuals to purchase the products. It's plausible that health awareness driven by education could play a pivotal role in this trend.

** Insight On Customer Income**

- Almost `60%` of the customers fall in the income group of (40k to 60k) dollars suggesting higher inclination of this income group people towards the products.
- Surprisingly `18%` of the customers fall in the income group of (<40) suggesting almost `77%` of the total customers fall in income group of below 60k and only `23%` of them falling in 60k and above income group

3 6- Recommendations

Marketing Campaigns for KP781

- The KP784 model exhibits a significant sales disparity in terms of gender, with only 18% of total sales attributed to female customers. To enhance this metric, it is recommended to implement targeted strategies such as offering special promotions and trials exclusively designed for the female customers.

Affordable Pricing and Payment Plans

- Given the target customer's age, education level, and income, it's important to offer the KP281 and KP481 Treadmill at an affordable price point. Additionally, consider providing flexible payment plans that allow customers to spread the cost over several months. This can make the treadmill more accessible to customers with varying budgets.

User-Friendly App Integration

- Create a user-friendly app that syncs with the treadmill. This app could track users' weekly running mileage, provide real-time feedback on their progress, and offer personalized recommendations for workouts based on their fitness scale and goals.This can enhance the overall treadmill experience and keep users engaged.

[1]:
```
!pip install nbconvert
```

```
Requirement already satisfied: nbconvert in /usr/local/lib/python3.10/dist-
packages (6.5.4)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages
(from nbconvert) (4.9.4)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (4.12.3)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages
(from nbconvert) (6.1.0)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (0.7.1)
```

```
Requirement already satisfied: entrypoints>=0.2.2 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (0.4)
Requirement already satisfied: jinja2>=3.0 in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (3.1.4)
Requirement already satisfied: jupyter-core>=4.7 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (5.7.2)
Requirement already satisfied: jupyterlab-pygments in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (0.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (2.1.5)
Requirement already satisfied: mistune<2,>=0.8.1 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (0.8.4)
Requirement already satisfied: nbclient>=0.5.0 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (0.10.0)
Requirement already satisfied: nbformat>=5.1 in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (5.10.4)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (24.0)
Requirement already satisfied: pandocfilters>=1.4.1 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (1.5.1)
Requirement already satisfied: pygments>=2.4.1 in
/usr/local/lib/python3.10/dist-packages (from nbconvert) (2.16.1)
Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (1.3.0)
Requirement already satisfied: traitlets>=5.0 in /usr/local/lib/python3.10/dist-
packages (from nbconvert) (5.7.1)
Requirement already satisfied: platformdirs>=2.5 in
/usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.7->nbconvert)
(4.2.1)
Requirement already satisfied: jupyter-client>=6.1.12 in
/usr/local/lib/python3.10/dist-packages (from nbclient>=0.5.0->nbconvert)
(6.1.12)
Requirement already satisfied: fastjsonschema>=2.15 in
/usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert) (2.19.1)
Requirement already satisfied: jsonschema>=2.6 in
/usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert) (4.19.2)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-
packages (from beautifulsoup4->nbconvert) (2.5)
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-
packages (from bleach->nbconvert) (1.16.0)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-
packages (from bleach->nbconvert) (0.5.1)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-
packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert) (23.2.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in
/usr/local/lib/python3.10/dist-packages (from
jsonschema>=2.6->nbformat>=5.1->nbconvert) (2023.12.1)
Requirement already satisfied: referencing>=0.28.4 in
```

```
/usr/local/lib/python3.10/dist-packages (from
jsonschema>=2.6->nbformat>=5.1->nbconvert) (0.35.1)
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-
packages (from jsonschema>=2.6->nbformat>=5.1->nbconvert) (0.18.1)
Requirement already satisfied: pyzmq>=13 in /usr/local/lib/python3.10/dist-
packages (from jupyter-client>=6.1.12->nbclient>=0.5.0->nbconvert) (24.0.1)
Requirement already satisfied: python-dateutil>=2.1 in
/usr/local/lib/python3.10/dist-packages (from jupyter-
client>=6.1.12->nbclient>=0.5.0->nbconvert) (2.8.2)
Requirement already satisfied: tornado>=4.1 in /usr/local/lib/python3.10/dist-
packages (from jupyter-client>=6.1.12->nbclient>=0.5.0->nbconvert) (6.3.3)
```

[2]:
```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
```

```
Reading package lists… Done
Building dependency tree… Done
Reading state information… Done
pandoc is already the newest version (2.9.2.1-3ubuntu2).
pandoc set to manually installed.
The following additional packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcommons-logging-java libcommons-parent-
java
  libfontbox-java libfontenc1 libgs9 libgs9-common libidn12 libijs-0.35
libjbig2dec0 libkpathsea6
  libpdfbox-java libptexenc1 libruby3.0 libsynctex2 libteckit0 libtexlua53
libtexluajit2 libwoff1
  libzzip-0-13 lmodern poppler-data preview-latex-style rake ruby ruby-net-
telnet ruby-rubygems
  ruby-webrick ruby-xmlrpc ruby3.0 rubygems-integration t1utils teckit tex-
common tex-gyre
  texlive-base texlive-binaries texlive-fonts-recommended texlive-latex-base
  texlive-latex-recommended texlive-pictures texlive-plain-generic tipa xfonts-
encodings
  xfonts-utils
Suggested packages:
  fonts-noto fonts-freefont-otf | fonts-freefont-ttf libavalon-framework-java
  libcommons-logging-java-doc libexcalibur-logkit-java liblog4j1.2-java poppler-
utils ghostscript
  fonts-japanese-mincho | fonts-ipafont-mincho fonts-japanese-gothic | fonts-
ipafont-gothic
  fonts-arphic-ukai fonts-arphic-uming fonts-nanum ri ruby-dev bundler debhelper
gv
  | postscript-viewer perl-tk xpdf | pdf-viewer xzdec texlive-fonts-recommended-
doc
  texlive-latex-base-doc python3-pygments icc-profiles libfile-which-perl
  libspreadsheet-parseexcel-perl texlive-latex-extra-doc texlive-latex-
```

```
recommended-doc
  texlive-luatex texlive-pstricks dot2tex prerex texlive-pictures-doc vprerex
default-jre-headless
  tipa-doc
The following NEW packages will be installed:
  dvisvgm fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-
texgyre
  fonts-urw-base35 libapache-pom-java libcommons-logging-java libcommons-parent-
java
  libfontbox-java libfontenc1 libgs9 libgs9-common libidn12 libijs-0.35
libjbig2dec0 libkpathsea6
  libpdfbox-java libptexenc1 libruby3.0 libsynctex2 libteckit0 libtexlua53
libtexluajit2 libwoff1
  libzzip-0-13 lmodern poppler-data preview-latex-style rake ruby ruby-net-
telnet ruby-rubygems
  ruby-webrick ruby-xmlrpc ruby3.0 rubygems-integration t1utils teckit tex-
common tex-gyre texlive
  texlive-base texlive-binaries texlive-fonts-recommended texlive-latex-base
texlive-latex-extra
  texlive-latex-recommended texlive-pictures texlive-plain-generic texlive-xetex
tipa
  xfonts-encodings xfonts-utils
0 upgraded, 55 newly installed, 0 to remove and 45 not upgraded.
Need to get 182 MB of archives.
After this operation, 572 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-droid-fallback all
1:6.0.1r16-1.1build1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-lato all 2.0-2.1
[2,696 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 poppler-data all
0.4.11-1 [2,171 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-common all 6.17
[33.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-urw-base35 all
20200910-1 [6,367 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9-common
all 9.55.0~dfsg1-0ubuntu5.6 [751 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libidn12 amd64
1.38-4ubuntu1 [60.0 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/main amd64 libijs-0.35 amd64
0.35-15build2 [16.5 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/main amd64 libjbig2dec0 amd64
0.19-3build2 [64.7 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libgs9 amd64
9.55.0~dfsg1-0ubuntu5.6 [5,031 kB]
Get:11 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libkpathsea6
amd64 2021.20210626.59705-1ubuntu0.2 [60.4 kB]
Get:12 http://archive.ubuntu.com/ubuntu jammy/main amd64 libwoff1 amd64
```

```
1.0.2-1build4 [45.2 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy/universe amd64 dvisvgm amd64
2.13.1-1 [1,221 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-lmodern all
2.004.5-6.1 [4,532 kB]
Get:15 http://archive.ubuntu.com/ubuntu jammy/main amd64 fonts-noto-mono all
20201225-1build1 [397 kB]
Get:16 http://archive.ubuntu.com/ubuntu jammy/universe amd64 fonts-texgyre all
20180621-3.1 [10.2 MB]
Get:17 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libapache-pom-java
all 18-1 [4,720 B]
Get:18 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-parent-
java all 43-1 [10.8 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcommons-logging-
java all 1.2-2 [60.3 kB]
Get:20 http://archive.ubuntu.com/ubuntu jammy/main amd64 libfontenc1 amd64
1:1.1.4-1build3 [14.7 kB]
Get:21 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libptexenc1
amd64 2021.20210626.59705-1ubuntu0.2 [39.1 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy/main amd64 rubygems-integration
all 1.18 [5,336 B]
Get:23 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby3.0 amd64
3.0.2-7ubuntu2.5 [50.1 kB]
Get:24 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-rubygems all
3.3.5-2 [228 kB]
Get:25 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby amd64 1:3.0~exp1
[5,100 B]
Get:26 http://archive.ubuntu.com/ubuntu jammy/main amd64 rake all 13.0.6-2 [61.7
kB]
Get:27 http://archive.ubuntu.com/ubuntu jammy/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:28 http://archive.ubuntu.com/ubuntu jammy/universe amd64 ruby-webrick all
1.7.0-3 [51.8 kB]
Get:29 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 ruby-xmlrpc all
0.3.2-1ubuntu0.1 [24.9 kB]
Get:30 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libruby3.0
amd64 3.0.2-7ubuntu2.5 [5,113 kB]
Get:31 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libsynctex2
amd64 2021.20210626.59705-1ubuntu0.2 [55.6 kB]
Get:32 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libteckit0 amd64
2.5.11+ds1-1 [421 kB]
Get:33 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexlua53
amd64 2021.20210626.59705-1ubuntu0.2 [120 kB]
Get:34 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtexluajit2
amd64 2021.20210626.59705-1ubuntu0.2 [267 kB]
Get:35 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libzzip-0-13 amd64
0.13.72+dfsg.1-1.1 [27.0 kB]
Get:36 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-encodings all
```

```
1:1.0.5-0ubuntu2 [578 kB]
Get:37 http://archive.ubuntu.com/ubuntu jammy/main amd64 xfonts-utils amd64
1:7.7+6build2 [94.6 kB]
Get:38 http://archive.ubuntu.com/ubuntu jammy/universe amd64 lmodern all
2.004.5-6.1 [9,471 kB]
Get:39 http://archive.ubuntu.com/ubuntu jammy/universe amd64 preview-latex-style
all 12.2-1ubuntu1 [185 kB]
Get:40 http://archive.ubuntu.com/ubuntu jammy/main amd64 t1utils amd64
1.41-4build2 [61.3 kB]
Get:41 http://archive.ubuntu.com/ubuntu jammy/universe amd64 teckit amd64
2.5.11+ds1-1 [699 kB]
Get:42 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tex-gyre all
20180621-3.1 [6,209 kB]
Get:43 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 texlive-
binaries amd64 2021.20210626.59705-1ubuntu0.2 [9,860 kB]
Get:44 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-base all
2021.20220204-1 [21.0 MB]
Get:45 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-fonts-
recommended all 2021.20220204-1 [4,972 kB]
Get:46 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-base
all 2021.20220204-1 [1,128 kB]
Get:47 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-
recommended all 2021.20220204-1 [14.4 MB]
Get:48 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive all
2021.20220204-1 [14.3 kB]
Get:49 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libfontbox-java all
1:1.8.16-2 [207 kB]
Get:50 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libpdfbox-java all
1:1.8.16-2 [5,199 kB]
Get:51 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-pictures
all 2021.20220204-1 [8,720 kB]
Get:52 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-latex-extra
all 2021.20220204-1 [13.9 MB]
Get:53 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-plain-
generic all 2021.20220204-1 [27.5 MB]
Get:54 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tipa all 2:1.3-21
[2,967 kB]
Get:55 http://archive.ubuntu.com/ubuntu jammy/universe amd64 texlive-xetex all
2021.20220204-1 [12.4 MB]
Fetched 182 MB in 5s (34.1 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 121918 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1build1_all.deb
…
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Selecting previously unselected package fonts-lato.
```

```
Preparing to unpack …/01-fonts-lato_2.0-2.1_all.deb …
Unpacking fonts-lato (2.0-2.1) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.11-1_all.deb …
Unpacking poppler-data (0.4.11-1) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.17_all.deb …
Unpacking tex-common (6.17) …
Selecting previously unselected package fonts-urw-base35.
Preparing to unpack …/04-fonts-urw-base35_20200910-1_all.deb …
Unpacking fonts-urw-base35 (20200910-1) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/05-libgs9-common_9.55.0~dfsg1-0ubuntu5.6_all.deb …
Unpacking libgs9-common (9.55.0~dfsg1-0ubuntu5.6) …
Selecting previously unselected package libidn12:amd64.
Preparing to unpack …/06-libidn12_1.38-4ubuntu1_amd64.deb …
Unpacking libidn12:amd64 (1.38-4ubuntu1) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/07-libijs-0.35_0.35-15build2_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-15build2) …
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/08-libjbig2dec0_0.19-3build2_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.19-3build2) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/09-libgs9_9.55.0~dfsg1-0ubuntu5.6_amd64.deb …
Unpacking libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.6) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/10-libkpathsea6_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libwoff1:amd64.
Preparing to unpack …/11-libwoff1_1.0.2-1build4_amd64.deb …
Unpacking libwoff1:amd64 (1.0.2-1build4) …
Selecting previously unselected package dvisvgm.
Preparing to unpack …/12-dvisvgm_2.13.1-1_amd64.deb …
Unpacking dvisvgm (2.13.1-1) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/13-fonts-lmodern_2.004.5-6.1_all.deb …
Unpacking fonts-lmodern (2.004.5-6.1) …
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/14-fonts-noto-mono_20201225-1build1_all.deb …
Unpacking fonts-noto-mono (20201225-1build1) …
Selecting previously unselected package fonts-texgyre.
Preparing to unpack …/15-fonts-texgyre_20180621-3.1_all.deb …
Unpacking fonts-texgyre (20180621-3.1) …
Selecting previously unselected package libapache-pom-java.
Preparing to unpack …/16-libapache-pom-java_18-1_all.deb …
Unpacking libapache-pom-java (18-1) …
```

```
Selecting previously unselected package libcommons-parent-java.
Preparing to unpack …/17-libcommons-parent-java_43-1_all.deb …
Unpacking libcommons-parent-java (43-1) …
Selecting previously unselected package libcommons-logging-java.
Preparing to unpack …/18-libcommons-logging-java_1.2-2_all.deb …
Unpacking libcommons-logging-java (1.2-2) …
Selecting previously unselected package libfontenc1:amd64.
Preparing to unpack …/19-libfontenc1_1%3a1.1.4-1build3_amd64.deb …
Unpacking libfontenc1:amd64 (1:1.1.4-1build3) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/20-libptexenc1_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/21-rubygems-integration_1.18_all.deb …
Unpacking rubygems-integration (1.18) …
Selecting previously unselected package ruby3.0.
Preparing to unpack …/22-ruby3.0_3.0.2-7ubuntu2.5_amd64.deb …
Unpacking ruby3.0 (3.0.2-7ubuntu2.5) …
Selecting previously unselected package ruby-rubygems.
Preparing to unpack …/23-ruby-rubygems_3.3.5-2_all.deb …
Unpacking ruby-rubygems (3.3.5-2) …
Selecting previously unselected package ruby.
Preparing to unpack …/24-ruby_1%3a3.0~exp1_amd64.deb …
Unpacking ruby (1:3.0~exp1) …
Selecting previously unselected package rake.
Preparing to unpack …/25-rake_13.0.6-2_all.deb …
Unpacking rake (13.0.6-2) …
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/26-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
Selecting previously unselected package ruby-webrick.
Preparing to unpack …/27-ruby-webrick_1.7.0-3_all.deb …
Unpacking ruby-webrick (1.7.0-3) …
Selecting previously unselected package ruby-xmlrpc.
Preparing to unpack …/28-ruby-xmlrpc_0.3.2-1ubuntu0.1_all.deb …
Unpacking ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Selecting previously unselected package libruby3.0:amd64.
Preparing to unpack …/29-libruby3.0_3.0.2-7ubuntu2.5_amd64.deb …
Unpacking libruby3.0:amd64 (3.0.2-7ubuntu2.5) …
Selecting previously unselected package libsynctex2:amd64.
Preparing to unpack …/30-libsynctex2_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libteckit0:amd64.
Preparing to unpack …/31-libteckit0_2.5.11+ds1-1_amd64.deb …
Unpacking libteckit0:amd64 (2.5.11+ds1-1) …
Selecting previously unselected package libtexlua53:amd64.
```

```
Preparing to unpack …/32-libtexlua53_2021.20210626.59705-1ubuntu0.2_amd64.deb
…
Unpacking libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/33-libtexluajit2_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/34-libzzip-0-13_0.13.72+dfsg.1-1.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Selecting previously unselected package xfonts-encodings.
Preparing to unpack …/35-xfonts-encodings_1%3a1.0.5-0ubuntu2_all.deb …
Unpacking xfonts-encodings (1:1.0.5-0ubuntu2) …
Selecting previously unselected package xfonts-utils.
Preparing to unpack …/36-xfonts-utils_1%3a7.7+6build2_amd64.deb …
Unpacking xfonts-utils (1:7.7+6build2) …
Selecting previously unselected package lmodern.
Preparing to unpack …/37-lmodern_2.004.5-6.1_all.deb …
Unpacking lmodern (2.004.5-6.1) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/38-preview-latex-style_12.2-1ubuntu1_all.deb …
Unpacking preview-latex-style (12.2-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/39-t1utils_1.41-4build2_amd64.deb …
Unpacking t1utils (1.41-4build2) …
Selecting previously unselected package teckit.
Preparing to unpack …/40-teckit_2.5.11+ds1-1_amd64.deb …
Unpacking teckit (2.5.11+ds1-1) …
Selecting previously unselected package tex-gyre.
Preparing to unpack …/41-tex-gyre_20180621-3.1_all.deb …
Unpacking tex-gyre (20180621-3.1) …
Selecting previously unselected package texlive-binaries.
Preparing to unpack …/42-texlive-
binaries_2021.20210626.59705-1ubuntu0.2_amd64.deb …
Unpacking texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/43-texlive-base_2021.20220204-1_all.deb …
Unpacking texlive-base (2021.20220204-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/44-texlive-fonts-recommended_2021.20220204-1_all.deb …
Unpacking texlive-fonts-recommended (2021.20220204-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/45-texlive-latex-base_2021.20220204-1_all.deb …
Unpacking texlive-latex-base (2021.20220204-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/46-texlive-latex-recommended_2021.20220204-1_all.deb …
Unpacking texlive-latex-recommended (2021.20220204-1) …
Selecting previously unselected package texlive.
```

```
Preparing to unpack …/47-texlive_2021.20220204-1_all.deb …
Unpacking texlive (2021.20220204-1) …
Selecting previously unselected package libfontbox-java.
Preparing to unpack …/48-libfontbox-java_1%3a1.8.16-2_all.deb …
Unpacking libfontbox-java (1:1.8.16-2) …
Selecting previously unselected package libpdfbox-java.
Preparing to unpack …/49-libpdfbox-java_1%3a1.8.16-2_all.deb …
Unpacking libpdfbox-java (1:1.8.16-2) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/50-texlive-pictures_2021.20220204-1_all.deb …
Unpacking texlive-pictures (2021.20220204-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/51-texlive-latex-extra_2021.20220204-1_all.deb …
Unpacking texlive-latex-extra (2021.20220204-1) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/52-texlive-plain-generic_2021.20220204-1_all.deb …
Unpacking texlive-plain-generic (2021.20220204-1) …
Selecting previously unselected package tipa.
Preparing to unpack …/53-tipa_2%3a1.3-21_all.deb …
Unpacking tipa (2:1.3-21) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/54-texlive-xetex_2021.20220204-1_all.deb …
Unpacking texlive-xetex (2021.20220204-1) …
Setting up fonts-lato (2.0-2.1) …
Setting up fonts-noto-mono (20201225-1build1) …
Setting up libwoff1:amd64 (1.0.2-1build4) …
Setting up libtexlua53:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libijs-0.35:amd64 (0.35-15build2) …
Setting up libtexluajit2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libfontbox-java (1:1.8.16-2) …
Setting up rubygems-integration (1.18) …
Setting up libzzip-0-13:amd64 (0.13.72+dfsg.1-1.1) …
Setting up fonts-urw-base35 (20200910-1) …
Setting up poppler-data (0.4.11-1) …
Setting up tex-common (6.17) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up libfontenc1:amd64 (1:1.1.4-1build3) …
Setting up libjbig2dec0:amd64 (0.19-3build2) …
Setting up libteckit0:amd64 (2.5.11+ds1-1) …
Setting up libapache-pom-java (18-1) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up xfonts-encodings (1:1.0.5-0ubuntu2) …
Setting up t1utils (1.41-4build2) …
Setting up libidn12:amd64 (1.38-4ubuntu1) …
Setting up fonts-texgyre (20180621-3.1) …
Setting up libkpathsea6:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up ruby-webrick (1.7.0-3) …
Setting up fonts-lmodern (2.004.5-6.1) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1build1) …
Setting up ruby-xmlrpc (0.3.2-1ubuntu0.1) …
Setting up libsynctex2:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up libgs9-common (9.55.0~dfsg1-0ubuntu5.6) …
Setting up teckit (2.5.11+ds1-1) …
Setting up libpdfbox-java (1:1.8.16-2) …
Setting up libgs9:amd64 (9.55.0~dfsg1-0ubuntu5.6) …
Setting up preview-latex-style (12.2-1ubuntu1) …
Setting up libcommons-parent-java (43-1) …
Setting up dvisvgm (2.13.1-1) …
Setting up libcommons-logging-java (1.2-2) …
Setting up xfonts-utils (1:7.7+6build2) …
Setting up libptexenc1:amd64 (2021.20210626.59705-1ubuntu0.2) …
Setting up texlive-binaries (2021.20210626.59705-1ubuntu0.2) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up lmodern (2.004.5-6.1) …
Setting up texlive-base (2021.20220204-1) …
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
/usr/bin/ucfr
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4: /var/lib/texmf/tex/generic/tex-
ini-files/pdftexconfig.tex
Setting up tex-gyre (20180621-3.1) …
Setting up texlive-plain-generic (2021.20220204-1) …
Setting up texlive-latex-base (2021.20220204-1) …
Setting up texlive-latex-recommended (2021.20220204-1) …
Setting up texlive-pictures (2021.20220204-1) …
Setting up texlive-fonts-recommended (2021.20220204-1) …
Setting up tipa (2:1.3-21) …
Setting up texlive (2021.20220204-1) …
Setting up texlive-latex-extra (2021.20220204-1) …
Setting up texlive-xetex (2021.20220204-1) …
Setting up rake (13.0.6-2) …
Setting up libruby3.0:amd64 (3.0.2-7ubuntu2.5) …
Setting up ruby3.0 (3.0.2-7ubuntu2.5) …
Setting up ruby (1:3.0~exp1) …
Setting up ruby-rubygems (3.3.5-2) …
Processing triggers for man-db (2.10.2-1) …
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) …
Processing triggers for libc-bin (2.35-0ubuntu3.4) …
/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic
link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

Processing triggers for tex-common (6.17) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time… done.
```

[13]:
```
!jupyter nbconvert --to pdf AER.ipynb
```

```
[NbConvertApp] WARNING | pattern 'AER.ipynb' matched no files
This application is used to convert notebook files (*.ipynb)
        to various other formats.

        WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.

Options
=======
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
To see all configurable class-options for some <cmd>, use:
    <cmd> --help-all

--debug
    set log level to logging.DEBUG (maximize logging output)
    Equivalent to: [--Application.log_level=10]
--show-config
    Show the application's configuration (human-readable format)
    Equivalent to: [--Application.show_config=True]
--show-config-json
    Show the application's configuration (json format)
    Equivalent to: [--Application.show_config_json=True]
```

```
--generate-config
    generate default config file
    Equivalent to: [--JupyterApp.generate_config=True]
-y
    Answer yes to any questions instead of prompting.
    Equivalent to: [--JupyterApp.answer_yes=True]
--execute
    Execute the notebook prior to export.
    Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
    Continue notebook execution even if one of the cells throws an error and
include the error message in the cell output (the default behaviour is to abort
conversion). This flag is only relevant if '--execute' was specified, too.
    Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
    read a single notebook file from stdin. Write the resulting notebook with
default basename 'notebook.*'
    Equivalent to: [--NbConvertApp.from_stdin=True]
--stdout
    Write notebook output to stdout instead of files.
    Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
    Run nbconvert in place, overwriting the existing notebook (only
            relevant when converting to notebook format)
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=]
--clear-output
    Clear output of current file and save in place,
            overwriting the existing notebook.
    Equivalent to: [--NbConvertApp.use_output_suffix=False
--NbConvertApp.export_format=notebook --FilesWriter.build_directory=
--ClearOutputPreprocessor.enabled=True]
--no-prompt
    Exclude input and output prompts from converted document.
    Equivalent to: [--TemplateExporter.exclude_input_prompt=True
--TemplateExporter.exclude_output_prompt=True]
--no-input
    Exclude input cells and output prompts from converted document.
            This mode is ideal for generating code-free reports.
    Equivalent to: [--TemplateExporter.exclude_output_prompt=True
--TemplateExporter.exclude_input=True
--TemplateExporter.exclude_input_prompt=True]
--allow-chromium-download
    Whether to allow downloading chromium if no suitable version is found on the
system.
    Equivalent to: [--WebPDFExporter.allow_chromium_download=True]
--disable-chromium-sandbox
    Disable chromium security sandbox when converting to PDF..
    Equivalent to: [--WebPDFExporter.disable_sandbox=True]
--show-input
    Shows code input. This flag is only useful for dejavu users.
    Equivalent to: [--TemplateExporter.exclude_input=False]
--embed-images
    Embed the images as base64 dataurls in the output. This flag is only useful
for the HTML/WebPDF/Slides exports.
    Equivalent to: [--HTMLExporter.embed_images=True]
--sanitize-html
    Whether the HTML in Markdown cells and cell outputs should be sanitized..
    Equivalent to: [--HTMLExporter.sanitize_html=True]
--log-level=<Enum>
    Set the log level by value or name.
    Choices: any of [0, 10, 20, 30, 40, 50, 'DEBUG', 'INFO', 'WARN', 'ERROR',
'CRITICAL']
    Default: 30
    Equivalent to: [--Application.log_level]
--config=<Unicode>
    Full path of a config file.
    Default: ''
    Equivalent to: [--JupyterApp.config_file]
--to=<Unicode>
    The export format to be used, either one of the built-in formats
            ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook',
'pdf', 'python', 'rst', 'script', 'slides', 'webpdf']
            or a dotted object name that represents the import path for an
            ``Exporter`` class
    Default: ''
    Equivalent to: [--NbConvertApp.export_format]
--template=<Unicode>
    Name of the template to use
    Default: ''
    Equivalent to: [--TemplateExporter.template_name]
--template-file=<Unicode>
    Name of the template file to use
    Default: None
    Equivalent to: [--TemplateExporter.template_file]
--theme=<Unicode>
    Template specific theme(e.g. the name of a JupyterLab CSS theme distributed
    as prebuilt extension for the lab template)
    Default: 'light'
    Equivalent to: [--HTMLExporter.theme]
--sanitize_html=<Bool>
    Whether the HTML in Markdown cells and cell outputs should be sanitized.This
    should be set to True by nbviewer or similar tools.
    Default: False
    Equivalent to: [--HTMLExporter.sanitize_html]
--writer=<DottedObjectName>
```

```
    Writer class used to write the
                                        results of the conversion
    Default: 'FilesWriter'
    Equivalent to: [--NbConvertApp.writer_class]
--post=<DottedOrNone>
    PostProcessor class used to write the
                                        results of the conversion
    Default: ''
    Equivalent to: [--NbConvertApp.postprocessor_class]
--output=<Unicode>
    overwrite base name use for output files.
                can only be used when converting one notebook at a time.
    Default: ''
    Equivalent to: [--NbConvertApp.output_base]
--output-dir=<Unicode>
    Directory to write output(s) to. Defaults
                                  to output to the directory of each notebook.
To recover
                                  previous default behaviour (outputting to the
current
                                  working directory) use . as the flag value.
    Default: ''
    Equivalent to: [--FilesWriter.build_directory]
--reveal-prefix=<Unicode>
    The URL prefix for reveal.js (version 3.x).
            This defaults to the reveal CDN, but can be any url pointing to a
copy
            of reveal.js.
            For speaker notes to work, this must be a relative path to a local
            copy of reveal.js: e.g., "reveal.js".
            If a relative path is given, it must be a subdirectory of the
            current directory (from which the server is run).
            See the usage documentation
            (https://nbconvert.readthedocs.io/en/latest/usage.html#reveal-js-
html-slideshow)
            for more details.
    Default: ''
    Equivalent to: [--SlidesExporter.reveal_url_prefix]
--nbformat=<Enum>
    The nbformat version to write.
            Use this to downgrade notebooks.
    Choices: any of [1, 2, 3, 4]
    Default: 4
    Equivalent to: [--NotebookExporter.nbformat_version]

Examples
--------
```

```
    The simplest way to use nbconvert is

            > jupyter nbconvert mynotebook.ipynb --to html

            Options include ['asciidoc', 'custom', 'html', 'latex', 'markdown',
'notebook', 'pdf', 'python', 'rst', 'script', 'slides', 'webpdf'].

            > jupyter nbconvert --to latex mynotebook.ipynb

            Both HTML and LaTeX support multiple output templates. LaTeX
includes
            'base', 'article' and 'report'.  HTML includes 'basic', 'lab' and
            'classic'. You can specify the flavor of the format used.

            > jupyter nbconvert --to html --template lab mynotebook.ipynb

            You can also pipe the output to stdout, rather than a file

            > jupyter nbconvert mynotebook.ipynb --stdout

            PDF is generated via latex

            > jupyter nbconvert mynotebook.ipynb --to pdf

            You can get (and serve) a Reveal.js-powered slideshow

            > jupyter nbconvert myslides.ipynb --to slides --post serve

            Multiple notebooks can be given at the command line in a couple of
            different ways:

            > jupyter nbconvert notebook*.ipynb
            > jupyter nbconvert notebook1.ipynb notebook2.ipynb

            or you can specify the notebooks list in a config file, containing::

                c.NbConvertApp.notebooks = ["my_notebook.ipynb"]

            > jupyter nbconvert --config mycfg.py

To see all available configurables, use `--help-all`.
```

[4]:
```python
from google.colab import drive
drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

[ ]:

```
!jupyter nbconvert --to pdf --output 1.pdf /content/drive/MyDrive/Aerofit/1.
↪ipynb
```