Aerofit Business Case using Aerofit Dataset

Step 1: Importing the Libraries and Data

Importing libraries for our business case

libraries

1. Pandas
2. Numpy
3. Pyplot
4. Seaborn

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

Insights into data:

Pandas and Numpy are used for data cleaning and analysis Pyplot and Seaborn are used for the visualization and graphical analysis of the data

Importing the data so we can do the operations

In [3]:
```
df=pd.read_csv('/content/drive/MyDrive/Colab Notebooks/aerofit_treadmill.txt')
df.head()
```

Out[3]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
0	KP281	18	Male	14	Single	3	4	29562	1
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	

Insights into data:

We have loaded the data into the colab for analyse the data.

Basic Analysis of the data

Length of data or number of rows

```
In [4]: len(df)
```

```
Out[4]: 180
```

Insights into data:

Len function gives us the number of rows we are having in the data which is 180 rows of data.

Recommendations from the data:

We have to analyse the given data to check how many null values are present what is the shape of the data and how many unique values are present in the data.

Step 2: Understanding the imported Data

Shape of dataset:

```
In [5]: df.shape
```

```
Out[5]: (180, 9)
```

Insights into data:

df.shape gives us the shape of the data which is 180 rows and 9 columns of data.

Checking Info of the Data

In [6]:
```
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
```

Insights into data:

We can see the column names and type of the data is object and int64 for column.

Checking for null values

In [7]:
```
df.isnull().sum()
```

Out[7]:

	0
Product	0
Age	0
Gender	0
Education	0
MaritalStatus	0
Usage	0
Fitness	0
Income	0
Miles	0

dtype: int64

Insights into data:

We do not have any missing values in the data. Which is a good thing for the analysis of the data.

Number of unique values per column in our data

```
In [8]: df.nunique()
```

Out[8]:

	0
Product	3
Age	32
Gender	2
Education	8
MaritalStatus	2
Usage	6
Fitness	5
Income	62
Miles	37

dtype: int64

Insights into data:

From the above observation, we can conclude that only Inncome, Miles and Age can be considered as Continuous, the rest of the columns though integers should be considered as categories.

Step 3: Basic analysis of the data

Categorical columns value counts checking

Product columns value counts checking

In [9]:
```
df['Product'].value_counts()
```

Out[9]:

Product	count
KP281	80
KP481	60
KP781	40

dtype: int64

Insights into data:

There are 3 products that are to be analysed KP281, KP481 and KP781.

Gender columns value counts checking

In [10]:
```
df['Gender'].value_counts()
```

Out[10]:

Gender	count
Male	104
Female	76

dtype: int64

Insights into data:

We have 2 Genders that are to be analysed Male and Female.

Education columns value counts checking

```
In [11]: df['Education'].value_counts()
```

Out[11]:

Education	count
16	85
14	55
18	23
15	5
13	5
12	3
21	3
20	1

dtype: int64

Insights into data:

There are 8 Educations that are to be analysed. All the values are in numerical values of how long the person has studied for.

MaritalStatus columns value counts checking

```
In [12]: df['MaritalStatus'].value_counts()
```

Out[12]:

MaritalStatus	count
Partnered	107
Single	73

dtype: int64

Insights into data:

We have 2 MaritalStatus that are to be analysed Partnered and Single.

Usage columns value counts checking

```
In [13]: df['Usage'].value_counts()
```

Out[13]:

Usage	count
3	69
4	52
2	33
5	17
6	7
7	2

dtype: int64

Insights into data:

There are 6 Usage that are to be analysed. The average number of times the customer plans to use the treadmill each week is represended using number values.

Fitness columns value counts checking

```
In [14]: df['Fitness'].value_counts()
```

Out[14]:

Fitness	count
3	97
5	31
2	26
4	24
1	2

dtype: int64

Insights into data:

There are 5 Fitness that are to be analysed. Self-rated fitness on a 1-to-5 scale, where 1 is the poor shape and 5 is the excellent shape.

Step 4: Indepth analysis of the data and Non-Graphical analysis

Income Analysis

In [15]:
```
df['Income'].describe()
```

Out[15]:

	Income
count	180.000000
mean	53719.577778
std	16506.684226
min	29562.000000
25%	44058.750000
50%	50596.500000
75%	58668.000000
max	104581.000000

dtype: float64

Insights into data:

The average income of Aerofit buyers is around 53,720, with a fairly wide spread (std ~ 16,500). The income distribution shows a mix of middle-income and high-income customers. The top 25% earn above 58,668, indicating a strong premium segment. The high max income (104k) suggests Aerofit attracts some affluent customers as well.

Recommendations from the data:

Introduce tiered marketing campaigns to target both mid-income and high-income customers. Promote financing or EMI options for lower-income buyers. Highlight premium features of KP781 to high-earning customers.

Income Analysis with Product

In [16]:
```python
df.groupby('Product')['Income'].describe()
```

Out[16]:

Product	count	mean	std	min	25%	50%	75%	
KP281	80.0	46418.025	9075.783190	29562.0	38658.00	46617.0	53439.0	6
KP481	60.0	48973.650	8653.989388	31836.0	44911.50	49459.5	53439.0	6
KP781	40.0	75441.575	18505.836720	48556.0	58204.75	76568.5	90886.0	10

Insights into data:

KP281 buyers have the lowest average income (~$46k), matching its entry-level pricing. KP481 buyers occupy the mid-income bracket (~$49k), showing balanced affordability. KP781 attracts high-income customers (~$75k), reinforcing it as a premium treadmill. Income variance for KP781 is high, showing diverse but mostly upper-tier buyers.

Recommendations from the data:

Position KP281 as the best value option for students, young professionals, and beginners. Promote KP481 to families and mid-income groups through bundle offers. Market KP781 as a luxury fitness product to affluent neighborhoods and corporate professionals.

Miles Analysis with Product

In [17]:
```
df.groupby('Product')['Miles'].describe()
```

Out[17]:

Product	count	mean	std	min	25%	50%	75%	max
KP281	80.0	82.787500	28.874102	38.0	66.0	85.0	94.0	188.0
KP481	60.0	87.933333	33.263135	21.0	64.0	85.0	106.0	212.0
KP781	40.0	166.900000	60.066544	80.0	120.0	160.0	200.0	360.0

Insights into data:

KP281 users run ~82 miles/week, showing moderate usage. KP481 users run ~88 miles/week, slightly higher and more consistent. KP781 users run ~167 miles/week, indicating very heavy usage and advanced training. KP781 has the highest variability, showing ultrarunners and intense users.

Recommendations from the data:

Highlight KP281 as ideal for light to moderate runners. Market KP481 as a good fit for regular fitness users wanting durability. Emphasize KP781 as a “professional training treadmill” for athletes and marathoners. Promote advanced monitoring features for KP781 to match high-mileage users.

Miles Analysis with Gender

In [18]:
```
df.groupby('Gender')['Miles'].describe()
```

Out[18]:

Gender	count	mean	std	min	25%	50%	75%	max
Female	76.0	90.013158	44.782882	21.0	66.0	85.0	100.0	280.0
Male	104.0	112.826923	54.702451	42.0	85.0	100.0	141.0	360.0

Insights into data:

Male customers run significantly more miles on average (113 miles/week) than females (90 miles/week). Males also show wider variability, including extremely high-mileage runners (up to 360 miles). This suggests men tend to use treadmills for more intense or high-volume training. Females show a more consistent mid-range running pattern.

Recommendations from the data:

Target male customers with campaigns emphasizing performance, durability, and speed training. Promote comfort, safety, and balanced workout benefits for female customers. Create gender-tailored content — e.g., strength-building for males, cardio/health focus for females. Use male-focused ads for KP781 due to their high-mileage behavior.

Miles Analysis with MaritalStatus

In [19]:
```
df.groupby('MaritalStatus')['Miles'].describe()
```

Out[19]:

	count	mean	std	min	25%	50%	75%	max
MaritalStatus								
Partnered	107.0	104.289720	59.230762	38.0	66.0	85.0	120.0	360.0
Single	73.0	101.589041	38.959678	21.0	75.0	95.0	113.0	200.0

Insights into data:

Partnered customers run slightly more miles on average (104 miles/week) than single customers (102 miles/week). Partnered users also have higher maximum usage (up to 360 miles), indicating some very heavy runners. Singles show lower variability, indicating more predictable mid-level usage. Both groups are active, but partnered buyers include the most extreme usage cases.

Recommendations from the data:

Market KP781 to partnered customers, especially couples doing fitness together. Offer family or partner discounts on KP481 to encourage household purchases. Promote KP281 to single customers as a compact and affordable personal fitness solution. Create “couple fitness challenge” campaigns to boost premium product sales.

Income Analysis with Age

```
In [20]: df['Age'].value_counts().sort_index(ascending=True)
```

Out[20]:

Age	count
18	1
19	4
20	5
21	7
22	7
23	18
24	12
25	25
26	12
27	7
28	9
29	6
30	7
31	6
32	4
33	8
34	6
35	8
36	1
37	2
38	7
39	1
40	5
41	1
42	1
43	1
44	1
45	2
46	1
47	2

	count
Age	
48	2
50	1

dtype: int64

In [21]:
```
df['Age_Group']=pd.cut(df['Age'], bins=[17,28,38,50] )
df.head()
```

Out[21]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mil
0	KP281	18	Male	14	Single	3	4	29562	1
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	

In [22]:
```
df.groupby('Age_Group')['Income'].describe()
```

```
/tmp/ipython-input-796820106.py:1: FutureWarning: The default of observed=False
is deprecated and will be changed to True in a future version of pandas. Pass o
bserved=False to retain current behavior or observed=True to adopt the future d
efault and silence this warning.
  df.groupby('Age_Group')['Income'].describe()
```

Out[22]:

	count	mean	std	min	25%	50%	75
Age_Group							
(17, 28]	107.0	47942.925234	13480.718754	29562.0	38658.0	45480.0	52302.
(28, 38]	55.0	60189.054545	16803.758609	37521.0	50028.0	53439.0	63672.
(38, 50]	18.0	68290.722222	16390.168781	53439.0	57987.0	60829.5	78764.

Insights into data:

Younger customers (17–28) have the lowest average income (47k), indicating early-career buyers. Middle age group (28–38) shows a higher average income (60k), aligning with stable mid-career earners. Older customers (38–50) have the highest incomes (68k), showing strong purchasing power. Income variability increases with age, suggesting diverse economic backgrounds among older users.

Recommendations from the data:

Promote KP281 to younger users with lower budgets and entry-level needs. Target mid-career professionals (28–38) with KP481 through value-for-money messaging. Position KP781 for older, high-income customers who seek premium, durable machines. Use age-segmented digital ads to improve conversion and ad efficiency.

Miles Analysis with Age

```
In [23]: df.groupby('Age_Group')['Miles'].describe()
```

```
/tmp/ipython-input-1624004416.py:1: FutureWarning: The default of observed=Fals
e is deprecated and will be changed to True in a future version of pandas. Pass
observed=False to retain current behavior or observed=True to adopt the future
default and silence this warning.
  df.groupby('Age_Group')['Miles'].describe()
```

Out[23]:

	count	mean	std	min	25%	50%	75%	max
Age_Group								
(17, 28]	107.0	100.046729	43.135238	38.0	66.0	94.0	113.0	240.0
(28, 38]	55.0	108.909091	66.418503	21.0	74.5	85.0	120.0	360.0
(38, 50]	18.0	104.444444	50.381162	42.0	66.0	89.5	129.0	200.0

Insights into data:

Younger users (17–28) run around 100 miles/week, showing moderate fitness engagement. Middle-aged users (28–38) run slightly more (~108 miles/week), with higher variance and several heavy runners. Older customers (38–50) maintain around 104 miles/week, indicating consistent but moderate usage. The highest mileage extremes come from the 28–38 age group.

Recommendations from the data:

Market KP281 to young users as a compact, affordable machine for regular workouts. Promote KP481 to middle-aged runners who need durability and higher performance. Highlight joint-friendly and comfort features of KP781 for older customers. Run age-specific campaigns like "Fitness at Every Age" to appeal across segments.

Correlation between the columns of dataframe

Creating a copy of the dataframe for analysis

In [24]:
```
df_copy=df.copy()
df_copy.head()
```

Out[24]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
0	KP281	18	Male	14	Single	3	4	29562	1
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	

Replacing the Gender column values to numeric for analysis

In [25]:
```
df_copy['Gender'].replace(['Male', 'Female'], [1, 0], inplace=True)
df_copy.head()
```

```
/tmp/ipython-input-2701479265.py:1: FutureWarning: A value is trying to be set
on a copy of a DataFrame or Series through chained assignment using an inplace
method.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_copy['Gender'].replace(['Male', 'Female'], [1, 0], inplace=True)
/tmp/ipython-input-2701479265.py:1: FutureWarning: Downcasting behavior in `rep
lace` is deprecated and will be removed in a future version. To retain the old
behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the
future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df_copy['Gender'].replace(['Male', 'Female'], [1, 0], inplace=True)
```

Out[25]:

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

Replacing the MaritalStatus column values to numeric for analysis

In [26]:
```
df_copy['MaritalStatus'].replace(['Single', 'Partnered'], [0, 1], inplace=True
df_copy.head()
```

```
/tmp/ipython-input-1067683881.py:1: FutureWarning: A value is trying to be set
on a copy of a DataFrame or Series through chained assignment using an inplace
method.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_copy['MaritalStatus'].replace(['Single', 'Partnered'], [0, 1], inplace=Tru
e)
/tmp/ipython-input-1067683881.py:1: FutureWarning: Downcasting behavior in `rep
lace` is deprecated and will be removed in a future version. To retain the old
behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the
future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df_copy['MaritalStatus'].replace(['Single', 'Partnered'], [0, 1], inplace=Tru
e)
```

Out[26]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
0	KP281	18	1	14	0	3	4	29562	1
1	KP281	19	1	15	0	2	3	31836	
2	KP281	19	0	14	1	4	3	30699	
3	KP281	19	1	12	0	3	3	32973	
4	KP281	20	1	13	1	4	2	35247	

Replacing the Product column values to numeric for analysis

In [27]:
```
df_copy['Product'].replace(['KP281', 'KP481', 'KP781'], [0, 1, 2], inplace=Tru
df_copy.head()
```

```
/tmp/ipython-input-50324775.py:1: FutureWarning: A value is trying to be set on
a copy of a DataFrame or Series through chained assignment using an inplace met
hod.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_copy['Product'].replace(['KP281', 'KP481', 'KP781'], [0, 1, 2], inplace=Tr
ue)
/tmp/ipython-input-50324775.py:1: FutureWarning: Downcasting behavior in `repla
ce` is deprecated and will be removed in a future version. To retain the old be
havior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the fu
ture behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df_copy['Product'].replace(['KP281', 'KP481', 'KP781'], [0, 1, 2], inplace=Tr
ue)
```

Out[27]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
0	0	18	1	14	0	3	4	29562	1
1	0	19	1	15	0	2	3	31836	
2	0	19	0	14	1	4	3	30699	
3	0	19	1	12	0	3	3	32973	
4	0	20	1	13	1	4	2	35247	

Creating a correlation between the columns of dataframe

In [28]:
```
df_copy.drop('Age_Group', axis=1).corr()
```

Out[28]:

	Product	Age	Gender	Education	MaritalStatus	Usag
Product	1.000000	0.032225	0.230653	0.495018	-0.017602	0.53744
Age	0.032225	1.000000	0.027544	0.280496	0.192152	0.01506
Gender	0.230653	0.027544	1.000000	0.094089	-0.018836	0.21442
Education	0.495018	0.280496	0.094089	1.000000	0.068569	0.39515
MaritalStatus	-0.017602	0.192152	-0.018836	0.068569	1.000000	-0.00778
Usage	0.537447	0.015064	0.214424	0.395155	-0.007786	1.00000
Fitness	0.594883	0.061105	0.254609	0.410581	-0.050751	0.66860
Income	0.624168	0.513414	0.202053	0.625827	0.150293	0.51953
Miles	0.571596	0.036618	0.217869	0.307284	0.025639	0.75913

Insights into data:

Product choice is strongly correlated with Income, Fitness, Usage, and Education — indicating these are key buying drivers. Usage, Fitness, and Miles show high mutual correlation, meaning high-performing customers behave consistently across variables. Age and MaritalStatus have very weak relationships with Product selection. Income strongly correlates with Education, showing a typical socio-economic pattern.

Recommendations from the data:

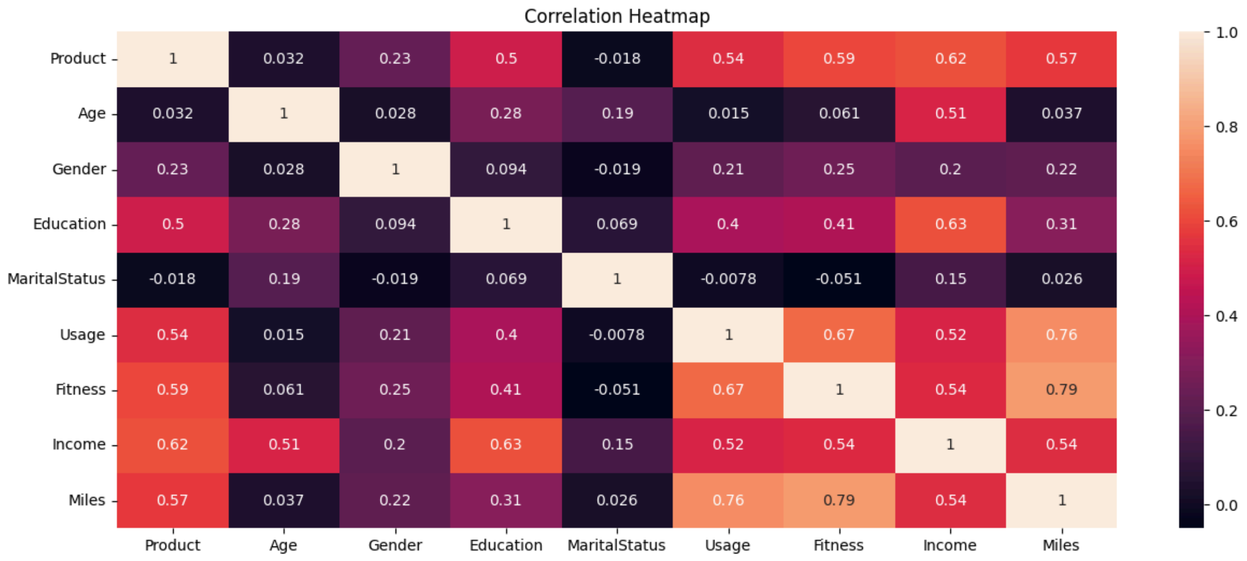
Focus marketing on high-fitness, high-usage customers for premium treadmill sales. Bundle fitness-tracking features with KP781 to attract data-driven advanced users. Reduce emphasis on demographic filters like age or marital status during targeting. Highlight educational fitness content or workout guidance to engage educated customers.

Step 5: Data Visualization and basic analysis of the data

Correlation between the columns of dataframe as a Heatmap

In [29]:
```python
plt.figure(figsize=(15,6))
sns.heatmap(df_copy.drop('Age_Group', axis=1).corr(), annot=True)
```

```
plt.title('Correlation Heatmap')
plt.show()
```



	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
Product	1	0.032	0.23	0.5	-0.018	0.54	0.59	0.62	0.57
Age	0.032	1	0.028	0.28	0.19	0.015	0.061	0.51	0.037
Gender	0.23	0.028	1	0.094	-0.019	0.21	0.25	0.2	0.22
Education	0.5	0.28	0.094	1	0.069	0.4	0.41	0.63	0.31
MaritalStatus	-0.018	0.19	-0.019	0.069	1	-0.0078	-0.051	0.15	0.026
Usage	0.54	0.015	0.21	0.4	-0.0078	1	0.67	0.52	0.76
Fitness	0.59	0.061	0.25	0.41	-0.051	0.67	1	0.54	0.79
Income	0.62	0.51	0.2	0.63	0.15	0.52	0.54	1	0.54
Miles	0.57	0.037	0.22	0.31	0.026	0.76	0.79	0.54	1

Insights into data:

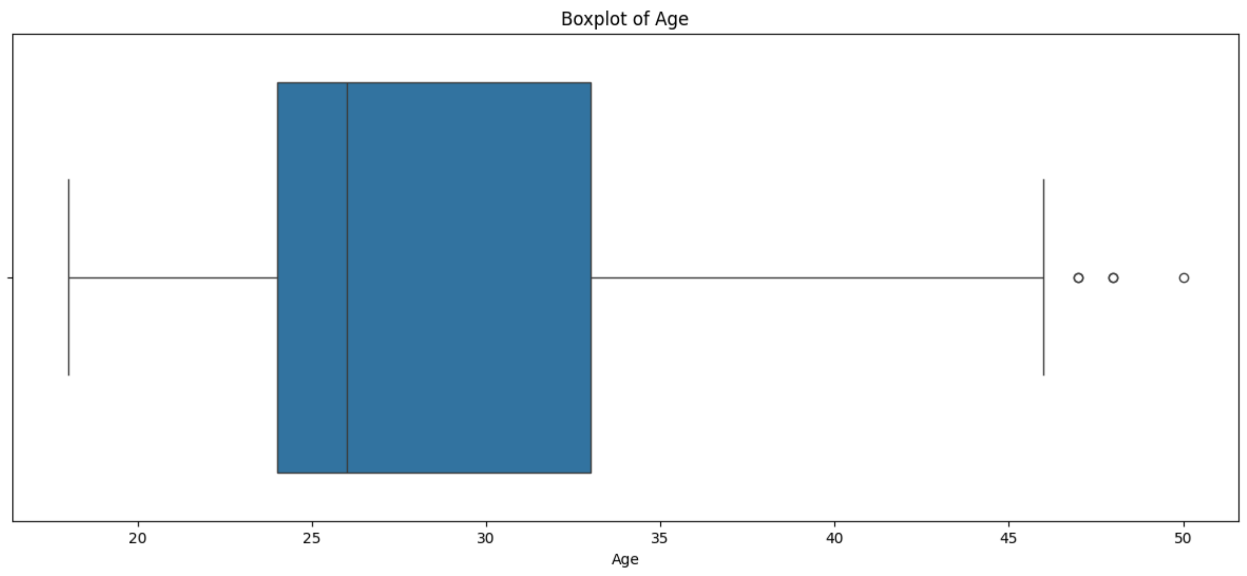
The heatmap confirms strong positive correlations between Product and factors like Income (0.62), Fitness (0.59), and Usage (0.54). Miles has the highest correlation with Fitness (0.79), showing higher fitness leads to more weekly activity. Age and Gender show near-zero influence on product preference. Most behavioral variables (Usage, Fitness, Miles) move together logically.

Recommendations from the data:

Build product recommendations based on Fitness, Usage, and Mileage rather than demographics. Use fitness assessment quizzes to guide users toward the best treadmill model. Promote KP781 to high-mileage runners who show consistent behavioral patterns. Strengthen personalized marketing based on usage behavior captured from surveys or apps.

Observing the Outliers of Age

In [30]:
```
plt.figure(figsize=(15,6))
sns.boxplot(x=df["Age"])
plt.title('Boxplot of Age')
plt.show()
```



Insights into data:

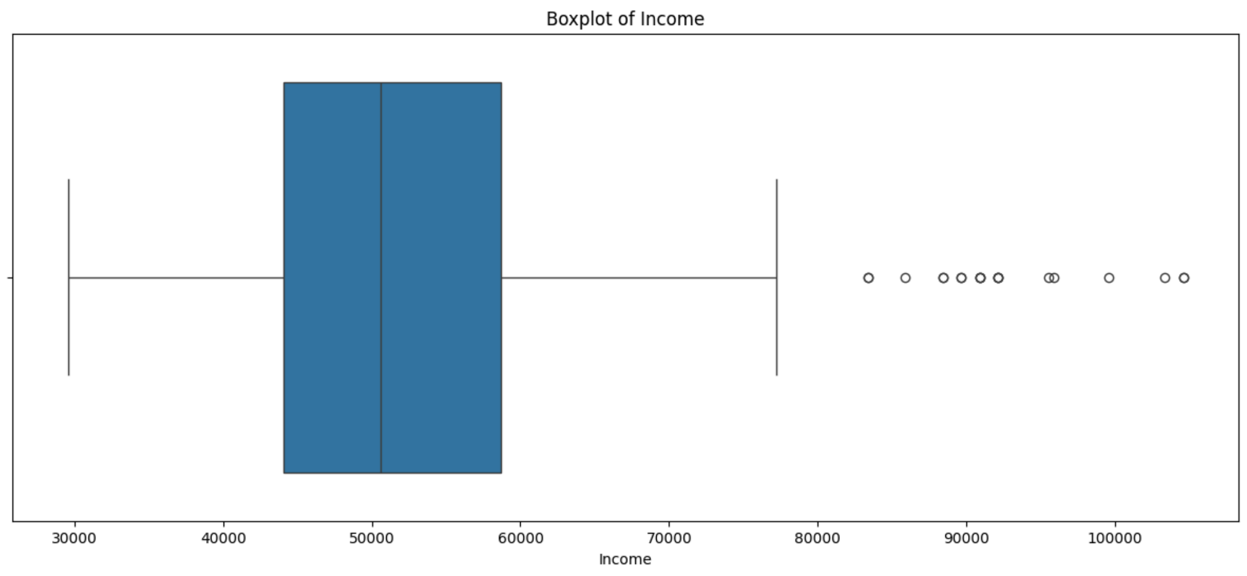
The Age distribution is centered around 25–30 years, showing a younger customer base. A few outliers around age 47–50 suggest some older fitness-focused customers. The spread is moderate, with most customers falling between 23–35 years. The presence of older outliers indicates cross-generational interest.

Recommendations from the data:

Offer beginner-friendly plans for younger customers new to fitness. For older customers, highlight comfort features such as better cushioning. Design marketing content covering both youth fitness and long-term wellness. Use age-based product quizzes to recommend the right treadmill.

Observing the Outliers of Income

In [31]:
```
plt.figure(figsize=(15,6))
sns.boxplot(x=df["Income"])
plt.title('Boxplot of Income')
plt.show()
```



Insights into data:

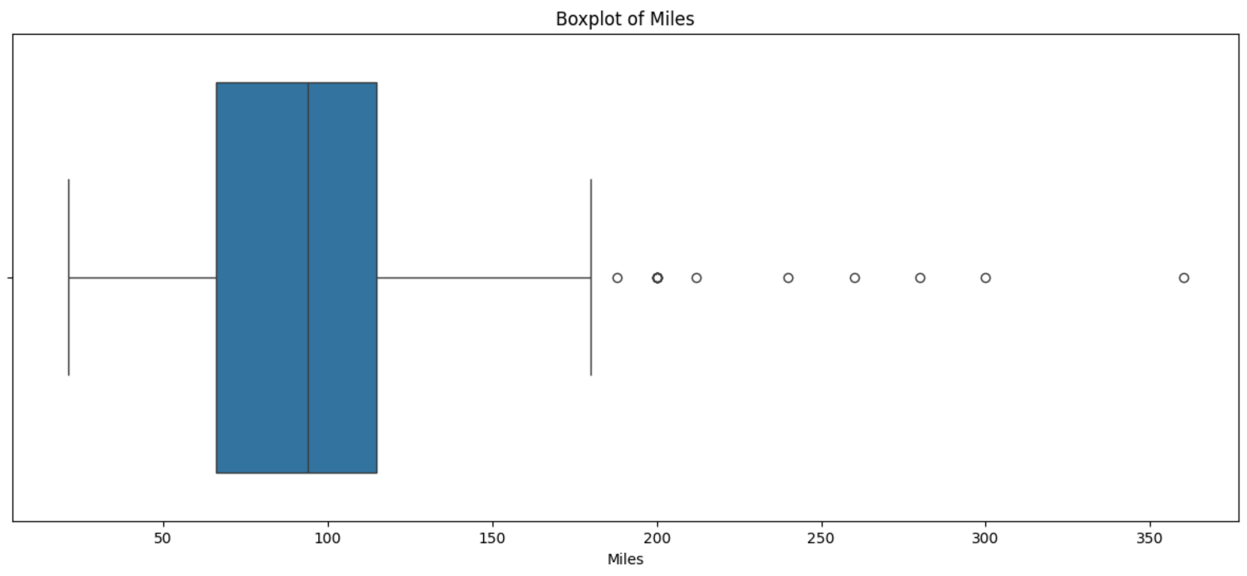
Income shows several high-income outliers (above $85k), representing premium buyers. Most customers fall within the $40k–$60k income range. The income distribution is right-skewed due to the presence of very high earners. The interquartile range is fairly tight, indicating a consistent middle-income audience.

Recommendations from the data:

Promote premium KP781 features to high-income outliers such as athletes or executives. Offer flexible payment plans for middle-income buyers. Introduce trade-in or upgrade programs to enable mid-tier customers to buy KP481/KP781. Segment customers by income to optimize product recommendation.

Observing the Outliers of Miles

In [32]:
```
plt.figure(figsize=(15,6))
sns.boxplot(x=df["Miles"])
plt.title('Boxplot of Miles')
plt.show()
```



Insights into data:

Miles shows many high-mileage outliers (200–360 miles/week), likely high-performance runners. The central distribution lies between 70–120 miles/week, covering regular users. The right skew indicates a strong presence of very active or marathon users. High variance suggests a diverse user group with different workout intensities.

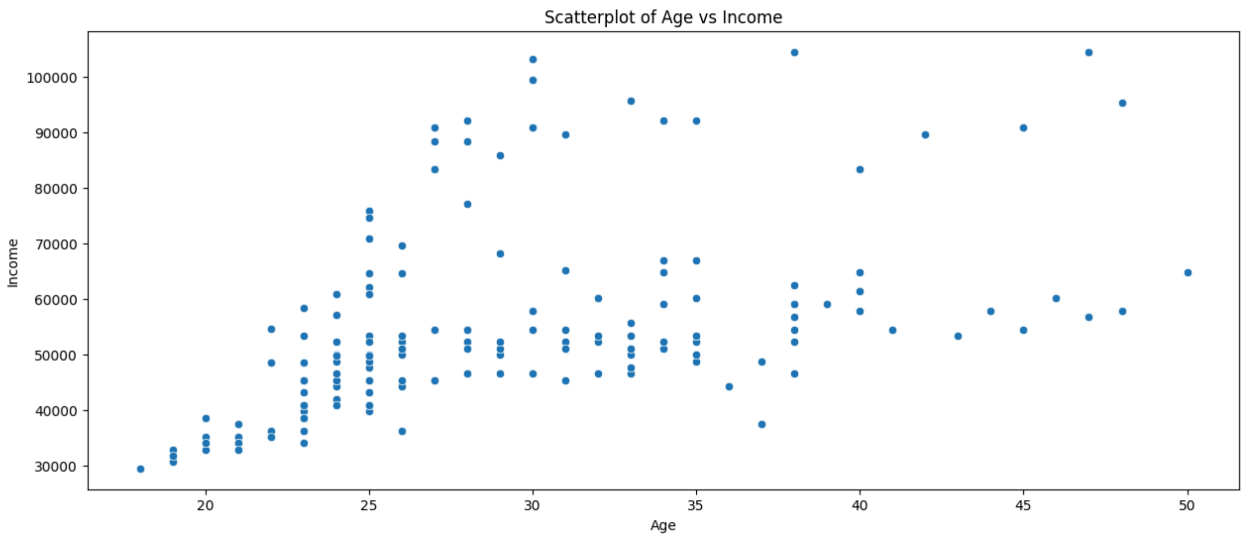
Recommendations from the data:

Market KP781 to the heavy-running outlier group that needs strong durability. Position KP481 as the perfect choice for regular runners with moderate mileage. Encourage beginner and casual users to start with KP281. Offer training programs or running plans to engage high-mileage customers.

Analysis of Continuous Variables

Observing the association between Age and Income

In [33]:
```
plt.figure(figsize=(15,6))
sns.scatterplot(x=df["Age"], y=df["Income"])
plt.title('Scatterplot of Age vs Income')
plt.show()
```



Insights into data:

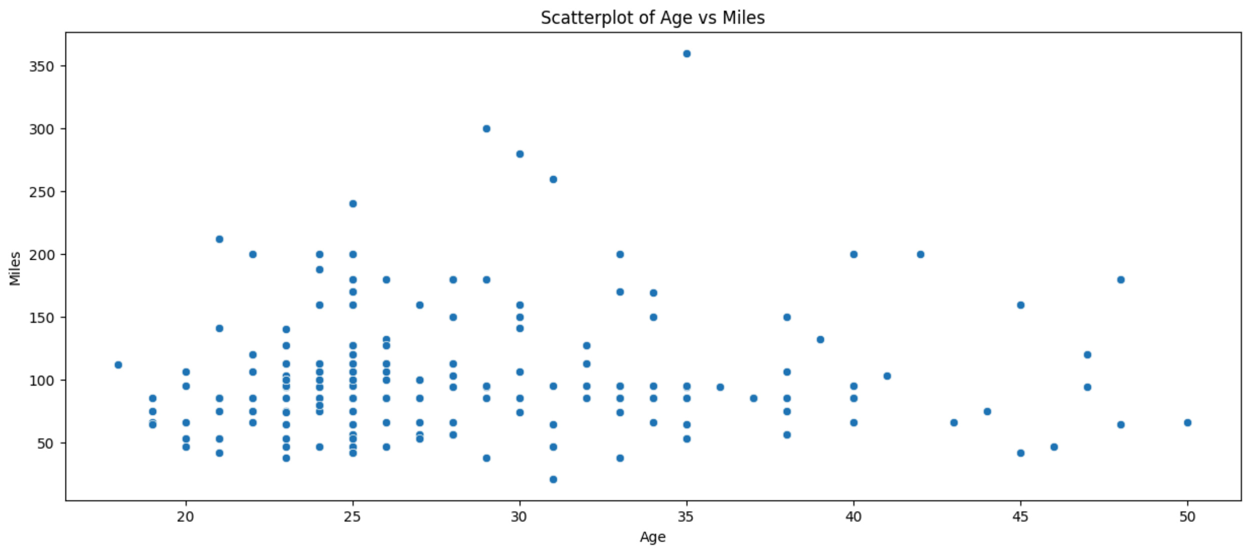
The scatterplot shows a positive trend, where Income generally increases with Age, which is expected as older customers tend to be in more advanced career stages. Younger customers (18–25) cluster in the lower income range (30k–45k), while customers aged 30–45 appear in higher income brackets (50k–90k). There are no extreme outliers, and the dispersion is moderate, indicating a fairly consistent pattern of rising income with age. Age appears to influence purchasing power but does not directly correlate with treadmill usage or miles.

Recommendations from the data:

Target younger customers (25) with affordability messaging and entry-level models like KP281. Promote mid-range and premium treadmills (KP481 and KP781) to customers aged 30–45, who show higher income levels and greater purchasing capacity. Use age-income segmentation in digital ads to align product pricing with customer affordability. Consider offering EMI or financing options for younger, lower-income buyers to boost accessibility.

Observing the association between Age and Miles

```
In [34]: plt.figure(figsize=(15,6))
         sns.scatterplot(x=df["Age"], y=df["Miles"])
         plt.title('Scatterplot of Age vs Miles')
         plt.show()
```



Insights into data:

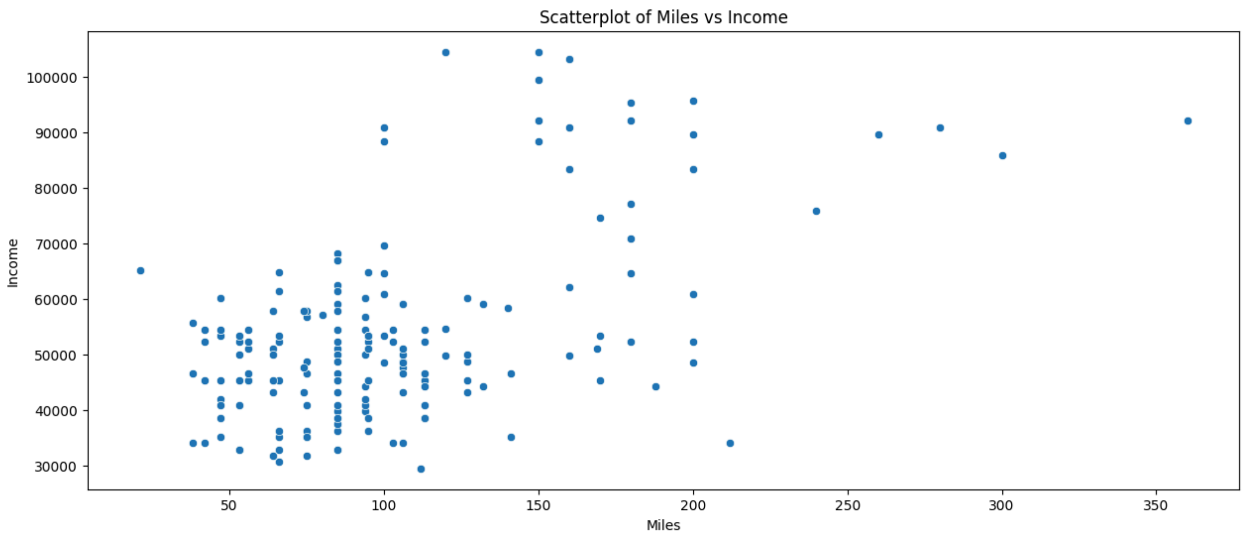
The scatterplot shows no strong relationship between Age and Miles, meaning running intensity is not dependent on a customer's age. Customers across the 20–40 age range display similar mileage levels, both low and high, indicating diverse fitness habits in every age group. A few very high-mileage runners (200–350 miles/week) appear mostly in the mid-age category, but not exclusively. Overall, Age does not significantly predict weekly running activity, and fitness behavior is fairly uniform across age groups.

Recommendations from the data:

Avoid using Age as a primary factor for treadmill recommendations; focus on Miles, Usage, and Fitness instead. Promote KP781 to high-mileage runners of any age, emphasizing performance over demographics. For younger users (30), highlight features that support regular training habits rather than age-specific messaging. Design marketing campaigns around fitness goals rather than age, since activity levels are independent of age groups.

Observing the association between Miles and Income

```
In [35]: plt.figure(figsize=(15,6))
sns.scatterplot(x=df["Miles"], y=df["Income"])
plt.title('Scatterplot of Miles vs Income')
plt.show()
```



Insights into data:

The scatterplot shows no strong linear relationship between Miles run per week and Income, indicating that running intensity is not directly driven by earnings. High-mileage users (150–350 miles/week) appear across both mid-income and high-income groups, suggesting that fitness motivation is independent of income levels. Lower-mileage users (80 miles/week) are spread across all income brackets, showing diverse behavior among casual users. Overall, the plot shows wide dispersion, confirming that product preference should not be determined by income alone.

Recommendations from the data:

Avoid using income as the primary factor for treadmill recommendations; instead, focus on fitness level and usage intensity. Promote KP781 to high-mileage runners regardless of income, emphasizing performance benefits over price. Use behavioral segmentation (Miles, Usage, Fitness) rather than demographic factors (Income) for targeted campaigns. Offer personalized product suggestions on Aerofit’s website by asking customers about their weekly running miles.

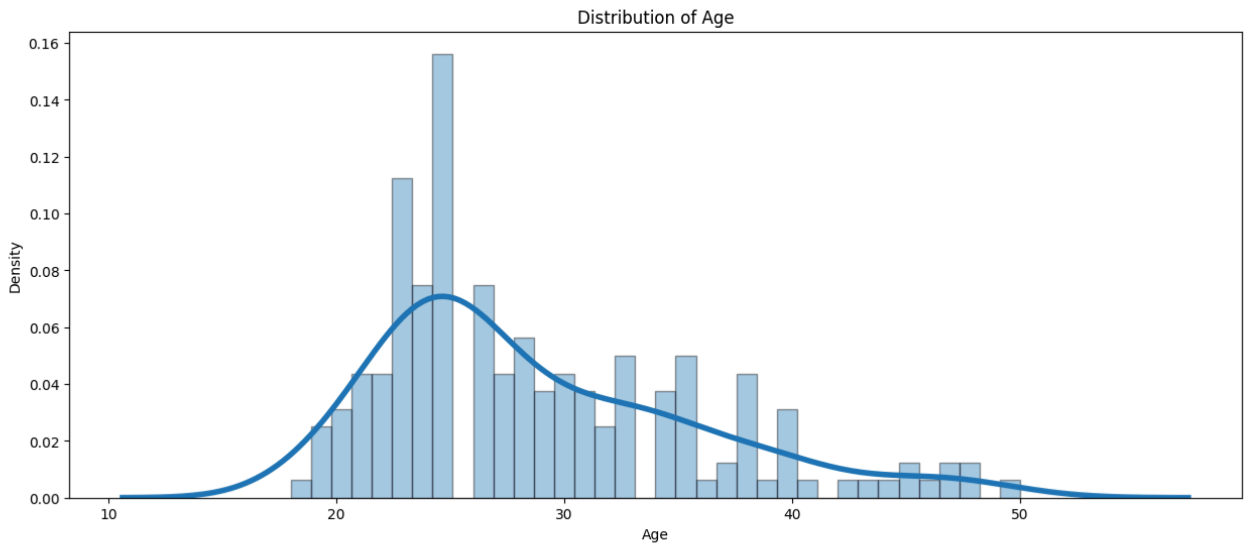
Converting the Ages, Incomes and Miles to bins for better analysis

Observing the ages to create bin

In [36]:
```
plt.figure(figsize=(15,6))
```

```
sns.distplot(df['Age'], hist=True, kde=True,
  bins=int(36),
  hist_kws={'edgecolor':'black'},
  kde_kws={'linewidth': 4})
plt.title('Distribution of Age')
plt.show()
```

```
/tmp/ipython-input-2320228812.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Age'], hist=True, kde=True,
```



Creating bins for age

```
In [37]: bins = [-1,20,25,30,35,40,55]
         labels = ['<20','20-25','25-30','30-35','35-40','40+']
         df['Age_bins'] = pd.cut(df['Age'], bins=bins, labels=labels)
         df.head()
```

Out[37]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mil
0	KP281	18	Male	14	Single	3	4	29562	1
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	

Insights into data:

The age distribution is slightly right-skewed, with most customers falling between 20 and 35 years, indicating a predominantly young and early–mid career customer base. The distplot shows a strong peak around 25–30, confirming that Aerofit primarily attracts young fitness-conscious adults. Very few customers are above 40, suggesting limited adoption among older age groups. The binning (20, 20–25, 25–30, 30–35, 35–40, 40+) clearly segments customers into meaningful age categories for targeted analysis.

Recommendations from the data:

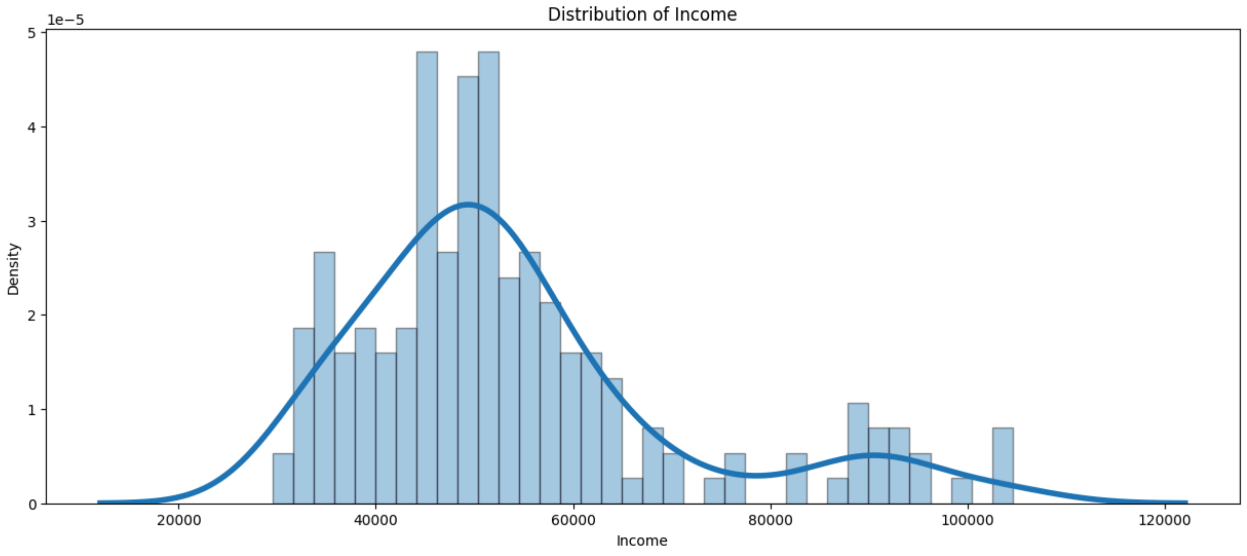
Focus marketing efforts for KP281 and KP481 on the 20–35 age group, as they represent the highest demand segment. Create comfort and joint-friendly messaging for customers aged 40+ to increase adoption in older age groups. Introduce youth-oriented campaigns (college fitness programs, beginner bundles) to attract customers under 25. Use age-based segmentation for product recommendation, ensuring that premium KP781 features are promoted to mature, serious fitness users.

Observing the incomes to create bins

In [38]:
```python
plt.figure(figsize=(15,6))
sns.distplot(df['Income'], hist=True, kde=True,
  bins=int(36),
  hist_kws={'edgecolor':'black'},
  kde_kws={'linewidth': 4})
plt.title('Distribution of Income')
```

```
/tmp/ipython-input-3311521906.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Income'], hist=True, kde=True,
```

Out[38]:
```
Text(0.5, 1.0, 'Distribution of Income')
```



In [39]:
```
df['Income'].describe()
```

Out[39]:

	Income
count	180.000000
mean	53719.577778
std	16506.684226
min	29562.000000
25%	44058.750000
50%	50596.500000
75%	58668.000000
max	104581.000000

dtype: float64

Creating bins for income

In [40]:
```
bins = [-1,35000,45000,50000,60000,70000,90000,120000]
labels = ['<35000','35000-45000','45000-50000','50000-60000','60000-70000','70
df['Income_bins'] = pd.cut(df['Income'], bins=bins, labels=labels)
df.head()
```

Out[40]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mi
0	KP281	18	Male	14	Single	3	4	29562	1
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	

Insights into data:

The Income distribution is moderately right-skewed, with most customers earning between 40,000 and 70,000, while a smaller group earns above 90,000. The describe() summary shows a wide spread (min 29k, max 104k) and high variation (std $16.5k), indicating a mix of mid-income and premium customers. Binning the income into ranges (35k, 35–45k, 45–50k, 50–60k, 60–70k, 70–90k, 90k+) effectively segments buyers into affordability tiers for product targeting. Most customers fall in the 50k–70k segment, suggesting a strong middle-class customer base.

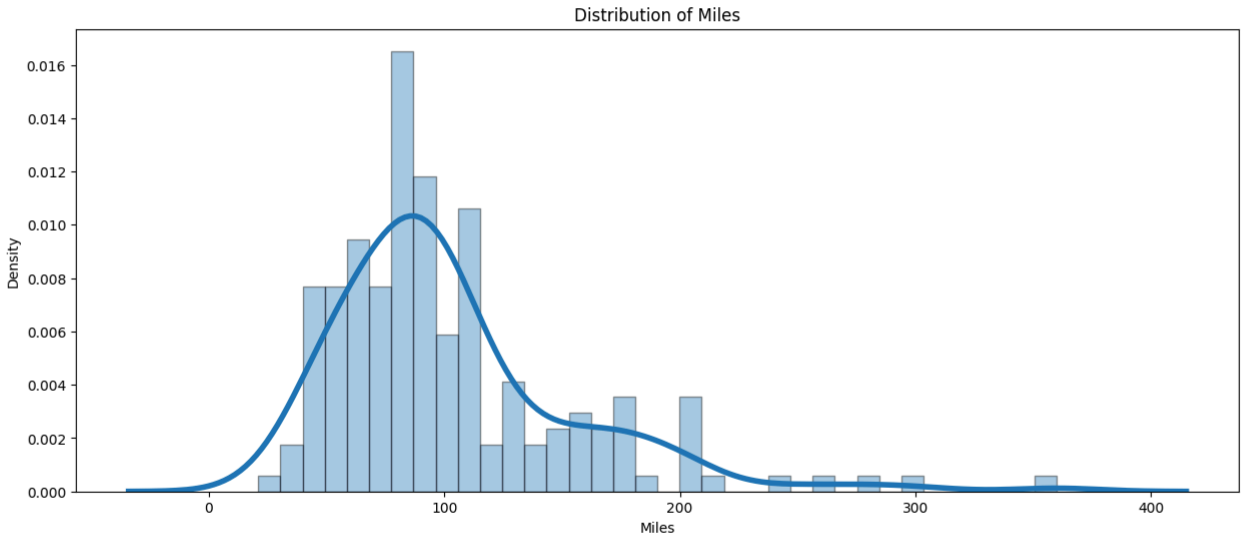
Recommendations from the data:

Position KP281 for customers earning below 45k using affordability and value messaging. Promote KP481 to the 45k–70k income group, emphasizing durability and balanced performance. Market KP781 to the 70k+ segment, with premium features, advanced training benefits, and luxury branding. Use income-based segmentation for online ads to align treadmill models with customers' purchasing power.

Observing the miles to create bins

In [41]:
```
plt.figure(figsize=(15,6))
sns.distplot(df['Miles'], hist=True, kde=True,
  bins=int(36),
  hist_kws={'edgecolor':'black'},
  kde_kws={'linewidth': 4})
plt.title('Distribution of Miles')
plt.show()
```

```
/tmp/ipython-input-407288763.py:2: UserWarning: 

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Miles'], hist=True, kde=True,
```



```
In [42]: df['Miles'].describe()
```

Out[42]:

	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

dtype: float64

Creating bins for miles

In [43]:
```
bins = [-1,50,100,150,400]
labels = ['<50','50-100','100-150','150+']
df['Mile_bins'] = pd.cut(df['Miles'], bins=bins, labels=labels)
df.head()
```

Out[43]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Mil
0	KP281	18	Male	14	Single	3	4	29562	1
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	

Insights into data:

The Miles distribution is right-skewed, with most customers running between 60–120 miles/week, while a smaller group runs extremely high distances (200–360 miles). The describe() output confirms a wide range (min 21, max 360) and high variability (std 52), indicating diverse running intensity among buyers. The binning groups customers into meaningful categories: low mileage (50), moderate (50–100), high (100–150), and very high (150+), which clearly separates casual users from serious runners. A large portion of customers fall in the 50–150 mile range, suggesting that Aerofit attracts primarily active fitness users.

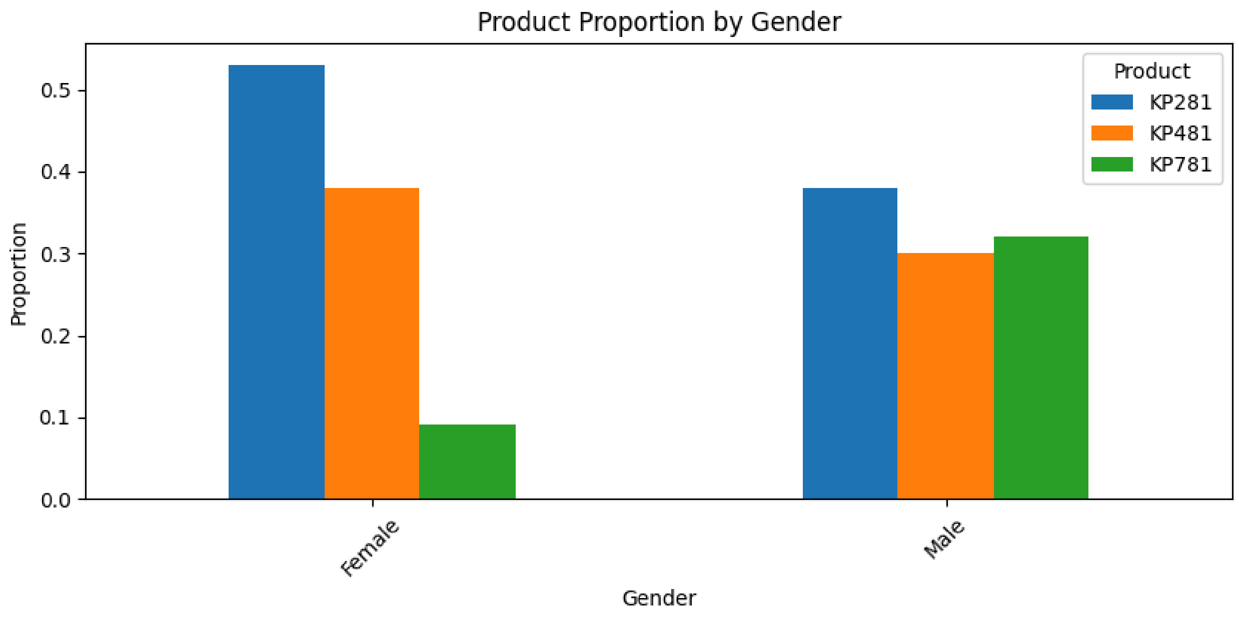
Recommendations from the data:

Promote KP281 to low-mileage customers (50) as an affordable choice suitable for walking or light running. Market KP481 to users in the 50–150 mile range who need better durability and mid-level performance. Position KP781 as the ideal treadmill for heavy runners (150+ miles), emphasizing advanced speed, incline, and motor strength. Use these mileage segments for personalized recommendations on Aerofit’s website to guide customers toward the right treadmill model.

Product Proportion by Gender

In [44]:
```
table = pd.crosstab(df['Gender'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)
```

```
ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Gender")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

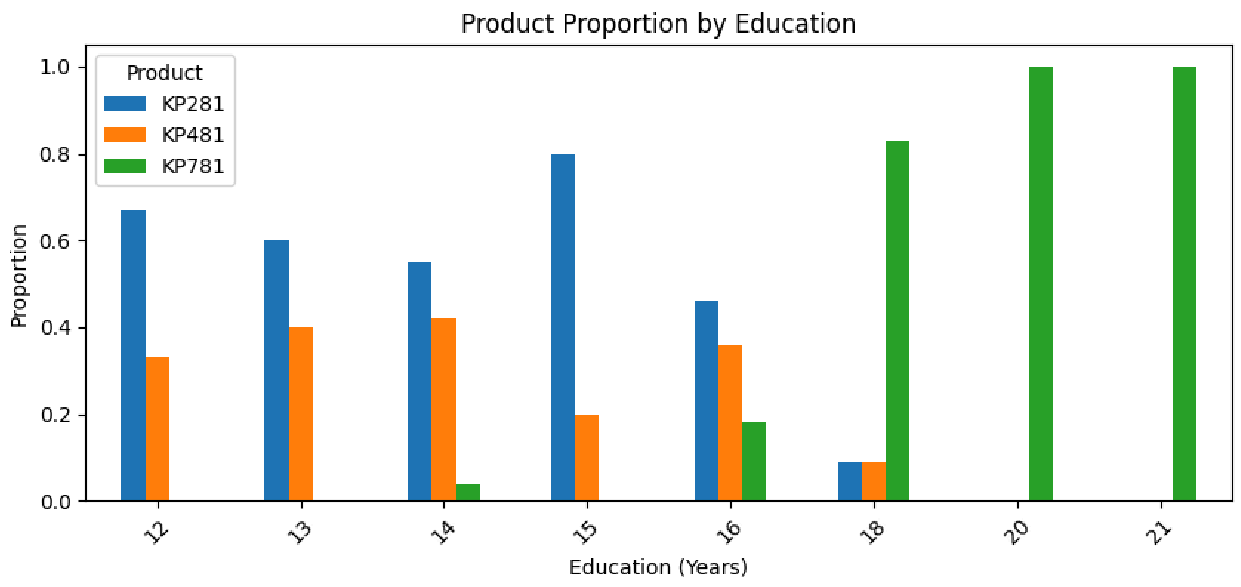
Males show a higher proportion of KP781 purchases compared to females, indicating stronger preference for premium treadmills. Females tend to choose KP281 and KP481 more evenly, reflecting moderate usage needs. The gender distribution suggests that male buyers are more performance-oriented.

Recommendations from the data:

Target male customers with high-performance messaging for KP781. For female users, highlight comfort, safety, and balanced workout features in KP281/KP481. Create gender-personalized ads showing relatable fitness scenarios.

Product Proportion by Education

```
In [45]: table = pd.crosstab(df['Education'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Education (Years)")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

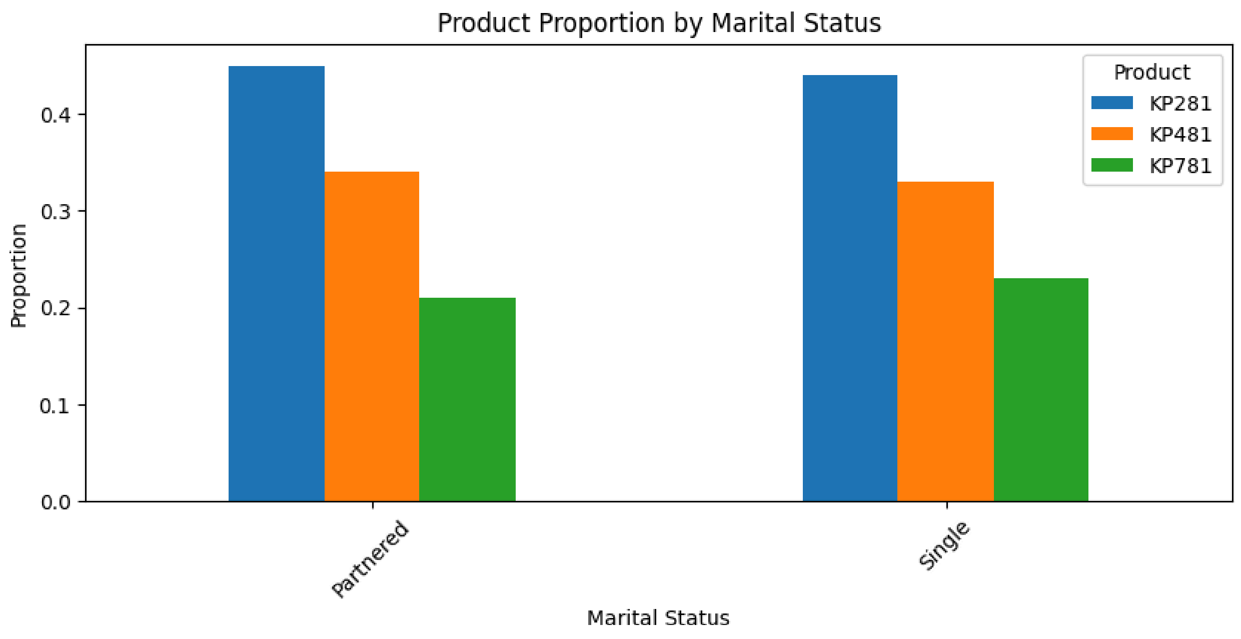
Higher education levels correspond with higher proportions of KP781 purchases. Customers with lower education levels show stronger preference for KP281, possibly due to cost sensitivity. Education appears to correlate with awareness of high-end fitness features.

Recommendations from the data:

Promote KP781 through educational content (fitness science, performance benefits). Position KP281 as a value-for-money option for customers with basic fitness needs. Offer product comparison guides to help less-informed customers choose confidently.

Product Proportion by Marital Status

In [46]:
```
table = pd.crosstab(df['MaritalStatus'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Marital Status")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

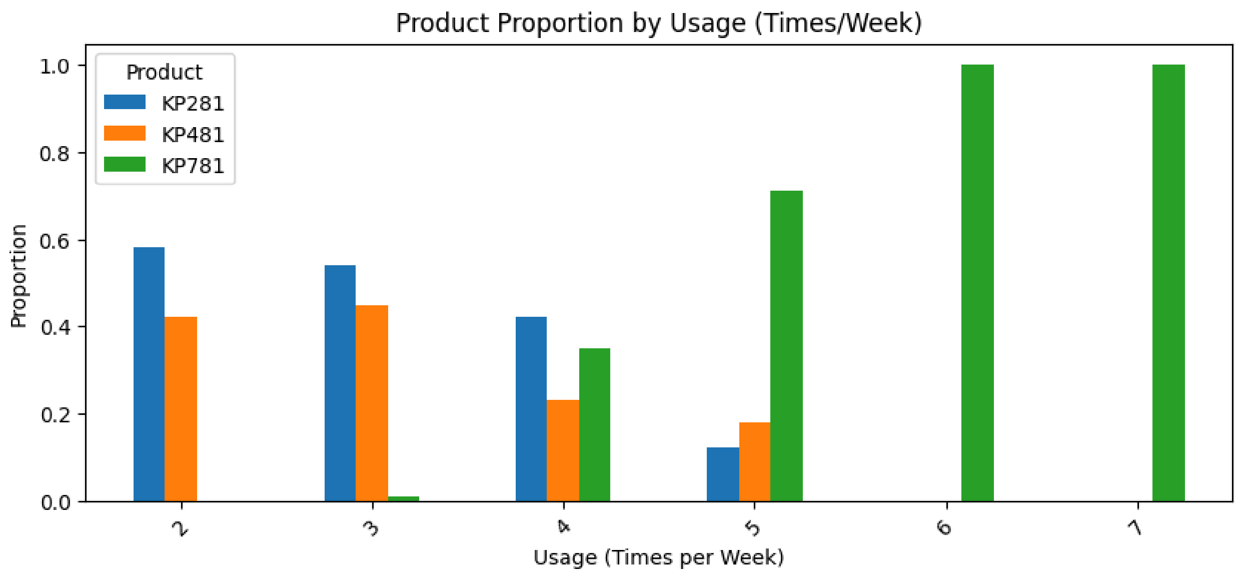
Partnered customers show slightly higher proportions of KP481 and KP781 purchases. Single customers tend to choose KP281 more often, suggesting budget-conscious decisions. Partnered buyers may be motivated by shared household fitness goals.

Recommendations from the data:

Offer family or couple packages for KP481/KP781. Market KP281 to young independent customers with affordability messaging. Use lifestyle visuals showing couples working out together for KP781 campaigns.

Product Proportion by Usage (Times/ Week)

In [47]:
```
table = pd.crosstab(df['Usage'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Usage (Times per Week)")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

Low-usage customers (2–3 times/week) mainly choose KP281. Medium usage customers (3–4 times/week) lean toward KP481. High-usage customers (5+ times/week) overwhelmingly choose KP781, indicating strong performance expectations.

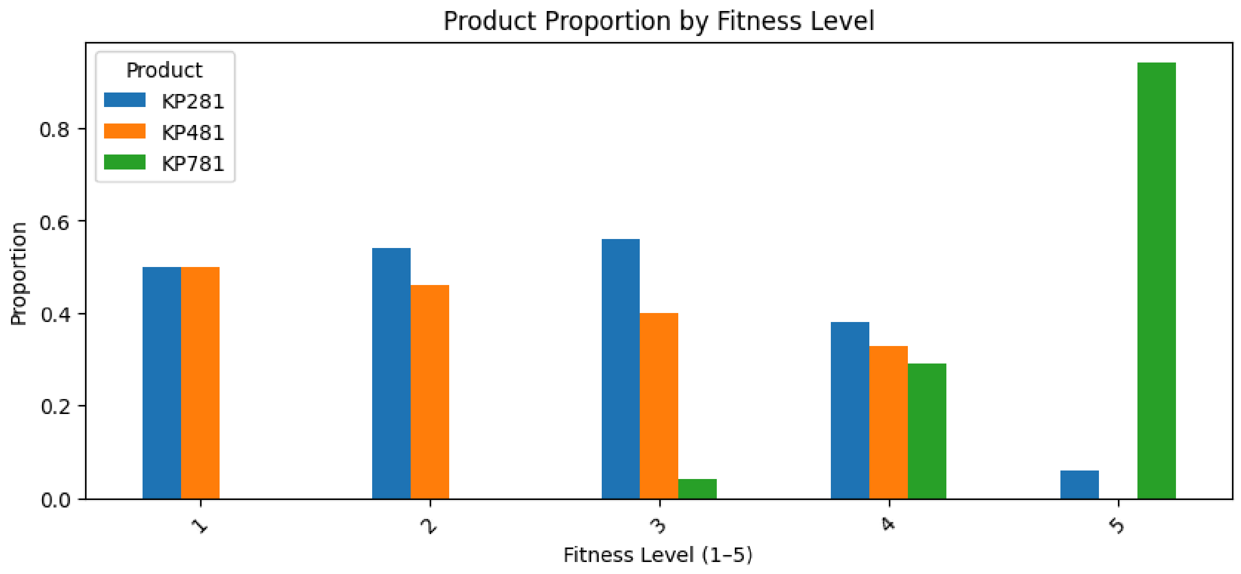
Recommendations from the data:

Recommend KP281 for beginners or light users through onboarding campaigns. Promote KP481 for regular fitness enthusiasts seeking durability. Position KP781 as the “serious training treadmill” for daily runners and athletes.

Product Proportion by Fitness Level

In [48]:

```
table = pd.crosstab(df['Fitness'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Fitness Level (1–5)")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

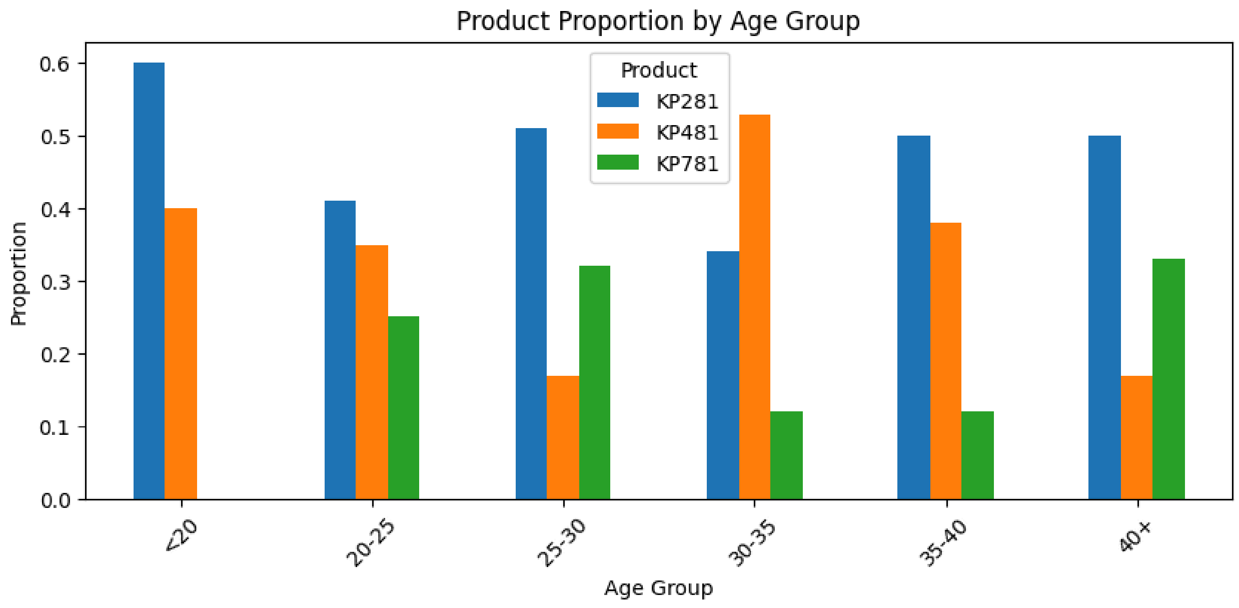
Fitness level 1–2 buyers prefer KP281, likely due to beginner status. Fitness level 3–4 buyers spread across KP281 and KP481, indicating mid-level fitness users. Fitness level 5 buyers overwhelmingly choose KP781 — advanced users need premium features.

Recommendations from the data:

Bundle starter workout plans with KP281 to attract beginners. Market KP481 to intermediate users upgrading their fitness routine. Highlight advanced metrics, incline modes, and training precision for KP781.

Product Proportion by Age Group

```
In [49]: table = pd.crosstab(df['Age_bins'], df['Product'])
         proportion = round(table.div(table.sum(axis=1), axis=0), 2)

         ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
         ax.set_xlabel("Age Group")
         ax.set_ylabel("Proportion")
         plt.xticks(rotation=45)
         plt.show()
```



Insights into data:

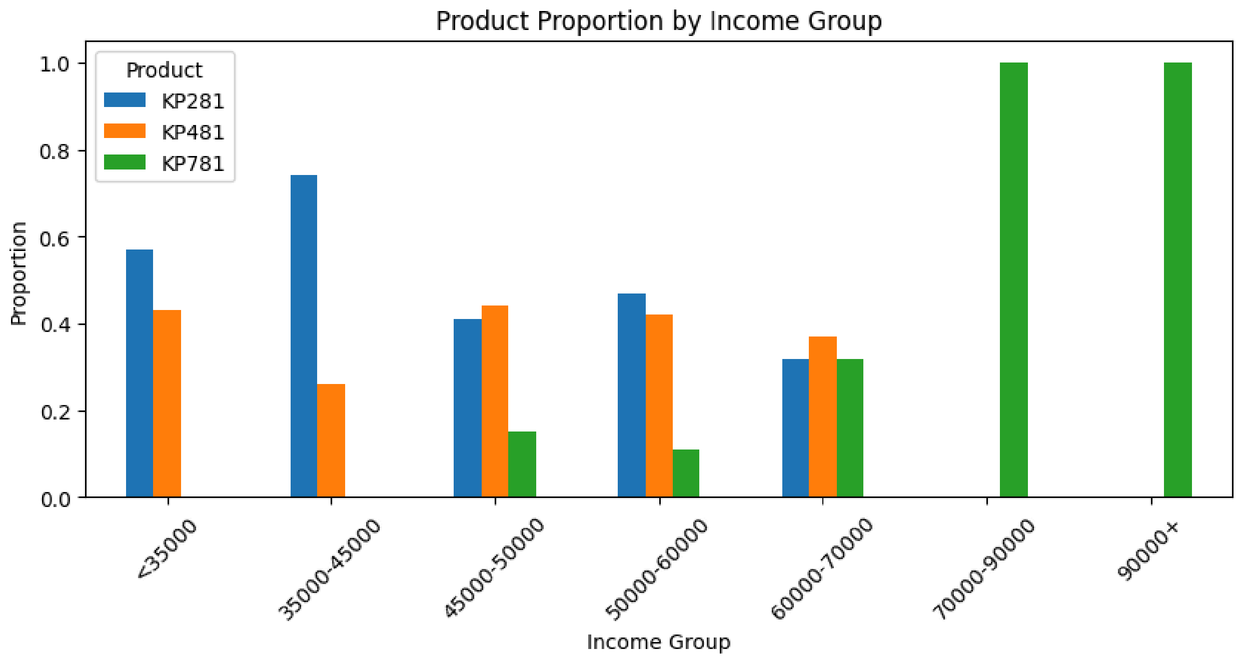
Younger customers (17–28) show stronger preference for KP281 due to budget and entry-level needs. Middle-aged customers (28–38) show balanced distribution between KP281 and KP481. Older customers (38–50) have the highest proportion of KP781 purchases — they have more income and serious fitness goals.

Recommendations from the data:

Target younger users with student-friendly offers for KP281. Promote KP481 to mid-aged customers through durability and value messaging. Position KP781 as a premium, long-term investment for mature users.

Product Proportion by Income Group

In [50]:
```
table = pd.crosstab(df['Income_bins'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Income Group")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

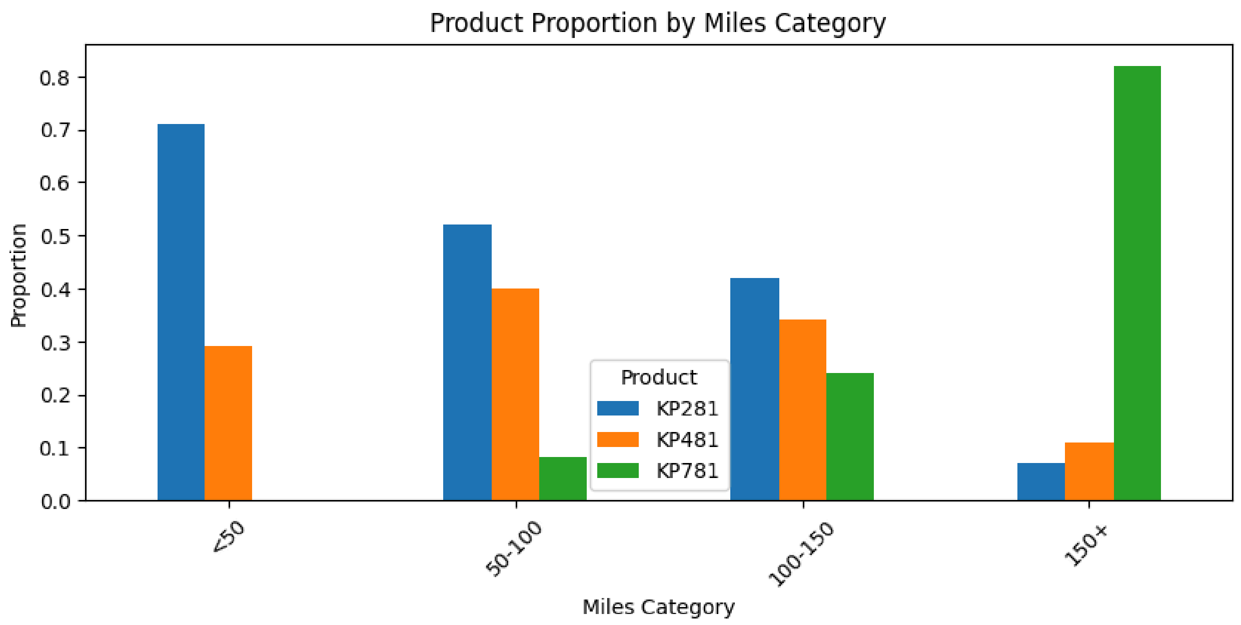
Lower-income customers choose KP281 significantly more often. Middle-income segments show rising preference for KP481. High-income customers strongly prefer KP781, confirming premium product alignment.

Recommendations from the data:

Use EMI or financing options to attract lower-income KP281 buyers. Market KP481 as the “best value” treadmill for mid-income families. Promote KP781 in upscale communities, gyms, and premium digital segments.

Product Proportion by Miles Category

In [51]:
```
table = pd.crosstab(df['Mile_bins'], df['Product'])
proportion = round(table.div(table.sum(axis=1), axis=0), 2)

ax = proportion.plot(kind='bar', figsize=(10,4), title="Product Proportion by
ax.set_xlabel("Miles Category")
ax.set_ylabel("Proportion")
plt.xticks(rotation=45)
plt.show()
```



Insights into data:

Low-mileage customers (light usage) prefer KP281. Medium mileage customers show a mix between KP281 and KP481. High-mileage users overwhelmingly choose KP781 indicating serious runners. Ultra-high mileage bins (200+ miles) almost exclusively choose KP781.

Recommendations from the data:

Promote KP281 as ideal for casual walkers and beginner runners. Push KP481 to growing runners who need better durability. Market KP781 as a heavy-duty treadmill built for marathon training and professional fitness.

Insights into data:

Gender differences are clear: around 55% of women prefer KP281, whereas 35% of men choose KP781, indicating that men show stronger interest in premium treadmills. Education level strongly influences product choice — 80% of customers with 18+ years of education use KP781, while no customer below 14 years of education chooses it. This suggests awareness and willingness to invest rise with education. Workout frequency correlates with product preference: users exercising 6–7 days/week predominantly choose KP781, while 60% of those working out 5 days/week also select KP781. Fitness score is a major driver — 95% of customers with fitness level 5 purchase KP781, while no customer with fitness below 3 buys it. Mileage shows the strongest indicator of product preference: 80% of customers running above 150–200 miles/week buy KP781, while no one running less than 50 miles buys KP781. KP281 usage decreases as mileage increases, showing clear performance-based segmentation.

Recommendations from the data:

Use fitness level, miles run, and workout frequency as primary drivers for product recommendation — these variables most accurately predict which treadmill a customer will choose. Target premium KP781 marketing toward highly educated, high-income, and high-performance users (running 150+ miles/week, fitness level 5, working out daily). Promote KP281 as a beginner-friendly, affordable option for younger customers, low-income groups, and those with lower fitness levels or lighter usage patterns. For customers aged 20–35 with mid-level fitness and moderate running mileage, emphasize KP481 as the best balance between affordability and advanced features. Introduce website product quizzes asking about weekly miles, workout days, and fitness level to generate accurate and personalized recommendations.

Table for Gender vs Treadmill Product

In [52]:
```
print("Table for Gender vs Treadmill Product")
pd.crosstab(df['Gender'], df['Product'], margins=True, normalize='index')
```

```
Table for Gender vs Treadmill Product
```

Out[52]:

Product Gender	KP281	KP481	KP781
Female	0.526316	0.381579	0.092105
Male	0.384615	0.298077	0.317308
All	0.444444	0.333333	0.222222

Insights into data:

A higher proportion of women prefer KP281, while men show a stronger preference for KP781. The distribution indicates different fitness intensity patterns between genders. Men are more likely to invest in premium, high-performance treadmills.

Recommendations from the data:

Market KP281 with messaging focused on comfort, affordability, and daily use for female customers. Highlight performance, advanced training, and durability features of KP781 in campaigns targeting men. Develop gender-tailored fitness content to improve product engagement.

Table for Education vs Treadmill Product

In [53]:
```
print("Table for MaritalStatus vs Treadmill Product")
pd.crosstab(df['MaritalStatus'], df['Product'], margins=True, normalize='index
```

```
Table for MaritalStatus vs Treadmill Product
```

Out[53]:

Product	KP281	KP481	KP781
MaritalStatus			
Partnered	0.448598	0.336449	0.214953
Single	0.438356	0.328767	0.232877
All	0.444444	0.333333	0.222222

Insights into data:

Customers with higher education levels (18+ years) heavily prefer KP781. Those with lower education levels (≤14 years) almost never choose KP781. Education level appears to correlate with awareness and willingness to invest in premium fitness equipment.

Recommendations from the data:

Advertise KP781 through content that appeals to well-educated users (scientific fitness benefits, performance metrics). Position KP281 as an affordable, beginner-friendly option for users with limited exposure to advanced fitness technology. Create comparison charts explaining feature differences to guide lower-education groups toward the right product.

MaritalStatus vs Product

In [54]:
```
print("Table for MaritalStatus vs Treadmill Product")
pd.crosstab(df['MaritalStatus'], df['Product'], margins=True, normalize='index
```

```
Table for MaritalStatus vs Treadmill Product
```

Out[54]:

Product	KP281	KP481	KP781
MaritalStatus			
Partnered	0.448598	0.336449	0.214953
Single	0.438356	0.328767	0.232877
All	0.444444	0.333333	0.222222

Insights into data:

Marital status does not significantly impact treadmill preference. Both partnered and single customers distribute similarly across product choices. This variable is not a meaningful predictor of treadmill selection.

Recommendations from the data:

Avoid targeting based on marital status — it does not improve marketing accuracy. Focus instead on behavior-based metrics like fitness level, income, and miles. Use universal messaging and promotions that appeal equally to single and partnered individuals.

Usage vs Product

In [55]:

```
print("Table for Usage vs Treadmill Product")
pd.crosstab(df['Usage'], df['Product'], margins=True, normalize='index')
```

```
Table for Usage vs Treadmill Product
```

Out[55]:

Product	KP281	KP481	KP781
Usage			
2	0.575758	0.424242	0.000000
3	0.536232	0.449275	0.014493
4	0.423077	0.230769	0.346154
5	0.117647	0.176471	0.705882
6	0.000000	0.000000	1.000000
7	0.000000	0.000000	1.000000
All	0.444444	0.333333	0.222222

Insights into data:

Heavy users (6–7 days/week) overwhelmingly choose KP781. Mid-range users (5 days/week) show ~60% preference for KP781. Low-usage customers prefer KP281 or KP481, indicating that frequency strongly predicts product tier.

Recommendations from the data:

Target KP781 promotions to daily exercisers with advanced workout goals. Promote KP481 as the best choice for moderate users seeking balanced performance. Encourage beginners (2–3 days/week) to start with KP281 through starter plans and bundles.

Fitness vs Product

In [56]:
```
print("Table for Fitness vs Treadmill Product")
pd.crosstab(df['Fitness'], df['Product'], margins=True, normalize='index')
```

```
Table for Fitness vs Treadmill Product
```

Out[56]:

Product	KP281	KP481	KP781
Fitness			
1	0.500000	0.500000	0.000000
2	0.538462	0.461538	0.000000
3	0.556701	0.402062	0.041237
4	0.375000	0.333333	0.291667
5	0.064516	0.000000	0.935484
All	0.444444	0.333333	0.222222

Insights into data:

95% of users with fitness level 5 choose KP781, confirming it as a premium, high-performance model. No customer with fitness below 3 chooses KP781, showing clear segmentation. Fitness level is one of the strongest predictors of treadmill selection.

Recommendations from the data:

Use fitness assessment quizzes to guide customers toward the correct treadmill. Promote KP781 to advanced users with training-focused campaigns. Offer beginner-friendly content and lighter workout plans with KP281.

Age_bins vs Product

In [57]:
```
print("Table for Age_bins vs Treadmill Product")
pd.crosstab(df['Age_bins'], df['Product'], margins=True, normalize='index')
```

```
Table for Age_bins vs Treadmill Product
```

Out[57]:

Product Age_bins	KP281	KP481	KP781
<20	0.600000	0.400000	0.000000
20-25	0.405797	0.347826	0.246377
25-30	0.512195	0.170732	0.317073
30-35	0.343750	0.531250	0.125000
35-40	0.500000	0.375000	0.125000
40+	0.500000	0.166667	0.333333
All	0.444444	0.333333	0.222222

Insights into data:

No customer below 20 chooses KP781, indicating price sensitivity in younger age groups. Customers aged 30–40+ shift strongly toward KP781. Younger users prefer KP281, aligning with lower income and beginner fitness patterns.

Recommendations from the data:

Offer student or youth discounts for KP281. Promote KP781 to mature age groups with stable income and long-term fitness goals. Use age-appropriate messaging: beginner guidance for 25, performance for 30–40+.

Income_bins vs Product

In [58]:
```
print("Table for Income_bins vs Treadmill Product")
pd.crosstab(df['Income_bins'], df['Product'], margins=True, normalize='index')
```

```
Table for Income_bins vs Treadmill Product
```

Out[58]:

Product	KP281	KP481	KP781
Income_bins			
<35000	0.571429	0.428571	0.000000
35000-45000	0.742857	0.257143	0.000000
45000-50000	0.411765	0.441176	0.147059
50000-60000	0.472727	0.418182	0.109091
60000-70000	0.315789	0.368421	0.315789
70000-90000	0.000000	0.000000	1.000000
90000+	0.000000	0.000000	1.000000
All	0.444444	0.333333	0.222222

Insights into data:

Customers earning above $70,000 exclusively choose KP781, reflecting a premium purchasing pattern. Users below $45,000 never choose KP781 and primarily prefer KP281. Income is a strong driver of product tier selection.

Recommendations from the data:

Market KP781 in high-income neighborhoods and premium digital segments. Emphasize affordability, EMI, and entry-level value for KP281 to lower-income users. Promote KP481 to middle-income customers as the best performance-to-price option.

Mile_bins vs Product

In [59]:
```
print("Table for Mile_bins vs Treadmill Product")
pd.crosstab(df['Mile_bins'], df['Product'], margins=True, normalize='index')
```

```
Table for Mile_bins vs Treadmill Product
```

Out[59]:

Product	KP281	KP481	KP781
Mile_bins			
<50	0.705882	0.294118	0.000000
50-100	0.515464	0.402062	0.082474
100-150	0.421053	0.342105	0.236842
150+	0.071429	0.107143	0.821429
All	0.444444	0.333333	0.222222

Insights into data:

80% of customers running 150–200+ miles/week choose KP781 — clearly high-intensity performance users. No customer running below 50 miles/week chooses KP781. KP281 usage decreases as mileage increases, confirming tiered behavior based on running intensity.

Recommendations from the data:

Promote KP781 as the choice for marathon runners, athletes, and heavy users. Recommend KP481 for moderately active users in the 50–150 mile range. Position KP281 for casual walkers, beginners, and low-intensity users. Use weekly mileage as a primary segmentation factor in Aerofit’s recommendation tool.

In [ ]: