In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import chi2_contingency
```

In [2]:
```
data = pd.read_csv("aerofit_treadmill.csv")
data
```

Out[2]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles
0	KP281	18	Male	14	Single	3	4	29562	112
1	KP281	19	Male	15	Single	2	3	31836	75
2	KP281	19	Female	14	Partnered	4	3	30699	66
3	KP281	19	Male	12	Single	3	3	32973	85
4	KP281	20	Male	13	Partnered	4	2	35247	47
...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200
176	KP781	42	Male	18	Single	5	4	89641	200
177	KP781	45	Male	16	Single	5	5	90886	160
178	KP781	47	Male	18	Partnered	4	5	104581	120
179	KP781	48	Male	18	Partnered	4	5	95508	180

180 rows × 9 columns

In [3]:
```
z = data.describe(include = 'int')
# As mean is greater than 50 percentile for Age, Usage, Fitness, Income amd Miles, these variable distribution are right skew
# As mean is less than median for Education, these variable distribution are left skewed.
col = []
out_count = []
for i in z.columns:
    IQR = z.loc['75%',i] - z.loc['25%',i]
    low_limit = z.loc['25%',i]-1.5*IQR
    upp_limit = z.loc['75%',i]+1.5*IQR
    out_count.append(data.loc[(data[i] < low_limit) | (data[i] > upp_limit),i].count())
    col.append(i)
# Generating a Dataframe of total outlier counts for each columns in the dataset
pd.DataFrame([col, out_count]).T.rename(columns = {0:"Col", 1: "Outlier_count"})
```

Out[3]:

	Col	Outlier_count
0	Age	5
1	Education	4
2	Usage	9
3	Fitness	2
4	Income	19
5	Miles	13

In [4]:
```
z
# As seen from table below:
# Age column: min is 18 yrs, max is 50yrs.
# Education column: min is 12 yrs, max is 21 yrs.
# Usage: min is 2 times a week where as max is 7 times a week.
# Fitness: min self rating is 1 and maximum self rating is 5
# Income: min Income is 29562 and max is 104581
# Miles: min is 21 whereas max is 360.
```

Out[4]:

	Age	Education	Usage	Fitness	Income	Miles
count	180.000000	180.000000	180.000000	180.000000	180.000000	180.000000
mean	28.788889	15.572222	3.455556	3.311111	53719.577778	103.194444
std	6.943498	1.617055	1.084797	0.958869	16506.684226	51.863605
min	18.000000	12.000000	2.000000	1.000000	29562.000000	21.000000
25%	24.000000	14.000000	3.000000	3.000000	44058.750000	66.000000
50%	26.000000	16.000000	3.000000	3.000000	50596.500000	94.000000
75%	33.000000	16.000000	4.000000	4.000000	58668.000000	114.750000
max	50.000000	21.000000	7.000000	5.000000	104581.000000	360.000000

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 180 entries, 0 to 179
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Product        180 non-null    object
 1   Age            180 non-null    int64 
 2   Gender         180 non-null    object
 3   Education      180 non-null    int64 
 4   MaritalStatus  180 non-null    object
 5   Usage          180 non-null    int64 
 6   Fitness        180 non-null    int64 
 7   Income         180 non-null    int64 
 8   Miles          180 non-null    int64 
dtypes: int64(6), object(3)
memory usage: 12.8+ KB
```

In [6]:
```
data.isna().sum(axis = 0)
# As can be seen in the dataset below, there are no datasets.
```

Out[6]:
```
Product          0
Age              0
Gender           0
Education        0
MaritalStatus    0
Usage            0
Fitness          0
Income           0
Miles            0
dtype: int64
```

In [7]:
```
data["MaritalStatus"].value_counts()
# There are 107 partnered and 73 Single customers in the dataset.
```

Out[7]:
```
Partnered    107
Single        73
Name: MaritalStatus, dtype: int64
```

In [8]:
```
data["Product"].value_counts()
# 80 customers have purchased KP281, 60 have purchased KP481 and 40 have purchased KP781.
```

Out[8]:
```
KP281    80
KP481    60
KP781    40
Name: Product, dtype: int64
```

In [9]:
```
data["Gender"].value_counts()
# 104 customers are Males and 76 customers are female in the dataset.
```

Out[9]:
```
Male      104
Female     76
Name: Gender, dtype: int64
```

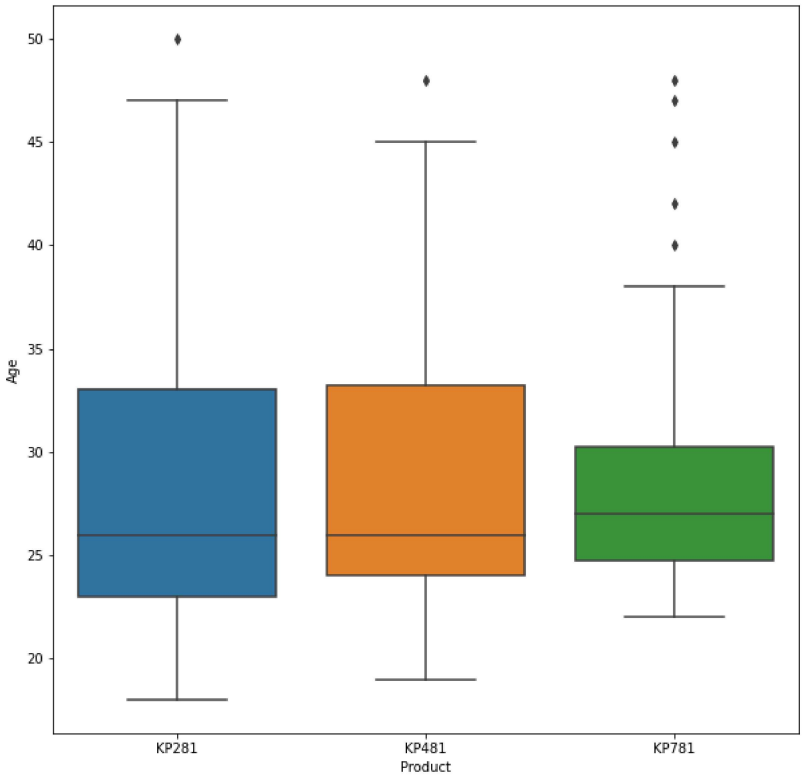
In [10]:
```
sns.pairplot(data)
plt.show()
# Observation: The Age, Education, Usage, Fitness,Income and Miles are all right skewed. This means they have an upper outlie
```

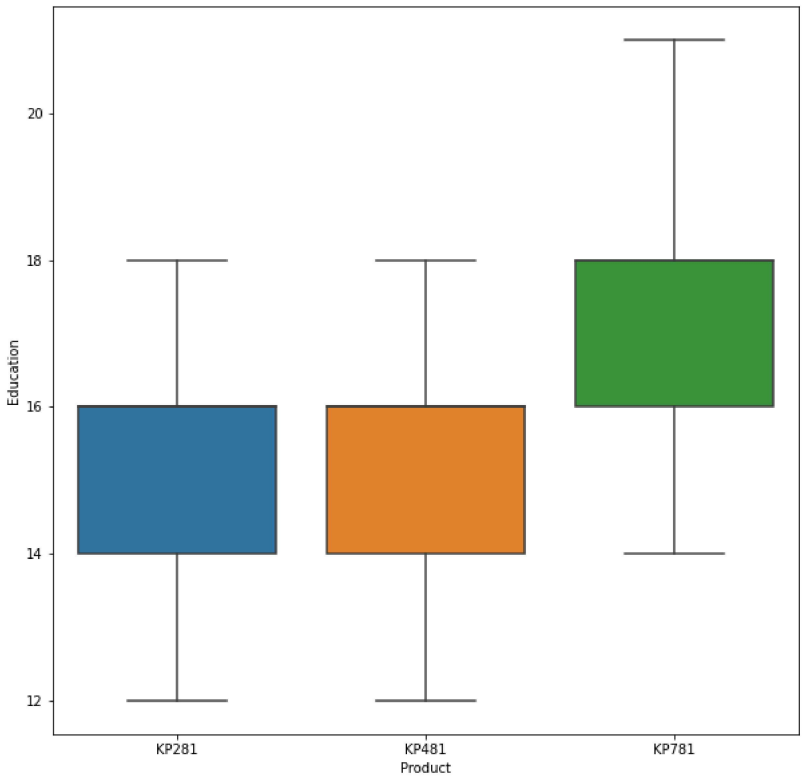
In [11]:
```
data.corr()
# As can be seen in the dataset, the following important interpretations are made.
# Usage and Fitness have a correlation od 0.66 (People who rate them self as fit tend to have higher frequency of usage for T
# Usage and Miles have a correlation of 0.75 (People who want to use Tradmill more frequently plan to run more miles on it.)
# Fitness and miles have a correlation of 0.78, (People who are fit tend to run more miles on Treadmill)
```

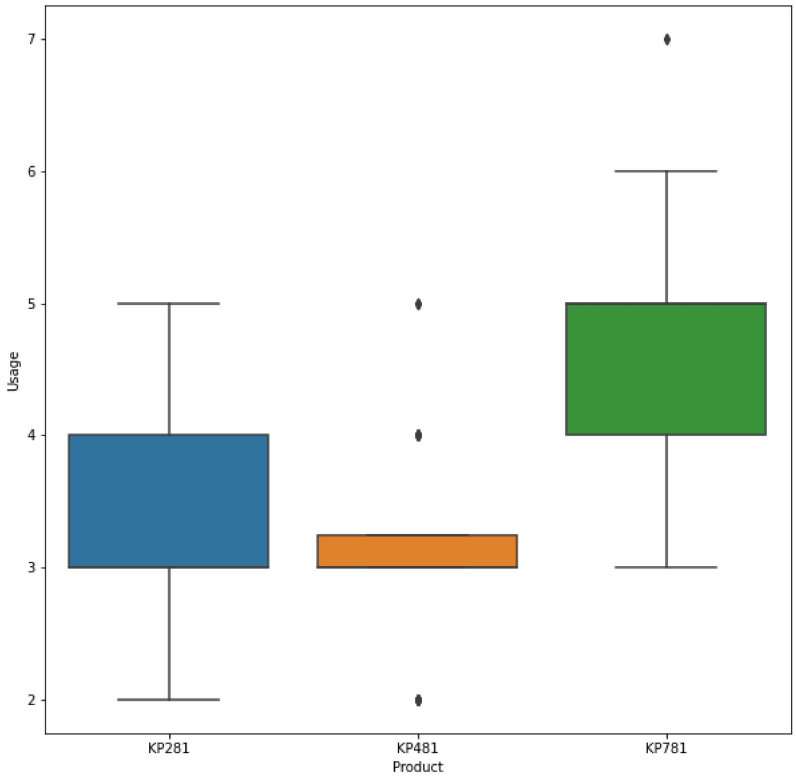
Out[11]:

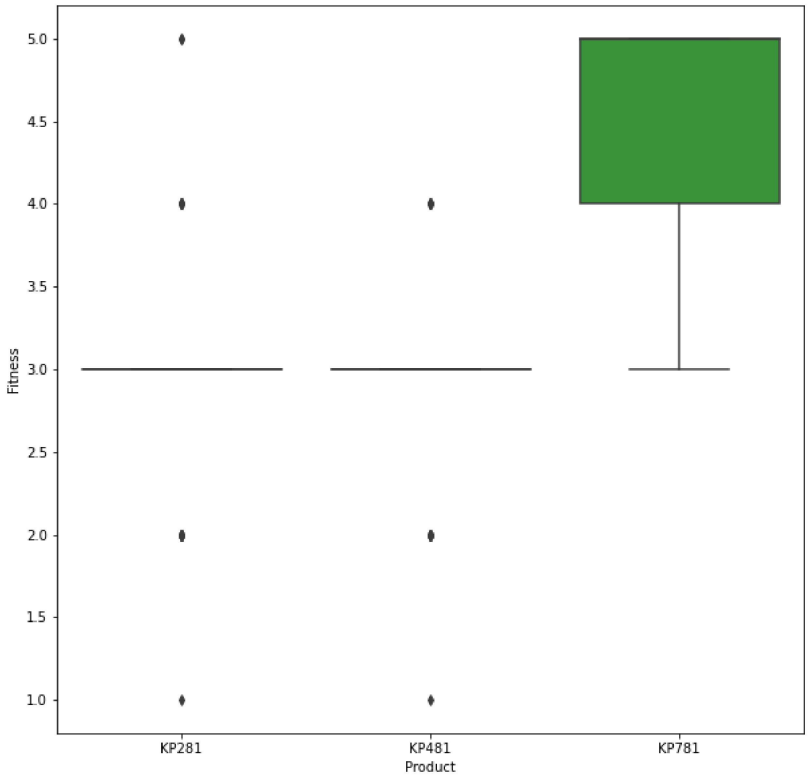
	Age	Education	Usage	Fitness	Income	Miles
Age	1.000000	0.280496	0.015064	0.061105	0.513414	0.036618
Education	0.280496	1.000000	0.395155	0.410581	0.625827	0.307284
Usage	0.015064	0.395155	1.000000	0.668606	0.519537	0.759130
Fitness	0.061105	0.410581	0.668606	1.000000	0.535005	0.785702
Income	0.513414	0.625827	0.519537	0.535005	1.000000	0.543473
Miles	0.036618	0.307284	0.759130	0.785702	0.543473	1.000000

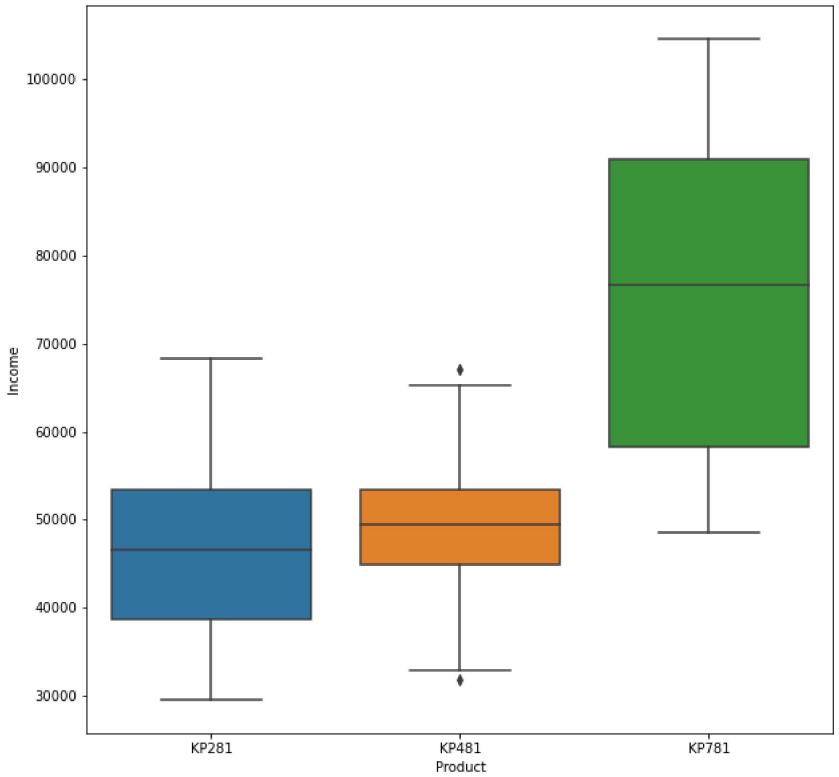
In [12]:
```
for i in z.columns:
    plt.figure(figsize = (10,10))
    sns.boxplot(y = data[i], x = data["Product"])
    plt.show
# Observation: (Age VS Model) We cannot use age alone as a factor for understanding the customer preferences for a particular
# box plot overlaps for all the three models. The age of customers who purchased KP281 has maximum variance and those who pur
# KP781 has minimum variance.
# (Education Vs Model): As can be seen from the data, 75 percent of customers who have purchased KP781 have education 16 Year
# more, where as approximately 25 percent of the customers who have purchased KP281 and KP481 have ducation above 16 Years.
# The customers who have purchased KP781 tend to be more educated as compared to those who purchased KP281 and KP481. The mir
# for customer who purchased KP781 is 14 yrs.
# 75 percent of the customers who purchased KP781 tend to use it 4 times or more in a week. Similarly, 75 percent of customer
# KP281 or KP481 tend to use it for 4 years or less.
# The minimum Fitness that a customer of KP781 rates self is 3.
# The minimum income of customer who has purchased KP781 is comparable to median income of customers who have purchased both
# KP281 and KP481. Income of KP781 customers is comparably higher than customers of KP281 and KP481.
# The Miles that a customer expects to run is comparably higher for KP781 as compared to KP281 and KP481.
```

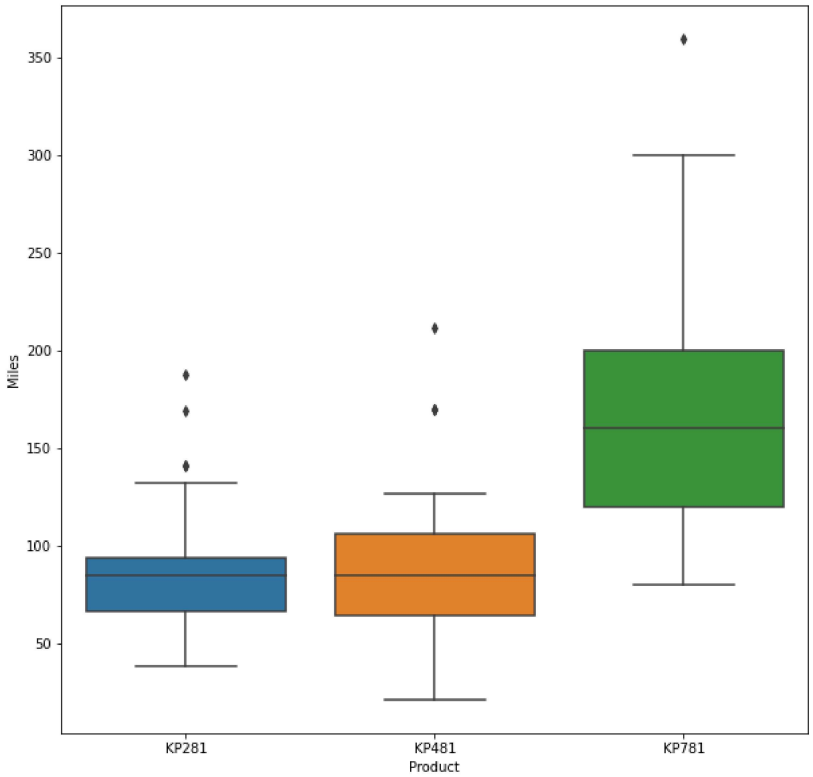





7
6
5
4
3
2
Usage
KP281
KP481
KP781
Product


5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Fitness
KP281
KP481
KP781
Product


100000
90000
80000
70000
60000
50000
40000
30000
Income
KP281
KP481
KP781
Product


350
300
250
200
150
100
50
Miles
KP281
KP481
KP781
Product

In [13]:
```
cond_prob = []
case = []
case_count = []
cond_count = []
j = np.quantile(data.loc[data["Product"] == "KP781", "Education"],[0.25])[0]
for i in data["Product"].unique():
    cped = len(data.loc[(data["Product"] == i) & (data["Education"] > j), ::])/len(data.loc[data["Education"] > j, ::])
    cond_prob.append(cped)
    case.append(f"P(Product = {i} | Education > {j})")
    case_count.append(len(data.loc[(data["Product"] == i) & (data["Education"] > j), ::]))
    cond_count.append(len(data.loc[data["Education"] > j, ::]))
    cped = len(data.loc[(data["Product"] == i) & (data["Education"] >= j), ::])/len(data.loc[data["Education"] >= j, ::])
    cond_prob.append(cped)
    case.append(f"P(Product = {i} | Education >= {j})")
    case_count.append(len(data.loc[(data["Product"] == i) & (data["Education"] >= j), ::]))
    cond_count.append(len(data.loc[data["Education"] >= j, ::]))

prob_dataframe = pd.DataFrame([case, cond_prob, case_count, cond_count]).T.rename(columns = {0:"Case",1: "Probability", 2:"Ca
prob_dataframe
# As can be seen from the data, 85 percent of consumers with Education greater than 16 years have purchased model KP781.
# Another interesing observation is, if we include 16 in the above probability, the probability of purchasing KP781 decreases
# to 0.33 when Education is greater than or equal to 16.
# If we calulate the no. of consumers with Education > 16, it is 27. While no. of customers with Education >= 16 is 112.
#print(len(data.loc[data["Education"] > j, ::]), len(data.loc[data["Education"] >= j, ::]))
```

Out[13]:

	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
5	P(Product = KP781 | Education >= 16.0)	0.339286	38	112

In [14]:
```
case = []
cond_prob = []
case_count = []
cond_count = []
for i in ["Usage","Fitness","Income","Miles"]:
    j = np.quantile(data.loc[data["Product"] == "KP781", i],[0.25])[0]
    for k in data["Product"].unique():
        cped = len(data.loc[(data["Product"] == k) & (data[i] > j), ::])/(len(data.loc[data[i] > j,::]))
        cond_prob.append(cped)
        case.append(f"P(Product = {k} | {i} > {j})")
        case_count.append(len(data.loc[(data["Product"] == k) & (data[i] > j), ::]))
        cond_count.append(len(data.loc[data[i] > j,::]))
        cped = len(data.loc[(data["Product"] == k) & (data[i] >= j), ::])/(len(data.loc[data[i] >= j,::]))
        cond_prob.append(cped)
        case.append(f"P(Product = {k} | {i} >= {j})")
        case_count.append(len(data.loc[(data["Product"] == k) & (data[i] >= j), ::]))
        cond_count.append(len(data.loc[data[i] >= j,::]))
prob_dataframe = prob_dataframe.append(pd.DataFrame([case, cond_prob,case_count,cond_count]).T.rename(columns = {0:"Case",1:'
prob_dataframe
# Observation: 80.7 % of users who wish to use the treadmill more than 4 times a week purchase KP781.
# 50% of users who wish to use the treadmill 4 or more times a week purchase KP281 and KP481.
# 93.5% of users who rate themselves at fitness greater than 4 have purchased KP781.
# 65.4% of users who rate themselves at fitness greater than or equal to 4 purchased KP781.
# 65.2% of users whose income is greater than 58205 purchased KP781
# 68.9% of users who expect to run 120 miles or more have purchased KP781
```

Out[14]:

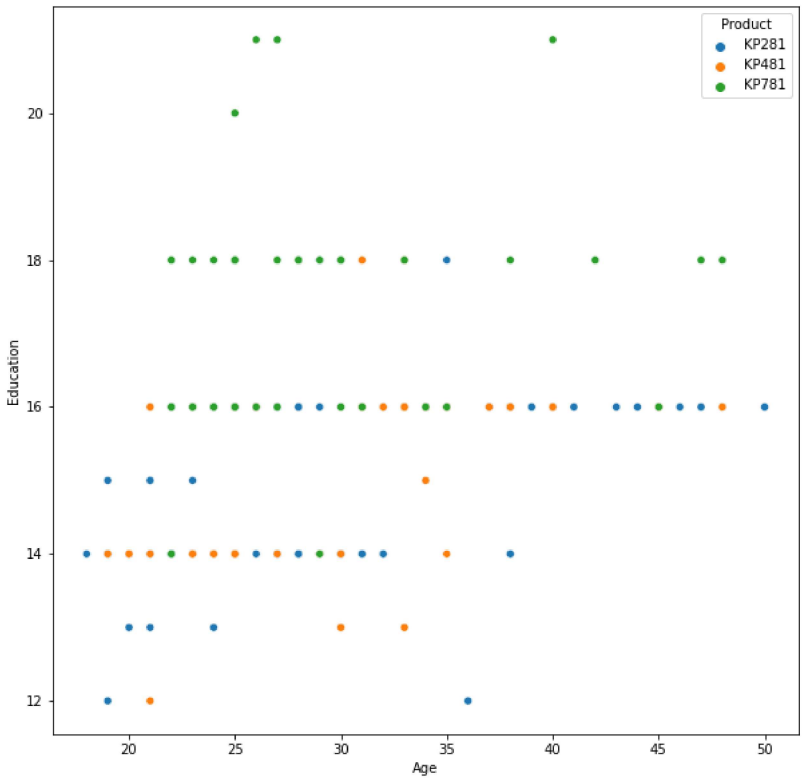
	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
5	P(Product = KP781 | Education >= 16.0)	0.339286	38	112
6	P(Product = KP281 | Usage > 4.0)	0.076923	2	26
7	P(Product = KP281 | Usage >= 4.0)	0.307692	24	78
8	P(Product = KP481 | Usage > 4.0)	0.115385	3	26
9	P(Product = KP481 | Usage >= 4.0)	0.192308	15	78
10	P(Product = KP781 | Usage > 4.0)	0.807692	21	26
11	P(Product = KP781 | Usage >= 4.0)	0.5	39	78
12	P(Product = KP281 | Fitness > 4.0)	0.064516	2	31
13	P(Product = KP281 | Fitness >= 4.0)	0.2	11	55
14	P(Product = KP481 | Fitness > 4.0)	0.0	0	31
15	P(Product = KP481 | Fitness >= 4.0)	0.145455	8	55
16	P(Product = KP781 | Fitness > 4.0)	0.935484	29	31
17	P(Product = KP781 | Fitness >= 4.0)	0.654545	36	55
18	P(Product = KP281 | Income > 58204.75)	0.152174	7	46
19	P(Product = KP281 | Income >= 58204.75)	0.152174	7	46
20	P(Product = KP481 | Income > 58204.75)	0.195652	9	46
21	P(Product = KP481 | Income >= 58204.75)	0.195652	9	46
22	P(Product = KP781 | Income > 58204.75)	0.652174	30	46
23	P(Product = KP781 | Income >= 58204.75)	0.652174	30	46
24	P(Product = KP281 | Miles > 120.0)	0.142857	6	42
25	P(Product = KP281 | Miles >= 120.0)	0.133333	6	45
26	P(Product = KP481 | Miles > 120.0)	0.190476	8	42
27	P(Product = KP481 | Miles >= 120.0)	0.177778	8	45
28	P(Product = KP781 | Miles > 120.0)	0.666667	28	42
29	P(Product = KP781 | Miles >= 120.0)	0.688889	31	45

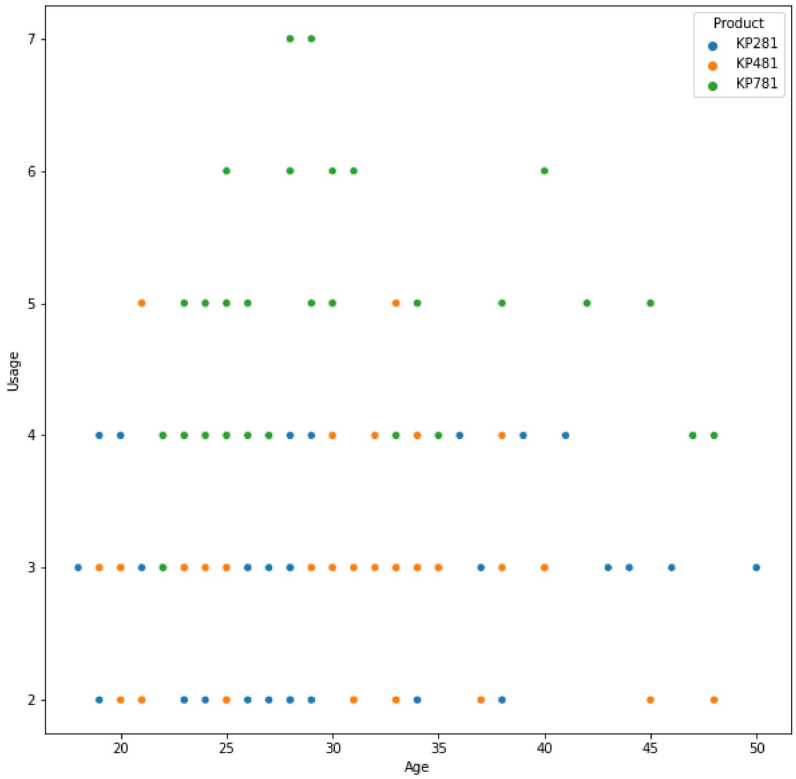
In [15]:
```
case = []
cond_prob = []
case_count = []
cond_count = []
for i in ["Gender", "MaritalStatus"]:
    for j in data[i].unique():
        for k in data["Product"].unique():
            cped = len(data.loc[(data[i] == j) & (data["Product"] == k)])/(len(data.loc[data[i] == j]))
            cond_prob.append(cped)
            case.append(f"P(Product = {k} | {i} = {j})")
            case_count.append(len(data.loc[(data[i] == j) & (data["Product"] == k)]))
            cond_count.append(len(data.loc[data[i] == j]))
prob_dataframe = prob_dataframe.append(pd.DataFrame([case, cond_prob,case_count,cond_count]).T.rename(columns = {0:"Case",1:'
prob_dataframe
# 52.6% of Female purchased KP281, whereas 38.4% of Male purchased KP281.
# 43.8% of customers with marital status Single purchased KP281.
# 44.8% of customers with marital status Partnered purchase KP281.
```

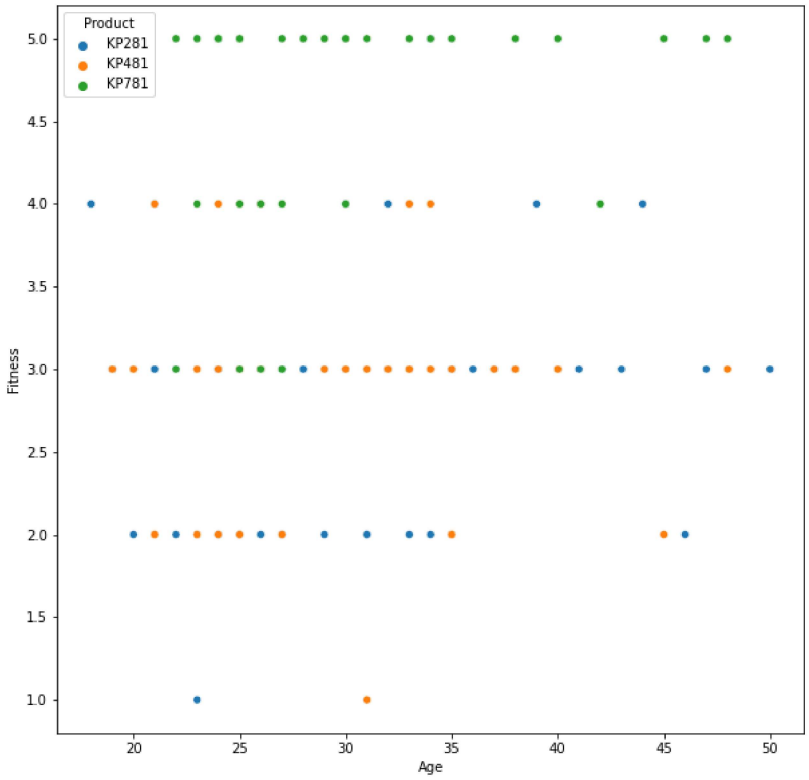
Out[15]:

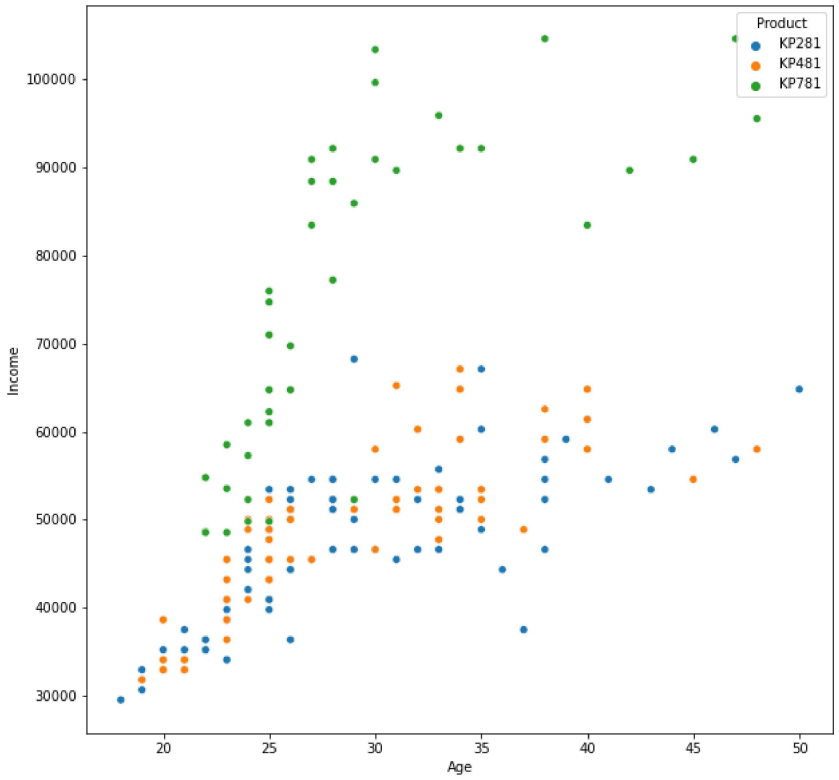
	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
5	P(Product = KP781 | Education >= 16.0)	0.339286	38	112
6	P(Product = KP281 | Usage > 4.0)	0.076923	2	26
7	P(Product = KP281 | Usage >= 4.0)	0.307692	24	78
8	P(Product = KP481 | Usage > 4.0)	0.115385	3	26
9	P(Product = KP481 | Usage >= 4.0)	0.192308	15	78
10	P(Product = KP781 | Usage > 4.0)	0.807692	21	26
11	P(Product = KP781 | Usage >= 4.0)	0.5	39	78
12	P(Product = KP281 | Fitness > 4.0)	0.064516	2	31
13	P(Product = KP281 | Fitness >= 4.0)	0.2	11	55
14	P(Product = KP481 | Fitness > 4.0)	0.0	0	31
15	P(Product = KP481 | Fitness >= 4.0)	0.145455	8	55
16	P(Product = KP781 | Fitness > 4.0)	0.935484	29	31
17	P(Product = KP781 | Fitness >= 4.0)	0.654545	36	55
18	P(Product = KP281 | Income > 58204.75)	0.152174	7	46
19	P(Product = KP281 | Income >= 58204.75)	0.152174	7	46
20	P(Product = KP481 | Income > 58204.75)	0.195652	9	46
21	P(Product = KP481 | Income >= 58204.75)	0.195652	9	46
22	P(Product = KP781 | Income > 58204.75)	0.652174	30	46
23	P(Product = KP781 | Income >= 58204.75)	0.652174	30	46
24	P(Product = KP281 | Miles > 120.0)	0.142857	6	42
25	P(Product = KP281 | Miles >= 120.0)	0.133333	6	45
26	P(Product = KP481 | Miles > 120.0)	0.190476	8	42
27	P(Product = KP481 | Miles >= 120.0)	0.177778	8	45
28	P(Product = KP781 | Miles > 120.0)	0.666667	28	42
29	P(Product = KP781 | Miles >= 120.0)	0.688889	31	45
30	P(Product = KP281 | Gender = Male)	0.384615	40	104
31	P(Product = KP481 | Gender = Male)	0.298077	31	104
32	P(Product = KP781 | Gender = Male)	0.317308	33	104
33	P(Product = KP281 | Gender = Female)	0.526316	40	76
34	P(Product = KP481 | Gender = Female)	0.381579	29	76
35	P(Product = KP781 | Gender = Female)	0.092105	7	76
36	P(Product = KP281 | MaritalStatus = Single)	0.438356	32	73
37	P(Product = KP481 | MaritalStatus = Single)	0.328767	24	73
38	P(Product = KP781 | MaritalStatus = Single)	0.232877	17	73
39	P(Product = KP281 | MaritalStatus = Partnered)	0.448598	48	107
40	P(Product = KP481 | MaritalStatus = Partnered)	0.336449	36	107
41	P(Product = KP781 | MaritalStatus = Partnered)	0.214953	23	107

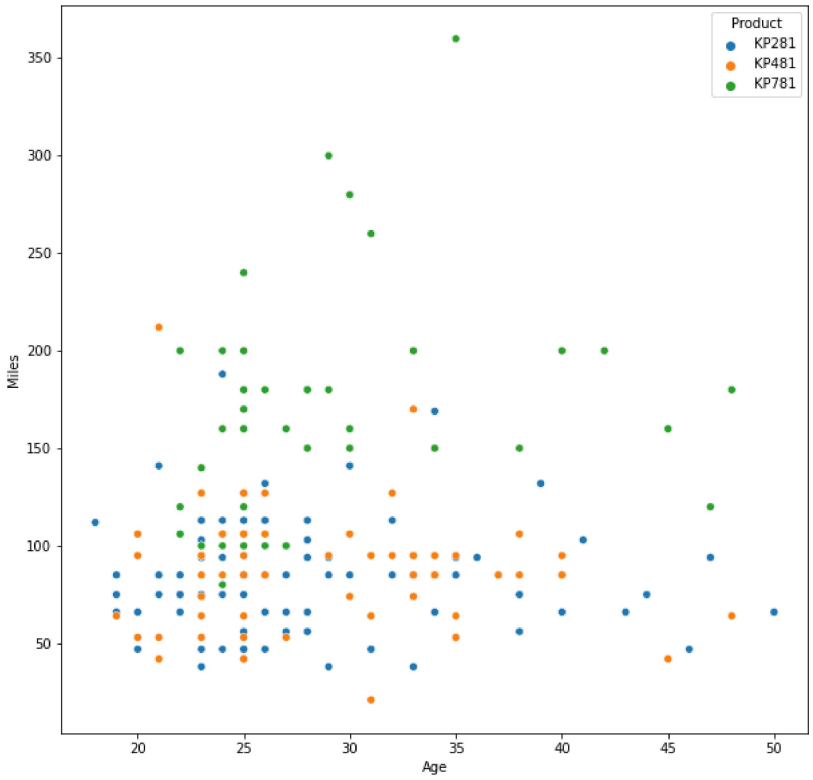
In [16]:
```
# Plotting scatterplots for all numrical data to identify any distinguishable customer characteristic for a particular produc
col = ["Age", "Education", "Usage", "Fitness", "Income", "Miles"]
for i in range(len(col)-1):
    for j in range(i+1, len(col)):
        plt.figure(figsize = (10,10))
        sns.scatterplot(x = data[col[i]], y = data[col[j]], hue = data["Product"])
        plt.show()
# Observation: Of the 15 graphs generated below, following conclusions are drawn and help in identifying potential customers
# for KP781
# From graph for Education Vs Age, it can be visualized that people with Education > 16 are more likely to buy KP781
# From graph for Usage Vs Age, it can be visualized that there is very low probability of customer buying KP781 if he wants t
# use the treadmill for less than 3 days. Similarly, if the usage is greater than 5 days, customer is very likely to buy KP78
# From graph for Fitness Vs Age, people who have a fitness rating > 4 are more likely to buy KP781 whereas those with fitness
# rating less than 4 are more likely to buy KP281 or KP481.
# From graph for Usage Vs Education, it can be concluded that customers with Usage > 4 and Education > 16 will purchase KP781
# sloping line can also be drawn on the same curve such that people who lie above the line will purchase KP871.
# From Graph for Income Vs Education, it can be concluded that if Income > 70000, the customer will most probably purchase KP
# If the income < 60000 and education <= 16 Years, the customer will purchase KP281 and KP481.
# From the graph for Miles Vs Income, a straight line can be drawn on the graph such that majority of people who lie above th
# use KP781 whereas people below the line use KP281 and KP481.
```

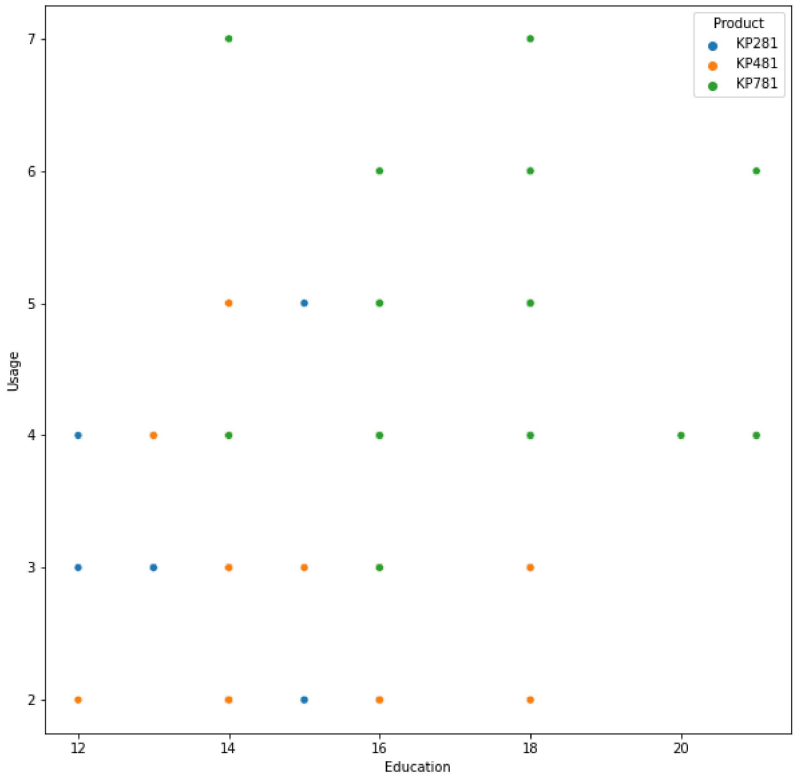


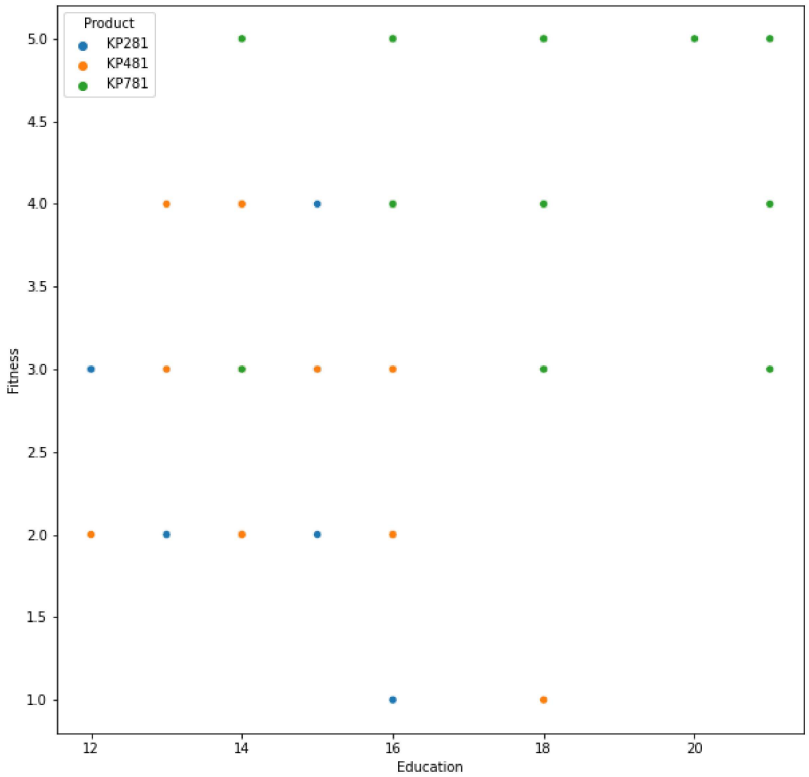

Product
KP281
KP481
KP781
Usage
Age
2
3
4
5
6
7
20
25
30
35
40
45
50

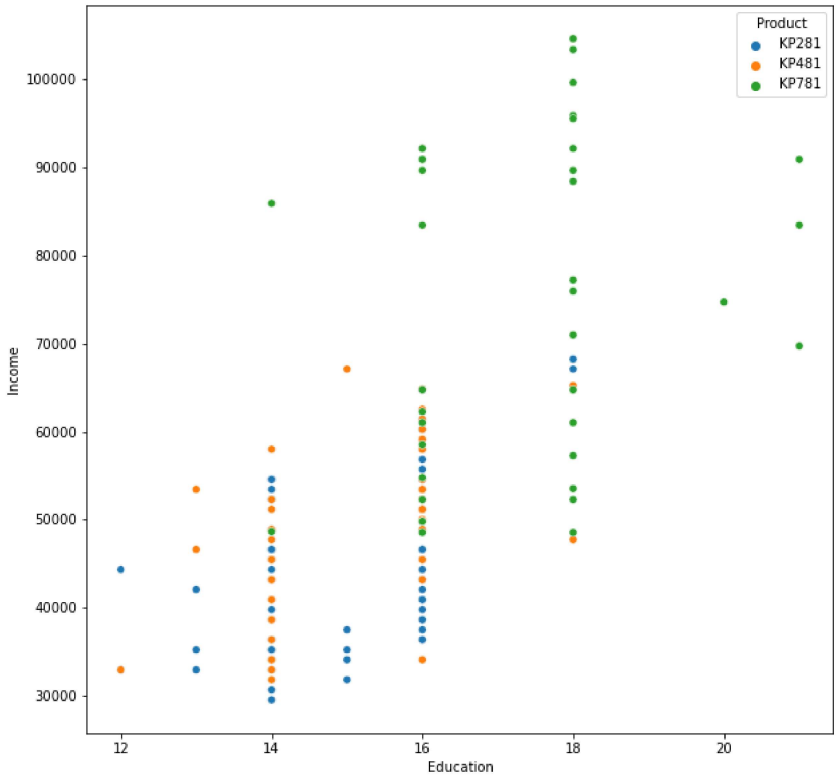

Product
KP281
KP481
KP781
Fitness
Age
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
20
25
30
35
40
45
50

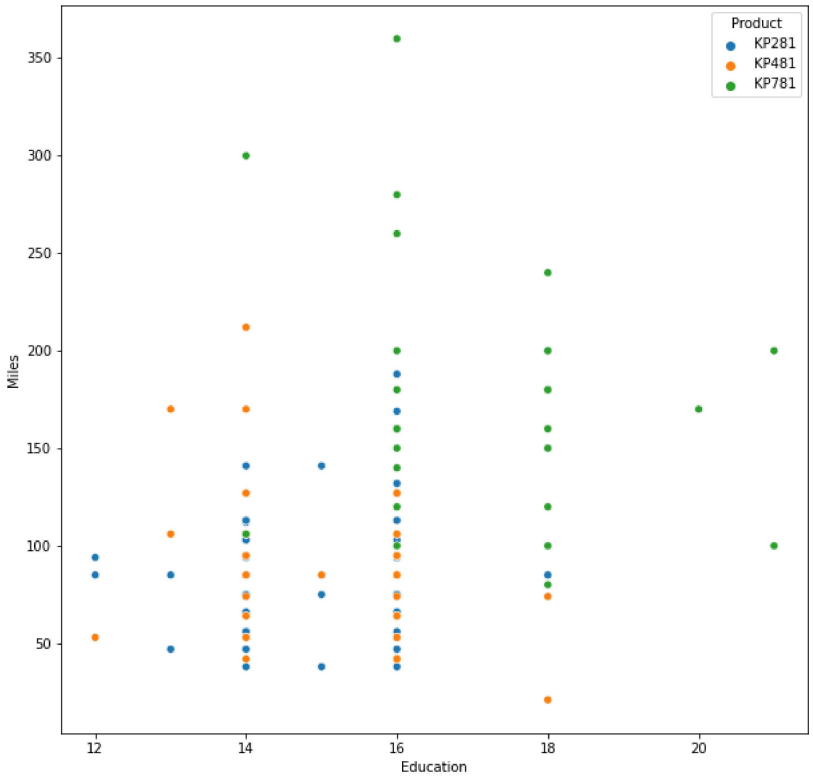

Product
KP281
KP481
KP781
Income
Age
100000
90000
80000
70000
60000
50000
40000
30000
20
25
30
35
40
45
50

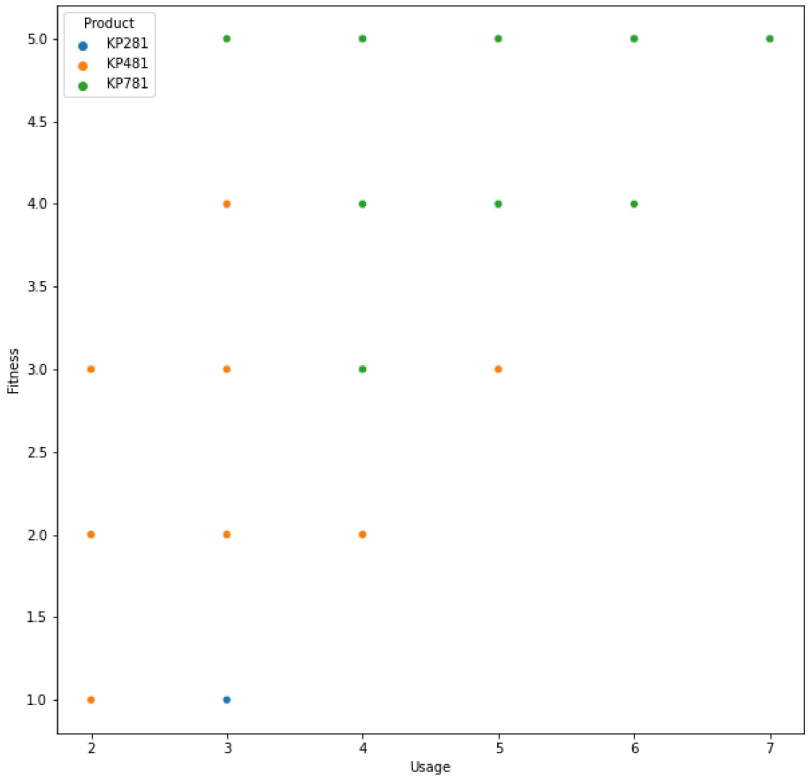

Product
KP281
KP481
KP781
Miles
Age
350
300
250
200
150
100
50
20
25
30
35
40
45
50

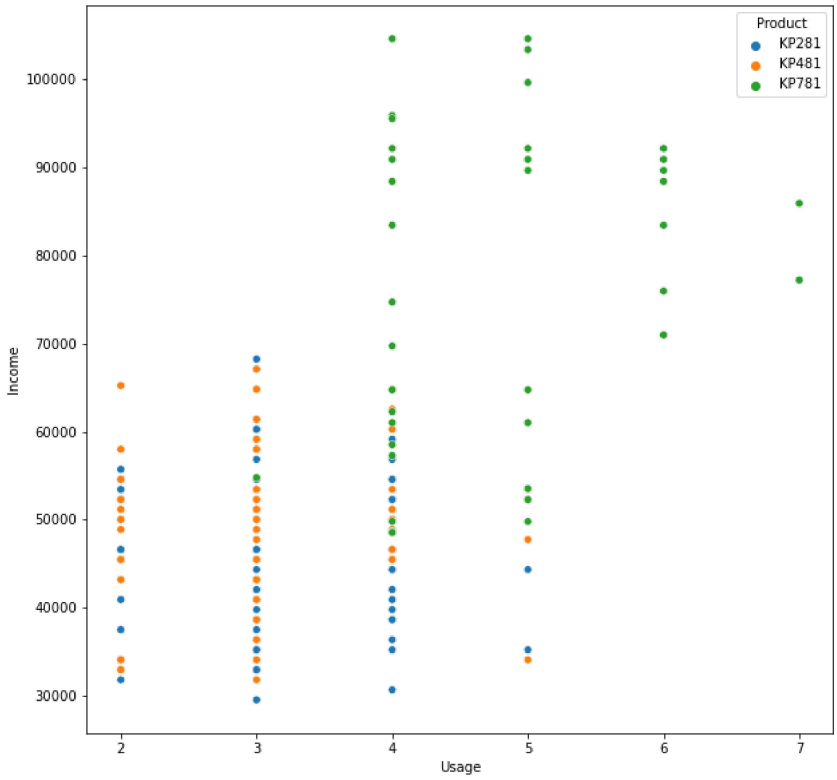

Product
KP281
KP481
KP781
Usage
Education
2
3
4
5
6
7
12
14
16
18
20

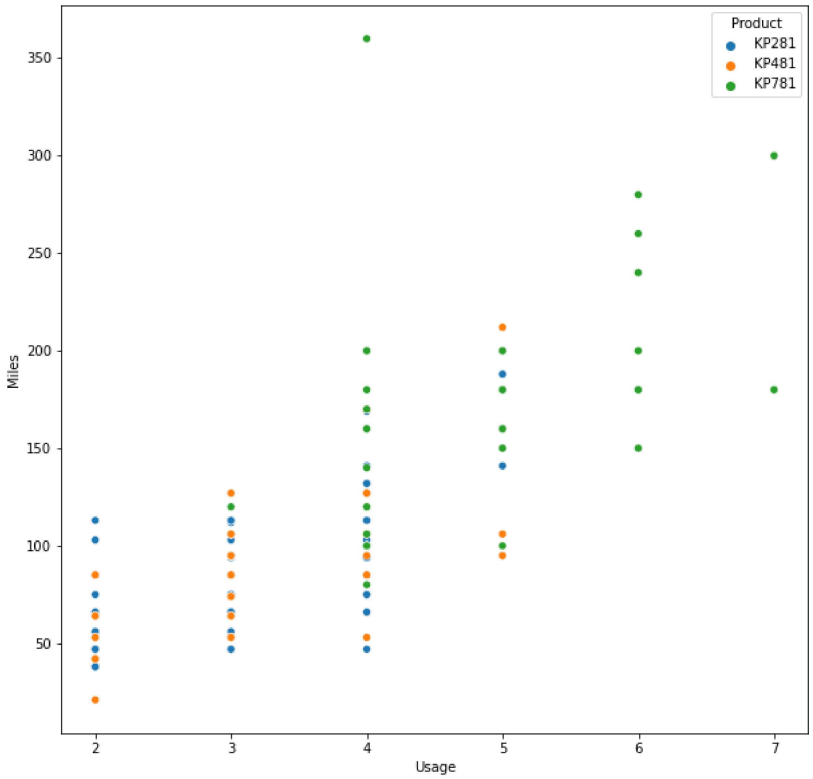

Product
KP281
KP481
KP781
Fitness
Education
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
12
14
16
18
20

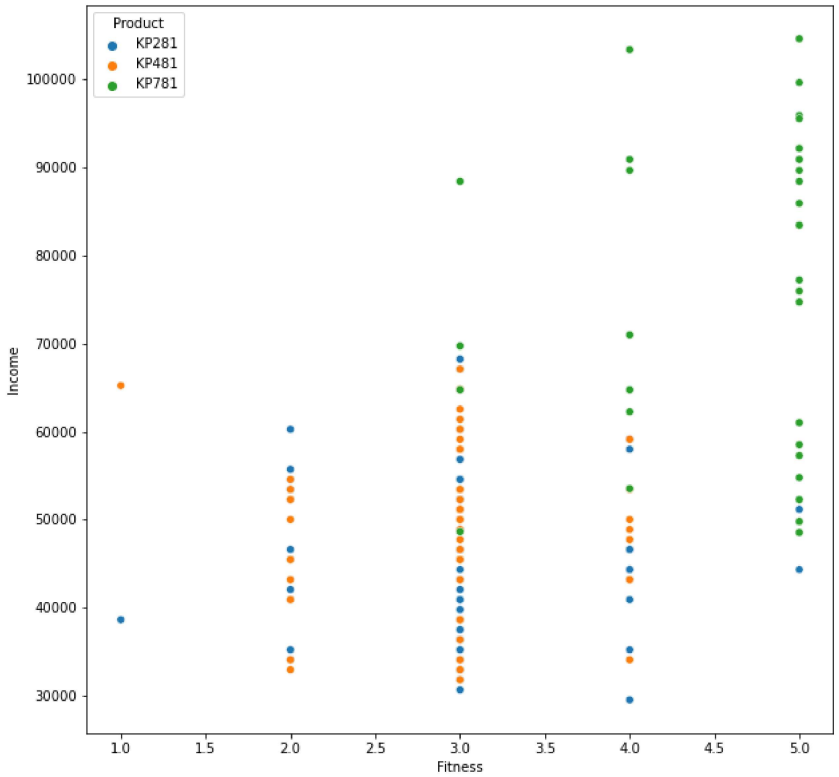

Product
KP281
KP481
KP781
Income
Education
100000
90000
80000
70000
60000
50000
40000
30000
12
14
16
18
20

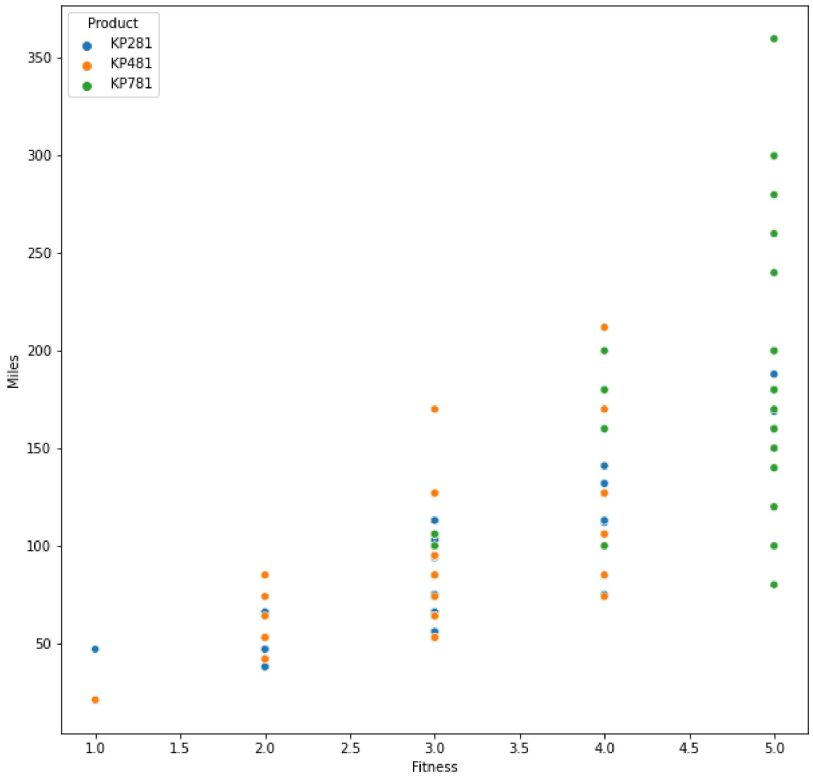

Product
KP281
KP481
KP781
Miles
Education
350
300
250
200
150
100
50
12
14
16
18
20

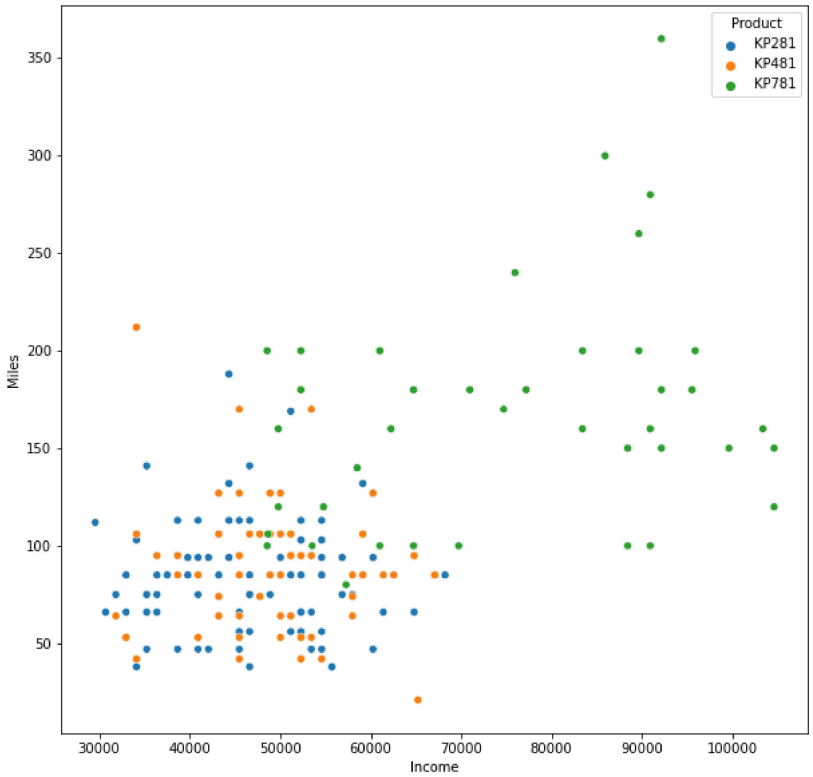

Product
KP281
KP481
KP781
Fitness
Usage
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
2
3
4
5
6
7


Product
KP281
KP481
KP781
Income
Usage
30000
40000
50000
60000
70000
80000
90000
100000
2
3
4
5
6
7

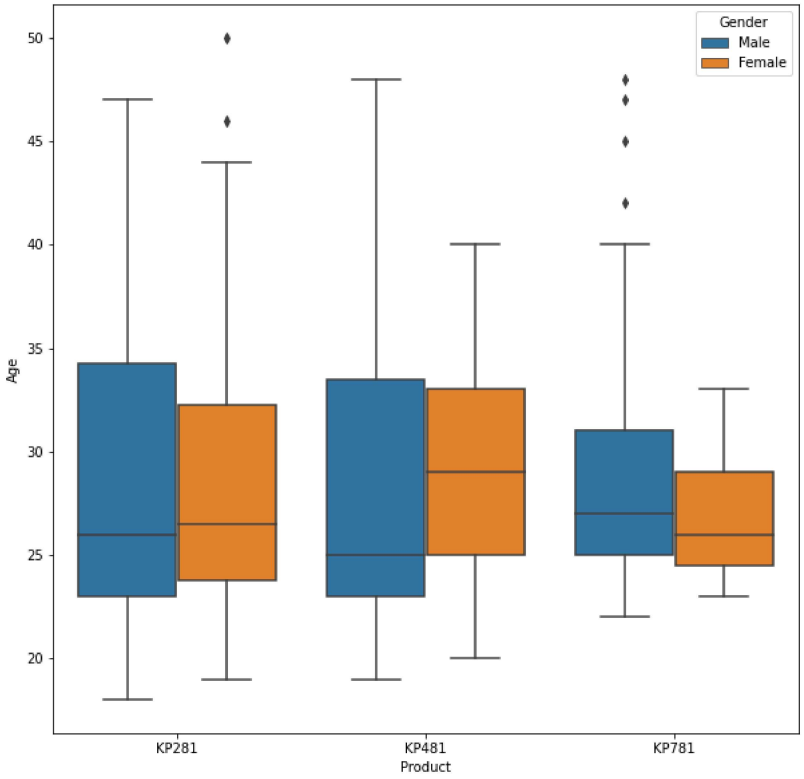

Product
KP281
KP481
KP781
Miles
Usage
350
300
250
200
150
100
50
2
3
4
5
6
7

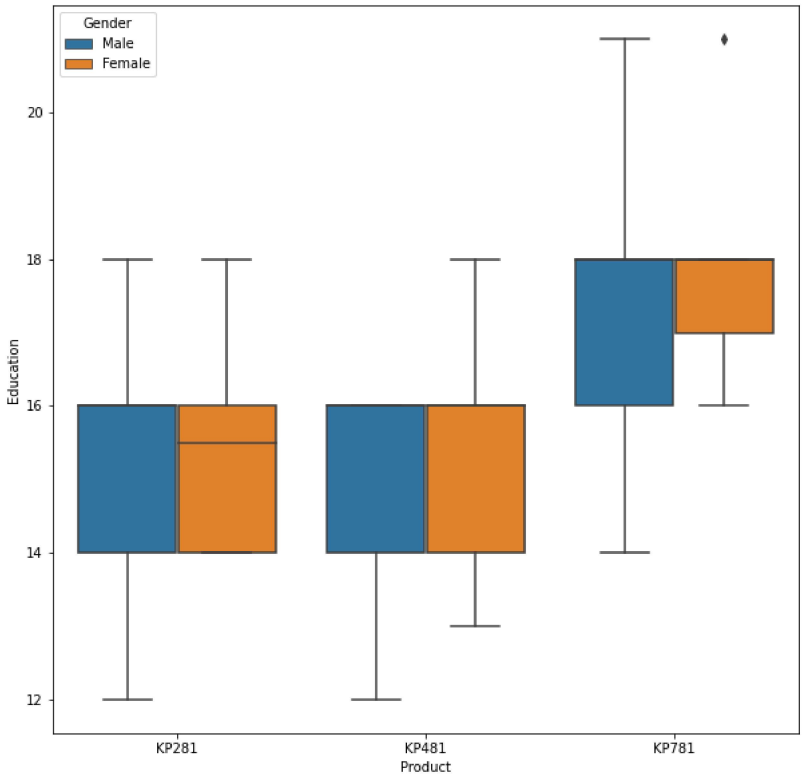

Product
KP281
KP481
KP781
Income
Fitness
100000
90000
80000
70000
60000
50000
40000
30000
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

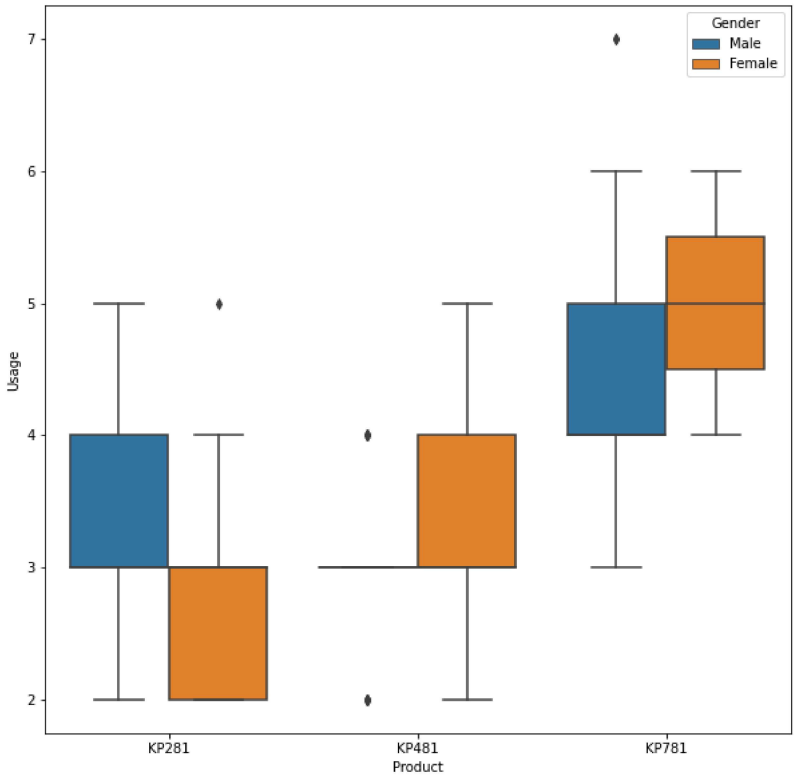

Product
KP281
KP481
KP781
Miles
Fitness
350
300
250
200
150
100
50
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

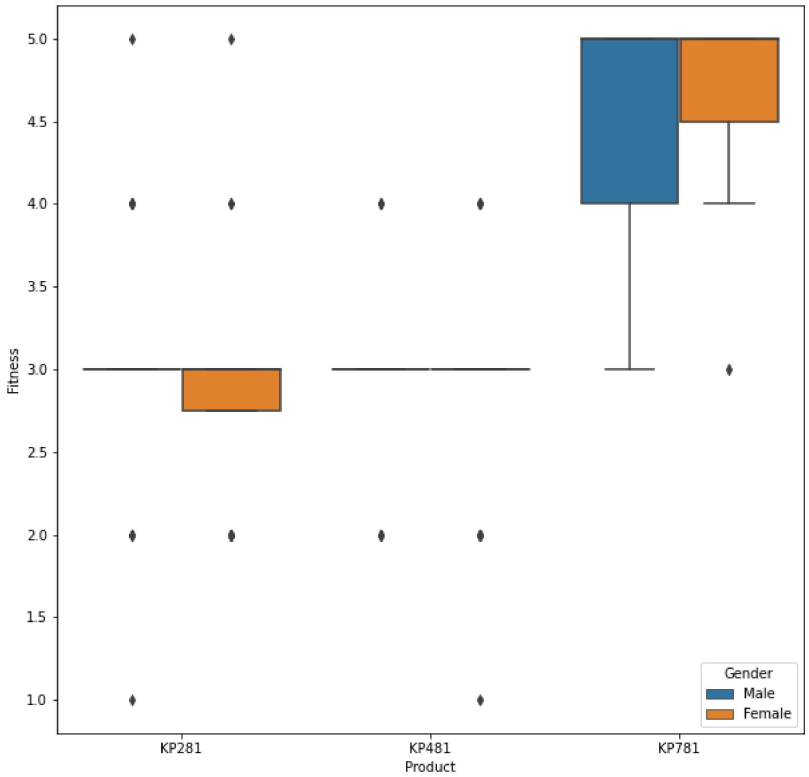

Product
KP281
KP481
KP781
Miles
Income
350
300
250
200
150
100
50
30000
40000
50000
60000
70000
80000
90000
100000

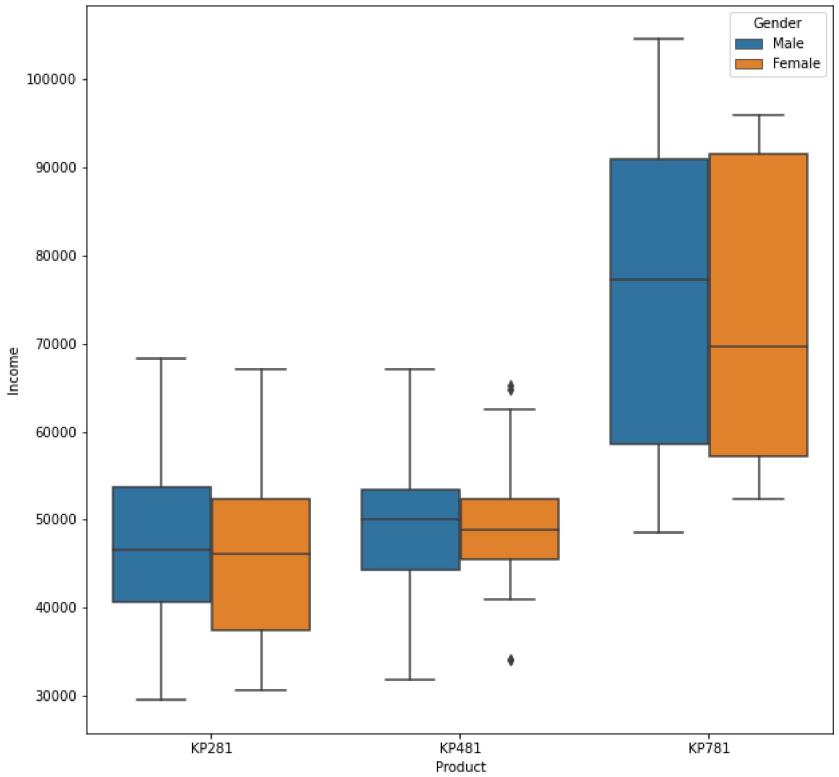
In [17]:
```
for i in z.columns:
    plt.figure(figsize = (10,10))
    sns.boxplot(y = data[i], x = data["Product"], hue = data["Gender"])
    plt.show
# From graph of Age Vs Product for both genders, we can conclude that age cannot be used to differentiate between customers d
# different genders for different products due to overlap between boxplots for different products.
# From graph for Education Vs Product for both genders, we can conclude that both Male and Female customers with education >
# have more chances of purchasing KP781.
# From the graph for Usage Vs Product, 75 percent of female customers who purchased KP281 tend to use it for 2 - 3 times a we
# 75 % of the Female who purchased KP481 tend to use it for 3 or more times a week. Similarly, all the females who purchased
# tend to use it for more than 4 times a week.
# Both Male and Female who want to use treadmill more than 5 times a week prefer to purchase KP781.
# Majority of the peopl who rate themslves 3 or more tend to purchase KP871.
# From the graph for Income Versus Product, all customers (Male and female) who have income greater than 55000 tend to purcha
# KP781.
# From the graph for Miles Versus Product, people with Miles > 100 prefer to purchase KP781 as compared to other two products
```

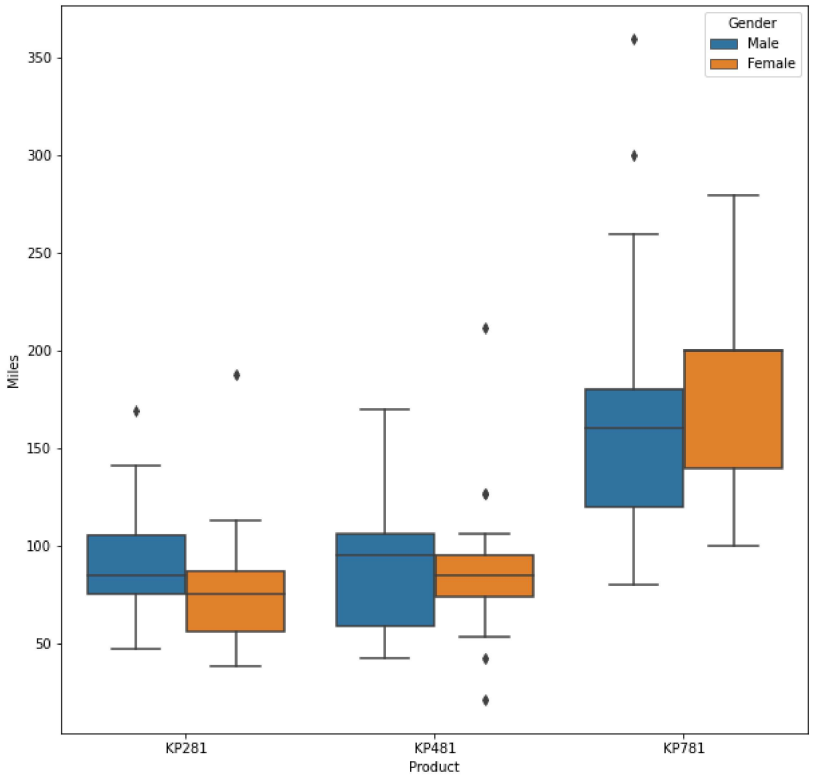




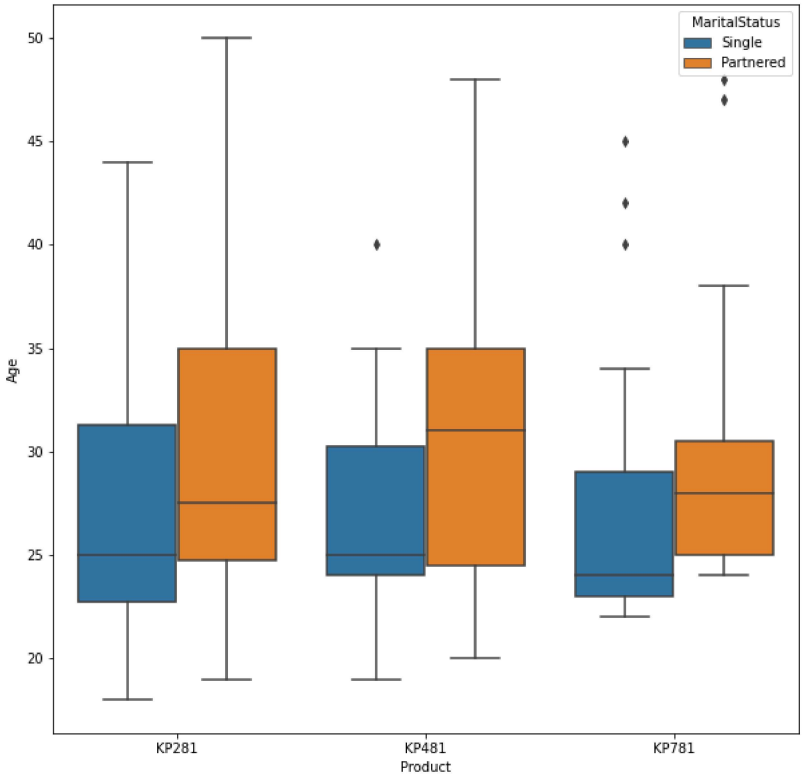

Gender
Male
Female
Usage
7
6
5
4
3
2
KP281
KP481
KP781
Product

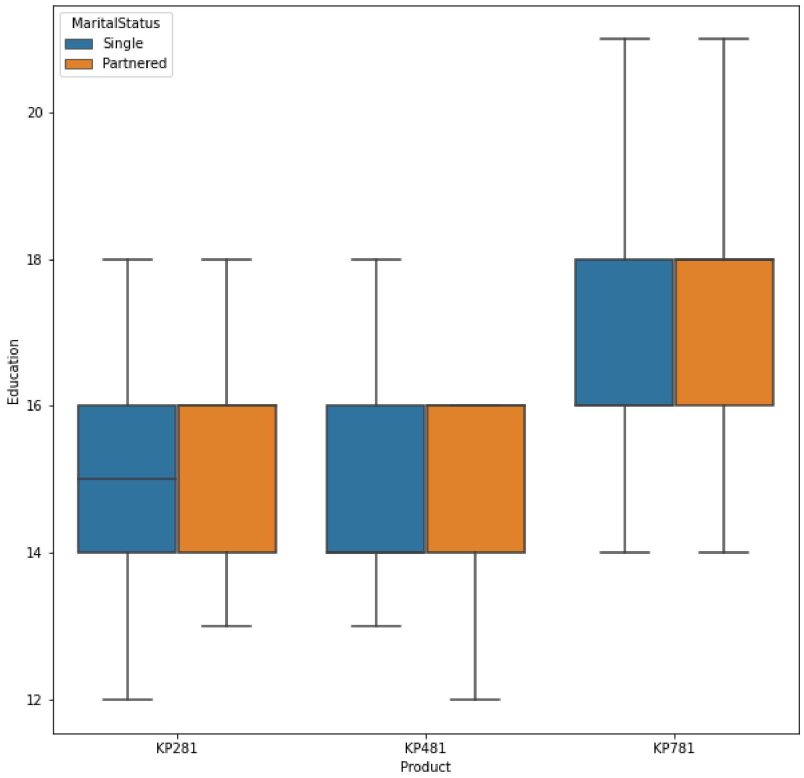

Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Gender
Male
Female
KP281
KP481
KP781
Product

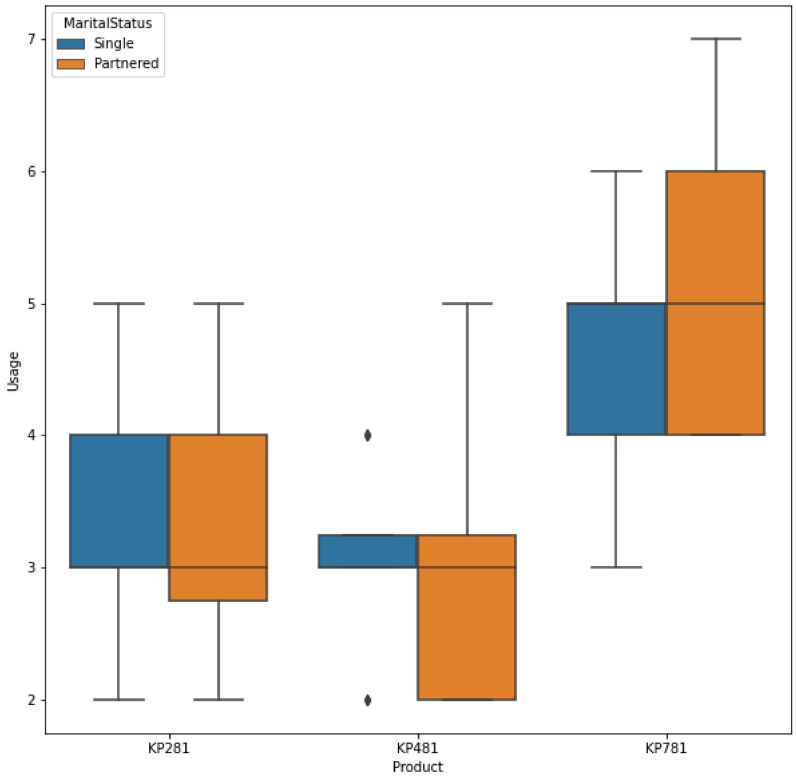

Gender
Male
Female
100000
90000
80000
70000
60000
50000
40000
30000
Income
KP281
KP481
KP781
Product

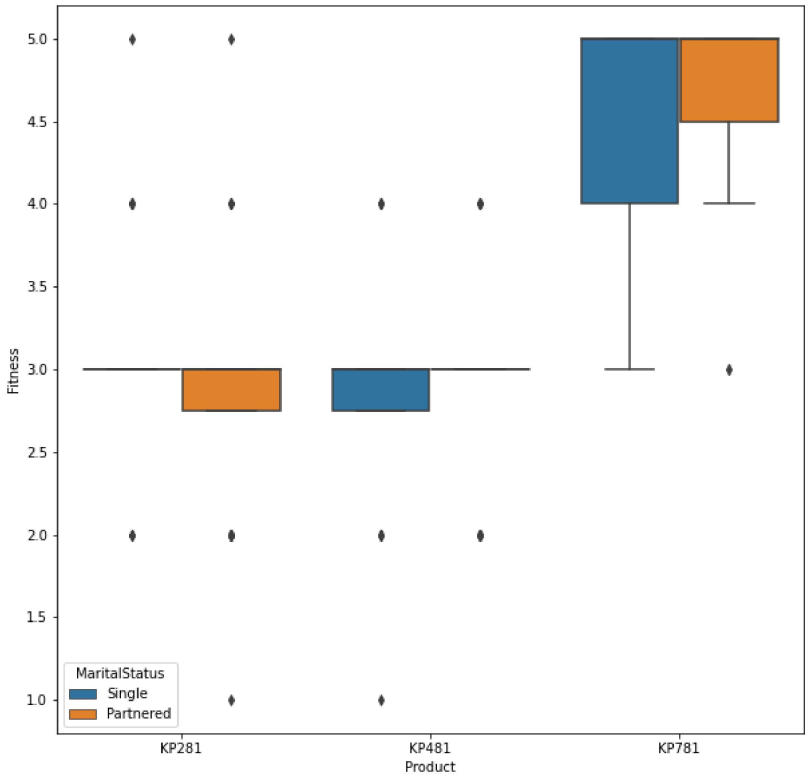

Gender
Male
Female
350
300
250
200
150
100
50
Miles
KP281
KP481
KP781
Product

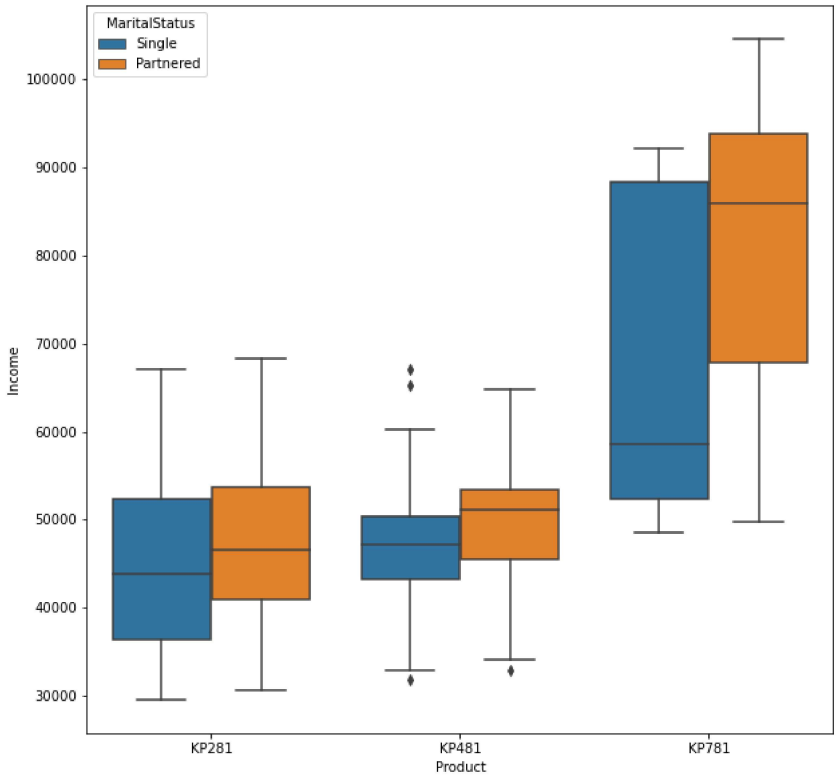
In [18]:
```
%matplotlib inline
for i in z.columns:
    plt.figure(figsize = (10,10))
    sns.boxplot(y = data[i], x = data["Product"], hue = data["MaritalStatus"])
    plt.show


# From graph for Age Vs Product for different Marital Status, we cannot draw much conclusion due to significant overlap betwe
# the boxplots One obvious observation is that Partnered customers tend to have more age as compared to Single ones.
# In the graph of Education Vs Produt, there is not much difference between Education years for KP281 and KP481, but for KP78
# 75% of users have education > 16 years.
# From graphs of Usage Vs Product,for KP781, 75 %  users of both Single and Partnered category tend to use the machine for at
# 4 times a week.
# From graph of income versus Product, The income of customers who purchase KP781 is higher as compared to income of users of
# KP281 and KP481 for both Single and Partnered customers.
# From graph of Miles versus Product, both the single and Partnered customers of KP781 tend to use it for longer miles as
# compared to other two Products.
```

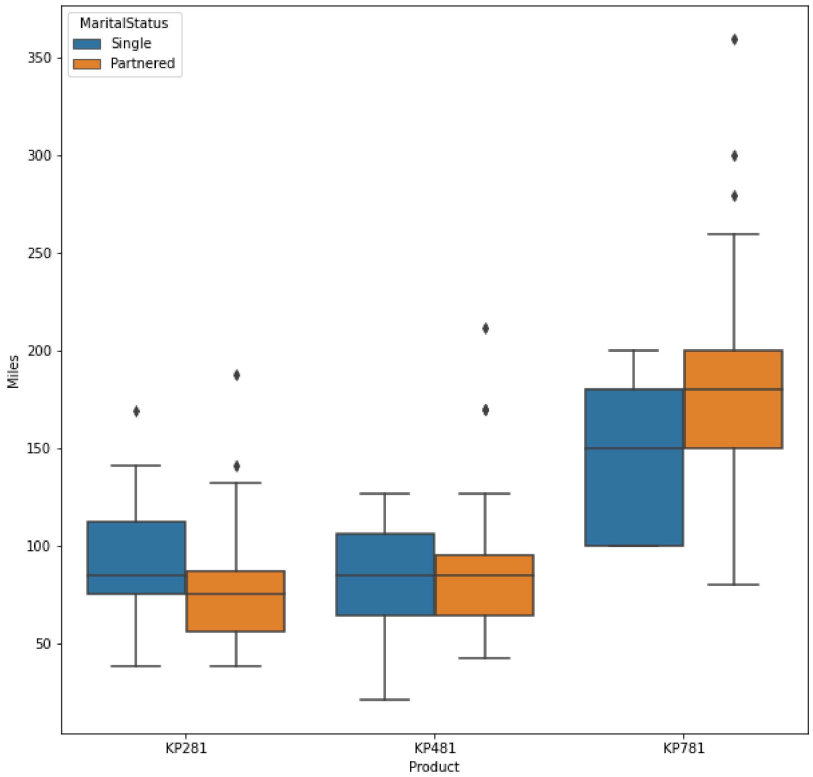





MaritalStatus
Single
Partnered
Usage
7
6
5
4
3
2
KP281
KP481
KP781
Product


Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
MaritalStatus
Single
Partnered
KP281
KP481
KP781
Product





In [19]:
```
def group(x):
    if (x["Gender"] == "Male") & (x["MaritalStatus"] == "Single"):
        return "MS"
    elif (x["Gender"] == "Male") & (x["MaritalStatus"] == "Partnered"):
        return "MP"
    elif (x["Gender"] == "Female") & (x["MaritalStatus"] == "Single"):
        return "FS"
    else:
        return "FP"
```

In [20]:
```
data["Combined_group"] = data.apply(group, axis = 1)
```

In [21]:
```
data
```

Out[21]:

	Product	Age	Gender	Education	MaritalStatus	Usage	Fitness	Income	Miles	Combined_group
0	KP281	18	Male	14	Single	3	4	29562	112	MS
1	KP281	19	Male	15	Single	2	3	31836	75	MS
2	KP281	19	Female	14	Partnered	4	3	30699	66	FP
3	KP281	19	Male	12	Single	3	3	32973	85	MS
4	KP281	20	Male	13	Partnered	4	2	35247	47	MP
...	...	...	...	...	...	...	...	...	...	...
175	KP781	40	Male	21	Single	6	5	83416	200	MS
176	KP781	42	Male	18	Single	5	4	89641	200	MS
177	KP781	45	Male	16	Single	5	5	90886	160	MS
178	KP781	47	Male	18	Partnered	4	5	104581	120	MP
179	KP781	48	Male	18	Partnered	4	5	95508	180	MP

180 rows × 10 columns

In [22]:
```
# Trying to understand if Gender and Product Purchased are independent using Chi square test:
chi2_contingency(pd.crosstab(data["Product"], data["Gender"]).values)
# As p value is less than 0.05, Gender and Product Purchased are not independent. Gender has some impact on Product purchased
# As seen from the contingency table, female have more preference for KP281 than expected.
# Similarly, Male have less preference for KP281 than expected.
# For KP781, females have less preference and males have more preference than expected.
```

Out[22]:
```
(12.923836032388664,
 0.0015617972833158714,
 2,
 array([[33.77777778, 46.22222222],
        [25.33333333, 34.66666667],
        [16.88888889, 23.11111111]]))
```

In [23]:
```
pd.crosstab(data["Product"], data["Gender"])
```

Out[23]:

Gender	Female	Male
Product		
KP281	40	40
KP481	29	31
KP781	7	33

In [24]:
```
pd.crosstab(data["Product"], data["MaritalStatus"])
```

Out[24]:

MaritalStatus	Partnered	Single
Product		
KP281	48	32
KP481	36	24
KP781	23	17

In [25]:
```
# Trying to understand if Product and Marital Status are independent using Chi square test:
chi2_contingency(pd.crosstab(data["Product"], data["MaritalStatus"]))
# As p value is 0.96, which is greater than 0.05, Product and Marital Status are independent.
```

Out[25]:
```
(0.0806554858532839,
 0.9604745988058153,
 2,
 array([[47.55555556, 32.44444444],
        [35.66666667, 24.33333333],
        [23.77777778, 16.22222222]]))
```

In [26]:
```
pd.crosstab(data["Product"], [data["Gender"], data["MaritalStatus"]])
```

Out[26]:

Gender	Female		Male	
MaritalStatus	Partnered	Single	Partnered	Single
Product				
KP281	27	13	21	19
KP481	15	14	21	10
KP781	4	3	19	14

In [27]:
```
# Trying to understand if Product is independent od Gender and Marital Satus combined
chi2_contingency(pd.crosstab(data["Product"], [data["Gender"], data["MaritalStatus"]]))
# As can be seen from table below, Female partnered candidates have more preferenec for KP281 than expected whereas Male part
# candidates have more preference for KP281 than expected.
```

Out[27]:
```
(16.485711680949464,
 0.011371287519110032,
 6,
 array([[20.44444444, 13.33333333, 27.11111111, 19.11111111],
        [15.33333333, 10.        , 20.33333333, 14.33333333],
        [10.22222222,  6.66666667, 13.55555556,  9.55555556]]))
```

In [28]:
```
g_ms_data = pd.DataFrame(pd.crosstab(data["Product"], [data["Gender"], data["MaritalStatus"]]))
col = []
for i in g_ms_data.columns:
    col.append (f"Gender = {i[0]}, Marital Status = {i[1]}")
g_ms_data.columns = col
g_ms_data
```

Out[28]:

Product	Gender = Female, Marital Status = Partnered	Gender = Female, Marital Status = Single	Gender = Male, Marital Status = Partnered	Gender = Male, Marital Status = Single
KP281	27	13	21	19
KP481	15	14	21	10
KP781	4	3	19	14

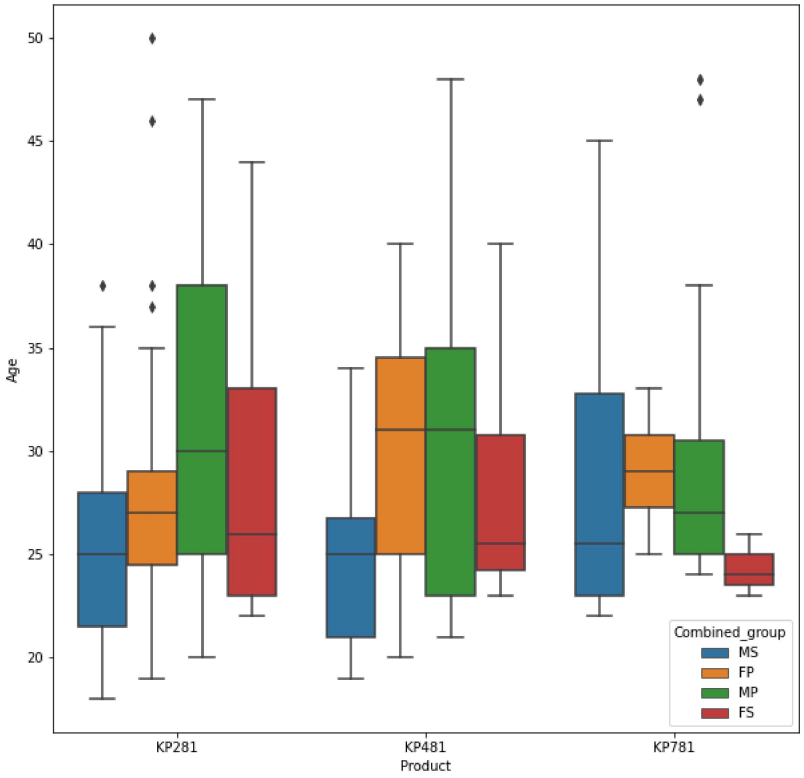
In [29]:
```
case = []
cond_prob = []
case_count = []
condition_count = []
for i in g_ms_data.index:
    for j in g_ms_data.columns:
        case.append(f"P(Product = {i} | {j})")
        cond_prob.append(g_ms_data.loc[i,j]/g_ms_data[j].sum())
        case_count.append(g_ms_data.loc[i,j])
        condition_count.append(g_ms_data[j].sum())

prob_dataframe = prob_dataframe.append(pd.DataFrame([case, cond_prob, case_count, condition_count]).T.rename(columns = {0:"Ca
```

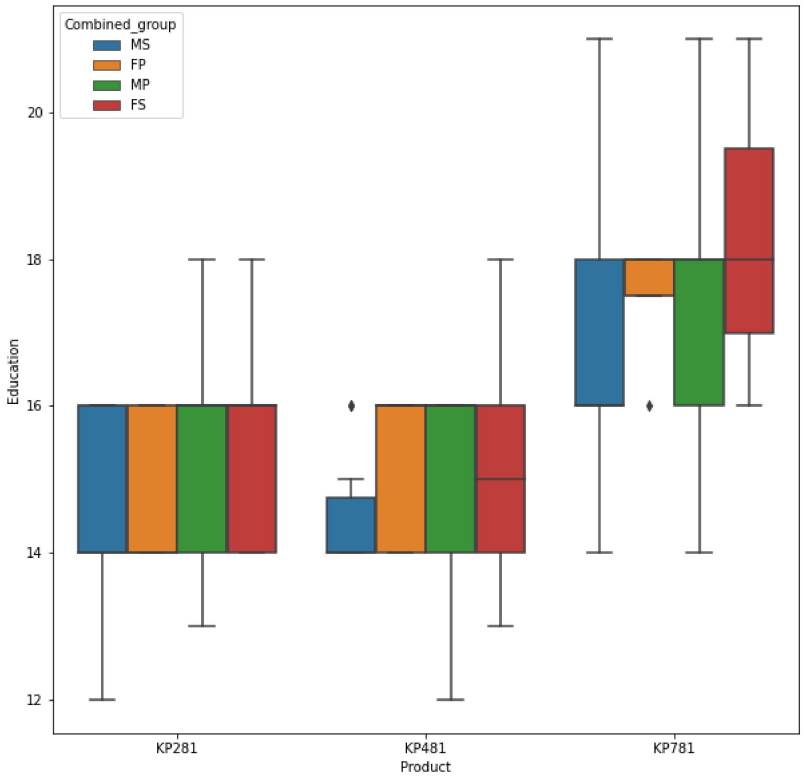
In [30]:
```
#prob_dataframe.to_csv("Prob_dataframe.csv")
prob_dataframe
```

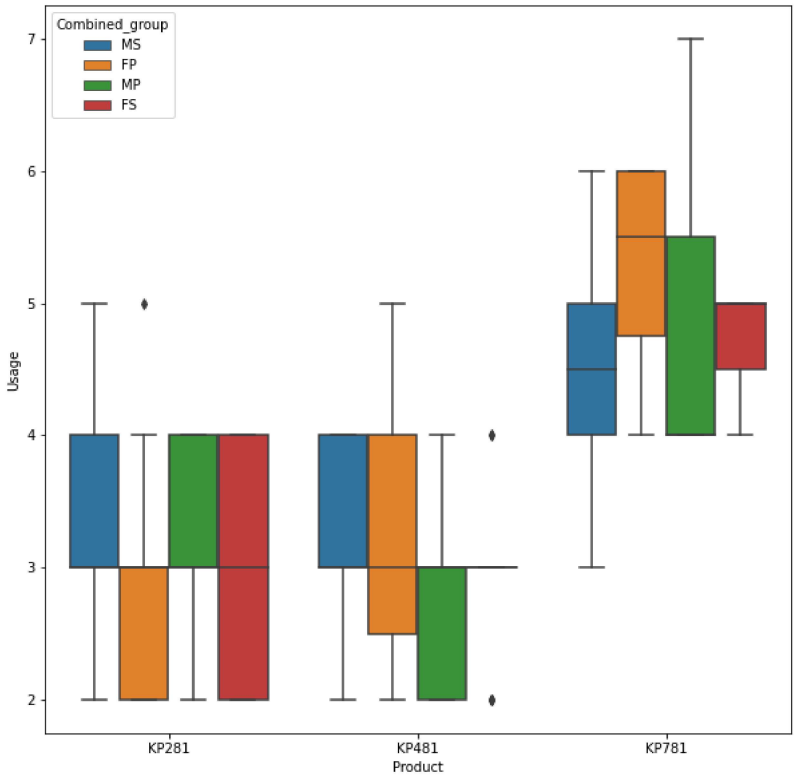
Out[30]:

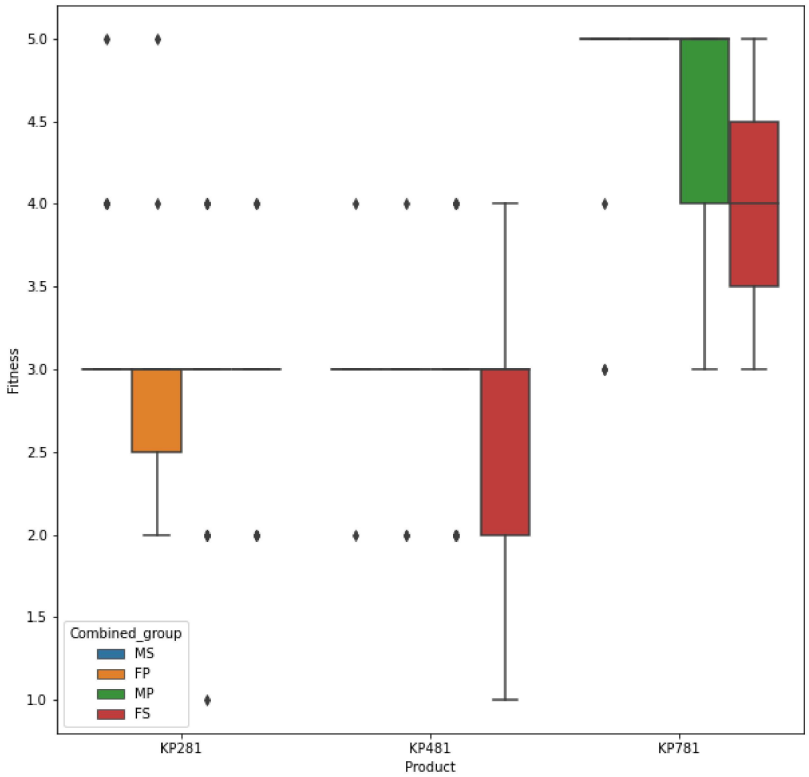
	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
5	P(Product = KP781 | Education >= 16.0)	0.339286	38	112
6	P(Product = KP281 | Usage > 4.0)	0.076923	2	26
7	P(Product = KP281 | Usage >= 4.0)	0.307692	24	78
8	P(Product = KP481 | Usage > 4.0)	0.115385	3	26
9	P(Product = KP481 | Usage >= 4.0)	0.192308	15	78
10	P(Product = KP781 | Usage > 4.0)	0.807692	21	26
11	P(Product = KP781 | Usage >= 4.0)	0.5	39	78
12	P(Product = KP281 | Fitness > 4.0)	0.064516	2	31
13	P(Product = KP281 | Fitness >= 4.0)	0.2	11	55
14	P(Product = KP481 | Fitness > 4.0)	0.0	0	31
15	P(Product = KP481 | Fitness >= 4.0)	0.145455	8	55
16	P(Product = KP781 | Fitness > 4.0)	0.935484	29	31
17	P(Product = KP781 | Fitness >= 4.0)	0.654545	36	55
18	P(Product = KP281 | Income > 58204.75)	0.152174	7	46
19	P(Product = KP281 | Income >= 58204.75)	0.152174	7	46
20	P(Product = KP481 | Income > 58204.75)	0.195652	9	46
21	P(Product = KP481 | Income >= 58204.75)	0.195652	9	46
22	P(Product = KP781 | Income > 58204.75)	0.652174	30	46
23	P(Product = KP781 | Income >= 58204.75)	0.652174	30	46
24	P(Product = KP281 | Miles > 120.0)	0.142857	6	42
25	P(Product = KP281 | Miles >= 120.0)	0.133333	6	45
26	P(Product = KP481 | Miles > 120.0)	0.190476	8	42
27	P(Product = KP481 | Miles >= 120.0)	0.177778	8	45
28	P(Product = KP781 | Miles > 120.0)	0.666667	28	42
29	P(Product = KP781 | Miles >= 120.0)	0.688889	31	45
30	P(Product = KP281 | Gender = Male)	0.384615	40	104
31	P(Product = KP481 | Gender = Male)	0.298077	31	104
32	P(Product = KP781 | Gender = Male)	0.317308	33	104
33	P(Product = KP281 | Gender = Female)	0.526316	40	76
34	P(Product = KP481 | Gender = Female)	0.381579	29	76
35	P(Product = KP781 | Gender = Female)	0.092105	7	76
36	P(Product = KP281 | MaritalStatus = Single)	0.438356	32	73
37	P(Product = KP481 | MaritalStatus = Single)	0.328767	24	73
38	P(Product = KP781 | MaritalStatus = Single)	0.232877	17	73
39	P(Product = KP281 | MaritalStatus = Partnered)	0.448598	48	107
40	P(Product = KP481 | MaritalStatus = Partnered)	0.336449	36	107
41	P(Product = KP781 | MaritalStatus = Partnered)	0.214953	23	107
42	P(Product = KP281 | Gender = Female, Marital S...	0.586957	27	46
43	P(Product = KP281 | Gender = Female, Marital S...	0.433333	13	30
44	P(Product = KP281 | Gender = Male, Marital Sta...	0.344262	21	61
45	P(Product = KP281 | Gender = Male, Marital Sta...	0.44186	19	43
46	P(Product = KP481 | Gender = Female, Marital S...	0.326087	15	46
47	P(Product = KP481 | Gender = Female, Marital S...	0.466667	14	30
48	P(Product = KP481 | Gender = Male, Marital Sta...	0.344262	21	61
49	P(Product = KP481 | Gender = Male, Marital Sta...	0.232558	10	43
50	P(Product = KP781 | Gender = Female, Marital S...	0.086957	4	46
51	P(Product = KP781 | Gender = Female, Marital S...	0.1	3	30
52	P(Product = KP781 | Gender = Male, Marital Sta...	0.311475	19	61
53	P(Product = KP781 | Gender = Male, Marital Sta...	0.325581	14	43

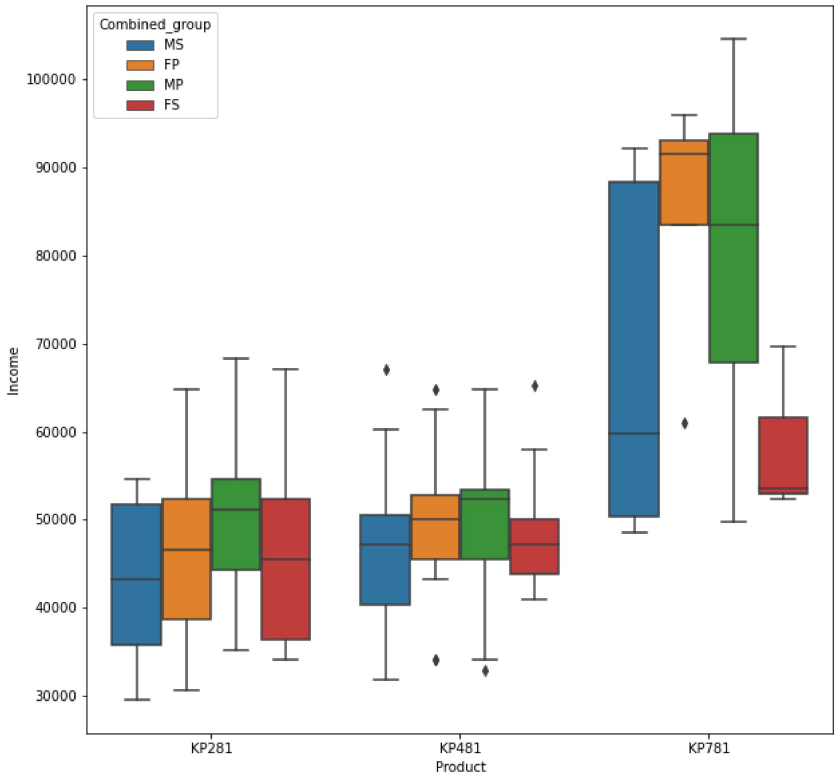
In [33]:
```
for i in ['Age','Education','Usage','Fitness', 'Income', 'Miles']:
    plt.figure(figsize = (10,10))
    sns.boxplot(x = data["Product"], y = data[i], hue = data["Combined_group"])
# From the prob_dataframe, 58.6% of Female Partnered candidatescustomers have purchased KP281. From age versus Product graph,
# this group ("FP") tends to have a lower median age for KP281 product as compared to KP781 and KP481 product.
# From the graph for Education Versus Product, all the four categories female Partnered,Female Single, Male Partnered, Male S
# tend to have higher education for product KP781 as compared to Product KP281 and KP481
# From graph of Usage Verus Product, 75 % of Male Partnered customers of KP281 tend to use 3-4 times in a week, whereas
# 75 percent of Male Partnered customers of KP481 tend to use it for 2-3 times a week. The minimum usage for all categories
# for KP781 is 3 times a week. All other categories except Male single in KP781 product tend to use it 4 or more times a week
# From graph of Fitness versus Product, all the four categories of customers for KP781 rate themselves 3 or above in Fitness.
# From graphs of Miles versus Product and Income Versus product, all four groups for KP781 have higher values of miles and
# Income as compared to other two Product groups.
```

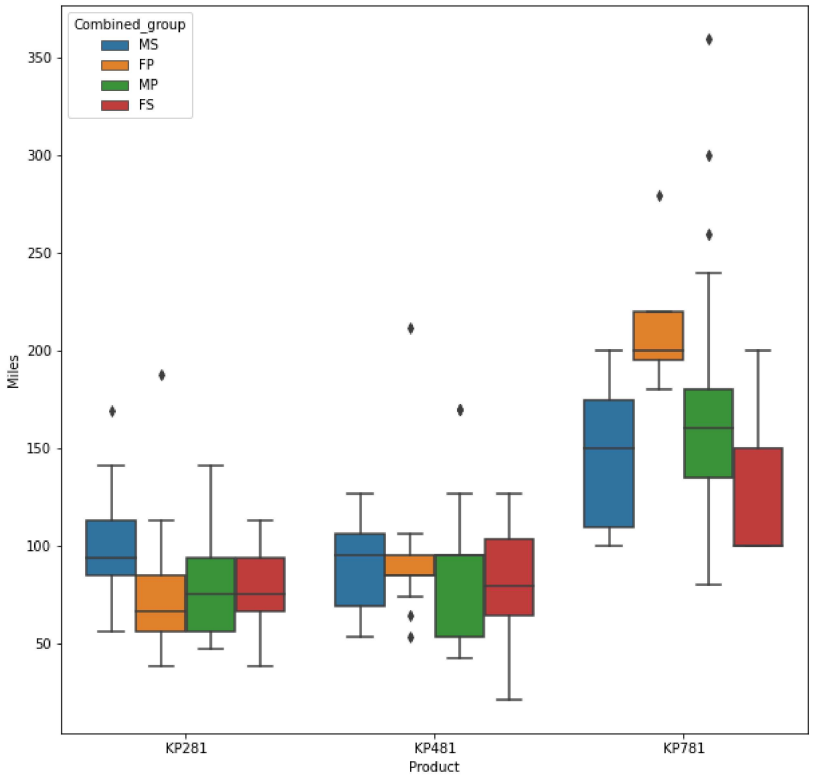





Combined_group
MS
FP
MP
FS
Usage
7
6
5
4
3
2
KP281
KP481
KP781
Product


Fitness
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Combined_group
MS
FP
MP
FS
KP281
KP481
KP781
Product


Combined_group
MS
FP
MP
FS
Income
100000
90000
80000
70000
60000
50000
40000
30000
KP281
KP481
KP781
Product


Combined_group
MS
FP
MP
FS
Miles
350
300
250
200
150
100
50
KP281
KP481
KP781
Product

In [34]:
```
prob_dataframe
```

Out[34]:

	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
5	P(Product = KP781 | Education >= 16.0)	0.339286	38	112
6	P(Product = KP281 | Usage > 4.0)	0.076923	2	26
7	P(Product = KP281 | Usage >= 4.0)	0.307692	24	78
8	P(Product = KP481 | Usage > 4.0)	0.115385	3	26
9	P(Product = KP481 | Usage >= 4.0)	0.192308	15	78
10	P(Product = KP781 | Usage > 4.0)	0.807692	21	26
11	P(Product = KP781 | Usage >= 4.0)	0.5	39	78
12	P(Product = KP281 | Fitness > 4.0)	0.064516	2	31
13	P(Product = KP281 | Fitness >= 4.0)	0.2	11	55
14	P(Product = KP481 | Fitness > 4.0)	0.0	0	31
15	P(Product = KP481 | Fitness >= 4.0)	0.145455	8	55
16	P(Product = KP781 | Fitness > 4.0)	0.935484	29	31
17	P(Product = KP781 | Fitness >= 4.0)	0.654545	36	55
18	P(Product = KP281 | Income > 58204.75)	0.152174	7	46
19	P(Product = KP281 | Income >= 58204.75)	0.152174	7	46
20	P(Product = KP481 | Income > 58204.75)	0.195652	9	46
21	P(Product = KP481 | Income >= 58204.75)	0.195652	9	46
22	P(Product = KP781 | Income > 58204.75)	0.652174	30	46
23	P(Product = KP781 | Income >= 58204.75)	0.652174	30	46
24	P(Product = KP281 | Miles > 120.0)	0.142857	6	42
25	P(Product = KP281 | Miles >= 120.0)	0.133333	6	45
26	P(Product = KP481 | Miles > 120.0)	0.190476	8	42
27	P(Product = KP481 | Miles >= 120.0)	0.177778	8	45
28	P(Product = KP781 | Miles > 120.0)	0.666667	28	42
29	P(Product = KP781 | Miles >= 120.0)	0.688889	31	45
30	P(Product = KP281 | Gender = Male)	0.384615	40	104
31	P(Product = KP481 | Gender = Male)	0.298077	31	104
32	P(Product = KP781 | Gender = Male)	0.317308	33	104
33	P(Product = KP281 | Gender = Female)	0.526316	40	76
34	P(Product = KP481 | Gender = Female)	0.381579	29	76
35	P(Product = KP781 | Gender = Female)	0.092105	7	76
36	P(Product = KP281 | MaritalStatus = Single)	0.438356	32	73
37	P(Product = KP481 | MaritalStatus = Single)	0.328767	24	73
38	P(Product = KP781 | MaritalStatus = Single)	0.232877	17	73
39	P(Product = KP281 | MaritalStatus = Partnered)	0.448598	48	107
40	P(Product = KP481 | MaritalStatus = Partnered)	0.336449	36	107
41	P(Product = KP781 | MaritalStatus = Partnered)	0.214953	23	107
42	P(Product = KP281 | Gender = Female, Marital S...	0.586957	27	46
43	P(Product = KP281 | Gender = Female, Marital S...	0.433333	13	30
44	P(Product = KP281 | Gender = Male, Marital Sta...	0.344262	21	61
45	P(Product = KP281 | Gender = Male, Marital Sta...	0.44186	19	43
46	P(Product = KP481 | Gender = Female, Marital S...	0.326087	15	46
47	P(Product = KP481 | Gender = Female, Marital S...	0.466667	14	30
48	P(Product = KP481 | Gender = Male, Marital Sta...	0.344262	21	61
49	P(Product = KP481 | Gender = Male, Marital Sta...	0.232558	10	43
50	P(Product = KP781 | Gender = Female, Marital S...	0.086957	4	46
51	P(Product = KP781 | Gender = Female, Marital S...	0.1	3	30
52	P(Product = KP781 | Gender = Male, Marital Sta...	0.311475	19	61
53	P(Product = KP781 | Gender = Male, Marital Sta...	0.325581	14	43

In [35]:
```
# Business Insights
# From the above Analysis, customers with the higher income, higher usage, higher fitness, higher miles and higher education
# appear to prefer purchasing KP781,
# The following data from the prob_dataframe confirm the same.
# P(Product = KP781| Usage > 4) = 80.7%
# P(Product = KP781, Fitness > 4) = 93.5%
# P(Product = KP781, Fitness >= 4) = 65.4%
# P(Product = KP781, Income> 58204) = 65.2%
# P(Product = KP781, Miles > 120) = 66.7%
# P(Product = KP781, Miles >= 120) = 68.9%
# Overall, 52.6% of Female have purchased KP281.
# P(Product = KP281 | Gender = Female, Marital Status = Partnered) = 58.7%
```

In [36]:
```
case = []
cond_prob = []
case_count = []
cond_count = []
x = ["Education", "Usage", "Fitness"]
for i in range(len(x)-1):
    for j in range(i+1, len(x)):
        for k in data[x[i]].unique():
            for l in data[x[j]].unique():
                z1 = data.loc[(data[x[i]]>=k) & (data[x[j]]>= l),::]
                for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} >= {k}, {x[j]} >= {l})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
                z1 = data.loc[(data[x[i]]<=k) & (data[x[j]]<= l),::]
                for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} <= {k}, {x[j]} <= {l})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
                z1 = data.loc[(data[x[i]]>=k) & (data[x[j]]<= l),::]
                for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} >= {k}, {x[j]} <= {l})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
                z1 = data.loc[(data[x[i]]<=k) & (data[x[j]]>= l),::]
                for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} <= {k}, {x[j]} >= {l})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')

    for j in range(min(data["Income"]), max(data["Income"]), 500):
        for k in data[x[i]].unique():
            z1 = data.loc[(data[x[i]]>=k) & (data["Income"]>= j),::]
            for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} >= {k}, Income >= {j})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
            z1 = data.loc[(data[x[i]]>=k) & (data["Income"]<= j),::]
            for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} >= {k}, Income <= {j})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
            z1 = data.loc[(data[x[i]]<=k) & (data["Income"]>= j),::]
            for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} <= {k}, Income >= {j})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
            z1 = data.loc[(data[x[i]]<=k) & (data["Income"]<= j),::]
            for m in data["Product"].unique():
                    z2 = z1.loc[z1["Product"] == m, ::]
                    case.append(f"P(Product = {m} | {x[i]} <= {k}, Income <= {j})")
                    case_count.append(len(z2))
                    cond_count.append(len(z1))
                    if len(z1) > 0:
                        cond_prob.append(len(z2)/ len(z1))
                    else:
                        cond_prob.append('-')
```

```
        for j in range(min(data["Miles"]), max(data["Miles"]), 40):
            for k in data[x[i]].unique():
                z1 = data.loc[(data[x[i]]>=k) & (data["Miles"]>= j),::]
                for m in data["Product"].unique():
                        z2 = z1.loc[z1["Product"] == m, ::]
                        case.append(f"P(Product = {m} | {x[i]} >= {k}, Miles >= {j})")
                        case_count.append(len(z2))
                        cond_count.append(len(z1))
                        if len(z1) > 0:
                            cond_prob.append(len(z2)/ len(z1))
                        else:
                            cond_prob.append('-')
                z1 = data.loc[(data[x[i]]>=k) & (data["Miles"]<= j),::]
                for m in data["Product"].unique():
                        z2 = z1.loc[z1["Product"] == m, ::]
                        case.append(f"P(Product = {m} | {x[i]} >= {k}, Miles <= {j})")
                        case_count.append(len(z2))
                        cond_count.append(len(z1))
                        if len(z1) > 0:
                            cond_prob.append(len(z2)/ len(z1))
                        else:
                            cond_prob.append('-')
                z1 = data.loc[(data[x[i]]<=k) & (data["Miles"]>= j),::]
                for m in data["Product"].unique():
                        z2 = z1.loc[z1["Product"] == m, ::]
                        case.append(f"P(Product = {m} | {x[i]} <= {k}, Miles >= {j})")
                        case_count.append(len(z2))
                        cond_count.append(len(z1))
                        if len(z1) > 0:
                            cond_prob.append(len(z2)/ len(z1))
                        else:
                            cond_prob.append('-')
                z1 = data.loc[(data[x[i]]<=k) & (data["Miles"]<= j),::]
                for m in data["Product"].unique():
                        z2 = z1.loc[z1["Product"] == m, ::]
                        case.append(f"P(Product = {m} | {x[i]} <= {k}, Miles <= {j})")
                        case_count.append(len(z2))
                        cond_count.append(len(z1))
                        if len(z1) > 0:
                            cond_prob.append(len(z2)/ len(z1))
                        else:
                            cond_prob.append('-')
```

In [37]:
```
prob_dataframe = prob_dataframe.append(pd.DataFrame([case, cond_prob, case_count, cond_count]).T.rename(columns = {0:"Case",
```

In [38]:
```
prob_dataframe.loc[prob_dataframe["Probability"] == '-',::] = 0
```

In [42]:
```
prob_dataframe[(prob_dataframe["Case count"] > 30) & (prob_dataframe["Probability"] >= 0.8)].sort_values("Probability",  asc
# From  prob_dataframe, P(Product = KP781 | Usage >= 4, Income >= 57062) = 0.885714
```

Out[42]:

	Case	Probability	Case count	Condition count
0	P(Product = KP781 | Usage >= 4, Income >= 57062)	0.885714	31	35
1	P(Product = KP781 | Usage >= 4, Income >= 55062)	0.861111	31	36
2	P(Product = KP781 | Usage >= 4, Income >= 55562)	0.861111	31	36
3	P(Product = KP781 | Usage >= 4, Income >= 56062)	0.861111	31	36
4	P(Product = KP781 | Usage >= 4, Income >= 56562)	0.861111	31	36

In [43]:
```
prob_dataframe[(prob_dataframe["Case count"] > 30) & (prob_dataframe["Probability"] >= 0.7) & (prob_dataframe["Probability"]
# From  prob_dataframe, P(Product = KP781 | Usage >= 4, Income >= 57062) = 0.885714
# P(Product = KP781 | Usage >= 4, Fitness >= 4)  = 0.7954
# P(Product = KP781 | Education >= 15, Fitness >= 4) = 0.778
```

Out[43]:

	Case	Probability	Case count	Condition count
0	P(Product = KP781 | Education >= 16, Fitness >...	0.795455	35	44
1	P(Product = KP781 | Usage >= 4, Fitness >= 4)	0.795455	35	44
2	P(Product = KP781 | Usage >= 4, Income >= 53562)	0.794872	31	39
3	P(Product = KP781 | Usage >= 4, Income >= 54062)	0.794872	31	39
4	P(Product = KP781 | Usage >= 4, Income >= 54562)	0.794872	31	39
5	P(Product = KP781 | Education >= 15, Fitness >...	0.777778	35	45
6	P(Product = KP781 | Usage >= 4, Income >= 52562)	0.761905	32	42
7	P(Product = KP781 | Usage >= 4, Income >= 53062)	0.761905	32	42
8	P(Product = KP781 | Usage >= 4, Income >= 51562)	0.73913	34	46
9	P(Product = KP781 | Usage >= 4, Income >= 52062)	0.73913	34	46
10	P(Product = KP781 | Usage >= 4, Income >= 50062)	0.708333	34	48
11	P(Product = KP781 | Usage >= 4, Income >= 50562)	0.708333	34	48
12	P(Product = KP781 | Usage >= 4, Income >= 51062)	0.708333	34	48

In [45]:
```
prob_dataframe[(prob_dataframe["Case count"] > 30) & (prob_dataframe["Probability"] >= 0.65) & (prob_dataframe["Probability"]
# From  prob_dataframe, P(Product = KP281 | Usage <= 4, Income <= 45062) = 0.695
# P(Product = KP281 | Usage <= 5, Income <= 45062) = 0.6938
# P(Product = KP281 | Usage >= 2, Income <= 44562) = 0.6938
```

Out[45]:

	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Usage <= 4, Income <= 45062)	0.695652	32	46
1	P(Product = KP281 | Education >= 13, Income <=...	0.695652	32	46
2	P(Product = KP281 | Usage <= 4, Income <= 44562)	0.695652	32	46
3	P(Product = KP281 | Education >= 13, Income <=...	0.695652	32	46
4	P(Product = KP281 | Education <= 18, Income <=...	0.693878	34	49
5	P(Product = KP281 | Usage <= 7, Income <= 45062)	0.693878	34	49
6	P(Product = KP281 | Usage <= 5, Income <= 45062)	0.693878	34	49
7	P(Product = KP281 | Usage >= 2, Income <= 45062)	0.693878	34	49
8	P(Product = KP281 | Usage <= 7, Income <= 44562)	0.693878	34	49
9	P(Product = KP281 | Usage <= 6, Income <= 44562)	0.693878	34	49
10	P(Product = KP281 | Usage <= 5, Income <= 44562)	0.693878	34	49
11	P(Product = KP281 | Usage >= 2, Income <= 44562)	0.693878	34	49
12	P(Product = KP281 | Education <= 21, Income <=...	0.693878	34	49
13	P(Product = KP281 | Education <= 20, Income <=...	0.693878	34	49
14	P(Product = KP281 | Usage <= 6, Income <= 45062)	0.693878	34	49
15	P(Product = KP281 | Education <= 16, Income <=...	0.693878	34	49
16	P(Product = KP281 | Education >= 12, Income <=...	0.693878	34	49
17	P(Product = KP281 | Education <= 21, Income <=...	0.693878	34	49
18	P(Product = KP281 | Education <= 20, Income <=...	0.693878	34	49
19	P(Product = KP281 | Education <= 18, Income <=...	0.693878	34	49
20	P(Product = KP281 | Education <= 16, Income <=...	0.693878	34	49
21	P(Product = KP281 | Education >= 12, Income <=...	0.693878	34	49
22	P(Product = KP781 | Usage >= 4, Income >= 49062)	0.692308	36	52
23	P(Product = KP781 | Usage >= 4, Income >= 49562)	0.692308	36	52
24	P(Product = KP781 | Miles >= 120.0)	0.688889	31	45
25	P(Product = KP781 | Usage >= 4, Income >= 48062)	0.684211	39	57
26	P(Product = KP781 | Usage >= 4, Income >= 48562)	0.672727	37	55
27	P(Product = KP781 | Usage >= 4, Income >= 47562)	0.672414	39	58
28	P(Product = KP781 | Usage >= 4, Income >= 47062)	0.672414	39	58
29	P(Product = KP781 | Education >= 14, Fitness >...	0.666667	36	54
30	P(Product = KP781 | Fitness >= 4.0)	0.654545	36	55
31	P(Product = KP781 | Usage <= 7, Fitness >= 4)	0.654545	36	55
32	P(Product = KP781 | Usage >= 2, Fitness >= 4)	0.654545	36	55
33	P(Product = KP781 | Education <= 21, Fitness >...	0.654545	36	55
34	P(Product = KP781 | Education >= 13, Fitness >...	0.654545	36	55
35	P(Product = KP781 | Education >= 12, Fitness >...	0.654545	36	55
36	P(Product = KP781 | Usage >= 3, Fitness >= 4)	0.654545	36	55

In [46]:
```
prob_dataframe
```

Out[46]:

	Case	Probability	Case count	Condition count
0	P(Product = KP281 | Education > 16.0)	0.074074	2	27
1	P(Product = KP281 | Education >= 16.0)	0.366071	41	112
2	P(Product = KP481 | Education > 16.0)	0.074074	2	27
3	P(Product = KP481 | Education >= 16.0)	0.294643	33	112
4	P(Product = KP781 | Education > 16.0)	0.851852	23	27
...	...	...	...	...
255145	P(Product = KP481 | Usage <= 7, Miles >= 341)	0.0	0	1
255146	P(Product = KP781 | Usage <= 7, Miles >= 341)	1.0	1	1
255147	P(Product = KP281 | Usage <= 7, Miles <= 341)	0.446927	80	179
255148	P(Product = KP481 | Usage <= 7, Miles <= 341)	0.335196	60	179
255149	P(Product = KP781 | Usage <= 7, Miles <= 341)	0.217877	39	179

255150 rows × 4 columns