ecommerce-eda-1

October 24, 2024

1 Shopping EDA

1.0.1 Importing Libraries & Dataset

[55]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from scipy.stats import f_oneway,kruskal
```

[2]:
```
shopping_df = pd.read_csv('shopping.csv')
shopping_df.head(5)
```

[2]:

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
0	0	0.0	0	0.0	1	0.000000	0.20	0.20	0.0	0.0	Feb	1	1	1	1	Returning_Visitor	False	False
1	0	0.0	0	0.0	2	64.000000	0.00	0.10	0.0	0.0	Feb	2						
2	0	0.0	0	0.0	1	0.000000	0.20	0.20	0.0	0.0	Feb	4						
3	0	0.0	0	0.0	2	2.666667	0.05	0.14	0.0	0.0	Feb	3						
4	0	0.0	0	0.0	10	627.500000	0.02	0.05	0.0	0.0	Feb	3						

```
1        2        1            2  Returning_Visitor    False    False
2        1        9            3  Returning_Visitor    False    False
3        2        2            4  Returning_Visitor    False    False
4        3        1            4  Returning_Visitor     True    False
```

1.0.2 Basic Metrics

[3]:
```
shopping_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Administrative           12330 non-null  int64
 1   Administrative_Duration  12330 non-null  float64
 2   Informational            12330 non-null  int64
 3   Informational_Duration   12330 non-null  float64
 4   ProductRelated           12330 non-null  int64
 5   ProductRelated_Duration  12330 non-null  float64
 6   BounceRates              12330 non-null  float64
 7   ExitRates                12330 non-null  float64
 8   PageValues               12330 non-null  float64
 9   SpecialDay               12330 non-null  float64
 10  Month                    12330 non-null  object
 11  OperatingSystems         12330 non-null  int64
 12  Browser                  12330 non-null  int64
 13  Region                   12330 non-null  int64
 14  TrafficType              12330 non-null  int64
 15  VisitorType              12330 non-null  object
 16  Weekend                  12330 non-null  bool
 17  Revenue                  12330 non-null  bool
dtypes: bool(2), float64(7), int64(7), object(2)
memory usage: 1.5+ MB
```

[4]:
```
shopping_df.isna().sum()
```

[4]:
```
Administrative             0
Administrative_Duration    0
Informational              0
Informational_Duration     0
ProductRelated             0
ProductRelated_Duration    0
BounceRates                0
ExitRates                  0
PageValues                 0
SpecialDay                 0
```

```
Month                    0
OperatingSystems         0
Browser                  0
Region                   0
TrafficType              0
VisitorType              0
Weekend                  0
Revenue                  0
dtype: int64
```

[5]:
```
shopping_df.duplicated().value_counts()
```

[5]:
```
False    12205
True       125
Name: count, dtype: int64
```

[6]:
```
shopping_df.describe()
```

[6]:

	Administrative	Administrative_Duration	Informational	\
count	12330.000000	12330.000000	12330.000000	
mean	2.315166	80.818611	0.503569	
std	3.321784	176.779107	1.270156	
min	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	
50%	1.000000	7.500000	0.000000	
75%	4.000000	93.256250	0.000000	
max	27.000000	3398.750000	24.000000	

	Informational_Duration	ProductRelated	ProductRelated_Duration	\
count	12330.000000	12330.000000	12330.000000	
mean	34.472398	31.731468	1194.746220	
std	140.749294	44.475503	1913.669288	
min	0.000000	0.000000	0.000000	
25%	0.000000	7.000000	184.137500	
50%	0.000000	18.000000	598.936905	
75%	0.000000	38.000000	1464.157214	
max	2549.375000	705.000000	63973.522230	

	BounceRates	ExitRates	PageValues	SpecialDay	\
count	12330.000000	12330.000000	12330.000000	12330.000000	
mean	0.022191	0.043073	5.889258	0.061427	
std	0.048488	0.048597	18.568437	0.198917	
min	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.014286	0.000000	0.000000	
50%	0.003112	0.025156	0.000000	0.000000	
75%	0.016813	0.050000	0.000000	0.000000	
max	0.200000	0.200000	361.763742	1.000000	

	OperatingSystems	Browser	Region	TrafficType
count	12330.000000	12330.000000	12330.000000	12330.000000
mean	2.124006	2.357097	3.147364	4.069586
std	0.911325	1.717277	2.401591	4.025169
min	1.000000	1.000000	1.000000	1.000000
25%	2.000000	2.000000	1.000000	2.000000
50%	2.000000	2.000000	3.000000	2.000000
75%	3.000000	2.000000	4.000000	4.000000
max	8.000000	13.000000	9.000000	20.000000

1.0.3 Non Graphical Analysis

[7]:
```
column =␣
  ↪['Administrative','Informational','ProductRelated','Month','OperatingSystems','Browser','Re
for i in column:
  print(shopping_df.value_counts(i))
  print('\n')
```

```
Administrative
0     5768
1     1354
2     1114
3      915
4      765
5      575
6      432
7      338
8      287
9      225
10     153
11     105
12      86
13      56
14      44
15      38
16      24
17      16
18      12
19       6
22       4
24       4
23       3
20       2
21       2
26       1
27       1
```

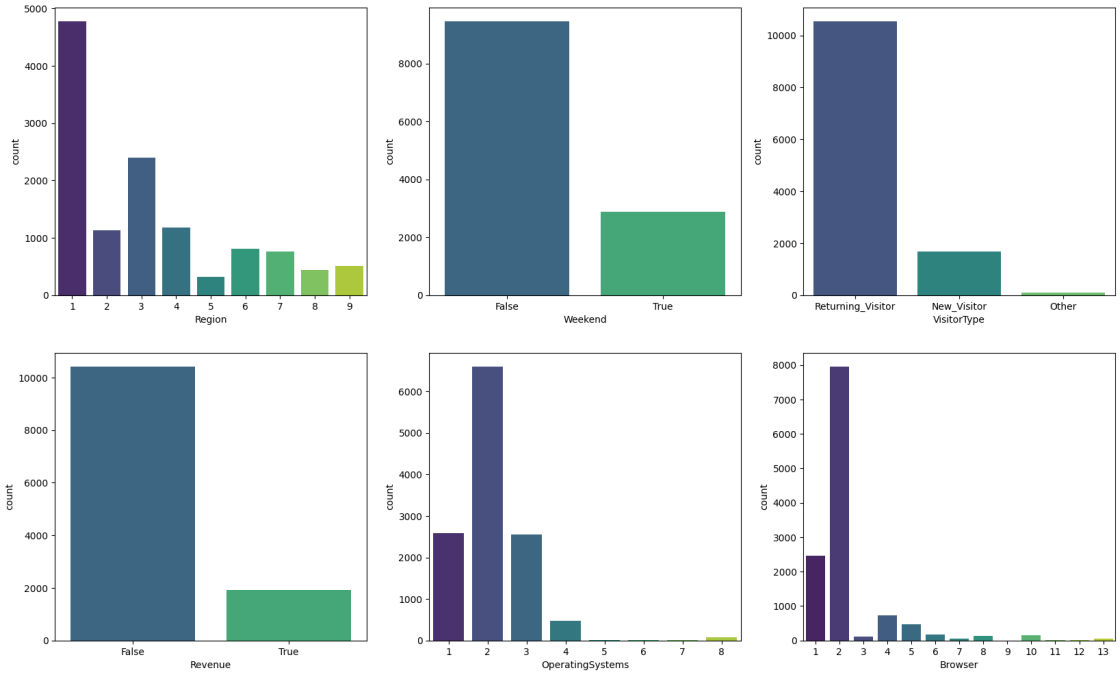
```
Name: count, dtype: int64


Informational
0     9699
1     1041
2      728
3      380
4      222
5       99
6       78
7       36
9       15
8       14
10       7
12       5
14       2
11       1
13       1
16       1
24       1
Name: count, dtype: int64


ProductRelated
1      622
2      465
3      458
4      404
6      396
      …
255      1
254      1
251      1
250      1
705      1
Name: count, Length: 311, dtype: int64


Month
May     3364
Nov     2998
Mar     1907
Dec     1727
Oct      549
Sep      448
Aug      433
Jul      432
```

```
June     288
Feb      184
Name: count, dtype: int64


OperatingSystems
2    6601
1    2585
3    2555
4     478
8      79
6      19
7       7
5       6
Name: count, dtype: int64


Browser
2     7961
1     2462
4      736
5      467
6      174
10     163
8      135
3      105
13      61
7       49
12      10
11       6
9        1
Name: count, dtype: int64


Region
1    4780
3    2403
4    1182
2    1136
6     805
7     761
9     511
8     434
5     318
Name: count, dtype: int64


TrafficType
```

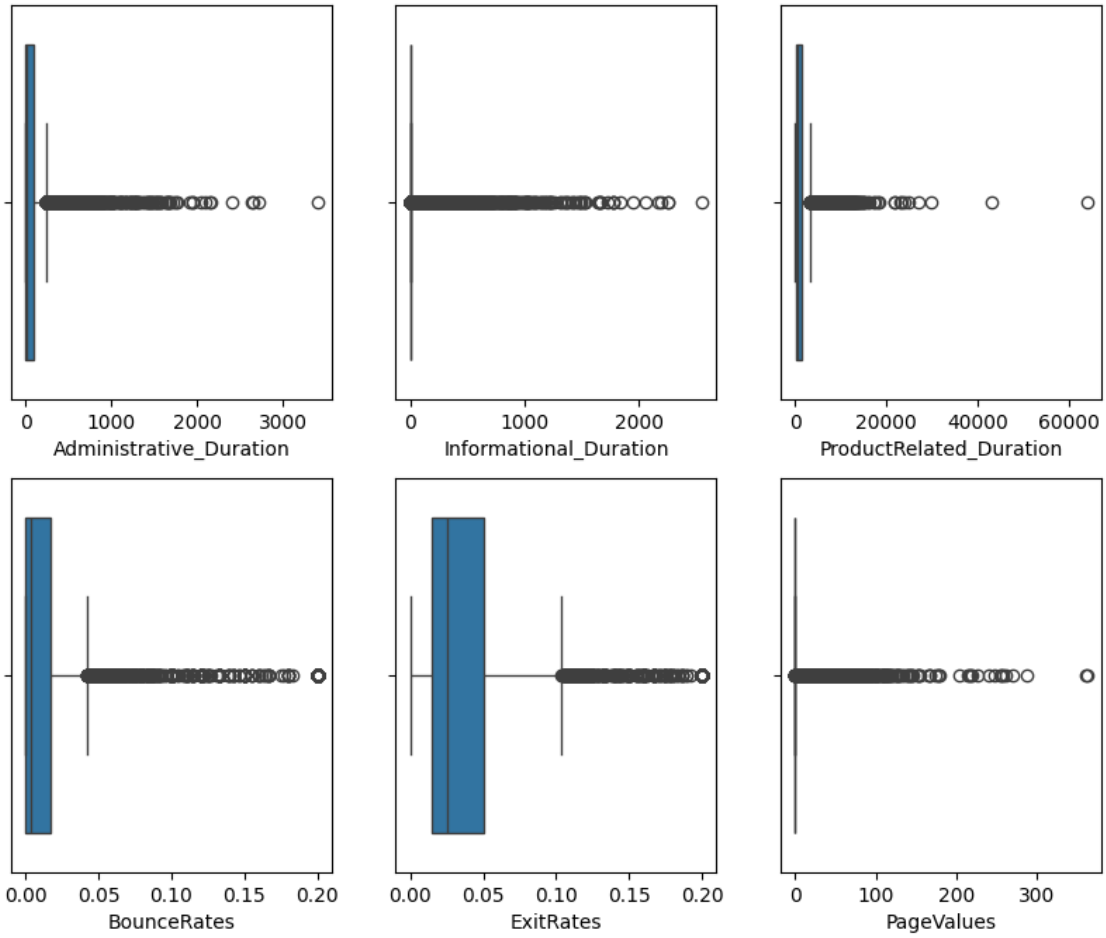
```
2     3913
1     2451
3     2052
4     1069
13     738
10     450
6      444
8      343
5      260
11     247
20     198
9       42
7       40
15      38
19      17
14      13
18      10
16       3
12       1
17       1
Name: count, dtype: int64


Weekend
False    9462
True     2868
Name: count, dtype: int64


VisitorType
Returning_Visitor    10551
New_Visitor           1694
Other                   85
Name: count, dtype: int64


Revenue
False    10422
True      1908
Name: count, dtype: int64
```

1.0.4 Graphical Analysis

[8]:
```
fig, axs = plt.subplots(nrows = 2, ncols = 3, figsize = (20,12))
sns.countplot(data = shopping_df, x = 'Region', ax = axs[0,0], palette =␣
 ↪'viridis')
sns.countplot(data = shopping_df, x = 'Weekend', ax = axs[0,1], palette =␣
 ↪'viridis')
sns.countplot(data = shopping_df, x = 'VisitorType', ax = axs[0,2], palette =␣
 ↪'viridis')
sns.countplot(data = shopping_df, x = 'Revenue', ax = axs[1,0], palette =␣
 ↪'viridis')
sns.countplot(data = shopping_df, x = 'OperatingSystems', ax = axs[1,1],␣
 ↪palette = 'viridis')
sns.countplot(data = shopping_df, x = 'Browser', ax = axs[1,2], palette =␣
 ↪'viridis')
plt.show()
```

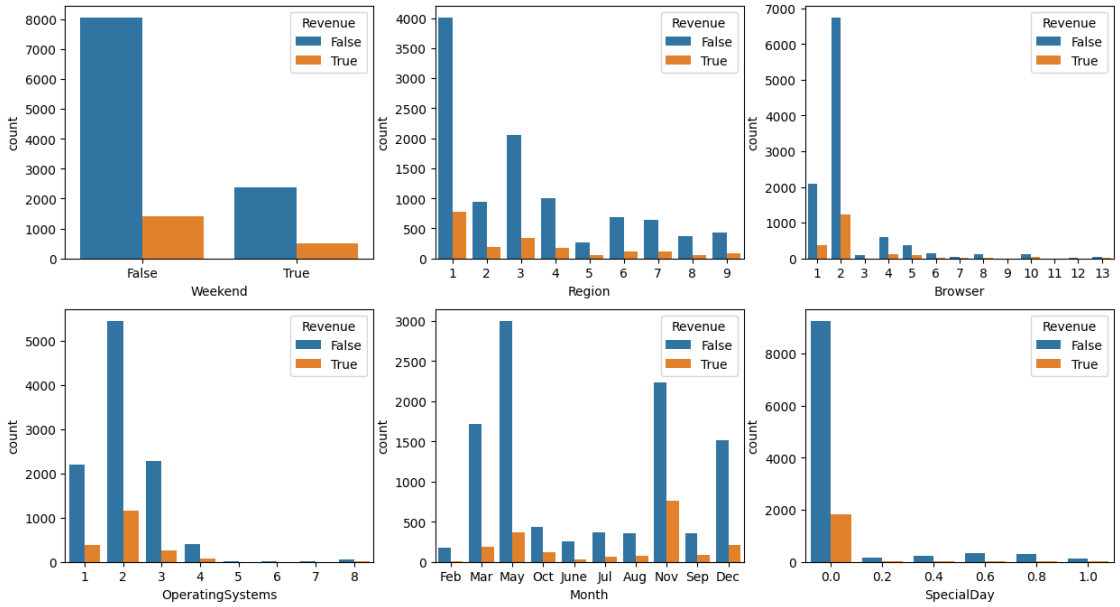


[9]:
```
fig, axs = plt.subplots(nrows = 2, ncols = 3, figsize = (10,8))
sns.boxplot(data = shopping_df, x = 'Administrative_Duration', ax = axs[0,0])
sns.boxplot(data = shopping_df, x = 'Informational_Duration', ax = axs[0,1])
sns.boxplot(data = shopping_df, x = 'ProductRelated_Duration', ax = axs[0,2])
sns.boxplot(data = shopping_df, x = 'BounceRates', ax = axs[1,0])
sns.boxplot(data = shopping_df, x = 'ExitRates', ax = axs[1,1])
sns.boxplot(data = shopping_df, x = 'PageValues', ax = axs[1,2])
plt.show()
```

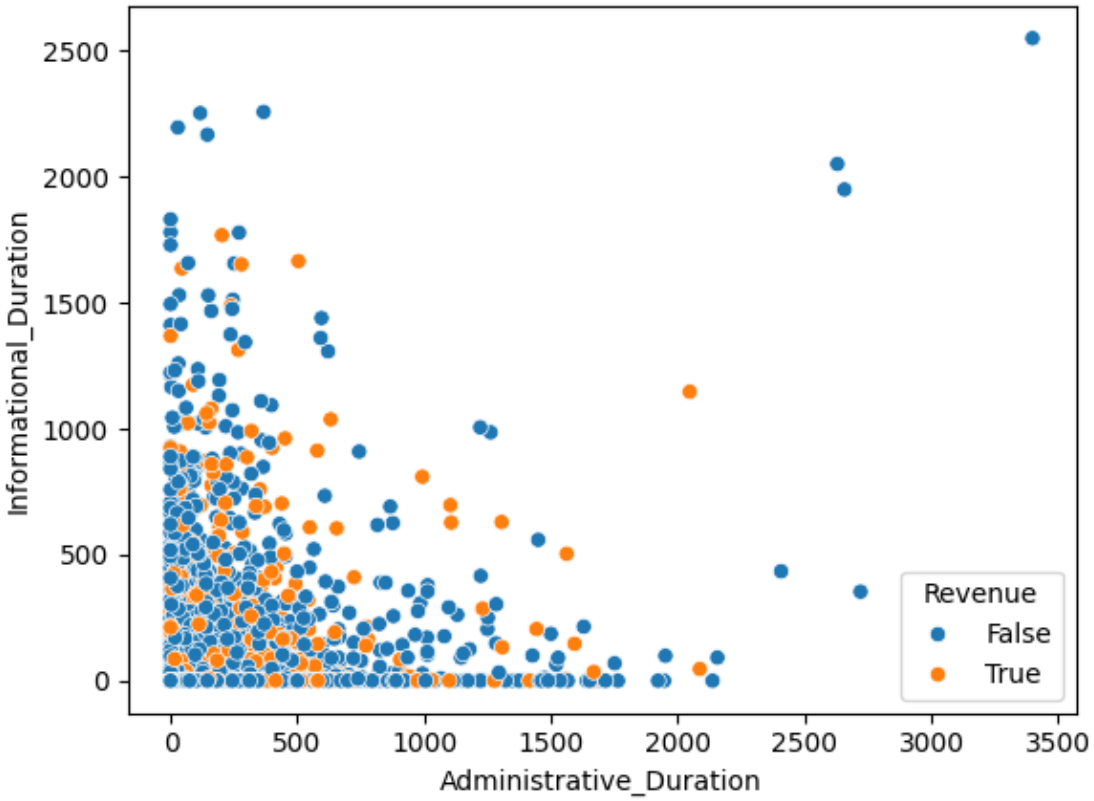


1.0.5 Bivariate Analysis

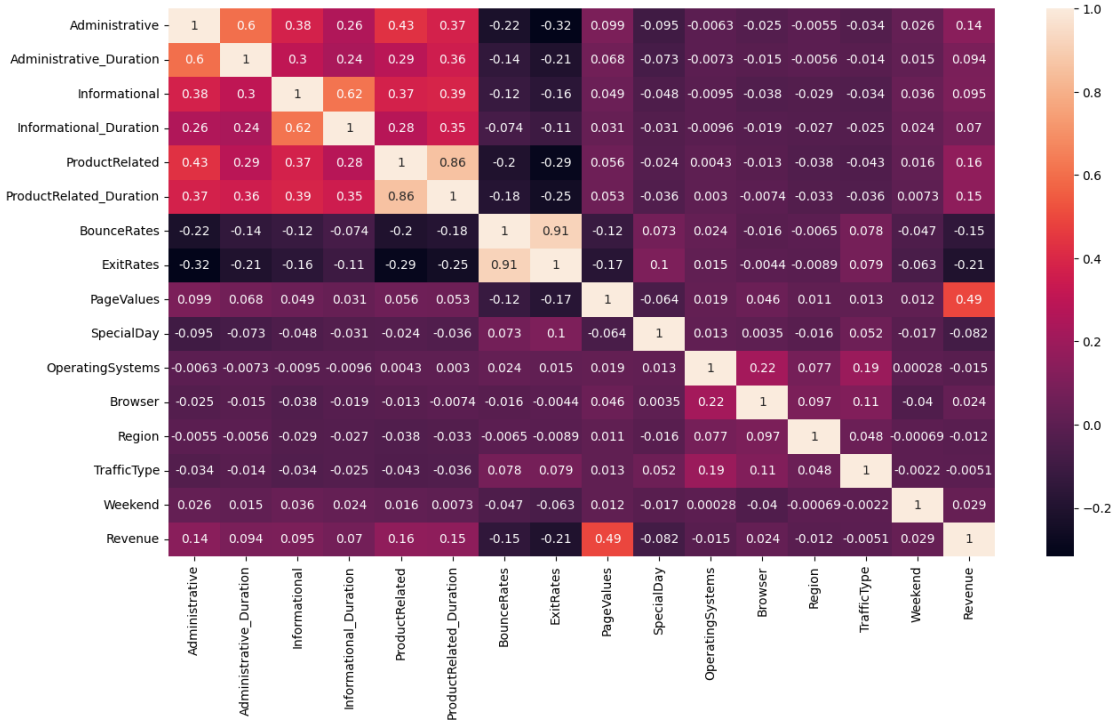
[10]:
```
fig, axs = plt.subplots(nrows = 2, ncols = 3, figsize = (15,8))
sns.countplot(data = shopping_df, hue = 'Revenue', x = 'Weekend',ax = axs[0,0])
sns.countplot(data = shopping_df, hue = 'Revenue', x = 'Region',ax = axs[0,1])
sns.countplot(data = shopping_df, hue = 'Revenue', x = 'Browser',ax = axs[0,2])
sns.countplot(data = shopping_df, hue = 'Revenue', x = 'OperatingSystems',ax =␣
 ↪axs[1,0])
sns.countplot(data = shopping_df, x = 'Month', hue = 'Revenue',ax = axs[1,1])
sns.countplot(data = shopping_df, x = 'SpecialDay', hue = 'Revenue',ax =␣
 ↪axs[1,2])
plt.show()
```



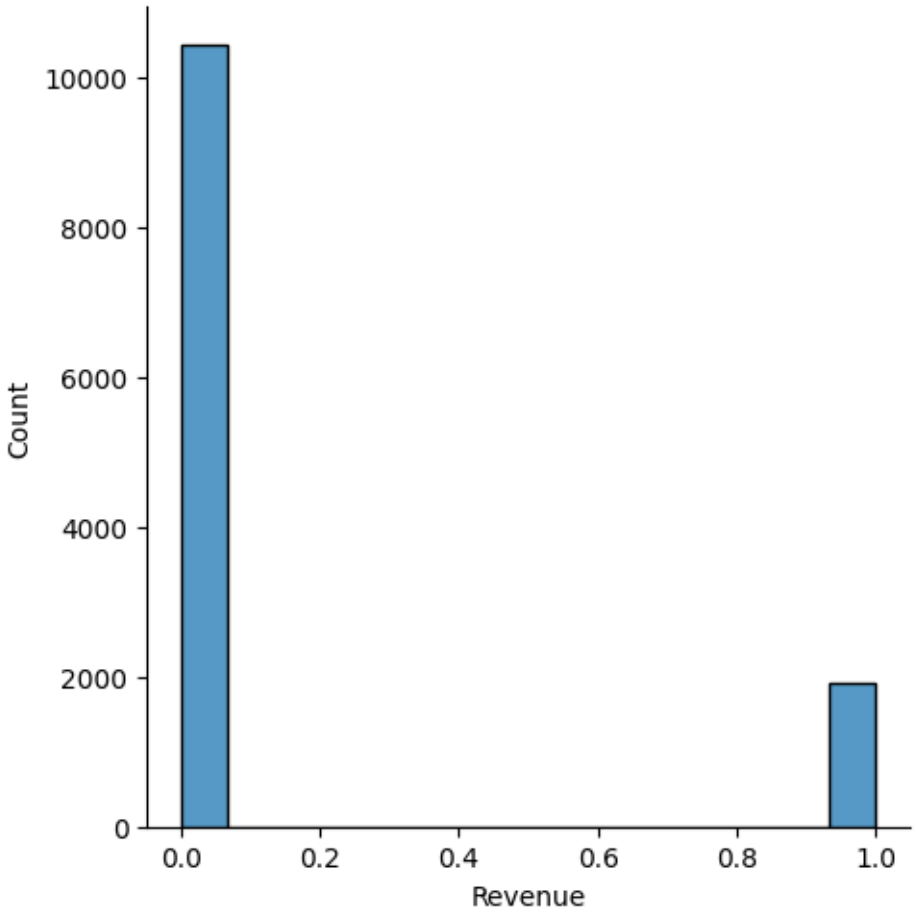
[11]:
```
sns.scatterplot(data = shopping_df, x = 'Administrative_Duration', y =␣
↪'Informational_Duration', hue = 'Revenue')
plt.show()
```



[12]:
```
plt.figure(figsize = (15,8))
cor = shopping_df.corr(numeric_only = True)
sns.heatmap(cor, annot = True)
plt.show()
```

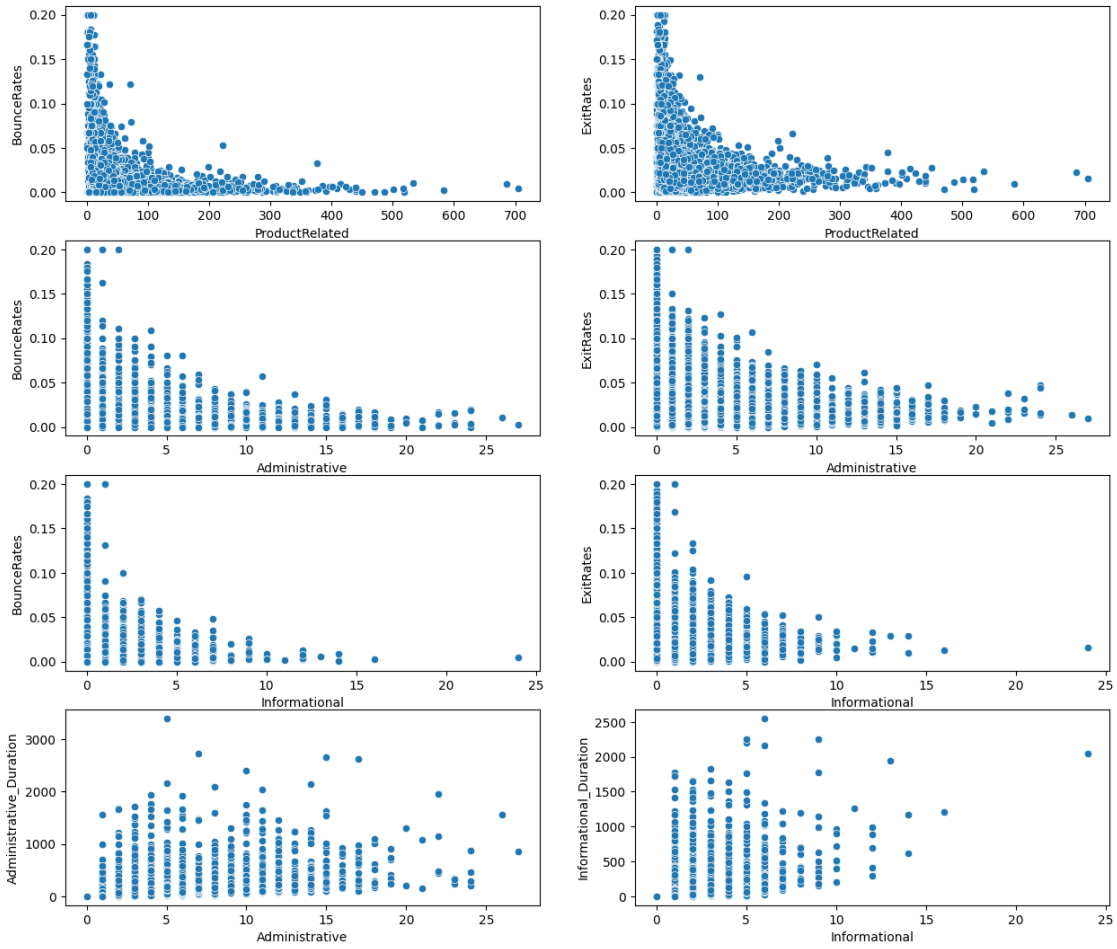


[13]:
```
sns.displot(data = shopping_df, x = 'Revenue')
plt.show()
```



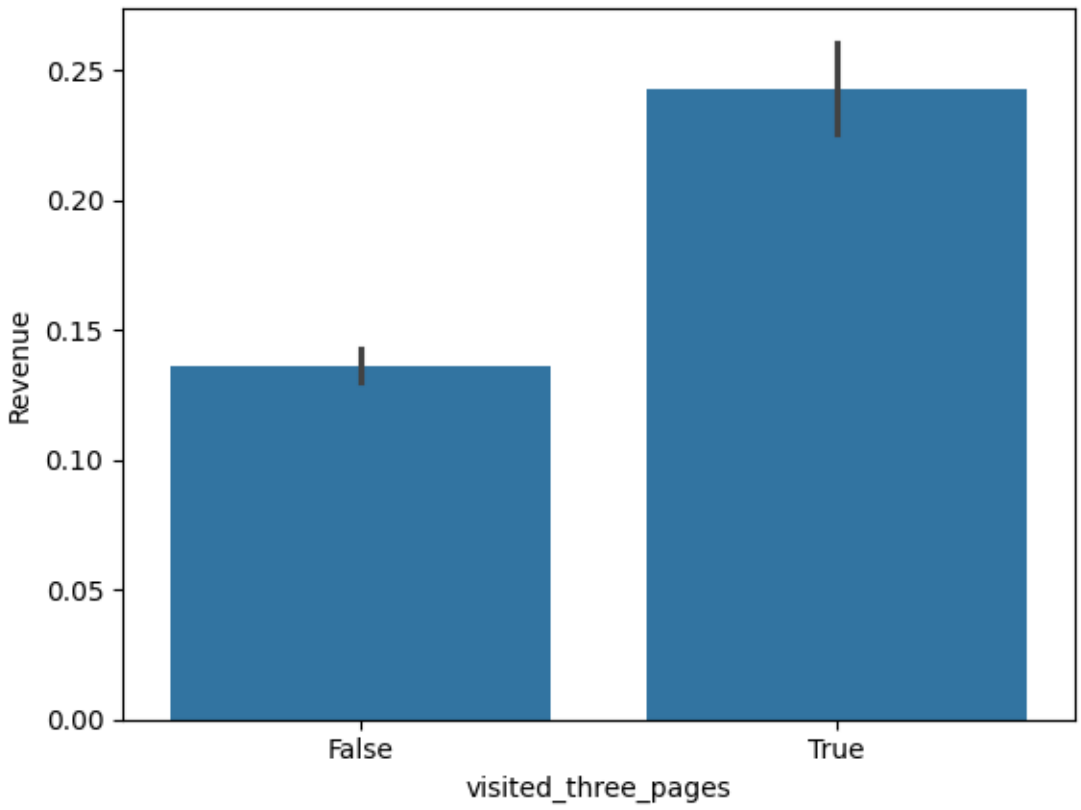
[14]:
```
fig, axs = plt.subplots(nrows = 4, ncols = 2, figsize = (15,13))
sns.scatterplot(data = shopping_df, x = 'ProductRelated', y = 'BounceRates',ax␣
 ↪= axs[0,0])
sns.scatterplot(data = shopping_df, x = 'ProductRelated', y = 'ExitRates',ax =␣
 ↪axs[0,1])
sns.scatterplot(data = shopping_df, x = 'Administrative', y = 'BounceRates',ax␣
 ↪= axs[1,0])
sns.scatterplot(data = shopping_df, x = 'Administrative', y = 'ExitRates',ax =␣
 ↪axs[1,1])
sns.scatterplot(data = shopping_df, x = 'Informational', y = 'BounceRates',ax =␣
 ↪axs[2,0])
sns.scatterplot(data = shopping_df, x = 'Informational', y = 'ExitRates',ax =␣
 ↪axs[2,1])
sns.scatterplot(data = shopping_df, x = 'Administrative', y =␣
 ↪'Administrative_Duration',ax = axs[3,0])
sns.scatterplot(data = shopping_df, x = 'Informational', y =␣
 ↪'Informational_Duration',ax = axs[3,1])
```

```
plt.show()
```

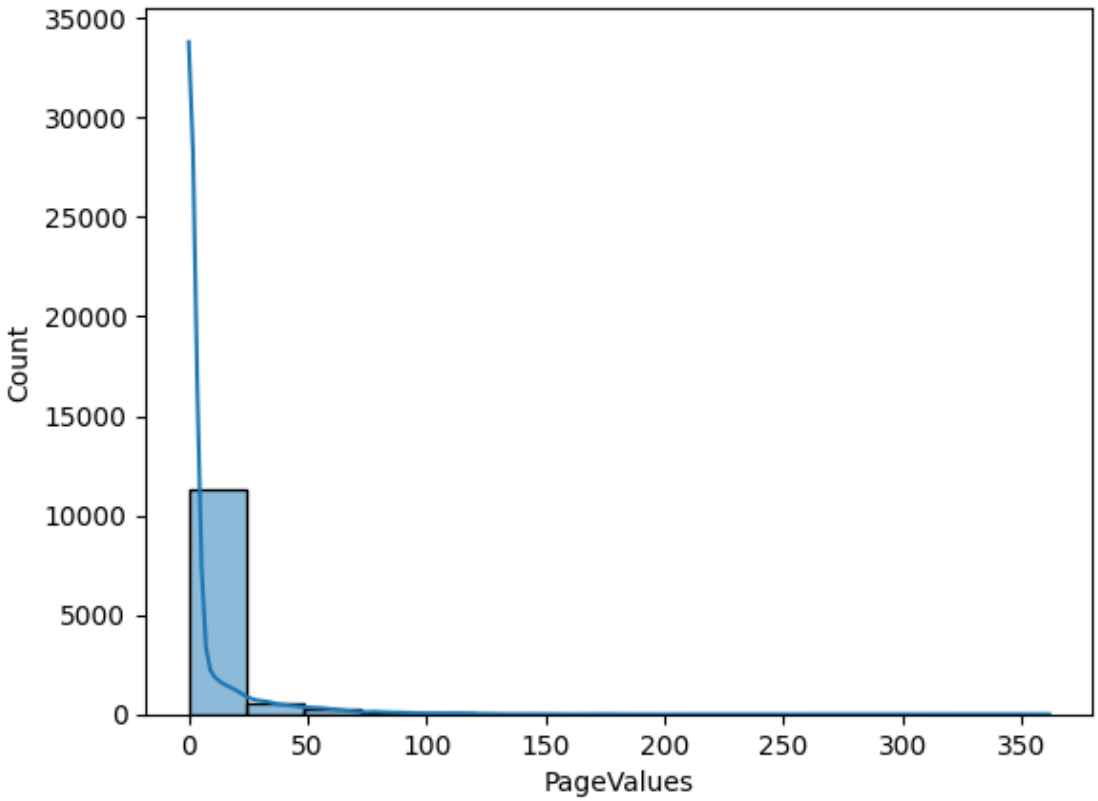


[15]:
```
shopping_df['visited_three_pages'] = (shopping_df['Administrative'] > 0) &␣
↪(shopping_df['Informational'] > 0) & (shopping_df['ProductRelated'] > 0)
```

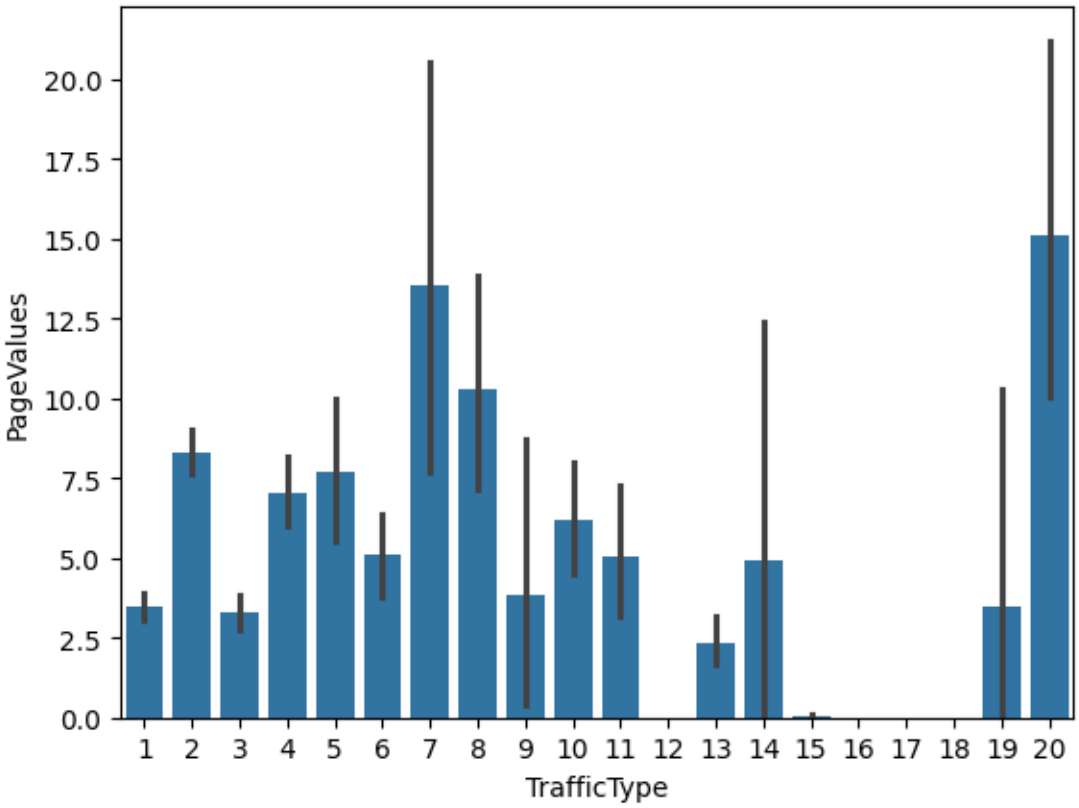
[16]:
```
sns.barplot(data = shopping_df, x = 'visited_three_pages', y = 'Revenue')
plt.show()
```



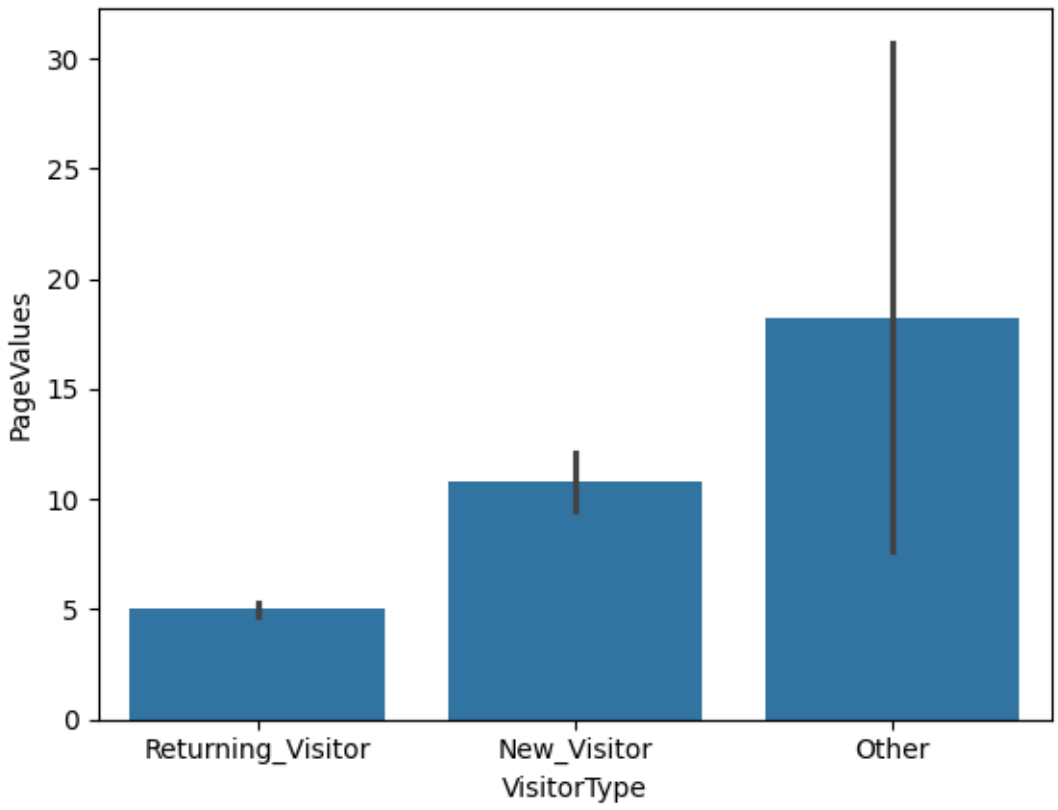
[17]:
```
sns.histplot(data = shopping_df, x = 'PageValues', kde = True)
plt.show()
```



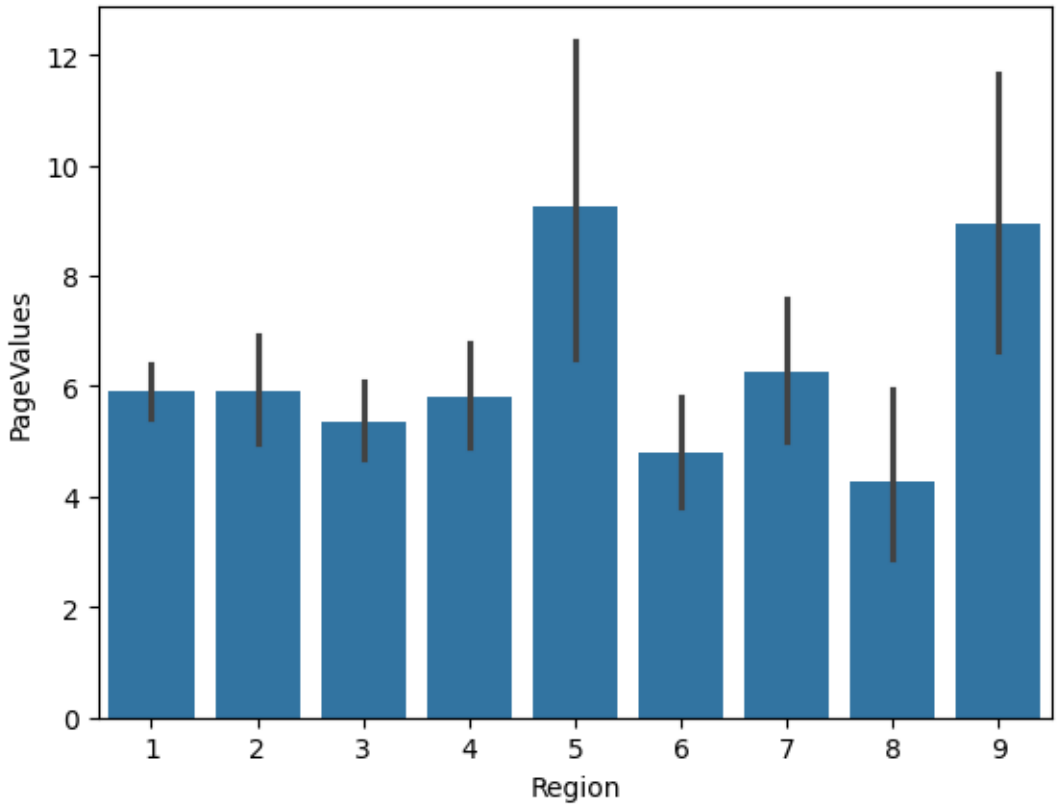
[18]:
```
sns.barplot(data=shopping_df, y='PageValues', x='TrafficType')
plt.show()
```



[19]:
```
sns.barplot(data=shopping_df, y='PageValues', x='VisitorType')
plt.show()
```



[20]:
```
sns.barplot(data=shopping_df, y='PageValues', x='Region')
plt.show()
```



[21]:
```
df_new = shopping_df[shopping_df['visited_three_pages'] == True]
df_new
```

[21]:

	Administrative	Administrative_Duration	Informational	\
29	1	6.000000	1	
57	4	56.000000	2	
103	2	31.000000	1	
109	6	326.250000	4	
161	2	58.000000	2	
…	…	…	…	
12287	8	167.910714	6	
12307	2	305.125000	3	
12311	1	0.000000	2	
12312	7	150.357143	1	
12313	3	16.000000	3	

	Informational_Duration	ProductRelated	ProductRelated_Duration	\
29	0.00	45	1582.750000	
57	120.00	36	998.741667	
103	16.00	36	2083.530952	
109	94.00	128	5062.213753	

```
161                    22.00             31               829.166667
…                      …            …                      …
12287                 547.75            111              6340.152381
12307                 368.25             27              1121.250000
12311                 211.25            144              4627.489571
12312                   9.00            221             11431.001240
12313                  86.00             15              2773.500000

       BounceRates  ExitRates  PageValues  SpecialDay Month  OperatingSystems  \
29        0.043478   0.050821   54.179764         0.4   Feb                 3
57        0.000000   0.014736   19.447079         0.2   Feb                 2
103       0.000000   0.013510    0.000000         0.8   Feb                 2
109       0.000855   0.017918    0.000000         0.0   Feb                 2
161       0.030303   0.040606    0.000000         0.0   Feb                 1
…              …          …           …          …    …                  …
12287     0.003361   0.009432   44.219794         0.0   Dec                 3
12307     0.020000   0.042857   39.519807         0.0   Dec                 3
12311     0.001361   0.020664    0.000000         0.0   Nov                 2
12312     0.011149   0.021904    1.582473         0.0   Nov                 2
12313     0.000000   0.030000   78.811725         0.0   Dec                 2

       Browser  Region  TrafficType        VisitorType  Weekend  Revenue  \
29           2       1            1  Returning_Visitor    False    False
57           2       4            1  Returning_Visitor    False    False
103          2       4            3  Returning_Visitor    False    False
109          5       1            3  Returning_Visitor    False    False
161          1       1            1  Returning_Visitor     True    False
…          …       …            …                  …       …       …
12287        2       6            2  Returning_Visitor    False    False
12307        2       1            2  Returning_Visitor    False    False
12311        2       1            2  Returning_Visitor    False     True
12312        5       1            2  Returning_Visitor     True     True
12313        2       1            2  Returning_Visitor    False     True

       visited_three_pages
29                    True
57                    True
103                   True
109                   True
161                   True
…                     …
12287                 True
12307                 True
12311                 True
12312                 True
12313                 True
```

```
[2167 rows x 19 columns]
```

[22]:
```
q1 = df_new['Administrative_Duration'].quantile(0.25)
q3 = df_new['Administrative_Duration'].quantile(0.75)
IQR = q3 - q1
df_new = df_new[~((df_new['Administrative_Duration'] < (q1 - 1.5*IQR)) |␣
 ↪(df_new['Administrative_Duration'] > (q3 + 1.5*IQR)))]
```

[23]:
```
q1 = df_new['Informational_Duration'].quantile(0.25)
q3 = df_new['Informational_Duration'].quantile(0.75)
IQR = q3 - q1
df_new = df_new[~((df_new['Informational_Duration'] < (q1 - 1.5*IQR)) |␣
 ↪(df_new['Informational_Duration'] > (q3 + 1.5*IQR)))]
```

[24]:
```
q1 = df_new['ProductRelated_Duration'].quantile(0.25)
q3 = df_new['ProductRelated_Duration'].quantile(0.75)
IQR = q3 - q1
df_new = df_new[~((df_new['ProductRelated_Duration'] < (q1 - 1.5*IQR)) |␣
 ↪(df_new['ProductRelated_Duration'] > (q3 + 1.5*IQR)))]
```

[25]:
```
df_new.shape
```

[25]:
```
(1707, 19)
```

[26]:
```
df_new['Administrative_Duration'].mean()
```

[26]:
```
141.68619144000408
```

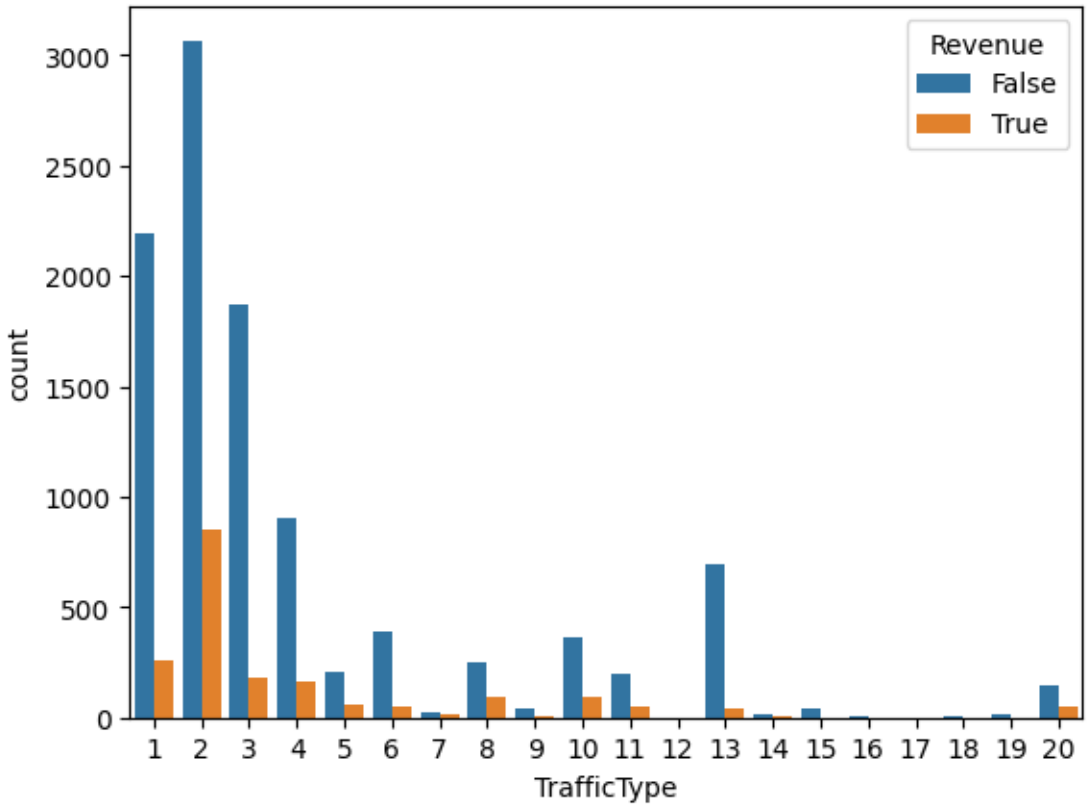
[27]:
```
df_new['Informational_Duration'].mean()
```

[27]:
```
94.71467943631633
```

[28]:
```
df_new['ProductRelated_Duration'].mean()
```

[28]:
```
1768.9865348270553
```

[29]:
```
sns.countplot(data = shopping_df, x = 'TrafficType', hue = 'Revenue')
plt.show()
```



1.1 Analysis

- Majority of Customers:
- Most customers come from Region 1. -The majority of customers are returning customers, as compared to new customers.
- Traffic and Conversion:
- 20% of users make transactions, while 80% just visit the website.
- The majority of customers use Operating System 2 and Browser 2.
- Traffic Type 2 contributes to the majority of the revenue.
- Transaction and Sales Patterns:
- The majority of transactions happen on weekdays.
- The majority of sales occur on weekdays.
- November is the month where the majority of revenue is generated.
- Most of the revenue occurs on Special Day 0 (this might need clarification, possibly an event or promotion).
- Revenue Insights:

- The majority of the revenue comes from Region 1.
- Browser 2 and Operating System 2 are responsible for the highest revenue.
- The majority of conversion happens when customers visit all three pages on the website.

1.2 Recommendations

- Targeted Marketing in Region 1:
- Increase marketing efforts specifically in Region 1.
- Tailor promotional campaigns to the demographics and preferences of this region.
- Enhance Customer Retention Strategies:
- Implement loyalty programs and personalized offers for returning customers.
- Introduce referral incentives to encourage advocacy among satisfied customers.
- Optimize Weekday Sales:
- Launch weekday-specific promotions or discounts to boost sales during peak days.
- Implement targeted email campaigns to remind customers of weekday offers.
- Increase Engagement for Website Visitors:
- Invest in retargeting ads to convert website visitors into customers. Launch email marketing campaigns targeting users who visit without making purchases.
- Enhance User Experience on Key Platforms:
- Optimize the website for Operating System 2 and Browser 2.
- Improve website load times, responsiveness, and usability on these platforms.
- Capitalize on Special Days and Events:
- Analyze the specifics of Special Day 0 for insights into successful promotions.
- Create events or promotions around key holidays in November to maximize revenue.

2 Champaign EDA

[33]:
```
camp=pd.read_csv("campaign.csv")
camp.head()
```

[33]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	\
0	1826	1970	Graduation	Divorced	$84,835.00	0	
1	1	1961	Graduation	Single	$57,091.00	0	
2	10476	1958	Graduation	Married	$67,267.00	0	
3	1386	1967	Graduation	Together	$32,474.00	1	
4	5371	1989	Graduation	Single	$21,474.00	1	

	Teenhome	Dt_Customer	Recency	MntWines	…	NumCatalogPurchases	\

```
0         0     6/16/14        0       189  …                   4
1         0     6/15/14        0       464  …                   3
2         1     5/13/14        0       134  …                   2
3         1     5/11/14        0        10  …                   0
4         0      4/8/14        0         6  …                   1

   NumStorePurchases  NumWebVisitsMonth  AcceptedCmp3  AcceptedCmp4  \
0                  6                  1             0             0
1                  7                  5             0             0
2                  5                  2             0             0
3                  2                  7             0             0
4                  2                  7             1             0

   AcceptedCmp5  AcceptedCmp1  AcceptedCmp2  Complain  Country
0             0             0             0         0       SP
1             0             0             1         0       CA
2             0             0             0         0       US
3             0             0             0         0      AUS
4             0             0             0         0       SP

[5 rows x 27 columns]
```

3 Basic Metrics

[34]:
```
camp.shape
```

[34]:
```
(2239, 27)
```

[35]:
```
camp.size
```

[35]:
```
60453
```

[36]:
```
camp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2239 entries, 0 to 2238
Data columns (total 27 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   ID                   2239 non-null   int64
 1   Year_Birth           2239 non-null   int64
 2   Education            2239 non-null   object
 3   Marital_Status       2239 non-null   object
 4   Income               2239 non-null   object
 5   Kidhome              2239 non-null   int64
 6   Teenhome             2239 non-null   int64
 7   Dt_Customer          2239 non-null   object
```

```
 8   Recency              2239 non-null   int64
 9   MntWines             2239 non-null   int64
 10  MntFruits            2239 non-null   int64
 11  MntMeatProducts      2239 non-null   int64
 12  MntFishProducts      2239 non-null   int64
 13  MntSweetProducts     2239 non-null   int64
 14  MntGoldProds         2239 non-null   int64
 15  NumDealsPurchases    2239 non-null   int64
 16  NumWebPurchases      2239 non-null   int64
 17  NumCatalogPurchases  2239 non-null   int64
 18  NumStorePurchases    2239 non-null   int64
 19  NumWebVisitsMonth    2239 non-null   int64
 20  AcceptedCmp3         2239 non-null   int64
 21  AcceptedCmp4         2239 non-null   int64
 22  AcceptedCmp5         2239 non-null   int64
 23  AcceptedCmp1         2239 non-null   int64
 24  AcceptedCmp2         2239 non-null   int64
 25  Complain             2239 non-null   int64
 26  Country              2239 non-null   object
dtypes: int64(22), object(5)
memory usage: 472.4+ KB
```

[37]:
```
camp.duplicated().sum()
```

[37]:
```
0
```

[38]:
```
camp.isna().sum()
```

[38]:
```
ID                     0
Year_Birth             0
Education              0
Marital_Status         0
Income                 0
Kidhome                0
Teenhome               0
Dt_Customer            0
Recency                0
MntWines               0
MntFruits              0
MntMeatProducts        0
MntFishProducts        0
MntSweetProducts       0
MntGoldProds           0
NumDealsPurchases      0
NumWebPurchases        0
NumCatalogPurchases    0
NumStorePurchases      0
```

```
NumWebVisitsMonth      0
AcceptedCmp3           0
AcceptedCmp4           0
AcceptedCmp5           0
AcceptedCmp1           0
AcceptedCmp2           0
Complain               0
Country                0
dtype: int64
```

[39]:
```
camp.describe()
```

[39]:

	ID	Year_Birth	Kidhome	Teenhome	Recency
count	2239.000000	2239.000000	2239.000000	2239.000000	2239.000000
mean	5590.444841	1968.802144	0.443948	0.506476	49.121036
std	3246.372471	11.985494	0.538390	0.544555	28.963662
min	0.000000	1893.000000	0.000000	0.000000	0.000000
25%	2827.500000	1959.000000	0.000000	0.000000	24.000000
50%	5455.000000	1970.000000	0.000000	0.000000	49.000000
75%	8423.500000	1977.000000	1.000000	1.000000	74.000000
max	11191.000000	1996.000000	2.000000	2.000000	99.000000

	MntWines	MntFruits	MntMeatProducts	MntFishProducts
count	2239.000000	2239.000000	2239.000000	2239.000000
mean	304.067441	26.307727	167.016525	37.538633
std	336.614830	39.781468	225.743829	54.637617
min	0.000000	0.000000	0.000000	0.000000
25%	24.000000	1.000000	16.000000	3.000000
50%	174.000000	8.000000	67.000000	12.000000
75%	504.500000	33.000000	232.000000	50.000000
max	1493.000000	199.000000	1725.000000	259.000000

	MntSweetProducts	…	NumWebPurchases	NumCatalogPurchases
count	2239.000000	…	2239.000000	2239.000000
mean	27.074587	…	4.085306	2.662796
std	41.286043	…	2.779240	2.923542
min	0.000000	…	0.000000	0.000000
25%	1.000000	…	2.000000	0.000000
50%	8.000000	…	4.000000	2.000000
75%	33.000000	…	6.000000	4.000000
max	263.000000	…	27.000000	28.000000

	NumStorePurchases	NumWebVisitsMonth	AcceptedCmp3	AcceptedCmp4
count	2239.000000	2239.000000	2239.000000	2239.000000
mean	5.791425	5.316213	0.072800	0.074587
std	3.251149	2.427144	0.259867	0.262782
min	0.000000	0.000000	0.000000	0.000000

```
25%              3.000000           3.000000      0.000000      0.000000
50%              5.000000           6.000000      0.000000      0.000000
75%              8.000000           7.000000      0.000000      0.000000
max             13.000000          20.000000      1.000000      1.000000

       AcceptedCmp5  AcceptedCmp1  AcceptedCmp2     Complain
count   2239.000000   2239.000000   2239.000000  2239.000000
mean       0.072800      0.064314      0.013399     0.009379
std        0.259867      0.245367      0.115001     0.096412
min        0.000000      0.000000      0.000000     0.000000
25%        0.000000      0.000000      0.000000     0.000000
50%        0.000000      0.000000      0.000000     0.000000
75%        0.000000      0.000000      0.000000     0.000000
max        1.000000      1.000000      1.000000     1.000000

[8 rows x 22 columns]
```

3.0.1 Feature Engineering

[40]:
```
camp['Income'] = camp['Income'].astype(str).str.replace('$', '', regex=False).
 ↪str.replace(',', '', regex=False)
camp['Income'] = camp['Income'].astype(float)
```

3.0.2 Non Graphical Analysis

[41]:
```
col=["Education","Marital_Status","Kidhome","NumDealsPurchases","NumWebPurchases",
 "NumWebVisitsMonth","NumCatalogPurchases","AcceptedCmp1","AcceptedCmp3",
 "AcceptedCmp3","AcceptedCmp4","AcceptedCmp5","Complain","Country"]
for i in col:
 print(camp.value_counts(i))
 print()
 print("*"*100)
```

```
Education
Graduation    1126
PhD            486
Master         370
2n Cycle       203
Basic           54
Name: count, dtype: int64

****************************************************************************************************
********************
Marital_Status
Married     864
Together    579
Single      480
```

```
Divorced    232
Widow        77
Alone         3
Absurd        2
YOLO          2
Name: count, dtype: int64

**********************************************************************************
********************
Kidhome
0    1293
1     898
2      48
Name: count, dtype: int64

**********************************************************************************
********************
NumDealsPurchases
1     970
2     497
3     297
4     188
5      94
6      61
0      46
7      40
8      14
9       8
15      7
10      5
11      5
12      4
13      3
Name: count, dtype: int64

**********************************************************************************
********************
NumWebPurchases
2     373
1     354
3     335
4     280
5     220
6     205
7     155
8     102
9      75
0      49
```

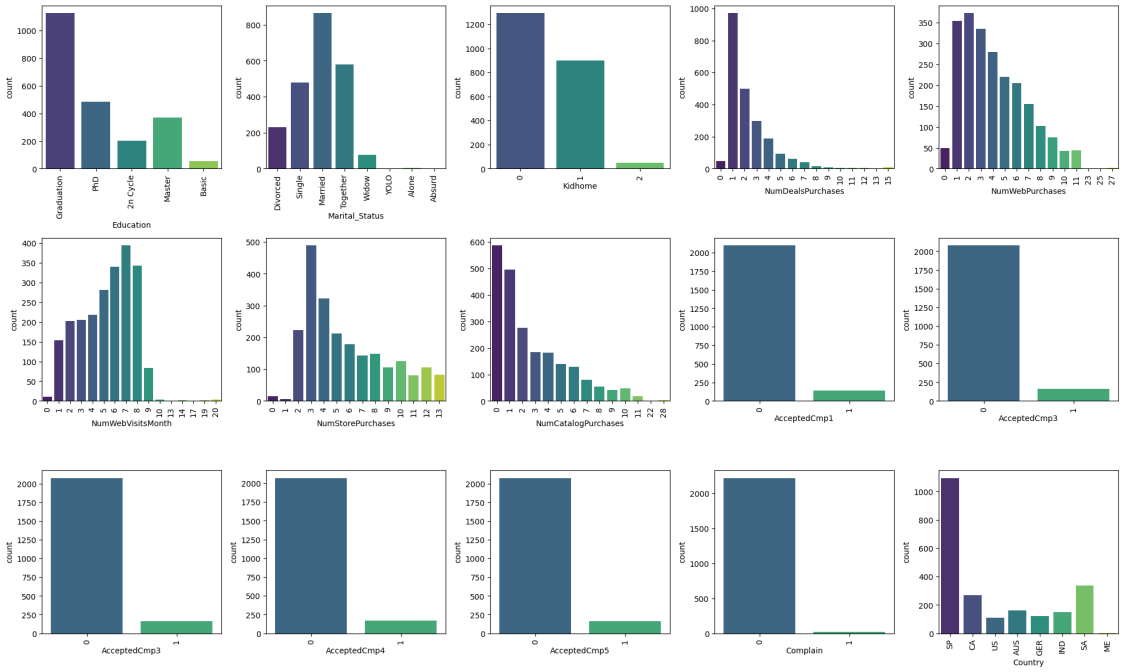
```
11     44
10     43
27      2
23      1
25      1
Name: count, dtype: int64

********************************************************************************
********************
NumWebVisitsMonth
7     393
8     342
6     339
5     281
4     218
3     205
2     202
1     153
9      83
0      11
10      3
20      3
14      2
19      2
13      1
17      1
Name: count, dtype: int64

********************************************************************************
********************
NumCatalogPurchases
0     586
1     496
2     276
3     184
4     182
5     140
6     128
7      79
8      55
10     48
9      42
11     19
28      3
22      1
Name: count, dtype: int64

********************************************************************************
```

```
********************
AcceptedCmp1
0    2095
1     144
Name: count, dtype: int64

********************************************************************************
********************
AcceptedCmp3
0    2076
1     163
Name: count, dtype: int64

********************************************************************************
********************
AcceptedCmp3
0    2076
1     163
Name: count, dtype: int64

********************************************************************************
********************
AcceptedCmp4
0    2072
1     167
Name: count, dtype: int64

********************************************************************************
********************
AcceptedCmp5
0    2076
1     163
Name: count, dtype: int64

********************************************************************************
********************
Complain
0    2218
1      21
Name: count, dtype: int64

********************************************************************************
********************
Country
SP     1095
SA      336
CA      268
AUS     160
```

```
IND      148
GER      120
US       109
ME         3
Name: count, dtype: int64

**************************************************************************************
********************
```

3.0.3 Graphical Analysis

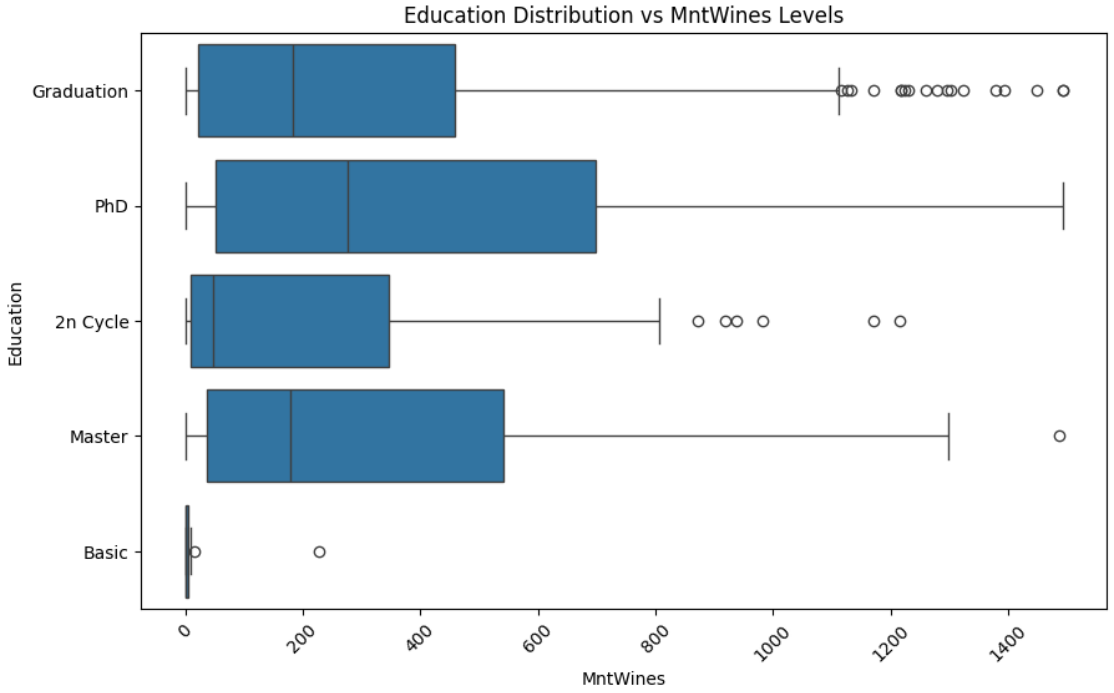
[43]:
```
fig, axs = plt.subplots(nrows=3, ncols=5, figsize=(20, 12))
columns = ["Education", "Marital_Status", "Kidhome", "NumDealsPurchases",␣
 ↪"NumWebPurchases",
 "NumWebVisitsMonth",'NumStorePurchases', "NumCatalogPurchases",␣
 ↪"AcceptedCmp1", "AcceptedCmp3",
 "AcceptedCmp3", "AcceptedCmp4", "AcceptedCmp5", "Complain", "Country"]
for ax, col in zip(axs.flatten(), columns):
 sns.countplot(data=camp, x=col, ax=ax, palette='viridis')
 ax.tick_params(axis='x', rotation=90)
plt.tight_layout()
plt.show()
```

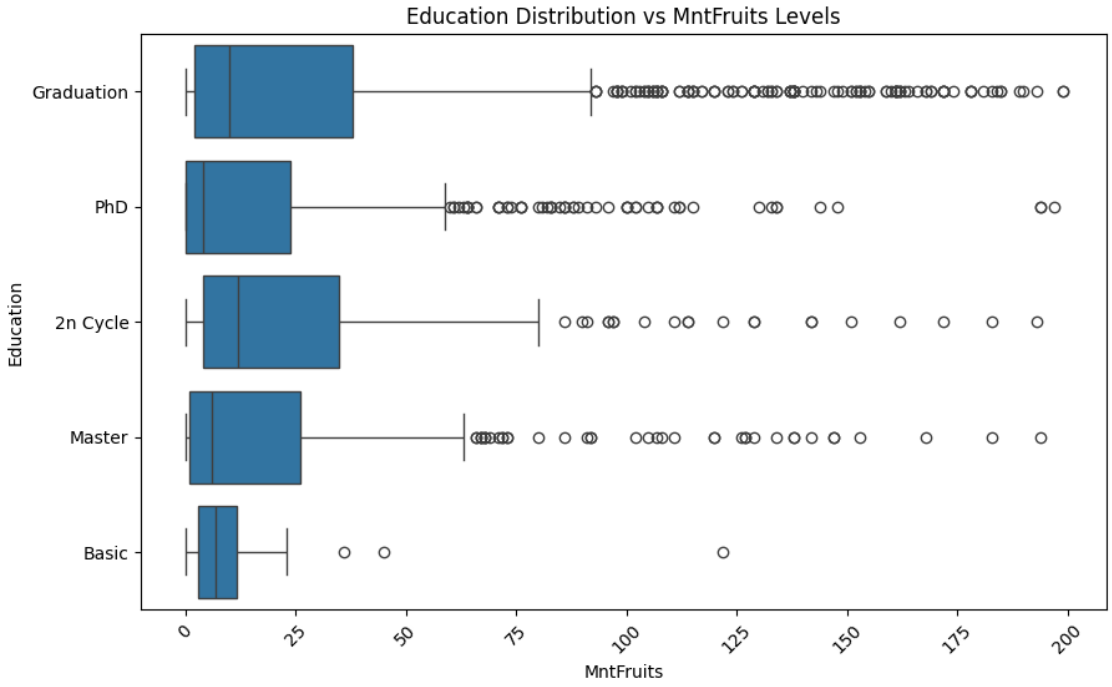


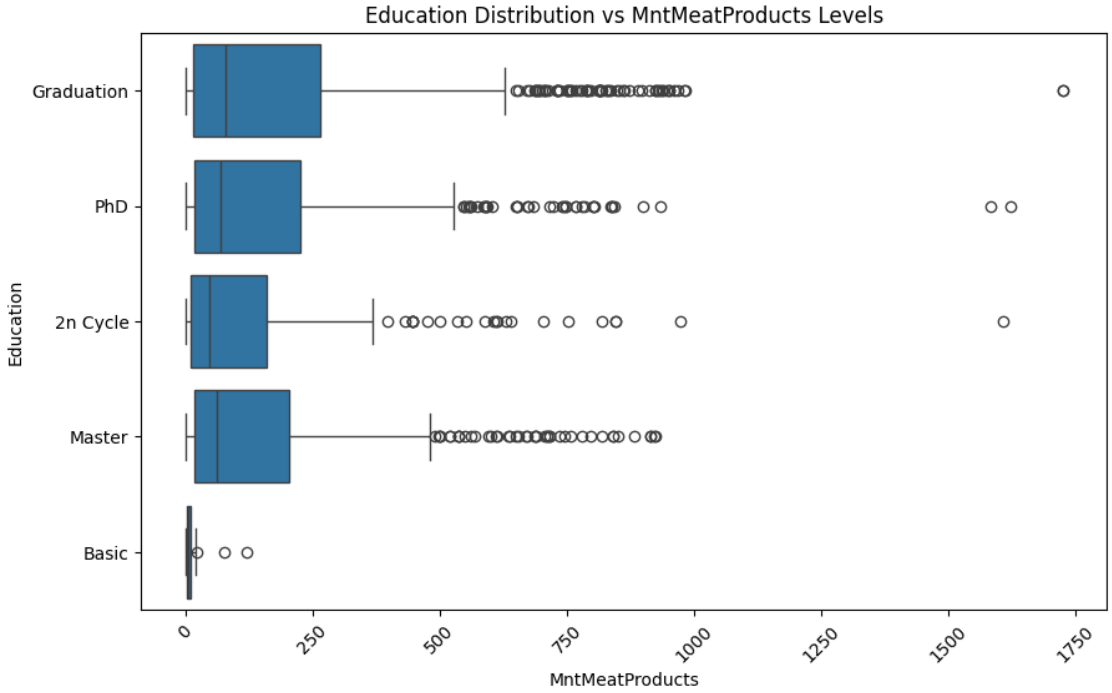
- The majority of customers are graduates.
- Most customers are married and living together.
- Most people have one child.

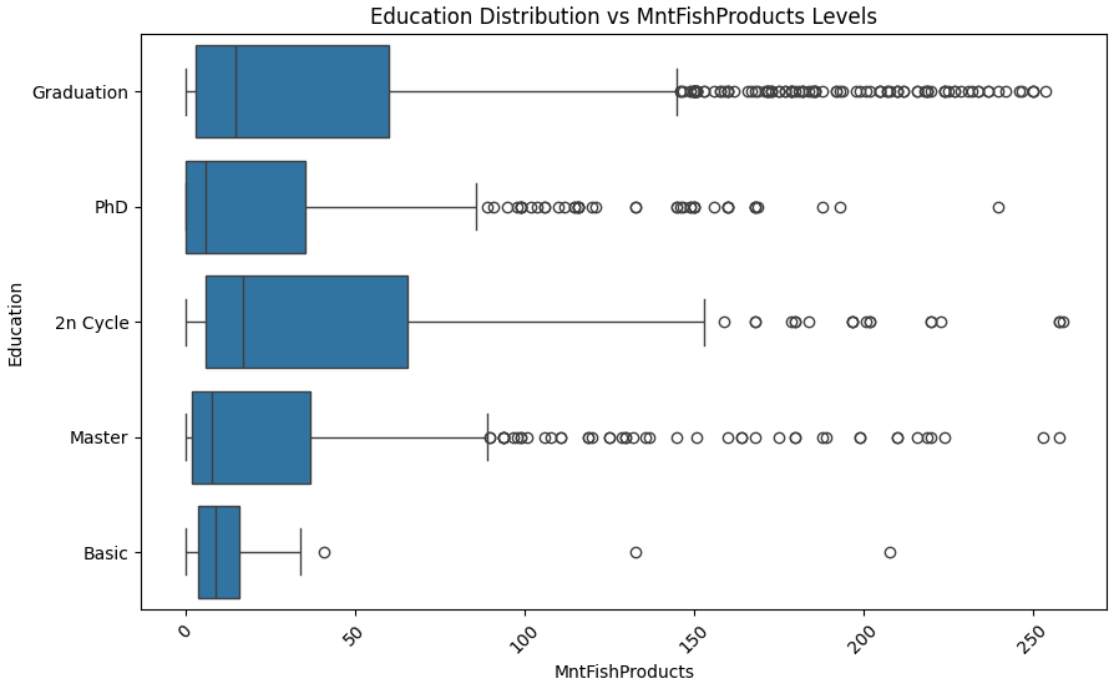
- Customers prefer buying one product, indicating a right-skewed distribution.
- People mostly buy two products from the website, with a slight difference between those buying one and three products.
- Customers tend to buy three products directly from the store.
- The majority of people do not prefer making purchases using a catalog.
- Customer acceptance of offers remains almost consistent until Campaign 4, but there is a sudden drop in Campaign 5.
- There are very few complaints, almost around 1-2%.
- Most purchases are made by customers from Spain.

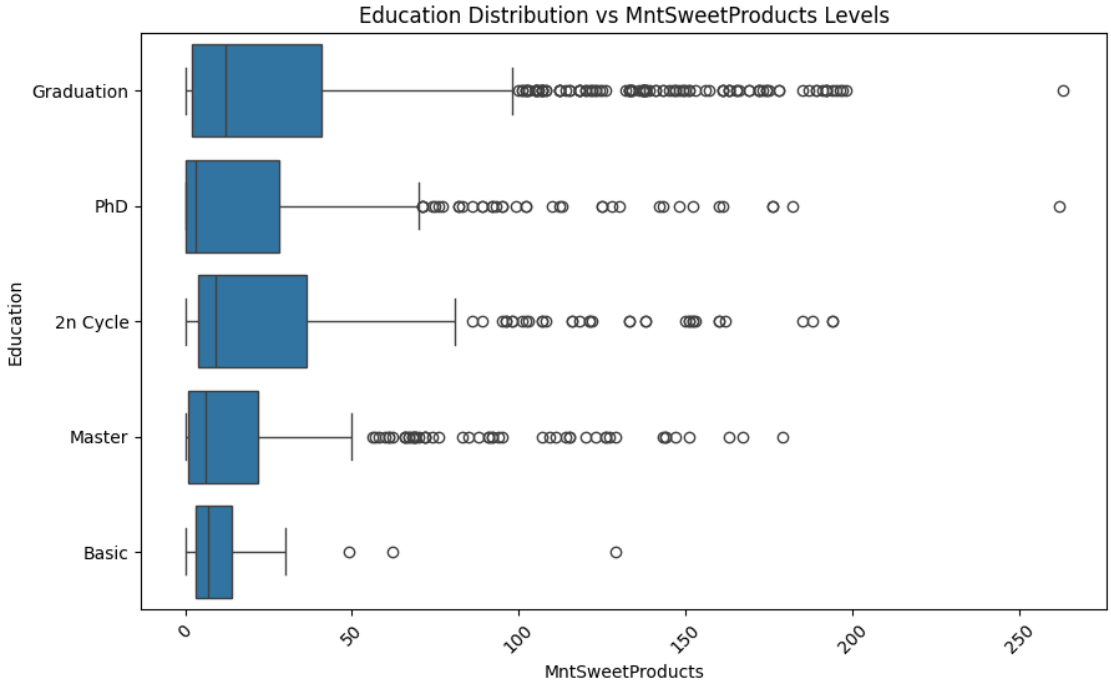
[45]:
```
col=['MntWines', 'MntFruits','MntMeatProducts', 'MntFishProducts',␣
 ↪'MntSweetProducts','MntGoldProds']
for i in col:
 plt.figure(figsize=(10, 6))
 sns.boxplot( x=i,y='Education', data=camp)
 plt.title(f'Education Distribution vs {i} Levels')
 plt.xticks(rotation=45)
 plt.show()
```

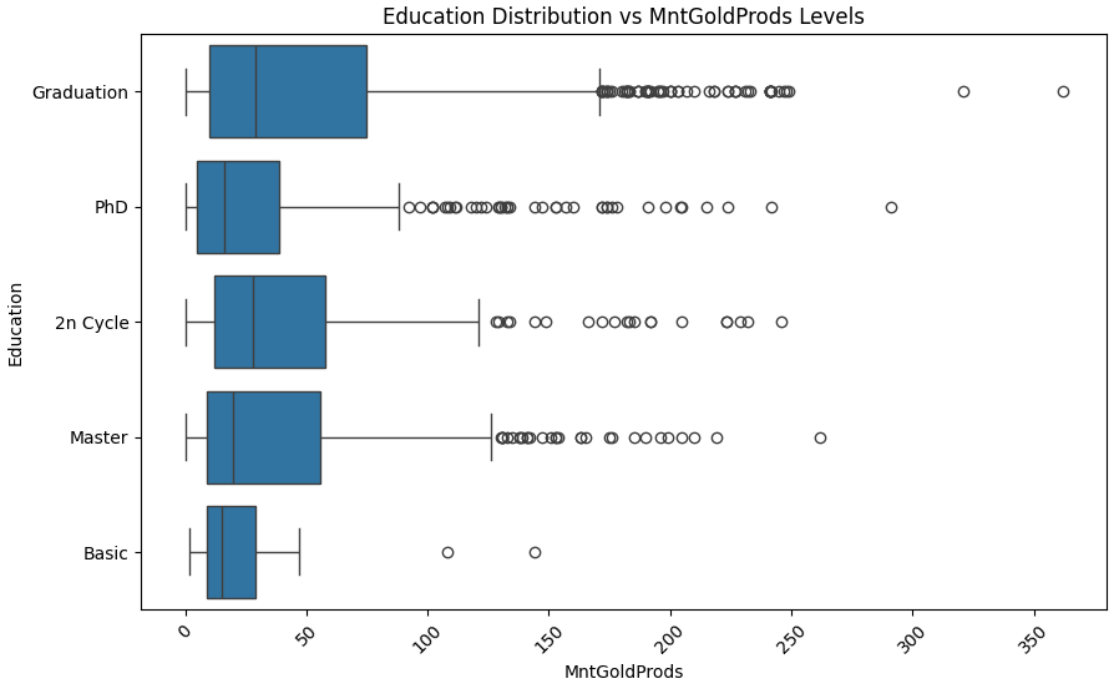



Education Distribution vs MntFruits Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
25
50
75
100
125
150
175
200
MntFruits


Education Distribution vs MntMeatProducts Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
250
500
750
1000
1250
1500
1750
MntMeatProducts

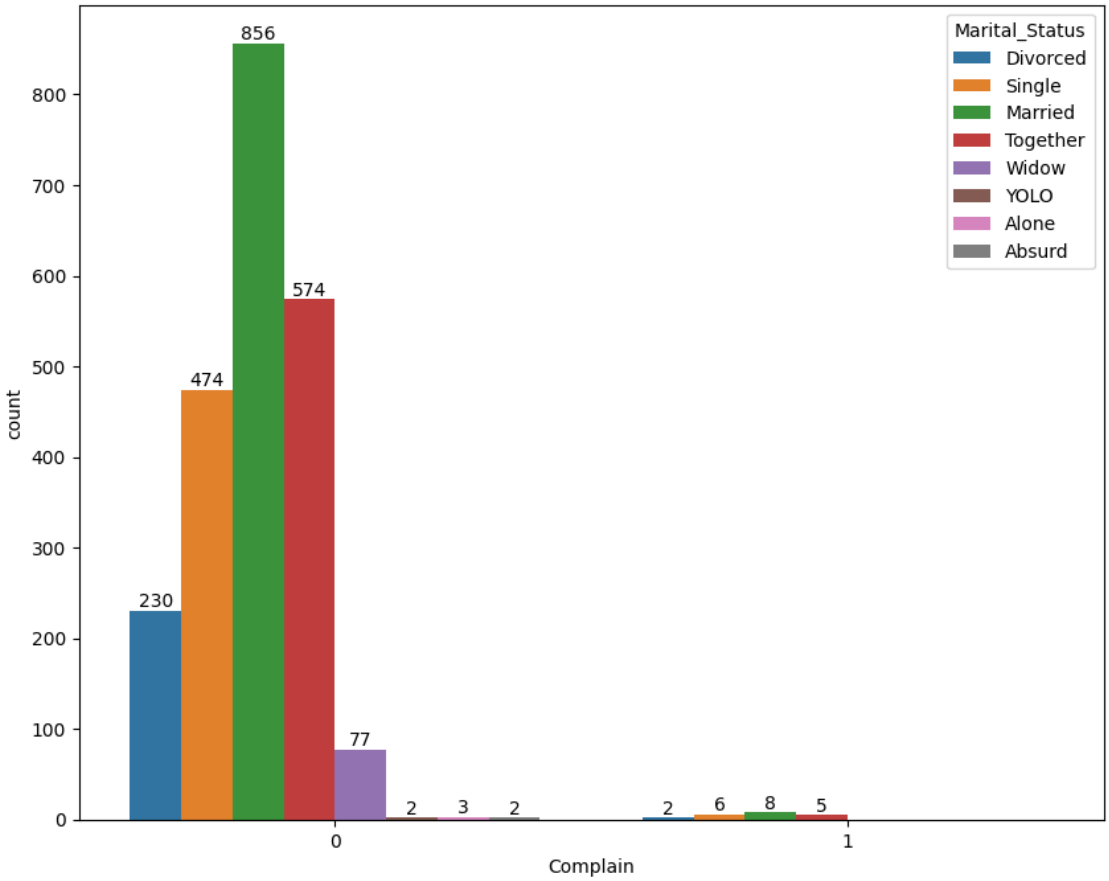

Education Distribution vs MntFishProducts Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
50
100
150
200
250
MntFishProducts


Education Distribution vs MntSweetProducts Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
50
100
150
200
250
MntSweetProducts

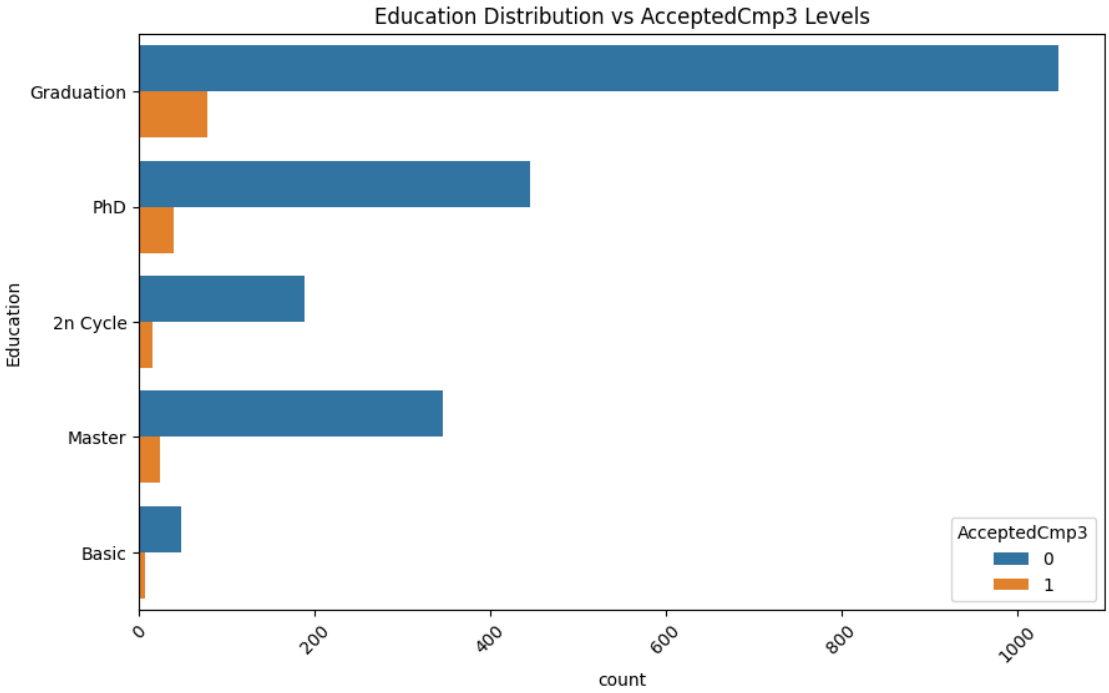


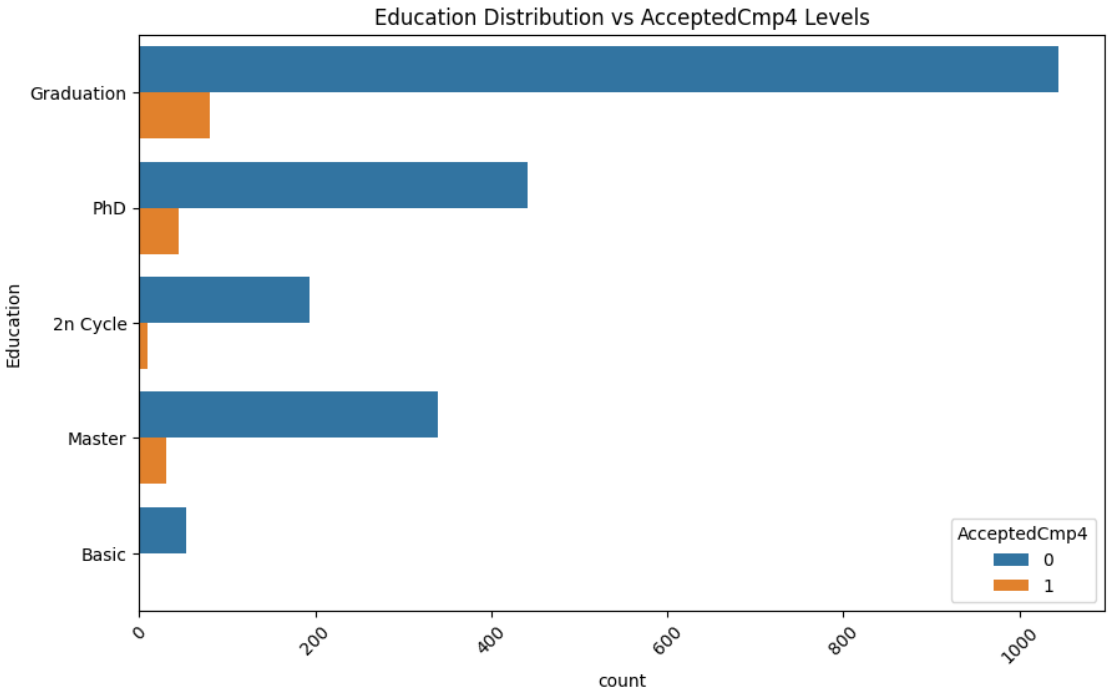
- People with a PhD degree spend most of their money on wine.
- Except for those with only basic education, the rest of the educated individuals tend to spend money on meat.
- It appears that customers with a graduation or second-cycle degree spend the most on fish products.
- Graduates primarily spend their money on gold.

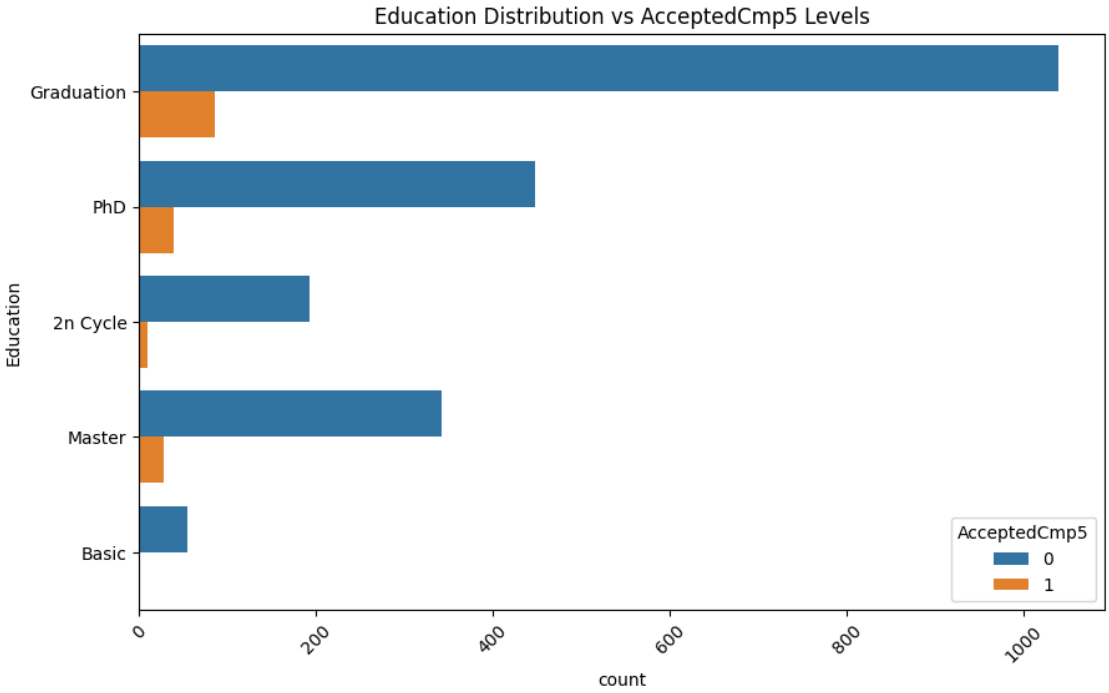
[46]:
```
plt.figure(figsize=(10,8))
sns.countplot(data=camp,x='Complain',hue='Marital_Status')
ax=plt.gca()
for i in ax.containers:
 ax.bar_label(i)
plt.show()
```

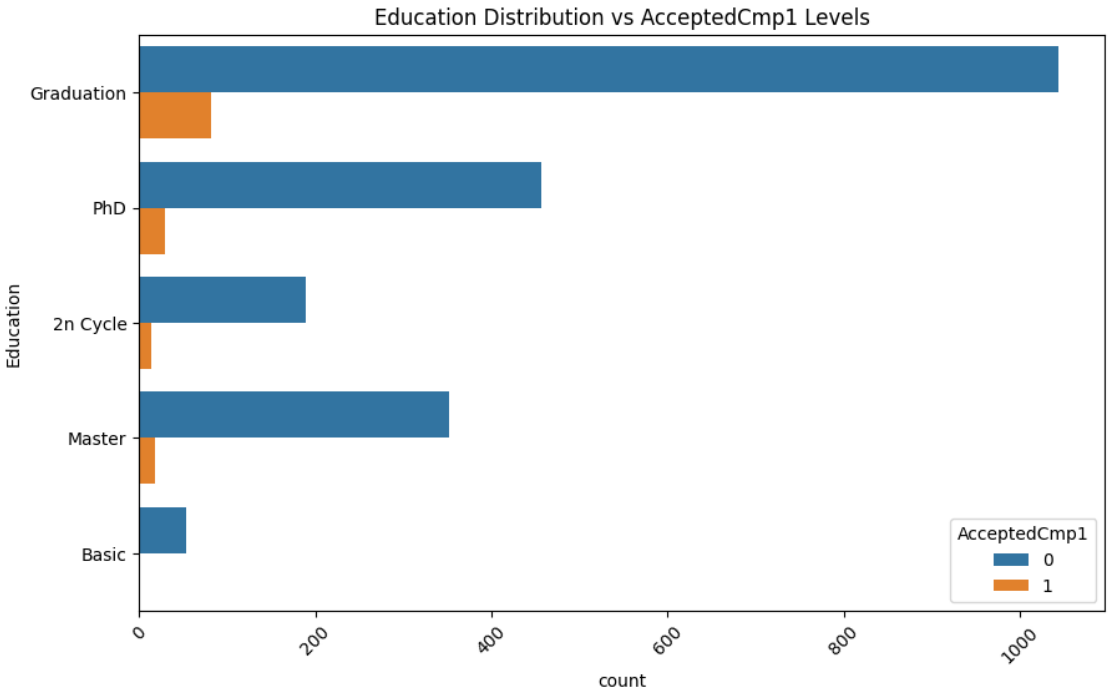


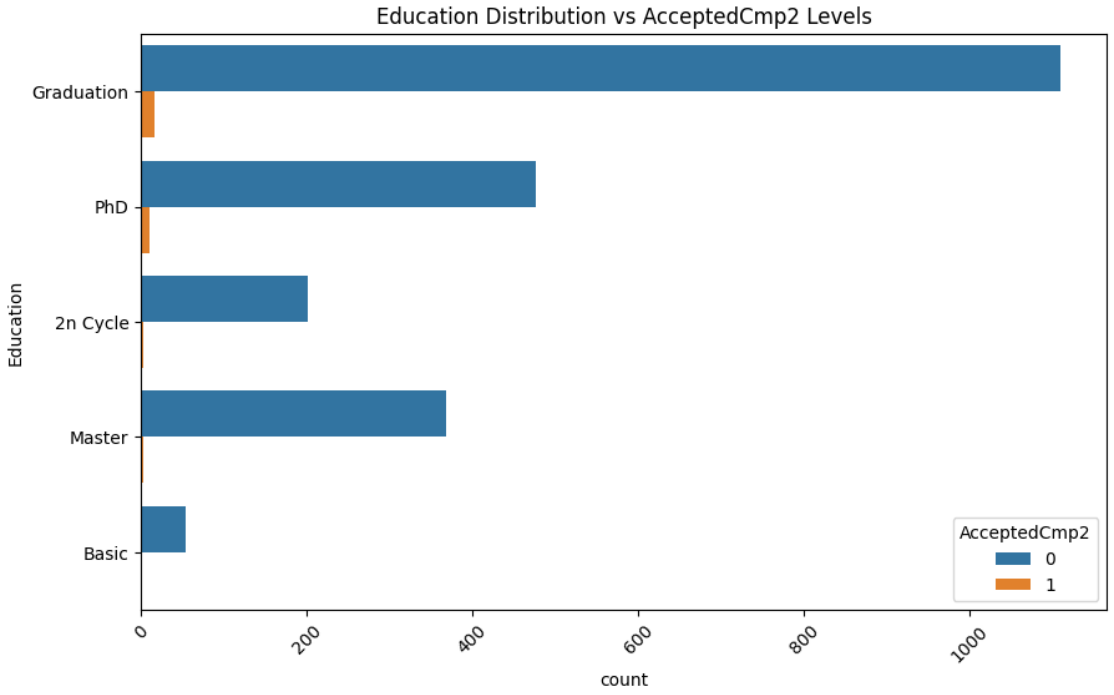
[47]:
```
col=['AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5',␣
 ↪'AcceptedCmp1','AcceptedCmp2']
for i in col:
 plt.figure(figsize=(10, 6))
 sns.countplot( hue=i,y='Education', data=camp)
 plt.title(f'Education Distribution vs {i} Levels')
 plt.xticks(rotation=45)
 plt.show()
```


Education Distribution vs AcceptedCmp3 Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
200
400
600
800
1000
count
AcceptedCmp3
0
1


Education Distribution vs AcceptedCmp4 Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
200
400
600
800
1000
count
AcceptedCmp4
0
1

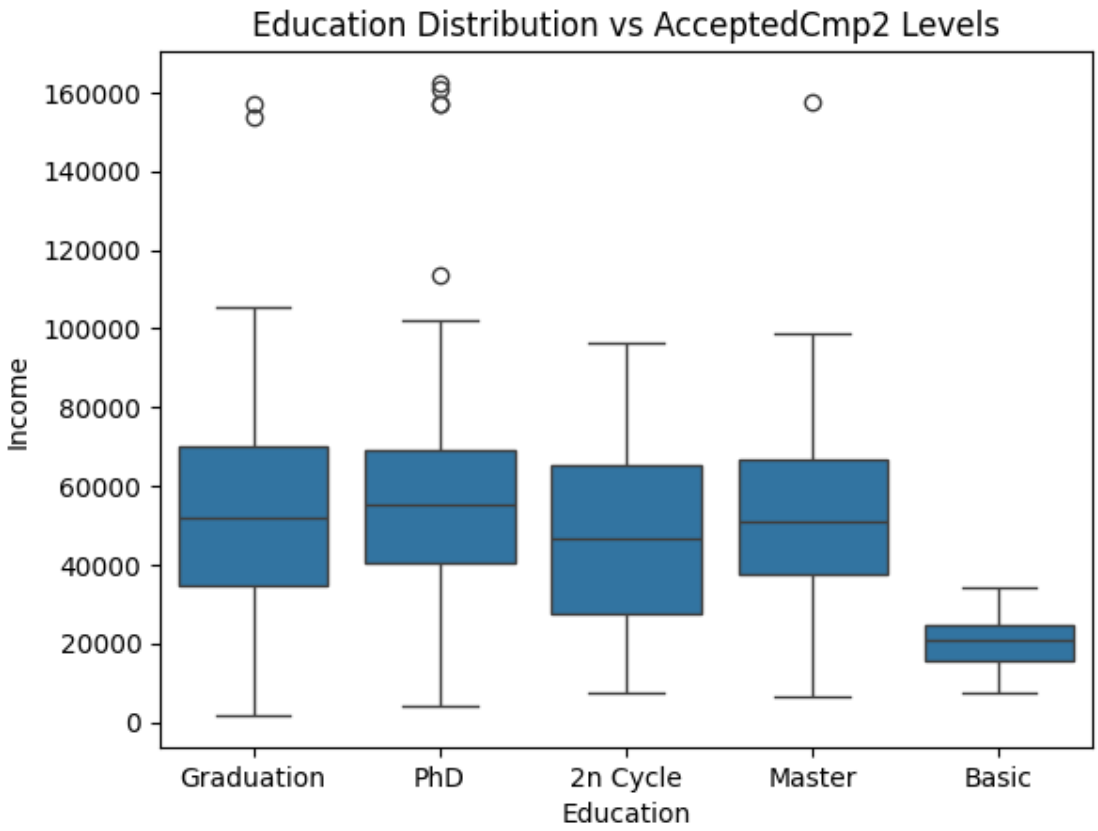

Education Distribution vs AcceptedCmp5 Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
200
400
600
800
1000
count
AcceptedCmp5
0
1


Education Distribution vs AcceptedCmp1 Levels
Graduation
PhD
2n Cycle
Master
Basic
Education
0
200
400
600
800
1000
count
AcceptedCmp1
0
1



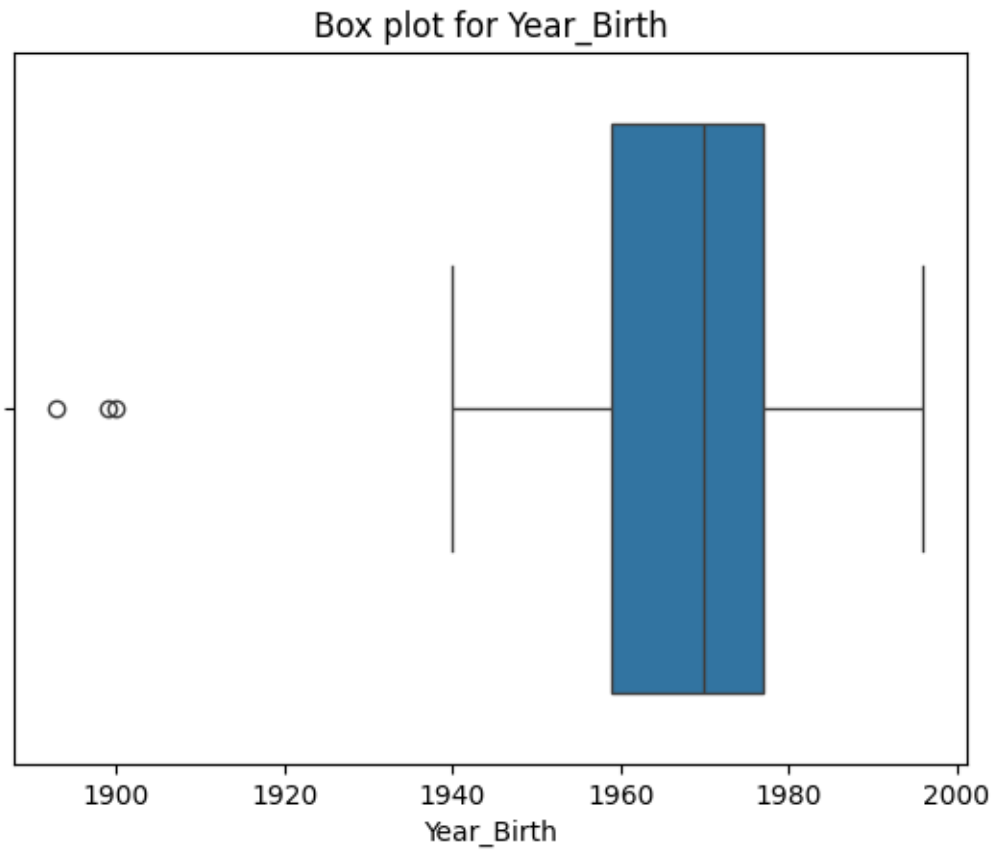
[49]:
```
sns.boxplot( x='Education',y='Income', data=camp)
plt.title(f'Education Distribution vs {i} Levels')
ax=plt.gca()
for i in ax.containers:
 ax.bar_label(i)
```

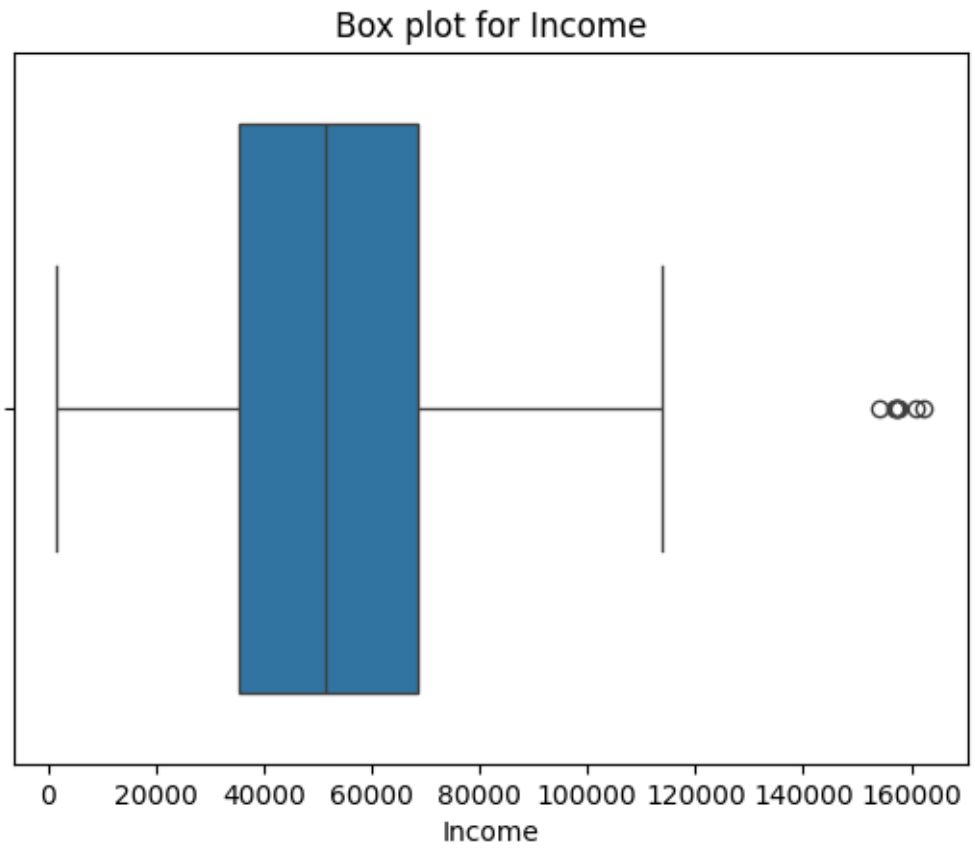
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-49-5754ff88bafb> in <cell line: 4>()
      3 ax=plt.gca()
      4 for i in ax.containers:
----> 5  ax.bar_label(i)

/usr/local/lib/python3.10/dist-packages/matplotlib/axes/_axes.py in␣
 ↪bar_label(self, container, labels, fmt, label_type, padding, **kwargs)
   2720         _api.check_in_list(['edge', 'center'], label_type=label_type)
   2721
-> 2722         bars = container.patches
   2723         errorbar = container.errorbar
   2724         datavalues = container.datavalues

AttributeError: 'BoxPlotContainer' object has no attribute 'patches'
```

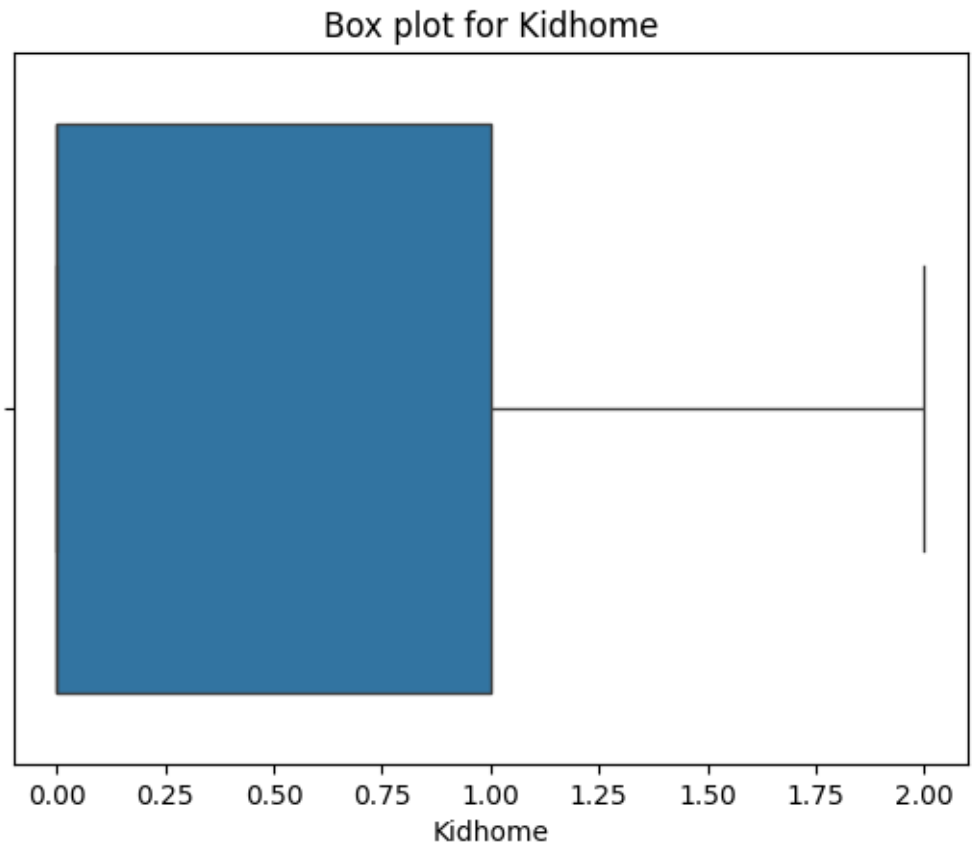


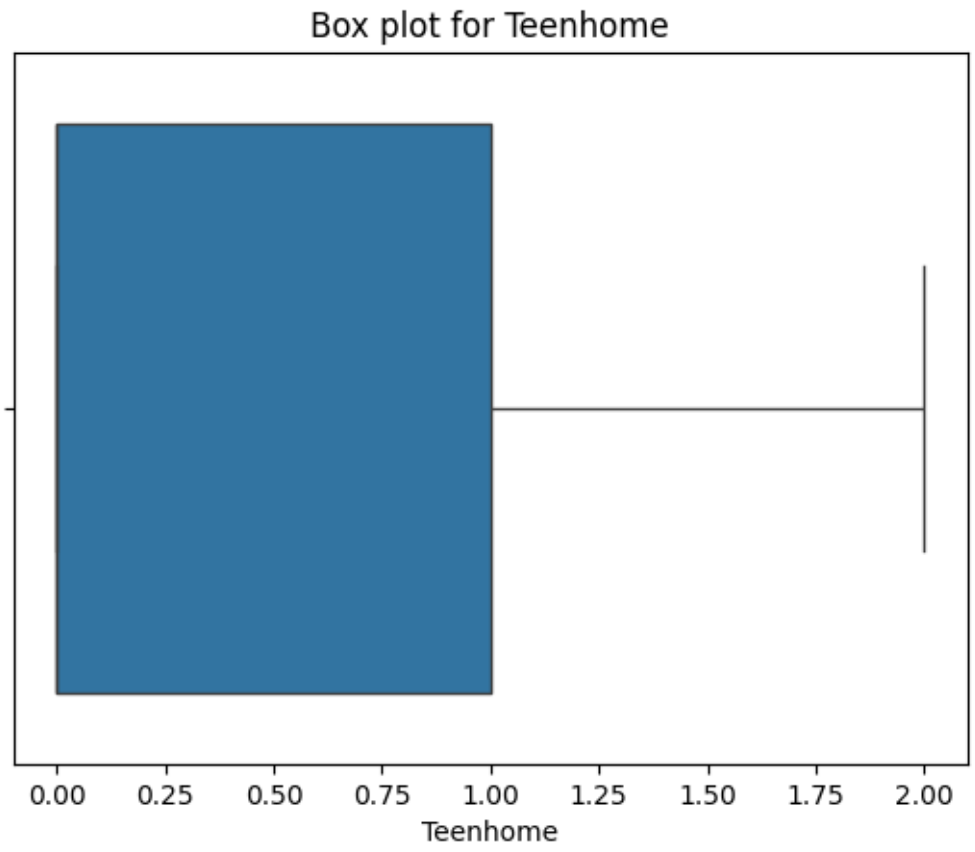
3.0.4 Outlier Detection using Boxplot

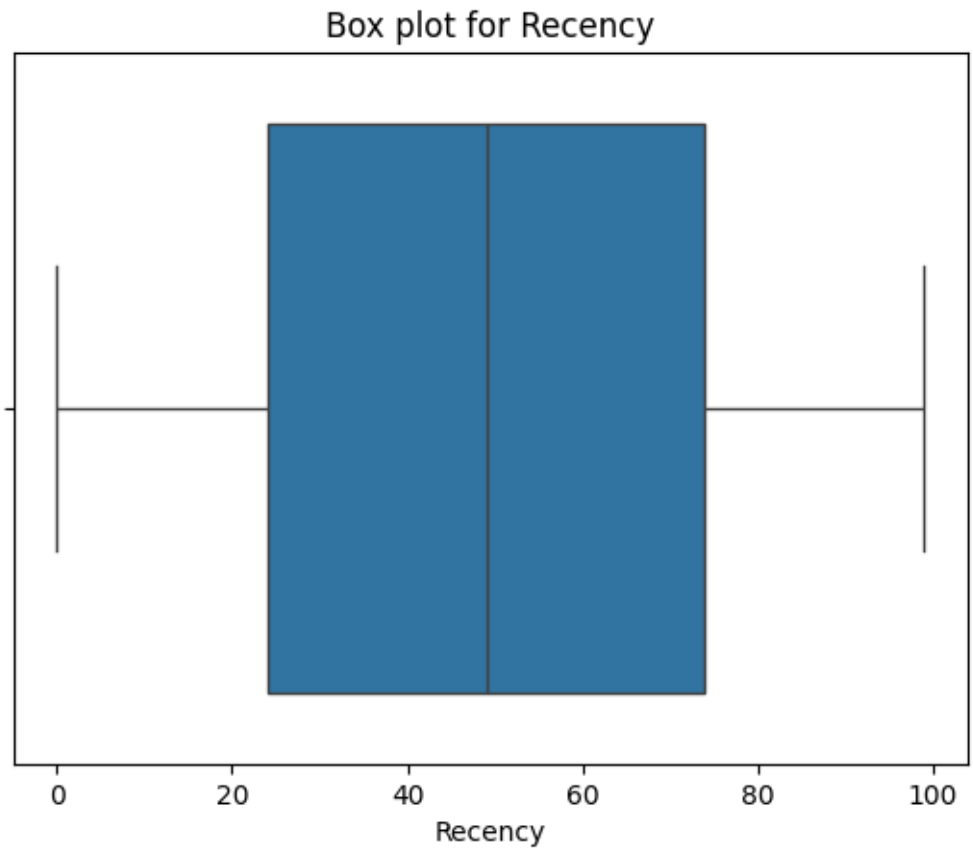
[50]:
```
col=['Year_Birth','Income', 'Kidhome',
 'Teenhome', 'Recency', 'MntWines', 'MntFruits',
 'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
 'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
 'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
 'AcceptedCmp2', 'Complain']
for i in col:
 sns.boxplot(data=camp,x=i)
 plt.title(f"Box plot for {i}")
 plt.show()
```

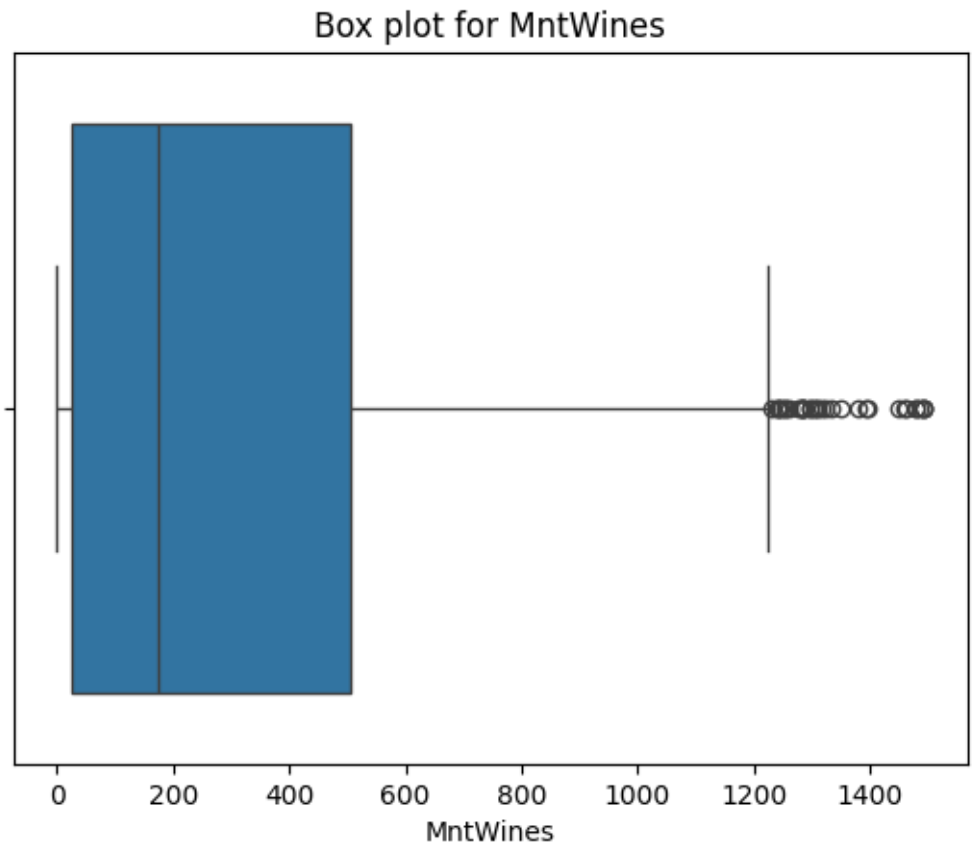

Box plot for Year_Birth
1900
1920
1940
1960
1980
2000
Year_Birth

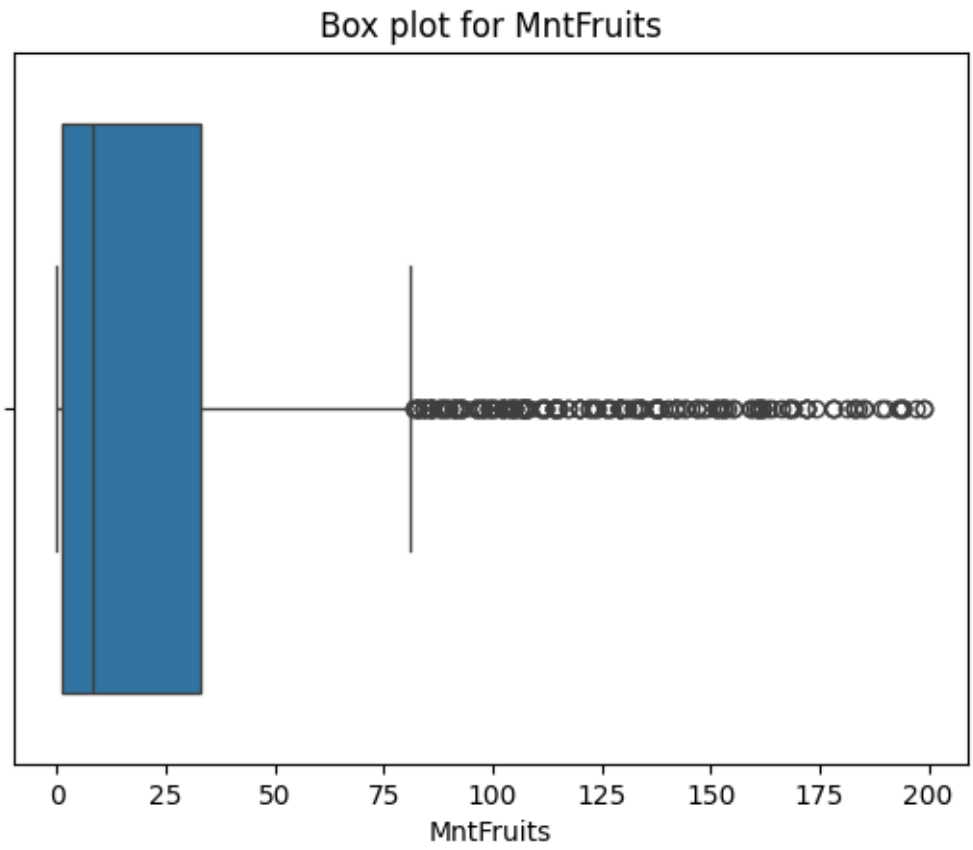

Box plot for Income
0
20000
40000
60000
80000
100000
120000
140000
160000
Income

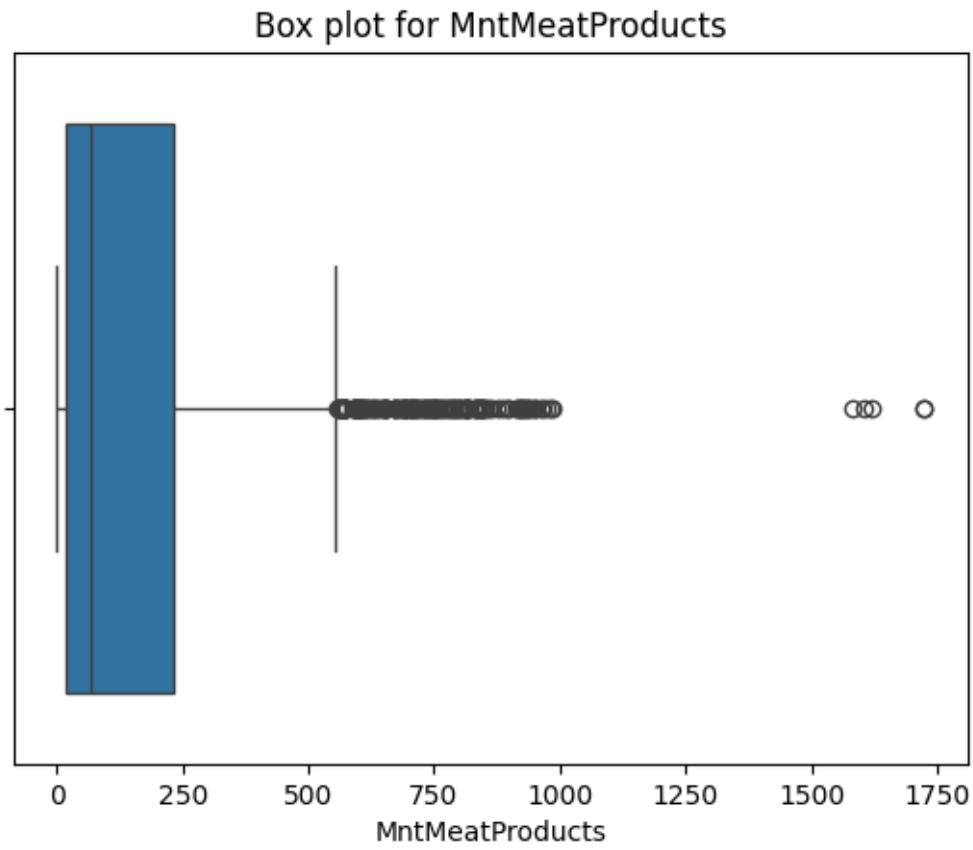

Box plot for Kidhome
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Kidhome

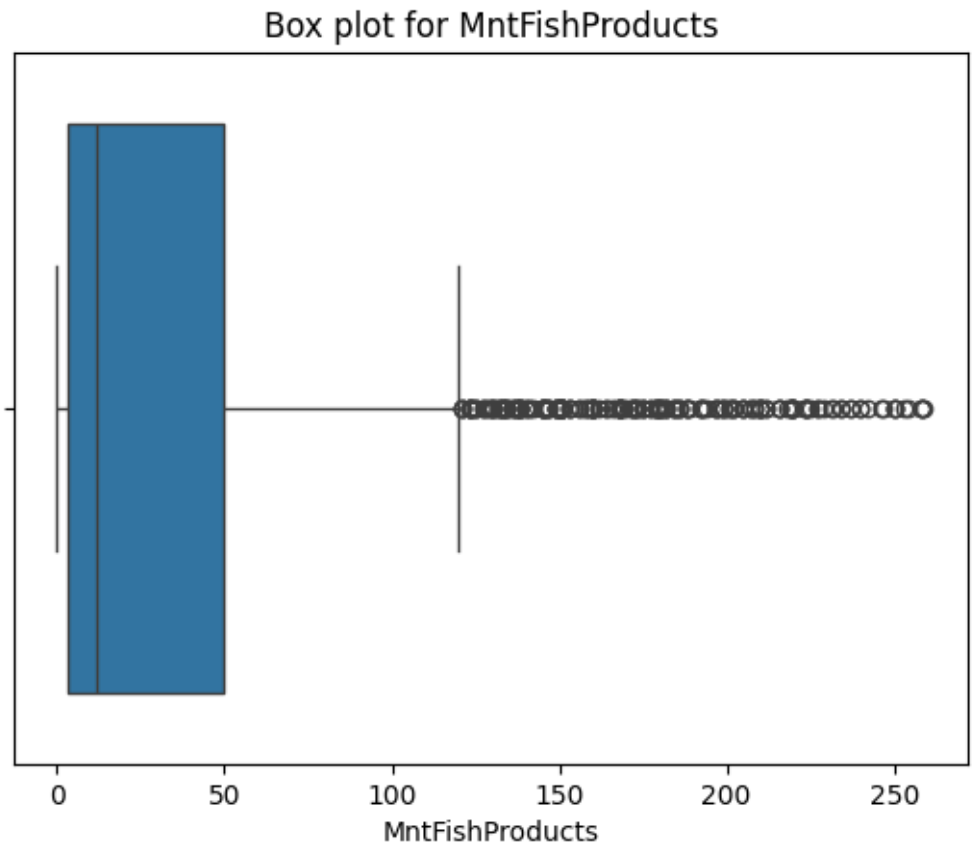

Box plot for Teenhome
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Teenhome

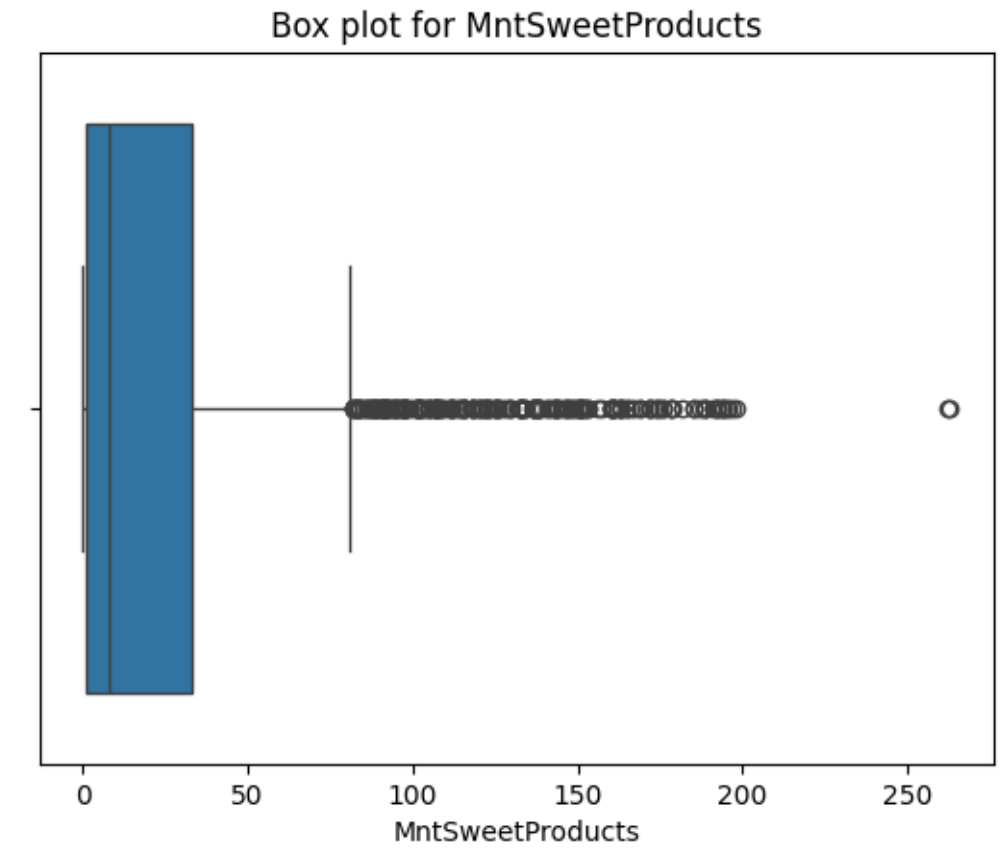

Box plot for Recency
0
20
40
60
80
100
Recency

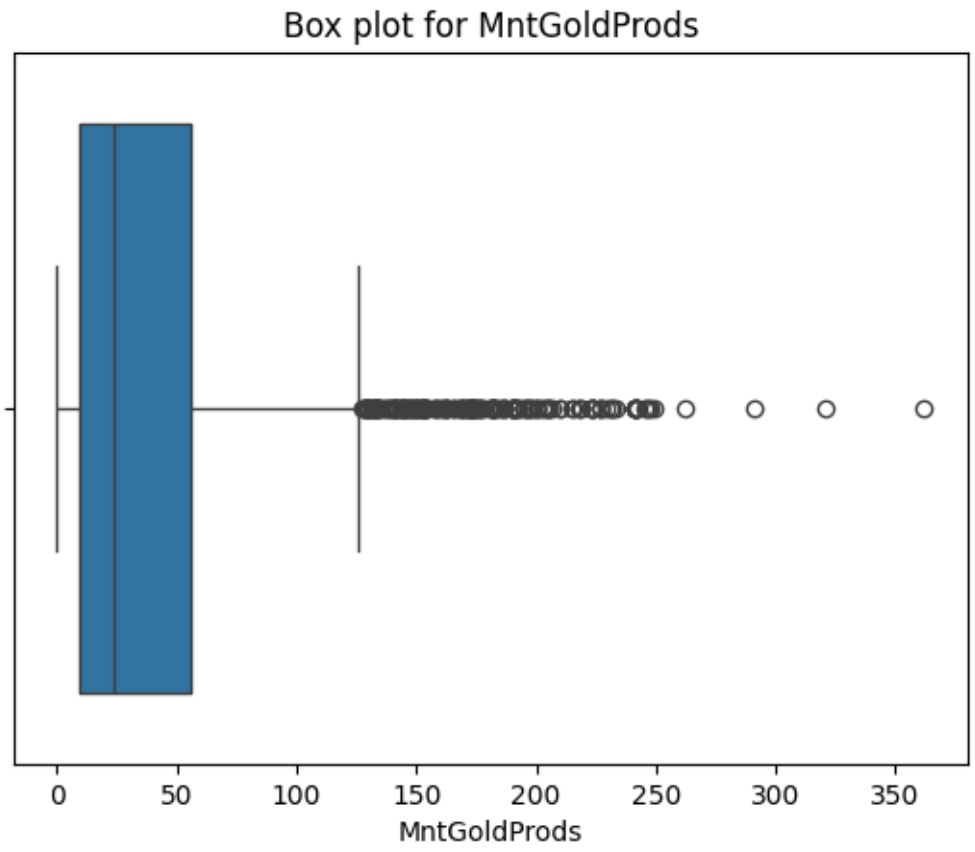

Box plot for MntWines
0
200
400
600
800
1000
1200
1400
MntWines

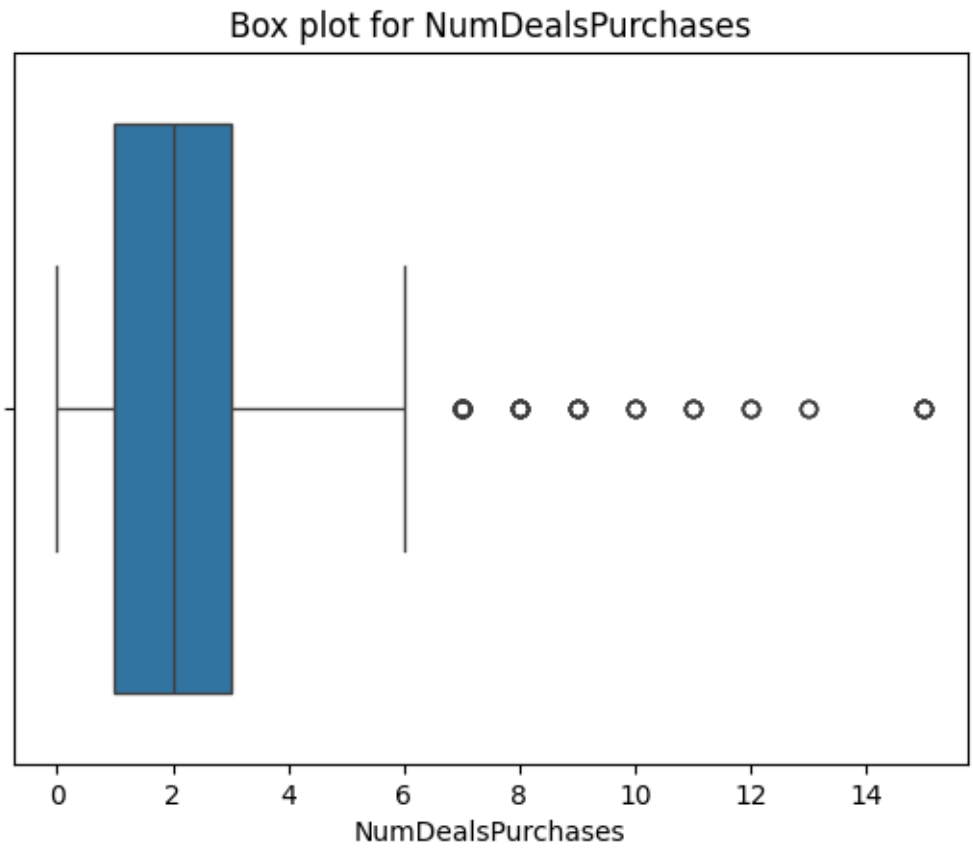

Box plot for MntFruits
0
25
50
75
100
125
150
175
200
MntFruits

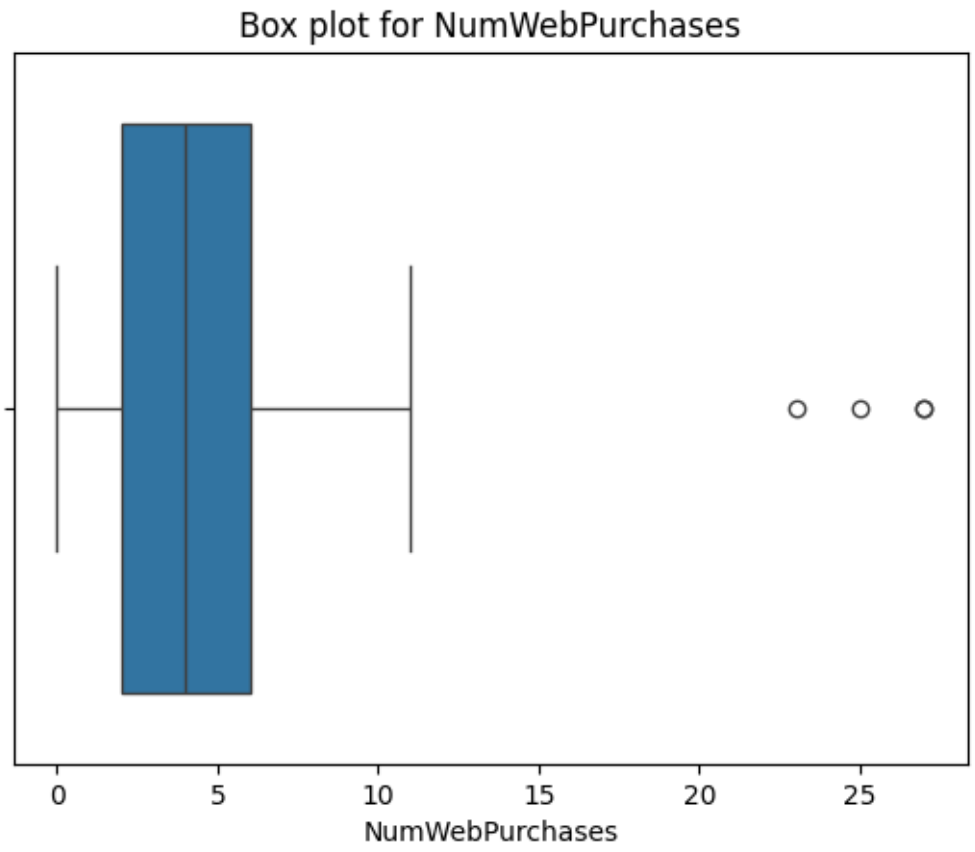

Box plot for MntMeatProducts
0
250
500
750
1000
1250
1500
1750
MntMeatProducts

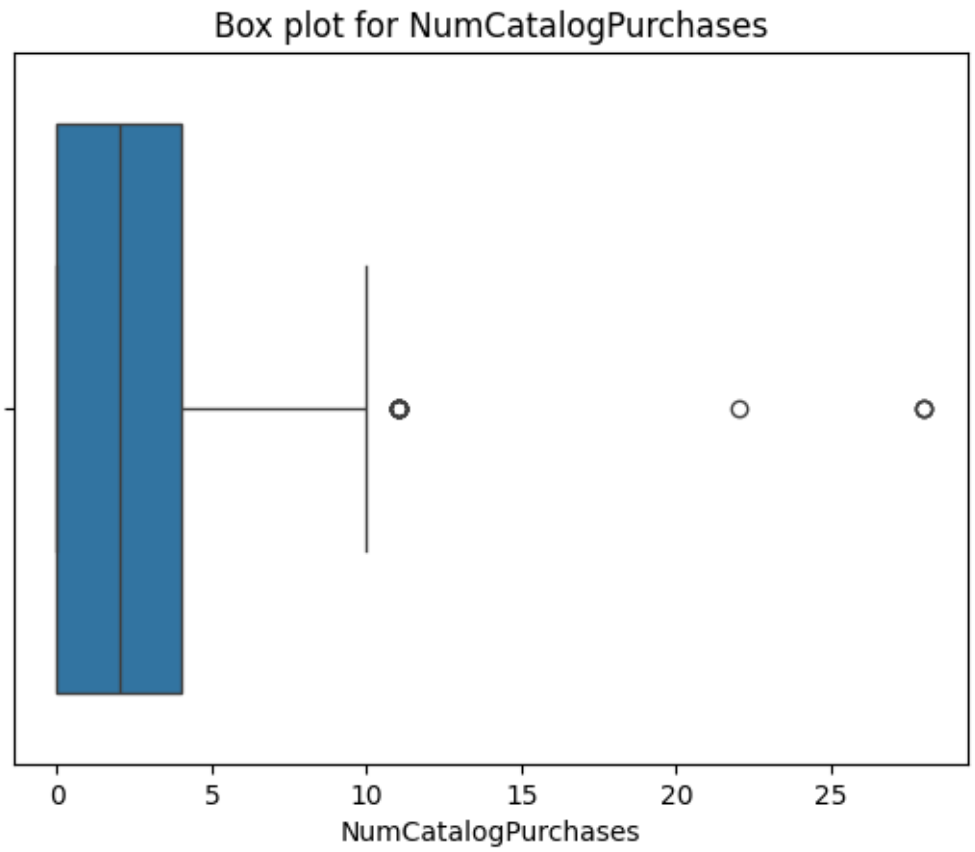

Box plot for MntFishProducts
0
50
100
150
200
250
MntFishProducts

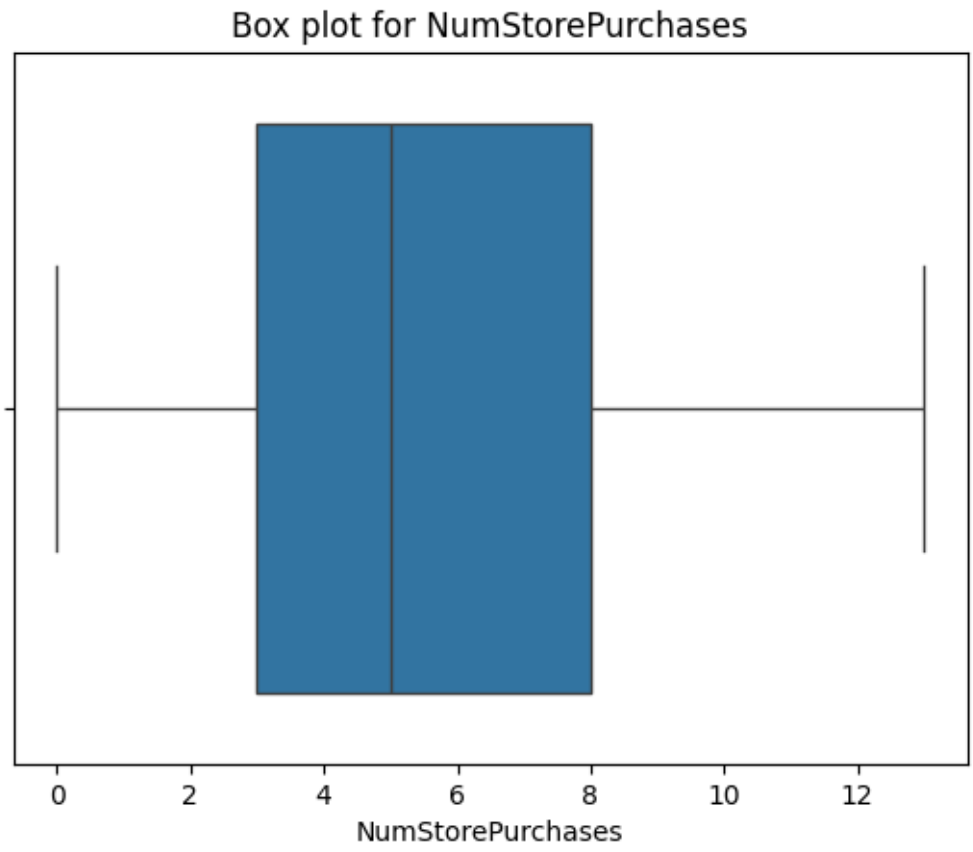

Box plot for MntSweetProducts
0
50
100
150
200
250
MntSweetProducts

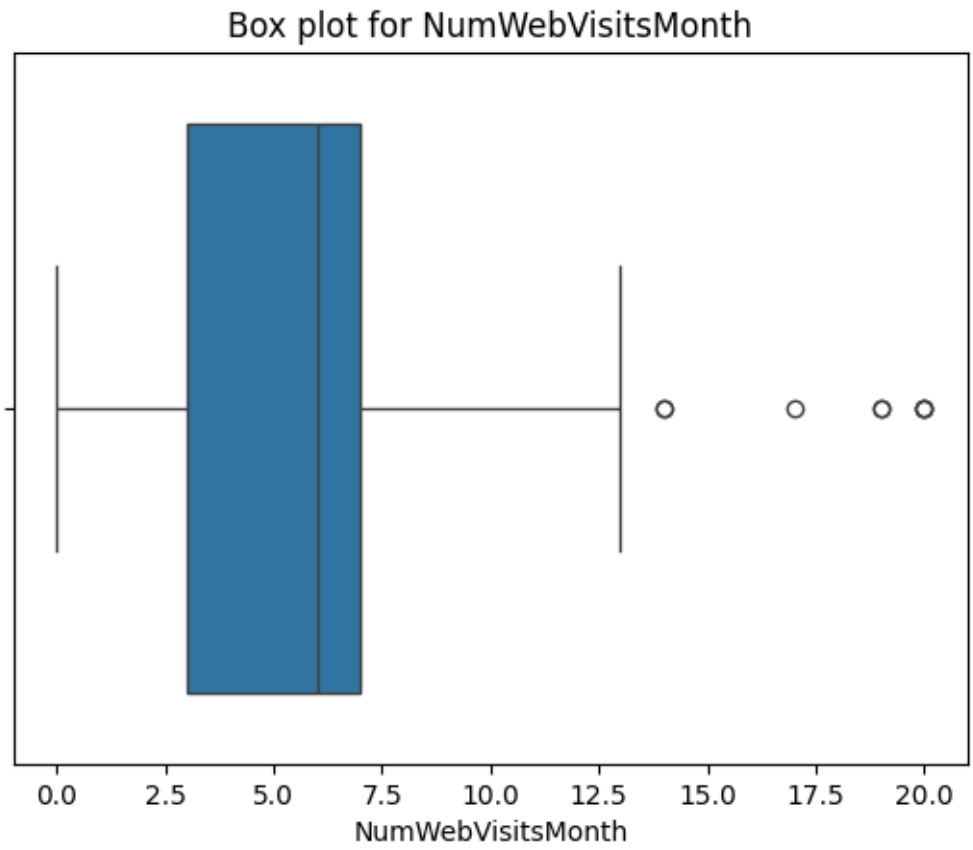

Box plot for MntGoldProds
0
50
100
150
200
250
300
350
MntGoldProds

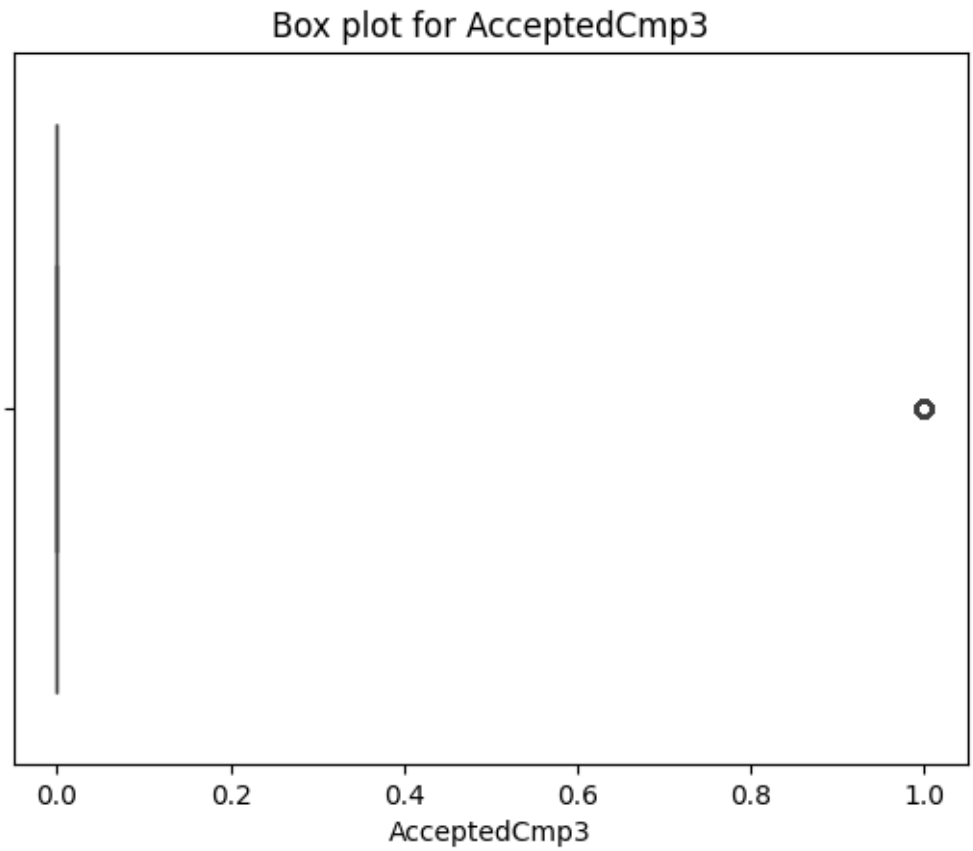

Box plot for NumDealsPurchases
0
2
4
6
8
10
12
14
NumDealsPurchases

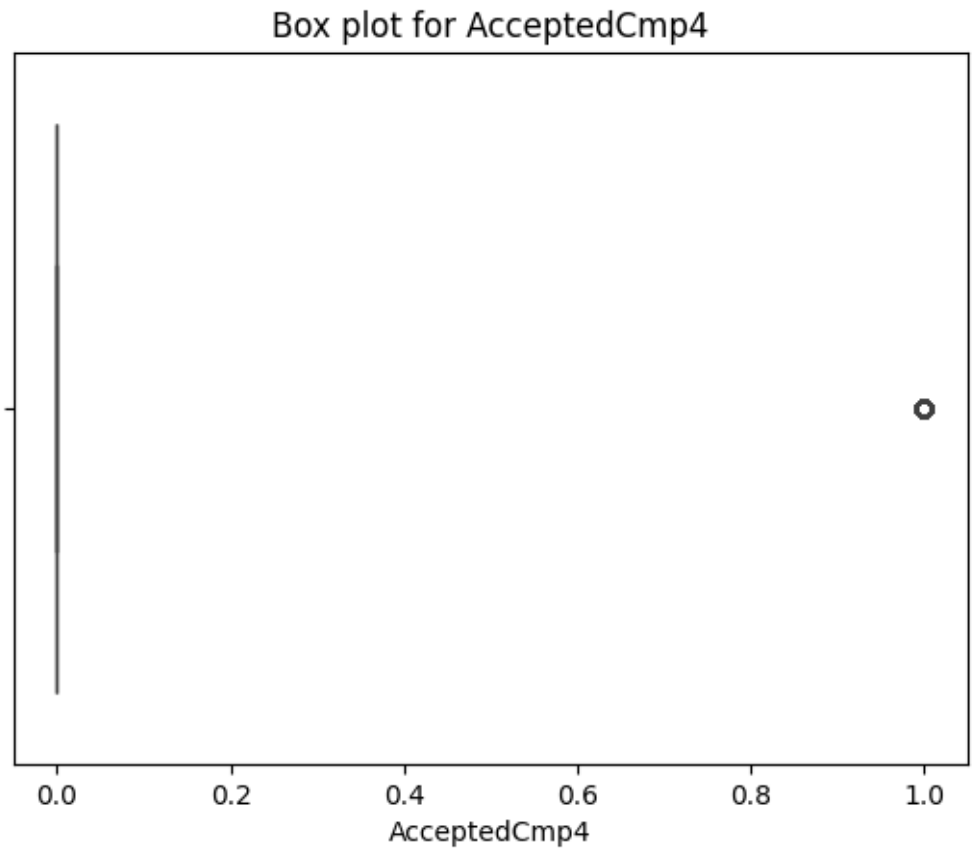

Box plot for NumWebPurchases
0
5
10
15
20
25
NumWebPurchases

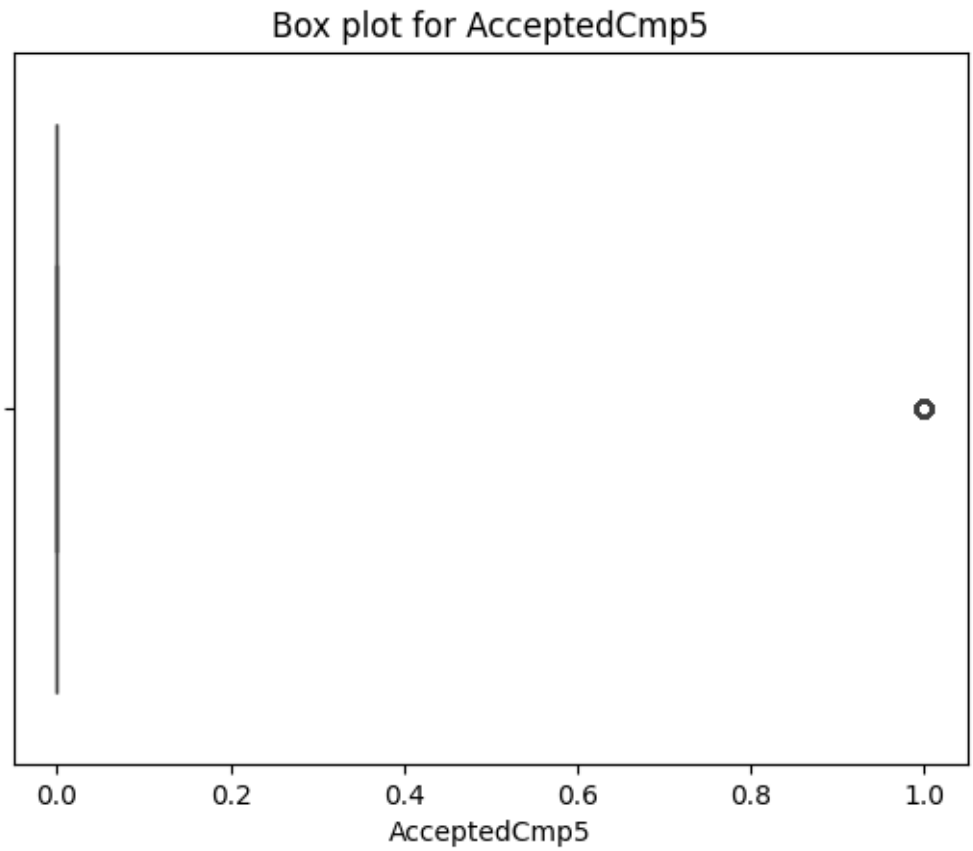

Box plot for NumCatalogPurchases
0
5
10
15
20
25
NumCatalogPurchases

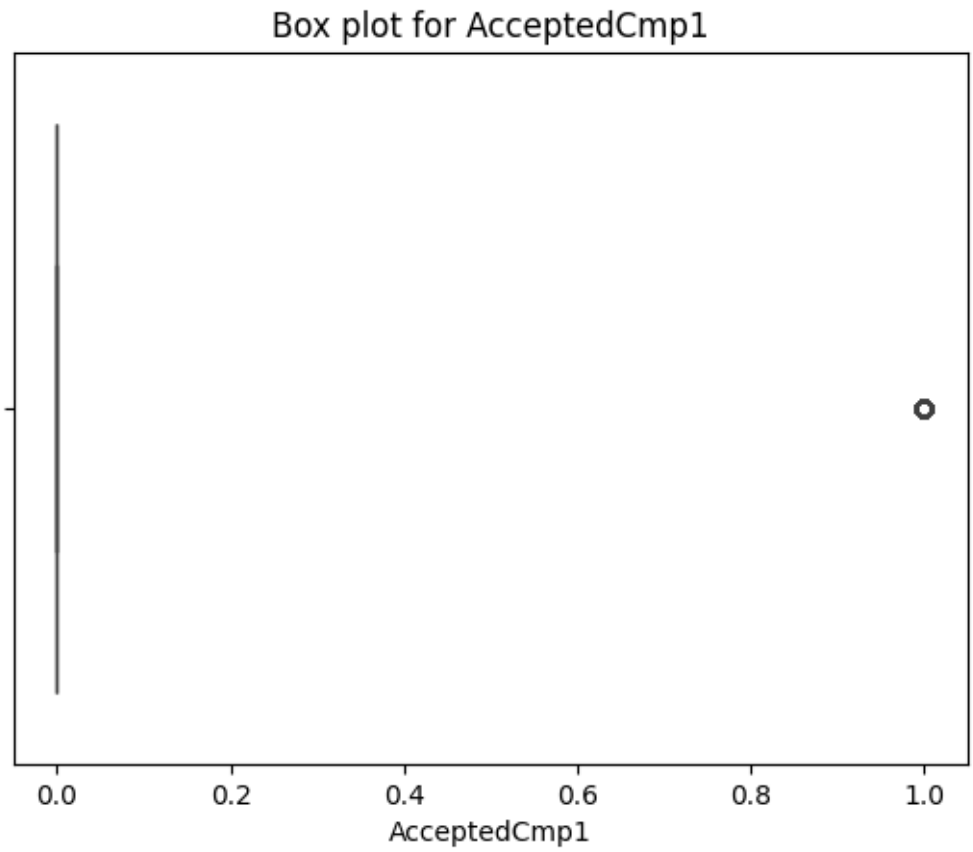

Box plot for NumStorePurchases
0
2
4
6
8
10
12
NumStorePurchases

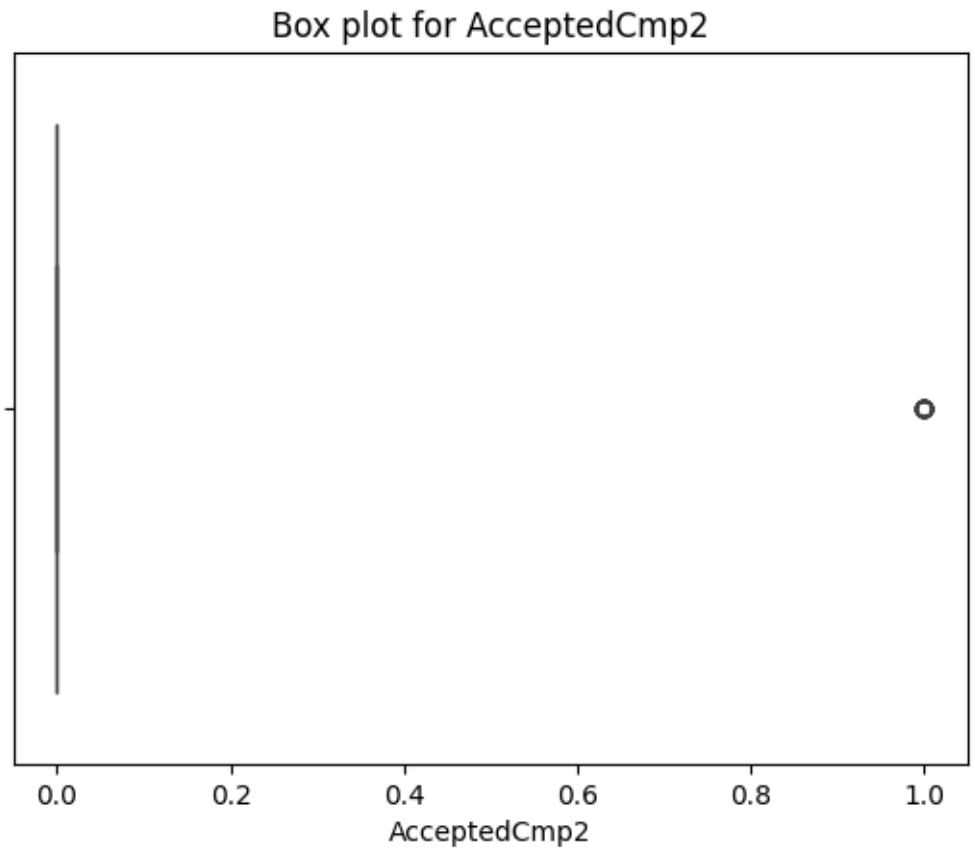

Box plot for NumWebVisitsMonth
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
NumWebVisitsMonth

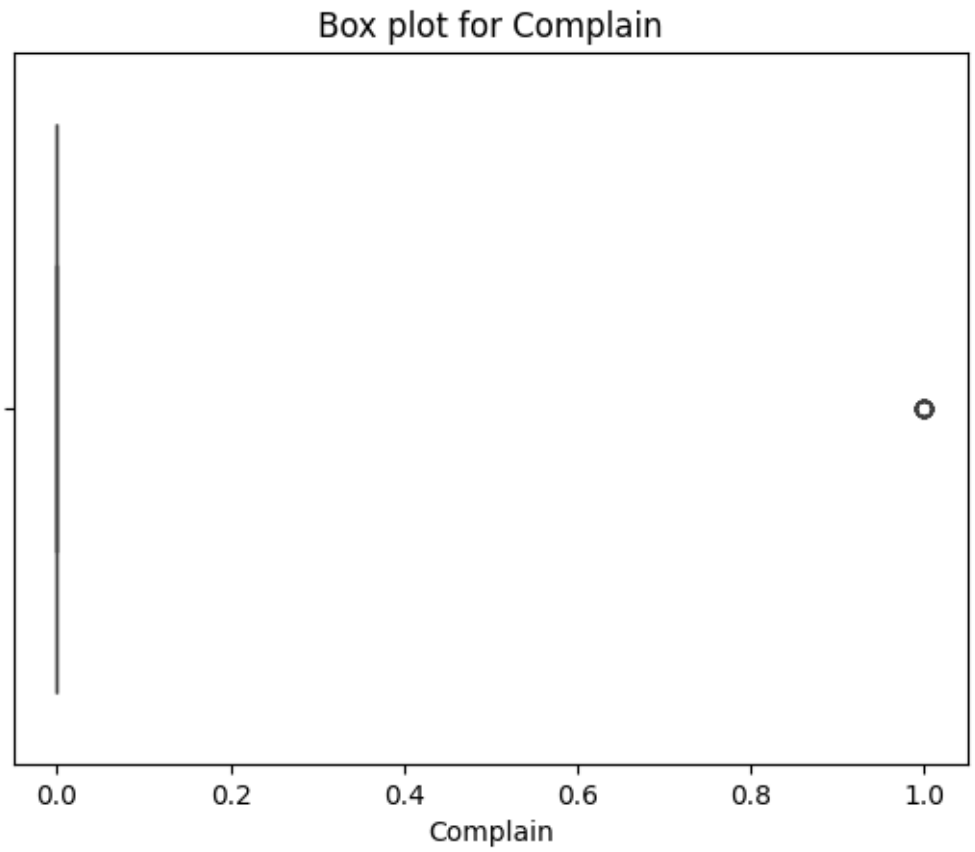

Box plot for AcceptedCmp3
0.0
0.2
0.4
0.6
0.8
1.0
AcceptedCmp3


Box plot for AcceptedCmp4
0.0
0.2
0.4
0.6
0.8
1.0
AcceptedCmp4


Box plot for AcceptedCmp5
0.0
0.2
0.4
0.6
0.8
1.0
AcceptedCmp5


Box plot for AcceptedCmp1
0.0
0.2
0.4
0.6
0.8
1.0
AcceptedCmp1


Box plot for AcceptedCmp2
0.0
0.2
0.4
0.6
0.8
1.0
AcceptedCmp2



3.0.5 Hypothesis Testing

- Null Hypothesis (H): Income is independent of education level.
- Alternative Hypothesis (H): Income is dependent on education level

[57]:
```
from scipy import stats
s,p = stats.f_oneway(*[camp[camp['Education'] == edu]['Income'].dropna() for␣
 ↪edu in camp['Education'].unique()])
print(f"stat value : {s} p_value : {p}")
if p < 0.05:
 print("Reject null")
else:
 print("Falied to reject")
```

```
stat value : 38.39294760713925 p_value : 4.188444786493969e-31
Reject null
```

- Null Hypothesis (H): Couples and singles spend the same amount on wine.
- Alternative Hypothesis (H): There is a difference in spending on wine between couples and singles

[60]:
```
camp['Marital_Status_Group'] = camp['Marital_Status'].replace({'Married':␣
 ↪'Couple', 'Together': 'Couple', 'Divorced': 'Alone', 'Single': 'Alone',␣
 ↪'Absurd': 'Alone', 'Widow': 'Alone'})
wine_couples = camp[camp['Marital_Status_Group'] == 'Couple']['MntWines']
wine_singles = camp[camp['Marital_Status_Group'] == 'Alone']['MntWines']
s,p = stats.ttest_ind(wine_couples, wine_singles, equal_var=False)
print(f"stat value : {s} p_value : {p}")
if p < 0.05:
 print("Reject null")
else:
 print("Falied to reject")
```

```
stat value : -0.26810064383953386 p_value : 0.7886557650817475
Falied to reject
```

- Null Hypothesis (H): Customers with lower income are not more likely to accept campaigns.
- Alternative Hypothesis (H): Customers with lower income are more likely to accept campaigns.

[62]:
```
median_income = camp['Income'].median()
camp['Income_Bracket'] = camp['Income'].apply(lambda x: 'Below Median' if x <␣
 ↪median_income else 'Above Median')
camp['Ever_Accepted_Campaign'] = (camp[['AcceptedCmp1', 'AcceptedCmp2',␣
 ↪'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum(axis=1) > 0).astype(int)
contingency_table = pd.crosstab(camp['Income_Bracket'],␣
 ↪camp['Ever_Accepted_Campaign'])
chi,p,_,_ = stats.chi2_contingency(contingency_table)
print(f'Chi-square value: {chi}, p-value: {p}')
alpha = 0.05
if p<alpha:
 print("Reject Ho")
else:
 print("Failed to reject Ho")
```

```
Chi-square value: 140.11555527497433, p-value: 2.5115657237830455e-32
Reject Ho
```

3.1 Insights:

- Education & Spending:
- Graduates and PhD holders are key spenders, with PhDs spending more on wine and graduates on gold.Except for basic-educated customers, others prefer spending on meat and fish products.
- Marital Status & Buying Behavior:
- Couples and singles show no significant difference in spending on wine.
- Most customers are married or living together, indicating a family-oriented customer base.
- Product Preferences:
- Majority buy one or two products, suggesting a skewed distribution toward fewer purchases.

- Customers purchasing three products tend to buy directly from stores
- Income & Campaign Acceptance:
- Income is dependent on education level.
- Lower-income customers are more likely to accept campaigns, indicating price sensitivity.
- Geographic & Offer Insights:
- Most purchases come from Spanish customers.
- Catalog purchases are less preferred, while direct offers seem effective initially but drop significantly by Campaign 5.
- Complaints are minimal, showing general satisfaction.

3.2 Recommendations:

- Targeted Campaigns:
- Focus campaigns on lower-income groups and highlight budget-friendly products to increase acceptance.
- Design promotions for graduates and PhD holders, leveraging their spending habits (wine and gold).
- Product Strategy:
- Diversify product offerings to encourage multi-product purchases.
- Highlight meat, fish, and wine in marketing campaigns, especially toward educated and family-oriented segments.
- Revisit Campaigns:
- Evaluate why acceptance drops by Campaign 5 and adjust the strategy to maintain engagement.
- Reduce emphasis on catalog sales and promote digital or store offers.
- Localization:
- Tailor marketing specifically for the Spanish demographic to capitalize on the existing customer base.

[ ]: