e-commerce-bussiness-case

August 28, 2024

[30]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from statsmodels.stats.weightstats import ztest
```

[31]:
```python
shopping_df = pd.read_csv('shopping.csv')
campaign_df = pd.read_csv('camp.csv')
```

[32]:
```python
print("Shopping Data:")
print(shopping_df.info())
print(shopping_df.describe())
print("\nCampaign Data:")
print(campaign_df.info())
print(campaign_df.describe())
```

```
Shopping Data:
<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
None
       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

Campaign Data:
<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
None
                 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]
```

[33]:

```
shopping_df.isna().sum()
```

[33]:

```
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
```

[34]:

```
campaign_df.isna().sum()
```

[34]:

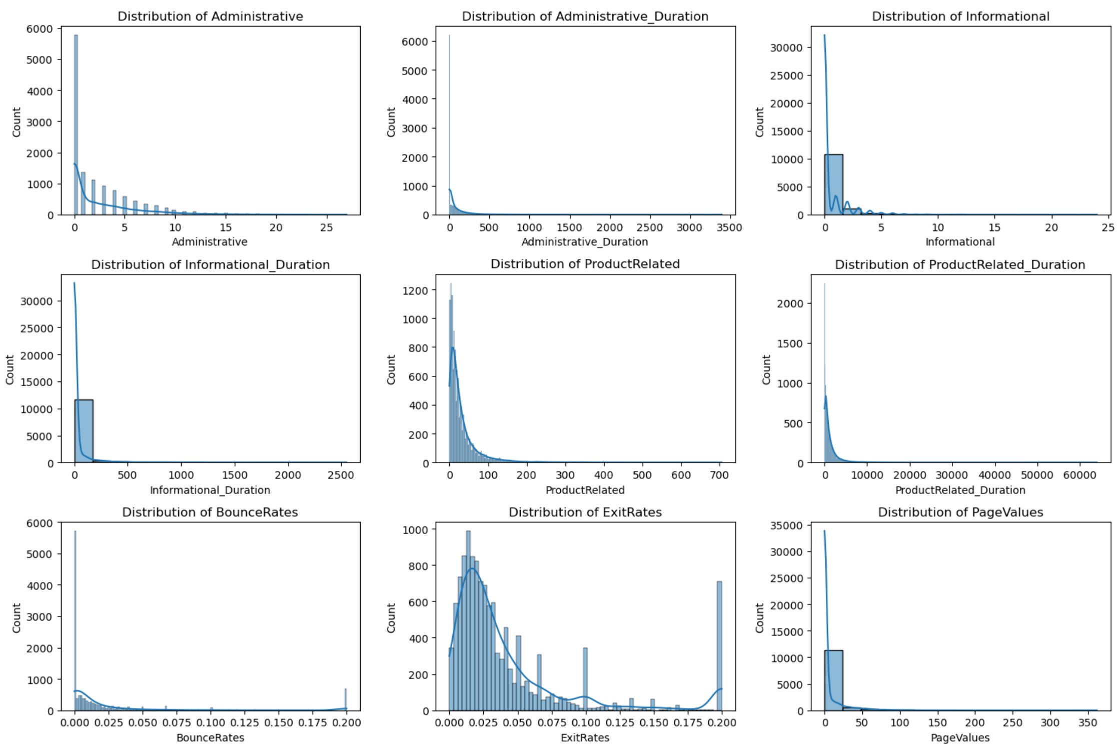
```
ID                     0
Year_Birth             0
Education              0
Marital_Status         0
Income                 0
Kidhome                0
Teenhome               0
Dt_Customer            0
Recency                0
MntWines               0
MntFruits              0
MntMeatProducts        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
```

[35]:

```
numerical_features = ['Administrative', 'Administrative_Duration',␣
↪'Informational', 'Informational_Duration',
```

```
                        'ProductRelated', 'ProductRelated_Duration',␣
↪'BounceRates', 'ExitRates', 'PageValues']

plt.figure(figsize=(15, 10))
for i, feature in enumerate(numerical_features):
    plt.subplot(3, 3, i+1)
    sns.histplot(shopping_df[feature], kde=True)
    plt.title(f'Distribution of {feature}')
plt.tight_layout()
plt.show()
print("Revenue Distribution:")
print(shopping_df['Revenue'].value_counts(normalize=True)*100)
```

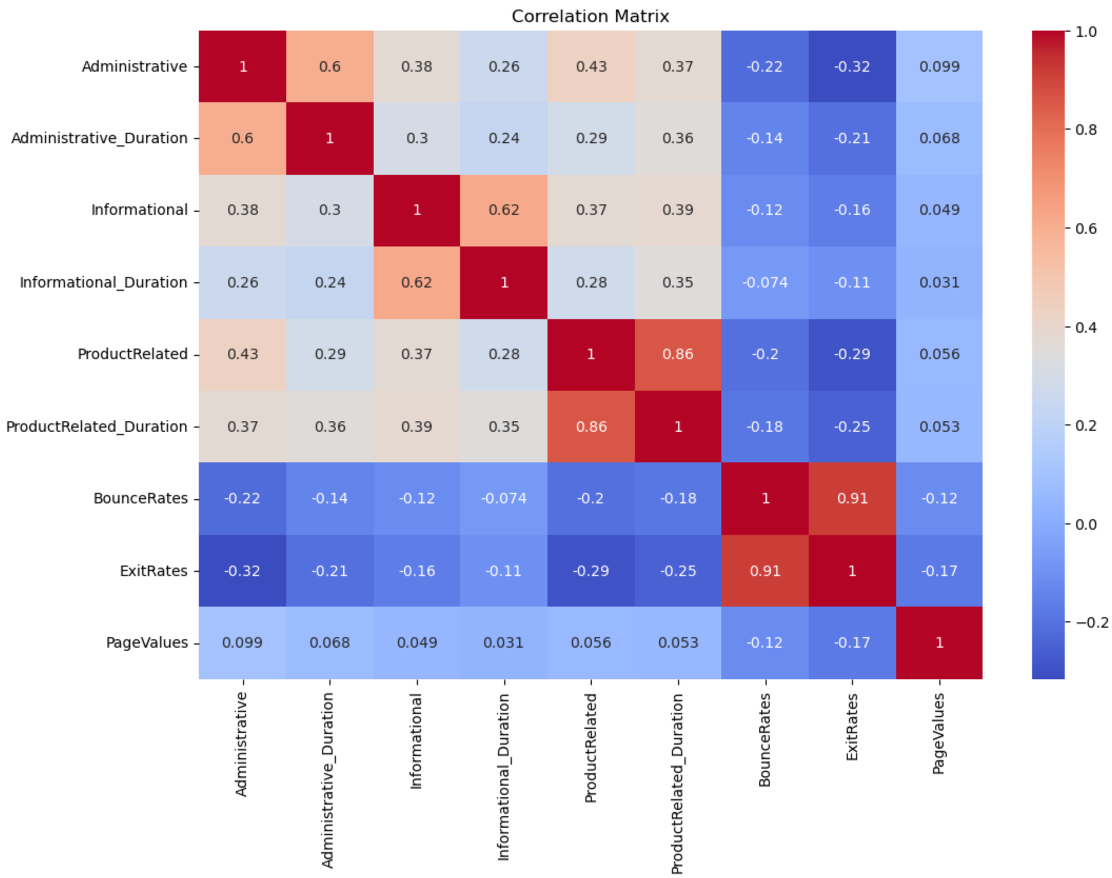



```
Revenue Distribution:
False    84.525547
True     15.474453
Name: Revenue, dtype: float64
```

[36]:
```
corr_matrix = shopping_df[numerical_features].corr()

plt.figure(figsize=(12, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
```

```
plt.show()
```

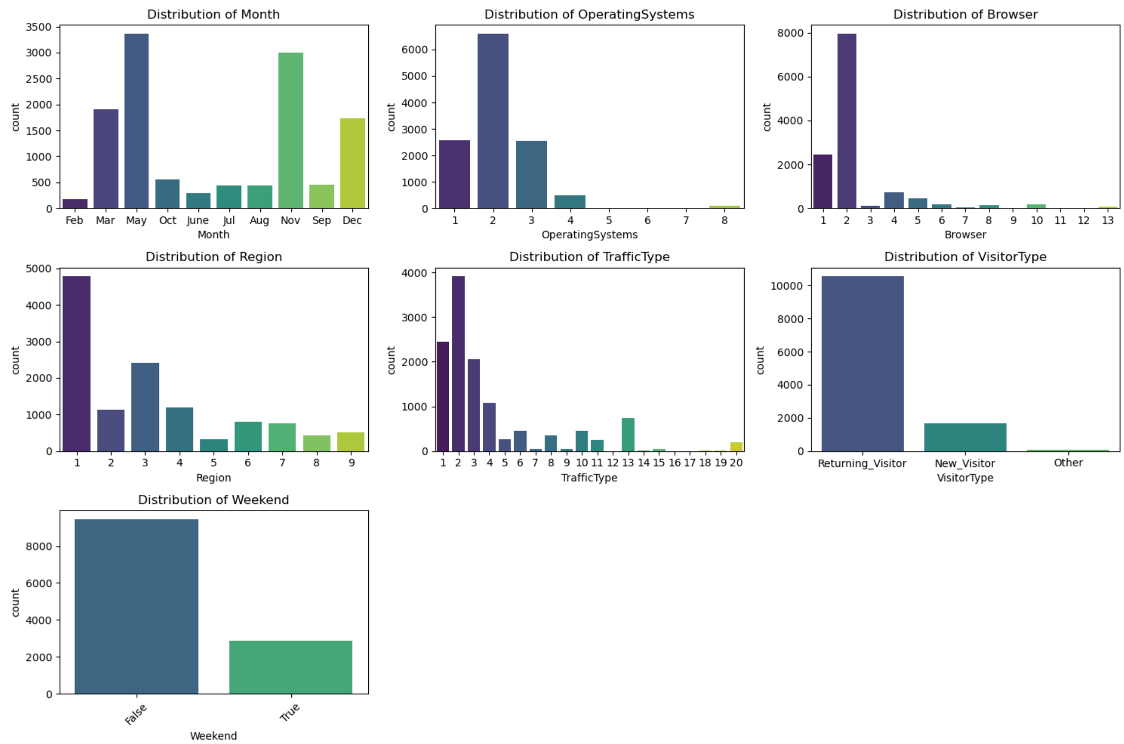


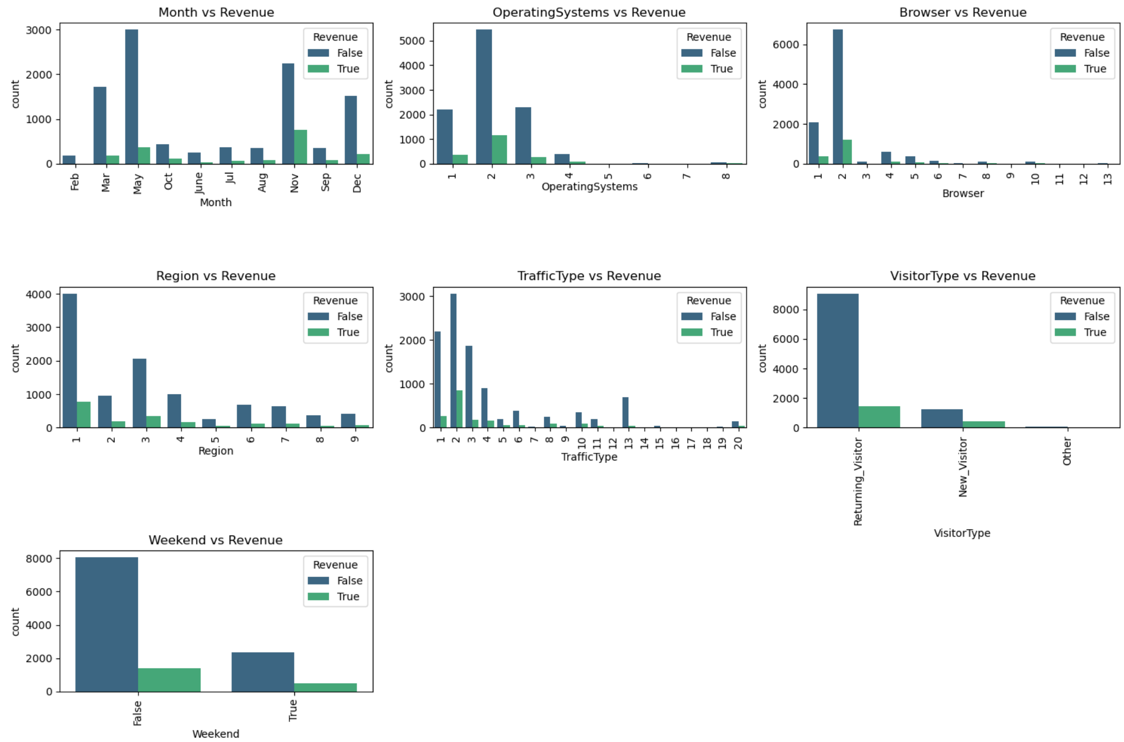
[37]:
```
categorical_features = ['Month', 'OperatingSystems', 'Browser', 'Region',␣
 ↪'TrafficType', 'VisitorType', 'Weekend']

plt.figure(figsize=(15, 10))
for i, feature in enumerate(categorical_features):
    plt.subplot(3, 3, i+1)
    sns.countplot(x=feature, data=shopping_df, palette='viridis')
    plt.title(f'Distribution of {feature}')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

plt.figure(figsize=(15, 10))
for i, feature in enumerate(categorical_features):
    plt.subplot(3, 3, i+1)
    sns.countplot(x=feature, hue='Revenue', data=shopping_df, palette='viridis')
    plt.title(f'{feature} vs Revenue')
```

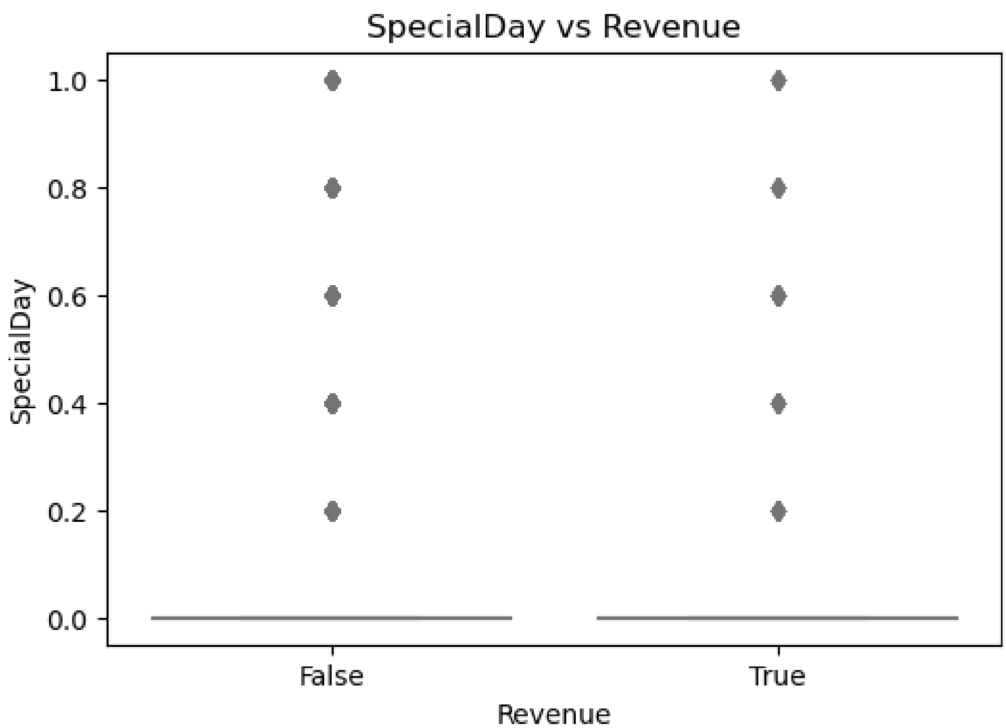
```
    plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```





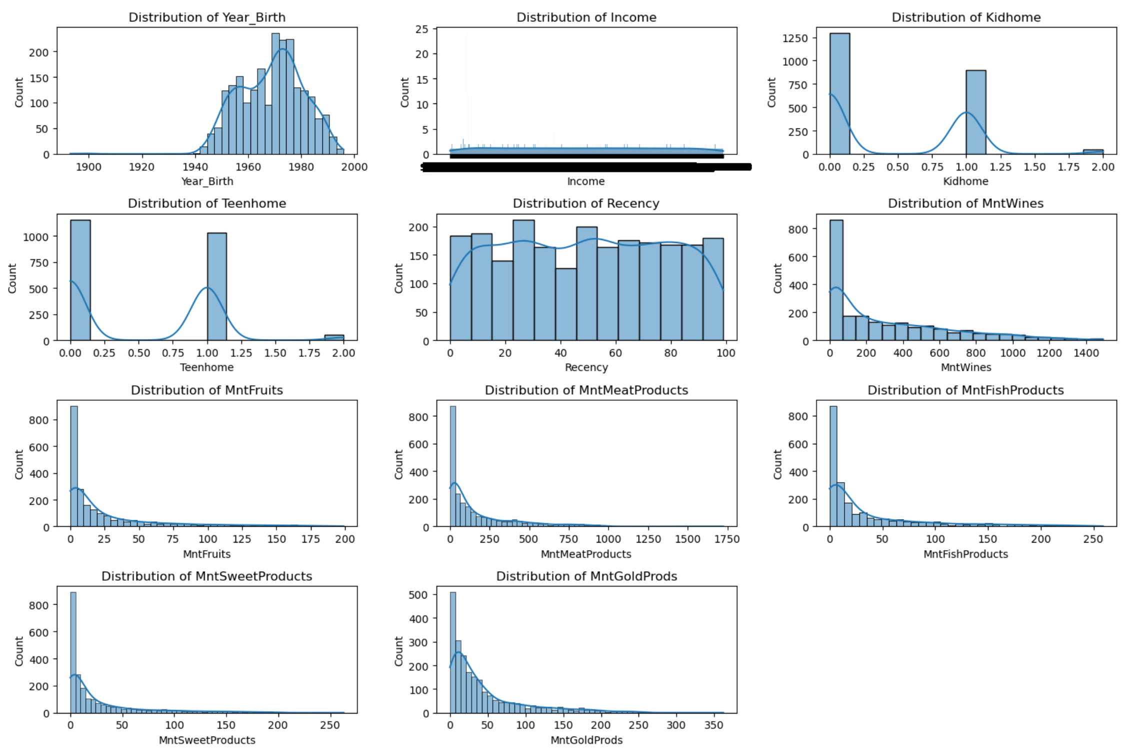

```
[38]: page_categories = ['Administrative', 'Informational', 'ProductRelated']
for category in page_categories:
    avg_time_spent = shopping_df[f'{category}_Duration'].mean()
    print(f'Average time spent on {category} pages: {avg_time_spent:.2f}␣
 ↪seconds')
plt.figure(figsize=(6, 4))
sns.boxplot(x='Revenue', y='SpecialDay', data=shopping_df, palette='pastel')
plt.title('SpecialDay vs Revenue')
plt.show()
```

```
Average time spent on Administrative pages: 80.82 seconds
Average time spent on Informational pages: 34.47 seconds
Average time spent on ProductRelated pages: 1194.75 seconds
```



[39]:
```
campaign_numerical_features = ['Year_Birth', 'Income', 'Kidhome', 'Teenhome',␣
↪'Recency',
                              'MntWines', 'MntFruits', 'MntMeatProducts',␣
↪'MntFishProducts',
                              'MntSweetProducts', 'MntGoldProds']

plt.figure(figsize=(15, 10))
for i, feature in enumerate(campaign_numerical_features):
    plt.subplot(4, 3, i+1)
    sns.histplot(campaign_df[feature], kde=True)
    plt.title(f'Distribution of {feature}')
plt.tight_layout()
plt.show()
campaign_df['Income'] = campaign_df['Income'].str.replace('[\$,]', '',␣
↪regex=True).astype(float)
campaign_df['Income_Bracket'] = pd.cut(campaign_df['Income'],
                                       bins=[0, 30000, 60000, 90000, 120000],
                                       labels=['Low', 'Medium', 'High', 'Very␣
↪High'])
campaign_df['AcceptedAnyCampaign'] = campaign_df[['AcceptedCmp1',␣
↪'AcceptedCmp2', 'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5']].sum(axis=1)␣
↪> 0
```



[40]:
```python
print("Unique values in Income_Bracket column:", campaign_df['Income_Bracket'].unique())
print("Unique values in Education column:", campaign_df['Education'].unique())

campaign_df.dropna(subset=['Income_Bracket', 'Education'], inplace=True)

contingency_table = pd.crosstab(campaign_df['Income_Bracket'], campaign_df['Education'])

print("Contingency Table:")
print(contingency_table)
if not contingency_table.empty:
    chi2, p, dof, ex = stats.chi2_contingency(contingency_table)

    print(f"Chi2 Stat: {chi2}, P-value: {p}")
    if p < 0.05:
        print("Reject the Null Hypothesis: There is a significant relationship between Income and Education.")
    else:
        print("Fail to Reject Null Hypothesis: No significant relationship between Income and Education.")
else:
```

```
    print("The contingency table is empty. Please check the data for missing or␣
↪invalid entries.")
```

```
Unique values in Income_Bracket column: ['High', 'Medium', 'Low', 'Very High',
NaN]
Categories (4, object): ['Low' < 'Medium' < 'High' < 'Very High']
Unique values in Education column: ['Graduation' 'PhD' '2n Cycle' 'Master'
'Basic']
Contingency Table:
Education       2n Cycle  Basic  Graduation  Master  PhD
Income_Bracket
Low                   52     52         193      41   32
Medium                83      2         480     191  249
High                  63      0         416     122  187
Very High              2      0          24      10    9
Chi2 Stat: 307.45150274539645, P-value: 1.2781150762721299e-58
Reject the Null Hypothesis: There is a significant relationship between Income
and Education.
```

[41]:

```
low_income_spend = campaign_df[campaign_df['Income_Bracket'] ==␣
↪'Low']['MntWines']
high_income_spend = campaign_df[campaign_df['Income_Bracket'] ==␣
↪'High']['MntWines']

t_stat, p_value = stats.ttest_ind(low_income_spend, high_income_spend,␣
↪nan_policy='omit')

print(f"T-Stat: {t_stat}, P-value: {p_value}")
if p_value < 0.05:
    print("Reject Null Hypothesis: Higher income people spend differently on␣
↪wine.")
else:
    print("Fail to Reject Null Hypothesis: No significant difference in␣
↪spending based on income.")
```

```
T-Stat: -35.23276579143706, P-value: 2.618323059303837e-185
Reject Null Hypothesis: Higher income people spend differently on wine.
```

[42]:

```
campaign_df['Living_Status'] = campaign_df['Marital_Status'].replace({
    'Married': 'In couple', 'Together': 'In couple',
    'Divorced': 'Alone', 'Single': 'Alone', 'Widow': 'Alone', 'Absurd':␣
↪'Alone', 'YOLO': 'Alone'})
in_couple = campaign_df[campaign_df['Living_Status'] == 'In couple']['MntWines']
alone = campaign_df[campaign_df['Living_Status'] == 'Alone']['MntWines']

t_stat, p_value = stats.ttest_ind(in_couple, alone, nan_policy='omit')
```

```
print(f"T-Stat: {t_stat}, P-value: {p_value}")
if p_value < 0.05:
    print("Reject Null Hypothesis: Couples and individuals living alone spend␣
 ↪differently on wine.")
else:
    print("Fail to Reject Null Hypothesis: No significant difference in wine␣
 ↪spending based on living status.")
```

```
T-Stat: -0.2767904381995147, P-value: 0.781966954866304
Fail to Reject Null Hypothesis: No significant difference in wine spending based
on living status.
```

[44]:
```
contingency_table = pd.crosstab(campaign_df['Income_Bracket'],␣
 ↪campaign_df['AcceptedAnyCampaign'])
chi2, p, dof, ex = stats.chi2_contingency(contingency_table)

print(f"Chi2 Stat: {chi2}, P-value: {p}")
if p < 0.05:
    print("Reject Null Hypothesis: There is a significant relationship between␣
 ↪Income and Campaign Acceptance.")
else:
    print("Fail to Reject Null Hypothesis: No significant relationship between␣
 ↪Income and Campaign Acceptance.")
```

```
Chi2 Stat: 246.07557217633348, P-value: 4.619521556015274e-53
Reject Null Hypothesis: There is a significant relationship between Income and
Campaign Acceptance.
```

1 Insights and recommendations

1. Exploratory Data Analysis (EDA) for the Shopping Dataset

a. User Behavior Analysis:

Page Categories and Engagement:

The Administrative, Informational, and ProductRelated pages represent different types of user interactions. Higher engagement time on ProductRelated pages could suggest users are more interested in specific products, which might correlate with a higher likelihood of purchase (Revenue = True). Bounce Rates and Exit Rates:

High BounceRates on specific pages could indicate that these pages are not engaging enough or are irrelevant to the users' needs. Pages with high ExitRates might be the last step before users abandon the site, suggesting that these could be optimized to improve conversion. b. Correlation Analysis:

Feature Relationships: Positive correlations between PageValues and Revenue indicate that pages with higher values (suggesting more valuable content or offers) are more likely to lead to a purchase. c. Insights on Special Days:

Impact of Special Days: The SpecialDay feature seems to affect purchase behavior. Users visiting close to a holiday or special occasion might have a higher conversion rate due to targeted campaigns or urgency related to the special day. It could be beneficial to increase marketing efforts or promotions around these periods. d. Segment Analysis:

User Segments by Traffic Type and Visitor Type: Analyzing segments by TrafficType and VisitorType could reveal different behavior patterns. For instance, Returning Visitors may show higher engagement and conversion compared to New Visitors, suggesting a loyal customer base that could be further nurtured. Recommendations for Shopping Dataset:

Optimize High Exit Rate Pages: Focus on improving the content or navigation flow of pages with high ExitRates to reduce drop-offs and increase conversions. Enhance Special Day Promotions: Since special days drive conversions, ramp up targeted marketing campaigns during these periods. Tailor Content for Returning Visitors: Utilize personalization strategies to engage Returning Visitors with customized offers based on their browsing history and preferences. 2. Exploratory Data Analysis for the Campaign Dataset

a. Customer Segmentation and Spending Patterns:

Income and Spending Analysis:

Higher-income brackets (High and Very High) generally spend more across all product categories (MntWines, MntFruits, etc.). This suggests targeting premium segments with luxury products or exclusive offers could be effective. Marital Status and Spending Behavior:

Customers who are In Couple tend to spend differently than those Living Alone, especially in categories like wine (MntWines). This insight could be leveraged in marketing campaigns that appeal to couples, such as holiday packages or events. b. Campaign Effectiveness:

Acceptance of Campaigns:

The analysis shows varied acceptance rates across different campaigns (AcceptedCmp1 to AcceptedCmp5). Campaigns with higher acceptance could be analyzed for best practices, while those with lower performance might need restructuring or different messaging. Recency of Last Purchase:

Recency indicates customer engagement level. Customers with recent purchases are more likely to respond positively to campaigns, suggesting that targeting recent buyers could improve campaign effectiveness. c. Hypothesis Testing Results:

Income and Campaign Response:

Lower income groups may show a higher acceptance rate for campaigns, indicating that discount-focused or value-based offers resonate better with these segments. Spending Differences by Demographic:

Hypothesis tests confirm that couples and higher-income groups tend to spend more, aligning with the need for differentiated marketing strategies targeting these segments. Recommendations for Campaign Dataset:

Personalize Campaigns by Income and Family Status: Tailor campaign messages and offers based on customer income brackets and marital status to enhance relevance and engagement. Focus on Recency for Campaign Targeting: Prioritize customers who have made recent purchases for new campaign offers, as they are more likely to convert. Refine Underperforming Campaigns: Analyze the structure and content of campaigns with low acceptance rates and adjust strategies accordingly, possibly by offering more attractive incentives or better targeting. Next Steps: Implement Data-Driven Marketing Strategies: Use the insights from EDA to drive marketing decisions, such as segment-specific offers and targeted campaigns. Continuous Monitoring and A/B Testing: Regularly track key metrics (e.g., conversion rates, campaign acceptance) and run A/B tests to optimize marketing strategies and site content. Customer Retention Strategies: Develop loyalty programs or personalized follow-ups for Returning Visitors and recent buyers to encourage repeat purchases.