Import Necessary libraries

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
```

Loading Dataset

[2]:
```
df_cs = pd.read_csv("Customers.csv")
df_pd = pd.read_csv("Products.csv")
df_ts = pd.read_csv("Transactions.csv")
```

[7]:
```
df_cs
```

[7]:

	CustomerID	CustomerName	Region	SignupDate
0	C0001	Lawrence Carroll	South America	2022-07-10
1	C0002	Elizabeth Lutz	Asia	2022-02-13
2	C0003	Michael Rivera	South America	2024-03-07
3	C0004	Kathleen Rodriguez	South America	2022-10-09
4	C0005	Laura Weber	Asia	2022-08-15
...	...	...	...	...
195	C0196	Laura Watts	Europe	2022-06-07
196	C0197	Christina Harvey	Europe	2023-03-21
197	C0198	Rebecca Ray	Europe	2022-02-27
198	C0199	Andrea Jenkins	Europe	2022-12-03
199	C0200	Kelly Cross	Asia	2023-06-11

200 rows × 4 columns

[8]:
```
df_pd
```

[8]:

	ProductID	ProductName	Category	Price
0	P001	ActiveWear Biography	Books	169.30
1	P002	ActiveWear Smartwatch	Electronics	346.30
2	P003	ComfortLiving Biography	Books	44.12
3	P004	BookWorld Rug	Home Decor	95.69
4	P005	TechPro T-Shirt	Clothing	429.31
...	...	...	...	...
95	P096	SoundWave Headphones	Electronics	307.47
96	P097	BookWorld Cookbook	Books	319.34
97	P098	SoundWave Laptop	Electronics	299.93
98	P099	SoundWave Mystery Book	Books	354.29
99	P100	HomeSense Sweater	Clothing	126.34

100 rows × 4 columns

[9]:
```
df_ts
```

[9]:

	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	TotalValue	Price
0	T00001	C0199	P067	2024-08-25 12:38:23	1	300.68	300.68
1	T00112	C0146	P067	2024-05-27 22:23:54	1	300.68	300.68
2	T00166	C0127	P067	2024-04-25 07:38:55	1	300.68	300.68
3	T00272	C0087	P067	2024-03-26 22:55:37	2	601.36	300.68
4	T00363	C0070	P067	2024-03-21 15:10:10	3	902.04	300.68
...	...	...	...	...	...	...	...
995	T00496	C0118	P037	2024-10-24 08:30:27	1	459.86	459.86
996	T00759	C0059	P037	2024-06-04 02:15:24	3	1379.58	459.86
997	T00922	C0018	P037	2024-04-05 13:05:32	4	1839.44	459.86
998	T00959	C0115	P037	2024-09-29 10:16:02	2	919.72	459.86
999	T00992	C0024	P037	2024-04-21 10:52:24	1	459.86	459.86

1000 rows × 7 columns

[13]:
```
print("Customers Info")
print(df_cs.info())
print("Products Info")
print(df_pd.info())
print("Transactions Info")
print(df_ts.info())
```

```
Customers Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   CustomerID    200 non-null    object
 1   CustomerName  200 non-null    object
 2   Region        200 non-null    object
 3   SignupDate    200 non-null    object
dtypes: object(4)
memory usage: 6.4+ KB
None
Products Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   ProductID    100 non-null    object
 1   ProductName  100 non-null    object
```

```
 2   Category     100 non-null    object
 3   Price        100 non-null    float64
dtypes: float64(1), object(3)
memory usage: 3.3+ KB
None
Transactions Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   TransactionID    1000 non-null   object
 1   CustomerID       1000 non-null   object
 2   ProductID        1000 non-null   object
 3   TransactionDate  1000 non-null   object
 4   Quantity         1000 non-null   int64
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
None
```

[14]:
```
print("Customers Describe")
print(df_cs.describe())
print("Products Describe")
print(df_pd.describe())
print("Transactions Describe")
print(df_ts.describe())
```

```
Customers Describe
       CustomerID      CustomerName         Region  SignupDate
count         200               200            200         200
unique        200               200              4         179
top         C0001  Lawrence Carroll  South America  2024-11-11
freq            1                 1             59           3
Products Describe
            Price
count  100.000000
mean   267.551700
std    143.219383
min     16.080000
25%    147.767500
50%    292.875000
75%    397.090000
max    497.760000
Transactions Describe
          Quantity   TotalValue       Price
count  1000.000000  1000.000000  1000.00000
mean      2.537000   689.995560   272.55407
std       1.117981   493.144478   140.73639
min       1.000000    16.080000    16.08000
25%       2.000000   295.295000   147.95000
50%       3.000000   588.880000   299.93000
75%       4.000000  1011.660000   404.40000
max       4.000000  1991.040000   497.76000
```

[30]:
```
# Convert date columns to datetime type
df_cs['SignupDate'] = pd.to_datetime(df_cs['SignupDate'])
df_ts['TransactionDate'] = pd.to_datetime(df_ts['TransactionDate'])
```

[16]:
```
# Data Cleaning - Check for Missing Values
print(df_cs.isnull().sum())
print(df_pd.isnull().sum())
print(df_ts.isnull().sum())
```

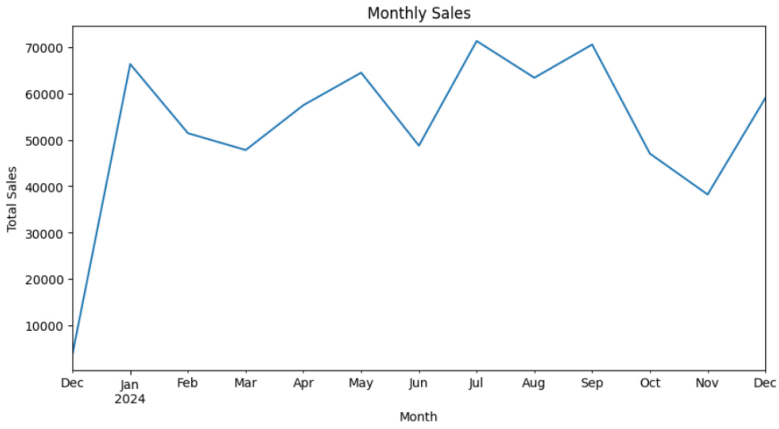
```
CustomerID      0
CustomerName    0
Region          0
SignupDate      0
dtype: int64
ProductID      0
ProductName    0
Category       0
Price          0
dtype: int64
TransactionID      0
CustomerID         0
ProductID          0
TransactionDate    0
Quantity           0
TotalValue         0
Price              0
dtype: int64
```

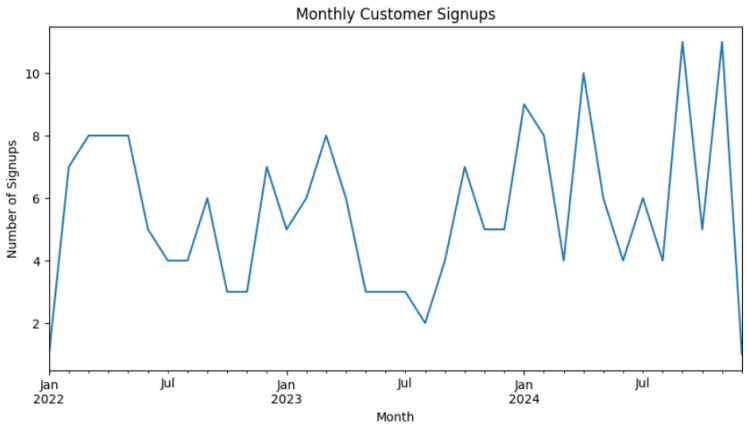
[29]:
```
# Customers
region_counts = df_cs['Region'].value_counts()
print(region_counts)

# Products
category_counts = df_pd['Category'].value_counts()
print(category_counts)
print(df_pd['Price'].describe())

# Transactions
print(df_ts['Quantity'].sum())
print(df_ts['TotalValue'].sum())
print(df_ts['TotalValue'].describe())
```

```
Region
South America    59
Europe           50
North America    46
Asia             45
Name: count, dtype: int64
Category
Books          26
Electronics    26
Clothing       25
Home Decor     23
Name: count, dtype: int64
count    100.000000
mean     267.551700
std      143.219383
min       16.080000
25%      147.767500
50%      292.875000
75%      397.090000
max      497.760000
Name: Price, dtype: float64
2537
689995.56
count    1000.000000
mean      689.995560
std       493.144478
min        16.080000
25%       295.295000
50%       588.880000
75%      1011.660000
max      1991.040000
Name: TotalValue, dtype: float64
```

Trends Over Time

[20]:
```
import matplotlib.pyplot as plt

df_ts['Month'] = df_ts['TransactionDate'].dt.to_period('M')
monthly_sales = df_ts.groupby('Month')['TotalValue'].sum()

plt.figure(figsize=(10, 5))
monthly_sales.plot(kind='line')
plt.title('Monthly Sales')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.show()

df_cs['SignupMonth'] = df_cs['SignupDate'].dt.to_period('M')
signups_by_month = df_cs.groupby('SignupMonth')['CustomerID'].count()

plt.figure(figsize=(10, 5))
signups_by_month.plot(kind='line')
plt.title('Monthly Customer Signups')
plt.xlabel('Month')
plt.ylabel('Number of Signups')
plt.show()
```

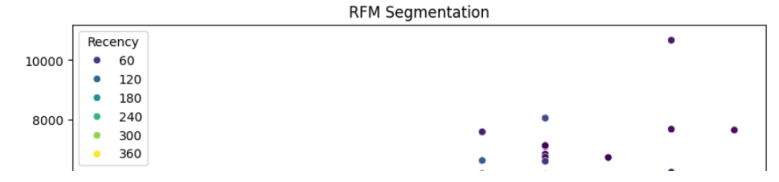


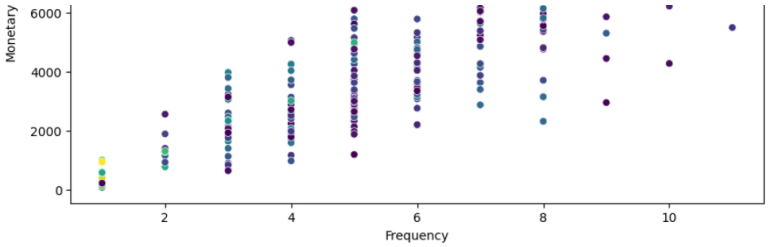


Customer Segmentation

[21]:
```
import datetime as dt

latest_date = df_ts['TransactionDate'].max() + dt.timedelta(days=1)
rfm = df_ts.groupby('CustomerID').agg({
    'TransactionDate': lambda x: (latest_date - x.max()).days,
    'TransactionID': 'count',
    'TotalValue': 'sum'
}).reset_index()

rfm.columns = ['CustomerID', 'Recency', 'Frequency', 'Monetary']

rfm['RFM_Score'] = (rfm['Recency'].rank(ascending=False) +
                    rfm['Frequency'].rank(ascending=True) +
                    rfm['Monetary'].rank(ascending=True))

print(rfm.head())

plt.figure(figsize=(10, 5))
sns.scatterplot(data=rfm, x='Frequency', y='Monetary', hue='Recency', palette='viridis')
plt.title('RFM Segmentation')
plt.xlabel('Frequency')
plt.ylabel('Monetary')
plt.show()
```

```
  CustomerID  Recency  Frequency  Monetary  RFM_Score
0      C0001       56          5   3354.52      317.5
1      C0002       26          4   1862.74      240.0
2      C0003      126          4   2725.38      177.5
3      C0004        5          8   5354.88      526.0
4      C0005       55          3   2034.24      191.0
```

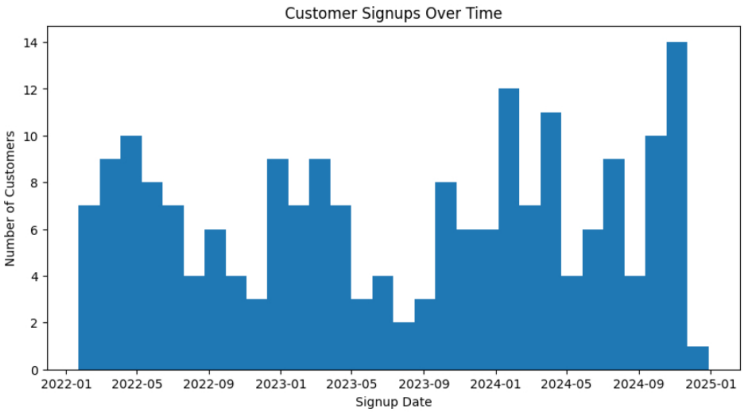




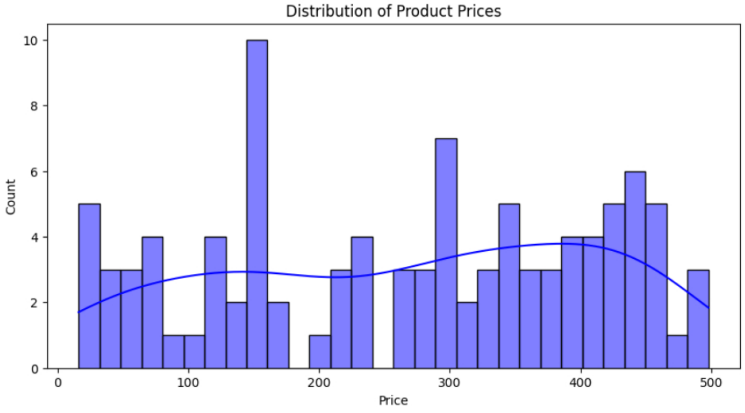
[22]:
```
plt.figure(figsize=(10, 5))
sns.countplot(data=df_cs, x='Region', palette='viridis')
plt.title('Number of Customers per Region')
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_22352\1004706191.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df_cs, x='Region', palette='viridis')
```

[25]:
```
plt.figure(figsize=(10, 5))
df_cs['SignupDate'].hist(bins=30, grid=False)
plt.title('Customer Signups Over Time')
plt.xlabel('Signup Date')
plt.ylabel('Number of Customers')
plt.show()
```

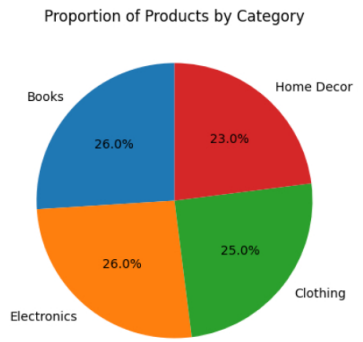


[24]:
```
plt.figure(figsize=(10, 5))
sns.histplot(df_pd['Price'], bins=30, kde=True, color='blue')
plt.title('Distribution of Product Prices')
plt.xlabel('Price')
plt.ylabel('Count')
plt.show()
```

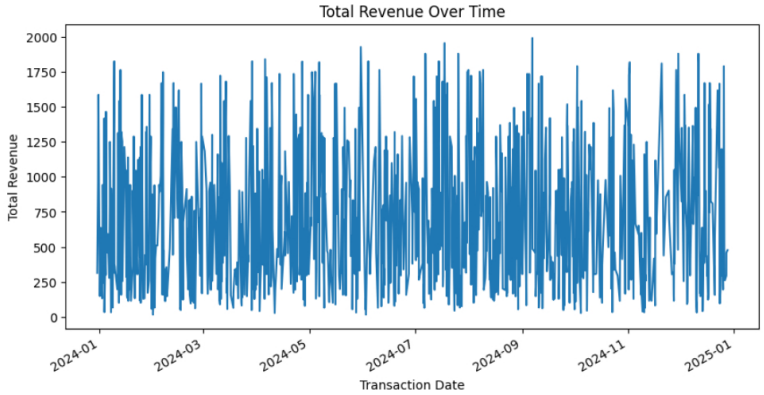


[26]:
```
plt.figure(figsize=(10, 5))
df_pd['Category'].value_counts().plot.pie(autopct='%1.1f%%', startangle=90)
plt.title('Proportion of Products by Category')
plt.ylabel('')
```

```
plt.show()
```



[31]:
```
plt.figure(figsize=(10, 5))
df_ts.groupby('TransactionDate')['TotalValue'].sum().plot()
plt.title('Total Revenue Over Time')
plt.xlabel('Transaction Date')
plt.ylabel('Total Revenue')
plt.show()
```



[33]:
```
plt.figure(figsize=(10, 5))
sns.scatterplot(data=df_ts, x='Quantity', y='TotalValue')
plt.title('Quantity vs. Total Value of Transactions')
plt.xlabel('Quantity')
plt.ylabel('Total Value')
plt.show()
```

Insight 1: The monthly sales trend shows a seasonal pattern with peaks during July to September months, indicating potential high-demand periods.

Insight 2: Customer segmentation analysis reveals that a significant portion of revenue comes from a small segment of high-value customers, suggesting a focus on retaining these customers could be beneficial.

Insight 3: The distribution of product categories indicates that a few categories dominate sales, suggesting a need to diversify the product range or focus marketing efforts on these high-performing categories.

Insight 4: The histogram of purchase amounts shows a right-skewed distribution, indicating that most transactions are of lower value, but there are occasional high-value purchases.

Insight 5: The pie chart of customer demographics shows a balanced distribution across products by different category like books, home decor, clothing and electronics, suggesting that marketing strategies should be tailored to appeal to a wide distribution range.

[ ]: