jupyter Online Sales(EDA) La

File Edit View Run Kernel Settings

Ma

Importing necessary L

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

[3]:
```
#Loading Online Sales Data
df = pd.read_csv(r'C:\Users\
df.head()
```

[3]:

	Transaction ID	Date
0	10001	2024-01-01
1	10002	2024-01-02
2	10003	2024-01-03
3	10004	2024-01-04
4	10005	2024-01-05

Basic Data Exploration

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.Da
RangeIndex: 240 entries, 0 t
Data columns (total 9 column
 #   Column           Non-N
---  ------           -----
 0   Transaction ID   240 n
 1   Date             240 n
 2   Product Category 240 n
 3   Product Name     240 n
 4   Units Sold       240 n
 5   Unit Price       240 n
 6   Total Revenue    240 n
 7   Region           240 n
```

ıst Checkpoint: 12 days ago

Help

ırkdown ⌄

.ibraries

```
plt
```

```
.pc\Desktop\60DaysDS\Online Sales Data.csv')
```

Product Category	Product Name	Units Sold	Unit Price	Total Revenue	Region	F
Electronics	iPhone 14 Pro	2	999.99	1999.98	North America	
Home Appliances	Dyson V11 Vacuum	1	499.99	499.99	Europe	
Clothing	Levi's 501 Jeans	3	69.99	209.97	Asia	
Books	The Da Vinci Code	4	15.99	63.96	North America	
Beauty Products	Neutrogena Skincare Set	1	89.99	89.99	Europe	

ı

```
ıtaFrame'>
:o 239
ıs):
Jull Count  Dtype
----------  -----
ıon-null    int64
ıon-null    object
ıon-null    object
ıon-null    object
ıon-null    int64
ıon-null    float64
ıon-null    float64
ıon-null    object
```

Trusted

JupyterLab Python 3 (ipykernel)

Payment Method
Credit Card
PayPal
Debit Card
Credit Card
PayPal

```
 7   Region           240 n
 8   Payment Method   240 n
dtypes: float64(2), int64(2)
memory usage: 17.0+ KB
```

[5]:
```python
df.describe()
```

[5]:

	Transaction ID	Units Sc
count	240.00000	240.0000
mean	10120.50000	2.1583
std	69.42622	1.3224
min	10001.00000	1.0000
25%	10060.75000	1.0000
50%	10120.50000	2.0000
75%	10180.25000	3.0000
max	10240.00000	10.0000

Data Cleaning

[6]:
```python
df.isnull().sum() # we have
```

[6]:
```
Transaction ID      0
Date                0
Product Category    0
Product Name        0
Units Sold          0
Unit Price          0
Total Revenue       0
Region              0
Payment Method      0
dtype: int64
```

[7]:
```python
df.duplicated().sum()
```

[7]: 0

datatypes

[8]:
```python
df['Date']=pd.to_datetime(df
df['Transaction ID']=df['Tra
df.dtypes
```

```
ion-null    object
ion-null    object
, object(5)
```

old	Unit Price	Total Revenue
00	240.000000	240.000000
33	236.395583	335.699375
54	429.446695	485.804469
00	6.500000	6.500000
00	29.500000	62.965000
00	89.990000	179.970000
00	249.990000	399.225000
00	3899.990000	3899.990000

```
missing values in this dataset
```

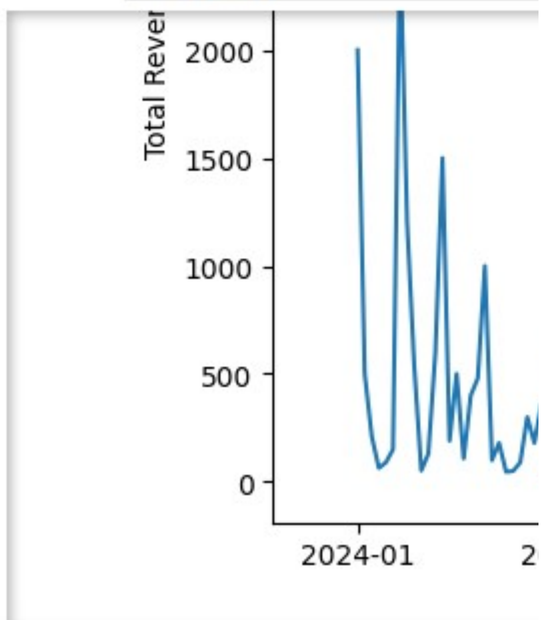
```
['Date'])
nsaction ID'].astype(str)
```

```
[8]: Transaction ID
     Date                  datetime
     Product Category
     Product Name
     Units Sold
     Unit Price                   f
     Total Revenue                f
     Region
     Payment Method
     dtype: object
```

Data analysis

Sales over time - To see the tre

```
[9]: sales_over_time = df.groupby
     #sales_over_time

     plt.figure(figsize=(12, 6))
     plt.plot(sales_over_time['Da
     plt.title('Sales Over Time')
     plt.xlabel('Date')
     plt.ylabel('Total Revenue')
     plt.show()
```



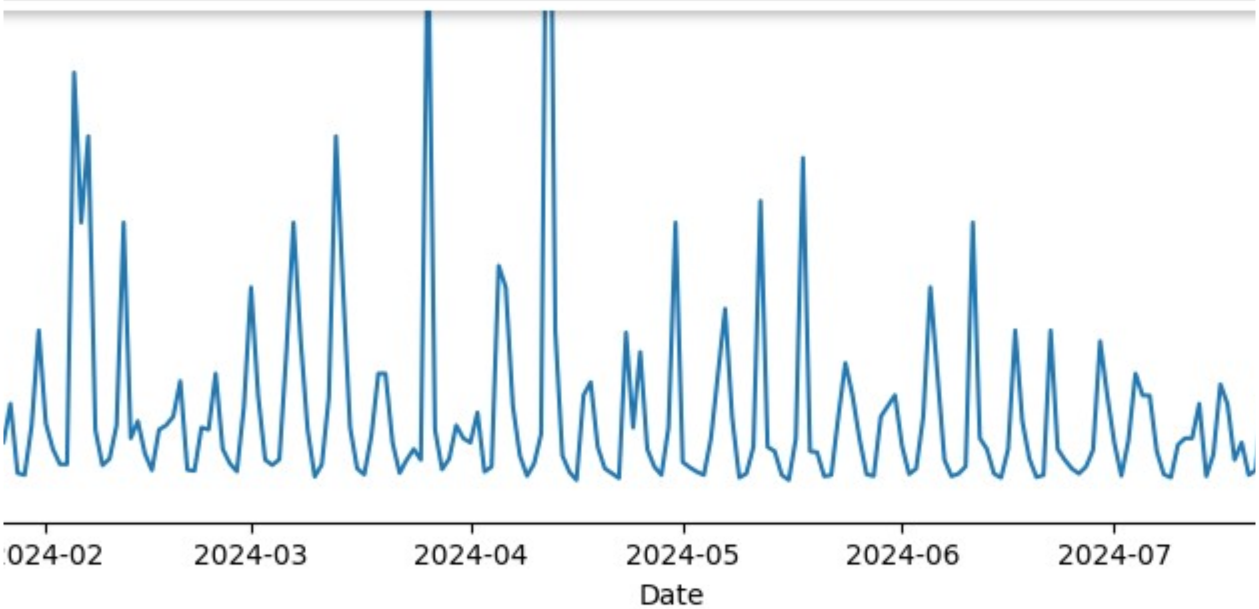

```
[ ]:
```

Lets Analyze, see patt

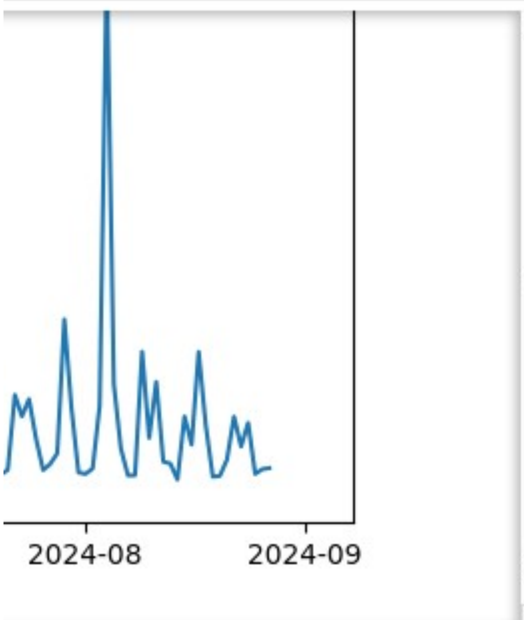
```
object
:64[ns]
object
object
 int64
float64
float64
object
object
```

nd

```
('Date')['Total Revenue'].sum().reset_index()


te'], sales_over_time['Total Revenue'])
```



erns and forecast Sales - Decomposition

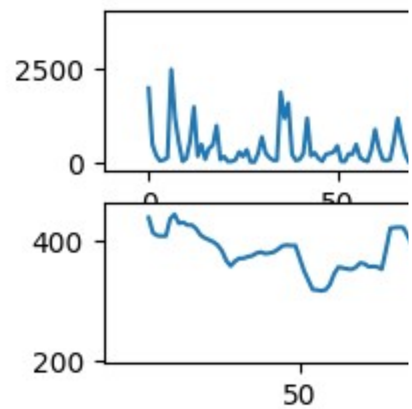


2024-08
2024-09

Time series Decomposition

[10]:
```
from statsmodels.tsa.seasona

result = seasonal_decompose(
plt.subplot(411)
plt.plot(result.observed, la
plt.legend(loc='best')

plt.subplot(412)
plt.plot(result.trend, label
plt.legend(loc='best')
```

[10]:
```
<matplotlib.legend.Legend at
```



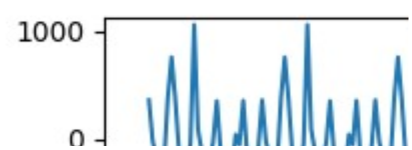
Observed Sales Data:This plot re
This indicates periods of high de

Trend: This plot shows the trend
end, the trend slightly recovers, s

Seasonality

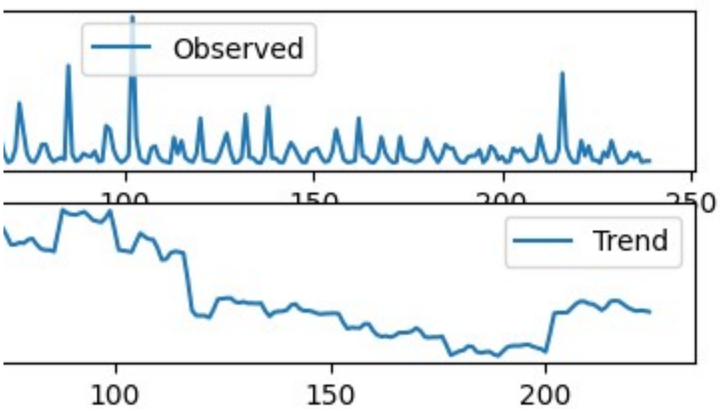
[11]:
```
plt.subplot(413)
plt.plot(result.seasonal, la
plt.legend(loc='best')
```

[11]:
```
<matplotlib.legend.Legend at
```



```
l import seasonal_decompose

'sales_over_time['Total Revenue'],model='additive', period=30)# additive model used( T ,

bel='Observed')


='Trend')
```

```
: 0x2233decabd0>
```

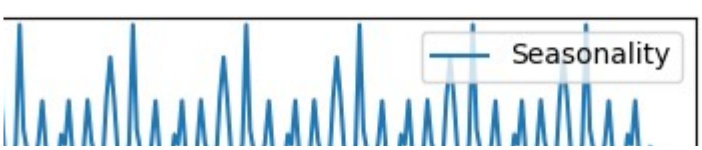


presents the raw sales data over time.We observe high sales on certain days. The data has significant f
mand or promotional events.

in the sales data after removing the seasonal and residual components. The trend indicates a gradual
showing a small upward movement. Which can be the indication of future growth

```
bel='Seasonality')
```

```
: 0x2233bbbe990>
```



```
s, E)
```

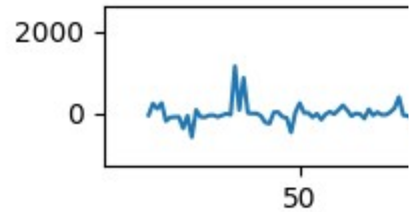
fluctuations , showing variability in sales

decrease in sales over time Towards the

Noise/Error

[12]:
```
plt.subplot(414)
plt.plot(result.resid, label
plt.legend(loc='best')
```

[12]: `<matplotlib.legend.Legend at`



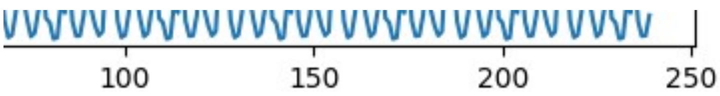
Sesonality The seasonality graph
correspond to weekly or monthly
by periodic factors.

Residual/Noise The residuals gr
presence of other factors beyond
factors or random events

[ ]:

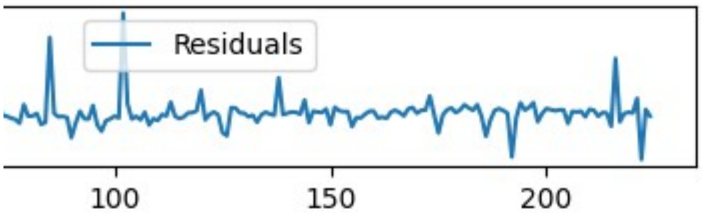
Sales by Region

[13]:
```
sales_by_region = df.groupby

plt.figure(figsize=(10, 6))
sns.barplot(x='Region', y='T
plt.title('Sales by Region')
plt.xlabel('Region')
plt.ylabel('Total Revenue')
plt.show()
```




```
.='Residuals')
```

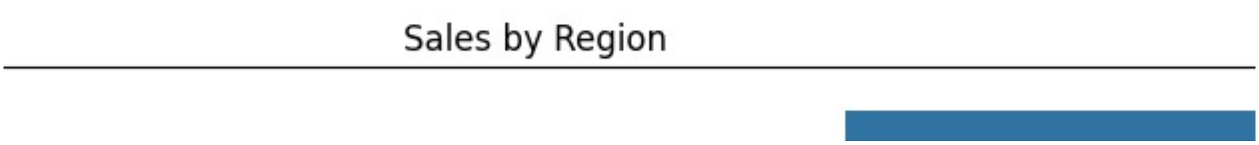
: 0x2233bc00260>



h displays a clear repeating pattern with peaks approximately every 20-25 time units, indicating a cons
y cycles, potentially driven by factors like weekends, or promotional events. seasonal pattern suggests

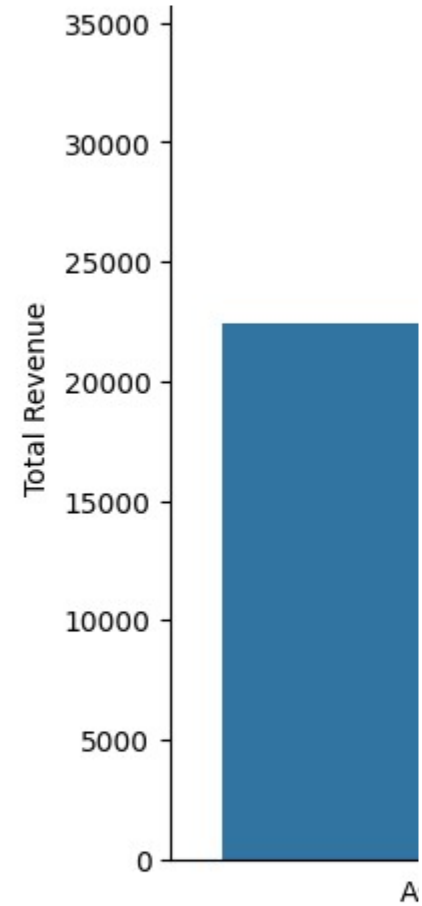
raph represents the noise component of the time series after removing the trend&season. The fluctuat
d seasonality The residuals exhibit some periods of higher volatility, around the middle and towards th

```
('Region')['Total Revenue'].sum().reset_index()


otal Revenue', data=sales_by_region)
```



sistent periodicity in sales peaks could
that online sales are heavily influenced

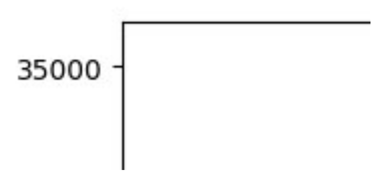
tions in the residuals indicate the
e end. This could be due to external

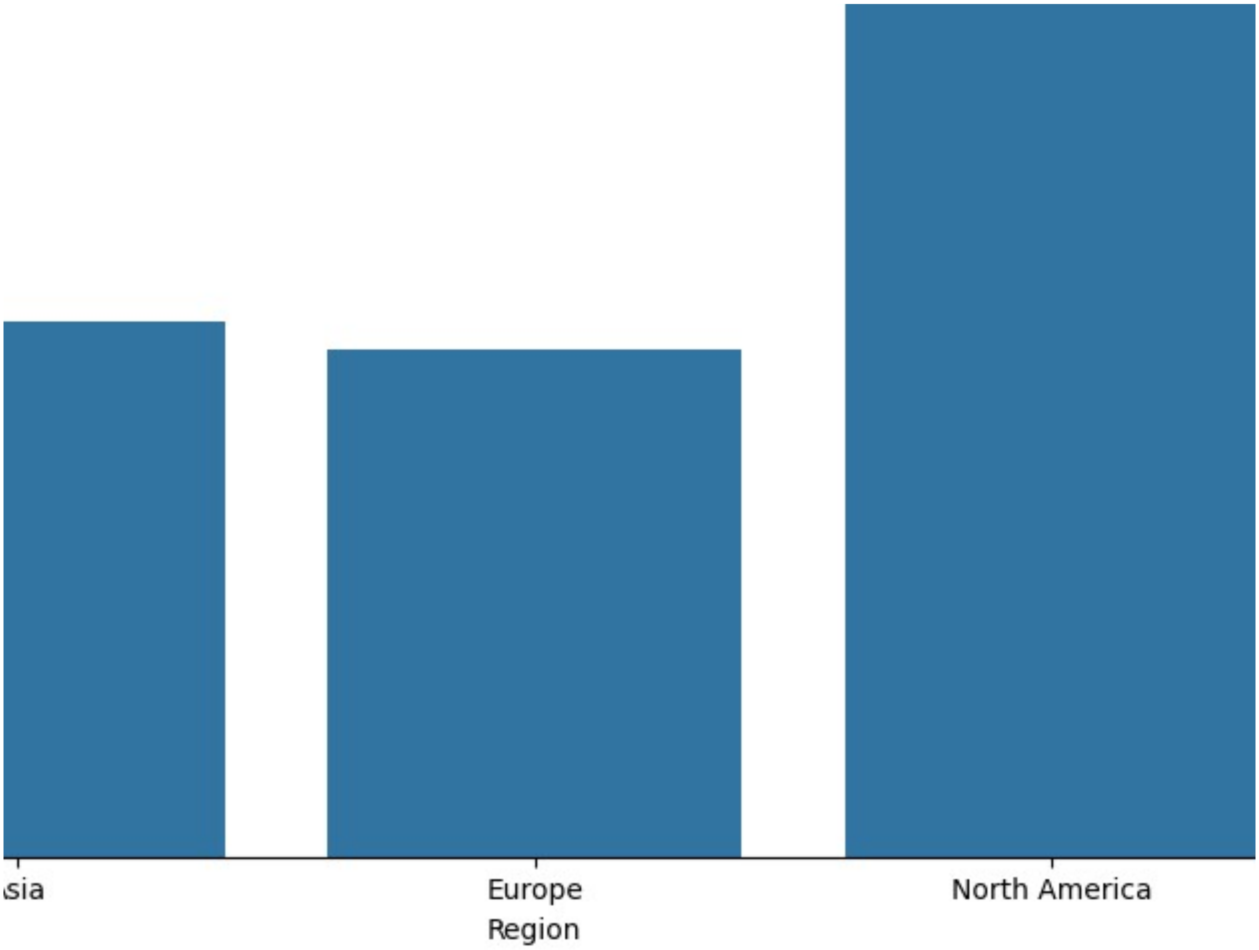


Sales by category

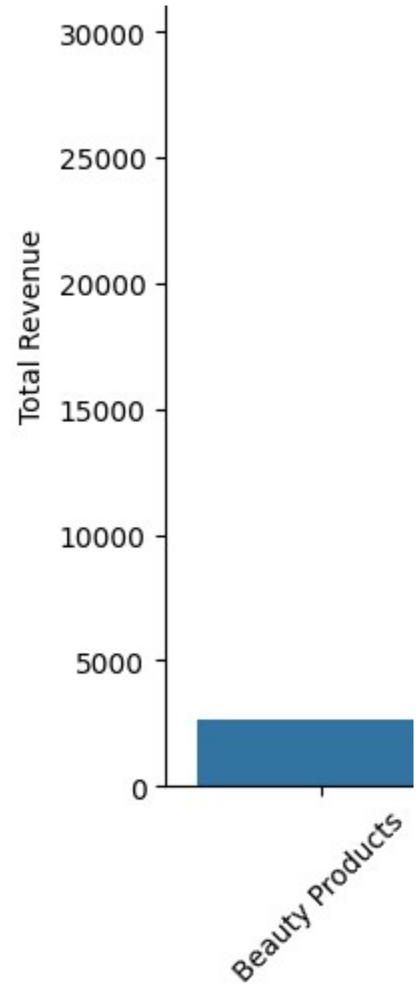
[14]:

```
sales_by_category = df.group

plt.figure(figsize=(12, 6))
sns.barplot(x='Product Categ
plt.title('Sales by Product 
plt.xlabel('Product Category
plt.ylabel('Total Revenue')
plt.xticks(rotation=45)
plt.show()
```



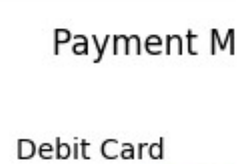


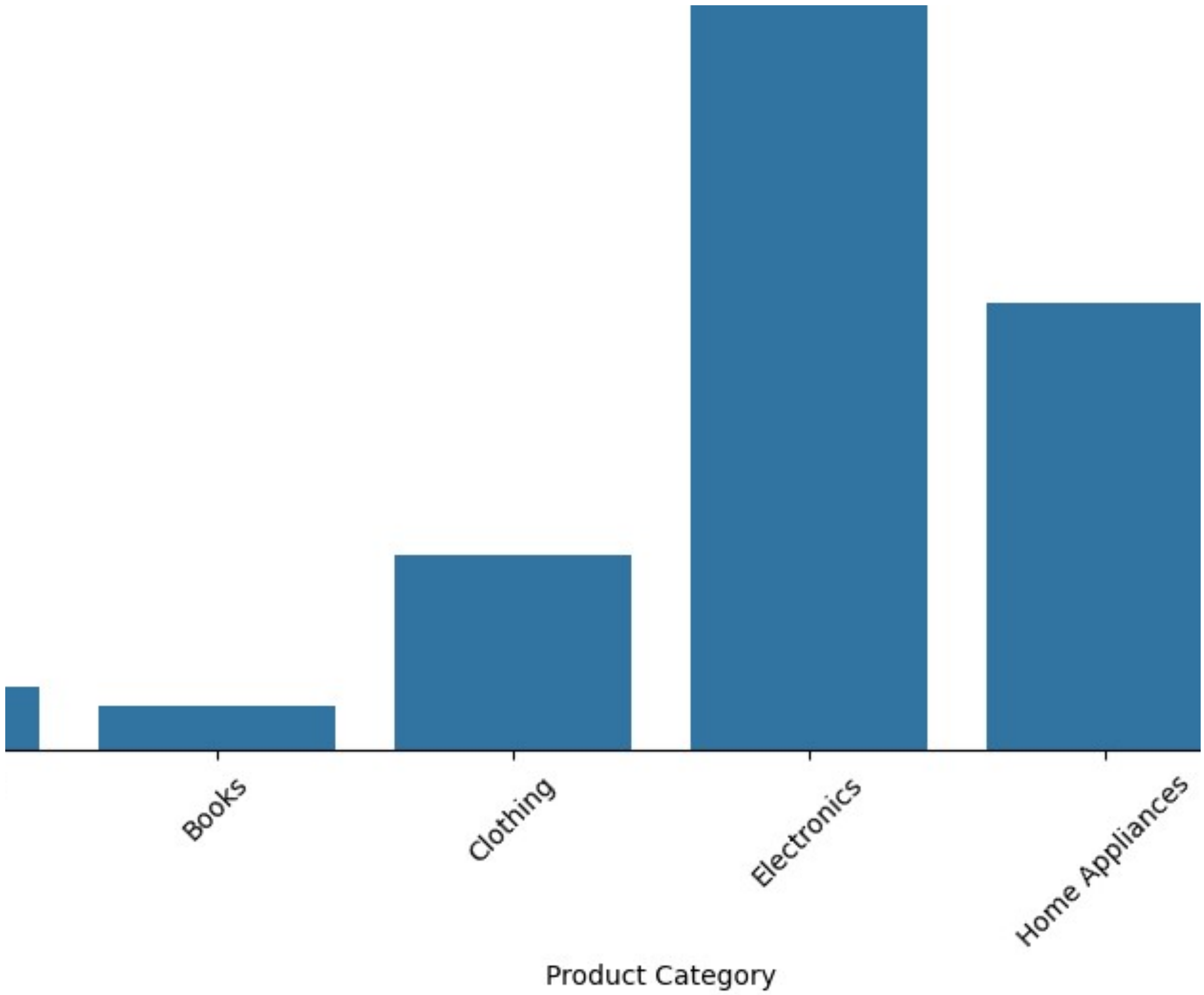

```
by('Product Category')['Total Revenue'].sum().reset_index()


gory', y='Total Revenue', data=sales_by_category)
 Category')
')
```



payment methods

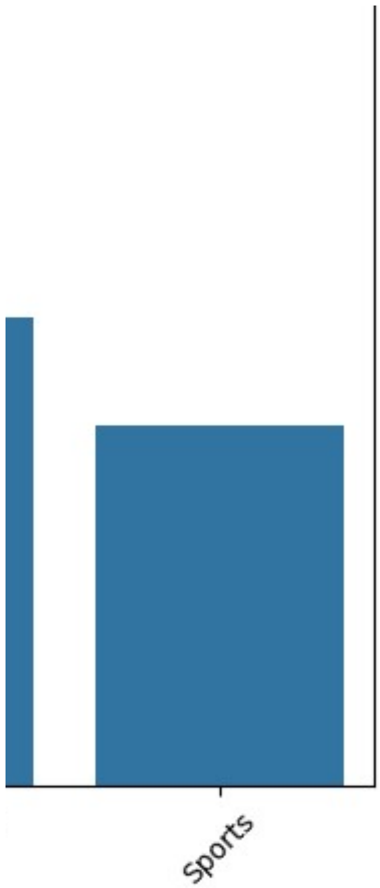
[15]:
```
#df.head()
payment_methods = df['Paymen


plt.figure(figsize=(8, 6))
payment_methods.plot.pie(aut
plt.title('Payment Methods D
plt.ylabel('')
plt.show()
```



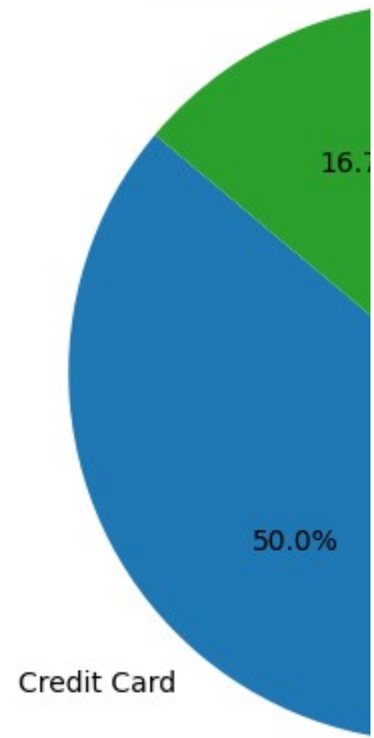



```
it Method'].value_counts()


topct='%1.1f%%', startangle=140)# the second % is used for showing % sign on the chart
Distribution')
```

lethods Distribution



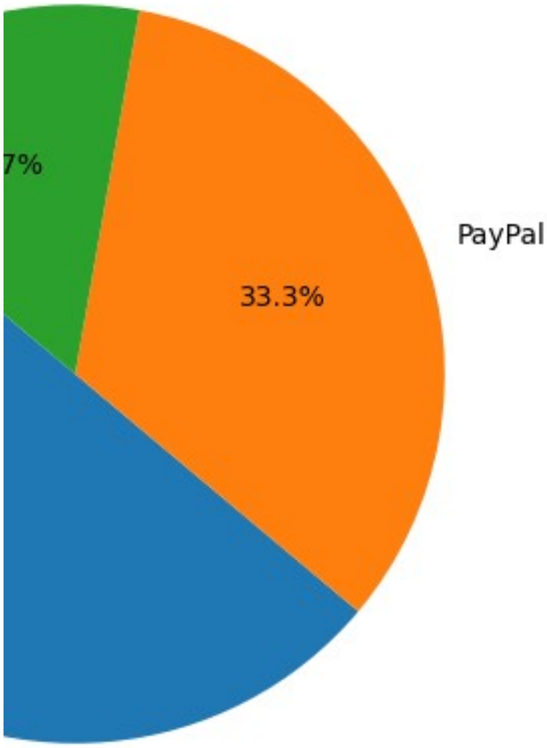
Sports



Heatmap of sales by r

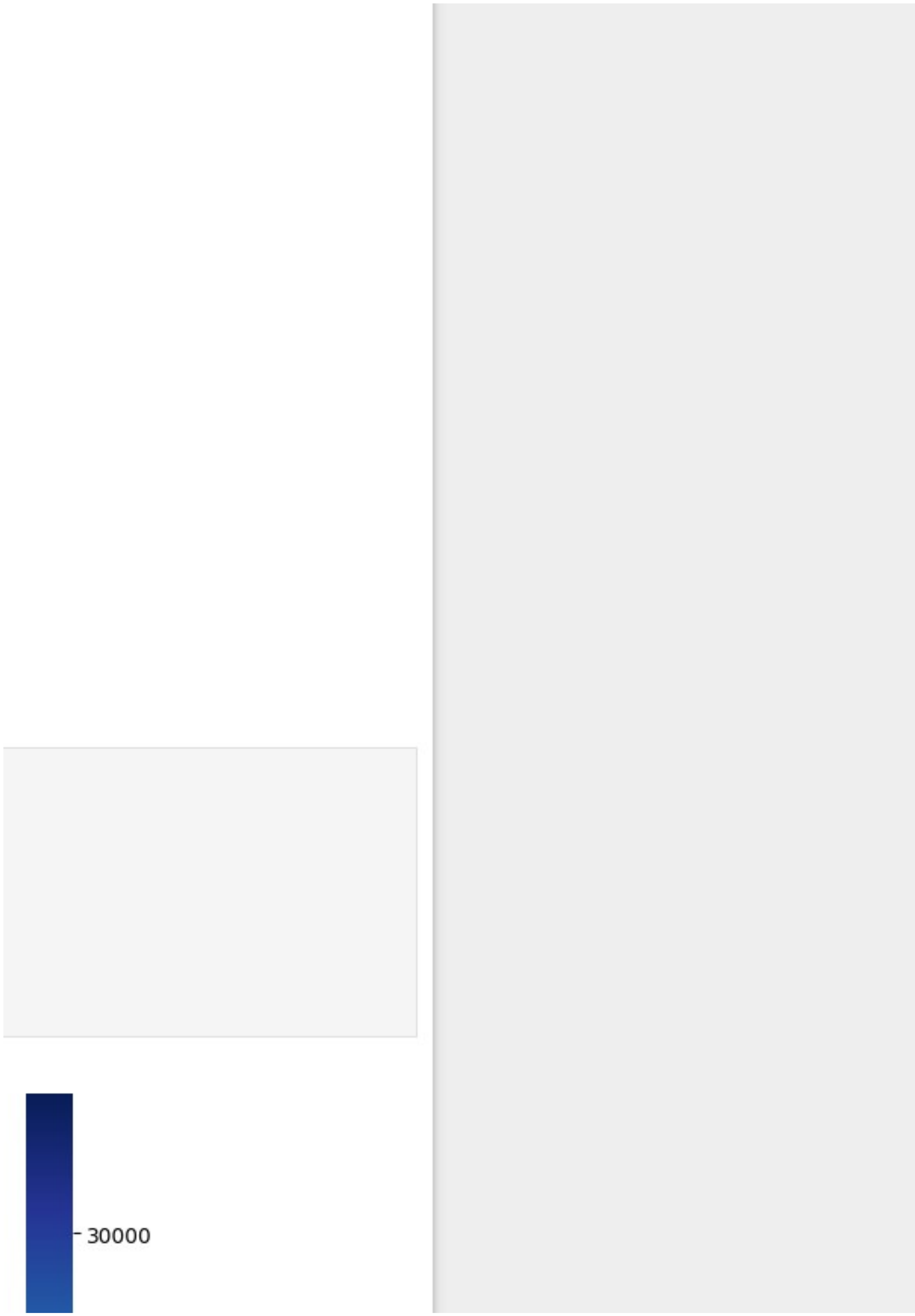
[16]:

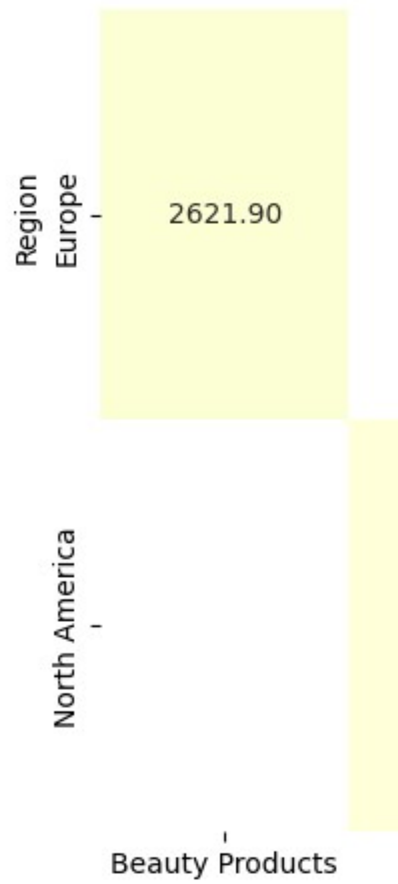
```
pivot_table = df.pivot_table

plt.figure(figsize=(12, 8))
sns.heatmap(pivot_table, ann
plt.title('Sales by Region a
plt.xlabel('Product Category
plt.ylabel('Region')
plt.show()
```



egion/product category

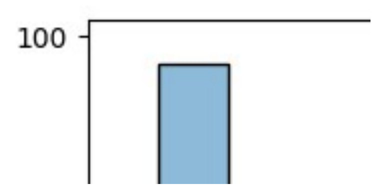
```
:(values='Total Revenue', index='Region', columns='Product Category', aggfunc='sum')


ıot=True, fmt=".2f", cmap="YlGnBu")
ınd Product Category')
')
```

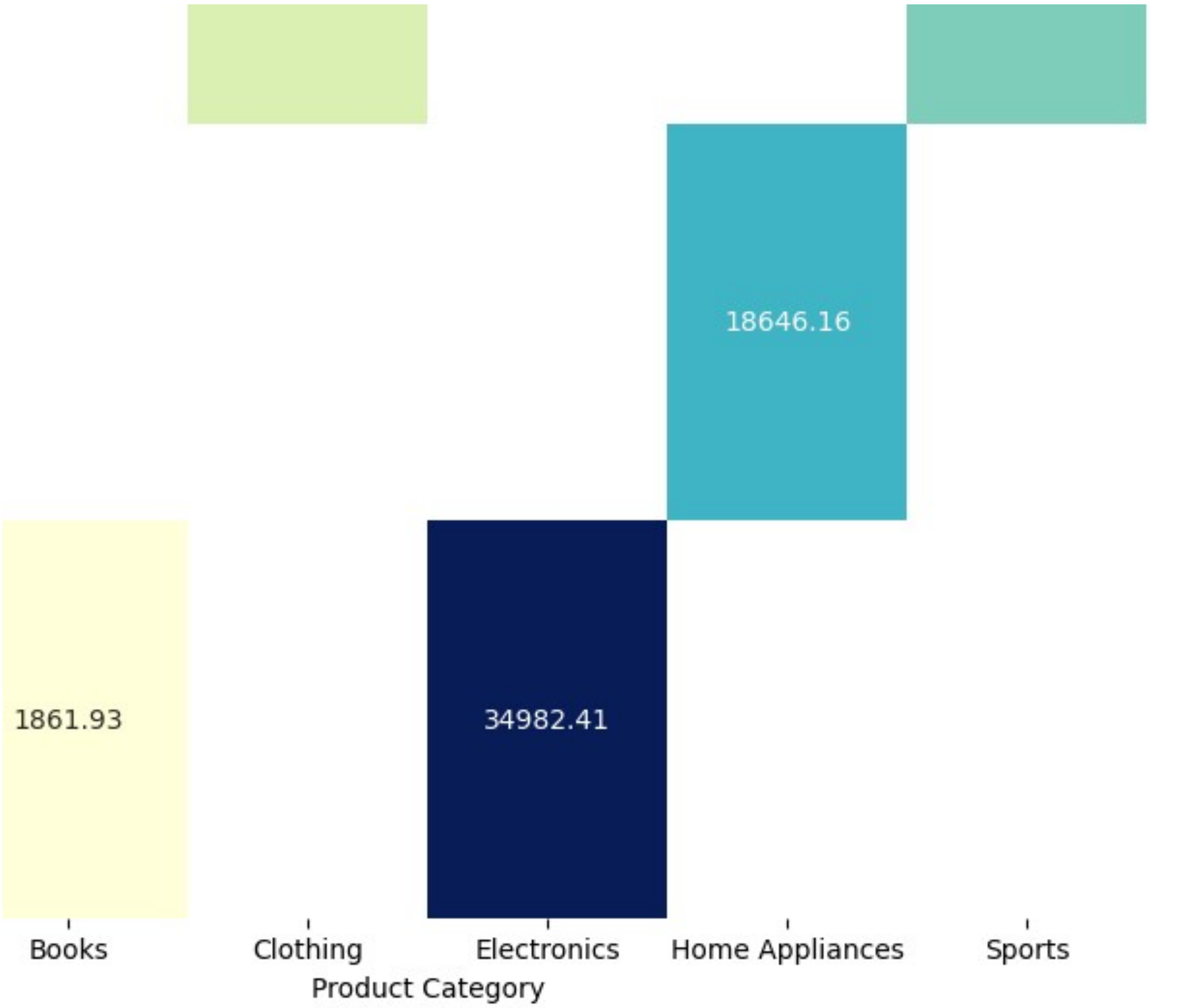

30000



Units Sold Distributio

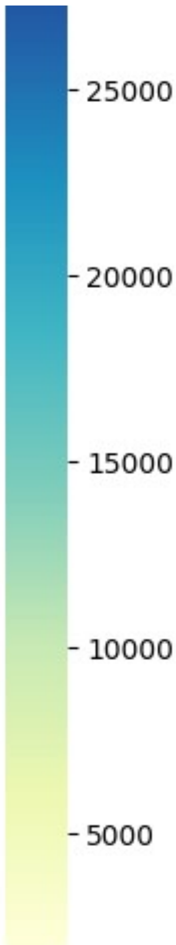
[17]:
```
# Distribution plot of units
plt.figure(figsize=(10, 6))
sns.histplot(df['Units Sold'
plt.title('Distribution of U
plt.xlabel('Units Sold')
plt.ylabel('Frequency')
plt.show()
```

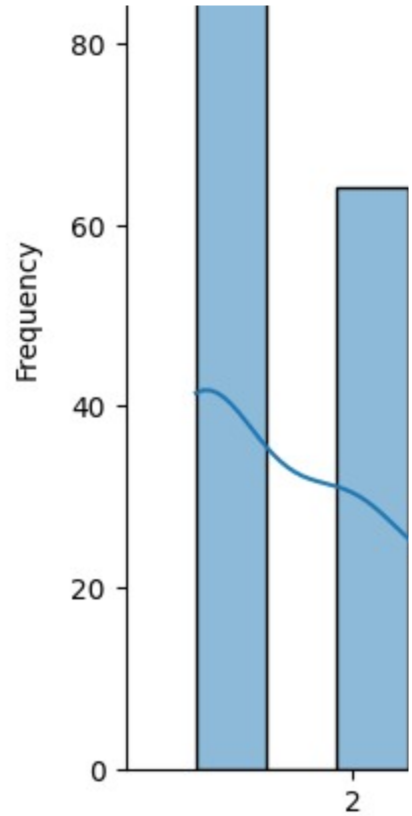




n

```
: sold

], kde=True, bins=20)
Units Sold')
```


25000
20000
15000
10000
5000

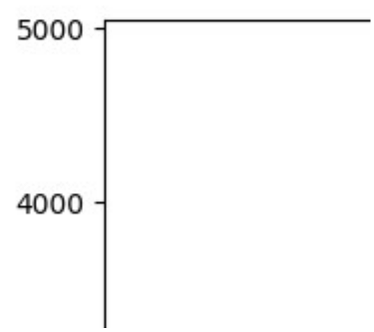


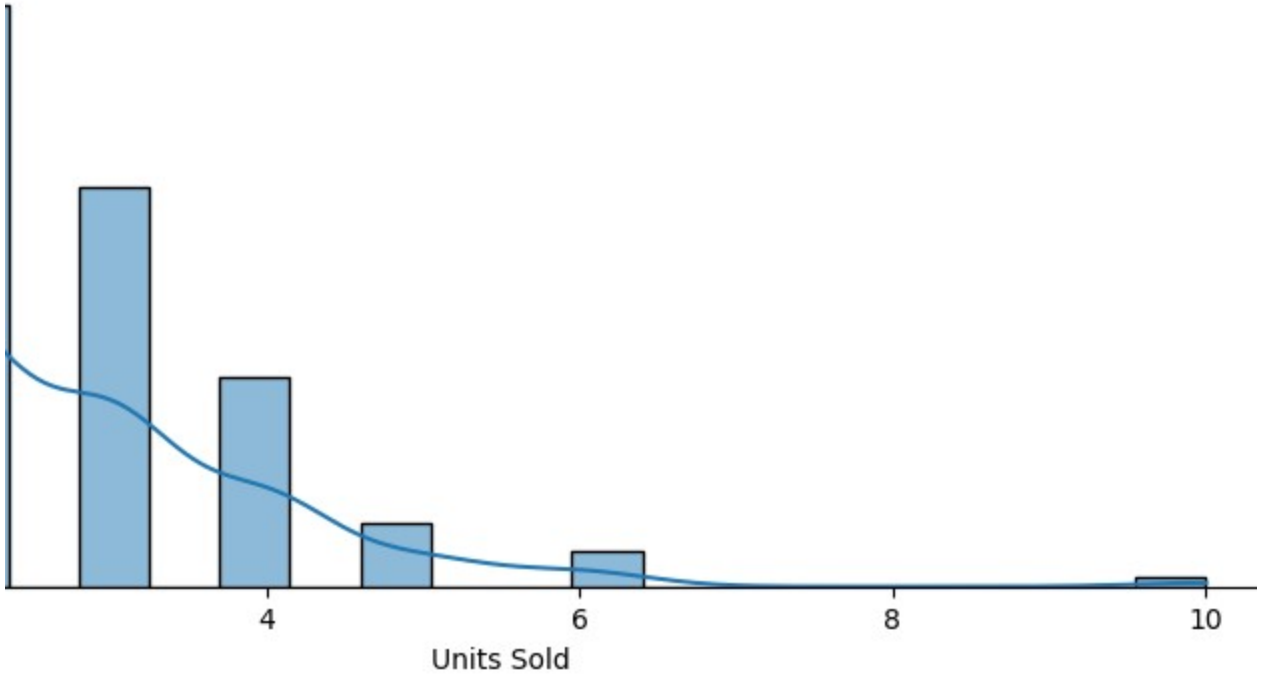
Regression Analysis

Unit price vs Total Revenue

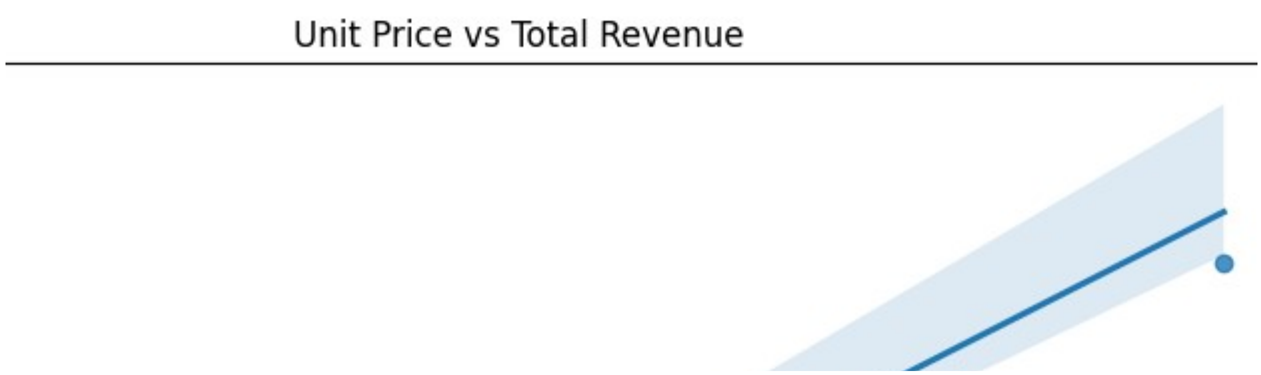
[18]:

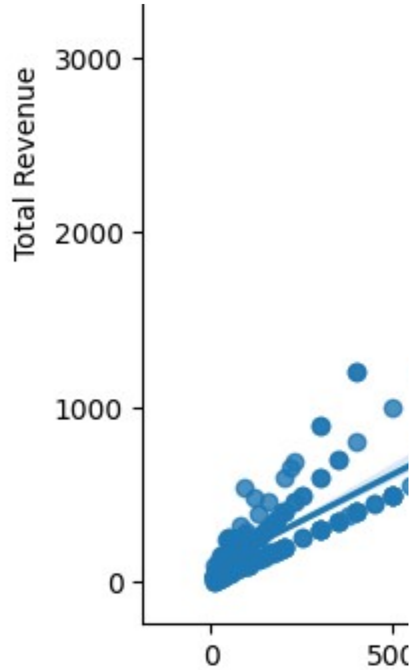
```
plt.figure(figsize=(10, 6))
sns.regplot(x='Unit Price',
plt.title('Unit Price vs Tot
plt.xlabel('Unit Price')
plt.ylabel('Total Revenue')
plt.show()
```






```
y='Total Revenue', data=df)# scatter with regression line
tal Revenue')
```





The positive slope of the Reg line

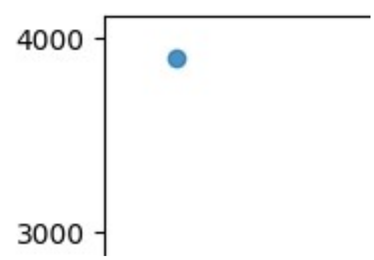
The scatter plot shows a clear up

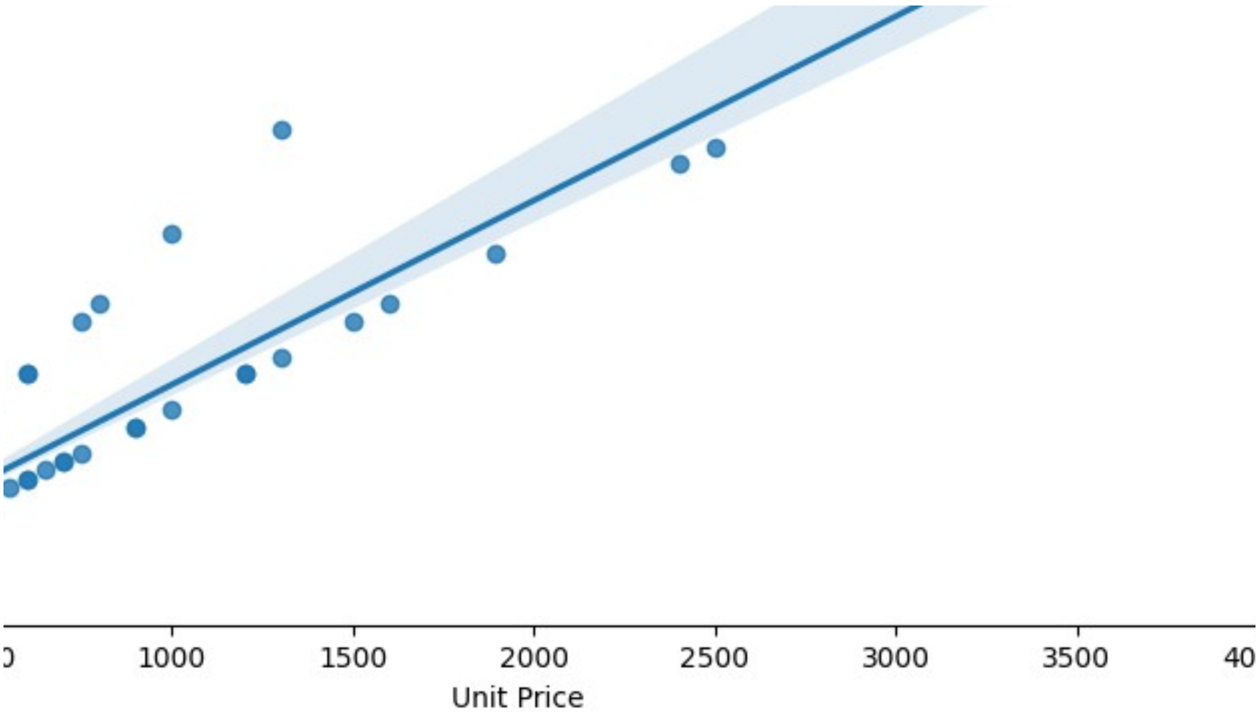
There are some dots that are dist
low for a given unit price. Outlier

Units Sold vs Total Revenue

[19]:

```
plt.figure(figsize=(10, 6))
sns.regplot(x='Units Sold',
plt.title('Units Sold vs Tot
plt.xlabel('Units Sold')
plt.ylabel('Total Revenue')
plt.show()
```





e indicates a +ive relationship between unit price and total revenue.

ward trend, suggesting a strong positive corr bw unit price and total revenue.

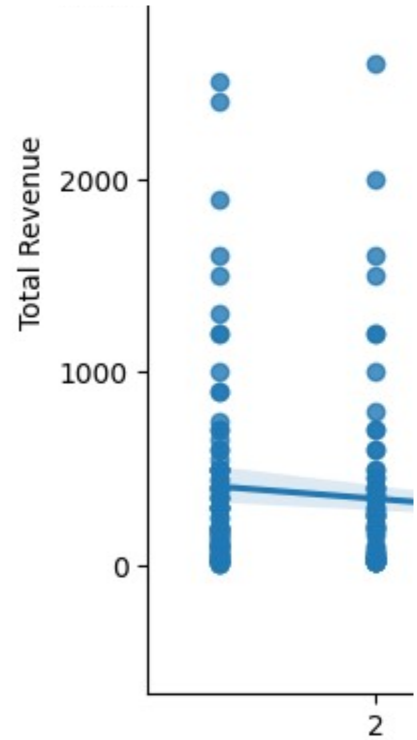
tant from the regression line, indicating outliers. These outliers could represent exceptional cases whe
rs can provide valuable insights, such as identifying products with exceptional performance

```
y='Total Revenue', data=df)
al Revenue')
```

Units Sold vs Total Revenue

00

ere the total revenue is unusually high or



There is a slight negative relatior
total revenue There is significant
result in a wide range of total rev

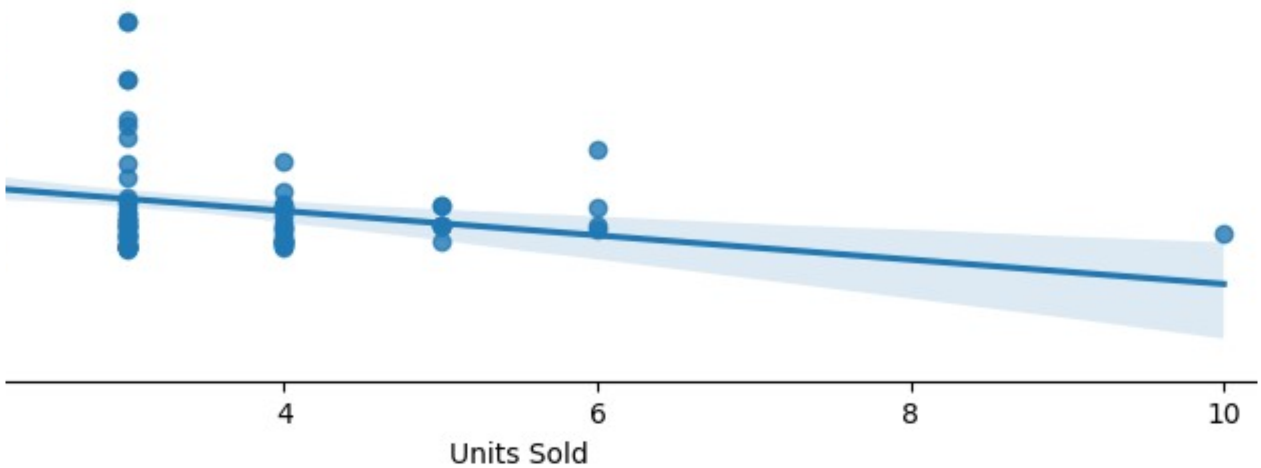
Outliers: A few outliers exist, esp

The trend suggests that as more

transaction with the hi

[20]:
```
transaction = df.loc[df['Tot
transaction
```

[20]:
```
Transaction ID
Date                2024-04-
Product Category
Product Name        Canon EO
Units Sold
Unit Price
Total Revenue
Region                    No
Payment Method
Name: 102, dtype: object
```



ıship between the number of units sold and total revenue This suggests that selling more units does r
: variability in total revenue when fewer units are sold (particularly between 1 and 3 units). This indicate
venues, influenced by unit prices.

ecially at higher total revenue levels for low units sold. These could represent high-value items sold in

units are sold, the total revenue tends to decrease slightly, which could be indicative of discounts or l

ighest total revenue

```
al Revenue'].idxmax()]
```

```
       10103
12 00:00:00
 Electronics
S R5 Camera
           1
     3899.99
     3899.99
rth America
 Credit Card
```

ıot necessarily correspond to higher
es that a small number of units sold can

ı small quantities.

lower-priced items being sold in bulk.

product with the high

```
[21]: top_product_units = df.loc[d
      top_product_units
```

```
[21]: Transaction ID
      Date                    20
      Product Category
      Product Name       Hanes Co
      Units Sold
      Unit Price
      Total Revenue
      Region
      Payment Method
      Name: 62, dtype: object
```

product with most rev

```
[22]: prod= df.groupby('Product Na
      prod
```

```
[22]: Product Name
      Canon EOS R5 Camera
      LG OLED TV
      MacBook Pro 16-inch
      Apple MacBook Pro 16-inch
      iPhone 14 Pro

      Neutrogena Hydro Boost Water
      Biore UV Aqua Rich Watery Es
      The Ordinary Hyaluronic Acid
      The Ordinary Caffeine Soluti
      The Ordinary Niacinamide Ser
      Name: Total Revenue, Length:
```

```
[23]: prod= df.groupby('Product Na
      top5=prod.head(5)
      plt.figure(figsize=(12, 6))
      sns.barplot(x=top5.index, y=
      plt.title('Total Revenue by
      plt.xlabel('Product Name')
      plt.ylabel('Total Revenue')
      plt.xticks(rotation=90)
      plt.show()
```

est units sold

```
lf['Units Sold'].idxmax()]
```

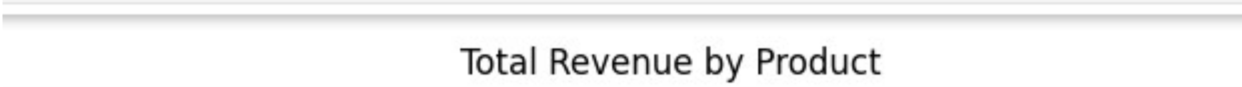
```
             10063
)24-03-03 00:00:00
          Clothing
mfortSoft T-Shirt
                10
              9.99
              99.9
              Asia
        Debit Card
```

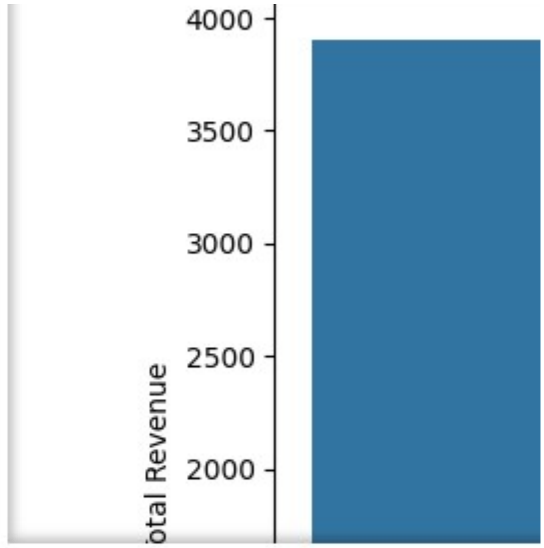
venue produced

```
me')['Total Revenue'].sum().sort_values(ascending = False)
```

```
                     3899.99
                     2599.98
                     2499.99
                     2399.00
                     1999.98
                       ...
 Gel                   16.99
sence Sunscreen        15.00
l Serum                 6.80
.on 5% + EGCG           6.70
'um                     6.50
 232, dtype: float64
```

```
me')['Total Revenue'].sum().sort_values(ascending = False)


top5.values)
Product')
```





Conclusion and Analys

Top Performing Regions: The a

Top Products:

The top 5 products by revenue a
management and marketing stra

Category Analysis:

Electronic, Sports and home appl

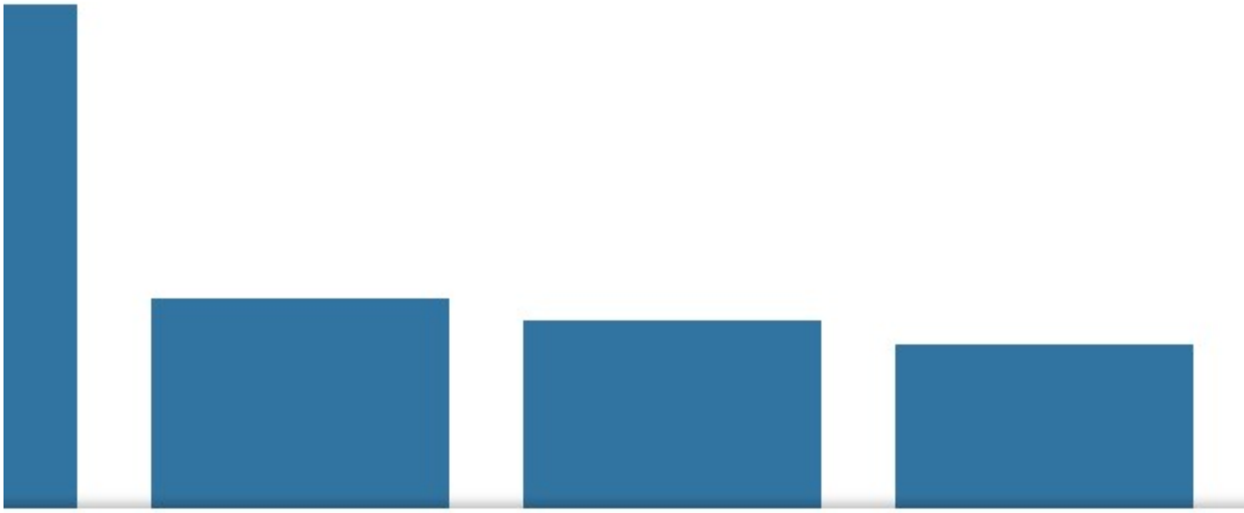
Seasonal Trends: Our Sales data

Payement Methods and Discou
and loyalty. If discounts significa

```
[25]: df.head()
```

[25]:

	Transaction ID	Date
0	10001	2024-01-01
1	10002	2024-01-02
2	10003	2024-01-03
3	10004	2024-01-04
4	10005	2024-01-05



sis

ınalysis identifies regions contributing the most to total revenue. Focus on (North America, Asia) for fu

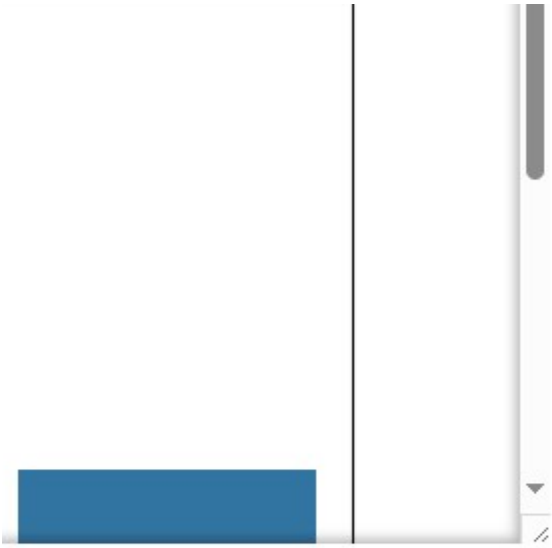
ıre Cannon Camera, LG TV, MacBook, and iphone 14 pro. These products are popular among custome
ıtegies.

liances Category drive signigficant revenue. These categories should be the focal point for expanding

ı over time reveal seasonal trends and peak periods. Preparing for these peaks with adequate invento

ınts The most used payment methods are Debitcard and paypal. Offering discounts on these method
ntly boost sales, they can be strategically used during off-peak times.

Product Category	Product Name	Units Sold	Unit Price	Total Revenue	Region	P
Electronics	iPhone 14 Pro	2	999.99	1999.98	North America	
Home Appliances	Dyson V11 Vacuum	1	499.99	499.99	Europe	
Clothing	Levi's 501 Jeans	3	69.99	209.97	Asia	
Books	The Da Vinci Code	4	15.99	63.96	North America	
Beauty Products	Neutrogena Skincare Set	1	89.99	89.99	Europe	



ıture marketing and sales efforts.

rs and should be prioritized in inventory

product lines

ry and marketing can optimize sales.

s could enhance customer satisfaction

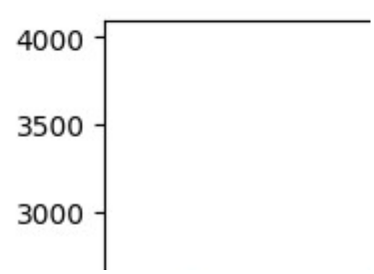
ayment Method
Credit Card
PayPal
Debit Card
Credit Card
PayPal

Sales Forecasting with

[26]:

```
from statsmodels.tsa.statesp
import matplotlib.pyplot as 


sales_data = df.groupby('Dat
sales_data.set_index('Date',


model = SARIMAX(sales_data, 
model_fit = model.fit(disp=F


forecast = model_fit.get_for
forecast_index = pd.date_ran
forecast_series = forecast.p
forecast_series.index = fore

plt.figure(figsize=(10, 5))#
plt.plot(sales_data, label='
plt.plot(forecast_series, la
plt.xlabel('Date')
plt.ylabel('Total Revenue')
plt.title('Sales Forecast')
plt.legend()
plt.show()
```

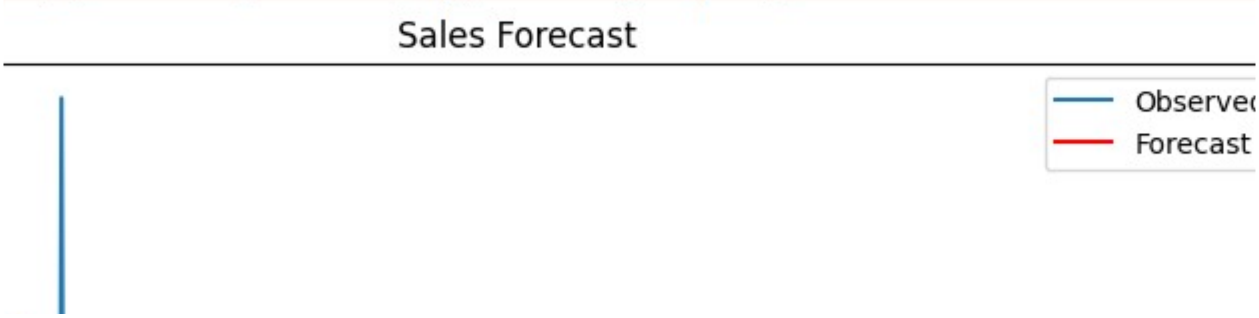
```
C:\Users\pc\AppData\Local\Pr
rovided, so inferred frequen
  self._init_dates(dates, fr
C:\Users\pc\AppData\Local\Pr
rovided, so inferred frequen
  self._init_dates(dates, fr
C:\Users\pc\AppData\Local\Te
E' instead.
  forecast_index = pd.date_r
```



ARIMA

```
ace.sarimax import SARIMAX
plt


e')['Total Revenue'].sum().reset_index()# aggrigate sales by date
 inplace=True)


order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))# MODEL - forcasts sales of next 12Months
alse)


ecast(steps=12)#FOrecast sales
ge(start=sales_data.index[-1], periods=12, freq='M')
redicted_mean
cast_index

visualization
Observed')
bel='Forecast', color='red')
```

```
ograms\Python\Python312\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueW
cy D will be used.
eq)
ograms\Python\Python312\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueW
cy D will be used.
eq)
mp\ipykernel_17996\3423825647.py:14: FutureWarning: 'M' is deprecated and will be remov

ange(start=sales_data.index[-1], periods=12, freq='M')
```

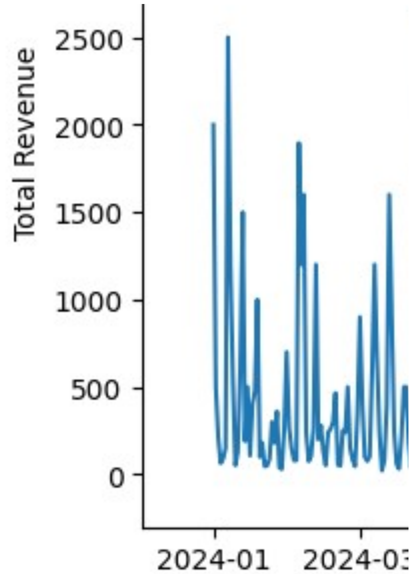


arning: No frequency information was p

arning: No frequency information was p

ed in a future version, please use 'M

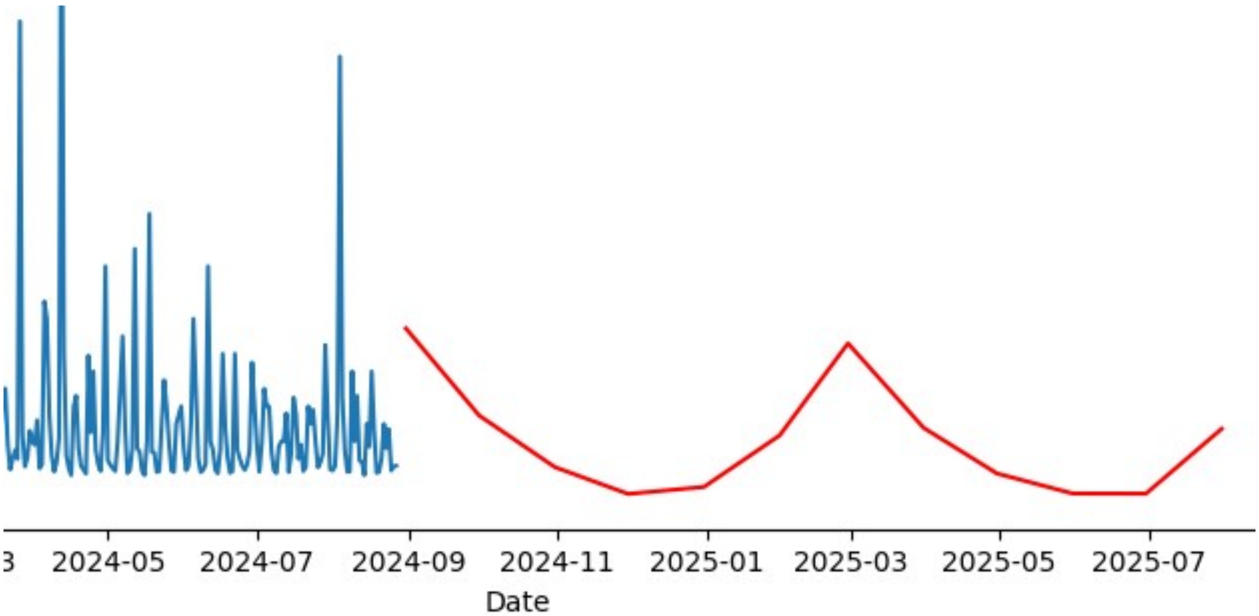
d



Forecast data - 2024/08 - 2025

Sales are expected to be low dur

[ ]:



/07 The model predicts 2 peaks in sept and another in Feb 2025. THese peaks may be due to seasona
ing october to Jan and then again from march to june.

al events, holiday or promtional periods.