Problem Statement

Sales forecasting is a crucial tool for retail organizations to develop strategies that align with demand and create promotions to enhance sales. Walmart, a billion-dollar American retail giant, operates over 10,000 stores worldwide and more than 4,000 across the United States (ref: https://corporate.walmart.com/about/location-facts). Accurate sales forecasting allows the company to manage inventory, predict revenue, and make informed decisions regarding new investments. Achieving predetermined targets early in the season can positively influence stock prices and shape investors' perceptions. On the other hand, missing these projections could significantly harm stock prices, which would be particularly detrimental for a large company like Walmart.

Aim

The goal of this project is to develop a model that can forecast store sales based on past sales patterns. With this model, decision-makers can make critical business decisions, maximize revenue, and improve sales in underperforming departments.

Data Exploration

In [1]:

```
# <== Importing Necessary Libraries ==>
import os
import math
import sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
# loading all datasets
df_features = pd.read_csv("/kaggle/input/walmart-sales-forecast/features.csv")
df_store = pd.read_csv("/kaggle/input/walmart-sales-forecast/stores.csv")
df_train = pd.read_csv("/kaggle/input/walmart-sales-forecast/train.csv")
df_test = pd.read_csv("/kaggle/input/walmart-sales-forecast/test.csv")
```

In [3]:

```
pd.options.display.max_rows=100
pd.pandas.set_option('display.max_columns', None)
```

Dataset Description

1. **Features.csv**: Additional data related to the store, department, and regional activity for the given dates.
2. **train.csv**: This is the historical training data, which covers 2010–02–05 to 2012-08-17.
3. **test.csv**: Similar to train.csv, we've to predict weekly sales.
4. **stores.csv**: Contains anonymized information about the 45 stores, indicating the type and size of the store.

In [4]:
```
# lowercase all the column names and replace spaces with _
df_train.columns = df_train.columns.str.lower().str.replace(' ', '_')
df_store.columns = df_store.columns.str.lower().str.replace(' ', '_')
df_features.columns = df_features.columns.str.lower().str.replace(' ', '_')
```

Exploring - Features.csv

In [5]:
```
print("Shape of dataset (rows x columns): ", df_features.shape)
```

```
Shape of dataset (rows x columns):  (8190, 12)
```

In [6]:
```
df_features.head(10)
```

Out[6]:

	store	date	temperature	fuel_price	markdown1	markdown2	markdov
0	1	2010-02-05	42.31	2.572	NaN	NaN	
1	1	2010-02-12	38.51	2.548	NaN	NaN	
2	1	2010-02-19	39.93	2.514	NaN	NaN	
3	1	2010-02-26	46.63	2.561	NaN	NaN	
4	1	2010-03-05	46.50	2.625	NaN	NaN	
5	1	2010-03-12	57.79	2.667	NaN	NaN	
6	1	2010-03-19	54.58	2.720	NaN	NaN	
7	1	2010-03-26	51.45	2.732	NaN	NaN	
8	1	2010-04-02	62.27	2.719	NaN	NaN	
9	1	2010-04-09	65.86	2.770	NaN	NaN	

In [7]:
```
df_features.head(-10)
```

Out[7]:

	store	date	temperature	fuel_price	markdown1	markdown2	marl
0	1	2010-02-05	42.31	2.572	NaN	NaN	
1	1	2010-02-12	38.51	2.548	NaN	NaN	
2	1	2010-02-19	39.93	2.514	NaN	NaN	
3	1	2010-02-26	46.63	2.561	NaN	NaN	
4	1	2010-03-05	46.50	2.625	NaN	NaN	
...	...	...	...	...	...	...	
8175	45	2013-04-19	56.27	3.676	1399.81	39.89	
8176	45	2013-04-26	50.64	3.615	1260.65	NaN	
8177	45	2013-05-03	56.07	3.592	8345.40	6.00	
8178	45	2013-05-10	58.86	3.583	4689.18	440.82	
8179	45	2013-05-17	60.59	3.614	4515.35	667.88	

8180 rows × 12 columns

In [8]:

```
# Display column names
print(df_features.columns)
```

```
Index(['store', 'date', 'temperature', 'fuel_price', 'markdown1', 'markdown2',
       'markdown3', 'markdown4', 'markdown5', 'cpi', 'unemployment',
       'isholiday'],
      dtype='object')
```

Column wise description

- Store: Denotes the Store Number
- Date: Dates (Data) from 2010 - 2012
- Temperature: Temperature in Fahrenheit
- Fuel Price: Price of Fuel that day
- Markdown1, Markdown2, Markdown3, Markdown4: Anonymized data related to promotional markdowns at Walmart
- CPI: The consumer price index. It measures the monthly changes in prices paid by US consumers
- Unemployement: Rate of Unemployment
- IsHoliday: Is that particular day holiday or working day

In [9]:

```
# Display summary information about the dataset
print(df_features.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8190 entries, 0 to 8189
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   store         8190 non-null   int64  
 1   date          8190 non-null   object 
 2   temperature   8190 non-null   float64
 3   fuel_price    8190 non-null   float64
 4   markdown1     4032 non-null   float64
 5   markdown2     2921 non-null   float64
 6   markdown3     3613 non-null   float64
 7   markdown4     3464 non-null   float64
 8   markdown5     4050 non-null   float64
 9   cpi           7605 non-null   float64
 10  unemployment  7605 non-null   float64
 11  isholiday     8190 non-null   bool   
dtypes: bool(1), float64(9), int64(1), object(1)
memory usage: 712.0+ KB
None
```

In [10]:
```
# Display statistical summary of the dataset
print(df_features.describe())
```

```
             store  temperature   fuel_price      markdown1      markdown2  \
count  8190.000000  8190.000000  8190.000000    4032.000000    2921.000000   
mean     23.000000    59.356198     3.405992    7032.371786    3384.176594   
std      12.987966    18.678607     0.431337    9262.747448    8793.583016   
min       1.000000    -7.290000     2.472000   -2781.450000    -265.760000   
25%      12.000000    45.902500     3.041000    1577.532500      68.880000   
50%      23.000000    60.710000     3.513000    4743.580000     364.570000   
75%      34.000000    73.880000     3.743000    8923.310000    2153.350000   
max      45.000000   101.950000     4.468000  103184.980000  104519.540000   

           markdown3     markdown4      markdown5          cpi  unemployment  
count    3613.000000   3464.000000    4050.000000  7605.000000   7605.000000  
mean     1760.100180   3292.935886    4132.216422   172.460809      7.826821  
std     11276.462208   6792.329861   13086.690278    39.738346      1.877259  
min      -179.260000      0.220000    -185.170000   126.064000      3.684000  
25%         6.600000    304.687500    1440.827500   132.364839      6.634000  
50%        36.260000   1176.425000    2727.135000   182.764003      7.806000  
75%       163.150000   3310.007500    4832.555000   213.932412      8.567000  
max    149483.310000  67474.850000  771448.100000   228.976456     14.313000  
```

In [11]:
```
# Check for missing values
print(df_features.isnull().sum())
```

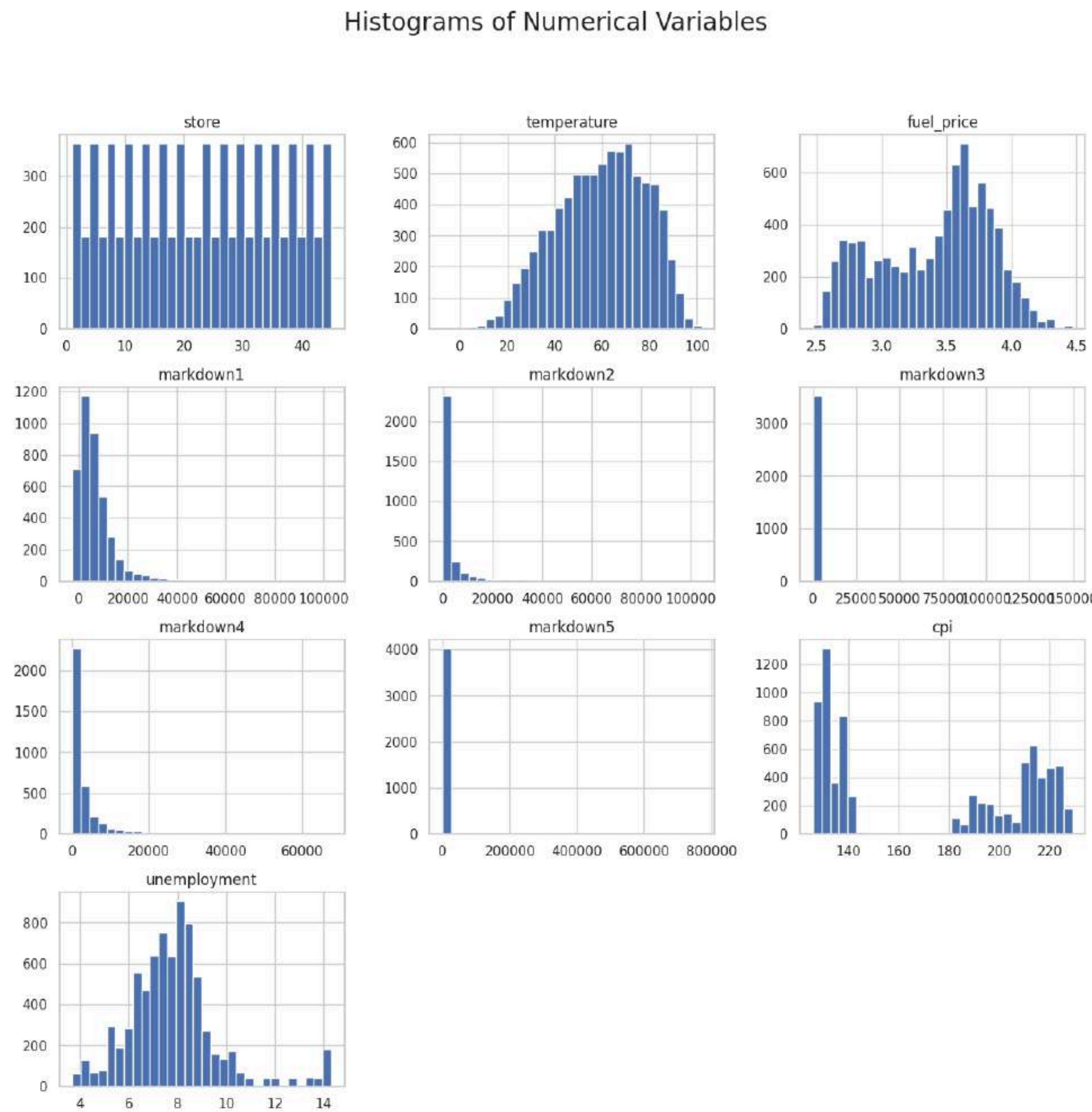
```
store              0
date               0
temperature        0
fuel_price         0
markdown1       4158
markdown2       5269
markdown3       4577
markdown4       4726
markdown5       4140
cpi              585
unemployment     585
isholiday          0
dtype: int64
```

In [12]:
```
# Number of unique values
df_features.nunique()
```

Out[12]:
```
store             45
date             182
temperature     4178
fuel_price      1011
markdown1       4023
markdown2       2715
markdown3       2885
markdown4       3405
markdown5       4045
cpi             2505
unemployment     404
isholiday          2
dtype: int64
```

Data Visualization

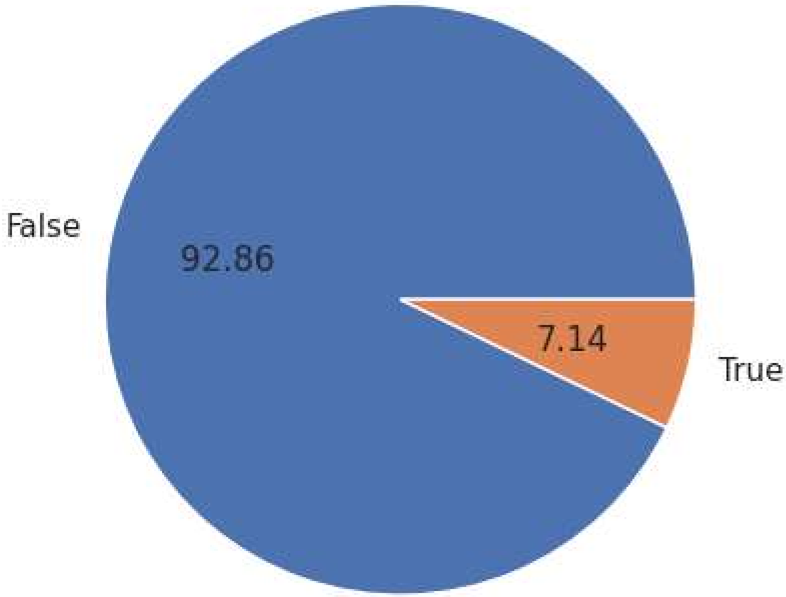
In [13]:
```
# Setting plot style
sns.set(style="whitegrid")

# Plot histograms for numerical variables
df_features.hist(bins=30, figsize=(15, 14))
plt.suptitle('Histograms of Numerical Variables', fontsize=20)
plt.show()
```



In [14]:

```
df_features.groupby('isholiday').size().plot(kind='pie', autopct='%.2f')
# False: is not a holiday, True: its a holiday
```

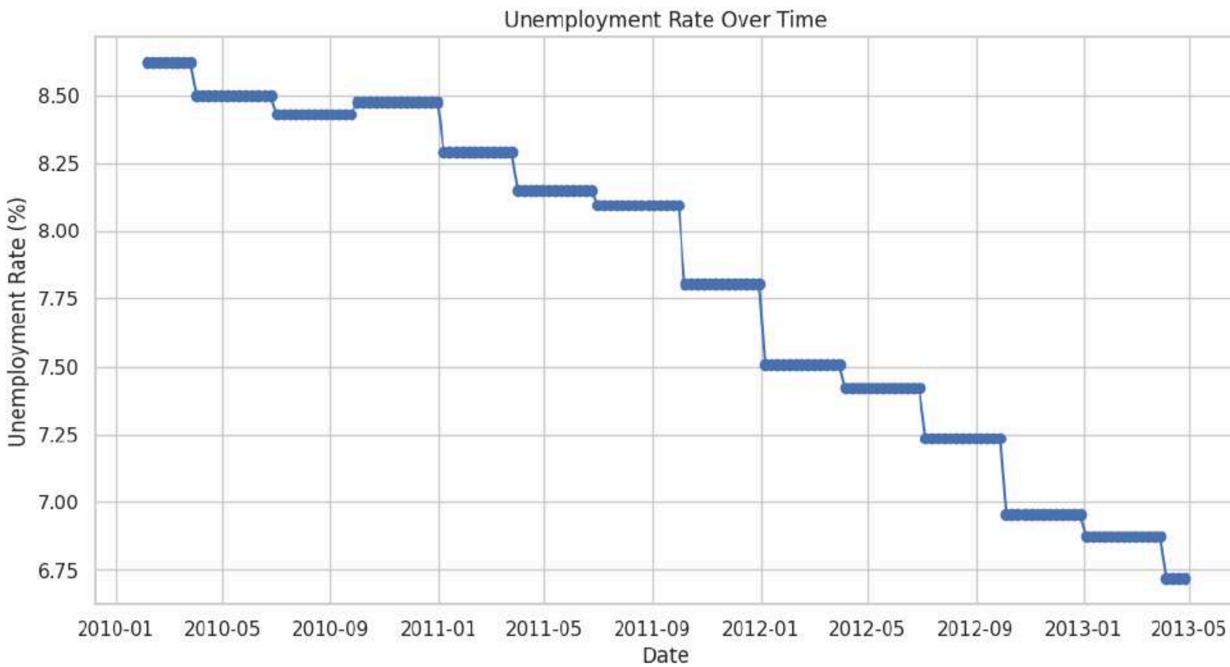
Out[14]: <Axes: >



Unemployment rate v/s time

In [15]:

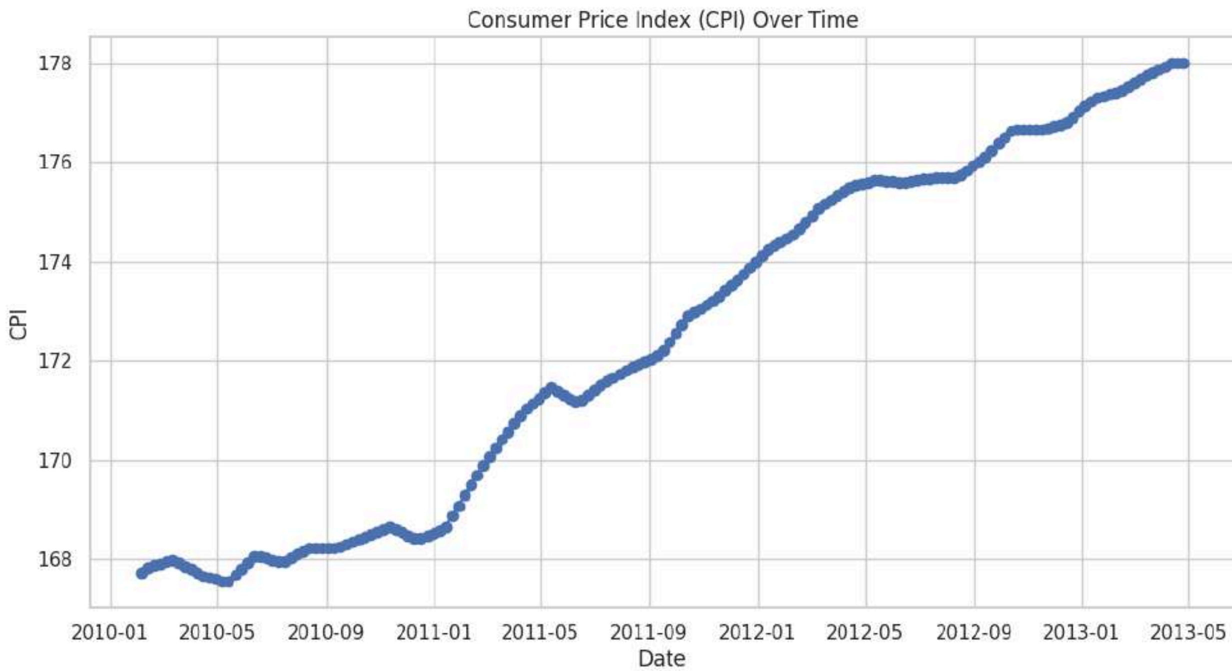
```
# Convert the 'date' column to datetime format
df_features['date'] = pd.to_datetime(df_features['date'])  # Let Pandas infer

# Group by 'date' and calculate the average unemployment rate
df_grouped = df_features.groupby('date')['unemployment'].mean().reset_index()

# Plotting the data
plt.figure(figsize=(12, 6))
plt.plot(df_grouped['date'], df_grouped['unemployment'], marker='o', linestyle

# Adding titles and labels
plt.title('Unemployment Rate Over Time')
plt.xlabel('Date')
plt.ylabel('Unemployment Rate (%)')

# Show plot
plt.grid(True)
plt.show()
```



CPI Index v/s time

In [16]:

```
# Group by 'Date' and calculate the average CPI
df_grouped_cpi = df_features.groupby('date')['cpi'].mean().reset_index()

# Plotting the data
plt.figure(figsize=(12, 6))
plt.plot(df_grouped_cpi['date'], df_grouped_cpi['cpi'], marker='o', linestyle=

# Adding titles and labels
plt.title('Consumer Price Index (CPI) Over Time')
plt.xlabel('Date')
plt.ylabel('CPI')

# Show plot
plt.grid(True)
plt.show()
```



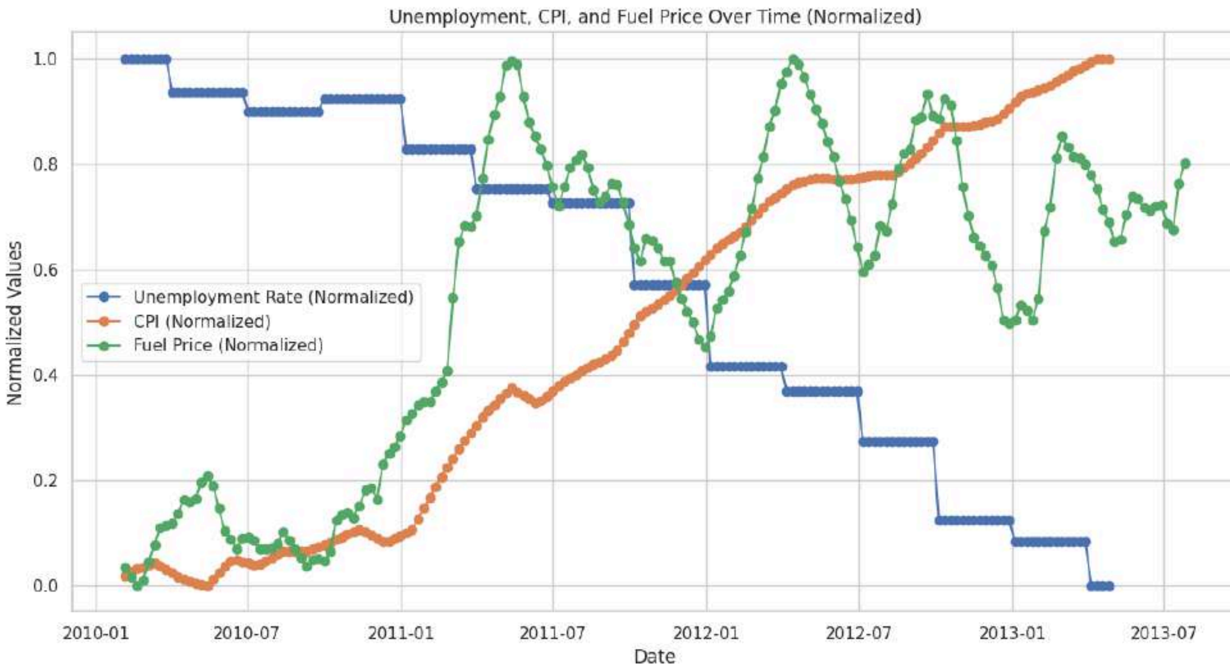
Fuel Price v/s time

In [17]:

```
# Group by 'Date' and calculate the average Fuel Price
df_grouped_fuel_price = df_features.groupby('date')['fuel_price'].mean().reset

# Plotting the data
plt.figure(figsize=(12, 6))
plt.plot(df_grouped_fuel_price['date'], df_grouped_fuel_price['fuel_price'], m

# Adding titles and labels
plt.title('Fuel Price Over Time')
plt.xlabel('Date')
plt.ylabel('Fuel Price')

# Show plot
plt.grid(True)
plt.show()
```

Normalized comparision of Unemployment rate v/s CPI v/s Fuel Price

In [18]:
```
# Group by 'Date' and calculate the average for each metric
df_grouped = df_features.groupby('date').mean().reset_index()

# Normalize the data
df_grouped['Unemployment_norm'] = (df_grouped['unemployment'] - df_grouped['ur
df_grouped['CPI_norm'] = (df_grouped['cpi'] - df_grouped['cpi'].min()) / (df_g
df_grouped['Fuel_Price_norm'] = (df_grouped['fuel_price'] - df_grouped['fuel_p

# Plotting the data
plt.figure(figsize=(14, 7))

# Plot each normalized metric
plt.plot(df_grouped['date'], df_grouped['Unemployment_norm'], marker='o', line
plt.plot(df_grouped['date'], df_grouped['CPI_norm'], marker='o', linestyle='-'
plt.plot(df_grouped['date'], df_grouped['Fuel_Price_norm'], marker='o', linest

# Adding titles and labels
plt.title('Unemployment, CPI, and Fuel Price Over Time (Normalized)')
plt.xlabel('Date')
plt.ylabel('Normalized Values')
plt.legend()

# Show plot
plt.grid(True)
plt.show()
```

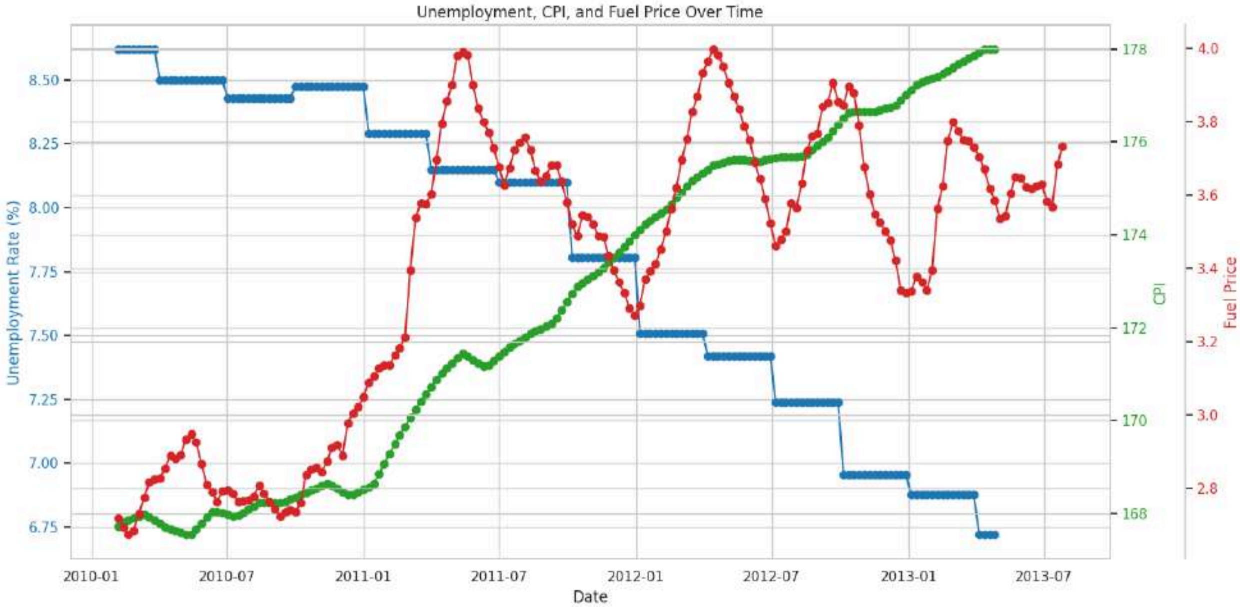


Comparing Unemployment v/s CPI v/s Fuel Price, one frame, multiple Y-Axis

In [19]:

```
# Group by 'Date' and calculate the average for each metric
df_grouped = df_features.groupby('date').mean().reset_index()

# Plotting the data
fig, ax1 = plt.subplots(figsize=(14, 7))

# Plotting Unemployment on primary y-axis
ax1.plot(df_grouped['date'], df_grouped['unemployment'], color='tab:blue', mar
ax1.set_xlabel('Date')
ax1.set_ylabel('Unemployment Rate (%)', color='tab:blue')
ax1.tick_params(axis='y', labelcolor='tab:blue')

# Creating a second y-axis for CPI
ax2 = ax1.twinx()
ax2.plot(df_grouped['date'], df_grouped['cpi'], color='tab:green', marker='o',
ax2.set_ylabel('CPI', color='tab:green')
ax2.tick_params(axis='y', labelcolor='tab:green')

# Creating a third y-axis for Fuel Price
ax3 = ax1.twinx()
ax3.spines['right'].set_position(('outward', 60))
ax3.plot(df_grouped['date'], df_grouped['fuel_price'], color='tab:red', marker
ax3.set_ylabel('Fuel Price', color='tab:red')
ax3.tick_params(axis='y', labelcolor='tab:red')

# Adding titles
plt.title('Unemployment, CPI, and Fuel Price Over Time')
```

```
# Show plot
fig.tight_layout()
plt.show()
```



Data Preprocessing

Taking care of missing values

In [20]:
```
# filling missing values
df_features['cpi'].fillna(df_features['cpi'].median(),inplace=True)
df_features['unemployment'].fillna(df_features['unemployment'].median(),inplac
```

```
/tmp/ipykernel_17/3803472090.py:2: FutureWarning: A value is trying to be set o
n a copy of a DataFrame or Series through chained assignment using an inplace m
ethod.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_features['cpi'].fillna(df_features['cpi'].median(),inplace=True)
/tmp/ipykernel_17/3803472090.py:3: FutureWarning: A value is trying to be set o
n a copy of a DataFrame or Series through chained assignment using an inplace m
ethod.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_features['unemployment'].fillna(df_features['unemployment'].median(),inpla
ce=True)
```

In [21]:

```
# replacing the markdown values with 0, as there is no information in the data
for i in range(1, 6):
    df_features["markdown" + str(i)] = df_features["markdown" + str(i)].apply(
    df_features["markdown" + str(i)].fillna(value=0, inplace=True)
```

```
/tmp/ipykernel_17/2139483102.py:4: FutureWarning: A value is trying to be set o
n a copy of a DataFrame or Series through chained assignment using an inplace m
ethod.
The behavior will change in pandas 3.0. This inplace method will never work bec
ause the intermediate object on which we are setting values always behaves as a
copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.me
thod({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, t
o perform the operation inplace on the original object.


  df_features["markdown" + str(i)].fillna(value=0, inplace=True)
```

In [22]:

```
df_features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8190 entries, 0 to 8189
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   store         8190 non-null   int64
 1   date          8190 non-null   datetime64[ns]
 2   temperature   8190 non-null   float64
 3   fuel_price    8190 non-null   float64
 4   markdown1     8190 non-null   float64
 5   markdown2     8190 non-null   float64
 6   markdown3     8190 non-null   float64
 7   markdown4     8190 non-null   float64
 8   markdown5     8190 non-null   float64
 9   cpi           8190 non-null   float64
 10  unemployment  8190 non-null   float64
 11  isholiday     8190 non-null   bool
dtypes: bool(1), datetime64[ns](1), float64(9), int64(1)
memory usage: 712.0 KB
```

In [23]:
```
df_features.head()
```

Out[23]:

	store	date	temperature	fuel_price	markdown1	markdown2	markdo
0	1	2010-02-05	42.31	2.572	0.0	0.0	
1	1	2010-02-12	38.51	2.548	0.0	0.0	
2	1	2010-02-19	39.93	2.514	0.0	0.0	
3	1	2010-02-26	46.63	2.561	0.0	0.0	
4	1	2010-03-05	46.50	2.625	0.0	0.0	

In [24]:
```
df_train.shape, df_store.shape, df_features.shape
```

Out[24]:
```
((421570, 5), (45, 3), (8190, 12))
```

Merging DataFrames

Merging 3 dataframes:

1. `features.csv` : Description already provided above.
2. `train.csv` : This is the historical training data, which covers 2010–02–05 to 2012–11–01, with following fields:
   - `Store` : *The store number*
   - `Dept` : *The department number*
   - `Date` : *The week*
   - `Weekly_Sales` : *Sales for the given department in the*

given store

   - `IsHoliday` : *Whether the week is a special holiday week*

3. `stores.csv` : This file contains anonymized information about the 45 stores, indicating the type and size of the store:

   - `Store` : *Stores numbered from 1 to 45*
   - `Type` : *Store type has been provided, there are 3 types — A, B and C*
   - `Size` : *Stores size has provided*

In [25]:
```
train_df = pd.merge(df_train,df_store,on='store',how='left')
```

In [26]:
```
train_df.head()
```

Out[26]:

	store	dept	date	weekly_sales	isholiday	type	size
0	1	1	2010-02-05	24924.50	False	A	151315
1	1	1	2010-02-12	46039.49	True	A	151315
2	1	1	2010-02-19	41595.55	False	A	151315
3	1	1	2010-02-26	19403.54	False	A	151315
4	1	1	2010-03-05	21827.90	False	A	151315

In [27]:
```
df_features.head()
```

Out[27]:

	store	date	temperature	fuel_price	markdown1	markdown2	markdo
0	1	2010-02-05	42.31	2.572	0.0	0.0	
1	1	2010-02-12	38.51	2.548	0.0	0.0	
2	1	2010-02-19	39.93	2.514	0.0	0.0	
3	1	2010-02-26	46.63	2.561	0.0	0.0	
4	1	2010-03-05	46.50	2.625	0.0	0.0	

In [28]:
```
print(df_features.columns)
print(df_train.columns)
print(df_store.columns)
```

```
Index(['store', 'date', 'temperature', 'fuel_price', 'markdown1', 'markdown2',
       'markdown3', 'markdown4', 'markdown5', 'cpi', 'unemployment',
       'isholiday'],
      dtype='object')
Index(['store', 'dept', 'date', 'weekly_sales', 'isholiday'], dtype='object')
Index(['store', 'type', 'size'], dtype='object')
```

Ensuring Consistency in merged dataframe

In [29]:
```
# Convert store columns to string to ensure consistency
df_features['store'] = df_features['store'].astype(str)
df_train['store'] = df_train['store'].astype(str)
df_store['store'] = df_store['store'].astype(str)

# Convert date columns to datetime if they are not already
df_features['date'] = pd.to_datetime(df_features['date'])
df_train['date'] = pd.to_datetime(df_train['date'])

# Strip any leading/trailing spaces in the store columns
df_features['store'] = df_features['store'].str.strip()
df_train['store'] = df_train['store'].str.strip()
df_store['store'] = df_store['store'].str.strip()
```

In [30]:
```
# Merge df_features with df_train on 'store' and 'date'
merged_df = pd.merge(df_features, df_train, on=['store', 'date'], how='inner')

# Merge the result with df_stores on 'store' only
final_merged_df = pd.merge(merged_df, df_store, on='store', how='inner')
```

In [31]:
```
final_merged_df.head()
```

Out[31]:

	store	date	temperature	fuel_price	markdown1	markdown2	markdo
0	1	2010-02-05	42.31	2.572	0.0	0.0	
1	1	2010-02-05	42.31	2.572	0.0	0.0	
2	1	2010-02-05	42.31	2.572	0.0	0.0	
3	1	2010-02-05	42.31	2.572	0.0	0.0	
4	1	2010-02-05	42.31	2.572	0.0	0.0	

In [32]:
```
final_merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 421570 entries, 0 to 421569
Data columns (total 17 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   store         421570 non-null  object
 1   date          421570 non-null  datetime64[ns]
 2   temperature   421570 non-null  float64
 3   fuel_price    421570 non-null  float64
 4   markdown1     421570 non-null  float64
 5   markdown2     421570 non-null  float64
 6   markdown3     421570 non-null  float64
 7   markdown4     421570 non-null  float64
 8   markdown5     421570 non-null  float64
 9   cpi           421570 non-null  float64
 10  unemployment  421570 non-null  float64
 11  isholiday_x   421570 non-null  bool
 12  dept          421570 non-null  int64
 13  weekly_sales  421570 non-null  float64
 14  isholiday_y   421570 non-null  bool
 15  type          421570 non-null  object
 16  size          421570 non-null  int64
dtypes: bool(2), datetime64[ns](1), float64(10), int64(2), object(2)
memory usage: 49.0+ MB
```

In [33]:

```
final_merged_df['date'] = pd.to_datetime(final_merged_df['date'],errors='coerc
final_merged_df.sort_values(by=['date'],inplace=True)
final_merged_df.set_index(final_merged_df.date, inplace=True)
final_merged_df.head()
```

Out[33]:

date	store	date	temperature	fuel_price	markdown1	markdown2
2010-02-05	1	2010-02-05	42.31	2.572	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0

In [34]:

```
#checking whether the column IsHoliday_x and IsHoliday_y are same or not
final_merged_df['isholiday_x'].isin(final_merged_df['isholiday_y']).all()
```

Out[34]: True

In [35]:

```
#Since these two columns are same so drop any one column and make another colu
final_merged_df.drop(columns='isholiday_x',inplace=True)
final_merged_df.rename(columns={"isholiday_y" : "IsHoliday"}, inplace=True)
final_merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 421570 entries, 2010-02-05 to 2012-10-26
Data columns (total 16 columns):
 #   Column        Non-Null Count   Dtype         
---  ------        --------------   -----         
 0   store         421570 non-null  object        
 1   date          421570 non-null  datetime64[ns]
 2   temperature   421570 non-null  float64       
 3   fuel_price    421570 non-null  float64       
 4   markdown1     421570 non-null  float64       
 5   markdown2     421570 non-null  float64       
 6   markdown3     421570 non-null  float64       
 7   markdown4     421570 non-null  float64       
 8   markdown5     421570 non-null  float64       
 9   cpi           421570 non-null  float64       
 10  unemployment  421570 non-null  float64       
 11  dept          421570 non-null  int64         
 12  weekly_sales  421570 non-null  float64       
 13  IsHoliday     421570 non-null  bool          
 14  type          421570 non-null  object        
 15  size          421570 non-null  int64         
dtypes: bool(1), datetime64[ns](1), float64(10), int64(2), object(2)
memory usage: 51.9+ MB
```

In [36]:
```python
final_merged_df.head()
```

Out[36]:

	store	date	temperature	fuel_price	markdown1	markdown2
date						
2010-02-05	1	2010-02-05	42.31	2.572	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0

In [37]:
```python
# final_merged_df['Year'] = final_merged_df['date'].dt.year
# final_merged_df['Month'] = final_merged_df['date'].dt.month
# # final_merged_df['Week'] = final_merged_df['date'].dt.week
```

In [38]:
```python
final_merged_df.head()
```

Out[38]:

	store	date	temperature	fuel_price	markdown1	markdown2
date						
2010-02-05	1	2010-02-05	42.31	2.572	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0

In [39]:

```
#Outlier Detection and Abnormalities
agg_data = final_merged_df.groupby(['store', 'dept']).weekly_sales.agg(['max',
agg_data.head()
```

Out[39]:

	store	dept	max	min	mean	median	std
0	1	1	57592.12	14537.37	22513.322937	18535.48	9854.349032
1	1	2	65615.36	35819.83	46102.090420	45561.85	3440.673222
2	1	3	51159.17	6165.73	13150.478042	10366.85	8708.978853
3	1	4	47893.23	32497.43	36964.154476	36579.96	2930.698313
4	1	5	85676.09	11570.27	24257.941119	21183.42	11330.286495

In [40]:

```
agg_data.isnull().sum()
```

Out[40]:

```
store      0
dept       0
max        0
min        0
mean       0
median     0
std       37
dtype: int64
```

In [41]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

final_merged_df['IsHoliday'] = le.fit_transform(final_merged_df['IsHoliday'])
final_merged_df['type'] = le.fit_transform(final_merged_df['type'])
```

In [42]:

```
final_merged_df.head()
```

Out[42]:

	store	date	temperature	fuel_price	markdown1	markdown2
date						
2010-02-05	1	2010-02-05	42.31	2.572	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0
2010-02-05	35	2010-02-05	27.19	2.784	0.0	0.0

In [43]:

```
final_merged_df['dept'].nunique()
```

Out[43]: 81

Note:

Store numbers begin from 1 to 45, department numbers are from 1 to 99, but some department numbers are missing such as there is no 88 or 89 etc. Total number of departments is 81.

In [44]:

```
final_merged_df['IsHoliday'].nunique()
```

Out[44]: 2

In [45]:

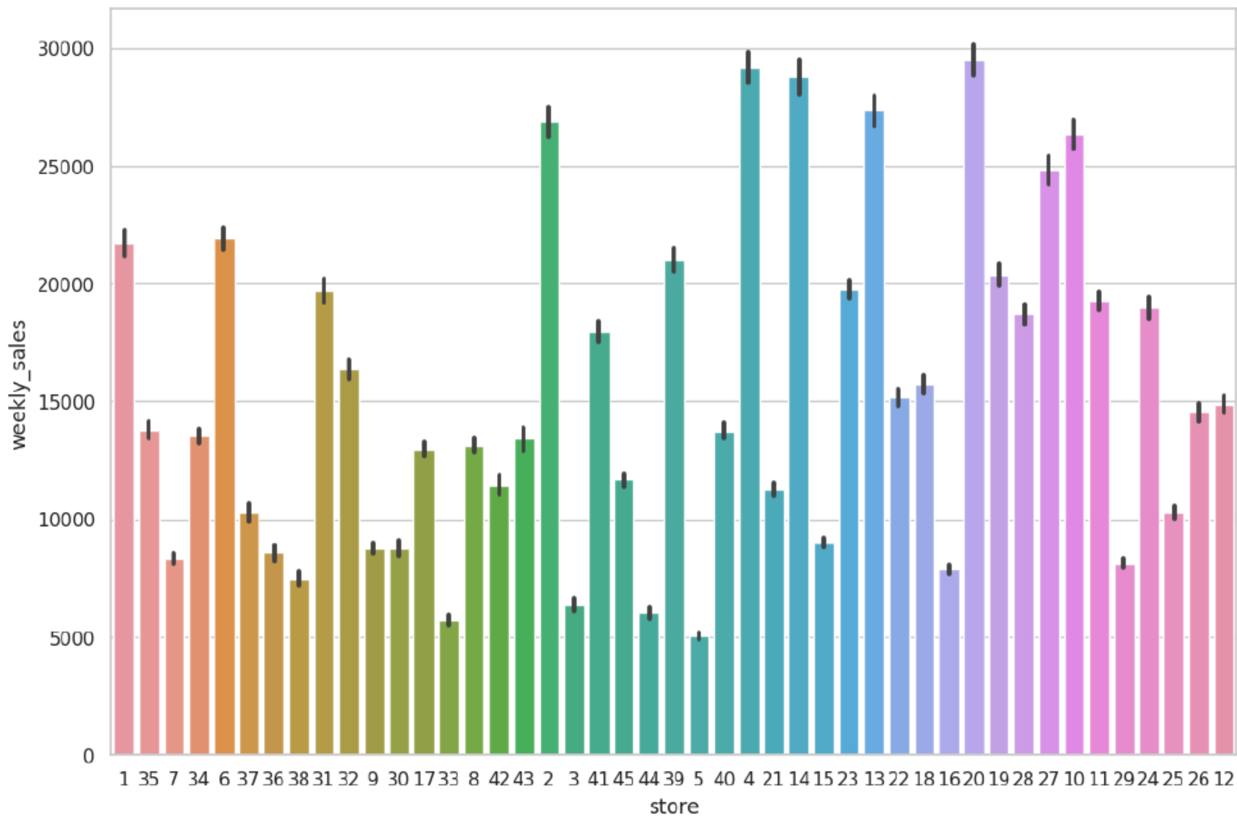
```
final_merged_df['type'].nunique()
```

Out[45]: 3

Plotting Stores by Weekly Sales

In [46]:

```
plt.figure(figsize=(12,8))
sns.barplot(x='store',y='weekly_sales',data=final_merged_df)
plt.show()
```



In [47]:

```
# <= Ranking stores by weekly sales (combined for entire dates) =>

# Ensuring 'store' is of type integer for sorting
final_merged_df['store'] = final_merged_df['store'].astype(int)

# Aggregating weekly sales by stores
total_sales = final_merged_df.groupby('store')['weekly_sales'].sum().reset_inc

# Sorting stores by total weekly sales in descending order
total_sales = total_sales.sort_values(by='weekly_sales', ascending=False)

print(total_sales)
```

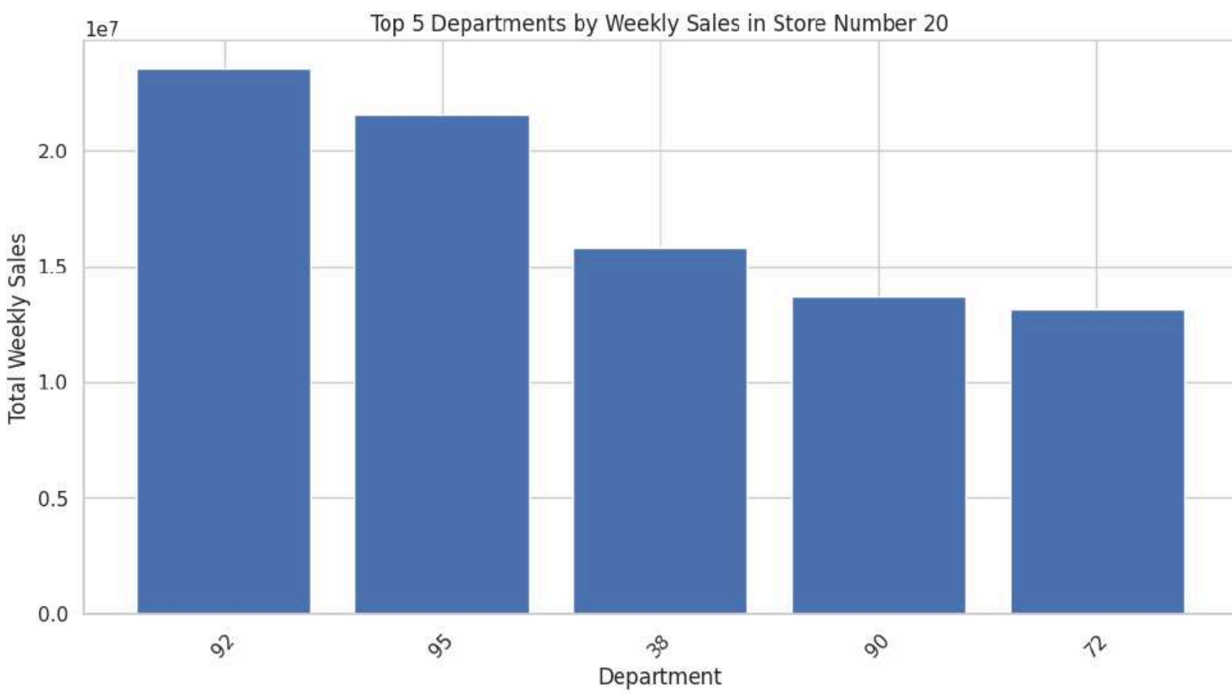
```
    store  weekly_sales
19     20  3.013978e+08
3       4  2.995440e+08
13     14  2.889999e+08
12     13  2.865177e+08
1       2  2.753824e+08
9      10  2.716177e+08
26     27  2.538559e+08
5       6  2.237561e+08
0       1  2.224028e+08
38     39  2.074455e+08
18     19  2.066349e+08
30     31  1.996139e+08
22     23  1.987506e+08
23     24  1.940160e+08
10     11  1.939628e+08
27     28  1.892637e+08
40     41  1.813419e+08
31     32  1.668192e+08
17     18  1.551147e+08
21     22  1.470756e+08
11     12  1.442872e+08
25     26  1.434164e+08
33     34  1.382498e+08
39     40  1.378703e+08
34     35  1.315207e+08
7       8  1.299512e+08
16     17  1.277821e+08
44     45  1.123953e+08
20     21  1.081179e+08
24     25  1.010612e+08
42     43  9.056544e+07
14     15  8.913368e+07
6       7  8.159828e+07
41     42  7.956575e+07
8       9  7.778922e+07
28     29  7.714155e+07
15     16  7.425243e+07
36     37  7.420274e+07
29     30  6.271689e+07
2       3  5.758674e+07
37     38  5.515963e+07
35     36  5.341221e+07
4       5  4.547569e+07
43     44  4.329309e+07
32     33  3.716022e+07
```

Note:

- Stores 20, 4, 14, 13, 2 are the top 5 stores by weekly sales
- Stores 38, 36, 5, 44, 33 are the bottom 5 stores by weekly sales

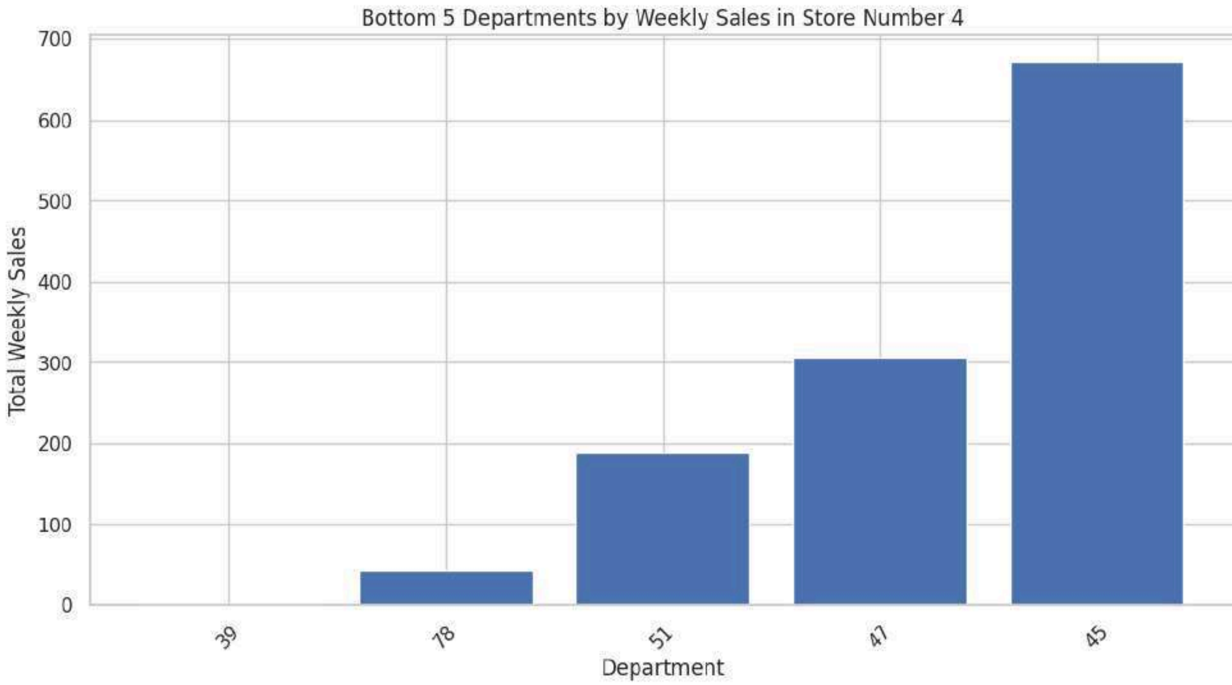
In [48]:
```
# <= Top 5 departments by weekly sales of Store 20 =>
```

```python
# Filtering the DataFrame for store number 20
store_20_df = final_merged_df[final_merged_df['store'] == 20]

# Aggregating weekly sales by department
department_sales = store_20_df.groupby('dept')['weekly_sales'].sum().reset_inc

# Sorting departments by total sales and get the top 5 departments
top_departments = department_sales.sort_values(by='weekly_sales', ascending=Fa

# Plotting
plt.figure(figsize=(12, 6))
plt.bar(top_departments['dept'].astype(str), top_departments['weekly_sales'])
plt.xlabel('Department')
plt.ylabel('Total Weekly Sales')
plt.title('Top 5 Departments by Weekly Sales in Store Number 20')
plt.xticks(rotation=45)
plt.show()
```



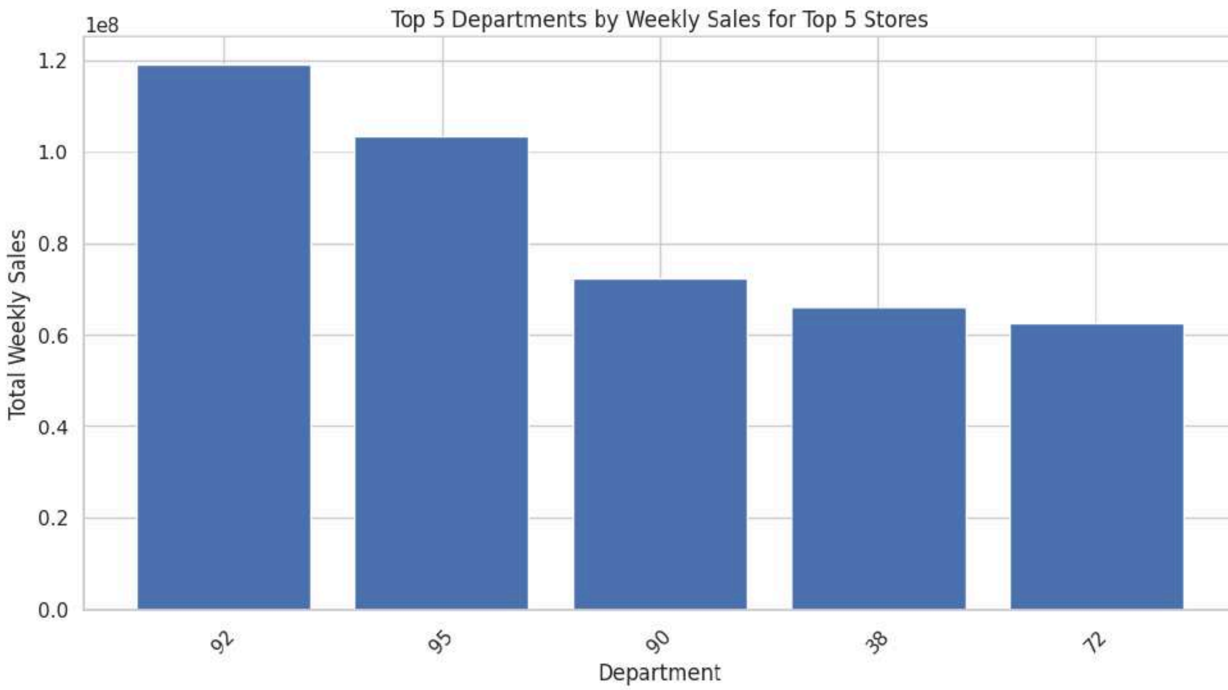
In [49]:
```python
# <= Bottom 5 departments by weekly sales of Store 4 =>

# Filtering the DataFrame for store number 4
store_4_df = final_merged_df[final_merged_df['store'] == 4]

# Aggregating weekly sales by department
department_sales = store_4_df.groupby('dept')['weekly_sales'].sum().reset_inde

# Sorting departments by total sales and get the bottom 5
btm_departments = department_sales.sort_values(by='weekly_sales', ascending=Tr

# Plotting
plt.figure(figsize=(12, 6))
```

```
plt.bar(btm_departments['dept'].astype(str), btm_departments['weekly_sales'])
plt.xlabel('Department')
plt.ylabel('Total Weekly Sales')
plt.title('Bottom 5 Departments by Weekly Sales in Store Number 4')
plt.xticks(rotation=45)
plt.show()
```



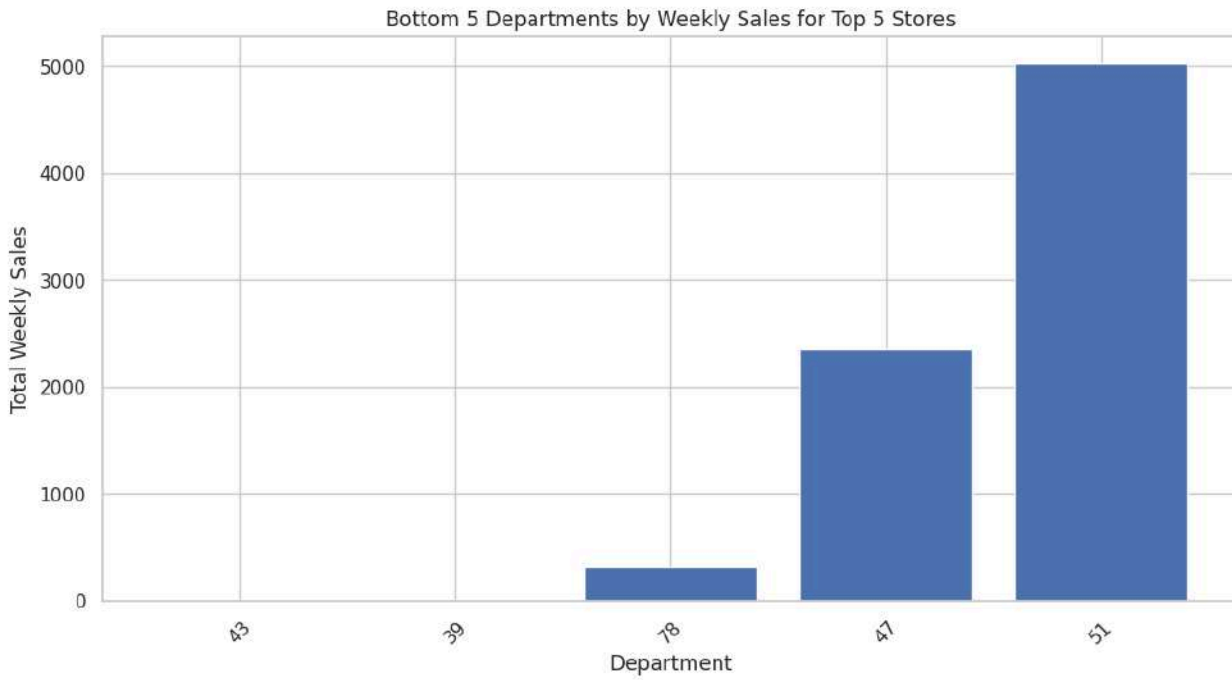
In [50]:

```
# <= Top 5 departments of Top 5 stores by weekly sales (Combined for entire du

# Define the top 5 stores
top_stores = [20, 4, 14, 13, 2]

# Filter the DataFrame for the top 5 stores
top_stores_df = final_merged_df[final_merged_df['store'].isin(top_stores)]

# Aggregating weekly sales by department
department_sales = top_stores_df.groupby('dept')['weekly_sales'].sum().reset_i

# Sorting departments by total sales and get the top 5
top_departments = department_sales.sort_values(by='weekly_sales', ascending=Fa

# Plotting
plt.figure(figsize=(12, 6))
plt.bar(top_departments['dept'].astype(str), top_departments['weekly_sales'])
plt.xlabel('Department')
plt.ylabel('Total Weekly Sales')
plt.title('Top 5 Departments by Weekly Sales for Top 5 Stores')
plt.xticks(rotation=45)
plt.show()
```



In [51]:

```
# <= Bottom 5 departments of Top 5 stores by weekly sales (Combined for entire

# Define the top 5 stores
top_stores = [20, 4, 14, 13, 2]

# Filter the DataFrame for the top 5 stores
top_stores_df = final_merged_df[final_merged_df['store'].isin(top_stores)]

# Aggregating weekly sales by department
department_sales = top_stores_df.groupby('dept')['weekly_sales'].sum().reset_i

# Sorting departments by total sales and get the bottom 5 department
btm_departments = department_sales.sort_values(by='weekly_sales', ascending=Tr

# Plotting
plt.figure(figsize=(12, 6))
plt.bar(btm_departments['dept'].astype(str), btm_departments['weekly_sales'])
plt.xlabel('Department')
plt.ylabel('Total Weekly Sales')
plt.title('Bottom 5 Departments by Weekly Sales for Top 5 Stores')
plt.xticks(rotation=45)
plt.show()
```



In [52]:
```
data = pd.read_csv('/kaggle/input/merged-data/merged-data.csv')
```

In [53]:
```
data.head()
```

Out[53]:

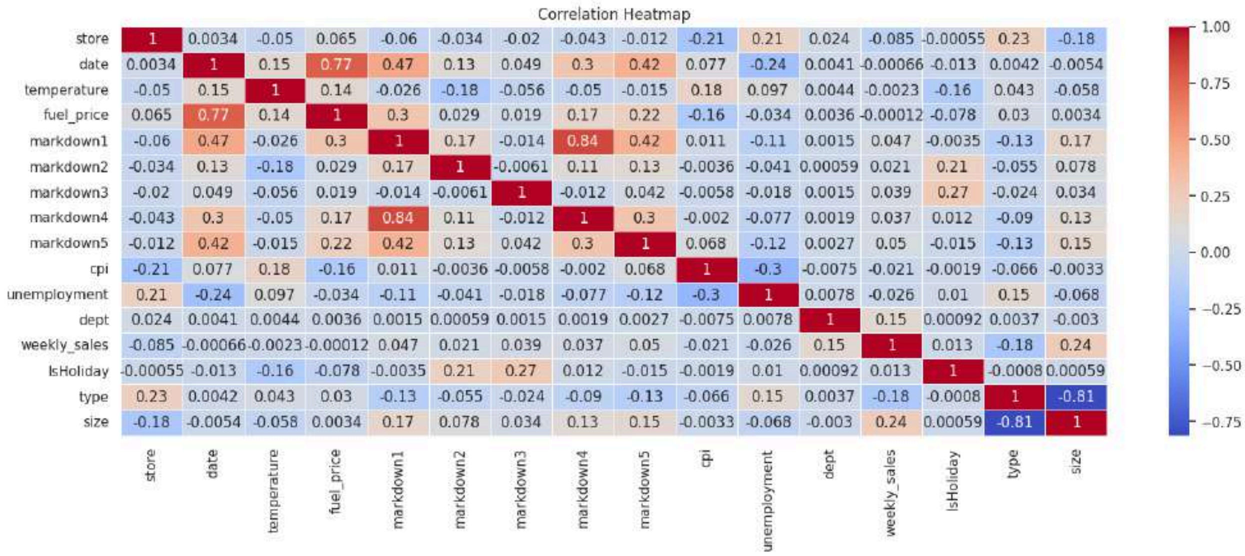
	store	date	temperature	fuel_price	markdown1	markdown2	markdow
0	20	2010-02-05	25.92	2.784	0.0	0.0	
1	30	2010-02-05	39.05	2.572	0.0	0.0	
2	30	2010-02-05	39.05	2.572	0.0	0.0	
3	30	2010-02-05	39.05	2.572	0.0	0.0	
4	30	2010-02-05	39.05	2.572	0.0	0.0	

In [54]:
```
# Convert the date column to datetime format if it's not already
data['date'] = pd.to_datetime(data['date'])

# Group by date and sum weekly sales
sales_over_time = data.groupby('date')['weekly_sales'].sum()
```

In [55]:
```
import plotly.graph_objects as go
import plotly.offline as po
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
pd.options.plotting.backend= "plotly"
```

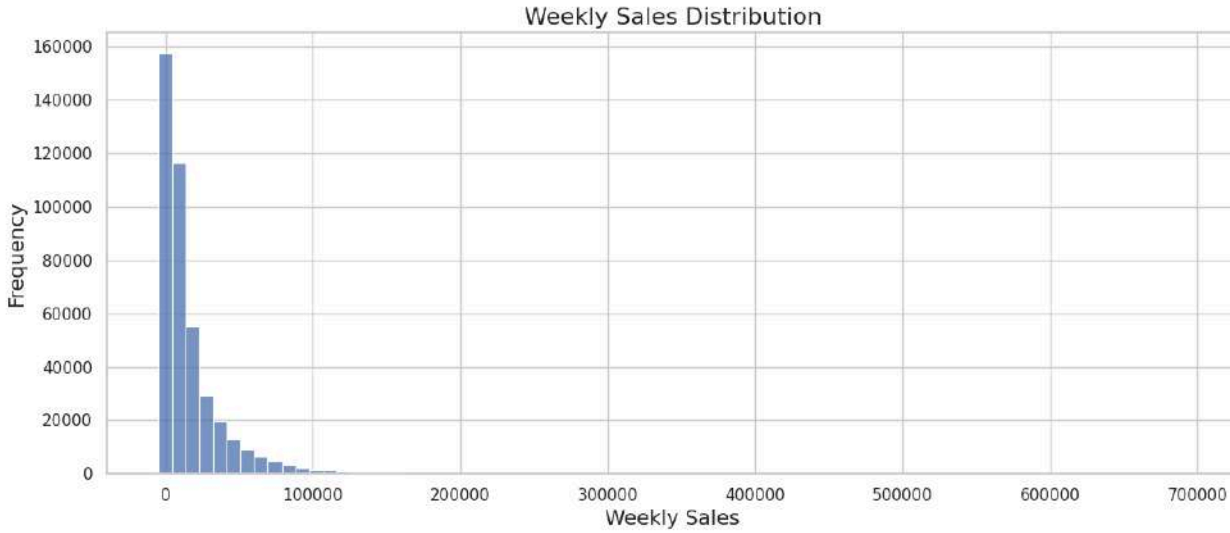
In [56]:
```
correlation_matrix = data.corr()

# Plot the heatmap
plt.figure(figsize=(18, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
```

```
plt.title('Correlation Heatmap')
plt.show()
```



In [57]:
```
# Set the size of the plot
plt.figure(figsize=(12, 6))

# Create the histogram with 75 bins
sns.histplot(data['weekly_sales'], bins=75, kde=False)

# Set the title and labels
plt.title('Weekly Sales Distribution', fontsize=16)
plt.xlabel('Weekly Sales', fontsize=14)
plt.ylabel('Frequency', fontsize=14)

# Set margins
plt.gcf().subplots_adjust(left=0.05, right=0.95, bottom=0.15, top=0.85)

# Show the plot
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarnin
g:

use_inf_as_na option is deprecated and will be removed in a future version. Con
vert inf values to NaN before operating instead.
```



In [58]:

```
# average of weekly sales by department (sorted)
avg_sales_dept = data.groupby("dept")["weekly_sales"].mean().sort_values()

fig = go.Figure()

fig.add_trace(go.Bar(y=avg_sales_dept.values,
                    text=avg_sales_dept.index,
                     textposition='outside')
             )


fig.update_traces(marker_color='darkcyan',
                  marker_line_color='darkcyan',
                  marker_line_width=1.5,
                 )

fig.update_layout(title_text='Average Weekly Sales by Department',
                  xaxis_title_text="Department Number",   # xaxis label
                  xaxis_showticklabels=False,
                  yaxis_title_text="Average Sales") # yaxis label)


fig.show()
```

In [59]:

```
# average of weekly sales by store (sorted)
avg_sales_store = data.groupby("store")["weekly_sales"].mean().sort_values()
fig = go.Figure()

fig.add_trace(go.Bar(y=avg_sales_store.values,
                    text=avg_sales_store.index,
                     textposition='outside'))


fig.update_traces(marker_color='rosybrown',
                  marker_line_color='darkmagenta',
                  marker_line_width=1.5,)

fig.update_layout(title_text='Average Weekly Sales by Store',
                  xaxis=dict(
                      title_text="Store Number",
                      showticklabels=False,
                      ticks='outside',
                      tickfont=dict(
                          family='Arial',
                          size=12,
```

```
                          color='rgb(82, 82, 82)')),
                       yaxis=dict(
                             title_text="Average Sales"))

fig.show()
```

In [60]:

```
# checking out the sum of sales by store type
fig = go.Figure()

fig.add_trace(go.Histogram(x=data["type"],
                          histnorm="percent",
                           xbins=dict(size=0.5),marker_color='#EB89B5',opacity

fig.update_layout(
    height=500, width=400,
    title_text="Store Types in %",
    xaxis_title_text="Type",
    yaxis_title_text="Ratio",
)

fig.show()
```

Note: Some department numbers are missing

- Top 5 Departments:

1. 92: Dry Grocery
2. 95: Grocery, Snacks, and Beverages
3. 90: Dairy
4. 38: Prescription Pharmacy
5. 72: Electronics

- Bottom 5 Departments:

1. 43: Toys (Unsual)
2. 39: Misc/Jewellery
3. 78: Ladieswear (Part of Fashion)
4. 47: Something in Fashion
5. 51: Sporting Goods

Modelling

In [61]:

```
data = pd.read_csv('/kaggle/input/merged-data/merged-data.csv')
```

In [62]:

```
# Converting the 'date' column to datetime format
data['date'] = pd.to_datetime(data['date'])

# Extracting year, month, and week from the date
data['year'] = data['date'].dt.year
data['month'] = data['date'].dt.month
data['week'] = data['date'].dt.isocalendar().week

# Checking for missing values
missing_values = data.isnull().sum()

# Displaying the updated dataset with new features and missing values
data.head(), missing_values
```

Out[62]:
```
(   store       date  temperature  fuel_price  markdown1  markdown2  markdown
3  \
 0     20 2010-02-05        25.92       2.784        0.0        0.0
0.0
 1     30 2010-02-05        39.05       2.572        0.0        0.0
0.0
 2     30 2010-02-05        39.05       2.572        0.0        0.0
0.0
 3     30 2010-02-05        39.05       2.572        0.0        0.0
0.0
 4     30 2010-02-05        39.05       2.572        0.0        0.0
0.0

    markdown4  markdown5         cpi  unemployment  dept  weekly_sales  \
 0        0.0        0.0  204.247194          8.19     1      46021.21
 1        0.0        0.0  210.752605          8.32    17        198.01
 2        0.0        0.0  210.752605          8.32    16        974.31
 3        0.0        0.0  210.752605          8.32    14       1134.75
 4        0.0        0.0  210.752605          8.32    13      12059.20

    IsHoliday  type    size  year  month  week
 0          0     0  203742  2010      2     5
 1          0     2   42988  2010      2     5
 2          0     2   42988  2010      2     5
 3          0     2   42988  2010      2     5
 4          0     2   42988  2010      2     5  ,
 store           0
 date            0
 temperature     0
 fuel_price      0
 markdown1       0
 markdown2       0
 markdown3       0
 markdown4       0
 markdown5       0
 cpi             0
 unemployment    0
 dept            0
 weekly_sales    0
 IsHoliday       0
 type            0
 size            0
 year            0
 month           0
 week            0
 dtype: int64)
```

Linear Models

In [63]:
```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```python
# Select the features for the model (excluding the target variable 'weekly_sal
features = data.drop(columns=['weekly_sales', 'date'])

# Target variable
target = data['weekly_sales']
```

In [64]:
```python
features
```

Out[64]:

	store	temperature	fuel_price	markdown1	markdown2	markdown3
0	20	25.92	2.784	0.00	0.00	0.00
1	30	39.05	2.572	0.00	0.00	0.00
2	30	39.05	2.572	0.00	0.00	0.00
3	30	39.05	2.572	0.00	0.00	0.00
4	30	39.05	2.572	0.00	0.00	0.00
...	...	...	...	...	...	...
421565	38	65.95	4.301	148.32	6.73	3.05
421566	38	65.95	4.301	148.32	6.73	3.05
421567	38	65.95	4.301	148.32	6.73	3.05
421568	28	65.95	4.301	6490.13	90.02	0.00
421569	9	69.52	3.506	512.23	3.00	8.00

421570 rows × 17 columns

In [65]:
```python
target
```

Out[65]:
```
0         46021.21
1           198.01
2           974.31
3          1134.75
4         12059.20
            ...
421565    41940.71
421566    22348.91
421567    18739.49
421568     8846.10
421569     4752.25
Name: weekly_sales, Length: 421570, dtype: float64
```

In [66]:
```python
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_siz

# Initialize and train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[66]:

```
▾ LinearRegression
LinearRegression()
```

In [67]:

```
# Making predictions on test set
y_pred = model.predict(X_test)
```

In [68]:

```
# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(mse)
print(r2)
```

```
470345208.5756179
0.0878091580181749
```

Note: As anticipated, Linear Regression does not yield good results because the data does not exhibit a linear relationship.

Lasso & Ridge Regression

In [69]:

```
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.linear_model import Ridge, Lasso
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor

# Feature Engineering: Adding interaction terms using PolynomialFeatures
poly = PolynomialFeatures(degree=2, interaction_only=True, include_bias=False)
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.transform(X_test)

# Apply scaling to the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train_poly)
X_test_scaled = scaler.transform(X_test_poly)

# Regularization: Ridge Regression
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train_scaled, y_train)
ridge_pred = ridge_model.predict(X_test_scaled)
ridge_mse = mean_squared_error(y_test, ridge_pred)
ridge_r2 = r2_score(y_test, ridge_pred)
```

In [70]:

```
print(ridge_r2 * 100)
```

```
11.581299737717154
```

In [71]:

```
# Regularization: Lasso Regression
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train_scaled, y_train)
```

```
lasso_pred = lasso_model.predict(X_test_scaled)
lasso_mse = mean_squared_error(y_test, lasso_pred)
lasso_r2 = r2_score(y_test, lasso_pred)
```

```
/opt/conda/lib/python3.10/site-packages/sklearn/linear_model/_coordinate_descen
t.py:631: ConvergenceWarning:

Objective did not converge. You might want to increase the number of iteration
s, check the scale of the features or consider increasing regularisation. Duali
ty gap: 7.245e+13, tolerance: 1.740e+10
```

In [72]:
```
print(lasso_r2 * 100)
```

```
11.536166550640758
```

Note: Ridge and Lasso models also do not produce improved results.

Tree Models

Model: Random Forest Regression (n_est = 50)

In [73]:
```
import time
```

In [74]:
```
# Start time
start_time = time.time()
rf_model_1 = RandomForestRegressor(n_estimators=50, random_state=42)
rf_model_1.fit(X_train, y_train)
rf_pred_1 = rf_model_1.predict(X_test)
rf_mse_1 = mean_squared_error(y_test, rf_pred_1)
rf_r2_1 = r2_score(y_test, rf_pred_1)
# End time
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", rf_r2_1)
```

```
Execution Time:  3.002178879578908
R^2 VALUE:  0.9746693932597383
```

Model: Random Forest Regression (n_est = 50, max_depth = 100)

In [75]:
```
# Model: Random Forest Regressor (n_estimators=50, max_depth=100)
# Start time
start_time = time.time()
rf_model_1_100 = RandomForestRegressor(n_estimators=50, max_depth=100, random_
rf_model_1_100.fit(X_train, y_train)
```

```
rf_pred_1_100  = rf_model_1_100.predict(X_test)
rf_mse_1_100 = mean_squared_error(y_test, rf_pred_1_100)
rf_r2_1_100 = r2_score(y_test, rf_pred_1_100)
# End time
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", rf_r2_1_100)
```

```
Execution Time:  3.0451568086942036
R^2 VALUE:  0.9746693932597383
```

Model: Random Forest Regression (n_est = 100)

In [76]:

```
# Model: Random Forest Regressor (n_estimators=100)
# Start time
start_time = time.time()
rf_model_2 = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model_2.fit(X_train, y_train)
rf_pred_2 = rf_model_2.predict(X_test)
rf_mse_2 = mean_squared_error(y_test, rf_pred_2)
rf_r2_2 = r2_score(y_test, rf_pred_2)
# End time
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", rf_r2_2)
```

```
Execution Time:  6.124047088623047
R^2 VALUE:  0.9752805326962908
```

Model: Random Forest Regression (n_est = 100, max_depth = 100)

In [77]:

```
# Model: Random Forest Regressor (n_estimators=100)
# Start time
start_time = time.time()
rf_model_2_100 = RandomForestRegressor(n_estimators=100, max_depth=100, random
rf_model_2_100.fit(X_train, y_train)
rf_pred_2_100 = rf_model_2_100.predict(X_test)
rf_mse_2_100 = mean_squared_error(y_test, rf_pred_2_100)
rf_r2_2_100 = r2_score(y_test, rf_pred_2_100)
# End time
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
```

```
print("R^2 VALUE: ", rf_r2_2_100)
```

```
Execution Time:  6.061469042301178
R^2 VALUE:  0.9752805326962908
```

Gradient Boosting (n_est = 50)

In [78]:
```
# Model: Gradient Boosting Regressor
start_time = time.time()
gb_model_1 = GradientBoostingRegressor(n_estimators=50, random_state=42)
gb_model_1.fit(X_train, y_train)
gb_pred_1 = gb_model_1.predict(X_test)
gb_mse_1 = mean_squared_error(y_test, gb_pred_1)
gb_r2_1 = r2_score(y_test, gb_pred_1)
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_1)
```

```
Execution Time:  0.6088111360867818
R^2 VALUE:  0.6520888500728567
```

Gradient Boosting (n_est = 100)

In [79]:
```
# Model: Gradient Boosting Regressor
start_time = time.time()
gb_model = GradientBoostingRegressor(n_estimators=100, random_state=42)
gb_model.fit(X_train, y_train)
gb_pred_2 = gb_model.predict(X_test)
gb_mse_2 = mean_squared_error(y_test, gb_pred_2)
gb_r2_2 = r2_score(y_test, gb_pred_2)
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_2)
```

```
Execution Time:  1.1966362516085307
R^2 VALUE:  0.7437162086119167
```

Gradient Boosting (n_est = 200)

In [80]:
```
# Model: Gradient Boosting Regressor
start_time = time.time()
gb_model = GradientBoostingRegressor(n_estimators=200, random_state=42)
gb_model.fit(X_train, y_train)
gb_pred_3 = gb_model.predict(X_test)
gb_mse_3 = mean_squared_error(y_test, gb_pred_3)
gb_r2_3 = r2_score(y_test, gb_pred_3)
```

```
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_3)
```

```
Execution Time:  2.4496328512827557
R^2 VALUE:  0.8027364728597444
```

XGB Regressor

In [81]:
```
from xgboost import XGBRegressor
xgbr = XGBRegressor(n_estimators = 50)
xgbr.fit(X_train, y_train)
xgb_pred = xgbr.predict(X_test)
xgb_mse = mean_squared_error(y_test, xgb_pred)
xgb_r2 = r2_score(y_test, xgb_pred)
print("R^2 VALUE: ", xgb_r2)
```

```
R^2 VALUE:  0.922024866550321
```

In [82]:
```
xgboost_accuracy = xgbr.score(X_test,y_test)*100
print("XGBoost Regressor Accuracy - ",xgboost_accuracy)
```

```
XGBoost Regressor Accuracy -  92.2024866550321
```

In [83]:
```
from xgboost import XGBRegressor
xgbr1 = XGBRegressor(n_estimators = 100)
xgbr1.fit(X_train, y_train)
xgb_pred_1 = xgbr.predict(X_test)
xgb_mse_1 = mean_squared_error(y_test, xgb_pred_1)
xgb_r2_1 = r2_score(y_test, xgb_pred_1)
print("R^2 VALUE: ", xgb_r2_1)
```

```
R^2 VALUE:  0.922024866550321
```

In [84]:
```
xgboost_accuracy = xgbr1.score(X_test,y_test)*100
print("XGBoost Regressor Accuracy - ",xgboost_accuracy)
```

```
XGBoost Regressor Accuracy -  94.32889703470117
```

In [85]:
```
xgbr2 = XGBRegressor(n_estimators=200)
xgbr2.fit(X_train, y_train)
xgb_pred_2 = xgbr2.predict(X_test)
xgb_mse_2 = mean_squared_error(y_test, xgb_pred_2)
xgb_r2_2 = r2_score(y_test, xgb_pred_2)
print("R^2 VALUE: ", xgb_r2_2)
```

```
R^2 VALUE:  0.9588031656895174
```

In [86]:
```
xgboost_accuracy = xgbr2.score(X_test,y_test)*100
print("XGBoost Regressor Accuracy - ",xgboost_accuracy)
```

```
XGBoost Regressor Accuracy -  95.88031656895174
```

Compiled Results

In [87]:
```
# Compile the results with MSE and R^2
results = {
    "Model": ["Ridge Regression", "Lasso Regression", "Random Forest", "Random
    "MSE": [ridge_mse, lasso_mse, rf_mse_1, rf_mse_1_100, rf_mse_2, rf_mse_2_1
    "R² Score": [ridge_r2, lasso_r2, rf_r2_1, rf_r2_1_100, rf_r2_2, rf_r2_2_10
}

results_df = pd.DataFrame(results)
results_df
```

Out[87]:

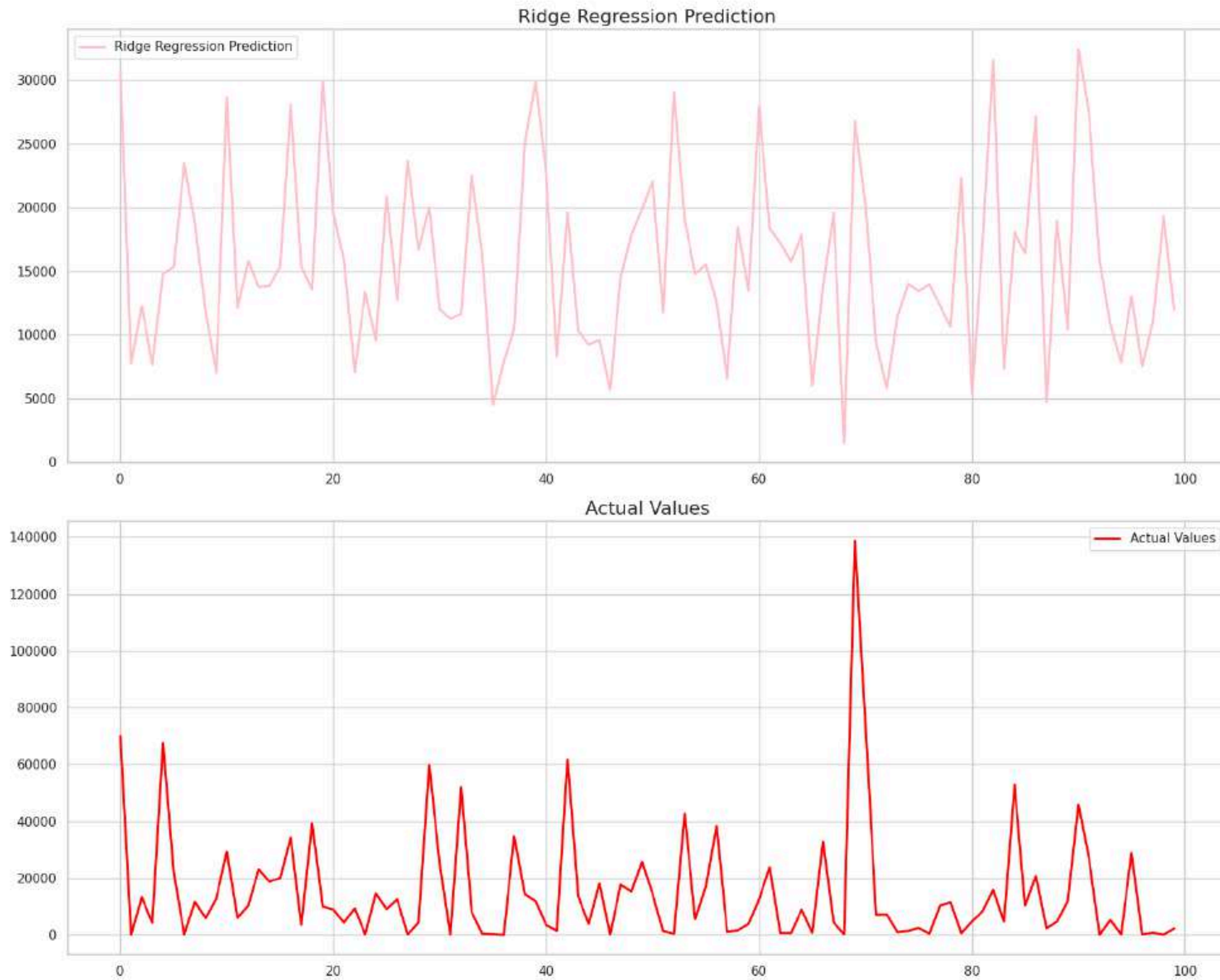
	Model	MSE	R² Score
0	Ridge Regression	4.559058e+08	0.115813
1	Lasso Regression	4.561385e+08	0.115362
2	Random Forest	1.306101e+07	0.974669
3	Random Forest	1.306101e+07	0.974669
4	Random Forest	1.274589e+07	0.975281
5	Random Forest	1.274589e+07	0.975281
6	Gradient Boosting	1.793905e+08	0.652089
7	Gradient Boosting	1.321454e+08	0.743716
8	Gradient Boosting	1.017133e+08	0.802736
9	XGB Regressor	4.020566e+07	0.922025
10	XGB Regressor	4.020566e+07	0.922025
11	XGB Regressor	2.124197e+07	0.958803

Model Train v/s Test

Ridge Regression

In [88]:
```
# Select a subset of the test data for visualization
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values

# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))

# First subplot: Ridge Regression predictions
ax[0].plot(ridge_pred[:n], label="Ridge Regression Prediction", linewidth=2.0,
ax[0].set_title('Ridge Regression Prediction', fontsize=16)
```

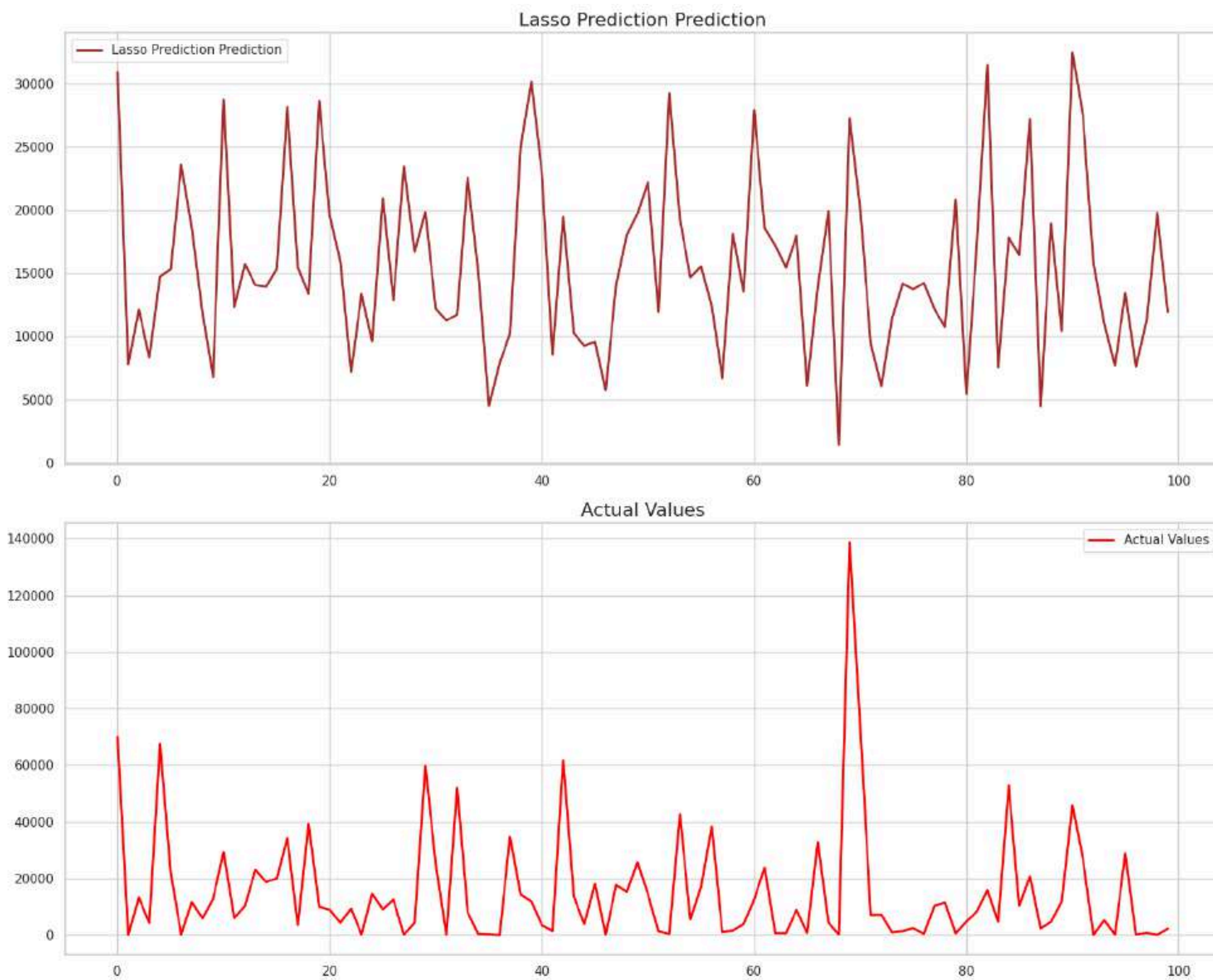
```
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
plt.tight_layout()

# Show the plot
plt.show()
```



Lasso Regression

In [89]:
```
# Select a subset of the test data for visualization
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values

# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))
```

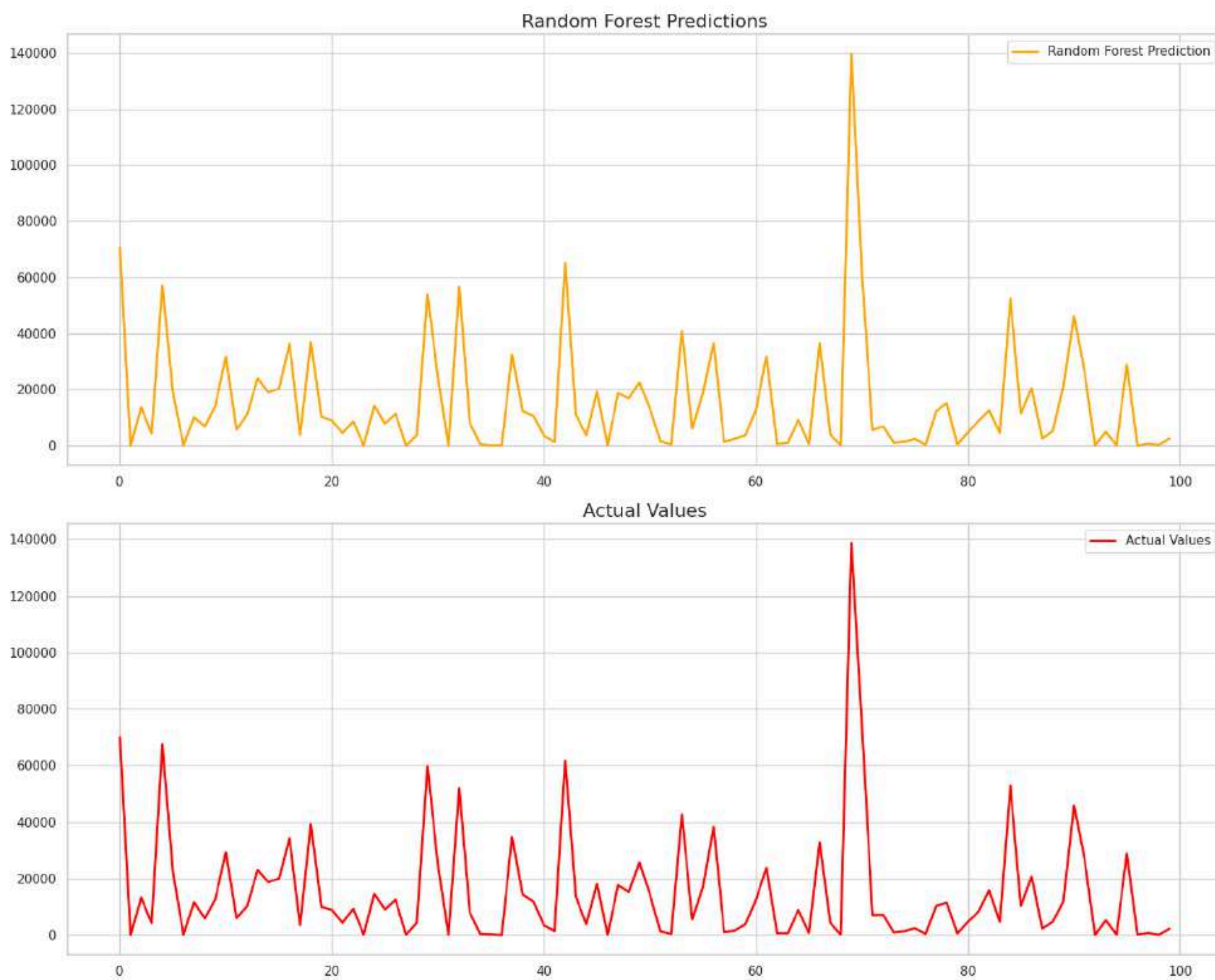
```
# First subplot: Lasso Prediction predictions
ax[0].plot(lasso_pred[:n], label="Lasso Prediction Prediction", linewidth=2.0,
ax[0].set_title('Lasso Prediction Prediction', fontsize=16)
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
plt.tight_layout()

# Show the plot
plt.show()
```



Random Forest Regressor

In [90]:
```
# Select a subset of the test data for visualization
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values
```

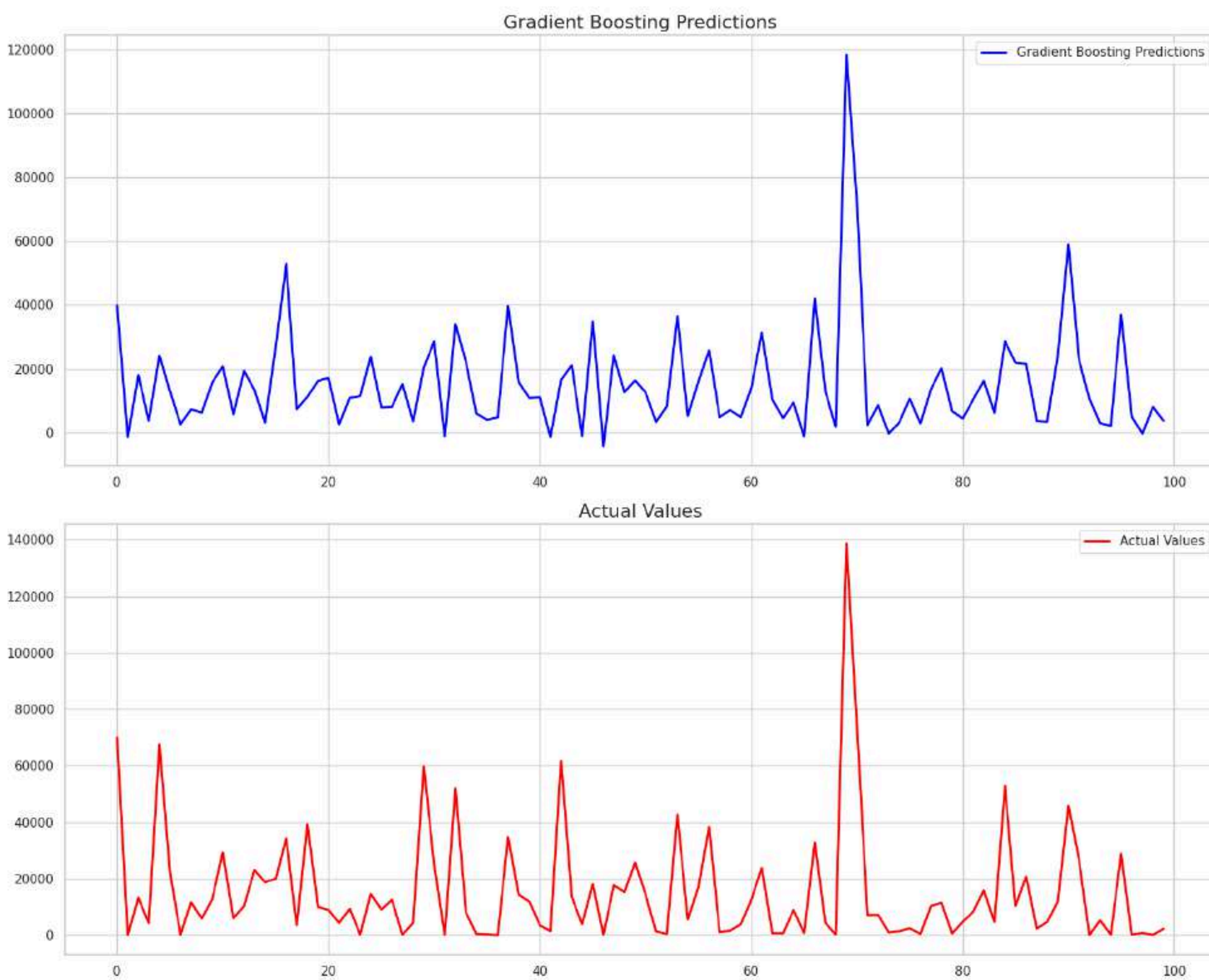
```
# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))

# First subplot: Random Forest predictions
ax[0].plot(rf_pred_2_100[:n], label="Random Forest Prediction", linewidth=2.0,
ax[0].set_title('Random Forest Predictions', fontsize=16)
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
plt.tight_layout()

# Show the plot
plt.show()
```



Gradient Boosting

In [91]:
```
# Select a subset of the test data for visualization
```

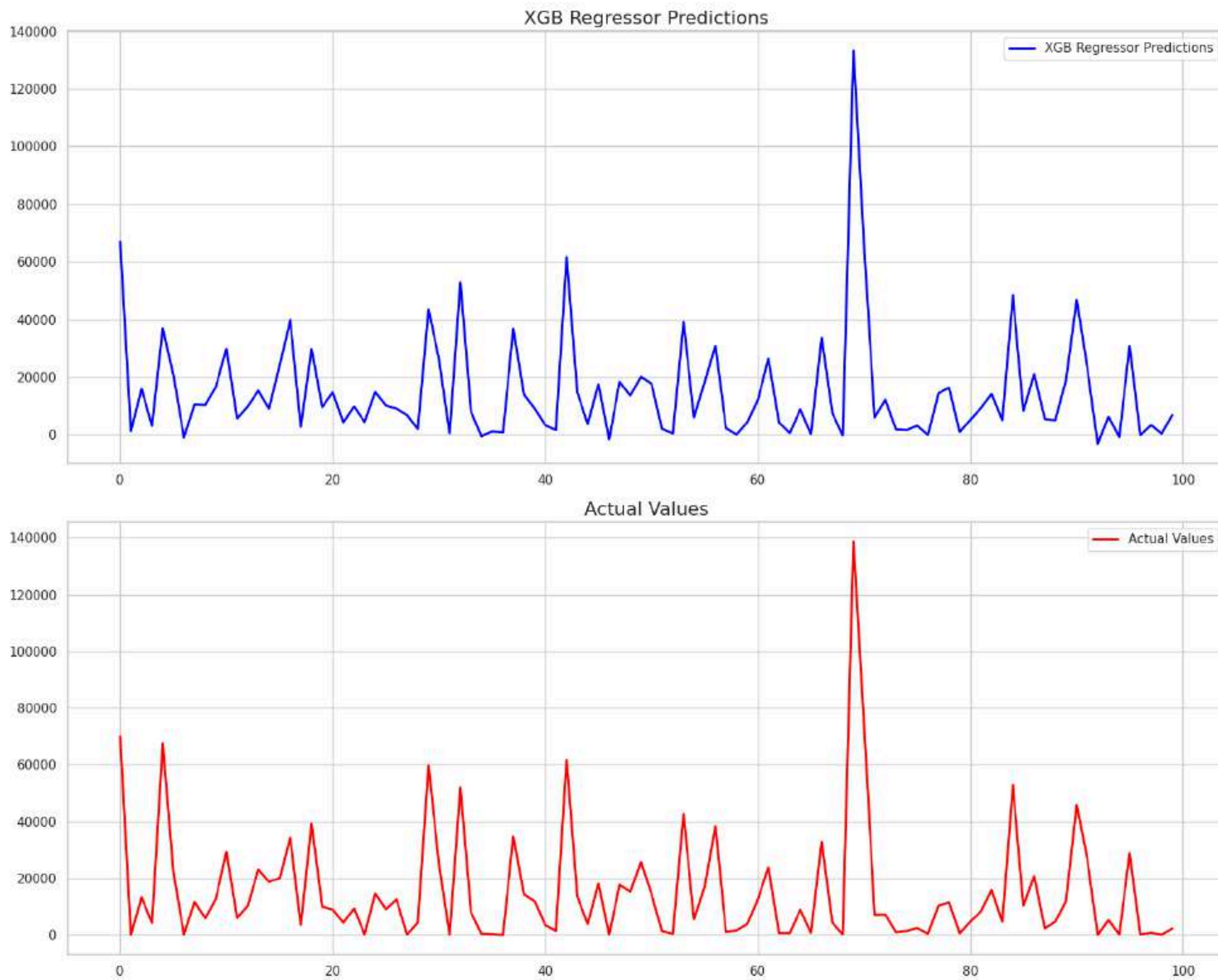
```python
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values

# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))

# First subplot: Gradient Boosting Predictions
ax[0].plot(gb_pred_3[:n], label="Gradient Boosting Predictions", linewidth=2.0
ax[0].set_title('Gradient Boosting Predictions', fontsize=16)
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
plt.tight_layout()

# Show the plot
plt.show()
```



XGB Regressor

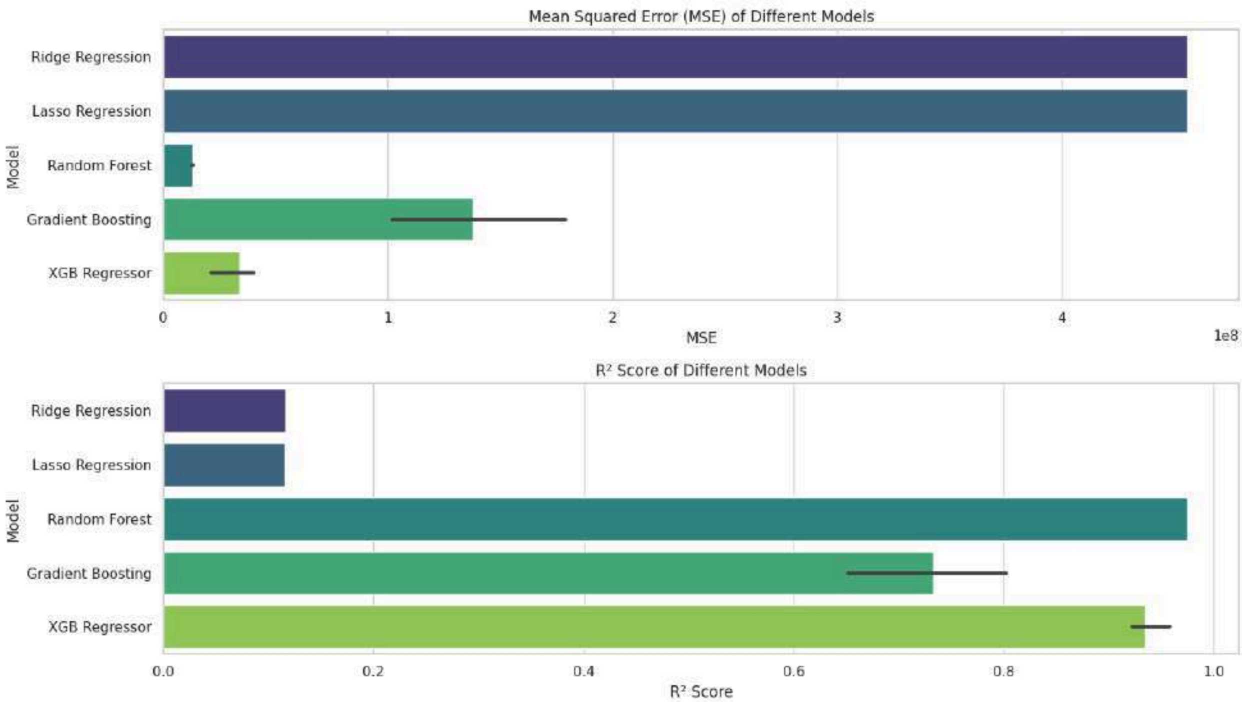
In [92]:

```
# Select a subset of the test data for visualization
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values

# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))

# First subplot: XGB Regressor Predictions
ax[0].plot(xgb_pred_2[:n], label="XGB Regressor Predictions", linewidth=2.0, c
ax[0].set_title('XGB Regressor Predictions', fontsize=16)
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
plt.tight_layout()

# Show the plot
plt.show()
```



In [93]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

results = {
    "Model": ["Ridge Regression", "Lasso Regression", "Random Forest", "Random
    "MSE": [ridge_mse, lasso_mse, rf_mse_1, rf_mse_1_100, rf_mse_2, rf_mse_2_1
    "R² Score": [ridge_r2, lasso_r2, rf_r2_1, rf_r2_1_100, rf_r2_2, rf_r2_2_10
}

# Convert the results dictionary to a DataFrame
results_df = pd.DataFrame(results)

# Set the style and size of the plots
sns.set(style="whitegrid")
plt.figure(figsize=(14, 8))

# Create a bar plot for MSE
plt.subplot(2, 1, 1)
sns.barplot(x="MSE", y="Model", data=results_df, palette="viridis")
plt.title('Mean Squared Error (MSE) of Different Models')
plt.xlabel('MSE')
plt.ylabel('Model')
```

```
# Create a bar plot for R² Score
plt.subplot(2, 1, 2)
sns.barplot(x="R² Score", y="Model", data=results_df, palette="viridis")
plt.title('R² Score of Different Models')
plt.xlabel('R² Score')
plt.ylabel('Model')

# Adjust the layout to prevent overlap
plt.tight_layout()

# Show the plots
plt.show()
```



Note: XGBoost model performs significantly better as it leverages multidimensional data.

Feature Engineering

In [94]:
```
!pip install shap
```

```
Requirement already satisfied: shap in /opt/conda/lib/python3.10/site-packages
(0.44.1)
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages
(from shap) (1.26.4)
Requirement already satisfied: scipy in /opt/conda/lib/python3.10/site-packages
(from shap) (1.14.0)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.10/site-p
ackages (from shap) (1.2.2)
Requirement already satisfied: pandas in /opt/conda/lib/python3.10/site-package
s (from shap) (2.2.2)
Requirement already satisfied: tqdm>=4.27.0 in /opt/conda/lib/python3.10/site-p
ackages (from shap) (4.66.4)
Requirement already satisfied: packaging>20.9 in /opt/conda/lib/python3.10/sit
e-packages (from shap) (21.3)
Requirement already satisfied: slicer==0.0.7 in /opt/conda/lib/python3.10/site-
packages (from shap) (0.0.7)
Requirement already satisfied: numba in /opt/conda/lib/python3.10/site-packages
(from shap) (0.58.1)
Requirement already satisfied: cloudpickle in /opt/conda/lib/python3.10/site-pa
ckages (from shap) (3.0.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/lib/pytho
n3.10/site-packages (from packaging>20.9->shap) (3.1.2)
Requirement already satisfied: llvmlite<0.42,>=0.41.0dev0 in /opt/conda/lib/pyt
hon3.10/site-packages (from numba->shap) (0.41.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python
3.10/site-packages (from pandas->shap) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-p
ackages (from pandas->shap) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.10/sit
e-packages (from pandas->shap) (2024.1)
Requirement already satisfied: joblib>=1.1.1 in /opt/conda/lib/python3.10/site-
packages (from scikit-learn->shap) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/lib/python3.1
0/site-packages (from scikit-learn->shap) (3.5.0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packa
ges (from python-dateutil>=2.8.2->pandas->shap) (1.16.0)
```

Shap:

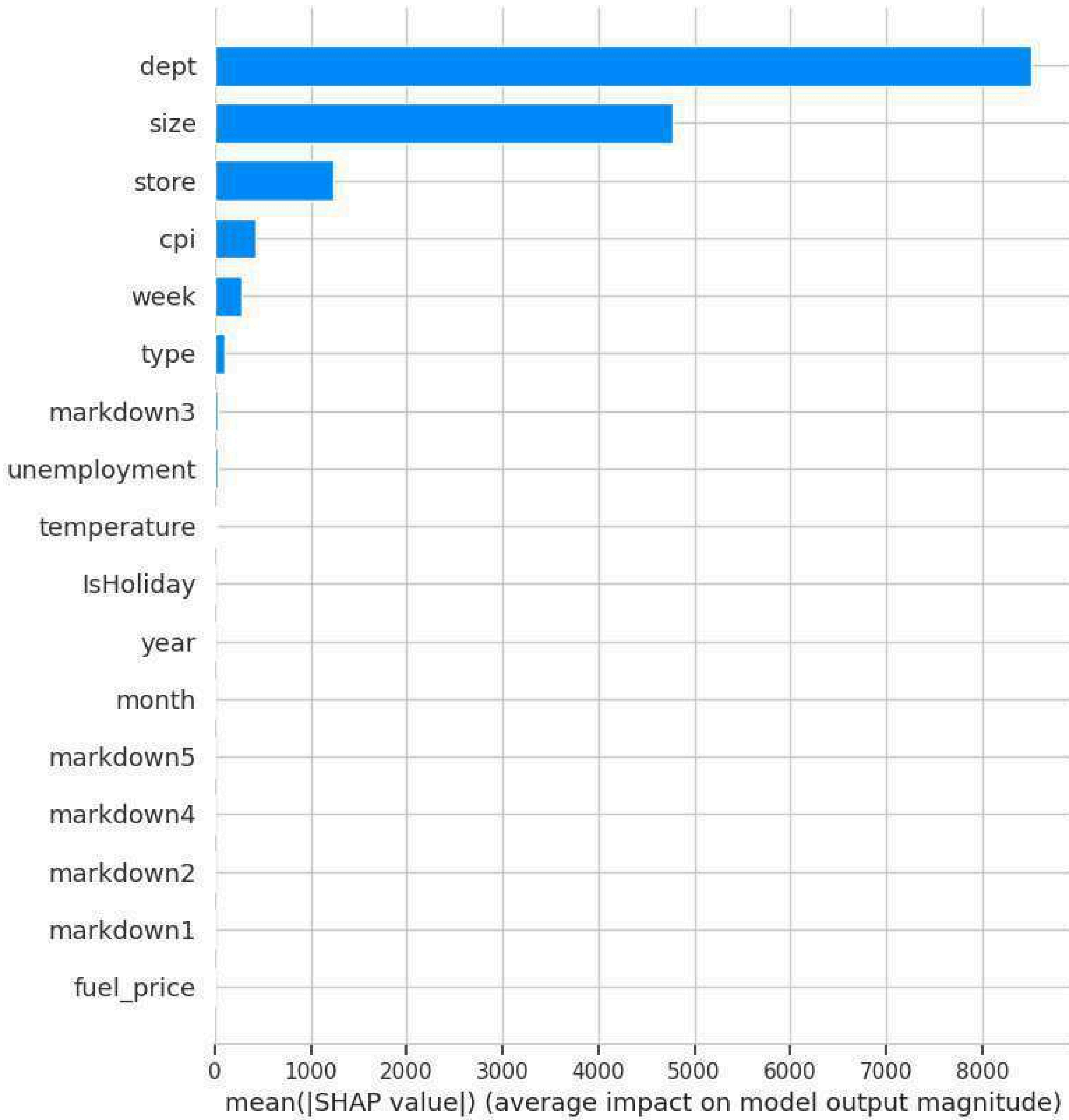
In [95]:

```python
import shap
import matplotlib.pyplot as plt
explainer_gb = shap.TreeExplainer(gb_model_1)
shap_values_gb = explainer_gb.shap_values(X_test)
```

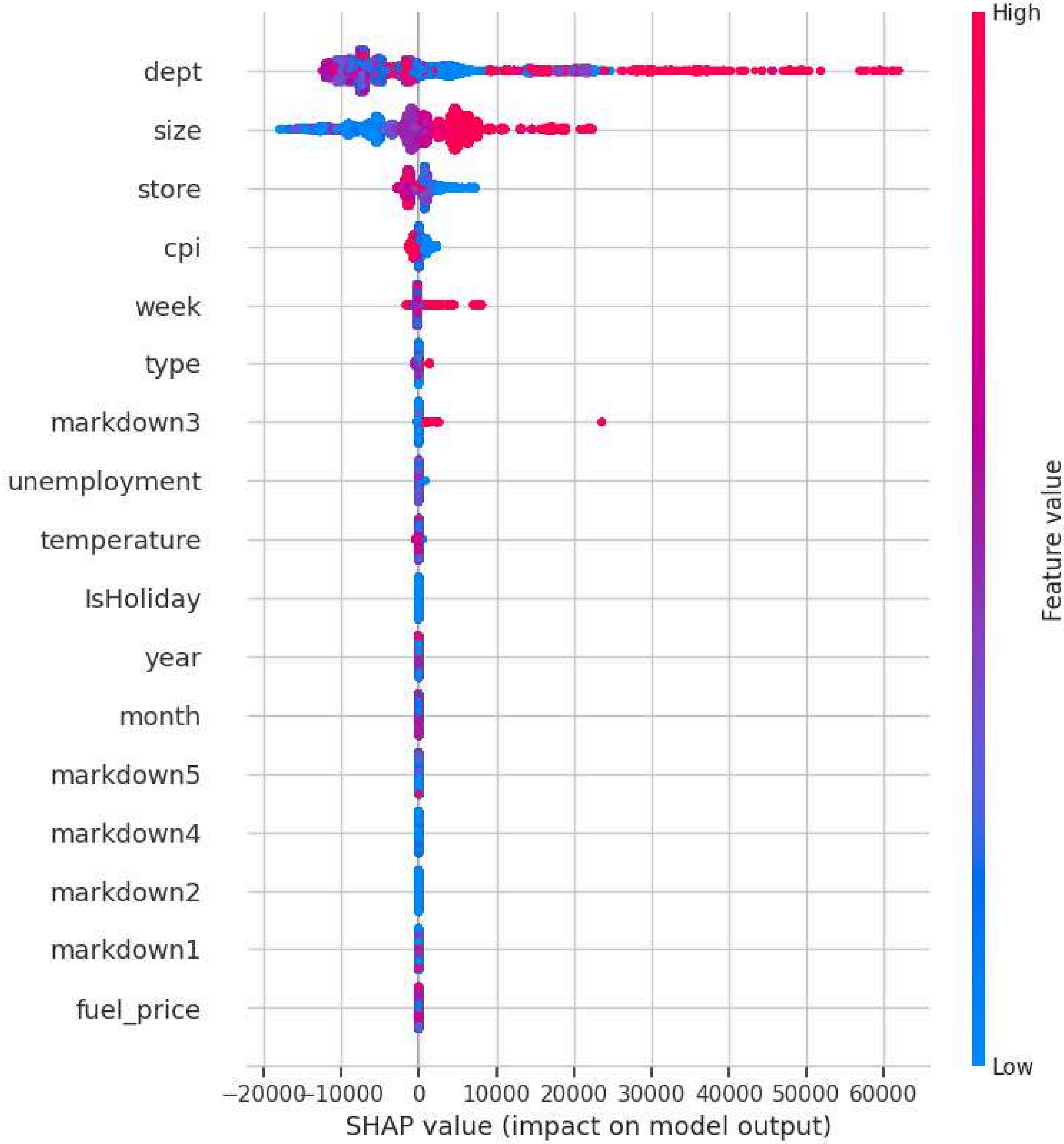
In [96]:

```python
# Visualize the summary plot to understand feature importance and effects
shap.summary_plot(shap_values_gb, X_test, plot_type="bar")  # Bar plot for fea

# Detailed summary plot (beeswarm)
shap.summary_plot(shap_values_gb, X_test)

shap.dependence_plot("size", shap_values_gb, X_test)

shap.force_plot(explainer_gb.expected_value, shap_values_gb[0,:], X_test.iloc[
```

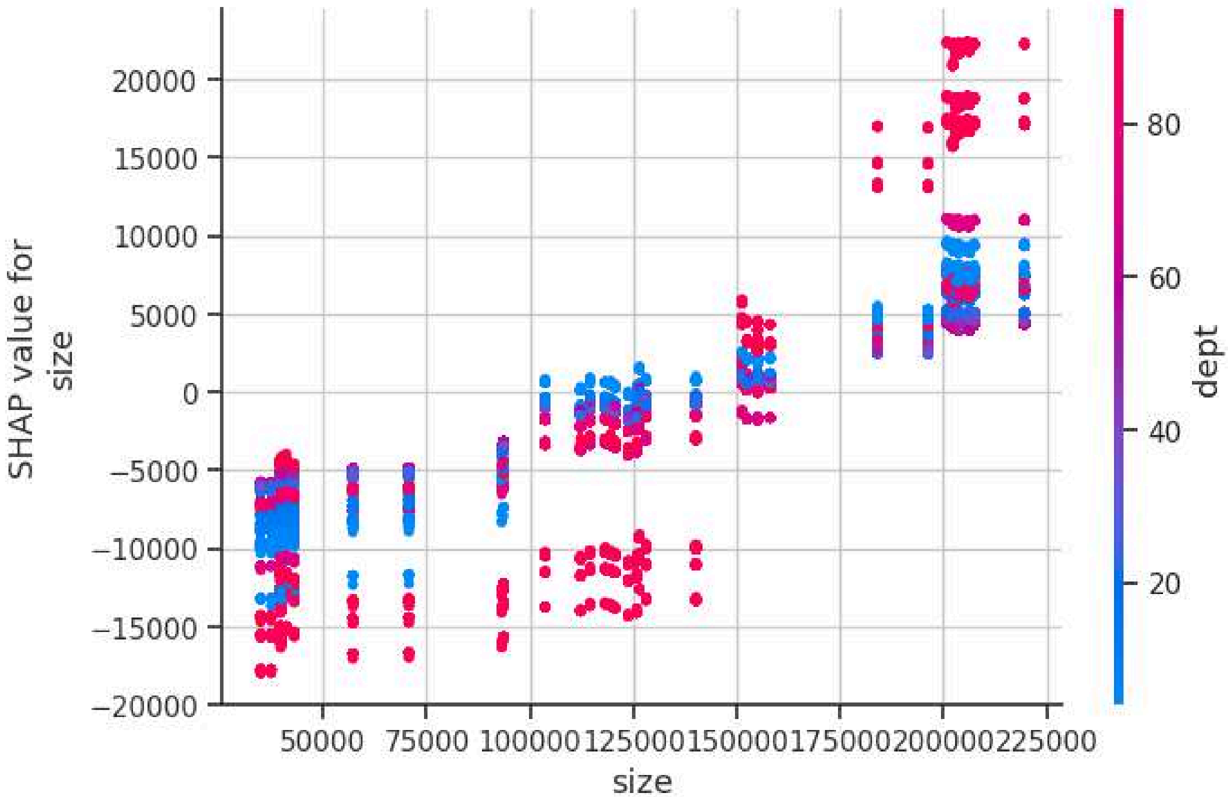
```
plt.show()
```

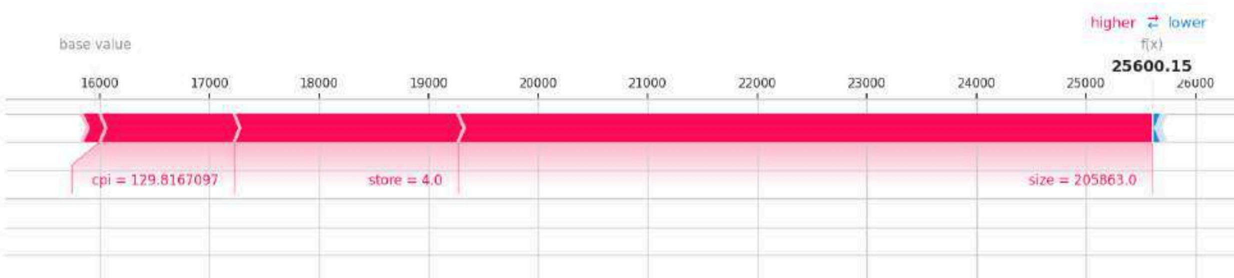


dept
size
store
cpi
week
type
markdown3
unemployment
temperature
IsHoliday
year
month
markdown5
markdown4
markdown2
markdown1
fuel_price
0
1000
2000
3000
4000
5000
6000
7000
8000
mean(|SHAP value|) (average impact on model output magnitude)



dept
size
store
cpi
week
type
markdown3
unemployment
temperature
IsHoliday
year
month
markdown5
markdown4
markdown2
markdown1
fuel_price
−20000
−10000
0
10000
20000
30000
40000
50000
60000
SHAP value (impact on model output)
High
Low
Feature value





Removing Low Value Features

In [97]:
```
# Example of removing low-importance features
X_train_fxt = X_train.drop(['month', 'temperature', 'fuel_price', 'markdown1',
X_test_fxt = X_test.drop(['month', 'temperature', 'fuel_price', 'markdown1', '
```

In [98]:
```
X_train_fxt
```

Out[98]:

	store	cpi	unemployment	dept	IsHoliday	type	size	week
138466	33	127.087677	9.27	87	1	0	39690	52
289214	26	136.588387	7.60	21	0	0	152513	51
52351	42	126.136065	9.52	1	0	2	39690	22
203504	34	129.049032	10.58	29	0	0	158114	22
233606	34	129.201581	10.64	30	0	0	158114	32
...	...	...	...	...	...	...	...	...
259178	16	195.026101	6.23	1	0	1	57197	41
365838	14	191.064610	8.57	6	0	0	200898	25
131932	1	211.405312	7.84	45	0	0	151315	50
146867	43	203.831516	10.40	32	0	2	41062	3
121958	13	126.607200	7.80	55	0	0	219622	46

337256 rows × 8 columns

In [99]:

```
X_test_fxt
```

Out[99]:

	store	cpi	unemployment	dept	IsHoliday	type	size	week
272342	4	129.816710	5.14	13	0	0	205863	45
176581	30	214.488691	7.93	42	0	2	42988	13
354212	32	197.588605	8.09	11	0	0	203007	21
281444	3	222.158952	7.20	26	0	1	37392	48
124208	23	132.836933	5.29	14	1	1	114533	47
...	...	...	...	...	...	...	...	...
415987	13	131.149968	5.62	20	0	0	219622	42
193068	23	134.514367	4.78	96	0	1	114533	18
346600	4	131.136000	4.31	85	0	0	205863	18
380513	10	130.719581	7.17	34	0	1	126512	30
189348	11	219.023610	7.57	36	0	0	207499	17

84314 rows × 8 columns

Modelling : After Feature Engineering

In [100…
```
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
```

In [101…
```
#Model: Gradient Boosting Regressor
import time
start_time = time.time()
gb_model_1 = GradientBoostingRegressor(n_estimators=50, random_state=42)
gb_model_1.fit(X_train_fxt, y_train)
gb_pred_1 = gb_model_1.predict(X_test_fxt)
gb_mse_1 = mean_squared_error(y_test, gb_pred_1)
gb_r2_1 = r2_score(y_test, gb_pred_1)
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_1)
```

```
Execution Time:  0.3382309834162394
R^2 VALUE:  0.6485625012424316
```

In [102…
```
start_time = time.time()
gb_model_2 = GradientBoostingRegressor(n_estimators=100, random_state=42)
gb_model_2.fit(X_train_fxt, y_train)
gb_pred_2 = gb_model_2.predict(X_test_fxt)
gb_mse_2 = mean_squared_error(y_test, gb_pred_2)
gb_r2_2 = r2_score(y_test, gb_pred_2)
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_2)
```

```
Execution Time:  0.6674617290496826
R^2 VALUE:  0.7422041591026212
```

In [103…
```
start_time = time.time()
gb_model_3 = GradientBoostingRegressor(n_estimators=200, random_state=42)
gb_model_3.fit(X_train_fxt, y_train)
gb_pred_3 = gb_model_3.predict(X_test_fxt)
gb_mse_3 = mean_squared_error(y_test, gb_pred_3)
gb_r2_3 = r2_score(y_test, gb_pred_3)
end_time = time.time()

# Calculate the execution time in minutes
execution_time = (end_time - start_time) / 60
print("Execution Time: ", execution_time)
print("R^2 VALUE: ", gb_r2_3)
```

```
Execution Time:  1.3894351879755655
R^2 VALUE:  0.8031918357739974
```

LightGBM

In [104…

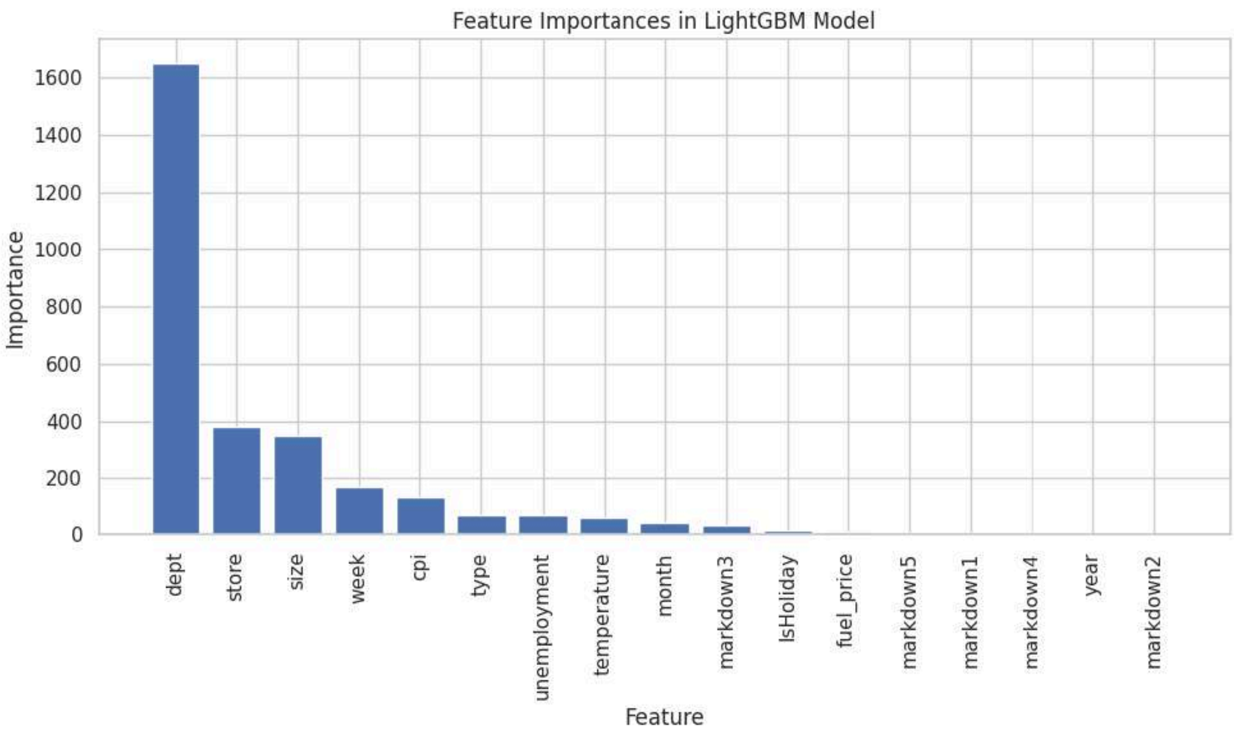
```
import lightgbm as lgb
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error

# Initialize and fit the LightGBM regressor model
lgb_regressor = lgb.LGBMRegressor(n_estimators=100, learning_rate=0.1)
lgb_regressor.fit(X_train, y_train)

# Make predictions
lgb_predictions = lgb_regressor.predict(X_test)

# Calculate R² value using the score method
r2_value_score_method = lgb_regressor.score(X_test, y_test)

# Calculate R² value using the r2_score function
r2_value_function = r2_score(y_test, lgb_predictions)

# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, lgb_predictions)

# Calculate Mean Squared Error (MSE)
mse = mean_squared_error(y_test, lgb_predictions)

# Calculate Root Mean Squared Error (RMSE)
rmse = mse ** 0.5

print(f"R² value using score method: {r2_value_score_method}")
print(f"R² value using r2_score function: {r2_value_function}")
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
```

```
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testi
ng was 0.071640 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 2521
[LightGBM] [Info] Number of data points in the train set: 337256, number of use
d features: 17
[LightGBM] [Info] Start training from score 15960.785333
R² value using score method: 0.9088740788799163
R² value using r2_score function: 0.9088740788799163
Mean Absolute Error (MAE): 4113.130165637804
Mean Squared Error (MSE): 46986483.971656725
Root Mean Squared Error (RMSE): 6854.6687718413295
```

In [105…

```
# Get feature importances
feature_importances = lgb_regressor.feature_importances_
features = X_train.columns

# Sort features by importance
indices = np.argsort(feature_importances)[::-1]
```

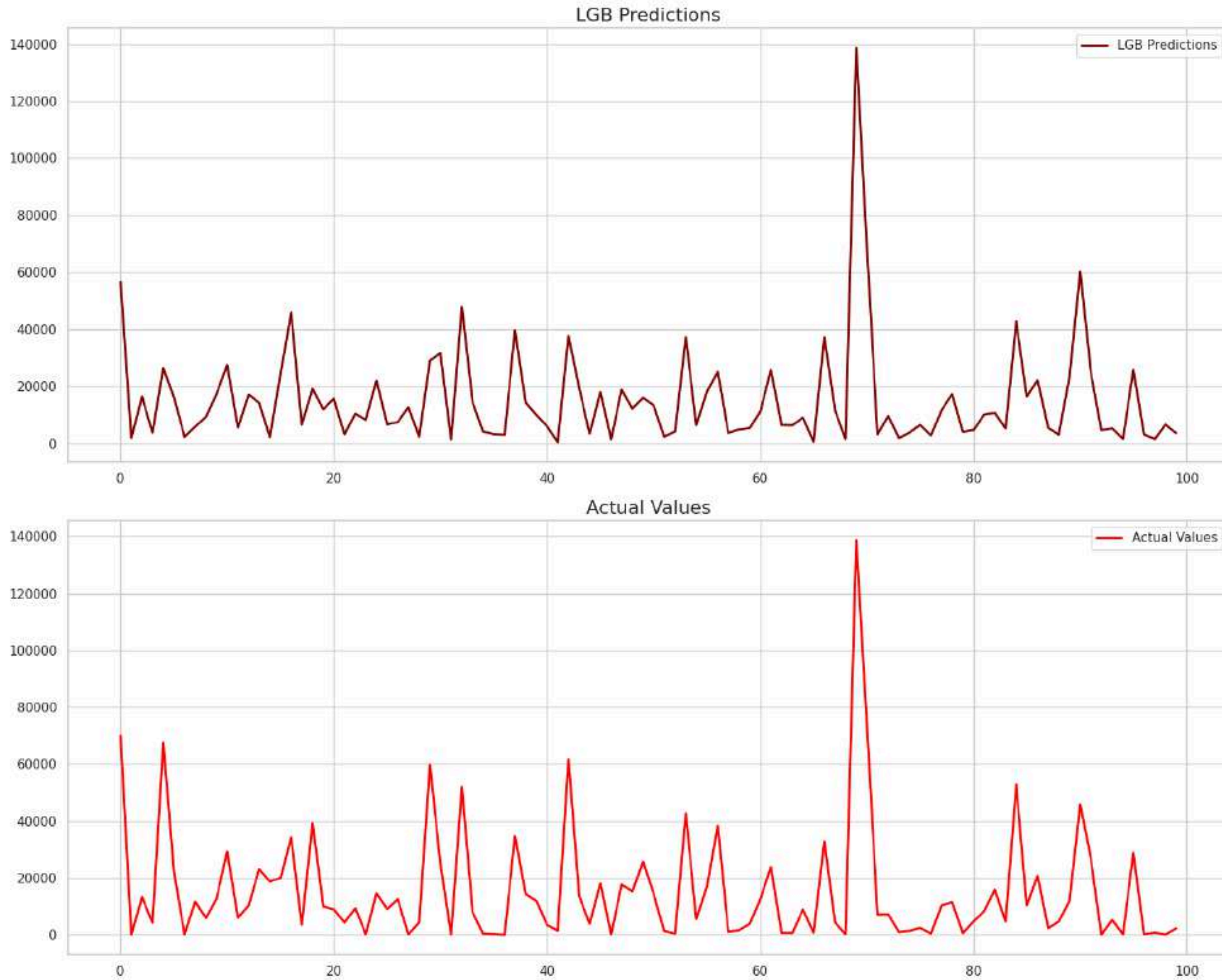
```
sorted_features = features[indices]
sorted_importances = feature_importances[indices]

# Plot feature importances
plt.figure(figsize=(10, 6))
plt.bar(range(len(sorted_features)), sorted_importances, align='center')
plt.xticks(range(len(sorted_features)), sorted_features, rotation=90)
plt.title('Feature Importances in LightGBM Model')
plt.xlabel('Feature')
plt.ylabel('Importance')
plt.tight_layout()
plt.show()
```



In [106…

```
# Select a subset of the test data for visualization
n = 100  # Number of points to plot
y_test_subset = y_test[:n].values

# Create subplots
fig, ax = plt.subplots(2, 1, figsize=(15,12))

# First subplot: LGB Predictions
ax[0].plot(lgb_predictions[:n], label="LGB Predictions", linewidth=2.0, color=
ax[0].set_title('LGB Predictions', fontsize=16)
ax[0].legend(loc="best")

# Second subplot: Actual values
ax[1].plot(y_test_subset, label="Actual Values", linewidth=2.0, color='red')
ax[1].set_title('Actual Values', fontsize=16)
ax[1].legend(loc="best")

# Adjust the layout
```

```
plt.tight_layout()

# Show the plot
plt.show()
```



Forecasting : ARIMA, Modified ARIMA, SARIMA

ARIMA

In [107…

```
from statsmodels.tsa.arima.model import ARIMA
```

In [108…

```
data['date'] = pd.to_datetime(data['date'])

# Aggregate the data by week
weekly_sales = data.groupby('date')['weekly_sales'].sum().reset_index()

# Set the date as the index
weekly_sales.set_index('date', inplace=True)
```

In [109…

```
# Define the training period
train = weekly_sales.iloc[:-52]  # Use all but the last year for training
test = weekly_sales.iloc[-52:]   # Use the last year for testing
```

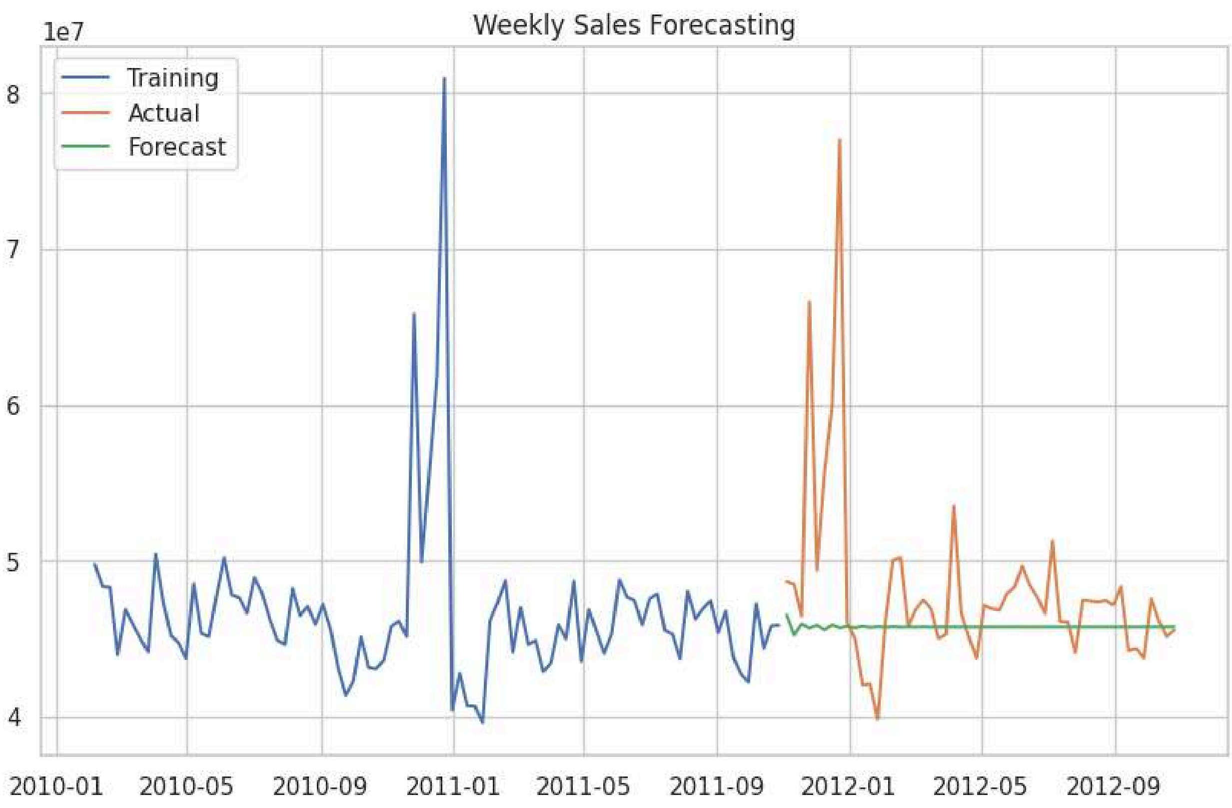
In [110…

```
# Fit the ARIMA model
model = ARIMA(train, order=(5,1,0))  # (p,d,q) parameters can be tuned
model_fit = model.fit()
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

In [111…

```
# Forecast for the test period
forecast = model_fit.forecast(steps=len(test))

# Plot the forecast against the actual values
plt.figure(figsize=(10, 6))
plt.plot(train.index, train, label='Training')
plt.plot(test.index, test, label='Actual')
plt.plot(test.index, forecast, label='Forecast')
plt.legend(loc='upper left')
plt.title('Weekly Sales Forecasting')
plt.show()
```



Tuning ARIMA Model

In [112…

```
from statsmodels.tsa.arima.model import ARIMA
import pandas as pd
import numpy as np

data['date'] = pd.to_datetime(data['date'])

# Aggregate the data by week
weekly_sales = data.groupby('date')['weekly_sales'].sum().reset_index()

# Set the date as the index
weekly_sales.set_index('date', inplace=True)

# Define the training period
train = weekly_sales.iloc[:-52]  # Use all but the last year for training
test = weekly_sales.iloc[-52:]   # Use the last year for testing

# Grid search for ARIMA parameters
best_aic = np.inf
best_order = None
best_model = None

# Define a range of p, d, q values to try
p_values = range(0, 6)
d_values = range(0, 2)
q_values = range(0, 6)

for p in p_values:
    for d in d_values:
        for q in q_values:
            try:
                model = ARIMA(train, order=(p, d, q))
                model_fit = model.fit()
                aic = model_fit.aic
                if aic < best_aic:
                    best_aic = aic
                    best_order = (p, d, q)
                    best_model = model_fit
            except Exception as e:
                continue

print(f"Best ARIMA order: {best_order} with AIC: {best_aic}")

# Use the best model for predictions
predictions = best_model.forecast(steps=len(test))

# Evaluate the model
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(test, predictions)
rmse = np.sqrt(mse)
```

```
print(f"RMSE for best ARIMA model: {rmse}")
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
```

```
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
```

```
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
```

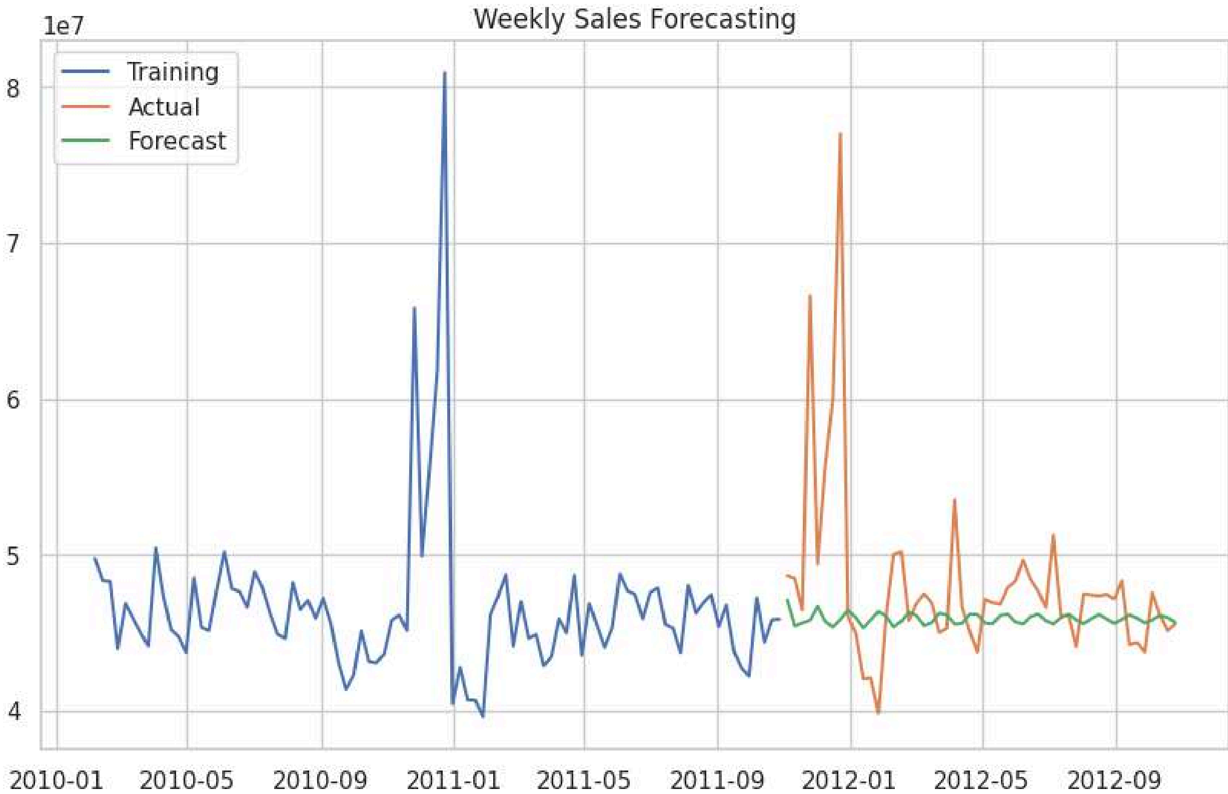
```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Non-stationary starting autoregressive parameters found. Using zeros as startin
g parameters.
Non-invertible starting MA parameters found. Using zeros as starting parameter
s.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Best ARIMA order: (0, 1, 5) with AIC: 3050.9341382069338
RMSE for best ARIMA model: 6263405.37778519
```

```
Maximum Likelihood optimization failed to converge. Check mle_retvals
```

In [113…

```python
# Forecast for the test period
forecast = model_fit.forecast(steps=len(test))

# Plot the forecast against the actual values
plt.figure(figsize=(10, 6))
plt.plot(train.index, train, label='Training')
plt.plot(test.index, test, label='Actual')
plt.plot(test.index, forecast, label='Forecast')
```

```
plt.legend(loc='upper left')
plt.title('Weekly Sales Forecasting')
plt.show()
```



SARIMA

In [114…

```
import pandas as pd
import numpy as np
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX
from sklearn.metrics import mean_squared_error
from tqdm import tqdm

data['date'] = pd.to_datetime(data['date'])

# Aggregate the data by week
weekly_sales = data.groupby('date')['weekly_sales'].sum().reset_index()

# Set the date as the index
weekly_sales.set_index('date', inplace=True)

# Define the training period
train = weekly_sales.iloc[:-52]
test = weekly_sales.iloc[-52:]

# Grid search for ARIMA/SARIMA parameters
best_aic = np.inf
```

```python
best_order = None
best_seasonal_order = None
best_model = None

# Define a range of p, d, q values to try
p_values = range(0, 3)
d_values = range(0, 2)
q_values = range(0, 3)
P_values = range(0, 2)
D_values = range(0, 2)
Q_values = range(0, 2)
S = 52  # Assuming weekly seasonality

for p in tqdm(p_values):
    for d in d_values:
        for q in q_values:
            for P in P_values:
                for D in D_values:
                    for Q in Q_values:
                        try:
                            model = SARIMAX(train, order=(p, d, q), seasonal_o
                            model_fit = model.fit(disp=False)
                            aic = model_fit.aic
                            if aic < best_aic:
                                best_aic = aic
                                best_order = (p, d, q)
                                best_seasonal_order = (P, D, Q, S)
                                best_model = model_fit
                        except Exception as e:
                            continue

print(f"Best SARIMA order: {best_order} with seasonal order: {best_seasonal_or

# Walk-forward validation
predictions = []
for i in range(len(test)):
    train_data = weekly_sales.iloc[:-(52-i)]
    model = SARIMAX(train_data, order=best_order, seasonal_order=best_seasonal
    model_fit = model.fit(disp=False)
    forecast = model_fit.forecast(steps=1)
    predictions.append(forecast.values[0])

# Evaluate the model
mse = mean_squared_error(test, predictions)
rmse = np.sqrt(mse)

print(f"RMSE for best SARIMA model: {rmse}")
```

```
  0%|          | 0/3 [00:00<?, ?it/s]No frequency information was provided, so
inferred frequency W-FRI will be used.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
 33%|███       | 1/3 [00:32<01:04, 32.17s/it]No frequency information was provi
ded, so inferred frequency W-FRI will be used.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
 67%|██████▋   | 2/3 [01:23<00:43, 43.24s/it]No frequency information was provi
ded, so inferred frequency W-FRI will be used.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
```

```
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Maximum Likelihood optimization failed to converge. Check mle_retvals
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
100%|██████████| 3/3 [02:29<00:00, 49.99s/it]
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
Best SARIMA order: (0, 0, 0) with seasonal order: (0, 1, 1, 52) and AIC: 4.0
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
```

```
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
invalid value encountered in divide
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

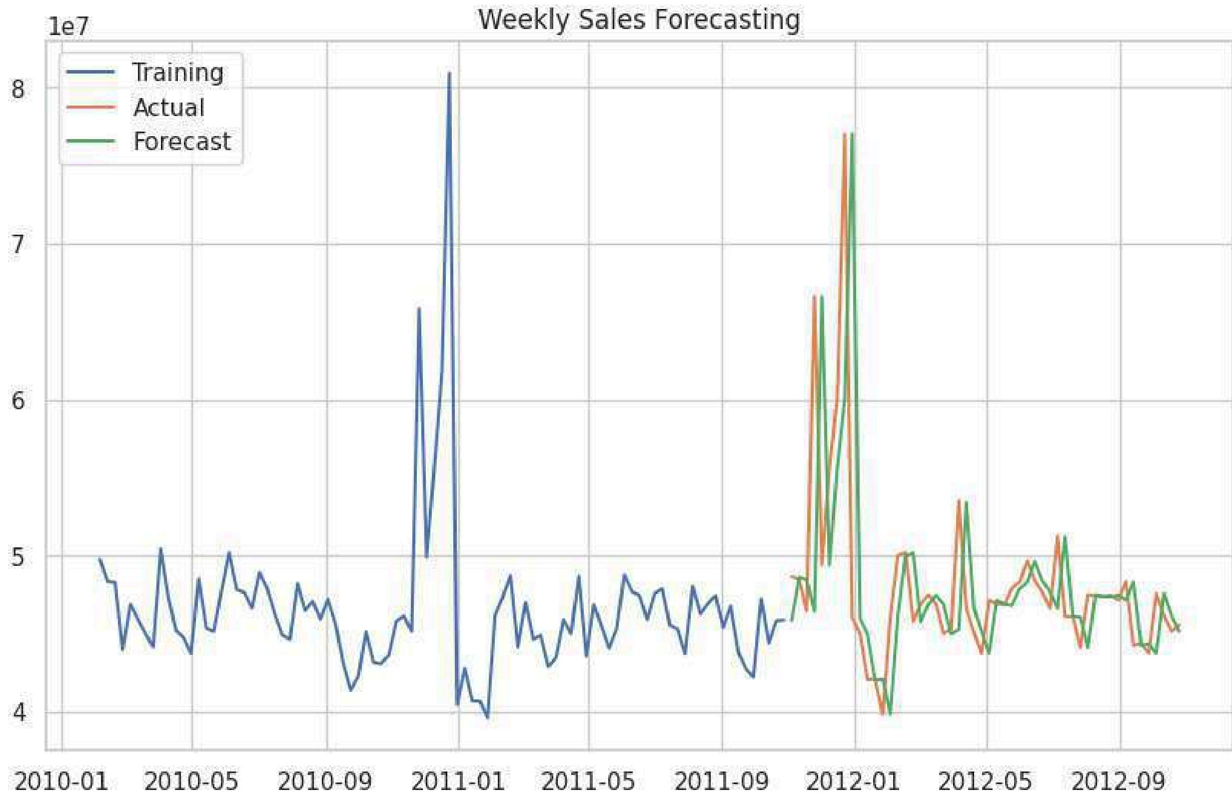
```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
```

```
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
No frequency information was provided, so inferred frequency W-FRI will be use
d.
Too few observations to estimate starting parameters for seasonal ARMA. All par
ameters except for variances will be set to zeros.
RMSE for best SARIMA model: 2128341.295087557
```

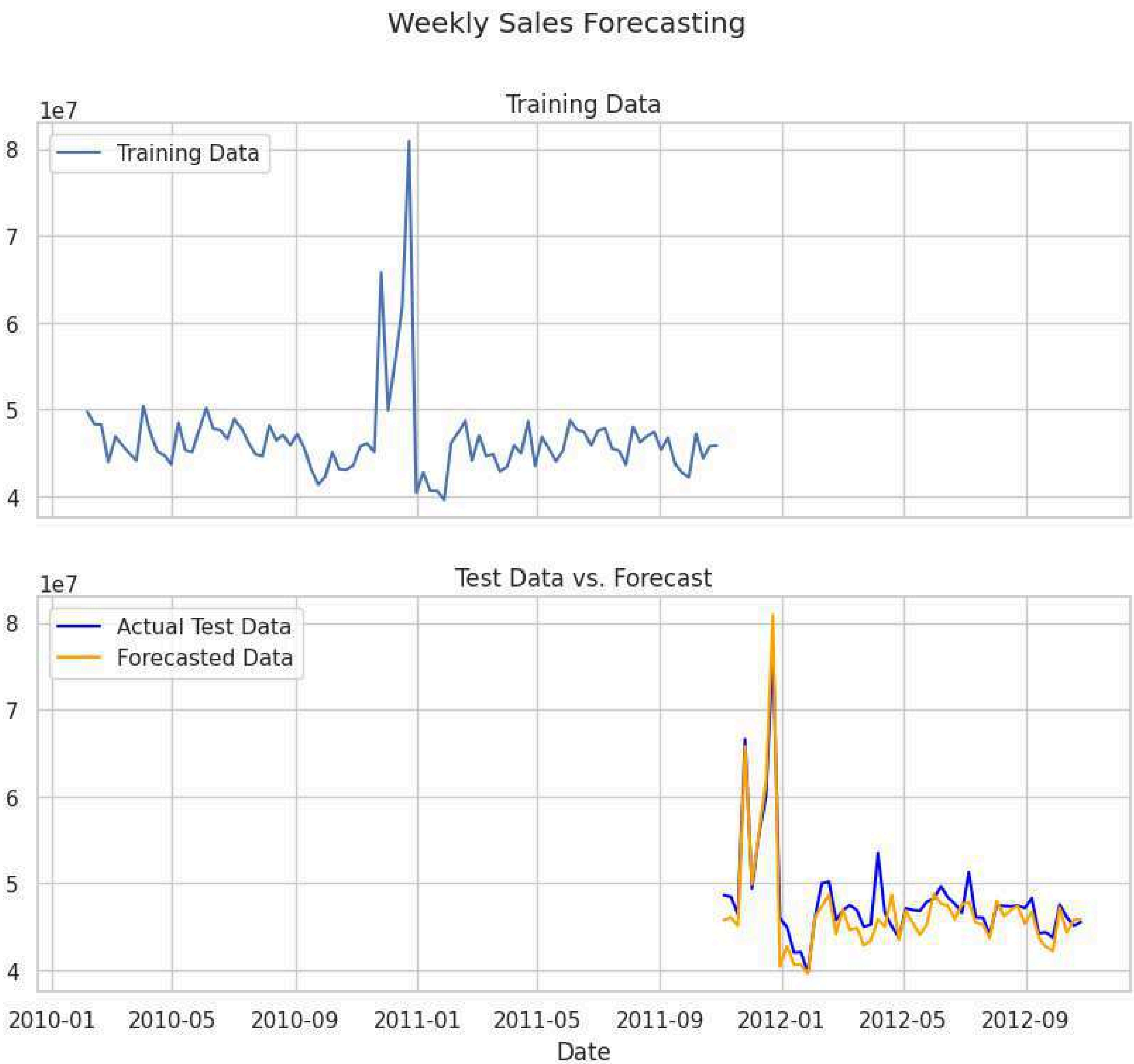
In [115…

```
# Forecast for the test period
forecast = model_fit.forecast(steps=len(test))

# Plot the forecast against the actual values
plt.figure(figsize=(10, 6))
plt.plot(train.index, train, label='Training')
plt.plot(test.index, test, label='Actual')
plt.plot(test.index, forecast, label='Forecast')
plt.legend(loc='upper left')
plt.title('Weekly Sales Forecasting')
plt.show()
```



In [116…

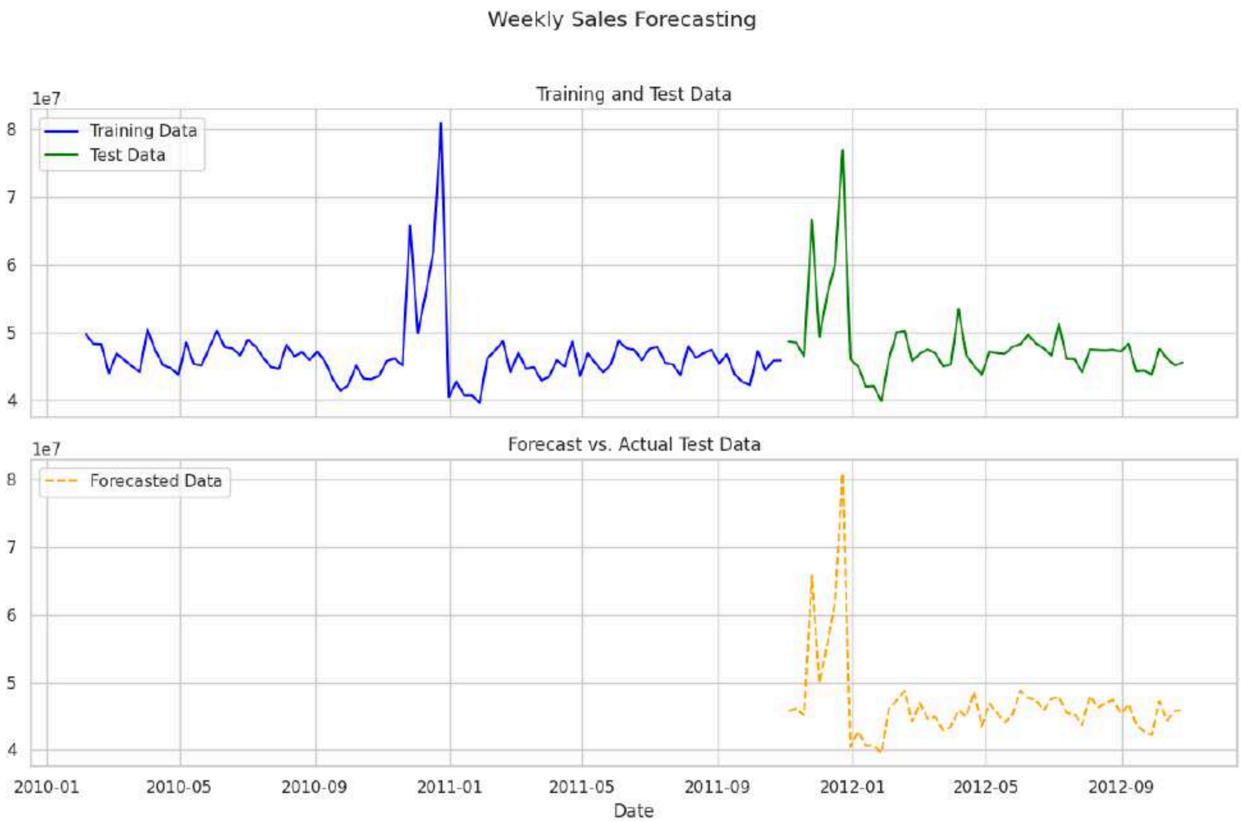
```
import matplotlib.pyplot as plt

# Forecast for the test period
forecast = best_model.forecast(steps=len(test))

# Plot the training and test data with forecasts in two subplots
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 8), sharex=True)

# Upper subplot for training data
ax1.plot(train.index, train, label='Training Data')
ax1.set_title('Training Data')
ax1.legend(loc='upper left')

# Lower subplot for test data and forecasts
ax2.plot(test.index, test, label='Actual Test Data', color='blue')
ax2.plot(test.index, forecast, label='Forecasted Data', color='orange')
ax2.set_title('Test Data vs. Forecast')
ax2.legend(loc='upper left')

# Set common x-axis label and overall title
fig.suptitle('Weekly Sales Forecasting')
plt.xlabel('Date')
plt.show()
```



In [117…

```
# Forecast for the test period
forecast = best_model.forecast(steps=len(test))

# Plot the training and test data in the upper plot, and forecast vs. actual t
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 8), sharex=True)

# Upper subplot for training and test data
ax1.plot(train.index, train, label='Training Data', color='blue')
ax1.plot(test.index, test, label='Test Data', color='green')
ax1.set_title('Training and Test Data')
ax1.legend(loc='upper left')

# Lower subplot for forecast vs. actual test data
# ax2.plot(test.index, test, label='Actual Test Data', color='green')
ax2.plot(test.index, forecast, label='Forecasted Data', color='orange', linest
ax2.set_title('Forecast vs. Actual Test Data')
ax2.legend(loc='upper left')
```

```
# Set common labels
fig.suptitle('Weekly Sales Forecasting')
plt.xlabel('Date')
plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```



In [118…

```
from sklearn.metrics import mean_absolute_error, mean_squared_error

# Calculate MAE
mae = mean_absolute_error(test, predictions)

# Calculate RMSE
rmse = np.sqrt(mean_squared_error(test, predictions))

# Calculate MAPE
mape = np.mean(np.abs((test.values - predictions) / test.values)) * 100

# Calculate Accuracy as a Percentage (1 - MAPE/100)
accuracy_percentage = 100 - mape

# Print the evaluation metrics
print(f"Mean Absolute Error (MAE): {mae}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"Mean Absolute Percentage Error (MAPE): {mape}%")
print(f"Accuracy: {accuracy_percentage}%")
```

```
Mean Absolute Error (MAE): 1597388.9843533235
Root Mean Squared Error (RMSE): 2128341.295087557
Mean Absolute Percentage Error (MAPE): 10.139488649093506%
Accuracy: 89.86051135090649%
```

In [ ]: