JL Project - Create 5 Year Forecast

In [124…
```
!pip install pandas matplotlib prophet
```

```
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-packages (2.2.2)
Requirement already satisfied: matplotlib in c:\users\admin\anaconda3\lib\site-packages (3.8.4)
Requirement already satisfied: prophet in c:\users\admin\anaconda3\lib\site-packages (1.1.6)
Requirement already satisfied: numpy>=1.26.0 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=8 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib) (3.0.9)
Requirement already satisfied: cmdstanpy>=1.0.4 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (1.2.4)
Requirement already satisfied: holidays<1,>=0.25 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (0.58)
Requirement already satisfied: tqdm>=4.36.1 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (4.66.4)
Requirement already satisfied: importlib-resources in c:\users\admin\anaconda3\lib\site-packages (from prophet) (6.4.5)
Requirement already satisfied: stanio<2.0.0,>=0.4.0 in c:\users\admin\anaconda3\lib\site-packages (from cmdstanpy>=1.0.4->prophet) (0.5.1)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site-packages (from tqdm>=4.36.1->prophet) (0.4.6)
```

In [125…
```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\admin\anaconda3\lib\site-packages (2.2.2)
Requirement already satisfied: numpy>=1.26.0 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\lib\site-packages (from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [126…
```
pip install chardet
```

```
Requirement already satisfied: chardet in c:\users\admin\anaconda3\lib\site-packages (4.0.0)
Note: you may need to restart the kernel to use updated packages.
```

In [128…
```
import chardet

# Read a sample of the file to detect its encoding
```

```
with open('sales_data.csv', 'rb') as file:
    raw_data = file.read(10000)  # Read the first 10,000 bytes
    result = chardet.detect(raw_data)
    encoding = result['encoding']
    print(f"Detected encoding: {encoding}")
```

```
Detected encoding: ISO-8859-1
```

In [130…

```
import pandas as pd

# Load the CSV with the detected encoding
df = pd.read_csv('sales_data.csv', encoding='ISO-8859-1')  # or use the detected encoding from the previous step

# Display the first few rows to verify successful loading
print(df.head())
```

```
   Quarter  Year YearQuarter      SalesOrderID    RevisionNumber  \
0        2  2011     2011-Q2  25/08/2019 00:00  08/01/1900 00:00
1        2  2011     2011-Q2  29/08/2019 00:00  08/01/1900 00:00
2        2  2011     2011-Q2  30/08/2019 00:00  08/01/1900 00:00
3        2  2011     2011-Q2  05/09/2019 00:00  08/01/1900 00:00
4        2  2011     2011-Q2  11/09/2019 00:00  08/01/1900 00:00

          OrderDate           DueDate          ShipDate  Status  \
0  01/06/2011 00:00  13/06/2011 00:00  08/06/2011 00:00       5
1  02/06/2011 00:00  14/06/2011 00:00  09/06/2011 00:00       5
2  02/06/2011 00:00  14/06/2011 00:00  09/06/2011 00:00       5
3  04/06/2011 00:00  16/06/2011 00:00  11/06/2011 00:00       5
4  05/06/2011 00:00  17/06/2011 00:00  12/06/2011 00:00       5

   OnlineOrderFlag  ... CreditCardApprovalCode CurrencyRateID   SubTotal  \
0             True  ...         1230194Vi41919            NaN  £3,578.27
1             True  ...          230189Vi56258            NaN  £3,578.27
2             True  ...          230298Vi99064            NaN  £3,578.27
3             True  ...          830288Vi85808            NaN  £3,578.27
4             True  ...         1130295Vi31203            NaN  £3,578.27

    TaxAmt  Freight   TotalDue  Comment                               rowguid  \
0  £286.26   £89.46  £3,953.99      NaN  9310C7F0-9A84-4CE9-BE08-F700FB1AADF7
1  £286.26   £89.46  £3,953.99      NaN  F02C4CB6-A5B5-4CE4-9473-CB084E383196
2  £286.26   £89.46  £3,953.99      NaN  0DA77D6E-223E-4BC6-A2ED-43B387692C68
3  £286.26   £89.46  £3,953.99      NaN  9DE30294-9066-4988-A3AD-09A0713348E5
4  £286.26   £89.46  £3,953.99      NaN  BF5155EB-C5BE-4245-8FC4-F801DB5B052D

       ModifiedDate Sales.SalesTerritory.CountryRegionCode
0  08/06/2011 00:00                                    US
1  09/06/2011 00:00                                    US
2  09/06/2011 00:00                                    US
3  11/06/2011 00:00                                    US
4  12/06/2011 00:00                                    US

[5 rows x 30 columns]
```

In [134…

```
pip install prophet
```

```
Requirement already satisfied: prophet in c:\users\admin\anaconda3\lib\site-packages (1.1.6)
Requirement already satisfied: cmdstanpy>=1.0.4 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (1.2.4)
Requirement already satisfied: numpy>=1.15.4 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (1.26.4)
Requirement already satisfied: matplotlib>=2.0.0 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (3.8.4)
Requirement already satisfied: pandas>=1.0.4 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (2.2.2)
Requirement already satisfied: holidays<1,>=0.25 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (0.58)
Requirement already satisfied: tqdm>=4.36.1 in c:\users\admin\anaconda3\lib\site-packages (from prophet) (4.66.4)
Requirement already satisfied: importlib-resources in c:\users\admin\anaconda3\lib\site-packages (from prophet) (6.4.5)
Requirement already satisfied: stanio<2.0.0,>=0.4.0 in c:\users\admin\anaconda3\lib\site-packages (from cmdstanpy>=1.0.4->prophet) (0.5.1)
Requirement already satisfied: python-dateutil in c:\users\admin\anaconda3\lib\site-packages (from holidays<1,>=0.25->prophet) (2.9.0.post0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (23.2)
Requirement already satisfied: pillow>=8 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=2.0.0->prophet) (3.0.9)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\anaconda3\lib\site-packages (from pandas>=1.0.4->prophet) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\anaconda3\lib\site-packages (from pandas>=1.0.4->prophet) (2023.3)
Requirement already satisfied: colorama in c:\users\admin\anaconda3\lib\site-packages (from tqdm>=4.36.1->prophet) (0.4.6)
Requirement already satisfied: six>=1.5 in c:\users\admin\anaconda3\lib\site-packages (from python-dateutil->holidays<1,>=0.25->prophet) (1.
16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [122…

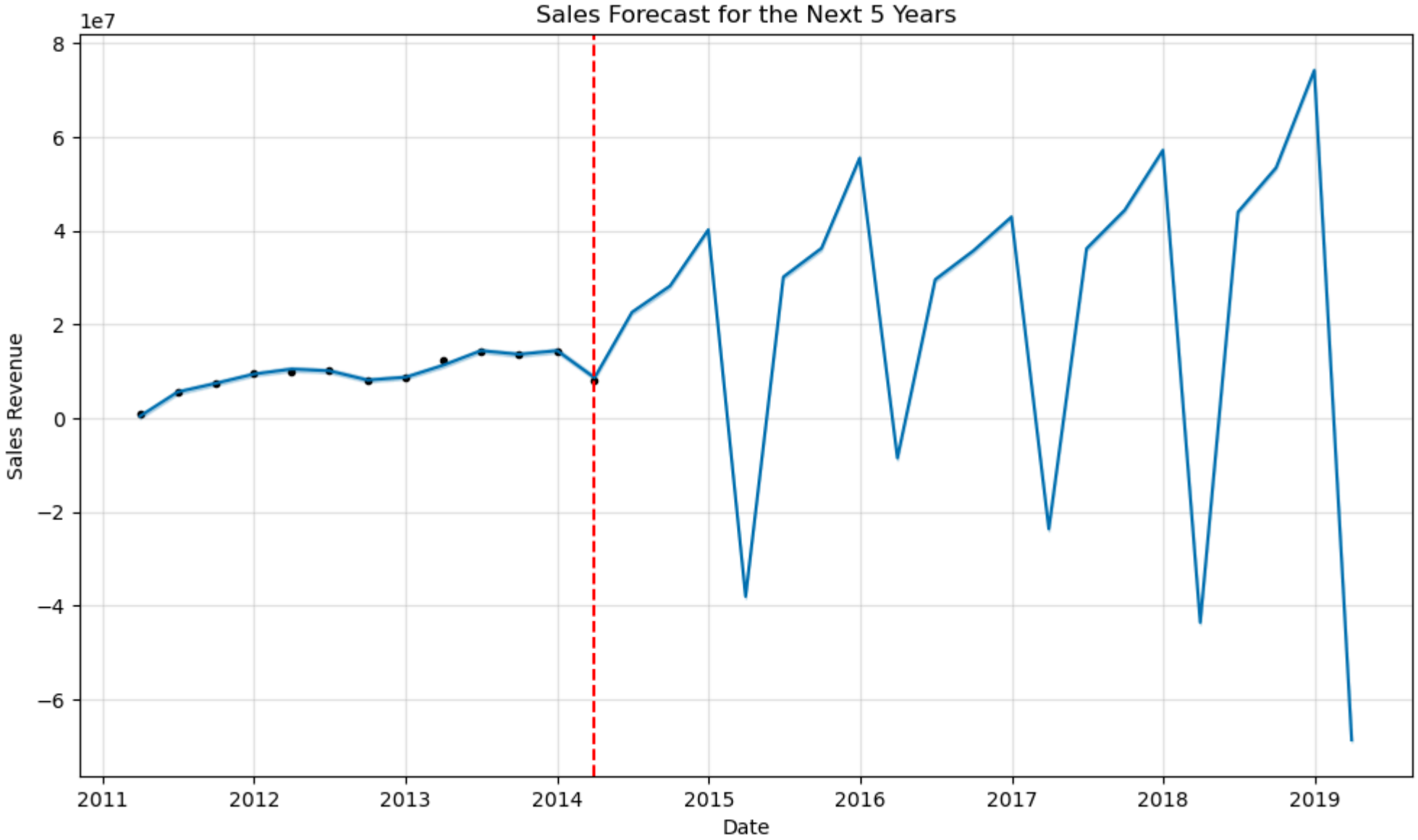
```python
# Import necessary libraries to handle data, visualize it, and perform forecasting.
import pandas as pd
import matplotlib.pyplot as plt
from prophet import Prophet

# I start by loading the sales data from a CSV file, ensuring to handle encoding issues.
df = pd.read_csv('sales_data.csv', encoding='ISO-8859-1')

# To confirm that the data has loaded correctly, I display the first few rows.
print("Initial Data:")
print(df.head())

# Next, I check the columns in the DataFrame to understand its structure better.
print("\nColumns in the DataFrame:")
print(df.columns)

# It's important for me to remove any unnecessary whitespace from the column names for easier access.
df.columns = df.columns.str.strip()

# I ensure that the date columns (OrderDate, DueDate, ShipDate) are in the correct datetime format.
df['OrderDate'] = pd.to_datetime(df['OrderDate'], format='%d/%m/%Y %H:%M', errors='coerce')
df['DueDate'] = pd.to_datetime(df['DueDate'], format='%d/%m/%Y %H:%M', errors='coerce')
```

```
df['ShipDate'] = pd.to_datetime(df['ShipDate'], format='%d/%m/%Y %H:%M', errors='coerce')

# I check to see if the conversion to datetime format was successful by displaying the relevant columns.
print("\nConverted OrderDate, DueDate, and ShipDate columns:")
print(df[['OrderDate', 'DueDate', 'ShipDate']].head())

# If the Revenue column doesn't exist, I create it by multiplying UnitPrice and Quantity.
if 'Revenue' not in df.columns:
    if 'UnitPrice' in df.columns and 'Quantity' in df.columns:
        df['Revenue'] = df['UnitPrice'] * df['Quantity']  # Calculate Revenue
    else:
        print("Cannot calculate Revenue, 'UnitPrice' and 'Quantity' columns are required.")

# To ensure the TotalDue column is usable, I clean its values by removing currency symbols and converting to float.
df['TotalDue'] = df['TotalDue'].replace('[\£,]', '', regex=True).astype(float)

# I then add additional columns for Year and YearQuarter to facilitate time-based analysis.
df['Year'] = df['OrderDate'].dt.year
df['YearQuarter'] = df['OrderDate'].dt.to_period('Q')  # Creates Year-Quarter period

# Since Prophet requires timestamps, I convert YearQuarter to a timestamp format.
df['YearQuarter'] = df['YearQuarter'].dt.to_timestamp()  # Convert to timestamp

# To verify that my modifications to the DataFrame were successful, I check the updated columns.
print("\nUpdated DataFrame Columns:")
print(df.columns)

# Next, I aggregate the sales data by YearQuarter to prepare for forecasting.
if 'TotalDue' in df.columns:
    quarterly_sales = df.groupby(['YearQuarter']).agg({'TotalDue': 'sum'}).reset_index()

    # I rename the columns to meet Prophet's requirements for forecasting.
    quarterly_sales.rename(columns={'YearQuarter': 'ds', 'TotalDue': 'y'}, inplace=True)

    # Now, I proceed to fit a Prophet model to the aggregated quarterly sales data.
    model = Prophet(seasonality_mode='multiplicative', yearly_seasonality=True)
    model.fit(quarterly_sales)

    # I create a future dataframe that represents the next 5 years (20 quarters) for my predictions.
    future = model.make_future_dataframe(periods=20, freq='QE')  # 20 quarters = 5 years

    # Using the fitted model, I forecast future sales.
    forecast = model.predict(future)

    # Finally, I visualize the forecasted sales data.
    fig = model.plot(forecast)
```

```
    plt.title('Sales Forecast for the Next 5 Years')
    plt.xlabel('Date')
    plt.ylabel('Sales Revenue')
    plt.axvline(x=pd.to_datetime(quarterly_sales['ds'].max()), color='red', linestyle='--')  # Historical cutoff line
    plt.show()
else:
    print("TotalDue column is not available; please check the input data.")

# As an optional step, I can save the modified DataFrame to a new CSV file for future reference.
output_file_path = 'modified_sales_data.csv'
df.to_csv(output_file_path, index=False)  # Save without the index column

print(f"\nDataFrame has been exported to '{output_file_path}'.")
```

```
Initial Data:
   SalesOrderID  RevisionNumber         OrderDate           DueDate  \
0         43702               8  01/06/2011 00:00  13/06/2011 00:00
1         43706               8  02/06/2011 00:00  14/06/2011 00:00
2         43707               8  02/06/2011 00:00  14/06/2011 00:00
3         43713               8  04/06/2011 00:00  16/06/2011 00:00
4         43719               8  05/06/2011 00:00  17/06/2011 00:00

            ShipDate  Status  OnlineOrderFlag SalesOrderNumber  \
0  08/06/2011 00:00       5             True          SO43702
1  09/06/2011 00:00       5             True          SO43706
2  09/06/2011 00:00       5             True          SO43707
3  11/06/2011 00:00       5             True          SO43713
4  12/06/2011 00:00       5             True          SO43719

  PurchaseOrderNumber   AccountNumber  ...   SubTotal   TaxAmt  Freight  \
0                 NaN  10-4030-027645  ...  £3,578.27  £286.26   £89.46
1                 NaN  10-4030-027621  ...  £3,578.27  £286.26   £89.46
2                 NaN  10-4030-027616  ...  £3,578.27  £286.26   £89.46
3                 NaN  10-4030-027601  ...  £3,578.27  £286.26   £89.46
4                 NaN  10-4030-027612  ...  £3,578.27  £286.26   £89.46

    TotalDue  Comment                                rowguid      ModifiedDate  \
0  £3,953.99      NaN  9310C7F0-9A84-4CE9-BE08-F700FB1AADF7  08/06/2011 00:00
1  £3,953.99      NaN  F02C4CB6-A5B5-4CE4-9473-CB084E383196  09/06/2011 00:00
2  £3,953.99      NaN  0DA77D6E-223E-4BC6-A2ED-43B387692C68  09/06/2011 00:00
3  £3,953.99      NaN  9DE30294-9066-4988-A3AD-09A0713348E5  11/06/2011 00:00
4  £3,953.99      NaN  BF5155EB-C5BE-4245-8FC4-F801DB5B052D  12/06/2011 00:00

  Quarter  Year YearQuarter
0       2  2011     2011-Q2
1       2  2011     2011-Q2
2       2  2011     2011-Q2
3       2  2011     2011-Q2
4       2  2011     2011-Q2

[5 rows x 29 columns]

Columns in the DataFrame:
Index(['SalesOrderID', 'RevisionNumber', 'OrderDate', 'DueDate', 'ShipDate',
       'Status', 'OnlineOrderFlag', 'SalesOrderNumber', 'PurchaseOrderNumber',
       'AccountNumber', 'CustomerID', 'SalesPersonID', 'TerritoryID',
       'BillToAddressID', 'ShipToAddressID', 'ShipMethodID', 'CreditCardID',
       'CreditCardApprovalCode', 'CurrencyRateID', 'SubTotal', 'TaxAmt',
       'Freight', 'TotalDue', 'Comment', 'rowguid', 'ModifiedDate', 'Quarter',
       'Year', 'YearQuarter'],
```

```
      dtype='object')

Converted OrderDate, DueDate, and ShipDate columns:
   OrderDate    DueDate   ShipDate
0 2011-06-01 2011-06-13 2011-06-08
1 2011-06-02 2011-06-14 2011-06-09
2 2011-06-02 2011-06-14 2011-06-09
3 2011-06-04 2011-06-16 2011-06-11
4 2011-06-05 2011-06-17 2011-06-12
Cannot calculate Revenue, 'UnitPrice' and 'Quantity' columns are required.

Updated DataFrame Columns:
Index(['SalesOrderID', 'RevisionNumber', 'OrderDate', 'DueDate', 'ShipDate',
       'Status', 'OnlineOrderFlag', 'SalesOrderNumber', 'PurchaseOrderNumber',
       'AccountNumber', 'CustomerID', 'SalesPersonID', 'TerritoryID',
       'BillToAddressID', 'ShipToAddressID', 'ShipMethodID', 'CreditCardID',
       'CreditCardApprovalCode', 'CurrencyRateID', 'SubTotal', 'TaxAmt',
       'Freight', 'TotalDue', 'Comment', 'rowguid', 'ModifiedDate', 'Quarter',
       'Year', 'YearQuarter'],
      dtype='object')
13:44:32 - cmdstanpy - INFO - Chain [1] start processing
13:44:34 - cmdstanpy - INFO - Chain [1] done processing
```




```
DataFrame has been exported to 'modified_sales_data.csv'.
```

In [142…

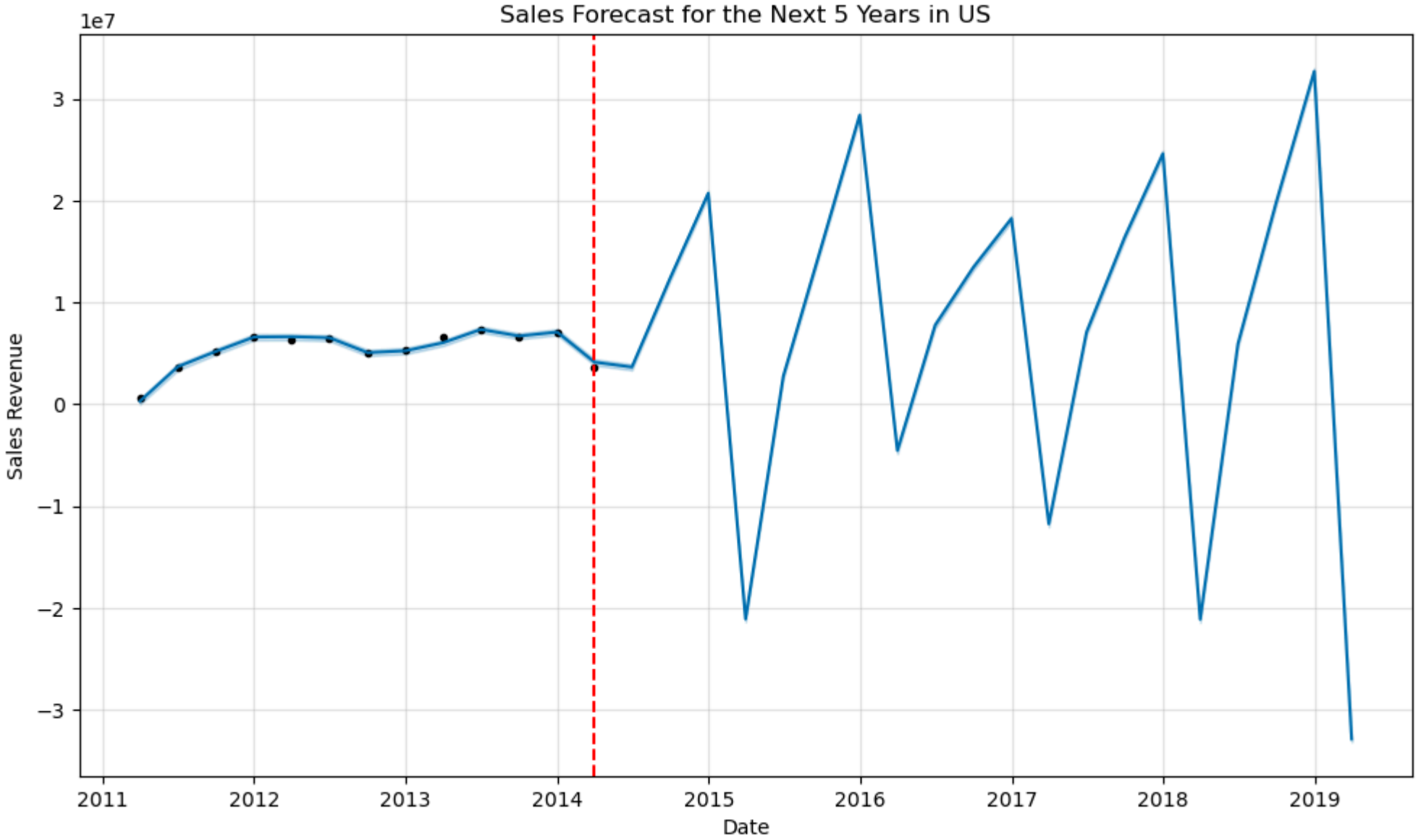
```python
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
from prophet import Prophet

# Load the sales data from the CSV file, handling encoding issues.
df = pd.read_csv('sales_data.csv', encoding='ISO-8859-1')

# Display the first few rows to verify successful loading
```

```
print("Initial Data:")
print(df.head())

# Print the columns in the DataFrame
print("\nColumns in the DataFrame:")
print(df.columns)

# Strip whitespace from the column names
df.columns = df.columns.str.strip()

# Ensure the OrderDate, DueDate, and ShipDate columns are in the correct datetime format
df['OrderDate'] = pd.to_datetime(df['OrderDate'], format='%d/%m/%Y %H:%M', errors='coerce')
df['DueDate'] = pd.to_datetime(df['DueDate'], format='%d/%m/%Y %H:%M', errors='coerce')
df['ShipDate'] = pd.to_datetime(df['ShipDate'], format='%d/%m/%Y %H:%M', errors='coerce')

# Check for any NaT values resulting from the conversion
print("\nConverted OrderDate, DueDate, and ShipDate columns:")
print(df[['OrderDate', 'DueDate', 'ShipDate']].head())

# Create a Revenue column if it doesn't exist
if 'Revenue' not in df.columns:
    if 'UnitPrice' in df.columns and 'Quantity' in df.columns:
        df['Revenue'] = df['UnitPrice'] * df['Quantity']  # Calculate Revenue
    else:
        print("Cannot calculate Revenue, 'UnitPrice' and 'Quantity' columns are required.")

# Ensure TotalDue is in the correct format
df['TotalDue'] = df['TotalDue'].replace('[\£,]', '', regex=True).astype(float)

# Add Year and YearQuarter columns for analysis
df['Year'] = df['OrderDate'].dt.year
df['YearQuarter'] = df['OrderDate'].dt.to_period('Q')  # Creates Year-Quarter period

# Convert YearQuarter to timestamp for Prophet
df['YearQuarter'] = df['YearQuarter'].dt.to_timestamp()  # Convert to timestamp

# Verify that the TotalDue column is created
print("\nUpdated DataFrame Columns:")
print(df.columns)

# Now, I'll group the sales data by SalesTerritory.CountryRegionCode and YearQuarter for forecasting
if 'TotalDue' in df.columns and 'Sales.SalesTerritory.CountryRegionCode' in df.columns:
    # Get the unique regions to loop through
    regions = df['Sales.SalesTerritory.CountryRegionCode'].unique()

    for region in regions:
```

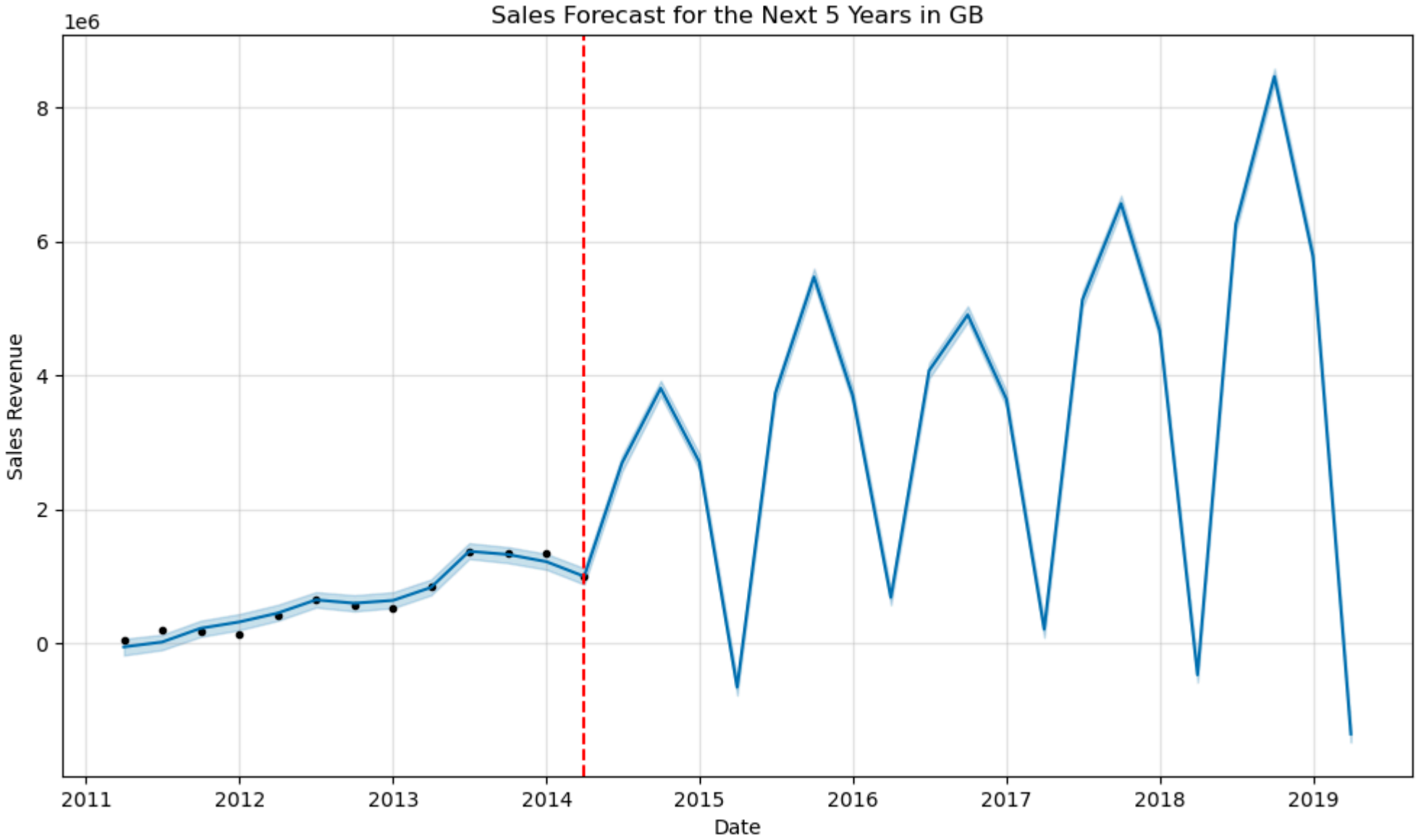
```
        # Filter data for the current region
        region_data = df[df['Sales.SalesTerritory.CountryRegionCode'] == region]

        # Aggregate sales data by YearQuarter for the current region
        quarterly_sales = region_data.groupby(['YearQuarter']).agg({'TotalDue': 'sum'}).reset_index()

        # Rename columns for Prophet
        quarterly_sales.rename(columns={'YearQuarter': 'ds', 'TotalDue': 'y'}, inplace=True)

        # Fit a Prophet model to the region's quarterly sales data
        model = Prophet(seasonality_mode='multiplicative', yearly_seasonality=True)
        model.fit(quarterly_sales)

        # Create a future dataframe that represents the next 5 years (20 quarters)
        future = model.make_future_dataframe(periods=20, freq='QE')  # Change frequency to 'QE'

        # Forecast the future sales
        forecast = model.predict(future)

        # Plot the forecast for the current region
        fig = model.plot(forecast)
        plt.title(f'Sales Forecast for the Next 5 Years in {region}')
        plt.xlabel('Date')
        plt.ylabel('Sales Revenue')
        plt.axvline(x=pd.to_datetime(quarterly_sales['ds'].max()), color='red', linestyle='--')  # Historical cutoff line
        plt.show()
else:
    print("TotalDue or Region column is not available; please check the input data.")

# Optionally, save the modified DataFrame to a new CSV file for future reference
output_file_path = 'modified_sales_data.csv'
df.to_csv(output_file_path, index=False)  # Save without the index column

print(f"\nDataFrame has been exported to '{output_file_path}'.")
```

```
Initial Data:
   Quarter  Year YearQuarter      SalesOrderID    RevisionNumber  \
0        2  2011     2011-Q2  25/08/2019 00:00  08/01/1900 00:00
1        2  2011     2011-Q2  29/08/2019 00:00  08/01/1900 00:00
2        2  2011     2011-Q2  30/08/2019 00:00  08/01/1900 00:00
3        2  2011     2011-Q2  05/09/2019 00:00  08/01/1900 00:00
4        2  2011     2011-Q2  11/09/2019 00:00  08/01/1900 00:00

          OrderDate           DueDate          ShipDate  Status  \
0  01/06/2011 00:00  13/06/2011 00:00  08/06/2011 00:00       5
1  02/06/2011 00:00  14/06/2011 00:00  09/06/2011 00:00       5
2  02/06/2011 00:00  14/06/2011 00:00  09/06/2011 00:00       5
3  04/06/2011 00:00  16/06/2011 00:00  11/06/2011 00:00       5
4  05/06/2011 00:00  17/06/2011 00:00  12/06/2011 00:00       5

   OnlineOrderFlag  ... CreditCardApprovalCode CurrencyRateID   SubTotal  \
0             True  ...         1230194Vi41919            NaN  £3,578.27
1             True  ...          230189Vi56258            NaN  £3,578.27
2             True  ...          230298Vi99064            NaN  £3,578.27
3             True  ...          830288Vi85808            NaN  £3,578.27
4             True  ...         1130295Vi31203            NaN  £3,578.27

    TaxAmt  Freight   TotalDue  Comment                               rowguid  \
0  £286.26   £89.46  £3,953.99      NaN  9310C7F0-9A84-4CE9-BE08-F700FB1AADF7
1  £286.26   £89.46  £3,953.99      NaN  F02C4CB6-A5B5-4CE4-9473-CB084E383196
2  £286.26   £89.46  £3,953.99      NaN  0DA77D6E-223E-4BC6-A2ED-43B387692C68
3  £286.26   £89.46  £3,953.99      NaN  9DE30294-9066-4988-A3AD-09A0713348E5
4  £286.26   £89.46  £3,953.99      NaN  BF5155EB-C5BE-4245-8FC4-F801DB5B052D

       ModifiedDate Sales.SalesTerritory.CountryRegionCode
0  08/06/2011 00:00                                    US
1  09/06/2011 00:00                                    US
2  09/06/2011 00:00                                    US
3  11/06/2011 00:00                                    US
4  12/06/2011 00:00                                    US

[5 rows x 30 columns]

Columns in the DataFrame:
Index(['Quarter', 'Year', 'YearQuarter', 'SalesOrderID', 'RevisionNumber',
       'OrderDate', 'DueDate', 'ShipDate', 'Status', 'OnlineOrderFlag',
       'SalesOrderNumber', 'PurchaseOrderNumber', 'AccountNumber',
       'CustomerID', 'SalesPersonID', 'TerritoryID', 'BillToAddressID',
       'ShipToAddressID', 'ShipMethodID', 'CreditCardID',
       'CreditCardApprovalCode', 'CurrencyRateID', 'SubTotal', 'TaxAmt',
       'Freight', 'TotalDue', 'Comment', 'rowguid', 'ModifiedDate',
```

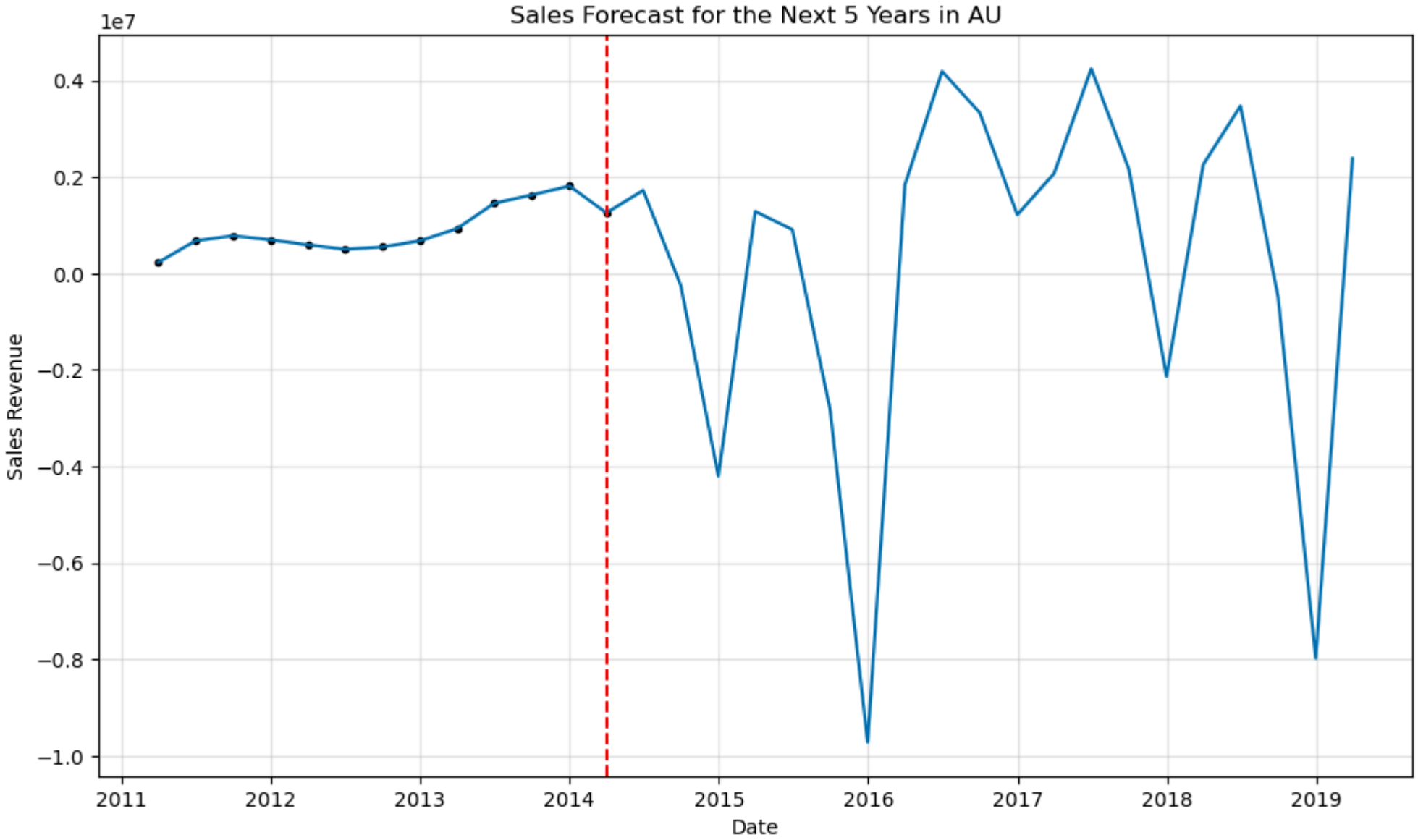
```
       'Sales.SalesTerritory.CountryRegionCode'],
      dtype='object')

Converted OrderDate, DueDate, and ShipDate columns:
   OrderDate    DueDate   ShipDate
0 2011-06-01 2011-06-13 2011-06-08
1 2011-06-02 2011-06-14 2011-06-09
2 2011-06-02 2011-06-14 2011-06-09
3 2011-06-04 2011-06-16 2011-06-11
4 2011-06-05 2011-06-17 2011-06-12
Cannot calculate Revenue, 'UnitPrice' and 'Quantity' columns are required.

Updated DataFrame Columns:
Index(['Quarter', 'Year', 'YearQuarter', 'SalesOrderID', 'RevisionNumber',
       'OrderDate', 'DueDate', 'ShipDate', 'Status', 'OnlineOrderFlag',
       'SalesOrderNumber', 'PurchaseOrderNumber', 'AccountNumber',
       'CustomerID', 'SalesPersonID', 'TerritoryID', 'BillToAddressID',
       'ShipToAddressID', 'ShipMethodID', 'CreditCardID',
       'CreditCardApprovalCode', 'CurrencyRateID', 'SubTotal', 'TaxAmt',
       'Freight', 'TotalDue', 'Comment', 'rowguid', 'ModifiedDate',
       'Sales.SalesTerritory.CountryRegionCode'],
      dtype='object')
14:08:49 - cmdstanpy - INFO - Chain [1] start processing
14:08:50 - cmdstanpy - INFO - Chain [1] done processing
```



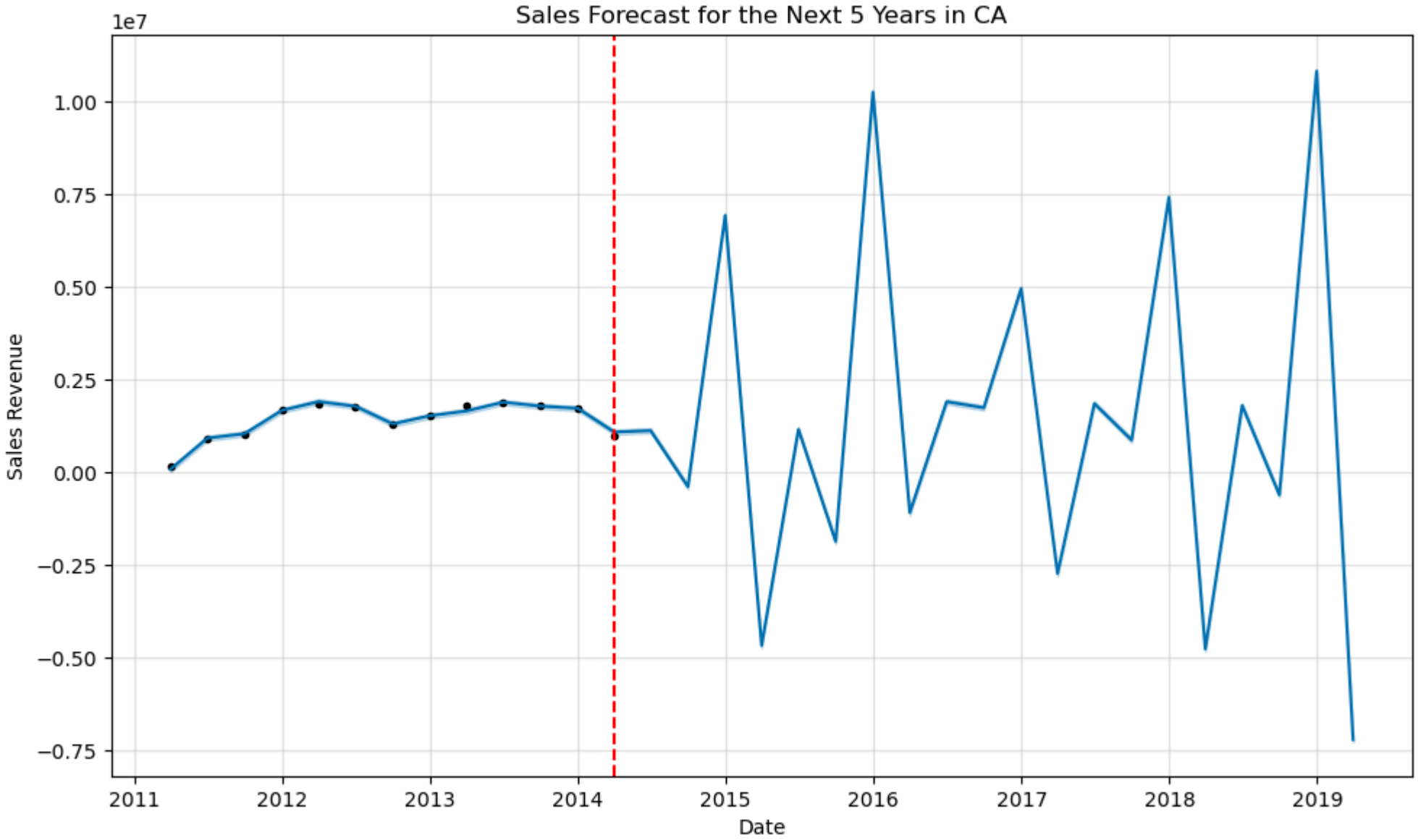

```
14:08:50 - cmdstanpy - INFO - Chain [1] start processing
14:08:50 - cmdstanpy - INFO - Chain [1] done processing
```



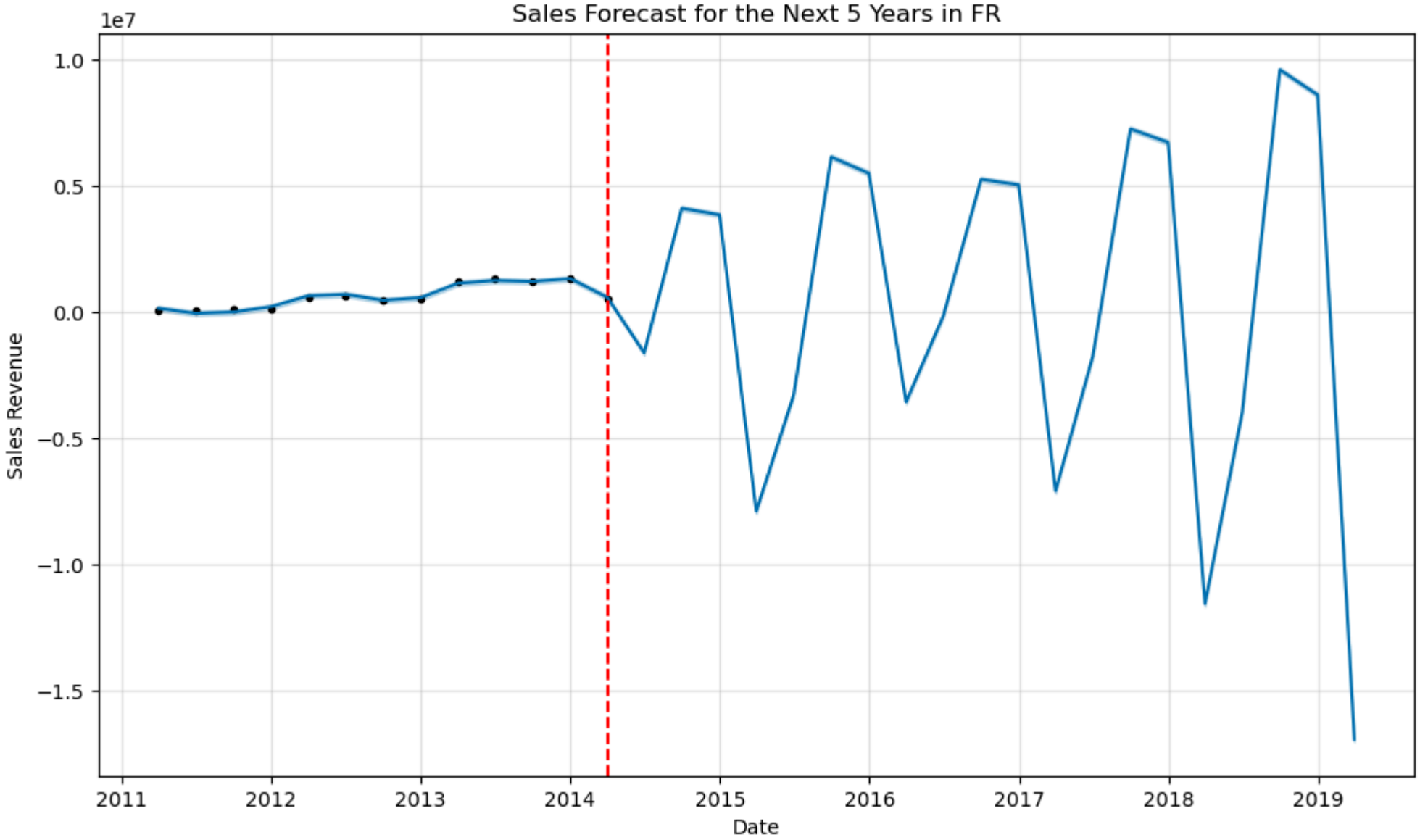

```
14:08:51 - cmdstanpy - INFO - Chain [1] start processing
14:08:52 - cmdstanpy - INFO - Chain [1] done processing
```



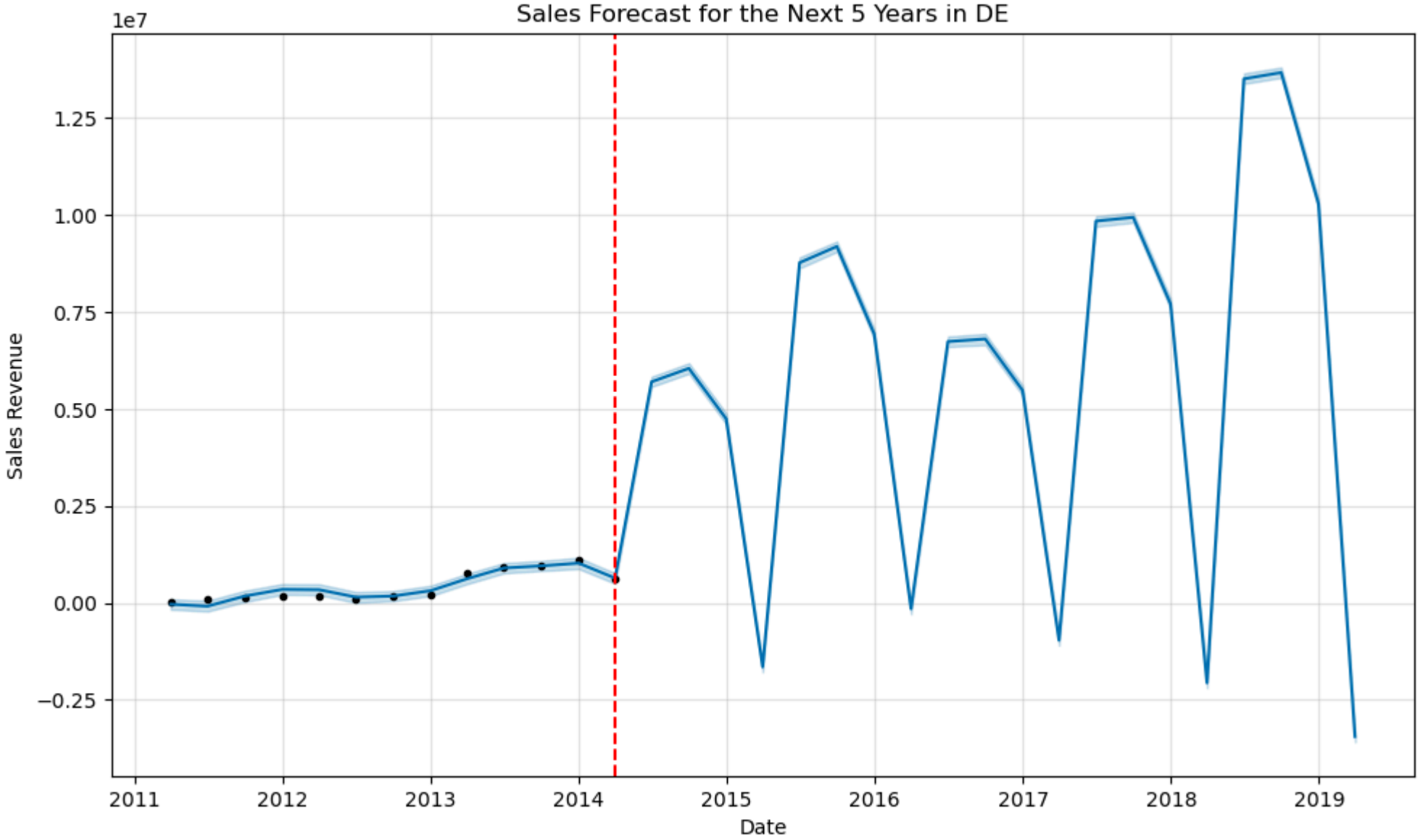

```
14:08:53 - cmdstanpy - INFO - Chain [1] start processing
14:08:53 - cmdstanpy - INFO - Chain [1] done processing
```




```
14:08:54 - cmdstanpy - INFO - Chain [1] start processing
14:08:54 - cmdstanpy - INFO - Chain [1] done processing
```




```
14:08:54 - cmdstanpy - INFO - Chain [1] start processing
14:08:54 - cmdstanpy - INFO - Chain [1] done processing
```




```
DataFrame has been exported to 'modified_sales_data.csv'.
```

In [ ]: