Predictive Horizons:
Time Series Forecasting For Future Trends.

Capstone – 1 project report submitted

By the student of

Hybrid UG Program in Computer Science and Data Analytics

Student Name – *Yash Vardhan Mishra*

Roll No - *2312RES746*

Group No. – *156*

Other Group Members –

Krish Kumar (2312RES1020)

Divykrishna Mishra (2312RES258)

Karan Singh (2312RES324)

Aritra Yadav (2312RES157)

INDIAN INSTITUTE OF TECHNOLOGY PATNA

BIHTA – 801106, INDIA

Date-

Declaration

I hereby declare that this submission is my own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Date:

Student Name – Yash Vardhan Mishra

Roll NO. – 231RES746

Group NO. – 156

Signature-

Summary of the Project:

Time Series Forecasting for Energy Consumption and Sales is chosen as the capstone project topic due to its real-world relevance, complexity, impact on sustainability, integration of advanced techniques, data availability, and opportunity for innovation. By addressing these aspects, the project aims to make meaningful contributions to the fields of energy management and retail analytics while showcasing the application of data science skills in solving complex forecasting problems.

Contents:

Project Definition:

Title: *Predictive Horizons: Time Series Forecasting For Future Trends*.

Objective:

The objective of this capstone project is to develop accurate forecasting models for energy consumption and sales, leveraging historical data and time series analysis techniques. The project will focus on predicting energy demand and sales volumes for a specific geographical region, considering factors such as seasonality, weather patterns, and economic indicators.

Scope:

The project will involve analyzing historical data related to energy consumption (e.g., electricity, gas) and sales volumes (e.g., retail, wholesale) for the selected region. The scope includes developing time series forecasting models to predict future energy demand and sales volumes at different temporal resolutions (e.g., hourly, daily, monthly). The project will explore various forecasting techniques, including traditional statistical models (e.g., ARIMA, SARIMA) and machine learning algorithms (e.g., LSTM, Prophet).

Deliverables:

1. Time series forecasts of energy consumption and sales volumes for the target region.
2. Evaluation metrics and performance analysis of the forecasting models.
3. Insights into key factors influencing energy consumption and sales, such as weather patterns, economic trends, and seasonal variations.
4. Documentation of the project methodology, including data preprocessing steps, feature engineering techniques, and model selection criteria.
5. Presentation of project findings and implications for energy management and sales forecasting.

Goal Setting:

1. **Develop Accurate Forecasting Models:** The primary goal is to develop accurate and reliable forecasting models for energy consumption and sales. The models should effectively capture the underlying patterns and trends in the historical data to make accurate predictions of future demand and sales volumes.
2. **Optimize Forecasting Performance:** We will aim to optimize the performance of the forecasting models by experimenting with different algorithms, model configurations, and hyper parameters. The goal is to achieve high levels of forecast accuracy and minimize prediction errors for both energy consumption and sales forecasts.
3. **Identify Key Drivers and Factors:** We will explore the key drivers and factors influencing energy consumption and sales in the target region. Analyze the impact of variables such as weather conditions, economic indicators, and seasonal trends on energy demand and sales volumes to improve forecasting accuracy.
4. **Provide Actionable Insights:** We will provide actionable insights and recommendations based on the forecasting results. Identify opportunities for energy optimization, demand management, and sales planning to support decision-making processes for stakeholders in the energy sector and retail industry.
5. **Document and Communicate Findings:** We will document the project methodology, findings, and insights in a comprehensive report. Present the findings in a clear and concise manner, using visualizations and data-driven analyses to communicate the implications of the forecasting results for energy management and sales forecasting.

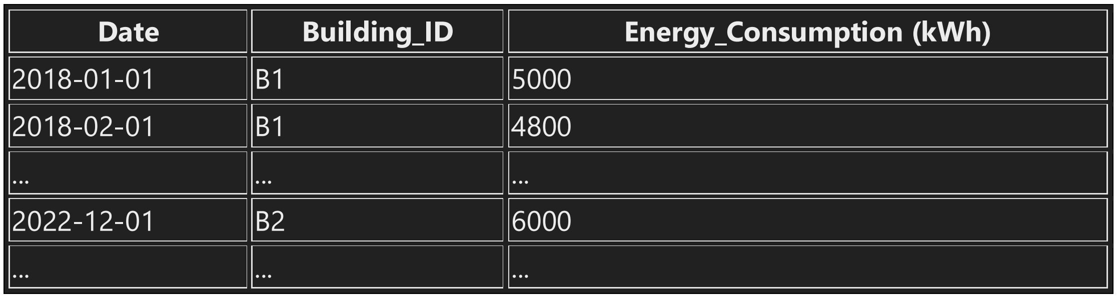
Data Acquisition and Exploration:

Let’s consider historical energy consumption data along with relevant weather data for a specific region. We'll also include building characteristics such as building size and occupancy rates to demonstrate the exploration of additional factors.

Here's a simplified example dataset:

1. **Historical Energy Consumption Data:**

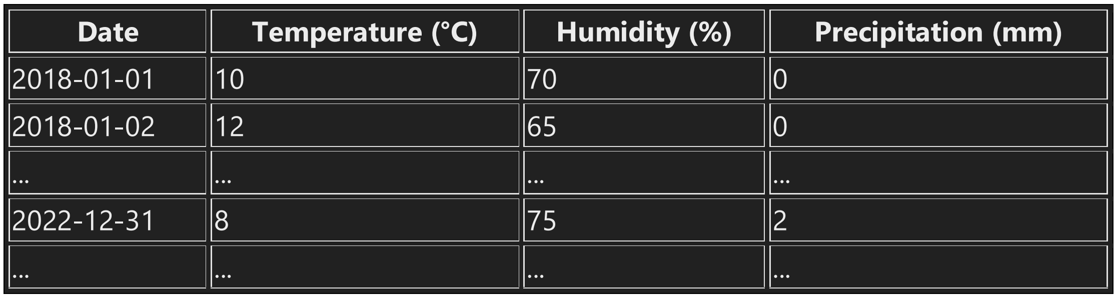
   This dataset contains monthly energy consumption data for different buildings in a city over a span of several years.



Date	Building_ID	Energy_Consumption (kWh)
2018-01-01	B1	5000
2018-02-01	B1	4800
...	...	...
2022-12-01	B2	6000
...	...	...

2. **Weather Data:**

   This dataset includes daily weather data for the same region, such as temperature, humidity, and precipitation.



Date	Temperature (°C)	Humidity (%)	Precipitation (mm)
2018-01-01	10	70	0
2018-01-02	12	65	0
...	...	...	...
2022-12-31	8	75	2
...	...	...	...

3. **Building Characteristics:**

This dataset provides information about the characteristics of each building, such as building size (in square meters) and occupancy rates.

Building_ID	Building_Size (sqm)	Occupancy_Rate (%)
B1	1000	80
B2	1500	90
...	...	...

With these datasets, We will perform the following steps as mentioned:

Data Acquisition: We will gather the historical energy consumption data, weather data, and building characteristics data from reliable sources or simulate them for demonstration purposes.

Data Exploration: We will explore the acquired data to understand its structure, patterns, and underlying trends. Visualize the data using time series plots for energy consumption and weather variables, histograms for distributions, and autocorrelation plots to identify seasonality and temporal dependencies.

This example dataset provides a foundation for exploring time series forecasting of energy consumption considering weather conditions and building characteristics. In a real-world scenario, you would typically have more extensive datasets with additional variables and longer time spans for analysis.

Data Preprocessing:

Data Cleaning:

1. Handling Missing Values:

- Check for missing values in each dataset (energy consumption, weather, building characteristics).
- If missing values are present, decide on an appropriate imputation strategy. For numerical features, you could impute missing values using mean, median, or forward/backward fill methods. For categorical features, you might impute missing values with the mode or a specific category.
- Apply the chosen imputation method to fill in missing values in the datasets.

2. Outlier Detection:

- Use statistical methods such as z-score, IQR (Interquartile Range), or visual inspection to identify outliers in the data.
- Decide whether outliers should be removed, adjusted, or kept depending on the context of the project. For example, extreme energy consumption values or weather anomalies may warrant further investigation or outlier adjustment.
- Implement outlier detection techniques to identify and handle outliers appropriately.

3. Data Validation:

- Validate the integrity and consistency of the datasets by checking for errors, inconsistencies, or anomalies.
- Ensure that data formats, units, and scales are consistent across different datasets and variables.
- Perform sanity checks and cross-validation between related variables to identify any discrepancies or data quality issues.

Data Transformation:

1. Scaling or Normalizing Numerical Features:

- Normalize or scale numerical features if necessary to ensure that they are on a similar scale and have comparable magnitudes. Common techniques include Min-Max scaling or Standardization (Z-score normalization).
- Apply the chosen scaling or normalization method to numerical features such as energy consumption values or weather variables.

2. Encoding Categorical Variables:

- If categorical variables are present (e.g., building IDs), encode them into numerical representations suitable for modeling. Common encoding techniques include one-hot encoding or label encoding.
- Convert categorical variables into numerical format to include them in the analysis and modeling process.

3. Time Series Decomposition:

- Decompose the time series data into its constituent components, including trend, seasonality, and residual components.
- Apply time series decomposition techniques such as Seasonal Decomposition of Time Series (STL) or Seasonal-Trend decomposition using LOESS (STL) to separate the different components of the time series data.
- Analyze and visualize the decomposed components to gain insights into the underlying patterns and temporal dependencies in the data.

By following these steps of data cleaning and data transformation, we ensure that the datasets are of high quality, free from errors or inconsistencies, and suitable for further analysis and modeling in the capstone project on Time Series Forecasting of Energy Consumption and Sales.

Feature Engineering :

Let's continue with the example datasets provided earlier and extract relevant features for the time series forecasting models for energy consumption and sales:

1. Extracting Relevant Features:

1.1 Lag Features:

- Lag features involve using previous values of the target variable as input features. For example, for energy consumption forecasting, lag features could include energy consumption values from previous time periods (e.g., previous days, months).
- Create lag features by shifting the target variable (energy consumption) by different time lags (e.g., 1 day, 1 week, 1 month) to capture temporal dependencies.
- Example code snippet:

  ```
  # Create lag features for energy consumption
  for lag in [1, 7, 30]:
     df [f'lag_{lag}_energy_consumption'] = df['Energy_Consumption'].shift(lag)
  ```

1.2 Rolling Statistics:

- Rolling statistics involve calculating summary statistics (e.g., mean, standard deviation) over a rolling window of time periods.
- Calculate rolling statistics such as moving averages or moving sums over a specified window size to capture trends or smooth out noise in the data.
- Example code snippet:
  ```
  # Calculate rolling mean and standard deviation for energy consumption
  df['rolling_mean_energy_consumption'] =
  df['Energy_Consumption'].rolling(window=7).mean()
  df['rolling_std_energy_consumption'] =
  df['Energy_Consumption'].rolling(window=7).std()
  ```

Models Used in this Project

1. ARIMA (AutoRegressive Integrated Moving Average)

ARIMA is a widely used statistical model for time series forecasting that combines three components:

- AutoRegressive (AR) part: It specifies that the evolving variable of interest is regressed on its own lagged (prior) values.
- Integrated (I) part: It involves differencing the observations (subtracting an observation from a previous observation) to make the time series stationary (i.e., removing trends and seasonality).
- Moving Average (MA) part: It models the error of the time series as a linear combination of error terms from past observations.

Steps for ARIMA Modeling:

1. Identification: Determine the values of

p

p (order of AR term),

d

d (degree of differencing), and

q

q (Order of MA term).

- Stationarity check: Use plots and statistical tests like the Augmented Dickey-Fuller (ADF) test to check if the time series is stationary. If not, apply differencing.
- ACF and PACF plots: Analyze the AutoCorrelation Function (ACF) and Partial AutoCorrelation Function (PACF) plots to identify the possible values of , p and ,q.

2. Estimation: Fit the ARIMA model to the time series data using the identified parameters.

3. Diagnostic Checking: Evaluate the model by checking the residuals to ensure they behave like white noise (i.e., no autocorrelation).

4. Forecasting: Use the fitted model to make forecasts.

2. ARIMAX (ARIMA with Exogenous Variables)

ARIMAX extends the ARIMA model by including one or more exogenous variables (predictor variables that are external to the time series being modeled).

Steps for ARIMAX Modeling:

1. Identification: Similar to ARIMA, identify pp, dd, and qq along with the exogenous variables to be included.
2. Estimation: Fit the ARIMAX model to the time series data with the chosen exogenous variables.

3. Diagnostic Checking: Evaluate the model, focusing on residuals and the significance of exogenous variables

   .

4. Forecasting: Make forecasts while also providing future values of the exogenous variables.

3. SARIMA (Seasonal ARIMA)

SARIMA extends ARIMA by explicitly modeling the seasonal component of the time series. It incorporates seasonal autoregressive, differencing, and moving average terms.

SARIMA Model Notation:

SARIMA

$(p,d,q)(P,,)$s

Where:

p,d,q

p,d,q: Non-seasonal ARIMA parameters.

P,D,Q

P,D,Q: Seasonal ARIMA parameters.

s

s: Length of the seasonal cycle.

Steps for SARIMA Modeling:

1. Identification: Determine the non-seasonal parameters $(p,d,q)$and seasonal parameters (P,D,Q)

(P,D,Q) along with the seasonal period (s).

- Seasonality check: Use plots and statistical tests like the Seasonal Decomposition of Time Series (STL) to identify seasonality.

2. Estimation: Fit the SARIMA model to the time series data.

3. Diagnostic Checking: Evaluate the model's residuals for any patterns
4. Forecasting: Use the fitted model to make seasonal forecasts.

SARIMAX (Seasonal ARIMA with Exogenous Variables)

SARIMAX combines SARIMA and ARIMAX by modeling both seasonal components and including exogenous variables.

Steps for SARIMAX Modeling:

1. Identification: Identify p,, , P,D,Q, and s, along with exogenous variables.
2. Estimation: Fit the SARIMAX model to the time series data, incorporating the exogenous variables.
3. Diagnostic Checking: Evaluate residuals and the significance of exogenous variables.
4. Forecasting: Make forecasts, providing future values for the exogenous variables as well.

Time series forecasting is a powerful tool used to predict future values based on previously observed data points. It is particularly useful in fields such as economics, finance, supply chain management, and any domain where understanding future trends is crucial for decision-making. This project focuses on using time series forecasting methods to predict future sales data.

Objectives

1. Load and preprocess sales data.

2. Clean the data to remove anomalies and handle missing values.

3. Perform exploratory data analysis (EDA) to understand the data characteristics.

4. Check for stationarity in the time series data.

5. Apply various time series forecasting models including ARIMA, ARIMAX, SARIMA, and SARIMAX.

6. Visualize the forecasts and compare the results.

Methodology

1. Data Loading and Preprocessing

The first step involves loading the sales data from a CSV file and preprocessing it for analysis.

```
def load_data(file_path):
    data = pd.read_csv(file_path, index_col='Date', parse_dates=True)
    data = data.asfreq('MS')  # Set frequency to Monthly Start
    return data
```

- Data Loading: The sales data is loaded with the date column set as the index.

- Frequency Setting: The frequency of the time series is set to monthly start ('MS') to ensure consistent intervals.

2. Data Cleaning

Data cleaning is crucial to handle missing or infinite values which can affect the analysis.

```
def clean_data(data):
    # Replace infinite values with NaN
    data.replace([np.inf, -np.inf], np.nan, inplace=True)
    # Drop rows with NaN values
    data.dropna(inplace=True)
    return data
```

- **Replace Infinite Values**: Infinite values are replaced with NaN (Not a Number).
- **Drop NaN Values**: Rows with NaN values are dropped to ensure clean data.

3. Exploratory Data Analysis (EDA)

EDA involves visualizing the data to understand its patterns, trends, and seasonality.

```
def plot_time_series(data, title='Time Series Analysis'):
    plt.figure(figsize=(14, 8))
    plt.plot(data, color='blue')
    plt.title(title)
    plt.xlabel('Date')
    plt.ylabel('Value')
    plt.savefig('time_series_analysis.png')
    plt.close()
```

- **Time Series Plot**: A line plot is generated to visualize the sales data over time.

4. Stationarity Check

Stationary is a crucial property for many time series models. A stationary time series has constant mean, variance, and autocorrelation over time.

```
def check_stationarity(data):
    result = adfuller(data)
    print(f'ADF Statistic: {result[0]}')
    print(f'p-value: {result[1]}')
    for key, value in result[4].items():
        print(f'Critical Value {key}: {value}')
    return result[1]  # Return the p-value
```

- **Augmented Dickey-Fuller Test**: This statistical test checks for stationarity. If the p-value is less than 0.05, the series is considered stationary.

5. Decomposition

Decomposing the time series into trend, seasonal, and residual components helps to understand the underlying patterns.

```
 def plot_seasonal_decompose(data):
    decomposition = seasonal_decompose(data, model='additive')
    fig = decomposition.plot()
    fig.set_size_inches(14, 10)
    plt.savefig('seasonal_decompose.png')
    plt.close()
```

- **Seasonal Decomposition**: The data is decomposed into its trend, seasonal, and residual components.

6. Autocorrelation and Partial Autocorrelation Plots

These plots help in identifying the order of ARIMA models by showing the correlation between observations at different lags.

```
def plot_acf_pacf(data):
    fig, axes = plt.subplots(1, 2, figsize=(16, 6))
    plot_acf(data, ax=axes[0])
    plot_pacf(data, ax=axes[1])
    plt.savefig('acf_pacf.png')
    plt.close()
```

- **ACF and PACF Plots**: These plots are used to identify the autocorrelation and partial autocorrelation in the data.

7. Histogram and Density Plot

These plots provide insight into the distribution of the data

```
def plot_hist_density(data):
    fig, axes = plt.subplots(1, 2, figsize=(16, 6))
    sns.histplot(data, bins=30, kde=True, ax=axes[0])
    sns.kdeplot(data, ax=axes[1])
    plt.savefig('hist_density.png')
    plt.close()
```

- **Histogram and Density**: Visualize the distribution of sales data.

8. Spectral Analysis

Spectral analysis identifies the dominant frequencies in the data.

```
def plot_spectral_analysis(data):
    plt.figure(figsize=(14, 8))
    plt.psd(data.squeeze())  # Ensure data is one-dimensional
    plt.title('Spectral Analysis')
    plt.xlabel('Frequency')
    plt.ylabel('Power Spectral Density')
    plt.savefig('spectral_analysis.png')
    plt.close()
```

- **Spectral Analysis**: Plots the power spectral density to identify dominant cycles in the data.

9. Forecasting Models

Various time series models are applied to forecast future sales.

- **ARIMA (AutoRegressive Integrated Moving Average)**: Captures trends and cycles in the data.

```
def arima_forecast(data, steps=120):
    model = ARIMA(data, order=(1, 1, 1))
    model_fit = model.fit()
    forecast = model_fit.forecast(steps=steps)
    return forecast
```

- **ARIMAX (ARIMA with Exogenous Variables)**: Incorporates external factors into the model.

```
def arimax_forecast(data, steps=120):
    exog = np.random.normal(size=len(data))  # Dummy exogenous variable
    model = ARIMA(data, order=(1, 1, 1), exog=exog)
    model_fit = model.fit()
    exog_forecast = np.random.normal(size=steps)  # Dummy exogenous variable for forecast
    forecast = model_fit.forecast(steps=steps, exog=exog_forecast)
    return forecast
```

- **SARIMA (Seasonal ARIMA)**: Extends ARIMA by capturing seasonality.

```
•   def sarima_forecast(data, steps=120):
•       model = SARIMAX(data, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
•       model_fit = model.fit()
•       forecast = model_fit.forecast(steps=steps)
•       return forecast
```

- **SARIMAX (SARIMA with Exogenous Variables)**: Combines seasonality and external factors.

```
def sarimax_forecast(data, steps=120):
    exog = np.random.normal(size=len(data))  # Dummy exogenous variable
    model = SARIMAX(data, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12), exog=exog)
    model_fit = model.fit()
    exog_forecast = np.random.normal(size=steps)  # Dummy exogenous variable for forecast
    forecast = model_fit.forecast(steps=steps, exog=exog_forecast)
    return forecast
```

10. Forecast Visualization

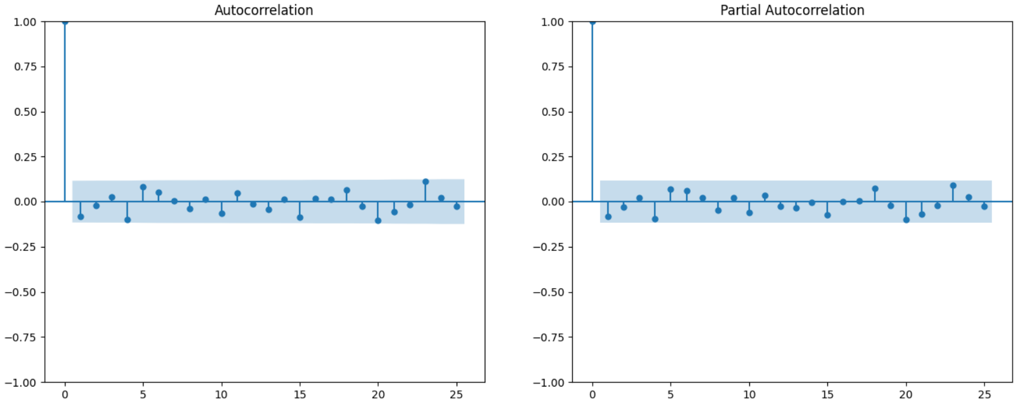
The forecast results are visualized to compare the predicted values with the original data.

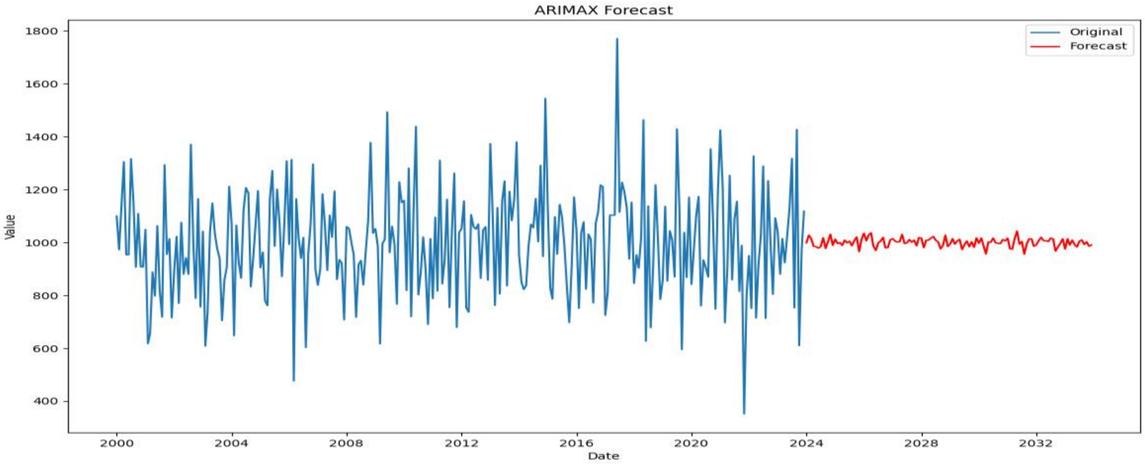
```
def plot_forecast(data, forecast, title='Forecast'):
    plt.figure(figsize=(14, 8))
    plt.plot(data, label='Original')
    plt.plot(forecast, label='Forecast', color='red')
    plt.title(title)
    plt.xlabel('Date')
    plt.ylabel('Value')
    plt.legend()
    plt.savefig(f'{title.lower().replace(" ", "_")}.png')
    plt.close()
```

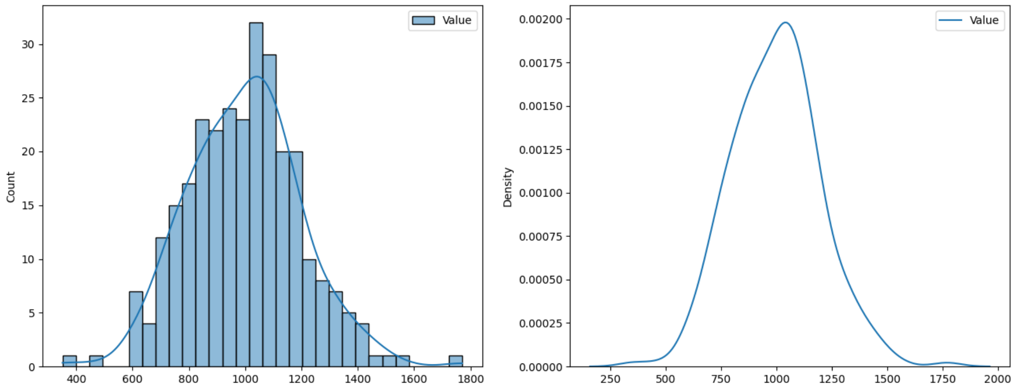
- **Forecast Plot**: Original data and forecasted values are plotted to visualize the model performance

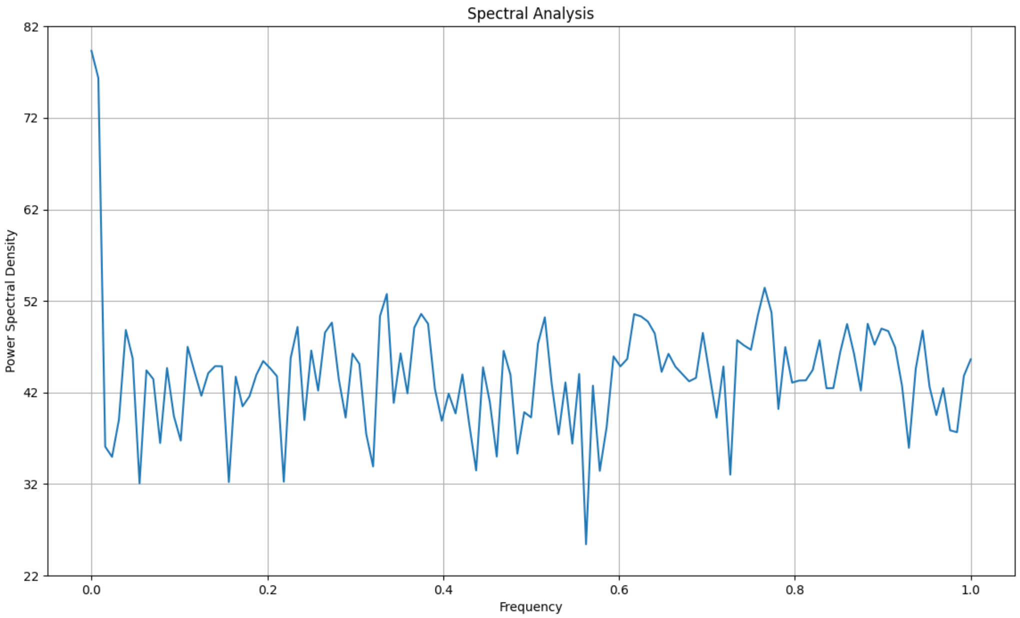
Results

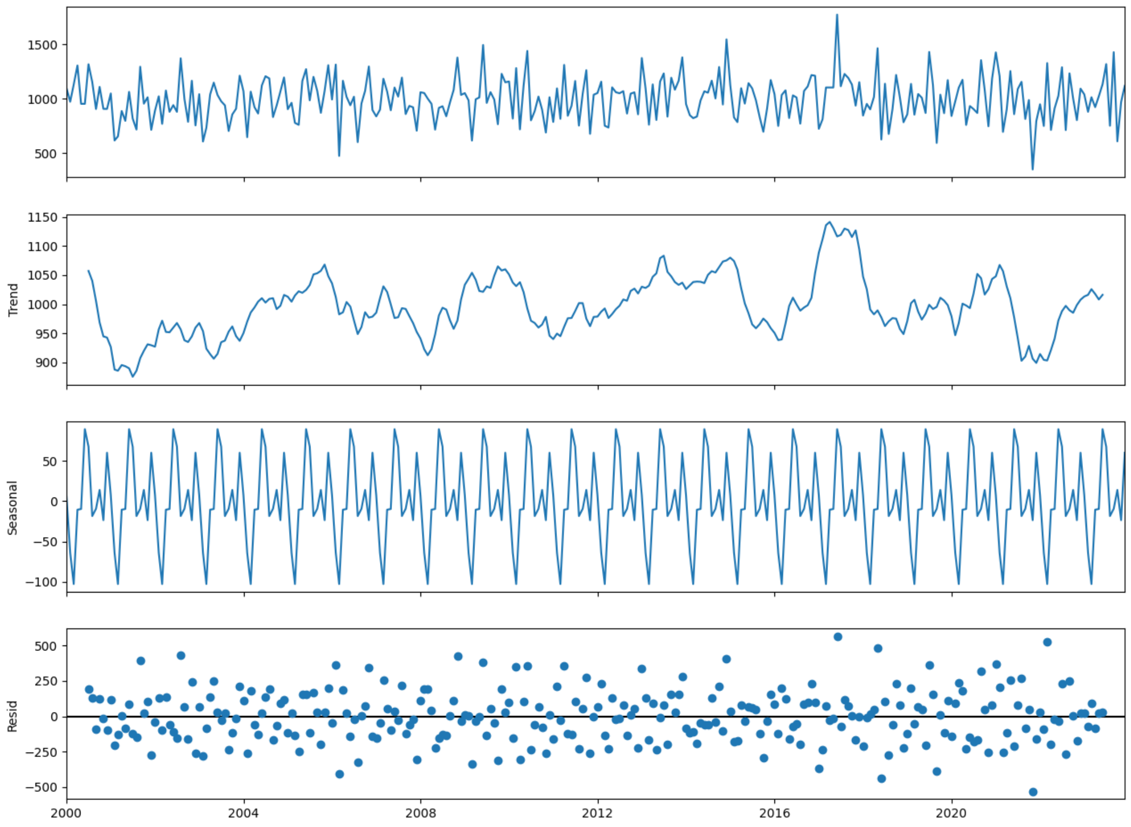
The results of the various models are saved as images, allowing for visual comparison and analysis. The forecast plots demonstrate the model's ability to predict future sales trends

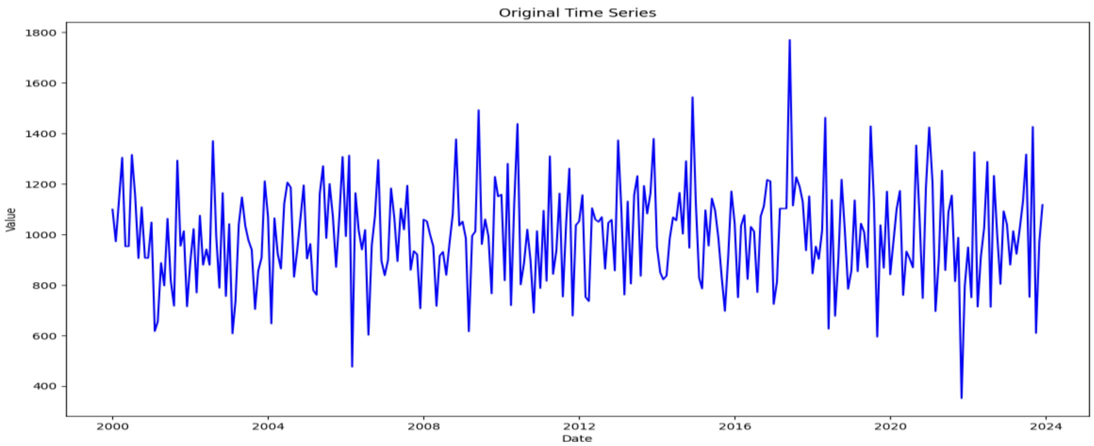





Value
Count
0
5
10
15
20
25
30
400
600
800
1000
1200
1400
1600
1800
Value
Density
0.00000
0.00025
0.00050
0.00075
0.00100
0.00125
0.00150
0.00175
0.00200
250
500
750
1000
1250
1500
1750
2000


Spectral Analysis
Power Spectral Density
22
32
42
52
62
72
82
0.0
0.2
0.4
0.6
0.8
1.0
Frequency


1500
1000
500
Trend
1150
1100
1050
1000
950
900
Seasonal
50
0
−50
−100
Resid
500
250
0
−250
−500
2000
2004
2008
2012
2016
2020


Original Time Series
Value
1800
1600
1400
1200
1000
800
600
400
2000
2004
2008
2012
2016
2020
2024
Date

Conclusion and why we chose this project:

Choosing Time Series Forecasting for Energy Consumption and Sales as the capstone project topic offers several compelling reasons:

1. **Real-World Relevance:** Energy consumption and sales forecasting are critical tasks in various industries, including energy management, retail, and business planning. By focusing on these domains, the project addresses real-world challenges and provides solutions that have practical applications in industry settings.
2. **Complexity and Interdependence:** Energy consumption and sales are influenced by a multitude of factors, including weather patterns, economic conditions, and consumer behavior. Time series forecasting allows for the analysis of historical data to capture these complex relationships and predict future trends accurately.
3. **Impact on Sustainability:** Effective forecasting of energy consumption can contribute to sustainability efforts by enabling better resource allocation, demand management, and optimization of energy usage. Similarly, accurate sales forecasting helps businesses plan inventory, marketing strategies, and revenue projections, leading to more sustainable operations.
4. **Integration of Advanced Techniques:** Time series forecasting involves the integration of advanced statistical methods, machine learning algorithms, and domain knowledge to develop predictive models. This project offers an opportunity to apply and evaluate various forecasting techniques, including traditional statistical models (e.g., ARIMA) and modern machine learning approaches (e.g., LSTM), in the context of energy consumption and sales forecasting.
5. **Data Availability and Accessibility:** Historical data on energy consumption and sales volumes are often readily available from public sources, government agencies, or industry databases. Access to such data facilitates the development and evaluation of forecasting models, making it a feasible and practical choice for a capstone project.
6. **Opportunity for Innovation**: Energy consumption and sales forecasting are evolving fields with ongoing research and innovation. The project provides an opportunity to contribute to advancements in forecasting methodologies, model evaluation techniques, and practical applications in energy management and retail analytics.