VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
SCHOOL OF INDUSTRIAL MANAGEMENT

□···☼···□


**CAPSTONE PROJECT**

# Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis

**NGUYEN DUY KHANG**

No.: 23-TA

**Ho Chi Minh City, May 2025**

**VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY**
**HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY**
**SCHOOL OF INDUSTRIAL MANAGEMENT**


**CAPSTONE PROJECT**

# Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis

**STUDENT NAME** : Nguyen Duy Khang
**STUDENT ID** : 2153429
**INSTRUCTOR** : Dr. Duong Vo Hung

**Ho Chi Minh City, May 2025**

## INSTRUCTOR'S COMMENTS

……

……

……

……

……

……..……

……

……

……

……

……

……

……

……

……

……

Ho Chi Minh City, May 2025

Signature

VIETNAM NATIONAL UNIVERSITY
HCMC UNIVERSITY OF TECHNOLOGY
Ref No. : _____/BKĐT

SOCIALIST REPUBLIC OF VIETNAM
Interdependence – Freedom - Happiness

SCHOOL OF INDUSTRIAL MANAGEMENT
DEPARTMENT: PRODUCTION & OPERATION MANAGEMENT

**CAPSTONE PROJECT**

STUDENT NAME: **NGUYEN DUY KHANG** STUDENT ID: **2153429**
SPECIALIZATION: **OPERATION AND SUPPLY CHAIN MANAGEMENT** CLASS: **CC21QCV1**

1. Title:

**Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis**

2. Thesis assignment (requirements for content and data):

- Review and compare traditional and machine learning time series forecasting methods using PyCaret's AutoML, focusing on sales forecasting and multiple metrics.
- Utilize real-world univariate daily sales data (2010–2018) from a major retail company in Bosnia and Herzegovina.
- Implement a quantitative experimental approach with PyCaret to automate model training, selection, and evaluation.

3. Date of assignment: **06/01/2025**
4. Date of completion: **08/05/2025**
5. Full name of Instructor: Advised on:
   **Duong Vo Hung, PhD** **100%**

The proposal is approved by the School/ Department
…./…/….

**HEAD OF DEPARTMENT**
(Sign and write full name)

**PRIMARY SUPERVISOR**
(Sign and write full name)

TS. Đường Võ Hùng

*FOR SCHOOL/ DEPARTMENT*
Approved by (initially examined by): …….
Department: …….
Date of defense: …….
Total mark: ……
Stored at: …….

# ACKNOWLEDGEMENT

Firstly, I would like to express my heartfelt gratitude to the School of Industrial Management at Ho Chi Minh University of Technology for offering a valuable course that has provided me with a profound understanding of my future career. This course has also given me the opportunity to apply the theoretical knowledge and skills I have acquired at the university in a practical, real-world context.

I am especially thankful to my supervisor, Dr. Duong Vo Hung, for his invaluable guidance, expertise, and insightful suggestions, all of which have been instrumental in the successful completion of my project.

Thank you sincerely.

Ho Chi Minh, May 12th, 2025

# ABSTRACT

Accurate sales forecasting is crucial for retail operations, but is challenged by data complexity, where traditional methods falter and optimal machine learning (ML) model selection is difficult. Automated Machine Learning (AutoML) frameworks like PyCaret offer streamlined model evaluation, yet their comparative effectiveness in real-world sales forecasting requires investigation. This study utilizes the PyCaret AutoML framework to compare the forecasting performance of numerous traditional statistical and ML models on real-world univariate retail sales data (Bosnia and Herzegovina, 2010-2018). Daily aggregated sales data underwent preprocessing, including interpolation and conditional deseasonalization/detrending, before systematic model training and evaluation via PyCaret. Performance and computational time were assessed across seven metrics (MASE, RMSSE, MAE, RMSE, MAPE, SMAPE). Results consistently showed the Polynomial Trend Forecaster achieving the highest accuracy across most error metrics, outperforming Exponential Smoothing, Croston, and various ML models (including tree-based ensembles). Despite preprocessing, ML models exhibited higher error rates on this dataset, while a clear trade-off between accuracy and computational speed was observed. The findings underscore the utility of AutoML for rigorous model comparison and demonstrate that, for this specific dataset, an optimized traditional model surpassed the evaluated ML approaches in forecasting accuracy.

***Keywords:*** *Sales Forecasting, Time Series Analysis, Automated Machine Learning (AutoML), PyCaret, Machine Learning, Retail Sales Data, Polynomial Trend Forecaster*

# TABLE OF CONTENTS

# List of figures

*Name of figure* *Page*

# List of tables

# List of acronyms

| Abbreviation | Full Name |
|---|---|
| ARIMA | AutoRegressive Integrated Moving Average |
| ETS | Error, Trend, Seasonal |
| MASE | Mean Absolute Scaled Error |
| RMSSE | Root Mean Squared Scaled Error |
| MAE | Mean Absolute Error |
| RMSE | Root Mean Squared Error |
| MAPE | Mean Absolute Percentage Error |
| SMAPE | Symmetric Mean Absolute Percentage Error |
| TT | Computational Time |
| AutoML | Automated Machine Learning |
| TBATS | Trigonometric Box-Cox ARMA Trend Seasonal |
| LSTM | Long Short-Term Memory |
| CNN | Convolutional Neural Network |
| RNN | Recurrent Neural Network |

# CHAPTER 1
# INTRODUCTION

## 1.1. BACKGROUND OF THE STUDY

In today's dynamic and rapidly changing business landscape, companies face increasing challenges in accurately forecasting sales. This complexity arises from the fluctuating nature of market demand, evolving industry trends, and shifting consumer behaviors. Businesses must continually adjust their sales strategies to align with economic conditions, seasonal variations, and broader market dynamics. Effective sales forecasting is essential for managing supply chains, optimizing inventory, controlling costs, and meeting customer expectations. While traditional forecasting approaches often rely on time series models and statistical techniques, they can struggle with large datasets and non-linear patterns, limiting their effectiveness in today's data-driven environment.

The significance of accurate sales forecasting must be emphasized. Studies by Jewel et al. and Saha et al. showed that poor forecasting can lead to stockouts or excess inventory, directly impacting financial performance and customer satisfaction(Jewel et al., 2024; Saha et al., 2022). Companies increasingly turn to advanced analytics and machine learning techniques in competitive markets across industries to improve their understanding of demand patterns. Although traditional models like ARIMA and exponential smoothing are widely used, they often need to capture more complex data structures and non-linear relationships. By incorporating machine learning methodologies, organizations can enhance forecast accuracy, reduce errors, and streamline supply chain operations.

Despite the advancements in machine learning and the recognized potential of these models for sales forecasting, there are significant gaps in the existing literature. Firstly, much of the research relies on simulated or laboratory-controlled data, which may not accurately reflect real-world time series complexities. For instance, Cyril et al. conducted a study about applying machine learning algorithms to forecast a modified dataset of Walmart's sales from 5/2/2010 to 26/10/2012, which acknowledges a limitation in the temporal aspect of the Walmart dataset, which may not fully reflect current market dynamics or recent changes in consumer behavior, economic conditions, or competitive landscapes (Cyril et al., 2024). Secondly, AutoML frameworks like PyCaret provide an automated approach to model selection and tuning, yet their usage in sales forecasting still needs to be explored. Several studies, including one by Sebnem et al., have focused only on predicting type 2 diabetes mellitus (T2DM) using machine learning techniques applied to phenotypic data. They used PyCaret to implement various algorithms for classification tasks rather than for forecasting(Sebnem et al., 2024). Moreover, a study by Yongjun et al. demonstrates the application of PyCaret for predictive modeling with an emphasis on sweetness analysis; it does not explicitly address forecasting methodologies or techniques within the context of time series data(Yongjun et al., 2024). Additionally, Karthika utilized PyCaret to streamline the general time series analysis process rather than for a sales forecasting study(Karthika,

2023). Finally, fine-tuning and configuring machine learning models requires a deep understanding and considerable effort. It is essential to investigate how effectively PyCaret can optimize the model training, testing, and selection process to improve sales forecast accuracy and efficiently reduce the effort of analyzing the complexities of sales data.

For the reasons above, the author selected the topic for research: “Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis.”

## 1.2. OBJECTIVES OF THE STUDY

### 1.2.1. General objective

This study aims to employ PyCaret, a low-code machine learning package in Python, to evaluate the performance of machine learning models compared to traditional forecasting methods for real-world sales prediction. The goal is to streamline the model selection and evaluation process, determining which forecasting approach—machine learning or traditional methods—is the most effective and practical for time series sales data.

### 1.2.2. Detailed objectives

**Objective 1**: Systematize traditional and machine learning methods' theoretical foundations across time series forecasting, sales forecasting, and AutoML.

**Objective 2**: Evaluate the performance of different machine learning models and traditional methods on time series data using real-world retail sales data.

**Objective 3**: Identify the most effective model based on the comparative analysis of eight metrics to enhance decision-making and operational efficiency in retail sales forecasting.

## 1.3. SIGNIFICANCE OF THE STUDY

This thesis, titled "*Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis*," addresses a crucial need in modern business operations: accurate sales forecasting. In today’s competitive and volatile markets, businesses face increasing challenges in predicting sales due to fluctuating consumer demands, seasonality, and external influences like economic shifts. Though widely used, traditional forecasting models often fall short in handling large datasets and complex non-linear relationships. This research bridges the gap by leveraging PyCaret, an Automated Machine Learning (AutoML) tool, to streamline and enhance forecasting.

PyCaret’s low-code interface democratizes the use of machine learning, allowing researchers and practitioners to experiment with diverse models without extensive programming expertise. This accessibility fosters innovation, enabling the exploration of advanced machine-learning models alongside traditional statistical techniques. By comparing their effectiveness on real-world sales data, this research provides actionable insights into model performance, aiding decision-makers in choosing optimal forecasting methods.

Furthermore, the thesis contributes to academic and practical fields by addressing gaps in AutoML's application to time series forecasting, particularly in retail sales. It evaluates how PyCaret simplifies preprocessing, automates model selection, and optimizes accuracy. These findings have broader implications for industries reliant on precise demand forecasting, from supply chain management to inventory control.

By emphasizing model efficiency and usability, this research not only advances theoretical knowledge but also empowers businesses to make data-driven decisions, reduce costs, and enhance customer satisfaction, showcasing the transformative potential of AutoML in practical applications.

## 1.4. SCOPE OF THE STUDY

### 1.4.1. Geographical scope

Actual daily sales data is utilized within a production environment at one of the biggest retail enterprises in Bosnia and Herzegovina.

### 1.4.2. Time scope

Secondary data: available daily sales data are from 2010 to 2018.

### 1.4.3. Content scope

Due to the limited computational resources and academic purposes, this study concentrates on univariate time series forecasting by leveraging PyCaret's AutoML capabilities to streamline model selection and evaluation, optimizing computational efficiency. The focus on univariate models limits the analysis to single-variable predictions, which may not account for external influences like promotions or macroeconomic factors. The analysis combines traditional statistical models and advanced machine learning algorithms focusing on preprocessing techniques such as deseasonalization and detrending. These techniques enhance the models' ability to capture underlying patterns by removing seasonal and trend components from the dataset. The performance of these models will be rigorously evaluated across seven metrics, thereby providing insightful and practical implications for retail data analysis.

### 1.4.4. Research object

Research object: The time series models used for sales forecasting for the dataset encompass traditional and machine learning methodologies. The primary objective was to compare and assess their effectiveness in predictive accuracy in actual retail data using the PyCaret tool.

Research subject: The sales data of a retail company in Bosnia and Herzegovina. This includes the historical daily sales records that will be used to apply and evaluate various time series forecasting models (traditional and machine learning).

# CHAPTER 2
# LITERATURE REVIEW

## 2.1. THEORETICAL FRAMEWORK

### 2.1.1. Time series

A time series is defined as a collection of data points that are organized in chronological order. The data points are consistently spaced over time, indicating that they are recorded at regular intervals such as hourly, minutely, monthly, or quarterly. Common examples of time series include the closing prices of stocks, household electricity consumption, and outdoor temperature readings(Peixeiro, 2022).

Qiao et al. also stated that time series models are utilized for analyzing observations that are equally spaced in time, and they include various methods such as moving average smoothing, exponential smoothing, and classic models like autoregressive and moving average models. These models help in understanding and predicting trends in data over time(Qiao et al., 2024).

### 2.1.2. Statistical forecasting

Statistical forecasting usually consists of two methods: time series forecasting and predictive forecasting. Time-series forecasting involves predicting future values based on previously observed values in a dataset over time, accounting for temporal dependencies. Predictive forecasting, while similar, focuses on using various predictive models to forecast outcomes, which may not necessarily be time-dependent(Zhiyuan et al., 2023).

Due to the scope of content, the author will mainly use time series forecasting approaches to illustrate the objective of this experimental work.

#### *2.1.2.1. Time series forecasting*

Time series forecasting is the method of predicting future values of a model by reviewing its past data. It has been verified that the volatility activation function is functional on time series(Kayim & Yilmaz, 2022).

Time series forecasting models are simple to understand and set up with limited data. Most of the managers in the industry still lean on these models to forecast their future values(Vandeput, 2023).

#### *2.1.2.2. Time series vs regression forecasting*

Time series and regression forecasting are two distinct approaches for modeling and forecasting, each with unique methodologies and applications. Time series analysis focuses on understanding and predicting data points collected or recorded at successive points in time. It emphasizes the temporal ordering of data and identifies patterns such as trends,

seasonality, and cycles, which are crucial for accurate forecasting. Techniques like ARIMA (autoregressive integrated moving average) and exponential smoothing are commonly used in time series analysis to model stationary and non-stationary data, respectively(Fatoumata & Christine, 2021a; Terence, 2019; Umesh Kumar et al., 2024). Time series models are particularly adept at handling data with temporal dependencies. They are often used in economics, finance, and environmental science for forecasting variables such as sales, stock prices, and weather conditions(Dimitrios et al., 2023; Zhenyu et al., 2021).

In contrast, regression forecasting is a statistical method that models the relationship between a dependent variable and one or more independent variables. It is not inherently concerned with the temporal order of data but rather with the strength and form of the relationship between variables. Regression models, such as linear regression, Bayesian regression, and machine learning-based regression techniques like Random Forest and Gradient Boosting Machines, are used to predict outcomes based on input features(Cagatay et al., 2019; "Statistical forecasting—regression and time series analysis," 2022). These models are versatile and can be applied to various problems, including when the data is not time-dependent, such as predicting customer behavior or product sales based on various predictors(Cagatay et al., 2019; Dimitrios et al., 2023).

A key difference between the two approaches is their handling of data characteristics. Time series models are specifically designed to capture temporal patterns and are often more suitable for datasets with clear time-dependent structures. They require preprocessing steps like decomposition to separate deterministic components from stochastic ones, ensuring that the model accurately captures the underlying temporal dynamics(Fatoumata & Christine, 2021a, 2021b). Regression models, on the other hand, are more flexible regarding input data and can incorporate a wide range of features, making them suitable for complex datasets where multiple factors influence the outcome("A comprehensive evaluation of statistical, machine learning and deep learning models for time series prediction," 2022). Ultimately, the choice between time series and regression methods depends on the specific characteristics of the data and the forecasting objectives, with time series models excelling in scenarios with temporal solid dependencies and regression models offering robust performance across diverse datasets(Dimitrios et al., 2023).

In this study, the author acknowledges certain limitations based on the distinctions highlighted in the literature between time series and regression analysis. Given the academic scope and computational constraints, this research exclusively focuses on univariate time series forecasting, thus omitting multivariate regression analysis and exploring external predictors that could impact retail sales. Furthermore, while time series models excel in capturing temporal dependencies, this study limits itself to PyCaret's AutoML capabilities, which may restrict the customization options available for advanced time series models. The analysis is confined to retail sales data, meaning findings may not generalize to other sectors or datasets without temporal structures. This study focuses only on a preselected range of models and evaluation metrics, potentially overlooking other approaches that may offer alternative insights for sales forecasting.

#### *2.1.2.3. Univariate time series data*

Univariate time series data is the most popular temporal data type, where a single numeric observation is recorded sequentially over equal periods. Only the variable observed and its relation to time is considered in this analysis(Kulkarni et al., 2023).

According to Feng et al., univariate time series forecasting involves predicting future values based solely on historical data of a single variable(Feng & Joey, 2023). Key characteristics of univariate time series include the ability to identify patterns, trends, and seasonal variations within the data("Construction Time Series Forecasting Using Univariate Time Series Models," 2023). Univariate time series is distinct from multivariate time series, as it does not involve associated factors or multiple variables, making it less complex and limiting the information available for analysis(Geeta, 2023).

Univariate time series techniques are applicable in various fields, including finance, retail, and economics, where forecasting based on historical data of a single variable is essential for decision-making(Kabbilawsh et al., 2022). Rik van et al. demonstrated how univariate time series can be effectively optimized for business objectives through active learning, showcasing its relevance in IT monitoring and anomaly detection fields(Rik van & Ger, 2022).

Generally, univariate time series data is widely used for forecasting when only a single variable's historical values, recorded at equal intervals, are available. This approach focuses on the relationship between the variable and time, enabling the identification of patterns, trends, and seasonal variations within the data. Unlike multivariate time series, which incorporates multiple variables, univariate time series analysis is more straightforward but provides limited information. It is commonly applied across fields like finance, retail, and economics for decision-making purposes, and its effectiveness has been demonstrated in applications such as IT monitoring and anomaly detection.

#### *2.1.2.4. Elements of a time series*

Time series is usually divided into trend, seasonality, and residuals. According to Vandeput, the demand components consist of level, trend, and seasonality(Vandeput, 2023). However, Lazzeri classifies time series elements into four categories: long-term movement or trend, seasonal short-term movements, cyclic short-term movements, and random or irregular fluctuations. According to him, seasonal movement is a known and fixed period, while cyclic variations are repeated patterns in a non-fixed time period(Lazzeri, 2020). The action of visualizing those components is the so-called seasonal decomposition. Seasonal decomposition is an effective tool for time series analysis; however, its conventional application may not produce precise results when the time series is characterized by amplitude outliers and extended gaps(Kayim & Yilmaz, 2022).

In summary, the author will classify this study's time series into three parts: trend, seasonality, and residuals.

*2.1.2.4.1. Trend*

According to Peixeiro, trends in time series analysis can be characterized as gradual shifts, illustrating the extent of increases or decreases over a specified period(Peixeiro, 2022). Moreover, the concept of trends in data, along with a novel neural network methodology for forecasting future time series, facilitates the extraction of two datasets—the trend and the residual. This creates two distinct learning sets for training (Yi et al., 2019). The direction of the trend—whether upward or downward—is often influenced by broader factors such as economic conditions, technological advancements, or demographic changes.

*2.1.2.4.2. Seasonality*

Ghysels et al. defined seasonality as the systematic intra-year movement in economic time series data, though not necessarily regular. It is caused by changes in weather, the calendar, and the timing of decisions, which directly or indirectly influence production and consumption decisions made by economic agents (Ghysels et al., 2006).

Nicholas mentions in his book that seasonality is caused by many different factors, namely repeated promotions or holidays of the year(Vandeput, 2023). He characterized seasonality as the variation in demand occurring across recurring cycles, including daily, weekly, or yearly intervals.

*2.1.2.4.3. Residuals*

Residuals are typically known as the “white noise” in time series analysis. According to Aloorravi, residuals refer to discrepancies or errors in time series data that persist after the removal of both trend and seasonal components(Aloorravi, 2024).

They are essential in assessing how well a model captures the underlying patterns in the data. When analyzing residuals, one key goal is to ensure that they are randomly distributed without any discernible patterns. If residuals are random, this suggests that the model has successfully captured the core structure of the time series. However, suppose there are clear patterns, such as trends or seasonality. In that case, it indicates that the model has not fully accounted for all the data’s components, implying the need for further refinement or a more suitable model.

### 2.1.3. Machine learning in time series forecasting

In their study, Vaishali et al. introduced machine learning, a subfield of artificial intelligence that focuses on developing algorithms that enable computers to learn from data without explicit programming. It utilizes statistical models to analyze data, identify patterns, and make predictions or decisions based on the data(Vaishali & Shiv Kumar, 2024). Machine learning was known from the early 1940s to the 1950s. The concept of “machine learning” was known as a mathematical model of neural networks by Walter

Pitts and Warren McCulloch in 1943. In 1952, Arthur Samuel wrote the first computer learning program for checkers, marking a significant step in machine learning(Firican, 2022).

The 1990s marked the introduction of Artificial Neural Networks (ANNs) to time series forecasting, allowing for modeling nonlinear relationships. However, early ANNs struggled with sequential data due to their inability to retain information from previous time steps. Recurrent Neural Networks (RNNs) were developed to address this, introducing feedback loops that allowed information to persist across time steps, making them suitable for time series data. Despite this advancement, RNNs faced issues like the vanishing gradient problem, which hindered their ability to learn long-term dependencies. In 1997, Long Short-Term Memory (LSTM) networks were introduced to overcome these limitations, incorporating mechanisms to retain information over extended periods and effectively capturing long-term dependencies in time series data. The 2010s witnessed significant progress in deep learning architectures for time series forecasting, with Convolutional Neural Networks (CNNs) and Temporal Convolutional Networks (TCNs) being adapted to capture temporal patterns. Additionally, attention mechanisms and Transformer models, initially developed for natural language processing, were applied to time series data, enhancing the ability to model complex dependencies(Casolaro et al., 2023).

In general, Machine learning (ML) has significantly advanced time series forecasting by enabling the modeling of complex, nonlinear patterns in sequential data(Li & Law, 2024). Traditional statistical methods, while foundational, often fall short in capturing intricate dependencies, especially in non-stationary and high-dimensional datasets(Kontopoulou et al., 2023). Integrating ML techniques, such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, has enhanced the accuracy and robustness of forecasts across various domains, including finance, healthcare, and meteorology(Casolaro et al., 2023). This evolution underscores the critical role of ML in developing sophisticated forecasting models that adapt to the complexities of real-world data(Liu & Wang, 2024).

### 2.1.4. Sales forecasting

Afiqah Bazlla et al. demonstrated that sales forecasting is the process of estimating future sales and revenue for a business, which is essential for effective inventory management, cash flow, and growth planning(Afiqah Bazlla Md et al., 2023).

Mee Tyng YooSales emphasizes that sales forecasting is a practical application of time series prediction that helps enterprises identify and utilize information to reduce costs and maximize profits("A Sales Forecasting Model for Coatings Industry via Econometric Models," 2023).

Sales forecasting is predicting future values based on known time series data. It involves selecting a forecasting approach that minimizes forecast errors, as suggested by Daifeng et al.(Daifeng, Xin, et al., 2023).

According to Asher et al., sales forecasting is the process of estimating future sales revenue based on historical data and market analysis(Asher et al., 2014). The author expresses agreement with the proposed ideas presented in this study and intentionally chooses to utilize this specific terminology to enhance clarity and precision in the discussion.

As highlighted in the "*Demand forecasting model for time-series pharmaceutical data using shallow and deep neural network model*" article, sales forecasting aims to scientifically assess future demand for pharmaceutical products, enabling companies to develop effective sales and marketing strategies. Accurate demand forecasting helps stabilize production against demand fluctuations, identify optimal times for marketing campaigns, and improve overall decision-making in the pharmaceutical industry. By utilizing various forecasting models, companies can better predict sales trends and enhance their competitive edge in the global market(Rathipriya et al., 2022).

The study by Robert et al. identifies several limitations and challenges in sales forecasting, including data quality issues, the unpredictability of consumer behavior, and external factors such as economic fluctuations. Additionally, integrating advanced forecasting techniques can be hindered by a lack of skilled personnel and resistance to change within organizations. These challenges can lead to forecast inaccuracies, ultimately affecting inventory management and overall business performance. As Robert et al. noted, addressing these limitations is crucial for improving sales forecasts' reliability.(Robert et al., 2019).

In summary, sales forecasting is vital for estimating future sales and revenue and managing inventory, cash flow, and growth planning. It involves using time series data to predict future values and selecting methods that minimize errors, helping reduce costs and maximize profits. By leveraging historical data and market analysis, sales forecasting supports strategic planning across industries, optimizing production, marketing, and decision-making. However, data quality issues, unpredictable consumer behavior, economic fluctuations, and organizational resistance to advanced techniques can impact forecast accuracy and overall business performance.

### 2.1.5. Automated Machine Learning

#### *2.1.5.1. Introduction to Automated Machine Learning*

Automated machine learning (AutoML) refers to automating the end-to-end process of applying machine learning to real-world problems, enabling high-performance machine learning pipelines with minimal user effort. According to Joshua et al., the primary purposes of AutoML include simplifying the model development process, making machine learning accessible to non-experts, and enhancing efficiency in tasks such as model selection, hyperparameter tuning, and feature engineering(Joshua et al., 2021).

The study by Dakuo et al. pointed out that AutoML, or automated machine learning, refers to techniques that automate the end-to-end process of applying machine learning to real-world problems. Its primary purposes include simplifying the data science lifecycle by

automating laborious tasks such as data preprocessing, algorithm selection, and model training. This allows data scientists and domain experts to focus on higher-level tasks, ultimately improving productivity and the quality of models produced(Dakuo et al., 2021).

In conclusion, AutoML, or automated machine learning, streamlines the entire machine learning process, making it more accessible to non-experts and significantly reducing the manual effort required in tasks like model selection, hyperparameter tuning, and data preprocessing. By automating these steps, AutoML enables faster, more efficient model development, allowing data scientists and domain experts to focus on higher-level insights, ultimately improving both productivity and the quality of the models produced.

#### *2.1.5.2. The Importance of AutoML in Time Series Forecasting*

The key advantages of using AutoML in time series forecasting include improved efficiency by combining multiple feature extraction methods and classifiers, resulting in a more robust and lightweight solution. Revin et al. underlined that the evolutionary algorithm used for pipeline generation automates the classification process, making it suitable for industrial data processing. Additionally, AutoML facilitates modeling and studying the properties of natural processes described as time series, enhancing the overall modeling quality compared to traditional machine learning approaches(Revin et al., 2023).

#### *2.1.5.3. Overview of PyCaret's AutoML Framework*

##### *2.1.5.3.1. Introduction to PyCaret*

PyCaret is an open-source, low-code machine-learning library in Python that automates the machine-learning workflow. It is designed to simplify the process of training and deploying models, providing a consistent API across various modules such as classification, regression, clustering, and time series forecasting("PyCaret — pycaret 3.0.4 documentation,").

Key features of PyCaret include:

- **Ease of Use**: Simplifies complex tasks with minimal code.
- **Model Agnostic**: Supports a wide range of algorithms from different libraries.
- **Interoperability**: Integrates seamlessly with popular Python libraries like pandas, scikit-learn, and matplotlib.
- **Automation**: Automates model selection, hyperparameter tuning, and ensembling.

##### *2.1.5.3.2. Comparison with Other AutoML Frameworks*

**Table 2.1** Comparison of some AutoML frameworks in Python

| Framework | Key Features | Time Series Support | Ease of Use | Algorithm Coverage | Deployment Capabilities | Reference |
|---|---|---|---|---|---|---|
| **PyCaret** | Low-code interface, model selection, feature engineering, ensembling | **Yes** | High (simple API) | Wide range, including statistical and ML models | Model export, MLflow integration | (Ali, 2020) |
| **H2O AutoML** | Scalable algorithms, ensemble stacking, model explainability | Limited (beta) | Moderate | GBMs, GLMs, Deep Learning, Stacked Ensembles | MOJO and POJO for Java deployment | ("H2O AutoML: Automatic Machine Learning — H2O 3.46.0.6 documentation,") |
| **Auto-Keras** | Neural Architecture Search, deep learning focus | No | Moderate | Deep learning models (CNNs, RNNs) | Model export as Keras models | (Jin et al., 2023) |
| **TPOT** | Genetic programming, pipeline optimization | No | Moderate | Scikit-learn compatible models | Exports Python code for pipelines | (Le et al., 2020) |

*Source: Author*

*2.1.5.3.3. Advantages of using PyCaret for time series forecasting*

PyCaret offers a user-friendly interface that simplifies time series forecasting by automating complex machine-learning tasks. Its high-level API enables users to implement forecasting models with minimal code, making it accessible to those with limited programming experience. This ease of use allows researchers to focus on analysis and interpretation rather than the intricacies of model development.

Moreover, PyCaret automates data preprocessing steps such as handling missing values, scaling, and transformation, ensuring that data is appropriately formatted for modeling. This automation reduces the potential for errors and inconsistencies, streamlining the workflow and saving valuable time in the data preparation phase. In addition, the tool includes automated feature engineering functionalities that generate lag features, rolling statistics, and time-based features. These capabilities enhance model performance by

capturing essential data patterns and trends without extensive manual intervention.

The library provides various time series models, including classical statistical methods such as ARIMA and modern machine learning algorithms like Prophet and Auto-ARIMA. This extensive model selection allows researchers to experiment with multiple approaches efficiently, facilitating the identification of the most suitable model for their specific datasets.

PyCaret is designed to handle large datasets efficiently by integrating optimized back-end libraries such as scikit-learn, XGBoost, and LightGBM. This integration enables fast computation and scalability, making it suitable for extensive time series data commonly encountered in academic research.

Despite its high level of automation, PyCaret allows for customization at various stages of the modeling process. Users can adjust model parameters, select specific features, and customize evaluation metrics. This flexibility enables researchers to tailor the tool to their particular research needs, enhancing the relevance and accuracy of their forecasting models.

Furthermore, PyCaret integrates with popular data science tools such as Pandas for data manipulation and Matplotlib for visualization. It supports integration with Jupyter notebooks and can be incorporated into existing Python workflows, enhancing its utility in an academic setting.

Finally, a growing community of users and contributors supports PyCaret. Comprehensive documentation and active forums provide assistance and resources, which are invaluable for troubleshooting and learning advanced features. This community support facilitates continuous learning and problem-solving.

#### *2.1.5.3.4. Disadvantages of using PyCaret for time series forecasting*

While PyCaret is user-friendly for basic tasks, leveraging its advanced customization features may require a deeper understanding of the underlying algorithms and parameters. This learning curve can pose a barrier for users needing a more extensive machine-learning experience. Moreover, the tool's automated processes may abstract the inner workings of models, potentially limiting insights into how predictions are generated. This abstraction could be a limitation for academic research that requires detailed model interpretability. Additionally, PyCaret relies on multiple external libraries, which may lead to compatibility issues or necessitate additional installations. This dependency could complicate the setup process in specific environments.

In conclusion, PyCaret offers significant advantages for time series forecasting in academic research. Its ease of use, diverse model selection, and automated feature engineering streamline the modeling process, allowing researchers to focus on analysis and interpretation. The tool's scalability and integration capabilities make it a practical choice for handling extensive datasets and complex workflows. While there are disadvantages, such as the potential learning curve for advanced customization and limited model interpretability, the benefits of using PyCaret outweigh these limitations.

## 2.2. TIME-SERIES FORECASTING MODELS

**Table 2.2** A comprehensive summary of the advantages and disadvantages of time series forecasting models

| Models | Advantages | Disadvantages |
|---|---|---|
| Naïve Forecaster | **Simplicity and Ease of Use:** The naive method is extremely simple to implement, requiring no complex calculations or parameter tuning. This makes it accessible for users with limited statistical or computational expertise(Svetunkov & Petropoulos, 2018).<br>**Benchmarking:** It serves as a useful baseline for evaluating the performance of more complex forecasting models. By comparing the accuracy of advanced models against the naive method, researchers can determine if the added complexity of a model is justified(Reich et al., 2016).<br>**Effective in Stable Environments:** The naive method can perform surprisingly well in situations where the time series data is stable or exhibits a strong trend. For instance, in steady-state models, the variance of forecast error using simple methods like the naive approach is often comparable to | **Lack of Adaptability:** The naive method does not account for seasonality, trends, or any other patterns in the data. This can lead to poor performance in time series with these characteristics(Liu et al., 2021).<br>**High Error in Volatile Series:** In time series with high volatility or noise, the naive method can result in significant forecast errors, as it does not incorporate any smoothing or error correction mechanisms(Ristanoski et al., 2013).<br>**No Underlying Statistical Model:** Unlike methods such as ARIMA or exponential smoothing, the naive method lacks a statistical foundation, which limits its ability to provide insights into the underlying data-generating process(Svetunkov & Petropoulos, 2018). |

| | | |
|---|---|---|
| | more sophisticated methods(Johnston et al., 1999). | |
| Seasonal Naïve Forecaster | **Simplicity and Ease of Use**: The Seasonal Naïve Forecaster is one of the simplest forecasting methods available. It requires minimal computational resources and is easy to understand and implement, making it accessible for organizations without advanced IT systems(Borucka, 2023).<br>**Effective for Strong Seasonal Patterns**: This method is particularly effective when the time series data has strong and consistent seasonal patterns. It can provide accurate forecasts in such scenarios without the need for complex modeling(Borucka, 2023; Giacalone et al., 2020).<br>**No Need for Parameter Estimation**: Unlike more complex models, the Seasonal Naïve Forecaster does not require parameter estimation or model fitting, which can be advantageous when data is limited or quick forecasts are needed(Borucka, 2023). | **Inability to Capture Trends**: The method does not account for trends or changes in the level of the time series over time. This can lead to inaccurate forecasts if the underlying data exhibits a trend(Borucka, 2023).<br>**Sensitivity to Data Anomalies**: The method can be sensitive to anomalies or outliers in the data, as it relies heavily on past observations. This can result in poor forecasting performance if the historical data contains irregularities(Li & Hinich, 2002).<br>**Limited to Short-Term Forecasting**: The Seasonal Naïve Forecaster is generally more suitable for short-term forecasting. Its performance may degrade over longer forecasting horizons, especially if the seasonal pattern changes(Giacalone et al., 2020).<br>**Lack of Flexibility**: The method is rigid and does not adapt to changes in the seasonal pattern or other external factors that might influence the time series(Kunst & Franses, 1998). |

| | | |
|---|---|---|
| Polynomial Trend Forecaster | **Flexibility and Adaptability**: Polynomial Trend Forecaster can adapt to the local level of smoothness in data, making it more flexible than traditional methods like smoothing splines. This adaptability allows it to capture complex patterns in time series data effectively(Ryan, 2014).<br>**High Resolution and Parsimony**: The method provides high-resolution and parsimonious estimates, meaning it can achieve theoretically unlimited resolution with fewer parameters. This is particularly beneficial in modeling nonstationary signals with time-varying amplitudes(Guotong et al., 1996).<br>**Noise Tolerance**: Polynomial Trend Forecaster is robust to various types of noise, including non-Gaussian and Gaussian noise, which enhances its reliability in real-world applications(Shamsunder et al., 1995). | **Overfitting Risk**: The flexibility of polynomial models can lead to overfitting, especially when the degree of the polynomial is too high relative to the data size. This can result in poor generalization of unseen data(Ryan, 2014).<br>**Computational Complexity**: Estimating polynomial trends can be computationally intensive, particularly for high-degree polynomials or large datasets. This complexity can limit the method's scalability(Craigmile et al., 2005).<br>**Confounding with Stochastic Fluctuations**: Polynomial trends can be confounded with low-frequency stochastic fluctuations, making it challenging to distinguish between deterministic trends and stochastic components in the data(Craigmile et al., 2005). |
| ARIMA | **Handling Seasonality and Autocorrelation**: ARIMA models are adept at managing seasonality and autocorrelation, making them suitable for evaluating interventions in health policies and other fields where these | **Complexity in Model Selection**: Selecting the appropriate ARIMA model can be complex, requiring careful consideration of model parameters and estimation methods to |

| | | |
|---|---|---|
| | factors are present(Andrea et al., 2021).<br>**Flexibility in Model Structure**: They allow for flexible modeling of different types of impacts, accommodating various data characteristics and intervention shapes(Andrea et al., 2021).<br>**Incorporation of Prior Information**: ARIMA models can integrate historical data and expert benchmarks, enhancing forecast accuracy by aligning with realistic scenarios and expert insights(Pierre, 1982). | ensure an accurate forecast(Nuno & Bonnie, 1996).<br>**Limited in Capturing Long-Range Dependencies**: While ARIMA models are effective for short-term dependencies, they may not perform as well with long-range dependent processes, where ARFIMA models might be more appropriate(Nuno & Bonnie, 1996).<br>**Inadequate for Non-linear Patterns**: ARIMA models may not be as effective in capturing non-linear patterns compared to other models like radial basis function networks, which have shown superior performance in certain forecasting tasks(Gwo-Fong & Lu-Hsien, 2005). |
| Exponential Smoothing | **Simplicity and Ease of Use**: Exponential smoothing models are straightforward to implement and require minimal computational resources, making them accessible for various applications(Everette & David, 1980).<br>**Adaptability**: The Holt-Winters method, a form of exponential smoothing, is particularly effective in adapting to changes in data patterns, | **Instability in Adaptive Models**: Adaptive smoothing models can produce unstable forecasts, especially when the mean demand is stable, which can be a significant drawback in certain applications(Everette & David, 1980).<br>**Sensitivity to Parameter Selection**: The performance of exponential smoothing models can deteriorate if the smoothing parameters are not |

| | | |
|---|---|---|
| | such as trends and seasonality, which enhances its predictive accuracy(Howard et al., 2007).<br>**Robustness**: Certain trend-adjusted smoothing models are robust forecasters, performing well even when the time series does not exhibit a clear trend(Everette & David, 1980).<br>**Error Variance Estimation**: State-space models for exponential smoothing provide exact analytic expressions for forecast error variances, facilitating the construction of prediction intervals(Rob et al., 2005). | appropriately chosen, as seen with the smoothed-error signal's performance at higher α values(Everette, 1983).<br>**Limited Interrelationship Handling**: Traditional univariate exponential smoothing models do not account for interrelationships between multiple time series, which can limit their effectiveness in complex forecasting scenarios(Phillip et al., 1982). |
| Theta Forecaster | **Simplicity and Interpretability**: The Theta method is relatively simple to implement and understand, making it accessible for practitioners who may not have a deep background in advanced statistical methods(Thomakos & Nikolopoulos, 2012).<br>**Competitive Performance**: In the M3 competition, the Theta method was shown to perform well, often comparable to more complex models like ARIMA, especially for univariate time series(Thomakos & Nikolopoulos, 2012). | **Limited Performance in Complex Scenarios**: While effective for univariate series, the Theta method may not perform as well as ensemble methods or advanced neural networks in complex, multivariate scenarios(Vaiciukynas et al., 2021).<br>**Suboptimal for All Data Types**: The method may not be universally optimal across all types of time series data, as evidenced by meta-learning approaches that outperform Theta in certain datasets(Vaiciukynas et al., 2021). |

| | | |
|---|---|---|
| | **Adaptability to Multivariate Forecasting**: Recent studies have extended the Theta method to multivariate time series, showing promising results in vector forecasting, which suggests its adaptability to more complex data structures(Thomakos & Nikolopoulos, 2015). | |
| Croston | **Specialized for Intermittent Demand**: Croston's method is specifically designed to handle sporadic demand patterns, making it suitable for industries with irregular sales, such as automotive and military supplies(Ruud & Laura, 2009).<br>**Improved Forecasting Accuracy**: Studies indicate that Croston's method, particularly when optimized, can outperform traditional methods like Moving Average and Single Exponential Smoothing in terms of accuracy and service level approximation(Fotios et al., 2013; Ruud & Laura, 2009).<br>**Flexibility with Modifications**: The method can be enhanced through empirical heuristics, such as optimizing smoothing parameters, which can lead to significant | **Inadequate for Zero Demand Periods**: Croston's method does not update forecasts during periods of zero demand, which can lead to inaccuracies in inventory management, especially in contexts of inventory obsolescence(Kamal & Kampan, 2021).<br>**Lack of Stochastic Foundation**: The method is criticized for its ad hoc nature, lacking a robust underlying stochastic model, which may limit its theoretical grounding and applicability in certain scenarios(Lydia Lidong & Robin John, 2005).<br>**Potential for Bias**: If not properly adjusted, Croston's method can produce biased forecasts, particularly when demand patterns change over time(Fotios et al., 2013). |

| | | |
|---|---|---|
| | improvements in forecasting performance(Fotios et al., 2013). | |
| BATS, TBATS | **Handling Complex Seasonality**: TBATS is specifically designed to manage multiple and non-integer seasonality, making it suitable for time series data with complex seasonal patterns, such as daily data with weekly and yearly cycles (Nasrin et al., 2024).<br>**Flexibility**: Both BATS and TBATS models incorporate Box-Cox transformations, ARMA errors, and trend components, providing flexibility in modeling various types of time series data(Nasrin et al., 2024).<br>**Improved Forecast Accuracy**: These models can lead to significant reductions in forecasting errors compared to traditional models like ARIMA, especially in datasets with intricate seasonal structures(Nasrin et al., 2024; Phillip, 1985). | **Complexity and Computational Cost**: The complexity of these models can lead to high computational costs, making them less feasible for very large datasets or real-time applications(Talkhi et al., 2024).<br>**Model Identification and Parameter Estimation**: Identifying the correct model order and estimating parameters can be challenging, which may affect the model's performance if not done accurately(Phillip, 1985; Wayne & Brett, 1998).<br>**Requirement for Expert Knowledge**: Selecting and tuning these models often requires expert knowledge, which may not always be available(Talkhi et al., 2024). |
| Linear Regression | **Simplicity and Interpretability**: Linear regression is straightforward to implement and understand, making it accessible for researchers and practitioners with basic statistical | **Higher Error Rates**: Linear regression can have higher prediction errors compared to more advanced techniques, particularly when the |

| | | |
|---|---|---|
| | knowledge(Andreas & Peter, 2011; Goce et al., 2013).<br>**Efficiency**: It requires less computational power compared to more complex models, which is beneficial when dealing with large datasets or when computational resources are limited(Goce et al., 2013).<br>**Robustness to Overfitting**: With fewer parameters than more complex models, linear regression is less prone to overfitting, especially when the number of predictors is limited(Tim & Klaus, 1979). | underlying data relationships are non-linear(Goce et al., 2013).<br>**Sensitivity to Outliers**: The method can be sensitive to outliers, which can significantly affect the model's performance and accuracy(Goce et al., 2013).<br>**Limited Handling of Complex Patterns**: Linear regression struggles with capturing complex patterns, such as seasonality and non-linear trends, which are common in time series data(Krishnan et al., 2013). |
| Lasso | **Implicit Model Selection:** Lasso performs implicit model selection by zeroing out insignificant coefficients, which is beneficial in high-dimensional settings where many predictors are irrelevant("Ridge regression revisited: Debiasing, thresholding and bootstrap," 2022).<br>**Granger Causal Analysis:** It is effective in identifying Granger causal influences in time series data, addressing overfitting and correlated noise issues(Proloy & Behtash, 2023). | **Sensitivity to Multicollinearity:** Lasso can struggle with multicollinearity, as it may arbitrarily select one variable over another when predictors are highly correlated("Elastic Net Penalized Quantile Regression Model and Empirical Mode Decomposition for Improving the Accuracy of the Model Selection," 2023).<br>**Bias in Parameter Estimation:** It may introduce bias in parameter estimation, especially when the true model is not sparse ("Ridge regression revisited: |

| | | |
|---|---|---|
| | | Debiasing, thresholding and bootstrap," 2022). |
| Ridge Regression | **Robustness to Multicollinearity:** Ridge regression is robust to multicollinearity, as it shrinks coefficients of correlated predictors without setting them to zero ("Elastic Net Penalized Quantile Regression Model and Empirical Mode Decomposition for Improving the Accuracy of the Model Selection," 2023).<br>**Ease of Computation:** It can be easily computed using a closed-form expression and is suitable for prediction tasks ("Ridge regression revisited: Debiasing, thresholding, and bootstrap," 2022). | **No Variable Selection:** Unlike Lasso, Ridge does not perform variable selection, which can be a limitation in high-dimensional data where sparsity is desired ("Ridge regression revisited: Debiasing, thresholding, and bootstrap," 2022).<br>**Potential for Bias:** It may introduce bias, particularly in high-dimensional settings, although debiasing techniques can mitigate this ("Ridge regression revisited: Debiasing, thresholding, and bootstrap," 2022). |
| Elastic Net | **Combined Strengths:** Elastic Net combines the strengths of Lasso and Ridge, making it suitable for datasets with multicollinearity and where variable selection is needed ("Elastic Net Penalized Quantile Regression Model and Empirical Mode Decomposition for Improving the Accuracy of the Model Selection," 2023).<br>**Improved Prediction Accuracy:** It is effective in improving prediction | **Hyperparameter Tuning Complexity:** The need to tune two hyperparameters (L1 and L2 penalties) can complicate model selection and increase computational cost(Giovanni, 2023).<br>**Specialized Scenarios:** It may not perform as well as specialized methods like WLadaLASSO in certain time series forecasting scenarios(Evandro & Flávio Augusto, 2016). |

| | | |
|---|---|---|
| | accuracy by balancing the penalties of Lasso and Ridge ("Elastic Net Penalized Quantile Regression Model and Empirical Mode Decomposition for Improving the Accuracy of the Model Selection," 2023). | |
| Least Angle Regression (LARS) | **Computational Efficiency**: LARS is computationally efficient, requiring a similar order of magnitude of computational effort as ordinary least squares, which is beneficial when dealing with large datasets or high-dimensional data(Bradley et al., 2004; Wanqing et al., 2017).<br>**Model Selection**: LARS provides a systematic approach to model selection, offering a less greedy alternative to traditional forward selection methods. It can efficiently compute all possible Lasso estimates, which is advantageous for selecting a parsimonious set of predictors(Bradley et al., 2004).<br>**Generalization Capability**: By incorporating L1 norm optimization, LARS can enhance model generalization capability, reducing prediction variance at the cost of some model bias(Wanqing et al., 2017). | **Assumptions and Limitations**: LARS assumes linearity in the parameters, which may not always hold in time series data, potentially limiting its applicability(Wanqing et al., 2017).<br>**Handling of Time Series Characteristics**: Unlike some Lasso variations, LARS does not inherently account for the natural ordering or temporal dependencies in time series data, which can be a drawback in certain forecasting scenarios(Monnie & Robert, 2019).<br>**Complexity in Implementation**: While LARS is efficient, its implementation can be complex, particularly when modifications are needed to handle specific characteristics of time series data(Bradley et al., 2004). |

| | | |
|---|---|---|
| | **Sparse Solutions**: LARS is effective in identifying sparse and stable sets of indicators, which is particularly useful in financial applications for predicting stock returns(Zitian & Shaohua, 2009). | |
| Bayesian Ridge | **Simultaneous Parameter Estimation**: Bayesian methods allow for the simultaneous estimation of multiple parameters, which is advantageous in complex models such as autoregressive moving-average models with stable innovations(Zuqiang & Nalini, 1998).<br>**Handling Uncertainty**: Bayesian Ridge provides a probabilistic approach to parameter estimation, offering a natural way to quantify uncertainty in predictions and parameter estimates(Benjamin et al., 2019).<br>**Flexibility in Model Specification**: The Bayesian framework can accommodate various model structures, including hierarchical models and nonlinear autoregression models, which can be tailored to specific time series characteristics(Antonio et al., 2005; Balgobin & Joseph, 1997). | **Computational Complexity**: Bayesian methods, including Bayesian Ridge, often require sophisticated algorithms like the Metropolis–Hastings or Gibbs Sampler, which can be computationally intensive and time-consuming(Robert & Ruey, 1994; Zuqiang & Nalini, 1998).<br>**Model Identifiability Issues**: In complex models, such as those involving neural networks, identifiability problems can arise, complicating model selection and interpretation(Antonio et al., 2005).<br>**Dependence on Prior Information**: The effectiveness of Bayesian Ridge can be heavily influenced by the choice of prior distributions, which may not always be straightforward to determine(Benjamin et al., 2019). |

| | | |
|---|---|---|
| | **Improved Precision**: When similar series are pooled, Bayesian methods can enhance the precision of estimation and forecasting, as demonstrated in panel data analysis(Balgobin & Joseph, 1997). | |
| Huber Regressor | **Robustness to Outliers**: The Huber Regressor is effective in handling outliers, which are common in time series data, by using a loss function that is less sensitive to extreme values compared to traditional least squares methods(Qiang et al., 2020; Qingsong et al., 2019).<br>**Adaptability to Non-Gaussian Data**: It performs well with non-Gaussian distributions, which are often encountered in real-world time series data, such as financial or medical datasets(Christopher, 2015; Yingjie et al.).<br>**Handling Non-Stationary Data**: The method can be adapted for non-stationary time series, providing robust forecasting capabilities by incorporating discrepancy measures for data distribution shifts(Yingjie et al.).<br>**Efficient Computation**: Despite its robustness, the computational cost of Huber-based methods is relatively low, | **Complexity in Parameter Selection**: Choosing the appropriate robustification parameter can be challenging, as it needs to adapt to the sample size and data characteristics for optimal performance(Qiang et al., 2020).<br>**Limited to Linear Models**: While Huber Regressor is robust, it is primarily a linear model, which may not capture complex nonlinear relationships in time series data(Goce et al., 2013).<br>**Sensitivity to Initial Conditions**: The performance of Huber Regressor can be sensitive to initial conditions and the choice of regularization parameters, which may require careful tuning(Qingsong et al., 2019). |

| | | |
|---|---|---|
| | making them suitable for large datasets(Christopher, 2015). | |
| Orthogonal Matching Pursuit (OMP) | **Accuracy in Sparse Signal Reconstruction**: OMP is known for its robust reconstruction capabilities, particularly in sparse data and signals, achieving low reconstruction errors and high exact recovery ability(Shuyang et al., 2024).<br>**Efficiency in High-Dimensional Data**: Generalized OMP with Singular Value Decomposition (SVD_GOMP) significantly improves computational efficiency while maintaining high accuracy, making it suitable for processing large datasets like five-dimensional seismic data("Generalized Orthogonal Matching Pursuit With Singular Value Decomposition," 2022; "High-Dimensional Generalized Orthogonal Matching Pursuit with Singular Value Decomposition," 2023).<br>**Adaptability with Self-Learning Dictionaries**: The use of self-learning dictionaries in OMP, such as in the TOMP-QR algorithm, enhances its ability to handle non-sparse signals by training the dictionary to better represent the data, thus improving | **Limitations in Dynamic Environments**: Traditional OMP is not well-suited for dynamic scenarios, as it struggles with time-varying data where channel delays and path gains change over time. Dynamic OMP addresses this limitation approaches like D*OMP, which are more efficient in such environments(Shuyang et al., 2024).<br>**Challenges with Non-Sparse Data**: OMP's performance can degrade with non-sparse signals, as seen in propeller signal reconstruction, where traditional fixed dictionaries fail to provide a sparse representation(Yanping et al., 2023).<br>**Neglect of Local Features**: The original OMP may overlook local features critical for accurate reconstruction, such as peak positions in seismic data. This issue is mitigated by strategies like Peak Colocalized OMP (PCOMP), which optimizes for both signal correlation and peak colocalization(Yongqing et al., 2020). |

| | | |
|---|---|---|
| | reconstruction accuracy and reducing running time(Yanping et al., 2023). | |
| Decision Tree Regressor | **Non-linear Relationships**: Decision Tree Regressors can model complex, non-linear relationships between variables, which is beneficial in time series data where such relationships are common(Snezhana Georgieva et al., 2019).<br>**Interpretability**: The tree structure provides a clear and interpretable model, making it easier to understand the decision-making process and the influence of different variables(Xiubin et al., 2024).<br>**Handling of Missing Data**: Decision Trees can handle missing values effectively, which is advantageous in time series data that often have gaps(Wei-Yin & Wei, 2013).<br>**Robustness to Outliers**: They are less sensitive to outliers compared to other regression models, which can improve the robustness of predictions(Snezhana Georgieva et al., 2019). | **Overfitting**: Decision Trees are prone to overfitting, especially with small datasets, as they can create overly complex models that do not generalize well to new data(Snezhana Georgieva et al., 2019).<br>**Temporal Dependencies**: Capturing temporal dependencies in time series data can be challenging for Decision Trees, as they do not inherently account for the order of data points(Wei-Yin & Wei, 2013).<br>**Bias in Variable Selection**: Traditional Decision Tree algorithms can exhibit selection bias, favoring variables with more levels or continuous variables with more potential split points(Wei-Yin & Wei, 2013).<br>**Computational Complexity**: While Decision Trees are generally fast, the computational cost can increase significantly with large datasets or when using ensemble methods like Random Forests(Saulo Martiello et al., 2022). |
| Random Forest | **Robust Performance**: Random Forest has shown robust performance in time | **Temporal Dependency Challenges**: RF may struggle with capturing |

| | | |
|---|---|---|
| | series prediction, particularly when incorporating time-lagged variables, as demonstrated in streamflow prediction studies where it outperformed traditional models like SWAT in daily predictions(Desalew Meseret et al., 2024).<br>**Stability and Interpretability**: The method is stable and interpretable, making it suitable for applications like wind speed forecasting, where it can identify important features and maintain focus on essential data despite noise(Cheng-Yu et al., 2023).<br>**Handling Nonlinear Relationships**: RF is effective in modeling complex nonlinear relationships, which is beneficial in scenarios like predicting air pollutant concentrations where human activities and policy changes introduce nonlinearity(Tunyang et al., 2023).<br>**Computational Efficiency**: Compared to other sophisticated models, RF is computationally efficient, which is advantageous in scenarios requiring quick predictions with limited data(Desalew Meseret et al., 2024). | temporal dependencies inherent in time series data, which can lead to large errors in numerical predictions over long time scales(Tunyang et al., 2023).<br>**High-Dimensional Data Handling**: While RF can handle high-dimensional data, it may require extensive feature engineering and selection to improve accuracy, as seen in wind speed forecasting, where new features were crucial for model performance(Cheng-Yu et al., 2023).<br>**Multi-Regime Complexity**: In multi-regime time series, RF needs to be combined with other models, like the extreme learning machine, to handle regime changes effectively, which adds complexity to the modeling process(Lin et al., 2017). |
| Extra Trees Regressor | **Handling Nonlinearity**: Extra Trees Regressor is adept at modeling | **Computational Complexity**: The ensemble nature of Extra Trees can |

| | | |
|---|---|---|
| | nonlinear relationships, which is crucial for time series data that often exhibit complex, nonlinear patterns. This is similar to the benefits seen with other tree-based methods like CART and mv-ARF, which are effective in capturing nonlinear dependencies in time series data(Kerem Sinan & Mustafa Gokce, 2018; Snezhana Georgieva et al., 2019).<br>**Robustness to Overfitting**: The randomness in feature selection and data sampling in Extra Trees helps in reducing overfitting, a common issue in time series forecasting. This robustness is also highlighted in the context of CART models, which are cross-validated to prevent overfitting(Snezhana Georgieva et al., 2019).<br>**Flexibility with Missing Data**: Extra Trees can handle missing values effectively, which is beneficial for time series data that may have gaps or irregularities. This flexibility is also noted in the Rand-TS framework, which deals with missing observations in time series(Berk & Mustafa Gokce, 2021). | lead to high computational costs, especially with large datasets or long time series. This is a common challenge with tree-based models, as noted in the computational difficulties associated with CART(Wei-Yin & Wei, 2013).<br>**Interpretability**: While Extra Trees can model complex patterns, the resulting models can be difficult to interpret, which is a drawback when understanding the underlying time series dynamics, which is important. This is a general issue with ensemble methods, which tend to sacrifice interpretability for accuracy(David et al., 2024). |

| | | |
|---|---|---|
| Gradient Boosting | **Accuracy and Performance**: Gradient Boosting models, such as the proposed PGBM, have demonstrated high accuracy in time series forecasting, achieving superior performance metrics like R2 scores and low error values compared to other models(Priyesh & Parida, 2024). The ensemble approach of Gradient Boosting, as seen in EGB-RNN, enhances the model's ability to fit complex time series data by integrating multiple neural network models(Shiqing et al., 2021).<br>**Speed and Scalability**: Variants like LightGBM focus on fast training performance, making Gradient Boosting suitable for large datasets and real-time applications(Candice et al., 2021). The use of efficient data representation and momentum-based optimization in Componentwise Boosting further reduces computation time and memory consumption("Accelerated Componentwise Gradient Boosting Using Efficient Data Representation and Momentum-Based Optimization," 2022). | **Computational Complexity**: Despite improvements, Gradient Boosting can still be computationally intensive, requiring significant memory and runtime, especially in its vanilla form("Accelerated Componentwise Gradient Boosting Using Efficient Data Representation and Momentum-Based Optimization," 2022). This can be a limitation for resource-constrained environments.<br>**Overfitting Risks**: The flexibility of Gradient Boosting models can lead to overfitting, particularly if not properly regularized. Techniques like stepsize restriction are employed to mitigate this risk, but they can complicate the model-tuning process(Donald et al., 2021).<br>**Model Interpretability**: While Componentwise Boosting aims to enhance interpretability, the complexity of ensemble models can make it challenging to understand the underlying data relationships("Accelerated Componentwise Gradient Boosting Using Efficient Data Representation and Momentum-Based Optimization", 2022). |

| | | |
|---|---|---|
| | **Probabilistic Forecasting**: By integrating with quantile regression, Gradient Boosting can provide probabilistic predictions, which are crucial for decision-making in uncertain environments(Priyesh & Parida, 2024). | |
| AdaBoost | **Improved Prediction Accuracy**: AdaBoost can significantly enhance the accuracy of time series forecasts by integrating multiple models, such as LSTM, to handle nonlinearity and irregularity in financial datasets(Shaolong et al., 2018).<br>**Robustness to Noise**: By using robust loss functions, AdaBoost can be adapted to improve its resistance to noise and outliers, which is crucial for maintaining accuracy in time series data with anomalies(Hong-Jie & Xi-Zhao, 2023).<br>**Versatility and Adaptability**: AdaBoost can be combined with other models, such as Broad Learning Systems, to improve stability and generalization, making it versatile for various time series applications(Yun et al., 2024). | **Sensitivity to Outliers**: AdaBoost is highly sensitive to outliers, which can deteriorate its performance in time series datasets with extreme values or noise(Hong-Jie & Xi-Zhao, 2023).<br>**Bias in Extreme Scenarios**: In scenarios with extreme demands, such as bike-sharing systems, AdaBoost may become biased, leading to increased prediction errors for normal demand situations(Lee & Kyoungok, 2024).<br>**Complexity and Computational Cost**: The integration of AdaBoost with other models, like LSTM or decision trees, can increase the complexity and computational cost, which may not be feasible for all time series applications(Lee & Kyoungok, 2024; Shaolong et al., 2018). |
| Light Gradient Boosting | **Fast Training Speed**: LightGBM is known for its extremely fast training | **Accuracy Trade-offs**: While LightGBM is fast, it is not always the |

| | | |
|---|---|---|
| | performance, which is beneficial for time series data that often involves large datasets. This speed is achieved through techniques like selective sampling of high-gradient instances(Candice et al., 2021).<br>**High Prediction Accuracy**: In various applications, such as predicting NO2 and PM2.5 levels, LightGBM has shown superior prediction accuracy compared to other models like Random Forests and XGBoost, especially at high concentrations, which is crucial for risk assessment(Tin et al., 2022).<br>**Cost Efficiency**: LightGBM can be implemented in a cost-efficient manner, using deep-boosted regression trees that are cheap to compute on average and without significant loss of accuracy(Peter et al., 2017).<br>**Versatility in Applications**: It has been successfully applied in diverse fields, from improving defrosting strategies in refrigerators to predicting chronic kidney disease progression, demonstrating its versatility in handling different types of time series data.(Chenxi et al., 2024; Hirotaka et al., 2024) | most accurate model. In some comparative studies, other models like CatBoost have shown better generalization accuracy(Candice et al., 2021).<br>**Complexity in Hyper-parameter Tuning**: The performance of LightGBM can be highly dependent on the tuning of its hyper-parameters, which can be complex and time-consuming(Candice et al., 2021).<br>**Sensitivity to Data Quality**: The accuracy of LightGBM can be affected by the quality of input data, as seen in applications where the model's performance varies based on the feature sets used(Chenxi et al., 2024). |

| | | |
|---|---|---|
| CatBoost | **Handling Categorical Data**: CatBoost is particularly adept at processing categorical features without requiring extensive preprocessing, which is beneficial in time series datasets that often include categorical variables(John & Taghi, 2020).<br>**Robustness to Missing Data**: The algorithm's ability to handle missing values effectively, as demonstrated in the prediction of carbon content in electric arc furnaces, enhances its applicability in time series data where missing entries are common(Hongbin et al., 2024).<br>**Feature Importance and Selection**: CatBoost's feature ranking capabilities, as seen in heart rate classification, allow for effective feature selection, which is crucial in time series analysis to reduce dimensionality and improve model performance(Dhananjay & Sivaraman, 2021).<br>**Accuracy and Efficiency**: The algorithm has shown high accuracy and efficiency in various applications, such as tire wear monitoring and heart rate classification, (Dhananjay & Sivaraman, 2021; Prasshanth & Sugumaran, 2024). | **Hyper-Parameter Sensitivity**: CatBoost requires careful tuning of hyper-parameters to achieve optimal performance, which can be a complex and time-consuming process, especially in time series contexts where data characteristics can vary significantly(John & Taghi, 2020).<br>**Computational Complexity**: The algorithm's computational demands can be high, particularly for large datasets, which may limit its practicality in real-time time series applications(John & Taghi, 2020).<br>**Limited Interpretability**: As with many ensemble methods, the interpretability of CatBoost models can be limited, posing challenges in understanding the underlying patterns in time series data(John & Taghi, 2020). |

## 2.3. RELATED LITERATURE

Sales forecasting is crucial to business strategy and operations, significantly influencing inventory management, budgeting, and resource allocation decisions. Accurate forecasts enable organizations to optimize supply chains, reduce costs, and enhance customer satisfaction by aligning production with anticipated market demand(Choi et al., 2018). Traditional forecasting methods, such as the AutoRegressive Integrated Moving Average (ARIMA) and Exponential Smoothing models, have long been foundational in time series analysis. These statistical models use historical data patterns to predict future sales, capturing linear trends and seasonality. However, according to Evangelos and Fotios, traditional statistical forecasting methods like ARIMA and Exponential Smoothing face several inherent limitations when applied to real-world sales data(Evangelos & Fotios, 2023). In their study, Maximiliano et al. found that one significant area for improvement is these methods' sensitivity to model parameters and assumptions, which can lead to suboptimal performance in dynamic and complex environments. They assert that seasonal exponential smoothing models can outperform simpler models under certain conditions. However, they are susceptible to demand frequency and smoothing parameters, making them less robust in varied real-world scenarios(Maximiliano et al., 2022).

Findings by Josef et al. show that these traditional methods often need help with data sparsity and heterogeneity. As Josef et al. observed in the horticultural industry, dataset-specific predictors and external factors significantly enhance forecasting accuracy compared to general models(Josef et al., 2024). Additionally, Andrés et al. underline the importance of considering temporal sampling effects, observing that the predictability of systems degrades with temporal sampling—a limitation that traditional models cannot fully mitigate. They note that this highlights the inability of conventional methods to recover lost predictability in partially observed systems(Andrés et al., 2020).

The advent of machine learning models has introduced powerful alternatives capable of modeling non-linear relationships and interactions within data. According to Yan et al., machine learning models have become increasingly adept at handling non-linear patterns in sales data, offering significant implications for business decision-making(Yan et al., 2019). They demonstrated that deep neural networks, such as the Deep Neural Framework (DNF) for sales forecasting in e-commerce, leverage sequence-to-sequence learning to model the impact of promotional campaigns and competing products, resulting in superior accuracy over traditional methods.

Ali and Chyi Lin reported that machine learning algorithms like Decision Trees and Random Forests in regional housing markets outperform traditional econometric models by capturing non-linear dynamics between housing prices and various microeconomic and socioeconomic factors(Ali & Chyi Lin, 2024). Similarly, Thais de Castro et al. observe that hybrid models combining Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN) further enhance forecasting by integrating temporal and exogenous data, thus improving accuracy and reducing computational complexity(Thais de Castro et al.,

2024).

Daifeng, Fengyun, et al. suggest that using ensemble learning and dynamic prediction models, such as the DSP-FAE, allows for effectively capturing changing sales patterns, optimizing inventory management, and reducing costs(Daifeng, Fengyun, et al., 2023). Additionally, Song and Yang demonstrate that models like M-GAN-XGBoost, which utilize advanced techniques such as Generative Adversarial Networks (GAN) and XGBoost, can adaptively predict sales, enabling precise marketing strategies and resource allocation(Song & Yang, 2021). Recent advancements in machine learning have significantly enhanced forecasting accuracy, providing businesses with valuable insights critical for strategic planning, marketing initiatives, and improving operational efficiency. As stated in "Improving Sales Forecasting Accuracy: A Tensor Factorization Approach with Demand Awareness," by uncovering non-linear patterns, these models facilitate more informed decision-making, ultimately enhancing business performance and competitiveness in dynamic markets("Improving Sales Forecasting Accuracy: A Tensor Factorization Approach with Demand Awareness," 2022).

Cyril et al. demonstrate that advanced machine learning models like XGBoost have achieved superior predictive accuracy in sales forecasting, attaining near-perfect R-squared values and significantly lower error rates than simpler models(Cyril et al., 2024). Moreover, Hanan Wagih and Mohamed Mamdouh found that a cognitive analytics framework, which integrates statistical, machine learning, and deep learning forecasting stages, improves accuracy by using the error of each model as an input feature for subsequent models, achieving up to 90% accuracy in sales predictions(Hanan Wagih & Mohamed Mamdouh, 2023).

Integrating Automated Machine Learning (AutoML) with forecasting algorithms significantly advances predictive analytics. According to George et al., AutoML tools such as AutoGluon, Auto-Sklearn, and PyCaret streamline the development of high-performing models by managing the complexities of time series data, thereby making machine learning more accessible to non-experts(George et al., 2024). They emphasize that this approach is particularly beneficial in retail sales forecasting, where diverse data types and temporal dependencies challenge traditional models.

Felipe Rooke da et al. highlighted that AutoML's optimization capabilities, such as using genetic algorithms for hyper-parameter tuning, further enhance model performance and surpass traditional benchmarks in time-series forecasting(Felipe Rooke da et al., 2022). The study "Automated Machine Learning for Time Series Prediction" supports these findings by demonstrating the effectiveness of AutoML in improving forecasting accuracy("Automated Machine Learning for Time Series Prediction," 2022). These advancements underscore the transformative potential of AutoML in enhancing sales prediction accuracy. As concluded by these researchers, the automation of model selection and optimization processes reduces errors in sales predictions and enables better inventory management and strategic decision-making in retail operations. Additionally, adopting AutoML tools can lead to more reliable and efficient forecasting models, ultimately driving

improved business outcomes.

Despite recent advancements, the current literature still needs to reveal a significant gap. Notably, more research needs to be focused on PyCaret-based sales forecasting, which represents a critical shortcoming in developing accurate and reliable forecasting models. PyCaret, a low-code machine learning library, offers a streamlined approach to model selection, hyperparameter tuning, and deployment, which could be particularly beneficial in contexts where diverse data sources and complex model requirements are involved(Ali, 2020). According to Abhishek and Oliver, using online search data for competitive analysis in the automobile industry exemplifies this complexity(Abhishek & Oliver, 2024). Similarly, the study "Forecasting sales using online review and search engine data: A method based on PCA–DSFOA–BPNN" integrates sentiment analysis and search engine data for forecasting sales("Forecasting sales using online review and search engine data: A method based on PCA–DSFOA–BPNN," 2022). These approaches highlight the need for models that can handle large datasets and complex relationships, a niche where PyCaret could excel by simplifying the model development process.

Josef et al. emphasize the superiority of machine-learning methods like XGBoost for handling multivariate data in the horticultural industry, suggesting that PyCaret's automated machine-learning capabilities could streamline such processes(Josef et al., 2024). Additionally, Odysseas et al., Shanhe, and Weixiong demonstrate the use of deep learning models like Long Short-Term Memory (LSTM) in various contexts, such as retail demand forecasting and identifying new market opportunities(Odysseas et al., 2022; Shanhe & Weixiong, 2020). These studies underscore the importance of advanced modeling techniques that PyCaret could facilitate by integrating deep learning frameworks.

Research by Abhishek and Oliver study indicates that current methodologies often require extensive parameter tuning and computational resources(Abhishek & Oliver, 2024). This complexity can be a barrier for small-scale operations or industries with limited technical resources. Therefore, the absence of PyCaret in these studies may hinder the accessibility and efficiency of developing robust forecasting models. As a result, incorporating PyCaret into sales forecasting research could democratize access to sophisticated modeling techniques. This integration can improve forecast accuracy and reliability across various sectors, making advanced forecasting tools more accessible to businesses with limited technical expertise.

In conclusion, while machine learning models offer promising enhancements over traditional methods, studies are clearly needed to apply these techniques to contemporary, real-world sales data. Additionally, the potential of AutoML tools like PyCaret in sales forecasting is yet to be fully explored. This research will contribute to the field by addressing these gaps, providing insights into the effectiveness of PyCaret for sales prediction, and determining whether it presents a more practical and efficient approach for time series sales data forecasting.

# CHAPTER 3
# EXPERIMENTAL METHODOLOGY

## 3.1. METHODOLOGY

This study employs a quantitative research approach, focusing meticulously on the analysis of numerical data and using measurable metrics to assess the performance of machine learning algorithms in forecasting applications. By centering on quantifiable variables, the research aims to provide objective, empirical evidence regarding the efficacy of various models and techniques within the machine learning domain. The quantitative methodology facilitates a systematic examination of data through statistical analysis, enabling the derivation of meaningful conclusions based on numerical findings. This approach allows for the identification of patterns, trends, and relationships within large datasets, which are crucial for the development of accurate and reliable forecasting models. The emphasis on measurable outcomes not only ensures the replicability of results but also enhances their generalizability to broader contexts. The study contributes to a robust framework for evaluating machine learning algorithms' performance by employing sophisticated statistical tools and quantitative measures. This, in turn, adds significant value to the existing body of knowledge in quantitative data analysis and predictive modeling, reinforcing the importance of quantitative methods in advancing technological applications in forecasting.

Characterized as applied research, this study also emphasizes the practical implementation of machine learning and automated machine learning (AutoML) frameworks to solve real-world forecasting problems. The primary objective is to bridge the gap between theoretical advancements in machine learning and their practical applications through sales forecasting. The research demonstrates how machine learning and AutoML can enhance forecasting accuracy and efficiency by deploying advanced computational techniques in actual forecasting scenarios. The study involves developing and testing machine learning models using a real-world dataset, thereby showcasing the applicability and effectiveness of these technologies in addressing complex forecasting challenges. This practical application provides valuable insights into implementation challenges, potential solutions, and the overall impact of machine learning technologies on contemporary forecasting practices. By focusing on real-world applications, the research supports the advancement of predictive analytics and encourages the integration of machine learning technologies into mainstream forecasting methodologies. This approach not only contributes to the practical field but also fosters innovation by highlighting the transformative potential of machine learning in solving tangible, real-world problems.

### 3.1.1. Research methods

#### *3.1.1.1. Document Research*

The research methodology builds upon an extensive review of theoretical frameworks and existing forecasting models, providing a solid foundation for the study. Central to this

review are statistical forecasting models, such as AutoRegressive Integrated Moving Average (ARIMA), advanced statistical models like BATS and TBATS, and machine learning models integrated into the PyCaret automated machine learning (AutoML) framework. These models' strengths, limitations, and applicability to various forecasting scenarios are systematically evaluated to guide the experimental design.

The Naïve model, recognized for its simplicity and ease of implementation in time series forecasting, is a benchmark for comparison. Advanced statistical models are incorporated to address complex seasonal patterns and high-frequency data, while machine learning models—augmented with preprocessing steps like deseasonalization and detrending—enable the study to explore nonlinear relationships within the data.

PyCaret's AutoML framework is a key enabler of this research, automating processes for model selection, hyperparameter tuning, and evaluation, thereby reducing the complexity traditionally associated with developing machine learning workflows. The comprehensive review also informs the selection of eight evaluation metrics—Mean Absolute Scaled Error (MASE), Root Mean Squared Scaled Error (RMSSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Symmetric Mean Absolute Percentage Error (SMAPE), and Computational Time (TT)—ensuring the models are evaluated across a broad spectrum of performance criteria.

By leveraging insights from existing studies, the research ensures alignment between theoretical knowledge and practical application, tailoring its methodological framework to address real-world sales forecasting challenges. Consequently, the literature review shapes the experimental phases, grounding them in established scholarly work while facilitating robust and reliable forecasting outcomes.

#### *3.1.1.2. Experimental Method*

The experimental method delineates a structured process for testing and evaluating various forecasting models, ensuring a rigorous and systematic approach to data analysis. The experimentation encompasses models facilitated by the PyCaret automated machine learning framework. Utilizing Python as the primary programming language, alongside libraries such as PyCaret, the study leverages powerful computational tools to implement and compare these models effectively. The experimental procedure is divided into distinct phases: training, testing, and performance evaluation. During the training phase, each model is fitted to the historical data to learn underlying patterns and relationships. The testing phase involves applying the trained models to unseen data to assess their predictive capabilities. Performance evaluation is conducted using a suite of metrics to quantify the accuracy and reliability of each model's forecasts. Additionally, the workflow of the experimental process is visually represented through flowcharts, providing a clear and concise overview of the steps involved. This visual representation elucidates the sequential progression from data preprocessing to model deployment and evaluation. By adhering to this comprehensive experimental framework, the study ensures that each model is rigorously tested under consistent conditions, facilitating a fair and objective comparison

of their forecasting performance. This methodological rigor not only enhances the validity of the findings but also contributes to the broader discourse on effective forecasting techniques in applied machine learning research.

### 3.1.2. Justification of methodology and models used

The primary objective of this study is to evaluate the performance of machine learning models in comparison to traditional forecasting methods for real-world sales prediction, utilizing the PyCaret framework. The selection of models is strategically aligned with this objective to ensure a comprehensive analysis encompassing a wide spectrum of forecasting approaches. The justification for the chosen models (see **table 3.1**) is as follows:

#### *3.1.2.1. Inclusion of Traditional Forecasting Methods:*

Naive Forecaster (naive), Seasonal Naive Forecaster (snaive), Exponential Smoothing (exp_smooth), ARIMA (arima), and ETS (ets) are classical time series models widely recognized in the forecasting domain.

These models serve as benchmarks due to their simplicity and historical significance in time series analysis. Including these methods allows for a direct comparison between traditional approaches and advanced machine learning models, fulfilling the study's goal of determining the most effective forecasting technique.

#### *3.1.2.2. Integration of Advanced Statistical Models:*

Auto ARIMA (auto_arima) and Theta Forecaster (theta) automate the model selection process within traditional methods, enhancing efficiency and potentially improving forecasting accuracy. BATS (bats) and TBATS (tbats) are advanced models capable of handling complex seasonal patterns and high-frequency data, which are common in sales forecasting.

#### *3.1.2.3. Incorporation of Machine Learning Models:*

A variety of machine learning algorithms are included, such as Decision Tree (dt_cds_dt), Random Forest (rf_cds_dt), Gradient Boosting (gbr_cds_dt), Light Gradient Boosting (lightgbm_cds_dt), and CatBoost Regressor (catboost_cds_dt). These models are known for capturing non-linear patterns and interactions within the data that traditional models might miss. Their inclusion is essential to evaluate whether machine learning offers a practical advantage over traditional methods in sales forecasting.

#### *3.1.2.4. Models with Conditional Deseasonalize and Detrending (*_cds_dt):*

Time series data often exhibit patterns such as trend and seasonality, which can significantly impact forecasting accuracy. Traditional statistical models are designed to capture these components explicitly. However, many machine learning models do not inherently account for trend and seasonality, potentially leading to suboptimal performance

when applied directly to raw time series data. To address this issue, PyCaret employs a preprocessing step involving deseasonalizing and detrending the data. The conditional aspect refers to applying these transformations only when statistical evidence suggests their presence. This ensures that essential patterns are not removed unnecessarily, preserving the integrity of the data for forecasting. This preprocessing step is particularly beneficial for machine learning models, which may not inherently handle seasonality and trends as effectively as traditional time series models.

#### *3.1.2.5. Use of PyCaret's Integrated Models:*

All selected models are integrated within PyCaret's model library, ensuring consistency in implementation and evaluation. PyCaret provides a unified interface for training, tuning, and comparing models, which streamlines the experimentation process and aligns with the study's goal of simplifying model selection and evaluation.

#### *3.1.2.6. Diversity in Modeling Approaches:*

The combination of statistical methods, automated algorithms, and machine learning models provides a diverse set of approaches. This diversity increases the likelihood of identifying the most effective and practical forecasting method for the given sales data.

#### *3.1.2.7. Relevance to Real-World Sales Data:*

Sales data often contain irregular patterns, intermittent demand, and multiple seasonal cycles. Models like **Croston (croston)** are specifically designed for intermittent demand forecasting, making them relevant for certain types of sales data. Advanced models like **STLF (stlf)** and **Polynomial Trend Forecaster (polytrend)** are included to capture complex trends and seasonal variations.

#### *3.1.2.8. Alignment with Research Objective:*

The selected models collectively enable a thorough evaluation of different forecasting methodologies. This alignment ensures that the study can effectively address its core objective: to determine which forecasting approach is the most effective and practical for time series sales data.

**Table 3.1** Overview of forecasting methods used in PyCaret

(for implementation details, see Appendix B, lines 20-21)

| ID | Name | Reference |
|---|---|---|
| naive | Naive Forecaster | sktime.forecasting.naive.NaiveForecaster |
| snaive | Seasonal Naive Forecaster | sktime.forecasting.naive.NaiveForecaster |

| | | |
|---|---|---|
| polytrend | Polynomial Trend Forecaster | sktime.forecasting.trend._polynomial_trend_forecaster |
| arima | ARIMA | sktime.forecasting.arima.ARIMA |
| exp_smooth | Exponential Smoothing | sktime.forecasting.exp_smoothing.ExponentialSmoothing |
| theta | Theta Forecaster | sktime.forecasting.theta.ThetaForecaster |
| croston | Croston | sktime.forecasting.croston.Croston |
| bats | BATS | sktime.forecasting.bats.BATS |
| tbats | TBATS | sktime.forecasting.tbats.TBATS |
| lr_cds_dt | Linear w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| en_cds_dt | Elastic Net w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| ridge_cds_dt | Ridge w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| lasso_cds_dt | Lasso w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| llar_cds_dt | Lasso Least Angle Regressor w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| br_cds_dt | Bayesian Ridge w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| huber_cds_dt | Huber w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |

| | | |
|---|---|---|
| omp_cds_dt | Orthogonal Matching Pursuit w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| dt_cds_dt | Decision Tree w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| rf_cds_dt | Random Forest w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| et_cds_dt | Extra Trees w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| gbr_cds_dt | Gradient Boosting w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| ada_cds_dt | AdaBoost w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| lightgbm_cds_dt | Light Gradient Boosting w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |
| catboost_cds_dt | CatBoost Regressor w/ Cond. Deseasonalize & Detrending | pycaret.containers.models.time_series.BaseCdsDtModel |

*Source: Ali, 2020*

### 3.1.3. AutoML framework

This work used PyCaret's AutoML framework, which offers a comprehensive suite of features that significantly streamline the model selection process. Key functionalities include initialization, model comparison, analysis, evaluation, and visualization of machine learning models. Among these, the setup() function is the foundation of PyCaret's workflow, enabling users to define preprocessing steps and configure the machine learning pipeline (see Appendix C, lines 14-18). This function requires the dataset and the target variable to be specified before any subsequent operations can be executed, thereby establishing a structured and efficient workflow.

The model comparison process in PyCaret is particularly robust. Utilizing the compare_models() function, PyCaret evaluates a variety of machine learning models against specified performance metrics and ranks them from best to worst (see Appendix C, lines 24-25). This ranking process considers not only the accuracy of the models but also the computational efficiency, as the time required for comparison varies based on the size and complexity of the dataset. This feature facilitates informed decision-making by enabling users to select the most suitable model for their data and objectives.

A key strength of PyCaret lies in its ability to enhance the interpretability of machine learning models through visualization tools. The plot_model() function generates graphical representations of performance metrics and diagnostic insights, providing a deeper understanding of the trained models' behavior. These visualizations are instrumental in diagnosing potential issues, evaluating performance, and refining model selection processes (see Appendix D).

Despite its user-friendly design, PyCaret assumes a basic level of familiarity with machine learning concepts and requires datasets to be preformatted according to specific guidelines. For example, users must ensure that the target variable is clearly defined and that data preprocessing steps align with PyCaret's setup process requirements. However, PyCaret compensates for these prerequisites by providing extensive documentation and tutorials, which offer practical examples tailored to diverse scenarios("PyCaret — pycaret 3.0.4 documentation,").

In the context of forecasting, PyCaret's simplified workflow is highly advantageous. The methodology employed in this study, as seen in the figure below, leveraged the following PyCaret functions: (1) initialization via setup(), (2) model training and comparison using compare_models(), and (3) performance analysis through plot_model(). Each function is critical in automating and optimizing the machine-learning process, thereby reducing the complexity traditionally associated with model development and selection. By integrating these features, PyCaret empowers users to focus on strategic analysis and insights rather than technical implementation.

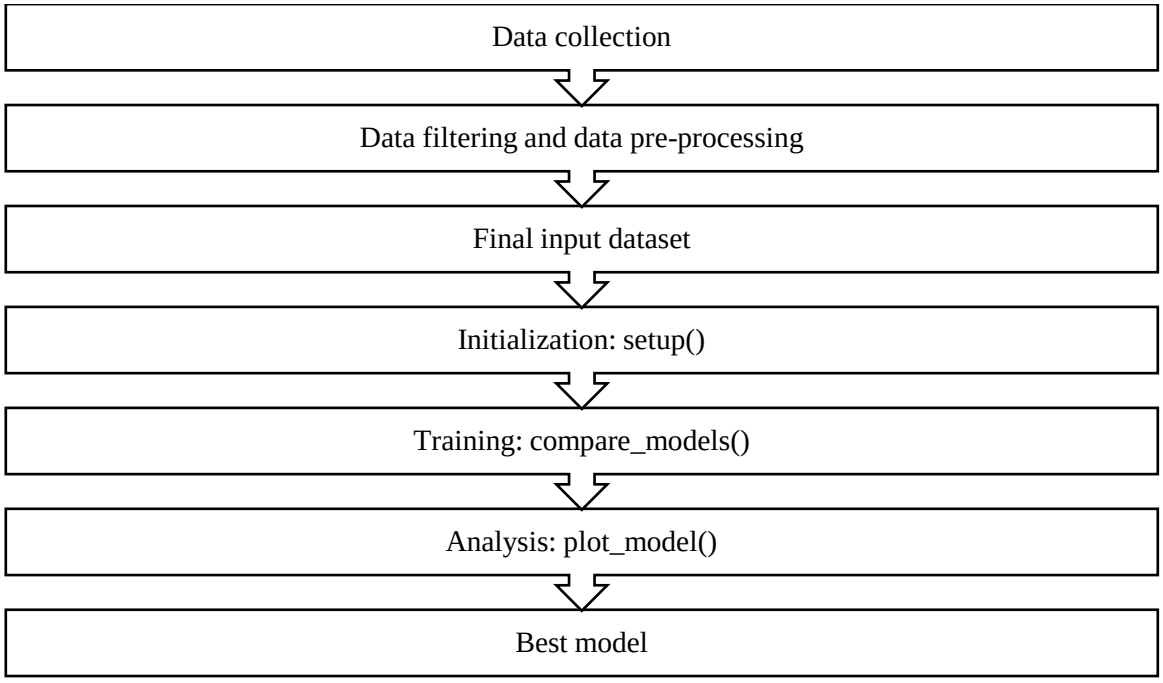


**Figure 3.1** Workflow of time series methodology in PyCaret

*Source: Author*

#### *3.1.3.1. Experimental set up*

The dataset was preprocessed to ensure consistency and suitability for time series forecasting. First, the data frequency was standardized to daily intervals using the `asfreq('D')` method (see Appendix C, line 8). This ensured uniform temporal spacing, which is critical for time series modeling. Subsequently, any missing values in the `total_sales` column, which may have arisen due to setting the daily frequency, were using `resample('D').interpolate()` method (see Appendix C, line 11). Forward-filling imputes missing values by propagating the last observed value forward, maintaining the continuity of the dataset. These preprocessing steps ensured that the data was clean and ready for the modeling phase.

The `setup()` function forms the foundation of this study's methodology (see Appendix C, lines 14-19). Key details about the setup process are described below:

**Initialization**: The `setup()` function initializes the PyCaret environment by defining the target variable, specifying the training and testing split ratio (default: 80:20), and setting preprocessing configurations such as scaling and encoding.

**Preprocessing**: The `setup()` function automates data preprocessing tasks such as handling missing values, scaling, and encoding. This ensures that the dataset is appropriately formatted for modeling without manual intervention.

The following configuration parameters were employed during the setup phase:

- **`data=df`**: Specifying the dataset input
- **`target='total_sales'`**: Specifying the variable to predict.
- **`session_id=123`**: For reproducibility.
- **`numeric_imputation_target='mean'`**: Replacing any missing values in column '`total_sales`' with the column's mean value.

By limiting the setup configuration to these two parameters, the study avoids introducing unnecessary complexities, making the methodology straightforward and replicable. Additional configurations such as data splitting, cross-validation, and preprocessing were handled automatically by PyCaret's default settings, ensuring minimal intervention.

### 3.1.4. Software and hardware

In this study, the author used the programming language Python version 3.11.9 | packaged by Anaconda, Inc. based in Austin, TX, United States on win64 and PyCaret 3.3.2. The device and window specifications are:

- Processor: Intel(R) Core(TM) Ultra 5 125H  3.60 GHz
- Installed RAM: 16.0 GB (15.4 GB usable)
- Edition: Windows 11 Pro
- Version: 23H2
- Installed on 2/22/2024
- OS build: 22631.4460
- Experience: Windows Feature Experience Pack 1000.22700.1047.0

## 3.2. DATA COLLECTION

The dataset employed in this study is a benchmark dataset for solving real-world sales forecasting problems. It contains real transactional sales data obtained experimentally in a production environment at one of the largest retail companies in Bosnia and Herzegovina. This dataset was sourced from the publicly available repository hosted by 4TU.ResearchData(Žunić, 2019), a trusted repository for high-quality research data.

The data was collected under real-world production conditions, providing an authentic representation of retail sales patterns. The downloaded dataset is in CSV format and renamed to 'real_world_sales_data.csv' for reproducible purposes (see Appendix A, line 6).

## 3.3. DATA PRE-PROCESSING

Several preprocessing steps were conducted to prepare the dataset for analysis and ensure the reliability of results. These steps included transforming data types, handling invalid values, aggregating data, and verifying the final dataset. Each step was implemented using Python and its associated libraries (see Appendix A and B).

### 3.3.1. Transforming data types

Two columns required data type transformations to ensure compatibility with subsequent analyses:

#### *3.3.1.1. Converting data types*

The **item_code** column, representing product identifiers, was converted from int64 to object to reflect its categorical nature and avoid numerical operations.

Initially stored as an object, the **date** column was converted to datetime64[ns] for efficient time-based operations such as sorting, filtering, and aggregation.

#### *3.3.1.2. Revenue calculation*

A new column, **total_sales**, was calculated for each transaction as total_sales = quantity * unit_price. This transformation created the primary feature for forecasting total daily revenue.

### 3.3.2. Handling invalid values

The dataset contained negative values in the quantity and unit_price columns. These values are invalid in the context of sales data (e.g., negative sales quantities or prices) and could distort the analysis. Rows containing these invalid values were removed.

Steps:

1. Removed rows where quantity < 0.

2. Removed rows where unit_price < 0.

### 3.3.3. Aggregating data for daily sales

Aggregation is crucial for transforming product-level data into a time-series format suitable for forecasting. The dataset was aggregated by date to compute daily total revenue.

Steps:

1. Grouped transactions by the date column.

2. Summed the sales for each day.

3. Created a new dataset containing only the date and the corresponding total sales.

## 3.4. EVALUATION METRICS

### 3.4.1. Mean Absolute Scaled Error (MASE)

MASE scales the absolute error by dividing it by the mean absolute error of a naïve forecast, enabling comparisons across datasets. It is particularly useful for evaluating forecasts across series with varying scales or seasonalities.

$$MASE = \frac{\frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|}{\frac{1}{n-1}\sum_{t=2}^{n}|y_t - y_{t-1}|}$$

where $y_t$ is the actual value, $\hat{y}_t$ is the forecasted value, and $n$ is the number of observations

### 3.4.2. Root Mean Squared Scaled Error (RMSSE)

RMSSE is similar to MASE but emphasizes larger errors by squaring them before scaling using the mean squared error of a naïve forecast. It is often used to penalize outliers and assess trends in scaled datasets.

$$RMSSE = \sqrt{\frac{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\frac{1}{n-1}\sum_{t=2}^{n}(y_t - y_{t-1})^2}}$$

### 3.4.3. Mean Absolute Error (MAE)

MAE calculates the average absolute difference between predicted and actual values, providing a simple and interpretable measure of error magnitude. It is ideal for straightforward error analysis when all errors are treated equally.

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_t|$$

### 3.4.4. Root Mean Squared Error (RMSE)

RMSE takes the square root of the average squared differences between predicted and actual values, penalizing larger errors more than MAE. This metric is commonly used in regression problems where significant outliers must be addressed.

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}$$

### 3.4.5. Mean Absolute Percentage Error (MAPE)

This metric expresses errors as a percentage of actual values, making it scale-independent but sensitive to small actual values. It is widely used in business or finance, where percentage errors offer clearer insights.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \widehat{y_t}}{y_t}\right| \times 100\%$$

### 3.4.6. Symmetric Mean Absolute Percentage Error (SMAPE)

SMAPE modifies MAPE by considering both the actual and predicted values in its denominator, making it less biased when actual values are low. It is preferred in scenarios where symmetric treatment of errors is needed.

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \widehat{y_i}|}{\frac{|y_i| + |\widehat{y_i}|}{2}} \times 100\%$$

Where:

$y_i$:The actual value or observed data point at index $i$.

$\widehat{y_i}$:The predicted value for the corresponding actual value $y_i$ at index $i$.

### 3.4.7. Computational time (TT)

Computation Time (TT) refers to the duration required for a model to complete its training and produce forecasts. This metric is critical in real-world scenarios where time efficiency plays a significant role.

Measurement: TT is usually expressed in seconds (s) and is tracked using PyCaret's system functions that monitor the time taken for the forecasting process.

# CHAPTER 4
# RESULTS AND DISCUSSION

## 4.1. EXPLORATORY DATA ANALYSIS

### 4.1.1. Data overview

The dataset used in this study comprises **412,357 records** spanning from **January 2, 2010, to September 28, 2018**, providing a comprehensive temporal scope for analysis. This dataset captures key information about retail sales transactions, including item-level details, quantities sold, pricing, and revenue generated. Below is a summary of the dataset's key attributes:

- **item_code**: A unique identifier for each product in the dataset.
- **date**: The date on which the transaction occurred, representing daily timestamps across the entire time span.
- **quantity**: The number of units sold for each item on a particular date.
- **unit_price**: The price per unit for each item.
- **total_sales**: The total sales generated for each item on a given day, calculated as the product of quantity and unit_price.

**Table 4.1** A sample of raw time series dataset

| item_code | date | quantity | unit_price | total_sales |
|---|---|---|---|---|
| 501001000001 | 2010-01-02 | 399 | 1.330 | 530.670 |
| 501001000001 | 2010-01-04 | 812 | 1.338 | 1086.456 |
| 501001000001 | 2010-01-05 | 516 | 1.331 | 686.796 |
| 501001000001 | 2010-01-06 | 1164 | 1.340 | 1559.760 |
| 501001000001 | 2010-01-08 | 1133 | 1.339 | 1517.087 |

The final input dataset contains two key features:

**date**: The daily timestamps.

**total_sales**: The total sales aggregated across all products.

**Table 4.2** A sample of the final input time series dataset

| date | total_sales |
|---|---|
| 2010-01-02 | 11407.120 |
| 2010-01-04 | 59894.618 |
| 2010-01-05 | 29215.334 |
| 2010-01-06 | 44770.444 |
| 2010-01-08 | 37141.293 |

### 4.1.2. Statistical summary

The dataset exhibits significant variability in key metrics, reflecting the diverse nature of retail transactions:

The mean daily quantity sold per transaction is **529 units**, which is accompanied by a large standard deviation of **3,565 units**, indicating considerable fluctuations in sales volume.

The unit_price ranges from **0.00 to 52.80**, with an average of **2.28**, while the median price is **1.09**, showing a skewed distribution of prices.

The sales generated per transaction range from **0.00 to 182,476.80**, averaging **446.06**. This highlights the presence of outliers, likely driven by large-scale transactions or high-value items.

**Table 4.3** Summary statistics of sales dataset

| | quantity | unit_price | total_sales |
|---|---|---|---|
| count | 412357.000000 | 412357.000000 | 412357.000000 |
| mean | 529.028058 | 2.276360 | 446.064211 |
| std | 3565.193467 | 3.970715 | 1889.128516 |
| min | 1.000000 | 0.000000 | 0.000000 |
| 25% | 31.000000 | 0.605000 | 38.000000 |
| 50% | 120.000000 | 1.090000 | 126.000000 |
| 75% | 372.000000 | 1.790000 | 354.000000 |

| | | | |
|---|---|---|---|
| max | 735400.000000 | 52.800000 | 182476.800000 |

### 4.1.3. Data distribution

The preliminary analysis indicates significant variability in sales volumes, marked by extreme peaks such as a single transaction involving 735,400 units. These anomalies suggest occurrences like bulk orders or promotional events that temporarily elevate sales figures. This level of variability implies that sporadic yet impactful transactions influence the overall sales performance, which could affect the accuracy of forecasting models if not properly accounted for.

Notable outliers are also present in both the quantity and total_sales columns. These outliers necessitate further exploratory data analysis to assess their nature and impact on forecasting accuracy. They could result from data entry errors, exceptional business events, or other factors causing sudden deviations in sales figures. Addressing these anomalies is essential to enhance the reliability of statistical analyses and predictive models.

Moreover, the dataset spans nearly nine years, providing a robust foundation for time series analysis. This extensive temporal coverage allows for the identification of long-term trends, seasonal patterns, and cyclical behaviors influencing sales performance. Analyzing such a comprehensive timeframe can yield valuable insights into market dynamics, consumer behavior, and the effectiveness of past strategies, thereby informing future decision-making processes.

### 4.1.4. Data visualization

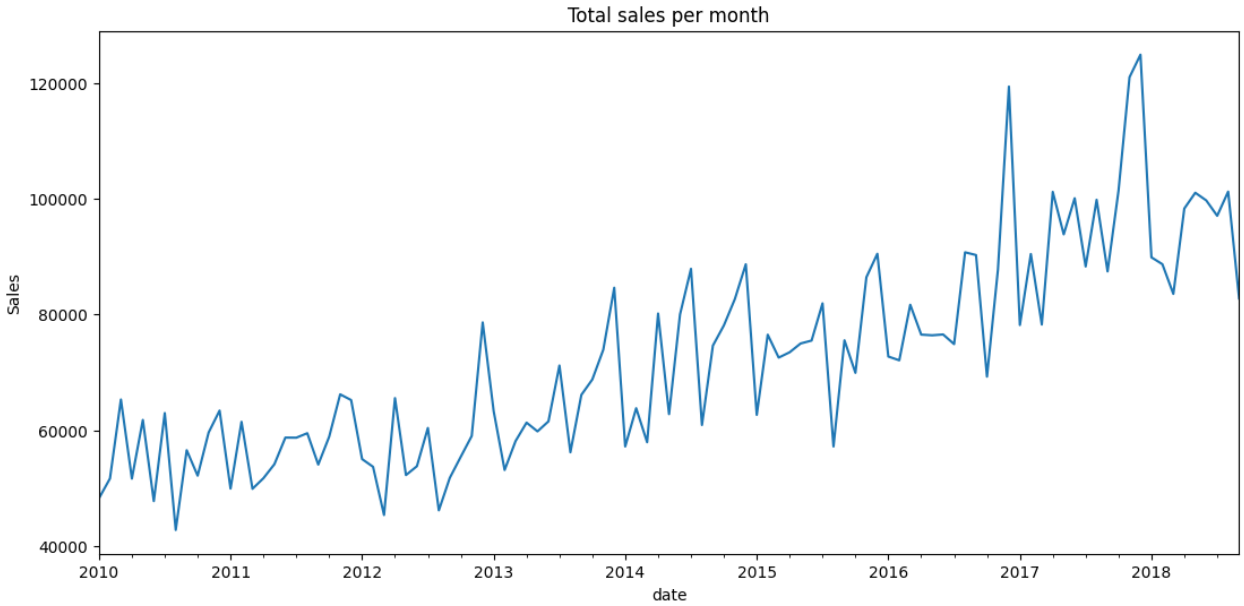


**Figure 4.1** Line chart of total sales from 2010-2018

(Source: Author)

The line chart displays the total monthly sales from 2010 to 2018. Overall, sales show a positive upward trend, though notable fluctuations can be observed throughout the period. In 2010, monthly sales remained relatively stable, ranging between 40,000 and 60,000 units. However, a gradual increase became apparent over the years, with monthly totals exceeding 80,000 by 2015. This upward momentum continued, culminating in a peak of over 120,000 in 2017.

Despite the general growth, sales displayed significant volatility, characterized by frequent rises and declines. Key peaks were noted in 2014, 2015, and 2017, each followed by sharp declines, suggesting a cyclical pattern. The data also indicates potential seasonality, as similar fluctuations appeared annually.

When comparing the early to the later years, monthly sales in 2018 were nearly double those in 2010, underscoring substantial growth over the eight-year span. In summary, the chart illustrates a positive trend in total monthly sales over time, influenced by both cyclical and seasonal variations.

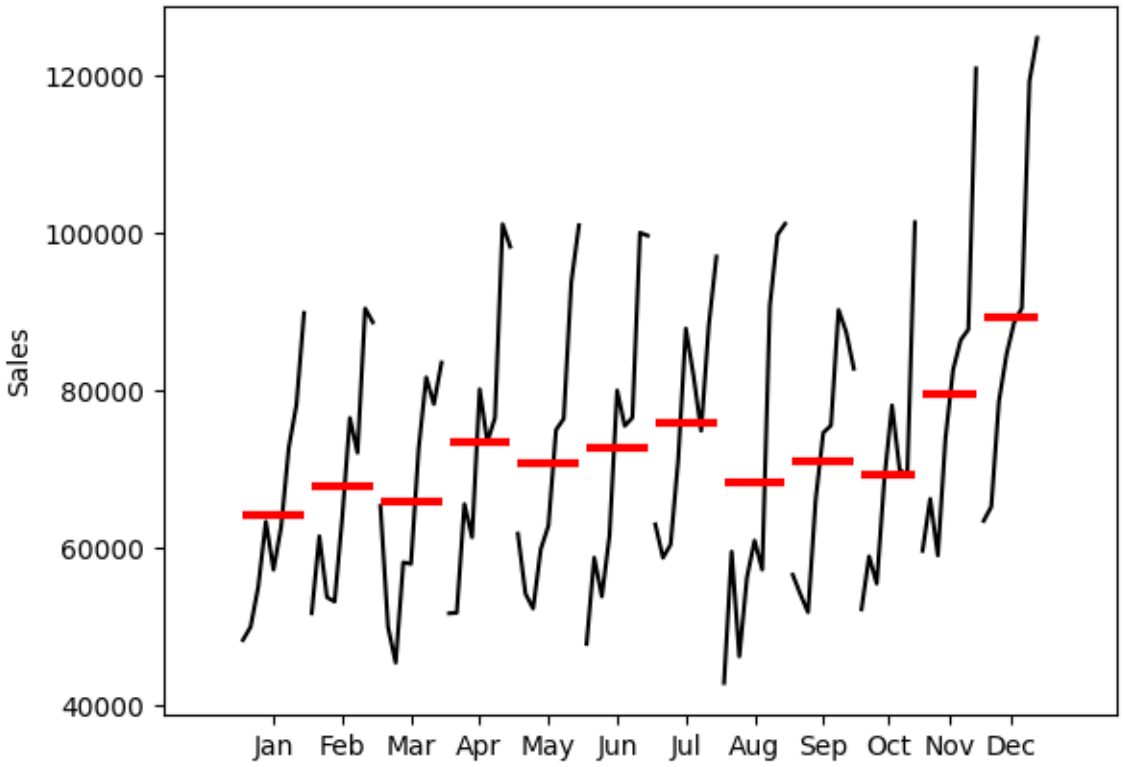


**Figure 4.2** Monthly sales patterns with seasonal averages

(Source: Author)

The second line chart illustrates the monthly sales pattern over several years, emphasizing the average sales for each month through red horizontal markers. The data reveals distinct seasonal trends and variability within each month.

Overall, November and December consistently record the highest average sales, surpassing 100,000. In contrast, January and February exhibit the lowest averages, typically around 60,000. Mid-year months, such as May to July, show moderate sales figures, with averages ranging between 70,000 and 90,000.

Despite the monthly averages, significant fluctuations are evident within most months. March, June, and September display notable variability, with sales frequently deviating from the monthly mean. Conversely, November and December are characterized by relatively stable high sales, suggesting a peak season for consumer activity.

In conclusion, the chart highlights a clear seasonality in sales, with lower figures at the beginning of the year, moderate growth in mid-year months, and a pronounced peak during the final quarter. The recurring patterns suggest potential factors like holidays or end-of-year events influencing consumer behavior.

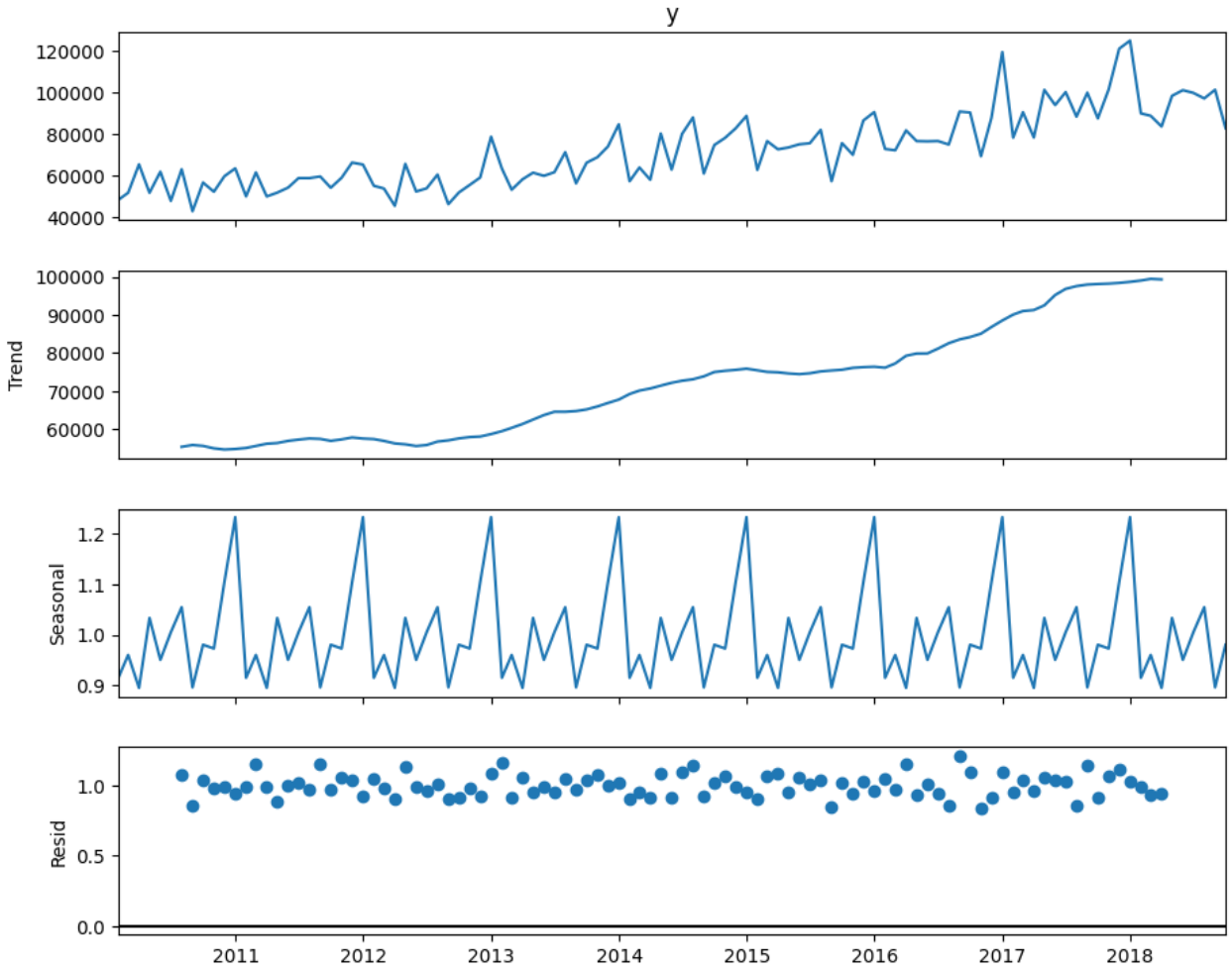


**Figure 4.3** Seasonal decomposition of total sales over time

(Source: Author)

The third chart illustrates the seasonal decomposition of total sales data from 2010 to 2018, divided into four components: observed values, trends, seasonal patterns, and residuals. The decomposition provides insights into the underlying trends and periodic sales behavior over time.

The observed data indicate a steady increase in sales over the eight-year period, with occasional fluctuations. This is corroborated by the trend component, which shows a gradual upward trajectory, particularly after 2015, when sales exceeded 100,000 on average.

Seasonal patterns are evident, with regular peaks and troughs occurring each year. These fluctuations suggest consistent consumer behavior, with specific months experiencing higher or lower sales. The residuals, representing unexplained variations, are relatively small and scattered, indicating that the trend and seasonality explain most of the data.

In summary, the chart highlights a clear upward trend in sales alongside strong seasonal

patterns. The minimal residual variation suggests that the decomposition effectively captures the key elements influencing sales over time.

*4.1.4.1. ACF plot*

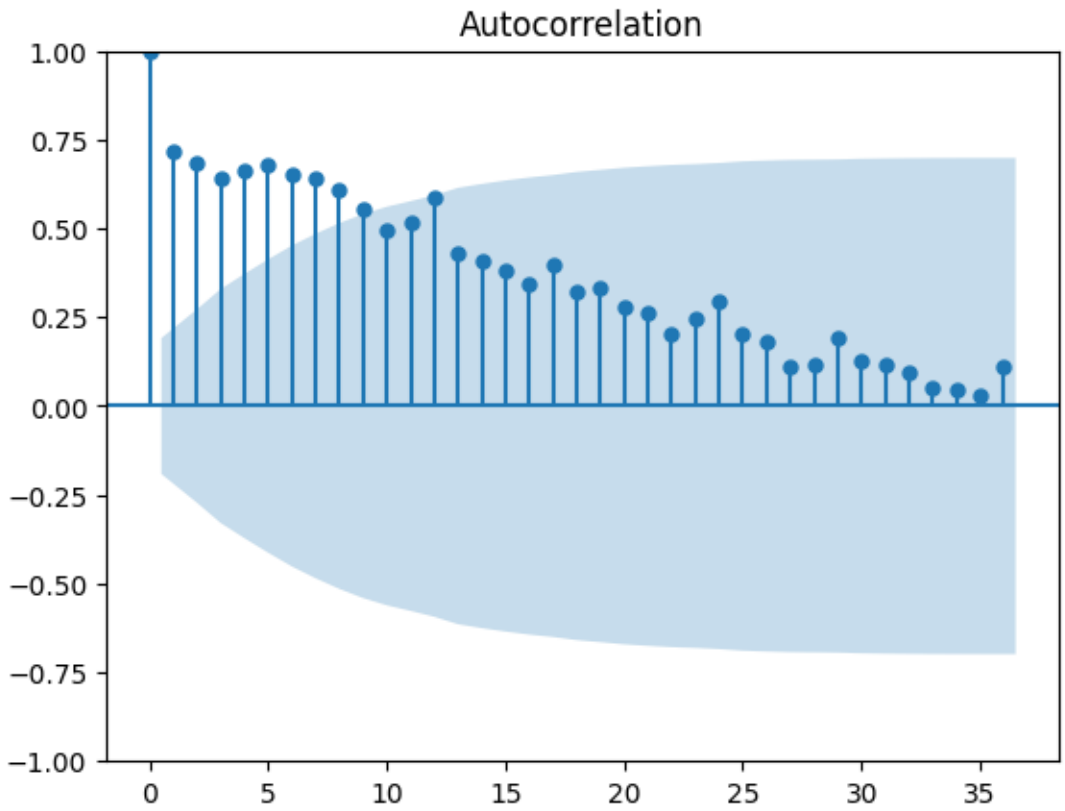


**Figure 4.4** ACF plot

The autocorrelation plot presented above illustrates the temporal dependency patterns of the sales data, which include historical daily sales information for a specific item (as defined by the column total_sales). The x-axis represents the lag values (in period), indicating the time intervals between observations. At the same time, the y-axis denotes the autocorrelation coefficients, which measure the strength and direction of the relationship between current sales and past sales values. The vertical lines indicate the magnitude of autocorrelation at each lag, and the shaded blue area represents the 95% confidence interval, within which correlations are deemed statistically insignificant.

The ACF plot reveals that sales exhibit a high degree of positive autocorrelation for shorter lags, with coefficients gradually decaying as the lag increases. This pattern indicates that sales data demonstrate persistence, meaning that current sales are strongly influenced by recent past sales, suggesting the presence of underlying patterns such as seasonality or trend. The autocorrelation coefficients decrease progressively but remain statistically significant for several lags before eventually falling within the confidence interval, which implies that random noise dominates as lag values increase.

This plot is critical for time series analysis, particularly in determining the structure of appropriate forecasting models. The high initial autocorrelation supports the use of autoregressive (AR) components, while the gradual decay indicates the potential for trend-based or seasonal models, such as ARIMA or SARIMA. The statistical insights derived from this plot can guide the development of robust predictive models to optimize sales forecasting, directly supporting strategic decision-making in inventory management and demand planning. The systematic evaluation of lag effects ensures that model parameters accurately capture temporal dependencies, thereby enhancing forecasting reliability and operational efficiency.

*4.1.4.2. PACF plot*

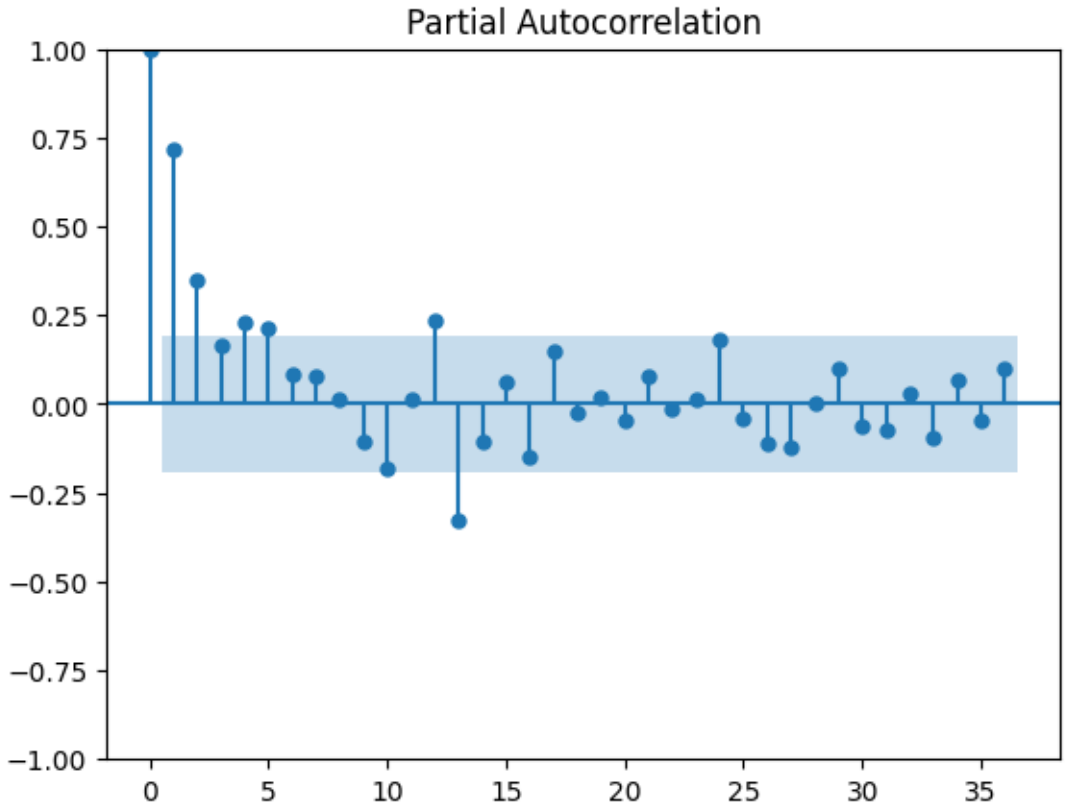


**Figure 4.5** PACF plot

The partial autocorrelation plot (PACF) shown above provides insights into the direct relationship between the sales data and its lagged values, eliminating the influence of intermediate lags. The x-axis represents the lag values (in period). At the same time, the y-axis denotes the partial autocorrelation coefficients, measuring the correlation strength between the current observation and past values after accounting for the effect of other lags. The vertical bars correspond to the partial autocorrelation at each lag, and the shaded blue region reflects the 95% confidence interval, indicating statistical insignificance when bars fall within this range.

This PACF plot highlights a **strong partial autocorrelation at lag 1**, which gradually diminishes as the lag increases. The significance of lag 1 indicates that the sales data exhibit short-term dependence, meaning that today's sales are closely influenced by the preceding day. Other statistically significant lags (those extending beyond the confidence interval) provide evidence of additional temporal dependencies, which may result from trends, seasonality, or other structural patterns in the data.

The PACF plot is instrumental in determining the order of the autoregressive (AR) component for time series forecasting models. For instance, the significant drop in partial autocorrelation beyond lag 1 suggests that an AR(1) or similar low-order model may sufficiently capture the time series' temporal structure. Combined with the autocorrelation plot (ACF), the PACF plot identifies model parameters for ARIMA or SARIMA models, frequently employed in sales forecasting. These insights contribute to optimizing inventory management and improving business decision-making by leveraging accurate and interpretable forecasts.

## 4.2. EXPERIMENTAL RESULTS AND DISCUSSION

Because the PyCaret library does not provide a plot_model() function to generate visualization results, the bar charts below are generated by running code in Python (see Appendix D).

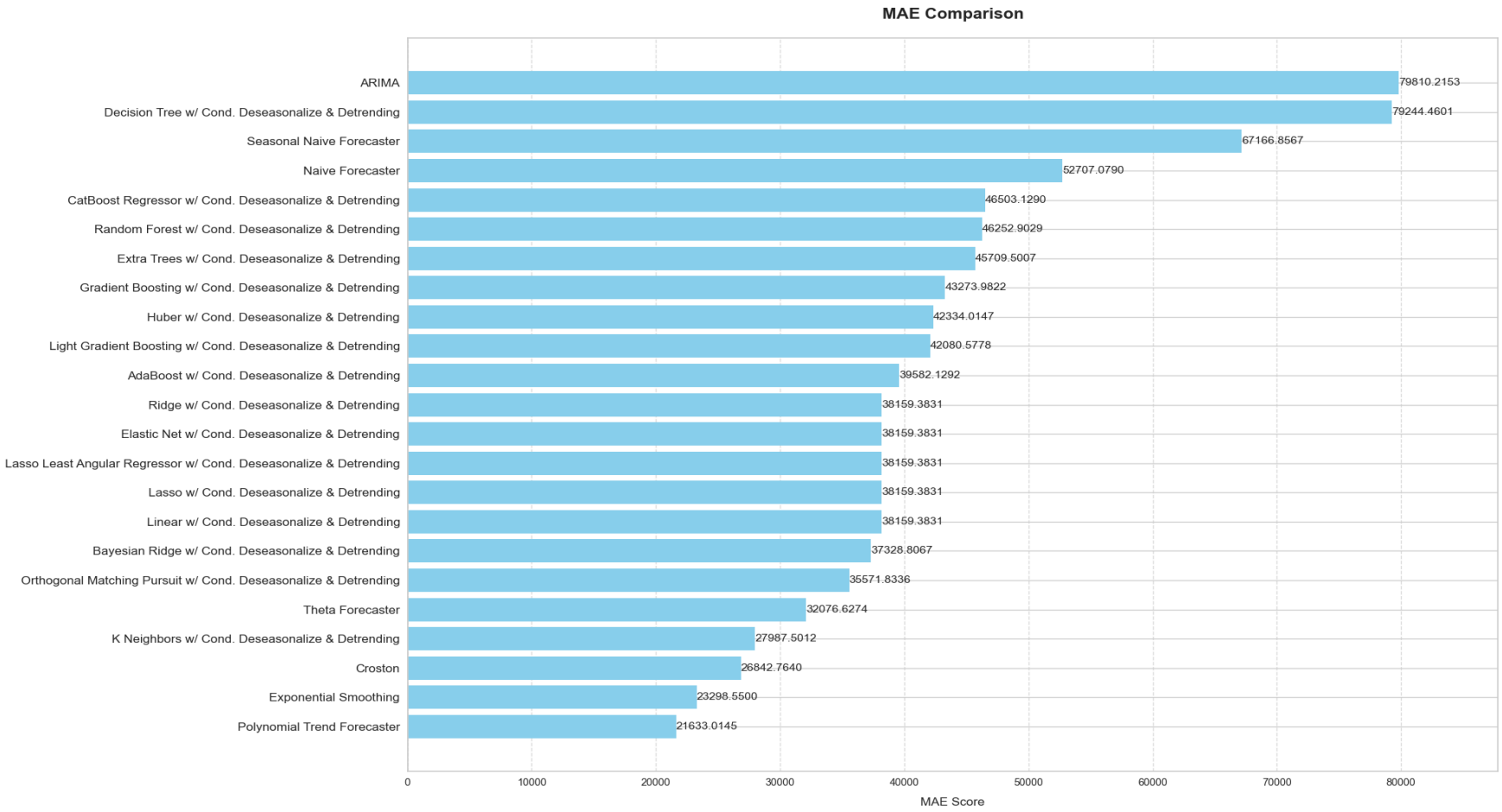


**Figure 4.6** MAE Comparison

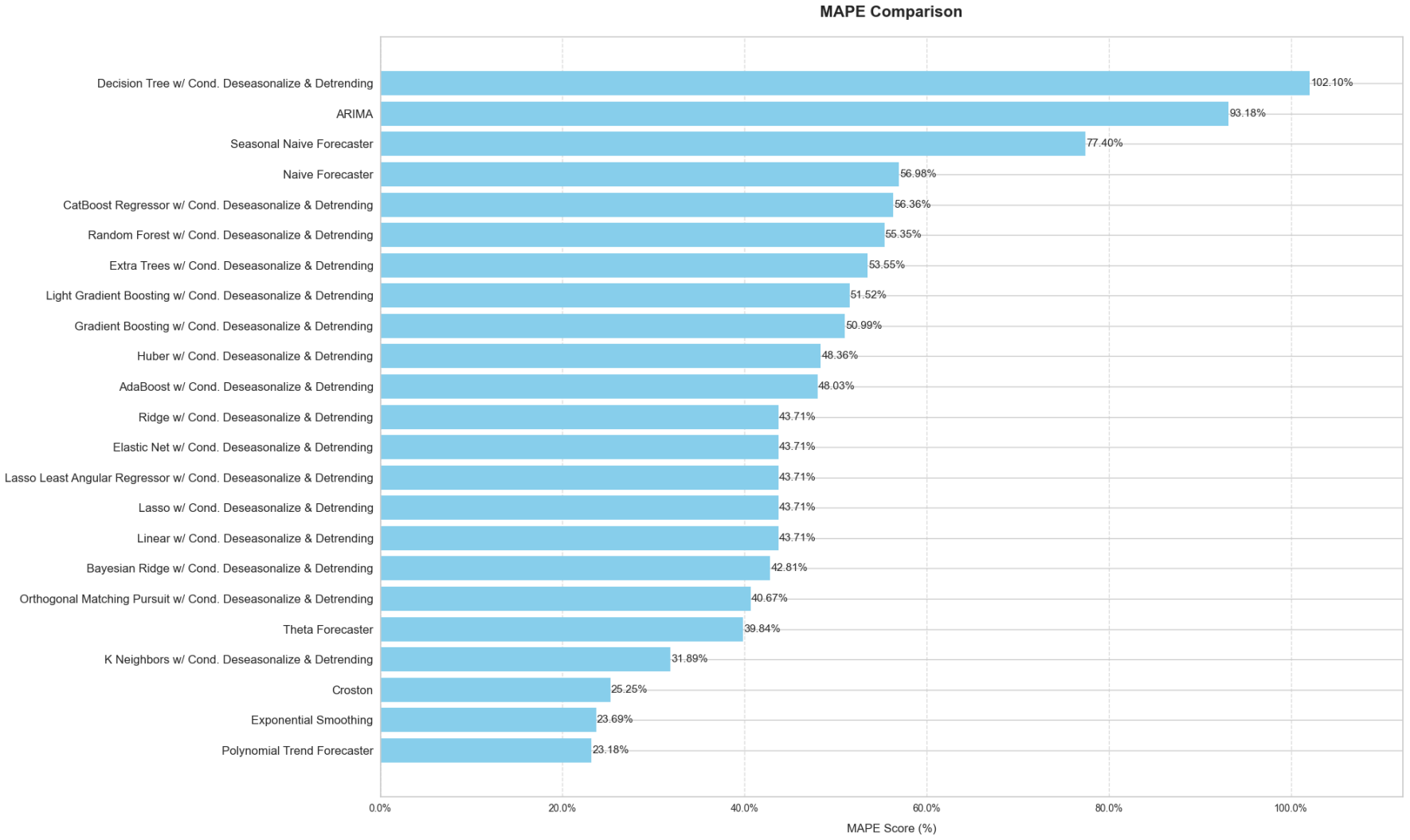


**Figure 4.7** MAPE Comparison

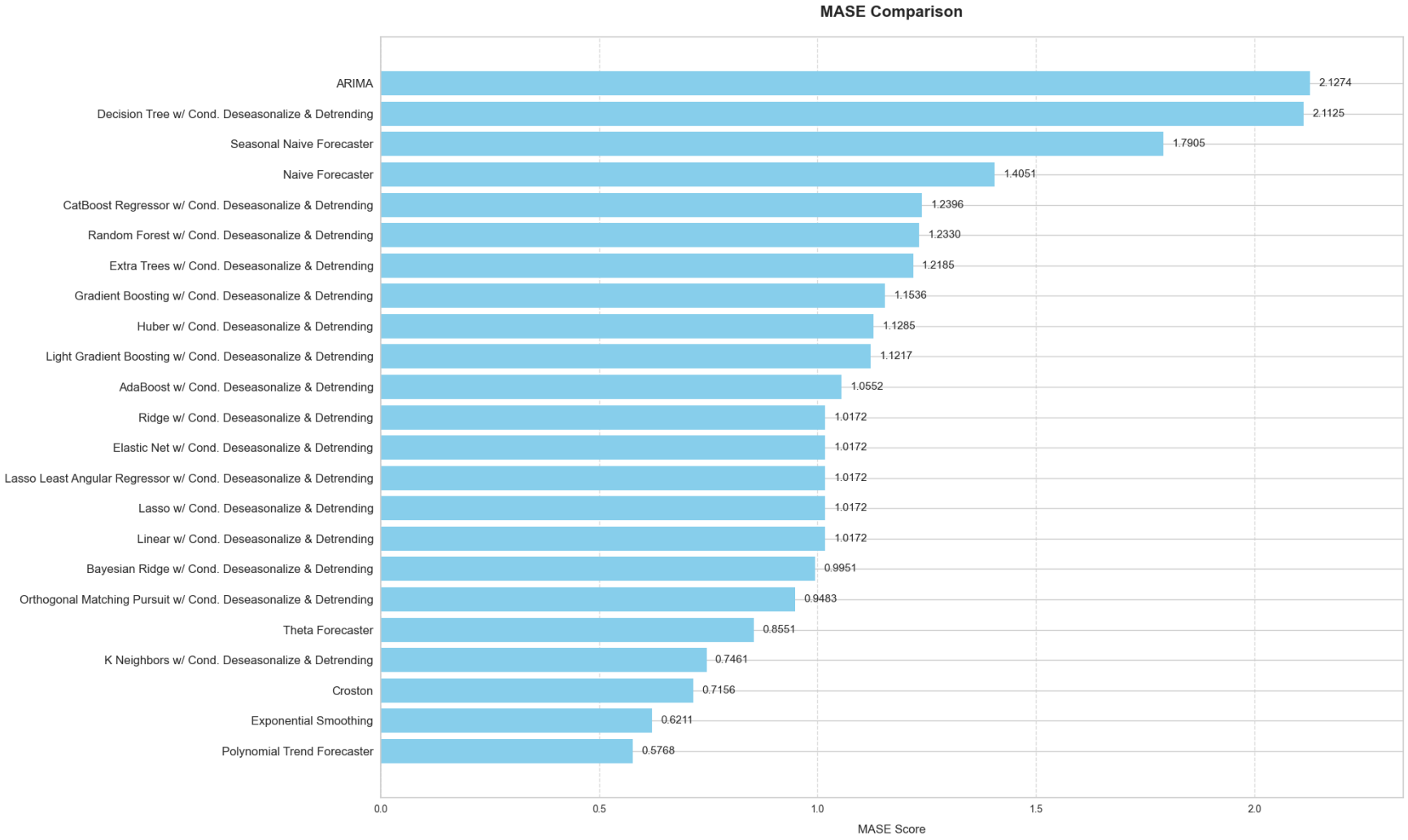


**Figure 4.8** MASE Comparison

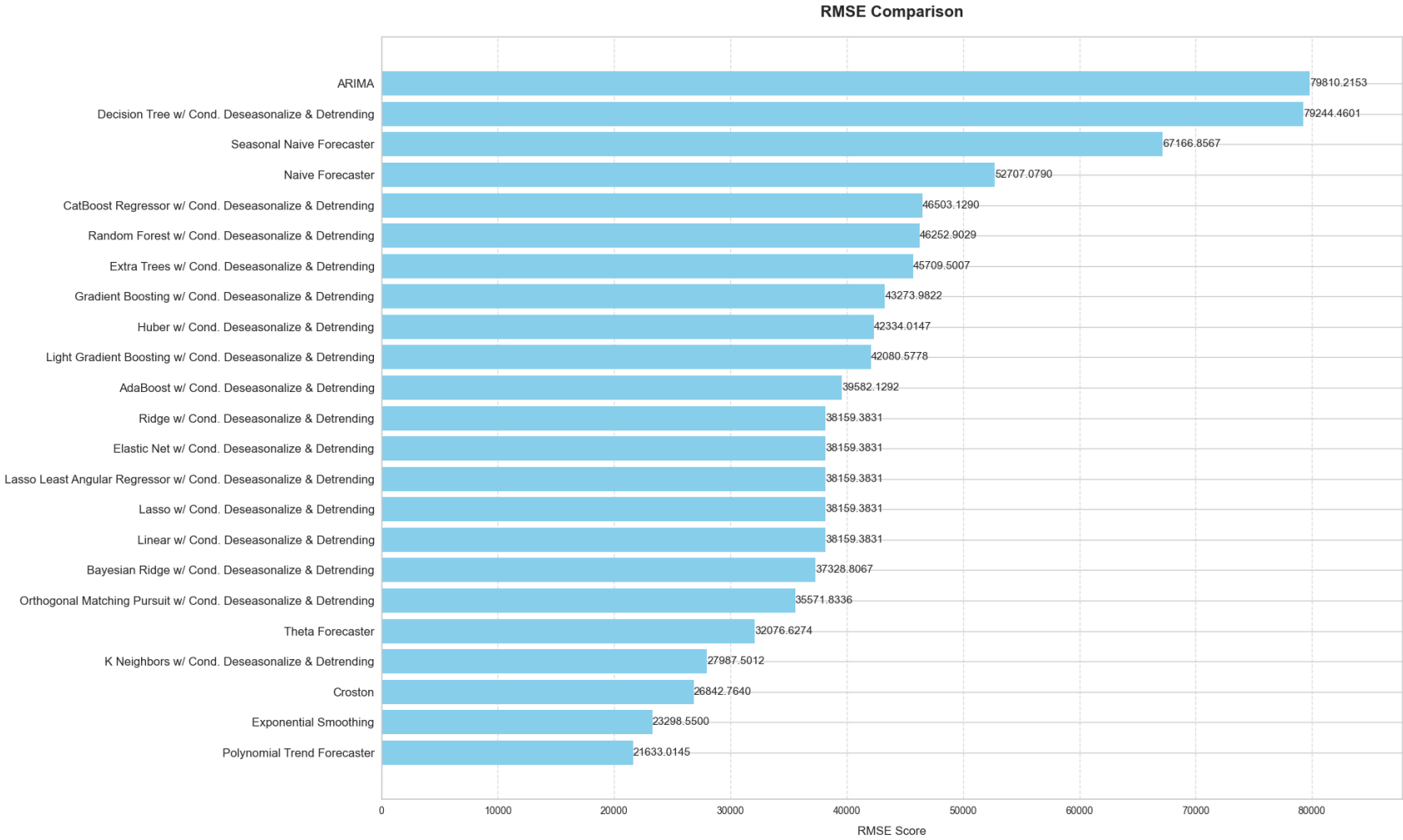


**Figure 4.9** RMSE Comparison

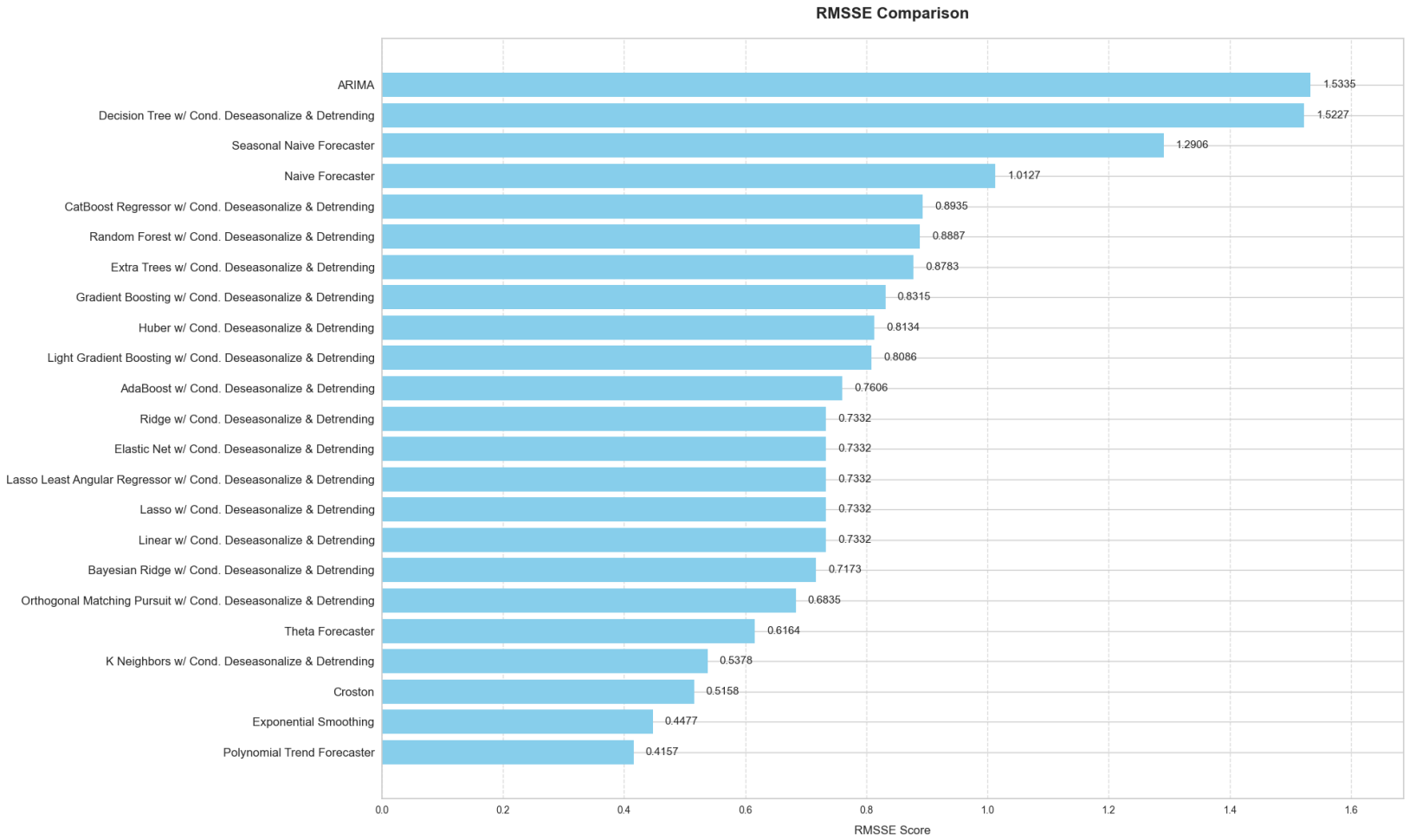


**Figure 4.10** RMSSE Comparison

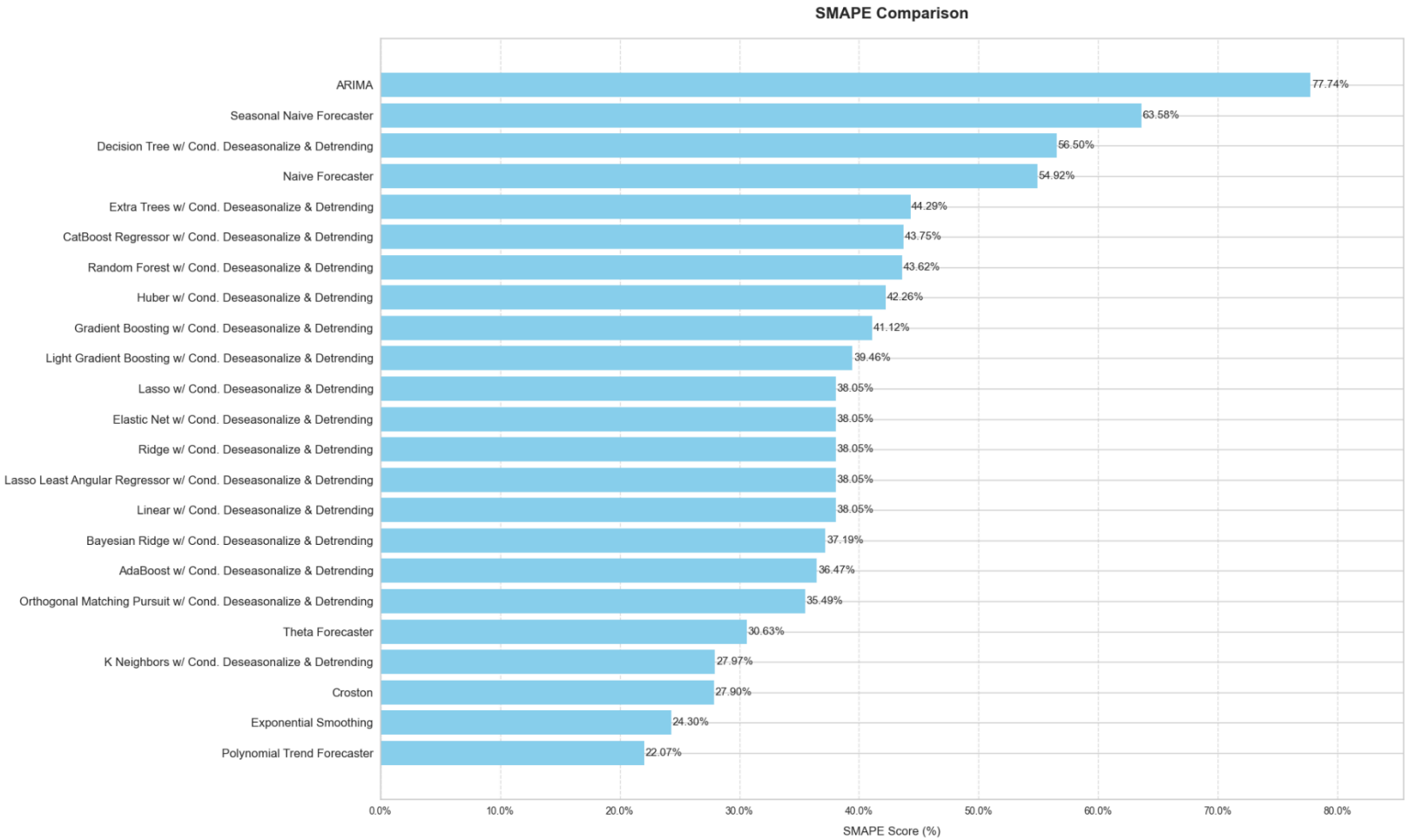


**Figure 4.11** SMAPE Comparison

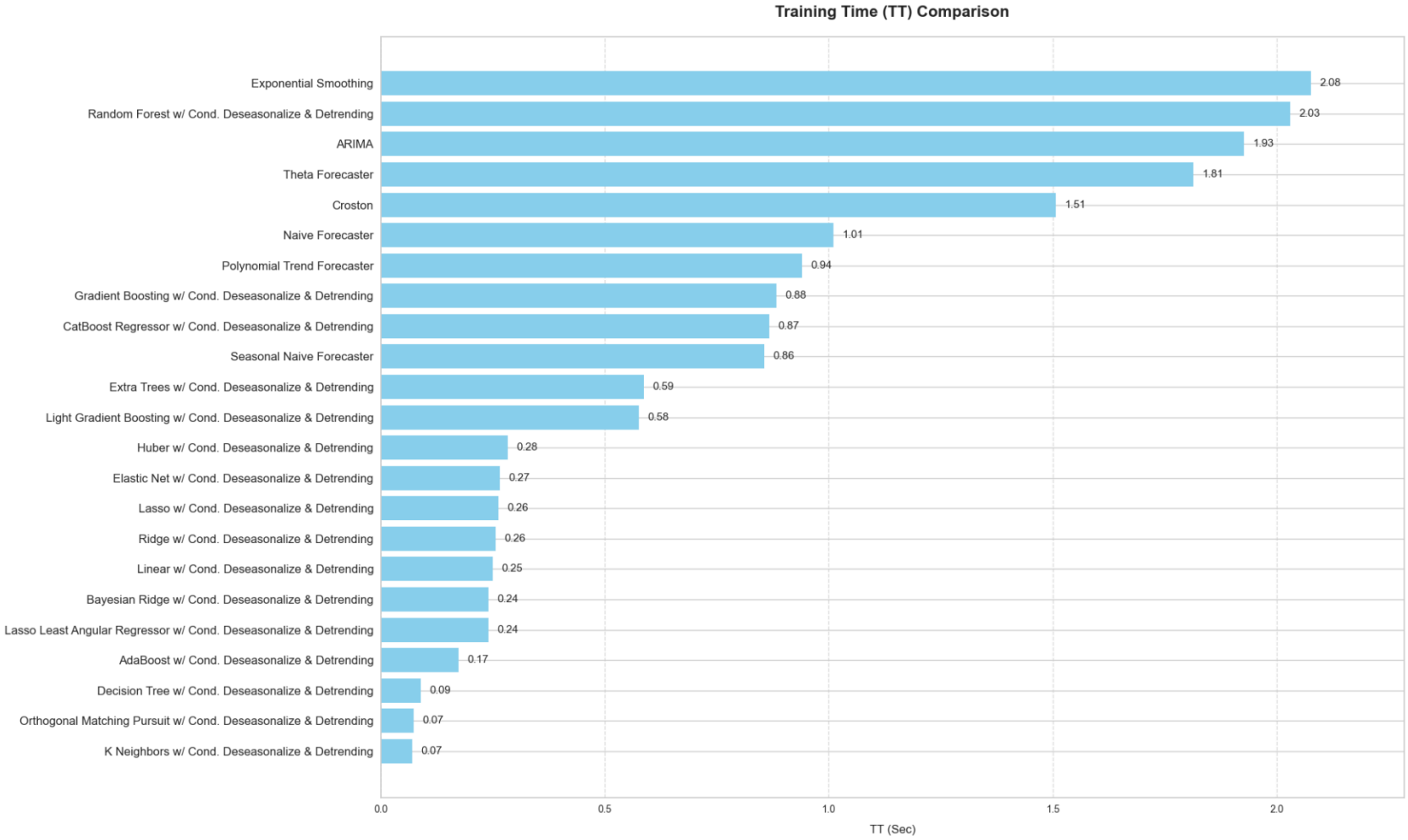


**Figure 4.12** Training Time (TT) Comparison

The bar charts above compare the performance of various forecasting models using different error metrics. Below are **the key observations**:

MASE and RMSSE:

**Top Performing Models**: Polynomial Trend Forecaster, Exponential Smoothing, Croston.

**Observation**: The Polynomial Trend Forecaster achieves the lowest MASE and RMSSE values, indicating strong accuracy in minimizing scaled errors.

MAE and RMSE:

**Top Performing Models**: Polynomial Trend Forecaster, Exponential Smoothing, Croston.

**Observation**: The Polynomial Trend Forecaster has the lowest MAE and RMSE, suggesting it effectively minimizes absolute and squared errors.

MAPE and SMAPE:

**Top Performing Models**: Polynomial Trend Forecaster, Exponential Smoothing, Croston.

**Observation**: These models show the lowest MAPE and SMAPE values, making them the most reliable in handling percentage-based errors.

TT (Training Time in Seconds):

**Fastest Models**: K Neighbors with Conditional Deseasonalization & Detrending, Orthogonal Matching Pursuit with Conditional Deseasonalization & Detrending, Decision Tree with Conditional Deseasonalization & Detrending.

**Observation**: These models train the fastest, making them computationally efficient for large-scale forecasting tasks.

**Recommendations**:

**Best Overall Model**: Polynomial Trend Forecaster
**Reason**: It consistently performs well across multiple error metrics while maintaining accuracy.

Runner-Up Models:

**Exponential Smoothing**: Strong accuracy across metrics with low computation time.

**Croston**: Competitive performance, particularly in percentage-based and scaled error metrics.

| | Model | MASE | RMSSE | MAE | RMSE | MAPE | SMAPE | TT (Sec) |
|---|---|---|---|---|---|---|---|---|
| polytrend | Polynomial Trend Forecaster | 0,5768 | 0,4157 | 21633,0145 | 21633,0145 | 0,2318 | 0,2207 | 0,94 |
| exp_smooth | Exponential Smoothing | 0,6211 | 0,4477 | 23298,5500 | 23298,5500 | 0,2369 | 0,2430 | 2,08 |
| croston | Croston | 0,7156 | 0,5158 | 26842,7640 | 26842,7640 | 0,2525 | 0,2790 | 1,51 |
| knn_cds_dt | K Neighbors w/ Cond. Deseasonalize & Detrending | 0,7461 | 0,5378 | 27987,5012 | 27987,5012 | 0,3189 | 0,2797 | 0,07 |
| theta | Theta Forecaster | 0,8551 | 0,6164 | 32076,6274 | 32076,6274 | 0,3984 | 0,3063 | 1,81 |
| omp_cds_dt | Orthogonal Matching Pursuit w/ Cond. Deseasonalize & Detrending | 0,9483 | 0,6835 | 35571,8336 | 35571,8336 | 0,4067 | 0,3549 | 0,07 |
| br_cds_dt | Bayesian Ridge w/ Cond. Deseasonalize & Detrending | 0,9951 | 0,7173 | 37328,8067 | 37328,8067 | 0,4281 | 0,3719 | 0,24 |
| en_cds_dt | Elastic Net w/ Cond. Deseasonalize & Detrending | 1,0172 | 0,7332 | 38159,3831 | 38159,3831 | 0,4371 | 0,3805 | 0,27 |
| ridge_cds_dt | Ridge w/ Cond. Deseasonalize & Detrending | 1,0172 | 0,7332 | 38159,3831 | 38159,3831 | 0,4371 | 0,3805 | 0,26 |
| lasso_cds_dt | Lasso w/ Cond. Deseasonalize & Detrending | 1,0172 | 0,7332 | 38159,3831 | 38159,3831 | 0,4371 | 0,3805 | 0,26 |
| lr_cds_dt | Linear w/ Cond. Deseasonalize & Detrending | 1,0172 | 0,7332 | 38159,3831 | 38159,3831 | 0,4371 | 0,3805 | 0,25 |
| llar_cds_dt | Lasso Least Angular Regressor w/ Cond. Deseasonalize & Detrending | 1,0172 | 0,7332 | 38159,3831 | 38159,3831 | 0,4371 | 0,3805 | 0,24 |
| ada_cds_dt | AdaBoost w/ Cond. Deseasonalize & Detrending | 1,0552 | 0,7606 | 39582,1292 | 39582,1292 | 0,4803 | 0,3647 | 0,17 |
| lightgbm_cds_dt | Light Gradient Boosting w/ Cond. Deseasonalize & Detrending | 1,1217 | 0,8086 | 42080,5778 | 42080,5778 | 0,5152 | 0,3946 | 0,58 |
| huber_cds_dt | Huber w/ Cond. Deseasonalize & Detrending | 1,1285 | 0,8134 | 42334,0147 | 42334,0147 | 0,4836 | 0,4226 | 0,28 |
| gbr_cds_dt | Gradient Boosting w/ Cond. Deseasonalize & Detrending | 1,1536 | 0,8315 | 43273,9822 | 43273,9822 | 0,5099 | 0,4112 | 0,88 |
| et_cds_dt | Extra Trees w/ Cond. Deseasonalize & Detrending | 1,2185 | 0,8783 | 45709,5007 | 45709,5007 | 0,5355 | 0,4429 | 0,59 |
| rf_cds_dt | Random Forest w/ Cond. Deseasonalize & Detrending | 1,2330 | 0,8887 | 46252,9029 | 46252,9029 | 0,5535 | 0,4362 | 2,03 |
| catboost_cds_dt | CatBoost Regressor w/ Cond. Deseasonalize & Detrending | 1,2396 | 0,8935 | 46503,1290 | 46503,1290 | 0,5636 | 0,4375 | 0,87 |
| naive | Naive Forecaster | 1,4051 | 1,0127 | 52707,0790 | 52707,0790 | 0,5698 | 0,5492 | 1,01 |
| snaive | Seasonal Naive Forecaster | 1,7905 | 1,2906 | 67166,8567 | 67166,8567 | 0,7740 | 0,6358 | 0,86 |
| dt_cds_dt | Decision Tree w/ Cond. Deseasonalize & Detrending | 2,1125 | 1,5227 | 79244,4601 | 79244,4601 | 1,0210 | 0,5650 | 0,09 |
| arima | ARIMA | 2,1274 | 1,5335 | 79810,2153 | 79810,2153 | 0,9318 | 0,7774 | 1,93 |

**Figure 4.13** Summary Comparison of seven metrics

**Polynomial Trend Forecaster Stands Out as the Most Accurate Model**
With the lowest MASE (0.5768), RMSSE (0.4157), MAE (21633.0145), and RMSE (21633.0145), the Polynomial Trend Forecaster demonstrates superior performance in minimizing both absolute and scaled errors. This suggests that a polynomial trend captures the underlying data structure effectively, likely benefiting from flexibility in modeling long-term patterns.

**Exponential Smoothing and Croston Also Show Competitive Performance**
While Exponential Smoothing follows closely behind the Polynomial Trend Forecaster with respectable accuracy across all metrics, its higher computation time (2.08s) raises concerns about efficiency. Croston, a method specialized for intermittent demand forecasting, performs well but struggles with slightly higher error rates than the top-performing models.

**Machine Learning Models with Conditional Deseasonalization & Detrending Lag Behind**
Models like K-Neighbors, Theta Forecaster, and Decision Trees with deseasonalization and detrending exhibit significantly higher error rates. Their MAPE and SMAPE scores indicate weaker generalization, suggesting that traditional statistical models outperform them for this specific dataset.

**The Trade-Off Between Accuracy and Computation Time is Clear**
The fastest models, such as K-Neighbors (0.07s) and Decision Tree (0.09s), sacrifice accuracy for efficiency. In contrast, the more accurate models (Polynomial Trend, Exponential Smoothing, Croston) require longer training times. This trade-off highlights the importance of choosing models based on specific forecasting needs—whether prioritizing speed or accuracy.

**Traditional Time Series Methods Still Dominate Over Machine Learning Models**
The results reinforce a well-known reality in time series forecasting—classical statistical

models, particularly Polynomial Trend Forecaster, Exponential Smoothing, and Croston, outperform complex machine learning techniques in accuracy. This emphasizes the necessity of careful feature engineering and transformation when applying machine learning to time series problems.

# CHAPTER 5
# CONCLUSION

## 5.1. SUMMARY

This capstone project, titled "Application of PyCaret's AutoML Framework for Optimizing Sales Forecasting in Time Series Analysis," was undertaken to evaluate the effectiveness of various forecasting models. The research utilized a substantial real-world dataset comprising 412,357 daily retail sales records from a company in Bosnia and Herzegovina, covering the period from January 2010 to September 2018.

The PyCaret AutoML framework served as the primary tool for this investigation. The univariate sales data underwent preprocessing, which included interpolation to handle missing values and conditional deseasonalization and detrending. Subsequently, a comprehensive range of traditional statistical models (such as ARIMA, Exponential Smoothing, and Croston) and machine learning ensembles (including tree-based methods) were systematically trained and benchmarked. Performance was assessed using seven distinct error metrics: MASE, RMSSE, MAE, RMSE, MAPE, SMAPE, along with Training Time (TT).

The empirical results consistently indicated that the Polynomial Trend Forecaster demonstrated the most robust performance across these metrics. Specifically, it achieved the lowest MASE (0.5768), RMSSE (0.4157), MAE (21633.01), and a leading MAPE of 23.18%. It was observed that, for this particular dataset, the machine learning models, despite preprocessing, generally exhibited higher error rates. A distinct trade-off between forecasting accuracy and computational time was also evident; for instance, K-Neighbors showed a very low training time (TT) of 0.07 seconds but with correspondingly lower accuracy.

In conclusion, this study highlights the utility of the PyCaret framework for rigorous model comparison in sales forecasting. The findings suggest that for the specific dataset analyzed, an optimized traditional model, the Polynomial Trend Forecaster, provided superior forecasting accuracy when compared to the evaluated machine learning approaches.

## 5.2. LIMITATION AND FUTURE CONSIDERATION

One of the primary limitations of this study is its focus on univariate time series forecasting. While this approach provides valuable insights into sales trends based solely on historical data, it does not account for external factors such as promotions, holidays, competitor pricing, or economic shifts, which can significantly impact demand. Future research should explore multivariate forecasting models that integrate additional variables to improve prediction accuracy.

Another limitation is the computational constraints associated with model selection and training. Although PyCaret’s AutoML simplifies the process, running multiple machine

learning models, particularly ensemble methods, can be resource-intensive. Businesses with limited computational power may face challenges deploying these models in real-time decision-making environments. Future studies should investigate cloud-based solutions or optimized model deployment strategies to address this issue.

Additionally, model interpretability remains a concern. While machine learning models outperform traditional statistical methods in accuracy, they function as "black boxes," making it difficult to understand why a model makes specific predictions. Businesses that require transparency in forecasting should consider explainable AI techniques to enhance trust and usability in decision-making.

Lastly, this study primarily evaluates forecasting performance based on historical data, assuming that past patterns continue into the future. However, unexpected disruptions, such as economic crises or supply chain shocks, can cause drastic demand fluctuations. Future research should explore adaptive and real-time forecasting models that can dynamically adjust predictions in response to sudden market changes.

# REFERENCES


[1] Abhishek, B., & Oliver, J. R. (2024). Enhanced sales forecasting model using textual search data: Fusing dynamics with big data. *International Journal of Research in Marketing*. https://doi.org/10.1016/j.ijresmar.2024.05.007

[2] Accelerated Componentwise Gradient Boosting Using Efficient Data Representation and Momentum-Based Optimization. (2022). *Journal of Computational and Graphical Statistics*, *32*(2), 631-641. https://doi.org/10.1080/10618600.2022.2116446

[3] Afiqah Bazlla Md, S., Aisyah Mat, J., Juhaida, I., Aszila, A., & Rozeleenda Abdul, R. (2023). Sales Forecasting Using Convolution Neural Network. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *30*(3), 290-301. https://doi.org/10.37934/araset.30.3.290301

[4] Ali, M. (2020). *PyCaret: An open source, low-code machine learning library in Python*. In https://www.pycaret.org

[5] Ali, S., & Chyi Lin, L. (2024). The non-linear dynamics of South Australian regional housing markets: A machine learning approach. *Applied Geography*. https://doi.org/10.1016/j.apgeog.2024.103248

[6] Aloorravi, S. (2024). *Mastering Time Series Analysis and Forecasting with Python: Bridging Theory and Practice Through Insights, Techniques, and Tools for Effective Time Series Analysis in Python (English Edition)*. Orange Education Pvt Limited. https://books.google.com.vn/books?id=_DH9EAAAQBAJ

[7] Andrea, L. S., Timothy, D., Sallie-Anne, P., & Sallie-Anne, P. (2021). Interrupted time series analysis using autoregressive integrated moving average (ARIMA) models: a guide for evaluating large-scale health interventions. *BMC Medical Research Methodology*, *21*(1), 1-12. https://doi.org/10.1186/S12874-021-01235-8

[8] Andreas, C. B., & Peter, H. D. (2011). An operational definition of a statistically meaningful trend. *PLOS ONE*, *6*(4). https://doi.org/10.1371/JOURNAL.PONE.0019241

[9] Andrés, A., Andrés, A., Zhishen, H., Emilio, F., & Kristina, L. (2020). Predictability limit of partially observed systems. *Scientific Reports*, *10*(1), 20427. https://doi.org/10.1038/S41598-020-77091-1

[10] Antonio, M., Diez, R. M., Insua, D. R., & M¨ller, P. (2005). Bayesian Analysis of Nonlinear Autoregression Models Based on Neural Networks. *Neural Computation*, *17*(2), 453-485. https://doi.org/10.1162/0899766053011537

[11] Asher, C., Russell, J. L., & Sarah, E. M. (2014). Forecasting Sales: A Model and Some Evidence from the Retail Industry*. *Contemporary Accounting Research*, *31*(2), 581-608. https://doi.org/10.1111/1911-3846.12040

[12] Automated Machine Learning for Time Series Prediction. (2022).

[13] Balgobin, N., & Joseph, D. P. (1997). A Bayesian analysis of autoregressive time series panel data. *Journal of Business & Economic Statistics*, *15*(3), 328-334. https://doi.org/10.1080/07350015.1997.10524710

[14] Benjamin, R., Michael, R., Guntram, W., Gregor, F. F., & Ursula, G. (2019). Estimating Parameters From Multiple Time Series of Population Dynamics Using Bayesian Inference. *Frontiers in Ecology and Evolution*, *6*. https://doi.org/10.3389/FEVO.2018.00234
[15] Berk, G., & Mustafa Gokce, B. (2021). Randomized trees for time series representation and similarity. *Pattern Recognition*, *120*, 108097. https://doi.org/10.1016/J.PATCOG.2021.108097
[16] Borucka, A. (2023). Seasonal Methods of Demand Forecasting in the Supply Chain as Support for the Company's Sustainable Growth. *Sustainability*, *15*(9).
[17] Bradley, E., Trevor, H., Iain, M. J., Robert, T., Hemant, I., Keith, K., Jean-Michel, L., Jean-Michel, L., Pascal, M., Pascal, M., David, M., David, M., Greg, R., Greg, R., Saharon, R., Saharon, R., Ji, Z., Robert, A. S., Berwin, A. T., & Sanford, W. (2004). Least angle regression. *Annals of Statistics*, *32*(2), 407-499. https://doi.org/10.1214/009053604000000067
[18] Cagatay, C., Kaan, E., Begum, A., & Akhan, A. (2019). Benchmarking of Regression Algorithms and Time Series Analysis Techniques for Sales Forecasting. *7*(1), 20-26. https://doi.org/10.17694/BAJECE.494920
[19] Candice, B., Anna, C., & Gonzalo, M.-M. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, *54*(3), 1937-1967. https://doi.org/10.1007/S10462-020-09896-5
[20] Casolaro, A., Capone, V., Iannuzzo, G., & Camastra, F. (2023). Deep learning for time series forecasting: advances and open problems. *Information*, *14*(11), 598. https://doi.org/10.3390/info14110598
[21] Cheng-Yu, H., Ke-Sheng, C., & Chihhsien, A. (2023). Utilizing the Random Forest Method for Short-Term Wind Speed Forecasting in the Coastal Area of Central Taiwan. *Energies*, *16*(3), 1374-1374. https://doi.org/10.3390/en16031374
[22] Chenxi, N., Haihong, H., Peipei, C., Qingdi, K., Shiyao, T., Kim Tiow, O., & Zhifeng, L. (2024). Light Gradient Boosting Machine (LightGBM) to forecasting data and assisting the defrosting strategy design of refrigerators. *International Journal of Refrigeration-revue Internationale Du Froid*. https://doi.org/10.1016/j.ijrefrig.2024.01.025
[23] Choi, T.-M., Wallace, S. W., & Wang, Y. (2018). Big Data Analytics in Operations Management. *Production and Operations Management*, *27*(10), 1868-1883. https://doi.org/https://doi.org/10.1111/poms.12838
[24] Christopher, D. K. (2015). Nonlinear Regression Huber–Kalman Filtering and Fixed-Interval Smoothing. *Journal of Guidance Control and Dynamics*, *38*(2), 322-330. https://doi.org/10.2514/1.G000799
[25] A comprehensive evaluation of statistical, machine learning and deep learning models for time series prediction. (2022).
[26] Construction Time Series Forecasting Using Univariate Time Series Models. (2023). In (pp. 7-43). https://doi.org/10.1007/978-3-031-27292-9_2
[27] Craigmile, P. F., Guttorp, P., & Percival, D. B. (2005). Wavelet-based parameter estimation for polynomial contaminated fractionally differenced processes. *IEEE*

*Transactions on Signal Processing*, *53*(8), 3151-3161. https://doi.org/10.1109/TSP.2005.851111

[28] Cyril, N., Samuel Charles, G., Gillian, N., P, A., A, F., Webnda, F., Adeyinka, O., & Sylla, N. (2024). Advancing Retail Predictions: Integrating Diverse Machine Learning Models for Accurate Walmart Sales Forecasting. *Asian Journal of Probability and Statistics*, *26*(7), 1-23. https://doi.org/10.9734/ajpas/2024/v26i7626

[29] Daifeng, L., Fengyun, G., Xin, L., Ruo, D., Dingquan, C., & Andrew David, M. (2023). Dynamic sales prediction with auto-learning and elastic-adjustment mechanism for inventory optimization. *Information Systems*, *119*, 102259-102259. https://doi.org/10.1016/j.is.2023.102259

[30] Daifeng, L., Xin, L., Fengyun, G., Ziyang, P., Dingquan, C., & Andrew David, M. (2023). A Universality-Distinction Mechanism-Based Multi-Step Sales Forecasting for Sales Prediction and Inventory Optimization. *Systems*, *11*(6), 311-311. https://doi.org/10.3390/systems11060311

[31] Dakuo, W., Josh, A., Justin, D. W., Erick, O., & Casey, D. (2021). AutoDS: Towards Human-Centered Automation of Data Science.

[32] David, G.-R., Matthew, M., Guilherme, A., Diego Furtado, S., & Anthony, B. (2024). Unsupervised feature based algorithms for time series extrinsic regression. *Data Mining and Knowledge Discovery*. https://doi.org/10.1007/s10618-024-01027-w

[33] Desalew Meseret, M., Holger, V., Alexander, K., Raj, C., Rohith, A. N. R., Alberto, M. S., Carmelo Conesa, G., & Evelyn, U. (2024). Streamflow Prediction with Time-Lag-Informed Random Forest and Its Performance Compared to SWAT in Diverse Catchments. *Water*, *16*(19), 2805-2805. https://doi.org/10.3390/w16192805

[34] Dhananjay, B., & Sivaraman, J. (2021). Analysis and classification of heart rate using CatBoost feature ranking model. *Biomedical Signal Processing and Control*, *68*, 102610. https://doi.org/10.1016/J.BSPC.2021.102610

[35] Dimitrios, E., Evangelos, S., Georgios, S., & Avi, A. (2023). Time series and regression methods for univariate environmental forecasting: An empirical evaluation. *Social Science Research Network*, 162580-162580. https://doi.org/10.2139/ssrn.4293044

[36] Donald, K. K. L., Ningyuan, C., & Hemant, I. (2021). Boosted nonparametric hazards with time-dependent covariates. *Annals of Statistics*, *49*(4), 2101-2128. https://doi.org/10.1214/20-AOS2028

[37] Elastic Net Penalized Quantile Regression Model and Empirical Mode Decomposition for Improving the Accuracy of the Model Selection. (2023). *IEEE Access*, *11*, 26152-26162. https://doi.org/10.1109/access.2023.3257032

[38] Evandro, K., & Flávio Augusto, Z. (2016). LASSO‐Type Penalties for Covariate Selection and Forecasting in Time Series. *Journal of Forecasting*, *35*(7), 592-612. https://doi.org/10.1002/FOR.2403

[39] Evangelos, S., & Fotios, P. (2023). On the update frequency of univariate forecasting models. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2023.08.056
[40] Everette, S. G. (1983). Automatic monitoring of forecast errors. *Journal of Forecasting*, *2*(1), 1-21. https://doi.org/10.1002/FOR.3980020103
[41] Everette, S. G., & David, G. D. (1980). Forecasting with exponential smoothing: some guidelines for model selection. *Decision Sciences*, *11*(2), 370-383. https://doi.org/10.1111/J.1540-5915.1980.TB01145.X
[42] Fatoumata, D., & Christine, S. (2021a). Analysis and modeling to forecast in time series: a systematic review. In.
[43] Fatoumata, D., & Christine, S. (2021b). Time Series Analysis and Modeling to Forecast: a Survey. In.
[44] Felipe Rooke da, S., Alex Borges, V., Heder, S. B., Victor Aquiles Soares de Barros, A., Lucas Ribeiro, P., & Helio, J. C. B. (2022). Automated Machine Learning for Time Series Prediction.
[45] Feng, W., & Joey, S. A. (2023). Contrasting Univariate and Multivariate Time Series Forecasting Methods for Sales: A Comparative Analysis. *Applied science and innovative research*, *7*(2), p127-p127. https://doi.org/10.22158/asir.v7n2p127
[46] Firican, G. (2022). The history of machine learning. *LightsOnData*. https://www.lightsondata.com/the-history-of-machine-learning/
[47] Forecasting sales using online review and search engine data: A method based on PCA–DSFOA–BPNN. (2022). *International Journal of Forecasting*, *38*(3), 1005-1024. https://doi.org/10.1016/j.ijforecast.2021.07.010
[48] Fotios, P., Konstantinos, N., Georgios, P. S., & Vassilios, A. (2013). Empirical heuristics for improving intermittent demand forecasting. *Industrial Management and Data Systems*, *113*(5), 683-696. https://doi.org/10.1108/02635571311324142
[49] Geeta, C. (2023). Comparison of Imputation Methods for Univariate Time Series. *International Journal on Recent and Innovation Trends in Computing and Communication*, *11*(2s), 286-292. https://doi.org/10.17762/ijritcc.v11i2s.6148
[50] Generalized Orthogonal Matching Pursuit With Singular Value Decomposition. (2022). *IEEE Geoscience and Remote Sensing Letters*, *19*, 1-5. https://doi.org/10.1109/lgrs.2021.3086492
[51] George, W., Utku, E., Omar Abdallah, M., Sullaiman Musah, L., Tahir Cetin, A., & Oguzhan, T. (2024). Time Series Forecasting Utilizing Automated Machine Learning (AutoML): A Comparative Analysis Study on Diverse Datasets. *Information*. https://doi.org/10.3390/info15010039
[52] Ghysels, E., Osborn, D. R., & Rodrigues, P. M. M. (2006). Chapter 13 Forecasting Seasonal Time Series. In G. Elliott, C. W. J. Granger, & A. Timmermann (Eds.), *Handbook of Economic Forecasting* (Vol. 1, pp. 659-711). Elsevier. https://doi.org/https://doi.org/10.1016/S1574-0706(05)01013-X
[53] Giacalone, M., Mattera, R., & Nissi, E. (2020). Economic indicators forecasting in presence of seasonal patterns: time series revision and prediction accuracy. *Quality & Quantity*, *54*(1), 67-84. https://doi.org/10.1007/s11135-019-00935-0

[54] Giovanni, C. (2023). Model Selection and Regularization. In (pp. 59-146). https://doi.org/10.1007/978-3-031-41337-7_3
[55] Goce, R., Goce, R., Wei, L., James, B., & James, B. (2013). Time Series Forecasting Using Distribution Enhanced Linear Regression. In (pp. 484-495). https://doi.org/10.1007/978-3-642-37453-1_40
[56] Guotong, Z., Giannakis, G. B., & Swami, A. (1996). On polynomial phase signals with time-varying amplitudes. *IEEE Transactions on Signal Processing*, *44*(4), 848-861. https://doi.org/10.1109/78.492538
[57] Gwo-Fong, L., & Lu-Hsien, C. (2005). Time series forecasting by combining the radial basis function network and the self-organizing map. *Hydrological Processes*, *19*(10), 1925-1937. https://doi.org/10.1002/HYP.5637
[58] H2O AutoML: Automatic Machine Learning — H2O 3.46.0.6 documentation. https://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html
[59] Hanan Wagih, E., & Mohamed Mamdouh, A. (2023). A Cognitive Analytics Framework for Improving Sales Prediction Accuracy. 89-92. https://doi.org/10.1109/niles59815.2023.10296687
[60] High-Dimensional Generalized Orthogonal Matching Pursuit with Singular Value Decomposition. (2023). *IEEE Geoscience and Remote Sensing Letters*, 1-1. https://doi.org/10.1109/lgrs.2023.3264623
[61] Hirotaka, S., Hiroki, Y., Kenichi, T., Hiroshi, K., Kimio, W., Masaharu, T., Hiroki, E., Tianchen, Z., Akihiko, O., Sakumi, K., Michio, S., Koichi, A., Tsuyoshi, W., & Kazama, J. J. (2024). Predicting CKD progression using time-series clustering and light gradient boosting machines. *Dental science reports*, *14*. https://doi.org/10.1038/s41598-024-52251-9
[62] Hong-Jie, X., & Xi-Zhao, W. (2023). Bounded exponential loss function based AdaBoost ensemble of OCSVMs. *Pattern Recognition*. https://doi.org/10.1016/j.patcog.2023.110191
[63] Hongbin, L., Hongchun, Z., Zhouhua, J., Huabing, L., Ce, Y., Hao, F., & Shucai, Z. (2024). A CatBoost‐Based Modeling Approach for Predicting End‐Point Carbon Content of Electric Arc Furnace. *Steel Research*. https://doi.org/10.1002/srin.202400053
[64] Howard, B., Sean, M., & Galit, S. (2007). Automated time series forecasting for biosurveillance. *Statistics in Medicine*, *26*(22), 4202-4218. https://doi.org/10.1002/SIM.2835
[65] Improving Sales Forecasting Accuracy: A Tensor Factorization Approach with Demand Awareness. (2022). *Informs Journal on Computing*, *34*(3), 1644-1660. https://doi.org/10.1287/ijoc.2021.1147
[66] Jewel, R., Linkon, A., Shaima, M., Badruddowza, Sarker, S., Shahid, R., Nabi, N., Rana, M. N. U., Shahriyar, M. A., Hasan, M., & Hossain, M. J. (2024). Comparative Analysis of Machine Learning Models for Accurate Retail Sales Demand Forecasting. *Journal of Computer Science and Technology Studies*, *6 (1)*, 204-210. https://doi.org/10.32996/jcsts.2024.6.1.23

[67] Jin, H., Chollet, F., Song, Q., & Hu, X. (2023). AutoKeras: An AutoML Library for Deep Learning. *Journal of Machine Learning Research*, *24*(6), 1-6. http://jmlr.org/papers/v24/20-1355.html

[68] John, H., & Taghi, M. K. (2020). CatBoost for big data: an interdisciplinary review. *Journal of Big Data*, *7*(1), 1-45. https://doi.org/10.1186/S40537-020-00369-8

[69] Johnston, F. R., Boyland, J. E., Meadows, M., & Shale, E. (1999). Some properties of a simple moving average when applied to forecasting a time series. *Journal of the Operational Research Society*, *50*(12), 1267-1271. https://doi.org/10.1057/palgrave.jors.2600823

[70] Josef, E., Florian, H., Viola, S., Claudia Guadarrama, S., Kelly, L.-T., Klaus, M., Thomas, H., & Domink, G. G. (2024). Forecasting seasonally fluctuating sales of perishable products in the horticultural industry. *Expert systems with applications*. https://doi.org/10.1016/j.eswa.2024.123438

[71] Joshua, C. O. K., German, S., & Surya, K. (2021). Automated Machine Learning for High-Throughput Image-Based Plant Phenotyping. *Remote Sensing*, *13*(5), 858. https://doi.org/10.3390/RS13050858

[72] Kabbilawsh, P., Deekonda Sathish, K., & Chithra, N. R. (2022). Performance evaluation of univariate time-series techniques for forecasting monthly rainfall data. *Journal of Water and Climate Change*. https://doi.org/10.2166/wcc.2022.107

[73] Kamal, S., & Kampan, M. (2021). Forecasting of intermittent demands under the risk of inventory obsolescence. *Journal of Forecasting*, *40*(6), 1054-1069. https://doi.org/10.1002/FOR.2761

[74] Karthika, G. (2023). Leveraging PyCaret for Time Series Analysis-A Low Code Approach. *Design of Single Chip Microcomputer Control System for Stepping Motor*, 1-4. https://doi.org/10.47363/jaicc/2023(2)314

[75] Kayim, F., & Yilmaz, A. (2022). Time Series Forecasting With Volatility Activation Function. *IEEE Access*, *10*, 104000-104010. https://doi.org/10.1109/access.2022.3211312

[76] Kerem Sinan, T., & Mustafa Gokce, B. (2018). Autoregressive forests for multivariate time series modeling. *Pattern Recognition*, *73*, 202-215. https://doi.org/10.1016/J.PATCOG.2017.08.016

[77] Kontopoulou, V. I., Panagopoulos, A. D., Kakkos, I., & Matsopoulos, G. K. (2023). A Review of ARIMA vs. Machine Learning Approaches for Time Series Forecasting in Data Driven Networks. *Future Internet*, *15*(8), 255. https://doi.org/10.3390/fi15080255

[78] Krishnan, B., Antonio, G., Shakoor, H., Liam, S., & Ben, A. (2013). Time series regression studies in environmental epidemiology. *International Journal of Epidemiology*, *42*(4), 1187-1195. https://doi.org/10.1093/IJE/DYT092

[79] Kulkarni, A. R., Shivananda, A., Kulkarni, A., & Krishnan, V. A. (2023). Statistical Univariate Modeling. In A. R. Kulkarni, A. Shivananda, A. Kulkarni, & V. A. Krishnan (Eds.), *Time Series Algorithms Recipes: Implement Machine Learning and Deep Learning Techniques with Python* (pp. 33-66). Apress. https://doi.org/10.1007/978-1-4842-8978-5_2

[80] Kunst, R. M., & Franses, P. H. (1998). The impact of seasonal constants on forecasting seasonally cointegrated time series. *Journal of Forecasting*, *17*(2), 109-124. https://doi.org/https://doi.org/10.1002/(SICI)1099-131X(199803)17:2<109::AID-FOR672>3.0.CO;2-U
[81] Lazzeri, F. (2020). *Machine Learning for Time Series Forecasting with Python*. Wiley. https://books.google.com.vn/books?id=5YUIEAAAQBAJ
[82] Le, T. T., Fu, W., & Moore, J. H. (2020). Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, *36*(1), 250-256.
[83] Lee, D. H., & Kyoungok, K. (2024). AdaBoost.RDT: AdaBoost integrated with Residual-based Decision Tree for Demand Prediction of Bike Sharing Systems under Extreme Demands. *IEEE Access*, 1-1. https://doi.org/10.1109/access.2024.3474017
[84] Li, T.-H., & Hinich, M. J. (2002). A Filter Bank Approach for Modeling and Forecasting Seasonal Patterns. *Technometrics*, *44*(1), 1-14. https://doi.org/10.1198/004017002753398182
[85] Li, W., & Law, K. L. E. (2024). Deep Learning Models for Time Series Forecasting: A Review. *IEEE Access*, *12*, 92306-92327. https://doi.org/10.1109/ACCESS.2024.3422528
[86] Lin, L., Fang, W., Xiaolong, X., & Shisheng, Z. (2017). Random forests-based extreme learning machine ensemble for multi-regime time series prediction. *Expert systems with applications*, *83*, 164-176. https://doi.org/10.1016/J.ESWA.2017.04.013
[87] Liu, X., & Wang, W. (2024). Deep Time Series Forecasting Models: A Comprehensive Survey. *Mathematics*, *12*(10), 1504. https://doi.org/10.3390/math12101504
[88] Liu, Z., Zhu, Z., Gao, J., & Xu, C. (2021). Forecast Methods for Time Series Data: A Survey. *IEEE Access*, *9*, 91896-91912. https://doi.org/10.1109/ACCESS.2021.3091162
[89] Lydia Lidong, S., & Robin John, H. (2005). Stochastic models underlying Croston's method for intermittent demand forecasting. *Journal of Forecasting*, *24*(6), 389-402. https://doi.org/10.1002/FOR.963
[90] Maximiliano, U., Eleni, V., & Jan, C. F. (2022). Exponential smoothing forecasts: taming the bullwhip effect when demand is seasonal. *International Journal of Production Research*, *61*(6), 1796-1813. https://doi.org/10.1080/00207543.2022.2048114
[91] Monnie, M., & Robert, A. Y. (2019). Comparison of Variable Selection Methods for Forecasting from Short Time Series.
[92] Nasrin, T., Narges Akhavan, F., Mehdi Jabbari, N., Ehsan, S., & Azadeh Jabbari, N. (2024). Using meta-learning to recommend an appropriate time-series forecasting model. *BMC Public Health*, *24*. https://doi.org/10.1186/s12889-023-17627-y
[93] Nuno, C., & Bonnie, K. R. (1996). Model selection and forecasting for long-range dependent processes. *Journal of Forecasting*, *15*(2), 107-125.

https://doi.org/10.1002/(SICI)1099-131X(199603)15:2<107::AID-FOR612>3.0.CO;2-D
[94] Odysseas, T., Vaia, M., & Athena, V. (2022). Design and development of a forecasting tool for the identification of new target markets by open time-series data and deep learning methods. *Applied Soft Computing*, *132*, 109843-109843. https://doi.org/10.1016/j.asoc.2022.109843
[95] Peixeiro, M. (2022). *Time Series Forecasting in Python*. Manning. https://books.google.com.vn/books?id=hqXczgEACAAJ
[96] Phillip, A. C. (1985). Forecasting time series:a comparative analysis of alternative classes of time series models. *Journal of Time Series Analysis*, *6*(4), 203-211. https://doi.org/10.1111/J.1467-9892.1985.TB00410.X
[97] Phillip, G. E., Joseph, A. M., Spivey, W. A., & William, J. W. (1982). Forecasting Applications of an Adaptive Multiple Exponential Smoothing Model. *Management Science*, *28*(9), 1035-1044. https://doi.org/10.1287/MNSC.28.9.1035
[98] Pierre, A. C. (1982). Prior Information and ARIMA Forecasting. *Journal of Forecasting*, *1*(4), 375-383. https://doi.org/10.1002/FOR.3980010405
[99] Prasshanth, C. V., & Sugumaran, V. (2024). Tire wear monitoring using feature fusion and CatBoost classifier. *Artificial Intelligence Review*, *57*(12). https://doi.org/10.1007/s10462-024-10999-6
[100] Priyesh, S., & Parida, S. K. (2024). A novel probabilistic gradient boosting model with multi-approach feature selection and iterative seasonal trend decomposition for short-term load forecasting. *Energy*. https://doi.org/10.1016/j.energy.2024.130975
[101] Proloy, D., & Behtash, B. (2023). Non-Asymptotic Guarantees for Reliable Identification of Granger Causality via the LASSO. *IEEE Transactions on Information Theory*, *69*(11), 7439-7460. https://doi.org/10.1109/tit.2023.3296336
[102] PyCaret — pycaret 3.0.4 documentation. https://pycaret.readthedocs.io/en/latest/index.html
[103] Qiang, S., Wen-Xin, Z., & Jianqing, F. (2020). Adaptive Huber Regression. *Journal of the American Statistical Association*, *115*(529), 254-265. https://doi.org/10.1080/01621459.2018.1543124
[104] Qiao, D., Xueqin, C., & Baoshan, H. (2024). Time series. In (pp. 181-196). https://doi.org/10.1016/b978-0-443-15928-2.00016-1
[105] Qingsong, W., Jingkun, G., Xiaomin, S., Liang, S., & Jian, T. (2019). RobustTrend: A Huber Loss with a Combined First and Second Order Difference Regularization for Time Series Trend Filtering.
[106] Rathipriya, R., Abdul Aziz Abdul, R., Dhamodharavadhani, S., Abdelrhman, M., & Yoganandan, G. (2022). Demand forecasting model for time-series pharmaceutical data using shallow and deep neural network model. *Neural Computing and Applications*, *35*(2), 1945-1957. https://doi.org/10.1007/s00521-022-07889-9
[107] Reich, N. G., Lessler, J., Sakrejda, K., Lauer, S. A., Iamsirithaworn, S., & Cummings, D. A. T. (2016). Case Study in Evaluating Time Series Prediction

Models Using the Relative Mean Absolute Error. *The American Statistician*, *70*(3), 285-292. https://doi.org/10.1080/00031305.2016.1148631

[108] Revin, I. E., Vladimir, P., Nikita, R. B., & Nikolay, O. N. (2023). Automated machine learning approach for time series classification pipelines using evolutionary optimization. *Knowledge Based Systems*, *268*, 110483-110483. https://doi.org/10.1016/j.knosys.2023.110483

[109] Ridge regression revisited: Debiasing, thresholding and bootstrap. (2022). *Annals of Statistics*, *50*(3). https://doi.org/10.1214/21-aos2156

[110] Rik van, L., & Ger, K. (2022). Anomaly detection in univariate time series incorporating active learning. *Journal of computational mathematics and data science*, *6*, 100072-100072. https://doi.org/10.1016/j.jcmds.2022.100072

[111] Ristanoski, G., Liu, W., & Bailey, J. (2013, 2013//). Time Series Forecasting Using Distribution Enhanced Linear Regression. Advances in Knowledge Discovery and Data Mining, Berlin, Heidelberg.

[112] Rob, J. H., Anne, B. K., Ord, J. K., & Ralph, D. S. (2005). Prediction intervals for exponential smoothing using two new classes of state space models. *Journal of Forecasting*, *24*(1), 17-37. https://doi.org/10.1002/FOR.938

[113] Robert, E. M., & Ruey, S. T. (1994). Bayesian analysis of autoregressive time series via the gibbs sampler. *Journal of Time Series Analysis*, *15*(2), 235-250. https://doi.org/10.1111/J.1467-9892.1994.TB00188.X

[114] Robert, F., Shaohui, M., Shaohui, M., & Stephan, K. (2019). Retail forecasting: Research and practice. *International Journal of Forecasting*. https://doi.org/10.1016/J.IJFORECAST.2019.06.004

[115] Ruud, H. T., & Laura, D. (2009). Forecasting intermittent demand: a comparative study. *Journal of the Operational Research Society*, *60*(3), 321-329. https://doi.org/10.1057/PALGRAVE.JORS.2602569

[116] Ryan, J. T. (2014). Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics*, *42*(1), 285-323. https://doi.org/10.1214/13-AOS1189

[117] Saha, P., Gudheniya, N., Mitra, R., Das, D., Narayana, S., & Tiwari, M. K. (2022). Demand Forecasting of a Multinational Retail Company using Deep Learning Frameworks. *IFAC-PapersOnLine*, *55*(10), 395-399. https://doi.org/https://doi.org/10.1016/j.ifacol.2022.09.425

[118] A Sales Forecasting Model for Coatings Industry via Econometric Models. (2023). *2*(1), 65-78. https://doi.org/10.59857/caot1017

[119] Saulo Martiello, M., Felipe Kenji, N., Celine, V., & André, C. P. L. F. d. C. (2022). Online Extra Trees Regressor. *IEEE transactions on neural networks and learning systems*, *PP*. https://doi.org/10.1109/TNNLS.2022.3212859

[120] Sebnem, G., Kubilay, A., & Fırat, H. (2024). PyCaret for Predicting Type 2 Diabetes: A Phenotype- and Gender-Based Approach with the "Nurses' Health Study" and the "Health Professionals' Follow-Up Study" Datasets. *Journal of Personalized Medicine*, *14*(8), 804-804. https://doi.org/10.3390/jpm14080804

[121] Shamsunder, S., Giannakis, G. B., & Friedlander, B. (1995). Estimating random amplitude polynomial phase signals: a cyclostationary approach. *IEEE*

*Transactions on Signal Processing*, *43*(2), 492-505. https://doi.org/10.1109/78.348131
[122] Shanhe, L., & Weixiong, R. (2020). Accurate Demand Forecasting for Retails with Deep Neural Networks.
[123] Shaolong, S., Yunjie, W., & Shouyang, W. (2018). AdaBoost-LSTM Ensemble Learning for Financial Time Series Forecasting. In (pp. 590-597). https://doi.org/10.1007/978-3-319-93713-7_55
[124] Shiqing, S., Fangfang, Q., & Pengcheng, N. (2021). Ensembles of Gradient Boosting Recurrent Neural Network for Time Series Data Prediction. *IEEE Access*, 1-1. https://doi.org/10.1109/ACCESS.2021.3082519
[125] Shuyang, J., Lianglong, D., Sichen, Z., Baoheng, L., & Xiaochuan, Z. (2024). A D* orthogonal matching pursuit algorithm for time-varying channel estimation. *Berkeley Program in Law & Economics*, *156*(5), 3158-3168. https://doi.org/10.1121/10.0034367
[126] Snezhana Georgieva, G.-I., Desislava Stoyanova, V., Stoimenova, M., Ivanov, A., & Iliycho Petkov, I. (2019). Regression trees modeling of time series for air pollution analysis and forecasting. *Neural Computing and Applications*, *31*(12), 9023-9039. https://doi.org/10.1007/S00521-019-04432-1
[127] Song, W., & Yang, Y. (2021). M-GAN-XGBOOST model for sales prediction and precision marketing strategy making of each product in online stores. *Drug Testing and Analysis*, *55*(5), 749-770. https://doi.org/10.1108/DTA-11-2020-0286
[128] Statistical forecasting—regression and time series analysis. (2022). In (pp. 709-722). https://doi.org/10.1016/b978-0-323-95112-8.00021-0
[129] Svetunkov, I., & Petropoulos, F. (2018). Old dog, new tricks: a modelling view of simple moving averages. *International Journal of Production Research*, *56*(18), 6034-6047. https://doi.org/10.1080/00207543.2017.1380326
[130] Terence, C. M. (2019). *Applied Time Series Analysis: A Practical Guide to Modeling and Forecasting*.
[131] Thais de Castro, M., Xue-Ming, Y., & Ek Peng, C. (2024). Hybrid convolutional long short-term memory models for sales forecasting in retail. *Journal of Forecasting*. https://doi.org/10.1002/for.3073
[132] Thomakos, D., & Nikolopoulos, K. (2012). Fathoming the theta method for a unit root process. *IMA Journal of Management Mathematics*, *25*(1), 105-124. https://doi.org/10.1093/imaman/dps030
[133] Thomakos, D. D., & Nikolopoulos, K. (2015). Forecasting Multivariate Time Series with the Theta Method. *Journal of Forecasting*, *34*(3), 220-229. https://doi.org/https://doi.org/10.1002/for.2334
[134] Tim, P. B., & Klaus, H. (1979). Techniques of linear prediction, with application to oceanic and atmospheric fields in the tropical Pacific. *Reviews of Geophysics*, *17*(5), 949-968. https://doi.org/10.1029/RG017I005P00949
[135] Tin, T., Shin, A., Hikari, S., Tomohito, M., & Akira, K. (2022). Incorporating Light Gradient Boosting Machine to land use regression model for estimating NO2 and

PM2.5 levels in Kansai region, Japan. *Environmental Modelling and Software*, *155*, 105447-105447. https://doi.org/10.1016/j.envsoft.2022.105447

[136] Tunyang, G., Tianzhen, J., Bingnan, L., Bin, A., & Haohai, S. (2023). Prediction of the Tropospheric NO2 Column Concentration and Distribution Using the Time Sequence-Based versus Influencing Factor-Based Random Forest Regression Model. *Sustainability*, *15*(3), 2748-2748. https://doi.org/10.3390/su15032748

[137] Umesh Kumar, S., Kannagi, A., & Meenu, S. (2024). Understanding Time Series Analysis for Improved Forecasting Techniques.

[138] Vaiciukynas, E., Danenas, P., Kontrimas, V., & Butleris, R. (2021). Two-Step Meta-Learning for Time-Series Forecasting Ensemble. *IEEE Access*, *9*, 62687-62696. https://doi.org/10.1109/ACCESS.2021.3074891

[139] Vaishali, J., & Shiv Kumar, T. (2024). Overview: machine learning. In (pp. 130-144). https://doi.org/10.58532/nbennurch183

[140] Vandeput, N. (2023). *Demand Forecasting Best Practices*. Manning. https://books.google.com.vn/books?id=C_u8EAAAQBAJ

[141] Wanqing, Z., Thomas, B., & Yacine, R. (2017). Efficient least angle regression for identification of linear-in-the-parameters models. *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences*, *473*(2198), 20160775-20160775. https://doi.org/10.1098/RSPA.2016.0775

[142] Wayne, F. V., & Brett, A. P. (1998). Time Series Analysis in Historiometry: A Comment on Simonton. *Journal of Personality*, *66*(3), 477-486. https://doi.org/10.1111/1467-6494.00020

[143] Wei-Yin, L., & Wei, Z. (2013). Regression trees for longitudinal and multiresponse data. *The Annals of Applied Statistics*, *7*(1), 495-522. https://doi.org/10.1214/12-AOAS596

[144] Xiubin, Z., Xing-fu, H., Lan, Y., Pedrycz, W., & Zhiwu, L. (2024). A Development of Fuzzy-Rule-Based Regression Models Through Using Decision Trees. *IEEE Transactions on Fuzzy Systems*, *32*, 2976-2986. https://doi.org/10.1109/tfuzz.2024.3365572

[145] Yan, Q., Chenliang, L., Han, D., Min, C., Yunwei, Q., & Yuming, D. (2019). A Deep Neural Framework for Sales Forecasting in E-Commerce.

[146] Yanping, S., Yi-Yang, N., & Kunde, Y. (2023). A fast threshold OMP based on self-learning dictionary for propeller signal reconstruction. *Ocean Engineering*. https://doi.org/10.1016/j.oceaneng.2023.115792

[147] Yi, D., Bu, S., & Kim, I. (2019). An Enhanced Algorithm of RNN Using Trend in Time-Series. *Symmetry*, *11*(7), 912. https://doi.org/10.3390/sym11070912

[148] Yingjie, W., Xianrui, Z., Fengxiang, H., Hong, C., & Dacheng, T. Huber Additive Models for Non-stationary Time Series Analysis.

[149] Yongjun, K., Yung-Cheol, B., & Sang-Joon, L. (2024). A Study on Sugar Content Improvement and Distribution Flow Response through Citrus Sugar Content Prediction Based on the PyCaret Library. *Horticulturae*, *10*(6), 630-630. https://doi.org/10.3390/horticulturae10060630

[150] Yongqing, L., Jun, W., Hui, L., & Peng, R. (2020). Peak Colocalized Orthogonal Matching Pursuit for Seismic Trace Decomposition. *IEEE Access*, *8*, 8620-8626. https://doi.org/10.1109/ACCESS.2020.2964095

[151] Yun, F., Zhiwen, Y., Kaixiang, Y., & Chen, C. L. P. (2024). AdaBoost-Stacking Based on Incremental Broad Learning System. *IEEE Transactions on Knowledge and Data Engineering*, 1-14. https://doi.org/10.1109/tkde.2024.3433587

[152] Zhenyu, L., Zhengtong, Z., Jing, G., & Cheng, X. (2021). Forecast Methods for Time Series Data: A Survey. *IEEE Access*, *9*, 91896-91912. https://doi.org/10.1109/ACCESS.2021.3091162

[153] Zhiyuan, Z., Alexander, R., & Prakash, B. A. (2023). Performative Time-Series Forecasting. *arXiv.org*, *abs/2310.06077*. https://doi.org/10.48550/arxiv.2310.06077

[154] Zitian, W., & Shaohua, T. (2009). Identifying idiosyncratic stock return indicators from large financial factor set via least angle regression. *Expert systems with applications*, *36*(4), 8350-8355. https://doi.org/10.1016/J.ESWA.2008.10.018

[155] Žunić, E. (2019). *Real-world sales forecasting benchmark data* (4TU.Centre for Research Data. https://doi.org/10.4121/UUID:B9F3DF9E-08BC-4331-96DA-7CABFA8970C0

[156] Zuqiang, Q., & Nalini, R. (1998). Bayesian Inference for Time Series with Stable Innovations. *Journal of Time Series Analysis*, *19*(2), 235-249. https://doi.org/10.1111/1467-9892.00088

# APPENDIX

## Appendix A: Data Preprocessing Code

```
# Import necessary libraries
import pandas as pd
import numpy as np

# Load dataset from download resources
data = pd.read_csv('real_world_sales_data.csv')

# Display the first few rows of the raw data
print(data.head())

# Display basic information about the dataset
data.info()

# Descriptive statistics for selected columns
print("Quantity and Unit Price Statistics:")
print(data[['quantity', 'unit_price']].describe())

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

# Clean data: Filter invalid values
sales_data = data[(data['quantity'] >= 0) & (data['unit_price'] > 0)].copy()

# Create a new column for total sales
sales_data['total_sales'] = sales_data['quantity'] * sales_data['unit_price']

# Display the cleaned dataset
print(sales_data.head())
```

## Appendix B: Data Aggregation and Transformation

```
# Group data by date and aggregate total sales
df = sales_data.groupby('date')['total_sales'].sum().reset_index()

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

# Display the first few rows of the aggregated data
print(df.head())
```

## Appendix C: Import PyCaret library and model setup

```
# Import PyCaret's time series library
from pycaret.time_series import *

# Convert index to datetime
df.index = pd.to_datetime(df.index)
```

```
# Set the daily frequency
df = df.asfreq('D')

# Missing dates and fill them
df = df.resample('D').interpolate()

# PyCaret's time series setup
exp_name = setup(
    data=df,
    target='total_sales',
    session_id=123,
    numeric_imputation_target='mean'
)

# Print all available models
models()

# Rank models but exclude model 'knn'
best_model = compare_models(exclude='knn')
```

## Appendix D: Plot models

```
# Get the model comparison results into a DataFrame
results = pull()  # This gets the results from PyCaret's compare_models

# Create a figure with subplots for each metric
fig, axes = plt.subplots(7, 1, figsize=(15, 35))  # Increased height to
accommodate more subplots
fig.suptitle('Model Performance Metrics Comparison', fontsize=16, y=0.95)

# Define all metrics to plot
metrics = ['MASE', 'RMSSE', 'MAE', 'RMSE', 'MAPE', 'SMAPE', 'TT (Sec)']
colors = ['skyblue'] * len(results)

# Plot each metric
for i, metric in enumerate(metrics):
    # Sort values for the current metric
    sorted_data = results.sort_values(by=metric, ascending=True)

    # Create horizontal bar plot
    axes[i].barh(range(len(sorted_data)), sorted_data[metric], color=colors)

    # Add model names as y-tick labels
    axes[i].set_yticks(range(len(sorted_data)))
    axes[i].set_yticklabels(sorted_data['Model'], fontsize=8)

    # Add value labels on the bars
    for j, v in enumerate(sorted_data[metric]):
        # Format numbers based on metric type
        if metric == 'TT (Sec)':
            value_text = f'v:.2f'
        elif metric in ['MAPE', 'SMAPE']:
            value_text = f'v:.2%'  # Format as percentage
```

```
        else:
            value_text = f'v:.4f'

        axes[i].text(v, j, value_text, va='center', fontsize=8)

    # Set title and adjust layout
    axes[i].set_title(f'metric Comparison', pad=20)
    axes[i].grid(True, axis='x', linestyle='--', alpha=0.7)

    # Format x-axis for percentage metrics
    if metric in ['MAPE', 'SMAPE']:
        axes[i].xaxis.set_major_formatter(plt.FuncFormatter(lambda x, p:
f'x:.1%'))

# Adjust layout to prevent overlap
plt.tight_layout()
plt.show()
```