Otto Kolehmainen

Demand Forecasting in Bicycle Retail

A Constructive Study on Time Series Forecasting

School of Technology and Innovations
Master of Science in Economics and Business Administration
Information Systems

Vaasa 2026

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Otto Kolehmainen
Title of the thesis: Demand Forecasting in Bicycle Retail
Degree: Economics and Business Administration
Degree Programme: Information Systems
Supervisor: Jouni Lampinen
Year: 2026 **Pages:** 73

ABSTRACT:

This thesis examines demand forecasting in bicycle retail and develops a forecasting model for the case company's weekly unit-level bicycle sales. The objective is to improve the accuracy and reliability of demand predictions in a retail context where seasonality, trend shifts, promotional effects, and other external factors directly influence inventory management, replenishment planning, and financial performance. The study is delimited to aggregated historical bicycle sales and to a univariate forecasting setting.

The forecasting artefact combines Prophet and XGBoost. Prophet is a decomposable time series model designed to represent trend and seasonal structures, while XGBoost is a gradient-boosted decision tree algorithm capable of learning nonlinear relationships from engineered features. The empirical process follows a constructive, design-oriented logic: the model is developed iteratively, evaluated against baseline forecasts, and assessed for practical applicability. The analysis uses historical unit-level sales data from the case company's online sales and three physical retail locations. The performance of the predictive model is evaluated using mean absolute error and root mean square error.

The results show that the hybrid approach outperforms the standalone baseline models in the primary case and in an additional validation using a synthetic time series from Kaggle. The developed artefact provides a more robust and data-driven forecasting framework for supporting inventory management, operational planning, and strategic decision making. The findings indicate that combining statistical time series modeling with machine learning can be suitable for retail demand forecasting when the model configuration, data limitations, and deployment requirements are carefully considered.

The main contribution of the thesis is the developed and evaluated hybrid forecasting artefact, together with evidence on its practical suitability and limitations in a small retail forecasting context.

KEYWORDS: Hybrid modelling, machine learning, predictive analytics, time series forecasting, demand forecasting, Prophet, XGBoost

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen yksikkö
Tekijä: Otto Kolehmainen
Tutkielman nimi: Demand Forecasting in Bicycle Retail
Tutkinto: Kauppatieteiden maisteri
Koulutusohjelma: Tietojärjestelmätiede
Työn ohjaaja: Jouni Lampinen
Valmistumisvuosi: 2026 **Sivumäärä:** 73

TIIVISTELMÄ:

Tämä tutkielma tarkastelee kysynnän ennustamista polkupyörien jälleenmyyntikontekstissa ja kehittää konstruktiivisella tutkimusotteella ennustemallin kohdeyrityksen kappalemääräisen polkupyörämyynnin ennustamiseen viikkotasolla. Tutkimuksen tavoitteena on parantaa kysyntäennusteiden tarkkuutta ja luotettavuutta ympäristössä, jossa kausivaihtelu, trendimuutokset, markkinointivaikutukset ja muut ulkoiset tekijät vaikuttavat varastonhallintaan, ostojen suunnitteluun ja taloudelliseen suorituskykyyn. Tutkimus rajautuu historialliseen, aggregoituun polkupyörämyyntiin ja univariaattiseen ennustamisasetelmaan.

Tutkielmassa kehitettävä artefakti yhdistää Prophetin ja XGBoostin. Prophet on hajotettava aikasarjamalli, jonka avulla voidaan kuvata trendi- ja kausirakenteita, kun taas XGBoost on gradienttitehostettuihin päätöspuihin perustuva koneoppimismalli, joka oppii epälineaarisia suhteita muodostetuista piirteistä. Tutkimusprosessi noudattaa konstruktiivista ja suunnittelutieteellistä logiikkaa: mallia kehitetään iteratiivisesti, arvioidaan vertailumalleja vasten ja tarkastellaan käytännön soveltuvuuden näkökulmasta. Empiirinen analyysi perustuu kohdeyrityksen verkkokaupan ja kolmen kivijalkamyymälän historialliseen kappalemääräiseen myyntiaineistoon.

Tutkielman tulokset osoittavat, että hybridimalli suoriutuu vertailumalleja paremmin sekä tutkielman päätapauksessa että lisävalidoinnissa, jossa käytetään Kagglesta peräisin olevaa synteettistä aikasarjaa. Kehitetty artefakti tarjoaa datalähtöisen ennustamisen viitekehyksen varastonhallinnan, toiminnan suunnittelun ja strategisen päätöksenteon tueksi. Tulosten perusteella tilastollisen aikasarjamallinnuksen ja koneoppimisen yhdistäminen soveltuu jälleenmyynnin kysynnän ennustamiseen, kun mallin konfiguraatio, aineiston rajoitteet ja käyttöönoton edellytykset huomioidaan kriittisesti.

Tutkielman keskeinen kontribuutio on kehitetty ja arvioitu hybridiennustamisen artefakti sekä sen käytännön soveltuvuutta ja rajoitteita koskeva arviointi pienemmän jälleenmyyntiyrityksen kontekstissa.

AVAINSANAT: Hybridimallinnus, koneoppiminen, ennakoiva analytiikka, aikasarjaennustaminen, kysynnän ennustaminen, Prophet, XGBoost

Contents

1 Introduction 9
1.1 Background and Motivation 9
1.2 Research Problem and Research Questions 10
1.3 Scope 11
1.4 Structure of the Thesis 12
2 Forecasting in Retail Decision Making 13
2.1 Statistical vs Machine Learning Based Forecasting Methods 14
2.2 Hybrid Models in Time Series Forecasting 15
3 Addressed Forecasting Methods 18
3.1 Prophet 18
3.2 XGBoost 23
3.3 Hybrid Model 27
4 Research Methodology 28
4.1 Constructive Research Approach 28
4.2 Construct Development Process 30
4.3 Evaluation Metrics and Criteria for the Construct 32
5 Empirical Study: Case Company 34
5.1 Data Preprocessing 34
5.2 Exploratory Data Analysis 35
5.3 Framework Implementation 39
5.3.1 Used Tooling 40
5.3.2 Initialisation 40
5.3.3 Baseline Forecasts 41
5.3.4 Prophet Model 42
5.3.5 XGBoost Models (Baseline & Hybrid) 43
5.3.6 Evaluation of Prediction Accuracy 45
5.3.7 Forecasting the Future 49
5.3.8 Applicability of the Construct in a Business Environment 51

6 Validation of the Construct Beyond the Primary Case 52
6.1 Prophet 54
6.2 XGBoost Models (Baseline & Hybrid) 55
6.3 Evaluation of Prediction Accuracy 56
6.4 Forecasting the Future 60
7 Discussion 62
7.1 Answers to Research Questions 62
7.2 Contributions 64
7.3 Limitations of the Study and Directions for Future Research 65
References 68
Appendices 73
Appendix 1. Source code of the proposed forecasting pipeline 73
Appendix 2. Feature Importance: Hybrid XGBoost (primary case) 73
Appendix 3. Feature Importance: Hybrid XGBoost (Kaggle) 73

Figures

Figure 1. Google Trends search statistics for 'cyclocross' search term 20
Figure 2. Trend and Seasonality components for 'cyclocross' Google Trends search term time series 21
Figure 3. Default Prophet model fit on ‘cycloross’ Google Trends search term time series 22
Figure 4. Results of tuning Prophet's yearly seasonality hyperparameter on ‘cyclocross’ search term time series from Google Trends 23
Figure 5. Architecture of a gradient boosted tree ensemble, adapted from Xu et al. (2023, p. 3) 24
Figure 6. DSRM Process Model (Peffers et al., 2007, p. 54) 30
Figure 7. Line plot of the three year historical unit-level bicycle sales 36
Figure 8. Year-over-year comparison of the three years of historical bicycle sales 37
Figure 9. Seasonal Decomposition of Time Series by Loess (STL) applied to the three years of historical bicycle sales 38
Figure 10. A High-Level Overview of the Proposed Hybrid Forecasting Framework 39
Figure 11. Example of expanding window cross-validation train-test splitting 41
Figure 12. Train-Test split of the historical three years of bicycle sales data 46
Figure 13. Results of the different forecasting models plotted against each other and the actual historical bicycle sales in the test set 47
Figure 14. Results of the best performing three forecasting models plotted against each other and the actual historical bicycle sales in the test set 49
Figure 15. Using the whole time series for training data when forecasting into the future 50
Figure 16. Forecasting a year into the future by training the models with three years of historical bicycle sales 51
Figure 17. Lineplot of the Kaggle Time Series Practice Dataset after applying the filtering query 53
Figure 18. Model predictions plotted against the test set over the three cross-validation folds 54

Figure 19. Comparing the development of mean absolute error between models over the completed three rounds of cross validation 59
Figure 20. Comparing the best performing models against each other and the actual values of the time series on the last completed cross-validation fold 60
Figure 21. Forecasting a year into the future after training the models with the whole time series from Kaggle 61

Tables

Table 1. Prophet Hyperparameters Selected for Tuning 19
Table 2. XGBoost Hyperparameters Selected for Tuning 26
Table 3. Results of Prophet Hyperparameter Tuning (primary case) 43
Table 4. Results of XGBoost Feature Selection (primary case) 44
Table 5. Results of XGBoost Hyperparameter Tuning (primary case) 45
Table 6. Validation Results for Each Forecasting Model (primary case) 47
Table 7. Results of Prophet Hyperparameter Tuning (Kaggle) 54
Table 8. Results of XGBoost Feature Selection (Kaggle) 55
Table 9. Results of XGBoost Hyperparameter Tuning (Kaggle) 56
Table 10. Validation Results for Each Forecasting Model (Kaggle) 57
Table 11. Hybrid XGBoost's Improvement Margin Over The Baselines (Kaggle) 58

Equations

Equation 1. Prophet 18
Equation 2. Prophet without holiday effects 19
Equation 3. Mean Absolute Error (MAE) 32
Equation 4. Root Mean Square Error (RMSE) 32
Equation 5. Improvement (%) 33
Equation 6. Mean Bias Error (MBE) 33
Equation 7. Multiplicative-Additive Scaling 35
Equation 8. Seasonal-Trend decomposition based on Loess (STL) 37

Equation 9. Determining the Number of Cross-Validation Folds 40

Equation 10. Seasonal naïve 42

Equation 11. Single-season average 42

Equation 12. Filtering Query for Kaggle's Time Series 52

1 Introduction

Demand forecasting can be a powerful tool in retail management, supporting the retailer's decision making on strategic, tactical and operational levels (Fildes et al., 2022, p. 1284). However, inaccurate predictions can cause stockouts, lost sales or overstocking, all of which can end up in costly outcomes for the retail organisation (Kourentzes et al., 2020, p. 1). For Fillarikauppa Cycli, a bicycle retailer operating in the Finnish domestic market through three physical stores and an online store, accurate prediction of sales volumes is important due to strong seasonal fluctuations. For Cycli, ineffective forecasting could lead to excess inventory or lost sales opportunities, negatively impacting profitability and customer satisfaction.

1.1 Background and Motivation

Traditional statistical models, including ARIMA and exponential smoothing are widely used for time series forecasting due to their simplicity and interpretable nature (Fatima & Rahimi, 2024, p. 1). These models are effective at capturing linear trends and seasonal patterns but often struggle with nonlinear patterns and dynamic dependencies inherent in retail sales data (Houssein et al., 2025, p. 1). In contrast, machine learning models such as Extreme Gradient Boosting (XGBoost) have shown strong performance in modeling nonlinear patterns and interactions but aren't built to capture temporal dependencies by design (Wang & Zain, 2025, p. 4056).

To overcome these limitations, hybrid forecasting approaches have gained popularity in both academic research and in industry settings. By combining statistical time series models with machine learning methods, hybrid frameworks aim to leverage the strengths of each approach. Statistical models provide interpretability and trend/seasonality decomposition, while machine learning models capture complex nonlinear relations (Kim et al., 2025, p. 76). In this thesis, Prophet refers to the decomposable time series model introduced by Taylor and Letham (2018), and XGBoost refers to the scalable gradient-boosted tree algorithm introduced by Chen and Guestrin

(2016). Combination of Prophet and XGBoost is an example of such a hybrid approach, where Prophet models the global time series structure and XGBoost learns additional nonlinear residual patterns.

Hybrid forecasting approaches have been proven to outperform the component models that they consist of in multiple domains (Sina et al., 2023, p. 11). However, their application in SME bicycle retail is sparsely documented. This thesis addresses this knowledge gap by developing a hybrid forecasting framework tailored for Fillarikauppa Cycli. The study integrates Prophet and XGBoost to model systematic trend and seasonal effects and to capture nonlinear relationships. The hybrid model is trained and evaluated on historical unit-level bicycle sales data and it's performance is compared against the standalone Prophet, XGBoost and two naïve baseline models to assess it's predictive performance.

1.2 Research Problem and Research Questions

Some components in bicycle sales data is best captured by a statistical approach, while other characterics can benefit from applying a machine learning model. For a bicycle retailer like Cycli, relying completely on either traditional statistical or machine learning models alone could lead to suboptimal predictions, potentially resulting in overstocking, stockouts, and lost revenue. There is potential in a hybrid modeling approach that could capture both systematic time series patterns and nonlinear residual effects.

Thus, the primary research question of this study is presented as: "How does a hybrid Prophet/XGBoost model perform when predicting bicycle sales compared to standalone Prophet, XGBoost and other baseline forecasts?"

Complementing the primary research question, the study addresses the following secondary research questions:

- Does aggregating unit-level bicycle sales data from case company's four sales channels over a three year window provide a sufficient data set for hybrid time-

series modeling?
- Which features carry the most weight in the hybrid model's predictions?

1.3 Scope

This study focuses on developing and evaluating a hybrid Prophet/XGBoost model for forecasting bicycle sales at Cycli, a bicycle retailer operating three physical stores and an online store in the Finnish domestic market. The research is limited to modeling unit-level sales data collected from these four sales channels over a three year historical period. The analysis emphasizes capturing trend and seasonal patterns with a statistical approach, while incorporating machine learning to model residual nonlinearities that the statistical time series method may not capture.

The explored modeling methods of the study are limited to Prophet and XGBoost. Other forecasting methods, such as neural networks, statistical approaches beyond Prophet, or alternative ensemble approaches are not considered in this study.

Time series forecasting tasks can be divided to either univariate or multivariate tasks depending on whether the modelling is done purely on past observations of the data (univariate), or if additional time series are used as explanatory variables for the target variable (multivariate) (Cerqueira et al., 2019). This study focuses on univariate modelling with engineered features of bicycle sales. The proposed construct forecasts a single target variable and exogenous features such as economic indicators, competitor activity, or weather information are not included in the modeling.

The scope is further defined by the use of a single aggregated target variable and engineered features derived from historical sales values. The broader limitations and implications of this scope are discussed in the final discussion chapter.

1.4 Structure of the Thesis

This thesis consists of seven main chapters; Introduction, Forecasting in Retail Decision Making, Addressed Forecasting Methods, Research Methodology, Empirical Study: Case Company, Validation of the Construct Beyond the Primary Case, and Discussion. The Introduction chapter provides background and motivation for the approach, and sets the research problem, questions, and scope of the study. Forecasting in Retail Decision Making emphasises the importance of data-driven decision making in modern retail business, and introduces statistical, machine learning, and hybrid forecasting methods. Addressed Forecasting Methods dives deeper into Prophet and XGBoost, describing their individual purposes and integration strategy in the hybrid model. Research Methodology justifies the Design Science/Constructive research approach of the thesis, and the construct's development process and evaluation criteria. The empirical and validation chapters present the data, implementation, results, and assessment of generalisability. The final Discussion chapter interprets the results, contributions, limitations, and directions for future work.

2 Forecasting in Retail Decision Making

Research shows that decision makers often base their decisions on narrow frames of reference or an understanding that is tied tightly in phenomena experienced in the past. This can lead to decisions that don't put enough emphasis on changes that could result both in opportunities and threats for the organisation within which the decisions are made (Wright & Goodwin, 2009, p. 814).

Research dating back 70 years indicate that people tend to prefer their own judgement in creating predictions over algorithms even when the prediction accuracy of the algorithms is proven superior. Building on this, recent studies have shown that people are significantly more tolerant to errors caused by human judgement than those made by algorithms (Harvey & De Baets, 2025, p. 532). However, in recent history there has been a paradigm shift in management practice and decision making due to the developments seen in big data, advanced analytics and artificial intelligence. These developments have turned data-driven decision making into an essential factor for organisational success (Sarioguz & Miser, 2024, p. 1642). Data driven decision making is a broad term, but can generally be described as the process of gathering, analysing and understanding data to draw conclusions and guide strategic and operational decisions (Sarioguz & Miser, 2024, p. 1642).

Retailers are required to make long-term strategic decisions in environments with dynamic competition and technological advancements. The dynamic nature of these factors result in strategic decision making in retail contexts typically relying on forecasts of the future (Fildes et al., 2022, p. 1284). On an operational level, retailers need to make accurate decisions to satisfy customer needs and avoid overstocking or stockouts. Accurate demand forecasts can be a crucial tool for retailers in planning and organizing purchasing, distribution, labor scheduling and post-sales services (Fildes et al., 2022, p. 1286).

2.1 Statistical vs Machine Learning Based Forecasting Methods

Traditional statistical forecasting methods such as ARIMA (Autoregressive Integrated Moving Average) and Exponential Smoothing (ES) are widely used in forecasting due to their simple model structures and easy interpretability (Fatima & Rahimi, 2024, p. 1). ARIMA models describe autocorrelations in the data while Exponential Smoothing approaches base their modelling on describing the trend and seasonality in the dataset (Hyndman et al., 2025).

Autocorrelation is a metric used in measuring the linear relationship between lagged values in a time series (Hyndman et al., 2025). When modeling a weekly time series, a lag of 4 would refer to the value of ***y*** four weeks ago, and lag of 52 to the value of ***y*** one year in the past. A forecasting model relying on autocorrelation could find short-term cyclic patterns by observing the correlation values of the 4 week lag and annual patterns from the correlations of the lag of 52. Exponential Smoothing is a forecasting method relying on the weighted averages of past observations, emphasizing recent observations over older ones. The mentioned exponentiality in the model's name refers to the observations receiving exponentially decaying weights as the observations get older (Hyndman et al., 2025).

Different machine learning approaches used in forecasting vary greatly in their model structure and ways of capturing patterns in time-series data. On the contrary to the linear trend capturing ability of classical approaches, machine learning models such as decision trees and random forests excel in identifying non-linear relationships in the dataset (Arora & Kolambe, 2024, p. 578). Gradient boosting algorithms, such as XGBoost used in the proposed forecasting framework of this study iteratively build predictive models by combining weak learners (Arora & Kolambe, 2024, p. 577).

Machine learning algorithms require time-dependent relationships in the data to be expressed through careful feature engineering (Houssein et al., 2025, p. 27). These features are then provided to the model as input features during the model training

process. Examples of time-dependent features include lags, moving averages, rolling statistics and calendar features (Karmaker, 2025, p. 4-9). Machine learning algorithms rely on these input features to learn time-dependent interactions and correlations in the data, as they are not able to capture temporal dynamics on their own.

Statistical approaches are fit to data via mathematical methods, often relying on assumptions of the data's structure, while machine learning models learn by iterative optimisation through trial and error. The architectural difference results in machine learning models often requiring larger datasets to converge and produce optimal results. Cerqueira et al. (2019) found increasing the sample size in time series forecasting problems to systematically improve the predictive performance of machine learning models when compared to traditional statistical approaches.

2.2 Hybrid Models in Time Series Forecasting

It is often hard to determine whether the patterns in a certain time series dataset is a result of a linear or non linear-processes. In an effort to determine this, choosing an appropriate model for a forecasting task in such a scenario often includes a process of trying and evaluating many different approaches (Kontopoulou et al., 2023, p. 22). However, many processes of the real world do not generate data that has purely linear or non-linear patterns. This motivates constructing forecasting approaches that integrate multiple models which are each able to capture linear or non-linear characteristics in the data (Kontopoulou et al., 2023, p. 22).

Liu's and Wen's (2025, p. 184-185) study on sales trend forecasting on e-commerce platforms notes that machine-, or deep learning approaches lack in interpretability and their ability to capture seasonal effects, while purely statistical forecasting methods struggle with high dimensions of covariates and strong effects caused by holiday periods or promotions. They describe Deep Learning architectures such as long short-term memory (LSTM), transformers and temporal convolutional networks as strong approaches for time series forecasting, but make a note of the associated data

requirements, lack of interpretability, computational costs, and demanding hyperparameter tuning process as obstacles for adoption in business contexts with resource constrains. They propose that the limitations that come with choosing either a statistical or a ML/DL approach can be addressed by combining the technologies using the trend and seasonality capturing ability of a traditional statistical approach (ARIMA/SARIMA, Exponential Smoothing, Prophet), and leveraging a gradient boosted tree ensemble (XGBoost, LightGBM) to learn nonlinear relationships and complex interdependencies between variables. According Liu and Wen (2025, p. 185) hybrid approaches such as this are implementable in real-world scenarios, while preserving interpretability and keeping the computational requirements manageable.

The M-competitions are well known and reputable competitions in the field of forecasting. The 4th edition of the competition was a continuation of it's three predecessors, ending in May of 2018 (Makridakis et al., 2018, p. 802). The M4 competition marked a paradigm shift in the field of forecasting, as it was the first iteration of the M-competitions to formally allow machine learning models in it's submissions (Makridakis et al., 2018, p. 802). The main objective of the M4 competition, like the previous competitions of the series is stated as: "learn how to improve the forecasting accuracy, and how such learning can be applied to advance the theory and practice of forecasting'' (Makridakis et al., 2018, p. 803). The competition was won by Slawek Smyl, who proposed a hybrid model combining both statistical and machine learning methods (Makridakis et al., 2018, p. 803), indicating the performance potential of hybrid approaches in forecasting tasks in 2018.

The M5 Accuracy competition followed the 4th edition of the forecasting competitions and was open for submissions from March to July in 2020 (Makridakis et al., 2022, p. 1347). The competition focused on retail forecasting with the objective of producing forecasts for Walmart's 42,840 hierarchical time series (Makridakis et al., 2022, p. 1346). This time the performance edge of machine learning approaches was clear: all top submissions were heavily based on machine learning, beating the statistical benchmarks

(Makridakis et al., 2022, p. 1356). The value of combining simple models was illustrated again with the top-performing submission incorporating six models in its predictions, and the runner-up combining the predictions of five models in it's trend estimations (Makridakis et al., 2022, p. 1356).

The latest M-competition, M6 was held from February to February in 2022-2023 and focused on financial forecasting of 100 publicly traded assets requiring participants to provide forecasts and investment strategies for each asset every four weeks (Makridakis et al., 2025, p. 9). One of the top objectives of the competition was to find out whether accurate forecasts can be translated into highly performing investements decisions (Makridakis et al., 2025, p. 50-51). This question was not explicitly answered with top teams in the forecasting task having on average mediocre investment decision performance and the high performing teams in investement decisions having varying levels of accuracy in their forecasts (Makridakis et al., 2025, p. 51). Still, the results of the competition show that combining multiple models in forecasting tasks yield superior results when comparing to forecasts made by a single model (Makridakis et al., 2025, p. 51-52), further supporting the potential of hybrid forecasting approaches.

3 Addressed Forecasting Methods

The proposed forecasting framework of this study combines two modelling techniques: Prophet, a nonlinear regression model which works best with time series data with strong seasonality and historical data containing multiple seasons (Hyndman et al., 2025), and XGBoost, a supervised machine learning model based on gradient-boosted decision tree ensembles (DMLC, n.d.).

3.1 Prophet

Prophet is a decomposable model, meaning that it breaks the time series data down to three main model components: trend, seasonality and holidays (Taylor & Letham, 2018, p. 38).

The Prophet model can be expressed with the following equation:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$ **Equation 1.** Prophet

Where ***g(t)*** describes the growth term (trend), ***s(t)*** describes the seasonality (eg. yearly/weekly), ***h(t)*** represents holiday effects, and ***εt*** is a term user to represent unexplained error or white noise in the observations (Hyndman et al., 2025).

Publications on forecasting often make a distinction between statistical and judgemental forecasting. Statistical forecasting is described as models that are fit on historical data, while judgemental forecasts (also referred to as managerial forecasts) are produced by human experts by methods they are confident to work well on their time series (Taylor & Letham, 2018, p. 42). Sanders (2005, as cited in Taylor & Letham, 2018, p. 43) argues both approaches to have their advantages. He suggests that producing statistical forecasts require less domain knowledge and required effort, while judgemental forecasts can accomodate more knowledge and respond better to changing environments while having the tradeoff of requiring intensive efforts from analysts.

Prophet is a forecasting framework aiming to gain benefits from both aforementioned modelling methodologies. It pursues its goal by providing easily adjustable model hyperparameters accompanied by visual tools to support the evaluation of the effects caused by the made hyperparameter changes (Taylor & Letham, 2018, p. 42-43).

The source code of Prophet defines 18 distinct hyperparameters that can be specified when initialising the model (Taylor & Letham, 2026). The values specified for each hyperparameter affect the model's performance by controlling trend, seasonality, holiday effects, changepoints, uncertainty estimation and data scaling. When a value for a hyperparameter is left unspecified, the model uses it's predefined default value (Taylor & Letham, 2026). The hybrid forecasting framework later introduced in this paper relies on the default value for all but five of the 18 hyperparameters. The selected hyperparameters to be optimised by the proposed construct are: *seasonality_mode*, *growth*, *yearly_seasonality*, *changepoints_prior_scale,* and *seasonality_prior_scale*.

Table 1. Prophet Hyperparameters Selected for Tuning

Hyperparameter	**Effect on the model**
seasonality_mode	Determines additive or multiplicative seasonality
growth	Sets the trend type (linear, logistic, flat)
yearly_seasonality	Enables or configures yearly patterns
changepoint_prior_scale	Controls trend flexibility at changepoints
seasonality_prior_scale	Adjusts the strength of seasonal effects

Selecting only these hyperparameters for tuning turns Prophet's previously introduced model structure into a simpler equation without the holiday effects ***h(t)***:

$$y(t) = g(t) + s(t) + \epsilon_t$$

Equation 2. Prophet without holiday effects

as this hyperparameter setup doesn’t specify any holidays or promotional periods.

An advantage that comes with using an additive model is inspecting each model component on their own (Taylor & Letham, 2018, p. 42). To illustrate this, we have a time series from Google Trends describing the global popularity of the search term “cyclocross”. Cyclocross is a cycling discipline in which competitions are raced primarly during the winter months with the first events being usually held in September. The sport has the most events during late December and early January with regular professional races in Belgium and the Netherlands. By inspecting the raw time series, we can distinguish a strong yearly seasonality aligning with the professional racing calendar.

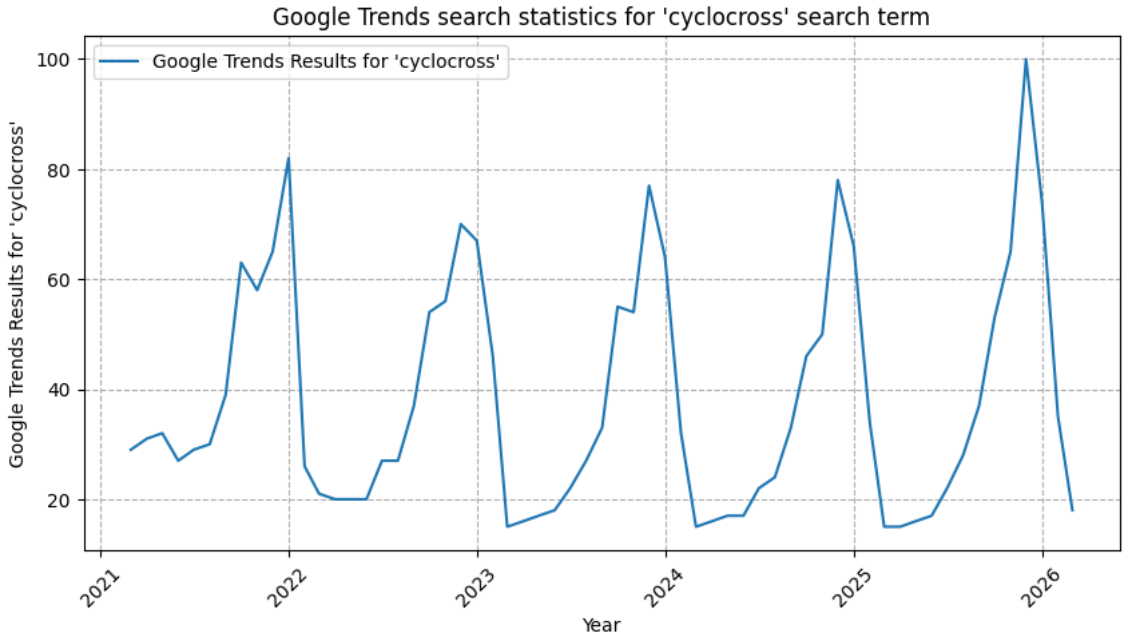


Figure 1. Google Trends search statistics for 'cyclocross' search term

To explicitly model and decompose the underlying trend and seasonality from the time series, we apply Prophet with default hyperparameter values and visualise the components separately:

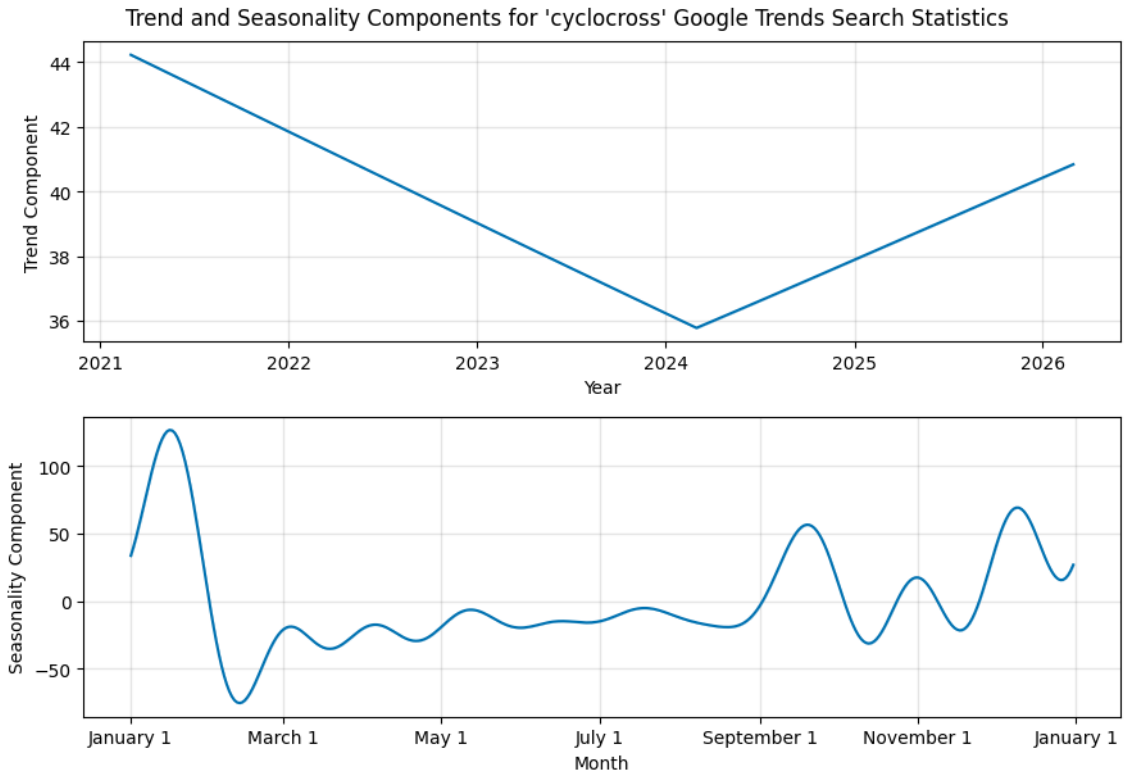


Figure 2. Trend and Seasonality components for 'cyclocross' Google Trends search term time series

The decomposed yearly seasonality confirms our hypothesis of the search acitivity being tied to the professional cycling calendar, as it shows more frequent activity from September onwards and peaks during January. The decomposed trend component visualises an overall declining interest up until the year 2024, during which it experiences an upward pivot.

Without any holidays/promotional periods specified in the model's initialisation, the time series model is based solely on the trend, seasonality, and residual components. In the illustration below we can see the in-sample predictions of the Prophet model from the earlier trend/seasonality illustration plotted on the 'cyclocross' search statistic time series.

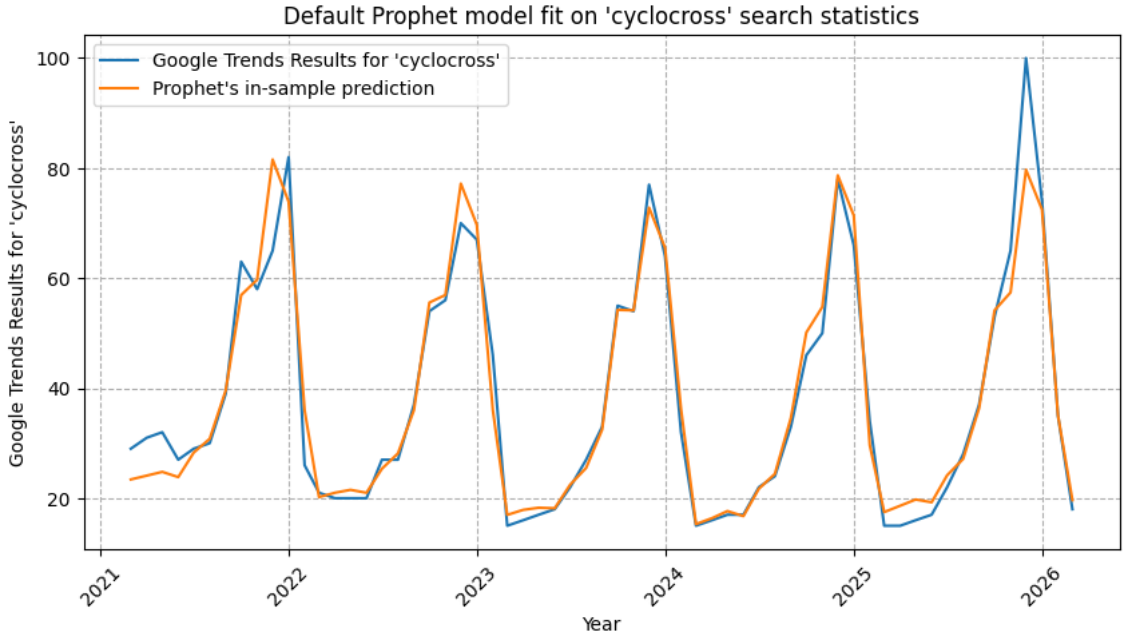


Figure 3. Default Prophet model fit on 'cycloross' Google Trends search term time series

To illustrate Prophet's hyperparameter tuning process, below is a visualisation of different Prophet's in-sample predictions from models with *yearly_seasonality* hyperparameter values of 1, 2, and 3 fit on the 'cyclocross' search statistic time series. No other hyperparameters are specified during the initialisation of the models.

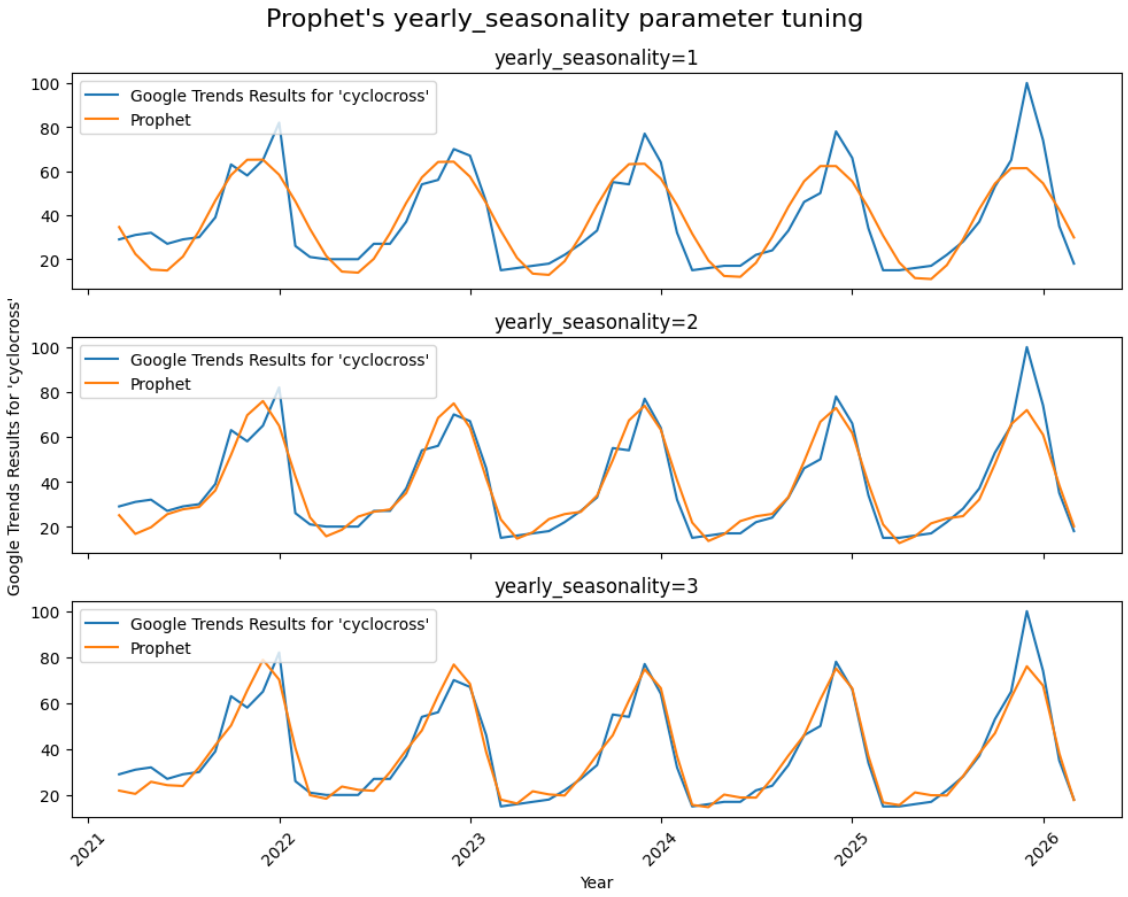


Figure 4. Results of tuning Prophet's yearly seasonality hyperparameter on 'cyclocross' search term time series from Google Trends

Judging visually, increasing the value of yearly_seasonality incrementally from 1 to 3 improves the fit of the model on this dataset. Finding optimal hyperparamater values with which the model learns the necessary patterns but doesn't fit on noise is however difficult manually when the data gets noisier and the amount of tuned hyperparameters increases. This problem is addressed with automated hyperparameter optimisation via randomised search in the pipeline of the proposed forecasting framework of this study.

3.2 XGBoost

XGBoost is a machine learning algorithm falling under the class of supervised machine learning algorithms (DMLC, n.d.). Being a supervised algorithm, it's goal is to learn the relationships between given input and output values, and later predict an output from

new input values (Liu & Wu, 2012). Supervised learning algorithms can be further divided into two distinct categories based on the tasks that they are designed to solve: classification algorithms and regression algorithms. The distinction can be made by looking at the output values; if the output values to be predicted consist of a limited number of values that represent class labels for the input data, we are talking about a classification task. Regression tasks on the other hand output continuous values (Liu & Wu, 2012). XGBoost can be used for both classification and regression tasks (XGBoost Contributors, 2026). Unless otherwise stated, references to XGBoost in this study refer to the regression algorithm, since forecasting the number of bicycles sold is a regression task.

XGBoost's architecture is based on optimising an ensemble of decision trees using gradient boosting (DMLC, n.d.), with the model's final prediction being obtained by summing the outputs from each individual tree in the ensemble (Chen & Guestrin, 2016, p. 786). Below is a simplified illustration on the architecture of a tree boosting algorithm, illustrating how each tree in the ensemble improves on the prediction error of the previous tree, with the prediction of the last tree being a sum of the whole ensemble:

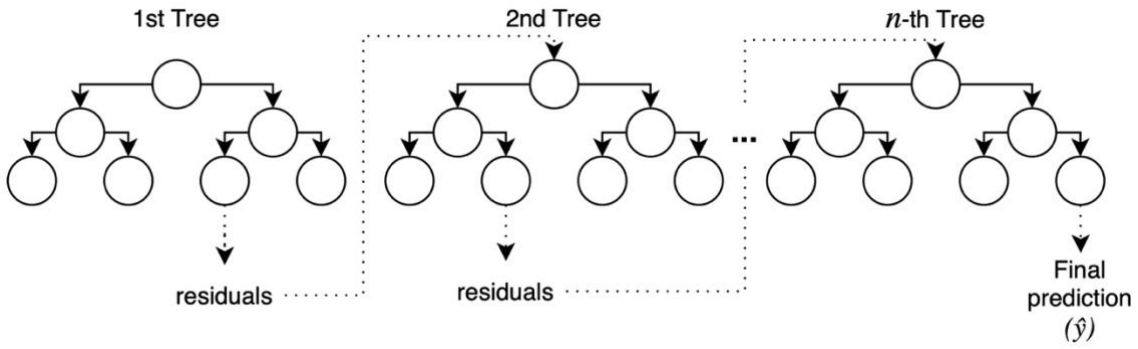


Figure 5. Architecture of a gradient boosted tree ensemble, adapted from Xu et al. (2023, p. 3)

XGBoost has multiple factors in the design of its architecture that differentiate it from a standard gradient boosted tree ensemble briefly introduced above. The properties characterising XGBoost include a regularised learning objective, the use of second-order gradients, sparsity-aware split finding, and system-level optimisation for scalable tree

boosting (Chen & Guestrin, 2016, p. 786-791). XGBoost is a powerful tool for prediction tasks in multiple domains, and its scalability to varying scenarios with fast runtimes when compared to alternative popular solutions makes it an attractive model with an ability to achieve accurate predictions and high computational efficiency (Chen & Guestrin, 2016, p. 785).

Although machine learning models used in regression tasks (such as XGBoost) aren't compatible with time series forecasting in their default configurations due to their inability to understand time and temporal dependencies, they can be applied to such tasks through purposeful feature engineering (Karmaker, 2025, p. 3). The feature engineering process essentially reformulates the time series forecasting problem into a supervised machine learning task suitable for XGBoost by creating features containing the needed temporal information from the timestamps and values of the time series.

The proposed forecasting approach of this thesis generates seven features for the XGBoost model: Prophet's prediction, 52 and 104 week lagged features of the target variable, month, week of year, quarter and timestep. The selection of an optimal feature set is implemented as an exhaustive grid search.

Feature engineering however adds a risk for introducing data leakage in the forecasting pipeline. Data leakage is a critical risk to be addressed in time series forecasting tasks, as it can result in compromised evaluation integrity of the models if left unnoticed (Albelali & Ahmed, 2025, p. 1). If future values get leaked to training sets, feature generation without access to future values during real-world forecasting will not match the inflated performance experienced during training. Leakage is often introduced to time series training data when features with temporal dependencies are generated before splitting the data into training and validation sets, informing the training set about future values (Albelali & Ahmed, 2025, p. 1). Avoiding leakage from the future to the training set can be achieved by generating only backward-looking features and computing statistics and scaling only after the training-test partition is completed.

Hyperparameter tuning is a central part of a machine learning pipeline, as the configuration of the specified hyperparameter values greatly affect the performance of the machine learning model (Rom et al., 2025, p. 1). The proposed forecasting framework of this thesis tunes the following XGBoost hyperparameters with a random search, with parameters left unspecified during initialisation resorting to their default values (descriptions adopted from DMLC, n.d.):

Table 2. XGBoost Hyperparameters Selected for Tuning

Hyperparameter	**Description**
objective	Specifies the learning task and the corresponding learning objective
n_estimators	Specifies the size of the forest to be trained
max_depth	Maximum depth of a tree
learning_rate	Step size shrinkage used in update to prevent overfitting
subsample	Subsample ratio of the training instances
colsample_bytree	Subsample ratio of columns when constructing each tree
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree
min_child_weight	Minimum sum of instance weight needed in a child
reg_alpha	L1 regularization term on weights
reg_lambda	L2 regularization term on weights

The specified hyperparameter value distributions and the hyperparameter values selected by the optimisation process of the forecasting pipeline for each of the XGBoost model instances are reported in *"XGBoost Models (Baseline & Hybrid)"* -chapters of this report.

3.3 Hybrid Model

The integration of Prophet and XGBoost is accomplished by feeding Prophet's forecast to XGBoost in it's feature set. Even though Prophet allows for explicit modelling of holidays and promotional events, the proposed forecasting framework of this thesis aims to keep the configuration of the models to a minimum by trusting on XGBoost to learn such patterns directly from the data. With this approach, the implementation of Prophet serves three purposes; 1. A baseline model that can be evaluated against other baselines and the more complex hybrid model. 2. A method for XGBoost's feature engineering, offering a feature that has already captured the global structure (trend, seasonality) of the data. 3. A tool in itself useful for analysing and understanding trend, seasonality, and residual patterns in the time series.

With the global structure of the data embedded in the Prophet's forecast, the task left for XGBoost is to decide the extent of which it should rely on the output of Prophet in relation to the other features in order to minimise prediction error in it's forecasting task.

4 Research Methodology

The research methodology followed in the construction of the proposed forecasting framework is based on three publications. The requirements for constructive research and definition for the construct are presented by Kari Lukka in his two publications from 2000 and 2001, while the followed research process model is adopted from Peffers et al (2007). The methodology combines constructive research, quantitative model evaluation, and qualitative practical validation. Constructive research frames the overall study because the thesis develops an artefact intended for practical use. Quantitative evaluation is used to compare forecasting accuracy across the hybrid model and baseline models, while the weak market test provides qualitative evidence that the artefact is relevant to the case company's decision-making context.

4.1 Constructive Research Approach

The core requirements for constructive research can be defined as follows:

1. Focuses on real world problems that require practical solutions
2. Produces an innovative construct designed to solve a real-world problem, including the development effort of the construct which tests it's practical applicability
3. Is a team effort with close collaboration among the researcher and the practice representatives expected to produce experimental learning
4. Is tied tightly to existing theoretical knowledge
5. Focuses on reflecting empirical findings back to the theory

(Lukka, 2000, p. 2)

Mapping of the constructive research criteria to the context of this thesis:

1. Challenging demand forecasting in bicycle retail where demand is highly seasonal
2. Hybrid demand forecasting model aiming to introduce a data driven decision support tool and improve demand prediction accuracy

3. Iterative approach to model development based on the real world sales data provided by the case company
4. Proposed forecasting method builds on existing forecasting theory by combining a statistical time-series method and machine learning
5. The empirical evaluation of the hybrid model communicates findings on forecasting performance that extend existing research

The construct in a constructive research approach has an infinite amount of definitions as it can be anything built or generated by a human, including but not limited to models, plans, diagrams, organisational structures, commercial products, information systems, etc. The common factor between these constructs is that they are not found, but invented and built in an effort aiming to construct something new and in the process contributing to shaping a new reality (Lukka, 2001).

The construct in this study is a hybrid time-series forecasting framework combining two predictive methods, Prophet and XGBoost. The development process of the artefact is an effort to create a forecasting approach solely based on historic values present in sales data, complementing the case company's existing managerial forecasting practices while exploring opportunities for systematic and data-driven decision making in their dynamic retail environment. As described by Lukka (2001), constructs aren't discovered but invented. Similarly, this study presents a purposefully developed forecasting framework that introduces a complementary predictive method to the case company's existing practices.

In this thesis, the artefact means the implemented forecasting framework rather than only an individual trained model. It includes the data preprocessing logic, feature generation, simple baselines, Prophet baseline, XGBoost baseline, hybrid XGBoost model, model-selection process, evaluation metrics, and the future-forecasting procedure. The artefact is therefore both a technical implementation and a decision-support construct that can be evaluated for accuracy, reproducibility, and practical usefulness.

4.2 Construct Development Process

Below is a summary of Peffers et al's (2007, p. 54) design science research methodology process model, introducing the different stages for a carrying out design science research in information systems research and the four different research entry points in their framework.

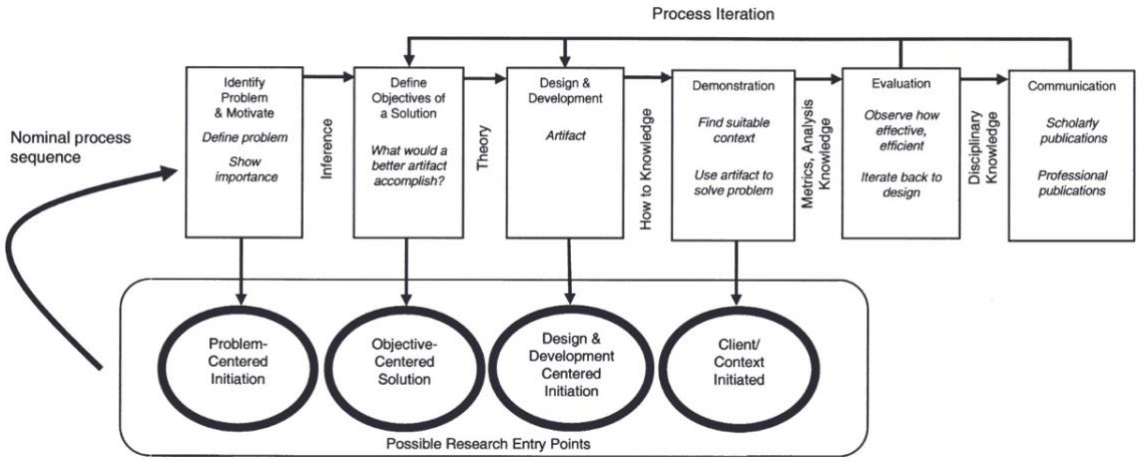


Figure 6. DSRM Process Model (Peffers et al., 2007, p. 54)

This study is best described as research with a "Design & Development Centered Initiation" as the target organisation was originally approached with an offer of a pro-bono demand forecasting model in exchange for access to their business data for this study. In contrast to a "Problem-Centered Initiation", where a specific business problem is the starting point, or an "Objective-Centered Inititation" with a defined and pursuable objective, this research was initiated with the artefact as the focal point. When the research process was initiated, there were no specified business problems to solve or an objective to pursue defined by the target organisation. By providing the forecasting model in exchange for data, the development phase in this study was used as an opportunity to identify organisational needs and refine the model's objectives through an iterative and collaborative effort with the target organisation.

The design science approach was selected because the research problem required building and evaluating a practical solution rather than only observing an existing process. Its iterative logic fits the study particularly well: model development and evaluation alternated, and each evaluation round informed the next development decision.

After the case company's participation in the thesis with their data was agreed on, the forecasting pipeline under development was ran with the initially provided dataset (consisting of only one product category). The initial dataset turned out to be quite noisy for such a forecasting task. After making the realisation that a dataset with more product categories was needed, a meeting was held with the case company's representatives which resulted in the decision of using a time series consisting of sales for all bicycle product categories. This dataset had much more distinct patterns and less noise, which was needed for the validation of the forecasting framework. Now with a reliable real-world dataset, and the hybrid forecasting model performing better than all of the baseline models the quantitative evaluation of the framework was considered as completed.

The main iterations consisted of testing the initial one-category dataset, identifying excessive noise, expanding the target series to all bicycle product categories, and programming workflows for tuning Prophet hyperparameters, selecting XGBoost feature sets, and tuning XGBoost hyperparameters in a pursue of accurate demand predictions. Finally, the implented pipeline was tested against both the primary case data and the additional Kaggle time series. In each round, forecast accuracy and practical interpretability guided whether the construct was refined further.

The development efforts and quantitative evaluation of the construct were followed by a demonstration session with the case company's representatives. The results of running the forecasting pipeline with the case company's data were communicated in this meeting with the results being presented largely in the same format as the results

presented in this report. The presentation was followed with discussions on the functionality of the forecasting framework and requirements for applying the construct in the case company's production environment. The framework's results sparking the interest of applying it in a real-world business environment marked the completion of the framework's qualitative evaluation.

4.3 Evaluation Metrics and Criteria for the Construct

The main metric for evaluating the point-level predictive accuracy of all the models in this study is mean absolute error (MAE):

$$\mathrm{MAE}(y, \hat{y}) = \frac{\sum_{i=0}^{N-1} |y_i - \hat{y}_i|}{N}$$

Equation 3. Mean Absolute Error (MAE)

where ***y*** is the true value of sold bikes and ***ŷ*** the value predicted by the model. MAE is chosen as the primary error metric due to it's easy explainability (Hyndman & Koehler, 2006, p. 687) to business stakeholders as it displays the magnitude of error in the same unit as the original data, in this case the number of bicycles sold. It also penalises large and small forecasting error with the same weights, further supporting it's interpretability.

The secondary evaluation metric used in this study is root mean square error (RMSE):

$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N}}$$

Equation 4. Root Mean Square Error (RMSE)

where ***y*** and ***ŷ*** maintain their denotations from the MAE formula. By squaring the errors before averaging and taking the square root, RMSE penalises large errors more than smaller ones. Using RMSE in paraller with MAE gives insight into which kind of prediction errors each model is struggling with or whether there are repeating residual patterns between models.

For comparing the performance of MAE and RMSE on a percentage scale, this thesis uses the following equation to compare the errors of two models:

$$\text{Improvement } (\%) = \frac{\text{Baseline Error} - \text{New Error}}{\text{Baseline Error}} \times 100$$

Equation 5. Improvement (%)

Lastly, mean bias error (MBE):

$$\text{MBE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)$$

Equation 6. Mean Bias Error (MBE)

is used do determine whether the models are consistently over- or underestimating in their predictions. The notation is consistent with the aforementioned MAE and RMSE formulas, where ***y*** and ***ŷ*** denote the actual and predicted values of the target variable.

As an addition to the quantitative point-level evaluation metrics, a weak market test is used as a qualitative evaluation criteria for the construct. A passed weak market test is a strong indication for the construct's practical applicability in a business setting. Passing a weak market test in this study's context would mean that the manager of the target organisation would be willing to apply the construct as a tool to support decision making (Rautiainen et al., 2017, p. 23). While the point-level performance of the proposed time-series forecasting method illustrates it's ability to generate predictions against baseline forecasts, the weak market test is an evaluation criteria that assesses whether the model's predictions and associated error margins meet the thresholds required by practical real-world use of the construct.

5 Empirical Study: Case Company

The case company discussed in this study is a bicycle retail chain located in Finland. The target organisation operates in the Finnish domestic market through four sales channels: three brick-and-mortar stores located in Pori, Seinäjoki and Vaasa, and an online store. This study focuses on predicting the amount of weekly unit-level bicycle sales (including bicycles from all bicycle product categories, one bicycle sale representing one unit in the data) from all of the aforementioned sales channels aggregated into one, collected from a three-year period. Modeling the sales data as one aggregated entity allows the study to work with a less volatile dataset than what selecting a smaller sample size by focusing on a single sales channel or limiting the amount of product categories would've resulted in.

As the case company already produces managerial forecasts based on historical sales performance and judgement, this study is an experimental effort aiming to broaden the case company's forecasting toolkit. The goal of this study is pursued by exploring possibilities of data-driven model building and decision making by implementing the case company's first hybrid time series forecasting model.

5.1 Data Preprocessing

The data source of this study is the case company's enterprise resource management system (ERP). The ERP system records the sales for each of the three brick and mortar stores in sales reports, which can then be exported in separate comma separated value (CSV) files for the purpose of this study's modeling. The sales of the case company's online store are distributed among these three locations depending on stock availability and other factors not relevant for this study, meaning that these sales reports contain the sales data of the online store among with the shop floor sales.

Time dependency of the recorded sales are indicated with the display of years and their week numbers in the exported reports. To make the data compatible with Prophet and

temporal feature engineering, the timestamps are converted to YYYY-MM-DD format and the exact dates are shifted to Thursdays, marking the middle points of the weeks. The three sales reports are then aggregated into one CSV file by summing the sales of the three stores for each time index. Lastly, the first and last week in the CSV file are removed as they might contain incomplete weekly sales information.

To preserve the actual sales figures a trade secret, each data point in the time series is transformed with a multiplicative-additive equation:

$$y' = Ay + b$$

Equation 7. Multiplicative-Additive Scaling

where ***y*** is the original value and ***y'*** the scaled value in the times series. ***A*** and ***b*** represent secret constants used in the transformation. This practice preserves the relative scale of the data, ensuring that the models learn the patterns present in the original data without exposing confidential figures.

To further protect the data privacy of the target organisation, the y-axis tickers are intentionally withheld on the plots representing the case company's data.

5.2 Exploratory Data Analysis

The Finnish bicycle retail industry has strong seasonality with sales increasing with the start of the cycling season, often peaking in the spring and early summer. This is phenomenon can already be seen in the lineplot of the raw data below.

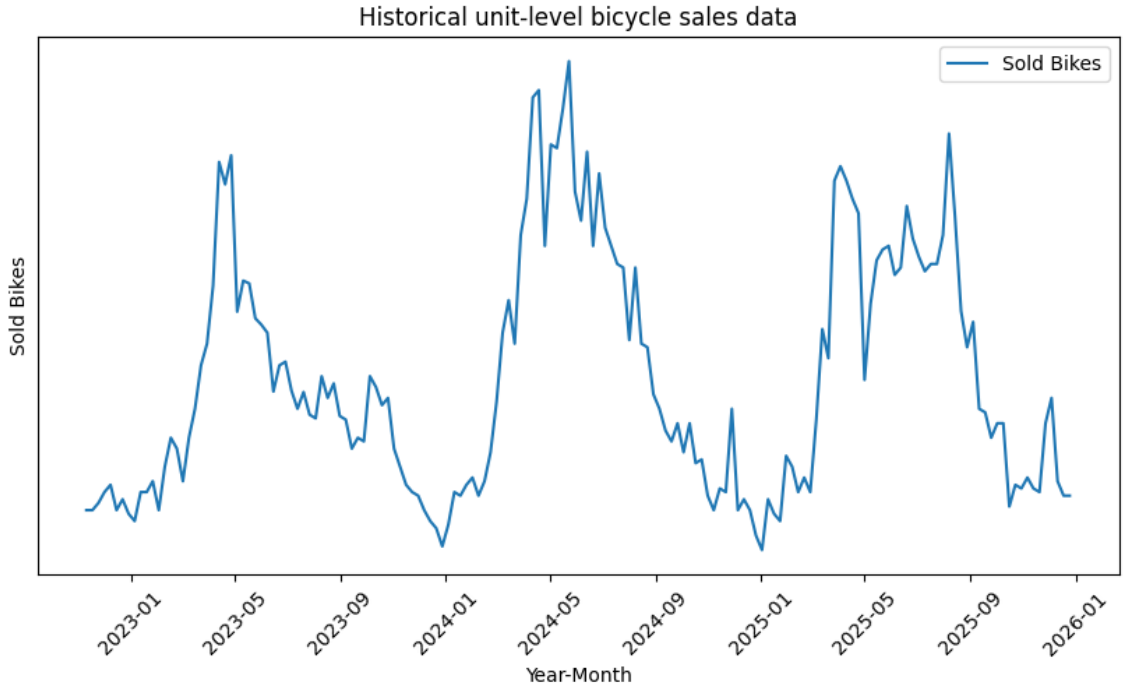


Figure 7. Line plot of the three year historical unit-level bicycle sales

When comparing different years against each other with a yearly seasonal plot, we can see that the different years follow a similar broad seasonal pattern. Comparing the characteristics of each year against the others, 2023 showed a peak during the start of the season, followed by a steady decline. The largest single week by sales volume was experienced during 2024, while 2025 shows a unique late-summer surge partly explained by the public's acceptance of the new normal after the discontinuation of tax-benefits for corporate bicycle leasing in Finland came to the public's attention earlier during the season. The news about the to-be legislative changes greatly characterised the 2025 season and was the main driver for the lower than expected sales during the months which high traffic was experienced in 2024.

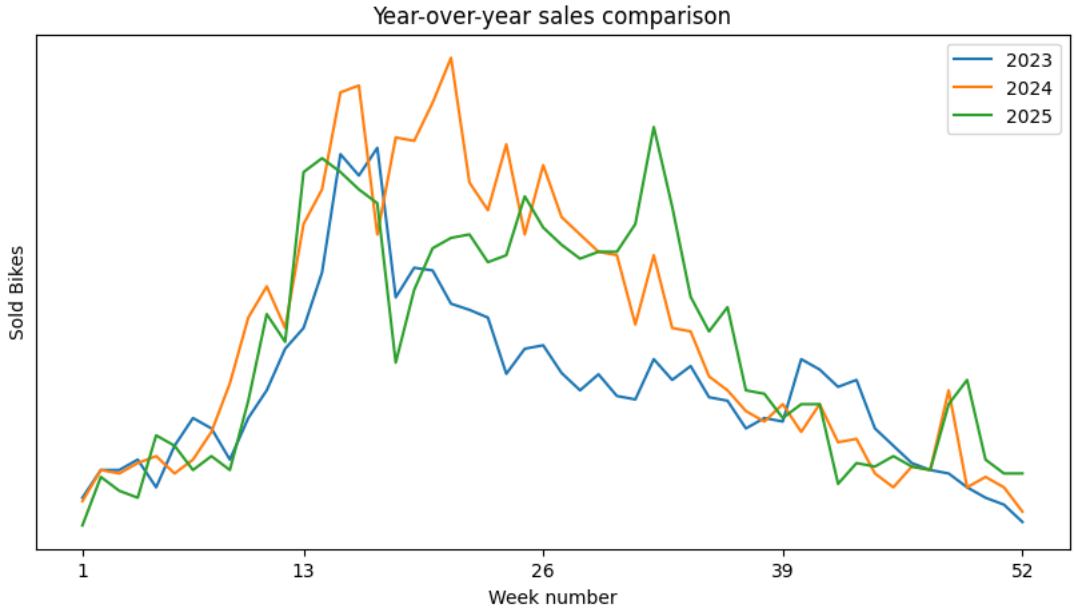


Figure 8. Year-over-year comparison of the three years of historical bicycle sales

Seasonal-trend decomposition based on Loess (STL) is a method for decomposing a time series (***Yv***) into three components:

$$Y_v = T_v + S_v + R_v$$

Equation 8. Seasonal-Trend decomposition based on Loess (STL)

Trend (***Tv***), Seasonality (***Sv***), and Remainder (***Rv***), for ***v=1*** to ***N*** (Cleveland et al., 1990, p. 3). Utilising the STL class implemented in the statsmodels.tsa.seasonal Python library (statsmodels, n.d.), we can run STL on the case company's data to visualise it's underlying trend, seasonality, and remainder components. This results in the four illustrations presented in the image below, which are, listing from top to bottom, the original time series ***(Y)***, trend ***(T)***, seasonality ***(S)***, and remainder ***(R)***.

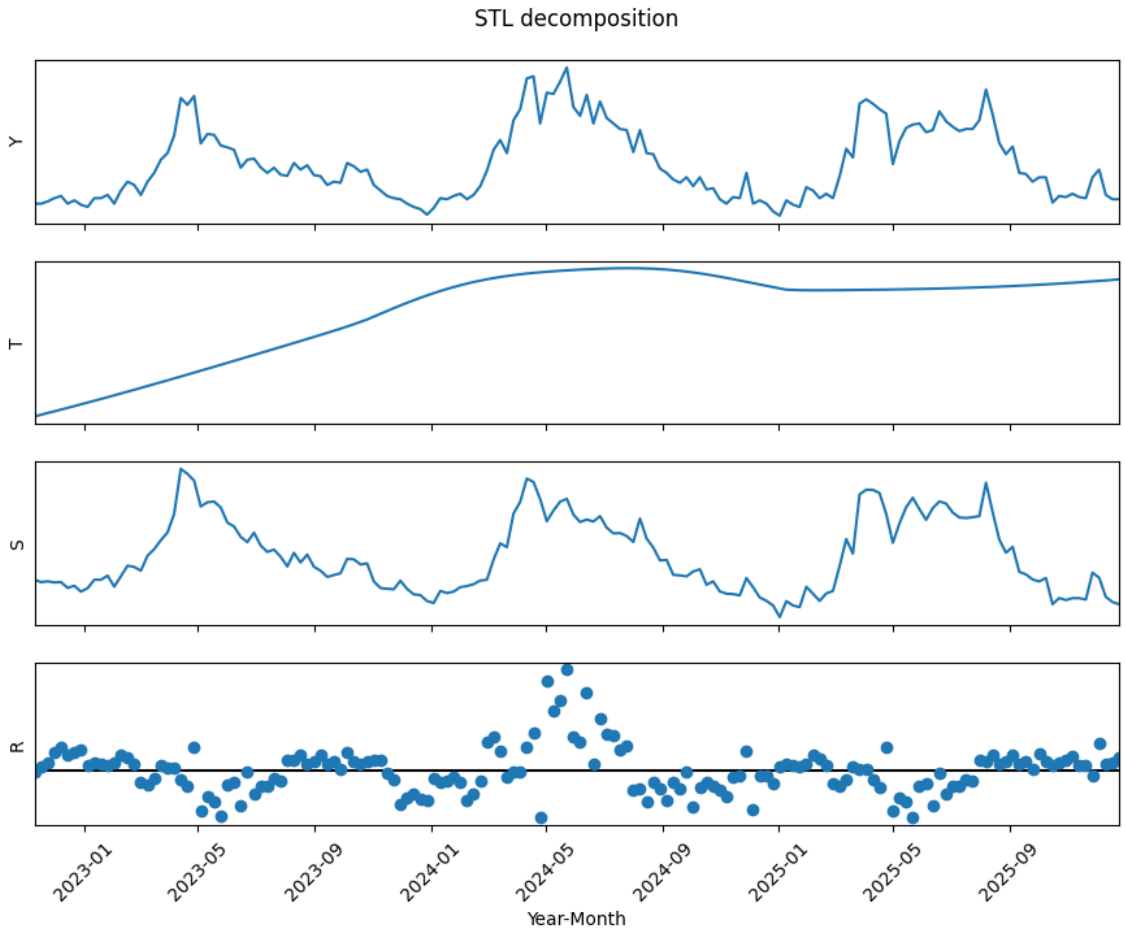


Figure 9. Seasonal Decomposition of Time Series by Loess (STL) applied to the three years of historical bicycle sales

The results of the STL reveal a non-stationary time series with a strong annual seasonality and a trend with growth from the beginning of 2023 up until mid-2024, after which it experiences a structural plateau. The residuals show a peak during the 2024 season, illustrating that the actual number of sold bikes was considerably higher during that period than what the trend and seasonality components alone describe.

The proposed forecasting framework of this study addresses the characteristics of the dataset discussed in this chapter by implementing a hybrid forecasting approach in which Prophet aims to capture the global structure of the data (trend, seasonality), and XGBoost is used to improve Prophet's forecast by receiving it's predictions in it's feature set.

5.3 Framework Implementation

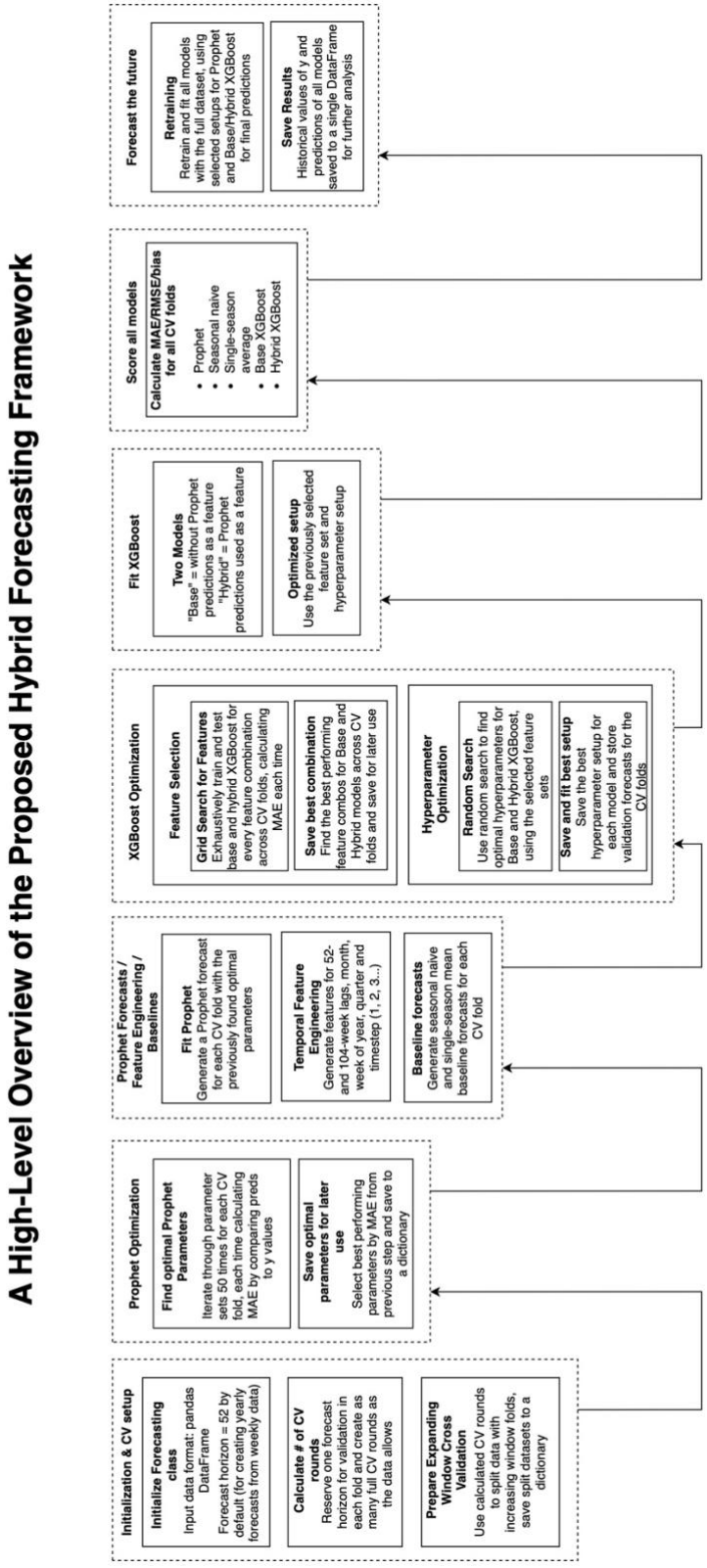


Figure 10. A High-Level Overview of the Proposed Hybrid Forecasting Framework

5.3.1 Used Tooling

The proposed forecasting framework is written in Python and utilises common data science and machine learning libraries such as NumPy, Pandas, SciPy, and scikit-learn with the main modeling tools from the Prophet and XGBoost Python packages.

5.3.2 Initialisation

The “Forecasting” class of the proposed Python program is designed to be imported to a programming environment such as a Jupyter Notebook used in producing the results of this study as a package. The forecasting pipeline is initialised by providing it with a Pandas dataframe as a parameter, containing a timestamped column “ds” and the target variable in a column “y”. The forecast horizon is set as default to 52 by the program (suitable for yearly forecasts with a weekly time series), but can be altered by specifying another horizon as a parameter during the initialisation of the program.

It should be noted that using a large sample size with a small forecasting horizon will lead to long computing times with feature and hyperparameter optimisation due to the data splitting logic which calculates the amount of cross-validation folds as:

$$N = \max\left(0, \left\lfloor \frac{L - 2H}{H} \right\rfloor\right)$$

Equation 9. Determining the Number of Cross-Validation Folds

where ***N*** is the number of cross-validation folds, ***L*** the sample size of the time series, and ***H*** the forecast horizon (52 by default). This logic ensures that the first cross-validation fold has at least a 2:1 ratio for training and testing data and then creates as many splits as possible with an expanding window with the size of the forecast horizon being the size of the test set. Below is a visualised example of what would be the the resulted cross-validation folds with a sample size of 318 and the default forecast horizon.

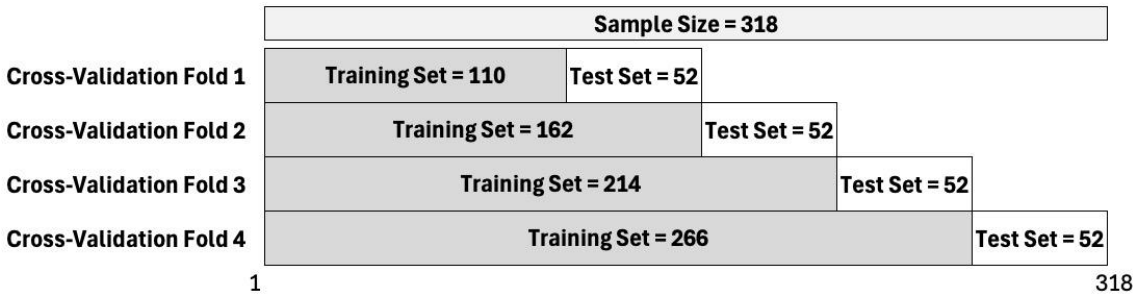


Figure 11. Example of expanding window cross-validation train-test splitting

After the initialisation of the "Forecasting" class, the pipeline of the framework is automatically run from end to end, without further required involvement of the end user in the process.

5.3.3 Baseline Forecasts

The pipeline of the forecasting framework fits five models in total; seasonal naïve, single-season average, Prophet, baseline XGBoost, and hybrid XGBoost. As the main proposition of this thesis is the hybrid XGBoost model, all of the other four can be considered as baselines. Three of the baselines; seasonal naïve, single-season average, and baseline XGBoost are standalone forecasting methods and completely separated from the hybrid XGBoost method. Prophet however acts as an independent baseline, but also as a method for feature engineering for the hybrid XGBoost approach.

The implementation of the baseline Prophet and XGBoost models are discussed in more detail in the following chapters. This section focuses on simple baselines which make naïve assumptions of the future resembling the past, making them very uncomplicated in their implementation, but surprisingly effective as forecasting methods (Hyndman & Athanasopoulos, 2021).

In time series forecasting with machine learning, model performance is often evaluated by using complex baseline models as a point of comparison (Beck et al., 2025, p. 394).

According to Beck et al. (2025, p. 394), the lack of a simple baselines may compromise the practical significance of the forecasting model's evaluation, as it doesn't take into account the unpredictability in highly volatile datasets. This study implements two simple baselines that do not rely on complex statistics:

1. Seasonal naïve, the value of the target variable from the previous season:

$$\hat{y}_{T+h} = y_{T+h-m}$$ **Equation 10.** Seasonal naïve

2. Single-season average, the average of the values of the target variable from the corresponding values in the previous season:

$$\hat{y}_{T+h} = \frac{1}{m}\sum_{i=1}^{m} y_{T-m+i}$$ **Equation 11.** Single-season average

In these expressions, $\hat{y}$ denotes the prediction made by the model, ***T*** the current time (eg. when changing from train to test data), ***h*** the forecast horizon, ***m*** the length of the season (equals the forecast horizon in the primary case of this study), and ***i*** the index of each observation within that season.

5.3.4 Prophet Model

Like mentioned before, Prophet acts both as a standalone forecasting method and a method for XGBoost's feature engineering in the proposed forecasting framework. For each cross-validation fold, the Prophet model is first fit on the training set, after which it generates predictions for the duration of the forecast horizon (equals the length of the test set) and calculates MAE against the true ***y*** values of the test set. Although the MAE could be calculated in-sample, this approach better simulates the circumstances in which

the actual predictions for the future are generated in with no access to the test set (the unknown values of ***y*** in the future).

The pipeline optimises Prophet's hyperparameters with random search, a search method used to maintain or improve predictive accuracy while keeping down the computational cost compared to an exhaustive grid search (Bergstra & Bengio, 2012, p. 281). For each cross-validation fold, the pipeline completes 50 iterations of fitting Prophet with random combinations of hyperparameter values specified below among the selected values for each hyperparameter via the optimisation process.

Table 3. Results of Prophet Hyperparameter Tuning (primary case)

Hyperparameter	**Hyperparameter Value Grid**	**Value Selected**
seasonality_mode	additive, multiplicative	additive
growth	linear, flat	flat
yearly_seasonality	True, 1, 2, 3	1
changepoint_prior_scale	0.001, 0.01, 0.1, 0.5	0.5
seasonality_prior_scale	0.01, 1.0, 10.0	10.0

The "Hyperparameter Value Grid" column in the table marks the values that the Prophet models were initialised with 50 times for each CV fold with random combinations, and the "Value Selected" marks the hyperparameter value that the pipeline found to produce the best results and which it selected for forecasting future values in the primary case of this study.

5.3.5 XGBoost Models (Baseline & Hybrid)

The feature optimisation process for XGBoost is implemented as an exhaustive grid search, with the pipeline fitting $2^6 - 1 = 63$ unique feature combinations for baseline XGBoost and $2^7 - 1 = 127$ for the hybrid approach on every cross validation fold without

specifying hyperparameters for the models. Best feature set is selected by choosing the set of features with the lowest average MAE over the completed cross-validation folds. Below are detailed descriptions of each of the features along with the results on which features got selected for each of the models during the optimisation pipeline with the case company's data.

Table 4. Results of XGBoost Feature Selection (primary case)

Feature	**Description**	**Selected for Base**	**Selected for Hybrid**
'yhat'	Prophet's forecast	-	YES
'lag_52w'	Value of y 52 weeks ago	NO	NO
'lag_104w'	Value of y 104 weeks ago	NO	YES
'month'	Values between 1-12	YES	NO
'weekofyear'	Values between 1-52	NO	NO
'quarter'	Values between 1-4	NO	YES
'timestep'	Values in range 1,2,3, … , n	NO	YES

After the completed feature selection process, the pipeline tunes the hyperparameters for both base and hybrid XGBoost models by initialising the models with the feature sets found to be optimal in the previous feature selection stage. The hyperparameter optimisation process is carried out with a randomised search of 50 iterations for both models with the provided hyperparameter value distributions on every cross-validation round. The hyperparameter set with the lowest average MAE over the cross-validation rounds is saved to be used with the optimised feature set to forecast the future. The table below specifies the hyperparameter value distributions and the individual values selected by the pipeline for each hyperparameter during the optimisation process with the case company's data.

Table 5. Results of XGBoost Hyperparameter Tuning (primary case)

Hyperparameter	**Hyperparameter Value Grid**	**Value Selected (Base)**	**Value Selected (Hybrid)**
objective	reg:squarederror, reg:absoluteerror	reg:absoluteerror	reg:absoluteerror
n_estimators	randint(50, 250)	100	147
max_depth	randint(2, 5)	2	4
learning_rate	uniform(0.01, 0.09)	0.047566990283390106	0.08979554340555938
subsample	uniform(0.6, 0.4)	0.7292811728083021	0.7312610669898928
colsample_bytree	uniform(0.6, 0.4)	0.6027808522124762	0.7587135308855554
gamma	uniform(0, 0.3)	0.1532241907732697	0.01523055931181908
min_child_weight	randint(1, 6)	1	4
reg_alpha	uniform(0, 0.5)	0.16880758570181398	0.21923706150904348
reg_lambda	uniform(1, 1.5)	2.4143645558687785	2.008039202942799

The “Hyperparameter Value Grid” column in the table marks the value ranges that the XGBoost models were initialised with 50 times for each CV fold with random combinations, and the “Value Selected (Base/Hybrid)” columns mark the hyperparameter values that the pipeline found to produce the best results for each XGBoost model and which it selected for forecasting future values in the primary case of this study.

5.3.6 Evaluation of Prediction Accuracy

The limited number of available historical data and the set 52-week forecast horizon allows for only one iteration for feature and hyperparameter optimisation for the primary case of this study. This essentially means that with the provided the case

company's data there is no cross-validation taking place, and the models get trained and validated on a single train-test split illustrated by the following image. The cross-validation logic of the proposed construct is however explored later in this study by providing it another time series dataset from Kaggle with a larger sample size.

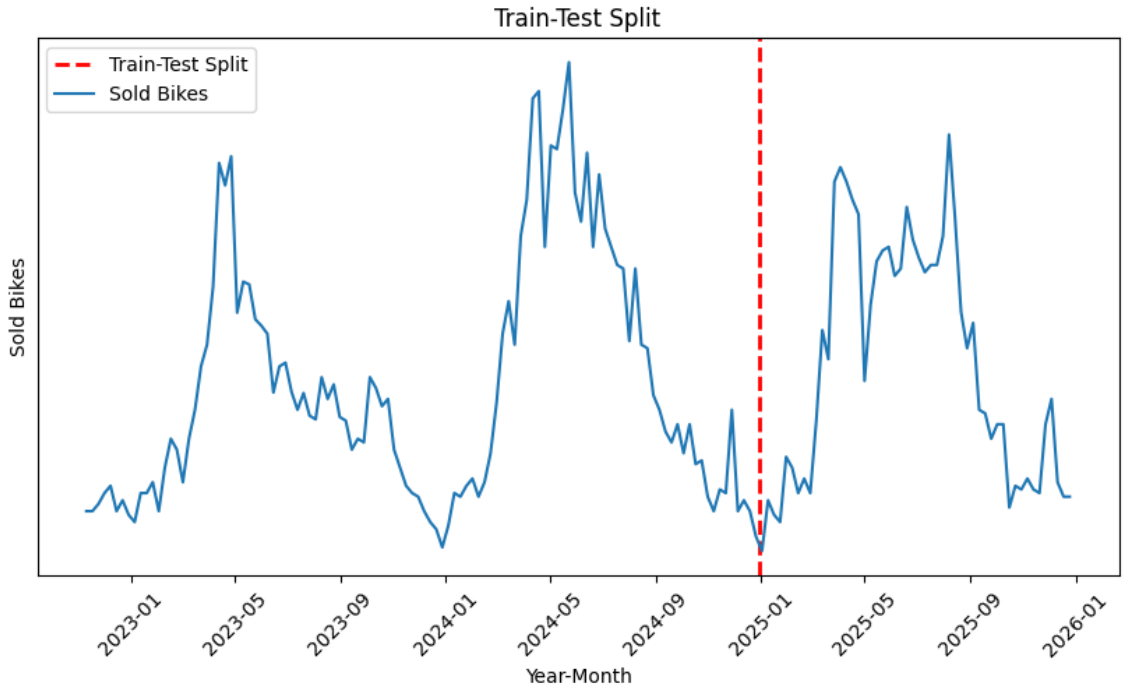


Figure 12. Train-Test split of the historical three years of bicycle sales data

Running the pipeline with the provided data, the framework saves the actual ***y*** values along with the predictions made by each model into a single Pandas DataFrame object separately for each cross-validation fold. Since our data allows for only one train-test split, a visual representation of the fit models against ***y*** values of the time series is illustrated below followed by a table of the point-level evaluation metrics for each individual model sorted by MAE from lowest to highest.

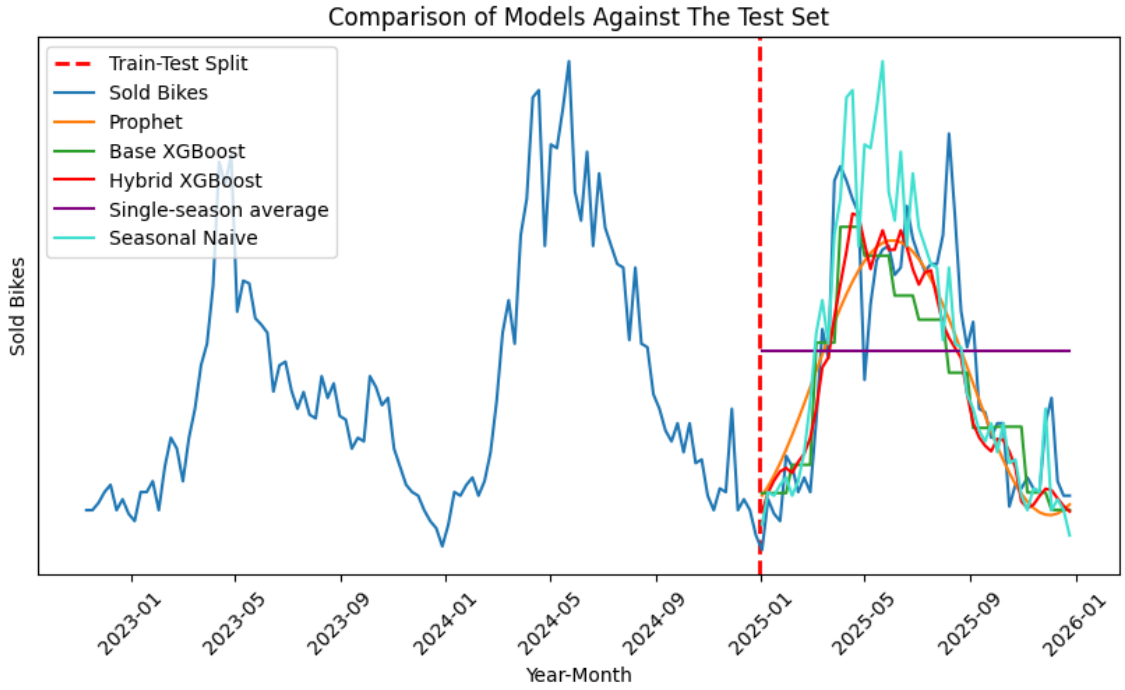


Figure 13. Results of the different forecasting models plotted against each other and the actual historical bicycle sales in the test set

Table 6. Validation Results for Each Forecasting Model (primary case)

Model	**MAE**	**RMSE**	**MBE**
Hybrid XGBoost	1390.947	1996.267	-623.054
Prophet	1559.973	2144.619	-472.525
Base XGBoost	1591.183	2253.389	-725.969
Seasonal naïve	1811.479	2532.7	429.722
Single-season average	3687.061	4052.95	429.722

Comparing the performance of the models, we notice that the hybrid XGBoost scores the lowest both with an MAE of 1390.947, and an RMSE of 1996.267. This informs us that on average the prediction errors of the hybrid XGBoost model are the lowest when penalising all errors equally and when putting more emphasis on large prediction errors. Comparing to the second most accurate model, standalone Prophet, the Hybrid XGBoost approach offers an improved prediction accuracy of 10.84% by MAE and 6.92% by RMSE. Compared to standalone XGBoost, the hybrid improves prediction accuracy by 12.58%

with MAE and 11.41% by RMSE. The naïve benchmarks populate the back end of the list by the hybrid model improving the accuracy of the Seasonal naïve by 23.21%/21.18% (MAE/RMSE). Single-season average scores the worst from our models with poor compared predictive performance, as was expected from a simple average model and strong seasonal fluctuations in the data.

Comparing the absolute values of the MBE's of the models, Seasonal naïve scores the lowest (with single-season average), followed by Prophet, Hybrid XGBoost, and Base XGBoost in the mentioned order. Comparing the signed values of the MBE's, the statistical, machine learning, and hybrid approaches show a tendency to systematically underpredict while the naïve methods show a tendency for overpredicting.

Visualising the predictions of the best performing three models against ***y*** values of the dataset, we can see result of the statistical modeling approach of Prophet being a smooth harmonic representation of the season. The opposite can be said of the baseline XGBoost's staircase-resembling graph. The influence of incorporating the results of Prophet's smooth seasonality representation in the hybrid XGBoost's feature set is evident when comparing it's output to the end result of the baseline machine learning approach.

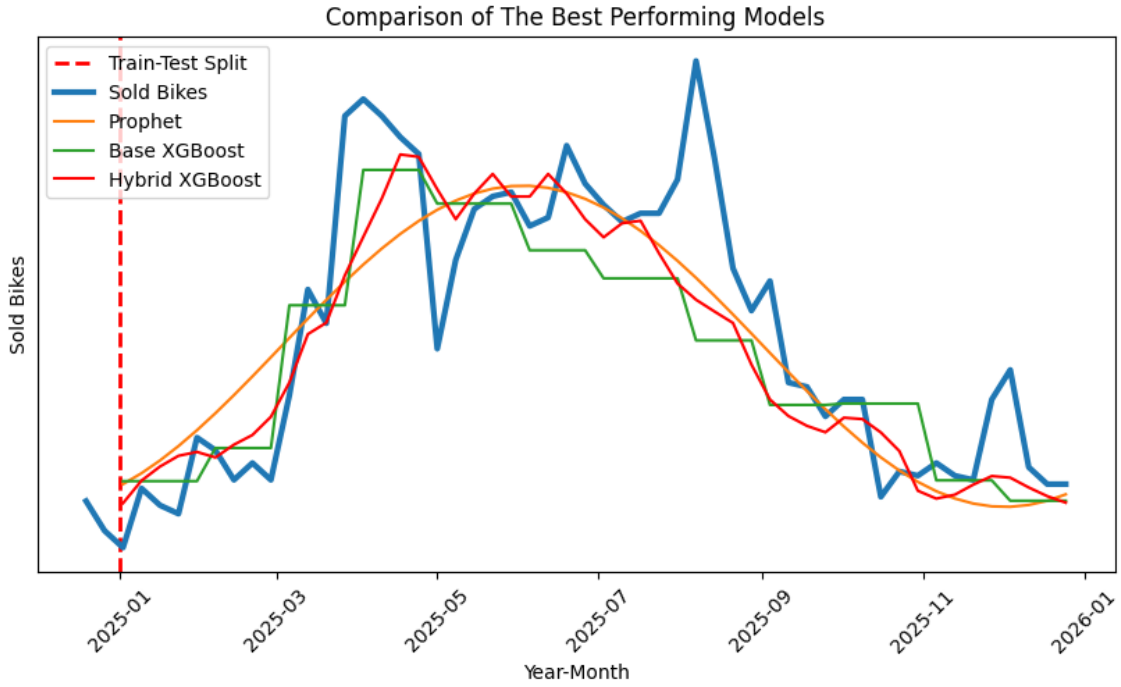


Figure 14. Results of the best performing three forecasting models plotted against each other and the actual historical bicycle sales in the test set

5.3.7 Forecasting the Future

The forecasting framework produces it's predictions of the future by training each statistical and machine learning model with the features and hyperparameters found to be optimal in the previous stage. The difference being this time the utilisation of the whole dataset as training data and having no test set during the time of producing the forecast to calculate prediction accuracies for the models.

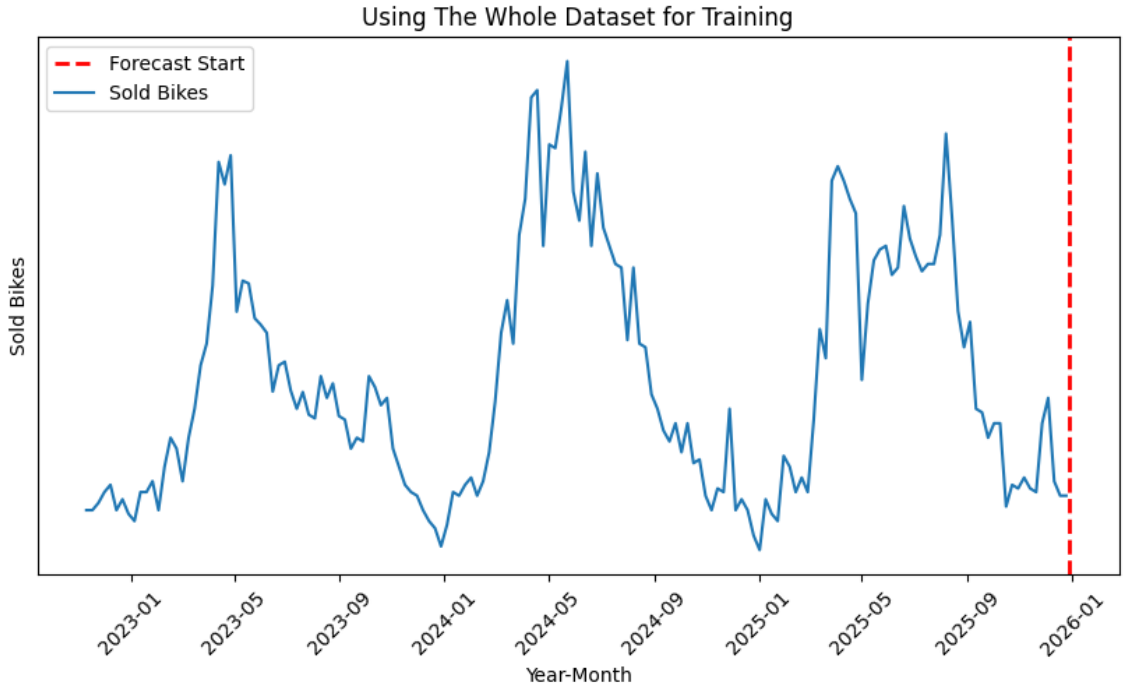


Figure 15. Using the whole time series for training data when forecasting into the future

When forecasting 52 weeks into the future we can see a similar pattern to the one that was seen in the previous training round; having the output of Prophet in the feature set of XGBoost smooths out it's behaviour. It is also interesting to see how the Hybrid XGBoost is the only model able to pick up on the recurring Black Friday/holiday season sales spike despite the limited sample size of the time series and the exclusion of exogenous variables in it's feature set.

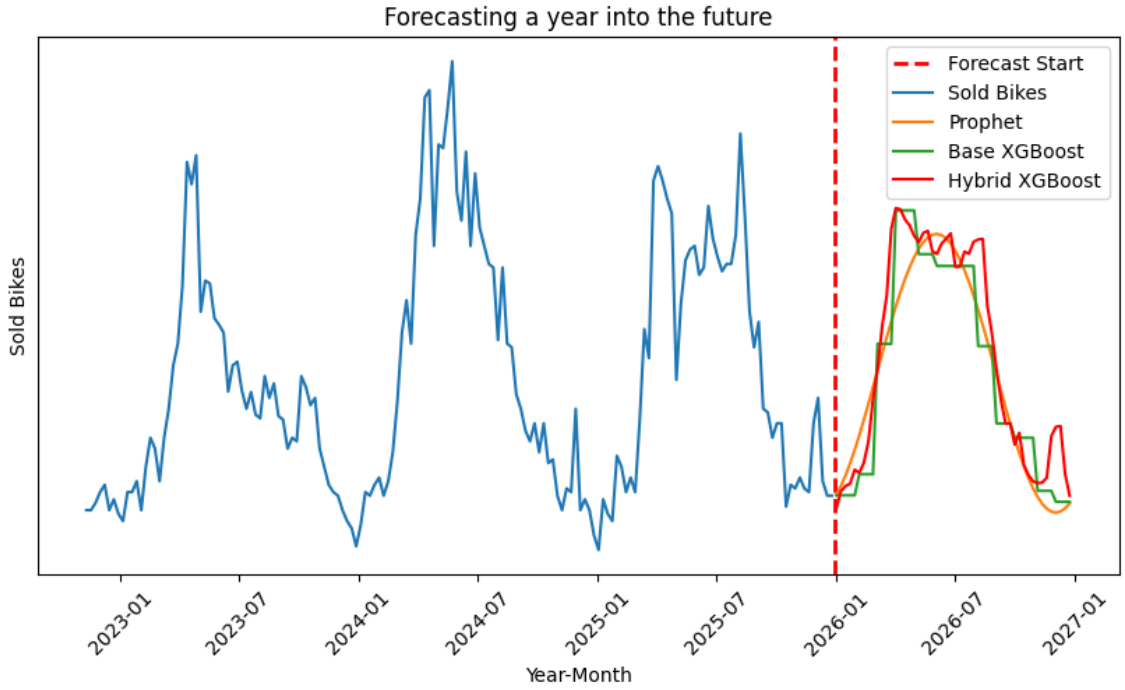


Figure 16. Forecasting a year into the future by training the models with three years of historical bicycle sales

5.3.8 Applicability of the Construct in a Business Environment

During the final meeting with the target organisation, the managerial board of the case company indicated a strong interest towards further utilisation of the demand forecasting pipeline and discussions were had on the architectural requirements for deploying the pipeline in production. With these discussions, the weak market test for the construct was considered as passed in line with the definition of Rautiainen et al. (2017) who describe the criteria for the test as the manager's willingness to apply the construct as a decision supporting tool. Although the production deployment of the forecasting framework is out-of-scope for this study, the passing of the weak market test provides empirical validation that the model addresses a genuine practical need within the target organisation's operations.

6 Validation of the Construct Beyond the Primary Case

To explore generalisability and display the cross-validation logic of the proposed construct, the forecasting pipeline is applied to an external dataset from Kaggle. The dataset contains simulated sales data spanning over 10 years from 2010 to 2019 with 7 stores and 10 products split to separate csv files for training and testing (Cortinhas, n.d.). For the demonstration purposes of this study, the focus is on a filtered sample from the training set of the data. The filtering query run on the data is defined as:

$$\mathcal{D}_{filtered} = \{x \in \mathcal{D} \mid x_{\text{store}} = 0,\ x_{\text{product}} = 0,\ 2011 \leq \text{year}(x_{\text{date}}) < 2016\}$$

Equation 12. Filtering Query for Kaggle's Time Series

With this, we end up with a dataset containing five years of data including points from years 2011 to 2015, containing observations only from store 0 and product 0. The original data has daily observations, but is aggregated to a weekly level in order to match the granularity of the data in the primary case of this study. Below is a lineplot of the dataset after the described preprocessing steps.

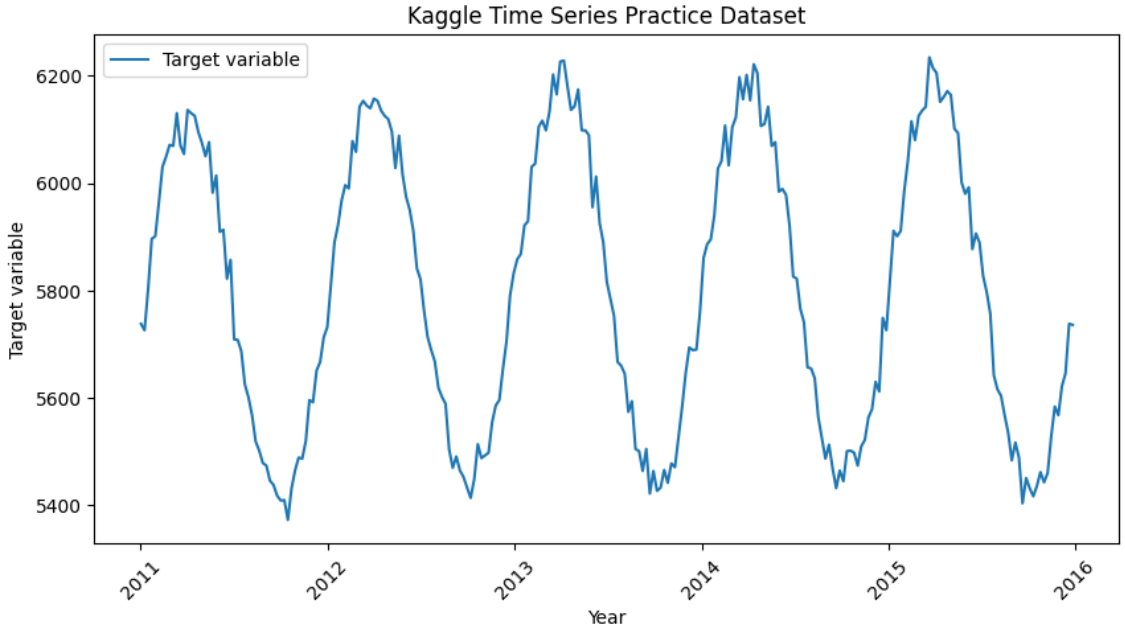


Figure 17. Lineplot of the Kaggle Time Series Practice Dataset after applying the filtering query

Now with a larger sample size compared to the data from the primary case of this study, the pipeline is able to complete multiple rounds cross-validation for identifying optimal features and hyperparameters. The larger amount of observations also allows for comparing the development of model performance with increasing amounts of historical data in the training sets. Below is an illustration visualising the realised train-test splits for each cross validation fold and the individual models fit against the ***y*** values on each round.

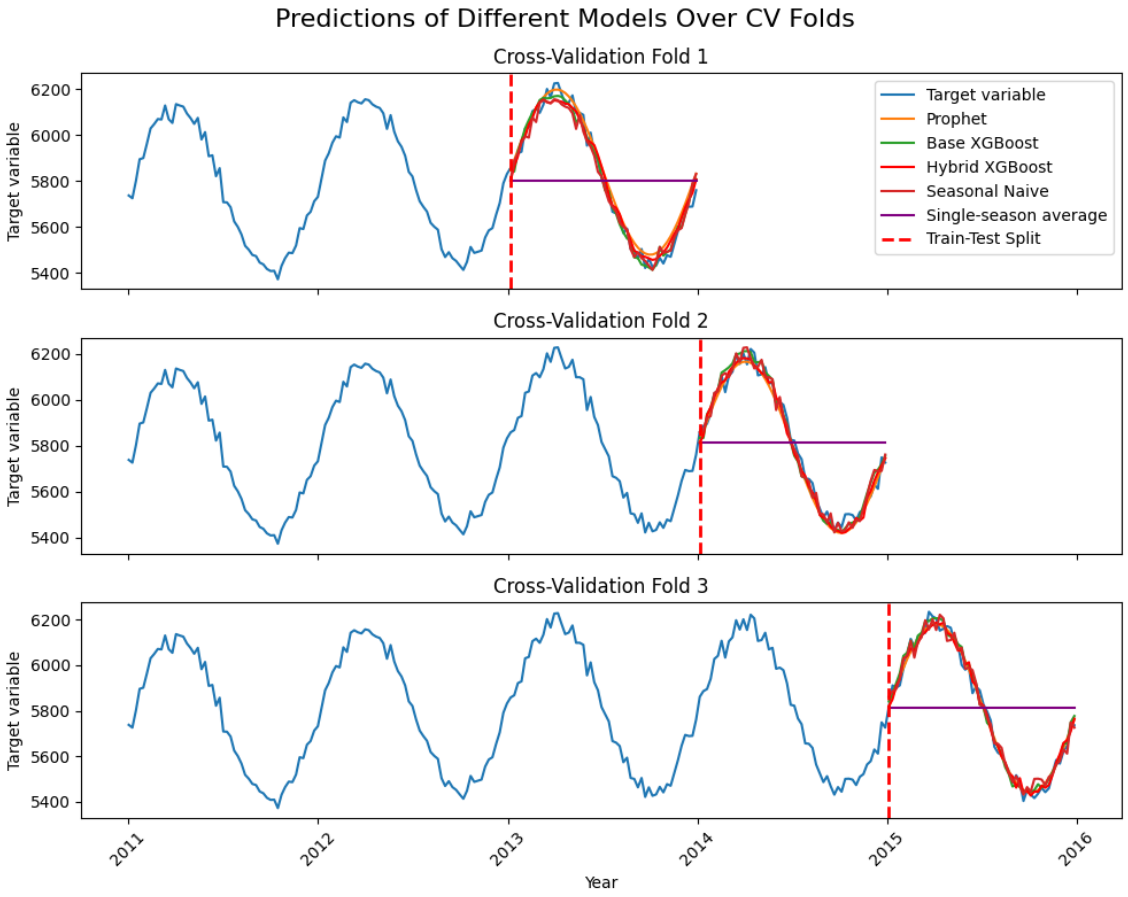


Figure 18. Model predictions plotted against the test set over the three cross-validation folds

6.1 Prophet

Results of Prophet's hyperparameter optimisation process with the Time Series Practice Dataset from Kaggle:

Table 7. Results of Prophet Hyperparameter Tuning (Kaggle)

Hyperparameter	**Hyperparameter Value Grid**	**Value Selected**
seasonality_mode	additive, multiplicative	multiplicative
growth	linear, flat	linear
yearly_seasonality	True, 1, 2, 3	1
changepoint_prior_scale	0.001, 0.01, 0.1, 0.5	0.01
seasonality_prior_scale	0.01, 1.0, 10.0	10.0

6.2 XGBoost Models (Baseline & Hybrid)

Results of the base and hybrid XGBoost model's feature selection process with the Time Series Practice Dataset from Kaggle:

Table 8. Results of XGBoost Feature Selection (Kaggle)

Feature	Description	Selected for Base	Selected for Hybrid
'yhat'	Prophet prediction	-	YES
'lag_52w'	Value of y 52 weeks ago	YES	NO
'lag_104w'	Value of y 104 weeks ago	NO	YES
'month'	Values between 1-12	NO	NO
'weekofyear'	Values between 1-52	YES	YES
'quarter'	Values between 1-4	NO	NO
'timestep'	Values in range 1,2,3, … , n	NO	NO

Results of the base and hybrid XGBoost model's hyperparameter tuning process with the Time Series Practice Dataset from Kaggle:

Table 9. Results of XGBoost Hyperparameter Tuning (Kaggle)

Hyperparameter	**Values tried/ range**	**Value Selected (Base)**	**Value Selected (Hybrid)**
objective	reg:squarederror, reg:absoluteerror	reg:squarederror	reg:absoluteerror
n_estimators	randint(50, 250)	145	168
max_depth	randint(2, 5)	4	4
learning_rate	uniform(0.01, 0.09)	0.05608837524693529	0.06114777430019244
subsample	uniform(0.6, 0.4)	0.7546941385202149	0.7572390898667042
colsample_bytree	uniform(0.6, 0.4)	0.871025744736913	0.9396893641976711
gamma	uniform(0, 0.3)	0.004976348678356846	0.197283867690103
min_child_weight	randint(1, 6)	4	3
reg_alpha	uniform(0, 0.5)	0.08718321450249572	0.12199482168954179
reg_lambda	uniform(1, 1.5)	2.036406607153699	2.4595158321286683

6.3 Evaluation of Prediction Accuracy

The following table contains the point-level error metrics for each predictive model over all completed cross-validation folds:

Table 10. Validation Results for Each Forecasting Model (Kaggle)

Model, CV Round	MAE	RMSE	MBE
Hybrid XGBoost			
Round 1	30.303	36.149	4.830
Round 2	30.158	35.792	-13.224
Round 3	24.136	29.936	0.801
Mean (Round 1-3)	28.199	33.959	-2.531
Base XGBoost			
Round 1	27.219	34.147	-2.853
Round 2	31.815	39.286	-0.797
Round 3	26.769	33.052	9.282
Mean (Round 1-3)	28.601	35.495	1.877
Prophet			
Round 1	35.032	41.615	26.962
Round 2	30.801	37.328	-22.247
Round 3	23.187	28.579	1.900
Mean (Round 1-3)	29.673	35.841	2.205
Seasonal naïve			
Round 1	34.865	41.592	-10.404
Round 2	31.385	39.567	-2.269
Round 3	30.596	39.256	6.365
Mean (Round 1-3)	32.282	40.138	-2.103
Single-season average			
Round 1	242.038	269.016	-10.404
Round 2	236.960	261.166	-2.269
Round 3	241.584	269.792	6.365
Mean (Round 1-3)	240.194	266.658	-2.103

When comparing the mean forecasting accuracy of the models over the three cross-validation rounds, Hybrid XGBoost scores the lowest with a MAE of 28.199 and an RMSE

of 33.959. The standalone machine learning and statistical approaches are again the second and third most accurate, this time with the baseline XGBoost being more accurate by MAE and RMSE comparing to the Prophet's forecast. Each of the statistical, machine learning and hybrid approaches again perform better than both of the simple baselines in prediction accuracy with the seasonal naïve being the relevant point of comparison from the simple models.

To summarise the mean forecasting accuracies, Hybrid XGBoost offers the following improvements in prediction accuracies by MAE and RMSE when comparing to the baseline models:

Table 11. Hybrid XGBoost's Improvement Margin Over The Baselines (Kaggle)

Model	**MAE**	**RMSE**
Base XGBoost	1.41%	4.33%
Prophet	4.97%	5.25%
Seasonal naïve	12.65%	15.39%
Single-season average	88.26%	87.26%

The visualised development of the MAE's below illustrates the comparative performance of the models over the three separate cross-validation folds excluding the single-season average. The graph informs us that no single model shows superior performance over the iterative validation process consistently. In fact the models swap their comparative order in the top four in each of the sequential validation rounds.

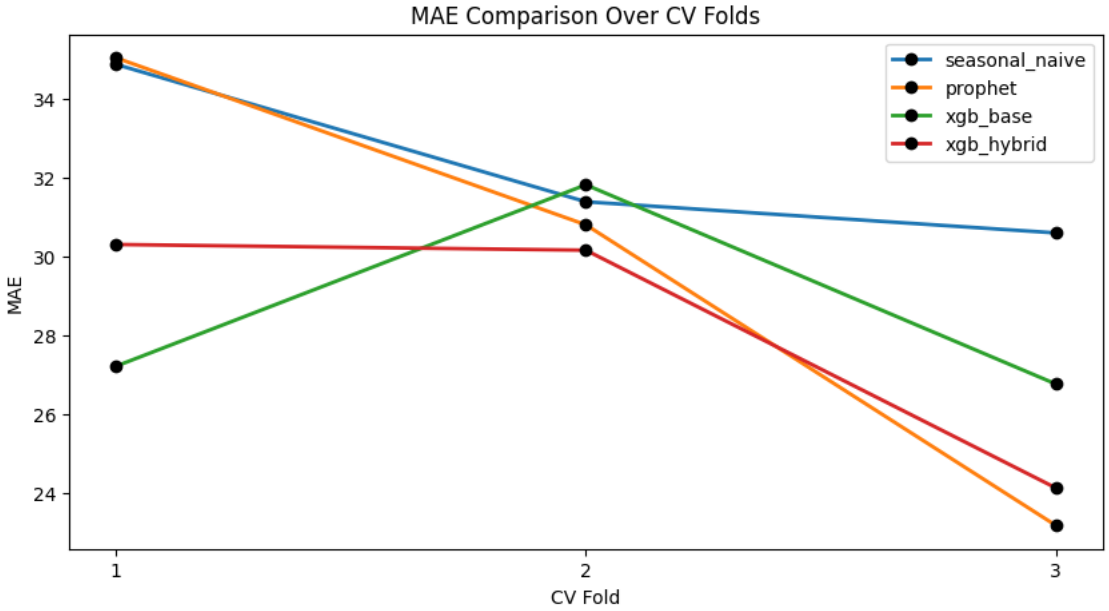


Figure 19. Comparing the development of mean absolute error between models over the completed three rounds of cross validation

Taking a closer look at the output of the statistical and machine learning models from the pipeline's last cross-validation fold, we can observe that the models behave very similarly comparing each individual forecasting approach. Each model is able to effectively capture the seasonal signal in the data. When comparing the baseline and hybrid machine learning approaches, the staircase-resembling decision boundaries present in the results of the primary case of this study can't be distinguished from the output of the baseline model with no access to Prophet's harmonic predictions.

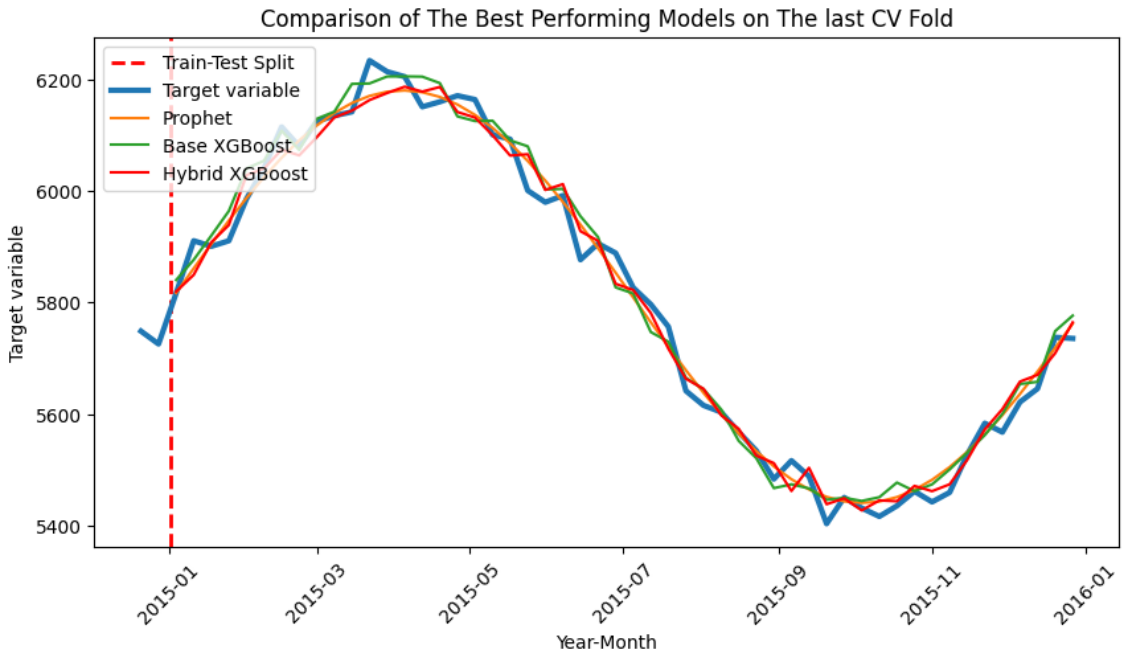


Figure 20. Comparing the best performing models against each other and the actual values of the time series on the last completed cross-validation fold

Further analysing the behaviour of the models by their demonstrated MBE's, the baseline XGBoost model scores the lowest absolute value of 1.877, followed by the simple models with their equal MBE at -2.103. Prophet has the second-largest absolute MBE with a score of 2.205, and hybrid XGBoost is the largest with -2.531. The signed MBE values indicate that the baseline XGBoost and Prophet models show a tendency for slightly overpredicting, while the simple baselines and the hybrid XGBoost model have a tendency for moderate underprediction.

6.4 Forecasting the Future

Predicting 52 weeks into the future produces three forecasts with the global behaviour of the predictions closely resembling each other:

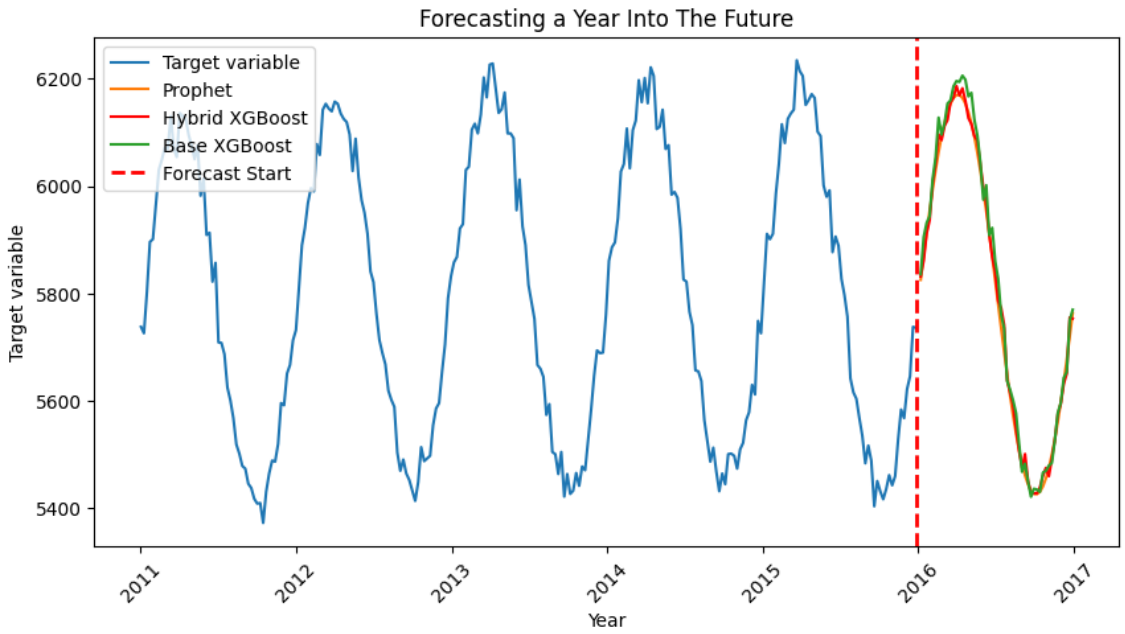


Figure 21. Forecasting a year into the future after training the models with the whole time series from Kaggle

7 Discussion

The results of this study support the use of a hybrid forecasting approach in univariate time series forecasting in bicycle retail. In the two cases discussed in this study, XGBoost trained with the output of a Prophet model in its feature set produced the most accurate forecasting results both by MAE and RMSE when compared to simple baselines and the standalone statistical and machine learning approaches. This discussion chapter answers the research questions, interprets what the findings mean in practice, presents the study's contribution, and evaluates the limitations and future development directions for the artefact.

7.1 Answers to Research Questions

The answer to the primary research question of this study: *"How does a hybrid Prophet/XGBoost model perform when predicting bicycle sales compared to standalone Prophet, XGBoost and other baseline forecasts?"* is clear. The Hybrid approach shows superior predictive accuracy when compared to the simple baselines and the standalone statistical and machine learning approaches both with the case company's sales data and during the explored generalisability of the framework outside of the primary case of the study. With the case company's data, the hybrid approach demonstrates a 10.84% improvement of prediction accuracy by MAE and 6.92% by RMSE over the second-most accurate model (Prophet). With the dataset from Kaggle, the hybrid approach shows a 1.41% improvement on prediction accuracy by MAE and 4.33% by RMSE over the second-most accurate model (standalone XGBoost).

In practical terms, the results mean that the hybrid model is useful when demand contains both smooth seasonal structure and irregular nonlinear variation. Prophet provides a stable representation of the seasonal sales pattern, while XGBoost can adjust the forecast using lagged and calendar-based features. The model could be less suitable in situations where future demand is driven primarily by information not present in

historical sales, such as sudden marketing campaigns, weather anomalies, supplier disruptions, or major product-mix changes.

Addressing the secondary research questions of this study:

"Does aggregating unit-level bicycle sales data from the case company's four sales channels over a three year window provide a sufficient data set for hybrid time-series modeling? - With machine learning models showing more potential with time series forecasting tasks comparing to statistical approaches as the sample size of the dataset increases (Cerqueira et al., 2019), the question of whether the three-year sample size of weekly data from the case company would be enough for a successful machine learning implementation was raised during the initial stages of this study. Another concern was the quality of the data, as initial experiments with data consisting of a single product category turned out to be too noisy for such modelling. Aggregating unit-level sales of all bicycle product categories from the case company's four sales channels however provided a stable-enough dataset for validating the forecasting methods. Coming back to the question regarding the suitability and size of case company's dataset, training the models on the case company's data resulted in Prophet demonstrating an improvement of 1.96% by MAE and 4.83% by RMSE over the standalone (non-hybrid) implementation of XGBoost, and the Hybrid approach showing an improvement of 12.58% by MAE and 11.41% by RMSE over the predictions of the standalone XGBoost and 10.84% (MAE), 6.92% (RMSE) over Prophet. These results indicate that while the statistical model has a slight edge on the pure machine learning approach, the hybrid model shows a significant improvement over both the standalone statistical and machine learning models.

"Which features carry the most weight in the hybrid model's predictions?" - The native plot_importance method in the xgboost Python package (DMLC, n.d.) produces useful visualisations for comparing the importance scores of individual features in the XGBoost models. Analysing the feature importance scores of the hybrid model trained with the case company's data to predict 52 weeks into the future, we can see that 'yhat'

(Prophet's forecast) is the dominant feature among the four selected features with an importance score of 510 (Apppendix 2). The three other features utilised in the modelling are timestep, 104 week lag feature of the target variable, and quarter with importance scores of 414, 202, and 67. With the hybrid model trained on the dataset from Kaggle, Prohphet's predictions are again the most dominant feature like with the primary case. Prophet's predictions have an importance score of 518, while the two other features, 104 week lag feature of the target variable and the week number of the year have importance scores of 444 and 386 (Appendix 3).

7.2 Contributions

This thesis makes practical contributions to the fields of time series forecasting and machine learning driven demand forecasting by developing, implementing, and evaluating a hybrid forecasting pipeline combining Prophet and XGBoost for operational demand forecasting in the case company's real-world retail environment. The proposed hybrid modeling framework is a practical and replicable solution demonstrating the combination of a decomposable time series model and gradient boosting for demand forecasting tasks.

The most important contribution of the thesis is the forecasting artefact itself: a replicable hybrid pipeline that operationalises Prophet as both a baseline model and a feature generator for XGBoost. Beyond the case company, the artefact is potentially applicable to similar retail contexts where demand is seasonal, historical weekly sales data are available, and the organisation needs a lightweight forecasting tool rather than a large enterprise forecasting platform.

The proposed hybrid solution is evaluated against four points of comparison; two simple baseline models (seasonal naïve, single-season average), standalone Prophet, and standalone XGBoost. Quantitative evaluation of each individual model is accomplished by communication of the point-level accuracies, where the hybrid solution is proven to be the most effective approach with a retail time series containing strong seasonal

effects as demonstrated by the two examples of this study. The quantitative assessment is complemented by the qualitative assessment of the forecasting framework in the form of a passed weak market test.

The study demonstrates how Prophet can be used to model a baseline for comparing the performance of more complex models, while simultaneously being utilised as a technique for feature engineering in the machine learning forecasting pipeline. Among feature engineering with Prophet, the study demonstrates the generation of six other features for reformulating the time series forecasting objective into a supervised regression task suitable for XGBoost. The study demonstrates three levels of model optimisation in its cross-validation framework; Prophet hyperparameter tuning with randomised search, exhaustive XGBoost feature selection, and randomised XGBoost hyperparameter tuning.

Finally, it should be noted that the development and empirical testing of the construct in this study were exclusively executed on consumer-grade hardware. With this, the study demonstrates the operational feasibility of the proposed hybrid framework within an environment with resource constraints. By utilising computationally efficient open-source Python libraries the architecture of the framework eliminates the need for high levels of computing power and capital investments often associated with enterprise grade artificial intelligence.

The implementation also supports replicability as the implemented data transformations, feature sets, tuning grids, and evaluation metrics are described with detail in the thesis. The source code of the implemented forecasting framework is also made available in the appendix of this report so that the modelling process can be inspected and rerun.

7.3 Limitations of the Study and Directions for Future Research

While the results of the hybrid forecasting approach show superior results compared to the simple baselines and standalone statistical and machine learning approaches, it is

important to note that the context of this study is limited to two cases, one with the case company's real-world data and the other on a synthetic dataset from Kaggle. It is possible that the proposed hybrid forecasting approach could perform different in relation to the selected baselines on datasets with varying amounts of historical availability or patterns different to the ones that are covered in this study.

Regarding the technical scope, this study focuses on the integration of a decomposable statistical approach and gradient boosting. The performance of the combined model is compared to the performance of its individual parts. It does not compare the performance of the hybrid approach with other relevant methodologies such as deep learning architectures like LSTMs.

The study excludes the use of exogenous variables such as weather, economic indicators, or marketing efforst from both the Prophet & XGBoost approaches. Including these factors in the modelling process could lead to more accurate prediction results, but were intentionally left out to keep the configuration of the pipeline to a minimum.

While this study evaluates the model with historical sales data, the real-world deployment of the model would require continouous retraining with new data, data pipeline automation, orchestration and model drift monitoring. The deployment of the model in a production environment was not tested during the study.

This study applied the proposed construct to a univariate forecasting task: forecasting unit-level aggregated bicycle demand by combining all of the bicycle product categories into one time series. While this approach is useful for testing the forecasting approach with a dataset that has considerably less noise than a dataset with less product categories in the primary case of the study, it doesn't take into account the varying pricing and profit margins of different bicycles. Thus, it could be useful for a retail company to predict sales on a single product category or SKU level, but this was left out of scope for this study and could be explored further in later research.

Overall, the artefact is suitable as a decision-support tool for tactical demand forecasting, especially for planning aggregated bicycle demand across a seasonal sales cycle. It should not be interpreted as a fully production-ready forecasting system without further work on automated data ingestion, monitoring, retraining, exception handling, and stakeholder-facing reporting.

References

Albelali, S., & Ahmed, M. (2025). *Hidden Leaks in Time Series Forecasting: How Data Leakage Affects LSTM Evaluation Across Configurations and Validation Strategies* (arXiv:2512.06932). arXiv. https://doi.org/10.48550/arXiv.2512.06932

Arora, S., & Kolambe, M. (2024). Forecasting the Future: A Comprehensive Review of Time Series Prediction Techniques. *Journal of Electrical Systems*, *20*(2s), 575–586. https://doi.org/10.52783/jes.1478

Beck, N., Dovern, J., & Vogl, S. (2025). Mind the naive forecast! A rigorous evaluation of forecasting models for time series with low predictability. *Applied Intelligence*, *55*(6), 395. https://doi.org/10.1007/s10489-025-06268-w

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*(10), 281–305.

Cerqueira, V., Torgo, L., & Soares, C. (2019). *Machine Learning vs Statistical Methods for Time Series Forecasting: Size Matters* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1909.13316

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Cleveland, R. B., Cleveland, W. S., & Terpenning, I. (1990). STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics, 6*(1), 3–33.

Cortinhas, S. (n.d.). *Time Series Practice Dataset* [Data set]. Retrieved 25 March 2026, from https://www.kaggle.com/datasets/samuelcortinhas/time-series-practice-dataset

DMLC. (n.d.). *XGBoost Documentation*. Retrieved 28 December 2025, from https://xgboost.readthedocs.io/en

Fatima, S. S. W., & Rahimi, A. (2024). A Review of Time-Series Forecasting Algorithms for Industrial Manufacturing Systems. *Machines*, *12*(6), 380. https://doi.org/10.3390/machines12060380

Fildes, R., Ma, S., & Kolassa, S. (2022). Retail forecasting: Research and practice. *International Journal of Forecasting*, *38*(4), 1283–1318. https://doi.org/10.1016/j.ijforecast.2019.06.004

Harvey, N., & De Baets, S. (2025). Factors affecting preferences between judgmental and algorithmic forecasts: Feedback, guidance and labeling effects. *International Journal of Forecasting*, *41*(2), 532–553. https://doi.org/10.1016/j.ijforecast.2024.08.002

Houssein, E. H., Mohamed, M., Younis, E. M. G., & Mohamed, W. M. (2025). Artificial intelligence and classical statistical models for time series forecasting: A comprehensive review. *Journal of Big Data*, *12*(1), 271. https://doi.org/10.1186/s40537-025-01318-z

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice* (Third edition). OTexts. OTexts.com/fpp3

Hyndman, R. J., Athanasopoulos, G., Garza, A., Challu, C., Mergenthaler Canseco, M., & Olivares, K. G. (2025). *Forecasting: Principles and Practice, the Pythonic Way*. https://otexts.com/fpppy/

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Karmaker, S. (2025). *Feature Engineering in Time Series Forecasting*. Zenodo. https://doi.org/10.5281/ZENODO.17573456

Kim, J., Kim, H., Kim, H., Lee, D., & Yoon, S. (2025). A comprehensive survey of deep learning for time series forecasting: Architectural diversity and open challenges. *Artificial Intelligence Review*, *58*(7), 216. https://doi.org/10.1007/s10462-025-11223-9

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

Kourentzes, N., Trapero, J. R., & Barrow, D. K. (2020). Optimising forecasting models for inventory planning. *International Journal of Production Economics*, *225*, 107597. https://doi.org/10.1016/j.ijpe.2019.107597

Liu, Q., & Wen, H. (2025). XGBoost and Time series Model Fusion for Sales Trend Forecasting on E-commerce Platforms. *Proceedings of the 2025 International Conference on Digital Society and Intelligent Computing*, 184–189. https://doi.org/10.1145/3788910.3788937

Liu, Q., & Wu, Y. (2012). Supervised Learning. In N. M. Seel (Ed.), *Encyclopedia of the Sciences of Learning* (pp. 3243–3245). Springer US. https://doi.org/10.1007/978-1-4419-1428-6_451

Lukka, K. (2000). *The key issues of applying the constructive approach to field research*.

Lukka, K. (2001). *Konstruktiivinen tutkimusote*.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, *34*(4), 802–808. https://doi.org/10.1016/j.ijforecast.2018.06.001

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, *38*(4), 1346–1364. https://doi.org/10.1016/j.ijforecast.2021.11.013

Makridakis, S., Spiliotis, E., Hollyman, R., Petropoulos, F., Swanson, N., & Gaba, A. (2025). The M6 forecasting competition: Bridging the gap between forecasting and investment decisions. *International Journal of Forecasting*, *41*(4), 1315–1354. https://doi.org/10.1016/j.ijforecast.2024.11.002

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Rautiainen, A., Sippola, K., & Mättö, T. (2017). Perspectives on relevance: The relevance test in the constructive research approach. *Management Accounting Research*, *34*, 19–29. https://doi.org/10.1016/j.mar.2016.07.001

Rom, A. R. M., Ibrahim, S., Fadzil, A. F. A., Mangshor, N. N. A., & Ghani, N. A. M. (2025). A Review of Hyperparameter Tuning Methods in Machine Learning. *2025 6th International Conference on Artificial Intelligence and Data Sciences (AiDAS)*, 1–6. https://doi.org/10.1109/AiDAS67696.2025.11213530

Sarioguz, O., & Miser, E. (2024). Data-Driven Decision-Making: Transforming Management in The Information Age. *International Research Journal of Modernization in Engineering Technology and Science*. https://doi.org/10.56726/IRJMETS49577

Sina, L. B., Secco, C. A., Blazevic, M., & Nazemi, K. (2023). Hybrid Forecasting Methods—A Systematic Review. *Electronics*, *12*(9), 2019. https://doi.org/10.3390/electronics12092019

statsmodels. (n.d.). *Statsmodels.tsa.seasonal.STL - API documentation*. Retrieved 20 April 2026, from https://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.STL.html

Taylor, S. J., & Letham, B. (2018). Forecasting at Scale. *The American Statistician*, *72*(1), 37–45. https://doi.org/10.1080/00031305.2017.1380080

Taylor, S. J., & Letham, B. (2026). *Prophet source code* [Computer software]. Facebook. https://github.com/facebook/prophet/tree/main

Wang, B., & Zain, A. B. M. (2025). A Hybrid XGBoost-LSTM Framework for Supply Chain Demand Forecasting: Empirical Evidence from Retail Multi-Store Data. *Journal of Cultural Analysis and Social Change*, 4056–4073. https://doi.org/10.64753/jcasc.v10i4.3736

Wright, G., & Goodwin, P. (2009). Decision making and planning under low levels of predictability: Enhancing the scenario method. *International Journal of Forecasting*, *25*(4), 813–825. https://doi.org/10.1016/j.ijforecast.2009.05.019

XGBoost Contributors. (2026). *XGBoost source code* [Computer software]. DMLC. https://github.com/dmlc/xgboost/tree/master

Xu, N., Wang, Z., Dai, Y., Li, Q., Zhu, W., Wang, R., & Finkelman, R. B. (2023). Prediction of higher heating value of coal based on gradient boosting regression tree model. *International Journal of Coal Geology*, *274*, 104293. https://doi.org/10.1016/j.coal.2023.104293

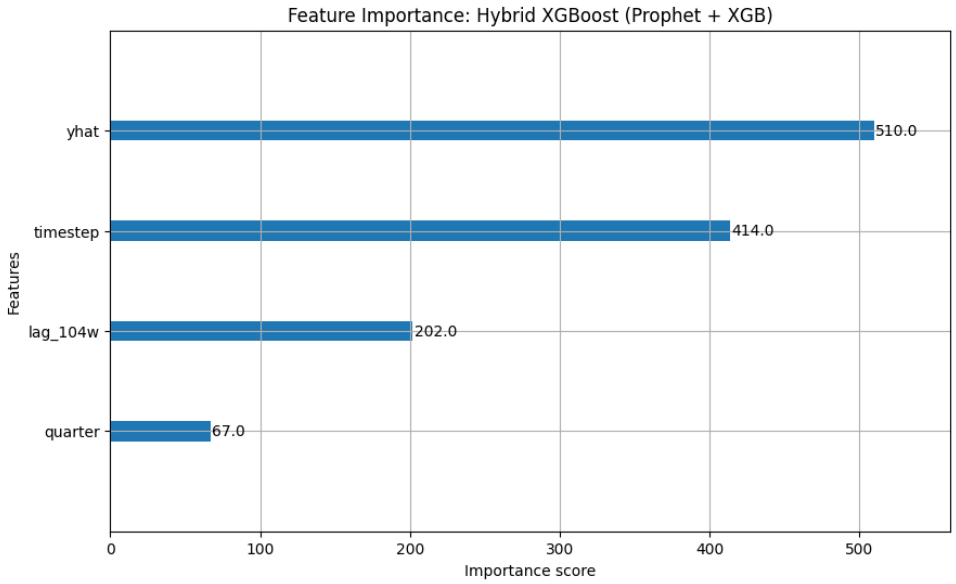
Zeng, S., Liu, C., Zhang, H., Zhang, B., & Zhao, Y. (2025). Short-Term Load Forecasting in Power Systems Based on the Prophet–BO–XGBoost Model. *Energies*, *18*(2), 227. https://doi.org/10.3390/en18020227

Appendices

Appendix 1. Source code of the proposed forecasting pipeline

https://github.com/ottokol/demand-forecasting-in-bicycle-retail/blob/main/forecasting.py

Appendix 2. Feature Importance: Hybrid XGBoost (primary case)



Appendix 3. Feature Importance: Hybrid XGBoost (Kaggle)