UPPSALA
UNIVERSITET

UPTEC STS 26029
Examensarbete 30 hp
Juni 2026

Forecasting Irregular Sales

A Machine Learning Approach to Predicting Customer Purchasing Behavior

Pontus Fredstam
Gustav Molin

Civilingenjörsprogrammet i system i teknik och samhälle

UPPSALA
UNIVERSITET

Forecasting Irregular Sales
A Machine Learning Approach to Predicting Customer Purchasing Behavior

Pontus Fredstam
Gustav Molin

Abstract

Sales forecasting is an important area for manufacturing companies. Whether it is accurate or not impacts the planning of production. This thesis is carried out in collaboration with Optinova and explores how a customer-level sales forecasting system can be created using machine learning. A recurring challenge in this work is the intermittent and lumpy characteristics of business-to-business (B2B) demand. Optinova's customers are no exception to this, with approximately 95% of the customers exhibiting intermittent or lumpy characteristics. Since standard forecasting approaches often assume continuous demand in the data, the zeroinflated data requires adapting these approaches.

To solve this issue, a two-stage hurdle model was constructed. In the first stage, a binary classifier predicts whether a customer will purchase or not in a given month. If the classifier predicts yes, a regression model then estimates the magnitude of that purchase. If the classifier predicts no, the revenue for that customer is set to zero. Two classifiers (XGBoost, LightGBM) and three regressors (XGBoost, LightGBM, Random Forest) created a total of six hurdle-pairs that were all trained and evaluated on the historical data. A feature engineering effort was performed to mimic demand planning concepts and customer behavioral metrics.

The results show that the best hurdle-pair (LightGBM classifier and XGBoost regressor) achieved a monthly mean absolute percentage error of 11.89%. When aggregated to a full year forecast, the best hurdle-pair achieved an absolute percentage error of 1.43%, as erratic monthly behavior is largely canceled out over a full year. The model systematically beats a naive baseline across all evaluation levels and captures seasonality trends in the historical data, as well as capturing over 80% of the variance in annual customer spend. The findings presented in this thesis suggest that machine learning can be used as a valuable decisionsupport tool for customer-level forecasting in B2B environments characterized by intermittent demand.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala

Handledare: Anindya Kimell Gupta Ämnesgranskare: Stefanos Kaxiras

Examinator: Elísabet Andrésdóttir

Populärvetenskaplig Sammanfattning

För tillverkande företag är det av betydelse att kunna förutse hur mycket kunder kommer att köpa under det kommande året. En bra prognos gör det möjligt att kunna planera produktionen, ha rätt lagernivå och fördela personal och resurser där det behövs. En dålig prognos kan istället leda till över- eller underproduktion, onödiga kostnader, försenade leveranser och i värsta fall förlorade kunder. Inom så kallad business-to-business försäljning, B2B, där företag säljer till andra företag snarare än till privatpersoner så har detta arbete länge utförts manuellt av säljare som uppskattar inköp utifrån sin erfarenhet av varje kund.

Denna studie har utförts i samarbete med Optinova, ett världsledande extruderingsföretag inom medicinska och industriella plastdetaljer. På Optinova skapas försäljningsprognoser idag antingen i samband med aktuell kund eller så uppskattar säljare kommande inköp genom sin kännedom om varje kund och historisk försäljningsdata, ett arbetssätt som bygger på gedigen branscherfarenhet. Detta arbete gavs möjligheten att undersöka huruvida maskininlärning kan fungera som ett komplement i detta prognostiseringsarbete. Maskininlärning är en form av artificiell intelligens (AI) där en datormodell själv lär sig mönster från data. Frågan som ställdes var alltså om det vore möjligt, utifrån Optinovas egen historiska data, att ta fram tillförlitliga prognoser 12 månader framåt per kund med hjälp av AI.

En central utmaning med att prognostisera kunders framtida inköp var det tydliga mönstret på hur regelbundet kunder köper i dessa sammanhang. Omkring 95% av företagets kunder har det som inom området kallas intermittent eller klumpig efterfrågan. Intermittent efterfrågan innebär att kunderna inte handlar varje månad, men att beloppen är relativt lika när de väl handlar. Klumpig efterfrågan innebär att kunderna också handlar sällan, men att beloppen dessutom varierar kraftigt mellan köpen. För majoriteten av månaderna har snittkunden ingen försäljning alls i datan. Detta är en utmaning för vanliga prognosmetoder som ofta antar ett jämnt köpmönster.

För att kunna hantera detta så utvecklades en modell uppdelad i två steg. Det första steget besvarar frågan: Kommer kunden köpa något under en viss månad, ja eller nej? Om svaret är ja går prognosen vidare till det andra steget som då uppskattar hur stort köpet förväntas bli. Om svaret är nej blir istället prognosen noll. Fördelen med en tvåstegsmodell är att de två frågorna kan hållas separerade och besvaras för sig själva. Modellen tränades på fem års historisk försäljningsdata. För varje kund beräknades en rad av beskrivande egenskaper, exempelvis hur ofta kunden brukar handla och det totala värdet kunden förväntas omsätta över tid. Modellen utvärderades sedan på data från år 2025, som hade hållits dold under hela utvecklingen för att resultaten inte skulle påverkas av att modellen sett svaren under träning.

Resultaten visar att modellen gav betydligt bättre prognoser än en enkel referensmetod, som bara antar att försäljningen skulle vara densamma som motsvarande månad föregående år. På årsbasis prognostiserades företagets totala intäkter med ett fel på endast 1.4% och på kundnivå kunde modellen förklara drygt 80% av variationen i hur mycket varje kund spenderar på ett år. Den genomförda analysen visar att tiden sedan senaste köp, hur regelbundet kunden brukar handla,

samt om det finns en inlagd order, är de starkaste signalerna för om en kund kommer köpa eller inte. För att bedöma hur stort köpet kommer bli är kundens långsiktiga värde viktigast.

Studien bör ses som en undersökning av vad som är möjligt snarare än ett färdigt system. Tanken är inte att AI ska ersätta befintliga processer eller specifik branschkunskap utan snarare kunna fungera som ett kompletterande beslutsstöd. Sammanfattningsvis visar studien att maskininlärning kan vara ett användbart verktyg även när datan är oregelbunden, och att en kombination av etablerade och existerande affärsbegrepp inom en organisation tillsammans med AI är ett lovande angreppssätt för försäljningsprognoser inom B2B.

Acknowledgments

We would like to express our gratitude to Optinova for giving us the opportunity to carry out this thesis and for providing us with the tools and domain knowledge needed. We are especially thankful to our supervisor, Anindya Kimell Gupta, for his guidance, support and continuous feedback that has been invaluable to the project.

We would also like to thank our subject reviewer, Stefanos Kaxiras, at the Department of Information Technology, Uppsala University, for the feedback provided along the way.

Pontus Fredstam & Gustav Molin

Uppsala, June 2026

Contents

1. Introduction

Manufacturing companies require accurate sales forecasting to manage production operations efficiently. In business-to-business (B2B) environments, sales forecasting has historically relied on human judgment, where sales representatives manually estimate expected revenue from customers based on experience or intuition [1]. This approach is inherently subjective, prone to cognitive bias and difficult to scale [2]. Forecasting solutions based on machine learning (ML) offer a data-driven perspective by using historical transactional patterns to produce predictions [1].

Although ML methods can often produce accurate results for sales forecasting at aggregate levels [1], individual customer forecasts are less explored in the literature [3]. In B2B environments, customer-level forecasting is both necessary and challenging. B2B revenue differs from regular consumer markets where the demand aggregates smoothly across large populations. Instead the demand is driven by a smaller number of customers each representing a distinct demand process. At the customer-level, purchasing behavior is usually more sporadic and irregular [4], which produces data dominated by zero activity, but with high variance in magnitude of purchases. This zero-inflated nature of the data poses a challenge for standard forecasting approaches that assume smooth, continuous demand [5].

This thesis addresses this challenge in the context of Optinova, a medium- to large-sized medtech manufacturing company, specializing in extrusion. At Optinova, the forecasting process spans approximately two weeks and occurs every quarter, each time forecasting the remainder of the fiscal year. Typically, in the absence of direct input from customers, the revenue forecasting process follows manual review of the historical transactional data. With a customer base dominated by intermittent and lumpy demand patterns, this process is both time consuming and difficult to scale. Forecasting over a 12-month horizon, individual customer demand uncertainty compounds significantly, making long-term forecasting considerably harder than short-term.

With an aim of identifying whether an ML based forecasting system can address these challenges, this thesis focuses on developing and evaluating a customer-level sales forecasting system at Optinova. The system is designed to produce monthly revenue forecasts across a 12-month horizon. Predictions are produced per individual customer, enabling both company-level aggregation and customer specific insights.

The thesis is structured in the following way: Chapter 2 provides the theoretical background necessary to understand the entire project. Chapter 3 describes the methodology used, following the Cross-Industry Standard Process for Machine Learning with Quality assurance (CRISP-ML(Q)) framework. Chapter 4 presents the results, followed by Chapter 5 where the results are discussed. Chapter 6 states the final conclusions of the thesis.

1.1 Research Questions

The following research questions guided the thesis:

1. How can a customer-level sales forecasting system be developed using machine learning for a medium- to large-sized medtech manufacturing company?
2. How accurately can machine learning be used to forecast customer purchasing behavior and sales revenue, and what drives variation in performance across forecast horizons?
3. What role do customer purchasing behavioral patterns play in the model's predictive performance, and what insights can be drawn from feature importance for customer understanding?

1.2 Delimitations

To stay within the scope of the thesis, several delimitations were necessary. The model requires historical transaction data to compute features, which means customers with no prior purchase history are excluded. As a consequence of this, the model's total revenue forecasts will not capture the full revenue of Optinova.

Performance is evaluated on a held-out historical test set rather than actual future predictions, as validating future results falls outside the time frame of the thesis. The deployment and monitoring of the model are similarly excluded, as the focus is on developing and evaluating the forecasting system rather than its implementation in production.

Finally, the model is trained exclusively on Optinova's data, meaning the findings may not generalize to other companies or industries. Forecasts are produced per customer and month, and do not differentiate between product types, due to Optinova producing customer specific products, which would create too sparse data for developing a reliable model.

2. Background

This chapter presents theoretical foundation for the thesis modeling choices by covering B2B sales forecasting and the characteristics of intermittent demand patterns, before describing underlying customer behavioral concepts and how they can motivate feature engineering. The chapter then presents why a two-part hurdle model can be motivated and which algorithms are relevant for each stage and how these components can be evaluated, before lastly describing how a model's predictions could be interpreted through SHAP (Shapley Additive exPlanations) feature importance analysis.

2.1 Business-to-Business Sales Forecasting

Sales forecasting in B2B environments has historically relied on subjective human judgment (qualitative methods) and basic time-series extrapolation. Conventional approaches, often embedded in Customer Relationship Management (CRM) systems, typically require sales personnel to manually assign probabilities to opportunities based on their intuition or experience [1]. However, this reliance on intuition is exposed to cognitive bias where sales representatives may intentionally underrate opportunities to avoid internal competition or overrate them to meet management pressure, often neglecting the complex dynamics of modern B2B transactions [1, 2].

Previous research states that the existing complexity of business dynamics has called for a move from subjective models to more systematic and data-driven reasoning [6]. Since the traditional forecasting systems are not fit for the big-data volumes or accuracy demands of today's market, the industry has shifted towards automated models based on historical data for objectivity and accuracy [7]. By accelerating the sales forecasting process and at the same time reducing its costs, the approach aims to increase overall sales revenue [8, 2].

When dealing with raw demand signals or specific product lines, B2B data is frequently characterized by intermittency and sparsity. Intermittent demand can be defined by sporadic occurrences with long runs of zeros [3]. When order sizes also vary, and not only when the order is occurring, the pattern is called "lumpy". The zero-inflated data creates impurity and high entropy in training data which can affect standard algorithm's performance negatively since they often assume smooth and continuous demand [3]. Short and sparse time series also demand models to generalize from a small amount of data. This can favor lower model complexity to avoid overfitting [9].

2.2 Customer Behavior Modeling

Customer lifetime value (CLV) is a measure of the estimated present value of the total future profits expected by an organization from a customer over the course of a certain amount of time [10]. CLV is most commonly used to identify profitable customers and develop strategies to target them [11].

A common framework for analyzing customer purchasing behavior is recency, frequency and monetary value (RFM). RFM describes three important aspects of customer behavior: recency

(the time since the last purchase), frequency (the number of purchases made within a given period), and monetary value (the amount spent during that period) [11, 12]. The framework is widely used for customer segmentation and provides important input to customer valuation models.

One challenge in customer behavior modeling is determining whether a customer remains active over time. This challenge differs between contractual and non-contractual business relationships. In contractual settings, customer attrition is directly observable through contract termination. In non-contractual settings, customers can stop making purchases without providing notice, making attrition difficult to observe directly. Consequently, the objective is often to estimate the probability that a customer is still active at a given point in time [13].

To estimate this probability, denoted as $P(alive)$, historical purchasing behavior can be analyzed through measures such as recency and frequency. One of the most widely used approaches is the Beta-Geometric/Negative Binomial Distribution (BG/NBD) model [13], which estimates the probability that a customer remains active based on their purchasing history. Estimates of $P(alive)$ are commonly incorporated into CLV calculations.

2.3 Two-Part Model

In B2B demand forecasting, fitting a single regression model is unreasonable, since it would systematically over-predict non-purchase months and under-predict big spenders. A two-part model addresses this by separating the zero and non-zero outcomes. It is an established statistical framework for imbalanced datasets with a high proportion of zero observations [14].

The two-part model breaks the problem down into two logical steps. First, it classifies the probability of an event occurring, expressed as $P(y > 0|\mathbf{x})$, where y is the outcome variable and x is the set of covariates. Second, conditional on an event occurring, the model uses a regression model to generate a prediction of magnitude of that outcome, expressed as $E(y|y > 0, \mathbf{x})$ [14]. Due to the separation of the two stages, the covariates do not necessarily need to be the same for the classification and the regression models. In practice, covariates often have different effects on the probability of an event compared to the magnitude of that event [15].

There are two primary ways to combine the two stages into a final prediction. The first alternative is to multiply the probability of the event occurring with the predicted magnitude, allowing the probability of the event occurring to act as a scaling factor that weighs the conditional outcome. The second approach is to use the classification stage as a binary gate, or "hurdle", that must be passed to pass onto the regressor. The outcome is only positive if the probability exceeds a specific threshold, otherwise the result is zero, regardless of what the regressor would have predicted [15].

Research shows that zeros in the data affect ML model learning and may lead to poor performance, and that a two-part ML approach yields the best results for lumpy, intermittent demand forecasting [5]. This shows that the two-part model is not limited to classical statistical models, as each stage can be replaced with any ML algorithm suited to the task.

2.4 Gradient-Boosting

Boosting algorithms represent a highly effective class of machine learning techniques designed to solve both classification and regression problems by combining many weak learners in sequence, where each addition improves on the shortcomings of the previous ensemble [16]. Instead of building independent models, gradient-boosting approaches the task by iteratively fitting new models to the negative gradients of previous models, essentially taking greedy steps in a gradient descent optimization [16, 7]. For each new iteration, a new tree is added to minimize the objective function. Equation 1 displays the objective function for gradient-boosted trees, which consists of two terms. First, the loss function ($l(y_i, \hat{y}_i)$), measures how well the current model fits training data. Second, $\Omega(f_k)$ penalizes the complexity of each tree f_k. This separation of loss and regularization makes gradient-boosting well suited for complex, non-linear problems while maintaining control over the bias and variance balance [8].

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{1}$$

Recent advancements in boosting have created specialized frameworks such as XGBoost (eXtreme Gradient-Boosting) and LightGBM (Light Gradient-Boosting Machine). These frameworks have specifically addressed the theoretical challenges of computational scalability and data sparsity [16]. Both models share the same gradient-boosting foundation but differ in how they are constructed.

XGBoost grows its trees level-wise first, meaning it finishes all nodes at the current depth before moving deeper, which produces balanced trees. Moreover, XGBoost extends the general objective function in Equation 1 by defining the regularization term explicitly as:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \tag{2}$$

where T is the number of leaves and w are the leaf output scores. The parameter γ is a hyper-parameter that controls the minimum loss reduction required to split a node. Higher γ produces simpler trees. λ applies a $L2$ penalty on leaf weights [16]. XGBoost also employs a sparsity-aware algorithm that automatically excludes missing values and zero-valued entries, which makes it advantageous for intermittent demand patterns [16].

LightGBM shares the same foundational concepts but rather than expanding all nodes at the current depth, it grows its trees leaf-wise. LightGBM always splits the leaf that allows for the biggest reduction in loss, regardless of depth [16]. This results in asymmetric trees that converge to lower loss faster than level-wise approach. The risk of this is overfitting on small datasets if the number of leaves is not controlled. The model complexity is therefore mainly constrained by the number of leaves.

2.5 Random Forest

Random Forest is an ensemble learning method that constructs a multitude of decision trees and outputs the average prediction for regression [8]. Unlike boosting, Random Forest builds all trees independently and in parallel, meaning no trees improve on the errors of the previous one. Random Forest relies on two primary sources of randomization to ensure robustness. Each tree trains on a random bootstrap sample of the original data, and at each node split, only a random subset of input variables is evaluated [16]. Since uncorrelated trees can protect each other from individual errors, this bagging approach effectively corrects the inherent habit of single decision trees to overfit to their training data [7, 8].

In the context of sales forecasting, Random Forest serves as an accurate baseline that handles non-linear relationships and high-dimensional categorical data exceptionally well. Ensemble methods like Random Forest can extract complex variable interactions and output feature importance metrics, often achieving superior recall and classification accuracy compared to simpler single-tree or probability models [8, 6]. However, Random Forest operates fundamentally as a "black box", meaning its internal decision making process lacks direct transparency. To satisfy business requirements for comprehensibility this opacity must often be resolved by pairing the algorithm with external explanation methodologies that decompose predictions into individual attribute contributions [6].

2.6 Model Evaluation

Regression and classification models are designed to solve fundamentally different prediction problems, and therefore require different evaluation metrics. This section presents the metrics most commonly used to evaluate the performance of classification and regression models.

2.6.1 Classification

Intermittent sales data poses one major challenge, being that the positive class (purchasers) constitutes only a small fraction of the dataset relative to the negative majority (non-purchasers). This complicates the evaluation of classifying purchasers from non-purchasers. Traditional classification evaluation measures, most notably accuracy, are imperfect for these class distributions because they fail to distinguish between error types and are easily skewed by the majority class [17]. In an imbalanced setting, accuracy is considered misleading [17].

Two commonly used metrics to evaluate classification models are precision and recall. Precision measures the proportion of predicted positive instances that are truly positive [17]. This reflects how often customers predicted to purchase actually made a purchase. Recall measures the proportion of true positive instances that the model correctly identifies [17], reflecting how efficient the model is at avoiding missing true purchasing customers.

The F_1-score functions as a summary of model performance by indicating the harmonic mean of precision and recall [17, 18]. Because the harmonic mean is sensitive to the lower of the two values (precision and recall), a high F_1-score serves as a guarantee that the model has achieved high score of both precision and recall. This prevents a misleadingly high evaluation of a model that excels

in one area while failing dramatically in the other [17].

Average precision (PR-AUC) further complements the F_1-score by summarizing the classifier's performance as the area under the precision-recall curve, where each point on the curve corresponds to a different threshold for classifying an instance as positive. This means PR-AUC is a threshold-independent metric. A higher PR-AUC indicates stronger overall discriminative ability. This makes PR-AUC particularly viable for skewed datasets, where model performance might not be clear from single-threshold metrics alone [17].

2.6.2 Regression

Four metrics are commonly used together to cover different dimensions of regression error: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and the Coefficient of Determination (R^2). The metrics in this section are defined based on a dataset of m observations, where y_i is the observed value, $\hat{y}_i$ is the predicted output, and $\bar{y}$ is the mean of the observed values. The most widely used measures are scale-dependent measures, such as MAE and RMSE [19]:

$$\text{MAE} = \frac{1}{m}\sum_{i=1}^{m}|\hat{y}_i - y_i| \tag{3}$$

$$\text{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2} \tag{4}$$

MAE averages the absolute errors and weights every miss equally, which makes it a strong candidate against right-skewed data. RMSE on the other hand squares the errors which means that large misses are punished harder. This can be useful when a few large revenue driving customers dominate the total error. A downside with both of these measures is that they are unbounded and carry units, which makes their raw value uninformative on their own. To interpret the value on its own, the target's range or scale would need to be provided.

To address the scale problem the error can instead be expressed in percentage terms with a metric such as MAPE [19]:

$$\text{MAPE} = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{5}$$

while MAPE is an intuitive and scale-independent metric, it is compromised by zero-value sensitivity, resulting in infinite or undefined values when the actual target values are zero [19].

R^2 complements these scale-dependent and percentage based metrics by providing a bounded measure [20]:

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{m}(\bar{y} - y_i)^2} \tag{6}$$

R^2 indicates the proportion of variance in the dependent variable explained by the model, and is uniquely informative because it generates a high score only if the model correctly predicts the majority of the ground truth elements relative to their distribution [21]. Furthermore, R^2 also signals poor predictions by producing negative values in cases where unbounded metrics might still produce seemingly acceptable scores [21].

2.7 Feature Importance

Complex machine learning models such as ensemble trees can often achieve high predictive performance, but the reason behind getting the results are often hard to understand, which creates interpretability challenges [22]. Shapley additive explanations (SHAP) is a unified framework designed to address this by mapping model outputs into contributions by features, which can be more easily interpreted [23].

The theoretical foundation of SHAP originates from game theory, by treating features as contributors to the prediction outcome. Features are attributed a value (SHAP value) that represent each feature's average marginal contribution across all possible combinations of features [23]. A key property of SHAP is that the final prediction for a specific instance can be expressed as the sum of each feature's individual contribution, directly increasing the interpretability of a model [22].

The results from SHAP can be visualized using a summary plot (often referred to as beeswarm-plot), which display each observation as a point, where the horizontal position in the plot reflects direction and magnitude of a feature's impact on the prediction (the SHAP value), while the color indicates if the feature value was high or low for that observation [23]. This allows both the global importance of a feature and its directional relationship with the outcome to be read from a single point.

3. Methodology

This thesis followed the Cross-Industry Standard Process model for the development of Machine Learning applications with Quality assurance methodology (CRISP-ML(Q)). The methodology is a modification to the Cross-Industry Standard Process model for Data Mining (CRISP-DM) framework, customized specifically to the lifecycle of machine learning applications and consists of six phases: Business & Data Understanding, Data Preparation, Modeling, Evaluation, Deployment, and Monitoring & Maintenance [24].

This approach was selected because it addresses the main drawbacks of regular data mining process models when applied to machine learning [24]. While CRISP-DM splits business and data understanding, CRISP-ML(Q) combines them, since data availability is crucial to assess the feasibility of a project. In addition to the other phases found in CRISP-DM, CRISP-ML(Q) introduces a dedicated Monitoring & Maintenance phase since ML models that imply real-time decisions face the risk of decline in performance over time. However, this is outside of the scope for this thesis and is therefore excluded from the methodology.

The methodology is structured around a two-part hurdle model, in which a classifier first predicts the occurrence of a purchase, and subsequently a regressor predicts the magnitude of the purchase. The motivation for the model choice is described further in Section 3.3.

3.1 Business & Data Understanding

The objective of this phase was to translate the business need into a formally specified machine learning task and to assess whether the available data was sufficient to support that task. Optinova operates in a non-contractual B2B environment, where a customer does not need to notify the company whether they will stop buying or not. This created a structural forecasting challenge. The absence of an order in a given period may have reflected either a natural purchasing cycle or early signs of customer churn.

The objective of the forecasting was specified as: given a customer's complete transaction history up to a snapshot date, predict that customer's revenue for each of the 12 subsequent calendar months. The 12-month horizon was chosen to align with the company's annual planning cycle. The output of the model was therefore a panel of predictions with one row per customer and forecast month. This output could be aggregated to produce a company-level revenue forecast for any sub-period within the horizons, such as monthly, quarterly or yearly.

This presented formulation departed from classical univariate time series forecasting in a fundamental way. Rather than modeling a single aggregate revenue series, the model operated at the individual customer-level, treating each customer as a distinct demand process.

A structured exploratory data analysis (EDA) was conducted before the data preparation to characterize the statistical properties of the data and to identify any transformations required before modeling. The EDA was organized around three analytical themes:

- Customer activity and lifetime patterns
- Demand intermittency characterization
- Seasonality and Lag Structure Analysis

The EDA results that motivate the model design choices are reported here and the modeling choices are in Section 3.3. The findings from each theme directly motivate specific design decisions in the data preparation and modeling phases.

3.1.1 Customer Activity and Lifetime Patterns

An analysis of customer activity density was done to gain insights into transaction frequencies and the number of occasional buyers. The analysis examined the total number of active purchasing months within the observation window, spanning from January 2020 to December 2025. The distribution of customer activity shown in Figure 3.1 revealed right-skew characteristic of B2B customer portfolios, in which a small number of long-tenured, high-frequency customers account for the majority of revenue. The median activity density for customers was five months of activity, which has important implications for feature engineering since lag features at horizons beyond the customer's observable history must be managed.

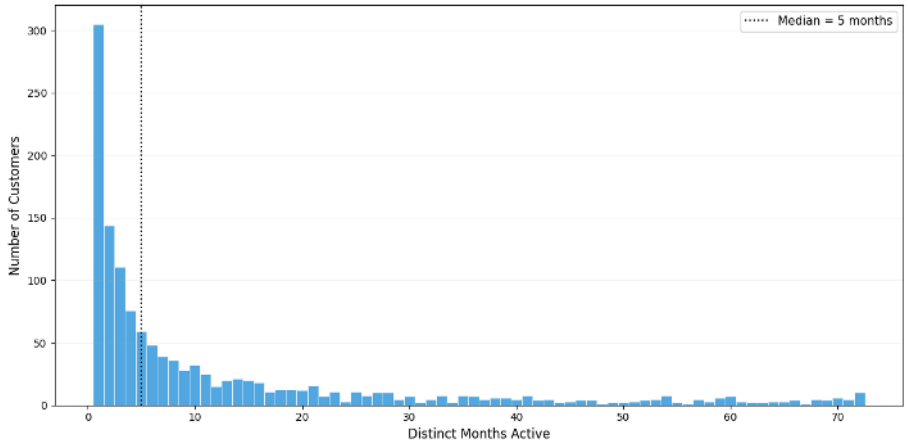

Figure 3.1: Distribution of customer activity density

3.1.2 Demand Intermittency Characterization

Given the non-contractual B2B setting, demand intermittency was expected to be present. To characterize this at a customer-level each customer's monthly revenue series was classified using the Syntetos-Boylan demand classification matrix [4].

The Syntetos-Boylan-Croston (SBC) framework classifies demand series along two dimensions: the average inter-demand interval (ADI), which measures the average number of periods between non-zero observations, and the squared coefficient of variation of demand sizes (CV^2) which measures variability in order magnitudes when demand does occur.

A demand series is classified as smooth when $ADI < 1.32$ and $CV^2 < 0.49$. The demand is considered erratic when $ADI < 1.32$ and $CV^2 \geq 0.49$. If $ADI \geq 1.32$ and $CV^2 < 0.49$, the

demand is categorized as intermittent. Finally, lumpy demand occurs when both $ADI \geq 1.32$ and $CV^2 \geq 0.49$ [4]. The SBC demand classification matrix can be seen in Figure 3.2.

$CV^2 \geq 0.49$	**Erratic**	**Lumpy**
$CV^2 < 0.49$	**Smooth**	**Intermittent**
	$ADI < 1.32$	$ADI \geq 1.32$

Figure 3.2: SBC demand classification matrix.

The results in Table 3.1 reveal an asymmetry between customer count and revenue contribution. Smooth customers with regular, predictable purchasing patterns covered only 2.8% of the customer base yet accounted for 40% of total revenue. By contrast, 94.6% of the customers fell into the intermittent or lumpy categories, collectively contributing around 38.6% of the total revenue. This pattern was consistent with the broader B2B industrial demand literature [3] and had implications for both modeling strategy and business interpretation. The high-revenue smooth segment could be well served by standard lag and rolling-mean features whereas the dominant intermittent and lumpy majority required different treatment.

Table 3.1: Summary of customer categories by revenue and demand patterns.

Category	**Customer %**	**Revenue Share %**	**Avg ADI**	**Avg CV^2**
Smooth	2.8	40	1.11	0.31
Lumpy	32.2	23	10.8	1.11
Intermittent	62.4	15.6	38.8	0.13
Erratic	2.6	21.4	1.16	1.02

This finding had three direct modeling implications. First, since standard accuracy metrics such as MAPE are analytically undefined when actual demand is zero, it was deemed unusable in this case since around 95% of the customers had zero-inflated transactional data. Second, the high occurrence of zero-valued observations indicated possible advantages of a two-part hurdle model architecture. Third, the ADI and CV^2 metrics derived during this analysis were included directly as customer-level features in the model, as they encode each customer's demand regularity in a compact and interpretable form that can be used as predictive features for the classification stage.

3.1.3 Seasonality and Lag Structure Analysis

Even though the primary forecasting target is per-customer revenue rather than aggregated company revenue, an analysis of the aggregate monthly revenue series was conducted to identify seasonal patterns and lag structures to inform feature engineering decisions for the ML models. The Autocorrelation Function (ACF) measures the correlation between a current value and each past lag, including indirect influence through intermediate lags. The Partial Autocorrelation Function (PACF) instead isolates the direct correlation at each lag by removing the intermediate dependencies.

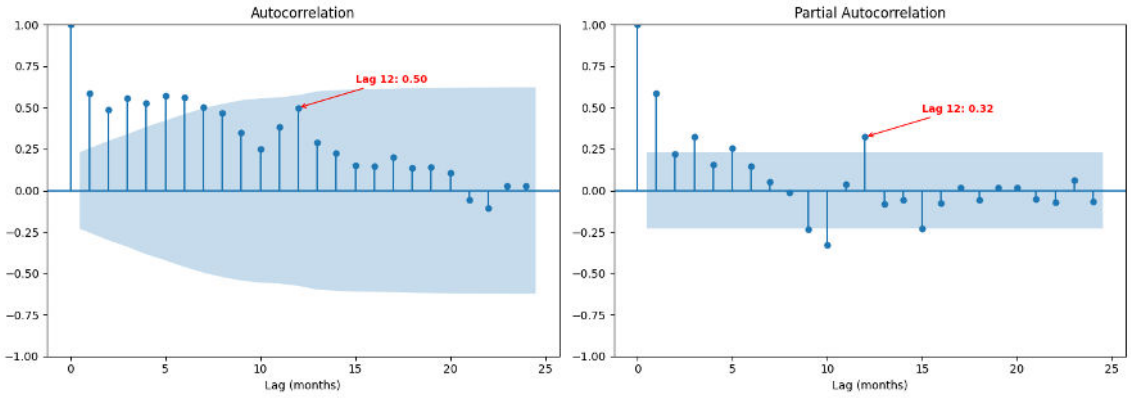

Figure 3.3: ACF (left) and PACF (right) of aggregate monthly revenue (lags 1-24).

The ACF exhibited a slow, persistent decay from 0.59 at lag 1 to 0.50 at lag 12, as shown in Figure 3.3. This indicates that there is a presence of a long-memory effect. The ACF and PACF showed significant spikes at lag 12. The PACF profile showed a dominant spike at lag 1 followed by gradual decay. These findings directly motivate the inclusion of lag-1, lag-3 and lag-12 revenue features as important components of the feature matrix.

A seasonal decomposition of the series was performed using the Seasonal and Trend decomposition using LOESS (STL) method with a period of 12 months to quantify the contributions of trend, seasonal, and residual components to total variance (see Figure 3.4). The trend component accounted for 57.4% of the total variance, indicating the need for features that capture the trend of customers. The seasonal component accounted for 26.4% of variance, aligning with the significant ACF spike at lag 12 and independently indicating the necessity of previous year spends, as well as calendar features. The remaining 16.2% was attributable to the irregular residual component. This analysis indicates that the systematic components, trend and seasonality, explain the majority of variation in aggregate demand, and are therefore necessary to capture in features.

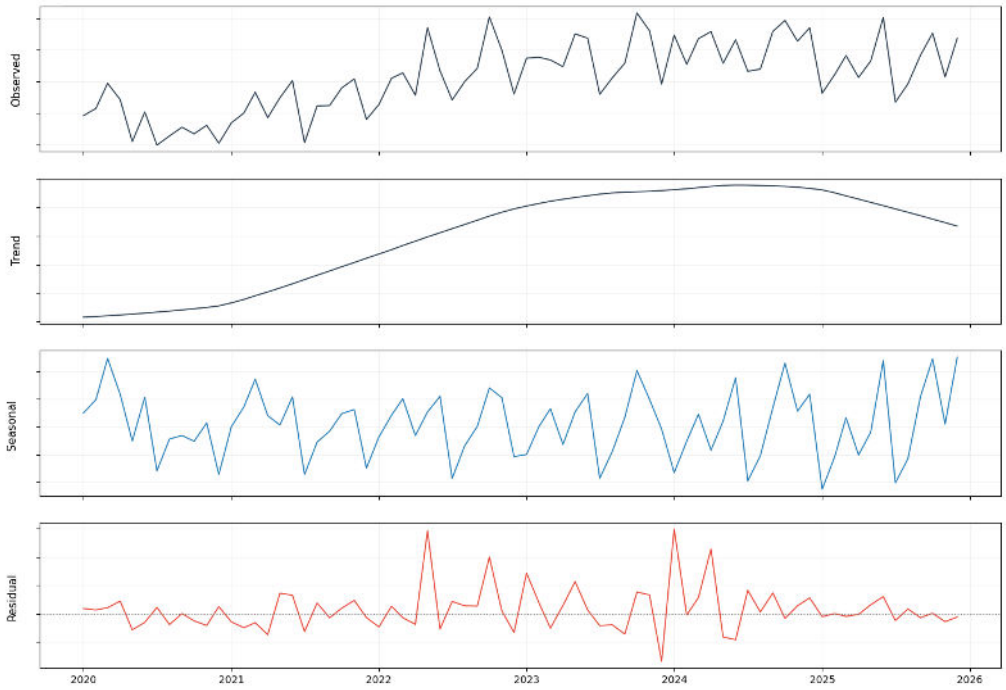

Figure 3.4: STL decomposition of revenue. Actual revenue magnitudes withheld for confidentiality.

3.2 Data Preparation

A large effort was spent on data preparation in this work. The preparation phase transformed the company's raw transactional data into a production-ready feature matrix that could be used for supervised ML. The data pipeline had five sequential stages that are visualized in Figure 3.5.

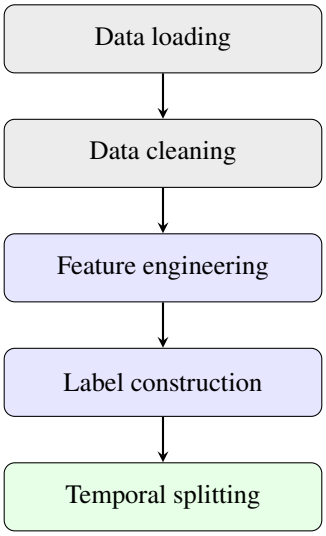

Figure 3.5: Data preparation pipeline overview.

3.2.1 Data Loading

Two datasets were retrieved from Optinova's data platform, consisting of data spanning January 2020 to January 2026:

1. **Sales data** – a transactional table consisting of all sales events at the line-item level, including order placement, delivery, and invoicing.
2. **Customer information** – customer attributes including sector classification (medical, industrial), geographic assignment, and organizational grouping.

No aggregation, cleaning, or feature creation was performed in this stage. The raw row counts are reported in Table 3.2. All necessary columns of the raw dataset can be seen in Appendix A.

Table 3.2: Row counts for the raw datasets (January 2020 - January 2026).

Dataset	**Rows**
Sales data	175,398
Customer information	8,979

3.2.2 Data Cleaning and Quality Assurance

A shared cleaning module served as the single transformation entry point for all datasets, both during training and at inference time. This architectural decision prevents training–serving skew, a common failure mode where transformations applied during training differ from those at inference [25].

From the two raw datasets, three cleaned datasets were derived through different filtering and aggregation strategies. The first two are both derived from the same sales data, but apply different business filters to extract complementary temporal perspectives:

1. **Order data** — the sales data filtered to external, invoiced sales, aggregated to one row per customer and order. This provides the historical signal: what revenue has been realized.
2. **Historical open orders** — the same sales data, but filtered to rows where the expected delivery date falls at least one calendar month after order placement. All revenue amounts were converted to EUR via monthly exchange rates. This provides the forward-looking signal: what demand was confirmed but not yet realized at each historical point in time.
3. **Customer information** — enriched with two binary flags `is_medical` and `is_industrial` and aggregated to one row per customer.

The reason for creating historical open orders is that in production, the forecasting system has access to a separate table of currently open orders, confirmed orders that have not yet been delivered. This table provides a strong forward-looking signal: if a customer has an open order for a given month, they are very likely to generate revenue for Optinova in that month. However, this production table only reflects the current state and no historical version exists. To give the model access to the same type of signal during training, the open order dataset was reconstructed to what it would have looked like at each historical point in time by filtering the sales data to orders that were confirmed

but pending delivery. This ensured the model learned from the same kind of information it would receive in production. Table 3.3 shows the row counts after cleaning. The reduction is primarily driven by the aggregation to order-level granularity and the exclusion of internal transactions.

Table 3.3: Cleaned datasets derived from the two source tables (January 2020 - January 2026).

Cleaned dataset	**Rows**
Order data	47,151
Open orders history	34,535
Customer information	3,021

3.2.3 Feature Engineering

The feature engineering was central to the performance of the hurdle model. Since the hurdle model consists of two different models that solve completely different problems, the features were designed separately for each stage. All features were derived from data exclusively before the snapshot date to prevent data leakage. This way, no information from the target could enter the features. The features were divided into static and dynamic features. Static features are identical over the forecasted months, meanwhile the dynamic features vary based on the month being forecasted. The resulting feature matrix consists of $C \times S \times H$ rows, where C is the number of customers, S is the number of snapshot dates, and H the number of horizons.

Separating static and dynamic features has both computational and conceptual advantages. By calculating static features once per snapshot and then reusing them across all forecast horizons extra computation is reduced by a factor of H horizons. The conceptual advantage is that a customer's identity can be understood by its historical identity such as purchase regularity, lifetime value and spending trajectory but also as the specific conditions of that customer connected to the forecasted month.

Classification features

The classification features were derived around three themes: purchase cadence, demand regularity and forward-looking signals. All features were developed to give insights into the core question of the classification: given a customer's historical behavior, how likely are they to purchase in a given month? All features used for the classification stage are presented in Table 3.4.

Purchase cadence tells the model how often a customer usually makes purchases. First, RFM features were derived using the RFM framework. The RFM library [26] also provided the tenure as the number of months since a customer made their first purchase. Furthermore, the total number of orders (`num_orders`) was included as a complementary intensity measure.

Purchase rate features were derived to quantify each customer's historical purchasing rhythm. The purchase rate over the entire lifetime of a customer was measured as the proportion of months with at least one order, while the 12-month rate captured the same proportion restricted to the prior 12 months. The ratio of these two rates formed a trend feature, where values above one indicate a growing purchase frequency and values below one indicate deceleration. Additionally, three

Table 3.4: Features used in the classification stage.

Feature	Type	Static/Dynamic
adi	Continuous	Static
cv2	Continuous	Static
idi_regularity	Continuous	Static
demand_density_recent	Continuous	Static
category	Categorical	Static
recency_since_last	Continuous	Static
frequency	Continuous	Static
num_orders	Continuous	Static
avg_frequency_per_period	Continuous	Static
purchase_rate_lifetime	Continuous	Static
purchase_rate_last_12m	Continuous	Static
purchase_rate_trend	Continuous	Static
replenishment_ratio	Continuous	Static
ordered_lag_{1,2,3,6,12}	Binary	Static
is_industrial	Binary	Static
is_medical	Binary	Static
has_open_order	Binary	Dynamic
bought_py	Binary	Dynamic
bought_2y_ago	Binary	Dynamic
purchase_rate_target_month	Continuous	Dynamic
times_target_month_occurred	Integer	Dynamic
target_month_vs_lifetime_rate	Continuous	Dynamic
replenishment_ratio_at_horizon	Continuous	Dynamic
is_{month} (12 indicators)	Binary	Dynamic
horizon_months	Integer	Dynamic

dynamic features were derived: the target-month purchase rate measures how often the customer has historically purchased in that specific calendar month, the number of times the target month has occurred during the customer's tenure, and the ratio of the target-month rate to the lifetime rate. Customers with fewer than six months of tenure received NaN for purchase rate features to avoid unrealistic estimates from short observation windows.

The replenishment ratio quantifies how overdue a customer is relative to their typical purchasing cadence, computed as the ratio of recency (months since last order) to the ADI. A value above 1.0 indicates the customer has waited longer than usual between purchases and below 1.0 indicates a purchase more recently than expected. A dynamic variant projects this ratio forward by adding the forecast horizon to recency, anticipating whether the customer will be overdue at the target month. This projected variant is capped at 5.0 to prevent extreme outliers from dominating tree splits for clearly churned customers.

Demand regularity describes the consistency of a customer's purchasing behavior over time. Four demand pattern metrics were derived per customer from the SBC framework: ADI, CV^2, inter-demand interval regularity and recent demand density. As established in Section 3.1.2, these metrics describe customer demand and are therefore expected to be important predictors for the classification stage due to the strong ADI to purchase probability relationship in intermittent demand settings. Furthermore, each customer was assigned to one of four categories based on their ADI and CV^2 (Smooth, Erratic, Intermittent, Lumpy), included as a one-hot encoded categorical feature.

Forward signals give the model direct evidence of probable activity in the target month. Primarily, a binary open-order flag, `has_open_order`, indicates whether a customer has committed to an

order for the target month. Binary open-order lag indicators at 1, 2, 3, 6, and 12 months prior to the snapshot encode recent backlog activity. Previous year purchasing was captured through binary indicators for the same calendar month one and two years prior. Calendar seasonality was encoded using 12 monthly indicator variables for the target month. The forecast horizon was included as a feature, allowing the model to calibrate for the natural sparsity of open-order backlog at distant horizons and the general growth of prediction uncertainty. Finally, since the purchasing patterns of Optinova's customers differ between the industrial and medical sectors, two binary features are included to encode whether a customer is industrial or medical. Since there were customers active in both sectors, the option of using only one binary feature was discarded.

Regressor features

The features used for the regressor were organized around four themes: Customer value, spend history, trend and momentum, and forward signals. Similarly to the classification stage, the regression features were constructed in order to answer the core question for a regression stage: given a customer's historical behavior, how much will they purchase for in a given month? All features used in the regression stage are presented in Table 3.5 and are described below.

Table 3.5: Features used in the regression stage.

Feature	Type	Static/Dynamic
adi	Continuous	Static
cv2	Continuous	Static
idi_regularity	Continuous	Static
demand_density_recent	Continuous	Static
category	Categorical	Static
clv	Continuous	Static
avg_monetary_per_order	Continuous	Static
clv_ratio	Continuous	Static
clv_ratio_log	Continuous	Static
clv_log_gradient	Continuous	Static
segment	Categorical	Static
revenue_lag_{1,2,3,12}	Continuous	Static
order_count_lag_{1,2,3,12}	Continuous	Static
ordered_lag_{1,2,3,12}	Binary	Static
revenue_rolling_mean_{3,6}	Continuous	Static
revenue_to_order_gap_lag_{1,2,6,12}	Continuous	Static
revenue_yoy_growth	Continuous	Static
spend_per_month_vs_mean	Continuous	Static
revenue_trend_3m	Continuous	Static
revenue_momentum	Continuous	Static
order_gap_trend	Continuous	Static
is_industrial	Binary	Static
is_medical	Binary	Static
ordered_amount_eur	Continuous	Dynamic
spend_py	Continuous	Dynamic
spend_2y_ago	Continuous	Dynamic
expected_spend_at_horizon	Continuous	Dynamic
spend_growth_for_horizon	Continuous	Dynamic
is_{month} (12 indicators)	Binary	Dynamic
horizon_months	Integer	Dynamic

Customer value features were used as an indicator of a customer's expected spending magnitude. The customer lifetime value (CLV) was calculated, using Equation 7, over a 12-month horizon.

$$CLV = \sum_{t=1}^{12} \bar{M} \times \bar{F} \times P(\text{alive}) \times \frac{1}{(1+i)^t} \tag{7}$$

In Equation 7, $\bar{M}$ denotes the average order value, $\bar{F}$ the average purchase frequency per month, $P(\text{alive})$ the BG/NBD probability that the customer is still active, and $i = 0.8\%$ is the monthly discount rate that weights future revenue less. Since CLV was calculated over a 12-month period, and i is set as a fixed constant, the discount can be considered a scalar that affects all customers equally. The discount was used as it is standard practice in CLV calculations [10]. The meaningful variation in CLV therefore comes exclusively from $\bar{M}$, $\bar{F}$, and $P(alive)$, making it sensitive to both spending magnitude and purchase frequency. Beyond CLV itself, two trajectory features derived from CLV were included. The CLV log gradient measures the long-term trajectory of the customer by comparing the current CLV against a prior CLV 12 months earlier, computed as $\ln(\text{CLV}_{\text{now}} + 1) - \ln(\text{CLV}_{\text{prior}} + 1)$, where positive values indicate growth and negative values indicate decline. In addition, the CLV ratio captures the short-term momentum by comparing the CLV derived from the most recent six months against the CLV from the preceding six months. The ratio was logarithmically transformed to ensure symmetric scaling around zero.

The segment feature was computed using K-means clustering with k=5 on three Z-score normalized features: purchasing recency, frequency and log-transformed CLV. Customers with fewer than two purchases were excluded from the clustering to ensure meaningful cluster separation. The cluster labels were assigned by ranking each cluster's median recency, frequency and CLV against decile boundaries, producing a combined score between 3 and 30 for each cluster, from which five segments were defined in falling order: Champions, High-value, Mid-tier, At-risk, Lost.

Spend history gives the model historical records of how much the customer has spent in recent months. Lag features were derived at fixed lookback windows of 1, 2, 3, 6, and 12 months prior to the snapshot date, capturing invoiced revenue and order counts at each window. Rolling means over three and six months provided smoothed baselines more robust to the level-shifting identified in Section 3.1.3. A revenue-to-order gap feature quantified unfulfilled demand at each window, providing a complementary possibility for a forward-looking signal to the open-order backlog.

Trend and momentum indicate whether a customer's spend is increasing or decreasing. Five relative growth and trend features were derived from the raw lag values. Year-over-year growth encoded long-term directional change. The spend per month over mean ratio compared the most recent month's revenue against the 12-month rolling mean, identifying customers whose latest activity deviates from their baseline. A 3-month revenue trend captured short-term slope. Revenue momentum distinguishes accelerating from decelerating customers. Finally, the order gap trend measured the directional change in unfulfilled demand. These features were left as NaN when input lags were unavailable, allowing tree-based models to learn optimal split directions for missing values.

Forward signals provide the regressor with concrete indications of future spend. The open orders available for the target month were included as a continuous EUR amount, indicating committed orders for the target month. Previous year spend for the same calendar month one and two years prior provided seasonal reference points. Two horizon-aware spend projections were derived: the expected spend at horizon and the year-over-year growth rate for the target month. These projections give the model a seasonal baseline and its recent trajectory for the specific month being forecasted. Finally, the same demand regularity metrics, calendar indicators, horizon and sector

features included in the classifier were also included in the regressor as they provide context on purchasing patterns relevant to the magnitude as well. The calendar months were important for catching seasonal magnitude differences between months.

3.2.4 Label Construction

For every row in the dataset two labels were created from the invoiced activity in the target month. This corresponded to the two stages of the two-part hurdle model.

First, the binary label y_{buy} was set to 1 if the customer's net invoiced revenue in the target month was above 0, otherwise the label was set to 0. This acts as the target label for the classification stage. The continuous label y_{spend} was set to the customer's total net invoiced revenue in the target month, and serves as the regression target label.

The demand intermittency characterization in the EDA established that the conditional revenue distribution, y_{spend} restricted to purchase months, is strongly right-skewed since 5.4% of customers produce 61.4% of the revenue. Both properties are theoretically consistent with a log-normal distribution, which would motivate a log transformation of the regression target prior to model training. This was evaluated but the log-transformed target produced larger aggregate percentage errors at the company-level. Because a small number of high-revenue customers contribute disproportionately to total revenue, the relative compression introduced by the log transform caused the model to systematically underestimate these customers' spend. Since accurate aggregate revenue forecasts are important for business operations, the raw y_{spend} was retained as the regression target.

3.2.5 Temporal Splitting

When creating the train, validation and test splits, all splits had to be strictly time-based in order to prevent data leakage. Random cross-validation is not applicable to this problem because it would mix future observations into training folds, constituting data leakage and producing optimistically biased performance estimates [27].

The data available spanned from January 2020 through December 2025. Training snapshots were set to begin in February 2021, leaving more than one year of prior history for feature computation (for example 12-month lags and CLV lookback). The training set used 34 monthly snapshots ranging from February 2021 through November 2023. Validation used a single snapshot at December 2023 with target dates consisting of the full year, January through December, of 2024. The test set used a single snapshot of December 2024 with targets of January through December 2025. The test set was held out and never used during model development or hyperparameter tuning and serves as the main point of evaluation. See Figure 3.6 for an overview of the train, validation and test splits.

The split boundary is defined on the target month of the forecast rather than the snapshot date. This distinction is important since a snapshot close to the validation boundary still contributes training rows for shorter horizons where the resulting target month does not cross the boundary. This maximizes the amount of training data without introducing leakage. Concretely, all customer-

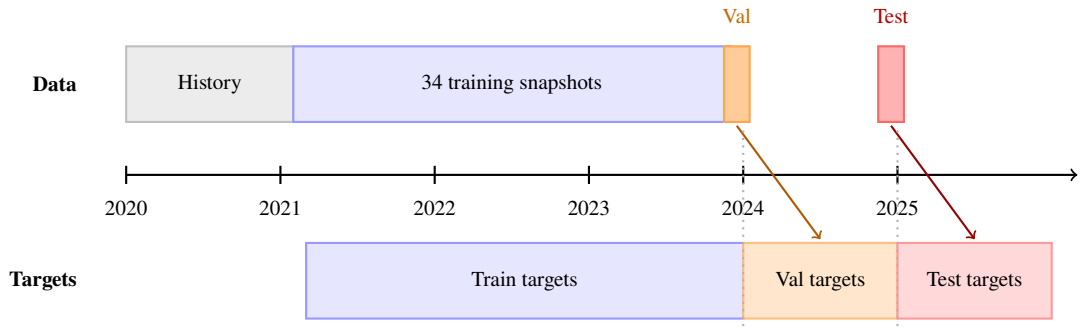

Figure 3.6: Temporal splits. Top row shows observation periods and snapshot cut-offs. The bottom row shows label windows. Arrows connect each snapshot to its 12-month target window. Dotted lines mark non-overlapping boundaries.

snapshot-horizon rows where the target month falls before January 2024 are assigned to the training set. This does, however, introduce more data consisting of lower horizon spans, meaning shorter horizons might produce more accurate results than long horizons as a result of this.

The validation and test sets each consist of a single snapshot with a full 12-month horizon lookahead. Using the full horizon for both ensures that hyperparameter tuning and final evaluation are performed under conditions identical to those of production inference, where a forecast for all months of the fiscal year is produced at once.

Walk-forward cross-validation, a common alternative for time series forecasting problems, was considered but not adopted. Given the 12-month forecast problem, each successive fold would need to advance the snapshot date by at least 12 months to avoid target month overlap between folds, leaving too few folds to be meaningful given the available observation window. Hence, walk-forward cross-validation was abandoned.

3.3 Modeling

The modeling phase describes how the model has been developed to solve the challenges presented in the thesis. It describes the hurdle model and how it was adopted, as well as which models were used and why.

3.3.1 Hurdle Model Architecture

The chosen modeling framework was structured as a two-part hurdle model, where a binary classifier first determines if a customer will generate any revenue in a target month or not. If the prediction is that a customer will generate revenue, the second stage is a regressor that predicts the revenue's amount [28]. If no purchase is predicted, the final forecasted revenue is zero and the regressor component will not be used.

This architecture is motivated by the demand intermittency findings in the EDA. With approximately 95% of customers classified as intermittent or lumpy, the data presented to any regression model is heavily zero-inflated. A single-stage regression model trained on this distribution would face problems. It would regress towards the training mean across a distribution dominated by zeros,

systematically over-predicting for customers who will not purchase. Further, it would receive a training signal diluted by the large numbers of uninformative zero-valued observations. The hurdle model was chosen to avoid these problems. The classifier is trained on the full dataset, including zeros, while the regressor only trains exclusively on positive-revenue rows, effectively operating on a conditional distribution that is free of zero inflation.

3.3.2 Purchase Classifier

The classifier uses two boosting classification models, XGBoost and LightGBM, trained to predict $y_{\text{buy}} \in \{0, 1\}$ for each customer-snapshot-horizon row. These models were selected because of how well they handle zero-dominated feature distributions [29]. Moreover, these models natively handle class imbalance without resampling, using the `scale_pos_weight` hyperparameter, which is important for the zero-inflated target.

The classifier outputs a probability of customer purchase for each row. To convert this into a binary purchase decision, a threshold of 0.5 was applied. Under symmetric misclassification costs, treating a false positive and a false negative as equally costly, 0.5 is a reasonable decision boundary. It is also a defensible threshold without an explicit asymmetric cost function, since any deviation from 0.5 implicitly assigns a higher cost to one error type, which requires domain specific justification. In the absence of a quantified cost structure, a threshold of 0.5 was adopted as the fixed decision rule rather than a tuned hyperparameter.

3.3.3 Revenue Regressors

Three regression models were evaluated as the regression component of the hurdle model. Each model was trained on the subset of training rows where $y_{\text{buy}} = 1$. All three were fitted to predict raw y_{spend} directly. Each regressor was evaluated separately, keeping all upstream components, such as the feature matrix, the classifiers and the train/val/test split, strictly identical across runs.

XGBoost is included as a gradient-boosted tree regressor. Its regularization terms ($L1$ and $L2$ penalties on leaf weights and tree depth) provide a direct control over the bias-variance trade-off on the sparse, right-skewed conditional distribution of y_{spend}. The sparsity-aware algorithm also handles any zero-filled features, introduced when a customer has no historical activity at a given lookback window, without requiring explicit imputation strategies.

LightGBM is included as a computationally efficient gradient-boosted alternative. Its leaf-wise tree growth and gradient-based one-sided sampling allow it to fit deeper trees on large feature matrices with lower memory overhead, which is relevant given the high-dimensionality of the feature matrix. Although LightGBM supports native handling of categorical features, this capability is not used in the thesis, in order to keep the feature representation identical across all three regressors.

Random Forest is included as a bagged-ensemble model. Unlike the two boosting methods, Random Forest builds trees independently on bootstrap samples rather than sequentially, making it less prone to overfitting on noisy observations in the conditional spend distribution. It also provides a natural reference point for assessing whether the sequential error-correction mechanism of gradient-boosting delivers measurable improvement in this setting. Because Random Forest

averages predictions across many trees, its output is bounded by the range of training values, a known limitation for growing customers whose future spend may exceed historical maximums, which is acknowledged as a constraint in the evaluation.

3.3.4 Hyperparameter Tuning

The hyperparameters for all three regressors and the classifier were tuned independently using Bayesian optimization, a more computationally efficient alternative to other available optimization methods [30]. The search uses the validation set, minimizing mean squared error (MSE). MSE was chosen over MAE since it penalizes large errors quadratically, which is appropriate given that high-revenue customers contribute disproportionately to total forecast error and represent the greatest business risk if systematically missed.

The hyperparameters of each model can be found in Appendix B. All models were tested on 150 trials each. All trial results, parameters, validation metrics and fitted model artifacts were logged, in order to track performance over multiple runs. The best performing configuration per model type is identified and used for the final evaluation on the held-out test set.

3.4 Evaluation

Model performance is evaluated for the classifier and regressor individually as well as the hurdle model in its entirety. The hurdle model is evaluated at three levels of aggregation to capture the forecast's full potential:

- Row-level predictions
- Monthly aggregate revenue
- Customer-level aggregate revenue over the full forecast horizon

As defined in Section 3.2.3, a row consists of one customer, snapshot date, and forecast-month horizon. Row-level predictions assess the model at its finest granularity and consist of one predicted revenue value versus the actual revenue value for each customer per forecast month. This is before any aggregation. The monthly and customer levels then aggregate the row-level predictions across all the customers within a month and across all twelve months for a given customer, respectively.

3.4.1 Naive Baseline

The model's performance was benchmarked against a naive seasonal baseline, in which each customer's predicted revenue for a given target month is set equal to that customer's invoiced revenue in the same calendar month one year prior. This baseline encodes the `spend_py` seasonal feature identified in the feature engineering section and represents a simple forecast which could be produced without a model. Customers with no transaction history in the corresponding prior-year month receive a baseline prediction of zero.

3.4.2 Classification

The purchase classifiers were evaluated independently of the regressor to isolate its contribution to the overall hurdle model performance. Since the positive class (purchase) is a minority in the training data, accuracy is not a suitable metric as it would be dominated by the majority class. Instead, the F_1-score is reported at the 0.5 decision threshold. Further, PR-AUC is additionally reported as a threshold-independent measure. False positive and false negative rates are reported explicitly to verify empirically that the symmetric misclassification cost assumption underlying the 0.5 threshold is reasonable in practice.

3.4.3 Regression

The three regression models, XGBoost, LightGBM and Random Forest, were evaluated in isolation on the subset of rows where $y_{\text{buy}} = 1$. MAE, RMSE and R^2 (conditional) are reported for each model on the validation and test sets. All upstream components are held constant across regressor comparisons, so performance differences are attributable exclusively to the choice of regressor.

3.4.4 Hurdle Model

Because the hurdle model incorporates both a classification and regression stage, the model's performance cannot be fully measured with one granularity of evaluation. From a company's perspective it is not only relevant how well the model predicts for each individual customer and month, but also how accurate the predictions are aggregated per month across all customers, or per customer across all months. These three perspectives motivate the structure used below.

Row-level

At the row level, three metrics were reported. MAE measured average prediction error in EUR and was robust to the right-skewed conditional spend distribution. Furthermore, RMSE was used because it penalizes large errors more heavily than MAE, which is appropriate given the high-revenue customers contribute disproportionately to total forecast error. MAPE was excluded from row-level evaluation because it is undefined when actual revenue is zero, which applies to the majority of customer-month observations.

Monthly-level

At the monthly aggregate level, aggregation ensures every month contains nonzero total revenue, making MAPE well-defined. Absolute percentage error (APE) was reported separately for each forecast horizon $h = 1, \ldots, 12$. The monthly MAPE was computed as the mean of these 12 APE values, capturing the average month by month accuracy. This allowed accuracy analysis over horizons as a comparison of different model configurations. This measure matters to confirm that the customer-level sales aggregates to an accurate monthly revenue forecast. Moreover, an annual aggregate APE was reported computed on the sum of all 12 months. This aggregate differed from the reported MAPE since it measured the error on the total revenue of the full year. Both metrics were reported since they answer different business questions.

Customer-level

At the customer aggregate level, MAE, RMSE and R^2 were reported. These metrics were applied to each customer's total predicted versus actual revenue summed over the full forecast horizon. Since zero total revenue over a 12-month horizon is a meaningful outcome, indicating a churned customer, metrics such as MAPE were excluded at this level, since MAPE is undefined for customers with zero actual spend. MAE captures the average absolute miss in EUR per customer, RMSE penalizes large deviations more heavily, which is appropriate given that errors on high-revenue customers carry disproportionate business consequences, and R^2 measures how well the model explains variance in customer-level spend relative to simply predicting the mean. A negative R^2 at this level would indicate that the model performs worse than a mean prediction, which serves as a meaningful lower bound check.

3.4.5 Feature Importance

Feature importance was computed using SHAP for the classification and regression pair with the best row-level MAE on the test-set. The MAE at row-level was used since it directly reflects the prediction error at the granularity the model was trained on. Including all configurations would provide little additional insights, since the top features are a property of the data rather than specific models. SHAP values were visualized using beeswarm-plots to analyze feature contributions on both stages of the hurdle model.

4. Results

In this chapter the different models performances and their driving features are presented. The evaluation of these results was performed on three different aggregation levels using a held-out test set with approximately 15,000 customer-month observations. This dataset originated from a snapshot date of December 2024 with its targets extending the full forecast window of 12 months to December 2025. All the metrics presented in this chapter are from this held-out set to make sure that there is no biased evaluation of the model's performance.

4.1 Classification

The binary classification stage of the hurdle model estimated the probability that a customer would make a purchase and hence generate revenue in the target month. For evaluating the classifier both XGBoost and LightGBM classifier configurations were used. The naive baseline was used, predicting the same outcome as previous year, to contextualize the performance of the classifiers.

Table 4.1: Classification performance (14,964 total observations).

Model	**Partition**	**Precision**	**Recall**	F_1**-score**	**PR-AUC**
XGBoost	Validation	**0.696**	**0.641**	**0.667**	0.740
XGBoost	Test	0.647	**0.655**	0.651	0.719
LightGBM	Validation	0.692	0.636	0.663	**0.741**
LightGBM	Test	**0.666**	0.638	**0.652**	**0.722**
Baseline	Validation	0.494	0.577	0.532	0.357
Baseline	Test	0.491	0.587	0.534	0.344

Table 4.1 shows how both learned classifiers outperform the naive baseline across all metrics. LightGBM achieved a slightly higher precision score than XGBoost on the test set, both with minimal precision loss on the test partition compared to the validation partition but with XGBoost scoring the highest. Both models also displayed false negative rates of 34–36%, which can be seen in Table 4.2. This is most likely a consequence of class imbalance where 13.5% (2,018 out of 14,964) of observations involved purchases. This class imbalance in the classifier is present downstream in the final forecast and establishes a performance ceiling in the pipeline on the hurdle model. Both models achieved PR-AUC scores greater than 0.74 in the validation set and dropped by approximately 2 percentage points for the test set. This modest degradation on all metrics indicates a stable generalization and performance across time periods. Given a random classifier would achieve a PR-AUC, equal to the positive-class prevalence (0.135), and that the naive baseline reaches only 0.344, the trained classifier's scores are a significant improvement over the naive baseline.

Table 4.2: Confusion matrices (test partition, 14,964 total observations).

Model	True Negatives	False Positives	False Negatives	True Positives
XGBoost	12,224	722	**696**	**1,322**
LightGBM	**12,301**	**645**	731	1,287
Baseline	11,716	1,230	834	1,184

4.2 Regression

The regression performance estimated the revenue conditional on a purchase occurring. It was evaluated exclusively on the rows where $y_{buy} = 1$, isolating the performance of the regressor from any misclassifications introduced by the classification stage. As a result, the regressor operates on the conditional distribution of positive revenue only, free from the zero-inflation discussed in Section 3.1.2. The naive baseline predicts each buyer's spend as their revenue in the same calendar month one year prior (`spend_py`).

Table 4.3: Regressor performance on the test-set with positive cases only (n = 2,018 buyers).

Regressor	MAE	RMSE	R^2
XGBoost	**14,628**	**35,696**	**0.578**
LightGBM	14,761	35,949	0.572
Random Forest	15,661	36,705	0.553
Baseline	19,382	45,931	0.300

The XGBoost regressor showed the best performance on the held-out test set among the three models with the lowest MAE of 14,628 EUR and R^2 of 0.578 explaining 57.8% of the conditional spend variance (see Table 4.3). LightGBM achieved almost identical results but slightly worse. The RMSE to MAE ratio of approximately 2.4 observed for all models reflected the wider range of revenue spend in the conditional distribution. The Random Forest regressor underperformed by $\sim$ 7% in comparison to the boosting alternatives. All three regressors substantially outperformed the naive baseline, which achieved an MAE of 19,382 EUR and R^2 of 0.300, meaning it explains about half of the variance that the XGBoost regressor captures. This result confirms that gradient-boosted regressors learn meaningful patterns beyond what Random Forest or a naive baseline can capture.

4.3 Hurdle Model

The hurdle model was evaluated as a classification and regression stage, combined in a pipeline. The outcome of the binary classification and the regression was evaluated against the held-out test set. In cases where only one hurdle model configuration is provided, LightGBM and XGBoost are used. As mentioned in Section 3.4, the results are provided on three levels of aggregation: row-level, monthly-level and customer-level.

4.3.1 Row-level

The complete hurdle model combined the classification and regression stage where the target variable being predicted was the result of the regression estimate where $P(buy) \geq 0.5$. Row-level evaluation reported metrics across all customer-month observations in the held-out test set, without any aggregations of prediction-actual pairs.

Table 4.4: Hurdle model performance on test data (all observations).

Configuration		MAE	RMSE	R^2
Classifier	**Regressor**			
LightGBM	XGBoost	**2,783**	**14,443**	**0.571**
LightGBM	LightGBM	2,802	14,529	0.566
XGBoost	XGBoost	2,828	14,475	0.569
XGBoost	LightGBM	2,842	14,545	0.565
LightGBM	Random Forest	2,966	15,067	0.533
XGBoost	Random Forest	3,017	15,112	0.530
Baseline		3,647	20,746	0.115

The best hurdle-pair on the held-out test set consisted of a LightGBM classifier and XGBoost regressor and achieved the lowest MAE value with a 23.7% improvement over the naive baseline, which is shown as a comparison point in Table 4.4. All evaluated hurdle-pairs outperformed the baseline across all metrics. The small gap between the two top hurdle-pair configurations indicates that classifier performance might drive variation more than regressor selection. This is consistent with the classifier's 34-36% false negative rate acting as a bottleneck on the captured revenue. The Random Forest regressor underperformed compared to the gradient-boosting alternatives and had the worst hurdle-pair result, but still exceeding the baseline.

The best R^2 score of 0.571 indicates that the model explained 57.1% of variance across all rows. This metric includes both the large number of actual zero rows and zero-predicted rows of customers who do not purchase in a given month. These are difficult to predict perfectly because of the uncertainty and sparsity of the transactional data. Rows where the classifier predicted $y_{\text{buy}} = 0$ resulted in a predicted revenue of zero EUR and only when $P(buy) \geq 0.5$ does the regressor contribute.

Row-level MAE and RMSE by forecast horizon for the best hurdle-pair (LightGBM classifier and XGBoost regressor) based on the held-out test set are presented in Table 4.5. MAE ranged from 2,111 EUR (January) to 3,215 EUR (June), with the early horizons showing lower error compared to mid-year.

Table 4.5: Performance metrics: MAE and RMSE (EUR) by forecast horizon for LightGBM classifier and XGBoost regressor.

Horizon	Month	MAE	RMSE
1	January	2,111	9,780
2	February	2,566	12,039
3	March	2,709	12,583
4	April	2,235	9,392
5	May	2,892	12,179
6	June	3,215	21,628
7	July	2,774	16,273
8	August	2,702	14,740
9	September	3,095	14,507
10	October	3,192	15,562
11	November	2,921	13,973
12	December	2,984	16,427
Overall	**Annual**	**2,783**	**14,443**

4.3.2 Monthly-level

By aggregating all customer-level predictions for each forecasted month, a full year forecast ranging from January to December 2025 could be created. These results were then compared against the actual monthly revenue totals and the naive baseline. This monthly-level evaluation directly answered how accurately the model could forecast company revenue for each month.

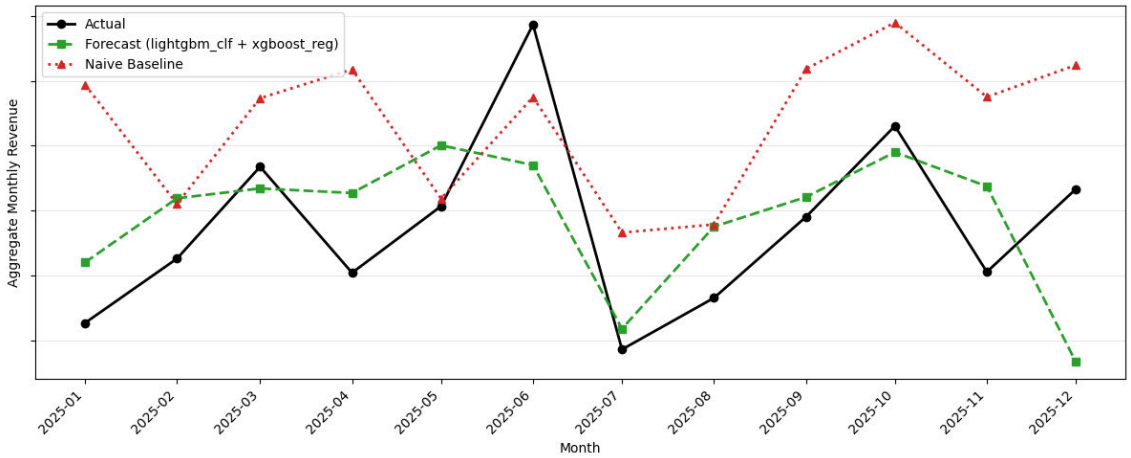

Figure 4.1: Monthly forecast comparison showing one model (LightGBM classifier and XGBoost regressor) compared to the naive baseline and the actual values for the test-set, Jan-Dec 2025. Actual revenue values withheld for confidentiality.

The model's monthly-level forecasts displayed as expected different behavioral patterns compared to the naive baseline. The boosting algorithms achieved smoother tracking with less extreme swings in comparison to the baseline's more erratic characteristics. The model differed the most compared to the baseline in January, April, July, September and November (see Figure 4.1) where the model captured structural patterns such as forward-looking signals and trend that were invisible to the naive baseline.

Table 4.6: Monthly-level evaluation of MAPE by configuration.

Configuration		Monthly MAPE
Classifier	**Regressor**	
LightGBM	XGBoost	**11.89%**
LightGBM	LightGBM	12.31%
XGBoost	XGBoost	13.64%
XGBoost	LightGBM	14.16%
LightGBM	Random Forest	18.84%
XGBoost	Random Forest	20.97%
Baseline		21.44%

Table 4.6 shows the comparison of monthly MAPE across the different model configurations. The hurdle combinations containing gradient-boosting regressors achieved the lowest MAPE between 11.89-14.16% and clearly beat the naive baseline while Random Forest regressors achieved between 18.84-20.97% and only beat the baseline by approximately 0.5 percentage points.

Table 4.7 presents the monthly-level forecasted performance across the horizons for the best model configuration (LightGBM classifier and XGBoost regressor) with regard to monthly absolute percent error (APE) on the held-out test set, also compared against the baseline for perspective. The naive baseline's APE revealed a large variance in performance. Some months such as May: 1.29%, March: 10.92% are naturally easier to predict from historical averages while others: (January: 50.49%, April: 38.96%) proved more difficult. The model's APE shows that the hurdle model achieved a better and more consistent score in most horizons but underperformed the baseline in four out of 12 months.

Table 4.7: Performance comparison: Baseline APE vs. Model APE by forecast horizon.

H	Month	Baseline APE	Model APE
1	January	50.49%	12.83%
2	February	10.18%	11.24%
3	March	10.92%	3.45%
4	April	38.96%	15.28%
5	May	1.29%	10.35%
6	June	9.41%	18.24%
7	July	26.23%	4.64%
8	August	14.78%	14.35%
9	September	25.59%	3.44%
10	October	15.47%	3.91%
11	November	33.45%	16.36%
12	December	20.47%	28.53%
Overall	**Average**	**21.44%**	**11.89%**

The earliest horizons predicted by the model showed the strongest model advantage. Horizon 1 achieved an APE score of 12.83% compared to the baseline's 50.49% due to its massive overprediction. Horizon 3 and 4 also showed similar results, while February represented an exception

where the model's 11.24% APE score slightly underperformed the baseline's 10.18%. In general, the model slightly overpredicted the first quarter, but was still superior to the baseline. Horizon 8 showed the smallest advantage where both model and baseline struggled with seasonal demand. May and June represented a structural weakness where the model underperformed by a wide margin, which suggests that these months might have deviated from the feature distributions observed in the training data.

Although the model's recovery appeared towards the end of the year after the clear seasonal decline of horizon 7 (July), horizon 12 failed to capture the actual revenue trend. This horizon represents the model's worst performance with 28.53% APE compared to the baseline's 20.47%, which is also clearly seen in Figure 4.1.

On the full aggregated 12-month forecast period, the LightGBM classifier and XGBoost regressor achieved an annual aggregate APE of 1.43%, which is drastically lower compared to the model's monthly MAPE of 11.89%. The result proved a strong annual accuracy regardless of erratic monthly-level performance. This annual accuracy followed the cancellation of directional errors where model's monthly under and over-estimates offset when summarized.

4.3.3 Customer-level

The evaluation on customer-level summarized customer's predicted and actual revenue across all 12 forecasted months. Three evaluation metrics per customer in the test set was produced.

Table 4.8 presents performance on customer-level. R^2 at a customer-level rose dramatically from 0.571 at row-level to 0.807 at customer-level. This reflected the aggregation's property of summarizing monthly predictions and actuals across the 12-month forecast window, where individual month errors offset one another to reveal true annual customer value.

Table 4.8: Model performance for customer-level.

Configuration		**Customer MAE**	**Customer RMSE**	**Customer R^2**
Classifier	**Regressor**			
LightGBM	XGBoost	22,341	**98,619**	**0.807**
XGBoost	XGBoost	23,063	99,592	0.803
XGBoost	LightGBM	23,248	100,040	0.801
LightGBM	LightGBM	23,526	99,468	0.799
LightGBM	Random Forest	24,258	108,741	0.765
XGBoost	Random Forest	25,053	109,766	0.761
Baseline		**21,992**	122,094	0.704

The customer-level MAE in Table 4.8 shows the average yearly forecast error per customer. The top four configuration-pairs performed similarly and differed by a MAE of 1,185 EUR. Similarly, the variation of R^2 was minimal. The top two configurations with regard to MAE and R^2 used the XGBoost regressor meanwhile the Random Forest configurations performed the worst across all metrics. Classification selection showed minimal impact on the tested pairs.

The baseline achieved a lower MAE than all configurations, while at the same time performing

considerably worse on all other metrics than the model. The R^2 score of 0.704 compared to the boosting algorithm's score of 0.807 as well as the baseline's high RMSE score of 122,094 indicates that the ML-model produced more stable predictions with fewer large deviations. The baseline's lower MAE but higher RMSE and R^2 is therefore indicative of systematically failing on high-revenue customers, which are important for the business.

4.4 Feature Importance

For the best performing model configuration on the held-out test set SHAP-based features importance were computed for both the classifier and regressor component. These results are derived from LightGBM classifier paired with a XGBoost regressor and offer insights for the model's predictive performance as well as business understanding.

4.4.1 Classification

The top 10 features used by the classification stage, ordered by SHAP value, are shown in Table 4.9. Recency and demand intermittency metrics, which are represented as the three highest ranked features, together accounted for 45.3% of classifier influence. At rank six the binary feature `has_open_order` demonstrated that an existing order status provided purchase signals beyond historical patterns. The industrial and medical binary features also showed importance in whether a customer would buy or not, indicating different behavioral patterns of the two segments.

Table 4.9: Classifier top 10 features by contribution.

Rank	**Feature**	**Mean** $\lvert SHAP \rvert$
1	recency_since_last	0.178
2	demand_density_recent	0.160
3	adi	0.115
4	avg_frequency_per_period	0.101
5	purchase_rate_lifetime	0.081
6	has_open_order	0.062
7	purchase_rate_last_12m	0.042
8	is_industrial	0.042
9	is_medical	0.034
10	purchase_rate_trend	0.029

The classifier beeswarm-plot (Figure 4.2) revealed clear behavioral patterns. For the top feature, `recency_since_last`, there was a clear directional trend. Blue points (indicating a low number of months since last purchase) strongly cluster on the right with positive SHAP values. Meanwhile, red points (indicating a long time since last purchase) cluster on the left, indicating negative SHAP values. This means that having a recent purchase makes the model significantly more likely to predict another purchase in the future.

Because of how the feature values were defined, these directional trends differed. The second most influential feature, `demand_density_recent`, shows the opposite color gradient with high demand being positive and low demand being negative for predicting a purchase. The

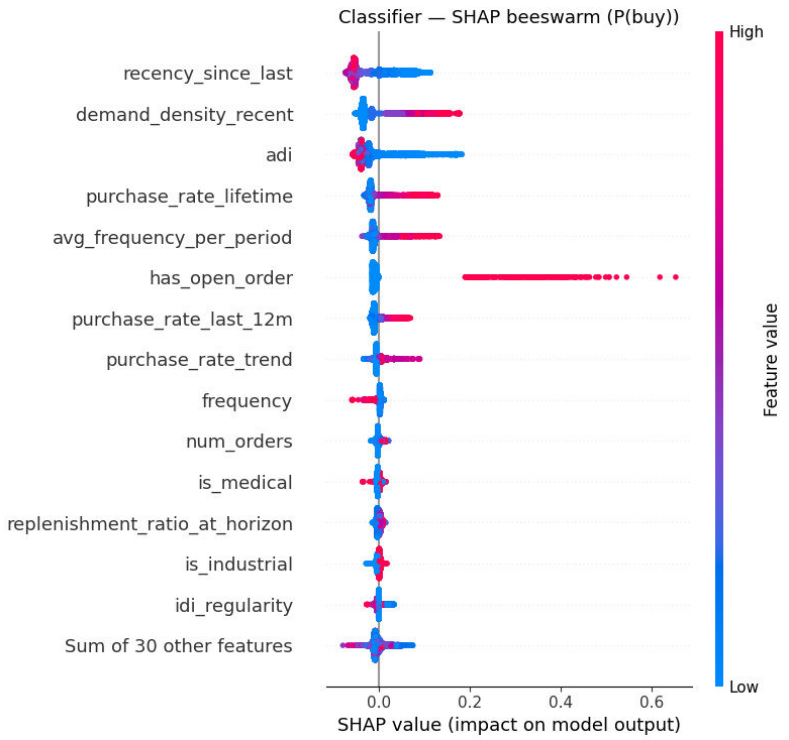

Figure 4.2: Classifier feature importance (SHAP beeswarm, P(buy))

`has_open_order` feature had the most notable pattern with a bimodal distribution. Since it was a binary feature the blue values represented no open order and concentrated near zero SHAP impact. Red values which represented having an open order spread far towards the right reaching SHAP values that exceeded 0.6. This distinct pattern indicated that having an active order was an extreme positive signal for predicting a purchase. The remaining features showed moderate spread.

4.4.2 Regressor

For the regression stage the feature importance differed from the classification stage. The customer characteristics, CLV, rolling mean over six months and the average monetary per order dominated the feature importance and accounted for approximately 50% of influence, as seen in Table 4.10. Similarly to the classifier, incoming orders and their amount `ordered_amount_eur`, are important flags for the model as well as industrial or medical market segment.

The regressor beeswarm-plot (Figure 4.3) displayed patterns in predicting the magnitude of the revenue. CLV showed a strong positive relationship with high values concentrated above SHAP values of 0.5. The `revenue_rolling_mean_6` feature displayed a bimodal distribution with low feature value clustering near zero and red in positive SHAP values. The `ordered_amount_eur` feature displayed proportional relationship with low feature value at negative SHAP values and high feature value at positive SHAP values.

The industrial and medical sector features showed a more clear effect on the regressor than the classifier, and had opposing effects on each other. If a customer was in the industrial segment the feature's values concentrated at negative SHAP while if the other way around it concentrated at positive SHAP. The `is_december` feature showed bimodal pattern when true (red), with clear

Table 4.10: Regressor top 10 features by contribution.

Rank	**Feature**	**Mean** $\lvert SHAP \rvert$
1	clv	0.253
2	revenue_rolling_mean_6	0.164
3	avg_monetary_per_order	0.098
4	revenue_rolling_mean_3	0.082
5	ordered_amount_eur	0.038
6	expected_spend_at_horizon	0.031
7	is_medical	0.029
8	is_industrial	0.028
9	horizon_months	0.026
10	adi	0.022

negative on the model output. While the feature was false (that is, any other month than December) the plot showed a slightly positive output in the model. Remaining features contributed modest spread around zero but leaning towards negative SHAP values.

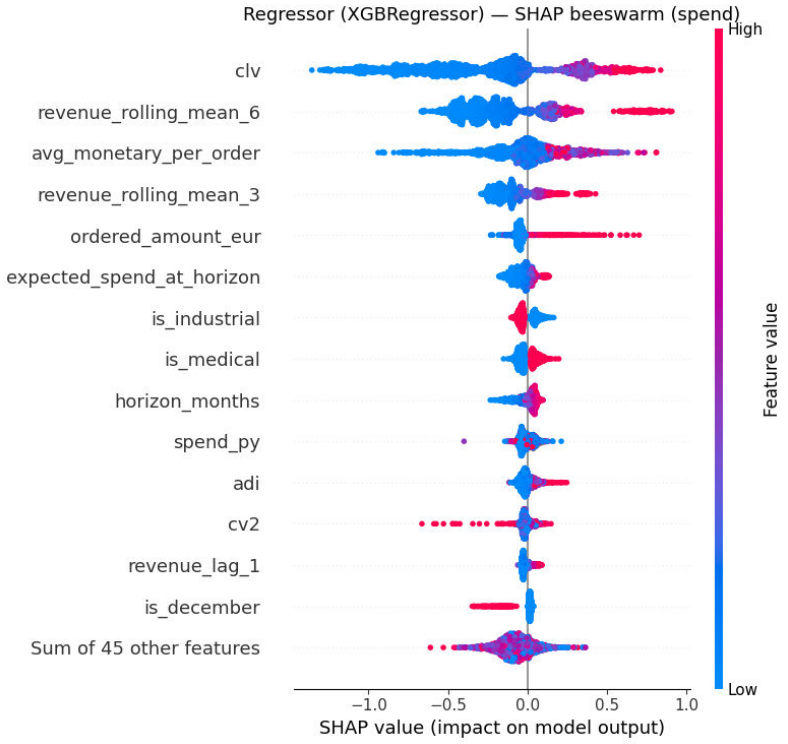

Figure 4.3: Regressor feature importance (SHAP beeswarm, conditional spend)

5. Discussion

The discussion is structured around the five key dimensions of the study. Section 5.1 reflects on the system development choices that motivated the hurdle architecture, while Section 5.2 evaluates the forecasting results in terms of granular and aggregate accuracy. Section 5.3 examines feature importance and its alignment with customer behavioral theory. Section 5.4 outlines the methodological limitations and scope of the data framework, and finally, Section 5.5 explores the sociotechnical implications of implementing the system within Optinova.

5.1 System Development

The results presented in Section 4 indicate that the hurdle model provides significant advantages compared to a naive baseline that simply estimates future spend as the previous year spend for the targeted month. This improvement on all levels of evaluation indicates that the model choice was appropriate for the characteristics of the data. While the model showed predictive ability to some extent, the zero-inflated, intermittent and non-contractual nature of Optinova's demand created challenges that technical improvements cannot fully clear up.

The framework does introduce a trade-off regarding incorrectly predicting that a customer will not buy. The classifier's false negative rate of 34-36% reflected the inherent difficulty of predicting events that are rare and the choice of 0.5 as the decision threshold. A threshold tuned against a revenue-weighted cost function would not necessarily fall at 0.5. However, this calibration was not performed in the thesis, since any deviation from 0.5 would have required specific domain justification or it would have to rely on cross-validation to tune the parameter. The 0.5 threshold should therefore be perceived as a defensible default for a proof-of-concept rather than an optimal operating point. This false negative rate could in theory produce larger errors than a standalone regression model. However, the regression evaluation in isolation showed substantially larger errors, indicating that the classifier successfully filters noise and that the false negative rate does not dominate the overall error.

Given the non-linear relationships in the customer data, gradient-boosting methods were chosen, specifically XGBoost and LightGBM, while Random Forest was included as a bagging alternative. The sequential error correction mechanism that is inherent to boosting was better suited to the complex and non-linear relationships in tabular customer data than the independent tree average of bagging in Random Forest. The LightGBM classifier combined with the XGBoost regressor was considered the best model configuration, though other configurations performed only slightly worse and still outperformed the baseline. The relatively small gap in performance between all top ranked configurations suggests that the architectural choices made as well as the feature engineering were more important than the fine grained model selection.

One of the largest efforts of this thesis was the feature engineering. Multiple conceptual frameworks were developed: RFM analysis, Syntetos-Boylan demand classification, lag structures and CLV. The integration of these features resulted in interpretable business dimensions and a feature set

of over 40 features. Because of the structural feature creation the result of the feature importance were enabled to be actionable for business understanding rather than purely technical outputs.

5.2 Forecasting Accuracy

The hurdle model achieved a strong performance at both customer and monthly aggregation levels, with row-level predictions displaying improvement over the naive baseline across most forecasted months. Based on the results from the held-out test set, the best model configuration was the LightGBM classifier and the XGBoost regressor which achieved MAE of 2,783 EUR at row level and a monthly MAPE of 11.89%. When aggregated annually to a single value the model had an APE of only 1.43% on the 12-month forecast window. These results show that a two-part hurdle architecture is applicable to the zero-inflated demand characteristics of Optinova's customer base.

When comparing the row-level errors to the customer-level aggregation, the coefficient of determination rose from 0.571 at row-level with predictions to 0.807 at customer-level. This finding indicates that individual month errors partly cancel when summarized across the 12-month horizons. This aggregation effect is important when interpreting the practical value, since quarterly and annual predictions for individual customers become more reliable and relevant than the single-month predictions. This indicates that the model's performance is stronger and more useful for capacity planning and resource allocation over longer periods.

The right-skewed conditional revenue distribution presents another challenge in the predictive model. The RMSE being approximately 2.4 times larger than the MAE signals there is substantial variance in the size of errors. High-revenue customers contribute disproportionately to MAE while representing a small fraction of the customer base. A single large customer mispredicted by a large sum could dominate RMSE even if hundreds of small customers are predicted accurately. This limitation is built into historical data-driven approaches since future spending cannot exceed observed maximums. Therefore customers with growing revenue are systematically underpredicted.

At customer-level, the baseline achieved a lower MAE than all model configurations. Despite the baseline achieving the lower MAE score, the models achieved better results in both RMSE and R^2. The baseline produces annual totals close to the actual on average, but with considerably higher variance, as reflected in the higher RMSE and lower R^2.

The temporal analysis showed variation across the forecasted horizon ranging from January to December 2025. The early months of the year showed the strongest model performance compared to the baseline, with January achieving 12.83% APE compared to the baseline's 50.49%. At mid-year the advantage disappeared, only to partially recover towards the final horizons. This decline in performance may reflect structural differences in the end of the year purchasing patterns including holiday closures, budget flash dynamics and accelerated ordering that the training data might not have represented correctly. The decline in performance also displays the complexity of forecasting late horizons, where forward-looking signals are not as prevalent as for the early horizons.

5.3 Feature Importance

The analysis of feature importance shows that customer purchasing behavioral patterns are indeed central to the model predictions. The two stages rely on fundamentally different signals between predicting a purchase versus predicting the magnitude of that purchase. For the classifier the most important features for the model were recency, demand density and average inter-demand interval which together accounted for 45.3% of influence. These features explained multiple patterns in non-contractual B2B purchasing. The existence of open orders proved to be a strong forward-looking signal with SHAP values up to 0.6 for customers with committed demand. The open order signal does however not dominate the total feature importance, since this feature is only available for customers with committed orders.

In the second stage of the two-part model, customer lifetime value and rolling averages dominated and explained together approximately 50% of the feature influence. The pattern suggests that the magnitude of revenue is better explained by historical customer value metrics rather than recent transactional spikes. The categorical segment feature indicating if a customer is in the medical or industrial sector shows opposing directional effects on revenue magnitude, suggesting that these segments have different purchasing patterns that the model has learned to separate.

These findings are consistent with existing literature on RFM analysis and customer segmentation. The contribution this thesis provides is utilizing these classical concepts as machine learning features rather than only using them as segmentation tools. The inclusion of demand regularity metrics derived from Syntetos-Boylan framework enabled the model to capture the degree of demand intermittency as predictive information rather than requiring separate handling of customers depending on their categorization as either intermittent or smooth.

The feature set for the classifier exceeded 40 features, and nearly 60 features for the regressor. Gradient-boosted trees generally handle noise in features well, but with the limited temporal depth of training data, some features might just introduce noise anyway. This risk is most pronounced at longer horizons, where derived forward-looking features carry less reliable signal, which may partly explain the model's weakest performance at horizon 12. One way of solving this could be using the SHAP feature importance and eliminate features with the lowest SHAP-value. The risk of doing this is that features with forward-looking signals, such as `has_open_order` and `ordered_amount_eur`, can receive relatively small SHAP values although they are very important signals, but only for a few customers. Removing these would therefore remove important signals.

Beyond realizing the value in features, SHAP-based feature importance provides Optinova with actionable insight into which customer behaviors drive revenue, offering a foundation for understanding different customers independently of the forecasting application.

5.4 Methodological Limitations and Scope

A real constraint of the confidence in generalization is the lack of longitudinal cross-validation. The single training period from February 2021 through November 2023 with a validation window for 2024 and test window for 2025 prevents real assessment whether performance is robust across

different time periods or if the model's performance is dependent on feature distributions only at the chosen snapshot dates. A walk-forward cross-validation would naturally partition time-series forecasting problems into multiple folds, but was infeasible due to the short observation window and the stated requirement to forecast 12 months ahead from each snapshot. With a larger validation horizon the confidence intervals on performance metrics could be estimated. The current setup does however not provide this support.

The model only captures the behavior of existing customers and therefore does not represent the full revenue picture of the company. Since new customers had no historical data for feature computation, including new customers would have required additional data points, which was out of this thesis scope. This represented a systematic forecast bias, since the actual revenue exceeded model predictions by default. The developed model is still operational as a prediction model of how existing customers could act in the future, which is a relevant use case for Optinova. To complement this forecasting system, another model could be developed that could estimate future revenue from new customers. Combining these two models would provide a more complete forecast, covering both predictable behavior from existing customers as well as the growth potential from new customers.

The system lacked prediction abilities for discontinuous changes such as major new contracts, supply chain redirects or strategic shifts that customers behavior were affected by. These cases may represent a large percentage of the customer base and made the time series analyzed more uncertain and unpredictable. Addressing this limitation would have required data points of external signals such as lead indicators, customer relationship management or strategic decisions made from the sales teams. This was beyond the scope of the thesis but would most likely add significant performance power to the model.

5.5 Operational and Sociotechnical Implications

For Optinova, the system serves several exploratory purposes which includes the validation whether machine learning can extract signals from transactional history, identify and understand existing forecast challenges and see which customer behavioral patterns are predictive and vice versa. These insights could inform more sophisticated approaches, whether or not they involve machine learning.

Beyond the merely technical perspective of this thesis, this work touches a sociotechnical perspective of how forecasting knowledge can be produced and handled at Optinova. The existing structure relies partly on the intuitive expertise of sales representatives' tacit knowledge on which customers will order, when they will order and how much. The model and architecture developed essentially encodes this tacit knowledge into measurable features such as recency, demand regularity and customer lifetime value. As described in Section 2.1 manual forecasting introduces the possibility of human bias in the form of incentive misalignment and cognitive bias. The model can offer a solution that is instead based solely on historical data rather than optimism or caution. It is however important to note that the current state of the proposed system cannot replace human expertise but must rather act as a support tool for decision making. The model's predictions could act as

a baseline for sales representatives to compare against. The SHAP-based feature importance is a useful tool since it prevents a black-box scenario and instead explains how each predictions are made. This strength could hopefully increase the model's trust by its users. This is one of the most crucial aspects of the operational value the system could provide, since a model that is technically sound but distrusted by its users is unlikely to deliver the gains that motivate its deployment.

6. Conclusions

This thesis developed and evaluated a B2B sales forecasting system on a customer-level. The core challenges that were addressed stemmed from the intermittent demand patterns present in the data. The conclusions below address each research question in turn.

Research question 1 asked how a customer-level sales forecasting system could be developed using machine learning for a manufacturing company. The answer that is presented in this thesis is using a two-part hurdle model. The first stage is a binary classifier which predicts whether a customer will purchase or not, yes or no. The regressor then predicts the revenue conditional on yes, and sets the prediction to zero if no. This system was developed within the CRISP-ML(Q) framework which emphasizes the value of data and business understanding. This phase directly motivated what features were needed to predict customer's demand. The feature engineering combined classical demand planning concepts such as RFM, Syntetos-Boylan classification and CLV. The final architecture used was explicitly created for the zero-inflated data observed and produces customer-month predictions across a 12-month horizon, aggregable to any sub-period.

Research question 2 asked how accurately machine learning can be used to forecast customer purchasing behavior and sales revenue, and what drives variation in performance across forecast horizons. The answer is that the best hurdle-pair configuration evaluated on the test set (LightGBM classifier and XGBoost regressor) outperformed the naive baseline across all evaluation levels, with the exception of customer-level MAE where the baseline's lower average error masked systematically larger deviations on high-revenue customers, as reflected in its higher RMSE and lower R^2. The model achieved an annual aggregate APE of 1.43% on the full year revenue total. This metric reflects the model's accuracy when summarizing the 12-month forecast to a single annual figure. At a customer-level, the model explained 80.7% of variance in annual spending in comparison to 70.4% for the baseline. The temporal analysis showed that the forecast accuracy varied across the 12-month horizon. While some months showed an APE of 3.45% (March), others showed 28.53% (December). Early horizons showed the strongest results in general, where forward-looking signals such as the presence of open orders are the most informative. Mid-year and December show the weakest results since there are fewer forward signals at long horizons.

Research question 3 asked what role customer purchasing behavioral patterns play in the model's predictive performance, and what insights could be drawn from feature importance for customer understanding. The answer is that the two hurdle stages depend on different aspects of behavior. For the binary classifier, recency, demand density and average inter-demand interval accounted for 45.3% of the total influence. The open-order flag was as expected one of the strongest forward signals for a customer. For the regressor stage CLV and rolling revenue means dominated and together accounted for more than half of the total influence. The feature importance showed that sector binaries (industrial/medical) show opposing directional effects and that the model has learned to separate these segment's purchasing patterns. These findings align with the existing literature on intermittent B2B data, and they provide the key insight that demand planning concepts can be used as machine learning features and not only as segmentation tools.

This thesis contributes methodologically by integrating established demand planning concepts with machine learning through a structured CRISP-ML(Q) workflow. While this thesis and its results are unique to the data used, the feature engineering approach and the multi-level evaluation are transferable to similar B2B forecasting problems. Future work could extend what is presented in this thesis with more relevant features consisting of external signals such as customer leads or signals from macroeconomic perspectives. Given specific customer categories or segments specific sub-models could be developed to capture patterns exclusive to those groups. Furthermore, it would be interesting to explore how this proposed solution could move from the exploratory phase into a production ready, real-time sales forecasting system, essentially continuing with the CRISP-ML(Q) framework.

References

[1] Rezazadeh, A. (2020), "A Generalized Flow for B2B Sales Predictive Modeling: An Azure Machine-Learning Approach", *Forecasting*, vol. 2, nr. 3, ss. 267–283.

[2] Xu, X., Tang, L., and Rangan, V. (2017), "Hitting your number or not? A Robust & Intelligent Sales Forecast System" in: *2017 IEEE International Conference on Big Data (Big Data)*, IEEE, ss. 3613–3622. DOI: 10.1109/BigData.2017.8258356.

[3] Giannopoulos, P. G., Dasaklis, T. K., Tsantilis, I., and Patsakis, C. (2025), "Machine learning algorithms in intermittent demand forecasting: a review", *International Journal of Production Research*, vol. 64, nr. 4.

[4] Syntetos, A. A., Boylan, J. E., and Croston, J. D. (2005), "On the categorization of demand patterns", *Journal of the Operational Research Society*, vol. 56, nr. 5, ss. 495–503.

[5] Rožanec, J. M., Petelin, G., Costa, J., Cerar, G., Bertalanič, B., Guček, M., Papa, G., and Mladenić, D. (2025), "Dealing with zero-inflated data: Achieving state-of-the-art with a two-fold machine learning approach", *Engineering Applications of Artificial Intelligence*, vol. 149, ss. 110339.

[6] Bohanec, M., Borštnar, M. K., and Robnik-Šikonja, M. (2017), "Explaining machine learning models in sales predictions", *Expert Systems with Applications*, vol. 71, ss. 416–428.

[7] Wisesa, O., Adriansyah, A., and Khalaf, O. I. (2020), "Prediction Analysis for Business To Business (B2B) Sales of Telecommunication Services using Machine Learning Techniques", *Majlesi Journal of Electrical Engineering*, vol. 14, nr. 4, ss. 145–153.

[8] Mortensen, S., Christison, M., Li, B., Zhu, A., and Venkatesan, R. (2019), "Predicting and Defining B2B Sales Success with Machine Learning" in: *2019 Systems and Information Engineering Design Symposium (SIEDS)*, IEEE, ss. 1–6. DOI: 10.1109/SIEDS.2019.8735626.

[9] Gupta, P., Batra, S. S., and Jayadeva (2017), "Sparse short-term time series forecasting models via minimum model complexity", *Neurocomputing*, vol. 243, ss. 1–11.

[10] Gupta, S., Hanssens, D., Hardie, B., Kahn, W., Kumar, V., Lin, N., Ravishanker, N., and Sriram, S. (2006), "Modeling Customer Lifetime Value", *Journal of service research*, vol. 9, nr. 2, ss. 139–155.

[11] Liu, D.-R. and Shih, Y.-Y. (2005), "Integrating AHP and data mining for product recommendation based on customer lifetime value", *Information & Management*, vol. 42, nr. 3, ss. 387–400.

[12] Khajvand, M., Zolfaghar, K., Ashoori, S., and Alizadeh, S. (2011), "Estimating customer lifetime value based on RFM analysis of customer purchase behavior: Case study", *Procedia Computer Science*, vol. 3, ss. 57–63.

[13] Fader, P. S. and Hardie, B. G. (2009), "Probability Models for Customer-Base Analysis", *Journal of interactive marketing*, vol. 23, nr. 1, ss. 61–69.

[14] Belotti, F., Deb, P., Manning, W. G., and Norton, E. C. (2015), "twopm: Two-part models", *The Stata Journal*, vol. 15, nr. 1, ss. 3–20.

[15] Garcia, B. (2013), "Implementation of a double-hurdle model", *The Stata Journal*, vol. 13, nr. 4, ss. 776–794.

[16] Bentéjac, C., Csörgő, A., and Martínez-Muñoz, G. (2021), "A comparative analysis of gradient boosting algorithms", *Artificial Intelligence Review*, vol. 54, ss. 1937–1967.

[17] Gu, Q., Zhu, L., Cai, Z., Liu, Y., Cai, Z., Li, Z., and Kang, Z. (2009), "Evaluation Measures of the Classification Performance of Imbalanced Data Sets" in: *Computational Intelligence and Intelligent Systems*, Springer Berlin Heidelberg: Berlin, Heidelberg, ss. 461–471.

[18] Colliot, O. (2023), Machine Learning for Brain Disorders. Springer Nature: New York.

[19] Hyndman, R. J. and Koehler, A. B. (2006), "Another look at measures of forecast accuracy", *International Journal of Forecasting*, vol. 22, nr. 4, ss. 679–688.

[20] Botchkarev, A. (2018), Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology. Available online: `http://arxiv.org/abs/1809.03006` (2026-03-15).

[21] Chicco, D., Warrens, M. J., and Jurman, G. (2021), "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation", *PeerJ Computer Science*, vol. 7, ss. 623.

[22] Antwarg, L., Miller, R. M., Shapira, B., and Rokach, L. (2021), "Explaining anomalies detected by autoencoders using Shapley Additive Explanations", *Expert systems with applications*, vol. 186, ss. 115736.

[23] Nohara, Y., Matsumoto, K., Soejima, H., and Nakashima, N. (2022), "Explanation of machine learning models using shapley additive explanation and application for real data in hospital", *Computer Methods and Programs in Biomedicine*, vol. 214, ss. 106584.

[24] Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., and Müller, K.-R. (2021), "Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology", *Machine learning and knowledge extraction*, vol. 3, nr. 2, ss. 392–413.

[25] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., and Dennison, D. (2015), "Hidden Technical Debt in Machine Learning Systems" in: *Advances in Neural Information Processing Systems*, Curran Associates: Red Hook, ss. 2503–2511.

[26] Sonwane, S. (2022), rfm: Python Package for RFM Analysis and Customer Segmentation. Available online: `https://pypi.org/project/rfm/` (2026-05-02).

[27] Cerqueira, V., Torgo, L., and Mozetič, I. (2020), "Evaluating time series forecasting models: an empirical study on performance estimation methods", *Machine Learning*, vol. 109, ss. 1997–2028.

[28] Mullahy, J. (1986), "Specification and testing of some modified count data models", *Journal of Econometrics*, vol. 33, nr. 3, ss. 341–365.

[29] Chen, T. and Guestrin, C. (2016), "XGBoost: A Scalable Tree Boosting System" in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery: San Francisco, California, USA, ss. 785–794. DOI: `10.1145/2939672.2939785`.

[30] Bergstra, J., Yamins, D., and Cox, D. (2013), “Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures” in: *Proceedings of the 30th International Conference on Machine Learning*, PMLR: Atlanta, Georgia, USA, ss. 115–123.

Appendix A

Table 1: Column descriptions for the sales dataset.

Column	Description
customer_code	Unique identifier for the customer
order_number	Unique identifier for the sales order
invoice_date	Date when the invoice was issued
sales_eur	Total sales amount in EUR
sales_type	Type of sale (External/Internal)
customer_group_name	Name of the customer group the customer belongs to

Table 2: Column descriptions for the customer information dataset.

Column	Description
customer_code	Unique identifier for the customer
company_no	Internal company number
customer_name	Registered name of the customer
customer_group_name	Name of the customer group
district	Geographic district code of the customer
seller_name	Name of the assigned sales representative
has_customer_group	Flag indicating customer group membership
parent	Parent segment from the customer segment hierarchy
country_code	ISO country code of the customer
created_on	Date when the customer record was created
sales_area	Geographic sales area derived from district

Table 3: Columns in the cleaned order_data dataset (one row per order).

Column	Description
customer_code	Unique customer identifier
order_number	Unique order identifier
sales_type	Type of sale (External/Internal)
invoice_date	Date when the invoice was issued
sales_category	Category of the sale
sales_eur	Total sales amount in EUR
customer_index	Numeric index fitted across all datasets via StringIndexer

Table 4: Columns in the cleaned `open_orders` dataset (one row per order).

Column	Description
`customer_code`	Unique customer identifier
`ordered_amount_eur`	Order amount in EUR
`order_date`	Date when the order was placed
`sales_period`	Delivery period (format `yyyy-MM`)
`customer_group_name`	Name of the customer group
`customer_index`	Numeric index fitted across all datasets via `StringIndexer`

Table 5: Columns in the cleaned `open_orders_history` dataset (one row per customer/order).

Column	Description
`customer_code`	Unique customer identifier
`ordered_amount_eur`	Order amount in EUR
`order_date`	Date when the order was placed
`sales_period`	Delivery period (format `yyyy-MM`)
`customer_index`	Numeric index fitted across all datasets via `StringIndexer`

Table 6: Columns in the cleaned `customer_information` dataset (one row per customer).

Column	Description
`customer_code`	Unique customer identifier
`is_medical`	Boolean flag
`is_industrial`	Boolean flag
`customer_group_name`	Name of the customer group
`country_code`	ISO country code of the customer
`sales_area`	Geographic sales area derived from the customer's district
`customer_index`	Numeric index fitted across all datasets via `StringIndexer`

Appendix B

Classifiers

Table 7: XGBoost Classifier Hyperparameters

Hyperparameter	Value
colsample_bytree	0.92
gamma	0.32
learning_rate	0.23
max_depth	5
min_child_weight	1.18
reg_alpha	0.31
reg_lambda	1.63
scale_pos_weight	1.5
subsample	0.53
n_estimators	16
random_state	42

Table 8: LightGBM Classifier Hyperparameters

Hyperparameter	Value
colsample_bytree	0.78
learning_rate	0.02
min_child_samples	50
num_leaves	31
reg_alpha	1.15
reg_lambda	1.31
scale_pos_weight	1.5
subsample	0.77
n_estimators	223
random_state	42

Regressors

Table 9: XGBoost Regressor Hyperparameters

Hyperparameter	Value
max_depth	7
min_child_weight	2
gamma	0.0
reg_alpha	0.01
reg_lambda	1.5
colsample_bytree	0.52
subsample	0.93
learning_rate	0.08
n_estimators	135
random_state	42

Table 10: LightGBM Regressor Hyperparameters

Hyperparameter	Value
colsample_bytree	0.51
learning_rate	0.09
max_depth	7
min_child_samples	50
num_leaves	63
reg_alpha	0.01
reg_lambda	1.5
subsample	0.94
n_estimators	132
random_state	42

Table 11: Random Forest Regressor Hyperparameters

Hyperparameter	Value
max_depth	15
min_samples_leaf	10
min_samples_split	20
max_features	0.7
max_samples	0.8
n_estimators	500
random_state	42