subtypelist subcaptionfont+=small,labelformat=parens,labelsep=space,skip=6pt,list=0,
*subcaption typelist1

Master Thesis Business Analytics

Predicting success of new product launches in the Dutch dairy market using machine learning methods

Author: Lars Willem van Dalen (2617114)

Vrije Universiteit Amsterdam

Ernst & Young x Royal FrieslandCampina

1st supervisor:	Karine da Silva Miras de Araujo	
daily supervisor:	Anne-Marèl Bos	(Ernst & Young)
2nd reader:	Ger Koole	

A thesis submitted in fulfillment of the requirements for the Master of Science degree in Business Analytics

February 11, 2024

Preface

As I approach the completion of my Master of Science in Business Analytics at Vrije Universiteit Amsterdam, I reflect on the incredible but tough journey that has brought me to this point. This thesis, titled “Predicting success of new product launches in the Dutch dairy market using machine learning methods", represents the culmination of my academic and practical experiences. During this project, I have found myself wandering in the fascinating worlds of the Dutch dairy market, machine learning and data preparation. It has been an incredible journey, and it has armed me with the skills and knowledge needed to tackle the complexities of data analytics and predictive modeling. Simultaneously, my time at Ernst & Young after pursuing two bachelor’s degrees in a mere four years, first as working student and later as intern, has further helped me to learn and improve in the field of business analytics. This thesis is my attempt to bridge the gap between theory and real-world application. My goal to learn more in the field of machine learning has been the driving force behind this research, and I hope the findings presented here contribute to the ever-evolving field of data analysis and prove to give valuable insights to Royal FrieslandCampina.

Acknowledgements

I would like to express my gratitude to the individuals who have supported and guided me throughout this research project.

I want to take a moment to express my immense gratitude for Anne-Marèl Bos from Ernst&Young for her valuable supervision and insights during this project. Her guidance has been essential in shaping the direction of this research and keeping me on track till completion. I would also like to extend my sincere thanks to the team at Royal FrieslandCampina, especially Fabian Dijkstra and Stefan Hulsbos, for their invaluable ideas, supervision, and insightful data discussions. Their industry knowledge and guidance have been vital in understanding the intricacies of the dairy market and the specific challenges faced by the intern company.

I would also like to thank Karine da Silva Miras de Araujo from Vrije Universiteit Amsterdam for her support as my supervisor. Additionally, I extend my appreciation to Ger Koole, the second reader from Vrije Universiteit, for his valuable suggestions and feedback.

In conclusion, I am thankful to Anne-Marèl Bos, Fabian Dijkstra, Karine Miras, Ger Koole, and all the individuals who have contributed to this research in any way. Their support and guidance have been invaluable in the successful completion of this study.

Abstract

Context. In the fast-paced and competitive Dutch dairy market, companies like Royal FrieslandCampina continuously seek innovative approaches to enhance their new product development strategies and increase the success rates of new product launches. While qualitative methods have traditionally been employed and are still the current way of working, there is an increased need to leverage sales data and competitor insights to predict the success of new product introductions in this fast moving consumer goods industry.

Goal. The goal of this research is to explore a possible method for predicting the success of new product launches in the dairy food market using a variety of machine learning methods, including logistic regression, support vector machines, decision trees and random forest. The study focuses on the Dutch market and aims to develop predictive models that can classify successful products relative to their direct competitors, possibly aiding the company in future decision making.

Method. To achieve the research goal, a comprehensive dataset containing historical sales data, competitor information, and time-based features is constructed. Feature engineering techniques are employed to extract relevant temporal patterns from the data and transform features into useable format. Various machine learning algorithms are implemented by training and validating on the dataset. The models are evaluated using appropriate performance metrics such as F1-Score, precision, recall and ROC AUC.

Results. The results conclude that the developed machine learning models exhibit moderate to promising predictive capabilities in determining the success of new product launches in the dutch dairy market. The time-based features, created through feature engineering, provide valuable insights into the temporal dynamics of the market and contribute significantly to the models' predictive performance. Branding shows to be of high importance shown by the impact encoded feature. The Random Forest model performs best overall with acceptable F1-Scores.

Conclusions. This research touches upon the effectiveness of different machine learning methods in predicting the success of new product launches in the dutch dairy market. By using historical sales data, these models could be further improved to provide a valuable decision support for new product development strategies. The introduction of time-based features enhances the models' ability to capture temporal patterns and improve predictions. The findings of this study form a basis to eventually enable companies like Royal FrieslandCampina to make more data-driven decisions, optimize resource allocation, and increase the likelihood of successful product introductions, ultimately improving market competitiveness.

Contents

List of Figures

List of Tables

Glossary

DART	ropouts meet Multiple Additive Regression Trees
DT	Decision Tree Model
DY	Drinking Yoghurts
EAN	European Article Number
FMCG	Fast Moving Consumer Goods
KPI	Key Performance Indicator
LASSO	Least Absolute Shrinkage and Selection Operator
LightGBM	Light Gradient Boosting Machine
LR	Logistic Regression model
ML	Machine Learning
NPD	New Product Development
QoQ	Quarter-on-Quarter
RF	Random Forest model
RFC	Royal FrieslandCampina
ROC AUC	Receiver operating characteristic - Area Under Curve
SGD	Stochastic Gradient Descent
SLP	Single-layer Perceptron
SVM	Support Vector Machine model
SY	Spoonable Yoghurts
WD	Weighted Distribution
XGBoost	Extreme Gradient Boosting

1

Introduction

The retail food & beverage industry is characterized by rapidly changing consumer preferences, intense competition, and an ever-evolving global marketplace. To remain competitive and meet emerging market demands, companies within this sector must constantly innovate their product offerings. New product development (NPD) plays a pivotal role in this dynamic context, acting as a cornerstone for growth, market share gains, and long-term sustainability.

Several studies have underscored the significance of NPD in the retail food & beverage domain. In the food industry, innovation is not just crucial for profitability but also for the survival of enterprises in an intensely competitive market(2). Furthermore, the tendency to buy new products has been strongly associated with consumer loyalty and repeat purchase behavior in the research of Steenkamp and Gielens (2003) (3). They argue that successful NPD can enhance brand loyalty and drive premium pricing strategies (3).

The introduction of new products allows retailers the opportunity to differentiate themselves in a crowded marketplace. As Porter (1980) emphasizes in his work on competitive business strategy, differentiation through product innovation serves as a powerful strategic tool, particularly in situations where competing based solely on price becomes untenable (4).

Moreover, in the dairy sector, the diversification of consumer preferred tastes has necessitated a constant flow of new product introductions. Companies are increasingly seeking ways to serve specialized market segments and offer differentiated flavors and textures. The variety of cultures, fermenting processes, and ingredient mix-ins further amplify the breadth of options available. Guiltinan (1999) suggests that the pace and quality of new product introductions are key determinants of market success in such segments (5).

As the complexity of consumer preferences grows, leveraging advanced technologies such as machine learning becomes imperative. Machine learning, with its ability to analyze vast and complex datasets, offers a valuable tool in understanding and predicting consumer preferences, allowing for more targeted and successful product innovations (6). This is particularly relevant in the food and beverage sector, where companies are constantly looking for ways to innovate and capture new market opportunities.

In conclusion, NPD is essential for companies in the retail food & beverage industries. Whether it is to meet the shifting tastes of consumers, outpace competitors, or carve out a distinct market position, innovative product development remains at the heart of strategic growth.

1.1 Problem statement

The scope of this research is the Dutch market, given its distinctiveness in product portfolio, brands, and categories. It is important to emphasize that the Dutch market holds particular significance for RoyalFrieslandCampina (RFC) due to its status as the company's home market and a major contributor to its revenue.

This thesis aims to explore the potential of machine learning in predicting new product success based on historical data in three specific categories within the Dutch market: Spoonable Yoghurts, Drinking Yoghurts, and Quarks. To achieve this, we utilize historical sales data provided by XXX.

Product success in this context is determined by two Key Performance Indicators (KPIs): Weighted Distribution (WD) and Rotation. The former, WD, assesses the product's availability across various geographical areas and retail channels within the Dutch market. The other KPI, Rotation, delves into the product's purchasing frequency by Dutch consumers. By analyzing these metrics, we aim to reveal the determinants of success for new products in product categories.

The ultimate outcome of this research will be insights into which factors are most important for product success and a classification of which products are likely to succeed. While it is possible to compile a top 50 list of potential successful products for further evaluation, it is suggested that such a comprehensive review might be more fitting for future studies. Using machine learning combined with historical data, we seek to uncover the various factors that influence product success, possibly aiding RFC's future decision-making in the Dutch market.

1.2 Organization

RoyalFrieslandCampina, commonly known as RFC, is a leading player in the dairy industry headquartered in the Netherlands with a vast global reach. As a pioneer in the dairy industry, their portfolio is diverse, boasting products like spoonable yoghurts, drinking yoghurts, and quarks. In order to remain competitive and meet evolving consumer demands, RFC is constantly exploring new product ideas and concepts. The success of these new products is often uncertain, as it depends on a range of factors such as product features, distribution channels, and consumer preferences.

My internship took place at RFC's office in Amersfoort, diving deep into the product success dynamics within the Dutch dairy sector. The research was spearheaded by a specialized team, known for their knack in leveraging data to inform product strategies. The ultimate aim of this research is to look into the possibilities to equip RFC with data-driven insights that will minimize product launch risks and optimize chances of winning in the ever-competitive dairy landscape of the Netherlands.

1.3 Thesis outline

Following the introduction, Chapter 2 provides background information on the definitions of successful product launches and introduces relevant machine learning and feature selection models.

Chapter 3, the literature review, highlights important and recent research related to product launches and NPD processes. The role and development of classification models in this area are also discussed.

In Chapter 4, we turn to the methodology. This chapter describes our dataset, its exploration, engineering, preprocessing, and the selection of key features for our research project. It explains the various models, their evaluation methods, and the overall approach taken to predict product launch success.

Chapter 5 presents the results. This chapter analyzes the performance of the machine learning models used, compares different models, and discusses the significance of various features.

Chapter 6 is the conclusion, where the findings related to product launches in the Dutch dairy market are summarized.

Chapter 7 offers a discussion on the study, touching upon its limitations and giving recommendations, while also suggesting potential directions for future research.

2

Background

The introduction of a new product to the market is a complex and capital intensive process, starting with an idea and ending with a market launch and subsequent evaluation of success. This section describes this intricate process within the context of the dairy industry in the Netherlands, followed by a description of the machine learning methods used in this research to forecast the market success of new dairy products. The models used include Logistic Regression, Support Vector Machines, Single-layer Perceptrons, Decision Trees, Random Forests, Extreme Gradient Boosting (XGBoost) and Light Gradient Boosting Machine (LightGBM)

2.1 New product development

The process of bringing a new product to the market is a challenging task, characterized by sequential stages, each being important to the product's eventual success. As outlined by Harmancioglu et al. (7), the process can be broken down in a series of core steps. These stages, visualized in figure 2.1, offer a comprehensive overview of the NPD process:

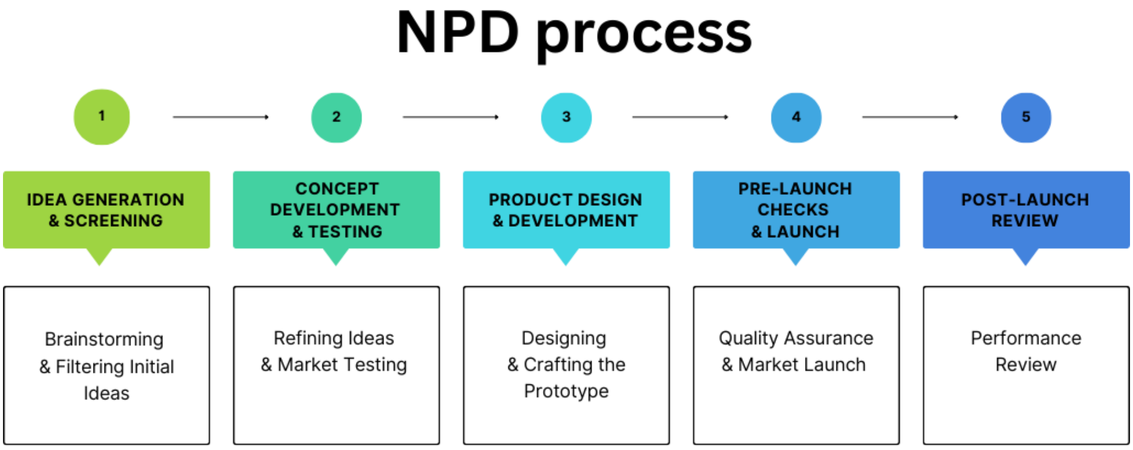
1. Idea generation and screening; At the outset, new product ideas emerge from diverse sources, including in-house teams, customers, competitors, and market research. Not all ideas move forward; only those with potential market and technical feasibility proceed (8).

2. Concept development and testing; Selected ideas proceed to form product concepts. These undergo testing among consumer groups to identify the target market and determine market acceptance (9).

3. Product design and development; Concepts form into prototypes during this phase. Refinements are made based on feedback, ensuring the product aligns with industry standards and all important consumer expectations (10).

4. Pre-launch checks and launch; Before its market debut, the product undergoes rigorous quality tests. Then the launch date is discussed with retail and promotional activities are determined (7).

5. Post-launch review; After launching, the product undergoes a detailed review where actual market outcomes are compared with initial expectations, leading to any necessary corrective actions. The post-launch review also prioritizes 'voice-of-customer' feedback, ensuring the product aligns closely with real-world customer preferences and reactions (11).


NPD process

1 IDEA GENERATION & SCREENING — Brainstorming & Filtering Initial Ideas

2 CONCEPT DEVELOPMENT & TESTING — Refining Ideas & Market Testing

3 PRODUCT DESIGN & DEVELOPMENT — Designing & Crafting the Prototype

4 PRE-LAUNCH CHECKS & LAUNCH — Quality Assurance & Market Launch

5 POST-LAUNCH REVIEW — Performance Review

Figure 2.1: New product development process.

2.2 NPD success

The launching a new product is a step-by-step process where each step plays a crucial role in how well the product performs. When looking at past research, it is clear that two things really matter: doing well in each phase, especially when making, testing, and introducing the product, and always keeping an eye on what customers want (12). This means listening to customers, knowing what is happening in the market, and understanding what the competition is up to (13). While it is good to know what the market in general wants, Von Hippel et al. state that it is equally as important to listen to specific customers who have clear and innovative ideas about what they want, customers often referred to as 'Lead Users' (14).

In the Dutch dairy sector, determining the success of product launches is a complex task. For the purposes of this research project, guided by the preferences of the host company, the emphasis is on actual sales performance. Two KPIs, namely Weighted Distribution (WD) and Rotation, stand out as most important factors for determining success. WD represents a percentage metric of the stores the product is sold in, adjusted for the size and significance of individual stores in the Netherlands. Rotation, on the other hand, serves as an indicator of consumer preference, indicating the average units sold per store for a particular product. These metrics, compared to relative market performance, form the basis of gauging the market success of new dairy products in this study. Figure 2.2 below provides a general visualization of how products can be categorized as being successful or not, based on their relative performance in terms of Rotation and WD.

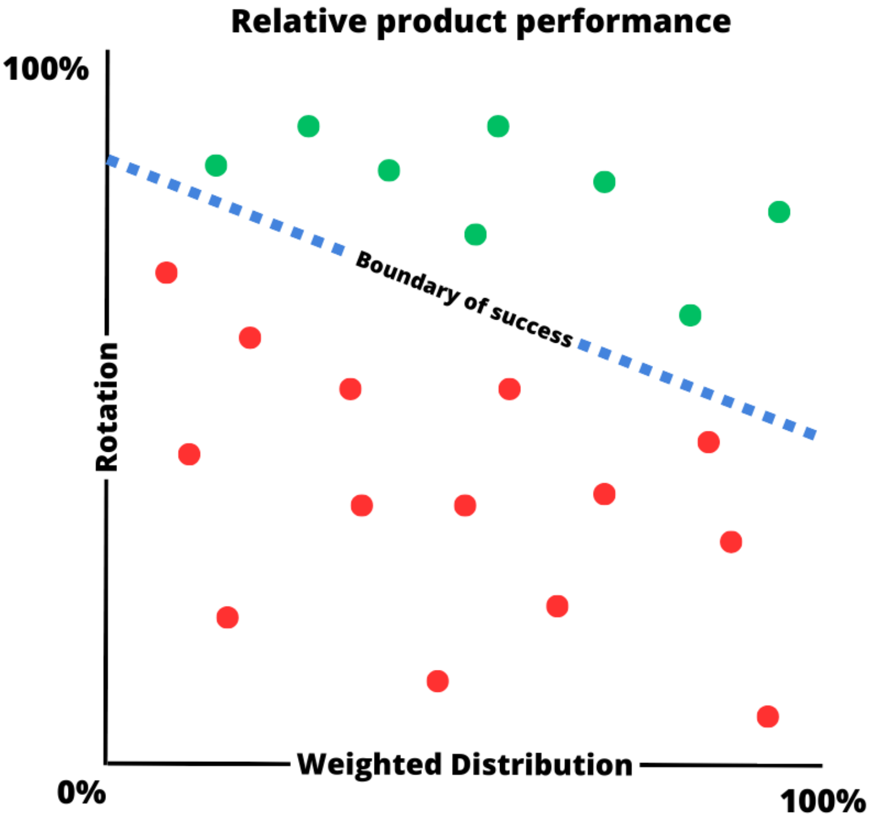


Figure 2.2: New product success based on relative market performance.

2.3 Machine learning models

Machine learning can be compared to cooking, if you start with good quality ingredients (data), you are more likely to create a tasty dish (accurate predictions). In other words: “Garbage in, Garbage out" (GIGO), meaning that if you train a model with poor data, you will probably get poor predictions (15).

It is crucial to consider that there is no one-size-fits-all recipe in machine learning. This phenomenon is often referred to as “No Free Lunch", meaning there is no single model that is superior for every task (16). Often multiple different models are tried to see which one works best for the specific problem. After all, whether simple or complex, every model comes with its strengths and challenges.

In this thesis, the project is focusing on supervised learning, where we give the model both the ingredients (data) and the recipe (algorithm) to help it make predictions. Both tree-based and linear models are explored to see which one is the best fit for our data and task.

2.3.1 Linear models

2.3.1.1 Logistic Regression (LR)

Logistic Regression employs a linear model for making predictions. Unlike linear regression, which predicts continuous values, Logistic Regression is used for predicting probabilities. It utilizes a logistic or sigmoid function to transform a linear combination of features into a value between 0 and 1, often referred to as the probability of an outcome (17).

The relation can be mathematically represented as:

$$log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_m X_m \tag{2.1}$$

In this equation, $\frac{\pi}{1-\pi}$ denotes the odds, the ratio of probabilities (17). The coefficients β_i represent the weights for each corresponding feature, X_i, being the value of the feature (18). The term β_0 serves as a reference point or often called intercept term, indicating the baseline (18).

One of the main advantages of this model is its clear interpretative nature of its predictions (19). A drawback of the model is that it supposes a linearity between the features and their log odds, this condition is often not fulfilled in real-world data (19). The model allows for certain hyperparameters to be tuned, striking a balance between model simplicity and its fit to training data.

Hyperparameters:

- *Regularization Type:* Determines the type of regularization applied. Options are L1 (Lasso) and L2 (Ridge). L1 tends to zero out insignificant features, while L2 shrinks coefficient magnitudes.

- *C (Inverse Regularization Strength):* A continuous value typically ranging between 0.01 and 100. A smaller value signifies stronger regularization.

2.3.1.2 Support Vector Machine (SVM)

Support Vector Machines (SVMs) represent a powerful classification model when working with high-dimensional spaces (20). An SVM tries to find the optimal hyperplane for separating the data into classes, with achieving a high as possible margin between class members (21). The data points closest to this hyperplane, the so-called support vectors, are fundamental in shaping this decision boundary. In its essence, SVM can be viewed as a Maximum Margin Classifier due to its unique ability to minimize empirical classification errors while maximizing geometric margins (21).

However, while SVMs excel in high-dimensional spaces and maintain resilience against outliers, their computational demands can render them less appropriate for datasets of considerable size (20). The model's performance and decision boundaries are influenced by key hyperparameters, which dictate the relationship between bias and variance, the nature of the decision boundary, and the training process's termination criteria (20).

Hyperparameters:

- *Kernel:* Defines the decision boundary type. Options include Linear, RBF, Polynomial, and Sigmoid. The kernel function assesses sample similarities.

- *C (Regularization Strength):* Represents the penalty parameter of the error term. A lower C value yields a smoother decision boundary, leaning towards higher bias. Conversely, a higher value focuses on correct classification of training examples but may run the risk of over-fitting.

- *Gamma:* This kernel coefficient is applicable to RBF, Polynomial, and Sigmoid kernels. It defines the 'influence' of each training sample within the feature space. A lower value suggests a widespread influence, whereas a larger value suggests influence to a closer range.

- *Tolerance:* A threshold for the stopping criterion, ensuring the solver achieves a specified precision.

- *Maximum Iterations:* Sets the limit on iterations during model fitting.

2.3.1.3 Single-layer Perceptron (SLP)

Being a simple neural network, the Single-layer Perceptron (SLP) is often used for classification and prediction tasks (22). The SLP consists of multiple inputs, often denoted by an input vector $X = \{X_1, X_2, \ldots, X_n\}$, and produces a singular output, O (23). The output is calculated as:

$$Ouput = f(V^T X + v_0) \tag{2.2}$$

where $f(net)$ symbolizes a non-linear activation function, commonly a sigmoid function defined as

$$f(net) = \frac{1}{1 + \exp(-net)} \tag{2.3}$$

The terms v_0 and $V = \{V_1, V_2, \ldots, V_p\}$ represent the bias and weights of the perceptron, respectively (22).

By computing a weighted aggregate of its inputs, augmented by a bias, the perceptron determines an activation. If the sigmoid function is used, the output is compared to the treshold value of 0.5 (22).

The linear decision boundary of the perceptron allows it to distinguish between two classes based on feature coordinates. However, it is essential to remember the perceptron's limitations, such as its inability to handle non-linearly separable data (22).

Hyperparameters:

- *Max Epochs:* The upper limit on the number of iterations over the training data set.

- *Alpha (L2 Regularization Parameter):* Penalizes larger weights to reduce over-fitting.

- *Convergence Tolerance:* Specifies the precision threshold triggering solver termination.

- *Early Stopping:* Option to halt training upon lack of validation score enhancement.

- *Initial Learning Rate:* Sets the onset rate for model learning.

- *Solver:* Choice between methods like ADAM and SGD to optimize weights, each having its set of hyperparameters.

For **ADAM**:

- *$Beta_1$ / $Beta_2$:* Exponential decay rates for initial and secondary moment estimations.

- *Epsilon:* A small value to prevent division by zero during optimization.

For **SGD**:

- *Momentum:* Enhances SGD speed and damps oscillations.

- *Learning Rate Annealing:* Adjusts the learning rate over time.

- *Power_t:* The exponent for inverse scaling learning rate.

2.3.2 Tree-based models

2.3.2.1 Decision Tree (DT)

The Decision Tree models is often referred to as an intuitive decision-making processes often visualized as flowcharts, like in Figure 2.3 (24). Within this model, features are depicted as internal nodes, the branches represent decision rules, and the outcomes are illustrated at the leaf nodes (25).

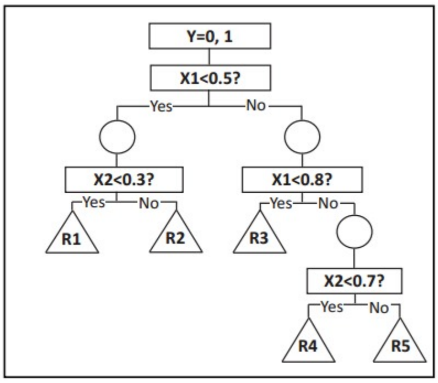


Figure 2.3: Example of a decision tree model (24).

The DT partitions data by selecting features that yield the highest information gain, trying to split the data subsets into perfectly seperated leaf nodes (25). One strong advantage of this model is its visual interpretabilty and its ability at managing both numerical and categorical data (24). It is important to note that the model can be prone to overfitting, especially with deeper trees that mimic the training data too closely. The importance of tuning hyperparameters, like the tree's depth and minimum samples for a split, should not be overstated as they play a crucial role in the model's predictive performance (24).

The mathematical formulation of DT, especially with criteria like entropy or the Gini impurity, illustrate the rationale behind determining splits. For instance, entropy can be represented as:

$$Entropy(S) = -p_+ \log_2(p_+) - p_- \log_2(p_-) \tag{2.4}$$

where p_+ and p_- signify the proportions of positive and negative samples, respectively (25).

Similarly, the Gini impurity is defined as:

$$Impurity(S) = 1 - (p_+^2 + p_-^2) \tag{2.5}$$

The Gini impurity quantifies the disorder in a set, with higher values denoting greater impurity (25).

Hyperparameters:

- *Maximum Depth:* Limits the tree's growth potential, controlling its complexity.

- *Criterion:* Methodology for assessing the quality of a split. Options are "Gini" and "Entropy" as discussed above.

- *Min. Samples per Leaf:* Minimum number of samples necessary in a leaf node, influencing the tree's granularity.

- *Split Strategy:* Strategic approach used for split selection at each node. Options include "Best", which opts for the most optimal split, and "Random", which selects the best arbitrary split.

2.3.2.2 Random Forest (RF)

The Random Forest model is an bagging ensemble technique that builds upon the idea of using multiple 'simple' decision trees to provide a more robust prediction (21). An example of this can be seen below in Figure 2.4

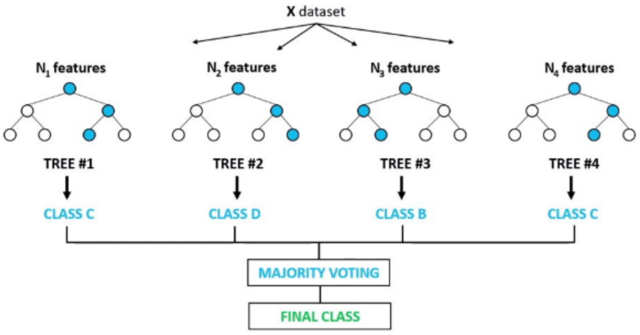


Figure 2.4: Example of a random forest model (26).

Instead of relying on a single tree, the RF model takes into account the outcomes of multiple trees, each constructed using a randomized subset of the data and features. The predictions from these trees are combined through a simple majority voting mechanism for classification tasks (21). This approach reduces overfitting, increases model robustness and results in better prediction accuracy at the expense of interpretability (27).

Hyperparameters:

- *Number of trees:* Specifies the total number of trees in the forest.

- *Feature sampling strategy:* Determines the approach to feature sampling, such as fixed proportion or square root.

- *Proportion of features to sample:* A value typically ranging between 0.1 and 0.7, dictating the fraction of features considered during each split.

- *Maximum depth of tree:* Maximum depth of each tree in the forest.

- *Minimum samples per leaf:* Sets the least amount of samples that a leaf node can have.

2.3.2.3 Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting, or XGBoost, is a potent classification model employing the boosting ensemble technique. It sequentially optimizes decision trees to correct previous trees' errors, minimizing a specific loss function, as shown in Figure 2.5. This approach ensures an enhancement in computational efficiency, a graceful handling of missing data, and effective in-built regularization techniques, making XGBoost a robust tool in diverse scenarios (28).

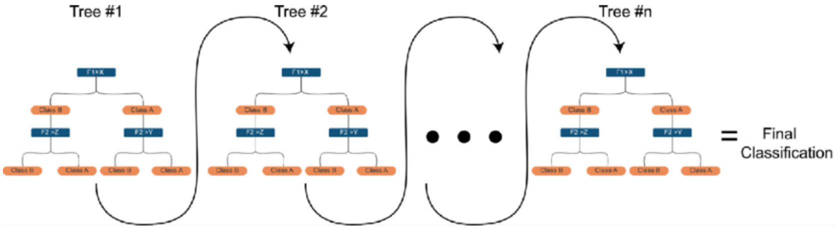


Figure 2.5: Sequential optimization in XGBoost model (29)

.

Regularization in XGBoost not only diminishes the risks of overfitting but also strikes a balance between model accuracy and complexity. One notable hyperparameter in XGBoost is the booster method. DART (Dropouts meet Multiple Additive Regression Trees) mitigates over-specialization in traditional MART (Multiple Additive Regression Trees), a scenario where late-added trees adversely influence few instances, reducing the model's performance on unseen data (30). Through the innovative use of dropouts, DART enhances the model's generalization capabilities and sensitivity to initial trees, resulting in improved performance across various tasks with large-scale datasets (30). Below an extensive list of hyperparameters is briefly explained.

Hyperparameters:

- *Booster:* Options include Gradient Boosted Trees and DART.

- *Tree method:* Options include Exact, Approx, Histogram and Automatic, with the latter relying on heuristics and dataset shape for selection.

- *Maximum number of trees:* Maximum total trees in the ensemble.

- *Early Stopping:* Enables early termination to prevent potential overfitting.
- *Early stopping rounds:* Number of rounds without improvement before stopping.
- *Max tree depth:* Maximum depth of a tree.
- *Learning rate:* Step size at each iteration.
- *Max delta step:* Limits the maximum step size during weight optimization.
- *L1 & L2 regularization:* Regularization terms added to the objective function.
- *Gamma:* Minimum loss reduction to make a split.
- *Minimum child weight:* Minimum sum of instance weights required in a child node.
- *Subsample ratio:* Ratio of training data sampled for building trees.
- *Columns subsample ratio for trees:* Ratio of features sampled for constructing each tree.
- *Columns subsample ratio for splits/levels:* Ratio of features sampled for each split.

2.3.2.4 Light Gradient Boosting Machine (LightGBM)

Light Gradient Boosting Machine, or LightGBM, is another gradient boosting model based on the decision tree algorithm. The model is especially useful for large datasets as it is designed for distributed and efficient training (31). Figure 2.6 illustrates a schematic representation of the model, emphasizing its leaf-wise growth approach in tree building as opposed to complete level-wise in XGBoost for example.

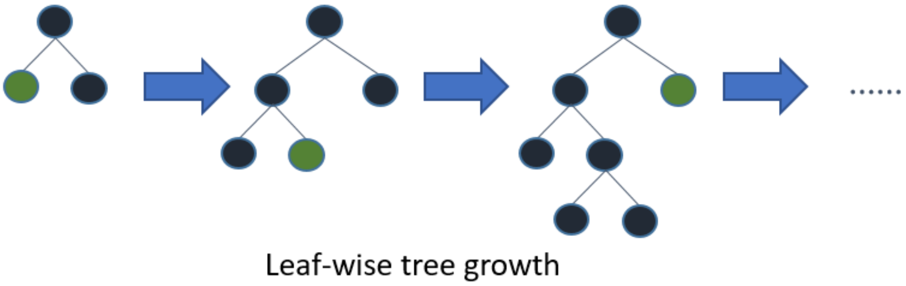


Figure 2.6: Leaf-wise tree growth in LightGBM (32).

As stated, the approach in XGBoost is level-wise tree growth, as LightGBM adopts a leaf-wise algorithm it can achieve lower loss than its counterpart, leading to improved accuracy (31). However, this also causes a higher risk of overfitting on smaller datasets, and careful tuning of hyperparameters is required to prevent this issue (31). Below, essential hyperparameters different from XGBoost are outlined and briefly explained.

Hyperparameters:

- *Number of leaves:* Maximum number of leaves in one tree. This is unique as it's directly related to LightGBM's leaf-wise tree growth, and it's crucial for the model's performance and speed.

- *Minimum split gain:* Minimum loss reduction required to make a further split. It provides a criterion for making additional partitioning of leaves.

- *Minimum leaf samples:* Minimum required instances in a leaf.

- *Bagging fraction and frequency:* Subsample ratio and frequency of data used during training.

2.3.3 Ensemble modelling

In machine learning, the term 'Ensemble modelling' refers to the combined decision of multiple models to try and enhance overall performance. This could be compared to collective intelligence or 'wisdom of crowds', where often a large group of people tend to make very accurate estimates or judgments (33).

One crucial advantage of this method is its ability to lower the risk of over-fitting and increase performance by utilizing the strengths of each of the integrated models (34). However, ensemble modelling also comes with drawbacks such as increased difficulty of results interpretation, one example for this is RF models (35).

Recent and commonly used techniques are bagging, boosting and stacking (36). Bagging uses multiple predictors trained on different dataset subsamples, used in for example the RF model (36). Boosting, sequentially trains weak learners to correct predecessors' errors, exemplified in methods like XGBoost or LightGBM (36). Stacking combines predictions from diverse algorithms (36).

In this research project, the stacking technique is implemented with the majority voting technique, meaning the final classification is determined by the majority vote of the individual models (37). A visual representation of this can be seen below in Figure 2.7.

2.4 Feature selection techniques

Feature selection, a crucial pre-processing step in the world of machine learning, is used to try and avoid superfluous and repetitive data. The process often improves the results from learning algorithms, making them not only more accurate but also easier to understand (39). With data dimensionality expanding exponentially in recent years, efficiency and effectiveness have become challenges for traditional feature selection and extraction

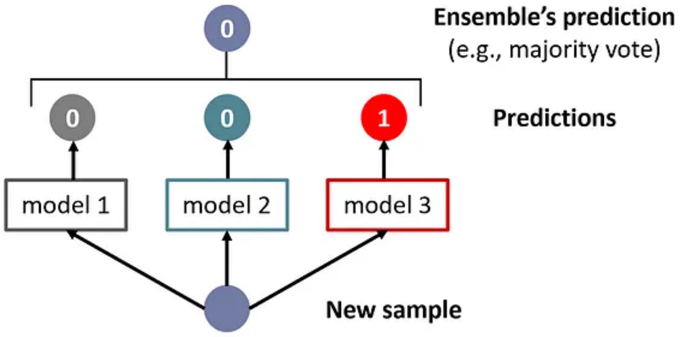


Figure 2.7: Schematic representation of majority vote ensemble classification(38).

methods, as discussed by Hall and Holmes (2003) (40). Various strategies have been introduced in machine learning and pattern recognition to improve the performance of learning algorithms and increase the predictive accuracy of classifiers.

In this study, we focus on three distinctive yet effective feature selection methods: *Correlation with Target*, *Tree-based Feature Selection*, and *LASSO Regression*. Each of these has been analyzed and recognized in previous literature for their unique strengths in the feature selection process.

The *Correlation with Target* approach is straightforward yet effective. It selects features that exhibit a strong Pearson correlation with the target variable, guaranteeing that the chosen features significantly linearly relate to the target, thereby serving as a solid basis for training models (41). A set threshold for minimum absolute correlation makes this a solid systematic process.

Introducing a more dynamic approach, the *Tree-based Feature Selection* technique starts by creating a Random Forest model for target prediction, subsequently selecting the most important features that contribute to the model's predictive accuracy (42). Eventually only the features with the highest significance are selected and used for training the models. While this method is insightful, it has a drawback in the fact that it tends to favor features with multiple categories.

Lastly, the *LASSO Regression* method is used. LASSO, or Least Absolute Shrinkage and Selection Operator, not only performs regularization but also selects the most important features by penalizing the coefficients of regression variables, which reduces variance without significantly increasing bias, especially in data with fewer observations and numerous features (43). In the research project, 3-fold cross-validation is used in this method to precisely identify the optimal regularization term, making sure that features with non-zero coefficients, which are essential for the model, are retained, enhancing the model's predictive accuracy avoiding overfitting.

Using the above mentioned methods, this research project aims to identify essential features that enhance the model's predictive power and highlight underlying data relationships.

3

Literature

Understanding the success of new product development and launches is a difficult challenge with a lot of variables that come into play. Various research approaches and methodologies have been employed to grasp, assess, and forecast the performance of new products in the market. Recognizing this broad topic, this literature review is mainly focused on examining papers that have researched new product development and innovation in this space as well as on examining papers that have utilized machine learning (ML) models to predict future success or demand of new launches.

This literature study aims to provide relevant insights for the overall research by concentrating on works that have leveraged ML techniques such as classification for predictive analysis. This section will methodically explore the existing body of knowledge, assessing the applicability and effectiveness of different ML models and feature selection methods in the context of pre-launch success prediction.

3.1 New product development fast moving consumer goods

In their 2002 work, Traill and Meulenberg (2) explored innovation dynamics within European food manufacturing, having their doubts on the split between 'demand-pull' and 'technology-push' models. They emphasized the need for a more varied framework considering company culture, strategic orientation, and structural variables like market type and company size to fully understand innovation trajectories. The relevance of this work to my research is profound, it gives a broader context on what might be important to take into account while launching new products or services. Traill and Meulenberg's (2) insights aid in refining the feature engineering process of the models by underscoring the importance of a company's innovation orientation—product, process, or market. Incorporating these aspects as quantitative variables could offer a potential enhancement in predictive accuracy. Additionally, their observations on the impact of product quality on development align with the possible approach of categorizing products on brand-level to determine success. Together these qualitative and quantitative analyses make for a holistic view of product launch outcomes. Overall, their findings underscore the necessity of a diversified approach in predictive modeling, a principle that informs and enhances my own methodology.

In his 1999 paper, Nijssen analyzed the multifaceted challenges and outcomes associated with line extensions in the consumer goods sector (44). He questioned the efficacy of using established brand names for new product introductions, especially when there is strong market competition, enhanced retailer power, and the evolving behaviour of consumers (44). The research is based on a survey of industry professionals and suggest that that the risk of cannibalization is significant and quality enhancements often do not add to success whereas new flavors and packaging might (44). Nijssen's findings are invaluable and his research around line extensions and therefore incremental innovation in products aligns with my own research. My research relies on historical sales data to forecast new product performance, which leads to the fact that the researched and created models of my work will be able to forecast for smaller changes or adjustments in products better than radical innovations. His identification of successful line extension characteristics—namely new flavors and packaging variations—provides a solid foundation for developing predictive features within my models (44). Just as Nijssen points out the potential pitfalls of cannibalization and the minimal value addition of some line extensions, perfect models must also differentiate between features that truly drive sales versus those that merely redistribute existing demand among similar products (44).

3.2 Machine learning studies

The 2017 paper by Quader et al. provides valuable insights into the application of machine learning for predicting movie success, offering parallels to my own research on product launch predictions (45). Employing SVM and Neural Networks, the study successfully forecasts box office performance using historical data from IMDb and similar sources, emphasizing pre-release features like budget and IMDb votes. This approach mirrors my use of historical sales data to predict new product success, although I focus on binary classification rather than the multi-tiered success categories used in their research. Both studies recognize the significance of pre-launch features, but the movie success model also contemplates post-release factors and wider economic conditions, suggesting avenues for my research to explore, such as the impact of market dynamics on product adoption. Their concept of ‘one away’ prediction accuracy also presents an intriguing alternative metric for evaluating predictive performance. The findings in the film industry underscore the potential of machine learning applications across various domains, highlighting opportunities for enhancing the predictive robustness of my models.

In a 2023 study, Arampatzis et al. look into pre-launch forecasting of new product sales in the fashion industry, a task similar to my own research (46). Their study employs a range of similar ML models, including Decision Trees, XGBoost, LightGBM, and Random Forest, leveraging product features as variables for predictions as in my research. Next to that, they have access to an extra feature which portrays the potential sales if the product had been launched earlier, this aids the models predictions and is something not available in my research. They provide a forecast of sales volumes in the first six weeks after launch, whereas I utilize historical sales data to categorize the first 26 weeks of a product launch being successful or not. The diversity in analytical methods used is a common thread in our research, reflecting a departure from a one-method-fits-all mentality to embrace a more explorative and comprehensive approach. In contrast to their study, my research does not focus on deep learning techniques and while they implement bayesian optimization or grid search, depending on the model, my research utilizes random search for hyperparameter optimization next to using K-fold Cross-Validation to ensure the robustness and generalizability of the predictive models.

In 2020, Narayanan et al. studied pre-launch product success prediction by leveraging the vast expanse of electronic Word of Mouth (e-WOM) data to forecast outcomes in the electronics domain (47). They implemented a Multithreaded Hash-join Resilient Distributed Dataset (MHRDD), which not only refines data quality by eradicating redundancies but also enhances prediction model performance. The study shows the significance of e-WOM data, including encompassing reviews, comments, and ratings, in shaping product quality and market success. Their research aligns with the trajectory of my research, which also harnesses ML techniques, but instead focussing on historical sales data. While Narayanan et al. integrate customer feedback into their models, my study is restricted by the absence of such direct consumer insights, focusing instead on quantifiable sales records and product features. The employment of similar machine learning methods, like Decision Trees and XGBoost, bridges our studies. However, the inclusion of product reviews and additional e-WOM data in Narayanan et al.’s model presents a layer of consumer sentiment analysis absent from my research. This dimension introduces a potential step for future research, where exploring such qualitative data could improve the prediction capabilities for products where historical sales data is scarce or non-representative, particularly for groundbreaking innovations. While both studies look into their unique datasets, consumer-generated content versus historical sales data, the overarching goal to decipher the code of product launch success through ML unites them.

4

Methodology

This chapter starts by presenting a detailed overview and comprehension of the datasets utilized in our research. Following this, we describe the sequential methodology used to prepare and extend the available data in order to be able to run the explained models on the resulting datasets. Next, the steps towards achieving insightful results in order to address the research goal are discussed.

4.1 Data

4.1.1 Data Description

4.1.2 Data Preparation

4.1.2.1 Unique product keys

4.1.2.2 Product characteristics data

4.1.2.3 New products

4.1.2.4 Handling missing values

4.1.2.5 Brand mapping

4.1.2.6 Feature Engineering

4.1.3 Binary Classification

4.1.4 Data preprocessing

4.1.4.1 Scaling Numerical Data

Scaling numerical values is a step that can not be forgotten as it ensures equal contribution of different features to a model's predictions. This study employed two scaling approaches for scaling numerical data: Minimum-Maximum (Min-Max) scaling and Average-Standard Deviation (Avg-Std) scaling.

Minimum-Maximum Scaling: Min-Max scaling normalizes the data within the range [0, 1]. Despite its benefits, it is sensitive to outliers due to the direct use of the minimum and maximum values. The normalization is given by the formula:

$$y = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4.1}$$

where x represents the original value, and y is the normalized value.

Average-Standard Deviation Scaling: Also known as Z-score normalization, Avg-Std scaling centers the data around the mean with a standard deviation of one. This method is more robust against outliers and maintains the shape of the original distribution, making it suitable for normally distributed features. The standardization formula is:

$$y = \frac{x - \bar{x}}{\sigma} \tag{4.2}$$

where x is the original feature value, y is the standardized value, $\bar{x}$ is the mean, and σ is the standard deviation.

Avg-Std scaling proved to be slightly more effective in our modeling process, likely due to its resilience to outliers and the preservation of the distribution's shape.

4.1.4.2 Handling High Cardinality: Impact Encoding

As discussed above, the categorical feature 'Brand' has very high cardinality. Therefore, one-hot encoding was determined to be impractical. Instead, impact encoding was used, which combines the brand-specific mean with the overall mean, providing a more generalizable encoding method. The impact encoded value is calculated as follows:

$$\text{Impact encoded value} = \frac{n \cdot \bar{Y}_{\text{cat}} + m \cdot \bar{Y}}{n + m} \tag{4.3}$$

where $\bar{Y}_{\text{cat}}$ is the category mean, $\bar{Y}$ is the global mean, n is the number of samples in the category, and m is a smoothing parameter that moderates the influence of the global mean. This encoding technique is especially beneficial for categories with limited samples as these are shifted more towards the overall mean, this reduces the risk of overfitting and enhances the model's ability to generalize.

4.2 Training, Validation and Test data split

In order to perform the modelling for this study, the data has been divided into distinct subsets to train, validate, and test the predictive models. The division is based on a time-based 5-fold cross-validation method, ensuring that the temporal order of the data is preserved. This approach is particularly crucial for our dataset as it is inherently time-series, with the created variable 'WeekinData' ranging from 1 to 283 serving as the chronological marker.

The 5-fold cross-validation technique with overlap is employed during the hyperparameter tuning phase. This method allows us to validate the model's performance across different time periods, ensuring that our findings are robust and not merely tailored to a specific moment in time. This method partitions the data into five overlapping folds, where each fold serves as a validation set once while the data preceding the validation set forms the training set. This process is shown in 4.1 below where blue is always the training set and green always the validation set.

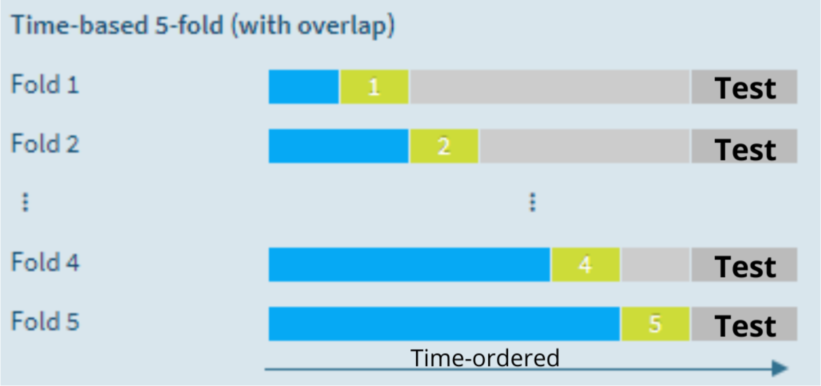


Figure 4.1: Time-based 5-fold training, validation and test set

In terms of the overall dataset, an 80/20 split ratio for training and testing is chosen. This ratio is chosen to provide a substantial amount of data for the model to learn from in the training set, while still reserving a considerable portion for final evaluation on the test set. The split is conducted in a chronological manner, where the first 80% of instances sorted on 'WeekinData' is allocated for training, and the remaining 20% for testing.

The integrity of the time-ordering is maintained throughout the data splitting process. This means that the model is always trained on past data and tested on future data relative to the training set. This is critical for time-series forecasting and ensures that the evaluation of the model is realistic and actually works similar as how the model would perform when deployed in a real-world setting.

Overall, data splitting process is designed to optimize model performance and generalization. It respects the temporal nature of the data and aligns with the best practices for time-series analysis.

4.3 Feature Selection

In this study, three different feature selection methods were evaluated independently to identify the most effective approach for the predictive models. These methods include *Correlation with Target*, *Tree-based Feature Selection*, and *LASSO Regression*. Each method was applied separately, and their outcomes were compared to determine which technique offered the most valuable insights for model enhancement.

The first method, *Correlation with Target*, operates on a straightforward yet powerful principle. It selects features based on their Pearson correlation with the target variable. This approach ensures that the chosen features have a significant linear relationship with the target, providing a strong foundation for model training. In this research, a set threshold for the minimum absolute correlation was employed to select the top 25 features, making the selection process both systematic and focused.

Next, the *Tree-based Feature Selection* technique was implemented. This method involves creating a Random Forest model, which in this study consists of 30 trees with a depth of 10, to predict the target variable. The model then identifies and keeps the top 25 most important features based on their contribution to the model's predictive accuracy. While insightful and powerful, this method tends to favor features with multiple categories, which is a consideration in interpreting the results.

Lastly, *LASSO Regression* was utilized. LASSO (Least Absolute Shrinkage and Selection Operator) is known for its functionality of both regularization and feature selection. This method penalizes the coefficients of regression variables, reducing variance without significantly increasing bias. This is particularly useful in scenarios with fewer observations and numerous features. In this study, a 3-fold cross-validation approach was adopted to determine the optimal regularization term from the set $\{0.01, 0.1, 1, 10, 100\}$. The process ensures that only features with non-zero coefficients, which are deemed essential, are retained in the final model.

The primary objective of exploring these three methods was not to combine their results but to compare their effectiveness in selecting the most relevant features.

Correlation with Target offers a straightforward, linear perspective. *Tree-based Feature Selection* provides an insight based on ensemble learning. *LASSO Regression* combines regularization with feature selection, beneficial for high-dimensional data.

By analyzing the outcomes of each method separately, the study aimed to identify which approach best aligns with the specific characteristics of our dataset and the modeling objectives.

4.4 Hyperparameter tuning

In this study, the Random Search approach for hyperparameter tuning was used. Hyperparameter tuning is a critical step in optimizing the performance of machine learning models. It involves experimenting with various settings to find the most effective combination of parameters for a given model.

Random Search is an efficient and practical alternative to Grid Search, especially when dealing with large parameter spaces or constraints on computational resources (48). Unlike Grid Search, which tests all possible combinations within the parameter space, Random Search selects random combinations of hyperparameters to evaluate. This approach can often result in comparable or superior results to Grid Search with significantly less computational overhead (48).

For this research, the search space limit was set to 50, balancing the breadth of the hyperparameter exploration with the practical constraints of computational resources. This limit makes sure that there is a thorough examination of the parameter space while preventing excessively long search times. The search was unconstrained by time limits, allowing the exploration process to be thorough and unbiased towards quicker, potentially less effective combinations. This approach ensures that the hyperparameter tuning is comprehensive and considers a wide range of potential model configurations.

The tuning process utilized a time-ordered 5-fold cross-validation with overlap, as previously discussed in Figure 4.1. This method of cross-validation maintains the temporal ordering of the data, which is crucial for time-series analysis. It resembles a realistic assesement of the model's performance over time, similar to how it would be implemented in a real world situation and avoids potential data leakage. The focus was on optimizing the models parameters for a higher F1-Score rather than accuracy. This decision was made due to the presence of class imbalance within the dataset. F1-Score, which harmonizes precision and recall, offers a more balanced metric in scenarios where each class's performance is important.

4.5 Experimental setup

In this section, we describe the setup for the twelve different experiments including the different pre-processing and scaling steps, discussing the machine learning models used and the different hyperparameters subject to tuning. The setup for all the models are presented in the tables below. The hyperparameters are tuned as discussed in section 4.4, the features are selected with three different methods as discussed in section 4.3 and the features are pre-processed in the different ways explained in section 4.1.4.

Logistic Regression A well-known basic model in the area of statistical modeling, Logistic Regression was chosen for its simplicity and efficiency in binary classification tasks. It also serves as a baseline model to which the performance of more complex algorithms can be compared. The different hyperparameters and ranges can be seen below in Table 4.1.

Parameter	**Description**	**Search Range or Value**
Regularization Type	Determines the type of regularization applied. L1 (Lasso) tends to zero out less significant features, while L2 (Ridge) shrinks coefficient magnitudes.	{L1, L2}
C (Inverse Regularization Strength)	A continuous value that controls the strength of regularization. A smaller value signifies stronger regularization.	{0.01, ..., 100}

Table 4.1: Description of hyperparameters for the Logistic Regression model

Support Vector Machine The Support Vector Machine (SVM) model, known for its effectiveness in high-dimensional spaces, was included for its robustness and versatility in handling both linear and non-linear boundaries. The hyperparameter *Gamma* was fixed at scale, to be the inverse of $\#features*variance$, the hyperparameter *Tolerance*, a threshold for the stopping criterion to achieve desired solver precision, was fixed at 0.001 and the *Maximum number of iterations* was set to unlimited. The other hyperparameters to be tuned are can be seen in Table 4.2 below.

Single-layer Perceptron For the Single-layer Perceptron model, a specific set of hyperparameters was chosen to facilitate effective training without the need for extensive tuning. The model utilized a hidden layer size fixed at 10 with the ReLU activation function. The

Parameter	**Description**	**Search Range or Value**
Kernel	Defines the decision boundary type with options including Linear, RBF, Polynomial, and Sigmoid. The kernel function evaluates the similarity between samples.	Linear, RBF, Polynomial, Sigmoid
C (Regularization Strength)	Represents the error term's penalty parameter. A lower C value smooths the decision boundary for higher bias, while a higher value aims for correct classification of training examples, risking over-fitting.	[0.1, 10]
Gamma	The kernel coefficient for RBF, Polynomial, and Sigmoid kernels, affecting each training sample's influence in the feature space. Lower values indicate a wider influence, while larger values are more localized.	Scale

Table 4.2: Description of hyperparameters for the SVM model

training process was guided by a maximum iteration limit of 200, ensuring sufficient learning while trying to prevent overfitting. The L2 regularization parameter (*Alpha*) was set to 0.001, adding a penalty to larger weights to also avoid overfitting.

The convergence of the model was controlled with a tolerance threshold of 0.0001, and early stopping was enabled to halt training if there was no significant improvement in the validation score, avoiding unnecessary further optimization. The initial learning rate was fixed at 0.001.

Regarding the optimization solver, two approaches were experimented with: ADAM and SGD (Stochastic Gradient Descent). For ADAM, the exponential decay rates for the first and second moment estimations (*Beta_1* and *Beta_2*) were fixed at 0.9 and 0.999, respectively, with an *Epsilon* value of 1e-8 to prevent division by zero. In the case of SGD, the momentum was set at 0.8 to enhance the speed of convergence and dampen oscillations, with the learning rate following a 'constant' annealing schedule. The *Power_t* exponent for the inverse scaling learning rate was set at 0.5, and Nesterov momentum was utilized to further refine the optimization process.

Both ADAM and SGD solvers were tested to determine their effectiveness in weight optimization, providing a comparative perspective on their impact on the model's performance.

Decision Tree The Decision Tree model was selected for its interpretability and the ease with which it handles feature interactions, providing clear insights into decision-making processes. The hyperparameters and their search space can be seen in Table 4.3.

Parameter	**Description**	**Search Range or Value**
Maximum Depth	Limits the tree's growth potential, controlling its complexity. A deeper tree captures more details but risks overfitting.	{3, ..., 10}
Criterion	Methodology for assessing the quality of a split. Gini impurity measures the frequency at which any element of the dataset will be mislabeled when randomly labeled, whereas Entropy is a measure of the purity of the split.	{Gini, Entropy}
Min. Samples per Leaf	Minimum number of samples necessary in a leaf node, influencing the tree's granularity and its ability to capture fine details in the data.	{1, ..., 20}
Split Strategy	Strategic approach used for split selection at each node. "Best" opts for the most optimal split, while "Random" selects a random split.	{Best, Random}

Table 4.3: Description of hyperparameters for the Decision Tree model

Random Forest The Random Forest model, an ensemble of decision trees, is known for its high accuracy and tendency to avoid overfitting, making it an ideal candidate for this study. The hyperparameter values and search ranges can be seen below in Table 4.4.

Extreme Gradient Boosting The XGBoost model was chosen for its state-of-the-art performance in numerous machine learning competitions. Its speed and efficiency make it a powerful tool to use on our dataset. The maximum number of trees in the ensemble, dictating the model's complexity, was set to 300. Early stopping was enabled with four rounds before stopping to avoid overfitting. The large amount of hyperparameters tuned can be seen in Table 4.5.

Light Gradient Boosting Machine The Light Gradient Boosting Machine (LGBM) is renowned for its performance and speed, especially on large datasets, it was included for testing purposes and possibility to perform as well as or better than other models. Similar

Parameter	**Description**	**Search Range or Value**
Number of Trees	Specifies the total number of trees in the forest, impacting the model's accuracy and computational complexity.	{80, ..., 120}
Feature Sampling Strategy	Determines the approach to feature sampling, like a fixed proportion of features used in each tree.	Fixed proportion
Proportion of Features to Sample	Dictates the fraction of features considered for each split, influencing model accuracy and overfitting.	{0.1, ..., 0.7}
Maximum Depth of Tree	Maximum depth allowed for each tree, affecting the model's complexity and potential for overfitting.	{6, ..., 20}
Minimum Samples per Leaf	The minimum number of samples required in a leaf node, influencing the granularity of the model.	{1, ..., 20}

Table 4.4: Description of hyperparameters for the Random Forest model

Parameter	**Description**	**Search Range or Value**
Booster	Booster method, options include Gradient Boosted Trees (GBT) and DART for the boosting process.	{GBT, DART}
Max Tree Depth	The maximum depth of a single tree, affecting the model's ability to capture data complexities.	{3, ..., 10}
Learning Rate	The step size at each iteration, a crucial parameter influencing the convergence of the model.	{0.1, ..., 0.5}
L1 Regularization	L1 regularization term added to the objective function to encourage sparsity.	{0, ..., 1}
L2 Regularization	L2 regularization term, which encourages smaller and more generalized weights in the model.	{0.01, ..., 1}
Gamma	The minimum loss reduction required for making a new split in the tree.	{0, ..., 1}
Minimum Child Weight	Minimum sum of instance weights needed in a child node, influencing the decision to make further splits.	{0.5, ..., 5}
Subsample Ratio	Proportion of training data sampled for building each tree, affecting variance and bias.	{0.5, ..., 1}
Columns Subsample Ratio for Trees	Ratio of features sampled for constructing each tree, aiding in feature selection.	{0.5, ..., 1}

Table 4.5: Description of hyperparameters for the XGBoost model

to the XGBoost model, early stopping was enabled with four rounds before stopping to avoid overfitting. The hyperparameters tuned can be seen below in Table 4.6.

Parameter	**Description**	**Search Range or Value**
Max Number of Trees	Specifies the upper limit on the number of trees in the ensemble.	{50, ..., 200}
Number of Leaves	Determines the maximum number of leaves per tree. Affects model complexity and overfitting risk.	{20, ..., 500}
Learning Rate	Controls the step size at each iteration of the model training.	{0.1, ..., 0.2}
L1 Regularization	L1 regularization term (Lasso), encouraging sparsity in the model.	{0, ..., 1}
L2 Regularization	L2 regularization term (Ridge), penalizing the magnitude of coefficients.	{0, ..., 1}
Minimal Gain for Split	The minimum gain required for executing a split in the tree.	{0, ..., 1}
Min Sum of Instance Weight in a Child	Sets the minimum sum of instance weights (hessian) needed in a child.	{0.001, ..., 1}
Columns Subsample Ratio for Trees	The ratio of features used for constructing each tree, affecting feature selection and model variance.	{0.5, ..., 1}

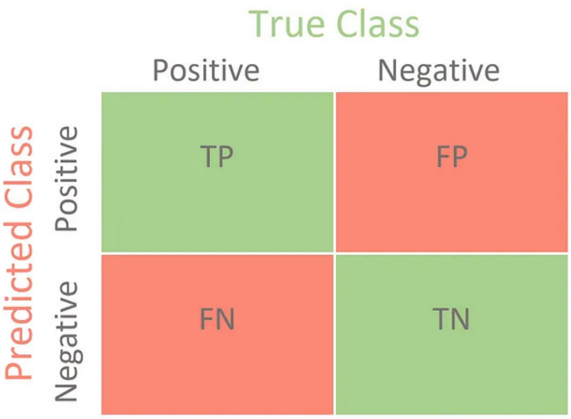
Table 4.6: Description of hyperparameters for the LightGBM model

4.6 Model Evaluation

Evaluating the performance of machine learning models with labelled data that has a class imbalance requires a careful selection of performance metrics to look at. Traditional measures like accuracy may not be sufficient in such scenarios as this would not give a fair result in highly imbalanced classes. Therefore, this study employs a range of metrics, including confusion matrices, F1-Score, precision, recall, and ROC-AUC, to provide a comprehensive assessment of model performance.

4.6.1 Confusion Matrix Analysis

As a start, confusion matrices are used to visualize each model's performance. These matrices offer a detailed breakdown of predictions into four categories: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) as can been seen below in Figure 4.2. They are particularly useful in understanding how well the models perform in identifying the minority class and in revealing the balance between sensitivity (recall) and specificity.



	Positive	Negative
Positive	TP	FP
Negative	FN	TN

Figure 4.2: Visual representation confusion matrix (1).

4.6.2 F1-Score, Precision, and Recall

Building upon the visual insights from confusion matrices, the F1-Score is selected as the primary metric for model comparison. It is the harmonic mean of precision and recall, providing a single metric that accounts for both the precision (the ratio of correctly predicted positive observations to the total predicted positive observations) and recall (the ratio of correctly predicted positive observations to all observations in the actual class). The equations for precision and recall can be derived directly from the confusion matrix as follows:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN} \tag{4.4}$$

The F1-Score can then be calculated as:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.5}$$

4.6.3 ROC-AUC Curve

In addition to the F1-Score, precision and recall, the Receiver Operating Characteristic (ROC) curve and its corresponding Area Under the Curve (AUC-ROC) are utilized. The ROC curve plots the True Positive Rate (Recall) against the False Positive Rate, providing insights into the model's performance across various thresholds. While the ROC-AUC is a popular metric, it is interpreted with caution in imbalanced datasets.

Together, these metrics together with the visual insights from the confusion matrices, create a robust and comprehensive evaluation of the models. They ensure that the selected model not only performs well across various thresholds but also addresses the specific challenges posed by the imbalanced nature of the dataset.

5

Results

5.1 Model performance on Spoonable Yoghurts

The modelling on the Spoonable Yoghurts dataset was extensive, using a variety of machine learning models tuned through an iterative hyperparameter tuning process. The evaluation of these models was conducted through a series of 12 experiments, each varying in feature selection, scaling techniques, and impact encoding values for the 'Brand' feature. This comprehensive approach was used to find the most effective combination of preprocessing and model tuning to accurately predict the target variable.

5.1.1 Experimentation Framework

With the objective of maximizing model performance and achieving the best results, the experimentation was structured around three choices in the preprocessing step:

- Feature Selection: Utilizing three distinct methods, as detailed in the methodology section 4.
- Feature Scaling: Using two different scaling techniques allowed for tests on how normalized data in different ways could influence performance.
- Brand Impact Encoding: The 'Brand' feature was encoded using two different smoothing factors, 52 and 104, to assess the impact of the level of smoothing on model performance.

The combination of these paramterers led to 12 unique experimental settings, each providing valuable insights into the modeling process. The specifics of these experiments can be seen in Table **??** in Appendix A. The performance of each model was assessed using the metrics discussed earlier in section 4 to get a broad view of predictive power, including F1-Score, accuracy, precision, recall, and ROC AUC. These metrics were collected and have all been combined in comprehensive tables for ease of comparison. All results for each experiment and model can be seen in the Appendix A.

5.2 Main Experiment

The focus is on the experiment that stood out for its superior performance, using tree-based feature selection, Avg/Std feature scaling to limit the effect of outliers and a smoothing factor of 104 for the impact encoded 'Brand' feature to improve generalisation and avoid overfitting of brands that had a small number of product launches. This section provides an in-depth analysis of the models within this experimental set-up.

5.2.1 Model-wise Performance Analysis

The breakdown of each model's performance is shown by the discussed metrics and confusion matrices that together offer a good view of each model's efficacy. F1-Scores, accuracy, precision, recall, and ROC AUC values provide quantifiable benchmarks for evaluating the effectiveness of the models. These metrics not only show the models' abilities to predict accurately but also give insights into the trade-offs between different types of errors they make. Additionally, the feature importance provide information on which predictors within the model have the most weight in influencing the outcome, offering insights into the decision making progress which could be interesting for the intern company. The confusion matrices are a clear visual tool for understanding the models' classification accuracy by presenting the correct and incorrect predictions in a format that's easy to interpret, highlighting the models' strengths and weaknesses in distinguishing between classes. The ROC AUC curves shown in Figure XXX further enrich this analysis by representing the trade-off between the true positive rate and the false positive rate at various thresholds. These curves are a testament to the models' ability to balance sensitivity and specificity, ultimately guiding the selection of an appropriate threshold for classification.

5.2.1.1 Linear regression

5.2.1.2 Support Vector Machine

5.2.1.3 Single-Layer Perceptron

5.2.1.4 Decision Tree

5.2.1.5 Random Forest

5.2.1.6 Extreme Gradient Boosting

5.2.1.7 Light Gradient Boosting Machine

5.3 Expansion to Quarks and Drinking Yoghurts Datasets

5.3.1 Comparative Performance Analysis

5.3.2 Model Efficacy

5.4 Ensemble Modeling Results for Spoonable Yoghurts

6

Conclusion

7

Discussion & Future research

References

[1] Joydwip Mohajon. **Confusion Matrix for Your Multi-Class Machine Learning Model — towardsdatascience.com**. `https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826`. [Accessed 12-12-2023]. ix, 31

[2] W Bruce Traill and Matthew Meulenberg. **Innovation in the food industry**. *Agribusiness: an International Journal*, **18**(1):1–21, 2002. 1, 17

[3] Jan-Benedict EM Steenkamp and Katrijn Gielens. **Consumer and market drivers of the trial probability of new consumer packaged goods**. *Journal of Consumer Research*, **30**(3):368–384, 2003. 1

[4] Michael E Porter and Competitive Strategy. **Techniques for analyzing industries and competitors**. *Competitive Strategy. New York: Free*, 1980. 1

[5] Joseph P Guiltinan. **Launch strategy, launch tactics, and demand outcomes**. *Journal of Product Innovation Management: AN INTERNATIONAL PUBLICATION OF THE PRODUCT DEVELOPMENT & MANAGEMENT ASSOCIATION*, **16**(6):509–529, 1999. 1

[6] Osvaldo Simeone. **A very brief introduction to machine learning with applications to communication systems**. *IEEE Transactions on Cognitive Communications and Networking*, **4**(4):648–664, 2018. 2

[7] Nukhet Harmancioglu, Regina C McNally, Roger J Calantone, and Serdar S Durmusoglu. **Your new product development (NPD) is only as good as your process: an exploratory analysis of new NPD process design and implementation**. *R&d Management*, **37**(5):399–424, 2007. 4, 5

[8] Daniel J Flint. **Compressing new product success-to-success cycle time: Deep customer value understanding and idea generation**. *Industrial marketing management*, **31**(4):305–315, 2002. 4

[9] Ricardo Henrique Da Silva, Paulo C Kaminski, and Fabiano Armellini. **Improving new product development innovation effectiveness by using problem solving tools during the conceptual development phase: Integrating Design Thinking and TRIZ**. *Creativity and Innovation Management*, **29**(4):685–700, 2020. 4

[10] Michael G Luchs, K Scott Swan, and Mariëlle EH Creusen. **Perspective: A review of marketing research on product design with directions for future research**. *Journal of Product Innovation Management*, **33**(3):320–341, 2016. 5

[11] Robert G Cooper. **What leading companies are doing to re-invent their NPD processes**. *PDMA Visions Magazine*, **32**(3), 2008. 5

[12] Roger J Calantone, Shawnee K Vickery, and Cornelia Dröge. **Business performance and strategic new product development activities: an empirical investigation**. *Journal of Product Innovation Management: An International Publication of the Product Development & Management Association*, **12**(3):214–223, 1995. 5

[13] Daniel R Denison and Aneil K Mishra. **Toward a theory of organizational culture and effectiveness**. *Organization science*, **6**(2):204–223, 1995. 5

[14] Eric Von Hippel. **Lead users: a source of novel product concepts**. *Management science*, **32**(7):791–805, 1986. 6

[15] R Stuart Geiger, Dominique Cope, Jamie Ip, Marsha Lotosh, Aayush Shah, Jenny Weng, and Rebekah Tang. **" Garbage In, Garbage Out" Revisited: What Do Machine Learning Application Papers Report About Human-Labeled Training Data?** *arXiv preprint arXiv:2107.02278*, 2021. 7

[16] David H Wolpert and William G Macready. **No free lunch theorems for optimization**. *IEEE transactions on evolutionary computation*, **1**(1):67–82, 1997. 7

[17] Alfred DeMaris. **A tutorial in logistic regression**. *Journal of Marriage and the Family*, pages 956–968, 1995. 7

[18] Michael P LaValley. **Logistic regression**. *Circulation*, **117**(18):2395–2399, 2008. 7

[19] S Philip Morgan and Jay D Teachman. **Logistic regression: Description, examples, and comparisons**. *Journal of Marriage and Family*, **50**(4):929–936, 1988. 7

[20] Shihong Yue, Ping Li, and Peiyi Hao. **SVM classification: Its contents and challenges**. *Applied Mathematics-A Journal of Chinese Universities*, **18**:332–342, 2003. 8

[21] Daud Muhajir, Muhammad Akbar, Affindi Bagaskara, and Retno Vinarti. **Improving classification algorithm on education dataset using hyperparameter tuning**. *Procedia Computer Science*, **197**:538–544, 2022. 8, 11

[22] Šarūnas Raudys. **On the universality of the single-layer perceptron model**. In *Neural Networks and Soft Computing: Proceedings of the Sixth International Conference on Neural Networks and Soft Computing, Zakopane, Poland, June 11–15, 2002*, pages 79–86. Springer, 2003. 9

[23] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. **Learning internal representations by error propagation. Parallel Distributed Processing: Exploration in the Microstructure of Cognition, vol. 1**. *Foundations*, pages 318–362, 1986. 9

[24] Yan-Yan Song and LU Ying. **Decision tree methods: applications for classification and prediction**. *Shanghai archives of psychiatry*, **27**(2):130, 2015. 10

[25] Anthony J Myles, Robert N Feudale, Yang Liu, Nathaniel A Woody, and Steven D Brown. **An introduction to decision tree modeling**. *Journal of Chemometrics: A Journal of the Chemometrics Society*, **18**(6):275–285, 2004. 10, 11

[26] Davis David. **Random Forest classifier tutorial: How to use tree-based algorithms for machine learning**, Aug 2020. 11

[27] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. **Do we need hundreds of classifiers to solve real world classification problems?** *The journal of machine learning research*, **15**(1):3133–3181, 2014. 11

[28] Santhanam Ramraj, Nishant Uzir, R Sunil, and Shatadeep Banerjee. **Experimenting XGBoost algorithm for prediction and classification of different datasets**. *International Journal of Control Theory and Applications*, **9**(40):651–662, 2016. 12

[29] Jimmy Abualdenien and Andre Borrmann. **Ensemble-learning approach for the classification of Levels Of Geometry (LOG) of building elements**. *Advanced Engineering Informatics*, **51**:101497, 01 2022. 12

[30] Rashmi Korlakai Vinayak and Ran Gilad-Bachrach. **Dart: Dropouts meet multiple additive regression trees**. In *Artificial Intelligence and Statistics*, pages 489–497. PMLR, 2015. 12

[31] Essam Al Daoud. **Comparison between XGBoost, LightGBM and CatBoost using a home credit dataset**. *International Journal of Computer and Information Engineering*, **13**(1):6–10, 2019. 13

[32] **Features x2014; LightGBM 4.1.0.99 documentation — lightgbm.readthedocs.io**. `https://lightgbm.readthedocs.io/en/latest/Features.html`. [Accessed 27-09-2023]. 13

[33] Derek A Epp. **Public policy and the wisdom of crowds**. *Cognitive systems research*, **43**:53–61, 2017. 14

[34] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. **A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches**. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **42**(4):463–484, 2011. 14

[35] Gérard Biau and Erwan Scornet. **A random forest guided tour**. *Test*, **25**:197–227, 2016. 14

[36] Thomas G Dietterich. **Ensemble methods in machine learning**. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000. 14

[37] Omer Sagi and Lior Rokach. **Ensemble learning: A survey**. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **8**(4):e1249, 2018. 14

[38] Dr. Roi Yehoshua. **Introduction to Ensemble Methods — pub.towardsai.net**. `https://pub.towardsai.net/introduction-to-ensemble-methods-226a5a421687`. [Accessed 27-09-2023]. 15

[39] Batta Mahesh. **Machine learning algorithms-a review**. *International Journal of Science and Research (IJSR).[Internet]*, **9**(1):381–386, 2020. 14

[40] Mark Andrew Hall and Geoffrey Holmes. **Benchmarking attribute selection techniques for discrete class data mining**. *IEEE Transactions on Knowledge and Data engineering*, **15**(6):1437–1447, 2003. 15

[41] Isabelle Guyon and André Elisseeff. **An introduction to variable and feature selection**. *Journal of machine learning research*, **3**(Mar):1157–1182, 2003. 15

[42] Kofi O Nti, Adebayo Adekoya, and Benjamin Weyori. **Random forest based feature selection of macroeconomic variables for stock market prediction**. *American Journal of Applied Sciences*, **16**(7):200–212, 2019. 15

[43] Robert Tibshirani. **Regression shrinkage and selection via the lasso**. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, **58**(1):267–288, 1996. 15

[44] Edwin J Nijssen. **Success factors of line extensions of fast-moving consumer goods**. *European Journal of Marketing*, **33**(5/6):450–474, 1999. 18

[45] Nahid Quader, Md Osman Gani, Dipankar Chaki, and Md Haider Ali. **A machine learning approach to predict movie box-office success**. In *2017 20th International Conference of Computer and Information Technology (ICCIT)*, pages 1–7. IEEE, 2017. 18

[46] Marios Arampatzis, G eorgios Theodoridis, and Athanasios Tsadiras. **Pre-launch Fashion Product Demand Forecasting Using Machine Learning Algorithms**. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 362–372. Springer, 2023. 19

[47] Sandhya Narayanan, Philip Samuel, and Mariamma Chacko. **Product Pre-Launch Prediction From Resilient Distributed e-WOM Data**. *IEEE Access*, **8**:167887–167899, 2020. 19

[48] James Bergstra and Yoshua Bengio. **Random search for hyper-parameter optimization.** *Journal of machine learning research*, **13**(2), 2012. 25

Appendix A

Appendix