Using Recency, Frequency and Monetary variables to predict Customer Lifetime Value with XGBoost

by Marius Errol Myburg

MYBMAR003

SUBMITTED TO THE UNIVERSITY OF CAPE TOWN

In partial fulfilment of the requirements for the degree

MSC IN INFORMATION TECHNOLOGY

Supervisor: Professor Sonia Berman

Date: July 2022

Plagiarism declaration

I, Marius Errol Myburg, hereby declare that the work on which this thesis is based is my original work (except where acknowledgements indicate otherwise) and that neither the whole work nor any part of it has been, is being, or is to be submitted for another degree in this or any other university. I authorise the University to reproduce for the purpose of research either the whole or any portion of the contents in any manner whatsoever.

Signature:……Signed by candidate…… Date: 25 July 2022

Abstract

The importance of machine learning techniques in the field of Customer Relationship Management (CRM) will continue to gain prominence in the coming years. A commonly used CRM metric called Customer Lifetime Value (CLV) is the value a customer will contribute while they are an active customer. This study investigated the ability of supervised machine learning models constructed with XGBoost to predict future CLV, as well as the likelihood that a customer will drop to a lower CLV in the future. One approach to determining CLV, called the RFM method, is done by isolating Recency (R), Frequency (F) and (M) Monetary values. The models produced in this study used this RFM approach and also assessed whether including temporal, product, and other customer transaction information assisted the XGBoost classifier in making better predictions.

The classification models were constructed by extracting each customer's RFM values and transaction information from a Fast Moving Consumer Goods dataset. Different variations of CLV were calculated through one- and two-dimensional K-means clustering of the M (Monetary), F and M (Profitability), F and R (Loyalty), as well as the R and M (Burgeoning) variables. Two additional CLV variations were also determined by isolating the M tercile segments and a commonly used weighted-RFM approach. To test the effectiveness of XGBoost in predicting future timeframes, the dataset was divided into three consecutive periods, where the first period formed the features used to predict the target CLV variables in the second and third periods. Models that predicted if CLV dropped to a lower value from the first to the second and from the first to the third periods were also constructed.

It was found that the XGBoost models were moderate to highly effective in classifying future CLV in both the second and third periods. The models also effectively predicted if CLV would drop to a lower value in both future periods. The ability to predict future CLV and CLV drop in the second period was only slightly better than the ability to predict the future CLV in the third period. Models constructed by adding additional temporal, product, and customer transaction information to the RFM values did not improve on those created that used only the RFM values. These findings illustrate the effectiveness of XGBoost as a predictor for future CLV and CLV drop, as well as affirming the efficacy of utilising RFM values to determine future CLV.

Acknowledgements

I would like to thank Professor Sonia Berman for her guidance and patience in helping this "old guy" finish his dissertation.

I want to thank my daughters Ella and Cara. Until this point in your lives, you had not known a dad who was not studying for many weekends during the year. You can now stop being surprised when you find out I am going with you and your mother to do things during the weekend. This was truly one of the most difficult things about this degree. Ella, you are so small yet so understanding, and I have learned so much from you about what it means to be content with what you have. Cara, I don't think you will remember this time, but I hope you continue to experience the world with such exuberance. I love you both very much. And remember: "The more that you read, the more things you will know. The more that you learn, the more places you'll go." – Dr Suess

Most importantly, Ilze. Learning information technology and machine learning concepts is hard. However, they are easy compared to the sacrifice of time lost with you as your husband and co-parent. When we decided that I would pursue this Master's degree, I don't think either of us understood the implications of that decision. No one can ever question your devotion to me as a life partner because actions, in your case, sacrifice, speak louder than words. So even though these are just words, I hope my future actions will be enough to show my gratitude for enabling me to complete this degree. Without you as my partner, I don't think it would have been possible. I love you always.

Definition of terms

Analytical Hierarchy Process (AHP)
A method that assists with complex decision-making. It uses experts to assist with the calculation of weight coefficients attributed to the factors that take part in the construction of a model [1].

Burgeoning Cluster
A cluster that is created from the Recency and Monetary values through the implementation of the K-means clustering algorithm.

CLV
"The present value of the future profit stream expected over a given time horizon of transacting with the customer" [2].

CLV Drop
A boolean value attributed to customers who moved to a lower CLV segment in future periods.

Customer Relationship Management (CRM)
"A strategy that companies use to build customer relationships, increase sales, improve customer service, and increase profitability" [3].

Feature
An input variable that is used by a machine learning algorithm to make predictions [4].

Fast Moving Consumer Goods (FMCG)
"Goods that have a short shelf-life because they are purchased frequently, consumed rapidly, are priced low and are sold in large quantities" [5].

Frequency
The number of transactions in a specific time interval.

Loyalty Cluster
A cluster that is created from the Recency and Frequency values through the implementation of the K-means clustering algorithm.

Monetary value
The amount of money spent during a specific period.

Monetary Cluster
A cluster that is created from the Monetary value through the implementation of the K-means clustering algorithm.

One-hot encoding
Transformation of categorical features to number arrays that machine learning algorithms will

Principal Component Analysis (PCA)
Principal Component Analysis (PCA) reduces dimensionality (number of features) in large datasets while preserving as much information as possible.

Profitability Cluster
A cluster that is created from the Frequency and Monetary values through the implementation of the K-means clustering algorithm.

Recency
The difference between the present time and the last purchase. The shorter the purchase time, the higher the Recency.

RFM Model
An analysis tool that makes use of Recency (R), Frequency (F) and Monetary value (M) to extract customer information for further analysis.

Segment
Sorting customers into distinctive groups which share the same characteristics to facilitate marketing or CRM strategies.

Synthetic Minority Oversampling Technique (SMOTE)
A technique that assists with situations where a minority class has insufficient training examples for a model to learn from through the creation of synthetic samples [6].

Weighted-RFM
A composite score that is created from the Recency (R), Frequency (F) and Monetary (M) values, where a different scaling weight is assigned to each of the variables.

List of figures

Figure 1. Traditional variables used to segment customers (p.15)

Figure 2. Segmentation of customers according to a Recency (R) and Frequency (F) Grid (p.19)

Figure 3. Box plot method in relation to the normal distribution curve (p.22)

Figure 4. An example of one-hot encoding (p.23)

Figure 5. Representation of the basic SMOTE process (p.24)

Figure 6. Steps involved in the K-means clustering process (p.25)

Figure 7. Graphical representation of the Elbow method (p.26)

Figure 8. The basic structure of a decision tree (p.28)

Figure 9. Transfer of information from a table to a basic decision tree (p.28)

Figure 10. Using the Gini impurity score to determine the leaves on a decision tree (p.29)

Figure 11. Calculation of the residual in an initial gradient boosting decision tree (p.30)

Figure 12. The first gradient boosted decision tree with output values (p.31)

Figure 13. Calculation of the residual in the second gradient boosting decision tree (p.31)

Figure 14. An example of how residuals and splits are created in the first steps of an XGBoost classification problem (p.32)

Figure 15. Example of a branch of an XGBoost decision tree showing the similarity scores used to calculate gain (p.33)

Figure 16. An example of how the new base_score for XGBoost is calculated (p.34)

Figure 17. Example of a Classification report (p.36)

Figure 18. An example of a Confusion matrix (p.37)

Figure 19. A Log Loss curve representing a model with good fit (p.38)

Figure 20. A ROC graph (p.39)

Figure 21. Two ROC graphs depicting the AUC for two models (p.40)

Figure 22. The CRISP-DM data mining life cycle (p.41)

Figure 23. Total sales value per week (p.48)

Figure 24. Total baskets per week (p.48)

Figure 25. Total Sales Value per household for the entire timeframe (p.49)

Figure 26. The Elbow method through the use of a regular loop and plotly (p.52)

Figure 27. Representation of Elbow method with the Yellowbrick library (p.52)

Figure 28. The cumulative variance of principal components (p.59)

Figure 29. Distribution of households by Loyalty clusters (p.61)

Figure 30. Distribution of households by Profitability clusters (p.63)

Figure 31. Distribution of households by Burgeoning clusters (p.64)

Figure 32. Confusion matrix for the Group 3 Monetary Cluster (C) (p.66)

Figure 33. Confusion matrix for the Group 3 Terciles (T) (p.66)

Figure 34. Number of households by Recency (in Appendix) (p.86)

Figure 35. Number of households by Frequency (in Appendix) (p.86)

Figure 36. Number of households by Monetary value (in Appendix) (p.87)

Figure 37. Confusion matrix for the Group 4 Burgeoning Cluster (C) (in Appendix) (p.87)

List of tables

Table 1. Categories proposed by Landis and Koch to evaluate the Cohen's Kappa metric (p.36)

Table 2. Description of data contained in transaction_data table (p.42)

Table 3. Description of data contained in the product table (p.43)

Table 4. Description of data contained in coupon_redempt table (p.43)

Table 5. Numpy, Matplotlib, Plotly, Scipy modules and functions used for data manipulation (p.46)

Table 6. Pandas functions used for data manipulation (p.46)

Table 7. Imbalanced-learn and XGBoost modules and functions used to construct models (p.47)

Table 8. Scikit-learn modules and functions used to construct models (p.47)

Table 9. Yellowbrick modules and functions used to construct models (p.48)

Table 10. Division of Groups (p.49)

Table 11. Comparison of variables before and after pruning (rounded to the nearest integer) (p.51)

Table 12. Average silhouette score for Group 3 clustering (p.53)

Table 13. Monetary value maximum and (standard deviation) for each tercile by Group (p.53)

Table 14. Number of households per RFM segment by group (p.54)

Table 15. The number of households in the final CLV Dropper experiments (p.55)

Table 16. Final classification experiments conducted, and their abbreviated names (p.56)

Table 17. CLV monetary Clusters (C) in target group 3 (and group 4 in brackets) (p.60)

Table 18. Loyalty clusters in target group 3 (and group 4 in brackets) (p.61)

Table 19. Profitability clusters in target group 3 (and group 4 in brackets) (p.62)

Table 20. Burgeoning (B) clusters in target group 3 (and group 4 in brackets) (p.63)

Table 21. Results of the CLV classification experiments (p.65)

Table 22. Feature importance by gain for the R CLV classification experiments (p.68)

Table 23. Feature importance by gain for the A CLV classification experiments (p.69)

Table 24. Results of the CLV Drop experiments (p.71)

Table 25. Feature importance by gain for the R Drop classification experiments (p.72)

Table 26. Feature importance by gain for the A Drop classification experiments (p.73)

Table of Contents

Plagiarism declaration 1

Abstract 2

Acknowledgements 3

Definition of terms 4

List of figures 6

List of tables 8

Table of Contents 9

1. Introduction 12

1.1 Introduction 12

1.2 Problem Statement 13

1.3 Significance of the study 13

1.4 Scope 14

1.5 Organization of this study 14

2. Literature review 15

2.1 Customer Lifetime Value 15

2.1.1 Segmentation and Customer Relationship Management 15

2.1.2 Customer Lifetime Value 16

2.1.3 Problems with predicting CLV 17

2.2 The Recency, Frequency and Monetary Value (RFM) model 18

2.2.1 Using RFM to calculate CLV 18

2.2.2 Advantages and disadvantages of the RFM model 20

2.3 Machine learning techniques 20

2.3.1 Data preparation 21

2.3.1.1 Outlier removal 21

2.3.1.2 Data standardisation 22

2.3.1.3 One hot encoding 23

2.3.1.4 Principal component analysis (PCA) 23

2.3.1.5 Synthetic Minority Oversampling Technique (SMOTE) 23

2.3.2 Customer segmentation through clustering 24

2.3.3 Predicting customer segments with XGBoost 27

2.3.3.1 Background and motivation for the use of XGBoost 27

2.3.3.2 Classification through XGBoost 28
2.3.4 Final model construction with Train_test_split 35
2.3.5 Hyperparameter optimisation and Cross-Validation 35
2.3.6 Classification evaluation metrics 36
2.3.6.1 Cohen's Kappa metric 36
2.3.6.2 Classification report and Confusion matrix 36
2.3.6.3 Learning curves 38
2.3.6.4 ROC/AUC curves 38
2.4 Conclusion 40
3. Methodology and Implementation 41
3.1 CRISP-DM 41
3.2 Dataset selection 42
3.3 Timeframes and target variables 43
3.4 Data preparation 45
3.5 Learning and Evaluation 45
3.6 Coding language and development environment 45
3.7 Determining the timeframe for the groups 48
3.8 Isolation of R F M variables 50
3.9 Outlier detection and removal 50
3.10 Creating CLV values 51
3.10.1 Clusters 51
3.10.2 Tercile groups 53
3.10.3 Weighted-RFM Segment 54
3.11 Creating the Drop variables 54
3.12 Extracting product and temporal purchase data 55
3.13 Classification experiments with XGBoost 56
3.13.1 SMOTE and scale_pos_weight 57
3.13.2 Hyperparameter optimisation 57
3.13.3 Dimensionality reduction in Added Features experiments 58
3.13.4 Evaluation metrics and the final model 59
4. Results 60
4.1 Clustering results 60
4.1.1 Monetary Clusters 60
4.1.2 Loyalty clusters 61

4.1.3 Profitability clusters 62
4.1.4 Burgeoning clusters 63
4.2 Classification Experiments 64
4.2.1 CLV 64
4.2.2 CLV Drop 71
5. Conclusion 75
5.1 Summary 75
5.2 Findings 75
5.3 Recommendations for Future research 76
6. References 77
7. Appendix 86

1. Introduction

1.1 Introduction

Modern information technology and the advent of data mining have enabled firms to employ techniques where the specific characteristics of every customer can be analysed. This analysis is then used to develop strategies to maintain a healthy business-client relationship with each customer [7]. This process is referred to as Customer Relationship Management (CRM). The rise in the importance of CRM [8] is accompanied by an equally significant increase in machine learning techniques employed by firms [9, 10]. At the convergence of these two fields, ample opportunity exists [11] to improve our ability to deliver more effective CRM strategies and expand on the scope of what machine learning models can offer.

The well-known CRM metric Customer Lifetime Value (CLV) measures the present value that a client will provide to a given company while they are an active customer [2, 12]. The main focus of this study is the prediction of future CLV segments through the development of machine learning models.

The first step in constructing these models will be to segment customers in a dataset to reflect CLV. This study will isolate multiple CLV segment variations through adapted traditional and modern machine learning approaches. The well-known RFM method determines the CLV for a customer through the isolation of the most recent purchase (Recency), how frequently a customer visited a retailer during a given period (Frequency) and how much they spent during this period (Monetary). The Recency (R), Frequency (F) and Monetary (M) variables used in the RFM approach will form the basis of the CLV segment variants used in this study. CLV variations will be created by implementing the K-means machine learning clustering algorithm with the RFM variables as the input dimensions. In addition, the Monetary value (M) will also be grouped according to quantiles. Lastly, an adaptation of the approach using weighted-RFM to calculate CLV will also be used.

The second step is to construct machine learning classification models with the use of XGBoost. XGBoost is a gradient boosting library which contains more than a single algorithm [13]. Current knowledge about customers will be used to predict the future CLV segments isolated in the first step above. Models that predict if a customer will drop to a lower CLV segment in the future will also be constructed. XGBoost was chosen to make these predictions due to its recent popularity and ability to function effectively in various use cases [14-17]. The models will then be evaluated, and conclusions drawn from the results of the predictions.

Many studies incorporating machine learning methods can be found that predict current CLV segments. However, articles that predict future CLV segments are far less abundant. The lack of research to predict if customers will move to a lower CLV segment in the future is even more pronounced. Of the papers surveyed for this study, none focused on the use of machine learning to specifically address the binary classification problem of whether customers would drop to lower CLV segments in future.

1.2 Problem Statement

This study is primarily aimed at individuals interested in extracting improved value from customer databases to improve their CRM practices. The data was sourced from a ‘brick-and-mortar’ Fast Moving Consumer Goods (FMCG) database.

With the worldwide decline in retail store visits, decision-makers in the FMGC industry will also be interested in the results of this study [18]. Traditional retail endpoints should use up-to-date techniques such as machine learning if they want to compete with online retailers. Many retailers also use a hybrid model that combines store visits and online sales. As such, the findings in this study could also assist data professionals in determining if similar models can be created from hybrid datasets containing brick-and-mortar and online store-related data.

The general goals of this study are to:

a) Determine how effective XGBoost classifications models are at predicting future CLV segments.

b) Determine how effective XGBoost classifications models are at predicting if a customer will drop to a lower CLV segment in the future.

c) Isolate two data sets as input for the models and compare how effectively each predicted CLV segments. One dataset will contain only the RFM values, while the second will include the RFM, temporal, product, and other customer transaction information.

d) To determine the effectiveness of the above models in predicting CLV segments for a period that does not follow directly after the one used as input features.

1.3 Significance of the study

Predicting the future CLV for a customer is particularly relevant because firms can use the data to plan marketing campaigns aimed at specific customer segments. The assumption that the current CLV of a customer will stay the same in the future introduces a measure of uncertainty. It is this uncertainty that this study will attempt to address. It would be beneficial to know with a higher level of certainty what CLV segment a customer will be in in the future. Improving targeting accuracy can make a substantial difference in the return of a firm’s marketing investments [19].

In addition, the models can also be used to determine the effectiveness of CRM strategies by comparing the predicted CLV and the actual CLV after said strategies. As a result, more effective CRM can be produced through iteration, leading to higher financial returns.

Predicting which customers would likely move to a lower CLV segment can serve as a direct input for a targeted CRM marketing campaign to reduce such ‘drops’. As with CLV prediction, the models can also be used to determine how effective CRM strategies are by comparing the predicted and actual occurrences of CLV segment drop.

Firms that can effectively leverage the power of data and machine learning in their CRM strategies could ultimately achieve a competitive advantage that will enable them to outperform those who do not employ these powerful methods [7].

1.4 Scope

CLV segments will be determined through RFM values. Many other methods are available to calculate CLV, but these fall outside the scope of this study.

The dataset used to construct the final classification models will include the isolated RFM variables and additional information about the time of purchase, product categories, and the number of different stores visited during a given period. The final models do not include customer demographic data because the subset of the data that contained this information was not large enough to construct effective models. The lack of customer demographic data in this dataset is an example of the general lack in the availability of appropriate customer transaction data, which also contains demographic information.

One variable indicating if customers used a coupon will be included, but no advanced marketing information will be considered. Accordingly, this study does not draw conclusions about the effect of marketing activities on the change in CLV over time.

The data used is extracted from a brick-and-mortar FMCG retail database. Therefore, the reader should not directly extrapolate the findings to service or business-to-business industries since they have vastly different characteristics from the FMCG industry.

The products were also not purchased online but from direct store visits. RFM values extracted from online purchases might display patterns different from direct store visits. Therefore, the reader should not assume that the models apply to online channels. However, the models can form the base for additional research that will make use of datasets containing data from online and store visits.

This study also does not make recommendations on the implementation of the models in CRM and marketing strategies. Additionally, it does not give information or recommendations on how to put the models into production.

1.5 Organization of this study

Chapter two provides background information about CLV and the machine learning techniques and methods used in this study. Chapter three will discuss the methodology to construct the models and how this was implemented to form the final models. In chapter four, the results of the K-means clustering and XGBoost classification models will be provided and discussed. Finally, in chapter five, the report will end with a summary of the work and provide recommendations for future research.

2. Literature review

This chapter is divided into two major sections. The first section will provide the background of CLV as a Customer Relationship Management (CRM) tool and, more specifically, the Recency, Frequency, and Monetary (RFM) model's role in determining CLV. The section starts with Customer segmentation and how it fits into the modern practice of CRM and CLV. Then an overview of determining CLV through the RFM approach and the advantages and disadvantages of the RFM approach are provided. The second part of this chapter explains the machine learning techniques and methods used in this study.

2.1 Customer Lifetime Value

In order to elaborate on the use of CLV, it is helpful to understand the vital role that segmentation plays in marketing and modern CRM practices. This section starts with a fundamental background on traditional segmentation before moving on to CLV.

2.1.1 Segmentation and Customer Relationship Management

With the advent of mass marketing between 1880 and 1950 in the United States, professionals in the marketing industry started to sort customers into groups called segments [20]. In this strategy, called Segmentation, the attributes of the segments are defined by the marketing professionals themselves.

In The Principles of Marketing [21], the authors describe marketing segmentation as "dividing large heterogeneous markets into smaller groups or segments to address each group's needs and wants". Marketing can be made more effective through product design aimed at specific segments, or more relevant costing approaches that better suit the requirements of each segment can be developed. Campaigns can also be finely tuned to reach specific segments for a higher return on marketing investment. The need to spend marketing funds on the correct customers is illustrated in studies showing that in some sectors, 45% of the revenue, and 70% of the profit, are generated by 15 % of customers [22].

Traditionally marketers have used various variables in combination, and on their own, to segment customers according to their characteristics, as Kotler, et al. [21] described in Figure 1.

Behavioural	**Demographic**		**Geographic**	**Psychographic**
Occasions (when buyers get the idea to buy, buy or use item)	Age	Occupation	World region or country	Social class
	Gender	Education	Province or state	Personality characteristics
Benefits sought	Family size	Religion	City or metro area	Lifestyle
User status (non-users, ex-users, potential users, regular users)	Family life cycle	Race	Density	
	Income	Generation	Climate	
	Nationality			

Figure 1. Traditional variables used to segment customers [21]

A significant shortcoming with the traditional segmentation strategy was that the needs of individual customers could not be addressed. It was too expensive and impractical, and the data was unavailable to develop methods to provide individualised strategies for each customer [7]. However, the challenges associated with addressing each customer's needs have been significantly reduced due to the advances in Information Technology and the internet. This personalised form of customer communication became known as Customer Relationship Management (CRM) and has become an integral part of many companies' strategies to create a sustainable advantage [7]. CRM enables companies to segment clients based on individual characteristics and not only the commonly used characteristics described in Figure 1.

By the latter part of the 1990s, the scope of CRM had grown to include many other enterprise departments, including resource planning, marketing, shipping and many more [23]. Research by Gartner indicated that in 2017 the global CRM software market became the most significant software market in the world [24]. The 2021 global CRM market was estimated to be USD 43,7 billion and expected to maintain a compounded annual growth of 10,6% from 2021 to 2028 [8].

As indicated by Fayerman [25], who divided the CRM ecosystem into operational, analytical, and collaborative components, CRM has the potential to produce data that can be analysed to develop beneficial strategies. Furthermore, he stated that a failure to develop the analytical component centred around the mining of this data would have a detrimental effect on the CRM strategy employed by an organisation.

In order to determine if a CRM strategy is providing the desired results, companies will use key performance indicators such as Customer Lifetime Value (CLV) [7]. CLV is the value that a company expects from a specific customer. It should not be confused with Customer Equity, the total value a firm expects to derive from its entire customer base [26].

2.1.2 Customer Lifetime Value

Customer Lifetime Value (CLV) was defined by the marketing pioneer Kotler [2] as "the present value of the future profit stream expected over a given time horizon of transacting with the customer". Using CLV as a segmentation tool has become widespread in many industries, especially as an analytical component [27]. CLV is also commonly used as segments rather than interacting with individual customers' CLV [28]. Interacting with customers as CLV segments realises many benefits that would not be possible through individual interaction [29].

In order to produce a CLV segment, the CLV of each customer is firstly determined, and then they are grouped according to specific CLV segments [30]. Gupta, et al. [31] confirmed a positive link between the value of a firm and its customer's CLV. In another study, a firm increased its revenue by a factor of 10 by reallocating its resources to focus on a CLV-based strategy [19].

The traditional importance of CLV as a business metric is further illustrated by its use as a:

- driver of shareholder value as a market-based asset [32].
- measurement device of the positive contribution of marketing efforts to a firm [33].
- resource allocation strategy tool that maximises profit generation [34].
- input to customer retention strategies [26].
- input to determine customer churn rate [35].
- a customer segmentation tool [26].

One long-standing method to calculate CLV is to isolate the recency of the last purchase, the frequency of purchases, and the monetary contribution of a specific customer. First introduced by Hughes [12], these values are then combined to form what is commonly known as the Recency, Frequency and Monetary (RFM) model. Some researchers have stated that the RFM model "may be the most powerful and simplest technique for generating knowledge from CRM data" [36]. Others have proven that the RFM variables can effectively predict CLV [37]. CLV, especially when calculated using the RFM model, is often used as a customer characteristic to segment present customers [38-41]. Section 2.2 will provide more details about the RFM model.

Another well-known method of predicting CLV is to make use of probability models. The probability of future purchase behaviour is calculated using the preceding period's behaviour [42]. Methods include linear regression [43], NBD Pareto model [44], extended Pareto/NBD model [45] and Markoff Chains [46]. Predicting future CLV has especially gained traction with the increase in data mining and machine learning using computer science models [43].

Also of importance to this study is another alternative that implements the quantile method to determine CLV. Miglautsch [47], who initially proposed this method, stated that using quantiles has the disadvantage of grouping customers who do not have the same shopping behaviour while breaking apart segments of customers that might have similar behavioural characteristics. However, this method is easy to understand and implement.

2.1.3 Problems with predicting CLV

CLV models are challenging to implement, given the problems with projecting individual purchase behaviour [48]. Early research noted that forecasting CLV could be problematic in some industries where the relationship with the specific client is very short. Bell, et al. [49] stated that it is doubtful that models using CLV can be extended to FMCG industry categories and that CLV is more appropriate for industries such as financial services or business-to-business relationships. An example of the effectiveness of segmenting customers by CLV in longer term relationships is given by Dursun and Caber [50].

It should be noted that the lifetime part of CLV is not the entire lifetime that a client will be associated with a business but rather a smaller specific period detailing the purchase behaviour [49]. With this in mind, another problem highlighted is the difficulty associated with determining what length of time should be used when assessing CLV [50]. Mulhern [48] stated that using longer timeframes is more resilient to customer behavioural anomalies but has the negative characteristic of losing relevance due to the data becoming outdated. Others agreed that customer databases help examine the current aspects of consumer behaviour, but it is doubtful that the same could be done for future long-term predictions [49].

2.2 The Recency, Frequency and Monetary Value (RFM) model

Recency, Frequency and Monetary values are used to construct the RFM model. A customer is considered to have a high CLV if all three values are higher than other customers' values [51].

Cheng and Chen [52] defined the variables as:

> Recency (R): This is the recency since the last purchase, i.e., the difference between the present time and the last time a purchase was made. The shorter the purchase time, the higher the Recency value will be.
>
> Frequency (F): The number of transactions in a specific time interval.
>
> Monetary value (M): This is represented by the money spent during a given period.

The RFM model is considered a very effective and easy-to-understand modeling technique that can be used to gain insights from CRM data [36]. RFM models are commonly used for customer segmentation and analysis, and there has been ongoing academic interest in this model [29]. The RFM model has been used to segment customers in diverse industries such as "health and beauty, textile, outfitter, healthcare, hairdressing, banking and tourism" [29].

Generally, customers with higher RFM scores also generate the most profit for a company [53]. Other studies have also found that customers with higher RFM scores will conduct more transactions and generate more income for a company [54]. Another point highlighting the models' practical usefulness is that customers with higher RFM scores are usually more responsive to marketing promotions [55].

The R, F and M variables in their own right can also provide helpful CRM information about a customer. Studies have shown that higher R and F values are associated with customers that are more likely to produce additional transactions [56]. Peker, et al. [29] stated that a higher value of R would indicate a higher likelihood of a customer revisiting a retailer in the future and used F to measure customer loyalty. Conversely, Wu, et al. [38] used M to indicate loyalty.

As the use of the RFM model has gained popularity, many researchers have tried to augment the model with additional components. However, others have excluded some of the variables based on the nature of the product or service [29]. Some examples of additional variables that are used are Length [52, 57], Average Item value [58], Group RFM [59], Profit [60], Periodicity [29], and RFM per product [61].

2.2.1 Using RFM to calculate CLV

The RFM variables need to be transformed into quantifiable measures in order to be used to determine CLV. The commonly used approach, referred to as the 'customer quantile method' [50], arranges the values from low to high and then groups them into quantiles. For example, if 20% quantiles are used, the top 20% in the frequency column would be in quantile 5. Doing this for all three values creates a 5x5x5 grid.

Different combinations of the RFM variables can then be used with a grid method to represent proposed customer characteristics. The resulting values can then be grouped again to create low, medium and high segments. For example, in Figure 2, the R and F values are used to segment customers according to possible characteristics.

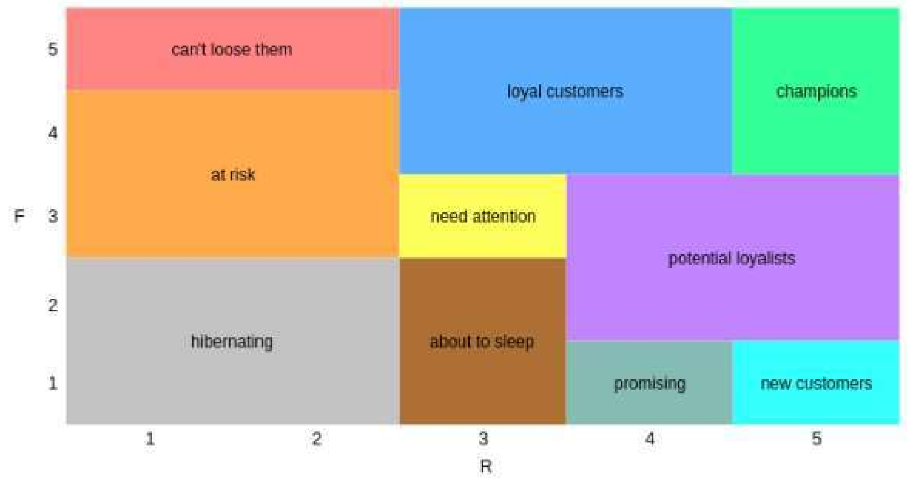


Figure 2. Segmentation of customers according to a Recency (R) and Frequency (F) Grid [62].

Alternatively, the sum of the three quantile values can be used as a composite value to form different segments. For example, if the RFM quantile values were 3, 2 and 3, the total would be 3 + 2 + 3 = 8. If three segments were used, with low: 3 to 7, medium: 8 to 12, high:13 to 15, then the example of 8 would be considered to have a medium CLV.

In the original RFM model, Hughes [12] combined the three variables to create a composite score for CLV, with all three variables assigned an equal weight [12]. After that, Stone [63] assigned different weights to each of the variables, depending on the characteristics of the business and the type of analysis. Assigning different weights to the variables is sometimes referred to as the weighted-RFM method [64].

For the weighted-RFM model, various authors have supplied options on how to decide what weighting should be assigned for each of the three RFM variables. For example, Stone [63] suggested ordering the variables by importance, starting from F, then R, and finally M. Other researchers suggested R as the most important because more recent clients have more potential for growth in the period under question [47]. Others gave M the highest value, while R received the lowest [65]. The biggest objection to scoring the variables in advance is that it is only based on prior generalised knowledge about the business sector [66].

A popular alternative to using pre-defined scaling factors is to scale the variables using industry experts' input [27, 67-69]. This method, known as the analytical hierarchy process (AHP), is still widely used but has been refined since its inception [40]. One of the most significant drawbacks of the AHP process is that it introduces bias and judgement errors due to the human factor [70]. In order to minimise this bias, some researchers have used the Fuzzy Analytical Hierarchy Process (FAHP) method, which considers the ambiguity of the variables [58].

An example of a study using the AHP process was described by Rahmadianti, et al. [27]. In the study, the sales and marketing divisions of Pharmaceutical and Medical Devices Distribution Companies are asked to provide the importance of the three variables. The relative importance of the three variables was then determined through a pair-wise comparison matrix. Many other studies have used the AHP and Fuzzy-AHP process to scale the RFM variables [27, 30, 58, 68].

2.2.2 Advantages and disadvantages of the RFM model

The extensive use of the RFM model can be attributed to the fact that it is easy to understand and interpret the results [71]. In addition, only a few transaction attributes are required to construct it [72], making it fast and straightforward to implement [73]. Also, as is evidenced by the large number of studies employing the RFM model and its variations, it forms the basis of much research on improving CRM techniques such as segmentation [74].

Hu and Yeh [51] stated that the RFM model is an effective tool for determining CLV, while Fader, et al. [75] showed that it could be used to build a useful CLV model. In addition, it is also shown to be effective in predicting response to marketing activities and increasing company profits [76].

It is also worthwhile to consider some negative aspects associated with the model:

1. There is usually a high correlation between the M and F values, which makes combining these variables less effective [77].
2. The models usually do not consider the influence of a company's marketing efforts, which might influence customer behaviour [43].
3. The model can effectively predict customer behaviour for the period directly after it was constructed. However, to be effective, it also needs to help predict the periods after that initial period [43].
4. It does not consider other user demographics (see Figure 1) commonly used to construct an effective marketing campaign [78].
5. The difference in importance that various industries assign to the three variables, and the bias involved in that process, can create ambiguity [50].
6. If each of the individual R, F and M values are divided into sections, for instance, quantiles, to calculate the combined RFM value, the chosen size of the quantile (terciles, quartiles and the like) might not offer enough granularity to generate the required result [79].

2.3 Machine learning techniques

The importance of machine learning and artificial intelligence in the CRM sphere has increased significantly in the past 20 years. A survey conducted by IDC calculated that artificial intelligence associated with CRM would contribute 1.1 trillion dollars from 2017 to 2021 [9]. Furthermore, according to a report by Zion Market research, the global AI market for CRM was $11,42 billion US dollars in 2017 and is expected to grow to $123.28 billion in 2024 [11]. One of the largest CRM software companies, Salesforce, stated in their 2018 Annual report that they would be integrating machine learning processes across most of the products they offer [10].

One can classify CRM processes into four dimensions to understand where the machine learning methods could help determine CLV. These dimensions, first described by Swift [80] and later repurposed by Chagas, et al. [81], are:

1. Identifying the most profitable customers through classification and segmentation.
2. Optimising customer attraction by removing wasted efforts.
3. Retaining customers through personalised marketing strategies, loyalty programs and compliant management.
4. Developing the relationship with the customer through increased transaction intensity and value offerings.

An example of the usefulness of machine learning in the first dimension is the implementation of unsupervised clustering algorithms to build models that segment customers according to their respective clusters. Classification models using supervised machine learning methods can then be used to predict the probable future CLV of a specific customer in a segment. The information obtained from the first two dimensions can then be helpful to activities pertaining to the third and fourth dimensions. For instance, considering the third dimension, loyalty programs could target customers that are predicted to be in future high and medium segments. In the fourth dimension, customers with a probable future medium CLV could be enticed with special value offerings in order to move them to a higher CLV segment. Using machine learning to directly predict which customers are likely to move from one segment to another will be helpful in the second, third and fourth dimensions.

In a review by Chagas, et al. [81] of a total of 119 journal articles from the Ebsco, IEEE, Science Direct and Emerald databases, 30 had Customer segmentation as a central element, and 16 had CLV as a significant element. They found that the most common machine learning algorithms employed in CRM were XGBoost, Logistic Regressions, Neural Networks, Decision Trees, k-Nearest Neighbour, K-means, and Fuzzy c-means. CLV articles referenced Random Forests, Support vector machines (SVM), K-Means, k-Nearest Neighbour and Neural networks, to mention a few. For the purposes of this study, it is notable that the papers included in the Chagas, et al. [81] review used XGBoost for Customer segmentation and Complaint management.

The following subsections outline data preparation, model creation using K-means clustering [82] and XGBoost [13], and the metrics commonly used to evaluate results.

2.3.1 Data preparation

The input to a machine learning model must be in the correct format conducive to effective modelling. Accordingly, this section will review the methods used in this study to prepare the input data for modelling.

1. Outlier removal

Some machine learning methods, such as clustering, are very sensitive to outliers, and their occurrence might have a detrimental effect on the final output of the model. Therefore, it is essential to remove outliers before the machine learning algorithm uses the data. "An outlier is a data point that differs significantly from the other data points in a given dataset" [83]. For instance, in the set of numbers 1,100,102,100,103, 99, the number 1 would be an outlier.

There are many standardised methods available to remove outliers for the purpose of machine learning modelling. These can include simple methods such as using standard deviation (or z-score) or box plots. More complex methods are DBScan Clustering, Isolation forests and Robust Random Cut Forest [84]. However, only those applicable to this study, namely the use of box plots and removal through z-score, will be discussed.

The standard deviation is the average amount of variability in a dataset. It is a measure of how far each value lies from the mean value if the data is normally distributed. According to the Empirical rule, if data is normally distributed, 68% of data will be within one standard deviation, 95% within two and 99.7% within three.

The z-score of a specific data point is calculated by subtracting the mean from the value and dividing the result by the standard deviation. As seen in Figure 3, a z-score of 1 will indicate that the point is one standard deviation from the mean. A commonly used method of eliminating outliers is to remove all the points that will fall outside a z-score of 3. In some extreme cases, a z-score of 4 can also be used [85].

An alternative method to the z-score is to use box plots. Box plots make use of the interquartile range (IQR). Values are arranged from the lowest to highest value as the first step in calculating the IQR. The values are then divided into quartiles, where quartile 1 (Q1) is the value such that 25% of the data is smaller than the value, and so forth. The IQR is then calculated by subtracting Q1 from Q3 [86].

According to the box plot method, outliers are all the values that fall outside of Q3 + 1.5 X IQR at the top end or Q1 – 1.5 X IQR at the lower end. These limits are generally represented on a graph as lines called whiskers [87]. Figure 3 illustrates the whiskers in relation to the normal distribution graph discussed earlier. For normally distributed data, the whiskers of the box plot would correspond roughly with a 2.698 standard deviation from the mean [86].

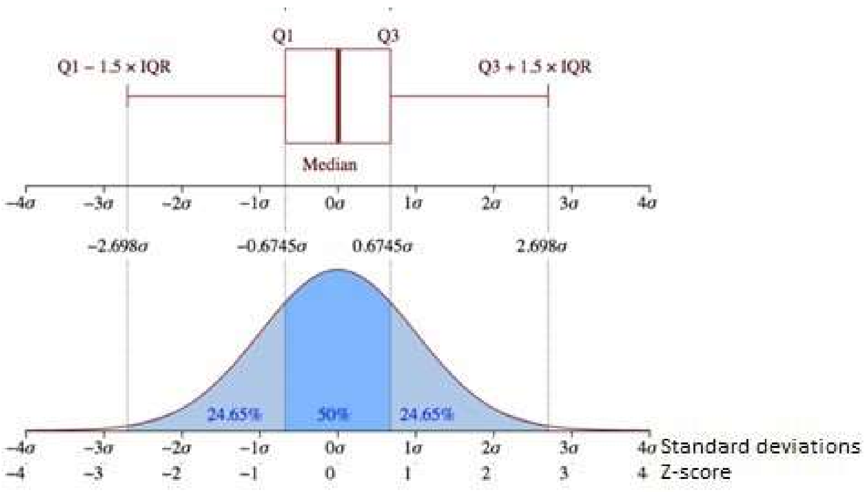


Figure 3. Box plot method in relation to the normal distribution curve [84].

2. Data standardisation

Some machine learning algorithms can be very sensitive to the scale of the data. Therefore, it is common practice to adjust the features used during modelling to minimise the effect of scale [88].

Standardisation, in which features are transformed to be on the same scale (for instance, between 0 and 1), is done to prevent features with large ranges from disproportionally influencing the algorithm. In addition, standardisation can reduce data inconsistencies and make comparing different features easier [89].

Distance-based clustering algorithms that calculate the distances between data points, such as K-means, can be affected severely by the scale of the data [90]. As such, it is vital to normalise data before these algorithms are employed. On the other hand, tree-based algorithms such as XGBoost will not be affected by scale because of how the trees are constructed.

Various methods of scaling data to a standard format can be accessed through the Scikit-learn library [91]. The StandardScaler ensures that for each feature, the mean is 0 and variance is 1, but there will not necessarily be a minimum or maximum value. The MinMaxScaler transforms data so that all points for the selected features are between 0 and 1. The data will thus contain a minimum and maximum value [88].

One negative aspect of the MinMaxScaler is that it does not handle outliers very well [92] and makes it essential to clean outliers from the data before its use. MinMaxScaler performs a linear transformation and uses the following formula to perform MinMax normalisation:

$$v'_i = \frac{v_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A.$$

The attribute to be mapped is v_i. min_A and max_A are the minimum and maximum values [93].

3. One hot encoding

A continuous variable can take on any value between the lowest and highest points of measurement, e.g., speed and distance [94-96]. On the other hand, a categorical or nominal variable can only take on a limited and fixed number of possible values, usually based on some qualitative property such as colour or model number [96]. Many machine learning algorithms (such as XGBoost) cannot function with categorical values and will require the values to be numeric [97]. One hot encoding can transform the data so that each categorial value represents a numerical value.

For instance, consider a series of data points that can be blue or green: [Blue, Blue, Green]. Figure 4 shows how one-hot-encoding will transform this series into features that the machine learning algorithm can use.

Original series

Index	Colour
1	Blue
2	Blue
3	Green

to

New features

Index	Colour_Blue	Colour_Green
1	1	0
2	1	0
3	0	1

Figure 4. An example of one-hot encoding

4. Principal component analysis (PCA)

Principal Component Analysis (PCA) reduces dimensionality (number of features) in large datasets while preserving as much information as possible. PCA does this by transforming the original dataset to create new variables, uncorrelated to each other, which maximise variance [98]. Because the transformed dataset will contain less information, it could reduce accuracy. However, the benefits include data that is easier to interpret and a faster algorithm run time [99]. It is important to note that feature standardisation is required before PCA is conducted.

5. Synthetic Minority Oversampling Technique (SMOTE)

Synthetic Minority Oversampling Technique (SMOTE) was first described by Chawla, et al. [100]. This technique was developed to assist with situations where the minority class has insufficient training examples for a model to learn from.

In the context of this study, the minority class can be a segment with far fewer customers than the other segments. Because the segment contains fewer samples than the others, it is more difficult for the machine learning algorithm to learn from the data and, in the case of a classification problem, to classify such samples correctly. To overcome this problem, SMOTE creates synthetic samples from the minority class that is added to the training dataset [6].

These synthetic samples are created by implementing the k-nearest neighbour algorithm and following these steps [101]:

1. The minority class is identified, and a random sample x_i from this class is selected.
2. The k-nearest neighbour algorithm finds its k nearest neighbours that belong to the minority class.
3. A synthetic sample is placed at the midpoint of the line between x_i and each of these k nearest neighbours, as shown in Figure 5.
4. The process is repeated until the classes are balanced.

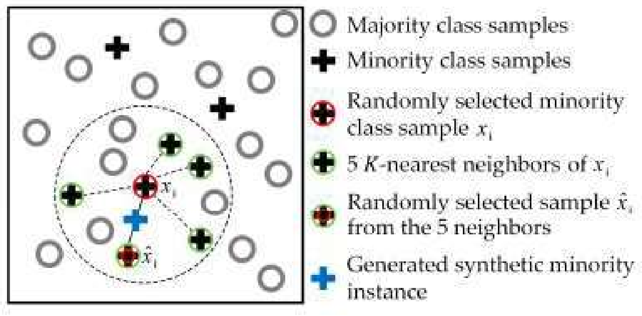

x_i
$\hat{x}_i$
Majority class samples
Minority class samples
Randomly selected minority class sample x_i
5 K-nearest neighbors of x_i
Randomly selected sample $\hat{x}_i$ from the 5 neighbors
Generated synthetic minority instance

Figure 5. Representation of the basic SMOTE process [102].

An essential aspect of SMOTE is that it should only be performed on the training dataset. If SMOTE is performed on the training and testing data, both might contain the same synthetic generated instances [103]. The final classification results for the model will then be artificially inflated, and the model will perform inadequately on new data in a production environment.

2.3.2 Customer segmentation through clustering

Using the RFM model to perform customer segmentation is a powerful method of determining CLV [51]. This segmentation can be accomplished using unsupervised machine learning algorithms broadly grouped as clustering algorithms. These clustering algorithms are called unsupervised learning algorithms because they will function independently, with minimal input from the user to discover the data structure [104].

A clustering algorithm will partition data into disjoint groups where members of a specific group share similar characteristics if compared to members of other groups [29]. Some of the most common clustering methods used in CRM studies using machine learning include K-means, K-Medoids, DBSCAN [105], Fuzzy Gustafson-Kessel [106] and fuzzy C-means [39, 107]. A review of the literature showed that abundant research had been conducted that made use of machine learning clustering to segment customers according to RFM [27, 29, 105, 108].

As early as 2009, Berry, et al. [109] stated that K-means was one of the most commonly used clustering algorithms. A review of the literature confirms this, with many examples readily available [27, 29, 58, 110-112]. Rahmadianti, et al. [27] stated that the popularity of K-means can be attributed to the "ease of implementation and fast execution" of the algorithm. A similar sentiment was expressed by Pramono, et al. [58], who also stated that it is a widely used method in customer segmentation. K-means is a versatile method used in multiple fields, including "data mining, statistical data analysis, pattern recognition, customer segmentation and other business applications" [29]. The K-means algorithm was thus selected for use in this study.

The K-means clustering technique groups a set of observations into a user-defined number of clusters (K). The method minimises variance within the final result [113]. The K-means clustering process is illustrated in Figure 6.

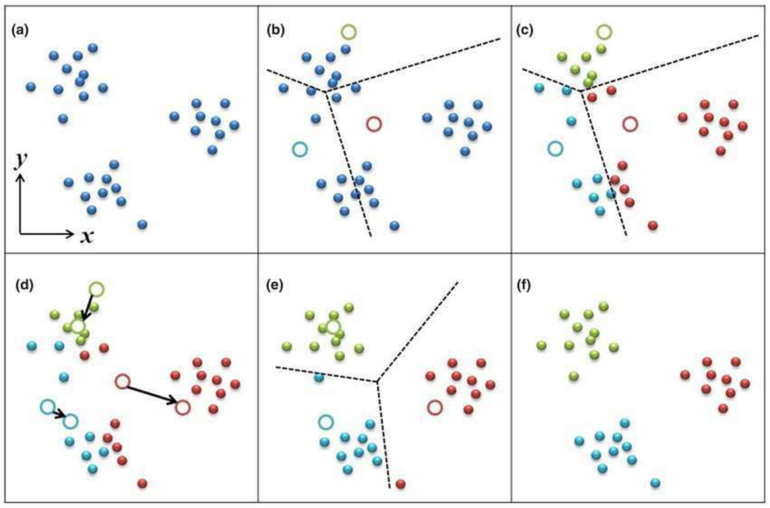


Figure 6. Steps involved in the K-means clustering process [114].

K-means creates clusters through the following steps:

1. The user selects the number of clusters, for instance, K=3, through a pre-determined business objective or other methods.
2. K distinct points on the data continuum are selected randomly from the data series, creating the possible centroids. The possible centroids are illustrated by (b) in Figure 6.
3. The Euclidean distance from each data point to the possible centroids is then determined.
4. Each data point is assigned to its nearest centroid.
5. The process then continues until all the points are assigned to a centroid, forming the first possible set of clusters. In figure 6 (c) the dotted lines representing cluster boundaries are illustrated. The mean is calculated for each group of values assigned to a specific centroid, becoming the new centroids. Then the distance from each point to the mean is measured, and the variance is calculated. The sum of the variances in each of the resulting clusters is recorded.
6. Steps 2 to 5 are repeated for a user-defined number of iterations or until there is no more improvement in the sum of variances.
7. The set of clusters with the lowest summed variance is the final clustering result, as illustrated in Figure 6 (f).

The Elbow method can be used to overcome the problem of a suboptimal choice of K for a specific dataset. This method creates multiple models starting with K = 2, up to a user-defined maximum for K. For each value of K, the sum of squared distances from each point to its assigned centre is calculated and recorded [115]. This final value is known as the Distortion score and is usually the default evaluation metric used in the Elbow method. In some datasets, there will be a clear point where the decrease in the distortion score is significantly less than for previous values of K. This point is referred to as the elbow of the plot and indicates the optimal number of clusters for the specific dataset [115]. Figure 7 is an example of the above process's graphical representation, which usually accompanies the Elbow method to assist the user in deciding on the value of K.

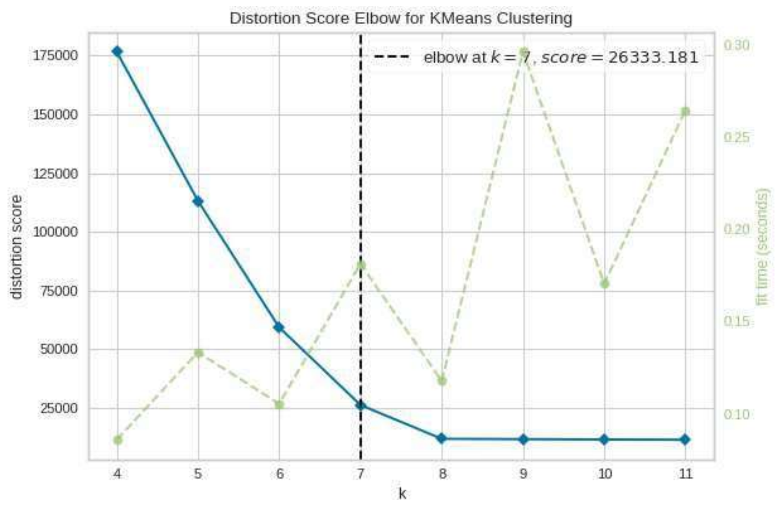


Figure 7. Graphical representation of the Elbow method [115].

As the number K increases, the inertia will always decrease because the values in the series will be closer to their centroids. When choosing the optimal number for K, the desired result is the minimum number of clusters that gives reasonable inertia.

The Elbow method will either rely on user judgement or use the point of greatest inflexion to indicate the elbow point on the graph [115]. This evaluation method can be problematic if clusters are very close, and the Elbow method might not be the correct method to determine K [116]. An alternative, the Silhouette score, can be used to be more precise.

Silhouette analysis is used to study the separation distances between clusters [91]. This uses the silhouette coefficient, where a coefficient of 1 indicates that a sample is far away from the neighbouring cluster. A coefficient of 0 indicates that the sample is very close to the decision boundary to which cluster it is assigned. A negative coefficient indicates that the sample might have been assigned to the wrong cluster [91]. The average silhouette score for each number of clusters can then be used to decide the optimal number of clusters. The Silhouette score is calculated with the formula: Silhouette score = *(b-a)/max(a,b),* where *a* equals the average distance between points in a cluster and *b* equals the average distance between clusters [91].

2.3.3 Predicting customer segments with XGBoost

One of this study's goals is to segment customers according to their CLV variations and predict their future CLV. This goal will be achieved by using XGBoost as a classifier, i.e., predicting which CLV segment or class a specific sample will belong to in future.

In a study similar to this one, Win and Bo [117] noted that predicting future customer segments as a classification problem is more informative than a regression one. In this paper, the authors calculated a continuous CLV using a formula incorporating RFM variables, churn rate and profit margin. Then they grouped the customers into two classes according to high and low CLV. After this, they used a Random Forest algorithm to predict the future CLV class with high accuracy.

Karaman [118] used the K-means clustering algorithm to cluster each customer's Recency, Frequency and Monetary values. A segment that combined all three RFM clusters was also included as a feature for the final model. The author then used these clustered values as the input features to predict future CLV classes with XGBoost. The Monetary value clusters were used to represent the CLV classes.

1. Background and motivation for the use of XGBoost

XGBoost was developed in 2014 and subsequently released as a standalone command-line program by Tianqi Chen [119]. He first presented their research at the SIGKDD Conference in 2016 as "an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable" [13]. The library to use XGBoost can be accessed through the Scikit-learn library. XGBoost is one of the most popular supervised machine learning methods used by data scientists worldwide [120], winning numerous Data Science competitions and challenges [121].

The author of the library describes the main benefits [13]:

- The system is very scalable and can be used in a wide assortment of scenarios due to the wide variety of optimisations or innovations.
- It runs more than ten times faster than other solutions that run on a single machine.
- Ability to effectively deal with sparse data containing missing values.
- Due to its out-of-core computation, it can handle hundreds of millions of examples on the same machine.
- Ease of use in creating an end-to-end system that scales to even larger datasets with a low number of cluster resources.

Furthermore, it outperforms other methods when smaller datasets are used [122].

While only three studies in the Chagas, et al. [81] review used the XGBoost algorithm, and none in the Customer Lifetime Cycle element used it, some examples of CRM techniques that made use of XGBoost are:

- Binary sentiment analysis: 92.5% accuracy compared to 75,7% in the baseline model [14].
- Customer churn: prediction accuracy was higher after customers were segmented. XGBoost performed better than the other tested algorithms, with an accuracy of 88% [15]. This study used a variation of the RFM model and the K-means clustering algorithm for segmentation.
- Cost of new customer acquisition: Monastyrskaya and Soloviev [16] showed that using

XGBoost could predict the cost per acquisition. K-means and the Elbow method were used to create segments in this study.

- Customer demographics prediction: CRM data was used to predict the age and gender of customers. XGBoost classified 89.03% of the genders correctly and performed better than other tested algorithms [17].

As seen from the studies above, using XGBoost as part of a CRM strategy has proven beneficial.

XGBoost is grouped under gradient boosted decision trees [13]. In order to understand the mechanism XGBoost uses to make classifications, it is easier to start with understanding the basics of Decision trees and Random Forests first. Therefore, these algorithms will be discussed in the following section before moving on to more details about XGBoost.

2. Classification through XGBoost

A decision tree uses a tree structure to represent a decision's route to predict a specific outcome [123]. An example is shown in Figure 8.

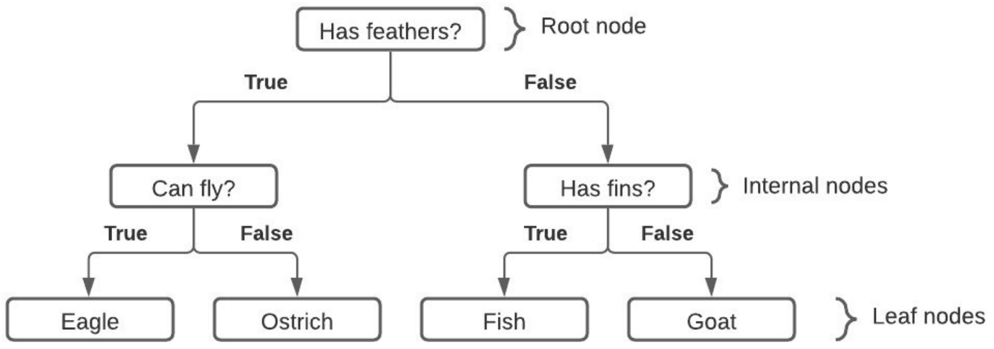
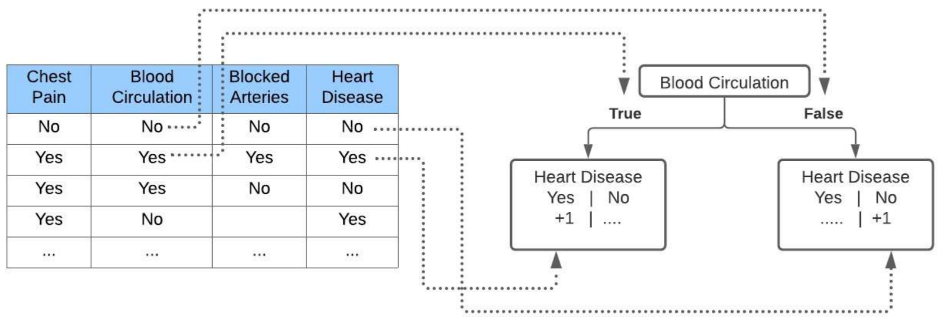


Figure 8. The basic structure of a decision tree

For a classification problem, the leaf nodes would be the classes to predict. The other nodes are derived from the features in the dataset used to make the predictions [88].

The first step in constructing a decision tree for classification would be to determine which feature will be at the root of the tree.



Chest Pain	Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No		Yes
...	...	...	...

Figure 9. Transfer of information from a table to a basic decision tree

In Figure 9, Chest Pain, Blood circulation and Blocked Arteries would be the features, while Heart Disease is the target variable. In order to determine if a feature will be the root node, each entry from the specific feature (e.g. Blood circulation) is evaluated for its ability to correctly predict the target class (Heart Disease) through the Gini impurity score formula:

Gini impurity = 1 – (probability of yes)^2 – (probability of no)^2.

The weighted average of the leaves (2 leaves in this example) is then calculated, resulting in the final Gini impurity for the specific feature. All features are then compared, and the one with the lowest Gini impurity will be the root node. At the next node, the process is repeated until a maximum user-defined depth is reached or there is no improvement in Gini impurity. All the branches should then terminate in a leaf node. The process is illustrated in Figure 10.

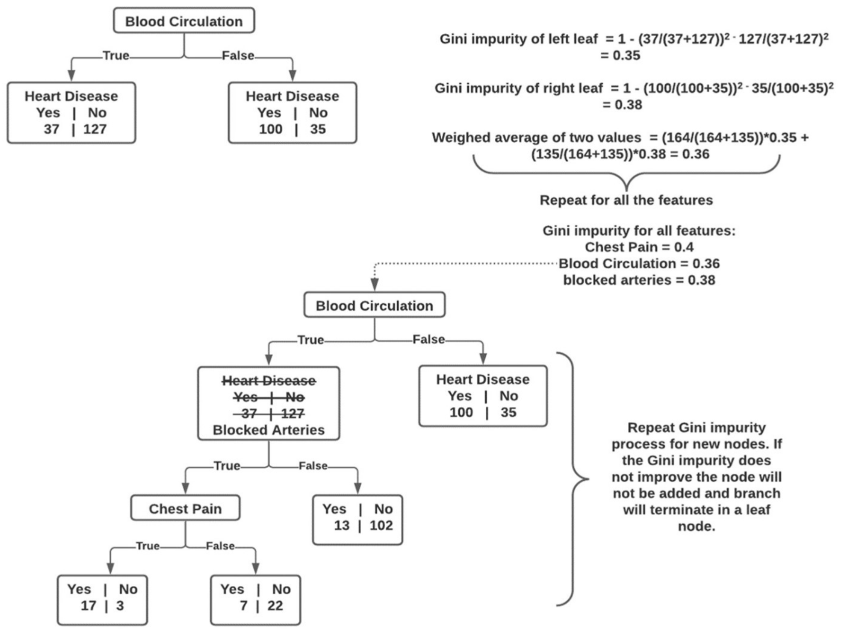

Blood Circulation
True
False
Heart Disease Yes | No 37 | 127
Heart Disease Yes | No 100 | 35
Gini impurity of left leaf = 1 - (37/(37+127))² - 127/(37+127)² = 0.35
Gini impurity of right leaf = 1 - (100/(100+35))² - 35/(100+35)² = 0.38
Weighed average of two values = (164/(164+135))*0.35 + (135/(164+135))*0.38 = 0.36
Repeat for all the features
Gini impurity for all features:
Chest Pain = 0.4
Blood Circulation = 0.36
blocked arteries = 0.38
Blood Circulation
True
False
~~Heart Disease Yes | No 37 | 127~~
Blocked Arteries
Heart Disease Yes | No 100 | 35
True
False
Chest Pain
Yes | No 13 | 102
True
False
Yes | No 17 | 3
Yes | No 7 | 22
Repeat Gini impurity process for new nodes. If the Gini impurity does not improve the node will not be added and branch will terminate in a leaf node.

Figure 10. Using the Gini impurity score to determine the leaves on a decision tree

As described by [124-127], a significant shortcomings of decision trees is that they are very good at making predictions with known data, but not with unseen data. This bias towards the training data is referred to in machine learning as overfitting. In order to mitigate the effect of overfitting, the random forest method was developed. Overfitting is reduced by constructing multiple trees and allowing each tree to 'vote' towards the final prediction. Constructing and combining multiple models to create a more effective one is grouped under ensemble methods [88].

Random forests have proven to be effective on a wide range of datasets for both regression and classification tasks. It is one of the most widely used machine learning methods [88]. The following steps describe how a random forest of decision trees are constructed and how the result is determined:

a) Bootstrapping: Select samples from the original dataset randomly and then add them to a new dataset [123].

b) Create multiple decision trees from randomly selected subsets of features. At each selection, the same number of features will be selected.
c) Input the first sample into each tree and determine the result by voting, i.e. the class with the highest count will be the prediction for that sample. Repeat this process for all the samples in the bootstrapped dataset. Steps a) to c) are referred to as bagging.
d) During bootstrapping, some samples were not included. These remaining samples, the out-of-bag dataset, are run through the process described above. The proportion of incorrectly classified samples is called the out-of-bag error.
e) The process returns to point b), and a different set of random features is chosen. The process is repeated until all possible combinations are checked or a user-defined limit is reached. Then, the combination with the lowest out-of-bag error is usually chosen as the final model.

An alternative to random forests is a gradient boosted decision tree, one of the most powerful and widely used methods in the supervised machine learning space [88]. XGBoost also creates gradient boosted decision trees, but it is unique in several ways.

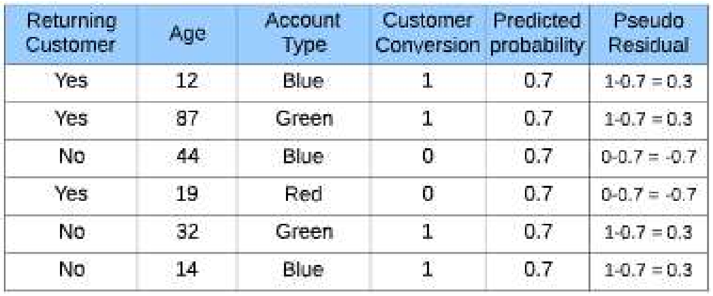
If one makes use of customer conversion prediction as an example, gradient boosted decision trees will follow the following steps to classify samples (where 1 = conversion and 0 = not conversion):

a) The process starts with the leaf nodes that represent the initial prediction. When gradient boosting is used for classification, the predictions are in terms of the log of the odds of being in a specific class. As such, the predictions need to be transformed through the following formula to get the probability:

Probability of positive class = $e^{\log(\text{odds})} / (1+e^{\log(\text{odds})})$

If the initial threshold is set to 0.5, every sample above 0.5 will be classified as a conversion. For this first step, the above formula equals 0.7, and the initial prediction will classify all samples as conversion customers.

b) The quality of the initial prediction is calculated by predicting the Pseudo Residuals for each sample with the formula: Pseudo Residuals = (Observed class – Predicted Class). Figure 11 is a representation of the example dataset and the Pseudo Residual calculation:



Returning Customer	Age	Account Type	Customer Conversion	Predicted probability	Pseudo Residual
Yes	12	Blue	1	0.7	1-0.7 = 0.3
Yes	87	Green	1	0.7	1-0.7 = 0.3
No	44	Blue	0	0.7	0-0.7 = -0.7
Yes	19	Red	0	0.7	0-0.7 = -0.7
No	32	Green	1	0.7	1-0.7 = 0.3
No	14	Blue	1	0.7	1-0.7 = 0.3

Figure 11. Calculation of the residual in an initial gradient boosting decision tree

c) The three features (Returning Customer, Age, and Account Type) are then used to construct a tree to predict the residuals. In constructing the tree, the user will limit the number of leaves. After that, the output value of each leaf is calculated using the transformation formula for classification using gradient boosting:

Output value = Σ Residual_i / Σ[Previous probability_i x (1- Previous probability_i)]

This step is illustrated in Figure 12 below:

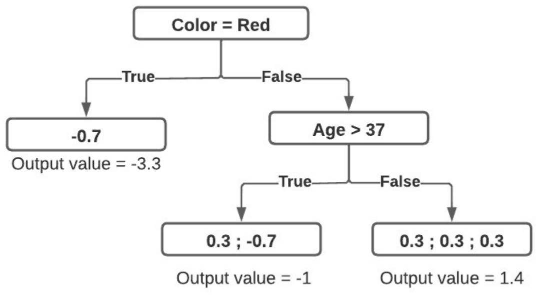


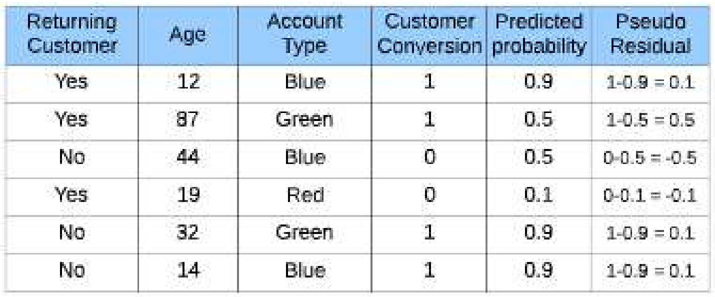
Figure 12. The first gradient boosted decision tree with output values

d) The prediction for each sample is then updated by combining the initial prediction, a user-defined learning rate (0.8 in this example), and the predictions from the constructed tree. For the first entry in Figure 12, the new prediction will be calculated as follows:

log(odds) prediction = 0.7 + (0.8 X1.4) = 1.8

log(odds) converted to a probability = $e^{1.8}$ / 1 + $e^{1.8}$ = 0.9

e) The residual is then calculated again as in b) above and illustrated in Figure 13:



Returning Customer	Age	Account Type	Customer Conversion	Predicted probability	Pseudo Residual
Yes	12	Blue	1	0.9	1-0.9 = 0.1
Yes	87	Green	1	0.5	1-0.5 = 0.5
No	44	Blue	0	0.5	0-0.5 = -0.5
Yes	19	Red	0	0.1	0-0.1 = -0.1
No	32	Green	1	0.9	1-0.9 = 0.1
No	14	Blue	1	0.9	1-0.9 = 0.1

Figure 13. Calculation of the residual in the second gradient boosting decision tree

f) New trees will then be constructed following the steps above and using the residuals until a user-specified amount of trees are created, or the residuals become very small.

A new value not used when the trees were constructed will then be used to test the effectiveness of the constructed model. For this test, the initial prediction value (0.7) is added to all the scaled output values when the new value is entered into all the contracted trees. This log(odds) value is then converted into a probability. If the final answer is > 0.5, the sample is classified as part of the converter class.

XGBoost is an abbreviation for eXtreme gradient boosting. It improves on gradient boosted decision trees in several ways. It is highly efficient, much faster than other existing systems, and highly scalable in almost all scenarios where it can be used [13]. In addition, it is a more regularised form of gradient boosting that uses advanced regularisation through it's hyperparameters [13].

One of the major innovations of XGboost is how the trees are built compared to other gradient boosted approaches. The rest of this section will provide a generalised example of how XGBoost constructs trees and the role that its parameters and hyperparameters play.

For classification problems involving more than two classes (multivariate classification), a boosted set of trees will be constructed for each class through a one-vs-all strategy, i.e., a sample could be in a specific class or in any of the other classes. Figure 14 shows four samples of drug dosages that were either classified as part of class 1 (probability =1) or not part of class 1 (probability =0).

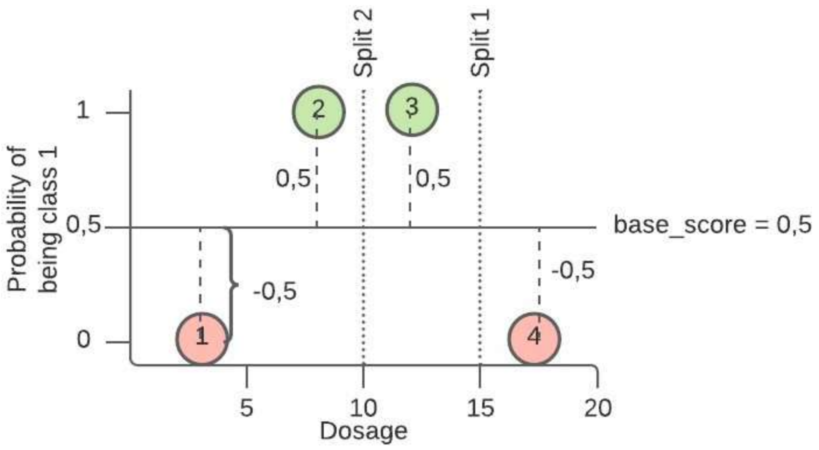


Figure 14. An example of how residuals and splits are created in the first steps of an XGBoost classification problem

With reference to Figure 14, the process of creating trees will then follow the following steps:

1. An initial estimate is set by default to 0.5; this can be adjusted by changing the base_score parameter, but this does not make a significant difference if the model is allowed to go through enough iterations [13].

2. Residuals are determined as the difference between the base_score and the values.

3. All the samples are considered as a single leaf, and a similarity score is calculated for each leaf through the following simplified formula:

   Similarity score = $\sum(\text{Residuals})^2$ +small? / [Previous probability * (1 – Previous probability) + λ]

   The prior probability is 0.5 as set by the base_score at the start.

4. The samples are grouped by splitting them to calculate gain. This gain will determine which split will give the best results. The following is the formula for gain:

   Gain = Left similarity + Right similarity – Root similarity

   For split one, the tree can be illustrated as shown in Figure 15 with calculated gain.

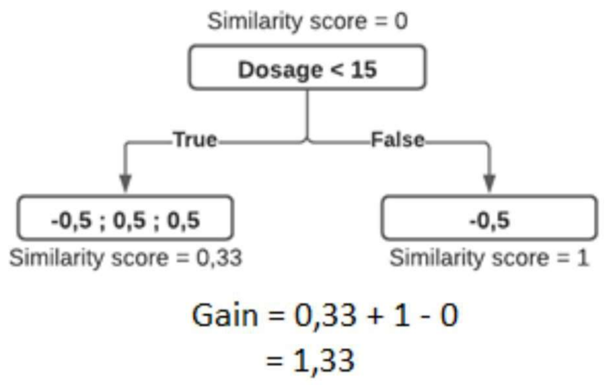


Figure 15. Example of a branch of an XGBoost decision tree showing the similarity scores used to calculate gain

A gain score is then calculated for all the possible splits (for example, split 2), and the split with the highest gain is chosen to be the correct one. The process is then repeated with the next node until all the samples have been allocated to a leaf.

The hyperparameter lambda or λ in the similarity score formula has a default value of 0. Raising this value will decrease the similarity score in each leaf. It is a regularisation hyperparameter because it decreases the influence of individual samples and lowers the likelihood of overfitting the model to the training data.

The tree can then be pruned by adjusting the gamma hyperparameter (ɣ). If the lowest branch has a gain smaller than gamma, it will not add a branch. Because lambda lowers the gain value, it makes branches more likely to be pruned by gamma. Lambda also reduces the influence a single observation will have by reducing the output value (see step 6).

5. The minimum number of samples in a leaf can also be determined by calculating the cover:

   Cover for classification = ∑([Previous probability * (1 – Previous probability)]

   The upper limit for cover is determined by the min_child_weight hyperparameter with the default set to 1. No new nodes will be formed if the calculated cover is less than the min_child_weight value. Similar to gamma, setting the min_child_weight value higher will make the tree more conservative.

6. An output value is then calculated for each leaf through the formula:

   Output value = $\sum \text{Residual}_i$ / (∑[Previous probability * (1 – Previous probability)] + λ)

7. As XGBoost is a gradient boosted decision tree, it will use the predictions from previous trees to generate new predictions. Firstly, the previous probability (base_score) will need to be converted to a log(odds) value through the formula:

   log(odds) = log(p/1-p)

   The previous probability log(odds) will then be added to the new output after it is scaled by a learning rate. The learning rate is adjusted through the *eta* hyperparameter and has a default value of 0.3. The learning rate reduces the amount of influence each tree will have after a boost cycle. The answer will then need to be converted back to a probability with the formula:

Probability = $e^{\log(\text{odds})}/(1 + e^{\log(\text{odds})})$

These new probabilities will then form the bases for the next prediction. With each boosting iteration, new probabilities will be calculated by adding each leaf's base score log(odds).

The first completed tree for the example is shown in Figure 16, for the situation where *lambda* and *gamma* are zero.

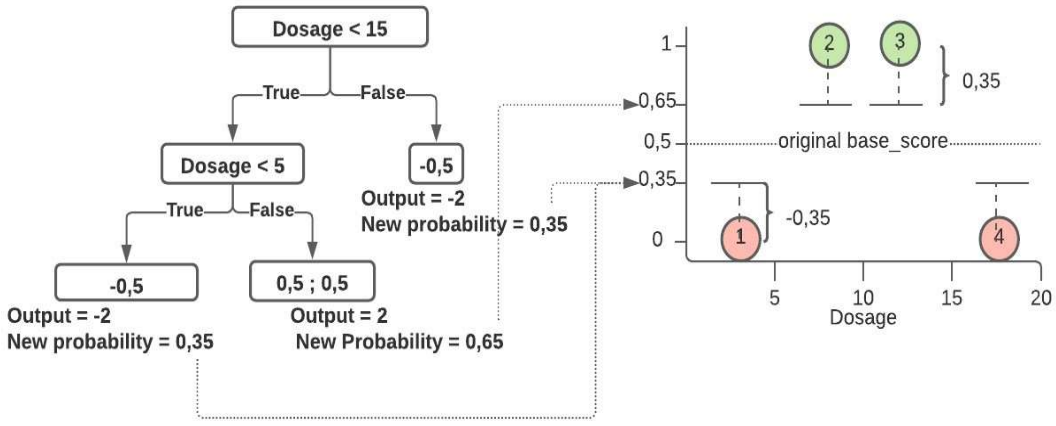


Figure 16. An example of how the new base_score for XGBoost is calculated

8. Steps 2 to 7 will then be repeated, with the residual values becoming gradually smaller as new trees are built through the gradient boosting action. The number of boosting rounds can be adjusted through the *n_estimators* parameter (default = 100).

   The number of boosting rounds can also be limited through the *early_stopping_rounds* parameter. Limiting the number of rounds reduces the possibility of overfitting on the training dataset by stopping the training process if there is no improvement in the test dataset after a given number of boosting rounds [128].

9. For multivariate classification, the objective should be set to *multi:softmax*. XGBoost will use the softmax formula to determine what class each sample will be classified as.

   After the ensembles of trees have been constructed, the samples from the testing or validation dataset are fed into the constructed trees to determine how accurate the trees are on unseen data.

Other important hyperparameters to consider are:

- *max_depth* (default = 6): Specifies the maximum depth of the trees. This value will increase the likelihood of overfitting when raised, due to deeper trees.
- *subsample* (default = 1): "Subsample ratio of the training instances. By setting it to 0.5, XGBoost would randomly sample half of the training data before growing trees, and this will prevent overfitting. Subsampling will occur once in every boosting iteration" [13].
- *reg_alpha* (default = 0): Another regularisation parameter similar to lambda. Raising the value will make the model more conservative.
- *merror*: The evaluation metric for multi-class classification. It is determined by the number

of wrong cases divided by all cases.

- *mlogloss*: The evaluation metric for multi-class negative log-likelihood. Log loss decreases as the predicted probability moves closer to the actual label [91].

2.3.4 Final model construction with Train_test_split

Train_test_split is a technique that is part of the Sckit-learn library and is used to split matrices into random train and test subsets [91]. It is usually used to produce the final model after hyperparameter optimisation and cross-validation have been conducted. Train_test_split divides the dataset into two parts:

1. **Train dataset:** The machine learning algorithm will use this dataset to train the model.

2. **Test dataset:** After the model has been fitted to the Training dataset, it tests the result on the Testing dataset. The model was not trained on this dataset, so the data is ‘unseen’.

Suppose a model produces an accurate classification for the Training dataset but not for the Testing dataset. In this situation, called overfitting, the algorithm learns from the data too well and struggles to generalise to other data (the Testing dataset), creating an ineffective model [123]. The results from the train_test_split technique can be used to determine how well the model generalises by comparing the results from fitting the Training and Test datasets to a model. This technique works well for supervised machine learning regression and classification problems, where the dataset is balanced, and enough data points are available [129].

2.3.5 Hyperparameter optimisation and Cross-Validation

Cross-Validation (CV) is the process of partitioning the data into groups of equal size called folds. The number of folds is commonly expressed as the value k, and thus the whole Cross-Validation process can be referred to as K-fold Cross-Validation [116].

If a dataset contains 100 data points and k is set to 10-fold, the data set will be split into ten groups of 10. Nine folds are then used to train the model, while the remaining fold is used to test the data. This process will then be repeated k-1 times with all the other combinations. The average testing result of all the folds is then used to estimate the performance.

The Cross-Validation process can also be used to select the best hyperparameters that are the best at generalising across the sample data for a specific model. Different combinations of hyperparameters can be used as the input for the algorithm during the CV process. The combination of hyperparameters that produces the best overall CV results will be the ones that best generalise across the data and prevent the model from overfitting to the training data.

There are multiple methods of hyperparameter optimisation in the Sckit-learn library, including GridSearchCV and RandomizedSearchCV. The GridSearchCV method will test every possible combination of hyperparameters, while RandomizedSearchCV requires a user to select how many iterations of random combinations will be used [91]. For both methods, the user will determine the possible range of hyperparameters to be evaluated. The hyperparameters are placed on a grid referred to as the parameter grid.

GridSearchCV is computationally very expensive due to the large number of combinations required to iterate through. For instance, if there are three hyperparameters and 10-fold cross-validation is used, the model will be constructed 241 times.

On the other hand, RandomizedSearchCV significantly reduces the modelling time and computational expense because the user can stipulate how many parameter combinations will be evaluated.

2.3.6 Classification evaluation metrics

After it has been constructed, the effectiveness of a model to make classifications and generalise to other datasets needs to be evaluated. This section will briefly describe the evaluation metrics used in this study.

1. Cohen's Kappa metric

Cohen's kappa is a metric for measuring the effectiveness of a classification model for both multi-class and imbalanced class problems [91]. The formula is:

$$\kappa = \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e}$$

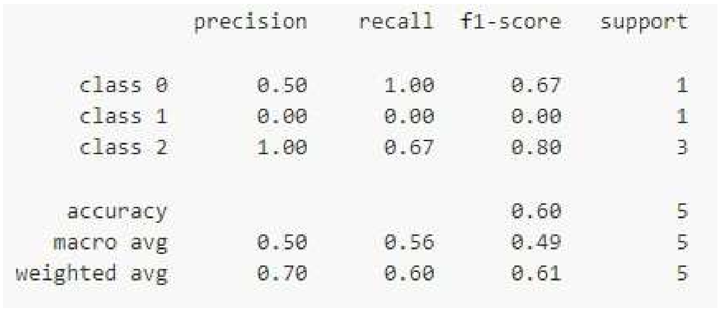
p_o is the observed agreement, and p_e is the expected agreement. This method will evaluate how much better a model is performing than a classifier that is simply guessing at random according to the frequency of each class. Some controversy exists surrounding this measure since there is no standard way to interpret the final result [130]. Landis and Koch [131] provided categories to evaluate the measure in Table 1.

	k
None	< 0
Slight	0 - 0,2
Fair	0,21 -0,4
Moderate	0,41 - 0,60
Substantial	0,61 - 0,8
Almost perfect	0,81 - 1

Table 1. Categories proposed by Landis and Koch to evaluate the Cohen's Kappa metric

2. Classification report and Confusion matrix

When evaluating a classification model, it is essential to consider multiple metrics. An effective Classification report can be displayed by importing the classifcation_report module from the sklearn.metrics library. Figure 17 shows an example of such a report.


```
              precision    recall  f1-score   support

     class 0       0.50      1.00      0.67         1
     class 1       0.00      0.00      0.00         1
     class 2       1.00      0.67      0.80         3

    accuracy                           0.60         5
   macro avg       0.50      0.56      0.49         5
weighted avg       0.70      0.60      0.61         5
```

Figure 17. Example of a Classification report [91]

Precision: This metric measures how many samples predicted to be in a class actually belong to that class. It does not consider any samples in that class that were predicted as being in a different class [91].

Recall: This metric gives the proportion of samples in a particular class that were classified as belonging to that class by the model. It does not take into account any values that were of another class but incorrectly classified as belonging in that class [91].

When it is essential to correctly identify as many members of a specific class as possible, and it does not matter that some samples from other classes would be incorrectly classified to be in that class, one will pay more attention to recall. In contrast, if it is more important that only the correct samples are included in a predicted class, one would pay more attention to the precision [88].

f1 score: Due to the trade-off between Precision and Recall, a useful metric that combines both is the f1 score or F-beta score. This score is the weighted harmonic mean of Precision and Recall [91]. An f1 score of 1 for a class means that all samples were classified correctly and no samples from other classes were incorrectly determined to be in that class. An f1 of 0 means that no samples were classified correctly, and all samples predicted to be in that class were incorrect [91].

Support: The number of samples in each class according to the ground truth or actual values [91].

Accuracy: The number of data points the model correctly classified for all the classes [91].

Macro average: The precision, recall, and f1-score metrics averages across all classes [91]..

Weighted average: The precision, recall, and f1-score metrics averages across all classes, weighted by the number of samples in that class [91]..

It is also helpful to display a confusion matrix to illustrate how effective the model was at classifying the classes. The confusion matrix can be generated through the plot_confusion_matrix function in the sklearn.metric library, as seen in Figure 18.

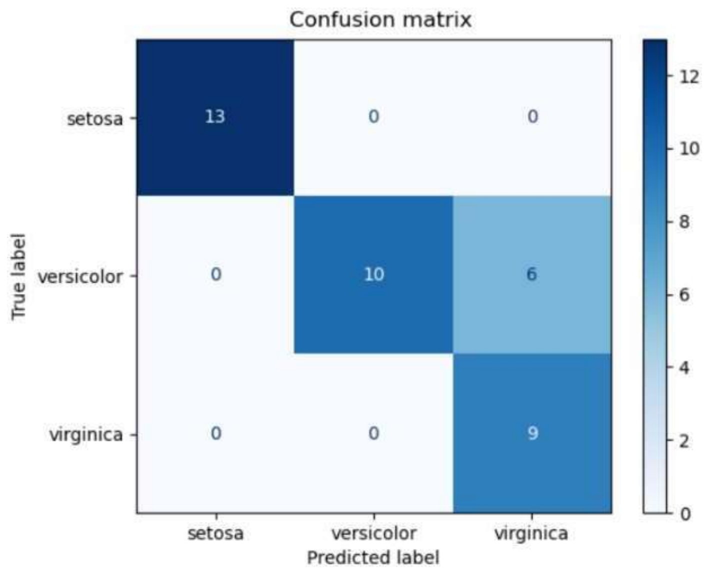


Figure 18. An example of a Confusion matrix [91]

The columns in the matrix correspond to what the algorithm predicted. The rows correspond to what the actual data contained, i.e., the known truth. For the virginica class in the example, the model placed 15 data points into that class, of which nine were correctly classified, but six were actually in the versicolor class. In this case, the precision would be 9/(0+6+9)* 100% = 60 %. For the versicolor class, there are 16 data points, of which the algorithm correctly classified 10. This corresponds to the recall metric, with a recall of 10/(0+16+0) *100% = 62.5%. The overall accuracy for the matrix is the sum of all the blocks where the classes intersect, divided by the total data points (13+10+9)/(13+0+0+0+10+6+0+0+9) *100 = 84,2%.

This visualisation makes it simpler to inspect the classes to make improvements. It is also helpful to display the confusion matrix as percentages instead of the actual number of values.

3. Learning curves

Learning curves are typically plotted with learning on the y-axis and experience on the x-axis. Learning curves are used in machine learning to demonstrate how the algorithm improves predictions over time [132]. Learning curves can also assist by demonstrating possible overfitting or underfitting.

Logistic error log loss (or Log Loss) is considered one of the most robust model evaluation metrics [133]. Plotting logistic error log loss on a curve can provide useful information, including if the model over- or under-fits. The lower the logistic error loss, the better. Figure 19 represents a Log Loss where the model did not over- or under-fit the data.

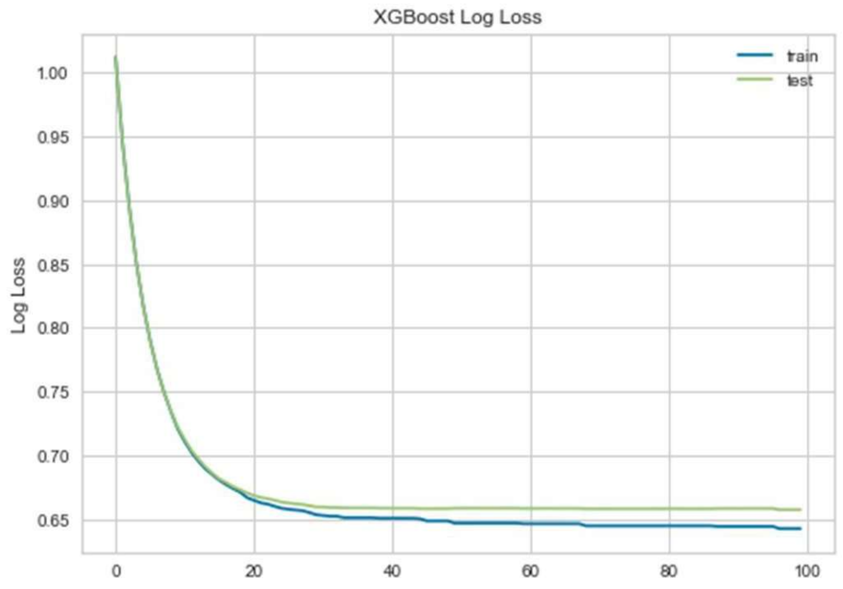


Figure 19. A Log Loss curve representing a model with good fit

4. ROC/AUC curves

The recall metric that can be computed from the confusion matrix is also commonly referred to as sensitivity, especially for binary classification, and expressed as a percentage, i.e. the percentage of samples in a class that the model correctly classified as members of that class. The specificity of a class is the percentage of samples not in a specific class that the model incorrectly classified as belonging to that class.

A classifier will decide if a sample belongs to a specific class based on a probability threshold. For instance, if the threshold is set to 0.5 and the classifier for a binary classification model determines that the probability for a sample to be in class 1 is 0.6, then it will classify that sample as class 1. However, if the threshold is changed to 0.7, it will classify the same sample as being in class 0. Adjusting the threshold is helpful in cases where it is more important to ensure a particular class is classified correctly (i.e. high recall or sensitivity), and specificity is of lesser importance (such as cancer or fraud identification). The Receiver Operator Characteristic graph, commonly known as the ROC curve, is a plot comparing the sensitivity and (1 - specificity) at each possible threshold point. The resulting graph can be illustrated as shown in Figure 20.

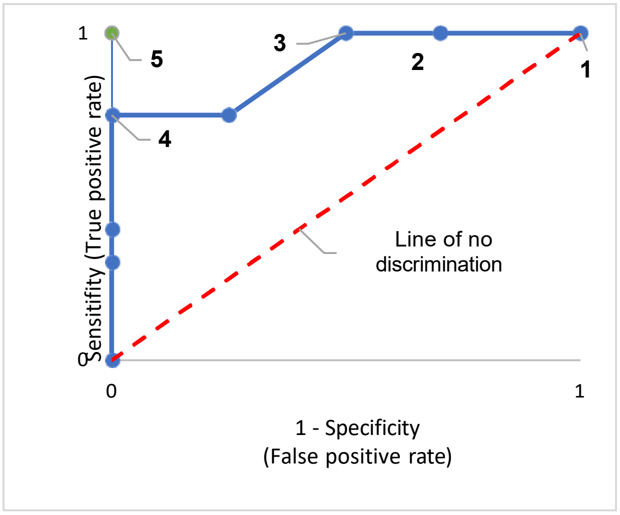


Figure 20. A ROC graph

In Figure 20, the red line is referred to as the line of no discrimination and represents all the points where the true positive rate will equal the false positive rate. In other words, it is where the proportion of correctly classified samples is the same as the proportion of incorrectly classified samples, and the model cannot distinguish between the positive and negative classes. A point on this line is similar to pure chance. Point number 1 is an example of such a scenario.

Point number 2 in Figure 20 is to the left of the line of no discrimination, and the proportion of correctly classified samples is greater than the proportion of incorrectly classified samples. At point number 4, there are no false positives. Depending on the number of false positives acceptable for the specific use case, the optimal threshold would be at either 3 or 4. If point number 5 were on the graph, this would be the threshold where all samples are correctly classified and no samples are incorrectly classified; in order words, a perfect model.

The ROC curve can also compare different models through the Area Under the Curve (AUC) method. The AUC measures the entire two-dimensional area under the ROC curve [134]. Figure 21 illustrates two models where the area underneath Model 1 is greater than the area underneath Model 2. According to their AUC scores, Model 1 is a better classifier than Model 2.

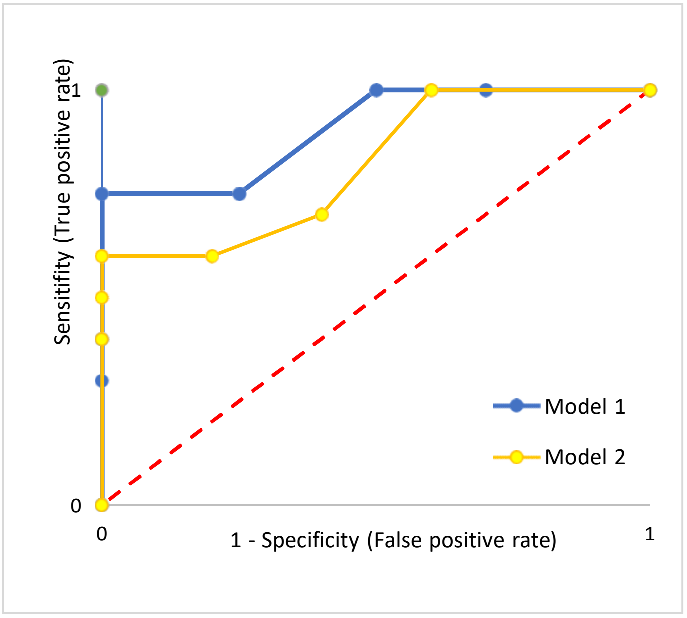


Figure 21. Two ROC graphs depicting the AUC for two models

For multiclass classification, a ROC curve with an AUC value would be calculated for each class in the model. The curves and values can also be used to form overall metrics called micro and macro ROC graphs and AUC scores [91]. The micro metric follows a one-vs-rest strategy where the ROC curve is computed from the sum of all true positives and false positives across all classes. In multiclass classification, the micro metric is preferable if there is a possible class imbalance. The macro metrics follow a one-vs-all strategy that simply takes the metrics' average for each class [91].

2.4 Conclusion

This chapter highlighted the importance of spending funds on the correct customer segments as an effective CRM strategy. As such, it is essential that research into improvements to CRM strategies, such as predicting future CLV segments with the RFM approach, need to be conducted. With more accurate predictions of CLV segments, firms can increase their revenue [19] and improve their tools and strategies that use this versatile metric.

Also highlighted was the increasing importance of machine learning techniques and the diverse use of XGBoost in the field of CRM. However, the lack of formal research using XGBoost to predict future CLV, particularly CLV drop, is notable and central to this study.

A brief review of the machine learning tactics and tools used in this study was also provided. The following chapters will elaborate on how these tactics and tools were implemented to construct and evaluate the models.

3. Methodology and Implementation

This study aims to determine how effective XGBoost is at predicting Customer Lifetime Value (CLV) using the Recency, Frequency and Monetary value (RFM) approach. This chapter will provide a broad overview of how this was conducted.

3.1 CRISP-DM

The CRISP-DM process [135] was followed to construct the final models. The final Deployment phase was excluded because it is out of the scope of this study. The phases of the process and how they are related to each other are illustrated in Figure 22.

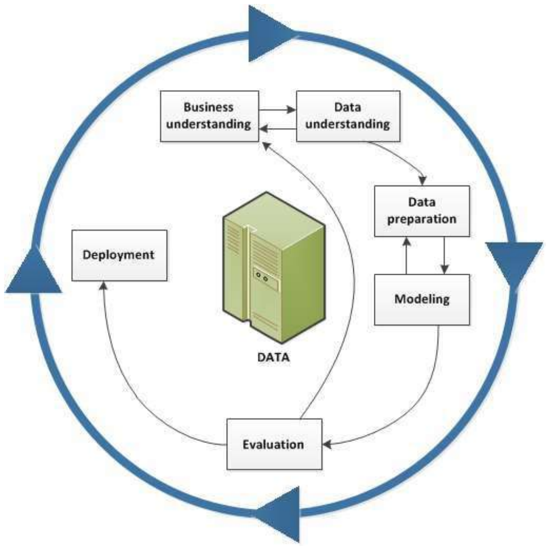


Figure 22. The CRISP-DM data mining life cycle [135]

The topic of this study started with exploratory research into the CRM and marketing use cases where machine learning could make a beneficial contribution. After determining that the prediction of CLV is a worthwhile avenue to pursue, a deeper understanding of the business problem was gained. The data understanding phase started with the sourcing and exploration of datasets that could provide the required RFM values to determine CLV.

After a suitable dataset which contained the correct type of stable customers was found, the data was prepared for modelling through the isolation of RFM and additional customer transaction information. The first iterations of modelling involved only the CLV segmentation created with clustering. Subsequent iterations of the CRISP-DM cycle added a more thorough understanding of how the RFM approach was used in practice and how to conduct effective modelling to predict CLV. As a consequence of the knowledge gained, additional customer segmentation methods were investigated. Further iterations aimed at improving the results of the models involved supplementing the minority classes with SMOTE, reducing the dimensionality of some models with PCA, and hyperparameter optimisation and cross-validation with RandomizedSeachCV.

3.2 Dataset selection

The first step is to extract the RFM features. Only basic information such as which customer made the purchase, when they purchased it, and the purchase amount, is required to extract the features. There are multiple public datasets available which contain this information. However, most only provide minimal additional data about purchases. Recall that a secondary goal of this study is to establish if adding additional purchase information to the RFM values could assist the classifier in achieving improved results.

The chosen dataset, which included the required additional information, was sourced from Dunnhumby, a global company specialising in consumer analytics and data science. The dataset is called "The Complete Journey" and can be freely downloaded on their website at: https://www.dunnhumby.com/source-files/ [136]. The dataset contains all the transactions made by a sample of 2500 customers who were regular shoppers at an unnamed Fast Moving Consumer Goods retail chain. Dunnhumby stated in the dataset's information that it is ideal for academic research [137]. Tables 2 to 4 describe the data contained in the three tables used for this study's purpose.

1. The *transaction_data* table contains 2 595 732 rows and 12 columns. Every line can be compared to a line found on a store receipt. It can be seen from this table that a unique customer was a household, which can contain more than one person, but will be viewed as a single customer hereafter. Table 2 is a description of the data contained in that table.

Variable	Description	Range or Count
HOUSEHOLD_KEY	Uniquely identifies each household (customer).	1 - 2500
BASKET_ID	Uniquely identifies purchase occasion.	276 484
DAY	Day when the transaction occurred.	1-711
PRODUCT_ID	Uniquely identifies each product.	92 339
QUANTITY	The number of each type of product purchased during the visit.	
SALES_VALUE	Amount of dollars retailer receives from the sale.	
STORE_ID	Identifies unique stores.	582
COUPON_MATCH_DISC	Discount applied due to retailers match of manufacturer coupon.	80
COUPON_DISC	Discount applied due to manufacturers coupon.	
RETAIL_DISC	Discount applied due to retrials loyalty card program.	
TRANS_TIME	Time of day that the transaction occurred.	
WEEK_NO	Week of transaction.	1 - 102

Table 2. Description of data contained in transaction_data table

2. The *product* table contains 92 353 rows and seven columns. It has information on each product sold at the retail chain. Information about this data can be found in Table 3.

Variable	Description	Unique values
PRODUCT_ID	Number that uniquely identifies each product	92339
DEPARTMENT	Groups similar products together	44
COMMODITY_DESC	Groups similar products together at a lower level	308
SUB_COMMODITY_DESC	Groups similar products together at the lowest level	2383
MANUFACTURER	Code that links products with their manufacturer	6476
BRAND	Indicates Private or National label brand	2
CURR_SIZE_OF_PRODUCT	Indicates package size	

Table 3. Description of data contained in the product table

3. The *coupon_redempt* table contains 2318 rows and four columns. It carries the information about which households redeemed coupons during the period of supplied data. A description of its information can be seen in Table 4.

Variable	Description	Range or Unique values
HOUSEHOLD_KEY	Uniquely identifies each household.	1 - 2500
DAY	The day when the transaction occurred	
COUPON_UCP	Uniquely identifies each coupon	566
CAMPAIGN	Uniquely identifies each campaign	30

Table 4. Description of data contained in coupon_redempt table

The *transaction_data* table was used to isolate and extract the RFM values for each customer and formed the first set of features, henceforth called the R (for **R**FM) subset. The *product* and *coupon_redempt* tables were used to add additional information to this dataset, and the resulting information formed the second feature set, henceforth called A (for **A**dditional Features).

3.3 Timeframes and target variables

Another goal of this work was to determine the effectiveness of the models in predicting CLV for a period that does not follow directly after the one used as input features (i.e. current knowledge). The R and A datasets were divided into 6-month periods that was used to create the Groups that would contain the feature set and target variables. Three groups were used in the experiments:

- Group 2 contained the input feature set and covered the second six month period,
- Group 3 (the third six-month period) followed directly after Group 2, (the third six month period) and contained the target variables to be predicted.
- Group 4 (the third six-month period) followed directly after Group 3 and contained another

distinct group of target variable values to predict.

Group 1 was not used, as will be explained in Section 3.7.

The following variations of CLV for Groups 3 and 4 were chosen as the target customer segments that the models would predict:

1. Monetary clusters (C): K-means clustering was used to cluster the isolated Monetary values.
2. Terciles (T): Customers were segmented by dividing their revenue value into terciles.
3. RFM Segment (S): Customers were segmented using a weighted-RFM method.
4. Profitability clusters (P): K-means clustering was used to create clusters from the isolated Monetary and Frequency values.
5. Loyalty clusters (L): K-means clustering was used to create clusters from the isolated Recency and Frequency values.
6. Burgeoning clusters (B): K-means clustering was used to create clusters from the isolated Monetary and Recency values.

An example of using monetary value clustering to create CLV segments, as in 1 above, can be seen in the example presented by Karaman [118]. Miglautsch [47] placed the RFM into quantiles to determine CLV and this served as an example for the tercile variant number 2. The third variant is an adaptation of the most common method of determining CLV with weighted-RFM values described in section 2.1.2. Examples of using two RFM variables as CLV variants, as in 4 to 6 above, can also be found [62, 138].

Boolean CLV Drop target variables were also created by determining whether or not customers moved to a lower segment between groups (i.e., Group 2 to Group 3 and Group 2 to Group 4).

Predicting which customers would likely move to a lower CLV can be a direct input for a targeted CRM marketing campaign to reduce such 'drops'. To predict which customer would drop to a lower CLV, those in the High and Medium categories in group 2 were isolated. Next, it was determined which of these dropped in Groups 3 and 4, i.e. which customers moved from High to Medium/Low or from Medium to Low. The M and P Drop datasets were too small for the classification experiments and were not used as CLV drop variants. As a result, the following CLV Drop features were created for both Group 3 and 4:

7. Loyalty Drop (LD): Customers that moved from a higher to lower cluster for Loyalty as determined in number 5 above.
8. Burgeoning Drop (BD): Customers that moved from a higher to lower cluster for Burgeoning as determined in number 6 above.
9. Tercile Drop (TD): Customers that moved from a higher to lower tercile as per 2 above.
10. Segment Drop (SD): Customers that moved from a higher to lower RFM segment created in number 3 above.

3.4 Data preparation

The first step in preparing the data for modelling was to isolate the total revenue and sales value per customer for the entire timeframe of the data. From this analysis, the Groups representing the periods were determined, and the RFM for each period was extracted. The result was the creation of the R dataset. Firstly, the RFM outliers for each group were removed from the R dataset and then the values were standardised so that higher R, F and M values were closer to one. Next, CLV variants were determined through K-means clustering, tercile isolation and the weighted-RFM method. The Boolean CLV drop values were calculated and added to the dataset by isolating those CLV values that dropped to a lower CLV from Group 2 to Group 3 and Group 2 to Group 4.

To create the A dataset, the additional customer transactional information was extracted and added to the R dataset. PCA was performed to reduce the dimensionality for the future CLV prediction. Each CLV drop variant had a slightly different dataset because only the customers who could experience a drop in Group 2 were included during the modelling. PCA was performed on each dataset, yielding slightly different final datasets for each CLV drop variant.

The created features exhibited moderate class imbalance, and as such, the SMOTE method was used to correct this imbalance during the final modelling. To be consistent, this was done for all the experiments, irrespective of the degree of imbalance. If there were no imbalance, SMOTE would have a negligible impact. The class imbalance was corrected for the CLV Drop models through the XGBoost hyperparameter scale_pos_weight, which was preferable due to the boolean nature of the classification.

3.5 Learning and Evaluation

Having created all the input features describing current knowledge and a variety of target variables representing customer segments and customers who dropped down, the effectiveness of XGBoost prediction was explored. The datasets were divided into training and test sets. Then 10-fold cross-validation with hyperparameter optimisation was done to produce the final models. The results of the experiments were then assessed through various evaluation metrics.

The remaining sections of this chapter will provide additional detail on the steps taken during the implementation process.

3.6 Coding language and development environment

At the end of 2018, the Python programming language was ranked as the most popular programming language for Machine Learning on Github, the world's largest code hosting platform [139]. By the end of 2020, it also ranked as the second most used language overall on Github [140]. Python is an open-source programming language with minimal restrictions on how it can be used [141]. In addition to its ease of use and low learning curve [142], the language provides access to many beneficial libraries for data manipulation and machine learning.

Considering the statements above, Python was the language of choice to conduct the experiments in this study. Described in the summary Tables 5 and 6 are the packages and their components used to manipulate the data for modelling. Tables 7,8 and 9 list the packages and functions used to create the final models and make the predictions.

Package	Module or function	Brief description	Webpage
Numpy [143]	absolute	Determine the absolute value.	https://numpy.org/doc/stable/reference/generated/numpy.absolute.html
	cumsum	The cumulative sumof a number of elements.	https://numpy.org/doc/stable/reference/generated/numpy.cumsum.html
	floor	Returns the floor value of a given decimal value.	https://numpy.org/doc/stable/reference/generated/numpy.floor.html
	tile	Construction of an array.	https://numpy.org/doc/stable/reference/generated/numpy.tile.html
Matplotlib [144]	pyplot	Plotting of data.	https://matplotlib.org/
Plotly [145]	graph_objs, iplot, plot	Plotting of data.	https://plotly.com/
Scipy [146]	z-score	To determine the Z score.	https://docs.scipy.org/doc/scipy/reference/stats.html

Table 5. Numpy, Matplotlib, Plotly and Scipy modules and functions used for data manipulation

Package	Module or function	Brief description	Webpage
Pandas [147]	agg	Aggregate over DataFrame using operations.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.agg.html
	count	Count cells for each column or row that contains a value.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.count.html
	concat	Concatenates pandas objects.	https://pandas.pydata.org/docs/reference/api/pandas.concat.html
	dropna	Removes missing values from a DataFrame.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html
	get_dummies	Converts a categorical variable into numerical one.	https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html
	groupby	Groups the variables in a DataFrame according to specified function.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html
	max	Returns the maximin value in a given DataFrame axis.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.max.html
	min	Returns the minimum value in a given DataFrame axis.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.min.html
	nunique	Counts the number of distinct elements in a DataFrame	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.nunique.html
	read_csv	Read a .csv file into a Pandas DataFrame."	https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html
	sum	Returns the sum of the values in a DataFrame.	https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.sum.html

Table 6. Pandas functions used for data manipulation

Package	Module or function	Brief description	Webpage
Imbalanced-learn [148]	make_pipeline	Constructs a Pipeline so that given estimators can be implemented in the correct order.	https://imbalanced-learn.org/stable/references/generated/imblearn.pipeline.make_pipeline.html
	SMOTE	Implementation of Synthetic Minority Over-sampling Technique	https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html
XGBoost [13]	plot_importance	Plot importance of features based on trees fitted.	https://xgboost.readthedocs.io/en/stable/python/python_api.html
	XGBClassifier	Implementation of XGBoost classification though API.	https://xgboost.readthedocs.io/en/stable/python/python_api.html

Table 7. Imbalanced-learn and XGBoost modules and functions used to construct models

Package	Module or function	Brief description	Webpage
Scikit-learn [91]	accuracy_score	Generates the accuracy classification score.	https://scikitlearn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html
	classification_report	Constructs a text based classification report.	https://scikit- learn.org/stable/modules/generated/sklearn.metrics.classification_report.html
	cohen_kappa_score	Generates the Cohen's kappa score	https://scikit- learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score.html
	KFold	Provides train/test indices to split data in train/test.	http://scikit- learn.org/stable/modules/generated/sklearn.model_selection.KFol d.html
	KMeans	Performs K-Means clustering.	https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
	MinMaxScaler	Scaling of features to a given range.	https://scikit- learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
	plot_confusion_matr ix	Plots the Confusion Matrix.	https://scikit- learn.org/stable/modules/generated/sklearn.metrics.plot_confusion_matrix.html
	silhouette_score	Calculates the mean Silhouette Coefficient.	https://scikit- learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html
	StratifiedKFold	Stratified K-Folds cross-validator.	https://scikit- learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html
	train_test_split	Splits a DataFrame into random train and test subsets.	https://scikit- learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Table 8. Scikit-learn modules and functions used to construct models

Yellowbrick [115]	KElbowVisualizer	Implements the elbow method and produces a visualisation of the result.	https://www.scikit-yb.org/en/latest/api/cluster/elbo w.html
	ROCAUC	Generates a ROC/AUC plot.	https://www.scikit-yb.org/en/latest/api/classifier/rocauc.html

Table 9. Yellowbrick modules and functions used to construct models

The Jupyter Notebook interactive development environment (IDE) was chosen as the platform for conducting the experiments. This platform is commonly used in data science and is referenced in many machine learning research articles [149]. This web-based IDE has a modular design that makes it ideal for data science and machine learning experiments [150]. The required Python packages were easily accessed after installing them through the commonly used *pip* tool [151].

3.7 Determining the timeframe for the groups

Figure 23 shows that the total sales value increased for the first 16 weeks and then stabilised. Figure 24 depicts the number of baskets, in other words, the number of visits to all retail outlets during the entire period. As expected, the trend is roughly the same as in Figure 23, with the number of visits stabilising after around week 20.

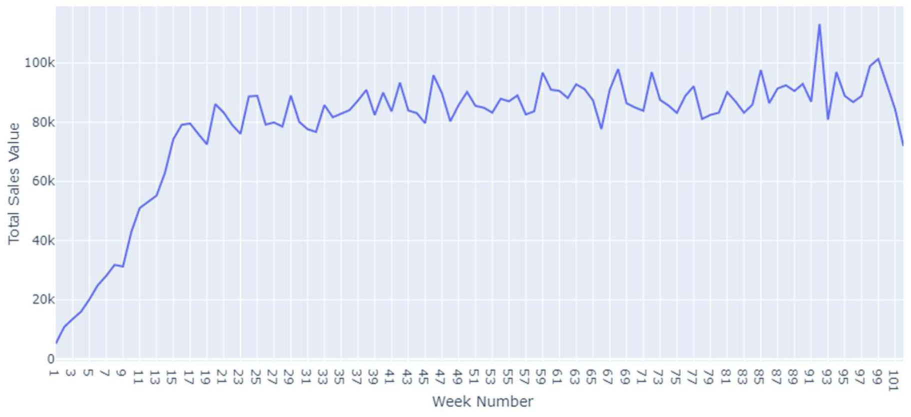


Figure 23. Total sales value per week

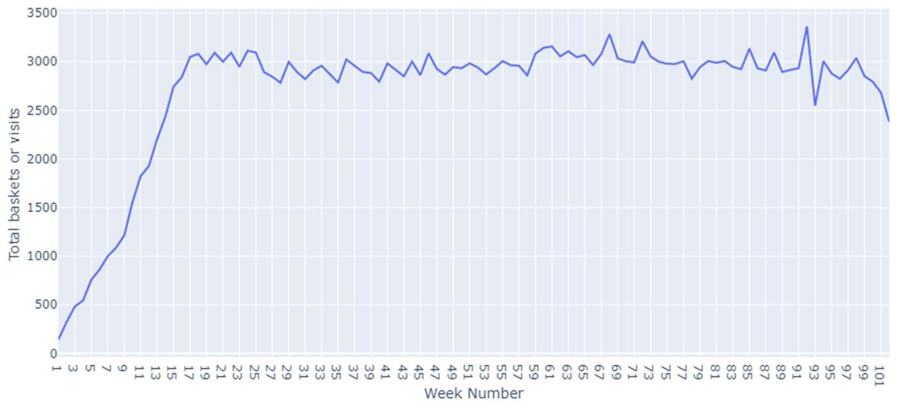


Figure 24. Total baskets per week

Figure 25 depicts the total sales per household over the entire timespan of the dataset with the range of Sales Value from 0 to 38319 and a mean value of 3223. The 25th percentile is at 970.74, the 50th percentile at 2157.75 and the 75th percentile at 4413.32. It can be seen from Figure 25 and the spread of the data that the Total Sales Value (for the entire period) is not uniformly distributed among the households.

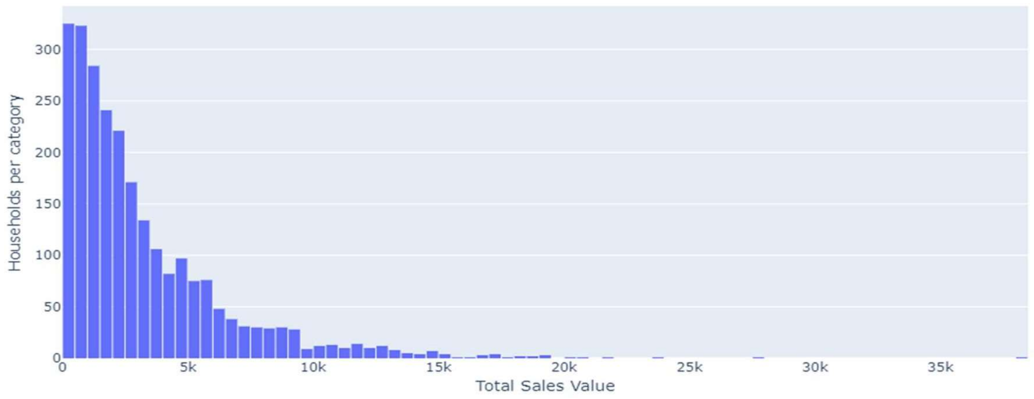


Figure 25. Total Sales Value per household for the entire timeframe

As discussed, periods need to be isolated to serve as CLV analysis periods. Periods of six months were chosen for each group because the dataset covers roughly two years and can thus be divided into four parts of equal length, consisting of 24 weeks. The 6 week period left at the end of the entire timeframe of the data after the groups were created was not included in the experimentation.

As the total sales value and baskets per week only stabilised to a consistent pattern after 20 weeks, the first part of the dataset (Group 1) was not used. The remaining groups were divided to constitute 24 full weeks each.

Table 10 describes the four groups, henceforth referred to as Groups 1 to 4:

Group	Used in experiments	Day	Week	Day of the Week	Weeks	Days
1	Not used	1 ↕ 173	0 ↕ 24	3 ↕ 7	24	173
2	Training the model	174 ↕ 348	25 ↕ 48	1 ↕ 7	24	175
3	To predict	349 ↕ 523	49 ↕ 72	1 ↕ 7	24	175
4	To predict	524 ↕ 698	73 ↕ 96	1 ↕ 7	24	175

Table 10. Division of Groups

3.8 Isolation of R F M variables

The data were grouped by the "household_key" variable through the Pandas groupby function and then aggregated to extract the required R, F and M values.

The M feature was extracted using the *sum* function on the SALES_VALUE variable. This variable was used in its original format because this is the actual amount of money that the retailer received from the purchase [137].

F was determined through the *nunique* function, which counted the number of unique baskets by BASKET_ID. It is assumed that every basket is a unique visit to the retailer. For R, the *max* function was used on the DAY variable to extract the last day a purchase was made.

3.9 Outlier detection and removal

The Pandas box plot method was applied as a first attempt to remove outliers. However, the standard box plot pruned the features too aggressively when using the standard calculation method for the whisker limits. For instance, for group 2, 9% of the M values, 9.7% of F variables and 14,8% of R variables were removed. To form a data set that contained the pruned values for all three variables, the upper limit of 14,8% from the R variable would need to be used. As such, following this approach would eliminate too many samples from the dataset.

In practice, the higher value variables might be part of the segment that is very valuable to the organisation. Therefore, removing too many variables from the top end of the dataset will eliminate valuable data that would create more representative segments. Because of the overly aggressive pruning of the box plot method, the Z-score method was used.

The Z-score was calculated by importing the SciPy statistical functions and using the *.zscore* function from the stats library. The Z-score for each variable was calculated and removed if above a pre-determined upper limit.

With the upper limit set to 3, 13,4% of the samples were removed. As with the boxplot method, this was deemed to be too aggressive. If the upper limit was changed to 4, only 7.1% of the samples were removed. Using an upper limit of z=4 to remove outliers is acceptable in certain situations [85]. Applying this method is preferred over the box plot method because it is simple and can be consistently applied to all the variables in one step.

For the R variable, the values were subtracted from the maximum value (DAY) for the group so that more recent values would be closer to 0. This subtraction was done to standardise the outlier removal process for the three RFM values. Next, the outliers were removed as described previously and then converted back so that the more recent values were closer to 175 (total number of days per group). This step was required to make the higher values for R (as for F and M) more beneficial in terms of customer characteristics.

The number of rows (households) for the resulting dataset changed from 2282 to 2119. Table 11 is a comparison of the variables before and after pruning. It shows that extreme outliers were removed, and resulting standard deviations were notably lower for the pruned dataset.

	Maximum before pruning	Maximum after pruning	Standard deviation before pruning	Standard deviation after pruning
Group 2 Monetary Value	8285	4768	988	842
Group 3 Monetary Value	12089	4938	1018	854
Group 4 Monetary Value	13352	4943	1031	878
Group 2 Frequency	413	163	36	26
Group 3 Frequency	463	177	37	26
Group 4 Frequency	340	161	33	26
Group 2 Recency	173	116	25	20
Group 3 Recency	174	113	25	19
Group 4 Recency	168	117	26	20

Table 11. Comparison of variables before and after pruning (rounded to the nearest integer)

3.10 Creating CLV values

As described in section 3.3, various CLV segmentations were created. These include those created from clusters, terciles and weighted-RFM Segments. This section describes how the different segments were created.

3.10.1 Clusters

Unsupervised clustering using the k-means algorithm was used to segment customers according to their respective CLV.

Before clustering, the non-categorical values were normalised to improve the clustering algorithm's efficiency [152]. The Scikit-learn *MinMaxScaler* function was employed for normalisation throughout this study.

The following cluster sets were formed:

Monetary (M): The Monetary value was used as the input for one-dimensional clustering. It is assumed that revenue generated from a customer is the most crucial aspect of that customer and can be used as a definition of CLV [118].

Loyalty (L): Two-dimensional clustering using F and R as dimensions. It is assumed that customers who have visited the retailer more recently and frequently during the measurement period will be more loyal than other segments.

Profitability (P): F and M variables form the two dimensions for clustering this CLV variation. It is assumed that customers that visit the retailer more often while spending more money than other customers will be a more profitable segment of the customer base.

Burgeoning (B): Two-dimensional clustering using R and M as dimensions. It is assumed that customers who spent a higher amount more recently will provide more value for the company in the future.

The RFM variables were also clustered on all three dimensions, but only for Group 2; the resulting cluster values were among the additional features in the A classification experiments. All CLV clusters created for Group 2 formed part of the features used to predict the target variables in Groups 3 and 4 in the A (Additional dataset) experiments. The R (RFM dataset) experiments had only the scaled R, F and M values as input features.

The k-means clustering algorithm described in section 2.3.2 was used to create clusters. The Elbow method was utilised to determine how many clusters (k) to use. Firstly, a loop was used to iterate through the number of clusters, and the resulting inertia for each loop was graphically illustrated. Figure 26 is a representation of the loop for the Group 2 Monetary value.

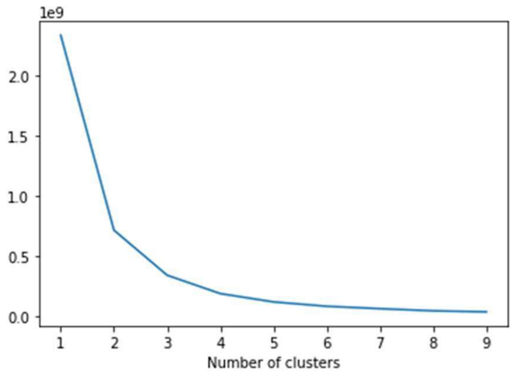


Figure 26. The Elbow method through the use of a regular loop and plotly

As can be seen from Figure 26, the actual ‘elbow point’ is not explicitly indicated through this simple method. This leaves it up to the user for interpretation. In certain instances, the elbow point is clear, but there might be inconsistent interpretations between observers in others. For this reason, the KElbowVisualizer tool from the Yellowbrick library was used because it explicitly indicated the elbow point, as can be seen in Figure 27:

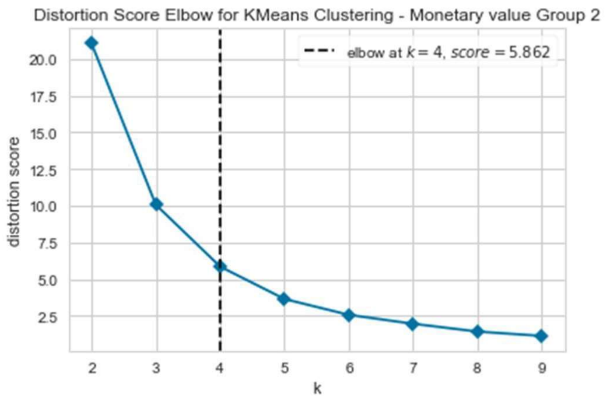


Figure 27. Representation of Elbow method with the Yellowbrick library

K is determined by the knee point detection algorithm, which determines the point of maximum curvature. The knee point also represents the pivot of the elbow in a "well-behaved clustering problem". K is illustrated by a dotted line and is only present if the method works [115]. In all cases, the dotted line was visible. In all but one outcome, the optimal number of clusters was 4. The only exception was Group 2 P, which had a k=5.

The silhouette score was used to validate the choice of four clusters. In contradiction with the Yellow Brick elbow number, this alternative method indicated that the 2 or 3 clusters would have the highest average silhouette score. Since 3 clusters would be more practical from a cluster size and marketing point of view than 2, K was set to 3 for all experiments. Table 12 shows a few of the average silhouette scores for Group 3:

	Average silhouette score				
Number of clusters	2	3	4	5	6
Monetary clusters	0,67	0,62	0,59	0,58	0,58
Profitability clusters	0,59	0,51	0,45	0,46	0,45
Loyalty clusters	0,58	0,52	0,46	0,47	0,45
Burgeoning clusters	0,45	0,54	0,44	0,47	0,44

Table 12. Average silhouette score for Group 3 clustering

After considering the silhouette scores, it was decided that k would equal three, resulting in three clustered groups and thus three classes that would be predicted during classification.

After the clustering was concluded, it yielded the following CLV variants (henceforth abbreviated as shown in brackets below):

- Monetary Cluster for Groups 3 and 4 (M3 and M4)
- Profitability clusters for Groups 3 and 4 (P3 and P4)
- Loyalty clusters for Groups 3 and 4 (L3 and L4)
- Burgeoning clusters for Groups 3 and 4 (B3 and B4)

3.10.2 Tercile groups

In order to be consistent with the final number of clusters chosen in section 3.10.1, the monetary value was divided into terciles which will form three segmented groups.

The *quantile and iloc* functions were used to extract the terciles. Customers were segmented into groups that were partitioned by the tercile values. These terciles were ranked from 0 for the lowest tercile to 2 for the highest tercile. Table 13. represents the maximum monetary value and standard deviation for each tercile by Group.

	Terciles					
	0		1		2	
	Households	Maximum M (std dev)	Households	Maximum M (std dev)	Households	Maximum M (std dev)
Group 2	700	352.57 (96.86)	698	915.29 (156.09)	721	4767.55 (797.45)
Group 3	699	362.55 (100.80)	699	997.24 (176.80)	721	4937.79 (781.51)
Group 4	699	377.31 (106.31)	699	1031.25 (188.23)	721	4943.46 (799.41)

Table 13. Monetary value maximum and (standard deviation) for each tercile by Group

3.10.3 Weighted-RFM Segment

Initially, all R, F and M combinations were iterated through on a scale of 1 to 5 for each value, before running the XGBoost classifier. Even though this improved accuracy, using this approach is not feasible because scaling the RFM features in this manner would make the model not generalisable. It was also found that even slight changes, such as changing the random seed of the model, made the weighting improvements negligible. As such, creating the RFM segment followed the general approach described in section 2.2.1.

To create the weighted-RFM segments, the R, F and M values were divided into quintiles and assigned a value between 1 and 5. The first quintile (< 0.2) was named one, and the last quintile (>= 0.8 to 1) was named five. The *quantile* and *iloc* functions were again used.

In order to substitute the AHP data (domain expert weightings) absent from this dataset, the gain obtained by each feature from the Monetary cluster and Tercile classification experiments was averaged and used to scale the RFM values.

The resulting scaling values chosen were 2 for recency (R), 1 for frequency (F) and 8,5 for Monetary (M). Next, the scaling values were multiplied by the quintile values, and the sum of the results was calculated. These totals had a range of 11,5 to 57,5. These values were then divided into three segments (Low, Medium and High) according to their respective terciles. Where multiple households had the same tercile values at the split, they were all included in the higher tercile.

This process was performed for all three groups resulting in the following breakdown of the count of customers in each segment in Table 14.

	Number of households		
Segment	Group 2	Group 3	Group 4
Low	697	524	677
Medium	799	823	649
High	723	772	793

Table 14. Number of households per RFM segment by group

The RFM segments for Group 3 (S3) and Group 4 (S4) will be predicted, while the Group 2 segments will form part of the model's features.

3.11 Creating the Drop variables

To predict which customer would drop to a lower CLV, those in the High and Medium categories in group 2 were isolated. Next, it was determined which of these dropped in Groups 3 and 4, i.e. which customers moved from High to Medium/Low, or from Medium to Low. If a drop in CLV occurred, the customer was given a value of 1. Households that did not drop received a 0 value. It was found that the resulting M Drop dataset and the P Drop dataset only had 726 and 872 households, respectively. These data sets were too small for the classification experiments because the results would not be useful and were not used. The remaining datasets and CLV variants are described in Table 15.

Type of Drop	Description	Dataset size (Households)	Abbreviated names
Tercile	Tercile 3 and 2 to lower tercile	1419	TD3 and TD4
RFM Segment	RFM Segment High and Medium to lower	1422	SD3 and SD4
Loyalty	High and Medium Loyalty cluster to lower	1869	LD3 and LD4
Burgeoning	High and Medium Burgeoning cluster to lower	1872	BD3 and BD4

Table 15. The number of households in the final CLV Dropper experiments

3.12 Extracting product and temporal purchase data

The following additional features to be used in the classification experiments were created by combining information from the *transaction*, *product* and *coupon_redempt* tables for each household for Group 2:

a) Average value per basket: Total value divided by unique baskets (*Avr_Value_per_basket)*
b) The highest and lowest basket value through the use of the *min* and *max* pandas function (*Max_value_Basket and Min_value_Basket*)
c) Sum of baskets (*BasketsPerMonday to BasketsPerSunday*) and value of products (*ValuePerMonday to ValuePerSunday*) for each day of the week.
d) The total number of baskets and total value for the weekdays and weekends were determined from the values created in c) above (*BasketsWeekdays, ValueWeekdays, BasketsWeekends, ValueWeekends*).
e) Value for each purchase period per hour. The purchase time for each product was reduced to the hour value. Then the total number of baskets purchased during the hour (*Time_1 to Time_24*) and the total sales value (*Value_1 to Value_24*) for that hour were determined.
f) The number of stores visited by each customer was counted through the STORE_ID variable. Households were then placed into five categories depending on how many stores they visited to form the *NumStoresVisted_group* variable.
g) SUB_COMMODITY_DESC and GASOLINE-REG UNLEADED had an abnormally large item count and were removed. Then the maximum number of items purchased per basket (*ItemsMaxPerBasket*), the average number of items per basket (*AverageItemsPerBasket*) and the average cost per item over all purchases (*AvrCostPerItem*) were determined.
h) The coupon_redempt sheet was used to determine how many coupons were used (*TotalCouponsUsed*) and if a coupon was used (*UsedCoupon*) during the period.
i) The total number of gasoline purchases (*NumberBaskets_Car*) and total spend on gasoline (*SalesTotal_Car*) were determined by isolating the GASOLINE-REG UNLEADED subcategory. These two values were then used to determine the average value of the amount of gasoline per visit (*Avr_ValuePerTank*).
j) The percentage of spend on national (*Percentage_National_Brand*) and house (*Percentage_Private_Brand*) brands were determined by totalling the spending.
k) The total percentage spent per DEPARTMENT was determined by dividing the total value of each DEPARTMENT category by the total value. This produced an additional 32 features with the prefix Value_, for instance, Value_PRODUCE.
l) Similar to k) above, the total spend per department was determined but not divided by the Value spent for each customer as a direct indication of the amount spent in each DEPARTMENT category a prefix Spend_ added (for instance, Spend_PRODUCE).

3.13 Classification experiments with XGBoost

The CLV and CLV Drop variations isolated in the previous sections became the dependent variables in the XGBoost classification experiments. Two sets of experiments were conducted. The first one used only the standardised RFM values as the features to create the R datasets.

The other used the standardised RFM values, the added features from section 3.12 and RFM Group 2 residuals from clustering and segment creation to form the A datasets. Many of the additional features in the A dataset had to be standardised or transformed with one hot encoding.

The final sets of experiments and the target variables are summarised in Table 16, which shows the abbreviated names of the 44 experiments.

Target variable	Input features and target timeframe			
	RFM only (R)		Added features (A)	
	Group 3	Group 4	Group 3	Group 4
C - monetary Cluster (Monetary cluster) – multiclass classification	R-3C	R-4C	A-3C	A-4C
T - Tercile (Terciles) – multiclass classification	R-3T	R-4T	A-3T	A-4T
S - Segment (weighted-RFM Segments) – multiclass classification	R-3S	R-4S	A-3S	A-4S
P - Profitability (2D Frequency and Monetary clusters) – multiclass classification	R-3P	R-4P	A-3P	A-4P
L - Loyalty (2D Recency and Frequency clusters) – multiclass classification	R-3L	R-4L	A-3L	A-4L
B - Burgeoning (2D Recency and Monetary clusters) – multiclass classification	R-3B	R-4B	A-3B	A-4B
TD - Tercile Drop (Terciles) – binary classification	R-3TD	R-4TD	A-3TD	A-4TD
SD - Segment Drop (Weighted-RFM Segments) – binary classification	R-3SD	R-4SD	A-3SD	A-4SD
LD - Loyalty Drop (2D Recency and Frequency clusters) – binary classification	R-3LD	R-4LD	A-3LD	A-4LD
BD - Burgeoning Drop (2D Recency and Monetary clusters) – binary classification	R-3BD	R-4BD	A-3BD	A-4BD
SD - Segment Drop (Weighted-RFM Segments) – binary classification	R-3SD	R-4SD	A-3SD	A-4SD

Table 16. Final classification experiments conducted, and their abbreviated names

3.13.1 SMOTE and scale_pos_weight

In initial experimentation, it was determined that the CLV segments formed after clustering were unbalanced. Therefore, SMOTE for the multiclass CLV values, and scale_pos_weight for the Boolean CLV Drop experiments were used to correct this imbalance during the training phase of the classification experiments.

SMOTE for the CLV values

The effectiveness of SMOTE was evaluated for the R and A subsets by comparing a model where it was implemented during hyperparameter optimisation to one where it was not. It was found that SMOTE offered a slight improvement in the micro averaged results (which consider the effect of the imbalance and was used for this study). The improved results might be attributed to the improvement in recall values for the minority class, which SMOTE can facilitate.

SMOTE was implemented through the pipeline function (from the *imblearn* library) with RandomizedSearchCV. The pipeline function ensures that the same synthetic samples do not occur in both the training and testing datasets and that SMOTE and cross-validation are done in the correct order.

Scale_pos_wieght for the CLV Drop models

For the CLV Drop experiments, which only contain two classes, the results from using SMOTE were compared with setting the scale_pos_weight XGBoost parameter to the appropriate level with the formula [153]:

Scale_pos_weight = (sum of negative instances) / (sum of positive instances)

The results showed no difference between the two. Because the *scale_pos_weight* parameter is so simple to implement, it was chosen above SMOTE.

For the CLV Drop models, the values of *scale_pos_weight* were determined through the formula above. However, the classifier did not manage to correctly classify the CLV Drop class if *scale_pos_weight* was set too low. Therefore, the higher of the Group 3 and 4 values were chosen. The parameter was then rounded to the nearest appropriate value. The value was also increased by one for the Loyalty and Burgeoning experiments because the preceding steps did not increase the value enough to produce a favourable result. The final parameter values for *scale_pos_weight* were 3.5 for the TD set, 5 for the SD set, 7 for the LD set, and 7.5 for the BD set. It was also established that the results improved as the value for the parameter was increased above these numbers. However, values higher than those mentioned were not used to ensure effective generalisation.

3.13.2 Hyperparameter optimisation

RandomizedSearchCV was used to conduct the hyperparameter optimisation with Cross-Validation set to 10 folds. As mentioned above, to use RandomizedSearchCV correctly with SMOTE, the *make_pipeline* function from the *imblearn* library was employed. After the optimal hyperparameters were determined, they were used in the final model, which did not use SMOTE in the training dataset.

An additional advantage of using RandomizedSearchCV over GridsearchCV is that a broader range of possible parameters can be evaluated since GridsearchCV would need to use all the possible combinations of parameters, which would take too much time. The most important hyperparameters to be tuned were chosen to be placed on the parameter grid. The final range of hyperparameters, with the step in brackets, was:

gamma: 3 to 24 (1)

eta (*learning_rate*): 0.00 to 1.95 (0.05)

reg_alpha: 0.0 to 1.9 (0.1)

reg_lamba: 0.0 to 2.9 (0.1)

max_depth: 3 to 39 (1)

min_child_weight: 2 to 19 (1)

subsample: 0.5 to 0.9 (0.1)

The evaluation metric in the model was set to mlogloss and scoring to accuracy for the CLV models because they were evaluated according to overall accuracy. The classifier failed to classify the Drop class if the evaluation was set to logloss and accuracy. It is assumed that XGBoost is mainly concerned with overall accuracy, and will favour the non-Drop class because it produces a higher accuracy score. This is similar to what was observed when not adjusting scale_pos_weight to favour the Drop class. The evaluation metric for the CLV Drop experiments was set to auc and scoring to roc_auc because the Drop class was the critical class to evaluate.

The number of RandomizedSearchCV iterations for all hyperparameter tuning sessions was set to 1500 to ensure that good hyperparameters were selected. For consistency, the number of iterations was kept the same for all the tuning sessions.

3.13.3 Dimensionality reduction in Added Features experiments

Running the experiments with the A dataset was problematic because the hyperparameter optimisation took much longer than the R dataset. For example, running the R-3C experiment with 1000 iterations took 1,5 hours, while running the same experiment with the added features (A-3C) took longer than 15 hours. Accordingly, PCA was used to reduce the dimensionality of the A dataset.

To test the effect of PCA on the results, the prementioned experiments (R-3C and A-3C) were compared with and without PCA employed. The difference in results was negligible, but the time it took to complete the A-3C hyperparameter tuning was substantially reduced. Because more time will be available, the A dataset could also run through the same amount of iterations of RandomizedSearchCV as the R dataset (1500). It is assumed that a higher number of iterations, and possible better hyperparameter combinations resulting from it, would ultimately compensate for the loss of information.

Thirty-one features were retained by setting PCA to keep the features that explained 95% of the cumulative variance. Figure 28 represents the cumulative explained variance as the number of components was increased.

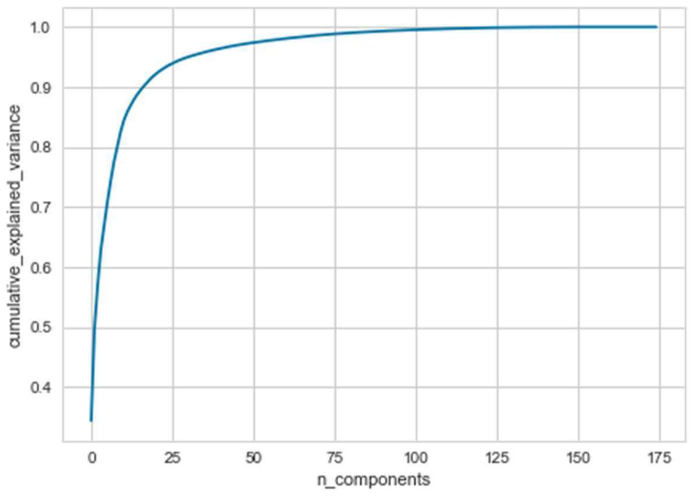


Figure 28. The cumulative variance of principal components

The feature names of transformed principal components were renamed back to the original feature names to evaluate which features contributed the most to the classifier. The PCA process created more than one principal component from the same original feature in some instances. A _dup suffix was added to the feature name in such cases.

3.13.4 Evaluation metrics and the final model

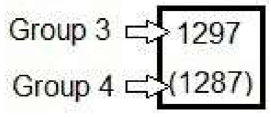
Log loss curves were used extensively during the initial experimental phases. They also served as visual confirmation of model fit during the latter stages. However, they were not included in the final evaluation of results in the following chapter due to the many other metrics that were included. After hyperparameter optimisation for each experiment had been concluded, a final model was created and evaluated. The train_test_split function split the dataset into training and testing datasets. Because the sample set is relatively small and the model was already validated with RandomizedSearchCV, a large testing set was used. Accordingly, train_test_split was set to use 50% as training data and 50% as testing data in the final model. The random seed for train_test_split was kept constant to ensure the same split was created for all the models.

To evaluate the performance of the models, the following metrics were used:

- Cohen Kappa Score.
- Overall accuracy scores of training and testing sets.
- Weighted average precision, recall and f1 for the CLV segments.
- The Drop class's precision, recall, and f1 scores were used for the CLV Drop models, instead of the weighted average.
- Confusion matrix set to normalize for a visual representation of the results.
- Feature importance.
- ROC/AUC curves with the AUC values.

4. Results

This chapter will present the modelling results. The results tables in this chapter will follow the convention in cells where Group 3 is at the top and Group 4 at the bottom in brackets, for instance:



4.1 Clustering results

The first undertaking was creating the target clusters for the Group 3 and Group 4 timeframes. For the four different clustering sets, a large cluster group containing the majority of households, a medium-sized cluster and a smaller cluster was formed. This imbalance is justification for using SMOTE to assist with classifying the smaller classes.

Recall that the values were standardised to fall between 0 and 1 to facilitate comparison between the values.

4.1.1 Monetary Clusters

	Low CLV	Medium CLV	High CLV
Number of Households	1297 (1287)	597 (638)	225 (194)
M Standard Deviation	0.05 (0.05)	0.07 (0.07)	0.13 (0.14)
M Range	0.00 - 0.17 (0.00 - 0.18)	0.17 - 0.42 (0.18 - 0.44)	0.42 - 1.00 (0.45 - 1.00)

Table 17. CLV monetary Clusters (C) in target group 3 (and group 4 in brackets)

As seen in Table 17, the Low CLV had the smallest range but contained the most samples by a fair amount. The Medium CLV cluster had a slightly larger range than the Low CLV one but contained less than half the number of samples. The High CLV cluster had a range of almost three times the other clusters and contained the least number of samples. This difference between the lower two clusters and the High one is further illustrated by its standard deviation, which is, on average, twice as much as the other two.

4.1.2 Loyalty clusters

Recall from section 3.3 that a recency score of 1 is the most recent visit, while a frequency score of 1 indicates a customer that made the most unique store visits. Table 18 is a description of the Loyalty clusters that were formed.

	Low Loyalty	Medium Loyalty	High Loyalty
Number of Households	227 (239)	1446 (1416)	446 (464)
R Standard Deviation	0.20 (0.20)	0.07 (0.06)	0.03 (0.03)
R Range	0.00 - 0.72 (0.00 - 0.72)	0.70 - 1.00 (0.71 - 1.00)	0.77 - 1.00 (0.75 - 1.00)
F Standard Deviation	0.04 (0.05)	0.07 (0.07)	0.15 (0.15)
F Range	0.00 - 0.31 (0.00 - 0.39)	0.00 - 0.28 (0.00 - 0.30)	0.25 - 1.00 (0.27 - 1.00)

Table 18. Loyalty clusters in target group 3 (and group 4 in brackets)

Figure 29 illustrates the L (Loyalty) clusters created where the colour of the data points indicates the different L clusters.

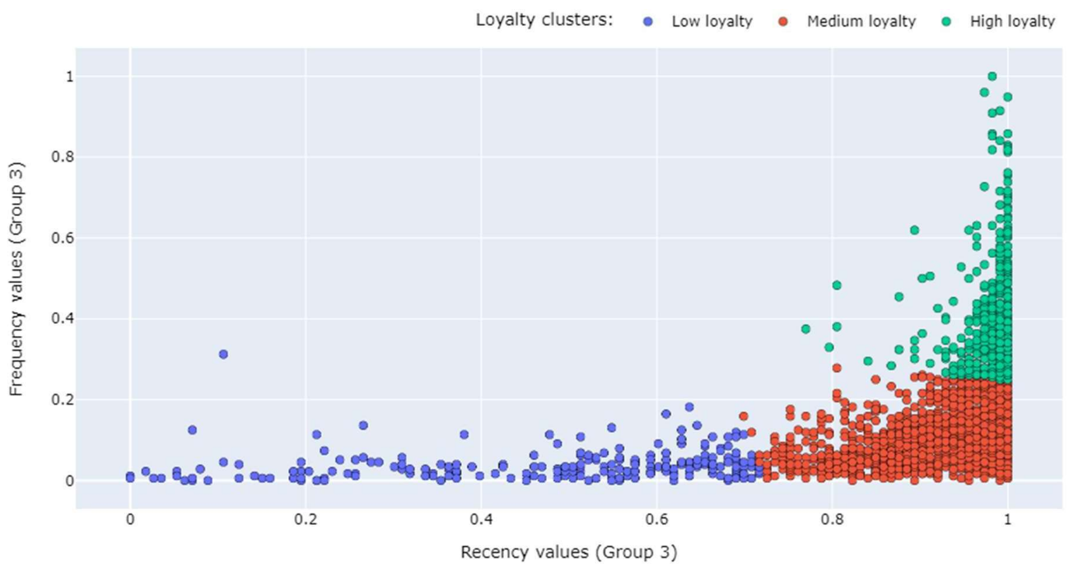


Figure 29. Distribution of households by Loyalty clusters

The Low Loyalty cluster had a far more extensive range than the other two on the Recency axis. Table 18 indicates that the higher F range for this cluster was at 0.28 for Group 3, but it can be seen from the scatter plot in Figure 29 that the maximum value was an outlier, and all the other

Frequency values were below 0.2. This pattern was similar for Group 4, with a Frequency outlier at 0.39, while the rest were below 0.237. This cluster also contained very few samples compared to the Medium Loyalty one.

The Medium Loyalty cluster was the biggest of the three by a fair amount and had the most similar standard deviations for the two dimensions compared to the other two clusters. The High Loyalty clusters had a much wider Frequency range than the other two clusters and were similar to the Low Loyalty cluster in that there is considerable variation in one dimension. In Figure 29, it can be visually determined that the High Loyalty cluster also contained a few samples that could be considered outliers. For this cluster, most of the Recency values were higher than 0.9, even though the lower limit of the range is indicated as 0.77 in Table 18.

4.1.3 Profitability clusters

Recall from section 3.3 that a Monetary score of 1 is the customer that produced the highest turnover, and a Frequency score of 1 indicates a customer that made the most unique store visits. Table 19 is a description of the Profitability clusters that were formed.

	Low Profit	Medium Profit	High Profit
Number of Households	1214 (1235)	685 (678)	220 (206)
F Standard Deviation	0.05 (0.05)	0.09 (0.10)	0.18 (0.18)
F Range	0.00 - 0.26 (0.00 - 0.28)	0.06 - 0.70 (0.07 - 0.61)	0.10 - 1.00 (0.12 - 1.00)
M Standard Deviation	0.05 (0.06)	0.09 (0.10)	0.16 (0.17)
M Range	0.00 - 0.24 (0.00 - 0.27)	0.05 - 0.53 (0.06 - 0.58)	0.08 - 1.00 (0.08 - 1.00)

Table 19. Profitability clusters in target group 3 (and group 4 in brackets)

P (Profitability) clusters are illustrated in Figure 30 through different colours and plotting the Frequency and Monetary values for each household.

The Low Profit cluster contained a significantly larger number of samples than the other two. It also had a far lower range and a smaller standard deviation on both dimensions.

Compared to the Low Profitability, the Medium Profitability cluster's range increased substantially, especially on the Frequency dimension, while the standard deviation almost doubled, and the number of samples was almost halved.

A similar situation to the previous one occurred when comparing the Medium to High Profitability clusters. The Monetary and Frequency spread covered almost the entire possible range, although the sample below a Frequency of 0.2 could be judged as an outlier.

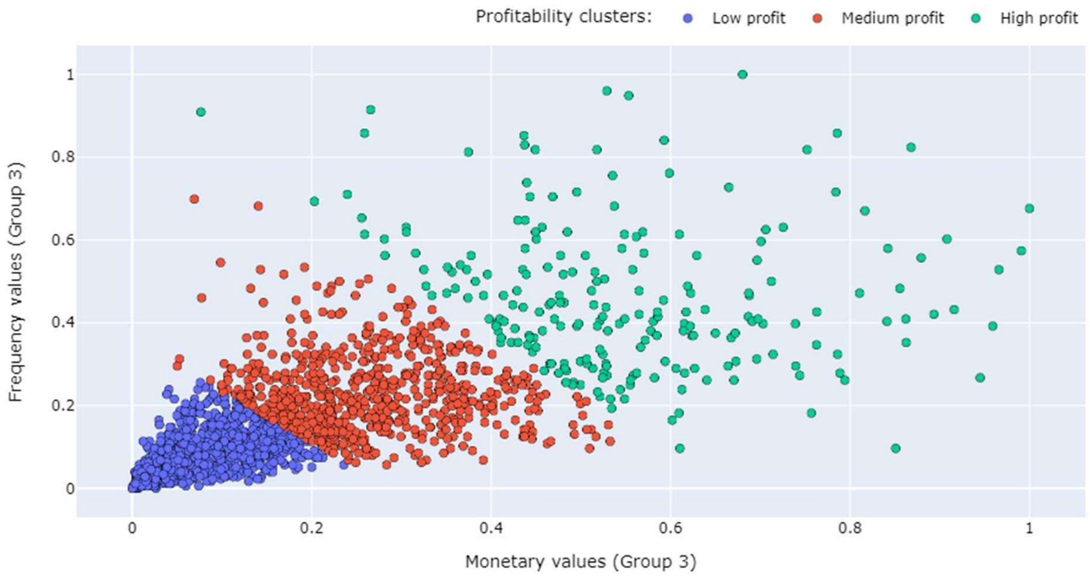


Figure 30. Distribution of households by Profitability clusters

It can be seen in Figure 30 that a general pattern emerges where the two dimensions are inversely related at the extremes of the ranges while showing a higher correlation towards the centre part of the ranges.

4.1.4 Burgeoning clusters

Recall from section 3.3 that a Monetary score of 1 is the customer that produced the highest turnover and that a recency score of 1 is the most recent visit. Table 20 is a description of the B clusters that were formed.

	Low Burgeoning	Medium Burgeoning	High Burgeoning
Number of Households	224 (234)	1426 (1430)	469 (455)
R Standard Deviation	0.20 (0.20)	0.07 (0.07)	0.04 (0.04)
R Range	0.00 - 0.72 (0.00 - 0.72)	0.69 - 1.00 (0.69 - 1.00)	0.77 - 1.00 (0.60 - 1.00)
M Standard Deviation	0.05 (0.05)	0.08 (0.08)	0.15 (0.15)
M Range	0.00 - 0.27 (0.00 - 0.33)	0.00 - 0.28 (0.00 - 0.30)	0.28 - 1.00 (0.29 - 1.00)

Table 20. Burgeoning (B) clusters in target group 3 (and group 4 in brackets)

As shown in Figure 31, the spread of the B scatter plot is very similar to that of the Loyalty one in Figure 29. The number of households in each cluster was also very similar. In Figure 31, the Recency and Monetary values for each household were placed on the plot and clusters were differentiated through colour.

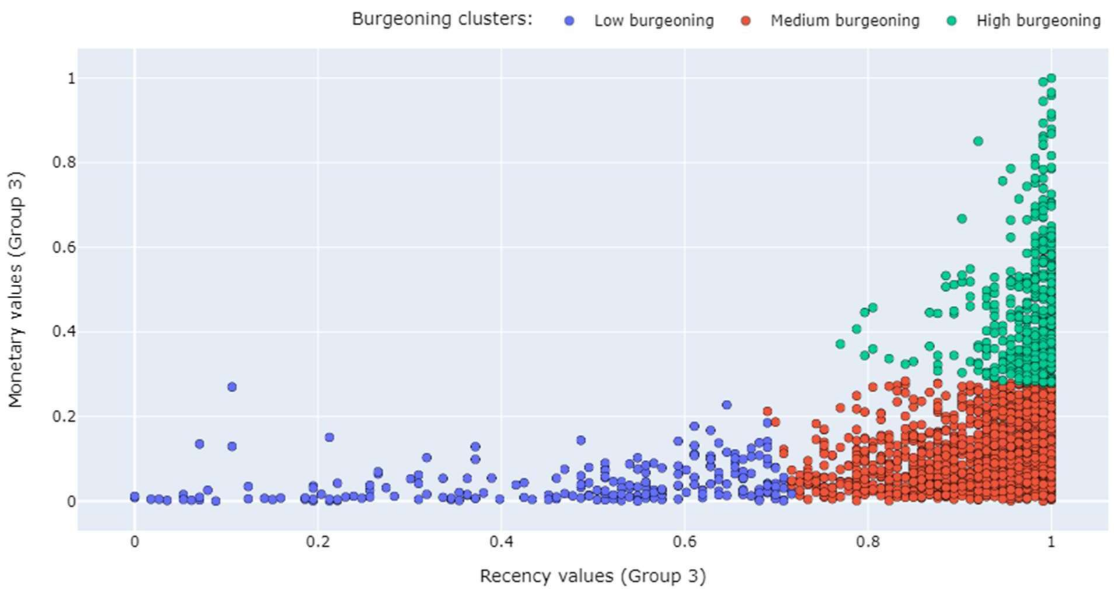


Figure 31. Distribution of households by Burgeoning clusters

The Low Burgeoning cluster has a far wider range than the other two on the Recency axis. This cluster also contains very few samples compared to the Medium Burgeoning one.

The Medium Burgeoning cluster is the biggest of the three by a fair amount and has the most similar standard deviations for the two dimensions compared to the other two clusters. The High Burgeoning cluster had a much wider Monetary range than the other two clusters and is similar to the Low Burgeoning cluster, in that there is considerable variation in one dimension.

4.2 Classification Experiments

PCA was only conducted on the A datasets in order to reduce dimensionality. Notably, the original R, F or M values for the A dataset did not remain as features after PCA. Only the derivatives of the R, F and M features, some *NumStoresVisted_group* categories, and product and temporal features remained in the final A feature set.

4.2.1 CLV

Table 21 is a summary of all the results that were obtained from the CLV classification experiments. Information about the evaluation metrics used can be found in section 2.3.6.

	Cohen's Kappa	Training Accuracy	Accuracy	Weighted Precision	Weighted Recall	Weighted f1-score	Micro AUC
R-3C (R-4C)	0.59 (0.53)	79.79 (75.26)	78.21 (75.57)	0.78 (0.75)	0.78 (0.76)	0.78 (0.75)	0.92 (0.89)
A-3C (A-4C)	0.55 (0.46)	78.19 (63.64)	76.89 (63.77)	0.77 (0.63)	0.77 (0.64)	0.76 (0.63)	0.91 (0.81)
R-3T (R-4T)	0.54 (0.48)	71.10 (65.34)	69.62 (65.38)	0.69 (0.65)	0.70 (0.65)	0.69 (0.65)	0.86 (0.82)
A-3T (A-4T)	0.52 (0.46)	68.93 (63.64)	68.02 (63.77)	0.69 (0.63)	0.68 (0.64)	0.68 (0.63)	0.85 (0.81)
R-3S (R-4S)	0.48 (0.43)	68.56 (60.62)	65.57 (62.45)	0.65 (0.61)	0.66 (0.62)	0.65 (0.61)	0.84 (0.79)
A-3S (A-4S)	0.44 (0.43)	68.08 (58.92)	63.53 (61.98)	0.64 (0.61)	0.64 (0.62)	0.64 (0.61)	0.81 (0.79)
R-3P (R-4P)	0.55 (0.46)	78.09 (72.14)	75.38 (70.19)	0.76 (0.71)	0.75 (0.70)	0.75 (0.70)	0.91 (0.88)
A-3P (A-4P)	0.56 (0.43)	77.24 (81.02)	75.94 (69.53)	0.77 (0.69)	0.76 (0.70)	0.76 (0.69)	0.91 (0.87)
R-3L (R-4L)	0.42 (0.36)	78.38 (73.56)	76.70 (74.25)	0.68 (0.66)	0.77 (0.74)	0.72 (0.69)	0.91 (0.88)
A-3L (A-4L)	0.40 (0.34)	83.57 (76.02)	75.85 (72.64)	0.70 (0.66)	0.76 (0.73)	0.71 (0.68)	0.89 (0.86)
R-3B (R-4B)	0.48 (0.35)	78.75 (76.86)	78.58 (74.06)	0.70 (0.71)	0.79 (0.71)	0.74 (0.69)	0.91 (0.88)
A-3B (A-4B)	0.48 (0.35)	84.99 (78.56)	77.92 (74.34)	0.74 (0.70)	0.78 (0.74)	0.74 (0.69)	0.90 (0.88)

Table 21. Results of the CLV classification experiments

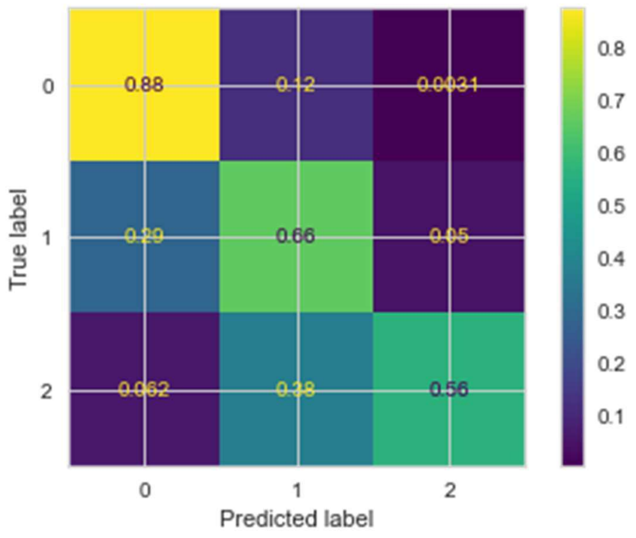


Figure 32. Confusion matrix for the Group 3 Monetary Cluster (C)

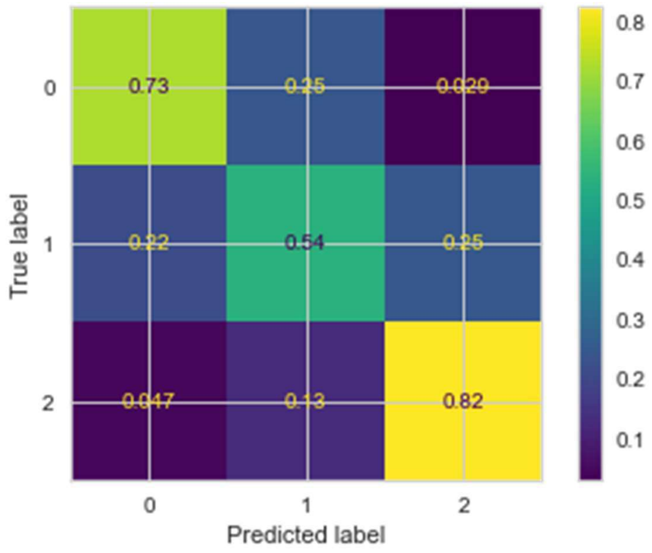


Figure 33. Confusion matrix for the Group 3 Terciles (T)

The Cohen's Kappa score for Group 3 ranged from 0.40 to 0.59, with an average of 0.50. According to the categories proposed by Landis and Koch [131], this indicates a moderate agreement between the actual and predicted values. The range for Group 4 was 0.34 to 0.53, with an average score of 0.43. This result indicates a fair to moderate agreement, with the average being just above the fair category at 0.42. For all the experiments except A-3P, the R experiments had a slightly higher score than the A ones.

When comparing the training and testing accuracies, these values were close to each other for most models. This proximity in values showed that the data fit the models well and that XGBoost could effectively use the data (without underfitting) and without overfitting to the training set. The exceptions to this were the A-3L, A-3B and A-4P experiments, for which the difference was large enough to describe the model as slightly overfitting to the training data. These three experiments would have benefitted from more iterations of RandomizedSearchCV. However, the A experiments took substantially longer to complete the same amount of iterations as the R ones, making an increase infeasible for this study.

An important aspect of this study was to evaluate if adding additional features, in this case, product and temporal customer purchase data and store visit data, would deliver an improved result over using the R, F and M values only. For all but one of the CLV experiments, the A models produced a slightly lower accuracy than the R ones. In the 4C experiment, the R model produced a substantially higher accuracy than the A one. Only the A-3P model had a slightly higher accuracy than the R-3P one. However, the difference is so minimal that it could be attributed to random variation. The same scenario can be seen when considering the Precision, Recall and f1-score, with only the A-3P recall score slightly better than R-3P. Accordingly, it can be concluded that for predicting CLV, the additional features used in this report did not provide any benefit over using only the traditional RFM values.

Considering the individual C monetary clusters, the Low CLV class had an f1-score above 0.80 for all the models, while the Medium and High classes had f1 scores around the 0.60 mark. An exception is the A-3C model, which had substantially lower f1 scores for the Medium and High clusters. Notably, XGBoost was equally effective in classifying the Medium and High clusters, even though the High CLV cluster was almost two-thirds smaller. The implementation of SMOTE might have assisted XGBoost with identifying the High CLV class. The Profitability experiments followed a similar pattern, except for the 4P models, where the High Profitability class had f1 scores around 0.50. Also notable is that the correlation between the C Cluster feature and the P Cluster feature is 88.3 % for Group 3 and 85.6 % for Group 4, showing the similarity between these two sets of experiments.

For the L experiments, XGBoost failed to effectively classify the Low Loyalty class ($f1<0.03$ for all experiments), even though this class had roughly the same number of samples as the High C Cluster. SMOTE appears not to have assisted the classifier in this case. The Medium Loyalty cluster was classified very effectively (all $f1>0.80$), while the High Loyalty had f1-scores between 0.67 and 0.61.

XGBoost also failed to classify the Low Burgeoning Cluster effectively with f1 scores < 0.07 for Groups 3 and 4. The Medium Burgeoning class was classified effectively, with f1-scores ranging from 0.73 to 0.85. The High Burgeoning class was also classified effectively for the R-3B model, with one of the highest f1-scores (0.86) for any given class in this report. For the 4B High Burgeoning class, the f1-score was around the 0.60 mark. As with the CLV Monetary Cluster and the P Cluster results, the B and L Cluster results were very similar. The B and L cluster features correlated 75.1 % and 74.0 % for Groups 3 and 4, respectively.

The micro-AUC scores for the experiments showed a good trade-off between precision and recall in most models. An exception was the S models, which had a notably lower AUC score than the other experiments.

Tables 22 and 23 describe the feature importance by gain for the R and A datasets.

	Features (Gain)			Number of features used
	1	2	3	
R-3C	Monetary (29.06)	Frequency (8.75)		2
R-4C	Monetary (33.48)			1
R-3T	Monetary (29.09)	Recency (9.96)	Frequency (9.90)	3
R-4T	Monetary (40.54)			1
R-3S	Monetary (25.63)	Frequency (7.40)	Recency (5.65)	3
R-4S	Monetary (47.96)	Frequency (25.73)		2
R-3P	Monetary (31.83)	Frequency (11.48)	Recency (6.41)	3
R-4P	Monetary (38.79)	Frequency (15.34)		2
R-3L	Frequency (24.52)	Recency (9.72)	Monetary (9.62)	3
R-4L	Frequency (13.76)	Recency (5.95)	Monetary (2.75)	3
R-3B	Monetary (27.75)	Frequency (11.50)	Recency (7.06)	3
R-4B	Monetary (10.85)	Recency (4.41)	Frequency (4.41)	3

Table 22. Feature importance by gain for the R CLV classification experiments

	Features (Gain)					Number of features used
	1	2	3	4	5	
A-3C	Profit Cluster_Low profit (29.21)	Profit Cluster_Medium profit (14.53)	Tercile_1 (13.74)	NumberOfStoresVisted_2 (13.06)	Recency Cluster_1_dup (12.86)	6
A-4C	Profit Cluster_Low profit (28.35)	Burgeoning_Low burgeoning (16.91)	NumberOfStoresVisted_2 (9.81)	Profit Cluster_Medium profit (9.34)	Tercile_1 (9.22)	16
A-3T	Profit Cluster_Low profit (71.79)	Burgeoning_Medium burgeoning (39.63)	NumberOfStoresVisted_2 (14.58)			3
A-4T	Profit Cluster_Low profit (28.35)	Burgeoning_Low burgeoning (16.91)	NumberOfStoresVisted_2 (9.81)	Profit Cluster_Medium profit (9.34)	Tercile_1 (9.22)	16
A-3S	Profit Cluster_Low profit (56.35)	Recency Cluster_0 (15.42)				2
A-4S	Profit Cluster_Low profit (54.38)	Burgeoning_Low burgeoning (15.33)	NumberOfStoresVisted_5_dup (14.24)			3
A-3P	Profit Cluster_Low profit (34.20)	Burgeoning_Medium burgeoning (16.57)	Profit Cluster_Medium profit (14.17)	Tercile_1 (10.86)	RFM_3D_Cluster_0 (10.71)	5
A-4P	Profit Cluster_Low profit (13.21)	Tercile_1 (5.00)	Profit Cluster_Medium profit (4.37)	RFM_3D_Cluster_0 (4.08)	CLV Segment_Low (3.84)	31
A-3L	Profit Cluster_Low profit (12.82)	Burgeoning_Medium burgeoning (4.87)	Tercile_1 (4.69)	Recency Cluster_1 (4.48)	CLV Segment_Medium (3.75)	31
A-4L	Profit Cluster_Low profit (10.47)	Tercile_1 (5.43)	Burgeoning_Low burgeoning (4.76)	NumberOfStoresVisted_5 (3.97)	Profit Cluster_Medium profit (3.88)	5
A-3B	Profit Cluster_Low profit (13.11)	Profit Cluster_Medium profit (4.83)	Tercile_1 (3.82)	NumberOfStoresVisted_2 (3.66)	RFM_3D_Cluster_1 (3.60)	31
A-4B	Profit Cluster_Low profit (12.69)	Recency Cluster_1_dup (6.24)	Burgeoning_Medium burgeoning (4.86)	Tercile_1 (4.72)	Profit Cluster_Medium profit (4.51)	31

Table 23. Feature importance by gain for the A CLV classification experiments

For all the R experiments where it was used, the Monetary feature was the most important and provided the most gain. In general, for all R experiments, Frequency was more or equally important than Recency.

The ‘Profit Cluster_Low profit’ feature was the most important feature in all the A experiments. The ‘Profit Cluster_Medium profit’ was also in the top 5 features for several models. The Medium and Low Burgeoning cluster and Tercile_1 features were also significant contributors to gain in most models.

Notably, the only features not derived from the RFM values present in the top 5 were NumberOfStoresVisted_2 and NumberOfStoresVisted_5, both created from the one-hot encoding process of NumberOfStoresVisted. The NumberOfStoresVisted_2 signifies customers who visited one or two different stores during the period, while NumberOfStoresVisted_5 signifies those who visited between seven and eleven unique stores. Other important features in the top 5 were those derived from the Recency Cluster, RFM_3D Cluster and CLV Segments. Most models relied on six or fewer features in the final model, with the exceptions being A-3L, A-3B and A-4P. These three were the models (mentioned above) that would have benefited from more iterations of hyperparameter optimisation. Thus, it can be assumed that given more iterations of RandomizedSearchCV, all of the A models would have ended with just a few important features.

In general, the results for the Group 4 prediction were only slightly lower than that of Group 3. This similarity of results is a notable finding since it was possible to classify the Group 4 sets with almost the same level of accuracy as Group 3, with only A-4C being an exception.

If a choice had to be made on which of these CLV values to consider, it would be advisable to choose the Monetary Cluster (C) models over the Profitability (P) ones since the metrics were slightly better and the model slightly less complex. These two model subsets are very similar, and if a frequency dimension is requested to be included, it would be just as effective. A choice between the Loyalty (L) and Burgeoning (B) models would favour the Burgeoning due to the slightly better results. The Burgeoning model also incorporated the Monetary value, which remains the most crucial consideration for profitability. The metrics were slightly more favourable for Tercile (T) models than the Segment (S) ones. The Tercile model is also far less complex to construct and easy to explain to stakeholders because it is a direct measure of income. Therefore, the Tercile models would be considered over the Segment ones.

As can be seen in Figures 32 and 33, there was also considerable variation in the accuracy with which the different classes. It can be seen in Figure 32 that the Monetary Cluster model was much better at predicting the 0 class than the other classes. Conversely, the Tercile model depicted in Figure 33 was much better at predicting the 2 class than the other classes. Accordingly, the choice which CLV to consider would also depend on the CRM strategy one intends to follow. One strategy might require targeting lower CLV values, while another might aim to retain the higher CLV clients.

Overall, the results indicate that Monetary Cluster (C) and Burgeoning Cluster (B) models would be the best choices and should be considered depending on the final strategy.

4.2.2 CLV Drop

Table 24 is a summary of results that were obtained from the CLV Drop classification experiments.

	Training Accuracy	Test Accuracy	Drop class Precision	Drop class Recall	Drop class f1-score
R-3TD (R-4TD)	59.66 (66.85)	58.17 (58.31)	0.33 (0.34)	0.81 (0.59)	0.47 (0.43)
A-3TD (A-4TD)	72.78 (67.00)	64.37 (53.94)	0.34 (0.32)	0.60 (0.66)	0.43 (0.43)
R-3SD (R-4SD)	64.56 (61.04)	61.88 (54.15)	0.30 (0.34)	0.72 (0.76)	0.42 (0.47)
A-3SD (A-4SD)	73.57 (55.70)	66.67 (48.48)	0.33 (0.31)	0.71 (0.76)	0.45 (0.44)
R-3LD (R-4LD)	69.91 (63.28)	67.27 (58.72)	0.25 (0.23)	0.76 (0.81)	0.38 (0.36)
A-3LD (A-4LD)	85.76 (83.51)	76.26 (76.90)	0.28 (0.31)	0.52 (0.49)	0.36 (0.36)
R-3BD (R-4BD)	75.07 (74.48)	76.92 (74.26)	0.35 (0.32)	0.93 (0.93)	0.51 (0.48)
A-3BD (A-4BD)	82.91 (81.20)	73.08 (72.65)	0.26 (0.31)	0.57 (0.66)	0.36 (0.42)

Table 24. Results of the CLV Drop experiments

The overall training and testing accuracy was included to evaluate the data's fit to the models. In most of the A models, there was a level of overfitting. The R-4 models also showed slight overfitting to the training data. In general, many Drop models would have benefited from more iterations of RandomizedSearchCV to find more suitable hyperparameters. Accordingly, the CLV Drop experiments will require more iterations of RandomizedSearchCV than the CLV experiments. It is important to recall that the Drop datasets were also smaller than those used in the CLV experiments, making them more prone to overfitting. However, except for the A-4TD experiment, the amount of overfitting is not large enough to state that the results of the other experiments should be ignored, and valuable conclusions can still be made from the results.

The most relevant evaluation metric for the Drop experiments is Recall. The households that dropped were the minority class in all the experiments and, in some instances, made up only a small part of the overall dataset. It would be an effective spend of marketing funds if it can be assured that a firm would effectively target most of the customers who are going to drop, since one then does not have to spend funds on the entire customer set. Precision is less critical because even if the target group for a CRM campaign contains many customers who will not drop, it will not have the same negative implications as, for example, an incorrect medical diagnosis. In summary, the marketing budget will still be used effectively in high CLV Drop scenarios because the Recall results show that the number of customers targeted during a marketing campaign will be a smaller percentage of the whole, while still reaching the vast majority of the Drop customers.

The precision for the Drop models ranged from 0.23 to 0.34, which in general would not be considered very good. However, due to the reason stated in the previous paragraph, this is less important. The low precision was also reflected in the low f1-scores for all these models.

The recall value for the Drop class in the L and B Drop models was relatively high and much better for the R models than the A ones. The SD models' results were almost the same for the R and A datasets. For the TD experiments, only the R-3TD model had the same level of recall as some of the other high-performing Drop experiments. The A-4TD model had a slightly higher recall than the R-4TD one, making it the only experiment where the result was in favour of the A feature set.

Except for the SD experiments, it seems as though XGBoost did not have nearly the same success with the Group 4 Drop experiments as it did with the Group 3 ones. Therefore, it can be concluded that the time gap caused by the Group 3 period adversely affected the ability to classify the Drop class for Group 4. This conclusion contrasts with the CLV experiments, where the different time frames resulted in a negligible difference.

Tables 25 and 26 describe the feature importance by gain for the R and A datasets.

	Features (Gain)			Number of features used
	1	2	3	
R-3TD	Monetary (17.54)	Frequency (8.02)	Recency (6.27)	3
R-4TD	Monetary (25.16)			1
R-3SD	Monetary (20.01)	Recency (23.24)	Frequency (12.00)	3
R-4SD	Monetary (30.37)	Frequency (19.73)	Recency (16.53)	3
R-3LD	Frequency (45.37)	Monetary (27.81)	Recency (23.08)	3
R-4LD	Frequency (47.10)	Recency (24.45)	Monetary (18.27)	3
R-3BD	Frequency (82.28)	Recency (37.76)	Monetary (24.58)	3
R-4BD	Frequency (41.20)	Monetary (18.28)	Recency (17.83)	3

Table 25. Feature importance by gain for the R Drop classification experiments

	Features (Gain)					Number of features used
	1	2	3	4	5	
A-3TD	Tercile_1 (34.33)	NumberOfStoresVisted_4 (22.00)	NumberOfStoresVisted_2 (21.77)	Recency Cluster_1 (21.70)	Profit Cluster_Medium profit (21.34)	30
A-4TD	NumberOfStoresVisted_4_dup (33.98)	Profit Cluster_High profit (28.93)	Recency Cluster_1 (27.95)	Frequency_Cluster_0 (25.27)	Profit Cluster_Medium profit (25.13)	28
A-3SD	Profit Cluster_Medium profit (34.52)	ValuePerSaturday (34.25)	Tercile_2 (33.99)	Value_MEAT_dup2 (32.34)	Value_19_dup (31.40)	29
A-4SD	Value_19 (25.78)	Profit Cluster_Low profit (25.75)	CLV Segment_Medium (23.52)	Frequency_Cluster_1 (22.95)	NumberOfStoresVisted_1 (22.42)	35
A-3LD	NumberOfStoresVisted_5 (46.54)	Profit Cluster_Low profit (44.87)	Monetary_Cluster_1 (43.45)	Frequency_Cluster_1 (33.22)	Recency (32.44)	35
A-4LD	NumberOfStoresVisted_5 (52.02)	Value_GROCERY (43.83)	RFM_3D_Cluster_1 (38.18)	Profit Cluster_Low profit (37.46)	Time_15 (36.84)	35
A-3BD	CLV Segment_Medium (48.33)	RFM_3D_Cluster_1 (38.56)	Frequency_Cluster_1 (34.38)	Spend_MEAT (34.25)	NumberOfStoresVisted_5 (30.21)	33
A-4BD	CLV Segment_Medium (53.57)	Monetary_Cluster_1 (28.61)	Profit Cluster_Low profit (28.15)	BasketsPerMonday (27.93)	RFM_3D_Cluster_1 (27.89)	34

Table 26. Feature importance by gain for the A Drop classification experiments

In the R-LD and R-BD models, Frequency was the most important feature. For all these models, the two remaining features provided roughly the same gain, with the only exception being R-3BD, where the Recency feature provided substantially more gain than the Monetary one. Recall from section 4.2.1 that the R-L models showed a similar trend. In contrast, the R-B model had Monetary as the most important feature, showing that Frequency played a more important role when classifying the BD experiments. Monetary value was most important for the R-TD and R-SD models. The R-4T and R-4TD models used only the Monetary feature to make classifications.

For the AD models, no single feature was dominant in all the models, unlike in the segment prediction experiments. Far more features, including those not associated with the R, F and M values, were in the top 5. NumberOfStoresVisited again played an important role, as did the one-hot encoded values associated with the CLV Segment, Frequency Cluster, Monetary Cluster, Recency Cluster, Profit Cluster and the 3D_RFM cluster.

After PCA, the LD models were the only ones that used any original R, F or M values. The original R value provided the fifth-largest amount of gain in the A-3LD model, but the value was only 11th for gain in the A-4LD model. Excluding the R value for the LD models, the finding that the R, F and M values were not included in the AD models were similar to what was found in the A ones.

In selecting the best model, Burgeoning Drop (BD) would be recommended over Loyalty Drop (LD) because of the high recall value and the fact that it incorporates the Monetary aspect. Furthermore, the Segment Drop (SD) model would be preferred since it was more consistent over both Groups than the Tercile Drop (TD) results. However, due to the arbitrary nature of the Segment Drop classes, the best Drop CLV model to use would be Burgeoning Drop.

5. Conclusion

This chapter will briefly summarise how the models used in this study were created. Then, the results will be used to examine the extent to which the goals of this work were achieved. Finally, some suggestions will be given for possible future research.

5.1 Summary

Individual customer purchase information was extracted from a database, and values were grouped according to each customer. RFM (recency, frequency and monetary value of purchases), temporal, product and store visit information was extracted for each customer. The RFM values with a z-score higher than four were determined to be outliers, and those customers' details were removed from the final dataset. The remaining data were used to construct XGBoost models for predicting future CLV (Customer Lifetime Value) customer segments.

Periods of roughly six months were isolated to serve as the length for CLV analysis timeframes. Four groups were created, with the first one discarded as unrepresentative because customer transactions were not consistent during this period. The second was used to create the features that XGBoost would use to make the predictions. The last two groups contained the target variables to be predicted.

Two data subsets were created, one which contained the RFM values and the other subset which also included the additional information. Principal Component Analysis (PCA) was used to reduce the dimensionality of the latter.

CLV segments were created through K-means clustering, and the following variants were investigated: Monetary Cluster, Loyalty, Profitability, and Burgeoning. CLV Segments were also created by isolating Monetary terciles and adapting a traditional approach for creating weighted-RFM segments. All these variations contained three segments that would become the classes the multivariate classification models predicted.

Models were created to predict if a drop in CLV segment would occur. Unique datasets were created for each of these, because only those customers that could drop in CLV were included. The CLV Drop models were binary classification models since they predicted drop or not.

Hyperparameter optimisation and Cross-Validation with RandomizedSearchCV were used to create models that would generalise to other datasets. Then a final model was produced, and the results were analysed through various evaluation metrics.

5.2 Findings

This work aimed to assess how XGBoost classification models would perform to solve a range of questions involving CLV. This section will address the goals set out in the Introduction chapter.

a) The first goal was to determine how effective XGBoost classifications models are at predicting future CLV. The models successfully predicted the CLV segments formed through clustering, at 78.58% accuracy, and had reasonable success in predicting the CLV tercile segment, at 69.62% accuracy. However, they fared less well at predicting the weighted-RFM segment at 65.57% accuracy.

b) The second was to determine how effective XGBoost classifications models are at predicting if a customer will drop to a lower CLV segment in the future. The models constructed using only RFM values were very good to excellent at predicting which customers would move to a lower CLV, achieving recall of 93% for the clustered segment drop. However, models that included temporal, product and store-visit data fared less well.

c) The third was to draw a comparison in the effectiveness of models created that only use the RFM variables to those which added temporal, product, and other customer transaction information. The added features did not improve the models and were far more complex and time-consuming to construct.

d) The final aim was to determine the effectiveness of the above models in predicting CLV for a period that does not follow directly after the one used as input. The results indicate that the models effectively predicted CLV for a period that does not follow directly after the one used as input features. The models were only slightly better at predicting the immediate future than they were at predicting the period following it.

5.3 Recommendations for Future research

The constructed models relied on relatively small datasets. Additional studies should try and replicate the results with larger datasets. With larger datasets, the Monetary and Profitability CLV Drop models could also be constructed.

Other classifiers should be evaluated on their effectiveness in predicting CLV and CLV drop.

Additional studies that explicitly use customer demographic information to predict CLV and CLV Drop could also be undertaken.

Other researchers should determine if the CLV and CLV drop prediction with XGBoost is as effective (as the ones created here) for business-to-business, pure online retail or brick and mortar-online-hybrid retail CRM datasets.

Finally, these findings can be used to develop a tool (for example, in the form of an API) that firms can use to assist with their CRM strategies.

6. References

[1] V. E. Krivonozhko, A. A. Piskunov, A. V. Lychev, and M. A. Piskunova, "Models and methods for decision making support in the negotiation process," *Scientific and technical information processing,* vol. 38, no. 6, pp. 440-448, 2012, doi: 10.3103/S0147688211060050.

[2] P. Kotler, "Marketing during Periods of Shortage," *Journal of marketing,* vol. 38, no. 3, pp. 20-29, 1974, doi: 10.1177/002224297403800305.

[3] "CRM 101: What is CRM?" Salesforce.com. https://www.salesforce.com/ca/crm/what-is-crm/ (accessed July 04, 2022).

[4] "Machine Learning Glossary." developers.google.com. https://developers.google.com/machine-learning/glossary?hl=en#feature (accessed July 04, 2022).

[5] W. Kenton. "Fast-Moving Consumer Goods (FMCG)." investopedia.com. https://www.investopedia.com/terms/f/fastmoving-consumer-goods-fmcg.asp (accessed July 05, 2022).

[6] J. Brownlee. "SMOTE for Imbalanced Classification with Python." machinelearningmastery.com. https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/ (accessed May, 2022, 2022).

[7] V. Kumar and W. Reinartz, *Customer Relationship Management Concept, Strategy, and Tools* (Customer Relationship Management). Berlin, Heidelberg: Springer Berlin Heidelberg, 2018.

[8] "Customer Relationship Management Market Size, Share & Trends Analysis Report By Solution (Customer Service, Customer Experience Management), By Deployment, By Enterprise Size, By End Use, And Segment Forecasts, 2021 - 2028." grandviewresearch.com. https://www.grandviewresearch.com/industry-analysis/customer-relationship-management-crm-market (accessed September 04, 2021).

[9] J. Gantz, G. Murry, D. Schubmehl, D. Vesset, and M. Wardley, "A Trillion-Dollar Boost: The Economic Impact of AI on Customer Relationship Management," IDC, 5 Speen Street Framingham, MA 01701 USA 508.872.8200, White Paper 2017. [Online]. Available: https://www.salesforce.com/content/dam/web/en_us/www/documents/white-papers/the-economic-impact-of-ai.pdf

[10] "2018 Sales Force Annual Report," 2018. [Online]. Available: https://s1.q4cdn.com/454432842/files/doc_financials/2018/Salesforce-FY18-Annual-Report.pdf

[11] "Artificial Intelligence (AI) Market - Global Industry Analysis," Zion Market Research, ZMR-3420, October, 2018 2018. [Online]. Available: https://www.zionmarketresearch.com/report/artificial-intelligence-market-for-crm

[12] A. M. Hughes, *Strategic database marketing : the masterplan for starting and managing a profitable, customer-based marketing program*. Chicago, Ill.: Probus Pub. Co. (in English), 1994.

[13] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *International Conference on Knowledge Discovery and Data Mining*, 2016: ACM, in KDD '16, pp. 785-794, doi: 10.1145/2939672.2939785.

[14] M. S. Seyfioğlu and M. U. Demirezen, "A hierarchical approach for sentiment analysis and categorization of Turkish written customer relationship management data," in *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2017: IEEE, pp. 361-365.

[15] Y. Zhuang, "Research on E-commerce Customer Churn Prediction Based on Improved Value Model and XG-Boost Algorithm," *Management Science and Engineering,* vol. 12, no. 3, pp. 51-56, 2018.

[16] M. M. Monastyrskaya and V. I. Soloviev, "Improving Customer Relationship Management based on Intelligent Analysis of User Behavior Patterns," in *2020 13th International Conference "Management of large-scale system development"(MLSD)*, 2020: IEEE, pp. 1-4.

[17] I. M. Al-Zuabi, A. Jafar, and K. Aljoumaa, "Predicting customer's gender and age depending on mobile phone data," *Journal of Big Data,* vol. 6, no. 1, pp. 1-16, 2019.

[18] "What The Decline Of Brick-And-Mortar Sales Means For Retail Marketing." forbes.com. https://www.forbes.com/sites/forbescommunicationscouncil/2021/08/30/what-the-decline-of-brick-and-mortar-sales-means-for-retail-marketing/?sh=6af48ab4e303 (accessed July 18, 2022).

[19] V. Kumar, R. Venkatesan, T. Bohling, and D. Beckmann, "Practice Prize Report--The Power of CLV: Managing Customer Lifetime Value at IBM," *Marketing science (Providence, R.I.),* vol. 27, no. 4, pp. 585-599, 2008, doi: 10.1287/mksc.1070.0319.

[20] R. S. Tedlow and G. G. Jones, *The Rise and Fall of Mass Marketing (RLE Marketing)* (Routledge Library Editions: Marketing). London: Taylor and Francis, 2014.

[21] P. Kotler, G. M. Armstrong, and M. Tait, *Principles of marketing: Global and Southern African perspectives*. Pearson Education South Africa, 2010.

[22] S. Ivanovic, K. Mikinac, and L. Perman, "CRM development in hospitality companies for the purpose of increasing the competitiveness in the tourist market," *UTMS Journal of Economics,* vol. 2, no. 1, pp. 59-68, 2011.

[23] A. Zuckerman. "Introduction to the history of CRM software." comparecamp.com. https://comparecamp.com/introduction-history-crm-software/ (accessed September 04, 2014).

[24] "Gartner Says CRM Became the Largest Software Market in 2017 and Will Be the Fastest Growing Software Market in 2018." gartner.com. https://www.gartner.com/en/newsroom/press-releases/2018-04-10-gartner-says-crm-became-the-largest-software-market-in-2017-and-will-be-the-fastest-growing-software-market-in-2018 (accessed April 10, 2022).

[25] M. Fayerman, "Customer Relationship Management," *New Directions for Institutional Research,* vol. 2002, no. 113, pp. 57-68, 2002/03/01 2002, doi: https://doi.org/10.1002/ir.37.

[26] S. Gupta *et al.*, "Modeling Customer Lifetime Value," *Journal of service research : JSR,* vol. 9, no. 2, pp. 139-155, 2006, doi: 10.1177/1094670506293810.

[27] R. Rahmadianti, A. Dhini, and E. Laoh, "Estimating Customer Lifetime Value using LRFM Model in Pharmaceutical and Medical Device Distribution Company," in *2020 International Conference on ICT for Smart Society (ICISS)*, 19-20 Nov. 2020 2020, vol. CFP2013V-ART, pp. 1-5, doi: 10.1109/ICISS50791.2020.9307592.

[28] M. Haenlein, A. M. Kaplan, and A. J. Beeser, "A Model to Determine Customer Lifetime Value in a Retail Banking Context," *European Management Journal,* vol. 25, no. 3, pp. 221-234, 2007/06/01/ 2007, doi: https://doi.org/10.1016/j.emj.2007.01.004.

[29] S. Peker, A. Kocyigit, and P. E. Eren, "LRFMP model for customer segmentation in the grocery retail industry: a case study," *Marketing intelligence & planning,* vol. 35, no. 4, pp. 544-559, 2017, doi: 10.1108/MIP-11-2016-0210.

[30] F. Safari, N. Safari, and G. A. Montazer, "Customer lifetime value determination based on RFM model," *Marketing intelligence & planning,* vol. 34, no. 4, pp. 446-461, 2016, doi: 10.1108/MIP-03-2015-0060.

[31] S. Gupta, D. R. Lehmann, and J. A. Stuart, "Valuing Customers," *Journal of marketing research,* vol. 41, no. 1, pp. 7-18, 2004, doi: 10.1509/jmkr.41.1.7.25084.

[32] R. K. Srivastava, T. A. Shervani, and L. Fahey, "Market-based assets and shareholder value: A framework for analysis," *Journal of marketing,* vol. 62, no. 1, pp. 2-18, 1998.

[33] R. T. Rust, T. Ambler, G. S. Carpenter, V. Kumar, and R. K. Srivastava, "Measuring Marketing Productivity: Current Knowledge and Future Directions," *Journal of marketing,* vol. 68, no. 4, pp. 76-89, 2004, doi: 10.1509/jmkg.68.4.76.42721.

[34] R. Venkatesan and V. Kumar, "A Customer Lifetime Value Framework for Customer Selection and Resource Allocation Strategy," *Journal of marketing,* vol. 68, no. 4, pp. 106-125, 2004, doi: 10.1509/jmkg.68.4.106.42728.

[35] N. Glady, B. Baesens, and C. Croux, "Modeling churn using customer lifetime value," *European journal of operational research,* vol. 197, no. 1, pp. 402-411, 2009, doi: 10.1016/j.ejor.2008.06.027.

[36] J. A. McCarty and M. Hastak, "Segmentation approaches in data-mining: A comparison of RFM, CHAID, and logistic regression," *Journal of business research,* vol. 60, no. 6, pp. 656-662, 2007.

[37] S. F. Peter, G. S. H. Bruce, and L. Ka Lok, "RFM and CLV: Using Iso-Value Curves for Customer Base Analysis," *Journal of marketing research,* vol. 42, no. 4, pp. 415-430, 2005, doi: 10.1509/jmkr.2005.42.4.415.

[38] J. Wu *et al.*, "An Empirical Study on Customer Segmentation by Purchase Behaviors Using a RFM Model and K-Means Algorithm," *Mathematical problems in engineering,* vol. 2020, 2020, doi: 10.1155/2020/8884227.

[39] S. Hidayat, R. Rismayati, M. Tajuddin, and N. L. P. Merawati, "Segmentation of university customers loyalty based on RFM analysis using fuzzy c-means clustering," *Jurnal Teknologi dan Sistem Komputer,* vol. 8, no. 2, pp. 133-139, 2020, doi: 10.14710/jtsiskom.8.2.2020.133-139.

[40] A. G. Aggarwal and S. Yadav, "Customer Segmentation Using Fuzzy-AHP and RFM Model," in *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 4-5 June 2020 2020, pp. 77-80, doi: 10.1109/ICRITO48877.2020.9197903. [Online]. Available: https://ieeexplore.ieee.org/document/9197903/

[41] M. Najib and H. Mulyati, "CUSTOMER SEGMENTATION ANALYSIS BASED ON THE CUSTOMER LIFETIME VALUE METHOD," *Jurnal Aplikasi Manajemen,* vol. 17, no. 3, pp. 408-415, 2019, doi: 10.21776/ub.jam.2019.017.03.04.

[42] J. R. Bernat, "Modelling customer lifetime value in a continuous, non-contractual time setting," Masters Thesis, Econometrics & Management Science, Erasmus University Rotterdam, 2018.

[43] S. Chen, "Estimating customer lifetime value using machine learning techniques," in *Data mining*, 2018, vol. 17.

[44] D. C. Schmittlein, D. G. Morrison, and R. Colombo, "Counting Your Customers: Who-Are They and What Will They Do Next?," *Management science,* vol. 33, no. 1, pp. 1-24, 1987, doi: 10.1287/mnsc.33.1.1.

[45] P. Jasek, L. Vrana, L. Sperkova, Z. Smutny, and M. Kobulsky, "Predictive performance of customer lifetime value models in e-commerce and the use of non-financial data," *Prague Economic Papers,* vol. 28, no. 6, pp. 648-669, 2019.

[46] P. E. Pfeifer and R. L. Carraway, "Modeling customer relationships as Markov chains," *Journal of interactive marketing,* vol. 14, no. 2, pp. 43-55, 2000.

[47] J. R. Miglautsch, "Thoughts on RFM scoring," *Journal of Database Marketing & Customer Strategy Management,* vol. 8, no. 1, pp. 67-72, 2000.

[48] F. J. Mulhern, "Customer profitability analysis: Measurement, concentration, and research directions," *Journal of Interactive Marketing,* vol. 13, no. 1, pp. 25-40, 1999/01/01/ 1999, doi: https://doi.org/10.1002/(SICI)1520-6653(199924)13:1<25::AID-DIR3>3.0.CO;2-L.

[49] D. Bell, J. Deighton, W. J. Reinartz, R. T. Rust, and G. Swartz, "Seven barriers to customer equity management," *Journal of service Research,* vol. 5, no. 1, pp. 77-85, 2002.

[50] A. Dursun and M. Caber, "Using data mining techniques for profiling profitable hotel customers: An application of RFM analysis," *Tourism management perspectives,* vol. 18, pp. 153-160, 2016, doi: 10.1016/j.tmp.2016.03.001.

[51] Y.-H. Hu and T.-W. Yeh, "Discovering valuable frequent patterns based on RFM analysis without customer identification information," *Knowledge-based systems,* vol. 61, pp. 76-88, 2014, doi: 10.1016/j.knosys.2014.02.009.

[52] C.-H. Cheng and Y.-S. Chen, "Classifying the segmentation of customer value via RFM model and RS theory," *Expert systems with applications,* vol. 36, no. 3, pp. 4176-4184, 2009, doi: 10.1016/j.eswa.2008.04.003.

[53] S. M. S. Hosseini, A. Maleki, and M. R. Gholamian, "Cluster analysis using data mining approach to develop CRM methodology to assess the customer loyalty," *Expert Systems with Applications,* vol. 37, no. 7, pp. 5259-5264, 2010.

[54] R. Srivastava, "Identification of customer clusters using rfm model: A case of diverse purchaser classification," *International Journal of Business Analytics and Intelligence,* vol. 4, no. 2, pp. 45-50, 2016.

[55] A. Hamzehei, M. Fathian, M. Gholamian, and H. Farvaresh, "A new methodology to study customer electrocardiogram using RFM analysis and clustering," *Management Science Letters,* vol. 1, no. 2, pp. 139-148, 2011.

[56] J. Wu and Z. Lin, "Research on customer segmentation model by clustering," presented at the Proceedings of the 7th international conference on Electronic commerce, Xi'an, China, 2005. [Online]. Available: https://doi-org.ezproxy.uct.ac.za/10.1145/1089551.1089610.

[57] W. J. Reinartz and V. Kumar, "On the profitability of long-life customers in a noncontractual setting: An empirical investigation and implications for marketing," *Journal of marketing,* vol. 64, no. 4, pp. 17-35, 2000.

[58] P. P. Pramono, I. Surjandari, and E. Laoh, "Estimating Customer Segmentation based on Customer Lifetime Value Using Two-Stage Clustering Method," in *2019 16th International Conference on Service Systems and Service Management (ICSSSM)*, 13-15 July 2019 2019, pp. 1-5, doi: 10.1109/ICSSSM.2019.8887704. [Online]. Available: https://ieeexplore.ieee.org/document/8887704/

[59] H.-C. Chang and H.-P. Tsai, "Group RFM analysis as a novel framework to discover better customer consumption behavior," *Expert Systems with Applications,* vol. 38, no. 12, pp. 14499-14513, 2011.

[60] K. Chen, Y.-H. Hu, and Y.-C. Hsieh, "Predicting customer churn from valuable B2B customers in the logistics industry: a case study," *Information systems and e-business management,* vol. 13, no. 3, pp. 475-494, 2015, doi: 10.1007/s10257-014-0264-1.

[61] R. Heldt, C. S. Silveira, and F. B. Luce, "Predicting customer value per product: From RFM to RFM/P," *Journal of business research,* vol. 127, pp. 444-453, 2021, doi: 10.1016/j.jbusres.2019.05.001.

[62] B. Cengiz. "CRM Analytics, RFM Analysis." medium.com. https://medium.com/codex/crm-analytics-rfm-analysis-90aa2ae55fbc (accessed May 11, 2022).

[63] B. Stone, *Successful Direct Marketing Methods, Lincoln-wood* (IL: NTC Business Books). 1995, pp. 37-57.

[64] J.-T. Wei, S.-Y. Lin, C.-C. Weng, and H.-H. Wu, "A case study of applying LRFM model in market segmentation of a children's dental clinic," *Expert systems with applications,* vol. 39, no. 5, pp. 5529-5533, 2012, doi: 10.1016/j.eswa.2011.11.066.

[65] M. Khajvand and M. J. Tarokh, "Analyzing customer segmentation based on customer value components (case study: a private bank)," *Advances in Industrial Engineering,* vol. 45, no. Special Issue, pp. 79-93, 2011.

[66] T. Hong and E. Kim, "Segmenting customers in online stores based on factors that affect the customer’s intention to purchase," *Expert systems with applications,* vol. 39, no. 2, pp. 2127-2131, 2012, doi: 10.1016/j.eswa.2011.07.114.

[67] U. Aloysius Matz Teguh, "Customer Loyalty Clustering Model Using K-Means Algorithm with LRIFMQ Parameters," *Inform: Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi,* vol. 5, no. 2, pp. 54-61, 2020, doi: 10.25139/inform.v0i1.2691.

[68] M. Ray and B. Mangaraj, "AHP Based Data Mining for Customer Segmentation Based on Customer Lifetime Value," *Integrated Intelligent Research (IIR),* vol. 5, no. 1, pp. 28 - 34, 2016.

[69] P. Amin, T. MohammadJafar, and A. Hossein, "Combining data mining and group decision making in retailer segmentation based on LRFMP variables," *International journal of industrial engineering & production research,* vol. 25, no. 3, pp. 197-206, 2014.

[70] P. J. Van Laarhoven and W. Pedrycz, "A fuzzy extension of Saaty's priority theory," *Fuzzy sets and Systems,* vol. 11, no. 1-3, pp. 229-241, 1983.

[71] C. Marcus, "A practical yet meaningful approach to customer segmentation," *Journal of Consumer Marketing,* vol. 15, no. 5, pp. 494-504, 1998, doi: 10.1108/07363769810235974.

[72] U. Kaymak, "Fuzzy target selection using RFM variables," in *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference (Cat. No. 01TH8569)*, 2001, vol. 2: IEEE, pp. 1038-1043, doi: 10.1109/NAFIPS.2001.944748. [Online]. Available: https://ieeexplore.ieee.org/document/944748/

[73] R. Kahan, "Using database marketing techniques to enhance your one-to-one marketing initiatives," *Journal of Consumer Marketing,* vol. 15, no. 5, pp. 491-493, 1998, doi: 10.1108/07363769810235965.

[74] R. Elsner, M. Krafft, and A. Huchzermeier, "Optimizing Rhenania's mail-order business through dynamic multilevel modeling (DMLM)," *Interfaces,* vol. 33, no. 1, pp. 50-66, 2003.

[75] P. S. Fader, B. G. Hardie, and K. L. Lee, "“Counting your customers” the easy way: An alternative to the Pareto/NBD model," *Marketing science,* vol. 24, no. 2, pp. 275-284, 2005.

[76] P. Baecke and D. Van den Poel, "Data augmentation by predicting spending pleasure using commercially available external data," *Journal of intelligent information systems,* vol. 36, no. 3, pp. 367-383, 2011.

[77] D. L. Olson, Q. Cao, C. Gu, and D. Lee, "Comparison of customer response models," *Service business,* vol. 3, no. 2, pp. 117-130, 2009, doi: 10.1007/s11628-009-0064-8.

[78] M. Fitzpatrick, "DATA MANAGEMENT. Statistical Analysis for Direct Marketers-In Plain English," *DIRECT MARKETING-GARDEN CITY-,* vol. 64, no. 4, pp. 54-56, 2001.

[79] R. C. Blattberg, B.-D. Kim, and S. A. Neslin, "RFM Analysis," in *Database Marketing: Analyzing and Managing Customers*, R. C. Blattberg, B.-D. Kim, and S. A. Neslin Eds. New York, NY: Springer New York, 2008, pp. 323-337.

[80] R. S. Swift, *Accelerating customer relationships: Using CRM and relationship technologies*. Prentice Hall Professional, 2001.

[81] B. N. Chagas, J. Viana, O. Reinhold, F. M. Lobato, A. F. Jacob Jr, and R. Alt, "A literature review of the current applications of machine learning and their practical implications," in *Web Intelligence*, 2020, no. Preprint: IOS Press, pp. 1-15.

[82] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281-297.

[83] F. E. Grubbs, "Procedures for Detecting Outlying Observations in Samples," *Technometrics,* vol. 11, no. 1, pp. 1-21, 1969, doi: 10.1080/00401706.1969.10490657.

[84] W. Badr. "5 Ways to Detect Outliers/Anomalies That Every Data Scientist Should Know (Python Code)." towardsdatascience.com. https://towardsdatascience.com/5-ways-to-detect-outliers-that-every-data-scientist-should-know-python-code-70a54335a623 (accessed May 22, 2022).

[85] R. E. Shiffler, "Maximum Z Scores and Outliers," *The American statistician,* vol. 42, no. 1, pp. 79-80, 1988, doi: 10.1080/00031305.1988.10475530.

[86] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

[87] "Box Plots in Python." plotly.com. https://plotly.com/python/box-plots/ (accessed May 22, 2022).

[88] A. C. Müller and S. Guido, *Introduction to machine learning with Python: a guide for data scientists*. O'Reilly Media, Inc., 2016.

[89] V. N. G. Raju, K. P. Lakshmi, V. M. Jain, A. Kalidindi, and V. Padma, "Study the Influence of Normalization/Transformation process on the Accuracy of Supervised Classification," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 20-22 Aug. 2020 2020, pp. 729-735, doi: 10.1109/ICSSIT48917.2020.9214160.

[90] Z. Zhang. "Understand Data Normalization in Machine Learning." towardsdatascience.com. https://towardsdatascience.com/understand-data-normalization-in-machine-learning-8ff3062101f0 (accessed May 14, 2022).

[91] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of machine learning research,* 2011, doi: 10.5555/1953048.2078195.

[92] "Normalization." codecademy.com. https://www.codecademy.com/article/normalization (accessed May 14, 2022).

[93] J. Han, M. Kamber, and J. Pei, "3 - Data Preprocessing," in *Data Mining (Third Edition)*, J. Han, M. Kamber, and J. Pei Eds. Boston: Morgan Kaufmann, 2012, pp. 83-124.

[94] C. McCue, *Data mining and predictive analysis: Intelligence gathering and crime analysis*. Butterworth-Heinemann, 2014.

[95] C. M. McCue, "Preface," in *Data Mining and Predictive Analysis*, C. McCue Ed. Burlington: Butterworth-Heinemann, 2007, pp. xv-xxiii.

[96] D. S. Starnes, D. Yates, and D. S. Moore, *The practice of statistics*. Macmillan, 2010.

[97] J. Brownlee. "Why One-Hot Encode Data in Machine Learning?" machinelearningmastery.com. https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/ (accessed May 14, 2022).

[98] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical transactions of the Royal Society of London. Series A: Mathematical, physical, and engineering sciences,* vol. 374, no. 2065, pp. 20150202-20150202, 2016, doi: 10.1098/rsta.2015.0202.

[99] Z. Jaadi. "A Step-by-Step Explanation of Principal Component Analysis (PCA)." builtin.com. https://builtin.com/data-science/step-step-explanation-principal-component-analysis (accessed May 14, 2022).

[100] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *The Journal of artificial intelligence research,* vol. 16, pp. 321-357, 2002, doi: 10.1613/jair.953.

[101] S. Kumar. "Stop using SMOTE to handle all your Imbalanced Data." towardsdatascience.com. https://towardsdatascience.com/stop-using-smote-to-handle-all-your-imbalanced-data-34403399d3be (accessed May 22, 2022).

[102] H. Ma *et al.*, "Integrating Growth and Environmental Parameters to Discriminate Powdery Mildew and Aphid of Winter Wheat Using Bi-Temporal Landsat-8 Imagery," *Remote Sensing,* vol. 11, p. 846, 04/08 2019, doi: 10.3390/rs11070846.

[103] S. Wager. "Testing Classification on Oversampled Imbalance Data." stats.stackexchange.com/. https://stats.stackexchange.com/questions/60180/testing-classification-on-oversampled-imbalance-data/111552#111552 (accessed June 15, 2022).

[104] J. Delua. "Supervised vs. Unsupervised Learning: What's the Difference?" ibm.com. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning (accessed May 22, 2022).

[105] R. W. Sembiring Brahmana, F. A. Mohammed, and K. Chairuang, "Customer Segmentation Based on RFM Model Using K-Means, K-Medoids, and DBSCAN Methods," *Lontar komputer,* vol. 11, no. 1, pp. 32-43, 2020, doi: 10.24843/LKJITI.2020.v11.i01.p04.

[106] Y. R. Rahadian and B. Syairudin, "Segmentation Analysis of Students in X Course with RFM Model and Clustering," *Jurnal sosial humaniora,* vol. Special Edition 2020, no. 1, pp. 59-79, 2020, doi: 10.12962/j24433527.v0i1.6776.

[107] A. J. Christy, A. Umamakeswari, L. Priyatharsini, and A. Neyaa, "RFM ranking – An effective approach to customer segmentation," *Journal of King Saud University. Computer and information sciences,* 2018, doi: 10.1016/j.jksuci.2018.09.004.

[108] R. Vohra, J. Pahareeya, A. Hussain, F. Ghali, and A. Lui, "Using Self Organizing Maps and K Means Clustering Based on RFM Model for Customer Segmentation in the Online Retail Business," in *International Conference on Intelligent Computing*, 2020: Springer, pp. 484-497.

[109] M. Berry, G. Linoff, and B. Lucas, "Data Mining Techniques: Theory and Practice," Data Miners Inc., 2009.

[110] M. Fitri, A. Sarifah Shakinah Syed, Y. Zeratul Izzah Mohd, H. Fachrudin, and A. Tubagus Mohammad Akhriza, "Segmentation Model of Customer Lifetime Value in Small and Medium Enterprise (SMEs) using K-Means Clustering and LRFM Model," *International Journal of Integrated Engineering,* vol. 11, no. 3, 09/03 2019. [Online]. Available: https://publisher.uthm.edu.my/ojs/index.php/ijie/article/view/4661.

[111] J. Zhou, L. Zhai, and A. A. Pantelous, "Market segmentation using high-dimensional sparse consumers data," *Expert systems with applications,* vol. 145, p. 113136, 2020, doi: 10.1016/j.eswa.2019.113136.

[112] P. Anitha and M. M. Patil, "RFM model for customer purchase behavior using K-Means algorithm," *Journal of King Saud University. Computer and information sciences,* 2019, doi: 10.1016/j.jksuci.2019.12.011.

[113] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory,* vol. 28, no. 2, pp. 129-137, 1982, doi: 10.1109/TIT.1982.1056489.

[114] Y.-Z. Chen and Y.-C. Lai, "Sparse dynamical Boltzmann machine for reconstructing complex networks with binary dynamics," *Physical Review E,* vol. 97, 03/28 2018, doi: 10.1103/PhysRevE.97.032317.

[115] B. Benjamin *et al.* "Yellowbrick." scikit-yb.org. http://www.scikit-yb.org/en/latest/ (accessed July 20 2022).

[116] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, Inc., 2019.

[117] T. T. Win and K. S. Bo, "Predicting Customer Class using Customer Lifetime Value with Random Forest Algorithm," in *2020 International Conference on Advanced Information Technologies (ICAIT)*, 4-5 Nov. 2020 2020, pp. 236-241, doi: 10.1109/ICAIT51105.2020.9261792. [Online]. Available: https://ieeexplore.ieee.org/document/9261792/

[118] B. Karaman. "Customer Lifetime Value Prediction." towardsdatascience.com. https://towardsdatascience.com/data-driven-growth-with-python-part-3-customer-lifetime-value-prediction-6017802f2e0f (accessed May 02, 2022).

[119] A. Labram. "Fitting data with XGBoost." actuaries.org.uk. https://www.actuaries.org.uk/news-and-insights/news/article-fitting-data-xgboost (accessed May 15, 2022).

[120] V. Morde. "XGBoost Algorithm: Long May She Reign!" towardsdatascience.com. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d (accessed May 15, 2022).

[121] "Awesome XGBoost." github.com. https://github.com/dmlc/xgboost/tree/master/demo#readme (accessed May 15, 2022).

[122] A. Floares, M. Ferisgan, D. Onita, A. Ciuparu, G. Calin, and F. Manolache, "The Smallest Sample Size for the Desired Diagnosis Accuracy," *International Journal of Oncology and Cancer Therapy,* 08/23 2017.

[123] J. Grus, *Data science from scratch: first principles with python*. O'Reilly Media, 2019.

[124] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009.

[125] K. Nordhausen, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition by Trevor Hastie, Robert Tibshirani, Jerome Friedman," vol. 77, ed. Oxford, UK: Blackwell Publishing Ltd, 2009, pp. 482-482.

[126] "The Comprehensive Guide to Model Validation Framework: What is a Robust Machine Learning Model?" odsc.medium.com. https://odsc.medium.com/the-comprehensive-guide-to-model-validation-framework-what-is-a-robust-machine-learning-model-7bdbc41c1702 (accessed May 15, 2022).

[127] G. J. M. Rosa, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction by HASTIE, T., TIBSHIRANI, R., and FRIEDMAN, J," *Biometrics,* vol. 66, no. 4, pp. 1315-1315, 2010, doi: 10.1111/j.1541-0420.2010.01516.x.

[128] J. Brownlee. "Avoid Overfitting By Early Stopping With XGBoost In Python." machinelearningmastery.com. https://machinelearningmastery.com/avoid-overfitting-by-early-stopping-with-xgboost-in-python/ (accessed May 22, 2022).

[129] J. Brownlee. "Train-Test Split for Evaluating Machine Learning Algorithms." machinelearningmastery.com. https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/ (accessed May 2022, 2022).

[130] R. Bakeman, V. Quera, D. McArthur, and B. F. Robinson, "Detecting Sequential Patterns and Determining Their Reliability With Fallible Observers," *Psychological methods,* vol. 2, no. 4, pp. 357-370, 1997, doi: 10.1037/1082-989X.2.4.357.

[131] J. R. Landis and G. G. Koch, "The Measurement of Observer Agreement for Categorical Data," *Biometrics,* vol. 33, no. 1, pp. 159-174, 1977, doi: 10.2307/2529310.

[132] M. J. Anzanello and F. S. Fogliatto, "Learning curve models and applications: Literature review and research directions," *International Journal of Industrial Ergonomics,* vol. 41, no. 5, pp. 573-583, 2011/09/01/ 2011, doi: https://doi.org/10.1016/j.ergon.2011.05.001.

[133] T. Bex. "Comprehensive Guide to Multiclass Classification Metrics." towardsdatascience.com. https://towardsdatascience.com/comprehensive-guide-on-multiclass-classification-metrics-af94cfb83fbd (accessed May 15, 2022).

[134] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters,* vol. 27, no. 8, pp. 861-874, 2006/06/01/ 2006, doi: https://doi.org/10.1016/j.patrec.2005.10.010.

[135] "CRISP-DM Help Overview." ibm.com. https://www.ibm.com/docs/en/spss-modeler/18.2.0?topic=dm-crisp-help-overview (accessed May 15, 2022).

[136] *The Complete Journey*, dunnhumby. [Online]. Available: https://www.dunnhumby.com/source-files

[137] "The Complete Journey: User Guide." dunnhumby. https://www.dunnhumby.com/source-files/ (accessed May 15, 2021).

[138] İ. Kabasakal, "Customer segmentation based on recency frequency monetary model: A case study in E-retailing," *Bilişim Teknolojileri Dergisi,* vol. 13, no. 1, pp. 47-56, 2020.
[139] T. Elliott. "The State of the Octoverse: machine learning." github.blog. https://github.blog/2019-01-24-the-state-of-the-octoverse-machine-learning/ (accessed November 7, 2021).
[140] K. Johnson. "GitHub: Python and TypeScript gain popularity among programming languages." https://venturebeat.com/2020/12/02/github-python-and-typescript-gain-popularity-among-programming-languages/ (accessed November 7, 2021).
[141] "History and License." python.org. https://docs.python.org/3/license.html (accessed April 23, 2022).
[142] "Quotes about Python." python.org. https://www.python.org/about/quotes/ (accessed April 23, 2022).
[143] C. Harris *et al.*, "Array programming with NumPy," *Nature,* vol. 585, no. 7825, p. 6, 2020, doi: 10.1038/s41586-020-2649-2.
[144] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in science & engineering,* vol. 9, no. 03, pp. 90-95, 2007.
[145] *Collaborative data science*. (2015). Plotly Technologies Inc, Montréal, QC. [Online]. Available: https://plot.ly
[146] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods,* vol. 17, p. 12, 2020, doi: 10.1038/s41592-019-0686-2.
[147] W. McKinney, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, 2010, vol. 445, no. 1: Austin, TX, pp. 51-56.
[148] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *The Journal of Machine Learning Research,* vol. 18, no. 1, pp. 559-563, 2017.
[149] D. Gutierrez. "Why You Should be Using Jupyter Notebooks." opendatascience.com. https://opendatascience.com/why-you-should-be-using-jupyter-notebooks/ (accessed March 22, 2022).
[150] "Jupyter." jupyter.org. https://jupyter.org/ (accessed April 22, 2022).
[151] "pip 22.0.4." pypi.org. https://pypi.org/project/pip/ (accessed March 14, 2022).
[152] D. Virmani, S. Taneja, and G. Malhotra, "Normalization based K means Clustering Algorithm," 2015, doi: https://doi.org/10.48550/arXiv.1503.00900.
[153] "XGBoost Parameters." xgboost.readthedocs.io. https://xgboost.readthedocs.io/en/latest/parameter.html#learning-task-parameters (accessed May 1, 2022).

7. Appendix

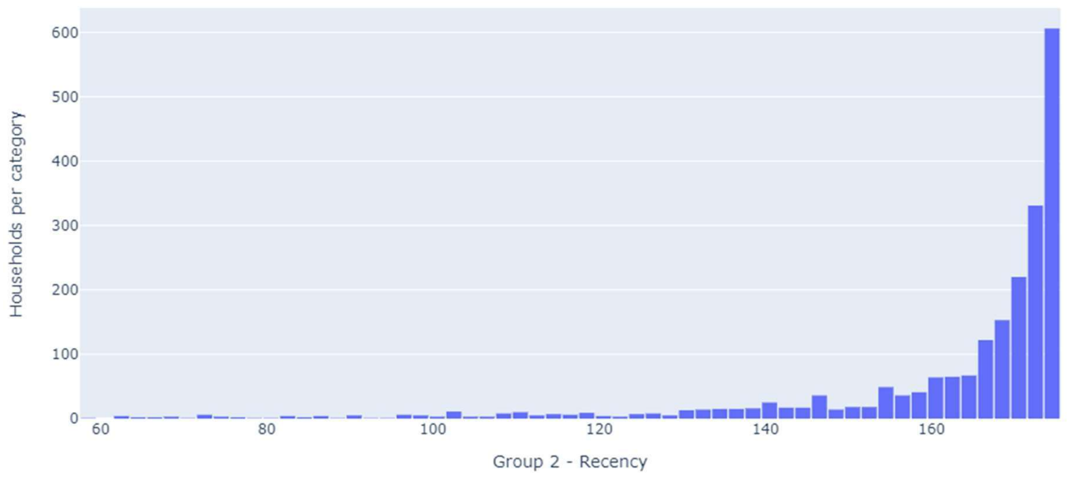


Figure 34. Number of households by Recency

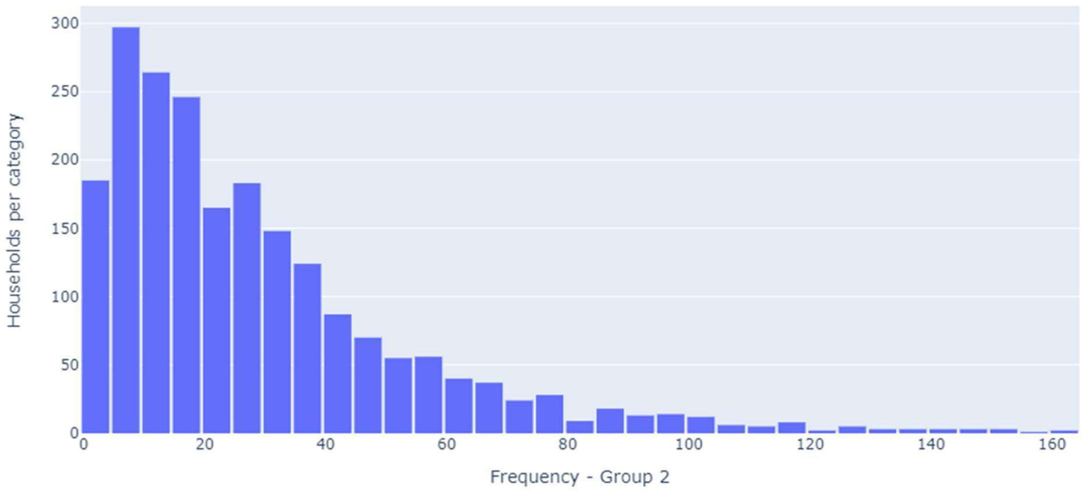


Figure 35. Number of households by Frequency

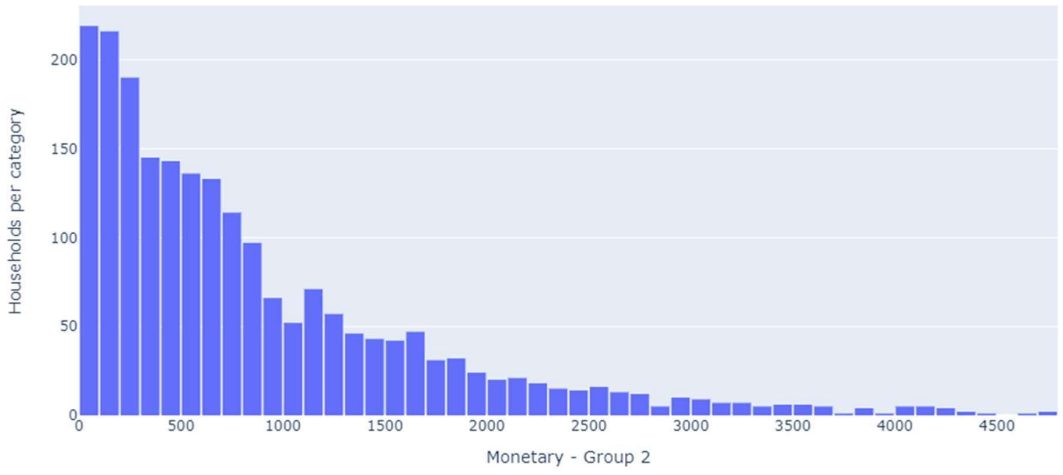


Figure 36. Number of households by Monetary value

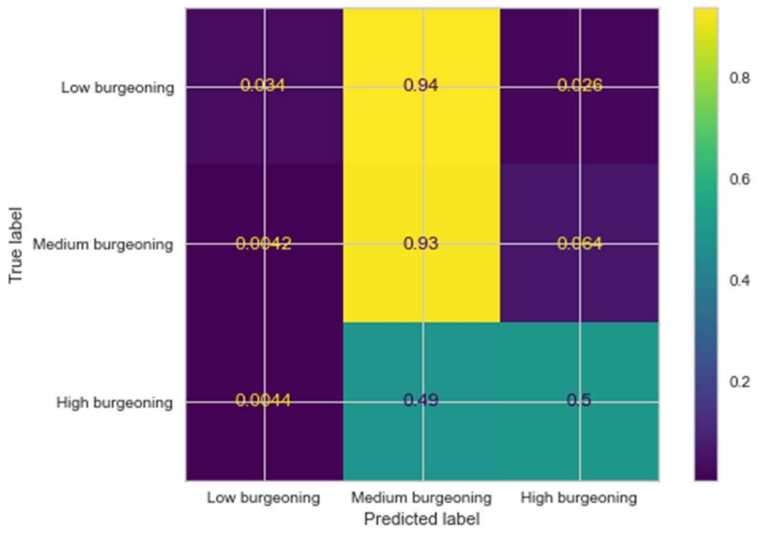


Figure 37. Confusion matrix for the Group 4 Burgeoning Cluster (C)