

BACHELOR

Customer churn prediction using machine learning in the domain of business-to-business

Nejatianrad, Mehrsa

Award date:
2023

Awarding institution:
Tilburg University

[Link to publication](#)

Disclaimer

This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

Customer churn prediction using machine learning in the domain of business-to-business

Mehrsa Nejatianrad

STUDENT NUMBER: 2059032 (TiU), 1650009 (TU/E)

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
JOINT BACHELOR OF SCIENCE IN DATA SCIENCE

Thesis committee:

Supervisor: Dr. X. Du

External Supervisor: Maria Baltoglou

Tilburg University
Law School
Institute for Law, Technology and Society
Tilburg, The Netherlands

Eindhoven University of Technology
Department of Mathematics and Computer Science
Eindhoven, The Netherlands

June 2023

Customer churn prediction using machine learning in the domain of business-to-business

Mehrsa Nejatianrad

In this study, a machine learning-based churn model is proposed for a subscription-based service provider in the context of business-to-business. In the proposed machine learning model, the following algorithms are compared; Random Forest, Gradient Boosting, and Logistic Regression. The data used in this study include customer usage data and customer information. The data is sampled in different ways; single-year cross-sectional data, an observation period of 12 months and a churn window of 6 months, and multi-year cross-sectional data, where the data from each year is treated independently. Since the data is imbalanced, the following techniques, SMOTE, RandomUnderSample, and RandomOverSample, are used to balance the data and to determine the most effective method. The result of the models is evaluated based on accuracy, precision, recall, F1-score, and ROC curve. In addition, Stratified K-Fold Cross Validation and Stratified-GroupKFold Cross Validation are used for single-year cross-sectional data and multi-year cross-sectional data, respectively, to detect overfitting. The result shows that the Gradient Boosting classifier, an ensemble learner, performs the best, and implementing balancing techniques can improve the performance.

1 Introduction

Bynder is a software-as-a-service (SaaS) company that provides a cloud-based digital asset management (DAM) platform. It allows marketing teams to create, find, and use digital assets, like images, videos, and documents. The main goal and purpose of Bynder's DAM platform are to tackle the challenges organizations encounter when managing their digital assets, such as inconsistent branding, lack of collaboration, disorganized files and folders, and security concerns. Businesses can use the DAM platform to quickly organize and find files, control access, and usage rights, share files internally and externally, and download files in the desired format. In addition, Bynder offers the following products; Brand Guidelines, Creative Workflow, Content Workflow, Studio, Dynamic Asset Transformation, Print Brand Templates, Bynder Analytics, and Integrations.

1.1 Importance of customer retention for SaaS companies

In the present digital world, businesses face increasing competition from all sides. As new companies enter the market regularly, standing out and remaining relevant has become more challenging than ever. This is especially true for businesses involved in the software-as-a-service (SaaS) sector, which has grown significantly over the past several years ([noa a](#)).

Bynder, a top digital asset management solutions supplier, follows this pattern. Bynder has had to put in a lot of effort to keep up its position as a market leader in the increasingly competitive market for digital asset management. Bynder's emphasis on client retention is one crucial aspect that has helped the company succeed.

For various reasons, customer retention has grown in importance to Bynder's performance. Firstly, keeping current clients is typically less expensive than finding new ones. According to the Harvard Business Review, acquiring a new customer might cost up to five times more than retaining an existing one (Gallo 2014). Bynder has acknowledged this and made an effort to maintain the satisfaction and engagement of its current clients.

Second, maintaining customers is crucial to developing brand loyalty and reputation (Kaiser and Würthner 2020). Customers are more likely to recommend a good or service to others when satisfied. This can result in a boost in word-of-mouth advertising and brand recognition (Ngoma and Ntale 2019). Bynder has established a positive reputation for providing high-quality digital asset management solutions. And is continuously making an effort to uphold this reputation by offering excellent support and customer care.

Lastly, a significant factor in revenue increase can be client retention. Companies can benefit from recurring revenue streams by keeping their current clients and making more profit through upselling and cross-selling. Bynder has invested in creating new features and products that can add value to its existing customer base after discovering the possibilities of this strategy.

Another strategy that they see has the potential to increase their retention rate and reduce the churn rate is by reporting on churn predictively. This allows Customer Success Managers (CSM) to act based on the predictions and prevent customers from churning. Thus, they started creating a churn prediction model in 2020. Their model is based on DAM (Digital Asset Management) login usage data, and it compares the logins of the customers over time (2 months over two months) and the Tier (ARR segment) they are in. In addition, the model distinguishes the phase of a customer ("new" up to "auto-pilot"). Currently, Bynder has seen that the model can predict customers that are at risk. However, it has an 84% accuracy, see (Eq. 1), 0.47% recall, see (Eq. 1), and 0.14% precision, see (Eq. 1).

1.2 Problem Statement

Although their model has high accuracy, it overfits customers as the data is imbalanced and new customers are difficult to predict as its usage data may vary significantly. As indicated by the low recall and precision scores, the model performs poorly in predicting churned customers. Therefore, the primary purpose of this study is to tackle these limitations by gaining more insight into their customers' behavior by implementing additional product usage data and determining the influential variables. And assess which supervised machine learning algorithm is most suitable for Bynder to achieve their goals.

1.3 Research Question

In order to fulfill the objectives of this study, the following research question has been formulated:

How can supervised machine learning methods and customer behavior analysis be used to predict customer churn in the domain of business-to-business?

2 Related Work

Subscription-based services have grown fast in recent years and rely entirely on customer revenue (Vanninen et al. 2022). As a result, customer churning significantly influences these organizations. There have been many studies conducted to predict customer churn accurately, particularly in the domain of business-to-customer (B2C). However, churn prediction in the context of business-to-business (B2B) has not been much investigated. Correctly predicting churned customers becomes even more critical for B2B companies, as each lost customer has a more significant negative impact on the company directly, specifically on the annual recurring revenue (ARR) (Figalist et al. 2019).

2.1 Binary churn prediction in the domain of B2C

Vafeiadis et al. examine the performance of different machine learning classifiers for customer churn prediction, (Vafeiadis et al. 2015). In this study, the data is provided by a telecom provider, consisting of 5000 samples and multiple features. The features include; call duration and number of texts sent. In addition, specific subscription information is also used, such as subscription period. The information about the subscription is included as well. The following classifiers were compared Artificial Neural Network (ANN), Support Vector Machine (SVN), Decision Tree (DT), Naïve Bayes, and Logistic Regression. They used boosting for Artificial Neural Network (ANN), Support Vector Machine (SVN), and Decision Tree (DT) to compare them to their boosted versions. To evaluate the performance of the different classifiers, they use precision, recall, accuracy, and F-measure, which are calculated from the confusion matrix. They concluded that boosting improves performance in terms of accuracy and F-measure. All the classifiers' accuracy ranges between 93-99 percent.

A study by Lalwani et al. regarding churn prediction using a machine learning approach investigated the following algorithms; Logistic Regression, Naïve Bayes, Support Vector Machine, Decision Tree, Random Forest, and various boosting algorithms such as XGBoost and AdaBoost (Lalwani et al. 2022a). The data used in this study was also from a telecom provider, which consisted of 7000 instances with 21 features. The data consisted of information about usage information and customer demographics. In their study, they follow the following framework; data preprocessing, feature analysis, feature selection, developing the models based on the mentioned algorithm, and evaluating the model. The evaluation metric is similar to the one used by Vafeiadis et al. They conclude that the boosted classifiers, XGBoost and AdaBoost, outperformed the other models and obtained the highest accuracy.

2.2 Binary churn prediction in the domain of B2B

In a study regarding churn prediction, Gordini & Veglio, (Gordini and Veglio 2017), customized the churn prediction model for the B2B e-commerce sector. They employ various techniques, including Support Vector Machines (SVM), Neural Networks, and Logistic Regression, to create their churn prediction model. The dataset they use for their research comes from a significant Italian-based online retailer and includes consumer transactional and demographic data. Their findings demonstrate that SVM out-

performs the other algorithms they took into consideration. According to the results, Gordini & Veglio, (Gordini and Veglio 2017) conclude that churn prediction is vital in B2B and that the choice of algorithm is critical because the outcomes of the churn prediction models vary.

In order to close the gap caused by customer churning in the B2B domain, Figalist et al., (Figalist et al. 2019) build a churn prediction model within a B2B domain. This study offers a method for mapping the data that integrates the stakeholders who have an impact on decisions both directly and indirectly. Several stakeholders were interviewed by Figalist et al., (Figalist et al. 2019), and data collected from a B2B organization was analyzed to gain a greater understanding of churn prediction. Data about consumers or end users are used to forecast churn. According to the findings of Figalist et al., churn prediction in the B2B sector is frequently more critical than in the B2C sector since these companies lose more business when a client churns than B2C enterprises.

3 Theoretical Concepts

In this section, the theoretical topics are explained to widen the reader's previous knowledge. Understanding the topics offered in this section is required to comprehend the study entirely.

3.1 Customer Relation Management (CRM)

Customer Relationship Management (CRM) can be seen as a strategy that aims to improve a company's profitability and customer relations. CRM allows businesses to understand customers' needs, wants, and behaviors (Khashab et al. 2022). Moreover, CRM uses this information to improve customer satisfaction and loyalty, reducing churn and business growth (Thompson 2004). CRM can be implemented in various ways depending on the company (Khashab et al. 2022). However, the concept stays the same. CRM systems are the technical part of CRM. They allow communication between the customer and the company. In addition, the customer data is stored and analyzed to gain insights regarding the company's customer base. Furthermore, customer data can then be used in CRM applications such as churn prediction or identifying upsell opportunities.

3.2 Machine Learning

Machine learning regards extracting knowledge from data and is a subset of AI and computer science. Machine learning is used when there is a need to understand the data and gain insights from it. Machine learning can be divided into supervised learning, unsupervised learning, and reinforcement learning. Depending on the scope of the problem and the data, whether there is a label class, one of these machine learning types can be used (Simeone 2018).

3.2.1 Data Preprocessing

Preprocessing data is a crucial step in the model's performance, including cleaning, integration, transformation, and reduction (Alasadi and Bhaya 2017).

3.2.2 Standard Scalar

Scaling is an essential step in data processing, as it can heavily influence the model's performance. The standard scalar is a scaling technique that works by ensuring each

feature has a mean of 0, and the variance is 1 (Muller and Guido 2016). It should be noted that this technique does not guarantee that the minimum and maximum values of the features are within a specific range (Muller and Guido 2016). Mathematically it can be represented as:

$$z = \frac{x - \mu}{\sigma} \quad (1)$$

where:

- z is the standardized feature
- x is the original feature
- μ is the mean of all x 's
- σ is the standard deviation of all the x 's

3.2.3 Quantile Transformation

The distribution of a given dataset can be changed using the quantile transformation approach to transform to a desired target distribution. It can transform the original data to uniform or normal distribution (Wijaya 2021). More specifically, the transformation uses the input data's cumulative distribution function (CDF) and a predetermined target distribution. The quantile transformer applies the quantile function to the dataset to transform it (Wijaya 2021). The quantile function is the inversed function of CDF (Wijaya 2021).

3.2.4 Supervised Machine Learning

Supervised learning is a type of machine learning which is mainly used for prediction problems (Jiang, Gradus, and Rosellini 2020). Models are trained using labeled data in supervised learning. Each sample in the training dataset has both the input features and the correct output or label corresponding to the input features (Muller and Guido 2016). Inputs and outputs are mapped by the model during training. The objective is to use this learned mapping to predict the label for unseen data precisely. Supervised learning can be divided into regression and classification. Regression is used when the target variable is continuous, for instance, height. On the other hand, classification is used when the target variable is categorical such as churned or not churned.

3.2.5 Logistic Regression

Logistic Regression is a statistical model used for binary classification problems. It is a form of regression analysis. However, the dependent variable here is categorical, unlike regression takes a continuous variable as the target. Logistic regression determines the probability for a sample to belong to a class. The probability falls between 0 and 1 and is continuous. It relies on a threshold function to make a decision, known as a Sigmoid or Logistic function. The following steps illustrate the mathematics behind logistic regression (Addagatla 2021).

1. Linear Regression: Firstly, a linear regression model is applied to the input features.

$$y = \beta_0 + \beta_1 * x_1$$

where:

- y is the output variable
- x_1, x_2, \dots, x_n are independent variables
- β_0 and β_1 are the coefficients of the model
- ϵ is the error term

2. Sigmoid function: this function takes any input value and transforms it into a number between 0 and 1.

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where:

- z is the output of the linear regression

3. Applying Sigmoid function to the output of Linear Regression:

$$P(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 * x_1)}}$$

where

- P is the probability of the positive class

4. Maximum Likelihood Estimation: The likelihood function is maximized to estimate the model's parameters. The likelihood function for a binary classification is:

$$L(\beta) = \prod_{i=1}^n p(x_i)^{y_i} * (1 - p(x_i))^{1-y_i}$$

where:

- x_i is the i th observation
- y_i is the actual class label for the i th observation
- $p(x_i)$ is the predicted probability of $y_i = 1$

The maximum likelihood assumes the independence of observations.

5. Gradient Descent: This is an optimization algorithm that is used to minimize a function iteratively. It determines function parameters that minimize a cost function as much as possible. The model parameters β_j (which can be set to zero or a small random number) are initialized. This process is repeated till the cost function has converged.

$$\beta_j = \beta_j - \alpha \frac{\partial J(\beta)}{\partial \beta_j}$$

where

- β_j is the j th model parameter
- α is the learning rate
- $J(\beta)$ is the cost function defined as:

$$J(\beta) = - \sum_{i=1}^n [y_i \log(p(x_i)) + (1 - y_i) \log(1 - p(x_i))]$$

- $\frac{\partial J(\beta)}{\partial \beta_j}$ is the partial derivative of the cost function with respect to β_j

3.2.6 Ensemble Learning

Ensemble learning is a machine learning approach in which numerous models (commonly called "base learners") are trained and integrated to achieve better results (Sagi and Rokach 2018). The underlying idea of ensemble learning is that a collection of "weak learners" can combine forces to become a "strong learner" (Sagi and Rokach 2018).

Bagging (Bootstrap Aggregating) is a technique used in ensemble learning. It works by dividing the original data into several subsets (with replacement), training a distinct model on each subset, then combining the predictions, usually by voting (classification) or averaging (regression) (Breiman 1996).

Another ensemble technique, boosting, trains models consecutively, with each new model trained to rectify the mistakes caused by the preceding ones (van Rijn et al. 2018). Models are weighted based on their accuracy, and the final prediction is usually a weighted composite of the predictions of the individual models (van Rijn et al. 2018).

Bagging can decrease variance and solves over-fitting issues in a model. While Boosting decreases bias (Oza 2004). In Bagging, equal weight is received by each model. On the other hand, in boosting, models are weighted based on their performance (Oza 2004).

3.2.7 Random Forest Classifier

Random Forest is an ensemble learner which can be used for both regression and classification (Biau and Scornet 2016). A random forest is a meta estimator that fits many decision tree classifiers on different sub-samples of the dataset and utilizes averaging to increase predicted accuracy and control over-fitting (Biau and Scornet 2016). The following steps show how the random forest classifier works (Biau and Scornet 2016).

1. Bootstrapping the data: Bootstrapping is a statistical method for estimating an estimator's sampling distribution by sampling with replacement from the original data sample. In the context of Random Forest, this entails producing several subsets (B) of the same dataset, each of the same size as the original (Egbert and Plonsky 2021). Each subset is generated by randomly drawing instances with replacements, which implies that some examples may appear numerous times in the subset, and others may not exist. This produces somewhat different models since they are based on slightly different datasets.
2. Building decision trees Each bootstrap sample from the previous phase is utilized to construct a new decision tree. Each tree aims to develop a set of rules that can accurately categorize or predict the target variable using data attributes (Biau and Scornet 2016).
 - (a) Feature selection: At each node of the tree, a decision must be taken regarding which feature to utilize to separate the data. This conclusion is based on the Gini Impurity or Entropy of each feature (Biau and Scornet 2016). These metrics assess the "purity" of the labels in each conceivable split. A pure node (all instances have the same label) has no impurity or entropy. The characteristic that results in the most significant decrease in impurity or entropy is

chosen for the split (Biau and Scornet 2016).

$$Gini(n) = 1 - \sum_{i=1}^C (p_{i|n})^2 \text{ or } Entropy(n) = - \sum_{i=1}^C p_{i|n} \log_2(p_{i|n})$$

where:

- C is the total number of classes in the dataset
 - $p_{i|n}$ is the proportion of samples at node n that belong to class i
- (b) Splitting is done based on choosing the splitting point that maximizes the Information Gain:

$$IG(A) = I(parent) - \sum_j \frac{N_j}{N} I(child_j)$$

where:

- $I(parent)$ is the impurity or entropy of the parent node
 - $I(child_j)$ is the impurity or entropy of the j th child node
 - N_j is the number of instances in the j th node
 - N is the total number of instances in the parent node
- (c) Stopping condition: the tree stops once the previously defined stopping condition is met
- (d) Making Predictions
3. Making Predictions for each point: The random forest can be used to make predictions after all B decision trees have been created. Each decision tree in the forest makes an independent prediction, and the random forest's final prediction is typically the most common class estimated by all the trees (Biau and Scornet 2016). The technique of voting on each model in an ensemble and using the majority vote as the final prediction is known as "bagging" or "bootstrap aggregating" (Biau and Scornet 2016).

$$Class_{prediction} = mode(Class_{prediction,tree1}, Class_{prediction,tree2}, ..., Class_{prediction,treeB})$$

where:

- B is the total number of trees
 - $Class_{prediction,treeB}$ is the predicted class for the b th tree
4. Estimating feature importance

$$FI_m = \frac{1}{B} \sum_{b=1}^B \sum_{t \in T_b} I(v(t) = m) \Delta i(s_t, t)$$

where:

- FI_m is the feature importance of feature m
- B is the total number of trees in the random forest
- T_b is the set of all nodes in tree b
- $v(t)$ is the feature used in node t

- $I(v(t) = m)$ is an indicator function that equals 1 if $v(t)$ equals m , and 0 otherwise
- s_t is the split point at node t
- $\Delta i(s_t, t)$ is the impurity decrease at node t

3.2.8 Gradient Boosting Classifier

Gradient Boosting is an ensemble learner which can be used for both regression and classification. The main concept underlying gradient boosting is integrating multiple simple models (weak learners) like shallow trees (Ayyadevara and Ayyadevara 2018). Each tree can only make good predictions on the part of the data, and so more and more trees are added to increase performance progressively. The following steps illustrate how gradient boosting works, (Natekin and Knoll 2013):

1. Initialize the model: Start with a baseline model that always predicts the majority. The prediction is denoted as $F_0(x)$
2. Iteratively add weak learners: $\forall m \in \{1, 2, \dots, M\}$, where M is the number of iterations
 - (a) Compute pseudo-residuals of the loss function with respect to the prediction of the model at the previous stage:

$$r_{im} = -\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}$$

where

- r_{im} are the residuals for the i^{th} instance at stage m
 - y_i is the actual target value for the i^{th} instance
 - $F_{m-1}(x_i)$ is the prediction of the model at the previous stage for the i^{th} instance
- (b) Fit weaker learners to the residual from the previous step. The result is a new model denoted by $h_m(x)$
 - (c) Compute a multiplier γ_m for the weak learner by solving the following optimization problem:

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^n L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$$

where

- n is the number of instances
 - the goal is to find the value of γ that minimizes the loss when the weak learner's predictions are added to the previous model's predictions
- (d) Update the model by adding the weak learner's predictions, scaled by the multiplier, to the predictions of the model at the previous stage:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

3. After all, iterations are complete, make predictions using the final model: $F_M(x)$

3.2.9 Imbalanced Learning

Imbalanced learning is a machine learning problem in which the classes are unequal. This refers to classification problems where one class has more instances than the rest, for example, fraud detection, churn prediction, and disease diagnosis. This imbalance can negatively influence the model's performance since most machine learning algorithms are biased toward the majority class. This means that they will not be able to predict the minority class and will most likely overfit the majority class. Several strategies aim to address this problem. These strategies work by removing instances from the majority class or adding instances to the minority class. These two methods can be combined as well.

3.2.9.1 Random Over-sampling

Random over-sampling is a simple way to address the class imbalance. It entails copying examples from the minority class at random. Although this strategy can help enhance the classifier's performance on the minority class, it has a significant drawback: it can induce overfitting because it creates perfect duplicates of the current samples (Le et al. 2019).

3.2.9.2 Random Under-sampling

Random under-sampling, on the other hand, attempts to balance the class distribution by removing majority class examples at random. When the amount of data is sufficient, this strategy can be effective. A new balanced dataset can be produced for further modeling by maintaining all samples in the minority class and randomly selecting an equal number of samples in the majority class (Weiss, McCarthy, and Zabar 2007). The disadvantage of this method is that the data from the majority class is dropped, meaning valuable information will be lost (Weiss, McCarthy, and Zabar 2007).

3.2.9.3 SMOTE (Synthetic Minority Over-sampling Technique)

SMOTE is an oversampling method that creates synthetic samples from the minority class instead of creating copies (Weiss, McCarthy, and Zabar 2007). This approach benefits from not losing valuable information that could be lost with under-sampling while avoiding overfitting, which can occur with random oversampling. This algorithm makes synthetic samples of the minority class based on the nearest neighbor.

3.2.10 Feature Selection

Feature selection is a critical factor for achieving a predictive model with high performance. Feature selection reduces the dimensionality of the input data, which helps the model avoid overfitting and have better performance (Ma and Huang 2008).

3.2.10.1 Recursive Feature Elimination with Cross-Validation

Recursive Feature Elimination with Cross-Validation (RFECV) is a feature selection method for predictive modeling that attempts to discover and rank the most informative features in a dataset (Mustaqim et al. 2021). Rank 1 indicates the feature is essential, and numbers greater than 1 are insignificant in predicting the target variable (Mustaqim et al. 2021). RFECV approach is made up of two major components: Recursive Feature Elimination (RFE) and Cross-Validation (CV)(Explained in Section 3.3.3).

RFE is an iterative feature elimination procedure (Wang, Xiao, and Wu 2019). It starts by training and assessing a model on all of the features in the dataset. The model's feature weights are sometimes used to determine which features are deleted. The model is retrained on the remaining features, and the process is repeated until a predefined stopping criterion, such as the number of features to pick, is fulfilled (Wang, Xiao, and Wu 2019).

RFECV combines RFE and CV to create a reliable method for selecting the appropriate features while avoiding overfitting (Wang, Xiao, and Wu 2019). Cross-validation tests the model's performance after each round of feature reduction in RFE (Wang, Xiao, and Wu 2019). The final feature set is selected as the one with the highest cross-validation score.

3.2.10.2 Principal Component Analysis

Principal Component Analysis (PCA) is a technique used for dimensional reduction. It should be noted that PCA should be only applied to scaled data since PCA is affected by the scale of the features. PCA first computes the covariance matrix of the data. The matrix's eigenvectors and eigenvalues are next computed, with eigenvectors determining the directions of the new feature space and eigenvalues determining their magnitude (Shlens 2014). The projection matrix is then formed, and the data is transformed by projecting the original dataset onto the new space using that matrix. The linearity of the data structure is a crucial assumption for PCA (Shlens 2014). The mathematics behind PCA is presented in the following steps;

1. Take the dataset without the class labels:
 - Compute the mean and covariance
 - Center the data by subtracting the mean of each feature from all the data points for that feature.
2. Calculate the covariance matrix for the dataset

$$Cov(X, \hat{Y}) = \frac{1}{N} \sum_{i=1}^n (x_i - \hat{x})(y_i - \hat{Y}) \quad (2)$$

Where;

- N is the number of data points
 - x_i and y_i are individual data
 - \hat{X} and \hat{Y} are the means
3. Calculate eigenvalues and eigen vectors:

$$|A - \lambda I| = 0 \quad (3)$$

$$(A - \lambda I)\mathbf{v} = 0 \quad (4)$$

Where;

- A is the matrix
- λ are the eigenvalues
- I is the identity matrix

- v is an eigenvector
4. Sort the eigenvalues in decreasing order and their corresponding eigenvectors
 5. Pick k eigenvalues and form a matrix of eigenvectors
 6. Transform the original dataset:

$$\text{Transformed Data} = \text{Feature matrix} * \text{top } k \text{ eigenvectors} \quad (5)$$

3.3 Model Evaluation

Machine learning models must be evaluated to see how well the model works. The most frequent metrics for evaluating binary classifiers are accuracy, recall, precision, and F1-score.

3.3.1 Confusion Matrix

A confusion matrix is a machine learning concept used to describe the classifier's performance. It includes information about the actual and predicted classifications ([Lalwani et al. 2022b](#)).

- True Positive (TP): True Positives are values correctly predicted as positive, in other words, cases where the actual class was positive, and the model correctly predicted the positive class. For example, in a churn classification, this would mean the customers who had churn, and we predicted by the model as churned.
- False Positive (FP): False Positives correspond to cases where the actual class was negative; however, the model predicted them as a positive class. For example, for churn classification, the test customers who had churned were classified as not churned.
- True Negative (TN): True Negatives are the values correctly predicted as negative. Meaning the cases where the actual class was negative, and the model correctly predicted it as negative. In churn, this would mean the model predicted these values as not churned, and they did not churn.
- False Negative (FN): False Negatives are when the actual class is positive while the model predicted it as a negative class.

Table 1: Confusion matrix

		Predicted	
		Positive (1)	Negative (0)
Actual	Negative (0)	TP	FN
	Positive (1)	FP	TN

Accuracy

Accuracy is a measure that displays the percentage of total correctly classified cases

(Deng et al. 2016). Accuracy is defined as, (Deng et al. 2016): Copy code

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

Precision

Precision is a metric that represents the percentage of accurately predicted positive events (Deng et al. 2016). The measure indicates how often the model correctly predicts the target class, in this case, churners. Precision is the accuracy of predicting a certain class and is calculated as follows, (Deng et al. 2016):

$$\text{Precision} = \frac{TP}{TP + FP} \quad (7)$$

Recall

The recall metric demonstrates the classifier's ability to identify positive examples (Deng et al. 2016). It demonstrates the binary classifier's capacity to recognize occurrences of a certain class (Deng et al. 2016). The following formula is used to calculate recall, (Deng et al. 2016):

$$\text{Recall} = \frac{TP}{TP + FN} \quad (8)$$

F-score

The F-score is used to assess the performance of a classifier. It considers both precision and recall and is defined as the harmonic mean of precision and recall (Deng et al. 2016). A higher F1-score means both recall and precision are high, with the max being 1 (Deng et al. 2016).

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (9)$$

3.3.2 ROC Curve and AUC

The AUC-ROC curve is a visual way to assess a classifier's performance at various threshold levels (Fawcett 2006). The ROC curve is formed by charting the recall versus the false positive rate (FPR), which is defined as, (Fawcett 2006):

$$\text{FPR} = \frac{FP}{FP + TN} \quad (10)$$

The AUC indicates how well the model can distinguish between classes. The higher the AUC, the better the model predicts instances belonging to the negative class as negative and positive instances as positive.

3.3.3 Cross Validation

K-fold cross-validation (KCV) is a popular machine learning technique for model selection and error estimation (Anguita et al. 2012). It works by dividing datasets into 'k' subsets. After that begins an iterative process of selecting some of the subsets for training and the other ones for evaluating the model performance, this process is

repeated 'k' times (Anguita et al. 2012). In the training set, each point appears 'k-1' times, while in the test set, each point appears precisely once (Anguita et al. 2012).

Stratified K-Fold Cross Validation is an enhanced version of KCV. This technique helps work classification problems with datasets where the class distribution is unbalanced (Prusty, Patnaik, and Dash 2022). It deals with the problem of unbalances by making sure that each of the folds contains the same proportion of observations with each label (Prusty, Patnaik, and Dash 2022).

Group K-Fold Cross Validation is a technique that is useful while working with data that contains groups that should not be split across training and test sets (Abhigyan 2021). It ensures that the same group (e.g., the same customer) is not represented in training and test sets. Otherwise, the model could learn customer-specific features, failing to generalize for new observations, resulting in overfitting.

StratifiedGroupKFold Cross Validation combines Group K-Fold Cross Validation and Stratified K-Fold Cross Validation. It tries to combine what both models do: preserving the distribution of classes and keeping each group within a single split (noa b).

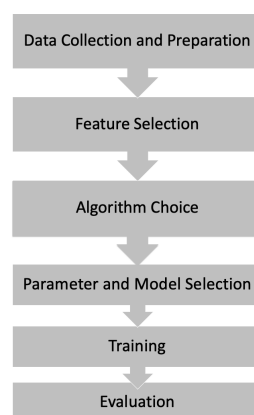
4 Research Method

The methodology used in this thesis is based on machine learning concepts. The data analysis and modeling were completed entirely in Python in a Jupyter Notebook environment with the following libraries; numpy, pandas, scikit-learn, and imbalanced-learn.

4.1 The Machine Learning Process

The proposed method in this study is based on the machine learning process proposed by Marsland (Marsland 2015). This machine learning process consists of data collection and preparation, feature selection, algorithm choice, parameter and model selection, training, and evaluation, as shown in Figure 1. This process is not linear; instead is iterative. It requires going back and forth to find the most suitable model.

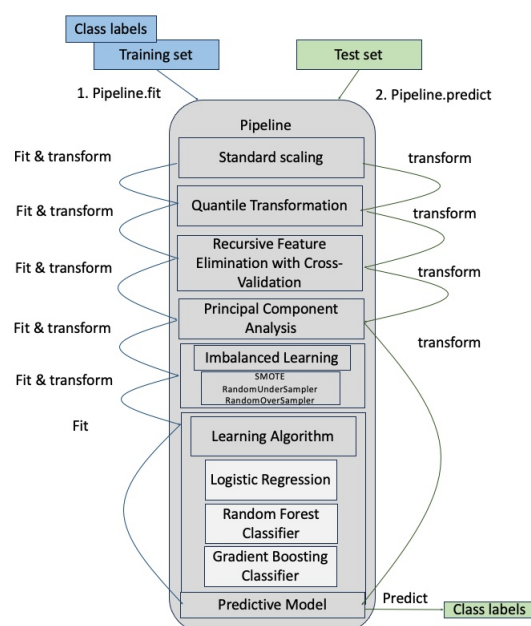
Figure 1: The machine learning process proposed by Marsland (Marsland 2015)



The machine learning pipeline used in this study is illustrated in Figure 2. A pipeline is described as a way to define several steps applied to the dataset sequentially. This integrates the typical sequence of data processing operations and modeling into a single scikit-learn estimator (Raschka and Mirjalili 2019). Implementing a pipeline allows the correct order of transformations to be applied to the dataset, preventing common mistakes such as data leakage (Raschka and Mirjalili 2019). It allows for performing grid searches and cross-validate all the model steps together. In addition, it provides a user-friendly interface for carrying out common machine learning tasks (Raschka and Mirjalili 2019).

Figure 2 shows that the pipeline first takes the training dataset, which includes the class labels; in this study, it refers to the column churned or not. Then standard scaling, quantile transformation, RFECV, PCA, and imbalanced learning are fit and transformed to the training dataset. The fit method trains each estimator subsequently. As a result, the data is changed at each step. Then the transform method applies these learned transformations from the fit method to the new dataset. As can be seen from Figure 2, for the learning algorithm, the method fit is only applied to train the models. Then the test data without the class labels are fed into the pipeline. The transform method is only applied to the test data since the test data should represent unseen data, and the model should not learn anything from the test data. In the last stage, the class label of test data is predicted by using the predictive model trained on the training dataset.

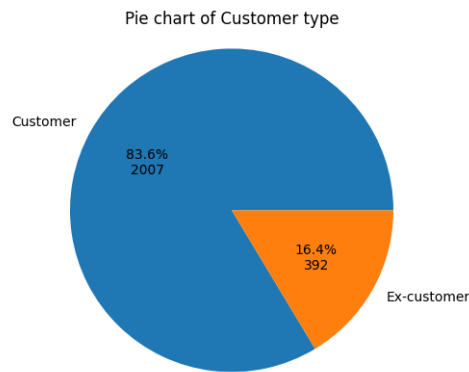
Figure 2: The machine learning process followed in this study, depicted as a pipeline diagram



4.1.1 Data Collection and Preparation

The dataset used in this study is given by Bynder. This B2B company offers a cloud-based digital asset management (DAM) platform for marketing teams to create, find, and use digital assets like images, videos, and documents. Bynder provides services to a wide variety of customers, such as businesses of all sizes, retail, education, and non-profit organizations. As shown in Figure 3, they have around 2400 customers today, of which 15% are churn customers.

Figure 3: Distribution of Customer (non-churners) and Ex-customers (churners) in the dataset.



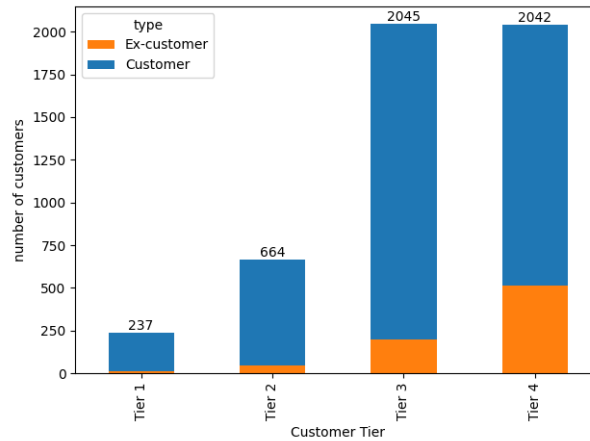
In previous studies, churn prediction in the domain of B2C has been thoroughly explored. The dataset used in such studies includes customer demographics such as gender and age, customer behavior such as transactional data, customer perceptions such as overall satisfaction, and macro-environment data. However, as mentioned previously, there hasn't been much research done for churn prediction in the domain of B2B. As discussed in Section 2. The following datasets are commonly used for churn prediction; customer product usage data and customer demographics data. The data used in this study is similar to previous studies' data.

The information regarding the customers was extracted from Salesforce. This dataset includes the following attributes per customer;

- **Type:** This feature represents whether the customer is active or churned, which is the target variable. Meaning this is the variable we are aiming to predict. This feature is based on whether or not the customer has renewed their contract.
- **Customer Tier:** This feature categorizes customers based on their Annual Recurrent Revenue (ARR). There are four tiers, with tier 1 being the

highest value customers. The distribution of this feature is shown in Figure 4. Tier 4 and 3 have the highest number of Ex-customers (churners).

Figure 4: Distribution of customer tier in the dataset.



- Company Size:** This feature categorizes customers based on the size of their company. It has the following values; SMB (small and medium-sized business), Enterprise, Strategic, MM - Small' (the smaller end of the mid-market), MM - Large (the larger end of the mid-market), Mid-Market (businesses with resources and size that are in the middle of the market). The distribution of the feature is shown in Figure 5, which shows that SMB has the highest number of customers in total.

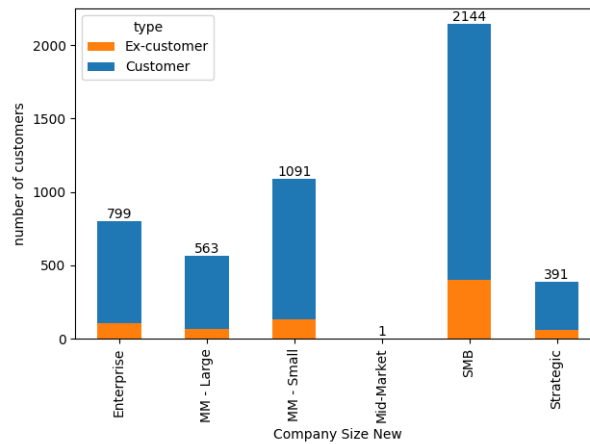
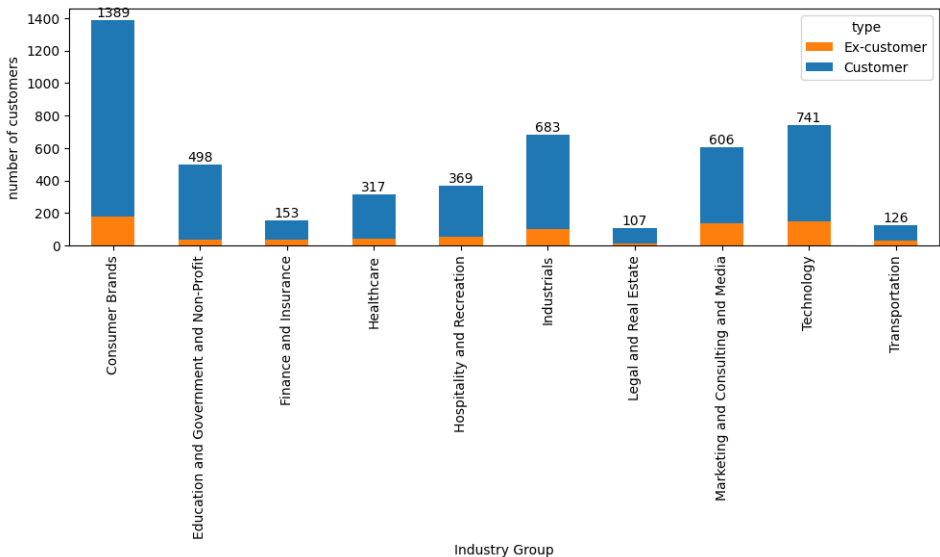


Figure 5: Distribution of company size in the dataset

- Industry:** This feature groups customers based on the following industry types; Education and Government and Non-Profit, Technology, Industrials, Consumer Brands, Finance and Insurance, Hospitality and Recreation, Marketing and Consulting and Media, Transportation,

Healthcare, and Legal and Real Estate. The distribution of the feature is shown in Figure 5, Marketing and Consulting and Media, Technology, and Consumer Brands have the highest number of ex-customers.

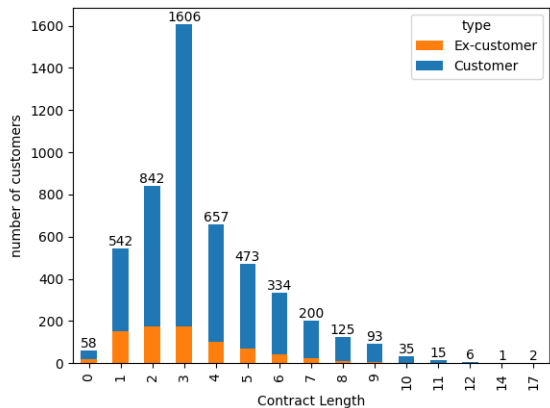
Figure 6: Distribution of industry group in the dataset



- Contract End Date: The date indicating when the current contract ends.
- Contract Start Date: This is the first contract start date of each customer.

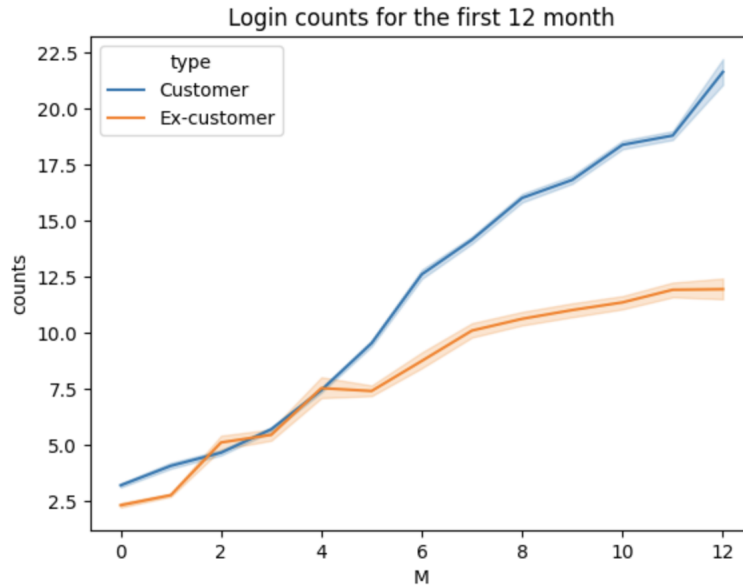
Figure 7 illustrates the distribution of contract length, the contract length for each customer was determined by subtracting their Contract End Date from their Contract Start Date.

Figure 7: Distribution of contract length in the dataset



In addition, the DAM usage data, which is time series data, was extracted from DataLake. This data was aggregated to show the number of activities per DAM feature in a month rather than a day. Figure 8 shows the trend for the login feature between customer and ex-customer for the first 12 months of a customer joining Bynder. The trend is quite similar between the two, which is an upward trend in the logins. There is an intersection from the second to the fourth month.

Figure 8: The login count of the first 12 months of customers (non-churners) and ex-customers (churners).



These two datasets were then merged on the account id. This resulted in a new dataset with each row representing a customer product usage and their information. The data from before 2018 was dropped as it was irrelevant to the current customers, and the DAM product had been changed quite a bit. Due to the limited size of the data and high imbalance, two methods of preparing data for the classification models were conducted.

Single Year Cross-Sectional Data

The First method uses an observation window of one year, which starts two years before the contract end date, shown in Figure 9. An observation period of one year was suitable since the data from previous years may not be relevant today, considering that DAM has changed through the years and new features have been added. A churn window of six months, starting a year before the contract end date. The churn window of 6 months was chosen because Bynder needed a sufficient amount of time to take action to prevent customers from churning.

The data is then aggregated so that each row represents a unique customer. This aggregation is done by the summation of all counts of a customer for each DAM feature during the observation period.

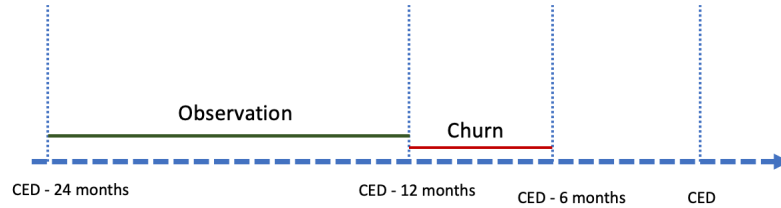


Figure 9: Observation window and churn window for churn classification

Multi-Year Cross-Sectional Data

The second method uses historical data of the customers and treats data from each year as a unique customer. For example, if customer 1 has been with Bynder from 2019 to 2023, customer 1 in 2019, and customer 1 in 2020 are seen as two separate customers. The data from each year is used to predict churn for that specific year. The target of the most recent year is determined the same way as the other method. However, for the previous years, they have been labeled as active customers. The data from this method is also aggregated. However, the aggregated data contains multiple rows per customer based on the year.

Before aggregating the data from both methods, the outliers are detected using the interquartile range (IQR) method. This approach works by calculating the quantiles and then the inter-quantile range (Vinutha, Poornima, and Sagar 2018). An outlier is defined as a data point either above or below the upper or lower border (Vinutha, Poornima, and Sagar 2018). Since the data is already limited in size and heavily imbalanced, dropping the outliers could result in even fewer churned customers making the churn prediction nearly impossible. Thus, the extreme outliers are replaced by the mean or the most common value.


The data from both methods contains customer information, which consists of categorical variables. These variables need to be converted to numerical values; therefore, one hot encoding was implemented. One hot encoding transfers categorical data into a new column with binary values, one meaning true and zero false. An example of this procedure is shown in Table 10, as mentioned previously, the column customer tier has four values, by applying one hot encoding, each of these values is assigned to a new column. For column tier 1, in row one, the value is 0, meaning this customer is not a tier 1 customer. However, row 2 shows value 1 for tier 1, which means this customer belongs to tier 1.

In addition, the data should be scaled since some machine learning models have normalization assumptions. It can also improve the performance of the model by making it easier for the model to learn and understand the data. There are different methods of scaling data, including; Min Max Scaler, Standard Scaler, Max Abs Scaler, Robust Scaler, and Quantile Transformer Scaler. For the purpose of this study, the standard scaler and quantile transformer is implemented.

Because of its simplicity and efficacy, the Standard Scaler is preferred over alternative methods. It changes the data to have a mean of 0 and a standard deviation of 1, retaining the original distribution (Muller and Guido 2016). It performs well when

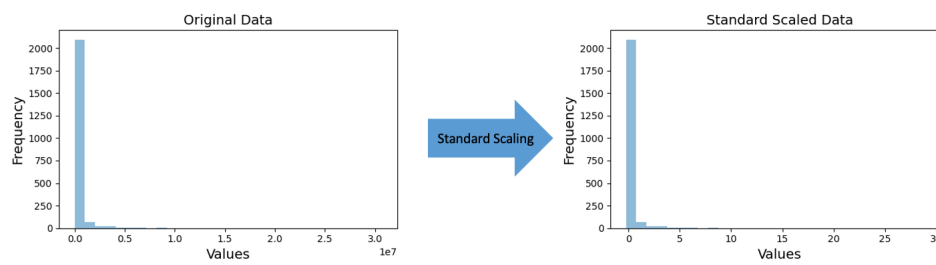
Figure 10: This an example of a binary representation of a categorical variable (customer tier) consisting of four different values

Categorical values				
Tier 1				
Tier 2				
Tier 3				
Tier 4				



Tier 1	Tier 2	Tier 3	Tier 4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

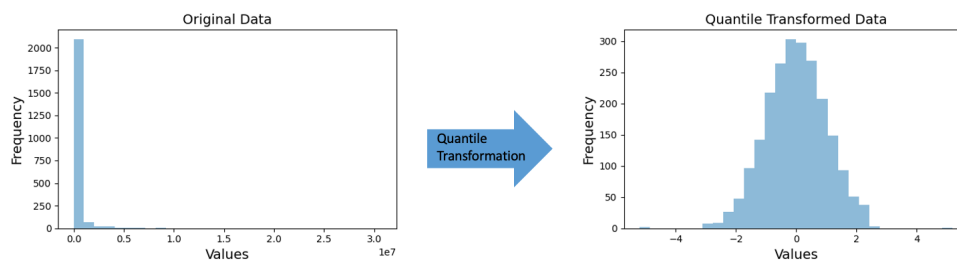
Figure 11: An example of standard scaling of the user login feature.



the data is normally distributed or unknown (Ferreira, Le, and Zincir-Heywood 2019). However, it is sensitive to outliers and cannot perform well with sparse data (Ferreira, Le, and Zincir-Heywood 2019). Figure 11 shows an example of how standard scalar transforms the data.

Quantile Transformation is employed since the data is not normally distributed and is heavily skewed. It limits the impact of outliers since it maps the data to a uniform or normal distribution, making it resistant to outliers (Ferreira, Le, and Zincir-Heywood 2019). Furthermore, it is especially beneficial when linear correlation measures are insufficient (Ferreira, Le, and Zincir-Heywood 2019).

Figure 12: An example of the quantile transformation of the user login feature.



As can be seen in Figure ??, the data is skewed, which is not desirable due to the following disadvantages; it can affect the training process of the model since it is difficult for the model to learn patterns in skewed data (Larasati, Hajji, and Dwiastuti 2019). It can make the model sensitive to outliers and less robust (Brys, Hubert, and Struyf 2003). It also can introduce bias in the model's performance because machine learning algorithms often assume the data follows a normal distribution, such as logistic regression. There are different methods for transforming the data distribution, such as box-cox transformation. However, for the purpose of this study, the quantile transformation method has been applied. It maps the data to a uniform or normal distribution, making it resistant to outliers (Ferreira, Le, and Zincir-Heywood 2019). Furthermore, it is especially beneficial when linear correlation measures are insufficient (Ferreira, Le, and Zincir-Heywood 2019).

4.2 Feature Selection

Feature selection is an essential step in machine learning because it lowers the number of predictors, increases the effectiveness of model training, and lowers memory requirements. It also helps with over-fitting. In this study, two dimensionality reduction techniques will be applied. One for selecting the most important features and one for dealing with the correlation between the features.

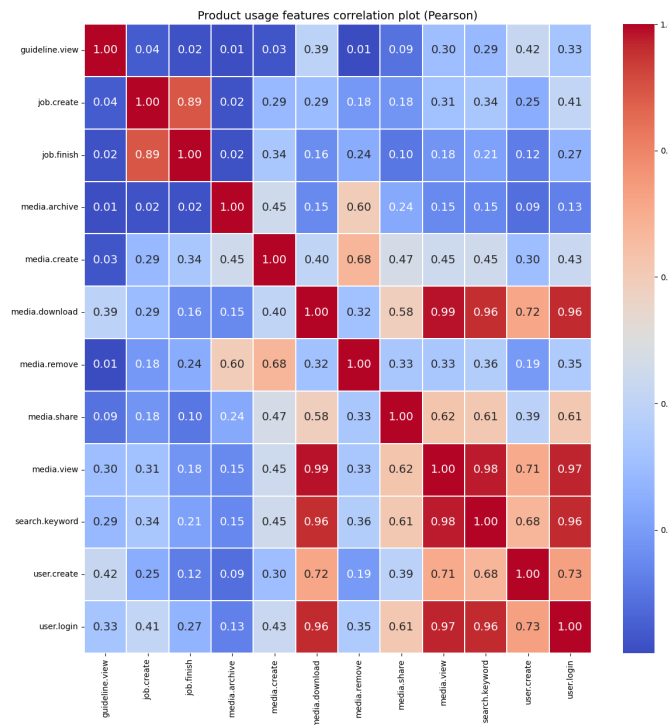
4.2.1 Recursive feature elimination with cross-validation

There are various techniques for selecting the most influential features, including k best feature, Pearson correlation, chi-squared, and the feature importance selected by the model. In this study, RFECV is chosen as it considers the interdependencies between features in addition to choosing the most informative features. RFECV determines the subset of most pertinent features and increases the model's precision by taking the model's performance during the elimination phase into account (Mustaqim et al. 2021). The benefit of RFECV is that it offers an automatic and reliable technique. Maximizing the number of features based on cross-validation lowers the danger of overfitting and ensures greater generalization to unknown data (Mustaqim et al. 2021). Additionally, by concentrating on the most crucial features, RFECV improves interpretability and enables a better understanding of the underlying connections between the predictors and the target variable (Mustaqim et al. 2021). In this study, RFECV is performed on the preprocessed dataset, as seen in Figure 2. The recall was chosen as the score for the cross-validation since a subset of features is desired that can predict the churned customers accurately. This process is done for each model, including logistic regression, random forest classifier, and gradient boosting classifier.

4.2.2 Principal Component Analysis

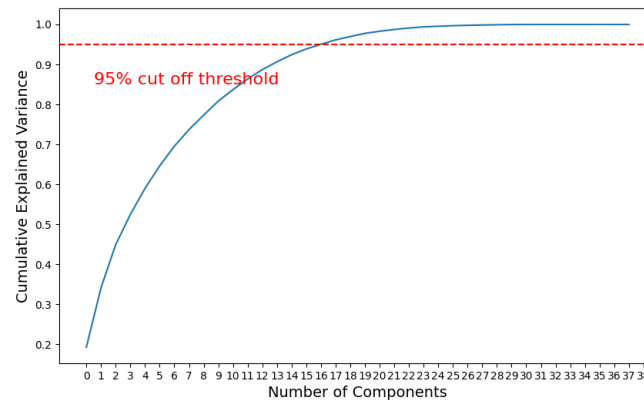
The DAM features are correlated with each other, as can be seen in Figure 13. If this issue isn't handled, it can result in redundancy and affect model performance. Thus, in this study, Principal Component Analysis (PCA) was performed to deal with these features. PCA is a statistical technique that transforms a dataset with potentially correlated features into a new set of uncorrelated variables called principal components (Hasan and Abdulazeez 2021).

Figure 13: Correlation plot of DAM usage features (Pearson)



The number of new uncorrelated variables can be specified with the parameter number of components (`n_components`). In order to find the most suitable number of variables, the cumulative explained variance ratio for PCA was plotted. The x-axis represents the number of components, while the y-axis represents the cumulative explained variance. The plot shows how much variance is explained as more components are added. As seen in Figure 14, at 95% cutoff point, 16 components can explain 95% of the total variance in the dataset. This procedure was performed after selecting the best features for each model to indicate the number of components.

Figure 14: Number of Components Required for Variance Explanation



4.3 Algorithm Choice

The optimal algorithm for churn prediction can vary. There are numerous binary classification algorithms, such as Naïve Bayes, Random Forest, XGBoost, Support Vector Machine, Logistic Regression, and Hidden Markov's Model. Previous studies have shown that ensemble learners outperform other classifiers. Thus, the following algorithms were chosen based on the previous research; gradient boosting classifier, which applies boosting, and Random Forest, which applies bagging. In addition, logistic regression is also extensively used for churn prediction. The result of these classifiers will be compared with the baseline model that Bynder is currently using.

4.4 Parameter and Model Selection

The majority of machine learning algorithms require hyperparameter tuning, which is a process of optimizing the parameters of a machine learning algorithm and choosing the parameters that result in better model performance. Due to time limitations, the default parameter and values for the algorithms and models were used in this study.

4.5 Training

The training method entails using labeled data to build a model that, ideally, performs well on unknown data. The dataset should be split into training data, used for training and fitting the model, and test data, used for evaluating the model's performance. The training and test size is split with the proportions of 75/25. This is due to the small number of churned customers; if the split ratio were higher, the number of churned customers in the test data would be too small to evaluate the model's performance. On the other hand, if the split ratio were lower, there would not be enough churned customers in the training data for the model to learn and predict churn accurately. For single year cross-sectional data, the data is split using `train_test_split` from `sklearn` library. It has a parameter called `stratify`, which ensures a proportional representation of classes in both sets. When set to the target variable, the resulting test and train data

will have the same proportions of churned versus non-churned. In this case, both will have around 15% churned customers and 85% active customers. On the other hand, for multi-year cross-sectional data, `train_test_apart_stratify` from `pandas_streaming.df` is used. This is similar to `train_test_split`; however, it has another parameter called `group`, which allows for preserving the group membership when spitting data. By setting the account id as the group parameter, `train_test_apart_stratify` ensures no customer will be in training and test data. Otherwise, this could lead to data leakage because the model would have access to information about the future.

Classification problems often do not have a balanced dataset meaning the dataset includes a majority class and a minority class. This is the case for churn classifications, as the majority of the dataset includes a smaller number of churned customers in comparison to active customers. Machine learning models do not perform well with an imbalanced dataset, as they overfit the majority class and cannot predict the minority class. In order to handle this issue, either undersampling or oversampling is applied to the data set. Undersampling removes data points from the majority class to make the dataset balance. While oversampling works by adding data points to the minority class through various techniques. These methods should only be applied to the training data since the test data should represent the actual distribution of classes to provide an accurate evaluation of the model's performance. In this study, `RandomUnderSampler`, for undersampling the majority class, `SMOTE`, and `RandomOverSampler` for oversampling the minority class is used.

4.5.1 Model Evaluation

The classifier's performance should be evaluated as a vital component of the machine learning process. The majority of the previous studies evaluated their models based on accuracy, precision, recall, and F-measure. To assess the performance of the predictive model, the mentioned indicators will be used. However, in the case of imbalanced data, using accuracy is not suggested. For instance, if there are 100 active customers and only then 10 churned customers, the model will be able to achieve a high accuracy just by predicting the active customer. To address this issue, the recall, precision, and F1-scores are mainly discussed. For the purpose of this study, the aim is to have a low false negative rate. Thus, recall is mainly used to assess the model's ability. In addition, the AUC curve will be used to assess how well the model distinguishes between positive and negative instances in a single metric. A higher value indicates better performance. To detect if the model is overfitting, stratified K-fold cross-validation for single-year cross-sectional data and stratified group K-fold cross-validation for multi-year cross-sectional data are used. Overfitting occurs when the model fails to generalize and, instead, fits the training dataset too closely.

5 Result & Analysis

In this section, the result of the three classifiers, Random Forest, Gradient Boosting, and Logistic Regression, is presented for both single-year cross-sectional data and multi-year. In Table 2, the overall performance of the three classifiers with single-year cross-sectional data is presented using accuracy, see (Eq. 6), F1-score, see (Eq. 8), recall, see (Eq. 6), and precision, see (Eq.7). Similarly, Table 3 represents the performance of these classifiers with multi-year cross-sectional data. To fully evaluate the performance of each model on the target class, in this case, the churned customers, a confusion matrix, for both classes, precision, recall, and F1-score, are stated. In addition, the AUC curve

of each model is plotted. Lastly, the result of K-fold cross-validation is also shown. For a description of the mentioned evaluation matrices, see Section 3.3.

Table 2: Performance Metrics of Different Models with single year cross-sectional data using the default imbalanced data and the three sampling methods. Each cell contains two values; the value to the left represents RFECV, and the right value represents without RFECV.

Single Year Cross-Sectional Data					
Model	Evaluation Matrices	Default	RandomUnderSampler	SMOTE	RandomOverSampler
Random Forest	Accuracy	0.85 / 0.86	0.80 / 0.68	0.76 / 0.79	0.83 / 0.83
	Recall	0.25 / 0.12	0.41 / 0.5	0.51 / 0.32	0.34 / 0.18
	F1-Score (Weighted average)	0.32 / 0.12	0.36 / 0.3	0.36 / 0.29	0.35 / 0.22
	Precision	0.43 / 0.45	0.32 / 0.22	0.29 / 0.26	0.37 / 0.29
Gradient Boosting	Accuracy	0.85 / 0.86	0.81 / 0.65	0.74 / 0.75	0.77 / 0.76
	Recall	0.16 / 0.13	0.38 / 0.59	0.53 / 0.53	0.47 / 0.49
	F1-Score (Weighted average)	0.22 / 0.21	0.36 / 0.32	0.36 / 0.37	0.36 / 0.35
	Precision	0.39 / 0.48	0.33 / 0.22	0.27 / 0.28	0.29 / 0.28
Logistic Regression	Accuracy	0.86 / 0.86	0.79 / 0.66	0.70 / 0.70	0.69 / 0.69
	Recall	0.09 / 0.13	0.34 / 0.63	0.66 / 0.68	0.68 / 0.67
	F1-Score (Weighted average)	0.15 / 0.21	0.31 / 0.34	0.37 / 0.38	0.38 / 0.37
	Precision	0.44 / 0.48	0.28 / 0.23	0.26 / 0.26	0.26 / 0.26

The results for these classifiers are presented so that each cell in the considered tables is represented by two values. The value to the left shows the classifier's performance by conducting feature selection using RFECV, and the value to the right represents the results without removing non-important features. This is the case for all the tables and confusion matrices.

The classifiers are evaluated using the default imbalanced dataset, and the datasets obtained after implementing RandomUnderSampler, RandomOverSampler, and SMOTE, see Section 3.2.9.

The result of all three classifiers for multi-year cross-sectional data performs better than for single-year cross-sectional data when considering recall and accuracy. While single-year cross-sectional data has a better performance when considering precision and f1-score. This is due to the trade-off between recall and precision, meaning increasing one will result in a decrease in the other one. The result of the classifier when using SMOTE and RandomOverSampler are pretty similar to each other since they are both oversampling strategies.

The result of the Random Forest classifier for single-year cross-sectional data has an accuracy ranging between 0.76-0.86, which indicates the model can classify the majority of test instances correctly. The highest recall value was 0.51, obtained by performing feature selection using RFECV and smote as a sampling method. On the other hand, Random Forest for multi-year cross-sectional data has an accuracy in the range of 0.81-0.95, which is slightly higher than the one achieved by using single-year cross-sectional

Table 3: Performance Metrics of Different Models with multi-year cross-sectional data using the default imbalanced data and the three sampling methods. Each cell contains two values; the value to the left represents RFECV, and the right value represents without RFECV.

Multi-Year Cross-Sectional Data					
Model	Evaluation Matrices	Default	RandomUnderSampler	SMOTE	RandomOverSampler
Random Forest	Accuracy	0.95 / 0.94	0.85 / 0.81	0.91 / 0.87	0.95 / 0.92
	Recall	0.07 / 0.07	0.62 / 0.45	0.38 / 0.36	0.13 / 0.12
	F1-Score (Weighted average)	0.12 / 0.10	0.31 / 0.19	0.21 / 0.12	0.19 / 0.13
	Precision	0.5 / 0.17	0.31 / 0.19	0.21 / 0.12	0.36 / 0.14
Gradient Boosting	Accuracy	0.95 / 0.95	0.81 / 0.66	0.74 / 0.80	0.78 / 0.80
	Recall	0.05 / 0.00	0.5 / 0.72	0.76 / 0.64	0.65 / 0.62
	F1-Score (Weighted average)	0.09 / 0.00	0.21 / 0.18	0.23 / 0.24	0.23 / 0.24
	Precision	0.31 / 0.00	0.13 / 0.18	0.13 / 0.15	0.14 / 0.15
Logistic Regression	Accuracy	0.95 / 0.95	0.85 / 0.71	0.70 / 0.70	0.69 / 0.69
	Recall	0.00 / 0.03	0.51 / 0.75	0.71 / 0.71	0.72 / 0.72
	F1-Score (Weighted average)	0.00 / 0.05	0.25 / 0.20	0.21 / 0.21	0.20 / 0.20
	Precision	0 / 0.66	0.16 / 0.11	0.12 / 0.12	0.12 / 0.12

data. The highest recall value is 0.62, achieved by RandomUnderSampler and RFECV, which is 10 percent higher than the single-year cross-sectional data.

The gradient boosting classifier has an accuracy of 0.65-0.86 for single-year cross-sectional data. In contrast, it achieves slightly better accuracy in multi-year cross-sectional data ranging between 0.66-0.95. The best recall score for single-year cross-sectional data is 0.59 by implementing RandomUnderSampler. On the other hand, for multi-year cross-sectional data, the highest recall value is 0.76 through implementing SMOTE and feature selection with RFECV.

The Logistic Regression classifier achieves an accuracy ranging from 0.69 to 0.86 for single-year cross-sectional data. The best recall value is achieved by using SMOTE or RandomOverSampler with RFECV. The Logistic Regression classifier with multi-year cross-sectional data has similar accuracy to single-year cross-sectional data, ranging between 0.69-0.95. The best recall value is obtained with RandomUnderSampler and RFECV, 0.75. From this, it can be concluded that the Gradient Boosting classifier for multi-year cross-sectional data with SMOTE and RFECV achieves the highest recall among all models, 0.76. In addition, it can be seen that although the default models have high accuracy, the recall, precision, and f1-score are close to 0. This means they can only predict the majority class and fail to estimate the minority class.

To evaluate the performance of each classifier on the churn class, class 1 (positive class), and active customer class, class 0 (negative class), from both single-year cross-sectional data and multi-year cross-sectional data, confusion matrix, AUC curve, and the related metrics for every classifier are presented. The relevant metrics can be calcu-

lated using the confusion matrix. In this study, precision refers to how often the model predicts churn correctly or the rate of correct churn predictions, as shown in (Eq. 7). By increasing precision, True Positives are increased, and False Positives are reduced. The recall is the rate the model can predict actual churners, as seen in (Eq. 8). By increasing recall, True Positives are increased, and False Negatives are reduced. The F1-score is a metric aggregating precision and recall and depicts a balance between both (Eq. 9). In addition, The ROC curve is plotted; it visualizes how well the model can distinguish between the two classes, illustrated in (Eq. 10)

Random Forest Classifier

Table 4: Performance of Random Forest classifier with single-year cross-sectional data

Single Year Cross-Sectional Data					
Random Forest					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.89 / 0.87	0.90 / 0.90	0.91 / 0.89	0.90 / 0.88
	1	0.43 / 0.45	0.32 / 0.22	0.29 / 0.26	0.37 / 0.29
Recall	0	0.95 / 0.98	0.86 / 0.71	0.80 / 0.86	0.91 / 0.93
	1	0.25 / 0.12	0.41 / 0.50	0.51 / 0.32	0.34 / 0.18
F1-Score (Weighted average)	0	0.92 / 0.92	0.88 / 0.80	0.85 / 0.87	0.90 / 0.90
	1	0.32 / 0.19	0.36 / 0.30	0.37 / 0.29	0.35 / 0.22

FP 25 / 11	TP 19 / 9
FN 57 / 67	TN 454 / 468

Table 5:
Confusion matrix of default imbalanced dataset with or without RFECV as feature selection.

FP 67 / 137	TP 31 / 38
FN 45 / 38	TN 412 / 342

Table 6:
Confusion matrix of RandomUnderSampler strategy with or without RFECV as feature selection.

FP 97 / 67	TP 39 / 24
FN 37 / 52	TN 382 / 412

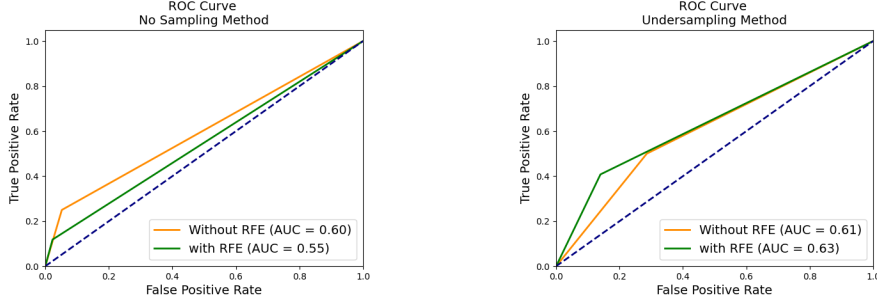
Table 7:
Confusion matrix of SMOTE strategy with or without RFECV as feature selection.

FP 45 / 35	TP 26 / 14
FN 50 / 62	TN 434 / 444

Table 8:
Confusion matrix of RandomOverSampler strategy with or without RFECV as feature selection.

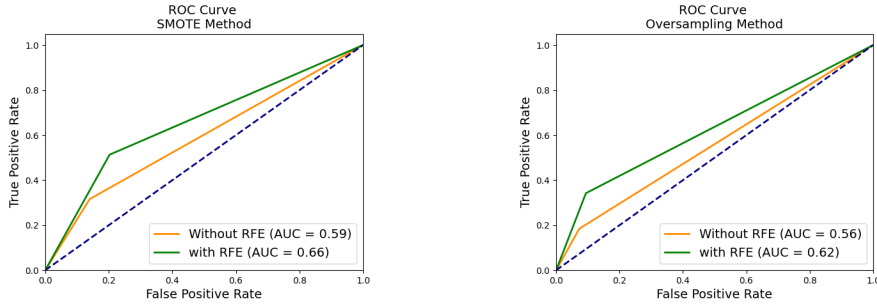
In Table 4, the results of Random Forest with single-year cross-sectional data, each cell contains two values; the value to the left represents the value with feature selection using RFECV, and the value to the right represents without it. The result shows that the Random Forest classifier does not perform well in predicting the churn class, class 1. Using the default imbalanced dataset, the non-churners, class 0, have better results than the churn class. By implementing RFECV, it achieves a recall of 0.25, which means the model can only correctly predict 25 percent of the actual churned customers. At the same time, the precision is 0.43 and 0.45 with or without RFECV, respectively. This indicates 0.45 or 0.43 of the customers that the model predicted as churned did

Figure 15: ROC curve of Random Forest classifier with single-year cross-sectional data



(a) ROC curve of default imbalanced dataset with or without RFECV as feature selection

(b) ROC curve of RandomUnderSampler strategy with or without RFECV as feature selection



(c) ROC curve of SMOTE strategy with or without RFECV as feature selection

(d) ROC curve of RandomOverSampler strategy with or without RFECV as feature selection

churn. The F1-score is 0.32 and 0.19 with and without RFECV. This implies the model performs poorly on the imbalanced training dataset. The model's performance increases by the sampling strategies and achieves the highest recall score of 0.51 with SMOTE and RFECV. This is also shown in the confusion matrix in Table 7, where the number of FN is 37, which is the lowest among the others. As shown in Figure 16b, the random forest classifier with single-year cross-sectional data performs poorly since the ROC curves are pretty close to the dotted line, which represents just random guessing.

The result of the Random Forest Classifier with multi-year cross-sectional data is presented in Table 9. The performance of the default imbalanced dataset is similar to the Random Forest classifier with single-year cross-sectional data. However, the best recall score for the churned class, class 1, is achieved using RandomUnderSampler. This can also be observed by looking at the confusion matrix presented in Table 11; the number of FN is 29, which is the lowest. However, the FP is 178, meaning the model classifies active customers as churned customers. Figure 9 illustrates that RandomOverSample results in a poor AUC score in comparison to the other sampling strategies. The best AUC score, 0.75, is achieved by RandomUnderSampler with RFECV. It can be seen in Figure 11 that feature selection using RFECV has improved the AUC by 10 percent.

Table 9: Performance Metrics of Random Forest Classifier with multi-year cross-sectional data

Multi Year Cross-Sectional Data					
Random Forest					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.95 / 0.95	0.98 / 0.97	0.97 / 0.96	0.96 / 0.95
	1	0.50 / 0.17	0.21 / 0.12	0.24 / 0.16	0.36 / 0.14
Recall	0	1.00 / 0.98	0.88 / 0.83	0.94 / 0.90	0.99 / 0.96
	1	0.07 / 0.07	0.62 / 0.45	0.38 / 0.36	0.13 / 0.12
F1-Score (Weighted average)	0	0.97 / 0.97	0.92 / 0.89	0.95 / 0.93	0.97 / 0.96
	1	0.12 / 0.10	0.31 / 0.19	0.30 / 0.22	0.19 / 0.13

FP	TP
5 / 24	5 / 5
FN	TN
71 / 71	1442 / 1423

Table 10: Confusion matrix of default imbalanced dataset with or without RFECV as feature selection

FP	TP
178 / 243	47 / 34
FN	TN
29 / 42	1269 / 1204

Table 11: Confusion matrix of RandomUnderSampler strategy with or without RFECV as feature selection

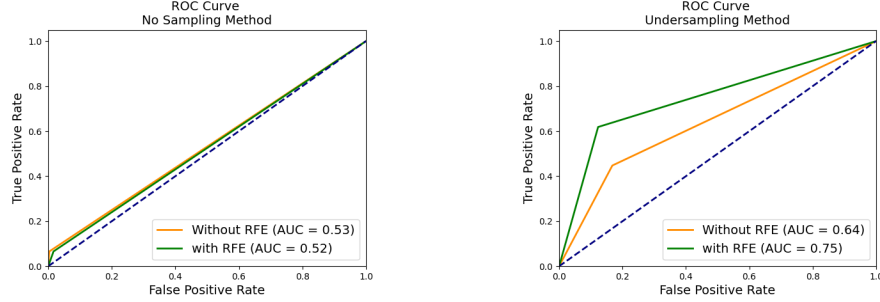
FP	TP
91 / 145	29 / 27
FN	TN
47 / 49	1356 / 1302

Table 12: Confusion matrix of SMOTE strategy with or without RFECV as feature selection

FP	TP
18 / 54	10 / 9
FN	TN
66 / 67	1429 / 1393

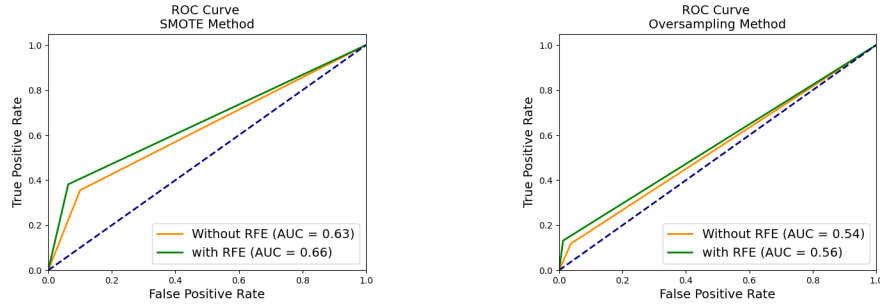
Table 13: Confusion matrix of RandomOverSampler strategy with or without RFECV as feature selection

Figure 16: ROC curve of Random Forest classifier with Multi-year cross-sectional data



(a) ROC curve of default imbalanced dataset with or without RFECV as feature selection

(b) ROC curve of RandomUnderSampler strategy with or without RFECV as feature selection



(c) ROC curve of SMOTE strategy with or without RFECV as feature selection

(d) ROC curve of RandomOverSampler strategy with or without RFECV as feature selection

Gradient Boosting Classifier

Table 14: Performance Metrics of Gradient Boosting Classifier with Single Year Cross-Sectional data

Single Year Cross-Sectional Data					
Gradient Boosting					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.88 / 0.88	0.90 / 0.91	0.91 / 0.91	0.91 / 0.91
	1	0.39 / 0.48	0.33 / 0.22	0.27 / 0.28	0.29 / 0.28
Recall	0	0.96 / 0.98	0.88 / 0.66	0.77 / 0.79	0.82 / 0.80
	1	0.16 / 0.13	0.38 / 0.59	0.53 / 0.53	0.47 / 0.49
F1-Score (Weighted average)	0	0.92 / 0.92	0.89 / 0.77	0.84 / 0.85	0.86 / 0.85
	1	0.22 / 0.21	0.36 / 0.32	0.36 / 0.37	0.36 / 0.35

In Table 14, the result of the Gradient Boosting classifier with single-year cross-sectional data is presented. The result shows that feature selection using RFECV did not

<table><tr><td>FP</td><td>TP</td></tr><tr><td>19 / 11</td><td>12 / 10</td></tr><tr><td>FN</td><td>TN</td></tr><tr><td>64 / 66</td><td>460 / 468</td></tr></table>	FP	TP	19 / 11	12 / 10	FN	TN	64 / 66	460 / 468	<table><tr><td>FP</td><td>TP</td></tr><tr><td>58 / 161</td><td>29 / 45</td></tr><tr><td>FN</td><td>TN</td></tr><tr><td>47 / 31</td><td>421 / 318</td></tr></table>	FP	TP	58 / 161	29 / 45	FN	TN	47 / 31	421 / 318	<table><tr><td>FP</td><td>TP</td></tr><tr><td>109 / 102</td><td>40 / 40</td></tr><tr><td>FN</td><td>TN</td></tr><tr><td>36 / 36</td><td>370 / 377</td></tr></table>	FP	TP	109 / 102	40 / 40	FN	TN	36 / 36	370 / 377	<table><tr><td>FP</td><td>TP</td></tr><tr><td>88 / 97</td><td>36 / 37</td></tr><tr><td>FN</td><td>TN</td></tr><tr><td>40 / 39</td><td>391 / 382</td></tr></table>	FP	TP	88 / 97	36 / 37	FN	TN	40 / 39	391 / 382
FP	TP																																		
19 / 11	12 / 10																																		
FN	TN																																		
64 / 66	460 / 468																																		
FP	TP																																		
58 / 161	29 / 45																																		
FN	TN																																		
47 / 31	421 / 318																																		
FP	TP																																		
109 / 102	40 / 40																																		
FN	TN																																		
36 / 36	370 / 377																																		
FP	TP																																		
88 / 97	36 / 37																																		
FN	TN																																		
40 / 39	391 / 382																																		
Table 15: Confusion matrix of default imbalanced dataset with or without RFECV as feature selection	Table 16: Confusion matrix of RandomUnderSampler strategy with or without RFECV as feature selection	Table 17: Confusion matrix of SMOTE strategy with or without RFECV as feature selection	Table 18: Confusion matrix of RandomOverSampler strategy with or without RFECV as feature selection																																

significantly influence performance. The gradient Boosting classifier obtains the highest recall with RandomUnderSample, 0.59. However, as mentioned, due to the trade-off between precision and recall, it has a precision of 22 and an F1-score of 0.32. Indicating the classifier is performing quite poorly.

Figure 17: ROC curve of Gradient Boosting classifier with single-year cross-sectional data

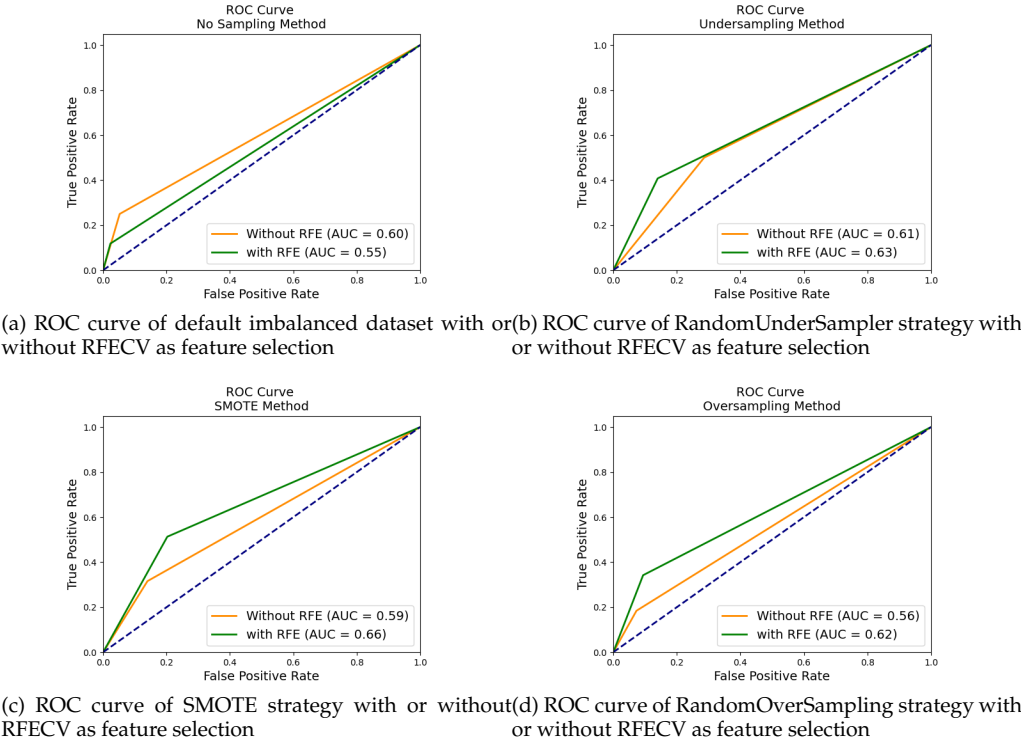


Table 19: Performance Metrics of Gradient Boosting Classifier with Multi-Year Cross-Sectional Data

Multi Year Cross-Sectional Data					
Gradinet boosting					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.95 / 0.95	0.97 / 0.98	0.98 / 0.98	0.98 / 0.98
	1	0.31 / 0.00	0.13 / 0.10	0.13 / 0.15	0.14 / 0.15
Recall	0	0.99 / 1.00	0.83 / 0.66	0.74 / 0.81	0.79 / 0.81
	1	0.05 / 0.00	0.50 / 0.72	0.76 / 0.64	0.66 / 0.62
F1-Score (Weighted average)	0	0.97 / 0.97	0.89 / 0.79	0.84 / 0.88	0.87 / 0.89
	1	0.09 / 0.00	0.21 / 0.18	0.23 / 0.24	0.23 / 0.24

FP 9 / 7	TP 4 / 0
FN 72 / 76	TN 1438 / 1440

Table 20:
Confusion matrix of default imbalanced dataset with or without RFECV as feature selection

FP 248 / 493	TP 38 / 55
FN 38 / 21	TN 1199 / 354

Table 21:
Confusion matrix of RandomUnderSampler strategy with or without RFECV as feature selection

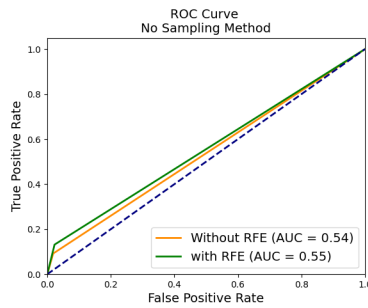
FP 378 / 280	TP 58 / 49
FN 18 / 27	TN 1069 / 1167

Table 22:
Confusion matrix of SMOTE strategy with or without RFECV as feature selection

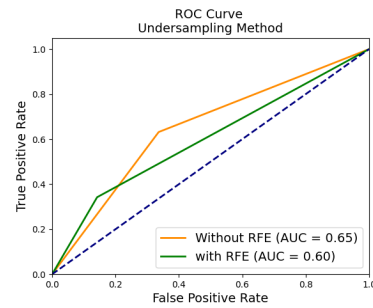
FP 308 / 274	TP 50 / 47
FN 26 / 29	TN 1139 / 1173

Table 23:
Confusion matrix of RandomOverSampler strategy with or without RFECV as feature selection

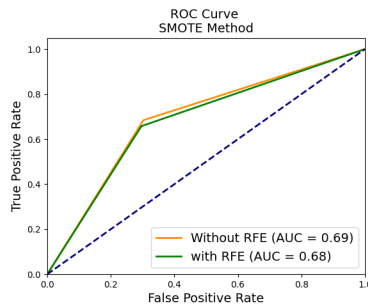
Figure 18: ROC curve of Gradient Boosting classifier with multi-year cross-sectional data



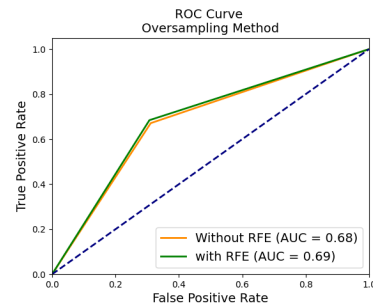
(a) ROC curve of default imbalanced dataset with or without RFECV as feature selection



(b) ROC curve of RandomUnderSampler strategy with or without RFECV as feature selection



(c) ROC curve of SMOTE strategy with or without RFECV as feature selection



(d) ROC curve of RandomOverSampler strategy with or without RFECV as feature selection

The ROC curves are similar to the ones from the gradient boosting with single-year cross-sectional data. However, there is an improvement in the AUC scores of SMOTE and RandomOverSampler. In Table 19, the result of the Gradient Boosting classifier with multi-year cross-sectional data is presented. The gradient-boosting classifier has a relatively high recall compared to the Gradient Boosting classifier with single-year cross-sectional data. It achieves the highest recall with SMOTE and RFECV, which is 0.76. Similar to the Gradient Boosting classifier with single-year cross-sectional data, the precision is relatively small, only 0.15%. Which then results in a small F1-score, 0.24. It has FN of 18 and TP of 58, the highest among the other sampling strategies.

Logistic Regression

As can be seen in Table 24 and Table 29, the logistic regression performs slightly better with multi-year cross-sectional data. The highest recall score in logistic regression with single-year cross-sectional data is 0.68 with SMORE for the churned class, while it has a recall score of 0.75 for multi-year cross-sectional data with RandomUnderSampling. In addition, applying RFECV for feature selection did not have a significant effect on the results.

Table 24: Performance Metrics of Logistic Regression Classifier with Multi-Year Cross-Sectional Data

Multi Year Cross-Sectional Data					
Logistic Regression					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.95 / 0.95	0.97 / 0.98	0.98 / 0.98	0.98 / 0.98
	1	0.00 / 0.67	0.16 / 0.12	0.12 / 0.12	0.12 / 0.11
Recall	0	1.00 / 1.00	0.86 / 0.71	0.73 / 0.73	0.72 / 0.72
	1	0.00 / 0.03	0.51 / 0.75	0.72 / 0.71	0.72 / 0.70
F1-Score (Weighted average)	0	0.97 / 0.97	0.91 / 0.82	0.84 / 0.84	0.83 / 0.83
	1	0.00 / 0.05	0.25 / 0.20	0.21 / 0.21	0.20 / 0.20

FP	TP
0 / 1	0 / 2
FN	TN
76 / 74	1477 / 1446

Table 25: Confusion matrix of default imbalanced dataset with or without RFECV as feature selection

FP	TP
199 / 425	39 / 57
FN	TN
37 / 19	1248 / 1022

Table 26: Confusion matrix of RandomUnderSample strategy with or without RFECV as feature selection

FP	TP
393 / 393	55 / 54
FN	TN
21 / 22	1054 / 1054

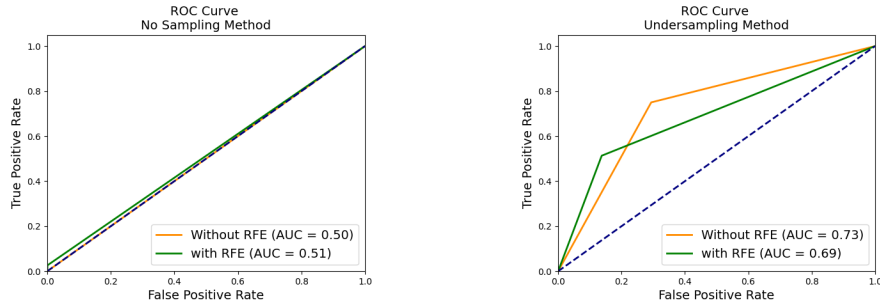
Table 27: Confusion matrix of RandomOverrSample strategy with or without RFECV as feature selection

FP	TP
410 / 408	55 / 53
FN	TN
21 / 23	1037 / 1039

Table 28: Confusion matrix of SMOTE strategy with or without RFECV as feature selection

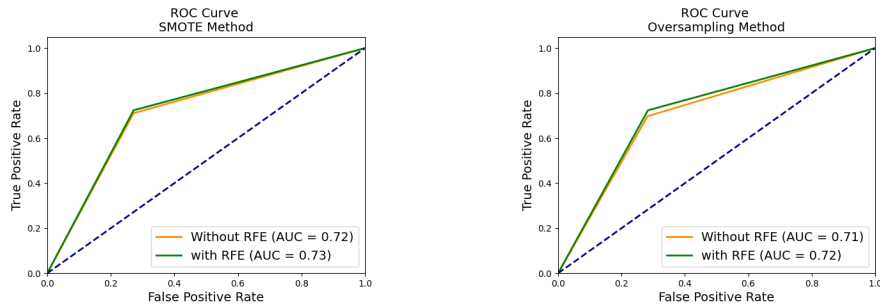
The ROC curve for the multi-year cross-sectional data is slightly better than the single-year cross-sectional data. The best AUC score is achieved by implementing SMOTE and RFECV, 0.73.

Figure 19: ROC curve of Logistic Regression for multi-year cross-sectional data



(a) ROC curve of default imbalanced dataset with or without RFECV as feature selection

(b) ROC curve of RandomUnderSampler strategy with or without RFECV as feature selection



(c) ROC curve of SMOTE strategy with or without RFECV as feature selection

(d) ROC curve of RandomOverSampler strategy with or without RFECV as feature selection

Table 29: Performance Metrics of Logistic Regression Classifier with Single-Year Cross-Sectional Data

Single Year Cross-Sectional Data					
Logistic Regression					
Evaluation Matrices		Default	RandomUnderSampler	SMOTE	RandomOverSampler
Precision	0	0.87 / 0.88	0.89 / 0.92	0.93 / 0.93	0.93 / 0.93
	1	0.44 / 0.48	0.28 / 0.23	0.26 / 0.26	0.26 / 0.26
Recall	0	0.98 / 0.98	0.86 / 0.66	0.70 / 0.70	0.69 / 0.69
	1	0.09 / 0.13	0.34 / 0.63	0.66 / 0.68	0.68 / 0.67
F1-Score (Weighted average)	0	0.92 / 0.92	0.87 / 0.77	0.80 / 0.80	0.80 / 0.79
	1	0.15 / 0.21	0.31 / 0.34	0.37 / 0.38	0.38 / 0.37

FP 9 / 11	TP 7 / 10
FN 69 / 66	TN 470 / 468

Table 30: Confusion matrix of default imbalanced dataset with or without RFECV as feature selection

FP 68 / 161	TP 26 / 48
FN 50 / 28	TN 411 / 318

Table 31: Confusion matrix of RandomUnderSampler strategy with or without RFECV as feature selection

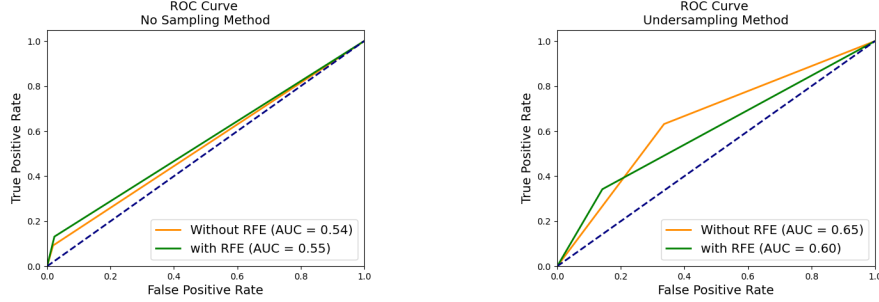
FP 142 / 145	TP 50 / 52
FN 26 / 24	TN 337 / 334

Table 32: Confusion matrix of SMOTE strategy with or without RFECV as feature selection

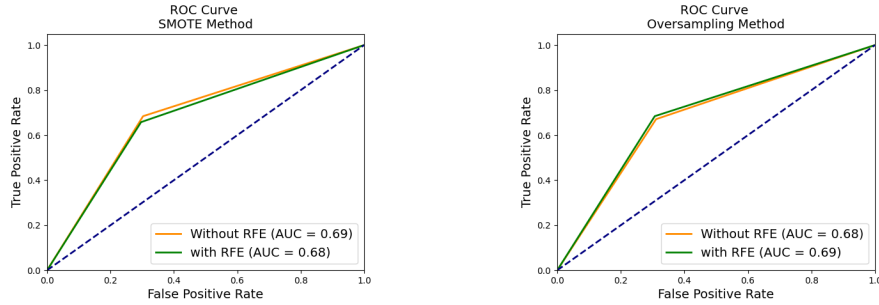
FP 147 / 149	TP 52 / 51
FN 24 / 25	TN 332 / 330

Table 33: Confusion matrix of RandomOverSampler strategy with or without RFECV as feature selection

Figure 20: ROC curve of Logistic Regression for single-year cross-sectional data



(a) ROC curve of default imbalanced dataset with or without RFECV as feature selection (b) ROC curve of RandomUnderSampler strategy with or without RFECV as feature selection



(c) ROC curve of SMOTE strategy with or without RFECV as feature selection (d) ROC curve of RandomOverSampler strategy with or without RFECV as feature selection

Table 34: The result of Stratified K-Fold Cross Validation of Different Models with multi-year cross-sectional data using the default imbalanced data and the three sampling methods. Each cell contains two values; the value to the left represents RFECV, and the value to the right represents without RFECV.

Multi-Year Cross-Sectional Data					
Model	Evaluation Matrices	Default	RandomUnderSampler	SMOTE	RandomOverSampler
Random Forest	Accuracy	0.86 / 0.86	0.79 / 0.69	0.76 / 0.81	0.84 / 0.83
	Recall	0.24 / 0.18	0.50 / 0.60	0.54 / 0.40	0.38 / 0.26
	Precision	0.46 / 0.44	0.33 / 0.25	0.29 / 0.34	0.41 / 0.36
Gradient Boosting	Accuracy	0.87 / 0.86	0.79 / 0.68	0.68 / 0.77	0.72 / 0.77
	Recall	0.13 / 0.16	0.30 / 0.65	0.50 / 0.51	0.47 / 0.49
	Precision	0.52 / 0.48	0.26 / 0.25	0.21 / 0.29	0.24 / 0.31
Logistic Regression	Accuracy	0.86 / 0.87	0.81 / 0.68	0.70 / 0.70	0.69 / 0.68
	Recall	0.09 / 0.12	0.45 / 0.65	0.65 / 0.64	0.64 / 0.64
	Precision	0.47 / 0.55	0.36 / 0.24	0.26 / 0.26	0.26 / 0.25

Table 34 and Table 35 show the result of K-fold cross-validation, the k was set to 10 folds. When comparing these results with Table 2 and Table 3, it can be concluded that the models do not overfit. As they have similar scores for accuracy, precision, and recall.

Based on the results, the best-performing model is the gradient boosting classifier with multi-year cross-sectional data, with RFECV and SMOTE implemented. It achieves the highest recall score, 0.76. However, this high recall score results in a low precision score of 0.13 and an F1-score of 0.23%. Comparing the result of this model to Bynder's baseline model, see Section 1.1, and the precision is relatively the same while their recall score is lower, 0.47%. Similarly, the AUC score is 0.68 due to the high false positive rate. The accuracy of their model is 10 percent better. However, as discussed previously, accuracy is not a good evaluation matrice for highly imbalanced data.

Table 35: The result of Stratified Group K-Fold Cross Validation of Different Models with multi-year cross-sectional data using the default imbalanced data and the three sampling methods. Each cell contains two values; the value to the left represents RFECV, and the value to the right represents without RFECV.

Multi-Year Cross-Sectional Data					
Model	Evaluation Matrices	Default	RandomUnderSampler	SMOTE	RandomOverSampler
Random Forest	Accuracy	0.95 / 0.94	0.86 / 0.83	0.91 / 0.88	0.94 / 0.93
	Recall	0.06 / 0.05	0.53 / 0.52	0.34 / 0.29	0.08 / 0.10
	Precision	0.40 / 0.13	0.19 / 0.15	0.25 / 0.15	0.27 / 0.15
Gradient Boosting	Accuracy	0.94 / 0.94	0.83 / 0.68	0.74 / 0.80	0.78 / 0.81
	Recall	0.05 / 0.03	0.49 / 0.73	0.65 / 0.56	0.56 / 0.56
	Precision	0.28 / 0.13	0.15 / 0.11	0.12 / 0.14	0.13 / 0.15
Logistic Regression	Accuracy	0.95 / 0.96	0.84 / 0.71	0.72 / 0.74	0.71 / 0.73
	Recall	0.00 / 0.01	0.48 / 0.70	0.69 / 0.69	0.69 / 0.70
	Precision	0.00 / 0.10	0.16 / 0.12	0.12 / 0.13	0.12 / 0.12

6 Discussion

In this section, the limitation of the method used in this study and possible future improvements are discussed.

6.1 Limitations

The purpose of this study was to develop a binary churn prediction model in the domain of B2B and compare the performance of different algorithms for the built model. The algorithms investigated in this study include Random Forest, Gradient Boosting Classifiers, and Logistic Regression. The effects of performing feature selection with RFECV and balancing the training dataset with SMOTE, RandomUnderSampler, and RandomOverSampler on the performance of each algorithm have been discussed.

Despite the fact that the study was carried out with extreme caution, many limitations must be mentioned. The following are critical limitations to consider:

1. It should be highlighted that each customer depending on various factors such as the industry type, seasonal patterns, and other unique circumstances, has significantly different DAM usage data. For instance, a company that performs data analysis for large sporting events might increase usage during years when significant events, such as the World

Cup, occur. In comparison, their DAM usage could decrease in the years between these events. However, the predictive model may not take these yearly cycles into account and interpret the drop in DAM usage as customer churning. This results in a higher number of false positives. Similarly, the seasonality patterns are not captured by logistic regression and random forest, while gradient boosting can inherently capture some degree of this seasonality. For example, a company in hospitality may experience higher DAM usage during the holiday seasons due to increased customer activities. Other factors influence DAM usage, such as economic fluctuations, if a key user within a customer's company leaves or the company undergoes restructuring, or project-based usage, meaning the DAM product is intensively used for a particular project, then the usage drops back to normal.

2. The churn rate could be influenced by external factors, for instance, economic climate, industry trends, or the emergence of competitive products. The predictive model developed in this study does not take these factors into consideration when predicting if a customer is churning or not.
3. The three algorithms, Logistic Regression, Random Forest, and Gradient Boosting, were not finely tuned, which can lead to suboptimal performance. The model may not achieve its full potential in predicting churn, which results in a low accuracy, precision, recall, and F1 score.
4. Using single-year Cross-Sectional Data may not represent the broader customer behavior, particularly if that year contained uncommon events.
5. Handling the outliers by using the mean or the most common value can result in a decrease in variability and influence the distribution of the data. In addition, it does not consider the possibility that the outliers may be due to changes in customer behavior or important events. Thus, this valuable information for churn prediction is lost.

6.2 Future Work

This research looks into the use of machine learning to forecast customer retention in a B2B context. In the future, it would be interesting to investigate the effect of adding additional data, such as reasons for churn and Zendesk data, which includes data on customer support.

The temporal patterns like seasonality and yearly trends could be further investigated. These temporal patterns can provide significant insights into customers' behavior and influence the churn rate. Incorporating them into the model could enhance the model's performance. For example, these temporal patterns could be added by adding new features that capture the temporal information.

In addition, the models presented in this study could be further improved by fine-tuning and exploring alternative optimization techniques. Different techniques for handling missing values could be investigated to find the most optimal technique, especially since it's important to keep valuable information that results in a better churn prediction. Similarly, different sampling strategies could be further investigated, specifically implementing techniques that include oversampling and undersampling. Also, different feature selection approaches could be examined, such as using the

feature importance provided by each model. Random Forest and Gradient Boosting inherently have these functionalities; however, for logistic regression, the magnitude of the estimated coefficients can be used, and the larger the magnitude coefficient, the more critical a feature is.

Furthermore, the model could be further generalized so it can be implemented for other SaaS companies. Determining which factors allow a model to be applied across different companies can lead to a more robust and widely applicable churn prediction model.

7 Conclusion

By analyzing the information provided in this thesis report, the research question can be answered. *How can supervised machine learning methods and customer behavior analysis be used to predict customer churn in the domain of business-to-business?* All the models perform poorly when both recall and precision are considered. However, the goal of this study was to reduce the number of cases where churn customers are classified as active customers, FN since losing customers in a B2B context is more damaging than falsely identifying active customers as churned. Taking that into consideration, the gradient boosting classifier with SMOTE and RFECV and multi-year cross-sectional data as input was able to achieve a decent recall of 0.76%.

In addition, it was observed that balancing the dataset results in a decrease in precision, which is correctly identifying churned customers. At the same time, it improves the recall score, the model's ability to classify churn correctly.

References

- a. The Netherlands: ACM's cloud services market study flags competition problems and prompts for amendments of the Data Act proposal.
 - b. `sklearn.model_selection.StratifiedGroupKFold`.
- Abhigyan. 2021. Cross-Validation Techniques.
- Addagatla, Arun. 2021. Maximum Likelihood Estimation in Logistic Regression.
- Alasadi, Suad A and Wesam S Bhaya. 2017. Review of data preprocessing techniques in data mining. *Journal of Engineering and Applied Sciences*, 12(16):4102–4107.
- Anguita, Davide, Luca Ghelardoni, Alessandro Ghio, Luca Oneto, and Sandro Ridella. 2012. The k-fold cross validation. In *ESANN*, pages 441–446.
- Ayyadevara, V Kishore and V Kishore Ayyadevara. 2018. Gradient boosting machine. *Pro machine learning algorithms: A hands-on approach to implementing algorithms in python and R*, pages 117–134.
- Biau, Gérard and Erwan Scornet. 2016. A random forest guided tour. *Test*, 25:197–227.
- Breiman, Leo. 1996. Bagging predictors. *Machine learning*, 24:123–140.
- Brys, Guy, Mia Hubert, and Anja Struyf. 2003. A comparison of some new measures of skewness. In *Developments in robust statistics: international conference on robust statistics 2001*, pages 98–113, Springer.
- Deng, Xinyang, Qi Liu, Yong Deng, and Sankaran Mahadevan. 2016. An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Information Sciences*, 340:250–261.
- Egbert, Jesse and Luke Plonsky. 2021. Bootstrapping techniques. In *A practical handbook of corpus linguistics*. Springer, pages 593–610.
- Fawcett, Tom. 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874.
- Ferreira, Pedro, Duc C Le, and Nur Zincir-Heywood. 2019. Exploring feature normalization and temporal information for machine learning based insider threat detection. In *2019 15th International Conference on Network and Service Management (CNSM)*, pages 1–7, IEEE.
- Figal, Iris, Christoph Elsner, Jan Bosch, and Helena Holmström Olsson. 2019. Customer Churn Prediction in B2B Contexts. In *Software Business, Lecture Notes in Business Information Processing*, pages 378–386, Springer International Publishing, Cham.
- Gallo, Amy. 2014. The Value of Keeping the Right Customers. *Harvard Business Review*.
- Gordini, Niccolò and Valerio Veglio. 2017. Customers churn prediction and marketing retention strategies. An application of support vector machines based on the AUC parameter-selection technique in B2B e-commerce industry. *Industrial Marketing Management*, 62:100–107.
- Hasan, Basna Mohammed Salih and Adnan Mohsin Abdulazeez. 2021. A review of principal component analysis algorithm for dimensionality reduction. *Journal of Soft Computing and Data Mining*, 2(1):20–30.
- Jiang, Tammy, Jaimie L. Gradus, and Anthony J. Rosellini. 2020. Supervised Machine Learning: A Brief Primer. *Behavior Therapy*, 51(5):675–687.
- Kaiser, Lena Katharina and Anna Federika Würthner. 2020. *Keeping SaaS business clients loyal : An exploratory multi-case study on how to design loyalty initiatives*. Ph.D. thesis.
- Khashab, Basel, Stephen Gulliver, Rami Ayoubi, and Carolyn Strong. 2022. Analysing enterprise resources for developing CRM framework in higher education institutions. *Journal of Enterprise Information Management*, 35(6):1639–1657.
- Lalwani, Praveen, Manas Kumar Mishra, Jasroop Singh Chadha, and Pratyush Sethi. 2022a. Customer churn prediction system: a machine learning approach. *Computing*, 104(2):271–294.
- Lalwani, Praveen, Manas Kumar Mishra, Jasroop Singh Chadha, and Pratyush Sethi. 2022b. Customer churn prediction system: a machine learning approach. *Computing*, pages 1–24.
- Larasati, Aisyah, Apif M Hajji, and Anik Dwiastuti. 2019. The relationship between data skewness and accuracy of artificial neural network predictive model. In *IOP Conference Series: Materials Science and Engineering*, volume 523, page 012070, IOP Publishing.
- Le, Tuong, Minh Thanh Vo, Bay Vo, Mi Young Lee, and Sung Wook Baik. 2019. A hybrid approach using oversampling technique and cost-sensitive learning for bankruptcy prediction. *Complexity*, 2019.
- Ma, Shuangge and Jian Huang. 2008. Penalized feature selection and classification in bioinformatics. *Briefings in bioinformatics*, 9(5):392–403.
- Marsland, Stephen. 2015. *Machine learning: an algorithmic perspective*. CRC press.
- Mustaqim, Adi Zaenul, Sumarni Adi, Yoga Pristyanto, and Yuli Astuti. 2021. The effect of recursive feature elimination with cross-validation (rfecv) feature selection algorithm toward

- classifier performance on credit card fraud detection. In *2021 International Conference on Artificial Intelligence and Computer Science Technology (ICAICST)*, pages 270–275, IEEE.
- Muller, Andreas C. and Sarah Guido. 2016. *Introduction to machine learning with Python: a guide for data scientists*, first edition edition. O'Reilly Media, Inc, Sebastopol, CA. OCLC: ocn895728667.
- Natekin, Alexey and Alois Knoll. 2013. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7:21.
- Ngoma, Muhammed and Peter Dithan Ntale. 2019. Word of mouth communication: A mediator of relationship marketing and customer loyalty. *Cogent Business & Management*, 6(1):1580123.
- Oza, Nikunj C. 2004. AveBoost2: Boosting for Noisy Data. In *Multiple Classifier Systems*, Lecture Notes in Computer Science, pages 31–40, Springer, Berlin, Heidelberg.
- Prusty, Sashikanta, Srikanta Patnaik, and Sujit Kumar Dash. 2022. Skcv: Stratified k-fold cross-validation on ml classifiers for predicting cervical cancer. *Frontiers in Nanotechnology*, 4:972421.
- Raschka, Sebastian and Vahid Mirjalili. 2019. *Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing Ltd.
- van Rijn, Jan N, Geoffrey Holmes, Bernhard Pfahringer, and Joaquin Vanschoren. 2018. The online performance estimation framework: heterogeneous ensemble learning for data streams. *Machine Learning*, 107:149–176.
- Sagi, Omer and Lior Rokach. 2018. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249.
- Shlens, Jonathon. 2014. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*.
- Simeone, Osvaldo. 2018. A Very Brief Introduction to Machine Learning With Applications to Communication Systems. *IEEE Transactions on Cognitive Communications and Networking*, 4(4):648–664.
- Thompson, Bob. 2004. Successful crm: Turning customer loyalty into profitability. *RighNow Technologies Publication*.
- Vafeiadis, T., K.I. Diamantaras, G. Sarigiannidis, and K.Ch. Chatzisavvas. 2015. A comparison of machine learning techniques for customer churn prediction. *Simulation Modelling Practice and Theory*, 55:1–9.
- Vanninen, Jarmo et al. 2022. Customer-facing functions in b2b saas company business model design: how vendors configure sales, marketing, and customer success.
- Vinutha, HP, B Poornima, and BM Sagar. 2018. Detection of outliers using interquartile range technique from intrusion dataset. In *Information and Decision Sciences: Proceedings of the 6th International Conference on FICTA*, pages 511–518, Springer.
- Wang, Canhua, Zhiyong Xiao, and Jianhua Wu. 2019. Functional connectivity-based classification of autism and control using svm-rfecv on rs-fmri data. *Physica Medica*, 65:99–105.
- Weiss, Gary M, Kate McCarthy, and Bibi Zabar. 2007. Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? *Dmin*, 7(35-41):24.
- Wijaya, Cornelliud Yudha. 2021. 5 Data Transformers to know from Scikit-Learn.