Master project business analytics

Machine learning predictions of customer churn from store and e-commerce data

Author:
N.D. Foree (2681569)

October 11, 2022

Supervisor:
R. Westerbeek (Sentient)

Supervisor:
dr. P.J. de Andrade Serra
Second Reader:
prof. dr. M. Hoogendoorn

Preface

The final challenge for completing the master Business Analytics at the VU University is writing a thesis during a 6 month intership at a company. The student is expected to solve a business problem or to give valuable insights regarding related questions. This graduation internship took place at Sentient Information Systems from April 2022 till October 2022.
Sentient offered me the possibility to complete my thesis, therefore I would like to thank the team of Sentient who gave me advice and helped me with my research. Especially, I want to thank my company supervisor Roos Westerbeek for her valuable input during my internship. In addition, I want to thank my VU supervisor, Paulo de Andrade Serra, for his support and guidance. At last, I want to thank Mark Hoogedoorn for being the second reader of this master project.

Abstract

One of the main challenges for companies is to predict customers' purchase behaviour. Churning customers have a negative impact on the business of companies. To counter churning customers, it is useful to flag customers who are about to churn, in this way a company can take proactive steps to try to retain the churning customers. In this report, it is evaluated which machine learning algorithms have the best predictive performance in predicting customer churn and which methods are beneficial for the predictive performance of the machine learning algorithms. From store and e-commerce data, multiple features are engineered and the relevance of the engineered features are discussed. Two different missing value strategies and two different time windows for training are applied and evaluated. Various machine learning methods for classification are applied to predict customer churn: logistic regression (LR), Naive Bayes (NB), K-nearest-neighbour (KNN), decision tree (DT), random forest (RF), support vector machines (SVM), artificial neural network (ANN) and gradient boosting (GB).

The best performing machine learning algorithm on accuracy (0.85), precision (0.773) and AUC score (0.927) is the random forest algorithm (RF). The best performing machine learning algorithm on recall (0.789) and F1 score (0.769) is the artificial neural network algorithm (ANN), where gradient boosting (GB) has an identical F1 score. The results are obtained with stratified K-fold cross-validation (K=5) and each model has their own feature selection and hyperparameter optimisation. The engineered feature that is the most beneficial for to the predictive performance of the algorithm/models is the recency (R) feature from the RFM model, however the other features from the RFM model are relevant as well. The tested extension of the RFM model: the clumpiness features and relational length feature proofs to be beneficial for the predictive performance of the ML algorithms. Other engineered features of added value are features derived from missing value categories, customer's most bought categories and the customer's most visited store features. Training algorithm/models on all the data instead of using a reduced time window is preferred. The recommended missing value strategy is for categorical features, to consider missing values as a separate category and for numeric features, if the correlation with other features is low, to apply mean/median imputation.

Contents

1 Introduction

E-commerce has grown rapidly because of the pandemic (Radcliffe, 2022). Coronavirus compelled customers to use internet and make it a habit in their daily routine (Abiad et al., 2020). Now that more and more countries are lifting covid restrictions; life slowly returns back to life before the pandemic. The loosening of covid restrictions is bad news for e-commerce companies, people are not forced anymore due to lifting restrictions restrictions to buy their products online. Maintaining customers is the most basic and the most important issue for commercial companies. Customer churn can be unavoidable for e-commerce companies as a result of the global trend of lifting covid regulations. Customer churn is defined as the propensity of customers to stop doing business with a company in a given period (Yu et al., 2011). In this report is assumed that a customer who has not bought a product for a year, is a churned customer. It is important for companies to retain as many customers as possible because customers are one of the most important assets for any firm (Gupta & Lehmann, 2003).
Releasing covid regulations will make competition in e-commerce more fierce, due to the expectation that e-commerce demand will decline. This will make customer retention become increasingly important for e-commerce companies. It is widely known in research and in the business domain that customer retention is much less costly than to acquire a new customer (Dubrovski, 2001; Reichheld et al., 2000; Reichheld & Sasser, 1990). Across a range of industries, increasing customer retention by as little as 5% could result in long-run profit increases between 25% and 95% (Reichheld et al., 2000; Reichheld & Sasser, 1990).
Customer relationship management (CRM) is therefore important for e-commerce companies. It is a challenge for e-commerce companies to predict customers' purchase behaviour (Renjith, 2015). Datamining plays an important role in customer relationship management (Kadiyala, Srivastava, et al., 2002). Data mining is defined by Turban et al. (2007) as "the process that uses statistical, mathematical, artificial intelligence and machine-learning techniques to extract and identify useful information and subsequently gain knowledge from large databases". One of the goals of CRM is to retain customers, an important first step is to classify customer with datamining techniques that have a high probability of churn. With this classification of churning customers, the companies can choose to be proactive: take their best effort to retain these classified customers. Many studies in the datamining domain have shown that customer churn can be predicted successfully with machine learning algorithms (Buckinx & Van den Poel, 2005; Gregory, 2018; Yu et al., 2011).

1.1 Problem statement

Customer churn has a negative impact on the business of companies. It is important to classify customers who run the risk of churn ahead of time, that way a company can make the best effort to retain the classified customers. It is widely known in business that it is easier and less expensive to retain customers than to acquire a new customer (Dubrovski, 2001; Reichheld et al., 2000; Reichheld & Sasser, 1990)

1.2 Information about host company

Sentient Information Systems B.V is a provider of cutting-edge technologies and services, offering scalable solutions for companies of all sizes. Sentient does a lot of projects in different sectors. Now Sentient want to improving their RFM software, by improving their predictive capabilities. Sentient wants to attract new customers with their RFM software. Their software includes a dashboard, this makes it easy for the clients to make analysis's. With their software they segment customers according the RFM model, to make it easy for employees of e-commerce companies to analyses their clientele. For example, they can analyse who their top customers are and what they have in common. They want to separate their self from other competitors in providing software which makes good segmentation, is easy in making analysis and makes good demonstrable predictions.

1.3 Objective

In this report the author has stated 2 main research questions:

1. Which machine learning algorithms have the best predictive performance in predicting customer churn?

2. What kind of methods improve the performance of machine learning algorithms in predicting customer churn?

More specifically for main research question 2, the author has developed 3 more specific sub questions:

- Which features in the dataset contribute to predictive performance of the machine learning algorithms?

- Which features can be engineered and improves the predictive performance of the machine learning algorithms?

- What is a missing value strategy that adds value to the predictive performance of the machine learning algorithms?

1.4 Structure of report

This report contains 7 sections and 6 sections in the appendix. The first section is the introduction section, first the main topic customer churn is introduced and described why it is important for business. In the same section is described: the problem statement, information about the host company, the objectives and the structure of the report. In the following section, the literature review section, relevant literature about predicting customer churn is discussed: customer relationship management (CRM), datamining, literature about the RFM model and extensions of the RFM model. In the following section, the data section, the data is described and from the dataset multiple features features are engineered. In the feature engineering section, the classification feature churn is engineered, which is the response variable. In the same section a univariate analysis and a multivariate analysis is executed. In the univariate analysis, 2 missing value strategies are proposed, the missing value strategies will be evaluated in the result section. Next in the method section, is described what a binary classification problem is, multiple performance evaluations for a classification problem are described and validation methods are discussed. At last, in the method section, 8 machine learning methods are described to arrive at a classification prediction. In the result section, first the results of the performance evaluation for all the algorithms/models, the feature selection and chosen hyperparameters are presented. Next in this section, the results of the performance of a time window change for training purposes are described and the result are presented for the robustness analysis the RFM extensions, the engineered features clumpiness (C) and relational length (L). Finally, in the result section, the result of the alternative missing value strategy is presented. In the last section, in the discussion section, is discussed: the results, some comments about not applied techniques, limitations, recommendations and suggestions for future work.

2 Literature review

In this section, the relevant literature on customer churn is discussed and the RFM model is discussed. First, the well-known topic in this domain called customer relation management (CRM) is discussed. The relevance of this topic and their 4 dimensions are discussed. Next, it's explained why the identification of customers is useful. Furthermore, it's discussed what kind of techniques one can use to analyse customer behaviour. At last, a widely used model for CRM, the RFM model and some extensions of the RFM model are discussed.

2.1 Customer relationship management (CRM)

Customer relationship management is a broad concept and refers to how a business manages its relationships with customers and potential customers. Customer relationship management (CRM) is a combination of people, processes and technology that seeks to understand a company's customers. Customer relationship management is a comprehensive approach that promises to maximize relationships with all customers (Chen & Popovich, 2003). Kracklauer et al. (2004), categorizes CRM on four dimensions: customer identification (1), customer attraction (2), customer retention (3), customer development (4).

1. Customer identification: The first step of CRM is customer identification; this includes customer segmentation and target customer analysis. Customer segmentation implies the segmentation of all customers into

smaller segments including customers with similar characteristics (Woo et al., 2005). Target customer analysis involves the identification of the most attractive customer groups for the company based on the customer characteristics of the group. Identification of customers that are risking to churn is part of this dimension as well.

2. Customer attraction: This dimension follows customer identification. Companies concentrate effort and allocate resources to attract the identified target groups. These customers can be attracted by direct market, or competitive advantages, such as price and other differentiation characteristics.

3. Customer retention: Customer satisfaction, which refers to the comparison of customers' expectations with his or her perception of being satisfied, is the essential condition for retaining customers (Kracklauer et al., 2004). Elements of customer retention include complaints management, loyalty programs and one-to-one marketing. One-to-one marketing refers to personalized marketing campaigns which are supported by analysing, detecting and predicting changes in customer behaviour. Loyalty programs involve campaigns or supporting activities which aim at maintaining a long-term relationship with customers.

4. Customer development: The focus of this dimension is to increase individual customer profitability, transaction value and transaction intensity. With customer development, the company tries to increase the customer value for the company with up/cross selling. Companies try to migrate customers to their most profitable customer segment.

This report focuses mainly on customer identification. Basically, identifying customers, customers that are going/risking to churn for the retention campaign.

2.2 Churn management

As part of the CRM strategy churn management is the art of identifying the valuable customers who are likely to churn and executing proactive steps to retain those customers. According to Lejeune (2001): "Churn management consist of developing the techniques that enable firms to keep their profitable customers and it aims at increasing customer loyalty". Loyal customers are defined by Lejeune (2001) as customers that "remain client of their original supplier even if a competitor proposes more advantageous conditions." An approach of customer churn management is proactive management, first predict the customer who might churn with datamining techniques and then use a strategy to prevent it.

2.3 Datamining

One of the import techniques used for customer relationship management is data mining. Chen and Popovich (2003) state that "Getting to 'know' each customer through data mining techniques and a customer-centric business strategy helps the organization to proactively and consistently offer (and sell) more products and services for improved customer retention and loyalty over longer periods of time".

Data mining is the process that uses a variety of data analysis and modelling techniques to discover patterns and relationships in data that may be used to make accurate predictions (Lakshmi & Raghunandhan, 2011). Khajvand et al. (2011) describe datamining as two categories: descriptive and predictive. Where clustering is a descriptive method and classification is a prescriptive method. Classification is the process of finding a model that describes and distinguish class labels, for the purpose of being able to use the model to predict the class of object whose class label is unknown (Han et al., 2011). Where they define data classification as a two-step process, a learning step, where the model is constructed and a classification step, where the model is used to predict the class label for the given data. They define clustering as the process of portioning a set of data objects into subset. Where each subset is a cluster, such that objects in a cluster are similar to one another and dissimilar to objects in other clusters. CRM and datamining offer foundations of the development of a competitive marketing strategy by analysing and understanding customer behaviours and characteristics, to gain and maintain potential customers and maximize customer value (Tsai, 2011). It is important to use the proper data mining tool to extract or generate effective information out of the customer database.

2.4 RFM model

The RFM model is already being used for about 30 years and still remains useful for business today. The RFM analytic model is proposed by Hughes (1994), it is a model that differentiates customers from data by 3 variables: interval of customer consumption, frequency and monetary amount. The definitions of RFM model are described as follows:

1. Recency of the last purchase (R). R represents recency, which refers to the interval between the time that the latest consuming behaviour happens and the present.

2. Frequency of the purchases (F). F represents frequency, which refers to the number of transactions in a particular period.

3. Monetary value of the purchases (M). M represents monetary, which refers to consumption in money terms in a particular period.

When R, F and M is determined for each customer, the customers are segmented, based on similar characteristics of the 3 classes: R, F and M. The RFM model has some extension such as LRFM model. Amine et al. (2015) extended the RFM model to the LRFM model: where the variable length is added to the model, length measures the time between the first and last visit/purchase of the customers. The RFM model is extended to the LRFM model because the

RFM model cannot discuss between long-life customers and short-life customers (Reinartz & Kumar, 2000). Or the extension RFMC where clumpiness is added to the model. Zhang et al. (2013) define clumpiness as the degree of nonconformity to equal spacing. According to Zhang et al. (2015) clumpiness adds to the predictive power, above and beyond RFM and firm marketing action, of both the churn, incidence, and monetary value.

3 Data

In this section, first general information about the datasets is discussed, then from the dataset multiple features are engineered, this is discussed in the feature engineering section. In this section is discussed as well how the classification feature is engineered. In the next section a univariate analysis is executed. In this section, transformations are executed and 2 missing values strategies are proposed. In the last section a multivariate analysis is executed: first a visualisation of the features with response variable churn, next a visualisation of the features with the highest mutual information gain with the response. At last, in the multivariate analysis section is information about the correlation between the features and a test of collinearity is executed. Unfortunately the last part is confidential.

3.1 Data description and exploration

The data consist of three datasets: customer dataset, order dataset and item dataset. In the customer dataset is information about the customer, the dataset contains 20000+ rows, where each row represents a customer. In the order dataset is an order id, customer id and information about the time and date of the order, there are 300000+ orders in this dataset. The order id can be linked to the orderitem dataset, in this way one could retrieve which items are bought in that specific order. The data covers a period of approximately 3 years. There are 20 percent of the customers labelled as not active, customers who churn, and 80 percent labelled as active, customers who do not churn. The variable active has a boolean value and indicates if a customer made a purchase in the last year of the datset.

3.2 Feature engineering

From the order dataset combined with item dataset multiple features can be engineered. First, in this section the classification feature is engineered, the class for each customer, churn and no churn which in the end we want to predict. Next, the RFM features are generated where the RFM model is extended with two features, *Clumpiness* and *RelationalLength*. In the next section more features are engineered about the frequency of returned items, the most often bought categories and the store where the customer bought the most.

3.2.1 Classification feature

In this report the data contains store and e-commerce data, clients can always come back to the company to make a purchase. This is different in a contractual setting, where it is clear when a customer churns, the customer terminates or does not extend the contract. That is way, in this case, an assumption have to be made to define a customer who churns. Together with the company supervisor a customer who churns is defined as a customer who has not made a purchase in 1 year. First, the last year of the data is used to classify customers as churn (1) or no churn (0) for the feature *Churn*. The opposite of this feature already existed in the provided dataset and is called active, the feature active indicates if a customer made a purchase in the last year. So, if active is 0, it indicates that the customer has churned, churn is 1, and if active is 1, it indicates that the customer did not churn, churn is 0.
In order to make useful predictions and to let the generated features make sense, from now on, the discussed data will be without the transaction data of the last year. To clarify with an example that some features wouldn't make sense, we take the generated feature *Recency*, if a correlation analysis with the response is made, then if *Recency* has a value greater than 365, this will directly indicate that a customer has churned. Therefore, is chosen to use for the further data analysis, a subset of the data which not includes the last year of the dataset. The last year is used to classify customers, so customers who have made a purchase only during the last year are removed from the dataset. No info is known for these customers because the last year is used for classification only. This results in 15490 customers in the dataset where 5000 are labelled as churn and 10490 are labelled as no churn.

3.2.2 RFM

From the order dataset combined with item dataset, multiple features can be engineered. The RFM features RFM stands for *Recency* (R), *Frequency* (F) and *Monetary* (M). *Recency* (R) is generated by counting the days between the last transaction and the last date known in the dataset. The *Frequency* (F) feature is generated by counting the number of purchases of a customer in the observed time period. The *Monetary* (M) feature is generated by adding the price of all the purchased items in the observed time period. Alternatively, one could take monetary average instead of monetary value, monetary average is monetary value divided by the purchase frequency of the customer, *Monetary_average* is used as a feature as well and analysed in this report.

3.2.3 Extensions RFM

The feature relationship length (L) is added as an extension to the RFM model. According to the findings of Anderson and Weitz (1989) older relations appear to be more stable than younger relations in a business context. Following on this finding, Buckinx and Van den Poel (2005) found that the length of relationship is an important predictor in their churn model based on non-contractual retail data. The *RelationalLength* feature is defined as the days in between the registration date or first purchase date and last purchase date. The feature *Clumpiness* (C) is another extension to the RFM model. According to Zhang et al. (2013), clumpiness is defined as the degree of nonconformity to equal spacing. The central properties of clumpiness are:

- Minimum: the measure should be at its minimum if the events are equally spaced.
- Maximum: the measure should be at its maximum if all the events are gathered together.
- Continuity: shifting event times by a small amount should only change the measure by a small amount.
- Convergence: as events move closer, the measure should increase. When the events move further apart, the measure should decrease.

Zhang et al. (2015) developed the following formula which follows the central properties:

$$h_p = 1 + \frac{\sum_{i=1}^{n+1} log(x_i) * x_i}{log(n+1)}$$

Where h_p is the clumpiness measure score for customer p, n is the amount of purchases for customer p and x_i is the inter event times (IET). The inter event times is the timesteps between the purchases, where the first IET is the timesteps between start date of the observed time period and the first purchase date. The last IET is the timesteps between the last purchase and the end time of the observed period plus 1. There have been tested different timesteps, daily, weekly or monthly, to calculate the IET. Zhang et al. (2015) note that: "if the selected time unit is too long, data points are aggregated and information about the IETs would be lost. On the other hand, a too short time unit would give rise to possibly over-fit patterns. Hence, the choice of t in practice should be matched to the scaling of a business's decision time periodicity." It is decided to use a weekly timestep for the IET computation, due to the low frequency of purchases in the dataset and the small difference of the values of the *Clumpiness* feature by using a weekly or daily timestep. Using a weekly timestep instead of a daily timestep has the advantages that the observe time period gets smaller, this makes to computation of the clumpiness measure score less time consuming. With the *Clumpiness* feature we can generate another feature, *ClumpinessCat* feature which indicate if a customer is clumpy or not clumpy based on their clumpiness measure score. To create this feature first a table of critical values is generated, with a Monte Carlo simulation with random sampling without replacement. To create the table of critical values from the Monte Carlo simulation we used a significance level of 0.95. The input of the table of critical values is the number of purchases for each customer. The clumpiness measure value is compared with the corresponding critical value of the purchase frequency n of

the customer, if the clumpiness measure score is greater than the critical value, the customer is classified as clumpy (1), if this is not the case the customer is classified as non-clumpy (0).
In table 1 is shown that there are many customers in the dataset with a low frequency of purchases, a purchase frequency lower than 3, for these customers the clumpiness measure score is not well defined. For the customers who only make one purchase, the clumpiness score will be high or low based on the date of the purchase, the closer the date of purchase is to the last date in the dataset, the higher the score will be. Similar for the customers with a purchase frequency of 2, the date of the last purchase has more effect on the clumpiness measure score than the inter event time between the 2 purchases. The paper of Zhang et al. (2015) lacks information about what to do when the frequency of events is small. Rao (2015) made a similar comment when reviewing the paper from Zhang et al. (2015), Rao (2015) stated that whether the clumpiness measure score is well defined if the number of events is small. After investigation of the clumpiness measure score for customers with a low frequency of purchases, it is decided that for the customers with a purchase frequency lower than 3, the clumpiness measure score is not valid, therefore these customers do not get a value for the *Clumpiness* feature. For those customers the *Clumpiness* feature will be imputed with a missing value strategy, stated in the missing value strategies section. In total, for the whole dataset, 45.9 percent of the total amount of customersthere have missing values for the *Clumpiness* feature. In total, for the whole dataset, there 38.1 percent is classified by the *ClumpinessCat* feature as not clumpy and 15.9 percent as clumpy by their clumpiness measure score. It is decided to classify the customers without a clumpiness measure value, as non-clumpy based on their low frequency of purchases. In total 84.1 percent of all the customers are classified as non-clumpy for the feature *ClumpinessCat*.

3.2.4 Other engineered features

Another generated feature is the frequency of returned items, as the feature *Neg_Frequency*. The returns are not included in the calculation of the *Monetary* feature due to inconsistency in the data, it occurs that items are returned, but in their purchasing history the item is not presented. Probably because those items are not bought with that particular customer id but with another id. Due to this inaccuracy, returned item are not included in the calculation of the *Monetary* feature.
More features are generated from the item dataset. The feature *StoreName*, where the customer bought most often their items and the customer's most purchased item category, for category 1, as *Cat1* and category 2, as *Cat2*. The customer most bought product categories and store where the customer bought the most could be possible predictors of customers' churn.

3.3 Univariate analysis

In this this section, a univariate analysis is executed, each variable is explored separately. In table 1 the descriptive statistics of the numeric features are shown. An indication of this table is that most of the generated features are positively skewed.

	mean	std	min	Q1	Q2	Q3	max
Recency	102.81	135.85	1	22	54	130	844
Monetary	1222.63	2588.24	1.5	180.84	480.88	1244.09	72060.19
Monetary_average	108.67	95.40	1.5	54.18	86.49	134.53	2192.6
Frequency	11.76	21.50	1	2	5	13	947
Neg_Frequency	2.09	7.65	0	0	0	2	403
RelationalLength	1262.36	1353.85	0	405	932	1708	6116
Clumpiness	0.21	0.13	0	0.11	0.17	0.26	0.87
Storedistance	19.43	27.50	0	3	9	22	174.43

Table 1: Descriptive statistics for the numeric features.

3.3.1 Data transformation

Table 1 indicates that monetary value has a lot of outliers. Removing the outliers will result in losing customer data, therefore a log 10 transformation is applied to reduce the variance of the *Monetary* feature. The distribution of *Monetary* after the log transformation is shown in figure 1. Figure 1 could indicate that *Monetary* is normally distributed after the log transformation. Although after performing Shapiro-Wilk test, the null hypothesis: the data is normally distributed, is rejected therefore *Monetary* is after the log transformation not normally distributed. However, the transformation reduces the variance of the *Monetary* and therefore the transformation is useful. Similar for *Monetary_average* the log transformation reduces the variance, however when performing the Shapiro-Wilk test, likewise the null hypothesis is rejected. All the numeric features are transformed by the following formula:

$$Xnew = (X - mean(X))/sd(X)$$

This is called standardisation in this way each numeric feature has a mean of 0 and a standard deviation of 1, this makes sure that a variable on a larger scale does not have an overloaded influence. For the machine learning algorithms we need to convert categorical features in to numbers. In order to use the categorical features, dummy variables are created for the categorical values. Each categorical value gets a specific column where 1 indicate if a customer belongs to this category and 0 if a customer belongs not to this category.

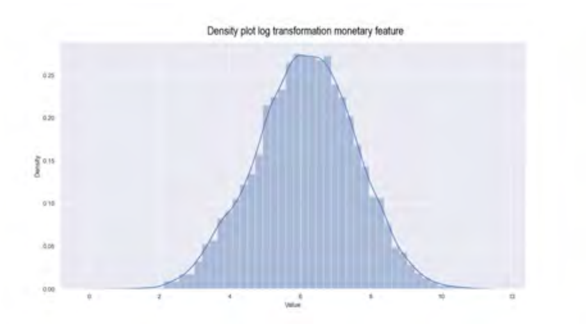


Figure 1: Density plot for the *Monetary* feature with log 10 transformation.

3.3.2 Missing value strategies

Some features have missing values, these values have to be processed for most of the machine learning algorithms, especially for machine learning algorithms that cannot cope with missing values. For the categorical values this can easily be solved by creating a new category, for the categorical values that are missing. It is not useful to try to impute the missing categorical values, the missing categorical values contain more information about customer churn then the categorical values. This can be viewed in the appendix in table **??** and **??**. However the numeric values have to be imputed with a missing value strategy, in this section, 2 different strategies are applied and tested.

Strategy 1: mean/median imputation
For the numeric features with missing values, for *Storedistance* and *Clumpiness*, a missing value strategy have to be implied. For the customers who have missing values for *Storedistance*, the *zip* feature has missing values as well, therefore it is not possible to use zip code to measure the distance to the closest store. The feature has outliers as shown in table 1, therefore it is decided to first impute the missing values, 2186 values, with the median, which is 9. For the customers who have a low frequency of purchases, lower than 3, the clumpiness measure score is not well defined as stated in the feature engineering section. It is decided to impute the values with a missing values strategy for the customers where the clumpiness measure is not well defined. In total, there are in the reduced dataset as stated in the classification feature section, 4979 missing values that have to be imputed for the *Clumpiness* feature. It is decided to impute these values with the mean of the *Clumpiness* feature which is 0.2067, for the customers who have a purchase frequency lower than 3.

Other missing value techniques
More advanced techniques to impute missing values could be interesting to evaluate, like nearest neighbour and regression imputation, however to use these techniques there have to be correlation with other features. The *ClumpinessCat* and *Frequency* feature have a correlation with *Clumpiness* feature of respec-

tively 0.61 and -0.30. The *ClumpinessCat* feature is not a useful correlated feature. Mainly because the classification of *ClumpinessCat* is based on the *Clumpiness* feature, which is for these low purchase frequency customers not valid and therefore missing. Beside those correlated features for the *Clumpiness* feature, the strongest correlation with the *Clumpiness* feature is the feature *Monetary* with a correlation coefficient of -0.34, which is a weak correlation. The strongest correlation with *Storedistance* in the dataset is 0.2, which is a weak correlation as well. Based on the fact that similar customers have missing value for the *Clumpiness* feature and the weak correlation with other features for the *Clumpiness* and *Storedistance* features, it is decided to not impute the missing values by a technique, which uses other features to impute the missing values. However, imputing missing values by the mean or median, result in less variance of the distribution of those imputed features, therefore another imputation technique is applied. This technique is compared with the mean, median imputation technique.

Strategy 2: Sample from distribution
The other imputation technique works as follows: first, a known distribution is fitted to the distribution of the numeric feature that has missing values. The distribution with the best fit, based on mean squared error (MSE) will be used to sample from. The sampled values which are used to imputed the missing values, will be in the min and max range of the feature. In this way the distribution and the variance of the feature with missing values will be disrupted at a minimum. The mean/median imputation disrupts the distribution and the variance of the feature because all the missing values get the same value, the mean or median. Both missing value strategies are evaluated and compared in the results and conclusion section.
The numeric features which we have to impute are *Storedistance* and *Clumpiness*. The distribution of *Storedistance* and the best fitted distribution is shown in figure 2. For the feature *Storedistance* a lognormal distribution with a shape, loc and scale parameter of respectively 1.28, -0.45 and 9.42 is the best fit, which result in a mean squared error (MSE) of approximately 0.002. The MSE indicates how well the fit is between the fitted distribution and the actual distribution. The MSE is calculated by $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y_i})^2$. Basically, the observed value Y_i, the values of the feature, is subtracted from the predicted value $\hat{Y_i}$ by the probability density function of the fitted distribution. The outcome of the subtraction for every observed and predicted value is squared, at last, the average of all the observations is determined, this is the MSE. Random numbers are generated for the missing values of the *Storedistance* feature from the fitted lognormal distribution, these numbers are bound to 0 and the maximum value of the *Storedistance* feature.
For the *Clumpiness* feature the best fit is a moyal distribution (Cordeiro et al., 2012) with parameters loc parameter of approximately 0.13 and a scale parameter of approximately 0.06, which result in a mean squared error of approximately 6.17. The distribution of the *Clumpiness* feature and the best fitted distribution, the moyal distribution, is shown in figure 3. The missing values of the *Clumpi-*

ness feature are likewise imputed by random numbers generated from the moyal distribution, however these numbers are bounded by 0 and the critical value for a frequency of purchases with value 2. If the random number exceeds this range than the value is the closest boundary value because the customers with a low frequency of purchase, smaller than 3, are classified as not clumpy as stated in the feature engineering section. This causes that the value cannot exceed the critical value for a purchase frequency of 2.

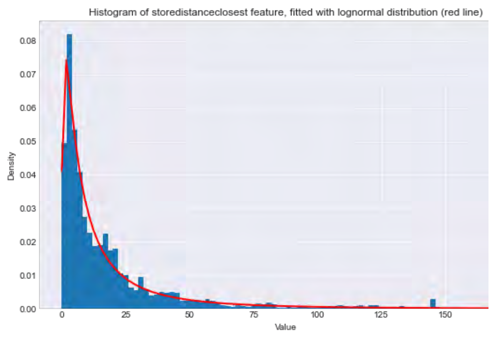


Figure 2: Histogram of the distribution of the feature *Storedistance*, with lognormal distribution as line fitted to the distribution of the feature *Storedistance*.

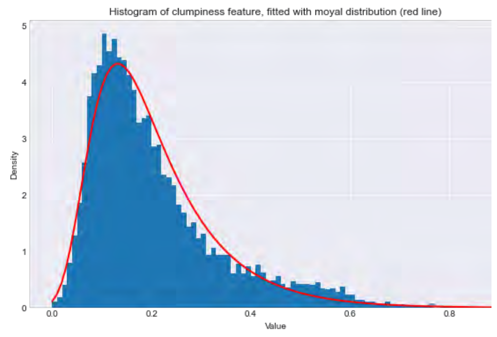


Figure 3: Histogram of the distribution of the feature *Clumpiness*, with moyal distribution as line fitted to the distribution of the feature *Clumpiness*.

3.4 Multivariate analysis

CONFIDENTIAL

4 Methods

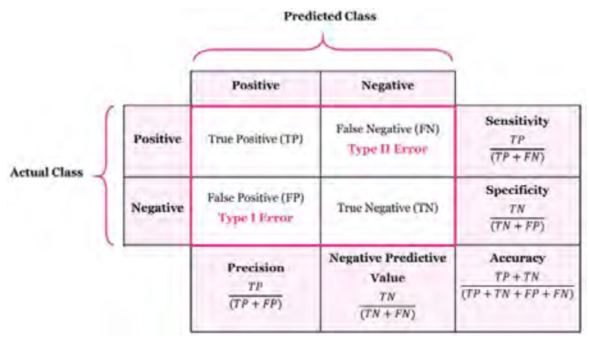
In this section, first is described what a binary classification is. Then, multiple performance evaluations methods are discussed for a classification model. Next, methods for validation are discussed. At last, different machine learning methods are described which will be used to predict the classification of customers. The section contains the following machine learning methods: LR, NB, KNN, DT, RF, SVM, ANN and GB.

4.1 Binary classification problem

In this report we want to predict if a customer churns or not. Churn is indicated by a 0 for no churn and 1 for churn, note this is a binary classification problem. To predict this classification, machine learning algorithms are used. Machine learning algorithms are methods that automatically create models from data. For churn prediction we use a binary classification algorithm. A binary classification algorithm is a supervised learning algorithm that categorizes new observations into 2 classes. It is a supervised learning algorithm because the customers are first labelled as churn or no churn, the algorithm use this label in combination with the features to train the model. After training the model, new observations are used to let the model predict the classification of the customer. The actual label, if available, can be used to evaluate the predictive performance of the model.

4.2 Performance evaluation

Confusion matrix, receiver operating characteristic (ROC) and area under the curve (AUC) are the most common evaluate methods to evaluate a classification model. From the confusion matrix other important evaluation scores can be derived, the accuracy, precision, F1 and recall score. In figure 4 the confusion matrix is shown with the performance metrics that can be derived from the confusion matrix.



		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) Type II Error	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Figure 4: Confusion matrix and the derived performance metrics from the confusion metric

Confusion matrix
The confusion matrix shows the correct and incorrect predictions of each class.

- True positives (TN): the model predicts a customer as churn and the customer actually churned.
- False positives (FP): the model predicts a customer as churn but the customer did not churn.
- True negatives (TN): the model predicts a customer as no churn and the customer did not churn.
- False negatives (FN): the model predicts the customer as no churn but actually the customer churned.
- The accuracy score: $\frac{TP+TN}{TP+TN+FN+FP}$.
- The precision score: $\frac{TP}{TP+FP}$.
- Recall also known as sensitivity: $\frac{TP}{TP+FN}$.
- The F1 score is the weighted average of precision and recall. This is especially a good performance metric if recall and precision are equally important. The following formula is used to determine the F1 score:

$$F1 = 2 * (Precision * Recall)/(Precision + Recall)$$

In the case of classifying a customer who churns or does not churn, recall is a more important metric than precision. Classifying a customer as churn when the customers is not churning (FP) is less undesirable than classifying a customer as no churn when the customers actually churned (FN). If a customer is classified incorrectly as a customer who not churns, the company cannot try to retain the customers and will therefore lose the customer. If the model classifies the customer as churn incorrectly, this will be less undesirable, the company will try to retain the customer but actually the customer was not intending to leave. This can cope with extra cost (e.g., labour cost, promotions etc.), however this cost is likely less for the company than losing a customer.

Receiver operating characteristic (ROC) and area under the curve (AUC)
Receiver operating characteristic curve, is a graphical plot that illustrates a trade-off between sensitivity and the false positive rate with a set of different thresholds. The sensitivity rate is on the y-axis and false positive rate is on the x-axis.
The false positive rate (FPR) is determined by the formula:

$$FPR = FP/(FP + TN)$$

The sensitivity or true positive rate (TPR) is determined by

$$TPR = FP/(FP + TN)$$

The curve can be interpreted as follows, the greater the gap between the diagonal and the curve, the better the performance of the model. If the curve is closer to the diagonal, this indicates a worse performance. Area under the curve as it names indicates, it measures the area underneath the ROC curve. The AUC score indicates the ability of the classification model to distinguish between the classes and is used as a summary from the ROC curve. The AUC score is a score between 0 and 1, where 1 indicates that the model can perfectly predict customers who churn and who do not churn. The AUC score will be used in this report.

4.3 Validation

4.3.1 K-fold cross-validation

The K-fold cross-validation technique is one of the most used approaches for model selection and error estimation of classifiers (Anguita et al., 2012). Cross-validation is a resample method used to evaluate machine learning models on a limited sample. K-fold cross-validation involves splitting the data into K equal sized subsamples. The machine learning algorithm is trained on each subsample and the performance of the fitted model is evaluated on the remains data. The benefit of using K-fold cross-validation is that all the data is used for training and validation, each datapoint is used for validation exactly once. The overall performance can be determined by taking the mean of the K performance evaluations.

- The data is split into independent K-folds, randomly.
- K-1 folds are used for training the algorithm/model and one fold is used for validation.
- This process is repeated K times, so there are K number of performance evaluation.
- The mean of K performance evaluation is used to estimate the overall performance of the machine learning technique.

4.3.2 Stratified K-fold cross-validation

Instead of randomly creating sub samples as with K-fold cross-validation, stratified K-fold cross-validation takes the distribution of the classification in the dataset in to account. It creates subsamples where the balance of the classification is the same as in the dataset. Therefore, when the class you want to predict is imbalanced, stratified K-fold cross-validation is preferred.

4.4 Methods

4.4.1 Logistic regression (LR)

Main idea
Logistic regression works very similar to linear regression, but with a binary response variable (Sperandei, 2014). Logistic regression uses a logistic function called a sigmoid function to map predictions as probabilities. The sigmoid function refers to an S-shaped curve that converts a value to a range between 0 and 1. A threshold value is used to classify the prediction as 0 or 1.

Assumptions

1. The response variable is binary.
2. The observations are independent.
3. There is no multicollinearity among explanatory variables.
4. There are no extreme outliers.
5. There is a linear relationship between explanatory variables and the logit of the response variable.
6. The sample size is sufficiently large.

Formulas
The sigmoid function is referred to as an activation function for logistic regression and is defined as (Hosmer Jr et al., 2013) : $f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$. Where x is in the form $b_0 + b_1X_1 + .. + b_nX_n$. This result in the following equation:

$$f(x) = \frac{\pi}{1-\pi} = \frac{1}{1+e^{-(b_0+b_1X_1+..+b_nX_n)}} = \frac{e^{b_0+b_1X_1+..+b_nX_n}}{1+e^{b_0+b_1X_1+..+b_nX_n}}$$

Where π indicates the probability of an event (e.g.,churn), b_0 is the intercept and β_n are the regression coefficients associated with the predictors X_n. The outcome of $f(x)$ is the prediction, which is a probability, then a threshold value is used to classify the prediction as 0 or 1.
The following equation (Sperandei, 2014) is used to fit the parameters of a logistic regression :

$$ln(\frac{\pi}{1-\pi}) = ln(\frac{e^{b_0+b_1X_1+..+b_nX_n}}{1+e^{b_0+b_1X_1+..+b_nX_n}}) = \beta_0 + \beta_1X_1 + .. + \beta_nX_n$$

Fitting
In the equations above $\frac{\pi}{1-\pi}$ is called the odds ratio, where π indicates the probability of an event. The ln of this odds ratio, known as the log odds, is equal to regression coefficients. This results in $ln(\frac{\pi}{1-\pi}) = \beta_0 + \beta_1X_1 + .. + \beta_nX_n$. Then similar as regression usually a maximum likelihood estimator is used to estimate the parameters.

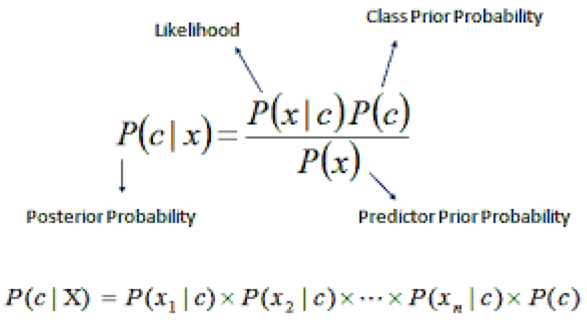
4.4.2 Naïve Bayes (NB)

Naïve Bayes is a classification technique based on Bayes theorem (Berrar, 2018). It is a probabilistic algorithm that returns a probability. It calculates the probability by count, how often each variable's distinct values occur for each class, and the prior probability of each class, which is a count up of all the instances of each class and divide by the total of all instances. These probabilities are used to predict the class of the target variable, a threshold is used to determine the class of the prediction.

Assumptions

1. All features are independent (conditional independence).
2. (Gaussian) Naive Bayes assumes that continuous values associated with each class are distributed according to a normal (Gaussian) distribution.

Formulas
In figure 5 the formula of Bayes theorem is represented (Rish et al., 2001).



$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Figure 5: Bayes theorem formula

- $P(c|x)$ is the posterior probability of class, c is the target, given predictor x.
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

Fitting
Each probability of the Bayes formula is determined, then the Bayes formula is used to determine $P(c|x)$ for each predictor (Berrar, 2018). For new observation if the predictor is present, the probability of the class given predictor x is multiplied to the other predictor present in the observation. For continuous values,

there are 3 approaches, the normal method (parametric approach), the kernel method (non parametric approach) and discretization (Bouckaert, 2004). By the discretized approach, the continuous variables are set in to bins, these bins are then treated as categorical values. The kernel method approximates $P(c \mid x)$ for a continuous variable c by a sum of so called kernels, which are functions centered around data points. The Gaussian Naive Bayes fits the continuous variables to a Gaussian distribution (normal distribution) by estimating the mean and standard deviation, usually with the maximum likelihood (Williams & Barber, 1998). Then the probability of the input variable is calculated using the probability density function.

4.4.3 K-nearest-neighbour (KNN)

Main idea
K-nearest-neighbour (KNN) classification is one of the simplest classification methods. KNN should be one of the first choices for a classification study when there is little knowledge about the distribution of the data (Peterson, 2009). KNN is called a non-parametric supervised learning method, basically KNN takes the K-nearest-neighbours of a new datapoint in to account. If the most K neighbours are labelled as churn, the KNN predicts the new datapoint as churn. K is a positive integer, the nearest neighbours are determined based on the distance between the datapoints, where Euclidean distance metric is used the most.

Assumption
KNN assumes the closer two given points are to each other, the more related and similar they are.

Formulas
The formula for the Euclidean distance is (Peterson, 2009):

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

Fitting
KNN is called a lazy learner algorithm because it does not learn from the training set immediately, instead it stores the dataset and at the time of classification, it performs an action on the dataset (Mulak & Talhar, 2015). So, no algorithms/models have to be fit to the data. Only the K-nearest neighbours have to be determined, by searching the K minimum distance values from the observation, using the Euclidean distance or another distance measure.

4.4.4 Decision tree (DT)

Main idea

Decision trees (DT) are among the most often used supervised machine learning algorithms, primarily due to their straightforward interpretation (Mrva et al., 2019). A decision tree consists of a root node, branch nodes and leaf nodes (Safavian & Landgrebe, 1991). The root node is the top node, in this node the data is split by a condition in to two branch nodes, which are nodes below the root node. In a branch node a similar split is made, based on a condition, to other branch nodes or a leaf nodes. In this leaf node the data is given a classification label, in this case churn or no churn. Decision trees are built by repeatedly splitting nodes based on the best splits, mostly evaluated by entropy level or Gini index. The process of splitting the data by a condition stops by a pre-determined stop criteria or if all class are separated perfectly. Normally machine learning algorithms are a black box, however with DT one could easily see what kind of decisions the machine learning algorithm made to split the data in to nodes. However, decision trees are prone to overfitting, to act against overfitting it is important to determine stop criteria.

Assumptions
No assumptions

Formulas
The decision of splits into branches or leaves is based on evaluation of goodness of the split. The most used evaluation techniques are based on entropy and Gini index. The entropy (Tangirala, 2020) is calculated by: $E(S) = \sum_{i=1}^{c} -p_i * log_2(p_i)$. S is the current state, c is the number of classes, and p_i is class i divided by the total number of datapoints in the current state. The information gain is based on entropy and calculated by: $IG(Y,X) = E(before) - \sum_{j=1}^{k} E(j, after)$. K is the number of subsets generated by the split, and (j, after) is subset j after the split. So, in other words, to calculate the information gain, first the entropy of the parent is calculated and then the entropy of the children. Then the split with the highest information gain is evaluated as the best split and is introduced to the DT. Alternatively instead of the information gain the Gini index could be calculated $Gini = 1 - \sum_{i=1}^{C}(p_i)^2$. The Gini index is calculated by subtracting the sum of the squared probabilities of each class from one. Then similar as information gain the difference between Gini index of the parent and children is determined, to evaluate the goodness of split. The highest Gini index difference between the parent and children is the best split and is introduced to the DT.

Fitting
First, the decision tree starts with one root node that contain all the training data, this root node is split in to branches and leaves. The splits are selected based on split conditions that best divide the dataset into homogenous subsets (Tangirala, 2020). The goodness of split is measured by Gini index or information gain. The decision tree makes splits until each class is separated perfectly or a pre-determined stop criteria is met.

4.4.5 Random forest (RF)

Main idea
The random forest algorithm is first described by Ho (1995) and extend by Breiman (2001). Random forest is an ensemble learning method, it combines multiple learning algorithms. RF makes use of bagging; N random training data sets are generated by random sampling from the training data (Breiman, 1996). A random forest classifier is a classifier consisting of a collection of tree-structured classifiers h(x,k), k = 1,...,N where k are independent identically distributed random vectors and each tree, h, casts a "vote" for the most popular class at input x (Breiman, 2001). So, a large number of decision trees are generated from the training data by randomly selected subset of the data. Each decision tree is built until each node ends in a leaf, one of the pre-determined stop criteria is met or the classes are perfect distinguish. The generated decision trees differ from each other, each DT is made from a different subset resulting in different splits and decision tree length. The predicted classification is based on "votes" from each individual DT, the majority of all votes, which is a classification, in this case churn or no churn, will be the predicted class. In comparison to DT, the greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. However, the RF is harder to understand than a DT because RF consist of the multiple decision trees (Breiman, 2001).

Assumptions
No assumptions

Formulas
Random forest is built with multiple decision trees who are trained on different subsets of training data, so the formulas for decision tree also apply here.

Fitting
The training data is first split into multiple subsets, each subset starts as a root node. This root node is split in to branches and leaves, based on the goodness of split measures, based on information gain or Gini index, until each class is separated perfectly or a pre-determined stop criteria is met.

4.4.6 SVM

Main idea
Introduced by Boser et al. (1992), they developed a supervised algorithm that has evolved to as what we know now as support vector machines (SVM). SVM separates the classes by hyperplane, which is a decision boundary that differentiates the two classes. The margin is calculated as the perpendicular distance from hyperplane to the closest points. These points are only relevant to determine the position of the hyperplane, these points are called support vectors, they support the hyperplane. The hyperplane that can separate the two classes with maximal margin is the optimal hyperplane. To separate non-linear data,

a kernel function is used, to transform the data in such a way that it is linear separable by a hyperplane.

Assumptions
SVM assumes the data is independent and identically distributed.

Formulas
The norm of the vector, also known as the Euclidean norm is calculated by $\|w\| = \sqrt{\sum_{i=1}^{n} w_n^2}$. The following equation have to be met: $y_i(w^T x_i + b) \geq 1 - \xi_i$ (Ukil, 2007). Where y_i, is a positive, 1, or a negative class, -1, ξ_i is the error term for the classes that can't be separated by the hyperplane. The margin we want to maximize is $max = \frac{2}{\|w\|}$. The objective function is: $min(\frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} \xi_i)$ such that $y_i(w^T x_i + b) \geq 1 - \xi_i$ for all $i = 0, 1, ..., n$ (Boser et al., 1992). To separate the classes from nonlinear data, kernel functions are used. The polynomial kernel is $k(x, y) = (x \cdot y + 1)^d$, where d is the degree of the polynomial. The sigmoid kernel is $k(x, y) = tanh(\alpha x^T y + c)$ and the RBF kernel is $k(x, y) = exp(-\frac{\|x-y\|^2}{2\sigma^2})$. There are more kernel function but these are the most often used.

Fitting
If the data can't be separated linear, than a kernel function have to be selected and applied. If the data is now linear separable then a hyperplane can be found with the maximum margin, that separate the classes as best as possible.

4.4.7 Artificial neural network (ANN)

Main idea
An artificial neural network (ANN) is a biologically inspired computational network, which consist of neurons and weighted connections between the neurons (Jain et al., 1996). In this report a multilayer perceptron (MLP) from the sklearn package is used. MLP is a feed forward artificial neural network, it consists of an input, output and one or multiple hidden layers (Ramchoun et al., 2016). Each layer is fully connected to the following layer, the output from each layer is passed to the next layer. The nodes of the layer use a nonlinear activation function (e.g., relu, sigmoid or tanh), except for the input layer (Jain et al., 1996). An ANN uses a supervised learning process, this process consists of forward propagation and backpropagation (Yegnanarayana, 2009).

Assumptions
No assumptions

Formulas
First, the input features x are processed with weights w and bias b, the first process in vector notation is: $Z^{[1]} = W^{[1]} X + b^{[1]}$ (Yegnanarayana, 2009). The outcome is transformed by $A^{[1]} = \sigma(Z^{[1]})$, where σ is the activation function.

In the next layer a similar process happens, only this time the outcome of the previous layer, $A^{[1]}$ is now the input: $Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$. The outcome, $Z^{[2]}$, is transformed by a nonlinear activation function as well. This process repeats until the last layer of the neural network, this process is called forward propagation, which results in a prediction. A threshold is used to turn the prediction in to a class.

Fitting
The neural network is trained by forward propagation and backpropagation. In the first step of the training of the neural network, random weights are given to the connections between the neurons, next repeatedly a process of forward and backward propagation is used until the model stops improving (Yegnanarayana, 2009). With forward propagation, input is given to the input layer and the output is generated on the current weights of the connections between the neurons, as explained in the formulas section. The loss of the output is calculated using the loss function, basically the expected outcome is compared with the outcome of the model. The outcome of the loss function is used to improve the model with backpropagation. Backpropagation computes the gradient of the loss function with respect to the weights of the network. Backpropagation computes the gradient one layer at the time, iterating backward starting from the last layer. The gradient of the loss function is used to give the nodes with higher error rates, less weights than nodes with lower error rates. The difference between ANN classifier and regressor is the loss function for classification e.g., a log-likelihood loss function is used and for a regressor e.g., the mean squared error.

4.4.8 Gradient boosting (GB)

Main idea
Gradient boosting is like RF an ensemble learning method, it uses multiple models to make a prediction. Each model is not built individually on a subset of data like RF, but each model is built sequentially, correcting the error of the previous model (Mayr et al., 2014). This is called boosting; a weak learner is trained on the training data and the next weak learner is trained on the errors of the previous model. A weak learner is usually decision trees with only one split. Boosting combines a set of weak learners into a strong learner to minimize the training errors.
This algorithm is named gradient boosting because the target outcomes for each case are set based on the gradient of the error with respect to the prediction. Each new model takes a step in the direction that minimizes the prediction error. Likewise as neural networks, gradient boosting makes use of the gradient descent algorithm. The gradient descent algorithm is a greedy algorithm, that finds the local minimum by iteratively moving the opposite direction of the gradient of the objective function (Ruder, 2016). The gradient of the objective function, is basically the partial derivative of the loss function with respect to the predictions, so it describes the steepness and the direction of the error func-

tion. Where a neural network tweaks the gradient descent model parameters, the gradient boosting algorithm descends the gradient by introducing new models. Instead of using gradient descent to estimate a parameter, the new weak learner is fitted against the negative gradient vector of the previous iteration (Mayr et al., 2014). Since the loss is inflated by mistakes, gradient descent will push the algorithm towards creating a new learner, that will focus on these mistakes that causes the loss to be high. So, it starts with a simple weak learner with each weak learner added to the ensemble, the algorithm takes a step towards problematic observations that are difficult to predict, since it only adds weak learners that minimizes the loss function. A threshold is used to turn the prediction in to a classification prediction.

Assumptions
No assumptions

Formulas

Algorithm 1 Gradient boosting algorithm

```
1: $F_0(x) = argmin_\rho \sum_{i=1}^{n} L(y_i, \rho)$
2: for m=1 to M do:
3:     $\tilde{y_i} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x)}\right]_{F(x)=F_{m-1}(x)} for\ i = 1, ..., n$
4:     $a_m = argmin_{a,\beta} \sum_{i=1}^{N} [\tilde{y_i} - \beta h(x_i : a)]^2$
5:     $\rho_m = argmin_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \rho h_m(x_i : a_m))$
6:     $F_m(x) = F_{m-1}(x) + v * \rho_m h(x : a_m)$
7: end for
```

In Algorithm 1 the gradient boosting algorithm from Friedman (2001), is shown. Comments for each row in Algorithm 1 are given below.

1. The first step is creating an initial constant value prediction F_0 that minimizes the expected value of the specified loss function for all predictions. The loss function for regression is mostly the mean squared error and for classification, this is mostly the log loss. The actual values are y_i and L is the loss function. The constant value prediction is a simple first prediction, that minimizes the overall loss.

2. Weak learners are created until M.

3. The residuals $\tilde{y_i}$ are compute by taking the derivative of the loss function and is multiply with -1, this is the negative gradient.

4. A weak learner is fit, that best fits the negative gradient vector minimising the error under the residuals by using a least squares minimization function.

5. Followed on (4) by a single parameter optimization from the original criterion, finding a $h(x : a)$ which minimize the lost function and which is feasible solution for (4).

6. At last, the model is updated with a new weak learner that minimizes the overall loss the most, where v is the learning rate parameter, the loop continues until M trees are created. The learning rate parameter and M, the amount of estimators are hyperparameters to tackle overfitting.

Fitting
First, a simple model, an initial constant value prediction, is determined that that minimizes the overall loss of the chosen loss function. From the loss function we compute the residuals by taking the negative gradient of the loss function. A new model is fitted on the residuals of the previous model as target variable. The predicted residuals times the learning rate are added to the previous predictions. Then residuals are calculated again and new models are fitted on the residuals as target variable. This process continues until the sum of residuals stays constant or if M iterations are achieved.

5 Results

The results of the machine learning algorithms are discussed in this section. Each algorithm/model is discussed separately, if there are assumptions then these are examined in the appendix in the assumptions section. In this section, for each algorithm/model the selected features and the chosen hyperparameter parameters are discussed. The main results are shown in table 2. All the results are obtained with a stratified K-fold cross-validation, where K=5. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. Next the results are shown of the algorithm/models trained and tested on a different time window, a time window of 1 year. In the next sub section, the results are shown for the algorithms/models without the *RelationalLength* feature and clumpiness features, for the algorithms/models where at least one of those features was part of their feature selection. In this way is investigated if the extension from RFM to RFMCL features improves the predictive performance of the algorithm/models. At last, the results are shown for the alternative missing value strategies as stated in the missing value strategy section, for the algorithm/models which including the numeric features where the missing values are imputed.

5.1 Logistic regression

Features and hyper parameters selections
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 46 features selected for logistic regression (LR). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid

search: the solver, the penalty and the C parameter. The best result of the grid search is the hyperparameter setting: C=1, penalty=l2, solver=newton-cg. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.2 Naive Bayes

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are shown. The features are selected by forward selection, in total there are 12 features selected for Naive Bayes (NB). From the multiple versions of Naive Bayes, the Gaussian Naive Bayes has the best predictive performance. Only the var_smoothing hyperparameters is tuned by a repeated stratified K-fold cross-validation grid search. The best result of the grid search is value 0.0001 for the hyperparameter var_smoothing. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.3 K-nearest-neighbour

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 20 features selected for K-nearest-neighbour (KNN). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the leaf size, the number of neighbours and the power parameter p. The best result of the grid search is the hyperparameter setting: leaf_size=1, n_neighbours=13 and p=1. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.4 Decision tree

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 28 features selected for the decision tree (DT). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the max depth, the min samples per leaf and the criterion parameter. The best result of the grid search is the hyperparameter setting: max_depth= 5, min_samples_leaf=145 and criterion=Gini. The features and the hyperparameters are selected by the highest obtained test accuracy score

from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.5 Random forest

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 28 features selected for random forest (RF). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the max depth, max features, the min samples split, the min samples leaf, bootstrap and the number of trees/estimators. The best result of the grid search is the hyperparameter setting: n_estimators=1200, min_samples_split=2, min_samples_leaf=2, bootstrap=True, max_features= sqrt and max_depth=20. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.6 SVM

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 27 features selected for support vector machine (SVM). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the kernel, the gamma and the C parameter. The best result of the grid search is the hyperparameter setting: kernel=rbf, gamma=0.05, and C=300. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.7 Artificial neural network

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 17 features selected for artificial neural network (ANN). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the solver, the learning rate init, the number of hidden layers, the learning rate, the alpha and the activation function. The best result of the grid search is the hyperparameter setting: solver=adam, learning_rate_init=0.001, hidden_layer_sizes=71, learning_rate=invscaling, alpha=0.001, activation=relu. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

5.8 Gradient boosting

Features and hyper parameters selection
In the appendix in tables **??**, **??** and **??** the selected features for each algorithms/models are revealed. The features are selected by forward selection, in total there are 27 features selected for gradient boosting (GB). The following hyperparameters are tuned by a repeated stratified K-fold cross-validation grid search: the max depth, max features, the min samples split, the min samples leaf, subsample, learning rate, the loss function and the number of trees/estimators. The best result of the grid search is the hyperparameter setting: max_depth=3, min_samples_split=2, min_samples_leaf=1, subsample=1, learning_rate=0.1, max_features=none, loss=log_loss and n_estimators=100. The features and the hyperparameters are selected by the highest obtained test accuracy score from the select/search method. The results of the predictive performance of all the algorithms/models are presented in table 2.

	train_acc	test_acc	precision	recall	F1	AUC
KNN (C)	0.857	0.836	0.746	0.756	0.749	0.906
NB (C)	0.813	0.811	0.689	0.757	0.721	0.882
DT	0.842	0.842	0.753	0.768	0.758	0.917
LR (C)	0.826	0.828	0.762	0.684	0.717	0.896
SVM (C)	0.849	0.843	0.749	0.780	0.761	0.912
RF (C,L)	0.948	0.850	0.773	0.765	0.765	0.927
ANN (C,L)	0.859	0.847	0.755	0.789	0.769	0.924
GB	0.854	0.849	0.762	0.783	0.769	0.925

Table 2: Results from the stratified K-fold cross-validation, K=5, for the algorithms/models where each algorithm/model has their own feature selection and own hyperparameter optimization, training on a time window of 2 year and 4 months. (C) Indicates that the algorithms/models have at least one of the features *Clumpiness* and *ClumpinessCat* in their feature selection, (L) indicates that the algorithms/models have *RelationalLength* in their feature selection. (C,L) Indicates that the algorithms/models includes both features in their feature selection.

5.9 Time window change

The time window which you train the algorithms/models on could be a parameter that increases or decreases the performance of the algorithms/models. As mentioned before, in this section, in table 2 the results of the all the machine learning algorithms are presented. These predictions are based on all the data available where the last year is used for classification, the algorithms are trained on approximately 2 year of data. Alternatively, the predictive algorithms/models could be evaluated with algorithms/models trained on a reduced time window. To evaluate if the time window is a parameter that increases or decreases the predictive performance of the algorithms/models, the algorithms/models are trained on a reduced time window, a time window of 1 year. In table 3, the predictive performance of the algorithms/models are shown, where the algorithms/models are trained on a reduced time window of 1 year.

	train_acc	test_acc	precision	recall	F1	AUC	Δ test_acc
KNN	0.847	0.822	0.738	0.752	0.743	0.900	-0.013
NB	0.796	0.794	0.688	0.735	0.709	0.856	-0.017
DT	0.830	0.830	0.738	0.786	0.760	0.909	-0.012
LR	0.812	0.808	0.746	0.675	0.705	0.885	-0.020
SVM	0.838	0.832	0.747	0.779	0.760	0.907	-0.011
RF	<u>0.939</u>	<u>0.836</u>	<u>0.762</u>	0.768	0.762	0.915	-0.014
ANN	0.848	0.834	0.746	<u>0.790</u>	<u>0.765</u>	0.916	-0.013
GB	0.840	0.832	0.748	0.779	0.760	<u>0.918</u>	-0.017

Table 3: Results from stratified K-fold cross-validation for the algorithms/models where each algorithm/model has their own feature selection and own hyperparameter optimization, trained on a reduced time window of 1 year. The last column is the difference with the test_acc of table 2.

5.10 RFM or RFMCL

In this report the RFM model is extended to the RFMCL model, *RelationalLength* feature (L) and the clumpiness features (C) which consist of 2 features, *Clumpiness* and *ClumpinessCat*, are added to the RFM model. To test if these features increase the predictive performance of the algorithms/models, a test is conducted with algorithms/models which include *RelationalLength* and/or *Clumpiness* and/or *ClumpinessCat* features in their feature selection by leaving out 1, 2 or all of these features.

The algorithms/models which include the Clumpiness and/or *ClumpinessCat* and/or *RelationalLength* features are trained on approximately 2 year of data without 1, 2 or all of the these features. In this way the predictive performance contribution of those added features could be evaluated. In table 4 the predictive performance is shown of the algorithms/models without all of the clumpiness features, so without at least one or both of the features *Clumpiness* and/or *ClumpinessCat*, if at least one or both of the features was present in their feature selection. In table 5 the algorithms/models are shown without the feature *RelationalLength*, for the algorithm/models were the feature *RelationalLength* was present in their feature selection.

At last, in table 6, the predictive performance is presented for the algorithms/models which included both features, but in this case is tested the performance without *RelationalLength* feature and at least one or both of the features *Clumpiness* and *ClumpinessCat*. The tables of this section could be compared with table 2 were (C) indicate the algorithms/models including at least 1 or both of the *Clumpiness* and/or *ClumpinessCat* feature, (L) indicate the algorithms/models with the *RelationalLength* feature in their feature selection, (C,L) indicate the algorithms/models including *RelationalLength* feature and at least one or both of the features *Clumpiness* and/or *ClumpinessCat* in their feature selection. For more details about the feature selection of each algorithm/model, one could look in the appendix in tables **??**, **??** and **??**.

	train_acc	test_acc	precision	recall	F1	AUC	Δ test_acc
KNN	0.856	0.833	0.741	0.749	0.743	0.906	-0.003
NB	0.809	0.809	0.687	0.754	0.717	0.881	-0.002
LR	0.825	0.824	0.755	0.682	0.713	0.896	-0.004
SVM	0.845	0.839	0.740	0.781	0.757	0.911	-0.004
RF	<u>0.938</u>	<u>0.847</u>	<u>0.766</u>	0.765	<u>0.762</u>	<u>0.926</u>	-0.003
ANN	0.858	0.844	0.758	<u>0.768</u>	0.760	0.923	-0.003

Table 4: Results from stratified K-fold cross-validation for the algorithms/models without the *Clumpiness* or *ClumpinessCat* features, for the algorithms/models which included at least one of the *Clumpiness* or *ClumpinessCat* feature in their feature selection. The last column is the difference with the test_acc of table 2.

	train_acc	test_acc	precision	recall	F1	AUC	Δ test_acc
RF	<u>0.934</u>	<u>0.849</u>	<u>0.773</u>	0.763	<u>0.765</u>	<u>0.925</u>	-0.001
ANN	0.857	0.845	0.755	<u>0.779</u>	0.764	0.923	-0.002

Table 5: Results from stratified K-fold cross-validation for the algorithms/models without the *RelationalLength* feature, for the algorithms/models which included the *RelationalLength* feature in their feature selection. The last column is the difference with the test_acc of table 2.

	train_acc	test_acc	precision	recall	F1	AUC	Δ test_acc
RF	<u>0.922</u>	<u>0.845</u>	<u>0.764</u>	0.762	0.759	<u>0.924</u>	-0.005
ANN	0.854	0.844	0.751	<u>0.785</u>	<u>0.764</u>	0.923	-0.003

Table 6: Results from stratified K-fold cross-validation for the algorithms/models without *RelationalLength* feature, and the clumpiness features, at least one of *Clumpiness* and *ClumpinessCat* feature, which included both features in their feature selection. The last column is the difference with the test_acc of table 2.

5.11 Different missing value strategy

In table 2 the missing values for the *Clumpiness* feature are imputed with the mean and for *Storedistance* feature with the median. Alternatively, a different missing value strategy is evaluated as mentioned in the missing value strategy section. The missing value strategy is based on fitting the best known distribution to the distribution of the feature. Next, a random value is generated from the known distribution, within the range of the distribution of the feature we want to impute. In this way, the imputation strategy tries not to disrupt the variance of that feature, this is different from mean/median imputation where the variance of the feature is reduced by the imputation. In table 7 the predictive performance of the algorithms/models are shown with the alternative missing value strategy. In table 7 the algorithms/models including the *Clumpiness* feature are indicated by (1), a (2) for the *Stordistanceclosest* feature and (1,2) if the algorithm/model includes both features. The algorithm/models in table 7 could be compared with performance of those algorithm/models in table 2, where for those features the mean/median imputation is applied, in this way one could evaluate the 2 missing value strategies.

	train_acc	test_acc	precision	recall	F1	AUC	Δ test_acc
NB (2)	0.813	0.809	0.692	0.742	0.715	0.880	-0.002
LR (1,2)	0.826	0.825	<u>0.757</u>	0.681	0.713	0.896	-0.003
RF (1,2)	<u>0.935</u>	<u>0.847</u>	0.755	<u>0.782</u>	<u>0.765</u>	<u>0.923</u>	-0.003
ANN (2)	0.860	0.846	0.755	<u>0.782</u>	<u>0.765</u>	<u>0.923</u>	-0.001

Table 7: Results from stratified K-fold cross-validation where each model has their own feature selection and for each model the hyperparameters are tuned. The algorithms/models are trained on a time window of 2 year and 4 months, where the missing values of the *Clumpiness* and *Stordistanceclosest* feature are imputed with the technique mention in the missing value strategy section, a technique that tries not to disrupt the variance of that feature. (1) Indicates the algorithm/model includes the *Clumpiness* feature in their feature selection, (2) indicates that the *Storedistance* feature is part of the feature selection and (1,2) indicates that both features are part of the feature selection of the algorithm/model. The last column is the difference with the test_acc of table 2.

6 Conclusion

In this report 2 main research questions are stated:

1. Which machine learning algorithms have the best predictive performance in predicting customer churn?
2. What kind of methods improve the performance of machine learning algorithms in predicting customer churn?

More specifically for main research question 2, 3 sub research questions are stated:

- Which features in the dataset contribute to predictive performance of the machine learning algorithms?

- Which features can be engineered and improves the predictive performance of the machine learning algorithms?
- What is a missing value strategy that adds value to the predictive performance of the machine learning algorithms?

In this section, the author will answer the research questions. Firstly, the author will go into the first main research question, the best performing machine learning algorithms. Secondly, the author will go into the second research question and it's sub research questions.

Best algorithm/model
The best performing algorithm/model based on the test accuracy, AUC and precision performance is the random forest algorithm (RF) trained on all the data, except the last year, with the mean/median imputation as missing value strategy, with hyperparameter optimisation as stated in the result section and including the features as stated in the appendix in tables **??**, **??** and **??**. The best performing algorithm/model based on the recall performance metric is artificial neural network (ANN). The ANN and the gradient boosting (GB) have both the highest score on F1, trained on all the data, except the last year, with the mean/median imputation as missing value strategy, with hyperparameter optimisation as stated in the result section and the features selection as stated in the appendix in tables **??**, **??** and **??**.

Best time window
The algorithms/models trained on all the data except the last year have a better performance than the algorithms/models trained on 1 year of the data, The algorithms/models trained on all the data except the last year, have approximately on average 0.015 improved test accuracy performance than trained on 1 year of the data.

Features in the dataset
CONFIDENTIAL

RFM or RFMCL
The engineered RFM features contribute to the predictive performance of the algorithms/models. From figure **??**, one could conclude that *Recency* feature is by far the most important predictor of customer churn. From the RFM features, the *Recency* feature ends up in all the feature selection of the algorith/models and the *Frequency* feature end up in almost all the algorithms/models feature selection. The *Monetary* feature is only part of the feature selection of the logistic regression model, probably because of the high correlation with the *Frequency* feature. Including the *Clumpiness* feature in the algorithms/models where at least one of the *Clumpiness* or *ClumpinessCat* feature is part of their feature selection, increases the test accuracy performance on average by approximately 0.0035. Including the *RelationalLength* feature in the algorithms/models which

include the *RelationalLength* feature in their feature selection, increases the test accuracy performance on average by approximately 0.0015. For the algorithms/models which include the *RelationalLength* feature and at least one of the *Clumpiness* or *ClumpinessCat* feature in their feature selection, including both features increase the test accuracy performance on average by approximately 0.004. So, extending the RFM features to RFMCL features improves overall the predictive performance of the algorithms/models, except for DT and GB because none of these features end up in their feature selection.

Other useful engineered features
The features about the customers' most often bought product category and the storename where the customer most often bought products, are categorical features that end up in most of the feature selection of the algorithms/models.

Missing value strategy
The mean/median imputation is the better of the 2 tested missing value strategy. The alternative missing value strategy where you try not to disrupt the variance of the feature has approximately on average 0.002 poorer test accuracy performance. This is valid for the algorithms/models were at least one of the imputed numeric features is present, so *Storedistance* and/or the *Clumpiness* feature is part of their feature selection. For the categorical variables consider the missing values as a separate category, turned out to be the best method to handle the missing values. The missing value categories, categorical values with NoInfo in their name, are one of the better categorical predictors of customer churn.

7 Discussion

In this section, first the results are discussed, then some notes about techniques tested but not applied. Next is discussed the limitations of the dataset, recommendations and at last suggestions about future work.

Results
Overall the predictive performance of all the algorithm/models are relevant, a test accuracy score is achieved with a range of 0.81 until 0.85. So, with the right feature selection and hyperparameter setting the performance of all algorithms/models are relevant. As expected, the more classical machine learning algorithms: LR, SVM, NB, DT and KNN, perform worse than the more advanced machine learning algorithms: RF, ANN and GB. From the more classical machine learning algorithms the worst performing algorithm based on test accuracy performance is the Naive Bayes (NB) algorithm and the best performing algorithm/model is the support vector machine (SVM) algorithm. The difference in performance of the worst and best performing classical machine learning algorithms is approximately 0.031.

The more advanced machine learning algorithm differ less in test accuracy performance than the more classical machine learning algorithms. The worst performing of the more advanced machine learning algorithms, based on the test accuracy performance, is the artificial neural network (ANN) model. The best performing is the random forest (RF) algorithm, ANN and RF differ in test accuracy approximately 0.003.
The best performing algorithm/model could be chosen on the most relevant performance score of the algorithm/model for the management. So, if for the management of a company, it is more important that the algorithm/model predicts customers who churn and who do not churn, correct, than the random forest (RF) algorithm is more suited. If the management wants most of the customers that churn, predicted as churn by the algorithm/model, despite more false positives, customers are predicted as churn who did not churn, then the management should consider to choose the algorithm/model with the best performance on recall, like ANN. Alternatively, the management could increase the recall performance of RF algorithm by using a different threshold. If the management wants a better accuracy performance than ANN but does not mind to have more false positives than RF, the management could consider the algorithm/model with the same F1 score as ANN, the gradient boosting (GB) algorithm, as stated in the conclusion section. Alternatively, the algorithm/model can be chosen on preferable characteristics of the algorithm/model: train time, explainability etc.

Techniques tested but not applied
In this research multiple techniques are tested to improve the predictive performance of the algorithm/models, these techniques did not end up in the report due to worse predictive performance or not making sense on this dataset. In the dataset is class imbalance, multiple techniques have been executed to balance response variable in the trainset. Balancing the response variable in the trainset for the algorithm/models has led to worse performance of the algorithm/models, therefore it is decided to not apply any of these techniques. It is tested whether K-means clustering improved the predictive performance of the algorithms/models, this resulted in not improving the predictive performance of the algorithms/models. PCA and MCA are tested to reduce the number of features, however little of the variance was explained by a reduced number of features, therefore it is decided to not apply one of those methods. Combining features and some other transformations are tested as well, however most of them resulted in worse predictive performance of the algorithms/models.

Limitations
The dataset consists mostly of transactional data where the frequency of purchases per customer was mainly low in the dataset, about 30 percent has a purchase frequency lower than 3. As stated in the feature engineering section, the clumpiness measure is not well defined for customers who have a low frequency of purchases. Therefore, as stated in the missing value strategy section for the low purchase frequency customers, the values for the *Clumpiness* feature are imputed by a missing value strategy. If the dataset consists of customers

with more frequent purchases, substantially more than 3, less values for the clumpiness measure score would be invalid, therefore none or less values have to be imputed with a missing value strategy. In this way the features *Clumpiness* and *ClumpinessCat* could be more valuable. Alternatively, if the customers bought items more frequently in the dataset, features or models based on time-series could be developed to improve the predictive performance of the algorithms/models. Another limitation is that the data does not contain customer behaviour data from the site, so e.g., log in time, page views, clicks etc. This data has information about customer behaviour and therefore could be valuable data to predict customer behaviour.

Recommendations
Based on this research, the author would recommend to use one of the advanced algorithm/models, so either RF, GB or ANN as the algorithm/model to predict customer churn. These algorithm/models with the right feature selection and hyperparameter optimisation are all relevant. Based on the accuracy performance, the author would suggest RF and based on recall the author would recommend ANN or RF using a different threshold, however this would result in more false positives than for RF, as stated in the conclusion section. Based on F1 score the author would recommend either ANN or GB, where GB has a slightly better accuracy than ANN.
In this report training and testing on a reduced time window decreased the predictive performance of all the algorithm/models. Therefore, it is preferred to use all the data instead of using a reduced time window of the dataset. The most important feature to predict customer churn is the *Recency* feature based on mutual information with the response variable. Other important features are the *Frequency* and the missing value category feature because they end up in most of the models. Where the missing value category is a categorical requirement that has not been filled in by the customer when enrolling in the company's customer database. So, this could indicate that missing enrolling requirements, used as a separate category, gives an indication about the customer or how he looks against doing business with the company. Removing the *RelationaLength* feature and the clumpiness features by the algorithm/model which included one or both of these features, decreased the performance of those algorithm/models. Therefore, the author would recommend to extent the RFM model to RFMCL model. It can be a valuable extension because the RFM model does not include customer relation length and how the purchase are spread over time, clumpy or not clumpy. The clumpiness features could especially be relevant for a dataset where the frequency of events is high, because if the frequency of events gets low, lower than 3, the clumpiness measure score is not well defined. Besides that *RelationaLength*, the *Clumpiness* and *ClumpinessCat* features are predictors of customer behaviour, those features could be relevant features for the analysing software as well. Companies could use these features for analysing customer behaviour, they could use these features to target, segmented or use different marketing/retention strategies for their clientele. As a missing value strategy, the author would suggest to consider the missing values for the categorical fea-

tures as a separate category. For the numeric features the author would suggest, if the correlation with other features is low, to use the mean/median imputation strategy.

Future work
Future work could be based on e-commerce data only, with data that contain customer behaviour data, e.g., login time, page visit etc. In this way, customer behaviour could be predicted more accurate by the algorithm/models. Another suggestion would be to test the clumpiness features on a dataset where the frequency of purchases or visits is more frequent, so is present in the data substantially more than 2. On such a dataset it could be interesting to derive features from timeseries data and/or make timeseries models to predict customer churn. Another suggestion for future work is to investigate if it is beneficial, to extend the segmentation from RFM to RFMCL for the companies' analysis and marketing/retention strategies. A more simple addition could be to test if adding those features to the software is beneficial for the companies' analysis and marketing/retention strategies.

8 Reference

References

Abiad, A., Arao, R. M., & Dagli, S. (2020). The economic impact of the covid-19 outbreak on developing asia.

Amine, A., Bouikhalene, B., Lbibb, R., et al. (2015). Customer segmentation model in e-commerce using clustering techniques and lrfm model: The case of online stores in morocco. *International Journal of Computer and Information Engineering*, *9*(8), 1993–2003.

Anderson, E., & Weitz, B. (1989). Determinants of continuity in conventional industrial channel dyads. *Marketing science*, *8*(4), 310–323.

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (2012). The 'k'in k-fold cross validation. *20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 441–446.

Berrar, D. (2018). Bayes' theorem and naive bayes classifier. *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, *403*.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory*, 144–152.

Bouckaert, R. R. (2004). Naive bayes classifiers that perform well with continuous variables. *Australasian joint conference on artificial intelligence*, 1089–1094.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Buckinx, W., & Van den Poel, D. (2005). Customer base analysis: Partial defection of behaviourally loyal clients in a non-contractual fmcg retail setting. *European journal of operational research*, *164*(1), 252–268.

Bujang, M. A., Sa'at, N., Bakar, T. M. I. T. A., Joo, L. C., et al. (2018). Sample size guidelines for logistic regression from observational studies with large population: Emphasis on the accuracy between statistics and parameters based on real life clinical data. *The Malaysian journal of medical sciences: MJMS*, *25*(4), 122.

Chen, I. J., & Popovich, K. (2003). Understanding customer relationship management (crm): People, process and technology. *Business process management journal*.

Cordeiro, G. M., Nobre, J. S., Pescim, R. R., & Ortega, E. M. (2012). The beta moyal: A useful skew distribution. *International Journal of Research and Reviews in Applied Sciences*, *10*(2), 171–192.

Dubrovski, D. (2001). The role of customer satisfaction in achieving business excellence. *Total quality management*, *12*(7-8), 920–925.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of statistics*, 1189–1232.

Gregory, B. (2018). Predicting customer churn: Extreme gradient boosting with temporal data. *arXiv preprint arXiv:1802.03396.*

Gupta, S., & Lehmann, D. R. (2003). Customers as assets. *Journal of Interactive marketing*, *17*(1), 9–24.

Han, J., Kamber, M., & Pei, J. (2011). Data mining: Concepts and. *Techniques (3rd ed), Morgan Kauffman.*

Ho, T. K. (1995). Random decision forests. *Proceedings of 3rd international conference on document analysis and recognition*, *1*, 278–282.

Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (Vol. 398). John Wiley & Sons.

Hughes, A. M. (1994, February). *Strategic database marketing: The masterplan for starting and managing a profitable, customer-based marketing program* (Hardcover). Probus Pub Co.

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31–44.

Kadiyala, S. S., Srivastava, A., et al. (2002). Data mining for customer relationship management. *International Business & Economics Research Journal (IBER)*, *1*(6).

Khajvand, M., Zolfaghar, K., Ashoori, S., & Alizadeh, S. (2011). Estimating customer lifetime value based on rfm analysis of customer purchase behavior: Case study. *Procedia Computer Science*, *3*, 57–63.

Kracklauer, A. H., Mills, D. Q., & Seifert, D. (2004). Customer management as the origin of collaborative customer relationship management. In *Collaborative customer relationship management* (pp. 3–6). Springer.

Lakshmi, B., & Raghunandhan, G. (2011). A conceptual overview of data mining. *2011 National Conference on Innovations in Emerging Technology*, 27–32.

Lejeune, M. A. (2001). Measuring the impact of data mining on churn management. *Internet Research.*

Mayr, A., Binder, H., Gefeller, O., & Schmid, M. (2014). The evolution of boosting algorithms. *Methods of information in medicine*, *53*(06), 419–427.

Mrva, J., Neupauer, Š., Hudec, L., Ševcech, J., & Kapec, P. (2019). Decision support in medical data using 3d decision tree visualisation. *2019 E-Health and Bioengineering Conference (EHB)*, 1–4.

Mulak, P., & Talhar, N. (2015). Analysis of distance measures using k-nearest neighbor algorithm on kdd dataset. *Int. J. Sci. Res*, *4*(7), 2319–7064.

Peterson, L. E. (2009). K-nearest neighbor. *Scholarpedia*, *4*(2), 1883.

Radcliffe, D. (2022). Ecommerce in publishing: Trends and strategies. *Available at SSRN 4075431.*

Ramchoun, H., Ghanou, Y., Ettaouil, M., & Janati Idrissi, M. A. (2016). Multilayer perceptron: Architecture optimization and training.

Rao, V. R. (2015). Comments on” predicting customer value using clumpiness from rfm to rfmc”. *Marketing Science*, 213–215.

Reichheld, F. F., Markey Jr, R. G., & Hopton, C. (2000). E-customer loyalty-applying the traditional rules of business for online success. *European Business Journal*, *12*(4), 173.

Reichheld, F. F., & Sasser, W. E. (1990). Zero defeofions: Quoliiy comes to services. *Harvard business review*, *68*(5), 105–111.

Reinartz, W. J., & Kumar, V. (2000). On the profitability of long-life customers in a noncontractual setting: An empirical investigation and implications for marketing. *Journal of marketing*, *64*(4), 17–35.

Renjith, S. (2015). An integrated framework to recommend personalized retention actions to control b2c e-commerce customer churn. *arXiv preprint arXiv:1511.06975*.

Rish, I., et al. (2001). An empirical study of the naive bayes classifier. *IJCAI 2001 workshop on empirical methods in artificial intelligence*, *3*(22), 41–46.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Safavian, S. R., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, *21*(3), 660–674.

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia medica*, *24*(1), 12–18.

Tangirala, S. (2020). Evaluating the impact of gini index and information gain on classification using decision tree classifier algorithm. *International Journal of Advanced Computer Science and Applications*, *11*(2), 612–619.

Tsai, H.-H. (2011). Research trends analysis by comparing data mining and customer relationship management through bibliometric methodology. *Scientometrics*, *87*(3), 425–450.

Turban, E., Aronson, J., & Sharda, R. (2007). *Decision support and business intelligence systems*. Pearson Prentice Hall.

Ukil, A. (2007). Support vector machine. In *Intelligent systems and signal processing in power engineering* (pp. 161–226). Springer.

Williams, C. K., & Barber, D. (1998). Bayesian classification with gaussian processes. *IEEE Transactions on pattern analysis and machine intelligence*, *20*(12), 1342–1351.

Woo, J. Y., Bae, S. M., & Park, S. C. (2005). Visualization method for customer targeting using customer map. *Expert Systems with Applications*, *28*(4), 763–772.

Yegnanarayana, B. (2009). *Artificial neural networks*. PHI Learning Pvt. Ltd.

Yu, X., Guo, S., Guo, J., & Huang, X. (2011). An extended support vector machine forecasting framework for customer churn in e-commerce. *Expert Systems with Applications*, *38*(3), 1425–1430.

Zhang, Y., Bradlow, E. T., & Small, D. S. (2013). New measures of clumpiness for incidence data. *Journal of Applied Statistics*, *40*(11), 2533–2548.

Zhang, Y., Bradlow, E. T., & Small, D. S. (2015). Predicting customer value using clumpiness: From rfm to rfmc. *Marketing Science*, *34*(2), 195–208.

9 Appendix

9.1 Assumptions algorithms/models

9.1.1 Logistic regression

1. The response variable is binary. This is true the response is a binary classification.
2. The observations are independent. Not sure, each customer has a unique id, however it is totally possible that a customer uses multiple ids.
3. There is no multicollinearity among explanatory variables. This is not true as mentioned in the multivariate analysis there exist multicollinearity among the explanatory variables.
4. There are no extreme outliers. This is not true there are outliers even after the log transformation of monetary and monetary average.
5. There is a linear relationship between explanatory variables and the logit of the response variable. This is true, in the multivariate analysis, there is found correlation between the response variable and the explanatory variables.
6. The sample size is sufficiently large. True, according to Bujang et al. (2018) $n = 100 + 50i$ where i refers to number of independent variables in the final model. There are in total 118 features so n=6000 is smaller than 15490 customers who are in the reduced model.

9.1.2 Naive bayes

1. All features are independent (conditional independence). Not true, as mentioned in the multivariate analysis there exist correlation between the features, therefore there is no conditional independence.
2. (Gaussian) Naive Bayes assumes that continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. Not true as mentioned in the univariate analysis, monetary value is after the log transformation not normally distributed. This applies for multiple numeric features, mostly because the distribution is skewed positive.

9.1.3 SVM

1. It assumes data is independent and identically distributed. Not true as mentioned in the multivariate analysis there exist correlation between the features.

9.2 Software listing

1. Python
2. Pandas
3. For the machine learning algorithms and validation the sklearn packages are used.
4. Matplotlib
5. Seaborn