FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

# Assignment of bachelor's thesis

| | |
|---|---|
| **Title:** | Estimating Customer Lifetime Value for Media Houses |
| **Student:** | Bekbolot Khudaiberdiev |
| **Supervisor:** | doc. Ing. Pavel Kordík, Ph.D. |
| **Study program:** | Informatics |
| **Branch / specialization:** | Knowledge Engineering |
| **Department:** | Department of Applied Mathematics |
| **Validity:** | until the end of summer semester 2022/2023 |

## Instructions

Survey methods for estimating customer lifetime value (CLV) in e-commerce. Suggest how to compute CLV in the media domain. Find open-source solutions that can be extended to estimate CLV, design and implement extensions showing the value of visitors to product owners of media houses. Evaluate extensions on a media dataset and discuss added value for product owners.

---

*Electronically approved by Ing. Karel Klouda, Ph.D. on 30 September 2021 in Prague.*

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

# Estimating Customer Lifetime Value for Media Houses

***Bekbolot Khudaiberdiev***

Department of Applied Mathematics
Supervisor: doc. Ing. Pavel Kordík, Ph.D.

May 9, 2022

# Acknowledgements

I would like to thank my supervisor doc. Ing. Pavel Kordík for his guidance and patience, and my family for their support and love.
Apama jana avalaryma bilim algany mumkunchuluk bergenine cheksiz yraazymyn jana rahmatymdy aytam.

# Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46 (6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity. However, all persons that makes use of the above license shall be obliged to grant a license at least in the same scope as defined above with respect to each and every work that is created (wholly or in part) based on the Work, by modifying the Work, by combining the Work with another work, by including the Work in a collection of works or by adapting the Work (including translation), and at the same time make available the source code of such work at least in a way and scope that are comparable to the way and scope in which the source code of the Work is made available.

In Prague on May 9, 2022 ……………………

### Citation of this thesis

Khudaiberdiev, Bekbolot. *Estimating Customer Lifetime Value for Media Houses.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2022.

# Abstrakt

Cílem této práce je prozkoumat metody predikce CLV v obchodním prostředí založeném na předplatném, porovnat výkon těchto metod na veřejně dostupném souboru dat a implementovat rozšíření open-source projektu, které pomůže mediálním domům odhadnout CLV. Modely sBG/NBD, posílený regersní strom a neuronová síť jsou porovnávány na veřejně dostupné datové sadě z hudební streamovací služby KKBox. Na základě experimentů je nejlepším modelem v predikci CLV na úrovni jednotlivce a kohorty je posílený regresní strom, který je následně integrován do rozšíření s otevřeným zdrojovým kódem. Konečným produktem této práce je widgetové rozšíření projektu REMP - open-source software, který pomáhá mediálním domům monetizovat jejich obsah.

**Klíčová slova** predikce CLV, RFM, open-source, gradient boosting regression, umělá neuronová síť, sBG/NBD

# Abstract

The goal of this work is to explore methods of predicting CLV in a subscription-based business setting, compare these methods' performance on a publicly available dataset and implement an open-source extension that will help media houses estimate CLV. The sBG/NBD, gradient boosting regressor, and neural network models are compared on a publicly available dataset from music streaming service KKBox. Based on experiments the best model in predicting CLV on individual and cohort levels is gradient boosting regressor, which is subsequently integrated into open-source extension. The end product of this work is the widget extension of project REMP - open-source software that helps media houses to monetize their content.

**Keywords** CLV prediction, RFM, open-source, gradient boosting regression, artificial neural network, sBG/NBD

# Contents

# List of Figures

# List of Tables

# Introduction

In the last two decades, media services such as movie rentals, magazines, radio, and newspapers have moved into an online form and changed their business models to adapt to the new environment. They had to change the monetization strategy to reach more clients through the internet and keep up with competitors. In many cases, physical goods are being replaced by their digital form, which decreases the cost and overhead of products, but arguably makes it harder for the end customers to pay for the product due to lower perceived value. One of the most popular forms of monetization of content is subscription-based services, where users pay periodically for access to content, and ad-based services, where advertisers pay for users' clicks or impressions on their ads. Subscription-based service is preferable as it has several advantages over other types of monetization, such as more stable revenue, more profit, a better understanding of customers, and greater security in times of financial crisis. Moreover, with the advent of online commerce and streaming services, the whole industry became more customer-oriented and competing for customer loyalty, especially for high-value customers. A small change in retention rate can have a big impact on total profit. As such, companies are willing to invest in incremental improvements as they can lead to a big difference in profits at scale. Consequently, every company tries to model the customers' behavior to their best ability. The more accurate the model, the more profits company can have. Since online behavior can be tracked and saved, it has been vital for companies to utilize that data for a better behavioral model of customers. And CRM[1] systems helped companies find high-value customers, allocating a disproportionate amount of resources to them, and serving them the best version of their products. Thus, building strong long-term relationships and ultimately making more profits from them. To first find the high-value customers, companies look into various metrics to distinguish them from other customers, but one metric that best describes the

[1] Customer Relationship Management

value of a customer to a company is the customer lifetime value.

The customer lifetime value, usually denoted as CLV[2], is a metric showing the expected value of a customer to a company during the whole relationship. A vital metric used in marketing resource allocation, acquisition campaigns, retention campaigns, behavioral user segmentation, etc. To estimate CLV with traditional methods only a transaction log is required. But with the advent of online media services, much more data is available about each user's behavior, making it tempting to use them for better accuracy of prediction. Machine learning algorithms and techniques have been used for CLV prediction, as they can better model the complex behavior at the individual level and use all the available data on an individual level. But, there is no consensus on the best method of predicting CLV in a contractual setting. Thus, the goal of this thesis is to compare state-of-the-art methods and algorithms for CLV prediction on their predictive power in the contractual discrete business context.

[2] Customer Lifetime Value

Chapter 1

# Literature review

The term customer lifetime value has been around at least since 1988 [6] and since then has been an integral part of customer's description in the context of customer-company relationship. It is denoted as CLV.[3] The term has also different definitions in various literature as shown in Figure 1.1. The formal definition is - *"The present value of the future cash flows attributed to the customer during his/her entire relationship with the company"* [7].

Essentially, CLV, in layman's terms, is the total profit earned from the customer during the entire relationship with a company. It can be calculated on different granularity levels: whole customer base, cohort, individual. Usually, companies are interested in the CLV of the whole customer base for company valuation purposes, cohort level for marketing campaign purposes, and individual level for finding high-value customers. One should also distinguish the type of relationship between a customer and a company. The diagram in Figure 1.2 neatly describes two axes that define all customer-company relationships. We differentiate between the type of relationship a customer has with a company and their opportunities for purchase. If there is a legal agreement on a service provided by a company to a customer it's called contractual. Conversely, when there is no agreement it's non-contractual. The key difference between those two is that in the first case company knows whether the customer churned after the renewal period, while in the latter case the best company can do is to guess. Furthermore, if a customer can make a variable amount of purchases it would fit into the continuous opportunity of transactions, whereas if purchases happen only at some period, it would fit into the discrete opportunity for transactions. The reason this taxonomy is important in the context of CLV prediction is that all 4 groups have their distinct purchase behavior. Whereas some predictive models can be used for a discrete type of purchase as well as continuous, the distinction between contractual and non-contractual always requires different models [2, p. 63]. Thus, the

[3] It is denoted as CLV, CLTV, LTV, RLV, but it will be referred to as CLV in this work and will denote the future value of a customer to a company.

Definitions of LTV

| Definition | Article |
|---|---|
| The present value of all future profits generated from a customer | Gupta and Lehmann (2003) |
| The net profit or loss to the firm from a customer over the entire life of transactions of that customer with the firm | Berger and Nasr (1998) |
| Expected profits from customers, exclusive of costs related to customer management | Blattberg and Deighton (1996) |
| The total discounted net profit that a customer generates during her life on the house list | Bitran and Mondschein (1996) |
| The net present value of the stream of contributions to profit that result from customer transactions and contacts with the company | Pearson (1996) |
| The net present value of a future stream of contributions to overheads and profit expected from the customer | Jackson (1994) |
| The net present value of all future contributions to overhead and profit | Roberts and Berger (1989) |
| The net present value of all future contributions to profit and overhead expected from the customer | Courtheoux (1995) |

Figure 1.1: Different definitions of CLV [1, p. 182].

methods and algorithms for predicting CLV need to be chosen according to the business context in which transactions are made to best model the real purchase behavior. Since the subscription-based services fall into a contractual discrete relationship with a customer, we will focus solely on this category and respective methods of CLV prediction.

CLV - can be calculated in various ways and at different granularity levels (i.e. individual, cohort, company levels). We should keep in mind that there's no one formula to compute a CLV and it's just an estimation [8, p. 4]. The basic formula for a contractual setting is given by

$$\sum_{t=1}^{\infty} E[v(t)] \frac{S(t)}{(1+d)^d}, \tag{1.1}$$

where $E[v(t)]$ is the expected net cash flow at period t, $S(t)$ is the survival rate at period $t, d$ is the discount rate. If we want to get RLV[4], t needs to

[4] Residual Lifetime Value

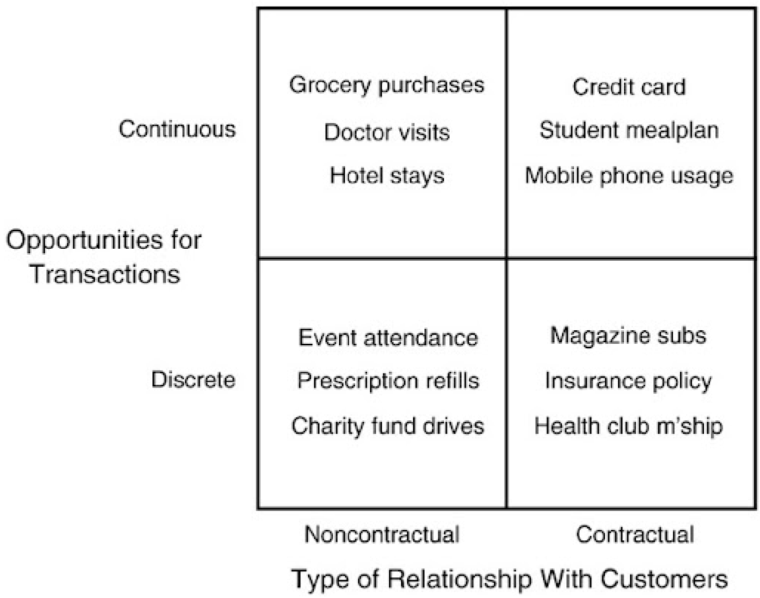
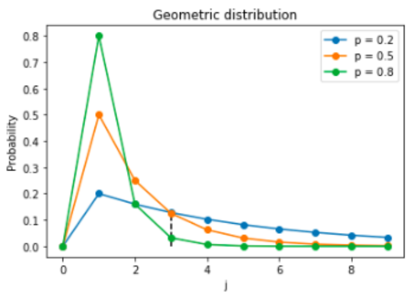



Figure 1.2: Types of relationship with customers[2].

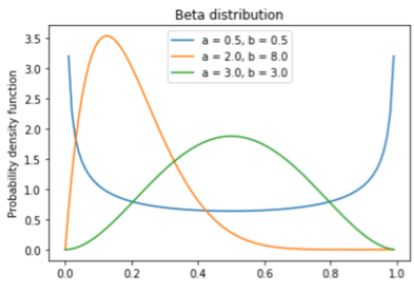
start from n+1, where n is the last period a customer renewed a contract. In most simplistic formulas E[v(t)] is assumed to be constant and factored out, leaving only survival rate as a function of t. The most common method is to divide it into a problem of predicting net cash flow, retention rate at the given period and combine them to get one numerical value. The benchmark for predicting CLV in a contractual setting is considered to be the sBG(shifted beta geometric) probabilistic model[9, p. 87]. Despite its simplicity, this model showed great results and is industry-accepted due to low demands for data, ability to project into any n-th future period, explainability of prediction, and low computational demand. Since probabilistic models use only RFM data, there have been efforts to incorporate more of the collected data on users, thus trying to increase the accuracy of prediction on an individual level. Most of those advances had been accomplished with machine learning methods and algorithms, ranging from decision trees, random forests to embeddings and deep neural networks. I will discuss those in more detail in the following chapters.

## 1.1 sBG/NBD model

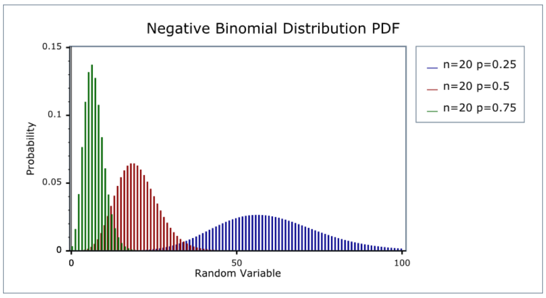
The shifted geometric model was proposed by Peter Fader and Bruce Hardie in their seminal work "How to project retention" [10]. The model was intended to account for an increase in the retention rates of cohorts as time progressed. It was a known problem that caused linear models to drop accuracy as the projection went into the future. One might assume at first, that it's because customers become more loyal with time and thus retention rate increase. But authors believed that these phenomena were due to high churners leaving early and only low churners staying from the original cohort as time passed. To test this hypothesis, they incorporated constant retention rate into the assumptions of their model. They assumed that retention rates followed geometric distribution. The geometric distribution shown the figure 1.3a was too simplistic and didnt́ account for variability in customers. Thus, to account for heterogeneity in retention rates across customers and respective cohorts, they incorporated instead beta geometric distribution with $\alpha, \beta$ parameters. The general shapes of the distributions with parameters of $\alpha, \beta$ is shown in the figure 1.3b. These shapes can be interpreted as survival rates of different cohorts. Along with survival rate it's necessary to model the frequency of transactions. The authors found that frequencies of transactions followed negative binomial distribution shown in figure 1.3c With models that predict retention rate and frequency of a customer one can predict expected number of transactions until that customer churns. If we can model the price of those transactions we get CLV.




(a) Geometric distribution[11].




(b) Beta-Geometric distribution[12].




(c) Negative binomial distribution [13].

Figure 1.3: Probability distributions used in sBG/NBD model.

The sBG/NBD model assumes that

- A customer's retention rate $1-\Theta$ is constant throughout the relationship with a company. Meaning that it is characterized by the (shifted) geometric distribution with probability mass function and survivor function of
  - $P(T=t|\Theta) = \Theta(1-\Theta)^{t-1}, t = 1, 2, 3, ...$
  - $S(t|\Theta) = (1-\Theta)^{t}, t = 1, 2, 3, ...$
  - Heterogeneity in $\Theta$ follows a beta distribution with pdf[5] of

$$f(\Theta, \alpha, \beta) = \frac{\Theta^{\alpha-1}(1-\Theta^{\beta-1}))}{B(\alpha, \beta)}, \alpha, \beta > 0, \tag{1.2}$$

  where $B$ is the beta function [10, p. 80].

Beta distribution was chosen, because of its flexibility and a good fit to model heterogeneity in churn probabilities. With the given model, we can predict the number of transactions for a given period of time. But we need the monetary values of transactions in order to predict CLV. The gamma-gamma submodel was incorporated for that reason. In the end, we will have two models that look like this:

- Expected number of transactions for a given period is given by

$$E(X(t)|r, \alpha, a, b) = \frac{a+b-c}{a-1}[1-(\frac{\alpha}{\alpha+t})^{r} F_1(r, b; a+b-1; \frac{t}{\alpha+t})], \tag{1.3}$$

  where $F_1$ is the Gaussian hypergeometric function,
  $r, \alpha, a, b$ are the fitted parameters of sBG/NBD model.

- Expected transaction value is given by

$$E(M|p, q, \gamma, m_x, x) = \frac{(\gamma + m_x x)p}{px + q - 1}, \tag{1.4}$$

  where $m_x$ is the average transaction value of a customer,
  x is the frequency of transactions by the customer,
  P,q and $\gamma$ are fitted parameters of the Gamma-Gamma model.

The sBG/NBD model will give the expected number of transactions for a given period, whereas the gamma-gamma model will give the expected monetary value of transactions. Together these two models predict the CLV of the customer for a given period.

[5] probability density function

## 1.2 MBG/NBD model

The so called MBG (modified beta-geometric distribution) model was introduced in 2007 with a slight change to the original BG/NBD model (8). It enables the newcoming users to be inactive after first purchase, thus improving the dropout rate prediction by incorporating those users that have only one transaction. The change is seen in the formula 1.5.

$$E(X(t)|r,\alpha,a,b) = \frac{b}{a-1}(1-(\frac{\alpha}{\alpha+t})^r {}_2F_1(r,b+1;a+b;\frac{t}{t+\alpha})) \tag{1.5}$$

## 1.3 ML models

With the advent of machine learning, ML methods and algorithms have been applied to a variety of problems, especially where predictions are required. The CLV prediction hasn't been an exception. Most of the classical methods require only RFM data on users, but ML methods are distinguished by being able to use all the data available. There is a lot of literature on CLV prediction using decision trees, random forests, and SVMs, but almost all of them are in a non-contractual continuous business setting. To compensate for the lack of research on CLV in a subscription-based setting, I will include studies on predicting churn rates as they are closely related to the problem of predicting CLV. Below are the examples of works that I would like to explore in more detail in further sections. In a work [14], the author explored the predictive power of LSTMs[6] and naive models and came to the conclusion that complex RNNs slightly outperformed naive models. In another paper[15] that researched the predictive power of two-stage hybrid models and naive method as a baseline, a two-stage decision tree model won, with the result of 99.73% accuracy, a significant improvement compared to the 96.74% accuracy of the baseline model. Unfortunately, the data was taken from a business with a non-contractual discrete business setting and can't be applied to a subscription-based business. But I hypothesize that the same level of improvement is possible in a contractual discrete setting since the underlying problem of modeling heterogeneous behavior of users is present in both types of businesses. Also, It was shown in a study predicting churn of cloud service users, that decision trees can have "an acceptably good accuracy" [16, p. 70] in their prediction and thus entice it to be tried on hybrid models. In the following sections, I will describe methods and algorithms of machine learning used in predicting CLV and try to point out their advantages.

### 1.3.1 Decision tree, random forest and gradient boosting

Decision tree learning is a supervised method of learning, which produces classification or regression models. The models can be described as a set of

[6] Long short-term memory

if-then rules, making the model's decisions explainable, whereas other ML methods can be black boxes. Explainability is one of the main advantages of this kind of model and a very powerful tool when used correctly. Thus making it a candidate for our problem of predicting CLV. Below is a simple definition that encapsulates the decision tree - *"A decision tree is a tree where each node represents a feature(attribute), each link(branch) represents a decision(rule) and each leaf represents an outcome(categorical or continuous value)"* [17]. The example of simple decision tree is illustrated in Figure 1.4. The process of training a decision tree involves the idea of finding an attribute that predicts the outcome better than other attributes. The characteristic that describes how much an attribute predicts the outcome is the information gain.

$$IG(T, a) = H(T) - H(T|a) \tag{1.6}$$

, where

$$H(X) = -\sum_{n=1}^{n} p(x_i) * log_2(x_i) \tag{1.7}$$

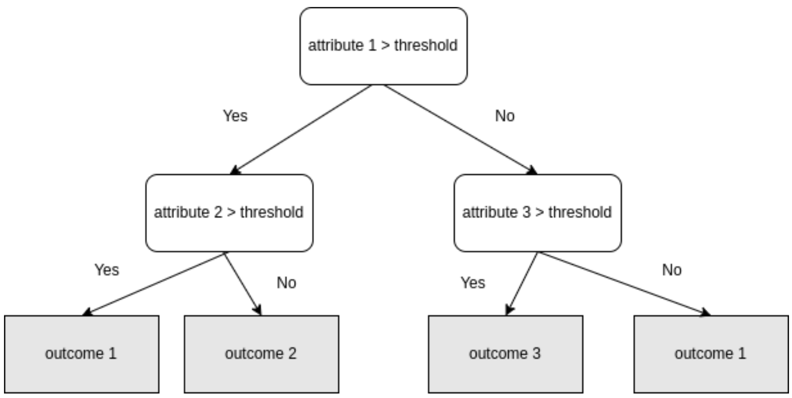
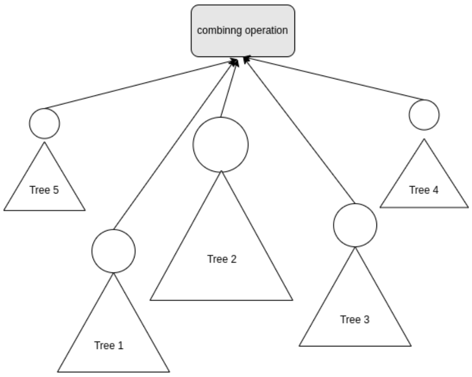
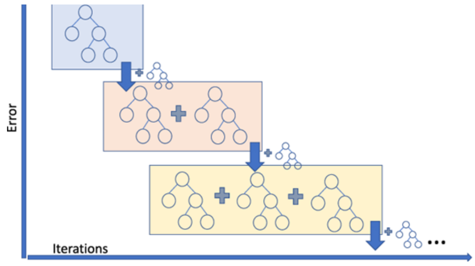


Figure 1.4: Illustration of a decision tree.

Although the decision tree showed good results, it can be unstable with regard to input data. A little change in training data can have a dramatic effect on the outcome of the model. To deal with this problem, the random forest was introduced [18]. A random forest belongs to ensemble methods where multiple weak trees are combined into a big complex model. It's constructed by training multiple decision trees where each of them votes for the final answer and the most common answer is returned. Thus making it more robust to changes in input data. An example of a simple random forest is shown in Figure 1.5a. To further improve on the idea of combining multiple decision trees the gradient boosting method was introduced. It works by iteratively training decision trees and improving on the errors that were made by the previous tree. In Figure 1.5b the illustration of training with gradient boosting methods is shown.




(a) Illustration of a random forest.




(b) Illustration of a gradient boosting method [19].

Figure 1.5: Illustrations of a random forest and gradient boosting tree.

In a 2008 study, the random forest was used in predicting the churn of Belgian newspaper subscribers [20]. Authors mainly focused on SVMs[7] as classifiers with RBF[8] as kernel, but after evaluating competing models, they concluded that random forest consistently outperformed both SVM and logistic regression in accuracy, AUC[9], and top-decile lift. This study is the only one to test random forests on newspaper subscribers and thus is the fittest for our purposes. Unfortunately, there are no studies comparing sBG/BB model with ML techniques in a contractual discrete setting. Nevertheless, there are studies comparing several ML methods between themselves in their predictive power of churn and CLV. For example, In a 2006 paper, the authors compared the predictive power of random forests and deep neural networks on churn rates [21]. The data was pulled from a Taiwanese telecom company with 160K subscribers and had to be oversampled to a smaller dataset due to the low churn rate. The results showed that random forest and neural networks showed decent results of 98% hit-rate in the top-decile lift of churners[21, p. 521]. Thus proving those methods to be effective. The NN[10] model showed results on par with random forest, even though it consisted of only one hidden layer with 18 neurons. Similarly, in the 2012 paper, the random forest outperformed Markov chain and logistic regression models in churn prediction of tv service subscribers [1, p. 182]. And there other studies like [22], [23], [24] where random forest performed on par with others ML methods or better Decision tree and its modifications as a random forest are very powerful tools as shown in previously mentioned papers. Its advantages include explainability, an abundance of research, lots of implementations for various usages, and relative ease of use. Although decision tree and random forest fail to model

[7] Support vector machine
[8] Radial basis function
[9] Area under the ROC Curve
[10] Neural Network

CLV and churn into the far future as other methods such as Markov chain, logistic regression, and probabilistic models that tend to perform better in the long term. I suggest comparing it with sBG/BB and state-of-the-art deep neural networks.

### 1.3.2 Deep NNs

Artificial neural networks (ANN or NN) have been very popular in the last decade, due to their success in modeling complex processes. It's a machine learning method that imitates the biological brain. It consists of many neurons that are weak learners, but in aggregate they can create a complex model. In Figure 1.8 the similarities are shown between biological neuron and artificial neuron. The individual neurons are combined into a network as shown in 1.6, where each neuron takes outputs from previous layer, computes the weighted sum of outputs and passes the result through non-linear activation function. The pass through activation function is the key element which enables the neurons to build the knowledge on top of each other. The learning process

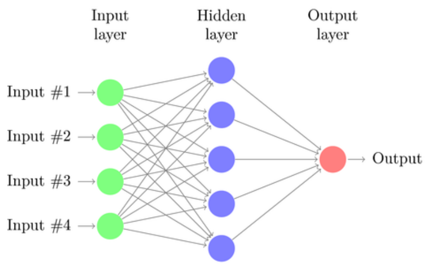



Figure 1.6: A simple neural network[3].

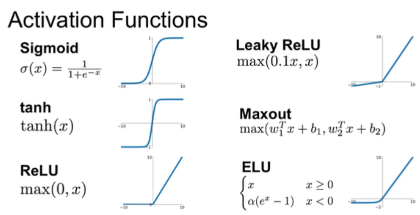



Figure 1.7: Most popular activation functions [4].

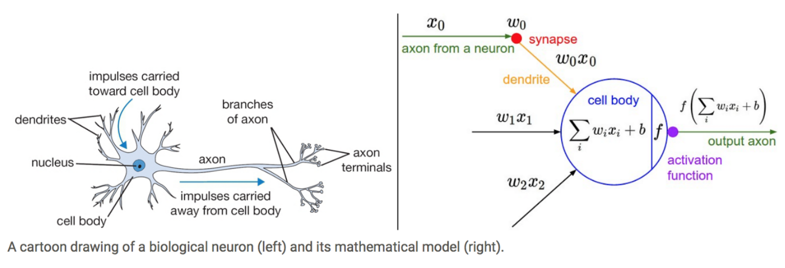



Figure 1.8: A comparison of a biological neuron and an artificial neuron[5].

is done with the backpropagation method. NNs and their modifications like RNN have shown great results in modeling and prediction throughout a wide range of problems.

Naturally, NNs were used to predict CLV and churn rates of customers, but there are very few studies on this topic. Even fewer studies are on CLV and churn prediction in contractual discrete settings, which further limits the comparison with other methods. In a previously mentioned study comparing simple RNN, LSTM, and GRU[11], all the models performed fairly well around 75% hit-rate in the top-decile lift [14, p. 36]. Unfortunately, the NN methods in this paper were not compared with the random forest and sBG/BB model, as this study is the only one I could find on CLV prediction with NNs in a contractual setting. Although there are no other papers on the topic, there are few on churn prediction in a contractual setting. One of them compared the predictive power of NN, decision tree, and logistic regression on churn prediction of wireless company's customers. Even though the NN model consisted of one hidden layer with up to 40 neurons, it outperformed other models [25]. Similarly, these papers [26], [27], [21] showed that NNs beat logistic regression, decision tree, and RBF in churn prediction. But the lack of more sophisticated deep NN models in almost all reviewed papers complicates the process of comparison, as modern deep NNs might show better results. Therefore, I suggest comparing modern NN architecture models with other methods in CLV and churn prediction.

[11] Gated recurrent unit

Chapter 2

# Analysis

## 2.1 Dataset

For the purpose of this work, ideally, I would test predictive models on a real dataset from subscription-based news media companies. Unfortunately, I couldn't find such datasets in open access. Generally, there are very few transactional datasets available that come from subscription-based services. Eventually, I chose the dataset from the Kaggle competition from KKBox company - a Taiwanese music streaming service operating in Southeast Asia. They organized a competition for predicting churn probabilities of users based on their listening behavior and transactional data. Although it's not a news-related dataset, nor CLV prediction competition, it's suitable for the purpose of this work, because KKBox is a subscription-based service, hence transactions are of contractual continuous nature, and churn prediction is closely related to CLV prediction. Below is the table 2.1 with all available datasets for our use. To simulate users purchasing behavior in the news media companies, only transactional data will be needed.

Table 2.1: Datasets provided by KKBox

| **Dataset** | **Description** | **#Lines** | **Size** |
|---|---|---|---|
| transactions.csv | Transactions of users from 2015-01-01 until 2017-03-31. | 22.9M | 2.13GB |
| user_logs.csv | Daily user logs describing listening behaviors of a user, collected from 2015-01-01 until 2017-03-31. | 258M | 16.1GB |
| member.csv | Users' personal information. Birthday, gender, city, etc. | 6.7M | 428MB |

Table 2.2: Attributes of users transactions

| Attribute | Type | Description |
|---|---|---|
| msno | string | Anonymized user ID |
| payment_method_id | int | Payment method ID |
| payment_plan_days | int | Length of membership plan in days |
| plan_list_price | float | Price of the plan in New Taiwan Dollar (NTD) |
| actual_amount_paid | float | Amount paid by the user in New Taiwan Dollar (NTD) |
| is_auto_renew | boolean | Indicates if subscription auto-renews after expiration |
| transaction_date | date | Date of the transaction |
| membership_expire_date | date | Expiration date of the subscription |
| is_cancel | boolean | Indicates whether or not the user canceled the membership in this transaction. |

Let's start by analyzing the demographics of customers. There are 6,7M users in the "members.csv" dataset. Gender-wise it's very uniform, with slightly more male customers. Age-wise there were a lot of meaningless values which I sorted out by taking only ones that were positive and less than 100. Most of the users lied between the ages of 15 and 40 with a mean age of 29.7 as shown in Figure 2.1. The majority of users were from 6 cities out of 22 as seen from Figure 2.1. I speculate that those are the 6 biggest cities in Taiwan and China.

The second file from dataset contains metadata on users' transactions, which were made to buy subscription plan or prolong existing ones. In Figure 2.2 there are 5 different kind of users that were taken from dataset. User 5 is the regular user, that prolongs their plan every month, whereas user 1 bought 4-month plan and changed to monthly plans.

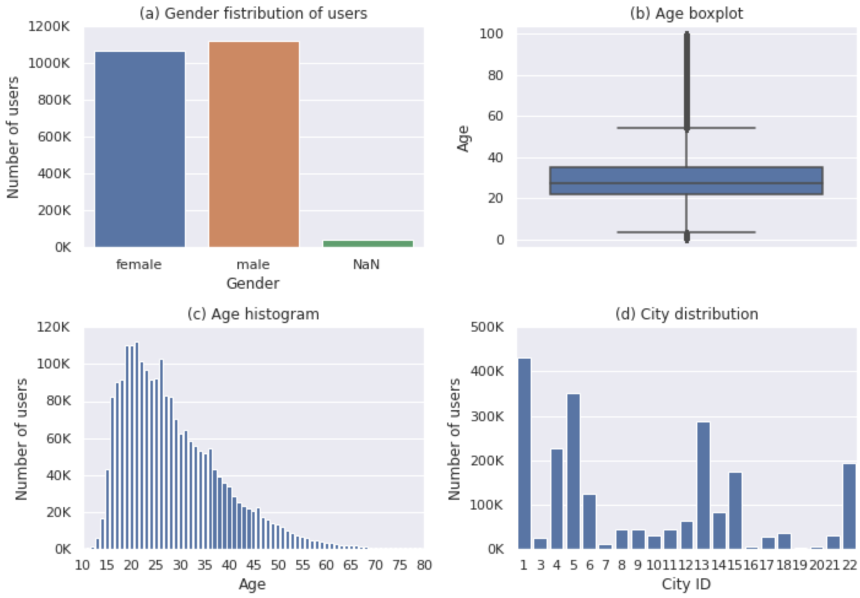



Figure 2.1: Users' distribution.

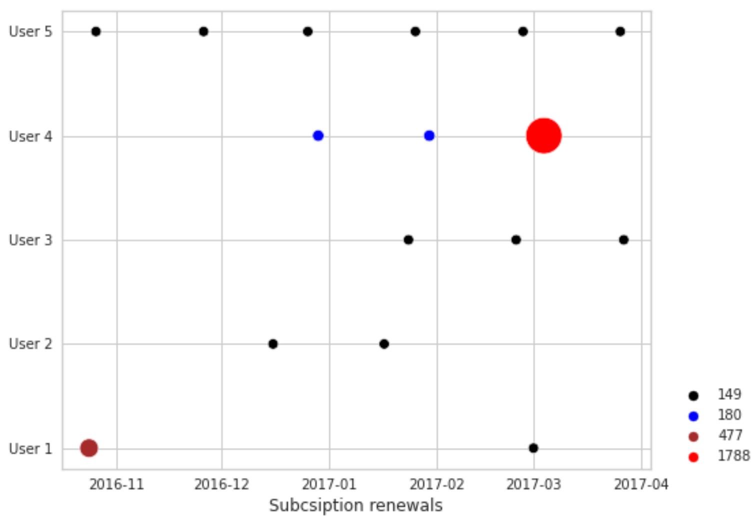



Figure 2.2: Example of users' transactions.

Figure 2.3 shows the distribution of transactions by date, day of the month, and day of the week. It's clear that there's a big spike in activity in the last days of the month, due to the expiration of the subscription. But apart from that, the distribution of transactions is uniform across days of the month and days of the week, with slight negligible variance. When it comes to users' spending, the Pareto principle doesn't quite hold in our case. 45.7% of users account for 80% of the revenue, as shown in figure 2.4. But still, detecting the top contributing users would lead to calculated decision-making in  expenses. From the above-mentioned transactional dataset, a truncated version of 5M rows was used to train and test models, as it contains all the needed information to get the RFM table of users and it is small enough to run on my laptop. The dataset consists of detailed information about user's transactions collected from 2015-01-01 till 2017-02-28. Below is the metadata description of the dataset.

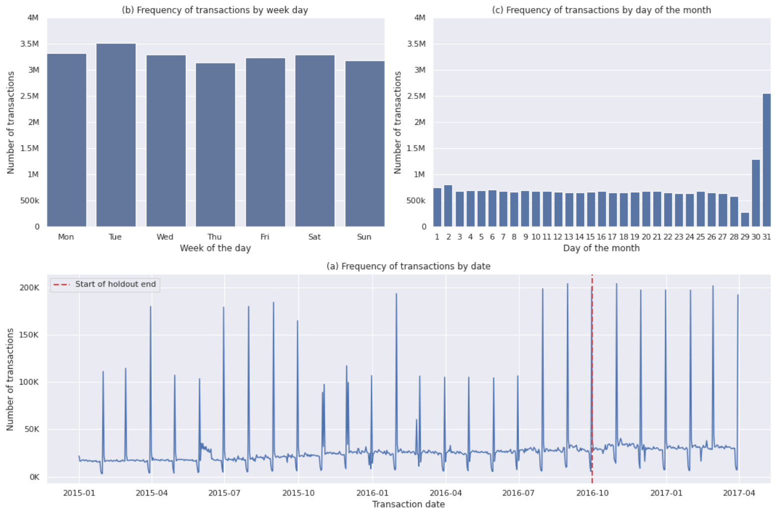



Figure 2.3: Users' gender, age, and city distributions.

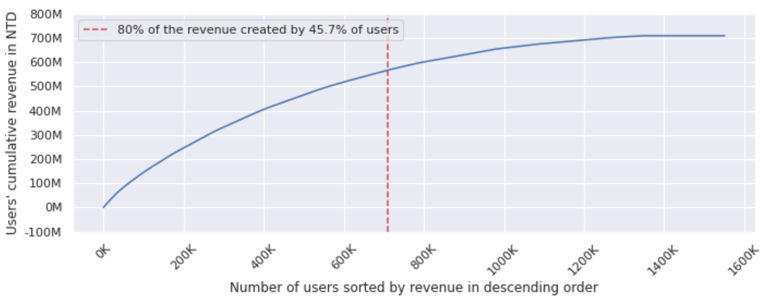



Figure 2.4: Distribution of users revenue sorted in descending order.

## 2.2 Open-source extension

While searching for an open-source project suitable for CLV predicting extension, I found REMP[12] - a software suite developed by Deník N[13] to help with readers' engagement and monetization of content. The project was funded by Google's DNI fund [28] and became a success story, with only 3 years needed to become profitable for Deník N. The suite includes a variety of modules, designed primarily for fast iteration and ease of use. Below is the table with the main modules of REMP.

Table 2.3: REMP modules

| **Name of the module** | **Functionality** |
|---|---|
| CRM | The backbone of the suite, which handles subscribers and subscriptions |
| Mailer | Compose emails, newsletters, layouts and templates |
| Campaign | Campaign management, a banner manager with A/B testing, geo-targeting, device targeting etc. |
| Beam | Segments, Rules, and Events |
| SSO | Single sign-on access to all tools of REMP |
| Pythia | The machine learning component for higher conversions |

The project is built with Nette[14] framework for frontend, MySQL[15] for database, and Redis[16] for the persistence layer. It offers great modularity and ease of extension. For the purpose of this work, the widget will be implemented in CRM module that shows the past and future CLV of a customer and a script in the Pythia module, that will calculate CLV data from user transactions.

[12] `https://remp2030.com`
[13] `https://denikn.cz`
[14] urlhttps://nette.org/en
[15] urlhttps://www.mysql.com
[16] urlhttps://redis.io

Chapter 3

# Implementation

## 3.1 Tools

### 3.1.1 Hardware

All the experiments were performed on the server provided by the faculty with the following specifications:

- **CPU** - Intel(R) Xeon(R) Gold 6254 CPU @ 3.10GHz
- **RAM** - 128 GB DDR6
- **GPU** - 2x NVIDIA A100 40 GB VRAM
- **OS** - Ubuntu 20.04.4 LTS

### 3.1.2 Software

Below is the software I used both for experimenting, training the models, and implementing the extension for the open-source media project. I chose these tools by the criteria of widespread adoption, comprehensive documentation, simplicity, and my knowledge of tools.

- **Python** [17] - is a high-level programming language, highly adopted in the ML community and ideal for fast prototyping. I used version 3.6 of python and pip [18] package management system for libraries.
- **Jupyter notebook** [19] - is the interactive web interface for python that provides repl capabilities and vastly increases the speed of prototyping and experimenting.
- **Lifetimes** [20] - is the framework for predicting CLV with probabilistic models developed by Spotify in 2015.
- **Scikit-learn** [21] - is the machine learning library with
- **Tensorflow** [22] - is a library for machine learning, specifically for implementing neural networks.
- **Keras** [23] - is the high-level python interface that provides API for creating neural networks. It was primarily designed to use tensorflow as backend, but can support other libraries as well.

[17] https://www.python.org/
[18] https://pip.pypa.io/en/stable/
[19] https://jupyter.org/
[20] https://lifetimes.readthedocs.io
[21] https://scikit-learn.org
[22] https://www.tensorflow.org/
[23] https://keras.io

- **REMP** [24] - is open-source software designed for the monetization of the content of media houses. It was created by a media house DenikN in 2018 to help small media teams get access to content monetization software that was otherwise unavailable to them.
- **PHP** [25] - is a programming language designed for web development and was created in 1994. It's widely adopted and has a long supported set of libraries and frameworks.
- **Javascript** [26] - is a high-level programming language designed for web development.
- **Google charts** [27] - is a javascript library designed by Google to create graphical charts.

[24] https://remp2030.com
[25] https://www.php.net
[26] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[27] https://developers.google.com/chart

## 3.2 Model Selection

As discovered in the first part of this work, there are three classes of models in the CLV prediction: probabilistic models, decision trees & random forests, and neural networks. To implement the extension the best model needs to be chosen between those three. Moreover, measuring the performance of those models requires a real dataset, preferably from news media companies, as it will best suit the use case of REMP.

### 3.2.1 Preprocessing

To simulate users' purchasing behavior in the news media companies, only transactional data will be needed. The transactional data then will be transformed into RFM table and fed into models for training.

Table 3.1: Attributes of transactions.csv dataset used for training

| Attribute | Type | Description |
|---|---|---|
| msno | string | Anonymized user ID |
| transaction_date | date | Date of the transaction |
| actual_amount_paid | float | Amount paid by the user in New Taiwan Dollar (NTD) |

Before training models on transactional data the dataset will be preprocessed and divided into training, validation and test sets. The preprocessing involves filtering the needed data, transforming it in accordance with respective model and split into sets. The transformation methods in this work will include conversion of transactional data to RFM table and changing features' distribution.

The RFM table is obtained by tools in lifetime library. It should be noted that there's an additional T column in the RFM table, the age of the customer, as it's required by the probabilistic model to function properly. So technically, we will extract RFMT values from the transactional dataset. Below are code snippets used in preprocessing of the dataset.

```
RFM = summary_data_from_transaction_data(
                df,
                customer_id_col='msno',
                datetime_col='transaction_date',
                monetary_value_col='actual_amount_paid',
                freq='M')
```

Table 3.2: RFM table example

| msno | frequency | recency | T | monetary_value |
|---|---|---|---|---|
| +++FOrTS7a... | 1.0 | 0.000000 | 5.946734 | 0.0 |
| +++IZseRRi... | 2.0 | 11.006386 | 15.967474 | 1693.5 |
| +++hVY1rZo... | 5.0 | 3.942586 | 3.942586 | 99.0 |
| +++l/EXNML... | 21.0 | 25.955358 | 25.955358 | 149.0 |
| +++snpr7pm... | 27.0 | 25.955358 | 25.955358 | 149.0 |

After the preprocessing the RFM data will look like in the table 3.2 and be ready to feed into predictive models. The change in features distribution is obtained by two methods:

- Normalization

  The ML methods, particularly neural neworks work better if the data is normalised. The most popular normalisation method is standardisation, which maps mean of features to 0 with standard deviation of 1, as shown in formula.

$$x' = \frac{x - \overline{x}}{\sigma}$$

- Box-cox transformation

  The box-cox transformation is the exponential conversion that is intended to get features with a distribution that best approximates the normal distribution. This transformation will be used on target CLV values when training the neural networks as work best if the target value is of normal distribution.

$$y(\lambda) = \begin{cases} \frac{\lambda^2 - 1}{\lambda}, & \text{if } \lambda \neq 0 \\ log(y), & \text{if } \lambda = 0 \end{cases}$$

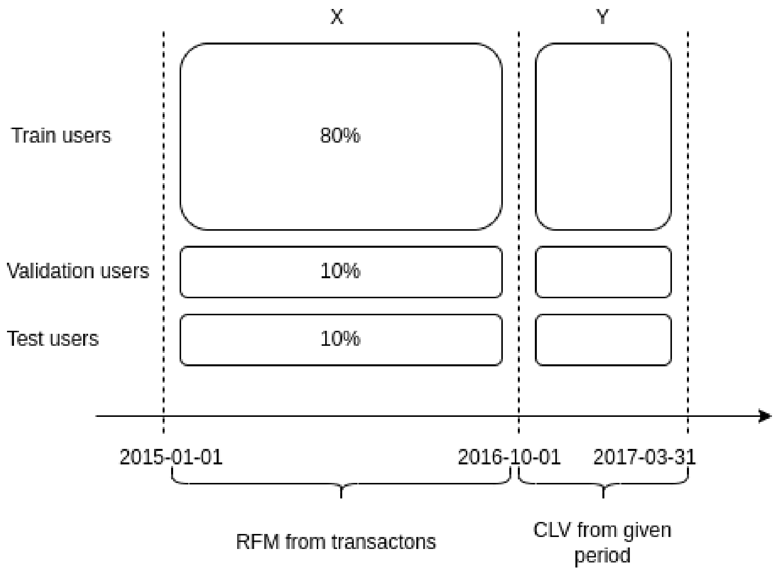



Figure 3.1: Split of RFM table into training, validation and test sets.

Below are the steps of preprocessing of transactional dataset:

1. User sets

   a) Filter out users with less than 5 transactions in the calibration period.

   b) Filter out users that don't have transactions in calibration or hold-out period.

   c) Filter out users that paid for greater than monthly subscriptions (e.g. quarterly, half-year, yearly and etc.).

   d) Randomly select 10% of users for validation, 10% of users for testing and the rest 80% for training.

2. Transaction sets

a) Select attributes msno, transaction_date, actual_amount_paid.

b) Divide transactions into calibration and holdout periods, with the start of the holdout period being 2016-09-30 (It's the end of the month so that subscription fee for the next month is counted in).

c) Filter out transactions that have zero amount paid (free subscriptions).

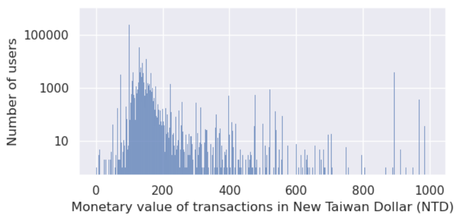
d) Convert transactions in the calibration period into the RFM table.

e) Convert transactions in the holdout period into CLV values.

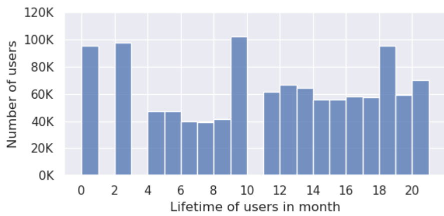
f) Divide the result X, and y values into training, validation, and testing sets according to resultant user sets in step 1.d.

3. Normalization

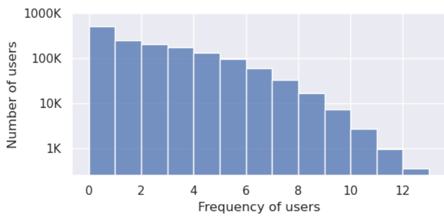
a) The mean and standard deviation is first calculated on training set and then applied to all sets, to avoid information leakage from training set.



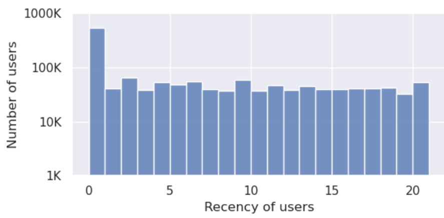

(a) Monetary values from RFM




(b) T values from RFM




(c) Frequency values from RFM




(d) Recency values from RFM

Figure 3.2: RFMT table distributions

If we inspect the final RFM data of the calibration period, as expected the majority of customers have few transactions as shown in the log-scaled histogram of frequency in Figure 3.2c. The vast majority of users have a frequency between 1 and 6, which can be attributed to users buying a year plan or a 6-month plan. The recency histogram shows, that a lot of customers in the past 2 years have churned, but still there's quite a big active cohort of

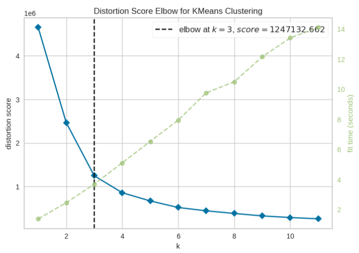



Figure 3.3: Distortion score elbow for k-means clustering

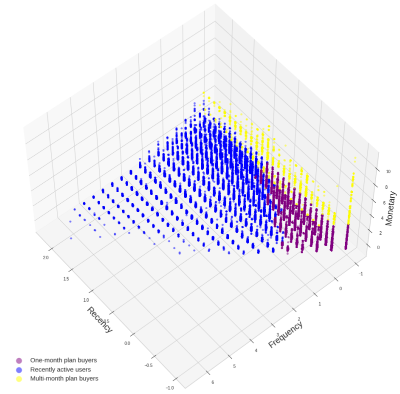



Figure 3.4: 3D Scatter plot of clusters in RFM

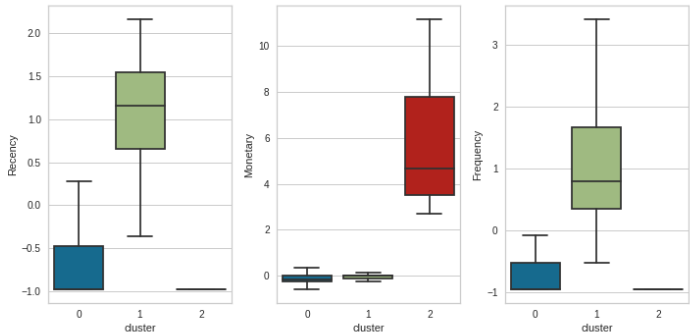



Figure 3.5: Boxplot of RFM values for each 3 clusters

users who have bought subscriptions in the last month. The monetary values of transactions, tell us a story about two groups of users. The first group is the users that pay for the monthly subscriptions and their transaction value lies in the interval from 0 to 300 which resembles the normal distribution. The majority of users belong to the first. But after 400 NTD, there are several spikes, which are sparsely located until 1000 NTD. Those are the users that buy 6-month, 12-month, and 24-month plans. To test this hypothesis I used k-means clustering to find out what groups of users exist. As shown in figure 2.9 the elbow method showed that there are 3 optimal clusters that divide users into separate groups according to their RFM values. Figure 3.4 illustrates the 3 groups of users, where yellow is the group of users that buy multi-month subscription plans, whereas the blue and purple groups buy more frequently. The k-means algorithm also divided users that have been active recently and others that probably churned: purple group is recently active users and the blue group is churned users.

### 3.2.2 Model comparison

In this chapter, I will compare predictive models on the KKBox's preprocessed dataset and discuss performances as well as training details.

#### 3.2.2.1 Methodology

To compare the models the appropriate metrics for regression problems need to be used. Below are the metrics I used in comparing models. Each metric has its unique characteristics and interpretation.

- **MAE** [28] - is a metric sensitive to outliers, but shows the error in the same scale as the measured data.

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|CLV_i - \widehat{CLV_i}|$$

- **MAPE** [29] - is a metric that calculated the percentage difference of predicted data, which simplifies the interpretation of results.

$$MAPE = \frac{1}{N}\sum_{n=1}^{N}|\frac{CLV_i - \widehat{CLV_i}}{CLV_i}|$$

- **RMSE** [30] -

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(CLV_i - \widehat{CLV_i})^2}$$

- **Top-k% hit-rate** - is the sensitivity on the top k% of users by CLV, where sensitivity is measured given by

$$Sensitivy = \frac{TP}{TP + FN}$$

  It's the most insightful metric for the purpose of this work as it shows how many of the high-value customers are identified by the model.

With these metrics, the models can be compared on their ability to predict CLV on an individual level as well as cohort level. The RMSE and MAPE metrics are appropriate to test the performance of models on individual levels, while top-percentile hit rates show us how well models do in finding the most valuable users.

The user dataset is split into training, validation and testing sets.

[28] Mean absolute error
[29] Mean absolute percentage error
[30] Root mean squared error

#### 3.2.2.2 Probabilistic model

To run sBG/BB model on our dataset we will use the lifetimes[31] python framework, which has primarily been implemented for CLV prediction by Shopify[32]. It offers a variety of models, as well as useful tools which will be used in our analysis. The process of predicting CLV consists of training shifted beta geometric model to predict the probability of churning of a customer and the number of transactions that will be made in the future and training gamma-gamma model to predict the monetary value of those transactions. The process is shown in the figure below.

[31] `https://lifetimes.readthedocs.io`
[32] `https://www.shopify.com`

The training of the model is easily accomplished in scikit-style, as shown below.

```
from lifetimes import BetaGeoFitter

bgf = BetaGeoFitter(penalizer_coef=0.01)
bgf.fit(df_user_RFT['F'], df_user_RFT['R'], df_user_RFT['T'])

ggf = GammaGammaFitter(penalizer_coef = 0.004)
ggf.fit(
    RFMT_cal_holdout_gg['frequency_cal'],
    RFMT_cal_holdout_gg['monetary_value_cal'])
```

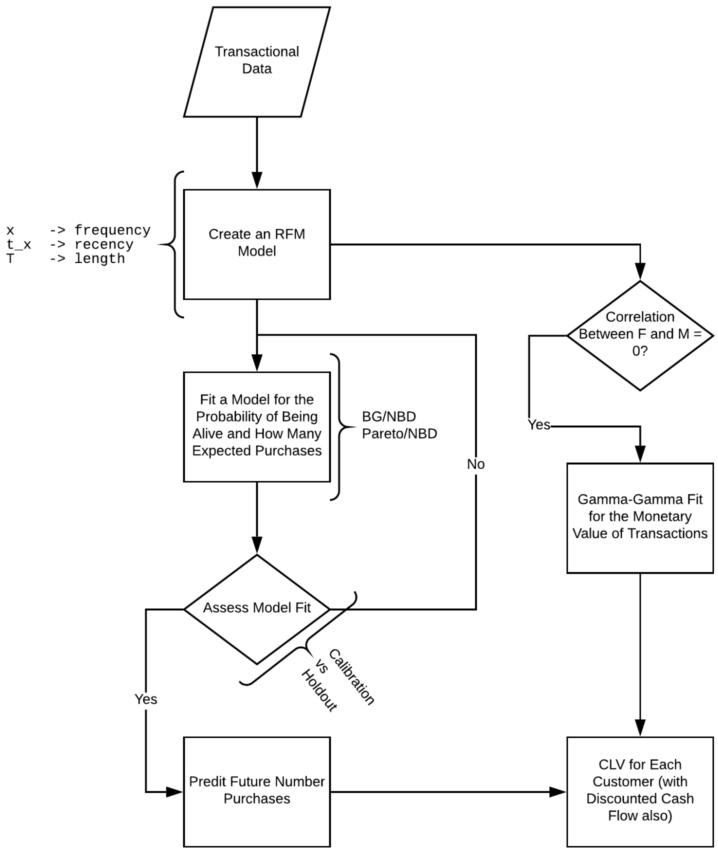
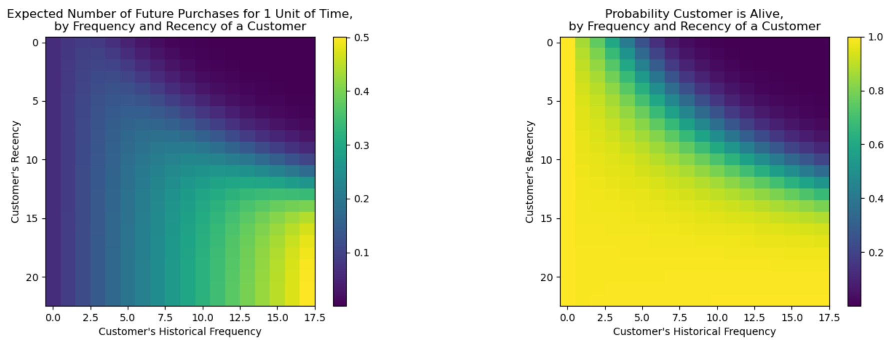



Figure 3.6: The flowchart of predicting CLV in lifetimes framework

There is only one hyperparameter to tune in both beta geometric model and gamma-gamma model - it's penalization coefficient that is applied to l2 norm on the parameters. I experimented with a wide range of values and found that $1 \times 10^{-6}$ for the beta-geometric model and $1 \times 10^{-5}$ for the gamma-gamma model work best. The final learned parameters:

- sBG/NBD model - a: 0.07, $\alpha$: 16.87, b: 8.71, r: 2.56
- Gamm-Gamma model - p: 48.08, q: 15.84, v: 43.91




(a) Purchase probability of customers. (b) Probability of customer being alive.

Figure 3.7: sBG/NBD models trained distributions.

I used four graphs First to determine the preliminary performance of the model we will employ four graphs that ensure that our model is on the right track as mentioned in a video lecture by Peter Hader [29].

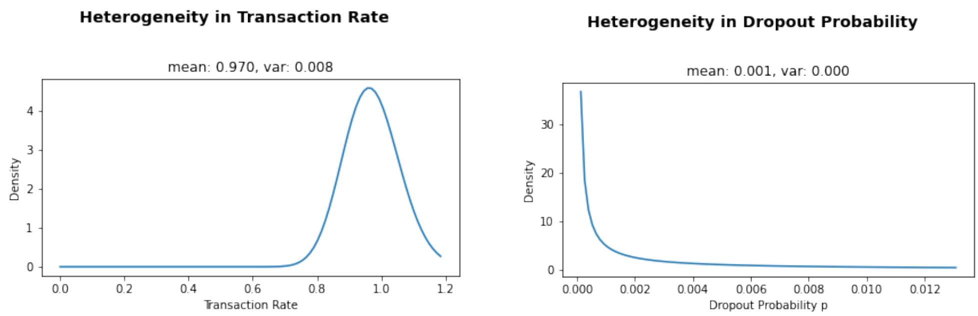
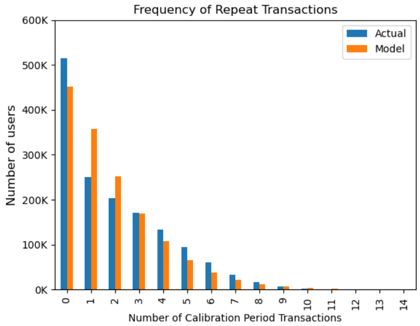
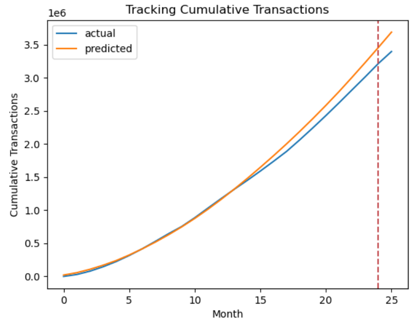



Figure 3.8: Learned distributions of transactions and dropout rates of users.

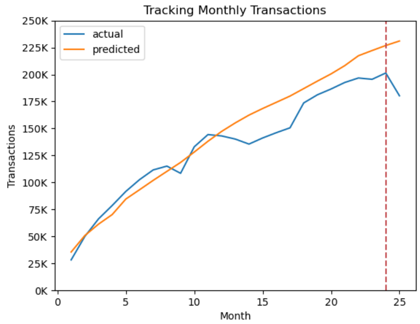
(a) sBG/NBD predictions of transaction frequencies in training period




(b) sBG/NBD predictions of cumulative transaction frequencies




(c) sBG/NBD predictions of monthly transaction frequencies in training period




(d) sBG/NBD predictions of monthly transaction frequencies in holdout period

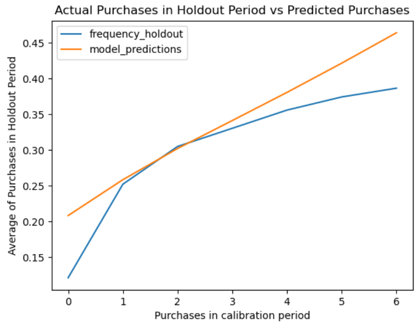



Figure 3.9: Monthly transactions and performance of holdout period

As seen in figure 3.9b, our sBG model's prediction on cumulative transactions is close to real data, although the model predicts higher revenue than in reality. The overestimation is even starker in the holdout period.

#### 3.2.2.3 Gradient Boosting Regressor

The second category of CLV predictive models is decision trees and random forests. Among those techniques, gradient boosting is the benchmark model in Kaggle[33] competitions, as well as in industry. The most popular and widely adopted implementation of gradient boosting is the XGBoost, but for the sake of simplicity and ease of use, I used scikit-learn's implementation. Before feeding the RFM table into the model, it was standardized, as it's always useful to do so in the models working with thresholds.

[33] `https://www.kaggle.com`

The training of the model is done in two lines as shown below.

```
from sklearn.ensemble import GradientBoostingRegressor

reg = GradientBoostingRegressor(
                loss= 'mae',
                random_state= 0,
                learning_rate= 0.01,
                n_estimators= 100)
reg.fit(X_train, y_train)
```

The hyperparameter optimization by grid search on validation set yielded results shown in the figures below.

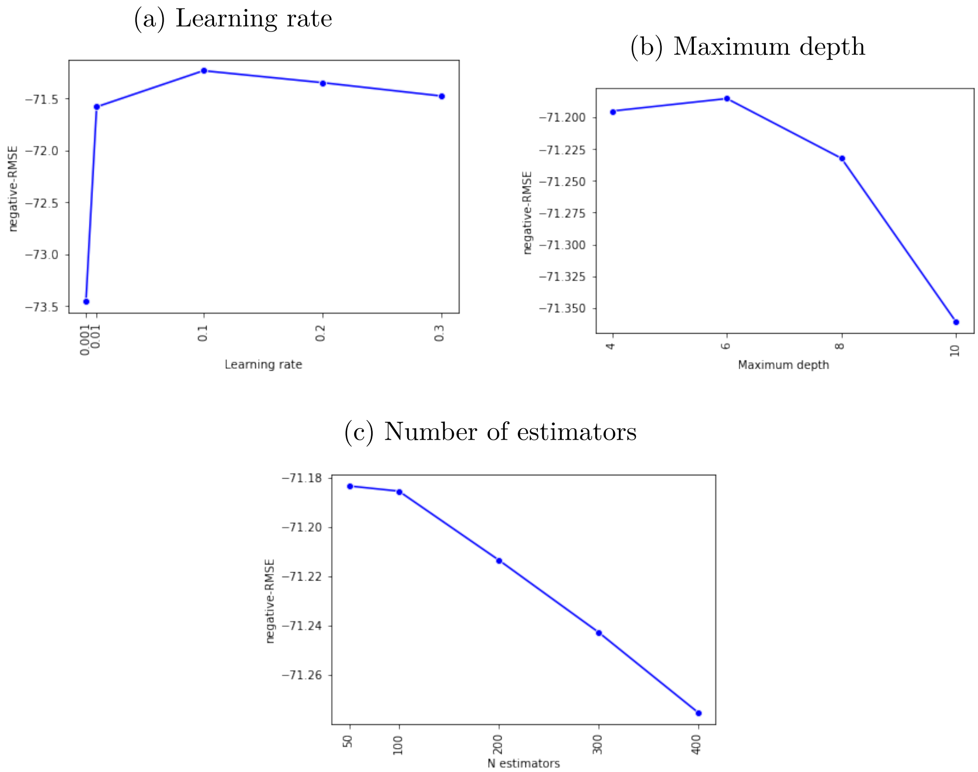



Figure 3.10: Hyperparameter optimization of Gradient Boosting Regressor

The final model was trained with hyperparameters shown in table 3.3.

Table 3.3: The hyperparameters of gradient boosting regressor

| **Hyperparameter** | **Value** |
|---|---|
| learning rate | 0.1 |
| max_depth | 6 |
| n_estimators | 50 |
| loss | squared_error |

#### 3.2.2.4 Neural Network

The NN model was implemented with the Tensorflow[34] library. Tensorflow was released by Google in 2019 and since then has been widely used in research and industry. It's highly optimized to run on GPUs and offers a wide range of functionalities. I chose a simple architecture for the purpose of this work, as shown in table 3.4. The standardized RFM table is fed into the first layer and CLV for the holdout period is computed in the output layer. At first, I used all the users for training but quickly learned that the model wasn't learning and the errors were high. It was due to the distribution of the target value as shown in figure 3.11. It was caused due to the majority of the users from the past 2 years churning or buying multi-month plans. For that reason, I had to train the data only on positive target values. Although it would create a bias towards non-churned users, it was necessary for the purpose of this work to do so. I suggest using this model as a multi-stage predictive model, by first classifying the churned uses out and then applying the following model.

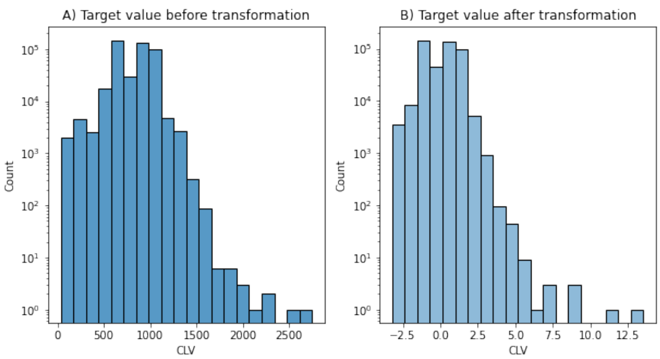



Figure 3.11: The histogram of target values in the training set before and after box-cox transformation.

[34] https://www.tensorflow.org

The training of the model was done with batch normalization and a dropout of 0.2 for each layer except the last. The target value was the total revenue in the test period.

Table 3.4: The architecture of the simple neural network

| **Layer** | **Input** | **Activation** | **Output** |
|---|---|---|---|
| Fully connected | 3x1 | ReLU | 128x1 |
| Fully connected | 128x1 | ReLU | 64x1 |
| Fully connected | 64x1 | ReLU | 32x1 |
| Fully connected | 32x1 | Linear | 1x1 |

Table 3.5: The hyperparameters of simple neural network

| **Hyperparameter** | **Value** |
|---|---|
| optimizer | Adam |
| learning rate | 0.1 |
| batch size | 256 |
| epcohs | 100 |
| dropout rate | 0.2 |

Table 3.6: The architecture of the complex neural network

| **Layer** | **Input** | **Activation** | **Output** |
|---|---|---|---|
| Fully connected | 3x1 | ReLU | 256x1 |
| Fully connected | 256x1 | ReLU | 128x1 |
| Fully connected | 128x1 | ReLU | 64x1 |
| Fully connected | 64x1 | ReLU | 32x1 |
| Fully connected | 32x1 | Linear | 1x1 |

Table 3.7: The hyperparameters of simple neural network

| **Hyperparameter** | **Value** |
|---|---|
| optimizer | Adam |
| learning rate | 0.1 |
| batch size | 256 |
| epcohs | 100 |
| dropout rate | 0.2 |

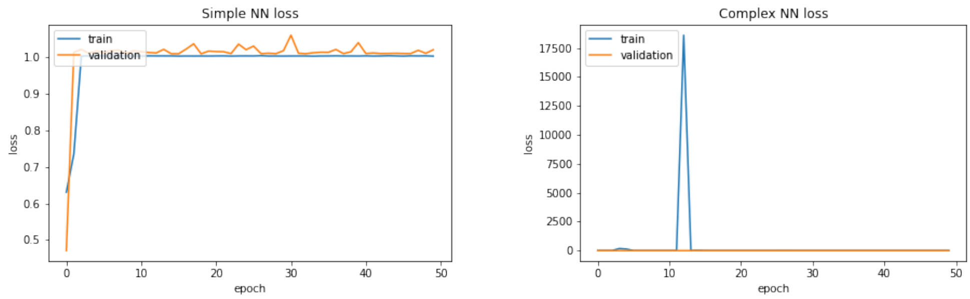



Figure 3.12: Loss function of simple and complex neural network models on train and validation sets

#### 3.2.2.5 Results

The results in table 3.8 shows that all models were not accurate on the individual level, but gradient boosting was extremely good at spotting the top 20 percentile users, which plays a more important role in marketing resource allocation than the individual prediction. Although the probabilistic models weren't accurate on individual level both models outperformed in top 5 percentile. All in all, the winner - gradient boosting regressor, beat other models in most metrics and most importantly was successful in identifying the most valuable users with great accuracy. It also has advantages such as explainability and scalability, which adds to its favor when used in real-world use-cases. The neural network models performed poorly. I attribute it to my inability to train them properly. With better approach I expect neural networks to work better compared to my results.

Table 3.8: Performance of models on test data

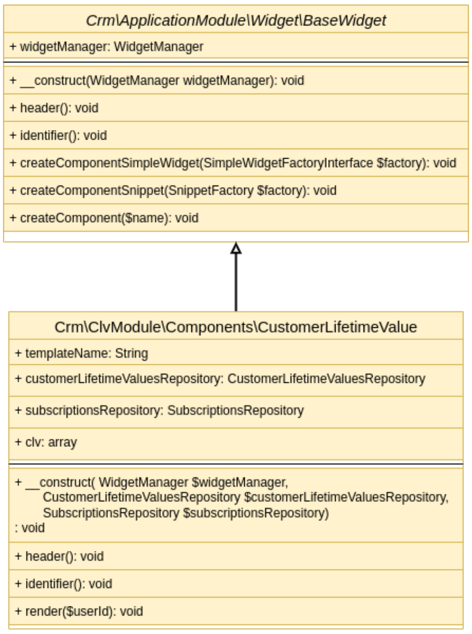
| **Model** | **MAE** | **MAPE** | **RMSE** | **Top-5% hit-rate** | **Top-10% hit-rate** | **Top-20% hit-rate** | Epoch | Time |
|---|---|---|---|---|---|---|---|---|
| sBG/NBD | 21753.07 | 28.8 | 22183.31 | **0.4** | **0.35** | 0.3 | | 16.5s |
| MBG/NBD | 21743.48 | 28.79 | 22172.59 | **0.4** | **0.35** | 0.3 | | 14.67 |
| Gradient Boosting Regressor | 58.94 | 0.13 | **17063.84** | 0.14 | 0.31 | **0.79** | | |
| **Gradient Boosting Regressor + scaling** | **58.24** | **0.12** | 17832.51 | 0.32 | **0.35** | **0.79** | | 8.02s |
| Simple NN | 798.24 | 1.00 | 680361.51 | 0.06 | 0.17 | 0.23 | 20 | 574.12s |
| Complex NN | 798.25 | 1.00 | 680374.97 | 0.06 | 0.17 | 0.23 | 20 | 698.07s |

## 3.3 CLV predicting widget

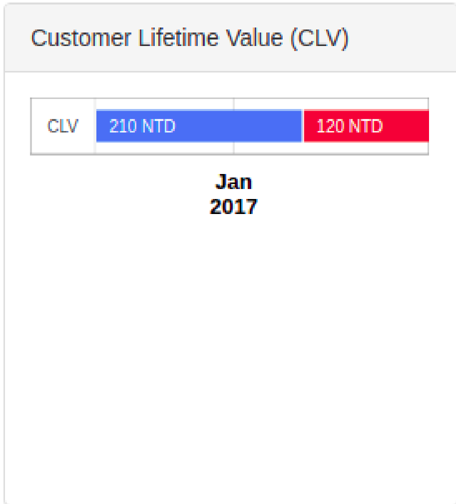
The widget will be a part of the CRM skeleton of REMP. The CRM offers an interface to extend the functionalities in a simple and modular way. The CRM is implemented in PHP programming language and primarily built with the Nette framework. To add any extension following steps need to done:

- Create folder app/modules/DemoModule
- Create class Crm/DemoModule/DemoModule extending Crm/ApplicationModule/CrmModule within the module folder.
- Register the module in app/config/config.neon file

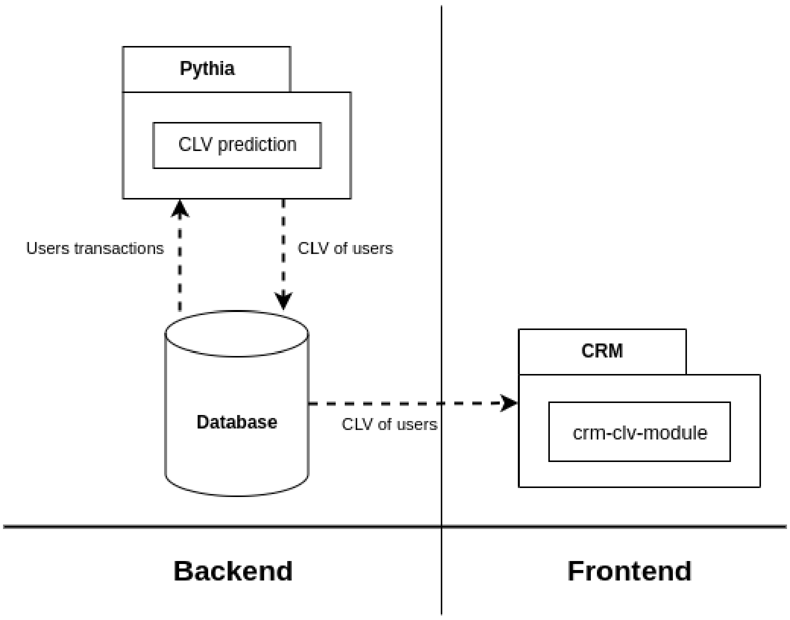
I used the existing CLV module's structure as a sample and modified it for the purposes of this work. The previous CLV module computed the past lifetime value and put it in a quartile chart, whereas my implementation will include the prediction of CLV into the future as shown in Figure 3.13b. For a widget to work properly the CLV predicting script needs to be run periodically to update data in database. The overall design of the system is shown in Figure 3.13c




(a) The UML diagram of the widget class.




(b) The CLV predicting widget




(c) Design of extension.

Figure 3.13: The widget design and implementation

# Conclusion

All models are wrong, but some are useful

—Goerge Box

In the first part of this work I introduced the concept of customer lifetime value and how it's defined by the business setting it's used in. Also, I summarized the methods and results of various predictive models on CLV and came to the conclusion that there are three main categories of methods used in CLV prediction: probabilistic, decision trees & random forests, and neural networks. Out of those three methods the gradient boosting model outperformed other models almost in all metrics. Meanwhile, probabilistic methods proved to be simple, easy-to-use, interpretable and computationally cheap. In the end, the grdient boosting model was incorporated into REMP's Pythia module, responsible for CLV prediction. Furthermore, the widget was implemented to make it possible for CRM users in media houses to see the customer's CLV and make decisions based on that. Therefore, the goal of this thesis was fulfilled, although not to the fullest extent. Hopefully, it will be useful for small to medium companies that cannot afford to employ dedicated teams, doing the research and development of software specifically aimed at CLV prediction.

## Further suggestions

The topic of CLV prediction in subscription-based services (a.k.a. contractual continuous business setting) is underresearched and there is a general lack of extensive comparative research on the methods of prediction. While probability models have proved to be the benchmark in non-contractual settings, it's not clear in the literature what methods are the best in a contractual setting. The sBG/BB model performed relatively well on our dataset, but it is limited by the fact that if a customer churns once, it's forever lost for the model, while in real-world customers can cancel and re-subscribe several times. For that reason, I suggest looking into Markov state models and more sophisticated neural network architectures.

# Bibliography

[1] Hwang, H.; Jung, T.; et al. An LTV model and customer segmentation based on customer value: a case study on the wireless telecommunication industry. *Expert Systems with Applications*, volume 26, no. 2, 2004: pp. 181–188, ISSN 09574174, doi:10.1016/S0957-4174(03)00133-7, [Online; visited on September 10, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0957417403001337

[2] Peter Fader, B. H. Probability Models for Customer-Base Analysis. *Journal of Interactive Marketing*, volume 23, no. 1, 2009: pp. 61–69, ISSN 10949968, doi:10.1016/j.intmar.2008.11.003, [Online; visited on August 12, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/S1094996808000108

[3] Ercoli, J. Neural Networks from the beginnings. 2021, [Online; visited on April 21, 2022]. Available from: https://medium.com/mlearning-ai/neural-networks-from-the-beginnings-470fbeab0fc8

[4] Yu, J. Multiple Angles of Arrival Estimation using Neural Networks. 2019, [Online; visited on April 20, 2022], 2002.00541. Available from: https://arxiv.org/abs/2002.00541

[5] Ching, D. The math behind Neural Networks. 2021, [Online; visited on April 21, 2022]. Available from: https://www.analyticsvidhya.com/blog/2021/12/how-do-neural-networks-really-work/

[6] Stone, M.; Shaw, R. Database marketing for competitive advantage. *Long Range Planning*, volume 20, no. 2, 1987: pp. 12–20, ISSN 00246301, doi:10.1016/0024-6301(87)90002-1, [Online; visited on August 1, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/0024630187900021

[7] Farris, P. W.; Bendle, N.; et al. *Marketing metrics: The definitive guide to measuring marketing performance.* Pearson Education, 2010.

[8] Peter Fader, B. H. What's Wrong With This CLV Formula? 2014, [Online; visited on December 2, 2021]. Available from: https://www.semanticscholar.org/paper/What%E2%80%99s-Wrong-With-This-CLV-Formula-Fader-Hardie/6326f8c18f65a0b544eac51266b123bbc2541c77

[9] Peter Fader, B. H. Customer-Base Valuation in a Contractual Setting: The Perils of Ignoring Heterogeneity. *Marketing Science*, volume 29, no. 1, 2010: pp. 85–93, ISSN 0732-2399, 1526-548X, doi: 10.1287/mksc.1080.0482, [Online; visited on August 3, 2021]. Available from: http://pubsonline.informs.org/doi/10.1287/mksc.1080.0482

[10] Peter Fader, B. H. How to project customer retention. *Journal of Interactive Marketing*, volume 21, no. 1, 2007: pp. 76–90, ISSN 10949968, doi:10.1002/dir.20074, number: 1. Available from: https://linkinghub.elsevier.com/retrieve/pii/S1094996807700233

[11] Lebedeva, E. Understanding the Customer Lifetime Value with Data Science. 2018, [Online; visited on December 12, 2021]. Available from: https://miro.medium.com/max/746/0*TiRrmWTMgQSRxP31

[12] Lebedeva, E. Understanding the Customer Lifetime Value with Data Science. 2018, [Online; visited on December 12, 2021]. Available from: https://miro.medium.com/max/838/1*gMtOpyWQ9L_znDYdaeXsoA.png

[13] Glen, S. Negative Binomial Experiment / Distribution: Definition, Examples. 2015, [Online; visited on December 14, 2021]. Available from: https://www.statisticshowto.com/wp-content/uploads/2015/04/negative-bimonial.png

[14] Jablecka, M. *Modelling CLV in the Insurance Industry Using Deep Learning Methods*. Master's thesis, KTH Royal Institute of Technology, 2020, [Online; visited on August 11, 2021]. Available from: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-273607DiVA

[15] Tsai, C.; Hu, Y.; et al. A comparative study of hybrid machine learning techniques for customer lifetime value prediction. *Kybernetes*, volume 42, no. 3, 2013: pp. 357–370, ISSN 0368-492X, doi: 10.1108/03684921311323626, [Online; visited on August 18, 2021]. Available from: https://www.emerald.com/insight/content/doi/10.1108/03684921311323626/full/html

[16] Prasasti, N.; Okada, M.; et al. Customer Lifetime Value and Defection Possibility Prediction Model Using Machine Learning: An Application to a Cloud-Based Software Company. In *Intelligent Information and Database Systems*, Springer International Publishing, 2014, pp. 62–71,

doi:10.1007/978-3-319-05458-2_7, [Online; visited on September 8, 2021]. Available from: https://doi.org/10.1007/978-3-319-05458-2_7

[17] Sanjeevi, M. Chapter 4: Decision Trees Algorithms. 2018, [Online; visited on August 21, 2021]. Available from: https://medium.com/deep-math-machine-learning-ai/chapter-4-decision-trees-algorithms-b93975f7a1f1

[18] Breiman, L. Random forests. *Machine learning*, volume 45, no. 1, 2001: pp. 5–32, number: 1 Publisher: Springer.

[19] Choudhury, A. Chapter 4: Decision Trees Algorithms. 2020, [Online; visited on April 29, 2022]. Available from: WhatisGradientBoosting?HowisitdifferentfromAdaBoost?

[20] Coussement, K.; Van den Poel, D. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert Systems with Applications*, volume 34, no. 1, 2008: pp. 313–327, ISSN 09574174, doi:10.1016/j.eswa.2006.09.038, [Online; visited on September 8, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0957417406002806

[21] Hung, S.-Y.; Yen, D. C.; et al. Applying data mining to telecom churn management. *Expert Systems with Applications*, volume 31, no. 3, 2006: pp. 515–524, number: 3 Publisher: Elsevier.

[22] Burez, J.; Van den Poel, D. CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services. *Expert Systems with Applications*, volume 32, no. 2, 2007: pp. 277–288, ISSN 09574174, doi:10.1016/j.eswa.2005.11.037, [Online; visited on September 5, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0957417405003374

[23] Saradhi, V. V.; Palshikar, G. K. Employee churn prediction. *Expert Systems with Applications*, volume 38, no. 3, 2011: pp. 1999–2006, ISSN 09574174, doi:10.1016/j.eswa.2010.07.134, [Online; visited on September 5, 2021]. Available from: https://linkinghub.elsevier.com/retrieve/pii/S0957417410007621

[24] Coussement, K.; Poel, D. V. d. Improving customer attrition prediction by integrating emotions from client/company interaction emails and evaluating multiple classifiers. *Expert Systems with Applications*, volume 36, no. 3, 2009: pp. 6127–6134, ISSN 0957-4174, doi:10.1016/j.eswa.2008.07.021, [Online; visited on September 13, 2021]. Available from: https://www.sciencedirect.com/science/article/pii/S095741740800479X

[25] Mozer, M.; Wolniewicz, R.; et al. Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Transactions on Neural Networks*, volume 11, no. 3, 2000: pp. 690–696, ISSN 1941-0093, doi:10.1109/72.846740, number: 3 Conference Name: IEEE Transactions on Neural Networks.

[26] Mena, C. G.; Caigny, A. D.; et al. Churn Prediction with Sequential Data and Deep Neural Networks. A Comparative Analysis. 2019, [Online; visited on September 28, 2021], 1909.11114.

[27] Brandusoiu, I. B.; Toderean, G. A Neural Networks Approach for Churn Prediction Modeling in Mobile Telecommunications Industry. *Annals of The University of Craiova*, volume 11, 2014: pp. 9–16, [Online; visited on August 22, 2021]. Available from: https://www.researchgate.net/publication/333672299_A_Neural_Networks_Approach_for_Churn_Prediction_Modeling_in_Mobile_Telecommunications_Industry

[28] Digital News Innovation Fund (DNI Fund). [Online; visited on February 7, 2022]. Available from: https://newsinitiative.withgoogle.com/dnifund

[29] Fader, P. Peter Fader, Wharton. Aug 2021, [Online; visited on January 20, 2021]. Available from: https://www.youtube.com/watch?v=guj2gVEEx4s

# Appendix A

# Acronyms

**CLV** Customer lifetime value

**CRM** Customer relationship management

**ML** Machine learning

**NN** Neural network

**DT** Decision Tree

**RF** Random Forest

**RFM** Recency, Frequency and Monetary table

Appendix **B**

# Contents of Enclosed CD

```
readme.txt ........................ CD contents description
src
  implementation .................. source code of implementation
    widget ........................ source code of widget extension
    notebooks ..................... jupyter notebooks of experiments
  thesis .......................... source code of thesis in LaTeX format
text .............................. thesis text directory
  thesis.pdf ...................... thesis in PDF
```

# Appendix C

# REMP widget installation

In this chapter the process of installing REMP and widget is described. Steps:

1. Clone the github repository into local machine

   ```
   git clone https://github.com/remp2020/crm-skeleton.git
   ```

2. Change directory to downloaded folder

   ```
   cd crm-skeleton
   ```

3. Prepare environment & configuration files

   a) `cp .env.example .env`

   b) ```
      cp app/config/config.local.example.neon
      ↪ app/config/config.local.neon
      ```

   c) ```
      cp docker-compose.override.example.yml
      ↪ docker-compose.override.yml
      ```

4. `docker-compose up`

5. `docker-compose exec crm /bin/bash`

6. `chmod -R a+rw temp log`

7. `composer install`

8. `exit docker container`

9. Install widget

   a) Copy the widget into the container from host

      ```
      docker cp crm-clv-prediction-module
      ↪ host\_ip:/var/www/html/extensions/
      ```

b) Enter application docker container

```
docker-compose exec crm /bin/bash
```

c) Open composer.json (You can use nano, but it needs to be installed with apt-get install nano)

d) Add "remp/crm-clv-prediction-module": "dev-main" into require list

e) change minimum-stability to dev

f) Add extension in app/config/config.neon

```
extensions:
    - Crm\ClvPredictionModule\DI\ClvPredictionModuleExtension
```

g) compose require remp/crm-clv-prediction-module

10. `composer install`
11. `php bin/command.php phinx:migrate`
12. `php bin/command.php application:generate\_key`
13. `php bin/command.php user:generate\_access`
14. `php bin/command.php api:generate\_access`
15. `php bin/command.php application:seed`
16. `php bin/command.php application:install\_assets`
17. Go to Adminer localhost:6080 and execute script from sql_script.sql.
    - Server: mysql
    - Username: root
    - Password: secret
    - Database: crm
18. Go to localhost and login
    - Username: admin@admin.sk
    - Passowrd: password
19. Go to Admin → Users → Any user
20. The widget will be shown in the right panel side