MASARYKOVA UNIVERZITA

EKONOMICKO-SPRÁVNÍ FAKULTA

Customer lifetime value and churn rate analysis using machine learning

Bakalářská práce

MARTIN MANDZÁK

Vedoucí práce: Ahad Zareravasan

Katedra podnikové ekonomiky a managementu
Program Podniková informatika

Brno 2025

MUNI
ECON

MUNI
ECON

MASARYKOVA UNIVERZITA
EKONOMICKO-SPRÁVNÍ FAKULTA
LIPOVÁ 41A, 602 00 BRNO
IČ: 00216224
DIČ: CZ00216224

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Akademický rok: 2024/2025

Student:	Martin Mandzák
Program:	Podniková informatika
Název práce:	Analýza hodnoty zákazníka a pravděpodobnost jeho odchodu pomocí strojového učení
Název práce anglicky:	Customer Lifetime Value and Churn Rate Analysis Using Machine Learning
Cíl práce, postup a použité metody:	**Cíl práce:** Cílem této diplomové práce je analyzovat pravděpodobnost odchodu zákazníků v reálné společnosti a navrhnout akční plány pro jednotlivé klastry zákazníků. **Postup práce a použité metody:** Práce bude založena na návrhu a implementaci algoritmů strojového učení se zaměřením na neuronové sítě, a to k predikci hodnoty zákazníka a pravděpodobnosti jeho odchodu. Tvorba modelu bude postavena na historických datech z podniku. Získané údaje budou nejprve zpracovány průzkumnou analýzou dat, na jejímž základě autor identifikuje vzorce odchodu zákazníků. Následně bude vytvořen funkční prototyp, který bude schopen generovat základní údaje o zákaznících dle jejich hodnoty (dle customer lifetime value). Posledním krokem bude navržení a validace doporučení (akčních plánů) pro jednotlivé skupiny zákazníků.
Rozsah grafických prací:	Podle pokynů vedoucího práce
Rozsah práce bez příloh:	35 – 45 stran
Literatura:	CHENG, CH a YS CHEN. Classifying the segmentation of customer value via RFM model and RS theory. *Expert Systems with Applications*. OXFORD: Elsevier, 2009, roč. 36, č. 3, s. 4176-4184. ISSN 0957-4174. Dostupné z: https://dx.doi.org/10.1016/j.eswa.2008.04.003. GALE, Bradley T. a Robert Chapman WOOD. *Managing customer value : creating quality and service that customers can see*. New York: Free Press, 1994, xxii, 424. ISBN 0029110459. LASÁK, Pavel a Maria KRÁLOVÁ. Teaching statistical RFM segmentation in Excel. Online. In *INTED2018 Proceedings*. Španělsko: IATED, 2018, s. 8225-8232. ISBN 978-84-697-9480-7. Dostupné z: https://dx.doi.org/10.21125/inted.2018.2000. KVÍČALA, Daniel, Maria KRÁLOVÁ a Petr SUCHÁNEK. The impact of online purchase behaviour on customer lifetime value. *JOURNAL OF MARKETING ANALYTICS*. ENGLAND: PALGRAVE MACMILLAN LTD, 2025, s. 1-18. ISSN 2050-3318. Dostupné z: https://dx.doi.org/10.1057/s41270-024-00328-9. *Value-ology Aligning sales and marketing to shape and deliver profitable customer value propositions*. Edited by Simon Kelly - Paul Johnston - Stacey Danheiser. Cham: Imprint: Palgrave Macmillan, 2017, XX, 199. ISBN 9783319456263.

STRANA 1 Z 2

Vedoucí práce:	doc. Ahad Zareravasan, PhD
Pracoviště vedoucího práce:	Katedra podnikové ekonomiky a managementu

Datum zadání práce: 7. 4. 2024

Termín odevzdání bakalářské práce a vložení do IS je uveden v platném harmonogramu akademického roku.

V Brně dne: 25. 4. 2025

Bibliografický záznam

Autor:	Martin Mandzák Ekonomicko-správní fakulta Masarykova univerzita Katedra podnikové ekonomiky a managementu
Název práce:	Customer lifetime value and churn rate analysis using machine learning
Studijní program:	Podniková informatika
Vedoucí práce:	Ahad Zareravasan
Rok:	2025
Počet stran:	73
Klíčová slova:	umelá inteligencia, clv, neurálna sieť, miera odchodu, data analýza

Bibliographic record

Author: Martin Mandzák
Faculty of Economics and Administration
Masaryk University
Department of Business Management

Title of Thesis: Customer lifetime value and churn rate analysis using machine learning

Degree Programme: Business Informatics

Supervisor: Ahad Zareravasan

Year: 2025

Number of Pages: 73

Keywords: artificial intelligence, clv, neural network, churn rate, data analysis

Anotace

Táto bakalárska práca predstavuje Kai, model hlbokého učenia vytvorený na predikciu hodnoty životného cyklu zákazníka (CLV) a miery odchodu zákazníkov (CCR) na základe transakčných dát od B2B spoločnosti. Model využíva adaptívnu optimalizáciu spolu s cyklickou mierou učenia pre stabilnejší a presnejší tréning. Kai dosiahol R^2 ~0,71 pri predikcii CLV a približne 78 % presnosť pri predikcii odchodu. Napriek obmedzeniam dát ukazuje, že machine learning má praktické využitie v podnikaní.

Abstract

This thesis introduces Kai, a deep learning model built to predict customer lifetime value (CLV) and churn rate (CCR) using transactional data from a B2B company. The model uses both adaptive optimization and a cyclic learning rate to improve training stability and accuracy. Kai reached an R^2 of ~0.71 for CLV and about 78 % accuracy for churn. While there are some limits due to dataset size and outliers, the results show that machine learning can offer real value in business decision-making and customer retention.

Declaration

Prohlašuji, že jsem bakalářskou práci na téma Customer lifetime value and churn rate analysis using machine learning vypracoval samostatně pod vedením Ahada Zareravasana a uvedl v ní všechny použité literární a jiné odborné zdroje v souladu s právními předpisy, vnitřními předpisy Masarykovy univerzity a vnitřními akty řízení Masarykovy univerzity a Ekonomicko-správní fakulty MU.

Brno December 1, 2024

....
Martin Mandzák

Acknowledgements

I'd like to thank my supervisor, Ahad Zareravasan, for his support and feedback throughout this thesis. Thanks also to the Department of Business Management for providing the tools and guidance I needed. Finally, I'm grateful to everyone who supported me along the way—it made a difference.

Table of contents

List of Figures

List of Tables

Glossary

B2B	–	Business to business
B2C	–	Business to customer
CCR	–	Customer Churn Rate
CLV	–	Customer Lifetime Value
ReLU	–	Rectified Linear Unit
RFM	–	Recency Frequency Monetary
KDE	–	Kernel Density Estimation

List of Appendices

Text Appendices

1 Introduction

Metrics such as customer lifetime value (CLV) and customer churn rate (CCR) are important indicators of productivity and profitability. Customer lifetime value (CLV) is defined by Gupta (2006) as "the present value of all future profits obtained from a customer over the life of his relationship with the firm". Customer lifetime value recognises a hierarchy in customers, meaning that some are more important than others. CLV helps with determining which customers are the most important, helping with prioritisation of processes and different tasks to keep these kinds of customers satisfied. CCR on the other hand serves as an indicator to help choose correct precautionary measures to keep it as low as possible. According to Neslin Scott (2006) CCR is managed by predicting which customers are the most likely to churn and finding ways of inducing them to stay. This approach potentially saves money by choosing specific customers to target hence not wasting money on customers would have stayed anyway. Combined, these indicators help businesses pick the most effective marketing strategies, customer support and resource allocation.

The main objective of this thesis is to develop a machine learning model that takes CLV and CCR into account. The model in question will focus on deep learning. According to Chollet (2021), deep learning is a branch of machine learning where models are long chains of geometric transformations. These operations are then grouped together into modules. Goodfellow (2017) adds onto this by specifying these modules into different kinds of layers, namely input, output and hidden layers. Each layer is then taking from the one before it and "feeding" the information to the next one. This process repeats until the final layer, also known as the output layer. For brevity, we have named this learning model Ace.

All the data used is sourced from an anonymous company. Going forward, we shall name this company "Company A" for convenience's sake. Company A specializes in manufacturing synthetic threads which can later be used for car tires, synthetic nets and ropes. This should bring said company real tangible benefit in a form of a report informing them about the nature of their customers.

In conclusion, this thesis will aim to successfully develop and train a machine learning model that will consider factors such as CLV and CCR. This will be achieved by using a deep learning model called "Kai" using data from an anonymous company, which will be referred to as Company A. Lastly, we will use the help of AI tools for purposes of grammar checking and stylistic edits.

1.1 Research statement

This thesis focuses on developing a functional machine learning algorithm that considers CLV and CCR as two of its' primary metrics. CLV which is the current customer value and CCR which is the rate at which customers leave are crucial in company decision-making. By developing Kai, we intend to give Company A valuable information on their customers, as well as demonstration of business informatics combined with economics principles such as marketing and resource allocation in the real world.

This thesis aims to develop and train Kai as a machine learning solution to CLV and CCR problem using data from Company A.

2 Literature review

In this literature review, I would like to go over the sources which will serve as a backbone for my thesis that will be used in different citations, quotes and paraphrases used throughout the thesis. My thesis is to serve the attempt in connecting the customer lifetime value (CLV) prediction and customer churn rate prediction. From a practical standpoint, with help of an anonymous company's data, my thesis should help increase the bottom line, or at the very least provide some insight into profits and customer retention.

Being a student of business informatics, I wanted to combine both the world of economics and the world of IT. From a coding standpoint, I would consider myself an intermediate programmer who could apply for a junior position and get hired. Regarding machine learning, however, I was completely new. This hindrance did not slow me down and I immediately started looking into how neural networks work and soon I started building my own prototypes. In the time of writing this, I have written a functional neural network which is able to recognize hand-written digits from the MNIST dataset. I got infatuated with the world of economics during my final years of secondary grammar school and this infatuation lasted throughout my years at the university. Being primarily at the faculty of economics and administration I have a good number of courses related to the subject under my belt. Still, I would not consider myself an expert. In contrast, I have come a great way ever since I started working on this thesis and I do believe in my ability to provide an objective result that will be of practical use.

Literature will be sorted thematically, that is, split in between IT and economics. I believe this to be the best course of action to avoid potential chaos and confusion. It will also make drawing comparisons easier and more eligible.

2.1 Deep learning

According to Chollet (2021), deep learning is a branch of machine learning where models are long chains of geometric transformations. These operations are then grouped together into modules. Goodfellow (2017) adds onto this by specifying these modules into different kinds of layers, namely input, output and hidden layers. This allows for abstractions which are helpful for modelling these networks.

2.2 Neural networks

Feedforward neural networks, also known as multilayer perceptrons (MLPs) are the most common type of deep learning models. "These models are called feedforward because information flows through the function being evaluated from xx through the intermediate computations used to define f, and finally to the output y. There are no feedback connections in which outputs of the model are fed back into itself." (Goodfellow, et al., 2006)

These networks are split into 3 notable layers, the input layer, the hidden layer, and the output layer. Every layer is comprised of any amount of node, or perceptrons, also called neurons, each with its' own individual bias, connected to every other node on the following layer. Each connection also carries a weight attached to it. We can imagine the weight being an indicator of how strongly the nodes connected by it affect each other. Moving forward, we will calculate the output of each neuron like this:

$$a = \sigma(z)$$
$$z = wx + b$$

Where **a** resembles the activation of the neuron, **σ** is the sigmoid function, **x** is the input, **w** represents the weight of the neuron and **b** represents the bias of the neuron.

The input layer, as the name suggest, takes in data relevant to the model in hand. That means, we have to 'clean' the data before putting it into the deep learning model. Data passed through the input layer gets processed within the hidden layer. The hidden layer does not necessarily have to be only 1 layer, it can be any number of layers that fulfil the same function within abstraction. The final, output, layer is where we get our results from and where we adjust the costs of each individual weight and bias on every level beforehand. In order to get correct results, we use a myriad of tools to optimise the process such as use of gradient descent, a cost function and backpropagation.

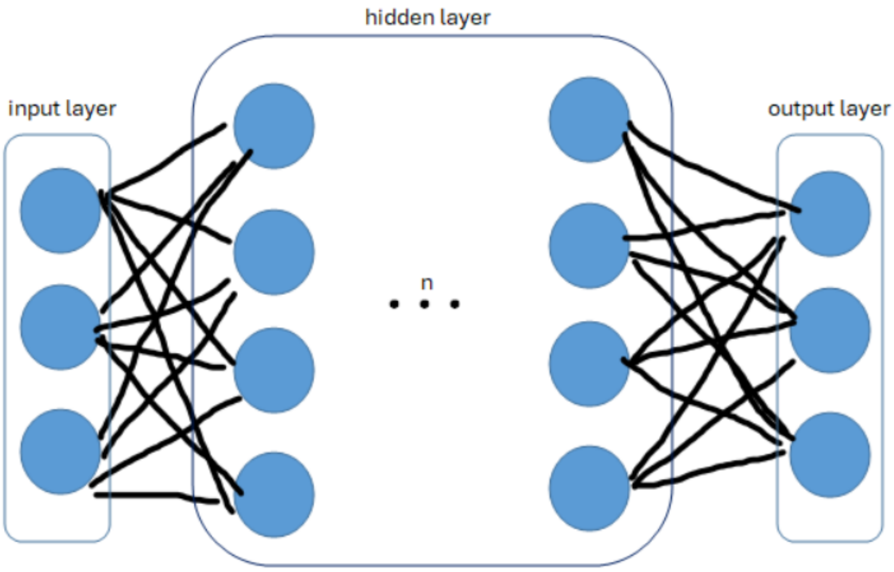


Figure 1: Visualisation of a neural network

2.3 Gradient descent

Due to the large quantity of different variables when working with neural networks, LeCun (1998) suggests the usage of gradient descent. Many publications as well as books compare gradient descent to a ball rolling downhill, eventually finding its' way into the valley. This method, though a correct solution, takes too much time and resources because datasets used for neural networks tend to be large. Goodfellow improves this method by using Stochastic Gradient Descent, which will be referred to as SGD, which shuffles the data and picks random batches for each epoch of the training drastically cutting down on time required for the network to learn. Hornik (1991) states that feedforward networks are approximators and the accuracy of this approximation is measured by the closeness of each function. Goodfellow as well as many others introduce cost functions to achieve this. Some of the most common ones are the sigmoid function or the ReLU function. All cost functions serve the same goal which is to adjust the weights and biases of each neuron in the network on a non-linear basis to accommodate for more than just a simple 1 or 0. According to Chollet, the trick to cost functions is to use a system where we score the neural network based on how far off it guessed from the desired target. Then we adjust the weights and biases for each epoch until we eliminate the error margin as much as possible.

Takamitsu (2015) proposed an adaptive algorithm adapting its' parameters during generative samples to prevent setting errors in their article. The proposed solution allows us to use the Bayesian variable selection faster than other available algorithms. Gradient descent has shortcomings as well. Schmidhuber (2015) identified a problem with gradient descent. Typical neural networks suffer from vanishing and exploding gradients. This means that the gradient either grows rapidly or grows completely out of bounds, known as the long-time lag problem. To counter this issue each recurrent neural network is trained unsupervised to predict its' next input. However, this should not affect this thesis since the dataset is not large enough to allow for this to occur.

2.4 Learning rate

Chollet (2017) stated that: "the purpose of the "weight update" step (represented by the preceding update_weights function) is to move the weights by "a bit" in a direction that will reduce the loss on this batch. The magnitude of the move is determined by the "learning rate," typically a small quantity." Learning rate is typically a smaller quantity because if it was a higher amount, the model could easily 'overshoot' the correct value. We can imagine our model's results jumping over and under the correct value repeatedly until eventually landing within the acceptable range. With a smaller value, these jumps are smaller as well, avoiding the problem of overshooting.

There are 2 types of advanced learning rate techniques. Firstly, there are adaptive learning rates and secondly, there are cyclic learning rates. Adaptive learning rates are more focused on adjusting the learning rate depending on previous performance during training. "The adaptation allows us to derive strong regret guarantees, which for some natural data distributions achieve better performance guarantees than previous algorithms." (Duchi et al., 2011) They later go on to prove that adaptive models are outperforming the non-adaptive ones. According to Smith (2017) their one disadvantage is that they tend to be computationally expensive.

Cyclic learning rates, on the other hand, are learning rates where the value moves periodically in between an upper and lower bound.

According to Li et al. (2020) increasing and decreasing the value of the learning rate is beneficial for the model's performance in the long run because of results shown in other sources such as Smith (2017). Smith (2017) then goes on to argue that cyclic learning rates can be envisioned intuitively by imagining the loss function topology where minimizing loss arises from finding saddle points rather than local minima.

2.5 Cost function

One of the main means of evaluating the effectiveness of our neural network is to use a cost function. The simplest example of a cost function is the quadratic cost function.

$$C = \frac{(y-a)^2}{2},$$

Where **C** is the cost function, **y** is the desired/expected output and **a** is the activation function of the neuron after receiving an input.

This causes the cost value to be higher whenever we are further away from the desired or expected value and since our model 'wants' to have the cost values be as low as possible, it knows to adjust the weights and biases of neurons accordingly via backpropagation. But as Nielsen (2019) correctly points out, "Although this example uses the same learning rate (η=0.15), we can see that learning starts out much more slowly. Indeed, for the first 150 or so learning epochs, the weights and biases don't change much at all. Then the learning kicks in and, much as in our first example, the neuron's output rapidly moves closer to 0.0."
To avoid this slow early learning other cost functions were invented, most notably the cross-entropy function and the ReLU function. The cross-entropy function can be defined as:

$$C = -\frac{1}{n}\sum_{x}^{n}[y\ln a + (1-y)\ln(1-a)]$$

With **n** being the total amount of data records and the sum representing the outputs of all training inputs together.

Nielsen (2019) continues to explain, that this function avoids the slow learning process because the weight learning is controlled by the error margin. To prove this, he elaborates by mathematically proving the formula to be more efficient. To spare us the details, considering they're past the scope of this thesis, we will simply just show the results:

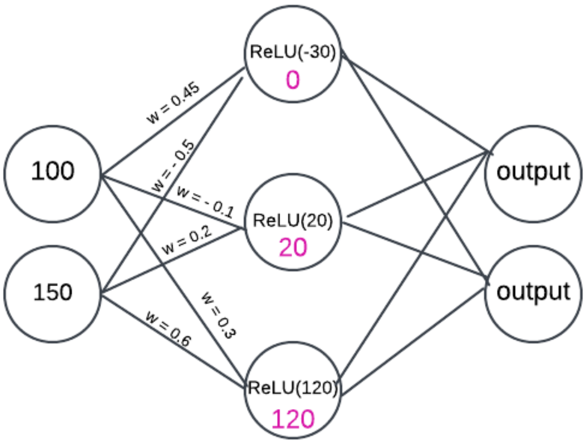
$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_{x}^{n} x_j(\sigma(z) - y)$$

Where **$\sigma(z)-y$** represents the error margin and x represents the input. Because of this, we can say that the ratio between the cost and weight partial derivatives of a specific neuron are equal to the average of the sum of all training inputs put through the neuron. This means that the bigger the error margin, the faster our neural network can learn.

ReLU (Rectified linear unit) function is an activation function that can be summarised as the maximum between an input consisting of the sum of weights of the previous layer and 0. The simplicity of ReLU does not prove its' unreliability. "It is shown that ReLU DNN models with sufficiently many layers (at least two) can reproduce all the linear finite element functions. This in some sense provides some theoretical explanation of the expressive power of deep learning models and the necessity of using deep layers in deep learning applications." (He, et al, 2018)
It can be portrayed by this expression:

$$ReLU = \max(0, x)$$

With x being the input, or in this case, the sum of all weight from the previous layer.



.

Figure 2: Innerworkings of ReLU on a single layer

2.6 Backpropagation

Now, that we have an understanding of a cost function, we can finally understand its' use. Backpropagation is an algorithm due to which our neural network can learn from the errors it has made in previous epochs. According to Nielsen (2019), the backpropagation algorithm stands on 4 equations:

E1:

$$\delta_j^L = \frac{\partial C}{\partial a_j^L} \cdot \sigma'(z_j^L)$$

E2:

$$\delta^L = ((w^{L+1})^T \cdot \delta^{L+1}) \odot \sigma'(z^L)$$

E3:

$$\delta_j^L = \frac{\partial C}{\partial b_j^L}$$

E4:

$$\frac{\partial C}{\partial w_{jk}^L} = a^{L-1} \cdot \delta_j^L$$

E1 calculates the error in the output layer by multiplying the ratio of cost and the output of the neurons with the **σ prime** of the weighed input **z.** The ratio of the partial derivatives serves as the indicator for how easy the neuron is to activate. Meanwhile, the sigmoid prime is the activation function itself.

E2 serves as the continuation of E1. It calculates the error of the following layer, meaning we use E1 at the start and continue with E2 until we reach the input layer.
E3 is the cost over bias of a specific neuron and **E4** is the rate of change of cost with respect to any weight.

2.7 Customer lifetime value

Customer lifetime value (CLV) is defined by Gupta (2006) as "the present value of all future profits obtained from a customer over the life of his relationship with the firm". Customer lifetime value recognises a hierarchy in customers, meaning that some are more important than others. Gupta goes on to explain 2 different approaches taken when estimating customer lifetime values. Firstly, we try to predict each individual customer's expected life based on retention models and calculate their current value. Alternatively, we can include the probability of retention is calculated within the customer lifetime value equation. Gupta argues that it may also be appropriate to consider a constant margin. Customer lifetime value is expanded upon by Borle (2008) by using a hierarchical Bayes approach to evaluate CLV while considering company's spending habits. Their model is that compared to 2 different applications of modelling customer lifetime value. First, it is compared to the NBD-Pareto model which is in its' very nature heuristic.

Wübben (2008) explains the NBD-Pareto model as a construct building upon the assumption that every customer purchase will follow the NBD model in a Pareto 80/20 distribution. This approach is not very precise; however, it stands as a solid approximation of customer lifetime values. Secondly, when it comes to targeting customers Borle's model was compared to the RFM value framework. Zhang (2015) elaborates on RFM by stating that the simplest model segments customer into 3 main groups based on recency, frequency, and monetary values. They suggest accounting for purchase clumpiness (C) as well. The reasoning behind this choice is rooted in the knowledge of purchase clumps being a strong predictor on an individual customer basis. Borle's model faired very well in relation to both approaches and so we will be using it going forward as an inspiration when modelling our own customer lifetime value model.

2.8 Customer churn rate

According to Neslin Scott (2006) customer churn rate (CCR) is managed by predicting which customers are the most likely to churn and finding ways of inducing them to stay. This approach potentially saves money by choosing specific customers to target hence not wasting money on customers would have stayed anyway. Lambrecht (2006) elaborated on this by explaining why people choose to stay with the main themes being convenience, taxi meter effect and overestimation. Lambrecht also shone light on the differences in between a tariff-based service and pay-per-use service. In either case, the more convenient option won which tended to be the tariff option, only exceptions being individuals with an unstable income (e. g. students). Another way of increasing the profits is increasing the number of future sales. Kuksov (2010) explored this idea by having 2 sets of customers, the second set of customers having their expected value and uncertainty affected by the reviews left behind by the first set. The results were heavily affected by current market growth because naturally, if the market is declining, people are less inclined to spend money resulting in the company being induced to better already existing reviews by adding frills. On the other hand, if the market is rapidly growing, cutting down on initial prices may be more beneficial. Moreover, the article considered the idea of introducing negative frills or hidden charges where instead of adding an additional unexpected value for the customer, they will have to pay extra to be able to keep using the product they had bought. Even though this approach is very effective we will restrain from using it because the company I will be working with operates on mostly business-to-business basis. Adding in hidden charges would most likely destroy any goodwill which could hurt the company in the long run.

Despite minor disagreements between authors, we can use the gathered information to model, build and train a sufficient neural network. This network, Ace, will focus on calculating customer lifetime values for each customer, predicting their lifetime length, and deducing the subsequent churn rate. The cost function will be modelled based on the knowledge gained from the literature focused on customer lifetime value, churn rate and retention rate reviewed above. The rest of the neural network will be built based on everything I have gathered in literature regarding deep learning and machine learning. Stochastic gradient descent will be used to speed up the training process. Result of our thesis will be functioning neural network, ready for use in the real world. Apart from this I will conclude an analysis of the data generated by Ace, drawing conclusions and possible recommendations for the company I will work with. To conclude this thesis, we will end on a discussion where the research question will be looked at once again to see whether we have answered them and if so, how they were answered.

2.9 Creating models using CLV and CCR

To be able to successfully build an economic model we need to understand how customer lifetime value is calculated. Reinartz (2003) states that CLV can be calculated as a sum of the ratio between the price and cost difference multiplied by the chance of the subject being a repeat customer and any discount rate at any given time within the chosen time period subtracted by any acquisition cost.

$$\text{CLV} = \sum_{t=0}^{T} \frac{(p_t - c_t) r_t}{(1+i)^t} - AC$$

Gupta (2004) also elaborated by clarifying that if we are only concerning ourselves with an infinite time frame and the rate of p_t and c_t are constant, we can use a simpler formula:

$$CLV = m \cdot \frac{r}{1 + i - r}$$

Where m represents a margin and CLV is calculated as margin multiplied by margin multiple.

This is the underlying principle behind all models explained later in this chapter. Although there are many of them, we have only decided to focus on the most notable and relevant ones.

2.9.1 RFM models

RFM models have been mentioned in literature (Filser, 1991) for over 30 years which makes them some of the oldest models currently still in use. RFM is comprised of 3 main components. Recency is understood as the time from the last purchase to current day. Frequency on the other hand is how close are different purchases in relation to each other. Lastly, monetary is contributed to the value of completed purchases.

“RFM models create "cells" or groups of customers based on three variables-Recency, Frequency, and Monetary value of their prior purchases. (Gupta et al., 2004) They go on to explain that the RFM models can be imagined as a 3-dimensional object. Easiest showcase of this would be a 3-dimensional graph using cartesian coordinates with the x axis being frequency, y axis being recency and z axis being the monetary value.

Recency is shown to be the most impactful metric, followed by frequency. Because of the different impact frequency, recency and value have on the result it is common practice to assign different weights to these metrics, with the most impactful one having the most weight.

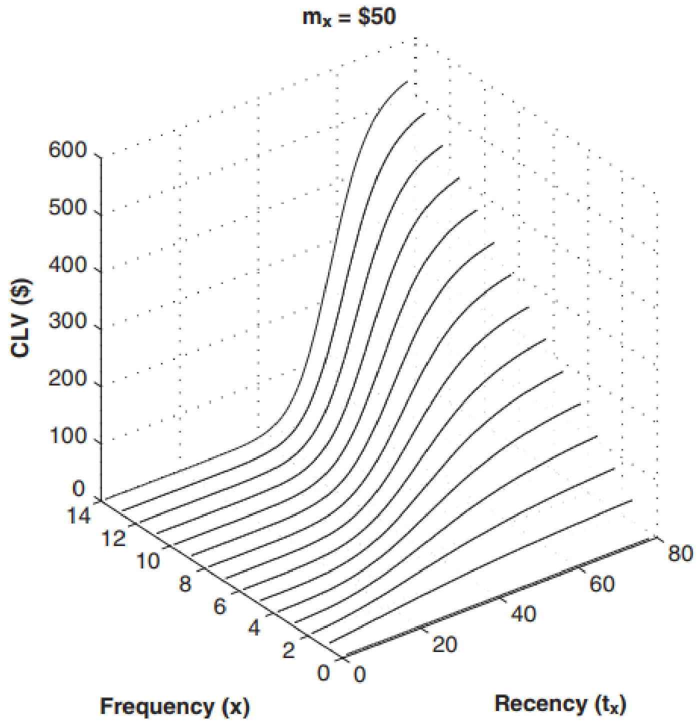


Figure 3: Graph depicting the dependency of frequency, recency and amount, Fader (2005)

This distribution of weights is what ultimately causes the most notable weakness of the RFM model. Due to its' nature as a scoring model, there is not a natural way to add a monetary value to results generated by it.

2.9.2 Probability models

Probability models assume that a customer is going to act in line with some probability distribution. The most common model is the Pareto/NDB model developed by Schmittlein, Morrison and Colombo (1987) which is according to Gupta (2004) best used for customers in a non-contractual setting.

Schmittlein et al. (1987) state that for this model to work, five assumptions are required to be true. Firstly, the customer is either presumed to be "alive" or not. While alive, the customer purchasing behaviour is in correlation with the Poisson process. Secondly, the customer is considered alive within a certain period after their last purchase. Thirdly, customers follow the gamma distribution when purchasing. Penultimately, the death rate of customers also follows a gamma distribution. And lastly, rates for purchasing and customer "death" are independent of each other.

Gamma distribution is defined as:

$$\Gamma(\alpha) = \int_0^\infty t^{\alpha-1}\, dt$$

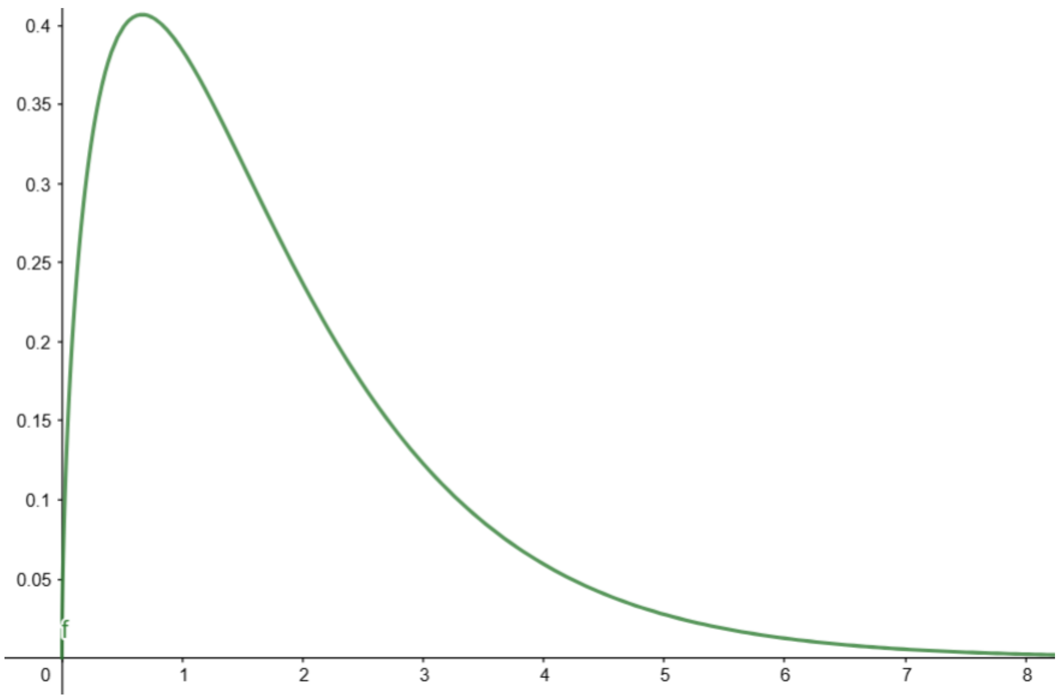


Figure 4: Graph of a Gamma distribution

Gupta et al. (2004) explain that the model only considered recency and frequency in its' final calculation. In practice, this means that the NDB/Pareto model also suffers from the lack of monetary aspect by default.

2.9.3 Econometric models

Econometric models are similar to probability models in spirit. These models also follow the NDB/Pareto formula but account for hazard within their calculations. CLV is achieved by combining customer acquisition, retention and expansion.

2.9.3.1 Customer acquisition

Customer acquisition is the process of convincing potential or "dead" customers to make their first purchase. Customer acquisition differs in between B2C and B2B businesses. The main difference is as follows:

"In a business-to-consumer context, customers may shift out of a targeted demographic, or their personal circumstances may change such that they no longer find value in an offering. In a business-to-business context, corporate customers may be lost due to acquisition by another organisation with firmly established supplier preferences, or if ceasing production of the goods and services for which the input was needed or ceasing trade." (Ang, 2006)

There can be drawn many parallels between B2B and B2C relationships in regard to customer acquisition. The main difference maker is the fact, that B2B is strictly contractually driven while B2C is not necessarily so. This means that B2B relationships are easier to track the history of, and as a result, the forecasting of future purchases is generally easier and more reliable than in a B2C relationship.

2.9.3.2 Customer retention

According to Gupta (2004) customer retention is the probability of a customer being a repeat buyer or in short, being "alive". When under a contract, the customer lets the company know whenever they wish to terminate the relationship and as a result becoming a "dead" customer. In a non-contractual relationship, we use different methods to predict whether the customer is still considered "alive" or not.

Customer retention is predicted by either hazard models or Markov models. Two main examples of hazard models are the accelerated failure time (AFT) and proportional hazard (PH) models.

2.9.3.3 Customer expansion

Customer expansion can come in many different forms. Conventionally, we speak of companies expanding to different markets, for example, an American company breaking into the European market. "The companies who have been able to provide these pan-European "one stop shopping" for supply chain services have been mainly American logistics providers who have expanded into the European market" (Jaldin, 2002)

Another way to expand would be to start producing a nuanced product that does not have a substitute within the company's pre-existing portfolio. Best examples of this approach are companies like Amazon, which have completely evaded the potential audience capture. Because customer expansion is not the focus of this thesis we won't give it any more attention.

2.9.4 Computer Science models

Computer science models are split into 2 main categories. Firstly, there are traditional parametric models relying on logit and hazard which are according to Friedman (1991) limited by their reliance on structured frameworks. Secondly, there are non-parametric machine learning models, most notably the support vector machines (SVM), decision trees and multivariate adaptive regression splines (MARS) which have shown higher performance when it comes to prediction and forecasting.

2.10 Customer purchase behaviour and game theory

Lemon et al. (2016) state that customer purchase is comprised of multiple phases: the prepurchase, purchase, and postpurchase. Prepurchase consists of factors affecting the potential customer before buying the product. These can involve things such as brand awareness but even something as simple as the want to purchase a product for a specific purpose. This is further developed by Shim et al. (2001) which proved positive results when customers were presented with buying the same product over the Internet. We can safely deduce from these sources that prepurchase also includes being the path of least resistance as people who were presented with the same product through the Internet were more likely to buy.

Postpurchase on the other hand, is the way the customer feels after the purchase has been completed. "This stage includes behaviours such as usage and consumption, postpurchase engagement, and service requests. Similar to the prepurchase stage, theoretically, this stage could extend temporally from the purchase to the end of the customer's life." (Lemon et al., 2016) This can include things such as the quality of the product as well as a generous return policy or help with maintenance via customer support.

"In contrast, respondents' recalled postpurchase thoughts differed in that (1) product information was recalled at a more aggregate level, that is, more product overall referents, (2) very few of the criteria that were recalled prior to purchase were mentioned, (3) a higher degree of both evaluation outcomes and emotion responses were recalled for postpurchase, and (4) some differences in types of recalled comparison standards were noted relative to prepurchase." (Gardial et al., 1994)

These inherent characteristics of customer purchasing behaviour is what allows for game theory to be utilised. Specifically, for prepurchasing we could speak of signalling games. Gintis (2009) defines a signal as: "A signal is a special sort of physical interaction between two agents. Like other physical interactions, a signal changes the physical constitution of the agents involved. But unlike interactions among non-living objects, or between a non-living object and a living agent, a signal is the product of a strategic dynamic between sender and receiver, each of whom is pursuing distinct but interrelated objectives." Signals being inherently strategic is supported by Leider (2019) wherein the subjects of the study were a part of an auction, signalling to each other about their respective bids. In terms of company strategies, signalling could be interpreted as getting a more premium packaging, pouring resources into advertising or even rebranding altogether. Instinctively, the customer might seek out a more premium looking product as a form of screening. Furthermore, Gintis (2009) goes on to state that in the problem of producers and projects if the producers are sufficiently wealthy, they achieve a truthful signalling equilibrium. This means, that the screening process a customer goes through is not inherently incorrect, though it is rare that all companies are sufficiently wealthy.

Fulfilling the customer postpurchase wants and needs could be done by learning from game theory's repeated games. Namely, an important aspect of postpurchase behaviour could be explained by the Folk theorem. Gintis (2009) states that if the signals of defection are of sufficiently reliable, and if players have sufficiently long-time horizons, the repeated game can attain Pareto-efficiency, or at least approximate Pareto-efficiency as closely as desired. In a B2B dynamic, every purchase being closed with a contract, the signals of defection are extremely clear and reliable. Customer postpurchase strategies can also be seen in the reputational equilibrium. “Consumers follow a trigger strategy, in which they buy the good in each period in which $q \geq q_a$, but if q< q_a in some period, they never buy from the firm again.” (Gintis, 2009) From a B2B perspective, a company could upkeep its’ reputation by holding its’ products to a certain standard, being easy to work with as both a supplier and a buyer, and naturally, safeguarding any trust accumulated throughout the business relationship.

3 Methodology

In this chapter we would like to focus on how we should go about answering our research questions. The focus of the following text will be understanding how we have built Kai as well as how we came to our final cost function. This will be interjected with snippets of code relevant to the topic at hand. The methodology will be comprised of three key stages: Data preparation, building Kai and evaluation. For a more concrete look at Kai, see appendix with the source code.

3.1 Data preparation

The dataset used for this thesis will be anonymised customer data from Company A. The dataset consists of over 2000 confirmed sales logs containing features such as the individual transaction amounts, dates of transactions and other. We took this data and calculated buying frequency as well as customer lifetime value (CLV).

First, we needed to clean the data. We decided to follow the principles stated within BCNF (Codd, 1987). According to the first principle, we have gotten rid of all null values and any other potentially corrupted data. Then we made sure that no data is transitive. This issue occurred after we deleted all null values but values such as re-warehousing persevered. Null values were more prevalent the older the data was. To account for outliers, we have decided to not include them in the learning dataset. We believe that this decision has helped us avoid needlessly large error margins in judgement within Kai.

Afterwards we have decided to cut off the year 2025 from testing considering the lack of data. Due to the nature of the company, we are working with, the number of transactions is not as large as a B2C company. Trying to predict the year 2025 would take away from the reliability of our results.

After we were done cleaning the data, it took this format:

Table 1: Example data used in training

Customer ID	Country	InvoiceNo	Extended Day	Year	Quantity	Revenue
0001	MG Madagaskar	12345	Feb, 12	2020	1670	7690
31251	DE Nemecko	251231	Jan, 26	2021	256	1567
92141	SK Slovensko	912315	Jun, 7	2022	4670	13680

Created by: Author

Every CustomerID has been ran through a hashing function, which will remain private, to guarantee anonymity.

Lastly, we needed to split our dataset into suitable parts for deep learning. According to Nguyen et al. (2021): "Overall, the results showed that the ML models' performance was significantly changed under different training/testing ratios. e results showed that the training/testing ratio of 70/30 was the most suitable one for training and validating the ML models." We decided to go split the data into 3 main parts. The largest chunk accounted for 50 % of the whole dataset and was used for Kai to learn on. We chose the initial 50 % because it accounted for exactly 1 year and we felt it to be the most representative when predicting the next year in later stages. The second part was validation which account for about 20 % of the dataset and was just under half of a year. It served to validate Kai's performance based on its' error margin. This 70 % of our dataset aligns with Nguyen's findings. The final 30 % of the data was used for prediction and final evaluation of Kai's accuracy. More can be found in the Appendix B.

```
X_train, y_train = get_features(data,
                                '2023-01-01',
                                '2023-06-01',
                                '2023-06-02',
                                '2023-12-31')
X_test, y_test = get_features(data,
                                '2024-01-01',
                                '2024-05-26',#40% of
                           year
                                '2024-05-27',#=20% of
                           2yrs
```

```
'2024-12-31')
```

Training data was split even further within the training process into randomly picked mini batches and ran over multiple epochs to speed up the learning process. As Fleuret (2023) states: “However, under reasonable assumptions of exchangeability, for instance, if the samples have been properly shuffled, any partial sum of Equation 3.2 is an unbiased estimator of the full sum, albeit noisy.” This means the result is not as precise as if Kai was running across all data, however, it is computationally more efficient. Due to redundancy in data caused by running hundreds of epochs this problem of noisiness has been nullified to a point of irrelevancy.

3.2 Building Kai

To build Kai we have chosen the programming language Python, for its’ relative ease of use and the utility of libraries such as NumPy, TensorFlow and Keras (Appendix B).

To successfully build Kai, we have been following the book Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2016) as well as Michael Nielsen’s e-book Neural networks and deep learning (2019) which has also taken its’ fundamentals from aforementioned Deep Learning.

We have chosen a model which takes in data from calculated customer lifetime value, the amount spent, date of last purchase as well as frequency of purchasing. Each node has its’ own bias and each edge connecting 2 nodes has a weight attached to it as well. For the purposes of viewing ease, we have decided to only connect each node of a layer to every node of the following layer only in between the input and the hidden layer. Reader should keep in mind that the connections in between layers are the same as in between the input and the hidden layer.

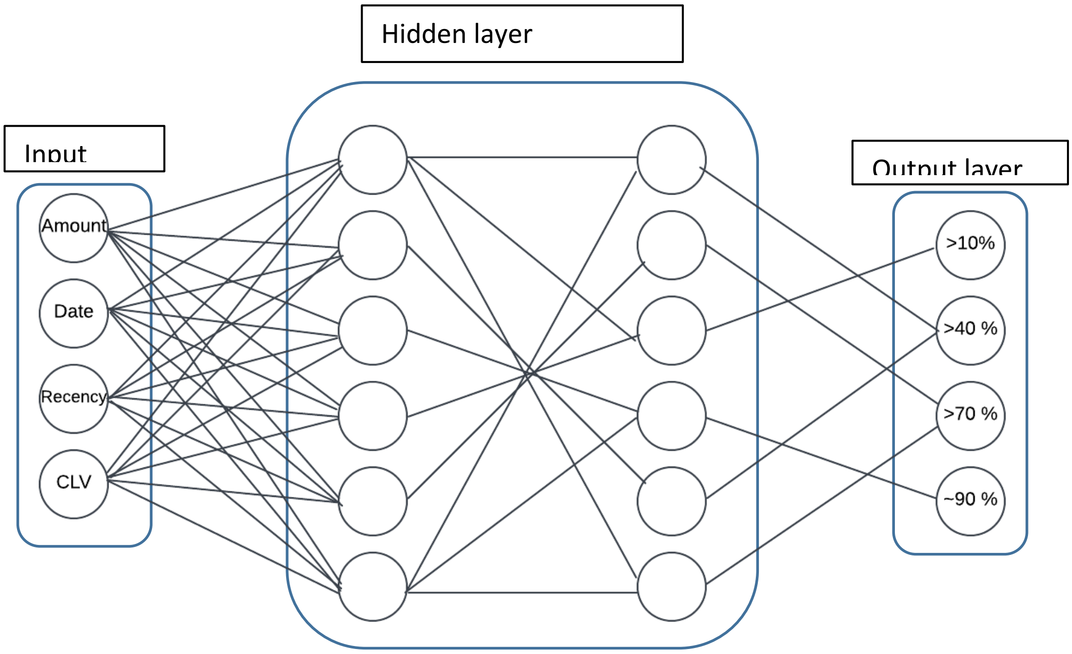


Figure 5: Abstract visualisation of Kai's inputs and outputs

We have taken our testing dataset and trained Kai by comparing Kai's results with the expected outcomes, then adjusting weights and biases of each node and edge accordingly.

For layers themselves, we have chosen Kai to use dense layers as mentioned by Huang et al. (2017). As mentioned in the academic paper by Huang et al., the dense networks are more parameter efficient since features are reused as opposed to relearnt. Apart from this benefit, dense layers also guarantee the connection between a node and every other node on previous and following layer which we deemed paramount to Kai's success.

For these adjustments we have decided to go with ReLU (Rectified Linear Unit) which is an activation function with outputs non-linear results, helping Kai to learn more complex patterns. "It is shown that ReLU DNN models with sufficiently many layers (at least two) can reproduce all the linear finite element functions. This in some sense provides some theoretical explanation of the expressive power of deep learning models and also the necessity of using deep layers in deep learning applications." (Juncai He, Lin Li, Jinchao Xu, Chunyue Zheng, 2018). The SoftMax ReLU activation function we have decided to go with looks like this:

$$f(x) = \ln(1 + e^x)$$

Regarding our learning rate, we have decided to use the adaptive optimizer Adam from the Keras library alongside our implementation of the cyclic learning rate to accommodate for Adam's downfalls. This combination works because Adam works to fine-tune the individual parameter updates while the cyclic learning rate periodically adjusts the global learning rate which helps improve Kai's ability to explain the variance in data. The cyclic learning rate can be found in the Appendix B.

Our output layer is comprised of the confidence Kai has in the likelihood of the customer to churn. We have only accepted customers as likely to churn if Kai believed them to have ~90 % churn probability. To accomplish these churn rate guesses we have employed the sigmoid activation function which returns values on an interval from 0 to 1.

3.3 Evaluation

In this chapter we will assess Kai's performance based on its' accuracy, reliability and overall effectiveness when predicting customer retention. Our analysis will use a myriad of performance metrics as aforementioned accuracy or even precision or sensitivity.

Firstly, we should talk about the evaluation of customer lifetime value accuracy. To best grade our model we have chosen the R^2 score. "The R^2 statistic has an interpretational advantage over the RSE, since unlike the RSE, it always lies between 0 and 1." (James, 2021) A model with a perfect prediction and explanation of variance would meet the R^2 score value of 1, 0 would be equal to guessing based on the mean and negative values would underperform even guessing. R^2 score grades the ability of Kai to explain variance in data. "If R^2 should tell something about the virtues of a model for some given population, the variation among the values of the independent variables should be representative of that population. The simplest way (and in most cases, the only way) to achieve this is to also sample the independent variables randomly from the population." (Helland, 1987) This synergises with our stochastic gradient descent method we chose for training Kai.

Another metric we have chosen to use was mean absolute error (mae). As the name suggest, this shows the average absolute difference from Kai's guess to the actual value. It essentially illustrates the error margin within our model. The lower the MAE, the better. We have considered calculating auxiliary metrics such as mean squared error (mse) and root mean squared error (RMSE). They function similarly to mean absolute error but due to their nature of being squared, they are more sensitive to outliers. Santhusitha and Karunasingha (2021) state: "Based on the (relative) smaller standard error of the estimators, RMSE is preferred for errors having platykurtic distributions, and MAE is preferred for leptokurtic distributions, and both are almost equally acceptable for normal or mesokurtic distributions." Considering our data follows the gamma distribution, which is a leptokurtic distribution, we have chosen mean absolute error over mean squared error or root mean squared error.

Churn rate accuracy will be measured as a percentage of correct guesses based on the training data. This will help us get the general idea of how reliable our system is.

Customer lifetime value will be graphically shown in two ways. Firstly, we'll look at a histogram comparing the predicted and actual customer lifetime values. Secondly, we will have a look at a KDE (Kernel Density Estimation) graph. Kernel density estimation will serve a similar purpose as the histogram but smoothed out to better represent the distribution of customers based on their overall value.

The potential limitations Kai may include bias in data considering this is a small sample for a machine learning project. On top of that a highly likely concern is the optimisation of Kai, meaning that it would not be able to efficiently perform on large-scale datasets. Although this is a shortcoming, we believe that Kai serves more of a proof of concept rather than a fully built corporate-ready solution, we believe that it is reasonable to leave Kai as it is.

4 Results

The penultimate chapter of this thesis will consist of the results we achieved through Kai. Moreover, the data used will be from a company A, which will remain anonymized as per their wishes. For best performance, we decided to run Kai for 100, 200 and 300 epochs firstly with hardcoded learning rate of 0.000069, secondly with a dynamically changing learning rate.

First training at 100 epochs was shallow and extremely inconsistent. We have observed churn rates anywhere from 60 % to 78 % and R^2 scores from -0.00325 to 0.55427. While the churn model accuracy was less affected by the lack of training, this inconsistency was shown on the graphs especially with KDE and the bar chart. The individual mean absolute error was in a range of ~67000 to ~81000.

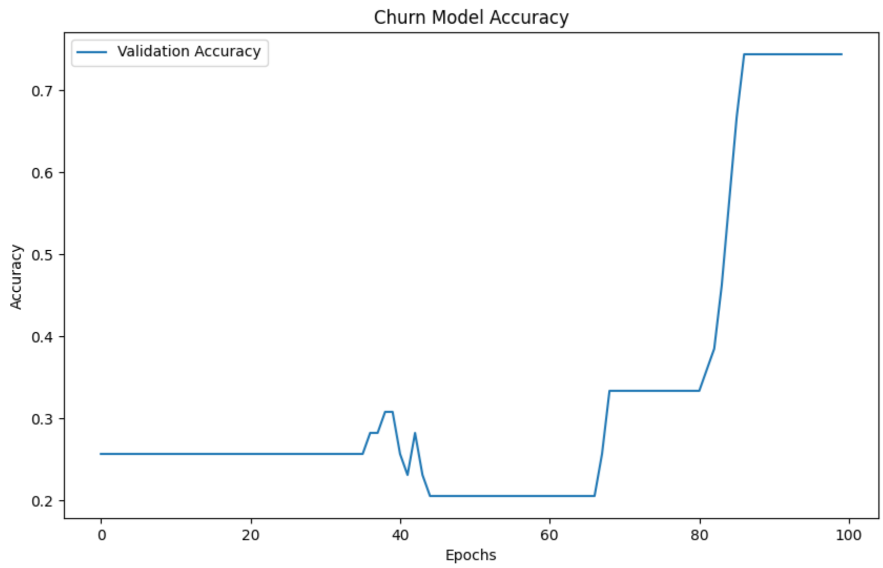


Figure 6: Kai churn rate training iteration: 100 epochs, static learning rate

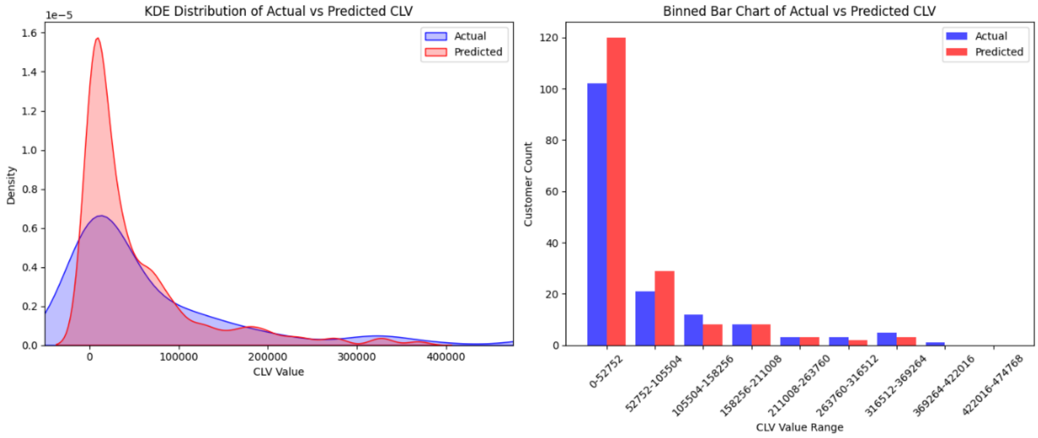


Figure 7: Kai customer lifetime value training iteration: 100 epochs, static learning rate

We can see the density reach 1.6 which means a high probability of any of the customers within the dataset being in CLV range <0, ~80000>. And while that is correct, it is not exactly precise.

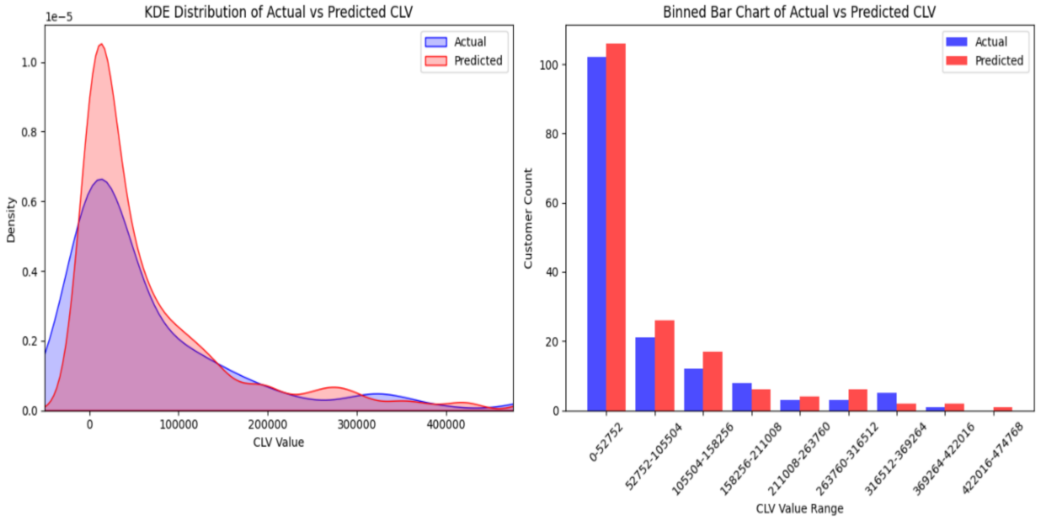


Figure 8: Kai customer lifetime value training iteration: 100 epochs, static learning rate, showing inconsistency

A notable aspect of this graph is the seemingly large negative CLV on the KDE visualisation. This can be interpreted as simply the function of KDE smoothing out the data. Explanation behind this can be found within the bar chart to its' right. With the amount of data in the first interval, it is logical that the smoothening process used for visualisation would have this undesirable effect. Something to note, however, is that this distribution perfectly follows the gamma distribution mentioned earlier within the thesis enforcing Kai's ability to predict customer lifetime value.

After running Kai for 200 epochs, we have achieved the churn rate accuracy from 0.78103 to 0.78296 or 78.1 % to 78.3 %. This percentage proves our model to be fairly accurate and reliable. By looking at the graph we can safely deduce that the learning curve stagnated for the first 30 epochs and immediately shot up afterwards, staying relatively stable until the end of the learning process. Another notable aspect is the dip observed from 100 epochs to 160 epochs.

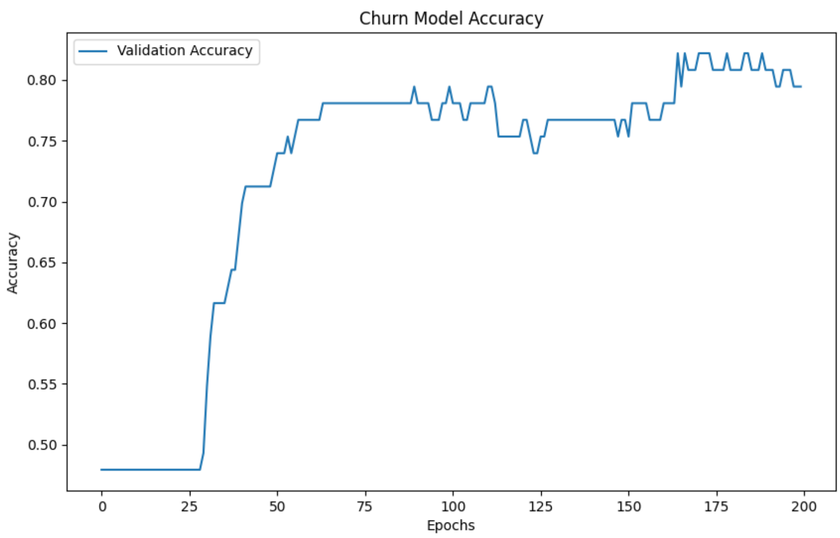


Figure 9: Kai churn rate training iteration: 200 epochs, static learning rate

For the actual customer lifetime value itself, Kai reached the R^2 score from 0.4459 to 0.5868 or 44.59 % to 58.68 %. This makes Kai adept at interpreting the variance in data. The Individual Mean Absolute Error was in range from ~63000 to ~74000.

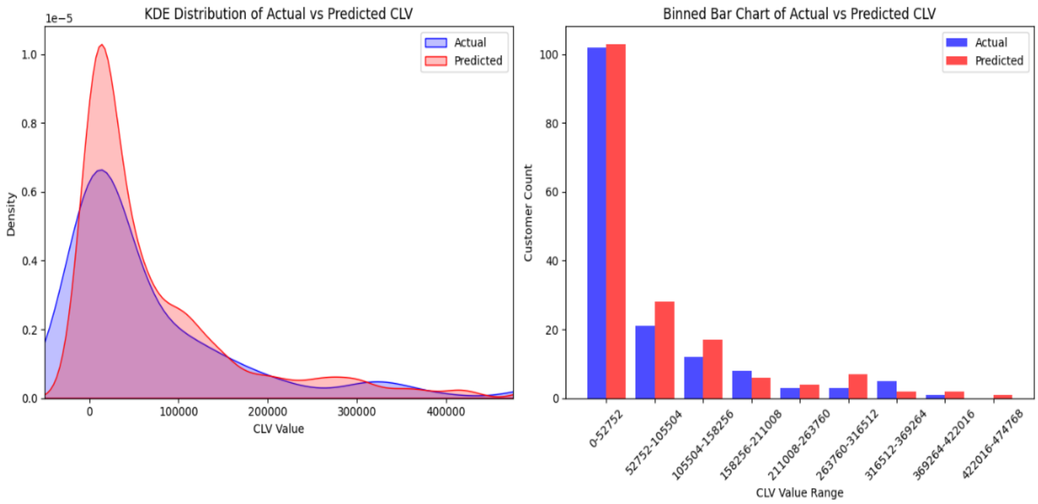


Figure 10: Kai customer lifetime value training iteration: 200 epochs, static learning rate

Ultimately, after 300 epochs, Kai's finished with an R^2 score within ranges of 0.54 to 0.58. The individual mean absolute error was within ~66000 to ~68000 making it a lot more reliable. With a static learning rate, we managed to get Kai within the reliable range.

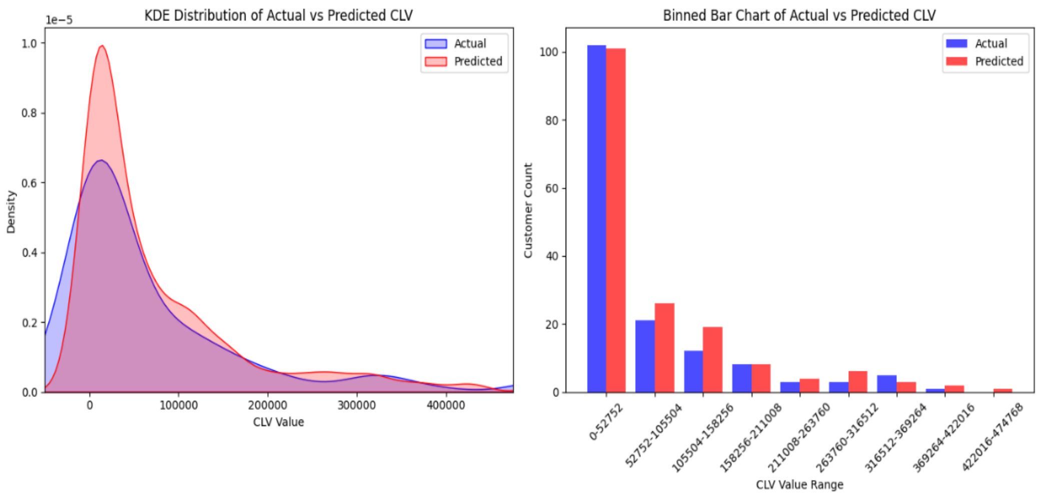


Figure 11: Kai customer lifetime value training iteration: 300 epochs, static learning rate

Afterwards, we have adjusted Kai's learning rate to be dynamic and change over time to maximise the effectiveness and accuracy of forecasting. We have found that after 100 epochs Kai performed with R^2 scores ranging from ~0.6 to ~0.67, varying by a large margin each time it was ran.

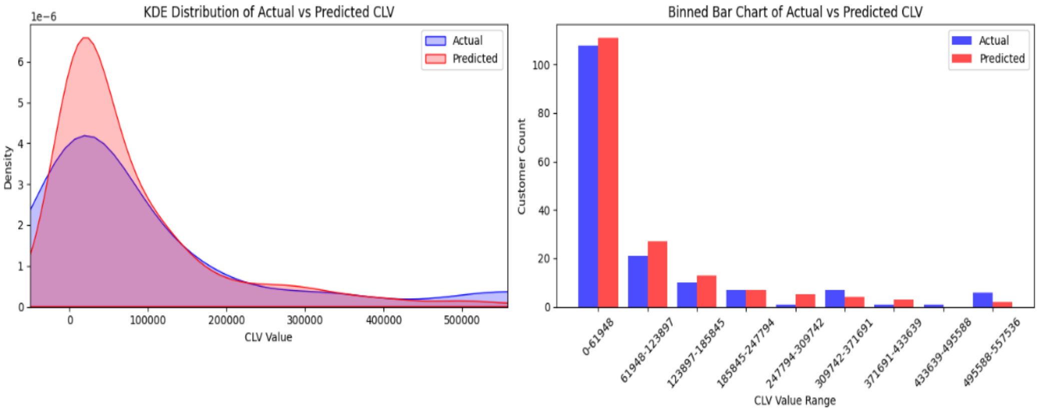


Figure 12: Kai customer lifetime value training iteration: 100 epochs, dynamic learning rate

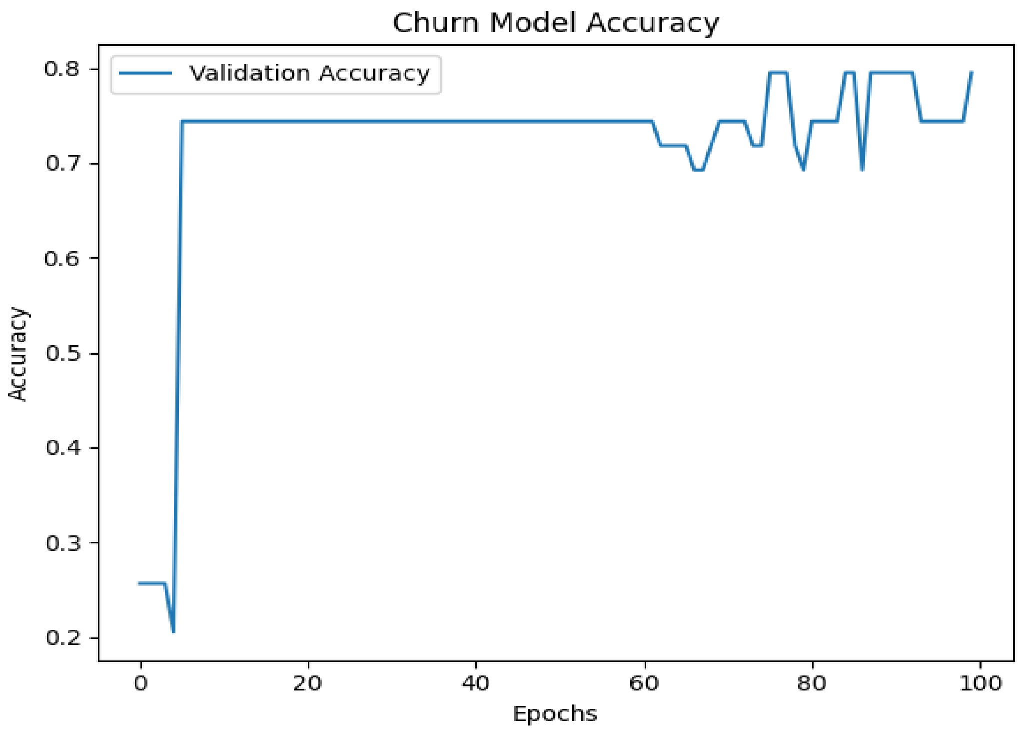


Figure 13: Kai churn rate training iteration: 100 epochs, dynamic learning rate

Churn rate of Kai after 100 epochs with this new approach to the learning rate made it a lot more stable, almost constant throughout the training process. It should be noted that even though Kai was a lot more unstable, its' worst performance was still achieving better results than its' static counterpart at 3 times the epochs.

Another notable aspect of the dynamic learning rate is the noticeable jump in the individual mean absolute error. In every training run the individual mean error jumped from ~60000 to ~80000. This is not necessarily an error within Kai since we could reason that the higher error margin is simply the result of predicting more variance within the data.

After 200 epochs Kai performed more consistently than before, which was to be expected. R^2 within this iteration of Kai were hitting values within the interval of 0.65733 to 0.71031 which could be considered moderately reliable. Individual absolute mean error was ranging from ~84000 to ~89000. This error is still considerably higher than on its' statically declared counterpart, however, the spread of the error is only ~5000 which further supports our idea of the error being so big simply due to the variance in data.

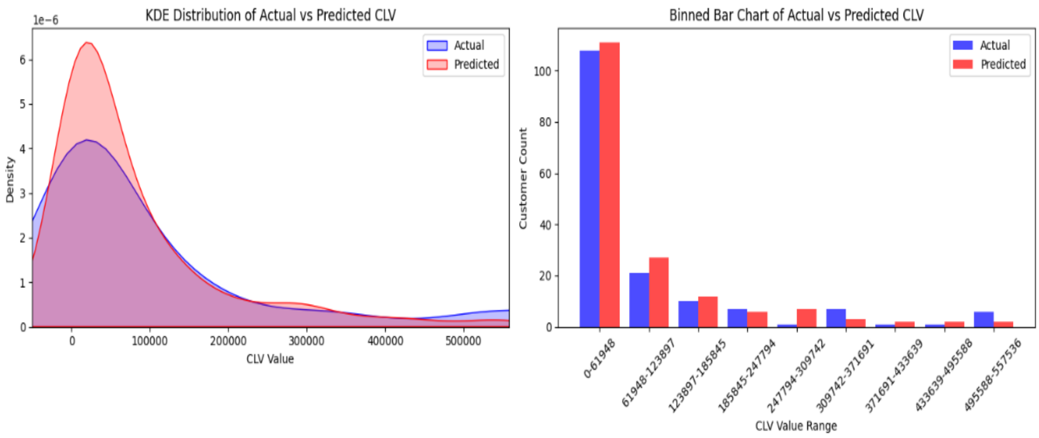


Figure 14: Kai customer lifetime value training iteration: 200 epochs, dynamic learning rate

Regarding the churn rate, this model managed to achieve a consistent accuracy of 0.7703 to 0.7722 or ~77 %. Although, it should be noted that we have observed some overfitting of data, meaning, that going forward we expect churn rate to get less reliable and accurate.

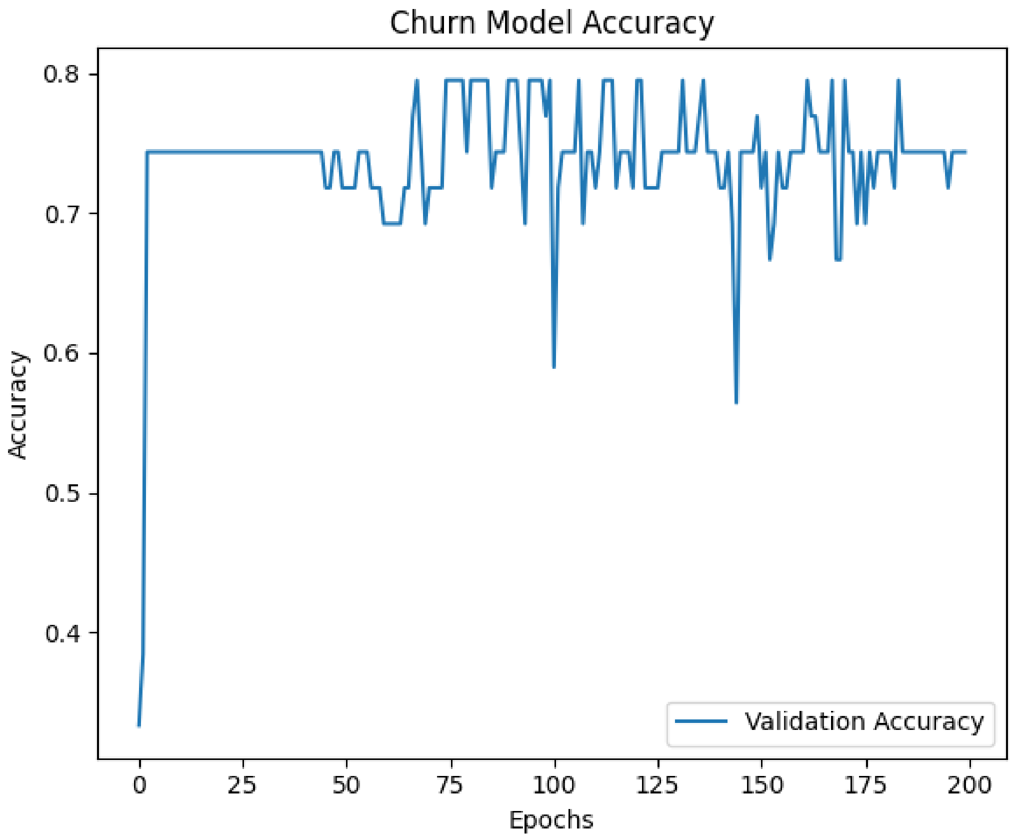


Figure 15: Kai churn rate training iteration: 200 epochs, dynamic learning rate

Aforementioned overfitting can be seen on this graph. The seemingly random plummets by upwards of 15 % accuracy can be explained by data that has already been validated being trained on again, generating a different result on the same variables.

The results after 300 epochs were reaching values in ranges of 0.6553 to 0.6904 which was a clear sign for us that adding more epochs would simply lower Kai's accuracy and we have reached the point of diminishing returns. Narayanan, et al. (2019) go on to try and fix this issue of diminishing returns using the PipeDream which is their algorithmic solution using bi-directional training. However, this solution was not suitable for Kai because of the dataset being comparably small in comparison to what a traditional deep neural network is trained on.

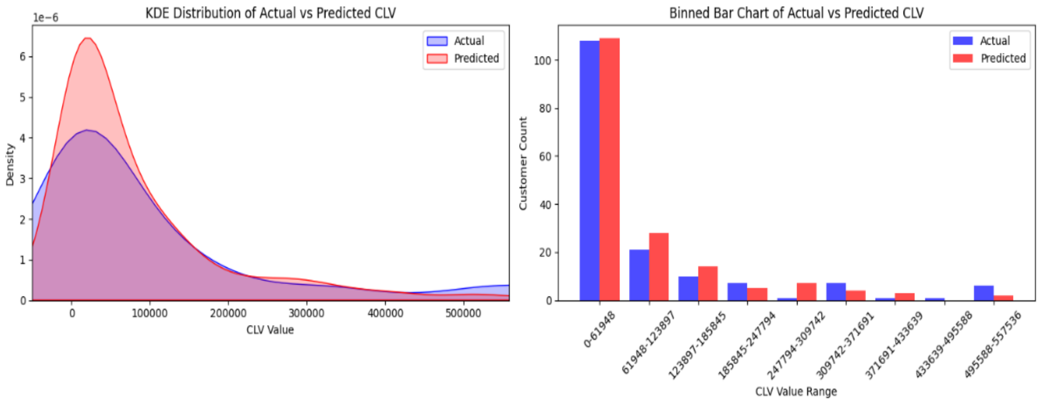


Figure 16: Kai customer lifetime value training iteration: 300 epochs, dynamic learning rate

Despite the performance dip regarding customer lifetime value forecasting, Kai performed better on 300 epochs in terms of the individual mean absolute error which has ranged from ~87000 to ~89000 considerably narrowing the margin of error. This may also be explained by Kai simply failing to explain the variance in data thus making less average error. Overfitting was prevalent during the churn rate analysis during this testing period to a point of being unusable. Though an average of accuracy was not out of line reaching around 70 % correct guesses, judging by the visual representation we can safely deduce that this number is nearly irrelevant showing little consistency.

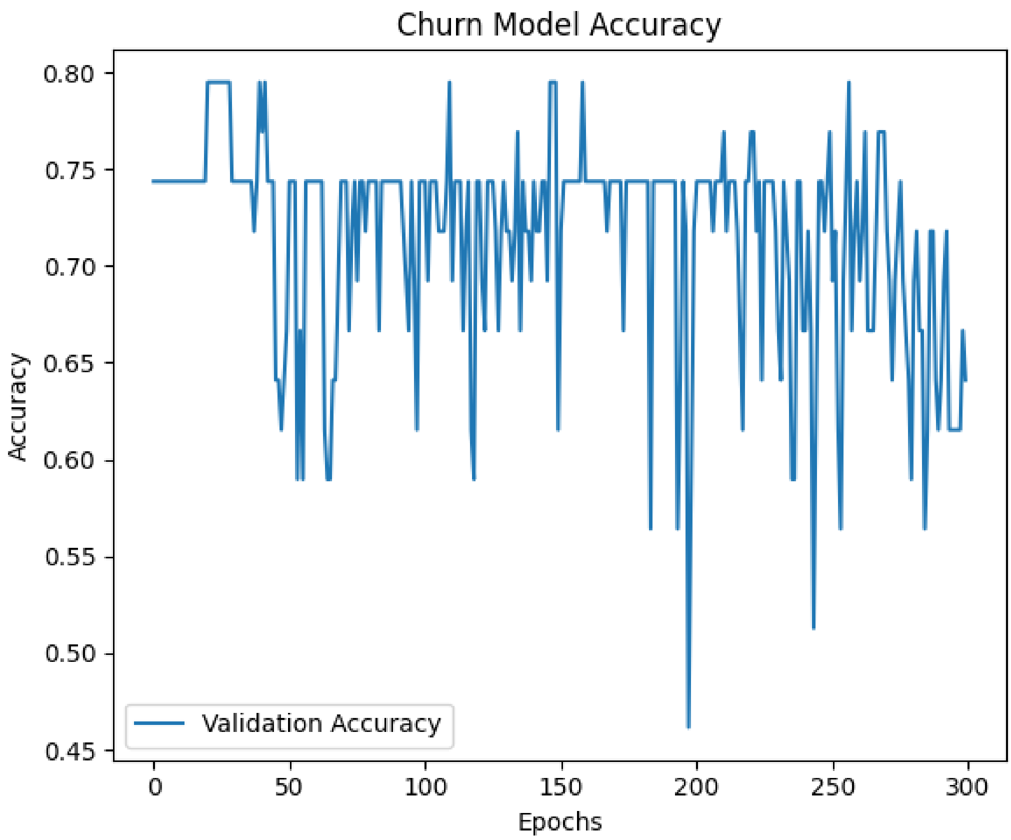


Figure 17: Kai churn rate training iteration: 300 epochs, dynamic learning rate

As shown on the graph, the churn rate prediction accuracy was jumping within a 30 % range, which is unacceptable and cannot be considered reliable. Because of this we have implemented an early_stop() function within the callback function inside Kai's code. This function essentially guaranteed that if Kai was confident on a certain value for *X* number of epochs, the training would stop early to avoid overfitting and consequent degradation of data. After thorough testing we have deduced the best ratio was 250 epochs with 50 epoch patience. Patience is the *X* number of epochs after which the training will stop if Kai feels strongly about a certain value being correct. For churn rate the number of epochs was the same as for customer lifetime value but the patience was slightly lower at 40 epochs until stopping the training. We have equated this discrepancy because of a slightly different activation function on one of the layers of the churn rate prediction model. This model uses the sigmoid function to better accommodate for the desired percentage results as the sigmoid function returns values in the interval from 0 to 1 (inclusive).

Kai has shown an R^2 score of ~0.71 on average with an average individual absolute mean error of ~85000. Churn rate was within the range of ~0.77 to ~0.81. This is further corroborated by the graph shown below.

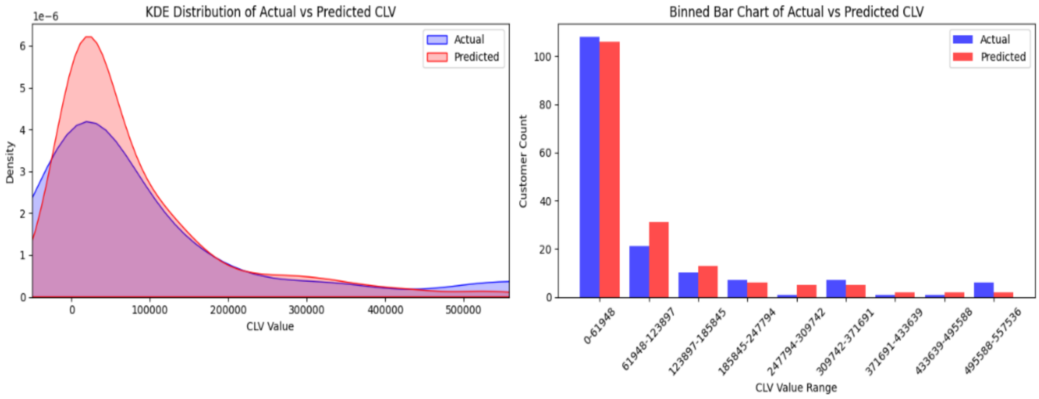


Figure 18: Kai customer lifetime value training iteration: 250 epochs, dynamic learning rate

Considering these are the best results Kai was able to predict, we have taken the liberty to create lists of top 10 customers based on different criteria. Firstly, we had Kai print out top customers based on their customer lifetime value.

Table 2: Top 10 customers by customer lifetime value

CustomerID	Predicted_clv	Actual_clv	Churn_rate_%
159258	1,137,090.50€	1,091,620.00€	0
92370	985,291.12€	1,468,450.00€	0
127550	956,131.44€	1,953,800.00€	0
198505	945,620.19€	967,650.00€	0
172027	940,428.19€	2,128,720.00€	0
70995	822,617.56€	1,915,780.00€	0
172216	691,931.31€	1,029,320.00€	0
71751	559,901.38€	600,380.00€	0
205827	500,368.47€	539,220.00€	0

Created by: Author

Kai undershot a lot of the top spenders by quite the considerable amount. This is to be expected considering the inherent nature of deep learning and neural networks. After getting rid of the upper 7.5 % of customers, Kai's accuracy noticeably improved. While some of the customers stayed in the top 10 with a slightly worse prediction, the ones that followed generally adhered to a trend of better forecasting.

Table 3: Top 10 customers by customer lifetime value, modified

CustomerID	Predicted_clv	Actual_clv	Churn_rate_%
172216	644,491.38€	1,029,320.00€	0
71751	605,028.38€	600,380.00€	0
205827	489,770.59€	539,220.00€	0
90409	467,795.94€	677,250.00€	0
174266	417,231.28€	628,570.00€	0
197506	393,705.44€	575,600.00€	0
50037	364,265.12€	311,360.00€	0
62189	361,369.44€	836,640.00€	0
50628	354,538.75€	555,530.00€	0

Created by: Author

As expected, Kai's forecasting of the customer lifetime value was considerably more accurate after moving closer to the median value. There are still some inaccuracies, namely the customers 62189 and 50628 but in general, we have observed a positive trend in accuracy.

Table 4: Top 10 customers by churn rate

CustomerID	Predicted_clv	Actual_clv	Churn_rate_%
105677	1,858.59€	0.00€	99.9964
93389	157.76€	1,550.00€	84.8093
143885	71.63€	300.00€	17.7605
65058	61,933.51€	0.00€	100.0000
74864	11,050.17€	18,590.00€	100.0000
94640	54,344.69€	0.00€	100.0000
137369	9,089.13€	12,110.00€	100.0000
138271	2,394.69€	0.00€	100.0000
150294	44,118.27€	0.00€	100.0000

Created by: Author

After sorting our customers by their likelihood of churning, Kai felt extremely confident in a good number of cases which amounted to about 27 out of 180. For brevity and the sake of avoiding the unnecessary bloat we have taken the liberty of not showing all of them but focusing on the ones that were not 100 % likely to churn. It is safe to assume that the customer 105677 was likely a refunded purchase but because of recency the churn is not 100 % probable. Moreover, customers 93389 and 143885 are likely to be one-time buyers and it would be in company's best interest to turn them into repeat customers.

4.1 Recommendations for practical use

We envision the Company A using Kai as means to generate notifications about potential customer churn until they reach a certain customer lifetime value percentile. Having a limitation such as this one will avoid the inaccuracies Kai shows for outliers due to its' nature being a deep learning model. With the top 20 % customers being handled manually by a person, the remaining 80 could be left to be checked by Kai. Considering Company A is strictly a B2B business, and the number of customers is comparatively small in relation to a B2C business, we believe it to be reasonable to put this cutoff for Kai to avoid confusion with Kai's potential inaccuracy on the outliers of our dataset.

As mentioned previously, we could use the data Kai is providing us to advertise to each customer individually. This could be done any time Kai detects a likelihood of defection. Sales representatives could be sent out to customers with high enough customer lifetime value if they are deemed as likely to churn. Company A could implement an automatic email 5 % flash discount to individual customers once their defection gets over a certain threshold. Furthermore, a membership system could be implemented. This would mean customers with higher perceived value would get more benefits or even better prices.

The threshold would be adjusted as necessary. With more development, we could make it so that customers with a lower customer lifetime value would have their threshold higher than those with a higher customer lifetime value. Instinctively, it is in Company A's best interest to protect the customers who have spent or are likely to spend more money.

5 Conclusion

At the very beginning of the thesis, we have set out to build and train a machine learning model using neural networks which would be capable of predicting the customer lifetime value and churn rate of customers. Our research statement was: “By developing Kai, we intend to give Company A valuable information on their customers, as well as demonstration of business informatics combined with economics principles such as marketing and resource allocation in the real world.”

Kai has shown to be a moderately reliable model reaching upwards of 0.7 R^2 score and has shown ~ 78 % churn rate prediction accuracy. We believe that these results give valuable insight to Company A regarding their customer base. As such, we are under the impression that we have successfully completed the first half of our research statement.

Because of the information generated by Kai, we believe that we have also successfully fulfilled the second half of our research statement. This information can be used to create advertising specifically tailored to each individual customer based on their overall customer lifetime value and the likelihood of churning. Using the information Kai provides could end up in a more efficient resource allocation and drive down the potential acquisition cost for customers who would have to be convinced to come back. Furthermore, Kai could be used as means of generating small discounts that would incentivize the customer to purchase a product as well as provide benefits to customers Kai perceives as more valuable. On the contrary, customers who are expected to churn could be stopped from doing so by sending sales representatives to them, ultimately driving down the reacquisition cost.

Frequency being such an important part of our model, we could predict waves of customer purchases based on the time of the year, better yet, we could predict when each individual customer is likely to purchase. This key characteristic of our model ties back to the individually tailored advertisements, giving out a slightly better deal whenever the customer is most likely to complete a purchase. We could also implement a system suggesting more quantity and more expensive goods to the biggest spenders while suggesting more affordable solutions to others.

However, our machine learning model is far from flawless. Improvements could be made along the entire chain of operations. The data we have gotten from the Company A has had plenty of null values and false positives such as rewarehousing being a part of the document. There is a chance that some important information got lost during data cleaning, skewing the results by some, albeit minimal, amount. We have also used a dataset consisting of around 36000 entries which is on the smaller side for what's generally used to train neural networks.

Regarding Kai itself, a team of experts with deeper knowledge of deep learning would most definitely be able to build a more performant and highly optimised version of Kai. With our limited exposure to the world of deep neural networks we believe that we have built a solid proof of concept which could be improved into a corporate-ready version.

This thesis has set out to model and develop a machine learning model which could both predict customer lifetime value as well as the possible customer churn rate. We ended up choosing a dense neural network to accomplish this, referred to as Kai. Kai was meant to demonstrate the practical application of neural networks in business information systems. To train, we have used a dataset given to us from the Company A. In doing so we have sought to give insight into the customer base of Company A, benefitting them by raising alarms whenever a valuable customer is under threat of churning. This information would help Company A with targeted advertising and thus more efficient resource allocation.

Kai was trained on data spanning multiple years, specifically from 2021 reaching to the end of 2024. Year 2025 was omitted due to the general lack of data considering the acquisition date of the dataset. After clearing the data of null values and internal transactions, the model was trained on 250 epochs using a combination of an adaptive and cyclic learning rate to avoid overfitting while still achieving fast convergence. (Smith, 2017) Our neural network has achieved an R^2 score of 0.71 solidifying it as a moderately reliable machine learning model. Churn rate was predicted with a success rate of approximately 78 %.

In spite of this, Kai has shown several limitations. As mentioned before, the dataset had to be cleared of internal transactions, potentially removing important data in the process or keeping the undesirable data. While this is acceptable for our proof-of-concept model, it would be a glaring issue had it been a corporate-ready solution. This limitation alone could have severely limited the accuracy of Kai. Furthermore, the dataset itself was comparatively small in relation to what is used for dense neural network training.

For future research, it would be sensible to test how Kai performs on a larger dataset, ideally diverse through different market types. Exploring Kai in a B2C setting could accomplish similar purpose. Moreover, Kai could be improved by developing a different architecture, more suited to accommodate for the outliers. Exploring the possibility of cloud computing for scalability is something to consider as well.

In summary, this thesis has shown a practical application of machine learning within company information systems. While our model is not an enterprise-ready solution, it has shown to be a functional prototype with practical utility and clear results.

Bibliography

1. Codd, E.F., 1974. *Recent Investigations in Relational Data Base Systems* (pp. 1017-1021). IBM Thomas J. Watson Research Division.

2. Chollet, F., 2021. *Deep learning with Python*. 2nd ed. Shelter Island, NY: Manning Publications.

3. Duchi, J., Hazan, E. and Singer, Y., 2011. Journal of Machine Learning Research. *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*, *12*, pp.2121-2159.

4. Gupta, S., Hanssens, D., Hardie, B., Kahn, W., Kumar, V., Lin, N., Ravishanker, N. and Sriram, S., 2006. Modeling customer lifetime value. *Journal of service research*, *9*(2), pp.139-155.

5. Reinartz, W.J. and Kumar, V., 2003. The impact of customer relationship characteristics on profitable lifetime duration. *Journal of marketing*, *67*(1), pp.77-99.

6. Fader, P.S., Hardie, B.G. and Lee, K.L., 2005. RFM and CLV: Using iso-value curves for customer base analysis. *Journal of marketing research*, *42*(4), pp.415-430.

7. Filser, M., 1991. Segmentation in direct marketing channels: a transaction-patterns based method. *International Review of Retail, Distribution and Consumer Research*, *1*(2), pp.201-215.

8. Fleuret, F., 2023. *The Little Book of Deep Learning*. [online] Available at: https://fleuret.org/public/lbdl.pdf [Accessed 1 April 2025].

9. Friedman, J.H., 1991. Multivariate adaptive regression splines. *The annals of statistics*, *19*(1), pp.1-67.

10. **Gintis, H.** (2009) *Game theory evolving: A problem-centered introduction to modeling strategic interaction*. 2nd edn. Princeton, NJ: Princeton University Press.

11. Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep learning*. Cambridge, MA: MIT Press.

12. He, Juncai & Li, Lin & Xu, Jinchao & Zheng, Chunyue. (2018). ReLU Deep Neural Networks and Linear Finite Elements. 10.48550/arXiv.1807.03973.

13. Helland, I.S., 1987. On the interpretation and use of R2 in regression analysis. *Biometrics*, pp.61-69.

14. Huang, G., Liu, Z., van der Maaten, L. and Weinberger, K. Q., 2018. Densely Connected Convolutional Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, [online] Available at: https://doi.org/10.1109/CVPR.2017.243 [Accessed 1 April 2025].

15. Jaldin, F. and Hanicke, M., 2002. Strategy for customer expansion. *rapport nr.: Masters Thesis*, (2001).

16. **James, G., Witten, D., Hastie, T. and Tibshirani, R.** (2021) *An Introduction to Statistical Learning: with Applications in R*. 2nd edn. Springer. Available at: https://www.casact.org/sites/default/files/2022-12/James-G.-et-al.-2nd-edition-Springer-2021.pdf (Accessed: 4 April 2025).

17. Karunasingha, D.S.K., 2022. Root mean square error or mean absolute error? Use their ratio as well. *Information Sciences*, *585*, pp.609-629.

18. Kuksov, Dmitri, and J. Miguel Villas-Boas. "When more alternatives lead to less choice." *Marketing Science* 29, no. 3 (2010): 507-524.

19. Lambrecht, A. and Skiera, B., 2006. Paying too much and being happy about it: Existence, causes, and consequences of tariff-choice biases. *Journal of marketing Research*, *43*(2), pp.212-223.

20. **Leider, S.** (2019) 'Behavioral analysis of strategic interactions: game theory, bargaining, and agency', in Donohue, K., Katok, E., and Leider, S. (eds.) *The handbook of behavioral operations*. Hoboken, NJ: Wiley, pp. 237–285.

21. Lemon, K. N. and Verhoef, P. C. (2016) 'Understanding customer experience throughout the customer journey', *Journal of Marketing*, 80(6), pp. 69–96. doi: 10.1509/jm.15.0420

22. Li, J. and Yang, X., 2020, October. A cyclical learning rate method in deep learning training. In *2020 International Conference on*

Computer, Information and Telecommunication Systems (CITS) (pp. 1-5). IEEE.

23. Narayanan, D., Harlap, A., Phanishayee, A., Seshadri, V., Devanur, N.R., Ganger, G.R., Gibbons, P.B. and Zaharia, M., 2019, October. PipeDream: Generalized pipeline parallelism for DNN training. In *Proceedings of the 27th ACM symposium on operating systems principles* (pp. 1-15).

24. Neslin, S.A., Grewal, D., Leghorn, R., Shankar, V., Teerling, M.L., Thomas, J.S. and Verhoef, P.C., 2006. Challenges and opportunities in multichannel customer management. *Journal of service research*, *9*(2), pp.95-112.

25. Nguyen, Q. H., Ly, H.-B., Pham, B. T., et al., 2021. Influence of Data Splitting on Performance of Machine Learning Models in Prediction of Shear Strength of Soil. *Mathematical Problems in Engineering*, [online] Available at: https://doi.org/10.1155/2021/4832864

26. **Nielsen, M.** (n.d.) *Neural Networks and Deep Learning*. Available at: https://neuralnetworksanddeeplearning.com (Accessed: 11 March 2025).

27. Schmittlein, D.C., Morrison, D.G. and Colombo, R., 1987. Counting your customers: Who-are they and what will they do next?. *Management science*, *33*(1), pp.1-24.

28. Shim, S., Eastlick, M.A., Lotz, S.L. and Warrington, P., 2001. An online prepurchase intentions model: the role of intention to

search: best overall paper award—The Sixth Triennial AMS/ACRA Retailing Conference, 2000☆. *Journal of retailing*, *77*(3), pp.397-416.

29. Smith, L.N., 2017. *Cyclical learning rates for training neural networks*. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. [online] IEEE, pp.464–472. Available at: https://arxiv.org/abs/1506.01186 [Accessed 8 Apr. 2025].

30. Swathika, R., Kumar, S.D., Srinidhi, N.N. and Harshitha, B.R., 2024. Harnessing Learn Rate Schedule for Adaptive Deep Learning in LoRaWAN-IoT Localization. *IEEE Access*.

31. Vatankhah, S. and Samizadeh, R., 2024. Considering customer lifetime network value in oligopoly markets with the use of game theory approach. *Scientia Iranica*, 31(22), pp. 2112–2127. Available at: https://doi.org/10.24200/sci.2023.56345.4675.

32. Ang, L. and Buttle, F., 2006. Managing for successful customer acquisition: An exploration. *Journal of Marketing Management*, *22*(3-4), pp.295-317.

33. Porter, M.E., 2008. The five competitive forces that shape strategy. *Harvard business review*, *86*(1), p.78.

Appendix A Example data

Custom-erID	Country	In-voiceNo	Ex-tended Day	Year	Quan-tity	Reve-nue
45276	MG Madagas-kar	20310400	Feb, 12	2020	1670	7690
244642	DE Nemecko	23310199	Jan, 31	2023	1060	10590
244642	DE Nemecko	23310200	Jan, 31	2023	40	420
244642	DE Nemecko	23310201	Jan, 31	2023	350	3370
244642	DE Nemecko	23310202	Jan, 31	2023	8060	78120
41718	PL Poľsko	23310298	Feb, 15	2023	1270	5780
73318	CZ Česká repub-lika	21311777	Jul, 13	2021	560	3660
73318	CZ Česká repub-lika	21311777	Jul, 13	2021	560	3090
73318	CZ Česká repub-lika	21311896	Jul, 27	2021	1110	6340
73318	CZ Česká repub-lika	21311896	Jul, 27	2021	1130	6710
180759	PL Poľsko	21310509	Feb, 24	2021	560	2590
51111	UA Ukrajina	22310073	Jan, 10	2022	570	3950
101470	PL Poľsko	23310369	Feb, 24	2023	560	3780
103416	IT Taliansko	24310540	Mar, 28	2024	10860	32140
62486	IT Taliansko	21310594	Mar, 4	2021	5400	21910
113497	DE Nemecko	22312550	Nov, 18	2022	690	5740
100788	US USA	21310482	Feb, 22	2021	50	360
73318	CZ Česká repub-lika	21311834	Jul, 20	2021	1820	7680
123993	PL Poľsko	20311874	Sep, 28	2020	450	1000
123993	PL Poľsko	20311874	Sep, 28	2020	320	710
200008	SK Slovensko	23311188	Jul, 7	2023	10000	24280
121051	CZ Česká repub-lika	21310489	Feb, 23	2021	100	750
121051	CZ Česká repub-lika	21310647	Mar, 9	2021	280	1420
121051	CZ Česká repub-lika	21310647	Mar, 9	2021	160	780
121051	CZ Česká repub-lika	21310647	Mar, 9	2021	380	1920

143858	TR Turecko	21310344	Feb, 8	2021	1090	6880
183918	CZ Česká republika	20311733	Sep, 8	2020	470	2700
183918	CZ Česká republika	20311734	Sep, 8	2020	90	500
84417	PL Poľsko	20311910	Oct, 1	2020	40	170
168408	PL Poľsko	20312512	Dec, 9	2020	6650	7320
168408	PL Poľsko	20312512	Dec, 9	2020	480	530
121051	CZ Česká republika	20311882	Sep, 29	2020	320	1220
121051	CZ Česká republika	20311882	Sep, 29	2020	340	1410
121051	CZ Česká republika	20312077	Oct, 20	2020	270	1020
121051	CZ Česká republika	20312077	Oct, 20	2020	160	680
121051	CZ Česká republika	20312227	Nov, 3	2020	170	640
121051	CZ Česká republika	20312227	Nov, 3	2020	170	700
121051	CZ Česká republika	20312318	Nov, 16	2020	350	1340
121051	CZ Česká republika	20312318	Nov, 16	2020	370	1510
59500	US USA	21310988	Apr, 16	2021	11180	90360
59500	US USA	21312486	Oct, 4	2021	14000	121600
59500	US USA	21312772	Oct, 29	2021	14000	128590
227054	BR Brazília	22310834	Mar, 30	2022	14000	113040
227054	BR Brazília	22311542	Jun, 24	2022	14000	125020
227054	BR Brazília	23310306	Feb, 16	2023	12600	112930
227054	BR Brazília	23310306	Feb, 16	2023	300	2690
70489	FR Francúzsko	21312040	Aug, 16	2021	1830	13970
70489	FR Francúzsko	21312583	Oct, 12	2021	1400	11130
70489	FR Francúzsko	21313031	Dec, 3	2021	1400	11130
70489	FR Francúzsko	22310895	Apr, 4	2022	1400	11860
70489	FR Francúzsko	22311369	Jun, 3	2022	1400	11860
70489	FR Francúzsko	22311781	Jul, 22	2022	1400	11860
70489	FR Francúzsko	22312697	Dec, 9	2022	1400	12840
70489	FR Francúzsko	23310238	Feb, 6	2023	700	6420
70489	FR Francúzsko	23310403	Mar, 3	2023	2100	19260
70489	FR Francúzsko	24310022	Jan, 4	2024	1400	11730

Appendix B Code snippets

B.1 Learning rate

```
class CyclicLR(keras.callbacks.Callback):
    def __init__(self, base_lr=1e-6, max_lr=1e-4,
step_size=2000):
        super().__init__()
        self.base_lr = base_lr
        self.max_lr = max_lr
        self.step_size = step_size
        self.iterations = 0

    def on_train_batch_begin(self, batch, logs=None):
        cycle = np.floor(1 + self.iterations / (2 *
self.step_size))
        x = np.abs(self.iterations / self.step_size - 2
* cycle + 1)
        lr = self.base_lr + (self.max_lr -
self.base_lr) * np.maximum(0, (1 - x))
        self.model.optimizer.learning_rate.assign(lr)
        self.iterations += 1
```

B.2 Removing Outliers

```
def remove_outliers(df, column):
    Q1 = df[column].quantile(0.005)
    Q3 = df[column].quantile(0.925)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] >= lower_bound) & (df[column]
<= upper_bound)]
```

B.3 Evaluation

```
def evaluate(actual, predictions,
y_test_churn_filtered, churn_preds_class,
churn_history=None):
    if actual.empty or predictions.size == 0:
```

```
        raise ValueError("Evaluation dataset is
empty!")

    MAE = mean_absolute_error(actual, predictions)
    MSE = mean_squared_error(actual, predictions)
    RMSE = np.sqrt(MSE)
    print(f"Total Sales Actual:
{np.round(actual.sum())}")
    print(f"Total Sales Predicted:
{np.round(predictions.sum())}")
    print(f"Individual R2 score: {r2_score(actual,
predictions)}")
    print(f"Individual Mean Absolute Error: {MAE}")
    print(f"Root Mean Squared Error: {RMSE}")
    print(f"RMSE/MAE ratio: {RMSE/MAE}")

    if len(y_test_churn_filtered) > 0:
        accuracy =
accuracy_score(y_test_churn_filtered,
churn_preds_class)
        print("Churn Model Accuracy:", accuracy)
    else:
        print("No samples remaining for churn
evaluation")

    # Churn Accuracy
    plt.figure(figsize=(6, 5))
    if churn_history is not None:
        plt.plot(churn_history.history['val_accuracy'],
label='Validation Accuracy')
        plt.title('Churn Model Accuracy')
        plt.xlabel('Epochs')
        plt.ylabel('Accuracy')
        plt.legend()
    plt.tight_layout()
    plt.show()

    # KDE and Binned Bar Chart together
    plt.figure(figsize=(15, 5))

    # KDE Plot
    plt.subplot(1, 2, 1)
    bins = np.linspace(0, np.percentile(actual, 90),
10)
    sns.kdeplot(actual, color='blue', label='Actual',
fill=True, bw_adjust=0.5)
```

```
    sns.kdeplot(predictions, color='red',
label='Predicted', fill=True, bw_adjust=0.5)
    plt.xlim(bins[0] - 50000, bins[-1])
    plt.xlabel('CLV Value')
    plt.ylabel('Density')
    plt.title('KDE Distribution of Actual vs Predicted
CLV')
    plt.legend()

    # Binned Bar Chart
    plt.subplot(1, 2, 2)
    actual_binned = np.histogram(actual, bins=bins)[0]
    predicted_binned = np.histogram(predictions,
bins=bins)[0]
    bin_labels = [f"{int(bins[i])}-{int(bins[i+1])}"
for i in range(len(bins)-1)]
    width = 0.4
    x = np.arange(len(bin_labels))

    plt.bar(x - width/2, actual_binned, width=width,
label='Actual', color='blue', alpha=0.7)
    plt.bar(x + width/2, predicted_binned, width=width,
label='Predicted', color='red', alpha=0.7)
    plt.xticks(x, bin_labels, rotation=45)
    plt.xlabel('CLV Value Range')
    plt.ylabel('Customer Count')
    plt.title('Binned Bar Chart of Actual vs Predicted
CLV')
    plt.legend()

    plt.tight_layout()
    plt.show()
    show_top_customers(filtered_df, churn_preds)
```

B.4 Show top customers

```
def show_top_customers(filtered_df, churn_preds,
top_n=10):
    churn_probs = pd.Series(churn_preds.ravel(),
index=X_test.index)
    churn_probs = churn_probs.loc[filtered_df.index]

    display_df = filtered_df.copy()
    display_df['churn_rate_%'] = (churn_probs *
100).round(4)
```

```
    curr_format = lambda x: f"{x:,.2f}€"
    display_df['predicted_clv'] =
display_df['dnn_preds'].apply(curr_format)
    display_df['actual_clv'] =
display_df['actual'].apply(curr_format)

    top_clv = display_df.sort_values('dnn_preds',
ascending=False).head(top_n)

    median_value = display_df['actual'].median()
    display_df['median_diff'] = (display_df['actual'] -
median_value).abs()
    median_clv =
display_df.sort_values('median_diff').head(top_n)

    high_churn = display_df[(display_df['churn_rate_%']
> 0) &

(display_df['churn_rate_%']!=
100)].sort_values('churn_rate_%',
ascending=False).head(top_n)
```