DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

Churn prediction using time series data

Customer churn in bank and insurance services

PATRICK GRANBERG

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Churn prediction using time series data

PATRICK GRANBERG

Master in Computer Science
Date: December 21, 2020
Supervisor: Mats Nordahl (KTH), Rasmus Persson (ICA Banken)
Examiner: Erik Fransén
School of Electrical Engineering and Computer Science
Host company: ICA Banken
Swedish title: Prediktion av kunduppsägelser med hjälp av tidsseriedata

Abstract

Customer churn is problematic for any business trying to expand their customer base. The acquisition of new customers to replace churned ones are associated with additional costs, whereas taking measures to retain existing customers may prove more cost efficient. As such, it is of interest to estimate the time until the occurrence of a potential churn for every customer in order to take preventive measures. The application of deep learning and machine learning to this type of problem using time series data is relatively new and there is a lot of recent research on this topic. This thesis is based on the assumption that early signs of churn can be detected by the temporal changes in customer behavior. Recurrent neural networks and more specifically long short-term memory (LSTM) and gated recurrent unit (GRU) are suitable contenders since they are designed to take the sequential time aspect of the data into account. Random forest (RF) and stochastic vector machine (SVM) are machine learning models that are frequently used in related research. The problem is solved through a classification approach, and a comparison is done with implementations using LSTM, GRU, RF, and SVM. According to the results, LSTM and GRU perform similarly while being slightly better than RF and SVM in the task of predicting customers that will churn in the coming six months, and that all models could potentially lead to cost savings according to simulations (using non-official but reasonable costs assigned to each prediction outcome). Predicting the time until churn is a more difficult problem and none of the models can give reliable estimates, but all models are significantly better than random predictions.

Keywords: churn time prediction, classification, lstm, gru, rf, svm

Sammanfattning

Kundbortfall är problematiskt för företag som försöker expandera sin kundbas. Förvärvandet av nya kunder för att ersätta förlorade kunder är associerat med extra kostnader, medan vidtagandet av åtgärder för att behålla kunder kan visa sig mer lönsamt. Som så är det av intresse att för varje kund ha pålitliga tidsestimat till en potentiell uppsägning kan tänkas inträffa så att förebyggande åtgärder kan vidtas. Applicerandet av djupinlärning och maskininlärning på denna typ av problem som involverar tidsseriedata är relativt nytt och det finns mycket ny forskning kring ämnet. Denna uppsats är baserad på antagandet att tidiga tecken på kundbortfall kan upptäckas genom kunders användarmönster över tid. Reccurent neural networks och mer specifikt long short-term memory (LSTM) och gated recurrent unit (GRU) är lämpliga modellval eftersom de är designade att ta hänsyn till den sekventiella tidsaspekten i tidsseriedata. Random forest (RF) och stochastic vector machine (SVM) är maskininlärningsmodeller som ofta används i relaterad forskning. Problemet löses genom en klassificeringsapproach, och en jämförelse utförs med implementationer av LSTM, GRU, RF och SVM. Resultaten visar att LSTM och GRU presterar likvärdigt samtidigt som de presterar bättre än RF och SVM på problemet om att förutspå kunder som kommer att säga upp sig inom det kommande halvåret, och att samtliga modeller potentiellt kan leda till kostnadsbesparingar enligt simuleringar (som använder icke-officiella men rimliga kostnader associerat till varje utfall). Att förutspå tid till en kunduppsägning är ett svårare problem och ingen av de framtagna modellerna kan ge pålitliga tidsestimat, men alla är signifikant bättre än slumpvisa gissningar.

Acknowledgements

I would like to thank my supervisor at ICA Banken, Rasmus Persson, for everything from helping me with data extraction to our discussions regarding various implementations. I would also like to express my gratitude to ICA Banken for allowing me the opportunity to use their data for research on a very interesting topic.

I would like to thank my supervisor at KTH, Mats Nordahl, and my examiner, Erik Fransén, for their constructive feedback that helped ensure the quality of the content in this thesis.

Finally, I would like to thank my family and friends for their support and encouragement which gave me the motivation to do my best.

Patrick Granberg
Stockholm, December 2020

Contents

1 Introduction **1**
- 1.1 Problem statement . . . 2
- 1.2 Goals . . . 2
- 1.3 Research question . . . 2
- 1.4 Delimitations . . . 3
- 1.5 Thesis outline . . . 3

2 Background **4**
- 2.1 Customer churn . . . 4
- 2.2 Survival analysis . . . 5
  - 2.2.1 Censoring . . . 6
  - 2.2.2 Cox proportional hazards . . . 7
  - 2.2.3 Kaplan-Meier estimator . . . 8
- 2.3 Deep learning . . . 8
  - 2.3.1 Artificial Neural Network . . . 8
  - 2.3.2 Recurrent Neural Network . . . 9
  - 2.3.3 Long Short-Term Memory . . . 10
  - 2.3.4 Gated Recurrent Unit . . . 10
  - 2.3.5 Earth Mover's Distance . . . 11
- 2.4 Machine learning . . . 11
  - 2.4.1 Decision Tree . . . 11
  - 2.4.2 Random Forest . . . 13
  - 2.4.3 Support Vector Machine . . . 13
- 2.5 Related work . . . 14

3 Method **18**
- 3.1 Data . . . 19
  - 3.1.1 Datapoint creation . . . 28
  - 3.1.2 Sampling . . . 31

3.1.3 Analysis of data . . . 32
3.2 Implementation . . . 35
3.3 Hyperparameter tuning . . . 38
3.4 Evaluation metrics . . . 39
3.5 Significance and effect size . . . 42

4 Results **44**
4.1 Model performance . . . 44
4.1.1 Churn prediction . . . 44
4.1.2 Churn time prediction . . . 49
4.2 Significance test . . . 52

5 Discussion **54**
5.1 Results . . . 54
5.1.1 Churn prediction . . . 54
5.1.2 Churn time prediction . . . 55
5.2 Future work . . . 55
5.3 Sustainability and ethics . . . 56

6 Conclusions **59**

Bibliography **61**

A Customer characteristics features **67**

B Customer segments features **68**

C Product groups features **69**

D Products features **70**

E Monetary features **71**

List of Figures

2.1 A figure illustrating censored subjects in an observation. . . . 7
2.2 A graph representation of a recurrent neural network. The input is denoted x and consist of several timesteps, the output is denoted y, the weights are denoted W, and the activation function is denoted h. The recurrent neural network graph with cycles is displayed on the left side, on the right side is the unfolded graph. 9
2.3 Example of a decision tree, illustrating a protocol for issuing a credit card to an applicant. 12

3.1 A figure showing the month at which customers entered the observation. 21
3.2 The duration that churners are active customers in the observation. 22
3.3 The duration that non-churners are active customers in the observation. 22
3.4 The number of customers that churned at a given month during observation. 23
3.5 Histogram of occurred churns per customer. 23
3.6 The real customer growth compared to the imaginary case where no customers churn. 24
3.7 Before and after comparison of the number of customers who satisfy the conditions set on the data. 25

3.8 Visual representation of customer data on a chronological timescale where month 1 represents the first month in the observed period. The data has been been normalized. A rough estimation of the features is as follows; Starting from the left, customer characteristics (0-5), customer segments (6-19), product groups (20-33), products (34-94), and monetary (95-106). In reality, the last two features are "observed time" and "moved" from the customer characteristic category. 27
3.9 A timeline of active customers during the observation. 29
3.10 A right-aligned timeline of active customers. The seven most recent months of non-churners are unusable and discarded. . . 29
3.11 Illustration of two sliding windows over the dataset with respect to the target variable (time-to-churn). 30
3.12 An illustration of oversampling and undersampling. 32
3.13 Scree plot that explain the variance of the principal components. 33
3.14 A 2d plot with respect to the two most significant PCA components. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes. 33
3.15 A 3d plot with respect to the three most significant PCA components. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes. 34
3.16 A 2d plot after applying t-SNE. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes. 35
3.17 A table of the predicted outcomes. 39
3.18 Varying the threshold to achieve different classification results. 40

4.1 A graph comparing the ROC-AUC of different models. 45
4.2 A graph comparing the average precision of different models. . 46
4.3 A graph of the precision and recall of every model for every threshold. 47
4.4 The cost associated with every model over all threshold levels. 48
4.5 Calculated profit compared to using no model. 48

4.6 Top 100 most important features learned by RF divided into sections of feature types. 51

List of Tables

3.1 A compact description of feature categories. 19
3.2 LSTM and GRU hyperparameter search. 38
3.3 RF hyperparameter search. *Churn time prediction parameters. 39
3.4 SVM hyperparameter search. *Churn time prediction parameters. 39
3.5 A conversion table of effect size according to Cohen's d. . . . 43

4.1 A table of binary churn prediction results. 45
4.2 Assigned costs to every binary prediction outcome. 48
4.3 A table of churn time prediction results. 49
4.4 A table of churn time prediction results when excluding features. 50
4.5 Description of feature types as seen in Figure 4.6. 51
4.6 A table of statistical significance and effect size results for the binary churn predictors. A Tukey HSD p-value of less than 0.01 suggests a significant difference between the models. . . 53
4.7 A table of statistical significance and effect size results for the churn time predictors. A Tukey HSD p-value of less than 0.01 suggests a significant difference between the models. 53

A.1 Customer characteristics features. 67

B.1 Customer segments features. 68

C.1 Product groups features. 69

D.1 Products features. 70

E.1 Monetary features. 71

Chapter 1

Introduction

Application of deep learning to business specific problems is a current trend sparked by the recent popularization and availability of deep learning frameworks. This revolution of deep learning has brought forth new possibilities to explore alternative solutions to already established methods as well as entirely new innovations only made possible by technological advancements.

Survival analysis has traditionally been solved by statistical methods (most notably Cox regression, sometimes called proportional hazards regression) and has predominantly been used in estimating lifetimes of people or products, hence the name survival analysis [1][2][3]. Since the usage of deep learning in the context of survival analysis is a relatively recent innovation, there is still a lot of room for research within this area.

Highly relevant to survival analysis is the concept of customer churn. It differs from survival analysis in the sense that prediction of churn answers the question if a customer will stop using the company's products (within a fixed time) while survival analysis is concerned with the time aspect. In other words, survival analysis gives an answer to the question of how long it takes before a churn will occur. For a business, this could be crucial information needed to enable further expansion and minimize economical loss associated with overhead costs from customer retention strategies. While seemingly a simple problem, the hidden complexity lies in incomplete data and the unpredictability of human beings.

Recent research in churn prediction makes use of the time-varying features in customer data by using recurrent neural networks (RNN), and more specifically, architectures such as long short-term memory (LSTM) and gated recurrent unit (GRU) [4]. However, Hassani et al. [5] note that the application of deep learning in banking is relatively limited considering the wide scale

of customer relationship management (CRM) in the sector. Other machine learning techniques are often used and include for example stochastic vector machine (SVM) and random forest (RF).

This thesis aims to examine prominent machine learning and deep learning techniques applied to survival analysis and churn prediction in the banking and insurance service sector. The work was done at ICA Banken, a Swedish bank that provided an anonymous dataset for the experiments.

1.1 Problem statement

Churning customers are sudden and problematic in a business sense. ICA Banken offers many different kinds of services one expects of a bank. The repertoire consists of bank accounts, credit cards, funds, insurances and loans. Those are the product categories in a bigger sense, each with a different number of sub-products. There are no set subscription periods for any of the products. As such, a customer can decide to terminate their services at any time. This makes it more difficult to intuitively understand when a customer might churn.

1.2 Goals

The desired outcome is a solution that can give reliable estimates of the remaining time until a customer might churn.

1.3 Research question

How well do different deep learning and machine learning-based solutions compare to each other in regard to predicting customer churn and estimating the time to churn?

Sub-questions

- Can classification based predictions give reliable results?
- Do RNNs such as LSTM and GRU have an advantage over machine learning models?

1.4 Delimitations

The scope of this thesis is limited by some constraints as defined below.

- A reasonable timeframe for taking action to mitigate churn is estimated to be within six months, meaning that a relatively high precision is preferred within this crucial timeframe.
- There will be no taking into account recurring termination of contracts due to the very few datapoints displaying this characteristic. The customers with multiple churns are discarded from the dataset.
- The assumption is that there exists temporal information that can be used for churn prediction. Therefore, datapoints with less than 13 months of history are discarded from the dataset.
- Only a small subset of all customer data will be used due to hardware constraints and the confidential nature of the data.

1.5 Thesis outline

Chapter 2 (Background) introduces the reader to the algorithms and metrics used in this thesis as well as a section describing some relevant work. Chapter 3 (Method) starts with describing the dataset, how features were selected and engineered followed by some data analysis, details of the developed models, and the evaluation metrics. Chapter 4 (Results) presents the results of the experiments. Chapter 5 (Discussion) contains a discussion of the results, suggestions of future work, and a discourse on sustainability and ethics. Chapter 6 (Conclusions) summarizes the main findings of this thesis.

Chapter 2

Background

Some noteworthy definitions and theoretical frameworks related to customer churn, survival analysis, deep learning, machine learning, and evaluation metrics are introduced in order to get a better understanding of the models developed.

2.1 Customer churn

Customer churn is a term describing the loss of customers. It is the event of a customer ceasing the use of a service or product that the company has to offer [6]. There are many definitions of churn that depend on the business model. Churn can for example be either recurring or only happen once, marking the end of a contract. There are even many finer nuances to all of this depending on the context. Given this somewhat loose formulation, there is a need to strictly define what it means in the context of this thesis. Hence, churn at ICA Banken was defined to occur when a customer terminates all of their contracts.

In a bigger context, it is common to look not only at individual customers but all of them. For this reason, we measure the churn rate or attrition rate, which is the rate at which customers terminate their services with a company. For a growing business, this would be lower than the pace at which new ones are acquired. Conversely, the retention rate measures the fraction of customers retained. According to Van Den Poel and Larivière [7], even the smallest increase of a single percentage may yield a substantial profit increase, a claim further strengthened by Zhao and Dang [8]. Similarly, according to Harvard Business Review, a decrease in customer defection rate of 5% can result in a 25%-85% increase in profits [9].

Companies may implement various strategies as a countermeasure to high

churn rates. However, sending out non targeted incentives to the wrong customers risks doing more harm than good for profits. Therefore, a targeted strategy is preferred [10].

According to Bhattacharya [11], the acquisition of new customers can be as much as six times more costly than retaining existing customers.

In an article by Bahnsen et al. [12], they argue that a churn model has three main points to fulfill. Avoiding false positives that adds cost with unnecessary offers, presenting an appropriate incentive to potential churners such that profit is maximized, and keeping the number of false negatives low.

2.2 Survival analysis

Survival analysis encompasses a collection of methods with the purpose of estimating the time until the occurrence of a specific event. In medical contexts it is often the death of a patient that is of concern, but the methods are well used in other fields as well [13][14]. It is also commonly used for estimating the failure of mechanical components, but can be applied to practically anything as long as there are observations of measured time until some sort of event.

The survival function denoted $S(t)$ gives the probability that the subject survives beyond time t. T is a continuous random variable with $F(t)$ as its cumulative distributive function (CDF). According to Equation 2.1, the survival function is constantly decreasing in the range from 1 to 0 as t goes from 0 to ∞.

$$S(t) = pr(T \geq t) = 1 - F(t) = \int_t^\infty f(x)dx \tag{2.1}$$
$$0 \leq t < \infty$$

Generally, any distribution could be used to represent $F(t)$ and a suitable one is usually decided by area knowledge of the event distributions. Commonly used ones include; Exponential, Weibull, Log-normal, Log-logistic, Gamma, and Exponential-Logarithmic distributions.

The hazard function λ in Equation 2.2 denotes the failure rate at which the studied observations experiences the event. The rate may increase or decrease as t increases. A hazard function must satisfy the two conditions of Equation 2.3 and Equation 2.4.

$$\lambda(t) = \lim_{dt \to 0} \frac{pr(t \leq T < t + dt)}{dt \cdot S(t)} = \frac{f(t)}{S(t)} = -\frac{S'(t)}{S(t)} \tag{2.2}$$

$$\forall u \geq 0,\ (\lambda(u) \geq 0) \tag{2.3}$$

$$\int_0^\infty \lambda(u)du = \infty \tag{2.4}$$

The cumulative hazard function denoted Λ describes the failure distribution and can be expressed through Equation 2.5. It is a non-decreasing function.

$$\Lambda(t) = \int_0^t \lambda(u)du \tag{2.5}$$

The survival function and hazard function are closely related and a convenient formula can be derived from parts of Equation 2.1 and 2.2.

$$\lambda(t) = \frac{f(t)}{S(t)} = -\frac{S'(t)}{S(t)} = -\frac{d}{dt}\, logS(t) \tag{2.6}$$

$$S(t) = exp\{-\int_0^t \lambda(u)du\} = exp\{-\Lambda(t)\} \tag{2.7}$$

Churn prediction can essentially be seen as a subset of survival analysis. In churn prediction, the time t is set to a fixed value, and the goal is to determine if the subject lives past time t or not, which still adheres to the definition of the survival function $S(t)$. Looking at churn this way, it can be modeled as a binary classification problem.

2.2.1 Censoring

Censoring is common in survival analysis, occurring when the time to events are not fully observable. The most common type is called *right censoring*, it is when the event expected at the end of a subject timeline has yet to occur. Contrary to that is *left censoring*, where events happened before the observation started. *Interval censoring* is when an event happens at some point between two observations.

Take for example the standard case of a clinical trial in which the death of patients is to be observed. During the observed time, there may have been some deaths while the majority still live at the end of the observation, resulting in censored records since there is no information about how long these persons might live. Censoring is illustrated in Figure 2.1, where the observed period has been limited to seven years from the beginning of the study.

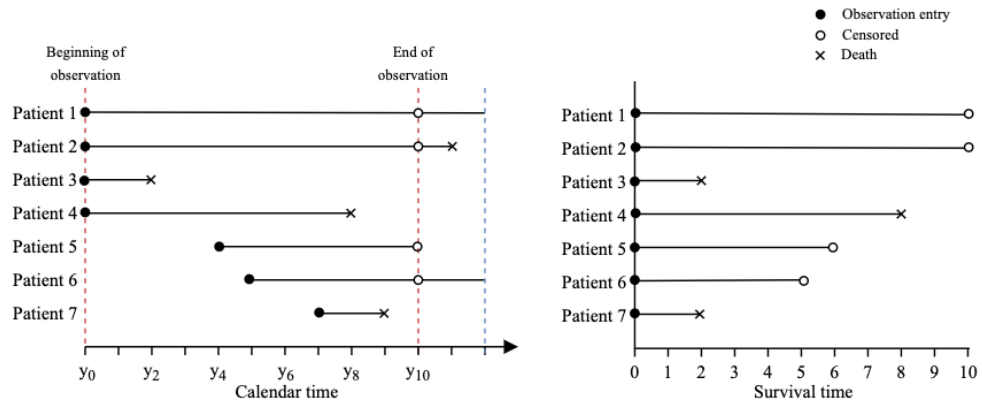

Figure 2.1: A figure illustrating censored subjects in an observation.

The representation of a subject δ_i is determined by the minimum of the elapsed time T_i for that subject and the censoring time C_i.

$$\delta_i = min(T_i, C_i) \tag{2.8}$$

Another characteristic of survival data is how the observations don't necessarily begin at the same point in time, and are therefore often shifted to the left in order to accommodate a common time axis starting on zero.

2.2.2 Cox proportional hazards

In 1972 Cox released the paper *Regression Models and Life-Tables* in which a novel model for estimating lifetimes of objects is proposed [1]. This technique (also called *Proportional hazards regression*), is highly used in survival analysis to this day [13]. Cox's work introduces a hazard function that incorporates the age-specific failure-rate associated with the aging of the subject.

$$\lambda(t) = \lambda_0(t)\, exp(\beta^T x) \tag{2.9}$$

$$log(\lambda(t)) = \lambda_0(t) + \beta^T x \tag{2.10}$$

In Equation 2.9 and 2.10 the covariates x and coefficients β for a subject δ_i are both vectors. The function $\lambda_0(t)$ is called the baseline hazard and can be disregarded since it is the same for any subject and can therefore be considered constant and canceled out. This is a useful property since in for example a Weibull model the hazard function $\lambda_0(t) = c \cdot log(t)$ can be disregarded. The

property implies that any difference in hazard between subjects solely depends on each subject's covariates. The two hazard functions in Equation 2.11 are proportional, an effect of the proportional hazards assumption.

$$\frac{\lambda_1(t)}{\lambda_2(t)} = \frac{\lambda_1(t)\, exp(x_1^T \beta)}{\lambda_2(t)\, exp(x_2^T \beta)} = exp((x_1^T - x_2^T)\, \beta) \tag{2.11}$$

2.2.3 Kaplan-Meier estimator

The Kaplan-Meier estimator is a non-parametric estimation of the survival function published in 1958 by Kaplan and Meier [15]. The function is described by Equation 2.12, which through the accumulation of estimations over t produces a curve that is true to the real survival function. Here, t_i are the ordered points in time after at least one event has occurred, d_i is the number of such events and n_i denotes the remaining samples that did not yet experience the event.

$$\widehat{S}(t) = \prod_{i:t_i<t} (1 - \frac{d_i}{n_i}) \tag{2.12}$$

2.3 Deep learning

Neural networks are the backbone of deep learning with varying types having emerged for different purposes. This chapter is a summary of some neural network types that through previous research have proved promising in solving the task of churn prediction and survival analysis.

2.3.1 Artificial Neural Network

Artificial neural networks are loosely modeled to simulate the neurons of biological brains. A neuron (also referred to as a node), can take several signal inputs of which it produces an output that can be passed on to other neurons. Signals initially correspond to feature data. Every connection of neurons has a corresponding weight representing its importance and an activation function that determines its output. A final output can be decided after all signals have passed through the chain of connected neurons.

Neurons are divided into sectional groups called layers. Each layer is connected to another layer in a structured manner according to some specifications. For example, a fully connected layer connects all neurons between two

layers, while a recurrent network allows for cyclic connections with both previous layers and the current layer. Neurons that have no incoming connections are called input nodes, while neurons with no outgoing connections are called output nodes. Neurons that have both are called hidden nodes, collectively referred to as a hidden layer. The network learns through optimization of its weights which is achieved by minimizing a cost function during model training.

2.3.2 Recurrent Neural Network

Recurrent neural networks allow for cycles to form between its hidden units. This gives RNNs the capabilities of keeping an internal state memory of its previous inputs, which is suitable for modeling sequential data.

The sequential predictions of an RNN are formed by the recurrent input of the networks output at every timestep. Recurrent neural networks need to be unfolded into a directed acyclic graph in order to simplify calculations of the gradient during backpropagation. It is then possible to use standard learning procedures used in regular feedforward architectures when training the network [16]. The unfolded architecture reveals that a hidden unit state is affected by preceding ones. A graph representation of recurrent neural networks can be seen in Figure 2.2.

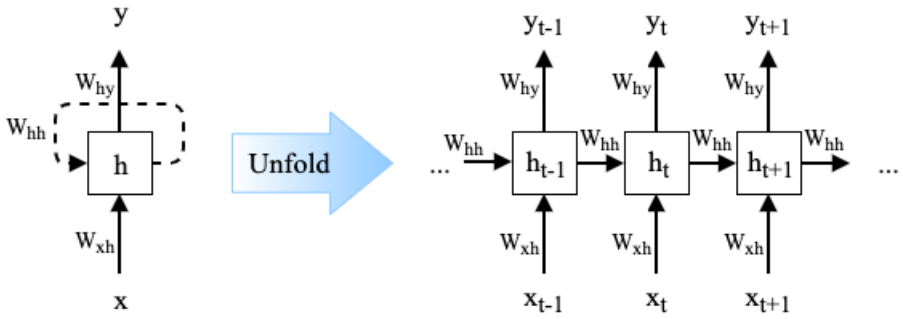

Figure 2.2: A graph representation of a recurrent neural network. The input is denoted x and consist of several timesteps, the output is denoted y, the weights are denoted W, and the activation function is denoted h. The recurrent neural network graph with cycles is displayed on the left side, on the right side is the unfolded graph.

2.3.3 Long Short-Term Memory

Introduced by Hochreiter and Schmidhuber [17] in 1997, the LSTM architechture is designed to learn long-term dependencies and therefore has the capability to store its memory for longer periods of time than the standard RNN. The difference lies in the LSTM cell.

The LSTM cell has some additional components and a more complex structure made up of the so-called forget, input, and output gates. It also has a cell state in addition to the hidden state that is updated after every gate. The forget gate decides is what information to forget, the input gate affects the importance of values to update and the output gate updates the hidden state of the cell. Operations of the LSTM cell are described in Equation 2.13 to 2.18.

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{2.13}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{2.14}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{2.15}$$

$$\tilde{c}_t = \sigma_g(W_c x_t + U_c h_{t-1} + b_c) \tag{2.16}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{2.17}$$

$$h_t = o_t \odot \sigma_h(c_t) \tag{2.18}$$

2.3.4 Gated Recurrent Unit

Gated recurrent unit is another variation of a recurrent neural network introduced in 2014 by Chung et al. [18]. The structure of the GRU cell is in many ways similar to LSTM. Inside the GRU cell are only two components, the reset and update gates. The reset gate decides what information to forget and what information to add, while the reset gate affects how much of the past information to forget. Opposed to the LSTM cell, GRU has no cell state and instead uses the hidden state for the same purpose. As a result of its simpler structure, training times are generally faster than LSTM. Operations of the GRU cell are described in Equation 2.19 to 2.22.

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \tag{2.19}$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \tag{2.20}$$

$$\hat{h}_t = \phi_h(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \tag{2.21}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{2.22}$$

2.3.5 Earth Mover's Distance

One-hot-encoding is often used in a multi-label classification problem to represent the class to which the data belongs. Also commonly used is the categorical cross-entropy loss for optimizing the predictions. During training and classification, each prediction consists of a probability distribution spread across all classes. Ideally, the correct class will have the highest value. Categorical cross-entropy does not take the relationship between classes into account.

For example, consider the correct class to be five where the model gives a guess of four. This is incorrect, but a guess of one is just as wrong in an unordered problem. In an ordered problem, a guess of one should be punished more which would theoretically push the distribution closer to class five.

Earth Mover's Distance (EMD) does just that. It redistributes the probabilities of a guess, minimizing the distance between the correct class and predicted class.

$$EMD = \sum_{i=1}^{m} \sum_{j=1}^{n} M_{ij} d_{ij} \tag{2.23}$$

2.4 Machine learning

2.4.1 Decision Tree

A decision tree is a widely used machine learning model with an underlying tree-based structure, used for both classification and regression. In graph theory, we say that a tree is a collection of nodes connected by edges. The decision tree is a so-called directed rooted tree, meaning that all paths are one-directional and originate from the root.

An example of a decision tree is showed in Figure 2.3. Each node represents a question and its edges are the path of the answers. Making a prediction is thus as simple as a traversal through the tree, passing each split point to

arrive at the answer. After one question has been answered we arrive at the next node. If there are no more nodes we will arrive at the leaf node which corresponds to the final classifying decision of the tree. The Gini index in Equation 2.24 is a measure of misclassification. The optimal question order is decided by minimizing the Gini index such that the lowest possible number of misclassifications occur.

$$I_G = 1 - \sum_{i=1}^{c} p_i^2 \tag{2.24}$$

A useful property of decision trees is that the rules are easily interpreted since the question and decisions at every point are explicit. In other words, the decision rules can be easily interpreted as opposed to many other models that lack such convenience. This makes it possible to for example obtain the importance of each feature.

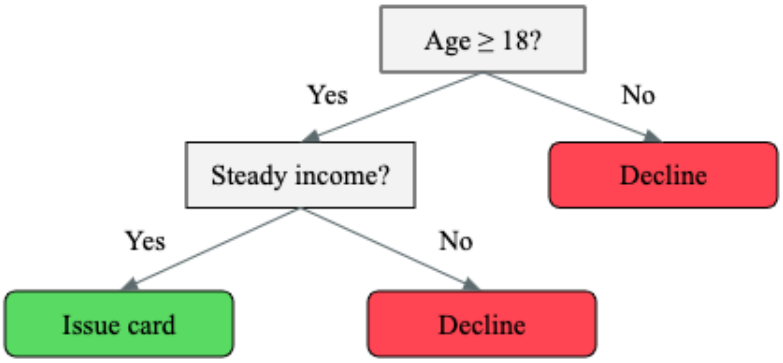

Figure 2.3: Example of a decision tree, illustrating a protocol for issuing a credit card to an applicant.

There are various ways in which a decision tree can be built. Some of the more well-known algorithms for doing so include ID3, C4.5, and CART. CART was introduced by Breiman et al. [19] in 1984 and is short for Classification and Regression Trees. Since scikit-learn (a machine learning library used in this thesis) implements an optimized version of CART, the focus will be on that.

The pseudo code for constructing a decision tree using CART:

1. Select the root node as the feature split with regard to Gini index.

2. Recursively assign rules to each child node and split node in two.

3. Stop splitting according to some predefined stopping criterion or when there is no more improvement of the Gini index.

4. Optimally prune the tree.

Boosting implies that a collection of learners are trained sequentially, with each one being dependent on preceding ones. A weighting mechanism compensates for earlier misclassifications such that the next tree ideally will perform better.

Bootstrap aggregation (or bagging) is a technique where a random subset sampling is applied when training several decision trees. Averaging the results is done in order to decrease variance.

Pruning is a means to counter overfitting by removing leaf nodes as long as there is no loss in information gain. This way, the model will generalize better to unseen data [20].

2.4.2 Random Forest

Random Forests are ensemble models built from a multitude of decision trees. The concept was introduced by Ho [20] in 1995 after which many additional contributions has been made by various authors. Ho proposed that each tree is to be built from a random subset of the feature space in order to form non-identical trees. The perhaps more commonly used implementation today was introduced by Breiman [21] in 2001, in which features are randomly selected for each node split. The idea behind a random sampling is that an ensemble of many diverse learners will generalize differently, complement each other and as a result perform better than its weakest component. The final decision of a random forest is given by the majority vote of all its decision trees.

2.4.3 Support Vector Machine

Support Vector Machines are another set of machine learning models used for classification and regression. The SVM algorithm was introduced by Vapnik et al. in the 1960s [22][23]. A later contribution by Boser et al. [24] called the "kernel trick" provides non-linear classification through input mapping to high dimensional space.

The idea behind SVM's is to create a linear separation of the input data. The line has a margin that is an adjustable hyperparameter. The algorithm aims to maximize the margin distance between points and the decision boundary

itself, ideally resulting in a perfect separation. The datapoints that lie closest to the boundary are called support vectors, since they influence the position of the boundary. Taking the two dimensional case as an example, the algorithm makes a first attempt at separating the data by a line through the data. The line is then slightly shifted for the better after calculating the distance of all datapoints to the boundary, minimizing the distance.

Given a set of data S, we say that a datapoint belongs to class 1 if the linear function is greater than the maximum margin, likewise, it belongs to the opposite class if less than the negative margin. The hyperplane separating the classes are given by Equation 2.26. The hinge loss given in Equation 2.27 is minimized to achieve the separation of classes.

$$S = \{(x_i, y_i)\}_{i=1}^{m} \tag{2.25}$$

$$\vec{w} \cdot \vec{x} - b = 0 \tag{2.26}$$

$$l(y) = \max(0, 1 - t \cdot y) \tag{2.27}$$

2.5 Related work

Having introduced concepts and technical frameworks relevant to the thesis, this chapter explores some previous work that has been done within the context of churn prediction and survival analysis.

Data

The typical data used in this type of application comes from periodic customer records, which are often structured as opposed to unstructured textual data. Structured data consisting of records over a time period is known as time series data. Research has shown that incorporating unstructured textual data can improve the predictive accuracy of convolutional neural networks [25]. Closely related, data mining can be used to extract data of for example customer dissatisfaction through an organization's databases [26].

While it is also important to take into account the time aspect of customer data, the question arises as to how long of a period is enough. Ballings and Van den Poel [27] presents a comparative study showing that almost 70% of the original data (spanning 16 years) could be discarded with minimal loss of predictive performance.

Since churning customers usually make up a relatively small part of the data it may be a good idea to take measures to counter eventual imbalances between classes. Some having been used to solve these issues are for example Synthetic Minority Over-sampling TEchnique (SMOTE) [28], undersampling, oversampling or similar which can improve prediction accuracy [29][30]. While SMOTE has been proved to result in better performance, its application may not be suitable for time series data as the synthetic data is generated through the distance of datapoints between classes.

Classification and Regression Tree (CART) can be used for selecting the most important features [29], but there are other ways such as ones based on stochastic vector machines (SVMs) [31]. Well-selected features can help build a good model but they don't convey what an SVN or CNN has learned, which is why there is value in conducting rule extraction of such models in order to make them more comprehensible [31].

Churn prediction

Statistical methods, logistic regression, decision trees, stochastic vector machines, and variations of neural networks are commonly used in churn prediction [4][25][32]. Mena et al. [4] showed that the usage of recurrent neural network variants such as LSTM performed very well compared to other methods. The same authors also show that using the results from the LSTM as features in logistic regression can yield positive results. However, the authors appear to aggregate features for the logistic regression in a way that time-varying information is lost which could be a factor as to why the inclusion of it (from the LSTM) results in large performance gains for logistic regression.

Chen et al. [33] used deep ensemble models, stacking multiple predictions from neural networks to measure the impact of social media variables on bank customer attrition. The study was done on data from a retail financial institution in Canada and showed competitive performance demonstrating the value of integrating external data. Closely related, Kaya et al. [34] studied the impact of behavioral attributes (features) in financial churn prediction using random forests with SVM-SMOTE oversampling showing that the inclusion of behavioral attributes in combination with demographic attributes yield better performance than using the attributes on their own.

Jain et al. [35] compares the performance of logistic regression, random forest, SVM, and XGboost on churn prediction within the banking, telecom, and IT sector. Their result shows that random forest had the best performance for the banking sector dataset and that XGBoost displayed similar performance while logistic regression and SVM performed noticeably worse. Dalmia et al.

[36] did a similar study of churn prediction in the banking sector comparing SVN, XGBoost, and k-nearest neighbors where XGBoost had the best performance and SVM the least. Pandey and Shukla [37] made an extensive study of churn prediction in the banking sector, applying Bayesian hyperparameter tuning on nine different models. Their results indicate that XGBoost has the best performance, but random forest and SVM performed just as well. The research also confirms that hyperparameter tuning leads to noticeable performance gains for most models.

Zhang et al. [38] propose a combined deep and shallow model approach called DSM utilizing both logistic regression and neural networks. Their experiments on churn data from a Chinese insurance company showed that the DSM performed better than standalone models such as convolutional neural networks (CNN), LSTM, random forest, and many more.

In addition to machine learning and deep learning are other novel approaches based on for example set theory and flow network graphs that also show great promise [39].

These examples are but some of the research that has been conducted in the area of customer churn, but relevant applications of churn prediction can be found in many other sectors such as online gaming [40][41], television [42], and telecom services [43][44] to mention a few.

Churn time prediction

Most of the search results that appear on survival analysis and time-to-event prediction are related to medical studies.

In a survey, Wang et al. [3] have examined the current state of survival analysis methods based on statistics and machine learning and discuss their advantages and disadvantages while presenting some successful applications of time-to-event prediction in various domains.

They describe random survival forests (RSF) tailored for analyzing right-censored survival data. This extension of RF has become well known for its state-of-the-art performance, with implementations available in both Python and R.

Katzman et al. [45] propose a Cox proportional hazards deep learning-based method named DeepSurv. In their paper, they compare DeepSurv, Cox regression, and RSF on several survival analysis datasets evaluated on the concordance index (C-index). The C-index is the most common metric in survival analysis and is a measure of how well the ordering of event times was predicted. Their experiments show that DeepSurv performs as well or better than other state-of-the-art survival models. Another proposed hybrid model of

neural networks and statistical methods utilizing Cox proportional hazards by Kvamme et al. [46] has shown promising results for time-to-event prediction. Both of these proposed models are available as free software packages.

Leger et al. [47] did a study where eleven statistical methods and machine learning methods were trained on pre-treatment image data to predict loco-regional tumor control and overall survival of patients, evaluated on the C-index as a metric. According to their study, Cox regression performed well together with tree-based methods (including RSF), full parametric models based on the Weibull distribution, and gradient boosted models.

Wang and Li [48] proposed the adaption of extreme learning machines (ELM) to survival analysis. The method called ELMCoxBAR is based on an ELM Cox model with an L0-based broken adaptive ridge (BAR) and was shown to outperform both Cox regression and random survival forests when evaluated on the integrated Brier score (IBS) and C-index.

Yousefi et al. [49] proposed SurvivalNet and compared it with Cox elastic net (CEN) and RSF on different datasets of cancer genomic profiles. According to the results, SurvivalNet has the better performance in most cases while RSF appeared to have the worst performance overall. The authors have provided an open-source implementation of SurvivalNet that features automatic training and evaluation among other features. In a similar study by Bice et al. [50], DeepSurv is compared to RSF and Cox regression. The results were evaluated on the C-index and indicate that both DeepSurv and RSF can perform considerably better than Cox regression. They also measure the distribution of errors in months, a result suggesting that all of the models overestimate the remaining time-to-event with a mean value of about 25 months.

The problem of survival analysis in its most straight forward form can be seen as the problem of customer churn prediction within multiple time frames, which can be achieved with standard classifiers as explained by Gür Ali and Arıtürk [51]. Their approach was shown to outperform statistical survival analysis methods.

Martinsson [52] proposes a method based on recurrent neural networks, where predicting the parameters of a Weibull distribution can be used for churn prediction. In theory, the model will train to push the probability distribution towards the correct value. The probability for surviving past time t can be derived for each customer. A useful property of this design is that all data close to the censoring point can be used.

Chapter 3

Method

This chapter describes the method for preparing the data and models, and how the experiments are to be validated. The models to be implemented are LSTM, GRU, RF, and SVM. The models will predict if a customer will churn within the coming six months, and estimate the time until the churn occurs. The models are designed to predict the coming six months using any month in the year as the starting point. The experiments in question are described with the results in Section 4, and measure the predictive performance of the models, but also examine the importance of different features.

It is logical to begin Section 3.1 with a description of the data and some figures to go along with it. The first thing described is the characteristics of the original data, which consists of monthly entries for each customer. This includes a description of the type of features in the data. The data is then processed in order to make sure that it is informative. Figures are then used to describe some statistics about the data. After that, a mathematical definition of the data representation is introduced. It describes the data of a customer in terms of matrices, accompanied by the visualization of data from a few customers. Section 3.1.1 describes how the data is further processed with regard to censoring (which is a characteristic of data in survival analysis problems) in order to create datapoints that can be used with the models. Datapoints are extracted by a method that is sometimes referred to as a sliding window, that creates smaller overlapping copies from the same data. Section 3.1.3 visualize the datapoints in order to get an understanding of the separation of classes.

Section 3.1.2 describe sampling methods for handling imbalanced datasets. Section 3.2 describes the model implementation and training of the models. Section 3.3 describes how the models were optimized through hyperparameter tuning. Section 3.4 describe the metrics used for evaluating the performance

of the models. Finally, 3.5 describes the statistical tests that are used to verify the significance of the results.

3.1 Data

ICA Banken has an extensive amount of data for all of their 800 000 customers since 2001. For the experiments of this thesis, data is only extracted for 52654 and 43368 randomly selected customers over non-overlapping periods of three years (36 months) respectively. Data from the first observation period are used solely for the training data, while data from the second period are split into test and validation data. The data contains monthly records of each customer with 130 features selected for potential use. Research has shown that keeping only the most relevant features can increase performance, while keeping excess features may have a negative impact. Some features were mere descriptions of their numerical representation and others had no variance. After removing those and adding a custom feature, the total number could be narrowed down to 107. The features can be divided into different categories as described by Table 3.1. Train, test, and validation data splits are taken from different non-overlapping time periods so as to make sure training generalizes well to unseen years. The periods used are January 2014 to December 2016 and January 2017 to December 2019. A more detailed description of the features can be seen in the Appendix.

Category	**#features**	**Example features**
Customer characteristics	6	Age, postal code, etc.
Customer segments	14	Customer engagement, lifestyle, etc.
Product groups	14	Insurance, banking, savings, etc.
Products	61	Account and card types, services, etc.
Monetary	12	Credit card spendings, etc.

Table 3.1: A compact description of feature categories.

Feature engineering
Only a few features were manually created in addition to the existing ones. For example, the five-digit postal code was shortened to the first two digits which represent regions within Sweden. But before that, a feature indicating if a customer had moved based on changes in the postal code was created.

Although, moves with no change in the postal code can not be extracted from this data.

A countdown that describes the time until a churn occurs was added together with information if the customer churned during a specific month. This extra information is used to define the target variables, that define the correct answer for the data used by the models.

Feature selection

A quick check for variance in the data allowed for the removal of a few features with zero variance. Unfortunately, this may have included some less commonly bought sub-products from the company offerings. However, if included, they may cause unexpected behavior on the occurrence of it in real-world data.

A correlation matrix was built over the original dataset in order to find correlations to the target variable (churn) or correlation between variables to remove unnecessary features. According to the numbers, there was no prominent correlation between the target variable and any other variables. Perhaps this method is not suitable for finding correlations in time series data. The only noticeable correlations were product categories that had a perfect correlation to their respective sub-products. No features could be excluded based on the correlation matrix.

Observation entry

Most customers in the dataset were first observed on the very first month of the observation period, as seen in Figure 3.1. That would either mean that they had been a customer since before the observation or became one at that point.

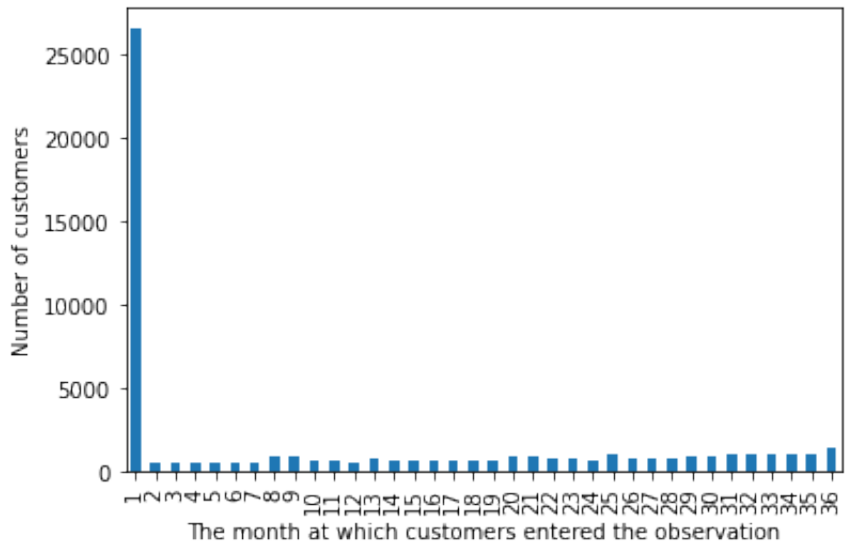

Figure 3.1: A figure showing the month at which customers entered the observation.

Active customer duration

It is difficult to say anything definite about the average duration that churners are active customers without looking at the time since they became customers (before observation started). In the observation, however, it can be seen in Figure 3.2 that a large number of customers churn quite early on (as early as after 1 month). This is problematic in the sense that information about those customers is minimal. For the problem to be data-driven, it was decided that the focus will be on those who have been customers for at least twelve months.

Naturally, non-churning customers as seen in Figure 3.3 will have a longer observed time dependent on the point of entry. Since most customers entered the observation in the first month it is only natural that non-churners have a similarly skewed distribution with the maximum activity length.

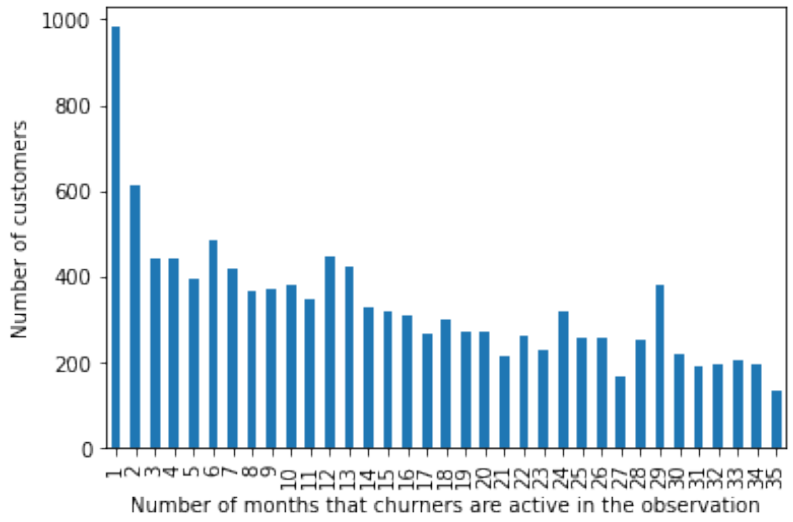

Figure 3.2: The duration that churners are active customers in the observation.

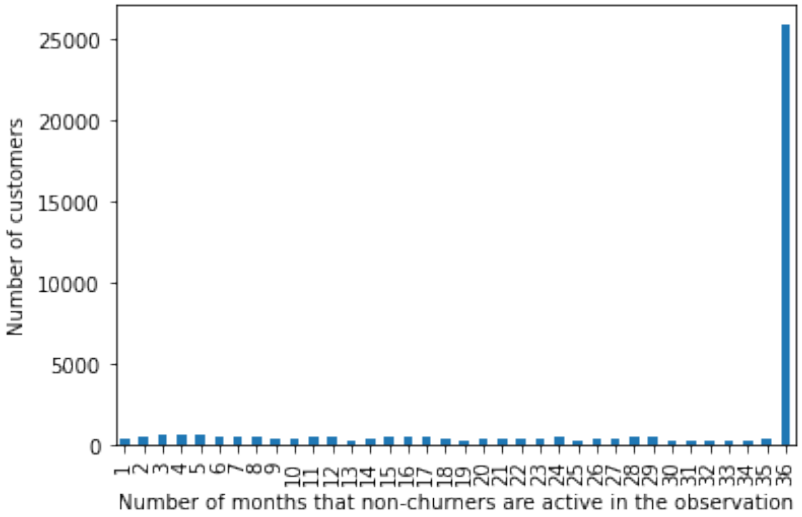

Figure 3.3: The duration that non-churners are active customers in the observation.

Periodical trends

Looking at Figure 3.4, there is a slightly visible pattern indicating that churn is more present during the summer and at New Year's. Some of the occasional spikes may have been caused by changes in terms of pricing, or perhaps even external market factors such as upcoming competitors and the state of the global economy.

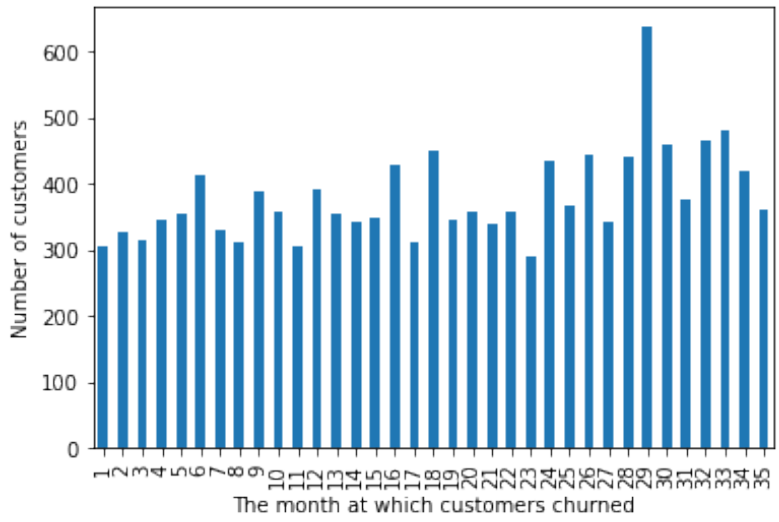

Figure 3.4: The number of customers that churned at a given month during observation.

Recurrent events

Figure 3.5 describes the occurrence of reoccurring churns, which are customers that end their all of their contracts but then become a customer again. It was decided that the scope of the thesis should be limited to only focus on non returning customers due to the low number of reoccurring churns available. As such, any customer with two or more churn occurrences is removed from the dataset.

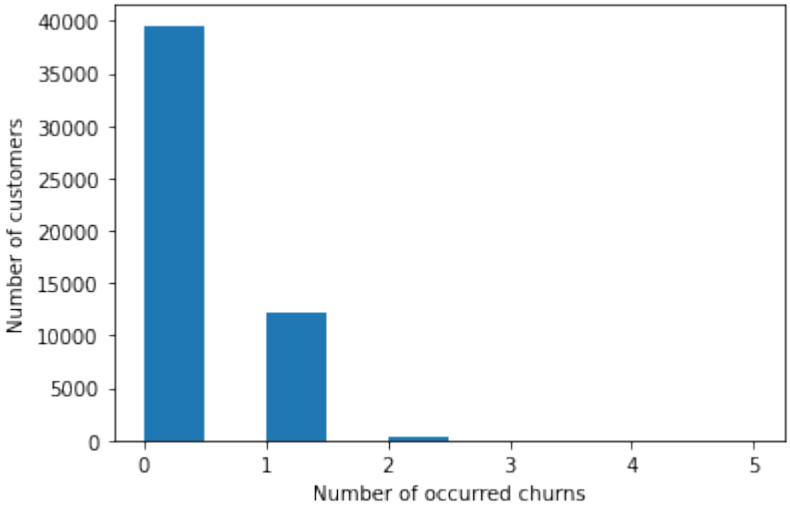

Figure 3.5: Histogram of occurred churns per customer.

Customer growth

The growth of the customer base is limited by the acquisition rate and churn rate. If the real active customer count during the observed period is compared to an imaginary churn free period as in Figure 3.6, then it can be seen that preventing churn could have a significant impact on growth.

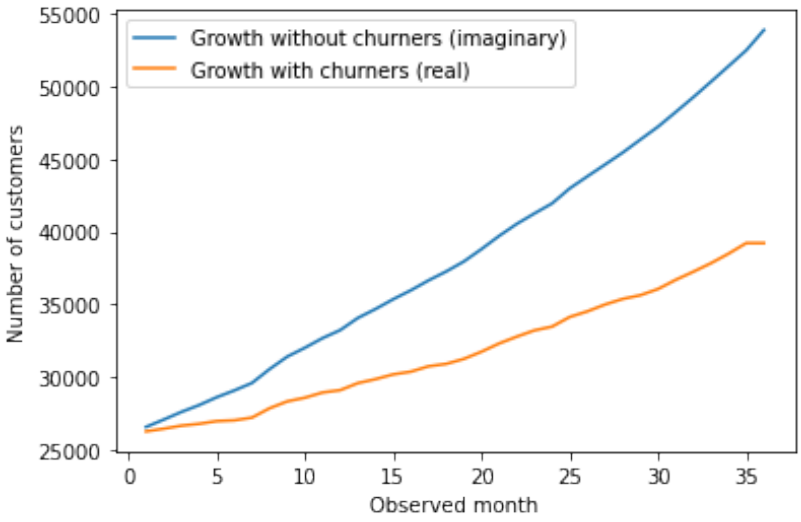

Figure 3.6: The real customer growth compared to the imaginary case where no customers churn.

Reduction of data

To the left in Figure 3.7 is the distribution of non-churners compared to churners before applying the conditions outlined in the delimitations, while the right side is afterward. Discarding data that don't satisfy the requirements has relatively little impact on non-churning customers but approximately halves the useful data of churning customers.

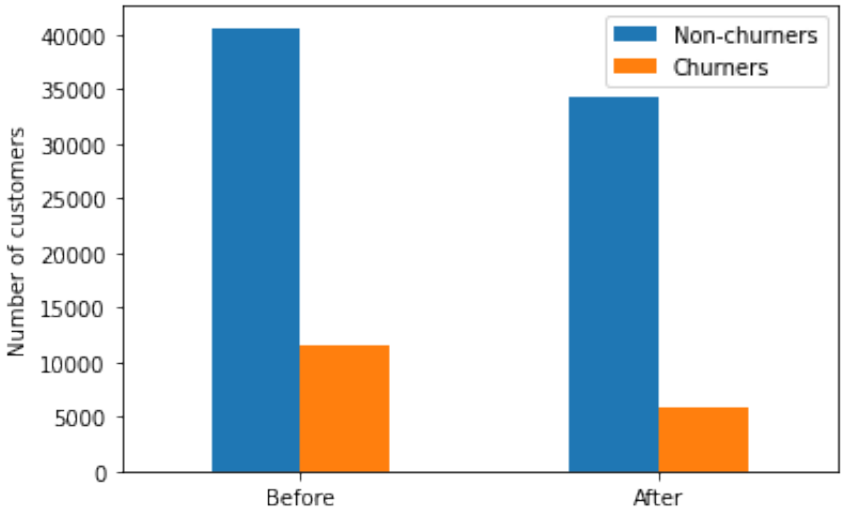

Figure 3.7: Before and after comparison of the number of customers who satisfy the conditions set on the data.

Mathematical definition of the data

The extracted data D for a customer contains a monthly history of length m, where m is at most 36. The data of a customer c is denoted D_c, represented as in Equation 3.1 where every d represents a feature at the given month.

$$D_c = \begin{bmatrix} d_{1,1} & d_{1,2} & \dots \\ \vdots & \ddots & \\ d_{m,1} & & d_{m,107} \end{bmatrix} \tag{3.1}$$

Similarly, a single datapoint of a customer is denoted X_i, where the maximum value for i depends on the length of the customer history. A single datapoint X_i as extracted by a sliding window is given by Equation 3.2 (the process of the sliding window will be described in the next section).

$$X_i = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots \\ \vdots & \ddots & \\ x_{12,1} & & x_{12,107} \end{bmatrix} \tag{3.2}$$

Data visualization

The data has been normalized for the visualizations in Figure 3.8. The figure represents the data from different customers during the whole observation period of 36 months. The observation period runs along the vertical axis, with

the first month being on the top of each heatmap, and the last month on the bottom.

The figure may seem overwhelming and difficult to interpret considering the number of features included. Similar features are not necessarily next to each other, but mostly follow the structure of the feature categories described in Table 3.1. However, a rough estimation should suffice for the purpose. Starting from the left, customer characteristics (0-5), customer segments (6-19), product groups (20-33), products (34-94), and monetary (95-106). In reality, the last two features are "observed time" and "moved" from the customer characteristic category. The reader is once again referred to the Appendix for details about the features in each category.

Some prominent vertical lines are appearing. For example, the thicker line at around feature 85 represents a small money flow, which is formed because spendings are represented as negative values. Consequently, small transactions get represented as a high value.

The first recorded entry in the observation can be seen at the top according to the month-timeline. Churning customers will have the last recorded entry before the end of the observation while non-churners may fill the observation completely.

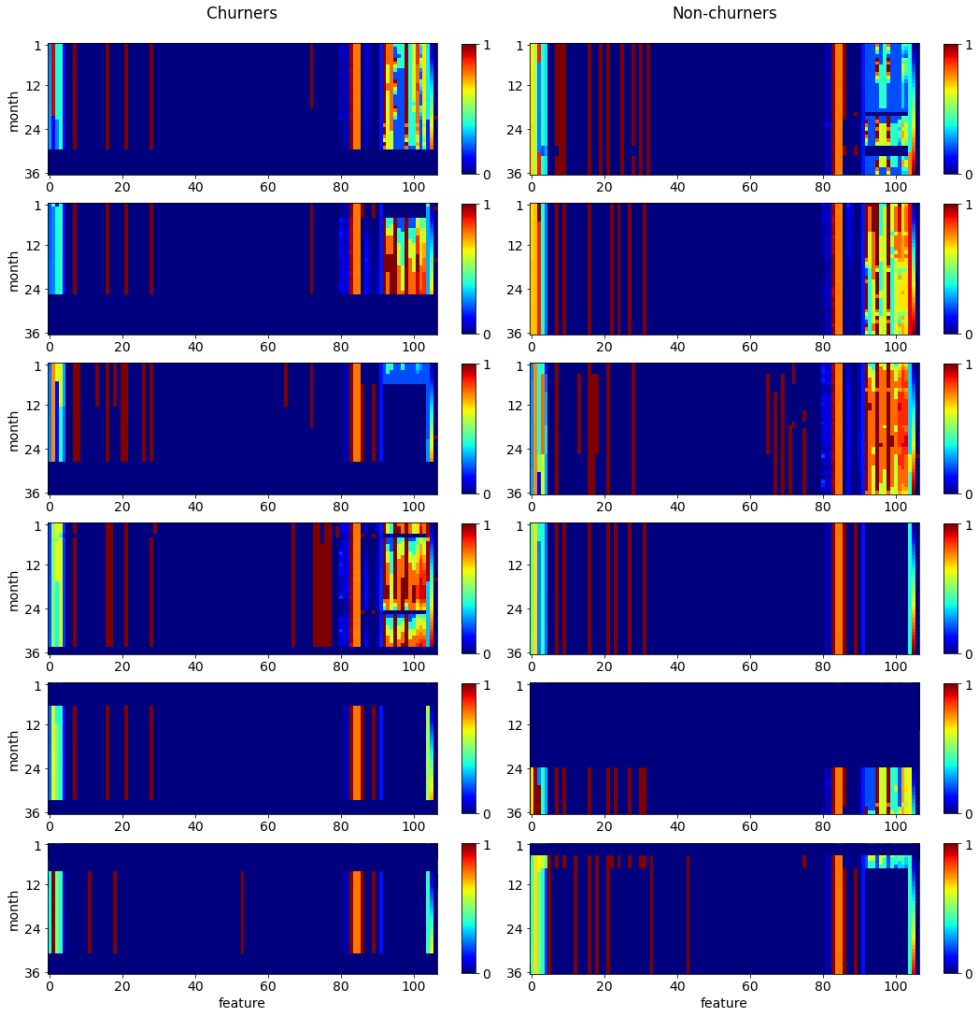

Figure 3.8: Visual representation of customer data on a chronological timescale where month 1 represents the first month in the observed period. The data has been been normalized. A rough estimation of the features is as follows; Starting from the left, customer characteristics (0-5), customer segments (6-19), product groups (20-33), products (34-94), and monetary (95-106). In reality, the last two features are "observed time" and "moved" from the customer characteristic category.

Looking closer at how the features vary through time for some customers, one may realize that there are a lot of seemingly static features and that distinguishing between churners and non-churners is not such a trivial task. The visualizations appear sparse because of a vast product repertoire and the fact that no customer will have all of these products. There is minimal interac-

tion from the customer other than card transactions since other features that customers can affect are products that are either active or not. In most cases, the only constantly varying data is the survival time and age in addition to the monetary-related features. RFM and customer segments are closely tied to transactions as well. Unfortunately, a large portion of entries in the dataset has no transaction data at all since not everyone uses their cards. Simply removing customers with no transaction data from the dataset is not a good idea since the absence of it reflects a common customer behavior.

3.1.1 Datapoint creation

The data is not suitable for direct input into the models and has to undergo some pre-processing to get the right formatting. This can be done in various ways, but one of the perhaps most logical ways is to see the data as right-aligned.

Churn definition
Customers are marked as churned when they no longer have a subsequent entry (month) recorded. The month that is marked as the "churn month" cannot be used as input data as it may contain direct information about the churn in the data. For example, imagine a customer that terminates their membership on the 10th of some month. Then, the monthly average expenses might get unusually low. In the real world, we don't have access to this data until after the end of the month though, which is one of the reasons why it shouldn't be included in training and that all predictions are given for a month ahead. Another reason is that there is no usefulness in detecting a churn after it has occurred.

Visualization of time
For illustrative purposes, it is convenient to have a visualization of the data. Initially, the data is arranged on a chronological timeline where each timestep represents a month of the year, as in Figure 3.9.

A customer may have been with ICA Banken since before the start of the observation period or having joined any month within the observation period. Some customers may have churned at the end of the observation period, or long after it, in which case such a datapoint is considered right-censored. The censoring will mostly affect these churning customers, marking them as censored. To counter the effect of such mislabeling, the most recent months available for non-churners have to be discarded.

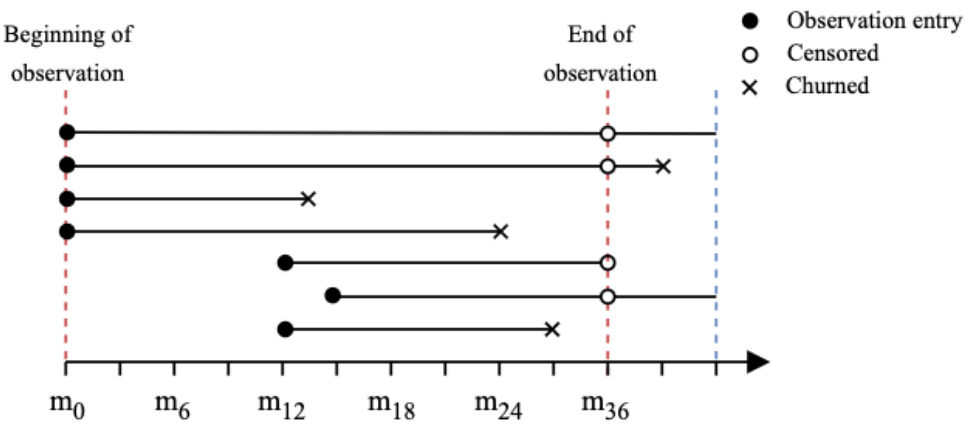

Figure 3.9: A timeline of active customers during the observation.

The survival time of a customer is defined by the elapsed time since the first appearance in the observation period, an aspect that is similar to traditional survival analysis. The time to churn is defined as the countdown towards the month the churn occurred. Non-churning customers are measured a bit differently as there is no real counter to any event. Instead, months close to the censoring point are discarded so that it can be said that *at least* m *months are left until churn*.

In a convenient visualization, the history of every customer is aligned towards the rightmost side as seen in Figure 3.10. There is no need to explicitly align non-churning customers since they are already right-aligned and censored by nature.

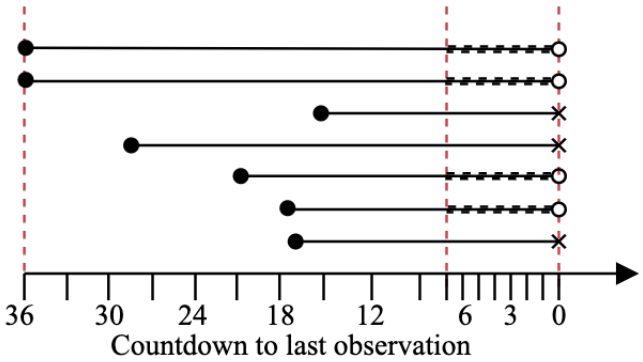

Figure 3.10: A right-aligned timeline of active customers. The seven most recent months of non-churners are unusable and discarded.

Sliding window

The whole sequence of a customer can not be directly put into the model. Instead, a short period of twelve months that constitutes a datapoint is used. Datapoints are created by capturing all of the data of a customer in multiple overlapping copies through a method often referred to as a sliding window. The window as defined by Equation 3.2 is slid over time to create datapoints. The use of twelve months is a compromise of excluding non-informative data while still having enough to infer any useful pattern.

Datapoints that captures more information is preferred but comes at the cost of not being able to fill the window completely. For example, a customer with 15 months of history would not fill a window length of 30 months. Furthermore, can a history of four months be reliably used for predicting six months ahead? These examples are some problems that need further research. For simplicity, a delimitation was put into place specifying that anything less than twelve months of history is not informative enough and therefore excluded from the data. At the same time, it was also decided that a window of twelve months is a reasonable starting point.

The delimitations prevent having non-informative datapoints of for instance one month after which customers churn. If these were to be included then the beginning months of non-churners would most likely be needed to let the model learn to differentiate them while taking into account that there can be single month churners where the beginning was censored in the observation dataset, something that will not happen on real-world data since only the latest months will be used. From the observation period, it is not possible to see the starting pattern of most non-churners.

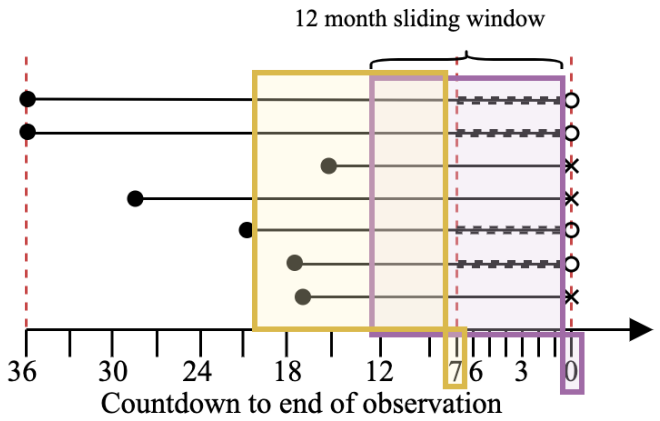

Figure 3.11: Illustration of two sliding windows over the dataset with respect to the target variable (time-to-churn).

As illustrated in Figure 3.11, a sliding window of twelve months is used to create the input data. The window is slid from one side to the other while making sure it is filled with data. The corresponding label for each such datapoint is the number of months that remain until churn. The customers who churned will get a well-defined target variable y equal to the time to churn (ranging from 1 to 6, as this is the most interesting range to ICA Banken). Anything over that will be labeled 7.

There is an exception to the last statement though. The censored customers can not possibly have a counter to the churning event. It can however be said that it is at least 7 months or more left if the starting position of the window is moved 7 months back. This solution comes with the caveat that the most recent 7 months cannot be used for training since it may be less than 7 months left if we look beyond the observation period. The amount of data for non-churners is many times larger than for the churning customers, making the loss of a few observed months less of a problem.

3.1.2 Sampling

The dataset is highly imbalanced, with a 5:1 ratio of non-churners to churners. Training a deep learning model with a highly imbalanced dataset will often result in the majority class being predicted since the optimization of the loss function will simply favor the majority class. To counter this, one can try to weight the importance of the classes to the inverse proportional ratio. This did however not prove useful in optimizing the loss and overall model performance. Other alternatives include the use of sampling techniques to even out class distribution. Figure 3.12 show two different sampling techniques.

Oversampling
Oversampling is a technique where the minority class is copied times over to achieve a 1:1 ratio between classes. In this case, it resulted in the models quickly overfitting to the duplicated data.

Undersampling
Undersampling selects a random subset of the majority class equal to the number of entries in the minority class. It results in a big loss of samples since the non-churning samples are over-represented. Still, this proved to be the most efficient solution.

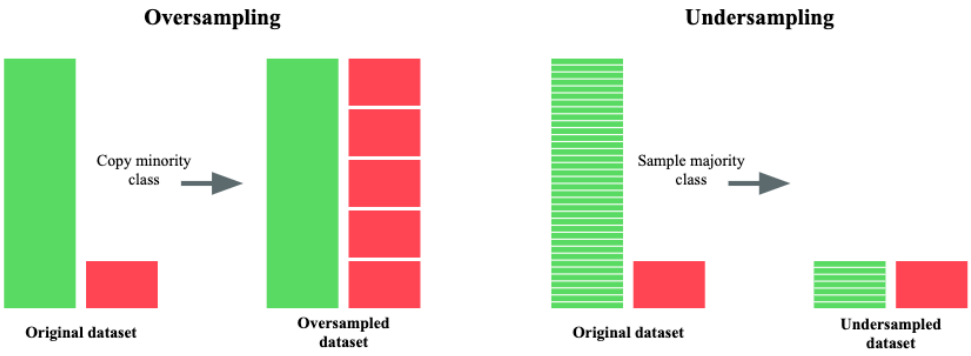

Figure 3.12: An illustration of oversampling and undersampling.

3.1.3 Analysis of data

PCA and t-SNE were applied and used to plot the most important components in an attempt to visually separate the different classes into clusters. In this case, PCA will just be seen as a visual aid to see if class separation is possible. The PCA components could potentially be used for k-means clustering or as additional features to the data.

The first step is to calculate the principal components. The data had to be flattened to a vector for this to be possible. The temporal features are still there but at the same time they are now seen as separate features instead of timesteps of a feature. The principal components were plotted in a scree plot to determine their significance and taking note of the Kaiser rule criteria, the horizontal line that forms as the eigenvalues levels off to determine the less significant components. The decision can be somewhat subjective, but in this case, the third component as seen in Figure 3.13 seems about right, indicating that any component including and after the third would be insignificant.

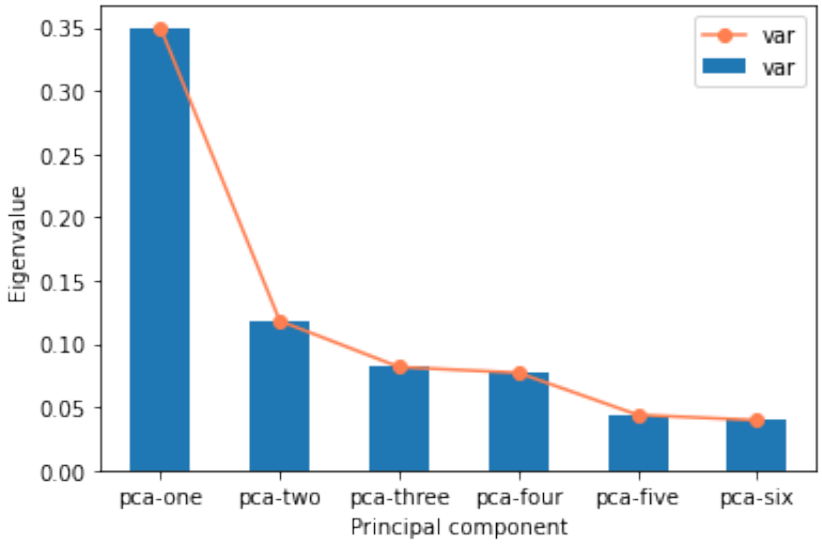

Figure 3.13: Scree plot that explain the variance of the principal components.

In Figure 3.14 through 3.15, the most significant PCA components have been plotted in 2d. Located to the left in the figures is the datapoints from the churn time prediction problem, while the datapoints from the churn prediction problem is on the right side. The figures reveals some distinct higher concentrations of the 7+ months class. Plotting the PCA of the churn prediction data displays a more clear separation. Unfortunately, datapoints from all classes intermingle a lot and can not be seen as a particularly good result compared to the visual separation in standard problems such as CIFAR-10 and MNIST.

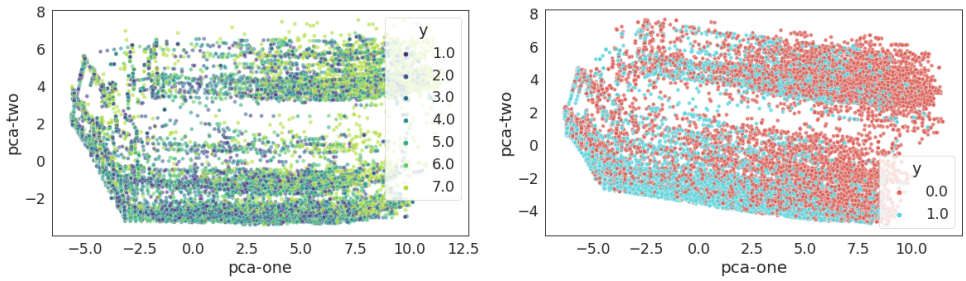

Figure 3.14: A 2d plot with respect to the two most significant PCA components. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes.

A 3d representation of the three most significant components in Figure 3.15

mostly revealed what had already been seen in the 2d graph. On the left side, the brighter dots represent the 7+ months class and on the right side, the blue dots represent the corresponding class. While the 7+ class are mostly separated towards one corner in the space, the task of distinguishing churn times would appear almost impossible and the fact that 7+ intermingle with churners over the whole space could be a bad sign. On the other hand, distinguishing churn in a binary sense could work since many churners are concentrated in the lower-left corner in the space.

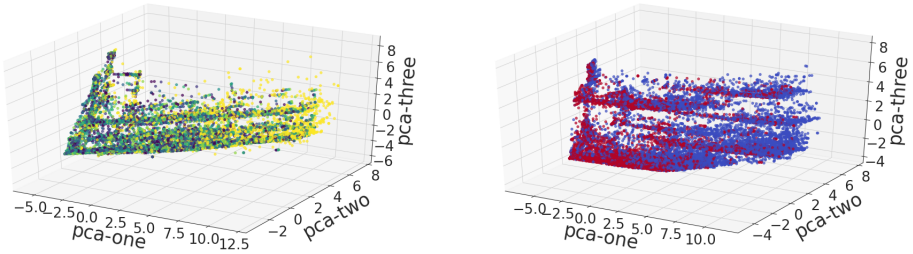

Figure 3.15: A 3d plot with respect to the three most significant PCA components. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes.

In a final attempt, t-SNE is applied to the principal components in order to further clarify the separation of classes in the two problems of churn prediction and churn time prediction. The t-SNE algorithm is very slow and the number of datapoints had to be decreased to 4267, which is about 14% of the datapoints after undersampling. This should not pose any problems since CIFAR-10 suffers from the same constraints and still yields a better separation than PCA alone. The result of t-SNE can be seen in Figure 3.16, with the datapoints of the churn time prediction problem on the left, and datapoints from the churn prediction problem on the right.

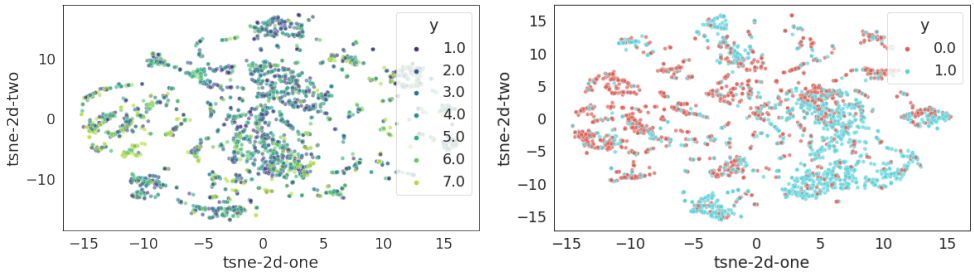

Figure 3.16: A 2d plot after applying t-SNE. Located on the left are datapoints from the churn time prediction problem that contain seven classes, and on the right side are datapoints from the churn prediction problem that contain two classes.

Unfortunately, the t-SNE graph looks a bit sparse and spread out with no apparent clusters emerging with regard to churn times. In contrast, there is a more prominent separation for the binary churn problem as the churning class has some clusters that don't blend with non-churners as much. Although, the seemingly better clustering could be an effect of fewer visible datapoints.

3.2 Implementation

This thesis is about predicting customer churn, and the period of interest is the nearest six months. Furthermore, it is not only interesting to know if a churn will occur, but also estimate how long it is before it happens. These two problems require a formulation of the problem is decided, and in this case, a classification framing is used. This means that any prediction made by the models to be implemented will be based on classification. The classes look a bit different depending on the problem.

The first problem will be referred to as *binary churn prediction*. The predictions will answer the question if a certain customer will churn or not in the coming six months. The natural answer to this question is either yes or no. The equivalent in binary terms is either one or zero. Datapoints are therefore separated into two classes of those who will churn within six months, labeled 1 (referred to as churners), and customers who do not churn within six months, labeled 0 (referred to as non-churners).

The second problem of predicting the time until a churn occurs will be referred to as *churn time prediction*. The predictions will answer the question of how many months remain until a churn occurs, with some caveats. There are

only seven classes. The classes numbered 1 to 6 will contain the datapoints with the corresponding amount of time left until churn. The seventh class, labeled 7+, contains datapoints of customers that have seven or more months left until churn. The seventh class is similar to the non-churner class of the binary churn prediction problem.

The data in both of the problems are essentially the same, with the only difference being in how the datapoints are separated into classes. An even class distribution is achieved through undersampling in both of the problems.

To summarize, the two problem formulations require the development of two separate models defined in the list below.

1. Binary churn prediction: A classification model predicting if a customer will churn within six months.

2. Churn time prediction: A classification model predicting the time until a customer churns.

A solution to these two problem formulations will be implemented with and evaluated on the following four models; GRU, LSTM, RF, and SVM.

GRU and LSTM
The GRU and LSTM models are implemented with Keras, using a TensorFlow backend. The architecture consists of a single GRU or LSTM layer with a sigmoidal activation function which is connected to a final dense layer with seven outputs for the churn time models and a single probability output for the binary models. Adding more layers quickly caused overfitting, indicating that one layer is sufficient to represent the problem. The churn time models use EMD loss while the binary churn models use binary cross-entropy loss.

RF and SVM
Both the RF and SVM implementations come from the scikit-learn library. Notably for the SVM is that there is no standard way of doing a multi-class prediction and instead uses two different heuristics. The two heuristics are; linear OVO (one vs one) and linear OVR (one vs rest). Linear OVO trains several linear classifiers for each unique label, comparing each one to all other ones individually. Linear OVR trains one linear classifier for every label, comparing it to all other labels collectively. For the binary churn predictions, SVM models are trained using different kernels (linear, and radial basis function (RBF)).

Model training and prediction

The models are trained with datapoints that contain a twelve-month history for some customer. Each datapoint has an associated target variable (time-to-churn) that defines which class it belongs to. During model training, the datapoints are shuffled. This is the expected behavior when training an RF or SVM.

When training a recurrent neural network such as GRU or LSTM, there are other possibilities that include using the datapoints of a customer in chronological order, one by one, thus building up a state in the model. A probable usage in another domain may be to have continuous time series data by the minute for every day during a number of years and use this data for predicting a few minutes ahead on an arbitrary day. New data would then arrive every minute, making it possible to make new predictions with the most recent data. However, the data of customers in this thesis are of varying length which makes it a bit different from the described case. Furthermore, there is little use for rolling predictions since the data consist of monthly records and is compiled monthly.

The building of states does however make it possible to use arbitrary lengths of data to make predictions. For example, all of the historical data of a customer since their first contract could be fed into the model, little by little in chronological order. After building up the state until the most recent month, a prediction could be made. However, research on customer churn has shown that many year's worths of data may be discarded with minimal effect on predictive performance [27]. A reason could be that the data from many years ago no longer reflect the recent behavior of customers. The way in which GRU and LSTM utilize their potential for sequential data in this thesis is solely through the state that is built up from each datapoint of twelve months, as opposed to using all datapoints to potentially build a state from years of data.

In this thesis, predictions are made through classification. When a datapoint of twelve months of historical data is fed into the model, a prediction is made about the future. The churn prediction model will predict if a customer will churn within six months from the most recent month in the datapoint, while the churn time prediction model predicts the time until a customer churns. Predictions can be made at any month of the year since the design is only dependent on the length of the observation period, and the model has no explicit knowledge of any specific month.

3.3 Hyperparameter tuning

Deep learning and machine learning models have a multitude of variable settings that affect the training of a model, called hyperparameters. It is often a good idea to spend some time searching for good parameters as they will vary by model and can have a big effect on accuracy. Two common ways to perform a hyperparameter optimization are grid search and random search.

A grid search trains a model with every combination of manually prespecified parameters and evaluates the performance either with a validation set or through cross-validation. The process is also called a parameter sweep, and such an extensive search can take a lot of time for larger models. In comparison, a random search saves time by randomly generating hyperparameters from a set range.

A random search is used in this thesis due to limited computing time. Evaluation is done on a validation set taken from a separate observation period than the training data. The deep learning models measure the averaged loss and accuracy of multiple runs since the execution is non-deterministic, even with measures taken to prevent such behavior. The machine learning models are evaluated once for every set of hyperparameters since the execution is deterministic.

A cross-validation approach was avoided because it would require every sample from a customer to be in either the train or validation set. The reason is that samples generated from a customer are temporally dependent and can belong to different classes that when mixed into both training and validation sets is cause for data leakage that leads to overfitting. A solution could be to cross-validate on the customers rather than the samples.

The following tables contain the hyperparameters and their respective values used in the random search. Some hyperparameters have a set of manually picked values that are randomly selected while others are randomly generated from within a specified range.

Hyperparameter	**Value**
Number of units	2 exp[(range(5, 9)]
Learning rate	range(1e-4, 1e-2)

Table 3.2: LSTM and GRU hyperparameter search.

Hyperparameter	Value
Number of estimators	range(5, 200)
Maximum depth	range(5, 20), range(5,200)*

Table 3.3: RF hyperparameter search. *Churn time prediction parameters.

Hyperparameter	Value
C	range(0.05, 2), range(0.05,0.1)*
Kernel	["linear", "rbf"], ["linear"]*

Table 3.4: SVM hyperparameter search. *Churn time prediction parameters.

3.4 Evaluation metrics

Binary classification
In binary classification, there are different types of correct and incorrect predictions. These are referred to as true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Each of those describes if a datapoint of either class is correctly classified or not. For a graphical representation, see Figure 3.17.

	Actual Positive	Actual Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

Figure 3.17: A table of the predicted outcomes.

Two useful metrics are sensitivity and specificity. The sensitivity tells us the fraction of correctly classified positives among all predicted positives, while the specificity tells us the fraction of correctly classified negatives among all predicted negatives.

$$Sensitivity\ (TPR) = \frac{TP}{TP + FN} \tag{3.3}$$

$$Specificity\ (TNR) = \frac{TN}{TN + FP} \tag{3.4}$$

Receiver Operating Characteristic (ROC)
The receiver operating characteristic curve graphically illustrates the true positive rate (TPR) against the false positive rate (FPR). The true positive rate is also called fall out and is given by the formula below.

$$Fall\ out\ (FPR) = \frac{FP}{FP + TN} \tag{3.5}$$

In practice, all predictions are sorted according to the predicted probability scores as seen in Figure 3.18 and then the decision threshold (normally fixed at 0.5) is varied from 0 to 1. At each threshold, we calculate and plot the TPR against the FPR which yields the ROC curve.

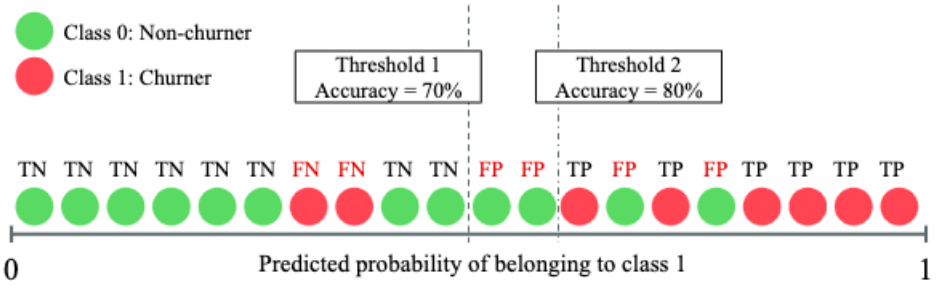

Figure 3.18: Varying the threshold to achieve different classification results.

Area Under the Curve (AUC)
The ROC curve can be used for trying to adjust the classification threshold to optimize for sensitivity or specificity. It is also useful when comparing different models, but it can be tricky to determine the better of two similar curves.

A curve that is over the other curve is generally preferred, but it is possible for the curves to intersect making it difficult to determine which is better. When comparing the ROC curves for different models we instead use a metric known as area under the curve. A perfectly random classifier would produce a straight line from the bottom left to the top right with an AUC of 0.5, while a perfect classifier theoretically achieves an AUC of 1.0. The combined area under the curve for a ROC curve is referred to as ROC-AUC.

PR-AUC
The precision and recall can be plotted alongside each other in what is called a Precision-Recall curve to show the trade-off of those metrics. At the rightmost end of the graph, the classification threshold is 1.0, meaning that all classes are classified as 1 at the cost of low precision. On the opposite end, the threshold is 0.0 and full precision is achieved at the cost of a low recall. The Precision-Recall curve can also be used to calculate an area under the curve, similar to ROC-AUC. In the case of precision and recall, the score is called PR-AUC or average precision.

F1-score
The F1-score used in binary classification measures the harmonic mean based on both precision and recall (same as sensitivity).

$$Precision = \frac{TP}{TP + FP} \tag{3.6}$$

$$F1\ score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.7}$$

Root Mean Squared Error (RMSE)
The root mean squared error (RMSE) is a measure of the difference between predictions and the true values, where larger errors are penalized more. Therefore, a low RMSE indicates better predictive performance. A useful property of RMSE is that it is in the same unit as the input, which makes it interpretable in terms of the target variable. However, the threshold for what to consider a good RMSE score is dependent on what is being predicted.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{3.8}$$

3.5 Significance and effect size

Analysis of Variance Test (ANOVA)
ANOVA is a collection of statistical methods that test for significant differences between the means of two or more independent sample groups. A follow-up test is then conducted in the case of having observed significant differences to determine which sample groups are significantly different. This can be done using for example the posthoc Tukey's HSD (honestly significant difference) test that compares all pairwise combinations of the sample groups.

Cohen's d
Effect size describes the magnitude of effect in an experiment. Typical usage is in medicine where the effect of several developed substances can be compared in regard to their overall potency. The effect size when comparing the outcome of two groups can be measured by for example Cohen's d.

$$s = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}} \tag{3.9}$$

$$d = \frac{\mu_1 - \mu_2}{s} \tag{3.10}$$

The calculated d indicates the number of standard deviations the groups differ by as given by Table 3.5 [53] [54].

Since the effect size can be measured through various metrics, Rice and Harris [55] have provided an equivalence table for transitioning between some of the more popular ones, including for example AUC. Furthermore, another formula called Cohen's f is used for expressing the effect size between two or more groups. Cohen's f is suitable to be used with ANOVA [56]. Depending on the measure of interest, Cohen's d or Cohen's f may be more appropriate.

Effect size	**d**
Very small	0.01
Small	0.2
Medium	0.5
Large	0.8
Very large	1.2
Huge	2.0

Table 3.5: A conversion table of effect size according to Cohen's d.

Chapter 4

Results

The results obtained from the conducted experiments are presented in two main sections, separating binary churn prediction and churn time prediction. The chapter ends with a significance test on the retrained models' performance.

4.1 Model performance

4.1.1 Churn prediction

In this section, the four churn prediction models that were developed are compared with regard to the metrics of accuracy, ROC-AUC, F1-score, precision, and recall. The performance of each model is presented in Table 4.1. Furthermore, the section will describe optimization with regard to precision and finally present a profit simulation of a real-world scenario.

The tuned models varied a lot on some metrics and performed similarly on other. In terms of accuracy, the deep learning models were generally better than RF and SVM, with LSTM performing slightly better than GRU. Comparing ROC-AUC, the deep learning models have a noticeable lead over the other models. Interestingly, SVM has a better ROC-AUC score than RF even though it's accuracy is lower than RF. Both GRU and LSTM displays the best F1-scores of 71%. It should be noted that the SVM had the number of datapoints in the training set limited to about 5% due to unreasonable computation time. Despite this, it performed similarly to RF.

Model	Accuracy	ROC-AUC	F1-score	Precision	Recall
LSTM	0.714	0.782	0.71	0.71	0.74
GRU	0.711	0.780	0.71	0.70	0.74
RF	0.689	0.716	0.69	0.80	0.50
SVM	0.673	0.740	0.67	0.64	0.78

Table 4.1: A table of binary churn prediction results.

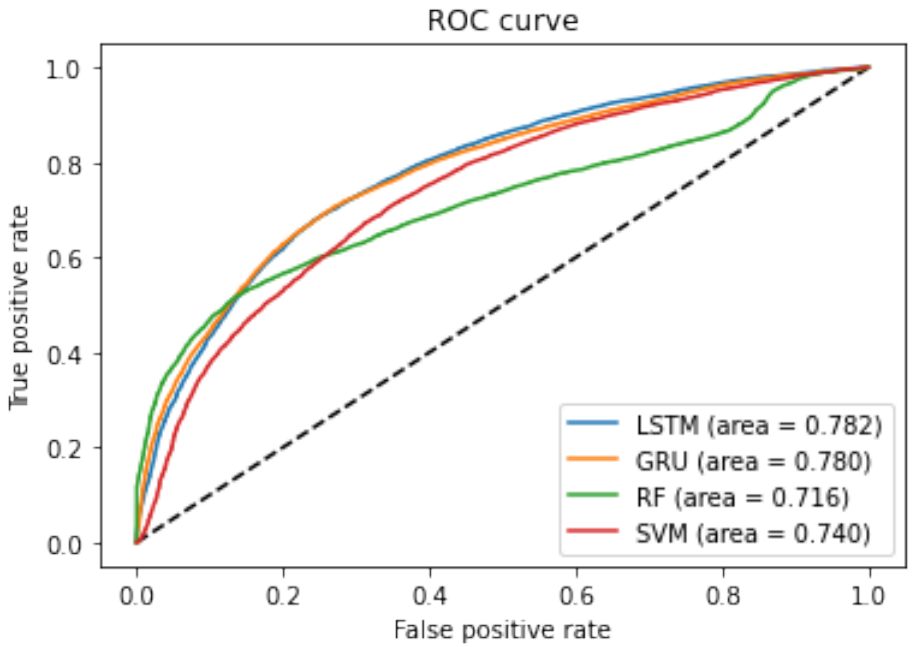

Figure 4.1: A graph comparing the ROC-AUC of different models.

The results can be compared to other studies of churn prediction to get a better sense of the achieved performance. In a similar experiment [4] of predicting contract terminations of a financial services provider with RFM features, the ROC-AUC for different models ranged from 0.741 to 0.779. The best result of this thesis is 0.782 which is within expectations.

The ability to identify churners is still relatively poor when the real-world distribution of churners is taken into account. Therefore, it becomes interesting to study if the classification threshold can be adjusted in order to minimize false positives and maximize true positives. Currently, even the best model will find more false positives than true positives in a real-world scenario which could be very costly if used in production.

Optimizing the threshold

The number of classified true positives is optimized to be as high as possible with regard to false positives. In Figure 4.2, we study the precision and recall in order to find an optimal threshold.

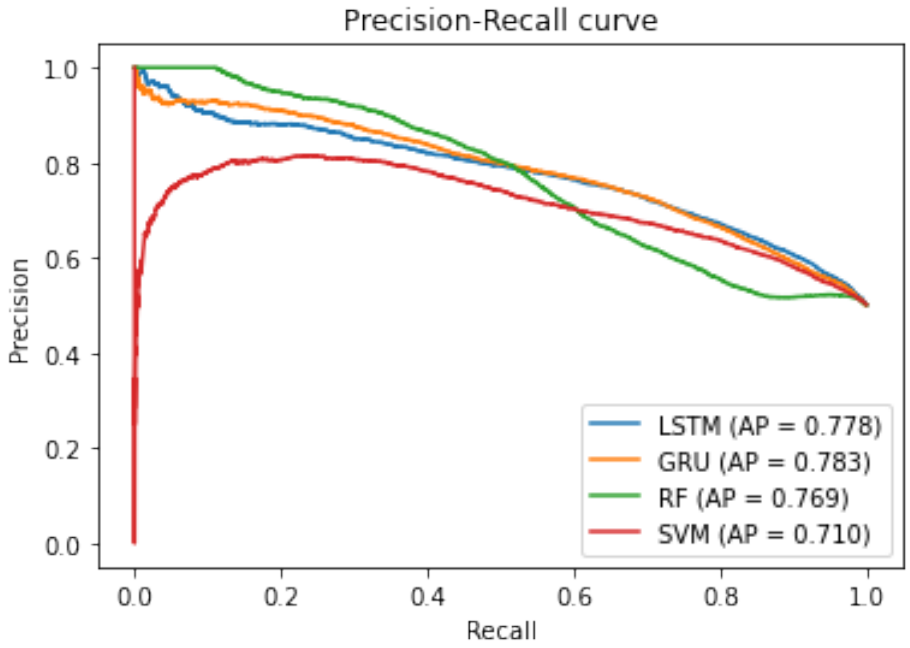

Figure 4.2: A graph comparing the average precision of different models.

The deep learning models display the highest average precision (also called PR-AUC) in the Precision-Recall graph. However, the RF model performs better in the high precision range where it has close to 100% precision with a recall of around 10%. In theory, the model can be almost 100% certain about the predicted churners with the caveat of only finding about 10% of them.

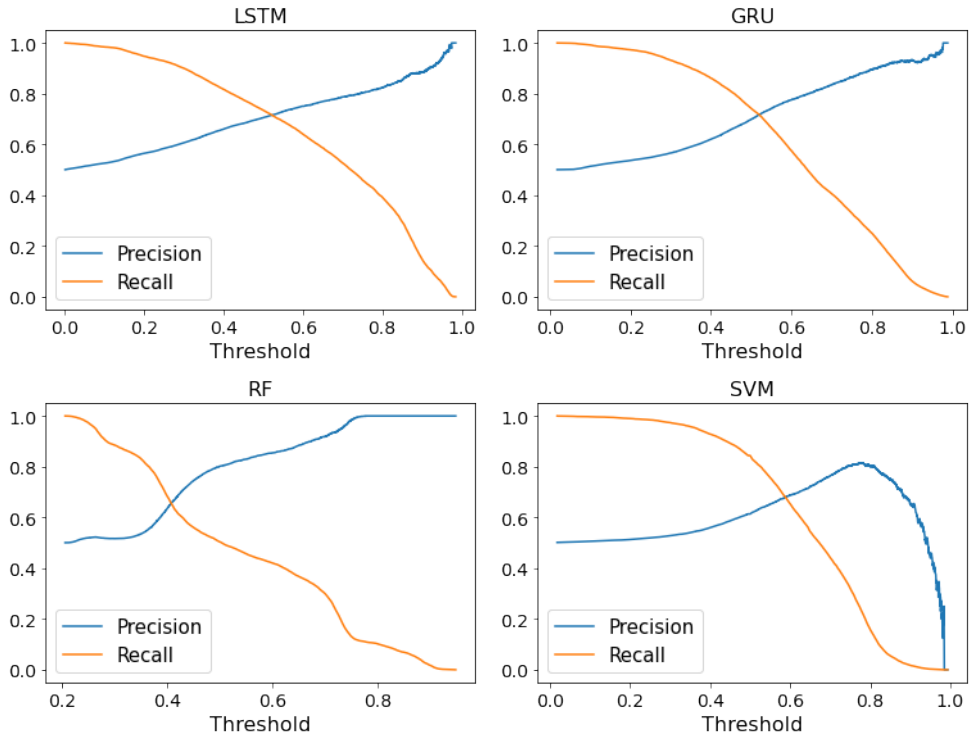

Figure 4.3: A graph of the precision and recall of every model for every threshold.

The best threshold value is subjective and depends on the criteria for what to optimize for, but in this case, the high precision of the RF model where recall is about 10% seems promising. The corresponding optimal threshold of about 0.77 can be derived from Figure 4.3.

Simulation

The model is only useful if it can generate profits. Therefore, a cost model is introduced to measure the costs related to the four different outcomes in binary classification. ICA Banken has not provided any official numbers to use in these calculations due to secrecy. Instead, the numbers used for calculations are imaginary but reasonably assumed.

In the example below, a random pick of 300 churners samples and 10000 non-churner samples are used to simulate a real-world-like scenario with a 3% churn rate over the coming six months. The costs assigned to every predicted outcome can be seen in Table 4.2. It should be noted that the assumption that all TP will take the offer is naive and should only be taken as a simple example to illustrate the point. Simulation results are seen in Figure 4.4 and Figure 4.5.

Type	Cost	Description
TN	0	Loyal customer, stays by default.
FN	1000	Churning customer, will need a replacement customer.
FP	100	Loyal customer, 100 SEK fee reduction.
TP	-900	Churning customer, 100 SEK fee reduction. Saves 900 by not having to acquire a replacement customer.

Table 4.2: Assigned costs to every binary prediction outcome.

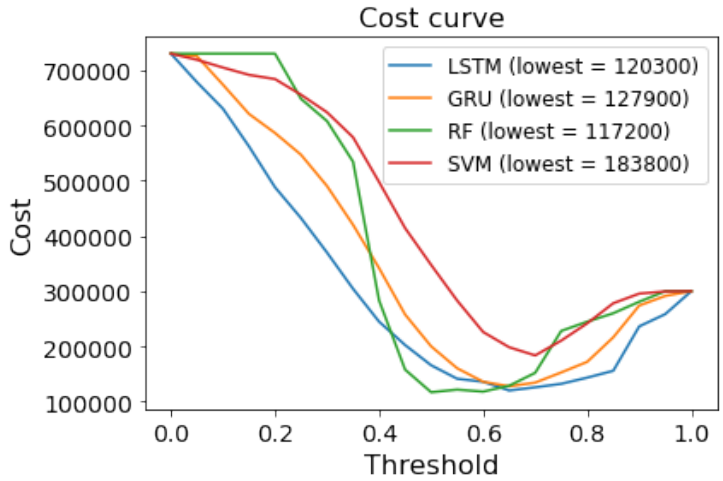

Figure 4.4: The cost associated with every model over all threshold levels.

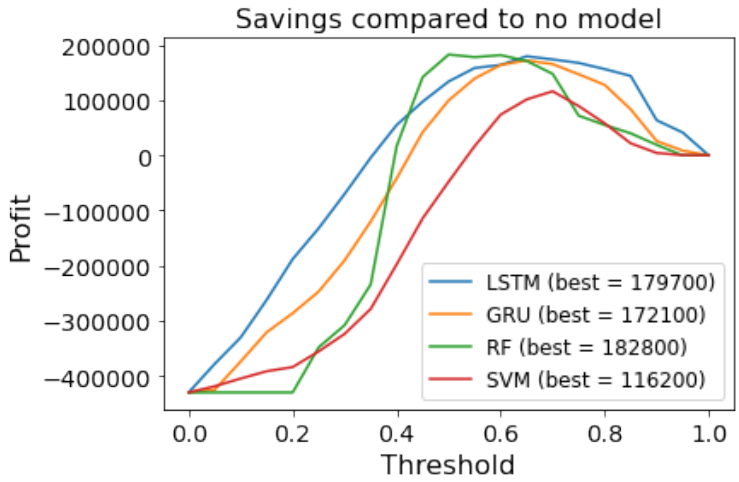

Figure 4.5: Calculated profit compared to using no model.

The RF model happens to be most profitable at the original threshold of 0.5. Interestingly, the optimal precision threshold at 0.77 is located between steep declines and appears quite low compared to GRU and LSTM at the same threshold. It may be an indicator that GRU and LSTM may perform better under typical conditions. The reason that 0.77 was not the best in this simulation is probably that the random samples no longer reflects a high concentration of churners with probability scores over 0.77. A multiple run cross-validation approach for simulating the variance in savings over each threshold value is recommended for reliability.

4.1.2 Churn time prediction

In this section, the four churn time prediction models that were developed are compared with regard to the metrics of RMSE, F1-score, and accuracy. The performance of each model is presented in Table 4.3. Additionally, the experiments on feature importance are presented.

The deep learning-based models quickly display a tendency to overfit to the data, and adding more layers just speeds up the process. Dropout did not appear to solve this problem either. The introduction of the EMD loss in the deep learning models had a minimal impact as well.

Model	**RMSE**	**F1-score**	**Accuracy**
Random (baseline)	2.80	0.14	0.135
LSTM	2.68	0.23	0.229
GRU	2.78	0.22	0.225
RF	2.99	0.21	0.205
SVM	2.53	0.21	0.210

Table 4.3: A table of churn time prediction results.

In terms of F1-score and accuracy, all of the models perform similarly. They all have a better F1-score and accuracy than the random baseline. The biggest difference between the models is reflected in the RMSE, which is noticeably higher for RF, even higher than the random baseline model. Using the specified problem formulation where seven classes represent the remaining time until churn, the RMSE can theoretically range from zero to six. Considering the even distribution of samples in each class, the range is constricted even more. Interestingly enough, the SVM implementation is the best performer in terms of RMSE by a noticeable margin.

An interesting finding for the SVM model is that the number of training samples can be as low as 15% and still achieve similar (and better) results than when using all of the data. This may be a question of hyperparameter tuning and that it just so happened that with fewer samples the SVM could build a better approximation of the data. Similar experiments were not made with the other models.

Feature importance

Most of the included features are related to the type of products a given customer has, and since there are a lot of products it makes for a lot of sparsity in the input data. Examining the relation that categories of features have on performance can help reduce the number of needed features and make possibly improve the predictive power.

The aim of the following experiment is to determine if a group of certain features are best included or left out in regard to model performance. It is done by retraining the model with a group of features excluded. The experiment is limited to the RF model, since it has the property of ranking the importance of the included features. No attempt to optimize the hyperparameters was done during this experiment. Table 4.4 show the random baseline and previously trained RF model in the top of the table, followed by RF models that exclude groups of the named features.

Model	**RMSE**	**F1-score**	**Accuracy**
Random (baseline)	2.83	0.14	0.135
RF	2.99	0.21	0.205
Survival time	3.16	0.19	0.190
Customer characteristics	3.38	0.15	0.149
Customer segments	2.70	0.22	0.216
Product groups	2.98	0.21	0.208
Products	2.86	0.21	0.206
Monetary	2.84	0.21	0.213

Table 4.4: A table of churn time prediction results when excluding features.

The performance of the original RF model was not very good to begin with, but there are still some interesting changes that take place when excluding some groups of features. By removing all customer segment related features, all metrics improved to become the all-time best for the RF model. Other excluded features either had a minimal impact or negative impact. The most im-

portant features appear to be customer characteristics and survival time. Products and monetary features which were believed to be very important turned out to have a minor impact on performance.

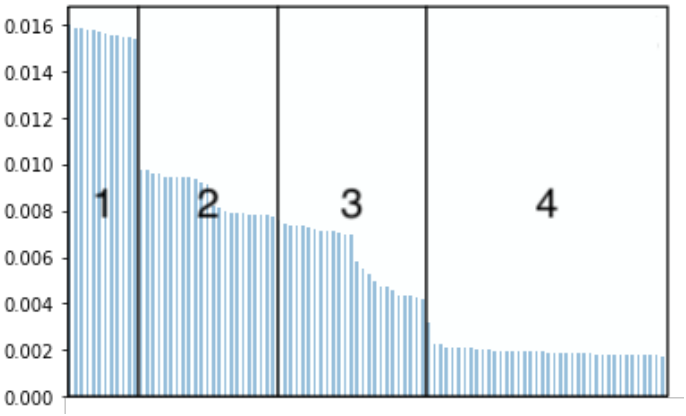

Figure 4.6: Top 100 most important features learned by RF divided into sections of feature types.

#	Feature type
1	Survival time
2	Customer characteristics
3	Customer segments
4	Monetary

Table 4.5: Description of feature types as seen in Figure 4.6.

Figure 4.6 show the score of the 100 most important features. Note that the features are column-based for the deep learning models but were flattened out to a vector for the RF model. It resulted in each original feature being divided into twelve new features representing each month of historical data in the sample. For example, the data contains monthly card transactions. The deep learning models see this as one feature with unique steps for each month (twelve months in this case). The machine learning models on the other hand, take a vector as input. Flattening the input then results in twelve separate features for the monthly card transactions. In other words, the original number of features gets multiplied by twelve. Figure 4.6 therefore usually show the same feature from different months in succession.

The numbered regions in Figure 4.6 represent different feature types, as described by Table 4.5. In Figure 4.6, the first region (1) contain all occurrences of survival time, making it the most important feature. The second region (2) contains two features from customer characteristics, these are age and postal code. The third region (3) contains two customer segment features. The fourth region (4) is mixed, containing six different monetary features with a single exception that is the number of active products during some month. There is not a single feature related to the products or product groups in the top 100 ranking. Coupled with small changes in their metrics in Table 4.4, they do not seem to matter much for predictions. The ranking is also strengthened by the fact that customer characteristics and customer segments had a visible impact on the results in Table 3.3. The noticeable drop in importance for monetary features as seen in Figure 4.6 is also in line with the results from the experiment.

4.2 Significance test

ANOVA and Tukey HSD both yield a multitude of measures, but only the p-values are considered. Cohen's d measures the effect size between model pairs. Each model was retrained 20 times using the earlier acquired optimal parameters.

Churn prediction
The resulting p-value (1.1102e-16) from ANOVA is lower than 0.05 and therefore suggests that one or more models are significantly different. The Tukey HSD test further suggests that all models are significantly different from each other, except for the pairing of LSTM and GRU where the difference is insignificant. It may be worth adding that the binary SVM model appeared to be sensitive to the initial random state, leading to a constant 10% performance loss compared to the best SVM model, which is also way worse than the average SVM model.

Churn time prediction
The resulting p-value (1.1102e-16) from ANOVA is lower than 0.05 and therefore suggests that one or more models are significantly different. The Tukey HSD test further suggests that all models are significantly better than the random baseline. Furthermore, the difference between the trained models is insignificant.

Model pair	Tukey HSD p-value	Tukey HSD inference	Cohen's d	Effect size
Baseline vs LSTM	0.0010053	significant	25.4	Huge
Baseline vs GRU	0.0010053	significant	13.0	Huge
Baseline vs RF	0.0010053	significant	95.0	Huge
Baseline vs SVM	0.0010053	significant	46.5	Huge
LSTM vs GRU	0.6251111	insignificant	0.3	Small
LSTM vs RF	0.0064411	significant	1.5	Very large
LSTM vs SVM	0.0010053	significant	15.5	Huge
GRU vs RF	0.0010053	significant	1.1	Large
GRU vs SVM	0.0010053	significant	7.6	Huge
RF vs SVM	0.0010053	significant	94.1	Huge

Table 4.6: A table of statistical significance and effect size results for the binary churn predictors. A Tukey HSD p-value of less than 0.01 suggests a significant difference between the models.

Model pair	Tukey HSD p-value	Tukey HSD inference	Cohen's d	Effect size
Baseline vs LSTM	0.0010053	significant	15.1	Huge
Baseline vs GRU	0.0010053	significant	15.8	Huge
Baseline vs RF	0.0010053	significant	39.4	Huge
Baseline vs SVM	0.0010053	significant	313.0	Huge
LSTM vs GRU	0.8999947	insignificant	0.1	Very small
LSTM vs RF	0.7598986	insignificant	0.3	Small
LSTM vs SVM	0.8013959	insignificant	0.3	Small
GRU vs RF	0.8999947	insignificant	0.2	Small
GRU vs SVM	0.8999947	insignificant	0.2	Small
RF vs SVM	0.8999947	insignificant	0.1	Very small

Table 4.7: A table of statistical significance and effect size results for the churn time predictors. A Tukey HSD p-value of less than 0.01 suggests a significant difference between the models.

Chapter 5

Discussion

In this chapter, we discuss and reflect upon the results, that showed promise in predicting churn within the coming six months while not giving reliable time estimates. The churn prediction models are potentially profitable according to simulations. Further, future work, sustainability, and ethics are discussed as well.

5.1 Results

5.1.1 Churn prediction

Even before starting this thesis, it was known that churn prediction is a difficult problem to solve accurately. If it was easy, everyone would be doing it to make profits. The standard type of binary churn prediction has been studied by many and the performance seems to be similar to what was achieved in this thesis. Even though the performance is similar, the deciding factor for considering whether a model is worth deploying is its profitability. The binary churn prediction models were shown to be profitable according to simulations using non-official costs associated with each prediction outcome. However, it still needs to be confirmed if the models are profitable when using official numbers in the calculations as the ratio of churners to non-churners and related costs of each outcome may affect the result. In terms of profit, there was little difference between LSTM, GRU, and RF. SVM did worse, perhaps due to its lower overall precision compare to other models considering that all other metrics were similar to RF. Given the speed of building an RF model, it may make more sense to favor it over LSTM and GRU.

The achieved performance is comparable to other studies on churn predic-

tion. Similar experiments in a financial setting achieved a ROC-AUC score of about 0.8 [4][34], which is about the same as the results obtained in this thesis. Jain et al. [35] achieved a ROC-AUC score of about 0.75. An experiment incorporating textual features into churn prediction of a financial service provider achieved an AUC score of 0.89 [25]. The ROC-AUC score was 0.85 without textual features, signifying that something about the data may be different from the data used in this thesis. Different characteristics of the data, churn definition, or business model will affect the results. A business model with twelve-month subscriptions may for example be easier to predict the outcome of than business models where subscriptions can be terminated at any time. Other research uses different metrics or measures different things, like measuring the accuracy and error rates on an unbalanced dataset, which makes direct comparisons difficult.

5.1.2 Churn time prediction

The other research topic was about estimating time until a churn might occur. This topic proved to be a very difficult problem where the developed models are significantly better than random predictions but fail to give reliable estimates. Initially, it was believed that recurrent neural network-based models would outperform other methods given their design that favor sequential data, but this proved to be an incorrect assumption. The models performed similarly with insignificant differences.

Comparing the churn time prediction results with other studies is difficult due to the classification approach taken in this thesis. Related research incorporates continuous estimations that use metrics that are not directly comparable to the RMSE, like the C-index or mean absolute error. The C-index measures the ordering of the predictions, which makes it unsuitable for understanding how far off the predictions are. In a study of patient survival time, Bice et al. [50] measured the distribution of errors in months, showing that the models overestimate the survival times with a mean value of about 25 months. The usefulness of the results will depend on the context of the problem. In general, discussions of results in the related research seemed to have a positive attitude.

5.2 Future work

There are a lot of possibilities that were left unexplored in this thesis, and there is some interesting research that could be explored. An interesting thing

to study is how the usage of complete customer history would affect the results instead of using just a small observation period. The appropriate length of historical data to use in every sample still need to be investigated. A length of twelve months was used in this thesis based on initial guesses and adjustments as the number of samples decreased when history length increases. A lot of historical data may not provide any gains in performance and could potentially be disregarded, as brought up in a research paper. During the experiments, it was noted that SVM performed comparably with significantly less training data than the rest. Finding out a reasonable amount of data to use for training could be another thing to investigate. Rather than looking for exact time estimates it could be useful to have different levels of urgency that encompass periods of multiple months. Since the features are believed to play a crucial part in prediction power, one could try to include some external information of for example competitor prices or the current economic situation. It is also important to understand how the model behaves over time for the same customer. Could it be that the predictions fluctuate a lot for the same customer every month, or is it steadily accurate for some customers but not so accurate for others? Does the model behave differently for different customer segments? Answering those questions might help to find a type of customer where the model works better.

The PCA components could be used as input features instead of the data. Visual analysis of class separation did at least look promising for binary churn prediction. Further experimenting and comparison with statistical survival analysis, random survival forests, WTTE-RNN, and neural network models incorporating Cox regression could give more insight into the poor performance of the developed models. It could very well be that no other model can give reliable estimates.

Before deployment, the churn model has to be tested on a much larger sample and cross-validated on real-world distributions of the data to make sure it generalizes well and have the threshold adjusted so that the model is still profitable.

5.3 Sustainability and ethics

Survival analysis in deep learning is at the forefront of current research. The initial problem was reformulated into a classification problem with some assumptions made about the predictability when using the current set of features. This thesis contributes to the real-world application of available theoretic frameworks based on churn prediction, survival analysis, machine learn-

ing, and deep learning.

Sustainability
Handling and processing of big data for deep learning require powerful hardware and can be big energy spenders. As businesses around the world grow to make use of big data and deep learning, the underlying hardware will inevitably put a lot of pressure on our environment and energy infrastructure. Rapid adaption of highly energy consuming hardware could give an incentive for the energy industry to ramp up effectiveness, but it might also give rise to political debate/disputes.

Depending on the company processes, successfully preventing churn may have other positive effects, such as minimizing the use of resources needed to produce physical documents related to the termination of contracts.

Ethics
Protecting the privacy of each customer is of utmost priority at ICA Banken. We assure that all data was de-personalized and anonymized before extracting the data to do any kind of analysis. Now, let us discuss the ethics of using each customers' data in order to affect their decisions based on our model outcome.

A customer that has the intention of terminating all of their services probably has their reason for doing so. If this intention can be detected in time, the business may try to act in order to keep their customer. This act could for example result in the customer getting beneficial offers in order to prevent them from leaving. In other words, this is specifically targeted advertising. At this point, the customer has a choice. Either accept the advantageous offer or not. If the customer did not have the slightest intention of actually terminating all of their services and received this mail, they would have a chance at better prices at no real loss. Similarly, churn prediction can be used as a tool for checking up on customers' satisfaction through for example surveys or phone calls.

In the underlying agenda of maximizing profits and minimizing losses, the customers face no actual economic harm or personal infringement. The fact that some select customers could get treated differently might upset others (perhaps longtime customers), who did not get any appreciation for their loyalty to the company. Therefore, how best to use the tool of churn prediction may be worth thinking about.

Another ethical concern arises from the nature of supervised learning as we let collected data form a model for our decisions. Sometimes, the data may be a bad generalization or make the model come to reflect unethical so-

cial behavior such as gender or race discrimination. For example, there have been numerous cases where underrepresented subgroups of customers get misclassified. In America, machine learning models for approving loan applications have through studies been found to be discriminating towards minorities. Achieving fairness in decision models can be tricky. In this thesis, we have taken measures such as not including gender in the dataset. There is various extensive research on how to mitigate discrimination in machine learning [57][58][59][60].

Chapter 6

Conclusions

This thesis compared deep learning and machine learning models on the task of churn prediction and churn time prediction. The problems were formulated as classification tasks, as opposed to the related methods of regression and survival analysis that result in estimations on a continuous timescale. The developed models are therefore only capable of predicting a set of predefined outcomes.

According to the experiments on churn prediction, LSTM and GRU had the overall best performance on most metrics. Statistical tests showed that all models are significantly better than random predictions and that there is no significant difference between LSTM and GRU. Both LSTM and GRU had a ROC-AUC score of at least 0.780, compared to RF (0.716). Simulations with reasonable assumptions showed that the developed churn prediction models can be profitable compared to not using any model at all. The simulations showed that LSTM, GRU, and RF can all yield a similar profit.

The developed models for churn time prediction is significantly better than random predictions, although the estimations are unreliable and not useful in practice. There is no significant difference between the trained models. The most important features that affect performance of the F1-scores negatively when excluded are customer characteristics (0.15) and the survival time within the observed period (0.19). Compare this with the original RF model (0.21). Further, the exclusion of customer segments actually improved the performance (0.22). With the knowledge of how features affect performance, it may be possible to exclude a lot of features in order to both improve training times and enhance model performance. The question of optimal historical data length for datapoints still stands and has to be examined.

The combination of both kinds of developed models is proposed as a solu-

tion to the churn time problem. The first step is to identify potential churners within the coming six months with a binary classification model while adjusting the classification threshold to optimize for maximum profit. Secondly, another model estimates the churn time for those customers only. While this solution does not improve the results of the churn time prediction model, it is more likely that the second model gets churners as the input, thus utilizing the small predictive power that is.

The churn prediction model by itself can be profitable, but the churn time prediction model needs further development. The work of this thesis can perhaps be considered a good starting point for future research.

Bibliography

[1] D. R. Cox. “Regression Models and Life-Tables”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 34.2 (1972), pp. 187–220.

[2] Erik Christensen. “Multivariate survival analysis using Cox’s regression model”. In: *Hepatology (Baltimore, Md.)* 7.6 (1987), pp. 1346–1358.

[3] Ping Wang, Yan Li, and Chandan Reddy. “Machine Learning for Survival Analysis: A Survey”. In: *ACM Computing Surveys (CSUR)* 51.6 (2019), pp. 1–36.

[4] C. Mena et al. “Churn Prediction with Sequential Data and Deep Neural Networks. A Comparative Analysis”. In: *arXiv* (Sept. 2019).

[5] Hossein Hassani et al. “Deep Learning and Implementations in Banking”. In: *Annals of data science* 7.3 (2020), pp. 433–446.

[6] Wagner Kamakura et al. “Choice Models and Customer Relationship Management”. In: *Marketing Letters* 16.3 (Dec. 2005), pp. 279–291.

[7] Dirk Van Den Poel and Bart Larivière. “Customer attrition analysis for financial services using proportional hazard models”. In: *European Journal of Operational Research* 157.1 (2004), pp. 196–217.

[8] J. Zhao and X. Dang. “Bank Customer Churn Prediction Based on Support Vector Machine: Taking a Commercial Bank’s VIP Customer Churn as the Example”. In: *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*. 2008, pp. 1–4.

[9] Amy Gallo. “The Value of Keeping the Right Customers”. In: *Harvard Business Review* (Oct. 2014).

[10] John Hadden et al. “Computer assisted customer churn management: State-of-the-art and future trends”. In: *Computers & Operations Research* 34.10 (2007), pp. 2902–2917.

[11] C. Bhattacharya. “When customers are members: Customer retention in paid membership contexts”. In: *Journal of the Academy of Marketing Science* 26.1 (1998), pp. 31–44.

[12] Alejandro Correa Bahnsen, Djamila Aouada, and Björn Ottersten. “A novel cost-sensitive framework for customer churn predictive modeling”. In: *Decision Analytics* 2:5 (June 2015).

[13] Paul D. Allison. *Event History and Survival Analysis*. 2nd ed. Thousand Oaks, California: SAGE Publications, Inc., Aug. 2014.

[14] Dirk F. Moore. *Applied Survival Analysis Using R*. Cham, Switzerland: Springer International Publishing, Jan. 2016.

[15] E. L. Kaplan and Paul Meier. “Nonparametric Estimation from Incomplete Observations”. In: *Journal of the American Statistical Association* 53.282 (1958), pp. 457–481.

[16] Filippo Maria Bianchi et al. “Recurrent Neural Networks for Short-Term Load Forecasting”. In: *SpringerBriefs in Computer Science* (2017).

[17] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80.

[18] Junyoung Chung et al. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *NIPS 2014 Deep Learning and Representation Learning Workshop* (2014).

[19] Leo Breiman et al. *Classification and Regression Trees*. Chapman & Hall, Jan. 1984.

[20] Tin Kam Ho. “Random decision forests”. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. 1995, pp. 278–282.

[21] Leo Breiman. “Random Forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32.

[22] Vladimir Vapnik. “Pattern recognition using generalized portrait method”. In: *Automation and Remote Control* 24 (1963), pp. 774–780.

[23] Vladimir Vapnik and Alexei Ya. Chervonenkis. “A note on one class of perceptrons”. In: *Automation and Remote Control* 25 (Jan. 1964).

[24] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. “A Training Algorithm for Optimal Margin Classifier”. In: *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory (COLT ’92)* 5 (July 1992), pp. 144–152.

[25] Arno De Caigny et al. “Incorporating textual information in customer churn prediction models based on a convolutional neural network”. In: *International Journal of Forecasting* 36 (4 2019), pp. 1563–1578.

[26] Abbas Keramati, Hajar Ghaneei, and Seyed Mirmohammadi. “Developing a prediction model for customer churn from electronic banking services using data mining”. In: *Financial Innovation* 2 (Aug. 2016), pp. 1–13.

[27] Michel Ballings and Dirk Van den Poel. “Customer event history for churn prediction: How long is long enough?” In: *Expert Systems with Applications* 39.18 (2012), pp. 13517–13522.

[28] Nitesh Chawla et al. “SMOTE: Synthetic Minority Over-sampling Technique”. In: *Journal of Artificial Intelligence Research (JAIR)* 16 (Jan. 2002), pp. 321–357.

[29] Dudyala Kumar and Vadlamani Ravi. “Predicting credit card customer churn in banks using data mining”. In: *International Journal of Data Analysis Techniques and Strategies* 1 (Feb. 2008), pp. 4–28.

[30] J. Burez and D. Van den Poel. “Handling class imbalance in customer churn prediction”. In: *Expert Systems with Applications* 36.3, Part 1 (2009), pp. 4626–4636.

[31] M.A.H. Farquad, Vadlamani Ravi, and S. Bapi Raju. “Churn prediction using comprehensible support vector machine: An analytical CRM application”. In: *Applied Soft Computing* 19 (2014), pp. 31–40.

[32] Teemu Mutanen, Sami Nousiainen, and Jussi Ahola. “Customer churn prediction - A case study in retail banking”. In: *Data Mining for Business Applications* 218 (Aug. 2010), pp. 77–83.

[33] Yuzhou Chen et al. “Deep Ensemble Classifiers and Peer Effects Analysis for Churn Forecasting in Retail Banking”. In: *Advances in Knowledge Discovery and Data Mining*. Cham, Switzerland: Springer International Publishing, 2018, pp. 373–385.

[34] Erdem Kaya et al. “Behavioral attributes and financial churn prediction”. In: *EPJ data science* 7.1 (2018), pp. 1–18.

[35] Himani Jain, Garima Yadav, and R Manoov. “Churn Prediction and Retention in Banking, Telecom and IT Sectors Using Machine Learning Techniques”. In: *Advances in Machine Learning and Computational Intelligence*. Algorithms for Intelligent Systems. Singapore: Springer Singapore, 2020, pp. 137–156.

[36] Hemlata Dalmia, Ch V S S Nikil, and Sandeep Kumar. "Churning of Bank Customers Using Supervised Learning". In: *Innovations in Electronics and Communication Engineering*. Vol. 107. Lecture Notes in Networks and Systems. Singapore: Springer Singapore, 2020, pp. 681–691.

[37] Akash Sampurnanand Pandey and K. K Shukla. "Application of Bayesian Automated Hyperparameter Tuning on Classifiers Predicting Customer Retention in Banking Industry". In: *Data Management, Analytics and Innovation*. Vol. 1175. Advances in Intelligent Systems and Computing. Singapore: Springer Singapore, 2020, pp. 83–100.

[38] Rong Zhang et al. "Deep and Shallow Model for Insurance Churn Prediction Service". In: IEEE, 2017, pp. 346–353.

[39] Chiun-Sin Lin, Gwo-Hshiung Tzeng, and Yang-Chieh Chin. "Combined rough set theory and flow network graph to predict customer churn in credit card accounts". In: *Expert Systems with Applications* 38.1 (2011), pp. 8–15.

[40] J. Runge et al. "Churn prediction for high-value players in casual social games". In: *2014 IEEE Conference on Computational Intelligence and Games*. 2014, pp. 1–8.

[41] Seungwook Kim et al. "Churn prediction of mobile and online casual games using play log data". In: *PloS one* 12.7 (2017).

[42] Jonathan Burez and Dirk Van den Poel. "CRM at a pay-TV company: Using analytical models to reduce customer attrition by targeted marketing for subscription services". In: *Expert Systems with Applications* 32.2 (2007), pp. 277–288.

[43] Anuj Sharma and Prabin Kumar Panigrahi. "A Neural Network based Approach for Predicting Customer Churn in Cellular Network Services". In: *International Journal of Computer Applications* 27 (2013).

[44] Sanjay Kumar and Manish Kumar. "Predicting Customer Churn Using Artificial Neural Network". In: *Engineering Applications of Neural Networks*. Vol. 1000. Communications in Computer and Information Science. Cham, Switzerland: Springer International Publishing, 2019, pp. 299–306.

[45] Jared L. Katzman et al. "DeepSurv: personalized treatment recommender system using a Cox proportional hazards deep neural network". In: *BMC Medical Research Methodology* 18.1 (Feb. 2018).

[46] Håvard Kvamme, Ørnulf Borgan, and Ida Scheel. "Time-to-Event Prediction with Neural Networks and Cox Regression". In: *Journal of Machine Learning Research* 20 (2019), 129:1–129:30.

[47] Stefan Leger et al. "A comparative study of machine learning methods for time-to-event survival data for radiomics risk modelling". In: *Scientific reports* 7.1 (2017), pp. 13206–11.

[48] Hong Wang and Gang Li. "Extreme learning machine Cox model for high-dimensional survival analysis". In: *Statistics in medicine* 38.12 (2019), pp. 2139–2156.

[49] Safoora Yousefi et al. "Predicting clinical outcomes from large scale cancer genomic profiles with deep survival models". In: *Scientific reports* 7.1:11707 (2017).

[50] Noah Bice et al. "Deep learning-based survival analysis for brain metastasis patients with the national cancer database". In: *Journal of applied clinical medical physics* 21.9 (2020), pp. 187–192.

[51] Özden Gür Ali and Umut Arıtürk. "Dynamic churn prediction framework with more effective use of rare event data: The case of private banking". In: *Expert Systems with Applications* 41.17 (2014), pp. 7889–7903.

[52] Egil Martinsson. "WTTE-RNN : Weibull Time To Event Recurrent Neural Network". Master's thesis. Chalmers University of Technology, 2017.

[53] Shlomo Sawilowsky. "New Effect Size Rules of Thumb". In: *Journal of Modern Applied Statistical Methods* 8 (Nov. 2009), pp. 597–599.

[54] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Mahwah, New Jersey: Lawrence Erlbaum Associates, 1988.

[55] Marnie E Rice and Grant T Harris. "Comparing Effect Sizes in Follow-Up Studies: ROC Area, Cohen's d, and r". In: *Law and human behavior* 29.5 (2005), pp. 615–620.

[56] Susan K. Grove and Daisha J. Cipher. *Statistics for Nursing Research: a Workbook for Evidence-Based Practice*. 2nd ed. Amsterdam, The Netherlands: Elsevier, 2016.

[57] Andreas Fuster et al. "Predictably Unequal? The Effects of Machine Learning on Credit Markets". In: *Banking & Insurance eJournal* (2017).

[58] Indre Zliobaite. "A survey on measuring indirect discrimination in machine learning". In: *arXiv* (Oct. 2015).

[59] Adrienne Yapo and Joseph Weiss. “Ethical Implications of Bias in Machine Learning”. In: *Hawaii International Conference on System Sciences (HICSS)*. Jan. 2018.

[60] Michael Veale and Reuben Binns. “Fairer machine learning in the real world: Mitigating discrimination without collecting sensitive data”. In: *Big Data & Society* 4.2 (Nov. 2017).

Appendix A

Customer characteristics features

Feature	Type	Description
Age	Integer	Current age.
Postal code	Integer	First two digits of postal code.
Student	Boolean	Student status.
ObservedTime	Integer	Active time during observation.
Moved	Boolean	Full postal code changed since last month.

Table A.1: Customer characteristics features.

Appendix B

Customer segments features

Feature	Type	Description
ICASegmentCode	Integer	Customer segment codes.
MosaicSegmentCode	Integer	Categorical expressed numerically.
RecencyICA	Integer	Recency score.
RecencyExt	Integer	Recency score.
RecencyTotal	Integer	Recency score.
FrequencyICA	Integer	Frequency score.
FrequencyExt	Integer	Frequency score.
FrequencyTotal	Integer	Frequency score.
MonetaryICA	Integer	Monetary score.
MonetaryExt	Integer	Monetary score.
MonetaryTotalt	Integer	Monetary score.
RFMICASUM	Integer	RFM score.
RFMEXTSUM	Integer	RFM score.
RFMTotSUM	Integer	RFM score.

Table B.1: Customer segments features.

Appendix C

Product groups features

Feature	Type	Description
Blanco	Boolean	Has unsecured debt.
Mortgage	Boolean	Has mortgage debt.
BankCard	Boolean	Has one of ICA card offerings.
ICACard	Boolean	Has an ICA membership card.
CardAndAccCredit	Boolean	Has card and available credit.
FamInsurance	Boolean	Has family insurance products.
VehicleInsurance	Boolean	Has some type of vehicle insurance.
LoanProtection	Boolean	Has loan protection.
CivilInsurance	Boolean	Has civil insurance products.
AnimalInsurance	Boolean	Has some type of animal insurance.
TravelInsurance	Boolean	Has some travel insurance product.
TransactionAcc	Boolean	Has some card product for transactions.
Savings	Boolean	Has some savings product.
Insurance	Boolean	Has some insurance product.

Table C.1: Product groups features.

Appendix D

Products features

Feature	Type	Description
61 products	Boolean	Product is active.

Table D.1: Products features.

Appendix E

Monetary features

Feature	Type	Description
NumTransactionsICA	Integer	Number of transactions at ICA.
NumTransactionsExt	Integer	Number of transactions externally.
NumPurchasesICA	Integer	Number of purchases at ICA.
NumPurchasesRolling12ICA	Integer	Number of purchases at ICA during the past 12 months.
NumPurchasesRolling12Ext	Integer	Number of purchases externally during the past 12 months.
VolumeICA	Float	Spendings at ICA.
VolumeExt	Float	Spendings externally.
Volume	Float	Spendings in total.
VolumeRolling12ICA	Float	Spendings in total at ICA during the past 12 months.
VolumeRolling12Ext	Float	Spendings in total externally during the past 12 months.
NumMoSincePurchaseICA	Integer	Number of months since last purchase.
NumMoSincePurchaseExt	Integer	Number of months since last external purchase.

Table E.1: Monetary features.

TRITA -EECS-EX-2020:931