TURUN YLIOPISTO

# MACHINE LEARNING APPLICATIONS FOR CENSORED DATA

Markus Viljanen

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. F OSA – TOM. 1 | TECHNICA – INFORMATICA | TURKU 2021

## University of Turku

Faculty of Science and Engineering
Department of Computing
Computer Science
MATTI

## Supervisors

| | |
|---|---|
| Tapio Pahikkala<br>University of Turku | Antti Airola<br>University of Turku |
| Jukka Heikkonen<br>University of Turku | |

## Reviewers

| | |
|---|---|
| Alexander Jung<br>Aalto University | Fabio Aiolli<br>University of Padua |

## Opponent

Arto Klami
University of Helsinki

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

ISBN 978-951-29-8415-2 (PDF)
ISSN 2736-9390 (Painettu)
ISSN 2736-9684 (Sähköinen)
Turku, Finland 2021

TURUN YLIOPISTO
Teknillinen tiedekunta
Tietotekniikan laitos
Tietojenkäsittelytiede
MARKUS VILJANEN: Machine Learning Applications for Censored Data
Väitöskirja, 189 s.
MATTI
huhtikuu 2021

## TIIVISTELMÄ

Kerätyn datan määrä on kasvanut kun digitalisoituminen on edennyt. Itse data ei kuitenkaan ole arvokasta, vaan tavoitteena on käyttää dataa tiedon hankkimiseen ja uusissa sovelluksissa. Suurin haaste onkin menetelmäkehityksessä: miten voidaan kehittää koneita jotka osaavat käyttää dataa hyödyksi? Monien alojen yhtymäkohtaa onkin kutsuttu Datatieteeksi (Data Science). Sen tavoitteena on ymmärtää, miten tietoa voidaan systemaattisesti saada sekä strukturoiduista että strukturoimattomista datajoukoista. Koneoppiminen voidaan nähdä osana datatiedettä, kun tavoitteena on rakentaa ennustavia malleja automaattisesti datasta ns. yleiseen oppimisalgoritmiin perustuen ja menetelmän fokus on ennustustarkkuudessa.

Monet käytännön ongelmat voidaan muotoilla kysymyksinä, jota kuvaamaan on kerätty dataa. Ratkaisu vaikuttaakin koneoppimisen kannalta helpolta: määritellään datajoukko syötteitä ja oikeita vastauksia, ja kun koneoppimista sovelletaan tähän datajoukkoon niin vastaus opitaan ennustamaan. Monissa käytännön ongelmissa oikeaa vastausta ei kuitenkaan ole täysin saatavilla, koska datan kerääminen voi kestää vuosia. Jos esimerkiksi halutaan ennustaa miten paljon rahaa eri asiakkaat kuluttavat elinkaarensa aikana, täytyisi periaatteessa odottaa kunnes yrityksen kaikki asiakkaat lopettavat ostosten tekemisen jotta nämä voidaan laskea yhteen lopullisen vastauksen saamiseksi. Kutsumme tämänkaltaista datajoukkoa ’sensuroiduksi’; oikeat vastaukset on havaittu vain osittain koska esimerkkien kerääminen syötteistä ja oikeista vastauksista voi kestää vuosia.

Tämä väitös esittelee koneoppimisen uusia sovelluksia sensuroituihin datajoukkoihin, ja tavoitteena on vastata kaikkein tärkeimpään kysymykseen kussakin sovelluksessa. Sovelluksina ovat mm. digitaalinen markkinointi, vertaislainaus, työttömyys ja pelisuosittelu. Ratkaisu ottaa huomioon sensuroinnin, siinä missä edelliset ratkaisut ovat saaneet vääristyneitä tuloksia tai keskittyneet ratkaisemaan yksinkertaisempaa ongelmaa datajoukoissa, joissa sensurointi ei ole ongelma. Ehdottamamme ratkaisu perustuu kolmeen vaiheeseen jossa yhdistyy ongelman matemaattinen ymmärrys ja koneoppiminen: 1) ongelma dekonstruoidaan parittaisena datana 2) koneoppimista sovelletaan puuttuvien parien ennustamiseen 3) oikea vastaus rekonstruoidaan ennustetuista pareista. Abstraktilla tasolla idea on kaikissa paperissa sama, mutta jokaisessa sovelluksessa hyödynnetään sitä varten suunniteltua koneoppimismenetelmää ja parittaista kuvausta.

ASIASANAT: Koneoppiminen, parittainen data, sensurointi.

UNIVERSITY OF TURKU
Faculty of Technology
Department of Computing
Computer Science
MARKUS VILJANEN: Machine Learning Applications for Censored Data
Doctoral Dissertation, 189 pp.
MATTI
April 2021

## ABSTRACT

The amount of data being gathered has increased tremendously as many aspects of our lives are becoming increasingly digital. Data alone is not useful, because the ultimate goal is to use the data to obtain new insights and create new applications. The largest challenge of computer science has been the largest on the algorithmic front: how can we create machines that help us do useful things with the data? To address this challenge, the field of data science has emerged as the systematic and interdisciplinary study of how knowledge can be extracted from both structed and unstructured data sets. Machine learning is a subfield of data science, where the task of building predictive models from data has been automated by a general learning algorithm and high prediction accuracy is the primary goal.

Many practical problems can be formulated as questions and there is often data that describes the problem. The solution therefore seems simple: formulate a data set of inputs and outputs, and then apply machine learning to these examples in order to learn to predict the outputs. However, many practical problems are such that the correct outputs are not available because it takes years to collect them. For example, if one wants to predict the total amount of money spent by different customers, in principle one has to wait until all customers have decided to stop buying to add all of the purchases together to get the answers. We say that the data is ’censored’; the correct answers are only partially available because we cannot wait potentially years to collect a data set of historical inputs and outputs.

This thesis presents new applications of machine learning to censored data sets, with the goal of answering the most relevant question in each application. These applications include digital marketing, peer-to-peer lending, unemployment, and game recommendation. Our solution takes into account the censoring in the data set, where previous applications have obtained biased results or used older data sets where censoring is not a problem. The solution is based on a three stage process that combines a mathematical description of the problem with machine learning: 1) deconstruct the problem as pairwise data, 2) apply machine learning to predict the missing pairs, 3) reconstruct the correct answer from these pairs. The abstract solution is similar in all domains, but the specific machine learning model and the pairwise description of the problem depends on the application.

KEYWORDS: Machine Learning, Pairwise learning, Censoring

# Contents

# List of abbreviations

| | |
|---|---|
| KDDM | Knowledge Discovery and Data Mining |
| LTV | LifeTime Value |
| RBF | Radial Basis Function |
| i.i.d. | Independent and Identically Distributed |
| MLE | Maximum Likelihood Estimate |
| GEE | Generalized Estimating Equation |
| MVN | Multivariate Normal Distribution |
| MCF | Mean Cumulative Function |
| P2P | Peer-to-Peer |
| DCF | Discounted Cash Flow |
| LGD | The Loss Given Default |
| EAD | Exposure at Default |
| SVD | Singular Value Decomposition |
| kNN | k Nearest Neighbour |
| ALS | Alternating Least Squares |
| CF | Collaborative Filtering |
| CB | Content Based |
| ME | Mean Error |
| MSE | Mean Squared Error |
| AUC | Area Under the ROC Curve |
| LM | Linear Model |
| LME | Linear Mixed Effects |
| LML | Linear Machine Learning |

# List of original publications

The thesis is based on the following publications:

I Viljanen, M., Airola, A., Heikkonen, J., & Pahikkala, T. (2017). Playtime measurement with survival analysis. IEEE Transactions on Games, 10(2), 128-138.

II Viljanen, M., Airola, A., Heikkonen, J., & Pahikkala, T. (2017, September). A/B-test of retention and monetization using the Cox model. In Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference.

III Viljanen, M., Airola, A., Majanoja, A. M., Heikkonen, J., & Pahikkala, T. (2020). Measuring player retention and monetization using the mean cumulative function. IEEE Transactions on Games, 12(1), 101-114.

IV Byanjankar, A., & Viljanen, M. (2019). Predicting Expected Profit in Ongoing Peer-to-Peer Loans with Survival Analysis Based Profit Scoring. In Intelligent Decision Technologies 2019 (pp. 15-26). Springer, Singapore.

V Viljanen, M. , Byanjankar, A., & Pahikkala, T. (2020). Predicting profitability of peer-to-peer loans with recovery models for censored data. In Intelligent Decision Technologies 2020 (pp. 15-25). Springer, Singapore.

VI Viljanen, M., & Pahikkala, T. (2020). Predicting Unemployment with Machine Learning Based on Registry Data. In 2020 14th International Conference on Research Challenges in Information Science (RCIS) *(pp. 352-368)*. IEEE.

VII Viljanen, M., Pahikkala, T., Vahlo, J., & Koponen, A. (2020). Content Based Player and Game Interaction Model for Game Recommendation in the Cold Start setting. IEEE Transactions on Games, *submitted*.

The following publications were left outside this thesis:

VIII Viljanen, M., Airola, A., Pahikkala, T., & Heikkonen, J. (2016, September). Modelling user retention in mobile games. In 2016 IEEE Conference on Computational Intelligence and Games (CIG) (pp. 1-8). IEEE..

IX Viljanen, M., Airola, A., Pahikkala, T., & Heikkonen, J. (2016, September). User activity decay in mobile games determined by simple differential equations?. In 2016 IEEE Conference on Computational Intelligence and Games (CIG) (pp. 1-8). IEEE.

X Numminen, R., Viljanen, M., & Pahikkala, T. (2019, August). Predicting the monetization percentage with survival analysis in free-to-play games. In 2019 IEEE Conference on Games (CoG) (pp. 1-8). IEEE.

XI Numminen, R., Viljanen, M. J., & Pahikkala, T. (2020). Bayesian inference for predicting the monetization percentage in free-to-play games. IEEE Transactions on Games.

XII Viljanen, M., Airola, A., & Pahikkala, T. (2020). Generalized vec trick for fast learning of pairwise kernel models, *submitted* http://arxiv.org/abs/2009.01054

# 1 Introduction

## 1.1 Understanding and predicting with data

The amount of data in the world is growing exponentially. For example, a 2011 study in Science [1] estimated that the yearly growth rates are 58% for computing capacity, 28% for bidirectional communication, and 23% for storage space. The ability to gather and access staggering amounts of data does not necessarily guarantee that one benefits from it. Many useful insights could potentially be obtained from the data if we only had the means to obtain them. While ubiquitous data sources undoubtedly offer greater opportunities to extract information, the primary challenge has been in the algorithmic front: how to make sense of it all? New approaches and algorithms are needed to transform the data deluge into a useful source of knowledge.

Technological developments have made gathering and processing data feasible in many applications. Early approaches to data analysis concentrated on storing data into databases and using it to measure and automate business processes. Data mining gained recognition as a possible solution to problems where knowledge needs to be automatically extracted from large and unstructured data sets. However, there has been increasing recognition that some problems can be formulated as well-defined questions that need accurate answers. The problem is therefore to give answers based on the data set with the help of computers. When questions are made repeatedly in a systematic fashion and the data does not have direct answers in every instance, we can automate the process by constructing a predictive model.

Research on predictive models has found that relatively simple models perform better than human experts in multiple fields, especially when the task requires to synthetize different sources of information into a prediction under uncertainty [2]. It is sometimes possible to outperform humans with simple rules even without explicit models [3]. However, there are many tasks where simple models have not had much success but predictive models based on machine learning have lead to substantial progress in the last two decades. These fields include computer vision, speech recognition, natural language processing, and robot control, to name a few [4]. The observation that data and predictive models can help in making more informed decisions has led to the concept of data driven decision making, where it is recognized that data is the ultimate arbiter of facts and business success [5]. Early approaches have now developed into a systematic study sometimes called ’Data Science’, an umbrella term for the scientific approach of reasoning from data [6].

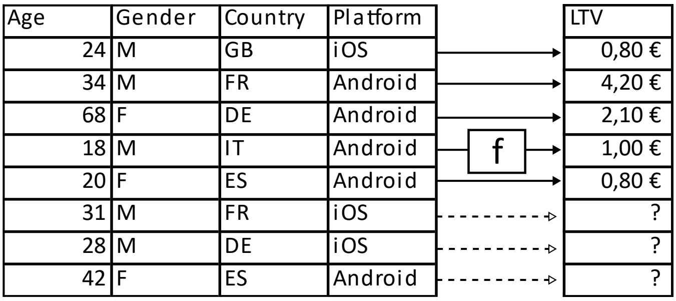
A project that utilizes data science can be thought of as a process that consists of several stages. The data has to be recorded, stored, processed, and then modelled. The model has to answer the business problem that was proposed and the resulting model needs to be implemented in production with all relevant factors considered. Several process models have been described in the knowledge discovery and data mining (KDDM) field to manage the approach of extracting knowledge from data [7]. One example is the CRISP-DM process model, which proposes that a project has six major stages:

1. Business understanding: The goal is to identify and understand a relevant business problem. It is then formulated as a data science problem.
2. Data understanding: A data set is collected and it is investigated to gain better insight into the problem.
3. Data preparation: The data set is processed for the purpose of modelling: this may include merging, cleaning, imputing missing values, scaling, standardizing, etc.
4. Modelling: A model is selected and optimized to fit the data set. Additional models and different modelling choices may be investigated.
5. Evaluation: The model performance is carefully evaluated in a setting that reflects the use case. We ask if the model answers the business problem.
6. Deployment: The model is implemented as a business application, where monitoring and maintenance are also considered.

The stages do not necessarily follow each other and the project may move back and forth between various stages. For example, it is quite typical that the project specification changes based on increased understanding of data and modelling results. One then goes back to the initial stages to refine the project for better results. Data science projects are often cyclical in this way. Solutions benefit from previous experiences and business understanding increases as the project moves forward.

## 1.2 Machine learning and censored data

Machine learning is a subfield of data science, where the goal is to develop algorithms that learn given a task from data without explicit instructions. Computers that learn can be very useful, because many problems have plenty of data available but it is difficult to invent a clear sequence of instructions to achieve the desired goal. Programming computers to perform complicated tasks with many steps that require explicit machine instructions can be too costly or even impossible. Machine learning is instead based on learning how to give the correct answer from the data itself. Given examples of inputs and correct outputs, we learn to predict an approximately correct output for any future input. This task is achieved by a generic learning algorithm. The learning algorithm searches a space of candidate programs to find an optimal program, which is judged relative to some performance metric. The performance metric is based on the difference of predicted outputs from the correct outputs and guides the learning process. All machine learning methods are based on this paradigm, but they differ in the learning setting, the space of possible programs and the optimization method used to find the optimal program.



| Age | Gender | Country | Platform | LTV |
|---|---|---|---|---|
| 24 | M | GB | iOS | 0,80 € |
| 34 | M | FR | Android | 4,20 € |
| 68 | F | DE | Android | 2,10 € |
| 18 | M | IT | Android | 1,00 € |
| 20 | F | ES | Android | 0,80 € |
| 31 | M | FR | iOS | ? |
| 28 | M | DE | iOS | ? |
| 42 | F | ES | Android | ? |

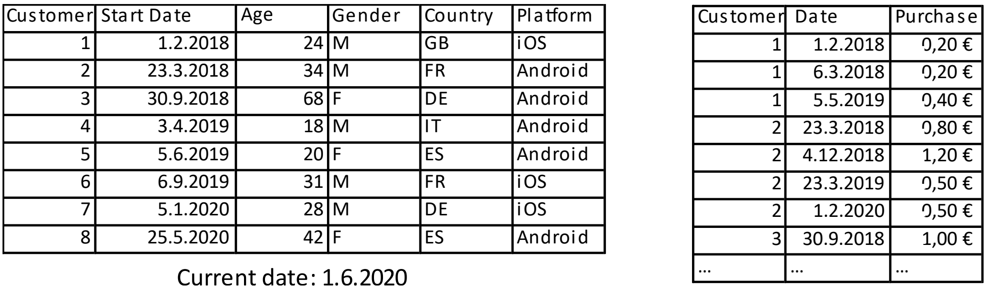
**Figure 1.** A simple example: each row describes a customer with their features (Age, Gender, Country, Platform) and the total amount of money spent. The task is to learn an unknown function *f* from this data set that predicts the total amount of money spent.

To illustrate this with a simple example, consider the task of predicting customer lifetime values (LTV) in **Figure 1**. The data set is a table that consists of examples of inputs and outputs as rows. The inputs are customer features: age, gender, country, and platform. The output is the measured LTV. Formally, denote the data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1,\dots,n}$ of examples with customer features $x_i \in \mathbb{R}^d$ and LTV $y_i \in \mathbb{R}$. The LTV values $y_i = f(x_i)$ are assumed to represent an unknown function $f$. The task is to learn a function $\hat{f}$ that approximates the unknown function $\hat{f} \approx f$. Machine learning considers accurate predictions as the most important problem and the function can be thought of as a black box: input goes in and output comes out. Having learned the function we can predict $y = \hat{f}(x)$ for any new input $x$, which means we can predict the LTV of new customers. For example, we could then choose to market only to customers where this value is higher than the cost of advertising to them.

The data science project looks simple on the surface. Ask a relevant question and transform the data into examples so that a machine learning model can be applied to it: vectors of inputs and correct answers as outputs. We then get a function $y = \hat{f}(x)$ that answers our question. However, there are many real world situations where the correct answer is not available in data. Businesses often cannot afford to wait for years to collect a historical data set of examples for their current problem. Instead, data is recorded in real time and the problem is to model the phenomena with increasing accuracy as new data arrives. Many problems therefore wish to model the eventual answer in a setting without examples of fully observed correct answers.

For example, the task of LTV prediction in **Figure 1** implicitly assumed that we have a historical data set spanning many years so that all customer purchases can be added together to get the correct LTVs. We only know the final values when the customers in the data have stopped purchasing altogether. However, a real business may not be able to wait for years to gather a data set that directly answers this question. In practise, we have a data set that looks like **Figure 2**, where we only know the purchases that have occurred so far. We are left wondering how this can be utilized to correctly calculate the LTV, so that machine learning can be used as a predictive model. In this study, we call all data sets where the correct answers are not fully observed as censored.

Censored data is not a new occurrence. In fact, it is a central focus in the fields of survival analysis [8] and reliability engineering [9]. Their models consider time-to-event data; the lifetime of a biological organism or time to equipment failure are classic examples. It has been shown that one needs to take into account censoring to obtain unbiased results. Additionally, it is often helpful to consider the special structure of the data to understand the event process that generated the data [10]. However, these fields analyse a special case of censored data because the outcome variable (time-to-event) is always positive and corresponds to the follow-up duration if censored. Some parts of our data sets are a direct instance of time-to-event data: for example the playtime of a player, the default time of a P2P loan, and a single unemployment spell length. In this thesis, the methods are often inspired by survival analysis and incorporate some aspects of their models, but they deal with more general questions. For example, we consider the lifetime value of a player, the profit of a P2P loan, and the lifetime unemployment of an individual.



| Customer | Start Date | Age | Gender | Country | Platform |
|---|---|---|---|---|---|
| 1 | 1.2.2018 | 24 | M | GB | iOS |
| 2 | 23.3.2018 | 34 | M | FR | Android |
| 3 | 30.9.2018 | 68 | F | DE | Android |
| 4 | 3.4.2019 | 18 | M | IT | Android |
| 5 | 5.6.2019 | 20 | F | ES | Android |
| 6 | 6.9.2019 | 31 | M | FR | iOS |
| 7 | 5.1.2020 | 28 | M | DE | iOS |
| 8 | 25.5.2020 | 42 | F | ES | Android |

Current date: 1.6.2020

| Customer | Date | Purchase |
|---|---|---|
| 1 | 1.2.2018 | 0,20 € |
| 1 | 6.3.2018 | 0,20 € |
| 1 | 5.5.2019 | 0,40 € |
| 2 | 23.3.2018 | 0,80 € |
| 2 | 4.12.2018 | 1,20 € |
| 2 | 23.3.2019 | 0,50 € |
| 2 | 1.2.2020 | 0,50 € |
| 3 | 30.9.2018 | 1,00 € |
| … | … | … |

**Figure 2.** A real data set may look like this: the total amount of money spent is not known, but we have a data base of purchases that have occurred so far. This means that the correct answers are not known due to a limited follow-up time (censored).

## 1.3 Research goal and methods

We have an important prediction problem but cannot apply standard methods in supervised machine learning because the correct answers have been only partially observed. In this research, we develop new models for machine learning in censored data sets that attempt to answer the most relevant question in each application. The applications are multidisciplinary: we develop models in marketing of digital products, peer-to-peer lending, unemployment, and game recommendation. These problems were motivated by our involvement in different collaboration projects with the industry and governmental institutions. These problems are quite new, except for the study of unemployment, and there has been a limited amount of research in applying machine learning to these problems. The development of new methods was motivated by the fact that we thought a satisfactory solution had not been given to the more difficult problems that often involve censored data.

In this thesis introduction, we discuss each method under an unified framework as pairwise data. We invented this abstraction afterwards to discuss every problem as an instance of a single problem, but it is not immediately obvious how a particular application is an instance of pairwise data. From a practical viewpoint, the models are more specialized than general machine learning models because they utilize the censored data in a special way. On a theoretical level, every solution is based on understanding the data set as pairs where some pairs are missing due to censoring. Machine learning can then be applied to the pairs and the final predictions are then constructed from the predictions for all of the pairs. For example, the LTV data can be understood as examples of purchases, where each purchase is associated with a given customer and time pair. The input is the customer features and time, the output is the purchase amount. Machine learning algorithms with a specific model formulation can then be applied to predict the purchase amounts at each customer and time pair. The final LTV prediction can be calculated by summing together the purchase amounts in these predicted pairs. The mathematical description of how we formulate the model and construct the final answer from these pairs depends on the domain, but the paradigm of understanding a censored data set as pairwise data to apply machine learning is shared between all of them.

The idea of formulating censored data as pairwise data has implications for all of the stages in the data science project. First, the business understanding stage can consider questions where the answers are not fully known. The data understanding stage requires deep knowledge of the problem in order to formulate it as a pairwise problem. The data preparation stage then formats the data as pairwise observations. The modelling stage may need to consider the specifics of how predictions are formed for the pairs: different functions, optimization methods, or assumptions about how the observations are related. Finally, the evaluation stage needs to take into account that many of the examples come from the same customer or time, and the data may not satisfy the standard independent and identically distributed (i.i.d) assumption that would be implicit in a straightforward train and test set split.

In this thesis, we summarize the research objectives in the following questions:

(Q1) How to formulate a specific mathematical description of the problem and the corresponding machine learning model to answer important business questions in censored data sets from various domains? Specifically, we consider the problem of measuring retention and monetization in digital products, predicting profits in P2P lending, predicting individual's lifetime risk of unemployment, and recommending video games to players under different settings.

(Q2) How to obtain unbiased evaluation of model accuracy in these applications, where the transformed data set has a pairwise structure and is therefore not independent and identically distributed, with different cross-validation strategies? How are different settings implied by pairwise data, for example predictions for new users or new time points, correctly taken into account in the validation?

The research results obtained to the first question provide evidence that machine learning can be applied in censored data settings and present new solutions that can be used in these applications. The results suggest a general modelling process based on pairwise understanding of the data that could be considered in future applications. The results to the second question provide validation strategies for our models that take into account the not i.i.d. aspect of the data and obtain valid prediction accuracy estimates. The different validation strategies also consider how to correctly measure prediction accuracy under different generalization settings implied by pairwise data.

## 1.4 Organization of the thesis

This thesis consists of two separate parts. Part I is the introduction to the thesis and consists of Chapters 1-4. Part II consists of the original research publications that are included in the thesis. Chapter 1 gives a general introduction to the subject and motivates the research questions considered in this thesis. Chapter 2 presents the theoretical framework behind machine learning and the models considered in this thesis. Chapter 3 summarizes the publications and the author's contributions. Chapter 4 concludes the thesis.

# 2 Theoretical foundation

## 2.1 Machine Learning

In many situations we do not have an explicit solution to a given prediction problem, but there is plenty of data to describe the problem. The data can be used to give an empirical solution. Often simple laws or generalizations are hypothesized to describe the phenomena to a high degree of accuracy and we verify whether the data follows them. This is the case in many fields of science and engineering [11]. These solutions often use some degree of expert knowledge, meaning knowledge about previous observations and established facts. However, many different problems can be thought of as instances of a single abstract problem: the process of constructing predictive models based on data. The systematic study of learning generic models from data is the domain of statistical and machine learning [12].

There are different scenarios where it is possible to learn from data. The learning problem can be divided into three categories based on the data set [13]: supervised, reinforcement, and unsupervised learning. In supervised learning, the data contains examples of (input, output) pairs. For example, in loan default prediction we have a data set of past loans with borrower information as input and loan default as output. In reinforcement learning, correct outputs are not necessarily known and the data set contains (input, some output, score) triplets. For example, if we are developing an artificial intelligence for games we may not know the correct action at every situation but we can give a score to each action based on the result. Finally, in unsupervised learning we have only (input)-vectors and we seek to discover structure in the data. For example, we could cluster people into ’personality types’ based on their answers in a questionnaire. Supervised learning is the most common and well-understood machine learning problem. This thesis focuses on supervised learning.

A simple example of a learning problem was given previously, where the goal was to predict customer LTVs from different demographic characteristics. The learning problem is to relate the input (demographic characteristics) to the outputs (LTVs). We do not know how the output is generated from the input and the relationship is probably too complex to implement manually. Even if a human would be able to make good predictions, the automation of the task could bring greatly increased speed and cost-savings. We therefore wish to automate the process. In machine learning, it is assumed that we have a data set of (input, output) examples. This data set then used to build a model of how the outputs relate to inputs, with the goal that outputs are predicted correctly for new inputs. It would be trivial to solve the problem for known examples by predicting the observed outputs for observed inputs. The crucial point is that machine learning has a focus on generalization, which means it seeks to optimize the accuracy of predictions for new inputs. In principle, any kind of black box algorithm could be considered; the primary measure of success is the predictive ability in new data [6].

We now formalize machine learning as an empirical risk minimization problem [14]. Denote an input $x \in \mathcal{X}$ and an output $y \in \mathcal{Y}$. The inputs belong to the set $\mathcal{X}$ that is the possible information about each observation, which is often a real valued vector $\mathcal{X} = \mathbb{R}^d$ of dimension $d$. The outputs belong to the set $\mathcal{Y}$, for example $\mathcal{Y} = \mathbb{R}$ in standard regression and $\mathcal{Y} = \{-1,1\}$ in binary classification. The data set $\mathcal{D} = \big((x_1, y_1), \dots, (x_n, y_n)\big)$ is used to learn a function $f: \mathcal{X} \to \mathcal{Y}$. For any given input $x \in \mathcal{X}$, the function should predict an approximately correct response $f(x) \in \mathcal{Y}$. For learning to be possible, the data set has to have something in common with new inputs. A standard assumption is that the examples $(x_\mathrm{i}, y_\mathrm{i})$ are generated independently from each other based on the same underlying probability distribution $P(x, y)$ on $\mathcal{X} \times \mathcal{Y}$ [15]. To generate an example, we first sample the input $x \in \mathcal{X}$ from the marginal distribution $P(x)$ and then the output $y \in \mathcal{Y}$ from the conditional distribution $P(y|x)$. This definition has two important charasterics of what we assume of the true relationship $P(y|x)$. First, it may be stochastic so that it is impossible to predict the output perfectly. Second, there is no assumption of what the functional form of the distribution looks like, so the goal may be to learn an arbitrary function. To measure how well a function fits the data set, we define a loss function $L: \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ which assigns a real valued loss $L\big(y, f(x)\big)$ based on the difference between true value $y$ and predicted value $f(x)$. The quality of predictions is measured by the expected loss:

$$\mathcal{R}_{L,P}(f) = \int_{\mathcal{X} \times \mathcal{Y}} L\big(y, f(x)\big) dP(x, y)$$

Because a function is defined to be better the smaller expected loss it has, the best approximation is provided by the function that minizes the expected loss:

$$\mathcal{R}^*_{L,P} = \inf_{f:\mathcal{X} \to \mathcal{Y}} \mathcal{R}_{L,P}(f)$$

In the practical setting, based on the data $\mathcal{D}$ we need to choose a function $f_D: \mathcal{X} \to \mathbb{R}$ such that its loss is close to the minimum loss $\mathcal{R}^*_{L,P}$. A learning method is an algorithm that assigns a function to any given data set. The learning method is said to be universally consistent if for any $P$ on $\mathcal{X} \times \mathcal{Y}$ the learning algorithm produces $f_D$ such that it approaches the best possible function as the sample size increases:

$$\mathcal{R}_{L,P}(f_D) \to \mathcal{R}^*_{L,P} \text{ as } n \to \infty$$

Several learning algorithms in machine learning can be shown to be universally consistent [16]. This is an interesting theoretical result, because it shows that many machine learning methods can be guaranteed to deliver asymptotically optimal performance.

In a real-world problem, we do not know the underlying probability distribution $P$. The idea is to learn a model using the data set $\mathcal{D}$, which we assume to be generated by this distribution. The expected loss of a given function can be approximated by the empirical loss in the data set:

$$\mathcal{R}_{L,D}(f) = \frac{1}{n}\sum_{i=1}^{n} L\big(y_i, f(x_i)\big)$$

By the law of large numbers [17], for a fixed function $f$, the empirical loss converges to the expected loss: $\mathcal{R}_{L,D}(f) \to \mathcal{R}_{L,P}(f)$ as $n \to \infty$. Because we cannot direcly minimize the expected loss, it is tempting to choose a function that minimizes the empirical loss $\inf_{f:\mathcal{X}\to\mathcal{Y}} \mathcal{R}_{L,D}(f)$. However, the problem is not so simple. A function that predicts $y_i$ at every $x_i$ and 0 elsewhere achieves this goal. This is called overfitting: a smaller loss in the data set may not translate to a smaller loss outside the data set. To avoid overfitting, one can choose a smaller set of functions $\mathcal{F}$ that contains a reasonably good approximation of the solution and minimize only over it:

$$\mathcal{R}^*_{L,D,\mathcal{F}} = \inf_{f\in\mathcal{F}} \mathcal{R}_{L,D}(f)$$

This approach is called empirical risk minization [12] and it can be used to produce an approximate solution to the infinite sample counterpart:

$$\mathcal{R}^*_{L,P,\mathcal{F}} = \inf_{f\in\mathcal{F}} \mathcal{R}_{L,P}(f)$$

The problem implicit in any learning method is to choose an appropriate function space $\mathcal{F}$. There are two fundamental and completing challenges in the selection. On one hand, we want to consider a limited function space $\mathcal{F}$ so that we do not overfit the data and the loss on data is reflective of the true loss $\mathcal{R}^*_{L,D,\mathcal{F}} \approx \mathcal{R}^*_{L,P,\mathcal{F}}$. If we have very complex functions the model may fit data very well but not generalize outside it. On the other hand, we want to consider an expressive function space $\mathcal{F}$ so that it is possible obtain a small approximation error $\mathcal{R}^*_{L,P,\mathcal{F}} \approx \mathcal{R}^*_{L,P}$. Limiting the choice of functions to simpler $\mathcal{F}$ causes a loss if it does not contain a good approximation to the function that minimizes the expected loss. In practise, it is necessary to obtain a good balance between these two extremes.

One popular approach is to use an expressive function space but have a constraint on the type of functions that are learned. These constraints can be relaxed with more data. In this approach, we define a non-negative functional $\Omega\colon \mathcal{F} \to \mathbb{R}^+$ called the regularizer that penalizes the complexity $\Omega(f)$ of each function $f$ by assigning larger values to more complex functions. The solution minimizes the regularized empirical risk: $\inf_{f \in \mathcal{F}} \mathcal{R}_{L,D}(f)$ subject to $\Omega(f) \leq C$, which is a subset of the original function space $\mathcal{F}$. An equivalent unconstrained problem is [18]:

$$\inf_{f \in \mathcal{F}} \mathcal{R}_{L,D}(f) + \lambda\Omega(f)$$

Assume that we have chosen a function $f \in \mathcal{F}$. How can we measure its expected risk if we do not know the underlying distribution $P(x, y)$? The simplest approach is to divide the data set $\mathcal{D}$ into training data $\mathcal{D}_{\text{train}}$ and test data $\mathcal{D}_{\text{test}}$. The empirical risk in training data is used to find the optimal function $f \in \mathcal{F}$. After the function $f$ is chosen, its empirical risk in the test data is measured to estimate the risk outside the training set. Formally, we have a data set $\mathcal{D} = \big((x_1, y_1), \dots, (x_n, y_n)\big)$. The data indices $I = \{1,2, \dots, n\}$ are divided into mutually disjoint index sets $I_{\text{train}}, I_{\text{test}} \subseteq I$ such that $I_{\text{train}} \cap I_{\text{test}} = \emptyset$ and $I_{\text{train}} \cup I_{\text{test}} = I$. These define a training set $\mathcal{D}_{\text{train}} = \big((x_i, y_i): i \in I_{\text{train}}\big)$ and test set $\mathcal{D}_{\text{test}} = \big((x_i, y_i): i \in I_{\text{test}}\big)$. The optimal function is found by minimizing the empirical risk in the training set $f^* = \inf_{f \in \mathcal{F}} \mathcal{R}_{L,D_{\text{train}}}(f)$. For the resulting function $f^*$, an unbiased estimate of the expected risk is $\mathcal{R}_{L,D_{\text{test}}}(f^*)$. Since the function $f^*$ is fixed, the empirical loss in the test set converges to the expected loss $\mathcal{R}_{L,P}(f^*)$ as $|\mathcal{D}_{\text{test}}| \to \infty$ by the law of large numbers.

Machine learning methods provide practical choices of the function space $\mathcal{F}$. Different methods have different function spaces, and sometimes the definition does not explicitly state a function space. Parametric methods assume that the function $f$ is defined by a vector of parameters $\alpha$. Nonparametric methods do not necessarily make any assumptions about the functional form of $f$, but instead seek to estimate the function directly from data [19]. In this case, the number of parameters is implicit and variable, it can increase with the number of observations. The parametric approach has a potential problem, because the true function does not necessarily match the assumed parametric form. With nonparametric approaches it is possile to use more flexible function spaces that can describe a wider range, or even any, functions. However, more flexible functions can increase the problem of overfitting.

Assume that a function $y = f(x)$ has the output $y \in \mathbb{R}$ and the input $x \in \mathbb{R}^d$. We give two examples of practical function spaces that occupy opposite ends of the function complexity spectrum. The simplest model specifies that the output is a linear function of input in terms of parameters $\alpha \in \mathbb{R}^d$:

$$f_\alpha(x) = \alpha_1 x_1 + \dots + \alpha_d x_d$$

To include a bias term, it is possible to concatenate a constant feature $x_1 = 1$ to the input vector. This defines the function space $\mathcal{F} = \{f_\alpha(x) : \alpha \in \mathbb{R}^d\}$. We can add regularization to improve the predictive ability of the model. This reduces the effect of irrelevant features and can result in even simpler models. There are several popular choices for the penalty term $\Omega(f_\alpha)$ added to the empirical risk [20]:

$$\Omega_0(f_\alpha) = \sum_{i=1}^{d} \mathbb{I}(\alpha_i \neq 0) \text{ (nonzero cofficients)}$$
$$\Omega_1(f_\alpha) = \sum_{i=1}^{d} |\alpha_i| \text{ (least absolute shrinkage)}$$
$$\Omega_2(f_\alpha) = \sum_{i=1}^{d} \|\alpha_i\|^2 \text{ (Tikhonov regularization)}$$

On the other side of the spectrum, kernel methods in machine learning consider highly non-linear function spaces in a non-parametric way [21]. Assume we are given a symmetric and positive definite function $k(x, x')$ over inputs, which is known as a kernel. The reproducing kernel Hilbert space $\mathcal{F}$ associated to $k$ is a function space obtained as the completion of the following function space $\mathcal{F}_0$:

$$\mathcal{F}_0 = \left\{\sum_{i=1}^{N} \alpha_i k(x_i, x) \ : N \in \mathbb{N},\ x_i \in \mathcal{X}, \alpha_i \in \mathbb{R}\right\}$$

where the inner product in $\mathcal{F}_0$ is defined as:

$$\left\langle \sum_{i=1}^{N} \alpha_i k(x_i, x), \sum_{j=1}^{M} \alpha'_j k(x'_j, x) \right\rangle_{\mathcal{F}_0} = \sum_{i=1}^{N} \sum_{j=1}^{M} \alpha_i \alpha'_j k(x_i, x'_j)$$

The inner product defines the model complexity penalty $\Omega(f) = \|f\|_{\mathcal{F}}^2 = \langle f, f \rangle_{\mathcal{F}}$ used in regularized empirical risk minimization. While the kernel method is fundamentally non-parametric, the function $f_\alpha$ that minimizes the empirical risk can be found with parametric optimization methods. This is based on the representer theorem [22], which implies that the function is a linear combination of kernel evaluations at data points $\mathcal{D} = \left((x_{\mathrm{i}}, y_{\mathrm{i}})\right)_{i=1}^{n}$ where $\alpha \in \mathbb{R}^n$:

$$f_\alpha(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x)$$

One choice of the kernel function is the Gaussian radial basis function (RBF), where $\gamma > 0$ is a fixed parameter known as the width of the kernel:

$$k(x, x') = \exp(-\gamma^{-2} \|x - x'\|^2)$$

For the RBF kernel, the resulting function space $\mathcal{F}$ implies universally consistent learning methods in many practical applications [23]. This means any function can be modelled asymptotically, as long as the choice of the regularization parameter $\lambda$ and kernel width $\gamma$ are done in a data-dependent way.

Model interpretability is one reason to prefer simpler models if different models competitive predictive accuracy [24]. The linear model is easily interpretable since each coefficient $\alpha_i$ is the influence of that feature to the prediction. With feature selection or least absolute shrinkage regularization we can potentially discard some features, obtaining even simpler models. Extending the linear model makes for more complex, but in principle still interpretable models. The most flexible machine learning models have multiple parameters with nonlinear effects and are difficult to interpret. These include kernel methods, random forests and neural networks, for example. This is yet another tradeoff; between interpretability and model complexity.

Finally, there may be a technical complication in the choice of the loss function $L$. Not every loss function that defines a empirical loss $\mathcal{R}_{L,D}$ can be easily optimized. For example, it is NP-hard to minimize the empirical risk in terms of accuracy of ‘incorrect’ (1) and ‘correct’ (0) classification [25]. Machine learning uses well-behaved loss functions, meaning those that can be solved with standard optimization methods in a reasonable time. In terms of the actual empirical risk of interest, the loss used in the optimization stage can be seen as a surrogate loss function. Fitting the function is based on the assumption that the surrogate loss approximately minimizes the empirical risk of interest. Two popular loss functions are the ‘squared loss’ for regression ($y \in \mathbb{R}$) and ‘logistic loss’ for classification ($y \in \{-1,1\}$), which we later show to result in standard linear regression and logistic regression models:

$$L_{\text{squared}}\big(y, f(x)\big) = \big(y - f(x)\big)^2$$

$$L_{\text{logistic}}\big(y, f(x)\big) = \log[1 + \exp(-yf(x)]$$

There are several possible optimization algorithms that find the optimal solution for the chosen loss function. If the loss function is smooth, in the sense that the matrix of second derivatives is continuous, it is possible to find the optimal function with Newton’s method, for example [26]. Assume that the function space is parametric. We then find the parameters $\alpha = (\alpha_1, \dots, \alpha_d)$ that minimize the penalized empirical risk $L(\alpha) = \mathcal{R}_{L,D}(f_\alpha) + \lambda\Omega(f_\alpha)$. Denote the $d$ length gradient vector as $L'(\alpha) = \partial L(\alpha)/\partial\alpha$ and the $d \times d$ Hessian matrix as $L''(\alpha) = \partial L(\alpha)/\partial\alpha\partial\alpha^T$. The optimal parameters $\hat{\alpha}$ minimize the empirical risk, so the gradient is zero $L'(\hat{\alpha}) = 0$ at the solution $\hat{\alpha}$. The Taylor series expansion of $L'(\alpha)$ around initial guess $\alpha^{(0)}$ is:

$$L'(\alpha) = L'\big(\alpha^{(0)}\big) + \big(\alpha - \alpha^{(0)}\big)L''\big(\alpha^{(0)}\big) + \big(\alpha - \alpha^{(0)}\big)^2/2!\,L'''\big(\alpha^{(0)}\big) + \cdots = 0$$

If we approximate the function by keeping the first two terms of this expansion, we can solve for α to obtain an estimate of the solution given the initial guess $\alpha^{(0)}$:

$$\alpha \approx \alpha^{(0)} - \big[L''\big(\alpha^{(0)}\big)\big]^{-1}L'\big(\alpha^{(0)}\big)$$

Given the α thus obtained, this can be iterated to obtain increasinly accurate new estimates, which results in the multivariate Newton’s method:

$$\alpha^{(r)} \approx \alpha^{(r-1)} - \big[L''\big(\alpha^{(r-1)}\big)\big]^{-1}L'\big(\alpha^{(r-1)}\big)$$

## 2.2 Statistics

We utilize arguments from both statistics and machine learning in this thesis. Statistics is related to machine learning. Both fit models to data but have a somewhat different emphasis [27] [28]. Statistics typically uses fully specified probabilistic models to explain the observed data as accurately as possible. The models are used to analyze the problem and interpret how different parameters affect the outcome.

This is in contrast to machine learning where the goal is predict as accurately as possible for new data. This perspective takes into account the phenomena of overfitting, and the model in fact can be seen as a black box used to give predictions. However, the most simple machine learning algorithms and standard regression methods in statistics are identical: least squares and logistic regression for example.

Denote the data set $\mathcal{D} = \left((x_1, y_1), \ldots, (x_n, y_n)\right)$, where each $(x_i, y_i)$ is an observation with input $x_i$ and output $y_i$. In the statistical approach, we assume that the data set is sampled from a probabilistic model $P(\mathcal{D}|\mathrm{\alpha})$ where $\alpha = (\alpha_1, \ldots, \alpha_d)$ is a vector of model parameters [29]. Given this model, in maximum likelihood estimation (MLE) we seek parameters that maximize the probability mass or density function of the data. For this reason, define the likelihood function $L(\alpha) = P(\mathcal{D}|\mathrm{\alpha})$ that we seek to maximize. It is again assumed that the observations are generated i.i.d. from the underlying probability distribution, similar to empirical risk minimization, so the likelihood factorizes $L(\alpha) = P(\mathcal{D}|\mathrm{\alpha}) = \prod_{i=1}^{n} P(y_i|x_i, \alpha)$. The maximum likelihood estimate is the parameter vector $\hat{\alpha}$ that maximizes the likelihood function $\hat{\alpha} = \operatorname{argmax}_{\alpha} L(\alpha)$ [30].

Practical algorithms typically use the log-likelihood $l(\alpha) = \log\left(L(\alpha)\right)$, because as a monotonic function it has the same solution with better numercial properties. To fit a model, we first hypothesize a parametric probability model $P(y|x, \alpha)$ and then find the MLE estimate for the parameter vector $\alpha = (\alpha_1, \ldots, \alpha_d)$ by maximizing the log-likelihood $l(\alpha)$. Denote the $d$ element gradient vector of the log-likelihood as $l'(\mathrm{\alpha}) = \partial l(\alpha)/\partial\alpha$ and the $d \times d$ Hessian matrix as $l''(\mathrm{\alpha}) = \partial l(\alpha)/\partial\alpha\partial\alpha^T$. Since the MLE parameters $\hat{\alpha}$ maximize the log-likelihood, the gradient is zero $l'(\hat{\alpha}) = 0$ at the solution $\hat{\alpha}$. We can again use the Newton's method, for example. Given an initial guess $\alpha^{(0)}$, repeat the iteration: $\mathrm{\alpha}^{(r)} \approx \alpha^{(r-1)} - \left[l''\left(\alpha^{(r-1)}\right)\right]^{-1} l'\left(\alpha^{(r-1)}\right)$ until convergence. This can be seen as a special case of empirical risk minimization, where the loss function is defined as the negative log-likelihood.

Maximum likelihood estimation has important mathematical guarantees, and asymptotic results about the parameter estimates can be used to justify its widespread application. Assuming a correct probabilistic model for $P(y|x, \alpha)$, if $\hat{\alpha}$ is the MLE and $\alpha$ is the true parameter vector that generated the data, as the sample size increases $n \to \infty$ we have asymptotically [30]:

1. The parameter estimates are consistent: they converge in probability to the true value $P(\|\hat{\alpha} - \alpha\| > \epsilon) \to 0$.
2. The parameter estimates are functionally invariant: $z(\hat{\alpha})$ is the MLE of $z(\alpha)$ for any function $z$.
3. The parameter estimates are efficient: no consistent estimator has lower variance.

Statistics typically considers confidence intervals and p-values of the parameters. Parameter inference in straightforward in correctly specified models [31]. Define the observed information matrix as the negative Hessian of the log-likelihood: $I(\alpha) = -l''(\mathrm{\alpha})$. The expected information matrix is the expectation $\mathcal{J}(\alpha) = \mathbb{E}[I(\alpha)]$. The MLE $\hat{\alpha}$ is asymptotically normally distributed with expected value $\alpha$ and covariance matrix is the inverse of expected information:

$$\hat{\alpha} - \alpha \sim MVN(0, \mathcal{J}(\alpha)^{-1})$$

For the purposes of statistical inference, $\alpha$ is not known and $\mathcal{J}(\alpha)$ can be replaced asymptotically by $\mathcal{J}(\hat{\alpha})$ or $\mathrm{I}(\hat{\alpha})$. This result is used to construct confidence intervals and p-values of the parameters, for example [32].

We now present two simple models from both a statistical and machine learning perspective [33]. We derive the methods from the statistical perspective and show how they correspond to an empirical risk minimization problem in machine learning. We present least squares regression as a regression method and logistic regression as a classification method. The statistical approach is motivated by specifying a parametric model $P(y|x, \alpha)$ that describes a probability model of the data. From a statistical viewpoint, we find the model parameters $\alpha$ as the maximum likelihood estimates. These are shown to equal empirical risk minimization problems in machine learning.

Least squares regression is a standard approach to regression. A straightforward derivation from simple probabilistic assumptions results in the model. Assume that data consists of outputs $y \in \mathbb{R}$ and inputs $x \in \mathbb{R}^d$. The output $y = f_\alpha(x)$ is a function of input $x$, where the function $f$ is specified by a parameter vector $\alpha$. We assume that the output also includes a stochastic noise term $\epsilon \sim \mathcal{N}(0, \sigma^2)$ that has constant variance $\sigma^2$:

$$y = f_\alpha(x) + \epsilon$$

Given input $x$ and noise $\epsilon$, this implies that the output has a Gaussian density:

$$P(y|x, \alpha) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y - f_\alpha(x)\right)^2}{2\sigma^2}\right)$$

Assuming that the observations are independent:

$$l(\alpha) = \sum_{i=1}^{n} \log[P(y_i|x_i, \alpha)] = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - f_\alpha(x))^2$$

From a statistical perspective we maximize the log-likelihood $\hat{\alpha} = \underset{\alpha}{\operatorname{argmax}}\, l(\alpha)$ to find the parameters. We could equivalently minimize the negative log-likelihood $\hat{\alpha} = \underset{\alpha}{\operatorname{argmin}} -cl(\alpha)$ multiplied by a constant. Taking $c = 1/n$ this corresponds to an empirical risk minimization with a squared loss $L\big(y_i, f_\alpha(x_i)\big) = \big(y_i - f_\alpha(x_i)\big)^2$:

$$\hat{\alpha} = \underset{\alpha \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \big(y_i - f_\alpha(x)\big)^2$$

In the training phase, we fit the model to the data set using this loss to find $\hat{\alpha}$. In the test phase, we predict values simply by $y = f_{\hat{\alpha}}(x)$.

Logistic regression is a standard approach to binary and multiclass classification. We show how simple assumptions result in the model . Assume that data consists of binary outputs $y \in \{-1,1\}$ and inputs $x \in \mathbb{R}^d$. Logistics regression models the log-odds of the output $y$ as a function $f_\alpha(x)$ of input $x$, where the function $f$ is specified by a parameter vector $\alpha$:

$$\log\left[\frac{P(y=1|x,\alpha)}{1-P(y=1|x,\alpha)}\right] = f_\alpha(x)$$

This implies that the probability of the binary response $y$ is given by:

$$P(y = 1|x,\alpha) = 1/(1 + \exp\big(-f_\alpha(x)\big)$$
$$P(y = -1|x,\alpha) = 1/(1 + \exp\big(f_\alpha(x)\big)$$

Assuming the observations are independent:

$$l(\alpha) = \sum_{i=1}^{n} \log[P(y_i|x_i,\alpha)] = -\sum_{i=1}^{n} \log\big[1 + \exp\big(-y_i f_\alpha(x)\big)\big]$$

From a statistical perspective we maximize the log-likelihood $\hat{\alpha} = \operatorname{argmax}_{\alpha} l(\alpha)$. We can again minimize the negative log-likelihood $\hat{\alpha} = \operatorname{argmin}_{\alpha} -cl(\alpha)$ multiplied by a constant. Taking $c = 1/n$ this corresponds to an empirical risk minimization with a logistic loss $L\big(y_i, f_\alpha(x_i)\big) = \log[1 + \exp(-y_i f_\alpha(x_i)]$:

$$\hat{\alpha} = \underset{\alpha \in \mathbb{R}^d}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \log[1 + \exp(-y_i f_\alpha(x_i)]$$

In the training phase, we fit the model to the data set using this loss to find $\hat{\alpha}$. In the test phase, we predict the probability of the outcome simply by $P(\mathrm{y} = 1|\mathrm{x}) = 1/(1 + \exp(-f_{\hat{\alpha}}(x))$. If predictions are required on a binary scale, we can convert these probabilities into $-1$ or 1 by using a threshold of 0.50, for example.

The machine learning perspective does not require a full probabilistic model of the data, merely that we have defined an empirical risk that we use to fit an arbitrary function $f_\alpha(x)$ to data. It is therefore not necessary to assume that the model is a correct probabilistic description of the phenomena in order to make predictions. For our goal, we fit the methods using these loss functions and say that the method is as good as the predictive ability on new samples. In addition, one can add regularization to further improve the predictive accuracy.

## 2.3 Overfitting example

We now present a simple example of a learning problem to illustrate the function space complexity tradeoff [34]. Let the true function be a third degree polynomial $y = -0.1 + 0.4x + 0.7x^2 - 0.2x^3 + \epsilon$ which includes a noise term $\epsilon \sim \mathcal{N}(0{,}0.5)$. The inputs are sampled uniformly $P(x) \sim \mathrm{Unif}(-1{,}1)$. The loss function is the squared error $L\big(y_i, f(x_i)\big) = \big(y_i - f(x_i)\big)^2$, which is a standard least squares problem in empirical risk minimization: find a function $f$ such that $\mathcal{R}_{L,D}(f) = \frac{1}{n}\sum_{i=1}^{n}\big(y_i - f(x_i)\big)^2$ is smallest. Because in this example we know the true function, it is possible to explicitly measure the expected loss $\mathcal{R}_{L,P}(f)$ for different functions. Higher degree polynomials are known to be particularly prone to overfitting [35], even though increasing the order of the polynomial achieves better fits to the data set. Consider three simple function spaces: linear functions $\mathcal{F}_1$, quadratic functions $\mathcal{F}_2$, and 5'th degree polynomials $\mathcal{F}_5$. The resulting best fits are displayed in **Figure 3**. We see that the simplest function underfits the target ($\mathcal{R}_{L,D}(f) = 0.17$, $\mathcal{R}_{L,P}(f) = 0.37$). The quadratic funtion fits both data and target ($\mathcal{R}_{L,D}(f) = 0.11$, $\mathcal{R}_{L,P}(f) = 0.27$). The 5'th degree polynomial suffers from overfitting: it provides the best approximation to the data in terms of squared error, but has a larger error relative to the true target ($\mathcal{R}_{L,D}(f) = 0.06$, $\mathcal{R}_{L,P}(f) = 0.35$).

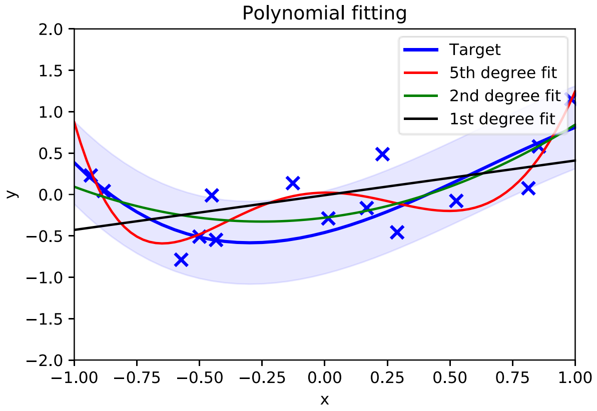


**Figure 3.** 1st, 2nd or 5th degree polynomials are fit to a random 3rd degree target, and it is clear that the 1st degree polynomial underfits and the 5th degree overfits.

The trade off between complex and simple functions can be explained with the bias-variance tradeoff. In the special case of a squared loss and a noisy target $y = g(x) + \epsilon$, it can be analyzed analytically with a simple formula. In the example we used a single realization of the data set. Assume now that the data set $D$ is random and we use the learning algorithm to get a function $f^{(D)}$ based on it. The average function we learn is denoted by $\bar{f}(x) = \mathbb{E}_{\mathcal{D}}\left[f^{(\mathcal{D})}(x)\right]$. Then the expected loss $\mathcal{R}_{L,P}\left(f^{(D)}\right) = \mathbb{E}_{x,y}\left[\left(y - f^{(D)}(x)\right)^2\right]$ can then be written over different realizations of the data as [36]:

$$\mathbb{E}_{\mathcal{D}}\left[\mathcal{R}_{L,P}\left(f^{(d)}\right)\right] = \mathbb{E}_x[\mathrm{bias}(x)] + \mathbb{E}_x[\mathrm{var}(x)] + \mathrm{var}(\epsilon)$$

where $\mathrm{bias}(x) = \mathbb{E}_y\left[\left(g(x) - \bar{f}(x)\right)^2\right]$ and $\mathrm{var}(x) = \mathbb{E}_{\mathcal{D}}\left[\left(f^{(\mathcal{D})} - \bar{f}(x)\right)^2\right]$. The bias term measures how well the average function approximates the true function, and we obtain a smaller bias if we use more complex functions that can approximate the true function better. The variance term measures how much the function would change if we used a different data set, and we obtain a smaller variance if we use more simple functions that do not vary too much with different realizations of the data. The noise term represents the stochastic noise implicit in the problem, and corresponds to the loss of the best possible function. We demonstrate the bias and the variance of the function spaces $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_5$ in **Figure 4**. It can be seen that function complexity decreases bias but increases variance. Because the final error is a sum of these terms, there exists a trade-off between optimal function complexity.

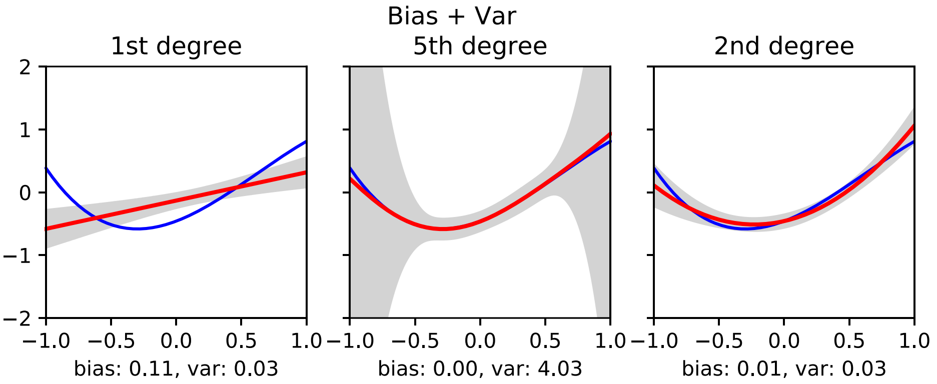


**Figure 4.** Relative to the true target (blue), over many iterations the 2nd degree polynomial has the best trade-off in bias (red line) and variance (shaded grey).

Regularization is based on the idea of using a function space with more complex functions and constraining it based on the data. Define a function space of d'th degree polynomials in terms of Legendre basis $\mathcal{F}^{(d)} = \left\{\sum_{i=0}^{d} \alpha_i L_i(x) : \bar{a} \in \mathbb{R}^{d+1}\right\}$ where $L_i(x)$ is the i'th Legendre polynomial. Examples of Legendre polynomials are shown in **Figure 5**, where it can be seen that the complexity of the learned function goes up with additional basis functions. The purpose of using Legendre polynomials is that the functions form an orthogonal basis. If we consider up to 5'th degree polynomials, any function can written as $f(x) = \alpha_0 + \alpha_1 L_1(x) + \cdots + \alpha_5 L_5(x)$. Furthermore, if the complexity penalty is a standard L2 norm $\|f\|^2 = \int_{-1}^{1} f(x)^2 dx$ , the regularization term is simply $\Omega(f) = \alpha_0^2 + \alpha_1^2 + \cdots + \alpha_5^2$. To obtain a simple function with a good generalization ability, we regularize this solution.

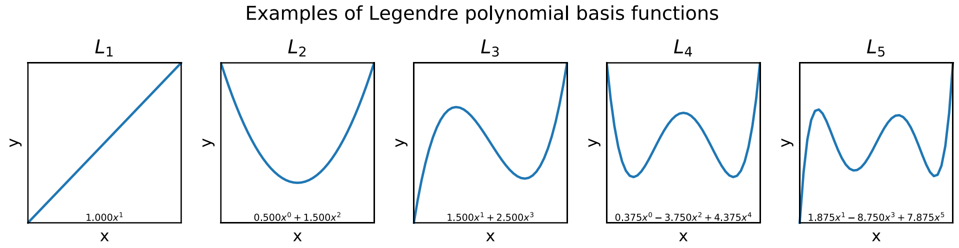


**Figure 5.** Visualization of the first five non-trivial Legendre.polynomial basis functions.

Regularization constrains the solutions to simpler polynomials, where the coefficients are bounded. We now find the function $f$ that minimizes the regularized empirical risk $\mathcal{R}_{L,D}(f) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i)\right)^2 + \lambda\Omega(f)$. Results for different choices of $\lambda$ are illustrated in **Figure 6**. It can be seen that the function space becomes simpler as the penalty increases: the solution starts at an unconstrained 5'th degree polynomial and eventually approaches a contant function. The expected loss for $\lambda = 0.2$ is similar to the second degree polynomial. For lower penalty the solution overfits, where as for larger penalties the solution is too simple. This means that it is not necessary to make a priori choice on the degree of the polynomials considered, if the regularization parameter is used to control the function complexity in a data-dependent way.

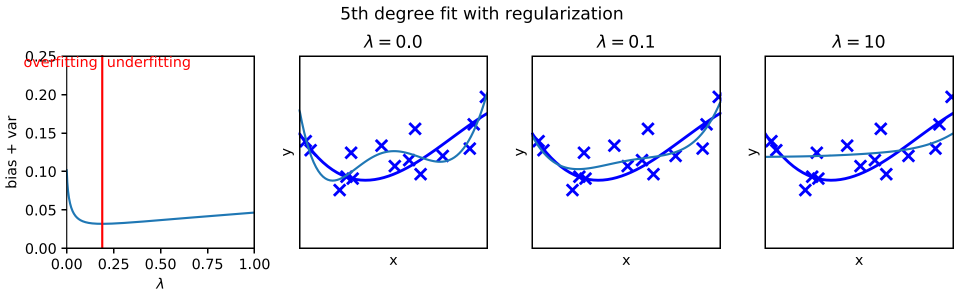


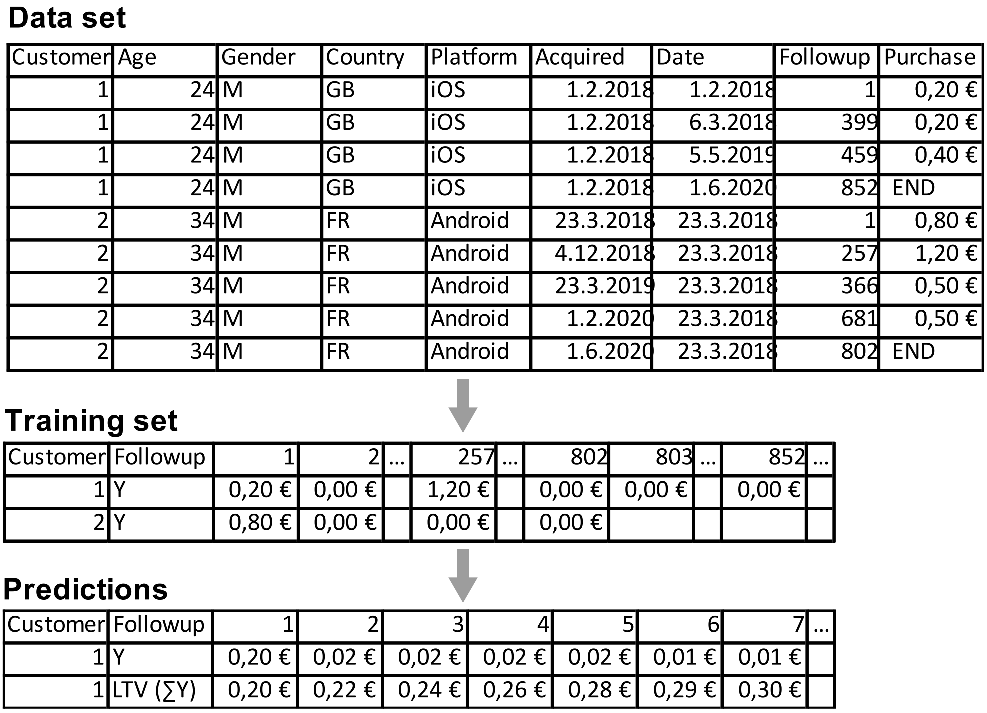
**Figure 6.** Regularization can be used to constrain the complexity of the learned function: the optimal regularization parameter occurs in the neighbourhood of 0.20.

## 2.4 Machine Learning for censored data

### 2.4.1 Definition as pairwise data

Censoring has been studied especially in survival analysis, which is a field of statistics that models the time to an event that may be censored. The outcome is the lifetime of a subject, or the total follow-up time if censored. In a more general formulation we can have more than two states ("alive", "dead"), which is known as life history analysis [37]. Life history analysis is a common mathematical framework to analyze censored data as a stochastic process, and has so far found applications in biostatistics, reliability engineering, actuarial science, demography, epidemiology, to name a few. Statistical analysis has usually focused on nonparametric methods for multiple subjects. There is a long history of methods for nonparametric survival curve analysis in the case of a single event [38] and nonparametric analysis of the cumulative intensity in the case of recurrent events [39]. Based on the underlying stochastic process, these methods have been extended to multiple event types [40] and transitions between different states [41]. Several statistical tests have been developed to assess treatment effects [42]. A popular regression model by Cox was developed to assess survival when subjects have covariates [43], which can also be generalized to recurrent events [44]. Reliability engineering in contrast has focused on parametric models, sometimes of a single process, as summarized in the review by Lawless [45]. This allows the prediction of future events in maintenance planning, for example. There has been increasing awareness that the data may not follow the simplifying stochastic process assumptions in probabilistic modelling, typically assuming that future events are independent of the past events to analytically model the data as a Poisson process, so simpler and more robust methods of analysis have been presented [46] [47]. Real-valued outcomes have been considered together with the stochastic process, for example in the analysis of medical costs [48] [49] [50]. Some of the nonparametric estimators in fact generalize directly to cumulative costs [51], and there are simple and robust ways to consider regression as well [52] [53].

Consider the previous example of a company that measures the LTV of customers. It is typically the case that some customers are relatively new and others have been with the company for some time. The new customers have had less time to make purchases, so they have smaller LTVs compared to older customers. This may not be because they are worse customers per se, just that they have not been followed for equally long. What the company really wants to find out is the LTV over time that results when the customer is followed until they eventually stop being customers of the company. The collected data set is censored with respect to the final LTV. This means we cannot directly use the measured LTVs, but need a smart way to take into account different follow-up times.



**Data set**

| Customer | Age | Gender | Country | Platform | Acquired | Date | Followup | Purchase |
|---|---|---|---|---|---|---|---|---|
| 1 | 24 | M | GB | iOS | 1.2.2018 | 1.2.2018 | 1 | 0,20 € |
| 1 | 24 | M | GB | iOS | 1.2.2018 | 6.3.2018 | 399 | 0,20 € |
| 1 | 24 | M | GB | iOS | 1.2.2018 | 5.5.2019 | 459 | 0,40 € |
| 1 | 24 | M | GB | iOS | 1.2.2018 | 1.6.2020 | 852 | END |
| 2 | 34 | M | FR | Android | 23.3.2018 | 23.3.2018 | 1 | 0,80 € |
| 2 | 34 | M | FR | Android | 4.12.2018 | 23.3.2018 | 257 | 1,20 € |
| 2 | 34 | M | FR | Android | 23.3.2019 | 23.3.2018 | 366 | 0,50 € |
| 2 | 34 | M | FR | Android | 1.2.2020 | 23.3.2018 | 681 | 0,50 € |
| 2 | 34 | M | FR | Android | 1.6.2020 | 23.3.2018 | 802 | END |

**Training set**

| Customer | Followup | 1 | 2 | … | 257 | … | 802 | 803 | … | 852 | … |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Y | 0,20 € | 0,00 € | | 1,20 € | | 0,00 € | 0,00 € | | 0,00 € | |
| 2 | Y | 0,80 € | 0,00 € | | 0,00 € | | 0,00 € | | | | |

**Predictions**

| Customer | Followup | 1 | 2 | 3 | 4 | 5 | 6 | 7 | … |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Y | 0,20 € | 0,02 € | 0,02 € | 0,02 € | 0,02 € | 0,01 € | 0,01 € | |
| 1 | LTV (∑Y) | 0,20 € | 0,22 € | 0,24 € | 0,26 € | 0,28 € | 0,29 € | 0,30 € | |

**Figure 7.** Three stage solution to censoring: 1) deconstruct data as (customer, follow-up)-pairs 2) predict for every pair 3) reconstruct LTV from predictions for the pairs.

In this thesis, we present four application domains that have to deal with censored data. Every model is developed independently, but all models share a similar solution to censoring. The solution is based on reformulating the data set as pairwise data and consists of three stages: 1) deconstruct the problem as pairwise data 2) use machine learning to get predictions for every pair 3) reconstruct the solution from the pairwise predictions. The idea is illustrated in **Figure 7** for the customer LTV problem in an abstract level. Instead of directly measuring the LTVs of customers by summing the purchases together, we can consider the purchase amount as at every followup time. The followup is the days that have passed since the customer was acquired. The data is then formulated as (‘customer’, ‘followup’)-pairs with ‘purchase’ as the output. The followups that were not observed yet are missing observations. For example, Customer ‘1’ makes their first 0.20€ purchase at followup 1, the second 0.20€ purchase at followup 399, etc., until the data was gathered at followup 852. This means we obtain the following (customer, followup) pairs: (1,1) with value 0.20€ , (1, 2) with value 0.00€, …, (1,852) with value 0.00€. We fit the model to this data set and then predict the purchase at every (customer, followup)-pair. To reconstruct the LTV, we add the predicted purchases at every customer and followup pair together to obtain the predicted LTV of the customer.

Formally, the problem is analyzed as follows. We say that the data set consists of individual and time pairs. Denote individual $i \in \{1, \dots, n\}$ and time $t \in \{1, \dots, m\}$. Individuals have features $x_i \in \mathcal{X}$and time points have features $z_t \in \mathcal{Z}$. If the data set was not censored, it would consists of input $x_{i,t} = (x_i, z_t) \in \mathcal{X} \times \mathcal{Z}$ and output $y_{i,t} \in \mathcal{Y}$ for every individual and time point pair. However, we may not have observed every pair. We therefore define observation indexes $k \in \{1, \dots, N\}$ and two mappings: $I(k)$ maps the observations index to individual index and $T(k)$ maps the observation index to time index. The data set is $\mathcal{D} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right), y_k\right)\right)_{k=1}^{N}$. Pairwise data is an important field of machine learning research [54] and in this form it has been considered in longitudinal analysis in statistics [55]. In this thesis the pairwise data is an abstraction that is unifies the different models. Each of our models is designed specifically for each task is question. General machine learning models for pairwise data and parametric statistical models for longitudinal data can be quite different from our models.

This pairwise abstraction is a special case of the empirical risk minimization framework presented earlier, where every input has a pairwise structure. The input space is now just a cartesian product $\mathcal{X} \times \mathcal{Z}$ and the output space is $\mathcal{Y}$. The learning goal is to find a function $f: \mathcal{X} \times \mathcal{Z} \to \mathcal{Y}$ that predicts the correct output of every pair. However, while the abstraction is just a special case of empirical risk minimization there are important practical differences that need to be taken into account when compared to the standard formulation. The special structure of pairwise data has three major implications that need to be considered in the applications:

1. The model may need to be formulated in terms of pairwise data to model the phenomena correctly [56].
2. The validation needs to be designed for pairwise data where the observations are not independent [57].
3. The statistical theory needs to take into accont that the observations are not indepedent [55].

We go through each of these in the following subchapters.

### 2.4.2 Models for pairwise data

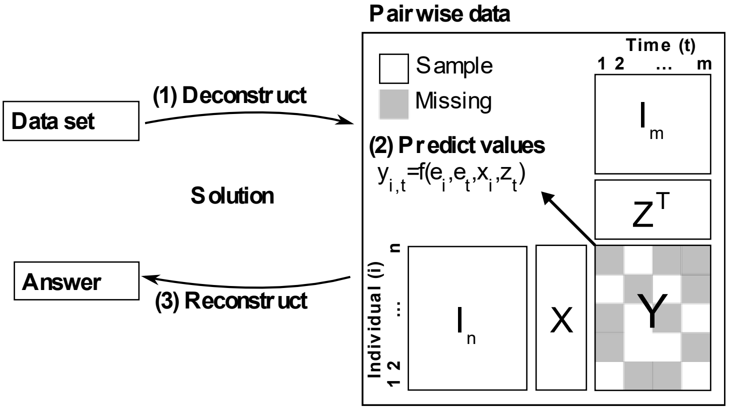


**Figure 8.** Each problem in this thesis can be formulated as a pairwise model applied to censored data with (user, time)-pairs, where we have user identifier ($I_n$) and features (X), time identifier ($I_m$) and features (Z), and the outcome (Y) for the pair.

Designing a model that is an accurate representation of data is very important. To describe the special structure in pairwise data we need either lots of data and very flexible models, or simple models designed for the particular task. We present many simple models that are designed specifically for pairwise data in the studies. We now recapitulate the models that will be presented in Section 3 and demonstrate that in the abstract level they can be seen as instances of a single problem

We briefly illustrate the different models by defining them as special cases of an abstract model that considers individual and time pairs. We invented this abstraction for the purpose of this thesis introduction, but was not obvious at the time of writing the individual studies. Consider the following one-hot encoding of the categorical feature that corresponds to the particular user and time. Let $e_i = \big(\mathbb{I}(\text{individual} = 1), \dots, \mathbb{I}(\text{individual} = n)\big)$ denote a binary vector that indicates the individual and $e_t = \big(\mathbb{I}(\text{time} = 1), \dots, \mathbb{I}(\text{time} = m)\big)$ denote a binary vector that indicates the time. Again, let $x_i \in \mathbb{R}^r$ denote the individual $i$ feature vector and $z_t \in \mathbb{R}^s$ the time $t$ feature vector. If individual features can change over time we denote them by $x_{i,t}$. The outer product of feature vectors $x_i * z_t = (x_1 z_1, \dots, x_1 z_s, \dots, x_r z_1, \dots, x_r z_s) \in \mathbb{R}^{rs}$ denotes interactions of features in vectors $x_i$ and $z_t$. The output for individual $i$ and time $t$ is denoted by $y_{i,t}$. The parameter vector is denoted by either $\alpha$ or $\beta$, whose dimension is defined by the number of features in question. We illustrate the pairwise model paradigm in **Figure 8**: the user indicator is an identity matrix $I_n$, the user features are a matrix $X \in \mathbb{R}^{n \times r}$ , the time indicator is an identity matrix $I_m$, the time features are a matrix $Z \in \mathbb{R}^{m \times s}$, and finally the outputs are a matrix $Y \in \mathbb{R}^{n \times m}$ where some entries can be missing.

In the first application, we predict the expected retention or monetization metrics. The model can be used for any censored metric which is calculated as a sum of observations that occur over time, and the observation $y_{i,t} \in \mathbb{R}^+$ of individual *i* at time *t* is predicted by $\mu_{i,t} = \mathbb{E}\left[y_{i,t}\right]$ where the model is:

$$\mu_{i,t} = \exp\left((e_t\ x_i) \cdot \alpha\right)$$

The second application considers the profit of peer-to-peer loans at the level of a single loan. Individual monthly payments are predicted to calculate profit with discounted cashflow (DCF) analysis. Two complementary models are used for this purpose together to predict the payments. The first model predicts the fixed monthly payments that are obtained according to the schedule, by modelling the default probability $y_{i,t} \in \{-1,1\}$ of loan *i* at time *t*. The prediction $\mu_{i,t} = \mathbb{E}\left[y_{i,t}\right] = P\left(y_{i,t} = 1\right)$ is known as the default rate. If a loan has defaulted, the second model predicts the monthly recovery payments $y'_{i,t} \in \mathbb{R}^+$ of loan *i* at time *t* as $\mu'_{i,t} = \mathbb{E}\left[y'_{i,t}\right]$. These models are defined as:

$$\mu_{i,t}/(1 - \mu_{i,t}) = \exp\left((e_t\ x_i) \cdot \alpha\right)$$
$$\mu'_{i,t} = \exp\left((e_t\ x_i) \cdot \beta\right)$$

The third application is a stochastic process model of individual’s unemployment history $\left(y_{i,t}\right)_{t \geq 0}$, where $y_{i,t} \in \{-1,1\}$ is the unemployment status. We assume that individual’s unemployment status is a Markov chain with individual specific transition rates. These two transition rates are predicted. The first parameter is the probability to exit unemployment $\mu_{i,t} = P\left(y_{i,t} = -1 | y_{i,t} = 1\right)$ and the second parameter is the probability to enter unemployment $\mu^*_{i,t} = P\left(y_{i,t} = 1 | y_{i,t} = -1\right)$. This is done with the following models:

$$\mu_{i,t}/(1 - \mu_{i,t}) = \exp\left(\left(e_i\ e_t\ x_{i,t}\right) \cdot \alpha\right)$$
$$\mu^*_{i,t}/(1 - \mu^*_{i,t}) = \exp\left(\left(e_i\ e_t\ x_{i,t}\right) \cdot \beta\right)$$

The fourth application considers game likes $y_{i,t} \in \{-1,1\}$ of user *i* and game *t*. The prediction is given as the expected like status $\mu_{i,t} = \mathbb{E}\left[y_{i,t}\right]$. We tested four new models. The first model predicts the game likes without considering any features for the user or the game. This is based on the multivariate normal distribution (MVN) $\mathcal{N}(\mu, \Sigma)$ model for the vectors of game likes $\left\{\left(y_{i,1}, \dots, y_{i,m}\right)\right\}_{i=1}^{n}$ for every user $i$. The model is fitted by computing mean $\mu$ and covariance $\Sigma$ of the game like vector. The mean vector is given by $\mu_t = \frac{1}{n}\sum_{i=1}^{n} y_{i,t}$ and the covariance matrix by $\Sigma_{t,u} = \frac{1}{n-1}\sum_{i=1}^{n}\left(y_{i,t} - \mu_t\right)\left(y_{i,u} - \mu_u\right)$. Assume the game likes are binary $y_{i,t} \in \{0,1\}$ and denote the liked games $\mathcal{L}_i = \left\{t : y_{i,t} = 1\right\}$ We predict the ranking of not yet liked games by the conditional expectation given the liked games:

$$\mu_{i,t} = \mathbb{E}\left[y_{i,t} | \{y_{i,t}\}_{t \in \mathcal{L}_i}, \mu, \Sigma\right]$$

Three remaining models use game features, user features, or both game and user features to predict the game likes:

$$\mu_{i,t} = (e_i * z_t) \cdot \alpha$$
$$\mu_{i,t} = (x_i * e_t) \cdot \alpha$$
$$\mu_{i,t} = (x_i * z_t) \cdot \alpha$$

There is no fundamental restriction why the models for censored data would have to be linear or log-linear, as was defined in the above models. The modelling stage could consider non-linear models in the same framework, but the linear models had a good accuracy and were easier to interpret from a statistical perspective. This can be very important in some applications [58]. These simple models include a type of nonparametric estimate, because the influence of time is never assumed to take a particular functional form since we estimate separate intercepts at each time point.

### 2.4.3 Validation for pairwise data

Standard validation procedures of machine learning models are based on the assumption that observations are generated independently and identically from the same distribution. This assumption is probably not valid for pairwise data, because there can be correlation within observations that belong to the same individual or time. For example, if one customer has been observed to make larger purchases at one point, they are probably more likely to make larger purchases at the following time points. Similarly, it is typical that customers make purchases and they then slowly churn out, so that the initial time points have some purchases and the later ones no purchases for a given customer.

Machine learning can be used to learn models from a pairwise data set, in the simplest case by ignoring the pairwise structure, but the validation has to be more complicated. The validation needs to take into account that many observations come from the same individual or time point, or otherwise the correlations may lead to positive bias in how well the model performs. When the model is implemented in the real world setting, it may have no data about a given individual or time point. The validation needs to reflect this fact. For example, if we predict the price of a stock tomorrow, but know the price both today and two days later, it can be quite easy to give a correct answers. The validation that corresponds to correct evaluation for the stock price time series data is not a simple train test set split, but a split where all future observations belong to the test set.

We use a similar idea in this thesis, where we design validation methods that aim to measure the true prediction accuracy in different prediction settings: for example we could measure how well the model predicts for new individuals or new time points, or both simultaneously. For example, in P2P loan prediction we can predict the profit in a new loan that we are thinking of investing in, or we can predict the future profit in an existing loan which someone is offering to sell to us. We do not assume a particular probabilistic model for the data, but instead formulate a way to define a test set that provides an unbiased performance estimate in each setting for any model [57]. For this reason, we define a total of four different test settings:

1. Setting 1: predict for observed individuals and observed times
2. Setting 2: predict for observed individuals and new times.
3. Setting 3: predict for new individuals and known times.
4. Setting 4: predict for new individuals and new times.

In Setting 1, we split the data set into training and test sets as before. The observation indices $k \in \{1, \dots, n\} = I$ are split into mutually disjoint sets $I_{\text{train}}, I_{\text{test}} \subseteq I$, which define a training set and test set

$$\mathcal{D}_{\text{train}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : k \in I_{\text{train}}\right)\right)_{k=1,\dots,N}$$
$$\mathcal{D}_{\text{test}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : k \in I_{\text{test}}\right)\right)_{k=1,\dots,N}$$

In setting 2, we split the time indices $t \in \{1, \dots, s\} = T$ into mutually disjoint index sets $T_{\text{train}}, T_{\text{test}} \subseteq T$, which define a training set and test set

$$\mathcal{D}_{\text{train}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : T(k) \in T_{\text{train}}\right)\right)_{k=1,\dots,N}$$
$$\mathcal{D}_{\text{test}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : T(k) \in T_{\text{test}}\right)\right)_{k=1,\dots,N}$$

In setting 3, we in turn split the individual indices $i \in \{1, \dots, r\} = U$ into mutually disjoint index sets $U_{\text{train}}, U_{\text{test}} \subseteq U$, which define a training and test set

$$\mathcal{D}_{\text{train}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : I(k) \in U_{\text{train}}\right)\right)_{k=1,\dots,N}$$
$$\mathcal{D}_{\text{test}} = \left(\left(\left(x_{I(k)}, z_{T(k)}\right),\ y_k : I(k) \in U_{\text{test}}\right)\right)_{k=1,\dots,N}$$

In setting 4, we split both the time indices $t \in \{1, \dots, s\} = T$ into mutually disjoint index sets $T_{\text{train}}, T_{\text{test}} \subseteq T$ and the individual indices $i \in \{1, \dots, r\} = U$ into mutually disjoint index sets $U_{\text{train}}, U_{\text{test}} \subseteq U$, which define a training and test set

$$\mathcal{D}_{\text{train}} = \Big(\big((x_{I(k)}, z_{T(k)}),\ y_k : I(k) \in U_{\text{train}}, T(k) \in T_{\text{train}}\big)\Big)_{k=1,\dots,N}$$
$$\mathcal{D}_{\text{test}} = \Big(\big((x_{I(k)}, z_{T(k)}),\ y_k : I(k) \in U_{\text{test}}, T(k) \in T_{\text{test}}\big)\Big)_{k=1,\dots,N}$$

### 2.4.4 Statistical inference for pairwise data

Parameter inference in the standard statistical paradigm assumes that the observations are generated independently from a probability density or mass function $P(y|x,\alpha)$. This implied the factorization of the likelihood $L(\alpha) = P(\mathcal{D}|\alpha) = \prod_{i=1}^{n} P(y_i|x_i,\alpha)$. However, the probability model $P(y|x,\alpha)$ may not be correct. In fact, if many observations belong to the same individual or time there is probably correlation and the observations are not independent. There are two solutions to this problem: either define a more complicated model that explicitly specifies how the observations are correlated, or fit a simpler model and aim to do inference even though the model does not claim to represent the probability distribution of the data set. We focus on the second case, and then the function $l(\alpha) = \log\big(L(\alpha)\big)$ is not a true log-likelihood function. However, even if the probability model is misspecified it still defines a function to maximize, or a corresponding empirical risk to minimize. We again seek parameters $\hat{\alpha}$ that maximize the function $l(\alpha)$, at which point the gradient is zero $l'(\hat{\alpha}) = 0$. Such functions are called estimating equations, since they provide estimates of model parameters but are not necessarily based on a likelihood [59].

Define the estimating equation $U_i$ for observation $i$ and $d$ model parameters:

$$U_i(y_i, x_i|\alpha) = \big(U_{i1}(y_i, x_i|\alpha), \dots, U_{id}(y_i, x_i|\alpha)\big)$$

The parameter estimates $\alpha$ are obtained by solving the following:

$$U(\alpha) = \textstyle\sum_{i=1}^{n} U_i(y_i, x_i|\alpha) = 0$$

Assume that the observations in $\mathcal{D}$ are generated independently according to the true distribution $G(y|x,\alpha)$, and there is a unique parameter vector $\alpha^*$ for which the expected value of the estimating equation is zero $\mathbb{E}_G[U(y_i, x_i|\alpha^*)] = 0$. Under certain regularity conditions, it can be shown that the estimating equation yields consistent and asymptotically normal estimates $\hat{\alpha}$ of $\alpha^*$. To state this result, we define the following matrices: $A_n(\alpha) = -\frac{1}{n}\frac{\partial U(\alpha)}{\partial \alpha^T}$, $A(\alpha) = \lim_{n\to\infty} \mathbb{E}_G[A_n(\alpha)]$, $B_n(\alpha) = -\frac{1}{n}\sum_{i=1}^{n} U_i(\alpha)U_i(\alpha)^T$, and $B(\alpha) = \lim_{n\to\infty} \mathbb{E}_G[B_n(\alpha)]$. The matrix $C_n(\alpha) = A_n(\alpha)^{-1}B_n(\alpha)A_n(\alpha)^{-1}$ has the limit $C(\alpha) = A(\alpha)^{-1}B(\alpha)A(\alpha)$ as $n \to \infty$, and:

$$n^{\frac{1}{2}}(\hat{\alpha} - \alpha^*) \sim MVN\big(0, C(\alpha^*)\big)$$

If in fact the model is based on a true likelihood $G(y|x,\alpha) = P(y|x,\alpha)$, so that $U_i(y_i, x_i|\alpha) = \partial \log\big(P(y_i|x_i,\alpha)\big)/\partial\alpha$, we have same results as before. The estimating equation gives the maximum likelihood estimate $\alpha$, is unbiased $\mathbb{E}[U(y_i, x_i|\alpha^*)] = 0$ for true parameters $\alpha^*$ and $A(\alpha) = B(\alpha) = \frac{1}{n}\mathcal{J}(\alpha) \Longrightarrow C(\alpha) = m\mathcal{J}^{-1}(\alpha)$ where $\mathcal{J}(\alpha) = \mathbb{E}[-l''(\alpha)] = \mathbb{E}[l'(\alpha)l'(\alpha)^T]$ is the expected information.

An important special case is a model which is not based on a true log-likelihood because there is pairwise structure in the data. Assume that the $n = \sum_{i=1}^{s} n_i$ observations can be divided into clusters of individual $i$ with $n_i$ observations each. In terms of the data set in $\mathcal{D}$, the individuals $i = 1, \dots,$ s are independent of each other but the observations $j = 1, \dots, n_i$ within the individual are not independent. The variance of the estimating equation can then be estimated via the cluster robust variance [60]:

$$B_n(\alpha) = \frac{1}{n}\sum_{i=1}^{m}\left[\sum_{j=1}^{n_i} U_{i,j}(\alpha)\right]\left[\sum_{j=1}^{n_i} U_{i,j}(\alpha)\right]^T$$

Longitudinal analysis is a field of statistics that deals with data that is clustered as (individual, time)-points. It considers many different partially or fully specified probability models for longitudinally clustered data. The generalized estimating equation (GEE) approach uses a standard regression model, such as least squares or logistic regression, and may include an explicit specification of how the observations are related or assumes them independent for fitting purposes. The latter is also called an independence working covariance structure. After fitting the model, the estimated variance the parameters is corrected for the correlation. Our approach that uses linear or logistic regression and corrects the variance of parameters corresponds to a GEE model with cluster robust confidence intervals [55].

# 3 Applications to censored data

In this chapter, we describe the new models proposed in this thesis. Each section describes a new model and how it can be used to analyse data in each application: 3.1 Retention and monetization, 3.2 Peer-to-Peer lending, 3.3 Unemployment, 3.4 Game recommendation. Individual papers that utilize these models are summarized in the next chapter “4. Research studies and results”.

## 3.1 Retention and monetization

### 3.1.1 Introduction

The foundation of any business is acquiring customers for a smaller cost than the profits they generate. It is often possible to measure the profits acquired from a single customer and the cost of acquiring them through marketing. If this difference is found to be positive, we should invest in marketing and the business model is viable. Sometimes the mechanisms of how a digital product brings in money, known as monetization, are not yet implemented or optimized. In this case the total amount of product use, known as retention, is often used as a proxy how much potential there is for monetization. For example, if a digital product shows advertisements, the amount of product use is in fact directly proportional to how much money is generated. This reality is reflected in a fundamental shift in how marketing can be approached as data science makes it an increasingly quantitative discipline.

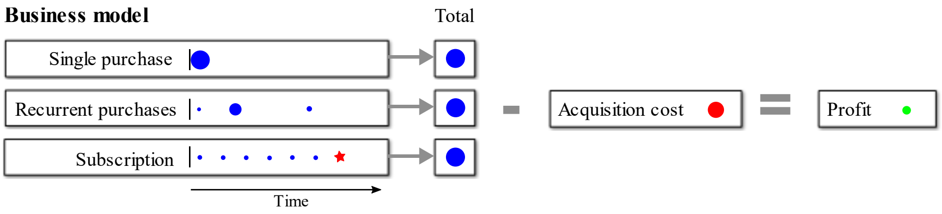


**Figure 9.** Example of three different business models: in each case we summarize the total purchases and deduct the acquisition cost to arrive at the profit per customer. The problem is how to estimate the expected purchase amount in each case.

Typically the user acquisition cost is known, so the problem is to estimate the ’profits acquired from a customer’. There are many different types of business models [61]. The following three are common in digital products and they are illustrated in **Figure 9**:

1. Single purchase: a single product is sold to a customer for a known price.
2. Recurrent purchases: a customer repeats purchases of variable amounts until they are no longer a customer (churn).
3. Subscription: a customer pays a fixed amount each month until they decide to cancel the subscription.

When a customer makes a single purchase for a known price, we directly have the total purchase amount. The problem is more difficult when customers repeat recurrent purchases of variable amounts until they decide to stop buying. This model probably describes most businesses, assuming that we can identify the customers.

The freemium model also falls under this category and is the most popular business model in digital products, especially games [62]. A freemium product is offered for free to attract as many customers as possible, and the product displays advertisements or additional content which is bought with real money [63]. The profit then depends on how actively and for long customers use the product or the paid extras, with the additional problem that we typically do not know if they have stopped using the product. When users stop using the product or stop being customers, we call this ’churn’. Calculating profit in the case of recurrent purchases of variable sizes, and unknown churn times, is the most difficult case and our methods focus on solving this problem. The subscription model can be seen as a special case. The purchases occur monthly for a fixed amount until the subscription is cancelled. We know when the customer cancels their subscription, so the problem is to simply model the time until cancellation. Even though the data is censored in the sense that not all customers have cancelled their subscription, standard methods in survival analysis can be used in this special case.

Player retention and churn have been analysed extensively in the game analytics literature. Academic literature understands ”retention” as an umbrella term that can refer to different engagement metrics, for example various measurements of player activity [64] [65] [66] [67] [68], session time [69], total sessions [70], total purchases [71], gates cleared [72], days active [73], playtime [74] [75], etc. Studies have not only measured retention and churn, but also predicted with Linear [70] [71], Logistic [69] [76] [77] [78], Cox [73] [74] [79] regression, and Hidden Markov models [80] [81]. A recent competition featured many advanced techniques for predicting churn and survival [82]. These metrics based on player totals could be modelled with standard survival analysis [83], but the problem is that player churn is in many cases unknown. This problem has been addressed by assuming that all players have churned [75] or defining a window of inactivity that defines the players as churned [77] [78] [80]. In contrast, industry understands retention as a specific ”retention rate” metric [85] that can be calculated as new data comes in. The retention rate calculates how many players return to the game every day from the day they started to play the game. The industry probably prefers this metric because it means that analytics can be used in real time game development scenarios, which are more time sensitive than academic studies that can analyse historical data sets. In these studies, we aimed to develop a new method that could be used to analyse the expected value of any metric in a similar fashion: lifetime value, playtime, total sessions, total purchases, etc.

### 3.1.2 Simulated data set

We need data to estimate the monetization or retention of customers. In this brief introduction, we use a simple simulated data set visualized in **Figure 10** and measure monetization as the expected number of purchases. This simulation can be used to verify that the method works. The studies use data sets from a real mobile game called 'Hipster Sheep', which is a free-to-play mobile game developed by a local game developer called Tribeflame Ltd. The simulated data has 100 customers that were followed for a maximum of 30 time units. The follow-ups vary by customer from 0 to 30 time units, based on arriving that many time units before the current time. Each customer makes purchase events, which occur seemingly random before the customer quits (churn). The churn times are typically unknown.

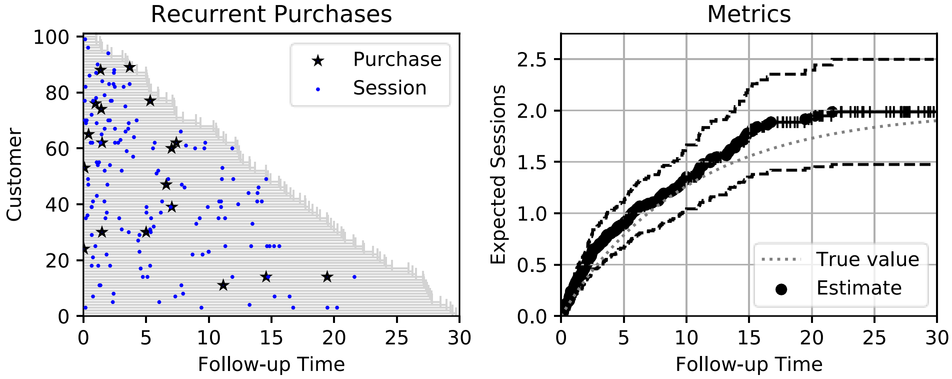


**Figure 10.** A simple simulated data set of purchases (left) and the estimated expected number of purchases calculated with our method contrasted to the true number of purchases based on the underlying process (right).

We now briefly explain how the data set was generated. To generate data for a single customer, we randomly sample a churn time, purchase times and a censoring time. The churn time is a random variable $T$ and the purchase times are random variables $T_1, T_2, \ldots, T_n$ where the number of purchases $n$ is also random. The total number of purchases at time $t$ is a random variable $N(t)$. Denote the number of purchases in a small interval $(t, t + \Delta t]$ by $\Delta N(t) = N(t + \Delta t) - N(t)$. Since no purchases occur after the churn time, we have $\Delta N(t) = 0$ for $t > T$. The process is then defined by two rates that we assume to be constant when we generate the simulated data: the churn rate $\mu$ and the purchase rate $\lambda$:

$$\mu = \lim_{\Delta t \to 0} P(t < T < t + \Delta t | T > t)/\Delta t$$
$$\lambda = \lim_{\Delta t \to 0} P(\Delta N(t) = 1 | T > t)/\Delta t$$

This implies that the churn time is $T \sim \text{Exponential}(\mu)$ with a survival function $S(t) = P(T > t) = \exp(-\mu t)$. The time between two purchases at $T_k$ and $T_{k+1}$ is denoted $\Delta T_k = T_{k+1} - T_k$, and this is also $\Delta T_k \sim \text{Exponential}(\lambda)$ with a survival function $S(t) = P(\Delta T_k > t) = \exp(-\lambda t)$.

The number of customers with a purchase in a small time interval is defined by $\gamma(t) = \lim_{\Delta t \to 0} P(\Delta N(t) = 1)/\Delta t$, and is known as the marginal purchase rate. The expected number of purchases at time *t* is given by its integral:

$$\mathbb{E}[N(t)] = \int_0^t \gamma(t)\, dt$$

Interesting mathematical results can be derived in this particular event process: many important aggregate statistics have a closed form. The marginal purchase rate can be expressed as $\gamma(t) = \lim_{\Delta t \to 0} P(\Delta N(t) = 1 | T > t) P(T > t)/\Delta t = \lambda \exp(-\mu t)$. The expected number of purchases at time $t$ is then:

$$\mathbb{E}[N(t)] = \int_0^t \gamma(t)\, dt = \frac{\lambda}{\mu}(1 - \exp(-\mu t))$$

The expected total number of purchases per customer is $\lim_{t \to \infty} \mathbb{E}[N(t)] = \lambda/\mu$ as $t \to \infty$. The probability distribution for the total number of purchases per customer has a Geometric distribution $\text{Geom}(\mu/(\lambda + \mu))$:

$$P[N(\infty) = n] = \int_0^\infty P(N(t) = n | T = t) P(T = t)\, dt$$
$$= \int_0^\infty \frac{(\lambda t)^n}{n!} \exp(-\lambda t)\, \mu \exp(-\mu t)\, dt = \frac{\mu}{\lambda + \mu}\left(\frac{\mu}{\lambda + \mu}\right)^n$$

We use the following procedure to generate the data. Each customer $i$ is defined by a churn time $t_i$, purchase times $t_{i,j}$ for $j = 1, \dots, n_i$, and censoring time $\tau_i$. First, sample a churn time $t_i \sim \text{Exponential}(\mu)$. Then, obtain the purchase times by sampling the time to next purchase $\Delta t_{i,j} \sim \text{Exponential}(\lambda)$. Initially $t_{i,0} = 0$ and the next purchase time is: $t_{i,j} = t_{i,j-i} + \Delta t_{i,j}$. Iterate while the new purchase time $t_{i,j}$ remains smaller than the churn time $t_i$. To generate the variable follow-up times, we sampled a censoring time $\tau_i \sim \text{Unif}(0, t_{\text{max}})$. Instead of the original data set, we then have the censoring time and only those purchases and churns that occurred before the censoring time. To generate the data, we used the churn rate $\mu = 0.1$ and the purchase rate $\lambda = 0.2$.

### 3.1.3 Retention and monetization model

There are several retention and monetization metrics that game developers are interested in [86]. Often we wish to estimate the amount of product use or money spent by a customer over time. These could be the number of sessions of purchases, for example. We do not know the underlying process $N(t)$, but wish to calculate an estimate $\widehat{\mathbb{E}}[N(t)]$ of $\mathbb{E}[N(t)]$ at time *t*. The data set in **Figure 10** was generated based on a constant churn rate and purchase rate, but in other data sets these rates could be customer specific or change over time. Because there is no guarantee that this particular parametric probabilistic model fits all data sets, it would be ideal to use a non-parametric method. In the papers (II&III), we introduced a statistical framework around a non-parametric method called the mean cumulative function (MCF) [87]. The MCF provides such an estimate of $\widehat{\mathbb{E}}[N(t)]$ and generalizes to real valued 'costs' $\widehat{\mathbb{E}}[C(t)]$ where $C(t) \in \mathbb{R}$. The 'cost' could be the total amount of product use in hours or purchase amounts in euros, for example.

We briefly present the central idea with the above data. Define the following for every customer *i*: $\Delta n_i(t)$ is the number of sessions at time *t* and $y_i(t) = \mathbb{I}(t \leq \tau_i)$ is an indicator of whether they are observable. We also define the aggregate values over all customers: $\Delta n(t) = \sum_{i=1}^{n} \Delta n_i(t)$ is the total number of sessions at time *t* and $y(t) = \sum_{i=1}^{n} y_i(t)$ is the total number of observable customers. Denote the distinct and ordered session times by $t_{(k)}$, meaning that $t_{(1)} < t_{(2)} < t_{(3)} < \cdots$ and $t_{(k)} \in \{t_{i,j}\}$. The following cumulative sum takes into account the number of observable customers and is unbiased estimator of the expected number of sessions at time *t*:

$$\widehat{\mathbb{E}}[N(t)] = \sum_{k:t_{(k)} \leq t} \frac{\Delta n(t_{(k)})}{y(t_{(k)})}$$

We characterize the customer $i$ at the time $t$ using the covariate vector $x_i(t) = \left(x_{i,1}(t), \dots, x_{i,d}(t)\right)$. For example, one could measure how the platform, country, age, product version, marketing campaign etc. affect retention or monetization. To model individual specific rates, one often makes the proportional rate assumption $\gamma_i(t) = \gamma_0(t) \exp\left(\beta^T x_i(t)\right)$, where $\gamma_0(t)$ is a baseline rate and the covariates $\exp\left(\beta^T x_i(t)\right)$ affect it by porpotional changes. An estimate of the expected number of sessions is a cumulative sum that takes into account the observable customers and covariates [31]:

$$\widehat{\mathbb{E}}[N_i(t)] = \sum_{k:t_{(k)} \leq t} \frac{\Delta n(t_{(k)})}{\sum_{j=1}^{n} y_j(t_{(k)}) \exp\left(\beta^T x_j(t_{(k)})\right)} \exp\left(\beta^T x_i(t_{(k)})\right)$$

The model is sometimes called semi-parametric because the baseline number of sessions is nonparametric and the covariates are described by a parameter vector $\beta$. To find an estimate $\hat{\beta}$ of the parameter vector $\beta$, one can solve the estimating equation $U(\beta) = 0$, which can be derived from a Poisson process likelihood [31]:

$$U(\beta) = \sum_{i=1}^{n} \sum_{k:t_{(k)} \leq t} y_i(t_{(k)}) x_i(t_{(k)}) \left[\Delta n_i(t_{(k)}) - \frac{\Delta n(t_{(k)})}{\sum_{j=1}^{n} y_j(t_{(k)}) \exp\left(\beta^T x_j(t_{(k)})\right)} \exp\left(\beta^T x_i(t_{(k)})\right)\right]$$

These equations can be motivated as maximum likelihood estimates of a Poisson process likelihood and they give valid estimates of the marginal rate functions. One is interested to know the variance of parameter estimates in statistics; to construct confidence intervals, perform significance tests, etc. Confidence intervals based on a Poisson process assumption are not valid in this case, and so called robust variance estimates are required. We go into details how these are calculated in Paper III.

The framework looks complicated because of technical details involved with a continuous time domain. However, it is possible to formulate the model in a standard regression setting if the time domain is discrete. Suppose we have customers $i = 1, \dots, n$ and time points $t = 1, \dots, m$. We define a month specific intercept vector $e_t = \left(\mathbb{I}(t = 1), \dots, \mathbb{I}(t = m)\right)^T$ with parameters $\alpha \in \mathbb{R}^m$ and a customer covariate vector $x_{i,t} \in \mathbb{R}^d$ at time $t$ with parameters $\beta \in \mathbb{R}^d$. The outcome $y_{i,t} \in \mathbb{R}$ for customer $i$ at time $t$ is defined if the customer was observable, i.e. not censored, before that time. The data set consists of triplets $\left\{\left(e_t, x_{i,t}, y_{i,t}\right)\right\}_{i=1,\dots,n, t=1,\dots,\tau_i}$. The outcome can be modelled with least squares regression:

$$\mathbb{E}\left[y_{i,t}\right] = \exp\left(\alpha^T e_t + \beta^T x_{i,t}\right)$$

The model includes non-parametric baseline estimates $\exp(\alpha_1), \dots, \exp(\alpha_m)$ of increments at times $t = 1, \dots, m$, and the individual specific factor $\exp\left(\beta^T x_{i,t}\right)$ that affects the baseline proportionally. One can estimate just the increments by including only the baseline $\alpha^T e_t$ without any covariates. It is possible to stratify the baseline by a cohort by adding an interaction between the cohorts and the intercepts, i.e. by including separate intercepts $\alpha_1^T z_{1,t}, \dots, \alpha_R^T z_{R,t}$ at every point for the $R$ cohorts.

There are many outcomes $y_{i,t}$ for customer $i$ at time $t$ that we can model: a binary activity status, the total number of sessions, the total number of purchases, the total product use time, the purchase amounts, etc. For example, in the previous problem we would model the total number of purchases in an interval $y_{i,t} = \mathbb{E}[\Delta N_i(t)]$. To get the expected number of purchases at time *t*, we first predict all of the increments $y_{i,1}, \dots, y_{i,t}$ and then calculate the cumulative sum:

$$N_i(t) = \sum_{k=1}^{t} y_{i,k}$$

Suppose the underlying data is continuous so that all of the purchase times are unique. As the number of time intervals goes to infinity, such that there is a unique time $t$ for every purchase $y_{i,t} > 0$ and $y_{i,x} = 0$ elsewhere, this asymptotically approaches the previous model. The confidence intervals of $\alpha$ and $\beta$ need to use the cluster robust sandwich estimate presented earlier.

### 3.1.4 Studies

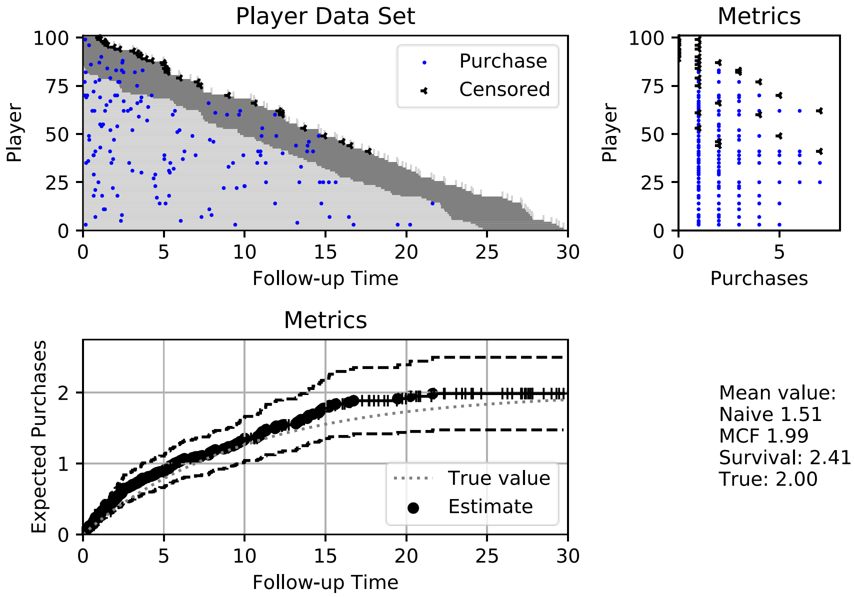


**Figure 11.** Example data set: metrics can be estimated 1.) ’horizontally’ which has a problem with censoring but standard method are applicable, or 2). ’vertically’ which is the idea underlying our new method.

In three studies (I, III, III) involving player retention and monetization, the methods that we investigated generalized standard retention metrics to continuous time, continuous values and even different follow-up lengths. Because discrete data is a special case of the continuous case, one can consider only the continuous case. We illustrate two views in **Figure 11**, where different metrics are obtained if the data is analysed ’horizontally’ or ’vertically’.

In the first study (Publication I), we investigated standard methods that have been developed for censored data in the field of Survival analysis [8]. We model metrics that have a value for every player $\{x_i\}_{i=1,\ldots,N}$, such as total playtime, sessions, level progression, time active, etc. This is illustrated in **Figure 11**, where we obtain total sessions per player ’summing horizontally’. However, the problem is that censoring means something different in this context. For players who have quit we know the final value of the metric. A value is said to be censored if it is larger than observed so far, i.e. if players have not yet quit we obtain more sessions. The problem is that we do not know who has quit and who has not. There are ad hoc solutions, such as defining a player who has played within the last 5 days as not having quit, making their total session count censored. Because these rules add bias to the data set, they do not fully solve the problem. In the figure, we for example see that the ’Survival’ estimate has a significant positive bias.

In the second and third studies (Publication II & III), we introduced a new non-parametric methods that solve the problem of censoring. The idea is to interpret the data set as recurrent events and associated values. For example, **in Figure 11** we display sessions and purchases, which have session lengths and purchase amounts. The idea is to estimate the expected value that has accumulated up to every follow-up time, by thinking of ’vertically summing’ the data set in a way that accounts for censoring. In the papers, we develop a comprehensive statistical framework for estimating the metric, confidence intervals, AB-tests, and regression. For example, in the bottom figure we estimated the average number of sessions for each follow-up time and see that 30 days seems sufficient to guess the final value to be around two sessions per player. Because we generated the data set with the churn rate $\mu = 0.1$ and the session rate $\lambda = 0.2$, the expected value is $\mathbb{E}[x_i(t)] = \frac{\lambda}{\mu}(1 - \exp(-\mu t))$ at time $t$. We should therefore eventually obtain $\lim_{t \to \infty} \mathbb{E}[x_i(t)] = \frac{\lambda}{\mu} = 2$ sessions. We in fact obtain the correct answer in the simulated data set.

## 3.2 Peer-To-Peer Lending

### 3.2.1 Introduction

Peer-to-peer (P2P) lending is a financing solution where online platforms act as intermediaries between individuals who seek to borrow money or invest by lending money [88]. Many different platforms offer this service. There are some differences between how the loans are handled, but all platforms are based on the same idea. Borrowers apply for a loan with their financial and demographic information that could include income, existing loans, purpose of the loan, age, country, education, etc. Lenders review the loan applications and choose the loans to invest in based on interest rates and risk in the loans [89].

Investors are attracted to the platforms by high interest rates. However, most loans have no collateral and defaults are quite common. If a loan defaults, an investor may lose some or all of the loan principal. This means that the loans have credit risk which needs to be compensated by setting interest rates high enough to cover the losses that occur. Most investors in peer-to-peer lending are not professionals, and setting the correct interest rate can be a challenging problem [90]. Almost every platform helps by providing credit ratings that indicate whether a loan has a high default risk, but ultimately the investors would like to estimate the profit in these loans. The credit risk, understood as the probability of loan default, has been analysed extensively with survival analysis based approaches [92] [93] [94] [95], and in a simplified form also with different machine learning models [96]. The profits have been analysed in a limited case where the data set only includes matured loans, i.e. historical loans that have now all either been repaid or defaulted [95].

If the payments in a loan are known, one can use discounted cashflow (DCF) analysis to calculate the profit. However, building a predictive model for the profits is difficult for real world data sets because the payments are censored. Many of the loans have several years of duration, and because the loans were issued recently we do not know all of their payments. It is clear that peer-to-peer lending platforms have this problem, but they try to provide some estimates nevertheless. Seemingly simple solutions lead to biased estimates of the profit: assuming that the future payments are made in full or not at all clearly results in overoptimistic or pessimistic results. Excluding on-going loans results in estimating lower profits than the reality, because loans that are more likely to survive are excluded more. An unbiased solution is to use an old data set where loans were issued so many years ago that we do not obtain any payments from them: all of the loans are either repaid or declared as lost principal. However, even if such a data set was available there is no guarantee that this data set accurately models current loans. In our papers, we developed a model to predict profits accurately despite the fact that some of the loan payments are not known. Our model predicts the expected monthly payments in censored data, and it incorporates the loan scheme so that investors can analyse the impact of different interest rates, default rates, and losses given defaults.

### 3.2.2 Bondora data set

Our two studies (Publication V & Publication IV) used a public peer-to-peer lending data provided by Bondora[1], which is a popular platform providing loans in Estonia, Finland, Spain and Slovakia. At the time of writing, the data set contained 144 031 loans with 113 columns. The columns include current loan status, borrower information and Bondora’s own predictions about the loan. Censoring is a problem: a significant fraction of the loans that were made even years ago remain censored.

We briefly use the mean cumulative function (MCF) developed in the previous section to estimate the aggregate profitability of loans in each year. In **Figure 12**, we assumed that an investor invested 1€ in each loan in the platform. We see that years 2010-2012 had shorter and profitable loans. A shift occurred in 2013 and 2014, with the platform making longer and possibly less profitable loans. In the loans from 2015 it appears that investors will recover the investment but have modest profits. The MCF is a reliable tool for estimating the aggregate payments in a portfolio of loans,

[1] https://www.bondora.com/fi/public-reports

but the regression model based on the MCF cannot be used for individual loans because cumulative payment curves of individual loans are not proportional.

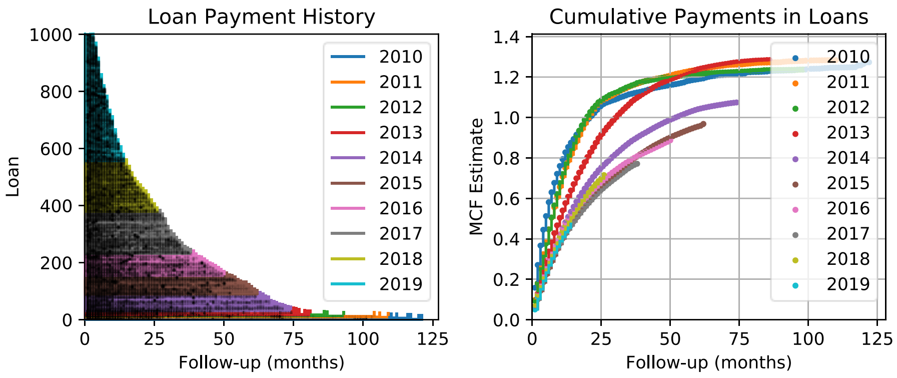


**Figure 12.** 1000 example loans in Bondora data and cumulative payments calculated with the MCF method by the loan year. It can be seen that the new loans have changed.

This fact is due to two underlying reasons illustrated in **Figure 13**. First, the cumulative payment curves look very different depending on the duration of the loan. Most loans make payments within schedule and some recovery payments are obtained thereafter. As the loan duration is increased, the payments are not shifted proportionally up or down as assumed by the MCF regression model, but extended over a longer duration. Second, even in loans of same duration, the cumulative payments are not proportional because defaults decrease the scheduled payments and increase the recovery payments. This means the initial part of the curve is shifted down and the second part is shifted up, which contradicts the proportionality assumption. For this reason, we develop a specialized model that includes knowledge about the loan schedule. The model has two parts: one for defaults and another for recoveries.

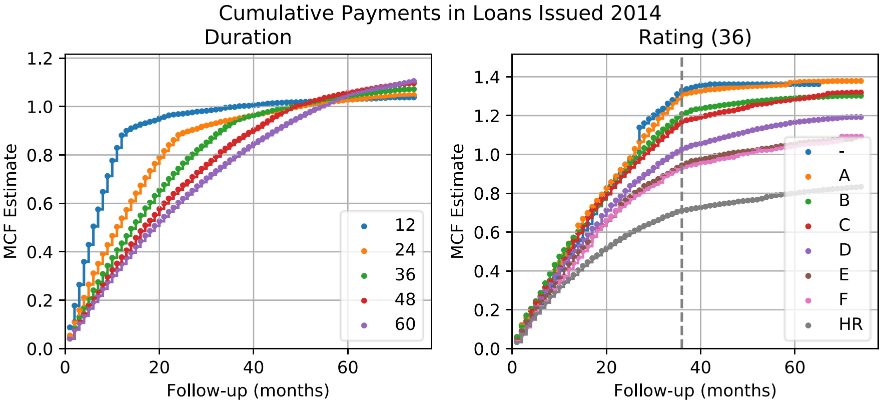


**Figure 13.** The cumulative payments are not proportional for two reasons: they have different duration and credit rating affects them non-proportionally. The regression framework around MCF cannot therefore be applied.

### 3.2.3 Default and recovery models

Peer-to-peer loans in Bondora are scheduled to have equal monthly payments for the duration of the loan. Each loan *i* is defined by three variables: the loan duration $n_i$, the interest rate $I_i$, and the loan amount $M_i$. From these variables, we calculate the monthly payments with the annuity formula $p_i = M_i I_i (1 + I_i)^n/((1 + I_i)^n - 1)$ [98]. In theory, the borrower should make a total of $n_i$ monthly payments for the duration of the loan, resulting in a sequence of payments $p_i$, $p_i$, ..., $p_i$, 0, 0, .... However, the borrowers may not make all of their scheduled monthly payments as agreed upon. A loan default occurs when a borrower is lacking a total amount of two consecutive monthly payments. This is a common threshold, since borrowers sometimes forget they had to make a monthly payment but do so after they are reminded. Once two payments are missed, it is likely that they stopped paying the loan. After this event, the lender aims to recover as much money as possible by getting the borrower to make some payments and eventually going through the courts to get a payment order. These payments rarely follow the loan schedule: sometimes a new schedule is set up, a part of loan is paid back, or the borrower is decleared unable to make payments. We therefore denote the actual monthly payments as $p_{i,1}$, $p_{i,2}$, $p_{i,3}$, ....

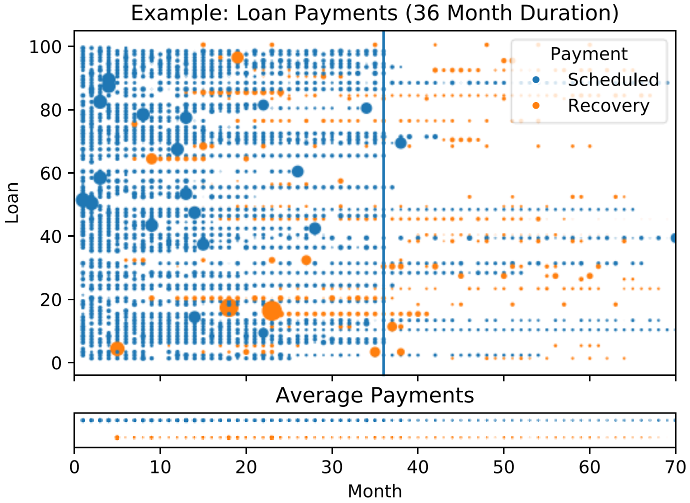


**Figure 14.** Example loan payments in 100 randomly selected loans of 36 month scheduled duration. Many loans make payments on schedule, but there are also significant recovery payments after loans have defaulted long to the future. The DCF analysis of the average payment corresponds to the profit in this portfolio.

An example of 100 random loans of 36 month duration is illustrated in **Figure 14**. We assume that one euro was invested in every loan, and illustrate their monthly payment amounts with the size of the dots. About half of the loans have either been paid on schedule or repaid early. The other half has defaulted, and we see that while some loans stop paying altogether, there are many recovery payments and some even recover the loan principal. Some of the loans have been rescheduled, but most of the loans either follow the original schedule until the loan is repaid or it defaults. In the bottom figure, we have illustrated the average monthly payment that investors would obtain if they invested one euro in a portfolio of these loans.

We can calculate the profit of a loan using discounted cashflow analysis (DCF), which is the standard approach in finance and takes into account the time value of money [98]. The monthly payments $p_{i,1}$, $p_{i,2}$, $p_{i,3}$, ...are discounted by the investors monthly profit requirement, known as the discount rate $r$, to arrive at the present value of the payments. The present value is the amount that the investor should pay for this loan to obtain the corresponding profit. The implicit profit requirement is the discount rate $r$ that makes the present value equal to the loan amount:

$$M_i = \sum_{t=1}^{\infty} \frac{p_{i,t}}{(1+r)^t}$$

However, the future payments in a new loan are initially unknown and our profit will be different depending on whether the loan defaults or not. For this reason, it is useful to think of the loan's monthly payments $p_{i,1}$, $p_{i,2}$, $p_{i,3}, \ldots$ as realizations of random variables $P_{i,1}$, $P_{i,2}$, $P_{i,3}, \ldots$ We then take the present value of the expected monthly payments, which is estimated by the average payment in **Figure 14**. The expected profit corresponds to the discount rate $r$ of a portfolio of infinitely many loans with the same characteristics:

$$M_i = \sum_{t=1}^{\infty} \frac{\mathbb{E}[P_{i,t}]}{(1+r)^t}$$

Our goal is to predict the expected monthly payments $E[P_{i,1}],E[P_{i,2}],E[P_{i,3}], \ldots$ in a loan and solve the discount rate $r$ which makes these payments equal to the loan amount. This is the predicted profit. Instead of predicting profits directly, we seek to predict the monthly payments from which the profit is calculated. A model based on predicting the monthly payments $p_{i,t}$ has an important benefit: it can be used with censored data. Because the training set consists of monthly payment observations, some payments can be censored in the sense that they are missing from the data. The model learns to predict all of the payments in a loan, as long as there are some loans that provide examples of later payments.

However, it is difficult to develop an accurate model for the monthly payments directly. Even without default uncertainty, the loan duration and interest rate would change the monthly payments in a complicated but completely deterministic way. We therefore add knowledge about how the monthly payments are calculated and model how they change based on loan features. We therefore developed two models: a default model for loan defaults and a loss given default model for recoveries after the default. The predictions from these two models imply the monthly payments.

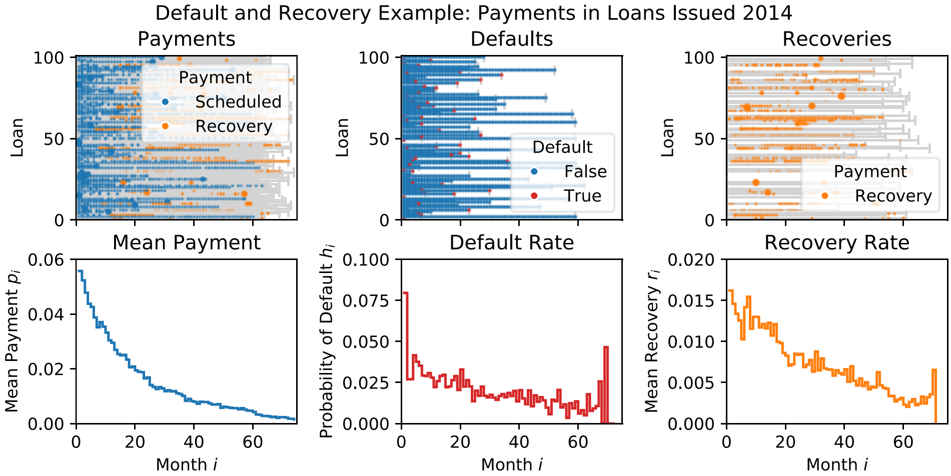


**Figure 15.** The data set is split into two complementary parts: the defaults can be described by the monthly default rate and the recoveries by the monthly recovery rate.

The models are based on splitting the data set into two as illustrated in **Figure 15**: one for payments on schedule and another for recovery payments. The second figure is the ’default part’ of the model. Every loan ether survives (False) or defaults (True) in every monthly interval, where the aggregate default rate is the proportion of observable loans that defaulted in that interval. The third figure is the ’recovery part’ of the model. Every loan that has defaulted makes recovery payments that total zero or above in every monthly interval up to their follow up time. The aggregate recovery rate is the average of recoveries in loans that are observable that interval.

The default model is based on predicting the number of monthly payments before a default occurs. We define that a loan defaulted on the monthly interval $t = 1,2,\ldots$ if it fell back a total of two scheduled monthly payments thereafter. Each monthly interval is a binary trial $Y_{i,t}$ of loan survival, where $Y_{i,t} = 0$ if the loan survives and $Y_{i,t} = 1$ if the loan defaults in the interval. Every loan therefore is a binary sequence $y_{i,1}, y_{i,2}, \ldots, y_{i,Ti}$, for example $0,0,0,1$ or $0,0,0,0,0,0$. The last observation is $1$ if the loan defaulted and $0$ if the loan was repaid or reached the follow-up limit. When $Y_{i,Ti} = 1$ we say $T_i$ is the default time, otherwise the default time is not known. The monthly default probability $\mu_{i,t} = E[Y_{i,t}] = P[Y_{i,t} = 1]$ in loans that survive to that interval is also known as the default rate. It does not matter than the loans are censored, because every binary trial is defined in terms of loans that are observed that month. We use logistic regression for loan specific default rates. Define the loan covariate vector as $x_i$, the corresponding parameter vector as $\beta$, and the month specific intercept as $\eta_t$. This is a discrete time analogue of the Cox proportional hazards model popular in survival analysis [99]:

$$\frac{\mu_{i,t}}{1-\mu_{i,t}} = \exp(\eta_t + \beta^T x_i)$$

The loss given default model is based on predicting values of monthly recovery payments after a default has occurred. We consider monthly recovery payments as a percentage of remaining principal, known as exposure at default. The payments in a month are summed together to obtain recovery payments following a default within monthly intervals $t = 1,2,\ldots.$ The goal is to predict the recoveries $R_{i,t}$ in each monthly interval. The monthly recovery $R_{i,t}$ is a real valued random variable. Each loan is therefore defined by a sequence of recoveries $R_{i,1}, R_{i,2}, \ldots, R_{i,Ci}$ from the first month after a default up to the last interval $C_i$ before the loan follow-up ended. Note that we have no observations if the loan did not default. The expected monthly recovery $\gamma_{i,t} = E[R_{i,t}]$ is the recovery rate. Again it does not matter than the loans are censored, because every recovery is defined in terms of loans that are observable that month after a default. We use least squares regression to model loan specific recovery rates. Define the loan covariate vector $x_i$, the parameter vector $\alpha$, and the month specific intercept $\theta_t$. We use a logarithmic link function to restrict the model to predict only positive values:

$$\gamma_{i,t} = \exp(\theta_t + \alpha^T x_i)$$

**Calculating payments from default and recovery rates**

After we train these two models, we can predict both the monthly default rates $\mu_{i,1}$, $\mu_{i,2}$, ... and the monthly recovery rates $\gamma_{i,1}$, $\gamma_{i,2}$, ... for any loan *i*. To calculate the expected monthly payments $E[P_{i,1}]$, $E[P_{i,2}]$, ... we proceed as follows. First, define the probabilities $f_{i,t} = P[T_i = t]$ of a default at month *t* and $s_{i,t} = P[T_i > t]$ of surviving month *t*. A loan of duration $n_i$ either defaults in a monthly interval $t = 1,\ldots,n_i$ or survives all of the monthly intervals. From the monthly default rates we can directly calculate the proportions of loans that default in each monthly interval:

$f_{i,1} = \mu_{i,1}$<br>
$f_{i,2} = (1- \mu_{i,1})\mu_{i,2}$<br>
$f_{i,3} = (1-\mu_{i,1})(1-\mu_{i,2})\mu_{i,,3}$<br>
…<br>
$f_{i,ni} = (1-\mu_{i,1})(1-\mu_{i,2})\ldots(1- \mu_{i,n(i-1)})\mu_{i,ni}$<br>
$s_{i,ni} = (1-\mu_{i,1})(1-\mu_{i,2})\ldots(1- \mu_{i,n(i-1)})(1- \mu_{i,ni})$

For a loan with default time *t* we obtain scheduled payments $p_{i,1}, \ldots, p_{i,t} = p_i$ up to the default time and recoveries $b_{i,1}\gamma_{i,1}$, $b_{i,1}\gamma_{i,2}$, ... thereafter. Note that the recoveries are calculated as the recovery rates $\gamma_{i,1}$, $\gamma_{i,2}$, ... multiplied by the remaining principal $b_{i,t}$ at time *t*. For a loan that does not default, we obtain the scheduled payments $p_i$. To calculate the predicted monthly payments, simply calculate these payments for every default time $T = 1,2,\ldots,n_i$, and the possibility of surviving $T > n_i$ . These are illustrated as rows in the following table.

| $T_i$ | **P(T$_i$=t)** | $\mathbf{P_{i,1}}$ | $\mathbf{P_{i,2}}$ | $\mathbf{P_{i,3}}$ | **…** | $\mathbf{P_{i,n-1}}$ | $\mathbf{P_{i,n}}$ | $\mathbf{P_{i,n+1}}$ | **…** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $f_{i,1}$ | $b_{i,1}\gamma_{i,1}$ | $b_{i,1}\gamma_{i,2}$ | $b_{i,1}\gamma_{i,3}$ | … | $b_{i,1}\gamma_{i,n-1}$ | $b_{i,1}\gamma_{i,n}$ | $b_{i,1}\gamma_{i,n+1}$ | … |
| 2 | $f_{i,2}$ | $p_i$ | $b_{i,2}\gamma_{i,1}$ | $b_{i,2}\gamma_{i,2}$ | … | $b_{i,2}\gamma_{i,n-2}$ | $b_{i,2}\gamma_{i,n-1}$ | $b_{i,2}\gamma_{i,n}$ | … |
| 3 | $f_{i,3}$ | $p_i$ | $p_i$ | $b_{i,3}\gamma_{i,1}$ | … | $b_{i,3}\gamma_{i,n-3}$ | $b_{i,3}\gamma_{i,n-2}$ | $b_{i,3}\gamma_{i,n-1}$ | … |
| … | … | … | … | … | … | … | … | … | … |
| $n_i$ | $f_{i,ni}$ | $p_i$ | $p_i$ | $p_i$ | … | $p_i$ | $b_{i,ni}\gamma_{i,1}$ | $b_{i,ni}\gamma_{i,2}$ | … |
|  | $s_{i,ni}$ | $p_i$ | $p_i$ | $p_i$ | … | $p_i$ | $p_i$ | 0 | … |
| $\sum$ | 1 | $E[P_{i,1}]$ | $E[P_{i,2}]$ | $E[P_{i,2}]$ | … | $E[P_{i,n-1}]$ | $E[P_{i,n}]$ | $E[P_{i,n+1}]$ | … |

To obtain the expected monthly payments, we sum the monthly payments in each row ($T_i$) weighted by the proportion of loans in each row ($P(T_i=t)$). This implies the following formula, with expected cashflows calculated from the scheduled payments and recoveries as:

$$E\left[P_{i,t}\right] = \mathbb{I}(t \le n_i) s_t p_i + \sum_{k=1}^{t} \gamma_{i,k} b_{t+1-k} f_{t+1-k}$$

**Constant default rate and loss given default**

It is possible to derive interesting mathematical results in the case that a loan has a constant default rate *h* over time and the present value of payments after default is a constant value *D*. To calculate the present value of payments after default for a given discount rate *r*, we use the DCF analysis:

$$PV\left(\gamma_{i,1}, \gamma_{i,2}, \dots | r\right) = \textstyle\sum_{t=1} \frac{\gamma_{i,t}}{(1+r)^t}.$$

The loss given default (LGD) is the difference between exposure at default (EAD), which is the loan balance remaining in the interval, and the present value:

$$D = PV\left(\gamma_{i,1}, \gamma_{i,2}, \dots | r\right) - 1.$$

where *D* is constant if the model for the recoveries does not include features such as the remaining loan balance, default time, default year, etc. Platforms deal with defaults differently, but in many of these cases it can be assumed that the LGD is time-invariant. Some platforms sell the defaulted loans to collection agencies for a percentage of principal remaining and *D* is simply the discount. Others offer buyback guarantees under some conditions, and then *D* is the probability that the platform itself defaults on these promises. In Bondora, we can estimate the LGD as the present value of the recovery payments, which we could calculate by assuming that investors in the platform have a 10% profit requirement, for example.

**Claim:** Define profit as the discount rate *r* such that the present value of expected payments is equal to the loan amount. Given a constant default rate *h*, loss given default *D*, and interest rate *I*, the profit independent of loan schedule and is given by:

$$r = hD + (1 - h)I$$

**Proof:** The proof is by induction from the last payment to the first. It is helpful to consider the **Figure 16**. The loan schedule is defined by the initial balance $B_0$, which is the loan amount. Then at months $t = 1,\dots,n$ either of two things happen to the loan balance $B_t$. The loan defaults with probability *h* and the present value is $(1+D)B_t$ at time $t+1$. Or, the loan survives with probability $1$-*h* and we compound the balance by the interest rate and divide it into payment $P_{t+1}$ and new balance $B_{t+1}$. In the later case $P_{t+1} + B_{t+1} = (1+I)B_t$. The loan schedule is any sequence of payments $P_1, \dots, P_n$ such that the loan eventually becomes repaid, i.e. $B_n = 0$.

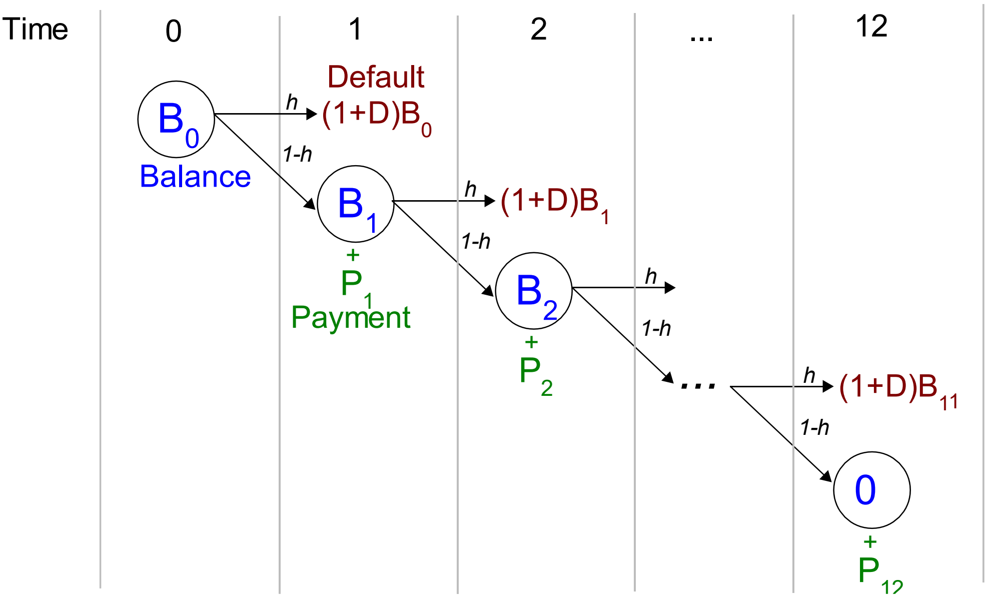


**Figure 16.** Illustration of the proof: going reverse from the last payment by induction.

Denote the present value of expected future payments for discount rate *r* at time *t* as $PV_t(r)$. We have the following recursive equation:

$$PV_{t-1}(r) = [h(1+D)B_{t-1} + (1-h)(PV_t(r) + P_t)]/(1+r)$$

Now, we claim that in this scheme the present value of future payments at time $t = 0$ is equal to the loan amount $B_0$ when the discount rate is $r = hD + (1\text{-}h)I$:

$$PV_0(hD + (1-h)I) = B_0$$

To show this, consider the last nonzero balance $B_{n-1}$. Because there are no future payments we have $PV_n(r) = 0$ and $P_n = (1+I)B_{n-1}$ therefore:

$$PV_{n-1}(r) = [h(1+D)B_{n-1} + (1-h)(0 + P_n)]/(1+r) = [h(1+D) + (1-h)(1+I)]B_{n-1}/(1+r)$$

Using the fact that $r = hD + (1\text{-}h)I \leftrightarrow 1+r = h(1+D) + (1\text{-}h)(1+I)$, we have:

$$PV_{n-1}(hD + (1-h)I) = B_{n-1}$$

This establishes the initial induction step, for some future time *t*. Using the recursive formula above, the expected present value at time *t* - *1* can now be derived

$$PV_{t-1}(hD + (1-h)I) = \frac{h(1+D)B_{t-1} + (1-h)(B_t + P_t)}{h(1+D) + (1-h)(1+I)} = \frac{h(1+D)B_{t-1} + (1-h)(1+I)B_{t-1}}{h(1+D) + (1-h)(1+I)} = B_{t-1}$$

The present value at time $t = 0$ is therefore $B_0$.

### 3.2.4 Studies

In the first study (Publication 4), we developed a special case of the model with a constant default rate and assumed that the loss given default is a known constant. We predicted the default time $T$ in censored data with survival analysis, assuming a parametric model $S(t) = P[T > t] = exp(-\lambda t)$ for the survival time. The idea is to fit the Cox regression model to measure the hazard $\lambda$ for each loan from censored data. This leads to a default rate $h = 1 – exp(-\lambda)$, which is the probability of default in each interval. The loss given default $D$ is Bondora's estimate of a percentage of remaining principal lost in the event of default, taking into account the time value of money. Given an interest rate $I$, *w*e then have the simple formula $i = (1-h)I + hD$ for the profit $i$. If the default rate is in fact constant, it is possible to estimate the true profit in a censored loan from a single interval: a loan profit is $I$ if it survives and $D$ if it defaults on this interval. The mean value in many such loans is the expected profit in a similar loan. We compared three approaches to detect the most profitable loans: Bondora's rating, the hazard estimate $\lambda$, and the profit estimate $i$. Picking the top 10% of loans with the profit estimate had the largest profit (8 vs. 10 vs. 20%).

In the second study (Publication 5), we generalized the model to a time-varying default rates and developed a model to estimate the loss given default in censored data. This is the complete model for censored loans. We modelled the default time $T$ directly with a discrete time analogue of a nonparametric Cox proportional hazards model. The probability of defaulting in each interval $\mu_{i,t}$ is predicted directly by the model. To model the loss given default, we used nonlinear least squares to predict the percentage of principal recovered $\gamma_{i,t}$ in each interval after the default. The expected cashflows in a loan can be calculated from $\mu_{i,t}$ and $\gamma_{i,t}$, after which profit is the discount rate that makes the cashflows equal to the loan amount. Since the loss given default model was a major generalization, we compared three approaches to selecting the defaulted loans with the lowest loss: the LGD estimate $D$, Bondora's LGD estimate, and Bondora's rating. We found that our model produced accurate estimates and the other models were not better than random.

## 3.3 Unemployment

### 3.3.1 Introduction

Unemployment is an important research topic with implications for individuals and governments alike [100]. Goverments and other institutions periodically produce macroeconometric labour market statistics, such as the population unemployment rate. Population level statistics do not consider that individuals experience different amounts of unemployment. Microeconometric studies have been performed at the individual level to assess the impact of government policies and individual's characteristics on unemployment [101]. For example, it is often found that health, age, gender, unemployment benefits, etc. affect the probability of unemployment [102]. Most studies use a regression model to assess the effect of a characteristic on the length of unemployment spells. However, the unemployment experience can be analysed from many perspectives; the risk of exiting unemployment, the risk of becoming unemployed, the risk of being unemployed, etc. In the end, the most important measure is probably the total amount of unemployment an individual experiences.

Machine learning methods focus on the predictive ability of the model on an independent test set. Unemployment seems like a natural application of machine learning: there are well-formulated prediction tasks that require accurate answers. Machine learning has been applied in the individual level to classify individuals as Long-Term Unemployed (LTU), see for example the references to many studies in [103]. We developed a predictive model in (Publication VI) for the full sequence of individual's labour market states. We model the unemployment status of an individual as a Markov chain with individual specific unemployment entry and exit probabilities. Similar models have been considered before in a statistical context [102]. The unemployment status is then implied by the transition probabilities of the Markov chain. The steady state probabilities imply that every individual has a lifetime unemployment prevalence that occurs in the long run. The model can be used to understand and predict different dynamics of unemployment, we evaluate how well the model is able to predict who exits unemployment, becomes unemployed, and is unemployed at a given time.

### 3.3.2 Registry data set

We obtained a research permission to an anonymized data set of unemployed people, which was collected by the ELY-centre (Centre for Economic Development, Transport and the Environment) in the Varsinais-Suomi economic area. The data set records all jobsekeers at the end of each month who are registered in the local unemployment agencies. This data set had not been used before in an unemployment study, but past studies have considered data obtained from unemployment registries. Every registry entry includes the time of collection, individual's anonymized identification code, jobseeker status, and personal information. Unemployed persons are required to register in order to receive unemployment benefits, so practically all unemployed people belong to the registry. A major advantage is that registry data is not biased by sample selection and subjective reporting; the registry is the population of all people who have been unemployed in Varsinais-Suomi during the follow-up.

The registry was collected from the beginning of 2013 to the end of 2017, which results in 60 monthly collections over the 5 years of follow-up. Our study used a random sample of 20 000 persons in the registry. We required the month to have been observed and the age to be 18-64 years for the observation to be included, otherwise the unemployment status is censored. The person is unemployed when their jobseeker status is recorded as 'unemployed' or 'laid off', and not unemployed when they have another jobseeker status or they are missing from the registry. The person information is included in every registry entry and it may change over time. We used the following features in the study to avoid identifying individuals: gender, work experience, age in 5 year buckets, level of education, and generic field of education.

### 3.3.3 Markov Chain model

Consider the data set visualization in **Figure 17**. Every individual experiences a sequence of labour market states over the follow-up period and different individuals have different amounts of unemployment. The population level perspective looks at the total number of people in each state, whereas the individual perspective investigates how different individuals contribute to these statistics. Our goal is to develop a model to predict the labour market status at the individual level, which for simplicity we consider to be unemployed or not. The model aims to explain the full individual labour market experience is as a binary sequence.

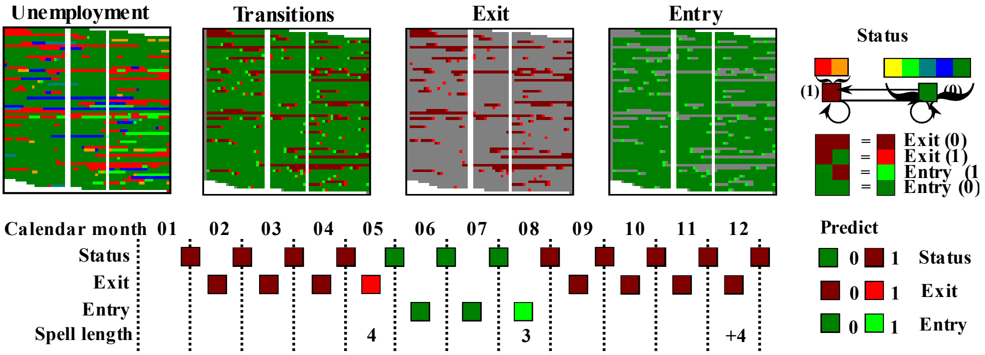


**Figure 17.** Individual's unemployment seems to consist of recurrent unemployment spells, where the individual transitions into and out of unemployment. It seems useful to model this process, which determines the unemployment status at any given time

The unemployment status of individual $i$ at time $t$ is denoted $x_{i,t} \in \{0,1\}$, and the censoring indicator $c_{i,t} \in \{0,1\}$ denotes whether the individual $i$ was observable (not censored) at time $t$. Formally, we model each individual as a stochastic process. We consider the unemployment status $x_{i,t}$ to be a realization of a random variable $X_{i,t} \in \{0,1\}$ in a stochastic process $\{X_{i,t}, t \geq 0\}$. Our goal is develop a probabilistic model for the individual unemployment sequence and predict the probability relevant unemployment events. The full model defines the probability of each unemployment sequence:

$$P\left(\{X_{i,t}, t \geq 0\}\right) = P\left(X_{i,t}, X_{i,t-1}, \dots, X_{i,0}\right)$$

We define two other important metrics: the probability of being unemployed at each time and the probability of transitioning in and out of unemployment, The state probability vector is the probability of each state at time *t*:

$$\overline{P}_{i,t} = \left(P\left(X_{i,t} = 0\right), P(X_{i,t} = 1)\right)$$

The transition probability matrix is the probability of transition from time $s$ to $t$:

$$\mathbb{P}_{i,s\to t} = \begin{pmatrix} P\left(X_{i,t} = 0 | X_{i,s} = 0\right) & P\left(X_{i,t} = 1 | X_{i,s} = 0\right) \\ P\left(X_{i,t} = 0 | X_{i,s} = 1\right) & P\left(X_{i,t} = 1 | X_{i,s} = 1\right) \end{pmatrix}$$

Looking at the **Figure 17**, it seems that individual labour market status consists of recurrent spells of unemployment, where it appears that different individuals have shorter or longer spells. We hypothesize that unemployment is a state which person exits and enters with some probability, and these transitions explain the observed data. This hypothesis defines the data through a simple generative process called the Markov Chain [105]. The Markov property states that the probability of being unemployed depends only on whether the previous observation was unemployed:

$$P\left(X_{i,t} = 1 | X_{i,t-1}, \dots, X_{i,0}\right) = P\left(X_{i,t} = 1 | X_{i,t-1}\right)$$

This condition implies that the process is defined in terms of two parameters: the probability of exiting and entering unemployment. This can be seen by expressing the probability of an unemployment sequence with the chain rule:

$$P\left(X_{i,t}, X_{i,t-1}, \dots, X_{i,0}\right) =$$
$$P\left(X_{i,t} | X_{i,t-1}, \dots, X_{i,0}\right) P\left(X_{i,t-1} | X_{i,t-2}, \dots, X_{i,0}\right) \dots P\left(X_{i,1} | X_{i,0}\right) P\left(X_{i,0}\right) =$$
$$P\left(X_{i,t} | X_{i,t-1}\right) P\left(X_{i,t-1} | X_{i,t-2}\right) \dots P\left(X_{i,1} | X_{i,0}\right) P\left(X_{i,0}\right)$$

These transition probabilities are the entries in a one-step transition matrix:

$$\mathbb{P}_{i,t-1\to t} = \begin{pmatrix} 1 - p_{i,t} & p_{i,t} \\ q_{i,t} & 1 - q_{i,t} \end{pmatrix} \text{ where } \begin{matrix} p_{i,t} = P\left(X_{i,t} = 1 | X_{i,t-1} = 0\right) \\ q_{i,t} = P\left(X_{i,t} = 0 | X_{i,t-1} = 1\right) \end{matrix}$$

In a Markov Chain, the transition probability matrix can then be expressed as the product of one-step transition matrices:

$$\mathbb{P}_{i,s\to t} = \prod_{k=s+1}^{t} \mathbb{P}_{i,k-1\to k}$$

An important special case is when the unemployment exit and entry rates are constant over time, i.e. $q_{i,t} = q_i$ and $p_{i,t} = p_i$. It makes sense to consider the rates as constant if we have a long term prespective and wish to ignore the effect of the economic cycle. We also have to use constant rates for future predictions, because it is impossible to know future changes in the baseline unemployment. As a benefit to this restriction, there are then well-known results about how the process evolves. In the case of time-constant rates, we have the following expression for the transition probability matrix $k$ time steps forward [106]:

$$\mathbb{P}_{i,s\to s+k} = \frac{1}{q_i+p_i}\begin{pmatrix} q_i & p_i \\ q_i & p_i \end{pmatrix} + \frac{(1-q_i-p_i)^k}{q_i+p_i}\begin{pmatrix} p_i & -p_i \\ -q_i & q_i \end{pmatrix}$$

The process converges to what are called the steady state probabilities:

$$\lim_{t\to\infty} \overline{P}_{i,t} = \left(\frac{q_i}{q_i+p_i}, \frac{p_i}{q_i+p_i}\right)$$

This means that in the long run, an individual with probabilities $q_i$ and $p_i$ can be predicted to spend $p_i/(q_i + p_i)$ of time in unemployment. If we have no knowledge of past unemployment states, it also makes sense to predict these probabilities. The probability of being unemployed is therefore determined by the probability of exiting and entering unemployment, and is influenced by both of them.

Individuals are either in or out of unemployment every month and they transition between these two states. Unemployment is considered as a negative for the society. We are therefore interested in predicting the individuals who are at the highest risk to be unemployed, to remain unemployed, and to become unemployed. We consider three prediction tasks for individual $i$ at time $t$: the unemployment probability $s_{i,t}$, the unemployment exit probability $q_{i,t}$ and the unemployment entry probability $p_{i,t}$. These are defined for the stochastic process $X_{i,t}$ as follows:

$$s_{i,t} = P(X_{i,t} = 1)$$
$$q_{i,t} = P(X_{i,t} = 0 | X_{i,t-1} = 1)$$
$$p_{i,t} = P(X_{i,t} = 1 | X_{i,t-1} = 0)$$

Predictions for new persons or future time points (Setting 2 & Setting 4 in chapter 2.3.4) are made using the constant transition probabilities. Given the unemployment status $x_{i,t}$ and the exit and entry probabilities $q_{i,t}$ and $p_{i,t}$ at the last observed time $t$, we make the future forward prediction of $k$ steps:

$$s_{i,t+k} = \frac{p_{i,t}}{q_{i,t}+p_{i,t}} - \frac{p_{i,t}}{q_{i,t}+p_{i,t}}\left(1 - q_{i,t} - p_{i,t}\right)^k \text{ if } x_{i,t} = 0$$
$$s_{i,t+k} = \frac{p_{i,t}}{q_{i,t}+p_{i,t}} + \frac{q_{i,t}}{q_{i,t}+p_{i,t}}\left(1 - q_{i,t} - p_{i,t}\right)^k \text{ if } x_{i,t} = 1$$

If we wish to predict lifetime unemployment or haven’t observed any previous unemployment states (Setting 3), we predict the steady state probability

$$s_{i,\infty} = \frac{p_{i,t}}{q_{i,t}+p_{i,t}}$$

We now explain how to fit the model to the data set. We model individual features age, gender, work experience, level and field of education with time-varying feature vectors $\overline{z}_{i,t} \in \mathbb{R}^d$. This is the registry information about person *i* at time *t*. We add a time-specific intercept to estimate the effect of the economic cycle during the training period. We add an individual-specific intercept to estimate differences between persons that are not explained by the covariates in the training period.

In the exit model, we have the following model parameters: coefficient vector $\overline{\alpha} \in \mathbb{R}^d$, time intercept vector $\overline{a} \in \mathbb{R}^t$, and individual intercept vector $\overline{u} \in \mathbb{R}^n$. In the entry model, we have the coefficient vector $\overline{\beta} \in \mathbb{R}^d$, time intercept vector $\overline{b} \in \mathbb{R}^t$, and individual intercept vector $\overline{v} \in \mathbb{R}^n$. We use the logistic regression model:

$$\frac{q_{i,t}}{1-q_{i,t}} = \exp\left(\overline{\alpha}^T \overline{z_{i,t}} + \overline{a}_t + \overline{u}_i\right)$$
$$\frac{p_{i,t}}{1-p_{i,t}} = \exp\left(\overline{\beta}^T \overline{z_{i,t}} + \overline{b}_t + \overline{v}_i\right)$$

First define the following index sets of times that are observable and relevant to each transition. The set $\mathbb{N}_{i,1}$ indexes the times relevant to unemployment exits: $\mathbb{N}_{i,1} = \{t \in \mathbb{N}: c_{i,t-1} = 1, c_{i,t} = 1, x_{i,t-1} = 1\}$. The index set $\mathbb{N}_{i,0}$ is the times relevant to unemployment entries: $\mathbb{N}_{i,0} = \{t \in \mathbb{N}: c_{i,t-1} = 1, c_{i,t} = 1, x_{i,t-1} = 0\}$. The individual's likelihood in the machine learning model that considers the exit and entry rates independent then factorizes into two separate models, as shown in the example in **Figure 17**:

$$L_{\overline{\alpha},\overline{a},\overline{u},\overline{\beta},\overline{b},\overline{v}}(x_{i,t}) = L_{\overline{\alpha},\overline{a},\overline{u}}(x_{i,t}) L_{\overline{\beta},\overline{b},\overline{v}}(x_{i,t})$$
$$L_{\overline{\alpha},\overline{a},\overline{u}}(x_{i,t}) = \prod_{k\in\mathbb{N}_{i,1}} (1-q_{i,k})^{\mathbb{I}(x_{i,k}=1)} q_{i,k}^{\mathbb{I}(x_{i,k}=0)}$$
$$L_{\overline{\beta},\overline{b},\overline{v}}(x_{i,t}) = \prod_{k\in\mathbb{N}_{i,1}} (1-p_{i,k})^{\mathbb{I}(x_{i,k}=0)} p_{i,k}^{\mathbb{I}(x_{i,k}=1)}$$

We use logistic regression where the coefficients are Tikhonov regularized by adding a squared coefficient penalty term for every coefficient vector. This means the solution is found through minimizing the penalized negative log likelihood:

$$\underset{\alpha\in\mathbb{R}^d,\beta\in\mathbb{R}^d,a\in\mathbb{R}^t,b\in\mathbb{R}^t,u\in\mathbb{R}^n,v\in\mathbb{R}^n}{\operatorname{argmin}} \left[-\log\left(\prod_{i=1\ldots n} L_{\overline{\alpha},\overline{a},\overline{u},\overline{\beta},\overline{b},\overline{v}}(x_{i,t})\right) + \lambda(\|\alpha\|^2 + \|\beta\|^2 + \|a\|^2 + \|b\|^2 + \|u\|^2 + \|v\|^2)\right]$$

## 3.4 Game recommendation

### 3.4.1 Introduction

Game recommendation is a natural application of recommender systems. Many platforms have gathered large datasets that describe how players have interacted with games and they have the problem of recommending interesting new games to play. A standard approach in such setting is collaborative filtering, which can recommend games based on similar player and game interactions. In games, recommender systems have been proposed based on collaborative filtering [107] [108], game content [109] [110], and different combinations of the two [109] [112]. While collaborative filtering has been found to produce the most accurate recommendations in many applications, it suffers from the cold-start problem [113]. If predictions are required for new players or new games without any interactions, it cannot make any predictions. This setting occurs in the real world, since new games are released all the time and platforms acquire new players that have not played any of their games.

For this reason, we investigated a new collaborative filtering algorithm and three new content based models in (Publication VII) that rely on game features, player features, or both. These methods are designed to predict as accurately as possible in four different settings: 1) predictions for known players and known games 2) predictions for new games 3) predictions for new players 4) predictions for new players and new games simultaneously. We develop a new collaborative filtering model for complete data based on multivariate normal distribution (MVN), a model based on game features (Tags), a model based on player features (Questions), and a cold-start model based on the interaction of game and player features (Tags X Questions). All models are simple, easy to interpret, and have mathematical shortcuts that allow fast training.

### 3.4.2 Data set and validation settings

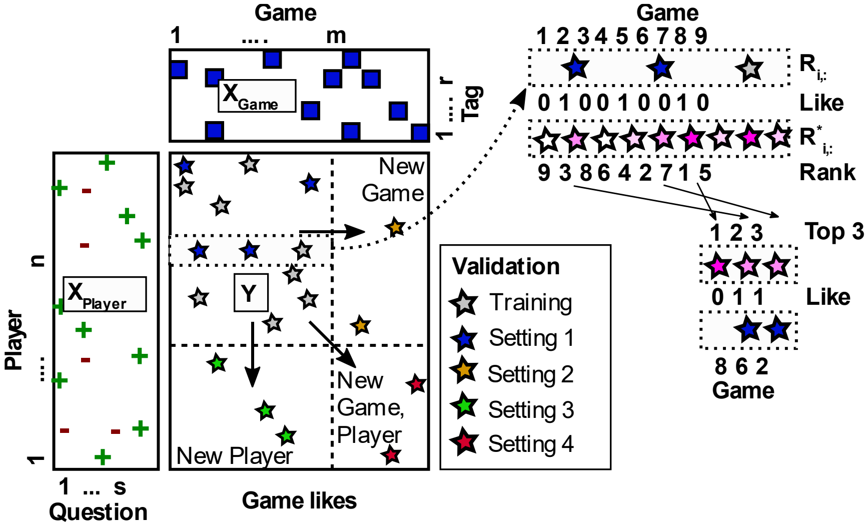


**Figure 18.** The training set (gray) consists of observed game likes and the different test sets (blue, yellow, green, red) correspond to different settings. For every player, the task is to predict a ranked list of games the player has not yet liked in the training set.

We define the problem as follows. Assume we have a total of $n$ players and $m$ games. Denote the player index $i \in \{1, ..., n\}$ and game index $j \in \{1, ..., m\}$. The game likes are a $n \times m$ binary game like matrix $Y \in \mathbb{R}^{n \times m}$ illustrated in **Figure 18**:

$$Y_{i,j} = \mathbb{I}(\text{player } i \text{ likes game } j)$$

We obtained a permission to use a data set of game likes collected by the Centre for Collaborative Research at Turku School of Economics. The data set has 15894 players and 6465 games with a total of 80916 game likes. Every player has mentioned their favourite games and the rest of the games are not their favourites. The data set is complete, which means the matrix has no missing entries. Complete data sets are typically investigated in the field of implicit recommendation [114]. It would be possible to create implicit game likes from game ownership, for example.

We have two sources of features: players and games. Player features are stored in a $n \times s$ matrix of player features $X_{\text{player}} \in \mathbb{R}^{n \times s}$ and game features in a $m \times r$ matrix of game features $X_{\text{game}} \in \mathbb{R}^{m \times r}$. Player features were created by asking the player a randomized set from 61 questions of gaming preferences ('Exploring the gameworld and its secrets', 'Commanding units or troops', ' Breeding, training and taking care of pets', …). The answers were on a Likert scale: strongly dislike (-2), dislike (-1), no preference (0), like (1), strongly like (2). We filled the missing value by 0 when a question was not asked, implying no stated preference. Game features were created by mining the Steam and Internet Games DataBase (IGDB) platforms game tags.

This resulted in a binary vector of length 379, where each element (0,1) indicates the presence or absence of each game tag.

The task is to predict a list of game recommendations for every player. Formally, the player has a game like vector $Y_{i,:}$ and the goal is to predict a real-valued score vector $Y_{i,:}^{*}$ of also length $m$. The ranking of elements in $Y_{i,:}^{*}$ is the order of games recommended. Denote the indices that sort the predictions in descending order as $R_{i,:}^{*}$. For example, if we have five games and player $i$ likes games $Y_{i,:} = (0,1,0,0,1)$, the predicted scores $Y_{i,:}^{*} = (0.1,1.1,0.8,0.7,2.1)$ imply a recommendation list $R_{i,:}^{*} = (5,2,3,4,1)$ which correctly ranks the liked games (1,1,0,0,0). The predictions for all players and games are a $n\ x\ m$ score matrix $Y^{*} \in \mathbb{R}^{n\times m}$, where the ranking of each row in $R^{*} \in \mathbb{R}^{n\times m}$ is the recommendation list for each player. Recommendation lists typically include only top $k$ games, which is called Top-N recommendation [114]. In this case, accuracy is measured by ranking metrics, and we considered the Precision@k and nDCG@k metrics in the study. The games liked in the training set are excluded from the recommendation list $R_{i,:}^{*}$ in the evaluation.

It is possible to consider four different tasks as illustrated in **Figure 18**. Denote all the training set players as $P \subseteq \{1,\ldots,n\}$ and games as $G \subseteq \{1,\ldots,m\}$. We require predictions for player $i$ and game $j$ in four different settings

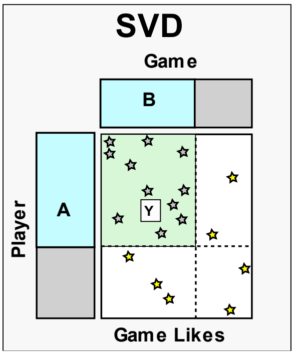
1. Setting 1: Standard setting ($i \in P$ and $j \in G$).
2. Setting 2: Prediction for new games ($i \in P$ and $j \notin G$).
3. Setting 3: Prediction for new players ($i \notin P$ and $j \in G$).
4. Setting 4: Prediction for new players and games ($i \notin P$ and $j \notin G$).

To evaluate performance in each setting, we divided players and games into four different validation sets. We sampled 25% of players into ’test players’ (new players) and 25% of games into ’test games’ (new games). Setting 1 validation set has 20% of ’training players’ that have more than 3 game likes. A random selection of 3 games is then the seed that belongs to the training set for these players, and the training set also includes all games of the remaining 80% of players. The missing game likes of the 20% subset are the Setting 1 validation set. Setting 2 validation set is the game like submatrix with ’training players’ and ’test games’. Setting 3 validation set is the game like submatrix with ’test players’ and ’training games’. Setting 4 validation set is the submatrix with ’test players’ and ’test games’.

### 3.4.3 Game recommendation models

We now present the three content based methods and motivate them through the baseline collaborative filtering algorithm called the singular value decomposition (SVD). We call them the Tags, Questions and Tags X Questions models. Content based methods were the main focus of the paper, since these allow generalization into new games (Setting 2), new players (Setting 3) or new games and players simultaneously (Setting 4).

#### 3.4.3.1 SVD



| B | World of Warcraft | The Elder Scrolls V: Skyrim | Overwatch | TETRIS | The Witcher 3: Wild Hunt |
|---|---|---|---|---|---|
| Factor #1 | -0.465727 | 0.531080 | -0.003032 | -0.057595 | 0.101931 |
| Factor #2 | -0.483305 | 0.692415 | 0.756132 | 0.670662 | 1.199870 |
| Factor #3 | 0.652601 | 1.037540 | 0.071260 | 1.103199 | 0.426366 |
| Factor #4 | -0.603743 | 0.120316 | 0.446428 | -0.280086 | -0.850675 |
| Factor #5 | 0.279523 | 1.190053 | -0.036263 | -0.514542 | 0.101282 |

| A | Factor #1 | Factor #2 | Factor #3 | Factor #4 | Factor #5 |
|---|---|---|---|---|---|
| 15888 | 0.001488 | 0.000513 | 0.000270 | 0.000792 | 0.000221 |
| 15889 | -0.000052 | 0.000353 | -0.000416 | 0.000362 | 0.000429 |
| 15890 | 0.001685 | 0.001098 | -0.000710 | 0.000640 | 0.000507 |
| 15891 | 0.002329 | 0.002705 | -0.002379 | 0.003477 | 0.003603 |
| 15892 | 0.000358 | -0.005193 | -0.000525 | 0.009831 | -0.000240 |
| 15893 | -0.001646 | 0.002552 | 0.005079 | 0.002306 | -0.004719 |
| 15894 | 0.000489 | 0.002492 | -0.003188 | 0.001557 | 0.001469 |

**Figure 19.** Collaborative filtering model based on SVD generalizes to Setting 1.

SVD is a standard collaborative filtering method that has been found to perform well in many recommendation problems [115], and we also use it as a baseline. As illustrated in **Figure 19**, the method does not use any game or player features. Instead, the model assumes that player *i* is described by *d* latent factors $a_{i1}$ , $a_{i2,}$ ..., $a_{id}$, and game *j* is described by *d* latent factors $b_{j1}$ , $b_{j2,}$ ..., $b_{ij}$. The prediction for player *i* and game *j* is the product of these latent factors:

$$y_{i,j} = a_{i,1}b_{j,1} + a_{i,2}b_{j,2} + \cdots + a_{i,d}b_{j,d}$$

Even though the model is linear, there rarely is a straightforward interpretation for the latent factors. The latent factors are initially unknown and we take them as model parameters. Denote the player factors as rows of the matrix $A \in \mathbb{R}^{n\times d}$ and the game factors as the rows of the matrix $B \in \mathbb{R}^{m\times d}$. Then the predicted scores are a matrix $Y^* = AB^T$, which is visualized in **Figure 19**. To find the optimal latent factors $\hat{A}$ and $\hat{B}$ we minimize the regularized least squares:

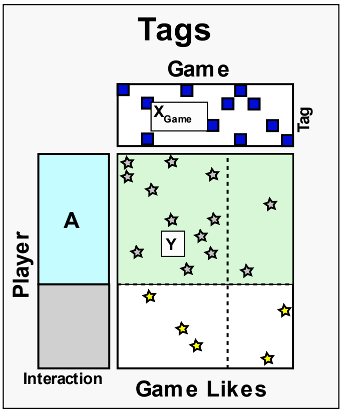
$$\hat{A}, \hat{B} = \underset{\mathrm{A}\in\mathbb{R}^{n\times d},\mathrm{B}\in\mathbb{R}^{m\times d}}{\operatorname{argmin}} ||Y - AB^T||_F^2 + \lambda||A||_F^2 + \lambda||B||_F^2$$

One approach to find the optimal parameters is alternating least squares (ALS), which is an iterative optimization method designed for this particular task [113]. Denote $A_{(t)}$ and $B_{(t)}$ as the parameter estimates at time *t*. Initialize $B_{(0)}$ with random values. Now assume that $B_{(t-1)}$ is fixed, and find the linear least squares solution for $A_{(t)}$. Then assume that the resulting $A_{(t)}$ is fixed and and find the least squares solution for $B_{(t)}$. Repeat these iteratively for *t = 1, 2, …* until the estimates converge or the maximum number of iterations is reached:

$$A_{(t)} = \left(B_{(t-1)}^T B_{(t-1)} + \lambda I\right)^{-1} B_{(t-1)}^T Y^T$$
$$B_{(t)} = \left(A_{(t)}^T A_{(t)} + \lambda I\right)^{-1} A_{(t)}^T Y$$

Because both $A$ and $B$ are parameters, the model has a large degree of flexibility to fit the data set and often achieves high accuracy. However, predictions can be made only for players and games that have likes in the training set, because the latent representations can be learned only for them.

### 3.4.3.2 Tags



| A | Puzzle | Real-time Strategy | Dating Sim | Racing | Lara Croft |
|---|---|---|---|---|---|
| 15888 | -2.37 | -0.04 | 8.37 | -0.79 | 0.07 |
| 15889 | -1.22 | 0.06 | -0.31 | 11.74 | 0.01 |
| 15890 | 0.88 | -0.11 | -0.28 | 5.66 | -0.18 |
| 15891 | 0.58 | 0.35 | 7.33 | -1.75 | -0.16 |
| 15892 | -2.52 | -3.71 | 7.96 | -2.63 | 0.09 |
| 15893 | -0.07 | 7.59 | -0.26 | -1.10 | 0.01 |
| 15894 | -0.74 | -0.82 | 0.09 | -3.34 | -0.06 |

**Figure 20.** Content model based on game features (Tags) generalizes.to Setting 1&2.

As illustrated in **Figure 20**, the first content model is based on game features. Assume that the game $j$ is described by features $x_{j,1}$, $x_{j,2}$, …, $x_{j,r}$. In our case, these features are binary indicators of game tags and we call this the 'Tags' model. The player *i* has an interaction with each game feature described by parameters $a_{i,1}$ , $a_{i,2}$, …, $a_{i,r}$, The prediction for player *i* and game *j* is the sum:

$$y_{i,j} = a_{i,1}x_{j,1} + a_{i,2}x_{j,2} + \cdots + a_{i,r}x_{j,r}$$

This corresponds to a linear model where players have separate coefficients that predict how each player interacts with the game tags. Each coefficient is the response to that particular tag, based on the games that the player has played. Denote the player interactions as rows of the matrix $A \in \mathbb{R}^{n \times r}$ and the game features as rows of the matrix $X_{\text{game}} \in \mathbb{R}^{m \times r}$. Then the predicted scores are a matrix $Y^* = AX_{\text{game}}^T$, visualized in **Figure 20**. To find the optimal parameters $\hat{A}$ we again minimize the regularized least squares:

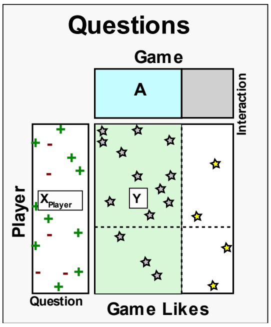
$$\hat{A} = \underset{A \in \mathbb{R}^{n \times r}}{\operatorname{argmin}} \left\| Y - AX_{\text{game}}^{T} \right\|_F^2 + \lambda \|A\|_F^2$$

The parameter matrix can be found directly with the least squares solution:

$$\hat{A}^T = \left(X_{\text{game}}^T X_{\text{game}} + \lambda I\right)^{-1} X_{\text{game}}^T Y^T$$

The factorization $Y^* = AX_{\text{game}}^T$ shows how the model can be thought of as a special case of the SVD where the latent game features are fixed to $X_{\text{game}}$. Because the model has less flexibility than the SVD, one would expect it to predict less accurately. However, the major advantage of using provided features is that predictions can be made for new games that have these features.

### 3.4.3.3 Questions



| A | World of Warcraft | The Elder Scrolls V: Skyrim | Overwatch | TETRIS | The Witcher 3: Wild Hunt |
|---|---|---|---|---|---|
| Exploring the gameworld and its secrets | 0.79 | 0.57 | 0.32 | -0.25 | 0.24 |
| Commanding units or troops | -0.06 | 0.04 | 0.13 | 0.40 | 0.10 |
| Breeding, training and taking care of pets | 0.24 | -0.15 | 0.06 | -0.36 | 0.16 |
| Sniping to eliminate | -1.11 | -1.21 | 0.62 | -0.96 | -0.76 |
| Challenges of fast reaction | -0.64 | -0.73 | 0.35 | 1.19 | -0.70 |
| Developing skills and abilities | 0.60 | 0.57 | 0.26 | -0.07 | 0.45 |
| Jumping on platforms and bouncing on walls | -0.19 | 0.26 | -0.01 | 0.21 | 0.26 |

**Figure 21.** Content model based on player features (Questions) generalizes to Setting 1&3.

As illustrated in **Figure 21**, the second content model is based on player features. Assume that player *i* is described by features $x_{i,1}$, $x_{i,2}$, ..., $x_{i,s}$. In our case, these features are questionnaire answers of gaming preferences and we call this the 'Questions' model. The game *j* has an interaction with each player feature described by parameters $a_{j,1}$ , $a_{j,2}$, ..., $a_{j,s}$, The prediction for player *i* and game *j* is the sum:

$$y_{i,j} = x_{i,1}a_{j,1} + x_{i,2}a_{j,2} + \cdots + x_{i,s}a_{j,s}$$

This corresponds to a linear model where games have separate coefficients that predict how each game interacts with the player preferences. Each coefficient is the reponse to that particular preference, based on the players that have played the game. Denote the player features as rows of the matrix $X_{\text{player}} \in \mathbb{R}^{n \times s}$ and the game interactions as rows of the matrix $A \in \mathbb{R}^{m \times s}$. Then the predicted scores are a matrix $Y^* = X_{\text{player}} A^T$, visualized in **Figure 21**. To find the optimal parameters $\hat{A}$ we again minimize the regularized least squares:

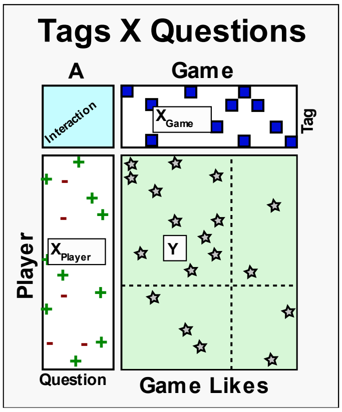
$$\hat{A} = \underset{A\in\mathbb{R}^{m\times s}}{\operatorname{argmin}} \left\| Y - X_{\text{player}} A^T \right\|_F^2 + \lambda \|A\|_F^2$$

The parameter matrix can be found directly with the least squares solution:

$$\hat{A}^T = \left( X_{\text{player}}^T X_{\text{player}} + \lambda I \right)^{-1} X_{\text{player}}^T Y$$

The factorization $Y^* = X_{\text{player}} A^T$ shows how the model can be thought of as a special case of the SVD where the latent player features are fixed to $X_{\text{player}}$. Again, because the model has less flexibility than the SVD one would expect it to predict less accurately. However, the major advantage of using provided features is that predictions can be made for new players that have these features.

### 3.4.3.4 Tags x Questions



| A | Puzzle | Real-time Strategy | Dating Sim | Racing | Lara Croft |
|---|---|---|---|---|---|
| Exploring the gameworld and its secrets | -0.04 | -0.03 | 0.87 | -0.18 | 0.58 |
| Commanding units or troops | 0.25 | -0.06 | 0.21 | 0.03 | -1.14 |
| Breeding, training and taking care of pets | -0.16 | -0.15 | 1.07 | -0.07 | 0.03 |
| Sniping to eliminate | -0.40 | 0.29 | -1.47 | 0.03 | 0.75 |
| Challenges of fast reaction | -0.20 | -0.28 | -1.18 | -0.02 | 0.95 |
| Developing skills and abilities | 0.08 | 0.01 | 0.52 | -0.16 | 2.55 |
| Jumping on platforms and bouncing on walls | -0.10 | -0.19 | 0.93 | 0.04 | 4.12 |

**Figure 22.** Content model based on both game features (Puzzle, Real-Time, Strategy, Dating Sim, …) and player features (likes Exploring the gameworld and its secrets, …) learns their interaction and generalizes to Setting 1&2&3&4.

As illustrated in **Figure 22**, the third content model is based on both player and game features. Assume the player $i$ is described by features $u_{i,1}$, $u_{i,2}$, …, $u_{i,s}$ and the game $j$ is described by features $v_{i,1}$, $v_{i,2}$, …, $v_{i,r}$. In our case, these were the questionnaire answers of gaming preferences and the binary indicators of game tags. To predict $y_{i,j}$ for every player $i$ and game $j$, we define a feature vector every pair as the interaction between every player feature and game feature: $u_{i,1}vi_{,1}$, $u_{i,1}v_{i,2}$, …, $u_{i,1}v_{i,r}$, …, $u_{i,s}v_{i,1}$, $u_{i,s}v_{i,2}$, …, $u_{i,s}v_{i,r}$. These features are the interactions between questionnaire answers and game tags. The strength of each interaction is described by parameters $a_{1,1}$, ..., $a_{1,r}$, ..., $a_{s,1}$, ..., $a_{s,r}$. The prediction for player $i$ and game $j$ is the sum of all interactions:

$$y_{i,j} = \sum_{k=1}^{r} \sum_{l=1}^{s} a_{k,l}\, u_{j,k} v_{j,l}$$

This is a linear model where each player preference and game tag has a separate coefficient that predicts how strongly they interact. Each coefficient is the strength of that interaction, based on the player preferences and game tags that occur together when a player likes a game. Denote the player features as rows of the matrix $X_{\text{player}} \in \mathbb{R}^{n \times s}$ and the game features as rows of the matrix $X_{\text{game}} \in \mathbb{R}^{m \times r}$. The feature matrix of every pair is then the Kronecker product $X_{\text{pair}} = X_{\text{player}} \otimes X_{\text{game}} \in \mathbb{R}^{nm \times sr}$. Denote the matrix of parameters as $A \in \mathbb{R}^{s \times r}$ and the vectorizing operation of stacking the columns as $\text{vec}(A) \in \mathbb{R}^{sr \times 1}$. Then the predicted scores are a matrix $\text{vec}(Y^*) = X_{\text{pair}}\, \text{vec}(A)$. The parameters are found by the regularized least squares:

$$A = \underset{\text{A} \in \mathbb{R}^{s \times r}}{\text{argmin}} \left\| \text{vec}(Y) - X_{\text{pair}}\, \text{vec}(A) \right\|_F^2 + \lambda \|\text{vec}(A)\|_F^2$$

The parameter matrix can be found directly with the least squares solution:

$$\text{vec}(A) = \left( X_{\text{pair}}^T X_{\text{pair}} + \lambda I \right)^{-1} X_{\text{pair}}^T\, \text{vec}(Y)$$

The factorization $Y^* = X_{\text{pair}}\, \text{vec}(A)$ shows the model is a standard regression model that predicts with the interaction of player and game features. The model where both feature matrices are given has even less flexibility than the models where either was fixed. However, the major advantage of using both features is that predictions can be made for both new players and new games.

# 4 Research Studies and Results

This chapter presents a brief summary of each publication included in the thesis. The following table gives a short description of each study: whether the modelling focus is on statistics or machine learning (Focus), what the feature vector is based on (Features), the machine learning model used for the pairwise data (Based on), and which new models are described based on a novel mathematical formulation as a pairwise problem (New Models).

| Paper | Focus | Features | Based on | New Models |
|---|---|---|---|---|
| **Playtime measurement with survival analysis. IEEE Transactions on Games** | Stats | Player | Survival Analysis | |
| **A/B-test of retention and monetization using the Cox model.** | Stats | Player | Survival Analysis | MCF (regression) |
| **Measuring player retention and monetization using the mean cumulative function** | Stats | Player | Linear regression (log) | MCF |
| **Predicting Expected Profit in Ongoing Peer-to-Peer Loans with Survival Analysis Based Profit Scoring.** | ML, Stats | Loan | Survival Analysis | Default (constant) |
| **Predicting profitability of peer-to-peer loans with recovery models for censored data.** | ML, Stats | Loan | Linear (log), Logistic regression | Default<br>Recovery |
| **Predicting Unemployment with Machine Learning Based on Registry Data.** | ML, Stats | Person | Regularized and mixed effects logistic regression | Entry<br>Exit<br>Prevalence |
| **Content Based Player and Game Interaction Model for Game Recommendation in the Cold Start setting.** | ML | Player, Game | Linear regression | MVN<br>Questions<br>Tags<br>QuestionsXTags |

# 4.1 Research publications

## 4.1.1 Retention and monetization

### 4.1.1.1 Playtime measurement with survival analysis

**Motivation**: Game developers routinely use game analytics to obtain insights into data gathered from players. Analytics in games has focused on player retention and churn in particular. Retention means that players are engaged with the game, and churn means that players quit the game, either momentarily or definitely. These are complementary and opposite concepts, and many ways have been devised to measure them. However, measurement presents unique challenges in games. Game developers typically wish to perform analytics before every player has churned from the game. For example, consider a game that is being developed or successful games that have long playtimes.  Because not all users have churned, their retention is not yet known and we call the data censored. In this study, we investigated how methods in survival analysis can be used to analyse player retention and churn in a timely and effective manner in censored data.

**Data and methods**: We analyse how survival analysis contributes to the measurement, visualization and comparison of playtime. We use data from the Hipster Sheep, previously named Hipster Maze, mobile game with 3753 players in versions 1.11, 1.15, and 1.18. We also illustrate the methods with a random sample of 10 players. Survival analysis is a field of statistics that can be used to analyse censored data when player retention is defined as a duration variable. Our focus is on playtime for clarity, but any player specific positive value could be used: session length, number of sessions, level progression or total subscription time, etc. Survival analysis is well-suited for the metrics in gaming, because it is developed for positive, non-normal and possibly censored data. It does not require parametric approaches, which may not match complex phenomena. While survival analysis solves the censoring problem, there is a complication with the fact that we may not know the censoring status of a player: players do not notify us whether they have churned or not. Various rules and churn prediction models have been used to impute churn status, and we assume that such a method has been used before analysis.

The survival curve is a funnel type estimate of how many percentage of players remain in in the game as a function of playtime. The hazard, which is the churn rate in the case of playtime, estimates the rate at which players churn from the game at different time points, Sudden increases from the trend could signify game design flaws and decreases point to content that players find engaging. The mean value aggregates the survival curve to a clear and unambiguous metric. Because many free-to-play games have a small segment of long term players, it is also useful investigate the quantiles. The 50% quantile known as the median can be thought of as a typical player. Finally, the log-rank test provides an AB-test by comparing the survival curves of different groups: it tests the null hypothesis that the survival curves are equal and can be used to see if the difference is statistically significant.

**Results**: Player churn seems to be initially high in free-to-play games as players try out the game and find it does not interest them, but decreases slowly over time. For example, in the version 1.18 we initially had 0.6 churns/h, halved to 0.3 churns/h during the first 4 hours, then stabilized to 0.2 churns/h from 10 hours onwards. We see that later versions managed to decrease initial churn, and the latest version may have just slightly better long term retention. There is a large difference in median and mean playtime, for example in the 1.18 version mean playtime was 2.41 hours and median 0.77 hours. A statistically significant improvement can be found from version 1.11, but no significant difference exists between versions 1.15 and 1.18.

Parametric approaches are more powerful if the specified model is correct, in the sense that they require less data and can be extrapolated to predict outside observed playtimes. However, the results can be significantly wrong if the assumed formula does not hold. The Weibull distribution is a reasonable but not perfect approximation of playtime in our data. Nonparametric approaches are considered as more robust, because they do not depend on specifying a correct model and instead base inferences on the data itself. Confidence intervals for these estimates are informative in data constrained industry applications. For example, a limited budget for user acquisition implies a limited sample size. One may wish to assess statistical significance before committing resources to big changes.

**Author's contribution**: The author was responsible for the study. Remaining authors provided comments on a draft of the paper and acted as supervisors.

#### 4.1.1.2 A/B-test of retention and monetization using the Cox model

**Motivation**: Many app developers use an iterative development model based on AB-testing. Different versions of an in-development app are provided to users, and a test is then done to determine which version performs best. Targeted advertising has also become common, because developers have a degree of freedom in choosing the users they wish to market to. In both of these cases, we would like to know which app version or user cohort is the best choice. There are several challenges of doing a test on the data: 1) user acquisition costs money so the sample size may be small 2) analysis needs to be performed during development, so there may be limited follow-up times, and often they differ between users 3) many users have not yet churned and we may not know who have or haven't, so we do not know final values of many metrics 4) in different user cohorts there may also be differences between platforms, countries, etc. which can bias analysis if not accounted for. We considered two important metrics: the retention rate and the purchase rate. Our goal is to develop an AB-test allows to estimate these metrics in censored data and test if the differences between groups are statistically significant.

**Data and methods**: We use data from an in-development mobile game, which was previously called Hipster Maze, but renamed to Hipster Sheep later. The game has used paid marketing to acquire users during development in order to track development progress and evaluate alternative designs in terms of user engagement and monetization. Users have been followed for limited time periods and acquired at different times, so their follow-up times are limited and may differ. We refer to this as data censoring.

The single event Cox model has been used to analyse player churn, but it has limited applicability because it requires knowing which players have churned. We propose using recurrent event Cox model, which makes it possible to analyse the retetion rate and the purchase rate without the churn problem. However, the fact that players churn implies that we need to use robust confidence intervals instead of the default ones based on a Poisson process assumption. The Cox model does not assume any parametric form for the metrics, but does assume that they are proportional in different groups. The model provides estimates of effect sizes and confidence intervals relative to the baseline group, implying how much better or worse the other choices are. The model is in fact a full regression model: we can take into account cofounding variables and get the effect of independent variables. We evaluate the proposed method in three real world game development scenarios: 1) we test how game progression speed affects player retention by randomly assigning players into normal/faster/fasters cohorts 2) we evaluate development progress in terms of retention and monetization by comparing successive game versions 3) we find the best user cohorts by estimating the effect of platform, country, and game version.

**Results**: The comparison of different game progression speeds implies that the ’faster’ cohort has the best retention. In this cohort the retention rate was about 18% higher, with no difference between ’normal’ and ’fastest’, but the differences are not yet statistically significant at the 95% confidence level. The comparison of different game development versions suggest that retention rate improved in versions 1.11, 1.15, 1.18, 1.2x and remained relatively stable in versions 1.31, 1.32, 1.33, 1.35. However, it seems that developers successfully focused on monetization, because the purchase rates were significantly improved in 1.31, 1.32, 1.33, 1.35. Android and ios have very different retention and purchase rates and were compared separately. We include the following independent variables in the analysis of marketing to different user groups: platform (ios/android), country (US/GB/AU/NL) and game version (1.31, 1.32, 1.33, 1.35). We found very large differences in the platforms: the ios platform had half the retention but quadruple monetization. There were no practical differences in retention rates between countries, but there was a very large difference in purchase rates: United Kingdom and Australia had quadruple but Netherlands only half the purchase rates. There is such a large variance between player session counts and so few purchases that it was difficult to establish statistical significance even with these sample sizes. This suggests game developers should exercise caution in interpreting the results from any AB-test.

**Author’s contribution**: The author was responsible for the study. Remaining authors provided comments on a draft of the paper and acted as supervisors.

#### 4.1.1.3 Measuring Player Retention and Monetization using the Mean Cumulative Function

**Motivation**: Products and services sold digitally have become an important source of revenue for many companies. Games in particular account for a majority of revenue in the app stores, and the field of game analytics investigates how their data can be analyse. The expected playtime and lifetime value are two very important metrics. Expected playtime can be used to measure how a game retains players and provides opportunities for monetization. Longer playtimes imply more friend invites, advertisements and premium content that could be purchased. Expected lifetime value (LTV) is a direct measure of profitability, because the investment return is the difference of the LTV and the acquisition cost of a user. Analytics used in the industry places high time-demand on the metrics. For example, app developers can estimate the profitability of an app by acquiring a group of users through paid marketing. They then follow the users for a time and record their purchases. They want to estimate the profitability from the data, but it is likely that many users still continue using the product. If they were acquired at different times they also have different follow-up lengths. We say that the data is censored, and we cannot calculate the metrics from it without bias. In practise, the developers would like to have a simple and robust method to estimates the expected value accumulated up to a given follow-up time, which is what we propose. Our method works for any expected value, so it can be used on a variety of metrics.

**Data and methods**: We use two data sources to evaluate the applicability of the method. The first data set is based on real game development scenarios in the mobile game Hipster Sheep. We use 10 000 players in an iOS beta test, 1800 players in 1.18 progression speed test, and 3200 players in 1.15 vs 1.18 version upgrade test. The second is based on the public 2016 ACM Internet Measurement Conference Steam data set. We processed the daily snapshot data to simulate a game launch scenario, by including only games that had a release date within the daily sampling window and players who started to play during that time. This resulted in 8 games with 240 to 1896 players per game.

Our method is based on defining the data set as recurrent events and associated costs, which have been investigated in reliability engineering and biostatistics. This definition corresponds to a problem in the field of recurrent event survival analysis. We further show that the so called robust nonparametric methods developed there can be applied to games, where the players do not fill parametric assumptions and their churn status is unknown. As a result of our search, we present and evaluate the following tools for game analytics. The mean cumulative function (MCF) can be used to estimate the expected value at time T, for example the expected LTV accumulated at each follow-up time. The robust variance formula provides confidence intervals for the MCF. The pointwise comparison can be used to calculate the difference of MCFs, for example compare players in different countries to find the country with the best monetization. Finally, the equality test is a straightforward test of the null hypothesis that the MCFs are equal, which can be used to verify if the differences found are statistically significant. The MCF is a generalization of standard statistical methods and current metrics used in the industry. Our methods are equal to performing the analysis with standard statistical methods when all players share the same censoring time. Furthermore, the MCF corresponds to the retention rate statistic if we estimate the expected number of distinct days played at.

**Results**: We have four different experiments to evaluate the proposed method. In the first experiment, we show that the MCF is a better tool for censored data than two simple approaches: exclude data to obtain a subset with a shared censoring time or ignore the problem altogether and assume missing data as zeros. Compared to our approach, the first approach is also unbiased but has larger variance, and the second approach is biased but has similar variance. The MCF comparison test is better than the standard test used on a subset of data, because it allows the entire data set to be used. In the second experiment, we compared the retention rate and our method. We found that in the 1.18 progression speed test, the retention rate was very noisy compared to our method, because the results depended on the day we picked whereas our method always identified the best version. In the 1.15 vs 1.18 version test, we found that results may differ whether we use short term or long term retention, whereas our method provided an unambiguous answer of what these retentions imply in total. In the third experiment, we used the MCF to calculate the expected lifetime value to evaluate whether there are benefits to generalizing the standard metrics. We found a large difference in LTVs, whereas the standard retention measurements did not differ. Fourth, we investigated how well these findings generalize to other games. We calculated the expected playtime in 8 different games with 31 days of follow-up using the MCF, and compared the estimates to SteamSpy data that gives the true playtimes. With one exception, a mobile game with few players and a very long tail, our results predict the final playtimes well. Our method is useful for a large variety of game genres and can quickly determine the relative order of the games. However, the usefulness may depend on the characteristics of the data set. Greater variance in playtimes implies that one needs to use larger sample sizes, and lower churn rates imply that one needs longer follow-ups, to estimate the final playtime.

**Author's contribution**: The author was responsible for the study. Anne-Maarit suggested a structure for an earlier draft of the paper. Remaining authors provided comments on a draft of the paper and acted as supervisors.

## 4.1.2 Peer-to-Peer lending

### 4.1.2.1 Predicting expected profit in on-going peer-to-peer loans with survival analysis based profit scoring

**Motivation**: P2P lending is a modern financing solution where an online platform connects individual borrowers and lenders. The borrowers make a loan application with their information and the lenders can choose the loans they invest in based on their investment criteria. The loans typically have a high default risk and high interest rates without any collateral. Past approaches have been based on analyzing credit risk, where the goal is to classify loans into different categories based on the risk of default. However, the high interest rates may or may not in fact compensate for the defaults, so selecting loans based on the classification does not necessarily answer the question. Investors ultimately wish to know the expected profit, which depends on both default risk and the interest rate. However, there is a problem with censored loans, because most loans are recent and have been scheduled to be repaid over many years. The full payment history of many loans has not been observed, so how can we calculate the profit in these loans? Our study develops a model that predicts the expected profit in the loans using survival analysis, which allows the analyst to use all loans in the modelling process, no matter how recent.

**Data and methods**: We use a public data set of 65675 loans and 112 features from the Bondora P2P Lending platform. The data set was filtered to include loans from January 2013 to October 2018, because Bondora's rating was implemented in this period and we compare our approach to it. The data describes demographic and financial information of the borrowers, the current state of the loans, and their payment behavior. The status of each loan is either current, repaid or default, where a loan is said to be in default if it is over 60 days late from last scheduled payment. Our data consisted of 36.5% of defaulted loans, 41.8% current loans, and 21.7% of repaid loans, which indicates that the loans are high risk and censoring is a significant problem. The interest rates have gone down from 25% in 2013 to 12% in 2018, which suggests that recent loans are different from the historical ones.

Our method has two stages: First, we predict monthly default probabilities using survival analysis. Then we use the predicted default probabilities, a loss given default estimate and the interest rate to predict the loan profit using a discounted cash flow (DCF) analysis of the expected monthly payments. If a loan defaults, depending on the platform we may either lose our money, obtain a percentage of principal as it is sold to a collection agency, or have to estimate the loss. For a constant default rate, we in fact have a simple formula $i = (1-h)I + hD$, where $h$ is the monthly default probability, $I$ is the interest rate, and $D$ is the loss given default. In this study, we assume a constant default rate and use Bondora's estimate for the loss given default.

**Results**: We divided the data into 20% test set and 80% training set. We measure the mean squared error (MSE) and area under the ROC curve (AUC) of monthly default and profit predictions, compared to what actually happened in the intervals. We have very low MSEs in the test set, 0.029 for default and 0.017 for profit, which are due to the fact that the actual quantities are very small to begin with. The AUC of predicting whether a loan defaults or not each month is 0.71, suggesting that the model does significantly better than random. We next analysed how interest rates, default rates, and predicted profits correlate. While most interest rates are between 25 and 35%, most of the predicted profits are between 5 and 15%, and there are many loans with an expected loss. This means a significant amount of the interest goes into covering the defaults. There is a strong correlation between the predicted monthly default rate and the interest rate, but the results suggest that it may be possible to choose more profitable loans where the interest more than compensates the expected defaults. Finally, we tested the model's ability to select more profitable loans by dividing the loan data into 10 portfolios from the best to worst loans. We can use different criteria to choose the best portfolio, and we had an average profit of 20% based on selecting the most profitable loans, 12% based on lowest credit risk in terms of predicted monthly default probabilities, and 10% based on the Bondora credit rating. The experiment suggests that our approach outperforms other approaches in predicting the expected profit.

**Author's contribution**: Ajay was responsible for experiments and the author of this thesis came up with the profit calculation idea. The paper was written in collaboration.

#### 4.1.2.2 Predicting profitability of peer-to-peer loans with recovery models for censored data

**Motivation**: Peer-to-peer (P2P) lending is the practice of lending money between individuals through an online platform. The borrowers apply for a loan with their financial information and the lenders bid for the loans by offering an interest rate. The lenders use the available information to decide who to offer loans to and at what price. The interest rates are ultimately based on the supply and demand of loans, but implicit in them is an assumption about the default risk and the loss given default. Setting an appropriate interest rate can be a challenging problem. The platforms attempt to help by providing statistics about past loans and often provide a rating model that categorizes low and high risk loans. However, an investor wants to ultimately know the expected profit of a loan. The profit can be calculated with discounted cashflow (DCF) analysis if one knows all of the payments in a loan. This is not possible because many loans are censored, meaning that are still ongoing and we do not know the future payments. How can we train a model when we do not know the profits in the training set? Simple solutions are problematic. Excluding ongoing loans creates bias because it removes loans more likely to survive. Assuming future payments are made in full or not at all creates an optimistic or pessimistic bias. Limiting the analysis to old loans that have had the possibility to be observed in full may not accurately predict profits in recent loans. In this study we therefore develop a model that uses censored loans to predict the expected future payments in any loan.

**Data and methods**: We use a public data set of P2P loans from the Bondora platform. At the time of writing, we had 119341 loans with 112 features. We limit the analysis to loans issued after 2013. The features describe borrower information, current loan status, loan payment history, and Bondora's own predictions about the loan. Each loan is said to be either repaid, current or late. A loan defaults if it is 60 days past its due payment. We extend the previous study by providing two models: a new default model that does not assume constant monthly default rates and a new loss given default (LGD) model that predicts the monthly recoveries thereafter in censored data. With these two models, we can calculate the expected monthly payments and the resulting profit. The default model is based on predicting whether a loan defaults in each monthly interval with logistic regression. The loss given default model is based on predicting the recovery payments as a percentage of principal in each monthly interval with least squares regression. We calculate the profit using DCF analysis, where profit is the required rate of return which makes the present value of the predicted monthly payments equal to the loan amount.

**Results**: Both models we presented are linear models: logistic regression for defaults and least squares regression for recoveries. We interpreted the coefficients of the models and found intuitive results: smaller default rate and higher recovery rate is predicted by earlier loan issuance year, better country and credit rating, high education, home ownership, stable job, lower debt load, existing customer, etc. We then compared our loss given default predictions to Bondora's. We calculated the the loss given default with DCF analysis assuming a 10% profit requirement. There is some correlation to Bondora's estimates, but we have smooth LGDs with 0.75 average compared to Bondora's discrete values with 0.55 average. We found a strong correlation between the interest rate, predicted default rate, and the predicted loss given default. This correlation means that individuals attempt to set the interest rate higher to compensate for a higher default probability or higher loss given default. However, these correlations are not perfectly aligned with the predicted profit, implying that there may be loans with higher or lower profits. We measured the prediction accuracy of the recovery model with mean error (ME) and mean squared error (MSE) by dividing the loans into 25% test set and 75% training set. The model is unbiased, meaning that ME is consistently zero, and the MSE is very low also. Finally, we compared our model to Bondora's LGD model and rating model by dividing the test set loans into 8 different portfolios based on LGD or rating values.

Our model correctly orders the portfolios and in fact predicts the actual LGDs very well, whereas the other two approaches do not perform better than random.

**Author's contribution**: The author was responsible for the study. Ajay wrote the related work section. Tapio acted as a supervisor.

### 4.1.3 Unemployment

#### 4.1.3.1 Predicting Unemployment with Machine Learning Based on Registry Data

**Motivation**: Unemployment is a significant issue for societies and individuals alike. To understand how an individual experiences unemployment, we need to consider the full labour market history of each person. This history consists of spells unemployment that alternate between spells of non-unemployment. One generally views the event of exiting unemployment as a positive event and entering unemployment as a negative event, where a major interest is to reduce the total amount of unemployment. The total time in unemployment is determined by both. The person spends less time in unemployment if the unemployment spells are short or the non-unemployment spells are long. In the study, we model the full labour market history of individuals and evaluate the predictive ability of the model in three separate tasks. These tasks are to predict the probability of exiting unemployment, entering unemployment and being unemployed at a given time.

**Data and methods**: Our data set was collected in the ELY Centre of Southwest Finland using the official unemployment registry. Individuals need register in order to receive unemployment benefits, so the registry can be considered to contain every unemployed individual. The unemployment registry was sampled at the end of the month during 2013-2017 to record individuals who were in the registry. The entries were processed into a data set where each observation has the unemployment status (in registry, not in registry, censored) with the person's information (gender, age, work experience, general level and field of education). From the 60 month follow-up and 128 937 persons, we took a random sample of 20 000 persons.

We model the unemployment status of a person as a discrete time Markov chain. The model is defined by the transition rates of unemployment exit and entry, which are assumed to be person specific. If the transition rates are constant, the probability of being unemployed converges over time to a steady state probability. This is the predicted lifetime unemployment of an individual. The transition rates can be learned based on two sources: person information and their labour market histories. The unemployment entries and exits are then influenced by the observed information, latent information inferred from the unemployment history, and randomness in finding or exiting a job. We compare three models for unemployment prediction.

The linear model (LM) includes only the covariates but not the subject specific intercepts. The linear mixed effects (LME) model includes the covariates and subject specific intercepts as random effects sampled from multivariate normal distribution. The linear machine learning (LML) model includes the covariates and subject specific intercepts as model parameters which are made well-conditioned by regularization.

**Results**: We evaluate the models on three different prediction tasks: predict the risk of unemployment exit (Exit), risk of unemployment entry (Entry) and the unemployment status (Prevalence). The unemployment status can be predicted as the long-run unemployment prevalence, or the transition probability in the Markov chain given an initial state. We evaluate the time-stratified AUC of three different models, measured separately in two tests sets. The training set (Train) consists of 10000 persons in years 2013-2016. The first test set (Test) consists of same persons in 2017, and the cold start test set (Cold) contains 10000 different persons in 2013-2017.

We make several interesting findings. Cold start prediction performance is the same for all three models, because the LME and LML models cannot learn person specific intercepts for persons who are not in training data. Performance based on only observed features is modest but better than random. Test set performance for future prediction of same persons are significantly improved in the LME and LML models, which indicates there are significant unobserved features that can be learned from labour market histories using the person specific rates assumption. Training set performance is overoptimistic in the LME and LML models, because they have a high degree of flexibility to fit the data. The Markov Chain assumption significantly improves prediction performance in the prevalence model: the closer the previous known state is in time, the more accurately we can predict the current state. Over time these converge to the lifetime unemployment prevalence prediction. The statistical model and the machine learning model result in very similar predictions. The covariates have intuitive effects that are consistent with previous findings, but there is still considerable variation between individuals that can be used to improve predictions by considering their past unemployment histories.

**Author's contribution**: Author was responsible for the study. Tapio acted as a supervisor.

### 4.1.4 Game recommendation

#### 4.1.4.1 Content Based Player and Game Interaction Model for Game Recommendation in the Cold Start setting

**Motivation**: Many different digital platforms sell games to players and they have the problem of recommending interesting new games to play. The platforms are interested in accurate recommendations because they can increase sales and player retention. From an analytics perspective, game developers and publishers are both interested in knowing the reasons why players like certain games in the platform. To solve this problem, game recommendation is a natural application of recommender systems. Recommender system models can be divided into collaborative filtering (CF) or content based (CB). The platforms have large databases of past player and game interactions, which the CF methods utilize. They also store information about players and the game content, which can be used as features in CB methods. Game information can be tags, genres, description, screenshots, etc. and player information could be based on implicit behaviour or asking players explicit questions about their preferences. These two approaches are complementary, and hybrid recommenders can be used to utilize both sources of information or combine the predictions.

Collaborative filtering is typically better in terms of predictive accuracy, unless predictions are required for players or games with few interactions. The case of no interactions is called the cold start setting and collaborative filtering cannot predict at all. While most research has considered historical data sets of observed player and game interactions, new games and players appear all the time and we need to be able to predict for them before we observe interactions. We therefore define four different settings and investigate new models in each setting: past games and past players with interactions (Setting 1), new games without players (Setting 2), new players without games (Setting 3), and both new games and new players simultaneously (Setting 4).

**Data and methods**: We use a private data set based on questionnaire answers of favourite games and playing motivations. The data set contains 15894 players and 6465 games with a total of 80916 game likes. Player features are obtained by asking a randomized set from 61 questions that describe playing motivations (’Engaging in battle’, ’Exploding and Destroying’, …) on a Likert scale of 1 to 5. Game features are obtained by mining the Steam and IGBD platforms for 379 game tags as a binary vector. The rating matrix $R_{i,j}$ contains a value 0 or 1 for every player and game pair depending on whether player liked the game.

We investigate the task of Top-N recommendation, which means that a score is predicted for every pair and $N$ games with the top score are recommended to the player. This is a ranking task, so we evaluate the methods with two ranking metrics. Precision@k counts the fraction of played games and nDCG@k penalizes the ranks of played games based on where they appear in the recommendation list of length $k$. We use kNN and SVD as baseline collaborative filtering methods. We present a new collaborative filtering method based on the multivariate normal distribution and three content based methods Tags, Questions, Tags X Questions. The content based methods can be seen as a special case of the SVD, where latent feature vectors have been restricted to use the provided features. In the Tags model, game tags are used as features and we learn a tag interaction vector for every player. In the Questions model, player questionnaire answers are used as features and we learn a question response vector for every game. In the Tags X Questions model, we use both features and learn the interactions between game tags and player questionnaire answers. All three models are linear models and we present computational shortcuts to train them in a reasonable time.

**Results**: Setting 1 and Setting 3 have the highest accuracy of predictions, Setting 2 appears more difficult, and Setting 4 is the most difficult. In Setting 1, collaborative filtering outperforms content based models and the MVN model delivers the highest accuracy. In Setting 2, only the Tags and Tags X Questions models are able to generalize to new games, and the more flexible Tags model has higher accuracy. In Setting 3, only the Questions and Tags X Questions models are able to generalize to new players, and the more flexible Questions model has higher accuracy. In Setting 4, only the Tags X Questions is able to generalize to new games and players simultaneously. The results indicate an important trade-off between generalization ability and higher accuracy: it is better for generalization to use the provided features, but higher accuracy can be obtained with latent features if they can be learned from the setting.

We also interpreted the MVN correlation matrix and the interaction coefficients of the three content based linear models. We found that the correlations were very intuitive and specific: very similar games are recommended. Player responses to game tags and game response to player preferences are logical. The Tags model learns to recommend games that have similar tags to the games the player has played, and the Questions model learns to recommend games to players that have similar preferences to those players who played the game. The Tags x Questions models model learns an interaction matrix by matching game preferences and game tags that are similar. However, the game preferences are asked with very generic questions and the quality of recommendations is therefore generic as well. Popularity bias was visible in recommendations, but it could be easily removed. This results in worse accuracy but better subjective quality. Each model performed the best and was the most useful in the setting for which it was designed, and therefore all of the models can be useful in different prediction tasks.

**Author’s contribution**: Author was responsible for the study. Aki and Jukka provided the data set and commented the results, and Tapio acted as a supervisor.

## 4.2 Research results

This section summarizes the research questions and discusses the main findings.

### 4.2.1 (RQ1): How to use machine learning and mathematical models to answer important business questions in censored data sets?

The first research question asked if it was possible to develop models that can be applied to censored data sets in different domains. The data sources included mobile game logs, public peer-to-peer lending histories, local unemployment registry, and player questionnaire. These data sources have a similar problem; they are censored in the sense that some of the data is missing. We investigated models that were specifically designed for every application and considered the fact that the data was censored in their formulation. The main benefit is that these models can be applied immediately when the data is being gathered instead of waiting for years to gather complete data sets.

The solutions are based on modelling the problem in three stages: 1) deconstruct the data set to pairwise data 2) train machine learning on the pairwise data and predict the missing values 3) reconstruct the answer from the pairwise predictions. This approach combines a mathematical formulation of the problem and specialized machine learning models: a mathematical understanding of the problem is required to implement the stages 1) and 3), and a machine learning model is needed for the stage 2). The different machine learning models can be seen as instances of a single problem where the data set consists of user and time pairs. However, the practical models use specialized problem specific formulations of how the intercepts and features are defined for the user and time pairs.

The first model solved the problem of measuring retention and monetization in data sets where customers can have different follow-up times. For example, we measured total playtime and lifetime value in mobile games. The new method broadened the application of analytics, since it is possible measure in real-time the expected value of any metric in any data set. The second model solved the problem of predicting profitability of peer-to-peer loans, where many loans in the data set are still on-going and we do not know their full payment histories. Previous approaches have either used historical data sets from years ago or answered simpler questions. Predicting loan profit accurately is the most important questions investors have and the new method means that is is possible even in current data sets. The third model describes an individual’s unemployment status as a Markov chain. The model makes it possible to model all aspects of the unemployment experience and predict future unemployment from demographic features or past unemployment. The fourth model investigated a game recommendation problem, where some player and game like pairs are given and the task is rank the remaining games for every player. We investigated different models that would make it possible to recommend games in four different settings when predictions are required for known players and games, new games, new players, or both simultaneously.

The results indicate that it is possible to develop accurate models for censored data. It is not necessary to limit the analysis to historical data sets where we have collected fully observed answers, or analyse only time-to-event outcomes that are considered in standard survival analysis, for example. These models can describe the underlying problem and censoring does not bias the results. The answers are more accurate when censoring is taken into account. On the other hand, in depth domain knowledge and mathematical understanding is required to deconstruct and reconstruct the problem as pairwise data, and then formulate an appropriate model for such data.

### 4.2.2 (RQ2) How to obtain unbiased model evaluation when the data is not independent and identically distributed?

The primary goal of machine learning is to give accurate predictions. A machine learning method can be seen as a black box that produces an output for a given input, and we evaluate the method based on how well it predicts for new data. It is therefore important to obtain accurate estimates of how well the model predicts in reality. In the standard setting we can split the data set into training and test sets: the model is fitted on the training set and evaluated on the test set. However, this relies on the assumption that the observations are independent and identically distributed (i.i.d.), which is not necessarily valid for pairwise data when many observations belong to the same user or time point.

Consider a pairwise data set that consists of user and time pairs. The prediction task was divided into four different settings, where predictions can be required for: 1) known users and known times, 2) new times, 3) new users, 4) new users and new times. The standard training and test set split measures the model performance in the first setting, and the correlations between observations can result in more accurate predictions than would occur for new users or new times. We explained how four different types of training and test set splits can be created to accurately measure the performance in each setting, which is the answer to the research question.

The overall prediction performance was reasonably good in our applications. It was significantly better than random but there was a large amount of random variation in the output that could not be explained by the features in every problem. We found that the prediction performance was very different in different settings. It is typically considerably more difficult to predict outside setting 1, where setting 4 was the most difficult. In the retention and monetization problem, the goal was more to measure the profitability of a given acquisition method than predict for individual customers. In peer-to-peer lending, we obtained considerably better profits by picking the most profitable loans as predicted by our model. In the unemployment problem, we were able to predict the future unemployment status of known persons reasonably well. In games, all four settings are generally valid based on the recommendation problem. The predictions were quantitatively and qualitatively very good in the first setting, but not so good in other settings.

It should be noted that predictive accuracy is not always straightforward or only measure of real-world utility of the method. We give a couple of examples. The goal can be to measure the profitability of acquiring different customer groups rather than to predict for individual customers, because we may not be able to pick the customers individually even if we know that some of them have larger purchases. Similarly, the unemployment office can prioritize services to unemployed people based on their risk of exiting unemployment; the fact that we cannot predict perfectly of who exists doesn’t mean the method is not useful. There is a lot of random variation that probably cannot be modelled, and the absolute performance is not a direct measure of the utility of the method. In the game recommendation task, the accuracy of the recommendations may not correlate how useful players find the recommendations. It is often possible to get a good accuracy by recommending the most popular games to everyone, even though players typically want to see games that are closely related to the games they like and they have not heard about.

# 5 Conclusions

## 5.1 Summary of the thesis

Machine learning is a field of data science that develops algorithms for problems where there is a clear predictive task and accuracy of the methods is a primary concern. However, practical applications sometimes cannot afford to wait for years of follow-up gather a data set of correct answers. In this case, the correct output is not fully observed, i.e. it is censored, and the problem needs to be formulated in a new way. The goal of this thesis was to investigate how machine learning can be applied to censored data sets in different applications to answer the most relevant business question.

Chapter 1 motivated the research goal. A simple example of LTV prediction was used to illustrate a censored data set. We discussed the basic steps in a data science project and how models for censored data impact the different stages. Chapter 2 formulated machine learning as an instance of empirical risk minimization and gave a mathematical definition of the learning problem. The relationship to statistics was briefly discussed. Machine learning for censored data was claimed to represent a pairwise learning problem, and various special aspects of pairwise prediction were discussed. Finally, four different practical applications of machine learning that implement the framework were presented. Chapter 3 briefly summarized the studies: motivation, data, methods, and study results. Finally, the research questions were revisited and results discussed in Chapter 4.

Earlier chapters discussed how machine learning can be used to give accurate predictions in different practical applications that have censored data. Models that can be applied to censored data can be used for real-time predictions and do not required long follow-up periods, which can be a major practical benefit. We reformulated each data set as a pairwise learning problem and defined a simple machine learning model for the pairwise data in each application. The models had a good predictive performance, which was typically somewhere in the middle from random predictions to perfect accuracy. It was not necessary to give perfect answers in these applications and any incremental benefit to random answers or simple baselines can be seen as beneficial. The validation methods for pairwise data needed to take into account the fact that every observation is a user and time pair: real world prediction tasks may require predictions for new users or new times. Indeed, the accuracy in these tasks was found to be significantly different, which implies that validation steps need to consider pairwise structure of the data.

To summarize, the thesis presented models in four different applications:

1. Retention and monetization: Many companies use analytics to track the success of their product with different metrics. The model measures how the expected value of any metric, such as lifetime value or total product use, accumulates as a function of follow-up length in censored data. The model can be used to implement real-time analytics.
2. Peer-to-peer lending: Investors make decisions on which loans to invest in based on their interest rate and credit rating, but ultimately they wish to know the profit. The model estimates the profit in censored data sets using DCF through two complementary models that incorporate the loan schedule: a default rate and a recovery rate model.
3. Unemployment: Understanding the dynamics of unemployment is an important research topic. The model describes the unemployment status of individuals in censored registry data as Markov Chains. The model predicts the probability of exiting, entering and being in unemployment. A lifetime individual unemployment prevalence is implied by the model.
4. Game recommendation: Games are recommended to players through many different channels, where some (player, game)-likes are known and used for recommending new (player, game)-likes. We investigated how four different models can make optimal recommendations in four different settings: collaborative filtering, new games, new players, and new players and games.

The experiments show that these models produce accurate predictions of the censored prediction targets. Simpler methods are limited on the type of questions they can consider; applied directly to censored data they produce inaccurate and less reliable results. We found interesting implications in many real world applications: the overall success of games in terms of playtime could be estimated very quickly when the game is launched, the profits in peer-to-peer lending may not be as high as though, that individual's lifetime unemployment is affected by both how fast they exit unemployment and how quickly they return to unemployment, and game recommendation is possible in every setting with different degrees of accuracy.

## 5.2 Future work

There are many possibilities for future work. We considered linear models, but more complicated models could be applied after the problem has been formulated as pairwise data. In particular, machine learning considers several non-linear models that have better performance for complex prediction tasks: kernel methods, random forests, neural networks, etc. The data sets in our problems consists of few high-level variables, so simple linear models are quite competitive. The linear models were formulated in a particular way in each task, and it would be possible to extend these models without considering general all-purpose models. For example, there could be second order interactions between the features or the effect of features could change over time.

Many other applications have the same problem, and it would be possible to apply the presented models to more data sets. While the pairwise formulation works in any similar task, there is no guarantee that the machine learning model is able to predict well in other tasks. Only studies that investigate a particular application can prove that the method works to predict in that task. Of course, the models can have important practical implications in the domain if they come to different conclusions when censored data is correctly modelled. For example, it was found that many new peer-to-peer loans might not be as profitable as the fully observed old loans many investors have trained their models with.

# Acknowledgements

I wish to thank my supervisors Tapio Pahikkala, Antti Airola, and Jukka Heikkonen. As the primary reason that lead me to doctorate studies, Tapio has been a motivating supervisor blessed with a great deal of patience, understanding and exotic thrills for research. Antti and Jukka have been very helpful and kind in many practical aspects of research. Without them it is possible this thesis would not have been completed, but in the hypothetical case that it were it would look quite close to the current thesis. I was given a wide latitude to work with my own ideas, in both good and bad, and hopefully at the end of it there was a good one or two. The Russian missile has found a target and the generals are cheering! Finally, in the day to day operations of our room 452B -the most likely exoplanet to have life!- uncountable unforgetful moments of education and serotonin exchange were shared with my fellow researchers Parisa Movahedi, Paavo Nevalainen, Jonne Pohjankukka, Ileana Montoya Perez, Jussi Toivonen, Pekka Naula, Anne-Maarit Majanoja, and Riikka Numminen. If I forgot someone from the list, don’t worry you were the most important. Research would not be the same without you all!

5.5.2021
*Markus Viljanen*

# References

[1] M. Hilbert and L. Priscila, “The world’s technological capacity to store, communicate, and compute information,” *Science,* vol. 332, no. 6025, pp. 60-65, 2011.

[2] D. Kahneman, Thinking, fast and slow, Macmillan, 2011.

[3] R. Dawes, “The robust beauty of improper linear models in decision making,” *American psychologist,* vol. 34, no. 7, p. 571, 1979.

[4] M. Jordan and T. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science,* vol. 349, no. 6245, pp. 255-260, 2015.

[5] A. McAfee, “Big data: the management revolution,” *Harvard business review,* vol. 90, no. 10, pp. 60-68, 2012.

[6] V. Dhar, “Data science and prediction,” *Communications of the ACM,* vol. 56, no. 12, pp. 64-73, 2013.

[7] L. Kurgan and P. Musilek, “A survey of Knowledge Discovery and Data Mining process models,” *The Knowledge Engineering Review,* vol. 21, no. 1, pp. 1-24, 2006.

[8] D. G. Kleinbaum and M. Klein, Survival analysis, Springer, 2010.

[9] M. Rausand and H. Arnljot, System reliability theory: models, statistical methods, and applications, John Wiley & Sons, 2003.

[10] O. Aalen, B. Ornulf and G. Hakon, Survival and event history analysis: a process point of view, Springer Science & Business Media, 2008.

[11] E. P. Wigner, “The unreasonable effectiveness of mathematics in the natural sciences,” *Mathematics and Science,* pp. 291-306, 1990.

[12] V. Vapnik, The nature of statistical learning theory, Springer science & business media, 2013.

[13] C. Vladimir and F. Mulier, Learning from data: concepts, theory, and methods, John Wiley & Sons, 2007.

[14] I. Steinwart and A. Christmann, Support vector machines, Springer Science & Business Media, 2008.

[15] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer and N. D. Lawrence, Dataset shift in machine learning, The MIT Press, 2009.

[16] S. W. Knox, Machine learning: a concise introduction, John Wiley & Sons, 2018.

[17] A. DasGupta, Probability for statistics and machine learning: fundamentals and advanced topics, Springer Science & Business Media, 2011.

[18] X. Zhu and A. B. Goldberg, Introduction to semi-supervised learning, Morgan & Claypool publishers, 2009.

[19] M. Hollander, A. W. Douglas and C. Eric, Nonparametric statistical methods, 2013: John Wiley & Sons.

[20] A. Faul, A Concise Introduction to Machine Learning, CRC Press, 2019.

[21] T. Hofmann, S. Bernhard and S. Alexander, "Kernel methods in machine learning," *The annals of statistics,* pp. 1171-1220, 2008.

[22] B. Schölkopf, H. Ralf and A. Smola, "A generalized representer theorem," in *International conference on computational learning theory,* Berlin, Heidelberg, 2001.

[23] I. Steinwart, "Consistency of support vector machines and other regularized kernel classifiers," *IEEE transactions on information theory,* vol. 51, no. 1, pp. 128-142, 2005.

[24] C. Molnar, Interpretable Machine Learning, Lulu.com, 2020.

[25] T. Nguyen and S. Sanner, "Algorithms for direct 0–1 loss optimization in binary classification," *International Conference on Machine Learning,* 2013.

[26] A. Ben-Tal and A. Nemirovski, Lectures on modern convex optimization: analysis, algorithms, and engineering applications, Society for industrial and applied mathematics, 2001.

[27] L. Breiman, "Statistical modeling: The two cultures (with comments and a rejoinder by the author)," *Statistical science,* vol. 16, no. 3, pp. 199-231, 2001.

[28] D. Bzdok, N. Altman and M. Krzywinski, "Points of significance: statistics versus machine learning," *Nature Methods,* vol. 15, no. 4, p. 233–234, 2018.

[29] J. K. Lindsey, Parametric statistical inference, Oxford University Press, 1996.

[30] L. Wasserman, All of statistics: a concise course in statistical inference, Springer Science & Business Media, 2013.

[31] R. J. Cook and J. Lawless, The statistical analysis of recurrent events, Springer Science & Business Media, 2007.

[32] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä and L. E. Meester, A Modern Introduction to Probability and Statistics: Understanding why and how., Springer Science & Business Media, 2005.

[33] K. Murphy, Machine learning: a probabilistic perspective, MIT press, 2012.

[34] C. Bishop, Pattern recognition and machine learning, Springer, 2006.

[35] D. M. Hawkins, ”The problem of overfitting," *Journal of chemical information and computer sciences,* osa/vuosik. 44, nro 1, pp. 1-12, 2004.

[36] F. Jerome, T. Hastie and R. Tibshirani, The elements of statistical learning, New York: Springer, 2001.

[37] A. Ø. Per Kragh, N. Borgan, H. Lid, A. Elja, S. Jon and A. Odd, "Counting process models for life history data: A review [with discussion and reply]," *Scandinavian Journal of Statistics,* pp. 97-158, 1985.

[38] E. Kaplan and M. Paul, "Nonparametric estimation from incomplete observation," *Journal of the American statistical association,* vol. 53.282, pp. 457-481, 1958.

[39] O. Aalen, "Nonparametric inference for a family of counting processes," *The Annals of Statistics,* pp. 701-726, 1978.

[40] R. L. Prentice, J. D. Kalbfleisch, A. V. J. Peterson, N. Flournoy, V. T. Farewell and N. E. Breslow, "The analysis of failure times in the presence of competing risks," *Biometrics,* no. 541-554, 1978.

[41] O. O. Aalen and S. Johansen, "An empirical transition matrix for non-homogeneous Markov chains based on censored observations," *Scandinavian Journal of Statistics,* pp. 141-150, 1978.

[42] P. K. Andersen, Ø. Borgan, R. Gill and N. Keiding, "Linear nonparametric tests for comparison of counting processes, with applications to censored survival data, correspondent paper," *International Statistical Review,* pp. 219-244, 1982.

[43] D. R. Cox, “Regression models and life-tables,” *Journal of the Royal Statistical Society: Series B (Methodological),* vol. 34, no. 2, pp. 187-202, 1972.

[44] P. K. Andersen and G. Richard D., “Cox's regression model for counting processes: a large sample study,” *The annals of statistics,* pp. 1100-1120, 1982.

[45] J. F. Lawless, “Statistical methods in reliability,” *Technometrics,* vol. 25, no. 4, pp. 305-316, 1983.

[46] J. F. Lawless, “The analysis of recurrent events for multiple subjects,” *Journal of the Royal Statistical Society: Series C (Applied Statistics),* vol. 44, no. 4, pp. 487-498, 1995.

[47] J. F. Lawless and N. Claude, “Some simple robust methods for the analysis of recurrent events,” *Technometrics,* vol. 37, no. 2, pp. 158-168, 1995.

[48] D. Y. Lin, E. J. Feuer, R. Etzioni and Y. Wax, “Estimating medical costs from incomplete follow-up data,” *Biometrics,* pp. 419-434, 1997.

[49] H. Bang and A. A. Tsiatis, “Estimating medical costs with censored data,” *Biometrika,* vol. 87, no. 2, pp. 329-343, 2000.

[50] R. J. Cook, J. F. Lawless and K.-A. Lee, “Cumulative processes related to event histories,” *SORT: statistics and operations research transactions,* vol. 27, no. 1, pp. 0013-030, 2003.

[51] W. Nelson, ”Confidence limits for recurrence data—applied to cost or number of product repairs,” *Technometrics,* osa/vuosik. 37, nro 2, pp. 147-157, 1995.

[52] S. L. Zeger and K.-Y. Liang, “Longitudinal data analysis for discrete and continuous outcomes,” *Biometrics,* pp. 121-130, 1986.

[53] S. L. Zeger, K.-Y. Liang and P. S. Albert, “Models for longitudinal data: a generalized estimating equation approach.,” *Biometrics,* pp. 1049-1060, 1988.

[54] M. Stock, T. Pahikkala, A. Airola, B. De Baets and W. Waegeman, “A comparative study of pairwise learning methods based on kernel ridge regression,” *Neural computation,* vol. 30, no. 8, pp. 2245-2283, 2018.

[55] F. Garrett, N. Laird and J. Ware, Applied longitudinal analysis, John Wiley & Sons, 2012.

[56] A. Airola and T. Pahikkala, “Fast Kronecker product kernel methods via generalized vec trick,” *IEEE transactions on neural networks and learning systems,* vol. 29, no. 8, pp. 3374-3387, 2017.

[57] Y. Park and E. M. Marcotte, “Flaws in evaluation schemes for pair-input computational predictions,” *Nature methods,* vol. 9, no. 12, p. 1134, 2012.

[58] C. Rudin, “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead,” *Nature Machine Intelligence,* vol. 1, no. 5, pp. 206-215, 2019.

[59] J. W. Hardin and J. M. Hilbe., Generalized estimating equations, New York: Chapman and Hall/CRC, 2003.

[60] J. Hardin and J. Hilbe, Generalized Linear Models and Extensions, Stata press, 2007.

[61] D. J. Teece, “Business models, business strategy and innovation.",” *Long range planning,* vol. 43, no. 2-3, pp. 172-194, 2010.

[62] E. B. Seufert, Freemium Economics, Boston: Morgan Kaufmann, 2014.

[63] R. Sifa, F. Hadiji, J. Runge, A. Drachen, K. Kersting and C. Bauckhage, “Predicting purchase decisions in mobile free to play games,” in *Proc. AIIDE Artificial Intelligence and Interactive Digital Entertainment Conf.,* 2015.

[64] W.-c. Feng, D. Brandt and D. Saha, “A long-term study of a popular MMORPG,” in *Proc. ACM SIGCOMM Network and System Support for Games Workshop,* New York, 2007.

[65] P.-Y. Tarng, K.-T. Chen and P. Huang, “An analysis of WoW players’ game hours,” in *Proc. ACM SIGCOMM Network and System Support for Games Workshop,* New York, 2008.

[66] M. Kwok and G. Yeung, “Characterization of user behavior in a multiplayer online game,” in *Proc. ACM SIGCHI Int. Advances in Computer Entertainment Technology Conf.,* New York, 2005.

[67] C. Chambers, W.-c. Feng, S. Sahu and D. Saha, “Measurementbased characterization of a collection of on-line games,” in *Proc. ACM SIGCOMM Internet Measurement Conf. USENIX Association,* 2005.

[68] D. Pittman and C. GauthierDickey, “A measurement study of virtual populations in massively multiplayer online games,” in *Proc. ACM SIGCOMM Network and System Support for Games Workshop,* New York, 2007.

[69] K.-T. Chen, P. Huang and C.-L. Lei, “Effect of network quality on player departure behavior in online games,” *IEEE Trans. Parallel Distrib. Syst.,* vol. 20, no. 5, pp. 593-606, 2009.

[70] B. G. Weber, M. Mateas and A. Jhala, “Using data mining to model player experience,” in *FDG Evaluating Player Experience in Games Workshop,* 2011.

[71] T. Debeauvais and C. V. Lopes, “Gate me if you can: The impact of gating mechanics on retention and revenues in Jelly Splash,” in *Society for the Advancement of the Science of Digital Games,* 2015.

[72] A. Isaksen, D. Gopstein and A. Nealen, “Exploring game space using survival analysis,” in *Society for the Advancement of the Science of Digital Games,* 2015.

[73] Á. Periáñez, A. Saas, A. Guitart and C. Magne, “Churn prediction in mobile social games: Towards a complete assessment using survival ensembles,” *Proc. IEEE International Conf. on Data Science and Advanced Analytics (DSAA),* 2016.

[74] T. Allart, G. Levieux, M. Pierfitte, A. Guilloux and S. Natkin, “Design influence on player retention : A method based on time varying survival analysis,” in *Proc. IEEE Computational Intelligence and Games Conf.,* 2016.

[75] R. Sifa, C. Bauckhage and A. Drachen, “The playtime principle: Largescale cross-games interest modeling,” in *Proc. IEEE Computational Intelligence and Games Conf.,* 2014.

[76] T. Debeauvais, C. V. Lopes, N. Yee and N. Ducheneaut, “Retention and progression: Seven months in World of Warcraft,” in *Proc. Int. Foundations of Digital Games Conf.,* 2014.

[77] F. Hadiji, R. Sifa, A. Drachen, C. Thurau, K. Kersting and C. Bauckhage, “Predicting player churn in the wild,” in *Proc. IEEE Computational Intelligence and Games Conf.,* 2014.

[78] J. Runge, P. Gao, F. Garcin and B. Faltings, “Churn prediction for highvalue players in casual social games,” in *Proc. IEEE Computational Intelligence and Games Conf.,* 2014.

[79] M. Viljanen, A. Airola, J. Heikkonen and T. Pahikkala, “A/B-test of retention and monetization using the Cox model,” in *Proc. AIIDE Artificial Intelligence and Interactive Digital Entertainment Conf.,* 2017.

[80] P. Rothenbuehler, J. Runge, F. Garcin and B. Faltings, “Hidden Markov models for churn prediction,” in *2015,* Proc. SAI Intelligent Systems Conf..

[81] M. Tamassia, W. Raffe, R. Sifa, A. Drachen, F. Zambetta and M. Hitchens, “Predicting player churn in Destiny: A hidden Markov models approach,” in *Proc. IEEE Computational Intelligence and Games Conf.,* 2016.

[82] E. Lee, Y. Jang, D. Yoon, J. Jeon, S.-i. Yang, S.-K. Lee, D.-W. Kim, P. P. Chen, A. Guitart, P. Bertens, Á. Periáñez, F. Hadiji, M. Müller, Y. Joo, J. Lee, I. Hwang and K.-J. Kim, “Game data mining competition on churn prediction and survival analysis using commercial game log data,” *IEEE Transactions on Games,* vol. 11, no. 3, pp. 215-226, 2018.

[83] M. Viljanen, A. Airola, J. Heikkonen and T. Pahikkala, “Playtime measurement with survival analysis,” *IEEE Transactions on Computational Intelligence and AI in Games,* 2017.

[84] T. V. Fields, ”Game industry metrics terminology and analytics case study,” tekijä: *Game Analytics*, London, Springer, 2013, pp. 53-71.

[85] W. B. Nelson, Recurrent events data analysis for product repairs, disease recurrences, and other applications, SIAM, 2003.

[86] J. Sviokla, “Breakthrough ideas: forget Citibank – borrow form Bob,” *Harvard Business Review,* no. February, 2009.

[87] C. Serrano-Cinca, B. Gutiérrez-Nieto and L. López-Palacios, “Determinants of default in P2P lending,” *PloS one,* vol. 10, no. 10, 2015.

[88] E. Lee and B. Lee, “Herding behavior in online P2P lending: An empirical investigation,” *Electronic Commerce Research and Applications,* vol. 11, no. 5, pp. 495-503, 2012.

[89] J. Li, S. Hsu, Z. Chen and Y. Chen, “Risks of p2p lending platforms in china: Modeling failure using a cox hazard model.,” *The Chinese Economy,* vol. 49, no. 3, pp. 161-172, 2016.

[90] F. Louzada, V. Cancho, M. J. de Oliveira and Y. Bao, “Modeling time to default on a personal loan portfolio in presence of disproportionate hazard rates.,” *Journal of Statistics Applications & Probability, An International Journal. J. Stat. Appl. Pro,* vol. 3, pp. 1-11, 2014.

[91] G. Andreeva, “European generic scoring models using survival analysis,” *Journal of the Operational research Society,* vol. 57, no. 10, pp. 1180-1187, 2006.

[92] A. Đurović, “Estimating probability of default on peer to peer market–survival analysis approach,” *Journal of Central Banking Theory and Practice,* vol. 6, no. 2, pp. 149-167, 2017.

[93] M. Malekipirbazari and V. Aksakalli, “Risk assessment in social lending via random forests,” *Expert Systems with Applications,* vol. 42, no. 10, pp. 4621-4631, 2015.

[94] C. Serrano-Cinca and B. Gutiérrez-Nieto, “The use of profit scoring as an alternative to credit scoring systems in peer-to-peer (P2P) lending,” *Decision Support Systems,* no. 89, pp. 113-122, 2016.

[95] S. A. Broverman, Mathematics of Investment and Credit, 5th Edition, ACTEX Publications, 2010.

[96] G. Tutz and M. Schmid, Modeling discrete time-to-event data, New York: Springer, 2016.

[97] E. Ernst and R. Uma, “Understanding unemployment flows,” *Oxford Review of Economic Policy,* vol. 27, no. 2, pp. 268-294, 2011.

[98] C. R. Wanberg, “The individual experience of unemployment,” *Annual reviewof psychology,* no. 63, pp. 369-396, 2012.

[99] P. J. Pedersen and N. Westergård-Nielsen, "Unemployment. A review of the evidence from panel data." in Economics of Unemployment, Edward Elgar Publishing, 2000.

[100] Í. M. d. R. de Troya, R. Chen, L. O. Moraes, P. Bajaj, J. Kupersmith, R. Ghani, N. B. Brás and L. Zejnilovic, “Predicting, explaining, and understanding risk of long-term unemployment,” in *32nd Conference on Neural Information Processing Systems*, 2018.

[101] C. J. Flinn and J. J. Heckman, “New methods for analyzing individual event histories,” *Sociological methodology,* no. 13, pp. 99-140, 1982.

[102] R. Serfozo, Basics of applied stochastic processes, Springer Science & Business Media, 2009.

[103] N. Privault, Understanding Markov Chains. Examples and Applications, Singapore: Publisher Springer-Verlag, 2013.

[104] N. Koenigstein, “The Xbox recommender system,” in *Proceedings of the sixth ACM conference on Recommender systems*, 2012.

[105] R. Sifa, C. Bauckhage and A. Drachen, “Archetypal Game Recommender Systems,” in *LWA,* 2014.

[106] J. O. Ryan, E. Kaltman, T. Hong, M. Mateas and N. Wardrip-Fruin, “People Tend to Like Related Games,” in *FDG,* 2015.

[107] M. Meidl, S. L. Lytinen and K. Raison, “Using game reviews to recommend games,” in *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2014.

[108] A. Chow, M.-H. N. Foo and G. Manai, “HybridRank: A Hybrid Content-Based Approach To Mobile Game Recommendations,” in *CBRecSys@ RecSys*, 2014.

[109] D. Jannach and L. Lerche, “Offline performance vs. subjective quality experience: A case study in video game recommendation,” in *In Proceedings of the Symposium on Applied Computing*, 2017.

[110] Y. Hu, K. Yehuda and V. Chris, “Collaborative filtering for implicit feedback datasets,” *Eighth IEEE International Conference on Data Mining,* 2008.

[111] P. Cremonesi, Y. Koren and R. Turrin, “Performance ofrecommender algorithms on top-n recommendation tasks,” *Proceedings of the fourth ACM conference on Recommender systems,* 2010.

[112] S. Rendle, Z. Li and K. Yehuda, “On the Difficulty of Evaluating Baselines: A Study on Recommender Systems. arXiv:1905.01395,” *arXiv preprint,* 2019.

**Viljanen, M., Airola, A., Heikkonen, J., & Pahikkala, T. (2017)**
**Playtime measurement with survival analysis**
IEEE Transactions on Games, 10(2), 128-138.

I

**Viljanen, M., Airola, A., Heikkonen, J., & Pahikkala, T. (2017)**
**A/B-test of retention and monetization using the Cox model**
In Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference

# II

**Viljanen, M., Airola, A., Majanoja, A. M., Heikkonen, J., & Pahikkala, T. (2020)**

**Measuring player retention and monetization using the mean cumulative function**

IEEE Transactions on Games, 12(1), 101-114.

III

**Byanjankar, A., & Viljanen, M. (2019)**
**Predicting Expected Profit in Ongoing Peer-to-Peer Loans with**
**Survival Analysis Based Profit Scoring**
In Intelligent Decision Technologies 2019 (pp. 15-26). Springer, Singapore.

IV

**Viljanen, M. , Byanjankar, A., & Pahikkala, T. (2020)**
**Predicting profitability of peer-to-peer loans with recovery models for censored data**
In Intelligent Decision Technologies 2020 (pp. 15-25). Springer, Singapore.

V

**Viljanen, M., & Pahikkala, T. (2020)**

**Predicting Unemployment with Machine Learning Based on Registry Data**

Predicting Unemployment with Machine Learning Based on Registry Data. In 2020 14th International Conference on Research Challenges in Information Science (RCIS) (pp. 352-368)

VI

**Viljanen, M., Pahikkala, T., Vahlo, J., & Koponen, A. (2020)**
**Content Based Player and Game Interaction Model for Game Recommendation in the Cold Start setting**
IEEE Transactions on Games, submitted

# VII