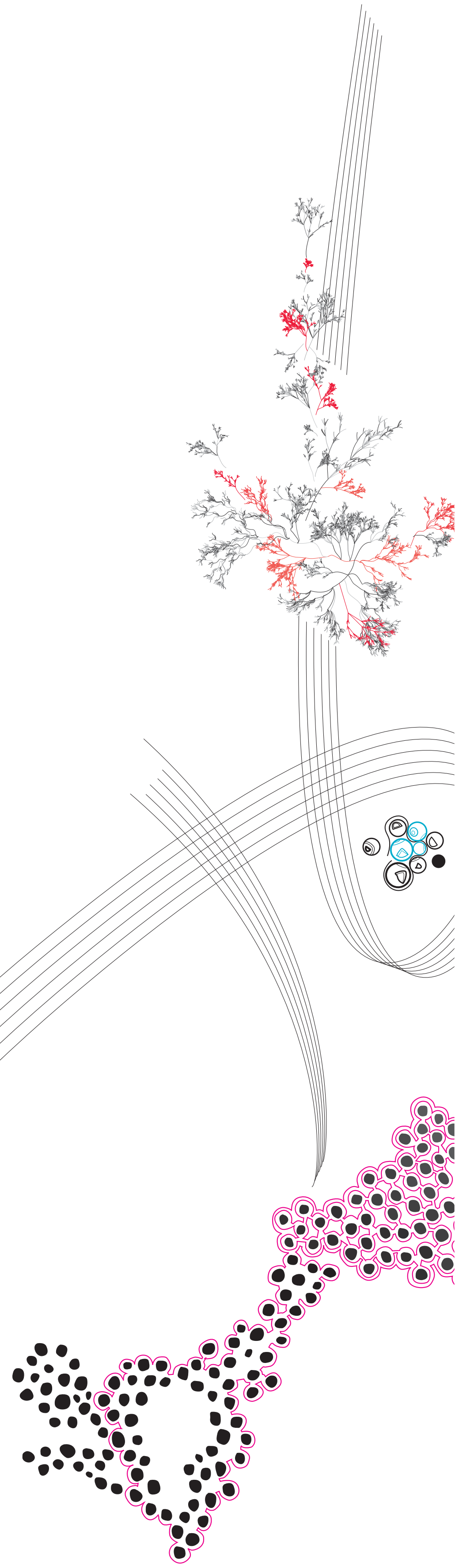
MSc Industrial Engineering
and Management



On machine learning approaches to forecast non-life insurers' loss reserves

Charly Hunsicker

Supervisor: Berend Roorda

December, 2023

Department of Industrial Engineering and Management
Faculty of Behavioural, Management and Social sciences

UNIVERSITY OF TWENTE.

"You don't have to reinvent the wheel, you just have to attach it to a new wagon."

– Mark McCormack

Preface

First and foremost, I would like to express my profound gratitude to everyone who has supported and guided me throughout the journey of completing this thesis. The experience has been both challenging and rewarding, and I could not have reached this milestone without the help and encouragement of many.

A special word of thanks goes to Lennart Niezen, my supervisor at KPMG. Lennart, your expertise, patience, and unwavering support have been invaluable to me. Your insights and feedback have not only shaped this research but have also contributed immensely to my personal and professional growth. I am truly grateful for the opportunity to work under your guidance.

I would also like to extend my heartfelt appreciation to Berend Roorda, my supervisor at the University of Twente. Berend, your academic prowess and dedication to excellence have been a constant source of inspiration. Your constructive critiques and encouragement have pushed me to delve deeper and strive for the best in my research endeavors.

To my peers, friends, and family, thank you for your understanding, encouragement, and the countless moments of solace you provided during the demanding times. Your belief in me has been a driving force, and I am forever grateful for your unwavering support.

Lastly, to anyone who has played a part, no matter how big or small, in this academic journey of mine, thank you. This accomplishment is as much yours as it is mine.

Abstract

This thesis, titled "On Machine Learning Approaches to Forecast Non-Life Insurers' Loss Reserves," has been conducted at KPMG Advisory N.V. in the department of Financial Risk Management (FRM). The study delves into the inherent risks faced by insurance companies, distinguishing between life and non-life insurance businesses. The primary focus is on the loss reserves of non-life insurance companies, which are crucial for ensuring the financial stability of these institutions. The loss reserve is divided into premium and loss reserves, with the latter being the main subject of this research. The research emphasizes the importance of accurate estimation of the loss reserve, as it impacts the pricing of insurance policies, compliance with regulations, and the overall financial standing of insurance companies. The study also explores the claim development patterns across different Lines of Business (LoBs) and the associated challenges in determining the appropriate reserve due to the varying nature of claims and their settlement timelines. The research builds upon previous work by exploring the application of machine learning models to predict the loss reserve of non-life insurers, aiming to provide a more accurate and efficient approach to this critical actuarial challenge.

Keywords: Machine learning, loss reserves, non-life insurance, actuarial risk, claim development, KPMG, Financial Risk Management

Contents

List of Figures

List of Tables

1 Introduction

With this introductory chapter we set the stage by presenting the organizational framework, then we delve into the critical concept of loss reserves in non-life insurance. We further explore the techniques for predicting these reserves and conclude with an overview of the research methodologies and context that underpin this study.

Contents

1.1 Organisation

We conduct this study at KPMG Advisory N.V. in Amstelveen, in the department of Financial Risk Management (FRM). KPMG offers advisory and audit related services. FRM is part of the risk and regulatory department which mainly focus on advisory. Furthermore, the FRM department provides risk-related advice to financial firms and institutions. The department is divided into four sub departments: asset management and pensions, actuarial insurance risk, banking and regulatory. An overview of the corporate structure is given in Figure 1.1.

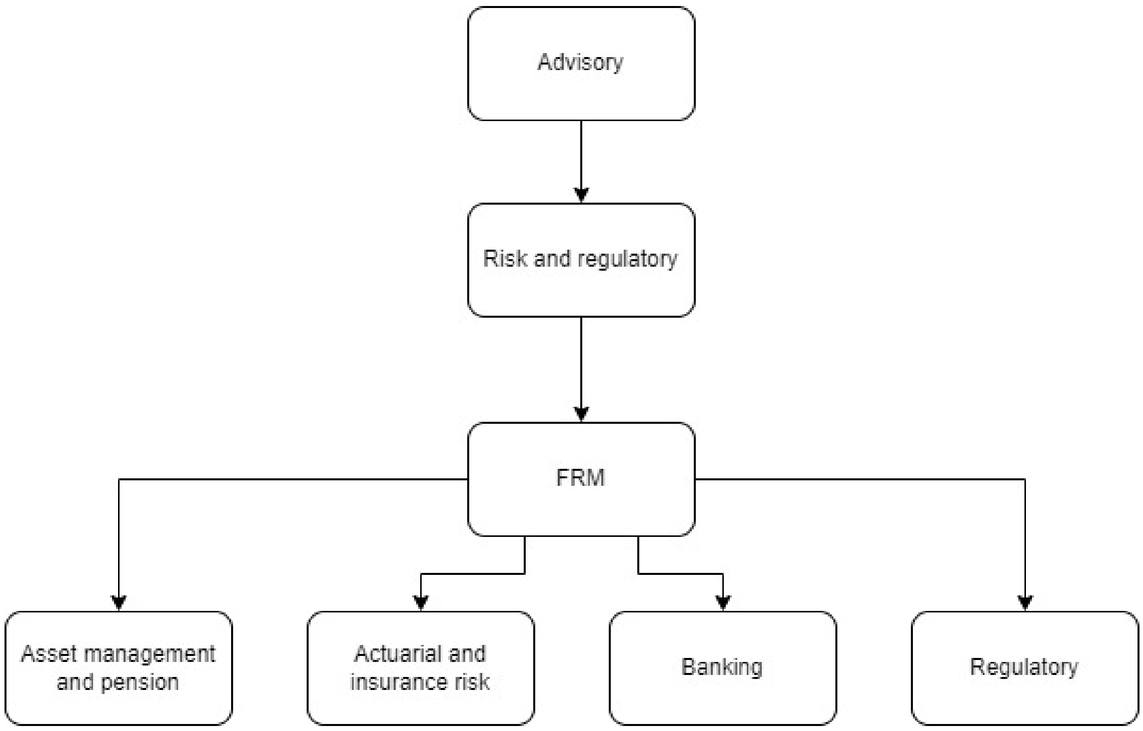


FIGURE 1.1: An overview of the corporate structure of KPMG and FRM.

FRM provides a wide range of financial risk services. They support banks, insurance companies asset managers and corporate and public clients in identifying assessing, managing, reporting and limiting the risks they face. Concerns about financial risk have spread throughout the world. In this environment, businesses of all shapes and sizes desire strong frameworks for managing financial risk that meet regulatory requirements, aid in better decision-making, and improve performance. To achieve these goals, financial institutions and other corporate clients collaborate with KPMG's FRM specialists. FRM raises issues for clients and works to improve procedures, governance, and strategy in a variety of areas, such as:

- Actuarial services & financial statement support
- Financial instruments accounting
- Insurance risk assessment
- Financial engineering

This thesis is tailored to the actuarial and insurance risk sub-department of FRM. Their focus is on the intricate modeling of insurance risk assessment, with a particular emphasis on actuarial loss reserving methods used in non-life sectors.

In 2019 there was a research done by P.L. Ruitenberg at the University of Twente for the FRM department called 'Adapting a hierarchical gaussian process model to predict the loss reserve of a non-life insurer'. The study delves into refining a hierarchical Gaussian process model to better predict the funds a non-life insurance company reserves for anticipated claims. This model aims to provide a more accurate estimation by considering multiple layers of data and relationships within the insurance sector. This research proposes potential improvements of a hierarchical Gaussian process model on the actuarial challenge of predicting the Loss Reserve of a non-life insurer. The initial research primarily utilized traditional methods and a Gaussian process model to forecast the loss reserve for life insurers. Building on this foundation, our thesis explores the use of machine learning models to predict the loss reserve for non-life insurance companies

1.2 Loss reserve in non-life insurance

Insurers are inherently vulnerable to different forms of risk due to the nature of their business. At the cost of a predetermined premium, insurance firms take on a specific risk that an individual (or firms) wants to mitigate (Kaas et al. 2008). In addition to the risks inherent in this business model (insurance risk), insurers face exposure to the financial market (market risk), a number of business risks, such as unexpected significant expenses, and operational risk, such as when protections or internal models fail.

There is a distinction between life and non-life insurance firms in the insurance industry. A life insurer writes life policies (e.g., pay out a predetermined sum to the beneficiaries in the event of the policyholder's death), whereas a non-life insurance firm writes all other types of insurance policies, such as vehicle, healthcare and property damage insurance. This distinction is made for both legal and product-related reasons: contract terms and claim categories differ. As a result, life and non-life insurance are typically modelled in distinct ways (Wüthrich & Merz, 2006).

Non-life insurance companies categorize their actuarial reserves into two main types: premium reserves and loss reserves. The premium reserve of non-life insurers encompasses premiums anticipated to be collected in the future. A loss reserve, as defined by Radtke, Schmidt, & Schnaus (2016), is an allocation for payment obligations stemming from incurred losses that remain unsettled. E. Frees (2018) highlighted several reasons underscoring the significance of accurately estimating loss reserves for insurance entities:

1. Loss reserves are perceived as debts the insurer owes to its policyholders.
2. Under-reserving can jeopardize the insurer's capability to meet claim obligations.
3. Overestimating reserves might portray the insurer's financial health as weaker than it truly is, potentially leading to a loss in market position.
4. The reserve estimation directly influences insurance product pricing, as it provides a projection of unpaid insurance costs.
5. Regulatory mandates necessitate the declaration of loss reserve values.
6. The general public, as consumers of insurance services, is keen on the financial robustness of these firms.
7. Many investors make decisions based on the reported loss reserve values.

The scope of this thesis is limited to the loss reserves of non-life insurance firms. The reserve is intended to cover losses or payments that will occur in the future, this results in a great deal of uncertainty in determining the size of this reserve. Therefore, this poses a danger for the insurance company: if the loss reserve is too low, the insurer may have difficulty paying claims. On the other hand, when the loss reserve is maintained at a high level, it can be negative to the business because that money cannot be used for other purposes.

In order to determine the loss reserve, non-life insurers can make distinctions across multiple Lines of Business (LoBs) (e.g. commercial or workers' liability). Every branch can have its own claim development pattern, and claims can have a delay in settlement that is specific to the type of claim. Liability products, for example, may face significant delays as a result of litigation or lawsuits (Kaas et al., 2008). As a result differences between the tail-distributions of LoBs can be distinguished. In short-tail LoBs losses are detected early, and with long-tail LoBs losses take longer to develop.

1.2.1 Claim timelines

The ultimate loss is the final amount paid once all claims have been settled. It often takes several years for this to occurs. At any point in time before then a portion of the ultimate loss will be paid and the remaining portion unpaid. The ultimate loss exists of unpaid losses, also known as the loss reserve and paid losses. The loss reserve is made up of two categories of claims: those that have incurred but not reported (IBNR) and those that are reported but not settled (RBNS). An overview of the loss structure is given in Figure 1.2

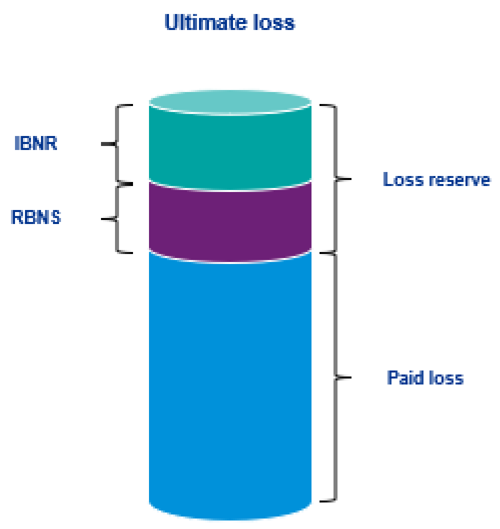


FIGURE 1.2: An overview of the loss structure of an insurer.

An actuary will assess probable damages in IBNR circumstances, and the insurance company may elect to set aside funds to cover the predicted losses. Losses recorded to an insurance company that have not been settled at the end of the accounting period are referred to as RBNS. Moreover, RBNS claims are estimated using an estimate of the severity of the loss based on information obtained during the claims settlement procedure.

An example of a claim's timeline is depicted in Figure 1.3. The claim arises at a specific point in time (t_1). There is a slight delay before the claim is reported to the insurer (t_2). After the notification, multiple payments (t_3, t_4, t_5) are made until the dossier is closed (t_6). Following that, it's possible that a dossier will be reopened if new information becomes available (T_7), after which additional payments occur (T_8). These additional payments can be recoveries for the insurer or extra payments to the insured. Finally, the claim is closed

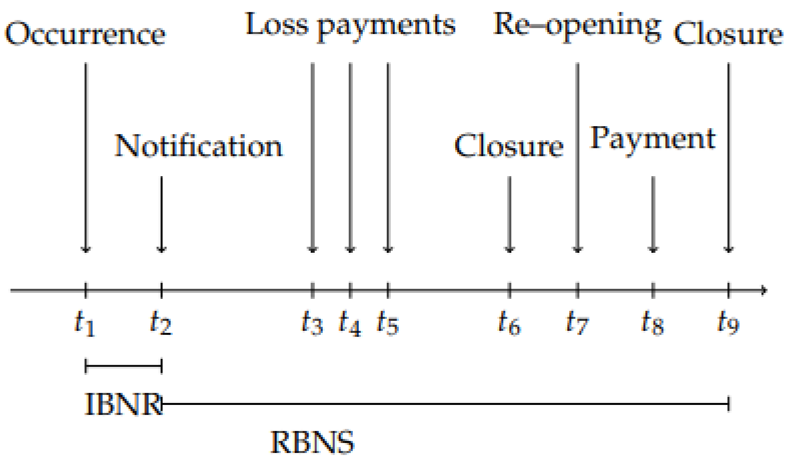


FIGURE 1.3: Example of a claim timeline, Source: Antonio and Plat (2014).

(t_9).

1.2.2 Claim triangles

Most traditional loss reserving techniques depend on creating two-dimensional matrices known as claim triangles. Claim triangles are created by compiling claims data during the timeline of a claim. The run-off matrices are then created from the claim data using a stochastic approach. This is in essence the claim triangles' empty cells filled in with predictions. The ultimate reserve value is calculated using these run-off triangles.

Payments or notifications of claims happen at various times, despite the fact that they start from the same moment in time, as mentioned in the previous section. Based on the period at which the claim incurred and the time at which the insurer has made payments on the claim, an overview of the total amount paid can be produced (cumulative or incremental). At last, we can define a specific calendar year in which the observed loss occurred. The properties of the claim triangle are given below:

- $i =$ Accident year $(i = 1, ..., I)$
- $j =$ development lag $(j = 1, ..., J)$
- $k =$ Calendar Year = $i + j$
- $z_{i,j} =$ Observed incremental loss
- $\hat{z}_{i,j} =$ Estimated incremental loss
- $c_{i,j} =$ Observed cumulative loss $= \sum_{j=1}^{J} z_{i,j}$
- $\hat{c}_{i,j} =$ Estimated cumulative loss $= \sum_{j=1}^{J} \hat{z}_{i,j}$
- $R_i =$ Observed loss reserve for accident year i = $\sum_{j=I-i+2}^{I} z_{i,j}$
- $\hat{R}_i =$ Estimated loss reserve for accident year i = $\sum_{j=I-i+2}^{I} \hat{z}_{i,j}$
- $R =$ Total loss reserve of the triangle $= \sum_{i=1}^{I} R_i$

- $\hat{R}$ = Total Estimated loss reserve of the triangle = $\sum_{i=1}^{I} \hat{R}_i$

With i as accident year (starting at 1), j as development lag and zi, j ($c_{i,j}$) as the (cumulative) loss, a triangle can be created, also known as a claim triangle. Each cell within this (cumulative) triangle is filled with the loss $z_{i,j}$ ($c_{i,j}$) which we hereafter refer to as cell. Claim triangles can either be incremental or cumulative, as shown in Table 1.1 and 1.2 respectively. Each observed loss is associated to an accident year and development lag, this is hereafter referred to as cell.

Incremental claim triangle				
	Development lag (j)			
Accident year	**1**	**2**	**3**	**4**
2001 (1)	1198	2245	689	257
2002 (2)	1253	3232	638	.
2003 (3)	1087	1465	.	.
2004 (4)	1238	.	.	.

TABLE 1.1: An example of an incremental claim triangle, as of December 31, 2004.

Cumulative claim triangle				
	Development lag (j)			
Accident year (i)	**1**	**2**	**3**	**4**
2001 (1)	1198	3443	4132	4389
2002 (2)	1253	4485	5123	.
2003 (3)	1087	2552	.	.
2004 (4)	1238	.	.	.

TABLE 1.2: An example of a cumulative claim triangle as of December 31, 2004.

We can also plot the data of the development of the claim by origin year as displayed in Figure 1.4. In a typical claim triangle, we can see that the cumulative observed loss increases when the development increases. Furthermore, we have fewer data points for later origin years, since only the data until 2004 are known. Several techniques to estimate the empty cells of the triangle will be discussed later. The sum of the empty cells of the incremental claim triangle is what we define as the loss reserve.

1.3 Predicting the loss reserve

In literature, there are two main categories into which insurance reserve prediction techniques can be classified: classical actuarial chain ladder or stochastic methods (Avanzi et al., 2016; Boratynska, 2017; Diers & Linde, 2013; BDjehiche & Lofdahl 2016; England et al., 2019; Feng & Yi. 2019; Ferriero, 2016; Frohlich & Weng, 2018; Gigante et al., 2019; Huang et al. 2015; Peters et al. 2017; Wahl et al. 2019), or machine learning based loss reserve prediction techniques (Wuthrich, 2018; Kuo, 2019; Gabrielli et al., 2018; Gabrielli, 2019; Lecun et al., 2015). An overview of different loss reserving prediction techniques is given in Figure 1.5.

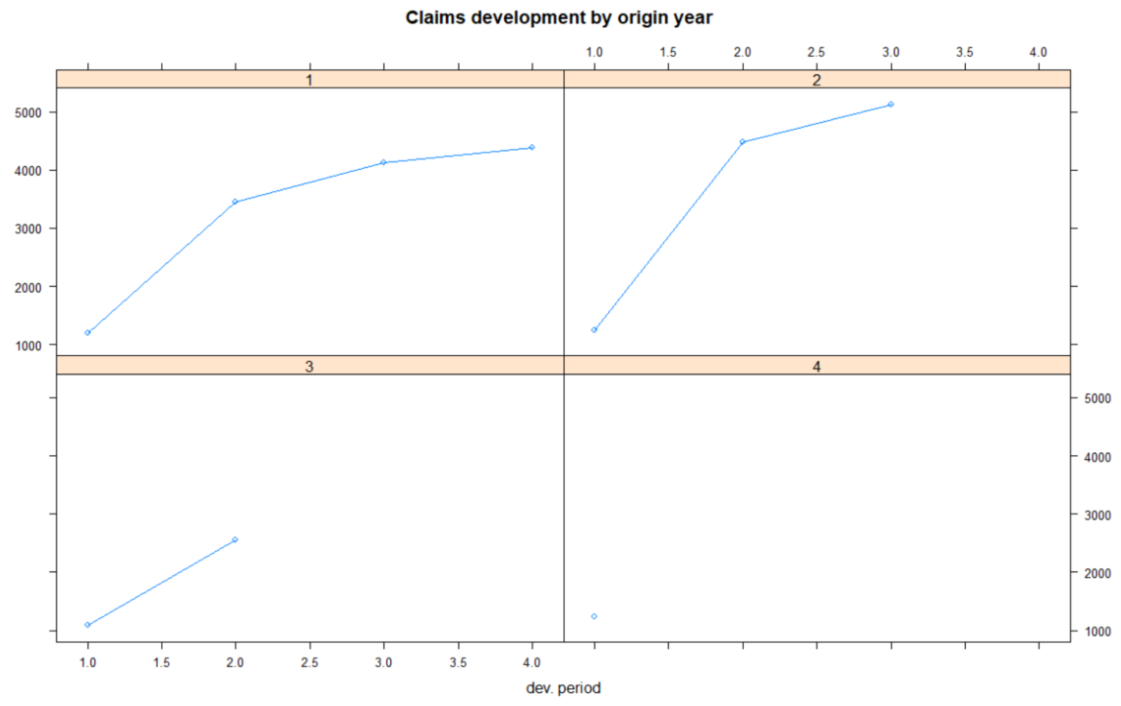


FIGURE 1.4: Claims development by origin year.

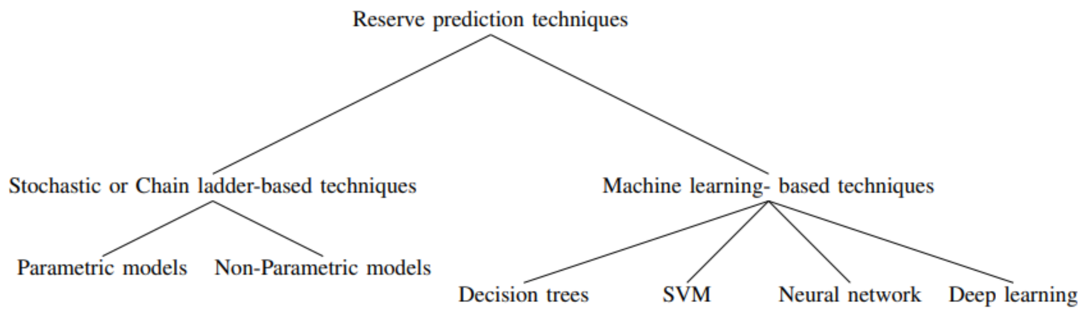

Reserve prediction techniques
Stochastic or Chain ladder-based techniques
Machine learning- based techniques
Parametric models
Non-Parametric models
Decision trees
SVM
Neural network
Deep learning

FIGURE 1.5: The taxonomy of loss reserve prediction methods, source: Taha et al., (2021).

1.3.1 Stochastic and chain ladder based techniques

The chain ladder method is the predominant approach for estimating loss reserves. Insurers use this method to forecast the required reserves based on extrapolating historical claim data. However, the chain ladder's effectiveness hinges on the assumption that past loss trends will persist in the future. If there is a shift in an insurer's claim patterns, the chain-ladder method may not yield precise estimates without appropriate modifications. We'll delve deeper into this technique in the subsequent chapter.

Stochastic reserving models operate on the premise that historical insurance claim patterns will persist. This means that the dynamics of claim activities remain relatively consistent over time. To predict the ultimate loss, these models employ basic statistical forecasting techniques like regression. While traditional loss reserve prediction methods are intuitive and quick to grasp, Avanzi et al. (2016) suggest that, especially with complex claim data, stochastic methods fall short in accuracy compared to machine learning techniques. This complexity might arise from random variations in claims data, which can diverge future trends from historical observations. Stochastic reserve techniques can be bifurcated into two main types: parametric and non-parametric models. Parametric models hinge on statistical distributions, aiming to pinpoint the best parameters for these distributions. For instance, in a normal distribution, they would identify optimal mean and standard deviation values to project future reserves. Conversely, non-parametric models do not lean on specific statistical distributions (Shi, 2014). They come into play when claim data does not conform to a known distribution. While parametric models often outperform and are faster than their non-parametric counterparts, they are inapplicable if data does not fit a known distribution. Machine learning solutions excel with intricate insurance data, especially when past claim-inducing behaviors aren't expected to recur. If claim patterns evolve, it becomes challenging for any model to accurately capture their development due to inherent randomness. Traditional models, like the chain ladder and linear regression, falter in predicting future claim facets in such scenarios. Hence, machine learning methods, which employ nonlinear predictive techniques, emerge as more potent tools for such data.es, for this kind of data, become more promising.

1.3.2 Machine learning based techniques

There is limited research done in the field of machine learning to predict the loss reserve. However, due to its success in numerous disciplines over the past few years, deep learning has recently gained attention in the loss reserving literature (Lecun et al., 2015). Wüthrich (2018) uses neural networks to synthesize claims data, Gabrielli et al. (2018) and Gabrielli (2019) embed classical parametric loss reserving models into neural networks, and Wüthrich (2018) extends the conventional chain ladder method with neural networks to incorporate claims features. The research by Gabrielli et al. (2018) and Gabrielli (2019) particularly suggests initializing a neural network so that, prior to training, it conforms to a classical model, such as the over-dispersed Poisson model. Lopes et al (2016) propose a regression tree method to calculate the conditional distribution of a variable that is censored off from direct observation. At last, Kuo (2019) proposes an approach to loss reserving based on deep neural networks, which allows for the incorporation of heterogeneous inputs into combined modeling of outstanding paid losses and claims.

1.4 Research

Literature suggests that machine learning models enhance the performance of predicting loss reserves in non-life insurance. We contribute to this literature by offering a comprehensive review of commonly used classical methods and juxtaposing their efficacy with machine learning techniques. Additionally, we aim to improve the performance of loss reserve predictions by introducing a novel machine learning algorithm tailored for forecasting non-life insurers' loss reserves.

1.4.1 Research Questions

We focus on improving the predictability of the loss reserve of non-life insurers, using machine learning techniques. Therefore, our research question is as follows:

How effective are machine learning techniques in enhancing the performance of loss reserve predictions for non-life insurers?

With the following sub-questions:

1. *Which metrics can be employed to assess the performance of loss reserve predictions in non-life insurance?*
2. *Which conventional methods have been historically utilized for loss reserve forecasting in non-life insurance*
3. *Which machine learning algorithms have been previously explored for enhancing loss reserve predictions in the non-life insurance sector?*
4. *Which features are important for influencing the performance of loss reserve predictions?*
5. *Can we improve the performance of predicting recoveries using machine learning?*
6. *Which validation techniques can ensure the reliability and robustness of our loss reserve prediction models?*
7. *Which specific machine learning approaches can be recommended to optimize the forecasting of loss reserves in non-life insurance?*

1.4.2 Scope

Our research is limited to chain ladder approaches and machine learning approaches to predict the loss reserve since Ruitenberg (2019) already did extensive research on stochastic based techniques. Furthermore, due to time restrictions, we do not include claim triangles with missing data. However, we will try to deal with recoveries, since this is a common phenomenon in practice.

2 Literature and background

In this chapter, we first discuss the traditional approaches to predicting the loss reserve. As discussed in the introduction we will focus on chain ladder techniques and view the stochastic techniques as out of scope. next, we discuss machine learning concepts used in this Thesis. At last, we will give a summary of machine learning techniques applied to loss reserve prediction in the body of literature.

Contents

2.1 Traditional approaches to predict the loss reserve

The predictability of the the loss-reserve of non-life insurers is the focus of this thesis. In the introduction, an overview has been given of the possible traditional approaches to predicting the loss reserve. We use the traditional approaches as benchmark for the machine learning approach. Therefore, we will discuss some of the most popular traditional chain ladder methods used in practice, namely, chain ladder, bootstrap chain ladder and LDF chain ladder.

2.1.1 Chain Ladder

The first and most commonly used traditional approach to predicting the loss reserve of non-life insurers is the chain ladder. It is a deterministic algorithm that forecasts claims based on historical data. Moreover, the chain ladder is predicated on the idea that claims develop proportionally from one development period to the next for all origin years. It has gained popularity thanks to Mack (1993, 1999), and it is extensively discussed in literature. Historical development factors may be found using I as the accident year variable and J as the development lag variable. These variables can be used to estimate future losses and the loss reserve. The Mack chain ladder relies on three assumptions to give an unbiased estimator for the IBNR claims. We follow the notation of Mack (1999) as described earlier, let $c_{i,k}$ denote the cumulative loss of origin period (e.g. accident year) $i = 1, \dots, m$, with losses known for development period year $k \leq n + 1 - i$. The model assumptions are given below:

$$CL1 : \mathbb{E}[DF_{i,k}|c_{i,1}, c_{i,2}, ..., c_{i,k}] = DF_k \qquad with, \qquad DF_{i,k} = \frac{c_{i,k+1}}{C_{i,k}} \tag{2.1}$$

$$CL2 : Var(\frac{c_{i,k+1}}{c_{i,k}}|c_{i,1}, c_{i,2}, ..., c_{i,k}) = \frac{\sigma_k^2}{w_{i,k}c_{i,k}^a} \tag{2.2}$$

$$CL3 : \{c_{i,1}, c_{i,2}, ..., c_{i,n}\}, \{c_{j,1}, c_{j,2}, ..., c_{j,n}\} \text{ are independent for origin period } i \neq j \tag{2.3}$$

Where DF denoted the development factor and $w \in [0, 1]$, $a \in 0, 1, 2$ The heuristic as described by Mack (1993) is as follows:

$$DF_j = \frac{\sum_{i=1}^{n+1-j} c_{i,j}}{\sum_{i=1}^{n+1-j} c_{i,j-1}} \qquad \hat{C}_{i,j} = c_{i,n+1-i} \prod_{n-i+2}^{j} DF_{n-i+2} \tag{2.4}$$

Where DF_j denotes the development factor with development lag j. If we apply this heuristic to the cumulative claim triangle of Table 1.2 we can find the development factor and ultimately the loss reserve. The results are shown in Table 2.1.

Example of a completed triangle					
	Development lag (j)				
Accident year	**1**	**2**	**3**	**4**	Loss reserve (R)
2001 (1)	1198	3443	4132	4389	0
2002 (2)	1253	4485	5123	**5442**	319
2003 (3)	1087	2552	**2979**	**3164**	612
2004 (4)	1238	**3667**	**4281**	**4547**	3309
DF		2,96	1,17	1,06	4240 = $\hat{R}$

TABLE 2.1: An example of an estimated chain ladder claim triangle.

We can also plot the chain ladder developments by origin period, including the standard error as shown in Figure 2.6. We can see that when the accident year increases the standard error increases. This is logical since there is more uncertainty due to less data.

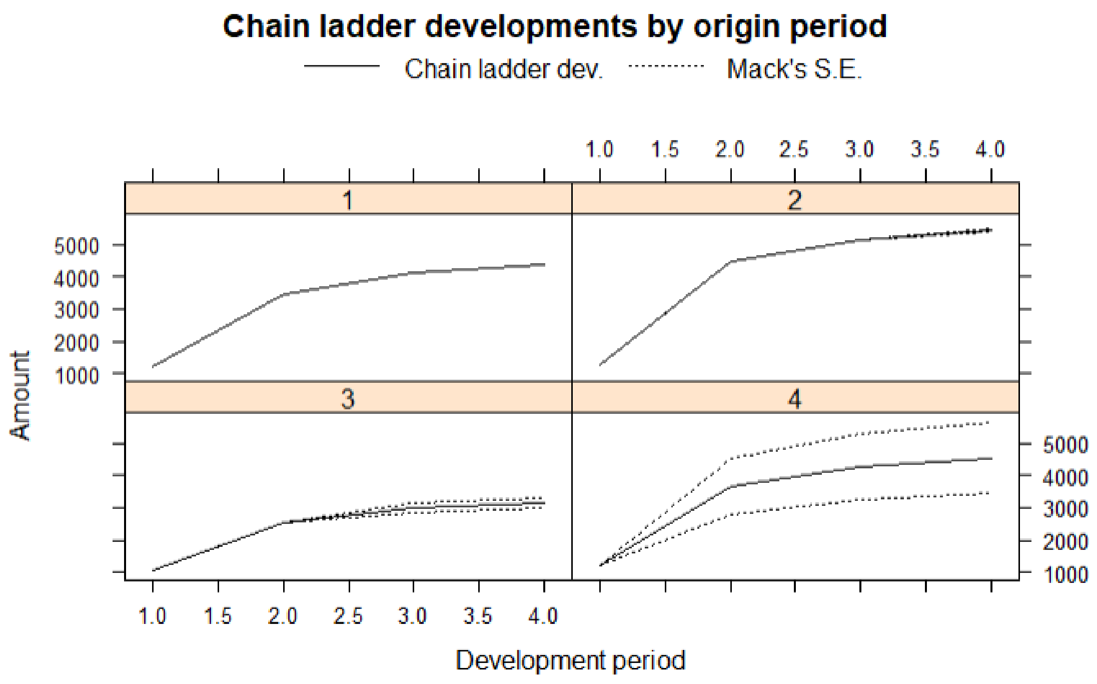


FIGURE 2.1: Chain ladder development by origin period, inculding standard error of the chain ladder estimate.

2.1.2 Bootstrap Chain Ladder

Using a single sample of data, bootstrapping (Efron & Tibshirani, 1993) is a strong yet straightforward method for extracting information that would typically require analytic methods. This can also be applied to single claim triangles. To generate numerous sets of pseudo-data that are compatible with the same underlying distribution, the process involves replacing samples of observed data with new samples obtained by sampling. The distribution of the statistics of interest can then be looked into to get additional insight, and they can be derived for each set of pseudo-data. The standard deviation of the set of means can be estimated as an estimate of the standard error of the mean, for instance, by taking the mean of each set of pseudo-data (England & Veral 2002).

The Bootstrap Chain Ladder follows the two-stage bootstrapping/simulation paper of England and Verrall (2002. The cumulative claims triangle is subjected to a standard chain ladder approach in the initial stage. We next use the scaled Pearson residuals to anticipate future incremental claims payments using the traditional chain ladder method, bootstrapping a predefined number of times. In the second stage, we simulate the process error using the presumptive process distribution and the bootstrap value as the mean. The set of reserves acquired in this manner makes up the predictive distribution, which can be used to derive summary statistics like mean, prediction error, or quantiles.

This two-stage bootstrapping/simulation method has the benefit of being simple to set up in a spreadsheet and not requiring complex calculations or sophisticated statistical software. Moreover, the results can give more in depth insights compared to the traditional chain ladder (England & Verrall, 2002).

When data are assumed to be independent and identically distributed (i.i.d.) the traditional bootstrapping approach: resampling with replacement from the original data points is applied. With regression type problems, however, the data is assumed to be independent but not identically distributed. This is because the means, and sometimes variances are dependent on the covariates. In regression analysis, bootstrapping typically involves the residuals rather than the original data points. This approach is chosen because the residuals are approximately independent and identically distributed (i.i.d.).

However, it is crucial to define residuals appropriately, for its specific problem. There are various expanded definitions of residuals for generalized linear models, with the precise form determined by the underlying modeling distribution (McCullagh & Nelder, 1989). To do bootstrapping on the over-dispersed Poisson chain-ladder model, we make use of the Pearson residuals (England & Verral, 2002). Pearson residuals are as defined in Equation 2.5

$$r_{i,j} = \frac{z_{i,j} - \mathbb{E}(\hat{z}_{i,j})}{\sqrt{\mathbb{E}(\hat{z}_{i,j})}} \tag{2.5}$$

The bootstrap method uses, from the Pearson residuals, replacement with resampling. We can invert Equation 2.5. Moreover, given a resampled Pearson residual and the estimated expected value, we can define the associated bootstrap incremental claim values (z^*). The formula for the associated incremental claim value is given in Equation 2.6

$$z^*_{i,j} = r^*_{i,j}\sqrt{\mathbb{E}(\hat{z}_{i,j})} + \mathbb{E}(\hat{z}_{i,j}) \tag{2.6}$$

We then fit the chain-ladder model to the bootstrap sample and obtain the forecast incremental claims payments. Next, in line with the method of England and Verral (2002), we replicate the process variance by simulating an observed claim for each cell of the future in the claim triangle. We use the bootstrap value as the mean and fit an over-dispersed Poisson distribution (which has the same results as the chain ladder heuristic).

At last, this procedure is repeated a large number of times (X), each run provides a new bootstrap value and completed claim triangle. This results in a set of reserves for each run. The set of reserves can be used to obtain the set of summary statistics.

When applying the bootstrap chain ladder to our example triangle of Table 1.2 the results are approximately the same as the Mack Chain ladder due to the simplicity of our triangle. However, plotting the results gives some extra insights as can be seen in Figure 2.2. For example, we can see that the distribution of the IBNR appears to be log-normal.

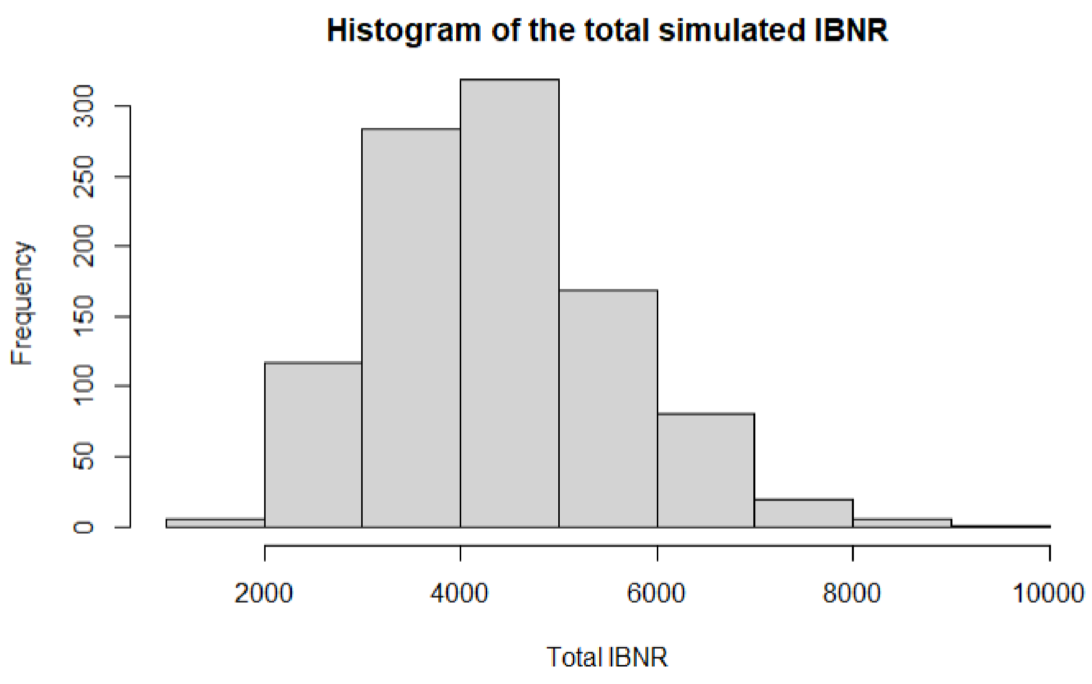


FIGURE 2.2: Histogram of IBNR of the bootstrap simulation with 999 runs.

2.1.3 Clark's LDF Method

In the field of loss development analysis, the Clark Loss Development Factor (LDF) approach offers a distinct perspective, as discussed in the context of Clark's 2003 study. This method hypothesizes that loss evolution aligns with a theoretical growth curve, analyzed using a longitudinal approach. Within this framework, a notable implementation is the LDF method, which interprets the growth curve in forms such as a step function or a piecewise linear model. The growth curve, in this context, is understood to represent the cumulative percentage of the total loss expected to manifest at each stage of a loss's origin period.

The LDF technique operates under the assumption that ultimate losses for each origin period are independent and unrelated. The objective is to fine-tune parameters for these ultimate losses and the growth curve to achieve the best fit with the data in a claims triangle, as referenced in Table 1.2. Clark's analysis includes two primary growth curve models: the Weibull function and the log-logistic function, each characterized by two parameters. The foundational assumption for the maximum likelihood estimation in this approach is the adherence of incremental losses to an Over-Dispersed Poisson (ODP) distribution.

This approach is exemplified in a modified claims triangle, demonstrating the Clark LDF method with a maximum age of 6 and a log-logistic distribution, as shown in Table 2.2. Compared to the chain ladder method, this example reveals a significant increase in the loss reserve estimates (4240 vs. 7245 with the Clark LDF method). This increase is attributed to the assumption that claim growth continues beyond the fourth development year to a maximum age of 6. Moreover, the assumption of a 'fat tail' in the log-logistic distribution leads to higher loss reserve estimations.

Example of a completed Clark LDF triangle						
	Development lag (j)					
Accident year	**1**	**2**	**3**	**4**	**Max**	Loss reserve (R)
2001 (1)	1198	3443	4132	4389	**4529**	140
2002 (2)	1253	4485	5123	**5554**	**5732**	609
2003 (3)	1087	2552	**3257**	**3532**	**3644**	1092
2004 (4)	1238	**4651**	**5937**	**6436**	**6642**	5404
DF		3,76	1,28	1,08	1,03	7245=$\hat{R}$

TABLE 2.2: An example of an Clark LDF chain claim triangle with max age = 6 and a log-logistic distribution.

2.2 Machine Learning to Predict the Loss Reserves

In this section, we elaborate on machine learning techniques to predict the loss reserve discussed in literature. As stated before in the introduction, there is limited research done in the field of machine learning to predict the loss reserve. However, due to its success in numerous disciplines over the past few years, deep learning has recently gained attention in the loss reserving literature (Lecun et al., 2015). We first give some background information on the different machine learning techniques used, such as neural networks, regression tree, XGBoost and auto machine learning. Finally we discuss the existing literature on predicting the loss reserve using machine learning techniques.

2.2.1 Neural networks

In comparative analysis, neural networks emerge as the most sophisticated and flexible machine learning method. Inspired by the human brain, they consist of interconnected layers of artificial neurons, or perceptrons, that learn through a process of trial and error. While a single perceptron possesses limited predictive capability, collectively, they form a network capable of capturing complex predictor interactions, significantly enhancing prediction accuracy. Even basic neural networks comprise hundreds of parameters, classifying them as highly parameterized models. Their versatility makes neural networks adept at approximating intricate non-linear functions and tackling challenging problems. However, their complexity also renders them among the least interpretable and transparent machine learning approaches.

A neural network consists of three different sorts of layers, with one or more perceptrons in each layer. Input patterns are gathered in the first layer, the input layer, which also holds the raw data for the predictor variables. The number of predictor variables in the predictor set must match the dimension of the perceptrons in the input layer. The hidden layer(s), the middle layer(s), adjusts the function weightings until the objective function of the neural network is optimized. You can build a neural network with any number of hidden layers. How shallow or deep the network architecture depends on the number of hidden layers. The perceptrons from the hidden levels are combined in the output layer, the last layer, to create a single classification or prediction signal.

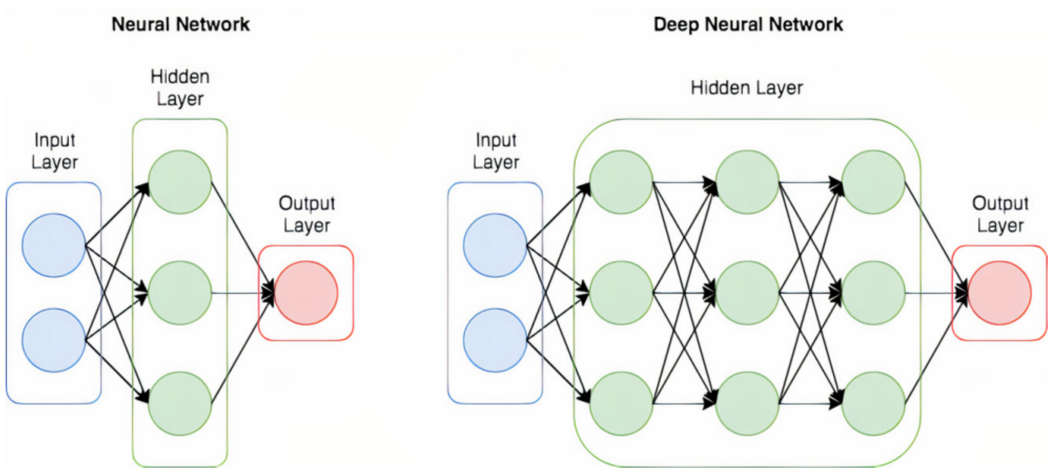


FIGURE 2.3: 1-layered and 3-layered feedforward neural network.

Basic perceptron A basic perceptron is the most straightforward type of neural network. A neural network is a basic perceptron containing an output node and a single input layer. The basic perceptron's model architecture is shown in Figure 2.4.

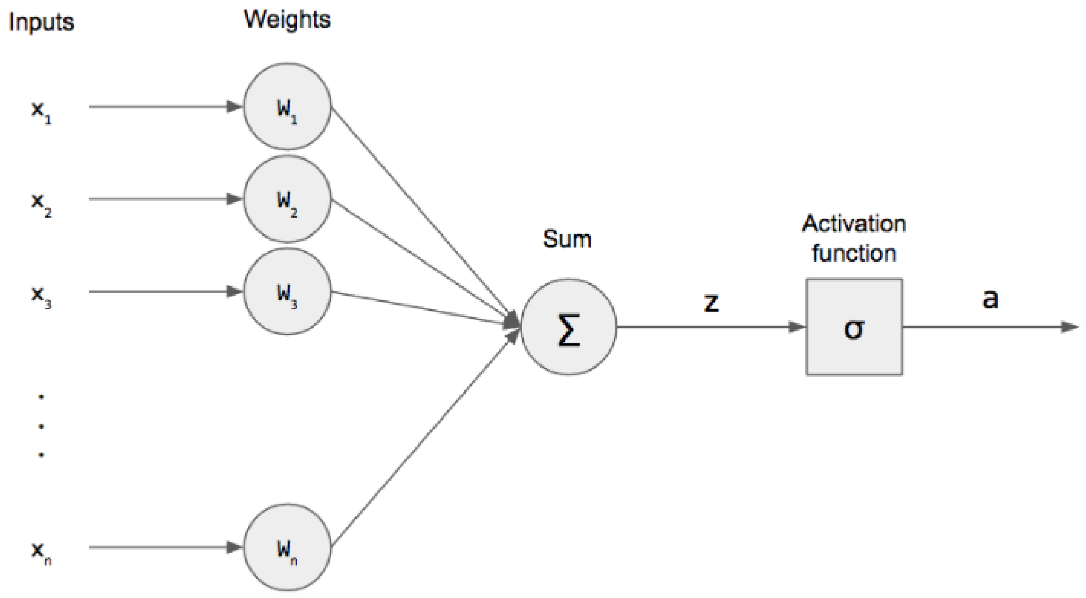


FIGURE 2.4: Perceptron model architecture.

The input layer includes an input layer with n nodes transmitting n features $\vec{x} = [x_1.x_2, ..., x_n]$ with edges of weight $\vec{w} = [w_1, w_2, ..., w_n]$ through the activation function $\phi()$ and at last to an output node. The output node is the predictor $\hat{y}$, or in our case $\hat{z}_{i,j}$. The predicted value $\hat{y}$ can be computed as follows:

$$\hat{y} = \phi(\vec{w} \cdot \vec{x}) = \phi(\sum_{j=1}^{n} w_j x_j) \tag{2.7}$$

The perceptron's goal is to approximate the observation by minimizing the difference between the estimation and observation by changing the weights $\vec{w}$. This is done with a loss function. One of the most common loss functions is the ordinary least squares loss function, which is computed as follows:

$$Minimize_{\vec{w}}(L) = \sum_{\vec{x},y \in D} (y - \hat{y})^2 = \sum_{\vec{x},y \in D} (y - \phi(\vec{w} \cdot \vec{x}))^2 \tag{2.8}$$

Where D refers to the given data set. A design choice within this perceptron is which activation function to use. The two most used activation functions for non-binary neural networks are given below:

$$\text{ReLU} : \phi(v) = max(0, v) \tag{2.9}$$

$$\text{Sigmoid} : \phi(v) = \frac{e^v}{e^v + 1} \tag{2.10}$$

Multilayer neural network A multilayer neural network contains additional computational layers besides the input and output layers. Considering a basic perceptron, the output layer is the sole layer performing computations, by giving weights to the inputs and applying the activation function. In a multilayer neural network, hidden layers are added. When there is more than one hidden layer in the neural network we speak of a deep neural network, as shown in Figure 2.3.

Learning algorithms For most neural networks the backpropagation algorithm is the learning algorithm. It is used to iteratively calculate the model parameters and weightings. This is a supervised, stochastic gradient descent (sgd) based approach. We can determine the direction of the steepest ascent by computing a function's gradient. This demonstrates the direction in which the weights should be changed to increase (or decrease) the outcome of the objective function as quickly as possible. The backpropagation algorithm works by estimating the objective function's negative gradient and then modifying the weights accordingly. Initially, backpropagation assigns random weights to all of the perceptron linkages, creating an initial function. The technique computes the associated objective function, often mean squared error (MSE), to gain information on how to efficiently change the weights to improve predictive performance. After defining the initial weights the algorithm iterates over the following three-step procedure:

1. Compute the gradient of the objective function using the current (or initial) weights, this starts from the output layer.
2. The algorithm slightly modifies the weights in the direction of the steepest descent in order to effectively reduce the MSE of the function. The predetermined learning rate parameter determines the size of the descent.
3. The objective function is re-calculated after adjusting the weights according to step 2.

This process is how a single training observation slightly changes the weights and biases in the neural network. The training algorithm should ideally repeat this process for each observation of the training set, however, this is computationally intensive and can be done in a more efficient way. Therefore, most neural networks use a gradient descent method, in which we iterate over a small subset of observations rather than all of the observations. This allows the algorithm to estimate the network using all of the training data while significantly accelerating computation. After running the algorithm over the full dataset a predefined number of times, stochastic gradient descent process finishes.

Iterations, batch sizes and epochs A neural network is trained for one cycle during an epoch using all of the training data. We only use each piece of information once within an epoch. Each epoch consists of one or more batches in which the neural network is trained using a portion of the dataset. We refer to the process of going through a batch of training examples as an iteration. Let us look at a straightforward example where we have 1000 data points, as shown in Figure 2.5, to clarify.

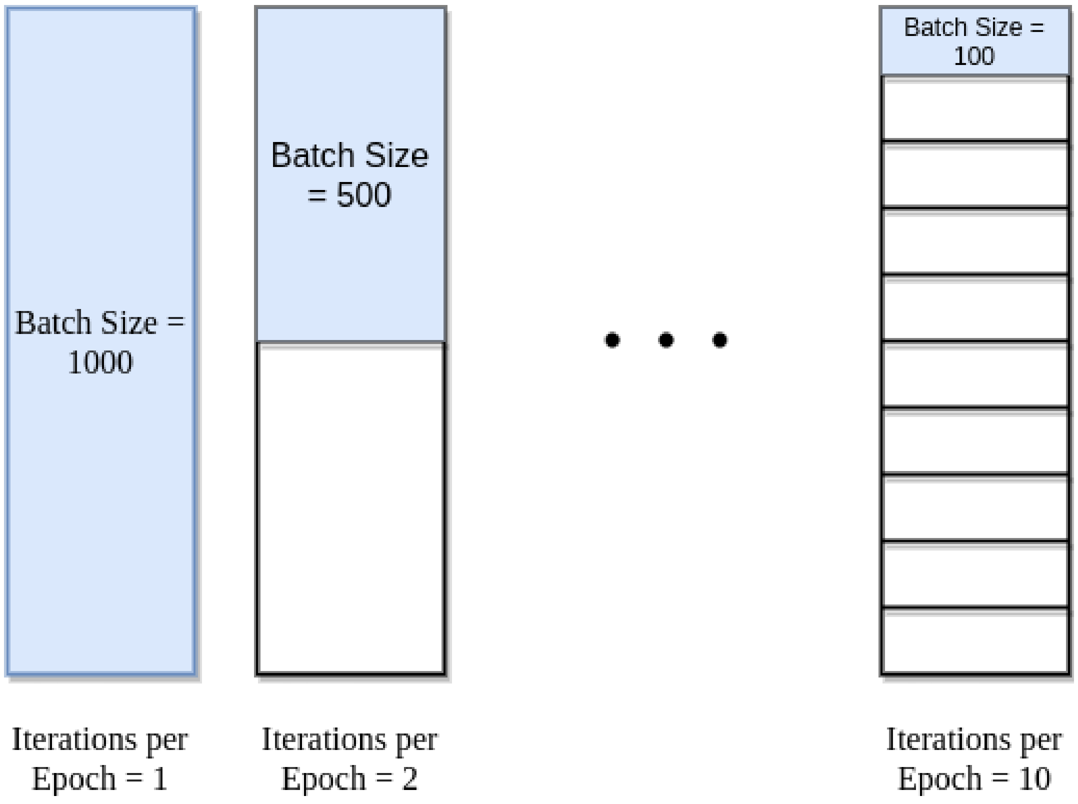


FIGURE 2.5: Visualisation of batches, epochs and iterations.

We can finish an epoch in one iteration if the batch size is 1000. Similarly to this, an epoch requires two iterations if the batch size is 500. As a result, an epoch requires 10 iterations to complete if the batch size is 100. The amount of data points is simply calculated for each epoch by multiplying the necessary number of iterations by the batch size.

2.2.2 Advanced tree-based methods

A tree-based method is one of the most interpretable machine learning techniques. It is a graphical approach towards prediction. The techniques separate the predictor space into various sections. A tree can be used to represent the separation into sections. Thus, the phrase "decision tree" is frequently employed. While generally easy to understand, the prediction quality of tree approaches is often not optimal. James et al. (2013) discuss the up and downsides of the technique:

- Trees are easy to interpret.
- Trees are similar to instinctive human decision making.
- Trees are easy to visualize.
- Trees are able to use categorical predictors without having to transform them (with for example one hot encoding)
- Trees normally do not have the best predictive quality as other techniques.

Regression trees Regression trees are non-parametric and have a different foundation than other machine learning techniques. Regression trees' primary goal is to aggregate related observations together into clusters. A regression tree is constructed in two steps. We first divide the data into j distinct, non-overlapping segments $[D_1, D_2, ..., D_j]$. Second, we create a split with objective to minimize the sum of squared residuals (SSR):

$$SSR = \sum_{j=1}^{J} \sum_{i_j} (y_i - \hat{y}_{D_j})^2 \tag{2.11}$$

Where the dependent variable's mean for the training observations contained in the j-th partition is denoted by the symbol $\hat{y}_{D_j}$. Using the residual data from the previous steps, the model chooses the best predictor and cut-off value to split the partition and proceeds to iteratively develop the regression tree. The term "binary recursive partitioning" describes this procedure. The regression tree cannot be constructed computationally by taking into account all splits. We substitute the "top-down greedy" algorithm instead. We begin at the top of the tree, when every observation belongs to a single partition covering the entire dataset. The space is then divided into two areas using the best split variable and value. This operation is repeated until a stopping requirement is met. By placing the observation in one of the clusters or leaves, the regression tree executes predictions. The predicted value is equal to the average of all the cluster's (training) observed values (Morgan, 2014).

Gradient boosting machines Boosting operates around the tenet that the next learner can correct the mistakes of the prior learner. Weak learners (such as a basic decision tree) who do only marginally better than random chance are utilized in boosting. Boosting focuses on building up these weak learners successively and removing the observations that a learner correctly understands at each step. In essence, the emphasis is on creating new, weak learners to manage the remaining, challenging observations at each phase. Gradient boosting machines are collections of consecutive trees, where each tree aims to enhance the output of the previous one. We consider mainly the gradient of the loss function, the difference between the value of a formed tree's predicted value and the observed value. When gradient boosting techniques are calibrated properly, Boehmke et al.(2019) argues that they are difficult to get outperformed by alternative algorithms.

Chen and Guestrin (2016) introduce Extreme Gradient Boosting (XGBoost). There are three system improvements offered by XGBoost. First, the technique uses a parallelized implementation to execute sequential tree construction. Hence, the branches are separately and concurrently produced inside the regression trees. The system also rearranges trees through the predetermined hyperparameter "max depth" as a criterion for halting. The computing performance is significantly enhanced by this backpropagation approach. Finally, XGBoost is able to use both the GPU and CPU for model estimation since it is built to maximize the effectiveness of the hardware that is already in place.

The inputs for the XGBoost algorithm are: 1) training set $\{(x_i, y_i)\}_{i=1}^{N}$, 2) a differentiable loss function $L(y, F(x))$, 3) weak learners M and 4) a learning rate α (Chen et al., 2016). The algorithm goes as follows (adopted from "XGBoost", 2022):

1. Initialize model with a constant value:

$$\hat{f}_{(0)}(x) = \arg\min_{\theta} \sum_{i=1}^{N} L(y_i, \theta)$$

2. For $m = 1$ to M :

   (a) Compute the 'gradients' and 'hessians':

$$\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)-\hat{f}_{(m-1)(x)}}$$

$$\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=\hat{f}_{(m-1)}(x)}$$

   (b) Fit a base learner (for example a decision tree) using the training set $\left\{x_i, -\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_\xi)}\right\}_{i=1}^{N}$ by solving the optimization problem below:

$$\hat{\phi}_m = \arg\min_{\phi\in\Phi} \sum_{i=1}^{N} \frac{1}{2}\hat{h}_m(x_i)\left[-\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)} - \phi(x_i)\right]^2.$$

$$\hat{f}_{n_k}(x) = \alpha\hat{\phi}_m(x)$$

   (c) Update the model:

$$\hat{f}_{(m)}(x) = \hat{f}_{(m-1)}(x) + \hat{f}_m(x)$$

3. Output:

$$\hat{f}(x) = \hat{f}_{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x)$$

2.2.3 Automated machine learning

In order to convert input data into a predictive model, there are several manual stages that could be automated and eliminated with the use of auto machine learning (AutoML). Additionally, AutoML reduces the expertise needed to create correct models, allowing anybody, from machine learning experts to beginners, to utilize it. Moreover, AutoML accelerates difficult stages of the machine learning workflow by automating repeated operations, such as feature selection and engineering, model building and training and hyper-parameter optimization (Waring et al., 2020). The technique we use for automated machine learning is an algorithm called stacked ensemble.

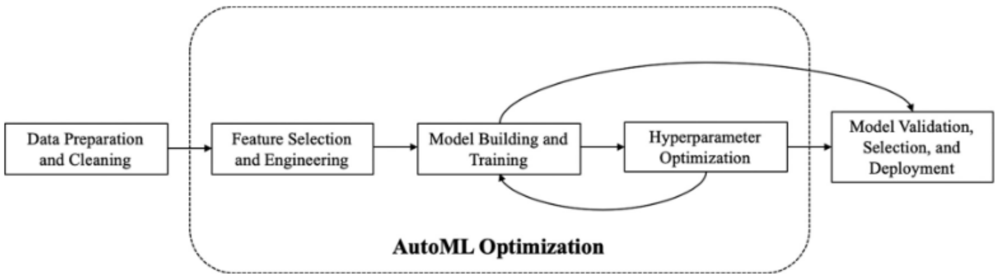

Data Preparation and Cleaning
Feature Selection and Engineering
Model Building and Training
Hyperparameter Optimization
Model Validation, Selection, and Deployment
AutoML Optimization

FIGURE 2.6: Typical components of a machine learning problem pipeline, adopted from: Waring et al. (2020).

Stacked ensemble Stacking, also called or stacked ensembles, is a class of algorithms that involves training a 'metalearner' to find the optimal combination of the base learners. Base learners in this case are machine learning alogithms such as XGBoost or deep neural networks. Metalearners are often generalizes linear models (GLM), but can be any type of algorithm as well. The theoretical steps to perform a stack are given below (Wolpert, 1992; Breiman, 1996; van der Laan et al., 2007; LeDell, 2015):

1. Ensemble initialization
   - Indicate the list (L) of base algorithms
   - Indicate the metal learning alrogithm
2. Ensemble training
   - Train all the base algorithms on the training data
   - Apply k-fold cross-validation to every trained base learners, then get the cross-validated predicted values from every algorithm.
   - A new N x L matrix is created by combining the N cross-validated predicted values from each of the L methods. The "level-one" data refers to this matrix and the original response vector. N stands for the training set's number of rows.
   - Utilize the level one data to train the metalearning algorithm. The L base learning models and the metalearning model make up the "ensemble model," which may be used to make predictions on a test set.
3. Predicting on new data
   - Generate predictions using the base learners.
   - To create the ensemble predictions, feed the predictions from the base learners into the metalearner.

AutoML with H2O H2O is a versatile open-source machine learning platform that supports integration with R, Python, Java, and Scala. It's designed to handle and scale large datasets efficiently. LeDell and Poirier (2020) presented H2O AutoML, an advanced automated machine learning solution within H2O. This solution automates the training process across a broad spectrum of potential models, including stacked ensembles. One of its standout features is the leaderboard, which ranks models based on various performance metrics and characteristics, such as training time and average prediction speed per row.

The efficacy of H2O AutoML is rooted in the efficient training capabilities of H2O's machine learning algorithms. Interestingly, H2O AutoML's performance often matches or even surpasses other frameworks that rely on intricate model tuning methods like Bayesian optimization or evolutionary algorithms, as noted by LeDell and Poirier (2020). The platform's ability to train a diverse set of algorithms, including Gradient Boosting Machines (GBMs), Random Forests, Deep Neural Networks, and Generalized Linear Models (GLMs), ensures a wide range of candidate models.

H2O AutoML has gained recognition in the machine learning community and is frequently employed as a benchmark to gauge the effectiveness of newly proposed machine learning structures (Kuo, 2020; LeDell Poirier, 2020). In this thesis, we leverage the AutoML functionality within the H2O package to validate the performance of our proposed models.

2.2.4 Data preprocessing techniques

One hot encoding One hot encoding is a technique used in data preprocessing to convert categorical data into a format suitable for machine learning algorithms. By employing this method, each unique category within a variable is represented as its own binary column. For a variable with 'n' observations that has 'd' unique categories, the result of one hot encoding will be 'd' binary columns. Each of these columns indicates the presence (1) or absence (0) of a specific category for every observation. As highlighted by Lantz (2013), one hot encoding provides a straightforward approach to transform categorical data into a binary matrix, making it compatible with many computational models

However, One-hot encoding categorical variables functions by mapping each category to a separate vector in a straightforward embedding. Using discrete entities, this technique converts each observation into a vector of zeros and a single one, that designates the observation's category. There are two main issues with the one-hot encoding method, first, The dimensionality of the converted vector becomes unmanageable for high cardinality variables, particularly those with numerous distinct categories. Second, The mapping is uninformed, meaning that similar categories are not positioned nearer to one another in the embedding space.

Embedding layers Embeddings are low-dimensional, continuously learnt vector representations of discrete variables used in neural networks. An embedding layer can be seen as a mapping layer of categorical or factorized features into a vector of continuous numbers (in this case between 0-1). Generally, neural network embeddings have 3 main purposes:

1. Within the embedding space, finding the nearest neighbour. Based on cluster categories we can then make recommendations.
2. To extract information of high cardinality variables and use this as input to a supervised machine learning model.
3. For visualisation of relations between different categories.

There are some design choices when making the embedding layer, namely, the loss function, the number of hidden layers, number of hidden units, number of epochs, number of batches and output dimensionality ($\mathbb{R}^k$).

In deep learning, categorical embedding is a popular method that has been used to forecast retail sales and recommender systems (Cheng et al. 2016; Guo and Berkhahn 2016). Moreover, Richman and Wuthrich (2018) employ embedding layers in the literature on actuarial science to capture regional characteristics in mortality forecasting. At last, Gabrielli et al. (2018) apply them to line-of-business elements in loss reserving.

2.2.5 Hyperparameter tuning

A Machine Learning model is defined as a mathematical model with a number of parameters that need to be derived from the data. We may fit the model parameters by using existing data to train a model. Hyperparameters, on the other hand, are a different class of parameters that cannot be directly learnt by routine training. Usually, they are established before to the start of the program itself. These parameters describe crucial model characteristics including complexity and learning rate. literature there are two main methods to determine the values of the hyper parameters of the models, namely grid search and random search (Liashchynskyi & Liashchynskyi, 2019).

Grid search Grid searches, which are the standard approach of hyperparameters optimization, essentially do a full search across a predetermined grid of the training algorithm's hyperparameters space. We might need to provide a boundary to utilize a grid search since the parameter space for some parameters in the machine learning method may include spaces with real or infinite values. High dimensional spaces are a problem for grid search, although it is frequently simple to parallelize since the hyperparameter values that the algorithm uses are typically independent of one another (Liashchynskyi & Liashchynskyi, 2019).

Random search Random search replaces the complete selection of all combinations (as done with a grid search) with a random selection subset. Discrete instances are easily applicable, but the technique is generalizable to continuous and mixed spaces. A random search can perform better than a grid search, particularly if only a few hyperparameters have an impact on how well the machine learning algorithm performs. Moreover, it can reduce computational effort significantly (Liashchynskyi & Liashchynskyi, 2019). However, because it has limited coverage of the input space, a better solution may be missed. a form of random sampling that tackles this problem is Latin hypercube sampling (LHS). Latin hypercube sampling aims to spread a group of samples almost uniformly over the design space to increase the coverage of the input space (Du et al. 2021; Iman, 2008).

2.2.6 Machine learning in loss reserving literature

Research in the domain of machine learning for predicting loss reserves has been somewhat limited. However, the field of deep learning, with its success across various disciplines, has started to garner attention in loss reserving studies, as indicated by Lecun et al. (2015) and Kuo (2019). This literature can be categorized into two main types: aggregated claim data structured in claim triangles and individual claim data. This review will focus on the former, as it aligns with the data used in this thesis.

Wüthrich (2018) explores the application of neural networks in chain-ladder reserving. They begin with the Mack chain ladder method as a foundation but extend its simplified assumptions (as outlined in Equations 2.1, 2.2, 2.3) to include individual claims information. Heterogeneous claims data are utilized as features in a neural network model to predict loss development factors (LDFs). Specifically, Wüthrich (2018) enhances homogeneous LDFs by incorporating heterogeneous individual claims feature information. This expanded approach allows for the analysis of specific claims and the adaptation to changing portfolio compositions. However, Wüthrich acknowledges that this is an initial step in a broader modeling endeavor and notes that only static feature information is considered, with no account for dynamic characteristics, which would require complex multidimensional stochastic process modeling.

Gabrielli et al. (2018) and Gabrielli (2019) propose a method of initializing neural networks to align with classical models, like the over-dispersed Poisson model, before training. They integrate classical parametric loss reserving models into neural network structures, taking into account claim counts and amounts. Starting with two separate over-dispersed Poisson (ccODP) models for both claim counts and amounts, they embed these into a neural network architecture. Additionally, they develop a boosting machine that enables joint modeling and learning of claim counts and amounts, beyond independent ccODP models, using a synthetic dataset spanning six lines of business. Their findings suggest this model outperforms the ccODP model for both claim numbers and amounts, and in some instances, it surpasses the single neural network double over-dispersed Poisson (NNDODP) model. Gabrielli et al. conclude that the multi-triangle model, representing all business lines within a single neural network, is the preferred approach, recommending the separate fitting of lines of business.

Kuo (2019) introduces a deep neural network-based approach to loss reserving that accommodates heterogeneous inputs and combines modeling of outstanding paid losses and claims. They utilize paid loss and case reserve history as time series inputs, employing gated recurrent units (GRU) for processing. The dataset, divided per line of business into 50 triangles, incorporates a company code embedding layer, and models are trained separately for each business line. Kuo posits that this multitask learning approach, as outlined by Caruana (1997), could enhance predictive capabilities by uncovering hidden features relevant to predicting paid losses. Kuo's findings indicate that deep neural network architecture can match the performance of existing stochastic reserving methods without expert input, suggesting potential for automation in model updating and report generation.

Song and Heo (2022) critique traditional loss reserve error evaluation methods for their biased estimation approach, lack of understanding of interactions and higher-order term relations, and reliance on linearity and correlated variables. They introduce a combined unsupervised-supervised method, employing cluster analysis and machine learning algorithms like boosting and support vector machine (SVM), along with hierarchical clustering and artificial neural networks (ANN). Their research reveals that distinct foundational factors influence each cluster's loss reserve forecasting, corroborating the findings of Kuo (2019), and Gabrielli et al. that segmenting data into separate lines of business enhances predictive accuracy. Additionally, they find that ANN and boosting models reduce root mean square error (RMSE) in loss reserve estimation and that historical data can be effective in predicting future loss reserves.

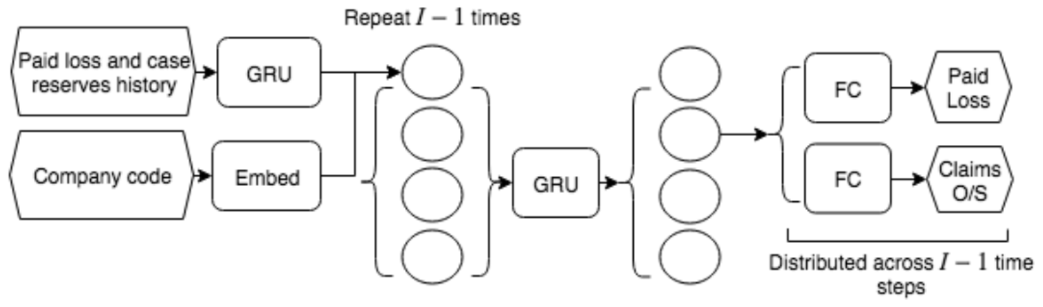

Repeat $I-1$ times
Paid loss and case reserves history
Company code
GRU
Embed
GRU
FC
FC
Paid Loss
Claims O/S
Distributed across $I-1$ time steps

FIGURE 2.7: Model architecture of the deep triangle model proposed by Kuo (2019). Embed denotes embedding layer, GRU denotes gated recurrent unit, FC denotes fully connected layer.

2.3 Recoveries

Recoveries refer to the amounts insurers receive from third parties, typically as reimbursements for claims the insurer has already paid. These can come from various sources, including subrogation, salvage, reinsurance, and deductibles. In the context of loss reserving, recoveries play a crucial role as they can significantly reduce the net amount of claims that an insurer has to pay. Therefore, accurately predicting recoveries is essential for insurers to estimate their future liabilities and maintain financial stability.

Understanding the potential for recoveries can help insurers manage risks more effectively. For instance, if an insurer knows that certain types of claims have a high likelihood of recovery, the insurer might be more willing to underwrite such risks. Moreover, recoveries can have a significant financial impact on an insurer's balance sheet. By accurately predicting recoveries, insurers can better manage their capital and ensure they have sufficient funds to meet future claims. Furthermore, accurate recovery predictions can influence pricing decisions. If an insurer consistently recovers a significant proportion of its claims, it might offer more competitive premiums to its policyholders.

Recent advancements in the field have introduced innovative methodologies to enhance the predictive performance of loss reserving models. Avanzi et al. (2022) proposed a framework to combine multiple stochastic loss reserving models, tailoring it for features inherent to reserving data, such as accident, development, and calendar effects. This ensemble approach could potentially be extended to consider recoveries, offering a more comprehensive prediction model.

Another significant contribution comes from Okine et al. (2021), who introduced a joint modeling framework that incorporates longitudinal payments of a claim into the intensity process of claim settlement. Their research suggests that overlooking the association between the payment process and the settlement process could introduce bias in predicting outstanding payments. This joint model framework, which considers both payments and settlements, could be pivotal in understanding recoveries.

The potential of deep learning in loss reserving has also been explored. Chaoubi et al. (2022) introduced a micro-level reserving approach using a Long Short-Term Memory (LSTM) neural network. This network not only classifies whether there's a payment or a recovery but also predicts the corresponding non-zero amount, emphasizing the potential of deep learning in capturing granular information for loss reserving.

Lastly, Woundjiagué et al. (2019) presented a hybrid log-Poisson regression model optimized for loss reserving. Their model, which uses a quadratic optimization program, has demonstrated superior predictive performance for incremental payments compared to traditional models. Such advancements could be instrumental in refining predictions related to recoveries.

In conclusion, while the literature on predicting recoveries in the context of loss reserving is still evolving, the methodologies and frameworks being developed offer promising avenues for more accurate and comprehensive predictions. As the insurance industry continues to leverage advanced analytics, there's a growing opportunity to focus on recoveries and develop sophisticated methods to predict them accurately.

3 Methodology

In our methodology chapter, we first outline the research set-up to provide a foundation for our study. We then delve into the data sources and collection methods we've employed. Lastly, we discuss the performance measures we use to evaluate and interpret our research findings

Contents

3.1 Research set-up

Our research aims to enhance the performance of predicting the loss reserve for non-life insurers.. For this purpose, we apply traditional models and machine learning models to predict the loss reserve of 178 claim triangles. We propose different machine learning architectures based on the findings in literature. Next, we compare the results, based on our predictability measures, of the different models and architectures. For this purpose we have split the data into a train and test set. In the following subsections, we first discuss the data and their characteristics. Thereafter, we define the train-test split. Finally, we define and discuss performance measures, and describe the comparison between the selected models.

3.2 Data

We employ the Schedule P triangles from the National Association of Insurance Commissioners (NAIC) (Meyers et al. 2011). The triangles include incurred claim and paid claim data with a 10-year development period for each accident year. The data set corresponds to claims from the accident years 1988 to 1997. The data includes the actual paid losses. This means that also the observed ultimate loss is given. Moreover, it is assumed that the all claims have been settled. The information is combined to create accident year-development year records in Schedule P data. Meyers (2015) provides a full description of the data set construction process. Following Meyers (2015) and Kuo (2019), we limit our analysis to a subset of the data that includes 50 firms in each of the four business lines of commercial auto, private personal car, workers' compensation, and other liabilities. This is done to make it easier to compare findings to earlier ones. Moreover, since dealing with missing data is out of our scope, we remove all triangles with missing data. This leaves us with a total of 178 claim triangles. Line of business, company code, accident year, development lag, incurred loss, cumulative paid loss, and net earned premium are the variables from the data set we use in our analysis. In this study, claims outstanding are calculated as incurred loss subtracted by cumulative paid loss. Company code indicates which insurer the triangle origins from, and is a categorical variable. As input for our models similar to Kuo (2019) we divide the cumulative (and incremental) paid losses by the net premium collected. This is ratio is called the observed (predicted) loss ratio and will be denoted as $z_{i,j}$ ($\hat{z}_{i,j}$). According to Kuo (2019) 'working with loss ratios makes training more tractable by normalizing values into a similar scale'. After the analysis, the predicted outcome is then again multiplied by the net premium of that year.

Since we have data from 1988 to 1997 we have the original claim triangles and the completed claim triangles. Examples of these claim triangles of our data set are given in and Table 3.1 and 3.2 respectively.

Single company cumulative claim triangle										
	Development lag									
Accident year	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1988	0,073	0,124	0,144	0,159	0,170	0,178	0,183	0,184	0,187	0,187
1989	0,118	0,226	0,260	0,300	0,382	0,395	0,401	0,404	0,406	NA
1990	0,178	0,302	0,451	0,538	0,546	0,553	0,557	0,560	NA	NA
1991	0,344	0,505	0,541	0,637	0,652	0,719	0,745	NA	NA	NA
1992	0,121	0,176	0,242	0,429	0,460	0,486	NA	NA	NA	NA
1993	0,131	0,240	0,435	0,577	0,612	NA	NA	NA	NA	NA
1994	0,145	0,255	0,411	0,476	NA	NA	NA	NA	NA	NA
1995	0,165	0,264	0,282	NA	NA	NA	NA	NA	NA	NA
1996	0,128	0,185	NA	NA	NA	NA	NA	NA	NA	NA
1997	0,167	NA	NA	NA	NA	NA	NA	NA	NA	NA

TABLE 3.1: Single company cumulative claim triangle, group code = 10022, LoB = commercial auto.

Single company completed cumulative claim triangle										
	Development lag									
Accident year	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1988	0,073	0,124	0,144	0,159	0,170	0,178	0,183	0,184	0,187	0,187
1989	0,118	0,226	0,260	0,300	0,382	0,395	0,401	0,404	0,406	0,406
1990	0,178	0,302	0,451	0,538	0,546	0,553	0,557	0,560	0,562	0,563
1991	0,344	0,505	0,541	0,637	0,652	0,719	0,745	0,746	0,747	0,748
1992	0,121	0,176	0,242	0,429	0,460	0,486	0,489	0,489	0,489	0,491
1993	0,131	0,240	0,435	0,577	0,612	0,618	0,621	0,684	0,686	0,686
1994	0,145	0,255	0,411	0,476	0,498	0,556	0,557	0,572	0,572	0,573
1995	0,165	0,264	0,282	0,299	0,306	0,307	0,310	0,309	0,309	0,309
1996	0,128	0,185	0,243	0,379	0,385	0,388	0,387	0,387	0,387	0,388
1997	0,167	0,283	0,301	0,384	0,407	0,421	0,427	0,427	0,427	0,427

TABLE 3.2: Single company completed cumulative claim triangle, group code = 10022, LoB = commercial auto.

An overview of the descriptive statistics of the observed incremental and cumulative loss ratios is given in Table 3.3. A histogram of the incremental and cumulative losses is given in Figures 3.1, 3.2 respectively. In Table 3.3 We can see that we have 17800 data points and that the mean of the incremental losses is 6% of the respective net premiums (the loss ratio). Moreover, the incremental losses have a standard deviation of 10%. The median (2%) is lower than the mean which means that the data is positively skewed. This is also in line with the findings of Figure 3.1 and the skewness (2,16). The minimum (maximum) value of the incremental loss ratio is -131% (153%). This is quite an extreme value since this means that there was a recovery (loss) of more than the net premium within a one year increment. However, when we look at the claim triangles of these extreme values we find that they are both in the same triangle (LoB: commercial auto, group code: 29440) within the same accident year and in consecutive development periods. This indicates that this is the result of an (human) error. The kurtosis of the incremental loss ratios is 14,19 this means that these loss ratios tend to have a fat right-sided tail or outliers, this is in line with findings of Figure 3.1. At last, the data set contains 756 negative incremental loss ratios, this means 4,2% of the observed losses were recoveries.

When looking at the cumulative loss ratios we can see that the mean, standard deviation and median are higher, since the cumulative losses are the sum of the incremental losses and the incremental losses are mostly positive, this is a logical conclusion. When looking at the histogram in figure 3.2, the skewness and kurtosis, we can see the data is positively skewed and has a right-sided tail. However the tail is flat relative to the incremental losses. Moreover, there are no negative data points when considering the cumulative loss ratios.

Data describtion data set		
	Incremental loss	**Cumulative loss**
Number of data points	17800	17800
Mean	0,06	0,54
Standard deviation	0,10	0,27
Median	0,02	0,54
Min	-1,31	0
Max	1,53	3,27
Range	2,84	3,27
Skew	2,16	0,84
Kurtosis	14,19	4,49
Number of negative data points	756	0
Pct. of negative data points	4,2%	0

TABLE 3.3: Describing statistics of incremental and cumulative observed loss ratios.

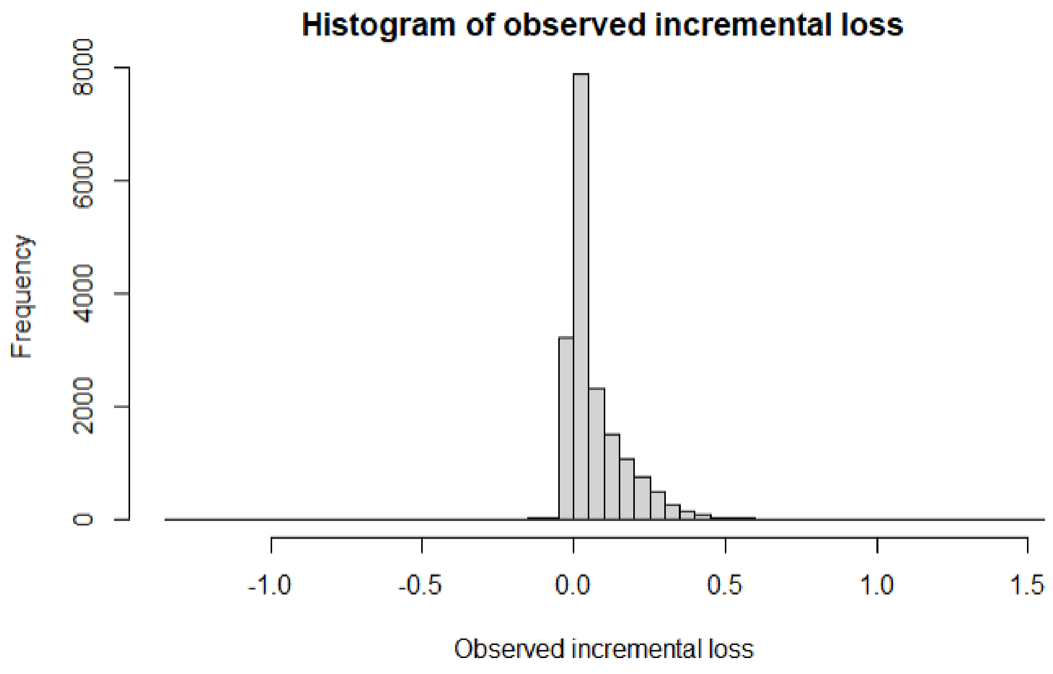


FIGURE 3.1: Histogram of the observed incremental losses provided by the data set provided by NAIC.

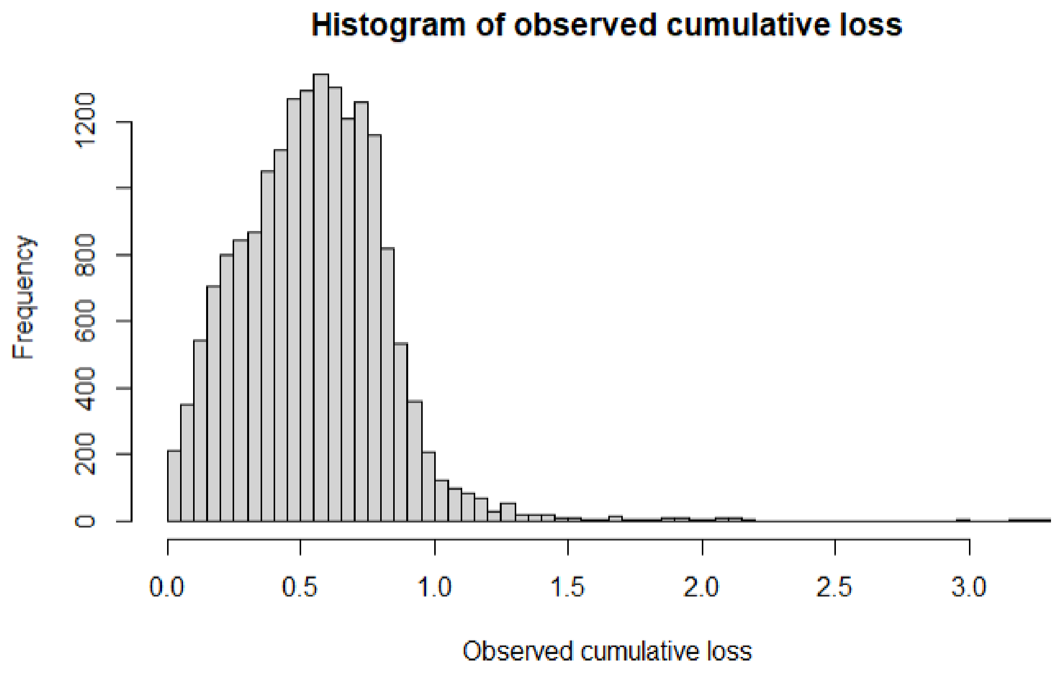


FIGURE 3.2: Histogram of the observed cumulative losses provided by the data set provided by NAIC.

3.2.1 Line of business

There are four lines of business we consider in this research, namely, commercial auto (CA), private passenger auto (PPA), workers' compensation (WC) and other liability (OL). The number of triangles per LoB are given in Table 3.4. These LoBs might have different characteristics, therefore we also look at the descriptive statistics and histogram of the incremental losses of each LoB. The descriptive statistics per LoB are given in Table 3.4, the histograms of the incremental loss ratios per line of business are given in Figures 3.3, 3.4, 3.5 and 3.6. First, we can see that other liability has significantly less data than the other lines of business, this might affect the predictability of this line of business. This is because some of the triangles had to be eliminated because they were out of scope for this study. Moreover, we can see a slight difference in mean and standard deviation between the triangles. For all lines of business, the median is, however, nearly identical. This means the distributions are slightly differently skewed. The kurtosis for the commercial auto and other liability lines of business are almost identical and significantly higher than the other LoBs. This means the distribution of the loss ratios of CA and OL LoBs have heavier tails and more outliers. We can also see that CA PPA and OL have outliers when looking at the minimum and maximum values. At last, we can see that PPA has the highest percentage of recoveries whereas WC has the lowest.

When looking at the different Histograms, we can see that for each LoB most observations are centred around 0, with a right-sided tail, and some negative outliers. We can see that CA and OL have heavier tails as is in line with the findings in Table 3.4. Moreover, an interesting observation is that the PPA line of business has a local peak at a loss ratio of 0.25.

Data describtion per LoB				
	CA	**PPA**	**WC**	**OL**
Number of data points	4700	4900	4600	3600
Number of triangles	47	49	46	36
Mean	0.07	0.08	0.06	0.05
Standard deviation	0.10	0.11	0.08	0.08
Median	0.02	0.02	0.02	0.02
Min	-1.31	-0.60	-0.21	-0.58
Max	1.53	0.95	0.61	1.08
Range	2.84	1.55	0.82	1.66
Skew	2.36	1.63	1.65	3.28
Kurtsosis	26.19	3.10	2.97	26.65
Number of negative data points	209	283	143	121
Pct. of negative data points	4.4%	5.8%	3.1%	3.4%

TABLE 3.4: Describing statistics of incremental observed loss ratios per line of business.

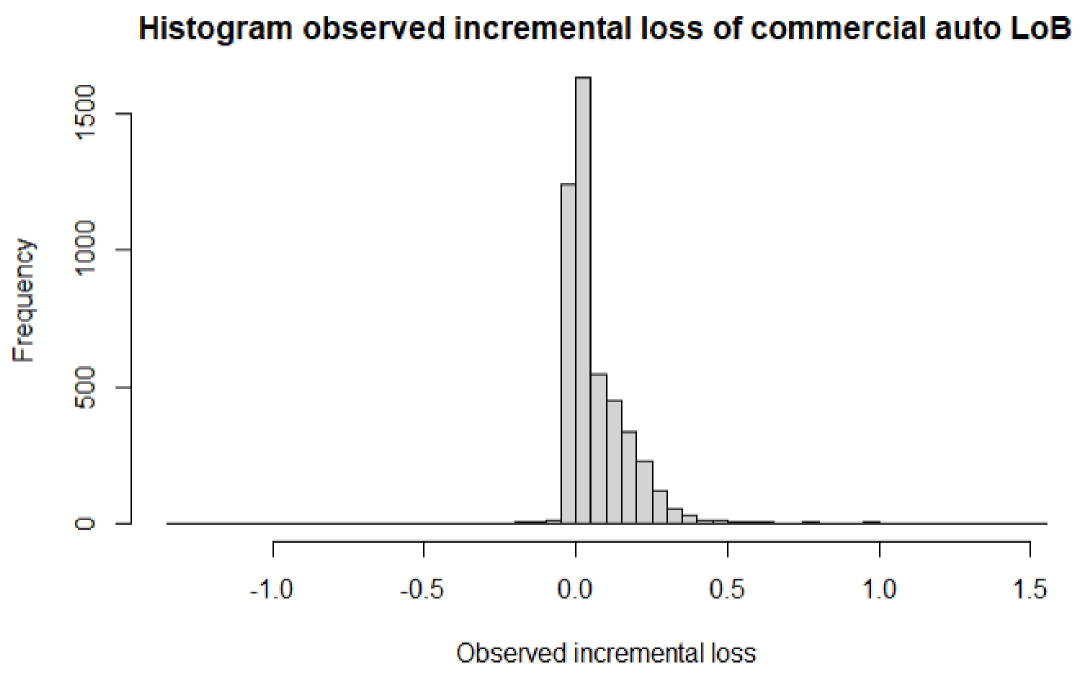


FIGURE 3.3: Histogram of the observed incremental loss of the commercial auto line of business.

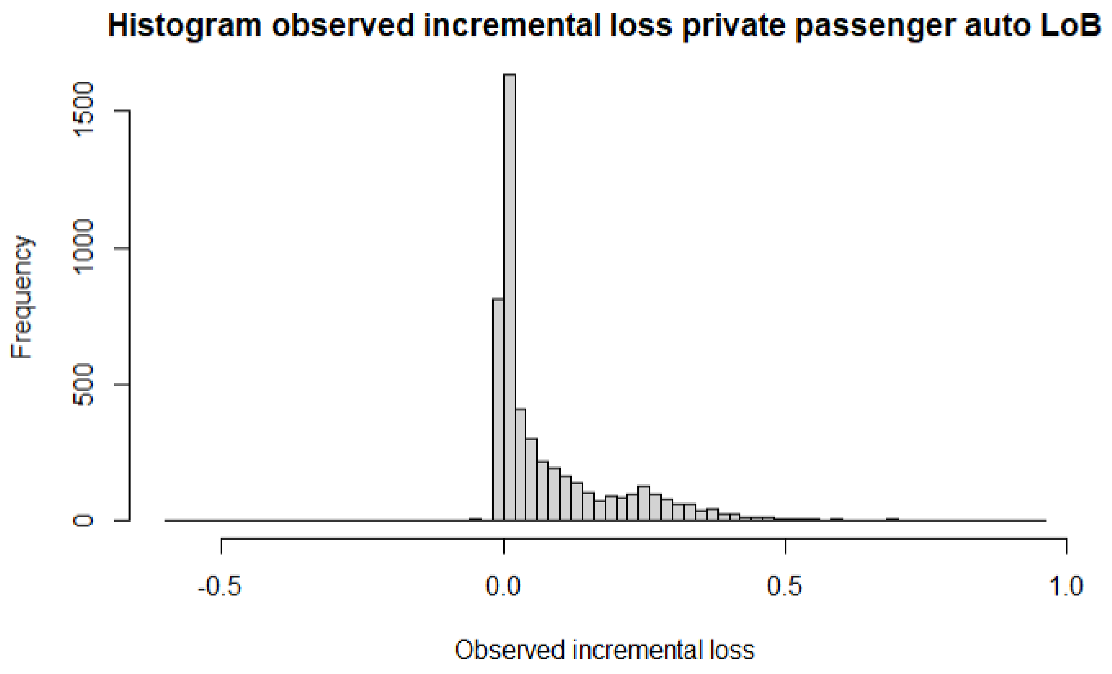


FIGURE 3.4: Histogram of the observed incremental loss of the private passenger auto line of business.

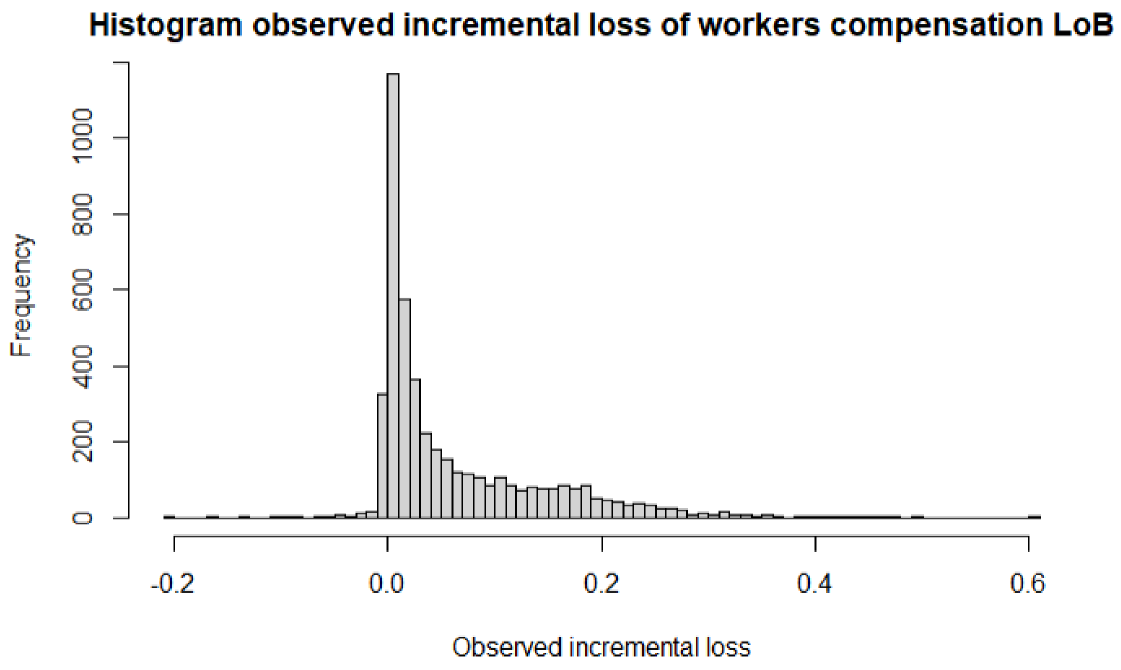


FIGURE 3.5: Histogram of the observed incremental loss of the workers' compensation line of business.

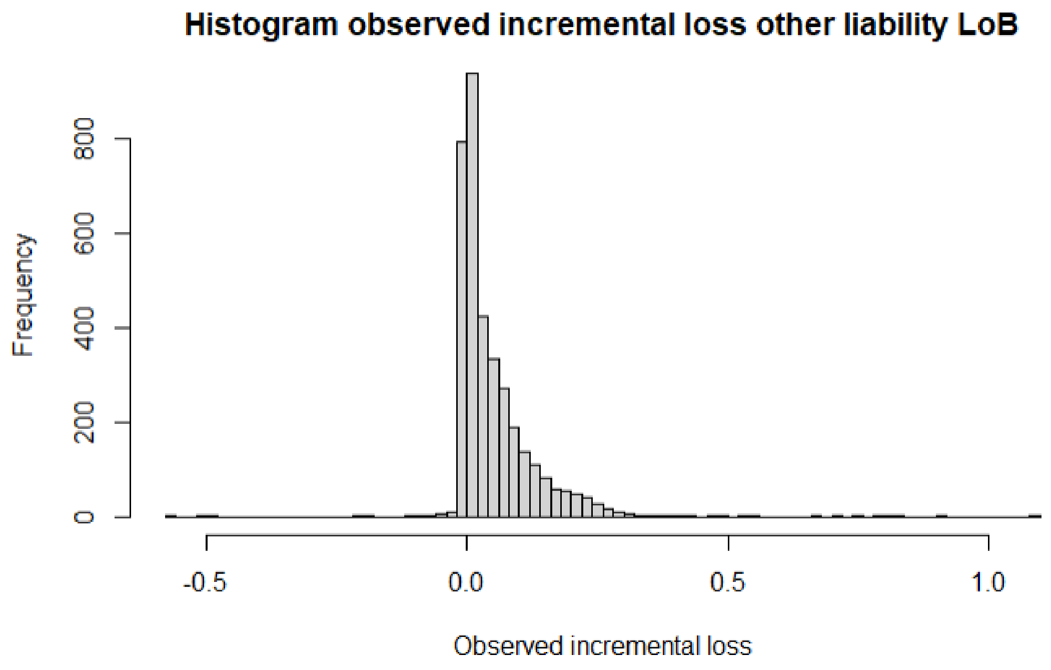


FIGURE 3.6: Histogram of the observed incremental loss of the other liability line of business.

3.2.2 Train-test split

As previously, we define indices $1 \leq i \leq I$ as accident years and $1 \leq j \leq J$ as development years. Furthermore, let $z_{i,j}$ denote the incremental paid loss. The observed data are then available to us at the conclusion of calendar year i is: $\{z_{i,j} : i = 1, ..., I; j = 1, ..., I - i + 1\}$. Finding predictions for future values $\{\hat{z}_{i,j} : i = 2, ..., I; j = i + 1, ..., I\}$ is the next step. Ultimately, we can calculate the estimated loss reserve for each accident year $i = 1, ..., I$ by solving:

$$\hat{R}_i = \sum_{j=I-i+2}^{I} \hat{z}_{i,j} \tag{3.1}$$

The estimated loss reserve per company (c) is given by:

$$\hat{R}_c = \sum_{i=2}^{I} \hat{R}_i \tag{3.2}$$

With our data set we use end of year 1997 as training set, the rest of development periods is used as test set. We give an example of a single triangle train test-split in Table 3.5.

Single company train-test split										
	Development lag									
Accident year	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1988	Train	Train	Train	Train	Train	Train	Train	Train	Train	Train
1989	Train	Train	Train	Train	Train	Train	Train	Train	Train	Test
1990	Train	Train	Train	Train	Train	Train	Train	Train	Test	Test
1991	Train	Train	Train	Train	Train	Train	Train	Test	Test	Test
1992	Train	Train	Train	Train	Train	Train	Test	Test	Test	Test
1993	Train	Train	Train	Train	Train	Test	Test	Test	Test	Test
1994	Train	Train	Train	Train	Test	Test	Test	Test	Test	Test
1995	Train	Train	Train	Test	Test	Test	Test	Test	Test	Test
1996	Train	Train	Test	Test	Test	Test	Test	Test	Test	Test
1997	Train	Test	Test	Test	Test	Test	Test	Test	Test	Test

TABLE 3.5: An example of a single company train test-split

Furthermore, we can look at the difference in data characteristics between train and test data. Since the training exists of claims with a relatively low development lag and test data claims have a relatively high development lag, we can expect some differences. Table 3.6 gives the descriptive statistics of the train and test data. Figures 3.7 and 3.8 give the histograms of the train and test observed incremental loss ratios respectively. We can see that the mean and standard deviation of the train set is higher. This is in line with expectations since when we consider the claim timeline most of the losses occur sooner in the development period. Moreover, we can see that the median of the test data is 0, with only a 6.5% of negative data we can conclude that in a majority of the years in the test data there were no claims or recoveries. This can also be concluded from Figure 3.8. Furthermore, we can see that the test data have a heavier tail, while the train data have the most extreme cases (losses and recoveries). Finally, we can see that the test data have a higher percentage of recoveries, this is consistent with the claim timeline since most of the recoveries happen later in the timeline.

Data describtion		
	Train data	**Test data**
Number of data points	9790	8010
Mean	0.1	0,02
Standard deviation	0.11	0.05
Median	0.07	0,00
Min	-1.31	-0,6
Max	1.53	0,85
Range	2.84	1,45
Skew	1.58	4,17
Kurtsosis	11.35	48.09
Number of negative data points	233	523
Pct. of negative data points	2.4%	6.5%

TABLE 3.6: Describing statistics of incremental observed loss ratios of the train and test set.

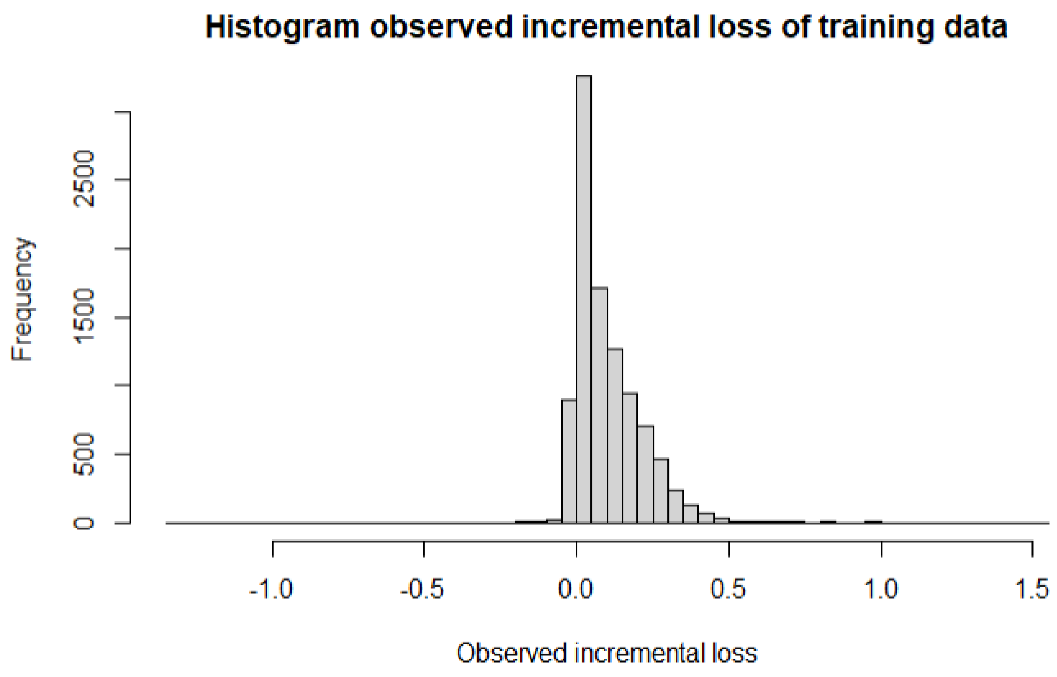


FIGURE 3.7: Histogram of the observed incremental loss of the commercial auto line of business.

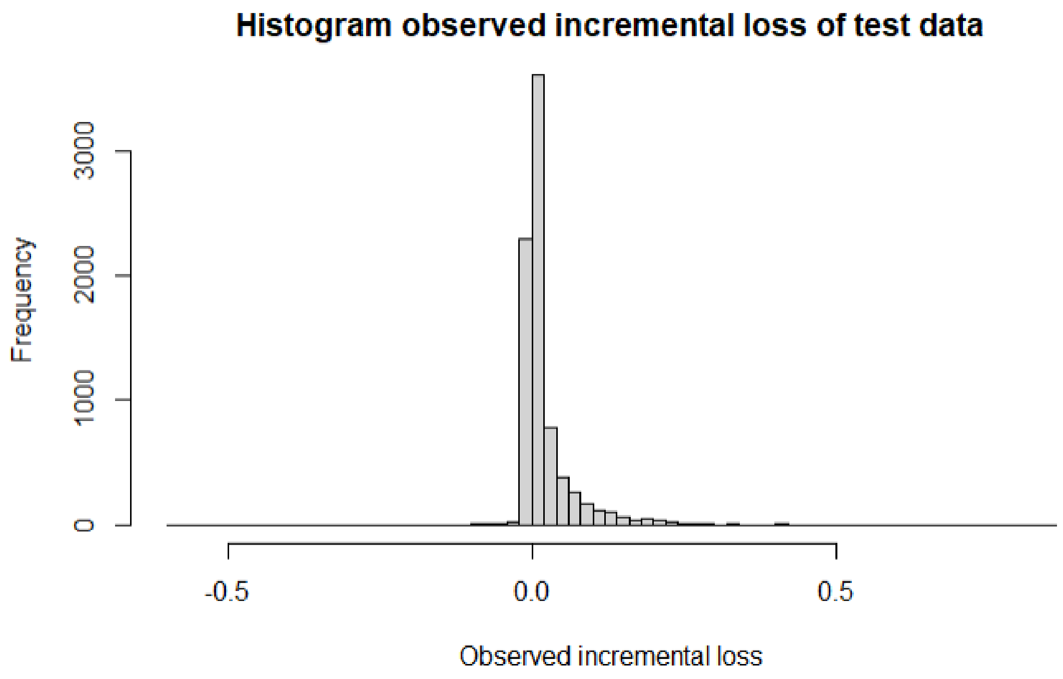


FIGURE 3.8: Histogram of the observed incremental loss of the commercial auto line of business.

Line of business For each LoB, we aggregate the training input. For example, for commercial auto LoB we use 47 triangles as training, since we have 55 training data points and 45 test training points per triangle we have in total 2585 training data points and 2115 test data points. The number of train and test data points per line of business are shown in Table 3.7.

Number of data points per LoB		
Line of business	**Train data points**	**Test data points**
Commercial auto	2585	2115
Private passenger auto	2695	2205
Workers compensation	2530	2070
Other liability	1980	1620
Total	**9790**	**8010**

TABLE 3.7: The number of train and test data points per line of business.

3.2.3 Response and predictor variables

In this thesis, each training sample is associated with an accident and development year pair which we call a cell ($z_{i,j}$). The sample refers to the loss ratio at the given accident year and development year (i, j), and denotes the response variable.

The predictor for the sample contains multiple components. First is the observed history as of the end of the calendar year associated with the cell which is given by: $\{z_{i,1}, z_{i,2}, ..., z_{i,j-1}\}$. As result for each cell, there are 9 input features of historical loss ratios, since there are at most 9 observations. Moreover, we give a value of 0 if there is no observed history. In other words, we make an effort to forecast future development of the accident year's observed history of loss ratios as of each evaluation date for which we have data. We use the observed history of the training set as input feature for our models. Ultimately, we are interested in predicting the loss ratios of the test set.

The next predictors of the sample are the factorized data, namely, accident year (10 unique values), development lag (10 unique values), calendar year (19 unique values), line of business (4 unique values)and group code (108 unique values). Depending on the model architecture and preprocessing techniques used these input features are altered. More elaboration on this will be in the Proposed models' chapter.

3.3 Performance measure

We want to assess the predictability of the loss reserve of a non-life insurer. Therefore we will consider the performance measure on a triangle (company) level. The estimated loss reserve of the training set on company level is given in Equation 3.2. We will compare the estimated loss reserve with the observed loss reserve of the training set. We define R^* as the observed loss reserve of the test set.

$$R_i^* = \sum_{j=I-i+2}^{I} z_{i,j} \tag{3.3}$$

The observed loss reserve of the test set on company (c) level is given by:

$$R_c^* = \sum_{i=2}^{I} R_i^* \tag{3.4}$$

As predictive performance measures, we use similarly as to Kuo (2019) root mean square error (RSME) and mean absolute error (MAE).

RMSE is the mean squared difference between the predicted value and the observed value. MAE is the mean of all absolute errors between the predicted and observed values. RMSE and MAE generate results in terms of the unit of the forecasted value, which is one of its benefits. For instance, using RMSE in a model to forecast the price of a house would provide the error in terms of house price, making it easier for end users to comprehend the performance of the model. In our case we look at loss reserve expressed in loss ratios, therefore the result will be in the same measurement unit. In this study we calculate the RSME and MAE per line of business. Given an n_{LoB} which is the number of triangles within a line of business we can calculate the RSME and MAE with the Equations below:

$$RSME_{LoB} = \sqrt{\frac{\sum_{c=1}^{n_{LoB}} (\hat{R}_c - R_c^*)^2}{n_{LoB}}} \tag{3.5}$$

$$MAE_{LoB} = \frac{\sum_{c=1}^{n_{LoB}} |\hat{R}_c - R_c^*|}{n_{LoB}} \tag{3.6}$$

3.3.1 Recoveries

In loss reserve prediction, recoveries refer to money gained back by the insurer. We want to know how accurately our models can predict the recoveries. For this we again use a RSME and MAE but ony consider the recoveries. In terms of claim $z_{i,j}$ a recovery is any $z_{i,j} < 0$. Therefore we define observed ($z^-_{i,j}$) and predicted ($\hat{z}^-_{i,j}$) as follows:

$$z^-_{i,j} = \begin{cases} z_{i,j}, & z_{i,j} < 0 \\ 0, & z_{i,j} \geq 0. \end{cases} \tag{3.7}$$

$$\hat{z}^-_{i,j} = \begin{cases} \hat{z}_{i,j}, & \hat{z}_{i,j} < 0 \\ 0, & \hat{z}_{i,j} \geq 0. \end{cases} \tag{3.8}$$

After defining the recoveries we can define the total (observed and predicted) loss reserve an insurer recovers per accident year:

$$R^-_i = \sum_{j=I-i+2}^{I} z^-_{i,j} \tag{3.9}$$

$$R^-_c = \sum_{i=2}^{I} R^-_i \tag{3.10}$$

Thereafter, we can define the total (observed and predicted) loss reserve an insurer recovers on company level (c).

$$\hat{R}^-_i = \sum_{j=I-i+2}^{I} \hat{z}^-_{i,j} \tag{3.11}$$

$$\hat{R}^-_c = \sum_{i=2}^{I} \hat{R}^-_i \tag{3.12}$$

At last we can define the RSME and MAE for the loss reserve recovered per line of business as follows:

$$RSME_{LoB} = \sqrt{\frac{\sum_{c=1}^{n_{LoB}} (\hat{R}^-_c - R^-_c)^2}{n_{LoB}}} \tag{3.13}$$

$$MAE_{LoB} = \frac{\sum_{c=1}^{n_{LoB}} |\hat{R}^-_c - R^-_c|}{n_{LoB}} \tag{3.14}$$

When analyzing the recoveries, it may be useful to assess not only how well the model predicts the magnitude of the recovery but also whether or not a recovery will take place within a triangle. Since recoveries frequently come from prior errors, it might be useful for actuaries to be aware when one is likely to occur inside a triangle. When it is possible to recognize that a recovery is taking place, one might investigate previous statements made by that particular triangle to seek for inaccuracies. As a result, we want to introduce the accuracy of predicting whether or not a recovery will take place within a triangle as metric. We define an observed (predicted) recovery taking place within the triangle if $R_c^- < 0$ ($\hat{R}_c^- < 0$) . If $R_c^- = 0$ ($\hat{R}_c^- = 0$) then there is no observed (predicted) recovery within the triangle. If the observation matches the prediction and there is a recovery in the triangle it is called a true positive (TP). If the observation matches the prediction and there is no recovery we call it a true negative (TN). If the prediction is that there is a recovery and the observation is that there is no recovery it is called a false positive (FP). At last, if we predict there is no recovery and there is a observed recovery it is called a false negative (FN). We define the accuracy metric as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.15}$$

4 Proposed models

In this chapter, the various machine learning model architectures developed and evaluated in this thesis are discussed. The focus is on proposing three distinct XGBoost model architectures and two autoML architectures.

The initial approach involves a base XGBoost algorithm, utilizing the complete dataset as input for the first architecture. Building upon this, elements informed by the literature review in Chapter 2 are incorporated. The first enhancement involves segmenting the dataset based on LoBs, a strategy recommended by Kuo (2019), Mayers (2015), Gabrielli et al. (2018), Gabrielli (2019), and Song and Heo (2022). Following this, the XGBoost architecture is further refined by introducing a company code embedding layer, as suggested by Kuo (2019). Finally, the architecture is augmented with Latin hypercube sampling for hyperparameter tuning.

For the autoML models, learning will be conducted on both the full dataset and the LoB-segregated datasets. These autoML models are introduced for validation of the proposed architectures and to uncover additional insights into which machine learning models are most effective for predicting loss reserves.

The primary emphasis is placed on XGBoost models, as there appears to be a lack of research using XGBoost with aggregated claims data in the existing literature. XGBoost has demonstrated superior performance across numerous research domains, outshining most other machine learning models. It's noteworthy that the current body of literature on loss reserve forecasting predominantly focuses on neural network applications. This thesis seeks to explore and expand upon the potential of XGBoost in this context.

Contents

4.1 XGBoost architectures

4.1.1 XGBoost benchmark

We use a base XGBoost model without elaborate data preprocessing techniques or hyperparameter tuning on the complete data set as a benchmark for our other models. We want to see if the addition of the proposed techniques and LoB splitting improves the predictability of the loss reserve. We will first discuss the predictor variables and then give an overview of the model architecture.

For this model architecture, we have two kinds of input features: the observed history of the loss ratios until the end of the calendar year, and the factorized data. Since we use this model as a benchmark all factorized data (accident year, development lag, calendar year, line of business and group code) are used as input. We use one hot encoding for our factorized data resulting in 151 (10+10+19+4+108) input features all existing of binary values.

Figure 4.1 gives an overview of the model architecture.

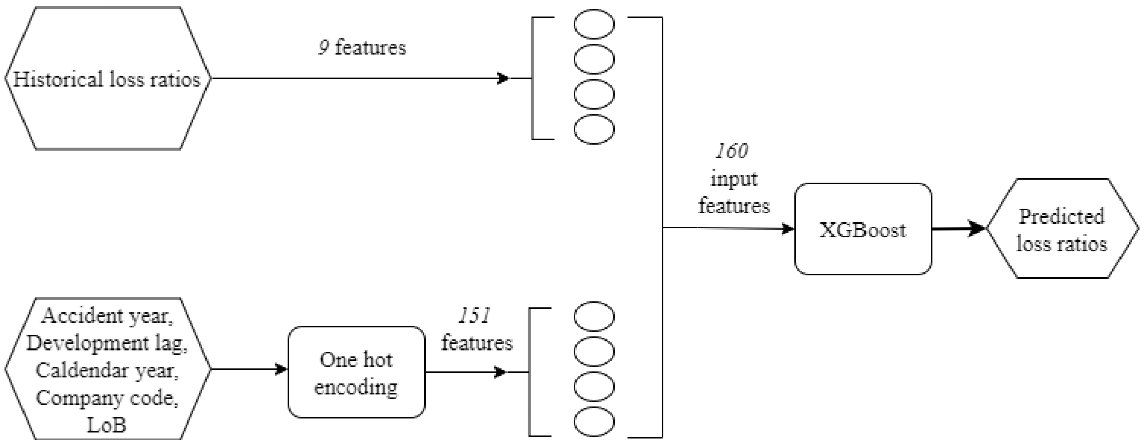


FIGURE 4.1: The base XGBoost architecture.

As discussed in the literature review, Kuo (2019), Gabrielli (2019), Gabrielli et al. (2018), Meyers (2015) and Song and Heo (2022) have found evidence that splitting the data set into groups based on the line of business of the insurer and learning separately on these can improve the predictability of the loss reserve. Therefore, we also apply this to our XGBoost-based approach.

The input features are similar to the base XGBoost architecture, however instead of treating LoB as factorized data, we create four different data sets based on the four lines of business (commercial auto, private passenger, workers' compensation and other liability). A separate XGBoost model is fit on each of the data sets and used to predict the data of the test set. Since we split the data set into four datasets this also partly tackles the problem of the high cardinality of the company code feature. Since there are no companies with more than one triangle in a line of business the number of features associated with the company code is simply the number of triangles within the LoB data set (n_{LoB}). An overview of the model architecture for a single LoB (commercial auto) is given in Figure 4.2. Note that for the other LoBs the model architecture looks similar, with as only difference being the input data set and the number of features associated with the group codes.

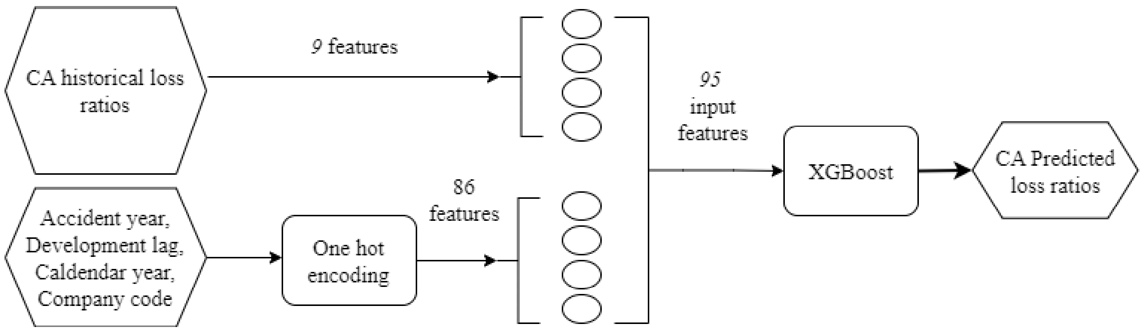

CA historical loss ratios
9 features
95 input features
XGBoost
CA Predicted loss ratios
Accident year, Development lag, Caldendar year, Company code
One hot encoding
86 features

FIGURE 4.2: XGBoost architecture of the commercial auto line of business.

4.1.2 Proposed XGBoost architecture

The proposed XGBoost architecture we combine the embedding layer and Latin hypercube sample hyperparameter tuning with our benchmark model, we will give an detailed overview of the used methodologies and models below.

Company code embedding The group code feature has a high cardinality, namely, 108 (or n_{LoB} for the LoB split data sets) unique group codes, one hot encoding is perhaps not the right technique to extract information from this feature. Moreover, there might be relations between the different companies that cannot be captured with one hot encoding (Kuo, 2019). Therefore, as suggested by Kuo (2019) and Gabriellie (2019), we use an embedding layer to transform our group code data.

For the embedding layer, group code and incremental loss ratio are our response and predictor variable respectively. In the context of our claims forecasting problem, similar companies are mapped to vectors that are relatively close to one another in terms of Euclidean distance. We train the model separately on the training data of each line of business. This results in a unique k-dimensional vector for each group code and LoB combination. For example, recalling the triangle example given in Table 1.1, each cell in this triangle would be linked to the k-dimensional vector. This is ultimately used as input features for the XGBoost model. Figure 4.3 gives an overview of the embedding process.

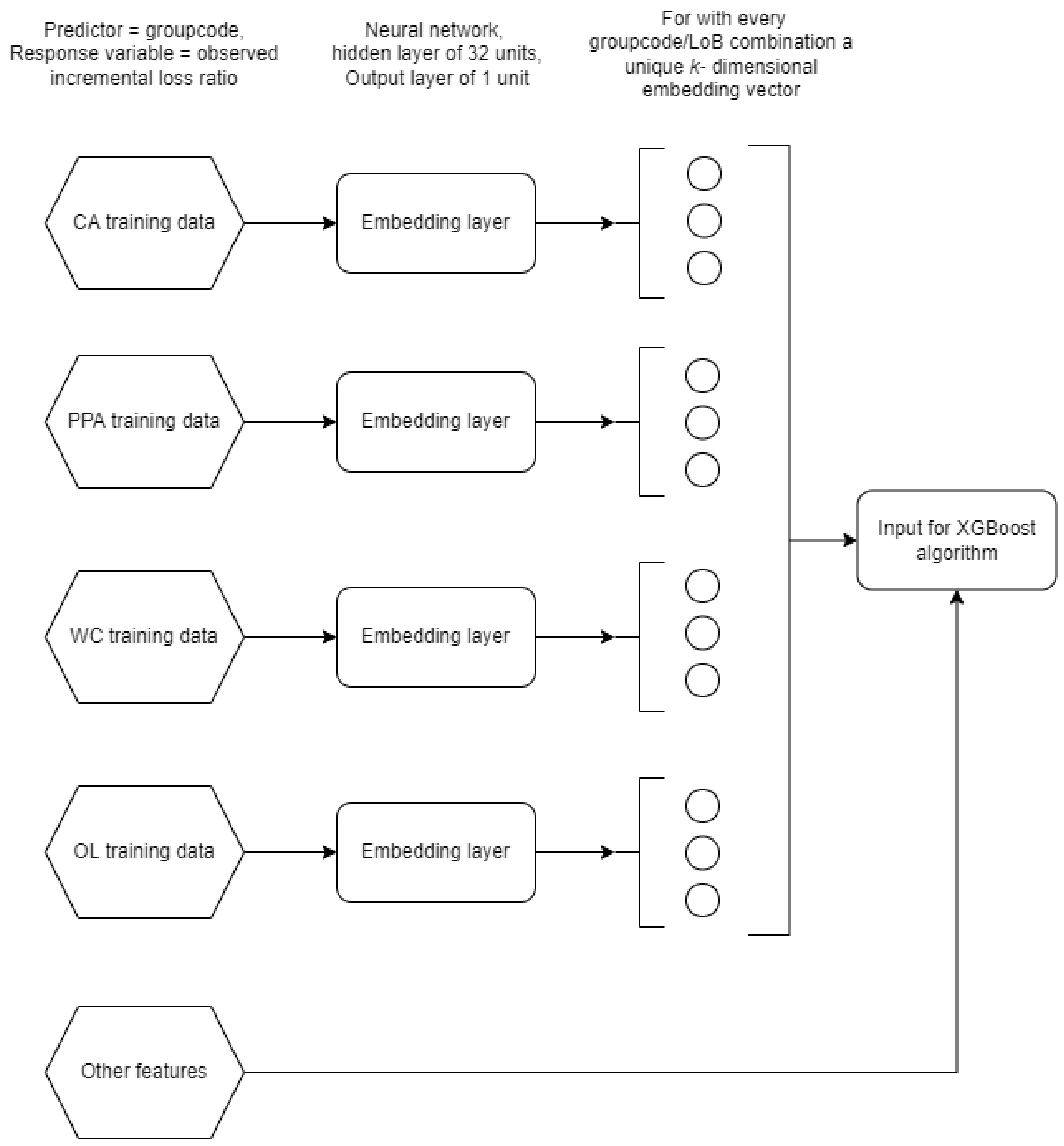


FIGURE 4.3: Overview of the company code embedding process

There are some design choices when making the embedding layer, namely, the loss function, the number of the hidden layers, number of hidden units, number of epochs, number of batches and output dimensionality ($\mathbb{R}^k$).

As loss function, since we want to predict numerical data, similar to a regression, we use the mean squared error (MSE) loss function. The mean squared error gives the average squared difference between the predicted loss and observed loss. The MSE associated with cell (i,j) is given in equation 4.1.

$$\mathrm{MSE}_{i,j} = \frac{1}{I-i+1-j} \sum_{k=j}^{I-i+1} (\hat{z}_{i,k} - z_{i,k})^2 \tag{4.1}$$

The loss function is computed as the average over the forecasted time steps of the mean squared error of the predictions.

As suggested by Kuo (2019), we use a single hidden layer with 32 hidden units with a ReLU activation function and a single unit output layer with a sigmoid activation function.

We use 90% of the training data to train the embedding layer, the other 10% are used for validation, to see if we're not overfitting. As shown in Figures 4.4, 4.5, 4.6 and 4.7, the training and validation loss function stabilises for each LoB after at most 35 epochs, therefore, we use 35 epochs as hyperparameter. Moreover, we see no signs of overfitting when using 35 epochs, since the training loss converges similarly to the validation loss. Furthermore, we use a batch size of 500 and the output is mapped to a fixed vector in ($\mathbb{R}^k$). In this study, we map to a vector in $\mathbb{R}^{10}$ since we want to capture the relations of the different companies without overfitting.

To summarize, in this process each company/LoB combination is mapped to a fixed length vector in $\mathbb{R}^{10}$. This is then used as input for our machine learning model.

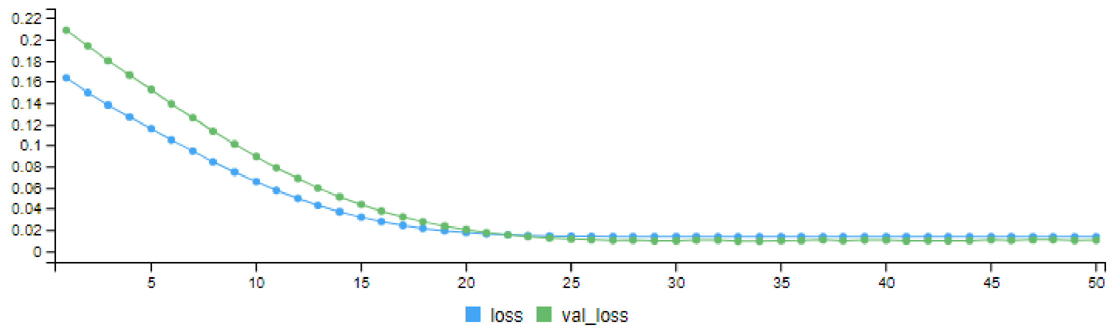


FIGURE 4.4: Training and validation MSE (loss) (y-axis) of the embedded layer compared to the number of epochs (x-axis) within the commercial auto LoB.

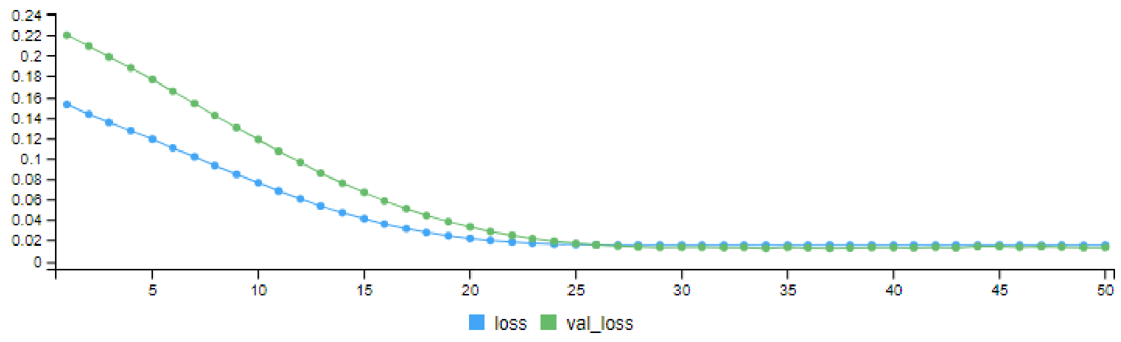


FIGURE 4.5: Training and validation MSE (loss) (y-axis) of the embedded layer compared to the number of epochs (x-axis) within the private passenger auto LoB.

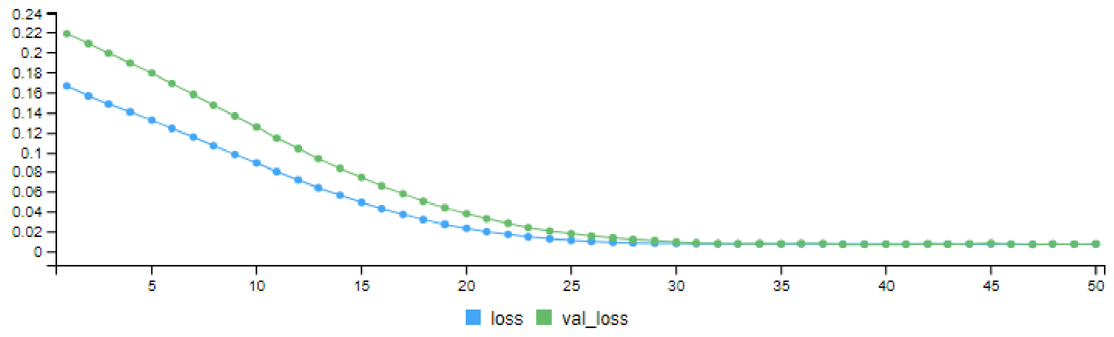


FIGURE 4.6: Training and validation MSE (loss) (y-axis) of the embedded layer compared to the number of epochs (x-axis) within the worker's compensation LoB.

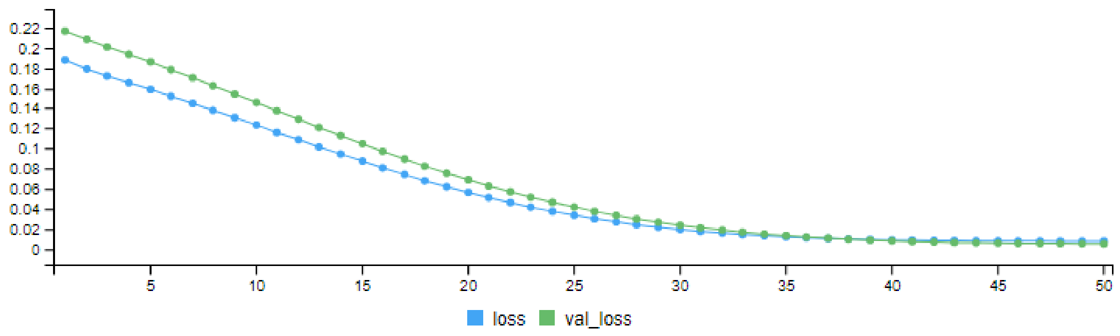


FIGURE 4.7: Training and validation MSE (loss) (y-axis) of the embedded layer compared to the number of epochs (x-axis) within the other liability LoB.

Hyperparameter tuning A hyperparameter is a particular kind of parameter that is specified before the learning process starts and is external to the model. It is predefined and has a direct impact on how well a model performs. XGBoost has the following hyperparameters that we try to tune:

- Learning rate: the learning rate defines the step size ath each iteration while the model optimizes toward its objective function. A model with a low learning rate slows down computation and needs more rounds to reduce residual error as much as a model with a high learning rate. However, it improves the likelihood of achieving the best optimal. The input space is [0,1]. with 0.3 as default.

- Loss reduction is a pseudo-regularisation parameter. It depends on the input of the other parameters. The higher loss reduction is, the higher the regularization. The input space is any integer and the default is 0.

- min n is the minimum number of data points in a node needed for subsequent node splitting. The input space can be any integer within [2,40].

- mtry: the number of randomly selected predictors, the input space can be any integer.

- Sample size: The siz of the data set utilized for modeling during an algorithm iteration. The input space is [0,1] and the default is 1.

- Tree depth: is the maximum depth per tree, it can be seen as the number of splits. A deeper tree may improve efficiency but also add complexity. The input space is any an integer greater than 0, the default is 6

Figure 4.8 shows the hyperparameter tuning approach. After splitting on the test and the training set, we fit the model on the training data. We use 5 fold cross validation for this as a compromise between having sufficient training examples per fold and having sufficient data to verify the performance and prevent overfitting. We use the latin hypercube sampling to create 500 hyperparameter sets, distributed nearly uniform over the input spaces of the hyperparameters. Therefore, we perform the cross validation process 500 times for each model. During each iteration, we test a different set of hyperparameters. We use the mean of the cross validated root mean squared error as a measure of performance of the model trained by the configuration of parameters. Due to runtime restrictions we use the Latin hypercube sampling to create the configuration sample. Since this approach is less computationally intense as a grid search and has a more complete coverage of the input space. Note that we repeat this process seperately for each line of business.

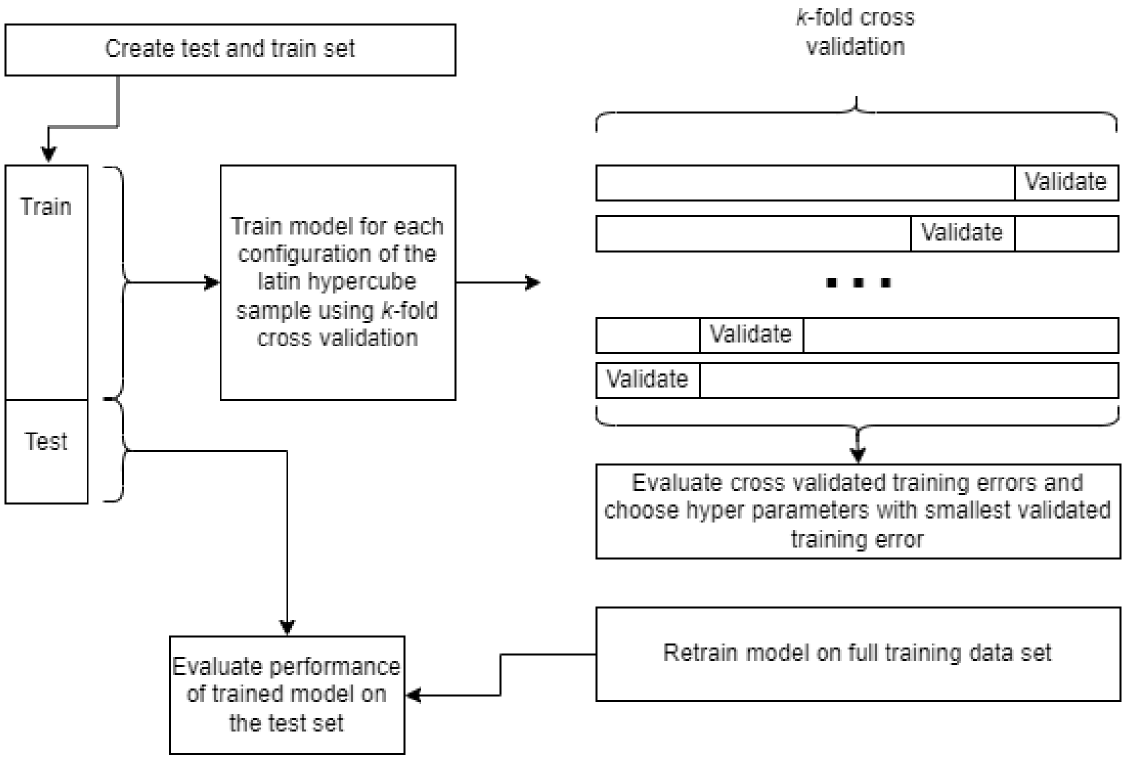

Create test and train set
Train
Test
Train model for each configuration of the latin hypercube sample using *k*-fold cross validation
k-fold cross validation
Validate
Validate
Validate
Validate
Evaluate cross validated training errors and choose hyper parameters with smallest validated training error
Retrain model on full training data set
Evaluate performance of trained model on the test set

FIGURE 4.8: An overview of the tuning process.

The in sample errors of the Latin grid sampling cross validation hypertuning approach of each line of business are shown in Figures 4.9, 4.12, 4.11, 4.12. Since each result of the individual parameter values depend on the values of the other parameters it is hard to draw conclusion based on these Figures. As can be seen the Latin hypercube samples creates hyperparamters sets that are almost uniformly distributed over the input space. Moreover, it seems that for each line of business in general a higher learn rate results in a lower error and a higher loss reduction results in a higher error. This means that the model needs relatively few rounds to reduce residual error.

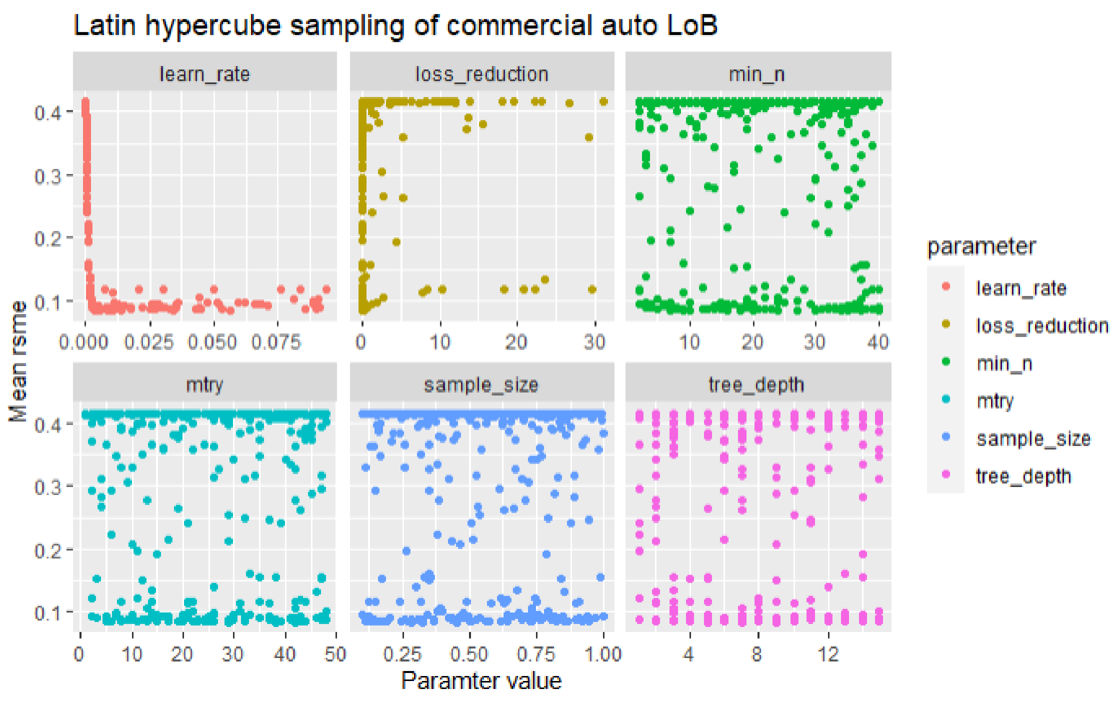


FIGURE 4.9: Results of hyperparameter tuning of the commercial auto LoB. The x-axis describes the hyperparameter values, the y-axis describes the in sample RSME..

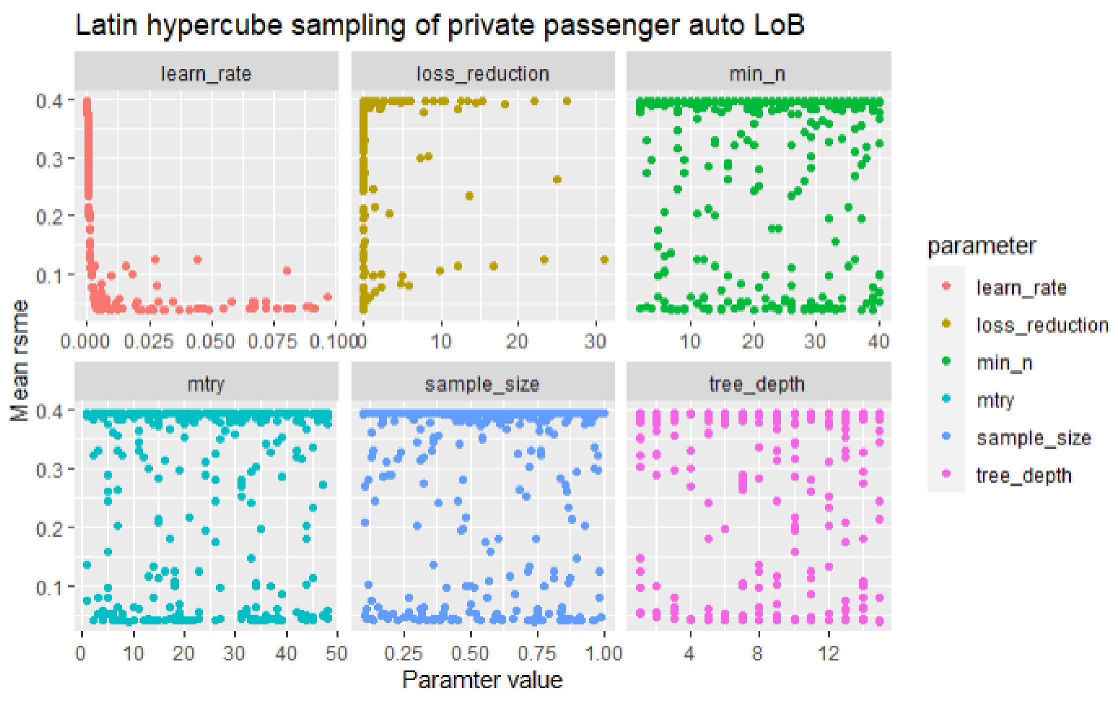


FIGURE 4.10: Results of hyperparameter tuning of the private passenger auto LoB. The x-axis describes the hyperparameter values, the y-axis describes the in sample RSME.

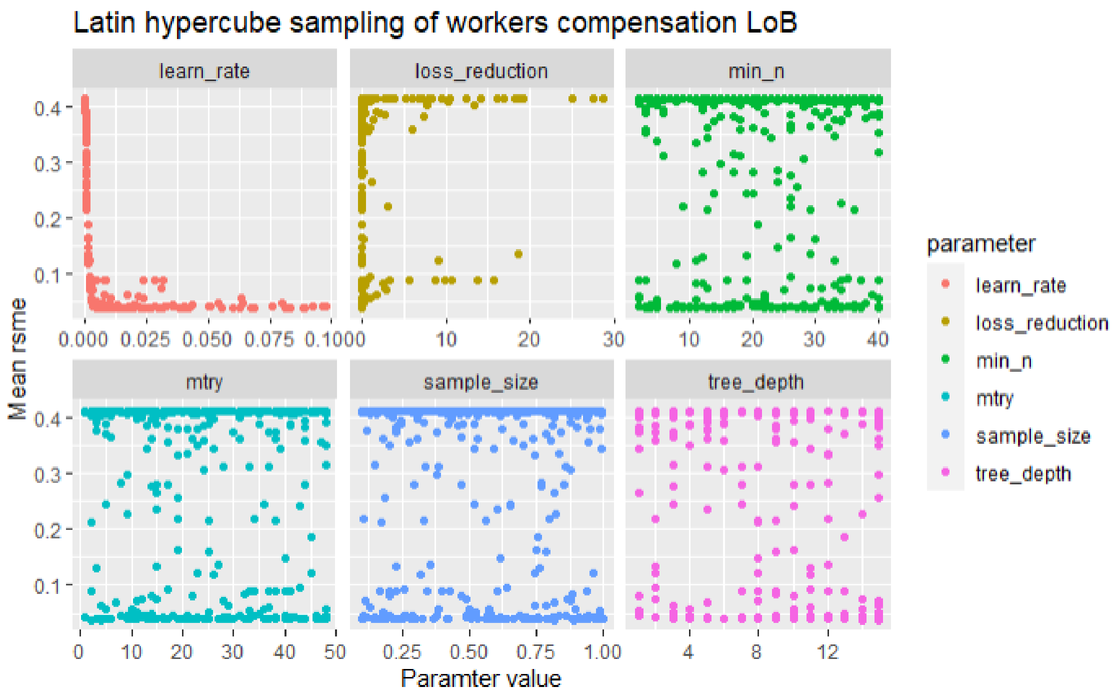


FIGURE 4.11: Results of the hyperparameter tuning of the workers' compensation LoB. The x-axis describes the hyperparameter values, the y-axis describes the in sample RSME.

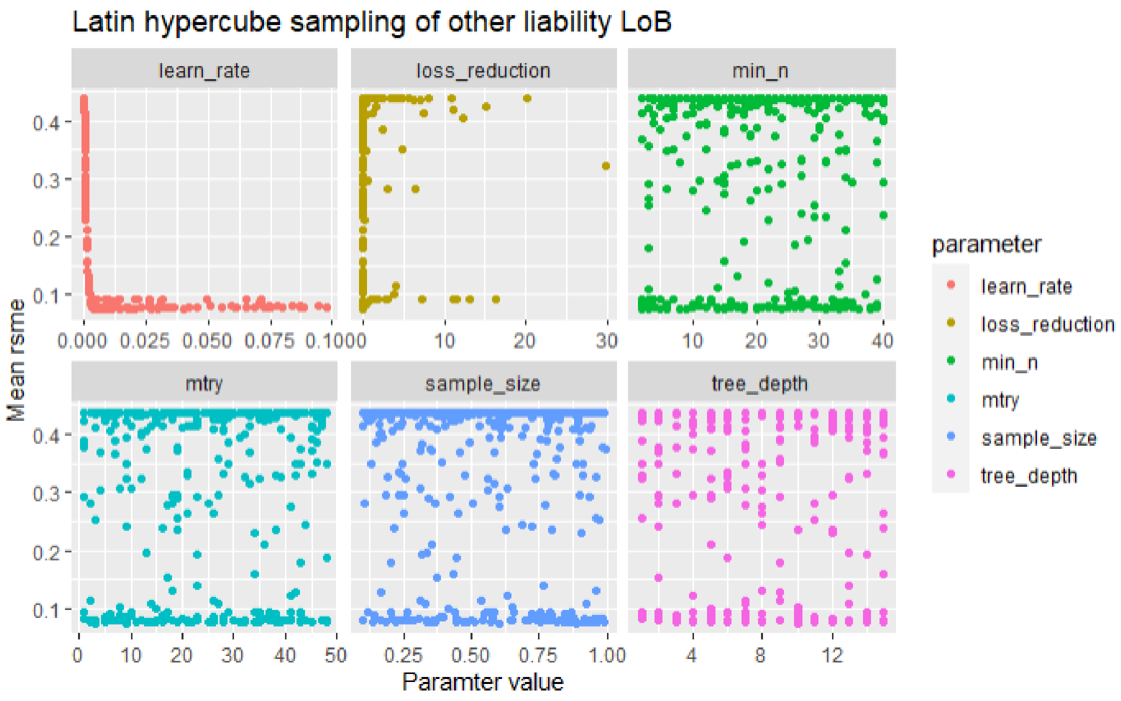


FIGURE 4.12: Results of hyperparameter tuning of the other liability LoB. The x-axis describes the hyperparameter values, the y-axis describes the in sample RSME.

Model architecture If we combine the embedding layer and Latin hypercube sample hyperparameter tuning with our benchmark model the result is the model architecture given in Figure 4.13. The figure gives a graphical overview of the proposed XGBoost model architecture for the commercial auto LoB. Note that the only difference between the different LoB model architectures are the input data. The number of features stay the same because we added the embedding layer. Each cell contains a vector in $\mathbb{R}^{10}$ where each element is employed as input feature for the XGBoost model.

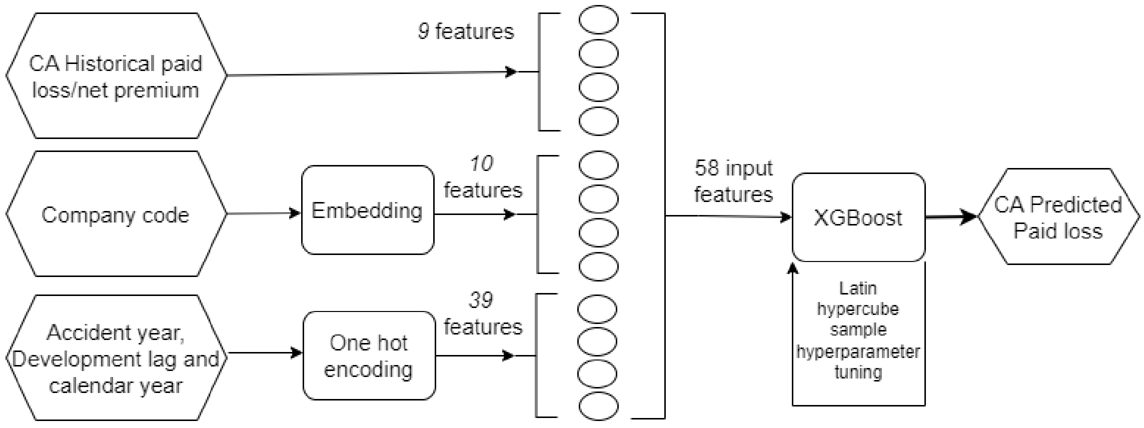

CA Historical paid loss/net premium
9 features
Company code
Embedding
10 features
Accident year, Development lag and calendar year
One hot encoding
39 features
58 input features
XGBoost
Latin hypercube sample hyperparameter tuning
CA Predicted Paid loss

FIGURE 4.13: XGBoost model architecture for commercial auto data set.

4.2 AutoML

AutoML, short for Automated Machine Learning, provides a method for the selection and optimization of machine learning algorithms. In our research, we utilize two distinct AutoML architectures, both of which are compared with the benchmark structures used for XGBoost.

4.2.1 Architectural Design

The first architecture uses the entire data set and is built upon the H2O AutoML algorithms. On the other hand, the second architecture is designed to work with H2O AutoML on separate data sets, each corresponding to a specific line of business. The integration of AutoML aims to validate the results of our research and to evaluate the effectiveness of XGBoost for the given problem.

4.2.2 Model Evaluation with AutoML

Based on the literature review, the AutoML function tests a variety of machine learning models. These models are evaluated using in-sample cross-validation. After this step, the function applies the stacked ensemble algorithm to combine the base learners.

4.2.3 Models in AutoML Function

The AutoML function includes several models: XGBoost, Gradient Boosting Machines (GBM), Random Forests, Deep Neural Networks, and Generalized Linear Models (GLM). Specifically, the H2O AutoML function starts by training and cross-validating a set of base learners, which consists of:

- Three pre-defined XGBoost GBM models.
- A grid of H2O GLMs.
- An H2O Extremely Randomized Trees (XRT) model.
- A standard H2O Random Forest (DRF).
- Five pre-configured H2O GBMs.
- A basic H2O Deep Neural Net.
- A random grid of XGBoost GBMs.
- A random grid of H2O Deep Neural Nets.

4.2.4 Selection Methodology

LeDell & Poirier (2020) describe the process behind the selection of these base models. They used a random search across different algorithmic families. The final step in their approach was the inclusion of a Generalized Linear Model (GLM) as the metalearner.

5 Results

In this chapter we discuss the following results, first, we elaborate on the loss reserve predictions and compare the performance of the different approaches; Second we go into (graphical) detail into the predictions of the loss ratios ($\hat{z}_{i,j}$) since this is the base for our loss reserve prediction $\hat{R}_c$; Third, we discuss the models ability to predict recoveries; At last, we will discuss the features that are important for predicting the loss reserve.

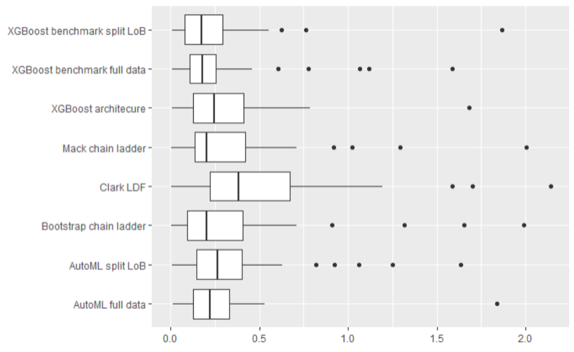
Contents

5.1 Predicting the loss reserve

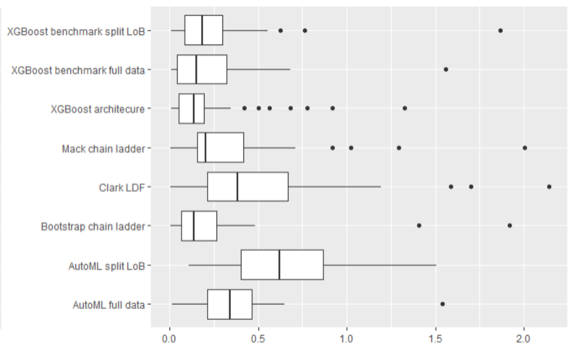
an in-depth evaluation of various predictive models tailored for estimating the loss reserve of non-life insurers across distinct lines of business is provided in Table 5.1. The two performance indicators, Mean Absolute Error (MAE) and Root Mean Squared Error (RSME), serve as the measures for this evaluation. These are as described earlier instrumental in gauging the accuracy and reliability of each model, with lower values being indicative of superior predictive capabilities.Table 5.1 shows the boxplot of the predicted loss reserve values. We first discuss the differences in line of businesses and thereafter discuss the predictive performance of each model and architecture.

Results								
	Line of business							
Models	Commercial auto		Private passenger auto		workers compensation		other liability	
	MAE	RSME	MAE	RSME	MAE	RSME	MAE	RSME
Mack chain ladder	0,33	0,49	0,22	0,37	0,29	0,39	0,43	0,65
Bootstrap chain ladder	0,42	0,67	0,26	0,57	0,30	0,40	0,46	0,67
Clark's LDF	0,52	0,69	0,39	0,51	0,32	0,43	0,58	0,81
XGBoost architecture	0,31	0,42	0,20	0,33	0,26	0,34	0,55	0,71
XGBoost full data	0,26	0,40	0,21	0,33	0,26	0,35	0,40	0,56
XGBoost split data	0,24	0,38	0,19	0,28	0,21	0,30	0,42	0,57
AutoML full data	0,26	0,37	0,36	0,43	0,24	0,32	0,44	0,64
AutoML split data	0,35	0,47	0,65	0,73	0,85	0,98	0,50	0,70

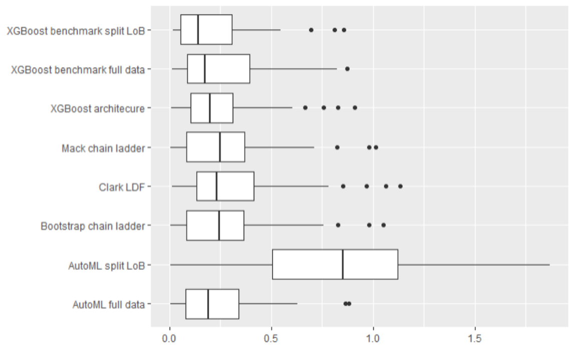
TABLE 5.1: The RSME and MAE results of all described models and architectures.



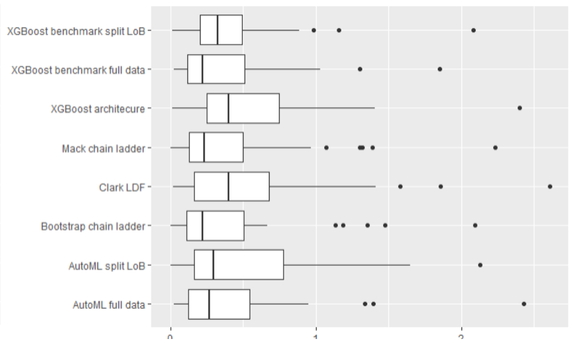
(A) commercial auto LoB.



(B) Private passenger auto LoB.



(C) Workers compesation LoB



(D) Other liability LoB

FIGURE 5.1: Boxplots of the residuals of the loss reserve for each line of business.

5.1.1 Line of Businesses

The LoBs under examination encompass Commercial Auto (CA), Private Passenger Auto (PPA), Workers Compensation (WC), and Other Liability (OL).

Delving deeper into the data description per LoB it is evident that there are variations in the number of data points, with PPA having the highest and OL having the lowest. This discrepancy in data volume might influence the predictive performance of models for each line. Furthermore, the skewness and kurtosis values indicate the distribution characteristics of the data. For instance, the high kurtosis value for CA and OL suggests a more peaked distribution with heavier tails, which could impact the model's ability to generalize well. The percentage of negative data points also varies across lines, with PPA having the highest percentage. Such negative values might represent specific challenges or anomalies in the data that models need to account for.

5.1.2 Model performances

Mack Chain Ladder: The Mack Chain Ladder is a traditional actuarial method that extends the basic chain ladder technique by providing prediction error estimates. For the LoB, it showed consistent performance, with Commercial Auto (CA) and Private Passenger Auto (PPA) having relatively lower RSME and MAE values. The method's reliance on deterministic assumptions might be a limitation when dealing with data that has high variability or skewness, as seen in some lines of business.

Bootstrap Chain Ladder: This method enhances the traditional chain ladder by incorporating a stochastic element through bootstrapping. While it provides a more probabilistic approach, its performance was slightly inferior to some of the more advanced models, especially for CA and PPA. One reason for its suboptimal performance could be that bootstrapping assumes data homogeneity. Given the presence of negative data points and the skewness in some lines of business, this assumption might be violated, affecting the model's accuracy.

Clark's LDF: Clark's Loss Development Factor model is a more sophisticated stochastic method that fits individual development factors for each development period. Despite its sophistication, it didn not outperform in all lines of business, especially in CA and PPA. The method relies heavily on the underlying distribution of the data. The high kurtosis and skewness values in some lines of business, like Other Liability (OL), might challenge the model's assumptions, leading to higher RSME and MAE values.

XGBoost Architecture: XGBoost, a gradient boosting algorithm, showed impressive performance across all lines of business, especially when trained on split data. Its ability to handle non-linear relationships and its robustness to outliers might explain its superior performance. The model's adaptability to different data distributions, as seen in the lines of business, underscores its versatility.

XGBoost Full Data: When trained on the full dataset, XGBoost continued to exhibit strong performance, especially in CA and PPA. This suggests that the model benefits from larger datasets, and its performance might further improve with even more data.

XGBoost Split Data: The split data variant of XGBoost, which presumably trains on a subset of the data and validates on another, still managed to outperform many traditional models. This highlights the model's efficiency even with limited data.

AutoML Full Data: Automated Machine Learning (AutoML) aims to automate the end-to-end process of machine learning. For the full data variant, its performance was commendable, especially in CA and PPA. The model's ability to automatically select the best algorithm and hyperparameters might be a contributing factor to its success.

AutoML Split Data: The performance dip observed in the split data variant of AutoML suggests that the model might not have had enough data to train on effectively. This underscores the importance of data volume in achieving optimal model performance.

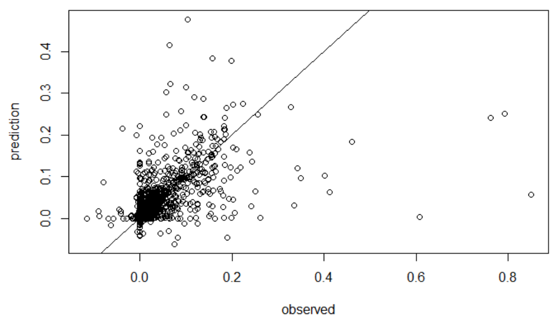
In summation: While traditional actuarial methods like the Mack Chain Ladder and Bootstrap Chain Ladder provide foundational approaches to loss reserve estimation, their deterministic and homogeneity assumptions might be challenged by data with high variability, skewness, or negative values. On the other hand, advanced models like XGBoost and AutoML leverage machine learning techniques to adapt to various data characteristics, often resulting in superior predictive performance. However, the choice of model should always consider the nuances of the data and the specific requirements of the task at hand.

5.2 Loss ratios

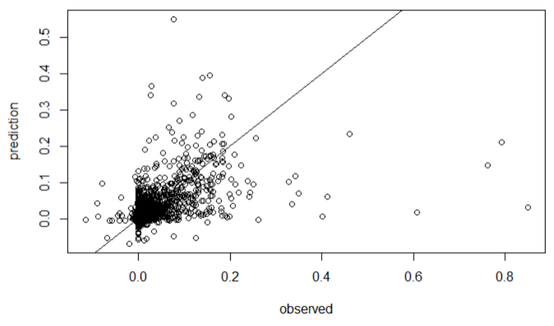
To gain a deeper understanding of the predictive performance across different lines of business, we turn to scatter plots. By plotting observed values against their corresponding predictions, scatter plots offer a clear snapshot of a model's accuracy and precision. The closer the data points cluster around the line y=x, the more accurate the model's predictions. As we delve into each LoB, these plots will illuminate the nuances of each model's performance. Note that we only analyse the best performing traditional model, namely chain ladder, To keep it concise and clear.

5.2.1 Commercial auto

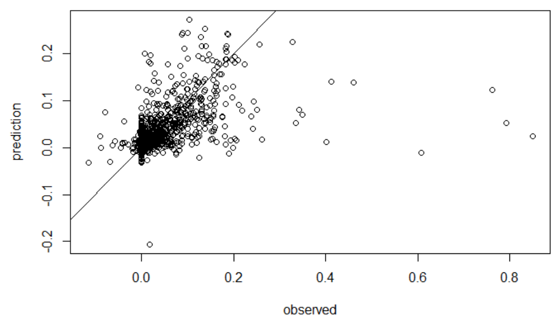
The predicted and observed loss of each observation in the test set of the CA LoB is given in Graph 5.2. We discuss the results below.



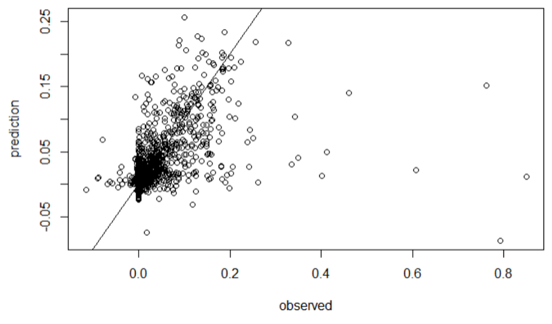
(A) Mack chain ladder.



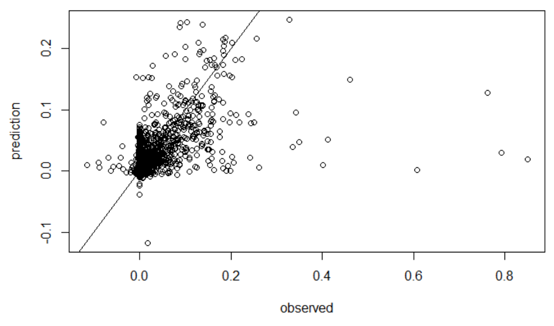
(B) XGBoost architecture.



(C) XGBoost model trained on full data set.



(D) XGBoost model trained on split data set.



(E) AutoML model trained on full data set.

FIGURE 5.2: Predicted and observed loss ratios of the different models for the CA LoB.

Both the Mack Chain Ladder and the XGBoost architecture exhibit similar patterns in their scatter plots. Many of their data points cluster around the y=x line, suggesting a commendable degree of accuracy. However, deviations, especially for higher loss ratios, hint at potential challenges in predicting extreme values or outliers. This indicates that while both models are generally reliable, they might occasionally falter with specific data points.

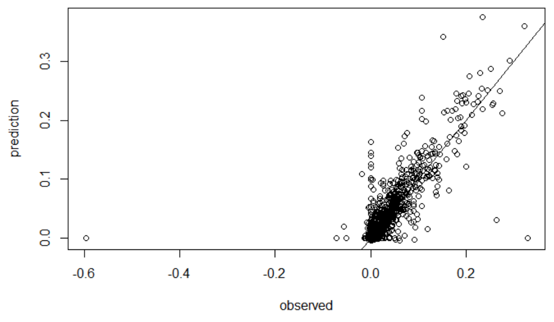
The scatter plots for XGBoost, whether trained on split or full data, showcase a consistent pattern. The split data variant displays a slightly tighter clustering around the ideal line, suggesting enhanced accuracy when trained on a subset. On the other hand, the full data variant, while still accurate, seems to be influenced by the broader variability inherent in a larger dataset.

The AutoML model, trained on full data, stands out distinctly. Its data points are densely packed around the y=x line, signaling superior predictive accuracy. This model's automated approach to algorithm selection and hyperparameter tuning appears to be especially effective for the Commercial Auto line of business.

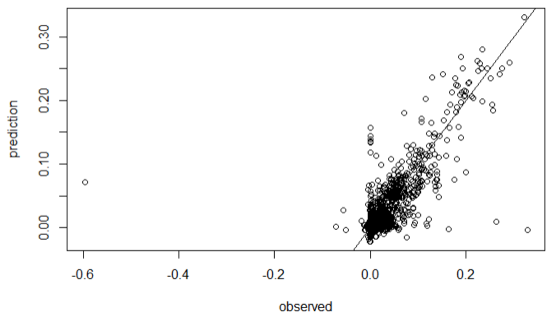
In summation, consistent with the results of the loss reserve predictions, it is evident that not all models yield equal results. The models that excel in this analysis, particularly the AutoML (Full Data) and XGBoost (Split Data), are also the ones that deliver the most accurate predictions for the Commercial Auto segment.

5.2.2 Private passenger auto

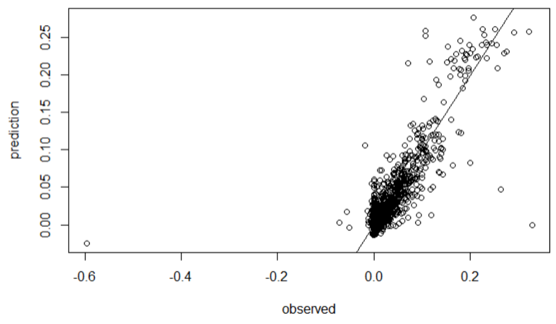
The predicted and observed loss of each observation in the test set of the PPA line of business is given in Graph 5.3. For the PPA LoB, the data points across all models are more densely clustered around the y=x line compared to the CA LoB. This suggests that the models, in general, have a higher predictive accuracy for the PPA segment. We discuss the remaining results below.



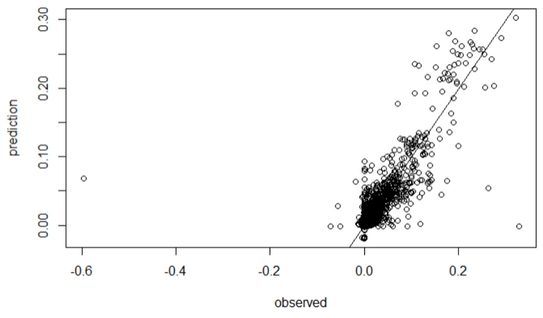
(A) Mack chain ladder.



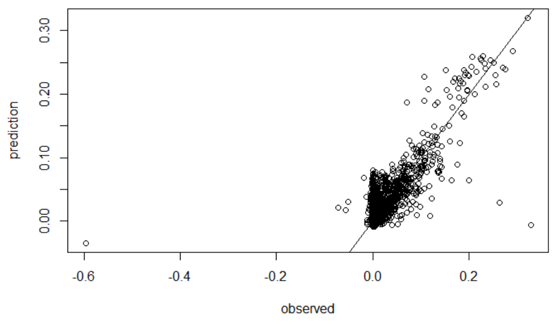
(B) XGBoost architecture.



(C) XGBoost model trained on full data set.



(D) XGBoost model trained on split data set.



(E) AutoML model trained on full data set.

FIGURE 5.3: Predicted and observed loss ratios of the different models for the private passenger auto LoB.

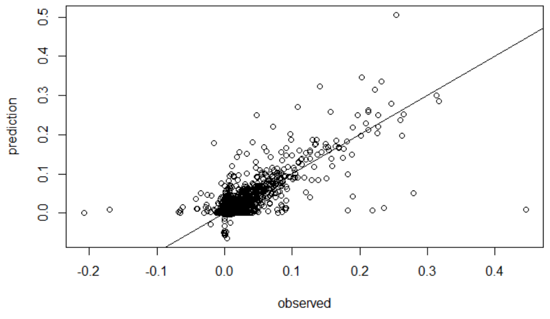
Both variants of the XGBoost model, whether trained on split or full data, emerge as top performers for the PPA line of business. The split data variant, in particular, displays an impressive clustering around the ideal line, with notably fewer outliers. This tight clustering, combined with its minimal outliers, underscores why the XGBoost model based on split data is the best performer among the lot.

While the XGBoost models shine, it is worth noting that the scatter plots for other methodologies also exhibit a decent accuracy, with their results appearing fairly consistent. The data points for these models are also closely packed around the y=x line, indicating that the differences in their predictive capabilities, especially for the PPA segment, are relatively marginal.

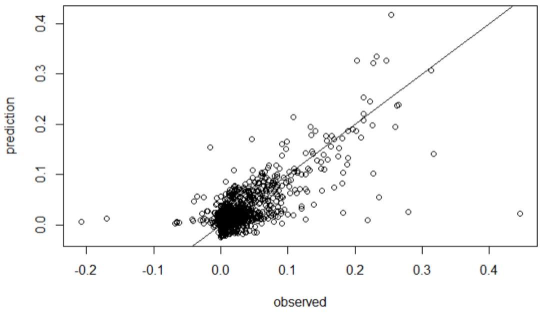
In summation, the PPA line of business scatter plots reveal a higher degree of predictive accuracy across all models compared to the CA segment. However, the XGBoost model trained on split data stands out, primarily due to its fewer outliers, making it the most reliable predictor for the PPA line of business.

5.2.3 Workers' compensation

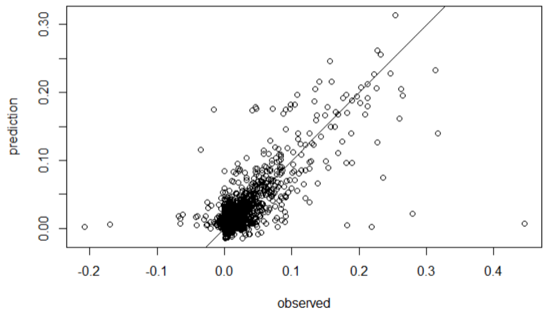
The predicted and observed loss of each observation in the test set of the WC line of business is given in graph 5.4. We discuss the results below.



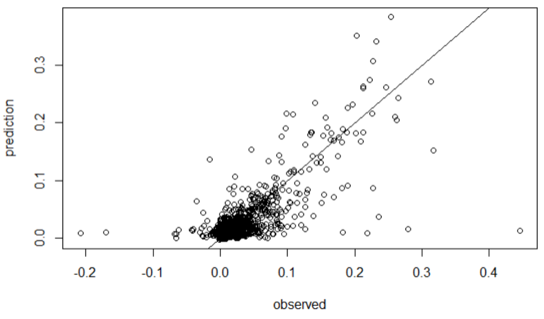
(A) Mack chain ladder.



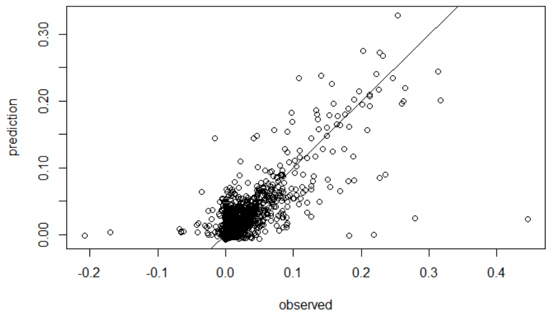
(B) XGBoost architecture.



(C) XGBoost model trained on full data set.



(D) XGBoost model trained on split data set.



(E) AutoML model trained on full data set.

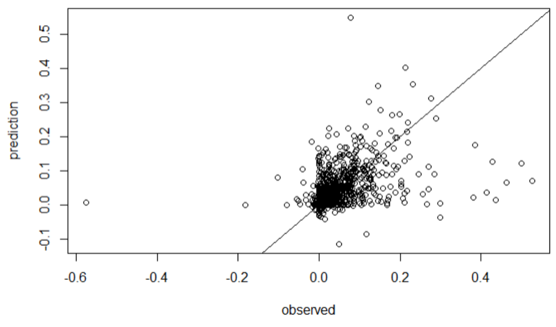
FIGURE 5.4: Predicted and observed loss ratios of the different models for the workers' compensation LoB.

For the Workers Compensation line of business, the scatter plots offer a visual comparison of the models' predictive capabilities. The XGBoost model trained on split data stands out, with most of its data points closely hugging the y=x line. The Mack Chain Ladder and XGBoost architecture display commendable results, but with slightly more dispersion. The AutoML model, trained on full data, showcases a distribution akin to the XGBoost models but with a few more deviations.

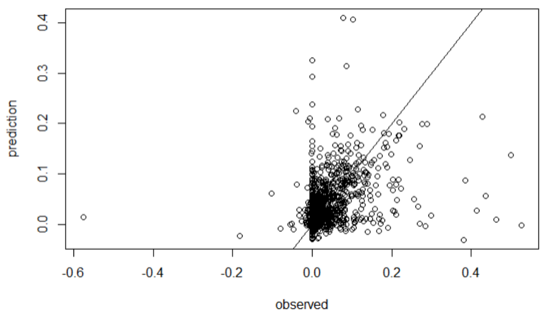
While all models present comparable performances for the Workers Compensation LoB, the XGBoost model trained on split data distinguishes itself. Most of its points are more concentrated around the y=x line, and its outliers are notably closer to this benchmark, underscoring its subtle superiority for this line of business.

5.2.4 Other liability

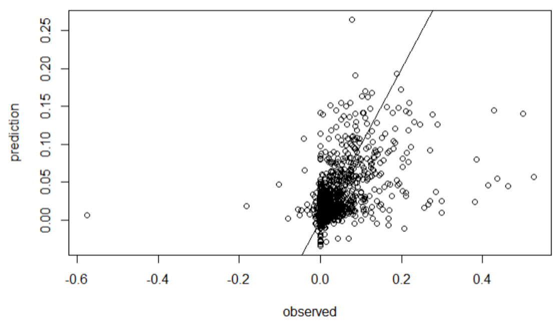
The predicted and observed loss of each observation in the test set of the OL line of business is given in Graph 5.5. For the Other Liability line of business, the scatter plots exhibit a more dispersed distribution compared to the previous lines of business. This increased spread might be attributed to the limited data available for this category and the fact that it encompasses a diverse range of LOBs, introducing more variability. We discuss the remaining results below.



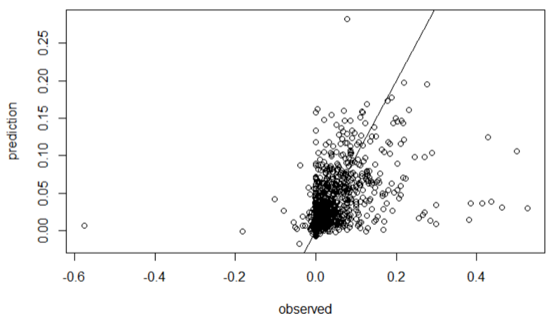
(A) Mack chain ladder.



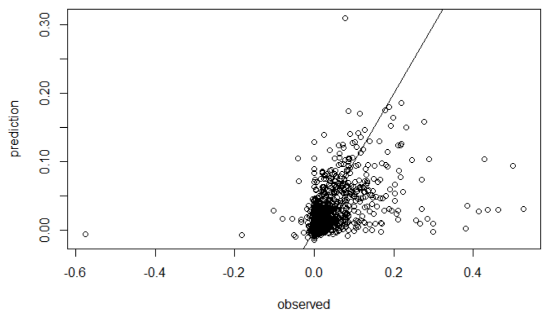
(B) XGBoost architecture.



(C) XGBoost model trained on full data set.



(D) XGBoost model trained on split data set.



(E) AutoML model trained on full data set.

FIGURE 5.5: Predicted and observed loss ratios of the different models for the other liability LoB.

The XGBoost models, particularly the one trained on full data, demonstrate slightly superior performance, with their data points being relatively closer to the y=x line. On the other hand, the Mack Chain Ladder and AutoML models display similar patterns, with their predictions spread out in a manner that suggests comparable predictive capabilities.

In summation, while the Other Liability LoB presents challenges due to its diverse nature and limited data, the XGBoost model trained on full data emerges as the most reliable. The Mack Chain Ladder and AutoML models, despite their merits, offer similar performances that are slightly overshadowed by the precision of the XGBoost models

5.3 Recoveries

We provide an overview of the performance of predicting recoveries of various models, as measured by RSME and MAE, across different lines of business in Table 5.2. A striking observation is the superior performance of the base XGBoost model, which consistently outperforms its more intricate counterparts, including the XGBoost architecture and the AutoML model. This suggests that the base XGBoost model, despite its simplicity relative to the other advanced models, captures the essential patterns in the data most effectively.

Furthermore, the traditional Mack chain ladder method, a stalwart in actuarial predictions, not only holds its ground but also surpasses the more complex models like the XGBoost architecture and AutoML. This underlines the robustness and enduring relevance of the chain ladder method in this context.

In terms of specific results, the XGBoost model with split data showcases some of the lowest RSME and MAE values, particularly in the other liability and workers compensation LoBs. This indicates its adeptness at making precise predictions when the data is segmented appropriately. On the other hand, while the XGBoost architecture and AutoML models bring sophisticated methodologies to the table, their performance does not necessarily translate to better predictive accuracy in this dataset.

In conclusion, the results underscore the efficacy of the base XGBoost model in predicting recoveries, even when juxtaposed against more complex models. Additionally, the enduring performance of the Mack chain ladder method serves as a testament to its robustness, further emphasizing that complexity does not always equate to superior predictive power.

Recovery results								
	Line of business							
Models	**Commercial auto**		**Private passenger auto**		**workers compensation**		**other liability**	
	MAE	RSME	MAE	RSME	MAE	RSME	MAE	RSME
Mack chain ladder	0,025	0,047	0,021	0,089	0,034	0,094	0,042	0,130
XGBoost architecture	0,065	0,121	0,033	0,083	0,051	0,107	0,060	0,135
XGBoost full data	0,043	0,084	0,039	0,098	0,036	0,080	0,070	0,176
XGBoost split data	0,017	0,049	0,011	0,052	0,013	0,042	0,005	0,018
AutoML full data	0,038	0,050	0,038	0,089	0,028	0,064	0,054	0,123

TABLE 5.2: The RSME and MAE results of the recoveries.

Table 5.3 presents the comparative accuracy of various models in predicting recoveries within a claim triangle. The objective was to ascertain the effectiveness of each model in forecasting whether a single recovery would occur within the claim triangle, juxtaposed against the actual outcome.

Our findings indicate a marked difference in the performance of traditional actuarial methods versus machine learning models. The Mack chain ladder, a conventional actuarial technique, yielded accuracy rates ranging from 43% to 51% across the different insurance categories. This performance, while consistent with expectations based on the deterministic nature of the model, was notably outperformed by machine learning approaches.

The XGBoost architecture, a gradient boosting framework, demonstrated superior accuracy, overall the XGBoost model with split data outperformed all other models. The XGBoost models consistently outperformed the Mack chain ladder. Similarly, the AutoML model, which leverages automated machine learning processes on full data, mirrored the high accuracy rates of the XGBoost models.

The observed disparity in performance can be attributed to the inherent characteristics of the models. Traditional models like the chain ladder are grounded in historical data patterns and often operate under deterministic assumptions. While they provide a foundational understanding of claim triangles, they may not capture intricate nuances and non-linear relationships inherent in the data. In contrast, machine learning models, with their capacity to discern and learn from complex patterns in extensive datasets, offer a more adaptive and potentially accurate predictive framework.

In summary, our results underscore the potential of machine learning models, particularly the XGBoost and AutoML frameworks, in enhancing the accuracy of recovery predictions within claim triangles. The findings suggest a promising direction for future research and potential integration of these models into insurance and actuarial practices.

Accuracy of recovery prediction					
Models	**Commercial auto**	**Private passenger auto**	**Workers compensation**	**Other liability**	**Total**
Mack chain ladder	49%	51%	43%	47%	48%
XGBoost architecture	83%	84%	61%	67%	74%
XGBoost full data	42%	51%	52%	39%	46%
XGBoost split data	81%	76%	74%	81%	78%
AutoML full data	83%	82%	63%	67%	74%

TABLE 5.3: Accuracy of predicting recoveries within the claim triangles.

5.4 Feature importance

We show the feature importance scores for the best-performing model, the XGBoost, when trained on split data across four LoBs: Commercial Auto, Private Passenger Auto, Workers Compensation, and Other Liability in Figure 5.6. We discuss the features below:

1. **Development Lag**: This feature stands out as the most influential across all categories, with the highest importance in Commercial Auto, Private Passenger Auto, and Workers Compensation. Its slightly reduced importance in the Other Liability

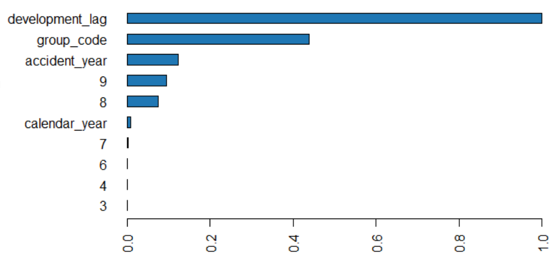
category suggests that the time taken for a claim to develop might be slightly less predictive in this category compared to the others.

2. **Company Code**: The importance of the company code varies across categories. It's most influential in the Other Liability category, indicating that specific companies might have distinct patterns or behaviors when it comes to other liability claims. In contrast, its influence is notably less in Private Passenger Auto. It suggest that further splitting lines of businesses might result in more information gain.

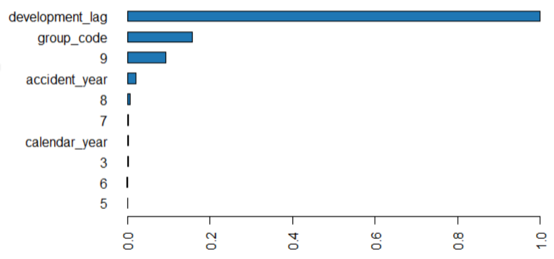
3. **Accident Year & Calendar Year**: Both these factorized data points show varying degrees of importance across the categories. The accident year has a moderate influence in Commercial Auto but is less significant in the other categories. The calendar year, representing the year in which the claim was made, has relatively low importance across all categories, suggesting that the specific year of claim might not be a strong predictor in this context.

4. **Time Series Data (9 to 3)**: The observations from the most recent periods (9 and 8) generally have higher importance scores across all categories, emphasizing their relevance in predicting recoveries. The importance of 9 is especially pronounced in the Other Liability category. As we move further back in time (7 to 4), the importance diminishes, which is consistent with the nature of claim triangles where more recent data tends to be more predictive. The absence of significant importance for 3 in most categories, and the lack of 2 and 1, aligns with the understanding that not every accident year has extensive data in the context of claim triangles.

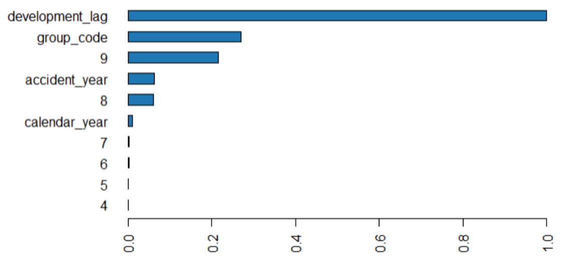
In summation, the results underscore the significance of recent time series data and development lag in predicting recoveries within claim triangles. While development lag overlaps with the assumption of the chain ladder, it is interesting to see that including the loss ratios of the 'known' triangle increases the predictability of the loss reserve. Furthermore, the significance of the company code information and the Line of Business (LoB) splitting in our results aligns with findings from existing literature. It has been documented that incorporating company-specific information and segmenting by Line of Business can enhance the predictability of the loss reserve. This segmentation allows for a more granular understanding of claim patterns and behaviors, which can be particularly beneficial when forecasting future liabilities. The consistency of our results with established literature not only validates our model's findings but also emphasizes the importance of considering these factors in loss reserve prediction models.



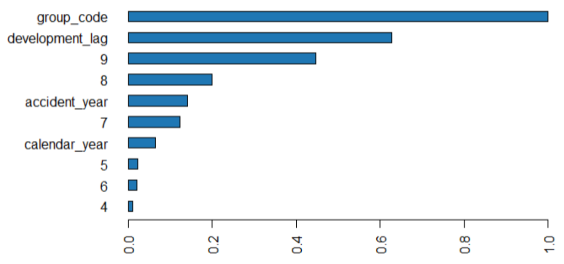
(A) commercial auto LoB.



(B) Private passenger auto LoB.



(C) Workers compesation LoB



(D) Other liability LoB

FIGURE 5.6: Feature importance of the XGBoost trained on the split datasets.

6 Conclusions

In concluding this research, it is important to revisit the initial questions that guided this study. The complex field of predicting the loss reserve of non life insurers, with its various challenges, led to the following inquiries:

How effective are machine learning techniques in enhancing the accuracy of loss reserve predictions for non-life insurers?

With the following sub-questions:

1. *Which metrics can be employed to assess the performance of loss reserve predictions in non-life insurance?*
2. *Which conventional methods have been historically utilized for loss reserve forecasting in non-life insurance*
3. *Which machine learning algorithms have been previously explored for enhancing loss reserve predictions in the non-life insurance sector?*
4. *Can we improve the performance of predicting recoveries using machine learning?*
5. *Which features are important for influencing the performance of loss reserve predictions?*
6. *Which specific machine learning approaches can be recommended to optimize the forecasting of loss reserves in non-life insurance?*
7. *Which validation techniques can ensure the reliability and robustness of our loss reserve prediction models?*

Each question led to a specific line of investigation, directing the research through traditional methods, advanced machine learning techniques, and the detailed area of recoveries. In this conclusion, we intend to integrate the insights and results obtained from these questions. A comprehensive summary will be provided, highlighting the research's main contributions, challenges faced, and the wider application for the non-life insurance field. We will first discuss the conclusions from the sub-questions and then conclude with the primary research question.

To assess the accuracy of loss reserve predictions in non-life insurance, we primarily utilize the RSME (Root Mean Square Error) and MAE (Mean Absolute Error) metrics. Both RSME and MAE serve as reliable indicators of the magnitude of errors between predicted and observed values, providing insights into the model's performance. For the specific task of predicting recoveries, RSME and MAE continue to play a pivotal role in evaluating the performance. These metrics offer a quantitative measure of how closely the model's predictions align with the actual observed recoveries. In addition to RSME and MAE, the accuracy metric is employed to gauge the model's capability to predict the occurrence of recoveries within a triangle. This metric provides a binary perspective, determining whether or not a recovery will take place, thereby complementing the quantitative insights offered by RSME and MAE. In summary, while RSME and MAE provide a comprehensive understanding of the model's performance in terms of quantitative prediction errors, the accuracy metric adds an additional layer of assessment, specifically tailored for predicting the occurrence of recoveries within a triangle.

Historically, several conventional methods have been employed for loss reserve forecasting in non-life insurance, with a few standing out due to their prominence in literature and practical applications. Among these, the Mack Chain Ladder, Bootstrap Chain Ladder, and Clark's LDF (Loss Development Factor) are particularly noteworthy. The Mack Chain Ladder is a widely recognized method, known for its robustness and adaptability. In the context of this research, it has been observed that the Mack Chain Ladder outperforms both the Bootstrap Chain Ladder and Clark's LDF, making it the most effective model among the three for predicting loss reserves. Furthermore, it's essential to acknowledge the role of expert judgment in the realm of loss reserve forecasting. In practice, experts often opt to exclude certain link ratios based on their domain knowledge and experience. This exclusion, while not incorporated in the models discussed in this research, is a common practice that can potentially enhance the performance of loss reserve predictions. By integrating such expert judgment with the computational prowess of models like the Mack Chain Ladder, there's potential for more accurate and reliable forecasts. In conclusion, while the Mack Chain Ladder, Bootstrap Chain Ladder, and Clark's LDF serve as foundational pillars in loss reserve forecasting, the integration of expert judgment offers an avenue for further refinement and optimization in the prediction process. however this is not included in this thesis.

The body of literature on loss reserve predictions in non-life insurance has witnessed a transition from traditional stalwarts to the incorporation of advanced machine learning techniques. While foundational methods like the Mack Chain Ladder, Bootstrap Chain Ladder, and Clark's LDF have been mainstays, the advent of machine learning has ushered in a new era of exploration and potential enhancement. Neural networks, as highlighted by Wüthrich (2018), have been employed to synthesize claims data, and there's been an extension of the conventional chain ladder method with neural networks to better incorporate claims features. Deep learning, especially, has been gaining traction, with Kuo (2019) proposing a loss reserving approach based on deep neural networks, which allows for a nuanced integration of diverse inputs into the modeling of outstanding paid losses and claims. Lopes et al. (2016) introduced an innovative regression tree method, aiming to compute the conditional distribution of a variable that's typically censored from direct observation. Additionally, there's been a notable effort, as seen in the works of Gabrielli et al. (2018) and Gabrielli (2019), to embed classical parametric loss reserving models into neural networks. Within the scope of this research, there's a distinct emphasis on the XGBoost algorithm. Recognized for its prowess across various domains, XGBoost's application in the realm of loss reserve predictions for non-life insurers is a novel endeavor. This research endeavors to harness the potential of XGBoost, especially given its uncharted status in this specific context and its renowned performance capabilities. In essence, the literature and this research collectively signal a promising trajectory for machine learning in enhancing loss reserve predictions, with ample avenues for future exploration. Moreover, in this thesis we apply some methods used in literuture such as company code embedding to try to improve the performance of the regular XGBoost model.

Machine learning, specifically the XGBoost model, has demonstrated a notable improvement in the performance of predicting recoveries when compared to traditional methods. By evaluating the RSME (Root Mean Square Error) and MAE (Mean Absolute Error) of the recovery loss ratios, it becomes evident that the XGBoost model offers a more accurate prediction. Additionally, when assessing the accuracy of predicting the occurrence of recoveries within a claim triangle, the XGBoost model again stands out, showcasing its superior predictive capabilities. In essence, while traditional methods have their merits, the application of machine learning, and in particular the XGBoost algorithm, brings forth a significant enhancement in the accuracy and reliability of predicting recoveries in the context of non-life insurance.

Traditional methods, such as the chain ladder, primarily focus on development lag as the key feature for loss reserve predictions. This approach, rooted in historical practices, offers a generalized perspective on loss reserving. However, our research findings suggest a more nuanced approach can yield enhanced results. Specifically, considering historical claims, especially the most recent ones, has shown to be pivotal in refining predictions. The inclusion of the company code as a feature provides an additional layer of granularity, capturing company-specific nuances that can influence loss reserve predictions. Furthermore, segmenting the data based on Lines of Business (LoB) has proven beneficial. Different LoBs have distinct characteristics and risk profiles, and by tailoring the models to cater to these individualities, the performance of our predictions has shown marked improvement. In essence, while development lag remains a cornerstone in traditional loss reserve forecasting, our results underscore the value of incorporating a broader set of features, such as recent claim history, company code, and LoB-specific data, to improve the performance of our models.

To ensure the reliability and robustness of our loss reserve prediction models, we employ a validation technique analogous to the one used with the chain ladder and other traditional methods. Specifically, we implement a train-test split based on the calendar year. The training data, derived from the earlier calendar years, is utilized to train the models. Subsequently, the models are tested using data from the later calendar years, serving as the test set. This approach not only mirrors the validation process of traditional methods like the chain ladder but also ensures that the models are exposed to a diverse range of data, enhancing their generalization capabilities. Importantly, this train-test split methodology based on calendar years is versatile and can be consistently applied across all models, ensuring a standardized validation process that bolsters the confidence in the predictive outcomes.

The utilization of machine learning, and more specifically the XGBoost algorithm, has demonstrated an enhancement in the performance of predicting recoveries. When the data is segmented based on Lines of Business (LoB), the XGBoost model consistently excels, outperforming other methods across all evaluation metrics. This pronounced superiority is evident in metrics such as RSME, MAE (and accuracy) for both normal loss reserve predictions and recovery predictions. The segmentation based on LoB allows the model to cater to the unique characteristics and risk profiles of each business line, leading to more tailored and accurate predictions. In summary, the application of XGBoost, especially when data is split based on Lines of Business, stands out as an effective approach in the domain of loss reserve and recovery predictions, offering an advancement over both traditional methods and other machine learning techniques.

The central research question of this thesis sought to understand the efficacy of machine learning techniques in enhancing the accuracy of loss reserve predictions for non-life insurers. As the insurance landscape becomes increasingly complex, the need for more accurate and reliable prediction models has never been more paramount. Traditional methods, while foundational, often fall short in capturing the intricate nuances and evolving dynamics of the non-life insurance sector. Machine learning, with its ability to process vast amounts of data and identify intricate patterns, has emerged as a promising alternative. The research has shown that machine learning models, particularly the XGBoost algorithm, have demonstrated significant improvements in predictive accuracy over traditional methods. This is especially evident when data is segmented based on Lines of Business (LoB), allowing the model to cater to the unique characteristics and risk profiles of each business line. Furthermore, the research underscores the importance of considering a broader set of features for loss reserve predictions. While traditional methods like the chain ladder primarily focus on development lag, the inclusion of features such as recent claim history, company code, and LoB-specific data has proven to enhance the performance of prediction models. The validation techniques employed in this research, particularly the train-test split based on calendar years, ensure the reliability and robustness of the prediction models. This approach not only mirrors the validation process of traditional methods but also bolsters confidence in the predictive outcomes. In conclusion, machine learning techniques, especially when tailored to the specific nuances of the non-life insurance sector, offer an advancement in the perfomance of loss reserve predictions. The findings of this research not only validate the potential of machine learning in this domain but also pave the way for further exploration and optimization in the future.

7 Discussion and limitations

Our research delved into the domain of non-life insurance, emphasizing the role of machine learning in improving loss reserve predictions. The results highlight the significant impact of machine learning, especially the XGBoost algorithm, in tackling the complex issues of the non-life insurance industry. Segmenting data by Lines of Business (LoB) and incorporating a wider range of features proved crucial in enhancing the models' predictive performance.

While traditional methods have served as the bedrock of loss reserve predictions for years, the dynamic and evolving landscape of non-life insurance necessitates a more nuanced and adaptive approach. Machine learning, with its ability to process vast datasets and discern intricate patterns, offers a promising avenue for future research and applications in this domain.

However, as with any research endeavor, it is crucial to acknowledge the inherent limitations and potential areas of improvement. One of the primary limitations of this research is the practical challenge associated with using multiple claim triangles across various Lines of Business (LoB). In a real-world scenario, insurers would need to share data to achieve this level of granularity and segmentation. Given the competitive nature of the insurance industry and concerns related to data privacy and proprietary information, such data sharing might be challenging to realize. A centralized organization overseeing this data sharing would be an ideal solution, but establishing such an entity comes with its own set of challenges, both logistical and regulatory.

Furthermore, the data utilized in this study are relatively old and originate from the USA. This poses questions about the generalizability of the findings to contemporary scenarios, especially in different geographical contexts like Europe or the Netherlands. The insurance landscape, regulatory environment, and risk profiles can vary significantly across regions and time periods. Thus, results in this study might not be universally applicable to all non-life insurance scenarios or datasets.

Another aspect to consider is the exclusion of expert judgment from the models. In the industry, experts often exclude certain link ratios based on their domain knowledge and experience. While the models showcased their predictive prowess, the integration of expert judgment could potentially enhance their performance further. Additionally, the speed of information processing is a crucial factor in real-time applications. While machine learning models, especially those as advanced as XGBoost, offer robust predictive capabilities, their computational complexity might pose challenges in scenarios where rapid information processing is essential. It remains uncertain whether the methods explored can seamlessly accommodate such real-time processing requirements.

8 Further research

Moving forward, there are multiple opportunities for more research in loss reserve predictions within non-life insurance. Combining expert opinions with machine learning forecasts presents a notable direction. Upcoming research could focus on developing hybrid models that merge the computational capabilities of algorithms with the specialized knowledge of industry experts.

Another significant area of exploration is the development of frameworks or platforms that facilitate secure and efficient data exchange among insurers. Addressing the challenges of data sharing, such platforms could ensure that multiple claim triangles across various LoBs can be used without compromising on data privacy or proprietary information.

Beyond XGBoost, the vast world of machine learning offers numerous algorithms that could be tailored for loss reserve predictions. Future studies could embark on a comparative journey, juxtaposing the performances of various algorithms in different contexts to unearth the most optimal techniques.

A particularly intriguing prospect is the analysis of individual claim data. With the advent of Natural Language Processing (NLP), there's untapped potential to mine insights from textual claim descriptions. Semantic analysis of these descriptions, powered by NLP, could provide early indicators of claim severity or the likelihood of litigation, factors pivotal in influencing reserve amounts.

Lastly, as the insurance industry gravitates towards real-time claim processing and dynamic pricing models, optimizing machine learning models for speed without compromising on accuracy becomes paramount. Ensuring that these models are not only theoretically robust but also practically applicable in fast-paced insurance scenarios will be a cornerstone of future research endeavors.

In essence, the findings of this research have opened the doors to a plethora of possibilities, each promising to further refine and revolutionize the domain of loss reserve predictions in non-life insurance.

References

[1] Kaas, R., Goovaerts, M., Dhaene, J., & Denuit, M. (2008). Modern Actuarial Risk Theory: Using R. https://doi.org/10.1007/978-3-540-70998-5doi:10.1007/978-3-540-70998-5.

[2] Merz, M., & Wüthrich, M. V. (2006). A credibility approach to the Munich chain-ladder method. *Blätter der DGVFM*, **27**, 619-628. https://doi.org/10.1007/BF02809220doi:10.1007/BF02809220.

[3] Radtke, M., Schmidt, K. D., & Schnaus, A. (Eds.). (2016). Handbook on Loss Reserving. Springer Cham. https://doi.org/10.1007/978-3-319-30056-6doi:10.1007/978-3-319-30056-6.

[4] Frees, E. W. (2018). Regression modeling with actuarial and financial applications. Cambridge University Press. https://doi.org/10.1017/CBO9780511814372 doi:CBO9780511814372

[5] Ruitenberg, P. L. (2019). Adapting a Hierarchical Gaussian Process model to predict the loss reserve of a non-life insurer (Master's essay). University of Twente, Behavioural, Management and Social Sciences. https://purl.utwente.nl/essays/77936https://purl.utwente.nl/essays/77936.

[6] Avanzi, B., Taylor, G., & Wong, B. (2015). Correlations between Insurance Lines of Business: An Illusion or a Real Phenomenon? Some Methodological Considerations. https://doi.org/10.1017/asb.2015.31doi:10.1017/asb.2015.31.

[7] Boratyńska, A. (2017). Robust Bayesian estimation and prediction of reserves in exponential model with quadratic variance function. *Insurance: Mathematics and Economics, Elsevier*, **76**(C), 135-140.

[8] Diers, D., & Linde, M. (2013). The multi-year non-life insurance risk in the additive loss reserving model. *Insurance: Mathematics and Economics*, **52**(3), 590-598. https://doi.org/10.1016/j.insmatheco.2013.03.004doi:10.1016/j.insmatheco.2013.03.004.

[9] Djehiche, B., & Löfdahl, B. (2016). Nonlinear reserving in life insurance: Aggregation and mean-field approximation. *Insurance: Mathematics and Economics*, **69**, 1-13. https://doi.org/10.1016/j.insmatheco.2016.04.002doi:10.1016/j.insmatheco.2016.04.002.

[10] England, P., Verrall, R. J., & Wüthrich, M. V. (2019). On the lifetime and one-year views of reserve risk, with application to IFRS 17 and Solvency II risk margins. *Insurance: Mathematics and Economics*, **85**, 74-88.

[11] Feng, R., & Yi, B. (2019). Quantitative modeling of risk management strategies: Stochastic reserving and hedging of variable annuity guaranteed benefits. *Insurance: Mathematics and Economics*, **85**. https://doi.org/10.1016/j.insmatheco.2018.12.003doi:10.1016/j.insmatheco.2018.12.003.

[12] Ferriero, A. (2016). Solvency capital estimation, reserving cycle and ultimate risk. *Insurance: Mathematics and Economics*, **68**(C), 162-168. https://doi.org/10.1016/j.insmatheco.2016.03.004doi:10.1016/j.insmatheco.2016.03.004.

[13] Fröhlich, A., & Weng, A. (2018). Parameter uncertainty and reserve risk under Solvency II. *Insurance: Mathematics and Economics*, **81**, 130-141. https://doi.org/10.1016/j.insmatheco.2017.10.004doi:10.1016/j.insmatheco.2017.10.004.

[14] Gigante, P., Picech, L., & Sigalotti, L. (2019). CALENDAR YEAR EFFECT MODELING FOR CLAIMS RESERVING IN HGLM. *ASTIN Bulletin*, **49**, 1-24. https://doi.org/10.1017/asb.2019.22doi:10.1017/asb.2019.22.

[15] Huang, J., Qiu, C., & Wu, X. (2015). Stochastic Loss Reserving in Discrete Time: Individual vs. Aggregate Data Models. *Communications in Statistics - Theory and Methods*, **44**(10), 2180-2206. https://doi.org/10.1080/03610926.2014.976473.

[16] Peters, G. W., Targino, R. S., & Wüthrich, M. V. (2017). Bayesian Modelling, Monte Carlo Sampling and Capital Allocation of Insurance Risks. *Risks*, **5**(4), 53. https://doi.org/10.3390/risks5040053.

[17] Wahl, F., Lindholm, M., & Verrall, R. (2019). The collective reserving model. *Insurance: Mathematics & Economics*, **87**, 34-50. https://doi.org/10.1016/j.insmatheco.2019.04.003.

[18] Wuthrich, M. V. (2018). Neural Networks Applied to Chain-Ladder Reserving. Available at SSRN: https://ssrn.com/abstract=2966126.

[19] Kuo, K. (2019). DeepTriangle: A Deep Learning Approach to Loss Reserving. *Risks*, **7**(3), 97. https://doi.org/10.3390/risks7030097.

[20] Gabrielli, A., & Wuthrich, M. V. (2018). Back-Testing the Chain-Ladder Method. Available at SSRN: https://ssrn.com/abstract=3211107https://ssrn.com/abstract=3211107.

[21] Gabrielli, A. (2019). A Neural Network Boosted Double Over-Dispersed Poisson Claims Reserving Model. Available at SSRN: https://ssrn.com/abstract=3365517.

[22] Avanzi, B., Taylor, G., & Wong, B. (2016). Stochastic loss reserving with the Tweedie distribution. *Insurance: Mathematics and Economics*, **70**, 327-340.

[23] Shi, Peng. (2014). A copula regression for modeling multivariate loss triangles and quantifying reserving variability. *Astin Bulletin*, **44**. https://doi.org/10.1017/asb.2013.23.

[24] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, **521**, 436–444. https://doi.org/10.1038/nature14539.

[25] Lopez, O., Milhaud, X., & Thérond, P.-E. (2019). A tree-based algorithm adapted to microlevel reserving and long development claims. *ASTIN Bulletin*, **49**(3), 741-762. https://doi.org/10.1017/asb.2019.12.

[26] Mack, T. (1999). The standard error of chain ladder reserve estimates: Recursive calculation and inclusion of a tail factor. *ASTIN Bulletin*, **29**(2), 361-366.

[27] Mack, T. (1993). Distribution-free calculation of the standard error of chain ladder reserve estimates. *ASTIN Bulletin*, **23**(2).

[28] Efron, B., & Tibshirani, R.J. (1994). *An Introduction to the Bootstrap* (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9780429246593.

[29] England, P.D. & Verrall, Richard. (2002). Stochastic Claims Reserving in General Insurance. *British Actuarial Journal*, **8**. https://doi.org/10.1017/S1357321700003809.

[30] McCullagh, P. and Nelder, J.A. (1989). *Generalized Linear Models*. 2nd Edition, Chapman and Hall, London. http://dx.doi.org/10.1007/978-1-4899-3242-6.

[31] Clark, David R. (2003). LDF Curve-Fitting and Stochastic Reserving: A Maximum Likelihood Approach. *Casualty Actuarial Society Forum*, Fall.

[32] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R*. Springer Texts in Statistics (Vol. 103).

[33] Waring, J., Lindvall, C., & Umeton, R. (2020). Automated machine learning: Review of the state-of-the-art and opportunities for healthcare. *Artificial Intelligence in Medicine*, **104**, 101822. https://doi.org/10.1016/j.artmed.2020.101822.

[34] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, **5**(2), 241-259. https://doi.org/10.1016/S0893-6080(05)80023-1.

[35] Breiman, L. (1996). Bagging predictors. *Machine Learning*, **24**(2), 123–140. https://doi.org/10.1007/BF00058655doi:10.1007/BF00058655.

[36] van der Laan, M. J., Polley, E. C., & Hubbard, A. E. (July 2007). Super Learner. U.C. Berkeley Division of Biostatistics Working Paper Series. Working Paper 222. https://biostats.bepress.com/ucbbiostat/paper222.

[37] Chen, T., Tang, L.-A., Sun, Y., Chen, Z., & Zhang, K. (2016). Entity Embedding-based Anomaly Detection for Heterogeneous Categorical Events. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'16)*. https://doi.org/10.48550/arXiv.1608.07502.

[38] LeDell, E. (2015). Scalable Ensemble Learning and Computationally Efficient Variance Estimation. UC Berkeley. ProQuest ID: LeDell_berkeley_0028E_15235. Merritt ID: ark:/13030/m5wt1xp7. Link.

[39] LeDell, E., & Poirier, S. (2020). H2O AutoML: Scalable Automatic Machine Learning. In *Proceedings of the 7th ICML Workshop on Automated Machine Learning*. H2O.ai, USA.

[40] Lantz, B. (2013). *Machine Learning with R: Learn how to use R to apply powerful machine learning methods and gain insight into real-world applications*. Packt Publishing.

[41] XGBoost. (n.d.). Wikipedia. Retrieved Month Day, Year, from https://en.wikipedia.org/wiki/XGBoost.

[42] Guo, C., & Berkhahn, F. (2016). Entity Embeddings of Categorical Variables. arXiv:1604.06737 [cs.LG]. https://doi.org/10.48550/arXiv.1604.06737.

[43] Liashchynskyi, P., & Liashchynskyi, P. (2019). Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS. arXiv:1912.06059. https://doi.org/10.48550/arXiv.1912.06059.

[44] Du, J., Grave, E., Gunel, B., Chaudhary, V., Celebi, O., Auli, M., Stoyanov, V., & Conneau, A. (2021). Self-training Improves Pre-training for Natural Language Understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 5408–5418). Association for Computational Linguistics.

[45] Iman, R. L. (2014). Latin Hypercube Sampling. In *Encyclopedia of Quantitative Risk Analysis and Assessment*. John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118445112.stat03803doi:10.1002/9781118445112.stat03803.

[46] Song, I. J., & Heo, W. (2022). Improving insurers' loss reserve error prediction: Adopting combined unsupervised-supervised machine learning techniques in risk management. *The Journal of Finance and Data Science*, **8**, 233-254. https://doi.org/10.1016/j.jfds.2022.09.003doi:10.1016/j.jfds.2022.09.003.

[47] Avanzi, B., Li, Y., Wong, B., & Xian, A. (2022). Ensemble distributional forecasting for insurance loss reserving. https://doi.org/10.48550/arXiv.2206.08541doi:10.48550/arXiv.2206.08541.

[48] Okine, A., Frees, E., & Shi, P. (2022). Joint Model Prediction and Application to Individual-Level Loss Reserving. *ASTIN Bulletin: The Journal of the IAA*, **52**(1), 91-116. https://doi.org/10.1017/asb.2021.28doi:10.1017/asb.2021.28.

[49] Chaoubi, I., Besse, C., Cossette, H., & Côté, M.-P. (2023). Micro-level reserving for general insurance claims using a long short-term memory network. *Applied Stochastic Models in Business and Industry*, **39**(3), 382-407. https://doi.org/10.1002/asmb.2750doi:10.1002/asmb.2750.

[50] Woundjiagué, A., Mbele Bidima, M. L. D., & Mwangi, R. W. (2019). An Estimation of a Hybrid Log-Poisson Regression Using a Quadratic Optimization Program for Optimal Loss Reserving in Insurance. *Advances in Fuzzy Systems*, **Volume 2019**, Article ID 1393946. https://doi.org/10.1155/2019/1393946doi:10.1155/2019/1393946.

[51] Meyers, G. (2015). *Stochastic Loss Reserving Using Bayesian MCMC Models*. CAS Monograph Series, Number 1. Casualty Actuarial Society.