Individual Claims Reserving: Using Machine Learning Methods

Dong Qiu

A Thesis
In
The Department
of
Mathematics and Statistics

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Science (Mathematics) at
Concordia University
Montreal, Quebec, Canada

December 2019

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Dong Qiu**

Entitled: **Individual Claims Reserving: Using Machine Learning Methods**

and submitted in partial fulfillment of the requirements for the degree of

Master of Science (Mathematics)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

______________________ Chair
Dr. Mélina Mailhot

______________________ Examiner
Dr. Frédéric Godin

______________________ Examiner
Dr. Mélina Mailhot

______________________ Supervisor
Dr. José Garrido

Approved by ______________________
Dr. Cody Hyndman, Chair
Department of Mathematics & Statistics

__________ Date

____
André G. Roy, Dean
Faculty of Arts and Science

Abstract

Individual Claims Reserving: Using Machine Learning Methods

Dong Qiu

To date, most methods for loss reserving are still used on aggregate data arranged in a triangular form such as the Chain-Ladder (CL) method and the over-dispersed Poisson (ODP) method. With the booming of machine learning methods and the significant increment of computing power, the loss of information resulting from the aggregation of the individual claims data into accident and development year buckets is no longer justifiable. Machine learning methods like Neural Networks (NN) and Random Forest (RF) are then applied and the results are compared with the traditional methods on both simulated data and real data (aggregate at company level).

Acknowledgements

First of all, I devote my greatest thanks to my supervisor Dr. José Garrido for his guidance throughout my study at Concordia. Each and every time after our meeting, I felt inspired and had my hands full of content to learn. He is always patient and thoughtful, and his advice is pivotal. Without his valuable instruction, this paper would not have been finished smoothly.

I also want to express my sincere gratitude to all the other teachers in the past three years of my study - especially Dr. Mélina Mailhot and Dr. Frédéric Godin, for their sincere opinions on my thesis.

In addition, I would also like to thank my close friends, for their sustained encouragement during the whole process of writing, and my boss for understanding my situation and being supportive of my personal development.

Last but not least, I would like to thank my mom Min Chen and my dad Youqian Qiu for their everlasting love. Without their support, I would not have chance to study here and to chase my dreams.

Contents

List of Figures

List of Tables

Introduction

In the case of non-life insurance, the related benefit is not paid to the insured necessarily as soon as the accident occurs, some years may pass between the actual occurrence and the final claim payment. There can be several reasons why the claim cannot be settled immediately, including further investigation, new information, court decisions, and so on.

This time gap is the reason why insurance companies must allocate sufficient loss reserves to cover any future payments for outstanding loss liabilities. To date, most methods for loss reserving still use aggregate data arranged in a triangular form, as shown in Table 1. Aggregate loss reserving data are placed in different cells for different accident and development years. Based on this, methods such as the Chain-Ladder (CL) or Bornhuetter-Ferguson (BF) algorithms are then applied to find some factors that can explain the development of payments from year to year, or to find the ultimate claims reserves at the finalization of the development period.

Wüthrich [2019] gives the notation of the aggregate claims reserves, where $X_{i,j}$ represents the payments made for claims with accident year i in development year j, and $C_{i,j} = \sum_{k=0}^{j} X_{i,k}$ as the total payments made for claims with accident year i until development year j. All the observations in the upper triangle are noted as $\mathcal{D}_I$, and the lower triangle $\mathcal{D}_I^c$ is the part that actuaries would like to predict.

Accident Year	Development Years					
i	0	1	$\dots$	j	$\dots$	$J-1$
1	$X_{1,0}$	$X_{1,1}$	$\dots$	$X_{1,j}$	$\dots$	$X_{1,J-1}$
$\vdots$						
i	$X_{i,0}$	$X_{i,1}$			$\dots$	
$\vdots$	*observations* $\mathcal{D}_I$			*to predict* $\mathcal{D}_I^c$		
$I-1$						
I	$X_{I,0}$	$X_{I,1}$	$\dots$	$X_{I,j}$	$\dots$	$X_{I,J-1}$

Table 1: Claims Development Triangle

Later on, in order to know how much the real payments may deviate from the predictions, it was natural to set the CL model into a stochastic framework. For the CL method, according to Wüthrich [2019], many stochastic models were developed, including the distribution-free CL model, the over-dispersed Poisson (ODP) model with MLE parameter estimates, and the Bayesian CL model. In Taylor and McGuire [2016], some other models are mentioned (such as the EDF Mack model, cross-classified models).

Nonetheless, with the booming of machine learning methods, and the significant increment of computing power, the loss of information resulting from the aggregation of the individual claims data into accident and development year buckets can now be prevented to a certain level. In Taylor et al. [2008], a GLM method is proposed to build the model for individual claim loss reserving in different cases.

Due to the pattern recognition capabilities of neural networks, Harej et al. [2017] used these on long-tailed and short-tailed claims, for reserving and pricing of the simulated data introduced by Taylor et al. [2008] mentioned above. Long-tailed and short-tailed patterns are modeled by two log-normal distributions with different parameters, and a copula is used to induce a dependence between them. There are 6,000 short-tailed claims and 4,000 long-tailed claims that are produced under this unified model. More samples are produced by allocating different numbers of claims from these two samples to each accident years.

Then, we start with the same approach as Harej et al. [2017], by applying machine learning methods on simulated data, to which we add Random Forest (RF) along with a modified version of the cascading method. The RF method is one of the most accurate statistical learning algorithms available. It produces a highly accurate classifier for most datasets, and it runs efficiently on large databases. It corrects decision trees' tendency to overfit their training set. Furthermore, instead of finding the development pattern of payments, the RF method puts the individual claims that have the same pattern or features into the same category and predicts the value by averaging all the values at the end node.

To eliminate the limitation of the cascading method, we found some approaches proposed by Aleandri [2017], by separating the predicting procedure into different parts. It allows us to predict the closing delay time, the final amount of payments, and then individual loss reserving. However, in this thesis, due to the lack of real reserving data, we are not able to use them for comparison purposes.

Chapter 1

Classical Methods

1.1 Chain Ladder Algorithm

The Chain Ladder (CL) method consists in a deterministic algorithm, which is widely used to forecast future claim reserves. The main idea of the CL algorithm (see Wüthrich [2019]) is based on the assumption that for all accident years i, the cumulative payments behave according to the same pattern of changes in the sequence of the coming development years. For a given accident year i, and development year j, the cumulative payments have the following relation

$$C_{i,j+1} \approx f_j C_{i,j}, \qquad i \in \{1, ..., I\},\ j \in \{1, ..., J\}, \tag{1.1}$$

where the f_j are called *CL factors*, *age-to-age factors* or *link ratios.*

Ultimately, we would like to know the cumulative claims reserves $C_{i,J-1}$ for each accident year, which can be calculated by equation

$$\hat{C}^{CL}_{i,J-1} = C_{i,I-i} \prod_{j=I-i}^{J-2} \hat{f}^{CL}_j, \qquad \text{with } i > I - J + 1, \tag{1.2}$$

and more generally, $\hat{C}^{CL}_{i,n} = C_{i,I-i} \prod_{j=I-i}^{n-1} \hat{f}^{CL}_j$ for $i + n > I$, where $\hat{f}^{CL}_j$ is the estimator for

f_j. In Wüthrich et al. [2010], estimators of f_j are given as

$$\hat{f}_j^{CL} = \frac{\sum_{i=1}^{i^*(j+1)} C_{i,j+1}}{\sum_{i=1}^{i^*(j+1)} C_{i,j}} = \sum_{i=1}^{i^*(j+1)} \frac{C_{i,j}}{\sum_{n=1}^{i^*(j+1)} C_{n,j}} \frac{C_{i,j+1}}{C_{i,j}}, \tag{1.3}$$

where $i^*(j) = I - j$ is the last observed development year, which is the last diagonal in the observed claims development triangle (See Table 1). If weights are added, then the weighted average CL factors can be represented by:

$$\hat{f}_j^{CL} = \sum_{k=1}^{J-1} \omega_{kj} \hat{f}_{ik}, \tag{1.4}$$

where $f_{ij} = \frac{C_{i,j+1}}{C_{i,j}}$ and $\sum_{i=1}^{J-j} \omega_{ij} = 1$. When $\omega_{ij} = \frac{C_{i,j}}{\sum_{i=1}^{J-j} C_{i,j}}$, then the weighted average CL factors in (1.4) reduce to those used in (1.3).

Easily we can calculate the total predicted CL reserve at time I for accident year $i > I - J + 1$, which is given as

$$\hat{X}_i^{CL} = \hat{C}_{i,J-1}^{CL} - C_{i,I-i} = C_{i,I-i}\Big(\prod_{j=I-i}^{J-2} \hat{f}_j^{CL} - 1 \Big). \tag{1.5}$$

Therefore, the total yearly predicted claims in development year J for all accident years can be written as:

$$\begin{aligned}
\hat{\mathcal{R}}_J^{CL} &= \sum_{i=1}^{I} (\hat{C}_{i,J} - \hat{C}_{i,J-1}) \\
&= \sum_{i=1}^{I} (C_{i,I-i} \prod_{j=I-i}^{J-1} \hat{f}_j^{CL} - C_{i,I-i} \prod_{j=I-i}^{J-2} \hat{f}_j^{CL}) \\
&= \sum_{i=1}^{I} C_{i,I-i} \hat{f}_{J-1}^{CL},
\end{aligned}$$

along with the total predicted claim reserve over all accident years, which is given by:

$$\hat{\mathcal{R}}^{CL} = \sum_{i > I-J+1} \hat{\mathcal{R}}_i^{CL}. \tag{1.6}$$

For example, Tables 1.1 and 1.2 use the synthetic data presented in Chapter 4 to illustrate the above formulas. They follow the steps of the CL algorithm by calculating the CL factors first; calculate the cumulative losses for the lower triangle; then compute the estimated loss reserves and compare them with the available data. The CL algorithm predicts overall reserves precisely when the claims are evenly distributed over different development patterns or when the line of business is small; however, it performs poorly when the distribution of the patterns varies more.

A Robust General Multivariate Chain Ladder (GMCL) Method

A non-life insurance company typically divides portfolios into m correlated sub-portfolios, where $m = 1, \ldots, M$ and M is the total number of sub-portfolios, so that certain homogeneity properties on each sub-portfolio are satisfied. The GMCL method is explained by Peremans et al. [2018]. The overall outstanding reserve $\mathcal{R}$ that will need to be paid in the future is defined as:

$$\mathcal{R} = \sum_{m=1}^{M} \sum_{i=1}^{I} (C_{i,J}^{(m)} - C_{i,J-i}^{(m)}). \tag{1.7}$$

Let $\mathbf{C}_{i,j} = (C_{i,j}^{(1)}, \ldots, C_{i,j}^{(M)})$ denote the vector of cumulative claims of accident period i and development period j for business line m. Consider the following model structure from development period j to $j+1$:

$$\mathbf{C}_{i,j+1} = \mathbf{A}_j + \mathbf{B}_j \mathbf{C}_{i,j} + \boldsymbol{\epsilon}_{i,j}, \quad i = 1, \ldots, I, \tag{1.8}$$

where $\mathbf{A}_j$ is the M vector containing the intercepts $\beta_{0,j}^{(1)}, ..., \beta_{0,j}^{(M)}$, $\mathbf{B}_j$ is the the $M \times M$ matrix that contains the development parameters $\beta_{1,j}^{(m)}, ..., \beta_{M,j}^{(m)}$ for run-off triangle m in row j, and $\boldsymbol{\epsilon}_{i,j} = (\epsilon_{i,j}^{(1)}, ..., \epsilon_{i,j}^{(M)})$ are independent (over i) and symmetrically distributed random vectors representing the error terms. Moreover, it is assumed that the errors $\boldsymbol{\epsilon}_{i,j}$ satisfy

$$\begin{aligned} &\mathbb{E}[\boldsymbol{\epsilon}_{i,j} | \mathcal{D}_{i,j}] = 0, \\ &\mathrm{Cov}[\boldsymbol{\epsilon}_{i,j} | \mathcal{D}_{i,j}] = \mathrm{diag}[\mathbf{C}_{i,j}]^{1/2} \boldsymbol{\Sigma}_k \mathrm{diag}[\mathbf{C}_{i,j}]^{1/2}, \end{aligned} \tag{1.9}$$

where $\mathcal{D}_{i,j}$ is the set of cumulative claims for accident year i up to and including development

Accident Year	Development Year (in thousand)																			
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1998	86286	173932	241731	291918	329325	357938	380474	398709	413789	426441	437161	446314	454158	460880	466638	471568	475809	479412	482532	485155
1999	86408	174178	242066	292313	329772	358413	380971	399208	414293	426962	437694	446848	454680	461398	467181	472125	476343	479973	483076	
2000	86334	174024	241884	292107	329577	358244	380824	399113	414219	426901	437663	446838	454712	461453	467241	472192	476433	480060		
2001	86301	173963	241778	291982	329424	358060	380627	398901	413982	426653	437417	446586	454434	461181	466952	471895	476126			
2002	86249	173860	241641	291793	329201	357832	380372	398609	413688	426340	437080	446247	454068	460815	466581	471507				
2003	86219	173801	241555	291702	329119	357744	380274	398529	413617	426277	437029	446191	454043	460762	466531					
2004	86306	173975	241783	291954	329377	358006	380550	398767	413840	426486	437224	446375	454214	460923						
2005	86062	173487	241116	291166	328533	357101	379587	397815	412879	425499	436241	445386	453237							
2006	86262	173889	241671	291846	329266	357897	380455	398719	413789	426460	437197	446368								
2007	86234	173827	241583	291714	329117	357707	380246	398447	413521	426153	436873									
2008	86195	173743	241488	291625	329028	357635	380165	398403	413505	426151										
2009	86290	173940	241751	291954	329405	358039	380618	398868	413964											
2010	86362	174089	241945	292151	329587	358204	380740	398995												
2011	86247	173853	241623	291793	329196	357807	380345													
2012	86263	173888	241674	291846	329254	357854														
2013	86399	174166	242032	292281	329732															
2014	86115	173581	241253	291365																
2015	86273	173912	241712																	
2016	86328	174018																		
2017	86166																			
f_j^{CL}	2.7823	2.4145	2.2544	2.1772	2.1244	2.0958	2.0767	2.0609	2.0492	2.0419	2.0359	2.0274	2.0272	2.0208	2.0180	2.0160	2.0136	2.0107	0.0054	

Table 1.1: Calculating CL Factors on Aggregate Data with CL Algorithm

Accident Year	Development Year (in thousand)																					
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Prediction $\hat{X}_i^{CL}$	Real data X_i^{CL}
1998																					0	0
1999																				485702	2625	2624
2000																			483174	485801	5740	5771
2001																		479746	482858	485482	9355	9371
2002																	475735	479352	482461	485084	13577	13581
2003																471465	475693	479310	482419	485041	18510	18541
2004															466694	471630	475860	479478	482588	485211	24287	24278
2005														459950	465708	470634	474855	478465	481569	484186	30949	30998
2006													454213	460940	466711	471648	475878	479496	482606	485229	38861	38885
2007												446027	453866	460588	466355	471287	475514	479129	482237	484858	47985	47955
2008											436885	446039	453879	460601	466368	471301	475528	479143	482251	484872	58721	58777
2009										426625	437371	446536	454384	461114	466887	471825	476057	479676	482787	485411	71447	71541
2010									414091	426756	437505	446672	454523	461255	467030	471970	476203	479823	482935	485560	86565	86428
2011								398585	413665	426317	437055	446213	454056	460781	466549	471484	475713	479329	482438	485061	104715	104708
2012							380395	398637	413719	426373	437112	446271	454115	460841	466611	471546	475775	479392	482502	485124	127270	127213
2013						358391	380966	399235	414340	427013	437769	446941	454797	461533	467311	472254	476489	480112	483226	485852	156120	155984
2014					328722	357294	379800	398013	413071	425705	436428	445573	453404	460120	465880	470808	475030	478642	481746	484365	192999	193305
2015				291891	329315	357938	380485	398731	413817	426473	437216	446377	454222	460950	466721	471657	475887	479505	482616	485239	243526	243357
2016			241854	292062	329508	358148	380708	398965	414059	426723	437472	446639	454489	461220	466994	471934	476167	479786	482898	485523	311505	311655
2017		173692	241401	291515	328891	357478	379995	398218	413284	425924	436653	445802	453637	460356	466120	471050	475275	478888	481994	484614	398447	398497
f_j^{CL}	2.7823	2.4145	2.2544	2.1772	2.1244	2.0958	2.0767	2.0609	2.0492	2.0419	2.0359	2.0274	2.0272	2.0208	2.0180	2.0160	2.0136	2.0107	0.0054		$\hat{X}^{CL}$ 1943212	X^{CL} 1943480

Table 1.2: Prediction on Aggregate Data with CL Algorithm

year j, $\mathbf{\Sigma}_k$ is a symmetric positive definite $M \times M$ matrix, and diag is the operator that turns its arguments into a diagonal matrix.

Therefore, the parameters $\mathbf{A}_j$, $\mathbf{B}_j$ and $\mathbf{\Sigma}_k$ are unknown model parameters and need to be estimated from historical claims in order to predict future losses.

The Seemingly Unrelated Regression (SUR) model is then used based on (1.8). For historical claims only, the following system of equations is obtained

$$\begin{pmatrix} \boldsymbol{y}_j^{(1)} \\ \vdots \\ \boldsymbol{y}_j^{(M)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{X}_j^{(1)} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \boldsymbol{X}_j^{(M)} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_j^{(1)} \\ \vdots \\ \boldsymbol{\beta}_j^{(M)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_j^{(1)} \\ \vdots \\ \boldsymbol{\epsilon}_j^{(M)} \end{pmatrix}, \quad \text{for } i = 1, \dots, n(j) \text{ with } n(j) = I - j, \tag{1.10}$$

where I is the latest accident period, $j = 0, 1, \cdots, J-1$ for $m = 1, \dots, M$ and where the following holds true:

- $\boldsymbol{y}_j^{(m)} = (C_{1,j+1}^{(m)}, \dots, C_{n(j),j+1}^{(m)})'$ is the $n(j)$ vector of all observed losses at development period $j+1$ from triangle m;

- $\boldsymbol{X}_j^{(m)} = ((1, C_{1,j}^{(m)})', \dots, (1, C_{n(j),j}^{(m)})')'$ is the $n(j) \times (M+1)$ matrix of the first $n(j)$ observations at development period j from each triangle, including the constant 1 as the intercept $C_{0,j}^{m}$. Hence, $\boldsymbol{X}_j^{(1)} = \dots = \boldsymbol{X}_j^{(M)}$;

- $\boldsymbol{\beta}_j^{(m)} = (\beta_{0,j}^{(m)}, \dots, \beta_{M,j}^{(m)})'$ is the $M+1$ vector of development parameters of triangle m, including the intercept, here $(\beta_{0,j}^{(m)}, \dots, \beta_{M,j}^{(m)})$ does not depend on subscript m, but it makes it easier to align with the rest part of the equation;

- $\boldsymbol{\epsilon}_j^{(m)} = (\epsilon_{1,j}^{(m)}, \dots, \epsilon_{n(j),j}^{(m)})'$ is the $n(j)$ vector of error terms of triangle m.

The set of the first $n(j)$ claims up to and including development period j is presented as $\mathcal{D}_j = \{C_{i,j} | 1 \le i \le n(j), j \le J-1\}$. From (1.9) it follows that

$$\operatorname{Cov}[\boldsymbol{\epsilon}_j | \mathcal{D}_j] = \mathbb{E}[\boldsymbol{\epsilon}_j \boldsymbol{\epsilon}_j' | \mathcal{D}_j] = \operatorname{diag}[\mathbf{C}_j]^{1/2} (\mathbf{\Sigma}_k \otimes \mathbf{I}_{n(j)}) \operatorname{diag}[\mathbf{C}_j]^{1/2}, \tag{1.11}$$

where $\boldsymbol{\epsilon}_j = (\boldsymbol{\epsilon}_j^{(1)'}, \dots, \boldsymbol{\epsilon}_j^{(M)'})'$, $\mathbf{C}_j = (\mathbf{C}_j^{(1)'}, \dots, \mathbf{C}_j^{(M)'})'$ with $\mathbf{C}_j^{(m)'} = (C_{1,j}^{(m)'}, \dots, C_{n(j),j}^{(m)'})$ for

$m = 1, ..., M$, and $\otimes$ is the Kronecker product. Pre-multiplying both sides of Equation (1.10) by diag$[\mathbf{C}_k]^{1/2}$ leads to the following linear regression model:

$$\begin{pmatrix} \boldsymbol{y}_j^{(1)*} \\ \vdots \\ \boldsymbol{y}_j^{(M)*} \end{pmatrix} = \begin{pmatrix} \boldsymbol{X}_j^{(1)*} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \boldsymbol{X}_j^{(M)*} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_j^{(1)} \\ \vdots \\ \boldsymbol{\beta}_j^{(M)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_j^{(1)*} \\ \vdots \\ \boldsymbol{\epsilon}_j^{(M)*} \end{pmatrix}, \tag{1.12}$$

with $\boldsymbol{y}_j^{(m)*} = \text{diag}[\mathbf{C}_k^{(m)}]^{-1/2}\boldsymbol{y}_j^{(m)}$, $\boldsymbol{X}_j^{(m)*} = \text{diag}[\mathbf{C}_k^{(m)}]^{-1/2}\boldsymbol{X}_j^{(m)}$, and $\boldsymbol{\epsilon}_j^{(m)*} = \text{diag}[\mathbf{C}_k^{(m)}]^{-1/2}\boldsymbol{\epsilon}_j^{(m)}$.

The generalized least squares (GLS) is an adaptation of least squares that can handle any type of correlation. Therefore, the estimator for the model in (1.12) becomes

$$\hat{\boldsymbol{\beta}}_j = (\boldsymbol{X}_j^{*'})(\boldsymbol{\Sigma}_j^{-1} \otimes \boldsymbol{I}_{n(j)})(\boldsymbol{X}_j^*)^{-1}\boldsymbol{X}_j^{*'}(\boldsymbol{\Sigma}_j^{-1} \otimes \boldsymbol{I}_{n(j)})\boldsymbol{y}_j^*, \tag{1.13}$$

where $\boldsymbol{X}_j^* = \text{diag}[\boldsymbol{X}_j^{(1)*}, ..., \boldsymbol{X}_j^{(M)*}]$ is a block diagonal matrix of size $n(j)M \times M(M+1)$, and $\boldsymbol{y}_j^* = (\boldsymbol{y}_j^{(1)*}, ..., \boldsymbol{y}_j^{(M)*})$. A feasible GLS (FGLS) estimator is usually introduced to estimate the unknown $\boldsymbol{\Sigma}_j$. FGLS replaces the unknown matrix $\boldsymbol{\Sigma}_j$ in (1.13) with $\hat{\boldsymbol{\Sigma}}_j = (\hat{\boldsymbol{\epsilon}}_j^{(1)*}, \dots, \hat{\boldsymbol{\epsilon}}_j^{(M)*})'(\hat{\boldsymbol{\epsilon}}_j^{(1)*}, \dots, \hat{\boldsymbol{\epsilon}}_j^{(M)*})/n(j)$, where $\hat{\boldsymbol{\epsilon}}_j^{(m)*}$ are the residuals obtained from estimating (1.12) by least squares.

It has been shown that FGLS estimators in the GMCL model are very sensitive to outliers. Therefore, a robust methodology is then proposed by Peremans et al. [2018] for reserve estimates and outlier detection by combining robust SUR estimators with the GMCL model.

The system of equations in (1.12) can be rewritten as another linear regression model by reordering the equations. Let $\boldsymbol{\mathcal{Y}}_{i,j}^*$, $\boldsymbol{\mathcal{X}}_{i,j}^*$ and $\boldsymbol{e}_{i,j}^*$ be the subvector or submatrix of $\boldsymbol{y}_j$, $\boldsymbol{X}_j$ and $\boldsymbol{\epsilon}_j^*$ respectively by extracting rows $i, i+n(j), ..., i+n(j)(M-1)$.

Then the system of equations in (5) is equivalent to

$$\boldsymbol{\mathcal{Y}}_{i,j}^* = \boldsymbol{\mathcal{X}}_{i,j}^*\boldsymbol{\beta}_j + \boldsymbol{e}_{i,j}^*, \quad i = 1, ..., n(k). \tag{1.14}$$

The Cov$[\boldsymbol{e}_{i,j}^*|\mathcal{D}_{i,j}]$ can be easily obtained by $\boldsymbol{\Sigma}_j$. Decompose the covariance matrix $\boldsymbol{\Sigma}_k$ into a shape component $\boldsymbol{\Gamma}_j$ and a scale parameter σ_j such that $\boldsymbol{\Sigma}_j = \sigma_j^2\boldsymbol{\Gamma}_j$ with the determinant of the matrix $|\boldsymbol{\Gamma}_j| = 1$.

Let $\boldsymbol{e}^*_{i,j}(\boldsymbol{b})$ be equal to $\boldsymbol{\mathcal{Y}}^*_{i,j} - \boldsymbol{\mathcal{X}}^*_{i,j}\boldsymbol{b}$ for any $M(M+1)$ vector $\boldsymbol{b}$ according to the SUR representation in (1.14). Then, given an initial estimator of the scale $\hat{\sigma}_j$, the MM-estimators $(\hat{\boldsymbol{\beta}}_j, \hat{\boldsymbol{\Gamma}}_j)$ minimize

$$\frac{1}{n(j)}\sum_{i=1}^{n(j)} \rho\left(\frac{\sqrt{\boldsymbol{e}^*_{i,j}(\boldsymbol{b})'\boldsymbol{G}^{-1}\boldsymbol{e}^*_{i,j}(\boldsymbol{b})}}{\hat{\sigma}_k}\right), \tag{1.15}$$

over all $M(M+1)$ vectors $\boldsymbol{b}$ and positive definite symmetric $M \times M$ matrices $\boldsymbol{G}$, where $\boldsymbol{G}$ represents $\boldsymbol{\Gamma}$, with the determinant of the matrix $|\boldsymbol{G}| = 1$. The MM-estimator for covariance is defined as $\hat{\boldsymbol{\Sigma}}_j = \hat{\sigma}_j^2\hat{\boldsymbol{\Gamma}}_j$. Evidently, taking $\rho(x) = x^2$ yields the iterated FGLS estimator. To be robust against outliers, it is necessary to consider bounded ρ functions. More specifically, we assume that the function ρ satisfies the following conditions:

- ρ is symmetric, twice continuously differentiable and satisfies $\rho(0) = 0$;
- ρ is strictly increasing on $[0, c]$ and constant on $[c, \infty]$ for some $c > 0$.

The most favored family of ρ functions for MM-estimators is the class of Tukey bi-square ρ functions given by $\rho(x) =\min(x^2/2 - x^4/2c^2 + x^6/6c^4, c^2/6)$. Under the SUR model with normally distributed errors, the tuning parameter $c > 0$ is usually chosen to obtain a certain level of asymptotic efficiency. In the paper of Peremans et al. [2018], the Tukey bi-square ρ function is always considered with tuning parameter $c = 5.1229$.

An initial estimator of scale $\hat{\sigma}_k$ is required for MM-estimators. This scale estimator should be robust in order for MM-estimators to be robust. Therefore, highly robust S-estimators have been introduced for SUR models to obtain a highly robust scale estimator.

Starting from the initial S-estimates, MM-estimates are computed simply by iterating the following estimating equations until convergence according to Maronna et al. [2006]:

$$\hat{\boldsymbol{\beta}}_j = (\boldsymbol{X}_j^{*'})(\hat{\boldsymbol{\Sigma}}_j^{-1} \otimes \boldsymbol{D}_{n(j)})\boldsymbol{X}_j^*)^{-1}\boldsymbol{X}_j^{*'}(\hat{\boldsymbol{\Sigma}}_j^{-1} \otimes \boldsymbol{D}_{n(j)})\boldsymbol{y}_j^*,$$

$$\hat{\boldsymbol{\Sigma}}_j = M(\boldsymbol{e}^*_{1,j}(\hat{\boldsymbol{\beta}}_j), \ldots, \boldsymbol{e}^*_{n(j),j}(\hat{\boldsymbol{\beta}}_j))\boldsymbol{D}_j(\hat{\boldsymbol{\beta}}_j), \ldots, \boldsymbol{e}^*_{n(j),j}(\hat{\boldsymbol{\beta}}_j))'\Big(\sum_{i=1}^{n(j)} \rho'(d_{1,j})d_{i,j}\Big),$$

with $\boldsymbol{D}_j = \text{diag}[\omega(d_{1,j}), \dots, \omega(d_{n(j),j})]$, where $\omega(x) = \rho'(x)/x$, $d^2_{i,j} = \boldsymbol{e}^*_{i,j}(\hat{\boldsymbol{\beta}}_j)'\hat{\boldsymbol{\Sigma}}_j^{-1}\boldsymbol{e}^*_{i,j}(\hat{\boldsymbol{\beta}}_j)$, and $\boldsymbol{e}^*_{i,j}(\hat{\boldsymbol{\beta}}_j) = \boldsymbol{\mathcal{Y}}^*_{i,j} - \boldsymbol{\mathcal{X}}^*_{i,j}\hat{\boldsymbol{\beta}}_j$ are the residuals derived from the representation in (1.14).

1.2 Bornhuetter-Ferguson Algorithm

The Bornhuetter-Ferguson (BF) algorithm is similar to the CL algorithm, but it adds an assumption on prior information $\hat{\mu}_i$ for the expected ultimate claims of accident year i. Once we have a claims development pattern $(\gamma_j)_{j=0,1,\dots,J-1}$, which is the proportion of total loss for any accident year observed in development year j, it will allow us to predict the reserves using

$$\hat{X}^{BF}_{i,j} \approx \gamma_j \hat{\mu}_i, \tag{1.16}$$

under the normalization $\sum_{j=0}^{J-1} \gamma_j = 1$, and the γ_j have to be estimated based on observations.

An expert should give the prior information $\hat{\mu}_i$ here instead of basing it on the past data in $\mathcal{D}_I$, which is the claim reserve information we have from the upper triangle. Wüthrich [2019] defines the following estimators for the development pattern,

$$\hat{\gamma}_0^{BF} = \hat{\beta}_0^{BF}, \tag{1.17}$$

$$\hat{\gamma}_j^{BF} = \hat{\beta}_j^{BF} - \hat{\beta}_{j-1}^{BF}, \qquad \text{for } j = 1, ..., J-2, \tag{1.18}$$

$$\hat{\gamma}_j^{BF} = 1 - \hat{\beta}_{J-2}^{BF}, \tag{1.19}$$

where $\beta_j^{BF} = \prod_{l=j}^{J-2} \frac{1}{\hat{f}_l^{CL}} = \frac{\prod_{l=o}^{j-1} \hat{f}_l^{CL}}{\prod_{l=0}^{J-2} \hat{f}_l^{CL}}$. Therefore the ultimate claim estimate $\hat{C}_{i,J-1}$, for $i > I - J + 1$, is estimated by the BF method

$$\hat{C}^{BF}_{i,J-1} = C_{i,I-i} + \hat{\mu}_i \sum_{j=I-i+1}^{J-1} \hat{\gamma}_j^{BF} = C_{i,I-i} + \hat{\mu}_i(1 - \hat{\beta}^{BF}_{I-i}), \tag{1.20}$$

and then the claim reserve at time I for accident year $i > I - J + 1$ is

$$\hat{\mathcal{R}}_i^{BF} = \hat{\mu}_i \sum_{j=I-i+1}^{J-1} \hat{\gamma}_j{}^{BF} = \hat{\mu}_i(1 - \hat{\beta}^{BF}_{I-i}). \tag{1.21}$$

Finally the total outstanding loss liabilities would just be the sum of claim reserves from all accident years

$$\hat{\mathcal{R}}^{BF} = \sum_{i>I-J+1} \hat{\mathcal{R}}_i^{BF}. \tag{1.22}$$

No examples are given for the BF methods, since for the simulated data, no prior information by experts is available. However, a comparison of the CL and BF algorithms is given in Wüthrich [2019], which shows that they have the same structure:

$$\hat{C}_{i,J-1}^{CL} = C_{i,I-i} + \hat{C}_{i,J-1}^{CL}(1 - \hat{\beta}_{I-i}^{BF}), \tag{1.23}$$

$$\hat{C}_{i,J-1}^{BF} = C_{i,I-i} + \hat{\mu}_i(1 - \hat{\beta}_{I-i}^{BF}). \tag{1.24}$$

The only difference is that for the BF method the external estimate $\hat{\mu}_i$ is used for the final claim and in the CL method, it is an observation based on the estimated $\hat{C}_{i,J-1}^{CL}$.

Chapter 2

Stochastic Methods

The previous chapter reviews some classical algorithms that provide ways to calculate the claim reserves, however as a measure of preciseness, there is also a need to validate the prediction uncertainty of these models.

In Wüthrich [2019], the conditional mean square error of prediction (MSEP) was used to validate model uncertainty. It is the most popular prediction uncertainty measure, and it can be calculated or estimated explicitly in many examples. Assume $\hat{X}$ is a $\mathcal{D}_I$-measurable response variable for the random variable X. The conditional MSEP is defined by

$$msep_{X|\mathcal{D}_I}(\hat{X}) = \mathbb{E}[(X - \hat{X})^2|\mathcal{D}_I], \tag{2.1}$$

which can also be written as

$$msep_{X|\mathcal{D}_I}(\hat{X}) = \mathbb{V}(X|\mathcal{D}_I) + (\mathbb{E}[X - \hat{X}|\mathcal{D}_I])^2, \tag{2.2}$$

where in the righthand side of the equation, the first part is called *process uncertainty*, and the second part is called *parameter estimation error* or *bias*. In order to minimize the conditional MSEP, $\hat{X}$ should be chosen such that its expected value is the same as $\mathbb{E}[X|\mathcal{D}_I]$, if all parameters are known and if we can calculate $\mathbb{E}[X|\mathcal{D}_I]$. In other cases, $\mathbb{E}[X|\mathcal{D}_I]$ needs to be estimated as accurately as possible, and the possible sources of parameter uncertainty in this estimation need to be determined.

In order to analyze this prediction uncertainty, the claim reserving algorithms must be set in a stochastic framework, which means the variables $\{X_i\}$ have to be put into a family, where each one is indexed by a parameter i, where i belongs to some index set I, and it has to be measurable with respect to some measurable space $\left(S, \sum\right)$.

Several stochastic methods are mentioned by Taylor and McGuire [2016] and by Wüthrich [2019], but only the following three main methods are reviewed here:

1. Mack models,
2. Cross-Classified models,
3. Bayesian CL models.

2.1 Mack Models

The non-parametric Mack model is the most basic of Mack models, which assumes the three following conditions:

(M1) Accident years are stochastic independent, i.e., incremental payments X_{i_1,j_1} and X_{i_2,j_2} are independent if $i_1 \neq i_2$,

(M2) for each i, cumulative payments $C_{i,j}$ form a Markov process, as j varies,

(M3) there exist $f_j > 0$, $j \in \{0, ..., J-1\}$, and $\sigma_j^2 > 0$, $j \in \{0, ..., J-1\}$, such that for all $i \in \{1, ..., I\}$ and $j \in \{1, ..., J-1\}$,

   (a) $\mathbb{E}[C_{i,j+1}|C_{i,j}] = f_j C_{i,j}$,

   (b) $\mathbb{V}[C_{i,j+1}|C_{i,j}] = \sigma_j^2 C_{i,j}$.

As we can see from these three conditions, and comparing to the previous CL algorithm, it is a stochastic model in the sense that it considers both expected values (f_j) and the variances (σ_j) of observations and the reason why it is called distribution-free is that it does not assume a known distribution for the observations, we will see the difference later when introducing other models.

Wüthrich et al. [2010] give unbiased estimators for both f_j and σ_j, with $n(j) = I - j$,

$$\hat{f}_j = \frac{\sum_{i=1}^{n(j+1)} C_{i,j+1}}{\sum_{i=1}^{n(j+1)} C_{i,j}} = \sum_{i=1}^{n(j+1)} \frac{C_{i,j}}{\sum_{k=1}^{n(j+1)} C_{k,j}} \frac{C_{i,j+1}}{C_{i,j}}, \tag{2.3}$$

$$\hat{\sigma}_j = \frac{1}{n(j+1)-1} \sum_{i=1}^{n(j+1)} C_{i,j}\Big(\frac{C_{i,j+1}}{C_{i,j}} - \hat{f}_j\Big)^2. \tag{2.4}$$

By adding the variance, it makes it possible to compute the MSEP for $\hat{C}$ using (2.2). In this case,

$$msep_{C|\mathcal{D}_I}(\hat{C}) = \mathbb{V}(C|\mathcal{D}_I) + (\mathbb{E}[C|\mathcal{D}_I] - \hat{C})^2, \tag{2.5}$$

where $\mathbb{E}[X|\mathcal{D}_I] - \hat{X} = 0$. By using (M3b) recursively, (2.5) can be written as

$$msep_{C|\mathcal{D}_I}(\hat{C}) = \mathbb{V}(C|\mathcal{D}_I) = \sum_{i=2}^{I} \Big(C_{i,I-i} \prod_{j=I-i}^{J-2} \sigma_j^2). \tag{2.6}$$

The CL factor and variance can be calculated using (2.3) and (2.4), except for the last variance since there will not be enough information. According to Mack [1993], if $\hat{f}_{J-2} = 1$ and if the claims development is believed to be finished after $J-2$ years, we can put $\hat{\sigma}_{J-2} = 0$. If not, we extrapolate the usually exponentially decreasing series $\sigma_1, ..., \sigma_{J-4}, \sigma_{J-3}$ by one additional member, for instance by log-linear regression or more simply by requiring that $\sigma_{J-4}/\sigma_{J-3} = \sigma_{J-3}/\sigma_{J-2}$ holds at least as long as $\sigma_{J-4} > \sigma_{J-3}$. This last possibility leads to:

$$\hat{\sigma}_{J-2} = min\{\frac{\hat{\sigma}_{J-3}^4}{\hat{\sigma}_{J-4}^2}; \hat{\sigma}_{J-3}^2; \hat{\sigma}_{J-4}^2\}. \tag{2.7}$$

Table 2.1 gives an example of the calculation of CL factors and variances for the data in Sample 3, which will be defined later in Chapter 4.

	0	1	2	3	4	5	6	7	8	9	10
f_j^{CL}	2.0163	1.3896	1.2077	1.1285	1.0864	1.0631	1.0483	1.0372	1.0308	1.0252	1.0214
σ_j	872.84	479.32	797.58	875.44	531.01	318.07	685.87	384.31	393.44	403.36	632.96
	11	12	13	14	15	16	17	18			*msep*
	1.0177	1.0140	1.0123	1.0104	1.0094	1.0078	1.0083	1.0046			-
	948.43	809.23	1760.62	1067.58	684.23	441.54	9.38	1.00			6,313,104,379.62

Table 2.1: CL Factor and Variance from Sample 3

Two other generalized linear models (GLMs) that are from the Mack family are extensions of this non-parametric Mack model: one of them is called Exponential Dispersion Family parametric Mack model (EDF Mack model), which simply replaces (M3b) above with a distributional assumption $X_{i,j+1}|C_{i,j} \sim EDF(\delta_{ij}, \phi_{ij}; a, b, c)$. The form of the variance allowed in the EDF Mack model is more general than in the non-parametric Mack model. The other extension is called Over-Dispersed Poisson (ODP) Mack model, which replaces (M3b) with another distributional assumption $X_{i,j+1}|C_{i,j} \sim ODP(\mu_{ij}, \phi_{ij})$.

According to Taylor and McGuire [2016], under the assumption of the ODP Mack model, and if in addition the dispersion parameters ϕ_{ij} are just column dependent ($\phi_{ij} = \phi_j$), then the $\hat{f}_j$ from (2.3) are minimum variance unbiased estimators (MVUEs) of the f_j. The $\hat{C}_{i,j}$ and $\hat{X}$, under the same condition are also MVUEs of $C_{i,j}$ and X.

This MVUE result is much stronger than the non-parametric Mack model that was referred to in the previous section, as the estimators here are minimum variance out of all unbiased estimators, not just out of the linear combinations of the $\hat{f}_j$.

With the definition above, the parameter estimates of the ODP Mack model can be calculated using the maximum likelihood method. However, Strascia and Tripodi [2018] used another method called quasi-likelihood function, which is defined by Wedderburn [1971] by specifying only the relation between mean and variance as follow:

$$\frac{k(X_{i,j}, \phi)}{\mu_{i,j}} = \frac{X_{i,j} - \mu_{i,j}}{V(\mu_{i,j})}, \tag{2.8}$$

or equivalently,

$$k(X_{i,j}, \phi) = \int_{X_{ij}}^{\mu_{ij}} \frac{X_{ij} - s}{\phi V(s)} ds. \tag{2.9}$$

Therefore over all $X_{i,j}$, the quasi-likelihood function can be written as,

$$K(X; \beta, \phi) = \sum_{i+j \leq t} \omega_{ij}\ k(X_{i,j}; \beta, \phi) = \sum_{i+j \leq t} \omega_{ij} \int_{X_{ij}}^{\mu_{ij}} \frac{X_{ij} - s}{\phi V(s)} ds, \tag{2.10}$$

where $\mu_{i,j}$ depends on β – the parameter estimates of this model. ϕ is the dispersion parameter independent from i and j, $\omega_{i,j}$ is the weight on each $K(X_{i,j}; \beta, \phi)$. t is introduced by Strascia and Tripodi [2018] as the balance-sheet year, all the data before t is known with certainty, while the data after is subjected to randomness. Then $V(\mu_{ij}) = h''(h'^{-1}(\mu_{ij}))$ is the so-called variance function of the GLM, and $h = g^{-1}$, where g is the link function. Maximizing this equation can be used to estimate the β parameters, due to the similar properties it has with the likelihood function.

In the ODP model with a logarithmic link-function instead, the relation between mean and variance is the following:

$$\mathbb{E}[X_{i,j+1}|C_{i,j}] = \mu_{ij} = e^{c+a_i+b_j} \qquad and \qquad \mathbb{V}[X_{i,j+1}|C_{i,j}] = \phi V(\mu_{ij}) = \phi\mu_{ij}. \tag{2.11}$$

By inserting these equations into (2.8), the expression of the quasi-likelihood function for the ODP model can be written as:

$$K(X;\beta,\phi) = \sum_{i+j\leq t} \frac{\omega_{ij}}{\phi}\Big[X_{ij}log\frac{\mu_{ij}}{X_{ij}} - \mu_{ij} + X_{ij}\Big], \tag{2.12}$$

the $\hat{\beta}$ estimate is calculated by searching for the $\beta = (c, a_1, \ldots, a_I, b_1, \ldots, b_J)^\top$ values that maximize Function K in (2.10), so that the observed data is the most probable. The optimization problem can be solved through the Gauss-Newton method.

One method often used to estimate the observed data model goodness of fit is to analyze the generalized Pearson residuals. In the ODP case, with $\omega_{ij} = 1$, the residuals are:

$$r_{ij} = \frac{X_{ij} - \hat{\mu}_{ij}}{\sqrt{\hat{\phi}\hat{\mu}_{ij}}}, \tag{2.13}$$

where $\hat{\phi}$ can be calculated through Pearson estimator $\hat{\phi} = \frac{1}{n-p}\sum_{i+j\geq t}\omega_{ij}\frac{(X_{ij}-\hat{\mu}_{ij})^2}{V(\hat{\mu_{ij}})}$, $n-p$ is the number of the model degrees of freedom, n is the number of the observed data, p is the number of parameters to be estimated.

Pearson statistics $\chi^2 = \sum_{i+j\leq t}\omega_{ij}\frac{X_{ij}-\hat{\mu}_{ij}}{V(\hat{\mu}_{ij})}$ is used in Strascia and Tripodi [2018] to calculate the overall discrepancy between empirical and theoretical data. Under the quasi-likelihood case, the deviance is:

$$D(\hat{\mu}, X) = -2\hat{\phi}K(X;\hat{\beta},\hat{\phi}). \tag{2.14}$$

Finally, according to Strascia and Tripodi [2018], for the ODP Mack Model, it is possible to calculate the MSEP for total reserve $\mathcal{R}$ with the following more compact form:

$$\begin{aligned} msep_{X|\mathcal{D}_I}(\hat{\mathcal{R}}) = \hat{\phi} \sum_{i+j>t} \hat{\mu}_{ij} + \sum_{i+j>t} \hat{\mu}_{ij}^2 x_{ij}^\top \hat{\mathbb{V}}[\hat{\beta}] x_{ij} \\ + \sum_{\substack{i_1+j_1>t \\ i_2+j_2>t \\ (i_1,j_1)\neq(i_2,j_2)}} \hat{\mu}_{ij_1} \hat{\mu}_{ij_2} x_{i_1,j_1}^\top \hat{\mathbb{V}}[\hat{\beta}] x_{i_2,j_2}, \end{aligned} \tag{2.15}$$

where $x_{i,j}$ is the dummy variables vector, which is used to code accident and development year.

Another concept is introduced by Strascia and Tripodi [2018] which is the Claims Development Result; it calculates if the claims reserve estimate in year t for accident year i is enough to pay the claims for the next year and new claims reserve at the next year $t+1$:

$$CDR_{i,t+1} = \hat{C}_{i,J}^{(t)} - \hat{C}_{i,J}^{(t+1)}, \tag{2.16}$$

where $\hat{C}_{i,J}^{(t)}$ is the ultimate cost estimate at time t for claims from accident year i. Here t is used as the superscript instead of using m , which was used to represent the different line of business earlier.

Particularly, it is a loss for the insurance company if $CDR_{i,t+1} < 0$, while it is a gain with a positive result.

In the chain ladder framework, the $CDR_{i,t+1}$ is written in the following way:

$$\widehat{CDR}_{i,t+1} = \hat{C}_{i,J}^{(t)} - \hat{C}_{i,J}^{(t+1)} = C_{i,t-i} \Big(\prod_{j=t-i}^{J-1} \hat{f}_j^{(t)} \Big) \Big(1 - \frac{C_{i,t-i+1}/C_{i,t-i}}{\hat{f}_{t-i}^{(t)}} \prod_{j=t-i+1}^{J-1} \frac{\hat{f}_j^{(t+1)}}{\hat{f}_j^{(t)}} \Big). \tag{2.17}$$

By adding a credibility factor $\alpha_j^{(t)} = \frac{C_{t-j,j}}{\sum_{i=1}^{t-j} C_{ij}}$, the ratio between two link ratios associated with the same j development year and estimated in a two-years-in-a-row balance sheet can be written as:

$$\frac{\hat{f}_j^{(t+1)}}{\hat{f}_j^{(t)}} = \alpha_j^{(t)} \frac{C_{t-j,j+1}/C_{t-j,j}}{\hat{f}_j^{(t)}} + (1 - \alpha_j^{(t)}). \tag{2.18}$$

By using (2.16) for the link ratios, the CDR estimator in (2.15) can also be written in the following way:

$$\widehat{CDR}_{i,t+1} = \hat{C}_{i,J}^{(t)} - \hat{C}_{i,J}^{(t)} \frac{C_{i,t-i+1}/C_{i,t-i}}{\hat{f}_{t-i}^{(t)}} \prod_{j=t-i+1}^{J-1} \left(\alpha_j^{(t)} \frac{C_{t-j,j+1}/C_{t-j,j}}{\hat{f}_j^{(t)}} + (1 - \alpha_j^{(t)}) \right). \tag{2.19}$$

Another additive loss reserving model for the multi-year non-life insurance risk was developed by Diers and Linde [2013], which is to elaborate on the risk modelling for non-life insurance companies by modelling insurance risk in a multi-year context. It is recommended to use this formula in case of attritional claims such as homogeneous and stable portfolios. However, for large claims, it should be modelled separately. Hahn [2017] also extended this model onto dependent lines of business by adding weights and covariance matrix.

2.2 Cross-Classified Model

Referred to as EDF cross-classified model (Taylor and McGuire [2016]), it is defined as:

(EDFCC1) The random variables $X_{ij} \in \mathcal{D}_I$ are stochastically independent.

(EDFCC2) For $i = 1, 2, ..., I$, and $j = 1, 2, ...J$

(a) $X_{ij} \sim EDF(\mu_{ij}, \phi_{ij}; a, b, c)$, ϕ_{ij} are dispersion parameters,

(b) $\mathbb{E}[X_{ij}] = \mu_{ij} = \alpha_i \beta_j$ for some parameters $\alpha_i, \beta_j > 0$, and,

(c) $\sum_{j=1}^{J} \beta_j = 1$.

Comparing to the regular Mack model (i.e. non-parametric Mack model), instead of having only f_j as a parameter, this model uses α_i and β_j as parameters for both row and column effects on the expected value of X_{ij}, we can regard α_i as the ultimate claim reserve for accident year i, and β_j as the portion of this total claim reserve amount in each development year j, which is similar to the assumption of the BF algorithm, but without prior information given by experts. It is obvious that regular Mack models apply to cumulative data, whereas cross-classified models apply to incremental data, which implies that the cross-classified model is more general.

If the modified $(EDFCC2a)$, $(EDFCC2b)$ are as in ODP form, which is the sub-family of EDF cross-classified family, along with the further condition $\phi_{ij} = \phi$, then it can be rewritten as

$$X_{ij} \sim ODP(\alpha_i\beta_j, \phi) = ODP(\mu_{ij}, \phi), \tag{2.20}$$

where

$$\mu_{ij} = exp(\ln \alpha_i + \ln \beta_j). \tag{2.21}$$

This modified model is called over-dispersed Poisson (ODP) cross-classified model. Wüthrich [2019] gives more details about this model, where

$$\frac{X_{ij}}{\phi} \sim Poi(\frac{\alpha_i\beta_j}{\phi}), \tag{2.22}$$

and we observe that,

$$\mathbb{E}[X_{ij}] = \alpha_i\beta_j, \tag{2.23}$$

$$\mathbb{V}(X_{ij}) = \phi\alpha_i\beta_j. \tag{2.24}$$

Two side constraints are commonly used in order to make the parameters α_i and β_j uniquely identifiable, which are

$$\alpha_1 = 1 \qquad \text{or} \qquad \sum_{j=1}^{J-2} \beta_j = 1. \tag{2.25}$$

The first option is more convenient in the application of GLM methods, the second option gives an explicit meaning to the pattern $(\beta_1, \beta_2, \ldots, \beta_{J-2})$, namely, that it corresponds to the cash flow pattern.

The best-estimate reserves at time I are given by

$$\hat{X} = \sum_{i+j>I} \mathbb{E}[X_{ij}|\mathcal{D}_I] = \sum_{i+j>I} \alpha_i\beta_j, \tag{2.26}$$

where $\mathcal{D}_I$ holds the same definition as from the previous subsections.

Then the MLE estimates of α_i and β_j are given by

$$\hat{\alpha}_i^{MLE} = \hat{C}_{i,J-1}^{CL} \qquad \text{and} \qquad \hat{\beta}_j^{MLE} = \Big(1 - \frac{1}{\hat{f}_{j-1}^{CL}}\Big) \prod_{k=j}^{J-2} \frac{1}{\hat{f}_k^{CL}}, \tag{2.27}$$

for $i = 1, \ldots, I$ and $j = 1, \ldots, J-1$. Moreover, $\hat{\beta}_0^{MLE} = \prod_{k=0}^{J-2} \frac{1}{\hat{f}_k^{CL}}$.

Since $\hat{C}_{i,J-1}^{CL}$ is the estimator for $\hat{\alpha}_i$, and $\sum_{j=1}^{J-2} \beta_j = 1$, on the overall level, the results should be almost the same as the results of the CL Mack model, but for the prediction of payments for each development year, the results will show some differences.

2.3 Bayesian CL Model

Wüthrich [2019] gives the gamma-gamma Bayesian CL model, which belongs to the exponential dispersion family (EDF) with conjugate priors. The advantage of this model is that the posterior distribution can be calculated analytically, which allows that all quantities of interest be determined in closed form.

Assume that giving fixed constants $\sigma_j > 0$, and $j = 0, ..., J-2$, the Bayesian CL model is defined as:

(a) Conditionally, given $\Theta = (\theta_0, \ldots, \theta_{J-2})$, the $C_{i,j}$ are independent (in i) Markov processes (in j) with conditional distributions,

$$F_{i,j+1} = \frac{C_{i,j+1}}{C_{i,j}} \Big| C_{i,j}, \Theta \sim \Gamma(C_{i,j}\sigma_j^{-2}, \theta_j C_{i,j}\sigma_j^{-2}). \tag{2.28}$$

(b) θ_j are independent and $\Gamma(\gamma_j, f_j(\gamma_j - 1))$-distributed with given prior constants $f_j > 0$, $\gamma_j > 1$.

(c) Θ and $C_{i,0}$ are independent and $C_{i,0} > 0$, $\mathbb{P}$-a.s.

For a given parameter Θ, the conditional mean is given by

$$\mathbb{E}[C_{i,j+1}|C_{i,j}, \Theta_j] = C_{i,j}\mathbb{E}[F_{i,j+1}|C_{i,j}, \Theta_j] = \theta_j^{-1} C_{i,j}. \tag{2.29}$$

From this, it can be found that θ_j^{-1} plays the role of the CL factor f_j here. The reason for choosing the prior parameters of the distribution of θ_j is that

$$\mathbb{E}[\theta_j^{-1}] = \frac{1}{\gamma_j - 1} f_j(\gamma_j - 1) = f_j. \tag{2.30}$$

where the γ_j here is different from the γ from equation (1.16).

Recall that the corresponding probability density function in the shape-rate parametrization for the gamma distribution $\Gamma(\alpha, \beta)$ is

$$f(x; \alpha, \beta) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}, \tag{2.31}$$

where $\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$ is the gamma function, when α is a complex number with a positive real part, while $\Gamma(\alpha) = (\alpha - 1)!$ when α is a positive integer.

Then the joint likelihood function of the observations $\mathcal{D}_I$ and the parameters Θ is given by

$$\begin{aligned} h(\mathcal{D}_I, \Theta) = \prod_{(i,j) \in \mathcal{I}_I, j>1} & \frac{(\frac{\theta_{j-1} C_{i,j-1}}{\sigma_{j-1}^2})^{\frac{C_{i,j-1}}{\sigma_{j-1}^2}}}{\Gamma(\frac{C_{i,j-1}}{\sigma_{j-1}^2})} F_{i,j}^{\frac{C_{i,j-1}}{\sigma_{j-1}^2} - 1} exp\{-\frac{\theta_{j-1} C_{i,j-1}}{\sigma_{j-1}^2} F_{i,j}\} \\ & \times g(C_{1,0}, ..., C_{I,0}) \prod_{0}^{J-2} \frac{(f_j(\gamma_j - 1))^{\gamma_j}}{\Gamma(\gamma_j)} \theta_j^{\gamma_j - 1} exp\{-\theta_j f_j(\gamma_j - 1)\}, \end{aligned} \tag{2.32}$$

where $\mathcal{I}_I$ is the index set of observations $\mathcal{D}_I$, and $g(C_{1,0}, ..., C_{I,0})$ denotes the density of the first column $j = 0$.

This allows to apply Bayes rule which provides the posterior of Θ, conditionally given $\mathcal{D}_I$,

$$h(\Theta | \mathcal{D}_I) \propto \prod_{j=0}^{J-2} \theta_j^{\gamma_j + \sum_{i=1}^{I-j-1} \frac{C_{i,j}}{\sigma_j^2} - 1} e^{-\theta_j \left[f_j(\gamma_j - 1) + \sum_{i=1}^{I-j-1} \frac{C_{i,j+1}}{\sigma_j^2}\right]} \tag{2.33}$$

which in turn proves that

$$\Theta | \mathcal{D}_I \sim \Gamma\left(\gamma_j + \sum_{i=1}^{I-j-1} \frac{C_{i,j}}{\sigma_j^2}, f_j(\gamma_j - 1) + \sum_{i=1}^{I-j-1} \frac{C_{i,j+1}}{\sigma_j^2}\right). \tag{2.34}$$

Under the assumptions of a gamma-gamma Bayesian CL model, the posterior Bayesian CL factors are given by

$$\hat{f}_j^{BCL} = \mathbb{E}[\theta_j^{-1}|\mathcal{D}_I] = \alpha_j \hat{f}_j^{CL} + (1-\alpha_j) f_j, \tag{2.35}$$

with CL factor estimate $\hat{f}_j^{CL}$ given by (1.3) and credibility weight

$$\alpha_j = \frac{\sum_{i=1}^{I-J-1} C_{i,j}}{\sum_{i=1}^{I-J-1} C_{i,j} + \sigma_j^2(\gamma_j - 1)} \in (0,1). \tag{2.36}$$

Therefore, the prediction for the accumulative payment for accident year i with $i+J-1 > I$ is

$$\hat{C}_{i,J-1}^{BCL} = \mathbb{E}[C_{i,J-1}|\mathcal{D}_I] = C_{i,I-i} \prod_{j=I-i}^{J-2} \hat{f}_j^{BCL}. \tag{2.37}$$

Consider the gamma-gamma Bayesian CL model with a non-informative prior $\gamma_j \to 1$, it is obvious that $\alpha_j = 1$ in this case, and therefore $\hat{f}_j^{BCL} = \hat{f}_j^{CL}$ and $\hat{C}_{i,J-1}^{BCL} = \hat{C}_{i,J-1}^{CL}$, which implies that the CL model is a special case of the Bayesian CL model.

For the conditional MSEP obtained earlier,

$$\begin{aligned} msep_{C_{i,J-1}|\mathcal{D}_I}(\hat{C}_{i,J-1}^{BCL}) &= \mathbb{V}(C_{i,J-1}|\mathcal{D}_I) + (\mathbb{E}[C_{i,J-1}|\mathcal{D}_I] - \hat{C}_{i,J-1}^{BCL})^2 \\ &= \mathbb{V}(C_{i,J-1}|\mathcal{D}_I). \end{aligned} \tag{2.38}$$

This shows the optimality of the Bayesian CL predictor within the model, and that what remains is the calculation of the conditional variance of the final claim.

Wüthrich [2019] gives the solution for conditional MSEP as:

$$\begin{aligned} &msep_{C_{i,J-1}|\mathcal{D}_I}(\hat{C}_{i,J-1}^{BCL}) = \hat{C}_{i,J-1}^{BCL}\Gamma_{I-i} + (\hat{C}_{i,J-1}^{BCL})^2\Delta_{I-i}, \\ &msep_{C_{i,J-1}|\mathcal{D}_I}(\sum_i \hat{C}_{i,J-1}^{BCL}) = \sum_i msep_{C_{i,J-1}|\mathcal{D}_I}(\hat{C}_{i,J-1}^{BCL}) + 2\sum_{i<l} \hat{C}_{i,J-1}^{BCL}\hat{C}_{l,J-1}^{BCL}\Delta_{I-i}, \end{aligned} \tag{2.39}$$

where Γ_k and Δ_k are defined as

$$\begin{aligned}\Gamma_k &= \sum_{j=k}^{J-2} \sigma_j^2 \prod_{n=j}^{J-2} \Big(\hat{f}_n^{BCL} \frac{\sigma_n^2(\gamma_n - 1) + \sum_{i=1}^{I-n-1} C_{i,n}}{\sigma_n^2(\gamma_n - 2) + \sum_{i=1}^{I-n-1} C_{i,n}} \Big), \\ \Delta_k &= \prod_{j=k}^{J-2} \Big(\frac{\sigma_n^2(\gamma_n - 1) + \sum_{i=1}^{I-n-1} C_{i,n}}{\sigma_n^2(\gamma_n - 2) + \sum_{i=1}^{I-n-1} C_{i,n}} \Big) - 1.\end{aligned} \tag{2.40}$$

Wüthrich [2019] also discovered that in the non-informative prior case, $\gamma_j \to 1$, the approximation of $msep_{C_{i,J-1}|\mathcal{D}_I}$ with non-informative priors and Mack's formula are very close. For many typical non-life insurance data sets, this observation still holds true and it suggests the use of the simpler formula.

There are many other stochastic methods, however they still share same pitfalls when applied to actual data. Therefore, Hartl [2014] provides a way to improve the performance of triangle GLMs using splitlinear rescaling and parametric resampling with a limited Pareto distribution. Meyers [2016] describes a method to fit a bivariate stochastic model that captures the dependencies between the two lines of insurance, given a Bayesian Markov chain Monte Carlo (MCMC) stochastic loss reserve model. Also, Korn [2015] gives some strategies on loss development modelling with curve fitting, credibility, and layer adjustments. Lally and Hartman [2018] applied Gaussian Process (GP) regression with input warping and several covariance functions for loss reserving prediction, which requires little input from the modeller.

Mulquiney [2006] applied artificial neural networks on loss reserves triangle data. Furtherer more, Jamal [2018] applied more machine learning methods on loss reserves triangle data, including random forest, neural network, gradient boosting machine and a boosted Tweedie compound poisson model.

Chapter 3

Individual Claim Reserving

The previous methods described above all focus on using the aggregate data triangle for loss reserving, assuming that insureds are independent. However, in real life, they are not necessarily independent, and there are different types of dependency between the claims of different insured. With the computational power we have now, it would be more accurate to predict the claim reserve of each insured separately without losing the dependence between claims, and then aggregate the claim reserves to produce the aggregate claim reserves prediction.

So as not to confuse it with aggregate claim reserving methods, here each individual claim payment is denoted as $x_{i,j}^{m}$, where m is the index of each insured, $m \in \mathbb{M} = \{1, 2, \ldots, M\}$, and i is the accident year, while j is the development year. Here each claim must have a fixed accident year, therefore the combinations of m and i are prescient.

Naturally, the aggregate claim reserve $X_{i,j}$ can be written as:

$$X_{i,j} = \sum_{m \in \Phi(i)} x_{i,j}^{m}, \tag{3.1}$$

where $\Phi(i) = \{m_i^{(1)}, m_i^{(2)}, ..., m_i^{(n)}, ..., m_i^{N_i}\}$ is the set of claims incurred in accident year i, $m_i^{(n)} \in \mathbb{M}$, $n = 1, 2, ..., N_i$, where N_i is the maximum number of claims in accident year i. So the cumulative claim reserve for the m-th insured over all development years would be written as

$$c_{i,j}^m = \sum_{k=1}^{j} x_{i,k}^m. \tag{3.2}$$

Then the cumulative aggregate claim reserve can be written as:

$$C_{i,j} = \sum_{k=1}^{j} X_{i,k} = \sum_{k=1}^{j} \sum_{m\in\Phi(i)} x_{i,k}^m = \sum_{m\in\Phi(i)} c_{i,j}^m. \tag{3.3}$$

Here our ultimate goal is to find a model that can predict each unknown $x_{i,j}^m$ in the lower triangle, knowing that here the total reserve for the most recent year can be calculated by

$$\mathcal{R} = \sum_{i=1}^{I} X_{i,J-i+1} = \sum_{i=1}^{I} \sum_{m\in\Phi(i)} x_{i,J-i+1}^m. \tag{3.4}$$

3.1 GLM Individual Claim Reserving Methods

In some companies, actuaries have already started using GLM methods to predict individual claim loss reserving.

Taylor et al. [2008] provides various forms of individual claim reserving models, showing how these methods perform more efficiently than aggregate models. To establish an individual model, let y^m be the response of interest for the m-th claim, and use v^m instead of X_m (to avoid being confused with aggregate reserve data $X_{i,j}$) to be the covariate vector for the m-th claim. It has the following basic form:

$$y^m \sim F(\cdot; v^m, \beta), \tag{3.5}$$

where F is some specific distribution function, β is a vector of parameters which are independent of the claims and needs to be estimated.

Denote

$$g^m = \mathbb{E}[y^m], \tag{3.6}$$

where, more explicitly

$$g^m = g(v^m, \beta), \tag{3.7}$$

for some unique function g. v^m can also be decomposed into three general parts as $v^{m(S)}$, $v^{m(T)}$, and $v^{m(U)}$ that stand for static, time, and unpredictable covariate vectors.

For the individual claim models that are only dependent on time covariates, there is a more obvious way to separate the time covariates into different components, namely

a^m = accident period;

d^m = development period when the claim finalized;

p^m = $a^m + d^m$ = experience period in which the claim finalization occurs;

t^m = operational time, which maps development time δ to an interval [0,1],

where in each case the superscript m indicates that the value of the time variable concerned is that observed for the m-th claim.

Here we changed the notation for operational time t as $t^m(\delta) = \sum_{i=1}^{N} \frac{I\{\tau_i \leq \delta\}}{N}$, where I is an indicator function, τ_i is the finalization time for claim i, and this definition gives the proportion of incurred claims that are finalized at or before that time (δ), for a given but arbitrary accident year, with N claims incurred.

Taylor et al. [2008] also explains that the model depends only on a^m and t^m, or else time related quantities other than a^m and t^m, do not require statistical case estimation (in short, statistical case estimation consists of estimating the ultimate cost of each claim). At the same time, depending on unpredictable covariates will make the model more complicated and will require to average the forecast; Therefore, the best model here would be one depending only on time and static covariates, as its forecast can be put into statistical case estimate form.

For a single accident period a, the model used to predict the m-th claim's loss reserve would now be:

$$y^{*m} = \int g(a, t^m(p^m), p^m, v^{*m(S)}; \hat{\beta}) dP(p^m | v^{*m(S)}), \tag{3.8}$$

where

- the * in y^{*m} represents that its value is a forecast;
- the integral is only over p^m because a is a single period, t^m depends on p^m only, and $v^{*m(S)}$ is static;
- $g(\cdot)$ is the statistical case estimate in respect of the m-th unfinalised claim, conditional on a, t^m , p^m , $v^{*m(S)}$;
- the $*$ in $v^{*m(S)}$ means that its response will only be observable in the future;
- t^m is the mapping of real-time p^m to operational time;
- and the measure $P(\cdot)$ on p^m may now depend on $v^{*m(S)}$.

This is equivalent to the GLM form which is usually known as: $g(y^{*m}) = v^{mT}\hat{\beta}$, where $g(\cdot)$ is the link function, $\hat{\beta}$ is a vector of estimated parameters, and the upper T denotes vector or matrix transposition, and here v^m can be decomposed into (a, t^m, p^m, $v^{*m(S)}$).

Actuaries see the models discussed above as "paid" models, for the reason that they only depend on paid losses. The insurer's various estimates of these losses through the lifetimes of the claims are not taken into consideration at all.

Another model which is usually referred to as "incurreds" model is a conventional alternative form of model forecasts for ultimate losses on the basis of the insurer's estimates at the valuation date. A statistical case estimation form of this is also considered in Taylor et al. [2008].

In the following, the term case estimate will be used to mean a subjective estimate of the ultimate incurred loss placed by an insurer on an individual claim. The estimate will usually be made by a claims assessor and is often referred to as a real estimate or manual estimate, where each claim involves a whole sequence of case estimates through its lifetime.

The factor relating case estimates and the ultimate incurred cost is considered as the general thrust of a model, and it is usually represented by a ratio referred to as the age-to-ultimate ratio, defined as:

$$R : \text{Age-to-ultimate ratio} = \frac{U : \text{Expected ultimate incurred loss}}{I : \text{Current case estimate}}, \tag{3.9}$$

where the "Expected" in the numerator is used in its statistical sense. A model like this will be referred to as a case estimates model. It corresponds to the conventional "incurreds" model.

However, the possibility of a zero numerator or denominator in (3.7) complicates the model, with the result that the model needs to consist of some sub-models, where I denotes the current case estimate, U denotes the ultimate incurred loss, therefore $R = U/I$. Here $F(.)$ denotes a distribution function.

More specifically, three sub-models are required:

- Sub-Model 1: $\mathbb{P}[U = 0]$,
- Sub-Model 2: $F(U|I = 0, U > 0)$,
- Sub-Model 3: $F(R|I > 0, U > 0)$.

Because in the case $I = 0$, the ratio R does not exist, and so the size of U must be modeled directly, rather than as $U = I \times R$, cases $I = 0$ and $I \neq 0$ are dealt with separately, as are the cases $U = 0$ and $U \neq 0$. Moreover, the distribution of U, with a discrete mass at $U = 0$, is best modeled by recognizing separately the mass and the remainder of the distribution (assumed continuous).

Under this model, the statistical case estimate for U is:

$$\mathbb{E}[U] = \mathbb{E}[U|U \neq 0]\mathbb{P}[U \neq 0]. \tag{3.10}$$

Consider a claim from accident period a, reported in development period r, and finalized in development period f, which is not the same as CL factor. It will carry a case estimate I_d at the end of development period $d = r, r+1, ..., f-1$, and then the ultimate cost U. This implies a total of $f - r$ records as shown below:

$$I_r \xrightarrow{\quad I_{r+1} \qquad I_{r+2} \qquad \ldots \qquad I_{f-2} \qquad I_{f-1} \quad} U$$

However, this would create difficulties of two types. First, the case estimates would become unpredictable dynamic covariates. Second, any feasible means of forecasting future case estimate development would be likely to involve a highly dubious Markov assumption as the future evolution only depends on the current state, and not taking the information from previous states.

It seems preferable to create observations of case estimate development, each taken over some periods from the end of some development period d $(= r, r+1, \dots, f-1)$ to finalization in development period f. Thus, all observation periods would end at finalization and would be age-to-ultimate, which has a structure as shown below:

$$\begin{array}{r}
I_r \xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad} U \\
I_{r+1} \xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad} U \\
I_{r+2} \xrightarrow{\qquad\qquad\qquad\qquad\qquad\qquad\qquad} U \\
\vdots \\
\vdots \\
I_{f-2} \xrightarrow{\qquad\qquad\qquad\quad} U \\
I_{f-1} \xrightarrow{\quad} U
\end{array}$$

Figure 3.1: Structure of Case Estimate Development Observations

It would require the replacement of GLMs by Generalized Estimating Equations (GEEs) if a dependency structure is added on observations, but this was not done by Taylor et al. [2008]. Instead, one record from the multi-period observations corresponding to each claim has been sampled at random. That is to say, an integer is selected randomly from the set $(d+u, d+u+1, \dots, f-1)$ where $d+u$ is the least value for which $I_{d+u} > 0$. If this integer is denoted $d+v$, then the value $R = U/I_{d+v}$ is taken as the observation to be modelled, as Sub-Model 2. A GLM may then be applied to these records since it removes the dependency while retaining a selection of records over different period lengths.

Then it would be possible to model the finalization rate (Sub-Model 3), which is defined as the number of claims finalized in the period divided by some measure of exposure to finalization, such as the average number open over the period. Therefore, future operational times may be derived and thus the finalization of claims corresponds to the advancement of operational time.

Taylor et al. [2008] provide a simple model of future claim finalisation with the form:

$$\Delta t_i(d) = t_i(d) - t_i(d-1) = \omega_i \delta_i(d), \tag{3.11}$$

where $t_i(d)$ denotes the operational time at the end of development period d of origin period i, and $\Delta t_i(d)$ denotes the increment in operational time over that development period. On the right side of equation (3.11), $\delta_i(d)$ is a selected increment, specific to the origin period, and ω_i is an adjustment factor which is usually close to 1, so that

$$t_i(\infty) = t_i(d^*) + \sum_{d=d^*+1}^{\infty} \Delta t_i(d) = t_i(d^*) + \omega_i \sum_{d=d^*+1}^{\infty} \delta_i(d) = 1, \tag{3.12}$$

where d^* is the value of d (for origin period i) at the valuation date.

According to Taylor et al. [2008], it is preferable to model such probabilities for each claim as dependent on the attributes of that claim. This is most naturally done by means of survival analysis. This means that, for each i, the i-th claim, with vector X_i of covariates, has a lifetime T_i, from reporting to finalization, assumed subject to a survival function $S(.)$ such that

$$\text{Prob}[T_i > t] = S_i(t; X_i). \tag{3.13}$$

The hazard rate associated with the i-th claim is $h(t) = -S'(t)/S(t)$. A convenient form for the present application is the proportional hazards form

$$h_i(t) = \exp[X_i^T \beta], \tag{3.14}$$

where β is a vector of parameters and the upper T denotes matrix transposition. This will be used to model the development patterns for both “paids” and “Outstandings” models in Chapter 4 with different parameters.

According to Taylor et al. [2008], it is often possible to achieve high efficiency with a model of the “paids” type that has a small number of parameters, certainly fewer than in most conventional actuarial models. The issues involved in the construction of such a model are relatively simple. Further improvements are possible, but possibly with considerable effort.

Even though in Taylor et al. [2008], the loss reserve forecast conditioned by case estimates did not, in itself, improve predictive efficiency, it did provide an alternative model that was largely stochastically independent of the paids model. The two models (“paids” and “incurreds”) could then be used to produce a blended estimate of higher efficiency than either one. Later this idea was applied by Harej et al. [2017] to simulate a data set. More details will be discussed in Chapter 4.

Another proposal by Zhao et al. [2009] with GLMs, suggests a model with a semi-parametric structure that can be used to fit the individual claims reserving with more flexibility.

3.2 Neural Networks With Cascading Method

Harej et al. [2017] give an example to predict the loss reserving of individual claims with artificial neural networks (ANNs) (which is popularly known as neural networks), more specifically with a multilayer perceptron (MLP). The triangular cascading architecture can be used to predict the lower triangle $\mathcal{D}_I^c$ in a step-by-step way. The data sets are simulated by the unified paids and incurreds model introduced in Taylor et al. [2008]. Cascading architectures will be explained in Figure 3.3.

There are three major parts to MLP: the input layer, the hidden layer, and the output layer. The input layer which can also be called input neurons contains some features that can be used for training and prediction purposes. Then the features are assigned with weights (ω) and are projected into the hidden layer with a combination function as hidden neurons. Then with another function, which is called activation function (σ), the hidden neurons are being projected to the output layer as output neurons. The reason why MLP is categorized as supervised learning is in the sense that it requires an output as the response to complete the model.

For activation functions, Harej et al. [2017] chose *Sigmoid* and *Hyperbolic tangent* activation functions because they fit well with the backpropagation algorithm in a way that they are continuous and differentiable:

$$\textbf{Sigmoid function: } \sigma(x) = \frac{1}{1+e^{-x}}, \qquad x \in \mathbb{R},$$

$$\textbf{Hyperbolic tangent: } \sigma(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}, \qquad x \in \mathbb{R}.$$

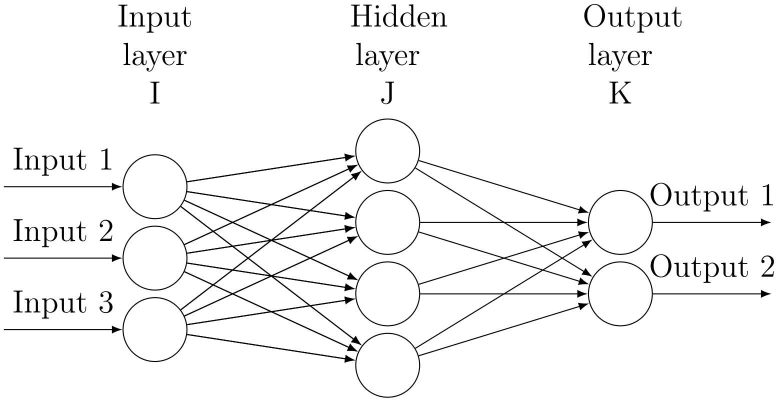

Figure 3.2: Graphical Representation of MLP with one Hidden Layer

Then in order to measure the quality of the model, mean square error (MSE) is used through the cost function:

$$C(\omega) = \sum_{\alpha} (y_\alpha(\omega) - t_\alpha)^2, \tag{3.15}$$

where ω is the weight, t_α is the α-th response in the output layer, and y_α is the predicted output for the α-th response in the output layer, with $y_\alpha = \sigma(z) = \sigma(\omega x)$.

Hence the model with the smallest MSE is, in relative terms, the best model. There are other possible cost functions (e.g., the AIC or BIC criteria) that could also be considered.

Harej et al. [2017] explain how to complete the triangle, step by step, for individual claims from different accident years; which is the method mentioned earlier, called cascading. The graphical representation of the cascading method is shown below. The steps are shown as the white colour numbers.

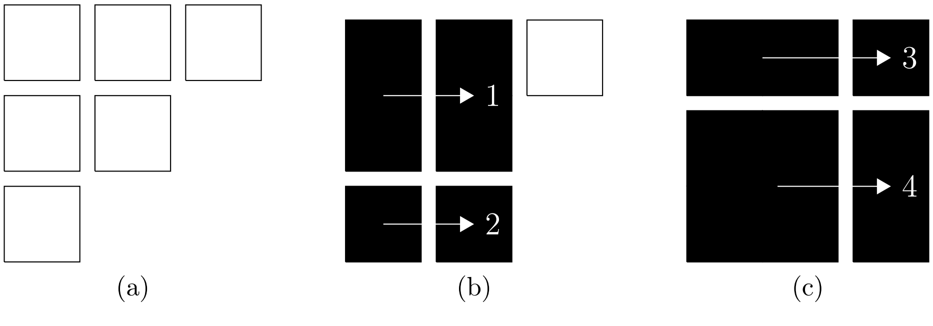

Figure 3.3: Graphical Representation of Cascading (Type 1)

It uses the most information from both previous accident years and future accident years to predict the claim reserve from the bottom left corner to the top right of the triangle by repeating Steps 1 to 4 as shown in black blocks from Figure 3.3, which are

Step 1 (Train): Train the model using the data from top left as predictors, and the data from top right as responses.

Step 2 (Predict): Use the trained model to predict the bottom right using the predictors from bottom left.

Step 3 (Train): Train the model again using the predictors from top left, and the data from top right as responses.

Step 4 (Predict): Use the trained model to predict the bottom right using the variables from bottom left.

However, Harej et al. [2017] mention a weakness using the cascading method, due to the similarity with the CL algorithm, it performs poorly on the data that contains claims with different developments patterns.

In order to compute the parameters of the ANN model, two learning algorithms were introduced by Harej et al. [2017]; One is a standard backpropagation algorithm (BA) algorithm, which is a first order learning method. A simple explanation will follow.

In order to minimize the cost function (3.13), the partial differential equations $\frac{\partial C(\omega)}{\partial \omega}$ for both output and hidden layers need to be calculated. A *sigmoid function* will be used here for the explanation because the differential equation is simpler to show. The differential equation of the cost function can be written as:

$$C'(\omega) = C(\omega)(1 - C(\omega)). \tag{3.16}$$

As shown in Figure 3.2, I, J, K are the sets of nodes from the input, hidden, and output layers; w_{ij} are defined as the weights going from the input layer to the hidden layer, and w_{jk} are defined as the weights from the hidden layer to the output layer.

For an output layer, y_j is the node from the hidden layer and y_k is the node from the output layer, then we have

$$\frac{\partial C(\omega)}{\partial \omega_{jk}} = y_j \delta_k, \tag{3.17}$$

where

$$\delta_k = y_k(1 - y_k)(y_k - t_k). \tag{3.18}$$

For a hidden layer, y_i is the node from the input layer and y_k is the node from the hidden layer, then we have

$$\frac{\partial C(\omega)}{\partial \omega_{ij}} = y_i \delta_j, \tag{3.19}$$

where

$$\delta_j = y_j(1 - y_j) \sum_{k \in K} \delta_k \omega_{jk}. \tag{3.20}$$

If the bias term θ were considered, it would be easy to find that:

$$\frac{\partial y}{\partial \theta} = 1. \tag{3.21}$$

Hence, the backpropagation algorithm is based on the following steps:

1. Run the neural network forward with input data to get the network output t_k;

2. For each output node, compute

$$\delta_k = y_k(1 - y_k)(y_k - t_k). \tag{3.22}$$

3. For each hidden node, calculate

$$\delta_j = y_j(1 - y_j) \sum_{k \in K} \delta_k \omega_{jk}. \tag{3.23}$$

4. Update each weight and bias as follows:

$$\omega_{\ell-1,\ell} = \omega_{\ell-1,\ell} + \Delta W = \omega_{\ell-1,\ell} - \eta \delta_\ell y_{\ell-1}, \tag{3.24}$$

$$\theta = \theta + \Delta\theta = \theta - \eta \delta_\ell, \tag{3.25}$$

where W represents both ω_{ij} and ω_{jk}, η is the learning rate, and ℓ represents the index of the layers (e.g j or k): if $\ell = k$, then $\ell - 1 = j$; if $\ell = j$, then $\ell - 1 = i$.

Repeating this process for some times, the optimized weight and bias can be found when the cost function is minimal. Since there might be a local minimum, the number of repetitions should not be too small. Meanwhile, a large number of repetitions can be time-consuming. Therefore, 500 epochs are the standard number of times that is used frequently.

The other learning algorithm is called scaled conjugate gradient backpropagation (SCG) algorithm (see Møller [1993]), which is a second-order learning method. A quadratic approximation of the cost function in the neighborhood of the point concerned was used, which can be computed by the first three elements of the Taylor expansion of the cost function,

$$C_{SCG}(\omega + \Delta\omega) \approx C(\omega) + C'(\omega)\Delta\omega + \frac{1}{2}C''(\omega)\Delta\omega. \tag{3.26}$$

In order to minimize C_{SCG}, the critical points need to be found where

$$C'_{SCG}(\Delta\omega) = C''(\omega)\Delta\omega + C'(\omega) = 0. \tag{3.27}$$

The final algorithm for SCG was given by Møller [1993], which has the following 9 steps:

1. Choose weight vector ω_1 and scalars $\sigma > 0$, $\lambda_1 > 0$, and $\bar{\lambda}_1 = 0$:

   Set $p_1 = r_1 = C'(\omega_1)$, $k = 1$ and success $=$ true.

2. If success $=$ true, then calculate second order information:

$$\sigma_k = \frac{\sigma}{|p_k|}, \tag{3.28}$$

$$s_k = \frac{C'(\omega_k + \sigma_k p_k) - C'(\omega_k)}{\sigma_k}, \tag{3.29}$$

$$\delta_k = p_k^T s_k. \tag{3.30}$$

3. Scale s_k:

$$s_k = s_k + (\lambda_k - \bar{\lambda}_k)p_k, \tag{3.31}$$

$$\delta_k = \delta_k + (\lambda_k - \bar{\lambda}_k)|p_k|^2. \tag{3.32}$$

4. If $\delta_k \leq 0$ then make the Hessian matrix positive definite:

$$s_k = s_k + (\lambda_k - 2\frac{\delta_k}{|p_k|^2})p_k, \tag{3.33}$$

$$\bar{\lambda}_k = 2(\lambda_k - \frac{\delta_k}{|p_k|^2}), \tag{3.34}$$

$$\delta_k = -\delta_k + \lambda_k |p_k|^2, \lambda_k = \bar{\lambda}_k. \tag{3.35}$$

5. Calculate the step size:

$$\mu_k = p_k^T r_k, \ \ \alpha_k = \frac{\mu_k}{\delta_k}. \tag{3.36}$$

6. Calculate the comparison parameter:

$$\Delta_k = \frac{2\delta[C(\omega_k) - C(\omega_k + \alpha_k p_k)]}{\mu_k^2}. \tag{3.37}$$

7. If $\Delta_k \geq 0$, then a successful reduction in error can be attained as:

$$\omega_{k+1} = \omega_k + \alpha_k p_k, \tag{3.38}$$

$$r_{k+1} = C'(\omega_{k+1}), \tag{3.39}$$

$$\lambda_k = 0, \ \ success = true. \tag{3.40}$$

7a. If $k \bmod N = 0$ then restart the algorithm: $p_{k+1} = r_{k+1} + \beta_k p_k$,

else create a new conjugate direction:

$$\beta_k = \frac{|r_{k+1}^2 - r_{k+1} r_k|}{\mu_k}, \tag{3.41}$$

$$p_{k+1} = r_{k+1} + \beta_k p_k. \tag{3.42}$$

7b. If $\Delta_k \geq 0.75$ then reduce the scale parameter: $\lambda_k = \frac{1}{2}\lambda_k$,

else a reduction in error is not possible: $\lambda_k = 4\lambda_k$, success $=$ false.

8. If $\Delta_k < 0.25$ then increase the scale parameter: $\lambda_k = 4\lambda_k$.

9. If the steepest descent direction $r_k \neq 0$ then set $k = k + 1$ and go to Step 2,

else terminate and return ω_{k+1} as the desired minimum.

Comparing SCG with the BA algorithm, we see that SCG outperforms BA because it does not need to adjust the learning rate continually. The solution suggested by this SCG algorithm is to reduce the number of iterations by looking for an optimal direction of descent.

From the result of Harej et al. [2017], it seems that an individual development of claims with an ANN cascading method might have a better performance on long-tailed claims.

Typically, ANNs predict better if paid and outstanding claims are used as an input, also ANNs sometimes predict better if input data are modified to ratios, but this is not always the case.

If a line of business is not homogeneous, ANN might differentiate claims with statistically different underlying patterns. More generally, CL may underperform when data is not well dealt by the specified model (data does not satisfy the model assumptions).

3.3 Random Forests With Modified Cascading Method

Random Forest (RF) is a method that combines the decision trees method with bagging and bootstrapping methods (See Hastie et al. [2017]). The advantage of using Random Forest is that it can prevent overfitting by averaging several decision trees and reduce the chance of stumbling across a classifier that does not perform well because of the relationship between the training data and testing data. Three major methods in RF will be explained in this section.

Decision trees, as the name suggests, involve dividing the features space into some regions with a tree diagram (see Figure 3.4). Performing predictions with a tree involves the following two main steps:

1. Divide the predictor space into J distinct and non-overlapping regions $R_1, ..., R_J$.

2. For every observation that falls into the region R_j, $j \in \{1, \ldots, J\}$ make the same prediction.

Normally, the mean (for a regression problem) or mode (for a classification problem, most common class) of the training features in each region is used to make predictions.

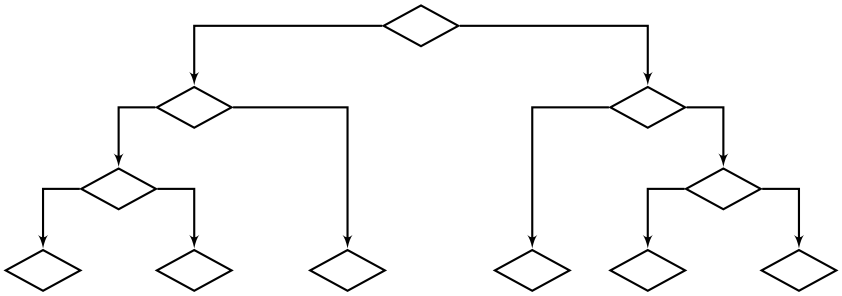

Figure 3.4: Decision Tree Graphical Representation

Applying this method to the loss reserving case falls into the regression problem, in order to find the best model, the prediction sum of square error (RSS) can be used as an objective function to assess the tree predictions quality:

$$\text{RSS} = \sum_{j=1}^{J} \sum_{i:x_i \in R_j} (y_i - \hat{y}_{R_j})^2, \tag{3.43}$$

where $\hat{y}_{R_j}$ is the predicted value when predictors lie in R_j.

As it is computationally unfeasible to consider every possible partition of the feature space into J boxes, therefore, some other approaches are required. Generally, recursive binary splitting is taken as an approach to solving the issue.

There are two features in recursive binary splitting:

1. Top-down: it begins at the top of the tree where all observations belong to a single region. Then it successively splits the predictor space.

2. Greedy: at each step, the best split is done. Does not look ahead and picks a split that will lead to a better tree in some future step.

The first step of recursive binary splitting is to select the predictor X_j and the cut point s such that the predictor space can be split into the two regions

$$R_1(j,s) := \{X|X_j < s\} \quad \text{and} \quad R_2(j,s) := \{X|X_j \geq s\},$$

which leads to the greatest possible reduction in RSS, i.e. minimizes

$$\sum_{i:x_i\in R_1(j,s)} (y_i - \hat{y}_{R_1(j,s)})^2 + \sum_{i:x_i\in R_2(j,s)} (y_i - \hat{y}_{R_2(j,s)})^2$$

across values of j and s.

This splitting process is repeated at each step by subdividing all of the subregions obtained from the previous step until a stopping criterion is reached. In the absence of a stopping criterion, the final tree would have n regions with one observation in each, which is an overfit. Even with the stopping criterion, the resulting tree is likely to produce overfitting.

Decision trees are easy to interpret but do not have the same level of prediction accuracy as some other methods, and it is not robust, because a slight change in the dataset can build a dramatically different tree. A smaller tree with fewer splits might lead to lower variance and better interpretation at the cost of a little bias. However, a seemingly worthless split early on in the tree might be followed by a split that leads to a significant reduction in RSS later on.

In order to improve this, other methods can be added upon decision trees, one of them is an approach called cost-complexity pruning, which is similar to ridge and lasso regressions as it also involves a tuning parameter α.

Cost complexity pruning involves identifying the subtree $T \subseteq T_0$ which minimizes

$$\sum_{m=1}^{|T|} \sum_{i:x_i\in Rm} (y_i - \hat{y}_{R_m})^2 + \alpha|T|,$$

where α is a tuning parameter, which can be selected through cross-validation. $|T|$ is the number of leaves (i.e. of subregions) of the subtree.

Baudry and Robert [2019] proposed another method called ExtraTrees algorithm based on this decision tree method. This algorithm builds an ensemble of unpruned regression trees with the traditional cascading method. The predictions of the trees are aggregated to yield the final prediction by a majority vote in classification problems and arithmetic average in regression problems. The main differences of this method compared to other tree-based ensemble methods are:

1. It splits nodes by choosing cut points fully at random

2. It uses the whole learning sample (rather than bootstrapping) to grow the trees

Another approach is called Bagging, which is a general procedure for reducing the variance of a learning method. By adding this approach into Random Forest, it involves the following steps:

1. Obtain N different training sets, (which requires bootstrapping, explained below).

2. Build a decision tree for each training set.

3. The final prediction for observation is an average of predictions from a large number N of decision trees:

$$\hat{f}_{avg}(x) = \frac{1}{N} \sum_{n=1}^{N} \hat{f}_n(x),$$

where $\hat{f}_n$ is the prediction at the n-th decision tree, x is the variables.

Since predictions from all B models are imperfectly correlated, the variance of the final prediction will be reduced:

$$\begin{aligned}
\mathbb{V}[\hat{f}_{avg}(x)] &= \frac{1}{N^2}\mathbb{V}[\sum_{n=1}^{N} \hat{f}_n(x)] \\
&= \frac{1}{N^2} \sum_{n_1=1}^{N} \sum_{n_2=1}^{N} Cov[\hat{f}_{n_1}(x); \hat{f}_{n_2}(x)] \\
&< \frac{1}{N^2} N^2 \mathbb{V}[\hat{f}_{n_i}(x)],
\end{aligned}$$

since when $n_1 \neq n_2$,

$$Cov[\hat{f}_{n_1}(x); \hat{f}_{n_2}(x)] < 1.$$

In practice, it is complicated to have N different training sets. Furthermore, splitting the training set into N subsets might generate training subsets that are too small. Hence another approach mentioned above was proposed – bootstrapping, where sample with replacement n times from the original training set was drawn to create the $n-th$ training set. This method

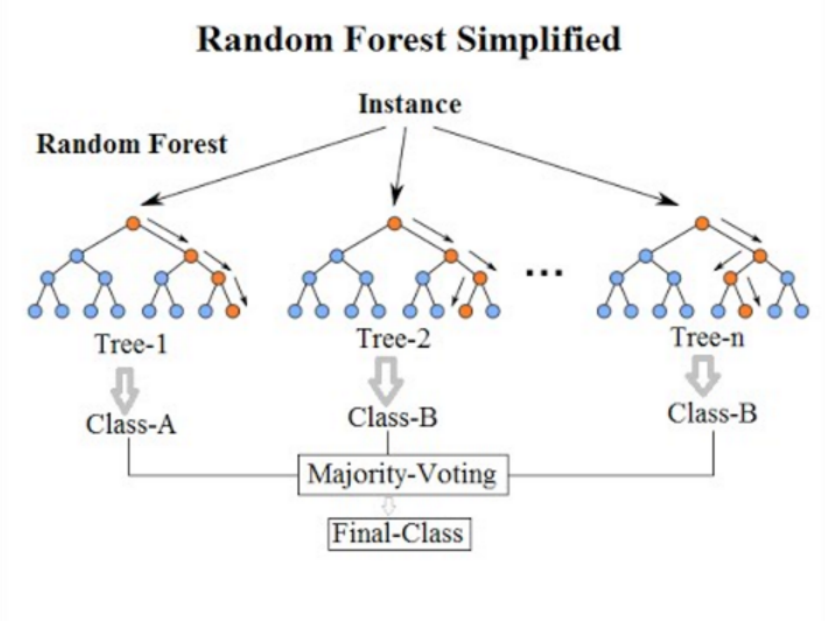

Figure 3.5: Random Forest Graphical Representation
Source: https://commons.wikimedia.org/wiki/File:Random_forest_diagram_complete.png

can improve the accuracy of prediction dramatically, and it can handle missing data easily. Unfortunately, it makes it difficult to interpret the resulting model.

The RF diagram shown in Figure 3.5 falls into the case for the classification problem. In order to predict the loss reserving amount, we need to apply the regression case of RF, which is also called *regression forest*. Instead of predicting class A or B at the end of each tree, it uses the average value of responses that fall into the same region and can be written as

$$\text{Prediction} = \frac{\sum_{\text{m in } \{i,j\} \text{ that fall into the same region}} \hat{x}^m_{i,j}}{\text{\# of m in } \{i,j\} \text{ that fall into the same region}} \tag{3.44}$$

as in the individual claims loss reserving case.

Due to the nature of the cascading method, the predictions that it made are based on both past and post information. Here I would like to propose a modified cascading method, which predicts the individual loss reserving following the timeline. We use only the past historical data of one accident year claims to predict the recent year payment, then apply this trend to the next accident year, to find the trend for the next year. In this way, for each accident year, the prediction will only be based on the past data, instead of "future data". For each accident year, we would have hundreds, even thousands of claims, which is plausible enough to be used for training the model. The process is shown in Figure 3.6, the steps are shown as the white colour numbers, which has 6 steps as follow:

Step 1 (Train): Train the model using the data from top left black block as predictors, and the data from top right black block as responses.

Step 2 (Predict): Use the trained model to predict the middle right black block using the predictors from middle left black block.

Step 3 (Train): Train the model again using the predictors from top left black block, and the data from top middle black block as responses .

Step 4 (Predict): Use the trained model to predict the bottom middle black block using the variables from bottom left black block.

Step 5: (Train): Train the model again using the predictors from top left black block, and the data from top right black block as responses.

Step 6 (Predict): Use the trained model to predict the bottom right black block using the variables from bottom left black block.

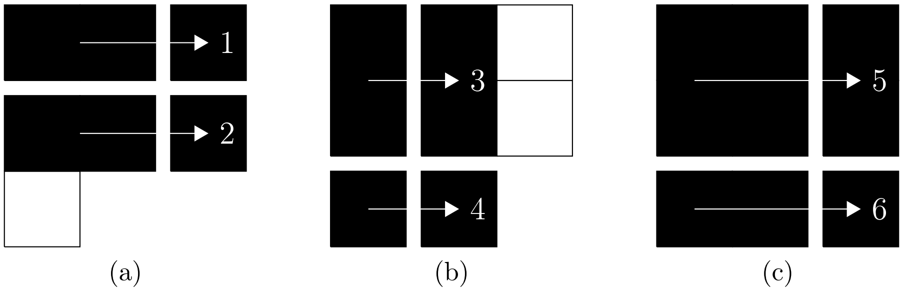

Figure 3.6: Graphical Representation of Cascading (Type 2)

3.4 Support Vector Machines On Triangle-Free Models

Previous sections applied machine learning methods on paid and outstanding data, however, it still has some limits due to the nature of the cascading method as it lacks data for the more recent accident years.

3.4.1 Triangle-Free Model

Ticconi [2018] uses another approach instead of the cascading method, which has the following steps for *Reported But Not Settled* (RBNS) and *Incurred But Not Reported* (IBNR):

- Prediction for the closing delay, in the RBNS case.
- Probability that an existing risk will cause a late reporting claim in a fixed time interval of $r*$, in the IBNR case.
- Estimation for outstanding liabilities.

For RBNS, a proper classification model is needed to estimate probabilities for the random variable closing delay $\boldsymbol{c}$ of each opened claim,

$$\hat{P}(c = c^*|\mathcal{D}_I) = \hat{P}(c = c^*|\boldsymbol{x}(t)). \tag{3.45}$$

Then a regression function f_1 will be build to provide an estimation of outstanding liabilities for RBNS claims

$$X_{i,r}(c) = f_1(\boldsymbol{x}(t)). \tag{3.46}$$

As for IBNR, there will be two tasks. The first task, a classification tool has to be used to predict the probabilities that each existing risk will cause a late reporting claim in a fixed time interval of r^*:

$$\hat{\mathbb{P}}(\mathbb{1}_{[n,\ j<r^*,\ \text{and}\ i\leq I,\ i+j>J-1]}|\mathcal{D}_I) = \hat{\mathbb{P}}(\mathbb{1}_{[n,\ j<r^*,\ \text{and}\ i\leq I,\ i+j>J-1]}|\boldsymbol{x}_n(t)) = p_{\mathbb{1}}(\boldsymbol{x}_n(t)). \tag{3.47}$$

$\boldsymbol{x}_n(t)$ is all the available information collected by the companies in time t. i, j represent the accident period and development period, $I,\ J-1$ are the most recent accident period and development period, and $n = 1, ..., N$ represents each claim as they need to be evaluated separately at the evaluation date.

Selecting a proper cut-off value k, it can be seen that the IBNR claims can be discerned from the rest of the claims according to the probability estimates produced by (3.43):

$$\mathbb{1}_{[n,\ j<r*,\ \text{and } i\leq I,\ i+j>J-1]} = \begin{cases} 1 & \text{if } p_{\mathbb{1}}(\boldsymbol{x}_n(t)) \geq k, \\ 0 & \text{if } p_{\mathbb{1}}(\boldsymbol{x}_n(t)) < k. \end{cases} \tag{3.48}$$

Hence, the second task is to provide an estimation of the outstanding liabilities for each risk marked as IBNR using a regression function f, which can be represented as

$$\hat{X}_{i,j} = f(\boldsymbol{x}_n(t)|\mathbb{1}_{[n,\ j<r*,\ \text{and } i\leq I,\ i+j>J-1]}). \tag{3.49}$$

There are a few regression methods mentioned in Ticconi [2018] and his extended research in 2019 (Ticconi [2019]), but it seems support vector machines (SVM) have the best results among other methods including ANN, and GLM, so only the SVM method will be introduced here.

3.4.2 Support Vector Machines

Introduced by Cortes and Vapnik [1995], SVM represents a good alternative to common machine learning tools. The idea behind SVM is to find, among all possible linear solutions to a regression or classification problem, the one being the farthest from the observed data, as it should be more resilient against noise, because the noise would have less chance to 'cross' the linear solution, a linearly separable binary classification example is shown in Figure 3.7.

As shown in Figure 3.7, the objective is to find a linear decision boundary d_n maximizing the so-called margin, i.e., the "tube" between observed data, represented in Figure 3.7 by two dotted lines.

Given two classes $\mathcal{C}_0$, $\mathcal{C}_1$, a convenient choice of the boundary is $d_n = sign(\mathbf{w}^T\boldsymbol{x}_n + b)$, where $\mathbf{w}$ is the weight vector, which implies that

$$\begin{cases} d_n = +1 & \text{if} \quad \boldsymbol{x}_n \in \mathcal{C}_1 \Leftrightarrow y_n = +1, \\ d_n = -1 & \text{if} \quad \boldsymbol{x}_n \in \mathcal{C}_0 \Leftrightarrow y_n = -1. \end{cases} \tag{3.50}$$

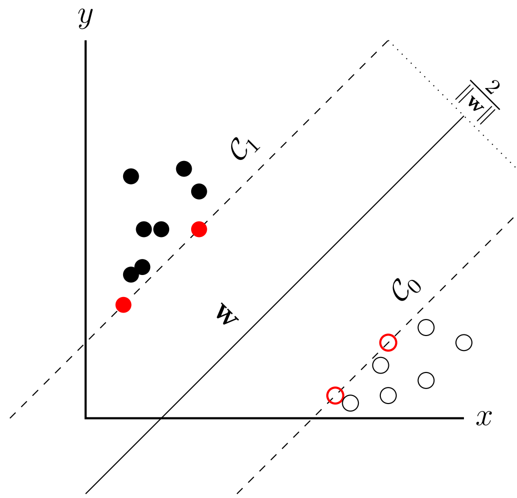

Figure 3.7: Graphical Representation of SVM

It can be shown that the margin equals $M = \frac{2}{\|\mathbf{w}\|}$, hence maximizing the margin entails minimizing over $\mathbf{w}$, which can be represented as

$$\begin{aligned} &\min \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}, \\ &\text{subject to} \quad y_n[\mathbf{w}^T\boldsymbol{x}_n + b], \quad \forall n = 1, \ldots, N, \end{aligned} \tag{3.51}$$

which is a quadratic optimum problem with n convex constraints - requiring that each data point is correctly classified, but it is not necessarily achievable, since the dataset might not be linearly separable.

Equation (3.45) can be solved employing Lagrange multipliers approach, which in dual space assumes the following form

$$\begin{aligned} &\max_{\mathbf{x_n},\mathbf{x_m}} \quad \mathcal{L} = \sum_{n=1}^{N}\lambda_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} y_n y_m \lambda_n \lambda_m \mathbf{x}_n^T\mathbf{x}_m, \\ &\text{subject to} \quad \lambda_n \geq 0, \quad \forall n = 1, \ldots, N, \\ &\sum_{n=1}^{N}\lambda_n y_n = 0, \end{aligned} \tag{3.52}$$

where λ_n are the multipliers and whose solution is given by

$$\begin{aligned} \mathbf{w}^* &= \sum_{n=1}^{N} \lambda_n^* y_n \mathbf{x}, \\ b^* &= \frac{1}{N_{SV}} \sum_{k=1}^{N_{SV}} (y_k - \mathbf{x}_k^T \mathbf{w}^*). \end{aligned} \tag{3.53}$$

Equation (3.47) gives the name to the algorithm: the support vectors are the only input vectors whose multiplier are different from 0, hence contributing to the final solution.

It should be noted that equation (3.46) only depends on the dot product of couples of input vectors $(\mathbf{x}_n, \mathbf{x}_m)$ and this can provide insight on both how inputs contribute to a final solution and how to deal with more complex problems.

A transformation can be applied on equation (3.46) by means of the so-called kernel trick, i.e. employing in equation (3.46) kernel functions $K(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)\phi(\mathbf{x}_m)$, that allow the quantification of vector similarity in transformed spaces, without computing dot products.

Even though it is impossible to determine the best kernel transformation, the radial kernel has shown the best performances empirically overall. Common kernel choices are listed in the Table 3.1.

Kernel	Function
Linear	$K(\mathbf{x}_n, \mathbf{x}_m) = \mathbf{x}_n^T \mathbf{x}_m$
Polynomial	$K(\mathbf{x}_n, \mathbf{x}_m) = (\mathbf{x}_n^T \mathbf{x}_m + 1)^k$
Sigmoid	$K(\mathbf{x}_n, \mathbf{x}_m) =\tanh(\mathbf{x}_n^T \mathbf{x}_m + b)$
Gaussian RBF	$K(\mathbf{x}_n, \mathbf{x}_m) = \exp\,(\frac{\lvert\lvert\mathbf{x}_n - \mathbf{x}_m\rvert\rvert^2}{2\sigma^2})$

Table 3.1: List of Most Common Kernel Functions.

Generalization for non-perfectly-separable problems, that are more common in practice is easily made by introducing a parameter C to delimit the influence of misclassified training points. Adopting a Gaussian RBF kernel, this translates (3.48) into

$$\begin{aligned} \max_{\mathbf{x_n},\mathbf{x_m}} \quad & \mathcal{L} = \sum_{n=1}^{N} \lambda_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} y_n y_m \lambda_n \lambda_m \mathrm{exp}(\frac{||\mathbf{x}_n - \mathbf{x}_m||^2}{2\sigma^2}), \\ \text{subject to} \quad & 0 \leq \lambda_n \leq C, \quad \forall n = 1, \ldots, N, \\ & \sum_{n=1}^{N} \lambda_n y_n = 0. \end{aligned} \tag{3.54}$$

While all of the methodologies presented by Ticconi [2018] have achieved both, comparable results and capacity concerning actual payments, SVMs seem to have shown the best training metrics overall, as SVMs are easier to interpret, are granted with a structure more stable and resilient, and do not require architecture designing comparing to GLMs and Neural Networks.

However, we will not be able to apply this method in this paper, due to the fact that we lack real data. The real data used by Ticconi [2018] are from an Italian insurance company, including information of the policyholder, the loan, the claims and even the macro-economic, but it was not made public.

Chapter 4

Implementations

4.1 Data Handling

4.1.1 The Model for Synthetic Data

Harej et al. [2017] give the model to simulate the individual claims data set based on the unified models described in Taylor et al. [2008]. There are three major components:

$$\begin{aligned}
&\text{Paids:} && P(t) = UF_P(t),\\
&\text{Outstandings:} && O(t) = UF_O(t),\\
&\text{Incurred:} && I(t) = P(t) + O(t).
\end{aligned}$$

where $F_P(t)$ and $F_O(t)$ are both lognormal distributions.

Paids are the payments that are already made and recorded, outstandings are the payments that are still waiting to be paid, and are estimated by the insurers. Incurred is merely the sum of paids and outstandings.

The variable U is the ultimate claim amount at ultimate, which follows a lognormal law $U \sim LN(\mu, \sigma)$; whereas $F_P(t)$, $F_O(t)$ are the development patterns, age to ultimate, and correspond to the cumulative distribution functions of lognormal variates $LN(\mu_t^P, \sigma_t)$ and $LN(\mu_t^O, \sigma_t)$ respectively.

To specify the development pattern of both paid and outstanding, the log-normal distribution for both cases are defined as:

$$F_P(t) \sim LN\Big(\Big[1 - e^{-\frac{t-\tau}{\lambda}}\Big]^{\alpha}, \sigma_t\Big), \qquad \text{for paid,} \tag{4.1}$$

$$F_O(t) \sim LN\Big(\alpha e^{-(\frac{t-\tau}{\lambda})^2}, \sigma_t\Big), \qquad \text{for outstandings.} \tag{4.2}$$

Finally, a Frank copula was used to link the paids and the outstandings, in order to induce a dependency between them across time. The data model assumes a higher dependency in the tail.

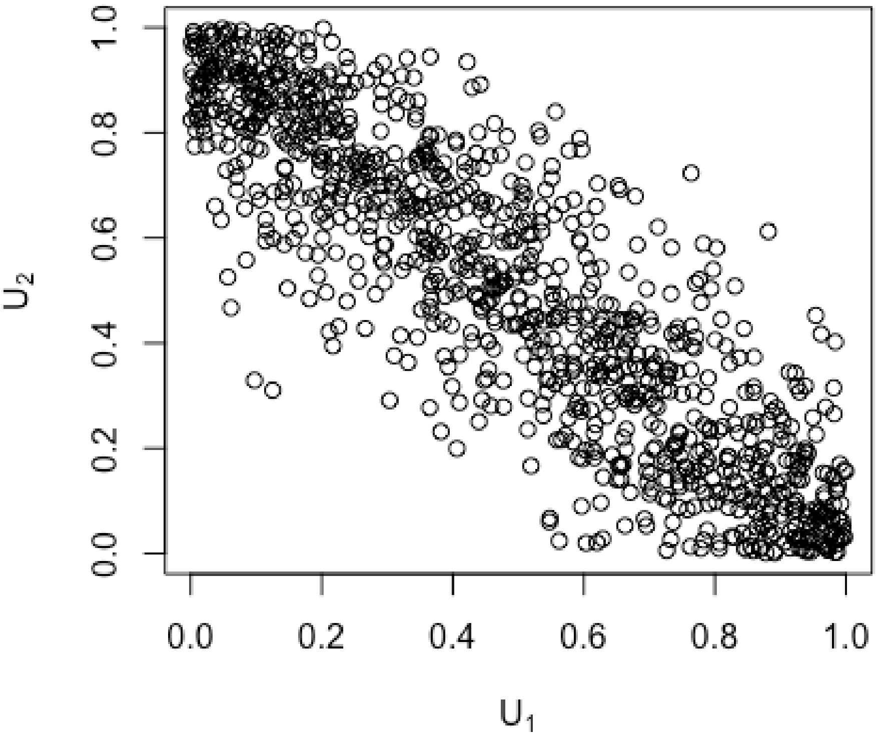

Figure 4.1: Frank Copula

In practice, Archimedean copulas are popular because they allow modeling dependence in arbitrarily high dimensions with only one parameter, governing the strength of the dependence. The Frank copula is a symmetric Archimedean copula; see for example Venter [2002] where θ is the Frank copula parameter:

$$C_\theta(u,v) = -\frac{1}{\theta}\ln\left(1+\frac{(e^{-\theta u}-1)(e^{-\theta v}-1)}{e^{-\theta}-1}\right), \qquad u,v\in[0,1], \tag{4.3}$$

with conditional distribution

$$C_1(u,v) = \frac{(e^{-\theta u}-1)(e^{-\theta v}-1)+e^{-\theta v}-1}{(e^{-\theta u}-1)(e^{-\theta v}-1)+e^{-\theta}-1}, \tag{4.4}$$

and the relationship between Kendall's tau τ and the Frank copula parameter θ is given by:

$$\tau = 1+\frac{4}{\theta}\big(D_1(\theta)-1\big), \tag{4.5}$$

where $D_1(\theta)=\frac{1}{\theta}\int_0^\theta \frac{t}{e^t-1}dt$.

Harej et al. [2017] chose $\theta<0$ to create a negative association between $F_P(t)$ and $F_O(t)$. In order to simulate the data for $P(t)$ and $O(t)$, the first step is to simulate u and p by random draws on $[0,1]$. Here p is considered a draw from the conditional distribution of $V \mid u$, since this has distribution function C_1, v can then be found as:

$$v = C_1^{-1}(p\mid u) = -\frac{1}{\theta}\ln\left(1+\frac{p(e^{-\theta}-1)}{1+(1-p)(e^{-\theta u}-1)}\right). \tag{4.6}$$

Therefore, the variables of interest t_1 and t_2 can be simulated by inverting the marginal distributions, i.e., $t_1=F_{t_1}^{-1}(u)$ and $t_2=F_{t_2}^{-1}(v)$.

To see the effect more properly, Harej et al. [2017] give two types of samples, one is short-tailed, and the other is long-tailed, which could correspond to material and non-material motor claim damages under motor third party liability. Sample 1 includes 6,000 short-tailed claims, while Sample 2 includes 4,000 long-tailed claims; the parameters used for both sample are shown in Table 4.1. Both samples were calibrated to have a mean ultimate amount of paid claims of 1 million. All standard deviations were set to be 2%.

Lopez [2019] gives another model for micro-level claim reserving with the presence of censoring, which also involves copulas. The model contains three parts: the first part is

	Short-tailed sample		Long-tailed sample	
	paids	outs	paids	outs
τ	-1.0	1.6	-3.0	2.0
λ	2.0	5.0	6.0	5.0
α	1.5	2.0	3.0	0.6

Table 4.1: Calibration Parameters of Sample 1 and Sample 2

to predict the delay time T, using a Cox regression model, and an accelerated failure time model (AFT); the second part is to predict the amount of a claim M, with a generalized linear model (GLM); then, it combines these two models with a copula. Frank's, Clayton's, and Gumbel's models were compared in the paper. A Clayton's model was suggested for its superior performance.

4.1.2 Mock Samples

With Sample 1 and Sample 2, which are both generated using the models from Taylor et al. [2008], Harej et al. [2017] produced three more samples by mixing these two. The first mixed sample (Sample 3) can be described as a sample that has a stable ratio between claims of different patterns between long-tailed claims and short-tailed claims. The next has a changing pattern from a high ratio of long-tailed claims towards a high ratio of short-tailed claims (Sample 4). The last sample (Sample 5) presents a sudden change in a ratio at last two accident years where there are more short-tailed claims and less or none long-tailed claims.

We present the allocation of these three samples in the following tables and provide the aggregate data in Table 1.1 of Sample 3 to illustrate how the traditional and stochastic methods work.

	Sample 3	
Accident Year	Sample 1 claims	Sample 2 claims
1998	300	200
1999	300	200
2000	300	200
2001	300	200
2002	300	200
2003	300	200
2004	300	200
2005	300	200
2006	300	200
2007	300	200
2008	300	200
2009	300	200
2010	300	200
2011	300	200
2012	300	200
2013	300	200
2014	300	200
2015	300	200
2016	300	200
2017	300	200

Table 4.2: Allocation of Sample 1 and Sample 2 in Sample 3

	Sample 4	
Accident Year	Sample 1 claims	Sample 2 claims
1998	15	390
1999	45	370
2000	75	350
2001	105	330
2002	135	310
2003	165	290
2004	195	270
2005	225	250
2006	255	230
2007	285	210
2008	315	190
2009	345	170
2010	375	150
2011	405	130
2012	435	110
2013	465	90
2014	495	70
2015	525	50
2016	555	30
2017	585	10

Table 4.3: Allocation of Sample 1 and Sample 2 in Sample 4

	Sample 5	
Accident Year	Sample 1 claims	Sample 2 claims
1998	280	220
1999	280	220
2000	280	220
2001	280	220
2002	280	220
2003	280	220
2004	280	220
2005	280	220
2006	280	220
2007	280	220
2008	280	220
2009	280	220
2010	280	220
2011	280	220
2012	280	220
2013	280	220
2014	280	220
2015	280	220
2016	460	40
2017	500	0

Table 4.4: Allocation of Sample 1 and Sample 2 in Sample 5

4.1.3 Loss Reserving Data from NAIC Schedule P

The data used here are from National Association of Insurance Commissioners (NAIC) Schedule P[1]. The dataset corresponds to claims from accident years 1988-1997, with development experience of 10 years for each accident year. Six data sets are available at the CAS website, but only four data sets are used here, including the *PP Auto Data Set*, *Commercial Auto Data Set*, *Product Liability Data Set*, *Other Liability Data Set*. However, the data is not completely available at an individual level, it is aggregate based on the company names.

A list of variables in the data is as follows:

- GRCODE NAIC: company code (including insurer groups and single insurers)

- GRNAME NAIC: company name (including insurer groups and single insurers)

- AccidentYear: Accident year (1988 to 1997)

- DevelopmentYear: Development year (1988 to 1997)

- DevelopmentLag: Development year (AY - 1987 + DY - 1987 - 1)

- IncurLoss_: Incurred losses and allocated expenses reported at year end

- CumPaidLoss_: Cumulative paid losses and allocated expenses at year end

- BulkLoss_: Bulk and IBNR reserves on net losses and defence and cost containment expenses reported at year end

- PostedReserve97_: Posted reserves in year 1997 taken from the Underwriting and Investment Exhibit - Part 2A, including net losses unpaid and unpaid loss adjustment expenses

- EarnedPremDIR_: Premiums earned at incurral year - direct and assumed

- EarnedPremCeded_: Premiums earned at incurral year - ceded

- EarnedPremNet_: Premiums earned at incurral year - net

- Single: 1 indicates a single entity, 0 indicates a group insurer

However, only the GRCODE NAIC, AccidentYear, DevelopmentYear, and CumPaidLoss are used to build the individual training sets, and there are 4,593 claims in total, the allocation in each accident year is as follow:

[1] Loss reserving data pulled from NAIC Schedule P: `https://www.casact.org/research/index.cfm?fa=loss_reserves_data`.

Accident Year	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997
Number of claims	413	427	433	439	456	475	470	482	498	500

Table 4.5: Number of Claims in Each Accident Year

4.2 Results

4.2.1 Simulated Data

For the traditional and stochastic methods, all the data from Samples 3 to 5 are first aggregated in the triangle form based on the accident years and development years; then we can apply the classical and stochastic methods.

Following are the ultimate claim amounts and claims loss reserving for Sample 3, 4 and 5 along with the mismatch rate, which is the percentage of the difference between prediction and real data over the real data.

The methods compared here include the CL algorithm, ODP Mack model, ANN (with cascading), and RF (with modified cascading) methods.

Sample 3: Stable ratio case

	Ultimate Claims of Sample 3 (in dollars)							
Accident Year	CL	mismatch	ODP Mack Model	mismatch	ANN	mismatch	RF	mismatch
1998	-	-	-	-	-	-	-	-
1999	485,702,128	0.000%	485,702,141	0.000%	485,713,783	0.003%	485,700,217	0.000%
2000	485,801,030	0.006%	485,800,878	0.007%	485,835,736	0.001%	485,800,306	0.007%
2001	485,482,809	0.003%	485,482,552	0.003%	485,552,231	0.011%	485,485,662	0.003%
2002	485,084,279	0.001%	485,084,440	0.001%	485,121,017	0.007%	485,060,627	0.006%
2003	485,041,436	0.006%	485,041,550	0.006%	485,024,279	0.010%	485,075,584	0.001%
2004	485,211,372	0.002%	485,212,037	0.002%	485,297,609	0.020%	485,209,508	0.002%
2005	484,186,605	0.010%	484,187,886	0.010%	484,249,785	0.003%	484,236,081	0.000%
2006	485,229,372	0.005%	485,228,763	0.005%	485,582,913	0.068%	485,254,448	0.000%
2007	484,858,506	0.006%	484,857,817	0.006%	485,347,030	0.107%	484,831,247	0.001%
2008	484,872,466	0.012%	484,872,922	0.012%	485,163,202	0.048%	484,940,046	0.002%
2009	485,411,975	0.019%	485,412,516	0.019%	486,077,010	0.118%	485,494,199	0.002%
2010	485,560,641	0.028%	485,561,119	0.028%	486,632,381	0.249%	485,416,928	0.002%
2011	485,061,104	0.001%	485,059,611	0.001%	486,877,746	0.376%	485,043,084	0.002%
2012	485,124,682	0.012%	485,126,197	0.012%	487,210,086	0.442%	485,064,744	0.001%
2013	485,852,926	0.028%	485,850,814	0.028%	488,224,865	0.517%	485,705,204	0.002%
2014	484,365,182	0.063%	484,365,770	0.063%	487,300,649	0.542%	484,664,568	0.001%
2015	485,239,255	0.035%	485,244,533	0.036%	486,604,879	0.316%	485,103,608	0.007%
2016	485,523,771	0.031%	485,524,902	0.031%	486,915,223	0.255%	485,661,160	0.003%
2017	484,614,480	0.010%	484,615,325	0.010%	487,227,180	0.529%	484,702,307	0.008%
Total	9,703,647,527	0.014%	9,218,231,773	0.014%	9,235,957,603	0.180%	9,218,449,526	0.002%
Reserve	1,943,211,588	0.014%	1,943,219,340	0.013%	1,960,945,170	0.899%	1,943,437,093	0.002%

Table 4.6: Predictions for Sample 3

Sample 4: From a high ratio of long-tailed claims to short-tailed claims

	Ultimate Claims of Sample 4 (in dollars)							
Accident Year	CL	mismatch	ODP Mack Model	mismatch	ANN	mismatch	RF	mismatch
1998	-	-	-	-	-	-	-	-
1999	500,766,726	0.006%	500,766,726	0.006%	500,795,465	0.000%	500,782,613	0.002%
2000	499,948,350	0.002%	499,948,353	0.002%	499,919,612	0.008%	499,947,492	0.003%
2001	500,194,589	0.003%	500,194,433	0.003%	500,129,440	0.016%	500,196,095	0.002%
2002	501,117,238	0.001%	501,116,758	0.001%	501,079,126	0.009%	501,117,913	0.001%
2003	499,957,449	0.007%	499,957,143	0.007%	499,913,252	0.016%	499,971,489	0.004%
2004	499,615,719	0.005%	499,615,569	0.005%	499,584,109	0.012%	499,647,365	0.001%
2005	500,382,186	0.001%	500,381,166	0.002%	500,434,001	0.009%	500,383,954	0.001%
2006	500,055,037	0.002%	500,055,051	0.002%	500,033,136	0.006%	500,045,696	0.004%
2007	500,057,391	0.003%	500,058,390	0.003%	499,944,202	0.020%	500,055,972	0.002%
2008	499,948,534	0.005%	499,949,002	0.005%	499,951,638	0.004%	499,951,415	0.004%
2009	499,833,709	0.003%	499,834,545	0.002%	499,753,905	0.018%	499,869,424	0.005%
2010	499,944,113	0.004%	499,942,227	0.004%	500,009,277	0.009%	499,942,324	0.004%
2011	499,937,871	0.001%	499,937,213	0.001%	499,906,180	0.007%	499,949,210	0.002%
2012	500,452,642	0.003%	500,451,461	0.003%	500,443,221	0.005%	500,471,178	0.001%
2013	500,620,229	0.003%	500,615,716	0.004%	500,630,190	0.001%	500,623,309	0.003%
2014	500,759,277	0.001%	500,763,405	0.002%	500,755,709	0.000%	500,743,718	0.002%
2015	500,043,465	0.006%	500,042,454	0.006%	500,083,096	0.002%	500,053,140	0.004%
2016	500,336,821	0.002%	500,345,298	0.001%	500,263,084	0.017%	500,334,520	0.003%
2017	499,785,826	0.001%	499,787,524	0.000%	499,787,966	0.000%	499,801,104	0.002%
Total	9,503,757,172	0.003%	9,503,762,433	0.003%	9,503,416,609	0.006%	9,503,887,929	0.001%
Reserve	1,034,791,457	0.024%	1,034,796,717	0.023%	1,034,450,893	0.057%	1,034,922,214	0.011%

Table 4.7: Predictions for Sample 4

Sample 5: Sudden change in ratio in the last two accident years

	Ultimate Claims of Sample 5 (in dollars)							
Accident Year	CL	mismatch	ODP Mack Model	mismatch	ANN	mismatch	RF	mismatch
1998	-	-	-	-	-	-	-	-
1999	484,516,515	0.002%	484,516,287	0.002%	484,394,541	0.027%	484,510,651	0.003%
2000	484,178,471	0.007%	484,178,976	0.007%	484,076,266	0.028%	484,196,892	0.003%
2001	483,765,744	0.006%	483,766,537	0.006%	483,657,545	0.029%	483,778,879	0.004%
2002	483,473,296	0.000%	483,472,791	0.000%	483,381,284	0.019%	483,454,671	0.004%
2003	483,549,727	0.007%	483,551,807	0.006%	483,470,427	0.023%	483,574,047	0.002%
2004	483,613,572	0.002%	483,617,573	0.001%	483,759,470	0.028%	483,606,770	0.003%
2005	483,277,055	0.008%	483,277,817	0.008%	483,686,116	0.077%	483,306,221	0.002%
2006	483,134,218	0.010%	483,134,553	0.010%	483,761,304	0.120%	483,219,923	0.008%
2007	483,624,352	0.004%	483,627,392	0.003%	483,958,056	0.065%	483,643,066	0.000%
2008	483,693,162	0.031%	483,696,087	0.030%	484,362,850	0.107%	483,821,793	0.004%
2009	483,515,289	0.014%	483,522,366	0.015%	484,642,854	0.247%	483,441,365	0.001%
2010	483,136,402	0.005%	483,139,203	0.004%	484,514,532	0.281%	483,133,660	0.005%
2011	484,165,156	0.024%	484,172,855	0.025%	481,917,655	0.441%	484,039,654	0.002%
2012	483,814,155	0.019%	483,814,552	0.019%	482,017,936	0.390%	483,916,850	0.003%
2013	483,559,820	0.010%	483,554,289	0.009%	482,113,134	0.289%	483,523,673	0.003%
2014	484,591,624	0.032%	484,593,432	0.033%	481,858,216	0.532%	484,403,538	0.007%
2015	483,019,516	0.059%	483,029,653	0.056%	482,249,714	0.218%	483,306,653	0.001%
2016	686,707,588	38.137%	686,711,995	38.138%	499,271,891	0.433%	497,110,780	0.002%
2017	723,564,880	44.731%	723,564,740	44.731%	502,865,647	0.586%	499,950,982	0.003%
Total	9,632,900,541	4.478%	9,632,942,903	4.478%	9,219,959,436	0.001%	9,219,940,070	0.001%
Reserve	2,366,596,125	21.131%	2,366,638,487	21.133%	1,953,655,020	0.005%	1,953,635,654	0.006%

Table 4.8: Predictions for Sample 5

Figure 4.2 - 4.4 give a more direct visual representation.

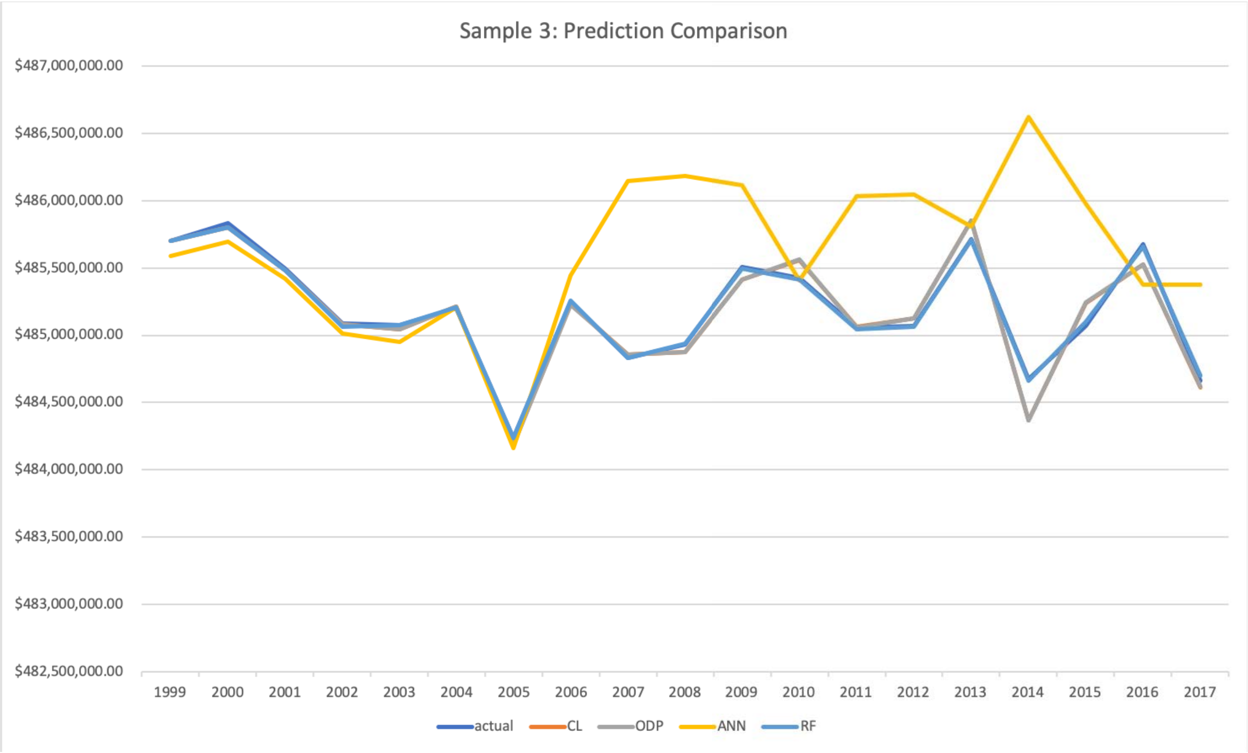

Figure 4.2: Prediction Comparison for Sample 3

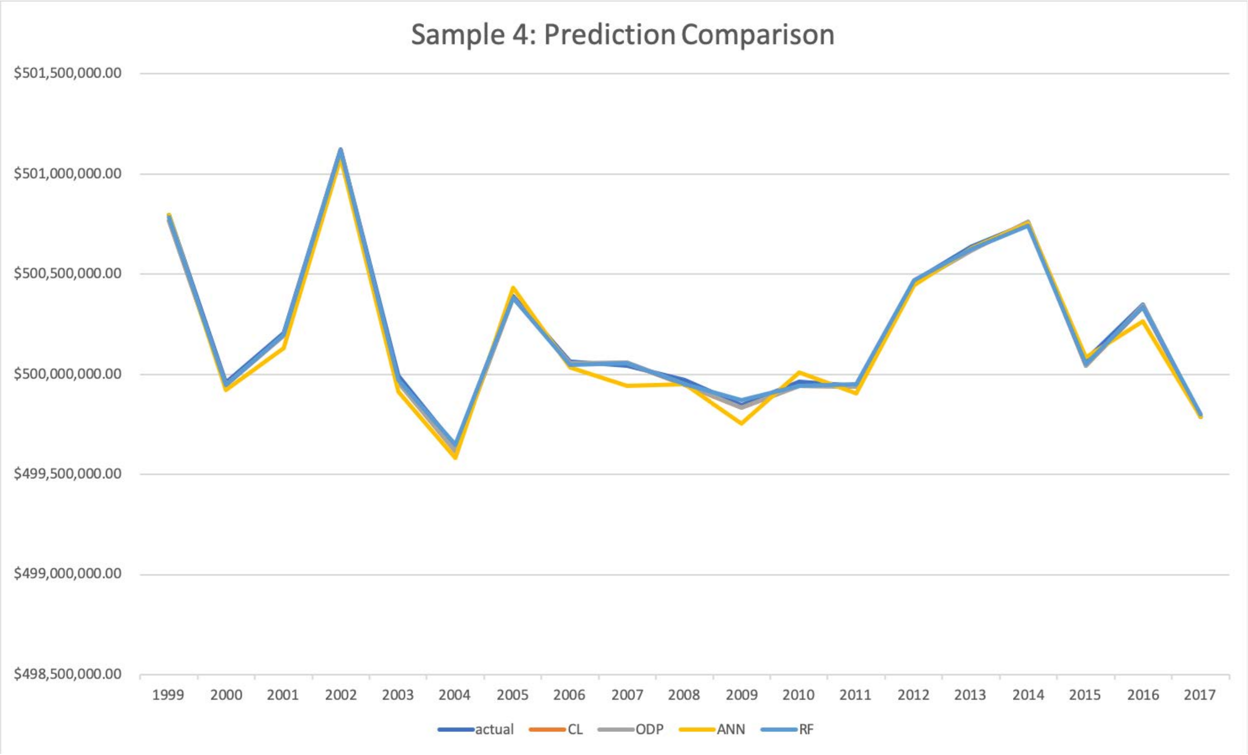

Figure 4.3: Prediction Comparison for Sample 4

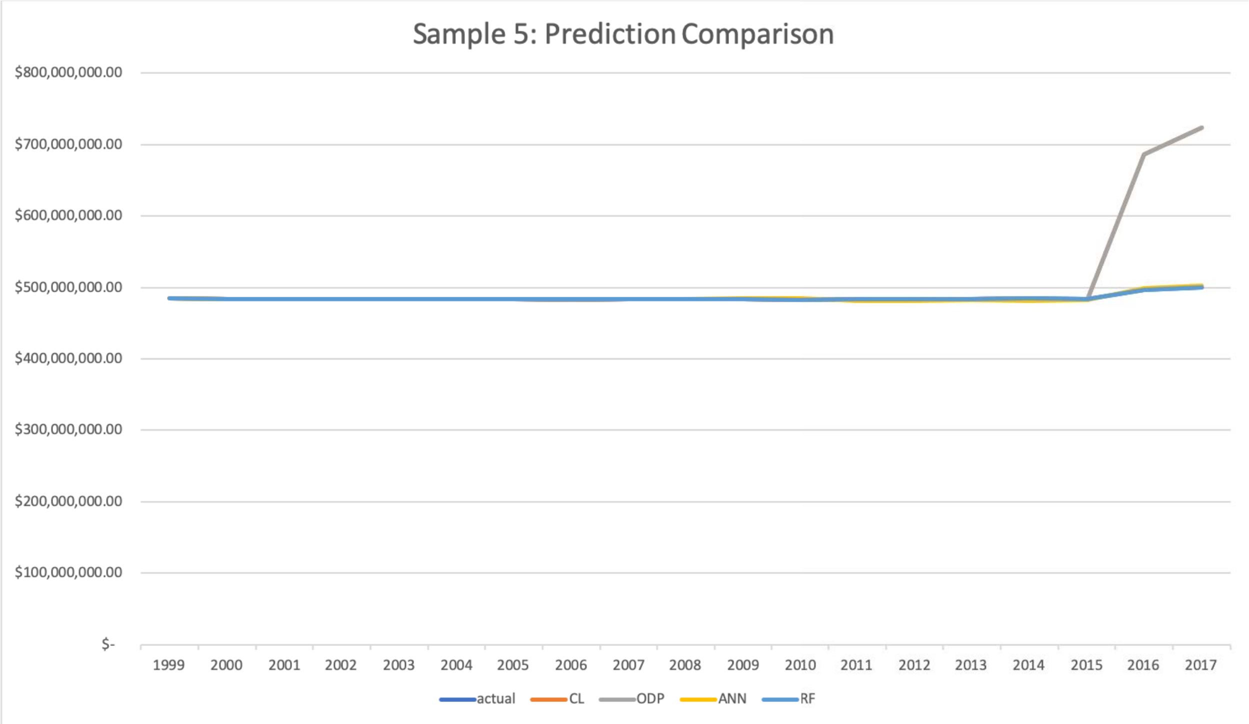

Figure 4.4: Prediction Comparison for Sample 5

In Figure 4.2, the prediction of ANN seems good until development year 8, when predictions start to drift away from actual values. This could be due to the decrease in size of the training data set. In Figure 4.3, we can see all four models perform well. However in Figure 4.4, there is a high peak at development year 17, due to the predictions from the traditional Chain Ladder (CL) and the over-dispersed poisson (ODP) methods, while the prediction of ANN and RF are still close to the actual value.

It is evident that the traditional Chain Ladder (CL) method is predicting well when the pattern varies on a small scale. The over-dispersed poisson (ODP) method can increase the performance slightly compared to the CL method. However, traditional methods surely have flaws when the pattern of the claims has a sudden change (as in Sample 5), which makes it unpredictable with methods that use aggregate data.

On the other hand, the ANN method does not necessarily perform better (see Sample 3). It can be strenuous to define the number of hidden layers and of neurons. At the same time, the performance is affected by the nature of the cascading method, which might cause less accurate predictions in recent years due to lack of information. However, individual claims reserving methods can track down the sudden changes and predict more precisely (as in Sample 5). It has to be noted that in Samples 3 and 4, only the ANN method over-estimated the reserve amount, while the rest of the methods under-estimated the reserve.

Moreover, we see that the random forest (RF) method with modified cascading performs better compared to the ANN method with regular cascading. Even in Sample 5, when ANN and RF show almost the same mismatch for reserves, the RF method has much better performance when it comes to predicting each ultimate claim from different accident years. The reserve amounts under the ANN and RF methods are much lower and obviously closer to the real data. The number might be slightly conservative, but there is still a huge gap (around 400 millions dollars) comparing to the CL and ODP Mack models, and this can save insurance companies a lot of money.

Above all, the RF method with the modified cascading method outperforms all the other methods. However, the running time of this method can very time-consuming as shown in Table 4.9.

	CL	ODP	ANN	RF
Time	1 second	2 minutes	8 minutes	1 hour 30 minutes

Table 4.9: Running Time of Different Methods

With more computational power, it is possible to run the model on the cloud service, which is easier and faster. Then, the real problem would be calculating the cost of running this cloud service and comparing it to the amount of money saved from more accurate reserves prediction.

By changing the parameters in Table 4.1 as in the following Table 4.10, we would have two new samples, short-tailed Sample a and long-tailed Sample b, which are used to confirm the results we obtained from Samples 1-5. Standard deviations were set to be 3%,

	Short-tailed sample		Long-tailed sample	
	paids	outs	paids	outs
τ	-1.5	1.2	-3.5	1.5
λ	1.5	3.5	5.0	4.5
α	1.0	1.5	2.5	0.4

Table 4.10: Calibration Parameters of Sample a and Sample b

After redoing the whole sampling process, three new samples can be produced. Samples c and d have the same allocation as Samples 3 and 4, whereas Sample e also has a sudden change in ratio, not in the last two accident years, but two years before the most recent year, as shown in the Table 4.11.

	Sample e	
Accident Year	Sample a claims	Sample b claims
1998	280	220
1999	280	220
2000	280	220
2001	280	220
2002	280	220
2003	280	220
2004	280	220
2005	280	220
2006	280	220
2007	280	220
2008	280	220
2009	280	220
2010	280	220
2011	280	220
2012	280	220
2013	280	220
2014	280	220
2015	420	80
2016	500	0
2017	280	220

Table 4.11: Allocation of Sample a and Sample b in Sample e

The prediction of loss reserves from the second set of samples are provided below.

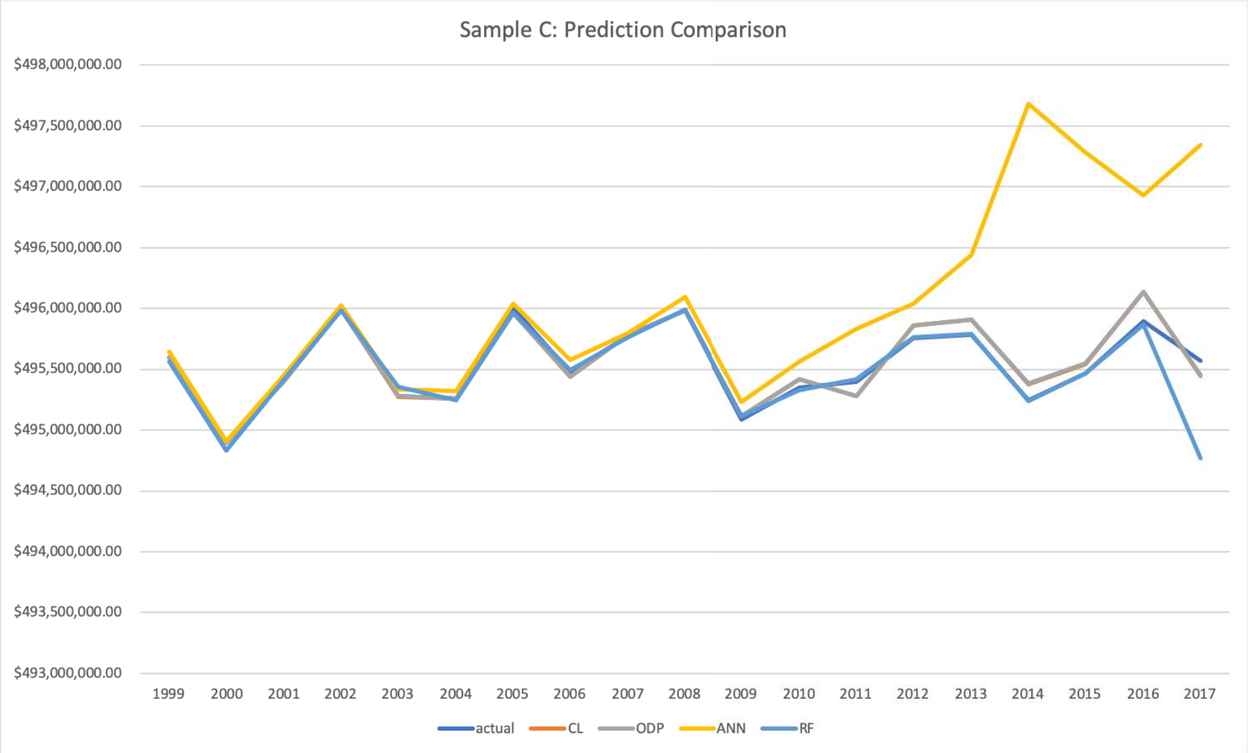

Figure 4.5: Graphical Prediction Comparison for Sample c

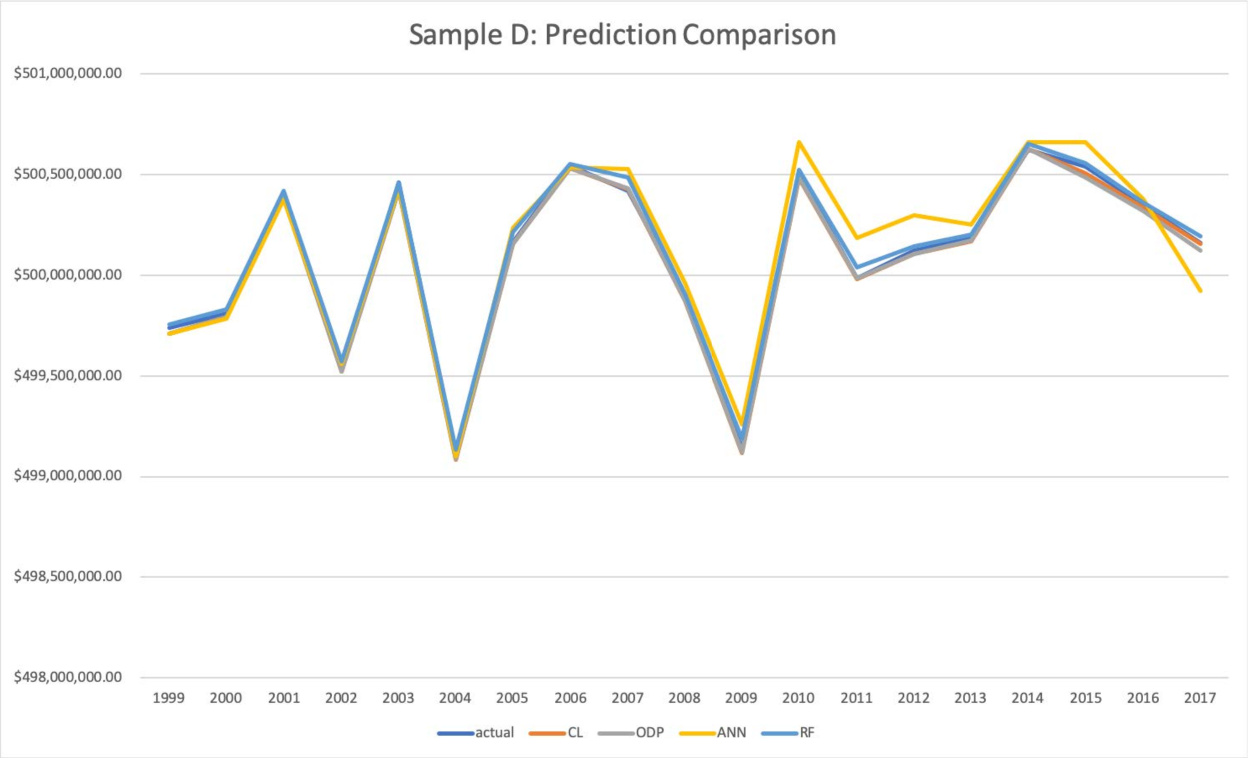

Figure 4.6: Graphical Prediction Comparison for Sample d

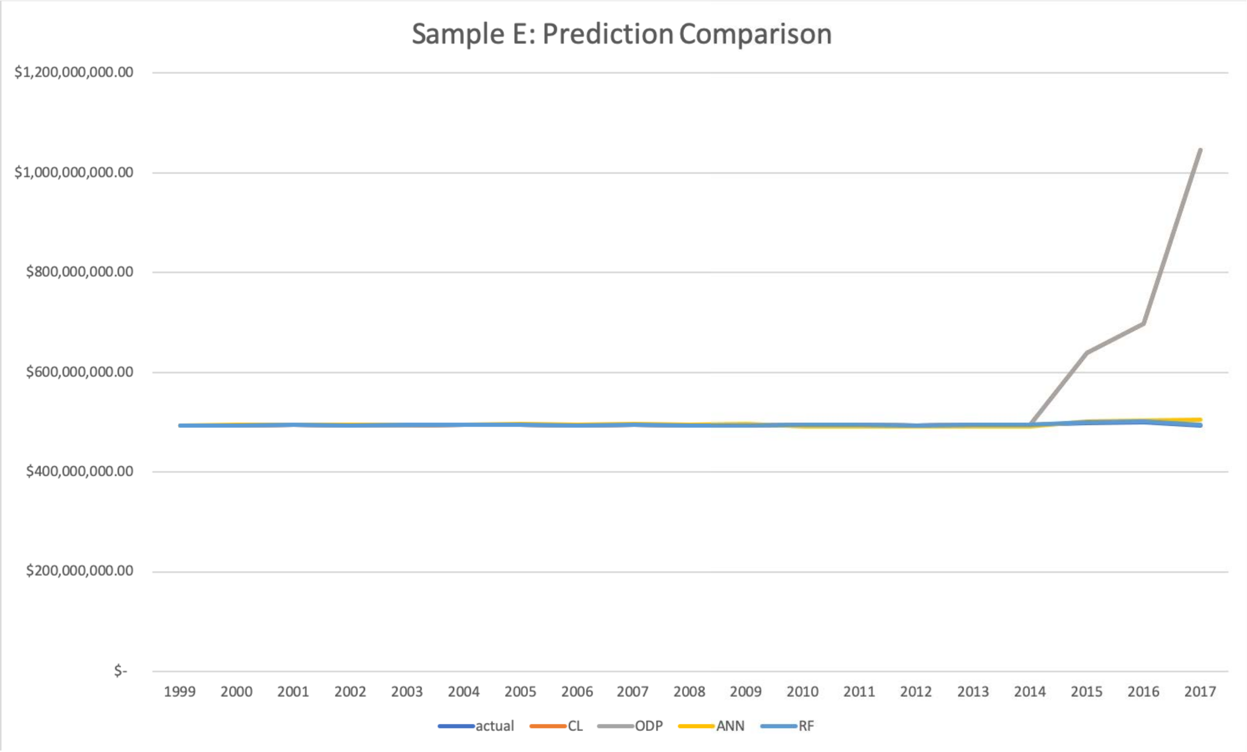

Figure 4.7: Graphical Prediction Comparison for Sample e

From Figures 4.5-4.7, the final result still yields the same conclusion as the result from Samples 3-5, that is when the pattern varies little, the aggregate methods perform well and can be useful. Between the different methods, there is only about 1%-2% difference in reserves. However, when the ratio gets more complicated, the individual method would outperform the aggregate method on a dramatic scale, which is about 10% difference in reserves. On the level of each accident year, the prediction from the aggregate methods can be much worse around the years when the ratio changes.

4.2.2 Real Data

The prediction of the real data is shown below.

Method	Loss reserves mismatch rate
CL	7.79%
ODP	5.07%
ANN	5.22%
RF	-256%
RF - INC	-4.95%

Table 4.12: Comparison of loss reserve

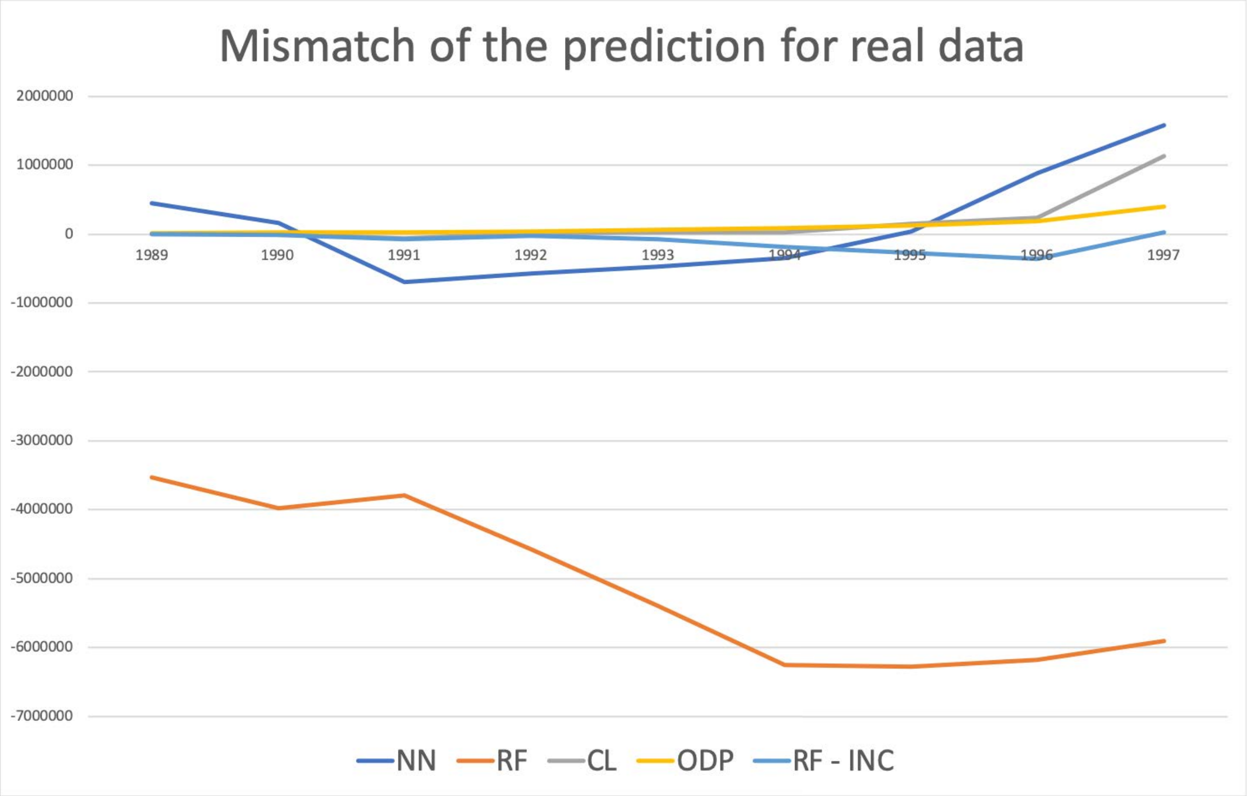

Figure 4.8: Graphical Prediction Comparison for Real Data

In Table 4.12, both ODP and ANN show a good loss reserves prediction for about 5% mismatch rate, while CL has 7.79%. However, RF using accumulative data shows a dramatic bad result with over 200% difference. After inspecting the data more carefully, the maximum value at development year 10 is 6,815,646 for claims from accident year 1988, while the maximum value is 10,512,108 for all accident years in development year 10. This explains why the predictions at development year 10 are much lower than the actual data, since the patterns from later years (1989 - 1997) are not apparent in accident year 1988. By using incremental data with the RF method, this issue gets resolved. The result (RF - INC) performs much better, with only -4.95% loss reserves mismatch rate.

Table 4.11 summarizes the advantages and disadvantages of the reserving methods compared in this thesis.

	Comparison			
	Structured Data		Unstructured Data	
	Advantage	Disadvantage	Advantage	Disadvantage
Group Methods	Easier and faster to compute; Good performance if the pattern does not vary too much	Bad performance when patterns vary a lot	N/A	
CL	Very basic, easy to interpret and apply;	Aggregate data only	N/A	
BF	Expert's advice can be considered; More details with each payments	Aggregate data only	N/A	
CC	Incremental payments are being presented	Aggregate data only	N/A	
Bayesian	Posterior distribution can be calculated analytically	Aggregate data only	N/A	
Individual Methods	More information can be included	Might be difficult to interpret	All variables can be considered;	Might be time-consuming
GLM	High efficiency with a small number of parameters	Does not necessarily improve predictive efficiency	N/A	

ANN	Ability to work with incomplete data	CPU intensive calculations;
		Risk of over-fitting;
		Determination of proper network structure (number of layers, and nodes)
			Good with unstructured data	N/A

RF	Less risk of overfitting	Even more time-consuming than other methods; Require a large amount of training data set	N/A	
SVM	More flexible when using the kernels; No local minima;	Might be sensitive to overfitting the model	Good on a large dimension of data	Might be sensitive to overfitting the model

Table 4.13: Advantages and Disadvantages of Different Methods

Conclusion and Further Research

The purpose of this thesis is to review methods that can be used for loss reserving prediction and compare the performance between classical methods and machine learning methods. The methods reviewed in this thesis include aggregate claims reserving methods (classical methods, stochastic methods), and individual claims reserving methods (GLMs, machine learning methods). We want to see if individual claims reserving methods should be adopted more generally in the insurance industry.

At a time when computational power was not sufficient to apply individual claims reserving methods, the classical methods like the Chain Ladder and Bornhuetter-Ferguson algorithms dominated. They show a good performance on aggregate data when it has a relatively steady pattern for the claims payments. However, insurance policies are becoming much more complex nowadays; the uncertainty of each payment creates a significant problem for the accuracy in predicting loss reserves.

To calculate the prediction uncertainty, the models need to be set in a stochastic framework; for instance the Mack models, cross-classified models, and Bayesian CL models. They provide a way to predict the aggregate loss reserves, and they also provide some metrics to compare the goodness-of-fit of each method, again, due to the nature of aggregated data, here a great amount of information is lost when the data is combined.

Over the last 10 years, the technology has developed and simultaneously the computational power has improved on a large scale. Therefore, individual claims reserving methods have to be brought in because they use more information for predictions. We reviewed some of these methods such as generalized linear models (GLM), artificial neural networks (ANN), random forests (RF), and support vector machines (SVM).

We simulated some claims to apply those different methods and compare the results. We used the same approach as Harej et al. [2017] did which is based on the GLM model from Taylor et al. [2008]. A set of real data was also pulled from NAIC Schedule P, however, the data are aggregated to company level instead of individual level.

Four different methods were then applied to two sets of samples and the real data, including the CL algorithm, ODP Mack model, ANNs, and RF methods, covering the different approaches, from classical methods to stochastic methods and individual claims reserving methods. In the end, traditional models seem to performs as well as individual methods in some cases, however, in other cases, their performance can be dramatically worse than the individual methods, especially in simulated data. In real cases, the individual methods perform as well as the traditional methods, but not necessarily better, as the training set is not sufficiently large in the real data we used.

Carrato and Visintin [2019] provide another approach in loss reserving prediction which combines traditional methods with machine learning methods. Their objective is to find the balance between the interpretability of traditional methods and the predictive power of ML methods. They suggest to cluster the claims first with machine learning for instance with k-means or Gaussian mixtures, then apply the traditional methods to predict the reserves. This is similar to the random forest method, which would put similar claims into the same nodes on a single decision tree.

In Table 4.11, all the aggregate and individual methods we reviewed in this thesis are compared, in order to see their advantages and disadvantages for structured data and unstructured data.

Future, research could include:

1. These methods need to be applied on real individual data (without any aggregation), to test the efficiency of these machine learning methods against the classical and stochastic methods, and find which method fit which case the best.

2. Harej et al. [2017] also suggested that it would be good to get rid of the cascading method. Ticconi [2018] provides an idea to escape from the cascading method and gives a way of predicting the loss reserves; however he/she applies the methods on both structured and unstructured data, while we only applied these methods on structured

data, as seen in the three samples.

3. Try deep learning methods such as recurrent neural networks (Rumelhart et al. [1986]) with long short-term memory (Hochreiter and Schmidhuber [1997]), a method which is good at classifying, processing and predicting based on time series data.

4. Then, it is possible to increase the performance with a hybrid method by combining different methods. Duval and Pigeon [2019] applied gradient boosting techniques to evaluate loss reserves in an individual framework. Then the techniques can also be applied on Random Forest, as a boosted trees method.

Bibliography

[1] Marco Aleandri. Case reserving in non-life practice using individual data and machine learning. Technical report, Univ. La Sapienza, Rome, Italy, 2017. Publication code: 3, ISSN:2279-798X, http://www.dss.uniroma1.it/it/node/6930.

[2] Maximilien Baudry and Christian Y. Robert. A machine learning approach for individual claims reserving in insurance. *Applied Stochastic Models in Business and Industry*, 35:1127–1155, October 2019.

[3] Alessandro Carrato and Michele Visintin. From the chain ladder to individual claims reserving using machine learning techniques. Technical report, ASTIN meetings, 2019. https://insurancedatascience.org/downloads/Zurich2019/Session%203/1_Alessandro_Carrato.pdf.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20 (3):273–297, 1995.

[5] Dorothea Diers and Marc Linde. The multi-year non-life insurance risk in the additive loss reserving model. *Insurance: Mathematics and Economics*, 52(3):590–598, January 2013.

[6] Francis Duval and Mathieu Pigeon. Gradient boosting techniques for individual loss reserving in non-life insurance. *Risks, MDPI, Open Access Journal*, 7(3):1–18, July 2019.

[7] Lukas Hahn. Multi-year non-life insurance risk of dependent lines of business in the

multivariate additive loss reserving model. *Insurance: Mathematics and Economics*, 75: 71–81, April 2017.

[8] Bor Harej, Roman Gächter, and Salma Jamal. Individual claim development with machine learning. Technical report, ASTIN meetings, 2017. http://www.actuaries.org/panama2017/docs/papers/1a_ASTIN_Paper_Harej.pdf.

[9] Thomas Hartl. A comparison of resampling methods for bootstrapping triangle GLMs. *Variance*, 11:60–73, 2014.

[10] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, Zurich, Switzerland, 2017.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[12] Salma Jamal. Machine learning and traditional methods synergy in non-life reserving. Technical report, ASTIN meetings, 2018. https://www.actuaries.org/IAA/Documents/ASTIN/ASTIN_MLTMS%20Report_SJAMAL.pdf.

[13] Uri Korn. Strategies for modeling loss development: Curve fitting, credibility, and layer adjustments. *Variance*, 11:95–117, 2015.

[14] Nathan Lally and Brian Hartman. Estimating loss reserves using hierarchical Bayesian Gaussian process regression with input warping. *Insurance: Mathematics and Economics*, 82:124–140, July 2018.

[15] Olivier Lopez. A censored copula model for micro-level claim reserving. *Insurance: Mathematics and Economics*, 87:1–14, 2019.

[16] Thomas Mack. Distribution-free calculation of the standard error of chain ladder reserve estimates. *ASTIN Bulletin: The Journal of the IAA*, 23(2):214–215, 1993.

[17] Ricardo A. Maronna, Douglas R. Martin, and Victor J. Yohai. *Robust Statistics: Theory and Methods*. John Wiley and Sons, New York, 2006.

[18] Glenn G Meyers. Dependencies in stochastic loss reserve models. *Variance*, 11:74–94, 2016.

[19] Martin F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, November 1993.

[20] Peter Mulquiney. Artificial neural networks in insurance loss reserving. In *9th Joint International Conference on Information Sciences (JCIS-06)*. Atlantis Press, 2006. ISBN 978-90-78677-01-7. `https://doi.org/10.2991/jcis.2006.67`.

[21] Kris Peremans, Stefan Van Aelst, and Tim Verdonck. A robust general multivariate chain ladder method. *Risks*, 6(4):108, 2018.

[22] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

[23] Stefano Strascia and Agostino Tripodi. Overdispersed-Poisson model in claims reserving: Closed tool for one-year volatility in GLM framework. *Risks*, 6(4):139, 2018.

[24] Greg Taylor and Gráinne McGuire. *Stochastic loss reserving using generalized linear models*. Number 3 in CAS Monograph. Casualty Actuarial Society, 2016.

[25] Greg Taylor, Gráinne McGuire, and James Sullivan. Individual claim loss reserving conditioned by case estimates. *Annals of Actuarial Science*, 3(1-2):215–256, 2008.

[26] Damiano Ticconi. Individual claims reserving in credit insurance using GLM and machine learning. Technical report, 12 2018. `https://www.researchgate.net/publication/329761060_Individual_claims_reserving_in_Credit_insurance_using_GLM_and_Machine_Learning`.

[27] Damiano Ticconi. Individual claims reserving in creditor protection insurance using machine learning. Technical report, 09 2019. `https://www.researchgate.net/publication/336073881_Individual_claims_reserving_in_creditor_protection_insurance_using_machine_learning?`

enrichId=rgreq-f17f25e57bc87b6ac0ff416274855b0b-XXX&enrichSource=Y292ZXJQYWdlOzMzNjA3Mzg4MTtBUzo4MDc0OTAzOTY2MzUxMzdAMTU2OTUzMjEyMTczNQ%3D%3D&el=1_x_3&_esc=publicationCoverPdf.

[28] Gary G Venter. Tails of copulas. *Proceedings of the Casualty Actuarial Society*, 89(171): 68–113, 2002.

[29] R. W. M. Wedderburn. Quasi-likelihood functions, generalized linear models and the Gauss-Newton method. *Biometrika*, 61:439–447, 1971.

[30] Mario V. Wüthrich. Non-life insurance: Mathematics & Statistics. *Available at SSRN 2319328*, 2019.

[31] Mario V. Wüthrich, Hans Bühlmann, and Hansjörg Furrer. *Market-Consistent Actuarial Valuation*. Springer, Heidelberg, Switzerland, 2nd edition, 2010.

[32] Xiao Bing Zhao, Xian Zhou, and Jing Long Wang. Semiparametric model for prediction of individual claim loss reserving. *Insurance: Mathematics and Economics*, 45:1–8, 08 2009. doi: 10.1016/j.insmatheco.2009.02.009.

Appendix

Listing 4.1: R Code to Generate the Simulated Data

```
################################################
mu = 1000000
sigma = 0.02
#Ultimate amount
U <- rlnorm(10000, log(mu), sd = sigma )
#Year
t <- seq(1,20,1)

################################################
##Sample 1 equation: short-tailed samples
short_mu1 <- (1 - exp(1)^(- ((t + 1)/2) ) )^1.5
short_mu2 <- 2 * exp(1)^(- ((t-1.6)/5)^2 )

##Sample 2 equation: long-tailed samples
long_mu1 <- ( 1 - exp(1)^(- ((t + 3)/6) ) )^3
long_mu2 <- 2 * exp(1)^(- ((t-2)/5)^0.6 )
################################################
#Copula needs to be added here to have dependency between Fp and Fo
### Frank copula
#install.packages("copula")
#install.packages("VineCopula")
```

```
library(copula)
set.seed(100)
#build the ralation
library(VineCopula)
################################################
#for Sample 1
################################################
my_dist <- mvdc(frankCopula(param = 1, dim = 2), margins = c("lnorm","lnorm
   "), paramMargins = list(list(meanlog = log(short_mu1), sdlog = sigma^2),
    list(meanlog = log(short_mu2), sdlog = sigma^2 )))
short_term_paids <- data.frame()
short_term_outstandings <- data.frame()
for(i in 1:6000)
{
   set.seed(i)
   new_data <- rMvdc(20, my_dist)
   new_data <- t(new_data)
   short_term_paids <- rbind(short_term_paids, U[i]*new_data[1,])
   short_term_outstandings <- rbind(short_term_outstandings, U[i]*new_data
      [2,])
}
colnames(short_term_paids) <- c
   (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)
colnames(short_term_outstandings) <- c
   (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)
################################################
#for Sample 2
################################################
my_dist2 <- mvdc(frankCopula(param = 1, dim = 2), margins = c("lnorm","lnorm
   "),
               paramMargins = list(list(meanlog = log(long_mu1), sdlog =
                  sigma^2), list(meanlog = log(long_mu2), sdlog = sigma^2 ))
                  )
long_term_paids <- data.frame()
long_term_outstandings <- data.frame()
for(i in 1:4000)
{
  set.seed(i+6000)
  new_data <- rMvdc(20, my_dist2)
  new_data <- t(new_data)
  long_term_paids <- rbind(long_term_paids, U[i+6000]*new_data[1,])
  long_term_outstandings <- rbind(long_term_outstandings, U[i+6000]*new_data
     [2,])
}
colnames(long_term_paids) <- c
   (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)
colnames(long_term_outstandings) <- c
   (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19)
################################################
#Sample 3
################################################
Sample3_paids <- data.frame()
Sample3_outstandings <- data.frame()
#500 claims (300 short-tailed + 200 long-tailed) a year
#20 years in total
for (k in 1:20)
{
  Sample3_paids <- rbind(Sample3_paids, short_term_paids[ (300*(k-1)+1) :
     (300*k),])
  Sample3_paids <- rbind(Sample3_paids, long_term_paids[ (200*(k-1)+1) :
      (200*k),])
}

for (k in 1:20)
{
  Sample3_outstandings <- rbind(Sample3_outstandings,
      short_term_outstandings[ (300*(k-1)+1) : (300*k),])
  Sample3_outstandings <- rbind(Sample3_outstandings, long_term_outstandings
      [ (200*(k-1)+1) : (200*k),])
}

write.csv(Sample3_paids, "~/Sample3_paids.csv")
write.csv(Sample3_outstandings, "~/Sample3_outstandings.csv")

################################################
#Sample 4
################################################
Sample4_paids <- data.frame()
Sample4_outstandings <- data.frame()
a <- seq(15, 585, 30)
pre <- 0
for (i in 1:20)
{
  addon <- a[i]
  A[i] <- a[i] + pre
  pre <- pre + a[i]
}
b <- seq(390,10,-20)
```

```
pre <- 0
for (i in 1:20)
{
  addon <- b[i]
  B[i] <- b[i] + pre
  pre <- pre + b[i]
}
A <- as.integer(A)
B <- as.integer(B)
Sample4_paids <- rbind(Sample4_paids, short_term_paids[ ( 1 ) : ( A[1] ),])
Sample4_paids <- rbind(Sample4_paids, short_term_paids[ ( 1 ) : ( B[1] ) ,])
for (k in 1:19)
{

  Sample4_paids <- rbind(Sample4_paids, short_term_paids[ (A[k] + 1 ) : ( A[
      k+1] ),])
  Sample4_paids <- rbind(Sample4_paids, short_term_paids[ (B[k]+1 ) : ( B[k
      +1] ) ,])
}



Sample4_outstandings <- rbind(Sample4_outstandings, short_term_outstandings[
     ( 1 ) : ( A[1] ),])
Sample4_outstandings <- rbind(Sample4_outstandings, long_term_outstandings[
    ( 1 ) : ( B[1] ) ,])
for (k in 1:19)
{
  Sample4_outstandings <- rbind(Sample4_outstandings,
      short_term_outstandings[ ( A[k] + 1 ) : (A[k+1] ),])
  Sample4_outstandings <- rbind(Sample4_outstandings, long_term_outstandings
      [ ( B[k]+1 ) : ( B[k+1] ) ,])
}
#claims in each year varies
#20 years in total
Sample4_paids
Sample4_outstandings

write.csv(Sample4_paids, "~/Sample4_paids.csv")
write.csv(Sample4_outstandings, "~/Sample4_outstandings.csv")

################################################
#Sample 5
################################################
Sample5_paids <- data.frame()
Sample5_outstandings <- data.frame()
#claims in each year varies
#20 years in total
a <- c(280,280,280,280,280,
       280,280,280,280,280,
       280,280,280,280,280,
       280,280,280,460,500)
pre <- 0
for (i in 1:20)
{
  addon <- a[i]
  A[i] <- a[i] + pre
  pre <- pre + a[i]
}
```

```
b <- c(220,220,220,220,220,
       220,220,220,220,220,
       220,220,220,220,220,
       220,220,220,40,0)
pre <- 0
for (i in 1:20)
{
  addon <- b[i]
  B[i] <- b[i] + pre
  pre <- pre + b[i]
}
A <- as.integer(A)
B <- as.integer(B)
A;B
#
Sample5_paids <- rbind(Sample5_paids, short_term_paids[ ( 1 ) : ( A[1] ) ,])
Sample5_paids <- rbind(Sample5_paids, long_term_paids[ ( 1 ) : ( B[1] ) ,])
for (k in 1:18)
{
  Sample5_paids <- rbind(Sample5_paids, short_term_paids[ (A[k] + 1 ) : ( A[
      k+1] ),])
  Sample5_paids <- rbind(Sample5_paids, long_term_paids[ (B[k] + 1 ) : ( B[k
      +1]) ,])
}
Sample5_paids <- rbind(Sample5_paids, short_term_paids[ (A[19] + 1 ) : ( A
    [20] ),])

Sample5_outstandings <- rbind(Sample5_outstandings, short_term_outstandings[
     ( 1 ) : ( A[1] ),])
```

```
Sample5_outstandings <- rbind(Sample5_outstandings, long_term_outstandings[
   ( 1 ) : ( B[1] ) ,])
for (k in 1:18)
{
  Sample5_outstandings <- rbind(Sample5_outstandings,
     short_term_outstandings[ ( A[k] + 1 ) : (A[k+1] ),])
  Sample5_outstandings <- rbind(Sample5_outstandings, long_term_outstandings
     [ ( B[k]+1 ) : ( B[k+1] ) ,])
}
Sample5_outstandings <- rbind(Sample5_outstandings, short_term_outstandings[
    (A[19] + 1 ) : ( A[20] ),])
```

Listing 4.2: R Code for CL and ODP Methods

```
library("ChainLadder")
sample3_agg_paids <- read.csv( "~/sample3_agg_paids.csv", header = T, sep=
   ",")
sample3_acc_paids <- sample3_agg_paids[,2:21]
Cumulative.Paid <- sample3_acc_paids
Incremetal.Paid <- as.data.frame.array(cum2incr(Cumulative.Paid))

#############
#CL model first
Cumulative.Paid <- as.data.frame.array(incr2cum(Incremetal.Paid))
M <- MackChainLadder(Cumulative.Paid, est.sigma="Mack")
f <- as.numeric(M$f)
alpha <- as.numeric(M$FullTriangle[,20])

#############
# ODP Mack Model
# Build the model with X|D ~ ODP (mu, phi)
```

```
Incremetal.Paid <- sample3_inc_paids

claims <- as.vector(t(Incremetal.Paid)) # Incremental paid as vector
n.origin <- nrow(Incremetal.Paid)
n.dev <- ncol(Incremetal.Paid)
origin <- factor(row <- rep(1:n.origin, n.dev)) # accident year as vector
dev <- factor(col <- rep(1:n.dev, each=n.origin)) # development year as
   vector
W <- data.frame(claims=claims, origin=origin, dev=dev)

model <- glm(claims ~ origin + dev, family = quasipoisson(), subset=!is.na(
   claims), data=W)
Model.Summary <- summary(model) # summary of glm model
Table.3 <- Model.Summary$coefficients[,c(’Estimate’,’Std. Error’)]
rownames(Table.3) <- c(’c’,paste(’a’,2:n.origin,sep=""),paste(’b’,1:(n.
   origin-1),sep=""))
colnames(Table.3) <- c(’Estimate’,’Std. Error’)
Table.3

#####rMSEP ultimate
library(ChainLadder)
Cumulative.Paid <- incr2cum(Incremetal.Paid)
set.seed(15870)
BS <- BootChainLadder(Cumulative.Paid,R=999, process.distr=c("od.pois"))
CDR.BS <- CDR(BS)

indexes <- which(as.numeric(origin)+as.numeric(dev)>n.origin+1)
origin.hat <- as.numeric(origin[indexes])
dev.hat <- as.numeric(dev[indexes])
```

```
X.hat <- matrix(0, nrow=length(indexes), ncol=n.origin+n.dev-1)
X.hat[,1] <- 1

for(i in 1:length(indexes)){
  X.hat[i,origin.hat[i]] <- 1
  X.hat[i,n.origin-1+dev.hat[i]] <- 1 }

Estimate.Parameters <- Model.Summary$coefficients[,’Estimate’]
Linear.Predictor <- X.hat%*%Estimate.Parameters
Estimate.Reserve <- exp(Linear.Predictor)

R.i.hat <- tapply(Estimate.Reserve,origin.hat,sum)

phi <- Model.Summary$dispersion
Process.Variance.R.i <- phi*tapply(Estimate.Reserve,origin.hat,sum) # Eq. 31

Var.beta <-Model.Summary$cov.scaled
Var.eta <- X.hat%*%Var.beta%*%t(X.hat) # Eq. 28
Parameters.Variance.R.i <- c() # Eq. 36
for(i in 2:n.origin){
  indexes <- which(origin.hat==i)
  Parameters.Variance.R.i[i-1] <- t(Estimate.Reserve[indexes])%*%Var.eta[
     indexes,indexes]%*%Estimate.Reserve[indexes] }

MSEP.R.i <- Process.Variance.R.i + Parameters.Variance.R.i # Eq. 30
rMSEP.R.i <- sqrt(MSEP.R.i)

MSEP.R <- sum(Process.Variance.R.i) + t(Estimate.Reserve)%*%Var.eta%*%
   Estimate.Reserve # Eq. 39
rMSEP.R <- sqrt(MSEP.R)
```

```
Table.4 <- matrix(0, nrow=n.origin+1, ncol= 6)
colnames(Table.4) <- c('R.BS','R.CT','delta.R','rMSEP.BS','rMSEP.CT','delta.
   rMSEP')

Table.4[,'R.BS'] <- CDR.BS[,'IBNR']
Table.4[,'R.CT'] <- c(0, R.i.hat, sum(R.i.hat))
Table.4[,'delta.R'] <-Table.4[,'R.CT'] / Table.4[,'R.BS'] -1
Table.4[,'rMSEP.BS'] <- CDR.BS[,'IBNR.S.E']
Table.4[,'rMSEP.CT'] <- c(0,rMSEP.R.i,rMSEP.R)
Table.4[,'delta.rMSEP'] <- Table.4[,'rMSEP.CT']/Table.4[,'rMSEP.BS']-1
#######

data.frame(CDR.BS[,'IBNR'])
```

Listing 4.3: R Code for ANN Method

```
#########################################
#Write a Loop to do the ANN
#########################################
fill_part_sample3_paids <- part_sample3_paids
#normalize the data
for (j in 1:20)
{
  fill_part_sample3_paids[,j] <- normalizeD(fill_part_sample3_paids[,j], min
     (fill_part_sample3_paids[,j], na.rm=TRUE),
                                        max(fill_part_sample3_paids[,j],na.
                                           rm=TRUE) )
}

set.seed(101)
```

```
for (i in 1:19) #19 development years
{
   nn <- mlp(fill_part_sample3_paids[ (1:(10000 - 500*i)) , (1:i)],
      fill_part_sample3_paids[ (1: (10000 - 500*i)) , (i+1)], size = c(2),
      maxit = 100,
            initFunc = "Randomize_Weights", initFuncParams = c(-0.3, 0.3),
            learnFunc = "SCG", learnFuncParams = c(0.2, 0),
            hiddenActFunc = "Act_Logistic", shufflePatterns = TRUE,
            outputActFunc = "Act_Logistic" ,
            inputsTest = NULL, targetsTest = NULL, pruneFunc = NULL,
            pruneFuncParams = NULL)
   predict <- predict(nn, data.frame(fill_part_sample3_paids[(10000 - 500*i
       + 1):(10000) , 1:i]))

   fill_part_sample3_paids[(10000 - 500*i + 1):(10000) ,i+1] <- predict
}

#denormorlize the data
for (j in 1:20)
{
  fill_part_sample3_paids[,j] <- denormalizeD(fill_part_sample3_paids[,j],
     min(part_sample3_paids[,j], na.rm=TRUE),
                                        max(part_sample3_paids[,j],na.rm=
                                           TRUE) )
}

#sum the prediction for the last year
prediction_R_nn <- vector()
for (i in 1:20)
{
  prediction_R_nn[i] <- sum(fill_part_sample3_paids[( (500*(i-1)+1) : (500*i
     ) ),20])
}
data.frame(prediction_R_nn)
```

Listing 4.4: R Code for RF Method

```
library(randomForest)
fill_part_sample3_paids_rf <- part_sample3_paids
for (j in 1:20)
{
  fill_part_sample3_paids_rf[,j] <- normalizeD(fill_part_sample3_paids_rf[,j
     ], min(fill_part_sample3_paids_rf[,j], na.rm=TRUE),
                                        max(fill_part_sample3_paids_rf[,j],
                                           na.rm=TRUE) )
}
set.seed(102)
#RF algorithm - loop - revised cascading
for (i in 1:19)
{
  for (j in 1:i)
  {
    rf <- randomForest( fill_part_sample3_paids_rf[ (1:(500*i)) , 1:(20-i +
       (j-1) )], fill_part_sample3_paids_rf[(1:(500*i)), (21-i + (j-1))],
       mtry=3,importance=TRUE)
    predict_rf <- predict(rf, fill_part_sample3_paids_rf[(1+500*i):(500*(i
       +1)),1:(20-i+(j-1))], type="response", norm.votes=TRUE,
                        predict.all=FALSE, proximity=FALSE, nodes=FALSE)
    fill_part_sample3_paids_rf[(1+500*i):(500*(i+1)), (20-i+j)] <-
       predict_rf
  }
```

```
}


for (j in 2:j)
{
last_part <- fill_part_sample3_paids_rf[ (1:(500*i)) , 1:(21-i + (j-1) )]
last_predict <- data.frame(fill_part_sample3_paids_rf[(1+500*i):(500*(i+1))
    ,1:(20-i+(j-1))])
colnames(last_predict) <- colnames(last_part)[1:j]


rf <- randomForest( X1 ~ X0 , data = last_part , mtry=1,importance=TRUE)
predict_rf <- predict(rf, last_predict , type="response", norm.votes=TRUE,
                    predict.all=FALSE, proximity=FALSE, nodes=FALSE)
fill_part_sample3_paids_rf[(1+500*i):(500*(i+1)), (20-i+j)] <- predict_rf
}
for (j in 19:i)
{
  rf <- randomForest( fill_part_sample3_paids_rf[ (1:(500*i)) , 1:(20-i + (j
      -1) )], fill_part_sample3_paids_rf[(1:(500*i)), (21-i + (j-1))], mtry
      =2,importance=TRUE)
  predict_rf <- predict(rf, fill_part_sample3_paids_rf[(1+500*i):(500*(i+1))
      ,1:(20-i+(j-1))], type="response", norm.votes=TRUE,
                      predict.all=FALSE, proximity=FALSE, nodes=FALSE)
  fill_part_sample3_paids_rf[(1+500*i):(500*(i+1)), (20-i+j)] <- predict_rf
}
for (j in 1:20)
{
  fill_part_sample3_paids_rf[,j] <- denormalizeD(fill_part_sample3_paids_rf
      [,j], min(part_sample3_paids[,j], na.rm=TRUE),
                                          max(part_sample3_paids[,j],na.rm=
                                              TRUE) )
```

```
}
#sum the prediction for the last year
prediction_R_rf <- vector()
for (i in 1:20)
{
  prediction_R_rf[i] <- sum(fill_part_sample3_paids_rf[( (500*(i-1)+1) :
      (500*i) ),20])
}
data.frame(prediction_R_rf)
```