Matthias Allmannsberger, Dr. Wolfgang Kreitmeier und Elisabeth Tercero

Micro-level Reserving – Implementation einzelschadenbasierter Reservierungsverfahren

Einleitung

Die Berechnung von Spätschadenreserven in Versicherungsunternehmen zählt zu den entscheidenden aktuariellen Prozessen mit sehr großer betriebswirtschaftlicher Bedeutung und Auswirkung.

Die versicherungsmathematischen Rechenverfahren sind demzufolge sehr zahl- und variantenreich, fußen aber in der Regel auf aggregierten Daten. Treibende Einflussfaktoren für die Höhe der Gesamtschadenreserve sind so nur sehr bedingt analysierbar, wie bereits an früherer Stelle erläutert wurde, vgl. „Der Aktuar" 4/2018, „Actuarial Data Analytics – der Weg zur Einzelschadenreservierung".

Auch wenn keine Daten über Einflussfaktoren vorliegen, so besteht trotzdem die Möglichkeit, auf nicht aggregierten Einzelschadendaten Reservierungsberechnungen durchzuführen.

Im vorliegenden Artikel wird aufgezeigt, wie die Implementation von solchen einzelschadenbasierten Reservierungsverfahren konkret bewerkstelligt und umgesetzt wurde. Nach einer konzeptionellen Vorstellung der verwendeten Verfahren werden die erhaltenen Rechenergebnisse mit denen aus klassischen Ansätzen verglichen. Es wird auch auf die verwendeten Frameworks und IT-Ökosysteme erläuternd eingegangen. In einem abschließenden Fazit werden die gewonnenen Ergebnisse eingeordnet und ein Ausblick gegeben.

Herkömmliche/ traditionelle Verfahren

In der täglichen Praxis wird die Berechnung von Spätschadenreserven auf aggregierter Ebene mit verschiedensten Verfahren durchgeführt. Einen sehr guten Überblick über die einzelnen Methoden liefert die Monografie von Wüthrich und Merz [WuM2008]. Ebenfalls umfassende und eher praxisorientierte Abhandlungen stellen die Handbücher von Radtke und Schmidt [RaS2012] bzw. Hindley [Hin2017] dar. Die zum Einsatz kommenden Verfahren können grob nach verteilungsfreien und verteilungsbasierten Ansätzen klassifiziert werden. Die unterliegenden stochastischen Modelle in der zweiten Verfahrensklasse entstammen verschiedensten Ansätzen. Es kommen allgemeine Regressionsansätze, Bayessche Methodiken oder auch Zeitreihenanalysetechniken vor.

All diesen Verfahren ist gemein, dass sie auf Abwicklungsdreiecken aufsetzen, die aus der Aggregation möglichst homogener Risikobestände entstehen. Bereits Mack [Mac1997, Kapitel 3.1.6] weist auf die Problematik hin, dass diese Aggregation aufgrund der allgemeinen Nichtadditivität der Spätschadenreserve großen Einfluss auf die Berechnung hat, jedoch ohne mathematische Modellbildung, sondern auf Grundlage von Domänenwissen erfolgt. Des Weiteren müssen Spätschadenreserven auf die dem Abwicklungsdreieck unterliegenden Einzelbestände oft rückverteilt werden, um Folgeprozesse zu unterstützen, z. B. Retrozession. A priori ist jedoch nicht klar, an Hand welcher Größe diese Rückverteilung erfolgen soll. Oft wird mithilfe des Prämienbeitrags auf die einzelnen unterliegenden Risiken aufgeteilt, was häufig zu betragsmäßig inadäquaten Reservesplittings führt.

In jüngster Zeit ist der Trend zu erkennen, das klassische Verfahren so erweitert bzw. verändert werden, dass die Berücksichtigung von dem Abwicklungsdreieck unterliegender bzw. zusätzlicher Information möglich wird. Als ein Beispiel sei das *Double-Chain-Ladder*-Verfahren genannt [MNV2012], bei dem neben dem Betragsdreieck auch ein Abwicklungsdreieck für die Schadenzahl in die Modellbildung einfließt. Sowohl für die klassischen Verfahren als auch das *Double-Chain-Ladder*-Verfahren gibt es umfangreiche Pakete in der Programmiersprache R, vgl. [chainladder, DCL].

Machine-Learning-Verfahren auf Einzelschadenbasis

Die erwähnten Nachteile einer aggregierten Verfahrensweise lassen sich weitestgehend vermeiden, wenn auf Ebene der Einzelschäden operiert wird.

Verfahrensklasse Deep Learning

Zarkadoulas [Zar2017] hat in diesem Kontext den Einsatz von *Deep Learning* untersucht. Dabei werden Künstliche Neuronale Netze anhand eines Abwicklungsdreiecks kaskadenförmig aufgebaut.

Betrachtet man einzelne Schäden über eine Zeitspanne von $t \in \mathbb{N}, t > 0$ Jahren, so bezeichnet $i \in \{1, \dots, t\}$ deren Anfalljahr, $m_i \in \mathbb{N}$ deren Anzahl im Anfalljahr i und $j \in \{1, \dots, t\}$ das Abwicklungsjahr zu dessen Ende sie betrachtet werden. Man beachte, dass für das Anfalljahr i und einen Schaden $m \in \{1, \dots, m_i\}$ die Schadensentwicklungen nur bis zum Ende eines Abwicklungsjahrs $j \leq t - i + 1$ bekannt sind. Der aktuelle Stand der bis dato geleisteten kumulierten Zahlungen wird mit *paid* benannt und der ausstehende Reservebetrag mit *outstanding*. Im Falle von echten Spätschäden werden diese mit Nullbeträgen zu den Zeiten geführt, in denen sie nicht vorliegen.

Als Daten- bzw. Prädiktionsgrundlage $x_{i,j}^m$ können sowohl die *paid*-Beträge, als auch die *incurred = paid + outstanding*-Beträge dienen und für ein $k \in \{t-i+2, \dots, t\}, k > 1$ definiere $y_{i,k}^m$ eine noch unbekannte Schadensentwicklung im Abwicklungsjahr k. Fasst man nun alle bekannten Schäden bzw. unbekannten Schadensentwicklungen pro Anfalljahr i zusammen zu $x_{i,j} \coloneqq (x_{i,j}^1, \dots, x_{i,j}^{m_i})$ bzw. $y_{i,k} \coloneqq (y_{i,k}^1, \dots, y_{i,k}^{m_i})$, so ergibt sich der in Abbildung 1 gezeigte Ablauf für die kaskadenförmige Vorhersage der Schadensentwicklung.

Die Konstruktion bzw. das Training der neuronalen Netze benutzt in jeder Stufe die bereits bekannten Werte des Schadendreiecks. Bei der Prädiktion gehen bekannte, aber auch die errechneten Werte aus vorherigen Stufen mit ein.

In den durchgeführten Experimenten folgen die einzelnen Netzwerke *(feedforward neural networks)* dabei einer gleichbleibenden Architektur, bestehend aus einer *verdeckten Schicht* umrahmt von *Eingangs- und Ausgabeschicht*. Einzig der Aufbau der *Eingangsschicht* selbst variiert, da die Anzahl der Eingangsknoten (oder: Eingangsneuronen) identisch ist mit der Anzahl der Abwicklungsjahre in der Trainingsmenge.

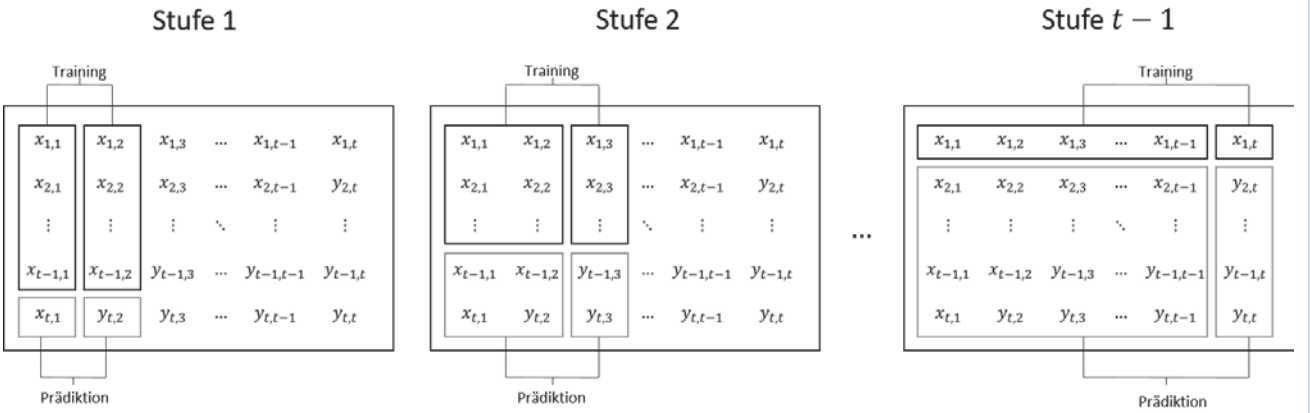
Für eine detaillierte Beschreibung der Funktionsweise und der Berechnungen eines Neuronalen Netzes sei auf Goodfellow et al. [GoBeCo2017] verwiesen. Erwähnt sei, dass sich hinter den einzelnen Knoten der *verdeckten Schicht* Gewichte befinden, die mit den Schadensvektoren aus der *Eingangsschicht* in einer Linearkombination und der nachfolgenden Anwendung einer sog. *Aktivierungsfunktion* zur Ausgabeschicht weitergeleitet werden. Das Trainieren des Modells erfolgt nach dem Prinzip der *Backpropagation*. Dabei werden die Gewichte initial zufällig erzeugt und dann in einem iterativen Vorgehen mithilfe eines Gradienten-Verfahrens anhand des quadratischen Fehlers der Vorhersage rückwirkend – d. h. von der Ausgabe- in Richtung der Eingabeschicht – optimiert.

In der zugrunde liegenden Implementierung wurden sowohl die Verfahren *Stochastic Gradient Descent* als auch *Adam Optimizer* genutzt, um diesen Fehler durch eine entsprechende Gewichtsanpassung zu minimieren. In der Dokumentation [KerasOpt] werden diese und weitere, ähnliche Algorithmen detailliert beschrieben. Ebenfalls kommt ein kreuzvalidierter Ansatz zur Überwachung des Trainingsfortschritts bzw. zur Bestimmung der Güte des Modells zum Einsatz. Dabei wird eine zufällige Teilmenge der Schadensdaten aus einem Trainingsdurchgang ausgeschlossen und anschließend der quadratische Fehler auf dieser Menge bestimmt.

Verfahrensklasse k-nearest neighbour

Betrachtet man den Trainings- und Prädiktionsprozess der einzelnen *Deep-Learning*-Modelle, so lassen sich diese sehr generisch in die kaskadenförmige Struktur einbetten. Dieses verallgemeinerte Konstrukt ermöglicht ebenfalls den beliebigen Austausch der zugrunde liegenden Machine-Learning-Modelle. Exemplarisch wurde dies anhand eines alternativen Verfahrens zur Einzelschadenprädiktion (cf. [Mac1997, Kapitel 3.4.5]), das einer *k-nearest-neighbour*-Methode [FrHaTi2001] folgt, gezeigt. Dabei wird der einzelne Schaden mithilfe des Verlaufs ähnlicher bekannter Schäden fortgesetzt. Die paid- und outstanding-Beträge eines fortzusetzenden Schadens werden jeweils um die gewichtete durchschnittliche Differenz ähnlicher Schäden erhöht bzw. verringert. Ein Schaden fällt dabei umso stärker ins Gewicht, je geringer sein euklidischer Abstand bezüglich der (paid,outstanding)-Vektoren zu dem fortzusetzenden ist. Aber auch weitere Berechnungsarten der Fortsetzungen werden dort aufgeführt, wie die (ungewichtete) Durchschnittsbildung, Minimum- oder Maximumwerte, die sowohl zu einer additiven Vorhersage, als auch zu einer multi-

Abbildung 1:
Stufen des kaskadenförmigen Vorhersagemodells



plikativen führen können. Für die Definition eines Ähnlichkeitsbegriffs gibt es vielfältige Optionen. Mack [Mac1997] gibt unter anderem eine Möglichkeit an, Schäden zu charakterisieren und sie so in ähnlichen Teilmengen zusammenzufassen. Beispielsweise werden dort Schäden als *offen, ausreguliert, wiederaufgelebt* oder *geschlossen* definiert, um so gleichzeitig auch den Schadensstatus zu modellieren bzw. deren mögliche Statusübergänge festzulegen, wie etwa, dass geschlossene Schäden geschlossen bleiben.

Kombination mit Clustering-Ansätzen

Abhängig von der Anzahl der Schäden und weiteren Schadenparametern, sind verfeinerte Definitionen der Schadenkategorie denkbar und sinnvoll. Der vorgestellte knn-Ansatz muss hierzu kanonisch erweitert werden. Zusätzlich zu dieser deterministischen Charakterisierung ist eine weitere Gruppierung ähnlicher Schäden bzw. Schadenverläufe durch Clustermethoden, wie k-means, k-medoids [FrHaTi2001, Schuh2018] möglich. Für diese Algorithmen könnten zum Beispiel die incurred-Beträge, das paid-zu-incurred-Verhältnis im ersten Abwicklungsjahr, der Schadensstatus in Zusammenhang mit einer Distanzmatrix oder eine Kombination aus diesen Daten in Betracht gezogen werden. Die Anzahl der Cluster kann wiederum durch Experten festgelegt oder mithilfe von Heuristiken gesteuert werden. Auch Ausreißer, die die Vorhersageergebnisse verzerren, können mit diesem Clustering-Ansatz behandelt werden.

Interpretierbarkeit und Einbau von Expertenwissen

Neben diesen und zahlreichen weiteren Varianten ist auch die Integration weiterer Algorithmen, wie baumbasierte Verfahren oder verallgemeinerte Regressionsmodelle [Carr2019], in die kaskadenförmige Einzelschadensvorhersage denkbar. Wo beispielsweise der *Deep-Learning*-Ansatz zu komplexeren und somit aber auch weniger interpretierbaren Berechnungen fähig ist, ist *k-nearest-neighbour* greifbarer und lässt mehr Eingriffsmöglichkeiten für den Einbau von Expertenwissen (Art und Anzahl von Schadenkategorien, Parameter für die Mittelwertsbildung etc.) zu. Um eine möglichst objektive Sichtweise zu wahren, ist das Erstellen und Gegenüberstellen der Vorhersagen aus verschiedenen Modellen daher dringend zu empfehlen.

Vergleich der Ergebnisse

Die Umsetzung der beiden Machine-Learning-Verfahren erfolgte mit künstlich erzeugten Schadendaten und Echtdaten aus einem Kraftfahrthaftpflichtportfolio.

Für die Erzeugung der künstlichen Schadendaten folgte man der in [Zar2017] beschriebenen Prozedur zur Erzeugung zufällig generierter

Abbildung 2:
Ultimates der Anfalljahre 2008 bis 2018

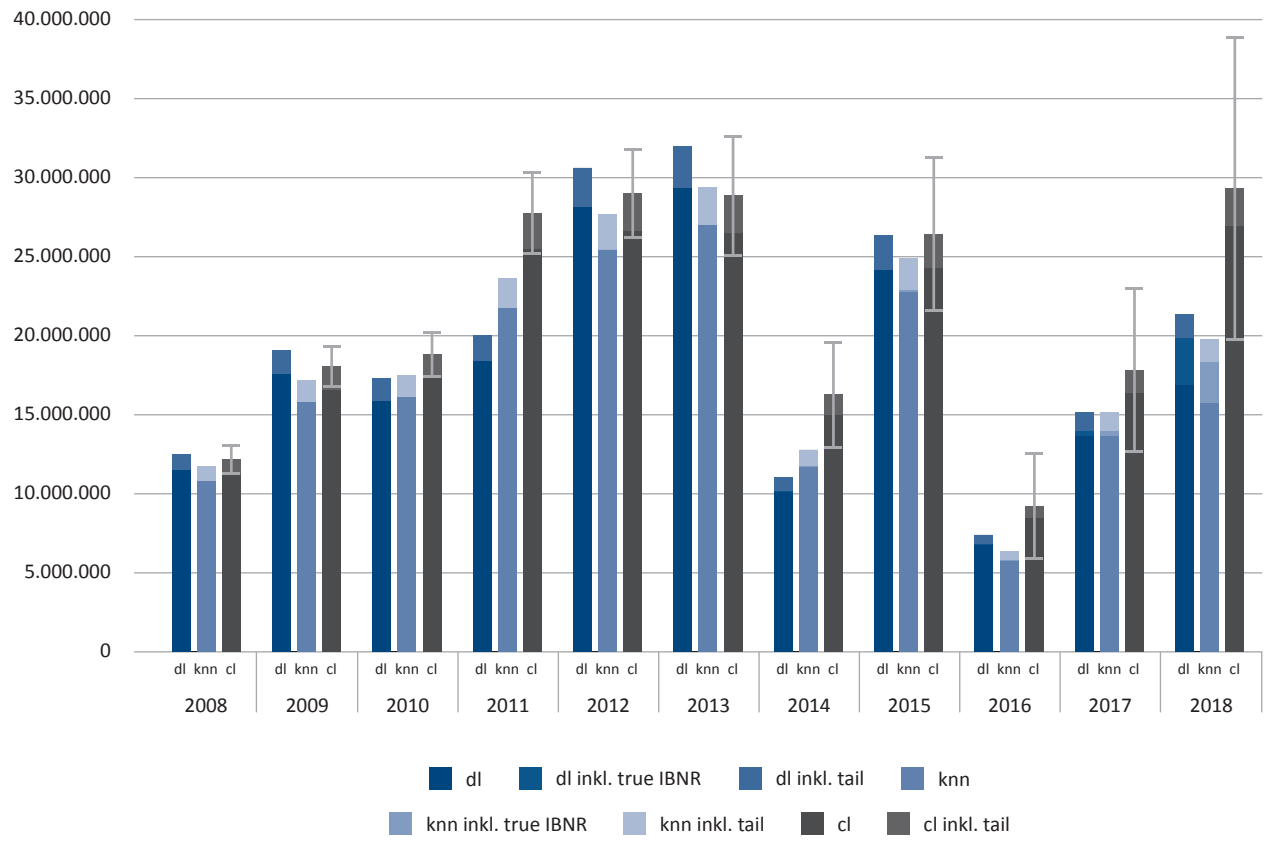
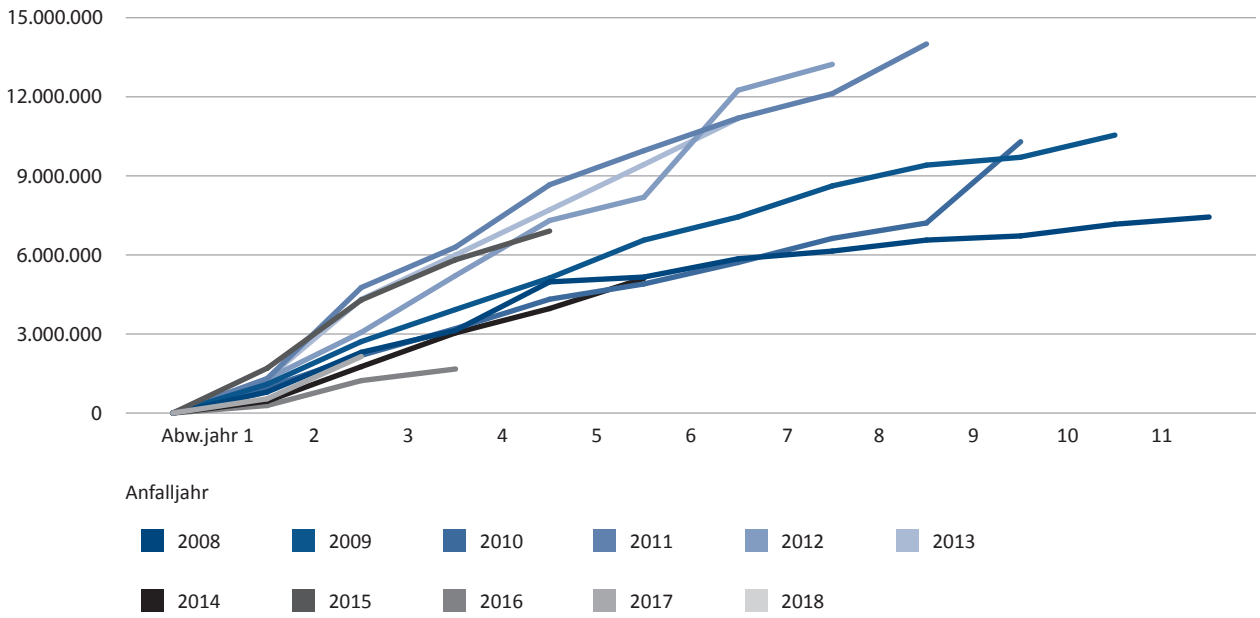


Abbildung 3:
Abwicklung (paid) der Anfalljahre 2008 bis 2018



Schadenswerte, die sowohl in Art und Anzahl verschieden ausgeprägt werden können. Da mit den dort beschriebenen Verteilungen auch gleichzeitig die Schadensfortsetzungen erstellt werden, also $x_{i,j}$ für sämtliche $i, j \in \{1, \ldots, t\}$ bekannt sind, wurde die Vorhersagequalität der Verfahren unter anderem daran gemessen.

Für die Evaluierung wurden speziell die Prädiktionen für das letzte Abwicklungsjahr, ultimates, in aggregierter Form betrachtet. Dies ermöglicht zusätzlich den Vergleich mit den eingangs erwähnten herkömmlichen Verfahren, wobei diese ohne manuelle Eingriffe, rein mechanistisch ausgeführt wurden. Hier zeigt sich, dass die *Deep-Learning-* und *k-nearest-neighbour*-Methode mit ihren Vorhersagen näher an den tatsächlichen Werten sind als *(Double) Chain Ladder*. Vermutlich gibt es aber sicherlich andersartig künstlich erzeugte Daten und weitere Konstellationen, bei denen ein gegenteiliges Resultat der Fall ist.

Für die Untersuchung der Echtdaten wurden die Berechnungen für die Anfalljahre 1997 bis 2018 angewandt, wobei hierzu die Vorhersage noch um eine Schätzung der echten Spätschäden sowie die Tailabwicklung erweitert wurde. Für die Spätschadenreserve wird Bezug genommen auf [Mac1997, Kapitel 3.4.5], da auch Informationen zum Volumenmaß, d. h. Jahreseinheiten, vorliegen. Die Bestimmung des Tails bzw. des Tailfaktors folgt einschlägigen Referenzen (siehe z. B. [CASTail2013]). Die Grafik in Abbildung 2 zeigt die ultimates der letzten 11 Anfalljahre für die einzelnen Verfahren (*Chain Ladder* = cl, *Deep Learning* = dl, *k-nearest-neighbour* = knn). Zusätzlich wurden hier noch die Schätzung der echten Spätschäden und die Tailabwicklung integriert. Aufgrund der Datenlage, werden echte Spätschäden („true IBNR“) nur in den jüngsten fünf Anfalljahren prognostiziert. Zusätzlich, stellt die *Chain-Ladder*-Methode noch einen Standardfehler bereit, der ebenfalls in das Balkendiagramm integriert ist.

Auffällig ist, dass vor allem in den Anfalljahren 2011, 2014 und 2018, das rein mechanistisch durchgeführte Chain-Ladder-Verfahren höhere Werte prognostiziert als die auf Einzelschäden basierenden Verfahren. Betrachtet man hierzu die aggregierten Zahlungsentwicklungen der einzelnen Anfalljahre (vgl. Abbildung 3), auf denen die Vorhersagen basieren, so scheinen Kalenderjahreseffekte eine mögliche Erklärung für diese Unterschiede zu liefern.

Technologische Umsetzung

Für eine Implementierung der beiden vorgestellten Verfahren auf Einzelschadenbasis war entscheidend, einen kompakten Rechenkern zu schaffen, der sowohl alleinstehend funktioniert als auch leicht integrierbar und erweiterbar ist. Neben diesen und folgenden Kriterien, fiel der Entschluss auf R. Als Open-Source-Programmiersprache bzw. -Programmierumgebung mit dem Hauptanwendungsgebiet der statistischen Berechnungen bietet R die nötigen Voraussetzungen für die Umsetzung. Gleichzeitig erweitert eine sehr aktive Community stetig den Funktionsumfang von R mittels optionaler Pakete, die unter *CRAN – The Comprehensive R Archive Network* [CRAN] frei zugänglich sind.

So gibt es beispielsweise die Pakete *ChainLadder* [chainladder] und DCL *(Double Chain Ladder)* [DCL], die die eingangs erwähnten her-

kömmlichen Verfahren bereitstellen und demnach für eigene Implementierungen angebunden werden können.

Neben dem *k-nearest-neighbour*-Ansatz, der weitestgehend mit den Standardfunktionalitäten von R implementierbar ist, wurde die Entwicklung der *Deep-Learning*-Variante durch ein solches Paket, Keras [Keras], ermöglicht. Keras bietet eine benutzerfreundliche Schnittstelle zwischen R und TensorFlow [TenFl]. Dieses vom Google-Brain-Team bereitgestellte Open Source Framework ermöglicht die Erstellung und die Handhabung beliebiger Neuronaler Netze, sowohl die Konstruktion der Netzwerkgraphen samt Datenfluss als auch das Training und die Validierung.

Beide Verfahren wurden wiederum selbst in einer Paketstruktur gekapselt, um den angestrebten Rechenkern zu implementieren. In einem allein lauffähigen und frei verfügbaren Prototypen [prot] wurden diese dann den herkömmlichen Methoden in einer Weboberfläche gegenübergestellt und deren Ergebnisse anhand von zufällig erzeugten Daten tabellarisch sowie visuell vergleichbar gemacht. Die App selbst wurde ebenfalls in R mittels shiny [shiny] bzw. shinydashboard [shinydashboard] und plotly [plotly] realisiert, womit sich die Frage nach der Integrierbarkeit noch nicht stellte. Um auch für weitere Software-Ökosysteme eine Aussage zu treffen, wurde exemplarisch die Integration in ein SAP-HANA-System erprobt. Hier ist es möglich der SAP-HANA-Datenbank die native Ausführung der Berechnungen über einen angebundenen R Server zu erlauben. Da R Code als SQL-Prozedur seitens des SAP-Systems hinterlegt werden kann, profitiert die Anbindung ebenfalls von möglichen Performanceverbesserungen der SAP-HANA-Datenbank bei der Datenbereitstellung.

Fazit und Ausblick

Die vorliegende prototypische Implementierung hat gezeigt, dass einzelschadenbasierte Spätschadenreservierung zu einer höheren Genauigkeit der Ultimate-Schätzung führen kann, das aber nicht grundsätzlich der Fall ist bzw. sein muss. Einzelschadenbasierte Verfahren sind als Ergänzung zu bestehenden Verfahren zu sehen und dürften Vorteile bieten, wenn inhomogene Portfolios vorliegen bzw. eine (meistens inadäquate) Rückverteilung der Reserve damit vermieden werden kann.

Bei den untersuchten Echtdaten ist zu beobachten, dass *k-nearest-neighbour* und *Deep Learning* als einzelschadenbasierte Methoden Ultimate-Prognosen ähnlicher Höhe abliefern. Um einzelschadenbasierte Methoden besser einordnen und ihre Prognosegüte bewerten zu können, ist die Entwicklung eines Verfahrens zur Fehlerabschätzung ein wichtiger nächster Schritt. Dies kann z. B. durch Kreuzvalidierung versucht werden [vgl. FrHaTi2001, Kapitel 7]. Für eine umfassende Erklär- und Bewertbarkeit der vorgestellten einzelscha-

Literaturverzeichnis

[Carr2019] CARRATO Alessandro, From the Chain Ladder to Individual Claims Reserving using Machine Learning techniques, IAA Section Colloquium 2019 Cape Town ASTIN/NON-LIFE

[CASTail2013] CAS Tail Factor Working Party. The Estimation of Loss Development Factors: A Summary Report. Casualty Actuarial Society E-Forum (2013): 37–40

[chainladder] ChainLadder: Statistical Methods and Models for Claims Reserving in General Insurance – R package, URL: https://cran.r-project.org/web/packages/ChainLadder/index.html abgerufen am: 24.01.2019

[CRAN] The Comprehensive R Archive Network – CRAN. URL: https://cran.r-project.org/ abgerufen am: 24.01.2019

[DCL] DCL: Claims Reserving under the Double Chain Ladder Model – R package, URL: https://cran.r-project.org/web/packages/DCL/index.html abgerufen am: 24.01.2019

[FrHaTi2001] FRIEDMAN, Jerome; HASTIE, Trevor; TIBSHIRANI, Robert. The elements of statistical learning. New York, NY, USA:: Springer series in statistics, 2001.

[GoBeCo2017] GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. The MIT Press, 2017.

[Hin2017] HINDLEY, David. Claims Reserving in General Insurance. Cambridge University Press, 2017.

[Keras] Keras: The Python Deep Learning library. URL: https://keras.io/ abgerufen am: 24.01.2019

[KerasOpt] keras: Usage of optimizers. URL: https://keras.io/optimizers/ abgerufen am: 24.01.2019

[Mac1997] MACK, Thomas. Schadenversicherungsmathematik. Verlag Versicherungswirtsch., 1997.

[MNV2012] MARTÍNEZ, Miranda María Dolores, NIELSEN, Jens Perch and VERRALL, Richard. Double chain ladder. ASTIN Bulletin: The Journal of the IAA 42.1 (2012): 59–76

[plotly] plotly: Create Interactive Web Graphics via 'plotly.js' – R package. URL: https://cran.r-project.org/web/packages/plotly/index.html abgerufen am: 29.01.2019

[prot] Micro-level Reserving Prototype. URL: http://minnosphere.westeurope.cloudapp.azure.com/ibnr/ abgerufen am: 24.01.2019

[RaS2012] RADTKE, Michael; SCHMIDT, Klaus D. Handbuch zur Schadenreservierung. VVW GmbH, 2012.

[Schuh2018] SCHUHMANN, Steffen (SCOR) Single Loss Development Models for the Projection of Individual Large Claims for Long Tail Non-Proportional Reinsurance Pricings. ICA 2018

[shiny] shiny: Web Application Framework for R – R package. URL: https://cran.r-project.org/web/packages/shiny/index.html abgerufen am: 29.01.2019

[shinydashboard] shinydashboard: Create Dashboards with 'Shiny' – R Package. URL: https://cran.r-project.org/web/packages/shinydashboard/index.html abgerufen am: 29.01.2019

[TenFl] TensorFlow: An open source machine learning framework for everyone. URL: https://www.tensorflow.org/ abgerufen am: 24.01.2019

[WuM2008] WÜTHRICH, Mario V.; MERZ, Michael. Stochastic claims reserving methods in insurance. John Wiley & Sons, 2008.

[Zar2017] ZARKADOULAS, Andreas (2017), Neural network algorithms for the development of individual losses, University of Lausanne, online

denbasierten Prognosemethoden sind Stabilitätsuntersuchungen hilfreich, die durch deterministische oder stochastische Klassenbildung (clustering) angestellt werden können. Zum Auffinden und zum Verständnis der Mechanismen, die die Höhe und Genauigkeit der einzelschadenbasierten Ultimate-Prognose beeinflussen, bedarf es weiterer Forschung. Zum Erzielen von Fortschritten sind aktuarielles Wissen und Data-Science-Techniken gleichermaßen notwendig.

Durch Nutzung moderner frei verfügbarer Frameworks ist die Implementierung der zugrunde liegenden Machine-Learning-Algorithmen relativ einfach geworden. Auch eine Anbindung an bestehende IT-Ökosysteme der Versicherungsunternehmen ist mit diesem Ansatz möglich.

Das umgesetzte Verfahren ist dabei hochgenerisch. Neben den vorgestellten und implementierten Algorithmen kann im Prinzip jede Algorithmenfamilie, die konzeptionell für die Fortschreibung der Abwicklung geeignet ist, verwendet werden. Dem Nachteil, dass *Deep Learning* als „black box" dem anwendenden Aktuar wenig Einblick in die zugrunde liegende Rechensystematik gibt, wurde versucht hier zu begegnen, indem *knn* als weiteres Modell implementiert wurde. Bei knn als transparentem Modell hat der Anwender Eingriffsmöglichkeiten zur Kalibrierung und kann direkt mit den Ergebnissen von *Deep Learning* vergleichen, um Ergebnisse besser einordnen zu können.

Auch der Begriff des Einzelschadens ist in der Modellbildung abstrahierbar. Für ein aggregiertes Abwicklungsdreieck, das aus mehreren Schadendaten aufgebaut ist, kann das vorgestellte Modell auf die einzelnen zugrunde liegenden Daten angewandt werden, die jedoch nicht notwendigerweise Einzelschäden sein müssen.

Matthias Allmannsberger ist Senior IT Consultant und Data Scientist in der msg systems group bzw. in dem dort ansässigen Innovationslabor minnosphere. Seine Arbeitsschwerpunkte liegen in der mathematischen Analyse, Evaluierung und Implementierung von Use Cases unter Einsatz von Methoden und Technologien aus dem Bereich des maschinellen Lernens. Er studierte Mathematik und Informatik an der Universität Passau mit dem Schwerpunkt auf Stochastik bzw. stochastische Prozesse.

Dr. Wolfgang Kreitmeier ist Aktuar (DAV) und Principal IT Consultant in der msg systems group bzw. in dem dort ansässigen Innovationslabor minnosphere. Ein Schwerpunkt seiner Tätigkeit liegt im Bereich Data Science und maschinelles Lernen in der Erst- und Rückversicherung. Dabei ist er sowohl steuernd und betreuend als auch operativ im Einsatz. Herr Kreitmeier studierte Physik mit Nebenfach Mathematik an der Ludwig-Maximilians-Universität München und promovierte an der Universität Passau in Mathematik.

Elisabeth Tercero ist Senior IT Consultant und Data Scientist in der msg systems group bzw. in dem dort ansässigen Innovationslabor minnosphere. Ihr Hauptaufgabengebiet liegt in der Umsetzung von Data Science Projekten, angefangen bei der Datenanalyse bis hin zur Implementierung von Prototypen und technischen Umsetzung. Sie studierte Computerwissenschaften an der Universidad de Castilla-La Mancha und machte ihr Abschlussprojekt – Fuzzy Classifier in der Neuromedizin – mit der Hochschule Trier.