Evaluierung, Analyse und Implementierung eines Machine Learning Verfahrens zur Verbesserung von fachlichen Entscheidungen

Bachelor-Thesis

Fakultät Informatik

zur Erlangung des akademischen Grades

Bachelor of Science

vorgelegt von:	Fabian Cotic
Studiengang:	Wirtschaftsinformatik
Matrikelnummer:	288710
Erstgutachter:	Prof. Dr. Christian Johner
Zweitgutachter:	Dr. Bernd Michelberger

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht sind und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Ort, Datum, Unterschrift

Vorwort

Die vorliegende Arbeit bildet den Abschluss meines Bachelor-Studiums der Fachrichtung Wirtschaftsinformatik an der HTWG Konstanz. Diese Arbeit ist im Zeitraum vom 01. Juni 2017 bis 31. August 2017 unter der Betreuung von Herrn Prof. Dr. Christian Johner und Dr. Bernd Michelberger entstanden. Das Thema der Arbeit entstand einerseits aus meinem persönlichen Interesse an Machine Learning, sowie steigenden Kundenanfragen bezüglich Machine Learning im Vertrieb der ACTICO GmbH. Aufgrund meiner vorigen Tätigkeiten im Unternehmen, kam mir die Ehre zuteil mein Thema selbst zu wählen und Mithilfe meines Betreuers Dr. Bernd Michelberger auszugestalten.

Johner ...

Bernd ...

ACTICO ...

Korrekturleser ...

Zu Letzt möchte ich mich bei meinen Eltern bedanken, die mir während meines Studiums immer liebevoll mit Rat und Tat zur Seite standen und mich finanziell unterstützen um mir so mein Studium erst zu ermöglichen.

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Abkürzungsverzeichnis

BKM Business Knowledge Model

BPMN Business Process Model and Notation

CORBA Common Object Request Broker Architecture

DM Decision Management

DMN Decision Model and Notation

DMS Decision Management Systems

DS Decision Service

DRD Decision Requirements Diagramm

OMG Object Management Group

ML Machine Learning

UML Unified Modeling Language

Abbildungsverzeichnis

1.1. Gründe für falsch getroffene fachliche Entscheidungen. 19
1.2. Aufbau der Arbeit. 22

2.1. Der Lebenszyklus von automatisierten Entscheidungen. 25
2.2. Ein beispielhaftes DRD. 26
2.3. Beispiel einer Entscheidungstabelle zur Berechnung des Zinssatzes. 27
2.4. Vorgehensweise zur Entwickelung eines Modells. 30
2.5. Aufbau eines Perzeptrons. 32
2.6. Modell eines Perzeptrons mit Beispiel-Daten. 33
2.7. Aufbau eines neuronalen Netzwerkes mit mehreren Schichten. 34
2.8. Neuronalesnetz mit Sigmoid-Neuronen und Bias. 35

3.1. Die Hauptkomponenten des Kreditrisikos. 39
3.2. Decision Requirements Diagram der Transaktionsbetrugs-Entscheidung. . 41
3.3. Vorgehensweise bei der Erstellung des Lerndatensatzes. 43

Tabellenverzeichnis

1.1. Forschungsfragen entlang der Kapitel. 21

2.1. Beispielhafter Aufbau der Trainingsdaten. 31

3.1. Zerlegung des kategorischen Bundesstaat-Attributs in Features. 44

Inhaltsverzeichnis

Abkürzungsverzeichnis **9**

Abbildungsverzeichnis **11**

Tabellenverzeichnis **13**

1. Motivation **17**
1.1. Problemstellung 18
1.2. Zielsetzung 20
1.3. Abgrenzung 21
1.4. Aufbau der Arbeit 22

2. Grundlagen **25**
2.1. Decision Management 25
2.2. Machine Learning 30
2.3. Technologien 34
2.3.1. Decision Management 35
2.3.2. Machine Learning 35

3. Konzeption **37**
3.1. Definition von Anwendungsfällen 37
3.1.1. Erkennung von Kreditausfällen 38
3.1.2. Erkennung von Betrugsversuchen 40
3.2. Model zur Erkennung von Kreditausfällen 42
3.2.1. Lerndaten 42
3.2.2. Modellbildung 46
3.3. Modell zur Erkennung von Betrugsversuchen 50
3.3.1. Lerndaten 50
3.3.2. Modellbildung 50

4. Implementierung **51**
4.1. Datenschema 51
4.2. Decision-Engine 51
4.3. Neuronales-Netzwerk 52
4.4. Optimierung 52

5. Analyse **53**
5.1. Vorgehensweise 53
5.2. Erkennung von Kreditausfällen 53
5.3. Erkennung von Betrugsversuchen bei Banktransaktionen 53
5.4. Zusammenfassung der Ergebnisse 53

6. Diskussion **55**

7. Zusammenfassung und Ausblick **57**

Literaturverzeichnis **59**

A. Anhang **65**

1. Motivation

Unternehmen treffen unzählige Entscheidungen jeden Tag. Wertschöpfende Entscheidungen tragen, wie ihr Name schon sagt, zur Wertschöpfung bei und beeinflussen somit unmittelbar den wirtschaftlichen Erfolg eines Unternehmens. Eine Bank beispielsweise muss einen Kreditnehmer auf dessen Fähigkeit, Raten zurückzahlen zu können, prüfen. Somit soll sichergestellt werden, dass die Bank den Betrag des Kredites sowie auch die ihr zustehenden Zinsen zurückbekommt. Dadurch steht und fällt der Erfolg einer Bank mit den Krediten, die sie vergibt [16, vgl. S. 1]. Prominentestes Beispiel hierfür ist die ehemalige US-amerikanische Investmentbank Lehmann Brothers, die im Zuge der Finanzkrise 2008 Insolvenz anmelden musste [36, vgl. S. 3]. Boomende Immobilien-Preise resultierten in höheren Sicherheiten von Privathaushalten, wodurch auch Kredite an Darlehnsnehmer mit geringer Bonität vergeben wurden (Subprime-Kredit) [7, vgl. S. 82-83]. Nach dem Platzen der Immobilienblase kam es zu Zahlungsausfällen und -störungen im US-amerikanischen Hypothekenmarkt [1, vgl. S. 6 ff.], was erst in einem Finanzcrash und dann in einer Weltwirtschaftskrise resultierte [17, vgl. S. 109].

Etwa im Juni 2009 waren die schlimmsten Auswirkungen der Finanzkrise überstanden, die amerikanische Wirtschaft begang wieder zu wachsen [24, vgl. S. 21] und Banken vergaben fleißig Kredite. Heutzutage werben Banken mit Kreditentscheidungen innerhalb von wenigen Sekunden. Das Ausfüllen eines Formulars ist ausreichend, um wenige Sekunden später eine Kreditentscheidung zu erhalten. Die Prüfung der Bonität geschieht automatisiert im Hintergrund durch eine zuvor definierte Entscheidungslogik. Die Entscheidungslogik wird von Fach-Experten erstellt, wobei die Fach-Experten im Wesentlichen zwei essentielle Dinge beachten:

1. Verhinderung der Vergabe von Krediten an Personen, die diese nicht zurückzahlen können (”default rate”)

2. Verhinderung der Nicht-Vergabe von Krediten an Personen, die eigentlich in der Lage sind, diese zurück zuzahlen, wodurch Einnahmen an Konkurrenten verloren gehen ("acceptance rate")

Banken sind somit angehalten eine gute Balance zwischen Sicherheit und Gewinn zu finden. Dazu muss die fachliche Entscheidungslogik drohende Zahlungsausfälle so genau wie möglich vorhersagen. Studien wie [50] zeigen, dass sieben Prozent der vergebenen Kredite im Euroraum mehr als 90 Tage im Zahlungsverzug sind. Das entspricht einem Volumen von einer Billionen Euro. Bei einem Zahlungsverzug von über 90 Tagen wird generell davon ausgegangen, dass ein Kredit nicht mehr in voller Höhe zurückgezahlt wird und somit zu einem Verlust-Geschäft führt [5, vgl. S. 8]. Könnte man die Entscheidungslogiken bei den Banken dahingehend optimieren, ein Prozent weniger "faule" Kredite zu vergeben, könnten große Summen eingespart werden, die sonst im Zuge einer Banken-Rettung dem Steuerzahler auferlegt werden.

1.1. Problemstellung

Das Einsparen der genannten Summen wäre erstrebenswert, allerdings gibt es unzählige Gründe, die den Finanzcrash ab 2007 verschuldet haben. Als Hauptgrund wird in der Literatur die umfangreiche Vergabe von Subprime-Krediten genannt [7, 36]. Bei der Vergabe dieser Kredite wurde das Risiko nicht korrekt identifiziert [36, vgl. S. 3], was zu unzähligen falschen fachlichen Entscheidungen führte. Daraus abgeleitet ergibt sich die Problemstellung dieser Arbeit, dass zu viele fachliche Entscheidungen falsch getroffen werden. Die Gründe hierfür werden nachfolgenden beschrieben und in Abbildung 1.1 zusammengefasst.

Ein Grund für die vielen falschen Entscheidungen, sind fehlerhafte Entscheidungslogiken und falsche Annahmen (Grund G1 - siehe Abbildung 1.1). Bei der Vergabe der Kredite wurden die Entscheidungen verfälscht, da die Entscheidungslogiken mit Eingabe-Daten befüllt wurden, die nicht der Realität entsprachen. Durch die unrealistisch hohen Immobilien-Preise, waren die Kredite mit Immobilien gesichert, die real weitaus weniger wert waren [20]. Die Kredite wurden dementsprechend auf Basis von nicht vorhandenen Sicherheiten berechnet.

Ein weiterer Grund ist, dass vergangene Entscheidungen nicht mehr reflektiert wurden. (Grund G2 - siehe Abbildung 1.1). Die Kredite wurden gebündelt unter den Banken

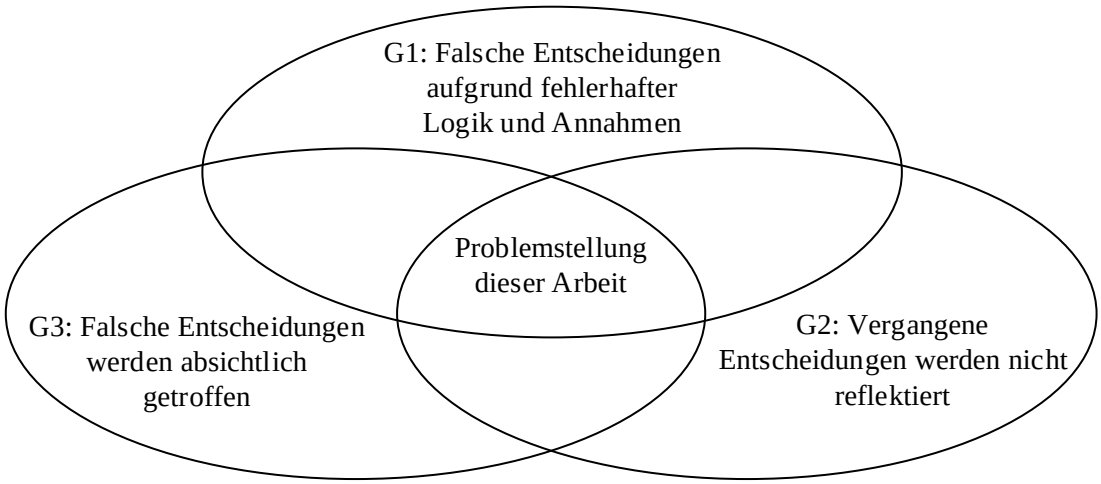

Abbildung 1.1.: Gründe für falsch getroffene fachliche Entscheidungen.

weiterverkauft [36, vgl. S. 2], ein Verlust-Risiko bestand somit nur für die kurze Zeit, in der ein Kredit-Bündel noch nicht weiterverkauft wurde. Saunders et al. sagen "The underwriters [...] had little incentive to screen and monitor the activities of borrowers for whom they originated loans" [43]. Nach der ursprünglichen Vergabe des Kredites, wurden also keine weitere Aktivitäten unternommen, um die damaligen Kredit-Entscheidungen weiterhin zu prüfen oder zu überwachen. Ein Kontrollverlust der Situation war nur eine Frage der Zeit.

Im Falle der Finanzkrise waren allerdings nicht nur fehlerhafte Entscheidungslogik, falsche Annahmen und mangelnde Reflektion schuld am Zusammenbrauch der Finanzmärkte. Absichtlich falsch getroffene Entscheidungen sind ebenso als Ursache der vielen falsch getroffenen Entscheidungen anzuführen (Grund G3 - siehe Abbildung 1.1). In einer Pressemitteilung des Vorstandes der US-Notenbank, wurden Bonus-Zahlungen an Bank-Mitarbeiter als weiterer Grund der Finanzkrise aufgezählt: "Flaws in incentive compensation practices were one of many factors contributing to the financial crisis. Inappropriate bonus or other compensation practices can incent senior executives or lower level employees, such as traders or mortgage officers, to take imprudent risks that significantly and adversely affect the firm" [18]. Mitarbeiter übertraten bewusst Risiken, nutzten mangelnde Regulationen zu ihrem Vorteil und ließen keine Möglichkeit aus ihren Gewinn auf Kosten von Stabilität und Sicherheit zu maximieren.

Zusammenfassend soll Abbildung 1.1 nochmals die Gründe aufzeigen die verantwortlich sind, dass zu viele falsche fachliche Entscheidungen getroffen werden.

1.2. Zielsetzung

Das übergeordnete Ziel ist die in 1.1 genannte Problemstellung, dass zu viele falsche fachliche Entscheidungen getroffen werden, zu verbessern, indem man die Anzahl falscher fachlichen Entscheidungen reduziert. Diese Arbeit soll hierzu ein Machine Learning (ML) Verfahren evaluieren, analysieren und implementieren, um fachliche Entscheidungen zu verbessern. Aus dieser Zielsetzung können die folgenden Teilziele abgeleitet werden.

Das *erste Teilziel* ist das Erstellen und die kontinuierliche Optimierung von Modellen, anhand von vergangen fachlichen Entscheidungen und deren Korrektheit. Dazu soll im ersten Schritt ein ML-Verfahren implementiert werden, das Modelle aus den Daten vergangener Entscheidungen generiert. Die Datensätze dieser Entscheidungen werden um ein weiteres Attribut ergänzt, die Performance. Die Performance beschreibt den Fall, der sich in der realen Welt als der richtige erwiesen hat. Bei der Kredit-Entscheidungslogik beispielsweise, wäre die Performance ob ein Kunde seinen Kredit zurückgezahlt hat oder nicht. Das ML-Verfahren, soll die Kombination von Input-Daten und Performance auf Gesetzmäßigkeiten prüfen und diese anschließend als Modell zur Verfügung stellen. Ein Modell beschreibt den Output eines ML-Verfahrens und bietet zwei fundamentale Mechanismen. Erstens soll das Modell in der Lage sein, weiter optimiert werden zu können ("lernen"), zweitens soll für einen gegebenen Input-Datensatz ein dazugehörender Output vorhergesagt werden ("evaluieren").

Das *zweite Teilziel* besteht darin Modelle heranzuziehen, um Entscheidungslogiken so zu verbessern, dass zukünftige fachliche Entscheidungen korrekter getroffen werden. Das erlernte Modell wird dazu verwendet, mehrere Datensätze zu evaluieren und den Ausgang deren Entscheidungen vorherzusagen. Die daraus resultierenden Ergebnisse sollen dann in den Entscheidungsfindungsprozess mit einfließen und somit zu mehr richtig getroffenen Entscheidungen verhelfen.

Auf Basis dieser Teilziele ergeben sich eine Reihe von Forschungsfragen:

- **Forschungsfrage 1:** Können ML-Verfahren verwendet werden, um fachliche Entscheidungen zu verbessern?
  - Was gibt es für einsetzbare ML-Verfahren?
  - Welche ML Verfahren eignen sich am besten?
- **Forschungsfrage 2:** Wie können einsetzbare ML-Verfahren anhand eines fachlichen Anwendungsfall angewendet werden?

- Welche Anwendungsfälle eignen sich hierzu?
- Was für Ergebnisse liefern diese?
- Wie können die Ergebnisse interpretiert werden?

- **Forschungsfrage 3:** Wie kann die Funktionsweise des ML-Verfahrens in ein Modell überführt werden, ohne dass dabei Performance-Daten offen gelegt oder zur Verfügung gestellt werden müssen?
- **Forschungsfrage 4:** Wie kann sichergestellt werden, dass die fachliche Entscheidung verbessert werden können?

Die vier Forschungsfragen werden in Tabelle /reftab:chapters 1.2 zu einem oder mehreren Kapiteln dieser Thesis zu geordnet.

#	Kap. 1	Kap. 2	Kap. 3	Kap. 4	Kap. 5	Kap. 6	Kap. 7
Forschungsfrage 1		×			×		
Forschungsfrage 2		×	×	×			
Forschungsfrage 3			×	×			
Forschungsfrage 4		×	×	×	×		

Tabelle 1.1.: Forschungsfragen entlang der Kapitel.

1.3. Abgrenzung

Einhergehend mit den zu beantwortenden Forschungsfragen findet auch folgende Abgrenzung statt: Es ist kein Ziel dieser Arbeit, einen eigenen ML-Algorithmus zu entwickeln. Die Entwicklung eines eigenen Algorithmus ist nicht notwendig, da in der Literatur und Praxis schon vielversprechende Algorithmen entwickelt worden sind. Vielmehr sollen bereits bestehende Algorithmen und Lösungen, die sich in der Praxis als erfolgreich bewiesen haben, intelligent verknüpft werden, um fachliche Entscheidungen zu verbessern. Ziel dieser Arbeit ist ausschließlich das Verbessern von automatisierbaren Entscheidungen. Entscheidungen können operativer und strategischer Natur sein, wobei sich nur operative Entscheidungen zur Automatisierung eignen.

1.4. Aufbau der Arbeit

Die vorliegende Bachelorarbeit besteht aus sieben Kapiteln, die sich nochmals in drei Teile (Teil I - Teil III) aufgliedern (siehe Abbildung 1.2).

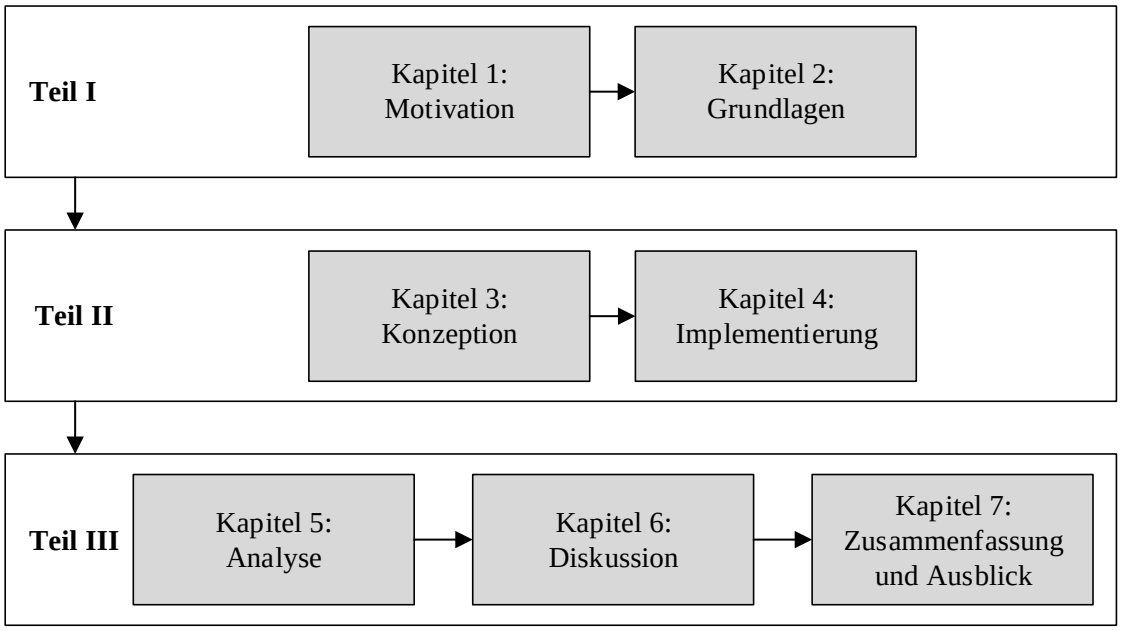

Abbildung 1.2.: Aufbau der Arbeit.

Teil 1 umfasst neben diesem einleitenden Kapitel die theoretischen Grundlagen, die für das weitere Verständnis benötigt werden. Kapitel 2.1 erläutert wie Entscheidungen abgebildet werden können, anschließend werden die Grundlagen von ML vermittelt (Kapitel 2.2), um im Anschluss verwendbare Technologien für Decison Management und ML zu identifizieren (Kapitel 2.3).

Teil 2 befasst sich mit der Konzeption des Prototypen (Kapitel 3), sowie dessen Implementierung (Kapitel 4) und stellt damit den Hauptteil dieser Arbeit dar. Hierzu sollen zuerst Anwendungsfälle genannt werden (Kapitel 3.1), wovon anschließend Anforderungen abgeleitet und spezifiziert werden (Kapitel 3.2). Mit dem Abschluss der Konzeption, wird die Implementierung des Prototypen beschrieben. Hierzu wird zunächst die Implementierung des Datenschemas dargelegt (Kapitel 4.1), bevor die Implementierung der zu verbessernden Entscheidungslogik (Kapitel 4.2) erläutert wird. Aufbauend auf der Entwicklung des Datenschemas und der Entscheidungslogik, kann das ML-Verfahren implementiert (Kapitel 4.3) und optimiert werden (Kapitel 4.4).

Teil 3 bildet den Schlussteil der Arbeit und umfasst die Analyse (Kapitel 5), die Diskussion (Kapitel 6) und die Zusammenfassung inklusive eines Ausblicks (Kapitel 7).

2. Grundlagen

Dieses Kapitel beschreibt die grundlegenden Konzepte, die für das weitere Verständnis dieser Arbeit erforderlich sind. Abschnitt 2.1 behandelt Decision Mangement und gibt Aufschluss darüber, wie Entscheidungen automatisiert und verwaltet werden. In Abschnitt 2.2 folgt eine Einführung in das Machine Learning. Abschließend nennt Abschnitt 2.3 einsetzbare Lösungen zur Umsetzung der genannten Konzepte.

2.1. Decision Management

Decision Management (DM) adressiert die Verwaltung von automatisierten Entscheidungen, über den kompletten Lebenszyklus von Entscheidungen hinweg. Dieser beinhaltet die Modellierung, Ausführung, Überwachung und die Optimierung, mit dem Ziel, fachliche Entscheidungen zu verbessern, indem der Lebenszyklus kontinuierlich durchlaufen wird. Abbildung 2.1 zeigt die einzelnen Phasen des Lebenszyklus.

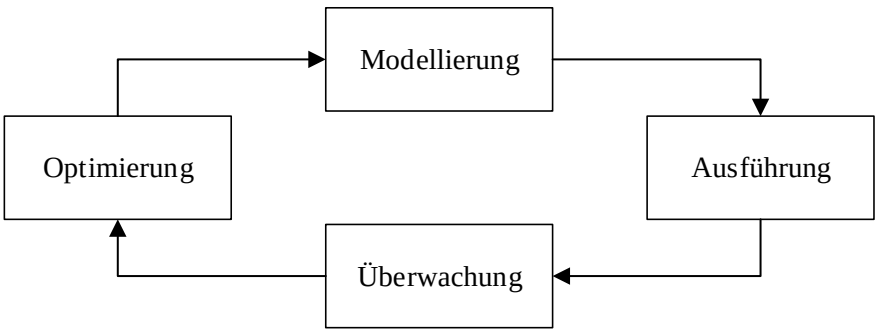

Abbildung 2.1.: Der Lebenszyklus von automatisierten Entscheidungen.

Modellierung . Zur Modellierung von Entscheidungen wurde 2015 der Standard Decison Model and Notation (DMN) von der Objekt Management Group veröffentlicht

[19, vgl. S. 7 ff.]. Rücker nennt die Schaffung eines Notationsstandards für Entscheidungen, der für Fach- und IT-Anwender gleichermaßen verständlich ist, als übergeordnetes Ziel von DMN [41, vgl. S. 40]. Die offizielle DMN-Spezifikation nennt die Erfüllung der folgenden drei Anforderungen als Ziel von DMN [19, vgl. S. 18 ff.]:

1. Modellierung von Entscheidungen, die von Menschen getroffen werden
2. Modellierung von Entscheidungen, die automatisiert getroffen werden
3. Implementierung von automatisierten Entscheidungen

Zur Umsetzung der genannten Anforderungen definiert DMN zwei verschiedene Ebenen: Die Entscheidungsanforderungsebene (deskriptiv) und die Entscheidungslogikebene (präskriptiv), beide zusammen bilden das *Decision Model*. Auf der Entscheidungsanforderungsebene werden die Anforderungen, die eine Entscheidung benötigt, in einem Decision Requirements Diagrams (DRD) modelliert. Ein DRD besteht aus Entscheidungen, Input-Daten, Business Knowledge Models und Knowledge Sources. Abbildung 2.2 [19, vgl. S. 21] zeigt die genannten Elemente eines beispielhaften DRD und deren Notation.

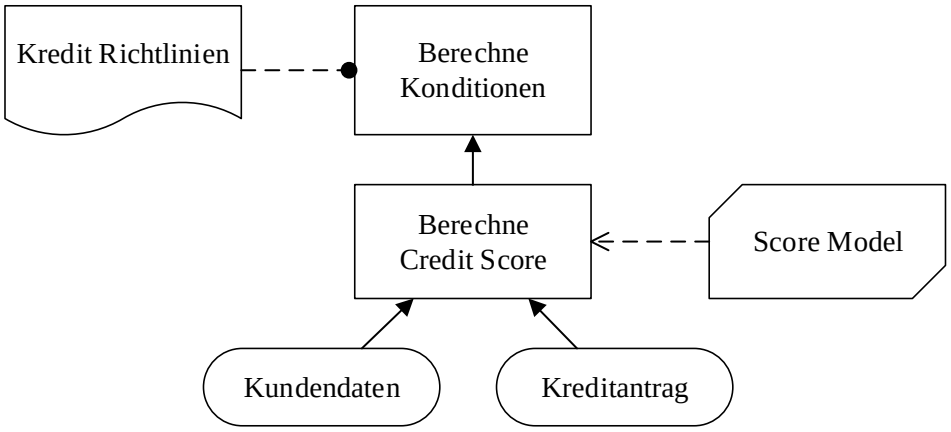

Abbildung 2.2.: Ein beispielhaftes DRD.

- Ein Input-Data-Element beschreibt Informationen, die als Eingabe für eine oder mehrere Entscheidungen dient [19, vgl. S. 30]. Abbildung 2.2 zeigt die beiden Input-Daten-Elemente Kundendaten und Kreditantrag.
- Ein Business Knowledge Model beschreibt eine Funktion die Fachwissen kapselt, beispielsweise als Geschäftsregel, Entscheidungstabelle oder als analytisches Model [19, vgl. S. 30]. In Abbildung 2.2 kapselt das Score Model das Fachwissen zur Berechnung des Credit Scores.

- Ein Decision-Element beschreibt eine Entscheidung. Sie bestimmt einen Output, anhand verschiedener Inputdaten, mithilfe von Entscheidungslogik [19, vgl. S. 20]. In Abbildung 2.2 wird der Credit Score mittels der Inputdaten Kundendaten und Kreditantrag berechnet. Anschließend wird der Credit Score an die darüber liegende Entscheidung, zur Berechnung der Konditionen, weitergegeben.
- Eine Knowledge-Source beschreibt die Quelle einer Entscheidung oder eines Business Knowledge Models. Das kann beispielsweise ein Dokument oder auch ein Fach-Experte sein [19, vgl. S. 18]. Im vorliegenden Beispiel werden die Konditionen des Kredites durch die Vorgaben der Kredit Richtlinien berechnet.

Nun gilt es sich der Entscheidungslogik-Ebene zu widmen, die insbesondere für die automatisierte Ausführung von Entscheidungen essentiell ist [19, vgl. S. 18]. DMN erlaubt den Import von existierenden Entscheidungslogik-Standards. Der wohl wichtigste Entscheidungslogik-Standard im Umfeld von DMN ist die Entscheidungstabelle. Taylor definiert Entscheidungstabellen als: "a look-up table where each cell represents an executable business rule. The conditions of the rule are columns or rows in the decision table and the intersection of the rows and columns shows the consequence of the rule" [48, S. 132]. Abbildung 2.3 zeigt eine Entscheidungstabelle zur Bestimmung des Zinses anhand eines Credit Scores. Wäre der Credit Score beispielsweise 632, würde die erste Zeile der Entscheidungstabelle zutreffen und der Zins 6 % betragen.

Berechne_Konditionen

U	Input +	Output +	Annotation
	Credit Score	Zinssatz	
	integer	double	
1	[600..650[	6,0	-
2	[650..700[	4,0	-
3	[700..750[	2,0	-
4	[750..800[	1,5	-
5	[800..850]	1,0	-
+			

Abbildung 2.3.: Beispiel einer Entscheidungstabelle zur Berechnung des Zinssatzes.

Ausführung. Um Decision Models ausführen zu können, müssen diese in einem geeigneten DMN-Tool modelliert werden. Zur vollständigen Automatisierung der Entscheidun-

gen, muss die Entscheidungslogik in der Lage sein, für jeden möglichen Satz von Input-Daten einen Entscheidungsausgang bestimmen zu können. Um eine effiziente Ausführung zu ermöglichen wird eine Decision Engine benötigt, die es erlaubt Entscheidungen in einer produktiven Einsatzumgebung auszuführen. "Eine Decision Engine ist ein Stück Software, das eine zustandslose Schnittstelle bereitstellt, um Entscheidungen auf Basis der in DMN definierten Entscheidungslogik zu fällen" [41, S. 41]. Nach der Inbetriebnahme der Decision Engine bedürfen die getroffenen Entscheidungen der Überwachung, um sicherzustellen, dass Entscheidungen so getroffen werden, dass deren Konsequenzen bestmöglich zur Erreichung der betriebswirtschaftlichen Ziele beitragen.

Überwachung. Taylor nennt vier Faktoren, die kontrolliert werden sollen und dazu führen, dass Entscheidungen verbessert werden müssen [48, vgl. S. 158]:

1. Änderungen an wirtschaftlichen Zielsetzungen
2. Neue Regularien oder Richtlinien
3. Änderungen an zugrundeliegenden Datenschemata
4. Gesamt Performance der Entscheidungen

Auf die ersten drei Faktoren muss bei deren Eintritt reagiert werden, wohingegen der vierte Faktor einer ständigen Untersuchung bedarf. Proaktive Änderungen an Entscheidungslogiken bieten Möglichkeiten, die Effektivität unserer Entscheidungen zu steigern, um somit den wirtschaftlichen Zielsetzungen näher zukommen. Wie effektiv oder gut eine Entscheidung ist definiert Taylor wie folgt: "The goals and key performance indicators or metrics of your business set a context that defines what a good or an effective decision looks like. The data you have collected over time gives you insight into what works and what doesn't, which is reflected in your current decision-making approach." [48, vgl. S. 159]. Die Analyse der Daten vergangener Entscheidungen ist somit unerlässlich, um proaktiv Entscheidungslogiken zu verbessern. Um eine effektive Überwachung und Verbesserung zu ermöglichen, müssen umfangreiche Datenmengen über die Effektivität von Entscheidungen gesammelt werden. Dazu zählen [48, vgl. S. 164]:

- Ausführungs-Daten: Hierzu zählen alle Daten die während der Ausführung einer Entscheidung erhoben werden können. Beispielsweise IDs, Zeitstempel, Eingabe-Daten, Ausgabedaten, die Quelle des Aufrufs, oder kontextbezogene Daten die während der Ausführung generiert wurden.

- Antwort-Daten: Meistens ergibt sich aus einer Entscheidung heraus eine Konsequenz für den Empfänger der Entscheidung. Um nochmals das Beispiel der Online-Kredit-Bewerbung aufzugreifen, nachdem der Kreditantrag evaluiert und das Ergebnis zurückgeliefert wurde, kann die Reaktion des Empfängers gespeichert werden. Der Empfänger kann das zurückgelieferte Kreditangebot ablehnen, zustimmen, mehr Informationen verlangen oder für später abspeichern. Gelingt es die Reaktion des Empfängers mit der Entscheidung zur späteren Analyse abzuspeichern, können daraus wertvolle Rückschlüsse über die Effektivität der Entscheidung gewonnen werden.
- Andere Unternehmens-Daten: Neben den Daten die direkt oder indirekt durch die Entscheidung und ihre Konsequenz entstehen, macht es Sinn weitere Unternehmensdaten an vergangene Entscheidungen zuknüpfen, die später in unserer Überwachungsumgebung zu sehen sein sollen. Bei einer Entscheidung zur Berechnung der Bonität beispielsweise würde die Information, ob es zu Zahlungsausfällen kam, verknüpft mit dem entsprechenden Entscheidungsdatensatz erlauben, direkte Aussagen über die Effektivität der Entscheidungslogik zu treffen.

Eine entsprechende Überwachungsumgebung müsste all diese Daten sammeln und aufbereiten, so dass der komplette Entscheidungsfindungsprozess und die daraus resultierenden Ereignisse im Nachhinein noch nachvollzogen werden können.

Optimierung. Um bestehende Entscheidungen optimieren zu können, bedarf es einer Simulations-Umgebung in der verschiedene Ansätze gegeneinander getestet werden können. Ein mögliches Szenario könnte so aussehen, dass man die Entscheidungslogik im Produktiveinsatz, in einem A/B Test gegen eine neue Entscheidungslogik testet. Testet man fortlaufend parallel verschiedene Ansätze kann man erkennen, welche sich langfristig am effektivsten auf wirtschaftliche Zielsetzungen auswirken [48, vgl. S. 173]. *Predictive Analytics* eignen sich um neue Entscheidungslogiken zu entwickeln. Predictive Analytics bezeichnet [3, vgl. S. 5] das Verwenden von Verfahren aus der Statistik, sowie dem Machine Learning um vorherzusehen, mit welcher Wahrscheinlichkeit ein ungewisses Ereignis eintritt.

2.2. Machine Learning

Machine Learning (ML) ist eines der zentralen Themengebiete innerhalb der *Künstlichen Intelligenz* (KI) [53, vgl. S. 2082] und beschreibt das Forschungsgebiet das Computer befähigt zu lernen, ohne explizit dafür programmiert worden zu sein [33, vgl. S. 1]. Lernen im Kontext von ML definieren Mitchell et al. als: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E" [32]. Soll beispielsweise ein Programm implementiert werden, das aufgrund vergangener Banktransaktionen (Experience E) lernen soll, wie man betrügerische Banktransaktionen erkennt (Task T), wird das Programm, nach erfolgreichem Lernen, besser darin sein betrügerische Banktransaktionen zu erkennen (Performance P). Der Zusammenhang zwischen Ursache und Wirkung soll mit Hilfe eines Modells beschrieben werden. Das Modell soll aus den Erfahrungen der Vergangenheit auf Ergebnisse in der Zukunft schließen können. Dabei wird versucht Eingabe- auf Ausgabewerte abzubilden, so dass die Differenz zwischen dem vorhergesagten Wert und dem realen Wert möglichst klein ist [30, vgl. S. 68]. Abbildung 2.4 veranschaulicht den Lernvorgang.

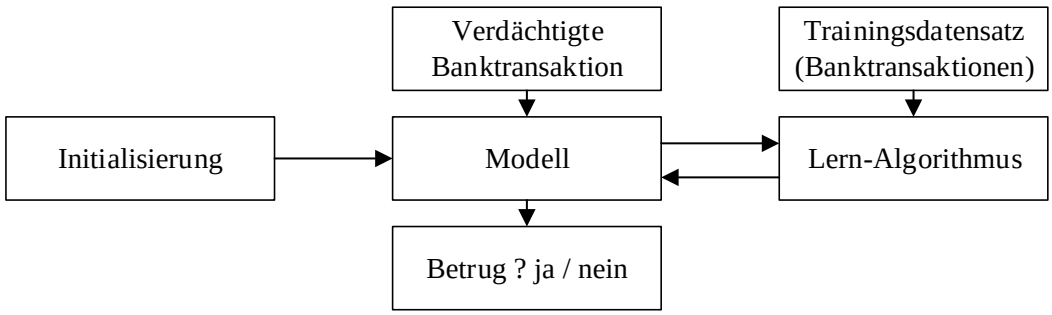

Abbildung 2.4.: Vorgehensweise zur Entwickelung eines Modells.

Die Banktransaktionen, die die Trainingsdaten bilden, besitzen verschiedene Attribute, die im Kontext von ML *Features* (Eingabedaten wie Alter, Geschlecht etc.) genannt werden. Neben den Attributen, die die Banktransaktion beschreiben, besitzen die Trainingsdatensätze ein *Label*. Das Label bildet die Erfahrung aus der gelernt werden soll. Im vorliegenden Beispiel wäre das Label ein boolescher Wert, der angibt, ob die vorliegende Banktransaktion betrügerisch war oder nicht. Tabelle 2.1 zeigt den beispielhaften Aufbau der Trainingsdaten.

Abhängig von der zugrundeliegenden Problemstellung gibt es mehrere Verfahren zur

Feature 1	Feature 2	Feature 3	Label
0.86743	0.123	0.34657	1
0.26926	0.756	0.93095	0
...	...	...	...

Tabelle 2.1.: Beispielhafter Aufbau der Trainingsdaten.

Modellbildung. Zur Auswahl eines geeigneten Lern-Algorithmus, muss die Problemstellung und die vorhandenen Daten validiert und einem der folgenden Verfahren zugeordnet werden:

- *Supervised-Learning*: Die Lerndatensätze bestehen aus Features und Labels. Das Modell wird trainiert, in dem man ihm vorgibt, wie Klassen richtig zugeordnet oder Werte richtig berechnet werden [30, vgl. S. 68].
- *Unsupervised-Learning*: Die Lerndatensätze besitzen keine Labels. Der Lern-Algorithmus versucht nur anhand der Inputvariablen Strukturen abzuleiten, wodurch dann das Modell gebildet wird [42, 30].
- *Semi-Supervised-Learning*: Eine Mischform zwischen Supervised-Learning und Unsupervised-Learning, wobei manche Trainingsdatensätze Labels besitzen können und andere nicht [54, vgl. S. 892].
- *Reinforcement-Learning*: Das Reinforcement-Learning basiert darauf, dass ein Agent lernt, wie er sich in einer bestimmten Umgebung verhalten soll, in dem er eine Belohnung beziehungsweise eine Strafe in Form eines Skalars bekommt, der angibt wie gut oder schlecht sich der Agent verhalten hat [29, vgl. S. 5].

Über die Lernverfahren hinaus, müssen auch die zu berechnenden Ergebnisse der Problemstellung klassifiziert werden, um einen geeigneten Algorithmus zu finden:

- *Klassifikation*: Unbekannte Eingabedaten werden einer Klasse zugeordnet [30, vgl. S. 68].
  - Beispiel 1: Katzenfotos sollen von Hundefotos unterschieden werden.
  - Beispiel 2: Auf MRT-Bildern von Lungen soll erkannt werden, ob ein Tumor bös- oder gutartig ist.
- *Regression*: Bestimmung eines numerischen Wertes für einen gegebenen Eingabedatensatz [42].

  - Beispiel 1: Preisberechnung eines Gebrauchtwaagens.
  - Beispiel 2: Vorhersage der verbleibenden Lebensdauer eines Motors.
- *Clustering*: Beschreibt das Gruppieren von Objekten in verschiedene Gruppen. Die Datensätze einer einzelnen Untergruppe sind sich sehr ähnlich in ihren Eigenschaften, wobei die Datensätze der verschiedenen Untergruppen sehr unterschiedlich in ihren Eigenschaften sind [42].
  - Beispiel 1: Smartphone Fotoalben, die Gesichter analysieren und Fotos nach Personen gruppieren.
  - Beispiel 2: Analysieren von Kundendaten zur Gruppierung verschiedener Kundenklassen, für gezielte Marketingkampagnen.

Konventionelle Informationssysteme bearbeiten Aufgaben für gewöhnlich schneller als der Mensch. Nicht desto trotz handelt der Mensch in vielen Disziplinen noch "intelligenter" als ein Computer, als Beispiel hierfür ist die Gesichts- und Spracherkennung zu nennen. Daher liegt es nahe, dass Forscher sich am menschlichen Gehirn inspirieren lassen beim Versuch intelligente Maschinen zu entwickeln. Die Entwicklung von künstlichen neuronalen Netzen geht zurück in die 40er Jahre, als die notwendigen biologischen und elektrotechnischen Voraussetzungen vorlagen [44, vgl. S. 1]. Grundbaustein der menschlichen Intelligenz bilden Neuronen, die über Ein- und Ausgangsverbindungen (Synapsen) mit bis zu 200000 anderen Neuronen verbunden sein können [46, vgl. S. 5]. Das *Perzeptron* ist ein künstliches Neuron, das in den 60er Jahren zur Mustererkennung entwickelt wurde [2, vgl. S. 86]. Abbildung 2.5 zeigt ein Perzeptron.

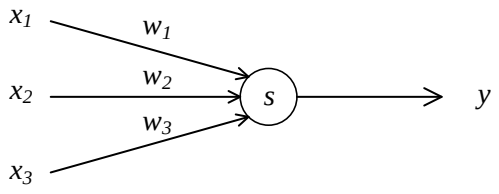

Abbildung 2.5.: Aufbau eines Perzeptrons.

Das Beispiel-Perzeptron besitzt die Inputvariablen *x1*, *x2*, *x3*, die Gewichte *w1*, *w2*, *w3*, sowie den Schwellenwert *s*. Jede Verbindung zwischen zwei Neuronen besitzt ein Gewicht, das den Einfluss eines Neuron A auf ein Neuron B beschreibt [2, vgl. S. 86]. Der Ausgabewert ermittelt sich aus dem Vergleich der gewichteten Summe $\sum_j w_j \times x_j$ und einem zuvor definiertem Schwellenwert [21, 27, 14]. Formel 2.1 [21, 14] zeigt den Vergleich nochmals als mathematischen Ausdruck:

$$y = \begin{cases} 0 \text{ if } \sum_j w_j \times x_j \leq s \\ 1 \text{ if } \sum_j w_j \times x_j > s \end{cases} \tag{2.1}$$

Wird der Schwellenwert s überstiegen, gibt das Perzeptron die 1 zurück anderen Falls die 0. Abbildung 2.6 soll die Funktionsweise des Perzeptrons nochmals mit Hilfe von Beispiel-Werten veranschaulichen. Aus der Abbildung kann der Schwellenwert $s = 6$ entnommen werden und mittels der Inputvariablen und Gewichten errechnet sich $\sum_j w_j \times x_j = 8$. Da $8 > 6$ wird der Schwellenwert überstiegen und das Perzeptron liefert die 1 zurück (vgl. Formel 2.1).

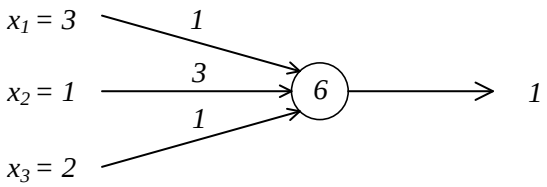

Abbildung 2.6.: Modell eines Perzeptrons mit Beispiel-Daten.

Neben dem Perzeptron gibt es noch weitere künstliche Neuronen, wie z.B. das Sigmoid-Neuron. Zum Verständnis von neuronalen Netzen, ist die Betrachtung des Perzeptrons an dieser Stelle allerdings genügend. Mit einem einzigen künstlichen Neuron ist es jedoch nicht möglich, komplexere Sachverhalte abzubilden. Dies wird erst durch die Kombination vieler künstlicher Neuronen möglich. Üblicherweise besteht ein künstliches neuronales Netz aus einer Eingabe-Schicht (*Input-Layer*), einer Versteckten-Schicht (*Hidden-Layer*) und der Ausgabe-Schicht (*Output-Layer*) [14, vgl. S. 291]. Abbildung 2.7 soll diesen Sachverhalt verdeutlichen.

Dabei ist die Anzahl der Neuronen in der Hidden-Layer variabel, wohingegen die Input-Layer so viele Neuronen wie Inputvariablen besitzt. Die Anzahl der Neuronen in der Output-Layer ergibt sich durch die Anzahl der insgesamt zu berechnenden Werte des neuronalen Netzes. Soll beispielsweise ein neuronales Netz zur Erkennung von Ziffern in Bildern entwickelt werden, gäbe es zehn Neuronen in der Output-Layer, wobei jedes Neuron eine Ziffer aus dem Bereich null bis neun repräsentiert.

Durch das Variieren der Gewichte und der Schwellenwerte, ergeben sich Modelle mit denen verschiedene Entscheidungsprozesse abgebildet werden können. Um ein Modell zu

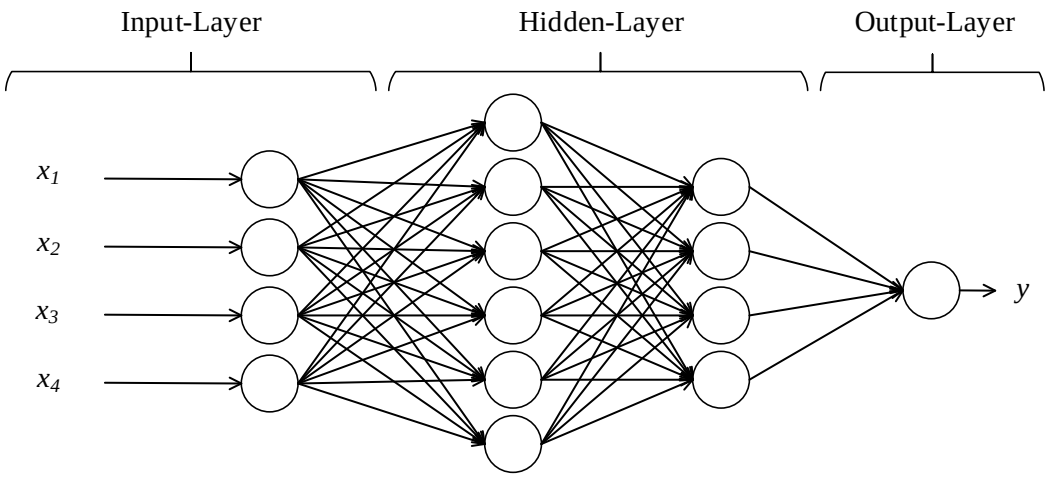

Abbildung 2.7.: Aufbau eines neuronalen Netzwerkes mit mehreren Schichten.

entwickeln muss das neuronale Netz lernen (trainiert werden). Im Kontext von neuronalen Netzen versteht man unter Lernen, die Modifikation der Gewichte [14]. Zur Modifikation dieser Parameter, werden Lern-Algorithmen wie der *Backpropagation-Algorithmus* verwendet. Der Backpropagation-Algorithmus wird für Supervised-Learning Problemstellungen verwendet [2, vgl. S. 89], indem iterativ Datensätze durch das Neuronale-Netz gegeben werden, so dass auf Basis der Abweichung zwischen berechnetem und tatsächlichem Wert (Fehler-Funktion), die Gewichte dahingehend aktualisiert werden das die Fehler-Funktion kleiner wird.

....

- Was ist Machine Learning

- Arten von Algorithmen

- Neuronale Netze

- Übergang: Es gibt viele verschiedene Algorithmen etc. allerdings macht es keinen Sinn mehr diese selbst zu implementieren -> Verwendung von Frameworks und somit überleiten auf nächste Kapitel

2.3. Technologien

- Verwendung von Frameworks erforderlich um Effizient zu sein.

- Rad nicht neu erfinden evtl. Studie zur Verwendung von Frameworks.

2.3.1. Decision Management

- ACTICO - SAS - FICO - Signavio - Camunda

2.3.2. Machine Learning

- Neuroph - Aerosolve - Deeplearning4j - Appache Spark

Die nachfolgende Rechnung soll das Gewicht-Update mit dem Backpropagation-Algorithmus erklären. Abbildung 2.8 zeigt ein beispielhaftes neuronales Netz mit Sigmoid-Neuronen. Ein Sigmoid-Neuron ist ein modifiziertes Perzeptron, dessen Ausgabewert jeden Wert zwischen null und eins annehmen kann.

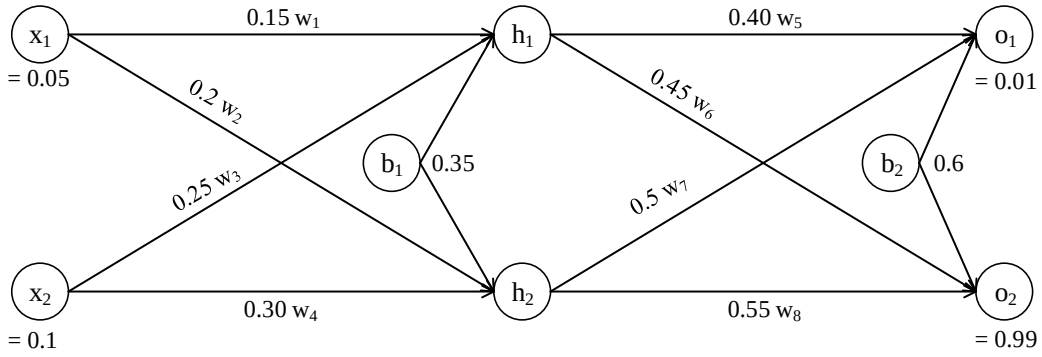

Abbildung 2.8.: Neuronalesnetz mit Sigmoid-Neuronen und Bias.

Das Neuronalenetz besitzt zwei Inputvariablen x_1 und x_2, zwei Outputvariablen o_1 und o_2 und zwei Biase b_1 und b_2. Die Gewichte w_n können den Kanten entnommen werden. Des weiteren enthält die Zeichnung den ersten Lerndatensatz. Für $x_1 = 0,05$ und $x_2 = 0,1$ sollen die Werte $o_1 = 0,01$ und $o_2 = 0,99$ vorhergesagt werden. Da das Gewicht-Update aus Basis der Fehlerrate geschieht, muss diese zuerst berechnet werden.

1. Der erste Schritt beinhaltet das Berechnen der Fehlerrate mittels der Squared-Error-Funktion (siehe Formel 2.2).

$$E_{total} = \sum \frac{1}{2}(target\text{-}output)^2 \tag{2.2}$$

Dazu muss der Output o_1 für den gegeben Lerndatensatz berechnet werden, indem wir den Output des Neurons h_1 berechnen:
$net_{h1} = w_1 * x_1 + w_2 * x_2 + b_1 * 1$

$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$

Einsetzen von $net_{h1} = 0,3775$ in die Sigmoid-Funktion:

$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0,593269992$
Wiederholen wir den selben Schritt für h_2

2.

$$\frac{\partial E_{total}}{\partial w} = \frac{\partial E_{total}}{\partial out} * \frac{\partial out}{\partial net} * \frac{\partial net}{\partial w} \tag{2.3}$$

3. Konzeption

Dieses Kapitel beschreibt die Konzeptionsphase des Prototypen, mit dem der Ansatz dieser Thesis geprüft wird. Hierzu werden in Kapitel 3.1 Anwendungsfälle definiert, um anschließend in Kapitel 3.2 und 3.3 relevante Themenfelder, zur Implementierung des Prototypen, auf Basis der Anwendungsfälle zu identifizieren. Die identifizierten Themenfelder bilden das technische Konzept zur Implementierung des Prototypen, der in Kapitel 4 dargestellt wird.

3.1. Definition von Anwendungsfällen

Dieser Abschnitt erläutert die Wahl der Anwendungsfälle , die in dieser Arbeit für die Konzeption und Implementierung verwendet werden. Die beiden auserwählten Anwendungsfälle *Erkennung von Kreditausfällen* und *Erkennung von Betrugsversuchen* werden in Kapitel 3.1.1 bzw. 3.1.2 im Detail erläutert. Darüber hinaus werden für jeden Anwendungsfall Szenarien definiert, mit denen das Potenzial von Online-Learning überprüft wird.

Die Anwendungsfälle wurden anhand der nachfolgenden Kriterien ausgewählt:

- **Allgemeingültigkeit:** Der Anwendungsfall muss generischer Natur sein, so dass ein entwickeltes Verfahren zur Lösung des Anwendungsfalls in verschiedenen Unternehmen und Branchen angewendet werden kann.
- **Realitätsbezug:** Der Anwendungsfall muss in seiner Komplexität repräsentativ dafür stehen, was auch in realen Softwareentwicklungsprojekten an Komplexität abverlangt wird.
- **Stellenwert des Anwendungsfalls:** Ein möglicher Anwendungsfall muss einen gewissen Stellenwert innerhalb einer Branche besitzen, so dass neue, besser funktionierende Verfahren einen messbaren wirtschaftlichen Mehrwert schaffen.

- **Automatisierte Entscheidungen:** Kern eines möglichen Anwendungsfalles muss das Treffen einer automatisierten operativen Entscheidung sein. Typischerweise gibt es bereits Lösungen, die auf Technologien, wie Business Process Management, Business Rules Management oder Decision Management, aufsetzen.

Anhand dieser Kriterien wurden die beiden Anwendungsfälle Erkennung von Kreditausfällen, sowie Erkennung von Betrugsversuchen bei Kreditkartentransaktionen, als am besten geeignet identifiziert. Darüber hinaus wurden auch die Anwendungsfälle Erkennung von Geldwäsche, sowie die Erkennung von Marktmanipulationen evaluiert und als weniger geeignet eingestuft, da beide Anwendungsfälle eher regulatorischen Zwecken dienen und weniger dem Kerngeschäft der Banken.

3.1.1. Erkennung von Kreditausfällen

Wie in Kapitel 1 bereits beschrieben, können Kredite, die nicht zurückgezahlt werden, zu hohen Verlusten bei Banken führen. Dies wiederum kann zu Finanzkrisen führen, die die komplette Weltbevölkerung in ihren Auswirkungen erreicht. Daher ist die korrekte Erkennung des Kreditrisikos ein elementarer Bestandteil des Kreditvergabeprozesses und maßgeblich verantwortlich für dessen Qualität. Die Eigenkapitalvorschrift Basel II ist für alle Banken in Mitgliedsländern der Europäischen Union verpflichtend [40] und hält darüber hinaus fest, wie Kreditrisiken zu berechnen sind. Abbildung 3.1 [25] zeigt die Hauptkomponenten zur Berechnung des Kreditrisikos.

Der erwartete Verlust (EL) bildet die Grundlage zur Bestimmung der Eigenkapitalanforderung an die Bank nach Basel II [13, vgl. S. 31]. Dabei hat das Kreditinstitut die Freiheit Modele zur Errechnung der Ausfallwahrscheinlichkeit (PD) selbst zu bestimmen, so lange spezifische Minimalanforderungen erfüllt werden. Basel II nennt keine bevorzugte Methode, oder Best-Practice zur Erstellung eines Models. Engelmann und Rauhmeier [13] nennen die Regressions- und Diskriminanzanalyse als die gängigsten Verfahren zur Modelbildung im Umfeld der Kreditvergabe.

Neben der Berechnung quantitativer Faktoren, fließen auch qualitative Faktoren in die Kreditentscheidung mit ein [35, vgl. S. 25 ff.]. Insbesondere bei Krediten an Firmenkunden besitzt die Betrachtung der qualitativen Faktoren einen, hohen Stellenwert. Als Beispiel für mögliche qualitative Faktoren wäre die Branche des Unternehmens oder die Erfahrung des Managements zu nennen. Im klassischen Privatkunden-Kreditgeschäft

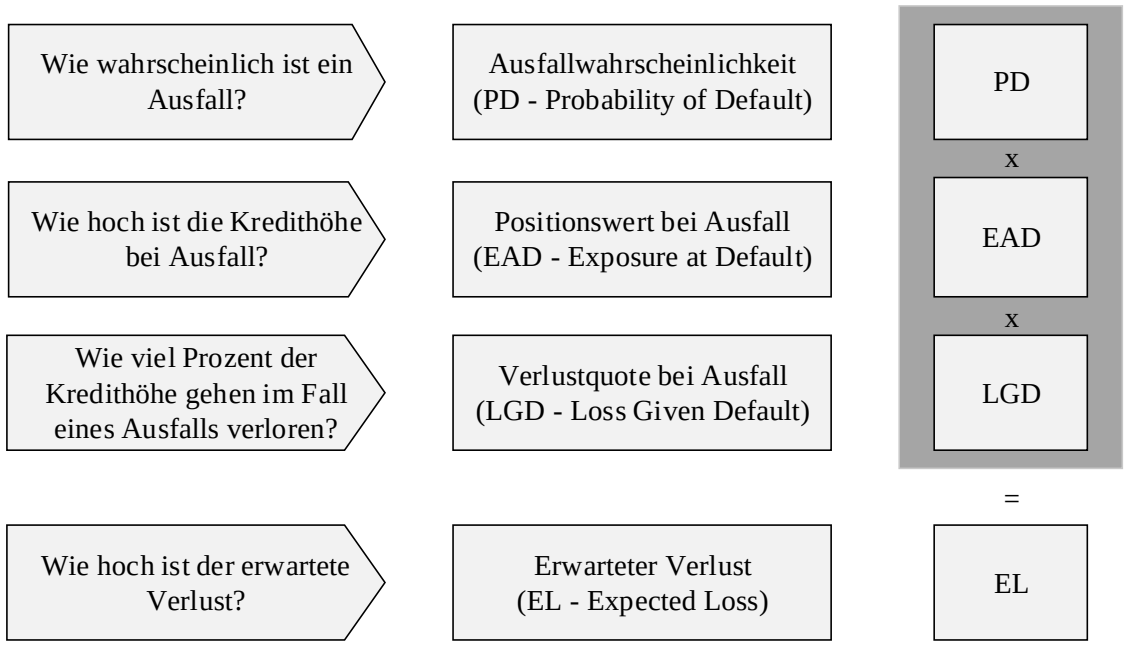

Abbildung 3.1.: Die Hauptkomponenten des Kreditrisikos.

sind komplett automatisierte Bewertungsverfahren die Regel [35, vgl. S. 31 ff.]. Unabhängig vom Kreditvergabeprozess als Ganzes muss bei Firmenkunden, wie Privatkunden, die Ausfallwahrscheinlichkeit ermittelt werden.

Um das Potenzial von Online-Learning (vgl. Kapitel 2.2) aufzuzeigen, werden für diesen Anwendungsfall die nachfolgende Szenarien definiert:

- *Ausfall von Krediten mit schlechtem Einkommen-Schulden-Verhältnis.* Das Einkommen-Schulden-Verhältnis ist eine aussagekräftige Bemessung der Bonität. Das Szenario sieht vor, dass die Kunden mit dem schlechtesten Einkommen-Schulden-Verhältnis, ihre Kredite nicht mehr stemmen können. Hierbei gibt es eine große Parallele zur Finanzkrise, den dadurch dass die Häuser zu wertvoll bewertet wurden, stimmte die errechnete Bonität des Kunden nicht mehr und Kredite fingen an auszufallen. Ähnlich wie bei der Finanzkrise, wird im hier gewählten Szenario ein Fehler in den Eingabe-Daten simuliert.
- *Ausfall von Krediten die an Mieteinnahmen gebunden sind.* Das zweite Szenario sieht vor, dass Kredite, die an Wohnungen mit mehreren Wohneinheiten gekoppelt sind, vermehrt ausfallen. Dabei wird unterstellt, dass manche Häuser mittels Mieteinnahmen finanziert werden. Durch den Ausfall der Mieteinnahmen kommt es zu einem Liquiditätsverlust des Kreditnehmers, was im Ausfall des Kredites endet.

Dieses Szenario simuliert den Eintritt zufälliger Ereignisse aus der Wirtschaft, die großen Einfluss auf die zu treffende Entscheidung haben.

3.1.2. Erkennung von Betrugsversuchen

Der zweite Anwendungsfall ist das Erkennen von betrügerischen Banktransaktionen. Studien wie [45] zeigen, dass durch Kreditkartenbetrug in den Vereinigten Staaten im Jahr 2009, ein Schaden von 190 Milliarden Dollar entstanden ist. Ebenso zeigen Studien [49, vgl. S. 24], dass im Jahr 2016 66 % aller Betrugsversuche erfolgreich sind.

ACTICO bietet in ihrer Compliance-Lösung [10] das Modul Name Matching Transaction (NMT) [34] an. Das NMT Modul prüft vor der Ausführung einer Banktransaktion, ob ein Betrugsverdacht vorliegt. Sollte eine Transaktion als verdächtig eingestuft werden, wird die Ausführung vorübergehend angehalten. In diesem Zeitraum nimmt die Bank Kontakt mit dem Kunden auf und prüft, ob die Transaktion so von ihm gewollt war und wieder freigegeben werden darf.

Eine Transaktion verbindet zwei Parteien, den Auftraggeber der Transaktion und den Empfänger der Transaktion. Sind beide Parteien Kunde der selben Bankinstitution ist keine Prüfung notwendig, da die Transaktion, im Falle eines Betrugs, problemlos widerrufen werden könnte. Sind die beiden Parteien Kunden verschiedener Bankinstitutionen (Entscheidung „Check fraud relevance“ in Abbildung 3.2) beginnt die eigentlich Betrugsprüfung.

Die anschließend ausgelöste Betrugsprüfung besteht aus vier Entscheidungen:

- **Check account (Black-List):** Ein Betrugsfall liegt vor, wenn der Empfänger auf einer „Black-List“ steht. Derartige Prüflisten werden von öffentlichen Ämtern aber auch von kommerziellen Anbietern veröffentlicht.
- **Check country:** Ein Betrugsfall liegt vor, wenn das Ziel-Land des Empfängers auf einer Prüfliste steht.
- **Check behaviour:** Ein Betrugsfall liegt vor, wenn Abweichung in den Nutzungs-Statistiken des Kunden entdeckt wurden. Hierzu zählen ungewöhnliche Zahlungsmethoden, Abweichungen von den üblichen Transaktionszeiten, Abweichungen zu den üblichen Beträgen vorheriger Transaktionen und Abweichungen von der üblichen Menge an Transaktionen in einem Monat.

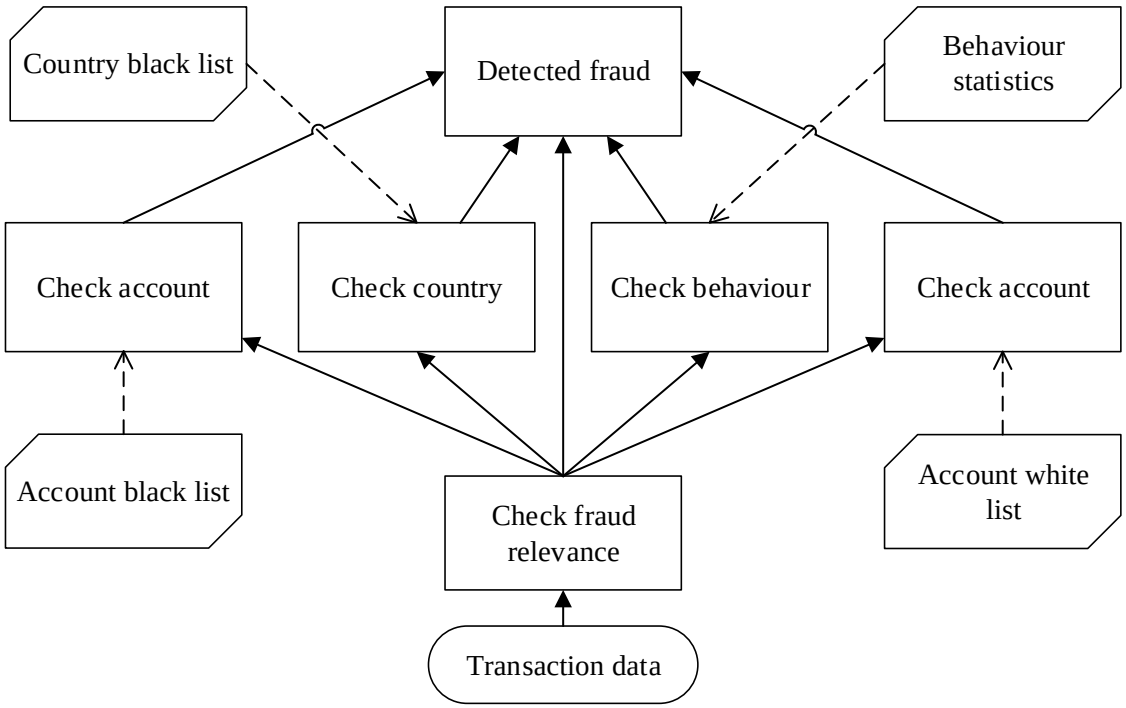

Abbildung 3.2.: Decision Requirements Diagram der Transaktionsbetrugs-Entscheidung.

- **Check account (White-List):** Ein Betrugsfall liegt nicht vor, wenn der Empfänger auf einer „White-List“ steht. Eine White-List ist eine Liste mit Bankkonten die als vertrauenswürdig eingestuft wurden. Transaktionen die der Bank-Kunde nachträglich freigeben hat, werden anschließend der White-List hinzugefügt.

Die letzte Entscheidung „Fraud detected“ empfängt die Ergebnisse der vorangegangenen Entscheidungen. Sollte bei keiner der vorangegangenen Entscheidung einen Verdacht auf Betrug vorliegen, liegt kein Betrug vor, selbst wenn der Transaktions-Empfänger nicht auf der White-List steht. Sollte nur eine der Entscheidungen einen Betrugsverdacht an „Fraud detected“ leiten, wird die komplette Transaktion gesperrt. Üblicherweise folgt bei Kunden von ACTICO an dieser Stelle einen Workflow, der den Bankkunden per SMS über die Sperrung informiert. Der Bankkunde kann die Transaktion anschließend auf seinem Mobiltelefon wieder freigeben (Transaktionen in Embargo-Länder, oder von Sanktionen betroffene Länder, können nicht wieder freigegeben werden).

3.2. Model zur Erkennung von Kreditausfällen

Im nachfolgenden Kapitel werden alle relevanten Themenfelder zur prototypischen Realisierung des ersten Anwendungsfalls vorgestellt. Zur Bildung eines Models, dass ausfallende Kredite identifizieren kann, müssen zuerst Lerndaten generiert werden (Kapitel 3.2.1). Die Lerndaten ermöglichen anschließend das Lernen unter Verwendung eines ML-Algorithmus, der mittels des Lernprozesses das Model bildet (Kapitel 3.2.2).

3.2.1. Lerndaten

Nach ausgiebiger Recherche konnten, über ein Webportal des amerikanischen Hypotheken-Rückversicherer Fannie Mae, geeignete anonymisierte Kreditantragsdaten gefunden werden [28]. Alle Datensätze entstammen Immobilien-Darlehen. Die Kreditantragsdatensätze sind über Identifikationsnummern mit Performancedatensätzen verbunden, die das Zahlungsverhalten und weitere Informationen monatlich erfassen [15].

Datenerzeugung. Zum Download stehen die Acquisition- (Kreditantragsdaten) und die Performance-Datei (Performancedaten) bereit. Der erste Schritt zur Modellbildung ist die Aufbereitung der Lerndaten. Diese bestehen aus Features und einem Label (vgl. Kapitel 2.2). Als Features dienen die Kreditantragsdaten aus der Acquisition-Datei, durch den Prozess des Feature Engineerings (siehe Exkurs 1) aufbereitet werden. Um die Erzeugung der Lerndaten abzuschließen muss noch das Label ermittelt werden. Das Label, beschreibt die Erfahrung aus der Vergangenheit, aus der gelernt werden soll. Im vorliegenden Anwendungsfall bedeutet dies konkret, ob der Kredit ausgefallen ist oder nicht. Zur Ermittlung des Labels eines Kreditantragsdatensatz muss die Performance-Datei auf Zahlungsverzögerungen untersucht werden.

Exkurs 1 (**Feature Engineering**). *"Feature Engineering* is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data." [11]. Feature Scaling und Feature Selection sind Teilmengen des Feature Engineerings. Zusammenfassend lässt sich festhalten, dass der Prozess der Transformation von Attributen aus den Rohdaten, zu Features die dem ML-Algorithmus verständlich sind, als Feature Engineering bezeichnet wird.

Jeder Kreditantragsdatensatz besitzt eine einzigartige Identifikationsnummer. Mittels dieser Identifikationsnummer können alle Performancedatensätze zu jenem Kredit identifiziert werden. Die Performancedatensätze sind mit Zeitstempeln gekennzeichnet, wobei der aktuellste Datensatz verwendet wird um den Zahlungsverzug, und damit das gesuchte Label, festzustellen. Die Spalte *Current Loan Delinquency Status* beziffert den aktuellen Zahlungsverzug in Monaten. Befindet sich der Kunde in Zahlungsverzug (Current Loan Delinquency Status > 0) wird dem Label der Wert 1 zugewiesen (Ausfall des Kredites). Liegt kein Zahlungsverzug vor (Current Loan Delinquency Status $= 0$), wird der Wert 0 zu gewiesen (kein Ausfall des Kredites). Wenn der Zahlungsverzug unbekannt ist, besitzt Current Loan Delinquency Status den Wert "X". In diesem Fall werden die betroffenen Datensätze übersprungen. Abbildung 3.3 soll den Prozess der Lerndatenerzeugung nochmals veranschaulichen.

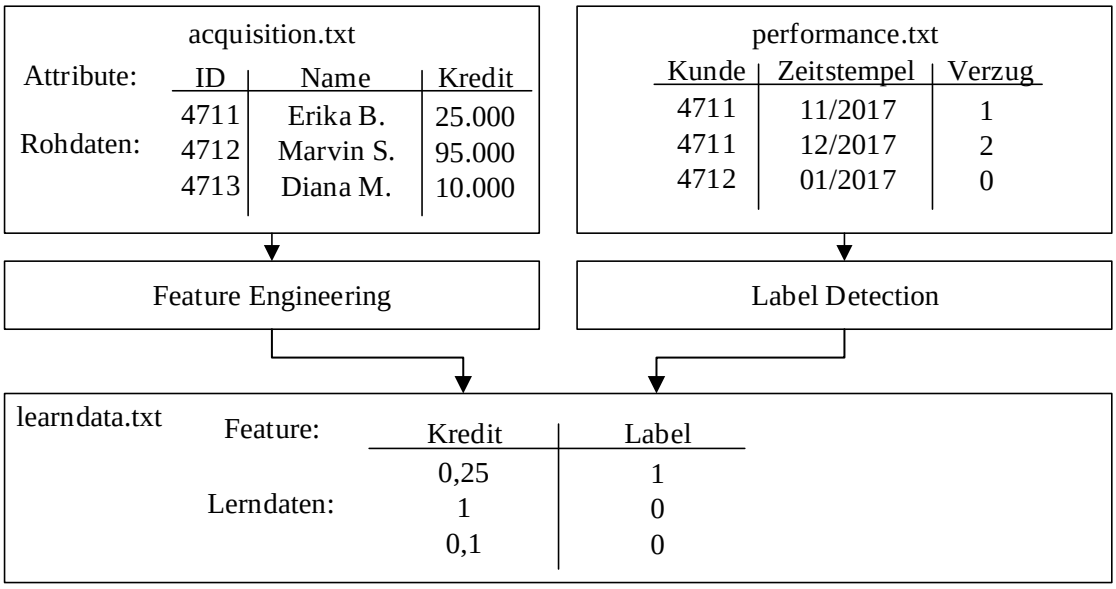

Abbildung 3.3.: Vorgehensweise bei der Erstellung des Lerndatensatzes.

Zusammenfassend sollte festgehalten werden, dass zur Bildung des Lerndatensatzes zwei Prozesse ablaufen müssen. Einer zur Bildung der Features, durch die Durchführung von Feature Engineering. Der andere Prozess, generiert das Label aus der Performancedatei (Nach dem oben beschriebenen Verfahren).

Im vorliegenden Anwendungsfall wurde Feature Engineering genutzt, um kategorische Attribute in ein Format zu überführen, so dass das neuronale Netz das Feature als kategorischen Wert und nicht als nummerischen Wert interpretiert. Hierzu muss jede

mögliche Klasse eines Attributes auf ein eigenes Feature abgebildet werden. Beispielsweise bedeutet das, dass aus dem Attribut Bundesstaat in den Ausgangsdaten, für jeden möglichen Wert ein Feature extrahiert werden muss. Die Vereinigten Staaten von Amerika besitzen 50 Bundesstaaten, dementsprechend wird das Attribut Bundesstaat in den Ausgangsdaten auf 50 Features in den Lerndaten zerlegt. Der Inhalt des Features, bildet ein boolescher Wert, der spezifiziert, ob der Kreditnehmer in diesem Bundesstaat lebt oder nicht. Nicht selten kommt es vor, dass ein Attribut in den Ausgangsdaten keinen Wert besitzt, also leer ist. Um dem neuronalen Netz diesen Sachverhalt verständlich zu machen wird ein weiteres Feature hinzugefügt, das einen booleschen Wert besitzt, der spezifiziert, ob der Bundesstaat im Kreditantragsdatensatz enthalten ist oder nicht. Tabelle 3.1 illustriert das eben genannte Beispiel nochmals.

state-is-unknown	is-in-alabama	is-in-alaska	is-in-arizona
0	0	0	1
1	0	0	0
...	...	...	...

Tabelle 3.1.: Zerlegung des kategorischen Bundesstaat-Attributs in Features.

Die eben genannte Vorgehensweise wird auch auf alle anderen kategorischen Attribute angewandt. Je nach Datenbankschema, müssen für Werte die „nullable“ sein dürfen, extra Features eingeführt werden, die dies entsprechend abbilden. Im vorliegenden Datensatz der Fannie Mae wurden die Attribute *Channel*, *First-Time-Home-Buyer-Indicator*, *Loan Purpose*, *Property Type*, *Occupancy Status*, *Property state* und *Co Borrower Credit Score* (vorhanden bzw. nicht vorhanden) kategorisch zerlegt. Somit ergeben sich aus den 18 Attributen der Rohdaten, 83 Features in den Lerndaten. Außerdem wurden die Attribute *Origination Date* und *First Payment Date* von einander subtrahiert. Ein Machine Learning Algorithmus, kann Daten im Datumsformat nicht zweckgemäß interpretieren. Aus diesem Grund wurde die Differenz in Monaten zwischen beiden Daten ermittelt und anschließend als Feature den Lerndaten hinzugefügt.

Exkurs 2 (**Feature Selection**)**.** *Feature Selection* beschreibt die Auswahl der aussagekräftigsten Features aus einem Datensatz [42, vgl. S. 503 ff.]. Aussagekräftig bedeutet in diesem Kontext, wie stark wirkt sich die Existenz des Features auf das zu berechnende Label aus. Der Kreditantragsdatensatz besteht aus vielen nützlichen Informationen zur Vorhersage der Ausfallwahrscheinlichkeit. Allerdings beinhalten solche Datensätze auch

sekundäre Informationen, wie z.B. den Namen des Kreditnehmers. Aus dem Namen des Kreditnehmers kann kein Rückschluss auf dessen Ausfallwahrscheinlichkeit gezogen werden. Daher macht es keinen Sinn, derartige Features mit in den Lerndatensatz aufzunehmen, da der Lerndatensatz und damit auch das neuronale Netz mehr Speicher und Rechenzeit benötigen würde. Zur Identifizierung der wichtigen Features gibt es grundsätzlich zwei Herangehensweisen: Die erste Herangehensweise identifiziert Features aufgrund von Domänenwissen im vorliegenden Anwendungsfall. Die zweite Herangehensweise ist die Durchführung einer Dimensionalitätsreduktion [42, vgl. S. 326]. Data Mining Tools wie Weka [38] oder KNIME [37] bieten Funktionen, die angeben, wie hoch der Einfluss eines Features auf das Label ist.

Für den vorliegenden Datensatz wurden keine Dimensionalitätsreduktion durchgeführt, da die Acquisition-Datei, die die Kreditantragsdatensätze enthält, die geringe Anzahl von 25 Attributen aufweist. Eine Feature-Selection, aufgrund von Performance-Gründen, ist somit nicht notwendig. Allerdings wurden aufgrund von Domänenwissen die Attribute *Mortage Insurance Percentage*, *Product Type*, *Mortage Insurance Type* und *Relocation Mortage Indicator* ausgeschlossen. Diese Attribute enthalten ausschließlich Informationen über die von Fannie Mae bereitgestellte Rückversicherung und tragen somit nicht zur Erkennung eines Kreditausfalles bei. Darüber hinaus würde das Kriterium der Allgemeingültigkeit verletzt werden, sowie zur Fälschung der Ergebnisse beitragen, da Banken zum Zeitpunkt der Kreditvergabe nicht über derartige Informationen verfügen.

Exkurs 3 (**Feature Scaling**)**.** *Feature Scaling* beschreibt ein Verfahren bei dem Daten auf einen festgelegten Bereich skaliert werden [42, vgl. S. 322]. Typischerweise weichen die Daten in ihren Werten pro Spalte stark ab. Allerdings sollten alle Werte einer Spalte auf einen Bereich zwischen null und eins skaliert werden. Der Grund für die Notwendigkeit von Feature Scaling ist, dass das Gradientenverfahren, welches Teil der komplexen Hintergrundberechnungen neuronaler Netze und vieler anderer ML-Algorithmen ist, lokale Minima schneller findet, wenn sich die Input-Variablen im selben Bereich befinden [8, vgl. S. 203].

Um das Gradientenverfahren beschleunigen zu können müssen die Kreditantragsdatensätze pro Feature (spaltenweise) skaliert werden. Die Skalierung der Daten geschieht unter Verwendung der Formel 3.1.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{3.1}$$

Die Skalierung wird immer pro Datenspalte vorgenommen, dazu wird zuerst der Größte wie der Kleinste Wert der Spalte berechnet, woraufhin anschließend die Formel auf jeden Wert der Spalte angewendet und durch x' ersetzt wird. Sicherheitshalber muss geprüft werden, dass $max(x)$ und $min(x)$ nicht gleich groß sind. Dies könnte dazu führen, dass versucht wird durch null zu teilen, was die Applikation zum Absturz bringen würde. Sollte $max(x)$ und $min(x)$ gleich groß sein, würde das bedeuten dass das Feature nur aus einem Wert besteht. Ein solches Feature wäre allerdings sinnlos, da es kein Rückschluss auf die zu identifizierende Klasse erlaubt. Ein solcher Fall sollte somit nie in der Realität auftreten. Nichts desto trotz muss der Prototyp eine entsprechende Fehlerbehandlung besitzen.

Die Kreditantragsdaten werden von Fannie Mae Quartalsweise veröffentlicht. Aus diesem Grund werden die Antragsdaten von Quartal eins bis drei, aus dem Jahr 2007 verwendet, um das Modell zu erlernen (Lerndaten). Die Akkuranz des erlernten Modells wird mittels der Daten aus Quartal vier überprüft (Evaluationsdaten). Es soll also untersucht werden, welche Vorhersagen für Quartal vier getroffen werden. Die Vorhersagen werden anschließend verglichen mit den in echt eingetretenen Kreditausfällen.

Die in Kapitel 3.1.1 vorgestellten Szenarien, werden durch die Manipulation der Lerndaten abgebildet. Für Szenario 1, wird anhand des Attributs *Debt-to-inocme-ratio* in den Ausgangsdaten identifiziert, welche Kredite Ausfallen werden. Dazu wird das Label der Kredite, deren Debt-to-inocme-ratio den Wert fünfzig übersteigt, auf eins gesetzt (Ausgefallen). Für Szenario 2, wird anhand des Attributs *Number of Units* geprüft, welche Kredite ausfallen werden. Alle Kredite, deren Number of Units größer vier ist, werden in Szenario 2 ausfallen.

3.2.2. Modellbildung

Nach der Erzeugung der Lerndaten, kann ein Algorithmus konzipiert werden, der aus den Lerndaten ein Modell zur Vorhersage von Kreditausfällen erlernt. Hierzu soll ein mehrschichtiges neuronales Netz verwendet werden. Aus den Lerndaten, ergibt sich die Anzahl an Eingabe- und Ausgabe-Neuronen. Die Lerndaten besitzen 83 Features, somit

besteht die Eingabe-Schicht aus 83 Neuronen. Aufgrund der Anzahl möglicher Werte für das Label, Kreditausfall oder kein Kreditausfall, ergibt sich die Anzahl zwei für die Ausgabe-Neuronen. Somit steht die Architektur der Ein- und Ausgabeschicht fest und es gilt die Anzahl der versteckten Schichten, sowie die Anzahl der Neuronen innerhalb der versteckten Schichten, zu bestimmen. Die Wahl dieser Parameter ist komplex und wirkt sich immens auf das Lernverhalten des neuronalen Netzes aus.

Exkurs 4 (Underfitting vs. Overfitting). *Underfitting* beschreibt den Sachverhalt, dass ein Modell weder eine gute Performance bei den Lerndaten, noch bei neuen Datensätzen, aufweisen kann [39]. Underfitting kann sehr leicht erkannt werden. Entweder anhand des ausbleibenden Lernfortschritt während des Lernens, oder anhand schlechter Vorhersagen des Modells. *Overfitting* [42, vgl. S.947 ff.] beschreibt den Sachverhalt, dass ein Modell die Lerndaten so exakt widerspiegelt, dass das Modell neue, unbekannte Datensätze nicht mehr korrekt vorhersagen kann. Das bedeutet, dass das Modell eine sehr geringe Fehlerrate gegenüber den Lerndaten, mit denen das Modell erlernt wurde, aufweist. Neue Datensätze allerdings, resultieren in sehr hohen Fehlerraten. Als Entwickler gilt es also ein Modell zu entwickeln, das sich in einem Optimum zwischen beiden Extremen befindet.

Bei der Wahl der Architektur, müssen die Parameter (Anzahl versteckte Schichten und Anzahl Neuronen pro versteckte Schicht) so gewählt werden, dass Under- wie auch Overfitting verhindert wird. Bezüglich des Vorgehens zur Wahl dieser Netzwerkparameter, ist sich die Wissenschaft uneinig. Publikationen wie [6, 47, 4] stellen Daumenregeln zur Wahl der Netzwerkparameter bereit. Blum behauptet [6] die Anzahl der Neuronen in den versteckten Schichten, sollte zwischen der Anzahl an Eingabe- und Ausgabe-Neuronen liegen. Swingler [47], wie auch Berry und Linoff [4] beteuern, dass sich niemals mehr Neuronen in einer versteckten Schicht befinden sollten, als die doppelte Anzahl der Eingabe-Neuronen. Andere Quellen [23] bezeichnen diese Daumenregeln als nutzlos, da andere wichtige Faktoren, wie beispielsweise die Anzahl der Lerndatensätze, außer Acht gelassen werden. Elisseeff und Paugam-Moisy stellen in ihrer Publikation „Size of multilayer networks for exact learning: analytic approach“ [12] ein Theorem, zur Berechnung der unteren und oberen Grenze der Anzahl Neuronen vor. Im Gegensatz zu den Ansätzen von [6, 47, 4], werden im Ansatz von Elisseeff und Paugam-Moisy, auch die Anzahl an Lerndatensätzen betrachtet. Darüber hinaus, treffen Elisseeff und Paugam-Moisy keine Aussage über die optimale Anzahl versteckter Neuronen, sondern bieten eine Formel

zur Berechnung der minimal notwendigen (Formel 3.2), beziehungsweise maximal ausreichenden Anzahl an Neuronen (Formel 3.3).

$$N_H = \left[\frac{N_P N_S}{N_I + N_S}\right] \tag{3.2}$$

$$N_H = 2\left[\frac{N_P N_S}{N_I + N_S}\right] N_S \tag{3.3}$$

Dabei beschreibt N_P die Anzahl der Lerndatensätze, N_S die Anzahl der Ausgabeneuronen (bzw. die Anzahl möglicher Labels) und N_I die Anzahl der Eingabeneuronen (bzw. die Anzahl der Features des Lerndatensatzes). Für diesen Lerndatensatz, ergibt sich unter der Verwendung von $N_P = 100000$, $N_S = 2$ und $N_I = 83$ einen Wert von $N_H \approx 2350$ notwendigen Neuronen pro Schicht (nach Formel 3.2).

Nun gilt es die Anzahl der versteckten Schichten zu bestimmen. Zur Abbildung eines linearen Problems, ist eine Ein- und Ausgabe-Schicht schon ausreichend [31]. Zur Erlernung komplexer, nicht linearen Zusammenhängen müssen, abhängig von den gewählten Aktivierungsfunktionen, mindestens ein bis zwei versteckte Schichten verwendet werden [22, 26].

Exkurs 5 (Vanishing Gradient Problem). *Vanishing Gradient Problem* beschreibt ein Problem, das beim Lernen, unter der Verwendung mehrschichtiger neuronale Netze, auftritt. Das Problem besteht darin, dass bei der Verwendung von n-Schichten die Fehlerrate, mittels der die Gewicht-Updates berechnet werden, exponentiell mit n sinkt. Dies führt zu einem langsameren Lernverhalten in den vorderen Schichten und dadurch, zu einem generell schlechteren, oder sogar ausbleibendem Lernverhalten. Sepp Hochreiter [51] beschrieb das Vanishing Gradient Problem 1991 als Erster. Es gilt bis heute, als große Herausforderung bei der Verwendung (mehrschichtiger) neuronaler Netze.

Um dem Vanishing Gradient Problem entgegen zu wirken, werden in diesem Anwendungsfall, so wie in [22, 26] empfohlen, zwei versteckte Schichten verwendet. Sollte festgestellt werden, dass zwei Schichten nicht ausreichend sind um die Komplexität des

abzubildenden Problems festzuhalten, könnte gegebenenfalls ein Benchmark mit einem dreischichtigen neuronalen Netz durchgeführt werden.

Exkurs 6 (**Regularisierung**)**.** *Regularisierung* beschreibt Techniken, deren Ziel es ist Overfitting (siehe Exkurs 4) zu verhindern [52].

Dahl et al. [9] empfehlen die Kombination der Regularisierungs-Techniken *Early stopping* und *Dropout* zur Verhinderung von Overfitting. Beide Techniken könnten auch einzeln verwendet werden, allerdings beweisen Dahl et al., dass sich durch die Kombination der Techniken die Fehlerrate der Vorhersagen des Modelles zusätzlich verbessert.

Exkurs 7 (**Early stopping**)**.** *Early stopping* beschreibt eine Technik, zur Verhinderung von Overfitting. Typischerweise wird vor dem Lernen des Netzes spezifiziert, wie oft der komplette Lerndatensatz das neuronale Netz passiert (Epoche). Wird der Parameter zu klein gewählt, könnte Underfitting eintreten. Wird er zu groß gewählt, könnte Overfitting eintreten. Early Stopping übernimmt die manuelle Wahl der Anzahl an Epochen, indem ein Teil der Lerndaten in einen Testdatensatz überführt wird. Anschließend werden nach jeder Epoche, die Vorhersagen des Models auf den Testdatensatz geprüft. Haben sich die Vorhersagen des Models in dieser Epoche verbessert, überschreibt der Algorithmus das alte Model und beginnt eine neue Epoche. Wenn nicht behält er das alte Model, verwirft das neu erlernte Model und beginnt eine neue Epoche.

Die bereits generierten Lerndaten, wurden für die Umsetzung von Early Stopping aufgeteilt. Hierzu werden die Datensätze der Lerndaten zuerst gemischt, so dass diese nicht mehr in einer zeitlichen Reihenfolge stehen. Anschließend werden 10000 Datensätze aus den Lerndaten ausgeschnitten und in eine weitere Textdatei kopiert, diese Datensätze werden ausschließlich zur Regularisierung mittels Early stopping verwendet.

Die nächsten Abschnitte:

Dropout

Zusammenfassung plus Überleitung

3.3. Modell zur Erkennung von Betrugsversuchen

3.3.1. Lerndaten

3.3.2. Modellbildung

4. Implementierung

- Dieses Kapitel beschreibt die Implementierung des (name?) Prototypen den wir zuvor konzipiert haben

4.1. Datenschema

- Ausgehend von den Zuvor in 3.2.1 und 3.2.2 beschriebenen Konzepten soll nun die Implementierung des Datenschemas beschrieben werden.

- H2-DB

- Sql-Script / Datenbankinhalt

- CSV Files bzw. deren Ersatz (in der Spätren Implementierung)

- Mit der erfolgreichen Implementierung der Datenschicht, haben wir nun eine Grundlage mit der wir später unser ML Verfahren sowie unsere Decison Engine füttern können.

4.2. Decision-Engine

- Implementierung Decision Engine

- Nach Implementierung der Decision Engine muss nun das ML Verfahren implementiert werden, das unsere Entscheidungen verbessern soll

4.3. Neuronales-Netzwerk

- Implementierung nach unseren Anforderungen. 1. Lernen können und 2. Evaluieren können.

- Erklären inwiefern DeepLearning4J leicht auf andere UseCases angewendet werden kann (Um das Kriterium der Allgemeingültigkeit zu erfüllen)

4.4. Optimierung

- Die Strukturen des NN wurden gesetzt. Nun müssen noch die Lernparameter bestimmt werden.

- Early Stopping using Best Model

- Trial and Error

- Tabelle aus meiner PPT

- Evaluierungsdaten können auch gelernt werden

5. Analyse

- Nach dem die Problemstellungen Px und Px gelöst wurden. - Kann die eigentliche Forschungsfrage dieser Thesis beantwortet werden

5.1. Vorgehensweise

- Hierzu sollen die Ergebnisse unseres Prototypen mit den Output-Werten der Decision Engine verglichen werden.

- Anhand der zwei Use Cases soll Verglichen werden, welches Verfahren (ML vs. DMN) statistisch bessere Ergebnisse liefert

5.2. Erkennung von Kreditausfällen

- Tabelle mit statistischen Werten

5.3. Erkennung von Betrugsversuchen bei Banktransaktionen

- Tabelle mit statistischen Werten

5.4. Zusammenfassung der Ergebnisse

- Tabelle mit statistischen Werten

6. Diskussion

In diesem Kapitel werden die Ergebnisse der vorliegenden Arbeit diskutiert und bewertet

-> Wie aussagekräftig sind die Ergebnisse aus 5.1 bzw. 5.2

-> Was bedeutet das für die Forschungsfrage

-> Welches Nutzen hat das für Anwender

-> Integration ACTICO Platform

-> White Box <-> Blackbox

7. Zusammenfassung und Ausblick

- Diese Kapitel fasst die vorliegende Arbeit zusammen und soll darüber hinaus einen Ausblick geben wie der Prototyp weiter entwickelt werden könnte, bzw. wie ein Pilot-Projekt aussehen könnte

Literaturverzeichnis

[1] Aaron J. Katzel, Thomas A. Russo. *The 2008 Financial Crisis and its Aftermath: Confronting the Next Debt Challenge*. Thomas Russo, 2010. URL: https://books.google.de/books?id=u515NZjrFuMC.

[2] Alpaydin, Ethem. *Machine Learning: The New AI (The MIT Press Essential Knowledge series)*. The MIT Press, 2016. ISBN: 978-0262529518.

[3] Barga, Roger et al. *Predictive analytics with Microsoft Azure machine learning*. Springer, 2015.

[4] Berry, Michael J. and Gordon Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Support*. New York, NY, USA: John Wiley & Sons, Inc., 1997. ISBN: 0471179809.

[5] Bloem, Adriaan M. and Russel Freeman. "The Treatment of Nonperforming Loans". In: *Statistics Department International Monetary Fund* (June 2005), pp. 1–15. URL: http://www.imf.org/external/pubs/ft/bop/2005/05-29.pdf.

[6] Blum, Adam. *Neural Networks in C++: An Object-oriented Framework for Building Connectionist Systems*. New York, NY, USA: John Wiley & Sons, Inc., 1992. ISBN: 0-471-53847-7.

[7] Brunnermeier, Markus K. "Deciphering the Liquidity and Credit Crunch 2007-2008". In: *Journal of Economic Perspectives* 23.1 (Mar. 2009). URL: http://www.aeaweb.org/articles?id=10.1257/jep.23.1.77.

[8] Cranganu, C., H. Luchian, and M.E. Breaban. *Artificial Intelligent Approaches in Petroleum Geosciences*. Springer International Publishing, 2015. ISBN: 9783319165318. URL: https://books.google.de/books?id=L5xwCAAAQBAJ.

[9] Dahl, George E, Tara N Sainath, and Geoffrey E Hinton. *Improving deep neural networks for LVCSR using rectified linear units and dropout*. IEEE, 2013.

[10] *Die Compliance Suite der ACTICO GmbH*. Letzter Aufruf 13.06.2017. 2017. URL: https://www.actico.com/de/loesungen/compliance.

[11] *Discover Feature Engineering, How to Engineer Features and How to Get Good at It*. Letzter Aufruf 13.06.2017. 2017. URL: http://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get-good-at-it/.

[12] Elisseeff, André and Hélène Paugam-Moisy. "Size of multilayer networks for exact learning: analytic approach". In: *Advances in Neural Information Processing Systems*. 1997, pp. 162–168.

[13] Engelmann, B. and R. Rauhmeier. *The Basel II Risk Parameters: Estimation, Validation, Stress Testing - with Applications to Loan Risk Management*. Springer Berlin Heidelberg, 2011. ISBN: 9783642161148. URL: https://books.google.de/books?id=ZgTEeXK8q8wC.

[14] Ertel, Wolfgang. "Neuronale Netze". In: *Grundkurs Künstliche Intelligenz: Eine praxisorientierte Einführung*. Wiesbaden: Springer Fachmedien Wiesbaden, 2016, pp. 265–311. ISBN: 978-3-658-13549-2. DOI: 10.1007/978-3-658-13549-2_9. URL: http://dx.doi.org/10.1007/978-3-658-13549-2_9.

[15] *Fannie Mae Single-Family Loan Performance Data*. Letzter Aufruf 16.05.2017. 2017. URL: http://www.fanniemae.com/portal/funding-the-market/data/loan-performance-data.html.

[16] Fight, Andrew. *Credit Risk Management*. Essential Capital Markets. Elsevier Science, 2004. ISBN: 9780080472409. URL: https://books.google.de/books?id=wd15dswnQWgC.

[17] Garnaut, R. and D. Llewellyn-Smith. *The Great Crash Of 2008*. Melbourne University Publishing, 2009. ISBN: 9780522860016. URL: https://books.google.de/books?id=hhayaM-knEIC.

[18] Governors of the Federal Reserve System, Board of. "Press Release announcing: "proposal designed to ensure that incentive compensation policies of banking organizations do not undermine the safety and soundness of their organizations"". In: (2009). URL: https://www.federalreserve.gov/newsevents/press/bcreg/20091022a.htm.

[19] Group, Object Management. *Decision Model and Notation (DMN) - Version 1.1*. Letzter Aufruf: 26. April 2017. URL: http://www.omg.org/spec/DMN/1.1/PDF/.

[20] Hardaway, R.M. *The Great American Housing Bubble: The Road to Collapse*. Praeger, 2011. ISBN: 9780313382284. URL: https://books.google.de/books?id=%5C_UKK41Qh8kcC.

[21] Haykin, Simon S. *Neural Networks: A Comprehensive Foundation*. International edition. Prentice Hall, 1999. ISBN: 9780132733502. URL: https://books.google.de/books?id=bX4pAQAAMAAJ.

[22] *How many hidden layers should I use?* Letzter Aufruf 27.06.2017. 2014. URL: http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-9.html.

[23] *How many hidden units should I use?* Letzter Aufruf 19.06.2017. 2014. URL: http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html.

[24] Hurd, Michael D and Susann Rohwedder. *Effects of the financial crisis and great recession on American households*. Tech. rep. National Bureau of Economic Research, 2010. URL: http://www.nber.org/papers/w16407.pdf.

[25] *Kreditrisikomanagement und Ratingverfahren*. Letzter Aufruf 08.06.2017. 2010. URL: http://www.finance.uni-mainz.de/Dateien/KreditrisikomanagementI_1.pdf.

[26] Larochelle, Hugo et al. "Exploring strategies for training deep neural networks". In: *Journal of Machine Learning Research* 10.Jan (2009), pp. 1–40.

[27] Lippmann, Richard. "An introduction to computing with neural nets". In: *IEEE Assp magazine* 4.2 (1987), pp. 4–22.

[28] *Loan Performance Data*. Registrierung erforderlich - Letzter Aufruf 16.05.2017. 2017. URL: https://loanperformancedata.fanniemae.com/lppub/index.html.

[29] Marco Wiering, Martijn van Otterlo. *Reinforcement Learning - State-of-the-Art*. Springer, 2012. ISBN: 978-3-642-27644-6. URL: https://link.springer.com/book/10.1007/978-3-642-27645-3.

[30] Masanori Fujita, Nicolaj Kirchhof. "Data Analytics in der Praxis - Regressionsanalyse". In: *Entwickler Magazin* 2017/03 (2017).

[31] McCullagh, P. and John A. Nelder. *Generalized Linear Models, Second Edition (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman and Hall/CRC, 1989. ISBN: 0412317605. URL: https://www.amazon.com/Generalized-Chapman-Monographs-Statistics-Probability/dp/0412317605?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0412317605.

[32] Mitchell, Tom M et al. *Machine learning. WCB*. 1997.

[33] Munoz, Andres. *Machine Learning and Optimization*. 2014. URL: https://www.cims.nyu.edu/~munoz/files/ml_optimization.pdf.

[34] *Name Matching Customer NMC*. Letzter Aufruf 13.06.2017. 2017. URL: https://www.actico.com/de/produkte/name-matching-customer-nmc.

[35] Nationalbank, Österreichische. "„Kreditvergabeprozess und Kreditrisikomanagement"". In: *Leitfadenreihe zum Kreditrisiko, Wien* (2004). Letzter Aufruf 08.06.2017. URL: https://www.oenb.at/dam/jcr:07fa4f29-4487-4b1d-a501-c01a9eaf65b2/leitfadenreihe_kreditvergabe_tcm14-11170.pdf.

[36] Naudé, Wim. "The financial crisis of 2008 and the developing countries". In: 2009/01 (2009). URL: http://hdl.handle.net/10419/84665.

[37] *Official Website of the KNIME Data Mining Tool*. Letzter Aufruf 13.06.2017. 2017. URL: https://www.knime.org/.

[38] *Official Website of the Weka Data Mining Tool*. Letzter Aufruf 13.06.2017. 2017. URL: http://www.cs.waikato.ac.nz/ml/weka/.

[39] *Overfitting and Underfitting With Machine Learning Algorithms*. Letzter Aufruf 19.06.2017. 2016. URL: http://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/.

[40] *Richtlinie 2013/36/EU des Europäischen Parlaments und des Rates*. Letzter Aufruf 08.06.2017. 2013. URL: http://eur-lex.europa.eu/legal-content/DE/TXT/PDF/?uri=CELEX:32013L0036.

[41] Rücker, Bernd. "Decision Model and Notation: Digitalisierung von Entscheidungen mit DMN". In: *OBJEKTSpektrum* 2016/05 (2016). URL: https://www.sigs-datacom.de/uploads/tx_dmjournals/ruecker_OS_05_16_TRBT.pdf.

[42] "Unsupervised Learning". In: *Encyclopedia of Machine Learning and Data Mining*. Ed. by Sammut, Claude and Geoffrey I. Webb. Boston, MA: Springer US, 2017, pp. 1304–1304. ISBN: 978-1-4899-7687-1. DOI: 10.1007/978-1-4899-7687-1_976. URL: http://dx.doi.org/10.1007/978-1-4899-7687-1_976.

[43] Saunders, A. and L. Allen. *Credit Risk Management In and Out of the Financial Crisis: New Approaches to Value at Risk and Other Paradigms*. Wiley Finance. Wiley, 2010. ISBN: 9780470622360. URL: https://books.google.de/books?id=Rr%5C_6y9evvowC.

[44] Schneider, H. *Historie der neuronalen Netze*. Letzter Auffgfgruf: 02. Mai 2017. URL: https://www.fbi.h-da.de/fileadmin/personal/h.schneider/public_html/ma-c-hi/arbeitsblaetter/nn-historie.pdf.

[45] *Solving the $190 billion Annual Fraud Problem*. Letzter Aufruf 19.05.2017. 2011. URL: https://www.forbes.com/sites/haydnshaughnessy/2011/03/24/solving-the-190-billion-annual-fraud-scam-more-on-jumio/#35fa8393390e.

[46] Strecker, Stefan. *Künstliche Neuronale Netze – Aufbau und Funktionsweise*. Letzter Auffgfgruf: 02. Mai 2017. Oct. 1997. URL: http://geb.uni-giessen.de/geb/volltexte/2004/1697/pdf/Apap_WI_1997_10.pdf.

[47] Swingler, K. *Applying Neural Networks: A Practical Guide*. Academic Press, 1996. ISBN: 9780126791709. URL: https://books.google.com.br/books?id=bq0YnP4BNKsC.

[48] Taylor, J. *Decision Management Systems: A Practical Guide to Using Business Rules and Predictive Analytics*. IBM Press. Pearson Education, 2011. ISBN: 9780132884440. URL: https://books.google.de/books?id=X2pJuopB2JsC.

[49] *The True Cost of Fraud Study*. Letzter Aufruf 19.05.2017 - Registrierung erforderlich. 2017. URL: http://images.solutions.lexisnexis.com/Web/LexisNexis/%7Bea78e9df-056e-46ed-b04c-e8bfbc526ffd%7D_2016_True_Cost_of_Fraud_Study_052516.pdf?elqTrackId=cc6d22e0b40d4a29a2931f28fb221092&elqaid=2567&elqat=2.

[50] Thomas Theobald Silke Tober, Benjamin Lojak. "IMK Finanzmarktstabilitätsreport 2016". In: *Institut für Makroökonomie und Konjunkturforschung* (Feb. 2017), pp. 1–22. URL: https://www.boeckler.de/pdf/p_imk_report_121_2017.pdf.

[51] *Untersuchungen zu dynamischen neuronalen Netzen*. Letzter Aufruf 27.06.2017. 1991. URL: http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhube
pdf.

[52] *What is regularization in machine learning?* Letzter Aufruf 27.06.2017. 2017. URL: https://www.quora.com/What-is-regularization-in-machine-learning.

[53] Wojtusiak, Janusz. "Machine Learning". In: *Encyclopedia of the Sciences of Learning*. Ed. by Seel, Norbert M. Boston, MA: Springer US, 2012. ISBN: 978-1-4419-1428-6. DOI: 10.1007/978-1-4419-1428-6_1927. URL: http://dx.doi.org/10.1007/978-1-4419-1428-6_1927.

[54] Zhu, Xiaojin. “Semi-Supervised Learning”. In: *Encyclopedia of Machine Learning*. Ed. by Sammut, Claude and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 892–897. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_749. URL: http://dx.doi.org/10.1007/978-0-387-30164-8_749.

A. Anhang

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Hier folgt Beispiel A.1 für Quellcode:

Listing A.1: XML-Konfiguration: settings.xml.

```
<servers>
  <server>
    <id>xxx</id>
    <username>(Benutzername)</username>
    <password>(Passwort)</password>
  </server>
  <server>
    <id>yyy</id>
    <username>(Benutzername)</username>
    <password>(Passwort)</password>
  </server>
</servers>
```