Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

Analyse von Visitentexten mittels maschinellem Lernen zur Vorhersage medizinischer Scores bei Intensivpatienten

Bachelorarbeit

zur Erlangung des akademischen Grades
Bachelor of Science (B. Sc.)

eingereicht von: Martin Hoffmann
geboren am: 06.09.1998
geboren in: Berlin

Gutachter: Prof. Dr. med. Dr. rer. nat. Felix Balzer
Dr. med. Fridtjof Schiefenhövel

eingereicht am: ..... verteidigt am: .....

Zusammenfassung

Die Verlegung eines Patienten auf eine Intensivstation erfolgt häufig infolge einer besonders schweren oder lebensbedrohlichen Erkrankung oder Verletzung. Insbesondere die digitale Kommunikation über den Gesundheitszustand der Patienten ist eine wichtige Vorraussetzung für ein optimales Therapieoutcome. Dennoch kommt es häufig zu unvollständigen, ungenauen oder widersprüchlichen Eintragungen. Das Ziel der vorliegenden Arbeit ist es, mithilfe von modernen Ansätzen des maschinellen Lernens medizinische Scores anhand von unstrukturierten Visitentexten vorherzusagen, um solche Missstände zu quantifizieren und zu beheben.

Inhaltsverzeichnis

1 Einführung

1.1 Datenerfassung auf Intensivstationen

Marx et al. [2015] verorten die Intensivmedizin im Spannungsfeld zwischen Heilen und Sterben. Eine Verlegung auf die Intensivstation erfolgt häufig infolge einer besonders schweren oder lebensbedrohlichen Erkrankung oder Verletzung. Ziel der Behandlung ist es, sofern möglich, den Patienten[1] so zu kurieren, dass diesem ein Weiterleben unabhängig von den besonderen technischen und personellen Möglichkeiten der Intensivmedizin möglich ist. Dafür wird eine besonders intensive Behandlung durch Ärzte und Pflegekräfte benötigt.

Technische Fortschritte und die Digitalisierung ermöglichen es, mehr Informationen über die Patienten zu erfassen und zu verarbeiten als je zuvor. Ärzte, Pflegekräfte, aber auch Angehörige werden so mit umfangreichen Mengen an Informationen konfrontiert. Dabei stellt die erfolgreiche Kommunikation zwischen Behandelnden eine wichtige Vorraussetzung für das Patientenoutcome dar. Marx et al. [2015] benennen Kommunikationsprobleme als wichtigen Faktor für erhöhte Krankenhausmortalitätsraten. Die Autoren beschreiben weiter, dass bis zu 50 % der klinisch relevanten Informationen, die noch in der Morgenvisite zwischen Ärzten ausgetauscht werden, schon in der Spätvisite des gleichen Tages nicht mehr übermittelt werden.

Die Frage der effektiven Datenerfassung auf Intensivstationen ist somit eine über Leben und Tod. Dennoch kommt es aus unterschiedlichen Gründen vor, dass Informationen über den Gesundheitszustand der Patienten ungenau oder in zu geringem Umfang digital erfasst werden.

Gegenstand der vorliegenden Arbeit ist der Versuch, mittels maschinellem Lernen einen Beitrag zur Lösung dieses Problems zu leisten.[2]

[1] Hier und im Rest der Arbeit umfasst das generische Maskulinum, sofern nicht anders angegeben, Personen beider Geschlechter.

[2] Der komplette Programmcode der Modelle, Skripte zur Datenaufbereitung und mehr liegen in folgendem Git-Repository vor: `https://github.com/mahathu/bachelorarbeit`

1.1.1 medizinische Scores

In *Die Intensivmedizin* [Marx et al., 2015] ist der Begriff des Scores folgendermaßen definiert:

> *„Ein Score ist der Versuch, eine komplexe klinische Situation auf einen eindimensionalen Punktwert abzubilden. Eine solche Reduktion verfolgt das Ziel, übergreifende Aspekte wie Schweregrad oder Prognose als Kombination einzelner Fakten objektiv zu fassen, um sie dann in unterschiedlichen Kollektiven vergleichend darstellen zu können.“*

Es handelt sich bei einem Score häufig um die Kombination mehrerer erfassbarer Werte, beispielsweise der Herzfrequenz oder dem Sauerstoffgehalt im Blut. Auch allgemeine Informationen über den Patienten wie das Alter oder bekannte Vorerkrankungen können je nach Score berücksichtigt werden. Die Bestimmung eines Score-Werts stellt also den Versuch dar, die komplexe, individuelle Situation eines Patienten auf einen numerischen Wert zu reduzieren. Dabei gehen unweigerlich Informationen verloren. Gleichzeitig erlaubt es die Erfassung von derartigen standardisierten Scores aber, auf einen Blick wichtige Informationen über den Zustand des Patienten zu erfassen. Ferner wird durch eine solche Reduktion auf das Wesentliche ermöglicht, den pathologischen Verlauf eines Patienten über einen längeren Zeitraum zu analysieren, oder die Symptomatik mehrerer Patienten leichter miteinander zu vergleichen. Ein weiterer Vorteil ist dass, unter Voraussetzung der richtigen Anwendung, die Vergabe von Scores weitestgehend unabhängig von der subjektiven Einschätzung des Arztes oder der Pflegekraft erfolgt [Marx et al., 2015]. Die Frage, ob es sich bei der Vorhersage der im Rahmen dieser Arbeit behandelten Scores um ein Regressions- oder ein Klassifikationsproblem handelt, wird in Abschnitt 1.2.3 weiter vertieft.

1.1.2 Ziel der Arbeit

Das Ziel der vorliegenden Arbeit ist es, mit Hilfe von maschinellem Lernen ein statistisches Modell zu entwickeln, um anhand von Freitexten medizinische Scores möglichst akkurat vorherzusagen. Die Entwicklung eines solchen Modells ermöglicht es unter anderem, die tatsächlich eingetragenen Werte mit den Vorhersagen des Modells zu vergleichen, um daraus Rückschlüsse über die Qualität der Datenerfassung an der Charité zu treffen.

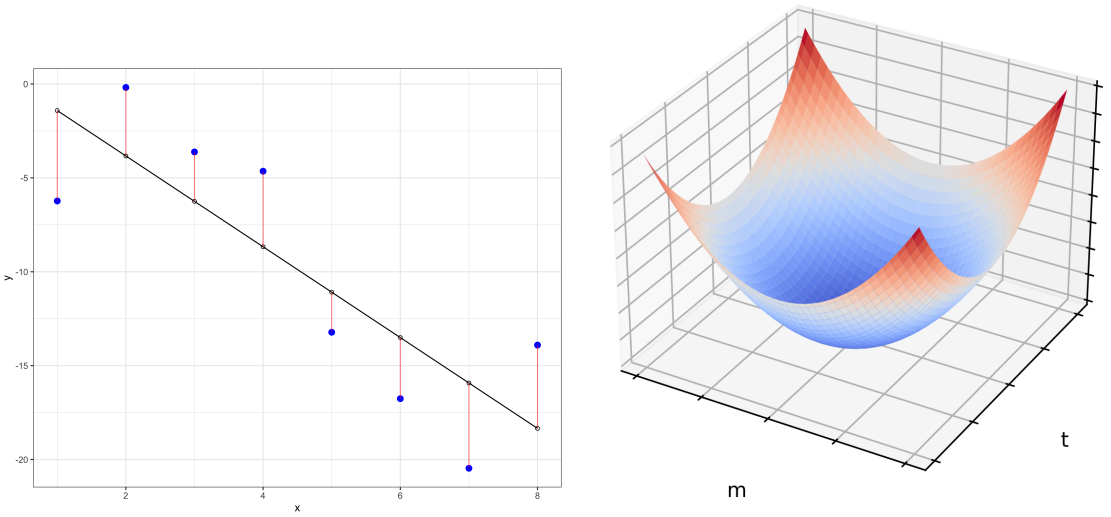
1.2 Maschinelles Lernen

Der Begriff Maschinelles Lernen kennzeichnet einen modernen Ansatz in der Forschung an künstlicher Intelligenz. Obgleich die Anfänge des maschinellen Lernens bereits mehrere Jahrzehnte zurückliegen [Samuel, 1959; Nilsson, 1965], hat es sich im Zeitalter von Big Data und besonders leistungsstarken Rechnern zu einem der Forschungsschwerpunkte der modernen angewandten Informatik entwickelt. Algorithmen aus dem Bereich des Maschinellen Lernens ermöglichen es, anhand von Eingabedaten Vorhersagen über Eigenschaften noch unbekannter Daten zu treffen [Mitchell, 1997]. Im gleichen Werk definiert der Autor diesen Prozess folgendermaßen:

> *"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."*

„Lernen" bezeichnet hierbei die besondere Fähigkeit eines Computerprogramms, sich anhand neuer Eingabedaten selbstständig anzupassen und zu verbessern. Dies unterscheidet das maschinelle Lernen von herkömmlichen Verfahren der inferentiellen Statistik. Im Allgemeinen wird der Prozess des Lernens durch den Versuch modelliert, eine vorher festgelegte, wohldefinierte Verlustfunktion zu minimieren. Die dafür existieren numerischen Optimierungsverfahren sind ein grundlegender Untersuchungsgegenstand in der Forschung am maschinellen Lernen.

Bei Regressionsproblemen stellt beispielsweise die mittlere absolute Abweichung (*mean absolute error*) der Vorhersagen zu den tatsächlichen Werten eine solche Verlustfunktion dar, die es zu minimieren gilt. Abbildung 1.1 zeigt einen Datensatz, dessen Werte einem annähernd linearen Verlauf folgen. Ihr Zusammenhang lässt sich also anhand einer Regressionsgerade $f(x) = mx + t$ (schwarze Linie) modellieren. Je nach Wahl von m und t ist die Funktionsgerade mehr oder weniger geeignet, die tatsächliche Verteilung der Werte zu beschreiben. Je besser die beiden Funktionsparameter gewählt werden, desto geringer sind die Residuen (rote Linien), und desto mehr nähert sich die Verlustfunktion ihrem Minimum an.



(a) lineare Regression

(b) Die Verlustfunktion hat ein Minimum, das erreicht wird, wenn m und t optimal gewählt werden.

Abbildung 1.1

1.2.1 Überwachtes Lernen

Algorithmen basierend auf Datensätzen, die sowohl Eingabedaten als auch die entsprechenden Ausgabedaten enthalten werden dem überwachten Lernen (*supervised learning*) zugeordnet[Russell und Norvig, 2020]. Es liegen also Wertepaare vor, die zu jedem Eingabewert auch den gesuchten Ausgabewert enthalten. Im Falle der vorliegenden Arbeit handelt es sich bei den Eingabedaten um Visitentexte und bei den Ausgabedaten um den dazugehörigen Wert eines bestimmten Scores.

Bei der Entwicklung eines solchen Modells werden die vorhandenen Datenpaare in zwei disjunkte Teilmengen aufgeteilt: Die Trainingsmenge enthält die Mehrheit (oft 80 - 90 %) der Wertepaare, während der Rest der Testmenge zugeordnet wird. Diese Aufteilung kann rein zufällig erfolgen, häufig wird allerdings versucht, die Häufigkeitsverteilung der möglichen Ein- und Ausgabewerte in beiden Mengen beizubehalten.

Ein neues Modell wird zunächst anhand der Paare im Trainingsset trainiert. Erst danach wird die Leistung des Modells bewertet, indem seine Vorhersagen auf die Eingabedaten der Testmenge mit den tatsächlichen Werten verglichen werden. Diese Unterteilung

ist notwendig, um eine Vorstellung der Leistung des Modells bei unbekannten Daten zu ermöglichen. Erst wenn auch bei Vorhersagen auf der Testmenge eine zufriedenstellende Genauigkeit erreicht wird hat das Modell einen praktischen Nutzen bei der Vorhersage von tatsächlich unbekannten Werten.

1.2.2 Overfitting

Wird ein Modell zu lange oder auf einem zu kleinen Trainings-Datensatz trainiert, werden unter Umständen in den Eingabedaten Muster erkannt, die nicht vorhanden sind, und zum Beispiel lediglich als Folge statistischer Ungenauigkeiten auftreten. Auf die Trainingsdaten wird so eine sehr hohe prädiktive Genauigkeit erreicht, die sich allerdings nicht auf unbekannte Daten übertragen lässt, da diese die vermeintlichen Muster nicht widerspiegeln. Dieses Problem, im maschinellen Lernen als overfitting (Überanpassung) bezeichnet, wird in Abbildung 1.2 anhand eines Beispiels verdeutlicht. Hier stellt die orangefarbene Linie ein Modell dar, das zwar sehr gut an die gegebenen Trainingspaare angepasst ist, aber bei neuen Werten deutlich schlechtere Vorhersagen treffen wird als das einfachere Modell (grüne Linie). Das erste Modell wurde in Anbetracht der wenigen Trainingsdaten zu lange trainiert und leidet unter overfitting.

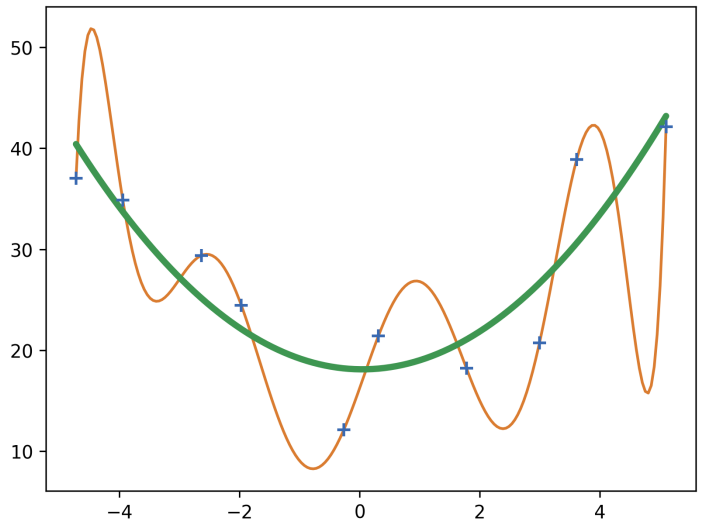


Abbildung 1.2: Eine Interpolationspolynom bildet die gegebenen Werte zwar exakt ab, spiegelt aber die tatsächliche Anordnung der Daten schlechter wieder als die grüne Linie.

Neben dem überwachten Lernen existieren weitere Arten des maschinellen Lernens, die aber nicht weiter Gegenstand dieser Arbeit sind. Insbesondere das unüberwachte Lernen, bei dem ein Algorithmus versucht, in ungelabelten Datensätzen vorher noch unbekannte Muster zu erkennen [Russell und Norvig, 2020], hat hohe Relevanz in anderen Anwendungsfällen des maschinellen Lernens.

1.2.3 Regression oder Klassifikation

Probleme aus dem Bereich des überwachten maschinellen Lernens lassen sich im Allgemeinen ferner in eine von zwei Unterkategorien einordnen: Regression und Klassifikation. Eine Regressionsanalyse verfolgt das Ziel, anhand von einer oder mehrerer unabhängiger Variablen Vorhersagen über eine abhängige Variable zu treffen. Bei der Klassifikation hingegen wird ein gegebener Wert einer oder mehreren Klassen aus einer endlichen Liste möglicher Klassen zugeordnet.[3] Da es sich bei den betrachteten medizinischen Scores um diskrete, ganzzahlige Werte aus einem endlichen Wertebereich handelt, liegt auch hier die Anwendung eines Klassifizierungs-Verfahrens nahe.

Würde man aber die verschiedenen möglichen Werte eines Scores als separate und voneinander unabhängige Klassen betrachten, so ginge die wichtige Information über deren Anordnung verloren. Mit Ausnahme des CAM-ICU[4] handelt es sich bei den im Rahmen dieser Arbeit behandelten Scores stets um eindimensionale, metrische Skalen. Im mathematischen Sinne stellen sie Totalordnungen dar: Sie erfüllen also die Anforderungen der Reflexivität, Antisymmetrie, Transitivität und Totalität. Bezeichne M die Menge aller möglichen Werte eines beliebigen medizinischen Scores. Es gilt dann für alle $a, b, c \in M$:

$$
\begin{aligned}
a \leq a & \qquad \text{(Reflexivität)} \\
a \leq b \wedge b \leq a \;\Rightarrow\; a = b & \qquad \text{(Antisymmetrie)} \\
a \leq b \wedge b \leq c \;\Rightarrow\; a \leq c & \qquad \text{(Transitivität)} \\
a \leq b \vee b \leq a & \qquad \text{(Totalität)}
\end{aligned}
$$

Entsprechend lassen sich die Werte eines Scores vergleichen und in ein Verhältnis

[3] Typische Anwendungsfälle sind beispielsweise Spam-Filter, die eine E-Mail als „Spam“ oder „Nicht Spam“ kategorisieren, oder Modelle zur Handschrifterkennung, bei dem jeweils ein handgeschriebenes Zeichen einem von endlich vielen möglichen Buchstaben aus einem gegebenen Alphabet zugeordnet wird.

[4] Dieser fällt entweder positiv oder negativ aus.

setzen. So ist ein RASS-Wert[5] von -4 (tief sediert) beispielsweise deutlich näher an -3 (mäßig sediert) als an $+1$ (unruhig). Bei gängigen Verfahren zur Klassifizierung ginge diese Information verloren, da bei Kenngrößen zur Bewertung solcher Modelle nur betrachtet werden kann, ob ein gegebener Eingabetext genau der richtigen Kategorie (dem richtigen Wert) zugeordnet wurde oder nicht.

Bei der vorliegenden Arbeit habe ich mich demnach dafür entschieden, die Vergabe von Scores anhand von Eingabetexten als klassisches Regressionsproblem zu betrachten und die Ausgaben der Modelle im Zweifelsfall auf den nächstmöglichen ganzzahligen Wert zu runden. Die Abweichung des vorhergesagten Werts eines Modells von dem nächstmöglichen Wert stellt somit sogar einen rudimentären Ansatz zur Bewertung der Konfidenz einzelner Vorhersagen dar.

1.2.4 Maschinelles Lernen in der Medizin

Das Vorhandensein großer, digital vorliegender Datensätze stellt eine geeignete Grundlage für ein breites Spektrum an Anwendungen des maschinellen Lernens in der Medizin dar [Chen et al., 2019]. Ziel dieses Abschnitts ist es, einen exemplarischen, aber keinesfalls erschöpfenden Überblick über einige weitere Anwendungsfälle zu geben.

Im Klinikalltag können Methoden des maschinellen Lernens Ärzten bei der Auswertung von Befunden sowie beim Erkennen klinisch relevanter Muster in Bilddaten unterstützen [Shah et al., 2019]. Auch bei der Diagnostik wird ML angewendet, beispielsweise zur computergestützten Auskultation [Reed et al., 2004]. In der Intensivmedizin werden mittels ML vorliegende Datensätze ausgewertet, unter anderem um Komplikationen vorherzusagen, Mortalitätsraten zu bestimmen und prognostische Modelle zu verbessern [Shillan et al., 2019; Krishnan und Sowmya Kamath, 2018].

Außerhalb der Klinik findet maschinelles Lernen auch im Bereich der Arzneimittelforschung Anwendung. Hier werden derartige Modelle unter anderem dazu eingesetzt, neue Targetproteine für bestimmte Wirkstoffe zu identifizieren, effizientere Wirkstoffkombinationen zu entwickeln und ein besseres Verständnis komplexer Krankheitsmechanismen zu erlangen [Vamathevan et al., 2019].

In der Virologie wird maschinelles Lernen unter anderem dazu angewandt, die Ausbreitung von Infektionskrankheiten zu modellieren [Arun und Neelakanta Iyer, 2020].

[5] Richmond Agitation-Sedation Scale

Weiterhin werden Risikofaktoren für eine Infektion ermittelt und Prädiktoren für die Mortalität bereits infizierter Personen identifiziert. Derartige Informationen ermöglichen es, Handlungsempfehlungen für behandelnde Ärzte und Pflegekräfte [Colubri et al., 2019] sowie politische Entscheidungsträger [Satu et al., 2020] zu formulieren.

Zuletzt sei die in der Medizin besonders relevante Problematik der Explainable AI erwähnt. Ohne spezielle Maßnahmen ähneln Modelle des maschinellen Lernens, insbesondere künstliche neuronale Netze, einer Art Black Box. Sie sind zwar oft in der Lage, beeindruckende Leistungen in ihrer Domäne zu erzielen, ihre genaue Wirkungsweise ist aber oft selbst für die Entwickler nicht vollumfänglich verständlich. Beispielsweise sollen sogenannte clinical decision support systems[6] Ärzten bei der Entscheidungsfindung im Klinikalltag assistieren, indem große Datenmengen automatisch analysiert und aufbereitet werden. Insbesondere bei Entscheidungen, die einen hohen Einfluss auf das Patientenoutcome haben, stellt sich die Frage, in welchem Maße den oft schwer nachvollziehbaren Empfehlungen einer Maschine vertraut werden sollte.

[6] klinische Entscheidungsunterstützungssysteme

2 Übersicht über die Daten

Auf den Intensivstationen der Charité wird das Patientendatenmanagementsystem COPRA eingesetzt, um Informationen wie etwa Patienten- und Behandlungsdaten zu erfassen, darzustellen und zu verarbeiten. Die Datensätze der vorliegenden Arbeit wurden beim Export aus dem System pseudonymisiert, indem patientenbezogene Daten wie Name und Geburtsdatum entfernt wurden. Insbesondere werden Zeitpunkte nicht in absoluter Form angegebenen, sondern in Relation zum Beginn des Krankenhausaufenthalts des entsprechenden Patienten. Rückschlüsse auf die Identität der Patienten, deren Daten hier betrachtet werden, sind somit ausgeschlossen. Eine Zuordnung der erfassten Werte und Visitentexte untereinander zum selben Patienten bleibt aber durch eine den Patienten eindeutig identifizierenden Zahlenkombination weiterhin möglich.

Nach dem Export lagen die Daten in Form mehrerer Datendateien vor. `patienten.csv` und `delir.csv` enthalten Metainformationen über die betrachteten Patienten. Die beiden Dateien `scores1.csv` und `scores2.xlsx` geben Aufschluss über die erfassten Scores und eingetragenen Freitexte. Jede Zeile enthält hierbei jeweils die VarID, den betreffenden Patienten, den Zeitpunkt sowie den eigentlichen erfassten Wert. Bei der VarID handelt es sich um einen ganzzahligen Wert, mit der jede Art von auf der Intensivstation erfasstem Wert bzw. eingetragenem Text Charité-intern repräsentiert und eindeutig identifiziert wird (siehe Tabelle 2.1).

2.1 vorliegende Datensätze

Der mir für diese Arbeit zur Verfügung gestellte Datensatz umfasst insgesamt 1357 Patienten-Aufenthalte auf den Intensivstationen der Charité Berlin aus dem Jahr 2017. Jeder Aufenthalt wird über eine numerische, inkrementell aufsteigende Nummer[1] eindeutig identifiziert. Ein Patient, der im Laufe seiner Behandlung mehrere Male auf eine Intensivstation verlegt wird, erhält für jeden separaten Aufenthalt eine neue Identifikationsnummer. Für jeden Patient liegen Informationen über Geschlecht, BMI[2], Alter zum

[1] in den exportierten Datensätzen als n_ID bezeichnet
[2] Body-Mass-Index

Zeitpunkt der Aufnahme und ob während des Aufenthalts ein Delir[3] diagnostiziert wurde vor. Weiterhin ist die Gesamtdauer des Aufenthalts auf der Intensivstation vermerkt, sowie ob der Patient während seines Aufenthalts verstorben ist. Bei den vorliegenden Patienten lag die Mortalität während des Aufenthalts bei etwa 17% ($n = 225$). Die beobachteten Aufenthalte verliefen über Zeiträume von wenigen Stunden bis zu mehreren Monaten. Die mittlere Aufenthaltsdauer während des Beobachtungszeitraums liegt bei etwa 7,2 Tagen, der Median beträgt 3,2 Tage. Fast die Hälfte aller Aufenthalte endete also nach weniger als drei Tagen. Nur etwas mehr als ein Viertel der Patienten ($n = 375$) wurden für eine Woche oder länger auf der Intensivstation behandelt.

Für jeden Patienten werden während der Dauer seines Aufenthalts medizinische Scores (siehe Abschnitt 1.1.1) bestimmt sowie Visitentexte geschrieben. Die Visitentexte werden digital erfasst und liegen im Unicode-Zeichensatz vor, folgen allerdings im Allgemeinen keinem einheitlichen Muster. Es handelt sich also um unstrukturierte Freitexte, und es liegt im Ermessen der behandelnden Ärzte bzw. Pflegekräfte, einen aussagekräftigen Text zu formulieren. Ebenso können sich Faktoren wie Zeitdruck, Stress oder Ablenkungen durch die Umgebung auf Umfang und Genauigkeit der eingetragenen Texte auswirken [Marx et al., 2015].

Tabelle 2.1 enthält eine Übersicht über alle Scores und Freitexte, die in dem gegebenen Datensatz erfasst wurden. Es folgt eine detailliertere Beschreibung derjenigen Werte, die für den Inhalt dieser Arbeit besondere Relevanz haben:

Glasgow Coma Scale Bei der Glasgow Coma Scale (GCS) wird die Schwere einer möglicherweise vorliegenden Bewusstseinsstörung anhand von drei Kategorien (Augenöffnen, verbale Antwort und motorische Reaktion) ermittelt. In jeder Kategorie wird eine Punktzahl ermittelt, die dann zu einem Endergebnis aufsummiert werden. Insgesamt sind so Werte zwischen einschließlich 3 und 15 möglich [Teasdale und Jennett, 1974; Marx et al., 2015].

RASS Die Richmond Agitation Sedation Scale (RASS) ist eine zehnstufige Messzahl, die den Grad der Sedierung eines Intensivpatienten beschreibt. Mögliche Werte liegen zwischen -5 („unarousable“/„nicht erweckbar“) und 4 („combative“/„streitlustig“). Bei

[3] F05.* gemäß ICD-10

[4] Entgegen der ursprünglichen Spezifikation ist an der Charité zusätzlich eine Eintragung des Werts 0 (für „wach, voll orientiert“) möglich.

VarID	Name	Wertebereich
20512769	Glasgow Coma Scale (GCS)	$v \in [3, 15]$
20512801	Behavior Pain Scale (BPS)	$v \in [3, 12]$
20512802	Delirium Detection Score (DDS)	$v \in [3, 35]$
22085815	Visite_ZNS	Freitext
22085820	Visite_Oberarzt	Freitext
22085836	Visite_Pflege	Freitext
22085897	Ramsay Sedation Scale	$v \in [0, 6]^4$
22085911	NRS/VAS (Visual Analogue Scale)	$v \in [0, 10]$
22086067	Vigilanz	Freitext*
22086158	Richmond Agitation Sedation Scale (RASS)	$v \in [-5, 4]$
22086169	CAM-ICU	$v \in \{\text{neg.}, \text{pos.}, \text{unmögl.}\}$
22086170	BPS-Bewertung	Freitext*
22086172	NRS/VAS Bedingungen	Freitext*

Tabelle 2.1: Übersicht aller erfassten Scores und Freitexte

den betrachteten Patienten wurde der Wert 0 („aufmerksam und ruhig“) am häufigsten erfasst, was dem erwünschten Wert entspricht, solange keine Sedierung indiziert ist. Die Autoren der Skala empfehlen eine Reihe von Stimuli, die dem Patienten präsentiert werden sollen, um eine besonders genaue Bestimmung des richtigen Wertes zu ermöglichen [Sessler et al., 2002]. Die RASS weist eine hohe Reliabilität und Validität auf und gilt im deutschsprachigen Raum als Goldstandard zum Monitoring der Sedierungstiefe [Marx et al., 2015; Müller et al., 2015].

Visite_ZNS, Visite_Oberarzt und Visite_Pflege bla bla bla (Fridtjof fragen)

2.2 Generierung von Schlüssel-Wert-Paaren

Eine besondere Herausforderung stellte dar, dass die verschiedenen Scores und Texte zu unterschiedlichen Zeiten und unabhängig voneinander eingetragen werden (siehe Abbildung 2.2). Um ein Machine-Learning Modell mit einer hohen Dimensionalität der Eingabevektoren zu trainieren ist eine hohe Anzahl von Trainingspaaren notwendig. Im konkreten Fall dieser Arbeit enthält jedes Wertepaar einen Visitentext sowie einen medi-

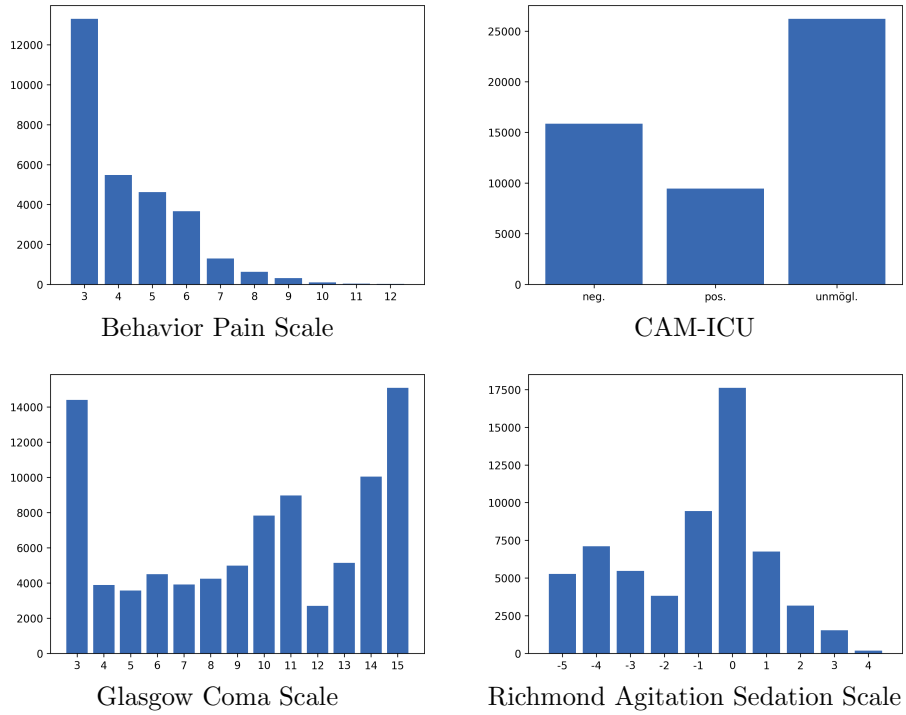


Abbildung 2.1: Histogramme einiger erfassten Scores

zinischen Score, der den in dem Text angegebenen Informationen über die Verfasssung des Patienten möglichst genau entspricht. Zusammen bilden diese Paare die Grundlage der Modelle, selbstständig noch nicht gesehene Texte bewerten zu können. Aufgrund der zeitlichen Unterschiede zwischen den Eintragungen von Texten und Scores erwies sich allerdings ebenjene Zuordnung zueinander als nicht trivial.

Ein möglicher naiver Ansatz wäre es, für einen bestimmten Zeitraum den Mittelwert aller Erhebungen eines Scores zu ermitteln, sowie alle Visitentexte, die in diesem Zeitraum erfasst wurden, aneinanderzuhängen. Wählt man beispielsweise einen Zeitraum von 4 Stunden, erhielte man so für jeden Patient, Visitentext und Score 6 Wertepaare pro Tag, mit denen sich ein statistisches Modell trainieren ließe.

Aus zwei Gründen habe ich mich allerdings für einen anderen Ansatz entschieden: Zum einen ist es durchaus möglich, dass sich ein bestimmter Wert innerhalb von wenigen Sekunden stark verändert – beispielsweise die RASS, wenn die Sedierung eines Patienten eingeleitet wird. Derartige Informationen könnten verloren gehen, wenn man von vornherein nur feste Zeiträume betrachtet. Weiterhin gibt es zwischen individuellen

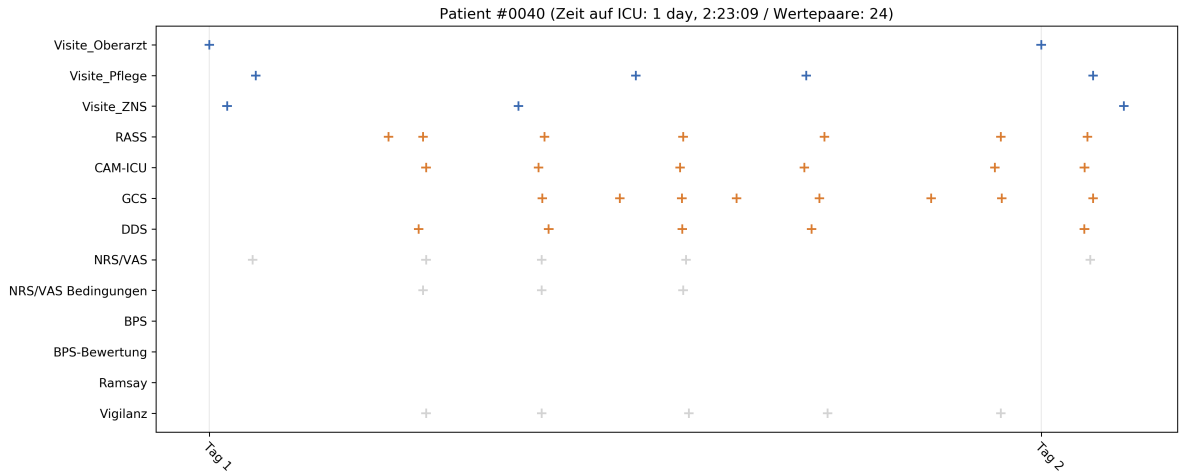


Abbildung 2.2: Die Visitentexte (blau) und Scores (orange) werden zeitlich unabhängig voneinander eingetragen bzw. erfasst.

Patienten hohe Schwankungen in der Frequenz und Häufigkeit, in denen bestimmte Scores erfasst werden (siehe Abbildung 2.4). Würde man einen festen Zeitraum bestimmen, in welchem die Werte stets zusammengefasst werden, so würden unter Umständen viele Informationen verloren gehen. Auf der anderen Seite kann es vorkommen, dass ein Score in einem Zeitraum gar nicht oder sehr selten erfasst wurde. Die Gefahr, dass sich die Erhebung eines Score dann auf einen ganz anderen Zeitpunkt als ein im gleichen Zeitraum geschriebener Text bezieht, wäre damit sehr groß. Die Korrelation zwischen Inhalt des Visitentexts und des Scores wäre womöglich sehr gering, und die entstehenden Wertepaare dementsprechend ungeeignet, ein effektives Modell zu trainieren.

Zur Generierung der Wertepaare habe ich mich demnach für eine andere Methodik entschieden. Die Implementierung liegt hierfür in `skripte/find_training_pairs.py` vor. Jeder Erhebung eines Scores wird derjenige Visitentext zugeordnet, der zeitlich betrachtet am nächsten zu dem erhobenen Score eingetragen wurde. Dabei kann der Text sowohl vor als auch nach der Eintragung des Scores verfasst worden sein. Weiterhin werden `Visite_Pflege`, `Visite_ZNS` und `Visite_Oberarzt` unabhängig voneinander betrachtet, sodass pro Eintragung eines Scores jeweils eine Zuordnung zu jedem der drei Texte erfolgt. In Abbildung 2.4 sind diese Zuordnungen durch Pfeile dargestellt. Beim späteren Training der Modelle werden dann nur solche Trainingspaare betrachtet, die den gewünschten Eingabetext beinhalten. Außerdem kann es auch mit dieser Methode vorkommen, dass in einigen Ausnahmefällen die Eintragungen von Score bzw. Text mit einem großen

zeitlichen Abstand erfolgt sind. Deswegen wird für jedes Wertepaar zusätzlich die zeitliche Differenz der beiden Eintragungen gespeichert. Das ermöglicht, dass vor dem Trainining eines Modells nur solche Wertepaare selektiert werden, deren zeitlicher Abstand unter einem festgelegten Maximalwert liegt. Da zunächst alle erdenklichen Wertepaare vorliegen kann dieser maximale Abstand je nach Bedarf besonders groß oder klein gewählt werden, beispielsweise dann, wenn ein Modell eine besonders hohe Anzahl an Trainingspaaren benötigt. Abbildung 2.3 zeigt die Anzahl der Wertepaare pro Kategorie von Eingabetext je nach maximal erlaubter Zeit zwischen den beiden Eintragungen.

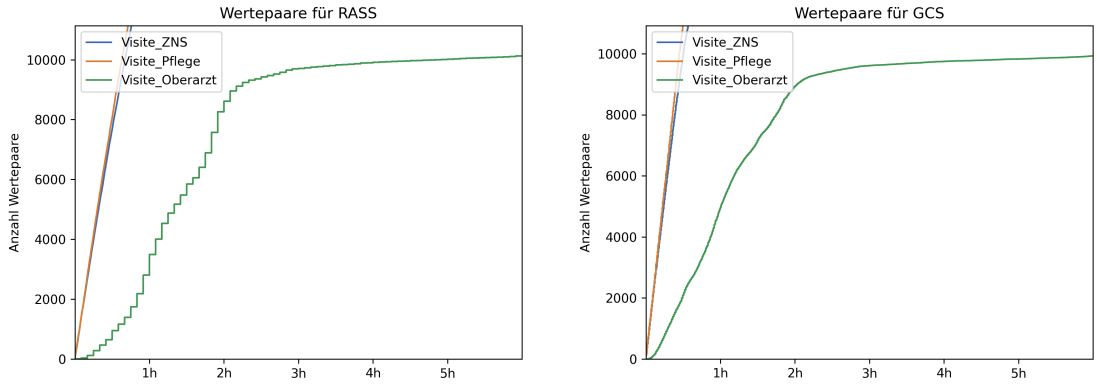


Abbildung 2.3: Anzahl Wertepaare bei maximal zulässiger Differenz zwischen Zeitpunkten

2.3 Genauigkeit der erfassten Daten

Neben einer möglichst genauen, automatischen Vergabe von medizinischen Scores anhand von Visitentexten stellt sich weiterhin die Frage, welche der gegebenen Eingabewerte überhaupt der Wahrheit entsprechen, beziehungsweise diese am genauesten abbilden. Die Modelle, die in Abschnitt 3 beschrieben werden, nehmen die eingetragenen Scores als „absolute Wahrheit“[5] an. Wir gehen also davon aus, dass diese Eintragungen die Kondition eines Patienten im Zweifelsfall genauer beschreiben als die Visitentexte, die zu einem ähnlichen Zeitpunkt verfasst wurden. Dies stellt allerdings eine Vereinfachung der Situation dar. Im Allgemeinen ist es fast unmöglich, nachträglich zu bestimmen, welcher Wert die Realität widerspiegelt, wenn Score und Text voneinander abweichen.

[5]Im maschinellen Lernen und in verwandten Bereichen auch als „ground truth“ bezeichnet

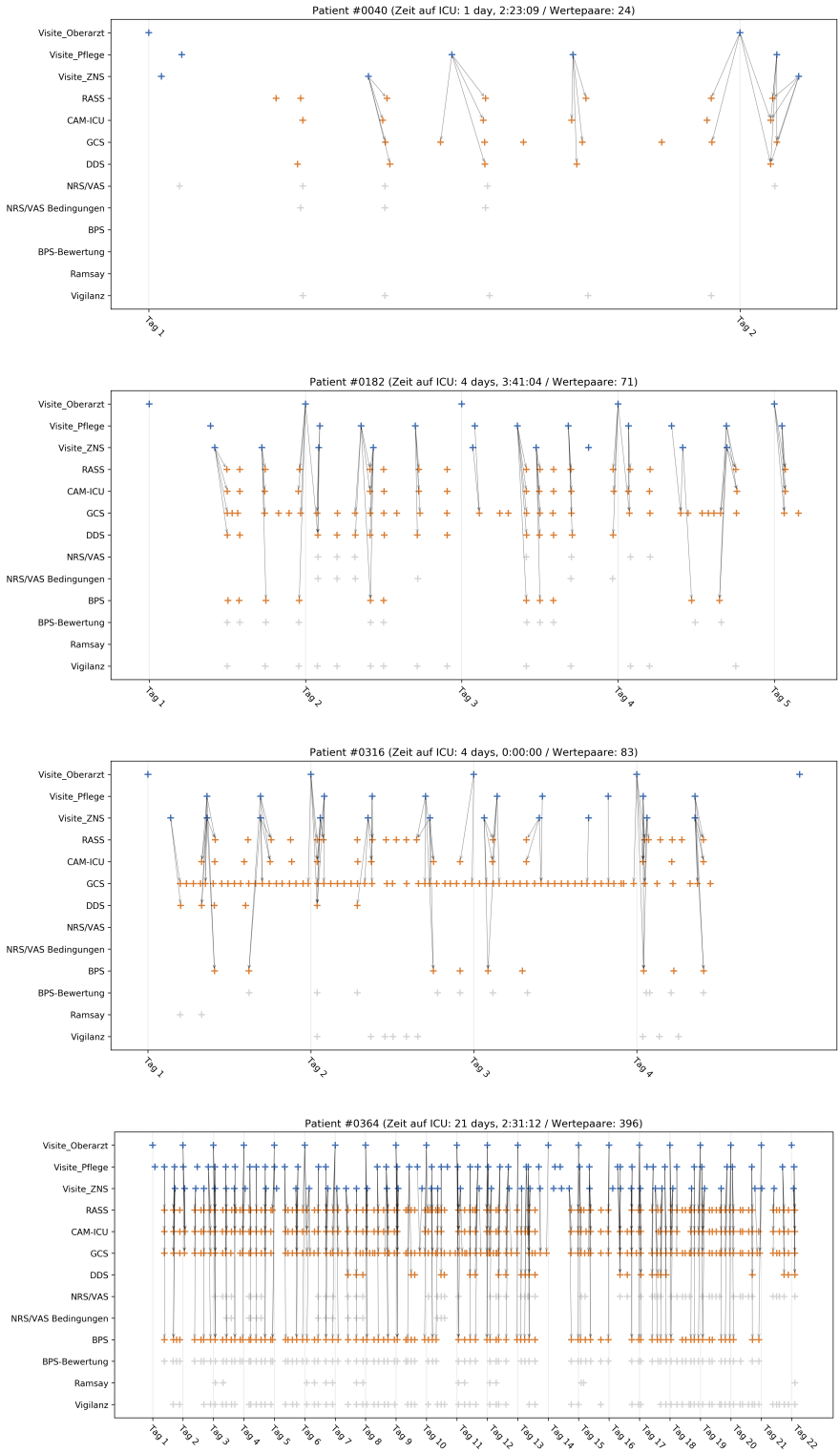


Abbildung 2.4: Übersicht der erfassten Werte einiger Patienten

Diese Art von Widerspruch tritt in den betrachteten Datensätzen häufig auf und kann mehrere Gründe haben. Zum einen kann es vorkommen, dass Ärzte oder Pflegekräfte aufgrund von Zeitmangel oder Stress einen Text nicht mit der nötigen Gewissenhaftigkeit oder im nötigen Umfang verfassen, um aus den gegebenen Informationen einen Score herzuleiten. Im Extremfall kann es sogar vorkommen, dass Texte oder Scores aus der vorhergehenden Erhebung unverändert übernommen werden, auch wenn sich der Zustand des betreffenden Patienten verändert hat. Wird aber gleichzeitig ein entsprechender Score bzw. Text aktualisiert kommt es zu einem Widerspruch. Ferner ist es selbst ohne menschlichen Fehler möglich, dass Score und Visitentext unterschiedliche Werte indizieren, auch wenn diese zeitlich nah beieinander liegen. Da sich insbesondere auf einer Intensivstation der Zustand eines Patienten sehr schnell ändern kann, kann es durchaus vorkommen, dass sich zwei zeitlich sehr nah beieinanderliegende Eintragungen inhaltlich stark unterscheiden, obwohl beide Werte zum Zeitpunkt ihrer Eintragung als realitätsgetreu angesehen werden können.

Diese Gründe sind nicht weiter Beobachtungsgegenstand der vorliegenden Arbeit. Dennoch muss diese Art von möglichen Unstimmigkeiten in den vorliegenden Daten bei der Konzeption, Entwicklung und Bewertung der Modelle beachtet werden. Insbesondere stellt die Qualität der Eingabedaten, ohne weitere Maßnahmen zur Datenbereinigung, womöglich eine obere Schranke für die Performance der Modelle dar.

3 Baseline-Modell

3.1 Datenaufbereitung

Die Verarbeitung natürlicher Sprache stellt eine besondere Herausforderung für das maschinelle Lernen dar. Im Allgemeinen sind Texte, also Aneinanderreihungen von Buchstaben und Symbolen variabler Länge, keine geeigneten Eingabedaten für die Modelle des überwachten maschinellen Lernens. Daher müssen sie zuerst in eine angemessene numerische Repräsentation überführt werden. Dabei ist es wichtig dass der für die Vorhersage relevante Inhalt des Textes so gut wie möglich erhalten bleibt. Dieser Prozess, bei dem jeweils ein Eingabetext auf einen abstrakten Vektor reeller Zahlen abgebildet wird (den sog. *feature vector*), wird als *word embedding* bezeichnet.

3.1.1 Tokenisierung

Bag-of-Words

Ein trivialer Ansatz ist das sogenannte Bag-of-Words-Modell. Hierbei wird zunächst jeder Eingabetext in eine Liste seiner *tokens* umgewandelt (Tokenisierung). Im einfachsten Fall ist jedes Wort, getrennt durch eines oder mehrere Leerzeichen, ein solches Token:

Abbildung 3.1

Anschließend wird jedem Wort, das in einem oder mehreren der Eingabetexte auftritt, eine feste Position in den feature-Vektoren zugeordnet. Die Länge der Vektoren entspricht somit der Anzahl einzigartiger Worte in allen Eingabetexten. Zuletzt werden die Vorkommnisse der Worte in jedem Text gezählt und ihre Summe an der entsprechenden Stelle des dazugehörigen Vektors eingetragen:

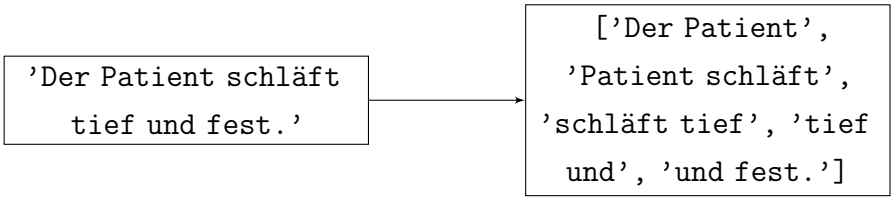
	Der	Patient	schläft	tief	und	fest	ist	nicht	agitiert	sediert
Der Patient schläft tief und fest	1	1	1	1	1	1	0	0	0	0
Patient ist nicht agitiert und schläft	0	1	1	0	1	0	1	1	1	0
Der Patient ist tief sediert	1	1	0	1	0	0	1	0	0	1

Tabelle 3.1: Absolute Häufigkeit von Worten in drei Beispieltexten

Die Eingabedaten bestehen somit aus einer Matrix, deren Zeilen die Eingabetexte und deren Spalten die Anzahl der Vorkommnisse eines bestimmten Wortes enthalten.

N-Gramme

Ein Nachteil dieser Art der Repräsentation ist, dass die Information über die Reihenfolge der Wörter innerhalb eines Textes verloren geht: Zwei Texte mit den gleichen Worten in einer unterschiedlichen Reihenfolge werden somit auf den gleichen Vektor abgebildet. Dieses Problem wird durch die Einführung sogenannter Bigramme vermindert: Hierbei werden jeweils zwei aufeinanderfolgende Worte zusammen als ein Token aufgefasst:



Das Bigramm ist eine spezielle Form des allgemeinen N-Gramms, bei dem n die Anzahl der aufeinanderfolgenden Wörter angibt, die zu einem Token zusammengesetzt werden. Die Verwendung von N-Grammen bei der Tokenisierung ermöglicht es somit, einige der semantischen Informationen des Ausgangstextes zu erhalten, die bei einem einfachen Bag-of-Words-Modell verloren gehen würden. Gleichzeitig wird die Dimensionalität der Eingabevektoren drastisch erhöht, was zu höheren Anforderungen an Rechenkapazität und Speicherplatz beim Trainieren des Modells führt.

Statt auf Wortebene lässt sich die Tokenisierung der Eingabetexte auch auf Zeichenebene durchführen. Die Repräsentation des Textes wird somit deutlich granularer und höherdimensionierter, indem bei einem N-Gramm statt n Worten n aufeinanderfolgende

Zeichen (d.h. Buchstaben oder Zahlen) zu einem Token zusammengesetzt werden. Bei einer Sonderform dieser sogenannten *character n-grams* werden nur solche Zeichenfolgen von Länge n betrachtet, die Teil eines einzelnen Wortes sind und nicht über Wortgrenzen hinausgehen. Die Auswirkungen der Wahl von n sowie des Verfahrens zur Tokenisierung werden im Abschnitt GridSearchCV genauer erläutert.

Tf-idf-Maß

Einige Worte (z.B. „Patient“, „Verlauf“) treten in vielen Texten auf und haben dadurch eine geringere Bedeutung bei der Vorhersage eines Score-Werts. Seltenere Worte hingegen, die eine hohe Bedeutung haben können, müssen demnach entsprechend gewichtet werden, um diesen bei der Verarbeitung durch das Modell einen höheren Einfluss auf die Ausgabe zu ermöglichen.

Zu diesem Zweck wird das sogenannte Tf-idf-Maß ermittelt, welches dem Produkt der *term frequency* und der *inverse document frequency* entspricht. Der Begriff *term frequency* bezeichnet hierbei lediglich die bereits berechnete absolute Häufigkeit eines Wortes (bzw. Tokens im Allgemeinen), d.h. die Summe seiner Vorkommnisse innerhalb eines Texts. Die *inverse document frequency* gibt die Aussagekraft eines bestimmten Wortes an, indem seine relative Häufigkeit in allen Texten bestimmt wird. Sei D die Menge aller Texte und t ein Token, so ist $\mathrm{idf}(t, D)$ der logarithmisch skalierte Kehrwert des Anteils derjenigen Texte aus D, in denen t mindestens ein mal vorkommt:

$$\mathrm{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

Hierbei ist zu beachten, dass $\mathrm{idf}(t, D)$ nur für solche t definiert ist, die in mindestens einem Text in D auftreten, da sonst $\{d \in D : t \in d\} = \emptyset$ mit $|\emptyset| = 0$.

	Der	Patient	schläft	tief	und	fest	ist	nicht	agitiert	sediert
Der Patient schläft tief und fest	.18	0	.18	.18	.18	.48	0	0	0	0
Patient ist nicht agitiert und schläft	0	0	.18	0	.18	0	.18	.48	.48	0
Der Patient ist tief sediert	.18	0	0	.18	0	0	.18	0	0	.48

Tabelle 3.2: Tf-idf-Maß von Worten in den gleichen Beispieltexten

In Tabelle 3.2 ist erkennbar, dass Worte, die in weniger Texten auftreten, einen höheren Tf-idf-Wert haben. „Patient“ tritt in allen betrachteten Texten auf und hat somit einen Wert von 0, da $\text{idf}('\text{Patient}', D) = \log \frac{3}{3} = 0$.

In der verwendeten Implementierung von scikit-learn werden die n-dimensionalen Tf-idf-Vektoren der Eingabetexte anschließend normiert, indem sie durch ihre euklidische Norm dividiert werden:

$$v_{norm} = \frac{v}{||v||_2} = \frac{v}{\sqrt{\sum_{i=0}^{n}(v_i)^2}}$$

3.1.2 weitere Datenbereinigung

Abbildung 3.1 zeigt exemplarisch die Überführung eines Beispieltexts in seine Tokens und verdeutlicht ein weiteres häufiges Problem der Textverarbeitung: Die Eingabedaten enthalten Satz- und Sonderzeichen, die eine sinnvolle Tokenisierung weiter erschweren. Der Beispieltext endet, wie viele der realen Eingabedaten, mit einem Punkt. Somit wird „fest.“ als neues Token erkannt, welches aus Sicht des Modells komplett unabhängig zu dem womöglich ebenfalls auftretenden „fest“ ist. Dies erhöht nicht nur die Dimensionalität der Eingabedaten unnötig, sondern erschwert es dem Modell auch, die Bedeutung betroffener Worte bei der Vorhersage der medizinischen Scores zu ermitteln[1].

Bevor ein Eingabetext für die Verarbeitung durch das Baseline-Modell tokenisiert wird, wird er in vier Schritten sukzessive vereinfacht und auf seine semantischen Kerninhalte reduziert (siehe Abbildung 3.2). Ziel ist es, statistisches Rauschen zu minimieren, sodass die Texte eine maximal hohe Aussagekraft bei der Bestimmung der medizinischen Scores haben.

Im ersten Schritt wird jedes großgeschriebene Satzzeichen durch den entsprechenden Kleinbuchstaben ersetzt. Somit spielt die Großschreibung bei der Unterscheidung der Tokens keine Rolle mehr. Danach werden sämtliche Sonderzeichen, d.h. solche, die nicht Teil des deutschen Alphabets und keine Zahl sind, aus dem Text entfernt, da diese ebenfalls keine Bedeutung bei der Bestimmung des passenden Score-Werts haben.

In Schritt drei wird jedes verbleibende Wort auf seinen Wortstamm zurückgeführt, da

[1] Die Abkürzung „pat“ für Patient tritt in den bereitgestellten Eingabedaten ebenfalls häufig (n = 37.103) und mit unterschiedlicher Großschreibung auf, wird aber in nur etwa 66.8 % (n = 24.810) der Fälle mit einem Punkt („pat.“) geschrieben. Ohne weitere Schritte zur Datenbereinigung würden diese beiden Worte komplett separat behandelt werden, obwohl sie semantisch identisch sind.

die verschiedenen syntaktischen Formen eines Wortes ebenfalls unerheblich sind. Hierbei findet eine Python-Implementierung des Stemming-Algorithmus[2] von Dr. Martin Porter Anwendung. Dieser entfernt Suffixe deutscher Worte anhand einer Folge wohldefinierter Regeln und benötigt somit kein Wörterbuch, um die Worte auf ihre Wortstämme zu reduzieren.

Schlussendlich werden sogenannte Stoppwörter entfernt. Dies sind Wörter, die in der deutschen Sprache häufig vorkommen und somit nur eine syntaktische Bedeutung haben, bei der Deutung des Inhalts eines Textes aber unerheblich sind. Grundlage hierfür bildet eine Liste von 232 deutschen Stoppwörtern aus dem Korpus des Open-Source-Projekts NLTK[3]. Da die Stammformreduktion bereits im vorherigen Schritt stattfand, muss die Liste der Stoppwörter entsprechend angepasst werden, um im Text die Worte korrekt zu entfernen. Insgesamt zwölf Worte wie „nicht“, „ohne“ und „keine“ wurden manuell aus der NLTK-Liste entfernt, da diese eine inhaltlich relevante Bedeutung in den Texten haben können.

3.2 Support Vector Machine

Die sogenannte *Support Vector Machine* ist ein Modell des überwachten Lernens, das einen hohen Grad an Generalisierung ermöglicht, d.h. auch bei neuen Daten, die nicht zum Trainieren des Modells genutzt wurden, eine hohe Genauigkeit erbringen [Awad und Khanna, 2015]. Während das Konzept der SVM ursprünglich zur Lösung von Klassifizierungsproblemen erdacht wurde, bezeichnet *Support Vector Regression* eine Modifikation des gleichen Konzepts zur Regression. Wie bei anderen Regressionsmodellen wird versucht, im n-dimensionalen Raum der Eingabedaten eine Hyperebene zu finden, die den Zusammenhang zwischen Ein- und Ausgabedaten möglichst genau modelliert und somit Vorhersagen über Eingaben (Visitentexte), deren dazugehörigen Ausgaben (Score-Werte) noch unbekannt sind, ermöglicht. Haben die Eingabedaten nur eine Dimension, entspricht dies anschaulich einer Regressionsgeraden, die durch die Punktwolke $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ der n Wertepaare gelegt wird. Sind die Eingabedaten zweidimensionale Vektoren, so entspricht die Ausgabe des Modells einer Ebene im 3-dimensionalen Raum. Die in Abschnitt 3.1 beschriebene Überführung der Eingabetexte

[2] http://snowball.tartarus.org/algorithms/german/stemmer.html
[3] Natural Language Toolkit: https://www.nltk.org/

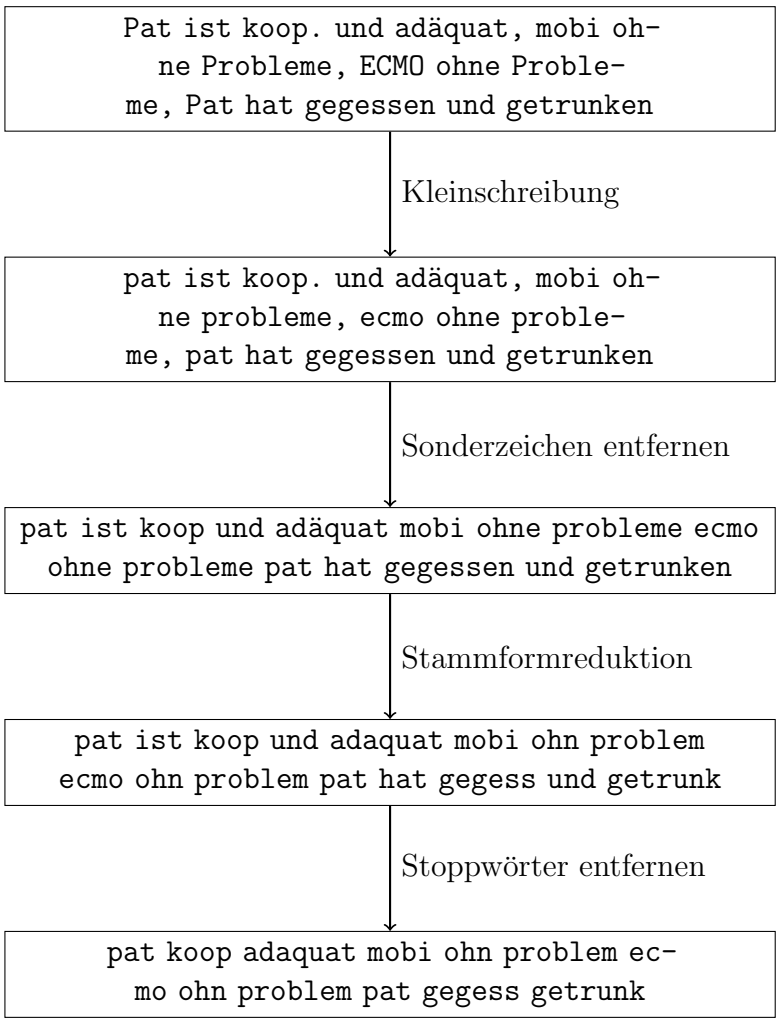


Abbildung 3.2

in reelle Vektoren liefert deutlich höherdimensionierte Daten, die weniger anschaulich sind, bei denen das Konzept aber das gleiche bleibt.

Eine solche Hyperebene (bzw. Gerade im 2-dimensionalen Raum) kann nur gute Ergebnisse liefern, wenn eine lineare Abhängigkeit zwischen Ein- und Ausgabedaten besteht. Da dies im Allgemeinen nicht der Fall ist, wird bei der SVM der Kernel-Trick angewendet. Hierbei werden die Daten in einen noch höher dimensionierten Raum überführt, bei dem eine lineare Trennbarkeit gegeben ist.

Reale Eingabedaten sind häufig ungenau und fehlerbelastet (siehe Abschnitt 2.3). Anhand dieser Daten ein Modell zu entwickeln, welches jedem möglichen neuen Visitentext den korrekten Score-Wert zuordnet ist somit unmöglich. Eine weitere Besonderheit der

SVM liegt bei der Berechnung der Verlustfunktion in der Einführung der Variable ϵ. Diese gibt eine maximal erlaubte Toleranz bei der Vorhersage der Ausgabedaten an. Liegt die Vorhersage des Modells weniger als ϵ von dem tatsächlichen Wert entfernt, so beträgt der Wert der Verlustfunktion für diesen Eingabewert 0. Die Einführung einer solchen Toleranz ermöglicht es, ein Modell zu trainieren, das weniger anfällig gegenüber widersprüchlichen Eingabedaten ist und somit eine bessere Generalisierung im Allgemeinen ermöglicht [Awad und Khanna, 2015].

3.2.1 Suche nach den besten Parametern

Für die Implementierung der *Support Vector Machine* wurde das Open-Source-Framework `scikit-learn` 0.24 [Pedregosa et al., 2011] verwendet. Neben den mathematischen Parametern, die sich iterativ durch den Lernprozess geeigneten Werten annähern, existieren eine Reihe an sogenannten *Hyperparametern*, die von dem Entwickler manuell festgelegt werden. Darunter fallen beispielsweise die Wahl von ϵ und C[4], sowie diverse Parameter bei der Vorverarbeitung der Eingabetexte. Das `scikit-learn`-Framework stellt für die Suche nach geeigneten Hyperparametern die Funktion `GridSearchCV` bereit. Diese durchläuft automatisch den gesamten Hyperparameter-Raum, indem automatisch für jede Kombination ein Modell trainiert und die Ergebnisse sortiert und gespeichert werden.

Kreuzvalidierung Zur Messung der Vorhersagegenauigkeit der trainierten Modelle kam die 5-fache Kreuzvalidierung zum Einsatz. Die Eingabedaten werden hierbei in fünf gleich große, disjunkte Teilmengen unterteilt. Das Modell wird zunächst auf eine Trainingsmenge, bestehend aus vier der fünf Teilmengen, trainiert. Anhand der noch ungesehenen Datenpaare aus der fünften Menge wird das Modell getestet. Als Verlustfunktion kommt *mean absolute error* zum Einsatz, d.h. die durchschnittliche Abweichung der Vorhersage zu dem tatsächlichen Wert. Anschließend wird der Prozess weitere vier Male wiederholt, wobei beim Training des Modells jedes mal eine andere Menge zu Testzwecken ausgelassen wird. Der Durchschnittswert der fünf Durchläufe wird zuletzt als Wert für die gewählte Hyperparameter-Kombination gespeichert.

[4]Bezeichnung für den sogenannten Regularisierungs-Parameter. Ein hoher Grad an Regularisierung führt dazu, ein einfaches Modell zu bevorzugen, das weniger von den konkreten Eingabedaten abhängt und overfitting verhindert.

Die folgenden Parameterkombinationen wurden untersucht:

Stufe	Parameter	mögliche Werte
word embedding	preprocessor	lower, lower+clean, lower+clean+stem, lower+clean+stem+rmstop
word embedding	analyzer	word, char, char_wb
word embedding	ngram range	$\{(n, n+m), n \in \{1,2,4,6,8\}, m \in 0,2,4,6,8,10\}$
SVR	kernel	linear, poly, rbf, sigmoid
SVR	ϵ	0.05, 0.1, 0.15
SVR	C	$\{10^{-n} \mid n \in \mathbb{Z} \cap [-1,3]\} \cup \{5,15\}$

Die höchste Performance wurde durch einen zeichenbasierten Tokenizer mit einer ngram range von 2 – 12, der auch über Wortgrenzen hinaus geht, erreicht. Es fließen also alle Zeichenketten mit einer Länge zwischen 2 und 12 Zeichen als Eingabevektoren in das Modell ein. Die Wahl von *epsilon* sowie das Filtern von Stoppwörtern hatte keinen statistisch signifikanten Einfluss auf das Ergebnis. Für den Regularisierungsparameter C erreichte ein Wert von 1 das beste Ergebnis (siehe Abbildung 3.3).

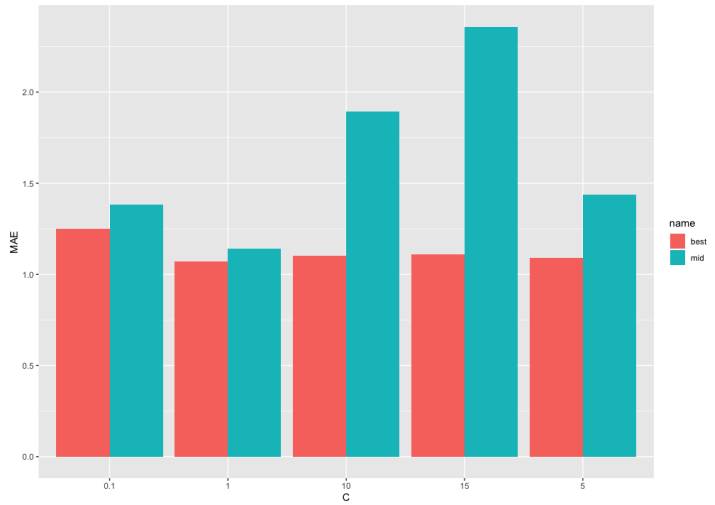


Abbildung 3.3: Geringste und durchschnittliche Abweichung nach Wahl des Regularisierungsparameters C

3.2.2 Ergebnisse

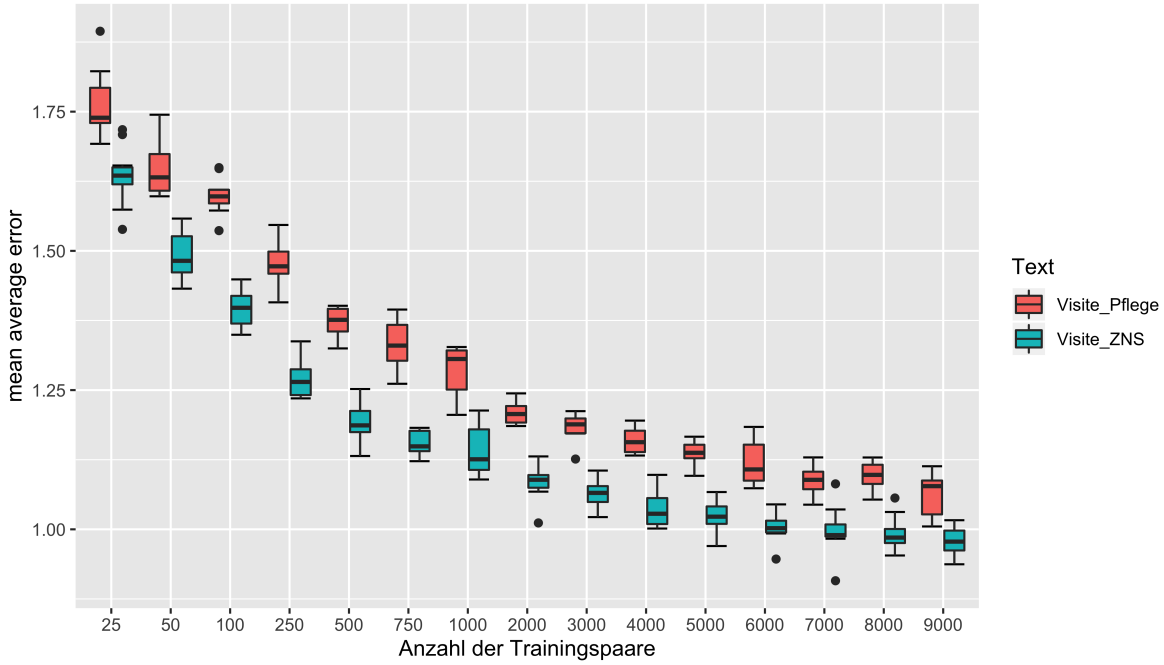
Nach der Ermittlung der besten Hyperparameter wurde für die Visitentexte `Visite_ZNS` und `Visite_Pflege` jeweils ein Modell zur Vorhersage der Scores RASS und GCS entwickelt. Jedes dieser vier Modelle wurde mit einer sukzessive steigenden Anzahl an Trainingspaaren trainiert, um den Zusammenhang zwischen der Anzahl der Eingabedaten sowie der Leistung des Modells bei noch unbekannten Daten zu ermitteln (siehe Abbildung 3.4). Für jeden der Trainingsvorgänge wurde eine Stichprobe der Größe n aus der Menge der entsprechenden Text-Wert-Paare betrachtet, deren Abstand zueinander 45 Minuten oder weniger betrug (siehe Abschnitt 2.2).

Die Wahl von 45 Minuten als maximaler Abstand zwischen den Eintragungen ermöglichte für alle vier Kombinationen, bis zu 9000 Paare zum Trainieren des Modells zu finden, deren zeitliche Nähe einen starken Zusammenhang zwischen Ein- und Ausgabewert ermöglicht. Um eine gute Vergleichbarkeit zwischen den verschiedenen Modellen zu ermöglichen wurde für alle Stichproben die Menge der Paare mit Abstand $\leq$ 45 Minuten betrachtet.

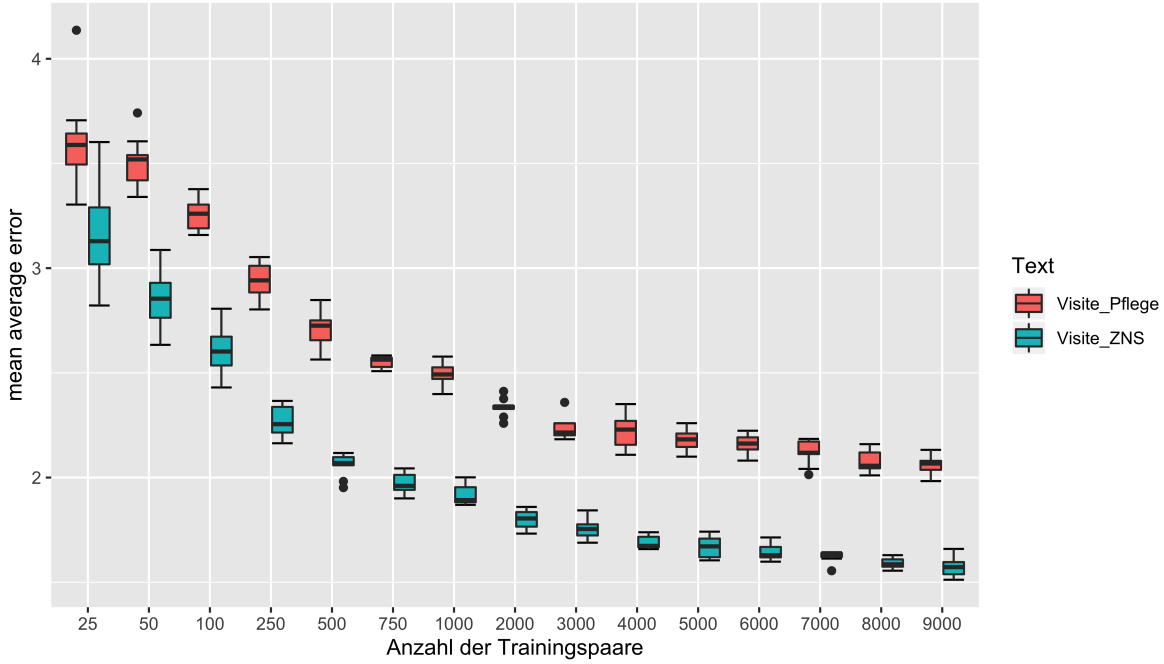
Unabhängig von der Anzahl der Trainingspaare wurde anschließend jedes Modell auf einen Testdatensatz der Größe n=1000 getestet. Bei der Wahl der Testdaten wurde darauf geachtet, nur solche Paare zu verwenden, die noch nicht beim Trainieren des Modells verwendet wurden. Eine Überschneidung hätte einen unfairen Vorteil für das Modell zur Folge und würde seine Leistung bei der Vorhersage von unbekannten Daten nicht korrekt wiederspiegeln.

Als Maßstab zur Bewertung der Modelle wurde der *mean absolute error* berechnet, d.h. die mittlere Abweichung der Vorhersagen des Modells zu den realen Daten, da dieses Maß eine intuitive Einschätzung der Qualität der Ausgaben ermöglicht. Der Trainings- und Testprozess wurde für jedes Modell und jede Stichprobengröße zehn mal wiederholt, um Ungleichmäßigkeiten in der Qualität der Trainingsdaten entgegenzuwirken.

In Abbildung 3.4 wird deutlich, dass unabhängig von der Art der Ein- und Ausgabedaten die Vorhersagegenauigkeit der Modelle steigt, je mehr Wertepaare zum Trainieren genutzt werden. Dieser Effekt ist bei einer geringen Anzahl an Trainingspaaren am stärksten ausgeprägt und lässt mit steigender Anzahl der Paare nach. Während sich die Abweichung der Vorhersagen einer unteren Schranke anzunähern scheint, steigt die benötigte Zeit für das Trainieren des Modells jedoch exponentiell (siehe Abbildung 3.5).



(a) Vorhersage von RASS



(b) Vorhersage von GCS

Abbildung 3.4

In der Praxis gilt es also, eine Abwägung zwischen der verfügbaren Rechenleistung und benötigten Vorhersagegenauigkeit des Modells zu treffen.

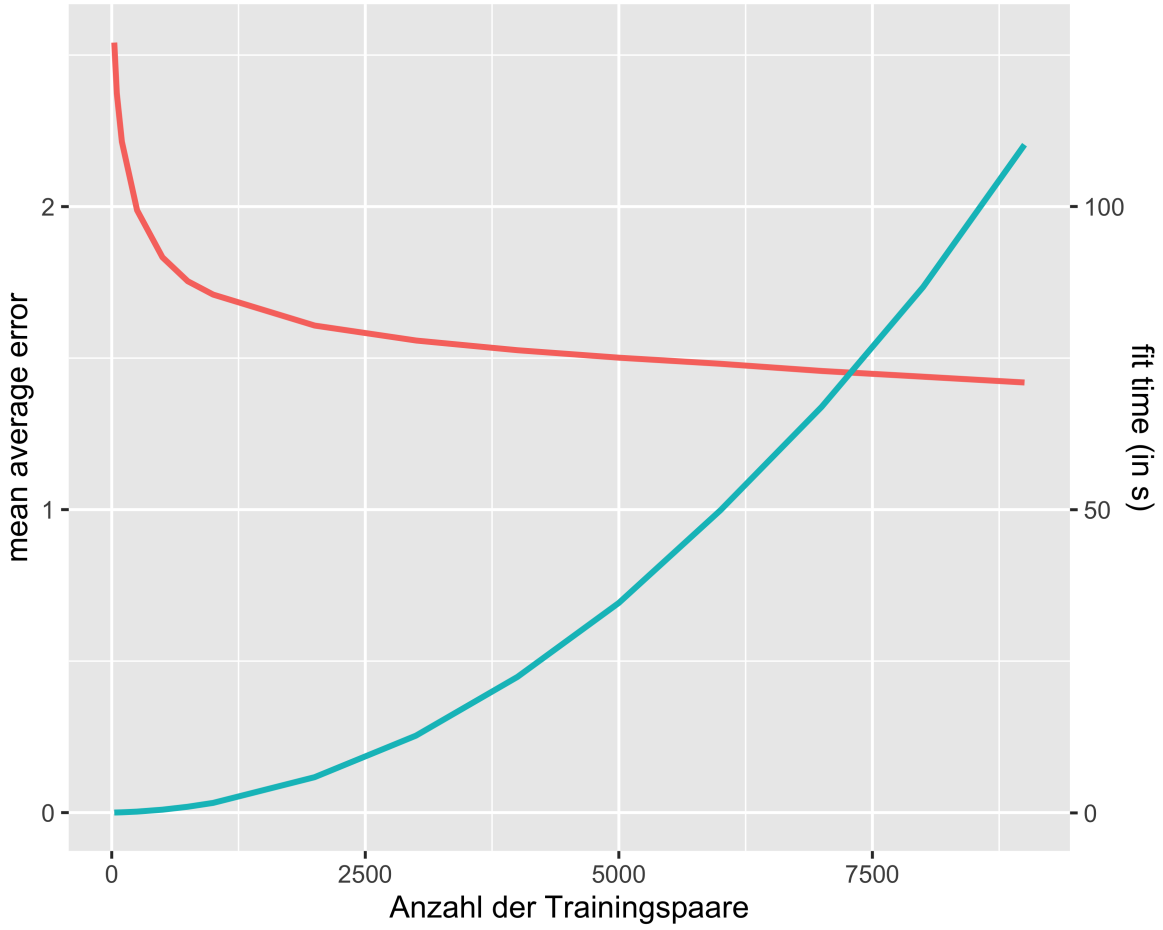


Abbildung 3.5: Mittlere Abweichung bei der Vorhersage von GCS und RASS in Zusammehang mit der benötigten Zeit zum Trainieren des Modells.

Verteilung der ausgegebenen Werte

Filtern von RASS-Vorkommnissen

35 % ($n = 8426$) der bereitgestellten Texte aus Kategorie `Visite_ZNS` enthalten die Zeichenkette `'RASS=n'` oder eine Variation davon. Dabei ist zu beachten, dass sich selbst diese Angaben häufig von Eintragungen des RASS-Wertes in enger zeitlichen Nähe unterscheiden (siehe Abschnitt 2.3).

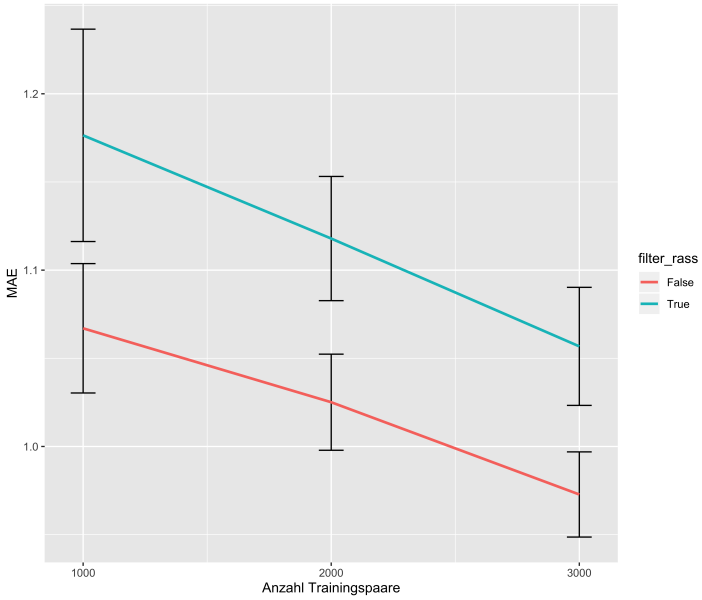


Abbildung 3.6

Um zu überprüfen, in welchem Umfang dieser Inhalt von dem Modell erkannt wird und in die Vorhersage des RASS-Wertes einfließt, wurden erneut je fünf Modelle mit 1000, 2000 und 3000 Trainingspaaren der Kombination `Visite_ZNS:RASS` entwickelt. Es wurden nur solche Trainingspaare verwendet, die die o.g. Zeichenkette enthalten. Für jede Stichprobe wurde zunächst ein reguläres Modell trainiert, sowie danach eine Variation, bei der Inhalte der Form `'RASS=n'` vor der Tokenisierung hinausgefiltert wurden.

Abbildung 3.6 zeigt, dass die Modelle, bei denen die Angabe eines expliziten RASS-Wertes im Voraus entfernt wurde, deutlich schlechter abschnitten als diejenige, denen diese Information zur Verfügung stand. Dies ist ein Indiz für die Fähigkeit des Modells, ohne explizites Wissen über die Bedeutung des Textes wichtige Informationen zu identifizieren und in die Berechnung des endgültigen Score-Wertes einfließen zu lassen.

4 Fazit

Im Rahmen der vorliegenden Arbeit wurde die Konzeption, Entwicklung und Bewertung eines Machine Learning-Modells zur Vorhersage medizinischer Score-Werte auf Basis von Visitentexten behandelt. Die Anwendung der Support Vector Regression wurde am Beispiel der Scores RASS und GCS sowie der Eingabetexte `Visite_ZNS` und `Visite_Pflege` vorgestellt. Das gleiche Konzept lässt sich mit nur kleinen Anpassungen auf andere Scores oder Eingabetexte übertragen.

Unter Verwendung der besten Hyperparameter (siehe Abschnitt 3.2.1) wurden folgende Ergebnisse bei der Vorhersage der Score-Werte ermittelt:

Eingabetext	Score	MAE
`Visite_ZNS`	RASS	1
`Visite_Pflege`	GCS	1
`Visite_ZNS`	RASS	1
`Visite_Pflege`	GCS	1

Tabelle 4.1: mittlere absolute Abweichung der Vorhersagen

Literaturverzeichnis

S. S. Arun und G. Neelakanta Iyer. On the analysis of COVID19 - novel corona viral disease pandemic spread data using machine learning techniques. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1222–1227. 2020.

Mariette Awad und Rahul Khanna. Support vector regression. In *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*, pages 67–80. Apress, Berkeley, CA, 2015. ISBN 978-1-4302-5990-9. doi:10.1007/978-1-4302-5990-9\%0084.

Po-Hsuan Cameron Chen, Yun Liu, und Lily Peng. How to develop machine learning models for healthcare. *Nature Materials*, 18(5):410–414, 2019. ISSN 1476-1122, 1476-4660. doi:10.1038/s41563-019-0345-0.

Andres Colubri, Mary-Anne Hartley, Mathew Siakor, Vanessa Wolfman, Tom Sesay, August Felix, Adam C. Levine, und Pardis C. Sabeti. Machine-Learning prognostic models from the 2014-16 Ebola outbreak: Data-harmonization challenges, validation strategies, and mHealth applications. *bioRxiv*, page 294587, 2019. doi:10.1101/294587.

Gokul S. Krishnan und S. Sowmya Kamath. A Supervised Learning Approach for ICU Mortality Prediction Based on Unstructured Electrocardiogram Text Reports. In **Max Silberztein, Faten Atigui, Elena Kornyshova, Elisabeth Métais, und Farid Meziane**, editors, *Natural Language Processing and Information Systems*, pages 126–134. Springer International Publishing, Cham, 2018. ISBN 978-3-319-91947-8.

Gernot Marx, Elke Muhl, Kai Zacharowski, und Stefan Zeuzem, editors. *Die Intensivmedizin*. Springer Medizin, Berlin Heidelberg, 12., vollständig überarbeitete, akualisierte und erweiterte auflage edition, 2015. ISBN 978-3-642-54952-6 978-3-642-54953-3.

Tom M. Mitchell. *Machine Learning*. McGraw-Hill Series in Computer Science.

McGraw-Hill, New York, NY, international ed., [reprint.] edition, 1997. ISBN 978-0-07-115467-3.

Anika Müller, Björn Weiß, Claudia Spies, und S3-Leitliniengruppe. Analgesie, Sedierung und Delirmanagement – Die neue S3-Leitlinie ,,Analgesie, Sedierung und Delirmanagement in der Intensivmedizin“ (DAS-Leitlinie 2015). *AINS - Anästhesiologie · Intensivmedizin · Notfallmedizin · Schmerztherapie*, 50(11/12):698–703, 2015. ISSN 0939-2661, 1439-1074. doi:10.1055/s-0041-107321.

Nils J Nilsson. Learning machines. 1965.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, und Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

Todd R. Reed, Nancy E. Reed, und Peter Fritzson. Heart sound analysis for symptom detection and computer-aided diagnosis. *Simulation Modelling Practice and Theory*, 12(2):129–146, 2004. ISSN 1569190X. doi:10.1016/j.simpat.2003.11.005.

Stuart J. Russell und Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Series in Artificial Intelligence. Pearson, Hoboken, fourth edition edition, 2020. ISBN 978-0-13-461099-3.

A. L. Samuel. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959. ISSN 0018-8646, 0018-8646. doi:10.1147/rd.33.0210.

Md. Shahriare Satu, Koushik Chandra Howlader, und Sheikh Mohammed Shariful Islam. Machine Learning-Based Approaches for Forecasting COVID-19 Cases in Bangladesh. *SSRN Electronic Journal*, 2020. ISSN 1556-5068. doi: 10.2139/ssrn.3614675.

Curtis N. Sessler, Mark S. Gosnell, Mary Jo Grap, Gretchen M. Brophy, Pam V. O'Neal, Kimberly A. Keane, Eljim P. Tesoro, und R. K. Elswick.

The Richmond Agitation–Sedation Scale: Validity and Reliability in Adult Intensive Care Unit Patients. *American Journal of Respiratory and Critical Care Medicine*, 166(10):1338–1344, 2002. ISSN 1073-449X, 1535-4970. doi:10.1164/rccm.2107138.

Pratik Shah, Francis Kendall, Sean Khozin, Ryan Goosen, Jianying Hu, Jason Laramie, Michael Ringel, und Nicholas Schork. Artificial intelligence and machine learning in clinical development: A translational perspective. *npj Digital Medicine*, 2(1):69, 2019. ISSN 2398-6352. doi:10.1038/s41746-019-0148-3.

Duncan Shillan, Jonathan A. C. Sterne, Alan Champneys, und Ben Gibbison. Use of machine learning to analyse routinely collected intensive care unit data: A systematic review. *Critical Care*, 23(1):284, 2019. ISSN 1364-8535. doi:10.1186/s13054-019-2564-9.

Graham Teasdale und Bryan Jennett. Assessment of Coma and Impaired Consciousness. A Practical Scale. *The Lancet*, 304(7872):81–84, 1974. ISSN 01406736. doi:10.1016/S0140-6736(74)91639-0.

Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, und Shanrong Zhao. Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery*, 18(6):463–477, 2019. ISSN 1474-1784. doi:10.1038/s41573-019-0024-5.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den 24. Juli 2020 .....