Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Identifikation von relevanten Metriken zur Analyse von Kommunikation in Entwicklerteams

Identifying relevant metrics to analyze communication in developement teams

Masterarbeit

im Studiengang Informatik

von

Alexander Specht

Prüfer: Prof. Dr. rer. nat. Kurt Schneider
Zweitprüfer: Dr. rer. nat. Jil Ann-Christin Klünder
Betreuer: Dr. rer. nat. Jil Ann-Christin Klünder

Hannover, 1. Oktober 2021

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 1. Oktober 2021

Alexander Specht

Zusammenfassung

Identifikation von relevanten Metriken zur Analyse von Kommunikation in Entwicklerteams

Einer der häufigsten Ursachen, die zum Scheitern von Softwareprojekten führt, ist die Kommunikation innerhalb von Entwicklungsteams. Nicht vorhandene oder unzureichende Kommunikation sowie negative Stimmung und fehlende Motivation innerhalb eines Entwicklungsteams gehören hierbei zu den größten Faktoren. Um die Stimmung innerhalb des Teams zu identifizieren, können Umfragen oder persönliche Gespräche durchgeführt werden. Allerdings nimmt die Durchführung dieser Methoden viel Zeit in Anspruch. Die Nutzung von automatisierten Methoden bieten deswegen gegenüber den zuvor genannten Methoden zeitliche Vorteile. Eine Methode, die hierfür entwickelt wurde, ist die Sentimentanalyse, mit welcher Aussagen in positiv, negativ oder neutral klassifiziert werden. Es wird angenommen, dass durch diese Methode das Stimmungsbild eines Teams ermittelt werden kann. In dieser Masterarbeit wird diese Annahme als Voraussetzungen verwendet. Der Fokus lag hierbei auf der Untersuchung von Methoden zur Messung von Stimmungen. Dafür wurden verschiedene Metriken evaluiert und anschließend mit einer logistischen Regressionanalyse auf ihre Signifikanz analysiert. Das Ergebnis der Analyse ist, dass es nicht ohne Weiteres möglich ist nur mit objektiven statistischen Metriken Stimmungen zu erkennen. Stattdessen hat sich gezeigt, dass mit Hilfe von Tools wie SentiStrength und den ausgewählten Metriken „Anzahl von Emoticons“, „Durchschnittliche Anzahl an Wörtern“ und dem „Informationsgehalt einer Nachricht“ eine bessere Erkennung von Stimmungen möglich ist.

Abstract

Identifying relevant metrics to analyze communication in developement teams

The main reason software projects fail is the communication within the teams. Not existing communciation, lack of communication, negative mood and lack of motivation within a team belongs to biggest factor for this issue. Possible ways to identify the mood are surveys or personal conversations but these methods consumes a lot of time. The use of automated methods is in contrast more time saving. The SentimentAnalysis was developed to classify statements into positive, negative and neutral statements. It is assumed that this methods can identify the mood of a team. This assumption is checked in this master thesis. The main focus lays on the investigation of mood measuring methods. Therefore different metrics are checked, defined and subsequently analysed with logistic regressions. The final result of this analysis shows, that objective statistic metrics alone are not able to identify mood pictures. In comparsion the use of tools like SentiStrength and the metrics „number of Emoticons“, „average number of words“ and the „information content of a message“ leads to a better identification of the mood.

Glossar

N-Grammen Ein N-Gramm ist eine Vorhersagetabelle, in der Buchstaben oder Wörter auflistet sind. Dabei wird angegeben mit welcher Wahrscheinlichkeit weitere Wörter oder Buchstaben auf den aktuellen Eintrag folgen. Ein Beispiel für die deutsche Sprache ist der Buchstabe e auf den am meisten ein n folgt. 9

Performanz Der Begriff Performance wird in dem Kontext dieser Masterarbeit für die allgemeine Evaluierung von Modellen der künstlichen Intelligenz verwendet. Dabei spielt das Genauigkeitsmaß (Accuracy) und der F1-Wert eine zentrale Rolle. Ist die Performance besser, so sind diese beiden Werte erhöht. 3, 5, 10, 16, 23, 41, 44, 50, 52, 55, 63

Text Mining Text Mining ist eine Disziplin, die sich mit der maschinellen Auswertung von Texten und daraus folgenden Daten beschäftigt. Dazu zählen das Vorbereiten, Analysieren und Evaluieren der gewonnen Daten. 5

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Softwareprojekte bringen in der heutigen Zeit viele Probleme mit sich. Während die Komplexität der Software steigt, steigen auch die Anforderungen an die Entwickelnden [31]. Neben den Anforderungen steigt auch die logistische und planungstechnische Komplexität für die Software [8].

1.1 Motivation

Die häufigsten Ursachen, die zum Scheitern von Softwareprojekten führen, sind fehlerhafte oder mangelnde Kommunikation sowie unklare Anforderungen [31]. Unklare Anforderungen liegen häufig durch mangelnde und unzureichende Dokumentation vor. Diese lassen sich je nach gegebener Organisationsstrukur, wie zum Beispiel agil oder klassisch, einfacher oder schwieriger mit dem Kunden ausarbeiten und nachzubessern [16]. Bei der Kommunikation hingegen ist eine Analyse, um die Ursachen von fehlerhafter oder mangelnder Kommunikation herauszufinden, schwieriger, da die Ursachen sehr komplex sein können[3]. Eine beispielhafte Ursache können persönliche Ursachen eines Entwickelnden sein. Als Lösung des Problems kann ein Gespräch stattfinden, in welchem die Ursachen aufgezeigt und Maßnahmen definiert werden können [26]. Bei einem Entwicklungsteams ist es auch möglich, dass die generelle Kommunikation fehlschlägt. Neben den zuvor genannten Ursachen zählen zu den häufigsten Problemen, dass einige Entwickelnde negative Stimmungen in Form von Kommentaren, Bemerkungen oder Rückmeldungen äußern. Kaur et al. [15] untersuchten diese Tatsache mit einem zeitlichen Verlauf von mehreren Monaten bei Entwickelnden.

Negative Stimmung ist eine Ursache, dass die Motivation bei Entwickelnden nachlässt. Dies wurde erstmals in dem Paper von Pang et al. [27] für soziale Netzwerke gezeigt. Das Messen solcher negativen Stimmung bildet somit ein essentiellen Aspekt, damit sich die grundlegende Stimmung und Motivation im Team verbessert. Stimmung als Indikator meint eine emotional geprägte Gemütserfassung und lässt sich in einzelne Emotionen, wie Freude, Fröhlichkeit, Trauer und Wut nach Lazarus et al. [17], einteilen. Für die ersten beiden Emotionen ergibt sich eine positive Stimmung und für die letzten beiden eine negative Stimmung. Kann weder positive noch negative Stimmung identifiziert werden, so liegt die neutrale Stimmung vor.

Für das Identifizieren von Stimmungen wurde die Sentimentanalyse entwickelt. Diese Analyse klassifiziert einzelne Sätze, Texte oder Aussagen nach ihrer Stimmung. Die Stimmungen sind, wie zuvor erwähnt, positiv, negativ und neutral. Aus der Gesamtheit aller Sätze bzw. Texte ergibt sich ein Gesamtstimmungsbild. Es ist auch möglich Stimmungen über Zeit aufzuzeichnen und einen Stimmungsverlauf zu generieren [15]. Aus diesem Stimmungsverlauf können Maßnahmen abgeleitet werden.

Für die Sentimentanalyse gibt es verschiedene technische Umsetzungen, wie zum Beispiel SentiStrength [33]. Dieses Tool bietet eine Analyse über die drei bekannten Stimmungen und generiert eine Gesamtstimmung für Sätze. Von dem Tool wird der Satz *„Why the hell is this not a bug?“* als negativ klassifiziert. Das Tool gibt als Grund die beiden Wörter *hell* und *bug* an. Beide Wörter wurden nach dem Tool als negativ klassifiziert. In dem Kontext des Software Engineering bietet die Klassifikation jedoch Diskussionsspielraum, denn dieser Satz kann sowohl positiv, negativ als auch sarkastisch wahrgenommen werden. Eine eindeutige Zuordnung ist somit nur bedingt möglich. Daher bildet sich das zentrale Thema dieser Masterarbeit, ob es möglich ist Stimmung zu messen und welche Wirkung die gemessenen Ergebnisse auf den Lesenden haben. Diese Erkenntnis soll dazu genutzt werden die Sentimentanalyse zu verbessern, damit eine genauere Vorhersage von Stimmung möglich ist. Diese verbesserte Vorhersage bietet die Möglichkeit besser die Stimmung von Entwickelnden zu analysieren und genauer auf Probleme innerhalb von Entiwcklungsteams einzugehen. Dadurch soll eine Verbesserung der Stimmung und der daraus folgenden Motivation

gewährleistet werden. Dies kann zu einer höheren Produktivität und zu einer verbesserten Gesamtstimmung innerhalb der Entiwcklungsteams führen [4].

SentiStrength ist domänenunspezifisch entwickelt worden. Für dieses Tool existieren verschiedene Varianten. Eine Variante von SentiStrength ist SentiStrengthSE [12] mit dem Fokus auf Software Engineering. In einer Studie wurde versucht zu messen, wann verschiedene Wörter im Software Engineering Kontext falsch klassifiziert wurden und konzipierte daraufhin ein neues Lexikon mit neu interpretierten Wörtern. Durch diese Änderung wurde das Genauigkeitsmaß (Accuracy) leicht verbessert. Der in dieser Masterarbeit verfolgte Ansatz behandelt die Möglichkeit Stimmung messbar zu machen. Es sollen für diesen Ansatz statistisch messbaren Metriken definiert werden, die das Genauigkeitsmaß verbessern. Das Resultat soll dazu verwendet werden, um die Gesamtstimmung im Team zu verbessern. Ein weiteres Resultat soll die Möglichkeit aufzeigen, dass verschiedene Metriken genutzt werden können, um eine Verbesserung der Ergebnisse zu bewirken. Bisher ist in der Literatur und Forschung dieser Ansatz nicht verfolgt worden und soll in der Zukunft weiteres Verbesserungspotential bieten.

1.2 Zielsetzung

Ziel dieser Masterarbeit ist es Metriken zu evaluieren, die eine Verbesserung des Genauigkeitsmaß bewirken. Dazu sollen geeignete Datensätze evaluiert werden, sodass sie in die Domäne des Software Engineering anknüpfen. Anhand dieser Datensätze sollen die Metriken untersucht werden. Dazu wird zuerst eine statistische Analyse vorgenommen. Bei der Analyse sollen die Metriken herausgefunden werden, die einen messbaren Unterschied zwischen den Stimmungen positiv, negativ und neutral erzeugen und nicht miteinander korrelieren. Im Falle einer Korrelation sollen die Metriken reduziert werden bis keine Korrelation mehr auftritt. Im Anschluss sollen die relevanten Metriken mit Hilfe der logistischen Regressionsanalyse auf einen signifikanten Einfluss auf die Stimmung evaluiert werden. Nachdem die signifikanten Metriken herausgefunden wurden, soll es möglich sein ein geeignetes Maschinenmodell zu trainieren. Dieses Modell soll aufzeigen, dass die Metrik SentiStrength durch die evaluierten Metriken in der Performanz verbessert wurde. Um dies zu überprüfen, sollen die Maschinenmodelle zuerst nur mit der Metrik SentiStrength trainiert werden und anschlie-

ßend sollen die evaluierten Metriken hinzugenommen werden. Nachdem das Maschinenmodell angepasst wurde, soll für die einzelnen Metriken ihr Einfluss auf die Stimmung verglichen werden. Das Resultat ist damit ein trainiertes Maschinenmodell, evaluierte Metriken und der Einfluss der Metriken auf die Stimmung. Diese Masterarbeit soll überprüfen, ob statistische Metriken genutzt werden können, um eine verbesserte Vorhersage von Stimmungen zu gewährleisten statt inhaltliche Anpassungen an bekannten Tools vorzunehmen.

1.3 Struktur

Die Masterarbeit gliedert sich in insgesamt acht Kapitel. Zu Beginn wird auf die Grundlagen sowie verwandte Arbeiten eingegangen. Die verwandten Arbeiten bilden die Basis für die darauf folgenden Kapitel. Im Kapitel Datensätze und Metriken werden Metriken auf ihre Relevanz geprüft und evaluiert. Im weiteren Verlauf werden die Metriken mit Hilfe der logistische Regressionsanalyse ausgewertet und auf Signifikanz geprüft. Im Kapitel des Lösungsansatzes wird ein möglicher Lösungsansatz zur Verbesserung des Genauigkeitsmaß erläutert. In der abschließend Diskussion werden die Ergebnisse reflektiert und weitere Aspekte diskutiert. Zum Abschluss werden in der Zusammenfassung die wichtigsten Resultate zusammengefasst und mögliche Ausblicke werden diskutiert.

Kapitel 2

Grundlagen

In diesem Kapitel befinden sich die Grundlagen, die für ein weiteres Verständnis der Masterarbeit relevant sind. Im ersten Teil wird die Sentimentanalyse mit ihrer Bedeutung für das Software Engineering erklärt. Dabei werden einige Tools vorgestellt, die eine technische Sentimentanalyse implementieren. Im späteren Verlauf des Kapitels werden Grundlagen zum maschinellem Lernen vorgestellt. Diese sind für einen Vergleich der Performanz im späteren Kapiteln essentiell. Im vorletzten Abschnitt wird die logistische Regressionsanalyse erläutert. Diese ist ein zentraler Bestandteil dieser Masterarbeit und dient der Untersuchung einzelner Metriken und ihren Einfluss auf die Stimmung. Im letzten Abschnitt werden die verwendeten Frameworks vorgestellt.

2.1 Sentimentanalyse

Die Sentimentanalyse ist ein Analyseverfahren aus dem Bereich des Text Mining. Die Analyse befasst sich mit der Stimmung von Texten, Sätzen und Aussagen. Dabei soll herausgefunden werden, welche Stimmung ein Text oder Aussage dem Lesenden vermittelt. Im Allgemeinen werden Texte in drei verschiedene Polaritäten eingeteilt: positiv, negativ und neutral. Dabei ist es möglich jeder Stimmung verschiedene Emotionen zuzuordnen, da Menschen diese besser einordnen können [17]. Dabei entsprechen positive Emotionen beispielsweise freudig, enthusiastisch und aktiv und negative Emotionen ängstlich, nervös und verärgert [19]. Diese Emotionen bilden zusammengefasst die drei verschiedenen Stimmungen. In der Literatur ist es üblich, dass die neutrale Emotionen dann verwendet wird, wenn sich weder eine positive noch eine negative Stimmung ableiten lässt. So wird auch in dieser Masterarbeit überwiegend die negative und positive Stimmung von

Aussagen analysiert.
Als Beispiel für eine Aussage aus dem Bereich Software Engineering ist folgende Aussage:

„Awesome! This tool is cool.“

Diese Aussage kommt aus einem Kommentar vom Stack Overflow Datensatz (Vergleiche Kapitel 4). Nach der Klassifikation aus dem Paper von Novielli et al. [25] haben Entwickelnden diese Aussage als positiv bewertet. Der Grund dafür ist, dass die Wörter „Awesome!“ und „cool“ positive Emotionen wie enthusiastisch auslösen. Werden die einzelnen Wörter des Satzes analysiert, so ist es nicht möglich weitere emotionale Stimmung festzustellen.

Die Hauptaufgabe der Sentimentanalyse ist es ein gesamtes Stimmungsbild zu erzeugen. Beispiele für Anwendungsgebiete solcher Stimmungsbilder können Kundenrezessionen, soziale Netzwerke und Nachrichten sein. In jedem dieser Gebiete ist es wichtig einen Überblick über Themen oder Produkten zu erhalten. Es ist zum Beispiel möglich herauszufinden, ob bestimmte Produkte häufig positive Rezessionen besitzen. Aus dieser Analyse ist es möglich Maßnahmen für die Produkte abzuleiten und eine Verbesserung bzw. Attraktivitätssteigerung bei den Produkten zu erwirken. In sozialen Netzwerken gab es in den letzten Jahren häufiger Probleme mit böswilligen Kommentaren. Durch die Sentimentanalyse ist es möglich diese automatisiert ausfindig zu machen [10] und zu entfernen.

Für die technische Realisierung der Sentimentanalyse existieren verschiedene Möglichkeiten, die in den folgenden Abschnitten genauer erklärt werden. Für die Realisierung existieren verschiedene Methodiken bzw. Vorgehensweise. Die erste Methodik beinhaltet das sogenannte **lexikonbasierte** Verfahren. Hierbei besitzt die Anwendung ein Lexikon, welches bereits von Entwickelnden nach ihrer Polarität klassifiziert wurde. Dabei erfolgt das Ergebnis durch das Evaluieren jedes einzelnen Wortes eines Satzes. Das resultierende Ergebnis bringt Auskunft über die Stimmung des Satzes und wie diese zu verstehen ist. Diese Auskunft wird durch das Zusammenfassen und Analysieren ermittelt. Die Polarität kann dabei unterschiedlich gewichtet sein. In der Literatur wird häufig für ein negativ geprägtes Wort (-1) und für ein positiv geprägtes Wort (+1) gewählt. 0 bildet dabei eine neutrale Stimmung. Je nach Verfahren kann der Intervall größer oder niedriger ausfallen. Ein Beispiel für eine mögliche erweiterte Unterteilung ist,

dass die negativen Wörter unterschiedlich stark negativ bzw. positiv gewertet werden, wodurch sich eine Skala von beispielsweise (-5) - (-1) für die negative Stimmung ergibt (vergleiche hierzu 2.1.1). (-5) repräsentiert dabei ein sehr negatives Wort. Als zweite Methodik neben der lexikonbasierten Variante gibt es die Möglichkeit nach **Keywords (keywordbasiert)** zu suchen. Keywords sind spezielle Wörter, Ausdrücke oder Phrasen, die eine Polarität innerhalb des Satzes erzeugen. Als Beispiel dient aus dem oberen Satz der Ausdruck „Awesome!“.
Die ersten beiden Methodiken zählen zu den linguistischen Analyse, da hier bewusst Wörter durchsucht werden, die eine bestimmte Stimmung erzeugen. Neben der linguistischen Analyse existiert die Möglichkeit ein besseres Verständnis über Texte mit Hilfe von **maschinellem Lernen** zu erhalten. Bekannt hierfür ist die Anwendung „Cloud Natural Language“ von Google[1]. Bei dieser Anwendung wurden Rezensionen von Nutzenden analysiert und anschließend mit einem neuronalen Netz trainiert. Durch das Einlesen vieler bekannter Rezensionen mit ihrer Wertung ist es möglich ein solide trainiertes Modell zu bekommen, damit zukünftige Kommentare klassifiziert werden können.

Bei der Analyse einzelner Wörter können fehlerhafte Vorhersagen entstehen, indem die Sätze beispielsweise doppelt verneint oder eine einfache Verneinung besitzen. Der Satz „This tool is **not** cool.“ besitzt nun eine Verneinung und diese führt dazu, dass nach der linguistischen Analyse dieser Satz insgesamt neutral bewertet werden würde. Der Grund für dieses Resultat liegt den Wörtern *not* (-1) und *cool* (+1) zugrunde. Daher werden bei diversen Verfahren die Lexika getrennt und einzeln interpretiert. Aus dem vorherigen Satz würde dann das negative Wort *not* als Verneinung eines zukünftigen Wortes dienen und den Satz als insgesamt negativ bewerten. Andere Lexika in diesem Kontext sind beispielsweise verstärkende Wörter wie sehr, viel oder mehr. Diese Wörter verstärken kommende Aussagen und ändern die Bewertung. Bei Methodiken aus dem maschinellen Lernen können diese Probleme auch auftreten. Wenn ausreichend Trainingsdaten für das Training verfügbar sind, so können diese Probleme umgangen werden.

Die Sentimentanalyse bietet sich für viele Anwendungsgebiete an, unter anderem bei der Analyse in Bereich des Software Engineering.

[1] https://cloud.google.com/natural-language

Für diese Domäne existieren einige Probleme, die mit Hilfe einiger der folgenden Tools behoben werden sollen. So besteht bei dieser Domäne das Problem, dass zum Beispiel das Wort „Bug“ als negativ klassifiziert werden würde, da dieses Wort einen Fehler im System aufzeigt und in allgemeinen Lexika (wie SentiStrength) als negativ gekennzeichnet wird. Im Bereich des Software Engineering muss ein Bug nicht negativ sein, da dieser dazu beitragen kann Fehler oder Sicherheitslücken aufzudecken. Weitere Wörter aus diesem Bereich wie „Super“ „„Error“ und „Value“ werden in diesem Kontext auch häufig anders klassifiziert [12] und führen zu einer Veränderung der Vorhersage.

2.1.1 SentiStrength

SentiStrength ist ein lexikonbasierter Klassifizierer, welcher die Möglichkeit bietet Sätze und Texte nach ihrer Stimmung zu klassifizieren. Das Prinzip ist hierbei, dass jedem Wort ein Wert zwischen (-1) - (-5) (sehr negativ) und (+1) - (+5) (sehr positiv) zugeordnet wird. 0 repräsentiert die neutrale Stimmung. Diese Zuordnung wird durch Lexika(Sentiment, Boosterwörter, Negationen, Film und weitere) bestimmt, die zuvor von Thelwall et al. [33] erstellt wurden. Nach dem Ende eines Satzes werden die Minima und Maxima der einzelnen Wörter bestimmt und anschließend zusammengefasst.

Wort	**Bewertung**
crying	-4
decrepit	-5
enjoy	+3
thrill	+4

Tabelle 2.1: Auszug aus dem SentiStrength Lexikon für polaritäre Wörter.

Ist der Stimmungswert > 0, so wird der Satz als positiv bewertet und bei einem Wert < 0 wird der Satz als negativ gewertet. Bei einem Wert von 0 wird der Satz als neutral bewertet. Die zuvor erwähnten Lexika bestehen zum Beispiel aus Boosterwörtern, wie sehr, viel oder wenig, aus Sentimentwörtern, diese sind Wörter mit einer bestimmten Stimmung (siehe Tabelle 2.1), aus Negationen und aus Phrasen. Insgesamt existieren 15 verschiedene Lexika. Jedes Wort aus einem dieser Lexika wird anders interpretiert und analysiert.

Als Beispiel für die Analyse mit SentiStrength dient folgender Satz:

„Why the hell is this not a bug?“

Dieser Satz ist ein Auszug aus dem Kapitel Datensätze und Metriken (siehe Kapitel 4). Nach SentiStrength haben die Wörter *hell* und *bug* eine Bewertung von (-2). Somit ist die negative Stimmung insgesamt (-2) und die positive Stimmung (1). Die positive Stimmung ergibt sich aus dem Kontext, dass jeder Satz immer bei (+1) und (-1) startet. Die Auswertung ergibt insgesamt eine Wertung von (-1), wenn die Summe aus den Werten gebildet wird. Wird aus dem Satz das Wort *hell* entfernt, so würde sich die Stimmung nach SentiStrength nicht ändern, da *bug* und *hell* die selbe Bewertung haben und das Maximum ausgewertet wird.

2.1.2 SentiStrengthSE

SentiStrength ist lexikonbasiert und für eine Sprache ohne spezifischer Domäne entwickelt worden. Für die spezielle Domäne des Software Engineering wurde von Islam und Zibran [12] ein angepasstes Lexikon entwickelt. In diesem Lexikon werden bekannte Wörter wie „decrease“ und „eliminate“ nicht mehr als negative Wörter gewertet. Mit Hilfe dieses angepassten Datensatzes können Kommentare und Aussagen von Plattformen wie Stack Overflow besser klassifiziert werden[2].

2.1.3 Senti4SD

Senti4SD ist eine Erweiterung von SentiStrength, einem Keyword-basierten Lexikon und einer semantischen Analyse. Die Keywords werden mit Hilfe von N-Grammen ermittelt und ausgewertet. Die semantische Analyse erfolgt durch das Zerlegen einzelner Ausdrücke in Vektoren. Diese Vektoren werden aufsummiert und ausgewertet. Das Ergebnis ist ein Verbesserung der Vorhersage für die Software Engineering Domäne [25].

2.1.4 SentiCR

Sentiment for Code Review ist ein Framework, welches maschinelles Lernen verwendet, um die Polarität von Sätzen festzustellen. Im Gegensatz zu den vorherigen Methoden wurde eine aufwendige Analyse

[2]Diese Aussage ergibt sich aus einer Analyse dem Paper von Islam und Zibran

von Code Reviews durchgeführt. Die Auswertung erfolgt in acht Teilschritten (vergleiche [1]).

2.1.5 NLTK

NLTK ist ein Framework, welches verschiedene Metriken und Auswertung für die Linguistik bietet. In diesem Framework befindet sich unter anderem eine Sentimentanalyse, die mit Hilfe von maschinellem Lernen eine Auswertung bietet. Sie klassifiziert die Werte zwischen (-1) (sehr negativ) und (+1) (sehr positiv). Neben der Sentimentanalyse gibt es die Möglichkeit Wortarten (Part-of-Speech Analyse) zu bestimmen.

2.2 Maschinelles Lernen

Dieser Abschnitt beinhaltet die Grundlagen zu ausgewählten Methoden des maschinellen Lernens und Möglichkeiten der Evaluierung diverser Maschinenmodelle. Mit diesen Modellen soll evaluiert werden, wie sich die Metriken, die im Kapitel 4 ausgewertet werden, im Bezug auf die Stimmung verhalten. Bewusst wurden die folgenden Modelle ausgewählt, da diese eine einfache Interpretation und Implementierung besitzen, aber auch für diese Zwecke einen ausreichenden Umfang bieten. Jedes dieser Modelle besitzt eine andere mathematische Vorgehensweisen, wodurch andere Resultate vorliegen können. Ein weiterer Grund für das Verwenden von maschinellen Lernen ist die Möglichkeit viele Daten auszuwerten, zu analysieren und zu interpretieren.

2.2.1 Metriken des maschinellen Lernens

Für die Interpretation der Ergebnisse werden in den kommenden Kapiteln Confusion Matrizen verwendet. Aus diesen ist es möglich abzulesen, ob ein Modell eine bestimmte Performanz aufweist und wie die Verteilung der überprüften Daten ist. Das Schema für die Confusion Matrix ist wie folgt:

	C1	C2	$\sum$
C1	TP (True positve)	FN(False negative)	P
C2	FP(False positive)	TN(True negative)	N
$\sum$	P'	N'	

Tabelle 2.2: Die Tabelle zeigt das generelle Schema der Confusion Matrix.

In dieser Confusion Matrix werden die klassifizierten Daten eingetragen, nachdem ein Modell trainiert wurde. Diese Confusion Matrix lässt sich für jedes Modell generieren. Für eine korrekte Klassifizierung beispielsweise wird der Wert TP um 1 erhöht. Für die einzelnen Metriken des maschinelles Lernens ergeben sich folgende Formeln mit Erklärungen:

Genauigkeitsmaß (Accuracy): Das Genauigkeitsmaß lässt sich durch folgende Formel ermitteln:

$$Genauigkeitsmaß(M) : \frac{TP + TN}{P + N}$$

Dieses Maß gibt an wie viele Aussagen korrekt klassifiziert wurden. Bei 100% sind alle Aussagen korrekt klassifiziert worden.

Präzision (Precision): Die Präzision lässt sich durch folgende Formel ermitteln:

$$Präzision(M) : \frac{TP}{TP + FP}$$

Dieses Maß gibt an wie viele Aussagen als positiv klassifziert wurden und welche auch tatsächlich positiv waren. Der Unterschied zur vorherigen Metrik ist, dass diese für eine konkrete Klasse berechnet wird.

Rückruf (Recall): Der Recall lässt sich durch folgende Formel ermitteln:

$$Recall(M) : \frac{TP}{TP + FN}$$

Dieses Maß gibt an wie viele Aussagen als positiv klassifziert wurden unabhängig davon, ob sie positiv waren. Bei dieser Metrik gilt das Ergebnis für eine konkrete Klasse.

F1-Wert: Der F1-Wert lässt sich durch folgende Formel ermitteln und besteht aus dem Recall und der Präzision:

$$F1 - Wert : \frac{2 \cdot precision(M) \cdot recall(M)}{precision(M) + recall(M)}$$

Dieses Maß ist ein Mittelmaß, welches das Verhältnis vom Recall und

der Präzision angibt. Je größer dieser Wert ist, desto genauer sind die Vorhersagen.Bei dieser Metrik gilt das Ergebnis für eine konkrete Klasse.

2.2.2 Support Vector Machine (SVM)

Die Support Vector Machine (SVM) ist ein Maschinenmodell, die eine Klassifikation und Regression durchführen kann. In den letzten Jahren ist sie ein beliebtes Modelle geworden, da sie komplexe Daten effizient auswerten kann [13]. Dabei besteht die SVM aus den beiden folgenden Komponenten:

Support Vektoren: Die Support Vektoren liegen an den Daten und markieren die Punkte, die am besten die Breite der Hyperbene beschreiben. Für die Breite gilt, dass nach dem maximalen Wert gesucht wird.

Hyperbene: Die Hyperbene ist der Bereich, der zwischen den Support Vektoren liegt und den Bereich zwischen den separierbaren Daten aufspannt.

Das Ziel der SVM ist es, dass Modell so zu trainieren, dass die Hyperebene die höchste Breite annimmt. Für das Trainieren und Anpassen der Breite werden die Gewichtsvektoren berechnet. Dafür muss die folgende allgemeine Gleichung erfüllt sein:

$$w \cdot x_i + b = 0 \ , \ i \in \mathbb{N}$$

Hierbei entspricht w dem Gewichtsvektor, x_i dem Eingabevektor und b ist der Bias. Für das Anpassen der Gewichtsvektor im konkreten Fall (+1/-1) muss folgende Formel erfüllt sein:

$$w \cdot x_i + b \geq +1, wenn \ y_i = +1$$
$$w \cdot x_i + b \leq -1, wenn \ y_i = -1$$

Hierbei entspricht y_i dem Label der Daten. Dieses Label kann den Wert +1/-1 annehmen. Das Modell wird trainiert, sodass die Abstände zwischen den Daten und der Hyperebene, einschließlich dem Faktor ζ, am höchsten sind. Der Faktor ζ kann hinzugefügt werden, um den Abstand der Support Vektoren zueinander zu verändern. Dieser kann hilfreich sein, damit verschiedene Ergebnisse evaluiert werden können.

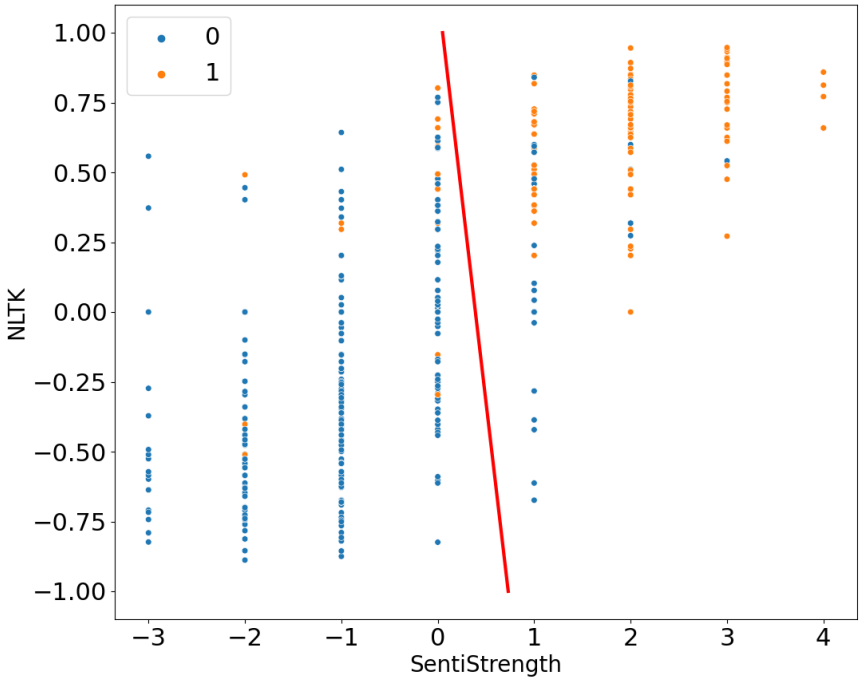

Abbildung 2.1: In dieser Grafik wurde eine SVM trainiert. Die rote Linie zeigt die Hyperebene. Verwendet wurde ein Training mit NLTK und SentiStrength für eine Stimmungsvorhersage (0:negativ, 1:positiv) des Datensatzes aus Kapitel 4.

Das Einbauen des Faktors ζ erfolgt durch das Anpassen der obigen Gleichung:

$$w \cdot x_i + b \geq 1 - \zeta, \; i \in \mathbb{N}, \zeta \in \mathbb{Q}$$

Der große Vorteil der SVM gegenüber anderen Verfahren ist, dass es möglich ist mit Hilfe des *Kernel Tricks* auch komplexere Daten zu verarbeiten, ohne dabei einen höheren Rechenaufwand zu erhalten. Bei dem *Kernel Tricks* werden die Daten aus einem Mehrdimensionalen Raum auf eine einfache Ebene projiziert. Durch die Projektion ist es möglich eine Hyperebene im einfachen Raum zu trainieren. Bekannte Kernel für die SVM sind: linear, polynomiell und radialbasiert.

2.2.3 Bayes Klassifkation

Die bayesche Klassifikation basiert auf den bayeschen Theorem:

$$P(C|A) = \frac{P(A|C)P(C)}{P(A)}$$

Hierbei ist $P(C|A)$ der Posterior, $P(A|C)$ die likelihood, $P(A)$ die Evidenz und $P(C)$ der Prior. Ziel hierbei ist es mit Hilfe des Theorems die verschiedenen Wahrscheinlichkeiten zu multiplizieren, sodass eine Vorhersage getroffen werden kann. Das Ermitteln einer Aussage erfolgt durch folgende Formel:

$$c = argmax_{c \in C} P(c) \prod P(A_i|c)$$

Dabei entspricht C der Klasse und $P(A_i|c)$ der Wahrscheinlichkeit, dass A_i eintritt und der Bedingung, dass c eingetreten ist. Dabei können bei der Vorhersage Korrekturen vorgenommen werden. Mögliche Anpassungen bzw Korrekturen sind:

$$Original : P(A_i|c) = \frac{N_{ic}}{N_c}$$
$$Laplace : P(A_i|c) = \frac{N_{ic} + 1}{N_c + k}$$
$$m - Anpassung : P(A_i|c) = \frac{N_{ic} + mp}{N_c + m}$$

In der Formel entspricht k der Anzahl der Klassen, m der Anpassungsparameter und p der Wahrscheinlichkeit, dass der Prior eintritt. Eine Anpassung muss nur dann vorgenommen werden, wenn beispielsweise eine Division durch 0 vorliegt. In diesem Fall kann die Laplace-Korrektur eingesetzt werden.

2.2.4 Decision Tree

Ein gerichteter Graph G ist eine Datenstruktur und besteht aus Kanten E und Knoten V. Jede Kanten wird durch zwei Knoten $e = (u, v) \in E$ beschrieben und verbindet die beiden Knoten in eine Richtung miteinander.

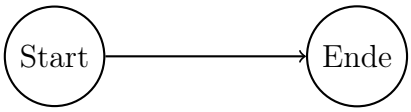

Abbildung 2.2: Dies ist ein gerichteter Graph mit zwei Knoten.

Ein Baum ist ein azyklischer zusammenhängender gerichteter Graph mit einer Wurzel und mehreren Blättern. Die Wurzel bildet bei einem Baum jener Knoten, der keine eingehenden Kanten besitzt, sondern nur ausgehende. Die Blätter besitzen im Gegensatz dazu nur eingehende Kanten, aber keine ausgehenden. Bei einem Entscheidungsbaum bildet die Wurzel den Anfang und jeder Knoten besitzt mit jeder möglichen Kante eine Wahlentscheidung. Diese Entscheidung wird durch bestimmte Kriterien bestimmt. Ein Beispiel dient folgender Entscheidungsbaum:

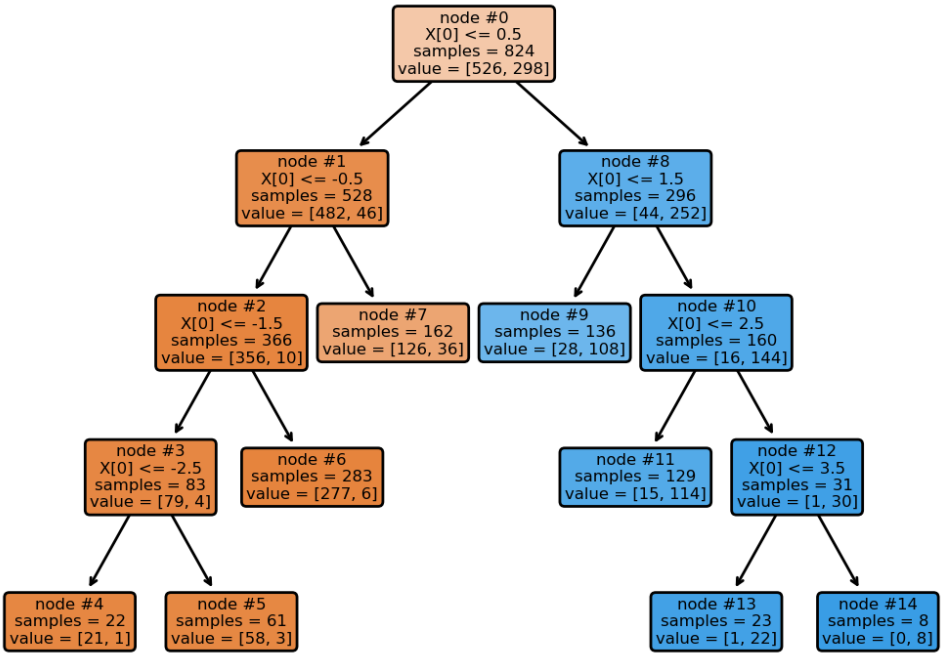

Abbildung 2.3: Die Grafik zeigt einen beispielhaften Entscheidungsbaum der Tiefe 4 mit SentiStrength als Metrik.

Die Summe der Entscheidungen an jeder Abzweigung bildet das Endergebnis. Bei jeder Abzweigung wird eine einfache Entscheidung getroffen. Die Sequenz der Entscheidungen von der Wurzel bis zu einem

Blatt nennt sich Pfad oder Entscheidungspfad. Damit ein Entscheidungsbaum aufgestellt werden kann, sind Trainingsdaten notwendig. Mit Hilfe der Trainingsdaten ist es möglich einen Baum aufzustellen. Für die Aufstellung existieren verschiedene Varianten. Als bekanntestes Beispiel wird der Gini-Index verwendet:

$$Gini(D) = 1 - \sum_{j=1}^{k} p_j^2$$

p repräsentiert hierbei die Wahrscheinlichkeit, dass ein Parameter eintritt. Für jede Eingabe wird die obige Formel angewendet. Der Parameter mit dem geringsten Gini-Index wird verwendet und eingesetzt. Danach wird das Verfahren solange durchgeführt bis das letzte Blatt eine Ausgabe erzeugt.

2.2.5 Random Forest

Beim Random Forest Verfahren werden zufällig Entscheidungsbäume erstellt und trainiert. Die Entscheidungsbäume mit der geringsten Wahrscheinlichkeit werden entfernt. Durch das Erstellen mehrerer Entscheidungsbäume lässt sich der Algorithmus parallelisieren. Die Vorhersage wird bestimmt durch das parallele Aufrufen der einzelnen trainierten Entscheidungsbäume und das evaluieren der einzelnen Ergebnisse.

2.2.6 Automated Machine Learning

Automated Machine Learning ist ein Verfahren aus dem Bereich des automatisierten maschinellem Lernens. Dieses Verfahren führt eine Hyperparameter-Optimierung durch. Hierbei werden die bekanntesten Verfahren wie die SVM oder Random Forest genutzt, um eine bessere Performanz zu erhalten [7]. Die Verfahren werden nacheinander trainiert und nach der besten Performanz ausgewählt. Dieses Framework lässt sich in Python einbinden.

2.3 Regressionsanalyse

Die logistische Regressionsanalyse ist ein Verfahren, bei dem geprüft werden soll, ob es einen Zusammenhang zwischen unabhängigen bzw. abhängigen Variablen gibt. Der wichtigste Unterschied zu einer einfachen Regression ist, dass das Modell auf binär verteilten Daten

zurückgreift. Daher eignet sich das Modell sehr gut zum Identifizieren von positiven und negativen Stimmungen. Dieses Analyseverfahren wird in den späteren Kapiteln dafür verwendet, verschiedene Metriken zu überprüfen und zu evaluieren. Dabei soll die Analyse mit Hilfe eines Signifikantstest ermitteln, welche unabhängigen Variablen welchen Einfluss auf die abhängige Variable hat. Das Modell nutzt als Beschränkung der Werte folgende Gleichung:

$$f(x) = \frac{1}{1 + e^{-x}}$$

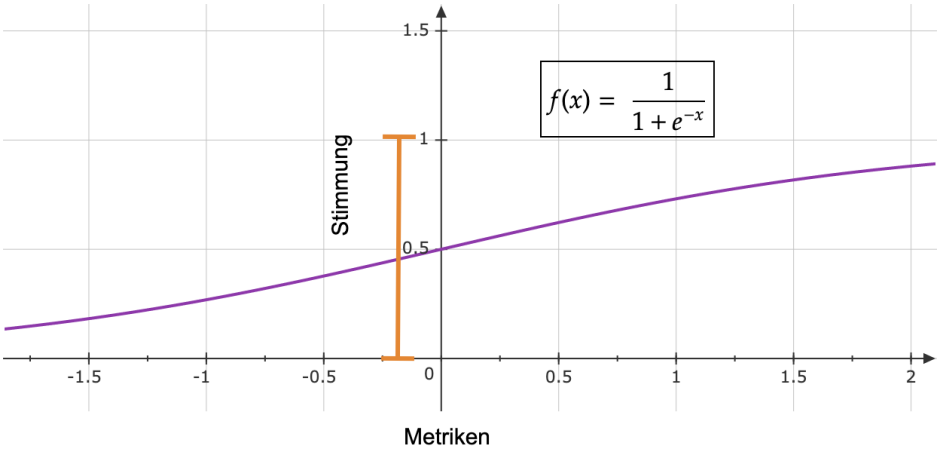

Abbildung 2.4: Die Grafik zeigt den Kurvenverlauf der logistischen Regressionsanalyse

Aus der Funktion lasen sich die Koeffizienten berechnen. Diese werden durch folgende Gleichung berechnet:

$$P(y = 1) = \frac{1}{1 + e^{(\beta_1 + \beta_2 * x_1 + ... + \epsilon)}}$$

Das β_i steht für die Koeffizienten, ϵ ist der Standardfehler, x_i bilden die unabhängigen Variablen und e steht für die eulersche Zahl. Ziel des Verfahrens ist es, die einzelnen Koeffizienten zu berechnen und anzupassen. Das Schätzen der einzelnen β_i erfolgt durch verschiedene Verfahren.

Eines der bekanntesten Verfahren ist die Kleinste-Quadrat-Schätzung:

$$\sum_{n=0}^{N} Y_i - f(X_i, \beta))^2$$

Hier wird das kleinste β gesucht, dass die Gleichung erfüllt. Im Anschluss an das Herausfinden der passenden Parameter können die unabhängigen Variablen auf ihren Einfluss auf das Modell evaluiert werden. Für das Modell gibt es verschiedene Indikatoren, die ermittelt werden, um verschiedene Aussagen für das Modell treffen zu können. Zum einen kann durch einen Signifikanztest ermittelt werden, ob eine Variable einen signifikanten Einfluss auf das Ergebnis hat. Dies wird im späteren Kapitel dafür verwendet, um festzustellen, ob eine Metrik einen Einfluss auf das Ergebnis hat und welcher Einfluss vorliegt. Ein weiterer Indikator gibt die Log-Likelihood an. Diese repräsentiert die Verschiebung bzw. den Einflussgröße auf das Modell. Ein negativer Wert steht für eine negative Tendenz der Einflussrichtung des trainierten Modells.

2.4 Frameworks

In diesem Kapitel werden die Frameworks kurz erläutert. Dabei werden nur jene erwähnt, die eine spezielle Funktionalität besitzen. Alle der folgenden Frameworks wurden in Python implementiert und sind frei erhältlich. Ausgeführt und installiert wurden die Erweiterungen in der PyCharm IDE von IntelliJ.

2.4.1 Statsmodel

Statsmodel ist ein Framework für Analyseverfahren[3]. Mit Hilfe dieses Frameworks ist es möglich die logistisch Regressionsanalyse durchzuführen. Dabei bietet sie auch die Möglichkeit einen Signifikanztest für jede Variable durchzuführen, die Log-Likelihood und das Genauigkeitsmaß des Modells zu berechnen. Ein weiterer Vorteil dieses Frameworks ist es, dass es kompatibel zu der Programmiersprache *R* ist und Formeln tranferiert werden können. Neben der logistischen Regressionsanalyse gibt es weitere Verfahren, wie die ANOVA oder statistische Analysen, die in dem Framework implementiert sind.

[3]https://www.statsmodels.org/stable/index.html

2.4.2 Sklearen

Sklearn ist ein sehr populäres Framework für künstliche Intelligenz[4]. In diesem Framework befinden sich die zuvor vorgestellten Maschinenmodelle und viele Weitere. Die Modelle bieten eine vollständige Übersicht und Anpassung der Parameter. Weiterhin bietet das Framework auch die Möglichkeiten Confusion Matrizen zu erzeugen. Diese finden in den kommenden Kapiteln ihren Einsatz.

2.4.3 Weitere Frameworks

Neben Sklearn bietet AutoSklearn eine Hyperparamteroptimierung ausgewählter Maschinenmodelle (Vergleiche Kapitel 2.2.6). Pandas und Numpy sind bekannte Frameworks für Tabellen und weiteren Datenstrukturen. Für SentiStrength und NLTK existiert jeweils eine Python Variante, die für die Analyse verwendet wird. Spellchecker ist ein Framework um Rechtschreibfehler zu erkennen und mögliche Verbesserungen durchzuführen. Für die Grafiken in den folgenden Kapiteln wurden die beiden Frameworks Seaborn und Matplotlib verwendet. Alle weiteren Funktionen und Methoden sind Standard der Python-Programmiersprache 3.8.

[4]https://scikit-learn.org/stable/index.html

Kapitel 3

Verwandte Arbeiten

In diesem Kapitel werden Verwandte Arbeiten beschrieben. Diese bilden die Basis für diese Masterarbeit und der einhergehenden Analyse. Den Anfang bzw. die Basis der Sentimentanalyse bildet das Paper von Bang und Lee [27]. Sie nutzten diese Analyse, um zu verstehen und zu lernen wie Menschen denken und fühlen. Dabei nutzten die Autoren Online-Blogs und Review-Seiten zur Auswertung. Ihr Ziel war es Texte als positiv oder negativ zu klassifizieren. Hierfür suchten die Autoren gezielt nach Wörtern und Ausdrücken, die eine Polarität aufzeigen. In einem vorherigen Paper aus dem Jahr 2008 [28] haben die Autoren bereits Schwierigkeiten aufgezeigt, dass einzelne Wörter eine negative Stimmung aufzeigen, jedoch ganze Sätze nicht zwangsläufig als negativ eingeordnet werden müssen. 2010 folgte auf die Analyse von Stimmung ein automatisiertes Tool SentiStrength von Thelwall et al. [33]. SentiStrength wurde mit Hilfe umfangreicher Studien erstellt. Dabei haben die Autoren Lexika erstellt in dem verschiedene Wörter enthalten sind, die eine Polarität bzw. Stimmung aufzeigen. In dem Tool wurde auch bedacht, dass es Wörter gibt, die eine verstärkte Wirkung auf Polaritäten haben (Boosterwörter). In den Folgepapern von Thelwall et al. wurden dem Tool weitere Sprachen hinzugefügt. Das Tool ist online frei verfügbar[1]. 2014 folgte das erste Paper über die Domäne des Software Engineering im Bezug auf die Sentimentanalyse. In dem Paper von Murgia et al. [20] wurden mehrere Entwickelnde zu Aussagen befragt und sollten diesen Emotionen zuordnen. Dabei haben die Forschenden herausgefunden, dass bei einem Auswerten von Aussagen lediglich zwei Entwickelnde notwendig sind, damit ein fundiertes Bild von einer Stimmung erzeugt werden kann. Eine weitere Erkenntnis aus den

[1] http://sentistrength.wlv.ac.uk/

Studien war, dass der Kontext zu verschiedenen Aussagen keine weitere Beeinflussung darstellt. Im Gegensatz dazu werden mehr als nur zwei Entwickelnden für eine Vorhersage benötigt. Das folgende Paper von Ortu et al. [18] behandelte das Klassifizieren von verschiedenen Aussagen bekannter Plattformen wie GitHub, Stack Overflow und JIRA. Dabei wurden diese Plattformen bewusst gewählt, da diese eine hohe Nutzendenanzahl haben und stellvertretend für eine Analyse im Bereich des Software Engineering sind. Die Klassifikation der Aussagen von den Plattformen erfolgte dabei nach einem Paper von Parrott [29] aus dem Jahr 2001. Der Autor hatte herausgefunden, dass sechs Emotionen die Stimmung des Menschen beschreiben können. Die Emotionen sind: Liebe, Freude, Überraschung, Zorn, Trauer und Angst. Nach diesen Emotionen sollten Entwickelnde die Aussagen klassifizieren. Das Resultat war, dass die Emotionen Liebe, Freude und Trauer am häufigsten vorgekommen sind. In zwei weiteren Studien klassifizierten Entwickelnde weitere Aussagen mit den drei vorher genannten Emotionen, wobei in die zweite Studie zusätzlich noch die Emotion Zorn für den Ausgleich aufgenommen wurde. Im Jahr 2017 wurde eine Weiterentwicklung von SentiStrength entwickelt: SentiStrengthSE [12]. Diese Weiterentwicklung beruht auf der Tatsache, dass SentiStrength für einen allgemeinen Kontext entwickelt wurde und für keine spezielle Domäne. In einer Studie haben Islam et al. herausgefunden, dass Wörter wie „decrease“ negative Stimmung im allgemeinen Kontext erzeugen[2]. Aufgrund dessen wurden die Lexika von dem Tool SentiStrength auf die spezielle Domäne Software Engineering angepasst. Durch das Anpassen wurden bessere Ergebnisse beim Vorhersagen von Stimmung erzielt. Im gleichen Jahr erschien das Tool SentiCR [1]. Dieses Tool ist auch eine Weiterentwicklung von SentiStrength, aber hier wurden die Datensätze angepasst und vorverarbeitet. Eine weitere Besonderheit ist, dass nicht die Lexika angepasst wurden, sondern zusätzliche Metriken und Ausdrücke im Kontext ausgewertet wurden. Neben der lexikonbasierten Analyse verfügt das Tool über eine semantische Analyse, bestehend aus Token. Das Resultat brachte auch eine Verbesserung der Vorhersage. Thelwall et al. entwickelten auch 2017 eine abgeänderte Variante von SentiStrength: TensiStrength [32]. Dieses Tool ermittelt statt einer Stimmung das Stresslevel einer Aussage. Dieses Tool soll dabei helfen herauszufinden, ob Personen gestresst sind, wenn sie Kommentare schreiben. Neben den Tools und Erweiterungen wurde im selben

[2]Nach SentiStrength (Vergleiche Kapitel 2.1.1)

Jahr von Jongeling et al. [14] eine ausführliche Studie durchgeführt, um zu ermitteln, wie die zuvor genannten Sentimentanalysetools funktionieren und Vorhersagen tätigen. Dabei hat sich ergeben, dass die Tools bei der Klassifikation bereits deutliche Unterschiede aufweisen. Die Tools haben im besten Fall eine Übereinstimmung der Vorhersage von 22%[3]. Daraus resultierte die Frage, dass wenn sich nicht die Tools auf eine Stimmung einigen, wie könnten es dann Entwickelnde. Eine weitere Studie von Nasif et al. [11] hatte 2018 sich die Frage gestellt, ob es möglich sei aus Kommentaren Höflichkeit abzuleiten. Dabei hatte sich herausgestellt, dass die Entwickelnden verschiedener Meinungen waren was Höflichkeit bedeutet.

2018 erschien das Paper von Calefato et al.[6] mit einem verbesserten Tool für den Bereich des Software Engineering: Senti4SD. Dabei wurden ähnliche Features genutzt wie bei SentiCR. Der Unterschied hierbei ist, dass dieses Tool für die vollständige Domäne des Software Engineering zuverlässige Ergebnisse liefern soll, statt lediglich Code Reviews. Im selben Jahr erschien ein Benchmark [24] bei dem mit Hilfe von künstlicher Intelligenz überprüft wurde, wie gut die einzelnen Tools im Vergleich sind. Dabei wurden die Tools als Trainingsdaten in eine SVM eingesetzt und als resultierendes Label wurde eine Klassifikation von Entwickelnden verwendet. Im Anschluss wurden die Tools paarweise in eine SVM integriert. Das Ergebnis aus dieser Studie war ein Genauigkeitsmaß von beispielsweise 87% bei Senti4SD. Julian Horstmann [9] hat 2019 in seiner Masterarbeit den Grundstein für diese Masterarbeit gelegt. In seiner Arbeit verglich er verschiedene Metriken und ihren Einfluss auf den Lesenden. Sein Kontext war eine Zusammenarbeit mit einem Unternehmen, woraus auch die Daten stammen. Viele der entwickelten und entdeckten Metriken wurden in dieser Arbeit weiterverwendet. In drei weiteren Papern von Novielli et al. [22][23][21] wurden weitere Studien über die Performanz von Sentimentanalysetools durchgeführt. Als Resultat wurde aufgezeigt, dass verschiedene Plattformen wie JIRA, Stack Overflow und GitHub sich zwar ähnlich in der Stimmung verhalten, aber das Genauigkeitsmaß bei einem trainierten Modell sich jedoch unterscheidet. Bei den Tools SentiCR und SentiStrengthSE war das Genauigkeitsmaß bei Cross-Plattformen deutlich schlechter als bei Senti4SD. 2021 wurde diese Studie fortgeführt mit weiteren Analysetools. Im selben Jahr wurde von Obaidi und Klünder [26] eine Literatursuche durchgeführt.

[3]Vergleich von SentiStrength und NLTK

Diese Literatursuche beschäftigte sich mit dem Zusammentragen vom aktuellen Forschungsstand der Sentimentanalyse. Dabei wurden 80 Paper mit einem Bezug auf die Domäne des Software Engineering analysiert und evaluiert. Das Ergebnis ist eine umfangreiche Sammlung von Problemen, Lösungsansätzen und Weiterentwicklungen der Sentimentanalyse.

Kapitel 4

Datensätze und Metriken

Dieses Kapitel beinhaltet eine Analyse von Metriken im Bezug auf die zu evaluierenden Datensätze. In der Einleitung wurde bereits erwähnt, dass die Sentimentanalyse Diskussionsspielraum bietet, da nicht jede Aussage von jeder Person gleich interpretiert wird (Vergleiche Kapitel 2). Statistische Metriken sollen nun dabei helfen eine unterstützende Funktion zu gewährleisten, um die Vorhersage von bekannte Tools zu verbessern, damit eine zukünftige Klassifikation besser gelingen kann. Dazu werden in diesem Kapitel verschiedene Metriken vorgestellt und welche Wirkung sie auf die Datensätze haben.

Im Kontext des Software Engineering bieten die Plattformen Stack Overflow, Jira und GitHub eine solide Basis für eine Analyse. Stack Overflow sowie Jira bieten durch die Konversationen mehrerer Entwickelnden eine bessere Referenz, da bei GitHub überwiegend Code mit Kommentaren zu finden ist. Deshalb sind die beiden folgenden Datensätze von Jira und Stack Overflow. Der Datensatz von Jira bietet einen Umfang von 500 Aussagen verschiedener Entwickelnden und wurde von Novielli et al. [24] bereits evaluiert. Der zweite Datensatz mit 4000 Einträgen wurde von Ortu et al. [18] evaluiert. Eine Einschränkung bei dem ersten Datensatz ist die Anzahl der Einträge.

In den zuvor genannten Papern wurden Entwickelnden für eine Analyse der Aussagen befragt. 4000 Einträge zu klassifizieren, nimmt trotz mehrerer Entwickelnden, viel Zeit ein, weswegen ein größerer Datensatz nur mit Hilfe von automatisierten Tools auswertbar wäre.

4.1 JIRA Datensatz

In dem zuvor genannten Paper von Novielli et al. [24] wurde ein Datensatz für die Auswertung verschiedener Sentimentanalyse Tools verwendet. Dieser Datensatz wurde von unterschiedlichen Entwickelnden nach verschiedenen emotionalen Wahrnehmung klassifiziert: fear (-), anger (-), sadness (-), joy (+), love (+) und suprise(+). Hierbei sind die positiven Emotionen mit (+) und negative Emotionen (-) gekennzeichnet. Diese Emotionen und ihre Klassifikation gehen aus dem Paper von Lazarus et al. [17] hervor. Anschließend wurden die Aussagen zusammengefasst, da die Aussagen verschiedene Emotionen (sowohl negativ als auch positiv) beinhalten können. Hierzu wurde die Anzahl der positiven Emotionen `pos()` und die Anzahl der negativen Emotionen `neg()` einer Aussage zusammengefasst:

$$f(Aussage) = \#pos(Aussage) - \#neg(Aussage)$$

Entstand ein Wert von 0, so ist diese Aussage mit neutraler Stimmung bewertet. In der ersten Auswertung gab es es zwei Aussagen, die einen Wert von (+2) und eine mit (-2) aufwiesen. Dies führt zu einer ungleichen Verteilung der Daten, da sie Ausreißer sind und somit wurden die Werte jeweils auf die einfachen Stimmungen (-1) und (+1) angepasst. Daraus ergibt sich für die Einteilung der Aussagen in die verschiedenen Stimmungen folgende Tabelle:

	Positiv	Neutral	Negativ
#	166 (33,2 %)	303 (60,6%)	31 (6,2%)

Tabelle 4.1: Die Tabelle zeigt die Anzahl der verschiedenen Stimmungen (positiv, negativ und neutral) bei dem Jira Datensatz.

Aus der Abbildung ist zu entnehmen, dass aus den insgesamt 500 Aussagen die meisten neutral sind und nur sehr wenige negativ. Die Daten sind weiterhin unausgeglichen (im Bezug auf positive und negative Stimmung) und die negative Stimmung bildet ein nur sehr kleinen Wert für eine ausführliche Analyse. Daher wurde ein weiterer größerer Datensatz gesucht.

4.2 Stack Overflow Datensatz

Der Datensatz von Stack Overflow [1] wurde in dem Paper von Ortu et al. [18] genauer analysiert. Hierbei wurden zum Trainieren insgesamt 4000 Einträge erstellt. Diese Einträge wurden in vier verschiedene Emotionen eingeteilt: anger(-), sadness (-), joy(+), love(+). Wie auch in dem vorherigen Datensatz wurden die Emotionen zusammengefügt und nach dem Ranking von Novielli et al. klassifiziert. Daraus ergibt sich für diesen Datensatz folgende Einträge:

	Positiv	Neutral	Negativ
#	371 (9,35 %)	2935 (74,0%)	660 (16,64%)

Tabelle 4.2: Die Tabelle zeigt die Anzahl der verschiedenen Stimmungen (positiv, negativ und neutral) bei dem Stack Overflow Datensatz.

Bei diesem Datensatz ist zwar die relative Anzahl der neutralen Aussagen deutlich höher (+13,4%) im Vergleich zum Jira Datensatz, aber der Datensatz besitzt dennoch weitaus mehr Einträge. Dies führt zu dem Ergebnis, dass dieser Datensatz für eine Analyse besser geeignet ist, da genügend Validierungsdaten für jede Stimmung bereitstehen. Damit Fehlverhalten und ungleiche Verteilungen der Daten nicht eintreffen, wurden aus dem Datensatz einige Daten entfernt. Hierzu zählen Jira Ticketnummern, Code-Ausschnitte und künstlich durch Leerzeichen gestreckte Kommentare. Diese wurden bewusst entfernt, da sie eine ungleiche Verteilung und Verfälschung der Daten bewirkt (Vergleiche dazu das Paper von Ortu et al. [18]). Insgesamt überwiegen die negativen Aussagen gegenüber der positiven aus dem Datensatz.

4.3 Einfache statistische Metriken

Dieses Unterkapitel befasst sich mit der Evaluation von einzelnen Metriken. Wie zuvor bereits diskutiert, ist der Datensatz von Jira schlechter geeignet für eine Evaluation von Metriken, da es lediglich 31 negative Einträge gibt. Deswegen wurde die folgende Analyse nur auf dem Datensatz von Stack Overflow durchgeführt.

[1] http://ansymore.uantwerpen.be/system/files/uploads/artefacts/alessandro/MSR16/archive3.zip

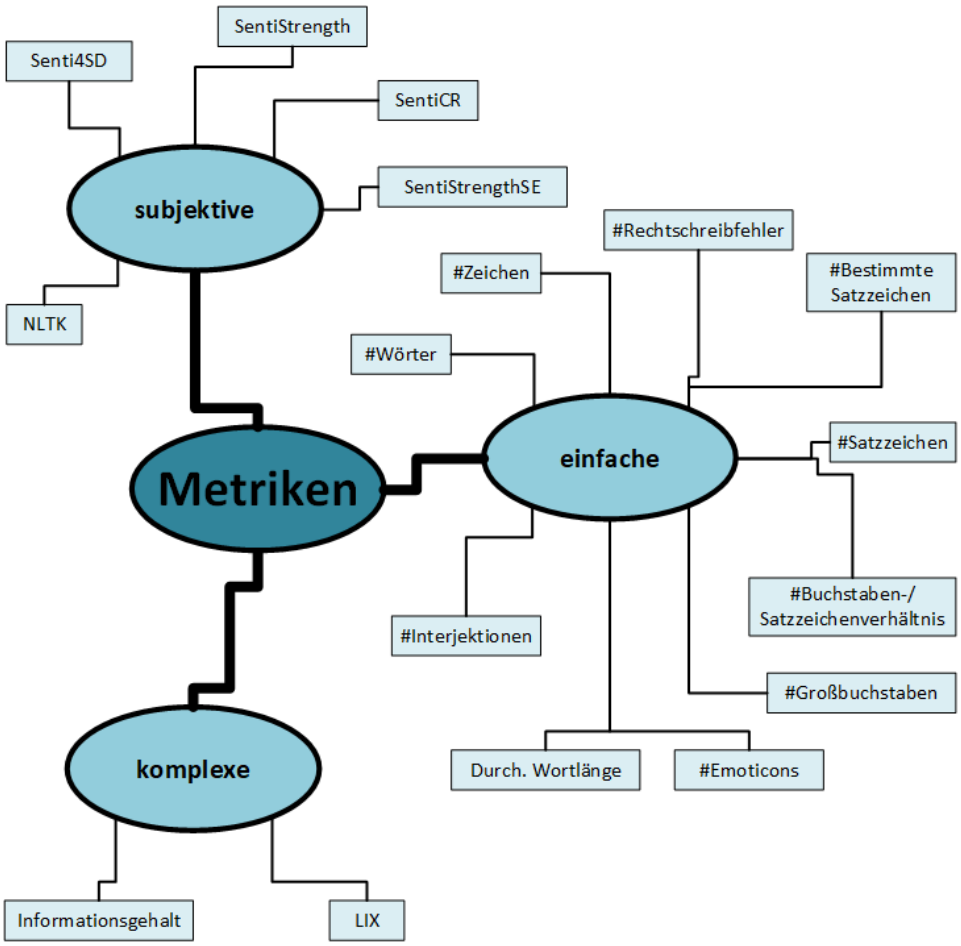

Abbildung 4.1: Die Abbildung zeigt alle evaluierten Metriken dieser Masterarbeit.

Die folgenden Metriken wurden mit Hilfe der Masterarbeit von Julian Horstmann [9] erstellt. In seiner Masterarbeit wurden viele Metriken bereits ausgewertet und im kommenden Abschnitt weiter verwendet. Die komplexeren Metriken „Informationsgehalt“ und „Satzkomplexität “ wurden aus einem vergangenen Projekt der europäischen Union entnommen bzw. abgeleitet [30]. Dieses Projekt beschäftigte sich mit der Analyse von von Risiken und Möglichkeiten für wirtschaftliche Gemeinschaften. Insgesamt wurden nur die folgenden Metriken auf dem Datensatz evaluiert. Einige weitere gefundene Metriken wurden in diesem Kapitel ausgelassen, da sie eine inhaltliche Analyse der Sätze durchführen, als Beispiel dafür: SentiStrength.

Die Metriken wurden in diesem Kapitel rein rechnerisch ausge-

wertet. Zu der Auswertung gehört zu jeder Metrik das Zählen der zutreffenden Inhalte. Ein Beispiel dafür ist: *„Thank you for reviewing.“*. Dieser Satz beinhaltet insgesamt 24 Zeichen, 4 Wörter und ein Satzzeichen. Diese Werte wurden für jede Aussage des Datensatzes ausgerechnet und daraus wurden die folgenden Boxplots erstellt und interpretiert. Ein Signifikanztest der Variablen findet im nächsten Kapitel statt.

4.3.1 Anzahl von Zeichen

Diese statistischen Metrik behandelt die Anzahl der Zeichen innerhalb einer Nachricht. Aus der dieser Metrik resultiert für die verschiedenen Stimmungen folgender Graph:

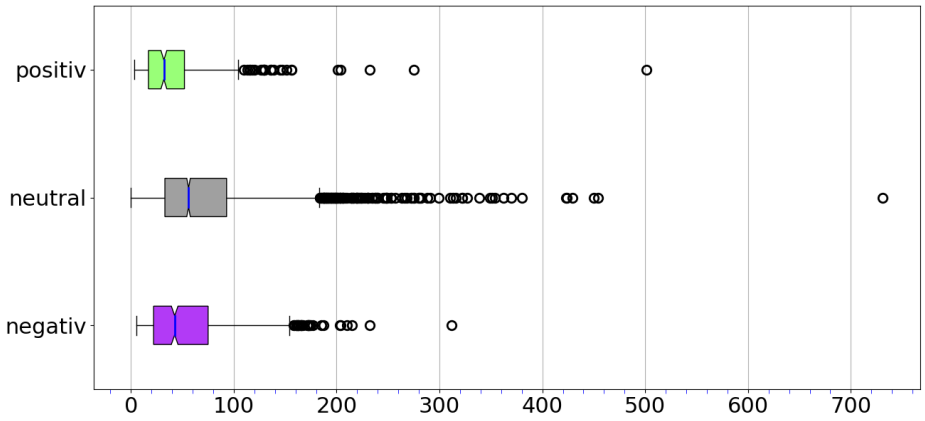

Abbildung 4.2: Der Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der Zeichenanzahl.

Aus dem Boxplot ist zu entnehmen, dass es zwischen negativ und positiv einige Unterschiede gibt. Wird zusätzlich die Standartabweichung für positiv (148.1) und negativ (185.8) sowie das Minimum: 9 (positiv)/ 13 (negativ) und Maximum: 752 (positiv)/ 640 (negativ) berechnet, so ist zu erkennen, dass die Streuung sehr hoch ist. Außerdem existiert eine relevante Menge, die eine Überschneidung mit der neutralen Stimmung besitzt und zu falschen Klassifikationen führen kann. Werden die neutralen Aussagen entfernt, so besitzen die negativen Aussagen etwas mehr Zeichen als die positiven Aussagen.

Die Metrik wurde berechnet durch das Zählen jedes einzelnen Zeichens einer Aussage.

4.3.2 Anzahl von Wörtern

Bei dieser Metrik wurde für die einzelnen Aussagen die durchschnittliche Anzahl von Wörtern berechnet.

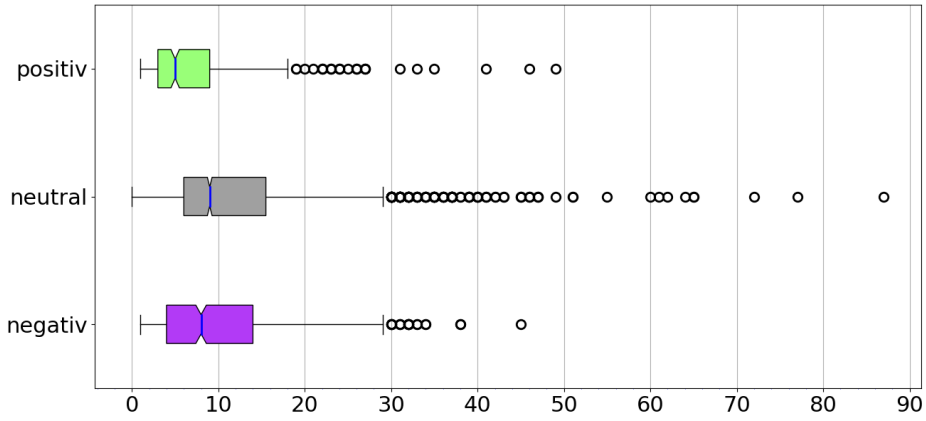

Abbildung 4.3: Der Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der durchschnittliche Wortzahl.

Das Boxplot zeigt auf, dass die Anzahl der Wörter etwas größer bei den negativen Kommentaren ist als bei den positiven. Für die positiven Kommentare ergibt sich ein Durchschnitt von 7,43 Wörtern und für die negativen Kommentare 9,86 Wörter. Dennoch verhält sich diese Metrik ähnlich zu der vorherigen Metrik, da die Standartabweichnung auch eine höhere Streuung der Daten aufweist (Positiv: 6,87 Wörter, Negativ: 7,76 Wörter). Die Anzahl der Wörter wurde mit Hilfe der `split()` Methode aus Python berechnet. Diese Methode teilt eine Zeichenkette in einzelne Wörter auf. Daraus wurde für jede Aussage die Anzahl der Wörter ermittelt und für jede Stimmung ein Durchschnitt gebildet.

4.3.3 Durchschnittliche Wortlänge

Bei der durchschnittlichen Wortlänge gibt es wie bei der durchschnittlichen Zeichenanzahl nur wenige Unterschiede zwischen den positiven sowie negativen Stimmungen.

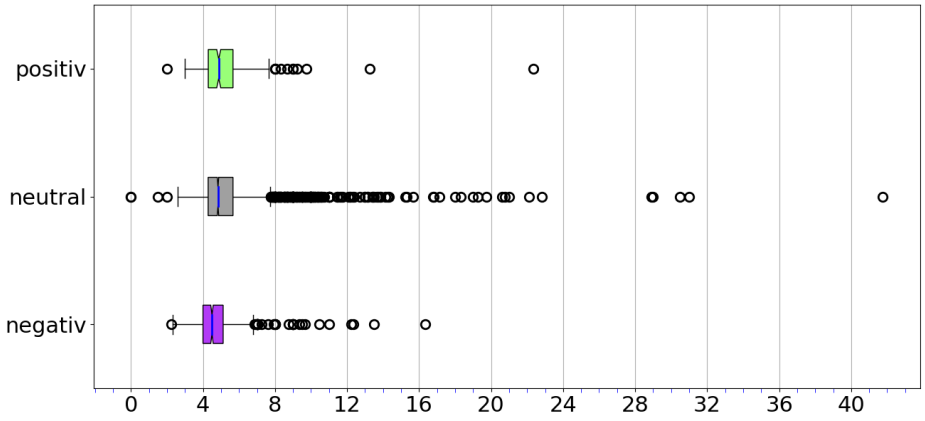

Abbildung 4.4: Das Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der durchschnittliche Wortlänge.

Die durchschnittliche Wortlänge weist mit 5,11 (positiv) und 4.71 (negativ) Wörtern ein nur sehr geringen Unterschied zwischen den Stimmungen auf. Diese Metrik korreliert mit der vorherigen Metrik und bewirkt nur einen geringen Unterschiede zwischen den beiden Stimmungen.

4.3.4 Anzahl von Satzzeichen

In vielen Foreneinträgen wie auch bei Nachrichten in sozialen Netzwerken kommt es häufig vor, dass eine Person, die negative Nachrichten verfasst, einen übermäßigen Gebrauch an Satzzeichen hat. Die häufigste Form dabei ist „!!!“. Mehrere Fragezeichen können auch eine Wirkung der Verdeutlichung des Standpunktes erzeugen [5].

Wie auch bei den vorherigen Metriken ist in dem Boxplot 4.5 zu erkennen, dass es nur minimale Unterschiede zwischen positiven und negativen Stimmungen (Positiv: 2,58 Wörter, Negativ: 3,00) gibt. Diese Unterschiede reichen jedoch nicht aus, um einen relevanten Unterschied zu erkennen. Einige Textbeispiele mit „!!!“ waren positiv, aber auch negativ. Als Hypothese wurde aufgestellt, dass generell mehrere Satzzeichen als Folge eine negative Stimmung äußern. Einige Beispiel widerlegen dies jedoch.

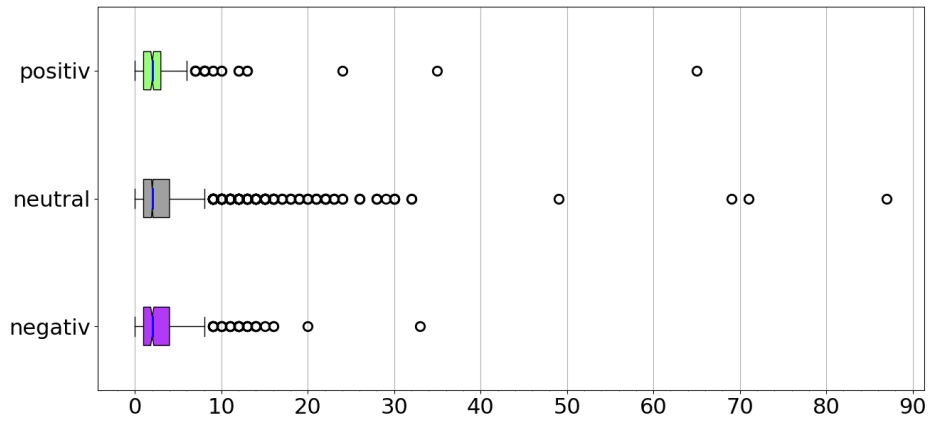

Abbildung 4.5: Der Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der Satzzeichenanzahl.

4.3.5 Bestimmte Satzeichen

Statt wie zuvor nach generellen Satzzeichen zu suchen, war die nächste Idee bestimmte Satzzeichenkombinationen, wie „ !!! “, „ . . . “ oder „ ??? “, zu suchen. Das Vorkommen von solchen Satzzeichenkombinationen ist bei beiden Datensätzen sehr spärlich („ !!! “ : 12, „ . . . “ :125, „ ??? “ : 3), weswegen diese Metrik vorerst vernachlässigt wird.

4.3.6 Buchstaben/Satzzeichen Verhältnis

Diese kombinierte Metrik erzeugt ähnliche Ergebnisse wie bei der generellen Anzahl von Buchstaben. Eine genauere Analyse wurde daher in dieser Arbeit nicht vorgenommen (vergleiche hierzu [9]).

4.3.7 Anzahl von Großbuchstaben

Häufig existieren in Artikeln, aber auch bei Plattformen wie Stack Overflow, Wörter sowie ganze Sätze, die komplett groß geschrieben werden, um damit eine negative Stimmung deutlich zu machen. Häufig werden groß geschriebene Wörter oder Texte mit schreien verbunden [5] und wirken auf den Lesenden negativ. Somit wäre diese Metrik ein Indikator, um Stimmungen zu erkennen. In den beiden Datensätzen befinden sich nur eine spärliche Anzahl dieser Wörter oder Sätze, wodurch sich diese Metrik nicht für eine weitere Analyse eignet.

Eine Studie [5] hatte ergeben, dass häufig negative Kommentare in sozialen Netzwerken auf Großbuchstaben zurückgreifen. Daher wäre eine Stimmungsanalyse mit dieser Metrik eher im Bereich sozialer Netzwerke zu empfehlen.

4.3.8 Rechtschreibfehler

Eine weitere Metrik bilden Rechtschreibfehler. In diesem Abschnitt wird lediglich falsche Rechtschreibung betrachtet. Eine Hypothese ist, dass Personen, die launisch sind, häufiger Rechtschreibfehler machen, sofern sie keine Rechtschreibüberprüfung nutzen.

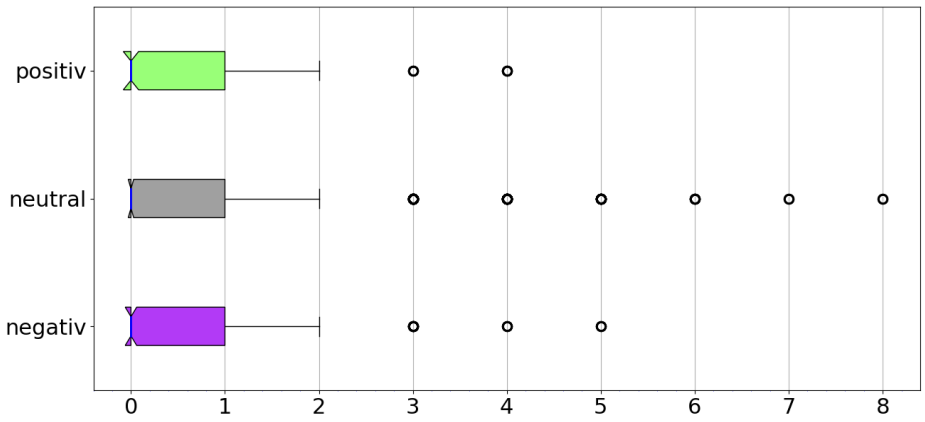

Abbildung 4.6: Der Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der Rechtschreibfehler.

Anhand der Grafik ist zu erkennen, dass weniger Fehler bei negativen Kommentaren anfallen. Bei der Analyse des Datensatzes fällt auf, dass die überwiegende Mehrheit der markierten Rechtschreibfehler Namen von Frameworks oder Tools sind bzw. Sonderzeichen, die fälschlicherweise einfügt wurden. Auch andere Abkürzungen werden als Fehler gekennzeichnet. Werden diese Wörter entfernt, dann ist die Summe der Resultate sehr gering und nicht aussagekräftig. Somit konnte die vorherige Hypothese vorerst widerlegt werden. Ein anderer Datensatz oder das Verbieten von Rechtschreibüberprüfungen könnte ein anderes Ergebnis liefern.

4.3.9 Interjektionen

Neben den bestimmten Satzzeichen gibt es auch die Möglichkeit nach Interjektionen zu suchen. Diese zeigen bestimmte Emotionen sehr gut auf. Beispiele hierfür sind „Wow!“ (+), „Great!“ (+) oder „No!“ (-). Bei diesen Beispielen lässt sich eine Polarität feststellen [2]. In den beiden Datensätzen existieren wie ebenfalls in einigen Metriken zuvor nur wenige Einträge mit solchen Ausrufen. Daher ist für diesen Datensatz eine Validierung der Metrik nicht durchführbar. Aber nach dem Paper von Balnat et al. [2] gibt es für die verschiedenen Interjektionen Emotionen, die den Wörtern zugeordnet werden können.

4.3.10 Emoticons

Emoticons sind bei textuellen Nachrichten in vielen sozialen Netzwerken wie auch in privaten Kurznachrichtendiensten ein Standard, um einfach und schnell Emotionen widerzuspiegeln. Beispiele für die Emoticons sind: „ :-) “ (+) oder „ :-(“ (-). Es hat sich in dem Datensatz gezeigt, dass diese häufiger vorkommen als bestimmte Satzzeichen oder Interjektionen. Zudem lassen sich Emotionen besser daraus ableiten.

Wird eine Kombination dieser Metrik und der Metrik mit den bestimmten Satzzeichen erstellt, so ist die Abdeckung innerhalb des Datensatzes etwas besser. Aus dieser Metrik wurde daher eine kombinierte Metrik bestehend aus Emoticons und Satzzeichen entwickelt. Für diese Metrik wurde in Python eine Liste mit allen vorkommenden Zeichen und Emoticons erstellt und anschließend nach ihrer Stimmung bewertet.

4.4 Komplexe statische Metriken

Neben den einfachen statistischen Metriken gibt es komplexere Metriken. Diese sind in der Berechnung aufwendiger und es werden zusätzliche Frameworks benötigt. Als Framework wurde NLTK (siehe Kapitel 2.1.5) verwendet. NLTK ist in der Lage jedes Wort einer Wortart der Part-of-Speech zuzuordnen.

4.4.1 Informationsgehalt

Der Informationsgehalt von Nachrichten wird in der Nachrichtentechnik durch die Entropie kodiert. Für die hier benutzte Metrik wurde eine abgewandelte Formel verwendet:

$$Informationsgehalt = \frac{\sum Content\ Wörter}{\sum Funktionale\ Wörter} \tag{4.1}$$

Zu den Content-Wörtern zählen folgende Wortarten: Verben, Adjektive und Nomen. Hierbei handelt es sich um Wörter, die einen gewissen Informationsgehalt vermitteln. Zum Beispiel:

„He doesn't like C#."

hat einen geringeren Informationsgehalt als

„Jim doesn't like C#."

Jim ist eine bestimmte Person und bietet weniger Interpretationsspielraum. Dadurch vermittelt dieser Satz einen höheren Informationsgehalt. Neben den Content-Wörtern gibt es die funktionale Wörter, die keine Informationen bzw weitere Details zum Inhalt des Satzes vermitteln. Hierzu zählen zum Beispiel Pronomen und Konjunktionen. Aus dieser Metrik ist es möglich abzuleiten, wie objektiv oder subjektiv ein Satz interpretierbar ist. Je mehr Informationen ein Satz vermittelt (>Informationsgehalt) desto objektiver, und weniger durch eigene Interpretation, lässt sich ein Satz analysieren.

Diese Metrik liefert als Ergebnis, dass der durchschnittliche Informationsgehalt bei negativer Stimmung bei 1,16 und bei positiver Stimmung 1,29 liegt. Ist der Wert über 1, so überwiegt die Anzahl der Content-Wörter und der Satz wirkt informativer.

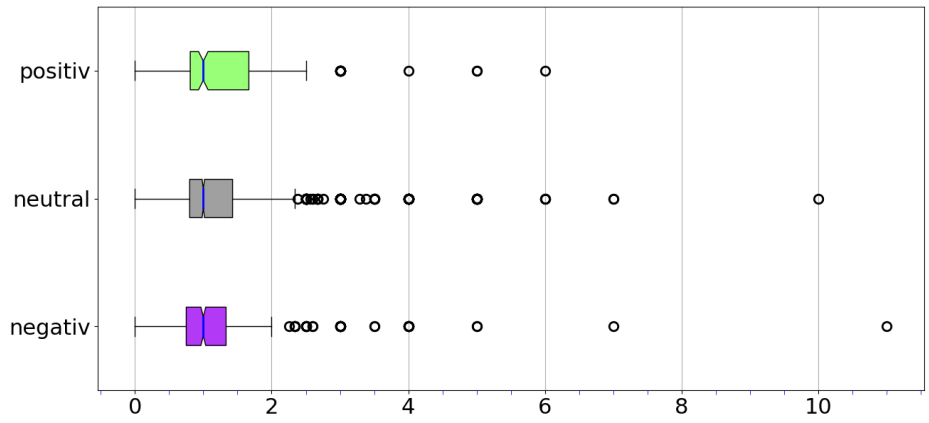

Abbildung 4.7: Der Boxplot zeigt die drei Stimmungen im Vergleich auf Grundlage der Informationsgehalt.

Aus dem Boxplot ist zu erkennen, dass es einige Ausreißer gibt. Diese Ausreißer sind dadurch entstanden, dass einige Sätze grammatikalische Fehler aufweisen oder falsch von NLTK analysiert wurden. Einer der Ausreißer ist „sorry, last patch was missing new files“. Bei diesem Satz sind nur Content-Wörter enthalten bis auf das Komma. In diesem Satz würde für den Lesenden zum einen ein Punkt am Ende fehlen und für den Kontext ein „the“ vor dem last. Dadurch würde sich der Informationsgehalt von 7,0 auf 2,33 verringern. Für den Ausreißer mit einem Informationsgehalt von 11 ergibt sich das Problem, dass dieser Satz einen Tag besitzt, der nicht nach dem üblichen Schema aufgebaut ist und somit nicht entfernt wurde. Dieser Tag besteht aus mehreren Sonderzeichen, die als einzelne Wörter erkannt und als Content-Wörter klassifiziert wurden. Dennoch ist zu erkennen, dass es trotz Ausreißern Unterschiede zwischen positiven, negativen und auch neutralen Stimmungen gibt.

Diese Metrik wurde mit Hilfe von NLTK berechnet. Die Sätze wurden in Token aufgeteilt, sodass jedes Wort klassifiziert werden kann. Für jedes Wort wurde dann die Wortart bestimmt und die Summen der funktionalen und Content-Wörtern gebildet. Dann wurde die Formel 4.4.1 angewendet.

4.4.2 LIX

Diese Metrik repräsentiert die Komplexität von Texten. Sie gibt außerdem an, wie einfach oder Komplex sich Texte lesen lassen:

$$LIX = \frac{\#W\ddot{o}rter}{\#S\ddot{a}tze} + \frac{\phi\ Lange\ W\ddot{o}rter\ *\ 100}{Gesamtw\ddot{o}rter}$$

wobei ϕ dem Durchschnitt entspricht. Bei dieser Metrik wird überprüft, wie schwer oder einfach sich Texte lesen lassen. Der Vergleich wurde hierbei gezogen, ob es sich um einen Text für Kinder handelt (untere Schranke) oder um einen wissenschaftlichen Artikel (obere Schranke). Diese Metrik soll in diesem Kontext herausfinden, ob es einen Unterschied bei der Komplexität zwischen positiver oder negativer Stimmung gibt. Das Problem und der Grund, warum diese Metrik nicht weiter analysiert wird, ist, dass sich die Formel auf ganze Texte und nicht auf einzelne Sätze konzentriert. Der Datensatz besitzt kaum Kommentare mit mehr als einen Satz, wodurch eine Analyse dieser Metrik nicht durchführbar bzw. sinnvoll ist.

Kapitel 5

Logistische Regressionsanalyse

Die im Kapitel 4.3 angewendete statistische Auswertung hat einen Überblick über die Struktur der Aussagen des Datensatzes gebracht und aufgezeigt welche Metriken generell für eine Analyse auf dem Datensatz geeignet sind. In diesem Kapitel soll nun eine logistisches Regressionsanalyse durchgeführt werden. Diese logistische Regressionsanalyse gibt an, wie ein Modell trainiert ist und gleichzeitig liefert sie die Antwort auf die Signifikanz der verschiedenen Metriken. Im vorherigen Kapitel wurden positive, neutral und negative Stimmungen analysiert und verglichen. Für die logistische Regressionsanalyse eignen sich nur zwei verschiedene Stimmungen im Vergleich, andernfalls kann es zu Problemen bei der Bestimmung und Zugehörigkeit der Aussagen kommen kann, da das Modell nur auf Werte zwischen 0 und 1 abbildet. Hierfür wäre es notwendig eine Skala einzuführen, die die drei Stimmungen präsentiert. Generell soll in diesem Kapitel auch geprüft werden welchen Effekt die verschiedenen Metriken auf positive und negative Stimmungen haben.

5.1 Metriken in der Analyse

Für die Analyse der positiven und negativen Affekte wurden aus dem Datensatz von Stack Overflow die neutralen Kommentare ausgelassen, sodass insgesamt 1031 Kommentare (positiv: 660, negativ: 371) für die Analyse übrig bleiben. Die positive Stimmung wurde dem Label 1 und negative Stimmung dem Label 0 zugeordnet (vergleiche Abbildung 5.1). Auf der X Achse wurden die Einflüsse der Variablen des trainierten Modells gekennzeichnet.

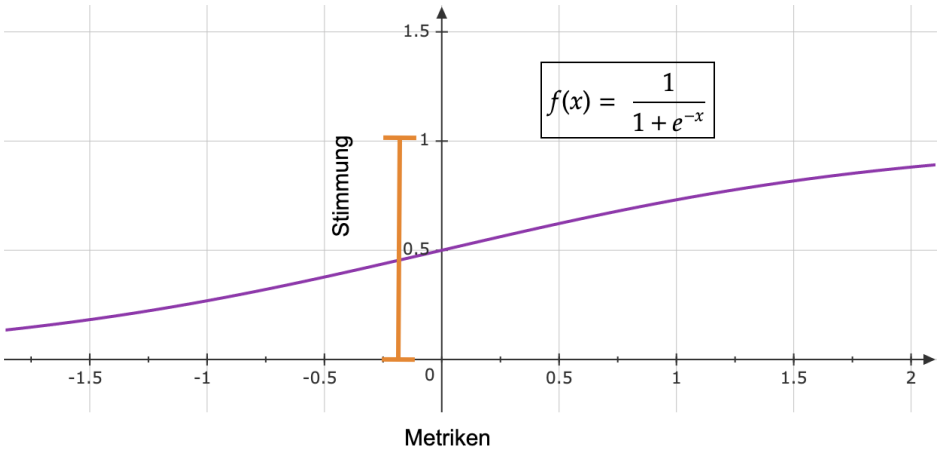

Abbildung 5.1: Diese Abbildung zeigt die logistische Regressionanalyse mit dem Stimmungslabel und den Metriken. Orange markiert hierbei Stimmung zwischen 0 und 1. Lila ist der Verlauf der Sigmoid-Funktion.

Für die weiterführende Analyse wurde das logistische Regressionsmodell mit den folgenden Variablen trainiert:

- Anzahl von Zeichen
- Anzahl von Wörtern
- Anzahl von Satzzeichen
- Anzahl Emoticons in Kombination mit Satzzeichen wie „ !!! “
- Informationsgehalt

Diese Variablen ergeben sich aus der Analyse aus dem vorherigen Kapitel. Die anderen Variablen wurden bewusst nicht für den Signifikanztest ausgewählt, da die Metriken kaum Inhalt darbieten oder, dass sie bereits durch andere Metriken abgedeckt sind, weil diese miteinander korrelieren. Ein Beispiel für die Korrelation von Metriken, sind Anzahl der Wörter und Anzahl der Zeichen einer Aussage.

Als Beispiel, wie die Daten in das logistische Regressionsmodell überführt werden, dient folgendes Beispiel aus dem Datensatz:

„Got it, thanks for the explanation. “

Für die Metriken ergeben sich die folgenden Werte:

Metrik	Werte
#Zeichen	36
#Wörtern	6
#Satzzeichen	2 (Ohne Leerzeichen)
#Emoticons	0 (Kein Vorkommen)
Informationsgehalt	0,6
Klassifiziertes Label	1 (Positiv)

Tabelle 5.1: Die Tabelle zeigt eine beispielhafte Eingabe der Daten für die jeweiligen Metriken für eine Klassifikation/Regression

Nach diesem Schema wurden die Werte von jeder Aussage aus dem Datensatz zusammengefasst und in das logistische Regressionsmodell überführt. Die Werte wurden zuvor noch normiert, sodass sie zwischen 0 und 1 liegen. Für das Training ergibt sich ein mehrdimensionales Array mit fünf Spalten und insgesamt 1031 Zeilen. Zu diesem mehrdimensionalen Array gehört noch ein Array mit einer Spalte und 1031 Zeilen, welche die Labels der Entwickelnden repräsentiert. Mit Hilfe der `train_test_split()` Methode aus dem Python Framework `sklearn` ist es möglich einen randomisierten Test- und Trainingsdatensatz zu erstellen. Der Traininsdatensatz wurde zum Trainieren des logistischen Regressionsmodell verwendet. Das erste Resultat wird in der Grafik 5.2 abgebildet. Die Grafik zeigt eine Confusion Matrix, welche aus den Testdatensatz und dem trainierten Modell besteht.

Aus der Abbildung 5.2 ist zu erkennen, dass das Genauigkeitsmaß bei dem trainierten Modell bei 65,21% liegt. Bei einer genauen Analyse der Verteilung fällt auf, dass 33,33% der Daten als negativ klassifiziert wurden, trotz dessen, dass sie positiv sind. Das Modell sagt eine negative Stimmung voraus, da die Wahrscheinlichkeit richtig zu liegen am höchsten ist. Dies liegt zum einen daran, dass der Datensatz überwiegend negative Einträge besitzt und die Regression somit zum Trainieren überwiegend die negative Trainingsdaten nutzt. Die Log-Likelihood für dieses Modell liegt bei -256,73 und lässt auf eine Verschiebung der Vorhersage in negativer Richtung schließen. Der F1 Wert liegt hierbei bei der positiven Vorhersage bei nur 10,00%. Für das negative Label hingegen ist es mit 78,44% deutlich besser. Wird zusätzlich der LR-Test mit mit dem Nullmodel evaluiert, so fällt auf, dass das trainierte Modell etwa die gleiche Performanz besitzt. Dies führt zu der Annahme, dass dieses Modell nur sehr limitiert

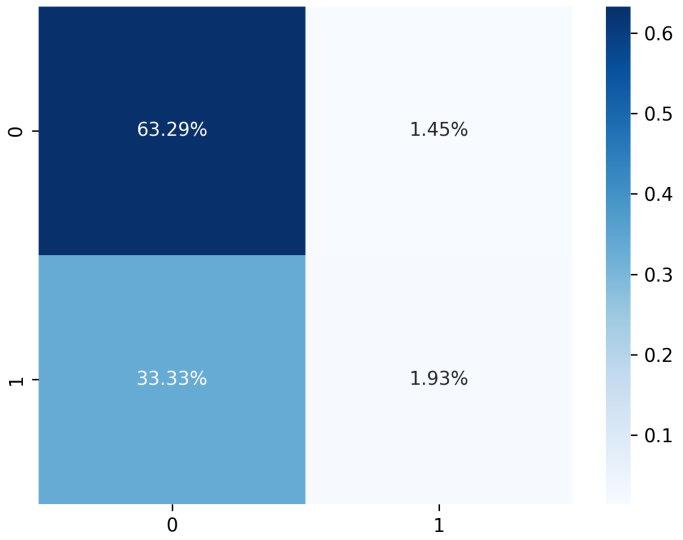

Abbildung 5.2: Die Confusionmatrix zeigt das Ergebnis der logistischen Regressionsanalyse. Hierbei wurden die Werte $< 0{,}5$ als negativ und alle Werte $\geq 0{,}5$ als positiv gewertet.

trainiert ist und keine besseren Werte als ein randomisiertes Modell generiert. Zum anderen liegt es auch daran, dass es nicht möglich ist die beiden Stimmungen zu unterscheiden, denn bei einem Vergleich mit der Anzahl der Einträge fällt auf, dass die positive und negative Stimmung etwa gleich viele Einträge besitzt. Aus diesem Grund wurden die Daten ausgeglichen, um eine bessere Verteilung zu generieren. Abbildung 5.3 zeigt zwar eine geringere Verbesserung der Werte, aber dennoch genügt dieses Ergebnis nicht aus, um genauere Vorhersagen durchzuführen. Werden die p-Werte der einzelnen Metriken verglichen und führt einen Signifikanztest durch, so ist das Resultat, dass die Anzahl der Satzzeichen (28,0%) und der Informationsgehalt (17,7%) nicht signifikant sind. Somit kommt diese Analyse zu dem Resultat, dass eine genauere Analyse der Metriken mit Hilfe des Modells nicht möglich ist.

Als Alternative zu dem logistischen Regressionsmodell bieten sich aktuelle Methoden aus dem maschinellen Lernen an. Als sehr populäres und robustes Modell [13] gilt die Support Vector Machine (siehe Kapitel 2.2.2). Für das Training wurde hierbei ein Trainingsset (80%) sowie ein Testset (20%) erstellt wie auch bei dem logistische Regressionsmodell.

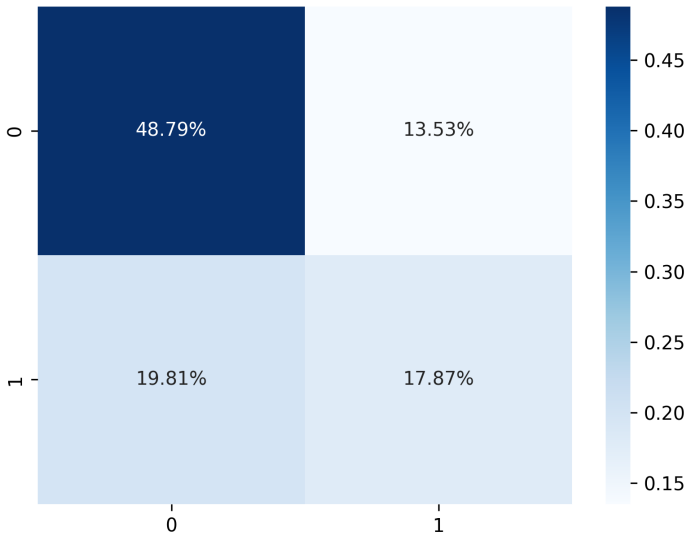

Abbildung 5.3: Die Confusionmatrix zeigt das Ergebnis der logistischen Regressionsanalyse mit gleich vielen positiven wie negativen Einträgen.

Weiterhin verbleiben die negativen Kommmentare bei dem Label 0 und positive Kommentare bei 1. Das Ergebnis für dieses trainierte Modell ist etwas besser als bei der logistischen Regressionsanalyse: Das Genauigkeitsmaß beträgt zwar 69,56%, aber die Confusion Matrix verhält sich ähnlich zu der Confusion Matrix mit ausgeglichenen Daten (Vergleiche Abbildung 5.4 und 5.3). Bei der SVM liegt die Präzision (Precision) bei 65,62% für das Label negativ (0) und bei der positiven Stimmung (1) bei 70,28%. Diese berechneten Werte sind für die Labels zwar ausreichend, aber bei einem Vergleich des Rückrufs (Recalls) der Label ist auffällig, dass dieser Wert bei der positiven Stimmung lediglich 28,76% ist. Bei einer Kombination zum F1-Wert liegt der Wert bei dem negativen Label bei 79,61% und bei positiven Label lediglich bei 40,00%. Dies zeigt auf, dass die Klasse der positiven Stimmungen nur schwer von der den negativen Aussagen differenzierbar ist. Eine Veränderung andere Parameter, wie dem ζ, führt zu keiner Verbesserung dieser Werte.

Ein weiteres Modell, welches andere Trainingsmethoden nutzt, ist der Decision Tree (siehe 2.2.4). Dieses trainierte Modell konnte ein Genauigkeitsmaß von etwa 64% erreichen und die Verteilung der Daten war zwar besser als in den vorherigen Modellen, hat aber für das

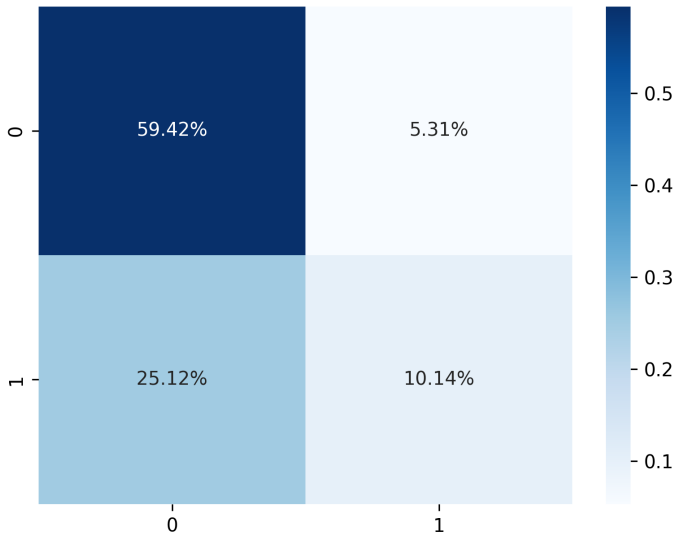

Abbildung 5.4: Die Confusionmatrix zeigt das Ergebnis der Support Vector Machine mit allen Metriken aus dem Anfang des Kapitels.

positive Label nur einen Recall von 49,31%. Bei einem zusätzlichen Vergleich des F1-Wertes der beiden Klassen (positiv: 67,95%, negativ: 46,45%) ist auffällig, dass die Labels nicht bzw. kaum differenzierbar sind. Ein weiteres Resultat ist, dass das Modell overfitted ist, da die Tiefe des Entscheidungsbaums 26 beträgt. Hierbei existieren aber lediglich fünf Eingabeparameter. Ein erweitertes Modell stellt der Random Forest Classifier (siehe 2.2.5) dar. Dieser führte zwar zu einem insgesamt ausgeglicheneren Modell (Abbildung 5.5 mit einem F1-Wert (positiv: 51,06%, negativ: 74,72%) als die SVM oder der logistische Regressionsanalyse, aber dennoch ist der F1-Wert nicht ausreichend hoch genug, um eine genauere Analyse durchzuführen.

Keines der trainierten Modelle konnte einen für eine genauere Analyse ausreichend guten F1-Wert erzielen. Aufgrund dessen wurde als letzte Möglichkeit AutoSklearn (2.2.6) verwendet. AutoSklearn bietet im Gegensatz zu einzelnen Maschinenmodellen eine Zusammenschluss von mehreren Maschinenmodellen. Dies soll dazu führen, dass eine bessere Performanz durch Hyperparamerteranpassung generiert wird. Mit Hilfe dieses Modells ist es möglich besser vorherzusagen, ob eine Klassifikation der Daten generell funktionieren würde. Das Ergebnis aus dem AutoSklearn Classifier ist ein Genauigkeitsmaß von 69,08%

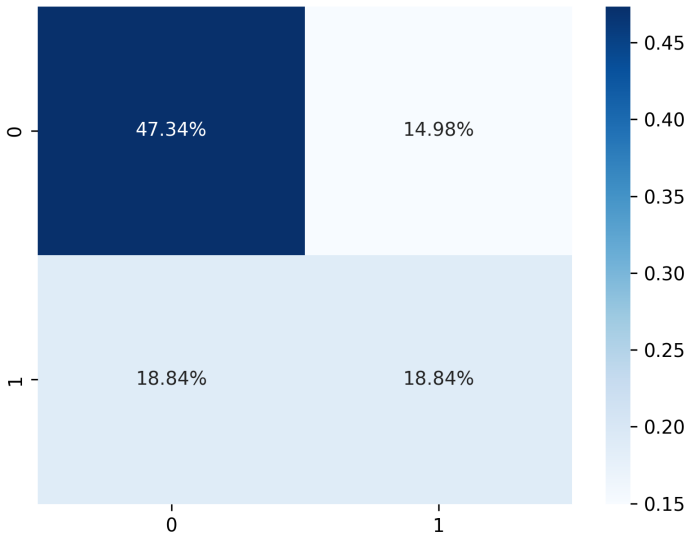

Abbildung 5.5: Die Confusionmatrix zeigt das Ergebnis der Random Forest Classifiers.

und liegt damit fast gleichauf mit der SVM. Der Recall liegt hierbei für die positiven Klasse auch bei 10,00% und bei der negativen Klasse 78,44%. Damit wurde gezeigt, dass keines der trainierten Modelle einen direkten Rückschluss auf die Polarität der Kommentare aufzeigen kann. Somit ist es nicht möglich herauszufinden, welche Metrik einen Einfluss und vor allem welchen Einfluss auf die Stimmung eines Entwickelnden haben. Aus diesen Resultaten folgt die Erkenntnis:

Erkenntnis 1

Es ist nicht möglich mit Hilfe der Metriken subjektive Stimmungen vorherzusagen.

In den Papern von Jongeling et al. [14], Murgia et al. [20] und Nasif et al. [11] wurde mehrfach diskutiert, dass selbst verschiedene Tools wie SentiStrength (2.1.1), SentiCR (2.1.4), SentiStrengthSE (2.1.2) und NLTK (2.1.5) keine eindeutigen Ergebnisse erzeugen. Bei dem Vergleich der einzelnen Tools von Jongeling et al. [14] wurde mit Hilfe von Cohens Kappa gezeigt, dass es lediglich eine Übereinstimmung der klassifizierten Werte von etwa 22% im Durchschnitt gab.

In diesem Kapitel wurden zu Beginn potentielle Metriken, die für die Datensätze in Frage kommen, mit Hilfe der logistischen Regressionsanalyse, ausgewertet. Es zeigte sich zwar, dass einige Metriken signifikanter waren als andere, aber dennoch ist keine genaue Interpretation möglich. Die Modelle waren mit Hilfe der Metriken nur sehr eingeschränkt trainiert. Das Entfernen einiger Daten, um damit einen ausgeglichenen Datensatz zu erhalten, führte auch zu keinem wesentlich besseren Ergebnis. Einige Metriken wurden im Kontext dieses Datensatzes nur theoretisch ausgewertet, wie zum Beispiel LIX oder Interjektionen. Diese beiden Metriken bieten jedoch eventuell bessere Ergebnisse. Aufgrund der Limitierung der Datensätze wurde diese Aussage nicht geprüft.

5.2 Neutrale Daten

Am Anfang dieses Kapitels wurde erwähnt, dass die neutral klassifizierten Kommentare aus der logistischen Regressionsanalyse und Evaluation entfernt wurden. Dies liegt daran, dass der überwiegende Teil der Daten von Entwickelnden neutral klassifiziert wurde und sich eine Ungleichheit bildet. Bei einer Beschränkung der Daten und dem damit folgenden Ausgleich der drei Stimmungen ist wiederholt auffällig, dass die dennoch neutrale Stimmung beim Training und den Resultaten überwiegt (siehe Abbildung 5.6). Trotz dessen, dass die neutralen Daten mit auf 600 beschränkten Einträgen nicht die höchste Anzahl an Trainingsmaterial besitzen, fällt der Schwerpunkt der Daten auf die neutrale Stimmung. Im Vergleich zu einer einfachen logistischen Regression ist zu erkennen, dass die negative Stimmung am häufigsten erkannt wurde und am zweitmeisten die neutrale Stimmung. Die positive Stimmung wurde in keinem Fall erkannt. Die Ergebnisse lassen darauf schließen, dass eine logistischen Regressionensanalyse mit drei Stimmungen nicht möglich ist.

Eine Alternative zur Regression bietet in diesem Fall wieder die SVM. Aus der Klassifikation ergibt sich ein Genauigkeitsmaß von 51,68%. Für die F1-Werte ergeben sich folgende Werte:

	negative Stimmung	neutrale Stimmung	positve Stimmung
F1	52,14%	58,40%	33,66%

Tabelle 5.2: In dieser Tabelle ist der F1-Wert der drei Stimmungen nach dem Training der SVM abgebildet.

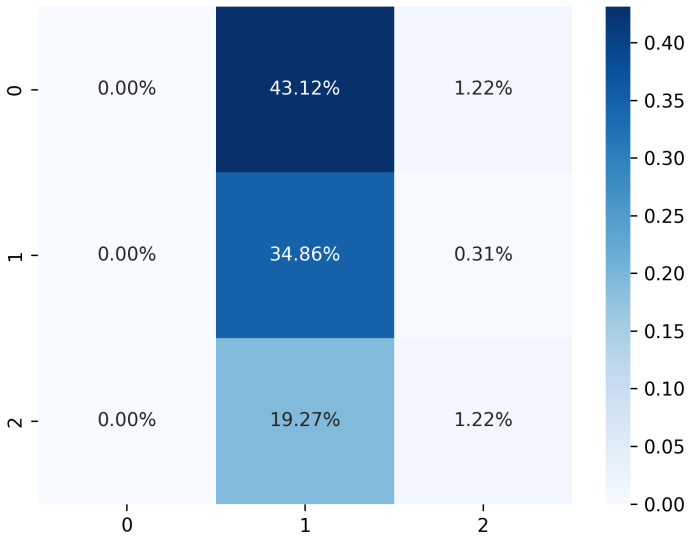

Abbildung 5.6: Die Confusionmatrix zeigt das Ergebnis der logistischen Regressionsanalyse mit allen drei Stimmungen und einem angepassten Datensatz mit Beschränkung auf 600 Einträge. 0 entspricht hierbei negative Stimmung, 1 neutral und 2 positiv.

Dieses Ergebnis zeigt, dass die positive Stimmung auch hier wieder etwas schlechter zu erkennen ist (Vergleiche Abbildung A.2 aus dem Anhang). Als Vergleich bietet sich auch ein Decision Tree an. Dieser erzeugt bei gleicher Eingabe ein Genauigkeitsmaß von 47,70% (Vergleiche Abbildung A.1 aus dem Anhang).

Kapitel 6

Lösungsansatz

Aufgrund der vorherigen Diskussion und der Erkenntnis 5.1 stellt sich nun die Frage, ob es weitere Möglichkeiten gibt, Metriken zur Vorhersage zu definieren. Der hier verwendete Lösungsansatz ergibt sich aus der Grafik: 6.1

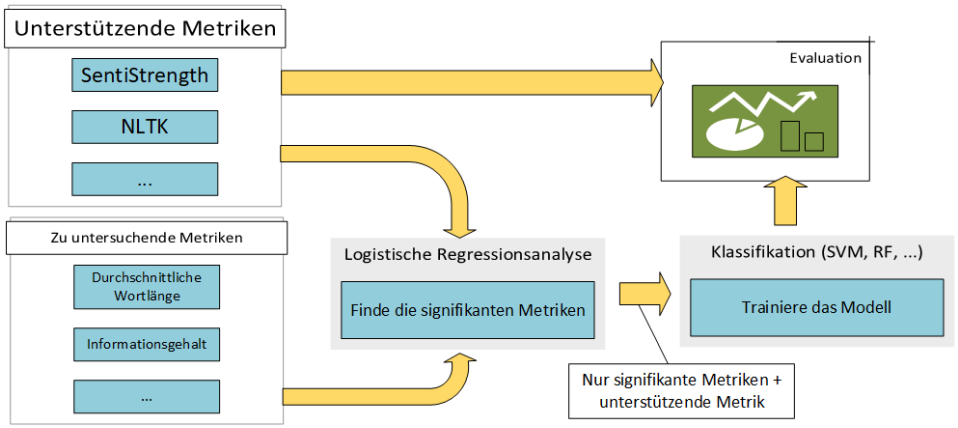

Abbildung 6.1: Die Grafik präsentiert die „Pipeline“ für die Evaluation bzw. den Lösungsansatz.

Statt lediglich die objektiven Metriken zu nutzen, um die Stimmung der Aussagen korrekt zu validieren, wurde hierbei der Ansatz verfolgt ein Modell aus subjektiven, sowie verstärkenden (objektiven) Metriken zu entwickeln, um die Klassifikation zu verbessern. Dieser Ansatz wird verfolgt, da es nicht möglich war ein Modell zu trainieren, welches nur mit Hilfe verschiedener objektiver Metriken die Stimmung von Entwickelnden vorhersagen sollte. Statt passende Metriken und Datensätze zu finden, bietet sich die Möglichkeit an vorhandene Tools mit zusätzlichen Metriken zu verbessern, um eine genauere

Klassifikation durchzuführen. Gleichzeitig ist es möglich die Einflüsse der einzelnen Metriken auf die Stimmung zu vergleichen. Am Ende dieses Kapitels wird ein trainiertes logistisches Regressionsmodell, ein Maschinenmodell und die Metriken mit ihren Einflüssen resultieren.

6.1 Evaluation der Metriken

Dieser Abschnitt ist in vier kleinere Abschnitte aufgeteilt. In dem ersten Teil wird evaluiert, wie die Performanz von SentiStrength ohne weiteren Metriken oder anderer Einflüsse ist. Diese Werte bilden die Referenz für die kommenden Analysen. Im zweiten Abschnitt werden die Metriken analysiert, die einen signifikanten Einfluss auf die Verbesserung der Ergebnisse haben. Im dritten Abschnitt werden die signifikanten Metriken verwendet, um damit ein Maschinenmodell zu trainieren. Diese Metriken helfen dabei weitere Anpassungen an den Maschinenmodell vorzunehmen, sodass die Performanz verbessert wird. Im letzten Abschnitt werden die einzelnen Metriken genauer untersucht, evaluiert und welchen Einfluss die einzelnen Metriken auf die Stimmung haben. Aus der Analyse wird eine Tendenz generiert, wie sich die Stimmung ändert je höher die berechneten Werte sind.

6.1.1 Evaluation von SentiStrength

Der erste Abschnitt befasst sich mit der logistischen Regressionsanalyse. Zuerst muss evaluiert werden welches Genauigkeitsmaß mit Hilfe von SentiStrength erreicht wird, wenn nur diese Metrik als Eingabe verwendet wird. Als Alternative neben SentiStrenth bietet sich NLTK an. Beide Tools bieten jeweils eine domänunabhängige Evaluation. Das Modell beinhaltet als Label positiv (1) und negativ (0). Die Eingabe in das Modell erfolgt durch die Auswertung der Aussagen durch SentiStrength. Die Eingabe für den Satz „Got it, thanks for the explanation.“ entspricht nach der Auswertung dem Wert 1 (positiv). Das erwartete Label aus dem Datensatz ist ebenfalls 1. Der Datensatz wurde wie zuvor in ein Trainingsdatensatz (80%) und in ein Testdatensatz (20%) aufgeteilt, damit overfitting vermieden wird. Das Resultat aus dieser Analyse ergab ein Genauigkeitsmaß von 79,22% und liefert somit ein besseres Ergebnis, als die Analyse aus dem vorherigen Kapitel. Die Precisison liegt für das positive Label bei 63,88% und bietet Verbesserungspotential. Insgesamt liegt der F1-Wert bei der negativen Stimmung bei 81,54% und für die positive Stimmung

bei 76,24% (Vergleiche Abbildung 6.2). Daraus folgt, dass es möglich ist mit diesem Modell eine Stimmung mit hoher Wahrscheinlichkeit vorauszusagen.

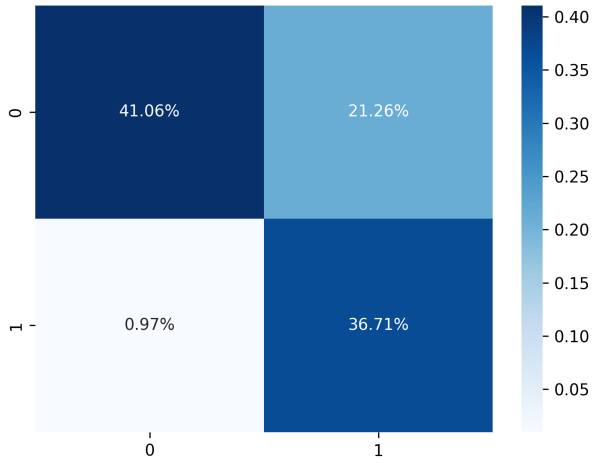

Abbildung 6.2: Die Confusion Matrix zeigt das logistische Regressionsmodell mit der Metrik „SentiStrength“.

6.1.2 Die signifikanten Metriken

Nachdem nun ein Referenzwert ermittelt wurde, wird im nächsten Schritt mit Hilfe der logistische Regressionsanalyse festgestellt werden, welche Metriken eine Verbesserung des Genauigkeitsmaß und des F1-Werte erzeugen. Insgesamt wird evaluiert, ob die Metriken einen generellen signifikanten Einfluss auf die Ergebnisse haben. Dazu werden die folgenden Metriken aus dem vorherigen Kapitel verwendet:

- Durchschnittliche Zeichenanzahl
- Durchschnittliche Anzahl der Wörter
- Durchschnittliche Anzahl von Satzzeichen
- Anzahl Emoticons in Kombination mit Satzzeichen wie „ !!! “
- Informationsgehalt

Diese Metriken wurden bewusst für eine Analyse ausgewählt, da für hohe Anzahl von Aussagen Werte ermittelt werden konnten. Die

Metrik „Anzahl Emoticons in Kombination mit Satzzeichen“ besitzt zwar nicht viele Werte, dafür bieten sie aber für die Feststellung der Polarität solide Vorhersagen (Vergleiche Diskussion 4.3).

Bei der Evaluation der Ergebnisse mit den fünf Metriken und SentiStrength fällt auf, dass die Metriken „Durchschnittliche Zeichenanzahl“ (91,8%), „Durchschnittliche Anzahl von Satzzeichen“ (75,3%) und „Durchschnittliche Anzahl von Wörtern“ (26,4%) nicht signifikant sind, sofern die reguläre Schranke bzw. das Signifikanzniveau von $< 5\%$ betrachtet. Auch fällt auf, dass die „Emoticons“ nicht signifikant sind, trotz dessen dass sie eine wichtige Rolle in der Identifizierung spielen. Dies liegt daran, dass wenn eine Aussage keine Emoticons oder besondere Satzzeichen besitzen, so wird der Wert auf 0 gesetzt und als neutral bewertet. Bei der Mehrheit der Aussagen existieren allerdings keine Emoticons oder besondere Satzzeichen, trotz dessen dass die Stimmung als positiv oder negativ klassifiziert wurde.

Zu der Analyse der einzelnen Metriken fehlt noch die Evaluation des generellen Modells. Bei einem Vergleich des Genauigkeitsmaßes ist auffällig, dass diese eine Verbesserung auf 87,92% erhalten hat. Die Klassifikationen wurden wie zuvor ermittelt, durch das Zuordnen der vorhergesagten Werte $< 0{,}5$ dem Label 0 und Werte $> 0{,}5$ dem Label 1. Die Confusion Matrix (vergleiche Abbildung 6.3) zeigt außerdem auf, dass das Modell deutlich ausgeglichener ist, als wenn die zusätzlichen Metriken nicht verwendet werden. Als Indikatoren für ein ausgeglicheneres Modell dient, wie auch in dem vorherigen Kapitel, der F1 Wert. Dieser hat bei der negativen Stimmungen einen Wert von 90,97% und bei positiver Stimmung einen Wert von 81,75%. Diese Werte zeigen, dass das Modell eine bessere Performanz liefert und weniger falsche Klassifikationen durchführt. Durch die logistische Regressionsanalyse wurde ermittelt, dass es nicht signifikante Metriken gibt. Werden die Metriken entfernt, die einen p-Wert $> 5\%$ haben, so kommt eine leichte Verschlechterung der Ergebnisse auf ein Genauigkeitsmaßes von 86,95% (vergleiche Abbildung 6.4) zustande. Der F1 Wert ist damit auch bei beiden Stimmungen leicht gesunken. Damit das ursprüngliche Genauigkeitsmaß wieder hergestellt wird, wurden sukzessiv die Metriken wieder aufgenommen und die Veränderung der Werte beobachtet. Dabei stellte sich heraus, dass durch das Hinzufügen der Metrik „Emoticons“ das ursprüngliche Genauigkeitsmaß wieder erreicht wurde. Daraus ergeben sich folgende Metriken als signifikant: Emoticons mit besonderen Satzzeichen und Informationsgehalt.

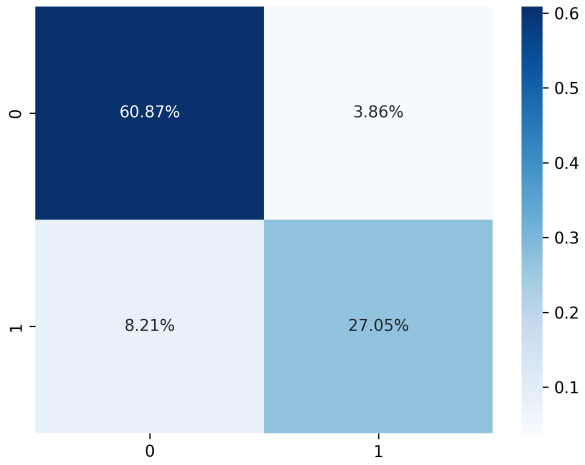

Abbildung 6.3: Die Confusion Matrix zeigt das logistische Regressionsmodell mit allen Metriken einschließlich „SentiStrength“an.

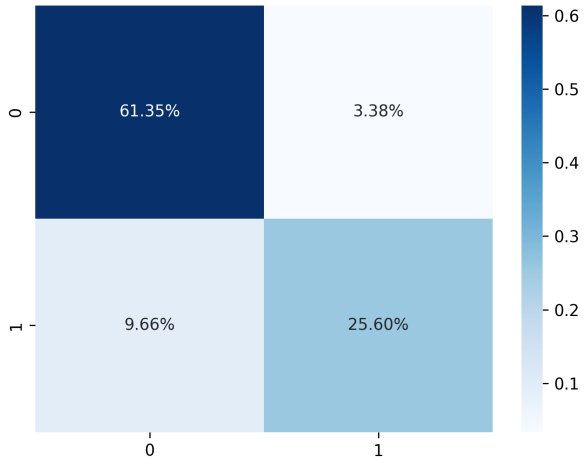

Abbildung 6.4: Die Confusion Matrix zeigt das logistische Regressionsmodell mit ausgewählten Metriken einschließlich „SentiStrength“an.

6.1.3 Training der Maschinenmodelle

Der nächste Schritt befasst sich nun mit dem Training verschiedener Maschinenmodelle. Hierfür wurde als Ausgang eine SVM benutzt.

Diese SVM nutzt die gleiche Konfiguration wie die logistische Regressionsanalyse. Nach dem Trainieren ergibt sich ein Genauigkeitsmaß von 89,85% und liegt damit höher als die Ergebnisse der logistischen Regressionsanalyse (vergleiche Confusion Matrix 6.5). Der Naive Bayes Klassifizierer besitzt ein Genauigkeitsmaß von 89,37% und der Random Forest Klassifizierer liegt mit einem Genauigkeitsmaß von 85,02% etwas unter den anderen Modellen. Der F1-Wert liegt bei der SVM bei 92,36% für die negative Stimmung und 84,89% bei den positiven Kommentaren. Diese Werte sind besser als die Ergebnisse, die nur allein durch SentiStrength erzeugt wurden.

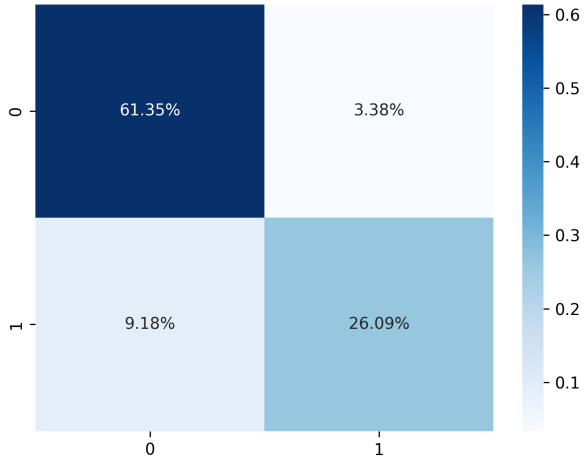

Abbildung 6.5: Die Confusion Matrix zeigt das Ergebnis der trainierten SVM mit den ausgewählten Metriken einschließlich „SentiStrength“ an.

Die Ergebnisse sind sehr positiv, haben aber eventuell zu einem Overfitting geführt, denn dies belegt der Decision Tree mit einer Tiefe von 16. Dabei entspricht das Genauigkeitsmaß einem Wert von 83,57%. Um Overfitting zu verhindern wurde bereits ein Test- und ein Trainingsdatensatz verwendet. Weitere Möglichkeiten, um Overfitting zu vermeiden sind:

- Metrik „Durchschnittliche Anzahl von Wörtern“ wieder in das Training einführen.
- Ein unabhängigen Datensatz zum Testen nutzen.
- Hyperparameter anpassen und beschränken.

- Andere unterstützende Metrik wie NLTK nutzen und validieren.

Beim zusätzlichen Verwenden der vierten Metrik „Durchschnittliche Anzahl von Wörtern“ ist das Genauigkeitsmaß etwas schlechter, aber die Maschinenmodelle sind etwas ausgeglichener durch die leicht verrauschten Daten. Bei einer Evaluation der vier Metriken mit Hilfe der logistischen Regressionsanalyse besitzt die neue Metrik in diesem Kontext einen p-Wert von 0,0% und ist somit signifikant, wobei die Metrik der Emoticons in diesem Kontext nicht mehr als signifikant (11,10%) angesehen wird. Daher wäre eine Überlegung diese Metrik zu entfernen. Dies führt zu einer minimalen Verschlechterung der Performanz und daher wird diese Metrik weiter verwendet. Eine andere Möglichkeit besteht darin die Metrik mit weiteren Inhalten weiterzuentwickeln:

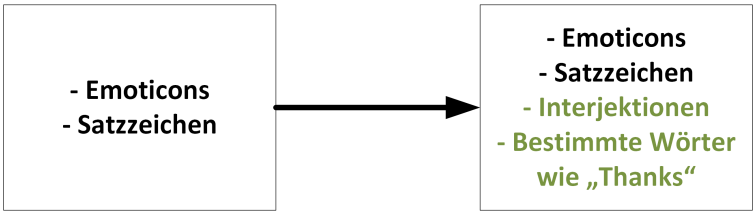

Abbildung 6.6: Weiterentwickelte Metrik für bessere Klassifikation von Emoticons.

Beim Hinzufügen der Interjektionen und bestimmten Wörtern als Metrik wird eine höhere Abdeckung der Aussagen erzielt, denn damit müsste in jedem Kommentar lediglich eines der vier Metriken enthalten sein, um eine Stimmung für die Aussage treffen zu können. Diese zusammengesetzte Metrik wurde in diesem Kontext noch nicht auf die Signifikanz und Performanz geprüft.

Einen unabhängigen Datensatz zu verwenden stellt sich als schwierig heraus, da ein Datensatz benötigt wird, welcher ähnliche Strukturen aufweist, wie der Ausgangsdatensatz (vergleiche Paper von Novielle et al. [22]). Die Datensätze von JIRA und Stack Overflow unterscheiden sich in ihrer Strukur und ihrem Inhalt. Ein Beispiel wäre hierfür Tickets bei JIRA, die mit einer Nummer versehen sind. Diese müssen vorverarbeiten und gegebenenfalls weiter angepasst werden. Die Hyperparameter anzupassen bzw. das Training zum Beispiel auf eine gewisse Anzahl an Iterationen zu beschränken, führt auch zu keiner Veränderung der Ergebnisse. Damit das Hyperparameterproblem ausgeschlossen werden

kann, wurde AutomML verwendet. Dieses Maschinenmodell kommt auf ein Genauigkeitsmaß von 86,75% und ist damit etwas schlechter als die SVM. Daraus resultiert, dass die Hyperparameter bereits einen optimalen Wert besitzen. Die letzte Möglichkeit ist das Verwenden von beispielsweise NLTK statt SentiStrength. Die Ergebnisse sollten sich hierbei aber nur leicht unterscheiden, da nach dem Paper von Jongeling et al. [14] zwar die Übereinstimmung von SentiStrength und NLTK bei nur etwa 22% liegt, aber die Unterscheidungen häufig nur einen Schritt auseinander liegen (positiv → neutral oder neutral → negativ). Neben NLTK bietet sich auch die Möglichkeit an ein domänenspezifisches Tool zu verwenden, damit wird ein gegebenenfalls anderes Resultat generiert und ein Vergleich ist möglich.

6.1.4 Resultate der Metriken

In diesem letzten Abschnitt geht es um die Auswertung der zuvor evaluierten Metriken. Dabei wird der Einfluss der Metriken auf die Stimmung näher untersucht.

Durchschnittliche Anzahl von Wörtern

Bei der Metrik für die Anzahl der Wörter hat sich ergeben, dass je mehr Wörter eine Aussage hat, desto höher ist die Wahrscheinlichkeit, dass die Aussage als negativ klassifiziert wird. Das Resultat korreliert mit der Analyse aus Kapitel 4. Hier wurde bei dem Boxplot 4.3 festgestellt, dass die negativen Aussagen im Durchschnitt mehr Wörter beinhalten als die positiven.

Anzahl von Emoticons mit Stimmung

Die Emoticons beeinflussen die Stimmung auf unterschiedliche Weise: Während „:-)“ eine positiv Stimmung vermittelt, zeigt „:-(“ eine negative Stimmung. Dieses Resultat spiegelt sich auch in den Koeffizienten der logistischen Regressionsanalyse wider. Wenn ein Emoticon enthalten war, der eine positive Stimmung vermittelt, so wurde diese Aussage auch als positive bewertet. Ein anderes Problem, welches zusätzlich zu dem Problem mit der Häufigkeit von Emoticons aufgetreten ist, ist, dass einige Aussagen aus sarkastischen Gründen Emoticons beinhalten, obwohl diese eine andere Stimmung suggerieren. Dieses Problem kann durch ein Tiefenverständnis der Sätze gelöst werden (zum Beispiel durch NLTK).

Informationsgehalt

In der Analyse aus dem Kapitel 4 wurde gezeigt, dass je höher der Informationsgehalt ist, desto eher wird eine Aussage als positiv gekennzeichnet. Die logistisch Regressionsanalyse kam zu einen ausgeglichenen Ergebnis. Der Koeffizient für diese Variable ist negativ und bewirkt somit eher eine negative Stimmung je höher dieser Wert ausfällt. Da dieser Koeffizient aber nur recht klein ist, hat dieser Wert nicht den größten Einfluss auf die Gesamtstimmung.

6.2 Weitere Resultate

Bei einer Evaluation des Datensatzes mit der zusätzlichen neutralen Stimmung resultiert ein sehr unausgeglichenes Modell. Denn die meisten Aussagen sind neutral (siehe Abbildung 6.7). Dieser Grund erklärt das Weglassen der neutralen Stimmungen aus den Modellen.

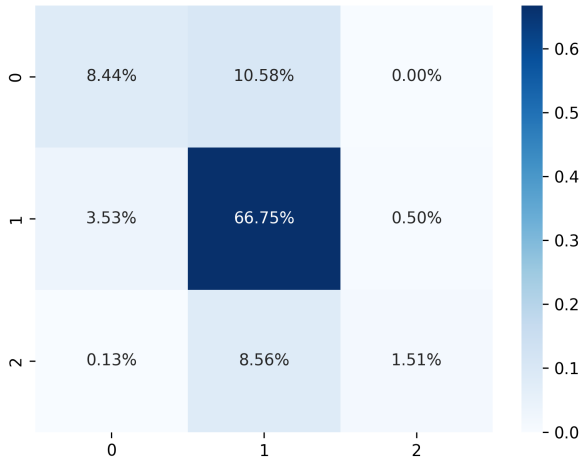

Abbildung 6.7: Die Confusion Matrix zeigt das Ergebnis der trainierten SVM. Hier wurde zusätzlich die neutrale Stimmung mit eingeführt.

Zwar liegt bei dem Modell das Genauigkeitsmaß bei 76,70%, aber dies liegt überwiegend an den neutralen Einträgen. Eine interessante Erkenntnis hierbei ist, dass lediglich 0,13% der Fälle statt positiv negativ und umgekehrt klassifiziert worden sind. Aus der vorherigen Erkenntnis ergibt sich, dass es größere Unterschiede zwischen den

positiven und negativen Stimmungen gibt, da es kaum Sprünge zwischen den beiden Klassen gab. Der F1 Wert für die einzelnen Metriken liegt bei den Stimmungen positiv (54,25%), neutral (85,20%) und negativ (24,74%) und damit hat lediglich die neutrale Stimmung einen soliden Wert. Das Resultat führt somit zu einem ähnlichen Ergebnis, wie die Analyse mit neutralen Werten aus dem Kapitel 5.

Ein weiteres Ergebnis aus der Analyse besteht darin die Aussagen von SentiStrength klassifizieren zu lassen. Das Resultat zeigt, dass es eine starke Verschiebung der Aussagen gibt. Insgesamt waren zuvor 9,35% der Aussagen positiv. Nachdem die Aussagen nach SentiStrength klassifiziert wurden, ist der Wert auf 28,11% gestiegen. Für die negative Stimmung ergab sich ein plus von 6,36%. Die Differenz des Zuwachses der beiden Stimmungen fällt auf die neutrale Stimmung (vergleiche Tabelle 6.1). Ein möglicher Grund ist, dass einzelne Wörter ausreichen die Stimmung in dem Kontext zu verschieben. Hingegen ist es Entwickelnden möglich ein Kontext zu betrachten und somit eine fundierter Meinung zu der aktuellen Stimmung haben.

	Positiv	Neutral	Negativ
#	1115 (28.11 %)	1939 (48.89%)	912 (23.00%)

Tabelle 6.1: Die Tabelle zeigt die Klassifikation des Stack Overflow Datensatzes nach SentiStrength.

Der nächste Schritt ist, die signifikanten Metriken aus dem vorherigen Teil zu übernehmen und ein Maschinenmodell zu trainieren. Die SVM kam im Zuge dessen auf ein Genauigkeitsmaß von 57,14%. Die Daten wurden wie auch zuvor aufgeteilt in positive Stimmung (Label 1) und negatives Stimmung (Label 0). Die neutralen Daten wurden wieder entfernt, da sie eine hohe Überschneidung mit den anderen Stimmungen aufweist. Bei einem Vergleich der Resultat aus Abbildung 6.8 fällt auf, dass auch hier kein eindeutiges Ergebnis für die Stimmung evaluierbar ist. Der einzige Unterschied zwischen dieser Grafik und der aus dem Kapitel 5 ist, dass hier die positive Stimmung überwiegt. Als Resultat ergibt sich ein ähnliches Ergebnis wie aus Kapitel 5, dass es nicht möglich ist die Stimmung mit den objektiven Metriken zu erkennen.

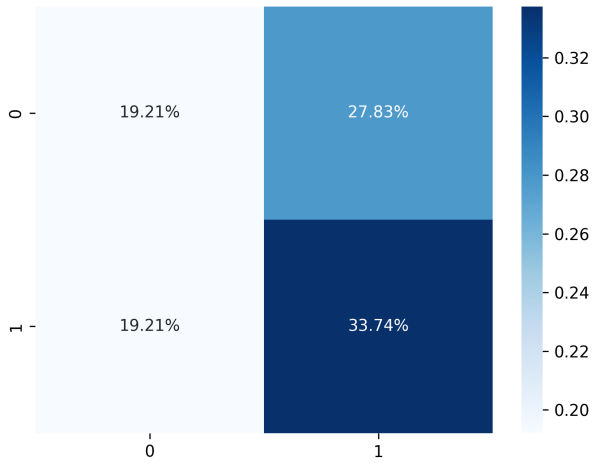

Abbildung 6.8: Die Confusion Matrix zeigt das Ergebnis der trainierten SVM, um die Stimmung von „SentiStrength“ vorherzusagen.

Kapitel 7

Diskussion

In der Einleitung wurde bereits diskutiert, dass Softwareprojekte beispielsweise häufig an Kommunikation scheitern. Die Sentimentanalyse soll dabei helfen Stimmungen von Entwickelnden wahrzunehmen und zu analysieren. Das Sentimentanalysetool SentiStrength besitzt zwar ein Genauigkeitsmaß von 79,22% und bietet damit eine solide Grundlage für die Klassifikation von Stimmung, es existiert gleichzeitig noch Raum für Verbesserung. Verschiedene Tools haben bereits eine Verbesserung der Vorhersage erzeugt, zum Beispiel durch das Anpassen der Datensätze, Überarbeitung der Lexika für die Domäne, Sichten von verschiedenen Keywords und das Vornehmen von semantischen Analysen. In dieser Masterarbeit wurde daher ein neuer Ansatz untersucht, ob es neben den zuvor genannten Anpassungen weitere Möglichkeiten gibt Stimmung vorherzusagen. Der Fokus lag dabei auf statistischen Metriken. Im Kapitel 5 wurde bereits gezeigt, dass keiner der Metriken eine fundierte Aussage über die Stimmung erzeugen kann, daher wurde direkt ein anschließender Ansatz verfolgt, bekannte Tools mit Hilfe von evaluierten Metriken zu verbessern. Das Resultat zeigte, dass es möglich ist mit statistischen Metriken diese Resultate zu verbessern. Diese Metriken wurden jedoch nur für zwei konkrete Datensätze evaluiert. Eine weitere Analyse für weitere Datensätze wäre daher notwendig. Führen diese Ergebnisse auch zu einer Verbesserung, so ist es möglich in der Zukunft diesen Ansatz weiter zu verfolgen. Eine weitere Frage wäre, ob sich einige Metriken auch ohne SentiStrength ausführen lassen, zum Beispiel durch das Wechseln der Sprache. Bisher wurden die Datensätze für die englische Sprache analysiert, da dies eine der häufigsten gesprochenen Sprachen ist. Unternehmen aus anderen Ländern, wie Deutschland oder Frankreich, nutzen eventuell andere Sprachen, wodurch sich die

Ergebnisse neu evaluieren lassen. Die Datenlage zu nicht-englischen Datensätzen ist sehr limitiert.

Neben der Evaluierung der Sentimentanalyse stellt sich die Frage, wie diese Erkenntnis im Bereich des Software Engineering verwendet werden kann. Verschiedene Analysetools wurden bereits in sozialen Netzwerken verwendet und konnten somit böswillige Kommentare identifizieren. Im Bereich der Softwareentwicklung innerhalb eines Unternehmens werden häufig Tools wie JIRA oder GitHub verwendet. Diese Kommentare/Nachrichten lassen sich automatisiert von einem Sentimentanalysetool auswerten und generieren eine Gesamtstimmung. Die Tools sind natürlich aus Datenschutzgründen schwierig einzusetzen, da alle Entwickelnden diese Entscheidung akzeptieren müssen. Nichtsdestotrotz zeigen die Tools die aktuelle Stimmung und dadurch implizit auch die Motivation des Teams auf. Bei häufig negativer Stimmung ist es möglich Probleme im Team anzusprechen. Aus diesen genannten Gründen ist es sinnvoll die Sentimentanalyse weiterzuentwickeln und zu verbessern. Denn mit einem Genauigkeitsmaß von etwa 80% wäre jede fünfte Aussage falsch und könnte zu Fehlinterpretation führen.

In der Vergangenheit wurde bereits von vielen Forschenden der Versuch unternommen, herauszufinden was Stimmung ist und ob es möglich ist diese zu messen. Der häufigste Ansatz dabei war Emotionen auf Stimmungen abzubilden. Dies führte zu dem Folgeproblem, dass nicht mehr die Stimmung (positiv, negativ und neutral) gemessen werden musste, sondern mehrere Emotionen. Andere Ansätze wie bei den Papern über SentiCR oder Senti4SD waren, dass verschiedene Metriken und andere Analysen verwendet werden, um eine Vorhersage zu verbessern. Ein anderes Beispiel ist das Tool NLTK, welches versucht durch neuronale Netze ein Tiefenverständnis über Aussagen zu erzeugen. Dieser Ansatz führte bislang zu einem soliden Ergebnis, war aber hinsichtlich der Klassifikation nicht wesentlich besser als andere lexikonbasierte Tools. Kombinationen von mehreren Tools führt zwar zu einer Verbesserung der Ergebnisse, aber das Maß der Genauigkeit wurde nicht besser als 87%.

In der Literatur sind bisher nur Ansätze verfolgt worden, die die Vorhersage der Stimmung durch inhaltliche Anpassung verschiedener Tools verbessern. Der Ansatz, dass statistische Methoden verwendet werden, ist neu und bietet viel Verbesserungs- und Analysepotential

für die Zunkunft. Auf einer Webseite[1] wurde mit der Aussage „Eine simple Auswertung nach Methoden der Statistik würde den zweiten Satz besser bewerten, [...]“ gesagt, dass einfache statistische Metriken den zweiten Satz „Ganz gut, erfüllt so seinen Zweck. “ als positiver bewerten würden als den ersten Satz „Bin begeistert! “. Aus der Analyse im Kapitel 4 würde diese übereinstimmen, denn je mehr Wörter oder Zeichen es gibt desto negativer war eine Aussage. Jedoch ergibt sich für den Informationsgehalt des ersten Satzes einen Wert von 2 und für den längeren Satz einen Wert von 1 und bildet somit einen Kontrast zum vorherigen Ergebnis. Eine weitere Forschung in diesem Bereich kann aus diesem Grund weitere Metriken zur verbesserten Vorhersage entwickeln.

7.1 Grenzen der Arbeit

Für diese Masterarbeit wurden mehrere Datensätze evaluiert. Zwei dieser Datensätze wurden für die Bestimmung der Metriken ausgewählt und bilden eine Einschränkung. Damit eine bessere Evaluation der Metriken und ihren Einfluss gelingen kann, werden verschiedene Datensätze oder eine Kreuzvalidierung mit verschiedenen Plattformen benötigt. Die Abweichung der Ergebnisse sollten nur minimal sein, da bereits von Novielli et al. [23] eine Kreuzvalidierung verschiedener Datensätze erfolgt ist. Ein weiterer Aspekt ist, dass SentiStrength und andere Sentimentanalysetools in den vorgestellten Papern bessere Performanz (von etwa 85%) lieferten als die Performanz der Tools in dieser Masterarbeit. Der Grund für dieses Ergebnis liegt an einer anderen Vorverarbeitung der Datensätze. In den Datensätzen aus den Papern wurden deutlich mehr Inhalte aus den Aussagen entfernt, zum Beispiel falsch geschriebene Wörter oder mehrere Zeichen hintereinander. Eine Möglichkeit dies zu umgehen und gleichzeitig die Metriken mit Hilfe von Kreuzvalidierung zu prüfen, ist der größere Datensatz von JIRA, Stack Overflow und GitHub mit etwa 2,5 mio Einträge. Bei diesem Datensatz existieren ausreichend viele Aussagen aus verschiedenen Bereichen und bilden somit eine solide Basis für eine Analyse. Ein Problem dieses Datensatzes ist, dass die Aussagen nicht von Entwickelnden klassifiziert wurden und sich dies aufgrund der hohen Anzahl der Aussagen nicht realisieren lässt. Es ist möglich diese Aussagen mit einem Sentimentanalysetool zu klassifizieren, doch

[1] https://www.ionos.de/digitalguide/online-marketing/web-analyse/was-ist-sentiment-analyse/

dies könnte zu einer Verfälschung der Ergebnisse führen.

Kapitel 8

Zusammenfassung und Ausblick

8.1 Zusammenfassung

In dieser Masterarbeit wurde zu Beginn die Frage gestellt, ob es möglich ist subjektive Wahrnehmung von Entwickelnden durch objektive Metriken herauszufinden. Es sollte herausgefunden werden, ob es möglich ist Stimmung im Team zu messen, damit eine Gesamtstimmung ableitbar ist. Diese Gesamtstimmung soll dazu verwendet werden, damit bei erhöhten Vorkommen von negative Texte, Kommentare oder Aussagen Probleme abgeleitet werden können und diese im Team angesprochen werden.

Bisher wurden verschiedene Tools verwendet, wie zum Beispiel SentiStrength oder NLTK, um Texte nach ihrer Stimmung zu klassifizieren. Ein Vergleich zwischen verschiedene Auswertungen der Tools und Entwickelnden stellte fest, dass es nur eine kleine Übereinstimmung der Aussagen gibt. Daher soll eine neue Möglichkeit entwickelt werden, um diese Übereinstimmung zu verbessern.

Damit die Übereinstimmung verbessert werden kann, wurde zuerst ein angemessener Datensatz exploriert und anschließend statistisch evaluiert. Für die beiden Datensätze wurden objektive bzw. statistische Metriken abgeleitet (Entstanden durch die Masterarbeit von Julian Horstmann [9]). Typische Metriken sind: Anzahl von Wörtern, Anzahl von Zeichen oder Anzahl von Satzzeichen. Bei diesen Metriken gab es größtenteils nur wenig Unterschiede zwischen den Stimmungen positiv, negativ und neutral. Aus diesem Grund wurde die neutrale Stimmung

entfernt. Mit den evaluierten Metriken wurde daraufhin eine SVM und ein logistische Regressionsmodel trainiert. Die entstandenen Modelle zeigten, dass die Modelle zwar ein hohes Genauigkeitsmaß besitzen, aber einen schlechten Ausgleich der Daten aufzeigen (Vergleiche F1-Wert 5). Daher wurden aus dem logistischen Regressionsmodell die signifikanten Metriken abgeleitet und wiederholt evaluiert. Das Ergebnis hieraus ist eine leichte Verbesserung der trainierten Modelle, aber nicht ausreichend um Erkenntnisse zu einzelnen Metriken zu erhalten, da die Anzahl der positiven sowie negativen Vorhersagen unausgeglichen sind.

Daraus folgt, dass die evaluierten Metriken nicht ausreichend sind, um ein trainiertes Modell zu erhalten. Aus diesem Grund wurde eine neue Metrik verwendet: SentiStrength. Diese Metrik ist eine subjektive Metrik, da die einzelnen Wörter von Forschenden klassifiziert wurden. Bei dem Versuch eine SVM mit SentiStrength als Metrik zu trainieren wurde ein Genauigkeitsmaß von 79,22% bei dem Datensatz von Stack Overflow generiert. Durch das Hinzufügen von objektiven Metriken, soll nun das Genauigkeitsmaß verbessert werden. Für diese Zwecke wurden die Metriken Informationsgehalt, durchschnittliche Anzahl Wörter und die Emoticons ausgewählt. Dadurch konnte die Vorhersage auf ein Genauigkeitsmaß von etwa 86% verbessert werden. Jeder dieser Metriken gilt bei der logistischen Regressionsanalyse als signifikant, bis auf die Emoticons, da sie nur sehr spärlich vorkommen.

Aus der gesamten Analyse entsteht die Problematik, dass die evaluierten Metriken nicht ausreichend sind, damit eine Vorhersage der Stimmung gelingt. Es ist aber möglich mit Hilfe von subjektiven Metriken die Stimmmungsvorhersage zu verbessern und führt dazu, dass diese Metriken genutzt werden könnne, damit Probleme bei Entwicklungsteams effizienter aufgezeigt werden können.

8.2 Ausblick

Bei der Betrachtung der Analyse zu den einzelnen Metriken stellt sich die Frage, ob es neben den evaluierten Metriken jene gibt, die die Vorhersage weiterhin verbessern oder sogar ohne weitere subjektive Metrik vorhersagen können. Ein weiterer Aspekt wäre, dass ein Tiefenverständnis für Sätze generiert wird und inhaltlich auswertet. Hierfür kann beispielsweise ein neuronales Netz verwendet werden.

Mit Hilfe des Tiefenverständnisses wäre es möglich die Satzstrukturen besser zu verstehen und weitere Erkenntnisse können daraus gewonnen werden. Die in der Einleitung erwähnte Anwendung von Google ist durch Nutzendenrezenssion trainiert worden. Für die Domäne des Software Engineering ergibt sich die Möglichkeit einen anderen passenden Datensatz zu finden und ein angepasstes Netz zu trainieren.

Ein weiterer spannender Aspekt ist TensiStrength [32]. Mit Hilfe dieses Tools ist es möglich aus Texten herauszufinden, ob die Sätze ein erhöhtes Stresslevel der Nutzenden aufweisen. Eine Untersuchung könnte hierbei herausfinden, ob sich beispielsweise das Stresslevel bei Entwickelnden oder der Schreibstil verändert, wenn gewisse Abgabefristen für Projekte kurz bevorstehen. Auch hierzu können wieder Metriken definiert werden, die eine unterstützende Wirkung auf die Analyse des Stresslevels haben.

Ein weitere Idee ist zu evaluieren, wie verschiedene andere Medien Einfluss auf die Stimmung haben und ob es möglich ist eine Kombination zu generieren, um ein Gesamtstimmungsbild zu berechnen. Die Sentimentanalyse ist zwar textbasiert, aber es besteht die Möglichkeit verschiedene Medien auch textuell umzuwandeln. Weiterhin wäre es auch möglich Stimmen, Bilder und Videos mit Hilfe von neuronalen Netzen auszuwerten und daraus Stimmungen abzuleiten.

Die Principle Component Analysis (PCA) bietet die Möglichkeit Dimensionen zu reduzieren. Diese Analyse bietet im Gegensatz zur logistischen Regressionsanalyse die Möglichkeit herauszufinden, welche Eingabeparameter nicht für eine Klassifikation notwendig sind. Mit diesem Verfahren lassen sich Metriken herausfinden, die nur einen schlechten Einfluss auf die Resultate haben.

Anhang A

Ein Anhang

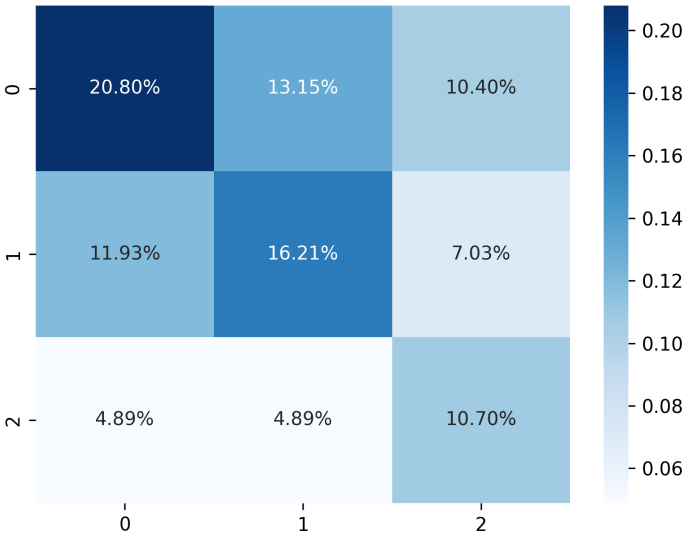

Abbildung A.1: Die Confusionmatrix zeigt das Ergebnis des Decision Trees mit allen drei Stimmungen und einem angepassten Dataset mit Beschränkung auf 600 Einträge für die neutrale Stimmung.

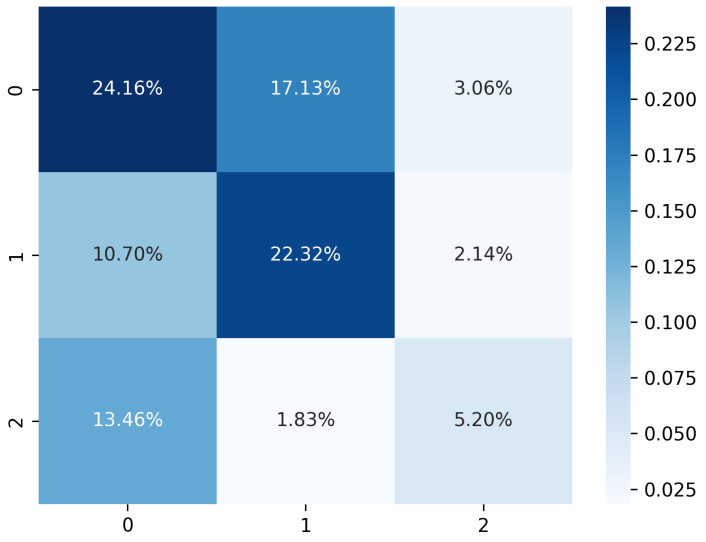

Abbildung A.2: Die Confusionmatrix zeigt das Ergebnis der Support Vector Machine mit allen drei Stimmungen und einem angepassten Dataset mit Beschränkung auf 600 Einträge für die neutrale Stimmung.

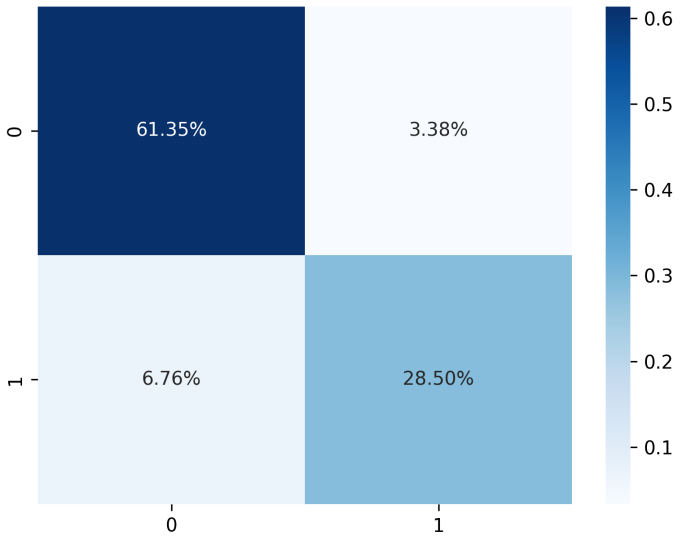

Abbildung A.3: Die Confusionmatrix zeigt das Ergebnis der Support Vector Machine mit allen drei Stimmungen und einem angepassten Dataset mit Beschränkung auf 600 Einträge für die neutrale Stimmung.

Literaturverzeichnis

[1] T. Ahmed, A. Bosu, A. Iqbal, and S. Rahimi. Senticr: A customized sentiment analysis tool for code review interactions, 2017.

[2] V. Balnat. Altes und neues zur interjektion, 2008.

[3] E. Bonnie. Complete collection of project management statistics 2015, 2015.

[4] S. Brand, T. Reimer, and K. Opwis. How do we learn in a negative mood? effects of a negative mood on transfer and learning, 2007.

[5] M. Butterick. All-caps text—meaning text with all the letters capitalized—is best used sparingly., 2018.

[6] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli. Sentiment polarity detection for software development, empirical software engineering, june 2018, volume 23, issue 3, pp 1352 - 1382, 2018.

[7] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter. Auto-sklearn 2.0: The next generation, 2019.

[8] P. Fitsilis. Measuring the complexity of software projects, 2009.

[9] J. Horstmann. Computer-gestützte analyse des kommunikationsverhaltens in entwicklerteams unter berücksichtigung digitaler medien, 2019.

[10] C. Iglesias and A. Moreno. Sentiment analysis for social media, 2019.

[11] N. Imtiaz. Sentiment and politeness analysis tools on developer discussions are unreliable, but so are people, 2018.

[12] R. M. Islam and M. Zibran. Leveraging automated sentiment analysis in software engineering, 2017.

[13] G. James, D. Witten, T. Hastie, and R. Tibshirani. Support vector machines, 2021.

[14] R. Jongeling, P. Sarkar, S. Datta, and A. Serebrenik. On negative results when using sentiment analysis tools for software engineering research, 2017.

[15] R. Kaur and K. K. Chahal. Analysis of developers' sentiments in commit comments, 2021.

[16] M. Kodali. Traceability of requirements in scrum software development process, 2015.

[17] R. Lazarus. Emotion and adaptation, 1993.

[18] O. Marco, M. Alessandro, D. Giuseppe, T. Parastou, T. Roberto, M. Michele, and A. Bram. The emotional side of software developers in jira, 2016.

[19] S. McAllister, A. Vincent, A. Hassett, M. Whipple, T. Oh, R. Benzo, and L. Toussaint. Psychological resilience, affective mechanisms and symptom burden in a tertiary-care sample of patients with fibromyalgia, 2013.

[20] A. Murgia, P. Tourani, B. Adams, and M. Ortu. Do developers feel emotions? an exploratory analysis of emotions in software artifacts, 2014.

[21] N. Novielli, F. Calefato, D. Dongiovanni, D. Girardi, and F. Lanubile. Can we use se-specific sentiment analysis tools in a cross-platform setting?, 17th international conference on mining software repositories, october 5–6, 2020, seoul, republic of korea, 2020.

[22] N. Novielli, F. Calefato, F. Lanubile, and A. Serebrenik. Assessment of se-specific sentiment analysis tools: An extended replication study, 2020.

[23] N. Novielli, F. Calefato, F. Lanubile, and A. Serebrenik. Assessment of off-the-shelf se-specific sentiment analysis tools: An extended replication study, 2021.

[24] N. Novielli, F. Lanubile, and D. Girardi. A benchmark study on sentiment analysis for software engineering research. In *2018*

IEEE/ACM 15th International Conference on Mining Software Repositories MSR, pages 364–375. IEEE;, 2018.

[25] N. Novielli, F. Maiorano, F. Lanubile, and F. Calefato. Sentiment polarity detection for software development, 2018.

[26] M. Obaidi and J. Klünder. Development and application of sentiment analysis tools in software engineering: A systematic literature review., 2001.

[27] B. Pang and L. Lee. Opinion mining and sentiment analysis, 2008.

[28] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques., 2008.

[29] W. Parrott. Emotions in social psychology., 2001.

[30] S. Staab. Robust – risk and opportunity management of huge-scale business community cooperation, 2013.

[31] G. Stepanek. *Software Projects Secrets: Why Projects Fail.* Apress, 2012.

[32] M. Thelwall. Tensistrength: Stress and relaxation magnitude detection for social media texts. information processing and management, 2017.

[33] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. Sentiment strength detection in short informal text., 2010.