Nachrichtenbasierte prädiktive Bewertung von Aktienhandelsstrategien

Bachelor-Thesis von Elias Heftrig aus Wiesbaden
Tag der Einreichung:

1. Gutachten: Prof. Dr. Johannes Fürnkranz
2. Gutachten: Dr. Eneldo Loza Mencía

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Informatik
Knowledge Engineering Group
Technische Universität Darmstadt

Nachrichtenbasierte prädiktive Bewertung von Aktienhandelsstrategien

Vorgelegte Bachelor-Thesis von Elias Heftrig aus Wiesbaden

1. Gutachten: Prof. Dr. Johannes Fürnkranz
2. Gutachten: Dr. Eneldo Loza Mencía

Tag der Einreichung:

Erklärung zur Bachelor-Thesis

Hiermit versichere ich, die vorliegende Bachelor-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 30. Oktober 2017

(E. Heftrig)

Inhaltsverzeichnis

Zusammenfassung

Der Siegeszug der Mikroelektronik ist im vollen Gange und die künstliche Intelligenz hat gute Aussichten darauf, den technologischen Meilenstein des 21. Jahrhunderts zu markieren. So verwundert es nicht, dass auch die Marktteilnehmer an den Börsen dieses Thema immer häufiger für sich entdecken, um daraus eine Vorteil gegenüber ihresgleichen zu schlagen. In dieser Bachelorarbeit wird daher untersucht, ob mittels einer Methode des Textmining anhand von Adhoc-Mitteilungen der Erfolg einer kurzfristigen Investition in die zugehörigen Aktien vorhergesagt werden kann. Im Zuge dessen werden drei Methoden untersucht, die Mitteilungen als numerische Vektoren zu repräsentieren. Wie sich herausstellt, gelingt es zwar nicht, mit hinreichender Genauigkeit die Investitionsentscheidungen mit der größten Rendite vorherzusagen, aber dennoch, den Grundstein für ein potentiell profitables Handelssystem zu legen. Unter den drei Methoden der Textrepräsentierung kann eine identifiziert werden, mit der es in diesem Kontext nicht gelingt, aussagekräftige Klassifikatoren zu konstruieren.

Abstract

The triumph of microelectronics has caught full momentum by today and artificial intelligence is on course for becoming the technological milestone of the 21^{st} century. No wonder, market players at the scurities exchanges everywhere are becoming more and more aware of that matter, trying to gain an advantage over one another. For that reason, this Bachelor's Thesis examines, whether one can predict the success of a short-term investment into shares by applying a text mining method to ad-hoc disclosures. Conducting the necessary studies, three methods of representing the disclosures by numerical vectors are investigated. As is shown, there is no success achieved predicting the investment decisions with the best yield with sufficient accuracy. Nevertheless, a foundation can be laid for a potentially profitable trading system. Among the three methods of text representation one can be identified that, in this context, is of no use constructing a sound classifier.

1 Einleitung

Ob als Spamfilter, bei der Verarbeitung von Anfragen durch Internetsuchmaschinen oder in digitalen Sprachassistenten, Verfahren aus der Domäne der künstlichen Intelligenz finden in einer Vielzahl von Einsatzszenarien Anwendung. Dank reger Forschung und der fortschreitend rasanten Steigerung des Potenzials mikroelektronischer Systeme werden unentwegt neue Anwendungsgebiete erschlossen und was vor Jahren noch als unpraktikabel oder undenkbar galt, erreicht immer häufiger den Stand der ökonomischen Realisierbarkeit. Aus dem informationstechnischen Alltag ist sie nur noch schwer wegzudenken und längst durchdringt die künstliche Intelligenz weitere wissenschaftliche Disziplinen, wie etwa die Regelungstechnik, Biotechnologie oder den Maschinenbau, in denen sie unter anderem neue Vorgänge und Methoden für Analyse und Entwurf ermöglicht.

So ist es wenig verwunderlich, dass dieser Trend auch den Börsenhandel erfasst, dessen Akteure stets auf der Suche nach strategischen Vorteilen sind. Das explizite oder implizite Antizipieren von Aktionen der anderen Marktteilnehmer ist dabei von zentraler Bedeutung. Viele Anleger treffen ihre Investitionsentscheidungen auf der Basis von Nachrichten, welche die Finanzinstrumente betreffen, mit denen sie handeln. Wird eine solche Meldung veröffentlicht, erfolgt bei entsprechender Relevanz eine Neubewertung der einzelnen Instrumente durch die Investoren und der Markt reagiert auf die neue Information durch das Anpassen der Kurse, zu denen die Investitionsobjekte gehandelt werden. Da dies nicht ohne zeitliche Verzögerung erfolgen kann, ist diese Kursanpassung stets mit einer Trägheit behaftet. Es resultiert ein Vorteil für denjenigen, der die Nachricht besonders schnell verarbeiten und auf sie reagieren kann.

Folglich ist es naheliegend, dies maschinell zu bewerkstelligen. Die Idee des computergestützten Handels ist zweifelsohne nichts grundsätzlich Neues. Bereits Jahr 2008 wurde etwa 43% des Volumens am Handel der deutschen Börse durch Algorithmen generiert und schon 2004 gelang es, mittels Textmining ein nennenswert profitables Handelssystem zu entwickeln[Mittermayer, 2004]. Jedoch stellt die korrekte Erfassung des Inhalts eines Textes bis heute eine schwer zu lösende Aufgabe bei der maschinellen Verarbeitung gesprochener Sprache, einer Subdisziplin der künstlichen Intelligenz, dar. Im Zuge des Aufkommens erneuter Beliebtheit künstlicher neuronalen Netze in den letzten Jahren haben sich die Herangehensweisen dazu indes erheblich verbessert. So wurde beispielsweise die auf ihnen basierende Methodenklasse Word2vec entwickelt, die ein effizientes Vorgehen bei der Erstellung verteilter Repräsentationen von Wörtern ermöglicht[Mikolov u. a., 2013], welche den Vorteil gegenüber gängigen lokalisierten Repräsentationen bieten, dass bei ihnen semantische und syntaktische Beziehungen zwischen den Wörtern erhalten sind. Abhängig von der konkreten Anwendung können diese ebenso wie die Kombination der beiden Repräsentationsmethoden zu einer substantiellen Verbesserung bei der automatisierten Klassifizierung von Texten beitragen[Lilleberg u. a., 2015].

Mit Blick auf jene Veränderungen soll in dieser Bachelorarbeit untersucht werden, ob mit Hilfe der verteilten Wortrepräsentationen eine Textmining-Methode entworfen werden kann, welche Mittel bereit stellt, mit denen ein gewinnbringendes Vorgehen beim automatisierten Handel mit Aktien ermöglicht wird. Im Speziellen soll ein Klassifikator trainiert werden, der die Anwendung einer profitablen, primitiven Aktienhandelsstrategie und die zugehörige Long- oder Short-Investmentposition zu einer Nachricht und Aktie hinreichend zuverlässig vorhersagt. Dabei soll ebenso untersucht werden, ob hierbei die Verwendung von verteilten Wortrepräsentationen in der Form von nach Word2vec erstellten Word Embddings einen Vorteil bezüglich der Performanz des Klassifzierers gegenüber der klassischen lokalisierten Repräsentierungsmethode „Bag of Words" und der Kombination beider Ansätze bringt. Als Ausgangsdatensatz dient dazu ein Archiv von deutschen Ad-hoc-Mitteilungen mit historischen Kursdaten der zugehörigen Aktien von den Handelsplätzen Xetra und Börse Frankfurt. Dabei werden die Verwendung von Word Embeddings als Textrepräsentierungsmethode und das direkte Klassifizieren nach der maximal profitablen Strategie als Neuheiten eingeführt. Hierbei soll eine untere Schranke für das Potenzial des Textminings im Kontext der Ad-hoc-Publizität und des deutschen Aktienmarktes aufgezeigt werden, um so im besten Fall ein Licht auf den wachsenden Einfluss der künstlichen Intelligenz auf die Gesellschaft im Allgemeinen und den Börsenhandel im Speziellen zu werfen.

Zum Durchführen der Untersuchungen wurde neben dem Beschaffen der verwendeten Daten der Textmining-Prozess definiert. Für jede zu untersuchende Fragestellung wurden Klassifizierungsprobleme formuliert und die resultierenden Klassifizierer evaluiert, um anhand ihrer Performanz Rückschlüsse auf die Textmining-Methode zu ziehen.

Wie sich herausstellt, sind Word Embeddings in diesem Kontext aller Wahrscheinlichkeit nach nicht dazu geeignet, die Mitteilungen als Dokumentvektoren darzustellen. Obschon die Trefferfrequenz bei beim Vorhersagen der profitabelsten Strategie unzureichend ist, gelingt es trotzdem, einen Klassifikator zu trainieren, der im statistischen Mittel eine positive Rendite erwirtschaften könnte und dabei seine Benchmarks schlägt.

Das restliche Dokument gliedert sich wie folgt: Zunächst werden in Abschnitt 2 die Grundlegenden Konzepte des maschinellen Lernens und Textminings sowie der Teilnahme am Aktienmarkt erklärt. Daraufhin erfolgt in Abschnitt 3 die Einordnung dieser Arbeit in ihren wissenschaftlichen Kontext. Es schließt sich daran in den Sektionen 4 und 5 ein Überblick über die verwendeten Daten und die Vorstellung der Untersuchungsmethodik der Fragestellungen dieser Bachelorarbeit an, worauf in Abschnitt 6 das Vorgehen bei der Evaluierung und ihre Ergebnisse erörtert werden. Abschließend wird diese Arbeit in Sektion 7 ihren Grundzügen zusammengefasst und Anknüpfunkte an sie aufgezeigt.

2 Grundlegende Konzepte

Das Datenanalyseverfahren, welches für die im Rahmen dieser Bachelorarbeit getätigten Untersuchungen angewendet wurde, ist eine Variante des Textminings. Bei diesem werden in natürlicher Sprache vorliegende, unstrukturierte Textdaten zu einem zu untersuchenden Sachverhalt gesammelt und in Form strukturierter Daten aufbereitet. Die darin enthaltenen Informationen und Informationsstrukturen werden daraufhin mit Hilfe von Methoden der Statistik oder des maschinellen Lernens extrahiert und können dann bezüglich der Fragestellungen analysiert werden. Im Folgenden wird auf die Konzepte eingegangen, die dieser Arbeit zugrunde liegen. Dabei wird zunächst das maschinelle Lernen vorgestellt, dann eine Variante des Textminings beschrieben und schließlich eine kurze Einführung in die relevanten Zusammenhänge zur Teilnahme am Börsenhandel gegeben.

2.1 Maschinelles Lernen und Klassifizierungsprobleme

Das maschinelle Lernen ist eine Subdisziplin der künstlichen Intelligenz. Es befasst sich mit der automatisierten Deduktion von in Datenmengen implizit enthaltenen Gesetzmäßigkeiten, dem „Lernen", und ermöglicht so Systemverhalten, das auf komplexem Wissen aufbaut, welches nicht programmatisch vorgegeben ist, sondern mit Hilfe eines Datensatzes induziert wurde. Von zentraler Bedeutung ist dabei die Abstraktion der in den Daten erfassten Muster, sodass ein systematischer Transfer des erlernten Wissens zur Anwendung auf weitere, fremde Daten gleicher Struktur mit einem möglichst geringen Transferfehler durchgeführt werden kann. Es lassen sich drei allgemeine Kategorien des maschinellen Lernens unterscheiden:

1. **Überwachtes Lernen**
   Beim überwachten Lernen wird das System mit Elementen einer Relation gespeist, welche mit Informationen über ein zu lernendes Konzept, den sogenannten Labels, versehen sind. Diese Labels können formal selbst als Elemente einer Relation betrachtet werden. Mit Hilfe der Beispieldaten soll eine Funktion, die sogenannte Hypothese, definiert werden, welche die Relation der Labels als Zielbereich und die der Eingabedaten als Definitionsbereich hat. Die Abbildungsvorschrift der Hypothese soll dabei so gewählt werden, dass ihre Berechnungen denen der als „Konzept" bezeichneten, unbekannten Funktion mit selbem Definitions- und Zielbereich entsprechen, mit welcher den Eingabedaten die Labels zugeordnet wurden. Ist der Zielraum kontinuierlich, wird das behandelte Konstruktionsproblem als „Regression" bezeichnet. Ist er diskret, handelt es sich um eine Klassifikationsproblem.

2. **Unüberwachtes Lernen**
   Beim unüberwachten Lernen verarbeitet das lernende System ebenfalls Elemente einer Relation, welche aber nicht mit Labels augmentiert wurden. Ohne auf solche vorgegebenen Informationen zurückzugreifen, sollen dabei Zusammenhänge in den Eingabedaten erfasst und darstellbar werden. Anwendungsgebiete des unüberwachten Lernens sind neben weiteren das Clustering, bei dem die Datenpunkte unter den Annahme vorhandener Ähnlichkeitsstrukturen in Kategorien unterteilt werden, und die Dimensionsreduktion, welche zum Ziel hat, die Relation der Eingabedaten auf eine ihr entsprechende mit geringerer Stellenzahl abzubilden.

3. **Bestärkendes Lernen**
   Das bestärkende maschinelle Lernen spielt besonders in der Robotik eine wichtige Rolle und folgt konzeptionell seinem natürlichen Vorbild. Anhand von Trajektorien aus Ausgangszustand-Aktion-Zielzustand-Tripeln, wie sie etwa mit Hilfe eines Markov'schen Entscheidungsprozesses generiert werden können und an deren Ende eine Belohnung entsprechend dem Erfolg der konsekutiven Aktionsanwendungen ausgezahlt wird, soll eine Strategie erlernt werden, die zur optimalen Belohnung führt. Analog zur in der Pädagogik bekannten Maxime „Learning by Doing" geschieht dies häufig explorativ unter Bedingungen, die denen bei der Anwendung des Gelernten sehr ähnlich sind, und iterativ im Sinne des Lernens nach Versuch und Irrtum.

Die in dieser Bachelorarbeit verwendete Variante des Textminings beinhaltet Klassifizierungsprobleme, also solche aus der Kategorie des überwachten maschinellen Lernens. Daher soll im Folgenden genauer auf diese eingegangen werden.

2.1.1 Klassifizierungsprobleme

Klassifizierungsprobleme behandeln die Einordnung von Objekten anhand ihrer Merkmale in Kategorien. Wie eingangs bereits angedeutet, handelt es sich dabei im Kontext des maschinellen Lernens dabei um Konstruktionsprobleme, bei denen anhand eines Satzes Trainingsdaten eine Funktion, ein sogenannter „Klassifikator" oder „Klassifizierer", ermittelt, man spricht von „trainiert" oder „gelernt" wird. Diese Funktion soll die angesprochene Einordnung für ein festes Merkmal-Kategorie-Schema und alle möglichen Objekte nach diesem Schema vornehmen, was explizit auch solche Objekte einschließt, die nicht im Trainingsdatensatz enthalten sind. Dabei werden Objekte nach jenem Schema als „Instanzen", deren Merkmale als „Attribute" oder „Features", die Kategorien als „Label" und deren Ausprägungen als

„Klassen" bezeichnet. Die Wertebereiche der Attribute können diskret oder kontinuierlich numerisch wie auch symbolisch sein. Besteht daran Bedarf, etwa durch einschränkende Anforderungen des zum Lernen verwendeten Algorithmus', lassen sich symbolische Attribute auf numerische Abbilden. Ein Label kann je nach konkretem Lernszenario mehrere mögliche Ausprägungen annehmen. Probleme mit zwei solcher Ausprägungen werden „binäre Klassifizierungsprobleme" genannt. Abhängig von der Problemstellung können einer Instanz gleichzeitig mehrere Kategorien, also mehrere Labels, zugewiesen werden.

Formal betrachtet sind bei einem Klassifizerungsproblem mit $n \in \mathbb{N}$ Attributen mit Wertebereichen V_j mit $j \in \{1, ..., n\}$ und $m \in \mathbb{N}$ Labels mit Wertebereichen W_k mit $k \in \{1, ..., m\}$ die Menge aller theoretisch möglichen Instanzen $X := V_1 \times V_2 \times ... \times V_n$ und die Menge aller theoretisch möglichen Label $Y := W_1 \times W_2 \times ... \times W_m$. Die Mengen der tatsächlich möglichen Instanzen und die Menge der tatsächlich möglichen Label induzieren Relationen auf ihren oben genannten Obermengen.So lässt sich eine Instanz i als Zeilenvektor der Attribute $\mathbf{x^{(i)}} = (x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)})$ und das Klassifikationsziel, selbiger Instanz als Zeilenvektor der Labels $\mathbf{y^{(i)}} = (y_1^{(i)}, y_2^{(i)}, ..., y_m^{(i)})$ darstellen. Mit der Matrix der t Trainingsinstanzen $X_T \in X^t$ und zugehörigen Trainingslabels $Y_T \in Y^t$ ergibt sich der Trainingsdatensatz als erweiterte Matrix $D = (X_T | Y_T)$. Auf diesem soll nach der Problemstellung das implizit als Muster enthaltene Konzept $f : X \to Y$ mit $f(\mathbf{x^{(i)}}) = \mathbf{y^{(i)}}$ als Hypothese $h : X \to Y$ mit $h \sim f$ angenähert werden. Dies erfolgt durch den konkreten Lernalgorithmus, den „Lerner".

Zur Ermittelung der Klassifizierungsfunktion existieren modellbasierte und modellfreie Ansätze. Der grundlegende Unterschied zwischen diesen ist, dass im Zuge der Ausführung eines modellbasierten Ansatzes eine explizite Repräsentation der Hypothese erfolgt, während modellfreie Ansätze den Funktionswert einer zu klassifizierenden Instanz direkt aus dem Trainingsdatensatz ableiten, also speicherbasiert arbeiten. Prominente Vertreter der ersten Klasse sind die auch in Abschnitt 2.1.4 vorgestellten künstlichen Neuronalen Netzwerke, bei welchen Topologie, Knoten und Kantengewichte das Modell darstellen, Einer der bekanntesten modellfreien Ansätze hingegen ist kNN, bei dem anhand eines Distamzmaßes zwischen der zu klassifizierenden Instanz und denen im Trainingsdatensatz die Mehrheitsklasse der k nächsten Nachbarn vorhergesagt wird. Die hier verwendeten Verfahren zum Lösen der Klassifizierungsprobleme sind modellbasiert, weshalb die Aussagen im Folgenden nicht notwendigerweise auch auf modellfreie Ansätze zutreffen.

Das Erlernen des Modells kann „offline" am Stück auf einem statischen Lerndatensatz erfolgen oder auch „online", wobei das Modell auf inkrementelle Weise mit neuen Daten aktualisiert wird und entsprechend nicht alle Informationen über die Beispieldaten während der gesamten Trainingszeit zur Verfügung stehen. Da das System beim maschinellen Lösen von Klassifizierungsproblemen anhand der Trainingsbeispiele auf das Konzept generalisierend lernt, spricht man auch von „induktivem Lernen". Nach Ockhams Rasiermesser werden dabei weniger komplexe Modelle den komplexeren vorgezogen. Die Anwendung der gelernten Funktion auf eine Instanz zur Bestimmung ihrer Label wird häufig auch „Vorhersage" genannt. Durch sie lassen sich zukünftige Sachverhalte, deren Merkmale gegenwärtig bekannt sind, mit vom konkreten Szenario abhängiger Korrektheit bestimmen.

Da die in der Textmining-Methode dieser Bachelorarbeit verwendeten Klassifizierungsprobleme mit genau einem Label auskommen, wird hier weiter auch nur auf diesen Fall eingegangen.

2.1.2 Evaluierung

Ein wichtiger Arbeitsschritt bei der Entwicklung und Verwendung von Lernalgorithmen und Modellen ist die Evaluierung der Güte ihrer Vorhersagen. Beim überwachten maschinellen Lernen sind zwei Fehlertypen von zentraler Bedeutung, der Fehler durch Bias und der Fehler durch Varianz. Ersterer ergibt sich durch die zu starke Verallgemeinerung der Hypothese, was deshalb auch „Underfitting" oder „Unteranpassung" an den Trainingsdatensatz genannt wird. Dabei bleiben zu viele Eigenschaften der im Datensatz enthaltenen Muster unerfasst und werden entsprechend nicht bei der Anwendung auf vorherzusagende Instanzen miteinbezogen. Der Fehler durch Varianz entsteht aufgrund von „Overfitting", also durch Überanpassung an den Trainingsdatensatz, bei der das Gegenteil eintritt: Es werden zu viele Eigenschaften der im Datensatz enthaltenen Muster von der Hypothese erfasst und damit auch ihm möglicherweise inhärente Ungenauigkeiten und Fehlinformationen, wie sie etwa durch statistisches Rauschen, Ausreißer und Messfehler induziert werden. Oder es werden schlicht für das Konzept wenig relevante Konkretisierungen der Instanzen als es charakterisierende Kriterien miteinbezogen. Bias und Varianz verhalten sich tendenziell gegenläufig. Verringert man etwa durch die gezielte Parameterwahl beim Training den Bias, so vergrößert sich die Varianz. Verringert man die Varianz, so vergrößert sich der Bias. Das Finden eines optimalen Kompromisses zwischen den beiden ist Teil des Entwurfs vieler Lernalgorithmen oder deren Parametrisierung.

Da das Volumen eines nicht-trivialen Vektorraumes exponentiell mit dessen Dimensionen steigt, wächst auch die Anzahl der beim Klassifizieren für statistisch signifikante Muster notwendigen Instanzen rapide mit der Attributanzahl. Diesem als „Fluch der Dimensionalität" bekannten Umstand und der Tatsache, dass eine Vielzahl an Daten ein Muster tendenziell besser darstellen kann als wenige dazu imstande sind, ist es geschuldet, dass es beim maschinellen Lernen generell im Sinne der Aussagekraft des Gelernten wünschenswert ist, über möglichst viele Trainingsdaten zu verfügen.

Die Einschätzung der Güte von Klassifikationsansäzen bezüglich der Korrektheit der durch sie getätigten Vorhersagen kann auf verschiedene Weisen erfolgen. Um diese Performanz zu quantifizieren, existiert eine Vielzahl an Evaluations-

metriken und Performanzmaßen. Im Folgenden werden zwei gängige Evaluierungsmethoden und vier Performanzmaße vorgestellt, welche auf im Kontext von Klassifizierungsproblemen mit genau einem Label anwendbar sind.

Evaluierungsmethoden

Die beiden prominentesten Verfahren zur Einschätzung der Performanz von Klassifikatoren bezüglich deren Korrektheit sind die Kreuzvalidierung und die Evaluation auf einem Testdatensatz. Die Performanz eines Modell wird allgemein nicht auf dessen Trainingsdatensatz evaluiert, da dabei der Fehler durch Varianz nicht erfasst werden kann und die Performanzmaße infolge dessen optimistische Ergebnisse liefern.

1. **Evaluation auf einem Testdatensatz**
   Die Evaluation auf einem Testdatensatz ist die klassische Vorgehensweise beim Evaluieren eines Klassifikators. Sie entspricht einer Simulation des praktischen Einsatzes. Dabei wird das Modell gegebenenfalls zunächst auf einem Trainingsdatensatz angelernt und dann mit Hilfe der Instanzen im Testdatensatz evaluiert. Diese sind, wie die Trainingsdaten auch, mit korrekten Klassifizierungen versehen und stellen den sogenannten „Goldstandard" oder „Ground Truth". Sie werden zum Vergleich mit den Klassifizierungsaussagen des Lerners verwendet. Dieser wird zu den jeweiligen Testinstanzen konsultiert und seine Aussagen mit der Ground Truth verglichen. Die Ergebnisse der Fehlermaße berechnen sich dann aus den Übereinstimmungen und Diskrepanzen bei den einzelnen Vergleichen. Bei hinreichenden Datensatzgrößen ist diese Methode effektiv, jedoch werden Instanzen zur Evaluierung vorgehalten, die insbesondere bei Datenknappheit einen teuren Verlust für das Training darstellen.

2. **Kreuzvalidierung**
   Die Kreuzvalidierung kommt hingegen ohne einen dedizierten Testdatensatz aus. Dazu wird der Trainingsdatensatz in eine vorher festgelegte Anzahl an $n \in \mathbb{N}$ möglichst gleich große, disjunkte Teilmengen unterteilt. Daraufhin erfolgen n einzelne Evaluationsdurchläufe, bei denen der Verbund aus $n-1$ der Teilmengen als Trainingsdatensatz und die verbleibende als Testdatensatz für das jeweils gelernte Modell fungieren. Die Aufteilung der Untermengen über die n Iterationen erfolgt so, dass jede einzelne Menge genau einmal die Funktion des Testdatensatzes einnimmt. Das Vorgehen entspricht dabei der n-fachen Evaluation des Lernalgorithmus' mit einem Testdatensatz. Ebenfalls analog dazu wird zum Bestimmen der Evaluationsmetriken die Menge der Vorhersage-Goldstandard-Paare ausgewertet. Generell erfordert die Kreuzvalidierung eines modellbasierten Klassifizierungsansatzes auf einem Datensatz mehr Rechenzeit als dessen einfache Evaluation mittels Testdatensatz gleichen Umfangs, da zu den n Untermengen insgesamt n Modelle trainiert werden müssen anstatt nur eines. Dafür benötigt diese Methode erheblich weniger Daten.

Evaluationsmetriken

Die hier vorgestellten Evaluationsmetriken dienen zur Quantifizierung der Performanz von Klassifizierern. Mit ihnen lassen sich in Abhängigkeit von Datensatzbeschaffenheit und Parametrisierung der Lernalgorithmen auch Rückschlüsse über die Tauglichkeit der zum Lernen verwendeten Verfahren ziehen. Im Kontext dieser Bachelorarbeit werden sie verwendet, um das Gespann aus den gesammelten Daten und der verwendeten Textmining-Methode anhand der Performanz ihrer Klassifizierer zu bewerten.

Viele der Gütemaße für die Klassifikation mit einem Label lassen sich aus einer Kontingenztafel berechnen. Dabei handelt es sich um eine Matrix, deren Zellen die absoluten oder relativen Häufigkeiten der Paare bestehend aus Urteil des Klassifizierers und Goldstandard enthalten. Hier behandelt werden Kontingenztafeln mit absoluten Häufigkeiten, deren Zeilen den Klassen der Ground Truth und Spalten denen der Vorhersagen entsprechen. Dabei ist jede Zeile und jede Spalte genau einer Klasse zugeordnet und jede Klasse genau einer Zeile und genau einer Spalte und alle Klassen vertreten. Die Reihenfolge der Klassen in den Zeilen ist gleich der in den Spalten, sodass die Zellen der Hauptdiagonalen die Anzahl der korrekt klassifizierten Instanzen für die jeweilige Klasse beinhalten. Ein Beispiel ist in Tabelle 2.1.2 zu finden. Es ergibt sich formal die folgende Definition.

Definition 2.1 (Kontingenztafel) *Sei C die Menge der Klassen des Klassifizierungsproblems. Die Abbildung $p : C \times C \to \mathbb{N}$ modelliere mittels $p(c_i, c_j) = n_{i,j}$ für alle Klassen $c_i, c_j \in C$, wie oft eine Instanz der Klasse c_i als der Klasse c_j zugehörig klassifiziert wurde. Dann ist die Kontingenztafel eine Matrix $K \in \mathbb{N}^{|C|\times|C|}$ für deren Zellen $k_{i,j}$ in Zeile i und Spalte j gilt, dass $k_{i,j} = p(c_i, c_j)$.*

Ein erwähnenswerter Spezialfall der Kontingenztafel ist die Konfusionsmatrix, welche einer Kontingenztafel für ein Zweiklassenproblem entspricht. Diese binären Klassifizierungsprobleme behandeln häufig nur die Fragestellung, ob die Instanzen zu einer speziellen Kategorie gehören, also als „positiv" klassifiziert werden, oder nicht, also „negativ" sind. Anderenfalls lassen sich die beiden Klassen auf „positiv" und „negativ" abbilden, sodass sich die folgenden vier Einordnungen für Klassifikationsaussagen identifizieren lassen.

1. *tp*, die *„true positives"*, also die Anzahl der Instanzen, die korrekterweise als „positiv" klassifiziert wurden

	PKW	LKW	Motorrad
PKW	6	2	1
LKW	2	6	1
Motorrad	1	2	6

Tabelle 1: Beispiel für eine Kontingenztafel

tp	*fp*
fn	*tn*

Tabelle 2: Beispiel für eine Konfusionsmatrix

2. *fp*, die *„false positives"*, also die Anzahl der Instanzen, die fälschlicherweise als „positiv" klassifiziert wurden
3. *tn*, die *„true negatives"*, also die Anzahl der Instanzen, die korrekterweise als „negativ" klassifiziert wurden
4. *fn*, die *„false negatives"*, also die Anzahl der Instanzen, die fälschlicherweise als „negativ" klassifiziert wurden

Mit dieser Notation ergibt sich das Schema der Konfusionsmatrix als das in 2.1.2 abzulesende.

Einige Fehlermaße sind in ihrer ursprünglichen Fassung nur für binäre Klassifizierungsprobleme definiert. Aus ihnen lassen sich jedoch entsprechende Evaluationsmetriken für Klassifizierungsprobleme mit mehr als zwei Klassen ableiten. Dazu werden die *tp*, *fp*, *tn* und *fn* einmal pro Klasse berechnet, wobei die Klasse selbst als „positiv" und die Klasse, die sich aus dem Verbund der verbleibenden ergibt, als „negativ" betrachtet wird. Zum Berechnen der Metrik für das Gesamtproblem gibt es zwei Vorgehensweisen, das Micro-Averaging und das Macro-Averaging. Beim Macro-Averaging wird das Maß für jede der Klassen berechnet und dann der Durchschnitt über die Ergebnisse gebildet. Das Micro-Averaging erfolgt durch die Aufsummierung der *tp*, *fp*, *tn* und *fn* über alle Klassen und anschließender Berechnung des Maßes aus den Summenwerten.

Definition 2.2 (Micro-Averaging, Macro-Averaging) *Sei $m : \mathbb{N}^4 \to \mathbb{R}$ ein Fehlermaß, das aus geordneten Tupeln von (tp, fp, tn, fn) berechnet wird, C die Menge der Klassen des Singlelabel-Klassifizierungsproblems und die tp_i, fp_i, tn_i, fn_i seien die Klassifizierungsfehlertypen der Klasse $c_i \in C$.*

Dann ist das durch Micro-Averaging gebildete Fehlermaß für das Gesamtproblem definiert als

$$m_{micro} := m(\sum_{i=1}^{|C|} tp_i, \sum_{i=1}^{|C|} fp_i, \sum_{i=1}^{|C|} tn_i, \sum_{i=1}^{|C|} fn_i)$$

Und das durch Macro-Averaging gebildete ist

$$m_{macro} := \frac{1}{|C|} \sum_{i=1}^{|C|} m(tp_i, fp_i, tn_i, fn_i)$$

Es folgt nun die Vorstellung der Evaluationsmetriken, die beim Lösen der Klassifizierungsprobleme dieser Arbeit verwendet wurden. Diese sind „Accuracy", „Precision", „Recall" und der „F1-Score". Letztere drei existieren sowohl in einer Micro- als auch einer Macro-Variante.

Definition 2.3 (Accuracy) *Sei $M \in \mathbb{N}^{n \times n}$ eine Kontingenztafel mit $n \in \mathbb{N}$ Klassen. Dann ist die Accuracy definiert durch*

$$Accuracy := \frac{\sum_{i=1}^{n} M_{i,i}}{\sum_{i=1,j=1}^{n} M_{i,j}}$$

Die Accuracy misst den relativen Anteil der korrekt eingeordneten unter den insgesamt klassifizierten Instanzen. Dementsprechend steht der maximale Accuracy-Wert von 1 für eine perfekte Klassifikationsperformanz und der Minimalwert 0 für die schlechtestmögliche.

Definition 2.4 (Precision) *Sei C die Menge der Klasse und tp_i und fp_i die bereits bekannten Größen zu der Klasse $c_i \in C$, dann ist die Precision für diese Klasse definiert als*

$$Precision_i := \frac{tp_i}{tp_i + fp_i}$$

Die Precision gibt den Anteil der korrekt klassifizierten Instanzen einer Klasse an der Anzahl aller Instanzen, die durch den Lerner dieser Klasse zugehörig eingeordnet wurden, wider. Folglich ist ein Wert von 0 minimal und ein Wert von 1 sowohl maximal als auch optimal.

Definition 2.5 (Recall) *Sei C die Menge der Klasse und tp_i und fn_i die bereits bekannten Größen zu der Klasse $c_i \in C$, dann ist der Recall für diese Klasse definiert als*

$$Recall_i := \frac{tp_i}{tp_i + fn_i}$$

Der Recall gibt das Maß für den Anteil der korrekt klassifizierten Instanzen einer Klasse unter allen tatsächlichen Instanzen jener Klasse an. Auch er hat sein Optimum bei 1, welches auch sein Maximum ist, und das Minimum bei 0.

Definition 2.6 (F1) *Seien $Precision_i$ und $Recall_i$ die bereits definierten Fehlermaße für die Klasse $c_i \in C$*

$$F_1^{(i)} := 2 * \frac{Precision_i * Recall_i}{Precision_i + Recall_i}$$

Das Effektivitätsmaß F1-Score ist das gewichtete harmonische Mittel aus Precision und Recall. Es erlaubt die Quantifizierung der Güte eines Klassifikators mit Hilfe der beiden Größen in nur einer Zahl.

2.1.3 Hyperparameter-Tuning

Im Kontext von Klassifizierungsproblemen gibt es verschiedene Parametertypen. Darunter sind die im Zuge des Lernens zu findenden Modellparameter, aber auch der Lernalgorithmus selbst wird, so sie gegeben sind, durch sogenannte Hyperparameter im Verhalten beeinflusst, was sich mittelbar auf die Performanz der resultierenden Klassifizierer auswirkt. Für beide Parametertypen existieren in der Regel Optimalwerte bezüglich der Performanz des resultierenden Lerners, die vom verwendeten Datensatz und dem zu lernenden Konzept abhängen. Während die Modellparameter im Allgemeinen durch den Lernprozess optimiert werden, ist das Finden guter Hyperparameter durch einen zusätzlichen Suchvorgang, das Hyperparameter-Tuning, vorzunehmen. Dazu gibt es verschiedene Herangehensweisen. Eine solche ist die Grid Search. Diese wird durchgeführt, indem für jeden Hyperparameter ein Wertebereich spezifiziert wird. Der Suchraum ergibt sich aus den Permutationen aller Werte der spezifizierten Bereiche. Die Suche selbst erfolgt durch konsekutives Trainieren und Evaluieren. Gefunden werden soll dabei die Konfiguration der Hyperparameter im Suchraum, für welche die Evaluationsmetrik, die „Target-Variable", den maximalen Wert annimmt. Neben der Grid Search können ebenso weitere Suchstrategien wie die Random Search oder allgemeine Suchalgorithmen zum Hyperparameter-Tuning eingesetzt werden.

2.1.4 Klassifizierungsalgorithmen

Im Folgenden wird auf zwei wichtige Vertreter von Methoden des modellbasierten, überwachten maschinellen Lernens eingegangen, die Support Vector Machine und künstliche neuronale Netze. Beide sind zum Lösen von Klassifizierungsproblemen geeignet und etablierte Verfahren, die für diese Bachelorarbeit von Relevanz sind.

SVMs

Als „Support Vector Machine" (SVM) werden unterschiedliche Repräsentanten eines Typs modellbasierter Verfahren des überwachten maschinellen Lernens bezeichnet. Diese haben als grundlegendes Prinzip die Verwendung einer zwischen den Klassen diskriminierenden Grenze im Instanzraum gemeinsam, die durch anhand der Trainingsdaten ermittelten Stützvektoren so bestimmt wird, dass sie die Klassen maximal voneinander trennt. Dank dieser Herangehensweise eignen sich SVMs gut zum Lösen von Klassifizierungsproblemen mit vielen Attributen, also hochdimensionalen Instanzräumen, dünnbesetzten Beispieldaten, bei denen viele Attributwerte einem Standardwert entsprechen, oder großen Trainingsdatensätzen mit vielen Instanzen. Daher bieten sie sich für die automatisierte optische Buchstabenerkennung wie auch zur Klassifizierung von als Vektoren repräsentierten Texten an[Joachims, 1998]. Es existiert eine Vielzahl konkreter Implementierungen von Klassifizierern auf der Basis der Support Vector Machine. Die folgenden Ausführungen beziehen sich auf Implementierungen von SVMs aus der quelloffenen Bibliothek LIBLINEAR[Fan u. a., 2008].

Der eingangs erwähnte Instanzraum ist ein Vektorraum, der anhand des Attributschemas so konstruiert wird, dass jedes numerische Attribut eine Dimension in ihm darstellt, und jedes symbolische Attribut mit n Ausprägungen auf n binäre Attribute abgebildet wird, welche ebenfalls als Dimensionen betrachtet werden. Diese nehmen für ein symbolisches Attribut entsprechend einer One-Hot-Codierung genau dann den Wert 1 an, wenn die jeweilige passende Ausprägung ausgedrückt werden soll, und 0 sonst. Die Lernaufgabe von zur Klassifizierung eingesetzten SVMs aus der LIBLINEAR-Bibliothek besteht darin, in jenem Vektorraum eine Hyperebene zu definieren, die als Grenze zwischen den Klassen der Datenpunkte im Raum fungiert und die Klassen so voneinander abgrenzt, dass der Abstand zwischen der Hyperebene und dem nächsten Vertreter einer der Klassen nach einer Norm maximal ist.

Die hier verwendete SVM aus der LIBLINEAR-Bibliothek wird durch das Lösen des Optimierungsproblems

$$min_{\omega} \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{l} max(0, 1 - y_i\omega^T\mathbf{x}_i)^2$$

mit dem Modellparameter ω, der Menge der Trainingsinstanzen l, deren Featurevektoren $\mathbf{x}_i$ und zugehörigen Labels y_i trainiert. Dabei ist der Penalty-Parameter C, der den Einfluss der Trainingsdaten aufs Modell reguliert, ein Hyperparameter, der den Trainingsprozess an sich beeinflusst. Die Anwendung eines Modells auf eine Instanz $\mathbf{x}$ erfolgt durch die Auswertung von

$$\omega^T\mathbf{x} > 0$$

Ist die Aussage wahr, wird sie als „positiv" klassifiziert. Ist er falsch als „negativ".

Die Verwendung einer Hyperebene als Grenzformalismus impliziert zwei Herausforderungen. Zum Einen bedeutet die lineare Trennung, dass der Klassifizierer nur zwischen diesseits und jenseits der Hyperebene unterscheiden kann. Dieser Einschränkung wird durch eine Klassenbinarisierung beigekommen. Die Implementierung der LIBLINEAR-Bibliothek verwendet dazu die „One-Against-All"-Methode. Durch sie wird ein Klassifizierungsproblem mit $n > 2$ Klassen auf n binäre Probleme abgebildet, die für jeweils eine der Klassen die Frage nach der Zugehörigkeit einer Instanz zu ihr mit „positiv" oder „negativ" beantworten. Im Optimalfall sagt bei der Anwendung des aus n Klassifikatoren bestehenden Modells nur eine der Klassen „positiv" voraus und alle anderen „negativ". Klassifizieren mehrere der einzelnen binären Klassifizierer die Instanz als „positiv", kommen Heurstiken als Tiebreaker zum Einsatz. Wird die Instanz von allen binären Klassifizierern als „negativ" klassifiziert, wird ebenso verfahren, um eine geeignete Klasse zu finden.

Zum Anderen müssen die Klassen im Raum der Trainingsinstanzen linear trennbar sein. Dies ist bei hochdimensionalen Problemen deutlich häufiger der Fall als bei niedrigdimensionalen. Die Modellfindung kann zudem durch das Verwenden von Toleranzen erleichtert werden. Trotzdem ist die lineare Trennbarkeit nicht immer gegeben. In diesem Fall kann kein den in der Praxis relevanten Anforderungen genügendes Modell erstellt werden.

Künstliche neuronale Netze

Künstliche neuronale Netze (ANN) beschreiben ein Ansatz zur Modellbildung beim maschinellen Lernen, dem gerichtete Graphen zugrunde liegen. ANN sind, stark vereinfacht, den natürlichen neuronalen Netzen nachempfunden, wie sie etwa im zentralen Nervensystem eines Säugetiers zu finden sind. Die Neuronen werden dabei durch die Knoten eines Graphen repräsentiert. Die Eingangskanten fungieren als Dentriten und die Ausgangskanten als Axone. In einem künstlichen neuronalen Netz existieren Eingabeknoten, über die Informationen ins Netzwerk eingespeist werden, Ausgabeknoten, welche die Ergebnisse der Berechnungen repräsentieren, und gegebenenfalls Zwischenknoten, die bei der Transformation der Daten im Rahmen der Berechnungen helfen.

Die Knoten sind je nach Ausgestaltung des Netzes in einer speziellen Topologie miteinander verwoben, wobei die Existenz von Zyklen und der längste einfache Pfad durchs Netzwerk von besonderer Bedeutung sind. Ist der Graph des neuronalen Netzes azyklisch, spricht man von einem vorwärtsgerichteten Netz. Ist er zyklisch, wird das Netz als „rekurrent" bezeichnet und ermöglicht durch Rückkoppelung der eingespeisten Informationen dynamisches Verhalten.Häufig werden die künstlichen Neuronen in den Netzwerken geschichtet angeordnet. Dabei sind die Eingabeknoten und Ausgabeknoten jeweils in einer Schicht organisiert, während die Schichten dazwischen, die sogenannten „Hidden Layer", jeweils über Äquivalenzklassen der Pfadlänge durch das Netz gruppiert werden.

Die Struktur eines ANNs hat einen starken Einfluss auf seine Eignung für spezielle Aufgaben des maschinellen Lernens. Da es berechenbarkeitstheoretisch äquivalent zur Turingmaschine ist, stellt das generelle Konzept des künstlichen Neuronalen Netzes keine Einschränkungen an die Anwendbarkeit auf die verschiedenen Problemstellungen des maschinellen Lernens. Abhängig von Neuronenmodell-, Konfiguration und Netztopologie lässt sich mit ihnen jede mathematische Funktion darstellen. Das Training des Modells erfolgt in aller Regel durch die Minimierung einer Kostenfunktion auf den Eingabedaten durch Anpassen der das Modell parametrisierenden Gewichte.

Zur Veranschaulichung wird im Folgenden beispielhaft ein vorwärtsgerichtetes neuronales Netz mit einer Hidden Layer für die Klassifikation mit einem Label betrachtet. Dieses verwendet das im Folgenden beschriebene McCulloch-Pitts-Neuronenmodell. Eines dieser Neuronen mit Bezeichner $i \in \mathbb{N}$ konsumiere den Vektor $\mathbf{a} = (a_0, a_1, ..., a_n)^T \in \mathbb{R}^n$ n verschiedener Eingangsvariablen und einer Schwellenwertkonstante $a_0 = -1$ mit den zugehörigen Gewichten $\mathbf{W_i} = (W_{i0}, W_{i1}, ..., W_{in})^T \in \mathbb{R}^n$, wobei W_ij für alle $j \in \mathbb{N} \cup \{0\}$ mit $j < n$ das Gewicht zum Eingang a_j darstellt und das n für alle Neuronen einer Schicht gleich sei. Die Kombination der Eingangswerte erfolgt durch die Eingabefunktion mittels gewichteter Summenbildung zu $in_i = \sum_{j=0}^{n} w_{ij}a_j$. Der Ausgangswert o_i eines Neurons wird durch die Aktivierungsfunktion $g : \mathbb{R} \to \mathbb{R}$ anhand des ermittelten in_i berechnet. Es bieten sich zum Codieren der Vorhersage einer Klasse anhand der Ausgangsknoten verschiedene Schemata an. Eines ist die feste Zuweisung eines jeden Knoten an genau eine Klasse, ohne dass es dabei zu wiederholten Zuweisungen an eine solche kommt. Ein Vorhersagevektor kann somit als one-hot-codierte Konkatenation der einzelnen Werte nach einem festen Schema verstanden werden. Mögliche Auswertungsvorschriften

für die Aktivierungsfunktionen sind unter anderen die Identität $id(in_i) = in_i$, mit der der Ausgang der Linearkombination der Eingänge entspricht, eine Sigmoid-Funktion $sig_a(in_i) = (1 + exp(-a * in_i))^{-1}$, welche das Kombinat der Eingangswerte mit einer durch den Parameter a bestimmten Steigung nichtlinear auf das Intervall $[0, 1]$ abbildet, oder eine Schwellenwertfunktion $hlim_v(in_i)$, die genau dann zu 1 auswertet, wenn $v \geq in_i$ und 0 sonst. Letztere beschert dem Neuron ein Aktivierungsverhalten nach dem „Alles-oder-nichts-Gesetz" seines natürlichen Vorbilds und erleichtert die vorgestellte Codierungsmethode für die Klassen durch die Ausgabeknoten Das Neuron mit Index i berechnet also den Wert seines Ausgangs $o_i = g(in_i) = g(\sum_{j=0}^{n} w_{ij} a_j)$.

Mit dem Verständnis des künstlichen Neurons lässt sich das beschriebene Netz nach dem aufbauen, das d Eingangswerte, c Ausgabeknoten und h Neuronen in der Hidden Layer hat. Dazu seien die d Eingabewerte des Netzes gegeben durch $\mathbf{x} = (x_0, x_1, ..., x_d)^T \in \mathbb{R}^d$, mit $x_0 = -1$. Die c Ausgabewerte seien beschrieben durch $y_1(\mathbf{x}), ..., y_c(\mathbf{x})$ und die Aktivierungsfunktion habe die Signatur $g : \mathbb{R} \to \mathbb{R}$. Die Gewichte der Eingangskanten der Hidden Layer mit h Knoten seien durch $W_{10}^{(1)}, ..., W_{hd}^{(1)}$ bezeichnet und die der Eingangskanten der Output Layer mit den c Ausgabeknoten als $W_{10}^{(2)}, ..., W_{ch}^{(2)}$, wobei ein Gewicht W_{ij} zur Kante von Knoten j einer Schicht zum Knoten i der nächsten Schicht gehöre und alle möglichen Kanten zwischen zwei aufeinanderfolgenden Schichten gezogen seien. Die Ausgangsfunktion des Gesamtnetzes, also die Hypothese, ergibt sich damit als $h(\mathbf{x}) = \mathbf{y} = (y_1(\mathbf{x}), ..., y_c(\mathbf{x}))^T$ und mit $k \in 1, ..., c$ ist

$$y_k(\mathbf{x}) = g^{(2)}\Big(\sum_{i=0}^{h} W_{ki}^{(2)} g^{(1)}(\sum_{j=0}^{d} W_{ij}^{(1)} x_j)\Big)$$

oder kompakt als

$$y_k(\mathbf{x}) = g^{(2)}\big(in_k^{(2)}\big)$$

mit

$$in_k^{(2)} = \sum_{i=0}^{h} W_{ki}^{(2)} g^{(1)}(in_i^{(1)})$$

und

$$in_i^{(1)} = \sum_{j=0}^{d} W_{ij}^{(1)} x_j$$

Zum Trainieren des Modells kann der Backpropagation-Algorithmus verwendet werden. Dies geschieht durch die Anpassung der Gewichte bei festgelegter Netztopologie. Dazu wird das neuronale Netz mit zufälligen Gewichten initiiert und so lange über den Trainingsdatensatz iteriert, bis die Hypothese $h_v(\mathbf{x})$ nach $v \in \mathbb{N}$ Anpassungen dem durch den Datensatz abgebildeten Konzept $f(\mathbf{x})$ so genau entspricht, dass keine weitere Gewichtsveränderung vonnöten ist. Die einzelnen Gewichtsaktualisierungen sind anhand einer Regel bestimmbar, welche sich aus der Minimierung einer Fehlerfunktion, hier beispielhaft der „Squared Error" $E(\mathbf{x}) = \frac{1}{2}(f(\mathbf{x}) - h(\mathbf{x})) * (f(\mathbf{x}) - h(\mathbf{x}))^T$, ergibt. Setzt man die Auswertungsvorschrift des neuronalen Netzes in die Fehlerformel ein, so fällt auf, dass die resultierende Funktion nicht konvex und somit nicht nur unter größerem Aufwand zu optimieren ist. Nicht-konvexe Fehlerlandschaften sind eine zentrale Schwierigkeit beim Training künstlicher neuronaler Netze, welche besonders bei komplexeren Topologien in Erscheinung tritt. Zudem sind die gelernten Modelle nur schwer Interpretierbar, da das Verhalten der Netze in ihre Struktur und Gewichte kodiert und somit nicht ohne Weiteres ablesbar ist. Aufgrund ihrer vielseitigen Anwendungsmöglichkeiten und Ausdrucksmächtigkeit sowie der guten Skalier- und Parallelisierbarkeit stellen künstliche neuronale Netze jedoch einen häufig verwendeten Ansatz zum Lösen von Problemen aus der Domäne des maschinellen Lernens dar. Dem interessierten Leser, der sich genauer über den Backpropagation-Algorithmus informieren möchte, sei die Lektüre von [Li u. a., 2012] empfohlen.

2.2 Textmining

Unter Textmining wird die algorithmische Extraktion von Wissen aus schwach- oder unstrukturierten textuellen Daten wie zum Beispiel einer Sammlung in gesprochener Sprache verfasster Dokumente verstanden. Diese kann je nach Analyseziel unter Anderem das Auffinden als existent bekannter Information aus einer Dokumentsammlung wie auch das Clustering von Dokumenten oder die Extraktion vermuteter semantischer Strukturen, Querverbindungen und Muster umfassen. Typische Anwendungsfelder des Textminings sind Betrugsdetektoren für Versicherungen, Spam-Filter und Contextual Advertising, welches es Werbetreibenden ermöglicht, automatisch inhaltlich auf die jeweiligen Webseiten angepasste Werbung anzuzeigen. Zweck des Textmining im Kontext dieser Bachelorarbeit ist die Extraktion von Signalen zum Anwenden bestimmter Handelsstrategien aus Nachrichten zu Aktien. Die Arbeitsschritte eines solchen Textmining-Prozesses, lassen sich unterteilen in die analytische Textaufbereitung, das Feature-Engineering und die Musterextraktion, woraufhin das Ausnutzen und Präsentieren der Analyseergebnisse erfolgen kann. Anhand der drei hervorgehobenen Schritte soll nun das Konzept des Textminings mit in natürlicher Sprache verfassten Dokumenten verdeutlicht werden.

2.2.1 Analytische Textaufbereitung

Die Textaufbereitung beschreibt das Herausarbeiten von syntaktischen und semantischen Elementen, sodass die linguistischen Eigenschaften der Texte weiterverarbeitet werden können. Zuerst wird hierzu die in Format und Darstellung möglicherweise heterogene Dokumentsammlung auf der Basis von Metadaten transformiert und gefiltert. Sollen nur Dokumente, die in einer bestimmten Sprache verfasst sind, weiterverarbeitet werden, so sind alle anderen aus der Sammlung zu entfernen. Zum Herausfiltern ungeeigneter Texte können weitere Ausschlusskriterien, etwa solche auf Basis der Textlänge zum Einsatz kommen. Dies kann dabei helfen, Ausreißern im Datensatz vorzubeugen, was die Qualität der extrahierten Muster begünstigt. Nach dem Vereinheitlichen und Filtern der Dokumentsammlung erfolgt die computerlinguistische Verarbeitung der Dokumente mittels Methoden des Natural Language Processings. Um stets auf Zwischenergebnisse und Ausgangsrepräsentationen zurückgreifen zu können, und Informationen, die auf verschiedenen Repräsentationen des selben Textes beruhen, extrahieren zu können, werden Transformationen häufig nicht direkt am Dokument vorgenommen, sondern als Annotationen zusammen mit den sprachwissenschaftlichen Analyseergebnissen am Objekt vermerkt. Die computerlinguistische Verarbeitung kann selbst wieder in vier Schritte unterteilt werden, die linguistische Vorverarbeitung und die Analysen von Morphologie, Syntax und Semantik.

Linguistische Vorverarbeitung

Bei der linguistischen Vorverarbeitung werden einfache Sprachliche Strukturen in der Zeichenkette des Textes eines Dokuments erfasst und ohne detaillierte Kenntnisse von Syntax oder Semantik der zugrundeliegenden Sprache Textelemente herausgestellt. Konkret soll hier auf das Tokenizing, Sentence-Splitting und Stopword-Tagging eingegangen werden.

- **Tokenizing**
  Das Tokenizing, verdeutscht auch „Tokenisierung" genannt, ist ein fundamentaler Arbeitsschritt bei der computerlinguistischen Verarbeitung eines Textes. Ähnlich der Tokenisierung beim Vorgang des Kompilierens eines Programms in einer Programmiersprache wird der Ausgangstext dabei in zusammenhängende Sinneinheiten, die sogenannten Tokens, segmentiert. Bei der Verarbeitung gesprochener Sprache wie etwa der deutschen sind das in der Regel Wörter und gegebenenfalls Satzzeichen. Das Tokenizing kann nach statischen Regeln erfolgen, diese sind jedoch stark an die zugrundeliegende Sprache gekoppelt und in der Regel nicht trivial. Als Beispiel sei der aus der Zeichenkette „Karl-Heinz steht seit 1995 voll auf Rock'n'Roll." gewonnener, den Sinneinheiten angemessener Tokenstrom „<Karl-Heinz><steht><seit><1995><voll><auf><Rock'n'Roll><.>" angeführt.

- **Sentence-Splitting**
  Das Sentence-Splitting ist ein Spezialfall des Tokenizing, bei dem die zu findenden Sinneinheiten Sätze sind. Es wird häufig mit der Tokenisierung von Wörtern kombiniert, um die automatisierte Erkennung von Wortarten in Sätzen, etwa unter Zuhilfenahme der zugrunde liegenden Grammatik, zu ermöglichen.

- **Stopword-Tagging**
  Bei der Erfassung des Inhalts eines Textes sind viele Wörter wenig relevant, da sie selbst keine Semantik tragen oder lediglich andere Wörter inhaltlich unterstützen. So erfüllen beispielsweise „das", „am", „auf", oder „schließlich" hauptsächlich syntaktische Funktionen, die für den Gedankenfluss des Lesers aber nicht für die Aussage des Gelesenen von Relevanz sind. Dahingegen geben Wörter wie „Pleite", „erfüllt", „Gewinnsteigerung" oder „zwangsversteigern" dem Text Bedeutung und können so als charakteristische Merkmale dienen. Beim Stopword-Tagging werden die semantisch wenig relevanten Wörter, die sogenannten „Stoppwörter", mit Hilfe eines Wörterbuchs in einem Tokenstrom identifiziert und als solche gekennzeichnet. So wird gewährleistet, dass sie bei Bedarf später ignoriert werden können.

Morphologische Analyse

Die morphologische Analyse beschäftigt sich mit der Form und dem Aufbau von Wörtern. Morpheme bilden dabei die kleinste sinntragende Einheit und stellen die Komponenten für ein Wort. So ist beispielsweise das Wort „Ausbildung" aus den Morphemen „Aus", „bild", und „ung" aufgebaut. Häufig verwendete morphologische Analysemethoden sind das Stemming und die Lemmatisierung. Deren Zweck ist es, die durch einen Tokenizer identifizierten Wörter auf eine ihre Wortfamilie repräsentierende, kanonische Form zu bringen. Diese Äquivalenzklassen erhalten so, gemessen an ihrer Erscheinungsfrequenz, ein größeres Gewicht im Text, welches Rückschlüsse auf deren Relevanz für das untersuchte Dokument zulässt. Die Lemmatisierung versucht die Rückführung der Wörter auf deren linguistische Wortstämme, nach Möglichkeit auf ihre Wörterbuchform. Beim Stemming werden die Grundformen für die Äquivalenzklassen nach Bedarf anhand einheitlicher Regeln gebildet. Sowohl die Lemmatisierung als auch das Stemming können zur Aussagekraft wortfrequenzbasierter Textauswertungsverfahren beitragen. Wenn beispielsweise in einer Textmenge die am häufigsten erwähnte Art der Fortbewegung zu finden ist, ist es sinnvoll, Wörter wie „gehen", „herausgegangen" und „gegangen" auf einen gemeinsamen Vertreter abzubilden und dessen Erscheinungen zu zählen. In anderen Anwendungsfällen kann dieses Vorgehen allerdings ungewollte Gleichbehandlungen von Wörtern induzieren, da zum Beispiel „gerecht" und „ungerecht" durch die Entfernung des Affix der selben morphologischen Äquivalenzklsse zugewiesen werden könnten, jedoch von gegensätzlicher Bedeutung sind.

Syntaktische Analyse

Ein weiterer Schritt beim Vorgang des Textminings ist die syntaktische Analyse. Diese hat die Identifikation von syntaktischen Elementen im Strom der Tokens zum Ziel, was in den Aufgabenbereich eines Parsers fällt. Der Parser nutzt sein Wissen über die Grammatik, welche der Sprache des Dokuments zugrunde liegt, um den Einheiten im Tokenstrom ihre syntaktische Funktion zuzuordnen. Für gesprochene, natürliche Sprachen im Allgemeinen und deutsch im Speziellen ist dies allerdings aufgrund komplexer Grammatiken häufig nicht eindeutig oder nur schwer zu bewerkstelligen, weshalb für die syntaktische Analyse stattdessen je nach Einsatzszenario ein Part-of-Speech-Tagger verwendet wird, welcher die Wörter anhand eines statistischen Modells ihre Wortarten und grammatischen Rollen zuweist.

Semantische Analyse

Den letzte der vier hier hervorgehobenen Schritte der analytischen Textaufbereitung ist die semantische Analyse. Verbreitete Erscheinungsformen derselben sind das Named-Entity-Tagging und Sentiment Detection. Ersteres befasst sich mit dem Identifizieren von Entitäten der realen Welt wie etwa Unternehmen, Produkte, Parteien, Personen oder auch Jahreszahlen im Strom der Tokens. Aus „Karl-Heinz steht seit 1995 voll auf Rock'n'Roll." wird so beispielsweise „Karl-Heinz$_{\text{[Person]}}$ steht seit 1995$_{\text{[Jahreszahl]}}$ voll auf Rock'n'Roll$_{\text{[Musikrichtung]}}$." Für die Erkennung der Named-Entities existieren speziell trainierte Klassifizierer und grammatikbasierte Methoden. Die Sentiment Detection bezeichnet das Klassifizieren der emotionalen Lage, die durch den geschrieben Text transportiert wird. Sie ist wie das Named-Entity-Tagging ein potentielles Ziel von Textmining und so Beispiel für das Wiederverwenden eines Textmining-Prozesses im Arbeitsschritt eines anderen.

2.2.2 Feature-Engineering

Das Feature-Engineering hat die Erstellung von Attributschemata für Algorithmen des maschinellen Lernens zum Gegenstand. Im Kontext der beschriebenen Textmining-Methode ist dies die Extraktion von Features aus den computerlinguistisch aufbereiteten Texten. Aus jedem der verarbeiteten Dokumente wird genau eine Instanz gebildet. Mögliche Attribute für die Instanzen sind dann etwa die Nennungsfrequenz bestimmter Named Entities oder die Anzahl der Wörter und Sätze. Aber auch nicht direkt den Texten entnommene Daten und Metadaten können ins Attributschema einfließen. Ist das inhaltliche Erfassen der Dokumente erwünscht, kann die Repräsentierung der Texte mittels Dokumentvektoren erfolgen. Hierzu wird ein Vektorraum spezifiziert und die Dokumente als Vektoren $\mathbf{d} \in \mathbb{R}^n$ mit $n \in \mathbb{N}$ Dimensionen in diesem repräsentiert.

Verteilte und lokalisiert Wortrepräsentationen

Zu diesem Zweck existieren verteilte und lokalisierte Wortrepräsentationen. Letztere zeichnen sich dadurch aus, dass jeder ihrer Informationsträger genau ein Objekt in der Gesamtheit seiner Informationen darstellt. Möchte man zum Beispiel Kraftfahrzeuge mit den Attributen „Farbe" und „Geschwindigkeit" mittels genereller lokalisierter Repräsentationen darstellen, lässt sich dies durch einen binären Vektor erreichen. In diesem Szenario lassen sich zum Beispiel drei Objekte durch einen one-hot-codierten, dreidimensionalen Vektor darstellen, in dem jedes Objekt seine eigene Dimension zugewiesen hat.

- „roter Sportwagen" mit den Eigenschaften „rot" und „schnell" als $(1, 0, 0)$
- „grüner Traktor" mit den Eigenschaften „grün" und „langsam" als $(0, 1, 0)$ und

	Benjamin	ärgert	Alfred	Jochen	hat	damit	nichts	zu	tun	Anstand	Bravo
T1	1	1	1	0	0	0	0	0	0	0	0
T2	1	1	1	0	0	0	0	0	0	0	0
T3	0	0	0	3	2	1	1	1	1	1	1

Tabelle 3: Veranschaulichung des Bag-of-Words Modells

- „blauer LKW" mit den Eigenschaften „blau" und „langsam" als $(0, 0, 1)$

Weist man stattdessen jeder Dimension eine Eigenschaft zu, so erhält man Vektoren mit nominalen Dimensionen, die zur verteilten Repräsentation des selben Sachverhalts geeignet sind. Die drei bekannten Objekte und ihre Codierung ergeben sich als

- „roter Sportwagen" mit den Eigenschaften „rot" und „schnell" als $(rot, schnell)$
- „grüner Traktor" mit den Eigenschaften „grün" und „langsam" als $(grün, langsam)$ und
- „blauer LKW" mit den Eigenschaften „blau" und „langsam" als $(blau, langsam)$

Der Vorteil der zweiten Methode liegt auf der Hand. Möchte man weitere Kraftfahrzeuge in diesem System repräsentieren, lässt sich das durch triviales Erstellen einer neuen Kombination von Eigenschaften erreichen. So könnte ein „blauer Sportwagen" mit den Eigenschaften „blau" und „schnell" einfach als $(blau, schnell)$ dargestellt werden. Möchte das selbe Objekt mit der lokalisierten Repräsentationsmethode hinzufügen, muss hingegen das Attributschema um eine Dimension erweitert werden. Dies offenbart einen Vorteil der verteilten Methode gegenüber der lokalisierten. Sie kann zu Attributräumen mit geringerer Dimension führen, was insbesondere bei Problemen mit verhältnismäßig wenigen Trainingsdaten helfen kann, den Effekten des Fluchs der Dimensionalität entgegenzuwirken. Anhand des „Bag-of-Words"-Schemas und von Word Embeddings sollen die Repräsentationsmethoden im Folgenden auf den Fall der Wortrepräsentation im Kontext der Dokumentklassifizierung übertragen werden.

Feature-Modell: Bag of Words

Das Bag-of-Words-Modell (BoW) ist eine Methode zur Textrepräsentierung, Sie zeichnet sich dadurch aus, dass jedes repräsentierte Wort seine eigene feste Dimension im durch die Attribute aufgespannten Vektorraum erhält, und ist somit als lokalisiert zu betrachten. Um einen Text nach dem BoW-Schema zu repräsentieren, wird zunächst die Menge der $k \in \mathbb{N}$ bekannten Wörter festgelegt und mit ihr die zu den k Wörtern gehörenden Dimensionen im Dokumentvektor. Der Wert eines Dokumentvektors in der Dimension eines Wortes ist dann seine Erscheinungsfrequenz im entsprechenden Dokument, welche anhand des Tokenstroms abgezählt wird. Nach diesem Schema können die Texte

- „Benjamin ärgert Alfred." (T1),
- „Alfred ärgert Benjamin." (T2) und
- „Jochen hat damit nichts zu tun. Jochen hat Anstand. Bravo, Jochen!" (T3)

zum in Tabelle 2.2.2 veranschaulichten Bag-of-Words-Modell verarbeitet werden.

Dabei wird ersichtlich, dass durch BoW Wortreihenfolgen und Grammatik vollständig missachtet werden und lediglich die Multiplizität der Worterscheinungen erfasst wird. Folglich ist die Abbildung der Semantik in das Modell verlustbehaftet. Der Grundgedanke hinter BoW ist, dass sich die inhaltliche Relevanz der Wörter in ihrer Frequenz im Dokument widerspiegelt. Naturgemäß neigen Stoppwörter allerdings dazu, mit hohen Zahlenwerten in die Dokumentrepräsentation einzufließen, weshalb hohe Werte in den entsprechenden Dimensionen des Vektors auch nicht alleine als Indikator für hohe semantische Relevanz der zugehörigen Wörter geeignet sind. Zudem neigt der Ansatz zur Erstellung dünnbesetzter und hochdimensionaler Feature-Matrizen in den resultierenden Datensätzen, was nicht für jeden Lernalgorithmus kein Problem darstellt. Um der Stoppwortproblematik beizukommen, kann beim Erstellen des Dokumentvektors statt der einfachen Termfrequenz der Wörter im Dokument deren TFxIDF-Wert verwendet werden.

Feature Weighting

TFxIDF ist ein Relevanzmaß für Wörter eines Dokuments in einer Dokumentsammlung und bietet sich daher zur Gewichtung der solchen in einer Textrepräsentierung[anderes Wort] an.
Es wird, im Groben durch die Multiplikation der Termfrequenz (TF), der absoluten Anzahl der Erscheinungen des Wortes im Dokument, mit der inversen Dokumentfrequenz (IDF) des Terms, also der inversen Anzahl der Dokumente, die dieses Wort beinhalten, gebildet. Der Grundgedanke ist, dass die für den Text relevanten Terme, die dessen Inhalt tragen,

	Alex	ist	Bernds	Vater	als	er	Teil	seiner	Familie	Elli	Mutter	sie	Sarah	Schwester
T4	1	2	1	2	1	1	1	1	1	0	0	0	0	0
T5	0	2	1	0	1	0	1	1	1	1	2	1	0	0
T6	0	2	1	0	1	0	1	1	1	0	0	1	1	2

Tabelle 4: Bag-of-Words-Modell zur Veranschaulichung von TFxIDF

	Alex	ist	Bernds	Vater	als	er	Teil	seiner	Familie	Elli	Mutter	sie	Sarah	Schwester
T4	1	0.66	0.33	2	0.33	1	0.33	0.33	0.33	0	0	0	0	0
T5	0	0.66	0.33	0	0.33	0	0.33	0.33	0.33	1	2	0.5	0	0
T6	0	0.66	0.33	0	0.33	0	0.33	0.33	0.33	0	0	0.5	1	2

Tabelle 5: Bag-of-Words-Modell zur Veranschaulichung von TFxIDF

häufig vorkommen und daher eine hohe TF bezüglich des Dokuments haben. Da allerdings auch Stopp- und Füllwörter eine hohes Aufkommen im Dokument haben aber nur wenig relevant sind, existiert Bedarf an einer Methode, deren Gewicht klein zu halten. Dazu wird die inverse Dokumentfrequenz verwendet. Die Motivation dahinter ist, dass Stopp- und Füllwörter in vielen Dokumenten der Sammlung vorkommen, während inhaltstragende Wörter des Dokuments in diesem tendenziell eine alleingestellte Rolle in der Sammlung spielen. Teilt man die TF eines Wortes im Dokument durch die Anzahl der Dokumente, in der diese vorkommt, so moderiert letztere erstere und ermöglicht so die Indikation der WortrelevanzRamos [2003].

Definition 2.7 (TFxIDF) *Sei D die Menge der Dokumente, also der Korpus, und W die Menge aller verschiedenen Wörter im Korpus, welche auch als „Wörterbuch" bezeichnet wird. Weiter berechne die Funktion $tf : W \times D \to \mathbb{Z}$ die Anzahl der Erscheinungen eines Wortes $w \in W$ im Dokument $d \in D$ und die Funktion $df : W \to \mathbb{Z}$ die Anzahl der Dokumente im Korpus, die das Wort $w \in W$ beinhalten. Dann wird die Relevanzgewichtung $tfidf : W \times D \to \mathbb{R}$ eines Wortes $w \in W$ in einem Dokument $d \in D$ berechnet als*

$$tfidf(w,d) := \frac{tf(w,d)}{df(w)}$$

Es existieren weitere Varianten, die eine Gewichtung für den Anteil von tf, df oder beider einführen, um etwa den Einfluss von Dokumentlängen auf das Maß zu mindern.

Zum Verdeutlichen des Konzepts soll nun ein Beispiel Betrachtet werden. Die Texte dazu sind

- „Alex ist Bernds Vater. Als Vater ist er Teil seiner Familie." (T4),
- „Elli ist Bernds Mutter. Als Mutter ist sie Teil seiner Familie." (T5) und
- „Sarah ist Bernds Schwester. Als Schwester sie ist Teil seiner Familie." (T6).

Das Bag-of-Words-Model ohne TFxIDF-Gewichtung dazu ist in Tabelle 2.2.2 zu sehen. Wendet man nun die TFxIDF-Gewichtung nach Definition 2.7 an, ergibt sich das aus Präsentationsgründen auf zwei Stellen gerundete in Tabelle 2.2.2 zu betrachtende Modell. Wie zu sehen ist, sind die Wörter, die dank der TFxIDF-Werte jetzt noch besser herausgestellt sind, allesamt solche, die für die Semantik der Texte relevant sind. Bei wachsender Größe der Dokumentsammlung ist zudem zu erwarten, dass sich die TFxIDF-Werter weitere inhaltstragende Wörter, wie „Bernds" oder „Familie", von denen der Stoppwörter abheben. Ebenfalls ist die Aussagekraft von „sie" gegenüber „Elli" gefallen, wodurch diese nicht mehr die gleiche Wertigkeit im Datensatz haben. Durch diese Unterscheidung wird es dem Lerner, der auf den Datensatz angewendet werden soll, vereinfacht, mit der Hilfe dieser Unterschiede spezifischer zu diskriminieren, weshalb eine bessere Vorhersageperformanz ermöglicht werden kann.

Feature-Modell: Word Embeddings

Eine weitere Art, Dokumente mit Hilfe von Vektoren darzustellen, ist, dies mittels Word Embeddings zu tun. Dazu wird jedes bekannte Wort $w \in W$ im Wörterbuch durch einen festen Vektor $v_w \in \mathbb{R}^n$ mit $n \in \mathbb{N}$ Dimensionen repräsentiert. Der das Dokument darstellende Vektor wird durch die Kombination der Wortvektoren erstellt, zum Beispiel durch Summenbildung.

Betrachtet man die Texte

- „Karl isst gerne Reis." (T7),
- „Karl isst gerne Birnen." (T8) und

- „Karl isst gerne Kiwis.” (T9)

mit dem zugehörigen Wörterbuch $D := \{\text{karl, isst, gerne, reis, birnen, kiwis}\}$ und die hier aus Präsentierungsgründen gewählten zugehörigen Einbettungen in den $\mathbb{R}^3$ $v_{\texttt{karl}} = (1,5,0)^T$, $v_{\texttt{isst}} = (-3,1,0)^T$, $v_{\texttt{gerne}} = (1,1,0)^T$, $v_{\texttt{reis}} = (0,0,2)^T$, $v_{\texttt{birnen}} = (0,0,5)^T$ und $v_{\texttt{kiwis}} = (0,0,7)^T$, dann sind die durch die Berechnung der Summe gebildete Dokumentvektoren dazu für T7 $(-1,7,2)^T$, für T8 $(-1,7,5)^T$ und für T9 $(-1,7,7)^T$. Hierbei wurden, wie in der Praxis nicht unüblich, Satzzeichen ignoriert und Majuskel wie Minuskel behandelt, also die Groß- bzw. Kleinschreibung missachtet. Wie durch das Beispiel ersichtlich wird, können mit entsprechenden Wortrepräsentanten im Vektorraum bei der Bildung der Dokumentvektoren semantische Ähnlichkeiten als numerische Ähnlichkeiten erhalten bleiben. Zudem wurde die Anzahl der zur Darstellung der Dokumente benötigten Dimensionen auf $n = 3$ gegenüber einem Bag-of-Words-Modell, welches hier $|D| = 6$ Dimensionen benötigt hätte, reduziert. Da bei der Verwendung von Word Embeddings nicht zwangsweise jedes Wort im Dokument seine eigene (oder gar keine) Dimension erhält, sondern die Repräsentierung der Wörter im Allgemeinen über mehrere Dimensionen verteilt ist, gehören sie zu den verteilten Repräsentierungen. Werden die eingebetteten Wörter mittels einer Summierung zum Dokumentvektor kombiniert, bleibt allerdings, wie beim Bag-of-Words-Modell, der Kontext des Wortes im spezifischen Dokument nicht erhalten.

Eine Methode zum Erstellen von Word Embeddings ist Word2vec[Mikolov u. a., 2013]. Bei dieser Abbildung der Wörter in den $\mathbb{R}^n$ bleiben viele semantische und syntaktische Relationen erhalten. Diese erhalten sich sogar über algebraische Funktionsanwendungen hinweg. Diese lassen sich sogar über algebraische Operationen darstellen. So gilt für die Repräsentanten der verwendeten Wörter im Vektorraum $Vater - Mann + Frau \sim Mutter$ und $ging - gehen + laufen \sim lief$. Bei Word2vec Word Embeddings handelt es sich um probabalistische Modelle der Kontexte, in denen die Wörter gewöhnlich stehen. Sie werden durch ein Neuronales Netz, das diese vorherzusagen versucht, trainiert.

Auf Word Embeddings lässt sich ebenfalls ein TFxIDF-Gewichtungsschema verwenden. Dazu wird bei der Berechnung der Dokumentvektoren jeder Wortvektor mit seinen TFxIDF-Gewicht verrechnet.

Wie Lillebereg und Zhang herausfanden, kann es für einen Lerner von Vorteil sein, Dokumentrepräsentationen von TFxIDF-gewichteten Word Embeddings mit einem TFxIDF-Gewichtete Bag-of-Words-Schema zu augmentieren[Lilleberg u. a., 2015].

2.2.3 Musterextraktion

Die Musterextraktion stellt den wesentlichen Schritt der hier vorgestellten Methode des Text Mining dar. Anhand der aus Text und Metadaten der Dokumente extrahierten sowie möglichen weiteren, textfremden Features werden die im Datensatz als inhärent vermutete Informationen und Querverbindungen sichtbar gemacht. Dies erfolgt durch ein Verfahren des maschinellen Lernens. Dabei sind Lernziele des überwachten aber auch des unüberwachten Lernens denkbar. So könnte das unüberwachte Lernen dazu eingesetz werden, Kategorisierungen in der Dokumentmenge zu finden, was bibliothekarische Tätigkeiten unterstützen würde. Im Fall des unüberwachten Lernens stellt die Sentiment Detection, die Stimmungserkennung einen klassischen Anwendungsfall des Text Minings dar. So könnte zum Beispiel ein Unternehmen die öffentliche Wahrnehmung durch die Klassifizierung der es betreffenden Twitter-Meldungen in „positiv” und „negativ” automatisiert analysieren und darauf aufbauend Marketing-Entscheidungen treffen. Bei einer solchen Klassifizierungsaufgabe ist insbesondere bei der Anwendung des Bag-of-Words-Modells zur Textrepräsentierung mit dem Aufkommen hochdimensionaler, dünnbesetzter Feature-Räume und großen Mengen an Trainingsinstanzen zu rechnen. Zur Lösung bieten sich daher speziell SVMs an, da sie entwurfsbedingt die durch die angeführten Eigenschaften implizierten Anforderungen erfüllen.

2.3 Grundlagen der Teilnahme an den Börsen

Nach der Einführung in die für diese Bachelorarbeit relevanten Aspekte des maschinellen Lernens und des Text Minings folgt nun eine kurze Einführung in die Begebenheiten des börslichen Handels und der Teilnahme an Aktienmärkten. Zuerst werden die wichtigen Grundbegriffe eingeführt und ein grober Überblick über den Börsenhandel geschaffen, dann folgt eine kurze Einführung in die Teilnahme am Aktienmarkt.

2.3.1 Grundbegriffe

Eine Börse ist ein nach Regeln und Gesetzen organisierter Markt, an dem Güter bestimmter Arten und Beschaffenheiten in untereinander austauschbaren, äquivalenten Einheiten gehandelt werden. Vertreter dieser Instrumente der Wertanlage, sind Rohstoffe wie Gold, Erdöl oder Kaffee, Devisen, also Fremdwährungen, Anleihen, welche eine Form des Schuldbriefs darstellen und Aktien, welche die Anteilseignerschaft des Besitzers am zugehörigen Unternehmen verbriefen. Zudem existieren zwei weitere nennenswerte Klassen von an Börsen gehandelten Gütern. Die Fonds stellen Bündel der Güter dar und abstrahieren so von den einzelnen Objekten, sodass diese anteilsweise und gestreut erworben werden können. Die derivativen Finanzprodukte zeichnen sich dadurch aus, dass ihnen ein anderer Wert, der sogenannte Basiswert, zugrunde liegt. Neben deren klassischen Unterarten der Optionen und Futures, die im ersten Fall das Recht und im zweiten die Pflicht des Besitzers verbriefen, den Basiswert zu einem bestimmten Zeitpunkt und Umfang mit einem bestimmten anderen Marktteilnehmer zu handeln, existieren viele weitere, welche je nach Ausgestaltung und Gesetzeslage sogar die Wette der Marktteilnehmer auf den Ausgang von Ereignissen wie der Wahl des US-Präsidenten abbilden können. Die Eigentumsübertragung der Güter findet häufig nicht physisch statt, sondern mittels Eigentumsnachweis oder virtuell durch die Übertragung eines äquivalenten Geldwertes.

Die Preisfeststellung für die an einer Börse gehandelten Güter, also ihrer Kurse, erfolgt je nach Art des Anlageinstruments unterschiedlich, unterliegt jedoch stets den Gesetzmäßigkeiten von Angebot und Nachfrage. Insbesondere ist die Markteffizienzhypothese auf Börsen anzuwenden. Sie besagt, dass der Kurs eines an der Börse gehandelten Guts stets die Gesamtheit der es betreffenden, öffentlich verfügbaren Informationen widerspiegelt. Ihr entgegen steht die Tatsache, dass neu veröffentliche Informationen naturgemäß mit einer Verzögerung durch ein Medium propagiert werden. So entsteht ein theoretisches Zeitfenster, ein „Window of Opportunity", in dem der Markt nicht als effizient zu betrachten ist, und das Teilnehmer, welche über die neue, noch nicht vollständig propagierte Information verfügen, zu ihren Gunsten ausnutzen können, so sie dazu fähig sind[Malkiel und Fama, 1970].

Die für diese Arbeit relevante Anlageklasse sind die Aktien. Dies sind Wertpapiere, welche ihrem Eigentümer bescheinigen, dass er einen bestimmten Anteil am Unternehmen hält, das sie ausgibt. Letzteres stellt den Emittenten dar. Der Emittent verfolgt mit der Ausgabe von Aktien die Absicht der Finanzierung, etwa von anstehenden Investitionen. Der Börsenkurs einer Aktie ist dabei nicht an das durch sie Repräsentierte Grundkapital des Unternehmens gebunden, sondern wird durch den Markt gebildet. Dies geschieht durch einen Aktienmakler oder Handelssystem und in der Regel nach umsatzmaximierenden, die „Liquidität" steigernden, Gesichtspunkten. Treffen mehrere Händler aufeinander und haben verschiedene Preisvorstellungen beim Kauf oder Verkauf der Aktien, so werden Stückzahl und Preis der einzelnen Geschäfte so bestimmt, dass die maximale Stückzahl an Aktien den Besitzer wechselt, ohne dass etwaige, von den Händlern gestellte Bedingungen, zum Beispiel an Preislimits oder die Gesamtausführung des Handels, verletzt werden. Durch diese Art der Preisfeststellung kann es insbesondere bei stark frequentierten Handelsplätzen vorkommen, dass sich der Kurs einer Aktie mehrere Male in einer Sekunde ändert. Intensives Kursänderungsverhalten wird als „hohe Volatilität" charakterisiert. Für die Angleichung der Aktienkurse über die Grenzen der Handelsplätze hinweg sorgen die Arbitrageure. Diese sind Händler, die Aktien oder andere Anlageinstrumente an dem einen Ort erwerben und an einem anderen wieder verkaufen, um Profite durch die meist geringen, regionalen Preisunterschiede zu erzielen. Damit sorgen sie dafür, dass die regionalen Preisunterschiede limitiert bleiben.

Emittenten von Aktien sind gesetzlich verpflichtet, Informationen, die für den Kurs des Wertpapiers relevant sind, zeitnah nach Bekanntwerden einem möglichst breiten Publikum zu veröffentlichen. Dies dient der Steigerung der Markteffizienz und -gerechtigkeit durch die Vermeidung von Bevorzugungen durch sogenannten Insider-Handel und geschieht in Form von Ad-hoc-Mitteilungen, welche in der Regel durch Kommunikationsdienstleister veröffentlicht werden. Dabei wird eine solche Mitteilung zuerst durch den Dienstleister an die Betreiber der Börsen, an denen die Aktie gehandelt wird, und die Bundesanstalt für Finanzdienstleistungsaufsicht übermittelt, Diese entscheiden bei Bedarf, den Handel mit diesem Wertpapier auszusetzen, falls eine zu extreme, den Markt destabilisierende Auswirkung der Information zu erwarten ist. Anschließend, nach bis zu einer halben Stunde, wird sie an internationale Nachrichtenagenturen weitergeleitet, welche die Meldung an die Öffentlichkeit tragen.

2.3.2 Teilnahme am börslichen Aktienhandel

Trafen die Händler an den Börsen früher noch persönlich aufeinander, um sich beim sogenannten Parketthandel mündlich über ihre Angebote zu verständigen, so erfolgt heute der Großteil des Börsenhandels computergestützt auf virtuellen

Marktplätzen. Mit einem Marktanteil von 90% im Aktienhandel stellt die Handelsplattform Xetra, ein Produkt der Frankfurter Wertpapierbörse, den größten deutschen Handelsplatz für Wertpapiere dar. Privatanleger handeln in aller Regel nicht direkt an der Börse sondern über Vermittler, die sogenannten Broker. Dies sind zumeist Banken, welche die Aufträge zum Handeln, die Wertpapierorders, gegen eine Gebühr durchführen und bei denen die Privatanleger Depots besitzen, in denen ihr Portfolio, die Sammlung ihrer aktuell gehaltenen börslichen Wertanlagen, beinhaltet ist.

Marktteilnehmer gehen beim Handel mit Anlageinstrumenten eine von zwei verschiedene Investitionspositionen ein, entweder die „Long"oder die „Short"-Position. Die „Long"-Position ist charakterisiert durch die Erwartung eines anstehenden Kursanstiegs des Instruments.Der Marktteilnehmer, der die „Short"-Position eingeht, erwartet den Fall des Kurses.In der Theorie ist es möglich, dass eine Aktie verkauft wird, die nicht Eigentum des Verkäufers ist. Bei dieser „Leerverkauf" genannten Art des Verkaufsleiht sich der Verkäufer die Aktie von einem Dritten, verkauft sie an de Käufer und muss selbst zu einem späteren Zeitpunkt einen Deckungskauf tätigen, um der Drittperson die Aktie in gleicher Stückzahl zurückzugeben. Da Leerverkäufe von Aktien in Krisenzeiten einen den Preisverfall verstärkenden Effekt haben, bergen sie eine systemische Gefahr für den Aktienmarkt, weshalb sie in den Ländern der Europäischen Union durch die Verordnung 236/2012 stark reglementiert sind, sodass sie etwa in Krisenzeiten durch die aufsehende Behörde verboten werden können. Das Eingehen dieser Art von Short-Positionen birgt zudem ein weiteres Risiko gegenüber der Long-Position. Während der Maximalgewinn beim Leerverkauf einer Aktie deren Wert zum Zeitpunkt des Handels entspricht, ist der maximal mögliche Verlust hingegen theoretisch unbeschränkt, da der Preis unbegrenzt steigen könnte. Generell ist der Umgang mit dem Investitionsrisiko beim börslichen Handel von zentraler Bedeutung. Da dieses Risiko und die mögliche Höhe des Gewinns dazu tendieren, miteinander zu wachsen oder zu fallen, ist ein Anleger im Allgemeinen gut damit beraten abzuwägen, ob er sein Kapital mit geringen Wachstumsaussichten und geringer Verlustwahrscheinlichkeit oder eher mit größeren Wachstumsaussichten aber größerer Verlustwahrscheinlichkeit investieren möchte. Die gleichzeitige Optimierung dieser beiden Größen stellt somit den wesentlichen Faktor für die Erfolgsaussichten einer Börsenhandelsstrategie dar.

Eine weitere Dimension, in der sich eine Wertanlage kategorisieren lässt, ist der Zeithorizont, also wie lange eine Position gehalten wird, bevor Gewinne oder Verluste realisiert werden. Während im hauptsächlich institutionellen Hochfrequenzhandel eine Position für den Bruchteil einer Sekunde bis zu wenigen Sekunden gehalten wird, schließen die Akteure im Day Trading ihre Positionen bis zum Ende des laufenden Tages wieder. Eine breitere Öffentlichkeit insbesondere privater Investoren agiert mit Horizonten, deren Bemessung in Tagen, Wochen, Monaten oder Jahren hinreichend Präzise ist.

Je nach Broker, Handelsplatz, gehandelter Instrumentenklasse und ob sie für den Eintritt oder den Austritt aus dem Markt getätigt wird, gibt es verschieden Ordertypen, mit denen der Kunde eines Brokers seine Wünsche, mit dem Markt zu interagieren, spezifizieren kann. Die geläufigsten sind die „Market Order", die „Limit Order", und die „Stoploss Order". Die Market Order erteilt den Auftrag zum Kauf oder Verkauf zur nächstmöglichen Gelegenheit entsprechend dem momentan für den Kunden günstigsten Kurs, wohingegen die Limit Order beim Kauf einen Maximalpreis und beim Verkauf einen Mindespreis enthält. Wird in einer festen Zeitspanne kein Kurs, der den Bedingungen der Limit Order genügt, festgestellt, wird sie nicht ausgeführt. Dieses Risiko ist bei der Market Order nicht gegeben, sie wird zeitnah zum Marktpreis durchgeführt, allerdings ohne Preisgarantien. Der Stoploss Order wird beim Eröffnen einer Position nach Manier der Market Order zusätzlich ein Preis, der sogenannte „Stoploss", beigefügt, dessen passieren des Kurses in die für den Kunden ungünstige Richtung das automatische Platzieren einer Market Order zum Schließen der Position zur Folge hat. Dies hilft dem Anleger bei der Beschränkung von Verlusten. Eine flexiblere Variante der Stoploss Order ist die „Trailing Stoploss Order". Bei dieser wird der Stoploss ebenso um eine feste Differenz von Kurs beim Eingehen der Position gewählt, aber der Stoploss-Preis bei für den Kunden günstigen Kursbewegungen stets so angepasst, dass seine Distanz zum aktuellen Kurs, die prozentual oder absolut bemessen ist, nie die initiale Distanz zum Kurs des Markteinstiegs überschreitet.

Die Performanz einer Anlagestrategie kann mit Hilfe des Sharpe Ratios quantifiziert werden. Er berechnet sich als der Erwartungswert der Renditen geteilt durch die Standardabweichung derselben, also als

Definition 2.8 (Sharpe Ratio) *Sei „Rendite" eine Zufallsvariable, welche die Renditen beim Handel mit Finanzinstrumenten beschreibt. Dann ist der Sharpe Ratio definiert als*

$$SharpeRatio := \frac{E[Rendite]}{\sqrt{Var[Rendite]}}$$

Werte unter 0 sind dabei als ungünstig anzusehen und solche > 1 als besonders günstig, da sie Strategien mit verhältnismäßig sicherem Gewinn charakterisieren.

3 Wissenschaftlicher Kontext

Dieser Abschnitt der Bachelorarbeit beschäftigt sich mit ihrer Einordnung in den zeitgenössischen wissenschaftlichen Kontext. Spezielles Augenmerk liegt dabei zunächst auf akademischen Arbeiten, die eine besondere thematische Nähe zu dieser Arbeit aufweisen oder durch grundlegende Erkenntnisse zu ihrer Konzeption beigetragen haben. Abschließend erfolgt eine kurze Ausführung darüber, wie sich diese Bachelorarbeit von den vorgestellten wissenschaftlichen Veröffentlichungen abgrenzt.

In einer erstmals 2004 veröffentlichten akademischen Untersuchung haben Muntermann und Guettler die Auswirkungen der Veröffentlichungen von Ad-hoc-Mitteilungen auf Preis und Handelsvolumen zugehöriger deutscher Aktien statistisch nachgewiesen[Muntermann und Guettler, 2007]. Ihren Ergebnissen zufolge benötigte der Markt damals bis zu 30 Minuten, um in Reaktion auf eine Nachricht den Kurs einer Aktie anzupassen. Zudem beobachteten sie, dass die Auswirkungen einer Veröffentlichung auf den Preis viel gehandelterAktien tendenziell geringer ausfielen und sowohl der Preis als auch das gehandelte Volumen substantiell stärker ausgeprägt abnormales Verhalten zeigten, je größer das Handelsvolumen am vorherigen Tag war.

Im selben Jahr benutzte Mittermayer Techniken des Textmining zum Entwurf eines nachrichtenverarbeitenden Handelssystems, das für den amerikanischen Aktienmarkt mittels der örtlichen Ad-hoc-Publizität Handelsempfehlungen gab[Mittermayer, 2004]. Dazu wurden die Mitteilungen durch ein TFxIDF-gewichtetes Bag-of-Words-Modell repräsentiert und mittels eines Zeithorizonts von bis einer Stunde nach ihrer Veröffentlichung sowie manuell festgelegten Schwellenwerten für die Kursbewegung innerhalb des Horizonts als „gute", „schlechte" und „indifferente" Nachrichten klassifiziert. Es konnte unter der Annahme fixer Transaktionskosten von 10$ und durch Zuhilfenahme einer SVM ein nach dem Zufallsprinzip handelndes System als Benchmark an Performanz übertroffen werden und in der Simulation profitabel gehandelt werden.

Aufbauend auf der Arbeit von Muntermann und Guettler[Muntermann und Guettler, 2007] haben Groth und Munterman auf der Basis von Textmining eine Investmentstrategie entworfen[Groth und Muntermann, 2009]. Dazu verwendeten sie durch den Finanzdienstleister DGAP veröffentlichte Ad-hoc-Mitteilungen, aus denen sie Stoppwörter entfernten und was dabei verblieb mit einem Stemmer transformierten, um aus den Dokumenten TFxIDF-gewichtete Bag-of-Words-Features für ein durch eine SVM zu lösendes Klassifizierungsproblem zu extrahieren. Vorhersageziel war dabei, ob sich eine Nachricht positiv oder negativ auf den Kurs der zugehörigen Aktie am Handelsplatz Xetra auswirkt, was als signifikante Kursbewegung in die entsprechende Richtung innerhalb der Stunde nach Publikation definiert wurde. Auch hier konnte in einer Simulation zufälliger Aktienhandel bezüglich der Rendite geschlagen werden.

2013 wurd durch Hagenau u.a. verschiedene Methoden, durch die DGAP publizierte Ad-hoc-Mitteilungen als Attributvektoren darzustellen, untersucht, um damit ein profitables Handelssystem zu entwickeln[Hagenau u. a., 2013]. Nachrichten wurden durch eine lineare SVM als „positiv" und „negativ" für den Kurs der entsprechenden Aktie klassifiziert und dabei analog das Kaufen und Leerverkaufen der Aktie mit Verlassen der Investment-Position am Ende des Handelstages ausgelöst. Durch die günstigste Feature-Konfiguration konnte bei einer Handelssimulation die Accuracy von 76.3% erreicht werden, wobei konstantes Raten der größten Klasse einen Accuracy-Wert von 58.3% geliefert hätte. Durch Verwenden der günstigsten Textrepräsentation, die Frequenzen der am häufigsten vorkommenden Kombination zweier Wörter in einem Fenster der Größe fünf, welche durch bi-normale Separation selektiert wurden, konnte eine Durchschnittliche Rendite von 1.8% pro Handel erreicht werden. Lilleberg und Zhang haben 2015 herausgefunden, dass die Kombination von TFxIDF-gewichteten, durch Word2vec erstellten Word Embeddings mit einem TFxIDF-gewichteten Bag-of-of-Words-Ansatz in Verbindung mit einer SVM die Klassifikationsperformanz gegenüber den einzelnen, für sich genommenen Methoden der Featurerepräsentierung verbessern kann[Lilleberg u. a., 2015]. Sie mutmaßten, dass Word2vec-Embeddings und Bag-of-Words-Features sich in der durch sie getragenen Semantik in geeigneten Anwendungsszenarien ergänzen könnten. Daher findet diese Methode der Textrepräsentierung auch in dieser Bachelorarbeit Anwendung.

In dieser Arbeit wird der Grundgedanke des Textminings von Mittermayer[Mittermayer, 2004] auf den deutschen Aktienmarkt übertragen. Während er die Wahl der Klassen auf Basis der Kursbewegung mit manuell gewählten Schwellwerten durchführte, ist hier das unmittelbare Klassifikationsziel die Anwendung einer primitiven Handelsstrategie. Die Vorhersage der konkreten zu verwendenden Strategie stellt in diesem Kontext ein Novum dar, denn auch Neumann und Hagenau[Hagenau u. a., 2013] haben durch die Wahl der Long- oder Short-Position als Vorhersageziel streng interpretiert nur die Auswirkungen einer Nachricht auf den Aktienkurs vorhergesagt. Eine weitere Novität ist die Textrepräsentierung mittels TFxIDF-gewichteter Word Embeddings und die Kombination derselben mit dem etablierten Bag-of-Words-Ansatz, wodurch eine Verbesserung der prädiktiven Performanz der im Rahmen des hier durchgeführten Textminings konstruierten Klassifzierer angestrebt wird.

4 Verwendete Daten

Die Datengrundlage für dies Bachelorarbeit bilden Nachrichten zu Aktien und die Kurse der Wertpapiere. Strukturell betrachtet handelt es sich also um in natürlicher Sprache verfasste Textdokumente und Zeitreihen. Anhand der Nachrichten sollen hier automatisch inhaltliche Merkmale identifiziert werden, die als Signal zum Anwenden primitiver Strategien beim Aktienhandel zu verwenden sind. Mit Hilfe der Kursdaten werden die im Schema der Merkmale repräsentierten Nachrichten mit Klassenwerten versehen und so die Instanzen erstellt, mit denen die in den Textmining-Prozess eingebetteten Klassifizierer trainiert und evaluiert werden. Im Folgenden werden die einzelnen Datensätze für Nachrichten und Kursverläufe vorgestellt und dann ein typischer Datensatz charakterisiert, der durch die Zusammenführung der beiden Datensammlungen bei der Erstellung von Instanzen entsteht.

4.1 Nachrichtendaten

Die die Aktien betreffenden Nachrichten, welche hier verabreitet wurden, sind Ad-hoc-Mitteilungen. Diese sind per definitionem relevant bei der Bewertung der zugehörigen Aktien durch Anleger und unterliegen gesetzlichen Regelungen unter anderem bezüglich dem Zeitpunkt ihrer Veröffentlichung und Zugänglichkeit. Die hier verwendeten Mitteilungen sind solche, die durch den Finanzdienstleiter DGAP (Deutsche Gesellschaft für Ad-hoc-Publizität) verbreitet und Archiviert wurden. Mit einem Marktanteil von etwa 80% aller in Deutschland veröffentlichten Ad-hoc-Meldungen im Jahr 2012 ist DGAP der größte Dienstleister dieser Art, was zu seiner Auswahl beitrug. Das online zugängliche Archiv der DGAP ist mit 77818 Ad-hoc-Mitteilungen über den Zeitraum vom ersten Juli 1996 bis zum zwölften April 2016 im hier verwendeten Datensatz repräsentiert. Die absolute Verteilung der Nachrichten über die Veröffentlichungsjahre ist Abbildung 4.1 zu entnehmen. Jeder der Nachrichten ist mit dem Zeitstempel ihrer Veröffentlichung versehen, welcher auf die Minute genau auflöst. Abbildung 4.1 zeigt die Verteilung der Nachrichten über die Stunde ihrer Veröffentlichung am Tag. Sie zeigt, dass viele Nachrichten vor neun und nach neuzehn Uhr und damit ausßerhalb der üblichen Handelszeiten veröffentlicht wurden. So zeichnet sich bereits ab, dass viele der 77818 Mitteilungen nicht zur Erstellung von Instanzen geeignet sein werden, da hier insbesondere die unmittelbaren Effekte der Veröffentlichungen von Interesse sind. Für 1801 verschiedene Aktien liegen Mitteilungen vor, darunter die aller 80 im April 2016 in den Indices DAX und MDAX vertretenen, welche sich durch ein besonders hohes Volumen am Markt auszeichnen. Die Menge der Nachrichtendaten ist aus maschineller Verarbeitungssicht uneinheitlich beschaffen. Sie sind nicht allesamt in deutscher Sprache verfasst, sondern zum Teil auf englisch, französisch und italienisch. Die Dokumente selbst sind teilweise mit ASCII-Ornamenten ausgeschmückt, zudem enthalten viele der Mitteilungen einen Textblock, der eine Kontaktadresse im Üblichen Schlüssel-Wert-Adressformat angibt. Eine Vereinheitlichung zum Zwecke der Rauschunterdrückung wird durch den in Abschnitt 5 beschriebenen Textmining-Prozess unternommen.

4.2 Kursdaten

Die für die Durchführung der Untersuchungen dieser Bachelorarbeit benötigten Kursdaten wurden von einem Finanzdienstleister erworben. Dabei wurden die historischen Kurse von 699 Aktien bezogen, darunter die aller 80 im April 2016 in DAX und MDAX vertretenen. Als Handelsplätze für die Selektion der Aktien wurden Xetra und die Börse Frankfurt gewählt, da sie besonders umsatzstark sind und aus größeren Datenmengen tendenziell bessere Muster extrahiert werden können. Für 422 der Aktien wurden die Xetra-Kurse bezogen, für alle 699 die der Börse Frankfurt. Die vorliegenden Kursdaten entsprechen der Historie aller Preisfeststellungen für die jeweiligen Aktien, sind also Zeitreihen der tatsächlich getätigten Verkäufe. Jeder Handel ist dabei durch ein Tripel aus sekundengenauem Zeitstempel, Preis in Euro und dem gehandelten Volumen repräsentiert. Die früheste registrierte Transaktion der Kurshistorien ist vom zweiten Februar 2002 und der späteste vom achten Juli 2016.

4.3 Datenpaare

Bei einer ersten Sichtung der Daten fällt die offensichtlich statistisch signifikante Auswirkung der Nachrichtenveröffentlichung auf Liquidität und Volatilität der Kurse auf. Wie Grafik 4.3 verdeutlicht, steigt die Anzahl der Preisfeststellungen und damit die der Transaktionen pro Minute einer Publikation sprunghaft an. Dies ist ein starker Indikator für einen gesteigerten Umsatz. Grafik 4.3 verdeutlicht hingegen auch die Auswirkung der Ad-hoc-Mitteilungen auf die Kurse der Wertpapiere. Es ist deutlich zu sehen, dass die Kursänderungen nach der Publikation einer Nachricht stärker werden, was durch den durchschnittlichen Absolutwert der prozentualen Abweichungen zwischen zwei Preisfeststellungen innerhalb der zugeordneten Minute vor der Publikation der Mitteilung dargestellt ist. Dies lässt auf signifikante Kursanpassungen hoffen, welche sich positiv auf die Aussagekraft der zu lernenden Modelle auswirken könnten.

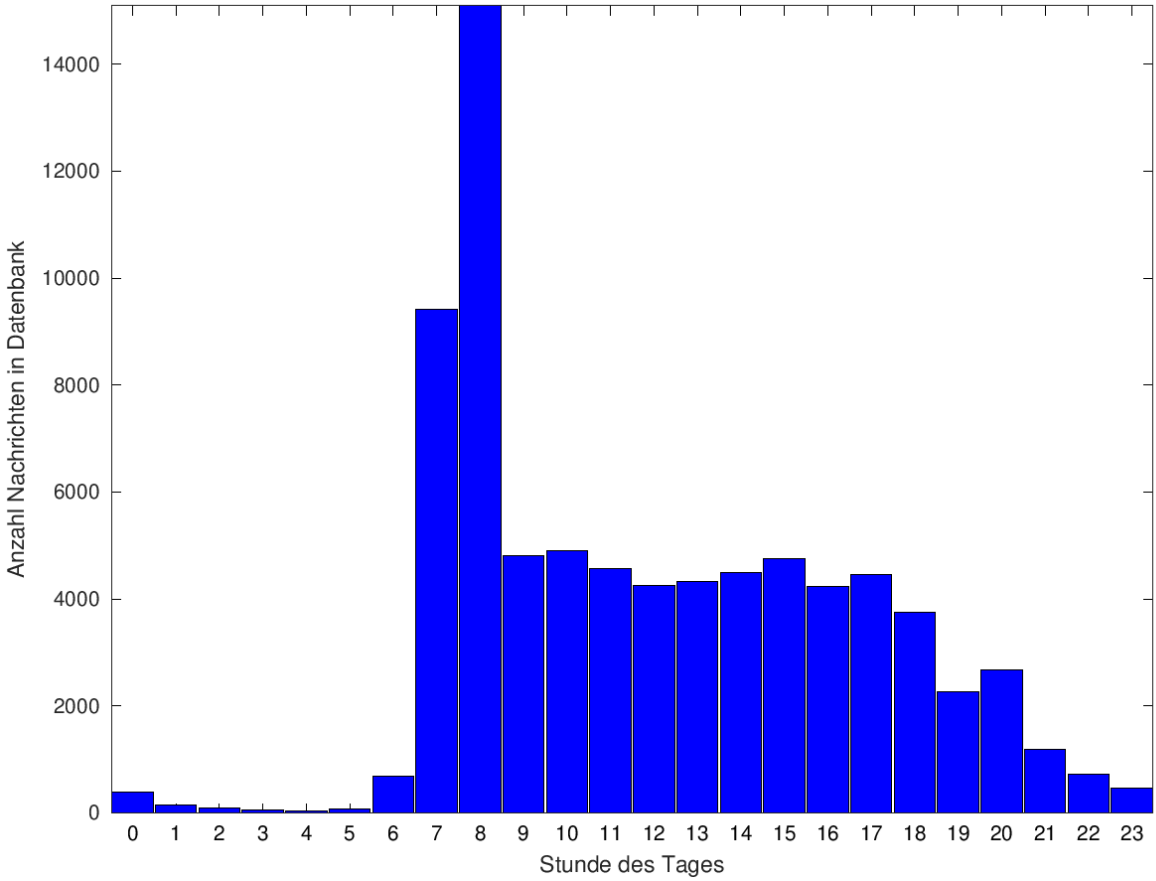

Abbildung 1: Verteilung aller Nachrichten über die Uhrzeit der Veröffentlichung

5 Methodik

Dieser Abschnitt der Bachelorarbeit widmet sich einer Übersicht über die Methodik, nach der ihr Thema behandelt wurde. Zentrale Problemstellung ist die Vorhersage der Performanz primitiver Aktienhandelsstrategien unter Verwendung einer Textmining-Methode. Zu diesem Zweck wurden drei abstrakte Strategietypen definiert, die sich dadurch unterscheiden, nach welchen Signalen eine eingegangene Investmentposition im Aktienmarkt wieder verlassen wird, also wann bei einer Long-Position der Verkauf einer gehaltenen Aktie und bei einer Short-Position der Deckungskauf erfolgt. Das Einnehmen einer solchen Investment-Position wird stets durch die Veröffentlichung einer Ad-hoc-Mitteilung ausgelöst und erfolgt unmittelbar. Mit dem angeführten Strategiebegriff lassen sich vier Fragestellungen formulieren.

1. Kann aus einer Menge primitiver Aktienhandelsstrategien diejenige mit dem größten Profit vorhergesagt werden?

2. Kann insbesondere vorhergesagt werden, ob eine solche Strategie in ihrer Long- oder Short-Variante profitabler ist?

3. Birgt eine von drei zur Anwendung kommenden Methoden, Dokumente als Features zu repräsentieren, dabei einen Vorteil gegenüber den anderen? Die Features werden mittels Bag-of-Words oder Word Embedding Modellen erstellt, die Kombination beider stellt die dritte Methode dar.

4. Könnte aus der Kombination der primitiven Strategien mit der Textmining-Methode ein profitables Vorgehen beim börslichen Handel mit Aktien abgeleitet werden?

Für die Programmierarbeiten, welche zur Beantwortung dieser Fragestellungen notwendig waren, wurde auf das quelloffene Textklassifizierungsframework DKPro-TC zurückgegriffen, das aus dem „Darmstadt Knowledge Processing Repository" des Fachgebiets Ubiquitous Knowledge Processing der TU Darmstadt stammt[Daxenberger u. a., 2014] . Zentrale Komponente von DKPro-TC ist Apache UIMA, ein Inhaltsanalyseframework, welches das Schreiben von Data-Mining-Anwendungen vereinfacht. Die Wahl von DKPro-TC begründete sich zudem darin, dass es mit Schnittstellen zu weiterer für das Machine Learning nützlicher Software wie zum Beispiel WEKA ausgestattet ist.

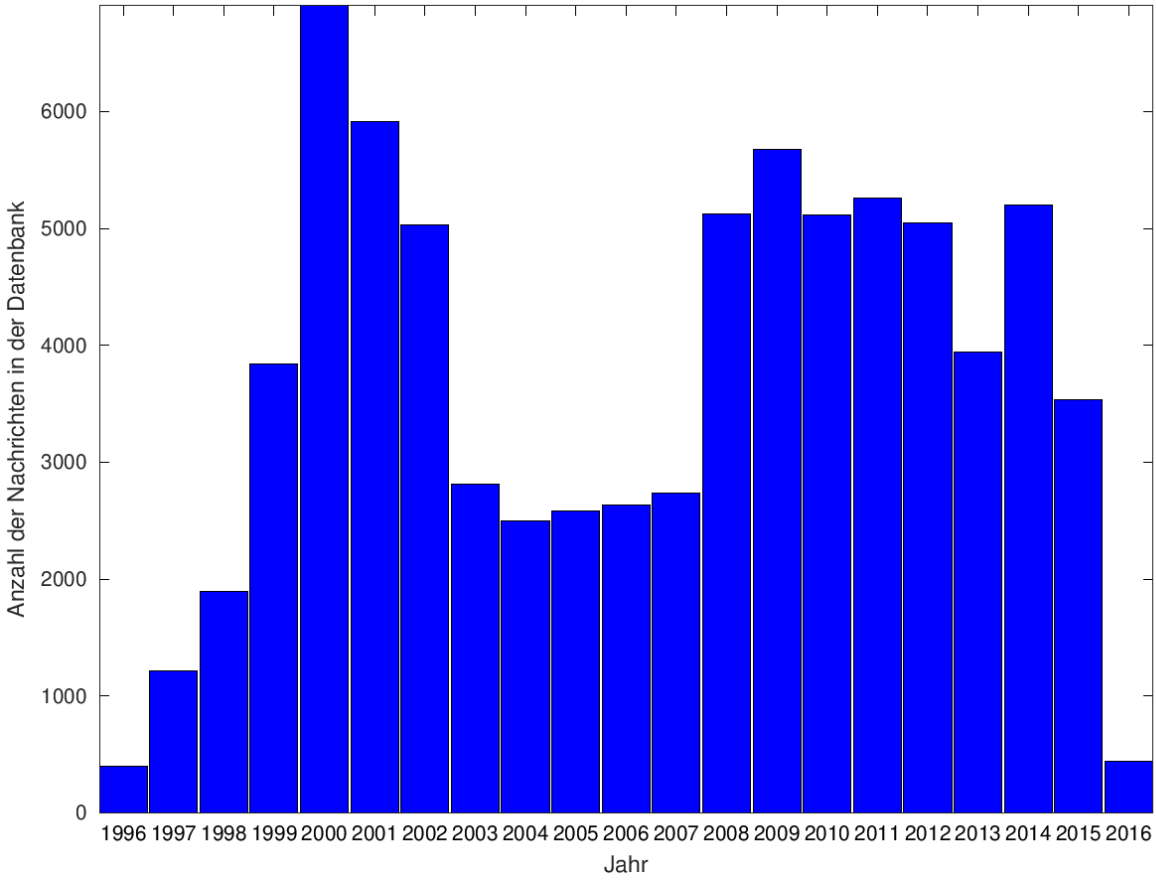

Abbildung 2: Verteilung aller Nachrichten über das Jahr der Veröffentlichung

Diese Voraussetzungen begünstigten die Implementierung des im Folgenden beschriebenen Textmining-Prozesses und die damit verbundenen Experimente.

5.1 Analytische Textaufbereitung

Als Vorverarbeitungsschritt wurden die zu klassifizierenden Ad-hoc-Mitteilungen auf Kollektions- und Dokumentebene gefiltert. Es wurden nur solche Nachrichten behalten, die als in deutscher Sprache verfasst klassifiziert wurden. Dazu wurde ein vortrainierter Textklassifizierer eingesetzt. Zusätzlich wurden die Nachrichten entfernt, für deren Aktie und Veröffentlichungstag keine Kursdaten zur Verfügung standen. Da sie als inhaltlich irrelevant zum Lösen der Aufgabestellung identifiziert wurden, erfolgte zum Zwecke der Rauschunterdrückung auf Dokumentebene ein Herausfiltern der 200 Textzeilen mit den meisten Duplikaten im Gesamtdatensatz. Mit der selben Absicht wurden diverse Zeilenanfänge, die zum Beispiel eindeutig zu einer Anschriftangabe gehörten, als Identifikatoren für zu löschende Zeilen verwendet. Die verbleibenden, bereinigten Dokumente wurden dann durch die analytische Textaufbereitung weiterbehandelt.

Das Tokenizing erfolgte durch einen einfachen Segmenter, der auf den Break-Iterator der Java-API zurückgreift und für die Tokenisierung von Wörtern in deutschen Texten geeignet ist, den „BreakIterator Segmenter" aus der Tool-Sammlung von DKPro-TC. Nach diesem Schritt wurde nicht mehr zwischen Groß- und Kleinschreibung unterschieden. Stoppwörter wurden hier, anders als beim im Konzeptabschnitt vorgestellten Textmining-Prozess nicht getaggt, sondern direkt aus dem zu analysierenden Text entfernt. Dabei wurden übliche deutsche Stoppwörter aus dem Tokenstrom herausgefiltert und zudem die Wörter, für die keine Embeddings vorlagen, um die Vergleichbarkeit zwischen den beiden Dokumentrepräsentierunge mittels Bag-of-Words und Word Embeddings zu gewährleisten. Als morphologischer Analyseschritt wurde für das Bag-of-Words-Modell ein Stemming-Vorgang durchgeführt. Dazu kam der Snowball-Stemmer, eine Implementierung des Stanford-Stemmers für die deutsche Sprache, zum Einsatz.

5.2 Feature-Engineering

Die Repräsentation der Dokumente erfolgte wie bereits erwähnt durch den Bag-of-Words-Ansatz und Word Embeddings, sowie eine Kombination von beiden. Weitere, dem Text fremde Features wurden nicht verwendet. Neumann und Hagenau

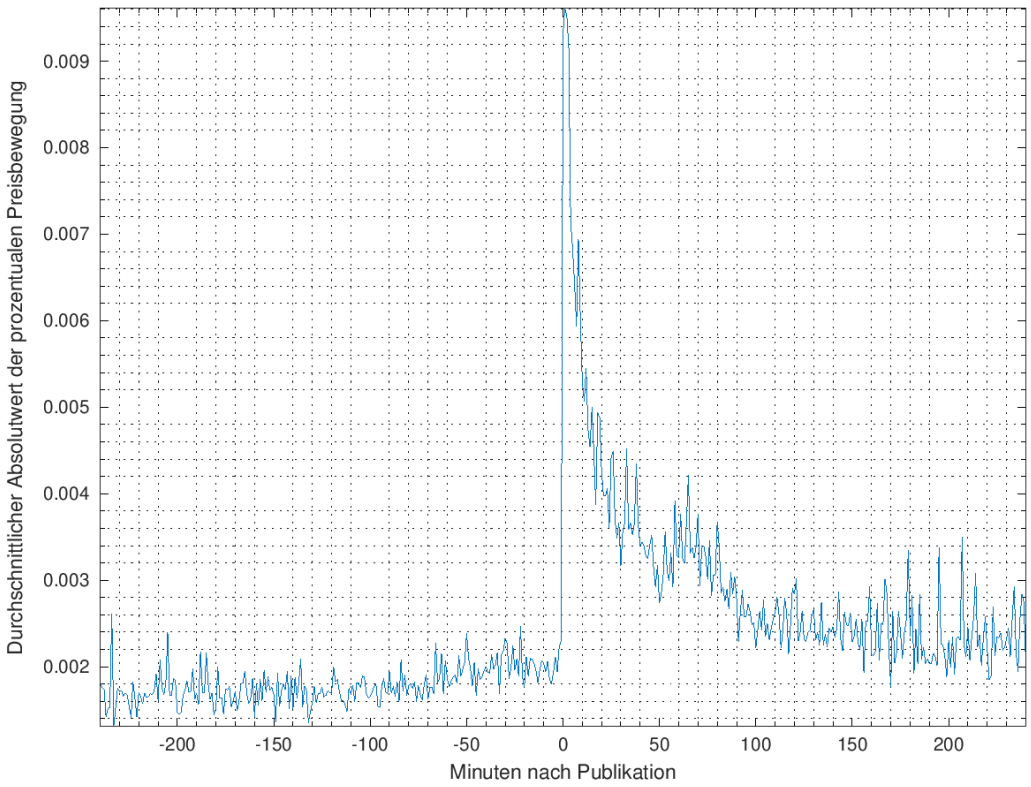

Abbildung 3: Absolutwerte der prozentualen Kursbewegung zur vorherigen Preisfeststellung zu Minuten nach Publikation

haben die Verwendung von Zweiwortphrasen in ihren Untersuchungen als günstige Wahl entdeckt[Hagenau u. a., 2013]. Hier fanden sie keine Anwendung, bieten sich aber als Ausgangspunkt für weiterführende Untersuchungen in der Thematik dieser Bachelorarbeit an. Der hier verwendete Bag-of-Words-Ansatz entspricht dem bereits Abschnitt 2 vorgestellten. Es wurde dabei ein 18774 Einträge umfassendes Wörterbuch verwendet. Diese Anzahl ergab sich aus der Verwendung des Whitelist-Filterns nach Wörtern mit bekannten Word Embeddings und der Verwendung des Stemmers, ohne den das Wörterbuch 25557 Einträge stark wäre. Jedes Wort in einem Dokumentvektor wurde durch seinen TFxIDF-Wert repräsentiert. Dieser wurde mit gewichteten TF und IDF nach Definition 5.1 berechnet. Diese Berechnung von TFxIDF hat den Vorteil, dass lange Dokumente nicht über einen potenziell aufgrund ihrer Länge größeren TF-Wert durch große Werte bei der Berechnung des Relevanzmaßes bevorzugt werden.

Definition 5.1 (Gewichtetes TFxIDF-Maß) *Sei D eine Dokumentkolletion, $d \in D$ Dokument, $t \in d$ ein Wort, n_t die Anzahl der Dokumente mit t in D, und f_t Frequenz von t in d, dann sind die Termfrequenz idf, die inverse Dokumentfrequenz idf und der zugehörige tfidf-Wert definiert als*

$$tf(t,d) := \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$idf(t,D) := log \frac{|D|}{n_t}$$

$$tfidf(t,d,D) := tf(t,d) * df(t,D)$$

Der verwendete Ansatz der Dokumentrepräsentierung mittels Word Embeddings wurde mit nach der Word2vec-Methode vortrainierten Embeddings realisiert. Diese wurden in Rahmen eines Projektes am Fachgebiet UKP der TU Darmstadt erstellt[Reimers u. a., 2014]. Ihr Training erfolgte auf verschiedenen Textdatensammlungen mit circa 116.000.000 Sätzen, darunter die deutsche Wikipedia und die Archive von Spiegel Online und Zeit Online. In die 100 Dimensionen umfassende Einbettung aufgenommen wurden dabei nur Wörter, die mindestens fünfmal im Datensatz vertreten waren. Für die Implementierungen der Experimente dieser Bachelorarbeit wurden aus technischen Gründen die speziellen Embeddings für „UNKNOWN", „<\\s>" und „PADDING" entfernt, sowie die Abbildung für die Zeichenfolge „?R", da sie

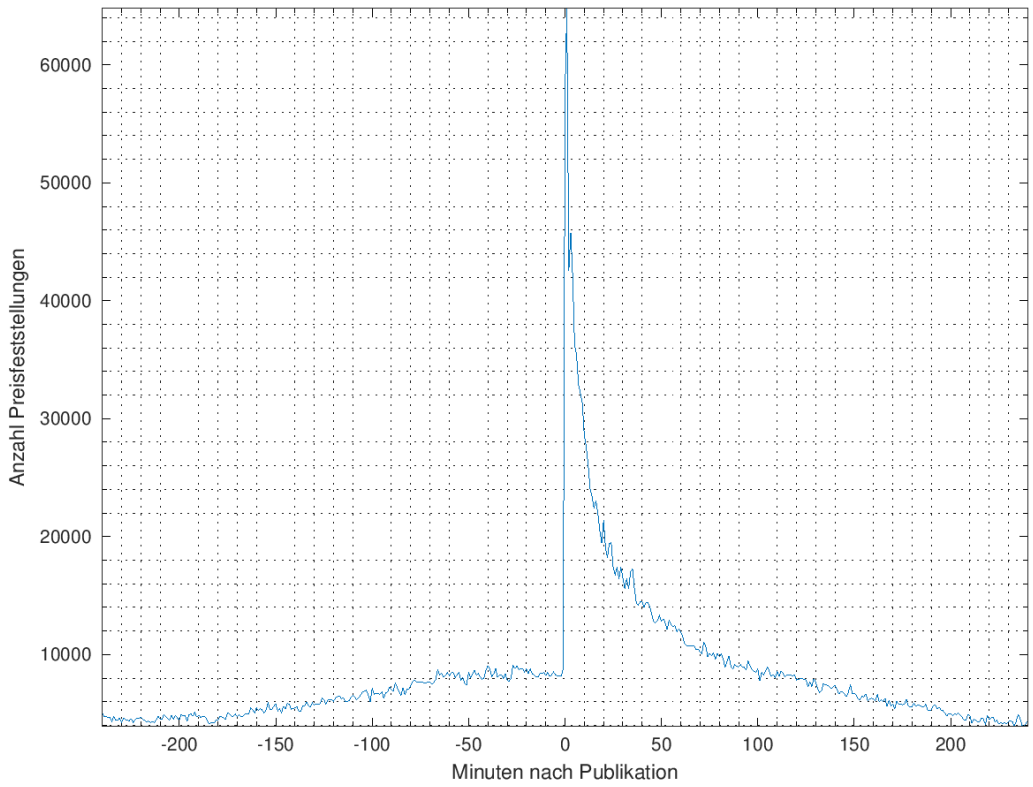

Abbildung 4: Absolute Anzahl der Preisfeststellungen zu Minuten nach Publikation

auf einen Vektor mit NaN-Wert in einer Dimension zielt, welcher nicht weiterverarbeitet werden sollte. Dabei war davon auszugehen, dass durch diese Modifikationen keine Einbußen in der Aussagekraft der gelernten Modelle hingenommen werden müssen, da keine inhaltstragenden Elemente davon betroffen waren. Nach Abzug der aus technischen Gründen entfernten Embeddings, lagen Einbettungen für 3.363.088 Wörter vor. Es kann insbesondere im Hinblick auf die Anzahl der Terme, für die Word Embeddings bekannt sind, lohnend sein, diese auf der eigenen Textsammlung selbst zu erstellen. Davon wurde hier Abstand genommen, da der zur Verfügung stehende Datensatz verhältnismäßig klein war und der dafür zu tätigende zusätzliche Aufwand der Bachelorarbeit nicht angemessen wäre. Somit stellt dieser Arbeitsschritt einen geeigneten Ausgangspunkt für zukünftige Arbeiten, die auf dieser Thesis aufbauen, dar.

Die einzelnen Dokumentvektoren wurden durch die Summierung über die Word Embeddings der in den Dokumenten vertreten Terme erstellt, wobei jedes Wort nach seinem wie in Definition 5.1 spezifierten TFxIDF-Wert gewichtet wurde Wörter, für die keine Embeddings bekannt waren, wurden bei der Summierung wie auch bei der Berechnung der TFxIDF-Werte ignoriert.

Als dritte Methode zur Repräsentation von Dokumenten als Vektoren kam die Kombination der Feature-Extraktion auf Basis von Bag-of-Words und Word Embeddings, wie sie oben beschrieben sind, zum Einsatz. Um das zu erreichen wurde der Bag-of-Words-Vektor eines Textes um den durch Summierung der Word Embeddings gebildeten Dokumentvektor erweitert, es entstand also für jedes Klassifizierungsproblem ein Attributschema mit 19774 Dimensionen.

5.3 Musterextraktion und Label-Engineering

Die Klassifizierungsprobleme, die zum Beantworten der Fragestellungen dieser Bachelorarbeit gelöst werden sollten, ergeben sich aus den oben genannten Fragestellungen und den konkreten Definitionen der zu bewertenden Strategien. Auf letztere soll an dieser Stelle genauer eingegangen werden.

Die Strategien

Wie in der Einleitung zu diesem Abschnitt erwähnt, unterscheiden sich die primitiven Strategien primär dadurch, als Reaktion auf welche Signale das Verlassen einer eingegangene Investitionsposition ausgelöst wird. Das Einnehmen einer solchen Position erfolgt stets in Resonanz auf die Veröffentlichung einer Ad-hoc-Mitteilung als Einstiegssignal. Zur Untersuchung der Fragestellungen dieser Bachelorarbeit wurden drei abstrakte Strategietypen verwendet, konkret jeweils in einer Long- und einer Short-Variante. Diese Typen sind:

1. „HoldN", bei dem die Eingegangene Position so lange gehalten wird, bis n Minuten vergangen sind oder der Handelstag endet;

2. „TrailingStoploss", bei dem eine Stoploss-Order mit anpassendem Stoploss simuliert wird. Die eingegangene Position wird dann verlassen, wenn sich der Kurs der Aktie relativ zum letzten besten Kurs zu $sl\%$ in die für die Investmentposition ungüngstige Richtung bewegt hat oder der Handelstag endet;

3. „TrailingStoplossN", der eine Kombination aus „HoldN" und „TrailingStoploss" darstellt. Dabei wird eine eingenommene Position dann wieder verlassen, wenn n Minuten seit dem Eingehen derselben vergangen sind, sich der Kurs der Aktie um $sl\%$ relativ zum letzten besten Kurs in die für die Investitionsposition ungüngstige Richtung bewegt hat, oder der Handelstag endet, was immer davon als erstes eintritt.

Ein Beispiel: Die Strategie „TrailingStoploss" wird in ihrer Long-Variante angewendet. Der Stoploss-Faktor sei 5% und der Einstiegspreis liege bei 100€ pro Aktie. Der Stoploss-Kurs, bei dessen Unterschreiten das Verlassen der Position ausgelöst wird, liegt also initial bei 95€. Nun steigt der notierte Kurs zwischenzeitlich auf 110€, der Stoploss-Auslöser wird also auf 104.50€ angepasst. Daraufhin sinkt der Kurs wieder auf 103.95€. Der Stoploss feuert und die Position wird zum nächstmöglichen Zeitpunkt wieder verlassen. Zu diesem steht der Kurs auf 103.50€. Die Anwendung der Strategie hat also abzüglich der Transferkosten einen relativen Gewinn von 3.5% eingefahren.

Die Strategieparameter

Zur Wahl geeigneter Parameter n für die Strategietypen mit expliziter Haltezeitbegrenzung und sl für jene, bei denen ein Stoploss verwendet wird, wurden Statistiken angefertigt, bei denen angenommen wurde, die Strategien seien mit einem perfekten Klassifikator benutzt worden. Es wurde dabei auf der Basis des erwarteten Gewinns $n = 180$ Minuten und $sl = 1.5\%$ identifiziert, die einen guten erwarteten Wert dafür gaben. Die Idee dahinter war, durch die Parameterwahl den möglichen Gewinn zu maximieren, während der Klassifikator durch die Identifikation der weniger profitablen Strategieanwendungen unter anderem die Reduktion des Verlustrisikos vornimmt. Gewinn und Verlust durch Strategieanwendungen werden in dieser Arbeit nicht als absolute Größe sondern immer als Rendite, also relativ zur Investition, betrachtet. Da hier keine konkrete Handelssimulation durchgeführt wurde, sondern die Performanz einer Textmining-Methode eruiert werden sollte, wurde stets mit einer unbestimmten Stückzahl von $m \in \mathbb{N}$ gleicher Aktien gehandelt. Dabei wurde vereinfachend angenommen, dass diese zu jeder Zeit gehandelt werden kann. Damit mögliche Fälle, in denen diese Annahme verletzt ist, die Untersuchungsergebnisse nicht verfälschen, wurde deren Anzahl gering gehalten, indem die Untersuchungen mit umsatzstarken Aktien an liquiden Märkten durchgeführt wurden und bei der Selektion besonderes Augenmerk auf prominente Aktien wie beispielsweise den in DAX und MDAX gelisteten gelegt wurde. Lagen für den Veröffentlichungstag einer Aktie Kursdaten sowohl von Xetra als auch von der Börse Frankfurt vor, wurden die Kurse von Xetra aus ebendiesem Grund bevorzugt verwendet, da der über Xetra generierte Umsatz den der Börse Frankfurt deutlich übersteigt.

Preisfeststellungen

Da die erworbenen Kursdaten mit sekundengenauen Zeitstempeln versehen sind aber manche Aktien bisweilen mehrere Male in der Sekunde gehandelt wurden, erfolgte eine Kompression der Kursdaten, bei der für eine Sekunde der Durchschnittspreis über alle in dieser Sekunde getätigten Verkäufe gebildet wurde. Dabei wurde ein festgestellter Preis nach dem gehandelten Volumen gewichtet. Mit den so ermittelten Sekundenkursen wurden die Einstiegs- und Ausstiegspreise für die Strategieanwendungen bestimmt. Als Einstiegskurs wurde der Preis verwendet, welcher zum Veröffentlichungszeitpunkt der zugehörigen Nachricht der zuletzt bekannte war. Da der Zeitstempel der Mitteilungen nur auf die Minute genau auflöst, musste hier zudem eine Ungenauigkeit in Kauf genommen werden. Es kann anhand der gegeben Daten nicht bestimmt werden, zu welcher Sekunde innerhalb der durch den Zeitstempel angegeben Minute die Nachricht veröffentlicht wurde. Im Einklang mit der Ermittlungsmethode für den Einstiegskurs wurde daher angenommen, dass alle Nachrichten zum Anbruch der Minute publik wurden, also zur Sekunde 0. Ein solches Vorgehen induziert einen nicht unerheblichen Bias in die Auswertungen der Strategieanwendungen. Das Eingehen einer Position zum Preis kurz vor der Veröffentlichung einer Ad-hoc-Mitteilung stellt eine Hypereffizienz beim Börsenhandel dar. In der Praxis ist die Zeitspanne vor der Veröffentlichung einer marktrelevanten Information bis zum Platzieren der Order im Markt als Reaktion darauf von großer Bedeutung. Wer schneller als die anderen Marktteilnehmer auf neues Wissen, dem der Markt mit einer Kursanpassung begegnet, reagiert, kann mehr Profit aus ihr schlagen und seine Verluste besser beschränken. Von großer Bedeutung sind dabei die zeitlichen Verzögerungen beim Erwerb der Information, deren Verarbeitung zur Reaktionsentscheidung sowie das Reagieren selbst. Die Minimierung solcher Zeitspannen wird in der Praxis so weit getrieben, dass Rechenzentren institutioneller Marktteilnehmer in physischer Nähe zu denen der Börsen betrieben werden, um die Latenzen im Rechnernetz zu minimieren. Da der Schwerpunkt dieser Arbeit jedoch weniger auf der praktischen Realisierbarkeit als auf der Erforschung des theoretischen Potenzials der Textmining-Methode liegt, ist der durch den erwähnten Bias induzierte Vorteil gegenüber den anderen Marktteilnehmern hier von geringer Relevanz.

Transferkosten wurden beim Kauf und Verkauf der Aktien zu je 0.1% der Kurse angesetzt und mit der Rendite bei der Strategieanwendung verrechnet. Nach Hagenau u.a. ist dies ein realistischer Wert, mit dem vor allem institutionelle Anleger rechnen können[Hagenau u. a., 2013].

Instanzerstellung

Mit dem Wissen um die Ausführungsweisen der Strategien kann nun die Instanzerstellung erarbeitet werden. Da die Fragestellungen 1 und 2 das Vorhersagen der profitabelsten Strategie beinhalten, entsprechen die Klassen der Klassifizierungsprobleme den Bezeichnern von konkreten Strategien. Eine Instanz ist also aus dem Dokumentvektor der Nachricht als Attributvektor und dem Bezeichner der konkreten Strategie aufgebaut. Da deren Parametrisierung für alle Klassifizierungsprobleme die selbe ist, wird sich nicht noch einmal zusätzlich mit angegeben. Ein Bezeichner für die Strategieanwendung von „HoldN" in der Long-Variante mit $n = 15$ wäre also einfach nur „HoldNLong".

Ein typischer so erstellter Datensatz umfasste insgesamt 2900 Instanzen, von denen die älteste im Oktober 2002 und die jüngste im April 2016 datiert waren. Im Schnitt ist also von etwa 207 Strategieanwendungen pro Jahr auszugehen. Eine Verteilung dieser Instanzen über die einzelnen Jahre ist Abbildung 5.3 zu entnehmen.

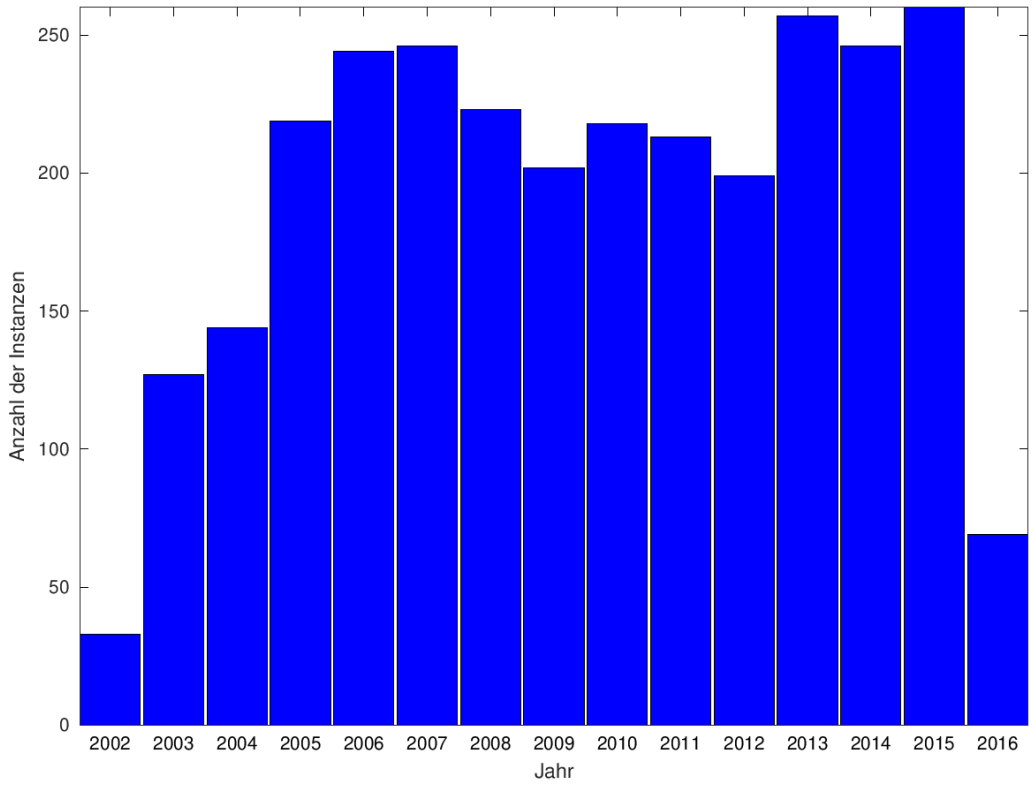

Abbildung 5: Verteilung der Instanzen über die Jahre

5.4 Der Lerner

Nachdem Strategien und Instanzerstellung beschrieben, und die Fragestellungen spezifiziert sind, können nun die konkreten Klassifizierungsprobleme definiert werden, welche für die Untersuchungen dieser Bachelorarbeit erforderlich sind.

Zum Beantworten von Fragestellung 2, für die anhand einer Instanz entschieden werden soll, ob eine Strategie in ihrer Long- oder Short-Variante profitabler ist, bilden diese Varianten die Klassen der binären Probleme, von denen eines für jeden Strategietyp definiert wurde. Analog zum Typ werden die Probleme im Folgenden mit „HoldN(2)", „TrailingStoploss(2)" und „TrailingStoplossN(2)" bezeichnet. Zusätzlich wurde für Fragestellung 1 ein weiteres Problem definiert, bei dem alle Strategietypen in ihren Long- und Short-Varianten als die profitabelste vorhergesagt werden können. Dieses Problem mit acht Klassen wird im Folgenden als All(1) bezeichnet. Eine Übersicht über die Klassifizierungsprobleme, die für die Fragestellungen 2 und 1 gestellt wurden, gibt Tabelle 5.4.

Für Fragestellung 4 ist es sinnvoll, das Ignorieren einer Nachricht zuzulassen, wenn die profitabelste Aktion einen Verlust bedeutet. Dies ist der Fall, sobald die Transferkosten die Rendite dieser Wahl übersteigen. Daher wurde jedes der vier bereits definierten Klassifizierungsprobleme noch einmal mit der zusätzlichen Klasse „IGNORE" behandelt. Analog

Klassifizierungsproblem	Klassen
HoldN(2)	*HoldNLong, HoldNShort*
TrailingStoploss(2)	*TrailingStoplossLong, TrailingStoplossShort*
TrailingStoplossN(2)	*TrailingStoplossNLong, TrailingStoplossNShort*
All(1)	*HoldNLong, HoldNShort, TrailingStoplossLong, TrailingStoplossShort, TrailingStoplossNLong, TrailingStoplossNShort*

Tabelle 6: Klassifizierungsprobleme und zugehörige Klassen für die Fragestellungen 1 und 2

Klassifizierungsproblem	Klassen
HoldN(4)	*HoldNLong, HoldNShort, IGNORE*
TrailingStoploss(4)	*TrailingStoplossLong, TrailingStoplossShort, IGNORE*
TrailingStoplossN(4)	*TrailingStoplossNLong, TrailingStoplossNShort, IGNORE*
All(4)	*HoldNLong, HoldNShort, TrailingStoplossLong, TrailingStoplossShort, TrailingStoplossNLong, TrailingStoplossNShort, IGNORE*

Tabelle 7: Klassifizierungsprobleme und zugehörige Klassen für die Fragestellung 4

dazu werden sie im Folgenden als „HoldN(4)", „TrailingStoploss(4)", „TrailingStoplossN(4)" und All(4) bezeichnet und in Tabelle 5.4 mit ihren Klassen aufgelistet.

Zum Beantworten von Fragestellung 3 wurde jedes der Probleme einmal mit jeder der drei Methoden zur Erstellung der Dokumentvektoren behandelt, also einmal für Bag-of-Words(BoW), einmal für die Methode, welche die Word Embeddings verwendet(WE) und einmal für die Kombination von beiden (BoW+WE). Es resultieren insgesamt 24 einzelne zu untersuchende Klassifizierungsprobleme, welche in 5.4 und 5.4 aufgeführt sind.

Der Lerner

Als Lerner für die vorgestellten Klassifizierungsprobleme kommt eine lineare SVM in der Implementierung der freien Bibliothek LIBLINEAR zum Einsatz, da diese sich besonders für die Klassifizierung von Texten eignet[Fan u. a., 2008]. Die Algorithmen dieser Bibliothek sind auf verschiedene Weisen parametrisiert. Die hier relevanten Freiheitsgrade sind die Fehlerfunktion, für die der L2-Loss verwendet wird, womit das zu lösende Optimierungsproblem dem in Abschnitt 2 definierten gleicht, und ob die Eingabedaten vor der Verarbeitung normalisiert werden sollen. Letztere Option wurde positiv eingestellt, um etwaigen (unwahrscheinlichen) Asymmetrien zwischen den numerischen Repräsentierungen der Word Embeddings und des Bag-of-Words-Ansatzes zu vermeiden. Die Wahl der Hyperparameters C, des dritten Freiheitsgrades, wurde mittels Hyperparameter-Tuning getroffen.

Parametertuning

Die Suche nach einem möglichst guten Hyperparameter C erfolgte nach der Suchstrategie Grid Search. Die Targetvariablen waren dabei für die Probleme der Fragestellung 1 und 2 nach der Vorhersagbarkeit der profitabelsten Strategie die Accuracy und für Probleme der Fragestellung 4 nach dem potenziellen, durch die Textmining-Methode generierbaren Profit der Sharpe Ratio. Als Suchraum für den zu tunenden Parameter wurde die Menge $\{2^n | n \in \mathbb{Z} : n \geq -15 \wedge n \leq 15\}$ verwendet. Das Parameter-Tuning erfolgte für jedes Klassifizierungsproblem auf einer zwei Drittel aller für es verfügbaren Instanzen umfassenden Teilmenge. Die Evaluation im Zuge des Tunings wurde mittels eines Testdatensatzes durchgeführt, welcher für jedes Problem aus einem drittel des Tuning-Datensatzes bestand, dessen verbleibende zwei drittel zum Trainieren der Tuning-Modelle benutzt wurden. Die resultierenden Werte sind für die Probleme der Fragestelungen 1 und 2 Tabelle 8 zu entnehmen und für Probleme der Fragestellung 4 sind sie in Tabelle 9 gelistet.

NOIGN	HoldN(2)	TrailingStoploss(2)	TrailingStoplossN(2)	All(1)
BoW	2.44140625E-4	0.001953125	9.765625E-4	4.8828125E-4
WE	0.0625	0.0625	0.0625	0.0625
BoW+WE	4.8828125E-4	0.001953125	9.765625E-4	0.0078125

Tabelle 8: Ergebnisse des Tunings für den Parameter C der SVM und Probleme ohne IGNORE-Klasse

IGN	HoldN(4)	TrailingStoploss(4)	TrailingStoplossN(4)	All(4)
BoW	0.0078125	0.125	0.125	0.0078125
WE	0.125	0.125	0.0625	0.125
BoW+WE	0.0078125	0.125	0.0625	0.25

Tabelle 9: Ergebnisse des Tunings für den Parameter C der SVM und Probleme mit IGNORE-Klasse

6 Experimente

Um die Fragestellungen dieser Bachelorarbeit zu beantworten wurden Experimente durchgeführt, in denen die im vorherigen Abschnitt spezifizierte Textmining-Methode evaluiert wurde. Dieser Abschnitt widmet sich der Darlegung dieser Experimente. Zunächst werden ihre Methodik und Durchführungsbedingungen erläutert, daraufhin erfolgt die Vorstellung und Interpretation der Ergebnisse

6.1 Methodik der Experimente

Wesentliches Mittel der Experimente dieser Bachelorarbeit ist die Evaluation der durch die Textmining-Methode erstellten Klassifikatoren und Modelle. Diese erfolgte nach dem Prinzip der Evaluation auf einem Testdatensatz. Dazu wurden die Gesamtdatensätze eines jeden Klassifizierungsproblems jeweils in eine zwei Drittel ihrer Instanzen umfassenden Partition der Trainingsdaten und eine aus dem verbleibenden Drittel ihrer Instanzen bestehenden Partition für Testdaten aufgeteilt. Die Partitionen der Trainingsdaten stimmten dabei mit den Datensätzen, welche für das Hyperparameter-Tuning verwendet wurden überein. Die Verteilung der Instanzen über die Partitionen eines Datensatzes erfolgte nach dem Zufallsprinzip, um zu verhindern, dass zeitliche und inhaltliche Zusammenhänge der Instanzen nicht durch etwaige Speicher- und Indexlokalitäten in die Instanzmengen propagiert werden.

An dieser Stelle sei noch einmal auf die gelockerten Durchführungsbedingungen zugunsten der Untersuchungen einer Textmining-Methode hingewiesen. Es wird erstens angenommen, dass die Aktien stets zum letzten bekannten Preis gehandelt werden können. Möglichen Verletzungen dieser Annahme wurde durch die Wahl liquider Aktien und liquider, stark frequentierter Handelsplätze entgegengewirkt. Zweitens werden Laufzeiten und Latenzen beim Eingang und Ausgang von Signalen als nichtig angenommen. Ferner erfolgt die Verarbeitung einer Nachricht durch die Unschärfe ihres Zeitstempels bisweilen vor ihrer tatsächlichen Veröffentlichung. Insbesondere letzteres führt dazu, dass die getätigten Käufe und Verkäufe effektiv mit Insiderhandel gleichzusetzen sind. Beide Eigenschaften sind im Kontext dieser Bachelorarbeit allerdings unerheblich, da die Experimente die Untersuchung der Textmining-Methode zum Gegenstand haben und keine Handelssimulation durchgeführt wird, bei der solche extern zu optimierenden Faktoren eine Rolle spielten.

Um die Vorhersagegüte der gelernter Klassifizierer zu quantifizieren kommt das Performanzmaß Accuracy und wo sinnvoll unterstützend F1 zum Einsatz. Die wirtschaftliche Leistungsfähigkeit der Modelle wird mittels der Durchschnittlichen Rendite und des Sharpe Ratios gemessen.

Die zu schlagenden Benchmarks sind die Vorhersage der Mehrheitsklasse und zudem unter Betrachtung der durch den Handel evaluierten Größen die nach Gleichverteilung randomisierte Auswahl der Klassen. Um die Wirtschaftlichkeit der durch die Klassifikatoren getätigten Handelsentscheidungen im Kontext ihrer Extremwerte einschätzen zu können, wurden die Ergebnisse des perfekten und die des schlechtesten möglichen Entscheiders erfasst. Um den Fragestellungen nach Vorhersagbarkeit und Profitabilität gerecht zu werden, sind die zum Vergleich mit den Benchmarks konsultierten Referenzperformanzmaße für die Klassifizierungsprobleme mit Ignorieren der Sharpe Ratio und für die Probleme ohne Ignorieren einer Nachricht die Accuracy.

6.2 Auswertung der Experimente

Ergebnistabellen

Um die Beantwortung der Fragestellungen dieser Bachelorarbeit zu vereinfachen, wurden die Ergebnisse der Experimente mittels drei Tabellentypen aufbereitet. Für jeden Strategietyp und die Kombination aller Strategien wurde mittels eines

Tabellentyps die Klassifiziererperformanz bezüglich Vorhersagegenauigkeit und wirtschaftlichen Potenzials, mittels eines anderen der Vergleich der Klassifizierer mit ihren Benchmarks und mit Hilfe eines weiteren die Klassenverteilungen von Trainingsdatensatz, Vorhersagen auf dem Testset und Goldstandards im Testset erfasst. Für die Strategietypen „HoldN" sind das die Tabellen 10, 11, 12 in der Reihenfolge ihrer obigen Erwähnungen. Die Experimentergebnisse des Strategietyps „TrailingStoploss" sind entsprechend in den Tabellen 13, 14 und 15 gelistet, die des Typs „TrailingStoplossN" in 16, 17 und 18 und die Ergebnisse für die Kombination aller Strategien „All" werden in 19, 21 und 21 aufgeführt. Zu Zwecken der Übersicht und Vergleichbarkeit wurde zu einem Strategietyp oder der Kombination aller Strategien in jeder der Tabellen die Klassifizierungsprobleme mit und ohne die Möglichkeit Nachrichten zu ignorieren und jedes der drei Attributschemata abgebildet.

Die Tabellen, welche der Frage nach der Klassifiziererperformanz gewidmet sind, geben unter anderem die durchschnittlich erzielte Rendite als „AvgReturn", den größten Profit als „MaxProfit", den größten Verlust als „MaxLoss', den Anteil der profitablen Entscheidungen als „Anteil Hits" und deren durchschnittliche Rendite als „AvgWinReturn", sowie den Anteil der verlustreichen Entscheidungen als „Anteil Misses" und deren durchschnittliche Rendite als „AvgLossReturn" an.

In den Tabellen, welche die Vergleiche mit den Benchmarks darstellen, steht „ExpReturn" für die erwartete Rendite, „Pred" für die Performanz des Klassifikators in der jeweiligen Performanzzahl, „Maj" für die des Benchmarks, bei dem die Mehrheitsklasse aus dem Trainingsdatensatz geraten wird, „Rnd" für die des Benchmarks, der die vorhersagen zufällig traf, und „Max" sowie „Min" für den mit Hilfe der Instanzen des Testdatensatzes maximal respektive minimal zu erreichenden Wert in einer der Evaluationsgrößen.

Der Tabellentyp der Klassenverteilungen gibt die der durch den Lerner getätigten Vorhersagen unter dem Kürzel „Pred" an, die der Testdaten als „Gold" und die des Trainingsdatensatzes als „Train".

Auswertung

Betrachtet man diese Tabellen so fällt zunächst auf, dass die Performanz aller Klassifizierer ohne Kontext auf den ersten Blick als prinzipiell gegeben interpretiert werden könnte. Bezieht man allerdings die Vergleichstabellen mit den Benchmarks dazu mit ein, ändert sich dieses Bild. In allen 24 Fällen wird der Benchmark, der die Mehrheitsklassen rät, nur knapp oder knapp nicht geschlagen, in dem Sinne, dass bezüglich der Accuracy mit einer Ausnahme nicht mehr als 3.5 Prozentpunkte und bezüglich des Sharpe Ratios nicht mehr als 0.05 Punkte Abweichung vom Benchmark zu betrachten sind. Der Benchmark, der nach Gleichverteilung die Klassen auswählt, wird hingegen konsistent übertroffen, was aber angesichts der ungleichen Verteilung der Instanzen in den Testdatensätzen und der Tatsache, dass der Mehrheitsklassenbenchmark ihn ebenfalls immer schlägt, ohne größere Aussagekraft ist.

Betrachtet man den dritten Tabellentyp fällt der Grund für dieses Phänomen auf. Mit Ausnahme der der Klassifizierungsprobleme, bei denen aus allen Strategietypen die profitabelste Strategie vorhergesagt werden soll, sind die Klassenverteilungen in den Testdatensätzen und in den Trainingsdatensätzen ausgewogen in dem Sinne, dass keine Klasse zu mehr als dem vierfachen einer anderen Klasse durch Instanzen repräsentiert ist. Im Unterschied dazu sind bei die Klassenverteilungen stark zulasten der Mehrheitsklasse aus den Trainingsdaten unausgeglichen. Dies lässt darauf zurückschließen, dass die zugehörigen SVMs gelernt haben, fast nur diese Klasse vorherzusagen.

Weil sie zwar gelegentlich beide Benchmarks schlagen, dies aber wie bereits angedeutet im Falle dessen, der die Mehrheitsklasse vorhersagt, nicht substanziell, und zudem auf weiteren Testdatensätzen, die zugunsten der anderen Klassen ungleich strukturiert sind, entsprechend Schlechte Performanzmaße produzieren würden, werden die gelernten Modelle hier als unbrauchbar und die Klassifizierungsprobleme dazu als nicht gelöst betrachtet. Da hiernach alle Probleme ungelöst verbleiben, bei denen Word Embeddings alleine zur Dokumentrepräsentierung benutzt wurden, lässt sich mutmaßen, dass diese für sich nicht ausreichend Information trugen, um nutzbare Muster in den Attributvektoren zu erzeugen. Da die zu repräsentierenden Texte sehr domänenspezifisches Vokabular aufwiesen und die Word Embeddings auf einer allgemeinen Textsammlung erstellt wurden, könnte es sein, dass die Repräsentationen spezifischer Vokabeln aufgrund ihrer geringen Verarbeitungsfrequenz beim Trainieren nur ungenau oder gar nicht abgebildet wurden. Das Lernen von Embeddings auf den Textdaten des Ausgangsdatensatzes könnte hier Abhilfe schaffen.

Die verbleibenden Klassifizierungsprobleme, bei denen nicht zu mehr als 95% nur eine Klasse vorhergesagt wurde, bestehen aus vieren, die Bag-of-Words als einzige Feature-Art verwendeten, und fünf, bei denen eine Kombination von BoW und Word Embeddings die Dokumente repräsentiert. Nicht ein einziges davon unterscheidet nur zwischen der Long- und Short-Variante eines Strategietyps, es haben alle die Möglichkeit zum Ignorieren einer Mitteilung als mögliches Klassifkationsziel. Fragestellung 2 kann nach dem Lackmustest der Klassenverteilung also negativ beantwortet werden.

Unter den verbleibenden Problemen befindet sich TrailingStoplossN(4) mit BoW+WE als Feature-Schema und der Option zu Ignorieren unter den Vorhersagezielen. Dessen Resultate sind bemerkenswert, da es seinen Majority-Benchmark bezüglich des Sharpe Ratios um knapp 25% schlägt, eine Klassenverteilung ähnlich der Verteilung im Testdatensatz aufweist und mit einer Accuracy von 0.507 nicht nur in einer Minderheit der Fälle die richtige Handlungsweise vorhersagt. Der zugehörige Klassifizierer ist zwar bezüglich der Accuracy um 1.5 Prozentpunkte weniger genau als das raten der Mehrheitsklasse, doch ist er dabei bezüglich der durchschnittlichen Rendite um etwa ein drittel profitabler als das

HOLDN	*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Accuracy	0.57200811	0.57707911	0.57200811	0.54259635	0.54259635	0.54563895
F1	0.36387097	0.40794595	0.36821359	0.31193291	0.26886978	0.31789532
AvgReturn	0.01140259	0.01250624	0.01139125	0.01387529	0.01270705	0.01536236
SharpeRatio	0.11643417	0.1279052	0.11631646	0.14171331	0.12999869	0.15723575
MaxProfit	1.3341625	1.3341625	1.3341625	1.3341625	1.3341625	1.3341625
AvgWinReturn	0.04892332	0.0497529	0.04891288	0.05080802	0.04984146	0.05195244
MaxLoss	-0.78844681	-0.78844681	-0.78844681	-0.78844681	-0.78844681	-0.78844681
AvgLossReturn	-0.03256443	-0.03149755	-0.03257683	-0.03044399	-0.03134374	-0.02900932
Anteil Hits	0.53955375	0.54158215	0.53955375	0.54545455	0.54259635	0.54805726
Anteil Misses	0.46044625	0.45841785	0.46044625	0.45454545	0.45740365	0.45194274

Tabelle 10: Performanzmaße für den Strategietyp „HOLDN"

HOLDN		*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Acuracy	*Pred*	0.57200811	0.57707911	0.57200811	0.54259635	0.54259635	0.54563895
	Maj	0.57200811	0.57200811	0.57200811	0.53955375	0.53955375	0.53955375
ExpReturn	*Pred*	0.01140259	0.01250624	0.01139125	0.01387529	0.01270705	0.01536236
	Maj	0.01140259	0.01140259	0.01140259	0.01140259	0.01140259	0.01140259
	Rnd	-0.00193728	-0.0021633	0.0001663	0.00219776	-0.00211194	0.00331738
	Max	0.03964754	0.03964754	0.03964754	0.04589302	0.04589302	0.04589302
	Min	-0.04367437	-0.04367437	-0.04367437	-0.04367437	-0.04367437	-0.04367437
SharpeRatio	*Pred*	0.11643417	0.1279052	0.11631646	0.14171331	0.12999869	0.15723575
	Maj	0.11643417	0.11643417	0.11643417	0.11643417	0.11643417	0.11643417
	Rnd	-0.01957525	-0.02186135	0.00168123	0.02053602	-0.02120483	0.03164844
	Max	0.4420217	0.4420217	0.4420217	0.48449927	0.48449927	0.48449927
	Min	-0.48647753	-0.48647753	-0.48647753	-0.48647753	-0.48647753	-0.48647753

Tabelle 11: Benchmark-Vergleiche für den Strategietyp „HOLDN"

statische Handeln nach dieser und hat ebenso einen höheren Sharpe Ratio, nach dem auch der Hyperparameter der trainierten SVM getunt wurde, geliefert. Mit Hilfe dieses Klassifizierers ließe sich folglich in der Theorie ein profitables Handelssystem ableiten, was Fragestellung 4 beantwortet. Nichtsdestoweniger sei festgehalten, dass der Sharpe Ratio zwar positiv und besser als der des Majority-Benchmarks ist, aber dennoch klein. Dies weist auf ein hohes Risiko bei der Anlage hin, was einer hohen Varianz in den Renditen geschuldet ist. Mindestens fraglich allerdings ist, ob sich die Anwendung der Textmining-Methode hinreichend skalieren lässt, dass die für ein ebenfalls in der Praxis profitables Handelssystem notwendigen statistischen Effekte wie etwa das schwache Gesetz der großen Zahlen greifen. Zudem empfiehlt sich angesichts der mageren Ausbeute bei den anderen Klassifizierungsproblemen eine weitere Untersuchung der Stabilität des Ansatzes bezüglich seiner Vorhersagen, etwa durch eine Kreuzvalidierung oder der Evaluation auf einem weiteren Testdatensatz.

Bezüglich der Eignung spezifischer Textrepräsentationen als Features gibt es keine weiteren Klarheiten. Es ist möglich, lässt sich aber beim gegebenen Wissensstand nicht belegen, dass die Word Embeddings in diesem Szenario keinen Nutzen bringen und vom Klassifizierer als Anhängsel an den eventuell aussagekräftigeren Bag-of-Words-Ansatz ignoriert werden. Offensichtlich ist aber, dass die Embeddings allein hier nicht von Nutzen waren, da ohne Bag-of-Words keine aussagekräftigen Muster in den Daten gefunden werden konnten. Die Textmining-Methode hat abschließend keine Mittel geliefert, mit denen zwischen der Long- und Short-Variante einer Strategie anhand von Nachrichten zu Aktien entschieden werden konnte. Die Vorhersagbarkeit einer konkreten Strategie aus der Menge aller definierten war ebenfalls nicht hinreichend stabil möglich. Während der Lerner für das entsprechende Klassifizierungsproblem All(4) mit BoW+WE zwar seinen Mehrheitsklassenbenchmark bezüglich des Sharpe Ratios schlägt, bleibt die Vorhersagegenauigkeit für die konkrete Strategie hinter dem Raten der Mehrheitsklasse zurück und mit einem Accuracy-Wert von 37.2% deutlich außerhalb eines Bereichs, in dem sich im gegebenen Szenario eine zuverlässige Klassifizierungsgenauigkeit definieren ließe.

HOLDN		*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
HoldNLong	*Pred*	1	0.96247465	0.9959432	0.87626775	0.96044625	0.88032454
	Gold	0.57200811	0.57200811	0.57200811	0.53955375	0.53955375	0.53955375
	Train	0.5830721	0.5830721	0.5830721	0.53866249	0.53866249	0.53866249
HoldNShort	*Pred*	0	0.03752535	0.0040568	0.11663286	0.03955375	0.11156187
	Gold	0.42799189	0.42799189	0.42799189	0.32860041	0.32860041	0.32860041
	Train	0.4169279	0.4169279	0.4169279	0.31870428	0.31870428	0.31870428
IGNORE	*Pred*	N/A	N/A	N/A	0.00709939	0	0.00811359
	Gold	N/A	N/A	N/A	0.13184584	0.13184584	0.13184584
	Train	N/A	N/A	N/A	0.14263323	0.14263323	0.14263323

Tabelle 12: Klassenverteilungen für den Strategietyp „HOLDN"

TSL	*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Accuracy	0.65212982	0.64300203	0.65111562	0.4979716	0.52941176	0.5
F1	0.41847513	0.40953664	0.41802958	0.39347179	0.25955968	0.39679252
AvgReturn	0.01226759	0.01175782	0.01228669	0.01179661	0.01222322	0.01174646
SharpeRatio	0.13243719	0.12701581	0.13264949	0.12482245	0.13193846	0.12456304
MaxProfit	1.4965	1.4965	1.4965	1.4965	1.4965	1.4965
AvgWinReturn	0.04758757	0.04755104	0.04760097	0.04883827	0.04753889	0.04877884
MaxLoss	-0.67029655	-0.67029655	-0.67029655	-1.05222951	-0.67029655	-1.05222951
AvgLossReturn	-0.02746738	-0.02753856	-0.02744188	-0.02950884	-0.02750691	-0.02934184
Anteil Hits	0.52941176	0.52332657	0.52941176	0.52721088	0.52941176	0.52595937
Anteil Misses	0.47058824	0.47667343	0.47058824	0.47278912	0.47058824	0.47404063

Tabelle 13: Performanzmaße für den Strategietyp „TSL"

TSL		*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Acuracy	*Pred*	0.65212982	0.64300203	0.65111562	0.4979716	0.52941176	0.5
	Maj	0.64503043	0.64503043	0.64503043	0.52332657	0.52332657	0.52332657
ExpReturn	*Pred*	0.01226759	0.01175782	0.01228669	0.01179661	0.01222322	0.01174646
	Maj	0.01198351	0.01198351	0.01198351	0.01198351	0.01198351	0.01198351
	Rnd	0.00116496	0.00177871	0.00156856	-0.00565902	-0.00240288	-0.00368899
	Max	0.03637017	0.03637017	0.03637017	0.04464302	0.04464302	0.04464302
	Min	-0.03557176	-0.03557176	-0.03557176	-0.03557176	-0.03557176	-0.03557176
SharpeRatio	*Pred*	0.13243719	0.12701581	0.13264949	0.12482245	0.13193846	0.12456304
	Maj	0.12943258	0.12943258	0.12943258	0.12943258	0.12943258	0.12943258
	Rnd	0.01213855	0.02218093	0.0168218	-0.07364443	-0.02839572	-0.04299583
	Max	0.4022197	0.4022197	0.4022197	0.46027201	0.46027201	0.46027201
	Min	-0.55889459	-0.55889459	-0.55889459	-0.55889459	-0.55889459	-0.55889459

Tabelle 14: Benchmark-Vergleiche für den Strategietyp „TSL"

TSL		*ohne IGNORE*			*mit IGNORE*		
		BoW	WE	BoW+WE	BoW	WE	BoW+WE
TrailingStopLossLong	*Pred*	0.98884381	0.98377282	0.98782961	0.6653144	0.96957404	0.6693712
	Gold	0.64503043	0.64503043	0.64503043	0.52332657	0.52332657	0.52332657
	Train	0.64942529	0.64942529	0.64942529	0.53500522	0.53500522	0.53500522
TrailingStopLossShort	*Pred*	0.01115619	0.01622718	0.01217039	0.22920892	0.03042596	0.22920892
	Gold	0.35496957	0.35496957	0.35496957	0.30831643	0.30831643	0.30831643
	Train	0.35057471	0.35057471	0.35057471	0.29937304	0.29937304	0.29937304
IGNORE	*Pred*	N/A	N/A	N/A	0.10547667	0	0.10141988
	Gold	N/A	N/A	N/A	0.168357	0.168357	0.168357
	Train	N/A	N/A	N/A	0.16562173	0.16562173	0.16562173

Tabelle 15: Klassenverteilungen für den Strategietyp „TSL"

TSLN	*ohne IGNORE*			*mit IGNORE*		
	BoW	WE	BoW+WE	BoW	WE	BoW+WE
Accuracy	0.64908722	0.64097363	0.64503043	0.4969574	0.52028398	0.50709939
F1	0.41464603	0.40366164	0.40013489	0.40066856	0.24552915	0.38952003
AvgReturn	0.01122855	0.01058217	0.01086245	0.01323585	0.01105459	0.01357045
SharpeRatio	0.1240434	0.11696572	0.12012358	0.15559095	0.12187606	0.14873841
MaxProfit	1.4965	1.4965	1.4965	1.4965	1.4965	1.4965
AvgWinReturn	0.044018	0.0439777	0.0438777	0.04660678	0.04457187	0.04720782
MaxLoss	-0.67029655	-0.67029655	-0.67029655	-0.55016667	-0.67029655	-0.55016667
AvgLossReturn	-0.02536054	-0.02549064	-0.02538407	-0.02393274	-0.02529714	-0.02444057
Anteil Hits	0.52738337	0.51926978	0.52332657	0.52691867	0.52028398	0.53052164
Anteil Misses	0.47261663	0.48073022	0.47667343	0.47308133	0.47971602	0.46947836

Tabelle 16: Performanzmaße für den Strategietyp „TSLN"

TSLN		*ohne IGNORE*			*mit IGNORE*		
		BoW	WE	BoW+WE	BoW	WE	BoW+WE
Acuracy	*Pred*	0.64908722	0.64097363	0.64503043	0.4969574	0.52028398	0.50709939
	Maj	0.64401623	0.64401623	0.64401623	0.52231237	0.52231237	0.52231237
ExpReturn	*Pred*	0.01122855	0.01058217	0.01086245	0.01323585	0.01105459	0.01357045
	Maj	0.01092318	0.01092318	0.01092318	0.01092318	0.01092318	0.01092318
	Rnd	-0.00111549	-0.00251625	-0.00087898	0.0007056	-0.00193467	0.00683574
	Max	0.03360758	0.03360758	0.03360758	0.04117102	0.04117102	0.04117102
	Min	-0.03339141	-0.03339141	-0.03339141	-0.03339141	-0.03339141	-0.03339141
SharpeRatio	*Pred*	0.1240434	0.11696572	0.12012358	0.15559095	0.12187606	0.14873841
	Maj	0.12074681	0.12074681	0.12074681	0.12074681	0.12074681	0.12074681
	Rnd	-0.01321825	-0.03372019	-0.01129473	0.00866665	-0.01998827	0.10283848
	Max	0.390638	0.390638	0.390638	0.4453933	0.4453933	0.4453933
	Min	-0.52376109	-0.52376109	-0.52376109	-0.52376109	-0.52376109	-0.52376109

Tabelle 17: Benchmark-Vergleiche für den Strategietyp „TSLN"

TSLN		*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
TrailingStopLossNLong	*Pred*	0.98884381	0.98681542	0.99492901	0.64705882	0.97768763	0.70283976
	Train	0.65151515	0.65151515	0.65151515	0.52664577	0.52664577	0.52664577
	Gold	0.64401623	0.64401623	0.64401623	0.52231237	0.52231237	0.52231237
TrailingStopLossNShort	*Pred*	0.01115619	0.01318458	0.00507099	0.23833671	0.02231237	0.21095335
	Train	0.34848485	0.34848485	0.34848485	0.29937304	0.29937304	0.29937304
	Gold	0.35598377	0.35598377	0.35598377	0.30933063	0.30933063	0.30933063
IGNORE	*Pred*	N/A	N/A	N/A	0.11460446	0	0.0862069
	Train	N/A	N/A	N/A	0.17398119	0.17398119	0.17398119
	Gold	N/A	N/A	N/A	0.168357	0.168357	0.168357

Tabelle 18: Klassenverteilungen für den Strategietyp „TSLN"

ALL	*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Accuracy	0.42799189	0.42900609	0.42596349	0.37221095	0.38032454	0.31338742
F1	0.0999053	0.10294479	0.13684602	0.12686668	0.09009578	0.18254386
AvgReturn	0.01198351	0.01193258	0.01399369	0.01430361	0.01274203	0.00949923
SharpeRatio	0.12943258	0.12893296	0.14972813	0.15156149	0.13711898	0.09776311
MaxProfit	1.4965	1.4965	1.4965	1.4965	1.4965	1.4965
AvgWinReturn	0.04781957	0.04761915	0.04910935	0.05051343	0.04820878	0.04819248
MaxLoss	-0.67029655	-0.67029655	-0.43014286	-0.43014286	-0.67029655	-1.05222951
AvgLossReturn	-0.02735991	-0.02740633	-0.02583447	-0.02580813	-0.02732085	-0.0311894
Anteil Hits	0.52332657	0.52434077	0.53144016	0.52556237	0.53042596	0.51256831
Anteil Misses	0.47667343	0.47565923	0.46855984	0.47443763	0.46957404	0.48743169

Tabelle 19: Performanzmaße für den Strategietyp „ALL"

ALL		*ohne IGNORE* BoW	WE	BoW+WE	*mit IGNORE* BoW	WE	BoW+WE
Acuracy	*Pred*	0.42799189	0.42900609	0.42596349	0.37221095	0.38032454	0.31338742
	Maj	0.42799189	0.42799189	0.42799189	0.37525355	0.37525355	0.37525355
ExpReturn	*Pred*	0.01198351	0.01193258	0.01399369	0.01430361	0.01274203	0.00949923
	Maj	0.01198351	0.01198351	0.01198351	0.01198351	0.01198351	0.01198351
	Rnd	0.0045162	-0.00206945	-0.00148973	0.00296408	0.0023448	-0.0031414
	Max	0.04457002	0.04457002	0.04457002	0.04880642	0.04880642	0.04880642
	Min	-0.04914169	-0.04914169	-0.04914169	-0.04914169	-0.04914169	-0.04914169
SharpeRatio	*Pred*	0.12943258	0.12893296	0.14972813	0.15156149	0.13711898	0.09776311
	Maj	0.12943258	0.12943258	0.12943258	0.12943258	0.12943258	0.12943258
	Rnd	0.0462858	-0.02521288	-0.01489589	0.03272168	0.02677487	-0.03456677
	Max	0.46870043	0.46870043	0.46870043	0.49654714	0.49654714	0.49654714
	Min	-0.54143998	-0.54143998	-0.54143998	-0.54143998	-0.54143998	-0.54143998

Tabelle 20: Benchmark-Vergleiche für den Strategietyp „ALL"

ALL		*ohne IGNORE*			*mit IGNORE*		
		BoW	WE	BoW+WE	BoW	WE	BoW+WE
HoldNLong	*Pred*	0	0	0.01217039	0.01115619	0	0.06490872
	Gold	0.10344828	0.10344828	0.10344828	0.10243408	0.10243408	0.10243408
	Train	0.09822362	0.09822362	0.09822362	0.09665622	0.09665622	0.09665622
HoldNShort	*Pred*	0	0.00811359	0.07707911	0.11257606	0.02738337	0.20486815
	Gold	0.2484787	0.2484787	0.2484787	0.22312373	0.22312373	0.22312373
	Train	0.23145246	0.23145246	0.23145246	0.21212121	0.21212121	0.21212121
TrailingStopLossLong	*Pred*	1	0.99087221	0.89046653	0.83874239	0.97160243	0.53549696
	Gold	0.42799189	0.42799189	0.42799189	0.37525355	0.37525355	0.37525355
	Train	0.44514107	0.44514107	0.44514107	0.38349007	0.38349007	0.38349007
TrailingStopLossShort	*Pred*	0	0.0010142	0.01014199	0.01318458	0.0010142	0.06186613
	Gold	0.09127789	0.09127789	0.09127789	0.09026369	0.09026369	0.09026369
	Train	0.09926855	0.09926855	0.09926855	0.09665622	0.09665622	0.09665622
TrailingStopLossNLong	*Pred*	0	0	0.00912779	0.01521298	0	0.05375254
	Gold	0.09432049	0.09432049	0.09432049	0.09127789	0.09127789	0.09127789
	Train	0.09665622	0.09665622	0.09665622	0.09508882	0.09508882	0.09508882
TrailingStopLossNShort	*Pred*	0	0	0.0010142	0.0010142	0	0.00709939
	Gold	0.03448276	0.03448276	0.03448276	0.03346856	0.03346856	0.03346856
	Train	0.0292581	0.0292581	0.0292581	0.02873563	0.02873563	0.02873563
IGNORE	*Pred*	N/A	N/A	N/A	0.00811359	0	0.07200811
	Gold	N/A	N/A	N/A	0.0841785	0.0841785	0.0841785
	Train	N/A	N/A	N/A	0.08725183	0.08725183	0.08725183

Tabelle 21: Klassenverteilungen für den Strategietyp „ALL"

7 Zusammenfassung und Ausblick

In dieser Bachelorarbeit wurde untersucht, ob sich mittels eines Textmining-Prozesses aus einer Menge primitiver Aktienhandelsstrategien anhand von Ad-hoc-Mitteilungen die Anwendung der profitabelsten vorhersagen lässt. Weiterhin wurde erforscht, ob der Prozess Mittel bietet, aus denen sich ein profitables Handeln am Aktienmarkt ableiten ließe. Zudem wurden drei Methoden, Texte zu numerischen Vektoren zu transformieren, in diesem Kontext miteinander verglichen.

Während nicht mit hinreichender Genauigkeit die Anwendung der profitabelsten Strategie oder eine ihrer Varianten vorhergesagt werden konnte, wurde dennoch ein Klassifizierer trainiert, der im statistischen mittel profitable Strategien oder den Verzicht auf das Handeln vorschlug, und sich damit in der Theorie zum konstruieren eines profitablen Handelssystems eignet. Aufgrund der durch den niedrigen Sharpe Ratio indizierten hohen Streuungen und der fraglichen Skalierbarkeit des Ansatzes ist die praktische Durchführbarkeit dessen allerdings ungewiss.

Unter den drei untersuchten Transformationsmethoden, mit denen sich aus Texten Dokumentvektoren gewinnen lassen, wurde mit den Word Embeddings eine identifiziert, mit der es nicht gelungen ist, aussagekräftige Klassifikationsmodelle zu trainieren. Es werden daher weiterführende Arbeiten empfohlen, die sich mit der Verwendung von auf dem Nachrichtendatensatz dieser Bachelorarbeit trainierten Word Embeddings mit analogen Fragestellungen befassen, um so potentiell tauglichere Klassifikatoren zu erhalten.

Von den 24 definierten Klassifizierungsproblemen konnten letztendlich nur vier als adäquat gelöst angesehen werden und auch bei diesen ließ die Performanz der gelernten Modelle zu wünschen übrig. Der Grund hierfür wird in einer unzureichenden Repräsentation zugrunde liegender Muster in den Attributen vermutet. Um der prekären Ausbeute beim Lösen der Klassifizierungsprobleme entgegenzuwirken, bietet es sich an, weitere Methoden, Features aus Texten zu extrahieren, in diesem Kontext zu erforschen. Als Beispiel seien die Zweiwortphrasen von Hagenau und Neumann[Hagenau u. a., 2013] angeführt. Eine weitere Möglichkeit, die Aussagekraft der Modelle zu erhöhen, wird in der Gewichtung der Trainingsinstanzen als Funktion der Rendite der mit ihr assoziierten Strategieanwendung gesehen. Dies könnte zu einer Verstärkung bestehender Muster in der Menge der Dokumentrepräsentationen führen, bei welcher der Lerner automatisch auf Nachrichten von höherer Brisanz konzentriert würde und unbedeutenderen Meldungen weniger Relevanz beimäße. Zudem könnte zur Steigerung der Rendite einzelner funktionierender Klassifizierer die Anzahl der gehandelten Aktien variabel anhand der Konfidenz des Lerners bezüglich der zu klassifizierenden Instanzen gewählt werden.

Literatur

[Daxenberger u. a. 2014] Daxenberger, Johannes ; Ferschke, Oliver ; Gurevych, Iryna ; Zesch, Torsten: DKPro TC: A Java-based Framework for Supervised Learning Experiments on Textual Data. In: Toutanova, Kristina (Hrsg.) ; Wu, Hua (Hrsg.): *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Association for Computational Linguistics, Juni 2014, S. 61–66

[Fan u. a. 2008] Fan, Rong-En ; Chang, Kai-Wei ; Hsieh, Cho-Jui ; Wang, Xiang-Rui ; Lin, Chih-Jen: LIBLINEAR: A Library for Large Linear Classification. In: *Journal of Machine Learning Research* 9 (2008), S. 1871–1874. – URL http://doi.acm.org/10.1145/1390681.1442794

[Groth und Muntermann 2009] Groth, Sven S. ; Muntermann, Jan: Supporting Investment Management Processes with Machine Learning Techniques. In: *Business Services: Konzepte, Technologien, Anwendungen. 9. Internationale Tagung Wirtschaftsinformatik 25.-27. Februar 2009, Wien*, URL http://aisel.aisnet.org/wi2009/107, 2009, S. 275–286

[Hagenau u. a. 2013] Hagenau, Michael ; Liebmann, Michael ; Neumann, Dirk: Automated news reading: Stock price prediction based on financial news using context-capturing features. In: *Decision Support Systems* 55 (2013), Nr. 3, S. 685–697. – URL https://doi.org/10.1016/j.dss.2013.02.006

[Joachims 1998] Joachims, Thorsten: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In: *Machine Learning: ECML-98, 10th European Conference on Machine Learning, Chemnitz, Germany, April 21-23, 1998, Proceedings*, URL https://doi.org/10.1007/BFb0026683, 1998, S. 137–142

[Li u. a. 2012] Li, Jing ; Cheng, Ji-hang ; Shi, Jing-yuan ; Huang, Fei: *Brief Introduction of Back Propagation (BP) Neural Network Algorithm and Its Improvement*. S. 553–558. In: Jin, David (Hrsg.) ; Lin, Sally (Hrsg.): *Advances in Computer Science and Information Engineering: Volume 2*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. – URL https://doi.org/10.1007/978-3-642-30223-$7_8$7. – –$ISBN$978 – 3 – 642 – 30223 – 7

[Lilleberg u. a. 2015] Lilleberg, J. ; Zhu, Y. ; Zhang, Y.: Support vector machines and Word2vec for text classification with semantic features. In: *2015 IEEE 14th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC)*, July 2015, S. 136–140

[Malkiel und Fama 1970] Malkiel, Burton G. ; Fama, Eugene F.: EFFICIENT CAPITAL MARKETS: A REVIEW OF THEORY AND EMPIRICAL WORK*. In: *The Journal of Finance* 25 (1970), Nr. 2, S. 383–417. – URL http://dx.doi.org/10.1111/j.1540-6261.1970.tb00518.x. – ISSN 1540-6261

[Mikolov u. a. 2013] Mikolov, Tomas ; Chen, Kai ; Corrado, Greg ; Dean, Jeffrey: *Efficient estimation of word representations in vector space*. 2013

[Mittermayer 2004] Mittermayer, M. A.: Forecasting Intraday stock price trends with text mining techniques. In: *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, Jan 2004, S. 10 pp.–

[Muntermann und Guettler 2007] Muntermann, Jan ; Guettler, Andre: Intraday stock price effects of ad hoc disclosures: the German case. In: *Journal of International Financial Markets, Institutions and Money* 17 (2007), Nr. 1, S. 1 – 24. – URL http://www.sciencedirect.com/science/article/pii/S1042443105000582. – ISSN 1042-4431

[Ramos 2003] Ramos, Juan: Using TF-IDF to determine word relevance in document queries. (2003), 01

[Reimers u. a. 2014] Reimers, Nils ; Eckle-Kohler, Judith ; Schnober, Carsten ; Kim, Jungi ; Gurevych, Iryna: GermEval-2014: Nested Named Entity Recognition with Neural Networks. In: Faass, Gertrud (Hrsg.) ; Ruppenhofer, Josef (Hrsg.): *Workshop Proceedings of the 12th Edition of the KONVENS Conference*, Universitätsverlag Hildesheim, Oktober 2014, S. 117–120

Abbildungsverzeichnis

Tabellenverzeichnis