Masterarbeit

Machine Learning in Ausschreibungsunterlagen

Identifikation relevanter Segmente bezüglich Zuschlagskriterien in Ausschreibungsunterlagen

eingereicht an der
Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität Bern

Institut für Wirtschaftsinformatik
Information Management

Dr. Matthias Stürmer

eingereicht von
Janik Endtner
von St. Gallen, SG
Matrikelnummer: 12-114-674

Pestalozzistrasse 11
3600 Thun
079 441 68 02
janik-endtner@hotmail.com

Bern, 12. Juli 2019

Zusammenfassung

In dieser Masterarbeit wird ein Verfahren entwickelt, um relevante Segmente bezüglich Zuschlagskriterien in Ausschreibungsunterlagen zu finden. Dazu wird der RandomForest Algorithmus mit Topic Modeling und Active Learning kombiniert. Mit einer geschichteten Zufallsauswahl mittels Topic Modeling wird die stark unausgeglichene Verteilung der positiven und negativen Klasse in den Daten adressiert. Damit kann die Anzahl der positiven Einträge im Trainingsdatensatz gegenüber einer zufälligen Auswahl erhöht werden. Parallel zum kombinierten Verfahren werden drei simplere Modelle implementiert und evaluiert. Bei diesen handelt es sich um ein RandomForest Modell ohne Topic Modeling und Active Learning, um ein Modell, das eine Klassifizierung direkt mit den Topic Models durchführt und um ein simples, auf Regex basierendes Patternmatching Modell. Es kann gezeigt werden, dass das kombinierte Modell mit einem Macro F_1 Score von 0.77 besser abschliesst als das Patternmatching Modell (Macro F_1 Score von 0.66), das Topic Modeling (Macro F_1 Score von 0.67) und das simple RandomForest Modell (Macro F_1 Score von 0.52). Mit Active Learning kann vor allem der F_1 Score der positiven Klasse erhöht werden. Über zehn Runden Active Learning steigt dieser relativ konstant.

In der Arbeit wird ausserdem ein Vorschlag gemacht, wie hierarchisch aufgebaute Textdokumente als Grundlage für Machine Learning Algorithmen dienen können.

Inhaltsverzeichnis

Glossar

Bag-of-Words Repräsentation	Darstellung eines Textes als eindeutige Worte im Text und deren absoluten Häufigkeit (vgl. Kapitel 2.1.3).
Confusion-Matrix	Darstellung von TP, TN, FP und FN in einer Matrix.
F_1 Score	Harmonisches Mittel zwischen Recall und Precision (vgl. Kapitel 2.2.3). Gütemass zwischen 0 und 1.
False Positives (FP)	Einträge, die vom Modell in die positive Klasse eingeteilt wurden, aber effektiv zur negativen Klasse gehören.
False Negatives (FN)	Einträge, die vom Modell in die negative Klasse eingeteilt wurden, aber effektiv zur positiven Klasse gehören.
Labeln	Manuelles Zuweisen von Einträgen zu einer Klasse. In dieser Arbeit: Kapitel werden der positiven oder der negativen Klasse zugeordnet.
Negative Klasse	In dieser Arbeit: Klasse der nicht relevanten Kapitel.
Nicht relevante Kapitel	Kapitel, die bezüglich Zuschlagskriterien nicht relevant sind.
Nicht-Zuschlag-Topics	Generierte Topics vom Topic Modeling, die nicht hauptsächlich Kapitel zu Zuschlagskriterien enthalten (vgl. Kapitel 2.2.3).
Overfitting	Modell, das für Trainingsdaten sehr gut performt, für die Testdaten aber sehr schlecht (Modell ist nicht generalisierbar).
Positive Klasse	In dieser Arbeit: Klasse der relevanten Kapitel.
Relevante Kapitel	Kapitel, die bezüglich Zuschlagskriterien relevant sind.
True Negatives (TN)	Einträge, die vom Modell in die negative Klasse eingeteilt wurden und effektiv der negativen Klasse angehören.
True Positives (TP)	Einträge, die vom Modell in die positive Klasse eingeteilt wurden und effektiv der positive Klasse angehören.
Unausgeglichener Datensatz	Datensatz, in dem mehr Einträge einer Klasse vorhanden sind als von einer anderen. In dieser Arbeit: Datensatz mit mehr Daten der negativen Klasse als der positiven.
Zuschlag-Topic	Generierte Topics vom Topic Modeling, die hauptsächlich Kapitel zu Zuschlagskriterien enthalten (vgl. Kapitel 2.2.3).

1 Einleitung

1.1 Ausgangslage

Simap.ch ist eine Plattform, auf der öffentliche Ausschreibungen der Schweiz publiziert werden. Alle Bundesorganisationen sowie die meisten kantonalen und lokalen Verwaltungen sind gesetzlich verpflichtet, Ausschreibungen auf Simap.ch zu veröffentlichen, falls diese einen Wert von 250'000 Schweizer Franken übersteigen (Krancher & Stürmer, 2018). Für jede Ausschreibung wird ein Meldungstext veröffentlicht, der ohne Anmeldung einsehbar ist und einen Überblick über einige Punkte der Ausschreibung gibt. Ausserdem werden Ausschreibungsunterlagen hochgeladen, welche alle für die Ausschreibung relevanten Informationen enthalten. Diese Unterlagen können nach dem Login heruntergeladen werden, solange die Frist für die Entgegennahme des Angebots noch nicht abgelaufen ist. Beim offenen Verfahren darf diese Frist in der Regel nicht kürzer als 40 Tage sein (N.N., 1996).

Für die Erarbeitung der Ausschreibungsunterlagen werden meist zahlreiche Arbeitsstunden investiert. Dementsprechend viel Wissen ist in den Dokumenten der Ausschreibungsunterlagen enthalten, welches nach dem Abschluss des Projekts verloren geht, oder zumindest nicht wiederverwendet wird. International existieren Plattformen wie Tenderlake (https://www.tenderlake.com/) oder Kompass-Nachhaltigkeit (https://www.kompass-nachhaltigkeit.de/), welche für den Austausch des gewonnenen Wissens sorgen. Kompass-Nachhaltigkeit bietet sogar die Möglichkeit, ganze Kriterienkataloge für spezifische Produktkategorien per Klick zu erstellen. Eine ähnliche Plattform fehlt auf dem Schweizer Markt bisher. Zwar ist es für grössere Beschaffungsstellen grundsätzlich möglich, eigene Ausschreibungen wieder zu verwenden. Eine Analyse der Simap Datenbank zeigt aber, dass 50% aller Beschaffungsstellen bisher maximal drei Ausschreibungen durchgeführt haben (Datenbank FDN, Stand 22. März 2019). Hierbei hatten sie keine Möglichkeit, bisherige Ausschreibungsdokumente als Vorlage zu verwenden.

1.2 Problemstellung

Die Forschungsstelle Digitale Nachhaltigkeit der Universität Bern hat eine Software entwickelt, die Ausschreibungsunterlagen täglich herunterlädt. Seit dem Start dieses Verfahrens im August 2017 ist der Datensatz der Forschungsstelle auf über 10’800 Ausschreibungsunterlagen angewachsen, was einer Datenmenge von rund 830 GB entspricht (Stand IntelliProcure, 26. Juni 2019). Täglich kommen rund 15 Ausschreibungsprojekte mit durchschnittlich 24 Dokumenten dazu. Meistens werden diese Dokumente als PDF veröffentlicht, aber auch andere Formate wie docx, xlsx, PNG, etc. sind möglich. Die heruntergeladenen Dokumente werden mit Elasticsearch indexiert (vgl. Gormley & Tong, 2015) und können über eine API durchsucht werden. Es ist möglich, einzelne oder mehrere Dokumente zu kopieren und weiterzuverarbeiten.

Die Forschungsstelle Digitale Nachhaltigkeit entwickelt eine Plattform (https://intelliprocue.ch), auf der die Benutzer die Ausschreibungsunterlagen nach Stichwörtern durchsuchen oder alle Dokumente zu einem Ausschreibungsprojekt anzeigen und herunterladen können. Die Ausschreibungsprojekte bestehen teilweise aus vielen unterschiedlichen Dokumenten, da sie neben allgemeinen Ausschreibungsunterlagen wie Pflichtenheft, Eignungskriterien oder Zuschlagskriterien noch sehr spezifische Dokumente wie Baupläne, Spezifikationen oder Projektpläne enthalten.

Tabelle 1 zeigt eine mögliche Struktur von Ausschreibungsdokumenten. Diese kann für die Beschaffungsobjekte stark variieren.

Tabelle 1: Beispielhafte Struktur von Ausschreibungsunterlagen eines Beschaffungsobjektes

Kategorie	Dokument	Typ
Pflichtenheft		
	Pflichtenheft	PDF
Anhänge		
	Eignungskriterien	PDF
	Zuschlagskriterien	PDF
	Technische Spezifikationen	PDF
	Preistabelle	xlsx
Beilagen		
	Baupläne	PNG
	Gesetze	PDF
	Systemanforderungen	PDF

Das Pflichtenheft bildet normalerweise den Ausgangspunkt für den Leser der Ausschreibungsunterlagen. Es beschreibt die wichtigsten Punkte, wie den Ablauf des Verfahrens, die Eignungs- und Zuschlagskriterien oder die Preiskurve, zusammenfassend. Daneben existieren Beilagen, die verschiedene Aspekte der Ausschreibung genauer erklären. Oftmals wird vom Pflichtenheft auf die detaillierteren Anhänge und Beilagen verwiesen. Zum Beispiel kann das Pflichtenheft das Kapitel «Zuschlagskriterien» enthalten, wobei der Inhalt des Kapitels lediglich aus einem Verweis besteht (z. B. «siehe Anhang 3, Zuschlagskriterien»). Die Dateien der Ausschreibung bestehen aus halbstrukturierten Formaten wie xlsx, docx oder xml und unstrukturierten Formaten wie PDF oder PNG. Bei PDF-Dateien kann der Text in maschinenlesbarer Form abgespeichert sein. Ist dies nicht der Fall, ist die Extraktion von Text aus den Dokumenten sehr schwierig und rechenintensiv (OCR muss verwendet werden). Auch bezüglich der Gliederung unterscheiden sich die Dokumente stark. Zum Beispiel behandelt das Pflichtenheft viele Themen und ist hierarchisch in Kapitel und Unterkapitel unterteilt (vgl. Abbildung 1). Dagegen existieren andere Dokumente, die nur ein Thema behandeln, und nicht hierarchisch aufgebaut sind (z.B. ein Dokument mit der Auflistung der einzelnen Zuschlagskriterien).

Abbildung 1: Hierarchischer Aufbau eines Pflichtenhefts (Beispiel)

Das Bundesgesetz über öffentliches Beschaffungswesen definiert Teile, die in Ausschreibungen vorkommen müssen (N.N., 1994). Zum Beispiel muss eine Beschaffungsstelle Informationen über die Eignungskriterien und deren Nachweise in den Ausschreibungsunterlagen bekanntgeben (Artikel 9, abs. 2, BöB). Technische Spezifikationen (Artikel 9, abs. 2, BöB) und Zuschlagskriterien (Artikel 21, abs. 2, BöB) müssen ebenfalls beschrieben werden.

1.3 Zielsetzung

Das Ziel dieser Masterarbeit ist es, relevante Segmente zu Zuschlagskriterien zu finden. Ein Verfahren wird entwickelt, das die Ausschreibungsunterlagen in Segmente unterteilt und diese in die Kategorien *relevant* (für Zuschlagskriterien) beziehungsweise *nicht relevant* zuordnet. Es soll soweit skalierbar sein, dass alle bisherigen Ausschreibungsprojekte verarbeitet werden können. Danach soll es täglich auf die neusten Beschaffungsprojekte angewendet werden können.

Das entwickelte Verfahren ist mehrstufig und kombiniert Topic Modeling mit überwachtem Lernen. Ein RandomForest Classifier wird verwendet (Breiman, 2001), welcher mit Active Learning in mehreren Iterationen verbessert wird. Das Resultat dieses kombinierten Ansatzes wird mit verschiedenen simpleren Verfahren verglichen. Ein RandomForest Modell ohne Topic Modeling und Active Learning wird trainiert und dessen Resultate evaluiert. Ausserdem wird die Klassifikation mit dem Topic Modeling direkt durchgeführt. Danach werden die Resultate mit den Ergebnissen eines simplen Regex-basierten Ansatzes verglichen.

In dieser Arbeit wird zudem untersucht, wie ein hierarchischer Aufbau von Kapiteln als Input für die Machine Learning Algorithmen dienen kann. Üblicherweise wird beim Text Mining der Text als Bag-of-Words Repräsentation vorverarbeitet (Aggarwal, 2018, S. 2). Dabei wird die hierarchische Struktur eines Dokuments nicht berücksichtigt. Dies scheint nicht ideal zu sein, da die Überschrift und die Überschrift des Eltern-Elements normalerweise einiges über den Inhalt eines Kapitels aussagen.

Diese Arbeit orientiert sich am wissenschaftlichen Ansatz des Design Science (Hevner, March, Park, & Ram, 2004). Hevner et al. unterscheiden zwischen Behavioral Science und Design Science. Während es bei Behavioral Science darum geht, Theorien zu entwickeln und zu testen, die menschliches oder organisationales Verhalten voraussagen, werden mit Design Science neue und innovative Artefakte erstellt, um damit menschliche und organisationale Grenzen auszuweiten. Mit dem Design Science Ansatz wird Wissen und Verständnis über ein Problem im Verlaufe der Entwicklung des Artefakts gewonnen.
Peffers et al. (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2014) beschreiben sechs Schritte für die erfolgreiche Anwendung von Design Science:

1. Problemstellung und Motivation
2. Definition der Lösungsobjekte
3. Design und Entwicklung
4. Demonstration
5. Evaluation
6. Kommunikation

Die Problemstellung und Motivation sowie die Definition der Lösungsobjekte werden in Kapitel 1 beschrieben. In Kapitel 2 wird auf das Design und die Entwicklung sowie die Demonstration des Verfahrens eingegangen. Die Diskussion in Kapitel 3 entspricht der Evaluation der Resultate. Der sechste Punkt im Design Science Verfahren ist die Kommunikation, welche durch diese Masterarbeit geschieht. Ausserdem wird der gesamte Code auf GitHub unter https://github.com/janikEndtner/machine-learning-intender-documents veröffentlicht.

2 Methodisches Vorgehen

In dieser Arbeit wird ein Verfahren vorgestellt, das relevante Segmente zu Zuschlagskriterien in einer grossen Menge von Textdokumenten identifizieren kann. Dieses Verfahren ist mehrstufig und besteht aus der Vorverarbeitung, dem Topic Modelling und der Klassifizierung mit einem RandomForest Modell.

Die Algorithmen, die für diese Arbeit benötigt werden, werden in Python geschrieben. Um eine bessere Strukturierung und Dokumentation zu ermöglichen, wird für bestimmte Codeteile auf Jupyter Notebook zurückgegriffen. Für die Vorverarbeitung der Textdokumente sowie das Trainieren und Anwenden der Machine Learning Modelle werden die Packages NLTK (vgl. Loper & Bird, 2002) und Scikit-Learn (vgl. Pedregosa et al., 2011) verwendet. Eine MySQL Datenbank wird aufgesetzt, in welcher verarbeitete Kapitel und Zwischenresultate gespeichert werden. Der detaillierte Aufbau der Datenbank ist in Anhang 1 ersichtlich.

Kapitel 2.1 handelt davon, wie die Dokumente vorverarbeitet werden müssen, damit die Machine Learning Algorithmen angewendet werden können. Kapitel 2.2 beschreibt, wie die in Kapitel 2.1.4 generierten Kapitel mit Hilfe von Topic Models in verschiedene thematische Topics eingeteilt werden können. Das Resultat des Topics Modelings dient als Grundlage für die in Kapitel 2.3 beschriebene RandomForest Klassifizierung.

2.1 Vorverarbeitung

Bevor mit dem Topic Modelling und der RandomForest Klassifizierung begonnen werden kann, müssen die Daten vorbereitet werden. Eine grosse Menge an Dokumenten soll vorselektiert werden, damit die Dokumente, die mit grosser Wahrscheinlichkeit keine relevanten Segmente beinhalten, nicht verarbeitet werden müssen. Dieser Vorgang wird in Kapitel 2.1.1 beschrieben. Da die Dokumente meist im unstrukturierten Format PDF Vorliegen, müssen sie in ein strukturierteres Format umgewandelt werden. Kapitel 2.1.2 beschreibt, wie die PDF Dokumente für die weitere Verwendung

in Word Dokumente konvertiert werden. Anschliessend werden die Word Dokumente in einzelne Kapitel unterteilt (Kapitel 2.1.3) und tokenisiert, stoppwortbereinigt und lemmatisiert (Kapitel 2.1.3).

2.1.1 Auswahl der Dokumente und Download

Am 16.4.2016 existierten 9423 Projekte in der Datenbank der Forschungsstelle Digitale Nachhaltigkeit bei welchen die Ausschreibungsunterlagen heruntergeladen wurden (siehe Kapitel 1.2). Diese dienen als Datengrundlage für die vorliegende Arbeit. Die Grösse der Ausschreibungsunterlagen betrug am erwähnten Zeitpunkt rund 720 GB, wobei ein Teil davon Textdateien wie Pflichtenhefte, Excel-Anhänge, etc. sind. Oftmals existieren daneben auch sehr grosse Dateien, insbesondere Baupläne. Um die Grösse des Datensatzes und somit die Rechenzeit zu reduzieren und die Güte der Resultate zu verbessern, werden die Dateien vorselektiert. Ziel dabei ist es, vor allem die ausserordentlich grossen Dateien zu entfernen. Dazu werden verschiedene Methoden getestet:

1. Ausschliessliche Verwendung von kleinen Dateien: Eine Schwelle für die maximale Dateigrösse wird gewählt (z.B. 10 MB). Das Problem dabei ist, dass grosse Dateien wie Baupläne zwar korrekt aussortiert wurden, überdurchschnittlich grosse Textdokumente wie lange Pflichtenhefte aber ebenfalls.
2. *Text Länge / Seiten* Verhältnis: Die Idee dabei ist es, die Länge des Textes zu bestimmen und durch die Anzahl Seiten zu dividieren. Wird eine bestimmte Schwelle unterschritten (z.B. 1000 Zeichen pro Seite), wird das Dokument nicht weiterverwendet. Alternativ wäre es möglich, einzelne Seiten aus den Dokumenten zu löschen, welche das Verhältnis nicht erfüllen. Das Problem dieses Ansatzes ist, dass das Summieren der Textlänge pro PDF Datei die Rechenzeit zu stark verlängert (unter Umständen fünf Minuten für ein grosses Dokument).
3. Verwendung von Dateien, welche bestimmte Wörter enthalten: Wie in Kapitel 1.2 erwähnt, wurden alle Ausschreibungsunterlagen zuvor mit ElasticSearch indexiert. Dieser Index kann nach bestimmten Stichworten durchsucht werden. Wird mindestens eines dieser Stichworte im Text gefunden, wird das

Dokument für die weitere Verarbeitung heruntergeladen. Dieser Ansatz ist erfolgreich und wird für die Arbeit verwendet.

Für jedes der 9423 Projekte wird ein ElasticSearch Request auf die Schnittstelle der Forschungsstelle Digitale Nachhaltigkeit mit folgenden Stichworten gesendet:

{ Eignungskriteri, Zuschlagskriteri*, Zulassungskriteri* }*

Der Stern ist ein Platzhalter für beliebig viele nachfolgende Buchstaben (ausser Leerzeichen). Dokumente, welche z.B. die Wörter «Eignungskriterium», «Eignungskriterien», «Zuschlagskriterien» oder «Zulassungskriterien» beinhalten, werden in dieser Abfrage gefunden. Für jedes Projekt werden die gefundenen Dokumente heruntergeladen und in einer MariaDB als binäre Representation abgespeichert. Insgesamt werden für die 9423 Projekte 10'043 relevante Dokumente selektiert, wobei es möglich ist, dass zu einem Projekt mehrere Dokumente gefunden werden. Zu 4197 Projekten werden keine Dokumente gefunden, womit 5226 Projekte mit Dokumenten verbleiben. Dies kann verschiedene Gründe haben:

- ElasticSearch wird nach deutschen Stichworten durchsucht. Die meisten Ausschreibungsdokumente sind unter anderem auf Deutsch vorhanden. Manchmal wird ein Projekt aber auch nur auf Französisch, Italienisch oder in einer anderen Sprache ausgeschrieben.
- Der Prozess der Indexierung mit ElasticSearch ist noch nicht ausgereift. Manche Dokumente sind schlecht oder gar nicht indexiert. Deshalb kann es sein, dass zu einem Projekt zwar Unterlagen vorhanden sind, diese zuvor aber nicht mit ElasticSearch indexiert wurden.

2.1.2 Umwandlung von PDF zu Word Dateien

Tabelle 2 zeigt eine Übersicht über die Formate der heruntergeladenen Dokumente.

Tabelle 2: Dateiendungen der heruntergeladenen Dokumente

Dateiendung	Anzahl	Prozent
pdf	7258	73%
docx	1501	15%
xlsx	415	4%
01s	410	4%
doc	179	2%
xls	116	1%
docm	109	<1%
xlsm	27	<1%
andere	28	<1%

Mit 7258 Dokumenten ist das Format PDF mit Abstand am meisten vertreten. An zweiter Stelle findet sich das Microsoft Office Word Format (docx). Danach folgen das Microsoft Excel Format (xlsx) und 01s. Letzteres ist ein Datenaustauschformat für den Normpositionen-Katalog NPK im Bauwesen (vgl. https://npkeditor.crb.ch/). Da der Aufwand pro berücksichtigtes Format stark zunimmt und der prozentuale Anteil von PDF und docx Dokumenten zusammen rund 88% beträgt, wird im Rahmen dieser Arbeit nur mit den Formaten PDF und docx gearbeitet.

PDF ist ein Format, das für das menschliche Auge entwickelt wurde. Es ist nicht für die Weiterverarbeitung von Maschinen konzipiert und deshalb maschinell schwierig lesbar. PDF Dateien beinhalten Raster- und Vektorelemente, oftmals aber auch Textelemente. Meistens ist es darum ziemlich trivial, den reinen Text aus PDF Dateien zu extrahieren. Alle strukturierten Elemente wie Überschriften, Tabellen, Absätze, etc. gehen jedoch so verloren (vgl. Abbildung 2 und Abbildung 3).

Dies ist ein Titel

1 Dies ist die Überschrift 1

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

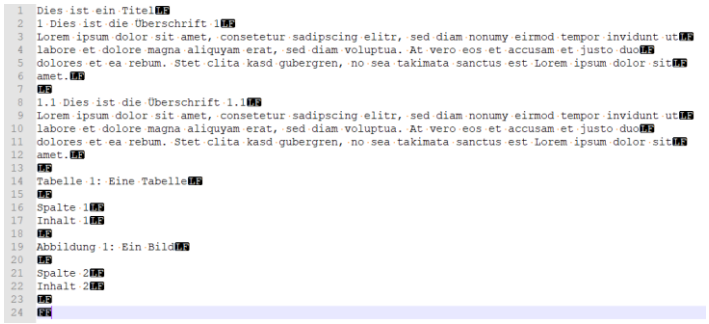
1.1 Dies ist die Überschrift 1.1

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

Tabelle 1: Eine Tabelle

Spalte 1	Spalte 2
Inhalt 1	Inhalt 2

Abbildung 2: Original (PDF Dokument)



```
Dies ist ein Titel
1 Dies ist die Überschrift 1
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut
labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo
dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet.

1.1 Dies ist die Überschrift 1.1
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut
labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo
dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet.

Tabelle 1: Eine Tabelle

Spalte 1
Inhalt 1

Abbildung 1: Ein Bild

Spalte 2
Inhalt 2
```

Abbildung 3: Nur Text (https://pdftotext.com)

Um die Gliederung in Kapitel zu ermöglichen, ist es für diese Arbeit wichtig, dass Strukturelemente wie Überschriften und Tabellen erhalten bleiben (vgl. Kapitel 2.1.4). Auf dem Markt existieren verschiedene Konvertierungstools, welche Text und teilweise auch Strukturinformationen aus PDF Dateien extrahieren können. Für diese Arbeit wurde die Software SolidCommander (SolidDocuments, 2019) gekauft, welche PDF in Word umwandelt und dabei die meisten Strukturinformationen beibehält (vgl. Abbildung 4). Dabei werden die ehemaligen Überschriften oftmals auch wieder als Überschriften klassiert (Formatvorlagen «Header 1», «Header 2», etc. in Word).

Abbildung 4: Konvertiert mit SolidCommander (Word Dokument)

SolidCommander ist ein Windows Programm, das auf einem Folder Watcher beruht. Sobald eine Datei in einem definierten Ordner (Input Ordner) abgelegt wurde, erkennt SolidCommander dies, konvertiert die Datei und legt die konvertierte Datei im Output-Ordner ab. Um den Konvertierungsprozess zu automatisieren, wurde ein Python Programm geschrieben, das in einem Intervall von einer Minute überprüft, ob sich Dokumente, die noch nicht konvertiert wurden, in der Datenbank befinden. Diese werden kopiert und im Input Ordner von SolidCommander abgelegt. Dank dem Folder Watcher erkennt Solid Commander, dass sich neue Dateien im Input Ordner befinden und konvertiert diese. Jede Minute überprüft wiederum ein Python Programm, ob sich neue Dateien im Output Ordner von SolidCommander befinden. Ist dies der Fall, werden sie als binäre Repräsentation in die Datenbank kopiert und aus dem Ordner gelöscht (vgl. Abbildung 5).

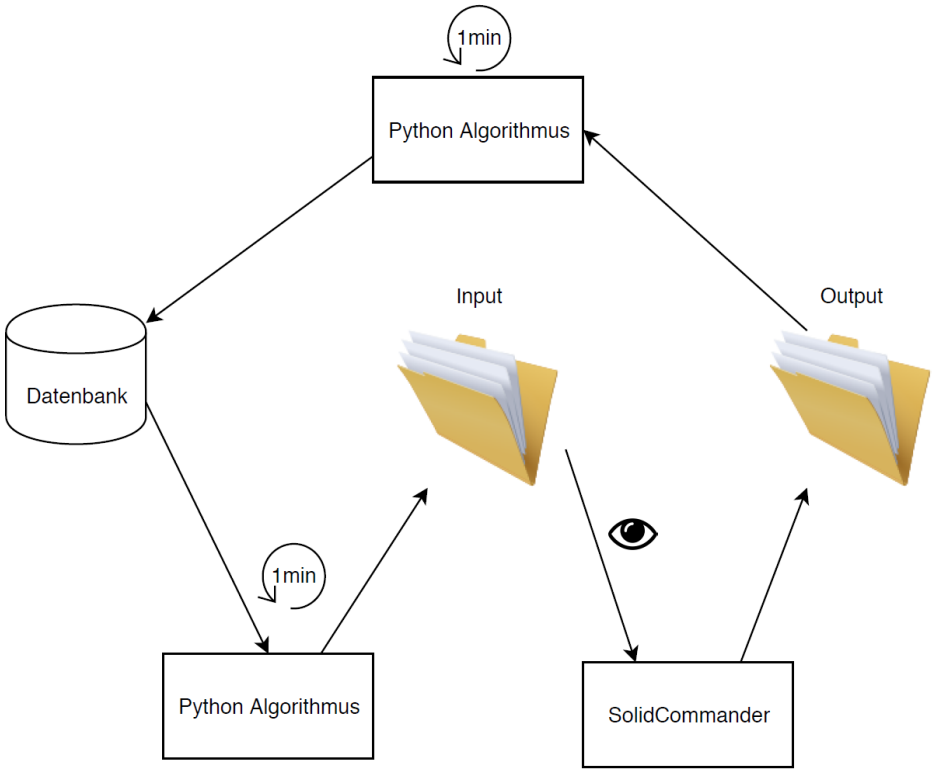


Abbildung 5: Ablauf Synchronisation

Nach der Konvertierung der 8759 Dateien (Siehe Kapitel 2.1.1) beträgt die Grösse der Dokumententabelle in der Datenbank 11.1 GB.

2.1.3 Tokenisierung, Stoppwortentfernung, und Lemmatisierung

Damit die Dokumente für das Topic Modeling und für die Klassifizierung verwendet werden können, müssen sie zuerst in eine Bag-of-Words-Repräsentation umgewandelt werden. Eine Bag-of-Words-Repräsentation eines Dokuments besteht aus einer Liste von allen Wörtern, die pro Dokument vorkommen, und deren absoluten Häufigkeiten. Nachfolgend zwei Beispielsätze:

Dies ist ein Beispiel

Ein Beispiel oder ein Exempel

Diese beiden Beispielsätze würden in folgende Bag-of-Words Repräsentation umgewandelt:

Tabelle 3: Beispiel Bag-of-Words Repräsentation

	beispiel	dies	ein	exempel	ist	oder
Satz1	1	1	1		1	
Satz2	1		2	1		1

Das Buch «Machine Learning for Text» (Aggarwal, 2018, S. 23–24) empfiehlt folgendes Vorgehen, um die Dokumente vorzuverarbeiten, damit sie in eine Bag-of-Words Repräsentationen umgewandelt werden können:

1. Entfernung aller Zeichen, die nicht Wort-Zeichen (Regex *\w*) sind
2. Einteilung der Sätze in Tokens (Tokenisierung)
3. Finden der Grundform (Lemma) für jedes Wort
4. Entfernung von Stoppworten
5. Erstellung der Bag-of-Words Repräsentationen mit Hilfe der Lemmata

Ein Tokeniser vom NLTK Package wird verwendet, um die Wörter in Tokens zu unterteilen und alle Zeichen zu entfernen, die nicht Wort-Zeichen sind. Danach wird anhand der im DataScience Blog der WZB vorgeschlagenen Methode (Konrad, 2016) ein POS-Tagging (Part-of-Speech) durchgeführt, wobei die Wörter ihren Wortarten, z.B. Adjektiv, Adverb, Nomen, etc. zugeordnet werden. Auf Basis des Ergebnisses vom POS-Tagging wird jedes Wort auf dessen Grundform (Lemma) reduziert (Konrad, 2017). Zum Beispiel ist das Lemma von «schreibt» sowie von «schrieb» das Wort «schreiben». Die Algorithmen für das POS Tagging, wie auch diejenigen für die Lemmatisierung, sind überwachte Machine Learning Algorithmen, die mit dem TIGER Korpus der Universität Stuttgart trainiert werden (vgl. Brants et al., 2004). Konrad schreibt, dass seine Methode 84% der Wörter dem richtigen Lemma zuordnen kann.

In einem Text gibt es normalerweise Wörter, die nicht viel über das Thema eines Textes aussagen. Solche Wörter sind zum Beispiel «der», «ein» oder «und». Diese Wörter werden im Text Mining Kontext Stoppworte genannt. Sie kommen üblicherweise in grosser Zahl und in nahezu jedem beliebigen Dokument vor. Werden die Wörter beim Machine Learning miteinbezogen, tragen sie üblicherweise nicht zu einer Verbesserung des Resultats bei oder verschlechtern es sogar (Aggarwal, 2018, S. 22). Deshalb werden sie aus dem Text entfernt. Mit Hilfe einer deutschen Stoppwortliste von NLTK wird der Text bereinigt.

Da das Vorverarbeiten der Dokumente eine hohe Rechenleistung erfordert, wird es mit Multiprocessing parallelisiert, damit gleichzeitig auf unterschiedlichen Prozessorkernen gerechnet werden kann. Ausserdem werden die Resultate für die spätere Weiterverarbeitung in einer Datenbank gespeichert. So wird verhindert, dass sie bei jeder Änderung der nachfolgenden Algorithmen neu berechnet werden müssen. Der ganze Vorverarbeitungsprozess dauert auf einem 4.01 GHz Prozessor mit acht Kernen und 32 GB RAM gut 12 Stunden. Dabei werden 173'843 unterschiedliche Lemmas extrahiert.

2.1.4 Einteilung der Dokumente in Kapitel (Segmentierung)

In dieser Arbeit geht es darum, relevante Textsegmente bezüglich Zuschlagskriterien zu identifizieren. Dazu müssen Texte zuerst in Textsegmente unterteilt werden. Diese Segmentierung kann beispielsweise nach strukturellen Informationen wie Zeilen, Absätzen oder Kapitel erfolgen. Ausserdem gibt es die Möglichkeit der Segmentierung nach Änderung des Unterthemas im Text (Aggarwal, 2018, S. 435). Beispiel dafür ist der TextTiling Algorithmus (Hearst, 1997). Dieser erkennt, wann sich das Thema in einem Text ändert und unterteilt diesen an den entsprechenden Stellen.
Da es in dieser Arbeit möglich ist, aus den PDF Dateien die Kapitel zu rekonstruieren (siehe Kapitel 2.1.2), wird die Segmentierung nach Kapiteln gewählt. Beim Schreiben des Textes trennt der Autor die Texte inhaltlich in Kapitel auf. Dabei kennt er den Inhalt und die Unterthemen des Dokuments normalerweise ziemlich gut. Deshalb kann davon ausgegangen werden, dass die vorhandene Unterteilung durch Kapitel besser ist, als eine automatische Unterteilung nach Unterthemen mittels TextTiling.

Dokumente sind normalerweise in Kapitel aufgeteilt, welche hierarchisch verschachtelt sein können. Deshalb ist es möglich, dass nach der Überschrift 1 direkt die Überschrift 1.1 folgt, ohne dass sich dazwischen Text befindet. Würde das Dokument bei jeder Überschrift getrennt, wäre das Kapitel 1 in diesem Fall leer. Um dies zu verhindern, werden Unterkapitel vervielfacht. Der Inhalt des Kapitels 1 aus dem vorherigen Beispiel besteht somit nicht nur aus dem Text bis zur Kapitelüberschrift 1.1, sondern beinhaltet auch den ganzen Text der Kapitel 1.1, 1.1.1, 1.1.2, 1.2, etc., bis die nächste Überschrift der gleichen Stufe folgt (hier Kapitel 2). Daneben existieren die einzelnen Unterkapitel als separate Kapitel und werden unabhängig weiter behandelt.

Da es für die einzelnen Kapitel aber relevant ist, in welchem Elternkapitel sie sich befinden, wird der Titel des Eltern- und des Grosseltern-Kapitels ebenfalls erfasst. Tabelle 4 zeigt die Attribute eines einzelnen Kapitels.

Tabelle 4: Attribute eines Kapitels

Attributname	Beschreibung
Text	Text des Kapitels (inklusive Text und Überschriften aller Unterkapitel)
Words	Liste an Lemmata, die in dem Kapitel gefunden wurden (inklusive Lemmata aller Unterkapitel)
Header	Überschrift des Kapitels
Header_Preprocessed	Lemmata der Überschrift des Kapitels
Parent_Header	Überschrift des Eltern-Kapitels
Parent_Header_Preprocessed	Lemmata der Überschrift des Eltern-Kapitels
Grandparent_Header	Überschrift des Grosseltern-Kapitels.
Grandparent_Header_Preprocessed	Lemmata der Überschrift des Grosseltern-Kapitels.

305’297 Kapitel werden in diesem Prozess extrahiert und vorverarbeitet.

2.2 Topic Modeling

Nachdem die Texte in unterschiedliche Kapitel eingeteilt wurden, sollen mit Hilfe von Text Mining Algorithmen diejenigen Kapitel gefunden werden, welche Informationen über Zuschlagskriterien bieten. Unüberwachte Algorithmen wie Topic Modeling sowie überwachte Verfahren zu Text Klassifizierung können dabei helfen. Sollen Kapitel mit einem Klassifizierungsalgorithmus in die Klassen *relevant* (für Zuschlagskriterien) bzw. *nicht relevant* eingeteilt werden, ist es nötig, zuerst Trainingsdaten zu generieren. Beim manuellen Klassifizieren (Labeln) von 1000 zufällig ausgewählten Trainingsdaten wird aber klar, wie unausgeglichen die jeweiligen Klassengrössen sind. Nur rund 3% aller Kapitel können der Klasse *relevant* zugeordnet werden. Da es aber für das Trainieren eines Klassifizierungsalgorithmus nötig ist, genügend Trainingsdaten für jede der Klassen zu haben, muss nach einem Weg gesucht werden, wie die Trainingsdaten besser aus allen Kapiteln ausgewählt werden können. Dazu werden Topic Modeling Algorithmen verwendet.

Mit Topic Modeling können Texte maschinell generierten Topics zugeordnet werden. Ein Algorithmus generiert eine Menge an Topics. Die Anzahl der Topics kann dabei mit dem Parameter *k* definiert werden. Diesen *k* Topics wird nun jeweils eine Gruppe von Wörtern zugewiesen, die in der Dokumentensammlung oftmals zusammen vorkommen. Dann berechnet der Algorithmus für jedes Dokument in der Sammlung, zu welchem Anteil es zu den jeweiligen Topics gehört. Zum Beispiel ist für ein Kapitel *X* das wichtigste Topic das Topic *A*, das zweitwichtigste das Topic *B*, usw.

Kapitel 2.2.1 gibt eine Kurzübersicht über Topic Modeling mit Nonegative Matrix Factorization. Im Kapitel 2.2.2 wird der LDA Algorithmus vorgestellt. Kapitel 2.2.3 erklärt, ob und aus welchen Gründen der LDA oder der NMF Algorithmus für die Arbeit bevorzugt wird und wie der Wert für den Parameter *k* ausgewählt wird.

2.2.1 Nonnegative Matrix Factorization (NMF)

Nonnegative Matrix Factorization (NMF) ist ein mathematisches Verfahren, bei dem eine Matrix *X* in zwei Matrizen *W* und *H* zerteilt wird. Die Multiplikation dieser beiden Matrizen ergibt annähernd wieder die Matrix *X* (vgl. Abbildung 6). Die Annahme

dabei ist, dass es keine negativen Zahlen in den Matrizen gibt. Um eine Näherung zur optimalen Lösung zu finden, wird oftmals eine numerische Approximation eingesetzt.

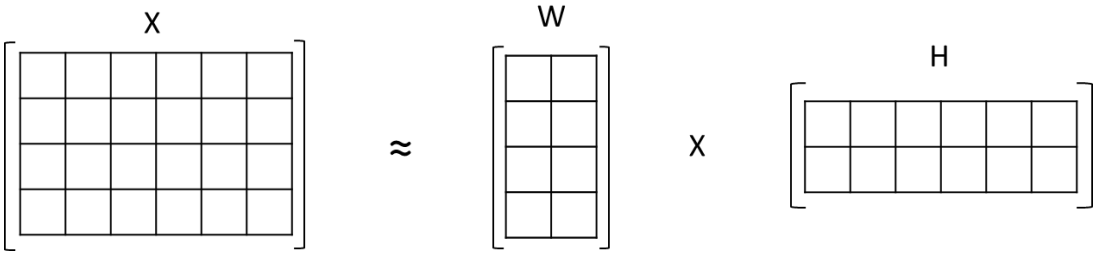


Abbildung 6: Nonnegative Matrix Factorization (NMF)

Die initiale Matrix *X* besteht aus *m* Zeilen und *n* Spalten. Die resultierende Matrix *W* besteht aus *m* Zeilen und *r* Spalten und die Matrix *H* aus *r* Zeilen und *n* Spalten. Der Parameter *r* kann dabei vorgegeben werden.
Auf das Problem vom Topic Modeling bezogen bedeutet das, dass die Matrix *X* eine Document-Term Matrix ist. Sie hat *m* Zeilen, wobei *m* die Anzahl Dokumente im Textkorpus sind. Ausserdem hat sie *n* Spalten, wobei die Spalten die unterschiedlichen Wörter sind, die in den Texten vorkommen. Mittels NMF wird sie in eine Matrix *W* umgewandelt, welche wiederum *m* Zeilen und *r* Spalten hat. *r* steht für die vorgegebene Anzahl Topics. *W* definiert also die Wichtigkeit der Topics für das jeweilige Dokument. Die zweite Matrix, die entsteht, definiert die Wichtigkeit der einzelnen Wörter im Korpus für das jeweilige Topic. In den Spalten finden sich wiederum die *n* unterschiedlichen Wörter des Korpus und in den Zeilen die *r* Topics.

Die Einträge in der Matrix *X* können die Häufigkeiten der Wörter sein. Ist dies der Fall, entspricht die Matrix einer Bag-of-Words Repräsentation. Eine andere Möglichkeit, welche auch in dieser Arbeit verwendet wurde, ist die Berechnung der TF-IDF Werte für die Wörter. TF-IDF bedeutet Term-Frequency * Inverse Document Frequency und wird wie folgt berechnet (Salton & Buckley, 1988):

$$w_d = f_{w,d} * \log(|D|/f_{w,D})$$

Dabei ist D der Dokumentkorpus, d ein individuelles Dokument dieses Korpus d ϵ D und w ist ein Wort. $f_{w,d}$ ist die Anzahl, bei der das Wort w im Dokument *d* vorkommt und $|D|$ ist die Anzahl der Dokumente im Korpus. $f_{w, D}$ ist die Anzahl, bei der das Wort *w* im ganzen Korpus vorkommt.

Arora et al. haben 2013 Algorithmen für NMF gezeigt, welche in polynomialer Zeit terminieren (Arora et al., 2013).

2.2.2 LDA Algorithmus

Der Latent Dirichlet Allocation (LDA) Algorithmus wurde 2003 von David Blei, Andrew Ng und Michael Jordan publiziert (Blei, Ng, & Jordan, 2003). LDA betrachtet Text als Bag-of-Words Repräsentation, was bedeutet, dass die Reihenfolge der Wörter im Text keine Rolle spielt. Die grundlegende Annahme des LDA Algorithmus ist, dass ein Text durch das Auswählen einer Menge an Themen (Topics) und anschliessend einer Menge an Wörtern für jedes Thema erstellt wurde. Unter dieser Annahme kann der LDA Algorithmus ein Reverse Engineering dieses Prozesses durchführen. Grob besteht der LDA Algorithmus aus folgenden Schritten (in Anlehnung an Doll, 2018):

1. Annahme, dass *k* Topics in allen Dokumenten vertreten sind.
2. Verteilen der *k* Topics über das Dokument *m*, indem jedem Wort ein Topic zugewiesen wird.
3. Dem Topic *w* ein neues Topic zuweisen. Für jedes Wort *w* im Dokument *m* wird die Annahme getroffen, dass dieses Wort dem falschen, aber alle anderen Wörter dem richtigen Topic zugewiesen wurden. Die Zuweisung wird als Wahrscheinlichkeitsfunktion berechnet und basiert auf zwei Kriterien:
    - Welche Topics gibt es im Dokument *m*?
    - Wie viele Male wurde das Wort *w* in allen anderen Dokumenten bereits zum jeweiligen Topic zugewiesen?
4. Die Schritte werden mehrere Male für die verschiedenen Dokumente wiederholt.

Als Resultat des LDA Algorithmus entstehen wiederum die beiden Matrizen *W* und *H* (vgl. Kapitel 2.2.1).

2.2.3 Auswahl des Topic Modeling Algorithmus und der Anzahl Topics

Für das Topic Modelling werden unterschiedliche Modelle berechnet und evaluiert. Sowohl mit dem LDA wie auch mit dem NMF Algorithmus werden Modelle mit 5, 10, …, 100 Topics berechnet (Parameter *k)*. Für jedes dieser Modelle werden danach die 10 wichtigsten Wörter pro Topic überprüft. Dabei wird entschieden, ob das Topic vor allem Kapitel zu Zuschlagskriterien beschreibt oder nicht. Zur Vereinfachung werden die Topics, welche Kapitel zu Zuschlagskriterien enthalten, im Weiteren *Zuschlag-Topics* und die anderen *Nicht-Zuschlag-Topics* genannt. Ein Topic, bei dem die wichtigsten Wörter «zuschlagskriterium», «zk», «angebot», «kriterium», usw. sind, wird demnach *Zuschlag-Topic* genannt, während ein Topic mit den Wörtern «bauherrschaft», «ab», «ausgangslage», usw. *Nicht-Zuschlag-Topic* genannt wird.

Die Güte von überwachten Algorithmen kann normalerweise mit den Testdaten und verschiedenen Gütemassen bestimmt werden. Bei unüberwachten Algorithmen ist es hingegen relativ schwierig herauszufinden, wie gut die Modelle sind. Qualitative Kriterien müssen definiert werden, anhand derer die Modelle evaluiert werden. Für die Entscheidung zwischen dem LDA und dem NMF Algorithmus werden deshalb zwei Kriterien bestimmt. Zum einen wird die Konstanz der 10 wichtigsten Wörter bei Änderung des Parameters *k* für die *Zuschlag-Topics* evaluiert. Abbildung 7 zeigt diese Auswertung für den LDA Algorithmus. Die erste Spalte definiert, um welches Modell es sich handelt (z.B. Modell mit 30 Topics). Aufgelistet werden die *Zuschlag-Topics*, von welchen es pro Modell mehrere geben kann. Deshalb kann es sein, dass zwei Zeilen für dasselbe Modell verwendet werden. Zum Beispiel gehören die beiden letzten Zeilen zum LDA Modell mit 100 Topics. Abbildung 8 zeigt dieselbe Auswertung für das NMF Modell.

topic_size	word1	word2	word3	word4	word5	word6	word7	word8	word9	word10
5	zuschlagskriterium	angebot	anforderung	projekt	eignungskriterium	bewertung	technisch	leistung	preis	referenzen
10	ausschreibung	verfahren	grundlage	leistung	nein	ja	submission	ausgangslage	projekt	phase
15	zuschlagskriterium	eignungskriterium	angabe	bewertung	nachweis	referenzen	schlüsselperson	formular	anhang	angebot
20	zuschlagskriterium	bewertung	referenzen	schlüsselperson	angabe	zk	kriterium	referenzobjekt	preis	funktion
25	zuschlagskriterium	bewertung	ausschreibungsunterlage	referenzen	schlüsselperson	angebot	preis	zk	punkt	formular
30	zuschlagskriterium	bewertung	referenzen	schlüsselperson	angabe	zk	angebot	kriterium	referenzobjekt	preis
35	zuschlagskriterium	bewertung	nachweis	beilage	referenzen	schlüsselperson	angebot	zk	angabe	kriterium
40	zuschlagskriterium	eignungskriterium	nachweis	bewertung	angebot	zk	kriterium	preis	erfüllen	qualität
45	zuschlagskriterium	bewertung	angebot	zk	punkt	preis	qualität	kriterium	gut	evaluation
50	zuschlagskriterium	eignungskriterium	bewertung	referenzen	schlüsselperson	angebot	zk	kriterium	referenzobjekt	erfüllen
55	zuschlagskriterium	bewertung	angebot	zk	kriterium	preis	qualität	erfüllung	gut	evaluation
60	zuschlagskriterium	bewertung	angebot	referenzen	zk	kriterium	preis	qualität	erfüllung	gut
65	zuschlagskriterium	bewertung	zk	kriterium	punkt	angebot	qualität	preis	gut	erfüllung
65	zuschlagskriterium	ziffer	gemäss	eignungsund	projektorganisation	entsprechen	grundsatz	liefern	beurteilung	qualitätssicherung
70	200	chf	eignungsund	32	kbob	51	umsetzung	mio	210	zuschlagskriterium
70	zuschlagskriterium	bewertung	referenzen	schlüsselperson	zk	kriterium	angebot	qualität	preis	referenz
75	zuschlagskriterium	bewertung	zk	kriterium	preis	angebot	qualität	erfüllung	gut	erhalten
80	zuschlagskriterium	bewertung	zk	preis	angebot	qualität	gut	erhalten	gewichtung	evaluation
85	zuschlagskriterium	bewertung	schlüsselperson	zk	kriterium	qualität	referenz	gut	erhalten	evaluation
90	zuschlagskriterium	bewertung	schlüsselperson	zk	qualität	angebot	referenz	gut	kriterium	erhalten
90	10	zuschlagskriterium	vergabe	max	eignungsund	grundsatz	nachstehend	min	vorgegeben	zk
95	zuschlagskriterium	preis	bewertung	zk	qualität	gut	angebot	erhalten	gewichtung	evaluation
100	zuschlagskriterium	bewertung	referenzen	zk	qualität	kriterium	gut	preis	erhalten	gewichtung
100	zuschlagskriterium	eignungsund	beurteilung	direkt	vereinbaren	zusammenstellung	zahlungsfrist	terminplan	sitz	eingang

Abbildung 7: Wichtigste Wörter der Zuschlag-Topics pro Modell (LDA Algorithmus)

topic_size	word1	word2	word3	word4	word5	word6	word7	word8	word9	word10
5	zuschlagskriterium	bewertung	preis	eignungskriterium	zk	referenzen	gewichtung	nachweis	kriterium	schlüsselperson
10	zuschlagskriterium	bewertung	preis	zk	gewichtung	kriterium	punkt	eignungsund	qualität	referenzen
15	zuschlagskriterium	bewertung	preis	zk	gewichtung	punkt	kriterium	eignungsund	referenzen	qualität
20	zuschlagskriterium	bewertung	preis	zk	gewichtung	kriterium	punkt	eignungsund	referenzen	qualität
25	zuschlagskriterium	bewertung	preis	zk	gewichtung	kriterium	punkt	referenzen	eignungsund	qualität
30	zuschlagskriterium	bewertung	preis	zk	gewichtung	kriterium	punkt	eignungsund	qualität	erhalten
35	zuschlagskriterium	zk	bewertung	gewichtung	eignungsund	kriterium	qualität	punkt	referenzen	evaluation
40	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	referenzen
45	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	max
50	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	max
55	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	max
60	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	max	evaluation	punkt	referenzen	bewerten
65	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	max	evaluation	punkt	referenzen	bewerten
70	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	max	evaluation	punkt	referenzen	bewerten
75	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	max
80	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	max	evaluation	punkt	10	bewerten
85	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	evaluation	max	punkt	referenzen	bewerten
90	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	evaluation	max	punkt	bewerten	angebotspreis
95	zuschlagskriterium	zk	gewichtung	eignungsund	qualität	evaluation	max	punkt	bewerten	referenzen
100	zuschlagskriterium	zk	gewichtung	eignungsund	bewertung	qualität	kriterium	punkt	evaluation	max

Abbildung 8: Wichtigste Wörter der Zuschlag-Topics pro Modell (NMF Algorithmus)

Es ist klar ersichtlich, dass das NMF Modell die konstanteren Ergebnisse liefert. Im Vergleich zum LDA Algorithmus ändern sich die wichtigsten Wörter der *Zuschlag-Topics* bei Änderung von *k* kaum. Deshalb wird davon ausgegangen, dass sich der NMF Algorithmus bei der Zuteilung sicherer ist und die Ergebnisse weniger auf Zufall beruhen.

Das zweite Kriterium, das für den Vergleich zwischen NMF und LDA Modell definiert wird, ist die Varianz der Anzahl Kapitel in den *Zuschlag-Topics* bei Änderung der Variable *k*. Jedes der Modelle berechnet für jedes der rund 300'000 Kapitel (vgl. Kapitel 2.1.4) das relevanteste Topic. Danach wird für jedes der Modelle die Anzahl Kapitel summiert, die als relevantestes Topic ein *Zuschlags-Topic* haben. Werden die Kurven der Anzahl Kapitel bei Änderung von *k* miteinander verglichen, zeigt sich, dass auch hier das NMF Modell (Abbildung 10) konstantere Ergebnisse liefert als das LDA Modell (Abbildung 9).

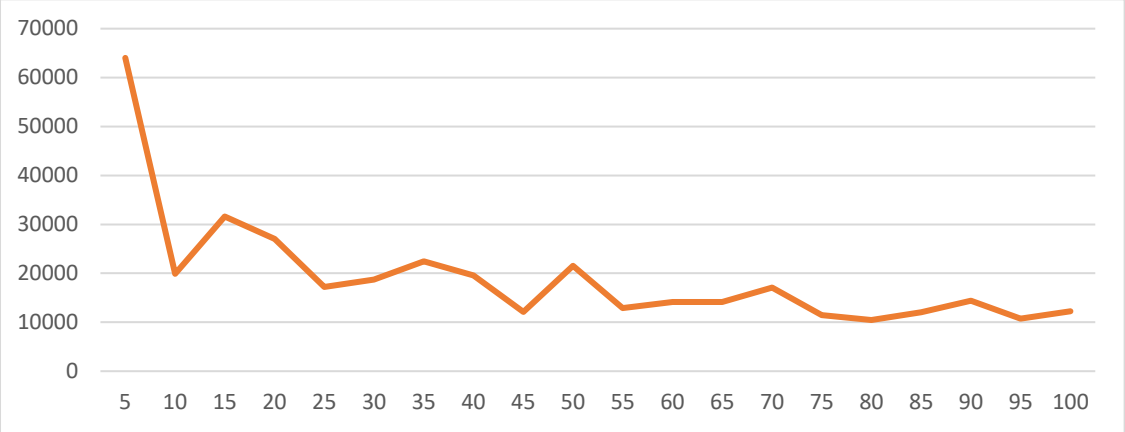


Abbildung 9: Anzahl Kapitel der Zuschlag-Topics (LDA Modell)

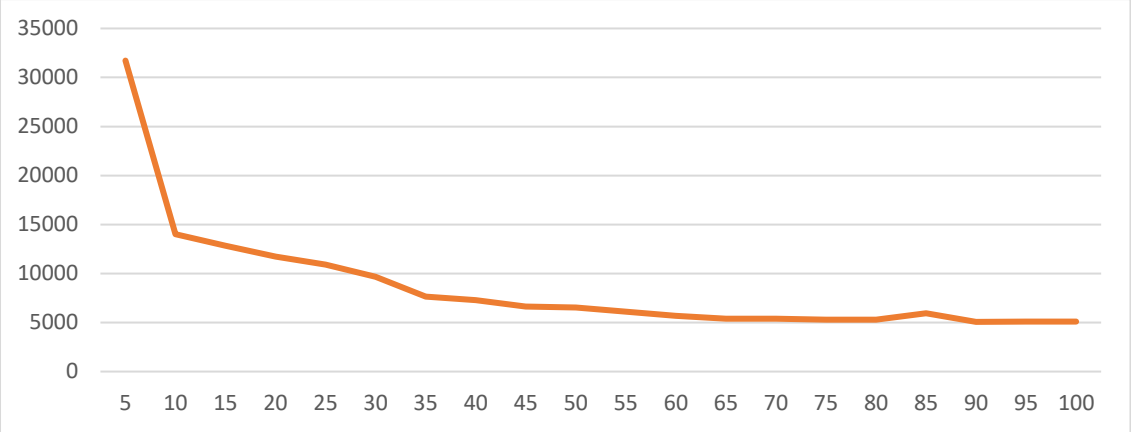


Abbildung 10: Anzahl Kapitel der Zuschlag-Topics (NMF Modell)

Aufgrund dieser beiden Kriterien wird das NMF Modell für die weitere Verwendung ausgewählt. Die Auswahl des optimalen Wertes für *k* wird anhand drei qualitativer Argumente getroffen:

1. Abbildung 8 zeigt, dass die wichtigsten 10 Wörter für die *Zuschlag-Topics* dann sinnvoller sind, wenn *k* grösser als 35 ist. Ab 35 Topics wird «zk» zum wichtigsten Wort. Dies erscheint vor allem darum nachvollziehbar, weil sich in den relevanten Kapiteln oftmals eine Aufzählung von Zuschlagskriterien befindet (ZK1, ZK2, …).

Ab 40 Kapitel wird das Wort «bewertung» weniger wichtig eingestuft. Dies erscheint ebenso sinnvoll, da die Kapitel zu der Bewertung der Zuschlagskriterien und die Kapitel, in denen die Zuschlagskriterien selber aufgelistet werden, oftmals nicht dieselben sind.

2. Abbildung 10 zeigt, dass die Anzahl Topics bis zu einem Wert von 35 für *k* rapide sinken, und danach nahezu konstant bleiben. Daraus wird geschlossen, dass die Topics zuvor etwas zu breit definiert wurden.
3. Da es sich beim Topic-Modelling lediglich um eine Vorselektion handelt, ist es wichtig, dass möglichst alle Kapitel über Zuschlagskriterien in das *Zuschlag-Topic* eingeteilt werden. Werden hingegen zu viele «falsche» Kandidaten in das *Zuschlag-Topic* eingeteilt, ist das weniger gravierend, da sie in den weiteren Schritten zusätzlich gefiltert werden. Deshalb wird ein Wert für *k* ausgewählt, der nicht zu hoch ist, da bei höherem *k* die Anzahl der Kapitel in den *Zuschlag-Topics* sinkt.

Aufgrund der oben genannten Argumente wird ein Wert von 40 für *k* ausgewählt. Die wichtigsten 10 Wörter für jedes Topic dieses Modells können im Anhang 2 eingesehen werden. Insgesamt werden 7281 Kapitel gefunden, bei denen als wichtigstes Topic ein *Zuschlag-Topic* berechnet wurde. Dies entspricht einem prozentualen Anteil von rund 2.2% aller Kapitel.

2.3 Klassifizierung

Um die Kapitel der Ausschreibungsdokumente bezüglich Zuschlagskriterien in relevante Kapitel, beziehungsweise nicht relevante Kapitel, einzuteilen, soll überwachtes Lernen angewendet werden. Zur Vereinfachung werden die Kapitel im Folgenden «relevante Kapitel» und «nicht relevante Kapitel» genannt. Bei einem binären Klassifizierungsproblem wird zwischen dem positiven und dem negativen Label unterschieden. In dieser Arbeit entspricht das positive Label der Klasse «relevante Kapitel» und das negative Label der Klasse «nicht relevante Kapitel».

Als Klassifizierungsalgorithmus wird ein RandomForestClassifier verwendet. Die Funktionsweise dieses Algorithmus wird im Kapitel 2.3.1 erklärt. Die Auswahl der

Trainingsdaten erfolgt mit Hilfe der zuvor berechneten Topic Models. Darauf wird im Kapitel 2.3.2 eingegangen. Nachdem die Trainingsdaten ausgewählt und manuell gelabelt wurden, wird ein erstes RandomForest Modell berechnet (Kapitel 2.3.3). Dieses Modell wird iterativ mit Hilfe von Active Learning verbessert (Kapitel 2.3.4). Kapitel 2.3.5 beschreibt ein simpleres Vorgehen, das anhand von zufällig ausgewählten Trainingsdaten ein RandomForst Modell trainiert. Dieses kann später für den Vergleich mit dem entwickelten kombinierten Verfahren verwendet werden. Kapitel 2.3.6 geht genauer darauf ein, wie mit dem zuvor berechneten Topic Model eine Klassifizierung durchgeführt werden kann. Zuletzt wird ein simpler Patternmatching Algorithmus beschrieben, welcher ebenfalls zum Vergleich mit den anderen Verfahren dient (Kapitel 2.3.7).

2.3.1 RandomForest Klassifizierung

RandomForest ist ein Algorithmus, der im Jahre 2001 von Breiman publiziert wurde (Breiman, 2001). Dabei wird eine zuvor definierte Anzahl an Entscheidungsbäumen generiert. Jeder dieser Entscheidungsbäume schlägt unabhängig von den anderen eine Klassifizierung für alle Einträge im Datensatz vor. Danach werden die Vorschläge konsolidiert und pro Eintrag diejenige Klasse mit den meisten Stimmen gewählt. Wird zum Beispiel ein RandomForest mit 1000 Bäumen berechnet und 700 dieser 1000 Bäume weisen einem Eintrag im Datensatz das positive Label zu, wird der Eintrag schliesslich der positiven Klasse zugewiesen.

Damit nicht immer derselbe Entscheidungsbaum erstellt wird, hat der RandomForest Algorithmus eine eigene Art und Weise, wie Entscheidungsbäume konstruiert werden. Während in Standardentscheidungsbäumen bei jedem Knoten (Verzweigung) die bestmögliche Entscheidungsvariable gesucht wird, um den Datensatz in zwei disjunkte Hälften zu unterteilen, wird beim RandomForest Algorithmus für jeden Knoten eine Menge an Entscheidungsvariablen zufällig selektiert, aus welcher dann die bestmögliche Variable gewählt wird (Liaw & Wiener, 2002).

Liaw & Wiener (Liaw & Wiener, 2002) zählen folgende Stärken des RandomForest Algorithmus auf:

- RandomForest Algorithmen performen sehr gut im Vergleich zu anderen Klassifizierungsalgorithmen wie Discriminant Analysis, Support Vector Machines und Neuronal Networks
- RandomForest Algorithmen sind benutzerfreundlich, da sie nur zwei Inputvariablen benötigen: Die Anzahl Bäume im RandomForest und die Anzahl Variablen, die an jedem Knoten zufällig ausgewählt werden

Weitere Stärken von RandomForest Modellen sind:

- Durch die zufällige Generierung von Entscheidungsbäumen wird die Gefahr für Overfitting reduziert (Breiman, 2001)
- RandomForest Modelle können für das Trainieren parallelisiert werden, indem die zu erstellenden Entscheidungsbäume auf verschiedene Rechner aufgeteilt werden
- Der Rechenaufwand steigt mit zunehmender Anzahl Bäume linear

Ein Nachteil des RandomForest Algorithmus ist, dass er als Blackbox agiert. Da die Einteilung der Einträge aus einer Kombination vieler unterschiedlicher Entscheidungsbäume geschieht, kann schlecht interpretiert werden, aufgrund welcher Attribute der Eintrag schlussendlich der positiven, respektive der negativen Klasse zugewiesen wurde.

Aufgrund der genannten Stärken und weil RandomForest Modelle in TextMining Anwendungen häufig eingesetzt werden (Aggarwal, 2018), wird für diese Arbeit der RandomForest Algorithmus verwendet.

2.3.2 Generierung der Trainingsdaten

Um das RandomForest Modell zu trainieren, müssen zuerst Trainingsdaten generiert werden. Das Problem dabei ist, dass es deutlich mehr nicht relevante als relevante Kapitel gibt (vgl. Kapitel 2.2). Deshalb werden die Trainingsdaten nach Topics ausgewählt (geschichtete Zufallsstichprobe). Jedes der rund 300'000 Kapitel wurde zuvor einem der 40 unterschiedlichen Topics zugeordnet. Aus dem *Zuschlag-Topic* werden

zufällig 100 Kapitel ausgewählt, während von den restlichen 39 Topics jeweils 10 selektiert werden. Insgesamt entsteht so ein Trainingsdatensatz mit 490 Daten, welcher manuell gelabelt werden muss. Durch das Oversampling des ersten Topics kann die Anzahl der positiven Klasse in den Trainingsdaten auf 68 erhöht werden, was rund 14% des Datensatzes entspricht.
Ausserdem muss ein Testdatensatz generiert werden. Da dieser möglichst optimal die Realität wiederspiegeln soll, werden 1000 Kategorien aus allen Kategorien zufällig ausgewählt, ohne die Resultate des Topics Modellings zu berücksichtigen. Der Testdatensatz besteht schlussendlich aus 969 (97%) Daten der negativen und 31 (3%) Daten der positiven Klasse.

2.3.3 Erstes RandomForest Modell

Auf Basis der 490 Trainingsdaten wird ein erstes RandomForest Modell trainiert. Dazu wird die RandomForestClassifier Implementation von Scikit-Learn verwendet. Für den RandomForestClassifier sind verschiedene Einstellungen mittels Parameter möglich. Während für viele davon die Standardwerte übernommen werden, werden einige auch manuell überschrieben:

- *n_estimators*: Definiert die Anzahl Entscheidungsbäume, die im RandomForest generiert werden. Eine höhere Anzahl von Bäumen bedeutet normalerweise besserer Performance, wobei der Effekt mit zunehmender Anzahl abnimmt (vgl. Mayumi Oshiro, Santoro Perez, & Baranauskas, 2012). Hingegen steigt der Ressourcenbedarf für die Berechnung des RandomForest pro zusätzlichen Baum linear. Da die Rechenzeit bei dieser Arbeit aber sowieso gering ausfällt (unter einer Minute für ein Modell) wird ein hoher Wert von 1000 Bäumen definiert.
- *random_state:* Um die Reproduzierbarkeit der Resultate zu ermöglichen, kann mit diesem Paramenter ein Seed festgelegt werden, so dass bei jeder Neuberechnung dieselben Zufallszahlen gezogen werden. Hier wird der Wert 0 übergeben.
- *class_weight:* Da der Datensatz stark unausgeglichen ist (mehr negative als positive Labels), kann mit diesem Parameter ein Gewicht pro Klasse definiert werden. «balanced» wird als Wert übergeben, was bedeutet, dass die

Klassengewichte automatisch bestimmt werden. Das Gewicht ist dabei die invers proportionale Häufigkeit der Klasse (vgl. scikit-learn.org, 2019b):

*n_samples / (n_classes * np.bincount(y))*

- *max_depth:* Limitiert die maximale Tiefe der Entscheidungsbäume. Mit einem kleinen Wert für *max_depth* kann das Risiko von Overfitting minimiert werden. Dafür wird die Performance tendenziell schlechter (Fraj, 2017). Verschiedene Werte für *max_depth* sollen in dieser Arbeit überprüft werden.

Damit das Risiko für Overfitting minimiert wird, sollen verschiedene Werte für *max_depth* getestet werden. Modelle mit unterschiedlichem *max_depth* werden mittels Cross Validation miteinander verglichen. Durch Cross Validation kann die Unabhängigkeit und Generalisierbarkeit der Modelle garantiert werden. Dabei werden die Trainingsdaten in *k* Teilmengen (Splits) unterteilt. In *k* Durchläufen (Folds) werden jeweils *k-1* Teilmengen verwendet, um das RandomForest Modell zu trainieren und die übrige Teilmenge dafür, das Modell zu validieren. Die Performance der Modelle wird danach über alle Durchläufe gemittelt und das finale Modell kann mit unabhängigen Testdaten evaluiert werden. Abbildung 11 zeigt einen schematischen Überblick über Cross Validation.

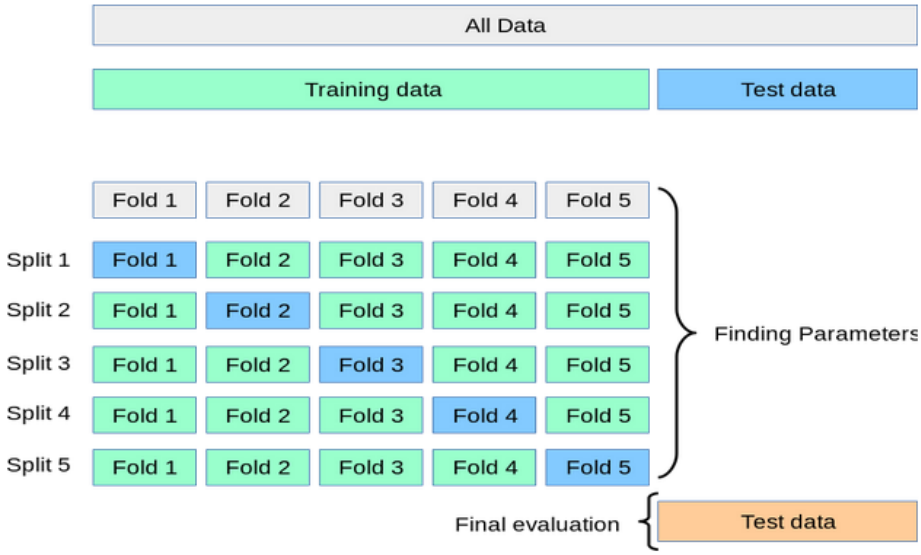


Abbildung 11: Cross Validation (scikit-learn.org, 2019a)

Modelle mit 1, 2, 5, 10, 15 und unlimitierter maximaler Tiefe werden mit Cross Validation trainiert und evaluiert. Als Performance Mass dient dabei der F_1 Score, welcher als harmonisches Mittel zwischen Precision und Recall definiert ist:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP+FP} \qquad Recall = \frac{TP}{TP+FN}$$

Für jede Klasse im Modell kann der F1 Score berechnet werden. Gibt es mehrere F_1 Scores im Modell, wird der Mittelwert dieser Scores als Macro F_1 Score bezeichnet. Der Vorteil vom F_1 Performance Mass ist, dass es im Gegensatz zu anderen Massen, wie z.B. der Accuracy (relative Anzahl richtig klassifizierter Elemente), auch für unausgeglichene Datensätze funktioniert.

Tabelle 5 zeigt die F_1 Scores für die unterschiedlichen Modelle sowie den Mittelwert davon. Die Modelle mit *max_depth* von 10, 5 und 2 haben die höchsten mittleren F_1 Scores. Durch Active Learning (vgl. Kapitel 2.3.4) nimmt die Komplexität der RandomForest Bäume im weiteren Verlauf der Arbeit tendenziell zu. Falls der Wert für *max_depth* zu klein gewählt wird, kann die zunehmende Komplexität unter Umständen nicht gut abgebildet werden, was zu sinkender Performance führt. Um dieses Risiko zu minimieren wird *max_depth* auf den Wert 10 festgelegt.

Tabelle 5: F1 Scores für unterschiedliche Werte von max_depth

Modell	None	15	10	5	2	1
Fold 1	0.78	0.81	0.81	0.81	0.81	0.69
Fold 2	0.72	0.68	0.72	0.74	0.74	0.74
Fold 3	0.72	0.80	0.80	0.80	0.80	0.80
Fold 4	0.69	0.69	0.74	0.72	0.72	0.74
Fold 5	0.66	0.74	0.78	0.78	0.78	0.78
Mittelwert	**0.71**	**0.74**	**0.77**	**0.77**	**0.77**	**0.75**

Ein neues RandomForest Modell wird mit *max_depth = 10* berechnet. Dazu werden die 490 Trainingsdaten wiederum in 80% Trainingsdaten und 20% Validierungsdaten unterteilt. Anschliessend wird das Modell mit den 1000 manuell gelabelten Testdaten getestet. Dabei erreicht es einen Macro F_1 Score von 0.70. Insgesamt werden 8 Kapitel der Klasse *relevant* und 969 Kapitel der Klasse *nicht relevant* richtig klassifiziert. 22 Kapitel der Klasse *relevant* und 1 Kapitel der Klasse *nicht relevant* werden falsch klassifiziert (vgl. Tabelle 6).

Tabelle 6: Confusion Matrix erstes RandomForest Modell

	Nicht relevant (klassifiziert)	*Relevant (klassifiziert)*
Nicht relevant (effektiv)	969 (TN)	1 (FP)
Relevant (effektiv)	22 (FN)	8 (TP)

2.3.4 Active Learning

Um ein gutes Modell mit überwachtem Lernen zu trainieren werden normalerweise viele Trainingsdaten benötigt. Diese müssen oftmals manuell unter hohem Zeitaufwand gelabelt werden. Eine Strategie zur Reduzierung der benötigten Trainingsdaten und somit Minimierung des manuellen Aufwandes ist Active Learning.

Die Hauptidee von Active Learning ist es, dass die Anzahl Trainingsdaten reduziert werden kann, wenn dem Algorithmus erlaubt wird, selbstständig die nächsten Trainingsdaten zu bestimmen, welche vom Menschen klassifiziert werden sollen (vgl. Settles, 2009). Settles beschreibt unterschiedliche Ansätze von Active Learning, darunter Query-by-Committee, Expected Model Change oder Uncertainty Sampling. Letzteres wird in dieser Arbeit verwendet, da der Ansatz intuitiv und einfach umzusetzen ist. Bei Uncertainity Sampling klassifiziert das zuvor trainierte Modell einige oder alle bisher ungelabelten Daten und bewertet, wie sicher es sich bei der Klassifikation des jeweiligen Eintrages ist. Beim RandomForest Modell wird diese Sicherheit durch den Vergleich der Bäume, die einem Eintrag die positiven Klasse zuweisen zu der Anzahl Bäume, die demselben Eintrag die negative Klasse zuweisen, berechnet.

Als Beispiel kann ein RandomForest mit 1000 Bäumen betrachtet werden. Wenn 600 der 1000 Bäume dem zu klassifizierenden Objekt das positive Label zuweisen und 400 das negative, ist sich der Algorithmus zu 60% sicher, dass es zur positiven Klasse gehört. Die Sicherheiten, die der RandomForest Algorithmus so berechnet, werden anschliessend aufsteigend sortiert und die *n* Elemente mit der kleinsten Sicherheit als neue Trainingsdaten ausgewählt.

Nachdem ein erstes RandomForest Modell mit 490 Trainingsdaten trainiert wurde (vgl. Kapitel 2.3.3), soll dieses mit 10 Runden Active Learning verbessert werden. Abbildung 12 stellt den Active Learning Prozess schematisch dar.

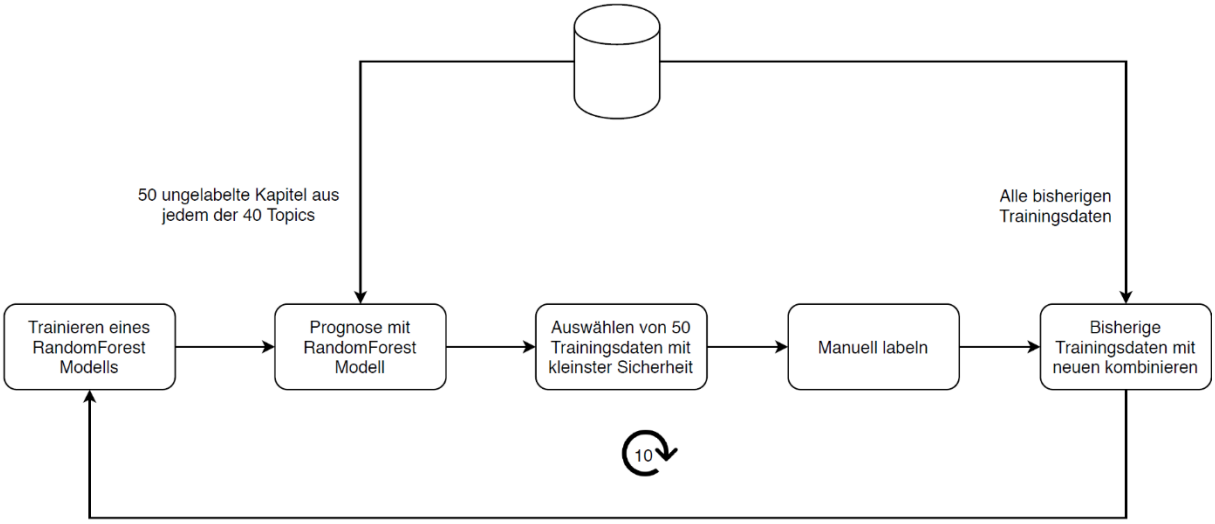


Abbildung 12: Active Learning Prozess

Zu Beginn jeder Active Learning Runde wird auf Basis der bisherigen Trainingsdaten ein RandomForest Modell trainiert. Danach wählt der Algorithmus 50 bisher ungelabelte Kapitel zufällig aus jedem der 40 Topics des Topic Models aus. Diese 2000 Kapitel werden mit dem trainierten Modell klassifiziert. Dabei bestimmte es die Sicherheiten, sortiert sie aufsteigend und bestimmt die 50 Kapitel mit der kleinsten Sicherheit als zukünftige Trainingsdaten. Nachdem die neuen Kapitel manuell gelabelt wurden, werden die bisherigen mit den neuen Trainingsdaten kombiniert. Danach beginnt eine neue Active Learning Runde.

Am Ende jeder Runde wird das aktuelle RandomForest Modell mit den 1000 Testdaten getestet. Die Confusion Matrix und die F_1 Scores werden berechnet. Nach der zehnten Runde sieht die Confusion Matrix folgendermassen aus:

Tabelle 7: Confusion Matrix nach Runde 10

	Nicht relevant (klassifiziert)	*Relevant (klassifiziert)*
Nicht relevant (effektiv)	960 (TN)	9 (FP)
Relevant (effektiv)	16 (FN)	15 (TP)

Abbildung 13 zeigt den Verlauf der F_1 Scores über die 10 Runden. Nach Runde 10 hat der F_1 Score für das positive Label einen Wert von 0.55 und für das negative Label einen Wert von 0.99. Detaillierte Auswertungen zu den einzelnen Runden werden in Anhang 2 aufgelistet.

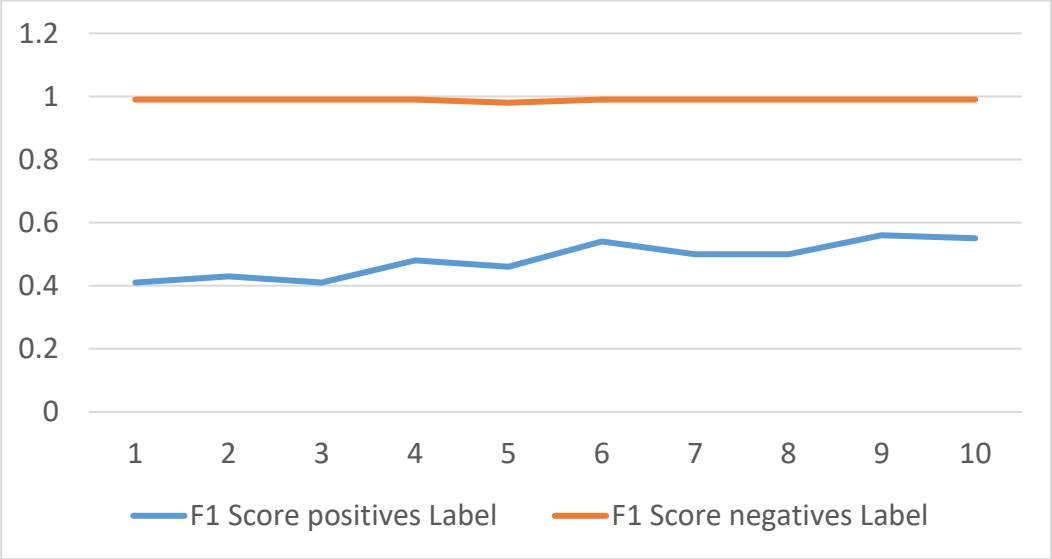


Abbildung 13: F1 Scores Active Learning

490 Trainingsdaten der ersten Runde plus 10 * 50 Trainingsdaten aller folgenden Active Learning Runden ergeben insgesamt einen Trainingsdatensatz mit 940 Einträgen. Von diesen 940 können nur 934 manuell einer Kategorie zugeordnet werden. Die restlichen sechs Kapitel sind auch für den Menschen so schwierig einzuteilen, dass sie

übersprungen werden müssen. Im finalen Trainingsdatensatz sind 263 Kapitel der Klasse *relevant* und 671 Kapitel der Klasse *nicht relevant* vertreten.

Nach der Durchführung der zehn Runden Active Learning wird das trainierte Modell auf alle Kapitel ohne Label (rund 300'000) angewendet. 5362 Kapitel werden als *relevant* klassifiziert.

2.3.5 Modell ohne Topic Modeling und Active Learning

Damit das entwickelte Verfahren zur Auswahl der Trainingsdaten mit Topic Modeling und Active Learning evaluiert werden kann, wird parallel ein RandomForest Modell mit zufällig ausgewählten Trainingsdaten trainiert. Auch für dieses Modell werden insgesamt 934 Daten manuell gelabelt (gleich viele wie beim Active Learning, vgl. Kapitel 2.3.4). Diese Trainingsdaten werden zufällig aus den rund 300'000 Kapitel ausgewählt, was dazu führt, dass nur 32 Kapitel (3.4%) der Klasse *relevant* zugeordnet werden können. Auf Basis dieses Trainingsdatensatzes wird ein RandomForest Modell mit den gleichen Parametern wie im ersten Modell trainiert (*class_weight='balanced', max_depth=10, n_estimators=1000*; vgl. Kapitel 2.3.3). Danach wird das Modell mit den 1000 Testdaten getestet. Tabelle 8 zeigt, wie das Modell die Testdaten klassifiziert.

Tabelle 8: Confusion Matrix (RandomForest Modell ohne Active Learning und Topic Modeling)

	Nicht relevant (klassifiziert)	*Relevant (klassifiziert)*
Nicht relevant (effektiv)	969 (TN)	1 (FP)
Relevant (effektiv)	22 (FN)	8 (TP)

Der F_1 Score für die positive Klasse ist 0.06 und für die negative Klasse 0.98. Der Macro F_1 Score, der daraus berechnet wird, beträgt 0.52.

2.3.6 Klassifizierung mit Topic Models

Aus den Topic Models von Kapitel 2.2 können die Kapitel auch direkt klassifiziert werden. Dies ist vor allem darum interessant, weil es sich beim Topic Modeling um unüberwachtes Lernen handelt. Im Gegensatz zu überwachten Algorithmen wie

RandomForest sind bei unüberwachten Algorithmen keine Trainingsdaten nötig. Falls die Topic Models genügend gute Resultate liefern würden, könnte für weitere Arbeiten der manuelle Aufwand zum Labeln der Daten eingespart werden. Um dies zu überprüfen wird wiederum der Testdatensatz mit 1000 gelabelten Testdaten verwendet. Alle Kapitel, welche als wichtigstes Topic ein *Zuschlag-Topic* haben, werden in die Klasse *relevant,* und alle weiteren in die Klasse *nicht relevant* eingeteilt. Für diese Klassifizierung kann wiederum eine Confusion Matrix erstellt werden.

Tabelle 9: Confusion Matrix (Topic Modeling Klassifizierung)

	Nicht relevant (klassifiziert)	*Relevant (klassifiziert)*
Nicht relevant (effektiv)	953 (TN)	16 (FP)
Relevant (effektiv)	21 (FN)	10 (TP)

Der F_1 Score für die positive Klasse beträgt bei der Topic Modeling Klassifizierung 0.35 und für die negative Klasse 0.98. Der Macro F_1 Score ist 0.67.

2.3.7 Patternmatching Klassifizierung

Um zu validieren, ob mit einem simplen Verfahren ebenfalls ein gutes Resultat erzielt werden kann, wird ein Patternmatching Algorithmus geschrieben. Die Überschrift sowie der Inhalt der Kapitel werden nach bestimmten Regex Patterns durchsucht. Wird eines davon gefunden, wird das Kapitel in die Klasse *relevant* eingeteilt, ansonsten in die Klasse *nicht relevant*. Folgende zwei Patterns werden definiert:

*\szk[0-9]**

*zuschlag\w**

Das erste Pattern findet Wörter, die mit «zk» beginnen. Eventuell folgen eine oder mehrere Zahlen. Beispiele sind «zk», «zk1» und «zk12». Das zweite Pattern findet alle Wörter, die mit «zuschlag» beginnen.

Tabelle 14 zeigt die Confusion Matrix für die Klassifizierung mit Patternmatching. Der F_1 Score ist 0.36 für die positive Klasse und 0.97 für die negative Klasse. Der Macro F_1 Score beträgt 0.66.

Tabelle 10: Confusion Matrix (Patternmatching Klassifizierung)

	Nicht relevant (klassifiziert)	*Relevant (klassifiziert)*
Nicht relevant (effektiv)	927 (TN)	42 (FP)
Relevant (effektiv)	15 (FN)	16 (TP)

3 Diskussion

In dieser Arbeit wird, aufgrund der 1000 Trainingsdaten, die Performance von den folgenden vier Modellen evaluiert.

1. RandomForest Modell mit Topic Modeling und Active Learning (Kapitel 2.3.4)
2. RandomForest Modell ohne Topic Modeling und Active Learning (Kapitel 2.3.5)
3. Klassifizierung mit Topic Models (Kapitel 2.3.6)
4. Patternmatching Klassifizierung (Kapitel 2.3.7)

Tabelle 11 zeigt den direkten Vergleich zwischen den Modellen. Das schlechteste Ergebnis erzielt das einfache RandomForest Modell mit einem Macro F_1 Score von 0.52. Dies liegt vor allem daran, dass aufgrund der unausgeglichenen Verteilung der positiven und der negativen Klasse in den Daten, zu wenige Einträge der positiven Klasse in den Trainingsdaten vorhanden sind. Das mit diesen Trainingsdaten trainierte RandomForest Modell teilt demnach zu wenige Kapitel in die positive Klasse ein, was zu einem schlechten F_1 Score für diese Klasse führt. Die Resultate von Pattern Matching und Topic Modeling zeigen, dass sich das Problem auch mit unüberwachten Verfahren lösen lässt. Die Resultate der beiden Modelle sind mit einem Macro F_1 Score von 0.66 für das Pattern Matching und einem Macro F_1 Score von 0.67 für das Topic Modeling ähnlich gut. Würde mehr Zeit für die Entwicklung der beiden Modelle aufgewendet, könnten diese unter Umständen weiter verbessert werden. Der Vorteil von Topic Modeling und Patternmatching ist, dass es sich dabei um unüberwachte Verfahren handelt und deshalb keine Trainingsdaten nötig sind, die unter grossem Aufwand manuell gelabelt werden müssen.

Das beste Resultat erzielt das, in dieser Arbeit hauptsächlich entwickelte, mit Topic Modeling und Active Learning kombinierte RandomForest Modell. Sowohl bei der positiven Klasse (F_1 Score von 0.55) als auch bei der negativen Klasse (F_1 Score 0.99) schliesst es besser ab, als die anderen Modelle. Mit 0.77 ist dementsprechend auch der Macro F_1 Score am höchsten.

Tabelle 11: Vergleich der Modelle

Modell	F_1 positive Klasse	F_1 negative Klasse	Macro F_1
Pattern Matching	0.36	0.97	0.66
Topic Modeling	0.35	0.98	0.67
Simpler RandomForest	0.06	0.98	0.52
Kombinierter RandomForest	0.55	0.99	0.77

Tabelle 12 zeigt die Reihenfolge der zehn wichtigsten Features beider Modelle im Vergleich. Die Spalte «Quelle» beinhaltet jeweils die Information, wo das Wort gefunden wurde, ob im Text, in der Überschrift oder in der Überschrift des Eltern Kapitels. Im linken Teil sind diejenigen Features markiert, welche im kombinierten Modell relevant sind, im einfachen Modell jedoch nicht und im rechten Teil diejenigen, die im einfachen Modell relevant sind, im kombinierten aber nicht. Die Wörter «schlüsselperson» und «qualität» werden nur im kombinierten Modell als wichtig eingestuft. Diese Wörter sind für Kapitel über Zuschlagskriterien relevant, weil eines der Zuschlagskriterien oftmals die Erfahrung der Schlüsselperson bewertet und ein weiteres die Qualität des Produkts. Interessant ist ausserdem, dass «preis» im einfachen Modell das zweitwichtigste Feature ist, im kombinierten Modell nur noch das zehntwichtigste. Eine mögliche Begründung dafür ist, dass «preis» üblicherweise nicht nur in Kapiteln über Zuschlagskriterien, sondern auch in diversen anderen auftritt. Zum Beispiel existiert in Ausschreibungen meist ein Kapitel über die Berechnung der Preiskurve. Dieses Kapitel wurde in den Trainingsdaten als *nicht relevant* eingeteilt. Im Vergleich zum simplen Modell berücksichtigt das kombinierte Modell mehr Features der Überschrift oder der Überschrift des Eltern Kapitels.

Im einfachen Modell werden Wörter wie «nr», «angabe» und «table» als wichtig eingestuft. Es ist nachvollziehbar, dass diese im kombinierten Modell an Relevanz verlieren, da es Wörter sind, die nicht speziell in Kapiteln über Zuschlagskriterien existieren. Zum Beispiel kommt das Wort «table» immer dann vor, wenn eine beliebige Tabelle im Text vorhanden ist. «table» ist die Markierung für den Start dieser Tabelle.

Tabelle 12: Wichtigste Features für einfaches und kombiniertes RandomForest Modell

Rang	Kombiniertes Modell		Einfaches Modell	
	Wort	Quelle	Wort	Quelle
1	zuschlagskriterium	Überschrift	zuschlagskriterium	Überschrift
2	zk	Überschrift	preis	Text
3	gewichtung	Text	zuschlagskriterium	Text
4	zuschlagskriterium	Text	zk	Text
5	angebot	Text	punkt	Text
6	punkt	Text	gewichtung	Text
7	zuschlagskriterium	Eltern-Überschrift	angebot	Text
8	schlüsselperson	Text	nr	Text
9	qualität	Text	angabe	Text
10	preis	Text	table	Text

Zusammengefasst kann gesagt werden, dass das kombinierte RandomForest Modell von allen Modellen am besten abschliesst. Um das einfache RandomForest Modell zu verbessern, müssten vor allem zusätzliche Labels der positiven Klasse gefunden werden. Wenn im Schnitt davon ausgegangen wird, dass 3% aller Kapitel der positiven Klasse angehören (vgl. Kapitel 2.3.2), müssten rund 8760 Trainingsdaten manuell klassifiziert werden. Der manuelle Aufwand wäre somit rund 9.3 Mal höher als beim kombinierten Modell. Ob die Performance eines einfachen Modells mit 8760 Trainingsdaten besser, gleich oder schlechter als die des kombinierten Modells ist, kann nicht abschliessend beurteilt werden. Ausserdem ist das Problem auch mit unüberwachten Verfahren lösbar. In dieser Arbeit wird mit diesen aber eine deutlich schlechtere Performance erzielt.

Wird das kombinierte RandomForest Modell genauer untersucht, zeigt sich, dass sich im Verlaufe der Runden vor allem die F_1 Scores der positiven Klasse verändern (vgl Abbildung 13). Nach der ersten Runde Active Learning beträgt der F1 Score für das positive Label 0.41. Dieser Wert steigt während den weiteren Runden auf 0.55. Der Wert für das negative Label bleibt über alle Runden, bei einem hohen Wert von 0.99,

relativ konstant. Es ist deutlich erkennbar, dass das Modell mehr Mühe hat, die positiven Labels richtig zu klassifizieren als die negativen. Mit Hilfe von Active Learning kann die Klassifizierung der positiven Labels verbessert werden. Wird die Confusion Matrix von Runde 1 (Tabelle 6) beziehungsweise Runde 10 (Tabelle 7) betrachtet, wird klar, dass das Modell nach der zehnten Runde insgesamt deutlich mehr Kapitel der positiven Klasse zuordnet als noch in der ersten Runde (24 statt 9), wobei auch die False Positives von 1 auf 9 Kapitel ansteigen. Dafür sinken die False Negatives von 22 auf 16.

Der Active Learning Algorithmus adressiert das Problem, indem er in jeder Runde überproportional viele Kapitel zum Trainieren auswählt, die schlussendlich der Klasse *relevant* zugeordnet werden. Werden Kapitel zufälligerweise aus allen Kapiteln ausgewählt, beträgt die Anzahl der relevanten Kapitel rund 3% (vgl. Kapitel 2.3.2), was 1.5 Kapiteln entspricht. Der Durchschnitt bei der Active Learning Selektion beträgt 21.6 Kapitel (43.2%). Tabelle 13 zeigt die Auswahl der Kapitel und ihre finale Einteilung in die Klassen.

Tabelle 13: Auswahl der Kapitel vom Active Learning Algorithmus (relevant vs. nicht relevant)

Runde	*relevant*	*nicht relevant*
1	26	24
2	18	30
3	27	23
4	28	22
5	16	33
6	23	26
7	18	31
8	20	30
9	19	30

Es gibt zwei Gründe, warum die Klassifizierung der positiven Klasse so viel schwieriger ist als die der negativen Klasse. Erstens sind auch nach Topic Modeling und allen Active Learning Runden deutlich mehr Kapitel der negativen Klasse als Kapitel der positiven Klasse in den Trainingsdaten vorhanden. Insgesamt gibt es 671 Trainings-

daten der Klasse *nicht relevant* und 263 der Klasse *relevant*. Zweitens sind Kapitel, die der Klasse *nicht relevant* angehören, oftmals einfacher zu klassifizieren. Dies wird insbesondere dann klar, wenn die Auswertung der prozentualen Anteile relevanter Kapitel pro Topic betrachtet wird. Tabelle 14 zeigt einen Ausschnitt davon. Die ganze Auswertung kann im Anhang 4 betrachtet werden.

Tabelle 14: Prozentualer Anteil relevanter Kapitel pro Topic

Rang	Topic ID	Anzahl	Relevant	Prozentual
1	4	292	167	57.2%
2	38	24	6	25.0%
3	32	130	30	23.1%
4	16	101	23	22.8%
…	…	…	…	…
38	39	81	0	0.0%
39	35	57	0	0.0%
40	26	35	0	0.0%
41	20	54	0	0.0%

Topic 39 kommt zum Beispiel 81 Mal in den Trainingsdaten vor, wird aber nie als *relevant* klassifiziert. Tabelle 15 zeigt beispielhaft ein Kapitel, das dem Topic 39 zugeordnet wurde. Es lässt sich gut erkennen, dass keine Anhaltspunkte bestehen, das Kapitel als relevantes Kapitel zu werten. Wörter wie «zuschlag», «zk», etc. kommen nicht vor. Deshalb ist es für die Algorithmen ebenfalls einfach, das Kapitel in die korrekte Klasse *nicht relevant* zu klassifizieren.

Tabelle 15: Beispiel eines Kapitels von Topic 39

Titel	6.5 Umgang mit Änderungen der Baukosten
Inhalt	Grundsätzlich besteht aufgrund einer Änderung der Baukosten kein Anspruch auf eine entsprechende Anpassung des Aufwands bzw. des Kostendachs. Eine Anpassung erfolgt nur, wenn sich aufgrund der Änderung auch zusätzliche Leistungen für den Anbieter ergeben. In diesem Fall erfolgt die Anpassung auf Basis der vorangehenden Ziffer.

Da es also viele Kapitel der negativen Klasse gibt, die einfach zu klassifizieren sind, ist die relative Anzahl Falschklassifikationen für diese Klasse gering und der F_1 Score dementsprechend hoch.

Bei der Analyse der Falschklassifikationen fällt auf, dass nicht immer der Algorithmus derjenige ist, der die Falschklassifikation verursacht hat, oder dass es zumindest fraglich ist, ob die Zuteilung des Menschen die bessere ist, als die des Algorithmus. Von den 16 False Negatives (FN) sind 10 klare Falschklassifikationen vom RandomForest Modell. Bei fünf Kapiteln ist es bei der erneuten Betrachtung nicht klar, welche Zuteilung die bessere war. Das Beispiel von Tabelle 16 zeigt, dass die Zuteilung nicht immer eindeutig ist. Der Mensch teilt dieses Kapitel in die Klasse *relevant* ein und der Algorithmus in die Klasse *nicht relevant*. Wird nur der Titel betrachtet, ist es eindeutig ein relevantes Kapitel, bei der alleinigen Betrachtung des Inhalts wiederum ist es eindeutig kein relevantes Kapitel. In der Kombination ist es fraglich, in welche Klasse das Kapitel eingeteilt werden sollte.

Tabelle 16: Unklare Falschklassifikation

Titel	ZK2: Bauprogramm (Abweichung der Bauzeit gegenüber Terminplan Bauherr)
Inhalt	1.16.1 gemäss Anhang Nr. : 7 1.16.2 gemäss Anhang Nr. : 7 1.16.3 gemäss Anhang Nr. : 7

Bei einem der 16 False Negatives ist es sogar so, dass der Algorithmus einen Fehler des Menschen bei der Klassifikation korrigiert hat. Tabelle 17 zeigt ein Kapitel, das der Mensch als *relevant* und der Algorithmus als *nicht relevant* klassifiziert hat. Bei der erneuten Betrachtung müsste die Zuteilung des Menschen korrigiert und dem Kapitel die Klasse *nicht relevant* zugewiesen werden.

Bei der Betrachtung dieses Beispiels wird klar, wie stark der Erfolg des Modells von der Güte der manuellen Klassifikation der Trainingsdaten abhängt. Während dem Active Learning Prozess stellt das Modell dem Menschen immer schwierigere Aufgaben, bei denen es auch für ihn nicht trivial ist, zu entscheiden, zu welcher Klasse das Kapitel gehört. Durch eine steigende Anzahl der Trainingsdaten können menschliche Fehler unter Umständen relativiert werden. Grundsätzlich aber kann das Modell maximal so gut sein, wie der Mensch zuvor die Trainingsdaten generiert.

Tabelle 17: Falschklassifikation vom Menschen

Titel	Das ZK „Preis“ wird wie folgt berechnet:
Inhalt	Die Angebotspreise werden gemäss Beschrieb in Dokument 1.1 „Allgemeine Bestimmungen“ bewertet.

Bei den False Positives wurden acht von neun Kapiteln klar von der Maschine falsch klassifiziert. Einige dieser Falschklassifikationen sind aussergewöhnlich und für einen Menschen nicht nachvollziehbar. Tabelle 18 zeigt zum Beispiel ein Kapitel, das weder im Titel noch im Text relevante Wörter wie «zk», «zuschlag», etc. beinhaltet. Trotzdem wird es der positiven Klasse zugewiesen. Hier macht sich der Nachteil des RandomForest Algorithmus bemerkbar, dass er durch die Kombination der vielen Entscheidungsbäume als eine Art Blackbox agiert. Dadurch ist es nicht möglich, nachzuvollziehen, warum das Kapitel der Tabelle 18 falsch klassifiziert wurde.

Tabelle 18: Falschklassifikation vom Algorithmus

Titel	Bogen
Inhalt	gemäss Punkt 7.3.3 DN table: 150 | St. | 24 | 65 | St. | 10 | 50 | St. | 10 |

Im Vergleich zu den anderen Modellen ist die Performance des kombinierten Random-Forest Modells gut. Wenn das Resultat jedoch unabhängig betrachtet wird, muss gesagt werden, dass ein F_1 Score von 0.55 für die positive Klasse nicht für jede Applikation ausreichend ist. Im Vergleich zu den 15 *True Positives* sind 9 *False Positives* und *16 False Negatives* immer noch viel. Der F_1 Score für die positive Klasse steigt im Verlaufe der Runden relativ konstant bis zum Schluss. Mit mehr Active Learning Runden könnte das Resultat mit grosser Wahrscheinlichkeit weiter verbessert werden, erfordert aber zusätzlichen, manuellen Aufwand.

Neben dem Vergleich der Modelle soll mit Arbeit erforscht werden, wie die hierarchische Struktur von Kapiteln für Machine Learning Algorithmen eingesetzt werden kann. Dazu werden die Kapitel bei den Überschriften der gleichen Ebene getrennt und die Unterkapitel dupliziert. Die Unterkapitel existieren separat, aber auch im Text der übergeordneten Kapitel. Der Titel der übergeordneten Kapitel wird als Attribut miteinbezogen (vgl. Kapitel 2.1.4). Diese Struktur ist erfolgreich, da dadurch keine leeren Kapitel entstehen. In Tabelle 12 ist ersichtlich, dass die Kapitelüberschriften oftmals als wichtiges Feature betrachtet werden. Das siebtwichtigste Wort «zuschlagskriterium» stammt sogar aus der Überschrift des Eltern Kapitels.

4 Schlussfolgerung und Ausblick

In dieser Masterarbeit wurde ein Verfahren entwickelt, das Topic Modeling mit einem RandomForest Algorithmus und Active Learning kombiniert, um bezüglich Zuschlagskriterien relevante Segmente in Ausschreibungsunterlagen zu finden. Parallel dazu wurden drei simplere Modelle evaluiert und mit dem kombinierten Modell verglichen. Bei den simpleren Modellen handelt es sich um ein RandomForest Modell ohne Topic Modeling und Active Learning, um ein Modell, das eine Klassifizierung direkt mit den Topic Models durchführt und um ein simples, auf Regex basierendes Patternmatching Modell. Es konnte gezeigt werden, dass das kombinierte Modell mit einem Macro F_1 Score von 0.77 deutlich besser abschliesst als das Patternmatching Modell (Macro F_1 Score von 0.66), das Topic Modeling (Macro F_1 Score von 0.67) und das simple RandomForest Modell (Macro F_1 Score von 0.52). Mit Active Learning konnte vor allen der F_1 Score der positiven Klasse verbessert werden. Über zehn Runden Active Learning steigt dieser relativ konstant.
Active Learning und Topic Models reduzieren den manuellen Aufwand, der für die Generierung der Trainingsdaten benötigt wird. In dieser Arbeit wurde berechnet, dass mit einem simplen RandomForest Modell ohne Topic Modeling und Active Learning der manuelle Aufwand 9.3 Mal höher wäre, um ein gleich gutes Resultat wie mit dem kombinierten Modell zu erzielen.

In dieser Arbeit wurde ausserdem ein Vorschlag gemacht, wie hierarchisch aufgebaute Textdokumente als Grundlage für Machine Learning Algorithmen dienen können. Dabei werden Dokumente jeweils bei der Überschrift der gleichen Ebene getrennt und die einzelnen Kapitel beinhalten jeweils auch deren Unterkapitel. Die Unterkapitel werden dupliziert und existieren zusätzlich als eigenständige Kapitel. Ausserdem werden die Überschriften der übergeordneten Kapitel als Attribute für die Kapitel erfasst.

Obwohl spannende Resultate erzielt wurden, gibt es bei allen Verfahren noch Verbesserungspotenzial. Zum Beispiel müsste erklärt werden, warum nur 5362 relevante Kapitel gefunden werden, wenn das Modell auf alle Kapitel ohne Label angewendet wird. Es ist gesetzlich geregelt, dass eine Ausschreibung Angaben zu den Zuschlagskriterien enthalten muss (Artikel 21, abs. 2, BöB). Deshalb müsste für jedes der 9423 Projekte

mindestens ein relevantes Kapitel gefunden werden. Einige Erklärungen dafür wurden bereits in der Arbeit geliefert. Zum Beispiel war die Indexierung mit ElasticSearch bisher noch nicht optimal, anderssprachige Ausschreibungen wurden ignoriert und nur PDF und docx Formate wurden unterstützt. Dies könnte in zusätzlichen Arbeiten verbessert werden. Ein weiterer Grund für die wenigen Kapitel ist aber auch der immer noch tiefe F_1 Score für die positive Klasse im kombinierten Modell. Im Vergleich zu den anderen Modellen ist der Wert zwar besser, dennoch ist ein F_1 Score von 0.55 immer noch kein hoher Wert. Es gibt verschiedene Ansätze, um das Resultat weiter zu verbessern. Erstens könnte die Anzahl der Active Learning Runden erhöht werden. Es wurde gezeigt, dass der F_1 Score für die positive Klasse im Verlaufe der Runden relativ konstant ansteigt (vgl. Abbildung 13). Iterativ könnten weitere Runden hinzugefügt werden bis der F_1 Score konvergiert. Dies ist aber mit zusätzlichem, manuellem Aufwand verbunden. Ausserdem wäre es möglich, statt dem RandomForest weitere Algorithmen wie Neuronale Netzwerke, Support Vector Machines, etc. zu testen oder die Parameter des RandomForest Modells weiter zu optimieren. Zuletzt könnten auch die Falschklassifikationen genauer untersucht werden, um Muster zu finden, wie der Algorithmus Fehler macht, und für diese Muster könnten spezifisch weitere Testdaten generiert werden.

Da das primäre Ziel dieser Arbeit war, ein kombiniertes Verfahren aus Topic Modeling, überwachtem Lernen und Active Learning zu erschaffen, wurden die unüberwachten Verfahren, die ebenfalls vorgestellt wurden, nicht perfektioniert. Sowohl das Topic Modeling wie auch das Patternmatching Verfahren könnten in einer zusätzlichen Arbeit verbessert werden.

Bezüglich der hierarchischen Struktur der Kapitel müsste untersucht werden, wie die Resultate interpretiert und weiterverarbeitet werden können. Wird zum Beispiel das Kapitel 1 eines Dokuments als relevant eingestuft und das Kapitel 1.2 ebenfalls, werden dann beide Kapitel behalten oder nur eines davon?

Mit dieser Arbeit kann gezeigt werden, dass das Potenzial für Machine Learning in Ausschreibungsunterlagen gross ist, aber auch noch viel Arbeit ansteht. Sie präsentiert eine von vielen Möglichkeiten, wie grosse, unstrukturierte Textdokumente wie Ausschreibungsunterlagen zukünftig maschinell weiterverarbeitet werden können.

Bibliografie

Aggarwal, C. C. (2018). *Machine Learning for Text*. Abgerufen von //www.springer.com/gp/book/9783319735306

Arora, S., Ge, R., Halpern, Y., Mimno, D., Moitra, A., Sontag, D., … Zhu, M. (2013). *A Practical Algorithm for Topic Modeling with Provable Guarantees*. 9.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, *3*(Jan), 993–1022.

Brants, S., Dipper, S., Eisenberg, P., Hansen-Schirra, S., König, E., Lezius, W., … Uszkoreit, H. (2004). TIGER: Linguistic Interpretation of a German Corpus. *Research on Language and Computation*, *2*(4), 597–620. https://doi.org/10.1007/s11168-004-7431-3

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Bundesgesetz über das öffentliche Beschaffungswesen (BöB). , SR 172.056.1 § (1994).

Doll, T. (2018, Juni 24). LDA Topic Modeling: An Explanation. Abgerufen von https://towardsdatascience.com/lda-topic-modeling-an-explanation-e184c90aadcd [Zugegriffen: 2019-06-28]

Fraj, M. B. (2017, Dezember 21). In Depth: Parameter tuning for Random Forest. Abgerufen von https://medium.com/all-things-ai/in-depth-parameter-tuning-for-random-forest-d67bb7e920d [Zugegriffen: 2019-06-28]

Gormley, C., & Tong, Z. (2015). *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine*. O'Reilly Media, Inc.

Hearst, M. A. (1997). TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Comput. Linguist.*, *23*(1), 33–64.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105. https://doi.org/10.2307/25148625

Konrad, M. (2016, Juli 13). Accurate Part-of-Speech tagging of German texts with NLTK. Abgerufen von https://datascience.blog.wzb.eu/2016/07/13/accurate-part-of-speech-tagging-of-german-texts-with-nltk/ [Zugegriffen: 2019-06-28]

Konrad, M. (2017, Mai 19). Lemmatization of German language text. Abgerufen von https://datascience.blog.wzb.eu/2017/05/19/lemmatization-of-german-language-text/ [Zugegriffen: 2019-06-28]

Krancher, O., & Stürmer, M. (2018). Explaining Multisourcing Decisions in Application Outsourcing. *ResearchGate*. Gehalten auf der Twenty-Sixth European Conference on Information Systems, Portsmouth. Abgerufen von https://www.researchgate.net/publication/324770459_Explaining_Multisourcing_Decisions_in_Application_Outsourcing

Liaw, A., & Wiener, M. (2002). *Classification and Regression by randomForest*. *2*, 6.

Loper, E., & Bird, S. (2002). NLTK: The Natural Language Toolkit. *arXiv:cs/0205028*. Abgerufen von http://arxiv.org/abs/cs/0205028

Mayumi Oshiro, T., Santoro Perez, P., & Baranauskas, J. (2012, Juli 1). *How Many Trees in a Random Forest? 7376*. https://doi.org/10.1007/978-3-642-31537-4_13

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*(Oct), 2825–2830.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2014). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*. https://doi.org/10.2753/MIS0742-1222240302

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24*(5), 513–523. https://doi.org/10.1016/0306-4573(88)90021-0

scikit-learn.org. (2019a). Cross-validation: evaluating estimator performance. Abgerufen von https://scikit-learn.org/stable/modules/cross_validation.html [Zugegriffen: 2019-06-16]

scikit-learn.org. (2019b). sklearn.ensemble.RandomForestClassifier. Abgerufen von https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html [Zugegriffen: 2019-06-28]

Settles, B. (2009). *Active Learning Literature Survey* [Technical Report]. Abgerufen von University of Wisconsin-Madison Department of Computer Sciences website: https://minds.wisconsin.edu/handle/1793/60660

SolidDocuments. (2019). SolidCommander (Version 10.0). Abgerufen von https://www.soliddocuments.com

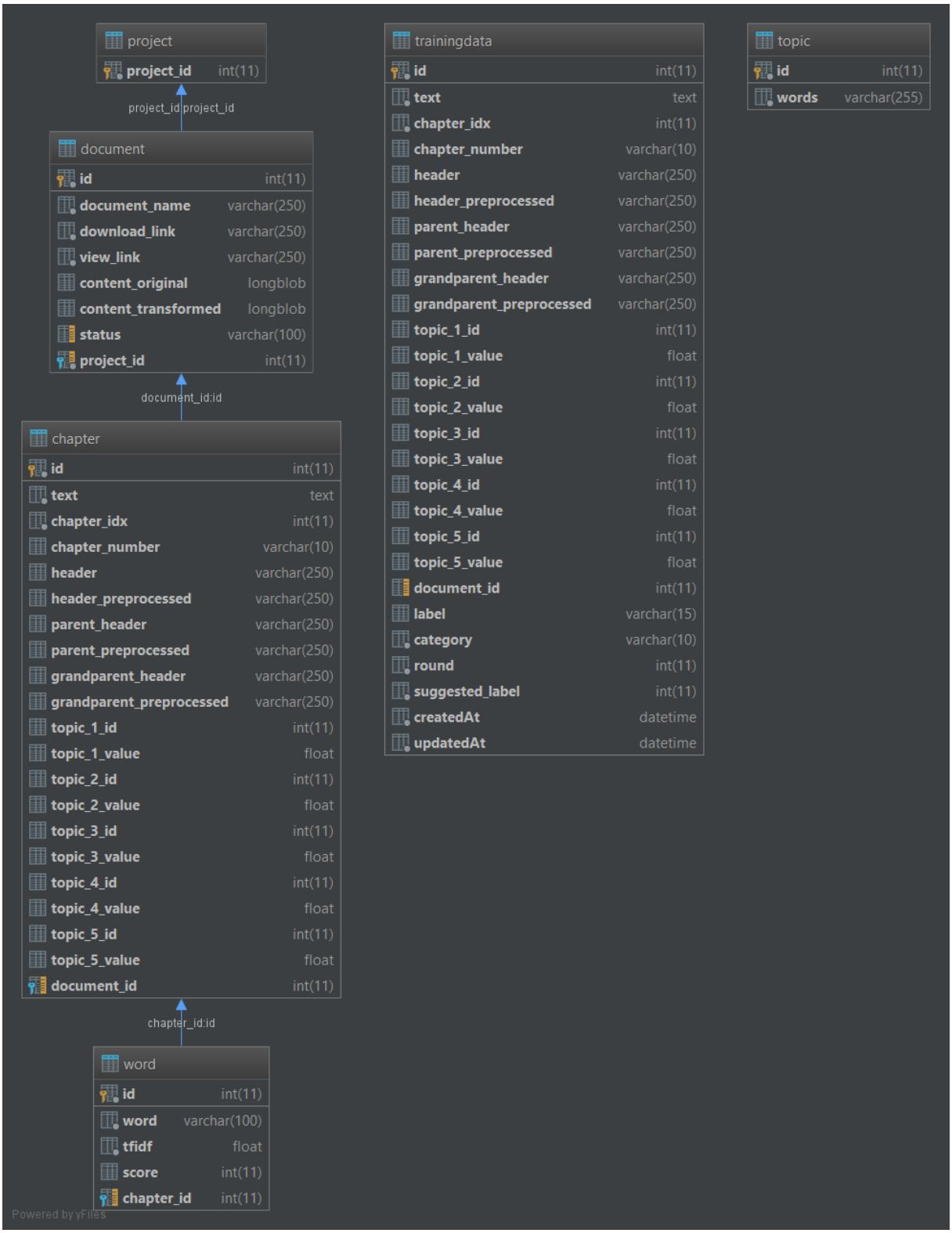
Übereinkommen über das öffentliche Beschaffungswesen. , 0.632.231.422 0.632.231.422 § (1996).

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang

Anhang 1: Struktur der Datenbank



Anhang 2: Wichtigste 10 Wörter pro Topic

0	angebot gültigkeit einreichung verbindlichkeit einreichen ab monat öffnung evaluation vollständig
1	art abs bauherr abnahme auftraggebers folgen prüfung mangel werk schriftlich
2	angabe weit vergabeverfahren allgemeine unternehmen ausschreibungsverfahren anbieter sonstig administrative unternehmensangabe
3	adresse name ort mail telefon plz fax auftraggeber ch auftraggebers
4	zuschlagskriterium zk gewichtung eignungsund bewertung qualität kriterium punkt evaluation referenzen
5	mm st inkl typ le stk dn fabrikat m2 50
6	bestimmung besonderer vergabeverfahren werkleistung weit übersicht 01 teil 102 objektspezifisch
7	information verhandlung administratives vertraulichkeit auftraggeber vertraulich wto behandeln abkommen weit
8	variante teilangebot zulassen nein ja zulässig option einreichen technisch administratives
9	ausschreibung organisation eignungsund gegenstand simap ausschreibungsunterlage vorliegend offen frage 200
10	allgemeine bedingung bauherr ausschreibungsverfahren ausschreibungsunterlage teilnahmebedingung bedingungen atb bauen nachhaltiges
11	verfahren offenes verfahrensart gatt wto offen öffentlich recht beschaffungswesen allgemeine
12	000 200 pos begriffsdefinition gelten betreffend ausmassbestimmung bedingung vergütungsregelung anwendungsregel
13	anforderung technisch erfüllen spezifikationen besonderer müssen regelwerk lösung erfüllung prüfung
14	2018 01 00 12 10 datum 11 uhr 2017 16
15	100 110 200 objekts zweckbestimmung lage umfang organisation arbeit 120
16	schlüsselperson projekt referenzen referenzobjekt funktion referenz referenzperson formular erfahrung name

17	beilage unterlage einzureichende dokument ausgefüllt einreichen formular ausschreibungsunterlage vollständig leistungsverzeichnis
18	gemäss ziffer werkvertrag vorgesehen geschäftsbedingung sicherheit finanzielle aufgabenbeschrieb vertragsurkunde teuerung
19	sprache deutsch ausschreibungsunterlage verfahrenssprache vergabeverfahren angebotsunterlage bedingung administratives frage offerte
20	zürich stadt vertragspartnerinnen ag verhaltenskodex hochbau kanton postfach amt bauen
21	total fr et montage transport armaturen lv übertragen le 800
22	teil submission grundlage submissionsunterlage flughafen ag ausschreibungsunterlage dokument zürich sehen
23	schutz baustelle person umgebung eigentum 500 boden massnahmen gewässer vorgabe
24	los aufteilung bauauftrag vorsehen verschieden ausgeschrieben folgen nein mehrere beschaffungen
25	anbieter anbieters selbstdeklaration bestätigung anbieterin unterschrift arbeitsbedingung dass arbeitsschutzbestimmung einhaltung
26	232 236 allgemein 81 51 installation 237 22 st inbetriebsetzung
27	bkp neubau 244 arbeitsgattung leistungsverzeichnis npk 211 zusammenstellung auftrag objekt
28	eignungskriterium nachweis leistungsfähigkeit erfüllen ek technisch kriterium wirtschaftlich jahr erfüllung
29	nr positionstext pos anhang artikel absatz referenzobjekt plan tel angebot
30	chf mwst versicherung exkl netto vergütung pro inkl fr eingabesumme
31	subunternehmer lieferant arbeitsgemeinschaft firma arge subunternehmen zulassen übernehmen 29 beteiligt
32	preis bewertung erhalten tief kosten punkt erfolgen punktzahl maximal menge
33	04 343 dämmung gruppe dn fr zusammenstellung neu km st
34	termin frist frage bauablauf bauprogramm schriftlich prämie 2019 strafe 600
35	norm sia 118 ergänzung 2013 regelwerk gelten allgemein bauarbeit richtlinie

36	bietergemeinschaft unternehmen zulassen unternehmensangabe beteiligt eignungsnachweis en land strasse kontaktperson
37	unternehmer bauleitung arbeit einzurechnen ausführung einheitspreis baustelle kosten bauherrschaft material
38	beschaffungsobjekt ort beschaffung option aufgabenbeschrieb vertrag administratives beginn ausführung nein
39	leistung ausgangslage beschreibung ausschreibungsgegenstand option einleitung umfang sowie auftrag gegenstand

Anhang 3: Detaillierte Auswertung Active Learning

Runde	TP	TN	FP	FN	F1 Positiv	F1 Negativ	F1 Macro
1	8	969	1	22	0.41	0.99	0.70
2	9	967	3	21	0.43	0.99	0.71
3	9	965	4	22	0.41	0.99	0.70
4	11	965	4	20	0.48	0.99	0.73
5	13	956	13	18	0.46	0.98	0.72
6	13	965	4	18	0.54	0.99	0.77
7	12	964	5	19	0.50	0.99	0.74
8	12	964	5	19	0.50	0.99	0.74
9	14	964	5	17	0.56	0.99	0.77
10	15	960	9	16	0.55	0.99	0.77

Anhang 4: Anzahl relevante Kategorien pro Topic

Rang	Topic ID	Anzahl	Relevant	Prozentual
1	4	292	167	57.2%
2	38	24	6	25.0%
3	32	130	30	23.1%
4	16	101	23	22.8%
5	2	59	12	20.3%
6	24	52	7	13.5%
7	22	50	6	12.0%
8	28	122	13	10.7%
9	0	93	8	8.6%
10	11	47	4	8.5%
11	21	49	4	8.2%
12	13	65	5	7.7%
13	34	58	4	6.9%
14	6	58	4	6.9%
15	27	45	3	6.7%
16	9	49	3	6.1%
17	30	55	3	5.5%
18	25	62	3	4.8%
19	18	47	2	4.3%
20	8	24	1	4.2%
21	29	55	2	3.6%
22	12	62	2	3.2%
23	36	32	1	3.1%
24	17	75	2	2.7%
25	31	39	1	2.6%
26	1	45	1	2.2%
27	Kein Topic	181	4	2.2%
28	15	50	1	2.0%

29	10	59	1	1.7%
30	37	138	2	1.5%
31	14	88	1	1.1%
32	7	60	0	0.0%
33	5	85	0	0.0%
34	19	38	0	0.0%
35	3	62	0	0.0%
36	33	29	0	0.0%
37	23	61	0	0.0%
38	39	81	0	0.0%
39	35	57	0	0.0%
40	26	35	0	0.0%
41	20	54	0	0.0%

Anhang 5: Code

Der Source Code für das entwickelte Verfahren kann im folgenden GitHub Repository gefunden werden:

https://github.com/janikEndtner/machine-learning-in-tender-documents

Dank

Zum Gelingen meiner Masterarbeit haben zahlreiche Personen beigetragen. Als erstes möchte ich mich bei Dr. Matthias Stürmer für die angenehme Zusammenarbeit bedanken. Er betreute die Arbeit sehr lösungsorientiert, praxisnah und immer euphorisch. Ausserdem möchte ich mich bei Prof. Dr. Philipp Baumann bedanken. Bei unterschiedlichen Machine Learning Problemen war er mit seinem Fachwissen eine grosse Hilfe. Zuletzt gilt mein Dank Christina Strycker und Doris Endtner für das Gegenlesen der Arbeit und dem Team der Forschungsstelle Digitale Nachhaltigkeit für die lehrreichen Diskussionen über diverse Programmierthemen.

Selbständigkeitserklärung

„Ich erkläre hiermit, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe o des Gesetzes vom 5. September 1996 über die Universität zum Entzug des aufgrund dieser Arbeit verliehenen Titels berechtigt ist.“

Handschriftliche Unterschrift

Bern, 12. Juli 2019 Janik Endtner

Veröffentlichung der Arbeit

I.d.R. werden schriftliche Arbeiten in der Bibliothek des Instituts für Wirtschaftsinformatik öffentlich zugänglich gemacht.

☒ Hiermit erlaube ich, meine Arbeit in der Bibliothek des Instituts für Wirtschaftsinformatik zu veröffentlichen.

☐ Ich möchte auf eine Veröffentlichung meiner Arbeit verzichten.

Falls eine Vertraulichkeitserklärung unterschrieben wurde, ist es Sache des Studierenden, das Einverständnis des Praxispartners einzuholen. Es muss der Arbeit eine schriftliche Bestätigung des Praxispartners beigelegt werden.

Die Benotung der Arbeit erfolgt unabhängig davon, ob die Arbeit veröffentlicht werden darf oder nicht.

Handschriftliche Unterschrift

Bern, 12. Juli 2019 Janik Endtner