UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR
THEORETISCHE INFORMATIK

Vorlage für die LaTeX-Klasse »uzl-thesis« zur Nutzung bei Bachelor- und Masterarbeiten an der Universität zu Lübeck

Template for the LaTeX Class "uzl-thesis" for Bachelor's and Master's Theses Written at the University of Lübeck

Bachelorarbeit

verfasst am
Institut für Theoretische Informatik

im Rahmen des Studiengangs
Informatik
der Universität zu Lübeck

vorgelegt von
Leonard

ausgegeben und betreut von
X

mit Unterstützung von
Harry Hilfreich

Lübeck, den 1. Januar 2021

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

_____

Leonard

Zusammenfassung

Es ist nicht leicht, eine Abschlussarbeit so zu schreiben, dass sie nicht nur inhaltlich gut ist, sondern es auch eine Freude ist, sie zu lesen. Diese Freude ist aber wichtig: Wenn die Person, die die Arbeit benoten soll, wenig Gefallen am Lesen der Arbeit findet, so wird sie auch wenig Gefallen an einer guten Note finden. Glücklicherweise gibt es einige Kniffe, gut lesbare Arbeiten zu schreiben. Am wichtigsten ist zweifelsohne, dass die Arbeit in gutem Deutsch oder Englisch verfasst wurde mit klarem Satzbau und gutem Sprachrhythmus, dass keine Rechtschreib- oder Grammatikfehlern im Text auftauchen und dass die Argumente der Autorin oder des Autors klar, logisch, verständlich und gut veranschaulicht dargestellt werden. Daneben sind aber auch gut lesbare Schriftbilder und ein angenehmes Layout hilfreich. Die Nutzung dieser LaTeX-Vorlage hilft der Schreiberin oder dem Schreiber dabei zumindest bei Letzterem: Sie umfasst gute, sofort nutzbare Designs und sie kümmert sich um viele typographische Details.

Abstract

It is not easy to write a thesis that does not only advance science, but that is also a pleasure to read. While the scientific contribution of a thesis is undoubtedly of greater importance, the impact of *writing well* should not be underestimated: If the person who grades a thesis finds no pleasure in the reading, that person are also unlikely to find pleasure in giving outstanding grades. A well-written text uses good German or English phrasing with a clear and correct sentence structure and language rhythm, there are no spelling mistakes and the author's arguments are presented in a clear, logical and understandable manner using well-chosen examples and explanations. In addition, a nice-to-read font and a pleasing layout are also helpful. The LaTeX class presented in this document helps with the latter: It contains a number of ready-to-use designs and takes care of many small typographical chores.

Danksagungen

This is the place where you can thank people and institutions, do not try to do this on the title page. The only exception is in case you wrote your thesis while working or staying at a company or abroad. Then you should use the `Weitere␣Unterstützung` key to provide a text (in German) that acknowledges the company or foreign institute. For instance, you could use texts like »Die Arbeit ist im Rahmen einer Tätigkeit bei der Firma Muster GmbH entstanden« or »Die Arbeit ist im Rahmen eines Forschungsaufenthalts beim Institut für Dieses und Jenes an der Universität Entenhausen entstanden«. Do not name and thank individual persons from the company or foreign institute on the title page, do that here.

Inhaltsverzeichnis

1

Einleitung

2

Hauptteil

2.1 Motivation

Die digitalisierte Welt generiert täglich riesige Mengen an neuen Informationen. Die Kapazitäten, die ein Mensch aufbringen kann, um solche Massen an Daten zu organisieren und zu verstehen, sind schon lange übertroffen. Laut Statista wurden 2018 33 Zettabyte an Daten generiert mit einer prognostizierten Steigerung bis 2025 um 530% auf 175 Zettabyte. Dieser dramatische Anstieg zeigt die Dringlichkeit für effiziente Algorithmen und Modelle der Datenverarbeitung. Topic Modeling (dt. Themenmodellierung) beschreibt eine Gruppe von Verfahren, die es ermöglichen, große elektronische Datensammlungen automatisiert zu durchsuchen, organisieren und zu verstehen. Es können Muster innerhalb der Daten entdeckt und Themen extrahiert werden. Dabei stellen Themenmodelle statistische Modelle dar, die Verwendung in der Inferenz abstrakter Themen in unsortierten Datenmengen finden. In einer Welt von exponentiell wachsenden Datenmengen finden Methoden der Themenmodellierung stetig eine breitere Anwendung. Bereits heute wird Themenmodellierung in vielen Bereichen der Wirtschaft, Wissenschaft und Informationstechnologie verwendet. Um semantische Folgerungen aus Datenmengen zu generieren, gibt es verschiedene Ansätze – in dieser Arbeit wird es um das generative Modell ‚Latent Dirichlet Allocation' gehen. Dabei werden ähnliche Wörter, die in ähnlichen Kontexten vorkommen in einem Cluster gruppiert.

2.2 Ziel

Diese Arbeit wird die Theorie der Themenmodellierung anhand des Beispiels des Zweckverband Ostholstein (ZVO) implementieren und die bezüglichen Parameter im Sinne der Auswertung bewerten. Der ZVO erhält jährlich ein große Menge an Kundenanfrage. Diese werden momentan händisch an die jeweils zuständige Abteilung weitergeleitet. Der Prozess soll zukünftig automatisch durch einen Klassifikationsmechanismus funktionieren. Nach der Implementation eines LDA Algorithmus zur Inferenz verschiedener Abteilungen aus den Kundenanfragen, kann die momentan händische Kategorisierung bewertet werden. Diese Arbeit beschäftigt sich mit der Vorhersage der Qualität des Klassifikators, indem die Qualität der manuell erstellten Kategorien und Kundenanfrage-

Gruppen untersucht und mit den Ergebnissen verschiedener Themenmodellierung verglichen wird. Das Ergebnis einer Themenmodellierung hängt stark von der Qualität der Daten ab, die sie als Input bekommt. Diese Daten durchlaufen eine Reinigungsphase, bevor sie klassifiziert werden, um sie in eine gut zu verarbeitende Form zu bringen.

2.3 Notation

- $\mathcal{K}$ ist die Anzahl der Themen in einem Topic-Modell $\mathcal{M}$
- Ein Model $\mathcal{M}$ repräsentiert den Korpus $\mathcal{D}$
- Eine Menge von Dokumenten ist ein Korpus $\mathcal{D}$
- Ein Dokument d eines Korpus ist eine Menge von $\mathcal{N}$-vielen Worten

2.4 Modellvergleich

Latent Dirichlet Allocation

Themenmodellierung besteht aus vielen Methoden, die meist verbreitete ist die „Latent Dirichlet Allocation (LDA)“, was als Bag of Word modelliert ist, also keine Kontextinformationen beinhaltet. Dieses Verfahren ist eine Weiterentwicklung des ‚PLSI‘, das durch zwei Dirichlet-Priors ergänzt wurde. LDA liegt ein generierender Prozess zugrunde, den zwei Dirichlet Verteilungen maßgeblich beeinflussen: die Dokument-Themen Verteilung, die die Ausprägungen verschiedener Themen in einem Dokument beschreibt, und die Themen-Wörter Verteilung, die die Wahrscheinlichkeit beschreibt, dass ein bestimmtes Wort in einer gewissen Regularität in einem Themenbereich vorkommt. Dabei geht man davon aus, dass ein Dokument eine Verteilung von Themen ist, während ein Thema als eine Verteilung über Wörter betrachtet wird. Die Wahrscheinlichkeit, dass ein bestimmtes Dokument generiert wird, ist das Produkt der Wahrscheinlichkeiten der beiden Verteilungen mit den Wahrscheinlichkeiten zweier multinomialen Verteilungen, die erst zufällig Topics, wie in der Dirichlet-Verteilung definiert, auswählen und aus diesen dann, mithilfe der zweiten Dirichlet-Verteilung, Wörter aus diesen Topics herleiten, wodurch das Enddokument entsteht. Das Enddokument wird höchstwahrscheinlich stark von dem gegebenen Dokument abweichen, jedoch kann durch anpassen der Dirichlet-Verteilungen ein Optimierungsproblem formuliert werden, nach dem die Dirchlet-Verteilungen gesucht werden, die ein möglichst ähnliches Dokument generieren.

Latent Semantic Analysis (LSA)

Ein anderes verbreitetes Verfahren ist das „Latent Semantic Analysis“ (LSA), welches auf das Finden von sogenannten Hauptkomponenten in Dokumenten abzielt. Dadurch können sowohl ähnliche Wörter gefunden, als auch Textbereiche, die inhaltliche Überschneidungen mit einem bestimmten Begriff haben, aber das Wort selber nicht enthalten, gefunden werden. Die Methode basiert auf dem Prinzip der Singulärwertzerlegung(SVD). Als Ausgangslage wird aus einer Textsammlung eine Term-Dokument-Matrix erstellt. Diese Matrix wird in der SVD als Produkt von drei Matrizen dargestellt, von denen die

mittlere eine Diagonalmatrix darstellt. Die Werte auf der Diagonalen lassen daraus die Topics der Textmenge ablesen. Auf das SVD Verfahren selbst hat der Entwickler wenig Einfluss. Um Rauschen zu verhindern, kann jedoch die Anfangsmatrix mithilfe der term-frequency und inverse-document-frequency verbessert werden, was sich auf das Gesamtergebnis auswirkt. LSA stellt sich als ein attraktives Verfahren heraus, da es Synonyme besser erkennen kann, als LDA und wird heutzutage unter anderem intensiv in dem Bereich des Digital Marketings genutzt.

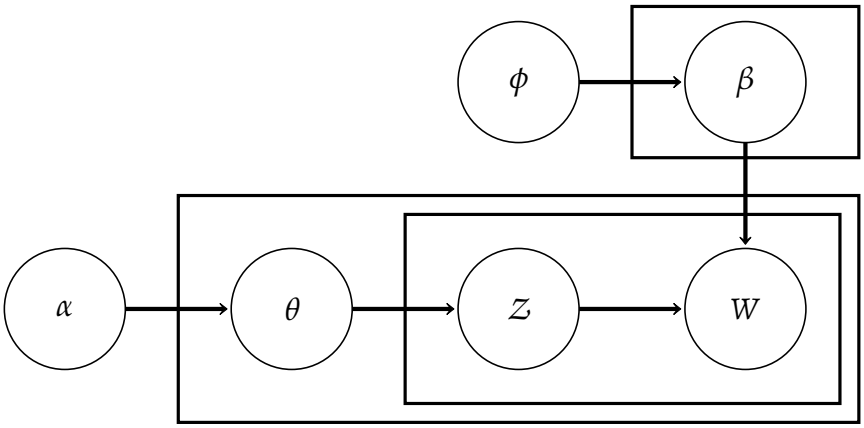
Non-Negative Matrix Factorization (NMF)

Ein weiteres Verfahren, das auch mit Matrizen funktioniert, wird „Non-Negative Matrix Factorization“ (NMF) genannt. Dabei wird eine Matrix, die Wörter auf Dokumente abbildet, in zwei Teilmatrizen faktorisiert. Die erste Teilmatrix stellt die Topics in Dokumenten, die zweite die Wörter in Topics dar. Dadurch kann Speicherplatz gespart, und Themen aufgedeckt werden. Das Verfahren beginnt mit zwei möglichen faktorisierten Matrizen und verbessert sich durch die Errorfunktion iterativ, bis das Ergebnis gut genug ist. Dabei werden die errechneten Werte mit der gegebenen Matrix verglichen und angepasst.

2.5 Grundlagen der Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation ist ein grundlegendes und bekanntes Verfahren aus der Natürlichen Sprachverarbeitung. Das Prinzip der Themenmodellierung basiert auf einer Menge an Dokumenten, die den Korpus darstellen. Dabei werden alle Dokumente als Menge von Wörtern angenommen, die als Bag of Words modelliert sind. Dabei hat weder die Reihenfolge, noch die Groß- und Kleinschreibung Einfluss auf das Ergebnis. Die Themen werden allein an der Vorkommenswahrscheinlichkeit der Wörter ohne Reihenfolgen- oder Kontextinformationen erkannt. Durch die Reduktion der Dimension wird die Effizienz gesteigert. Somit wird also jedes Dokument durch eine Verteilung der enthaltenen Wörter repräsentiert.

Bezüglich der Namensgeben, steht Latentfür alles, was wir im Vorhinein nicht kennen. Im Fall LDA handelt es sich um die Themen, die in einem Dokument zu einem bestimmten Teil vertreten sind. ”Dirichlet”beschreibt eine Verteilung von Verteilungen. Dies ist vergleichbar mit einem Würfel, bei dem regulierbar ist, wie gleichmäßig die Zahlen gewürfelt werden. Dabei ist der Würfel eine Verteilung und die Aufteilung der Gleichmäßigkeit auch. Bei der Topic-Modellierung bedeutet Dirichlet eine Verteilung von Topics in Dokumenten und eine Verteilung von Wörtern in Topics. Die Ällocation”weist mithilfe der errechneten Dirichlet-Verteilungen Topics Wörter und Dokumenten Topics zu. Eine Besonderheit bei der Themenerkennung mit LDA ist, dass die Anzahl der gesuchten Themen K vorgegeben werden muss. Oft ist diese vorher jedoch nicht bekannt und muss über Hilfsverfahren, wie der Perplexitätsberechnung ermittelt werden. Die Funktionsweise von LDA ist über folgende graphische Abbildung beschrieben:



Dabei beschreibt W als einzige nicht verborgene Variable eines von N Wörtern des Dokuments. Das Wort ist semantisch einem Thema Z zugeordnet. Das Thema wiederum hängt von der Themen-Verteilung theta des Dokuments ab, das als ein Elment der M vorliegenden Dokumente betrachtet wird. Neben dem Thema, wird jedes Wort auch von der jeweilgen Thema-Wort-Verteilung der K Themen beeinflusst. Das Modell und dessen Verteilungen kann durch die Parameter α und β angepasst werden. α kann bestimmt die Intensität der Dokument-Themen-Verteilung, während β die der Themen-Wort-Verteilung beeinflusst. Bei einem großen α ist die Verteilung der Themen in einem Dokument ähnlicher. Zusätzlich werden bei LDA zwei Bedingungen verfolgt, die von den beiden Parametern beeinflusst werden können. Erstens strebt man für alle Wort eines bestimmten Dokuments so wenig zugeordnete Themen an, wie möglich. Zweitens soll ein Thema über so wenig relevante Wörter wie möglich verfügen. Die beiden Ziele stehen in einer Wechselbeziehung zueinander, da eine minimale Anzahl an vertretenen Themen in einem Dokument zu maximal vielen Wörtern in diesen Themen führt. Die minimale Anzahl an Themen wäre erreicht, wenn man alle Wörter eines Dokuments einem Thema zuweist. Dadurch verfügt das Thema jedoch über alle Wörter des Dokuments. α befindet sich in dem Bereicht $[0, 1]$ mit sinnvollen Werten zwischen $[0.01, 0.1]$, während $\beta = 0.01$ durchschnittlich die besten Ergebnisse liefert. Große Werte führen zu einer Gleichverteilung, die wiederum eine Verschlechterung der Perplexität bedeutet. Somit bietet die Perplexität ein Mittel, um α und β optimal für die individuelle Anwendung zu finden.

Bei LDA werden zwei Verteilungen aus den Dokumenten $d \in \mathcal{D}$ und $k \in \mathcal{K}$gelernt: die Dokument-Topic-Verteilung θ und die Topic-Wort-Verteilung ϕ. Dabei gibt die Dokument-Topic-Verteilung an, mit welcher Wahrscheinlichkeit das Dokument zu jedem Themen gehört. Die Topic-Wort-Verteilung berechnet die Wahrscheinlichkeit, dass ein Wort einem Thema angehört. $\mathcal{M} = LDA(\mathcal{D})$ beschreibt ein LDA Modell, das auf der Dokumentenmenge/Korpus $\mathcal{D}$ trainiert wurde.

Der generative Prozess

Der Algorithmus hinter LDA generiert neue Dokumente mithilfe von Dirichletverteilungen $Dir(\gamma)$ und Multinomialverteilungen $Multinom(\delta)$. Die Verteilungen θ und ϕ werden errechnet, indem iterativ neue Dokumente über andere Verteilung generiert werden, bis das generierte Dokument die Anforderungen befriedigt, dann können die Verteilungen

abgelesen werden, mit denen das Dokument erstellt wurde. Der Prozess verläuft folgendermaßen:

1. Wähle ein θ als $Dir(\alpha)$
2. Wähle ein ϕ als $Dir(\beta)$
3. Für jedes Wort w and Stelle $i = 1, ..., N$ im Dokument d:
   3.1 Wähle ein Thema $z_{d,i}$ als $Multinom(\theta_d)$
   3.2 Wähle ein Wort $w_{d,i}$ als $Multinom(\phi_{zd,i})$

Somit kann der Algorithmus nun neue Dokumente erstellen und das Ergebnis durch die Parameter, wie Alpha und Beta, anpassen, bis das Ergebnis ähnlich genug zu dem Anfangsdokument ist. Dann ist die Verteilung der Themen in diesem Dokument bekannt. Bei der Anwendung von LDA für praktische Problemstellungen, geht LDA das Prinzip rückwärts durch, d.h. für bestehende Gruppen an Dokumenten werden Verteilungen gesucht, durch die das Dokument generiert hätte werden können.

$$P(w, z, \theta, \phi, \alpha, \beta) = \prod P(\theta, \alpha) \cdot \prod P(\alpha, \beta) \cdot \prod P(z,) \cdot \prod P(w \mid \phi) \tag{2.1}$$

Die Formel beschreibt die totale Wahrscheinlichkeit des LDA Modells. Sie setzt sich zusammen aus den Produkten der Dirichlet Verteilung der Topics und der Wörter zusammen mit den multinomialen Verteilungen der Themen und Wörter. Die Schwierigkeit des Algorithmus besteht in der Berechnung der θ-Verteilung der gegebenen Dokumente für latente Variablen. Dies lässt sich durch folgende Wahrscheinlichkeitsverteilung ausdrücken:

$$P(\theta, z \mid w, \alpha, \beta) = \frac{P(\theta, z, w \mid \alpha, \beta)}{P(w \mid \alpha, \beta)} \tag{2.2}$$

Die Formel berechnet die Wahrscheinlichkeit der Verteilung unter einem bestimmten Thema gegeben der α und β Parameter und dem bekannten Wort. Die Wahrscheinlichkeit kann nicht exakt bestimmt werden, weshalb Verfahren wie Gibbs Sampling diese approximieren.

2.6 Daten

Die Daten liegen in folgendem Format vor:

	filename	subject-message	Ablesung	Allgemeiner Schriftverkehr	BM-Beschwerden	KA-Kommunaler Abfall	MaKo-Klärfälle	ZSB-Aktionen	ZSB-BAV	ZSB-Zählerdatenmanagement	Forderungsmana
0	2015/11/14658664	zählerstände verbrauchsstellen kundennummer ma...	1	0	0	0	0	0	0	0	
1	2015/11/14669252	eigentumsänderung niederschlagswasserbeseitigu...	0	0	0	0	0	1	0	0	
2	2015/11/14848548	ovg urteil schreiben schmidt verbandsvorsteher...	0	0	0	0	0	0	0	0	
3	2015/11/14844310	kontaktformular holding ggf kundenummer anrede...	0	0	0	0	0	0	0	0	
4	2015/11/14789155	shop eutin kundennummer herrn gewerbe standort...	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	
133040	2019/09/27562984 2013/12/10037247	überweisungsbeleg	0	1	0	0	0	0	0	0	

Relevant für die Auswertung sind die subject-message und die jeweilige Abteilung. Die Tabelle verfügt über eine Matrix mit 18 Abteilungen, von denen pro subject-message eine mit einer 1 versehen ist. Dies beschreibt die Abteilung, der diese Anfrage manuell zugeordnet wurde. Die Daten in subject-message sind bereits bereinigt, also liegen vor, wie in diesem Beispiel, der Zeile 0:

```
In [9]: df.at[0,'subject-message']
Out[9]: 'zählerstände verbrauchsstellen kundennummer markieren lieferantenwechsel anrede frau vorname rosemarie nachname h
        orstmann straße ruschkamp plz ort scharbeutz telefonnummer mobilnummer faxnummer adresse ablesedatum datum zähler
        zählerart erdgas zählernummer zählerstand zählerart wählen zählernummer zählerstand zählerart wählen zählernummer
        zählerstand'
```

Um die Einträge in eine computer-lesbare Form zu verwandeln, muss ein Dictionary erstellt werden, dass alle Wörter auf eine Anzahl ihrer Vorkommen abbildet. Dafür müssen die Wörter als alleinige Listeneinträge einlesbar sein:

```
In [4]: df.at[0,'subject-message'].split()
Out[4]: ['zählerstände',
         'verbrauchsstellen',
         'kundennummer',
         'markieren',
         'lieferantenwechsel',
         'anrede',
```

Gibbs Sampling

Gibbs Sampling ist eine Technik, um ein LDA Modell zu trainieren. Für das menschliche Auge kann ein Wort eines Dokuments intuitiv einem Thema zugordnet werden. Für den Computer ist dies schwer, da er die Semantik des Wortes nicht kennt. Mit Gibbs Sampling werden Wörter iterativ Topics zugewiesen basierend auf allen restlichen Wörtern und Dokumenten. Dabei ist zum Einen relevant, was die führende Topic in dem jeweiligen Dokument ist und zum Anderen welcher Topic die gleichen Wörter im gesamten Korpus bereits am häufigsten zugeordnet wurden. Die beiden Beobachtungen werden multipliziert, um die Kombination mit der maximalen Wahrscheinlichkeit zu finden. Um zu verhindern, dass ein Wort einem Topic nicht zugewiesen werden kann, das sich nicht in dem bestimmten Dokument befindet, wird der Anzahl der jeweils vorkommenden Themen im Dokument α und der Anzahl der belegten Themen des gleichen Wörter in anderen Dokumenten β addiert. Wurde ein Wort einem Topic zugewiesen wird das nächste

Wort betrachtet. Nun ist zu beachten, dass sich die Verteilungen zur Berechnung jedes mal ändern. Somit haben wir einen iterativen Lernfortschritt bei der Topicverteilung.

Hellinger Distanz

In der Wahrscheinlichkeitsrechnung kann die Hellinger Distanz als Maß genutzt werden, um die Ähnlichkeit zweier Verteilungen zu berechnen. Für diese Anwendung bietet sich die Hellinger Distanz an, da sie oft im Kontext von Modellen und Topics verwendet wird. Zusätzlich lässt sie sich gut in Gensim Implementationen einbauen. Neben der Verteilungsvergleich kann auch eine Hellingermatrix erstellt werden, die jedes Topic des einen Modells mit jedem Topic eines zweiten Modells vergleicht. Somit wird ist es möglich die Ähnlichkeit zwischen zwei Modellen festzustellen. Formal errechnet sich die Distanz zwischen zwei diskreten Verteilungen X und Y aus folgender Gleichung:

$$H(X,Y) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{k}(\sqrt{x_i} - \sqrt{y_i})^2} \tag{2.3}$$

Datenreinigung

Bevor eine Themenmodellierung auf Daten durchgeführt werden kann, müssen die Daten einem Prozess unterzogen werden. Dieser beginnt mit der Datenaquise, also der Akquirierung bestimmter relevanter Daten. Im Falle der ZVO bedeutet dies, dass es genügend Kundenanfragen gibt, die verarbeitet werden können. Wenn diese Daten bestehen, werden sie auf die relevanten Wörter reduziert, aus denen eine bedeutsame Inferenz von Informationen möglich ist, sodass unter anderem die sogeneannten „Stop-Words“, also eine Menge von Verbindungswörtern entfernt werden. Ein anderer Schritt der Datenreinigung ist das Transponieren aller Wörter in kleine Buchstaben, um eine Einheitlichkeit zu erlangen, da das Bag of Words Modell keine Reihenfolge mehr beachtet und somit große Satzanfänge irrelevant werden. Wenn die Daten in der gewünschten Form vorliegen, beginnt der Schritt des Featureengineerings. Für einen Computer sind Wörter nicht so leicht zu verarbeiten, wie Zahlen, weshalb in diesem Schritt eine Quantisierung der Wörter und Überführung dieser in eine zahlenbasierte Form vorgenommen wird. Dies kann zum Beispiel in Form eines Bag-of-Words Modells, Dictionary oder TF-IDF, also einer relativen Vorkommensauflistung verschiedner Wörter über Dokumente umgesetzt werden. Nachdem die Daten in eine für den Computer kompatiblen Form gebracht wurden, kann das Themenmodell entwickelt werden.

2.7 Ansätze

In diesem Abschnitt werden verschiedene Möglichkeiten implementiert und analysiert, wie LDA im Sinne der ZVO genutzt werden kann. Dabei ist die Zielfrage, wie am besten für ein unbekanntes Dokument die bestimmte Abteilung gefunden werden kann. Dafür müssen Korpora bestehen, die bereits durch Verteilung definiert sind, sodass die

Dokument-Themen Verteilung für das neues Dokument inferiert werden kann. Wenn die Themenverteilung des neuen Dokuments gegeben ist, kann über Vergleiche der Verteilungen mit Dokumenten oder Durschnitten von Korpora eine Abteilung für das Dokument identifiziert werden.

Die Möglichkeiten, die sich für diesen Zweck ergeben sind folgende:

1. **Ein großer Korpus**
   Alle Dokumente ergeben ein Korpus. Das neue Dokument wird mit dem gesamten Korpus verglichen und auf thematische Kompatibilität geprüft. Ein Gesamtbild der thematischen Aufteilung aller Dokumente kann eine mögliche effektivere Neuverteilung der Kategorien schlussfolgern.
2. **18 Korpora mit Durchschnitt**
   Jede Abteilung stellt einen Korpus da, deren Verteilung mit der des neuen Dokuments verglichen wird. In diesem Fall hat jede der 18 Abteilungen eine Dokument-Themen Verteilung, die die Abteilung inhaltlich von den anderen unterscheidet. Dazu wird ein neues Dokument in jedem einzelnen Korpus integriert, um eine Dokument-Themen Verteilung für das neue Dokument auf Basis der korpusabhängigen Wörter-Themen Verteilung zu inferieren.
3. **18 Korpora mit Mehrheitsentscheid**
   Jede Abteilung stellt einen Korpus dar, deren Dokumente alle einzeln mit der Verteilung des neuen Dokuments verglichen werden, das dann quantitativ einer Abteilung zugeordnet wird. Die Wörter-Themen Verteilung ist für alle dokumente eines Korpus gleich, während für jedes Dokument eine Dokument-Themen Verteilung errechnet wird. Wenn die Dokument-Themen Verteilung der bestehenden Dokumente und die des neuen Dokuments bereitstehen, können diese auf Ähnlichkeit überprüft werden. Da bereits bekannt ist, welcher Abteilung jedes Dokument angehört, stellen die Top X ähnlichsten Dokumente eine Verteilung der Abteilungen dar, denen das neue Dokument inhaltlich am ähnlichsten ist.

Möglichkeit 1: Ein großer Korpus Groppe

Ziel dieser Untersuchung ist, der Vergleich der Themen, die sich durch ein LDA Modell ergeben mit den händisch eingeteilten Themen. Dafür wird ein Korpus auf allen Dokumenten generiert, der eine Themenverteilung erzeugt. Um herauszufinden, welches Dokument, zu welchem Thema gehört, werden die Themenverteilungen der einzelnen Dokumente inferiert. Die durch LDA erzeugte Themen werden mit den händisch geordneten Themen verglichen, indem Überschneidungen der enthaltenen Dokumente quantifiziert werden. Da LDA die Themen nicht benennen kann ,sondern nur die Verteilung der Wörter auflistet, müssen die von LDA gefundenen Gruppen noch den händischen zugeordnet werden. Die Zuordnung, die in Summe die meisten Übereinstimmungen zwischen den Gruppen ergibt, stellt die realistischste Interpretation der LDA Themen dar. Das Ergebnis kann Aufschluss über die vorgegeben Themen geben. Bei einem perfekten Modell und optimaler Themeneinteilung müssten die Themen mit einer sehr hohen Überschneidungsrate eindeutig einteilbar sein.

Dabei werden zuerst alle Anfragedaten in einen String zusammengefügt, der als Grundlage für das Wörterbuch und den Korpus dient. Durch den Aufruf des LDA Modells wird der String in eine vorgegebene Anzahl an Themen eingeteilt, basierend auf häufig zusammen auftretenden Wörtern des Wörterbuchs.

Die Überschneidungen sind in einer Matrix aufgetragen, die für jedes Thema aus dem LDA Modell die Anzahl der Überschneidungen mit jedem manuell erzeugten Thema speichert. Mit einer Max-Funktion wird das maximale Element aus jeder Zeile gefunden und dessen Index in einer Liste gespeichert. Dieser Index repräsentiert die Abteilung, mit der das jeweilige Thema des LDA Modells am wahrscheinlichsten zugeordnet wäre. Der Liste lässt sich folgend interpretieren:

```
maxmatrix = [9, 10, 0, 9, 10, 10, 0, 10, 10, 0, 1, 2, 1, 9, 10, 9, 10,
    9]
```

Eine optimale Zuordnung ist erreicht, wenn alle Zahlen von Eins bis 18 ohne Duplikate in der Liste in einer beliebigen Reihenfolge vorkommen. Das würde bedeuten, dass einem Thema von LDA genau ein Thema der ZVO am wahrscheinlichsten zugeordnet ist. In der Liste wird jedoch deutlich, dass viele Themen des LDA Modells mit dem 9. und 10. Thema der ZVO kompatibel wären. Somit ist zumindest keine optimale Zuordnung des Modellergebnisses möglich, was entweder auf die Aufteilung der Themen oder das Modell zurückzuführen ist.

Möglichkeit 2 : 18 Korpora mit Durchschnitt Felix

Als weitere Möglichkeit wird aus jeder der 18 vorgegebenen Abteilungen ein eigener Korpus generiert. Somit verfügt jeder Korpus über eine Dokument-Topic-Verteilung, die die Vorkommenswahrscheinlichkeit der 18 Themen in dem jeweiligen Korpus repräsentiert. Ziel der Untersuchung ist die Klassifikationsfähigkeit eines unbekannten Dokuments. Da jeder Korpus aus verschiedenen Dokumenten besteht, gibt es in 18 Korpora auch 18 verschiedene Topic-Word-Verteilungen. Pro Korpus wird für das neue Dokument die Dokument-Topic-Verteilung auf Basis der individuellen Topic-Word-Verteilung generiert. Dafür ist die Dokument-Topic-Verteilung der einzelnen Korpora relevant, die sich aus dem Durchschnitt aller Dokument-Topic-Verteilung der Dokumente eines jeweiligen Korpus ergibt. Aus dem Vergleich der Dokument-Topic-Verteilung des neuen Dokuments mit der des Korpus ergibt sich, wie gut das Dokument zu der Abteilung passt. Dieser Vergleich kann z.B. durch die Berechnung des Hellinger-Abstands der Verteilungen durchgeführt werden. Ist das Ergebnis klein, sind die Verteilungen ähnlich zu einander. Das neue Dokument wird dem Korpus zugeteilt, dessen Abstandsberechnung das minimale Ergebnis zeigt. Um zu analysieren, wie gut das Modell unbekannte Dokumente zuweist, kann nun die berechnetet Abteilung mit dem vorgelabelten Ergebnis verglichen werden. Bei einem Dokument ist das Ergebnis entweder richtig oder falsch, führt man den Vorgang jedoch mit mehreren Dokumenten aus, kann eine Quote errechnet werden. Diese sagt aus, wie gut die Dokumente klassifiziert wurden.

Das Ergebnis kann sich durch Anpassung der Parameter im Modell verändern. Diese Änderungen werden wir untersuchen.

Aktuelles Problem: Durschnittsberechnung (hardgecoded) dauert für alle mehrere Tage

Möglichkeit 3: 18 Korpora mit Mehrheitsentscheid Felix2

Die Unterscheidung dieser Methode zu Möglichkeit 2 ist, dass sich die Topic-Wort-Verteilung eines Korpus nicht aus dem Durchschnitt aller ergibt, sondern das unbekannte Dokument mit allen Verteilungen einzeln verglichen wird. Die Ähnlichkeit wird dann durch die Hellinger-Distanz bestimmt. Dabei berechnet man die Distanz zwischen der Verteilung des neuen Dokuments und der jeden Dokuments enthalten im Korpus. Der Durchschnitt beschreibt die Ähnlichkeit zu dem Korpus und ergibt sich aus der Summe aller Distanzen geteilt durch die Anzahl der Vergleiche. Bei 18 Korpora wird das neue Dokument dem zugeordnet, der die höchste Ähnlichkeit errechnet hat. Als Code wird dies folgendermaßen abgebildet:

```
sim = 0

new_doc = df.at[unbekannt_docs_abt0[0], 'subject_message']
bow = corpora.Dictionary([abt0]).doc2bow(new_doc.split())
t_new_doc = lda0.get_document_topics(bow)

for a in range(18):
    for n in range(len(globals()['index_abt%s' % a])):
        doc = df.at[(globals()['index_abt%s' %
            a][n]),'subject_message']
        bow = corpora.Dictionary([(globals()['abt%s' %
            a])]).doc2bow(doc.split())
        t = globals()['lda%s' % a].get_document_topics(bow)
        sim += hellinger(t_new_doc,t)
    sim = sim/len((globals()['index_abt%s' % a]))
```

Für das in diesem Algorithmus genutztes unbekanntes Dokument wurde diese Ähnlichkeitsliste errechnet:

```
sim = [0.18631196155691188, 0.9805342692097637, 0.9596672538794954,
    0.9853267249062526, 0.9807896917303207, 0.9872277594688493,
    0.9778663668837273,0.9733184898947137, 0.9816813043505935,
    0.9712383191132694, 0.9609198366954252, 1.01269095883496,
    0.9789156307013527, 0.947041755787613, 0.9751426522996282,
    0.9768546403218383, 0.9745868410322677, 0.9810759260918432]
```

Diese Liste enthält die Stärke der Ähnlichkeit des neuen Dokuments zu den 18 Korpora. Die Hellinger Distanz ist klein, wenn die Argumente ähnlich sind und groß, wenn sie verschieden sind. In der Liste ist der erste Eintrag mit Abstand am geringsten, weshalb das Dokument dem ersten Korpus, also der Abteilung 0 zugeordnet werden würde. Im Algorithmus sieht man, dass das neue Dokument tatsächlich aus Abteilung 0 gewählt wurde. Somit war die Zuordnung erfolgreich.

Um eine aussagekräftigere Quote zu erhalten wird dieser Algorithmus für ein unbekanntes Dokument jeden Korpus durchgeführt. Die jeweiligen Zuordnungen werden in einer Liste gespeichert. Der Index besagt aus welchem Korpus ein Dokument entnommen wurde und der Wert, den Korpus, dem es zugeordnet wurde. Das Ergebnis zeigt folgendes:

```
assignment = [0, 14,  2,  8, 4, 5]
```

Eine perfekte Zuteilung wäre erreicht, wenn die Liste mit den Zahlen 0 bis 17 in aufsteigender Reihenfolge gefüllt wäre. Die Quote von richtig zugeordneten Dokumenten liegt somit bei X

Alternativ zu Möglichkeit 1 (Erster Versuch)

Das Ziel ist die Generierung einer Themenverteilung über alle Dokumente im Korpus. Diese könnte Aufschluss über die generelle Kategorien der Abteilungen geben. Dafür müssen zuerst alle Daten in einen großen Korpus vereinigt werden, das ist in der Methode „returnAll()“ implementiert. Danach wird diese in eine Liste aufgeteilt und in ein Dictionary verwandelt. Diese wird dann genutzt um mit der „gensim.models.LdaMulticore()“-Methode in einen LDA Modell verwandelt wird. Für dieses Modell wurde die Themenanzahl auf 18 gesetzt, da dies die aktuelle Anzahl der Themen bei dem ZVO ist.

```
In [11]: #Zusammenfügen aller Anfragen in eine große Datenmenge
         def returnAll(data):
             collected=''
             print('Step 1: started merging')
             for x in range(0,133044):
                 collected+= data.at[x,'subject-message']
             print('Step 2: finished merging')
             return collected


         #Zusammengefügte Daten in eine Liste splitten
         splitData = returnAll(df).split()
         print('Step 3 finished splitting')

         # Testausgabe nach Split
             #for x in range(1,5):
             #    print([SplitData[x]])

         # Dictionary für Topic-Wortverteilung

         wordIDs = corpora.Dictionary([splitData])
         print('Step 4: dictionary created')

         # Definition Korpus
         var = [splitData]
         corpus = [wordIDs.doc2bow(text) for text in var]
             #print(corpus[:1][0][:30])

         #LDA MODEL mit Themen-Wortverteilung pro Topic Wiedergabe

         num_top = 18

         lda_mod = gensim.models.LdaMulticore(corpus=corpus, id2word=wordIDs, num_topics=num_top, alpha=0.8)

         pprint(lda_mod.print_topics())
         doc_lda = lda_mod[corpus]

         print('Step 5 Lda Model generated')

         #Topic-Document-distribution
```

```
Step 1: started merging
Step 2: finished merging
Step 3 finished splitting
Step 4: dictionary created
[(0,
  '0.012*"zweckverband" + 0.010*"betreff" + 0.009*"danken" + '
  '0.008*"ostholstein" + 0.008*"sierksdorf" + 0.008*"nachricht" + 0.007*"and" '
  '+ 0.007*"email" + 0.007*"the" + 0.006*"gesine"'),
 (1,
  '0.012*"zweckverband" + 0.010*"ostholstein" + 0.010*"sierksdorf" + '
  '0.009*"nachricht" + 0.008*"betreff" + 0.008*"the" + 0.007*"danken" + '
  '0.007*"lübeck" + 0.006*"gmbh" + 0.006*"hyperlink"'),
 (2,
  '0.013*"ostholstein" + 0.012*"sierksdorf" + 0.011*"nachricht" + '
  '0.011*"zweckverband" + 0.009*"betreff" + 0.008*"lübeck" + 0.008*"danken" + '
  '0.008*"the" + 0.007*"frau" + 0.006*"homepage"'),
 (3,
  '0.012*"sierksdorf" + 0.011*"ostholstein" + 0.011*"zweckverband" + '
  '0.011*"nachricht" + 0.009*"the" + 0.007*"danken" + 0.007*"hyperlink" + '
  '0.006*"sitzen" + 0.006*"email" + 0.006*"körperschaft"'),
 (4,
  '0.013*"sierksdorf" + 0.013*"ostholstein" + 0.011*"danken" + '
  '0.011*"zweckverband" + 0.009*"frau" + 0.009*"the" + 0.009*"betreff" + '
  '0.008*"nachricht" + 0.006*"hra" + 0.006*"email"'),
 (5,
  '0.012*"zweckverband" + 0.010*"ostholstein" + 0.009*"the" + '
  '0.009*"sierksdorf" + 0.008*"email" + 0.008*"betreff" + 0.008*"danken" + '
  '0.007*"nachricht" + 0.007*"sitzen" + 0.006*"hyperlink"'),
 (6,
  '0.013*"sierksdorf" + 0.012*"zweckverband" + 0.011*"ostholstein" + '
  '0.010*"danken" + 0.007*"nachricht" + 0.007*"frau" + 0.007*"the" + '
  '0.007*"email" + 0.006*"betreff" + 0.006*"and"'),
 (7,
  '0.011*"nachricht" + 0.010*"zweckverband" + 0.010*"ostholstein" + '
  '0.008*"betreff" + 0.007*"danken" + 0.007*"email" + 0.007*"the" + '
  '0.007*"lübeck" + 0.007*"sierksdorf" + 0.007*"frau"'),
 (8,
  '0.012*"ostholstein" + 0.010*"sierksdorf" + 0.009*"zweckverband" + '
  '0.009*"the" + 0.007*"betreff" + 0.007*"frau" + 0.006*"danken" + '
  '0.006*"nachricht" + 0.006*"email" + 0.006*"datum"'),
 (9,
  '0.013*"ostholstein" + 0.011*"sierksdorf" + 0.010*"zweckverband" + '
  '0.009*"nachricht" + 0.008*"the" + 0.008*"betreff" + 0.007*"frau" + '
  '0.007*"danken" + 0.006*"hyperlink" + 0.006*"wagrienring"'),
 (10,
  '0.011*"zweckverband" + 0.009*"ostholstein" + 0.009*"the" + '
  '0.009*"sierksdorf" + 0.008*"nachricht" + 0.008*"danken" + 0.008*"betreff" + '
  '0.008*"email" + 0.007*"frau" + 0.006*"lübeck"'),
 (11,
  '0.012*"sierksdorf" + 0.011*"ostholstein" + 0.010*"the" + '
  '0.008*"zweckverband" + 0.008*"email" + 0.007*"nachricht" + 0.007*"danken" + '
  '0.007*"and" + 0.006*"lübeck" + 0.006*"frau"'),
 (12,
  '0.012*"sierksdorf" + 0.010*"the" + 0.010*"zweckverband" + 0.010*"nachricht" '
  '+ 0.009*"ostholstein" + 0.009*"betreff" + 0.008*"danken" + 0.007*"email" + '
  '0.007*"information" + 0.006*"lübeck"'),
 (13,
  '0.011*"zweckverband" + 0.011*"ostholstein" + 0.011*"nachricht" + '
  '0.009*"the" + 0.009*"sierksdorf" + 0.008*"betreff" + 0.007*"frau" + '
  '0.007*"lübeck" + 0.006*"and" + 0.006*"danken"'),
 (14,
  '0.011*"sierksdorf" + 0.011*"danken" + 0.010*"zweckverband" + '
  '0.010*"betreff" + 0.009*"ostholstein" + 0.008*"the" + 0.008*"nachricht" + '
  '0.007*"frau" + 0.007*"and" + 0.006*"lübeck"'),
 (15,
  '0.011*"ostholstein" + 0.011*"zweckverband" + 0.010*"the" + '
  '0.009*"sierksdorf" + 0.009*"nachricht" + 0.008*"danken" + 0.008*"betreff" + '
  '0.006*"frau" + 0.006*"lübeck" + 0.006*"email"'),
 (16,
  '0.010*"nachricht" + 0.010*"zweckverband" + 0.009*"sierksdorf" + '
  '0.009*"betreff" + 0.008*"the" + 0.008*"ostholstein" + 0.007*"frau" + '
  '0.007*"email" + 0.006*"danken" + 0.006*"homepage"'),
 (17,
  '0.015*"ostholstein" + 0.012*"sierksdorf" + 0.009*"zweckverband" + '
  '0.009*"the" + 0.007*"nachricht" + 0.007*"betreff" + 0.007*"and" + '
  '0.006*"email" + 0.006*"frau" + 0.006*"danken"')]
Step 5 Lda Model generated
```

Der Output zeigt eine Liste mit 18 Elementen. Jedes Element beschreibt die Wortverteilung in einem der 18 Themen. Dabei kann LDA das Thema nicht semantisch benennen, sondern nur Cluster an häufig zusammen vorkommenden Wörtern formen. Daher kommt auch der „Latent"-Teil in der Benennung. Neben der Themen-Wortverteilung ist die Dokument-Themenverteilung von Interessen, also wie häufig kommt jedes Topic in der Gesamtmenge aller Anfragedaten vor. Diese Information errechnet man sich, indem man die folgende Methode aufruft:

```
In [44]: all_topics = lda_mod.get_document_topics(corpus, per_word_topics=True)

         print(doc_topics)

         [(0, 0.018413449), (1, 0.06006566), (2, 0.06351237), (3, 0.014391888), (4, 0.11020222), (5, 0.12415801), (6, 0.030
         072724), (7, 0.014063389), (8, 0.013174435), (10, 0.15869582), (11, 0.032017518), (12, 0.027906504), (13, 0.114994
         764), (14, 0.051106744), (15, 0.042484146), (16, 0.09394227), (17, 0.02679523)]
```

Aus dieser Ausgabe wird nun ersichtlich, wie wahrscheinlich es ist, dass ein bestimmtes Thema in den Daten vorkommt. Das 10. Thema ist mit 15,9% das am häufigsten Vorkommende Thema und damit sind die Wörter „zweckverband", „ostholstein" und „sierksdorf" die am stärksten vertretenen Wörter im Gesamtkorpus. Dieses Ergebnis überrascht nur bedingt, da dies der Name der Organisation ist und demensprechend häufig in Anfragen vorkommt.

Wenn man den gleichen Quelltext mit einer geringeren Themenanzahl aufruft, kommt man auf ein ähnliches Ergebnis, hier mit Beispiel 4:

```
In [32]: #Zusammenfügen aller Anfragen in eine große Datenmenge
         def returnAll(data):
             collected2=''
             print('Step 1: started merging')
             for x in range(0,133044):
                 collected2+= data.at[x,'subject-message']
             print('Step 2: finished merging')
             return collected2


         #Zusammengefügte Daten in eine Liste splitten
         splitData2 = returnAll(df).split()
         print('Step 3 finished splitting')

         # Testausgabe nach Split
             #for x in range(1,5):
             #    print([SplitData[x]])

         # Dictionary für Topic-Wortverteilung

         wordIDs2 = corpora.Dictionary([splitData2])
         print('Step 4: dictionary created')

         # Definition Korpus
         var2 = [splitData2]
         corpus2 = [wordIDs2.doc2bow(text) for text in var2]
             #print(corpus[:1][0][:30])

         #LDA MODEL mit Themen-Wortverteilung pro Topic Wiedergabe

         num_top2 = 4

         lda_mod2 = gensim.models.LdaMulticore(corpus=corpus, id2word=wordIDs, num_topics=num_top2, alpha=0.8)

         pprint(lda_mod2.print_topics())
         doc_lda2 = lda_mod2[corpus2]

         print('Step 5: Lda Model generated')

         # Dokument-Themenverteilung ausgeben
         bow2 = lda_mod.id2word.doc2bow(splitData) # convert to bag of words format first
         doc_topics2, word_topics2, phi_values2 = lda_mod2.get_document_topics(bow2, per_word_topics=True)

         word_topics

         all_topics2 = lda_mod2.get_document_topics(corpus2, per_word_topics=True)
         print(doc_topics2)
         print('Step 6: Dokument-Themenverteilung finished')

         Step 1: started merging
         Step 2: finished merging
         Step 3 finished splitting
         Step 4: dictionary created
         [(0,
           '0.011*"nachricht" + 0.010*"ostholstein" + 0.009*"danken" + '
           '0.009*"sierksdorf" + 0.008*"zweckverband" + 0.007*"the" + 0.007*"betreff" + '
           '0.007*"frau" + 0.007*"this" + 0.006*"email"'),
          (1,
           '0.012*"sierksdorf" + 0.010*"the" + 0.010*"ostholstein" + '
           '0.009*"zweckverband" + 0.008*"nachricht" + 0.008*"betreff" + 0.007*"danken" '
           '+ 0.006*"frau" + 0.006*"and" + 0.006*"strohmeyer"'),
          (2,
           '0.012*"ostholstein" + 0.012*"sierksdorf" + 0.012*"zweckverband" + '
           '0.009*"danken" + 0.008*"betreff" + 0.007*"nachricht" + 0.007*"frau" + '
           '0.007*"the" + 0.007*"email" + 0.006*"lübeck"'),
          (3,
           '0.012*"zweckverband" + 0.011*"ostholstein" + 0.010*"nachricht" + '
           '0.010*"sierksdorf" + 0.009*"the" + 0.008*"danken" + 0.007*"betreff" + '
           '0.007*"lübeck" + 0.006*"frau" + 0.006*"email"')]
         Step 5: Lda Model generated
         [(0, 0.16449031), (1, 0.19620994), (2, 0.3637842), (3, 0.27551556)]
         Step 6: Dokument-Themenverteilung finished
```

Wieder hat das meistvertrene Thema die Wörter „Zweckverband“, „sierksdorf“ und „ostholstein“ als häufigste vorkommende Elemente. Als Einflussfaktor dient die Dirichlet Variable Alpha, die beeinflusst, wie stark die Wahrscheinlichkeitsunterschiede zwischen den einzelnen Werten ist. Im vorherigen Beispiel war Alpha 0,8 von 1.0. Im folgenden ist es 0,2 von 1.0:

```
Step 1: started merging
Step 2: finished merging
Step 3 finished splitting
Step 4: dictionary created
[(0,
  '0.013*"ostholstein" + 0.013*"zweckverband" + 0.010*"sierksdorf" + '
  '0.009*"nachricht" + 0.009*"betreff" + 0.008*"frau" + 0.008*"danken" + '
  '0.007*"the" + 0.006*"lübeck" + 0.006*"homepage"'),
 (1,
  '0.012*"ostholstein" + 0.011*"sierksdorf" + 0.009*"zweckverband" + '
  '0.009*"the" + 0.009*"danken" + 0.008*"nachricht" + 0.007*"betreff" + '
  '0.007*"email" + 0.006*"and" + 0.006*"lübeck"'),
 (2,
  '0.012*"sierksdorf" + 0.011*"zweckverband" + 0.009*"the" + 0.009*"nachricht" '
  '+ 0.009*"ostholstein" + 0.008*"betreff" + 0.006*"danken" + 0.006*"frau" + '
  '0.006*"hyperlink" + 0.006*"email"'),
 (3,
  '0.011*"sierksdorf" + 0.010*"zweckverband" + 0.009*"ostholstein" + '
  '0.009*"nachricht" + 0.009*"danken" + 0.009*"the" + 0.008*"betreff" + '
  '0.007*"email" + 0.007*"frau" + 0.006*"lübeck"')]
Step 5: Lda Model generated
[(0, 0.31668642), (1, 0.29446924), (2, 0.1640696), (3, 0.22477475)]
Step 6: Dokument-Themenverteilung finished
```

Was bringt mir das?

Möglichkeit 4 ??

2.8 Auswertung

Quellcode

Perplexität

ZVO

3

Conclusion

...