Institut für Parallele und Verteilte Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 364

Erschließen von Freitextfeldern mittels Text Mining und die Qualität der gewonnenen Informationen

Marco Link

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr.-Ing. habil. Bernhard Mitschang
Betreuer/in:	M.Sc. Cornelia Kiefer
Beginn am:	18. August 2016
Beendet am:	10. Februar 2017
CR-Nummer:	H.3.3, I.2.7

Kurzfassung

Vermehrt fallen innerhalb von Firmen neben den einfach auszuwertenden strukturierten Daten, auch unstrukturierte Daten in Form von Freitexten an. In dieser Ausarbeitung werden Techniken zur Strukturierung von Freitexten sowie verwandte Arbeiten und Vor- und Nachteile der Nutzung von Freitexten vorgestellt. Der Fokus liegt auf der Repräsentation der Daten als Vektoren und der Filterung von Stoppwörtern. Außerdem wird ein Prototyp zum Clustern von Freitextfeldern vorgestellt und auf einen Datensatz der NHTSA angewendet. Durch die Anwendung des Prototyps auf den NHTSA-Datensatz wird geklärt, inwiefern dieser Informationen in den Freitextfelder enthält, die nicht in den strukturierten Daten enthalten sind. Und ob das Clustering zu vollständigeren Informationen, das heißt zur erhöhter Datenqualität führt. Die Beantwortung geschieht durch Datenanalysen auf den vom Prototyp erweiterten Datensatz. Eine zusätzliche Anwendung und Auswertung des Prototyps, findet auf einen Datensatz aus der Industrie statt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

Abkürzungsverzeichnis

Abkürzung	Bedeutung	Erstes Vorkommen
CSV	Comma-separated values	35
ICA	Incident Categorization & Analysis	31
NER	Named-Entity-Recognition	28
NHTSA	National Highway Traffic Safety Administration	6
NLP	Natural Language Processing	15
NLTK	Natural Language Toolkit	41
POS	Part-of-Speech	21
TAKMI	Text Analysis and Knowledge MIning	31
TF-IDF	Term Frequency-Inverse Document Frequency	23
WSD	word-sense disambiguation	21

1 Einleitung

Der Wandel hin zur Industrie 4.0 ist mittlerweile in aller Munde und lässt auf einen Innovationssprung hoffen. Durch zusätzliche Vernetzung und Kommunikation von Maschinen sowie das Ansammeln aller möglichen Informationen, beschäftigen sich viele Fragen mit dem Umgang solcher voluminösen Datensammlungen.

Die anfallenden Informationen können verschiedene Eigenschaften aufweisen und werden dementsprechend unterteilt. Die Darstellung und die semantische Interpretation von strukturierten Informationen werden durch die Angabe von Schemata eingegrenzt. Indes zeichnen sich unstrukturierte Informationen durch die Abwesenheit einer semantischen Vorgabe aus und können aus einer beliebigen Abfolge aus Zeichen bestehen. [1]

Texte, die in natürlicher Sprache, wie etwa Deutsch oder Englisch, verfasst worden sind, sind Beispiele für unstrukturierte Informationen. Es wird ein Überblick über Techniken zur Strukturierung von Freitexten gegeben sowie verwandte Arbeiten und Vor- und Nachteile der Nutzung von Freitextfeldern vorgestellt. Für eine Auswertung mittels Computer gilt es vorerst die Freitexte mittels Natural Language Processing (NLP) Methoden zu verarbeiten. Entlang einer Pipeline werden hierfür je nach Anwendung verschiedene Schritte zur Verarbeitung der Freitexte durchgeführt. Innerhalb dieser Ausarbeitung werden vorallem die in den strukturierten Daten enthaltenen Informationen aus den unstrukturierten Daten herausgefiltert, damit neue in den Freitextfeldern enthaltenen Informationen nicht verdeckt werden. Die vorverarbeiteten Freitexte werden mithilfe der strukturierten Daten gruppiert und in eine Vektorraumdarstellung überführt. Die eigentliche Auswertung geschieht mittels Text Mining Methoden, genauer Clustering. Es wird ein Prototyp vorgestellt, der speziell für die Vorverarbeitung und Clustering der Freitextfelder mit Bezug auf den strukturierten Daten entwickelt wurde. Der Prototyp wird auf zwei Datensätzen, einen von der NHTSA und einen aus der Industrie angewendet. Anschließende Datenanalysen geben Aufschluss ob der angewendete Prototyp zu einer Erhöhung der Datenqualität führt. [2]

1.1 Motivation

Durch insbesondere soziale Netzwerke kommt es zu einer regelrechten Flut an unstrukturierten Daten, die Informationen versprechen und die Firmen gerne ausgewertet wissen wollen, um bessere Entscheidungen treffen zu können. Eine weitere Quelle für unstrukturierte Informationen liefern Werksarbeiter oder auch Kunden in Form von Freitextfeldern. Die in den

Freitextfeldern angegebenen Informationen sollen die in den strukturierten Daten enthaltenen Informationen ergänzen. Durch die schiere Menge an Daten ist es jedoch kaum möglich solche Freitexte manuell von Menschen auswerten zu lassen. Sowohl aus zeitlichen, als auch aus Kostengründen wäre dies nicht profitabel. Automatische Analysen durch Computer sollen die in den Freitextfeldern enthalten Informationen extrahieren und zugänglich machen. [2]

1.2 Aufbau der Arbeit

Der Aufbau der Arbeit untergliedert sich wie folgt. Zunächst werden im Kapitel 2 die Grundlagen beschrieben, die beim Verständnis der weiteren Arbeit helfen. Im Kapitel 3 werden verwandte Arbeiten vorgestellt. Als Nächstes gilt es im Kapitel 4, die Forschungsfrage und das Untersuchungsziel zu klären. Der für diese Ausarbeitung entwickelte Prototyp wird in den folgenden Kapiteln vorgestellt. Im Kapitel 5 wird das zugrunde liegende Konzept erläutert. Kapitel 6 behandelt dann die Umsetzung des Konzepts und die Implementierung sowie die verwendeten Bibliotheken. Die Anwendung des Prototyps auf reale Datensätze wird im Kapitel 7 gezeigt. Im Kapitel 8 werden daraufhin auf den Ausgaben des Prototyps Datenanalysen vollzogen, um einen potenziellen Informationsgewinn durch die vom Prototyp hinzugefügten Informationen festzustellen. Zum Schluss werden in Kapitel 9 eine Zusammenfassung und ein Ausblick gegeben.

2 Grundlagen

Im Folgenden werden die Grundlagen für ein Verständnis der Ausarbeitung vorgestellt.

2.1 Daten- und Informationsqualität

Unter Qualität kann im allgemeinen die „Eignung für eine bestimmte Nutzung“ [1] verstanden werden. Dabei bezieht sich Datenqualität auf strukturierte Daten, also solche Daten, deren Struktur und semantische Interpretation bereits vorgegeben ist. Dies trifft beispielsweise auf relationale Datenbanken mit ihren Schemadefinitionen zu. Eine niedrige Datenqualität kann beispielsweise Geschäftsprozesse negativ beeinflussen und eine niedrige Schemaqualität kann zu Redundanzen und Unregelmäßigkeiten in Datenbanken führen. Es gibt sowohl Dimensionen für die Datenqualität, als auch für die Schemaqualität. [1]

Beispiele für Datenqualitätsdimensionen [1]:

- **Syntaktische Genauigkeit:** Sei D eine festgelegte Domäne mit den möglichen Elementen die verwendet werden können, so bezieht sich die syntaktische Genauigkeit auf die Nähe zu einem möglichen Element in der Domäne D. Sei also beispielsweise die Domäne D gegeben mit Fahrzeugherstellern. So wäre das Element v=*Toyoto* syntaktisch unkorrekt, da dieses Element nicht in der Domäne D enthalten ist. Die syntaktische Genauigkeit kann beispielsweise mittels Editierdistanz berechnet werden und in diesem Beispiel beträge die Editierdistanz zu dem syntaktisch korrekten Wert v'=*Toyota* eins.
- **Semantische Genauigkeit:** Die semantische Genauigkeit bezieht sich auf die Nähe zum richtigen Element aus der Domäne D. So wäre zwar in der relationalen Tabelle 2.1 Toyota in der Domäne D bezüglich Autos vorhanden, jedoch ist mit Bezug auf das Fahrzeugmodell *Passat*, dieses semantisch inkorrekt.
- **Vollständigkeit:** Unter Vollständigkeit kann allgemein verstanden werden, wie umfangreich Daten für eine spezifische Aufgabe sind. In Bezug auf relationalen Tabellen kann darunter auch der Grad an Übereinstimmung der Tabelle mit der realen Welt verstanden werden. Innerhalb dieser Ausarbeitung wird versucht, diese Dimension zu verbessern.
- **Konsistenz:** Konsistenz bezieht sich auf das Nichteinhalten von semantischen Regeln, die sich zum Beispiel auf eine relationale Tabelle beziehen.

ID	Fahrzeughersteller	Fahrzeugmodell
1	Toyota	Prius
2	Toyota	Passat

Tabelle 2.1: Beispiel - Relationale Tabelle mit Fahrzeugmodellen

Viele Informationen sind allerdings in unstrukturierten Daten enthalten, wie beispielsweise Bildern oder Büchern. Ohne zugrunde liegendes Datenmodell fällt die Verwendung durch einen Computer schwerer. Für die Qualitätsdimensionen bezüglich unstrukturierten Daten müssen verschiedene Datenkonsumenten beachtet werden. [3]

Bei der Bewertung der Qualität werden die Daten D mit verschiedenen idealen Daten verglichen [3]:

- **Interpretierbarkeit:** Sind die Daten so wie vom derzeitigen Konsumenten erwartet?
- **Relevanz:** Sind die Daten ideal für eine Aufgabe?
- **Genauigkeit:** Sind die Daten konform mit der realen Welt?

2.2 Text Mining

Das Ziel von Text Mining ist von ähnlicher Natur wie das des Data Minings. Im Grunde geht es um die Informationssuche und Informationsgewinnung aus Datenquellen, durch das Auffinden von interessanten Mustern. Allerdings sind die Datenquellen bei Text Mining Textsammlungen und dementsprechend sind die zu findenden Muster nicht in wohlstrukturierten Datenbanken zu finden, sondern in den unstrukturierten Texten. Dennoch gibt es viele Ähnlichkeiten bezüglich des Aufbaus zu Data Mining. [4]

Zu den Schritten, die getätigt werden bis zur Informationsgewinnung, gehören [4]:

- Anwendung von Routinen zur Vorverarbeitung auf den Datenquellen.
- Anwendung von Algorithmen zur Entdeckung von Mustern.
- Präsentieren der Ergebnisse, Visualisierung der Ergebnisse.

Der Hauptunterschied bezüglich Data Mining besteht bei den angewendeten Vorverarbeitungsschritten. Bei Text Mining ist es nämlich notwendig repräsentative Merkmale aus den natürlichsprachlichen Datenquellen zu erkennen und herauszufiltern und damit eine strukturierte Zwischenform aus den Texten zu erstellen. Für die Vorverarbeitungsschritte werden Techniken aus dem NLP verwendet. Hingegen der Verarbeitung von natürlicher Sprache durch Menschen, bei denen die Vorgänge simultan ablaufen, teilt der Computer sich die notwendigen Aufgaben in einfachere Einzelaufgaben, gemäß teile und herrsche, auf. [4]

Listing 2.1 Eine Beschwerde aus dem NHTSA-Datensatz.

```
WHEN REAR ENDED WITH FOOT ON BRAKE, DRIVERS SHOULDER BELT DIDN'T RESTRAIN FROM MAKING
    CONTACT, RESULTING IN AN INJURY. ESTIMATED SPEED OF IMPACT 30 MPH.
```

2.3 Natural Language Processing

Durch das Internet und geschäftsinterne Intranet stehen mittlerweile Informationen hauptsächlich in Form von unstrukturierten Daten, wie Freitexten, bereit, aus denen Informationen gewonnen werden können. Dies kann mittels NLP erreicht werden. Unter NLP ist die Analyse oder das Synthetisieren von natürlichen Sprachen in schriftlicher oder gesprochener Form zu verstehen. Zu unterscheiden sind die natürlichen Sprachen, die von den Menschen zur Kommunikation untereinander verwendet werden, von den formalen Sprachen, beispielsweise den Programmiersprachen, die für Computer verständlicher sind. Problematisch bei der Verarbeitung der natürlichen Sprachen sind die Ambiguitäten, die in diesen innewohnen. Aufgrund der Komplexität bei der Verarbeitung von natürlichen Sprachen wird der Vorgang in einzelne Teilvorgänge pipelineartig heruntergebrochen. Je nach konkreter Aufgabenstellung einer NLP-Applikation setzt sich die NLP-Pipeline zusammen. [5]

Abbildung 2.1 stellt eine beispielhafte NLP-Pipeline dar. Als Eingabe bekommt die Pipeline ein Freitextfeld aus dem NHTSA-Datensatz (siehe Abschnitt 7.1.1), dieses wird in seine Bestandteile aufgeteilt durch die Tokenisation. Weiterhin werden Punktuationen entfernt sowie alle Buchstaben zu Kleinbuchstaben gewandelt. Außerdem werden nicht aussagekräftige Wörter, sogenannte Stoppwörter, entfernt. Die einzelnen Wörter werden im letzten Schritt noch auf ihr Wortstamm reduziert. Die genaue Beschreibung der erwähnten Vorverarbeitungsschritte, erfolgt im Folgenden.

2.4 Tokenisierung

Der erste Schritt von vielen bei der Verarbeitung von natürlicher Sprache, ist die Segmentierung von Wörtern oder Sätzen in Tokens. Intuitiv kann die Wortsegmentierung durch Erkennen von Leerzeichen erreicht werden, da zum Beispiel in Englisch regulär damit die Wörter voneinander getrennt werden. Jedoch ist es nicht ausreichend, da angewendet auf den Beispieltext in Listing 2.1 Tokens gebildet werden, die neben dem eigentlichen Wort auch noch Satzzeichen enthalten können, wie beispielsweise **'BRAKE,'** oder **'INJURY.'**. Die nächste Verfeinerung wäre das Beachten von Satzzeichen als Wortgrenzen, jedoch gibt es auch hierfür Ausnahmen wie **'B.Sc.'** oder Gleitkommazahlen, die beachtet werden müssen. Weiterhin möglich, wäre das Ersetzen von Wörtern, die durch Apostrophe verkürzt sind, beispielsweise **'DIDN'T'**. Allerdings müssen auch hierfür wieder Ausnahmen beachtet werden. Es wäre auch möglich, dass der Tokenisierer Komposita, wie **'SHOULDER BELT'**, als ein Wort erkennt, hierfür wäre allerdings ein Wörterbuch nötig. Die Aufteilung der Sätze geschieht zumeist an den Satzzeichen, da vor allem Frage- und Ausrufzeichen auf das Ende eines Satzes hinweisen.

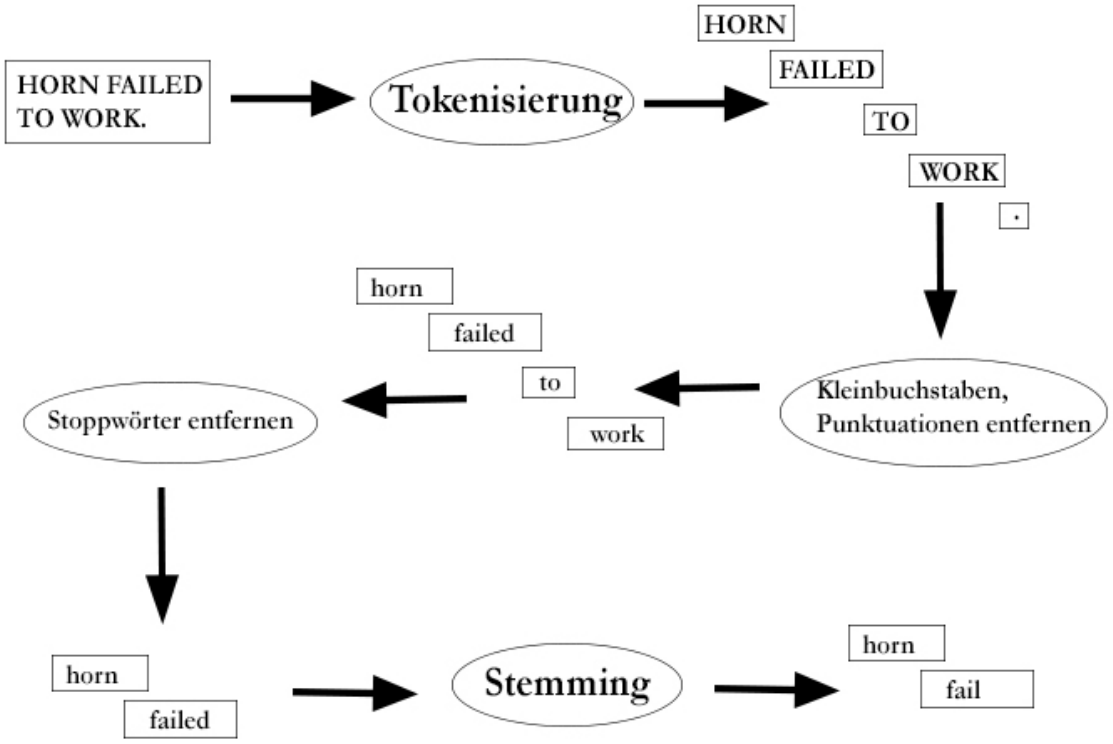

Abbildung 2.1: Beispielhafte NLP-Pipeline

Allerdings muss besonders auf die Mehrdeutigkeit des Satzzeichens Punkt geachtet werden, da beispielsweise Abkürzungen häufig einen Punkt enthalten und dies somit kein Satzende ist. Viele Implementierungen von Tokenisierern basieren auf der Anwendungen von Regeln, die mithilfe von regulären Ausdrücken umgesetzt werden. [6]

2.5 Rechtschreibkorrektur

Die ohnehin anspruchsvolle Verarbeitung von natürlichsprachlichen Texten wird durch Rechtschreibfehler erhöht. So können Rechtschreibfehler während Text Mining Vorgängen zu Missinterpretationen führen und beispielsweise beim Clustering zu falschen Gruppierungen. Für die Rechtschreibüberprüfung wird häufig ein Wörterbuch eingesetzt, um falsch geschriebene Wörter zu erkennen. [7]

Unterschieden wird zwischen [7]:

- „non-word errors“, dies sind tatsächlich falsch geschriebene Wörter, die folglich nicht im Wörterbuch enthalten sind und als Fehler markiert werden. Beispielsweise „Ich fahre ein *Aubo*“ anstelle von *Auto*.
- „real-word errors“, dies sind falsch geschriebene Wörter, die aber trotzdem ein valides Wort ergeben. Beispielsweise „Ich fahre ein *Autor*“ anstelle von *Auto*.

Werkzeuge für die Erkennung von Rechtschreibfehlern nutzen häufig Part-of-Speech (POS) Tagging, siehe Abschnitt 2.7, und Editierdistanzen für das Auffinden von Rechtschreibfehlern und für Korrekturvorschläge. Für die Erkennung von „real-word errors“ werden typischerweise N-Gramm basierte Techniken sowie statistisches Lernen mithilfe von Trainingsdaten verwendet. [7]

2.6 Stoppwort

Es gibt Wörter, die in Texten häufig Verwendung finden, jeodch zu dem eigentlichen Informationsgehalt eines Textes nichts beitragen. Diese Wörter haben ausschließlich eine grammatikalische Funktion und werden im Gegensatz zu Wörtern mit einer bedeutungstragenden Funktion für Text Mining Methoden nicht benötigt. Solche Wörter werden auch Stoppwörter genannt und werden in einer sogenannten Stoppwortliste gesammelt. Das Wort **'the'** wäre ein Beispiel für ein solches Stoppwort. [6]

2.7 Part-of-Speech Tagging

Der nächste wichtige Schritt bei der Verarbeitung von natürliche Sprache, ist das Annotieren der Wortarten der zuvor identifizierten Tokens. Die Wortart liefert Informationen über ein Wort und dessen benachbarte Wörter. Zum Beispiel kann dadurch word-sense disambiguation (WSD) betrieben werden, also die Auflösung der Mehrdeutigkeiten eines Worts durch den Kontext. Zum Beispiel kann **Bank** je nach Kontext für für ein Geldinstitut oder für eine Sitzgelegenheit stehen. Für den Vorgang wird ein vordefiniertes Tagset verwendet. Dieses definiert alle möglichen Wortarten, die zugewiesen werden können. Das Penn Treebank Tagset[1] beispielsweise enthält 45 verschiedene Wortarten. [6] Dem Token **BRAKE** aus der Beispielbeschwerde 2.1 könnte das Tag *NN* für Substantiv zugewiesen werden. Da das Wort **BRAKE** jedoch auch ein Verb sein könnte, kann eine sinnvolle Entscheidung nur mithilfe des Kontextes getroffen werden.

[1] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/data/Penn-Treebank-Tagset.pdf

2.8 Stemming / Lemmatisierung

Sowohl Stemming als auch Lemmatisierung zielen darauf ab, ein Wort auf eine Normalform zu bringen. Bei Stemming ist die Normalform die Rückführung auf einen künstlichen Wortstamm. Insbesondere bei der Suche im Internet ist es von Vorteil davon Gebrauch zu machen, damit beispielsweise bei der Suche nach **Autos** auch die Singularform **Auto** mit einbezogen wird. Beim Stemming werden ausschließlich die Wortendungen eines Worts entfernt. Hierbei spielen beispielsweise die Wortart oder andere Informationen über dieses Wort keine Bedeutung. Im Gegensatz dazu muss bei der Lemmatisierung das Wort in einem Wörterbuch nachgeschlagen werden um das Wort auf seine tatsächliche Grundform, das Lemma, zu reduzieren. Beispielsweise ist **fahren** das gemeinsame Lemma von **fährt** und **fuhr** und es werden also beide Wörter auf das Lemma **fahren** abgebildet. [6]

2.9 Vektorraumdarstellung

Clustering-Algorithmen und andere Text Mining Methoden können nichts mit dem reinen unstrukturierten Text anfangen. Deshalb wird während der Vorverarbeitungsphase eine andere Darstellung der reinen Texte gewählt. Die Texte werden zu Merkmalsvektoren umgewandelt. Die einzelnen Merkmale können dann noch gewichtet werden. Eine gebräuchliche Form der Vektorraumdarstellung ist die Bag-of-Words Darstellung. Diese nutzt einfach alle Wörter in einem Text als Merkmale und damit beträgt die Größe des Vektorraums die Gesamtheit aller unterschiedlichen Wörter aus allen Texten. Die binäre Gewichtung ist wohl die einfachste Gewichtung für die Merkmale. Diese bildet die An- oder Abwesenheit eines Worts in einem Text ab. [7]

Für ein Beispiel werden folgende Freitexte verwendet:

1. `HORN FAILED`
2. `AIRBAG FAILED`
3. `AIRBAG AND HORN FAILED`
4. `AIRBAG AND HORN`

Die Tabelle 2.2 zeigt die Bag-of-Words Darstellung mit binärer Gewichtung. 1 drückt aus, dass das Wort in dem Text vorkommt und 0, dass das Wort nicht in dem Text vorkommt. Je nach angewendeten Vorverarbeitungsschritten, kann die endgültige Vektorraumdarstellung anders aussehen. In diesem Beispiel wurden die Satzzeichen und Stoppwörter entfernt sowie die Wörter auf den Wortstamm reduziert. In Abbildung 2.2 sind die Vektoren in einem Diagramm dargestellt.

HORN	AIRBAG	FAIL
1	0	1
0	1	1
1	1	1
1	1	0

Tabelle 2.2: Beispiel einer Bag-of-Words Darstellung

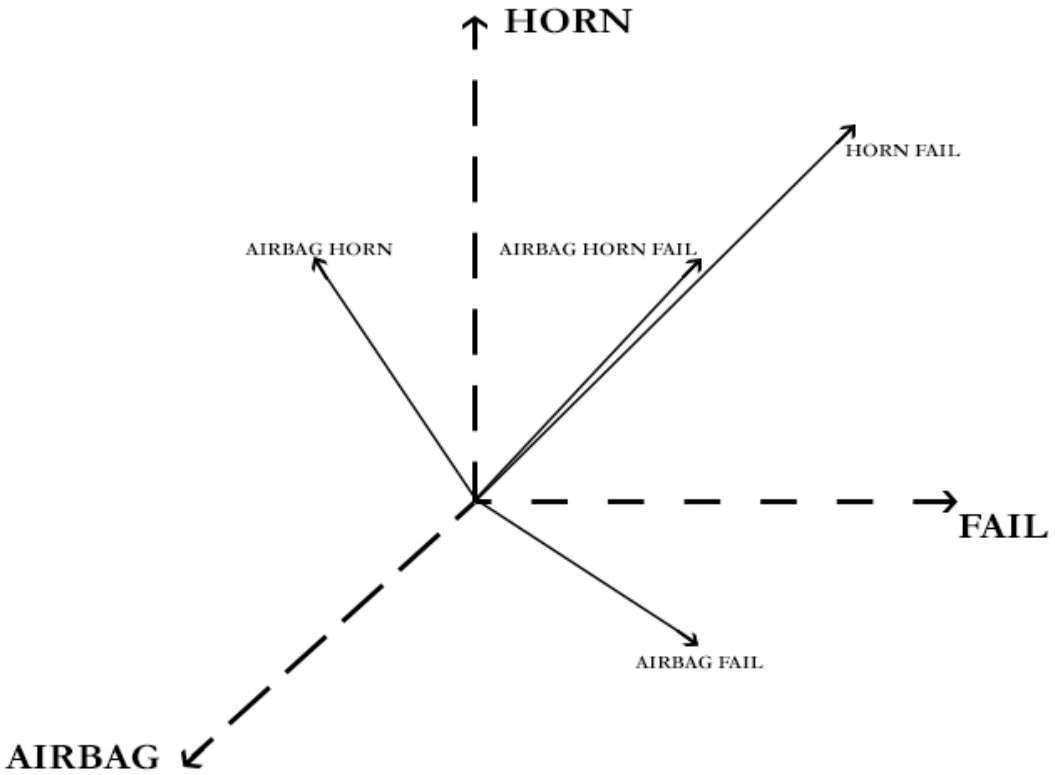

Abbildung 2.2: Beispielhafte Vektorraumdarstellung

2.9.1 Term Frequency-Inverse Document Frequency

Term Frequency-Inverse Document Frequency (TF-IDF) ist ein statistisches Verfahren für die Gewichtung der Merkmalsvektoren. Anders als bei der binären Gewichtung, repräsentieren die Gewichte, die mit TF-IDF erstellt sind die Wichtigkeit der Merkmale im Text und in der gesamten Textsammlung. Dabei weist TF-IDF Termen eine hohe Gewichtung zu, wenn diese häufig in einer kleinen Anzahl von Texten auftauchen. Bei TF-IDF wird das Merkmalsgewicht eines Texts durch zwei Faktoren bestimmt [8]:

- **Termhäufigkeit tf(i,j):** Wie oft taucht ein Term *j* im Text *i* auf.

- **Dokumentenhäufigkeit df(j):** Wie oft taucht ein Term *j* in der gesamten Textsammlung auf.

Die **inverse Dokumentenhäufigkeit idf(j)** ist eine Heuristik für die Annahme, dass Wörter die in vielen Texten innerhalb einer Textsammlung auftauchen, niedrig gewichtet werden sollten, da diese keine gute Unterscheidung der einzelnen Texte zulassen. [9]

Die Formel für die inverse Dokumentenhäufigkeit eines Terms j **idf(j)** ist wie folgt [8]:

$$idf(j) = \log(N/df(j)) \tag{2.1}$$

Dabei steht N für die Anzahl der Texte innerhalb einer Textsammlung und $df(j)$ für die Dokumentenhäufigkeit von j

Die Gewichtung eines Terms j eines Textes i ($\omega(i,j)$) mit TF-IDF ergibt sich aus der Termhäufigkeit $tf(i,j)$ und der inversen Dokumentenhäufigkeit $idf(j)$ wie folgt [8]:

$$\omega(i,j) = tf(i,j) * idf(j) = tf(i,j) * \log(N/df(j)) \tag{2.2}$$

2.10 Clustering

Clustering ist eine unüberwachte Lernmethode zur Gruppierung von Objekten. Das bedeutet, dass im Voraus keine zusätzlichen Informationen, außer den Objekten und deren enthaltenen Informationen bekannt sind und ähnliche Objekte zusammen in möglichst bedeutungsvolle Cluster gruppiert werden. Im Wesentlichen ist das Clustering ein Optimierungsproblem, da als Ziel, die möglichst besten Gruppierungen von Objekten gefunden werden sollen. Für das Herausfinden von guten Gruppierungen kommen oftmals Ähnlichkeitsfunktionen zum Einsatz. Diese geben durch einen reellen Wert an, wie ähnlich sich zwei Objekte sind und können dementsprechend bei starker Ähnlichkeit zusammen gruppiert werden. [4]

Zu den häufig verwendeten Ähnlichkeitsfunktionen zählen [4]:

- die euklidische Distanz
- die Kosinus-Ähnlichkeit, die am häufigsten beim Clustering für Textdokumente zum Einsatz kommt.

Zur Bestimmung der Ähnlichkeit der Objekte, werden deren Merkmale verwendet. Die Generierung der Merkmalsmengen der Objekte wird als „feature extraction“ bezeichnet und „feature selection“ bezeichnet das Herausfinden der besten Merkmale zur Ähnlichkeitsbestimmung. [4]

Clustering-Algorithmen können wie folgt eingeteilt werden [4]:

- Ein **partitionierender Algorithmus** teilt alle Objekte disjunkt in Gruppen ein.
- Ein **hierarchischer Algorithmus** erzeugt eine verschachtelte Abfolge von Partitionen.

Weiterhin können beide unterteilt werden in [4]:

- **hard clustering:** alle Objekte werden jeweils nur exakt einem Cluster zugeordnet
- **soft clustering:** alle Objekte können jedem Cluster zu einem gewissen Grad zugeteilt werden

Die verschiedene Clustering-Algorithmen unterschieden sich weiterhin durch ihre Strategien [4]:

- Bei **agglomerativen Algorithmen** wird jedes Objekt zu Beginn einem separaten Cluster zugeordnet und schrittweise die Cluster miteinander vereinigt bis ein Stoppkriterium erfüllt worden ist.
- Bei **divisiven Algorithmen** werden alle Objekte zu einem Cluster zugeteilt und solange aufgeteilt bis ein Stoppkriterium erfüllt worden ist.
- **„Shuffling"-Algorithmen** betreiben eine iterative Umverteilung der Objekte in Cluster.

2.10.1 k-means Algorithmus

Zu den meist verwendeten Algorithmen zählt der k-means Algorithmus. Dieser Algorithmus ist ein hart partitionierender Algorithmus. [4]

Sei M die Anzahl der zu clusternden Elemente und N die Anzahl an verschiedenen Variablen der einzelnen Objekte. Dann beschreibt $A(I, J)(1 \leq I \leq M, 1 \leq J \leq N)$ den Wert des I. Objekts der J. Variable. Eine Partition $P(M, K)$ besteht aus $1, 2, \ldots, K$ Clustern und jedes Objekt kann nur in einem Cluster enthalten sein. $B(L, J)$ gibt den Durchschnitt der J. Variable über den Objekten die in dem L. Cluster liegen an. $N(L)$ ist die Anzahl der Objekte in dem L. Cluster. [10]

Die Distanz zwischen dem I. Objekt und dem L. Cluster beträgt [10]:

$$D(I, L) = \sqrt{\sum_{1 \leq J \leq N} (A(I, J) - B(I, J))^2} \tag{2.3}$$

$L(I)$ gibt den Cluster an, der I enthält, $D(I, L(I))$ ist die euklidische Distanz zwischen dem I. Objekt und dem Cluster in dem I enthalten ist. Die Fehlerrate der Partition $P(M, K)$ beträgt [10]:

$$e[P(M, K)] = \sum_{1 \leq I \leq M} D(I, L(I))^2 \tag{2.4}$$

Der Algorithmus funktioniert wie folgt [10]:

1. Gegeben sind die anfänglichen Clustern $1, 2, \ldots, K$. Berechne die Durchschnitte aller Cluster $B(L, J)(1 \leq L \leq K, 1 \leq J \leq N)$ und die anfängliche Fehlerrate $e[P(M, K)](1 \leq I \leq M)$.

2. Als nächstes werden für das erste Objekt und für alle Cluster L die Änderungen der Fehlerraten berechnet, die bei dem Verschieben des ersten Objekts aus dem Cluster $L(1)$ in den Cluster L entstehen würde.

$$\frac{N(L) * D(1,L)^2}{N(L)+1} - \frac{N((L(1)) * D(1,L(1))^2}{N(L(1))-1} \tag{2.5}$$

   Im Falle eines negativen Ergebnisses würde die Fehlerrate durch das Verschieben verringert werden. Also wird jener Cluster L ausgewählt, der für den größten negativsten Wert bei der Änderung der Fehlerrate sorgt und damit auch die Fehlerrate am stärksten verringern wird. Das erste Objekt aus dem Cluster $L(1)$ wird dann in den Cluster L verschoben. Außerdem werden die Cluster-Durchschnitte der Cluster $L(1)$ und L angepasst und die negative Änderung der Fehlerrate wird zur Fehlerrate $e[P(M,K)]$ addiert.

3. Der zweite Schritt wird nun für alle restlichen Objekte $I(2 \leq I \leq M)$ wiederholt.

4. Stoppe, sobald das Verschieben irgendeines Objekts zu keiner besseren Fehlerrate mehr führt. Ansonsten fahre mit 2. fort.

2.11 Silhouette Koeffizient

Mit dieser Metrik kann festgestellt werden, wie gut ein Objekt *i* in seinem Cluster *A* liegt. Dazu ist lediglich eine gegebene Partitionierung und eine Menge mit allen Entfernungen zwischen Objekten nötig. Für jedes Objekt *i* kann daraus der Silhouette-Wert *s(i)* berechnet werden. Für diesen muss zuerst die durchschnittliche Entferungen des Objekts *i* zu allen anderen Objekten im Cluster *A* (*a(i)*) und die durchschnittliche Entfernung des Objekts *i* zu allen Objekten aus einem Cluster *C* (*d(i,C)*) errechnet werden. [11]

Des Weiteren muss *d(i,C)* für alle Cluster ungleich *A* berechnet werden, wobei der minimale Wert genommen wird [11]:

$$b(i) = minimum\ d(i,C) = d(i,B)\ C \neq A \tag{2.6}$$

d(i,B) wird als Nachbar des Objekts *i* bezeichnet und gilt als zweitbeste Wahl, falls Objekt *i* nicht dem Cluster *A* zugeordnet wäre.

Der Wert *s(i)* berechnet sich dann wie folgt [11]:

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \tag{2.7}$$

Das Ergebnis ist ein Wert von -1 bis +1. Dies kann interpretiert werden als undichtes Clustering, bei -1, bis hin zu einem sehr dichten Clustering, bei +1. Ein Wert von 0 kennzeichnet überlappende Cluster. [11]

3 Verwandte Arbeiten

Dieses Kapitel stellt verwandte Arbeiten sowie Vor- und Nachteile der Nutzung von Freitexten vor.

3.1 Text Mining auf Freitextfeldern

[12] verwendet Freitextfelder aus der Pillreports-Datenbank, um mögliche verfälschte Ecstasy Drogen zu bestimmen. Darunter zählen solche, die Bestandteile aufweisen, die nicht vergleichbar mit MDMA sind. Als strukturiertes Feld enthält die Datenbank eine potenzielle Warnung in Form von ja oder nein über eine Droge. Außerdem enthalten sind Beschreibungen und Nutzerberichte als Freitextfelder. Auf dieser Datenbank werden Text Mining Methoden, unter anderem zur Klassifizierung, angewendet. Um eine bestmögliche Klassifizierung zu erzielen, wurden verschiedene Vorgehensweisen durchgeführt und miteinander verglichen. Für die eigentliche Klassifizierung wurden zum einen Support Vector Machine als auch Naive-Bayes getestet. Auch die Vektorraumdarstellung der Beschreibungen wurde auf verschiedene Arten durchgeführt. Zu den allgemein verwendeten Vorverarbeitungsschritten zählen die Tokenisierung der Wörter, die Umwandlung in Kleinbuchstaben, das Zählen der Worthäufigkeit, die Aufteilung des Textes in N-gramme, das Entfernen von Stoppwörtern, das Anwenden des Porter Stemmers und das Entfernen der selten auftretenden Wörter.

Einige Publikationen beschäftigen sich mit dem Informationsgewinn aus den sozialen Netzwerken. Hierzu zählt vor allem der Microblogging-Dienst Twitter. [13] [14] [15] [16] wollen durch Twitter-Nachrichten Informationen über das Verkehrsgeschehen erhalten. Hierzu wird Tokenisierung, Herausfiltern von Stoppwörtern, Stemmen und Herausfiltern von nicht relevanten Wortstämmen als Vorverarbeitungsschritte sowie Support Vector Machine für die Klassifikation verwendet. [16] verfeinert den Prozess noch durch einen vorgeschlagenen Clustering-Algorithmus. Bei diesem wird eine neue Twitter-Nachricht mit den bereits in der Trainingsmenge enthaltenen Twitter-Nachrichten mittels euklidische Distanz verglichen. Gegebenenfalls wird die neue Twitter-Nachricht zu den normalen Nachrichten oder zu den verkehrsbezogenen Nachrichten gruppiert.

3.2 Clustering auf Freitextfeldern

[17] wollen Unregelmäßigkeiten in Raumfahrtsystemen aufdecken. Für die Analyse werden die Wartungsarbeitsberichte in Form von Freitexten verwendet. Weiterhin werden diese in eine Term-Dokument Matrix (Bag-of-Words) überführt. Bei der Vorverarbeitung werden Stoppwörter entfernt und TF-IDF (siehe Abschnitt 2.9.1) angewendet. Außerdem wird das Spectral Clustering Verfahren verwendet.

Weitere Anwendungen sind im Bereich des „opinion minings“ zu finden. In [18] werden beispielsweise Autobeurteilungen ausgewertet. Hierzu wird ein eigenes soft Clustering Verfahren angewendet, aber es wird auch ein kleiner Anteil der Autobeurteilungen annotiert und für das Trainieren eines Klassifikators zur Sentiment-Analyse verwendet. Für die Vorverarbeitung werden Stoppwörter entfernt sowie Porter Stemmer und TF-IDF angewendet.

In Australien wurden über Jahre hinweg Formulare von verwundeten Arbeitern ausgefüllt, die einen Schadensersatzanspruch geltend machen wollten. Diese beinhalten Informationen über die Verwundung und sollen mittels Text Mining und Clustering zur Prävention von Unfällen eingesetzt werden [19].

[20] verwendet ebenso Twitter-Nachrichten und stellt ein Framework zur Erkennung von disruptiven Ereignissen vor. Beispiele für disruptive Ereignisse sind terroristische Angriffe oder allgemein Verbrechen. Zu den Vorverarbeitungsschritten bei den Twitter-Nachrichten gehört unter anderem das Entfernen von Stoppwörtern und das Stemmen der Wörter. Weiterhin werden die irrelevanten Twitter-Nachrichten mittels eines Naive-Bayes-Klassifikators klassifiziert, so dass am Ende nur die Nachrichten mit Bezug auf Ereignissen übrig bleiben. Auf den aussortierten Nachrichten wird ein Online-Clustering-Algorithmus angewendet, um die Thematik eines Ereignis zu identifizieren. Auch Named-Entity-Recognition (NER) kommt zum Einsatz, um beispielsweise Orte und Straßennamen herauszufinden.

[21] wendet einen dichtebasierten online Clustering-Algorithmus (IncrementalDBSCAN) an, um für Ereignisse räumliche und zeitliche Informationen zu erhalten. Die Twitter-Nachrichten werden in die Bag-of-Words Darstellung zerlegt.

[12] verwendet die Nutzerberichte aus der Pillreports-Datenbank für ein Clusteringverfahren, um mögliche Muster bezüglich den Erfahrungen der Nutzer zu erkennen. Hierfür wird der k-means Clustering-Algorithmus mit der Kosinus-Ähnlichkeit angewendet. Für die Vektorraumdarstellung werden Stoppwörter, Satzzeichen und Nummern entfernt sowie Stemming angewandt. Eine Gewichtung findet mit TF-IDF statt.

3.3 Integration von strukturierten Daten und Freitexten

Der NHTSA-Datensatz (siehe Abschnitt 7.1.1) beinhaltet eine Ansammlung von sicherheitsrelevanten Defektbeschwerden von Autos. Dieser Datensatz stellt sowohl strukturierte Informa-

tionen wie das Automodell, Baujahr und die Anzahl der Verletzten beziehungsweise Toten, als auch eine Darstellung des Zwischenfalls samt Konsequenzen aus Sicht des Konsumenten in Freitextform bereit. In [22] kommt der NHTSA-Datensatz zum Einsatz um Informationen über schwerwiegende Unfälle zu gewinnen, in denen Menschen verletzt oder gestorben sind. Für die eigentliche Analyse wird ausschließlich das Freitextfeld mit den Beschwerdetexten verwendet. Die strukturierten Felder, die die Anzahl der Verletzten und der Toten beinhalten, werden verwendet um den Datensatz zu filtern. Zu den Vorverarbeitungsschritten gehört das Umwandeln der Freitexte in eine Korpusdatenstruktur, Umwandeln der Wörter in Kleinbuchstaben, Entfernen von Satzzeichen, Entfernen von generischen und domänenspezifischen Stoppwörter und Stemmen. Weiterhin wird mittels Latent Semantic Analysis die Dimensionalität der Term-Dokument Matrix reduziert. Für das Clustering wird ein hierarchischer Clustering-Algorithmus verwendet. Wie innerhalb dieser Ausarbeitung wird der NHTSA-Datensatz automatisch mittels Clustering analysiert. Auch wird ein strukturiertes Feld für eine Gruppierung des Datensatzes verwendet. Allerdings dient dies in [22] lediglich dazu, einen kleinen Ausschnitt, nämlich die fatalen Beschwerden, für die Analyse auszuwählen. Auch werden die in den strukturierten Daten enthaltenen Informationen für die Clusteranalyse ignoriert. Dadurch werden Clusterzentren generiert, dessen häufigsten Terme weitestgehend in den strukturierten Daten, hauptsächlich den Komponentenbeschreibungen, enthalten sind. Innerhalb dieser Ausarbeitung wird hingegen der gesamte Datensatz geclustert und mithilfe der Komponentenbeschreibungen gruppiert. Die Informationen aus den strukturierten Daten werden zu Stoppwortlisten hinzugefügt, um einen möglichen Mehrwert durch die Freitextfelder zu eruieren.

[23] verwenden ebenso die strukturierten Daten bezüglich spezifischer Komponentenbeschreibungen aus dem NHTSA-Datensatz und unstrukturierte Daten aus Honda, Toyota und Chevrolet spezifischen Diskussionsforen. Aus den Quellen der Diskussionforen werden Benutzername, Zeitpunkt der Veröffentlichung und der eigentliche Freitext extrahiert. Auf den Freitexten werden WSD, word categorization und Stemming angewendet. Ein Auszug aus den Daten wird von Domänenexperten manuell annotiert. Der Datensatz wird zuerst mit Komponentenbeschreibungen klassifiziert. Zu den hinzugefügten Informationen zählt, ob ein Defekt bereits diskutiert wird, wie kritisch der Defekt ist und welche Bauteile in Verbindung zum Defekt stehen. Diese 15 Komponentenbeschreibungen wurden aus den 339 in dem NHTSA-Datensatz zusammengefasst und enthalten zum Beispiel „Airbag“ oder „Braking“. Auf den Daten wurde weiterhin Sentiment-Analyse angewendet, um herauszufinden, ob eine häufige Anzahl an Wörtern, die eine negative Gefühlslage ausdrücken, auf Fahrzeugdefekte schließen lässt. Auch innerhalb dieser Ausarbeitung wird der NHTSA-Datensatz und deren Komponentenbeschreibungen sowie Freitexte verwendet. Während allerdings [23] Freitexte aus herstellerspezifischen Foren bezieht, wird innerhalb dieser Ausarbeitung die Beschwerdebeschreibungen aus dem NHTSA-Datensatz verwendet. Weiterhin werden in [23] die Komponentenbeschreibungen nur für die Auswahl von Freitexten aus den Diskussionforen und als Vorlage für die Klassenannotationen verwendet. Außerdem zielt [23] darauf ab zu zeigen, dass auf den Freitexten erfolgreich Sentiment-Analyse angewendet werden kann und damit die Freitexte in sicherheitsrelevante Defekte, Performanzdefekte und keine Defekte klassifizieren zu können. Hingegen wird in

dieser Ausarbeitung geklärt, inwiefern Freitextfelder Informationen enthalten, die nicht in den strukturierten Daten enthalten sind.

[24] nutzt die Daten eines Service Centers, um Informationen über die zu erwarteten Kosten verschiedener Service-Anfragen zu gewinnen. Es sollen Vorhersagen getroffen werden können, ob eine Service-Anfrage schnell erledigt werden kann oder länger dauern wird. Hierzu wird ein Klassifizierungsmodell basierend auf Naive-Bayes erstellt. Hierfür kommen neben den strukturierten Daten, wie die aufgewandte Zeit zur Behebung des Problems, auch die Fallbeschreibung in Form eines Freitextfelds zum Einsatz. Tokenisation, Aufteilung der Wörter in verschiedene Kategorien, Umwandlung in Kleinbuchstaben, Entfernen von Stoppwörtern, Rechtschreibkorrektur und Stemming werden bis zur endgültigen Vektorraumdarstellung der Freitexte angewendet. Vor der Klassifizierung werden die vorverarbeiteten Freitextfelder geclustert. Hierzu wird ein spezieller hierarchisch agglomerativer Clustering-Algorithmus verwendet (hMETIS partitioning technique).

In [25] werden Filmbewertungen in Form von Freitextfeldern genutzt. Weiterhin werden als strukturierte Daten auch das typische „star rating“ und das Genre des bewerteten Films verwendet. Aus diesen Informationen sollen die Meinungen zu den Filmen gewonnen werden und auch gezielt Filmvorschläge gegeben werden können. Dies geschieht bei einem vollautomatischen Ansatz durch die Verwendung von Latent-Dirichlet-Allocation für das Clustering.

[26] hilft Netzbetreibern die Auswertung ihrer Trouble-Ticket-Systeme zu ermöglichen, damit diese ihre Wartungsarbeiten besser koordinieren können. Dazu werden die strukturierten Informationen, wie etwa die Zeit wann ein Ticket erstellt wurde, sowie die unstrukturierten Daten, die die Informationen über das eigentliche Problem bereithalten eingesetzt. Für das Clustering wird ein agglomerativer hierarchischer Algorithmus verwendet.

In [27] wird ein Framework zur stetigen Verbesserung von Reparaturaufträgen und damit auch der Kundenzufriedenheit im Automobilbereich vorgestellt. Als strukturierte Daten stehen beispielsweise die aus einem Auto auslesbaren Fehlercodes, die zur Reparatur benötigte Arbeitszeit und die Information über ausgewechselte Ersatzteile bereit sowie vordefinierte „labor codes“, die zur Charakterisierung der durchgeführten Reparaturaktionen dienen. Als unstrukturierte Daten stehen die Reparaturbeschreibungen der Techniker zur Verfügung. Ein Teil des Frameworks versucht mittels „association rule mining“ Auffälligkeiten bei den begangenen Reparaturen als auch die Ursachen für diese herauszufinden. Außerdem soll von den am besten durchgeführten Reparaturvorgängen gelernt werden. Dies geschieht mittels „case-based-reasoning“ und Text Mining. Hierfür werden jene Reparaturbeschreibungen genutzt, die auch mit passenden „labor codes“ verknüpft sind. Zu den Text Mining Schritten gehören auch drei verschiedene hierarchische Clustering-Algorithmen. Clustering wird basierend auf den Bauteilen, den Bauteilen und den Symptomen sowie auf den Bauteilen, den Symptomen und den getätigten Reparaturaktionen durchgeführt. Die hierfür benötigten Informationen werden vorher durch andere Text Mining Komponenten sichergestellt.

[28] wertet Daten bezüglich Verkehrsunfällen in Queensland, Australien aus den Jahren 2004 bis 2005 aus. Strukturierte Daten wie die Wetterlage, als auch die Unfallbeschreibung als

unstrukturierte Daten kommen zum Einsatz. Zu den Vorverarbeitungsschritten der Unfallbeschreibungen gehört das Entfernen von Punktuationen, die Korrektur der dadurch entstanden inkorrekten Wörter, Auflösung von Abkürzungen, alle Großbuchstaben durch Kleinbuchstaben zu ersetzen, Rechtschreibfehler zu korrigieren, häufig vorkommende Phrasen zu einem Wort zusammenzufassen und das Entfernen von Stoppwörtern. Eingesetzt wurde das Leximancer tool[1], das auf der Bayes-Theorie basiert um Cluster zu bilden.

[29] haben einen Prototyp namens Text Analysis and Knowledge MIning (TAKMI) entwickelt, der auch auf große textuelle Datenbanken, als auch zusätzlich auf strukturierte Daten angewendet werden kann. Die Besonderheit bei TAKMI ist die spezielle Darstellung der Texte zur Verarbeitung, um den Informationsverlust durch eine Bag-of-Words Darstellung zu umgehen. Wörter und Phrasen werden deren semantische Eigenschaften angehängt, also ob es sich beispielsweise bei „Washington“ um eine Person oder ein Ort handelt. Eine weitere Annotation beinhaltet die Absicht. So kann „fail“ Teil einer Beschwerde, eines Lobes oder auch einer Frage sein. Zuletzt werden die Abhängigkeiten zwischen Wörtern und Phrasen in die Textrepräsentation eingebracht. Um dies zu realisieren kommt ein von Domänenexperten erstelltes semantisches Wörterbuch, ein POS Tagger, ein Chunker und heuristische Regeln zum Einsatz.

[30] stellt das Incident Categorization & Analysis (ICA) System vor. Dieses verwendet Anrufprotokolle in Form von unstrukturierten Freitextfeldern, die vom technischen Support erstellt wurden. Damit sollen Informationen über häufige Beschwerden gewonnen werden, um für diese sinnvolle Diagnosen, Unterstützung und Dokumentationen bereitstellen zu können. Die Anrufprotokolle werden nach bestimmten Produktlinien gefiltert. Dies geschieht mittels den strukturierten Daten, beispielsweise der Modellnummer. Auf den Freitextfeldern wird dann ein eigens entwickelter Clustering-Algorithmus angewendet, um potenzielle Kategorien zu generieren. Mithilfe der vorgeschlagenen Kategorien erstellt ein Domänenexperte tatsächliche Kategorien und weist ihnen Beispiele zu. Diese Informationen dienen dazu, einen Klassifikator für jede Kategorie zu erstellen.

3.4 Qualität / Nutzengewinn von Freitextfeldern

Es mangelt an Maßnahmen zur Sicherstellung der Qualität der unstrukturierten Daten, obwohl diese immer mehr in Entscheidungsprozessen Verwendung finden [3]. Um die Datenqualität von Texten beurteilen zu können, werden Dimensionen und ausführbare Indikatoren entlang der Datenanalyse-Pipeline in Erwägung gezogen.

Die vorgestellten Arbeiten erhofften sich allesamt einen Mehrwert durch die Einbeziehung der Freitextfelder und viele haben auch einen tatsächlichen Nutzengewinn dadurch erzielt. So haben [24] festgestellt, das einzig durch die Verwendung der strukturierten Daten, die Klassifizierung

[1] https://info.leximancer.com/

eine größere Tendenz hat, Service-Anfragen als schnell bearbeitbar zu klassifizieren. Diese Tendenz verringert sich durch die Hinzunahme der Freitextfelder, wodurch auch vermehrt Service-Anfragen als länger andauernd zu erledigen klassifiziert werden.

[25] haben festgestellt, dass die mittlere quadratische Abweichung bei allen Clusteringansätzen durch das Hinzunehmen der Filmbewertungen reduziert wird.

[22] konnte durch die Analyse des NHTSA-Datensatzes Cluster bilden, die zusätzlich mit den Datumsinformationen bezüglich des Zeitpunktes des Vorfalls und der Meldung, schwerwiegende Vorfälle auch mit der zeitlichen Entwicklung zeigen. Hierbei konnten insbesondere Zusammenhänge zwischen zwei großen Rückrufaktionen und eine hohe Anzahl an eingehenden Vorfallmeldungen innerhalb eines Zeitraums festgestellt werden.

[26] konnten durch die Verwendung der unstrukturierten Problembeschreibungen die Trouble-Tickets hierarchisch gruppieren. Dadurch konnte zum Beispiel in Fallstudien festgestellt werden, dass das Aktualisieren der Routersoftware zu den am häufigsten auftretenden Wartungsfällen gehört und das damit einhergehend, die Router-Hersteller möglichst mehr Anstrengungen in nahtlose Softwareaktualisierungen investieren sollten.

[28] hebt hervor, dass vielerlei Versuche betrieben worden sind, mittels Data Mining und anderen Techniken, Straßenunfälle zu analysieren. Allerdings haben alle diese Ansätze wichtige Informationen, die in den Unfallbeschreibungen enthalten sind, außer acht gelassen. Als Konsequenz konnten nur beschränkt inhaltsvolle Schlussfolgerungen getroffen werden und insbesondere die kognitiven Aspekte, die zum Unfall führten, wurden nicht analysiert.

Der Prototyp TAKMI [29] wurde in einem Kundencenter zur Analyse von Kundenaufzeichnungen verwendet. Diese beinhalten strukturierte Daten, wie die Informationen über den Kunden, als auch Freitextfelder, um das Anliegen konkretisieren zu können. Die strukturierten Daten wurden zwar schon innerhalb des Kundencenters zur Analyse verwendet, jedoch wurden die unstrukturierten Daten weitestgehend nicht beachtet. Die Arbeiter im Kundencenter, die manuell die Kundenaufzeichnungen auswerten mussten, wurden durch die Verwendung von TAKMI entlastet und die Qualität der gelieferten Ergebnisse wurde verbessert. Weiterhin ist es nun möglich, die große Menge an textuellen Informationen zu verarbeiten, anstelle von nur einigen wenigen.

[30] berichtet, dass das ICA System bereits positiv innerhalb von Hewlett-Packard eingesetzt wurde. So konnte bei einer Produktlinie ein Problem aufgedeckt werden, das durch die vorherigen Analysen mittels den strukturierten Daten nicht entdeckt worden ist. Außerdem konnte das neue System auch umgekehrt aufzeigen, dass ein vermeintliches Problem, doch keines ist.

4 Forschungsfrage und Untersuchungsziel

Die im Kapitel 3 vorgestellten verwandten Arbeiten versäumen es oftmals, die unstrukturierten Informationen als Ergänzung zu den strukturierten Informationen anzusehen. Text Mining Methoden werden auf unstrukturierte Daten angewandt ohne eine vorherige Filterung der Informationen, die bereits in den strukturierten Daten enthalten sind. Dadurch wurden Informationen aus den unstrukturierten Daten erschlossen, die auch mit weniger Aufwand aus den strukturierten Daten gewonnen werden könnten. Die eingangs erwähnte Situation, dass vermehrt unstrukturierte Daten anfallen und auch strukturierte Datensätzen häufig mit unstrukturierten Informationen, wie Freitexten ergänzt werden, lässt folgende Frage zu:
Können automatisiert Informationen aus den unstrukturierten Daten gewonnen werden, die die Informationen aus den strukturierten Daten ergänzen?
Dies würde zu einer Erhöhung der Datenqualitätsdimension Vollständigkeit (siehe Abschnitt 2.1) führen. Für die automatische Auswertung kommt das Clustering zum Einsatz. Dabei liegt kein Fokus auf den verschiedenartigen Clustering-Algorithmen. Vielmehr liegt der Fokus auf der Vorverarbeitung und insbesondere auf dem Entfernen von Stoppwörtern sowie dem Herausfiltern von bereits in den strukturierten Daten enthaltenen Informationen. Das Herausfiltern der qualitativ hochwertigen Informationen aus den strukturierten Daten geschieht, damit diese beim Clustering nicht neue in den unstrukturierten Daten enthaltenen Informationen verdecken. Die Freitextfelder werden vor dem Clustering in eine Vektorraumdarstellung überführt. Für die automatische Clusteranalyse wird ein Prototyp entwickelt, der eine Clusteranalyse mit Bezug auf die strukturierten Informationen durchführt und der auf den NHTSA-Datensatz (siehe Abschnitt 7.1.1) und auf einen Datensatz aus der Industrie (siehe Abschnitt 7.1.2) angewendet wird. Für die Feststellung eines Informationsgewinns werden Datenanalysen mit Bezug auf die strukturierten Daten durchgeführt. Hierzu werden auch die Distanzen zu den Clusterzentren verwendet um nahe am Zentrum und damit qualitativ hochwertige gewonnene Informationen zu präsentieren. [2]

5 Konzept

Dieses Kapitel beschäftigt sich mit dem Konzept, das dem Prototyp zugrunde liegt. Da der Prototyp den NHTSA-Datensatz (siehe Abschnitt 7.1.1) verarbeiten können soll, gilt es diesen einlesen zu können. Der Datensatz wird als Datenbank im Microsoft Access Datenbankformat (ACCDB) vorliegen. Wohingegen der Industriedatensatz (siehe Abschnitt 7.1.2) im Comma-separated values (CSV) Format eingelesen wird. Somit sollte der Prototyp in der Lage sein die angegebenen Formate einlesen zu können. Wie eingangs in Kapitel 4 erwähnt, soll es möglich die strukturierten Daten eines Datensatzes zur Gruppierung, beziehungsweise zur Kategorisierung verwenden zu können. Damit soll erreicht werden, dass eine automatische Analyse auf einer Teilmenge eines Datensatzes ausgeführt werden kann. Zusätzlich soll der Prototyp eine Möglichkeit bereitstellen, aus den Komponentenbeschreibungen des NHTSA-Datensatzes vereinfachtere Komponentenbegriffe zu erstellen. Um eine sinnvolle automatische Analyse zu ermöglichen, ist es erforderlich, Vorverarbeitungsschritte für die zu analysierenden Freitextfelder bereitzustellen. Notwendige Vorverarbeitungsschritte sind mindestens die Tokenisierung der Freitextfelder und das Entfernen von Stoppwörtern. Der k-means Algorithmus wird für die automatische Analyse der Freitextfelder zum Einsatz kommen. Für sinnvolle Ergebnisse ist eine Vektorisierung der vorverarbeiteten Freitextfelder nötig und entscheidend. Die Ergebnisse des Clusteringprozesses sollen gespeichert werden, wobei das Clusterzentrum jedes Eintrags im Datensatz sowie die Distanz zu diesem und die Bedeutung des Clusterzentrums festgehalten werden sollen. Einfache visuelle Darstellungen sind ebenso sinnvoll für das Verständnis der Clusteringergebnisse und sollen vom Prototypen bereitgestellt werden. Grundsätzlich ist es nötig, den gesamten Clusteringprozess, also das Einlesen des Datensatzes mit der Angabe der zu verwendenden Kategorien, die Vorverarbeitung der Freitextfelder, die Vektorisierung und das eigentliche Clustering möglichst einfach und präzise als Nutzer definieren zu können. Weiterhin soll der Prototyp erweiterbar sein. Es sollte möglich sein, neue Clustering-Algorithmen oder Vorverarbeitungsschritte hinzuzufügen.

Die konzeptuellen Hauptbestandteile des Prototyps lassen sich wie folgt zusammenfassen:

- Einlesen der Clusteringkonfiguration
- Einlesen des Datensatzes
- Erstellen vereinfachter Komponentenbeschreibungen (nur für NHTSA-Datensatz)
- Vorverarbeitung der Freitextfelder
- Vektorisierung der Freitextfelder

- Clustern der Freitextfelder
- Schreiben der Clusteringergebnisse
- Visualisierung der Clusteringergebnisse
- Erweiterbarkeit

5.1 Funktionsweise

Zuallererst wird eine Konfigurationsdatei eingelesen. Aus dieser Konfigurationsdatei wird ein Clusteringprozess erstellt. Die Eingabemethode liest einen Datensatz im spezifizierten Format, beispielsweise CSV, ein und behält nur jene, die mit den angegebenen Kategorien übereinstimmen. Die gefilterten Freitextfelder werden dann an die Vorverarbeitungspipeline weitergegeben. Die Vorverarbeitungspipeline stellt sicher, dass die Freitexte sukzessive durch alle spezifizierten Vorverarbeitungsschritte vorverarbeitet werden. Am Ende der Pipeline werden die vorverarbeiteten Freitextfelder wieder zurückgegeben und dem Vektorisierer übergeben, der aus den Freitextfeldern eine Term-Dokument Matrix erstellt. Diese wird wiederum dem Clusterer übergeben, der aus der Term-Dokument Matrix Clusterzentren bildet und jeden Eintrag aus dem Datensatz einem Clusterzentrum zuordnet. Zum Schluss werden die Ergebnisse des Clusterers gespeichert und Visualisierungen erstellt.

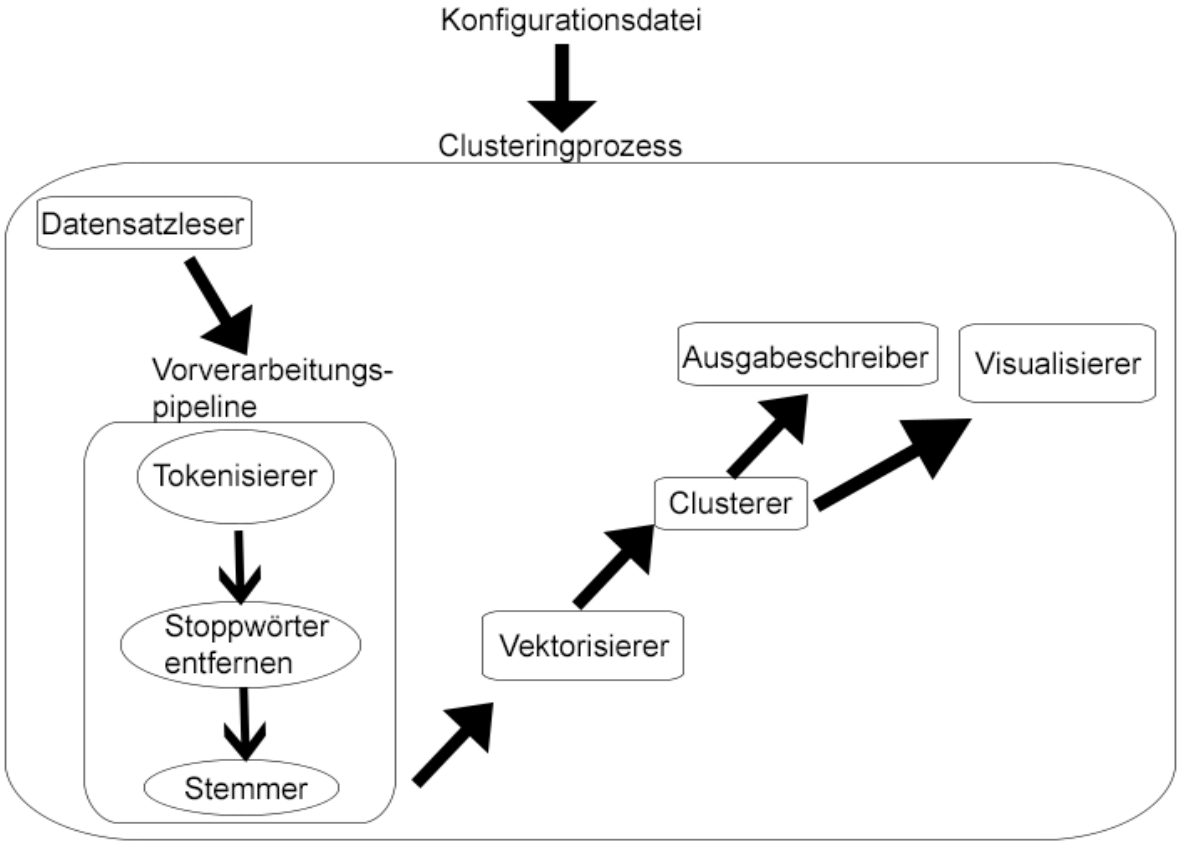

Abbildung 5.1: Prototyp - Aufbau eines Clusteringprozess

6 Implementierung

Im Folgenden werden zuerst die verwendeten Bibliotheken vorgestellt, die für den Prototyp verwendet wurden. Anschließend wird die Umsetzung des Konzepts erläutert.

6.1 Verwendete Werkzeuge

6.1.1 Python

Python[1] ist eine interpretierte, höhere Programmiersprache. Sie ist in der Wissenschaft sehr verbreitet und findet häufig Anwendung gerade bei explorativen oder auch bei rechnerisch gestützten Forschungsaufgaben. Zu den Gründen der Durchsetzung zählen wohl die einfach gehaltene und verständliche Syntax sowie der Reichtum an eingebauten Datentypen, wie etwa Strings, Listen und Dictionaries. [31] Aus diesen Gründen und der damit verbundenen Einsteigerfreundlichkeit, wird Python als Programmiersprache für den Prototyp verwendet. Des Weiteren gibt es eine große Auswahl an nützlichen Bibliotheken für den Prototyp, die im Folgenden vorgestellt werden.

6.1.2 ConfigObj

Die in Python geschriebene Bibliothek ConfigObj[2] dient zum Einlesen und Schreiben von Konfigurationsdateien. ConfigObj zeichnet sich durch eine einfache Handhabung und einen großen Funktionsumfang aus. Zu den Funktionen zählen unter anderem [32]:

- Unterstützung von Untersektionen – es kann beliebig tief genestet werden
- Konfigurationsdateien können Kommentare enthalten
- Unterstützung von Listen
- Unicode-Unterstützung
- Validierung der Konfigurationsdatei

[1] https://www.python.org/
[2] https://pypi.python.org/pypi/configobj/5.0.6

ConfigObj wurde für das Parsen von Konfigurationsdateien verwendet, da hiermit gut lesbare Konfigurationsdateien mit verständlicher Syntax verwendet werden können. Außerdem können die Konfigurationsdateien gegen ein eigens definiertes Schema geprüft und validiert werden. Des Weiteren wird ConfigObj auch in größeren Projekten eingesetzt, beispielsweise für die ebenso verwendete Bibliothek Matplotlib (siehe Abschnitt 6.1.3). [32]

6.1.3 Matplotlib

Für die Erstellung von Visualisierungen wird die Python Bibliothek Matplotlib[3] verwendet. Matplotlib ermöglicht eine grafische Darstellung von 2D Diagrammen. Zu den unterstützten Diagrammarten zählen unter anderem [33]:

- Balkendiagramm
- Tortendiagramm
- Kurvendiagramm

Verwendet wird Matplotlib für den Prototyp, als auch später für die Datenanalyse (siehe Kapitel 8) da es hiermit möglich ist, simple Diagramme einfach und mit wenig Code zu erstellen. Durch die hohe Anpassbarkeit können allerdings auch Diagramme nach den eigenen Bedürfnissen erstellt werden. [33]

6.1.4 scikit-learn

Scikit-learn[4] beherbergt eine Sammlung von machine-learning Algorithmen, die durch Python und konsistente Schnittstellen einfach und insbesondere auch für Einsteiger nutzbar gemacht werden. Zu den machine-learning Algorithmen zählen zahlreiche Clustering-Algorithmen, wie zum Beispiel DBSCAN oder Mean Shift. Zu den bereitgestellten Clustering-Algorithmen zählt auch der in Abschnitt 2.10.1 vorgestellte und verwendete k-Means Algorithmus. Weiterhin für den Prototyp nützlich sind die Vektorisierer, wie TF-IDF Vektorisierer, die in scikit-learn enthalten sind sowie Metriken zur Auswertung der Algorithmen und vieles mehr. [34]

6.1.5 NumPy

NumPy-Arrays[5] wurden entwickelt um Rechnungen mit Arrays in Python zu beschleunigen. Diese zeichnen sich durch ihre Performanz aus. In der Wissenschaft ist NumPy eine häufig

[3] http://matplotlib.org/
[4] http://scikit-learn.org/stable/
[5] http://www.numpy.org/

genutzte Bibliothek. In Bezug auf die Vektorisierung von Texten ist die Unterstützung von multidimensionalen Arrays von Vorteil. [35]

Des Weiteren verwendet scikit-learn NumPy-Arrays zur Repräsentation von Daten. Dies trifft sowohl auf die eingehenden, als auch auf die ausgehenden Daten zu. Deshalb werden die Daten innerhalb des Prototyps ebenfalls als NumPy-Arrays dargestellt.[34]

6.1.6 PyODBC

PyODBC[6] ist eine Bibliothek, die eine bequeme Anbindung von ODBC Datenbanken verspricht. Verwendet wird diese Bibliothek, da sie einfach gehalten ist und ohne Probleme auch auf großen Datenbanken (NHTSA-Datensatz) funktioniert. Zu den unterstützten Datenbankformaten zählt auch das verwendete Microsoft Access Datenbankformat (accdb), welches im Prototyp Verwendung findet. [36]

6.1.7 Natural Language Toolkit

Das Natural Language Toolkit (NLTK)[7] ist eine Bibliothek mit einer Sammlung an Modulen zur Verarbeitung von natürlicher Sprache. Zu den Modulen zählen unter anderem[37]:

- Korpuszugriff
- Tokenisierung
- POS Tagging
- Stemming
- Chunking
- Parsing

Durch die Fülle an Modulen, insbesondere Stemmer und Tokenisierer, dient NLTK als Grundpfeiler für die Vorverarbeitungsaufgaben im Prototyp.

[6] https://pypi.python.org/pypi/pyodbc/4.0.3
[7] http://www.nltk.org/

6.1.8 Textacy

Wie NLTK ist auch Textacy[8] eine Bibliothek für die Verarbeitung von natürlichsprachlichen Texten. Jedoch liegt der Fokus von Textacy mehr auf aufwändigeren Vorgängen, wie etwa dem Erkennen der Sprache oder der NER und ist beispielsweise bezüglich Tokenisierung und POS Tagging auf andere Bibliotheken angewiesen. Im Hinblick auf den Prototyp sind die Funktionen bezüglich der Vorverarbeitung von Texten hilfreich. Diese können zur Verbesserung der Textqualität durch Normalisierung und Säuberung dieser beitragen. Hierzu zählen unter anderem die Normalisierung von Leerzeichen, als auch das Ausschreiben von englischen Kontraktionen (didn't -> did not). [38]

6.1.9 TextBlob

TextBlob[9] ist ebenso eine Bibliothek zur Verarbeitung von natürlicher Sprache. TextBlob verwendet intern auch die Bibliothek NLTK und bietet einen ähnlichen Funktionsumfang bezüglich gebräuchlichen NLP-Aufgaben. Diese können bequem durch einfache Schnittstellen verwendet werden. Innerhalb des Prototyps wird TextBlob für die automatische Rechtschreibkorrektur verwendet, da NLTK diese Funktion nicht mitbringt. [39]

6.2 Einlesen der Clusteringkonfiguration

Konfigurationsdateien dienen als Schnittstelle zwischen dem Nutzer und dem Prototyp. In einer Konfigurationsdatei kann ein gesamter Clusteringprozess spezifiziert werden. Für das Einlesen der Konfigurationsdateien, wird die im Abschnitt 6.1.2 vorgestellte Bibliothek ConfigObj eingebunden. Da es möglich sein soll, einen Datensatz bezüglich Kategorien aufzuteilen und zu Clustern, werden alle Dateien im Verzeichnis „config“ die mit „.conf“ enden versucht einzulesen und bei einer korrekten Konfigurationsdatei das Clustering der Konfigurationsdatei entsprechend ausgeführt. Somit kann mit einer Ausführung des Prototyps ein gesamter Datensatz verarbeitet werden. Die Korrektheit einer Konfigurationsdatei wird mithilfe einer vorgefertigten Spezifikation validiert, die die Struktur, sowie die Datentypen der einzelnen möglichen Werte einer Konfigurationsdatei festlegt. ConfigObj ermöglicht bereits eine einfach anzuwendende Validierung mittels dieser vorgefertigten Spezifikation und vergleicht alle einzulesenden Konfigurationsdateien damit. Bei fehlerhaften Konfigurationen werden die existierenden Fehler der Konfiguration ausgegeben und falls sie vorhanden ist, mit der nächsten Konfigurationsdatei weitergemacht. Durch die Spezifikation wird ebenso die Verwendung von Standardwerten ermöglicht, wodurch die erstellten Konfigurationsdateien übersichtlicher ausfallen, da nicht unbedingt alles genau spezifiziert werden muss. Aus der eingelesenen

[8] https://pypi.python.org/pypi/textacy
[9] https://pypi.python.org/pypi/textblob

Konfigurationsdatei werden alle nötigen Instanzen für den eigentlichen Clusteringprozess erstellt und weitergegeben.

6.3 Einlesen des Datensatzes

Wie bereits geschildert liegt der NHTSA-Datensatz für den Prototyp im Microsoft Access Datenbankformat (accdb) vor. Die in Abschnitt 6.1.6 vorgestellte Bibliothek PyODBC kann dieses Datenbankformat, als auch andere Formate einlesen. Dies geschieht durch einen Connection-String der den Pfad und das Format der Datenbank beinhaltet. Ebenso ist es möglich passwortgeschützte Datenbanken einzulesen. Wenn eine Datenbankverbindung steht, wird durch einen Cursor die Datenbank mittels SQL-Statements ausgelesen. Es werden grundsätzlich alle Informationen bei einem Eintrag eingelesen und im Speicher gehalten, jedoch wird durch das SQL-Statement gemäß der Kategorien eine Auswahl der im Speicher zu haltenden und später zu verarbeitenden Einträge getroffen. Die Informationen über die Datenbanken und über die Kategorien werden über die Konfigurationsdatei festgelegt. Ebenso muss spezifiziert werden, welche Spalten innerhalb der Datenbank die Freitextfelder enthalten und auch wie der Name der Datenbanktabelle lautet, in der die Einträge zu finden sind. Es ist möglich jeweils bei den Angaben zu den Kategorien, aber auch bei den Angaben zu den Freitextfeldern eine Liste von Werten zu übergeben, die beachtet werden sollen. Datensätze können auch im CSV Format eingelesen werden. Dabei kann in der Konfigurationsdatei das Encoding sowie das Trennzeichen definiert werden. Ebenso können Angaben über die Freitextfelder und Kategorien getroffen werden. Für den eigentlichen Einlesevorgang wird Pythons mitgliefertes CSV Modul[10] verwendet. Nach dem Einlesen zurückgegeben wird zum einen der nach den angegebenen Kategorien sortierte Datensatz. Und zum anderen nur die Freitextfelder für die nachfolgenden Verarbeitungsschritte.

6.4 Erstellen vereinfachter Komponentenbeschreibungen (nur für NHTSA-Datensatz)

Für die Analyse des NHTSA-Datensatzes, vergleiche Abschnitt 7.1.1, werden zwar die Komponentenbeschreibungen zur Gruppierung verwendet, jedoch in einer allgemeineren Form als in den Daten vorhanden. Dafür wird als erster Schritt, noch bevor der Datensatz für das Clustering eingelesen wird, der Datensatz mit diesen allgemeineren Kategorien versehen. Dies geschieht durch das Auslesen der Komponentenbeschreibungen für jeden einzelnen Eintrag. Diese Beschreibungen werden anhand der Doppelpunkte aufgetrennt. Die an der ersten Stelle stehenden Oberkategorien werden entnommen und in die Datenbank als eigenes Feld eingefügt. Es werden zudem ähnliche Komponentenbeschreibungen zusammengefasst.

[10] https://docs.python.org/3/library/csv.html

Listing 6.1 Beispielimplementierung eines Vorverarbeitungsschritts - Regex-Substitution

```
class RegexSubstitution(PreprocessBase):
   """Class for searching for regular expressions and when found, it will be replaced
       with a specified substitution."""

   (...)

   def transform_string(self, text):
      return re.sub(self._regex, self._substitution, text)

   def transform_tokens(self, tokens):
      return [re.sub(self._regex, self._substitution, token) for token in tokens]
```

6.5 Vorverarbeiten der Freitextfelder

Wie für NLP-Aufgaben üblich (vergleiche Abschnitt 2.3) werden auch im Prototyp die einzelnen Vorverarbeitungsschritte sequenziell entlang einer Pipeline ablaufen. Ein Vorverarbeitungsschritt muss sowohl die abstrakten Methoden zur Vorverarbeitung von ganzen Strings, als auch für bereits tokenisierten Text implementieren. Dies wird damit begründet, dass die Pipeline frei durch den Nutzer festgelegt werden kann und somit ein Vorverarbeitungsschritt sowohl vor, als auch nach einem Tokenisierer eingesetzt werden kann. Die Reihenfolge, in der die Vorverarbeitungsschritte ablaufen, wird durch die Reihenfolge in der Konfigurationsdatei vorgegeben. Listing 6.1 zeigt die Implementierung eines Vorverarbeitungsschritts für die Suche und das Ersetzen von regulären Ausdrücken. Dabei sind die zwei möglichen Fälle gut ersichtlich. Zum Einen ist es möglich, dass ein Vorverarbeitungsschritt einen String übergeben bekommt. In diesem Fall wird der String von links beginnend durchsucht und alle übereinstimmenden nicht überlappenden Muster werden durch den festgelegten Ersetzungsstring ersetzt. Dabei kann der Ersetzungsstring ebenfalls ein regulärer Ausdruck sein und sowohl dieser als auch der reguläre Ausdruck, nachdem gesucht werden soll, können in der Konfigurationsdatei spezifiziert werden. Zum Anderen ist es möglich, dass ein Vorverarbeitungsschritt eine Liste von Tokens übergeben bekommt, falls dieser nach einem Tokenisierer vorkommt. Im Beispiel wird die Liste der Tokens der Reihe nach durchgegangen und für jeden einzelnen Token wird nach dem spezifizierten regulären Ausdruck gesucht und falls das Muster gefunden wird, ersetzt.

Die Vorverarbeitungsschritte werden wie folgt eingeteilt:

- **Grundlegende Vorverarbeitungsschritte zur Normalisierung und Säuberung von Freitexten:** Darunter fällt die Umwandlung der Freitexte in Kleinbuchstaben. Komplexere Aufgaben werden durch die in Abschnitt 6.1.8 vorgestellte Bibliothek Textacy realisiert und können einfach verwendet werden. Zu diesen zählen das Ausschreiben von Kontraktionen, die Normalisierung von Leerzeichen und die Entfernung von URLs, E-Mails, Telefonnummern, Nummern, Satzzeichen, Währungszeichen oder Akzente. Eine englische Rechtschreibkorrektur wird durch die Bibliothek TextBlob (siehe Abschnitt 6.1.9) durchgeführt.

- **Tokenisierung:** Für die möglichen Tokenisierer wird die in Abschnitt 6.1.7 vorgestellte Bibliothek NLTK verwendet. Dabei kann zwischen einem Whitespace Tokenizer[11] oder den standardmäßig verwendeten Penn Treebank Tokenizer[12] gewählt werden. Kollokationen, also Wörter die in der Regel gemeinsam vorkommen, werden durch NLTKs Multi-Word Expression Tokenizer[13] erstellt. Die Kollokationen werden in einer eigenen Textdatei definiert, wobei jede Zeile eine Kollokation enthält. Diese Datei wird, falls in der Konfigurationsdatei definiert, eingelesen und dem Multi-Word Expression Tokenizer übergeben.
- **Stoppwörter Entfernen:** Für das Entfernen der Stoppwörter wird ebenfalls auf NLTK zurückgegriffen. Dabei wird NLTKs Word List Corpus Reader[14] für das Einlesen einer Stoppwortliste verwendet. Es ist somit möglich die in NLTK verfügbaren Stoppwortlisten, wie beispielsweise Englisch, zu verwenden oder eigene Stoppwortlisten zu erstellen und in einer Konfigurationsdatei den Pfad zur Liste anzugeben. Dabei enthält jede Zeile genau ein Stoppwort.
- **Wortersetzung:** Wie bereits erwähnt, können in der Konfigurationsdatei reguläre Ausdrücke angegeben werden nach denen gesucht werden soll und der String, mit dem ein gefundenes Muster ersetzt werden soll. Weiterhin können in einer Textdatei Synonyme definiert werden. Dabei steht in jeder Zeile ein Synonympaar. Dieses besteht aus dem Wort nachdem gesucht werden soll und mit einem Tabulatorzeichen getrennt, das Wort mit dem es ersetzt werden soll. Der Pfad zu der Textdatei mit den Synonymen kann in der Konfigurationsdatei angegeben werden, wodurch diese eingelesen wird. Ein spezielleres und eigens implementiertes Verfahren dient zur Ersetzung von Wörtern abhängig vom Kontext. Diese Kontextsynonyme können in der Konfigurationsdatei definiert werden. Hierfür müssen zum einen die Hauptwörter angegeben werden, nach denen zuerst gesucht werden soll. Weiterhin müssen Kontextwörter angegeben werden. Diese werden gesucht, sobald ein definiertes Hauptwort gefunden wurde. Außerdem muss noch die Umgebung definiert werden, in der, bei erfolgreichem Auffinden eines Hauptworts, nach den definierten Kontextwörtern gesucht werden soll. Dabei kann angegeben werden, wie viele Wörter vor und wie viele Wörter nach dem gefundenen Hauptwort gesucht werden soll. Sobald ein Hauptwort und Kontextwort in der spezifizierten Umgebung gefunden wurde, wird dieses und das gefundene Kontextwort durch den in der Konfigurationsdatei angegebenen String ersetzt. Das zweite Beispiel 'notdeploy' in der Beispielkonfiguration im Listing 7.1 zeigt eine praktische Anwendung für Kontextsynonyme. Dabei wird nach den Hauptwörtern, beispielsweise 'deploy' gesucht, die auf eine Auslösung des Airbags hindeuten. Durch das Auffinden der angegebenen Kontextwörtern, beispielsweise 'failed', in der definierten Umgebung, kann auf eine gegensätzliche Aussage geschlossen werden.

[11] http://www.nltk.org/_modules/nltk/tokenize/regexp.html#WhitespaceTokenizer
[12] http://www.nltk.org/_modules/nltk/tokenize/treebank.html#TreebankWordTokenizer
[13] http://www.nltk.org/_modules/nltk/tokenize/mwe.html#MWETokenizer
[14] http://www.nltk.org/_modules/nltk/corpus/reader/wordlist.html

Dieser Gegensatz wird dann durch die Ersetzung mit des String 'notdeploy' kenntlich gemacht.

- **Stemming:** Die Rückführung auf den Wortstamm ist oftmals ein sinnvoller Schritt für die Vorverarbeitung von Freitexten. Für diesen Vorverarbeitungsschritt kann der Porter Stemmer[15] von NLTK verwendet werden.

6.6 Vektorisierung der Freitextfelder

Die Vektorisierung geschieht durch die Einbindung der in Abschnitt 6.1.4 vorgestellten Bibliothek scikit-learn. Der Vektorisierer erhält die vorverarbeiteten Freitextfelder und erstellt daraus eine Term-Dokument Matrix. Durch scikit-learn wird die Wahl zwischen Count Vectorizer[16] und TF-IDF Vectorizer[17] ermöglicht. Die hauptsächlichen Einstellungen können über die Konfigurationsdateien spezifiziert werden. Hierzu zählen unter anderem das minimale oder maximale Vorkommen von Wörtern oder die insgesamt maximale Anzahl von Wörtern. Die angesprochenen Vektorisierer verlangen als Eingabe eine Sammlung von Texten und können nicht mit tokenisierten Texten umgehen. Damit jedoch eigene Tokenisierer in den Vorverarbeitungsschritten verwendet werden können, werden die tokenisierten Texte wieder als ein String zusammengefügt, jedoch werden Tabulatorzeichen anstelle von üblichen Leerzeichen zwischen Tokens eingesetzt. Der Vektorisierer trennt dann die Strings anhand der Tabulatorzeichen auf, wodurch die vorherige Tokenisierung wiederhergestellt wird. Dies sorgt allerdings für die Einschränkung, dass die ursprünglichen Freitexte keine Tabulatorzeichen enthalten dürfen, da dies ansonsten zu einer falschen Tokenisierung führen könnte.

6.7 Clustering der Freitextfelder

Auch das Clustering wird durch scikit-learn (siehe Abschnitt 6.1.4) ermöglicht. Da der Fokus in dieser Arbeit allerdings nicht auf den Clustering-Algorithmus liegt, wird lediglich der k-Means Algorithmus aus der Bibliothek verwendet, obwohl scikit-learn einige mehr bereitstellen würde. Grundsätzlich ist es aber möglich, den Prototyp auch mit anderen Clustering-Algorithmen aus scikit-learn zu betreiben. Der Clusterer erhält die vorverarbeiteten und vektorisierten Freitextfelder, also die Term-Dokument Matrix als Eingabe. Auch die Parameter für den Clusterer können in den Konfigurationsdateien gesetzt werden. Hierzu zählen unter anderem die Anzahl der Clusterzentren und die Anzahl der maximalen Iterationen, wobei für alle Parameter bereits Standardwerte vordefiniert sind.

[15] http://www.nltk.org/_modules/nltk/stem/porter.html#PorterStemmer
[16] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
[17] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

6.8 Speichern der Clusteringergebnisse

Die Ergebnisse der Clusteranalyse sollen auch gespeichert werden können. Dafür wird der zu Beginn eingelesene Datensatz wieder als CSV Datei ausgegeben. Zudem werden noch die aus der Clusteranalyse gewonnenen Informationen in die CSV Datei gespeichert. Zu jedem Eintrag kommt die Beschriftung des Clusterzentrums, der Term, der am meisten zum Clusterzentrum beitragt, die Koordinate des Terms, die Distanz zum Clusterzentrum und falls vorhanden noch weitere Clusterterme hinzu. Die meisten Informationen können mit wenig Aufwand aus den Rückgabeergebnissen von scikit-learn ausgelesen werden. Die Distanz ist die euklidische Distanz. Die hierfür verwendete Funktion befindet sich in der ebenfalls von scikit-learn verwendeten Bibliothek SciPy[18]. Dieser Funktion wird die Term-Dokument Matrix von einem einzelnen Eintrag und das Clusterzentrum übergeben. Ebenso werden allgemeine Informationen zum Clusteringvorgang als Textdatei abgespeichert. Hierzu zählen Informationen bezüglich der Ausmaße der Term-Document Matrix sowie zu den Merkmalen der Matrix. Außerdem werden Informationen über den Vektorisierer und den Clusterer sowie über die gesetzten Parametern mitgespeichert. Des Weiteren werden noch die gefundenen Cluster mit der Anzahl der zugeordneten Einträge zum Cluster sowie die Terme, die ein Cluster kennzeichnen mit den zughörigen Gewichtungen gespeichert.

6.9 Visualisierungen

Mithilfe der in Abschnitt 6.1.3 vorgestellten Bibliothek Matplotlib können Diagramme zu den Clusteranalysen erstellt werden. Eine Orientierung zur Implementation liefern dabei die Beispiele von scikit-learn[19] Für die Darstellung der Cluster wird die Größe der Term-Dokument Matrix minimiert. Die einzelnen visualisierten Einträge erhalten eine Farbe entsprechend ihrer Clusterzugehörigkeit. Wenn die Anzahl der Einträge nicht zu groß ist, kann auch ein Diagramm mit den Silhouette-Score Werten (siehe Abschnitt 2.11) erstellt werden. Silhouette-Score ist bereits in scikit-learn implementiert. Es wird zum einen der durchschnittliche Wert für alle Einträge, als auch der Wert eines jeden einzelnen Eintrags visualisiert. Für die Visualisierungen wird wieder die Bibliothek Matplotlib verwendet. Ob Diagramme erstellt werden sollen kann in den Konfigurationsdateien festgelegt werden.

6.10 Erweiterbarkeit

Damit eine einfache Erweiterbarkeit sichergestellt ist, wurden die einzelnen Aufgabenbereiche des Prototyps in Module aufgetrennt und abstrakte Klassen mit abstrakten Methoden defi-

[18] https://www.scipy.org/
[19] http://scikit-learn.org/stable/auto_examples/index.html

Listing 6.2 Erweiterung des Prototyps - GermanStemmer

```
class GermanStemmer(PreprocessBase):
   """Class for applying the german stemmer from nltk."""
   def __init__(self):
      self._stemmer = stem.snowball.GermanStemmer()

   def transform_string(self, text):
      return " ".join(self.transform_tokens(text.split()))

   def transform_tokens(self, tokens):
      return [self._stemmer.stem(token) for token in tokens]
```

niert, von denen geerbt werden kann. Zusätzlich ist der Prototyp durchgehend kommentiert. Grundsätzlich gilt, dass es für neue Komponenten meistens reicht von den vorgegebenen abstrakten Klassen zu erben und die Methoden zu implementieren. Für neue Komponenten zum Einlesen oder Schreiben von Datensätzen muss lediglich die Methode zum Einlesen oder Schreiben angepasst werden. Ähnlich verhält es sich auch mit der Erstellung von Kategorien, die standardmäßig nur für den NHTSA-Datensatz funktioniert. Wenn nötig kann auch hier eine eigene Implementierung hinzugefügt werden. Auch neue Vorverarbeitungsschritte können ohne Weiteres implementiert werden. Hierfür wird lediglich vorausgesetzt, dass diese die Methoden zur Verarbeitung von Tokens und Strings implementieren. Ebenso können zusätzliche Methoden für die Visualisierung, für das Clustering oder die Vektorisierung bequem hinzugefügt werden. Am Ende muss für jede Erweiterung sichergestellt werden, dass diese auch durch die Konfigurationsdateien benutzt werden kann. Hierzu gilt es die Spezifikation für valide Konfigurationsdateien abzuändern, ebenso wie die Klasse zum Lesen der Konfigurationsdateien.

Für ein Beispiel zur Erweiterbarkeit wird die Nutzung des GermanStemmers[20] aus NLTK zum Prototyp hinzugefügt. Listing 6.2 zeigt die Implementierung für die eigentliche Funktionalität. Da es sich um einen Vorverarbeitungsschritt handelt, wird der neue Stemmer von der Basisklasse abgeleitet und muss die abstrakten Methoden zur Transformierung von einem String und die Transformierung von Tokens implementieren.

Die Abänderung der Klasse zum Konfigurationsleser folgt als nächstes. Listing 6.3 zeigt diese Änderung für die Erstellung der Vorverarbeitungspipeline. Die Methode, die für das Einlesen der Vorverarbeitungsschritte und der Erstellung der Vorverarbeitungspipeline zuständig ist, muss den Stemmer instanziieren und der Pipeline übergeben, falls in einer Konfigurationsdatei der Stemmer angegeben ist.

Als letztes gilt es, in der Spezifikation für valide Konfigurationen den Parameter 'GermanStemmer' hinzuzufügen, siehe Listing 6.4

[20] http://www.nltk.org/_modules/nltk/stem/snowball.html#GermanStemmer

Listing 6.3 Erweiterung des Prototyps - Konfigurationsleser

```
def handle_preprocessing(self, preprocessing_dict):
(...)
   elif entry == 'stemmer':
      stem = preprocessingStep[entry]
      (...)
      elif stem == 'GermanStemmer':
         pipeline.add_preprocessig_step(stemmer.GermanStemmer())
```

Listing 6.4 Erweiterung des Prototyps - Konfigurationsspezifikation

```
[PREPROCESSING]
   (..)
      stemmer = option(PorterStemmer, GermanStemmer, default=None)
```

7 Demonstration

Innerhalb diesem Kapitel wird der entwickelte Prototyp auf realen Datensätzen angewendet. Zunächst werden die zwei Datensätze vorgestellt. Anschließend wird die Anwendung des Prototyps auf dem NHTSA-Datensatz beschrieben und die vom Prototyp erstellten Ausgaben erklärt. Weiterhin wird die Anwendung auf den Industriedatensatz geschildert.

7.1 Daten

7.1.1 NHTSA-Datensatz

Die NHTSA wurde einst 1970 mit dem Auftrag gegründet die Anzahl der Toten, Verletzten und die wirtschaftlichen Folgen durch Fahrzeugunfälle auf den amerikanischen Straßen zu reduzieren. Zu den Verantwortlichkeiten zählen das Setzen von Sicherheitsstandards, die Defekte zu untersuchen und Rückrufaktionen zu starten. Die NHTSA ist dazu ermächtigt im Falle, dass ein Fahrzeug nicht den Sicherheitsstandards entspricht oder sicherheitsrelevante Defekte an einem Fahrzeug existieren, diese zu untersuchen und gegebenenfalls auch das Fahrzeug zurückzurufen. Ein sicherheitsrelevanter Defekt liegt vor, wenn an einem Fahrzeug oder an dem Zubehör Probleme vorliegen, die eine Gefahr für die Sicherheit darstellen und die unter einer Reihe von baugleichen Autos verbreitet sind oder wenn das Zubehör vom selben Typ oder Hersteller ist. [40]

Die NHTSA vehicle owner's complaint database beinhaltet mehr als 1,3 Millionen Berichte über Zwischenfälle mit Fahrzeugen (Stand 29.09.2016) [41]. Beschwerden können seit 1995 beispielsweise über das Internet oder über die Hotline eingereicht werden [22].

Insgesamt gibt es 49 Felder pro Beschwerdeeintrag (Stand 29.09.2016) [41]. Die Felder beinhalten Angaben über das Fahrzeug (beispielsweise Hersteller, Modelljahr, Automarke und Modell und eine spezifische Komponentenbeschreibung), die Anzahl der Verletzten und Toten sowie einiges an zusätzlichen Informationen. Außerdem ist eine Beschreibung zu dem Vorfall als Freitextfeld für jeden Eintrag enthalten. [22]
In der Tabelle 7.1 ist ein Ausschnitt eines Eintrags aus dem NHTSA-Datensatz zu sehen.

Hersteller	Modell	Komponentenbeschreibung	Beschwerdebeschreibung
PONTIAC	SUNFIRE	AIR BAGS	NO DEPLOYMENT OF AIR BAG DURING VEHICLE COLLISION

Tabelle 7.1: Ausschnitt einer Beschwerde aus dem NHTSA-Datensatz

7.1.2 Industriedatensatz

Ein zweiter Datensatz enthält die erfassten Informationen über Stillstände auf einer Fertigungslinie. Der Industriedatensatz liegt im CSV Format vor. Der Datensatz besteht aus 152.687 Einträgen. In den strukturierten Feldern sind unter anderem spezifische Fehlercodes und die Dauer für einen Stillstand in Sekunden enthalten. Weiterhin liegen Informationen über die Ursache sowie weitere Anmerkungen in Form von zwei Freitextfeldern vor. Werksmitarbeiter können diese direkt ausfüllen.

7.2 Anwenden des Prototyps auf den NHTSA-Datensatz

Der implementierte Prototyp wird verwendet, um den gesamten NHTSA-Datensatz zu clustern. Hierzu wird der im CSV Format vorliegende Datensatz in eine Microsoft Access Datenbank umgewandelt. Dieser Datensatz wird in dem Prototyp eingegeben. Zuallererst werden die Kategorien für den Datensatz aus den Komponentenbeschreibungen erstellt. Abbildung 7.1 zeigt die erstellten Kategorien und die Anzahl der einzelnen Kategorien.

Kategorie	Vorkommen
ENGINE AND ENGINE COOLING	151.570
BRAKES	137.598
ELECTRICAL SYSTEM	137.494
POWER TRAIN	129.340
AIR BAGS	88.017
STRUCTURE	84.036
TIRES	81.053
(...)	(...)
	1321562

Abbildung 7.1: Auflistung der erstellten Kategorien samt Anzahl

Für jede einzelne Kategorie wird eine Konfigurationsdatei erstellt, um den gesamten NHTSA-Datensatz entsprechend zu clustern.

Listing 7.1 zeigt einen Ausschnitt einer beispielhaften Konfigurationsdatei für die Kategorie Air Bags. Gut zu erkennen ist die Aufgliederung der einzelnen Aufgaben auch innerhalb der Konfigurationsdatei. Der erste Abschnitt beschäftigt sich mit der Eingabe. Gefordert wird das Einlesen einer Microsoft Access Datenbank. Der zugehörige Pfad zu dieser sowie der Tabellenname werden ebenso angegeben. Die Unterkategorie spezifiziert die Spalte, in der die Kategorien vorzufinden sind, die Kategorien nach denen aussortiert werden soll und die Spalten, in denen die Freitexte zu finden sind. Nachfolgend kann ein Ordner für die Ausgabe angegeben werden und ob Diagramme erstellt werden sollen oder nicht. Außerdem ist die Pipeline der Vorverarbeitung angegeben. Diese wird der Übersicht halber in einzelne Schritte unterteilt und wird später der Reihe nach von oben nach unten ablaufen. Auch zu sehen ist die Definition von Kontextsynonymen als Vorverarbeitungsschritt. Wenn beispielsweise vor ‚available' das Wort ‚not' steht, sollen diese durch das Wort ‚unavailable' ersetzt werden. Am Ende sind noch die Parameter für den Vektorisierer und Clusterer angegeben.

Die Ausgabe des Clusteringprozess besteht wie festgelegt, neben den textuellen Ergebnissen, auch aus den erstellten Diagrammen.

Listing 7.2 zeigt einen Ausschnitt aus den gespeicherten Clusterinformationen vergleiche Abschnitt 6.8. Aus diesen können die Informationen über die genauen Clusterzentren sowie den verwendeten Vektorisierer und Clusterer entnommen werden. In der Tabelle 7.2 ist der Ausschnitt aus dem vom Prototyp erweiterten NHTSA-Datensatz dargestellt. Die erste Beschwerde ist die aus der Tabelle 7.1. Diese wurde dem 3. Cluster zugeordnet und hat zu dessen Clusterzentrum eine Distanz von 0. Dies liegt daran, da nach dem Herausfiltern der Stoppwörter und durch die Ersetzung von 'NO DEPLOYMENT' zu 'notdeploy' mittels Kontextsynonym (siehe Abschnitt 6.5 kein weiterer Term außer 'notdeploy' vorhanden ist. Dies ist auch der Hauptterm des 3. Cluster und aufgrund der maximalen Gewichtung von 1 sind in diesem Cluster nur Beschwerden zugeordnet, die nach der Vorverarbeitung nur noch aus 'notdeploy' bestehen. Weitere ausschlaggebende Terme innerhalb dieses Cluster existieren nicht. Die Abbildung 7.2 stellt die Cluster zweidimensional dar. Die Cluster sind alle mit unterschiedlichen Farben dargestellt und die Clusterzentren werden durch die verschiedenen Nummern illustriert. Zu sehen ist, dass besonders der 3. Cluster sehr dicht ist und alle Objekte nah am Clusterzentrum sind. Allerdings ist der Rest stark überlappend, dies liegt auch daran, dass die vielen Dimensionen der Term-Dokument Matrix zweidimensional dargestellt werden.

7.3 Anwenden des Prototyps auf den Industriedatensatz

Für eine zweite Evaluation des Prototyps wurde er zusätzlich auf Industriedaten aus der Fertigung angewendet. Die in den strukturierten Daten enthaltenen spezifischen Fehlercodes dienen zur Kategorisierung. Für jeden Fehlercode liegt eine automatisiert erstellte Konfigurationsdatei

Listing 7.1 Beispielhafte Konfigurationsdatei

```
[INPUT]
input_type = MSACCESS #Eingabeformat MSACCESS oder CSV
input_path = "C:\\FLAT_CMPL.accdb" #Pfad zum Datensatz
table_name = 'FLAT_CMPL' #Tabelle in der Datenbank

      [[CATEGORY]]
      column_with_category = Category #Spalte mit den Kategorien zur Gruppierung
      categories_to_choose = 'AIR BAGS', #Kategorien nach denen gruppiert werden soll
      columns_with_text_fields = "Feld20", #Spalten mit den Freitextfeldern

[OUTPUT]
save_plot = True #Visualisierungen speichern?

[PREPROCESSING]

      [[step1]]
      lowercase = True #Umwandeln in Kleinbuchstaben
      remove_punctuation = True #Satzzeichen entfernen
      unpack_contractions = True #Kontraktionen expandieren

            [[[CONTEXT-SYNONYMS]]] #Kontextsynonyme
                  [[[[UNAVAILABLE]]]]
                  #'not available' ersetzen durch 'unavailable'
                  main_words = 'available',
                  context_words = 'not',
                  before = 1
                  after = 0
                  substitution = 'unavailable'

                  [[[[notdeploy]]]]
                  #Beispielsweise 'deployment failed' ersetzen durch 'notdeploy'
                  main_words = 'deploy', 'deployment'
                  context_words = 'failed', 'not'
                  before = 5
                  after = 1
                  substitution = 'notdeploy'

[VECTORIZING]
vectorizer = TF-IDF #Wahl des Vektorisierers. Alternative CountVectorizer
min_df = 0.02 #Minimales Vorkommen eines Worts im Datensatz

[CLUSTERING]
algorithm = kmeans #Wahl des Clustering-Algorithmus
n_clusters = 12 #Anzahl der Cluster
```

Listing 7.2 Ausgabe der Cluster-Informationen

```
SHAPE: 88017, 225
FEATURES:
acceler, act, activ, addit, address, advis, affect, ago, air bag warning light, (...)

vectorizer: TfidfVectorizer(binary=True (...))

Clustering sparse data with KMeans(n_clusters=12 (...))

Cluster 0 22708 : sensor 0.041, deploy 0.036, avail 0.030, (...)
Cluster 1 7254 : air bag warning light 0.444, (...)
Cluster 2 6284 : deploy 0.570, (...)
Cluster 3 4473 : notdeploy 1.000, (...)
Cluster 4 15226 : notdeploy 0.269, front end 0.075, (...)
Cluster 5 8725 : unavail 0.557, nhtsa campaign 0.549, (...)
Cluster 6 4813 : seat belt 0.361, notdeploy 0.121, (...)
Cluster 7 9304 : light 0.320, (...)
(...)
```

Beschwerde (gekürzt)	Cluster	Distanz	Hauptterm	Gewichtung Hauptterm	Clusterterme
NO DEPLOYMENT OF AIR BAG	3	0	notdeploy	1	notdeploy
AIR BAG WARNING LIGHT KEEP COMING ON	1	0,585	air bag warning light	0,444	air bag warning light
NHTSA CAMPAIGN NUMBER (...) PARTS (...) WERE NOT AVAILABLE	5	0,271	unavail	0,557	unavail, nhtsa campaign

Tabelle 7.2: Ausschnitt aus einem vom Prototyp erweitertem Datensatz

für den Prototyp vor. Analog zu der Vorgehensweise für den NHTSA-Datensatz in Abschnitt 7.2 werden auch hier die strukturierten Daten über die verwendeten Kategorien genutzt, um die bereits sauber vorhandenen Informationen aus den Freitexten herauszunehmen. Dies geschieht durch das Setzen der Begriffserklärungen für die Fehlercodes auf eine Stoppwortliste. Es werden ferner eine deutsche Stoppwortliste und ein Stemmer[1] für die deutsche Sprache verwendet. Für das Clustering werden beide existenten Freitextfelder verwendet, damit potenziell viele Informationen erschlossen werden können. Um einige Rechtschreibfehler zu verbessern sowie

[1] http://www.nltk.org/_modules/nltk/stem/snowball.html#GermanStemmer

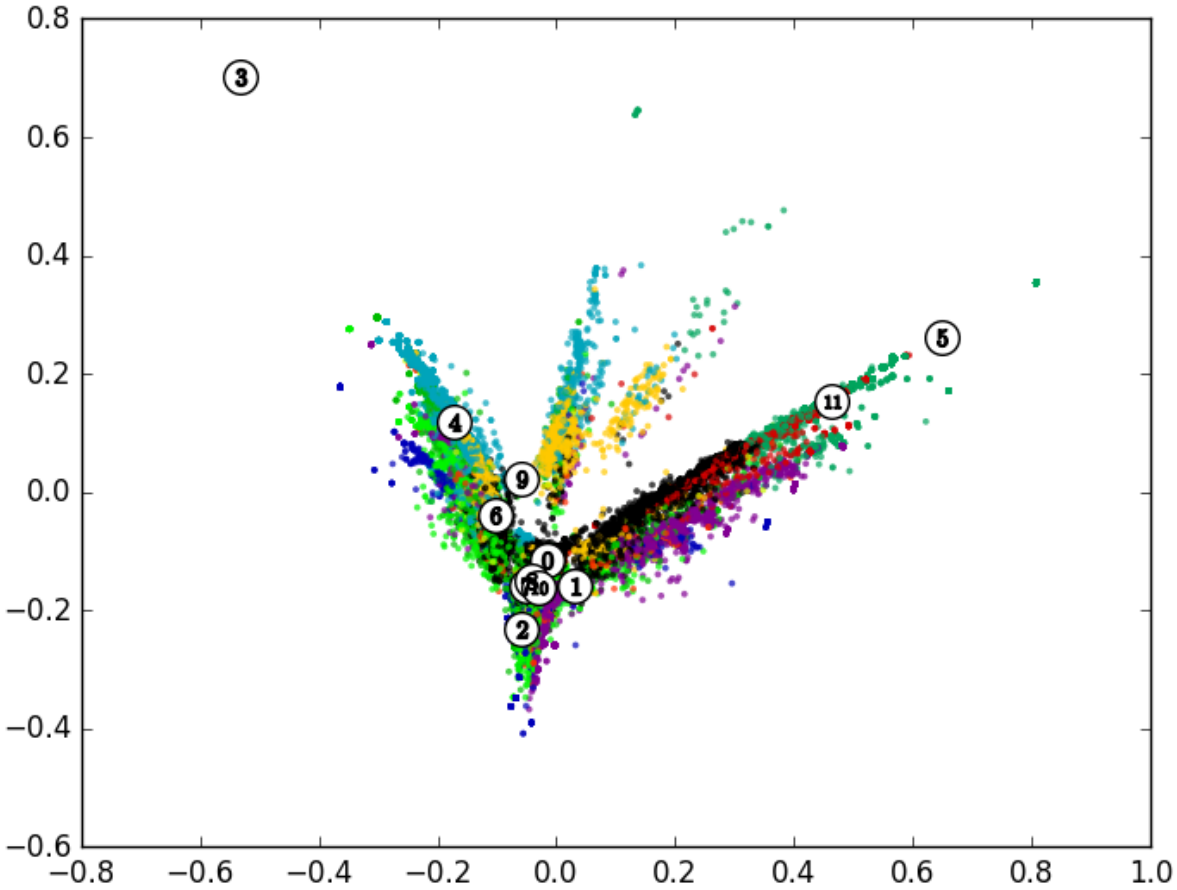

Abbildung 7.2: Beispielhafte visuelle Ausgabe des Prototyps

verschiedene Formulierungen anzugleichen, werden hinzukommend einige reguläre Ausdrücke, Synonyme und Kontextsynonyme (siehe Abschnitt 6.5) zu den Vorverarbeitungsschritten hinzugefügt.

8 Ergebnisse und Evaluation

Im folgenden Kapitel werden die vom entwickelten Prototyp hinzugefügten Informationen ausgewertet. Hierfür werden verschiedene Datenanalysen auf den erweiterten Datensätzen angewendet. Dadurch kann festgestellt werden, ob die Qualität der erweiterten Datensätze durch die Anwendung des Prototyps erhöht wurde.

8.1 NHTSA-Datensatz

Der im Abschnitt 7.1.1 vorgestellte NHTSA-Datensatz kann zwar durch den entwickelten Prototyp (siehe Kapitel 6) automatisch in Cluster unterteilt werden, jedoch bleibt herauszufinden, ob die automatische Cluster-Analyse der Freitextfelder auch zu zusätzlichen Informationen führt, die nicht in den strukturierten Daten enthalten sind. Hierzu werden die Ergebnisse vom Prototyp (Abschnitt 7.2) wieder zu einer Microsoft Access Datenbank vereinigt. Auf diesem erweiterten Datensatz werden sodann Datenanalysen vollzogen. Dazu werden SQL-Abfragen verwendet, um die zu analysierenden Einträge zu erhalten. Diese werden mithilfe von Python (Abschnitt 6.1.1) analysiert. Anschließend werden die Ergebnisse der Datenanalysen zur Veranschaulichung durch Matplotlib (Abschnitt 6.1.3) als Diagramme dargestellt. Außerdem wurden all jene Einträge vor der Datenanalyse herausgefiltert, die zu einem Cluster zugeordnet wurden, dessen Hauptterm aus dem leeren String besteht. Weiterhin wurden Einträge entfernt, die zu einem Cluster zugeordnet wurden, dessen Hauptterm eine niedrige Gewichtung hat. Dies wird durchgeführt, da mittels TF-IDF (siehe Abschnitt 2.9.1) aussagekräftige Terme innerhalb einer Textsammlung hoch gewichtet werden und solche Cluster vorwiegend sich stark unterscheidende Freitexte enthalten. Die im Listing 8.1 enthaltenen Beispielbeschwerdetexte, die alle zum selben Cluster und mit demselben Abstand zum Clusterzentrum innerhalb der Komponente Airbags zugeordnet worden sind, illustrieren die Problematik. Somit fallen 369.534 Einträge, ungefähr 28 %, aus dem Datensatz weg und es bleiben insgesamt 952.028 Einträge für die Datenanalysen übrig. Es werden im Folgenden Beispielbeschwerdebeschreibungen vorgestellt, die sich nah am Clusterzentrum befinden, also die die euklidische Distanz minimieren. Diese Informationen wurden vom entwickelten Prototyp (siehe Abschnitt 6.8) zur Verfügung gestellt.

Zu allererst wird die Häufigkeit der als strukturierte Informationen vorliegenden Komponentenbeschreibungen mit der Häufigkeit der hinzugefügten Cluster-Terme verglichen. Dadurch kann ein Überblick über den Datensatz und ein Einblick über die am häufigsten betroffenen

Listing 8.1 Beispiel für selten vorkommende Beschwerden, die zum selben Cluster zugeordnet wurden sind

```
SPRING FAILURE IN AIRBAG.

HEAD ON COLLISION AT 50MPH FAILED TO SET OFF AIRBAG

(...) CUSTOMER SHOULD NOT HAVE TO PAY FOR REPLACING A FAILTY PART AS AND AIR BAG CHIP.
```

Listing 8.2 SQL-Befehl zur Filterung der zu analysierenden Beschwerden

```
SELECT CMPLID, MAKETXT, MODELTXT, YEARTXT, FIRE COMPDESC, ClusterMainTerm, Distance,
    ClusterMainTermWeight from FLAT_CMPL WHERE ClusterMainTerm NOT IN ('') AND
    ClusterMainTermWeight > 0.1
```

Komponenten gewonnen werden. Der SQL-Befehl in Listing 8.2 liest alle Einträge aus, die nicht in einem Cluster, dessen Hauptterm der leere String ist oder in einem Cluster mit niedriger Gewichtung des Hauptterms enthalten sind. Nachdem die Gesamtanzahlen mit Python festgestellt wurden, können diese mithilfe der Matplotlib visualisiert werden. Für alle Visualisierungen gilt, dass die eigentlichen Stämme der Cluster-Terme, durch Grundformen ersetzt worden sind. Abbildung 8.1 zeigt die häufigsten Komponentenbeschreibungen und die häufigsten Cluster-Terme. Die Komponentenbeschreibungen waren bereits in den strukturierten Daten des NHTSA-Datensatz enthalten. Dabei können die Komponentenbeschreibungen unterschiedlich spezifisch sein. Alle haben gemein, dass sie erst einmal eine Oberkategorie bezüglich der Beschwerde angeben. Diese können dann weiterhin verfeinert werden. Eine Verfeinerung der Komponentenbeschreibung ist nach einem ‚:' vorzufinden (siehe Abschnitt 6.4).

In der Abbildung 8.1 ist zu sehen, dass insbesondere die generellen Oberkategorien der Komponentenbeschreibungen häufig vorkommen. Dabei sind Probleme mit elektrischen Systemen (‚ELECTRICAL SYSTEM') besonders häufig vorzufinden. Auch Beschwerden bezüglich Airbags 'AIR BAGS', Antriebssträngen (‚POWER TRAIN') insbesondere mit Automatikgetrieben (‚POWER TRAIN:AUTOMATIC TRANSMISSION') sind häufig. In Verbindung mit den Cluster-Termen sind insbesondere Probleme mit dem Getriebe (‚transmission') zu sehen: „TRANSMISSION FAILURE AT 105,000 MILES CAR IS BEING REPAIRED BOBS TRANSMISSION, GRAND RAPIDS, MICHIGAN"

Weiterhin liefert die Datenanalyse unter der Verwendung der neu durch den Prototypen gewonnenen Informationen, Hinweise auf das Vernehmen von unüblichen Geräuschen 'noise'. So sind damit Beschwerden bezüglich den elektrischen Systemen zu finden: „IT MAKES A LOUD NOISE AND NO 1 KNOW WHAT IT IS" Aber auch bezüglich Automatikgetrieben: „TRANSMISSION MAKES LOUD WHINING NOISE." Oder bezüglich der Lenkung 'STEERING': „WHINING NOISE WHEN TURNING STEERING WHEEL. HAS BEEN IN SHOP 4 TIMES. WITH SAME PROBLEM."

Ebenso enthalten viele Beschwerden den Cluster-Term ‚acceleration'. Dies weist auf Probleme mit der Beschleunigung hin und steht in Verbindung mit einigen Komponentenbeschreibun-

gen. So sind etwa die Geschwindigkeitsregelung (‚VEHICLE SPEED CONTROL‘): „VEHICLE ACCELERATED BY ITSELF AND CAUSED AN ACCIDENT.“ Oder auch die Komponente bezüglich des Antriebsstrangs (‚POWER TRAIN:AUTOMATIC TRANSMISSION‘): „THE VEHICLE IS NOT ACCELERATING PROPERLY. DEALER WILL REPLACE THE TRANSMISSION.“ betroffen.

Der nächste Cluster-Term ‚unavailable‘ ist ein interessanter und häufig vorkommender Begriff. Bei genauerer Betrachtung beziehen sich Beschwerden in diesem Cluster auf bestehende Rückrufaktionen. Diese verlaufen wohl teilweise nicht plangemäß und insbesondere sind die Austauschteile nicht vorhanden (‚unavailable‘) wodurch sich Reparaturen zeitlich hinziehen. Dabei gibt es Komponentenübergreifend Probleme. Beispielsweise bezüglich des Airbags: „(...) THE PART TO DO THE REPAIR WAS UNAVAILABLE. THE CONTACT STATED THAT THE MANUFACTURER EXCEEDED A REASONABLE AMOUNT OF TIME FOR THE RECALL REPAIR. (...)“

Eine weitere neu durch die hinzugefügten Cluster-Terme gewonnene Information betrifft das Abwürgen (‚stall‘) des Motors. Dieses Problem tritt wieder Komponentenübergreifend auf und ist beispielsweise bezüglich den elektrischen Systemen oder bezüglich des Antriebsstrangs vorzufinden. „VEHICLE STALLED DUE TO AN ELECTRICAL PROBLEM.“ „ENGINE STALLS WHEN APPROACHING A STOP.“

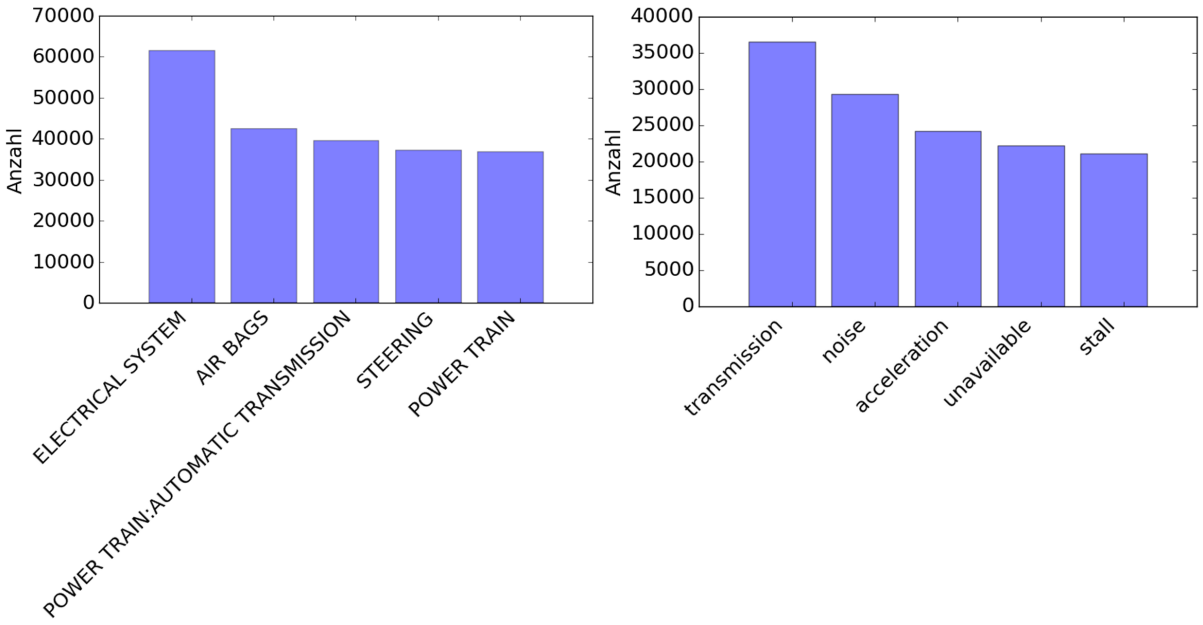

Abbildung 8.1: Häufigste Komponentenbeschreibungen und Cluster-Terme

Schon durch die erste Datenanalyse konnten die insgesamt häufigsten Beschwerden weiter verfeinert und zusätzliche Informationen gewonnen werden. Dabei sind die häufigsten Cluster-Terme nicht spezifisch für eine Komponentenbeschreibung.

Eine weitere Datenanalyse betrifft die Beschwerden, bei denen es zu tödlichen Unfällen gekommen ist. Die Ergebnisse der Analyse sind in Abbildung 8.2 dargestellt. Hierfür wird das strukturierte Feld ‚Deaths' verwendet um nur jene Beschwerden zu analysieren, bei denen es zu mindestens einem Todesfall gekommen ist. Insgesamt werden dadurch 3426 Beschwerden extrahiert. Wenig verwunderlich taucht die Komponente Airbags bei vielen Beschwerden mit Todesopfern auf. Dieser ist ein wichtiger Schutzmechanismus bei Aufprallunfällen und ein Defekt kann zu fatalen Folgen führen. So zeigt der erste Cluster-Term ziemlich deutlich einen Mehrwert zu der bloßen Komponentenbeschreibung Airbag. Dieser lässt einen Rückschluss auf den Grund für die fatale Folge zu, nämlich das Nichtauslösen (‚notdeploy') des Airbags: „NO DEPLOYMENT OF AIR BAG IN ACCIDENT." Doch auch die Auslösung der Airbags kann zu schweren Verletzungen führen und bis zum Tod führen. So enthalten Beschwerden Informationen über eine heftige Auslösung (‚deploy') des Airbags: „DEPLOYMENT OF PASSENGER'S AIRBAG WAS VIOLENT, CAUSING FATALITY. (ATTORNEY FOR CLIENT)" Ein ebenso wichtiger Schutzmechanismus bei Aufprallunfällen ist der Anschnallgurt. Dies wird auch durch die häufige Erwähnung in Beschwerden verdeutlicht. Insbesondere in Verbindung mit der Komponente Airbags tritt dies auf: „CAR WAS HIT ON SIDE. AIRBAGS DID NOT DEPLOY AND PASSENGER SEAT BELT MALFUNCTIONED RESULTING IN DEATH OF PASSENGER." Auch zu tödlichen Unfällen kann eine Reifenpanne (‚blowout') führen. Dieser Cluster-Term kommt nur bezüglich der Komponente Reifen (‚TIRES') vor: „USED TIRES PURCHASED 6 DAYS BEFORE BLOWOUT WHICH INJURED 3 AND KILLED 2."

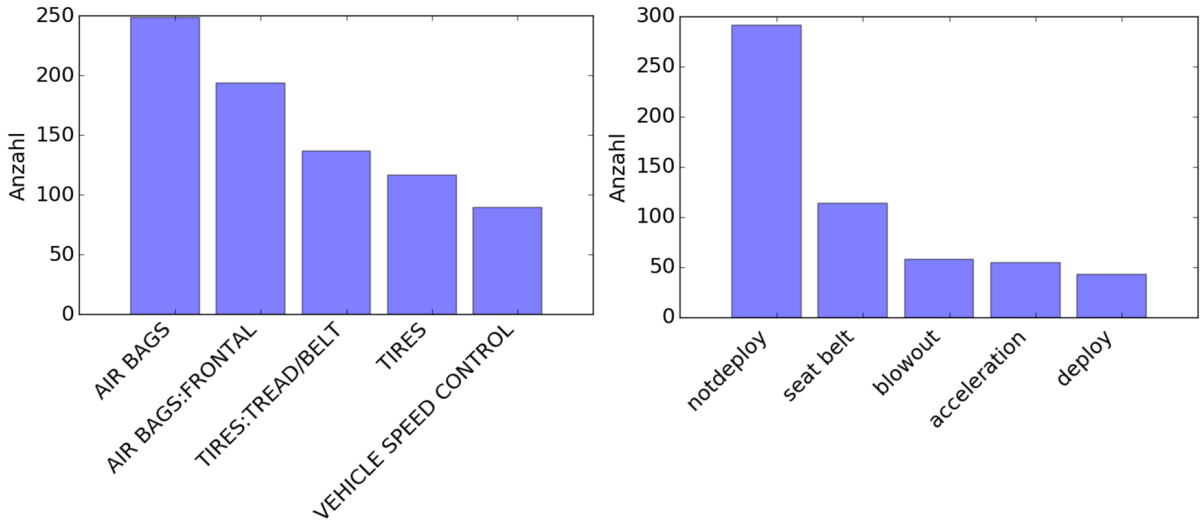

Abbildung 8.2: Häufigste Komponentenbeschreibungen und Cluster-Terme mit fatalen Folgen

Ein weiteres strukturiertes Feld in dem NHTSA-Datensatz gibt an, ob ein Fahrzeug in einem Feuer involviert war. Für die nächste Datenanalyse werden also nur jene Beschwerden analysiert, bei denen das strukturierte Feld bezüglich Feuer mit 'ja' gesetzt ist. Weiterhin werden nun auch die Felder mit den Hersteller- und Modellangaben des Autos benutzt, um die Häufigkeit der Komponentenbeschreibungen und Cluster-Terme in Verbindung mit der Häufigkeit der konkreten Fahrzeugmodelle zu bringen. Bei der ersten Betrachtung der Abbildung 8.3 fällt auf,

dass die häufigen Fahrzeugmodelle nicht dieselben sind je nachdem ob häufige Komponenten oder häufige Cluster-Terme betrachtet werden. Die Komponentenbeschreibungen zeigen, dass die Oberkategorien elektrische Systeme und Geschwindigkeitskontrolle des Fahrzeugs am häufigsten auftreten, wenn Fahrzeuge in Feuer involviert waren. Insbesondere häufig Erwähnung findet das Fahrzeugmodell Ford F-150. Durch die Betrachtung der Cluster-Terme können mehr Informationen gewonnen werden. So ist häufig von defekten Schaltern (‚switch') die Rede, die auch als Ursache für das Feuer gilt. Bei näherer Betrachtung geht es konkreter um die Schalter der Tempomate: „CRUISE CONTROL SWITCH CAUSED TRUCK TO CATCH FIRE." Auch das Fahrzeugmodell Ford Expedition hat wohl gehäuft diese Probleme. Der nächst häufige Cluster-Term in Bezug auf das Fahrzeugmodell Ford F-150 weist darauf hin, dass es häufig zu Feuerentwicklung unter der Motorhaube gekommen ist: „THE CONSUMER PARKED THE VEHICLE THEN FIVE MINUTES LATER BLACK SMOKE CAME FROM UNDER THE HOOD." Bei der Betrachtung der ausführlichen Komponentenbeschreibung ‚ELECTRICAL SYSTEM:WIRING:FRONT UNDERHOOD', zeigt sich jedoch, dass diese Informationen auch schon in den strukturierten Daten vorhanden ist. Die Fahrzeugmodelle Volkswagen Jetta und Passat tauchen zwar nicht unter den häufigsten Komponentenbeschreibung auf. Jedoch können durch den Cluster-Term 'heater' Informationen über mögliche Beschwerden mit der Heizung gewonnen werden. Bei der Betrachtung der Freitexte zeigt sich, dass es konkrete um die Sitzheizung geht, die zu Feuerentwicklungen führen: „WHILE DRIVING WITH THE SEAT HEATER TURNED ON, NOTICED THAT THE SEAT BECAME VERY HOT, AND IT STARTING FEELING LIKE IT WAS BURNING MY LEGS. SMELLED SMOKE, AND REALIZED THE SEAT WAS BURNING; (...)" Durch die Betrachtung einiger Komponentenbeschreibungen 'SEATS:FRONT ASSEMBlY:SEAT HEATER/COOLER' zeigt sich jedoch, dass diese Information auch in den strukturierten Daten enthalten. Allerdings enthalten auch einige nur die allgemein gehaltene Oberkategorie 'SEATS' als Komponentenbeschreibung. Diese könnten also mit der zusätzlichen Informationen aus den Freitexten ergänzt werden.

Wie in Abschnitt 7.1.1 erwähnt ist die NHTSA auch dazu berechtigt Rückrufaktionen von Autos zu veranlassen und bietet dementsprechend einen Datensatz[1] an, der Informationen über diese enthält. Für die nächste Datenanalyse werden die beiden Tabellen, 'Fahrzeugbeschwerden' und 'Rückrufaktionen', miteinander verknüpft. Damit kann beispielsweise der Frage nachgegangen werden, welche Komponentenbeschreibungen und Cluster-Terme häufig bei Rückrufaktionen Erwähnung finden. Hierzu wird wie in Listing 8.3 gezeigt ein Inner Join auf die Felder Fahrzeughersteller, Fahrzeugmodell, Baujahr und Komponentenbeschreibung durchgeführt. Das Feld mit dem Baujahr ist nur entscheidend, falls in der Rückrufaktion ein genaues Baujahr vermerkt ist.

Die Ergebnisse sind in der Abbildung 8.4 illustriert. Bei den häufigsten Cluster-Termen tauchen vor allem ‚unavailable' und ‚nhtsa campaign' auf. Die beiden Cluster-Terme verweisen auf bekannte Rückrufaktionen und drücken einen Missstand bei der Abwickelung der Reparaturen aus. Durch die Betrachtung der Cluster zeigt sich, dass unter anderem die Komponenten

[1] http://www-odi.nhtsa.dot.gov/downloads/

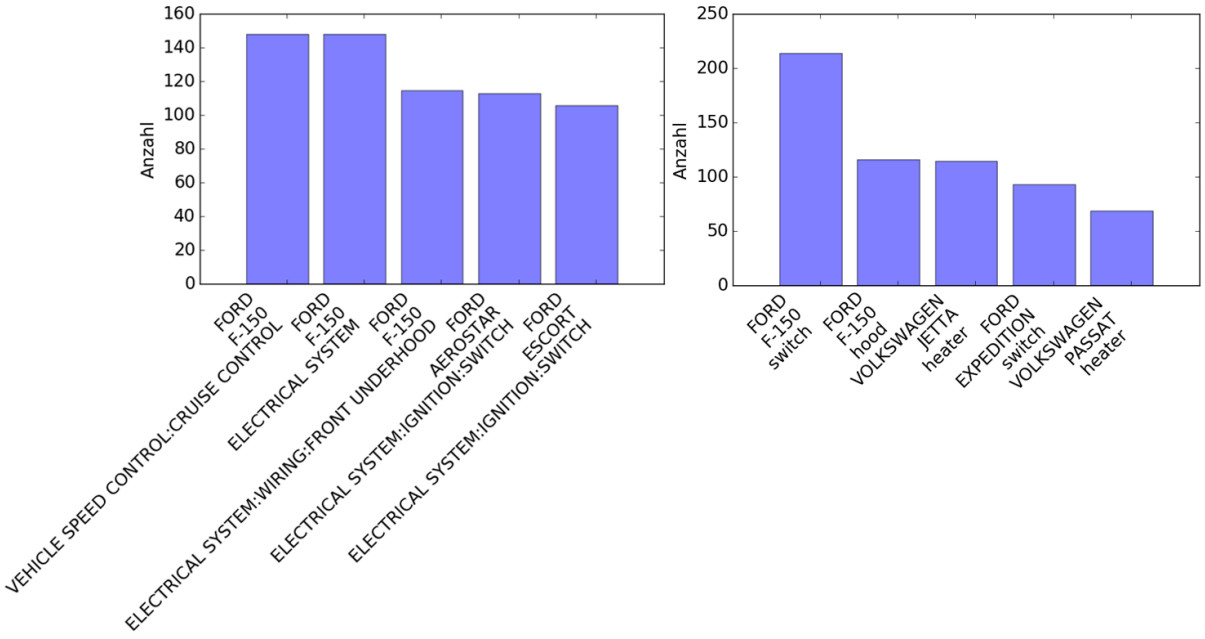

Abbildung 8.3: Häufigste Fahrzeugmodelle, die in Feuer involviert waren sowie deren Komponentenbeschreibungen und Cluster-Terme

Airbags (‚AIR BAGS') und Motor sowie Motorkühlung (‚ENGINE AND ENGINE COOLING') von Defekten bezüglich der Sensorik (‚sensor') betroffen sind: „PASSENGER SIDE AIRBAG SENSOR IN SEAT FAILED. 70K MILES." „THE CONSUMER RECEIVED A RECALL NOTICE (04V357000) FOR THE CAM SHAFT POSITION SENSOR." Auch die bereits vorgestellten Probleme mit dem Nichtauslösen (‚notedeploy') des Airbags und Probleme mit dem Getriebe (‚transmission') führen häufig zu Rückrufaktionen.

Weiter können noch die Gründe für Rückrufe spezifisch für Fahrzeugmodelle betrachtet werden, siehe Abbildung 8.5. Dabei werden zuerst die Beschwerden herausgefiltert, die die Cluster-Terme ‚unavailable' und ‚nhtsa campaign' aufweisen. Diese kommen nämlich häufig vor und beziehen sich auf Versäumnisse bei bereits bestehenden Rückrufaktionen von Fahrzeugen. Die verbleibende Daten enthalten häufige Beschwerden die das Fahrzeugmodell Ford Explorer bezüglich Radialreifen betreffen und hier wohl insbesondere das Reifenprofil (‚TIRES:TREAD/BELT '). Weiterhin häufig enthalten ist das Fahrzeugmodell Chevrolet Blazer bezüglich der hydraulischen Antiblockierbremsen (‚SERVICE BRAKES, HYDRAULIC:ANTILOCK '). Bezüglich des ersten Fahrzeugs gibt der Cluster-Term ‚tread separation' Aufschluss über die Rückrufaktion. Dabei handelt es sich um den bekannten Fall der spontanen Laufflächenabtrennung bei Firestone Reifen[2]: „TREAD SEPARATION AT 25MPH ON UNPAVED ROAD. (...)" Beim Chevrolet Blazer ist häufig der Cluster-Term ‚floor' vorzufinden. Erst bei Betrachtung der Beschwerdebeschreibungen wird die Bedeutung ersichtlich. Bei einer Betätigung der Bremsen

[2] http://www.stern.nyu.edu/om/faculty/zemel/ford_firestone.pdf

Listing 8.3 SQL-Befehl für die Verknüpfung der Beschwerden und der Rückrufe

```
SELECT FLAT_CMPL.COMPLID, FLAT_CMPL.MAKETXT, FLAT_CMPL.MODELTXT, FLAT_CMPL.YEARTXT,
    FLAT_CMPL.COMPDESC, FLAT_CMPL.CDESCR, FLAT_CMPL.ClusterMainterm, FLAT_CMPL.Distance,
    FLAT_CMPL.ClusterMainTermWeight, FLAT_RCL.RECORD_ID
FROM FLAT_CMPL INNER JOIN FLAT_RCL ON ((FLAT_CMPL.MAKETXT = FLAT_RCL. MAKETXT) AND
    (FLAT_CMPL.MODELTXT = FLAT_RCL.MODELTXT) AND (FLAT_CMPL.COMPDESC =
    FLAT_RCL.COMPNAME)) AND ((FLAT_CMPL.YEARTXT = FLAT_RCL.YEARTXT) OR FLAT_RCL.YEARTXT =
    9999) WHERE ClusterMainTerm NOT IN ('') AND ClusterMainTermWeight > 0.1;
```

geht das Bremspedal komplett bis zum Boden, jedoch findet keine Bremswirkung statt: „WHEN APPLYING THE BRAKES THE VEHICLE DOES NOT STOP, AND THE PEDAL GOES TO THE FLOOR. DEALER CAN'T FIND PROBLEM." Das Fahrzeugmodell Jeep Liberty ist dem Cluster-Term nach oft aufgrund von Fehlern des unteren Kugelgelenks (‚lower ball joint') zurückgerufen worden: „LOWER BALL JOINT FAILED WHILE DRIVING UNDER 15 MILES PER HOUR."

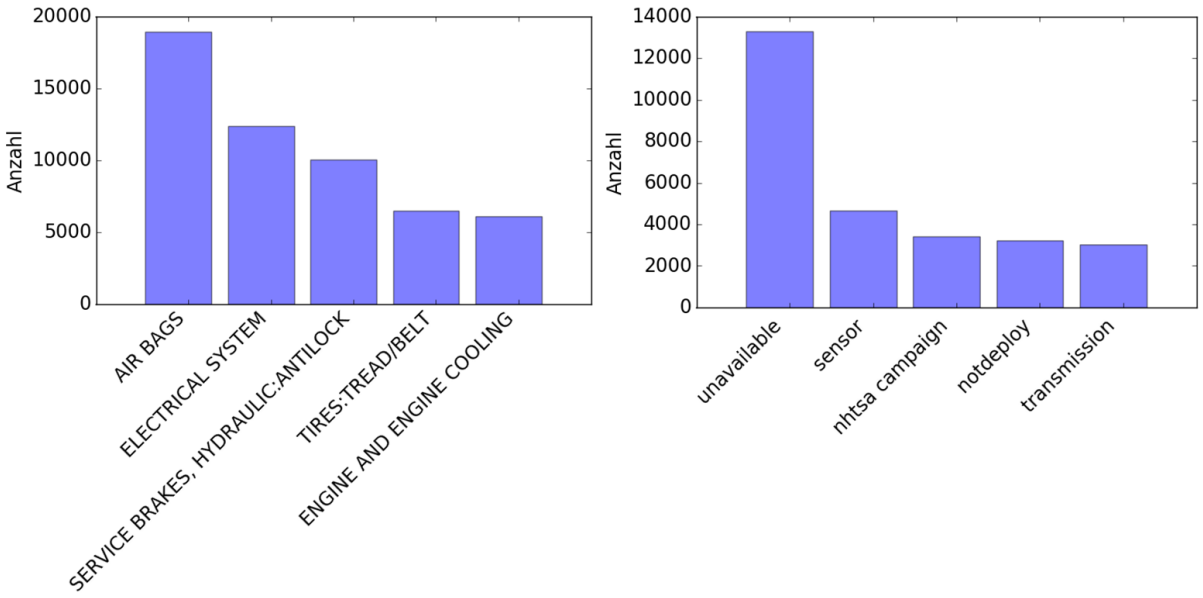

Abbildung 8.4: Häufigste Komponentenbeschreibungen und Cluster-Terme bei Rückrufen

Eine letzte Datenanalyse beschäftigt sich mit den zeitlichen Änderungen der Komponentenbeschreibungen beziehungsweise den Cluster-Termen. Hierzu wird zusätzlich das strukturierte Feld verwendet, dass das Datum des Vorfalls enthält. Es werden die vier häufigsten Cluster-Terme und die drei häufigsten Komponentenbeschreibungen aus den Jahren 2000, 2005, 2010 und 2015 betrachtet. Die Ergebnisse sind in Abbildung 8.6 zu sehen. Das auffälligste ist das enorme Aufkommen des Cluster-Terms ‚unavailable' mit einem Maximum im Jahr 2015. Ähnlich verhält es sich mit dem verwandten Cluster-Term ‚nhtsa campaign'. Dies lässt vermuten, dass das Problem mit den Reparaturen bei bestehenden Rückrufaktionen ein neueres Problem ist. Das Problem ist wohl unter anderem stark verbunden mit den Komponenten elektrisches

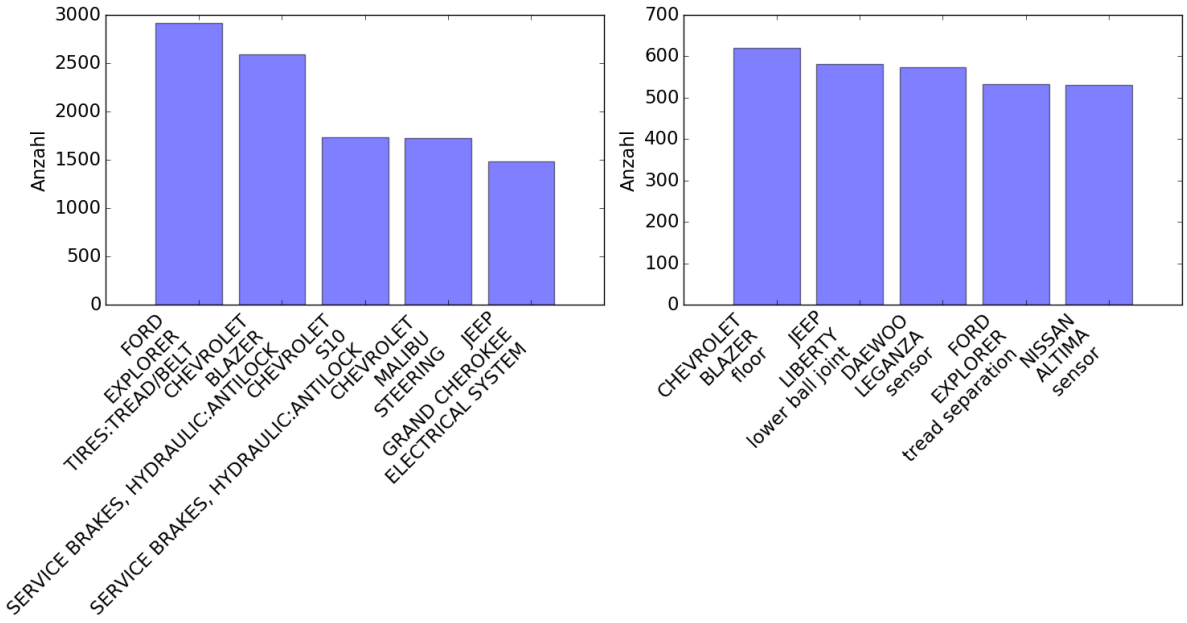

Abbildung 8.5: Häufigste rückgerufene Fahrzeuge und die Komponentenbeschreibungen sowie Cluster-Terme

System (‚ELECTRICAL SYSTEM') und Airbags (‚AIR BAGS'). Ebenso deutlich zu sehen ist das Aufkommen des Problems mit der Laufflächenablösung bei Reifen im Jahr 2000. Die darauf bezogenen Komponenten ‚TIRES' und ‚TIRES:TREAD/BELT' sind auch im Jahr 2000 stark angestiegen. Das Problem mit der Laufflächenablösung ist seit 2005 wohl nicht mehr aufgetreten. Im Jahr 2010 gibt es ein Maximum bezüglich Problemen mit der Tankuhr (‚gauge'). Diese Spitze ist wohl stark abhängig von der Komponente Benzin-Kraftstoffsystem (‚FUEL SYSTEM, GASOLINE'): „GAS GAUGE QUIT WORKING. WE NEED A RECALL."

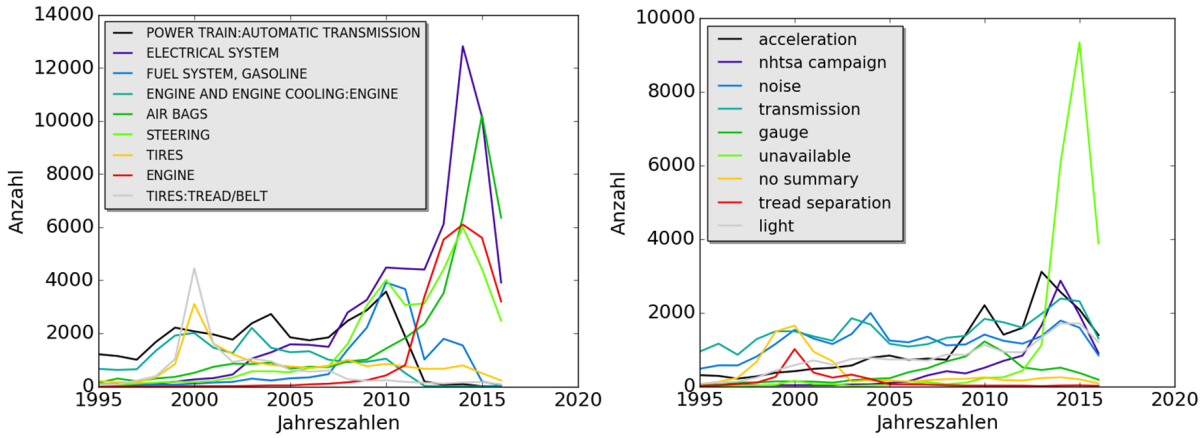

Abbildung 8.6: Zeitliche Änderung der häufigsten Komponentenbeschreibungen und Cluster-Terme aus den Jahren 2000, 2005, 2010 und 2015

Es zeigt sich, dass durch das automatische Clustering in Verbindung mit den Datenanalysen ergänzende Informationen zu den strukturierten Feldern gewonnen wurden. Darunter fallen Informationen über Beschwerden mit der Beschleunigung oder dass es zu ungewohnt lauten Geräuschen kam. Auch Probleme mit dem Getriebe fanden häufig Erwähnung. Des Weiteren betrifft eine gewonnene Information fehlende Reparaturteile für bestehende Rückrufaktionen. Insbesondere interessant ist auch der hohe Anteil an Nicht- beziehungsweise Fehlauslösungen von Airbags, die sehr häufig fatale Folgen haben. Defekte Schalter bei Tempomaten oder defekte Sitzheizungen können Auslöser für Brände bei Autos sein. Durch die Verknüpfung der Beschwerden mit den angeordneten Rückrufen konnte durch die Freitextfelder das bekannte Problem mit der Laufflächenabtrennung festgestellt werden. Auch weisen rückgerufene Autos häufig Fehler mit den Sensoren oder eine schlechte Bremskraft auf. Zeitliche Trends wie etwa häufigere Erwähnungen von defekten Tankuhren können ebenso festgestellt werden. Bezüglich den defekten Sitzheizungen zeigt sich allerdings eine mögliche Verbesserung des entwickelten Prototyps. Da für die Erstellung der vereinfachten Komponentenbeschreibungen (siehe Abschnitt 6.4) und für die Stoppwörter lediglich die Oberkategorien zum Einsatz kamen, wurden genauer spezifizierte Informationen aus den Komponentenbeschreibungen hierfür ignoriert.

8.2 Industriedatensatz

Für die Analysen der Industriedaten werden die speziellen Fehlercodes sowie die vom Prototyp hinzugefügten Cluster-Terme (siehe Abschnitt 6.8) durch abstraktere Begriffe ersetzt. Dies sorgt zum einen für eine bessere Verständlichkeit, da spezifische Begriffe aus der Fertigung vereinfacht werden und zum anderen sorgt dies für die Sicherstellung der Datenvertraulichkeit. Wie bei der Datenanalyse des NHTSA-Datensatzes in Abschnitt 7.1.1 werden genauso auch hier Cluster, bestehend aus einem leeren String oder deren Cluster-Terme eine niedrige Gewichtung aufweisen entfernt.

Für die erste Analyse werden häufig auftretende Fehlercodes und häufig auftretende Cluster-Terme miteinander verglichen. Die in Abbildung 8.7 auftretenden Fehlercodes sind sehr grobgranular gehalten und lassen nur wenig Schlüsse auf die genauen Ursachen für die Stillstände auf der Fertigungslinie zu. Die häufigste Fehlerkategorie 'Sonstiges' liefert keine zusätzlichen Information bezüglich des Stillstands. Des Weiteren sind durch die in den strukturierten Daten enthaltenen Fehlercodes wenn überhaupt nur allgemein gehaltene Informationen bekannt, dass beispielsweise Probleme durch fehlerhafte Zukaufteile, das Rüsten, den Wechsel von Werkzeugen sowie Kalibrierungen häufig ursächlich für Stillstände sind. Indes weisen die vom Prototyp erstellten Cluster präzisere Informationen auf. Durch diese wurden insbesondere elektrische Prüfungen, aber auch Wiederholprüfungen als sehr häufige Ursache für Stillstände identifiziert. Auch einzelne Gehäusekomponenten eines Produkts, Parameterabweichungen und Probleme mit einem Sonderprozess werden als häufige Gründe ausgemacht.

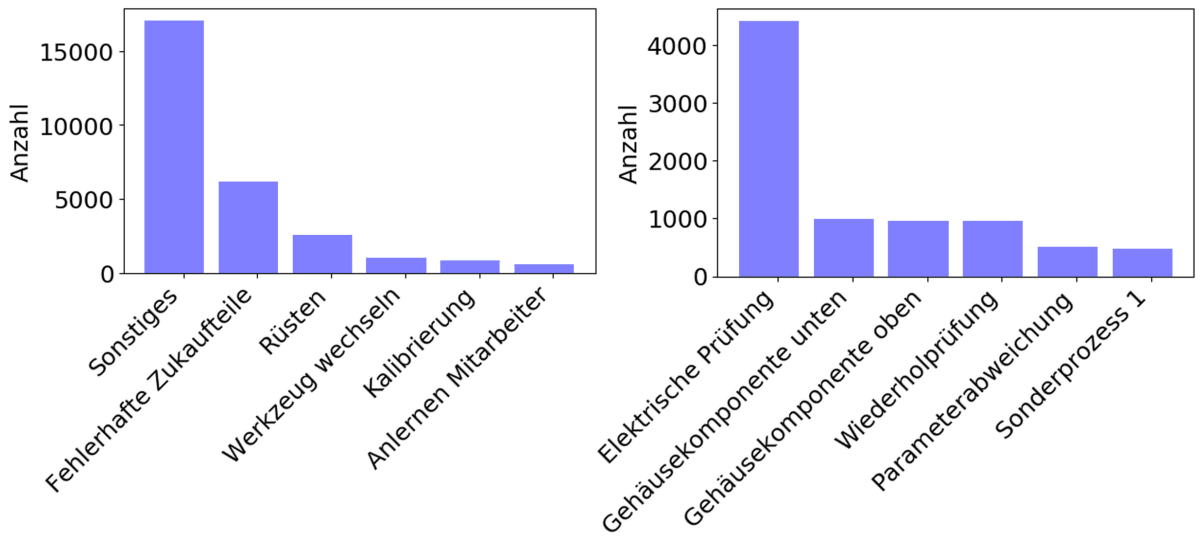

Abbildung 8.7: Häufig auftretende Fehlercodes und häufig auftretende Cluster-Terme in den Industriedaten

Für jeden Fehlercode können gegebenenfalls noch spezifische Informationen über die vom Prototyp erstellten Cluster gewonnen werden. So zeigen etwa die Cluster in 'Kalibrierung' Probleme an zwei spezifischen Sonderprozessen auf. Innerhalb der Fehlerkategorie 'Rüsten' (vergleiche Abbildung 8.7) können mithilfe des Clustering einzelne konkrete Rüstvorgänge identifiziert werden. Und auch zu nichtssagenden Fehlerkategorien wie 'Sonstiges' (vergleiche Abbildung 8.7) können weitere Informationen über die erstellten Cluster gewonnen werden, wie etwa dass Probleme mit Einzelteilen und außer Betrieb gesetzten Maschinen zu Stillstandszeiten auf der Produktionslinie führen. Auch bezüglich 'Fehlerhafte Zukaufteile' können weitere Informationen aufgrund der Cluster entdeckt werden: Häufig sind innerhalb dieser Fehlerkategorie Probleme mit einem Prüfparameter in der elektrischen Prüfung, Probleme mit verklemmten spezifisch benannten Teilen sowie häufige Wiederholprüfungen.

Abbildung 8.8 zeigt die Ergebnisse einer weiteren Datenanalyse. Hierfür wurde zusätzlich die benötigte Zeit für die Behebung eines Stillstands mit einbezogen. Damit wurde für jeden Fehlercode und für jeden Cluster-Term die durchschnittliche aufgewendete Dauer für die Behebung berechnet. Tabelle 8.1 zeigt für den analysierten Datensatz die durchschnittliche, minimale und maximale Stillstandsdauer über alle Fehlercodes und Cluster-Terme. Einzelne Cluster-Terme und Fehlercodes, die insgesamt seltener als fünf mal auftauchen, wurden aus den Ergebnissen herausgefiltert. Während die strukturierten Informationen nicht näher spezifizierte Wechsel ('XX Wechsel'), Defekte in Fixierung, Antrieb und Adaptierung sowie die Instandhaltung und defekte Lichtschranken als Gründe für lange Stillstandszeiten anzeigen, weisen die identifizierten Cluster auf detailliertere Prozesse hin. So zeigt die Analyse etwa, dass die Durchführung spezifischer Rüstvorgänge durchschnittlich viel Zeit beansprucht. Außerdem ist ersichtlich, dass ein spezifischer Sonderprozess für besonders viel Stillstand sorgt. Auch ist

Anzahl Einträge	minimale Dauer in Sekunden	maximale Dauer in Sekunden	durchschnittliche Dauer in Sekunden
39770	0	4318	600

Tabelle 8.1: Überblick über die Stillstandszeiten

der durchschnittliche Stillstand aufgrund von verschlissenen Zuführschienen hoch. Weiterhin ist der zeitliche Aufwand für die Justierung eines spezifischen Teils durchschnittlich hoch.

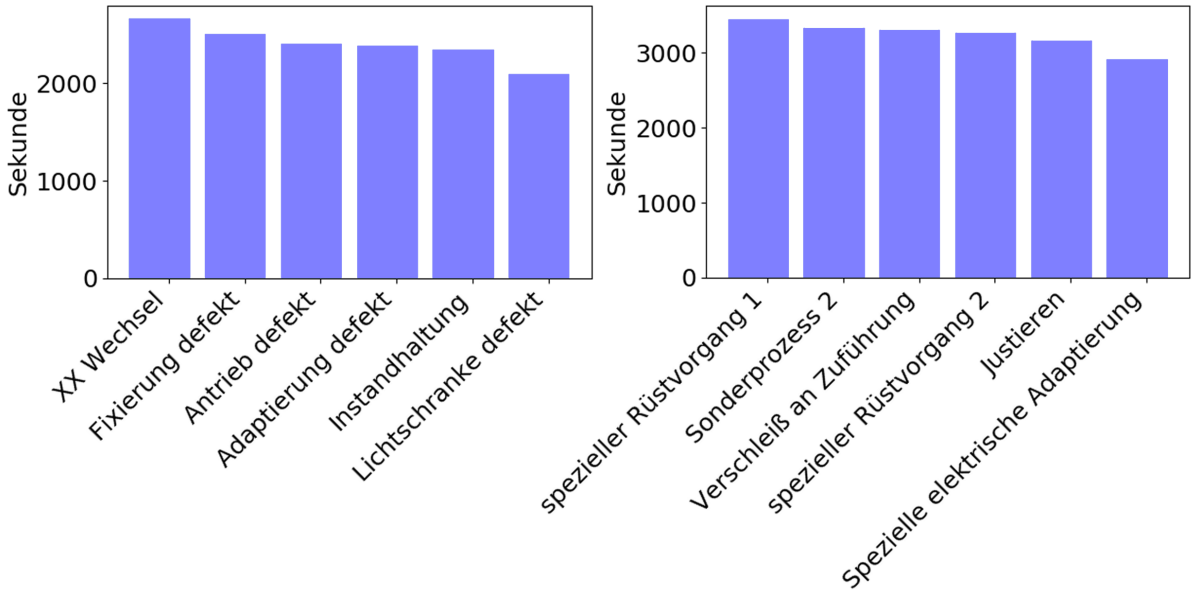

Abbildung 8.8: Durchschnittliche aufgewendete Zeit für die Behebung eines Stillstands je Fehlercode und je Cluster-Term

Abschließend hat sich gezeigt, dass durch die Anwendung des Prototyps auf den Industriedatensatz zusätzliche Informationen aus den Freitexten gewonnen werden konnten. Durch die hinzugefügten Cluster-Terme konnten spezifische Sonderprozesse ausgemacht werden, die mit einigen Fehlercodes in Verbindung stehen. Auch hat der Prototyp spezifische Rüstvorgänge erkannt, die ergänzende Informationen zum allgemein gehaltenen Code liefern. Spezifische Prüfungen, die durchgeführt worden sind, konnten ebenso mittels den Freitextfeldern erschlossen werden. Einzelne Komponenten, bezüglich eines Produkts innerhalb einer Fertigungslinie, die Probleme aufweisen, konnten ebenso ausfindig gemacht werden. Zusammenfassend bestätigen diese Ergebnisse, dass die Vollständigkeit der in den Daten enthaltenen Informationen mithilfe eines Erschließens der Freitextfelder verbessert werden kann.

9 Zusammenfassung & Ausblick

In dieser Ausarbeitung wurde ein Prototyp für das automatisches Clustering von Freitexten entwickelt. Dieser bezieht auch strukturierte Informationen mit ein, um vorab eine Gruppierung des Datensatzes durchzuführen und bietet eine Schnittstelle für den Benutzer, um den gesamten Ablauf des Clusteringprozess einfach definieren zu können. Mit dem Prototyp konnte durch die Definierung von Stoppwortlisten die in den strukturierten Feldern enthaltenen Informationen herausgefiltert werden, damit die neuen, in den Freitextfeldern enthalten Informationen, nicht verdeckt werden.

Weiterhin konnte der entwickelte Prototyp erfolgreich auf zwei unterschiedliche Datensätze, die sowohl strukturierte, als auch unstrukturierte Informationen besitzen, angewendet werden. Die durch den Prototyp erschlossenen Informationen, wurden mittels Datenanalysen untersucht. Dadurch konnte festgestellt werden, dass für beide Datensätze mithilfe des Prototypen aus den Freitextfeldern zusätzliche Informationen gewonnen wurden, die nicht in den strukturierten Daten enthalten sind. Es konnte gezeigt werden, das eine automatisierte Auswertung der unstrukturierten Informationen zu vollständigeren Daten und damit zu einer erhöhten Datenqualität führt.

Die untersuchten Datensätze wiesen beide domänenspezifische Informationen auf mit vielerlei Fachtermini und Abkürzungen. Für den Clusteringprozess wurden viele Begriffe und Abkürzungen auf die Stoppwortliste gesetzt sowie Synonyme für diese definiert. Es ist anzunehmen, dass aufgrund von fehlendem domänenspezifischen Wissen, Informationen aus den unstrukturierten Daten verloren gegangen sind. Zukünftig wäre also durch insbesondere verbesserte Vorverarbeitungsschritte, beispielsweise durch Verwendung von Taxonomien oder Ontologien, eine Verbesserung des Clusteringprozesses denkbar und damit ein erhöhter Informationsgewinn erzielbar. Auch interessant wäre es die Auswirkung durch Verwendung anderer Clustering-Algorithmen zu untersuchen. In Bezug auf den NHTSA-Datensatz wäre eine bessere Nutzung der Komponentenbeschreibung, durch hinzunehmen der Unterkategorien möglich.

Literaturverzeichnis

[1] C. Batini, M. Scannapieco, *Data and Information Quality: Dimensions, Principles and Techniques*. Springer, 2016.

[2] C. Kiefer, „Unlocking information from free text fields“, Unveröffentlichter Bericht des Instituts für Parallele und Verteilte Systeme, Stuttgart, 2016.

[3] ——, „Assessing the quality of unstructured data: an initial overview“, in *Proceedings of the LWDA 2016 Proceedings (LWDA)*, (Potsdam, Germany, 12. Sep. 2016), R. Krestel, D. Mottin, E. Müller, Hrsg., Ser. CEUR Workshop Proceedings, Aachen, 2016, S. 62–73. Adresse: http://ceur-ws.org/Vol-1670/#paper-25.

[4] R. Feldman, J. Sanger, *The text mining handbook: advanced approaches in analyzing unstructured data*. Cambridge University Press, 2007.

[5] P. Jackson, I. Moulinier, *Natural language processing for online applications: Text retrieval, extraction and categorization*. John Benjamins Publishing, 2007, Bd. 5.

[6] J. H. M. Daniel Jurafsky, *Speech and Language Processing*, 2. Aufl. Prentice Hall, 2008.

[7] Y. Huang, „Intelligent typo correction and text categorization using machine learning and ontology networks“, Diss., University of Michigan-Dearborn, 2014.

[8] D. L. Lee, H. Chuang, K. Seamons, „Document ranking and the vector-space model“, *IEEE software*, Bd. 14, Nr. 2, S. 67–75, 1997.

[9] S. Robertson, „Understanding inverse document frequency: on theoretical arguments for idf“, *Journal of documentation*, Bd. 60, Nr. 5, S. 503–520, 2004.

[10] J. A. Hartigan, *Clustering algorithms*, Ser. Wiley series in probability and mathematical statistics. New York: J. Wiley & Sons, 1975, ISBN: 0-471-35645-X.

[11] P. J. Rousseeuw, „Silhouettes: a graphical aid to the interpretation and validation of cluster analysis“, *Journal of computational and applied mathematics*, Bd. 20, S. 53–65, 1987.

[12] B. Carter, M. Hofmann, „An analysis into using unstructured non-expert text in the illicit drug domain“, in *Advance Computing Conference (IACC), 2014 IEEE International*, Feb. 2014, S. 651–657.

[13] K. E. Kumar, H. A. Ahmed, „Estimation of traffic with accuracy through twitter stream analysis“, *International journal of innovative technologies*, Bd. 4, Nr. 8, S. 1317–1324, 2016.

[14] A. Jalaparthi, A. S. Kumar, „Monitoring and analysis of real time detection of traffic from twitter stream analysis“, *International journal of scientific research in computer science and engineering*, Bd. 4, Nr. 03, S. 30–36, 2016.

[15] E. D'Andrea, P. Ducange, B. Lazzerini, F. Marcelloni, „Real-time detection of traffic from twitter stream analysis“, *IEEE Transactions on Intelligent Transportation Systems*, Bd. 16, Nr. 4, S. 2269–2283, Aug. 2015, ISSN: 1524-9050.

[16] S. Bhosale, S. Kokate, „Traffic detection using tweets on twitter social network“, *International Journal*, Bd. 4, Nr. 7, 2016.

[17] A. Srivastava, B. Zane-Ulman, „Discovering recurring anomalies in text reports regarding complex space systems“, in *IEEE Aerospace Conference*, 2005, S. 37.

[18] M. Gamon, A. Aue, S. Corston-Oliver, E. Ringger, „Pulse: mining customer opinions from free text“, in *Advances in Intelligent Data Analysis VI: 6th International Symposium on Intelligent Data Analysis, IDA 2005, Madrid, Spain, September 8-10, 2005. Proceedings*, A. F. Famili, J. N. Kok, J. M. Peña, A. Siebes, A. Feelders, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, S. 121–132, ISBN: 978-3-540-31926-9.

[19] B. Brooks, „Shifting the focus of strategic occupational injury prevention: mining free-text, workers compensation claims data“, *Safety Science*, Bd. 46, Nr. 1, S. 1–21, 2008, ISSN: 0925-7535.

[20] N. Alsaedi, P. Burnap, „Arabic event detection in social media“, in *Computational Linguistics and Intelligent Text Processing: 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part I*, A. Gelbukh, Hrsg. Cham: Springer International Publishing, 2015, S. 384–401, ISBN: 978-3-319-18111-0.

[21] C.-H. Lee, „Mining spatio-temporal information on microblogging streams using a density-based online clustering method“, *Expert Systems with Applications*, Bd. 39, Nr. 10, S. 9623–9641, 2012, ISSN: 0957-4174.

[22] M. Ghazizadeh, A. D. McDonald, J. D. Lee, „Text mining to decipher free-response consumer complaints: insights from the nhtsa vehicle owner's complaint database“, *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 2014.

[23] A. S. Abrahams, J. Jiao, G. A. Wang, W. Fan, „Vehicle defect discovery from social media“, *Decision Support Systems*, Bd. 54, Nr. 1, S. 87–97, 2012, ISSN: 0167-9236.

[24] P.-N. Tan, H. Blau, S. Harp, R. Goldman, „Textual data mining of service center call records“, in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Ser. KDD '00, Boston, Massachusetts, USA: ACM, 2000, S. 417–423, ISBN: 1-58113-233-6.

[25] N. Jakob, S. H. Weber, M. C. Müller, I. Gurevych, „Beyond the stars: exploiting free-text user reviews to improve the accuracy of movie recommendations“, in *Proceedings of the 1st International CIKM Workshop on Topic-sentiment Analysis for Mass Opinion*, Ser. TSA '09, Hong Kong, China: ACM, 2009, S. 57–64, ISBN: 978-1-60558-805-6.

[26] A. Medem, M. I. Akodjenou, R. Teixeira, „Troubleminer: mining network trouble tickets“, in *Integrated Network Management-Workshops, 2009. IM '09. IFIP/IEEE International Symposium on*, Juni 2009, S. 113–119.

[27] R. Chougule, D. Rajpathak, P. Bandyopadhyay, „An integrated framework for effective service and repair in the automotive domain: an application of association mining and case-based-reasoning“, *Computers in Industry*, Bd. 62, Nr. 7, S. 742–754, 2011, ISSN: 0166-3615.

[28] R. Nayak, N. Piyatrapoomi, J. Weligamage, „Application of text mining in analysing road crashes for road asset management“, in *Engineering Asset Lifecycle Management: Proceedings of the 4th World Congress on Engineering Asset Management (WCEAM 2009), 28-30 September 2009*, D. Kiritsis, C. Emmanouilidis, A. Koronios, J. Mathew, Hrsg. London: Springer London, 2010, S. 49–58, ISBN: 978-0-85729-320-6.

[29] T. Nasukawa, T. Nagano, „Text analysis and knowledge mining system“, *IBM Systems Journal*, Bd. 40, Nr. 4, S. 967–984, 2001, ISSN: 0018-8670.

[30] G. Forman, E. Kirshenbaum, J. Suermondt, „Pragmatic text mining: minimizing human effort to quantify many issues in call logs“, in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Ser. KDD '06, Philadelphia, PA, USA: ACM, 2006, S. 852–861, ISBN: 1-59593-339-5.

[31] K. J. Millman, M. Aivazis, „Python for scientists and engineers“, *Computing in Science & Engineering*, Bd. 13, Nr. 2, S. 9–12, 2011.

[32] M. Foord, N. Larosa, R. Dennis, E. Courtwright, *Configobj 5 introduction and reference*, Aug. 2014. Adresse: https://configobj.readthedocs.io/en/latest/configobj.html.

[33] S. Tosi, *Matplotlib for Python developers*. Packt Publishing Ltd, 2009.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., „Scikit-learn: machine learning in python“, *Journal of Machine Learning Research*, Bd. 12, Nr. Oct, S. 2825–2830, 2011.

[35] S. Van Der Walt, S. C. Colbert, G. Varoquaux, „The numpy array: a structure for efficient numerical computation“, *Computing in Science & Engineering*, Bd. 13, Nr. 2, S. 22–30, 2011.

[36] *Python odbc bridge*. Adresse: http://mkleehammer.github.io/pyodbc/.

[37] S. Bird, „Nltk: the natural language toolkit“, in *Proceedings of the COLING/ACL on Interactive presentation sessions*, Association for Computational Linguistics, 2006, S. 69–72.

[38] B. DeWilde, *Textacy: higher-level nlp built on spacy*. Adresse: http://textacy.readthedocs.io/en/latest/.

[39] *Textblob: simplified text processing*. Adresse: https://textblob.readthedocs.io/en/dev/.

[40] M. Pecht, A. Ramakrishnan, J. Fazio, C. E. Nash, „The role of the u.s national highway traffic safety administration in automotive electronics reliability and safety assessment“, *IEEE Transactions on Components and Packaging Technologies*, Bd. 28, Nr. 3, S. 571–580, Sep. 2005, ISSN: 1521-3331.

[41] *National highway traffic safety administration, office of defects investigation. (2016). vehicle owner's complaint database.* Adresse: http://www-odi.nhtsa.dot.gov/downloads/.

Alle URLs wurden zuletzt am 16. 01. 2017 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift