UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4

AGKI
artificial intelligence research koblenz

Entwicklung eines Klassifizierungswerkzeugs für Spam und Ham

Studienarbeit
im Studiengang Informatik

vorgelegt von:
Michael Schulze
Matrikelnummer: 203210847
E-Mail: schulzemich@uni-koblenz.de

Betreuer: Prof. Dr. Ulrich Furbach, Institut für Informatik
Dipl.-Inf. Markus Maron, Institut für Informatik
Erstgutacher: Prof. Dr. Ulrich Furbach, Institut für Informatik
Zweitgutacher: Dipl.-Inf. Markus Maron, Institut für Informatik

Koblenz, den 3. März 2008

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden. Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.

Koblenz, den 3. März 2008

Michael Schulze

Inhaltsverzeichnis

Abbildungsverzeichnis

Zusammenfassung

In dieser Arbeit werden einige bekannte Spam Klassifizierungsmethoden, welche für viele Anwendungen schon im Einsatz sind, kurz erläutert, um sie in einem neuen Szenario zu analysieren. In diesem Szenario wird nicht wie bei E-Mails üblich davon ausgegangen, dass die Nachrichten unbegrenzter Länge sein können, sondern dass es sich hierbei um Nachrichten begrenzter Länge handelt. Auch die Darstellungsform und der vermutliche Inhalt solcher Nachrichten werden aus einem speziellen Kontext heraus betrachtet. So wird davon ausgegangen, dass die Nachrichten auf einem Ticker eines Bildschirmes zu sehen sind, welcher in einer Universität angebracht ist. Somit werden die Nachrichteninhalte sich eher auf das universitäre Umfeld beziehen, wobei angenommen wird, dass die Nachrichteninhalte nur von einer abgeschlossenen Gruppe von Personen eingestellt werden.
Nach der Erzeugung einiger Spam- und Hamnachrichten, die auf die Kriterien des Szenarios zutreffen, werden Klassifizierungsmethoden evaluiert. Am Ende der Analyse folgt eine Diskussion über die Verwendbarkeit jener Methoden in diesem Szenario. Abgeschlossen wird diese Untersuchung von einem Programm, welches die entsprechenden Methoden zur Klassifizierung implementiert.

1 Einleitung und Motivation

Die Übertragung von Nachrichten geschieht in immer vielfältigerer Weise und dient den unterschiedlichsten Zwecken. Wurden früher Nachrichten in Briefform verfasst und dann an den entsprechenden Empfänger gesendet, so war das eine mit Kosten verbundene Methode. Ob Werbenachrichten so versendet werden sollten, musste demnach gut durchdacht werden. Heute stellt sich im Bereich der E-Mails eine andere Situation dar, hier ist die Übertragung einer Nachricht nahezu kostenfrei.
Da unerwünschte Nachrichten immer mit Kosten für den Empfänger verbunden sind, wurden Techniken zur Vermeidung solcher Nachrichten entwickelt, welche jede mögliche Information, die aus einer E-Mail zu gewinnen ist, ausnutzen. Nach den ersten 'Key- oder Badwordlisten' folgte bald die Anwendung statistischer Filter. [12]
Weitere Bereiche sind vorstellbar, in denen nur gewisse Nachrichten erwünscht sind, jedoch die Form, sei es die Art der Übertragung oder die Beschränkung des Inhaltes auf einfache Buchstaben (hier im Gegensatz zu HTML Inhalten gemeint), von einer E-Mail stark abweicht.
Ein solches etwas spezielleres Szenario wird in dieser Arbeit analysiert. Um diesen Gegenstand weiter betrachten zu können, wird an dieser Stelle eine kurze Situationsbeschreibung vorweg genommen. Motiviert durch ein Projekt an der Universität Koblenz, bei dem es unter anderem darum ging, Nachrichten als Ticker auf Bildschirmen zu präsentieren, eingestellt durch eine kleine Gruppe, soll in Zukunft jeder an der Universität eingeschriebene Student Nachrichten an die Bildschirme senden dürfen. So sind in dem von uns gestellten Szenario die Nachrichten i.A. nicht an eine bestimmte Person gerichtet und die Anzahl der Buchstaben einer Nachricht begrenzt. Ebenso der Inhalt ist einfacher textueller Natur. Die Nachrichten können dann von einer großen Gruppe eingestellt werden, die jedoch zahlenmäßig begrenzt ist. Eine von dieser Gruppe eingestellte Nachricht ist aber auch über die begrenzte Gruppe hinaus für die Öffentlichkeit sichtbar.
Nun kann man sich gut vorstellen, dass nicht jede Nachricht über den Ticker laufen soll. Welche Nachrichten unerwünscht, also Spamnachrichten sind und welche Nachrichten erwünscht, also Hamnachrichten sind, bestimmt der Betreiber des Systems. So kann man sich leicht vorstellen, dass gewisse Nachrichten bei einem Betreiber Spam wären, aber bei einem anderen Betreiber Ham.
Welche Methoden nun zur Klassifizierung der Texte in die Klassen Spam und Ham vorgesehen sind, wird unter anderem Anhand einer Analyse dieser speziellen Situation entschieden und begründet. Die Analyse bildet die Grundlage für das entsprechende Programm, welches in Abschnitt 4 beschrieben wird.

Zu Beginn dieser Arbeit werden erst die theoretischen Grundlagen, welche zur späteren Analyse der Klassifizierer benötigt werden, erklärt. Dann folgen Überlegungen über Ham- und Spamtexte. Aufgrund dieser Überlegungen wird ein erster Datensatz mit Spam- und Hamtexten erstellt und eine Auswahl an Klassifizierern getroffen. Diese ausgewählten Klassifizierer werden im Weiteren für ein abschließendes Programm analysiert. Nach Vorstellung dieses Programms wird dieser auch aufgrund neuer Daten evaluiert.

2 Grundlagen

In diesem Kapitel sind die theoretischen Grundlagen für das Verständnis der späteren Analyse zu finden. Es wird versucht bei der Erklärung auf mathematische Details zu verzichten und nur ein allgemeines Verständnis für die einzelnen Verfahren zu geben. Um eine vertiefte Einsicht in die hier beschriebenen Grundlagen zu bekommen wird auf die Literatur verwiesen.

2.1 Merkmalsvektor

In einem Merkmalsvektor werden die verschiedenen Merkmale eines Objekts repräsentiert. Jede Dimension entspricht einem Merkmal. Meist beschreibt eine natürliche Zahl dieses Merkmal. Da geschriebener Text analysiert werden soll, werden als Merkmale des Mermalsvektors die Vorkommenshäufigkeiten der Wörter des Textes als Merkmale definiert. Wie die Wörter aus einem Text generiert werden, sei hier noch nicht genauer erklärt.
Es wird grundsätzlich zwischen zwei Arten von Merkmalsvektoren unterschieden. Zum einen die binären Merkmalsvektoren. Hier werden die einzelnen Merkmale auf eine null oder eins abgebildet. Eine eins beschreibt, dass ein bestimmtes Merkmal eines Objektes vorhanden ist, eine null beschreibt, dass ein bestimmtes Merkmal nicht vorhanden ist. Auf diesen Fall angewandt, kommt ein bestimmtes Wort in dem Text vor oder nicht.
Zum anderen kann der Merkmalsvektor auch mächtiger gestaltet werden. So kann in jede natürliche Zahl jeder Dimension auch noch mehr Information über dieses Merkmal modelliert oder codiert werden. Soll beispielsweise der Preis eines Fahrzeugs in einem Merkmalsvektor modellieren werden, könnte eine natürliche Zahl den Preis des Autos in Cent in der entsprechenden Dimension beschreiben.
Im Weiteren werden aufgrund der Vorkommenshäufigkeit von Worten die Merkmalsvektoren konstruiert. Hierzu ein kurzes Beispiel:
Es soll eine Instanz des oben beschriebenen Merkmalsvektors, in dem die Vorkommenshäufigkeit der Worte als Merkmal auftritt, aus folgendem Text gebildet werden:

```
Wenn Fliegen hinter Fliegen fliegen fliegen Fliegen
Fliegen hinterher.
```

Groß- und Kleinschreibung werden außer Acht gelassen, demnach besteht der Merkmalsvektor aus vier Merkmalen (vier verschiedene Wörter, Punkt außer Acht gelas-

sen). Der Merkmalsvektor hat die Dimension vier. Die Instanzen sehen wie folgt aus: $\begin{pmatrix} 1 \\ 6 \\ 1 \\ 1 \end{pmatrix}$ und binär: $\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$. [6]
Als Merkmalsvektorraum wird ein Vektorraum mit der Dimension des Merkmalsvektors bezeichnet.

2.2 Bayessche Klassifikation

Die wohl am Häufigsten benutzte und am weitesten verbreitete Klassifizierungsmethode für die Textklassifizierung von Spam und Ham E-Mails ist 'Naive Baise'. Jedes Merkmal, hier die Worthäufigkeit, wird unabhängig von allen anderen Merkmalen betrachtet. Es wird die Vorkommenswahrscheinlichkeit jedes Merkmals in Spam bzw. Ham klassifizierten Datensätzen, also der Trainingsdaten, zugrunde gelegt, um die Gesamtwahrscheinlichkeit einer Klassenzugehörigkeit zu errechnen.
Das 'Naive Baise' Wahrscheinlichkeitsmodell sieht wie folgt aus:

$$P(A|M_1, M_2, ..., M_n) = \frac{P(A) * P(M_1, M_2, ...M_n|A)}{P(M_1, M_2, ..., M_n)}$$

wobei A eine Klassenvariable ist, in dem Fall ist $A \in \{Spam, Ham\}$. $M_1, M_2, ..., M_n$ sind Merkmale, also die Dimensionen des zu klassifizierenden Merkmalvektors. Da der Nenner, sowohl für $A = Spam$, als auch für $A = Ham$ identisch ist, kann dieser in der weiteren Berechnung vernachlässigt werden, da nur das Verhältnis interessant ist, demnach welche Wahrscheinlichkeit größer ist. Mittels der bedingten Wahrscheinlichkeitsregeln nach dem Bayes Theorem und der Annahme, dass jedes Merkmal unabhängig von den anderen Merkmalen auftritt, kann das Wahrscheinlichkeitsmodell wie folgt dargestellt werden:

$$P(A|M_1, M_2, ..., M_n) =$$

$$P(A) * P(M_1, M_2, ..., M_n|A) =$$

$$P(A) * P(M_1|A) * P(M_2|A, M_1) * ... * P(M_n|A, M_1, M_2, ..., M_{n-1}) =$$

$$P(A) * P(M_1|A) * P(M_2|A) * ... * P(M_n|A)$$

Die Annahme, dass jedes Merkmal des Merkmalvektors unabhängig von den anderen Merkmalen auftritt, bedeutet in dem Fall, sofern die Merkmale die Häufigkeiten des Vorkommens eines ganz bestimmten Wortes sind, dass jedes Wort unabhängig von den anderen Wörtern eines Textes auftritt. Dies ist ein sehr stark vereinfachtes Modell für die Klassifizierung natürlich sprachlicher Texte, jedoch hat sich diese Methode in der Praxis bewährt.
Um die Wahrscheinlichkeit des Modells errechnen zu können, wird die sogenannte Prior Probability $P(A)$ benötigt. Diese kann aus den Trainingsdaten folgendermaßen

errechnet werden:

$$P(a) = \frac{Anzahl\ der\ Trainingsdaten\ der\ Kategorie\ a}{Anzahl\ der\ gesamten\ Trainingsdaten}$$

mit $a \in \{Ham, Spam\}$. Die Wahrscheinlichkeit eines Merkmals, die Frequenz eines Wortes in einem Text, unter der Bedingung $a \in \{Ham, Spam\}$ berechnet man, indem man die Frequenz des zugrunde liegenden Wortes des Merkmals aller a kategorisierten Texte durch die Anzahl aller Wörter der Kategorie a teilt, also

$$P(m|a) = \frac{Frequenz\ von\ m\ in\ den\ a\ kategorisierten\ Trainingsdaten}{Alle\ Woerter\ der\ Kategorie\ a}$$

. Die Daten werden nun wie folgt klassifiziert:

$$max_a(P(a)\Pi_{i=1}^{n} P(f_i|a_i))$$

mit $a \in \{Ham, Spam\}$ und $i, n \in$ N. [3] [10]
Der Bayessche Klassifizierer, wie er in Paul Grahams Paper vorgestellt worden ist, hat in vielen Bereichen, seien es Internetforen oder E-Mails, Einzug gefunden. Sei, dass jedes Wort in dem zu klassifizierenden Text unhabhängig von den anderen Wörtern auftritt. Zuerst wird die Spamwahrscheinlichkeit für jedes Wort in dem zu klassifizierendem Text berechnet. Hierzu wird die Häufigkeit aller Vorkommen jedes Wortes in den Spamdaten, wie auch in den Hamdaten benötigt. Sei Hs_{wort} die Häufigkeit des Wortes 'wort' in allen Spamdaten und sei Hh_{wort} die Häufigkeit des Wortes 'wort' in allen Hamdaten. Desweiteren sei Hs die Anzahl aller Wörter in den Spamdaten und Hh die Anzahl aller Wörter in den Hamdaten. Nun wird die Spamwahrscheinlichkeit p_s eines Wortes wie folgt berechnet:

$$p_s = \frac{\frac{Hs_{wort}}{Hs}}{\frac{Hs_{wort}}{Hs} + \frac{Hh_{wort}}{Hh}}$$

Zum Klassifizieren werden nur die Worte benutzt, die mehr als fünfmal auftreten und von allen diesen Worten nur die 15 aussagekräftigsten. Mit aussagekräftig ist gemeint, dass ein Wort umso aussagekräftiger ist, desto weiter das errechnete p_s von 0,5 entfernt ist. Die Wahrscheinlichkeit P_{text}, ob ein zu klassifizierender Text eher Spam oder eher Ham ist, lässt sich nun wie folgt berechnen [5]:

$$P_{text} = \frac{\Pi_{i=1}^{15} p_{s_i}}{\Pi_{i=1}^{15} p_{s_i} + \Pi_{i=1}^{15}(1 - p_{s_i})}$$

2.3 K-nearest neighbor Algorithmus

Der Nearest neighbor Algorithmus beruht auf einer sehr einfachen Idee. Man hat wie üblich eine Menge von Trainingsdaten, als Instanzen des in Abschnitt 2.1 beschriebenen Merkmalsvektors. Soll ein neuer Text aufgrund dieser Trainingsdaten

kategorisiert werden, wird der Text jener Klasse zugeordnet, die die Instanz des Merkmalsvektors der Trainingsdaten mit der des zu klassifizierenden Textes im Merkmalsvektorraum mit geringstem Abstand hat. Ist in den Trainingsdaten ein identischer Vektor vorhanden, wird der zu klassifizierende Text genau diese Klasse des Vektors erhalten.
Eine Verallgemeinerung dieses Verfahrens ist der k-Nearest neighbor Algorithmus. Hier wird die Entscheidung für eine Klasse nicht aufgrund eines Nachbarn entschieden, sondern aufgrund von k-Nachbarn mit $k > 0$. Befinden sich unter den k-Nachbarn unterschiedlich klassifizierte Vektoren, könnte man beispielsweise das arithmetische Mittel dazu verwenden, um den Text einer bestimmten Klasse zuzuordnen.
Der Abstand zwischen zwei Vektoren im hochdimensionalen Raum kann auf ganz unterschiedliche Art und Weise berechnet werden. Es empfiehlt sich, Verfahren zu benutzen, welche die Distanz zweier Vektoren sehr schnell berechnen, da dieses Verfahren bei immer größer werdenden Trainingsdaten sonst sehr langsam werden. [3]

2.4 Entscheidungsbaum

Ein Entscheidungsbaum repräsentiert Daten, hier wieder die Menge der Instanzen des beschriebenen Merkmalsvektors der Trainingsdaten, in der Art und Weise, dass man an jedem Knoten, den der Baum hat, eine neue 'Frage' beantworten muss, und so einen Pfad von der Wurzel beginnend bis zu einem Blatt durchläuft, wobei die Blätter des Entscheidungsbaums einer Klasse entsprechen. Ein Beispiel eines solchen Entscheidungsbaums ist in Abbildung 2.1 zu sehen.
Nun sollen die Merkmalsvektoren in einer etwas anderen Art und Weise betrachtet werden. Wie in Abschnitt 2.1 erklärt, beschreibt jede Dimension des Vektors ein Merkmal. Im Falle der Spam- und Hamklassifizierung von Texten haben alle Merkmale die Form 'Das Wort X kommt n mal in diesem Text vor'. Nun können 'Fragen' an die einzelnen Dimensionen gestellt werden. Wie zum Beispiel: 'Kommt das Wort 'dumm' mehr als nullmal in dem zu klassifizierendem Text vor? Wenn ja und alle Datensätze der Trainingsdaten, in denen das Wort 'dumm' ebenfalls mehr als nullmal vorkommt, würden beispielsweise alle als Spam klassifiziert, so wird der zu klassifizierende Datensatz ebenfalls als Spam klassifiziert. Kommt in dem Text das Wort 'dumm' aber nicht vor, werden alle Trainingsdaten, in denen das Wort 'dumm' nicht vor kommt, betrachtet. Diese Datensätze wären jetzt zum einen Teil der Klasse Spam, zum anderen Teil der Klasse Ham zuzuordnen. Über die Klasse des Datensatzes kann man also noch keine genaue Aussage machen. Das heisst, es müssen weitere Fragen über andere Merkmale gestellt werden, bis der Datensatz eindeutig kategorisiert ist.
In einem Entscheidungsbaum 'beschreibt' jeder innere Knoten eine solche Frage. Je nach Antwort wird für den einen oder anderen Pfad entschieden. Jedoch bleibt die Frage, wie ein solcher Entscheidungsbaum konstruiert wird. Die einfachste Methode ist es, jeden Datensatz als einen Pfad, von der Wurzel beginnend, zu

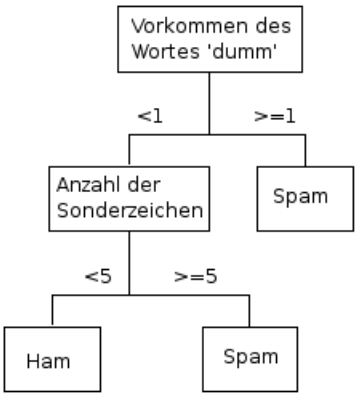

Abbildung 2.1: Entscheidungsbaum

beschreiben. Um die Konstruktion und Problematik eines Entscheidungsbaums zu verdeutlichen, wird ein Entscheidungsbaum aufgrund folgender Daten erstellt:

DS	ZA	Sonderzeichenanzahl	'dumm'	Spam/Ham
1	15	2	0	Ham
2	178	4	0	Ham
3	72	5	0	Ham
4	34	20	0	Spam
5	55	4	1	Spam
6	190	100	1	Spam

DS steht für Datensatz, ZA für Zeichenanzahl, 'dumm' für Vorkommenshäufigkeit dieses Wortes. In der Spalte Spam/Ham ist zu sehen, ob dieser Datensatz Ham oder Spam ist.
Der aus diesem Datensatz konstruierte Entscheidungsbaum ist in Abbildung 2.1 zu sehen. Dieser Baum wurde mittels des J48 [4] Alghoritmus erstellt. Hier sei noch gesagt, dass es eine Vielzahl von richtigen Entscheidungsbäumen für eine Datensatzmenge gibt.
Bei der automatischen Erstellung eines Entscheidungsbaums kommt es vor allem auf die Wahl des am besten geeigneten Attributes an. Eine Möglichkeit ist, die Attribute aufgrund ihres Informationsgehaltes auszuwählen. Eine Formel zur Berechnung des Informationsgehaltes von Daten ist aus der Informationstheorie bekannt. [10] [4]

2.5 Supported Vector Machine - SVM

Der SVM-Algorithmus versucht zwischen den aus den Daten erzeugten Instanzen des Merkmalsvektors eine Trennlinie oder, je nach Dimension, eine Fläche oder sogar Hyperebene zwischen den Instanzen der einen und den erzeugten Instanzen der

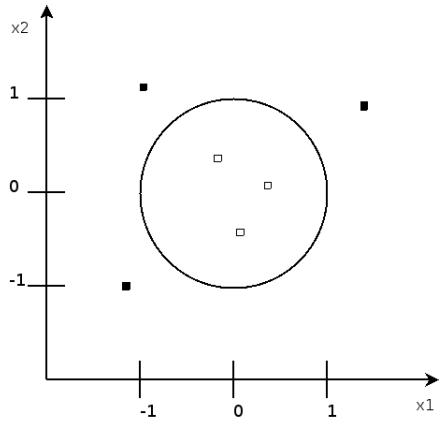

Abbildung 2.2: Merkmalsvektorraum

anderern Klasse zu ziehen. Die Klassen sind wieder Ham und Spam. Dies ist in dem Merkmalsvektorraum nicht immer direkt möglich. Um nun eine geeignete Trennlinie zu finden, wird die Dimension des Merkmalvektorraums erhöht, indem aus den vorhandenen Merkmalen neue Merkmale errechnet werden. Hierzu ein Beispiel:
Angenommen, man hätte einen zwei dimensionalen Merkmalsvektorraum, wie er in Abbildung 2.2 zu sehen ist, man hat demnach genau zwei Merkmale (x_1, x_2). Die erzeugten Instanzen des Merkmalsvektors aus den positiven, Hamdaten, seien diejenigen in dem Kreis und die erzeugten Instanzen des Merkmalsvektors aus negativen Daten seien diejenigen außerhalb des Kreises. Die Daten sind in diesem Vektorraum nicht linear teilbar. Nun werden aus den vorhandenen Merkmalen x_1 und x_2 die neuen Merkmale wie folgt berechnet:

$$\sqrt{2 * x_1 * x_2}$$

$$x_1^2$$

$$x_2^2$$

So erhält man einen neuen, diesmal drei dimensionalen Vektorraum, in dem alle Daten linear teilbar sind. Diese Fläche ist in Abbildung 2.3 zu sehen. Die positiven Instanzen befinden sich 'links' von der Fläche und die negativen 'rechts' von der Fläche.
Es ist nicht immer möglich in dem errechneten Mermalsvektorraum, der nur um eine Dimension größer ist als der ursprüngliche Merkmalsvektorraum, eine Trennlinie zu ziehen. In bestimmten Fällen kann es passieren, dass bei einer Anzahl von konstruierten Merkmalsvektoren aus den Ham- und Spamdaten a dies erst bei einer Dimension von $a - 1$ oder sogar größer möglich ist. Alternativ kann auch eine Trennlinie konstruiert werden, bei der sich nur einige wenige Daten auf der falschen Seite der Trennlinie befinden.

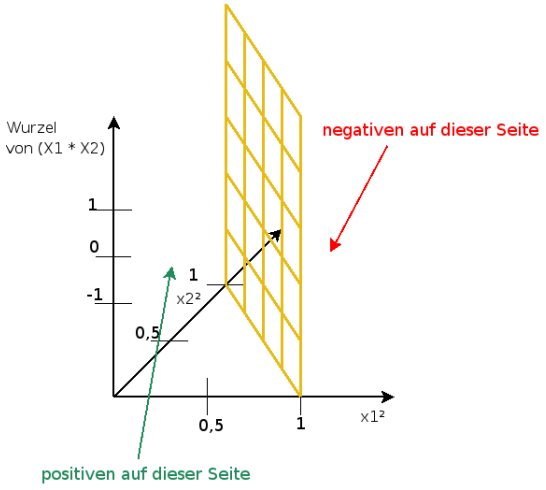

Abbildung 2.3: Neu errechneter Vektorraum

Soll nun ein neuer Datensatz klassifiziert werden, muss nur ermittelt werden, auf welcher Seite der errechneten Tennlinie sich dieser befindet. [3] [10]

2.6 Sonstiges

Bei einem statistischen Test zur Beurteilung eines Klassifizierers werden die positiven Daten, die mittels dieses Klassifizieres auch richtig klassifiziert worden sind, als 'true positives' bezeichnet. Dem gegenüber stehen die positiven Daten, welche von einem Klassifizierer falsch klassifiziert worden sind. Diese nennt man 'false positives'. Die negativen Daten, welche ein Klassifizierer richtig klassifiziert hat, nennt man 'true negatives' und die falsch klassifizierten Daten entsprechend 'false negatives'.
Die Maße Precision und Recall berechnen sich wie folgt:

$$Precision = \frac{true\ positives}{true\ positives + false\ positives}$$

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

Es seien die 'true positives' die Spamnachrichten, die richtig klassifiziert und die 'false positives' die Hamnachrichten, die richtig klassifiziert worden sind. Dann gibt Precision an, wieviel Prozent aller als Spam klassifizierten Nachrichten tatsächlich Spam sind und Recall wieviel von den gesamten Spamnachrichten als Spam klassifiziert worden sind.
Wenn nur ein großer Datensatz zur Verfügung steht, d.h. die Trainingsdaten sind noch nicht von den Testdaten getrennt, gibt es mehrere Möglichkeiten, wie solche

Daten validiert werden können. Trennt man diese Daten zu Trainings- und Testdaten aufgrund einer bestimmten Prozentzahl, dass heißt, wenn beispielsweise 100 Daten vorliegen und man sich für 25 Prozent entscheidet, dann werden die ersten 25 Daten zum Trainieren des Klassifizierers genutzt und die restlichen 75 Prozent zum Testen. Das wird in dieser Ausarbeitung mit 'percentage split' bezeichnet.
Eine andere Möglichkeit ist, die 100 Daten aus dem Beispiel in 10 gleich große Datenmengen zu teilen. Demnach hier je zehn Datensätze. Nun kann die erste Datenmenge zum Testen separat von den anderen erzeugten Datenmengen gehalten werden und mit den anderen 9 Datenmengen der Klassifizierer trainiert werden (Testmenge, Trainingsmenge). Dann wird mit der zurückgehaltenen Datenmenge der Klassifizierer getestet. Nun macht man das ganze zehnmal und benutzt jedesmal eine andere Datenmenge zum Testen. Die so erhaltenen Testergebnisse werden gemittelt. Dieses Ergebnis wird dann als Gesamtergebnis genutzt. Diese Methode zur Validierung eines Klassifizierers wird im Folgenden mit dem englischen Begriff 'cross validation' beschrieben.
Wird in dieser Ausarbeitung von Sonderzeichen geredet, sind damit ausschließlich die Zeichen '!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<', '=', '>', '?', '@', '[', '\', ']', '^', '_', '{', '|', '}' und '~' gemeint.
Ein 'whitespace character' (zu deutsch Leerraum Zeichen) ist eines der Zeichen Tabulator, Zeilenvorschub, Wagenrücklauf oder Zeilenumbruch, Seitenumbruch und Vertikaler Tabulator. Im Weiteren wird der englische Ausdruck 'whitespace character' für diese Menge von Zeichen benutzt, jedoch zusammengeschrieben, also Whitespacecharacter. Dies soll verdeutlichen, dass es sich hier um eine Menge von Zeichen handelt.

3 Analyse und Klassifizierung kurzer Nachrichten

In diesem Kapitel wird die im Bezug auf E-Mails etwas andere Situation im Detail erklärt, um dann auf dieser Grundlage eine geeignete Menge von Nachrichten für diese Situation zu finden. Es werden Überlegungen angestellt, wie für diese Situation zutreffende Nachrichten aussehen können, um dann geeignete Klassifizierungswerkzeuge zu finden. Einige ausgewählte Klassifizierungswerkzeuge werden danach evaluiert werden.

3.1 Situationsbeschreibung

Für viele Bereiche wurden Texte analysiert, um diese in die verschiedensten Kategorien zu klassifizieren. Gerade die Klassifizierung nach Spam und Ham ist heute durch E-Mails sehr bekannt. Jedoch sind bei der Klassifizierung von E-Mails etwas andere Bedingungen gegeben. Eine kurze Beschreibung der Situation wurde in Kapitel 1 gegeben, diese wird nun noch ausführlicher beschrieben.
Es wird davon ausgegangen, dass die Nachrichten eine Zeichenlänge von 250 Zeichen nicht überschreiten. Desweiteren laufen die Nachrichten als Lauftext über Bildschirme. Dadurch ist auch eine Einschränkung der Nachrichten auf 250 Zeichen durchaus gerechtfertig, da einige Tests mit Lauftextnachrichten ergeben haben, dass mit steigender Länge und somit meist mit steigender Komplexität dieser Nachrichten, der Inhalt in Lauftextform für den Betrachter nur noch schwer zu erfassen ist.
Lauftextnachrichten sollen zukünftig von jeder Person auf die Bildschirme gesendet werden können, welche eine Rechnerkennung der Universität Koblenz besitzt. Zum jetzigen Zeitpunkt ist der Personenkreis, der Nachrichten auf den Bilschirm senden kann, auf nur sehr wenige Personen beschränkt (weniger als 10). Sobald der Personenkreis auf alle Personen mit einer Rechnerkennung erweitert worden ist, sind mehrere tausend Personen grundsätzlich in der Lage, Nachrichten auf die Bildschirme zu senden. Im folgendem Text werden Nachrichten, die als Lauftext über die Bildschirme laufen, nur noch Tickernachrichten genannt. Bei einem solch großen Personenkreis erhöht sich die Missbrauchgefahr um ein Vielfaches. Diese Gefahr wird durch eine Pseudoanonymität der Benutzer verstärkt. Mit Pseudoanonymität ist gemeint, dass dem Benutzer unter Umständen nicht bewusst ist, dass er identifizierbar ist, da man nicht davon ausgehen kann, dass irgendwelche Daten, die den Benutzer identifizieren, an die Tickernachricht angehängt werden.

Die Nachrichten sollen zukünftig über verschiedene Interfaces auf die Bildschirme gesendet werden können. Ein Interface kann eine Homepage sein, auf der einfach der Text, der über den Ticker laufen soll, eingestellt wird. Auch vom Handy aus sollen in Zukunft via Bluetooth Nachrichten an gewisse Accesspoints gesendet werden können, welche diese wiederum weiter an die Bildschirme senden. Jedoch wird bei beiden Verfahren sichergestellt sein, dass der Absender über die Rechnerkennung ermittelbar ist. Auf die Verfahren, wie die Nachrichten zu den Bildschirmen kommen, wird in dieser Arbeit nicht näher eingegangen. Ebenso die Frage, wie ein Benutzer, der eine Nachricht auf die Bildschirme senden will, eindeutig identifiziert werden kann, wird hier für den Leser nicht beantwortet. [7]

3.2 Besonderheiten des Szenarios

Durch das oben beschriebene Szeanrio ergeben sich einige Besonderheiten bei der Klassifizierung und der damit verbundenen Analyse der Nachrichten. Diese werden nun näher erklärt. Obwohl es schon viele Verfahren und Studien der Spam- und Hamklassifizierung gibt, besonders im Bereich der E-Mails, ist die oben beschriebene Situation nur teilweise mit diesen vergleichbar.
Im Bereich der E-Mails gibt es sehr große vorklassifizierte Datensätze. Auf diese kann bei der Entwicklung eines automatisierten Klassifizierers zurückgegriffen werden. Solche Daten gibt es für die vorgestellte Situation nicht. Nun liegt es nahe, Datensätze aus E-Mails zu extrahieren, die auf die beschriebene Situation passen. Dies ist aus verschiedenen Gründen nur wenig sinnvoll.
Zum einen besitzen die Nachrichten in den E-Mails meist mehr als 250 Zeichen. Darüber hinaus sind E-Mails, die weniger als 250 Zeichen lang sind, nicht repräsentativ für eine Nachricht, wie sie vermutlich über den Bildschirm laufen wird. Durch diese Einschränkung der Zeichenlänge einer Nachricht ist zu erwarten, dass auch die Tickernachrichten eine andere Formulierung besitzen werden. So kann man davon ausgehen, dass beispielsweise Abkürzungen, wie sie in Kleinanzeigen von Zeitungen zu finden sind, häufiger genutzt werden.
Nicht nur die unterschiedliche Länge ist ein signifikanter Unterschied zwischen E-Mails und Tickernachrichten, sondern auch die Form, in der eine solche Nachricht verfasst wird. E-Mails sind meist personalisiert und richten sich an nur wenige, ausgewählte Benutzer. Somit werden hier Informationen ausgetauscht, die sich unter Umständen auf vorangegangene Gespräche beziehen. Auch ein solches Szenario ist bei den Tickernachrichten nicht zu erwarten. Es wird nicht davon ausgegangen, dass über den Ticker komplexere Dialoge geführt werden, da es für solche Fälle andere effektivere Alternativen geben wird.
Da die Bildschirme, über welche die Tickernachrichten laufen werden, in der Universität angebracht sind, sind die 'Leser' dieser Tickernachrichten zum größten Teil Studenten. Eine kleinere Gruppe sind Angestellte und sonstige Mitarbeiter der Universität. Auch die Nachrichten können, da diese nur von Personen mit einer Rechnerkennung der Universität Koblenz eingestellt werden können, nur von Studenten

oder Mitarbeiter der Universität eingestellt werden. Andere Personen besitzen keine Rechnerkennung und somit auch keine Möglichkeit Nachrichten auf den Bildschirmen zu präsentieren.
Nun werden nur Spamnachrichten näher betrachtet. Es besteht vermutlich auch ein Motivationsunterschied zwischen dem Verfasser und Sender von Spamnachrichten als E-Mail und dem Verfasser und Sender von Spamtickernachrichten. Bei E-Mails ist Spam meist einfach unerwünschte Werbung für noch des öfteren unseriöse Produkte. Auch Angebote von unseriösen Geschäften können Spam bei E-Mails sein. Es ist eher selten, dass ein Benutzer auf solche unseriösen Geschäfte eingeht. Werden E-Mails klassifiziert, so befinden sich die als Spam klassifizierten Nachrichten meist in einem besonderen Ordner, der hin und wieder nach falsch klassifizierten Nachrichten durchsucht werden soll. So kommt der Empfänger einer E-Mail trotzdem in Kontakt mit vielen Spamnachrichten. Somit dient eine Klassifizierung grundsätzlich nicht zum Schutz des Empfängers vor Werbung von Produkten, die er nicht benötigt, sondern eine automatisierte Klassifizierung soll ihm in seinem normalen Arbeitsablauf Zeit ersparen. Bei den Tickernachrichten ist aber Werbung grundsätzlich erlaubt, nur einige Produkte sind ausgeschlossen. An dieser Stelle stellt sich also die Frage, wie Spam als Tickernachricht zu charakterisieren ist.
Man kann Spam als Tickernachricht nicht vorab und in jedem Detail ganz klar charakterisieren. Spam als E-Mail könnte man so charakterisieren:

> "Der Begriff Spam bezeichnet unverlangte zugesandte Massenemails. Unverlangt ist eine E-Mail dann, wenn das Einverständnis des Empfängers zum Empfang der Nachricht nicht vorliegt und nicht zu erwarten ist. Massenemail bedeutet, dass der Empfänger die Nachricht nur als einer von vielen erhält. " Seite 13, [12]

Wie gesehen wurde, ist der Empfänger in der vorgestellten Situation eine große Gruppe von Personen, die nicht im vorhinein festgelegt ist. Eben genau die Personen, welche die Nachricht lesen, wenn diese über den Bildschirm läuft. Somit kann ein Angebot für die eine Person unerwünscht sein und für die andere nicht. Derjenige, der entscheidet, welche Nachrichten über den Bildschirm laufen, ist die Person, die den Klassifizierer administriert. Jedoch ist auch hier nicht vorhersagbar, welche Nachrichten von dieser Person als Spam und welche als Ham charakterisiert werden. Es kann aber davon ausgegangen werden, dass diese Person keine Beleidigungen, unseriöse Werbung oder Werbung für Dinge auf den Ticker möchte, die i.A. als anstößig gelten. Auch ist davon auszugehen, dass Nachrichten, die 'ohne Inhalt' über die Bildschirme laufen, als Spam zu klassifizieren sind.
Aus diesen Vorüberlegungen ist nun das größte Problem, gute vorklassifizierte Spam und Ham Daten zu erhalten. Der erste Schritt war somit, einen geeigneten Datensatz von Spam- und Hamnachrichten zu erstellen. Um diesen geeignet zu wählen, wurde versucht, aus den obigen Vorüberlegungen verschiedene Klassen von Spam zu konstruieren. Hierzu mehr im folgendem Kapitel.
Eine weitere Problematik besteht darin, dass bis auf den Absender keine weiteren Metadaten analysiert werden können, wie es beispielsweise bei E-Mails möglich ist.

Hier liegt neben dem Text der Header der E-Mail vor, aus dem weitere wichtige Informationen zur Klassifizierung gewonnen werden können.

3.3 Verwendung einiger Verfahren zur Klassifizierung

Um geeignete Verfahren zu wählen, wird sich erst damit befasst, welche Arten von Spam auftreten können. Hier werden grundsätzlich drei verschiedene Klassen von Spamnachrichten unterschieden.

1. Nachrichten, die ein Wort beinhalten, welches auf keinen Fall über den Ticker laufen soll.

2. Nachrichten, bei denen alle Worte für sich betrachtet keine besondere Spamauffälligkeit geben, aber aus dem Zusammenhang eine solche Nachricht nicht erwünscht ist.

3. Nachrichten, die keinen Inhalt haben und nur aus belanglosen Worten bestehen oder aus scheinbar 'zufällig' aneinander gereihten Zeichen.

Die einfachste zu erkennende Klasse von Spamnachrichten ist die 1. Klasse. Hier liegt die Idee nahe, einfach alle Wörter der Nachricht zu untersuchen. Sobald eines dieser Wörter, identisch oder nahezu identisch mit einem Wort aus einer vorher angelegten Liste ist, einem sogenannten Badword, ist diese Tickernachricht als Spam zu klassifizieren.
Jedoch ist der Mensch in der Lage Wörter oder Buchstaben so zu ersetzen, dass diese ein anderes Wort ergeben, welches aber noch mit dem Badword assoziiert wird. Auch gibt es Möglichkeiten, einzelne Buchstaben eines Satzes zu markieren, so dass die markierten Buchstaben ein Badword bilden. Solche Möglichkeiten des Versteckens eines Wortes werden nun genauer betrachtet.
Eine Möglichkeit ist, einzelne Buchstaben des Wortes durch andere, ähnlich aussehende Zeichen zu ersetzen. So könnte beispielsweise aus dem Wort 'doof', das Wort 'd00f' werden. Die zwei 'o' in der Mitte wurden durch zwei nullen ausgetauscht.
Eine weitere Möglichkeit ist, wahllos belanglose Zeichen zwischen das Badword zu streuen. So könnte beispielsweise aus dem Wort 'idiot' das Wort 'i()dio!t' werden, welches für den Menschen ohne Probleme noch als das Ursprungswort 'idiot' lesbar ist. Auch kann, wie oben schon erwähnt versucht werden, durch Markierung bestimmter Zeichen innerhalb eines Textes ein Wort zu bilden. Sei es, dass alles klein geschrieben ist, bis auf bestimmte Buchstaben, oder sei es, dass bestimmte Sonderzeichen benutzt werden, um diese Buchstaben zu kennzeichnen. So kann aus dem Text 'heute abend gehe ich aber sicherlich einkaufen' durch voransetzen von einem Ausrufezeichen vor bestimmte Buchstaben das Wort 'arsch' markiert werden. Der Text sähe dann so aus: 'heute abend gehe ich !abe!r !si!c!helich einkaufen'. Weglassen von Buchstaben und das Verändern der Position von Buchstaben in Wörtern, also Anagrammbildung, ist ebenfalls eine Methode, um das Badword im Text zu

verstecken. So könnte aus dem Wort 'doof' das Wort 'dof' werden und aus dem Wort 'Idiot' 'Idito'.
Im Anhang C finden Sie eine Liste von selbst erstellten Spam- und Hamtexten. Die Spamtexte wurden größtenteils aufgrund der gemachten Vorüberlegungen konstruiert. Die Hamtexte in Anhang C sind zum Teil frei erfunden, einige aber auch inspiriert durch eine Newsgroup an der Universität Koblenz.
Spam, der Badwords enthält, kann mittels regulärer Ausdrücke klassifiziert werden. Besonders gehören dazu Badwords, die mittels Buchstabenersetzung, Weglassen von einem Buchstaben der doppelt vorkommt, Streuen von belanglosen Zeichen in das Wort oder das Vervielfachen von Buchstaben konstruiert wurden. Hierzu können passende reguläre Ausdrücke gefunden werden. Sehr schwer zu erkennen sind Badwords, bei denen entweder ein einfaches, für den Menschen noch lesbares Anagramm gebildet wurde oder Badwords, bei denen Buchstaben weggelassen wurden. Solche Worte als Badwords zu erkennen ist fast nicht möglich. Aus einem normalen Wort automatisiert einen regulären Ausdruck zu bilden, der Anagramme des Badwords oder Badwords konstruiert durch weglassen von Buchstaben erkennt, ist fast nicht möglich, da es vorkommen kann, dass so ein neues, in der Sprache gebräuchliches Wort entsteht, welches dann jedesmal als Badword erkannt wird, aber keines ist. Hierzu ein kleines Beispiel:
Angenommen, der Benutzer des Programms hätte das Badword 'Huren' den Daten hinzugefügt. Hierzu würde automatisch ein regulärer Ausdruck gebildet werden, der auch auf alle Wörter des Badwords passt, bei denen nur 2 nebeneinander liegende Buchstaben vertauscht sein können. Nun gibt jemand den Hamtext 'Verkaufe Uhren im Wert von ...' ein. Dieser Text würde sofort als Spam erkannt, da er das Wort 'Uhren' enthält und dies ein Wort ist, welches durch das Vertauschen von 2 Buchstaben aus dem Wort 'Huren' entstanden ist. Ähnlich verhält es sich mit dem Weglassen von Buchstaben in einem Badword.
Jedoch ist hier schon zu erkennen, dass reguläre Ausdrücke als ein geeignetes Mittel zur Erkennung von Spam Nachrichten eingesetzt werden können.
Auch ist im Anhang C Spam zu finden, der oben als Typ 2 Spam beschrieben wurde. Diese Art von Spam ist, wie schon gesagt nicht an einem einzigen Wort fest zu machen. Demnach sind hier reguläre Ausdrücke nicht besonders geeignet.
Es gibt eine schier unzählige Art, wie Spamsätze aussehen können, die kein einziges Badword haben. Hier liegt die Vermutung aber nahe, dass auch bestimmte Wörter in solchen Texten gehäuft vorkommen. Für solche Fälle sind Lernmethoden aus der künstlichen Intelligenz durchaus sinnvoll. Solche Klassifizierungsmethoden werden schon in vielen bekannten Spamfilter aus dem Bereich der E-Mails verwandt. Eine Analyse der Klassifizierungsmethoden folgt im nächsten Kapitel. Aufgrund von den aus dem Bereich der E-Mails bekannten Klassifizierern wurde entschieden, die folgenden zu testen:

- Naive Baise (NB)
- K-nearest neighbor Algorithmus (KNN)

- Supported Vector Machine (SVM)
- Entscheidungsbaum bzw. Decision Tree (J48) (DTJ)

Diese Klassifizierer haben im Falle von Spammails gezeigt, dass sie sehr gute Erkennungsraten liefern.
Um eine bessere Erkennungsrate für den Typ 3 Spam zu erreichen, könnte man die Nachricht auf folgende Informationen untersuchen:

- Textmindestlänge - Anzahl aller Zeichen innerhalb des zu klassifizierenden Textes
- Anzahl verschiedener Buchstaben im Text - Wieviele verschiedene Buchstaben sind in dem zu klassifizierenden Text vorhanden
- Verhältnis Whitespacecharacter zur Textlänge - Besitzt der Text eine übermäßig große Anzahl von Whitespacecharacter
- Verhältnis Sonderzeichen zur Textlänge - Besitzt der Text eine übermäßig große Anzahl an Sonderzeichen
- Sonderzeichenanzahl zwischen Buchstaben - Wurden im Text zwischen Buchstaben Sonderzeichen eingefügt
- Entropie von n-Tupeln

Der Ansatz der Entropie wird im nächsten Kapitel genauer untersucht werden. Die anderen Ansätze bedürfen keiner weiteren Analyse. Im Weiteren wird beschrieben, welche Art von Spam mit Hilfe der oben im Text genannten Punkte gewonnenen Information wahrscheinlich erkannt werden kann. Hieraus ergibt sich gleichzeitig das Verständnis, weshalb die meisten genannten Punkte keiner weiteren Analyse bedürfen.
Bei der Mindesttextlänge kann davon ausgegangen werden, dass erst ein Text ab einer gewissen Buchstabenanzahl eine sinnvolle Nachricht ergibt. Texte mit nur wenigen Buchstaben enthalten meistens keine sinnvolle Information.
Auch bei Texten mit nur wenigen verschiedenen Buchstaben kann davon ausgegangen werden, dass es sich hier um eine Nachricht ohne wirklichen Inhalt handelt. Eine Nachricht die beispielsweise nur aus einem Buchstaben besteht, ist mit großer Wahrscheinlichkeit keine sinnvolle Nachricht.
Bei zu häufiger Verwendung von Whitespacecharactern ist davon auszugehen, dass es sich um keine wirkliche Nachricht mit sinnvollem Text handelt. Dies ist in Abhängigkeit zu der Anzahl der Gesamtzeichen zu sehen.
Nachrichten mit einer zu großen Anzahl von Sonderzeichen im Verhältnis zur Gesamtzeichenanzahl sind mit großer Wahrscheinlichkeit keine Hamtexte. Hier kann davon ausgegangen werden, dass die Sonderzeichen entweder irgendwelche Buchstaben ersetzen sollen, in einem solchen Fall ist der Text eher unseriös oder dass

irgendwelche Markierungen im Text vorgenommen worden sind, wie sie weiter oben kurz vorgestellt wurden. Auch besteht die Möglichkeit, dass Sonderzeichen, wahllos aneinander gereiht, den Ticker einfach nur stören sollen. Jedoch gibt es auch Fälle, in denen sich die Sonderzeichen Anzahl erhöht im Verhältnis zum Gesamttext. So war in der Vergangenheit des Öfteren zu beobachten, dass Personen, welche ihre Nachricht für besonders wichtig erachteten, den Anfang und das Ende des Textes mit einigen Ausrufezeichen versahen, weches ebenfalls ein Sonderzeichen ist.
Anders als in dem zuletzt beschriebenen Fall verhält es sich bei Sonderzeichen, welche direkt von weiteren Buchstaben umschlossen sind. Hier kann davon ausgegangen werden, dass es sich entweder um Markierung bestimmter Worte handelt, oder wahllos auf den Tasten der Computertastatur herumgetippt wurde.
Die Entropie soll Aufschluss darüber geben, ob es sich um einen gültigen Text handelt. Dieser Wert ist aber nur sinnvoll, wenn es für eine bestimmte Art von Spamtext Werte gibt, die sich signifikant von allen anderen Werten des Spamtextes unterscheiden.

3.4 Analyse der Klassifizierungsmethoden

Um die verschiedenen Klassifizierer, NB, KNN, SVM und DTJ zu analysieren wurden 18 verschiedene Datensätze im arff-Format [1] erstellt. Diese wurden aus den Spam- und Hamtexten im Anhang C gefertigt. Die Datensätze unterscheiden sich hinsichtlich der Vorverarbeitung der Texte. Wie diese verschiedenen Datensätze erzeugt worden sind, kann im Anhang D eingesehen werden. Danach wurden mittels *Weka* [2] einige Klassifizierungsmethoden analysiert.
Die einzelnen Merkmalsvektoren im arff-Format unterscheiden sich aufgrund verschiedenster Vorverarbeitung der Wörter und jeder Datensatz besitzt somit unterschiedliche Merkmale. Aufgrund welcher Eigenschaften sich diese Datensätze im Einzelnen unterscheiden, wird nun näher beschrieben.
Jedes Merkmal beschreibt grundsätzlich das Vorhandensein eines Wortes in einem Text. Ein Unterscheidungspunkt ist, ob es sich bei dem Merkmalsvektor um einen binären Merkmalsvektor handelt oder nicht. Eine genaue Übersicht der einzelnen verwendeten Datensätze mit Auswertung befindet sich in Anhang D. In der Spalte Vektormodell ist verzeichnet, ob es sich um ein 'normales' Vektormodell handelt oder um ein binäres. 'Normal' bedeutet hier, dass in den Dimensionen des Merkmalvektors die Häufigkeit des Vorkommens eines Wortes als natürliche Zahl eingetragen ist, wobei binär mit einer 0 oder 1 darstellt, ob dieses Wort vorkommt. So sind die Daten der Datensätze 1 bis 6 und die Datensätze 11 bis 18 als ein 'normaler' Merkmalsvektor, die Datensätze 7 bis 10 als ein binärer Merkmalsvektor repräsentiert.
Welche Worte für ein Merkmal genutzt werden, unterscheidet sich hinsichtlich der Länge der Zeichen des Wortes. In den Datensätzen wird zwischen Wörtern ab den Zeichenlängen von 1, 3 und 7 unterschieden. Die Datensätze 1, 4, 7 und 9 beachten nur Wörter ab einer Zeichenlänge von 1, die Datensätze 2, 5, 8, 10, 11, 12, 13, 14, 16, 17 und 18 beachten nur Wörter ab einer Zeichenlänge von 3 und die Datensätze

3, 6 und 15 beachten nur Wörter ab einer Zeichenlänge von 7.
Auch wird unterschieden, welche Wörter zu einem Merkmal, in der oben beschriebenen Art und Weise, zusammengefasst werden. So kann das Wort 'KATZE' und das Wort 'katze' zusammengefasst in einem Merkmal repräsentiert werden oder nicht. Es wird also differenziert, ob bei der Merkmalsvektorerstellung auf Groß- und Kleinschreibung geachtet wurde. Die Datensätze 1, 2, 3, 7, 8, 11, 14, 16 und 18 verzichten auf eine Unterscheidung der Groß- und Kleinschreibung. Die Datensätze 4, 5, 6, 9, 10, 12, 13, 15, und 17 repräsentieren einen Merkmalsvektor, bei dem die Groß- und Kleinschreibung beachtet wurde.
Des Weiteren wird unterschieden, welche Zeichen in einem Wort ersetzt werden. So könnte man das Wort 'h_a_l_l_o' ebenso wie das Wort 'hallo' in einem Merkmal repräsentieren, wenn die Unterstriche in der Vorverarbeitung der Wörter zur Merkmalsgewinnung durch das leere Wort ersetzt würden. Welche Zeichen nicht durch das leere Wort ersetzt worden sind, ist in der Spalte 'Behandlung der Sonderzeichen' zu finden.
Es gibt auch einige Zeichen, welche als 'ein Zeichen' zusammengefasst werden können. Zum Beispiel könnte man ein '@' als den Buchstaben 'a' interpretieren. So können in der Vorverarbeitung der Wörter solche Zeichen oder auch Buchstaben in einen anderen überführt werden. Welche Zeichen bei welchem Datensatz in welches andere Zeichen überführt wurden, um dann aus diesem neu gewonnenem Wort ein Merkmal in oben beschriebener Weise des Merkmalvektors zu erstellen, ist in der Spalte 'Zeichenersetzungen' zu sehen.
Jeder Datensatz wurde dreimal erstellt, die Instanzen des Merkmalvektors der Spam- und Hamdaten aber in ihrer Reihenfolge in der arff Repräsentation unterschiedlich durchmischt. Da die Daten mittels 'cross validation' oder mittels 'percentage split' ausgewertet wurden, kann hier ein Unterschied in den ausgewerteten Werten auftreten. Welche Datensätze mittels cross validation oder percentage split ausgewertet wurden, ist in der Spalte Testmethode aufgeführt.

Naive Baise

An dieser Stelle wird die Auswertung der Daten in Kurzform repräsentiert. Eine vollständige Auswertung kann im Anhang D aus den Tabellen entnommen werden. Hier die Werte im Durchschnitt des NB-Verfahrens. Zur Auswertung wurde die Klasse 'weka.classifiers.bayes.NaiveBayes' des Weka-Klassifizierers benutzt.

Recall	Precision	richtig erkannt	falsch erkannt
0,8616	0,8030	0,7474	0,2526

Die oben gezeigten Werte von Recall, Precision, richtig erkannt und falsch erkannt sind Durchschnittsdaten aus allen durchgeführten Tests mit den drei mal 18 Datensätzen. Getestet wurde mittels den Ham- und Spamdaten aus Anhang C. Wie oben beschrieben wird nicht im Vorfeld explizit zwischen Testdatensatz und Trainingsdatensatz unterschieden. Die Daten wurden aufgrund von 'cross validation' oder 'percentage split' erhoben. Wie man dem Anhang entnehmen kann, handelt es sich

um 50 Ham- und 125 Spamnachrichten, also ca. 29 % Hamnachrichten und 71 % Spamnachrichten.
Alle Daten, Recall, Precision, usw., beziehen sich, in diesem Kapitel wie auch in den entsprechenden Anhängen, auf die Klasse Spam. D.h., wenn die Ergebnisse des NB-Verfahrens betrachtet werden, sind im Schnitt ca. 86 % der Daten als Spam klassifiziert worden, die tatsächlich auch Spam sind und ca. 80 % von allen Spamdaten sind auch wirklich als Spam klassifiziert worden.
Die Werte der richtig und falsch erkannten Daten sind keinesfalls zufriedenstellend, denn wenn alle Daten als Spam klassifiziert würden, hätte man ca. 71 % richtig und 29 % falsch erkannt. Schaut man sich die Daten im Anhang etwas detailierter an, sieht man, dass die Prozentzahlen von richtig und falsch erkannten Daten schwanken. Auch innerhalb der verschieden sortierten Datensätze schwanken die Ergebnisse stark.

K-nearest neighbor Algorithmus

Nun die Durchschnittswerte des K-nearest neighbor Algorithmus. Zur Auswertung wurde die Klasse 'weka.classifiers.lazy.IBK' des Weka-Klassifizierers benutzt.

Recall	Precision	richtig erkannt	falsch erkannt
0,9993	0,7172	0,7180	0,2820

Die zusammengefassten Daten scheinen auf den ersten Blick ziemlich ähnlich der Daten des NB-Verfahrens. Werden die Daten wieder im Detail betrachtet, sieht man, dass genau das oben beschriebene Szenario eingetreten ist, nämlich, dass fast alle Daten einfach als Spam kategorisiert worden sind. Dieser Umstand ändert sich auch nicht mit größer werdendem 'k'.

Supported Vector Machine

Nun die Durchschnittswerte des SVM-Verfahrens. Zur Auswertung wurde die Klasse 'weka.classifiers.functions.SMO' des Weka-Klassifizierers benutzt.

Recall	Precision	richtig erkannt	falsch erkannt
0,9673	0,7525	0,7486	0,2514

Hier scheinen die Daten wiederum passabel zu sein. Schaut man sich die Auswertung dennoch im Detail an, erkennt man, dass auch hier fast alle Datensätze als Spam kategorisiert worden sind. Zwar nicht in dem gleichen Ausmaß, wie im Falle des K-nearest neighbor Algorithmus, aber trotzdem ist ebenfalls keine Aussage über die Güte des Algorithmus aufgrund der vorliegenden Daten möglich.

Desicion Tree

Nun die Durchschnittswerte des Entscheidungsbaums. Zur Auswertung wurde die Klasse 'weka.classifiers.trees.J48' des Weka-Klassifizierers benutzt.

Recall	Precision	richtig erkannt	falsch erkannt
0,9010	0,7211	0,6794	0,3206

Teilweise kann ein DT erstellt werden, der einfach jeden Datensatz als Spam klassifiziert. Konzentriert man sich nur auf die anderen ausgewerteten Daten, sehen diese auf den ersten Blick besser aus. Betrachtet man hingegen den Baum im Detail, sieht man, dass aufgrund von Wörtern, welche an sich nichts über die Klassifizierung von Spam und Ham sagen dürften, die Daten klassifiziert wurden. Dazu ein Beispiel aus dem Datensatz 5.0. Hier werden die Daten anhand der Worte 'Das', 'Die', 'Ich', 'Für', 'Bad' und 'Hat' klassifiziert. Diese Wörter können in jeder Klasse, ob Spam oder Ham in gleicher Weise auftreten.
Die meisten Klassifizierer scheinen keine aussagekräftigen Daten ergeben zu haben, da der selbst erstellte Korpus von Spam- und Hamdaten viel zu klein ist. Betrachtet man die erstellten Merkmalsvektoren, so erkennt man viele 0-Dimensionen. Auch ist nicht zu unterschätzen, dass es sich hier nur um fiktive Daten handelt. Dies bedeutet, dass sowohl die benutzten Worte, die durchschnittliche Länge aller Daten, als auch der Spam- und Hamanteil aufgrund von Vermutungen erstellt worden sind. Demnach kann eine Fallstudie erst durchgeführt werden, wenn das System eine längere Zeit im Einsatz ist und der Korpus der Spam- und Hamtexte ausreichend groß ist.
Auch mit Hilfe der Entropie war keine weitere Klassifikation der Daten in die Klassen Spam und Ham möglich, da die untersuchten Texte zu kurz sind. Es wurde die absolute Entropie [8] für n-Tupel von Buchstaben mit $n > 0$ und $n <= 5$ untersucht. Wirkliche Ausreißer in den Spamdaten gibt es keine, so, dass keine klare Grenze gezogen werden kann mit Hilfe derer man mit großer Wahrscheinlichkeit sagen könnte, dass ein Text mit einer Entropie größer oder kleiner einer bestimmten Grenze Spam oder Ham ist. Hier eine Tabelle, zwischen welchen Werten die absolute Entropie für n-Tupel mit $n > 0$ und $n <= 5$ streut.

Spam oder Ham	n	Entropie MIN	Entropie MAX
Spam	1	2,0	4,2
Spam	2	2,0	6,2
Spam	3	2,0	6,7
Spam	4	2,0	6,8
Spam	5	2,0	6,9
Ham	1	2,5	4,2
Ham	2	3,4	6,5
Ham	3	3,7	7,3
Ham	4	3,7	7,4
Ham	5	3,7	7,4

Das n in der Tabelle steht für die Tupelgröße. In der Spalte 'Entropie MIN' wird der kleinste Entropiewert der Ham- oder Spamdaten mit einer untersuchten Tupellänge der Größe n eingetragen, in der Spalte 'Entropie MAX' entsprechend der größte. Zur Entropieberechnung wurden nur die Buchstaben 'a-z', 'A-Z' und der Unterstrich '_' ausgewertet.

Die Entropie von 2,0 der Spamdaten ergibt sich aus dem Spamtext 45 des Anhangs C. Der Spamtext 46, solche Art von Spam sollte mittels der Entropieberechnung eindeutig klassifiziert werden, welcher aus wahllos zusammengefügten Buchstaben besteht, besitzt die Entropien mit aufsteigendem n von 1 bis 5: 3.1536, 4.6819, 5.09, 5.2479 und 5.2479.

Zusammenfassende Auswertung von NB, KNN, SVM und DTJ

Um nun eine Wahl für oder gegen einen Klassifizierer (NB, KNN, SVM, DTJ) treffen zu können, müssen die Daten im Anhang D noch etwas genauer betrachtet werden. Hierzu wird mit NB begonnen. Alle in diesem Abschnitt erklärten und analysierten Werte sind im Anhang D zu finden.
In der Tabelle der Testergebnisse mit der laufenden Nummer 3 wurden prozentual die meisten Nachrichten richtig klassifiziert (ca. 81% aller Nachrichten wurden hier richtig klassifiziert). Der niedrigste Wert für die 'richtig erkannt' Daten wurde in der Zeile mit der laufenden Nummer 2 mit der Testmethode 'percentage split' erzielt (hier wurden ca. 69 % aller Nachrichten richtig klassifiziert). Da dieser Wert aber durch die Testmethode 'percentage split' erzielt worden ist, kann davon ausgegangen werden, das die Testmenge nicht wirklich repräsentativ war. Man kann im allgemeinen sagen, dass Ergebnisse, die mittels der Testmethode 'cross validation' gewonnen wurden glaubwürdiger sind. Der größte Recall Wert ist in der Zeile mit der laufenden Nummer 6 abzulesen und der größte Wert für Precision in der Zeile mit der laufenden Nummer 7 (größter Recall: 98%, größter Precision: 85%). Werden die Werte, die mittels der Testmethode 'percentage split' gewonnen wurden außer Acht gelassen, so schwankt der prozentuale Wert für 'richtig erkannt' zwischen ca. 71% und ca. 81%. Betrachtet man die Werte von TP, FP, FN und TN so fällte auf, dass die Werte von TN und FN bei manschen Testergebnissen übermäßig klein werden. Dies fällt in den Spalten mit der laufenden Nummer 3, 6, 2 mit 'percentage split', 5 mit 'percentage split' und besonders bei der Auswertung mit der alufenden Nummer 15 auf. Hier wurden übermäßig viele Daten als Spam klassifiziert.
Der KNN-Algorithmus lieferte auf den Testdaten keine brauchbaren Ergebnisse. Dieser hat in nahezu allen Testläufen fast alle Daten einfach als Spam klassifiziert.
Die Testergebnisse des DTJ ergeben wie oben schon erwähnt ein unterschiedliches Bild. Einige Ergebnisse in den Zeilen mit der laufenden Nummer 3, 6, 15, 17 und 18 sind unbrauchbar, da hier immer alle Texte einfach als Spam klassifiziert worden sind. Die anderen Daten sehen auf den ersten Blick vielversprechend aus, obwohl hier im Gegensatz zu NB der DTJ geneigt ist, die Texte als Spam zu kategorisieren. Dies macht sich in den Werten FN und TN bemerkbar. Besonders die Werte von TN in den Zeilen mit der laufenden Nummer 2, 5, 8, 10, 2 mit 'percentage split', 5 mit 'percentage split', 11, 12, 13, 14 und 16 haben sehr kleine Werte für TN. Das heißt also, dass an diesen Stellen eine übermäßig große Zahl von Hamnachrichten als Spam klassifiziert worden ist. Schaut man sich desweiteren den aus den Daten erstellten Baum an, stellt man fest, dass, wie oben schon erwähnt, anhand von Wortvorkommen klassifiziert wird, die i.A. keine Aussage über Ham und Spam

zulassen dürfen. Dies sind meist Wörter des alltäglichen Gebrauchs wie 'IM' oder 'IST'. Lässt man diese sogenannten Stopwörter bei der Erstellung des Baumes weg, sind die Auswertungsdaten bei weitem nicht mehr so gut. Siehe hierzu die Spalten mit den laufenden Nummern 15 bis 18.
Die Testergebnisse des SVM Verfahrens haben alle kleine FN und TN Werte. Der kleinste FN Wert ist in der Zeile mit der laufenden Nummer 16, 17 und 2 mit 'percentage split' zu finden und ist 1, der Größte zu findende FN Wert ist 7. Der kleinste TN Wert ist 1 und der größte TN Wert ist 16, wobei dieser Wert ein ausreißer ist. Insgesamt lässt sich zu dem SVM Verfahren sagen, dass dieses übermäßig stark geneigt ist die Testtexte als Spam zu kategorisieren.
Der NB Klassifizierer hat mit Abstand die besten Daten geliefert. In vielen Fällen kamen durchaus gute Ergebnisse in einer Art und Weise zustande, dass man annehmen darf, dass diese im laufenden Betrieb noch besser werden. Das heißt, dass hier die guten Zahlen nicht dadurch zustande kamen, dass einfach nahezu alle Daten als Spam kategorisiert worden sind.
Im Gegensatz dazu scheint der KNN für diese Art von Klassifizierungsproblem keine angebrachte Methode zu sein.
Der DTJ hat ebenfalls ein fragwürdiges Ergebnis geliefert, nachdem die erstellten Bäume im Detail angeschaut wurden.
Auch das SVM Verfahren lieferte ein Ergebnis, welches sich dadurch als noch unbrauchbar erwiesen hat, dass zu viele Hamtexte als Spam kategorisiert worden sind. Dies ist insofern auf keinen Fall gewünscht, da man einen Negativeffekt hinsichtlich der zukünftigen Benutzer des Systems ausschließen will. Wenn man annimmt, dass von 3 geschriebenen Nachrichten nur eine auf dem Bildschirm erscheinen wird, kann davon ausgegangen werden, dass dieses System sehr schnell an Akzeptanz verlieren wird.

3.5 Anforderung an das Nachrichtenklassifizierungswerkzeug

Aus der oben gezeigten Analyse der Situation ergeben sich einige Anforderungen an das Programm. Wie beschrieben, darf nur ein begrenzter Benutzerkreis, von dem jeder eindeutig identifizierbar ist, Nachrichten auf die Bildschirme senden. Daraus folgt zwangsläufig, dass jeder Benutzer eine eindeutige Adresse besitzen muss, über die er identifiziert werden kann. Innerhalb dieser Benutzer gibt es eine gewisse Hierarchie. Somit ist es erforderlich, Nachrichten einiger Benutzer ohne Filterung auf den Bildschirm zu senden, Nachrichten anderer Benutzer zu filtern und Nachrichten negativ auffallender Benutzer zu blockieren. Dies muss von dem Programmadministrator einstellbar sein.
Um sowohl Nachrichten mit Badwords, als auch gewisse Abwandlungen der Badwords zu filtern, muss der Programmadministrator in der Lage sein, Badwords einer Liste hinzuzufügen oder auch wieder zu entfernen. Des Weiteren muss der Programmadministrator befähigt sein, einen regulären Ausdruck zu den Badwords hin-

zu zugeben, anhand welchem die zu klassifizierenden Nachrichten untersucht werden können. Für gewisse Klassen von Wörtern ist eine automatisierte Erstellung von regulären Ausdrücken von übergebenen Badwords wünschenswert.
Auch ist es wünschenswert, dass das Programm über ein analysiertes Klassifizierungswerkzeug verfügt. Es ist stark anzunehmen, dass während des Betriebes des Programms aufgrund der dort zugrundeliegenden erhöhten und realistischeren Datenmenge, die Ergebnisse der Klassifizierungswerkzeuge stark verbessert werden. Hierzu wird ein bayesscher Klassifizierer implementiert.
Ebenso wünschenswert ist es, bestimmte Daten anhand ihres 'Aussehens', wie Textmindestlänge, Anzahl verschiedener Buchstaben im Text, Verhältnis Whitespacecharacter zu Textlänge, Verhältnis Anzahl der Sonderzeichen im Text zu Textlänge und Sonderzeichenanzahl mitten im Text, zu klassifizieren. Die Detaileinstellungen sollten in einer Konfigurationsdatei einstellbar sein.
Natürlich muss das Programm ein Interface besitzen, mit dem es Daten jeglicher Art, sei es Spam, Ham, Absender oder Badwords verwalten kann. Selbstverständlich muss auch ein Interface zum Klassifizieren neuer Texte vorhanden sein. Viele der Einstellungen, nicht nur die oben schon genannten, sollten in einer gut beschriebenen Konfigurationsdatei vorgenommen werden können. Andere Einstellungen, eher programmnahe Einstellungen, können im Programm selber eingestellt werden.
Das Programm soll übersichtlich gestaltet sein, damit andere Personen Erweiterungen einfach und schnell vornehmen können. Hierzu gehören natürlich auch ausreichende, programminterne Kommentare.

4 Implementierung

In diesem Kapitel erhalten Sie alle wichtigen Informationen über das zugehörige Programm dieser Arbeit. Es wird beschrieben, wie das Programm gestartet werden kann, es werden wichtige Kommentare zum Programmcode gegeben und es wird beschrieben welche Einstellungen vorgenommen werden können. Auch wird beschrieben, wie die einzelnen Klassifizierer des Programms arbeiten.

4.1 Allgemeiner Überblick

Das Programm ist in Java programmiert und kann sowohl Spam, Ham, noch nicht kategorisierte Nachrichten als auch Absender und Badwords verwalten. Auch wichtige Einstellungen bezüglich der Filtermethoden kann das Programm vornehmen.
Einige Grundeinstellungen können in einer Konfigurationsdatei eingestellt werden. Hierzu gehören neben Datenbankeinstellungen auch Einstellungen für die vorhandenen Filtermechanismen. Die Konfigurationsdatei kann bei Start des Programms aus einem Verzeichnis ausgelesen werden. Ist dort keine Konfigurationsdatei zu finden, wird eine Standardkonfiguration genutzt.
Das Programm kann in einem Konsolenmodus aufgerufen werden, um dort mit verschiedenen Befehlen zum einen die Datenbank zu bearbeiten oder zum anderen den Filter aufzurufen. Möchte man jedoch nur einen Befehl ausführen, sei es, um sich bestimmte Nachrichten aus der Klasse Spam anzuschauen, oder um einen Text mit einem Absender durch einen bestimmten Filter filtern zu lassen, kann das Programm mit dem entsprechenden Parameter gestartet werden.

4.2 Anforderungen

Um das Programm kompilieren zu können und später zu starten ist eine Installation des *JDK 1.4* oder höher erforderlich. Das Programm arbeitet auf einer darunterliegenden *MYSQL Datenbank*. Der *MYSQL JDBC Driver*, um auf die *MYSQL Datenbank* zugreifen zu können, liegt dem Programm bei. Desweiteren werden noch zwei Pakete benötigt, um *Connection Pooling* zu realisieren. Diese liegen ebenfalls dem Programm bei (*commons-dbcp-1.2.2.jar*, *commons-pool-1.4.jar*). Ausserdem wird ein *MYSQL 5.0* Datenbankserver benötigt.

4.3 Programmstart

Um nach Einstellen der Konfigurationen in die Programmkonsole zu gelangen, wird das Programm einfach ohne Parameter aufgerufen.

```
java -jar Filter.jar
```

Sollte man die Verzeichnisstruktur in der das mitgelieferte Programm als jar-Datei liegt geändert haben, ist dafür Sorge zu tragen, dass die oben erwähnten jar-Dateien in den Klassenpfad eingebunden werden.
Dass man sich in der Programmkonsole befindet, erkennt man daran, dass die Zeile des Konsolenfensters nun mit '»' beginnt. Jetzt kann der Nutzer des Programms die Programmbefehle eingeben, welche in Kapitel 4.4 beschrieben sind. Ist der Befehl vollständig eingegeben, kann dieser mit der Returntaste ausgeführt werden.
Möchte man hingegen nur einen oder wenige Befehle ausführen, kann man den entsprechenden Befehl an das Programm übergeben. Hierzu kann der Programmparameter '-com' genutzt werden, gefolgt von dem entsprechenden Befehl. Ein Aufruf für einen SHOW-Befehl sähe beispielsweise wie folgt aus:

```
java -jar Filter.jar -com "SHOW Spam 12..50"
```

Dieser Befehl würde alle Spamnachrichten mit einer id >= 12 und einer id <= 50 ausgeben. Es sei zum besseren Verständnis dieses Beispieles vorweg gesagt, dass alle Befehle Groß- und Kleinschreibung ignorieren.
Eine Liste der Programmbefehle mit Beschreibung ist in Anhang 6 zu finden.

4.4 Befehle

Grundsätzlich gibt es fünf verschiedene Arten oder Typen von Befehlen, die meist mit verschiedenen Befehlsparametern, welche im Weiteren auch Befehlselemente genannt werden, aufgerufen werden können. Es sind die Befehle 'SHOW', 'ADD', 'DELETE', 'CHANGE' und 'FILTER'. Nun wird die Auswirkung der einzelnen Befehle und Befehlsparameter erklärt. Um eine übersichtliche Programmbefehlssyntax zu erhalten wird auf Anhang 6 verwiesen.

Funktionsüberblick

Der SHOW-Befehl dient dem Anzeigen verschiedener in der Datenbank befindlicher Daten. Mit Hilfe des FILTER-Befehls kann ein bestimmter Text analysiert werden. Die restlichen Befehle dienen der Manipulation der Daten. Wie die Namen der Befehle schon sagen, können mit dem ADD-Befehl Daten hinzugefügt, mit dem DEL-Befehl Daten entfernt und mit dem CHANGE-Befehl Daten verschoben werden. Der CHANGE-Befehl dient vor allem dazu, noch nicht sortierte Texte in die entsprechenden Kategorien zu sortieren. Texte können auch von der einen Kategorie in die andere Kategorie verschoben werden.

SHOW-Befehl

Der SHOW-Befehl, welcher wie erwähnt zum Anzeigen von Daten dienlich ist, erwartet zwingend direkt nach der Angabe von *SHOW* die Angabe, welche Art von Inhalt angezeigt werden soll. Hierzu kann entweder der Befehlsparameter *SPAM*, *HAM*, *UNCERTAIN*, *ADDRESS* oder *BADWORD* übergeben werden. Der Befehlsparameter *SPAM* zeigt an, dass der entsprechende Befehl auf die Spamdaten auszuführen ist, *HAM*, dass dieser auf die Hamdaten auszuführen ist und *UNCERTAIN*, dass dieser auf die noch nicht kategorisierten Texte auszuführen ist. Der Befehlsparameter *BADWORD* gibt an, dass der Befehl auf die Daten der nicht zulässigen Wörter auszuführen ist und *ADDRESS* auf die Daten der Absender. Dies gilt für die Befehle *ADD*, *DEL* und *CHANGE* in gleicher Weise. Ohne weitere Parameter würde dieser Befehl alle Daten der entsprechenden Kategorie anzeigen. Dies kann aber sehr ungünstig sein, wenn man nur einen bestimmten Dateneintrag oder einen bestimmten Bereich von Daten durchsuchen möchte. Um diese Möglichkeit zu bieten, kann zum einen der Befehlsparameter *MAXLENGTH* übergeben werden und zum anderen besteht die Möglichkeit, sich einen Text einer bestimmten 'id' oder eines bestimmten 'id'-Bereiches, anzeigen zu lassen. Mit 'id' ist die Datenbankid gemeint, welche gleichzeitig der Primarykey der entsprechenden Datenbanktabelle ist. Nähres hierzu befindet sich in Abschnitt 4.6.
Der oben erwähnte Parameter *MAXLENGTH* wird dazu verwendet, die Maximallänge der anzuzeigenden Texte anzugeben, welche direkt hinter dem Parameter durch ein Gleichheitszeichen getrennt zu setzen ist. Bei einer Maximallänge von 753 sähe dass wie folgt aus:

```
MAXLENGTH=753
```

Dieser Befehlsparameter kann nicht zusammen mit dem Befehlsparameter *BADWORD* benutzt werden.
Um die Nachricht nur einer bestimmten 'id' anzuzeigen, kann diese einfach hinter den Befehl angehängt werden. Möchte man einen 'id'-Bereich betrachten, wird dieser durch zwei Punkte getrennt, wobei die untere Schranke, somit auch die kleinere Zahl, links der beiden Punkte steht und die obere Schranke, also somit auch die größere Zahl, rechts der beiden Punkte steht. Ein Bereich von einschließlich der 'id' 876 bis einschließlich der 'id' 7777 würde wie folgt angegeben:

```
876..7777
```

ADD-Befehl

Auch der ADD-Befehl erwartet direkt nach der Eingabe von *ADD* einen der Befehlsparameter *SPAM*, *HAM*, *UNCERTAIN*, *ADDRESS* oder *BADWORD*. Er dient dem Hinzufügen von Daten. Die Bezüge der Befehlsparameter *SPAM*, *HAM*, *UNCERTAIN*, *ADDRESS* und *BADWORD* sind die gleichen wie bei dem schon beschriebenen SHOW-Befehl. Im Zusammenhang mit den Parametern *SPAM*, *HAM*

oder *UNCERTAIN* muss der Text, welcher den Daten hinzugefügt werden soll, durch den Befehlsparameter *TEXT*, mit einem Gleichheitszeichen getrennt und von runden Klammern umschlossen, übergeben werden. Wird der Parameter *ADDRESS* genutzt, wird der Absender, durch den Parameter mit Gleichheitszeichen *ADDR* getrennt übergeben. Bei einem Absender 'HaNs' sieht das wie folgt aus:

```
ADDR=HaNs
```

Groß- und Kleinschreibung wird bei der Absenderadresse nicht beachtet. Das bedeutet, dass sowohl 'HaNs' als auch 'HANS' hinzugefügt werden können, es aber bei der Auswertung keinen Unterschied macht. Die Angabe des zweiten Absenders 'HANS' würde die Daten des Ersten überschreiben. Wenn gewünscht, können noch die Optionen *STOP* und *FILTER* übergeben werden. Durch ein Gleichheitszeichen getrennt wird durch eine 1 für true und eine 0 für false angegeben, ob dieses oder jenes Flag für diesen Absender gesetzt werden soll. Der Nutzen dieser Flags und deren Auswertung werden bei der Funktionsweise der Filtermechanismen beschrieben. Wird der ADD-Befehl mit dem Parameter *BADWORD* benutzt, muss ein weiterer Parameter, nämlich der Parameter *NORM*, zwingend gesetzt sein. Hier wird durch ein Gleichheitszeichen getrennt das 'normale Wort' in runden Klammern direkt an diesen Parameter gesetzt. Das 'normale Wort' ist das Wort, nach dem der Text gefiltert werden soll. Komplexere Abwandlungen hierzu können durch einen regulären Ausdruck beschrieben werden, der mittels des Parameters *REG*, wieder durch ein Gleichheitszeichen getrennt und von runden Klammern umschlossen, angegeben wird. Wie gültige reguläre Ausdrücke aussehen, finden Sie in der Spezifikation [11]. Wird zu einem 'normalen Wort' kein regulärer Ausdruck übergeben, wird ein regulärer Ausdruck automatisch erzeugt. Wie der reguläre Ausdruck erzeugt werden soll, wird in der Konfigurationsdatei angegeben. Hier noch ein wichtiger Hinweis: Kommen in dem normalen Wort oder in dem regulären Ausdruck eines Badwords runde Klammern vor, so muss diesen ein Backslash vorangestellt werden. Dieses wird nicht mit ausgewertet.

DELETE-Befehl

Der 'DELETE-Befehl' löscht Daten einer bestimmten 'id', wieder die der Datenbank, bzw. die eines bestimmten 'id'-Bereiches. Zwingend erforderlich ist die Angabe einer der Befehlsparameter *SPAM*, *HAM*, *UNCERTAIN*, *ADDRESS* oder *BADWORD*. Welcher Parameter sich auf welche Daten bezieht, wurde oben beschrieben. Nun muss noch eine 'id' angegeben werden, welche gelöscht werden soll. Soll ein 'id'-Bereich gelöscht werden, kann die '..'-Schreibweise, wie oben beschrieben, benutzt werden. Der DELETE-Befehl fragt vor dem Löschen nicht nach, ob wirklich gelöscht werden soll. Soll vor der Eingabe gefragt werden, ob wirklich gelöscht werden soll, ist der DEL-Befehl zu verwenden.

FILTER-Befehl

Der FILTER-Befehl erwartet zwingend den Parameter *TEXT*. Hinter diesen Parameter wird durch ein Gleichheitszeichen getrennt in runden Klammern der zu filternde Text angegeben. Mit welchem Filter(n) dieser Text nun analysiert werden soll, wird in der Konfigurationsdatei angegeben. Abweichend davon können gewisse Filter zur Programmlaufzeit ein- oder ausgeschaltet werden. Hierzu stehen die Befehlsparameter *NBO*, *RUL*, *ADDRCHECK* und *BADW* zur Verfügung. Wird einer dieser Parameter übergeben, so wird der entsprechende Filter mit den spezifischen Einstellungen genutzt, um den übergebenen Text nach Ham oder Spam zu klassifizieren. Dabei bezieht sich der Parameter *RUL* auf den regelbasierten Filter, *NBO* auf die bayessche Klassifizierungsmethode, *BADW* auf die Badwordklassifizierung und *ADDRCHECK* auf die Absenderüberprüfung. Zur Absenderüberprüfung ist der Parameter *ADDRESSOR* zwingend erforderlich. Der Absender wird, durch ein Gleichheitszeichen getrennt in runden Klammern eingeschlossen, an den Parameter *ADDRESSOR* gehängt. Ist die Absenderüberprüfung ausgeschaltet, wird die Nachricht jedes Absenders nur mittels der angegebenen Klassifizierer klassifiziert. Bei ausgeschalteter Absenderüberprüfung ist die Absenderübergabe nicht erforderlich. Ist die Absenderüberprüfung eingeschaltet und der Absender unbekannt, so wird diese Nachricht als Spam klassifiziert. Hier ist nicht gemeint, dass eine klassifizierte Nachricht auch direkt den entsprechenden Daten zugeordnet wird, sondern vielmehr, dass der Benutzer oder Anwender des Programms die Rückmeldung erhält, dass es sich bei dieser Nachricht um Spam handelt. Gefilterte Nachrichten werden automatisch in den *UNCERTAIN* Daten gespeichert und können vom Programmadministrator zu einem späteren Zeitpunkt in den entsprechenden Ordner verschoben werden. Auch bei der Parameterübergabe des FILTER-Befehls müssen den runden Klammern im zu filternden Text und im Absender ein Backslash vorangesetzt werden.
Zwei bis jetzt noch nicht erwähnte Programmbefehle sind *QUIT* und *EXIT*. Beide beenden das Programm, sofern dieses im Konsolenmodus gestartet worden ist.

4.5 Konfigurationsdatei

In der Konfigurationsdatei werden die wichtigsten Programmeinstellungen vorgenommen. Wie eine solche Konfigurationsdatei aussieht und welche Einstellungen dort vorgenommen werden können ist in der Konfigurationsdatei selber, wie auch im Anhang B beschrieben. Dort sehen Sie den Auszug der Defaultkonfigurationsdatei, wie sie bei Auslieferung des Programms vorzufinden ist. An dieser Stelle wird nur eine kurze Auflistung über die Möglichkeiten gezeigt, was in der Konfigurationsdatei einstellbar ist. Die Konfigurationsdatei muss relativ vom Programm aus in einem Verzeichnis 'data' liegen.

- Dateilog ein- oder ausschalten
- Standardausgabenlog ein- oder ausschalten

- Bayessche Klassifizierungsmethode ein- oder ausschalten
- Badworduntersuchung ein- oder ausschalten
- Regelbasierten Klassifizierer ein- oder ausschalten
- Wenn regelbasierter Klassifizierer eingeschaltet ist, Angabe der Mindestzeichenanzahl des Textes
- Wenn regelbasierter Klassifizierer eingeschaltet ist, Angabe, wieviele verschiedene Buchstaben im Text mindestens vorhanden sein müssen
- Wenn regelbasierter Klassifizierer eingeschaltet ist, Angabe vom Höchstvorkommen von Whitespacecharactern in Prozent
- Wenn regelbasierter Klassifizierer eingeschaltet ist, Angabe vom Höchstvorkommen von Sonderzeichen in Prozent
- Wenn regelbasierter Klassifizierer eingeschaltet ist, Angabe der maximalen Sonderzeichenanzahl, die zwischen zwei Buchstaben oder Zahlen steht
- Wahl einer Strategie zur automatischen Generierung regulärer Ausdrücke
- Mindestwortlänge, die bei der bayesschen Klassifizierungsmethode genutzt werden soll
- Absenderüberprüfung ein- oder ausschalten
- Angabe der Datenbankinformationen

4.6 Datenbank

Ein Datenbankdump liegt dem Programm bei. Die Datenbank besteht aus sechs Tabellen (Abbildung 4.1) :

```
mysql> show tables;
+---------------------+
| Tables_in_spamfilter |
+---------------------+
| addressors          |
| badwords            |
| bayes_words         |
| ham                 |
| not_sorted_text     |
| spam                |
+---------------------+
6 rows in set (0.00 sec)
```

Abbildung 4.1: Datenbanktabellen

```
mysql> describe addressors;
+-----------+------------+------+-----+---------+----------------+
| Field     | Type       | Null | Key | Default | Extra          |
+-----------+------------+------+-----+---------+----------------+
| id        | int(11)    | NO   | PRI | NULL    | auto_increment |
| addressor | text       | YES  |     | NULL    |                |
| filter    | tinyint(1) | YES  |     | NULL    |                |
| stop      | tinyint(1) | YES  |     | NULL    |                |
+-----------+------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

Abbildung 4.2: Beschreibung der Tabelle 'addressors'

```
mysql> describe badwords;
+--------------+---------+------+-----+---------+----------------+
| Field        | Type    | Null | Key | Default | Extra          |
+--------------+---------+------+-----+---------+----------------+
| id           | int(11) | NO   | PRI | NULL    | auto_increment |
| normal_word  | text    | YES  |     | NULL    |                |
| regular_word | text    | YES  |     | NULL    |                |
+--------------+---------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

Abbildung 4.3: Beschreibung der Tabelle 'badwords'

- addressors - Hier werden die Absender verwaltet
- badwords - Hier werden die Badwords verwaltet
- bayes_words - Eine Liste von Wörtern, die bei der Klassifizierung mittels des bayesschen Klassifizierers generiert wurde
- ham - Hier werden alle Hamtrainingsdaten gespeichert
- spam - Hier werden alle Spamtrainingsdaten gespeichert
- not_sorted_text - Hier sind die noch nicht zu den Trainingsdaten hinzugefügten Texte gespeichert

Eine Beschreibung der Tabelle Addressors, wie sie von der Datenbank mittels des 'describe' Befehls zurück gegeben wird ist in Abbildung 4.2 zu sehen. Die Spalte 'id' ist die eindeutige ID des Absenders. Die Namen der Absender stehen in der Spalte 'addressor'. Die Spalten 'stop' und 'filter' sind entweder null oder eins. Wie diese ausgewertet werden, wurde bereits erklärt.
Eine Beschreibung der Tabelle 'badwords', wie sie mittels des 'describe' Befehls zurück gegeben wird, ist in Abbildung 4.3 zu sehen. Die Spalte 'id' ist die eindeutige ID der Badwords. In der Spalte 'normal_word' ist das Badword in seiner normalen Form, in der Spalte 'regular_word' sind die zu den normalen Badwords regulären Ausdrücke gespeichert.

```
mysql> describe spam;
+-------------+-----------+------+-----+-------------------+----------------+
| Field       | Type      | Null | Key | Default           | Extra          |
+-------------+-----------+------+-----+-------------------+----------------+
| id          | int(11)   | NO   | PRI | NULL              | auto_increment |
| spam_text   | text      | YES  |     | NULL              |                |
| insert_time | timestamp | YES  |     | CURRENT_TIMESTAMP |                |
+-------------+-----------+------+-----+-------------------+----------------+
3 rows in set (0.00 sec)

mysql> describe ham;
+-------------+-----------+------+-----+-------------------+----------------+
| Field       | Type      | Null | Key | Default           | Extra          |
+-------------+-----------+------+-----+-------------------+----------------+
| id          | int(11)   | NO   | PRI | NULL              | auto_increment |
| ham_text    | text      | YES  |     | NULL              |                |
| insert_time | timestamp | YES  |     | CURRENT_TIMESTAMP |                |
+-------------+-----------+------+-----+-------------------+----------------+
3 rows in set (0.00 sec)
```

Abbildung 4.4: Beschreibung der Tabellen 'spam' und 'ham'

```
mysql> describe not_sorted_text;
+----------------+---------+------+-----+---------+----------------+
| Field          | Type    | Null | Key | Default | Extra          |
+----------------+---------+------+-----+---------+----------------+
| id             | int(11) | NO   | PRI | NULL    | auto_increment |
| uncertain_text | text    | YES  |     | NULL    |                |
| addressor      | text    | YES  |     | NULL    |                |
+----------------+---------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

Abbildung 4.5: Beschreibung der Tabelle 'not_sorted_text'

Die Tabellen 'spam' und 'ham' sind nahezu identisch aufgebaut. Siehe hierzu Abbildung 4.4. In der Spalte 'id' ist ein eindeutiger Schlüssel zu den Ham- bzw. Spamtexten zu finden, wie sie in der Spalte 'ham_text' bzw. 'spam_text' gespeichert sind. Die Spalte 'insert_time' gibt an, wann die entsprechenden Spalten hinzugefügt wurden.
In der Tabelle 'not_sorted_text', siehe hierzu Abbildung 4.5, hat jeder Datensatz wieder einen eindeutigen Schlüssel, der in der Spalte 'id' gespeichert ist. In der Spalte 'uncertain_text' ist der noch nicht zu den Trainingsdaten klassifizierte Text. Jeder vom Programm klassifizierte Text wird hier gespeichert, bis dieser verschoben oder gelöscht wird. Von wem dieser Text stammt, wird in die Tabelle 'addressor' eingetragen. Bei unbekanntem Absender ist der entsprechende Eintrag in der 'addressor' Tabelle 'NULL'.
Die Tabelle 'bayes_words', siehe hierzu Abbildung 4.6, hat wieder eine Spalte 'id', in der wieder ein eindeutiger Schlüssel jedes Datensatzes eingetragen ist. Die generierten Wörter sind in der Spalte 'word' eingetragen. Die Spalten 'count_spam' und 'count_ham' beinhalten positive ganze Zahlen, deren Nutzen zu einem späteren Zeitpunkt ersichtlich wird. In der Spalte 'generated_time' ist die Zeit des

```
mysql> describe bayes_words;
+----------------+-----------+------+-----+-------------------+----------------+
| Field          | Type      | Null | Key | Default           | Extra          |
+----------------+-----------+------+-----+-------------------+----------------+
| id             | int(11)   | NO   | PRI | NULL              | auto_increment |
| word           | text      | YES  |     | NULL              |                |
| count_spam     | int(11)   | YES  |     | NULL              |                |
| count_ham      | int(11)   | YES  |     | NULL              |                |
| generated_time | timestamp | YES  |     | CURRENT_TIMESTAMP |                |
+----------------+-----------+------+-----+-------------------+----------------+
5 rows in set (0.00 sec)
```

Abbildung 4.6: Beschreibung der Tabelle 'bayes_words'

Hinzufügens der einzelnen Wörter gespeichert.

4.7 Funktionsweise der implementierten Filtermethoden

Im Weiteren werden die Einstellungsmöglichkeiten und die Funktionsweise der einzelnen Filter beschrieben.

Addressor Filter

Wird der Text mit dem Absenderfilter klassifiziert, ist die Angabe eines Absenders, wie in Kapitel 4.4 beschrieben, notwendig. Ist dieser Absender in der Datenbank nicht vorhanden, wird dieser Text grundsätzlich als Spam klassifiziert. Der Absender besitzt zwei Eigenschaften. Zum einen die Eigenschaft 'stop'. Diese kann entweder wahr oder falsch sein. In der Datenbank wird für wahr eine 1 und für falsch eine 0 gesetzt. Ist die Eigenschaft eines Absenders mit einer 1 gesetzt, so wird jeder von diesem Absender zu klassifizierende Text als Spam klassifiziert. Ist diese Eigenschaft hingegen mit einer 0 gesetzt, wird die andere Eigenschaft namens 'filter' ausgewertet. Ist die Eigenschaft in diesem Falle mit einer 0 gesetzt, werden Nachrichten von diesem Absender immer als Ham klassifiziert. Ist sie hingegen mit einer 1 gesetzt obliegt es den anderen Klassifizierungsmethoden, ob diese Nachricht Spam oder Ham ist.

regelbasierter Filter

Der regelbasierte Filter wertet den Text anhand bestimmter Regeln aus. Nach welchen Regeln der Text ausgewertet werden soll, kann in der Konfigurationsdatei angegeben werden. Es kann bestimmt werden, wieviele Zeichen der Text mindestens haben muss, aus wievielen verschiedenen Buchstaben dieser mindestens bestehen muss, wieviel Prozent der Zeichen des Textes höchstens Sonderzeichen sein dürfen, wieviel Prozent der Zeichen des Textes höchstens Whitespacecharacter sein dürfen und wieviele Sonderzeichen höchstens von Buchstaben oder Zahlen direkt umschlossen sein dürfen. Wie diese einzelnen Filter ein- bzw. ausgestellt werden können, kann der Dokumentation der Konfigurationsdatei entnommen werden.

Naive Baise Klassifizierer

Vorweg sei gesagt, dass der bayessche Klassifizierer auf vorberechnete Daten in der Datenbank zurückgreift. Um die Wahrscheinlichkeit zu berechnen, ob ein vorliegender Text Ham oder Spam ist, wird die Anzahl der einzelnen Wörter aus den Hamdaten und aus den Spamdaten benötigt. Diese Daten können vorberechnet werden, so dass bei dem eigentlichen Klassifizierungsvorgang der Rechenaufwand relativ gering ist. Dies gilt jedoch nicht immer.
Wie oben der Datenbankbeschreibung entnommen werden kann, ist in den Tabellen 'ham' und 'spam', in denen die Ham- und Spamdaten gespeichert sind, eine Spalte 'insert_time', die als Defaultwert immer den aktuellen Zeitstempel setzt. Werden also neue Daten hinzugefügt, wird hier der Zeitstempel des Hinzufügens gespeichert. Genauso ist es in der Tabelle 'bayes_words'. Hier wird in der Spalte 'gene-

rated_time' immer der aktuelle Zeitstempel bei Speicherung des entsprechenden Wortes angegeben. 'bayes_words' ist gleichzeitig die Tabelle, die nach einmaliger Generierung die oben beschriebenen Spam- und Hamwörter enthalten soll. Vor jeder Klassifizierung mittels des Naive Baise Verfahrens wird der aktuellste Zeitstempel der Tabelle 'bayes_words' mit dem aktuellsten Zeitstempel der Tabellen 'spam' und 'ham' verglichen. Ist der Zeitstempel der 'ham' oder 'spam' Tabelle aktueller, als jener der 'bayes_words', werden die Worte neu generiert, da nun veränderte Daten im Vergleich zu dem Zeitpunkt vorliegen müssen, als die Wörter generiert worden sind.
Auch wird vor jeder Klassifizierung mittels des bayesschen Filters die aktuelle Anzahl der Zeilen in den Tabellen 'ham' und 'spam' mit der Anzahl der 'spam' und 'ham' Zeilen zum Zeitpunkt der Generierung verglichen. Sind diese unterschiedlich, wurden entweder Spam- oder Hamdaten gelöscht oder hinzugefügt.
Natürlich werden die Wörter auch generiert, wenn diese noch nie generiert wurden, sprich die Tabelle 'bayes_words' keine Einträge besitzt. Dies sind die drei Szenarien, an denen die Generierung der verschiedenen Wörter aus den Spam- und Hamdaten erneut vorgenommen wird. Bei der Wortgenerierung wird die Groß- und Kleinschreibung ignoriert.
Des Weiteren werden vor jeder Filterung die Gesamtzahlen aller Spam bzw. Hamwörter ermittelt. Diese werden für die Klassifizierung des aktuell zu klassifizierenden Textes zwischengespeichert.
Die Unterscheidung, ob ein bestimmter Text als Ham oder als Spam klassifiziert wird, wird zum Teil auf Grundlage des vorgestellten Naive Baise Verfahrens von Paul Graham entschieden. [5] In dem Verfahren von Paul Graham wird eine Höchstanzahl an Wörten aus einem Text gesucht, die die größte Aussage über die Kategorisierung nach Spam und Ham machen. Auf die Angabe einer Höchstanzahl wird in diesem Programm verzichtet, da die Nachrichtentexte nie mehr als 250 Zeichen besitzen und diese meist dadurch in ihrer Wortanzahl sehr gering sind. Paul Graham entwickelte das Verfahren für E-Mails, hier können natürlich eine Vielzahl von Wörter auftreten. Um die entgültige Klassifizierung vorzunehmen, wurde auf das Verfahren von Robinson [9] zurückgegriffen.
Um die Wörter zu klassifizieren muss die Spamwahrscheinlichkeit jedes Wortes, welches im zu klassifizierenden Text vorkommt, berechnet werden. Bleibt die Frage, welche Wahrscheinlichkeit solche Wörter besitzen, die sowohl in den Hamdaten als auch in den Spamdaten nicht vorhanden sind und welche Wahrscheinlichkeit Wörter besitzen, die nur in den Ham- bzw. nur in den Spamdaten vorhanden sind. Ein Wort, welches nur in den Hamdaten vorkommt, bekommt eine Spamwahrscheinlichkeit von 0,0001. Ein Wort, welches nur in den Spamdaten, aber nicht in den Hamdaten auftaucht, bekommt eine Spamwahrscheinlichkeit von 0,9999. Ist ein Wort sowohl in den Spam- als auch in den Hamdaten nicht vorhanden, so bekommt dieses eine Spamwahrscheinlichkeit von 1,0. Diese Wahrscheinlichkeiten können in der Methode 'calculateProbForOneWord' in der Klasse 'NaiveBaiseFilter' geändert werden.

Badwordfilter - Reguläre Ausdrücke

Es besteht die Möglichkeit den Text nach bestimmten Badwords zu durchsuchen. Kommt eines dieser Badwords in dem zu filternden Text vor, ist dieser Text mit großer Wahrscheinlichkeit Spam. Man kann nicht grundsätzlich davon ausgehen, dass ein Text, in dem ein Badword gefunden wurde, als Spam zu klassifizieren ist. Diese Problematik wurde schon in Abschnitt 3 besprochen.

Nach welchen Wörtern der Text zu durchsuchen ist, kann vom Benutzer des Programms festgelegt werden. Denn es ist möglich, einen regulären Ausdruck mit dem Badword zu übergeben, mittels welchem die zu klassifizierenden Texte durchsucht werden. Ist in dem Text ein Wort, welches mittels des regulären Ausdrucks erzeugt werden kann, so wurde mit hoher Wahrscheinlichkeit ein Badword gefunden. Das 'normale Wort', das zwangsweise bei dem Hinzufügen eines neuen Badwords mittels des 'ADD-Befehls' übergeben wird, soll der Identifizierung des regulären Ausdrucks zugrunde liegenden Wortes dienen. Dies macht insofern Sinn, da es bei der Suche durch den Benutzer nach bestimmten Wörtern in der vorhandenen Badwordliste der Datenbank das Wort mit Abwandlungen dargestellt, durch einen regulären Ausdruck nur schwer zu erkennen ist. Da der zu filternde Text nur mittels des regulären Ausdrucks durchsucht wird, kann der Text im grunde genommen durch eine Vielzahl von Wörtern durchsucht werden, die ähnlich dem zugehörigen 'normalen Wort' sind. Wird beispielsweise das Badword 'dummkopf' hinzugefügt und ein passender regulärer Ausdruck '[dD][uU][mM][mM][kK][oO][pP][fF]' übergeben, so würde das Programm nicht nur in dem Text 'Du bist ein dummkopf' das Badword finden, sondern auch in dem beispielsweise zu filternden Text 'Du bist ein DuMMkOpf'. Für einen Überblick und eine Erklärung der regulären Ausdrücke, wie sie verwendet werden, betrachten Sie bitte [11].

Wie man der Spezifikation des entsprechenden Programmbefehles in Kapitel 4.4 entnehmen kann, muss nicht zwangsläufig vom Benutzer ein regulärer Ausdruck übergeben werden. Geschieht dies nicht, so wird automatisiert ein regulärer Ausdruck passend zu dem normalen Wort erzeugt. Auf welche Weise dies geschehen soll, kann in der Konfigurationsdatei angegeben werden. Betrachten Sie hierzu Anhang B.

Dem Benutzer bleiben sechs Möglichkeiten um auszuwählen, in welcher Art und Weise ein regulärer Ausdruck erzeugt werden soll. Im Allgemeinen lässt sich sagen, dass die regulären Ausdrücke um so mächtiger sind, je größer die Zahl ist mit der sie in der Konfigurationsdatei gewählt werden. Jedoch ist davon abzuraten, für jedes normale Wort die Konstruktion des regulären Ausdrucks mittels des mächtigsten Verfahrens zu nutzen. Am besten ist immer eine vom Benutzer durchdachte eigene Repräsentation des Wortes durch einen regulären Ausdruck. Jedoch kann dies gerade bei großen Listen, Badwordlisten, aus denen mehrere tausend Badwords hinzugefügt werden sollen, zu einen unzumutbaren Aufwand führen. Wird jedoch jedes Wort aus dieser Badwordliste so hinzugefügt, dass der reguläre Ausdruck immer durch das mächtigste Verfahren erzeugt wird, kann es in Zukunft bei der Klassifizierung von Texten zu einer Vielzahl von falsch positiven Klassifizierungen kommen,

da Abwandlungen des Wortes öfter in anderen Worten oder in Teilen von Sätzen vorkommen. Im Falle einer Wortliste sind geeignete Klassen von Wörtern zu bilden, bei denen die Worte der einen Klasse beispielsweise durch das Verfahren 3 den Badworddaten hinzugefügt wird und die Worte der anderen Klasse durch das Verfahren 5. Wie die einzelnen Verfahren funktionieren und was die Zahl hinter dem Wort 'Verfahren' angibt ist in diesem Abschnitt weiter unten beschrieben. Die Wörter könnten beispielsweise aufgrund ihrer Länge zu verschiedenen Klassen zusammengefasst werden.
Im Weiteren werden auch die Verfahren mit einer Zahl zwischen 0 und 5 unterschieden. Diese ist die Zahl in der Konfigurationsdatei der Option 'reg_ex_strategy'. Wird beispielsweise von dem 'Verfahren 3' gesprochen, so ist dies das Verfahren zur automatisierten Erzeugung regulärer Ausdrücke, welches ausgeführt wird, wenn in der Konfigurationsdatei die Option 'reg_ex_strategy' mit 3 gesetzt wurde.
Nun folgt eine nähere Erläuterung der Verfahren, wie die regulären Ausdrücke aus den vom Benutzer übergebenen Wörtern automatisch erzeugt werden. Ausserdem wird versucht ein intuitives Gefühl durch passende Beispiele zu bilden, welche Abwandlungen der Badwords, die für den Menschen noch verstehbar sind, durch diese Art der Erzeugung des regulären Ausdrucks gefunden werden. Der reguläre Ausdruck wird immer Buchstabe für Buchstabe aus dem normalen Wort erzeugt. Die so erzeugten Teile in der Reihenfolge belassen wieder konkateniert, bilden den entsprechenden Ausdruck des übergebenen normalen Wortes. Somit genügt die Angabe, welches Zeichen durch welchen regulären Ausdruck ersetzt wird.

Erklärung des Verfahrens 0 zur automatischen Erzeugung regulärer Ausdrücke

Das einfachste Verfahren ist das Verfahren 0. Dieses wird standardmäßig genutzt, sprich, wenn keine oder eine defekte Konfigurationsdatei gegeben ist oder in der Konfigurationsdatei die entsprechende Option nicht gesetzt ist. Hier wird die Mächtigkeit von regulären Ausdrücken kaum ausgenutzt. Aus der übergebenen Eingabe wird ein regulärer Ausdruck erzeugt, der genau die Wörter findet, bei denen fast jedes Zeichen mit dem normalen Wort übereinstimmt. Eine einfache Änderung der Groß- Kleinschreibung in einem zu filternden Text würde schon zu keiner Übereinstimmung mehr führen. Die Buchstaben groß A und klein a bis groß Z und klein z, so wie die Ziffern 0 bis 9 und der Unterstrich '_' bleiben im regulären Ausdruck erhalten. Die Sonderzeichen werden durch den regulären Ausdruck \p{Punct} ersetzt. Das Eurozeichen '€' bleibt im regulären Ausdruck ebenfalls erhalten. Die Whitespacecharacters werden durch den regulären Ausdruck \s ersetzt. Alle weiteren Zeichen werden durch den regulären Ausdruck '.' repräsentiert, der wie in der Spezifikation der regulären Ausdrücke [11] zu lesen ist, ein beliebiges Zeichen repräsentiert.

```
Beispiele:

Idiot - idiot
arMleuchter112 - arMleuchter112
ange!ber! - ange\p{Punct}ber\p{Punct}
Kinds Kopf - Kinds Kopf
```

Erklärung des Verfahrens 1 zur automatischen Erzeugung regulärer Ausdrücke

Ein etwas mächtigeres Verfahren zur Erzeugung der regulären Ausdrücke ist das Verfahren 1. Hier werden die Buchstaben im Gegensatz zu Verfahren 0 durch einen regulären Ausdruck repräsentiert, der sowohl den entsprechenden Klein-, als auch den entsprechenden Großbuchstaben repräsentiert. Ein beliebiger kleiner Buchstabe i wird zu dem regulären Ausdruck '$[i\ I]$', wobei I der entsprechende Großbuchstabe des Kleinbuchstabens i ist. Analog hierzu wird auch ein beliebiger Großbuchstabe I zu dem regulären Ausdruck '$[i\ I]$'. Alle anderen Zeichen werden analog zu dem Verfahren 0 zu einem regulären Ausdruck umgewandelt.

```
Beispiele:

Idiot - [iI][dD][iI][oO][tT]
arMleuchter112 - [aA][rR][mM][lL][eE][uU][cC][hH][tT][eE][rR]112
ange!ber! - [aA][nN][gG][eE]\p{Punct}[bB][eE][rR]\p{Punct}
Kinds Kopf - [kK][iI][nN][dD][sS] [kK][oO][pP][fF]
```

Erklärung des Verfahrens 2 zur automatischen Erzeugung regulärer Ausdrücke

Das Verfahren 2 wandelt alle Zeichen identisch dem Verfahren 1 in einen regulären Ausdruck um, bis auf die Whitespacecharacters. Diese werden nun durch den regulären Ausdruck '\s*' ersetzt. Dadurch können nun auch Badwords im zu klassifizierendem Text gefunden werden, welche sich von dem normalen Wort durch die Anzahl der Whitespacecharacters unterscheiden, sei es, dass Whitespacercharacters weggelassen werden, oder sei es, dass welche hinzugefügt werden.

```
Beispiele:

Idiot - [iI][dD][iI][oO][tT]
arMleuchter112 - [aA][rR][mM][lL][eE][uU][cC][hH][tT][eE][rR]112
ange!ber! - [aA][nN][gG][eE]\p{Punct}[bB][eE][rR]\p{Punct}
Kinds Kopf - [kK][iI][nN][dD][sS]\s*[kK][oO][pP][fF]
```

Erklärung des Verfahrens 3 zur automatischen Erzeugung regulärer Ausdrücke

Das Verfahren 3 kann verwendet werden, wenn man den Verdacht hat, dass die Zeichen '@', '€' und '|' mit einem Buchstaben assoziiert werden könnten. So liegt es nahe, dass das '@' Zeichen mit einem 'a' assoziiert werden könnte, das '€' Zeichen mit einem 'e' und der senkrechte Strich '|' mit einem 'i'. Aus diesem Grund wird das Zeichen '@' des normalen Wortes durch '[@aA]' ersetzt, das Zeichen '€' des normalen Wortes durch '[€eE]' und der senkrechte Strich des normalen Wortes durch den regulären Ausdruck '[iI|]'. Alle anderen Zeichen werden analog zu dem Verfahren 2 behandelt.

Beispiele:

```
Idiot - [iI][dD][iI][oO][tT]
arMleuchter112 - [aA][rR][mM][lL][eE][uU][cC][hH][tT][eE][rR]112
ange!ber! - [aA][nN][gG][eE]\p{Punct}[bB][eE][rR]\p{Punct}
Kinds Kopf - [kK][iI][nN][dD][sS]\s*[kK][oO][pP][fF]
€ber - [€eE][bB][eE][rR]
```

Erklärung des Verfahrens 4 zur automatischen Erzeugung regulärer Ausdrücke

Das Verfahren 3 behandelt schon einige wenige Zeichen, welche der Mensch mit anderen Buchstaben assoziieren könnte. Jedoch werden eben nur wenige Zeichen so interpretiert, dass diese auch Buchstaben sein könnten. Im Verfahren 4 wird zum einen davon ausgegeangen, dass gewisse Buchstaben durch andere Buchstaben oder sonstige Zeichen mit Ähnlichkeit des Buchstabens des normalen Wortes ersetzt werden können, so dass der Text für den Menschen noch ohne Probleme lesbar ist. Auch in umgekehrter Richtung wird nun davon ausgegangen, dass ein Buchstabe, welcher nicht aus dem 'deutschen' Alphabet stammt, oder andere 'exotische' Zeichen auch durch Ersetzung anderer, wiederum ähnlicher Buchstaben oder sonstiger Zeichen, ersetzt werden kann, so dass das Badword für den Menschen noch lesbar ist. Es folgt eine Tabelle von Zeichen, von denen angenommen werden kann, dass ein Mensch diese Buchstaben mit anderen Buchstaben oder Zeichen, welche in der 2. Spalte der Tabelle eingetragen sind, assoziiert. Desweiteren ist hinter einigen Buchstaben oder Zeichen die entsprechende Unicodenummer angegeben [13], damit die Zeichen im Programmcode besser wiederzuerkennen sind, da diese dort durch die Unicodenummer identifiziert werden.

Zeichen (Unicodenummer)	Könnte assizoiert werden mit den Zeichen
3	3, €, È, É, Ê, Ë, e, E
€	3, €, È, É, Ê, Ë, e, E
È (00C8)	3, €, È, É, Ê, Ë, e, E
É (00C9)	3, €, È, É, Ê, Ë, e, E
Ê (00CA)	3, €, È, É, Ê, Ë, e, E
Ë (00CB)	3, €, È, É, Ê, Ë, e, E
e	3, €, È, É, Ê, Ë, e, E
E	3, €, È, É, Ê, Ë, e, E
À (00C0)	À, Á, Â, Ã, Ä, Å, @, a, A
Á (00C1)	À, Á, Â, Ã, Ä, Å, @, a, A
Â (00C2)	À, Á, Â, Ã, Ä, Å, @, a, A
Ã (00C3)	À, Á, Â, Ã, Ä, Å, @, a, A
Ä (00C4)	À, Á, Â, Ã, Ä, Å, @, a, A
Å (00C5)	À, Á, Â, Ã, Ä, Å, @, a, A
@	À, Á, Â, Ã, Ä, Å, @, a, A
a	À, Á, Â, Ã, Ä, Å, @, a, A
A	À, Á, Â, Ã, Ä, Å, @, a, A
Ì (00CC)	Ì, Í, Î, Ï, |, i, I
Í (00CD)	Ì, Í, Î, Ï, |, i, I
Î (00CE)	Ì, Í, Î, Ï, |, i, I
Ï (00CF)	Ì, Í, Î, Ï, |, i, I
|	Ì, Í, Î, Ï, |, i, I
i	Ì, Í, Î, Ï, |, i, I
I	Ì, Í, Î, Ï, |, i, I
Ò (00D2)	Ò, Ó, Ô, Õ, Ö, 0, o, O
Ó (00D3)	Ò, Ó, Ô, Õ, Ö, 0, o, O
Ô (00D4)	Ò, Ó, Ô, Õ, Ö, 0, o, O
Õ (00D5)	Ò, Ó, Ô, Õ, Ö, 0, o, O
Ö (00D6)	Ò, Ó, Ô, Õ, Ö, 0, o, O
0	Ò, Ó, Ô, Õ, Ö, 0, o, O
O	Ò, Ó, Ô, Õ, Ö, 0, o, O
o	Ò, Ó, Ô, Õ, Ö, 0, o, O
¢ (00A2)	¢, ©, ç, €, c, C
© (00A9)	¢, ©, ç, €, c, C
ç (00E7)	¢, ©, ç, €, c, C
€	¢, ©, ç, €, c, C
c	¢, ©, ç, €, c, C
C	¢, ©, ç, €, c, C
® (00AE)	® , r, R
r	® , r, R
R	® , r, R

Tritt ein Zeichen der 1. Spalte der gezeigten Tabelle in dem normalen Wort auf, so wird dieses ersetzt durch den regulären Ausdruck '[<1. Zeichen der 2. Spalte><2. Zeichen der 2. Spalte>...<letztes Zeichen der 2. Spalte>]', wobei die spitzen Klammern entsprechend ersetzt werden müssen. Die Zeichen, die in der 1. Spalte mit einer Unicodenummer versehen wurden, werden in dem regulären Ausdruck durch eben diese Unicodenummer repräsentiert. Mit '\uxxxx', wobei 'xxxx' die entsprechende Unicodenummer ist. Alle anderen Zeichen werden entsprechend Verfahren 3 in einen regulären Ausdruck überführt.

Beispiele:

```
Idiot -
     [iI\|\u00CC\u00CD\u00CE\u00CF][dD][iI\|\u00CC\u00CD\u00CE\u00CF]
     [oO0\u00D2\u00D3\u00D4\u00D5\u00D6][tT]

arMleuchter112 -
     [aA@\u00C0\u00C1\u00C2\u00C3\u00C4\u00C5\u00AA][rR\u00AE][mM]
     [lL][eE3€\u00C8\u00C9\u00CA\u00CB][uU][cC€\u00A2\u00A9\u00E7]
     [hH][tT][eE3€\u00C8\u00C9\u00CA\u00CB][rR\u00AE]
     [3€\u00C8\u00C9\u00CA\u00CB][3€\u00C8\u00C9\u00CA\u00CB]
     [3€\u00C8\u00C9\u00CA\u00CB]

ange!ber! -
     [aA@\u00C0\u00C1\u00C2\u00C3\u00C4\u00C5\u00AA][nN][gG]
     [eE3€\u00C8\u00C9\u00CA\u00CB]\p{Punct}[bB]
     [eE3€\u00C8\u00C9\u00CA\u00CB][rR\u00AE]\p{Punct}

Kinds Kopf -
     [kK][iI\|\u00CC\u00CD\u00CE\u00CF][nN][dD][sS]\s*[kK]
     [oO0\u00D2\u00D3\u00D4\u00D5\u00D6][pP][fF]

€ber -
     [3€\u00C8\u00C9\u00CA\u00CBeE][bB][eE3€\u00C8\u00C9\u00CA\u00CB]
     [rR\u00AE]
```

Erklärung des Verfahrens 5 zur automatischen Erzeugung regulärer Ausdrücke

Das Verfahren 5 unterscheidet sich nur gering von dem Verfahren 4. Hier werden alle Zeichen identisch dem Verfahren 4 in einen regulären Ausdruck überführt, jedoch wird hier davon ausgegangen, dass beliebig viele Sonderzeichen zwischen den einzelnen Buchstaben, Zahlen oder sonstigen Zeichen auftreten können. Somit wird hinter jede Ersetzung des Verfahrens 4 noch der reguläre Ausdruck '\p{Punct}*' konkateniert. Auch wird angenommen, dass in einem zu filterndem Text ein Badword

auftaucht, in dem Buchstaben verdoppelt wurden. Um auch eine solche Abwandlung eines Wortes feststellen zu können, wird direkt hinter den eckigen Klammern des regulären Ausdrucks, der durch die Ersetzung eines Zeichens entstanden ist, ein '+' konkateniert.

```
Beispiele:

Idiot -
     [3€\u00C8\u00C9\u00CA\u00CBeE]\p{Punct}*[bB]+\p{Punct}*
     [eE3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*[rR\u00AE]+\p{Punct}*

arMleuchter112 -
     [aA@\u00C0\u00C1\u00C2\u00C3\u00C4\u00C5\u00AA]+\p{Punct}*
     [rR\u00AE]+\p{Punct}*[mM]+\p{Punct}*[lL]+\p{Punct}*
     [eE3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*[uU]+\p{Punct}*
     [cC€\u00A2\u00A9\u00E7]+\p{Punct}*[hH]+\p{Punct}*[tT]+\p{Punct}*
     [eE3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*[rR\u00AE]+\p{Punct}*
     [3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*
     [3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*
     [3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*

ange!ber! -
     [aA@\u00C0\u00C1\u00C2\u00C3\u00C4\u00C5\u00AA]+\p{Punct}*[nN]+
     \p{Punct}*[gG]+\p{Punct}*[eE3€\u00C8\u00C9\u00CA\u00CB]+
     \p{Punct}*\p{Punct}*[bB]+\p{Punct}*
     [eE3€\u00C8\u00C9\u00CA\u00CB]+\p{Punct}*
     [rR\u00AE]+\p{Punct}*\p{Punct}*

Kinds Kopf -
     [kK]+\p{Punct}*[iI\|\u00CC\u00CD\u00CE\u00CF]+\p{Punct}*[nN]+
     \p{Punct}*[dD]+\p{Punct}*[sS]+\p{Punct}*\s*[kK]+\p{Punct}*
     [oO0\u00D2\u00D3\u00D4\u00D5\u00D6]+\p{Punct}*[pP]+\p{Punct}*
     [fF]+\p{Punct}*
```

4.8 Interner Programmablauf und Programmcodeübersicht

In der Klasse 'Filter' ist die 'main' Methode implementiert. In dieser wird nur unterschieden, ob das Programm im Konsolenmodus aufgerufen werden soll oder ob nur ein übergebener Befehl ausgeführt werden soll. In beiden Fällen wird ein Objekt der Klasse 'ConsoleMode' erzeugt und die entsprechenden Methoden aufgerufen.
Die Klasse 'ConsoleMode' verwaltet die Befehlseingaben des Benutzers. Dazu stellt ein Objekt dieser Klasse die Methoden 'oneCommandFromConsole' und 'startSpam-

filterConsole' bereit. Wird die Methode 'oneCommandFromConsole' mit einer übergebenen Benutzereingabe aufgerufen, wird versucht, diesen Programmbefehl auszuführen und die entsprechende Ausgabe auf die Standardausgabe auszugeben. Kann der Benutzerbefehl aufgrund eines syntaktischen Fehlers des Programmbefehls nicht ausgeführt werden, wird eine Hilfemeldung auf die Standardausgabe ausgegeben. Die Methode 'getConsoleHelp' in der Klasse 'ConsoleMode' gibt einen String dieser Hilfemeldung zurück. In dieser Methode ist auch bei Bedarf die Hilfemeldung zu ändern. Wird die Methode 'startSpamfilterConsole' aufgerufen, wird solange die Standardeingabe zeilenweise eingelesen, bis der Benutzer den Programmbefehl 'quit' oder 'exit' eingibt. Jede Eingabe wird als Programmbefehl interpretiert und es wird versucht diesen auszuführen.
Die Klassen 'AddCommand', 'ChangeCommand', 'ShowCommand', 'DelCommand' und 'FilterCommand' implementieren alle das Interface 'Command'. Ein Klassendiagramm zur Übersicht über das Zusammenspiel dieser Klassen ist im Anhang F zu sehen. Das Interface stellt die Schnittstelle 'runCommand' zur Verfügung. Die Klassen repräsentieren die entsprechenden Programmbefehle. Die Klasse 'AddCommand' beispielsweise den ADD-Befehl usw.. Bei Aufruf der Methode 'runCommand', diese Schnittstelle ist natürlich in jeder Klasse entsprechend implementiert, wird der Befehl des entsprechenden Objektes mit den Eigenschaften des Objektes ausgeführt und gibt ein Objekt der Klasse 'OutputInfos' zurück. Zu der Klasse 'OutputInfos' später mehr. Ebenfalls sind in jeder der 'Befehlsklassen' statische Methoden implementiert, die aus einem übergebenem String das entsprechende Befehlsobjekt erzeugen und dieses zurückgeben. Ist in der Stringrepräsentation ein Fehler, so, dass das Objekt nicht erzeugt werden kann, wird 'NULL' zurückgegeben. Bei einem Rückgabewert 'NULL' liegt also ein syntaktischer Fehler bei der Befehlseingabe vor.
In dem oben genannten zurückgegebenen Objekt der Klasse 'OutputInfos' sind die Rückgabedaten, die ein Befehl erzeugt, repräsentiert. Die Klasse 'OutputInfos' ist als UML-Klassendiagramm im Anhang F zu sehen. Eine solche Repräsentation der Daten hat den Vorteil, dass diese von der Darstellung, wie sie dem Benutzer gezeigt werden sollen, getrennt sind. Für die Ausgabe auf der Standardausgabe sind die entsprechenden Methoden schon vorhanden. Diese sind ebenfalls in der Klasse 'OutputInfos'. Will man beispielsweise auf die Standardausgabe nach einem ausgeführten ADD-Befehl eine entsprechende Rückgabe aus den in dem erhaltenen Objekt der Klasse 'OutputInfos' an den Benutzer geben, kann man die Methode 'printConsoleOutputForAddOrder' aufrufen. Für andere ausgeführte Programmbefehle entsprechend 'printConsoleOutputForFilterOrder', 'printConsoleOutputForChangeOrder' u.s.w..
Wird ein Programmbefehl des Typs 'Filter' aufgerufen, werden die Rückgabeinformationen in einem Objekt der Klasse 'TextEvaluation' repräsentiert. Dieses Objekt mit den Informationen ist eine Eigenschaft des Rückgabeobjektes einer Befehlsausführung der Klasse 'OutputInfos'. Die Klasse 'OutputInfos' aggregiert demnach die Klasse 'TextEvaluation'.
Die Klasse 'Addressor', siehe für ein Klassendiagramm dieser Klasse Anhang F, repräsentiert einen Absender mit seiner ID, seinem Namen, ob dieser gefiltert werden

soll und ob dieser Absender Nachrichten auf die Bildschirme senden darf oder nicht. Die Klasse 'Badword', siehe für ein Klassendiagramm dieser Klasse Anhang F, repräsentiert ein Badword mit seiner ID, das normale Badword und das Badword in Form eines regulären Ausdrucks.
Die Klassen 'Log' und 'Config' wurden mittels eines Singletonpattern implementiert. Mittels der Klasse 'Log' kann ein String in eine Logdatei oder/und auf die Standardausgabe geschrieben werden. Dies hängt von den Einstellungen in der entsprechenden Konfigurationsdatei ab. Siehe hierzu Kapitel 4.5. Die Logdatei wird beim ersten Schreiben in diese erzeugt. Ihr Name besteht aus

`'Tag'_'Monat'_'Jahr'_'Stunden'_'Minuten'_'Sekunden'.log`

Der in "' gefasste Text ist durch die Angaben beziehend auf ihr Erzeugungsdatum passend zu ersetzen. Eine Logdatei, welche beispielsweise am 22.04.2001 um 11:26:33 Uhr erzeugt wurde, hätte den Namen

`22_04_2001_11_26_33.log`

Die entsprechenden erzeugten Logdateien befinden sich im Ordner 'data/log' relativ vom Programm ausgehend.
Die Klasse 'Config' liest beim ersten Zugriff auf eine der Konfigurationseigenschaften die Daten aus der entsprechenden Konfigurationsdatei. Ist keine Konfigurationsdatei vorhanden, werden die Eigenschaften übernommen, wie sie in der Klasse 'Config' implementiert sind. Die Klasse Config besitzt eine Reihe von privaten Klassenattributen des Typs 'String'. Diese geben an, wie die einzelnen Optionen in der Konfigurationsdatei heißen. Möchte der Benutzer den Namen einer Option in der Konfigurationsdatei ändern, muss nur der Wert des entsprechenden Klassenattributs geändert werden. Eine Schnittstelle zur Änderung der Konfiguration im laufenden Betrieb des Programms bietet die Klasse 'Config' nicht.
Die Klassen 'AddressorFilter', 'BadwordFilter', 'NaiveBaiseFilter' und 'RulesFilter' implementieren alle das Interface 'SpamFilter', welches die Schnittstelle 'runFilter' bereitstellt. Siehe für ein Klassendiagramm, welches das Zusammenspiel dieser Klassen beschreibt, Anhang F. Diese Schnittstelle ist in den Klassen entsprechend implementiert. Übergibt man der Methode 'runFilter' der entsprechenden Klassen, in der diese implementiert ist, den zu klassifizierenden Text und ein Objekt der Klasse 'OutputInfos', so wird der Text mit der entsprechenden Klassifizierungsmethode klassifiziert. Die Auswertung der Klassifizierung enthält das übergebene Objekt der Klasse 'OutputInfos'. Wird in dieser Art und Weise die 'runFilter' Methode der Klasse 'AddressorFilter' ausgeführt, wird der Text aufgrund der Absenderinformationen klassifiziert. Wird die 'runFilter' Methode der Klasse 'RulesFilter' ausgeführt, so wird der Text aufgrund des regelbasierten Klassifizieres klassifiziert, u.s.w.. Welcher Klassifizierer genutzt wird, ist am Klassennamen zu erkennen.
Die Klasse 'DBConnection' beschreibt die Schnittstelle zur Datenbank. Ein Objekt dieser Klasse besitzt die Methoden 'runQueryUpdate' und 'runQuerySelect'. Diesen Methoden wird ein SQL-Befehl übergeben, der dann auf der in der Konfigurationsdatei beschriebenen Datenbank ausgeführt wird. 'runQueryUpdate' stellt

die Schnittstelle für eine 'UPDATE', 'INSERT' oder 'DELETE' SQL-Anweisung bereit. Im Gegensatz zu der Methode 'runQuerySelect', die die Schnittstelle für eine 'SELECT' SQL-Anweisung bereitstellt. Beide Methoden geben ein Objekt der Klasse 'MyDBResultSet' zurück. Mittels diesen Objektes können die Werte, die die Datenbank zurückgegeben hat, ausgelesen werden. Danach kann die verwendete Datenbankverbindung zurück an den 'Connectionpool' gegeben werden. Im ganzen Programm wird ein Objekt der Klasse 'DBConnection' nur einmal erzeugt und dann an die entsprechenden Objekte übergeben.
Die Klasse 'StringUtils' stellt eine Reihe von Methoden bereit, die zur Stringverarbeitung verwendet werden. Hierzu gehört unter anderem die Methode 'makeRegularExpression', die aus einem übergebenen String und einer natürlichen Zahl, die angibt wie der reguläre Ausdruck erzeugt werden soll, einen regulären Ausdruck zurück gibt. Diese Methode nutzt zur Erstellung der regulären Ausdrücke die Methoden aus der Klasse 'RegExMaker'. Siehe hierzu das entsprechende Klassendiagramm in Anhang F. Die in der Klasse implementierten Methoden wandeln den übergebenen String in einen regulären Ausdruck um und geben diesen als String zurück. Die Methode 'makeRegularExpression' aus der Klasse 'StringUtils' wählt lediglich die richtige Methode aus. Ebenfalls stellt die Klasse 'StringUtils' Methoden zur Verfügung, die dem regelbasierten Klassifizierer helfen, die Texte zu analysieren.
Die Klasse 'ProgramInternalConfigurations' stellt eine Schnittstelle zu der Datei '.intern' zur Verfügung, in der Daten gespeichert sind, die auch noch nach einem Neustart des Programms zur Verfügung stehen müssen. Mit den Methoden 'getLastHamCount' und 'getLastSpamCount' kann auf die Anzahl der in der Datenbank vorhandenen Zeilen der Tabelle 'ham' bzw. 'spam' zugegriffen werden, die mittels den Methoden 'setLastHamCount' bzw. 'setLastSpamCount' in diese Datei gespeichert wurden. Diese Information muss beispielsweise so gespeichert werden, dass auch nach einem Programmneustart noch darauf zugegriffen werden kann, da bei einer veränderten 'spam' oder 'ham' Tabelle der bayessche Klassifizierer erneut viele Daten aus diesen Informationen vorberechnen muss.
In dem Interface 'TableConstans' befinden sich Konstanten, die die Datenbanktabellennamen speichern. Wird der Name einer Tabelle in der Datenbank geändert, muss lediglich die entsprechende Konstante geändert werden.
Im Interface 'OrderConstants' können die Programmbefehle angepasst werden. Soll beispielsweise der Befehl 'SHOW' zukünftig 'ZEIGE' heißen, muss nur die entsprechende Konstante angepasst werden. Siehe hierzu Anhang F.

5 Güte des programmierten Klassifizierers

Nun wird das oben beschriebene Programm mittels der alten, als auch mittels neuer Spam- und Hamnachrichten evaluiert. Die neuen Nachrichten wurden aufgrund einer Bitte an eine Gruppe von Studenten, welche vorher mit der Eingangs beschriebenen Situation vertraut gemacht wurden, zugesandt. Die neuen Ham- und Spamnachrichten sind von 9 Personen dieser Gruppe. Insgesamt konnten so 91 Nachrichten gewonnen werden, bei denen man annehmen kann, dass diese ein realistischeres 'Bild' über die auf den Ticker zu erwartenden Nachrichten geben. Die neuen Nachrichten können im Anhang F eingesehen werden. Es wurden in der Konfigurationsdatei folgende Einstellungen vorgenommen:

- naive_baise=1
- rule=1
- min_text_length=10
- different_chars=8
- whitespace_chars=33.0
- special_chars=33.0
- norm_spec_norm=4
- bad_words=1
- naive_baise_min_word_length=4
- addressors=0

Von den 91 Tickernachrichten wurden 'von Hand' 41 Nachrichten als Spam klassifiziert und 50 als Ham, also besteht dieser Datensatz aus ca. 45% Hamnachrichten und 55% Spamnachrichten. Zuerst wurden ausschließlich aus diesen Nachrichten 3 verschiedene Test- und Trainingsmengen gebildet. Die Testmengen bestehen je aus 7 zufällig ausgewählten Spam- und 7 zufällig ausgewählten Hamnachrichten. Diese 3 Testläufe ergaben folgende Ergebnisse:

Testlauf 1:

Recall	Precision	richtig erkannt	falsch erkannt
0,71	0,83	0,79	0,21

Testlauf 2:

Recall	Precision	richtig erkannt	falsch erkannt
0,85	0,67	0,71	0,29

Testlauf 3:

Recall	Precision	richtig erkannt	falsch erkannt
1,00	0,88	0,93	0,07

Der Durchschnitt dieser 3 Testreihen ist:

Recall	Precision	richtig erkannt	falsch erkannt
0,85	0,79	0,81	0,19

Während der Validierung der Daten konnte man beobachten, dass die meisten Nachrichten aufgrund der Badwords und aufgrund des bayesschen Klassifikators richtig klassifiziert wurde. Der regelbasierte Filter hat nur in wenigen Fällen eine Spamnachricht auch als Spam erkannt. So wurden von allen 42 Nachrichten aus den 3 erstellten Testmengen nur 3 Spamnachrichten durch den regelbasierten Filter erkannt und 1 Hamnachricht aufgrund des regelbasierten Filters fälschlicherweise als Spam kategorisiert.
In einer zweiten Testsession wurden alle von mir erstellten Spam- und Hamtexte als Trainingsmenge verwendet und die neu erhaltenen Texte als Testmenge benutzt. Die Testmenge bestand folglich aus 45% Hamnachrichten und 55% Spamnachrichten. Dies führte zu folgendem Ergebnis:

Recall	Precision	richtig erkannt	falsch erkannt
0,67	0,85	0,75	0,25

In einer letzten Testreihe wurden sowohl die neu erhaltenen Spam- und Hamnachrichten, als auch die schon erstellten Nachrichten durchmischt und aus diesen ein Testset von 70 Nachrichten generiert. Die verbliebenen Nachrichten wurden als Trainingsmenge benutzt. In der Testmenge befanden sich 35 Spam- und 21 Hamnachrichten. Die Evaluation mit der hier beschriebenen Testmenge ergab folgendes Ergebnis:

Recall	Precision	richtig erkannt	falsch erkannt
0,68	0,68	0,61	0,39

Wie man sehen kann sind diese Evaluierungsergebnisse nicht so gut wie jene in den ersten drei Testreihen, die nur aufgrund der neu erhaltenen Nachrichten erstellt worden sind. Der Grund für diese Tatsache wurde nicht weiter untersucht.
Nun wurden wieder alle oben vorgestellten Klassifizierer (NB, KNN, SVM, DTJ) mit den neu erhaltenen Texten und den schon vorher erstellten Texten mittels *weka* evaluiert. Es wurden keine Ersetzungen oder sonstige Änderungen an den Worten, wie in manchen Testreihen oben geschehen, vorgenommen. Die eingesetzte Validierungsmethode war 'cross validation'. Dies führte zu folgenden Ergebnissen:
Naive Baise:

Recall	Precision	richtig erkannt	falsch erkannt
0,84	0,69	0,67	0,33

k-nearest neighbor Algorithmus:

Recall	Precision	richtig erkannt	falsch erkannt
0,99	0,63	0,62	0,38

Supported Vector machine:

Recall	Precision	richtig erkannt	falsch erkannt
0,95	0,68	0,7	0,3

Entscheidungsbaum (J48):

Recall	Precision	richtig erkannt	falsch erkannt
1	0,62	0,62	0,38

Die Ergebnisse, besonders die, die nur mittels der neuen Nachrichten gewonnen wurden sind meiner Meinung nach durchaus brauchbar. Wenn man davon ausgeht, dass diese Nachrichten repräsentativer für die beschriebene Situation sind, sind durchaus hohe Erkennungsraten von Spamtexten zu erwarten. Jedoch darf dieses Ergebnis nicht über die Tatsache hinweg täuschen, dass es sich auch hierbei um Nachrichten handelt, die auf Annahmen beruhen.

6 Ausblick

Trotz der Analyse des gestellten Szenarios bleiben einige Fragen offen. Dazu gehören besonders die Fragen 'Wie werden Spam- oder auch Hamnachrichten in Zukunft wirklich aussehen?' oder 'Gibt es Arten von Spam, die bei der gemachten Analyse nicht berücksichtigt worden sind?'. Diese Fragen lassen sich natürlich erst klären, wenn jeder Student Nachrichten auf die Bildschirme senden kann und diese zur Analyse zur Verfügung stehen.
Das Programm kann durch das Design flexibel an bestimmte Situationen angepasst werden, dennoch, selbst mit aussreichenden Datensätzen und noch so sorgfältiger Analyse der Situation wird die menschliche Kreativität jeden Filter auf kurze oder lange Sicht überlisten können, so dass eine 100 % ige Erkennung von Spamnachrichten mit gleichzeitiger 100 % iger Erkennung von Hamnachrichten nie möglich sein wird. Auch die Frage danach, welche Nachrichten nun Spam sind, ist mit jedem neuem Programmadministrator neu zu stellen.
So bleibt abzuwarten, welche Filtermethoden sich für diese Situation bewähren werden. Eine Erweiterung und dauernde Anpassung der Konfiguration des programmierten Filters bleibt unerlässlich. Natürlich ist auch eine permanente Weiterentwicklung des Programms durchaus sinnvoll. So könnte man weitere Klassifizierer mittels *Weka* evaluieren und diese dann auch sofort mittels *Weka* in das Programm integrieren. Die direkte Integration spart zwar Entwicklungszeit, möchte man aber flexlibel den Code anpassen können, halte ich eine eigene Implementation der Verfahren für unerlässlich.

Anhang A - Programmbefehle

Allgemeines

Um die Syntax der Befehle zu beschreiben, wurde sich stark an das Schema angelehnt, welches in den 'MySQL X Reference Manual' verwendet wird, wobei 'X' für eine beliebige Version steht. Bei Eingabe der Befehle braucht auf Groß-/Kleinschreibung nicht geachtet zu werden. Davon ausgeschlossen sind natürlich die übergebenen Texte.
Wörter in Großbuchstaben sind Schlüsselwörter. Auch Sonderzeichen, insbesondere der Punkt '.', das Gleichheitszeichen '=', die sich öffnende runde Klammer '(' und die sich schließende runde Klammer ')' sind Schlüsselwörter bzw. hier Schlüsselzeichen. Die Zeichen '[', ']', '{', '}' , '|' wie auch das größer '>' und kleiner '<' Zeichen sind keine Schlüsselwörter. Befehle in eckigen Klammern sind optional. Durch einen senkrechten Strich getrennte Elemente sind Alternativen. Der senkrechte Strich kann nur innerhalb von eckigen oder geschweiften Klammern zum Einsatz kommen. Wenn aus einer Menge von Alternativen ein Element ausgewählt werden muss, müssen die geschweiften Klammern genutzt werden. Befehlsteile, welche von den größer und kleiner Zeichen umschlossen sind, können in ihrer Reihenfolge vertauscht werden. Die Elemente des Befehls, welche vertauscht werden können, sind mit geschweiften Klammern umgeben. Auch kann die Reihenfolge dieser Elemente nur innerhalb der größer und kleiner Zeichen vertauscht werden. Kleingeschriebene Wörter sind Platzhalter für etwas Bestimmtes. Das kleingeschriebene Wort beschreibt, was an Stelle dieses stehen kann. Zur Verdeutlichung der Elemente der Befehle, hier eine kurze Erklärung:

Das Element 'natürliche_zahl' kann durch jede beliebige Zahl >= 0 ersetzt werden. Die Zahl '7663' wäre eine gültige natürliche Zahl.
Die Elemente 'id', 'lowid' und 'highid' können ebenfalls durch eine natürliche Zahl ersetzt werden.
Ein 'natürlichsprachiger_text' ist ein Text bestehend aus allen sichtbaren Zeichen. 'Hallo Welt :-\)' wäre ein gültiger 'natürlichsprachiger_text', wobei darauf zu achten ist, dass bei einigen Textübergaben die runden Klammern innerhalb eines solchen Textes ein Backslash vorangestellt haben müssen. Siehe hierzu die Erläuterung der Befehle in Kapitel 4.4. Dieser wird bei der Auswertung des Befehls nicht beachtet.
Ein 'gültiger_regulärer_ausdruck' ist jeder reguläre Ausdruck, der der Konvention der Java API [11] entspricht. Ein gültiger regulärer Ausdruck wäre

'[aBc*].\\p{Punct}'.

SHOW - Befehl

Syntax

```
SHOW
{SPAM|HAM|UNCERTAIN|ADDRESS}
< {[MAXLENGTH=natürliche_zahl]} {id|lowid..highid} >
```

ODER

```
SHOW BADWORD [{id|lowid..highid}]
```

Der SHOW-Befehl zeigt die Datenbankeinträge einer bestimmten Tabelle an. Wird nur das Schlüsselwort *SHOW* mit Angabe eines der Schlüsselwörter *SPAM*, *HAM*, *UNCERTAIN*, *ADDRESS* oder *BADWORD* verwendet, so werden alle Einträge der entsprechenden Datenbanktabelle in überarbeiteter Form zurückgegeben. Wird desweiteren eine 'id' mit angegeben, so wird nur der Datenbankeintrag dieser übergebenen id zurückgegeben, sofern einer vorhanden ist. Wird eine durch '..' getrennte high- und lowid mit übergeben, werden alle vorhandenen Datenbankeinträge, die >= der 'lowid' und <= der 'highid' sind ebenfalls in überarbeitender Form zurückgegeben. Wird desweiteren noch durch das Schlüsselwort *MAXLENGTH* eine Maximallänge der Datenbankinhalte übergeben, dies ist nicht mit dem Schlüsselwort *BADWORD* möglich, so werden nur diejenigen Datenbankeinträge zurückgegeben, bei denen die Zeichenlänge des 'Hauptinhaltes' <= der durch das Schlüsselwort *MAXLENGTH* übergebenen Maximallänge ist.
Durch das Schlüsselwort *SPAM* werden die als Spam kategorisierten Texte angesprochen. Der Hauptinhalt, auf den sich die Angabe der Maximallänge bezieht, ist der als Spam kategorisierte Text selbst.
Durch das Schlüsselwort *HAM* werden die als Ham kategorisierten Texte angesprochen. Die Angabe der Maximallänge der Zeichen bezieht sich auf den als Ham kategorisierten Text selbst.
Durch das Schlüsselwort *UNCERTAIN* werden die noch nicht einsortierten Texte angesprochen. Die Angabe der Maximallänge der Zeichen bezieht sich auf den Text dieser Texte.
Mit dem Schlüsselwort *ADDRESS* werden die hinterlegten Absender angesprochen. Die Angabe der Maximallänge bezieht sich auf die Bezeichnung des Absenders.
Das Schlüsselwort *BADWORD* bezieht sich auf die hinterlegten Badwords.

DEL - Befehl

Syntax

```
DEL
{SPAM|HAM|UNCERTAIN|BADWORD|ADDRESS}
[{id|lowid..highid}]
```

Mittels des DEL-Befehls können Spamnachrichten, Hamnachrichten, noch nicht kategorisierte Nachrichten, Badwords oder Absender gelöscht werden. Eine Löschung durch Zuhilfenahme des DEL-Befehls ist nicht wieder rückgängig zu machen. Vor Ausführung des Del-Befehls wird der Benutzer gefragt, ob er die Löschung wirklich vornehmen will. Antwortet dieser mit 'ja', so wird der Befehl ausgeführt. Antwortet der Benutzer hingegen mit 'nein', wird dieser Befehl nicht ausgeführt.
Welche Angaben gelöscht werden sollen, wird durch die Schlüsselwörter *SPAM*, *HAM*, *UNCERTAIN*, *BADWORD* oder *ADDRESS* angegeben. Der Bezug dieser Schlüsselwörter ist in Kapitel 6 beschrieben.

DELETE - Befehl

Syntax

```
DELETE
{SPAM|HAM|UNCERTAIN|BADWORD|ADDRESS}
[{id|lowid..highid}]
```

Der DELETE-Befehl macht genau das Gleiche wie der DEL-Befehl, fragt jedoch den Benutzer nicht, ob dieser Befehl ausgeführt werden soll.

ADD - Befehl

Syntax

```
ADD
{SPAM|HAM|UNCERTAIN}
(natürlichsprachlicher_text)
```

ODER

```
ADD BADWORD
< {[REG=(gültiger_regulärer_ausdruck)]} {NORM=(natürlichsprachlicher_text)} >
```

ODER

```
ADD ADDRESS
< {[STOP=booleanwert]} {[FILTER=booleanwert]} {ADDR=(natürlichsprachlicher_text)} >
```

Mit dem ADD-Befehl können Daten hinzugefügt werden. Welche Daten hinzugefügt werden sollen, wird durch die Schlüsselwörter *SPAM*, *HAM*, *UNCERTAIN*, *BADWORD* oder *ADDRESS* bestimmt.

CHANGE - Befehl

Syntax

```
CHANGE {SPAM|HAM|UNCERTAIN} {id|lowid..highid} {SPAM|HAM|UNCERTAIN}
```

Der Changebefehl dient zum Verschieben von Datensätzen.

FILTER - Befehl

Syntax

```
FILTER
< {[[-]NBO]} {[[-]RUL]} {[-ADDRCHECK]} {[[-]BADW]}
  {[ADDRESSOR=(natürlichsprachlicher_text)]}
  {TEXT=(natürlichsprachlicher_text)} >
```

Mittels der oben beschriebenen Syntax des Filterbefehls kann ein Text klassifiziert werden. Diese Syntax gilt nur, wenn die Absenderüberprüfung ausgeschaltet ist. Wird diese bei der Klassifizierung mit genutzt, kann folgende Syntax des Filterbefehls genutzt werden.

```
FILTER
< {[[-]NBO]} {[[-]RUL]} {[ADDRCHECK]} {[[-]BADW]}
  {ADDRESSOR=(natürlichsprachlicher_text)}
  {TEXT=(natürlichsprachlicher_text)} >
```

Programm beenden

Syntax

```
{QUIT|EXIT}
```

Mit diesem Befehl kann man das Programm aus dem Konsolenmodus beenden.

Anhang B - Eine Kopie der Standardkonfigurationsdatei

```
%% --------------------------------Allgemeines-----------------------
%% Diese Datei ist ein Beispielkonfigurationsdatei mit den gesetzten
%% Defaultwerten, die auch ohne diese angenommen wuerden. Jede Option
%% ist als Kommentar beschrieben. Kommentare beginnen mit '%%'
%% und sind nur fuer diese Zeile gueltig. Es ist nicht moeglich eine
%% Option zu setzen und dahinter einen Kommentar anzubringen oder
%% umgekehrt. Auf Gross- und Kleinschreibung ist zu achten.

%%---------------------------------------------------------------------
%% Wird 'file_log' auf '1' oder 'on' gesetzt, wird ein Logfile
%% erstellt. Das Logfile befindet sich in dem Verzeichnis vom dem
%% Programmcode ausgehend './data/log/.'. Nach jedem erneuten
%% Programmstart wird ein neues Logfile erstellt, welches als Dateiname
%% einen aktuellen Zeitstempel des Programmstarts hat. Soll hingegen
%% kein Logfile erstellt und geschrieben werden, ist die Option
%% 'file_log' auf '0' oder 'off' zu setzen. Alternativ zu 'file_log'
%% kann 'filelog' angegeben werden.
%% Beispiele:
%% file_log=1
%% filelog=on
%% file_log=off

file_log=0

%%---------------------------------------------------------------------
%% Neben des Schreibens des Logs in eine Datei, koennen die Logeintraege
%% auch zusaetzlich, sofern die Option 'file_log' auf 'on' oder '1' ist,
%% oder exklusiv, die Option 'file_log' ist nicht auf 'on' oder '1',
%% auf die Konsole ausgegeben werden. Soll dies geschehen, ist die
%% Option 'std_out_log' mit einer '1' oder 'on' zu setzen. Soll das Log
%% nicht auf die Konsole ausgegeben werden, ist die Option
%% 'std_out_log' mit einer '0' oder 'off' zu setzen. Alternativ zu
%% 'std_out_log' kann auch 'stdout_log', 'std_outlog' oder 'stdoutlog'
```

```
%% angegeben werden.
%% Beispiele:
%% std_out_log=0
%% stdout_log=on
%% std_outlog=1
%% stdoutlog=off

std_out_log=0

%%-----------------------------------------------------------------------
%% Mit der Option 'naive_baise' kann angegeben werden, ob eine
%% Textklassifizierung mit dem Naive-Baise Verfahren standardmaessig
%% genutzt werden soll. Wie man von dieser Einstellung bei der
%% Klassifizierung von Texten abweicht, kann in der zu dem Programm
%% zugehoerigen Dokumentation nachgelesen werden. Ist die Option
%% 'naive_baise' mit einer '1' oder 'on' gesetzt, wird dieser Filter
%% standardmaessig mit genutzt. Ist diese Option mit einer '0' oder 'off'
%% gesetzt, wird dieser Filter standardmaessig nicht mitgenutzt.
%% Alternativ zu 'naive_baise' kann auch die Option 'naivebaise'
%% angegeben werden.
%% Beispiele:
%% naive_baise=0
%% naivebaise=on
%% naive_baise=1

naive_baise=1

%%-----------------------------------------------------------------------
%% Mit der Option 'rule' kann angegeben werden, ob eine Textklassi-
%% fizierung mittels des regelbasierten Filters geschehen soll. Wie
%% man von dieser Einstellung bei der Klassifizierung von Texten
%% abweicht, kann in der zu dem Programm zugehoerigen Dokumentation
%% nachgelesen werden. Ebenso kann dort nachgelesen werden, welche
%% Regeln zum Klassifizieren von Texten herangezogen werden. Diese
%% koennen auch innerhalb der Konfigurationsdatei ein oder ausge-
%% schaltet werden. Diese Option gibt an, ob der regelbasierte
%% Filter grundsaetzlich mit benutzt werden soll. Ist die Option
%% 'rule' mit einer '0' oder 'off' gesetzt, so wird diese Klassi-
%% fikationsmethode nicht mit genutzt. Wird die Option hingegen mit
%% einer 1 oder auf 'on' gesetzt, wird diese Klassifikationsmethode
%% standardmaessig mit genutzt.
%% Beispiele:
%% rule=0
%% rule=1
```

```
%% rule=off
%% rule=on

rule=on

%%-----------------------------------------------------------------------
%% Mit der Option 'min_text_length' kann angegeben werden, welche
%% Zeichenanzahl der zu filternde Text mindestens besitzen muss. Be-
%% steht der zu filternde Text aus weniger Zeichen, wird dieser als
%% Spam klassifiziert. Die Option ist mit einer natuerlichen Zahl
%% groesser gleich 0 zu besetzen. Wird diese Option mit der Zahl 0 ge-
%% setzt, ist dieser Filter ausgeschaltet. Alternativ zur Option
%% 'min_text_length' koennen die Optionen 'mintext_length',
%% 'min_textlength' oder 'mintextlength' angegeben werden.
%% Beispiele:
%% min_text_length=0
%% mintext_length=945367
%% min_textlength=12
%% mintextlength=100

min_text_length=1

%%-----------------------------------------------------------------------
%% Mit der Option 'different_chars' kann angegeben werden, wieviele
%% verschiedene Zeichen aus dem Alphabet sowohl klein als auch gross A-Z
%% in dem zu klassifizierenden Text mindestens enthalten sein muessen.
%% Der Text 'halLo Welt!' besitzt demnach 6 verschiedene Zeichen. Alter-
%% nativ zur Option 'different_chars' kann die Option 'differentchars'
%% angegeben werden. Diese Option erwartet eine natuerliche Zahl
%% groesser gleich 0 und kleiner gleich 26. Wird diese Option mit 0
%% gesetzt, wird der Text nach dieser Regel nicht klassifiziert.
%% Beispiele:
%% different_chars=0
%% differentchars=4
%% different_chars=26
%% differentchars=16

different_chars=2

%%-----------------------------------------------------------------------
%% Mit der Option 'whitespace_chars' kann in Prozent angegeben werden,
%% wieviel Prozent der Zeichen des zu klassifizierenden Textes
%% hoechstens Whitespacecharacters sein duerfen. Besitzt der Text im
%% Verhaeltnis zu der Gesamtzeichenanzahl mehr als in dieser Option
```

```
%% angegebene Prozentzahl an Whitespacecharacter, wird der
%% Text als Spam klassifiziert. Diese Option erwartet eine Zahl mit
%% hoestens 3 Nachkommastellen, welche groesser gleich 0.0 und kleiner
%% gleich 100.0 ist. Das Komma muss als Punkt notiert werden. Alterna-
%% tiv zur Option 'whitespace_chars' kann auch die Option
%% 'whitespacechars' gesetzt werden. Ist diese Option mit 100.0
%% gesetzt, wird der Text nach dieser Regel nicht klassifiziert.
%% Welche Zeichen zu den Whitespacecharacter gehoeren, entnehmen Sie
%% bitte der schriftlichen Dokumentation.
%% Beispiele:
%% whitespace_chars=0
%% whitespacechars=2.01
%% whitespace_chars=100.0
%% whitespacechars=47.597

whitespace_chars=99.0

%%------------------------------------------------------------------
%% Mit der Option 'special_chars' kann in Prozent angegeben werden,
%% wieviel Prozent der Zeichen des zu klassifizierenden Textes
%% hoechstens Sonderzeichen sein duerfen. Besitzt der Text im
%% Verhaeltnis zu der Gesamtzeichenanzahl mehr als in dieser Option
%% angegebene Prozentzahl an Sonderzeichen, so wird der
%% Text als Spam klassifiziert. Diese Option erwartet eine Zahl mit
%% hoestens 3 Nachkommastellen, welche groesser gleich 0.0 und kleiner
%% gleich 100.0 ist. Das Komma muss als Punkt notiert werden. Alterna-
%% tiv zur Option 'special_chars' kann auch die Option
%% 'specialchars' gesetzt werden. Ist diese Option mit 100.0
%% gesetzt, so wird der Text nach dieser Regel nicht klassifiziert.
%% Welche Zeichen zu den Sonderzeichen gehoeren, entnehmen Sie bitte
%% der schriftlichen Dokumentation.
%% Beispiele:
%% special_chars=0
%% specialchars=7.76
%% special_chars=100.0
%% specialchars=33.332

special_chars=99.0

%%------------------------------------------------------------------
%% Mit der Option 'norm_spec_norm' kann angegeben werden, wie oft
%% Sonderzeichen hoechstens direkt zwischen Buchstaben oder Zahlen
%% stehen duerfen. Hierzu ein Beispiel: 'Hal!o Welt! Du bist gross.'
%% Hier wuerde nur das Sonderzeichen Ausrufezeichen einmal direkt
```

```
%% zwischen zwei Buchstaben stehen, naemlich bei 'Hal!o'.
%% Das Ausrufezeichen von 'Welt! ' und der Punkt von 'gross.' werden
%% nicht gezaehlt, da das Ausrufezeichen von 'Welt! ' nicht direkt von
%% einem Buchstaben oder einer Zahl gefolgt wird. Bei dem Punkt von
%% 'gross.' ist der Text zu Ende und wird somit auch nicht direkt von
%% einer Zahl oder einem Buchstaben gefolgt. Diese Option erwartet
%% eine natuerliche Zahl groesser 0, die angibt wie oft es hoechstens
%% auftreten darf, dass ein Sonderzeichen von Buchstaben oder Zahlen
%% umschlossen wird. Welche Zeichen zu den Sonderzeichen gehoeren,
%% entnehmen sie bitte der schriftlichen Dokumentation. Alternativ zu
%% der Option 'norm_spec_norm' koennen die Optionen 'normspec_norm',
%% 'norm_specnorm' oder 'normspecnorm' angegeben werden. Ist die Zahl
%% von norm_spec_norm groesser, als der zu klassifizierende Text,
%% wird der Text nach dieser Regel nicht klassifiziert.
%% Beispiele:
%% norm_spec_norm=0
%% norm_spec_norm=2
%% norm_spec_norm=44
%% norm_spec_norm=932648

norm_spec_norm=250

%%----------------------------------------------------------------------
%% Mit der Option 'bad_words' kann angegeben werden, ob eine Suche
%% nach bestimmten Woertern zur Textklassifikation standardmaessig mit
%% genutzt werden soll. Wie man von dieser Einstellung bei der
%% Klassifizierung von Texten abweicht, kann in der zu dem Programm
%% zugehoerigen Dokumentation nachgelesen werden. Ist die Option
%% 'bad_words' mit einer '1' oder 'on' gesetzt, wird diese
%% Klassifikationsmethode standardmaessig mit genutzt. Ist diese Option
%% mit einer '0' oder 'off' gestzt, wird diese
%% Klassifikationsmethode standardmaessig nicht mitgenutzt. Alternativ
%% zu 'bad_words' kann auch die Option 'badwords' angegeben werden.
%% Beispiele:
%% bad_words=0
%% badwords=on
%% bad_words=off

bad_words=1

%%----------------------------------------------------------------------
%% Bei dem Eintrag eines neuen Badwords kann auf die Angabe eines
%% dazugehoerigen regulaeren Ausdrucks verzichtet werden. Mit der
%% Option 'reg_ex_strategy' kann angegeben werden, welche Strategie
```

```
%% zur automatischen Generierung eines regulaeren Ausdrucks angewandt
%% werden soll. Naeheres zu den einzelnen Strategien finden Sie in der
%% zu dem Programm zugehoerigen Dokumentation. Die Angabe der Strategie
%% erfolgt durch Angabe einer positiven natuerlichen Zahl inklusive 0,
%% wobei bei einer Angabe einer Zahl, die keiner Strategie entspricht,
%% die zugehoerige Strategie der Zahl 0 gewaehlt wird. Eine
%% genauere Erklaerung befindet sich, wie schon erwaehnt, in der
%% Programmdokumentation. Insgesamt gibt es 6 Strategien, die Strategie
%% 0,1,2,3,4 oder 5.
%% Alternativ zur Option 'reg_ex_strategy' koennen auch einer der
%% Optionen 'regex_strategy', 'reg_exstrategy' oder 'regexstrategy'
%% angegeben werden.
%% Beispiele
%% reg_ex_strategy=0
%% regex_strategy=3
%% reg_exstrategy=5
%% regexstrategy=4

reg_ex_strategy=0

%%--------------------------------------------------------------------
%% Mit der Option 'naive_baise_min_word_length' kann die Mindestlaenge
%% der Woerter angegeben werden, die bei diesem
%% Klassifikationsverfahren beruecksichtigt werden sollen. Naeheres zu
%% der Verwendung dieser Option finden Sie in der zu dem Programm
%% zugehoerigen Dokumentation. Alternativ zur Option
%% 'naive_baise_min_word_length' kann auch
%% 'naivebaise_min_word_length', 'naive_baisemin_word_length',
%% 'naive_baise_minword_length', 'naive_baise_min_wordlength',
%% 'naivebaisemin_word_length', 'naive_baiseminword_length',
%% 'naive_baise_minwordlength' oder 'naivebaise_min_wordlength'
%% angegeben werden. Eine Angabe kleiner gleich 0 ist nicht vorgesehen
%% und nicht spezifiziert. Nur positive Zahlen groesser 0 sind fuer diese
%% Option gueltig.
%% Beispiele
%% naive_baisemin_word_length=746
%% naivebaise_min_wordlength=9
%% naive_baiseminword_length=76
%% naive_baise_min_wordlength=1

naive_baise_min_word_length=4

%%--------------------------------------------------------------------
%% Mit der Option 'addressors' kann angegeben werden, ob eine Suche
```

```
%% nach bestimmten Absendern zur Textfilterung standardmaessig genutzt
%% werden soll. Wie man von dieser Einstellung bei der Filterung von
%% Texten abweicht, kann in der zu dem Programm zugehoerigen
%% Dokumentation nachgelesen werden. Ist die Option 'addressors' mit
%% einer '1' oder 'on' gesetzt, wird diese Filtermethode
%% standardmaessig mitgenutzt. Ist diese Option mit einer '0' oder
%% 'off' gesetzt, so wird diese Klassifikationsmethode standardmaessig
%% nicht genutzt.
%% Beispiele:
%% addressors=on
%% addressors=off
%% addressors=0
%% addressors=1

addressors=1

%%--------------------------------------------------------------------
%% Die folgenden Optionen sind 'Datenbankoptionen'. Sie dienen zur
%% Einstellung der Verbindung zur MYSQL-Datenbank. Im folgenden eine
%% kurze Beschreibung dieser Optionen:
%% db_name %% Name der Datenbank - Angabe in natuerlicher Sprache
%% db_user %% Benutzername - Angabe in natuerlicher Sprache
%% db_pass %% Benutzerpasswort - Angabe in natuerlicher Sprache
%% db_host %% Host der Datenbank - Angabe in natuerlicher Sprache
%% Alternativ zu 'db_name' kann auch die Option 'dbname' angegeben
%% werden. Alternativ zu 'db_user' kann auch die Option 'dbuser'
%% angegeben werden. Alternativ zu 'db_pass' kann auch die Option
%% 'dbpass' angegeben werden. Alternativ zu 'db_host' koennen auch die
%% Optionen 'dbhost', 'db_location' oder 'dblocation' angegeben
%% werden.
%% Beispiele:
%% ----------
%% db_name=meinedatenbank
%% db_user=root
%% db_pass=987HgHH
%% dbhost=94.111.123.1
%% ----------
%% db_name=irgendeinedatenbank
%% db_user=fritz
%% db_pass=123456789?
%% dblocation=localhost

db_name=spamfilter
db_pass=12345
```

```
db_user=root
db_host=localhost
```

Anhang C - Spam- und Hamtexte

Spam

1. KGVV ist zum kotzen.
2. Du Arschloch!
3. Eddy ist ein Arsch.
4. Heute !!!Verkaufe Viagra!!! 1,99€ je halbe Tab.
5. Herr Maier ist eine Sau?
6. Heute Vjakra im Angebot! Zu kaufen bei Frau Stumm.
7. Auch Marihuäna kann man hier kaufen. No Problemo!
8. Myst, schon wieder so eine Vorlesung!
9. Mit der alten würde ich auch gern mal fykkän.
10. Wer macht die SCHAISS KOTSE im Raum 777 weg?
11. Wenn ich den A_&_r_._s_c_h erwische ;-/
12. Der Alte frisst wie eine alte **S#a+u**
13. *!Alle P§o_t_’_e’nß“(m)i+t+t+e+l im Angebot!!!*
14. Der Häbärt ist eine -s-a-u
15. Dau bist \\b\\l\\o\\e\\d\\
16. Sieben CHapter EIngangß Essen
17. SAg aber nUr das mit der Veranstaltung
18. Klage nicht sOndern Teile das Essen Zum FriEden
19. MIt Sieben mußT Du gehen
20. Wir Ueberlegen TrotZdem,egal ob dau dat willst.
21. Der Herr S. aus P. ist ein rAsch!

22. Helmut kann nicht fikcen.
23. Kauft Anbolika!!!
24. Du altes ferkl
25. Der Ganze FB 19 ist doch böd
26. !D!as !o!rdentliche D!o!r!f!.
27. *F*ieber *crie*kt ni*ema*nd der zum Arzt geht
28. Meine ->Katze ist h->o->tter ->zum ->erledigen
29. +v+iel +arbeit ist das +ganz und ga+r nicht! +a
30. -S_ie will -a_-u_ch leckeres Essen!!!
31. Frau Pelz fikt gerne.
32. Her Grusepeter ist ein richtiger arch
33. Alle FB 18ler sind dof
34. Der Henning ist ein ferkl
35. Kauft doch endlich Anabolka
36. Als Hans plötzlich wieder Anabolika 1,80 geht er
37. Veranstalltung fällt aus doof sind die FB66'er
38. Rechnernetztze ist heute im Raum 7 Marihuana
39. Haus Hund geht auch Vorlesung Arsch soll das Herr
40. Niemand geht zu Wutz mit nach Hause Her Hipp
41. Willst Du mich verführen? Dann ruf an: 0190666666
42. Nimm drogen und Du bist alle Sorgen los
43. geile Schnecken auf www.XXXhasteXXX.com
44. Die Veranstaltung 'Belohnen' ist richtig scheiße!
45. Hure
46. kdj skfh wlfjojfolwe lskf snfjnj nnjdj jjj ls
47. Bring Dich um!
48. Pornos zum downloaden: www.load.de

49. Fick Dich ins Knie Manuel.
50. Carl ist schwul!
51. Ich bring euch alle um.
52. Verkaufe s€xuelle Erfahrungen.
53. Spende für die Taliban: Ktnr.: 875003754 BLZ: 550999888
54. Gabi hat ein Gesicht wie ein Affe.
55. Kinderpornos zu verkaufen: www.hsgggdf.ru
56. Sieg Heil! Sieg Heil!
57. Hoch lebe Adolf Hitler!
58. Ausländer Raus!
59. Neger haben in Deutschland nichts zu suchen!
60. Kleiner, ich beiß Dir in die Nüsse.
61. JWKSDHKnwoefHHD jjkd*++wq kqwd kllmlml
62. Hund fall Zufall gesehen darum weil oben.
63. Ich mach Dich fertig Du sau.
64. Deutschland Deutschland über alles!
65. Nudel Fresser!
66. Herr Hanz Dampf mit seinem dicken Bauch ist eine verdammte Drecksau.
67. Natürlich kann dieses Arsch nichts für seine stupiden Argumentationen. Der ist halt behindert.
68. Behinderte haben es schwer. Es ist darauf zu achten, wie schwer es Frau Maier von den Juristen hat!
69. Wir danken allen Leuten, die nicht bei dem Aufbau für jhgfds geholfen haben. Ihr Ärsche !
70. Ein Dank an die Sportler, die immer das beste Anaboliker verkaufen.
71. Wenn ich so d00f wär wie die Angelisten, dann würd ich nicht so den dicken machen
72. Heute im Sonderangebot !!! pot€nzmittel für jedermann! Im Sonderangebot für nur 1,99 €. !!!!

73. Ich Kotz€ auf die alte. Geht auf keinen Fall zur Frau Kieselerde. Die gibt nur schlechte Noten.

74. Ein herzlichen Gruß an die Person mit dem #Schild Ems-AA-54. Fahr Dein Scheiz Auto weg.

75. Arschl0ch, Wixer, Hürensohn, Deine Mutter kenn ich schon Herr Feldenkirch. ;-)

76. Achtung, Achtung, wir warnen vor ferk'e| auf dem dem Dach vom D-Gebäude, die andere Leute anspucken

77. A_____R__ _____S__ _____C__ _____H__ _____

78. Wir wünschen der fetten Säü an der Theke ein Fettreiches Weihnachtsfest.

79. An den @rsch, der mir eine Macke ins Auto gefahren hat. Sofort melden unter: 01000 8379153

80. Hallo Elke, es tut mir wirklich leid, aber Du siehst aus, als hätt Dir jemand in Gesicht gek@tzt:-

81. Du, Ob es Oben in der caFete besser schmeckt. Genau das seid Ihr aus dem Fachbereich 998!

82. Kleider machen nOch lange keine leuTe. Wers glaubt, Zu dEm sollte mal der Psychologe kommen.

83. dein hunger ist Auch uneRsättlich! wenn ich mir so etwas anSCHaue. ganz großes kino für das Stubi

84. die fete am montag war So gut, dass iCH mEIne meinung in den groSSbuchstabEn kuntgetan habe.

85. der habicht Fängt dEn fuchs früh moRgens und nicht erst wenn es Kein zurück gibt rEgnet L.

86. Warum ist Dein hcsrA so fett Frau Hannelore Müller? Antworte mir!

87. Suche Freundin, blond, 90 - 60 - 90, dicke Dinger und geilen schrA.

88. Sabine, wenn ich so bldö wär wie Du, dann würde ich vom Dach des D - Gebäude springen

89. IIIIhhh, ich hab grad gesehen Hans, wie Du einen Popel aus Deiner Nase geholt Hast! Ferkle

90. Bilder von fetten Ärhcsen zu sehen auf www.haste-noch-nicht-gesehen.

91. Als ich am D!ienstag mo!rgen !ohne murren au!fgestanden bin, machte ich mir erst mal ein Müsli.

92. Morgen-s-tund hat Gold im Mund. Wer das gl-au-bt, ist selbst dran schuld.

93. Ein Spruch, von dëm ich so“viel halte, wie ich makiere ist, der frühe Vo“gel f“ängt den Wurm.

94. Seht meine b+lusses und ihr wisst, was ich vo+n de+m netten Herr Prof. Kid+denberger halte.

95. Es le?b?en die Sonderzeichen, die einem Text eine Botsc?haft be?ifügen. ?Nur ?di?e F?ragezeichen ?t

96. Hallo Frau Schulz, ich sag ihnen einfach mal, was ich von ihnen halte: Sie sind beindert!

97. Ich habe gesehen, dass mir irgend ein asch den Spiegel abgefahren hat. ICH ERWISCH DICH!

98. Dr. Schall ist ein alter Lustmolch. der schaut den kleinen Studentinnen immer auf den arch

99. Ihr seit doch alle Beindert. Damit meine ich nicht, dass ihr Inder seit.

100. Lacht ihr nur ihr helden. Ich werde euch mit koze beschmieren, auf dass ihr stinkt.

101. Das ist alles belangloser Text, um nur ein Wort zu verstecken, nämlich Ficken!

102. Eines Abends Veranstaltung fällt aus und verschiebt kotze raum wort erkennen verkaufe nichts.

103. Eine Idee muss man einfach haben ferkel um berühmt zu werden ohne sonderzeichen.

104. Ein Hase ging gerne spazieren und traf ganz zufällig auf ein blöd Hamster.

105. Wer glaubt als jsf sleifj lwkflkw jkjf arsch ölksf kljflwk lksfjlkw kljflkwfje eeppüqölliwiüp

106. Der dicke Professor behindert meinen Blick zur Tafel.

107. Arschloh, wixer, Hurensohn, deine Mutter kenn ich schon!

108. !!!!!!!!!! Das Mensaessen schmeckt scheiße !!!!!!!!!!

109. Wahlveranstaltung der DVU. 19.06.2008! Jeder ist WILLKOMMEN.

110. Vordiplomsnoten verschiedenster Studenten: 3,7 Michael Klassen - 2,3 Hans Franz Sturm.

111. www.geile-Frauen-immer willig.org/studentin - SAU GEIL.

112. Stehst auch Du auf kleine Kinder? Dann melde Dich bei Prof. Halle

113. !!

114. *”’_:;;==“))(§/$%(()“(89287637/77777(/§%!HDKWÄÄÄÖÖLD kedo093 pe-üwfwefpopoiu

115. Nacktschnecken zum anfassewn und bumsen. Am Montag im Kranit.

116. Sollte mich noch einmal jemand zuparken und ich erwisch den, der sollte besser schon mal rennen und sich nicht erwischen lassen.

117. NPD Veranstaltung, Dienstag 13 Uhr in der Rheinmoselhalle. Natürlich gibt es Freibier für alle. Werde auch Du ein Mitglied der NPD.

118. HSGGttajfehKJJKKUjkah kkdj jdsj jkdj skjdh jkskkkkI jBU janbdjiaid kjss988nb ajdnkjajd aoldhjja8 kdjja

119. Suche offene Mitbewohnerin, die mal was neues erleben will! Du bist experimentierfrudig, dann melde Dich: 02633/8934

120. Bist du weiblich, geil, willig und hast dicke Äpfel? Du benötigst eine praktische Aufklärungsstunde? Dann schreib mir eine Mail: jonas@web.de

121. ! !

122. 5 5 5 5 ;

123. Hi Erno du alter Schwanz. Wie geht es Dir? lange nichts mehr von Dir gehört:-) Hab Dich im Internet wieder gefunden: www.myFilm.de/fkk/erno01.jpg

124. Die Brillenschlange, die mir gegen meinen Wagen gefahren ist, bei mir melden! Sonst gibt es paar auf die Eier (sofern männlich)!

125. Das ist der Hammer, Sind die hinter der Mensatheke mal wieder freundlich (ironie) Die blöde SAU.

Ham

1. Verkaufe alten Schrank. Tel: 0261/123456
2. Wohnung in Bad Ems zu vermieten/verkaufen
3. Ich liebe Dich Anna!
4. Benötige macbook hilfe! defnick@uni...
5. Nachhilfe in Mattte gesucht!

6. Raum TI2 verlegt. Neuer Raum G122.
7. Ich liebe Heike!
8. Heute Abend Fete im Dreams! Eintritt 2€!
9. Schlüssel gefunden. Abgegeben an der Pforte.
10. Christa, ich will ein Kind von Dir!
11. Das Mensaessen ist heute köstlich.
12. Achtung: Falsche “Kabel DeutschlandTTechniker unterwegs!
13. Verkaufe blauen Polo. Baujahr 90, Verhandlungsbasis 1500€. Meine Telefonnr. ist 0261/67553445
14. Vortrag über 'Adolf Hitler' am 20.04.2010 um 20 Uhr in der Aula.
15. Wir gratulieren Frau Dr Krieskram zu ihrer Habilitation. Das Laufteam Nord.
16. Ich bin jung, muskulös und 1,80 groß. Wenn Du mich kennenlernen willst, melde Dich: geilerBursch@gmx.de
17. Wurde zugeparkt. Der Fahrer mit dem Kennzeichen Ko-jd-999 möge doch bitte umgehend sein Auto wegfahren.
18. Vortrag zur 'Behinderten Förderung' findet nächste Woche Dienstag um 19 Uhr im D224 statt.
19. Maria, ich liiiiiiiiiieeeeeeeeeebe Dich!!!!!!!!!!
20. Verkaufe Bücher AnalysisI und AnalysisII. Wie neu! Bei Interesse einfach melden: schuhgrat@uni-koblenz.de
21. Jetzt anmelden im ASTA als OPA. JEDER der schon was von der Uni weiss ist willkommen.
22. Die Universitätsbib ist zwischen dem 21. - 28.08. geschlossen. Während dieser Zeit können die Bücher im Raum C111 zurückgegeben werden.
23. Jemand mit Erfahrung über AMD-Prozessoren gesucht. Will mir einen neuen Computer kaufen. Antwort bitte in die Newsgroup infko-general
24. Fraunhofer FIT sucht AR-Testspieler für das Spiel 'Timewarp'. Infos zum Projekt auf: http://www.ipcity.eu/?page_id=10!
25. Schaut euch in der Newsgroup den Beitrag mit dem Subject: 'Das Verfassen guter akademischer Texte ist ein Haftgrund'. Das ist ja wohl die Höhe!

26. Das Auto mit dem Nummernschild 'EMS-KH-2334' sofort wegfahren. Das Auto steht im absulutem Halteverbot und behindert einige Fahrzeuge.

27. Die Kevag will zum ersten September die Strompreise erhöhen. Ich rufe zum Protest auf. - Jetzt den Stromanbieter wechseln.

28. Mein Roller (Piaggio Sfera 50) ist krank. Wer kann mir helfen?!? Ich habe keine Lust zum teuren EXperten zu gehen. thorstSchu@uni-koblenz.de

29. Welches Ferkel hat seinen Haufen im Männerklo im B-Gebäude neben die Toilette gesetzt? Das STINKT da wie die Sau.

30. Wer will mich kennenlernen: weiblich, 1,75 m, liebevoll. Suche jemanden der mit mir die einsamen Winterabende teilt.

31. Zwischnmieter gesucht für den Zeitraum vom 01.10.2007 bis 31.01.2007. Wohnung: Koblenz City, 2 - Zimmer, Küche, Bad. Warm 375 €. Tel: 0261-676654

32. Hat jemand Vorschläge für eine gute Celeron-Kühlung (466er in SLOT-1)? Antwort an ich@uni-koblenz.de

33. Kann jemand noch auf den Rechner linux.uni-koblenz.de zugreifen? Bitte in der NG-general antworten. Danke!

34. Hat hier jemand ne Idee wo man einen Twain Treiber für en TTarga TS 600cSScanner bekommen kann? sabineter@uni-koblenz.de

35. Bitte dran Denken: Die Rückmeldefrist läuft zum 01.10.2004 aus. Bitte rechtzeitig Beitrag überweisen.

36. Ich saß am Montag in der Vorlesung 'Einführung in die Pädagogik' und sah in der ersten Reihe ein blondes Mädchen mit einem Zopf. Falls Du das Mädchen bist, melde Dich unter hans@uni-...

37. Auch dieses Jahr wieder im KKDFG: Große Heiligabend NUßPARTY. Eintritt natürlich frei. Am 24.12 ab 20 Uhr. Nüsse ebenfalls kostenfrei, so viele Du nur essen kannst, ob Walnuss oder Haselnuss.

38. Hallo, hier im Wohnheim Äuf dem hellen Weyerffunktioniert weder das Aufrufen von Websites, noch das Abfragen von eMails. Woran liegt das? Icq, IRC oder Newsgroups funktionieren ohne Probleme.

39. Vom 21.08.2007 bis 28.08.2007 sind ALLE Server der Universität-Koblenz down. Wir bitten um Verständnis. Das Rechenzentrum wünscht Euch eine entspannte vorlesungsfreie Zeit.

40. Suche jemanden, der DINA3 einlaminieren kann oder zumindest mir sagen kann, wer so etwas mahct. Es geht um 3 bis 5 Plakate. Antwort bitte an schreibMir113@gmx.de. Anne Sterk.

41. Aufklärung in der Schule rückt immer stärker in den Hintergrund und AIDs verbreitet sich wieder immer mehr. Ein Seminar 'Aufklärung in der Schule' findet am Do, den 07.07.08 in Raum B017 statt.

42. Bitte melden Sie sich für Prüfungen bei mir, neben der Anmeldung beim Prüfungsamt(!), auch auf meiner Homepage http://www.peter-roedler.de - Button links oben - an. (Roedler)

43. Die Nachklausur zur Vorlesung 'Logik für Informatiker' findet am Montag den 22.10.07 um 18 Uhr c.t. in D028 statt.Wer an der Klausur teilnehmen will, meldet sich bitte über Metoo an.

44. An alle Betroffenen: Die große Sporthalle (S14) ist laut Auskunft der Universitätsverwaltung bis zum 05.01. für die Nutzung gesperrt. MfG, Jan Krahler

45. Suche Nachmieter für Wohnung! Triererstr. 105, 1 Zimmer, Küche, Bad, Keller, Fahrradkeller, 36m2, 3. Laminat. Waschmaschinen zur Mitbenutzung im Haus, ca. 220€, kalt ab 01.08.07 oder früher.

46. Die bei der Info(rmations)-Veranstaltung angekuendigte mailingliste fuer Interessierte des Kompaktkurses IPCV in Lappeenranta ist fertig unter mailhost.uni-koblenz.de/mailman/listinfo/ipcv

47. Ich habe heute in Metternich im Raum MA 003 meinen Collegeblock vergessen!! Ich würde mich sehr freuen, wenn der Finder sich bei mir meldet... andi@uni-...

48. Neue Modems [56 K, V.90] gibt es in allen Märkten (MediaMarkt, Saturn, VOBIS etc.) ab ca. 90 EURO, wenn sie nicht gerade von den Marktführeren ELSA oder 3Com sein müssen...

49. Wir möchten nochmals herzlich einladen zu unserem Vortrag im Rahmen des Koblenzer Wirtschaftsinformatik Forums am Donnerstag, 08. Juni 2000 um 16:00 h im Raum MA 120.

50. ich biete eine mifahrgelegenheit von koblenz nach hamburg an (sogar bis rendsburg), zwischenstops sind möglich (z.B. Osnabrück, Leverkusen, Münster, Bremen etc). katFerch@uni-koblenz.de

Anhang D - Spam- und Hamauswertung

Hier eine Übersicht der Datensätze. Im folgenden wird nur noch die laufende Nummer angegeben, um den Datensatz zu identifizieren. In der Spalte 'GK' wird mit 'ja' angezeigt, dass die Groß- und Kleinschreibung bei der Merkmalsvektorerstellung berücksichtigt wurde, ein 'nein' bedeutet im Gegensatz dazu, dass diese nicht berücksichtigt wurde. In der Spalte 'erlaubte Zeichen' wird angegeben, welche Zeichen nicht durch das leere Wort ersetzt wurden. In der Spalte 'Ersetzungen' ist angegeben, welche Zeichen durch ein anderes Zeichen ersetzt worden sind. Diese Ersetzung erfolgte vor der 'Filterung' der erlaubten Zeichen. Die Spalte 'VK' gibt an, welches Vektormodell benutzt wurde, also wie die Worte als Merkmal repräsentiert worden sind. 'binär' gibt an, dass im entsprechenden Merkmal nur verzeichnet wurde, ob das Wort im Text vorhanden war oder nicht. 'normal' gibt in dem entsprechenden Merkmal an, wie oft das Wort in dem Text vorkam.

Lfdnr.	GK	Zeichen	erlaubte Zeichen	Ersetzungen	VK
1	nein	1	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
2	nein	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
3	nein	7	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
4	ja	1	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
5	ja	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
6	ja	7	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	normal
7	nein	1	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	binär

Lfdnr. : Laufende Nummer [Zahl größer 0]

GK: Groß-/Kleinschreibung beachtet [ja/nein]

Zeichen: Wieviele Zeichen muss das Wort mindestens besitzen [Zahl größer 0]

Erlaubte Zeichen: Welche Zeichen werden im Wort nicht entfernt [natürlich sprachlicher Text]

Ersetzungen: Welche Zeichen wurden durch einen Buchstaben ersetzt oder wurden ausgeschlossen... [natürlich sprachlicher Text]

VK: Vektormodell - war der benutzte Vektor numerisch oder binär [normal/binär]

Lfdnr.	GK	Zeichen	'Erlaubte' Zeichen	Ersetzungen	VK
8	nein	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	binär
9	ja	1	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	binär
10	ja	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß, _	Keine Ersetzungen	binär
11	nein	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß	Keine Ersetzungen	normal
12	ja	3	Buchstaben a bis Z und A - Z, Zahlen und ä, Ä, ö, Ö, ü, Ü, ß	Keine Ersetzungen	normal
13	ja	3	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ae' und 'ä' wird zu a, ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a	normal

Lfdnr. : Laufende Nummer [Zahl größer 0]

GK: Groß-/Kleinschreibung beachtet [ja/nein]

Zeichen: Wieviele Zeichen muss das Wort mindestens besitzen [Zahl größer 0]

Erlaubte Zeichen: Welche Zeichen werden im Wort nicht entfernt [natürlich sprachlicher Text]

Ersetzungen: Welche Zeichen wurden durch einen Buchstaben ersetzt oder wurden ausgeschlossen... [natürlich sprachlicher Text]

VK: Vektormodell - war der benutzte Vektor numerisch oder binär [normal/binär]

Lfdnr.	GK	Zeichen	'Erlaubte' Zeichen	Ersetzungen	VK
14	nein	3	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ae' und 'ä' wird zu 'a', ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a	normal
15	ja	7	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ae' und 'ä' wird zu 'a', ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a, Stopwörter ausgeschlossen!	normal
16	nein	3	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ae' und 'ä' wird zu 'a', ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a, Stopwörter ausgeschlossen!	normal

Lfdnr. : Laufende Nummer [Zahl größer 0]

GK: Groß-/Kleinschreibung beachtet [ja/nein]

Zeichen: Wieviele Zeichen muss das Wort mindestens besitzen [Zahl größer 0]

Erlaubte Zeichen: Welche Zeichen werden im Wort nicht entfernt [natürlich sprachlicher Text]

Ersetzungen: Welche Zeichen wurden durch einen Buchstaben ersetzt oder wurden ausgeschlossen... [natürlich sprachlicher Text]

VK: Vektormodell - war der benutzte Vektor numerisch oder binär [normal/binär]

Lfdnr.	GK	Zeichen	'Erlaubte' Zeichen	Ersetzungen	VK
17	ja	3	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ä' wird zu 'ae' und 'Ä' wird zu 'Ae', ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a, Stopwörter ausgeschlossen!	normal
18	nein	3	Buchstaben a bis Z und A bis Z, Zahlen und ß	'.' wird zu ' ', 'ä' wird zu 'ae' und 'Ä' wird zu 'Ae', ... (Alle Umlaute in dieser Art und Weise überführen), '€' wird zu 'e', '@' wird zu a, Stopwörter ausgeschlossen!	normal

Lfdnr. : Laufende Nummer [Zahl größer 0]

GK: Groß-/Kleinschreibung beachtet [ja/nein]

Zeichen: Wieviele Zeichen muss das Wort mindestens besitzen [Zahl größer 0]

Erlaubte Zeichen: Welche Zeichen werden im Wort nicht entfernt [natürlich sprachlicher Text]

Ersetzungen: Welche Zeichen wurden durch einen Buchstaben ersetzt oder wurden ausgeschlossen... [natürlich sprachlicher Text]

VK: Vektormodell - war der benutzte Vektor numerisch oder binär [normal/binär]

Hier sehen sie eine vollständige Übersicht über die Testergebnisse verschiedener Klassifikationsverfahren. Die Laufende Nummer, Spalte 'Lfdnr.' gibt an, auf welchen Datensatz dieses Testergebnis sich bezieht. In der Spalte 'TP' kann die absolute Zahl der 'true positives' abgelesen werden, in der Spalte 'FP' die 'false positives', in der Spalte 'FN' die 'false negatives' und in der Spalte 'TN' die 'true negatives'. In den Spalten 'richtig erkannt' und 'falsch erkannt' sind die Angaben in Prozent, wieviele von den gesamten Daten richtig bzw. wieviele falsch erkannt wurden. In der Spalte 'TM' kann entnommen werden, welche Testmethode verwendet wurde. 'cv' steht für die Testmethode 'cross validation' und 'ps' für 'percentage split'. Die Bedeutung der Spalten 'Recall' und 'Precision' kann dem Grundlagenkapitel entnommen werden. Die 'true positives' sind die richtig klassifizierten Spamnachrichten, die 'true negatives' die richtig Klassifizierten Hamnachrichten usw.

Testergebnisse 'Naive Baise':

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
1	103	21	22	29	0,824	0,8306	0,7542	0,2457	cv
1	106	23	19	27	0,848	0,8217	0,76	0,24	cv
1	102	19	23	31	0,816	0,8429	0,76	0,24	cv
2	102	25	23	25	0,816	0,8031	0,7257	0,2742	cv
2	103	25	22	25	0,824	0,8046	0,7314	0,2685	cv
2	103	25	22	25	0,824	0,8046	0,7314	0,2685	cv
3	121	27	4	13	0,968	0,8175	0,8121	0,1878	cv
3	121	40	4	10	0,968	0,7515	0,7485	0,2514	cv
3	120	37	5	13	0,96	0,7643	0,76	0,24	cv
4	102	20	23	30	0,816	0,8360	0,7542	0,2457	cv
4	98	23	27	27	0,784	0,8099	0,7142	0,2857	cv
4	103	20	22	30	0,824	0,8373	0,76	0,24	cv
5	97	23	28	27	0,776	0,8083	0,7085	0,2914	cv
5	101	21	24	29	0,808	0,8278	0,7428	0,2571	cv
5	101	22	24	28	0,808	0,8211	0,7371	0,2628	cv
6	121	37	4	13	0,968	0,7658	0,7657	0,2342	cv
6	121	41	3	9	0,9758	0,7469	0,7471	0,2528	cv
6	121	39	4	11	0,968	0,7562	0,7542	0,2457	cv
7	109	23	16	27	0,872	0,8257	0,7771	0,2228	cv
7	109	19	19	31	0,8515	0,8515	0,7865	0,2134	cv
7	109	24	16	26	0,872	0,8195	0,7714	0,2285	cv
8	103	28	22	22	0,824	0,7862	0,7142	0,2857	cv
8	100	24	25	26	0,8	0,8064	0,72	0,28	cv
8	98	25	27	25	0,784	0,7967	0,7028	0,2971	cv
9	100	24	25	26	0,8	0,8064	0,72	0,28	cv
9	101	21	24	29	0,808	0,8278	0,7428	0,2571	cv
9	103	25	22	25	0,824	0,8046	0,7314	0,2685	cv
10	100	22	25	28	0,8	0,8196	0,7314	0,2685	cv
10	103	21	22	29	0,824	0,8306	0,7542	0,2457	cv
10	98	23	27	27	0,784	0,8099	0,7142	0,2857	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
2	54	12	13	9	0,8059	0,8181	0,7159	0,2840	ps
2	54	23	4	7	0,9310	0,7012	0,6931	0,3068	ps
2	53	15	10	10	0,8412	0,7794	0,7159	0,2840	ps
5	60	12	6	10	0,9090	0,8333	0,7954	0,2045	ps
5	57	17	4	10	0,9344	0,7702	0,7613	0,2386	ps
5	53	13	10	12	0,8412	0,8030	0,7386	0,2613	ps
11	99	25	26	25	0,792	0,7983	0,7085	0,2914	cv
11	103	25	22	25	0,824	0,8046	0,7314	0,2685	cv
11	104	22	21	28	0,832	0,8253	0,7542	0,2457	cv
12	99	22	26	28	0,792	0,8181	0,7257	0,2742	cv
12	107	21	18	29	0,856	0,8359	0,7771	0,2228	cv
12	100	22	25	28	0,8	0,8196	0,7314	0,2685	cv
13	101	20	24	30	0,808	0,8347	0,7485	0,2514	cv
13	98	22	27	28	0,784	0,8166	0,72	0,28	cv
13	103	19	21	33	0,8306	0,84426	0,7727	0,2272	cv
14	106	25	19	25	0,848	0,8091	0,7485	0,2514	cv
14	105	24	20	26	0,84	0,8139	0,7485	0,2514	cv
14	101	26	24	24	0,808	0,7952	0,7142	0,2857	cv
15	125	46	0	4	1	0,7309	0,7371	0,2628	cv
15	125	46	0	4	1	0,7309	0,7371	0,2628	cv
15	125	46	0	4	1	0,7309	0,7371	0,2628	cv
16	112	28	13	22	0,896	0,8	0,7657	0,2342	cv
16	110	28	15	22	0,88	0,7971	0,7542	0,2457	cv
16	112	28	13	22	0,896	0,8	0,7657	0,2342	cv
17	118	28	7	22	0,944	0,8082	0,8	0,2	cv
17	115	27	10	23	0,92	0,8098	0,7885	0,2114	cv
17	118	28	7	22	0,944	0,8082	0,8	0,2	cv
18	112	29	13	21	0,896	0,7943	0,76	0,24	cv
18	113	28	12	22	0,904	0,8014	0,7714	0,2285	cv
18	115	27	10	23	0,92	0,8098	0,7885	0,2114	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Testergebnisse des 'K-Nearest neighbor Algorithmus':

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
1	125	49	0	1	1	0,7183	0,72	0,28	cv
1	125	49	0	1	1	0,7183	0,72	0,28	cv
1	125	49	0	1	1	0,7183	0,72	0,28	cv
2	125	49	0	1	1	0,7183	0,72	0,28	cv
2	125	49	0	1	1	0,7183	0,72	0,28	cv
2	125	49	0	1	1	0,7183	0,72	0,28	cv
3	125	49	0	1	1	0,7183	0,72	0,28	cv
3	125	49	0	1	1	0,7183	0,72	0,28	cv
3	125	49	0	1	1	0,7183	0,72	0,28	cv
4	125	49	0	1	1	0,7183	0,72	0,28	cv
4	125	50	0	0	1	0,7142	0,7142	0,2857	cv
4	125	49	0	1	1	0,7183	0,72	0,28	cv
5	125	49	0	1	1	0,7183	0,72	0,28	cv
5	125	49	0	1	1	0,7183	0,72	0,28	cv
5	125	49	0	1	1	0,7183	0,72	0,28	cv
6	125	49	0	1	1	0,7183	0,72	0,28	cv
6	125	49	0	1	1	0,7183	0,72	0,28	cv
6	125	49	0	1	1	0,7183	0,72	0,28	cv
7	125	49	0	1	1	0,7183	0,72	0,28	cv
7	124	49	1	1	0,992	0,7167	0,7142	0,2857	cv
7	124	49	1	1	0,992	0,7167	0,7142	0,2857	cv
8	125	49	0	1	1	0,7183	0,72	0,28	cv
8	125	49	0	1	1	0,7183	0,72	0,28	cv
8	125	49	0	1	1	0,7183	0,72	0,28	cv
9	125	50	0	0	1	0,7142	0,7142	0,2857	cv
9	125	49	0	1	1	0,7183	0,72	0,28	cv
9	125	49	0	1	1	0,7183	0,72	0,28	cv
10	125	49	0	1	1	0,71839	0,72	0,28	cv
10	125	49	0	1	1	0,71839	0,72	0,28	cv
10	125	49	0	1	1	0,71839	0,72	0,28	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
2	67	21	0	0	1	0,7613	0,7613	0,2386	ps
2	58	30	0	0	1	0,6590	0,6590	0,3409	ps
2	63	25	0	0	1	0,7159	0,7159	0,2840	ps
5	66	21	0	1	1	0,7586	0,7613	0,2386	ps
5	61	26	0	1	1	0,7011	0,7045	0,2954	ps
5	63	25	0	0	1	0,7159	0,7159	0,2840	ps
11	125	49	0	1	1	0,7183	0,72	0,28	cv
11	125	49	0	1	1	0,7183	0,72	0,28	cv
11	125	49	0	1	1	0,7183	0,72	0,28	cv
12	125	49	0	1	1	0,7183	0,72	0,28	cv
12	125	50	0	0	1	0,7142	0,7142	0,2857	cv
12	125	49	0	1	1	0,7183	0,72	0,28	cv
13	125	49	0	1	1	0,7183	0,72	0,28	cv
13	125	49	0	1	1	0,7183	0,72	0,28	cv
13	125	49	0	1	1	0,7183	0,72	0,28	cv
14	125	50	0	0	1	0,7142	0,7142	0,2857	cv
14	125	49	0	1	1	0,7183	0,72	0,28	cv
14	125	49	0	1	1	0,7183	0,72	0,28	cv
15	124	50	1	0	0,992	0,7126	0,70857	0,2914	cv
15	124	50	1	0	0,992	0,7126	0,70857	0,2914	cv
15	124	50	1	0	0,992	0,7126	0,70857	0,2914	cv
16	125	50	0	0	1	0,7142	0,7142	0,2857	cv
16	125	50	0	0	1	0,7142	0,7142	0,2857	cv
16	125	50	0	0	1	0,7142	0,7142	0,2857	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Testergebnisse der 'Supported Vector Machine':

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
1	120	39	5	11	0,96	0,7547	0,7485	0,2514	cv
1	119	37	6	13	0,952	0,7628	0,7542	0,2457	cv
1	119	38	6	12	0,952	0,7579	0,7485	0,2514	cv
2	121	40	4	10	0,968	0,7515	0,7485	0,2514	cv
2	121	40	4	10	0,968	0,7515	0,7485	0,2514	cv
2	119	40	6	10	0,952	0,7484	0,7371	0,2628	cv
3	122	45	3	5	0,976	0,7305	0,7257	0,2742	cv
3	123	45	2	5	0,984	0,7321	0,7314	0,2685	cv
3	122	45	3	5	0,976	0,7305	0,7257	0,2742	cv
4	121	38	4	12	0,968	0,7610	0,76	0,24	cv
4	121	36	4	14	0,968	0,7707	0,7714	0,2285	cv
4	121	38	4	12	0,968	0,7610	0,76	0,24	cv
5	121	37	4	13	0,968	0,7658	0,7657	0,2342	cv
5	121	40	4	10	0,968	0,7515	0,7485	0,2514	cv
5	119	37	6	13	0,952	0,7628	0,7542	0,2457	cv
6	123	44	2	6	0,984	0,7365	0,7371	0,2628	cv
6	123	45	2	5	0,984	0,7321	0,7314	0,2685	cv
6	123	44	2	6	0,984	0,7365	0,7371	0,2628	cv
7	118	36	7	14	0,944	0,7662	0,7542	0,2457	cv
7	118	36	7	14	0,944	0,7662	0,7542	0,2457	cv
7	119	36	6	14	0,952	0,7677	0,76	0,24	cv
8	119	38	6	12	0,952	0,7579	0,7485	0,2514	cv
8	119	36	6	14	0,952	0,7677	0,76	0,24	cv
8	119	38	6	12	0,952	0,7579	0,7485	0,2514	cv
9	120	32	5	18	0,96	0,7894	0,7885	0,2114	cv
9	119	34	6	16	0,952	0,7777	0,7714	0,2285	cv
9	118	35	7	15	0,944	0,7712	0,76	0,24	cv
10	119	37	6	13	0,952	0,7628	0,7542	0,2457	cv
10	119	34	6	16	0,952	0,7777	0,7714	0,2285	cv
10	120	37	5	13	0,96	0,7643	0,76	0,24	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
2	63	13	4	8	0,9402	0,8289	0,8068	0,1931	ps
2	57	27	1	3	0,9827	0,6785	0,6818	0,3181	ps
2	61	24	2	1	0,9682	0,7176	0,7045	0,2954	ps
5	64	16	2	6	0,9696	0,8	0,7954	0,2045	ps
5	59	25	2	2	0,9672	0,7023	0,6931	0,3068	ps
5	60	21	3	4	0,9523	0,7407	0,7272	0,2727	ps
11	121	40	4	10	0,968	0,7515	0,7485	0,2514	cv
11	120	41	5	9	0,96	0,7453	0,7371	0,2628	cv
11	122	39	3	11	0,976	0,7577	0,76	0,24	cv
12	120	38	5	12	0,96	0,7594	0,7542	0,2457	cv
12	120	39	5	11	0,96	0,7547	0,7485	0,2514	cv
12	121	39	4	11	0,968	0,7562	0,7542	0,2457	cv
13	122	39	3	11	0,976	0,7577	0,76	0,24	cv
13	120	37	5	13	0,96	0,7643	0,76	0,24	cv
13	121	37	4	13	0,968	0,7658	0,7657	0,2342	cv
14	121	39	4	11	0,968	0,7562	0,7542	0,2457	cv
14	120	40	5	10	0,96	0,75	0,7428	0,2571	cv
14	123	38	2	12	0,984	0,7639	0,7714	0,2285	cv
15	122	48	3	2	0,976	0,7176	0,7085	0,2914	cv
15	123	48	2	2	0,984	0,7192	0,7142	0,2857	cv
15	123	48	2	2	0,984	0,7192	0,7142	0,2857	cv
16	123	41	2	9	0,984	0,75	0,7542	0,2457	cv
16	122	41	3	9	0,976	0,7484	0,7485	0,2514	cv
16	124	42	1	8	0,992	0,7469	0,7542	0,2457	cv
17	124	41	1	9	0,992	0,7515	0,76	0,24	cv
17	122	42	3	8	0,976	0,7439	0,7428	0,2571	cv
17	123	44	2	6	0,984	0,7365	0,7371	0,2628	cv
18	123	41	2	9	0,984	0,75	0,7542	0,2457	cv
18	123	42	2	8	0,984	0,7454	0,7485	0,2514	cv
18	123	42	2	8	0,984	0,7454	0,7485	0,2514	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Testergebnisse des Entscheidungsbaums:

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
1	105	34	20	16	0,84	0,7553	0,6914	0,3085	cv
1	110	36	15	14	0,88	0,7534	0,7085	0,2914	cv
1	109	36	16	14	0,872	0,7517	0,7028	0,2971	cv
2	110	45	15	5	0,88	0,7096	0,6571	0,3428	cv
2	110	48	15	2	0,88	0,6962	0,64	0,36	cv
2	109	44	16	6	0,872	0,7124	0,6571	0,3428	cv
3	125	50	0	0	1	0,7142	0,7142	0,2857	cv
3	125	50	0	0	1	0,7142	0,7142	0,2857	cv
3	125	50	0	0	1	0,7142	0,7142	0,2857	cv
4	104	35	21	15	0,832	0,7482	0,68	0,32	cv
4	104	33	21	17	0,832	0,7591	0,6914	0,3085	cv
4	106	39	19	11	0,848	0,7310	0,6685	0,3314	cv
5	105	45	20	5	0,84	0,7	0,6285	0,3714	cv
5	100	39	25	11	0,8	0,7194	0,6342	0,3657	cv
5	101	43	24	7	0,808	0,7013	0,6171	0,3828	cv
6	125	50	0	0	1	0,7142	0,7142	0,2857	cv
6	125	50	0	0	1	0,7142	0,7142	0,2857	cv
6	125	50	0	0	1	0,7142	0,7142	0,2857	cv
7	109	32	16	18	0,872	0,7730	0,7257	0,2742	cv
7	108	39	17	11	0,864	0,7346	0,68	0,32	cv
7	111	37	14	13	0,888	0,75	0,7085	0,2914	cv
8	107	46	18	4	0,856	0,6993	0,6342	0,3657	cv
8	113	45	12	5	0,904	0,7151	0,6742	0,3257	cv
8	109	47	16	3	0,872	0,6987	0,64	0,36	cv
9	102	33	23	17	0,816	0,7555	0,68	0,32	cv
9	104	34	21	16	0,832	0,7536	0,6857	0,3142	cv
9	106	33	19	17	0,848	0,7625	0,7028	0,2971	cv
10	100	41	25	9	0,8	0,7092	0,6228	0,3771	cv
10	107	40	18	10	0,856	0,7278	0,6685	0,3314	cv
10	101	43	24	7	0,808	0,7013	0,6171	0,3828	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Lfdnr.	TP	FP	FN	TN	Recall	Precision	richtig erkannt	falsch erkannt	TM
2	62	17	5	4	0,9253	0,7848	0,75	0,25	ps
2	56	26	2	4	0,9655	0,6829	0,6818	0,3181	ps
2	61	23	2	2	0,9682	0,7261	0,7159	0,2840	ps
5	54	19	12	3	0,8181	0,7397	0,6477	0,3522	ps
5	57	24	4	3	0,9344	0,7037	0,6818	0,3181	ps
5	53	22	10	3	0,8412	0,7066	0,6363	0,3636	ps
11	115	45	10	5	0,92	0,7187	0,6857	0,3142	cv
11	106	45	19	5	0,848	0,7019	0,6342	0,3657	cv
11	115	49	10	1	0,92	0,7012	0,6628	0,3371	cv
12	104	42	21	8	0,832	0,7123	0,64	0,36	cv
12	104	44	21	6	0,832	0,7027	0,6285	0,3714	cv
12	101	44	24	6	0,808	0,6965	0,6114	0,3885	cv
13	103	41	22	9	0,824	0,7152	0,64	0,36	cv
13	100	41	25	9	0,8	0,7092	0,6228	0,3771	cv
13	108	43	17	7	0,864	0,7152	0,6571	0,3428	cv
14	114	46	11	4	0,912	0,7125	0,6742	0,3257	cv
14	111	48	14	2	0,888	0,6981	0,64571	0,3542	cv
14	109	43	14	7	0,8861	0,7171	0,6705	0,3294	cv
15	125	50	0	0	1	0,7142	0,7142	0,2857	cv
15	125	50	0	0	1	0,7142	0,7142	0,2857	cv
15	125	50	0	0	1	0,7142	0,7142	0,2857	cv
16	118	45	7	5	0,944	0,7239	0,7028	0,2971	cv
16	121	45	4	5	0,968	0,7289	0,72	0,28	cv
16	120	44	5	6	0,96	0,7317	0,72	0,28	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
17	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv
18	125	50	0	0	1	0,7142	0,7142	0,2857	cv

Lfdnr. : Laufende Nummer [Zahl größer 0]

TP: True positives [Zahl größer gleich 0]

FP: False positives [Zahl größer gleich 0]

FN: False negatives [Zahl größer gleich 0]

TN: True negatives [Zahl größer gleich 0]

TM: Testmethode - cross validation oder percentage split [cv/ps]

Anhang E - Spam- und Hamnachrichten von Studenten

Spam

1. IMler sind doof.
2. Ich wollte nur mal mitteilen, dass IMler gar nicht so doof sind, wie es häufig den Anschein hat.
3. Ob man den Ticker-Filter überlisten kann? :-)
4. Meine Katze ist entlaufen. Schwarzes Fell, getigert, wild. Bitte ruft mich an: 0190/2324521
5. Prof. F00bar stinkt! Ich bin Gott! Alhambra!
6. Wer das Essen 1 isst, isst nie wieder was. Glaubt mir, ich spreche aus Erfahrung.
7. filou ist ein Saftarsch.
8. Das Essen war heute mal wieder scheisse gewesen.
9. Die Dicke von der Theke ist voll unfreundlich.
10. Der Müller stinkt.
11. Alle die IM studieren sind hohl im Kopp.
12. Das Auto yx-z12394 steht so schlecht, dass muss bestimmt ne GP Maus fahren.
13. Auf www.Ultr0r,de gibt es alle Lösungen zu den GP-Klausuren.
14. Mensa-Menü 1 würde ich heute nicht empfehlen, ich komm nicht mehr vom Klo runter!
15. Noch eine Datenbankenübung, und ich kann für nichts mehr garantieren
16. KO-XX 1234 du Vollidiot. Fahr deine Kiste weg oder sie wird abgeschleppt
17. Prof. Foo ist ein absoluter Tr0tt3l

18. Komm ich jetzt ins Fernsehen?
19. dani rulez!!1!
20. laaaaangweilig....
21. asdf asdf asdf
22. Wer das liest ist doof!
23. Das Essen schmeckt heute wieder unter aller SAU
24. Menü 3 ist nicht absolut ungenießbar!! L
25. So ein Schweinefrass.
26. Wer so kocht hat'se doch nicht mehr alle.
27. Der Juppes hat einen an der Waffel!
28. So schlecht kocht ja nicht mal der Höffer.
29. Ihr seid ja alle total d**f. Ihr könnt mich mal kreuzweise.
30. Who the fock is Alice? Denecke is Alice!
31. Na Seppel du alte Schwuchtel
32. hic, heac, hoc, der Lehrer hat einen Stock, sum, fui, esse, damit kriegt er auf die Fresse.
33. Zum Valentinstag gibt es heute in der Mensa für jedes Weibchen ein Herz-Spiegelei.
34. Skandal, Campusnewsbox fällt Bastelschlampe auf den Kopf, Sie denkt nun sie Studiere GP.
35. Zu lange warten in der Mensaschlange? Sprengt das E/F Gebäude!
36. !"§%&/()=?*''° ¨¬''#£!"§%&/()=?*''° ¨¬''#£!"§%&/()=?*''° ¨¬''#£
37. Seid ihr eigentlich alle zu blöd zum Autofahren? Wieder 5 Parkreihen ihre Vollidioten!
38. Ingo ist ein Arschloch
39. Marcel ist ne Schwuchtel
40. GP-Mäuse mit IQ unter Zimmertemperatur blockieren wieder die halbe Mensa
41. Irgendwo unter den unterbelichteten Gestalten hier an der Uni muss es doch intelligentes Leben geben. Nicht leicht zu finden.

Ham

1. Über den Wolken, scha-la-la, muss die Freiheit wohl Grenzenlos sein...
2. Mein Lob an die Küche!
3. Hat jemand Lust mit mir OOPM zu lernen? (0160-1234567)
4. Trainingspartner/in für Ausdauerschwimmen gesucht! (swim@uni-koblenz.de)
5. Attraktive GP-Maus sucht wohlhabenden Informatiker für gemeinsamen Lebensabend.
6. Dies ist eine Tickernachricht.
7. Morgen Abend Party in der Triererstrasse 99.
8. Ich finde es eine Sauerei, wie manche Leute piarken!
9. Ich grüsse alle die mich kennen!
10. Ich grüsse meine Freundin Gertrude!
11. AIESEC sucht Helfer für eine Veranstaltung zum Thema Auslandsstudium.
12. Der Uni-Ball findet am 12.11.2008 in der Aula statt. Wir laden herzlich alle ein.
13. Suche Lernpartner für Medientechnik bei Herrn Dröge. Bin fleissig und gewillt.
14. Warum stinkt es auf dem Campus wieder so?
15. Huhu, ich möchte alle Leute Grüßen, die heute auch Geburtstag haben.
16. Esst heute um Gottes Willen nicht das vegetarische Menü.
17. Die Lerngruppe zu ST und DB trifft sich heute in d028.
18. Hallo Flo mein schatz, ich liebe dich ganz dolle.
19. Die Vorlesung Ëntwicklungspsychologieffindet heute in Raum E313 statt.
20. Heute abend im Unikino "Being John Malkovich“, Beginn 20 Uhr!
21. Die neue Astarix ist online, www.uni-koblenz.de/ astarix
22. blauer Golf, Kennzeichen KO-XX 1234: Licht brennt noch!
23. Ich wurde eingeparkt! Fahrer des Autos KO-XX 1234, bitte fahr dein Fahrzeug weg.

24. Neben den Top News von Spiegel, Heise usw. jetzt auch die Schlagzeilen der Astarix direkt auf Euer Handy.
25. Am 6.6. findet der Firmenlauf statt. Die Uni will als größte Gruppe starten. Anmelden unter www.uni-läuft-mit.de. Gebühr und T-Shirt wird vor der Firma Campussponsor übernommen.
26. Firmenlauf: Ziel sind 400 Teilnehmer von der Uni. Gemeldet sind bereits 234. Jetzt mitmachen und mitbegeistern.
27. Geht wählen! (Nur noch heute bis 17 Uhr.)
28. Hab meinen pulli verloren. Der ist weiß mit roten punkten. Hat ihn jemand gesehen??!
29. NK-KC 425 steht auf meinem gemieteten Parkplatz!
30. Posterverkauf vor D028 - Schaut vorbei!
31. Spitzen Essen! Weiter so! Ihr seid die Besten.
32. Ich grüße die Süße Blonde vorne rechts an der Käsetheke. J
33. Heute werden im D-Gebäude aktuelle Informationen zur anstehenden Wahl ausgelegt.
34. a, ae, de, sine, cum, pro und prae stehen im Ablativ
35. Falschparker an die Wand: falschparker.uni-koblenz.de
36. AMD steht kurz vor dem Aus, Intel erwägt Kauf und anschließende Schließung.
37. Steve Jobs und Steve Ballmer kommen anläßlich der Koblenz Expo an die Uni.
38. Die TUS verliert gegen Alemania Aachen mit 0:10 auf dem Tivoli. Trainer vor dem Aus?
39. Habe eine Schwarze Geldbörse mit Diddelmaussticker gefunden. Es liegt in B214 zur Abholung bereit.
40. Nachhilfe von Studenten für Studenten: Kurs zur Wiederholung von Mathe A findet am 23.7.08 in G210 Statt.
41. Kevin + Simone = Liebe
42. Mac-User Treffen findet heute nicht statt, sondern wurde auf kommende Woche Mittwoch verschoben.
43. Hallo ihr Studenten, tolles Wetter heute! Kommt heute Abend zu meiner Grillparty

44. Moin zusammen, hab meinen rosa Lieblingsfüller verloren, wer ihn hat bitte Mail an schnuffeltuch@uni-koblenz.de
45. Heute Abend Fussball schauen im Stubi! Wer ist dabei?
46. Sonderkurse von Studenten für Studenten, zum Thema - Miteinander in der Mensa - Jacken statt Badetücher!
47. Ich liebe meinen Schatz Knuffi! *herzchen* Tina
48. Das Auto mit dem Kennzeichen ww-ak-221 wird gerade abgeschleppt
49. Verkaufe meine Ü-Eier Sammlung - Infos unter 0160-8882221
50. Das Auto mit dem Kennzeichen MYK-XX-0815 hat noch Licht an.

Anhang F - Klassendiagramme

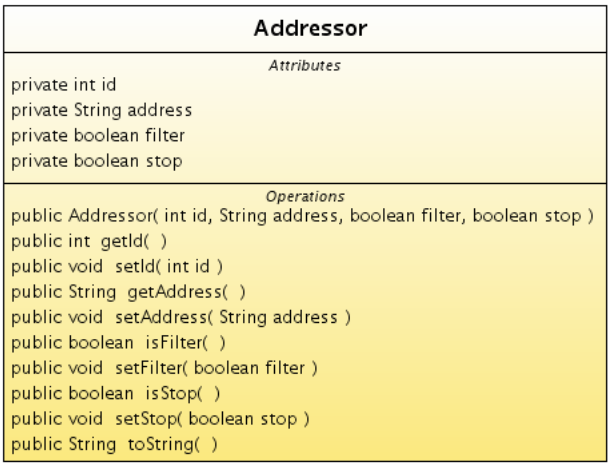

Abbildung .1: Die Klasse 'Addressor'

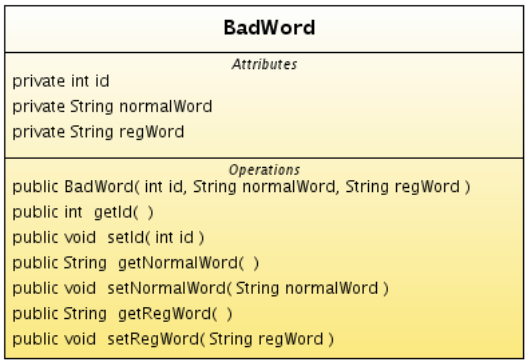

Abbildung .2: Die Klasse 'BadWord'

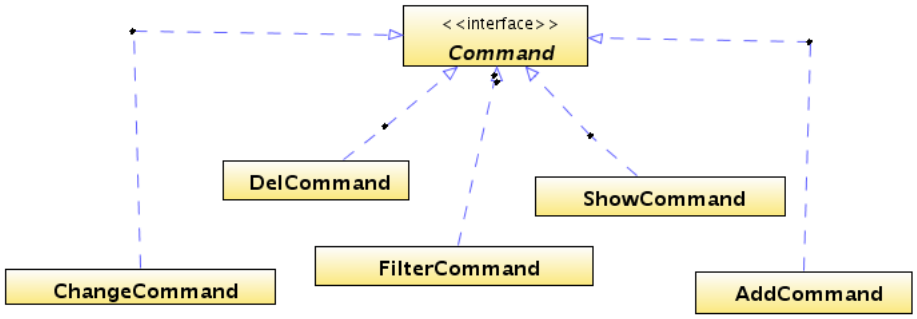

Abbildung .3: Klassendiagramm - Überblick über die 'command' Klassen und das Interface 'Command'

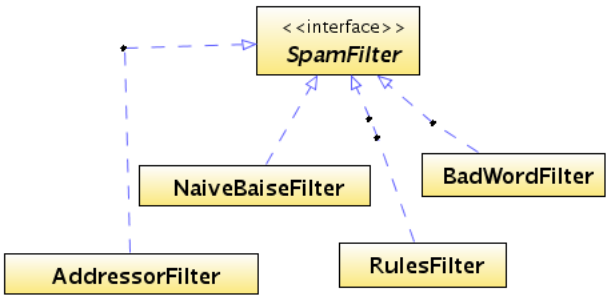

Abbildung .4: Klassendiagramm - Überblick über die 'filter' Klassen und das Interface 'SpamFilter'

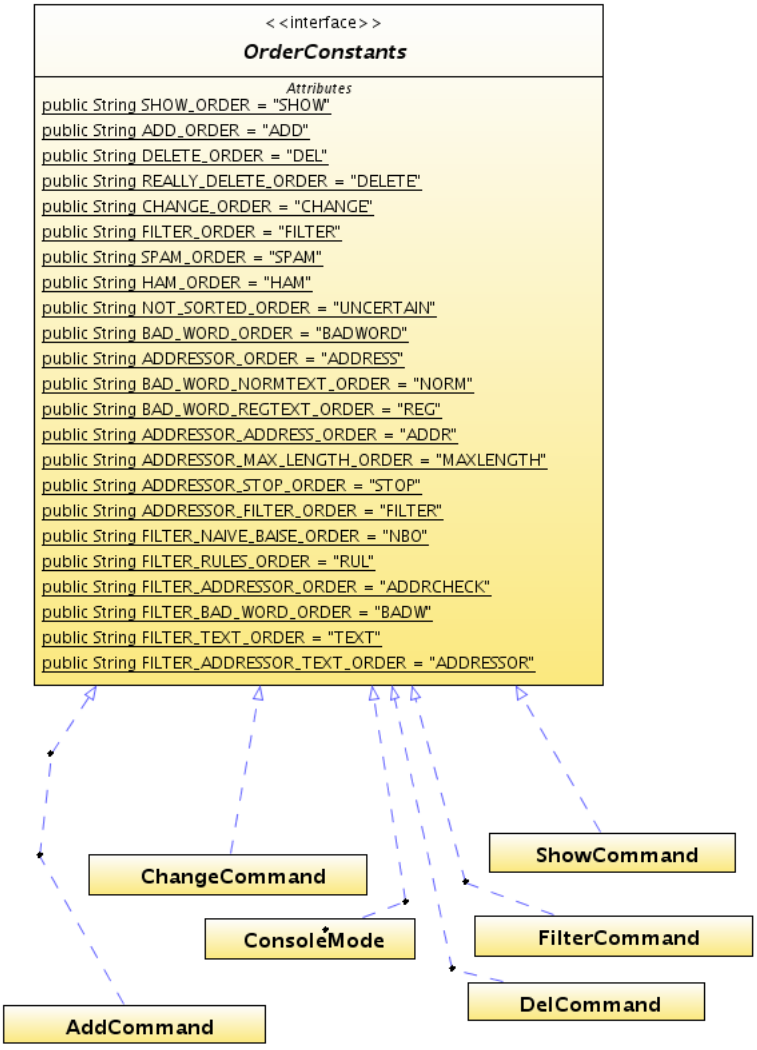

Abbildung .5: Die Befehlskonstanten

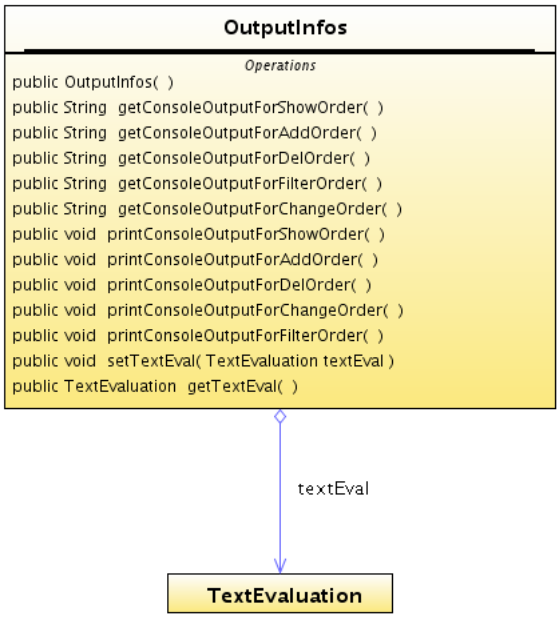

Abbildung .6: Die Klasse 'OutputInfos'

Abbildung .7: Die Klassen 'StringUtils' und 'regExMaker'

Literaturverzeichnis

[1] Weka arff-datei, 2008. http://weka.sourceforge.net/wekadoc/index.php/en:ARFF_%283.4.11%29.

[2] Weka software, 2008. http://www.cs.waikato.ac.nz/ml/weka/.

[3] Kai Blankenhorn. Spam-filterung mittels maschinellem lernen. Master's thesis, Fachhochschule Furtwangen, 2002.

[4] J. Cleve. Entscheidungsbaumlernen, 2008. http://www.wi.hs-wismar.de/ cleve/vorl/dm2160207807/skript/node37.html - [Online; Stand 27. Januar 2008].

[5] Paul Graham. A plan for spam, 2002. http://www.paulgraham.com/spam.html - [Online; Stand 27. Januar 2008].

[6] Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. Gb - Mit Press, 2003.

[7] Markus Maron, Kevin Read, and Michael Schulze. Campus news - artifical intelligence methods combined for an intelligent information network. In *Ambient Intelligence*, 2007.

[8] Matthias Mensch. Die entropie natürlicher sprache. Master's thesis, Universität Koblenz, 2008.

[9] G. Robinson. Spam detection, 2008. http://radio.weblogs.com/0101454/stories/2002/09/16/spamDetection.html - [Online; Stand 27. Januar 2008].

[10] Stuart Russell and Peter Norvig. *Artificial Intelligence, A Modern Approach*. Pearson Education, 2003.

[11] SUN, 2008. http://java.sun.com/j2se/1.4.2/docs/api/ - [Online; Stand 27. Januar 2008].

[12] Jochen Topf, Matthias Etrich, Joerg Heidrich, Leslie Romeo, Marco Thorbrügge, and Bert Ungerer. *Anti - Spam, Unerwünschte E-Mails erkennen und abwehren*. Bundesamt für Sicherheit in der Informationstechnik, 2005.

[13] Wikipedia. Liste der unicode-zeichen — wikipedia, die freie enzyklopädie, 2008. http://de.wikipedia.org/w/index.php?title=Liste_der_Unicode-Zeichenoldid=41450137 - [Online; Stand 27. Januar 2008].