

TEKSTMINING

VAN TEKST NAAR KLANTINTELLIGENTIE

STAGEVERSLAG




AZIZ MOHAMMADI

TEKSTMINING

VAN TEKST NAAR KLANTINTELLIGENTIE

AZIZ MOHAMMADI

STAGEVERSLAG

Vrije Universiteit Amsterdam 
Faculteit der Exacte Wetenschappen
Master Business Mathematics and Informatics
De Boelelaan 1081a
1081 HV Amsterdam

Stagebedrijf:
ING Nederland
Customer Intelligence
Acanthus B.02.154
Bijlmerdreef 24
1102 AT Amsterdam

ING 

Oktober 2008

VOORWOORD

Dit stageverslag vormt de afsluiting van mijn studie Bedrijfswiskunde en Informatica aan de Vrije Universiteit te Amsterdam. Ik heb, gedurende een periode van 6 maanden, stage gelopen op de afdeling Customer Intelligence van ING Nederland. Tijdens deze uiterst leerzame periode heb ik mij bezig gehouden met het beantwoorden van de vraag of collecties van antwoorden, die verkregen zijn middels open vragen in tevredenheidsonderzoeken, geanalyseerd kunnen worden met behulp van tekstminingtechnieken.

Mijn dank gaat uit naar de verschillende personen die hebben bijgedragen aan het succesvolle verloop van mijn stage. Ten eerste wil ik Maarten Terpstra bedanken voor zijn begeleiding vanuit ING. Maarten heeft mij de vrijheid gegeven om op geheel eigen wijze de stageopdracht uit te voeren. Hij was altijd zeer enthousiast wanneer ik weer eens met veelbelovende resultaten kwam aanzetten. Dit heeft er voor gezorgd dat ik altijd met een grote voldoening aan mijn stageopdracht heb gewerkt. Voorts gaat mijn dank uit naar Wojtek Kowalczyk en Marianne Jonker, mijn stagebegeleiders vanuit de Vrije Universiteit. Wojtek Kowalczyk was altijd bereid om tijd vrij te maken voor het bieden van hulp en begeleiding. Hij stond altijd klaar om met mij mee te denken wanneer ik een vraag had, of ergens niet uit kwam. Mijn dank hiervoor. Marianne Jonker wil ik bedanken voor het lezen van het stageverslag en het bieden van suggesties voor verbeterpunten. Daarnaast wil ik de overige collega's van Customer Intelligence bedanken voor de gezellige sfeer en de leuke lunchpauzes. Dankzij jullie heb ik mij vanaf het begin thuis gevoeld op de afdeling.

At last, but not least, wil ik mijn ouders en mijn familie bedanken voor hun steun. Zonder hen zou ik nooit in staat zijn geweest om zover te komen.

Aziz Mohammadi

Amsterdam, oktober 2008

SAMENVATTING

VERTROUWELIJK

INHOUDSOPGAVE

Voorwoord	i
Samenvatting	iii
1 Inleiding	7
1.1 ING, Customer Intelligence en Onderzoek & Advies	7
1.1.1 ING	7
1.1.2 Customer Intelligence	7
1.1.3 Onderzoek & Advies	9
1.2 Achtergrond van de stage	9
1.3 Probleemstelling	9
1.4 Doelstelling	9
1.4.1 Toegevoegde waarde van het onderzoek voor CI	9
1.4.2 Doel van het onderzoek	9
1.5 Opbouw rapport	10
2 Tekstmining	11
2.1 Inleiding	11
2.2 Definitie van tekstmining	11
2.3 Tekstmining taken	13
2.3.1 Het classificeren van tekstdocumenten	13
2.3.2 Het clusteren van tekstdocumenten	13
2.3.3 Het samenvatten van tekst	14
2.3.4 Trefwoordextractie	15
2.3.5 Taalherkenning	15
2.4 Gerelateerde onderzoeksgebieden	16
2.4.1 Information Retrieval (IR)	16
2.4.2 Natural Language Processing (NLP)	16
2.4.3 Information Extraction (IE)	16
2.5 Toepassingen van tekstmining	17
2.5.1 Bioinformatica	17
2.5.2 Het filteren van spam	17
2.5.3 Business Intelligence	18
2.5.4 Terrorisme	18
2.5.5 Patentanalyse	19
3 Het tekstmining proces	21
3.1 Inleiding	21
3.2 Tekstdocumenten voorbehandelen	21
3.2.1 Het omzetten van het document in ASCII formaat	21
3.2.2 Het verwijderen van tekens	22
3.2.3 Filtering en stemming van woorden	22
3.3 Vectorruimte model	22
3.4 TF-IDF weegschema	23
3.5 Featureselectie	24
3.5.1 Featureselectie technieken	25
3.6 Feature transformatie	27
3.6.1 Principal Components Analyse (PCA)	27
3.6.2 Singuliere Waarden Decompositie	29
3.6.3 Latent Semantic Indexing (LSI)	29

4 Automatische tekstclassificatie	33
4.1 Inleiding	33
4.2 Definitie van tekstclassificatie	33
4.3 Classificatietechnieken.....	34
4.3.1 Naïve Bayes.....	34
4.3.2 k-Nearest Neighbor.....	36
4.3.3 Support Vector Machines	38
4.3.4 BoosTexter	39
4.4 Evaluatie van de prestaties van de classificatietechnieken	42
4.4.1 k-fold cross-validation en leave-one-out cross-validation.....	42
4.4.2 Prestatiematen.....	43
4.4.3 Onderlinge vergelijking van de classificatietechnieken	44
5 Het clusteren van tekst	45
5.1 Inleiding	45
5.2 Het algemene clusteringsprobleem	45
5.2.1 Probleemrepresentatie	45
5.2.2 Gelijkenismaten.....	46
5.3 Clusteringalgoritmen.....	47
5.3.1 K-means algoritme	48
5.3.2 Expectation Maximization Clusteringalgoritmen.....	49
5.3.3 Hierarchische Agglomeratieve Clustering (HAC).....	49
5.3.4 Overige clusteringalgoritmen	50
5.4 Evaluatie van de clusters.....	51
5.4.1 Indices.....	51
5.4.2 Vergelijkende maten.....	52
5.5 Het clusteren van tekstdocumenten.....	53
6 Onderzoeksopzet	55
7 Classificatie-experimenten collectie administratie	57
8 Classificatie-experimenten collectie telefoon	59
9 Experimenteren met clusteren	61
Conclusies	63
Aanbevelingen	65
Literatuurlijst	67
A Datasets	71
B XML proces configuratie files in Rapidminer	73

HOOFDSTUK 1

INLEIDING

1.1 ING, CUSTOMER INTELLIGENCE EN ONDERZOEK & ADVIES¹

1.1.1 ING

ING is een financiële dienstverlener die van oorsprong Nederlands is en die in ruim vijftig landen actief is verspreid over Europa, de Verenigde Staten, Canada, Latijns-Amerika, Azië en Australië. ING is in 1991 ontstaan uit een fusie tussen Nationale-Nederlanden en de NMB Postbank Groep. De naam Internationale Nederlanden Groep werd al snel afgekort tot ING. Wereldwijd werken er ongeveer 125.000 medewerkers bij ING die zich bezig houden met vier soorten diensten: bankieren, beleggen, levensverzekeringen en pensioenen. Deze medewerkers bedienen ruim 75 miljoen klanten.

In 2007 was volgens Forbes ING naar grootte het 9^e bedrijf ter wereld. De nettowinst bedroeg in dit jaar €9.241 miljoen en het beheerd vermogen was gelijk aan €637 miljard. Volgens de Interbrand-top 100, een lijst van de 100 grootste merken ter wereld, staat ING op de 81^{ste} plaats. Titelsponsor zijn van het ING Renault Formule 1-team heeft hier zeker aan bijgedragen.

Binnen Nederland werken er ongeveer 33.000 medewerkers voor bekende merken als Nationale-Nederlanden, ING Bank, Postbank en WestlandUtrecht.

1.1.2 CUSTOMER INTELLIGENCE

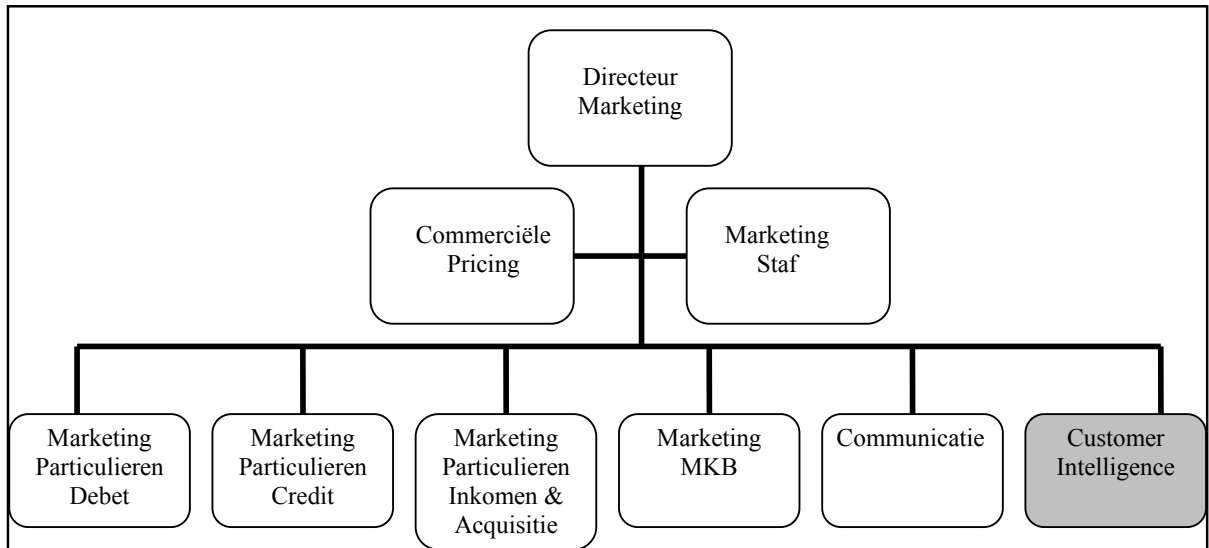
Customer Intelligence (CI) valt binnen ING Retail NL onder Marketing (zie figuur 1.1). CI stelt zich ten doel de marktwerking van ING Retail NL optimaal te ondersteunen door de kennis over de particuliere en kleinzakelijke markten en klanten te combineren met hoogwaardige expertise op het gebied van database marketing. Op basis van deze kennis adviseert CI marketing op verschillende marketing, sales en credit risk gebieden. Daarnaast is CI de expert in het leveren van klant- en markt kennisdiensten en het managen en het optimaliseren van de klantcontactstrategie.

De belangrijkste taken van CI zijn:

- Het beheren en “verrijken” van informatie alsmede het beschikbaar stellen van klant- en marktkennis uit interne en externe bronnen;
- Ontwikkeling van een Customer Relationship Manager (CRM) visie- en strategie;
- Het verbeteren van CRM processen- en systemen;
- Het uitvoeren van database onderzoek en het begeleiden van extern marktonderzoek naar consumenten, bedrijven en markten;
- Het beheren en exploiteren van datawarehouses en datamarts;
- Het bieden van specialistische ondersteuning van de uitvoering van campagnes en marketingacties;
- Het analyseren, adviseren en beheren van klantregieregels binnen Retail NL (multichannel consultancy);
- Het opbouwen, ontsluiten en beschikbaar stellen van kennis en het rapporteren over concurrentie analyse en trends (kennismanagement);
- Komen met verbetervoorstellen – en prioriteiten op grond van bijvoorbeeld klanttevredenheid;

¹ De inhoud van deze paragraaf is gebaseerd op het interne rapport: Vervolgadviesaanvraag Organisatorische inrichting van ING Retail, Marketing, december 2007.

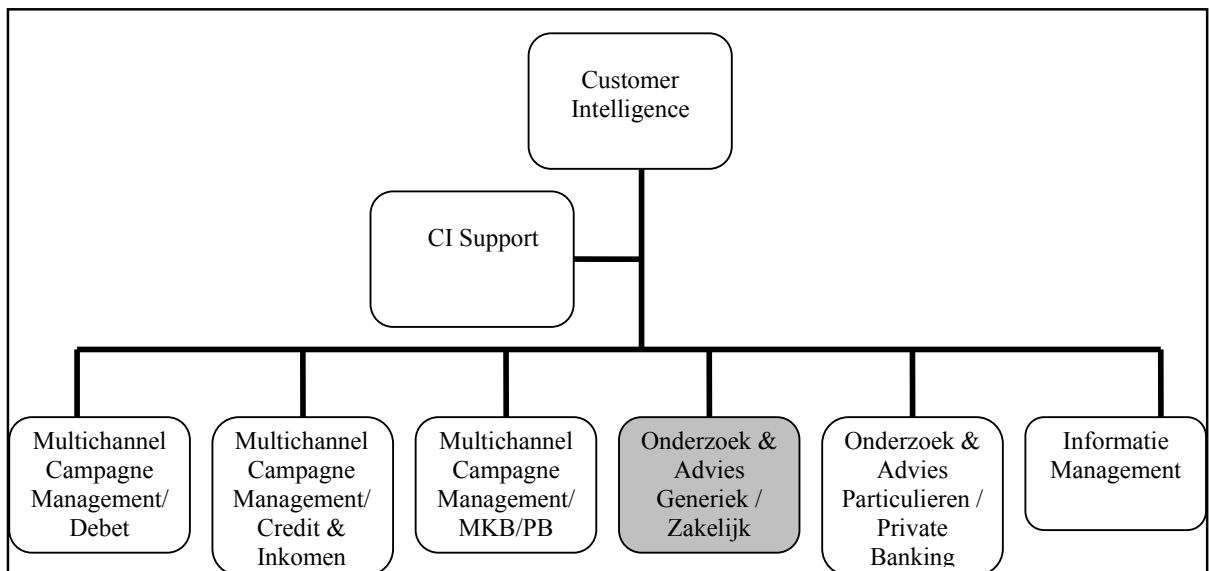
- Het initiëren en uitvoeren van fundamenteel lang-cyclisch onderzoek naar (drijvers van) gedrag van consumenten en bedrijven en het aan de hand van de resultaten van dit onderzoek adviseren inzake aanpassing van de marketing P's.
- Het in opdracht ontwikkelen van credit risk gerelateerde instrumenten (acceptatie score cards, revisie- en overstandsbeheermodellen en Basel-II rapportagemodellen).



Figuur 1.1: Organogram van de afdeling marketing.

CI is zelf weer onderverdeeld in zeven verschillende afdelingen (zie figuur 1.2):

- Stafafdeling CI support;
- Multichannel Campagne Management Debet;
- Multichannel Campagne Management Credit & Inkomen;
- Multichannel Campagne Management MKB/P;
- Onderzoek & Advies Generiek / Zakelijk
- Onderzoek & Advies Particulieren/Private Banking
- Informatiemanagement.



Figuur 1.2: Organogram van de afdeling Customer Intelligence.

1.1.3 ONDERZOEK & ADVIES

De stageopdracht is uitgevoerd in opdracht van de afdeling Onderzoek en Advies. Deze afdeling is verantwoordelijk voor het initiëren, adviseren en uitvoeren van onderzoek naar consumenten, bedrijven en markten (domeinen: klanten, productafname, kanalen, kosten, rendement, risico, concurrentie en geografische markten). Hierbij wordt onderscheid gemaakt tussen extern onderzoek en intern database onderzoek.

De afdeling Onderzoek & Advies / Generiek Zakelijk is tevens verantwoordelijk voor:

- Het initiëren, ontwikkelen van draagvlak en het uitvoeren van fundamenteel onderzoek naar de (drijvers van) klantgedrag in de financiële dienstverlening en het op basis hiervan uitbrengen van adviezen;
- Aanbevelingen doen over de verbeterprioriteiten naar aanleiding van klanttevredenheids-onderzoek en overige relevante informatie met betrekking tot klanttevredenheid;

1.2 ACHTERGROND VAN DE STAGE

VERTROUWELIJK

1.3 PROBLEEMSTELLING

VERTROUWELIJK

1.4 DOELSTELLING

1.4.1 TOEGEVOEGDE WAARDE VAN HET ONDERZOEK VOOR CI

VERTROUWELIJK

1.4.2 DOEL VAN HET ONDERZOEK

Het hoofddoel van het onderzoek is vaststellen of de antwoorden die gegeven zijn op de open vragen met behulp van tekstmining automatisch geanalyseerd kunnen worden, zodanig dat er inzicht verkregen kan worden in de achterliggende oorzaken van de problemen die de klanten hebben. Op basis van dit onderzoek worden er aanbevelingen gedaan over de bruikbaarheid van de onderzoeksmethode voor regulier tevredenheidsonderzoek en wordt er ingegaan op het nut van tekstmining in andere toepassingen binnen CI en ING.

Het onderzoeksrapport dient ook een ander doel. Het vormt namelijk een inleiding tot het vakgebied van tekstmining, de technieken die hierin centraal staan en de mogelijkheden die deze technieken bieden. In het rapport wordt stap voor stap, op een duidelijke manier, aangegeven hoe men tekstminingstechnieken kan toepassen op collecties van tekstdocumenten². Dit rapport stelt de afdeling Customer Intelligence in staat om in de toekomst zelfstandig tekstmininganalyses uit te voeren op collecties van tekstdocumenten.

² Een verzameling antwoorden is niets anders dan een collectie van korte tekstdocumenten. Elk antwoord kan namelijk opgeslagen worden als een apart tekstdocument.

1.5 OPBOUW RAPPORT

Het vervolg van dit rapport is op de volgende manier opgebouwd. Hoofdstuk 2 t/m 5 vormen het theoriegedeelte van dit rapport en zijn het resultaat van een zeer uitgebreide literatuurstudie. Dit gedeelte dient als referentiekader voor het praktijkgedeelte van dit rapport. In hoofdstuk 2 wordt ten eerste het vakgebied van tekstmining uitgebreid behandeld. Hierbij wordt er onder andere ingegaan op de vraag wat er onder tekstmining wordt verstaan en in welke toepassingen het een voorname rol speelt. Vervolgens wordt in hoofdstuk 3 uitgebreid stil gestaan bij het tekstmining proces en de stappen waaruit dit proces is opgebouwd. Daaropvolgend worden in hoofdstuk 4 de classificatietechnieken die gebruikt zijn in het onderzoek uitvoerig besproken. In hoofdstuk 5 komen ten slotte de clusteringtechnieken aan bod.

Hoofdstuk 6 t/m 9 vormen samen het praktijkgedeelte van dit rapport. In hoofdstuk 6 wordt de opzet van het onderzoek behandeld evenals de gebruikte dataset en de gehanteerde programmatuur. Vervolgens worden in hoofdstuk 7 en 8 de uitkomsten van de classificatieexperimenten en analyses besproken die uitgevoerd zijn op twee verschillende collecties van toelichtingen. In hoofdstuk 9 worden de resultaten van de uitgevoerde clusteringexperimenten behandeld. De conclusies komen vervolgens aan bod in hoofdstuk 10 en in hoofdstuk 11 worden ten slotte de aanbevelingen gerapporteerd.

HOOFDSTUK 2

TEKSTMINING

2.1 INLEIDING

Tegenwoordig leven wij in een informatietijdperk waarin de digitale informatie die wij elke dag moeten verwerken met een steeds grotere snelheid toeneemt. Elke dag worden wij geconfronteerd met tientallen e-mails, documenten en online (nieuws)berichten. Ook de hoeveelheid informatie die beschikbaar is op het Web neemt exponentieel toe en verdubbelt zich om de 8 maanden [6]. Het aantal online documenten werd een paar jaar geleden al geschat op 550 miljard, wat overeenkomt met 7.5 petabyte aan data die beschikbaar is op websites en in publieke databanken³ [4]. Enkele belangrijke factoren die deze ontwikkeling hebben mogelijk gemaakt zijn: de opkomst van de PC en tekstverwerkingsprogrammatuur, scanners, tekstherkenningstechnologie, het internet en steeds goedkoper wordende dataopslag.

Alhoewel wij steeds meer data tot onze beschikking hebben blijft ons vermogen om deze data op te nemen en te verwerken constant. Een persoon die 1 pagina per minuut leest heeft wel 5.7 miljoen jaar nodig om alle informatie die beschikbaar is op het web te lezen [19]. Zoekmachines vergroten dit probleem door een grote hoeveelheid aan informatie middels enkele toetsaanslagen beschikbaar te maken. Men is daarom op zoek naar geavanceerde technieken waarmee de grote hoeveelheden aan data automatisch verwerkt en toegankelijk gemaakt kunnen worden.

Tekstmining is een nieuw en uitdagend multidisciplinair onderzoeksgebied dat het bovenstaande probleem, ook wel “information overload” genoemd, tracht op te lossen door verschillende technieken, die ondermeer afkomstig zijn uit de vakgebieden Information Retrieval (IR), Natural Language Processing (NLP) en Information Extraction (IE), te combineren. In dit hoofdstuk zullen wij het gebied van tekstmining nader beschrijven. In de volgende paragraaf zal als eerst aangegeven worden wat er in de literatuur onder tekstmining wordt verstaan. Hier opvolgend zullen de vakgebieden waaraan tekstmining zijn technieken ontleent kort behandeld worden, waarna er in de laatste paragraaf enkele succesvolle toepassingen uit verschillende vakgebieden beschreven zullen worden waarin tekstmining een prominente rol speelt.

2.2 DEFINITIE VAN TEKSTMINING

Informeel wordt onder tekstmining, ook wel het ontdekken van kennis in tekst genoemd, het automatisch en intelligent analyseren van teksten door een computer verstaan, met als doel het vinden van nieuwe en interessante feiten, relaties en kennis in grote hoeveelheden tekst, die aangewend kunnen worden om een bepaald doel te realiseren [19]. Tekstmining is een zeer breed vakgebied dat op verschillende manieren omschreven kan worden. Zo beschrijven Nahm en Mooney in [27] tekstmining als “zoeken naar patronen in ongestructureerde tekst”. In [8] geven Doore en anderen aan dat tekstmining “analytische functies toepast, die afkomstig zijn uit het vakgebied datamining, op informatie die middels ingewikkelde analysetechnieken wordt gedestilleerd uit ongestructureerde tekstdocumenten”. Tan beschrijft op zijn beurt tekstmining in [37] als “het proces waarbij interessante en niet-triviale patronen of kennis worden geëxtraheerd uit tekstdocumenten”.

³ Een petabyte (10^{15} bytes) komt overeen met 1 miljoen gigabytes.

Volgens Hotho en anderen in [15] is de definitie van tekstmining afhankelijk van het onderzoeksgebied waarin het wordt toegepast. Zij geven drie verschillende definities van tekstmining:

1. **Tekstmining = Het extraheren van informatie.** In deze definitie wordt aangenomen dat tekstmining simpelweg het extraheren van informatie of feiten uit tekst behelst. Een voorbeeld hiervan is een programma dat cv's inleest en gegevens zoals persoonsnaam, adres, werkervaring en dergelijke, onttrekt aan deze documenten.

Hearst en Sehgal stellen echter in [13 en 34] dat het extraheren van informatie uit tekst geen "echte" tekstmining is. Volgens Hearst en Sehgal kan het doel van "echte" tekstmining omschreven worden als het ontdekken van nieuwe en tot dan toe nog onbekende relaties en verbanden in tekst. Information Extraction voldoet in hun ogen niet aan deze definitie omdat er geen nieuwe verbanden worden ontdekt, maar er simpelweg feiten worden onttrokken over voorgespecificeerde entiteiten of gebeurtenissen die reeds aanwezig zijn in de tekst. Wel onderkennen Hearst en Sehgal dat het extraheren van informatie een belangrijke stap kan zijn in het tekstminingproces.

2. **Tekstmining = Tekstuele datamining.** Tekstmining kan ook gedefinieerd worden als het toepassen van algoritmen en methoden afkomstig uit de vakgebieden van machine learning en statistiek op tekst met als doel het vinden van bruikbare patronen.

Veel mensen verwarren tekstuele datamining met gewone datamining. Het grote verschil tussen deze twee is dat het doel van datamining omschreven kan worden als het onttrekken van verborgen, tot dan toe niet bekende en bruikbare informatie uit data. In het geval van tekstuele datamining is de informatie die onttrokken dient te worden echter niet verborgen, maar is deze duidelijk en expliciet gespecificeerd in de tekst. Een gebrek aan tijd maakt het voor de meeste mensen echter praktisch onmogelijk om alle tekst zelf te lezen waardoor de informatie die zich in de tekst bevindt onbekend blijft. Het uiteindelijke doel van tekstuele datamining is dan ook om de tekst in een bruikbare vorm te gieten die drukbezette personen een helpende hand kan bieden bij het verwerken van alle tekst of door computers nader geanalyseerd kan worden [39].

3. **Tekstmining = Knowledge Discovery Process.** Volgens Fayyad [10] kan Knowledge Discovery in Databases als volgt gedefinieerd worden: "Knowledge Discovery in Databases (KDD) is het niet-triviale proces dat geldige, nieuwe, potentieel bruikbare en begrijpbare patronen in data identificeert". Hearst definieert in [14] tekstmining als "het ontdekken van nieuwe of tot dan toe niet bekende informatie door het automatisch extraheren van informatie uit geschreven bronnen met behulp van de computer". Stel dat een document bijvoorbeeld een relatie vaststelt tussen onderwerp A en B en een ander document stelt een relatie vast tussen onderwerp B en C. Indien deze twee documenten worden gecombineerd kan wellicht een nieuwe relatie vastgesteld worden tussen document A en C die nog niet bekend was.

De overeenkomst tussen de definitie van Hearst en de definitie van Fayyad is dat de nadruk specifiek gelegd wordt op het ontdekken van nieuwe informatie. Deze nadruk ontbreekt in de definities die onder punt 1 en 2 zijn vermeld.

In dit rapport beschouwen wij tekstmining voornamelijk als tekstuele datamining. Onze focus ligt voornamelijk op methoden en technieken die gebruikt kunnen worden om bruikbare informatie uit tekst te extraheren met als doel het classificeren of clusteren van verschillende tekstdocumenten in verschillende (voorgedefinieerde) groepen.

2.3 TEKSTMINING TAKEN

2.3.1 HET CLASSIFICEREN VAN TEKSTDOCUMENTEN

Het classificeren of categoriseren van objecten is een thema dat vaak terugkomt wanneer men complexe data analyseert. Het doel is om een gegeven dataobject te classificeren in een op voorhand opgestelde verzameling van categorieën. In het geval dat het gaat om collecties van tekstdocumenten wordt dit ook wel tekstcategorisatie genoemd: het proces waarbij de correcte categorie (of categorieën) waartoe elk tekstdocument behoort gevonden wordt, gegeven een verzameling van categorieën (onderwerpen) en een collectie van tekstdocumenten [31].

In de jaren 60 is men begonnen met de studie van automatische tekst categorisatie. Destijds was het doel om de wetenschappelijke literatuur te indexeren met behulp van een beperkt vocabularium. Pas in de jaren '90 ontwikkelde dit vakgebied zich volledig door de steeds toenemende beschikbare hoeveelheid van tekstdocumenten in digitale vorm en de behoefte om deze documenten te organiseren voor makkelijker gebruik. Tegenwoordig wordt tekstcategorisatie in verschillende toepassingen gebruikt. De meest bekende toepassingen worden in paragraaf 2.5 beschreven.

Er zijn twee benaderingen van tekstcategorisatie [31]. De eerste is de zogenaamde “knowledge engineering” benadering waarbij een expert zijn kennis over de categorieën direct codeert in het systeem, meestal gebruik makend van classificatieregels. De andere benadering is de “Machine Learning” benadering waarbij een classifier wordt getraind op basis van een verzameling voorgeclassificeerde tekstdocumenten. In de context van tekstcategorisatie presteert het knowledge engineering systeem meestal beter dan machine learning systemen alhoewel het verschil tussen deze twee steeds kleiner wordt. Het nadeel van de knowledge engineering benadering is de zogenaamde “knowledge acquisition bottleneck”: de grote hoeveelheid aan gespecialiseerde werknemers en kennis die nodig is om de classificatieregels op te stellen en te onderhouden. Om deze reden is het meest recente werk dat tekstcategorisatie betreft geconcentreerd op de machine learning benadering, waarvoor alleen een verzameling van voorgeclassificeerde tekstdocumenten benodigd is die veel goedkoper te produceren is.

2.3.2 HET CLUSTEREN VAN TEKSTDOCUMENTEN

Tekst (of document) clustering maakt deel uit van het grotere vakgebied dat dataclustering is genaamd. Het doel van documentclustering is het ontdekken van natuurlijke groepen (clusters) waarin de documenten geïnclassificeerd kunnen worden. De clusters geven hierbij een overzicht van het aantal categorieën (onderwerpen) waaruit de documentcollectie bestaat. Een cluster kan gezien worden als een verzameling van documenten die veel op elkaar lijken. Indien de tekstdocumenten worden gerepresenteerd als featurevectoren die worden weergegeven in een meerdimensionale vectorruimte, dan kan een cluster ook gezien worden als een verzameling van featurevectoren die dicht bij elkaar liggen⁴. Voor deze clusters geldt dat de afstand tussen twee featurevectoren die deel uitmaken van hetzelfde cluster kleiner is dan afstand tussen elke featurevector in het betreffende cluster en een featurevector uit een ander cluster.

In tabel 2.1 is als voorbeeld een collectie van nieuwsartikelen weergegeven⁵. Deze collectie kan gegroepeerd worden aan de hand van de onderwerpen van de artikelen. Elk artikel is weergegeven als een verzameling van woord-frequentie paren (w,c) , waar w een woord voorstelt en c het aantal keer

⁴ Featurevectoren en de manier waarop tekst wordt omgezet in deze vectoren komen uitgebreid aan bod in paragraaf 3.2.

⁵ Dit voorbeeld is ontleend aan: Tan, P., M. Steinbach en V. Kumar (2006), *Introduction To Data Mining*. Addison-Wesley, Pearson Education Inc.

aangeeft dat het betreffende woord in het artikel voorkomt. Er komen twee natuurlijke clusters voor in de documentcollectie. De eerste cluster bestaat uit de eerste vier artikelen die nieuws over de economie bevatten. De tweede cluster bestaat uit de laatste vier artikelen die nieuws over de gezondheidszorg bevatten. Een goed clusteringalgoritme moet, op basis van de overeenkomsten tussen de verschillende woorden die voorkomen in de artikelen, in staat zijn om de twee clusters te identificeren.

Tabel 2.1: Collectie van nieuwsartikelen.

Artikel	Woorden
1	dollar: 1, industry: 4, country: 2, loan: 3, deal: 2, government: 2
2	machinery: 2, labor: 3, market: 4, industry: 2, work: 3, country: 1
3	job: 5, inflation: 3, rise: 2, jobless: 2, market: 3, country: 2, index: 3
4	domestic: 3, forecast: 2, gain: 1, market: 2, scale: 3, price: 2
5	patient: 4, symptom: 2, drug: 3, health: 2, clinic: 2, doctor: 2
6	pharmaceutical: 2, company: 3, drug: 2, vaccine: 1, flu: 3
7	death: 2, cancer: 4, drug: 3, public: 4, health: 3, director: 2
8	medical: 2, cost: 3, increase: 2, patient: 2, health: 3, care: 1

Documentclustering dient niet verward te worden met documentclassificatie. In een classificatieprobleem is het aantal categorieën (en hun eigenschappen) op voorhand bekend en worden de tekstdocumenten aan deze categorieën toegekend. In een clusteringprobleem zijn zowel het aantal categorieën als de eigenschappen van deze categorieën op voorhand niet bekend. Ook de wijze waarop de tekstdocumenten zijn verspreid over de gevonden categorieën is niet bekend. Clustering wordt om deze reden ook wel ongesuperviseerd leren genoemd, terwijl documentclassificatie een vorm van supergeviserd leren is. In hoofdstuk 4 en 5 zullen beide vormen van leren uitgebreid beschreven worden.

2.3.3 HET SAMENVATTEN VAN TEKST

Met het intreden van het informatietijdperk worden er dagelijks duizenden documenten in digitale vorm verspreid op het internet. Om effectief gebruik te maken van deze online-documenten is het cruciaal om snel de essentie van deze documenten te achterhalen. Een samenvatting is hier uitermate geschikt voor.

Om een goede samenvatting te genereren dienen de meest belangrijke stukken informatie in het tekstdocument geïdentificeerd te worden, de irrelevante stukken informatie verwijderd te worden, de details geminimaliseerd te worden en dient het resulterende geheel in een compacte vorm gerapporteerd te worden. Hiervoor worden samenvattingsalgoritmes gebruikt die verschillende principes hanteren. In veel gevallen worden complete zinnen uit het tekstdocument geselecteerd en aan elkaar geplakt. De selectie van deze zinnen wordt bijvoorbeeld bepaald door de aanwezigheid van nieuwe informatie. Signalen daarvoor zijn bijvoorbeeld onbepaalde lidwoorden en volledige betitelingen van personen of bedrijven. Structuursignalen als “vervolgens” en “tot slot” markeren eveneens geschikte kernzinnen. Daarnaast worden ook zinnen geselecteerd die frequente woorden en frasen bevatten.

Naast kernzinnen kunnen er ook kernparagrafen geïdentificeerd worden. Het gaat hier meestal om de eerste alinea en de laatste paar alinea's van een document. Er bestaat een duidelijke “trade-off” tussen samenvatten op basis van alinea's en het samenvatten op basis van individuele zinnen. Enerzijds is een complete alinea coherenter dan een aaneenrijging van afzonderlijk geselecteerde zinnen. Anderzijds zal de samenvatting hierdoor ook minder compact zijn dan gewenst omdat de complete alinea's vaak ook overbodig materiaal bevatten.

Er zijn inmiddels enkele succesvolle systemen ontwikkeld in bepaalde vakgebieden die in staat zijn om een goede samenvatting te produceren. Het gaat hier bijvoorbeeld om het genereren van samenvattingen van weer, financiële en medische databases. Een nadeel is dat de meeste samenvattingssoftware met name voor de Engelse taal wordt ontwikkeld. Open source software die Nederlandse teksten kan samenvatten is zeer schaars.

2.3.4 TREFWOORDEXTRACTIE

Trefwoorden zijn significante woorden in een tekstdocument die een beschrijving geven van de inhoud. Trefwoorden worden in verschillende toepassingen gebruikt. Zo worden trefwoorden bijvoorbeeld in wetenschappelijke artikelen gebruikt om inzicht te geven in de onderwerpen die centraal staan. Daarnaast worden zij ook gebruikt door zoekmachines die met behulp van deze trefwoorden betere zoekresultaten retourneren in een kortere tijd. Omdat trefwoorden de belangrijkste onderwerpen van een tekst weergegeven, kunnen deze woorden in het proces van tekstcategorisatie gebruikt worden om de tekstdocumenten met elkaar te vergelijken. Samenvattend kunnen wij stellen dat trefwoorden bruikbaar zijn om grote hoeveelheden tekst te scannen in een korte tijd.

Het extraheren van trefwoorden is zeer moeilijk en kost ook erg veel tijd. Dit maakt het bijna onmogelijk om trefwoorden handmatig te extraheren. Er is daarom behoefte naar een geautomatiseerd proces dat trefwoorden extraheert uit tekstdocumenten. Machine Learning technieken beschouwen het extraheren van trefwoorden als een classificatieprobleem⁶. Een document bestaat uit een verzameling woorden en het doel is om te vast te stellen of een woord behoort tot de categorie van trefwoorden of de categorie van normale woorden. Net als in andere Machine Learning technieken dient er een trainingsset voorhanden te zijn die gebruikt kan worden om een model te trainen dat in staat is om de trefwoorden te onderscheiden van de normale woorden. Dit model kan dan toegepast worden om de trefwoorden uit nieuwe tekstdocumenten te extraheren. Veelal worden deze Machine Learning technieken gecombineerd met het Term Frequency-Inverse Document Frequency weegschema (zie paragraaf 3.4 voor een gedetailleerde beschrijving van dit weegschema).

2.3.5 TAALHERKENNING

Het doel van taalherkenning is om (automatisch) vast te stellen in welke taal een bepaald stuk tekst is geschreven. Dit kan in sommige gevallen erg handig zijn. Een goed voorbeeld hiervan is de spelling- en grammaticacontrole van Microsoft Word. De juiste spelling en grammatica zijn afhankelijk van de taal waarin een tekst is geschreven. Word tracht automatisch de taal te herkennen waarin een tekst is geschreven op basis van de eerste zinnen die worden getypt in een document. Vaak gaat dit goed, maar het komt ook voor dat Word het niet bij het rechte eind heeft en de juiste taal handmatig geselecteerd dient te worden. Een ander voorbeeld is Google. Deze zoekmachine maakt het mogelijk om te zoeken naar webpagina's die geschreven zijn in een bepaalde taal en vertaalt desgewenst de gevonden resultaten in een andere voorgespecificeerde taal.

Software voor taalherkenning is over het algemeen gebaseerd op de zogenaamde N-gram-analyse. Een N-gram is een reeks van N letters. Zo bestaat een digram uit 2 en een triagram uit 3 letters. Het woord "sport" bestaat uit de trigrammen "spo", "por", "ort". Voor een taal waarvan het alfabet bestaat uit 26 letters geldt in principe dat er 26^3 mogelijk triagrammen bestaan. Deze taal wordt gekarakteriseerd door de deelverzameling van triagrammen die echt voorkomen en hun frequenties. Aan de hand van een collectie van teksten kan een systeem de triagramverdeling van een bepaalde taal afleiden. Deze verdelingen kan vervolgens gebruikt worden om de taal van een onbekend stuk tekst vast te stellen. Des te groter de hoeveelheid informatie is waarop het systeem is getraind des te betrouwbaarder de

⁶ Bron: Uzun, Y., *Keyword Extraction Using Naive Bayes*. Bilkent University, Turkey.

analyse is. Over het algemeen zijn 1 à 2 pagina's meer dan voldoende. Dit principe werkt ook goed in het geval dat spraak is omgezet in tekst.

2.4 GERELATEERDE ONDERZOEKSGBIEDEN

2.4.1 INFORMATION RETRIEVAL (IR)

Information retrieval is een relatief oud onderzoeksgebied dat sinds 1975 bestaat en aan populariteit heeft gewonnen sinds de introductie van het Web en de hiermee gepaard gaande vraag naar geavanceerde zoekmachines. Het doel van information retrieval is het snel en nauwkeurig (terug)vinden van relevante documenten die het antwoord bevatten op een vraag van een gebruiker. Het gaat hierbij dus niet om het vinden van het antwoord zelf, maar puur om het vinden van documenten die mogelijk het antwoord bevatten. Meestal dient de gebruiker hiervoor zijn vraag in de vorm van een query te specificeren [29]. Het aantal documenten dat geretourneerd dient te worden kan vooraf gespecificeerd worden door de gebruiker en kan variëren van 1 document tot alle documenten in de gehele collectie. De geretourneerde documenten worden gesorteerd aan de hand van een bepaalde similariteitsscore die aangeeft hoe groot de overeenkomst is tussen de query en het betreffende document.

Alhoewel de definitie van information retrieval gebaseerd is op het idee van vraag en antwoord, worden ook systemen die documenten vinden op basis van sleutelwoorden, zoals de meeste zoeksystemen, veelvuldig information retrieval systemen genoemd [15].

2.4.2 NATURAL LANGUAGE PROCESSING (NLP)

Het algemene doel van NLP is het bereiken van een beter begrip van de natuurlijke taal met behulp van de computer [15]. Hierbij vindt er een interactie plaats tussen de gebruiker en de computer. Zo kan de gebruiker middels het stellen van vragen informatie extraheren uit de dataset. Hiervoor dient iedere vraag in 'natuurlijke taal' (zoals Engels of Nederlands) te worden geanalyseerd waarna het vervolgens wordt geconverteerd naar een formele taal (zoals SPARQL). Deze benadering stelt de gebruiker in staat om een vraag in te typen, die vervolgens op de achtergrond realtime wordt geconverteerd naar een query die direct kan worden uitgevoerd ([16] en [29]). Het ultieme doel van NLP is het creëren van software die computers in staat stelt om de natuurlijke taal die mensen gebruiken te begrijpen en zelf te produceren [34].

NLP is een belangrijk onderdeel van tekstmining. Het gebruik van NLP-technieken stelt tekstminingtools in staat om dichterbij de semantiek van een tekstbron te komen. Dit is belangrijk aangezien tekstminingsystemen in de toekomst steeds meer in staat moeten zijn om de gevonden informatie te verklaren [34].

2.4.3 INFORMATION EXTRACTION (IE)

Tekst die in een natuurlijke taal is geschreven bestaat uit veel informatie die niet direct bruikbaar is voor automatische analyse door een computer. Computers kunnen echter wel gebruikt worden om snel door een grote verzameling van tekst te bladeren om bruikbare informatie te extraheren uit woorden, zinsdelen en passages. Hierdoor kan IE gezien worden als een beperkte vorm van Natural Language Processing, waarbij wij op voorhand weten naar welke semantische informatie wij op zoek zijn. De hoofdtaak is om bepaalde delen van de tekst te extraheren en hieraan specifieke attributen toe te kennen.

IE bestaat uit een serie van opeenvolgende stappen die toegepast dienen te worden op de tekst om deze geschikt te maken voor verwerking met de computer. Onder deze stappen bevinden zich tokenization (tekst omzetten in tekens), sentence segmentation (opsplitsen van zinnen), part-of-speech assignment en de identificatie van bepaalde entiteiten zoals persoonsnamen, locatienamen en organisatienamen. Op een hoger niveau worden deze zinsdelen en passages verwerkt, semantisch geïnterpreteerd en geïntegreerd. Uiteindelijk worden de stukken informatie ingevoerd in een database waarna de gebruiker er taalverwerkingsmodules en dataminingstechnieken op toe kan passen.

2.5 TOEPASSINGEN VAN TEKSTMINING⁷

2.5.1 BIOINFORMATICA

Het aantal wetenschappelijke artikelen dat via het internet en grootschalige bibliografische databanken verspreid wordt is explosief toegenomen. Zo is het aantal wetenschappelijke artikelen in MEDLINE, de belangrijkste databank van de U.S. National Library of Medicine die voornamelijk wetenschappelijke informatie over geneeskunde bevat, sinds 1950 exponentieel toegenomen. Vandaag de dag bevat MEDLINE gegevens over meer dan 15 miljoen publicaties [12]. Deze gegevens bestaan onder andere uit de titel van het artikel, een samenvatting en een set van handmatig toegekende metadata termen (ook wel MeSH termen genoemd). De omvang van de MEDLINE databank maakt het voor onderzoekers moeilijk om op de hoogte te blijven van alle gepubliceerde literatuur in hun vakgebied. Daarnaast dienen de meeste onderzoekers ook op de hoogte te blijven van de ontwikkelingen in gerelateerde vakgebieden. Dit vormt een grote uitdaging in de bioinformatica. Tekstminingtools bieden in deze situatie uitkomst. Met behulp van tekstmining kan de literatuur snel gescand worden op relevante informatie en kan er hulp geboden worden bij het ontdekken van nieuwe relaties en het aandragen van nieuwe hypothesen door verschillende artikelen met elkaar te linken.

Een bepaald subprobleem dat in de bioinformatica veel aandacht krijgt van tekstmining onderzoekers is gen- en eiwitanalyse. Dit is voornamelijk te wijten aan de grote hoeveelheid beschikbare literatuur die betrekking heeft op genen en eiwitten en is deels te wijten aan de grote aandacht die genonderzoek momenteel krijgt. Automatische identificatie van gen- en eiwitnamen in wetenschappelijke artikelen is een belangrijk onderdeel van dit onderzoek [38]. Deze identificatie maakt het voor onderzoekers namelijk makkelijker om de informatie die beschikbaar is in bijvoorbeeld MEDLINE te relateren aan informatie in verwante databanken zoals LocusLink. De grootste uitdaging in dit subprobleem vormt de dubbelzinnige naamgeving van genen en eiwitten.

Vaak wordt voor de identificatie van gen- en eiwitnamen gebruik gemaakt van Support Vector Machines. De huidige nauwkeurigheden die middels deze technieken gerealiseerd worden zijn echter niet bevredigend. Momenteel wordt er om deze reden onderzoek gedaan naar het gebruik van een verfijnde mix van externe bronnen, zoals lijsten met sleutelwoorden, die moeten zorgen voor een verbetering van de nauwkeurigheid.

2.5.2 HET FILTEREN VAN SPAM

Ongewenste e-mail, oftewel spam, is de laatste jaren een steeds groter probleem aan het worden. Dagelijks worden wij geconfronteerd met meerdere ongevraagde e-mails waarin wordt geadverteerd voor dubieuze producten. Op het eerste gezicht lijkt het filteren van spam op een standaard probleem dat toebehoort aan de categorie documentclassificatie: verdeel de inkomende e-mails in twee groepen 'spam' en 'geen spam' en train de classificatiealgoritmen op de grote hoeveelheid data die in de vorm van e-mails beschikbaar is. Het probleem is echter dat de trainingdata snel verouderd. Personen die

⁷ De toepassingen die in deze paragraaf worden beschreven zijn voornamelijk ontleed aan [15, 34 en 39].

spam verspreiden bedenken namelijk steeds nieuwe technieken om de spamfilters op het verkeerde been te zetten. Een classificatietechniek die getraind is op oude spam mails zal hierdoor mogelijk niet in staat zijn om de nieuwe vorm van spam mails te ontdekken.

De eerste vorm van spamfilters verwijderde simpelweg alle e-mails die verdachte woorden bevatte welke gerelateerd konden worden aan onderwerpen zoals seks, medicijnen en kwakzalverij. Het nadeel hiervan was dat ook legitieme e-mails die deze woorden bevatte verwijderd werden. Er moest dus tijdens de trainingsfase een juiste balans gevonden waarin het verwijderen van spam niet ten koste ging van de legitieme e-mails. De ontwerpers van de spamfilters maakte om deze reden gebruik van Bayesiaanse tekstclassificatieschema's. Het probleem was echter dat de spammers zich zeer snel aanpaste aan de spamfilters door de verdachte woorden fout te spellen, door de e-mail te overweldigen met legitieme tekst die vaak werd afgedrukt in een wit lettertype zodat deze alleen zichtbaar was voor de filter of door simpelweg een bijlage of een link toe te voegen die de meeste e-mail lezers automatisch downloaden. Deze tactieken zette de meeste spamfilters op het verkeerde been en dwong de ontwerpers van de spam filters opnieuw tot nadenken.

Tegenwoordig maken de meeste geavanceerde spamfilters gebruik van Bayesiaanse netwerken waarmee de eigenschappen van spam mails geleerd kunnen worden. Deze eigenschappen worden geleerd aan de hand van e-mails waarvan de gebruiker van tevoren heeft aangegeven of deze al dan niet als spam beschouwd kunnen worden. Aan de hand van de geleerde eigenschappen worden de e-mails gefilterd. Daarnaast worden de geleerde netwerken regelmatig geüpdatet door deze te voorzien van nieuwe trainingsdata. Veel (technische) details van de spamfilters worden expres niet bekend gemaakt omdat deze informatie door spammers gebruikt kan worden om de betreffende filters te omzeilen. Deze strijd, tussen de spammers enerzijds en de spamfilters anderzijds, zal helaas nog lang voortduren. Spammers zullen altijd op zoek blijven naar nieuwe manieren om de spamfilters te slim af te zijn omdat het verspreiden van spam een lucratieve bezigheid zal blijven.

2.5.3 BUSINESS INTELLIGENCE

Bedrijven streven er naar om de risico's die belangrijke beslissingen met zich meebrengen in kaart te brengen en zoveel mogelijk te minimaliseren. De meeste datamining technieken zijn speciaal voor deze taak ontwikkeld. Deze technieken dienen de onzekerheid die besluitvorming met zich meebrengt te verminderen. Het probleem met deze technieken is dat zij alleen kunnen helpen tot een bepaald punt omdat het merendeel van de data in een bedrijf alleen beschikbaar is in de vorm van tekst. Hierbij kan gedacht worden aan rapporten, memos, e-mails, enzovoort. Aangezien datamining niet direct toegepast kan worden op tekst biedt dit veel interessante mogelijkheden voor tekstmining. Met behulp van technieken uit de tekstmining kan veel tijd en moeite bespaard worden. Zo kunnen de werknemers met behulp van deze technieken belangrijke woorden en patronen extraheren uit documenten, documenten in groepen clusteren, volstaan met het lezen van de geproduceerde samenvattingen en de onderliggende documenten alleen te raadplegen indien dit nodig is.

Tekstmining kan daarnaast ook als aanvulling gebruikt worden op datamining. Datamining technieken kunnen bijvoorbeeld gebruikt worden om een bepaalde gebeurtenis te onthullen terwijl tekstmining technieken gebruikt kunnen worden voor het verklaren van deze gebeurtenis. Tekstmining kan ook gebruikt worden om impliciete links tussen diverse bedrijven in kaart te brengen. Met behulp van deze links zijn onderzoekers in staat gebleken om de meest dominante bedrijven in verschillende industrieën in kaart te brengen evenals de relatie tussen de verschillende industrieën [5].

2.5.4 TERRORISME

De afgelopen jaren is de wereld opgeschrikt door hevige aanvallen van terrorisme waarbij vooral de Verenigde Staten het moest ontgelden. Vandaag de dag wordt er daarom veel tijd en geld besteed aan

het bestrijden van terrorisme. Hierbij worden zeer recentelijk ook tekstmining technieken ingezet. Overheidsinstelling investeren steeds meer tijd en middelen in de surveillance van meerdere vormen van communicatie, zoals e-mail. Gezien de schaal van dit probleem kost het handmatig monitoren van alle e-mails teveel tijd. Automatische tekstmining technieken kunnen in dit geval een grote rol van betekenis spelen.

Verscheidende onderzoekers zijn er reeds in geslaagd om met behulp van tekstmining technieken virussen te identificeren in de huidige medische literatuur die gebruikt kunnen worden als biologische wapens [36]. De onderzoekers gingen hierbij als volgt te werk: de beschikbare literatuur in Medline die betrekking heeft op virussen werd opgedeeld in twee verschillende delen. Het ene gedeelte bestond uit de documenten die de schadelijke werking van de virussen behandelt en het andere gedeelte bestond uit de documenten waarin de verspreiding van virussen besproken wordt. Het idee hierachter is dat een virus dat als biologisch wapen gebruikt kan worden een schadelijke werking moet hebben en snel verspreid moet kunnen worden. Op basis van de twee opgestelde groepen creëerden zij een lijst met virustermen die in beide groepen voorkwamen. Zij concludeerden dat de meeste virussen waarvan bekend was dat zij als biologisch wapen gebruikt konden worden op deze lijst voorkomen. De overige virussen die op de lijst staan zijn volgens de onderzoekers potentiële biologische wapens.

2.5.5 PATENTANALYSE

In de afgelopen jaren heeft de analyse van patenten zich ontwikkeld tot een groot toepassingsgebied. De redenen hiervoor zijn een toegenomen aantal patentaanvragen en de progressie die gemaakt is op het gebied van tekst classificatie, die ons in staat stelt om deze technieken nu ook op het gebied van patentanalyse te gebruiken. Tegenwoordig worden er supergeïsoleerde en ongesuperviseerde technieken gebruikt om patentaanvragen te analyseren. De lengte van de patentaanvragen en het grote aantal patenten waaruit het corpus bestaat vormen hierbij een grote uitdaging. Gemiddeld bestaan patentaanvragen uit 5000 woorden waarmee zij over het algemeen langer zijn dan gewone tekstdocumenten.

In verschillende studies zijn de prestaties van state-of-the-art classificatietechnieken geanalyseerd. Zo worden in het artikel van Koster et al. zeer goede resultaten gerapporteerd⁸. De betreffende onderzoekers zijn in staat gebleken om 16000 patenten met een error van slechts 3% te classificeren in 16 categorieën. Deze goede resultaten worden mede mogelijk gemaakt door de grote hoeveelheid aan beschikbare trainingsdocumenten.

Tekst clusteringtechnieken worden in de context van patentanalyse meestal door grote bedrijven gebruikt om de analyse van patenten te ondersteunen door de betreffende collectie patenten te structureren en visueel in kaart te brengen. Deze technieken zijn in veel commerciële producten terug te vinden. Daarnaast zijn deze technieken ook interessant voor onderzoekers omdat er nog steeds ruimte is voor verbeteringen. Bedrijven zoals IBM bieden producten aan die gebruikt kunnen worden bij het analyseren van patentaanvragen. Daarnaast beschikt IBM ook over een eigen patentdatabase, Delphion genaamd. Deze database wordt door veel onderzoekers gebruikt om na te gaan of iemand reeds het onderzoek heeft uitgevoerd dat zij van plan zijn om uit te voeren. In deze database kan gebruik worden gemaakt van verschillende zoektechnieken, zoals Boolean logic, query search en phrase search. Daarnaast bestaat de mogelijkheid om de zoekresultaten grafisch weer te geven in de vorm van patent clusters, waarbij de relaties tussen de verschillende clusters duidelijk wordt aangegeven.

⁸ Koster, C.H.A., M. Seutter en J. Beney (2001), classifying patent applications with winnow. In *proceedings Benelearn*, Antwerpen.

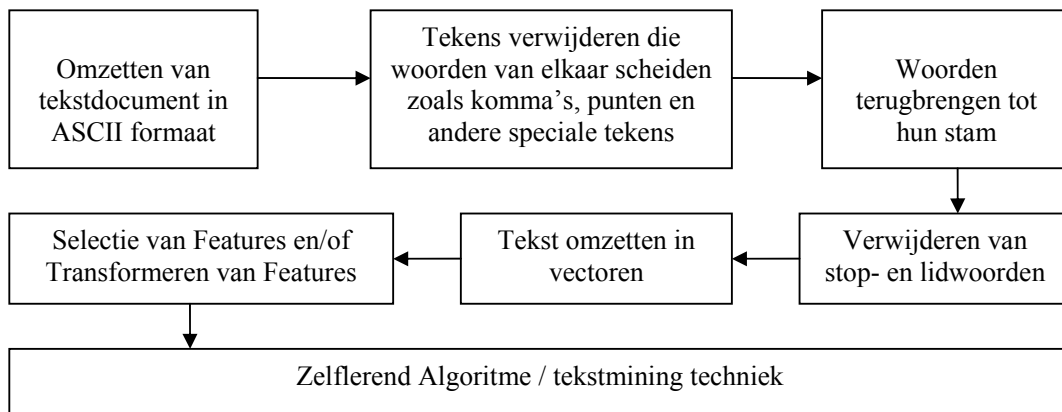
HOOFDSTUK 3

HET TEKSTMINING PROCES

3.1 INLEIDING

De meest gebruikte classifiers of zelflerende algoritmes kunnen niet direct toegepast worden op de tekstdocumenten in hun originele vorm. Daarom worden in de eerste stap van het tekstmining proces de tekstdocumenten geconverteerd naar een meer handelbare representatie. Om deze representatie te verkrijgen dient het tekstdocument de eerste vijf stappen van het proces dat is afgebeeld in figuur 3.1 te doorlopen. Gedurende deze stappen wordt het tekstdocument ingelezen en omgezet in een datastructuur (meestal een featurevector) die geschikter is voor verdere verwerking (door een computer) dan een simpel tekstbestand. Hierna kunnen de irrelevante woorden verwijderd worden uit deze verzameling of kan de gehele verzameling getransformeerd worden. Nadat alle tekstdocumenten op deze manier zijn voorbehandeld kan men daadwerkelijk beginnen met het toepassen van de analysetechnieken.

In dit hoofdstuk zullen wij uitgebreid ingaan op de stappen die hierboven zijn beschreven en afgebeeld zijn in figuur 3.1. In de volgende paragraaf zullen eerst de eerste vier stappen waarin het document wordt voorbehandeld voor verdere verwerking aan bod komen. In de hier opvolgende paragraaf zal het vectorruimte model behandeld worden. Daaropvolgend zullen in paragraaf 3.4 enkele technieken besproken worden waarmee irrelevante woorden geselecteerd en verwijderd kunnen worden. Afsluitend zal in de laatste paragraaf ruime aandacht geschonken worden aan technieken waarmee de features getransformeerd kunnen worden.



Figuur 3.1: De zeven stappen die in het tekstmining proces doorlopen dienen te worden.

3.2 TEKSTDUMENTEN VOORBEHANDELEN

3.2.1 HET OMZETTEN VAN HET DOCUMENT IN ASCII FORMAAT

De originele documenten die gebruikt worden in het tekstmining proces kunnen verschillende formaten hebben, zoals pdf, doc, html en txt. De eerste stap in het tekstmining proces is het omzetten van deze documenten in een standaardformaat wat in dit geval het ASCII formaat is. Hierbij dient er wel rekening gehouden te worden met het feit dat sommige documenten extra informatie bevatten. Zo

bevatten HTML pagina's tags zoals titel, hyperlink, headings en trefwoorden. E-mails bevatten op hun beurt weer een gedeelte dat speciaal gereserveerd is voor het onderwerp van de mail. De structuur van deze documenten kan dus waardevolle informatie bevatten. Bij het omzetten van deze formaten dient rekening gehouden te worden met deze "rijkere" structuur.

3.2.2 HET VERWIJDEREN VAN TEKENS

Om alle woorden te selecteren die in een tekstdocument voorkomen wordt het tekstdocument ingelezen en omgezet in ASCII formaat waarna het weergegeven wordt als een willekeurige rij van woorden waarbij alle leestekens zijn verwijderd en waarbij tabs en andere niet-tekstuele tekens zijn vervangen door een spatie. Het tekstdocument wordt nu gerepresenteerd door de overgebleven verzameling van unieke woorden. Deze verzameling kan op zijn beurt weer verder behandeld worden.

3.2.3 FILTERING EN STEMMING VAN WOORDEN

Om de dimensies van het vocabularium, dat bestaat uit alle unieke woorden die voorkomen in de gehele verzameling van tekstdocumenten, te reduceren kan gebruik worden gemaakt van filtering- en stemmingstechnieken.

Filteringstechnieken verwijderen woorden uit het vocabularium en daarmee dus ook uit de tekstdocumenten. Een standaard filteringstechniek die vaak toegepast wordt is het filteren van stopwoorden. Het idee hierachter is dat de woorden verwijderd worden die nauwelijks tot geen informatie bevatten en die alleen voor verwarring zorgen in de betreffende classificatie- en clustertechnieken. Het gaat hier bijvoorbeeld om lidwoorden, voegwoorden en voorzetsels. Vaak worden in de praktijk ook woorden die heel vaak voorkomen verwijderd omdat deze woorden weinig onderscheidend vermogen bezitten. Daarnaast worden ook vaak de woorden die erg weinig voorkomen verwijderd uit het vocabularium omdat de kans groot is dat deze woorden statistisch niet relevant zijn [12].

Stemmingstechnieken proberen woorden tot hun stam of basisvorm te herleiden. De stam van een werkwoord is een grondvorm waarvan andere werkwoordelijke vormen afgeleid kunnen worden [Van Dale Groot woordenboek hedendaags Nederlands]. Een voorbeeld hiervan is: "helpen", "helpt", "hielp", "hielpen", "geholpen" die alle dezelfde stam hebben ("help"). Tijdens het stemmen worden alle woorden vervangen door hun stam. Stemming speelt een belangrijke rol in classificatie omdat het zorgt voor een reductie in het aantal verschillende woordvormen en speelt eveneens een belangrijke rol in zoektechnieken omdat het een alternatief biedt voor de letterlijke zoekterm. Voor de Engelse taal kan gebruik worden gemaakt van de stemmer die geïntroduceerd en ontwikkeld is door Porter [28]. Hij heeft een set van regels gedefinieerd die op een iteratieve wijze Engelse woorden herleidt tot hun stam. Een soortgelijke stemmer voor de Nederlandse taal hebben wij echter niet kunnen vinden.

Binnen de tekstmining twijfelt men echter of stemming daadwerkelijk zorgt voor een betere prestatie van de algoritmen. Zo is uit sommige onderzoeken gebleken dat stemming de effectiviteit van de algoritmen schaadt [35].

3.3 VECTORRUIMTEMODEL

Tijdens het inlezen van de tekstdocumenten telt men alle woorden die voorkomen in een document waarbij de volgorde van de woorden en de structuur van de zinnen wordt genegeerd. De resulterende aantallen worden bewaard in een (*document* × *woord*) matrix. Hierbij stelt elke kolom een woord voor en elke rij een document (zie figuur 3.2). Soms worden zinnen of bepaalde combinaties van

woorden (zoals “internet bankieren”) ook behandeld als één woord. Alle m woorden die in minstens één document voorkomen vormen tezamen het vocabularium. Een waarde $x_{i,j}$ geeft het aantal keer aan dat woord j voorkomt in document i . Dit wordt ook wel de “bag of words” representatie genoemd.

Elk document (oftewel rij) kan voorgesteld worden als een punt, een coördinaat of een vector in de m -dimensionale vectorruimte waarin elk woord een dimensie voorstelt. De $(document \times woord)$ matrix kan erg groot worden indien men een grote verzameling tekstdocumenten tot zijn beschikking heeft. De orde van grootte van n kan tientallen miljoenen zijn. De grootte van het vocabularium is begrensd door het aantal unieke woorden of andere tekenreeksen die voorkomen in de collectie van tekstdocumenten [19]. De grootte van het vocabularium kan drastisch gereduceerd worden door het voorbehandelen van de tekst (paragraaf 3.2) en door het toepassen van dimensie-reductie technieken (paragraaf 3.4).

	Woord 1	Woord 2	Woord 3	...	Woord m
Document 1	2	0	4	...	3
Document 2	5	1	2	...	6
Document 3	3	9	0	...	1
Document 4	0	0	2	...	4
...
Document n	1	0	0	...	0

Figuur 3.2: De $(document \times woord)$ matrix.

De $(document \times woord)$ matrix bestaat grotendeels uit nullen. Hierdoor is het niet noodzakelijk om de gehele matrix op te slaan, maar kan men volstaan met het opslaan van de elementen die ongelijk zijn aan nul.

3.4 TF-IDF WEEGSCHEMA

Vaak worden de aantallen in de $(document \times woord)$ matrix in figuur 3.2 gewogen door middel van het **TF-IDF** (Term Frequency – Inverse Document Frequency) weegschema waarbij TF wordt vermenigvuldigd met IDF. Dit weegschema is ook een vorm van feature selectie. Term frequency, oftewel woord frequentie, aangegeven met $TF(w, d)$, geeft het aantal keer aan dat woord w voorkomt in document d . Document frequency, $DF(w)$, geeft het aantal documenten aan waarin het woord w minstens één keer voorkomt. De inverse document frequency geeft aan hoe belangrijk een bepaald woord in het algemeen is en kan aan de hand van de document frequency afgeleid worden. IDF is als volgt gedefinieerd:

$$IDF(w) = \log\left(\frac{|D|}{DF(w)}\right), \quad (3.1)$$

waarbij $|D|$ het totale aantal documenten in de verzameling voorstelt. Op basis van de bovenstaande formule kunnen wij afleiden dat de inverse document frequency van een woord laag is indien het woord in veel documenten voorkomt en op zijn hoogst is indien het woord maar in 1 document voorkomt. Nu geldt dat de waarde TF-IDF voor woord w in document d als volgt wordt bepaald:

$$TFIDF(w, d) = TF(w, d) \cdot IDF(w). \quad (3.2)$$

$TFIDF(w, d)$ stelt het gewicht van woord w in document d voor. Volgens het TF-IDF weegschema is woord w een belangrijk/karakteristiek woord voor document d indien het woord vaak voorkomt in document d (hoge TF) maar niet of nauwelijks voorkomt in andere documenten (lage IDF). In dit geval zal $TFIDF(w, d)$ een grote waarde hebben. Indien het woord w in veel documenten voorkomt zal ook de waarde $TFIDF(w, d)$ laag zijn. Het weegschema zorgt er uiteindelijk voor dat de meest voorkomende termen een laag gewicht krijgen en daardoor geen belangrijke rol meer spelen bij het toepassen van de tekstmining technieken. Soms komt het voor dat de gewichten $TFIDF(w, d)$ genormaliseerd worden zodat alle documenten dezelfde (Euclidische) lengte⁹ hebben. In dit geval geldt:

$$TFIDF(w, d) = \frac{TF(w) \cdot IDF(w)}{\sqrt{\sum_{i=1}^m (TF(w_i)^2 \cdot IDF(w_i)^2)}}. \quad (3.3)$$

Op basis van het TF-IDF weegschema kan bijvoorbeeld de gelijkheid tussen twee documenten d_1 en d_2 berekend worden (of de gelijkheid tussen een document en een query). Deze gelijkheid wordt berekend door het inwendig product van de twee vectoren die de documenten voorstellen te berekenen. In het geval dat de lengtes van de vectoren genormaliseerd zijn komt dit overeen met de cosinus van de hoek tussen de twee documenten:

$$\text{Gelijkheid}(d_1, d_2) = \sum_{i=1}^m TFIDF(d_1, w_i) \cdot TFIDF(d_2, w_i). \quad (3.4)$$

Daarnaast kan ook de afstand tussen twee vectoren berekend worden met behulp van de Euclidische afstand. Deze afstand tussen twee documenten d_1 en d_2 wordt als volgt berekend:

$$\text{afstand}(d_1, d_2) = \sqrt{\sum_{i=1}^m (TFIDF(d_1, w_i) - TFIDF(d_2, w_i))^2}. \quad (3.5)$$

De Euclidische afstand kan echter alleen gebruikt worden in het geval dat de vectoren die de documenten representeren genormaliseerd zijn omdat verschillen in lengtes kunnen zorgen voor een kleinere afstand tussen vectoren die weinig identieke woorden bevatten, maar een kleine lengte hebben (d.w.z. uit weinig componenten bestaan) en een grotere afstand tussen vectoren die juist veel identieke woorden bevatten, maar een grote lengte hebben. Juist in het laatste geval dient de afstand tussen de twee documenten kleiner te zijn dan in het eerste geval omdat vectoren die veel identieke woorden bevatten ook meer op elkaar moeten lijken en dus ook dichter bij elkaar moeten liggen.

3.5 FEATURESELECTIE

Het doel van feature¹⁰ selectie is het reduceren van de dimensies van de (*document* × *woord*) matrix door woorden uit het vocabularium te verwijderen die irrelevant beschouwd worden voor het classificeren of clusteren van de tekstdocumenten. Getracht wordt om de zogenaamde “curse of dimensionality” te verminderen.

⁹ De Euclidische lengte van een n-dimensionale vector $x = (x_1, x_2, \dots, x_n)$ is gedefinieerd als

$$|x| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}.$$

¹⁰ In deze context is een feature niets anders dan een woord.

De voordelen van feature selectie zijn:

1. Een kleinere (*document* × *woord*) matrix.
2. Minder geheugengebruik door de computer waardoor het trainen van de classificatiealgoritmen ook sneller gaat.
3. Een verkleining van de algehele zoekruimte waardoor de classificatiealgoritmen sneller in staat zijn om een hypothese te vinden die de patronen in de data zo goed mogelijk beschrijft.
4. In de meeste gevallen verwijdert het ook ruis uit de data waardoor de classificatiealgoritmen beter presteren.

Een bijkomend voordeel is dat feature selectie het fenomeen van overfitting tegen gaat. In het geval van overfitting leert het algoritme bepaalde specifieke karakteristieken van de trainingdata die niet gedeeld worden door de gehele verzameling aan data. Hierdoor is het algoritme minder goed in staat om te generaliseren, wat de nauwkeurigheid bij het classificeren van nieuwe tekstdocumenten niet ten goede komt.

3.5.1 FEATURESELECTIE TECHNIKEN

In tabel 3.1 zijn enkele (statistische) featureselectie technieken weergegeven die in de praktijk vaak gebruikt worden, namelijk: chi-kwadraat, information gain, odds ratio, (log) probability ratio en document frequency [11]¹¹. Aan de hand van deze technieken kan voor elk woord in het vocabularium een bepaalde score worden berekend op basis waarvan een rangschikking kan worden gemaakt. Deze rangschikking kan daarna gebruikt worden om de woorden te selecteren met de hoogste scores die dan dienst zullen doen als het nieuwe vocabularium. In het vervolg van deze subparagraaf zal elke feature selectietechniek kort beschreven worden.

Chi-kwadraat is een statistiek die in een experimentele omgeving gebruikt wordt om te bepalen hoe groot de afwijking (of afhankelijkheid) van de waarneming is ten op zichte van het verwachte resultaat dat vooraf wordt gespecificeerd in de nulhypothese [35]. Hierbij geldt dat des te lager de waarde van deze statistiek is, des te lager de afhankelijkheid van de waarneming is. In de context van tekstmining gebruikt men deze statistiek om de afhankelijkheid tussen een woord en een bepaalde categorie te meten. Een kleine waarde voor deze statistiek geeft aan dat het woord veelal onafhankelijk voorkomt van de klasse. Aangezien wij juist op zoek zijn naar de woorden die een hoge afhankelijkheid vertonen met de categorie waarin zij voorkomen, selecteren wij per categorie die woorden met de grootste waarde voor de chi-kwadraat statistiek.

Information gain berekent het aantal bits aan informatie dat nodig is om de categorie waartoe een bepaald tekstdocument behoort te voorspellen op basis van het gegeven dat een bepaald woord wel of niet in het tekstdocument voorkomt. Een hoge information gain geeft aan dat het wel of niet aanwezig zijn van het woord bruikbare informatie oplevert die gebruikt kan worden om vast te stellen in welke categorie een willekeurig tekstdocument thuis hoort. Men berekent doorgaans de information gain voor alle woorden in het vocabularium en selecteert de woorden met de hoogste information gain.

Odds ratio geeft de verhouding aan tussen de waarschijnlijkheid (odds) dat een woord voorkomt in de positieve categorie en de waarschijnlijkheid (odds) dat het woord voorkomt in de negatieve categorie. Een odds ratio die gelijk is aan 1 houdt in dat het even waarschijnlijk is dat het woord in beide categorieën voorkomt. Een odds ratio groter dan 1 geeft aan dat het waarschijnlijker is dat het betreffende woord in de positieve categorie voorkomt en een odds ratio kleiner dan 1 geeft daarentegen aan dat het waarschijnlijker is dat het woord in de negatieve categorie voorkomt. Deze maat heeft dus een voorkeur voor woorden die voorkomen in de positieve categorie die daardoor ook in de top van de geproduceerde ranking terecht zullen komen. In de literatuur is het bekend dat de odds ratio goed werkt in combinatie met het Naïve Bayes algoritme.

¹¹ Tabel 3.1 en 3.2 zijn ontleend aan [11].

(Log) probability ratio geeft de verhouding aan tussen de kans op een bepaald woord gegeven het feit dat wij ons bevinden in een bepaalde categorie c en de kans op hetzelfde woord maar nu gegeven het feit dat wij ons niet bevinden in deze categorie. Woorden met een grote (log) probability krijgen hierbij de voorkeur omdat zij een indicatie geven tot welke categorie een tekstdocument behoort.

Document frequency (oftewel documentfrequentie) geeft simpelweg aan in hoeveel documenten het betreffende woord voorkomt. Experimenten hebben aangetoond dat de nauwkeurigheid van de classificatietechnieken (zoals Naïve Bayes en k-Nearest Neighbor) niet vermindert indien de top 10% van de woorden die het meest voorkomen gebruikt wordt [30]. Dit is op het eerste gezicht in strijd met de welbekende wet uit IR die stelt dat woorden met een lage tot gemiddelde document frequency de meeste informatie bevatten. Dit is echter niet het geval omdat de meeste woorden een zeer lage document frequency hebben waardoor de top 10% in de meeste gevallen ook de woorden met een lage tot gemiddelde document frequency bevat.

Tabel 3.1: De algemeen bekende featureselectie technieken.

Feature selectietechniek	Wiskundige Formule
Chi-kwadraat	$\chi^2(f_i, c_j) = \frac{ D \cdot (\#(c_i, f_j) \#(\bar{c}_i, \bar{f}_j) - \#(c_i, \bar{f}_j) \#(\bar{c}_i, f_j))^2}{(\#(c_i, f_j) + \#(c_i, \bar{f}_j)) \cdot (\#(\bar{c}_i, f_j) + \#(\bar{c}_i, \bar{f}_j)) \cdot (\#(c_i, f_j) + \#(\bar{c}_i, f_j)) \cdot (\#(c_i, \bar{f}_j) + \#(\bar{c}_i, \bar{f}_j))}$
Information gain	$IG(t) = -\sum_{i=1}^m P(c_i) \log P(c_i) + P(t) \sum_{i=1}^m P(c_i t) \log P(c_i t) + P(\bar{t}) \sum_{i=1}^m P(c_i \bar{t}) \log P(c_i \bar{t})$
Odds ratio	$OddsRatio(f_i, c_j) = \log \frac{P(f_i c_j) (1 - P(f_i \neg c_j))}{(1 - P(f_i c_j)) P(f_i \neg c_j)}$
Log probability ratio	$Log ProbRatio(w) = \log \frac{P(w c)}{P(w \neg c)}$
Document frequency	$DF(t) = \#(t)$

Tabel 3.2: Betekenis van de symbolen die gehanteerd worden in tabel 3.1.

Symbool	Betekenis
c	Een voorgedefinieerde categorie of klasse.
C	De verzameling van alle voorgedefinieerde categorieën.
d	Een tekstdocument afkomstig uit de trainingset.
D	De gehele verzameling van tekstdocumenten.
t, f of w	Een term, feature of een woord.
$P(c)$ of $P(c_i)$	De kans op categorie c of c_i , d.w.z. de frequentie waarmee deze categorie voorkomt in de trainingset.
$P(\neg c)$ of $P(\bar{c})$	De kans dat categorie c niet voorkomt.
$P(c t)$	De kans dat een tekstdocument behoort tot categorie c gegeven het feit dat term t voorkomt in het betreffende tekstdocument.
$P(c \bar{t})$	De kans dat een tekstdocument behoort tot categorie c gegeven het feit dat term t niet voorkomt in het betreffende tekstdocument.
$P(c, t)$	De kans dat categorie c en term t gelijktijdig voorkomen.
$DF(t_i)$	De document frequentie van term t_i .
$\#(c, t)$	Het aantal documenten die term t bevatten en behoren tot categorie c .
$\#(t)$	Het aantal documenten dat term t bevat.

3.6 FEATURE TRANSFORMATIE

Feature transformatie verschilt wezenlijk van featureselectie maar heeft hetzelfde doel: het terugbrengen van de grootte van het vocabularium en daarmee de dimensies van de (*document* × *woord*) matrix. In het geval van feature transformatie wordt er een kunstmatig vergaarde set van nieuwe features opgesteld. In de praktijk betekent dit dat de originele (*document* × *woord*) matrix wordt getransformeerd in een nieuwe matrix waarvan de dimensies (veel) kleiner zijn, en waarvan de cellen de kunstmatig berekende features bevatten. De achterliggende reden waarom men kunstmatige features gebruikt, in plaats van de natuurlijke woorden die voorkomen in een taal, is dat dankzij polysemie (meerduidigheid), homonymie (gelijkluidende maar in betekenis verschillende woorden) en synonymie (woorden met een gelijke betekenis) woorden niet altijd de optimale features zijn [30]. Door deze woorden te transformeren kan een kunstmatige verzameling van features verkregen worden waarmee de tekstdocumenten gerepresenteerd kunnen worden en welke niet lijden onder de eigenschappen van de natuurlijke taal.

3.6.1 PRINCIPAL COMPONENTS ANALYSE (PCA)

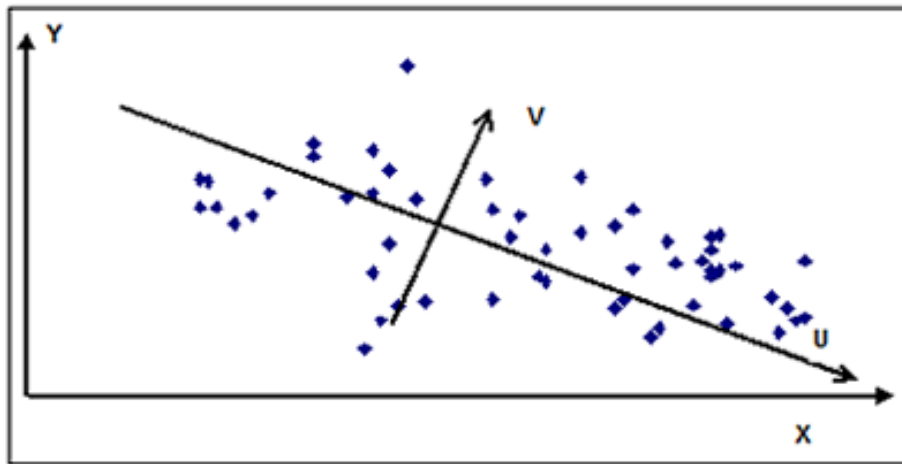
De tekstdocumenten die gerepresenteerd worden door de (*document* × *woord*) matrix kunnen weergegeven worden als punten in een m-dimensionale ruimte. De woorden geven hierbij de coördinaten van de tekstdocumenten in deze ruimte aan. De positie van de assen kan men echter zelf bepalen [30]. Zo kan men ervoor kiezen om alle documenten te transformeren in een ander coördinatenstelsel, dat niet gedefinieerd is volgens één of andere conventie, maar dat gerechtvaardigd wordt door de data zelf. Onafhankelijk van het coördinatenstelsel dat men gebruikt, heeft de puntenwolk een bepaalde variantie in elke richting die aangeeft wat de spreiding is rond het gemiddelde. Een merkwaardig verschijnsel is dat wanneer de variantie in elke richting bij elkaar wordt opgeteld en men hetzelfde doet wanneer de punten zijn getransformeerd naar een ander coördinatenstelsel, uiteindelijk dezelfde totale variantie wordt verkregen. Dit geldt zolang het coördinatenstelsel orthogonaal is, dat wil zeggen dat alle assen loodrecht op elkaar staan.

Het idee achter PCA is dat men een coördinatenstelsel gebruikt dat afhankelijk is van de beschikbare data. Om dit coördinatenstelsel op te stellen gaat men als volgt te werk: de eerste as wordt geplaatst langs de richting waarin de variantie het grootst is. Dit wordt ook wel de eerste principal component genoemd. De tweede as kan nu overal geplaatst worden zolang deze as loodrecht staat op de eerst gekozen as. Naast deze restrictie dient men de as zodanig te plaatsen dat de som van de varianties gemaximaliseerd wordt. Deze as vormt dan de tweede principal component. Op deze manier gaat men verder en plaatst men de nieuwe assen zodanig dat de som van de varianties langs alle reeds geplaatste assen maximaal is. In het geval van tekstdocumenten geldt dat de principal components een lineaire combinatie zijn van de woorden in het vocabularium.

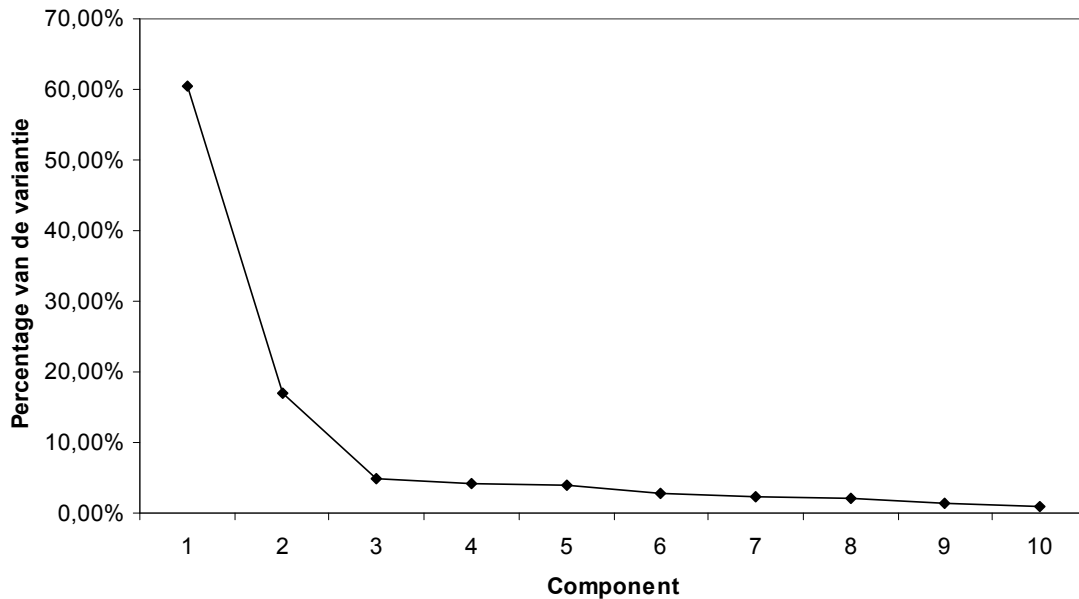
Het bovenstaande proces is geïllustreerd in figuur 3.3. In deze figuur is een verzameling datapunten weergegeven in het X-Y assenstelsel. Duidelijk is dat de variantie langs de U-as het grootst is. Deze as vormt dan ook de eerste principal component en is een lineaire combinatie van de X- en Y-as. De tweede principal component is de V-as. Deze as staat loodrecht op de U-as en is ook een lineaire combinatie van de X- en Y-as. De U- en V-as vormen nu het nieuwe assenstelsel. De principal component analyse heeft dus uiteindelijk geleid tot een verschuiving van de X- en Y-as.

Globaal werkt PCA op de volgende manier: als eerst wordt het gemiddelde van elke rij (dimensie) berekend en wordt dit gemiddelde afgetrokken van alle componenten die zich bevinden in deze rij. Daarna wordt de covariantiematrix opgesteld en worden de eigenvectoren en eigenwaarden van deze matrix berekend. De eigenvectoren staan loodrecht op elkaar en beschrijven de achterliggende patronen in de data. Deze eigenvectoren vormen de assen van het nieuwe coördinatenstelsel en worden aan de hand van de bijbehorende eigenwaarde, die aangeeft wat de verklaarde variantie is langs elke

eigenvector, gesorteerd van hoog naar laag. De originele covariantiematrix heeft m eigenvectoren en de dimensiereductie vindt plaats door alleen de grootse p ($p \ll m$) eigenvectoren te selecteren. Hierdoor zal de getransformeerde matrix ook p dimensies hebben. Men kan er ook voor kiezen om de eigenvectoren zodanig te kiezen dat de som van de verklaarde varianties een bepaalde voorgedefinieerde grootte heeft. Uit figuur 3.4 blijkt bijvoorbeeld dat de eerste 3 componenten samen meer dan 80% van de variantie verklaren. De laatste stap is het transformeren van de originele ($document \times woord$) matrix naar het nieuwe coördinatenstelsel.



Figuur 3.3: Het resultaat van PCA toegepast op een verzameling datapunten in het X-Y vlak (bron: http://www.med.govt.nz/templates/MultipageDocumentPage___1065.aspx).



Figuur 3.4: Het percentage van de verklaarde variantie uitgezet tegen elke component.

3.6.2 SINGULIERE WAARDEN DECOMPOSITIE

Een singuliere waarden decompositie (Engels: Singular Value Decomposition) van een $(m \times n)$ matrix A , waarvan de elementen afkomstig zijn uit de verzameling van reële getallen, is een representatie van deze matrix als een product van 3 matrices:

$$A = U\Sigma V^T, \quad (3.6)$$

waarbij U een kolom-orthonormale $(m \times r)$ matrix is, Σ een diagonale $(r \times r)$ matrix is, V een kolom-orthonormale $(n \times r)$ matrix voorstelt en r de rang¹² van de matrix A voorstelt. De term kolom-orthonormaal betekent dat de kolomvectoren genormaliseerd zijn (dat wil zeggen dat deze vectoren een lengte hebben die gelijk is aan 1) en loodrecht op elkaar staan. Dit houdt in dat het inwendige product van deze twee vectoren gelijk is aan 0 en dat $UU^T = V^T V = I$, waarbij I de identiteitsmatrix voorstelt. De diagonale elementen van de matrix Σ zijn de singuliere waarden van A en worden van groot naar klein gerangschikt.

3.6.3 LATENT SEMANTIC INDEXING (LSI)

LSI is een dimensiereductie techniek die gebruik maakt van de singuliere waarden decompositie. Deze techniek houdt in dat de originele data die in de vorm van een $(document \times woord)$ matrix is gegoten door middel van de singuliere waarden decompositie opgedeeld wordt in componenten die lineair onafhankelijk zijn. Deze componenten benaderen zo goed mogelijk de originele data langs elke dimensie. Het merendeel van deze componenten kan genegeerd worden omdat deze erg klein zijn. Dit resulteert in een benadering van de originele data die uit veel minder dimensies bestaat. De singuliere waarden decompositie zorgt er daarnaast voor dat de documenten die al op voorhand enigszins op elkaar lijken nog meer op elkaar gaan lijken en dat de verschillen tussen de documenten die niet op elkaar lijken groter worden. In de praktijk betekent dit dat de documenten die hetzelfde onderwerp hebben meer overeenkomsten zullen vertonen na de singuliere waarden decompositie zonder dat alle woorden die in deze documenten voorkomen precies hetzelfde zijn.

De dimensiereductie gaat als volgt in zijn werk. Als eerst wordt er een rechthoekige $(document \times woord)$ matrix A opgesteld. De kolommen van deze matrix stellen de vectorrepresentaties van de documenten voor. Het element A_{wd} geeft aan hoe vaak het woord w voorkomt in document d . Eventueel kan op deze matrix ook het TF-IDF weegschema toegepast worden.

Nadat de matrix A is opgesteld wordt de singuliere waarden decompositie op deze matrix toegepast. Hierbij zijn wij dus op zoek naar drie matrices zodanig dat het product van deze matrices voldoet aan:

$$A = U\Sigma V^T, \quad (3.7)$$

waarbij U en V orthonormaal zijn en Σ diagonaal is. Om de bovenstaande stappen te illustreren maken wij gebruik van de volgende $(woord \times document)$ matrix¹³:

¹² De rang van een matrix is het maximale aantal lineair onafhankelijke rijen of kolommen van een matrix.

¹³ Dit voorbeeld is ontleend aan [2].

$$A = \begin{pmatrix} 2 & 0 & 8 & 6 & 0 \\ 1 & 6 & 0 & 1 & 7 \\ 5 & 0 & 7 & 4 & 0 \\ 7 & 0 & 8 & 5 & 0 \\ 0 & 10 & 0 & 0 & 7 \end{pmatrix}$$

De singuliere waarden decompositie van deze matrix is dan:

$$A = \begin{pmatrix} -0.54 & 0.07 & 0.82 & -0.11 & 0.12 \\ -0.10 & -0.59 & -0.11 & -0.79 & -0.06 \\ -0.53 & 0.06 & -0.21 & 0.12 & -0.81 \\ -0.65 & 0.07 & -0.51 & 0.06 & 0.56 \\ -0.06 & -0.80 & 0.09 & 0.59 & 0.04 \end{pmatrix} \begin{pmatrix} 17.92 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0 & 15.17 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 3.56 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.98 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.35 \end{pmatrix} \begin{pmatrix} -0.46 & 0.02 & -0.87 & -0.00 & 0.17 \\ -0.07 & -0.76 & 0.06 & 0.60 & 0.23 \\ -0.74 & 0.10 & 0.28 & 0.22 & -0.56 \\ -0.48 & 0.03 & 0.40 & -0.33 & 0.70 \\ -0.07 & -0.64 & -0.04 & -0.69 & -0.32 \end{pmatrix}$$

In de matrix U worden de woorden gerepresenteerd als rijvectoren die bestaan uit lineair onafhankelijke componenten. De patronen waarin de woorden samen voorkomen kunnen aan de hand van de tekens van de getallen in deze matrix worden afgeleid. Zo kunnen wij bijvoorbeeld aan de hand van de tekens in de tweede kolom zien dat woord 2 en 5 samen voorkomen in een groep en dat de woorden 1, 3 en 4 in een andere groep voorkomen. De matrix Σ bevat de singuliere waarden die op de diagonaal geordend zijn van groot naar klein. Deze waarden geven de variantie van de lineair onafhankelijke componenten langs elke dimensie aan. Duidelijk is dat de eerste twee singuliere waarden de meeste variantie bevatten wat betekent dat de data dus het best gereduceerd kan worden naar deze twee dimensies.

Nadat de singuliere waarden decompositie is uitgevoerd vindt de daadwerkelijke dimensiereductie plaats. Hiervoor worden alleen de k grootste singuliere waarden in de matrix Σ geselecteerd en worden de overige singuliere waarden gelijk gesteld aan 0, wat resulteert in een matrix Σ' . Men kan aantonen dat de matrix met rang k :

$$A' = U' \Sigma' V'^T, \quad (3.8)$$

een benadering is van de matrix A met de kleinste fout (in kleinste kwadraten opzicht). Het verwijderen van de kleinste singuliere waarden zorgt er voor dat de ruis in de (*woord* \times *document*) matrix effectief wordt verwijderd. Als gevolg hiervan worden de verborgen (Engels: latent) overeenkomsten tussen de documenten zichtbaar. Daarnaast geeft deze reductie LSI het intelligente vermogen om semantisch gerelateerde termen te groeperen waarbij gebruik wordt van het feit dat woorden die dezelfde betekenis hebben vaak samen voorkomen.

In het bovenstaande voorbeeld kunnen wij bijvoorbeeld de dimensies reduceren door alleen de grootste drie singuliere waarden in de matrix Σ te selecteren. Om een benadering van A te verkrijgen in drie dimensies dienen wij de eerste drie kolomvectoren van de matrix U te selecteren en de eerste drie rijvectoren van de matrix V^T . De benadering van A in drie dimensies ziet er dan als volgt uit:

$$\hat{A} = \begin{pmatrix} -0.54 & 0.07 & 0.82 \\ -0.10 & -0.59 & -0.11 \\ -0.53 & 0.06 & -0.21 \\ -0.65 & 0.07 & -0.51 \\ -0.06 & -0.80 & 0.09 \end{pmatrix} \begin{pmatrix} 17.92 & 0.0 & 0.0 \\ 0.0 & 15.17 & 0.0 \\ 0.0 & 0.0 & 3.56 \end{pmatrix} \begin{pmatrix} -0.46 & 0.02 & -0.87 & -0.00 & 0.17 \\ -0.07 & -0.76 & 0.06 & 0.60 & 0.23 \\ -0.74 & 0.10 & 0.28 & 0.22 & -0.56 \end{pmatrix} = \begin{pmatrix} 2.29 & -0.66 & 9.33 & 1.25 & -3.09 \\ 1.77 & 6.76 & 0.90 & -5.50 & -2.13 \\ 4.86 & -0.96 & 8.01 & 0.38 & -0.97 \\ 6.62 & -1.23 & 9.58 & 0.24 & -0.71 \\ 1.14 & 9.19 & 0.33 & -7.19 & -3.13 \end{pmatrix}$$

In de praktijk is het echter niet de bedoeling om de gehele matrix \hat{A} te construeren, maar dienen de gereduceerde dimensiematrices gebruikt te worden om de woorden en documenten die veel op elkaar lijken te traceren. De documenten worden in de matrix V nu weergegeven als rijvectoren (omdat wij nu werken met de matrix V' in plaats van V^T) en de overeenkomsten tussen documenten worden nu verkregen door het vergelijken van de rijen in de matrix $V'\Sigma$. Door bijvoorbeeld gebruik te maken van de cosinus maat kan een clustering van de documenten verkregen worden. Tevens kan de matrix $V'\Sigma$ gebruikt worden om de documenten te categoriseren.

De woorden waaruit het vocabularium bestaat worden weergegeven als rijvectoren in de matrix U en de overeenkomsten tussen deze woorden kunnen verkregen worden door de rijen in de matrix $U'\Sigma$ met elkaar te vergelijken [2].

AUTOMATISCHE TEKSTCLASSIFICATIE

4.1 INLEIDING

Automatische tekstclassificatie behelst het automatisch classificeren van tekstdocumenten in voorgedefinieerde categorieën en is een vorm van gesuperviseerd leren: alle categorieën zijn van tevoren vastgesteld en alle tekstdocumenten zijn voorzien van een label dat aangeeft tot welke categorie het behoort. De ongesuperviseerde vorm van tekstclassificatie, clustering genaamd, wordt in hoofdstuk 5 besproken.

In de loop der jaren zijn er in de wetenschap verschillende methoden ontwikkeld en onderzocht die gebruikt kunnen worden om tekstdocumenten automatisch te classificeren. In dit hoofdstuk zullen wij de best presterende tekstclassificatie technieken beschrijven. Hierbij zal kort ingegaan worden op de achterliggende theorie en de voor- en nadelen van elke techniek. Als eerst zal in de volgende paragraaf een formele definitie gegeven worden van tekstclassificatie. In de hierop volgende paragraaf zullen verschillende classificatietechnieken beschreven worden. Aan bod komen achtereenvolgens de technieken: Naïve Bayes (NB), k-Nearest Neighbor (kNN), Support Vector Machines (SVM) en BoosTexter. De laatste stap in het classificatieproces is het vaststellen van de prestaties van het opgestelde model. De technieken die hiervoor gebruikt kunnen worden zijn het onderwerp van de laatste paragraaf. In deze paragraaf zal tevens een onderlinge vergelijking van de prestaties van de beschreven technieken terug te vinden zijn.

4.2 DEFINITIE VAN TEKSTCLASSIFICATIE

Tekstclassificatie is gericht op het indelen van documenten in voorgedefinieerde categorieën [26]. Een voorbeeld is het automatisch indelen van een klacht in één van de volgende categorieën: “lange wachttijd”, “hoge gesprekskosten” of “onvriendelijke medewerkers”. Het uiteindelijke doel is om op basis van deze trainingsset en een analyse van de karakteristieken van de teksten die deel uitmaken van deze trainingsset een classificatiemodel:

$$f : D \rightarrow C; \quad f(d) = c,$$

te creëren dat in staat is om het correcte label toe te kennen aan een tekstdocument d dat deel uitmaakt van een grotere collectie $D = \{d_1, \dots, d_n\}$. Dit model dient de (onbekende) targetfunctie $\tilde{f} : D \rightarrow C$, die de juiste manier aangeeft waarop de documenten gelabeld dienen te worden, zo goed mogelijk te benaderen.

Voordat men begint met het daadwerkelijk categoriseren van de tekstdocumenten dient er eerst een classificatiemodel opgesteld te worden. Dit model wordt gegenereerd aan de hand van een bepaalde classificatietechniek, die een algoritme toepast op de collectie van tekstdocumenten om de relatie tussen de woorden (features) en de voorgedefinieerde categorieën te modelleren. Het model dat gecreëerd wordt door het betreffende algoritme moet in staat zijn om nieuwe tekstdocumenten met een hoge nauwkeurigheid in de juiste categorie te classificeren. Deze modellen moeten dus goed in staat zijn om te generaliseren.

Het classificatiemodel wordt aan de hand van een trainingsset opgesteld. Deze trainingsset $D_{train} = \{d_1, \dots, d_n\}$ bestaat uit een verzameling tekstdocumenten die reeds voorzien zijn van het

correcte label $c \in C = \{c_1, \dots, c_{|C|}\}$ dat aangeeft tot welke categorie het document behoort. Elk tekstdocument behoort maar tot één categorie en krijgt dus maximaal één label toegekend. Dit wordt ook wel single-label classificatie genoemd [35]. Het geval waarin aan één tekstdocument 0 tot $|C|$ categorieën toegekend kunnen worden, wordt multilabel classificatie genoemd.

Het opgestelde classificatiemodel wordt vervolgens toegepast op een testset D_{test} dat bestaat uit tekstdocumenten waarvan het label voorspeld dient te worden door het classificatiemodel. Op basis van deze testset worden de prestaties van het model vastgesteld. Deze testset mag niet gebruikt worden tijdens het trainen van het classificatiemodel.

Het labelen van de documenten in de trainingsset wordt vaak met de hand gedaan. Hiervoor leest men alle tekstdocumenten in de trainingsset door en wordt aan de hand hiervan een lijst met categorieën opgesteld. Op basis van deze lijst worden alle documenten voorzien van een label dat aangeeft tot welke categorie het betreffende document behoort. Dit proces kan erg tijds- en arbeidsintensief zijn indien de trainingsset erg groot is (meer dan 1.000 tekstdocumenten). Daarom kan er in theorie ook gebruik gemaakt worden van ongelabelde tekstdocumenten om de prestatie van het classificatiemodel (ook wel classifier genoemd) op te schroeven. De ongelabelde tekstdocumenten zijn namelijk heel goedkoop te verkrijgen. Zo wordt in het artikel van Nigam et al. een algoritme besproken dat gebruikt kan worden om een classifier te trainen op basis van gelabelde en ongelabelde tekstdocumenten¹⁴. Dit algoritme combineert hiervoor Expectation-Maximization met een Naïve Bayes classifier. De uitgevoerde experimenten laten zien dat het gebruik van de ongelabelde tekstdocumenten de classificatie error tot 33% kan verminderen.

4.3 CLASSIFICATIETECHNIKEN

4.3.1 NAIVE BAYES

Formule van Bayes

De methode van Naïve Bayes is een methode waarin waarschijnlijkheid en statistiek centraal staan en is gebaseerd op de theorie van Thomas Bayes, een wiskundige die leefde in de 18^e eeuw. Bayes heeft destijds een formule ontwikkeld die aangeeft op welke manier de kans op een bepaalde gebeurtenis A wordt beïnvloed door een andere gebeurtenis B:

$$P(A|B) = \frac{P(B|A)P(A)}{p(B)}. \quad (4.1)$$

Deze formule wordt de formule van Bayes genoemd (in de literatuur staat deze formule bekend als Bayes' theorem).

De Naïve Bayes classifier toegepast op discrete attributen

De Naïve Bayes classifier gaat uit van de aanname dat de woorden w_1, \dots, w_{n_i} die voorkomen in een document d_i gegeneerd zijn aan de hand van een probabilistisch mechanisme en dat de categorie $c(d_i)$ van een tekstdocument d_i een bepaalde relatie heeft met de woorden die in het tekstdocument voorkomen [15]. Dit kan wiskundig met de voorwaardelijke kans $p(c_i | w_1, \dots, w_{n_i})$ beschreven worden. Deze kans geeft aan hoe waarschijnlijk het is dat het document d_i behoort tot categorie c_i gegeven de

¹⁴ Nigam, K., A. McCallum, S. Thrun, T. Mitchell (1998), learning to classify text from labeled and unlabeled documents. In *Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence*, pp. 792–799.

informatie dat de woorden w_1, \dots, w_{n_i} in het document voorkomen. Volgens de formule van Bayes kunnen wij deze voorwaardelijke kans herschrijven als:

$$p(c_i | w_1, \dots, w_{n_i}) = \frac{p(w_1, \dots, w_{n_i} | c_i) p(c_i)}{p(w_1, \dots, w_{n_i})}. \quad (4.2)$$

De kans $p(c_i)$ geeft aan wat de a priori kans is dat een document behoort tot categorie c_i . Meestal kiest men ervoor om deze kans gelijk te stellen aan de relatieve frequentie waarmee de categorie c_i voorkomt in de trainingsset

$$p(c_i) = \frac{\text{Het aantal keer dat categorie } c_i \text{ voorkomt in de trainingsset}}{|D|}. \quad (4.3)$$

De conditionele kans die is weergegeven in formule (4.2) is de a posteriori kans dat een document die de woorden w_1, \dots, w_{n_i} bevat behoort tot categorie c_i . Een tekstdocument wordt toegekend aan de categorie die de grootste a posteriori kans heeft. Aangezien de term in de noemer van formule (4.3) een normalisatiefactor is, kan men deze bij het berekenen van de a posteriori kans weglaten. In plaats daarvan normaliseert men de resulterende kansen zodanig dat de som van de kansen $p(c_i | w_1, \dots, w_{n_i})$ over alle mogelijke categorieën c_i gelijk is aan 1.

Het berekenen van de a posteriori kans met behulp van formule (4.2) is echter niet praktisch. Dit komt doordat een schatting van de kans $p(w_1, \dots, w_{n_i} | c_i)$ alleen verkregen kan worden indien wij beschikken over een zeer grote trainingsset. De reden hiervoor is dat er vele verschillende combinaties van w_1, \dots, w_{n_i} mogelijk zijn. Daarnaast dienen wij elke combinatie ook verschillende keren tegen te komen in de trainingsset om een goede schatting te verkrijgen van deze kans.

Om het bovenstaande probleem te verhelpen is het gebruikelijk om aan te nemen dat een bepaald woord, dat voorkomt in een tekstdocument dat behoort tot een bepaalde categorie, onafhankelijk voorkomt van de overige woorden in het tekstdocument. Deze “naïeve” aanname stelt ons in staat om de kans $p(w_1, \dots, w_{n_i} | c_i)$ te schrijven als: $p(w_1, \dots, w_{n_i} | c_i) = \prod_{j=1}^{n_i} p(w_j | c_i)$. De kansen $p(w_j | c_i)$ zijn veel makkelijker te schatten uit de data. Dit leidt uiteindelijk tot de volgende definitie van de Naïve Bayes classifier:

$$c_{NB} = \arg \max_{c_i \in C} p(c_i) \prod_{j=1}^{n_i} p(w_j | c_i), \quad (4.4)$$

waarbij c_{NB} de categorie voorstelt waartoe het document volgens de Naïve Bayes classifier met de grootste kans behoort.

De Naïve Bayes classifier toegepast op continue attributen

Het toepassen van de dimensie-reductie technieken die beschreven zijn in paragraaf 3.5 leidt ertoe dat de woorden, welke binaire attributen zijn, worden getransformeerd in continue numerieke attributen. Continue attributen worden in de Naïve Bayes classifier echter op een andere manier behandeld dan discrete attributen. In de Naïve Bayes classifier wordt namelijk de aanname gemaakt dat de continue attributen binnen elke categorie een normale verdeling hebben [21]. De normale verdeling wordt gekarakteriseerd door twee parameters: het gemiddelde μ en de variantie σ^2 . Deze parameters kunnen eenvoudig uit de data geschat worden. De normale verdeling stelt ons in staat om de kans $p(x_1, \dots, x_{n_i} | c_i)$ als volgt te schrijven:

$$p(x_1, \dots, x_{n_i} | c_i) = g(x; \mu_{c_i}, \sigma_{c_i}^2) \quad (4.5)$$

waarbij

$$g(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}. \quad (4.6)$$

de dichtheidsfunctie van een normale verdeling is¹⁵.

Men kan gebruik maken van “kernel density estimation” indien men op voorhand vermoedt dat de continue attributen niet normaal verdeeld zijn. In dit geval wordt er geen specifieke verdeling voor de continue attributen verondersteld, maar wordt deze met behulp van “kernels” geschat uit de data [21].

Om het bovenstaande proces te verduidelijken beschouwen wij een kleine dataset die bestaat uit twee categorieën (+ en -), een discreet attribuut X_1 die twee waarden kan aannemen a en b , en een continu attribuut X_2 . Op basis van de trainingsset $\{(+, a, 1.0), (+, b, 2.5), (+, a, 3.0), (-, b, 4.5), (-, b, 5.0)\}$ worden de volgende schattingen geproduceerd door de Naïve Bayes classifier:

$$\begin{aligned} p(C=+) &= 3/5; p(C=-) = 2/5 \\ p(X_1 = a | C=+) &= 2/3; p(X_1 = a | C=-) = 0 \\ p(X_1 = b | C=+) &= 1/3; p(X_1 = b | C=-) = 1 \\ p(X_2 = x | C=+) &= g(x, 2.167, 1.083); p(X_2 = x | C=-) = g(x, 4.75, 0.125). \end{aligned}$$

De “Naïve” aanname dat woorden in een tekstdocument onafhankelijk voorkomen van de overige woorden in het betreffende tekstdocument is onrealistisch. De kans is bijvoorbeeld vele male groter dat het woord filter in een mail voorkomt indien het woord spam ook in deze mail voorkomt. Ondanks deze foutieve aanname blijkt de Naïve Bayes classifier in de praktijk goed te werken. Het is een zeer efficiënte techniek die een korte rekentijd heeft en relatief weinig computergeheugen vereist. Feit is wel dat uit de wetenschappelijke literatuur blijkt dat de prestatie van de Naïve Bayes classificatietechniek op het gebied van tekstmining enigszins achterblijft bij de andere technieken [15]. Desalniettemin wordt de Naïve Bayes classifier vanwege zijn eenvoud en efficiëntie verreweg het meest toegepast in real-life applicaties.

4.3.2 K-NEAREST NEIGHBOR

k -Nearest Neighbor behoort tot de categorie van “lazy learners” omdat zij de categorie waartoe een tekstdocument behoort pas bepalen indien het document ter classificatie wordt aangeboden aan het algoritme. Tijdens de trainingsfase stelt deze classifier geen model op zoals dat wel het geval is bij de Naïve Bayes classifier, maar worden alle tekstdocumenten met de bijbehorende labels simpelweg opgeslagen in een interne datastructuur. Om een tekstdocument te classificeren wordt het document vergeleken met de k documenten uit de trainingsset die er het meest op lijken. Deze tekstdocumenten worden de k “nearest neighbors” genoemd. De categorie die het meest voorkomt onder deze k tekstdocumenten vormt dan tevens de categorie van het nieuwe tekstdocument. In de literatuur worden

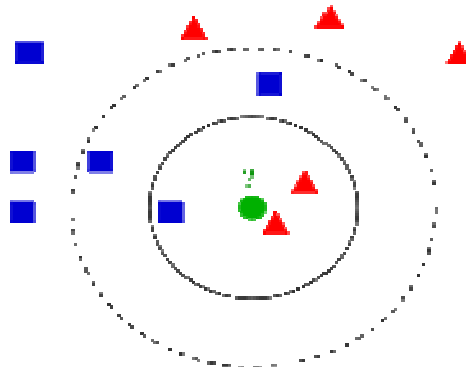
¹⁵ Vergelijking 4.5 is strikt genomen niet correct omdat de kans dat een reële random variabele exact gelijk is aan een willekeurige waarde gelijk is aan nul. In dit geval maken wij gebruik van een integraal die aangeeft wat

de kans is dat de random variabele in een bepaald interval ligt: $p(x \leq X \leq x + \Delta) = \int_x^{x+\Delta} g(x; \mu, \sigma^2)$. Volgens de

definitie van een afgeleide geldt nu $\lim_{\Delta \rightarrow 0} p(x \leq X \leq x + \Delta) / \Delta = g(x; \mu, \sigma^2)$. Dus voor een zeer kleine constante Δ geldt $p(X = x) = g(x; \mu, \sigma^2) \cdot \Delta$. De constante Δ valt uiteindelijk weg tijdens de normalisatie waardoor het gebruik van vergelijking 4.5 gerechtvaardigd is [21].

er verschillende varianten van dit algoritme besproken. Zo is er de “distance-weighted” variant waarin de gelijkenis die de neighbors met het tekstdocument vertonen wordt meegewogen bij het bepalen van de categorie [31].

De werkwijze van het k-Nearest Neighbor algoritme kan aan de hand van figuur 4.1 nader toegelicht worden. In deze figuur stelt de groene cirkel een testinstantie voor die geclassificeerd moet worden in een van de volgende categorieën: driehoek of vierkant. Indien wij de 3 dichtstbijzijnde burens bekijken ($k = 3$) en ons dus alleen beperken tot de instanties die voorkomen binnen de kleinste cirkel (weergegeven met een zwarte doorgetrokken streep), dan dient de testinstantie geclassificeerd te worden in de categorie van driehoeken omdat in deze cirkel 2 driehoeken voorkomen en maar 1 vierkant. Bekijken wij echter de 5 dichtstbijzijnde burens ($k = 5$) dan dient de testinstantie geclassificeerd te worden in de categorie van vierkanten omdat 3 van de 5 dichtstbijzijnde burens vierkanten zijn tegen 2 burens die behoren tot de driehoeken. De volgende vraag rijst nu: stel wij bekijken de twee dichtstbijzijnde burens ($k = 2$) en wij komen tot de conclusie dat de ene buur behoort tot de categorie vierkant en de andere buurt tot de categorie driehoek. In welke categorie moeten wij dan de testinstantie classificeren. In deze gevallen is het gebruikelijk om aan de testinstantie de categorie van de dichtstbijzijnde buur toe te kennen.



Figuur 4.1: De werking van het k-Nearest Neighbor algoritme nader toegelicht.
Bron: [http://en.wikipedia.org/wiki/Nearest_neighbor_\(pattern_recognition\)](http://en.wikipedia.org/wiki/Nearest_neighbor_(pattern_recognition)).

De prestatie van de k-Nearest Neighbor classifier valt of staat met de maat die gebruikt wordt om de gelijkenis tussen twee verschillende documenten te bepalen. Er is een grote hoeveelheid aan rekenkundige maten beschikbaar in het gebied van tekstmining waarmee de gelijkenis tussen twee tekstdocumenten bepaald kan worden [15]. Een mogelijkheid is om simpelweg alle gemeenschappelijke woorden te tellen. Dit aantal dient dan wel genormaliseerd te worden om zo het effect dat de lengte van verschillende tekstdocumenten heeft op dit aantal teniet te doen. De kans is namelijk groot dat lange documenten onderling meer identieke woorden zullen bevatten dan korte documenten. Aan de andere kant hebben woorden een verschillende informatieve waarde waarmee op deze manier geen rekening wordt gehouden. Een manier om wel met dit gegeven rekening te houden is door gebruik te maken van de cosinus gelijkenismaat. De cosinus tussen twee vectoren \vec{d}_1 en \vec{d}_2 met dimensie m is gedefinieerd als het inwendige product van de twee vectoren:

$$\cos(d_1, d_2) = \frac{\sum_{i=1}^m d_1(i) \cdot d_2(i)}{\|d_1\| \cdot \|d_2\|}. \quad (4.7)$$

De cosinus meet feitelijk de hoek tussen de twee documenten. Des te kleiner deze hoek, des te groter de overeenkomst tussen de twee tekstdocumenten. Een andere veel gebruikte maat is de Euclidische afstand tussen twee tekstdocumenten:

$$\text{afstand}(d_1, d_2) = \sqrt{\sum_{i=1}^m (d_1(i) - d_2(i))^2}. \quad (4.8)$$

Een beschrijving van andere veel gebruikte maten kan gevonden worden in [1].

k -Nearest Neighbors vereist dat het aantal neighbors k op voorhand gespecificeerd wordt. Met behulp van een validatieset en n -fold cross-validation (zie paragraaf 3.4.1) kan de optimale k bepaald worden. Het is ook goed mogelijk dat deze waarde vooraf bepaald kan worden. In de wetenschappelijke literatuur worden afhankelijk van de gebruikte dataverzamelingen verschillende waarden voor k gebruikt. Zo gebruiken Larkey en Croft in [23] $k = 20$, terwijl in het geval van Yang [40] $30 \leq k \leq 45$ het effectiefst blijkt te zijn.

k -Nearest Neighbor is een van de best presterende tekst classifiers. Het voordeel van dit algoritme is dat het niet vereist dat de categorieën lineair scheidbaar zijn. Een nadeel is wel dat het algoritme een relatief hoge rekentijd heeft omdat voor elk document, dat ter classificatie aangeboden wordt, de gelijkenis berekend dient te worden met elk ander tekstdocument in de trainingsset. Deze berekeningen vergen daarnaast ook veel computergeheugen. Er zijn echter wel implementaties beschikbaar die gebruik maken van efficiënte indexeringsstechnieken die ervoor zorgen dat er minder berekeningen uitgevoerd dienen te worden.

4.3.3 SUPPORT VECTOR MACHINES

Het Support Vector Machine algoritme is zeer accuraat in het classificeren van tekstdocumenten. Alhoewel het een gecompliceerd algoritme is dat in principe veel tijd kost om uit te voeren, bestaan er inmiddels verschillende efficiënte implementaties die een kleine rekentijd hebben. De afgelopen jaren is dit algoritme meerdere keren met succes toegepast op verschillende classificatietaken [9, 20 en 24]. Zoals gewoonlijk wordt ook in dit geval een tekstdocument d voorgesteld door een (mogelijk gewogen) vector $\vec{w}_d = (w_{d1}, \dots, w_{dN})$. Een enkele Support Vector Machine kan alleen twee categorieën scheiden: een positieve categorie c_1 (aangegeven met $y = +1$) en een negatieve categorie c_2 (aangegeven met $y = -1$). In de vectorruimte waarin de tekstdocumenten worden weergegeven kan nu een vlak gedefinieerd worden door in de volgende lineaire vergelijking y gelijk te stellen aan 0:

$$y = f(\vec{w}_d) = b_0 + \sum_{j=1}^N b_j w_{dj}. \quad (4.9)$$

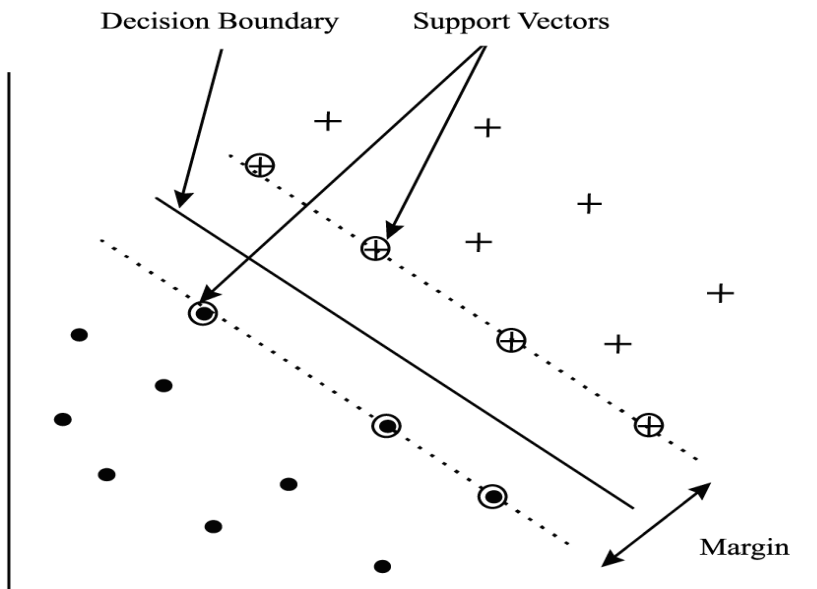
Het SVM algoritme bepaalt een vlak dat tussen de positieve en negatieve instanties in de trainingsset gelokaliseerd is (dit is de zogenaamde decision boundary in figuur 4.3). De parameters b_j worden zodanig bepaald dat de afstand (ook wel margin genoemd) tussen het scheidende vlak en de dichtstbijzijnde positieve trainingsinstanties en negatieve trainingsinstanties gemaximaliseerd wordt, zoals is weergegeven in figuur 4.3. Dit komt neer op het oplossen van een kwadratisch optimalisatie probleem met restricties dat voor een grote trainingsset efficiënt uitgevoerd kan worden.

De documenten die het dichtst bij het scheidende vlak liggen worden de support vectors genoemd. Deze support vectors bepalen de daadwerkelijke locatie van het vlak. Meestal is alleen een klein deel van de tekstdocumenten in de trainingsset een support vector. De rest van de trainingsdocumenten hebben geen invloed op de getrainde classifier. In dit opzicht is het SVM algoritme uniek vergeleken met de andere classificatiealgoritmen die beschikbaar zijn binnen de tekstmining [31].

Een nieuw document dat gerepresenteerd wordt door de vector \vec{w}_d wordt geclassificeerd in categorie c_1 indien de waarde $f(\vec{w}_d) > 0$ is en anders in categorie c_2 . Indien er geen vlak geconstrueerd kan worden

dat in staat is om de positieve en negatieve trainingsinstanties lineair van elkaar te scheiden, wordt er een vlak gecreëerd zodanig dat het aantal trainingsinstanties dat zich aan de verkeerde kant van het vlak bevindt geminimaliseerd wordt. De SVM classifier is door middel van enige transformaties ook in staat om meerdere categorieën van elkaar te onderscheiden.

De belangrijkste eigenschap van de SVM classifier is dat het trainen van deze classifier voor het grootste deel onafhankelijk is van het aantal dimensies in het vectorruimte model. Het vereist zelden featureselectie omdat het in staat is om zelf die trainingsinstanties (support vectors) te selecteren die benodigd zijn voor een goede classificatie. Hierdoor treedt overfitting bij de SVM classifier nauwelijks op wat deze classifier uitermate geschikt maakt voor de classificatie van tekstdocumenten. Een nadeel is dat de SVM classifier erg veel parameters heeft waarvan de invloed op de werking van de classifier onbekend is.



Figuur 4.3: Een Support Vector Machine (oftewel decision boundary) in een twee dimensionale ruimte.
Bron: <http://www.emeraldinsight.com/fig/0701041001002.png>

4.3.4 BOOSTEXTER

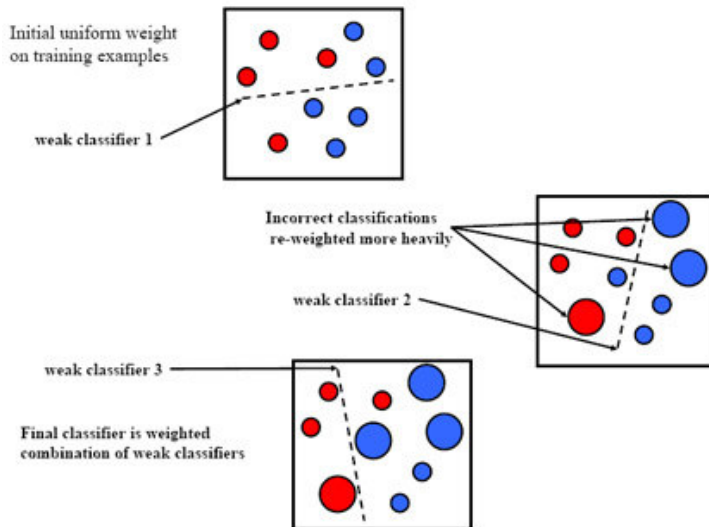
Boosting is een methode die de kwaliteit van de classificaties dient te verhogen door verschillende zwakke hypothesen (of classifiers) die een lage nauwkeurigheid hebben te combineren tot een enkele regel, ook wel de gecombineerde hypothese genoemd (zie figuur 4.4). Het kenmerk van boosting is dat de zwakke classifiers na elkaar worden getraind waarbij er een set van gewichten over zowel de verzameling van trainingsdocumenten als de bijbehorende labels wordt bijgehouden. Tijdens het boosten krijgen de trainingsdocumenten die moeilijk te classificeren zijn samen met de bijbehorende labels een steeds groter gewicht en krijgen de trainingsdocumenten en de bijbehorende labels die makkelijk te classificeren zijn een steeds lager gewicht. Het doel is om de zwakke classifiers te dwingen zich te concentreren op de trainingsdocumenten en labels die het moeilijkst te classificeren zijn en de grootste bijdrage leveren aan het vinden van een zeer nauwkeurige classifier.

Er bestaan verschillende varianten van boosting. Het boostingalgoritme dat in deze paragraaf wordt beschreven is gebaseerd op het artikel van Schapire & Singer [33]. In dit artikel worden vier verschillende versies van dit algoritme besproken. Wij zullen ons in deze paragraaf hoofdzakelijk concentreren op het *real AdaBoost.MH* algoritme.

Voordat wij het algoritme gaan beschrijven zullen wij eerst enkele notaties invoeren die ook gebruikt worden in [33]. Laat D de verzameling van tekstdocumenten zijn en laat C de eindige verzameling van voorgedefinieerde categorieën zijn waarvan de grootte wordt aangegeven met $k = |C|$. In [33] wordt ervan uitgegaan dat een document tot verschillende categorieën kan behoren en dus meerdere labels kan hebben. Een gelabeld tekstdocument wordt aangegeven met (d, Y) , waarbij $Y \subseteq C$ de verzameling van labels voorstelt die zijn toegekend aan tekstdocument d . In het speciale geval dat elk tekstdocument maar 1 label heeft geldt dat $|Y| = 1$ voor alle tekstdocumenten. Voor $Y \subseteq C$ definiëren wij daarnaast als volgt de functie $Y[l]$ voor $l \in Y$:

$$Y[l] = \begin{cases} +1 & \text{als } l \in Y \\ -1 & \text{als } l \notin Y \end{cases} \quad (4.10)$$

Het algoritme dat in deze paragraaf wordt beschreven produceert voor elk tekstdocument een ranking van alle mogelijke labels waarbij de meest geschikte labels aan de top van de ranking verschijnen. Er wordt een functie van de vorm $f : D \times C \rightarrow R$ geproduceerd met de eigenschap dat voor een label l_1 dat op een hogere plaats in de ranking staat dan het label l_2 geldt dat $f(d, l_1) > f(d, l_2)$. Als Y de verzameling van labels is die hoort bij tekstdocument d dan zal een succesvol algoritme in staat zijn om de labels die in Y voorkomen hoger te plaatsen in de ranking dan de labels die niet in Y voorkomen.



$$H(x) = \text{sign}(\alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x))$$

Figuur 4.4: Schematische weergave van Boosting.

Wij beschrijven als eerst het *AdaBoost.MH* algoritme:

Gegeven: $(d_1, Y_1), \dots, (d_m, Y_m)$ waar $d_i \in D, Y_i \subseteq C$.

Initialiseer $D_1(i, l) = \frac{1}{m \cdot |C|}$.

For $t = 1, \dots, T$:

- Geef verdeling D_i door aan de zwakke classifier.
- Stel de beste zwakke hypothese op: $h_t : D \times C \rightarrow R$.
- Update:

$$D_{t+1}(i,l) = \frac{D_t(i,l) \exp(-\alpha_t Y_i[l] h_t(x_i, l))}{Z_t} \quad (4.11)$$

waarbij Z_t een normalisatiefactor is die zodanig gekozen wordt dat D_{t+1} een distributie voorstelt.

- Retourneer de gecombineerde hypothese:

$$f(d,l) = \sum_{t=1}^T \alpha_t h_t(d,l). \quad (4.12)$$

Laat S een reeks van trainingsdocumenten zijn met bijbehorende labels $\langle (d_1, Y_1), \dots, (d_m, Y_m) \rangle$ waarbij $d_i \in D$ en $Y_i \subseteq C$. In het bovenstaande algoritme stelt D_t de verdeling van de gewichten over de trainingsdocumenten en de bijbehorende labels voor. Aan de start van het algoritme zijn deze gewichten uniform verdeeld. In elke iteratie $t \in T$ wordt deze verdeling samen met de reeks S doorgegeven aan de zwakke classifier die op basis hiervan een zwakke hypothese $h_t : D \times C \rightarrow R$ opstelt. Het teken van $h_t(d,l)$ wordt hierbij gezien als een voorspelling van het feit of het label wel of niet wordt toegekend aan het document. De grootte van de voorspelling $|h(d,l)|$ dient als maat voor de betrouwbaarheid van de voorspelling.

Hier opvolgend wordt een parameter α_t gekozen waarna de verdeling D_t wordt bijgewerkt. In het geval dat α_t positief is wordt de verdeling op een manier bijgewerkt die ervoor zorgt dat de gewichten van niet juist geclassificeerde document-label paren toeneemt. De gecombineerde hypothese produceert uiteindelijk voor elk tekstdocument een ranking van de labels op basis van een gewogen gemiddelde van de T zwakke hypotheses.

Tot nu toe zij wij nog niet ingegaan op de zwakke classifiers die in het AdaBoost algoritme gebruikt worden. De zwakke classifiers kunnen gezien worden als een beslisboom die test of een bepaald woord wel of niet in het document voorkomt. Op basis van de uitkomst van deze test retourneert de zwakke classifier voorspellingen en betrouwbaarheidsindicaties voor elke document-label paar. In formele notatie geeft w een bepaald woord aan, en betekent $w \in d$ dat woord w voorkomt in document d . De zwakke classifier h_w maakt voorspellingen van de volgende vorm:

$$h_w(d,l) = \begin{cases} c_{0l}, & \text{als } w \in d \\ c_{1l}, & \text{als } w \notin d \end{cases} \quad (4.13)$$

waarbij geldt dat $c_{0l}, c_{1l} \in R$.

In *real AdaBoost.MH* zijn c_{jl} ($j \in \{0,1\}$) reële waarden die voor een willekeurig woord w op de volgende manier berekend kunnen worden: laat $X_0 = \{d : w \notin d\}$ en $X_1 = \{d : w \in d\}$ (in woorden: X_0 is de verzameling van documenten waarin het woord w niet in voorkomt en X_1 is precies het tegenovergestelde hiervan). Gegeven de huidige verdeling D_t , berekenen wij voor elk mogelijk label l , voor $j \in \{0,1\}$ en voor $b \in \{-1,+1\}$:

$$W_b^{jl} = \sum_{i=1}^m D_t(i,l) \chi(d_i \in X_j \wedge Y_i[l] = b), \quad (4.14)$$

met

$$\chi(\pi) = \begin{cases} 1 & \text{als } \pi \text{ geldt} \\ 0 & \text{anders.} \end{cases} \quad (4.15)$$

Om de leesbaarheid te vergroten schrijven wij W_+^{jl} in plaats van W_{+1}^{jl} en W_-^{jl} in plaats van W_{-1}^{jl} . In woorden: W_+^{jl} (W_-^{jl}) is het gewicht (met respect tot de verdeling D_t) van de documenten in partitie X_j die wel (niet) het label l hebben. De waarden c_{jl} dienen zodanig gekozen te worden dat de functie Z_t geminimaliseerd wordt. Het kan aangetoond worden [32] dat Z_t voor een bepaalde term geminimaliseerd wordt indien

$$c_{jl} = \frac{1}{2} \ln \left(\frac{W_+^{jl}}{W_-^{jl}} \right) \text{ en } \alpha = 1. \quad (4.16)$$

Dit impliceert dat $Z_t = 2 \sum_{j \in \{0,1\}} \sum_{l \in C} \sqrt{W_+^{jl} W_-^{jl}}$ waaruit volgt dat wij kiezen voor de term w waarvoor de waarde van Z_t het kleinst is als zwakke hypothese in iteratie t . Indien W_+^{jl} of W_-^{jl} heel klein of gelijk aan nul zijn kan de term c_{jl} erg groot worden. Dit kan voor numerieke problemen zorgen en kan de neiging tot overfitting vergroten. In de praktijk wordt er daarom gebruik gemaakt van zogenaamde “gladde” waarden:

$$c_{jl} = \frac{1}{2} \ln \left(\frac{W_+^{jl} + \varepsilon}{W_-^{jl} + \varepsilon} \right). \quad (4.17)$$

In de experimenten van Schapire en Singer [33] wordt ε gelijk gesteld aan $1/m \cdot |C|$.

Men kan aantonen dat boosting sterk gerelateerd is aan Support Vector Machines omdat boosting ook de margin tussen de trainingsinstanties maximaliseert. Om deze reden is boosting, net als Support Vector Machines, ook resistent tegen overfitting [31].

4.4 EVALUATIE VAN DE PRESTATIES VAN DE CLASSIFICATIETECHNIKEN

4.4.1 K-FOLD CROSS-VALIDATION EN LEAVE-ONE-OUT CROSS-VALIDATION

De prestaties van de classificatietechnieken worden vaak experimenteel geëvalueerd [31]. Een methode die hier vaak voor gebruikt wordt is k -fold cross-validation. Hierbij wordt de gehele collectie van tekstdocumenten random ingedeeld in k gelijke delen. In elk deel komt de verdeling van de tekstdocumenten over de verzameling van categorieën ongeveer overeen met de verdeling van de tekstdocumenten in de gehele tekstcollectie. Hierna wordt het model getraind op $k-1$ delen. Het deel dat niet gebruikt is tijdens de training wordt gebruikt om de nauwkeurigheid van het model te bepalen. In totaal wordt het model k maal getraind en wordt elk deel van de data 1 keer gebruikt voor de nauwkeurigheidsbepaling. Het gemiddelde van deze k nauwkeurigheidsschattingen vormt dan een schatting van de werkelijke nauwkeurigheid van het model. In de praktijk maakt men bijna altijd gebruik van 5 of 10-fold cross-validation.

Indien de beschikbare collectie van tekstdocumenten niet groot is, kan men ervoor kiezen om gebruik te maken van leave-one-out cross validation. Leave-one-out cross-validation is niets anders dan n -fold

cross-validation waarbij n het aantal documenten in de gehele tekstcollectie voorstelt. Het model wordt dus getraind op $n-1$ tekstdocumenten en de nauwkeurigheid van het model wordt vastgesteld met het niet gebruikte document: een 1 betekent dat het document correct is geïdentificeerd en een 0 betekent dat het foutief is geïdentificeerd. Dit levert uiteindelijk een rij met een lengte van n op, die bestaat uit nullen en enen, die voor elk van de n documenten aangeeft of het goed dan wel is fout geïdentificeerd. Het gemiddelde van deze rij vormt dan een schatting van de nauwkeurigheid van het model. Met behulp van deze methode kan het maximale uit een tekstcollectie gehaald worden om een zo'n goed mogelijke nauwkeurigheidsschatting te verkrijgen.

4.4.2 PRESTATIEMATEN

De meest eenvoudige maat die gebruikt kan worden om de prestatie van een getraind model aan te geven is de nauwkeurigheid, welke gedefinieerd is als het aantal correct geïdentificeerde documenten als fractie van het totaal aantal geïdentificeerde documenten. Deze prestatie maat geeft in sommige gevallen een vertekend beeld. Stel bijvoorbeeld dat wij beschikken over een collectie die bestaat uit 100 documenten waarvan 10 documenten behoren tot categorie A en 90 documenten behoren tot categorie B. Een classifier kan nu een nauwkeurigheid halen van 90% door simpelweg alle documenten te classificeren in categorie B. Iemand die echter voornamelijk geïnteresseerd is in de documenten die behoren tot categorie A zal weinig waarde hechten aan deze classifier omdat hij niet in staat is om de tekstdocumenten die behoren tot categorie A te onderscheiden van de documenten die behoren tot categorie B, ondanks de hoge nauwkeurigheid van de classifier.

In de praktijk wordt om deze reden gebruik gemaakt van andere prestatie maten. De meest voorkomende zijn precision en recall. Precision geeft de fractie aan van het totaal aantal documenten dat geïdentificeerd is in een bepaalde categorie en daadwerkelijk behoort tot die categorie:

$$\text{precision} = \frac{\text{aantal documenten dat correct geïdentificeerd is in categorie } c_i}{\text{totaal aantal documenten dat geïdentificeerd is in categorie } c_i}. \quad (4.18)$$

Stel bijvoorbeeld dat 100 documenten in categorie A geïdentificeerd zijn waarvan 80 documenten ook daadwerkelijk behoren tot deze categorie en de overige 20 documenten behoren tot een andere categorie. De precision voor categorie A is in dit geval gelijk aan $80/100 = 0.8$.

Recall geeft de fractie van het totaal aantal documenten aan dat behoort tot een bepaalde categorie en ook daadwerkelijk in deze categorie is geïdentificeerd (m.a.w. de fractie documenten die behoren tot een bepaalde categorie c_i en die door de classifier zijn teruggevonden):

$$\text{recall} = \frac{\text{aantal documenten dat correct geïdentificeerd is in categorie } c_i}{\text{totaal aantal documenten dat behoort tot categorie } c_i}. \quad (4.19)$$

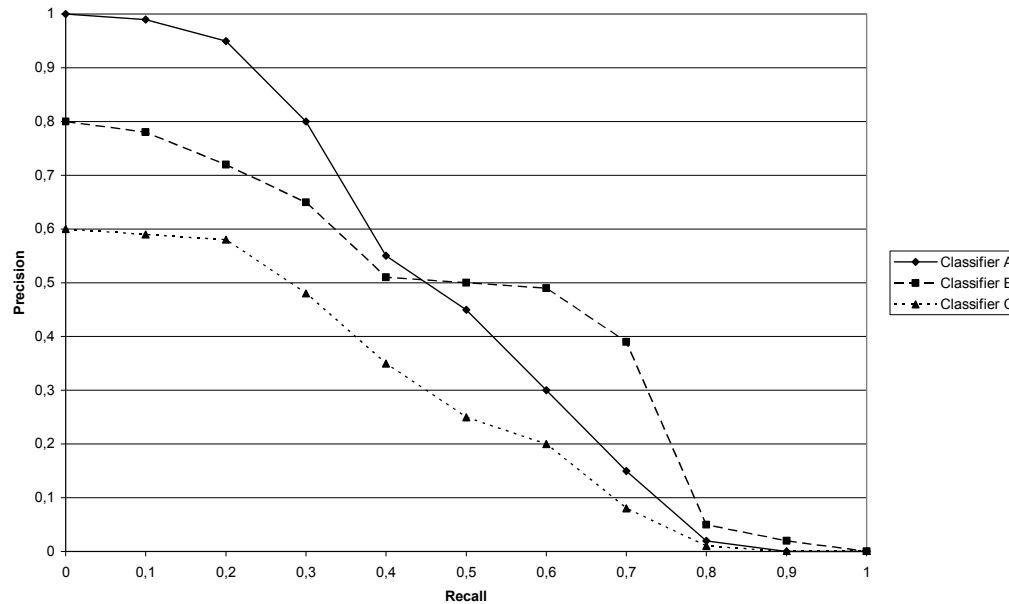
Stel dat er 50 documenten zijn die behoren tot categorie B en de betreffende classifier is in staat om 30 van deze 50 tekstdocumenten correct te classificeren in deze categorie. De recall voor categorie B is in dit geval gelijk aan $30/50 = 0.6$.

Natuurlijk bestaat er een wisselwerking tussen deze twee prestatie maten. Zo is het in sommige classificatiemodellen mogelijk om de precision te verhogen ten koste van de recall of andersom, door aanpassing van de drempel (threshold) van het betreffende classificatiemodel. Een voorbeeld van deze wisselwerking is in figuur 4.5 weergegeven. Uit deze figuur blijkt dat classifier A de hoogste precisie heeft die gecombineerd wordt met een zeer lage recall. Classifier B heeft daarentegen de hoogste precisie voor de hoogste recall waarden. Classifier C is in beide opzichten duidelijk ondergeschikt aan classifier A en B. Wij kunnen echter geen duidelijk onderscheid maken tussen classifier A en B. De keuze tussen deze twee classifiers is namelijk afhankelijk van de recall die wij vereisen.

De F_1 -score is een prestatie maat die de recall en precisie combineert om zo de overall prestatie van het model te bepalen:

$$F_1 = \frac{2}{1/\text{recall} + 1/\text{precision}}. \quad (4.20)$$

In [35] kan een uitgebreide beschrijving gevonden worden van andere (veelgebruikte) prestatie maten.



Figuur 4.5: Een grafische weergave van de wisselwerking tussen recall en precisie.

4.4.3 ONDERLINGE VERGELIJKING VAN DE CLASSIFICATIETECHNIKEN

De vraag welke classifier het beste is kan moeilijk beantwoord worden omdat de resultaten die in verschillende wetenschappelijke publicaties vermeld worden moeilijk met elkaar vergeleken kunnen worden doordat de onderzoeken in verschillende experimentele omgevingen zijn uitgevoerd. Volgens de meeste onderzoekers zijn SVM, AdaBoost, kNN en regressiemethoden de best presterende classifiers [31]. Er is echter onvoldoende statistisch bewijs om de beste van deze classifiers aan te wijzen.

Rocchio en Naïve Bayes presteren onder alle classifiers het slechtst, maar worden wel vaak gebruikt in boosting algoritmes of als een lid van een groter classificatie comité. De resultaten wat neurale netwerken en beslisbomen betreft zijn gemixt. Sommige experimenten hebben aangetoond dat deze methoden een zwakke prestatie leveren terwijl in andere experimenten deze technieken dezelfde nauwkeurigheid behalen als Support Vector Machines¹⁶.

¹⁶ Een goede beschrijving van de technieken: Rocchio, neurale netwerken en beslisbomen kan gevonden worden in [35].

HOOFDSTUK 5

HET CLUSTEREN VAN TEKST

5.1 INLEIDING

Clustering is een ongesuperviseerd proces waarin tekstdocumenten geïnclassificeerd worden in homogene groepen die clusters worden genoemd [31]. In hoofdstuk 4 hebben wij het automatische tekstclassificatie proces beschreven waarin men de beschikking heeft over een op voorhand gelabelde verzameling van trainingsdocumenten. Het doel is om aan de hand van deze verzameling de beschrijvingen van de verschillende categorieën te “leren” op basis waarvan nieuwe tekstdocument geïnclassificeerd kunnen worden in een van de voorgedefinieerde groepen. In het geval van clustering zijn de documenten niet voorzien van een label en is het de bedoeling om deze ongelabelde tekstdocumenten te groeperen zonder gebruik te maken van enige aanvullende informatie. De labels die op basis van deze clustering aan de tekstdocumenten worden toegekend zijn dus volledig verkregen uit de data en zijn dus niet op voorhand opgesteld door de gebruiker.

In dit hoofdstuk zullen wij beschrijven hoe clustering gebruikt kan worden in een tekstuele context. In de volgende paragraaf zal eerst het algemene clusteringprobleem omschreven worden waarna in paragraaf 5.3 verschillende clusteringtechnieken aan bod zullen komen. Hier opvolgend zal in paragraaf 5.4 aangegeven worden hoe de kwaliteit van de gevonden clustering geëvalueerd kan worden. In de laatste paragraaf zal tenslotte kort besproken worden op welke wijze de clusteringtechnieken toegepast kunnen worden op tekstdocumenten.

5.2 HET ALGEMENE CLUSTERINGSPROBLEEM

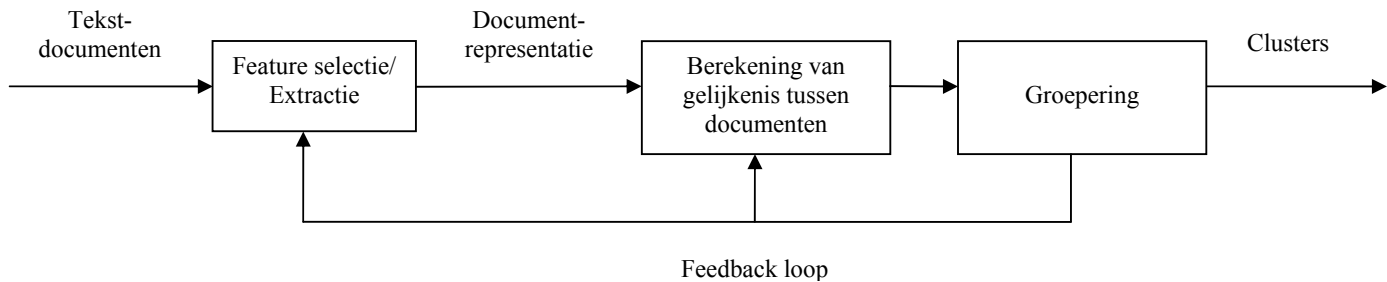
Volgens Jain en anderen in [18] bestaat het clusteringproces uit de volgende stappen:

- (1) Probleemrepresentatie inclusief het extraheren en/of selecteren van features.
- (2) Het selecteren van een maat waarmee de gelijkheid tussen verschillende objecten bepaald kan worden.
- (3) Het daadwerkelijk clusteren van de objecten.
- (4) Data-abstractie (indien gewenst), en
- (5) Evaluatie (indien gewenst).

Deze stappen zijn grafisch weergegeven in figuur 5.1. In deze figuur is ook een feedback stap opgenomen waarmee aangegeven wordt dat de resulterende clustering invloed kan hebben op de featureselectie en de berekening van de overeenkomsten tussen verschillende objecten.

5.2.1 PROBLEEMREPRESENTATIE

Alle clusteringproblemen zijn in essentie optimalisatieproblemen [31]. Het doel is om de beste groepering van de data te selecteren uit alle mogelijke groeperingen. In een goede clustering zijn de objecten die veel op elkaar lijken bij elkaar gegroepeerd en zijn de documenten die niet op elkaar lijken ver van elkaar verwijderd. Om de objecten te clusteren is een clusteringtechniek nodig die gebruik maakt van een functie die de gelijkheid tussen verschillende objecten kan bepalen. Daarnaast dient de clusteringtechniek op de hoogte te zijn van het feit dat documenten die veel overeenkomsten vertonen in dezelfde cluster geïnclassificeerd dienen te worden en dat documenten die weinig overeenkomsten vertonen aan verschillende clusters toegekend dienen te worden.



Figuur 5.1: Schematische weergave van het clusteringproces.

De functie die de overeenkomsten berekent kent aan twee objecten een reële waarde toe die aangeeft hoe groot de gelijkens is tussen deze objecten. Om deze overeenkomst te berekenen dient deze functie in staat te zijn om de interne structuur van de objecten met elkaar te kunnen vergelijken. Deze interne structuur bestaat uit een verzameling features¹⁷ die het betreffende object karakteriseren. Featureselectie en feature-extractie zijn technieken waarmee een geschikte set features gevonden kan worden die tijdens het clusteren gebruikt kan worden. Zoals reeds is beschreven in hoofdstuk 3 behelst feature selectie het selecteren van de meest effectieve subset van de originele features die gebruikt kan worden in de clustering. Feature-extractie is daarentegen het proces waarbij de originele features getransformeerd worden in nieuwe “synthetische” features.

5.2.2 GELIJKENISMATEN

In overeenstemming met het proces van tekstclassificatie maken wij ook in het geval van clustering gebruik van het vectorruimte model waarin tekstdocumenten worden voorgesteld als (gewogen) vectoren in een hoogdimensionale featureruimte. In het vectorruimte model is de gelijkensmaat meestal gebaseerd op de afstand tussen deze vectoren. Des te kleiner de afstand tussen twee vectoren, des te groter de overeenkomsten. Een populaire maat is ook hier de Euclidische afstand:

$$D(x_i, x_j) = \sqrt{\sum_m (x_{im} - x_{jm})^2}, \quad (5.1)$$

wat een specifiek geval is van de Minkowski functie ($p = 2$):

$$D_p(x_i, x_j) = \left(\sum_m |x_{im} - x_{jm}|^p \right)^{1/p}. \quad (5.2)$$

Volgens [25] werkt de Euclidische afstand goed indien de clusters in de dataset compact of geïsoleerd zijn. Het nadeel van de Minkowski functie is dat de grootst geschaalde feature de neiging heeft om de overige features te domineren.

De maat die het meest gebruikt wordt voor het clusteren van tekstdocumenten is echter de cosinus maat die de hoek meet tussen de twee tekstdocumenten [31]:

¹⁷ In de context van tekstdocumenten zijn features niets anders dan woorden.

$$\text{Sim}(x_i, x_j) = (x'_i \cdot x'_j) = \sum_k x'_{ik} \cdot x'_{jk}, \quad (5.3)$$

waarbij x' de genormaliseerde vector $x = x/|x|$ voorstelt¹⁸. Het grote voordeel van de cosinusmaat is dat hij onafhankelijk is van de lengte van de vectoren. Hierdoor is het mogelijk om vectoren die dezelfde compositie, maar verschillende totalen hebben, op dezelfde manier te behandelen. Dit verklaart waarom deze maat zo vaak gebruikt wordt voor tekstdocumenten.

Er zijn verschillende andere maten beschikbaar die elk hun eigen karakteristieke eigenschappen hebben. Een uitgebreide selectie kan gevonden worden in [18].

5.3 CLUSTERINGALGORITMEN

Er bestaan verschillende clusteringalgoritmen die op basis van hun specifieke kenmerken onderscheiden kunnen worden. Het eerste onderscheid dat gemaakt kan worden is tussen partitionele en hiërarchische clusteringmethoden. Een partitionele clusteringmethode produceert een enkele partitie van een verzameling objecten in verschillende groepen. Een hiërarchische clusteringmethode produceert een geneste reeks van partities. Beide vormen kunnen zowel een “harde” als “zachte” clustering produceren. In een harde clustering wordt elk object aan maximaal 1 cluster toegekend. In een “zachte” clustering kan elk object tot meerdere categorieën behoren waarbij een kansverdeling aangeeft wat de kans is dat het object behoort tot een specifiek cluster.

De hiërarchische clusteringmethodes kunnen verder opgedeeld worden in agglomeratieve (Engels: agglomerative) en divisieve (Engels: divisive) algoritmen. In het agglomeratieve algoritme wordt eerst elk object in één apart cluster ingedeeld waarna deze clusters succesvol worden samengevoegd tot grotere clusters, totdat aan een bepaald stopcriterium is voldaan. Divisieve algoritmen beginnen juist met één groot cluster waarin alle objecten zich bevinden, waarna deze opgedeeld wordt in steeds kleinere clusters totdat het stopcriterium is bereikt.

Alle clusteringproblemen zijn in essentie optimalisatieproblemen. Het doel is om aan de hand van een bepaalde criteriumfunctie de beste clustering van de objecten te kiezen uit alle mogelijke clusteringen. De criteriumfunctie geeft hierbij aan wat de kwaliteit is van de resulterende clustering. Een goed voorbeeld van een criteriumfunctie is de zogenaamde “sum of squared error (SSE)”. Om deze criteriumfunctie te berekenen wordt eerst de error van elk object bepaald, d.w.z. de afstand van het object tot het centrum van het dichtstbijzijnde cluster, waarna de som van de gekwadrateerde errors wordt berekend. De clustering waarvan de SSE het kleinst is heeft de voorkeur omdat dit betekent dat de clustercentra een goede representatie zijn van de objecten die deel uitmaken van het betreffende cluster.

Het kost echter veel rekentijd om het clustering optimalisatieprobleem op te lossen ongeacht het gekozen clusteringalgoritme. Het “brute-force” algoritme voor een harde partitionele clustering van n objecten in k clusters dient $k^n / k!$ mogelijke partities te evalueren [31]. Voor het groeperen van 100 objecten in 5 clusters dienen dus $5^{100} / 5! = 6.57 \cdot 10^{67}$ partities geëvalueerd te worden. Dit maakt het in de meeste gevallen onmogelijk om het optimalisatieprobleem exact op te lossen waardoor er in de praktijk vaak gebruik wordt gemaakt van één of ander “greedy” heuristiek.

De meest gebruikte clusteringalgoritmen zijn k-means (hard en partitioneel), het op Expectation Maximization gebaseerde mixture resolving algoritme (zacht en partitioneel) en het HAC algoritme (hiërarchisch en agglomeratief). In het vervolg van deze paragraaf zullen deze algoritmen nader beschreven worden.

¹⁸ $|x| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$ stelt de lengte van een vector $x = (x_1, x_2, \dots, x_n)$ voor. Een genormaliseerde vector heeft een lengte die gelijk is aan 1.

5.3.1 K-MEANS ALGORITME

Het k-means algoritme partitioneert een collectie van vectoren $\{x_1, x_2, \dots, x_n\}$ in een verzameling clusters $\{C_1, C_2, \dots, C_k\}$ zodanig dat de vectoren die een samenhang vertonen zijn ondergebracht in dezelfde clusters terwijl de ongerelateerde vectoren juist in verschillende clusters zijn ondergebracht [31]. Het algoritme vereist dat er op voorhand k initiële clusters worden gespecificeerd. Deze clusters kunnen random uit de verzameling van vectoren worden gekozen of kunnen op basis van a priori kennis worden gespecificeerd.

Het algoritme werkt als volgt:

- (1) Initialiseer k clustercentra random of op basis van a priori kennis.
- (2) Wijs elke vector toe aan het dichtstbijzijnde clustercentrum.
- (3) Herbereken de clustercentra M_i :

$$M_i = |C_i|^{-1} \sum_{x \in C_i} x, \quad (5.4)$$

en ken elke vector toe aan het (nieuwe) dichtstbijzijnde clustercentrum.

- (4) Als het convergentiecriteria niet bereikt is ga dan weer naar terug naar stap 2. Typische convergentiecriteria zijn: geen of minimale nieuwe toewijzingen van objecten aan nieuwe clusters of een minimale daling van de kwadratische fout.

Het doel van het k-means algoritme is om de gelijkheid tussen de vectoren in een cluster en de betreffende clustercentra te maximaliseren. In het geval dat de vectoren tekstdocumenten voorstellen wordt deze grootheid ook wel de cohesie van een cluster genoemd. De criteriumfunctie is dan gedefinieerd als:

$$\max \sum_{i=1}^k \sum_{x \in C_i} \cos(x, M_i),$$

waarbij de gelijkheid tussen een tekstdocument en het dichtstbijzijnde clustercentrum wordt bepaald middels de cosinus. De criteriumfunctie in (5.5) wordt ook wel de totale cohesie van een clustering genoemd¹⁹. In de literatuur is het algemeen bekend dat het k-means algoritme altijd convergeert naar een lokaal maximum van de criteriumfunctie [31].

Het k-means algoritme is populair vanwege zijn eenvoud en efficiëntie [31]. Een nadeel is de gevoeligheid van het k-means algoritme voor de initiële selectie van de k clustercentra wat een groot probleem vormt. Wanneer een slechte initiële selectie random wordt gekozen kunnen de resulterende clusters suboptimaal zijn. Een manier om dit probleem te omzeilen is door het algoritme verschillende keren uit te voeren waarbij er random steeds nieuwe clustercentra worden gekozen.

Een ander probleem vormt het bepalen van het aantal clusters k dat leidt tot de beste clustering. Wanneer dit aantal niet op voorhand bekend is kan men het algoritme verschillende malen uitvoeren voor verschillende waarden van k . Op basis van de gebruikte kwaliteitsfunctie kan dan de optimale k geselecteerd worden.

¹⁹ Bron: Tan, P., M. Steinbach en V. Kumar (2006), *Introduction To Data Mining*. Addison-Wesley, Pearson Education Inc.

5.3.2 EXPECTATION MAXIMIZATION CLUSTERINGALGORITMEN

De basis van statistische clustering is een statistisch model dat finite mixtures genoemd wordt. Een mixture is een verzameling van k kansverdelingen die de k clusters representeren. Elke verdeling geeft aan wat de kans $P(x | C_i)$ is dat een bepaald object door een bepaalde verzameling features wordt gekarakteriseerd, gegeven het feit dat dit object behoort tot het specifieke cluster die de verdeling representeert [39]. Elk cluster heeft een verschillende kansverdeling en elk object behoort maar tot één van deze clusters, maar men weet niet op voorhand welk cluster dit is. Omdat een eindige hoeveelheid aan informatie niet voldoende is om een object met 100% zekerheid aan 1 cluster toe te kennen, krijgt elk object een kansverdeling toegewezen die aangeeft wat de kans is dat het object tot een bepaald cluster C_i behoort.

In de praktijk gebruikt men vaak een multivariate normale verdeling om elk cluster te representeren. Deze verdeling wordt gekenmerkt door enkele parameters die benodigd zijn bij het berekenen van de kans $P(x | C_i)$. Deze parameters zijn op voorhand echter niet bekend en dienen dus aan de hand van de data geschat te worden. Expectation Maximization (EM) is een algemene methode waarmee deze parameters geschat kunnen worden [31]. In sommige gevallen geldt dat het EM-algoritme gelijk is aan het k-means algoritme. Het volgende algoritme produceert een clustering van de data waarbij er gebruik gemaakt wordt van EM om de verschillende parameters te schatten:

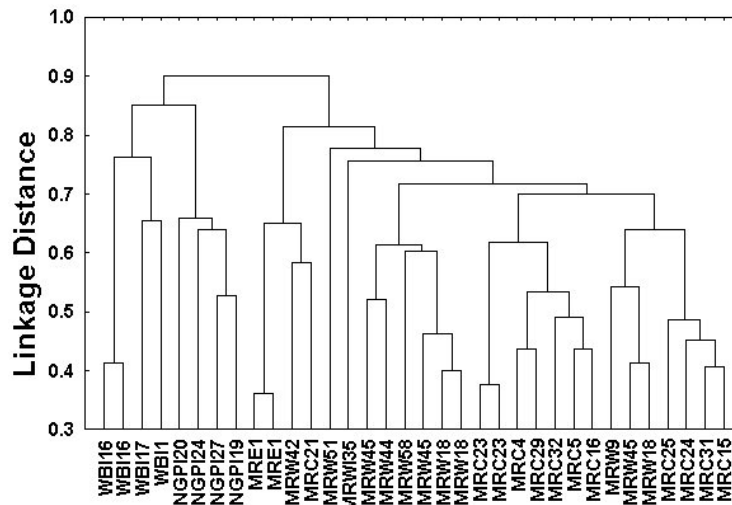
- (1) Initialisatie: de initiële parameters van de k verdelingen worden random geselecteerd of worden door de gebruiker op een slimme manier geschat.
- (2) Expectation-stap: bereken de kansen $P(C_i | x) = (P(x | C_i)P(C_i)) / P(x)$ voor alle objecten x door gebruik te maken van de huidige parameters van de verdelingen²⁰. Deze kansen stellen de “verwachte” kansen voor dat een object x tot categorie C_i hoort. Aan de hand van deze kansen worden alle objecten voorzien van een nieuw label. Een object wordt aan de categorie toegekend waarvoor de kans $P(C_i | x)$ het grootst is.
- (3) Maximization-stap: schat opnieuw de parameters van de verdelingen zodanig dat de likelihood van de verdelingen, gegeven de data, gemaximaliseerd wordt.
- (4) Als het convergentie criterium niet bereikt is ga dan weer terug naar stap 2. Er is sprake van convergentie indien de veranderingen in de log-likelihood na elke iteratie erg klein worden en het label van de objecten niet meer verandert.

Het EM-algoritme convergeert gegarandeerd naar een maximum, maar dit hoeft echter niet het globale maximum te zijn. Om de kans op het vinden van het globale maximum te verhogen dient het bovenstaande proces meerdere keren herhaald te worden met steeds andere initiële parameters voor de k verdelingen. De “overall likelihood”, die verkregen wordt door de kansen $P(C_i | x)$ voor alle objecten met elkaar te vermenigvuldigen, kan hierbij gebruikt worden om de verschillende runs van het algoritme met elkaar te vergelijken. De beste run is diegene waarvan de overall likelihood het hoogst is [39].

5.3.3 HIERARCHISCHE AGGLOMERATIEVE CLUSTERING (HAC)

Het HAC algoritme produceert een geneste reeks van partities in de vorm van een dendogram: een boom die de hiërarchische structuur representeert en de hoogte aangeeft waarop de clusters van samenstelling veranderen (zie figuur 5.2). Deze boom kan op elke gewenste hoogte worden afgebroken om zo de data in verschillende varianten te clusteren.

²⁰De kans $P(x)$ hoeft uiteindelijk niet berekend te worden omdat deze tijdens de normalisatie wegvalt.



Figuur 5.2: Een dendrogram die een hiërarchische clustering weergeeft.

Tijdens de start van het HAC algoritme bevinden alle objecten zich in aparte clusters waarna de clusters die het meest op elkaar lijken, volgens een of ander criteria, worden samengevoegd. Het algoritme eindigt wanneer alle clusters zijn samengevoegd tot één groot cluster. Het algoritme werkt als volgt:

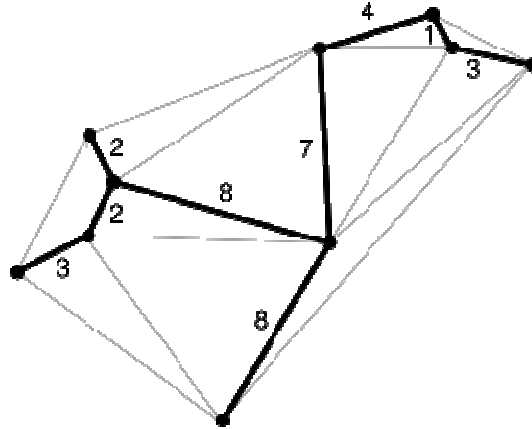
- (1) Initialisatie: elk object wordt in een apart cluster ingedeeld.
- (2) Iteratie: vind het paar clusters dat het meest op elkaar lijkt en voeg deze samen.
- (3) Stopcriterium: wanneer alle objecten in een cluster zijn samengevoegd.

Er zijn verschillende manieren waarop de overeenkomsten tussen de clusters bepaald kunnen worden [18, 31]. Volgens de single-link methode is de overeenkomst tussen twee clusters gedefinieerd als het maximum van de overeenkomsten tussen alle mogelijke paren (waarbij één object afkomstig is uit de ene cluster en het tweede object uit de andere cluster) die geconstrueerd kunnen worden met behulp van de twee clusters. In het geval van de complete-link methode is de overeenkomst juist gedefinieerd als het minimum van de overeenkomsten tussen deze paren van objecten. De single-link methode resulteert in clusters die lang en dun zijn, terwijl de complete-link methode resulteert in compacte clusters. Ervaring heeft uitgewezen dat de complete-link methode in de praktijk bruikbaarere resultaten oplevert [18, 31].

5.3.4 OVERIGE CLUSTERINGALGORITMEN

Er bestaan verschillende algoritmen die gebaseerd zijn op grafen (zie figuur 5.3). De bekendste is gebaseerd op de minimaal opspannende boom van een graaf. De minimaal opspannende boom bestaat uit de kanten van een graaf zodanig dat alle punten in de graaf met elkaar verbonden zijn en de som van de lengtes van de kanten zo klein mogelijk is. De minimaal opspannende boom is in de onderstaande graaf dikgedrukt. De clusters worden gegenereerd door de langste kanten in de minimaal opspannende boom te verwijderen. De hiërarchische methoden zijn gerelateerd aan de deze methode. Single-link clusters zijn namelijk subgrafenvan de minimaal opspannende boom.

Nearest Neighbor clustering kent een object toe aan het cluster waar de dichtstbijzijnde gelabelde buur toe behoort onder de voorwaarde dat de overeenkomst met deze buur voldoende groot is. Dit proces gaat door totdat alle objecten voorzien zijn van een label. In het **Buckshot** algoritme wordt het HAC algoritme gebruikt om een goede eerste partitionering van de data te verkrijgen. Deze partitionering wordt vervolgens in het k-means algoritme gebruikt als startsituatie [31].



Figuur 5.3: De minimaal spannende boom (vetgedrukt) van een graaf.

5.4 EVALUATIE VAN DE CLUSTERS

Het evalueren van de resultaten van een clustering is een erg belangrijke stap in het clusteringproces. Vaak zijn wij geïnteresseerd in het feit of een clustering geschikt is voor gebruik in een bepaalde toepassing. Over het algemeen zijn er twee manieren waarop de resultaten van de clustering geëvalueerd kunnen worden. Men kan er aan de ene kant voor kiezen om gebruik te maken van indices waarmee de eigenschappen van de clusters beschreven kunnen worden, of men kan aan de andere kant er voor kiezen om alle objecten op voorhand te voorzien van het juiste label dat vergeleken kan worden met het label dat is toegekend aan de objecten middels de clustering [15]. Wij zullen beide manieren in het vervolg van deze paragraaf bespreken.

5.4.1 INDICES

De indices maken geen gebruik van een op voorhand opgestelde labeling en evalueren de kwaliteit van een cluster op basis van statistische eigenschappen. In de literatuur kan men een groot aantal verschillende indices vinden (zie [15, 18, 35]). Een van de bekendste indices is de gemiddelde kwadratische fout (Engels: mean squared error). Een andere veel gebruikte index is de Davies-Bouldin index. Het grote nadeel van beide indices is dat de berekende kwaliteit altijd toeneemt naarmate het aantal clusters groter wordt. In [22] wordt een alternatieve maat besproken, de silhouette coëfficiënt, die deze nadelige eigenschap niet heeft. Wij zullen deze indices in het vervolg introduceren.

Gemiddelde kwadratische fout

Indien men het aantal dimensies en het aantal clusters constant houdt kan de gemiddelde kwadratische fout gebruikt worden voor de evaluatie van de clustering [15]. De gemiddelde kwadratische fout is een maat voor de compactheid van de gevormde clusters en is voor een gegeven clustering P als volgt gedefinieerd:

$$MSE(P) = \sum_{p \in P} MSE(p), \quad (5.5)$$

waarbij de gemiddelde kwadratische fout voor een cluster p gegeven wordt door:

$$MSE(p) = \sum_{d \in p} \text{afstand}(d, \mu_p)^2, \quad (5.6)$$

en $\mu_p = \frac{1}{p} \sum_{d \in p} \vec{t}_d$ het centrum van de clusters p is en *afstand* een afstandsmaat voorstelt die de afstand tussen een object d en het centrum van zijn cluster aangeeft.

Davies-Bouldin index

De Davies-Bouldin index [7] is een functie van de ratio tussen de som van de spreiding in een cluster en de afstand tussen de clusters onderling. Deze index is als volgt gedefinieerd:

$$DB = \frac{1}{n} \sum_{i=1}^n \max_{i \neq j} \left(\frac{S_n(\mu_i) + S_n(\mu_j)}{S(\mu_i, \mu_j)} \right), \quad (5.7)$$

waarbij n het aantal clusters voorstelt, S_n de gemiddelde afstand tussen alle objecten in een cluster en het clustercentrum weergeeft en $S(\mu_i, \mu_j)$ de afstand tussen de verschillende clustercentra representeert. De ratio is klein indien de clusters compact en ver van elkaar verwijderd zijn. Het gevolg hiervan is dat deze index een kleine waarde zal hebben indien wij te maken hebben met een goede clustering.

Silhouette coëfficiënt

De silhouette coëfficiënt is een index die onafhankelijk is van het aantal clusters. Het belangrijkste idee achter deze coëfficiënt is het bepalen van de locatie van een object in een ruimte met respect tot de cluster waartoe het document behoort en de dichtstbijzijnde cluster. In een goede clustering ligt het object dicht bij zijn eigen cluster terwijl in een slechte cluster het object dicht bij de dichtstbijzijnde cluster ligt. De silhouette coëfficiënt kan op de volgende manier worden berekend:

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}, \quad (5.8)$$

waarbij $a(i)$ aangeeft in welke mate object i gemiddeld verschilt van de overige objecten in dezelfde cluster en $b(i)$ de gemiddelde mate weergeeft waarin object i verschilt van de objecten in de dichtstbijzijnde cluster.

Uit de formule in (5.8) kan afgeleid worden dat $-1 \leq S(i) \leq 1$. Als $S(i)$ dichtbij 1 ligt betekent dit dat het object goed geclusterd is en dat het aan de juiste cluster is toegekend. Indien $S(i)$ gelijk is aan 0 betekent dit dat het object ook aan een andere cluster had kunnen worden toegekend. In dit geval ligt het object namelijk tussen twee clusters in en is de afstand van het object tot beide clusters even groot. Een $S(i)$ die in de buurt ligt van -1 houdt in dat het object simpelweg niet juist geclusterd is.

Men kan de gehele clustering evalueren door het gemiddelde te nemen van de silhouette coëfficiënten van alle objecten in de dataset. De grootste gemiddelde waarde geeft de beste clustering aan, en daarmee dus ook het optimale aantal clusters.

5.4.2 VERGELIJKENDE MATEN

De purity maat is gebaseerd op de bekende precision maat uit het vakgebied van information retrieval [15]. Elke resulterende cluster p als element van een grotere partitie P van de gehele dataset D wordt gezien als de verzameling documenten die geretourneerd wordt als het resultaat op een query. Elke verzameling l , element van een grotere partitie L die wordt verkregen door de objecten handmatig te labelen stelt de gewenste verzameling van objecten voor die geretourneerd dient te worden. Door de partities P en L op deze manier te bekijken zijn wij in staat om deze partities met elkaar te vergelijken middels de precision, recall en F-score.

De **precision** van een cluster $p \in P$ voor een gegeven categorie $l \in L$ wordt gegeven door:

$$Precision(p, l) := \frac{|p \cap l|}{|p|}. \quad (5.9)$$

De algemene waarde voor de purity wordt berekend door het gewogen gemiddelde te nemen van de maximale precision waarden:

$$Purity(p, l) := \sum_{p \in P} \frac{|P|}{|D|} \max_{l \in L} Precision(p, l). \quad (5.10)$$

Het tegengestelde van purity is inverse purity:

$$InversePurity(p, l) := \sum_{p \in P} \frac{|P|}{|D|} \max_{l \in L} Recall(p, l), \quad (5.11)$$

waarbij $Recall(p, l) := Precision(l, p)$.

De F-measure die gebaseerd is op de F-score is gedefinieerd als:

$$F - Measure(p, l) := \sum_{l \in L} \frac{|L|}{|D|} \max_{p \in P} \frac{2 \cdot Recall(p, l) \cdot Precision(p, l)}{Recall(p, l) + Precision(p, l)}. \quad (5.12)$$

De drie bovenstaande maten retourneren waarden die in het interval $[0, 1]$ liggen waarbij de waarde 1 optimale overeenkomst tussen p en l aangeeft. Purity meet de homogeniteit van de resulterende clusters wanneer deze vergeleken worden met de voorgedefinieerde categorisatie, terwijl inverse purity meet in welke mate de voorgedefinieerde categorieën stabiel zijn wanneer zij worden opgedeeld in verschillende clusters. Hieruit volgt dat purity een “optimale” waarde van 1 behaalt indien het aantal clusters k gelijk is aan $|D|$. De inverse purity bereikt daarentegen zijn “optimale” waarde indien k gelijk is aan 1. De F-measure heeft dezelfde werking als de inverse purity, maar devalueert de hele grote clusters doordat het de individuele precisie van deze clusters meeneemt in de evaluatie.

5.5 HET CLUSTEREN VAN TEKSTDOCUMENTEN

De clusteringalgoritmen en de evaluatietechnieken die in paragraaf 5.2 en 5.3 zijn beschreven zijn direct toepasbaar op een collectie tekstdocumenten nadat deze zijn omgezet in vectoren in een bepaalde vectorruimte, zoals is besproken in hoofdstuk 3. Het grootste probleem is echter dat een grote collectie tekstdocumenten er voor kan zorgen dat de dimensies van deze ruimte erg groot zijn wat een negatief effect kan hebben op de clusteringalgoritmen. Om dit probleem te verhelpen kan men gebruik maken van technieken die deze dimensies kunnen reduceren. Hiervoor kunnen de selectiemethoden die in paragraaf 3.4 zijn besproken gebruikt worden. Een andere dimensie-reductie techniek die in tekstmining steeds meer aan populariteit wint is Latent Semantic Indexing (LSI). Deze techniek is uitgebreid beschreven in paragraaf 3.5.

Nadat de dimensies zijn gereduceerd kunnen de clusters opgesteld worden. De laatste en misschien wel belangrijkste stap is het genereren van een zinvolle en beknopte beschrijving van de clusters die gebruikt kan worden voor verdere automatische verwerking of geconsumeerd kan worden door de gebruiker. Het creëren van een beschrijving die door machines verwerkt kan worden is het makkelijkst. Hiervoor kunnen simpelweg de clustercentra of de opgestelde probabilistische modellen gebruikt worden.

In het geval dat er tekstdocumenten geclusterd worden willen wij ieder cluster voorzien van een zinvol label. Voor sommige toepassingen is een zinvol label namelijk even belangrijk als een goede clustering. Een goed label bestaat uit een klein aantal termen die op een precieze wijze de betreffende cluster onderscheidt van de overige clusters [31]. Indien wij bijvoorbeeld documenten clusteren met als onderwerp “jaguar” willen wij het liefst twee clusters zien: een cluster met het label “dier” en een andere cluster met het label “auto”.

Er zijn verschillende manieren waarop deze labels automatisch gegenereerd kunnen worden:

- (1) De titels van verschillende documenten die behoren tot een bepaald cluster kunnen gebruikt worden voor het opstellen van de labels.
- (2) De woorden die het meest voorkomen in de tekstdocumenten die zich in een cluster bevinden kunnen als label fungeren. Een heuristiek die vaak wordt toegepast is het tonen van de 5 of 10 woorden die het vaakste voorkomen in de vector die het centrum van een cluster voor stelt.
- (3) Een kenmerkend telbaar zelfstandig naamwoord, indien deze gevonden kan worden is mogelijk het beste label voor een cluster.

HOOFDSTUK 6

ONDERZOEKSOPZET

VERTROUWELIJK

HOOFDSTUK 7

CLASSIFICATIE-EXPERIMENTEN COLLECTIE ADMINISTRATIE

VERTROUWELIJK

HOOFDSTUK 8

CLASSIFICATIE-EXPERIMENTEN COLLECTIE TELEFOON

VERTROUWELIJK

HOOFDSTUK 9

EXPERIMENTEREN MET CLUSTEREN

VERTROUWELIJK

HOOFDSTUK 10

CONCLUSIES

VERTROUWELIJK

HOOFDSTUK 11
AANBEVELINGEN

VERTROUWELIJK

LITERATUURLIJST

- [1] Baeza-Yates, R. en B. Ribeiro-Neto (1999), *Modern Information Retrieval*. Addison Wesley Longman.
- [2] Baker, K. (2005), Singular Value Decomposition Tutorial.
(abraxas.sprachwiss.uni-konstanz.de/~mayer/courses/algomorph/inc/Singular_Value_Decomposition_Tutorial.pdf)
- [3] Balog, K. (2004), *An Intelligent Support System for Developing Text Classifiers*. Afstudeerscriptie, Vrije Universiteit, Amsterdam.
- [4] Bergman, M.K, The deep web: Surfacing hidden value.
(<http://www.press.umich.edu/jep/07-01/bergman.html>)
- [5] Bernstein, A., S. Clearwater, S. Hill, C. Perlich en F. Provost (2002). Discovering knowledge from relational data extracted from business news. *MRDM02*, blz. 7-20.
- [6] Brandhof, W. van den (2007), *Gebruik je hersens (1^e druk)*. Ruitenberg Boek.
- [7] Davies, D.L. en D.W. Bouldin (1979), Cluster Separation Measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1(2), blz. 95-104.
- [8] Dörre, J., P. Gerstl en R. Seiffert (1999), Text mining: Finding Nuggets in Mountains of Textual data. *Knowledge Discovery and Data Mining*, blz. 398-401.
- [9] Dumais, S., J. Platt, D. Heckerman en M. Sahami (1998), Inductive learning algorithms and representations for text categorization. In *7th Int. Conf. on Information and Knowledge Management*.
- [10] Fayyad, U.M., G. Piatetsky-Shapiro en P. Smyth (1996), Knowledge discovery and datamining: Towards a unifying framework. *Knowledge discovery and Data mining*, blz. 82-88.
- [11] Forman, G. (2003), An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research* 3(2003), blz. 1289-1305.
- [12] Frakes, W.B. en R. Baeza-Yates (1992), *Information Retrieval: Data Structures & Algorithms*. Prentice Hall, New Jersey.
- [13] Hearst, M.A. (1999), Untangling text data mining. In *Proceedings of ACL '99: the 37th Annual Meeting of the association for Computational Linguistics*.
- [14] Hearst, M.A., What is text mining? (2003).
(<http://people.ischool.berkeley.edu/~hearst/text-mining.html>)
- [15] Hotho, A., A. Nürnberger en G. Paaß (2005), A Brief Survey of Text Mining.
(www.kde.cs.uni-kassel.de/hotho/pub/2005/hotho05TextMining.pdf)
- [16] Ikonomakis, M., S. Kotsiantis en V. Tampakas (2005), Text Classification Using Machine Learning Techniques. *WSEAS transactions on computers*, 8(4), blz. 966-974.
- [17] Jackson, P. en I. Moulinier (2002), *Natural Language Processing for Online Applications*. Johns Benjamins.

- [18] Jain, J.K., M.N. Murty en P.J. Flynn (1999), Data Clustering: A Review. *ACM Computing Surveys* 31(3), blz. 264-323.
- [19] Janssens, F. (2007), *Clustering of scientific fields by integrating text mining and bibliometrics*. Proefschrift voorgedragen tot het behalen van het doctoraat in de ingenieurswetenschappen. Katholieke universiteit Leuven, Leuven.
- [20] Joachims, T. (1998), Text categorization with support vector machines: Learning with many relevant features. *European Conference on Machine Learning*.
- [21] John, G.H. en P. Langley (1995), Estimating Continuous Distributions in Bayesian Classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, blz. 338-345.
- [22] Kaufman L. en P.J. Rousseeuw (1990), *Finding groups in data: an introduction to cluster analysis*. Wiley, New York.
- [23] Larkey, L.S. en W.B. Croft (1996). Combining Classifiers in Text Categorization. *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*.
- [24] Leopold, E. en J. Kindermann (2002), Text categorization with support vector machines. How to represent texts in input space? *Machine Learning* (46), blz. 423-444.
- [25] Mao, J. en A.K. Jain (1996), A self-organizing network for hyperellipsoidal clustering (HEC). *IEEE Trans. Neural Networks* (7), blz. 16-29.
- [26] Mitchell, T.M. (1997), *Machine Learning*. The McGraw-Hill Companies.
- [27] Nahm, U.Y. en R.J. Mooney (2003), Text mining with information extraction. *Multilingualism and Electronic Language Management: Proceedings of the 4th International MIDP Colloquium*. (<http://www.cs.utexas.edu/users/ml/papers/discotex-melm-03.pdf>)
- [28] M. Porter (1980), An algorithm for suffix stripping. *Program*, blz. 130-137.
- [29] Rijsbergen, C.J. van (1979), *Information Retrieval (2e druk)*. Afdeling Computer Science, Universiteit van Glasgow.
- [30] Russel, S. en P. Norfig (2003), *Artificial Intelligence: A Modern Approach*. Pearson Education.
- [31] Sanger, J. en R. Feldman (2006), *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured data*. Cambridge University Press.
- [32] Schapire, R. E. en Y. Singer (1998), Improved boosting algorithms using confidence-rated predictions. *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, blz. 80-91.
- [33] Schapire, R. E. en Y. Singer (2000), BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning* (39), blz. 135-168.
- [34] Sehgal, A.Y. (2004), *Text mining: the search for novelty in text*. Ph. D. Comprehensive Examination report, Universiteit van Iowa, Verenigde Staten. (<http://www.cs.uiowa.edu/~sehgal/>)
- [35] Sebastiani, F. (2002), Machine Learning in Automated Text Categorization. *ACM Computing Surveys* 1(34), blz. 1-47.

- [36] Swanson, D.R., N.R. Smalheiser en A. Bookstein (2001), *Information discovery from complementary literatures: Categorizing viruses as potential weapons*. Journal of the American Society for Information Science And Technology 52(10), blz. 797-812.
- [37] Tan, A., Text mining: The state of art and the challenges (1999). *Proceedings of the Pacific Asia Conference on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge discovery from Advanced Databases*, blz. 65-70.
- [38] Weeber, M., W.J.A. Schrijvenaars, E.M. van Mulligen, B. Mons, R. Jelier, C. van der Eijk en J.A. Kors (2003), *Ambiguity of human gene symbols in locuslink en medline: Creating an inventory and a disambiguation test collection*. Proceedings of AMIA Symposium, blz. 704-708.
- [39] Witten, I.H. en F. Eibe (2005), *Data mining: practical machine learning tools and techniques (2e druk)*. Morgan Kaufmann Publishers.
- [40] Yang, Y. (2001), A Study on Thresholding Strategies for Text Categorization. *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*.
- [41] Zhao, Y. en G. Karypis (2005), Hierarchical Clustering Algorithms for Document Datasets. *Data Mining and Knowledge Discovery*, 10, blz. 141-168.

BIJLAGE A
DATASETS

VERTROUWELIJK

BIJLAGE B

XML PROCES CONFIGURATIE FILES IN RAPIDMINER

VERTROUWELIJK

