Vrije Universiteit Brussel

Faculteit Wetenschappen
Departement Computerwetenschappen

Gevoelsanalyse in het Nederlands

Yannick Merckx

Bachelorproef

Rolnummer: 500294

Promotor: Yann-Michaël De Hauwere
Begeleiders: Maarten Deville
Peter Vrancx

Juni 2015

Samenvatting

Gevoelsanalyse is een populair gegeven binnen de Machine Learning. In deze bachelorproef gaan we op zoek of het mogelijk is om aan de hand van enkele eenvoudige machine learning technieken een gevoelsanalyse uit te voeren. Specifieker focussen we ons op de Nederlandse taal, waarvan het naslagwerk vandaag de dag eerder beperkt is. Als onderwerp van de gevoelsanalyse worden film-, boek- en muziekrecensies aan de hand van een algoritme beoordeeld of ze een positieve of negatieve emotie uitdrukken. Voor het onderzoek bekijken we de theoretische kant van een gevoelsanalyse, waar we de mogelijke technieken bespreken. Daarnaast wordt ook de praktische zijde uitgewerkt waar we de theoretische kennis gaan omzetten in een experiment. Dit experiment toont aan dat het mogelijk is om gevoelsanalyse uit te voeren op het Nederlands.

Dank woord

Het maken van een bachelorproef doe je nooit alleen, daarom ook een woord van dank aan enkele mensen waarop ik gedurende mijn eindwerkproces steeds kon terugvallen. Als eerste zou ik graag mijn begeleiders, Maarten Deville en Peter Vranckx, van harte willen bedanken voor hun grote steun en inzet gedurende het hele jaar. Ze waren gedurende het hele jaar altijd beschikbaar om op al mijn vragen een snel antwoord te geven. Als laatste wens ik ook mijn dank uit te drukken aan mijn promotor, Yann-Michaël De Hauwere. Bij de korte evaluaties was hij altijd aanwezig en stond hij mij altijd bij met raad en daad.

Inhoud

Hoofdstuk 1

Introductie

Vandaag de dag beschikken we over een enorme hoeveelheid aan digitale informatie. En iedere dag wordt deze hoeveelheid groter en groter. In deze *"Age of Big Data"* (Lohr (2012)) bestaat de uitdaging erin om uit deze grote hoeveelheid data, door middel van analyse bepaalde inzichten te krijgen.
Vele hebben dit probleem proberen aan te pakken, waarbij men zich bezig hield met structuur brengen in deze grote dataset en zich voornamelijk concentreerde op onderwerp-gebaseerde classificatie. Echter met de opkomst van sociale media, blogs, reviewsites is een groeiende interesse ontstaan voor gevoelsanalyse. Het onderzoek dat hiernaar gebeurd, heeft ook de aandacht van bedrijven en spelen hier ook een rol in. Bijna al het onderzoek dat gebeurd is de afgelopen jaren zijn uitgevoerd op Engelse teksten en er is zeer weinig te vinden over onderzoek naar gevoelsanalyse op het Nederlands. Deels omdat het Engels een wereldtaal is en het Nederlands niet, maar ook door de bedrijven die dergelijk onderzoek binnenshuis houden.
Voor deze bachelorproef kijken we of de bevindingen over gevoelsanalyse op het Engels gelijk toepasbaar zijn op het Nederlands en of er verschillen zijn tussen het Nederlands en het Engels, waarbij men met een gevoelsanalyse rekening mee moet houden.
We hebben op basis van de voorbereiding op de bachelorproef in het 1ste semester en een verdere verdieping in de literatuur over gevoelsanalyse een selectie gemaakt van technieken die we gaan vergelijken tijdens het experiment.In hoofdstuk 2 worden deze technieken met hun theoretische achtergrond en de selectie van deze technieken besproken. Vervolgens gaan we in hoofdstuk 3 over naar drie experimenten. We vergelijken in het eerste experiment de resultaten van de technieken op Nederlandse en Engelse filmrecensies. Vervolgens concentreren we ons op de woordenschat en onderzoeken we de classificatie op basis van een Engels en Nederlands geannoteerde woordenlijst van gevoelens. Als laatste vergelijken we mogelijke invloeden van het onderwerp van een tekst en vergelijken we de prestatie van Nederlandse gevoelsanalyse op verschillende recensies met verschillende onderwerpen. Na het experiment vormen we een conclusie of de gevonden technieken voor gevoelsanalyse op het Engels al dan niet gelijk toepasbaar zijn op het Nederlands.

Hoofdstuk 2

Lectuur

In dit hoofdstuk wordt de theoretische achtergrond en de keuze voor de technieken die we gebruiken tijdens het experiment toegelicht. We bespreken in 2.1 de gebruikte voorstelling van de dataset. In 2.2 bespreken we enkele optimalisatie technieken die kunnen helpen om de classificatieprestaties te verbeteren. Daarna volgen in 2.3 de zelflerende algoritmes zelf, waarin de werking van en de keuze voor de algoritmes wordt uitgelegd.

2.1 Voorstelling dataset

De voorstelling van de data is een eerste element van het experiment waarmee men rekening moet houden. We kunnen bijvoorbeeld rauwe data meegeven aan het zelflerende algoritme of we kunnen de tekst omvormen naar een vector die het aantal voorkomens van ieder woord in de tekst bevat. Voor het experiment kiezen we het tweede voorbeeld, waarbij we een document voorstellen als een vector met daarin de woordfrequentie. Dit wordt de vector space methode genoemd en wordt door Turney et al. (2010) beschouwd als onderdeel van de oplossing voor de problematiek rond semantische analyse. Verder is deze voorstelling een populaire methode binnen het onderzoek naar gevoelsanalyse op het Engels en heeft dit zijn werking al aangetoond. Zie bijvoorbeeld Pang et al. (2002) en Maas et al. (2011a).

2.1.1 Vector Space Methode

De vector space methode (VSM) is een methode waarbij we een document als een vector voorstellen waarbij ieder element overeenkomt met een woord en zijn frequentie in het document. De elementen van de vector worden ook wel features genoemd. Als men concreet een document voorstelt kan men zeggen dat document j voorgesteld wordt door d_j met f_{ij} de frequentie van het woord w_i. Met de frequentie f_{ij} bedoelt men het totaal aantal voorkomens van het woord w_i in document j. Het aantal verschillende woorden in het document stelt men voor door n_w, wat eveneens de dimensie is van de vector. Het document j kan dus als volgt worden voorgesteld:

$$d_j = \begin{bmatrix} f_{1j} \\ f_{2j} \\ \vdots \\ f_{n_w j} \end{bmatrix}$$

Een belangrijk inzicht bij de vector space methode is dat een document voorgesteld wordt als een groep van woorden. Er wordt geen rekening gehouden met de volgorde waarin de woorden

in het document voorkomen. Vaak ziet men ook dat de vector vaak ijl is en vanwege de grote hoeveelheid aan woorden in een document heel groot. Als we nu niet één document, maar meerdere documenten nemen en we zeggen dat het aantal documenten gelijk is aan n_d, resulteert dit in een matrix waarbij iedere kolom een document voorstelt.

$$D = \begin{array}{c} \text{Documenten} \\ \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1n_d} \\ f_{21} & f_{22} & \cdots & f_{2n_d} \\ \vdots & \vdots & \ddots & \vdots \\ f_{n_w j} & f_{n_w 2} & \cdots & f_{n_w n_d} \end{bmatrix} \end{array} Woorden$$

Deze matrix wordt een ***terms-documents matrix (TDM)*** genoemd. Wanneer men spreekt van een ***documents-terms matrix (DTM)***, spreekt men een getransponeerde terms-documents matrix. Een rij van een DTM stelt dan een document voor. In het experiment stellen we onze data voor aan de hand van een documents-terms matrix De voorstelling in een matrix geeft inzicht en biedt veel meer mogelijkheden om de data te analyseren. Bijvoorbeeld overeenkomstige woordfrequenties tussen twee documenten kan duiden dat documenten over hetzelfde onderwerp gaan of eenzelfde mening uitdrukken. In de praktijk is gebleken dat documenten vergelijken op basis van woordfrequentie niet altijd de gewenste resultaten oplevert en Pang et al. (2002) toont aan dat er ruimte is voor verbetering door middel van Pre-Processing technieken.

2.2 Technieken voor Pre-Processing

Zoals we in 2.1.1 al vermeldde kan pre-processing voor verbetering van de classifiers zorgen. De pre-pocestechnieken die we gebruiken in deze bachelorproef zijn al eerder gebruikt door Pang et al. (2002) en Wang & Wan (2011) en hadden een positief effect.

2.2.1 Bag of Words

De eerste techniek is Bag of Words. Dit is niet echt een pre-procestechniek, maar eerder een referentiepunt voor de andere pre-procestechnieken. Het steunt op het principe waarop de vector space methode zich baseert, waarbij ieder document wordt voorgesteld door zijn woordfrequenties. Het is de basistechniek die wordt uitgevoerd bij een gevoelsanalyse aan de hand van de VSM.

2.2.2 Verwijderen van stopwoorden

Wat men vaak ziet in het Nederlands, maar ook in taal algemeen, is dat er veel stopwoorden worden gebruikt. Stopwoorden als “klopt” en “eigenlijk” zeggen niet veel over teksten of ze nu positief of negatief zijn. Als een bepaald woord niet bijdraagt voor het algoritme kunnen we stopwoorden beschouwen als ruis in de dataset. Ruis vertroebelt het beeld van het concept dat we het algoritme willen aanleren en proberen we te elimineren. Daarom beschouwt men het verwijderen van stopwoorden en leestekens ook als een manier van pre-processing.

2.2.3 Term weighting

Als we terugkijken naar de vector space methode, waarbij we enkel rekening houden met de woordfrequentie, kan men zeggen dat niet elk woord evenveel doorweegt. Een woord dat in alle documenten voorkomt biedt geen of minder waardevolle informatie, dan een woord dat zelden voorkomt. En hierop baseert term weighting zich. Het gaat een wegingsfactor introduceren.

Ieder woord krijgt een gewicht toegewezen, dat weergeeft hoe belangrijk het woord is. Neem als voorbeeld een hoop recensies van de film "Pulp Fiction" en de woorden "Pulp" en "excellent". "Pulp" is een woord dat voorkomt in de titel van de film en komt ongetwijfeld in elke recensie voor. "Excellent" daarentegen is een woord dat enkel maar voorkomt wanneer de recensent de film fantastisch vond, het zal niet in elk document voorkomen en is waardevolle informatie. Term weighting zal dus bij dit voorbeeld "excellent" een groter gewicht toewijzen dan "Pulp". De kwantiteit van dit gewicht wordt vaak de **inverse document frequency (idf)** genoemd en wordt bepaald aan de hand van volgende formule:

$$w_i : idf_i = -log_2[P(w_i)]$$

met $P(w_i)$ de priori probability dat woord w_i voorkomt in het document.
De inverse document frequency geeft het algemeen belang van het woord w_i weer. Men kan dit benaderen door het logaritme te nemen van het aantal documenten waar w_i in voorkomt en het totaal aantal documenten. Een andere nuttige kwantiteit is de **term frequency** tf_{ij}. Deze geeft het belang weer van het woord w_i binnen in het document d_j en wordt als volgt genoteerd:

$$tf_{ij} = \frac{f_{ij}}{\sum_{i=1}^{n_w} f_{ij}}$$

tf_{ij} wordt berekend door de frequentie, het aantal voorkomens, van een woord w_i in document d_j te delen door de som van alle woordfrequenties in document d_j. Met deze twee kwantiteiten kan men een nieuwe begrip introduceren: de **tf-idf score**. Wat overeenkomt met het product van tf en idf.

$$\text{tf-idf score} = tf.idf_{ij} = idf_i.tf_{ij}$$

De tf-idf matrix bekomt men dan door alle woordfrequenties van het terms-document matrix te vervangen door de tf-idf score. Er bestaan nog uitbreiding op term weighting (**?**), maar voor het experiment houden we het bij de standaard tf-idf weighting.

2.2.4 Bigram Collocaties

Bigrams Collocaties is een techniek waarbij men op zoek gaat naar paren van woorden die een hoge waarschijnlijkheid hebben om samen voor te komen en een extra bron van informatie kunnen vormen. In het onderzoek van Pang et al. (2002) bleken bigrams niet voor een verbeterde prestatie te zorgen, al mag men de nuttigheid van bigrams niet onderschatten. Toch nemen we bigrams als een van de technieken,**?** toonde echter aan dat bigrams een nuttig kenmerk vormen voor het oplossen van woord zin ambiguÃŕteit. Pang et al. (2002) merkt dan ook zelf op in zijn onderzoek dat bigram features mogelijks evenwaardig zijn met unigram features.
De bepaling van de informatieve waarde van de bigrams is gebaseerd op de frequentie van het bigram en de frequenties van de andere bigrams. Als men een overzicht krijgt over de frequenties introduceert men een metriek, die met behulp van de frequenties mogelijke verbanden kan blootleggen. Chi-kwadraat is zo'n metriek die er zich toe leent. De Chi-kwadraattoets is een statistische toets die het mogelijk maakt om de onafhankelijkheid tussen waarnemingen te onderzoeken. Bij Bigram Collocaties onderzoekt men via de Chi-kwadraattoets de afhankelijkheid tussen twee woorden. Hoe grotere de afhankelijkheid, hoe hoger de score.

Chi-Kwadraattoets

De Chi-Kwadraattoets is een techniek uit de statistiek die gebruikt kan worden als een onafhankelijkheidstoets voor waarnemingen. De reden waarom we deze toets voor Bigram collocatie

gebruiken is dat de toets parametervrij is. Wat wil zeggen dat er bij de start van de chi-kwadraattoets geen aanname over de populatie of het gemiddelde wordt verwacht. In deze sectie leggen we aan de hand van een voorbeeld uit hoe de chi-kwadraattoets juist deze afhankelijkheid bepaald.
Neem als voorbeeld het bigram *(heel , goed)*. Zoals bij iedere statistische test neemt men eerst een nulhypothese aan. Voor de chi-kwadraattoets is dit ook het geval. De toets neemt als nulhypothese aan dat beide woorden onafhankelijk van elkaar zijn en elkaars voorkomen niet beïnvloeden. Men vergelijkt de waargenomen frequenties van de woorden met de verwachte frequenties wanneer de woorden onafhankelijk zouden zijn. Als deze waarden te veel verschillen kan men de nulhypothese verwerpen en de alternatieve hypothese aannemen, namelijk dat de woorden afhankelijk zijn van elkaar.

Om de afhankelijkheid van woorden te bepalen, kijken we naar volgende gegevens:

- het aantal voorkomens van het woord in een bigram
- het aantal voorkomens van het woord in een bigram met het ander woord waar we de afhankelijkheid van onderzoeken
- het totaal aantal bigrams
- het aantal voorkomens van het ander woord in een bigram.

Als we voor het voorbeeld *(heel , goed)* bovenstaande gegevens in een kruistabel gieten krijgen we de volgende 2x2 tabel:

	w1= heel	w1 $\neq$ heel
w2 = goed	9 (heel goed)	7893 (bv. niet goed)
w2 $\neq$ goed	3632 (bv. heel slecht)	13498000 (bv. boeiende thesis)

We weten nu naar wat we moeten kijken bij het analyseren van de afhankelijkheid maar er mist nog een weging, een onderlinge verhouding tussen de kenmerken. De Chi-Kwadraatsom biedt hier de oplossingen en geeft die weging. De toetsingsgrootheid voor de Chi-kwadraattoets wordt gedefinieerd aan de hand van de volgende formule:

$$\chi^2 = \sum_{i,j} \frac{(O_i j - E_i j)^2}{E_i j}$$

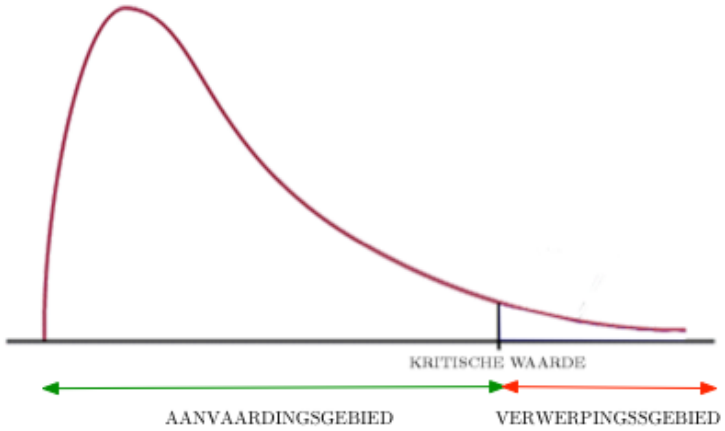
Waarbij $O_i j$ het aantal keer dat het paar (i, j) voorkomt. $E_i j$ stelt de voorspelde waarden voor als de woorden onafhankelijk moesten voorkomen
$E_i j$ wordt bepaald door volgende formule:

$$E_i j = \frac{O_i *}{N} + \frac{O_* j}{N} * N = \frac{O_i * O_* j}{N}$$

met $\frac{O_i *}{N}$ de marginale probabiliteit dat i als eerste deel van het bigram voorkomt en $\frac{O_j *}{N}$ de marginale probabiliteit dat j als tweede deel van het bigram voorkomt. N stelt het totaal aantal bigrams voor. Toegepast op het voorbeeld geeft dit voor het bigram "(heel , goed)":

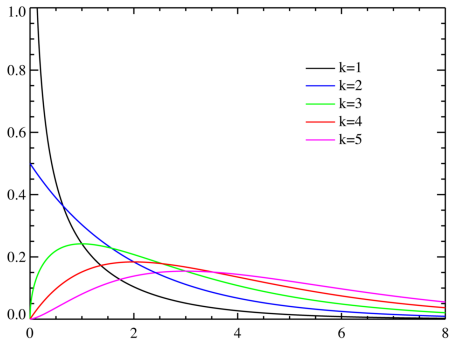
$$E_1 1 = \frac{9 + 3632}{N} + \frac{9 + 7893}{N} * N \approx 0,0085$$

Als laatste onderdeel berekenen we de χ^2-score, bepalen we het aantal vrijheidsgraden en zoeken we de χ^2 distributie op met de berekende vrijheidsgraad. Stel dat het vooropgestelde betrouwbaarheidsinterval 95% bedraagt dan kunnen we de kritische waarde bepalen voor significantielevel α = 0,005. Als de berekende χ^2-score in het verwerpingsgebied ligt, kan de nulhypothese verworpen worden en kan het bigram beschouwd worden als afhankelijk.
Onderstaande afbeelding illustreert hoe de verwerping of aanvaarding van een nulhypothese juist in zijn werking gaat



Figuur 2.1: Illustratie eenzijdige-toets van een χ^2-distributie (Originele afbeelding: http://www.philender.com/courses/intro/notes3/xdist.gif)

Kort samengevat baseert de Chi-kwadraattoets zich op de afwijking tussen de geobserveerde frequentie en de verwachte frequentie. Hoe groter het verschil, hoe waarschijnlijker men de nulhypothese kan verwerpen. En dit is waar men zich bij Bigram Collocatie op gaat baseren.



Figuur 2.2: Chi-square distributies met K vrijheidsgraden (Bron: `http://upload.wikimedia.org/wikipedia/commons/2/21/Chi-square_distributionPDF.png`)

2.2.5 Best feature selection

Als we duizenden documenten verwerken, is het te voorspellen dat er enorm veel woorden algemeen voorkomen in de documenten, maar niet veel informatie bijdragen over het document zelf. Het is sterk vergelijkbaar met de voorgaande techniek in 2.2.2 bij het verwijderen van stopwoorden. Veel voorkomende features kunnen voor het document niet als iets identificerend dienen en zorgen voor ruis in de dataset. Daarom kan men verkiezen om deze low-information features te verwijderen zodanig dat men enkel de features overhoudt die echt iets zeggen over een document. Het bepalen van de informatiewinst kan gebeuren aan de hand van het aantal voorkomens in de verschillende klassen. Als een bepaalde feature voornamelijk in positieve documenten voorkomt en amper in negatieve documenten, kan men afleiden dat deze feature zeer informatief is omtrent positieve documenten. Als metriek om de informatiewinst te meten kan men wederom χ^2 uit 2.2.4 gebruiken. Chi-kwadraat laat ons namelijk toe om de correlatie tussen een bepaalde feature en de klassen te meten.

2.2.6 Latent Semantic Analysis

Latent Semantic Analysis is een wiskundige techniek gebaseerd op statistische berekeningen, waar van aangetoond dat deze zeer nuttig is bij het analyseren van grote collecties tekstdata (Furnas et al. (1988)). Met LSA probeert men een notie te krijgen van de semantische informatie en meer bepaald het semantisch verband tussen woorden. Bijvoorbeeld als we zoeken naar documenten met het woord "economie", willen we ook documenten met "financiën" terugkrijgen. Voor LSA zijn twee woorden semantisch gerelateerd als ze gebruikt worden in dezelfde context. Met het concrete voorbeeld kunnen we zeggen dat er een semantisch verband is tussen twee woorden als ze vaak voorkomen in dezelfde documenten.
Merk op dat bij Latent Semantic Analysis het belangrijk is dat ieder woord naar één concept verwijst.
Analytisch wordt LSA toegepast door **Singular Value Decomposition (SVD)** toe te passen op de terms-documents matrix. SVD is een concept uit de lineaire algebra en zegt dat een matrix A opgesplitst kan worden als een product van matrixen namelijk

$$A = U\Sigma V^T$$

De reductie van de dimensie gebeurt aan de hand van volgend principe. Neem matrix A met rang r.

$$A = U\Sigma V^T = \underbrace{\begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_r\end{bmatrix}}_{\text{Kolommen } A} \underbrace{\begin{bmatrix}\mathbf{u}_{r+1} & \dots & \mathbf{u}_m\end{bmatrix}}_{\text{Nul } A^T} \begin{bmatrix}\sigma_1 & 0 & \dots & 0 & 0 & \dots & 0\\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0\\ \dots & & & & & & \\ 0 & 0 & \dots & \sigma_r & 0 & \dots & 0\\ 0 & 0 & \dots & 0 & 0 & \dots & 0\\ \dots & & & & & & \\ 0 & 0 & \dots & 0 & 0 & \dots & 0\end{bmatrix} \begin{bmatrix}\mathbf{v}_1^T\\ \mathbf{v}_2^T\\ \dots\\ \mathbf{v}_r^T\\ \mathbf{v}_{r+1}^T\\ \dots\\ \mathbf{v}_n^T\end{bmatrix} \begin{matrix}\left.\vphantom{\begin{matrix}a\\b\\c\\d\end{matrix}}\right\}\text{Rijen } A\\ \left.\vphantom{\begin{matrix}a\\b\\c\end{matrix}}\right\}\text{Nul } A\end{matrix}$$

U is de unitaire matrix waarbij men $u_1, u_2, ..., u_r$ de linker singuliere vectors noemt. Deze stellen een document met zijn features voor. V^T is de geconjugeerde getransponeerde matrix van V. $v_1, v_2, ..., v_r$ noemt men de rechter singuliere vectors en stellen de woorden met hun features over alle documenten voor. Σ is een diagonaal matrix met singuliere waarden $\sigma_1, \sigma_2, .., \sigma_r$' op de diagonaal. De reductie van een terms-documents matrix naar een dimensie van K gebeurt door de hoogste K singuliere waarden te nemen in Σ met de overeenkomstige singuliere vectoren uit U en V. Doordat men de dimensionaliteit van de vectoren kan beperken door semantisch

gelijkaardige woorden bijeen te voegen. Dit laat toe om een soort van context groepen te creëren en zo een zeker inzicht te krijgen in de dataset. Het is dan ook gebleken dat SVD toepassen een zeer nuttige eerste stap is bij text mining (**?**), omdat men nieuwe meer efficiënte features krijgt. De nieuwe features geven meer duidelijkheid en inzicht en kunnen dienen als input voor het zelflerende algoritme.

2.3 Leermethode

Voor het experiment hebben moeten we ook het algoritme bepalen dat de data gaat classificeren, ook wel classifier genoemd. Voor het algoritme gaan we beroep doen op de Machine learning en gebruik maken van supervised learning technieken. Deze technieken vereiseen dat men het algoritme eerst traint met een dataset die voorbeelden bevat over het concept dat we willen aanleren. De trainingsset bevat zowel de inputwaarden als de verwachte outputwaarde voor de input en men verwacht dat het algoritme hier verbanden in kan vinden zodanig dat het voor willekeurige inputwaarden de juiste outputwaarde kan bepalen. Ye et al. (2009) toont echter aan dat supervised learning technieken een goede prestatie hebben bij gevoelsanalyse, terwijl dit niet het geval is bij unsupervised learning (Rothfels & Tibshirani (2010)).
Concreter kiezen we voor de Naive Bayes Classifier en de Decision Tree als supervised learning technieken voor het experiment. De Naive Bayes Classifier is een heel praktische aanpak voor bepaalde leerproblemen (Mitchell (1997)). Bijvoorbeeld onderzoekers Michie et al. (1994) tonen aan dat de prestatie van de Naive Bayes Classifier gelijkaardig of in sommige gevallen zelfs beter is dan andere leeralgoritmen, zoals beslissingsbomen en neurale netwerken onderzocht. Decision Trees zijn eveneens een populaire methode en werd ondermeer gebruikt door Zhang et al. (2008) voor een gevoelsanalyse op productrecensies en klanten feedback.

2.3.1 Naive Bayes Classifier

De Naive Bayes Classifier is gebaseerd op Bayesiaans redeneren. Bayesiaans redeneren is een aanpak die gevolgen trekt op basis van probabiliteit. Het is gebaseerd op de veronderstelling dat bepaalde hoeveelheden die ons interesseren probabilistisch verdeeld zijn en door te redeneren over die probabiliteit samen met de trainingsdata er optimale beslissingen kunnen genomen worden.

De werking van de Naive Bayes Classifier is volledig gebaseerd op probabiliteit. Neem als inputwaarden $x_1, x_2, x_3, ..., x_n$ en als de te voorspellen outputwaarde y_{res}. Nu moet de classifier voor de inputwaarden $x_1, x_2, x_3, ..., x_n$ de correct y_{res} voorspellen. Volgens het Bayesiaans redenering is, gebaseerd op $x_1, x_2, x_3, ..., x_n$, y_{res} de outputwaarde met de grootste waarschijnlijkheid. We kunnen dit neerschrijven als:

$$y_{res} = \arg\max_{y_i \in Y} P(y_i | x_1, x_2, x_3, ..., x_n)$$

Aan de hand van het Bayes theorema kunnen we dit herschrijven als

$$y_{res} = \arg\max_{y_i \in Y} \frac{P(x_1, x_2, x_3, ..., x_n | y_i) P(y_i)}{P(x_1, x_2, x_3, ..., x_n)}$$

Merk op $P(x_1, x_2, x_3, ..., x_n)$ is gelijk aan 1, aangezien dit gegeven is dus

$$y_{res} = \arg\max_{y_i \in Y} P(x_1, x_2, x_3, ..., x_n | y_i) P(y_i)$$

De twee componenten kunnen bepaald worden aan de hand van de trainingsset. $P(y_i)$ kunnen we bepalen door het aantal voorkomens van y_i in de trainingsset te tellen. $P(x_1, x_2, x_3, ..., x_n|y_i)$ is moeilijker af te leiden aan de hand van de trainingsset aangezien we meerdere voorkomens van $x_1, x_2, x_3, ..., x_n$ naar y_i moeten hebben om een goede schatting te kunnen maken. Indien we een heel grote trainingsset hebben is dit mogelijk, anders niet. Om dit toch te kunnen afleiden, gaat de Naive Bayes Classifier er van uit dat elke x_i uit $x_1, x_2, x_3, ..., x_n$ onafhankelijk is ten opzichte van de outputwaarde y_i. Wat betekent dat we het product van iedere probabiliteit kunnen nemen en $P(x_1, x_2, x_3, ..., x_n|y_i)$ kunnen herschrijven als $\prod_i P(x_i|y_i)$.

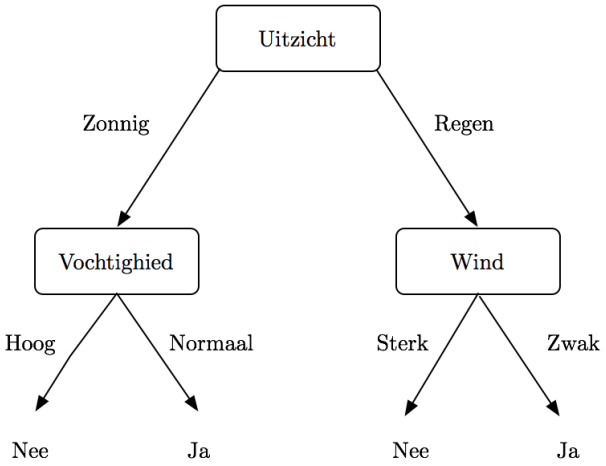
Voor het maken van voorspelling maakt het gebruik van probabiliteit, gebaseerde op de trainingsset en waar het aanneemt dat ieder feature onafhankelijk is tot de outputwaarde. Samengevat kunnen we dit schrijven als

$$y_{NBres} = \arg\max_{y_i \in Y} P(y_i) \prod_i P(x_i|y_i)$$

Ten slotte stellen we de verzameling van al deze probabiliteiten samen als de hypothese van de Naive Bayes Classifier.

2.3.2 Decision Tree

Decision Trees of Beslissingsbomen zijn een van de meest gebruikte en praktische methode voor inductieve gevolgtrekking (Mitchell (1997)). De methode is robust met ruis op de data en houdt rekening met discrete klassen. De classifier gaat een beslissingsboom proberen op te stellen aan de hand van de trainingsdata. Na de training krijgt men een beslissingsboom die de hypothese moet voorstellen. Wanneer het getrainde algoritme onbekende data krijgt, gaat het inductief de output bepalen voor de inputwaarden. Men kan een beslissingsboom voorstellen als een disjuncte set van als-dan regels.
Onderstaande afbeelding is een voorbeeld van zo'n beslissingsboom die bepaald of het weer goed genoeg is om basketbal buiten te spelen. De bladeren van de boom stellen de verschillende outputwaarden voor. In dit geval zien we dat er een boom is opgesteld voor twee discrete klassen namelijk ja en nee. In de nodes staan testen beschreven die de het pad van de inputwaarden naar de outputwaarde bepalen. Merk op dat de bepaling altijd top-down gebeurd.



Figuur 2.3: Voorbeeld van een beslissingsboom

Hoofdstuk 3

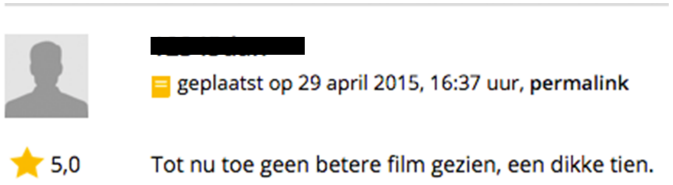
Experiment

Nu we alle achterliggende theorie van het experiment in hoofdstuk 2 gezien hebben, kunnen we van start gaan.
Om te weten of er technieken voor Engelse gevoelsanalyse gelijk toepasbaar zijn op het Nederlands en te achterhalen wat juist deze verschillen zijn, hebben we het onderzoek in drie experimenten opgesplitst. De gevoelsanalyse voor deze experimenten bestaat er telkens uit om te bepalen of de gegeven tekst een positive of negatieve emotie uitdrukt. Als eerste experiment vergelijken we de prestaties van de eerder besproken technieken in hoofdstuk 2 op het Engels met die van het Nederlands. Vervolgens hopen we nog iets meer inzicht te krijgen in de verschillen door een classificatie uit te voeren op basis van Engels en Nederlands geannoteerde woordenlijsten van gevoelens. Als laatste gaan we nog iets dieper in op de Nederlandse gevoelsanalyse en kijken we hoe deze zich gedraagt op data met verschillende onderwerpen.
Voor dat we kunnen beginnen aan de experiment, moeten we eerst over een dataset beschikken waarmee we de algoritmes kunnen trainen en testen. In 3.1 gaan we dieper op de dataset. Uiteindelijk is het beschikken over een goede dataset even belangrijk als het beschikken over goede technieken voor dit experiment.

3.1 De Dataset

Als data voor het experiment is er gekozen voor gebruikersrecensies, meer bepaald film-, muziek- en boekrecensies. Recensies bieden alles wat we nodig hebben. Een recensie drukt of wel positieve, negatieve of neutrale mening uit. Maar omdat er meestal een rating aanwezig is bij de review, is het gemakkelijk om de data automatisch te labelen en enkel de positieve en negatieve reviews op te nemen in onze dataset. Verder door het grote aanbod aan reviewsites is het aanbod aan film-, boek- en muziekrecensies enorm en maakt het gebruik van gebruikersrecensies de recensies nog eens toegankelijk en niet te specifiek.
Voor deze experimenten is de dataset met Engelse filmrecensies afkomstig van een eerder onderzoek door Maas et al. (2011b). De recensies deze website zijn afkomstig van `http://www.imdb.com/` en zijn eveneens afkomstig van gebruikers. Voor de Nederlandse reviews waren er geen datasets beschikbaar en moesten deze gescraped worden. De websites `moviemeter.nl`, `boekmeter.nl` en `muziekmeter.nl` vormde de perfecte bron aan informatie om te scrapen. Ze bevatten allemaal toplijsten met films, boeken of muziekalbums waarop vele gebruikers hun persoonlijke mening plaatsen. Verder was er bij iedere recensie duidelijk een score aanwezig, die het mogelijk maakte om de recensies automatische te labelen. Belangrijk om te vermelden is dat zowel bij het labelen van de Engelse als de Nederlandse dataset dezelfde voorwaarden werd gerespecteerd. Enkel hoog

gepolariseerde recensies werden beschouwd in de dataset. Onderzoek rond polarisatie classificatie (Maas et al. (2011b)) ondersteund deze keuze. Een recensie werd negatief gelabeld als het een score had van 4/10 of minder. Een positieve labeling werd gegeven aan recensies met een score van 6/10 of meer.



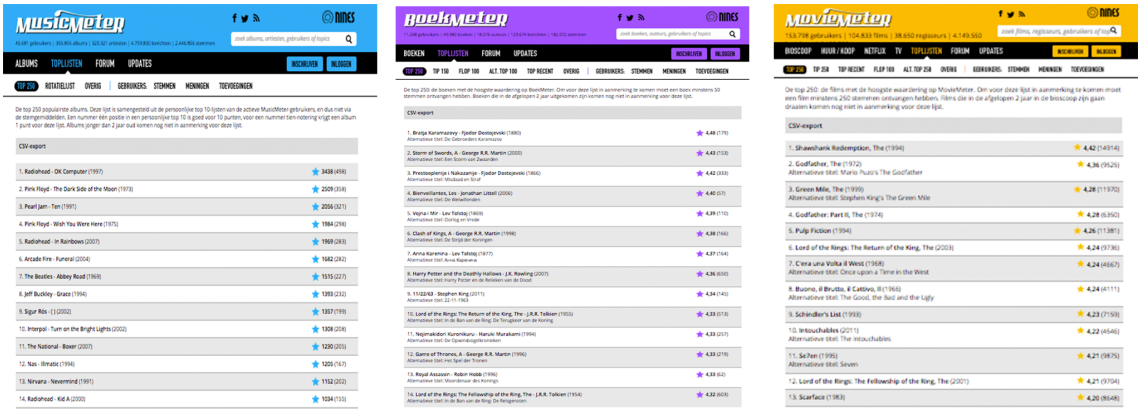
Figuur 3.1: Een voorbeeld van een positieve commentaar op `moviemeter.nl`

Alle Nederlandse recensies zijn afkomstig van de "All Time Top 250"-toplijst op de betreffende website. Onderstaande linkertabel geeft het aantal verzamelde Nederlandse recensies van ieder onderwerp weer, waarbij een onderscheid wordt gemaakt tussen positief en negatief. Analoog wordt dit in de rechtertabel voor de Engelse recensies weergegeven.

	Films	**Muziek**	**Boeken**
Positief	197358	15197	146
Negatief	17978	3019	3719

Tabel 3.1: Aantal verzamelende recensies

Om nog een beter inzicht te krijgen over de dataset geven onderstaande tabellen nog wat extra statistieken weer over de datasets.
HIER KOMEN NOG TABELLEN



Figuur 3.2: de “All Time Top 250”-toplijsten op de websites

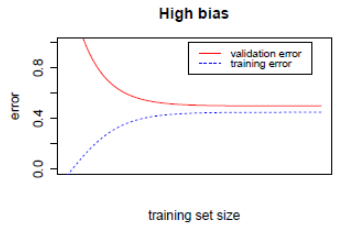
Voor het tweede experiment hebben we als woordenlijsten het *Opinion lexicon* gebruikt, dat voor het eerst werd samengesteld door Hu & Liu (2004). De woordenlijsten bestaan uit een lijst met negatieve en een lijst met positieve woorden. De lijsten bevatten in totaal ongeveer 6800 woorden en zijn enkel in het Engels verkrijgbaar. De Nederlandse woordenlijsten hebben we verkregen door de Engelse lijsten te vertalen met behulp van Google vertalen.

Onderstaande tabel geeft weer hoe de woordenlijsten zich tegenover elkaar verhouden.
HIER KOMT NOG EEN TABEL

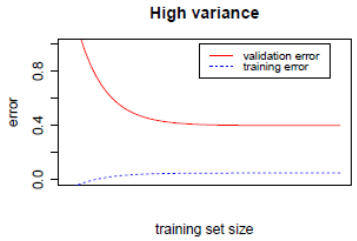
3.2 Naive Bayes Classifier met hetzelfde onderwerp voor trainings- en testset

Als eerste experiment gaan we kijken hoe de prestaties van een classifier zijn bij het trainen en testen met data van dezelfde soort. Bijvoorbeeld we trainen met een trainingsset van filmrecensies en testen het getrainde algoritme op een testset van filmrecensies. Dit gaan we voor zowel film-, boek- als muziekrecensies doen. Als classifier nemen we de Naive Bayes Classifier. Zoals vermeld in sectie 2.3.1 is dit een goede eerste keuze als leermethode. Verder geven we de data mee aan de classifier als een Bag of Words met TFIDF-weging.
Algemeen voor alle experimenten zijn de resultaten berekend als gemiddelde over dertig runs. Dit wil zeggen dat er telkens bij iedere run een nieuwe Naive Bayes Classifier wordt aangemaakt en vervolgens getraind en getest wordt met een andere trainings- en testset als de andere runs. Om te verzekeren dat bij elke run de trainingsset en testset verschillend zijn wordt bij iedere run de trainingsset en testset aangemaakt door een bepaald aantal willekeurig uit de grote pool van recensies te selecteren. Na het uitvoeren van die runs wordt hier het gemiddelde van genomen. De resultaten van experimenten bestaan uit de classificatieprecisie van zowel de trainings- als testset, de standaard afwijking, de confusion matrix en het betrouwbaarheidsinterval voor 95%. Het betrouwbaarheidsinterval wordt als *(gemiddeld ; linkerlimiet ; rechterlimiet)* genoteerd. Een confusion matrix geeft aan hoeveel van elke outputmogelijkheid er juist zijn geïdentificeerd door de classifier en hoeveel er fout als juist zijn geïdentificeerd. Als laatste wordt er ook de learning curve bekeken om over- of underfitting uit te sluiten. De learning curve geeft het verloop van de precisie van de classifier weer voor de trainings- en validatieset. Op basis van het verloop en de

ligging van de curve kan men detecteren of men te maken heeft met over- of underfitting. Figuur 3.3 en 3.4 illustreren hoe men overfitting kan herkennen aan de hand van de learning curve. Merk op zowel de trainingsset als de testset altijd evenwichtig zijn verdeeld. Dit wil zeggen dat er telkens 1/2 van het totaal aantal samples bestaat uit positieve recensies en 1/2 uit negatieve recensies.



Figuur 3.3: Learning curve van een dataset met hoge bias. Wat duidt op underfitting [Bron: VUB-Cursus Machine Learning]



Figuur 3.4: Learning curve van een dataset met hoge variantie. Wat duidt op overfitting [Bron: VUB-Cursus Machine Learning]

		voorspelde waarde	
		p	**n**
eigelijke waarde	**p′**	Waar Positief	Vals Negatief
	n′	Vals Positief	Waar Negatief

Tabel 3.2: Illustratie van de confusion matrix

3.2.1 Filmrecensies als trainings- en testset

Eerst trainen en testen we de Naive Bayes Classifier met filmrecensies. De trainingsset bestaat uit 6000 samples en de testset uit 2000 samples. Zoals eerder vermeld werden deze samples at random geselecteerd en is het volgende resultaat het gemiddelde van 30 runs.

Standaard afwijking = 0,0094
95% betrouwbaarheidsinterval = (0,7064 ; 0,7030 ; 0,7101)

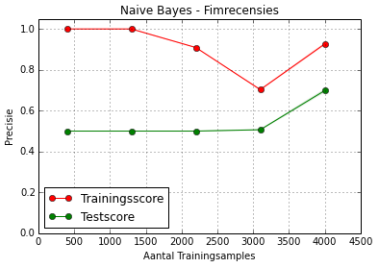
	Precisie
Trainingsset	90,52%
Testset	70,66%

Tabel 3.3: Classificatieprecisie Naive Bayes Classifier, getraind op filmrecensies

	P	**N**
P'	824	175
N'	410	589

Tabel 3.4: Confusion matrix van de testset door de Naive Bayes Classifier, getraind op filmrecensies

Zoals men kan zien aan de resultaten zijn de prestaties goed. Een classificatieprecisie van 70% voor een onbekende set met filmrecensies is een goede prestatie. Verder is het betrouwbaarheidsinterval heel klein, wat maakt dat we met 95% kunnen zeggen dat de classificatie van filmrecensies door een Naive Baiyes Classifier, getraind op filmrecensies, met een precisie tussen 70% en 71% gebeurd. De confusion matrix geeft ons ook een inzicht in wat er juist en fout geclassificeerd is. We zien dat de classifier overwegend beter positieve recensies kan identificeren dan negatieve.
Ten slotte wat we ook kunnen afleiden uit de cijfers, waar we zien dat zowel de test- als trainingsset goed presteren, zien we aan de learning curve dat we geen over- of underfitting hebben.



Figuur 3.5: Learning curve van de training van de Naive Bayes Classifier op filmrecensies

3.2.2 Muziekrecensies als trainings- en testset

Nu trainen en testen we de Naive Bayes Classifier met muziekrecensie. De trainingsset bestaat uit 6000 samples en de testset uit 2000 samples. Wederom werden deze samples at random geselecteerd en is het volgende resultaat het gemiddelde van 30 runs.

Standaard afwijking = 0,0096
95% betrouwbaarheidsinterval = (0,8262 ; 0,8226 ; 0,8299)

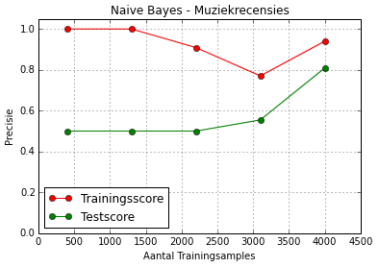
Hier zien we eveneens goede resultaten. Een classificatieprecisie van 82% voor een onbekende set met muziekrecensies is eveneens een goede prestatie. Wederom is het betrouwbaarheidsinterval heel klein, wat maakt dat we met 95% kunnen zeggen dat de classificatie van muziekrecensies door een Naive Baiyes Classifier, getraind op muziekrecensies, met een precisie van 82% gebeurd. De confusion matrix toont ons opnieuw dat de classifier beter om kan met positieve recensies. Wederom geeft onderstaande learning curve uitsluiting van over- of underfitting .

	Precisie
Trainingsset	93,44%
Testset	82,62%

Tabel 3.5: Classificatieprecisie Naive Bayes Classifier, getraind op muziekrecensies

	P	**N**
P'	879	120
N'	227	772

Tabel 3.6: Confusion matrix van de testset door de Naive Bayes Classifier, getraind op muziekrecensies



Figuur 3.6: Learning curve van de training van de Naive Bayes Classifier op muziekrecensies

3.2.3 Boekrecensies als trainings- en testset

Als laatste trainen en testen we de Naive Bayes Classifier met boekrecensies. De trainingsset bestaat uit 218 samples en de testset uit 74 samples. De samples zijn wederom at random geselecteerd en het resultaat is het gemiddelde van 30 runs.

Standaard afwijking = 0,0640
95% betrouwbaarheidsinterval = (0,7176 ; 0,6932 ; 0,7419)

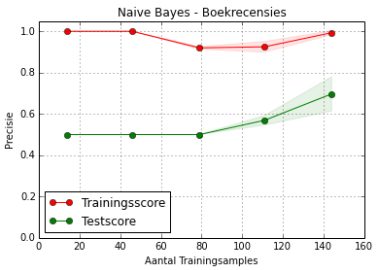
	Precisie
Trainingsset	99,43%
Testset	71,76%

Tabel 3.7: Classificatieprecisie Naive Bayes Classifier, getraind op boekrecensies

	P	**N**
P'	31	5
N'	15	21

Tabel 3.8: Confusion matrix van de testset door de Naive Bayes Classifier, getraind op boekrecensies

Hier zien we ook goede resultaten. Een classificatieprecisie van 72% voor een onbekende set met boekrecensies is een eveneens een goede prestatie. Hier zien we wel dat het betrouwbaarheidsinterval ruimer en is met 95% zekerheid te zeggen, dat de classificatieprecisie zich tussen de 69% en 74% situeert, wat nog altijd acceptabel is. De confusion matrix toont ons opnieuw dat de classifier beter om kan met positieve recensies, al valt dit te nuanceren, aangezien we maar een hele klein pool hebben aan boekrecensies ten opzichte van de rest. Onderstaande learning curve sluit opnieuw over- en underfitting uit.



Figuur 3.7: Learning curve van de training van de Naive Bayes Classifier op boekrecensies

Nu we de prestaties weten van de Naive Bayes classifier met als trainings- en testset hetzelfde onderwerp. Kunnen we eens kijken wat de prestaties zijn met een verschillend onderwerp voor de trainingsset en testset.

3.3 Naive Bayes Classifier met een verschillend onderwerp voor trainings- en testset

In 3.2 zagen we al dat de classificatie met een Naive Bayes Classifier, waarbij trainings- en testset tot hetzelfde onderwerp behoren, goede resultaten oplevert. Nu gaan we kijken of dit ook het geval is wanneer trainingsset en testset verschillend zijn. Wederom alle samples worden at random geselecteerd en de resultaten weerspiegelen telkens het gemiddelde van 30 runs.

3.3.1 Filmrecensies als trainingsset

Als eerste nemen we een getrainde Naive Bayes Classifier op filmrecensies en bekijken we de resultaten op een testset van muziek- en boekrecensies. De Naive Bayes Classifier is telkens getraind met 6000 samples.

Muziekrecensies als testset

De testset bestaat uit 2000 samples waarvan 1/2 positieve en 1/2 negatieve recensies.

Standaard afwijking = 0,01467
95% betrouwbaarheidsinterval = (0,6207 ; 0,6152 ; 0,6263)

	Precisie
Testset	62,07%

Tabel 3.9: Classificatieprecisie Naive Bayes Classifier, getraind op filmrecensies, getest op muziekrecensies

	P	**N**
P'	655	345
N'	413	586

Tabel 3.10: Confusion matrix van de testset ,bestaande uit muziekrecensies, door de Naive Bayes Classifier, getraind op filmrecensies

De prestatie is minder dan de prestaties in 3.2, maar 62% is zeker aanvaardbaar. Ook het

betrouwbaarheidsinterval is klein wat wil zeggen dat we met 95% zekerheid kunnen zeggen dat een Naive Bayes Classifier getraind op filmrecensies, muziekrecensies net 61%-62% precisie kan classificeren.

Boekrecensies als testset

De testset bestaat uit 146 samples waarvan 1/2 positief en 1/2 negatief.

Standaard afwijking = 0,03714
95% betrouwbaarheidsinterval = (0,6586 ; 0,6446 ; 0,6728)

	Precisie
Testset	65,87%

Tabel 3.11: Classificatieprecisie Naive Bayes Classifier, getraind op filmrecensies, getest op boekrecensies

	P	**N**
P'	54	18
N'	31	41

Tabel 3.12: Confusion matrix van de testset, bestaande uit boekrecensies, door de Naive Bayes Classifier, getraind op filmrecensies

De prestatie is in dezelfde lijn als de resultaten bij muziek.

3.3.2 Muziekrecensies als trainingsset

Als tweede nemen we een getrainde Naive Bayes Classifier op muziekrecensies en bekijken we de resultaten op een testset van film- en boekrecensies. De Naive Bayes Classifier is telkens getraind met 6000 samples.

Filmrecensies als testset

De testset bestaat uit 2000 samples waarvan 1/2 positief en 1/2 negatief.

Standaard afwijking = 0,01146
95% betrouwbaarheidsinterval = (0,6107 ; 0,6063 ; 0,6150)

	Precisie
Testset	61,07%

Tabel 3.13: Classificatieprecisie Naive Bayes Classifier, getraind op muziekrecensies, getest op filmrecensies

	P	**N**
P'	691	308
N'	469	530

Tabel 3.14: Confusion matrix van de testset ,bestaande uit filmrecensies, door de Naive Bayes Classifier, getraind op muziekrecensies

De resultaten zijn aanvaardbaar met een classificatieprecisie van 61% en een klein betrouwbaarheidsinterval van 95%.

Boekrecensies als testset

De testset bestaat uit 146 samples waarvan 1/2 positief en 1/2 negatief.

Standaard afwijking = 0,03519
95% betrouwbaarheidsinterval = (0,6146 ; 0,6012 ; 0,6280)

	Precisie
Testset	61,46%

Tabel 3.15: Classificatieprecisie Naive Bayes Classifier, getraind op muziekrecensies, getest op boekrecensies

	P	**N**
P'	43	29
N'	26	46

Tabel 3.16: Confusion matrix van de testset, bestaande uit boekrecensies, door de Naive Bayes Classifier, getraind op muziekrecensies

Een gelijkaardige prestatie als filmrecensies, met een gemiddelde classificatieprecisie van 61% en eveneens een klein betrouwbaarheidsinterval van 95%.

3.3.3 Boekrecensies als trainingsset

Als laatste nemen we een getrainde Naive Bayes Classifier op boekrecensies en bekijken we de resultaten op een testset van film- en muziekrecensies. De Naive Bayes Classifier is telkens getraind met 276 samples.

Filmrecensies als testset

De testset bestaat uit 2000 samples waarvan 1/2 positief en 1/2 negatief.

Standaard afwijking = 0,01812
95% betrouwbaarheidsinterval = (0,5625 ; 0,5556 ; 0,5693)

	Precisie
Testset	56,25%

Tabel 3.17: Classificatieprecisie Naive Bayes Classifier, getraind op boekrecensies, getest op filmrecensies

	P	**N**
P'	539	460
N'	414	585

Tabel 3.18: Confusion matrix van de testset ,bestaande uit filmrecensies, door de Naive Bayes Classifier, getraind op boekrecensies

Met 56% kunnen we spreken van een slechte prestatie. Net iets meer dan de helft van de classificaties wordt juist geclassificeerd. Het betrouwbaarheidsinterval is ook klein, wat betekent dat we met 95% zekerheid kunnen zeggen dat de classificatieprecisie zich tussen 56% - 57% situeert. Wat nogmaals de slechte prestatie bevestigd.

Muziekrecensies als testset

De testset bestaat uit 2000 samples waarvan 1/2 positief en 1/2 negatief.
Standaard afwijking = 0,01448

95% betrouwbaarheidsinterval = (0,5647 ; 0,5592, 0,5702)

	Precisie
Testset	56,47%

Tabel 3.19: Classificatieprecisie Naive Bayes Classifier, getraind op boekrecensies, getest op muziekrecensies

	P	**N**
P'	604	395
N'	475	524

Tabel 3.20: Confusion matrix van de testset, bestaande uit muziekrecensies, door de Naive Bayes Classifier, getraind op boekrecensies

Een classificatieprecisie van 56% is niet goed. Dit is analoog als bij voorgaande sectie met als testset filmrecensies

De resultaten op muziek- en filmrecensies zijn met 56%, minder als de vorige in 3.3.1 en 3.3.3, waar we rond de 60% liggen.

3.4 Conclusie experiment

Nu we alle mogelijke combinaties van training en testing hebben uitgevoerd is het interessant om alle resultaten naast elkaar te leggen. Onderstaande tabel geeft nog eens alle classificatiescores in een kruistabel weer.

	Films	**Muziek**	**Boeken**
Films	70,66%	61,00%	56,25%
Muziek	62,07%	82,62%	56,47%
Boeken	65,87%	61,46%	71,76%

Tabel 3.21: Kruistabel van alle classificatieresultaten uit 3.3 en 3.2 met de kolommen het onderwerp van de trainingsset en de rijen het onderwerp van de testset.

Op basis van de tabel kunnen we zeggen dat het trainen en testen met het zelfde onderwerp het beste resultaat geeft. Verder presteren muziek en films goed op een vreemde testset. Hierbij bedoelen we een testset die over een andere onderwerp gaat dan waar het algoritme op getraind is. Boeken presteren hier minder. Dit kan te wijten zijn aan de kleinere dataset, waardoor er een bias of voorkeur ontstaat op bijvoorbeeld auteursnamen. Die bias op auteursnamen kan helpen bij het classificeren van boeken om positieve of negatieve recensies te herkennen, maar gaat niet helpen in de classificatie van een set over een ander onderwerp. Ook de prestatie van muziek springt in het oog met meer dan 10% verschil tussen de andere classifiers met de trainingsset en testset over hetzelfde onderwerp. Maar het daalt wel 20% wanneer het een vreemde set moet classificeren. Dit duidt eveneens op een bias of voorkeur op een bepaald feature die specifiek is voor muziek waardoor het classificeren van muziekrecensies veel beter gaat. Nog een interessant inzicht is dat filmrecensies een goede trainingsset blijken te zijn voor boekrecensies. Ondanks de kleine dataset van boeken, kunnen we toch aan de hand van het betrouwbaarheidsinterval met 95% zekerheid zeggen dat een getrainde Naive Bayes Classifier op filmrecensies, boekrecensies met een precisie tussen de 64% en 67% zal classificeren.

Nog een andere manier om de resultaten bij elkaar te leggen, is het bekijken van de confusion matrixen. We kunnen hier een percentueel gemiddelde van nemen en kijken hoe gemiddeld de classificatie verloopt bij sectie 3.2 waar de sets over hetzelfde onderwerp zijn en sectie 3.3, waar ze verschillend zijn.

	P	**N**
P'	43%	6%
N'	18%	31%

Tabel 3.22: Gemiddelde confusion matrix in percent voor een Naive Bayes Classifier, waar trainings- en testset over hetzelfde onderwerp gaan

	P	**N**
P'	32%	18%
N'	21%	29%

Tabel 3.23: Gemiddelde confusion matrix in percent voor een Naive Bayes Classifier, waar trainings- en testset over een verschillend onderwerp gaan

Voor beide matrixen, ziet men duidelijk dat positieve recensies beter geïdentificeerd worden. Na het herbekijken van de datasets, is een mogelijk verklaring dat mensen zich bij een positieve recensie zich veel expressiever en uitgebreider uitdrukken dan bij een negatieve recensie. Hierdoor krijgt de classifier meer informatie over de features van een positief document waardoor het beter het concept “Positief” kan bepalen

Hoofdstuk 4

Conclusie

In deze bachelorproef hadden we als onderwerp gevoelsanalyse in het Nederlands. De theorie gaf ons de basis voor het onderzoek, waarbij werd uitgelegd hoe we de gevoelsanalyse konden uitvoeren aan de hand van machine learning. Er werd toegelicht hoe we de data konden meegeven en optimaliseren voor het zelflerende algoritme. Vervolgens werd er ingegaan op de zelflerende algoritmes, waarbij de Naive Bayes Classifier en Beslissingsbomen werden besproken. We zagen ook bias en variantie, twee begrippen uit de machine learning waar men rekening mee moet houden tijdens de analyse.
Vervolgens kwamen we tot het experiment in hoofdstuk 3. Voor het experiment trachtten we een eigen gevoelsanalyse uit te voeren op Nederlandse tekst aan de hand van eenvoudige machine learning technieken. Concreter gingen we film- ,boek- en muziekrecensies analyseren en bepalen of de recensie positief of negatief is. Voor de analyse werd er beroep gedaan op de Naive Bayes Classifier. Met de Naive Bayes Classifier hebben met alle mogelijke permutaties tussen trainings- en testset geëxperimenteerd. Als we alle resultaten naast elkaar leggen en vergelijken kunnen we besluiten dat het mogelijk is om een gevoelsanalyse aan de hand van de Naive Bayes Classifier uit te voeren, waarbij de beste prestatie zich voordoet wanneer de trainingsset en testset over hetzelfde onderwerp gaan. Bijvoorbeeld wanneer men het algoritme traint op muziekrecensies en vervolgens een gevoelsanalyse uitvoert op muziekrecensies. Als laatste hebben we ook opgemerkt dat over het algemeen positieve recensies juister geïdentificeerd worden.

Literatuur

Bullinaria, J. A. (2004). *Bias and variance, under-fitting and over-fitting.* http://www.cs.bham.ac.uk/~jxb/NN/l9.pdf. (Accessed: 2014-27-05)

Furnas, G. W., Deerwester, S., Dumais, S. T., Landauer, T. K., Harshman, R. A., Streeter, L. A. & Lochbaum, K. E. (1988). Information retrieval using a singular value decomposition model of latent semantic structure. In *Proceedings of the 11th annual international acm sigir conference on research and development in information retrieval* (pp. 465–480).

Goodness-of fit test, a nonparametric test. (z. j.). http://www2.cedarcrest.edu/academic/bio/hale/biostat/session22links/basics.html. (Accessed: 2015-05-23)

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J. & Tibshirani, R. (2009). *The elements of statistical learning* (Dl. 2) (nr. 1). Springer.

Hu, M. & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth acm sigkdd international conference on knowledge discovery and data mining* (pp. 168–177).

Landauer, T. K., Foltz, P. W. & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, *25*(2-3), 259-284. Verkregen van http://dx.doi.org/10.1080/01638539809545028 doi: 10.1080/01638539809545028

Latent semantic analysis (lsa) tutorial. (z. j.). http://www.puffinwarellc.com/index.php/news-and-articles/articles/33-latent-semantic-analysis-tutorial.html?showall=1. (Accessed: 2014-15-11)

Liu, M. & Yang, J. (2012). An improvement of tfidf weighting in text categorization. *International Proceedings of Computer Science and Information Technology*, 44–47.

Lohr, S. (2012). The age of big data. *New York Times*, *11*.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011a). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1* (pp. 142–150).

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011b, June). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies* (pp. 142–150). Portland, Oregon, USA: Association for Computational Linguistics. Verkregen van http://www.aclweb.org/anthology/P11-1015

Manning, C. D., Raghavan, P. & Schütze, H. (2008). *Introduction to information retrieval* (Dl. 1). Cambridge university press Cambridge.

Manning, C. D. & Schütze, H. (1999). *Foundations of statistical natural language processing.* MIT press.

Mantrach Amin, H. B. M. S., Nicolas Vanzeebroeck. (z. j.). *Machine learning course ulb: Text mining.* `https://ai.vub.ac.be/sites/default/files/textmining2011.pdf`. (Accessed: 2014-15-11)

Martineau, J. & Finin, T. (2009). Delta tfidf: An improved feature space for sentiment analysis..

McKinney, W. (2012). *Python for data analysis: Data wrangling with pandas, numpy, and ipython.* "O'Reilly Media, Inc.
.

Michie, D., Spiegelhalter, D. J. & Taylor, C. (1994). *Machine learning, neural and statistical classification.*

Mitchell, T. M. (1997). Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, *45*.

Ng, A. (z. j.). *Machine learning course.* `https://class.coursera.org/ml-005/lecture/preview`. (Accessed: 2014-15-11)

Paltoglou, G. & Thelwall, M. (2010). A study of information retrieval weighting schemes for sentiment analysis. In *Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 1386–1395).

Pang, B. & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, *2*(1-2), 1–135.

Pang, B., Lee, L. & Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the acl-02 conference on empirical methods in natural language processing-volume 10* (pp. 79–86).

Pedersen, T. (2001). A decision tree of bigrams is an accurate predictor of word sense. In *Proceedings of the second meeting of the north american chapter of the association for computational linguistics on language technologies* (pp. 1–8).

Petitpierre, D. & Russell, G. (1995). Mmorph-the multext morphology program. *Multext deliverable report for the task*, *2*(1).

Rothfels, J. & Tibshirani, J. (2010). Unsupervised sentiment classification of english movie reviews using automatic selection of positive and negative sentiment items. *CS224N-Final Project*.

Samuel, A. L. (2000). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, *44*(1.2), 206–226.

Turney, P. D., Pantel, P. et al. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, *37*(1), 141–188.

A tutorial on clustering algorithms. (z. j.). `http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/index.html`. (Accessed: 2015-01-11)

Wakade, S., Shekar, C., Liszka, K. J. & Chan, C.-C. (2012). Text mining for sentiment analysis of twitter data. In *International conference on information and knowledge engineering (ikeÕ12)* (pp. 109–114).

Wang, L. & Wan, Y. (2011). Sentiment classification of documents based on latent semantic analysis. In *Advanced research on computer education, simulation and modeling* (pp. 356–361). Springer.

Ye, Q., Zhang, Z. & Law, R. (2009). Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert Systems with Applications*, *36*(3), 6527–6535.

Zhang, C., Zuo, W., Peng, T. & He, F. (2008). Sentiment classification for chinese reviews using machine learning methods based on string kernel. In *Convergence and hybrid information technology, 2008. iccit'08. third international conference on* (Dl. 2, pp. 909–914).