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 populaire gegeven binnen de Machine Learning. Over gevoelsanalyse op de Engelse taal vindt men voldoende naslag werk (http://nlp.stanford.edu/sentiment/), maar in het Nederlands is dit eerder beperkt. Dit heeft deels te maken met het kleine bereik van de Nederlandse taal wat het onderzoek ernaar al minder interessant maakt en als men er onderzoek naar zou doen, dit vaak door een bedrijf wordt uitgevoerd waarbij de resultaten onder bedrijfsgeheim vallen. Daarom concentreren wee bij deze bachelorproef op gevoelsanalyse op de Nederlandse taal. We experimenteren met enkele algemene technieken uit de Machine Learning die ons een degelijke gevoelsanalyse kunnen opleveren en proberen ten slotte een besluit te trekken of we met enkele algemene technieken effectief een acceptabele gevoelsanalyse op het Nederlands kunnen uitvoeren.

Dank woord

Ik wil graag mijn begeleiders, Maarten Deville en Peter Vranckx, van harte bedanken voor de grote steun en inzet gedurende het hele jaar. Ze waren altijd beschikbaar gedurende het hele jaar om op al mijn vragen een snel antwoord te geven. Als laatste wens ik mijn dank uit te drukken aan mijn promotor, Yann-Michaël De Hauwere. Bij de korte evaluaties was hij altijd aanwezig en stond altijd bij voor raad en daad.

Inhoud

Hoofdstuk 1

Introductie

Vandaag de dag is informatie nog nooit zo belangrijk geweest. Iedere dag komt er ook enorm veel informatie bij. Kijk maar naar social media, waar iedere dag duizende gebruikers hun mening uiten over alledaags dingen. Het is dan ook zeer interessant om die data te analyseren en daar een zekere kennis uit te vergaren. Vanwege de grote hoeveelheid aan data is het onmogelijk om een programma manueel te schrijven, dat enige kennis uit die data kan halen. Machine learning biedt hier de oplossing. Dit is een onderzoeksdomein binnen de Artificiële Intelligentie dat zich toespitst op zelflerende algoritmes. In deze Bachelorproef onderzoeken we of we met enkele technieken uit de machine learning een goede analyse kunnen uitvoeren op tekst. Concreter wordt er gefocust op de gevoelsanalyse van Nederlandse tekst, waarbij een programma beslist of de gegeven tekst een positief of negatief gevoel uitdrukt. Zoals vermeld voeren we het onderzoek uit met technieken uit de Machine Learning, specifieker binnen de Machine learning situeren we ons onder supervised learning, waarbij we het algoritme trainen 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 zelflerende algoritme hier verbanden in kan vinden zodanig dat het voor willekeurige inputwaarden de juiste outputwaarde kan bepalen.
We in dit onderzoek een gevoel af te leiden, wat men kan beschouwen een het labelen van tekst met een bepaald. Wat ons binnen de machine learning tot een classificatieprobleem brengt waarbij verwachte outputwaarde voor bepaalde inputwaarden zich beperkt tot een kleine set van mogelijkheden.
Nu dat we weten waar we ons precies situeren namelijk binnen de Machine learning en meer bepaald bij supervised learning met classificatieproblemen, kunnen we de vraag stellen welke data we juist nodig hebben voor de gevoelsanalyse op uit te voeren. We hadden al bepaald dat we deze uitvoeren op Nederlandse tekst, maar dit is niet specifiek genoeg. Het vinden van Nederlandse tekst is geen probleem. Men kan een grote hoeveelheid tekst uit krantenartikels, boeken, handleidingen verzamelen, maar dit volstaat niet. Er moet specifiek gezocht worden naar teksten die duidelijke informatie geeft over een gevoel of een emotie en ze moet duidelijk met dat gevoel gelabeld zijn zodanig dat het algoritme hieruit kan leren. De keuze om de gevoelens te veralgemenen en onder te verdelen in twee groepen namelijk positieve en negatieve gevoelens, geeft ons duidelijke labeling. Verder geeft het inzicht over waar men de Nederlandse tekst moet gaan zoeken. Een eerste idee was het verzamelen van tweets rond een bepaalde case bijvoorbeeld de nieuwe uurregeling van de NMBS en deze manueel te labelen of eventueel de labeling te automatiseren aan de hand van hashtags. Na het nader bekijken van deze dataset is gebleken dat er nogal veel sarcasme heerst op Twitter en niet de informatie bevat die we zoeken. Het

verzamelen van reviews en meer bepaald over films, boeken en muziek biedt de oplossing. Reviews bieden alles wat men nodig heeft voor het onderzoek. Een review is of wel positief of negatief en de rating die meestal aanwezig is bij een review stelt ons in staat om de review correct te labelen. De keuze om films, boeken en muziek is een beredeneerde keuze. Het aanbod is enorm, meestal niet te specifiek en het is toegankelijk. De datasets die in deze bachelorproef worden gebruikt zijn afkomstig van moviemeter.be, boekmeter.be en muziekmeter.be . Productreviews was ook een mogelijke optie, maar deze bleken te specifiek te zijn waardoor het moeilijk wordt om het algoritme op algemeen sentiment te trainen.
Samengevat is deze thesis opgesplitst in drie delen namelijke de achtergrondinformatie, het onderzoek en de conclusie. Als achtergrondinformatie worden alle technieken die tijdens die het onderzoek worden toegepast besproken. Zoals eerder besproken zijn dit technieken die zich onder supervised learning situeren en een oplossing bieden voor classificatie problemen. Vervolgens bespreken we het onderzoek, waarbij we de reviews van films, boeken en muziek als datatset nemen. Als laatste vormen we een conclusie over de mogelijkheid om met enkele technieken uit de machine learning een geslaagde gevoelsanalyse kunnen uitvoeren op Nederlandse tekst.

Hoofdstuk 2

Achtergrondinformatie

Zoals eerder vermeld situeren we ons voor het onderzoek we onder supervised learning waarbij we inputwaarden, in dit geval Nederlandse tekst, mappen naar een bepaalde outputwaarde. Die outputwaarde zou dan een gevoel moet omvatten. Concreter hadden gezegd dat we Nederlandse reviews nemen en deze mappen naar de outputwaarde negatief of positief.

In dit hoofdstuk wordt er achtergrondinformatie gegeven over wat er juist allemaal wordt gebruikt tijdens het onderzoek en hoe dit juist in elkaar zit. We behandelen de voorstelling van de documenten waarbij we kijken hoe we die data nu juist kunnen voorstellen om het te laten verwerken door het algoritme. Vervolgens bekijken we de technieken van pre-processing. Er zijn enkele optimalisaties die kunnen uitgevoerd worden op de dataset voordat men het gebruikt voor de training van het algoritme en deze gaat men bespreken. Dan worden de classifiers besproken. Dit zijn de algoritme die een bepaalde techniek gebruiken om te leren classificeren en deze worden wederom besproken. Als laatste behandelen we de mogelijke valkuilen waar we tijdens het onderzoek rekening mee moeten houden.

2.1 Voorstelling dataset

De voorstelling van de data vormt al een eerste struikelblok wanneer we tekst gaan analyseren en hier moet over nagedacht worden. Bij dit onderzoek is er gekozen voor de vector space methode omdat deze voorstelling algemeen aanvaard wordt door de meeste classifiers, pre-processing en optimalisaties toelaat en intuïtief in elkaar zit.

2.1.1 Vector Space Methode

De vector space methode 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 $\mathbf{d}_j$ 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 het 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. Dit resulteert in een matrix waarbij iedere kolom een document voorstelt.

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

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 onderzoek stellen we onze data voor aan de hand van een documents-term matrix De voorstelling brengt geeft ons inzicht en biedt veel meer mogelijkheden om de data te analyseren. We kunnen bijvoorbeeld al eenvoudig een afleiding maken over het verband tussen documenten. Door de euclidische afstand te bepalen tussen twee rijen in de DTM kunnen we al zien of de documenten gelijkaardig zijn of niet. Stel men heeft twee documenten met een kleine euclidische afstand. Dit wil eigelijk zeggen dat de vectorvoorstelling van de documenten gelijkaardig is, wat neerkomt op een overeenkomstige woordfrequenties en dus bijvoorbeeld 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. Vaak is het nog altijd moeilijk om verschillend groepen tussen de documenten te onderscheiden. Daarom kan men nog extra verfijningen zoals bijvoorbeeld ***term weighting***, ***Latent Semantic Analysis (LSA)***...

2.2 Technieken voor Pre-Processing

Het voor verwerken of pre-processen van een dataset kan op verschillend manieren gebeuren. Men kan bepaalde data filteren zoals het verwijderen van stopwoorden en leestekens of data toevoegen zoals bij Bigram Collocaties. Men kan ook het DTM analyseren en en een nieuwe weging van de features introduceren zoals bijvoorbeeld bij term weighting of LSA gebeurd. Het doel van de pre-processing is de data zo goed mogelijk voor te bereiden zodaning het algoritme een duidelijk beeld kan krijgen over hoe en naar wat het de data moet classificeren.

2.2.1 Bag of Words

Bag of Words is de eenvoudigste methode die er is en is het principe waarop de vector space methode zich baseert. Ieder document wordt beschouwd als een zak met woorden, waarbij de woorden in het document de kenmerken of de features van het document voorstellen. Bag of

Words wordt beschouwd als de eenvoudigste techniek, omdat bij de techniek geen rekening houdt met spelling, woordorde of voorkomens. Dit gaat wel het geval zijn bij andere technieken.

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 k̈loptën ëigenlijkz̈eggen 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 enkel rekening gehouden wordt 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 recensist de film fantastisch vond, het zal niet in elk document voorkomen en is waardevolle informatie. Term weighting zal dus bij dit voorbeeld "excellent" een grotere gewicht toewijzen als "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. Deze matrix wordt bijvoorbeeld vaak gebruikt om de gelijkenissen tussen twee documenten te bepalen op basis van cosinusgelijkenis.

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. 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 ondermeer voor Bigram collocatie gebruiken is dat het parametervrije toets is. Hiermee bedoeld men dat er voor de chi-kwadraattoets bij de start van de toets geen aannames over de populatie of gemiddelde 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. 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 alternative hypothese aannemen, namelijk dat de woorden afhankelijk zijn van elkaar.

Om de afhankelijkheid van woorden te bekijken kijken we naar enkele gegevens namelijk:

- 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.

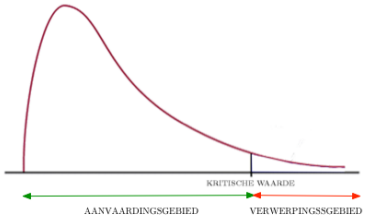
Als we voor het voorbeeld (heel , goed) bovenstaande gegevens in een kruistabel gieten krijgen we volgende 2x2 tabel:

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

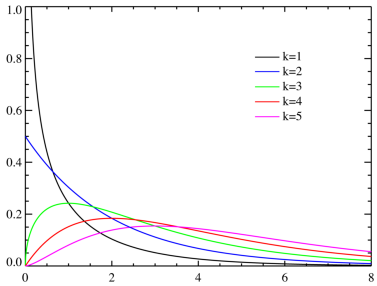
We weten nu naar wat we moeten kijken bij het analyseren van de afhankelijkheid maar er mist nog een weging, een onderlinge verhouding van de kenmerken.

De Chi-Kwadraatsom biedt hier de oplossingen en geeft ons die weging. De toetsingsgrootheid voor de Chi-kwadraattoets wordt gedefinieerd aan de hand van de volgende formule:

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



Figuur 2.1: Illustratie eenzijdige-toets van een χ^2-distributie



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

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

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

met $\frac{O_i*}{N}$ de marginale probabiliteit dat i als eerste deel van het bigram voorkomt en $\frac{O_j*}{N}$ de marginale probabilitiet 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):

$$E_11 = \frac{9 + 3632}{N} + \frac{9 + 7893}{N} * N\hat{a}\L\acute{L}0,0085$$

Als laatste onderdeel berekenen we de χ^2-score, bepalen het aantal vrijheidsgraden en zoeken de χ^2 distributie op met de berekende vrijheidsgraad. Stel dat het vooropgestelde betrouwbaardheidsinterval 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 kunnen ze beschouwd worden als afhankelijk.

Onderstaande afbeelding geeft een illustratie over hoe de verwerping of aanvaarding van een nulhypothese juist in werking gaat

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 gaat baseren.

2.2.5 Best feature selection

Als we duizende 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 Bag of Words en het verwijderen van stopwoorden. Veel voorkomende features kunnen voor het document niet identificerend dienen en zorgen voor ruis op de dataset. Daarom kan men verkiezen om deze low-information features te verwijderen zodanig dat men enkel de features overhouden die echt iets zeggen over een document. Het bepalen van de informatiewinst kan gebeuren aan de hand van het aantal voorkomens in de verschillende klasses. 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 gebruiken. Chi-kwadraat laat ons namelijk toe om de correlatie tussen een bepaalde feature en de klasses te meten.

2.2.6 Latent Semantic Analysis

Latent Semantic Analysis is een wiskundige techniek gebaseerd op statistische berekeningen. 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 2 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 LSM toegepast door **Singular Value Decomposition (SVD)** toe te passen op de terms-document 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

$$A = U\Sigma V^T = \underbrace{\begin{bmatrix}\mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_r\end{bmatrix}}_{\text{Kolommen } A} \underbrace{\begin{bmatrix}\mathbf{u}_{r+1} & \dots & \mathbf{u}_m\end{bmatrix}}_{\text{Nul } A^T} \begin{bmatrix}\sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & 0 & \dots & \sigma_k & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & 0 & \dots & 0 & 0 & \dots & 0\end{bmatrix} \begin{bmatrix}\mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \dots \\ \mathbf{v}_r^T \\ \mathbf{v}_{r+1}^T \\ \dots \\ \mathbf{v}_n^T\end{bmatrix} \begin{matrix}\} \text{Rijen } A \\ \\ \\ \\ \} \text{Nul } A\end{matrix}$$

U is de unitaire matrix waarbij men $u_1, u_2, ..., u_n$ de linker singuliere vectors noemt. Deze stellen een document met zijn features voor. V^T is de geconjugeerde getransponeerde matrix van V. $v_1, v_2, ..., v_n$ noemt men de rechter singuliere vectors en stellen de woorden met hun features over alle documenten voor. Σ is diagonaal matrix met singuliere waarden $\sigma_1, \sigma_2, .., \sigma_n$' op de diagonaal. De reductie van een term-document 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. Laat dit het 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 Maas et al. (2011),dus ook voor ons onderzoek, omdat men nieuwe meer efficiënte features krijgt. De nieuwe features geven meer duidelijkheid en inzicht en kunnen dienen als input voor een algoritme.

2.3 Classifiers

2.3.1 Naive Bayes Classifier

2.3.2 Decision Tree

2.4 Valkuilen onderzoek

2.4.1 Overfitting

2.4.2 Bais

2.4.3 Variantie

Hoofdstuk 3

Het Onderzoek

Voor deze bachelorproef onderzoek we of we een algemene technieken binnen de machine learning ons een goede analyse van gevoelens kunnen geven op Nederlandse tekst. Verder situeert men zich met het onderzoek onder supervised learning waarbij we data proberen te classificeren. Zoals eerder vermeldt duidt supervised learning op een gelabelde dataset waar zowel de inputwaarden als de outputwaarde gekend zijn. Concreter hadden we voor het onderzoek de gevoelens veralgemeent tot positief en negatief en een dataset opgebouwd uit reviews van films, muziek en boeken, afkomstig van `wwww.moviemeter.nl`, `wwww.muziekmeter.nl` en `wwww.boekmeter.nl`.

Het onderzoek is opgedeeld in twee delen. Er is Initiëel deel waarbij we de classifiers trainen met een dataset en een onderscheidt maken op basis van de pre-processing techniek. Vervolgens worden de resultaten bekeken op besis van hun precisie. In het tweede deel volgt er een verdere uitdieping van interessante resultaten uit deel één.

3.1 Initiële onderzoek

3.2 Werkwijze

Als hoofddoel willen we te weten komen of het mogelijk is om met algemene technieken uit de machine learning goede classificatieresultaten kunnen behalen. Goede classificatieresultaten uit zich in de precisie waar de classifier met classificeert. In dit onderzoek gaat dit de metriek zijn waarop we bepalen of een classificatie goed of slecht verloopt.
We gebruiken de eerder besproken classifiers, namelijk een Naive Bayes classifier en Decission tree als zelflerende algoritmen en trainen Naive bayes classifier telkens met een ander voorbewerkte dataset. De Decission tree trainen we enkel met een LSA voorbewerkte dataset. Als laatste analyseren de telkens de resultaten op basis van de testset en de voorbewerkingstechniek die gebruikt wordt op de dataset.

3.3 Resultaten

Belangrijk om te melden dat volgende resultaten afkomstig zijn door de classifiers te trainen met een moviedataset van 8000 samples, random gekozen, waarvan 6000 trainsamples en 2000 testsamples en dit gemiddeld genomen over 10 runs. Merk op dat zowel de testset als trainigsset evenwichtig verdeeld zijn. Dit wil zich dat precies de helft van de set positief is en de andere helft negatief.

Verder wordt er telkens paarsgewijs getest. De testset is altijd hetzelfde voorbewerkt als de trainingsset van de classifier.

Het eerste wat hier is weergegeven, zijn de resultaten van een Naive Bayes Classifier met als trainingsset filmrecensies en als testset film- , muziek en boekrecensies.

Pre-processing techniek	**Precisie Trainingsset**	**Precisie Testset**
Bag of words	86,66%	58,30%
Remove stopwords	94,68%	64,19%
Bigram (n=200)	99,26%	61,75%
Bigram (n=50)	94,68%	64,19%
Best Features + bigram (n = 200)	94,89%	64,70%

Tabel 3.1: Resultaten van de Naive Bayes Classifier met als trainingsset en testset filmrecensies

Pre-processing techniek	**Precisie**
Bag of words	53,81%
Remove stopwords	60,04%
Bigram (n=50)	58,27%
Bigram (n=200)	59,75%
Best Features	54,94%
Best Features + bigram (n = 200)	61,49%

Tabel 3.2: Resultaten van een Naive bayes classifier met een als trainingsset filmrecensies en als testset boekrecensies

Pre-processing techniek	**Precisie**
Bag of words	53,17%
Remove stopwords	58,18%
Bigram (n=50)	56,48%
Bigram (n=200)	57,05%
Best Features	54,24%
Best Features + bigram (n = 200)	58,72%

Tabel 3.3: Resultaten van een Naive bayes classifier met een als trainingsset filmrecensies en als testset muziekrecensies

Wat meteen is dat er niet echt een beduidend verschil is tussen de pre-procestechnieken. Ook zien we in tabel 3.BLA dat de beste resultaten worden behaald wanneer de train- en testset van dezelfde soort zijn.

Volgende classificatieresultaten komen voort uit de classificatie waarbij de train- en testresultaten telkens voorbewerkt zijn door alle stopwoorden en leestekens te verwijderen. Dan de sets om te vormen naar een tfidf-matrix en tenslotte LSA uit te voeren en ieder document te reduceren naar 2 features.

Model	**Precisie Trainingsset**	**Precisie Testset**
Decision Tree	99,77%	99,22%
Naive Bayes classifier	99,82%	51,35%

Tabel 3.4: Resultaten van de classifiers met als trainingsset en testset filmrecensies, beide voorbewerkt door LSA

Deze resultaten zijn zeer interessant. Er is duidelijk een groot verschil tussen de precisie van de testset voor de Decision Tree en de Naive Bayes classifier. De Naive Bayes classifier kan totaal niets opmaken uit de LSA features die een document voorstellen. Zeker als je weet als men random zou sorteren de precisie 50% zou bedragen. Aan de andere hand zijn de resultaten van de decision tree in combinatie met de lsa features verrassend goed. Met bij een perfecte classificatie

kan de decision tree heel goed het onderscheidt maken tussen positieve en negatieve filmrecensies. Wat een veel betere prestatie is, dan de Naive bayes classifier waarbij men hoogste 64,70% haalde.

Dit resultaat kan men niet negeren er vergt een dieper onderzoek. Wat we voorlopig weten op basis van de resultaten is:

- De Naive Bayes Classifier heeft in het algmeen een slechte classificatieprestatie, zowel wanneer train- en testset van dezelfde soort of verschillend zijn.
- De classificatie verbetert niet beduidend wanneer men een andere pre-processing techniek gebruikt.
- Classificatie aan de hand van LSA features in combinatie met een decision tree classifier levert goede resultaten op wanneer train- en testset van dezelfde soort zijn.

Nu we dit weten, bekijken we de combinatie van LSA en de decision tree classifier wat meer in detail. In de uitdieping gaan we bekijken hoe de classificatieresultaten zich verhouden wanneer men een zelfde of verschillende train- en testset gebruikt. Verder gaan we deze resultaten met die van de Naive Bayes Classifier in dezelfde omstandigheden. Als laatste proberen we classificatieresultaten van de decision tree grafisch in kaart te brengen en te kijken waarom het niet of wel goed presteert.

3.4 Uitdieping onderzoek

Onderstaande tabel geeft alle classicatieresultaten weer, waarbij de kolommen het type van de trainingsset voorstelt en de rijen de testsets.

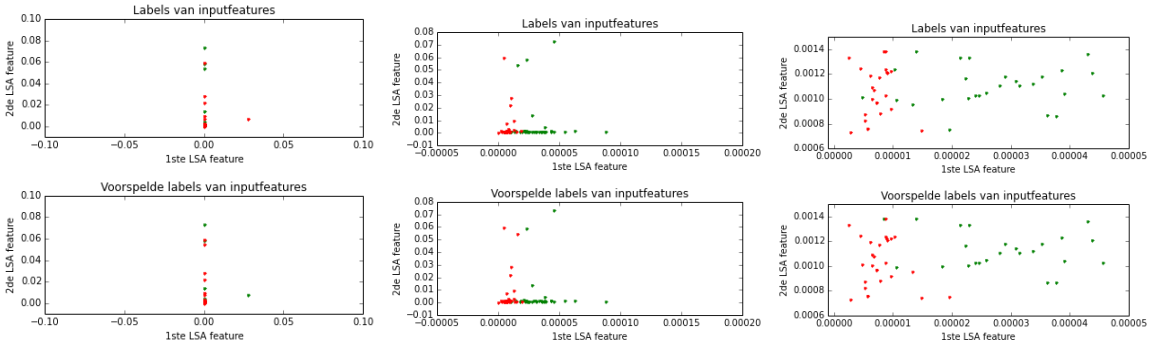
Testset	Movies	Muziek	Boeken
Movies	98,90%	44,01%	47,56%
Muziek	57,60%	98,67%	45,93%
Boeken	42,83%	51,89%	94,32%

Tabel 3.5: Resultaten van de classificatie met een Decision Tree, getrained op LSA features met verschillende soorten trainings- en testsets

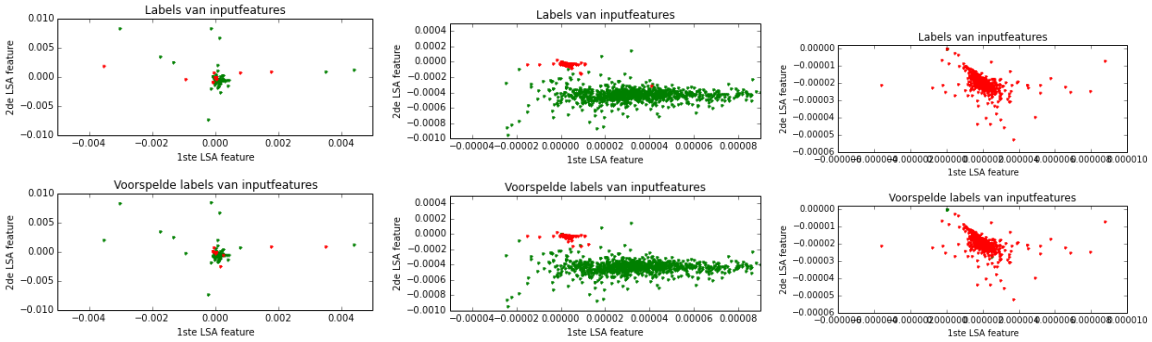
Testset	Movies	Muziek	Boeken
Movies	0.6215		
Muziek	53,165%		
Boeken	53,81%		0.693243243243

Tabel 3.6: Resultaten van de classificatie van een Naive Bayes Classifier, getrained met verschillende soorten trainings- en testsets volgens Bag 0f Words

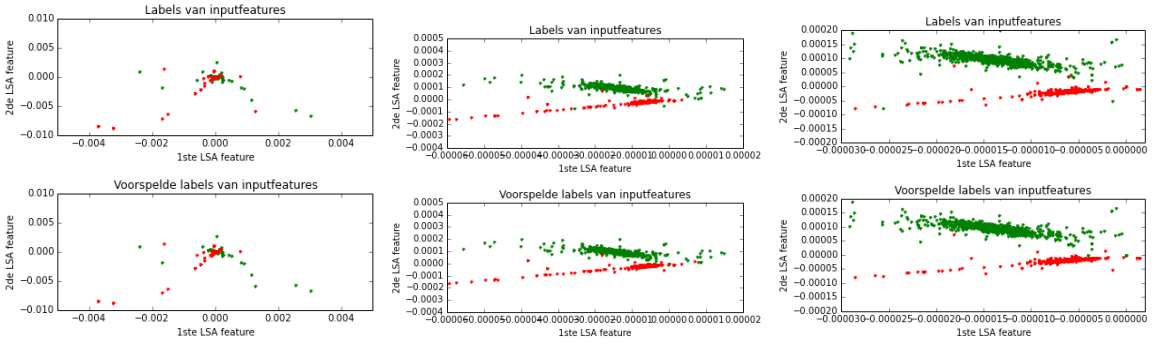
Als we naar de resultaten kijken zien we dat van alle combinaties de classificatieresultaten waarbij de trainingsset en testset hetzelfde zijn, we de beste resultaten verkrijgen. Verder valt het opnieuw op dat LSA in combinatie met een decision tree uitermate goed presteert, zowel voor de film- als de boek- en muziekrecensies. De classificaties van testsets, waarbij de trainingsset een ander onderwerp had, presteren voor beide classifiers slecht.



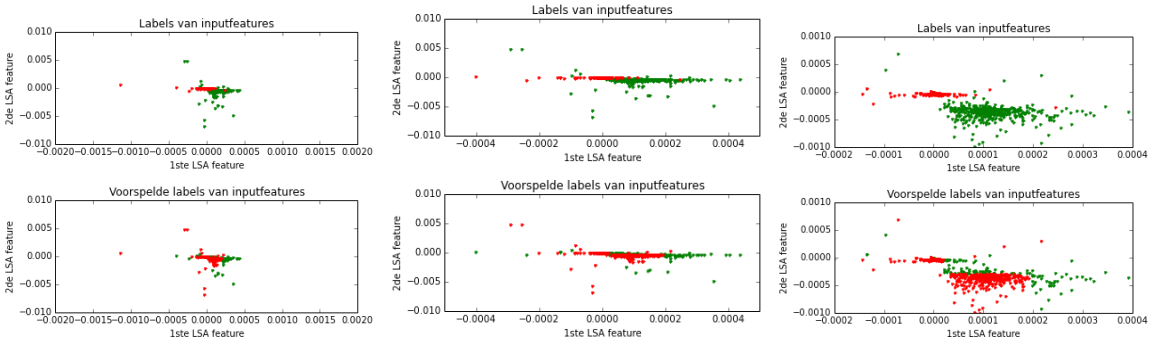
Figuur 3.1: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als train- en testset boekrecensies



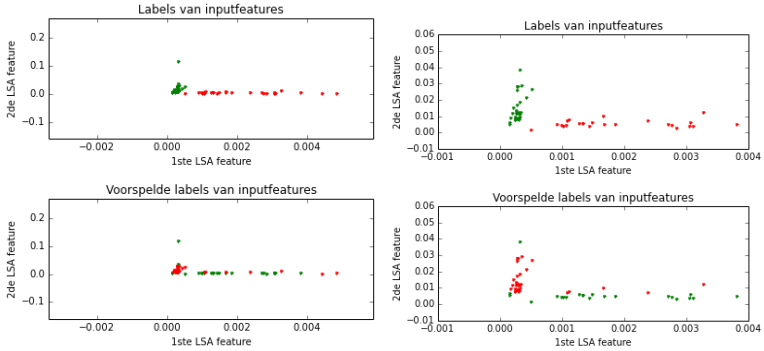
Figuur 3.2: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als train- en testset filmrecensies



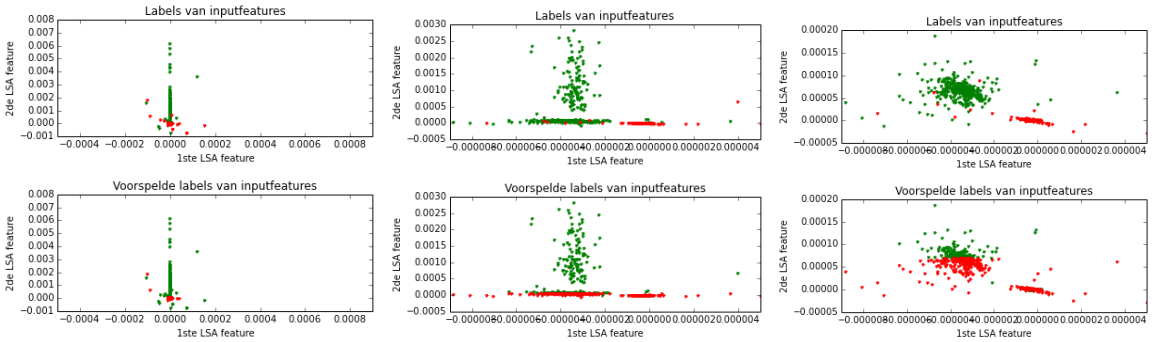
Figuur 3.3: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als train- en testset muziekrecensies



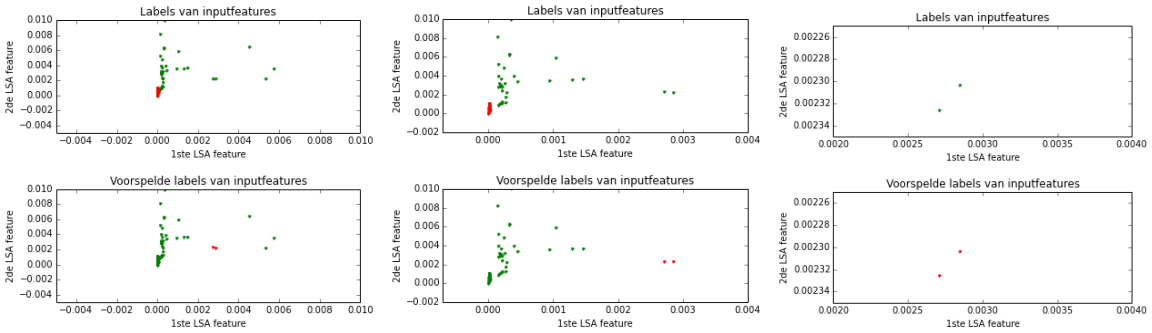
Figuur 3.4: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset muziekrecensies en testset filmrecensies



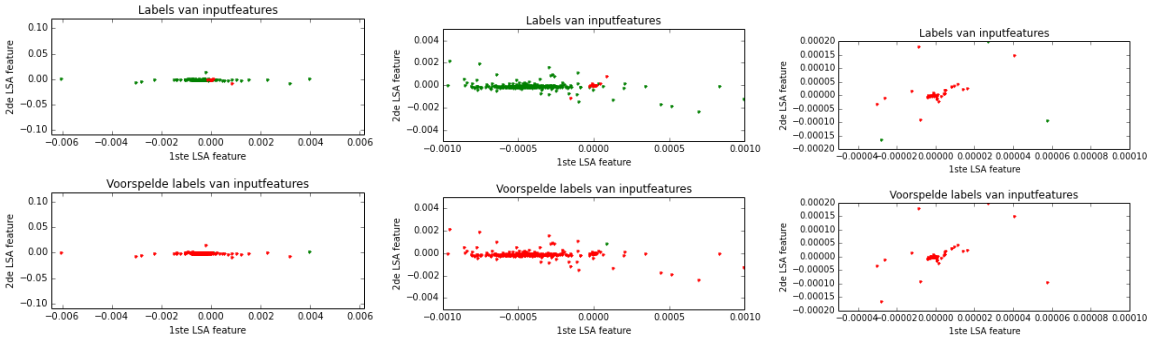
Figuur 3.5: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset muziekrecensies en testset boekrecensies



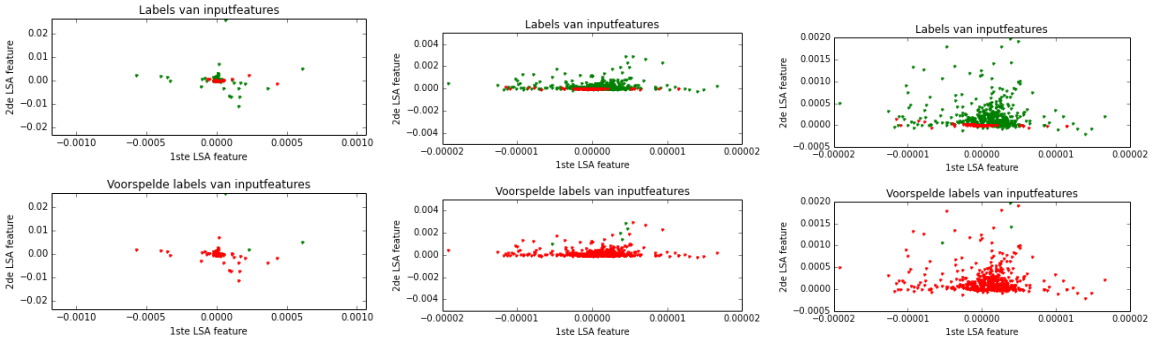
Figuur 3.6: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset filmrecensies en testset muziekrecensies



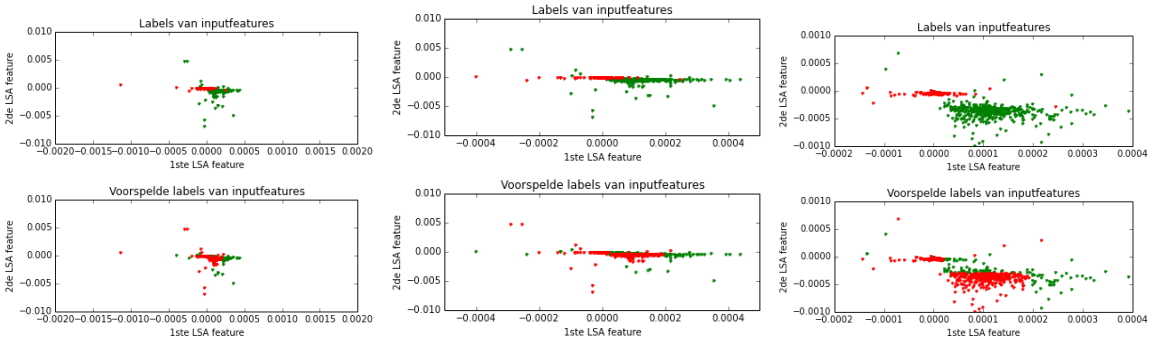
Figuur 3.7: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset filmrecensies en testset boekrecensies



Figuur 3.8: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset boekrecensies en testset filmrecensies



Figuur 3.9: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset boekrecensies en testset muziekrecensies



Figuur 3.10: Plot van resultaten van de classificatie met een decision Tree, getrained op LSA features met als trainset muziekrecensies en testset filmrecensies

Hoofdstuk 4

Conclusie

In deze voorbereiding hebben we omschreven wat Machine Learning juist omvat. Namelijk het onderzoeken en ontwikkeling van zelflerende algoritmes, die hoofdzakelijk uit drie stappen bestaan, namelijk data verzamelen, verwerken en analyseren. Naargelang het soort data en wat deze weergeeft bestaan er binnen het domein van Machine Learning verschillende technieken met hun specifieke eigenschappen en voorbeelden. Voor situaties waarin we op voorhand over voldoende data beschikken en we duidelijk weten wat deze data betekend, bespraken we in sectie 2.2 verschillende supervised learning technieken die in staat zijn om een hypothese te formuleren op basis van de gegeven data. We maakte een duidelijk onderscheid welke problemen er zich kunnen voordoen zoals een classificatie probleem versus een regressie probleem en hoe men deze moet oplossen.
We hebben gezien dat een classificatie probleem zich onderscheidt van een regressie probleem door dat de output van de hypothese zich beperkt tot een kleine set van mogelijkheden, wat bij een regressie probleem een hele reeks van mogelijkheden is. Vervolgens hebben we een specifieke techniek besproken, namelijk de vector space methode. Een methode die van toepassing is bij text mining. Deze methode kan men op verschillende manieren verfijnen. Zo raadde we aan in sectie 3.1 om document pre-processing toe te passen op de dataset voor de verwerking van de data. Verder werd er ook aangeraden in sectie 3.2.1 om de standaard vector space methode te optimaliseren met technieken zoals Latent Semantic Analysis (LSA) en Term weighting. Ten slotte hebben we een proefopstelling opgesteld waarbij de techniek LSA wordt toepast en de efficiëntie en werking van Latent Semantic Analysis nogmaals wordt bevestigd.

Literatuur

Decomposing signals in components (matrix factorization problems). (z. j.). http://scikit-learn.org/stable/modules/decomposition.html. (Accessed: 2014-30-11)

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

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)

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011). 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).

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)

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

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)

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

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.

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