KU LEUVEN

FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN

TECHNOLOGIECAMPUS GENT

Sentiment Analysis van commentaren van gebruikers op routes

Joran DE WILDE

Promotor(en): Prof. dr. ir. G. Vanden Berghe

Co-promotor(en): M. Van Lancker
P. Brackman (Bedrijf)

Masterproef ingediend tot het behalen van de graad van master of Science in de industriële wetenschappen: Industrieel Ingenieur ICT

Academiejaar 2018 - 2019

Voorwoord

In dit voorwoord zou ik graag enkele mensen bedanken die mij gesteund en begeleid hebben tijdens mijn studie en het voltooien van dit eindwerk. Ten eerste mijn promotor Greet Vanden Berghe en co-promotor Michiel Van Lancker, voor de goede begeleiding die ervoor zorgde dat de thesis in goede banen geleid werd. Verder zou ik ook graag het personeel van RouteYou bedanken, in het bijzonder Pascal Brackman, Kevin Baker en Pieter Vandermeulen, voor de opvolging en bijstand tijdens het realiseren van dit eindwerk. Ook zou ik ook mijn appreciatie willen tonen voor mijn medestudenten waarmee ik vele uren samen heb zitten werken in een aangename werksfeer. Tot slot zou ik ook nog mijn ouders, vrienden en familie willen bedanken voor de voortdurende steun tijdens mijn studies en eindwerk.

Joran De Wilde
Gent
14 mei 2019

Samenvatting

Vele online diensten geven vandaag de dag de mogelijkheid aan gebruikers om commentaar te plaatsen. Gebruikers kunnen zo medegebruikers inlichten over de kwaliteit van een product alsook feedback geven aan het platform. Deze kunnen bedrijven vervolgens gebruiken om hun product te verbeteren. In deze thesis worden er commentaren gegeven op fiets-, wandel- en motorroutes. Het beoordelen van deze commentaren gebeurt vandaag de dag nog vaak manueel, en neemt wekelijks enkele uren tijd in beslag. Deze beoordelingen zijn belangrijk om de kwaliteit van de route te kunnen inschatten, en hiermee de betere routes te laten opvallen. De dataset bestaande uit al deze commentaren is echter beperkt in grootte. In deze studie zal gewerkt worden met twee machineleertechnieken die goed presteren wanneer er slechts een kleine dataset beschikbaar is. Enerzijds wordt er gewerkt via een Rule Based aanpak waarbij een methodiek wordt voorgesteld waarbij via training en test data systematisch nieuwe regels worden toegevoegd. Anderzijds wordt ook een Multinomial Naive Bayes model getraind op de beschikbare data. Vervolgens worden deze twee verschillende technieken vergeleken met elkaar om na te gaan welke van de twee het best presteert voor dit concrete probleem. Hierbij wordt gevonden dat het Rule Based Machine Learning model met een accuraatheid van 75% beter presteert dan het Multinomial Naive Bayes model, dat een accuraatheid haalt van 72%. Verder wordt ook onderzocht wat de invloed op de werking van de modellen is wanneer er gewerkt wordt met naar Nederlands vertaalde commentaren, waarbij tot de conclusie gekomen wordt dat de accuraatheid van vertaalde commentaren 8% lager lag dan deze van origineel-Nederlandstalige commentaren.

Inhoudsopgave

Voorwoord iii

Samenvatting iv

Inhoud vii

Figurenlijst ix

Tabellenlijst x

1 Inleiding 1

2 Literatuurstudie 3

2.1 Sentiment Analysis . . . 3

2.2 Technieken om aan Sentiment Analysis te doen . . . 4

2.2.1 Representatie . . . 4

2.2.2 Rule Based Learning . . . 6

2.2.3 Naïve Bayes . . . 7

2.2.4 Support Vector Machines . . . 9

2.2.5 Neurale Netwerken . . . 11

2.2.6 Besluit . . . 13

2.3 Bestaande software . . . 13

2.3.1 Natural Language Toolkit (NLTK) . . . 13

2.3.2 Textblob library . . . 14

2.3.3 RapidMiner . . . 14

2.3.4 Repustate . . . 15

2.4 Besluit . . . 16

3 Data exploratie 17

3.1 Verwerving . . . 17

3.2 Algemene statistieken . . . 18
3.3 Soorten commentaren . . . 19
3.4 Besluit . . . 20

4 Preprocessing and Noise Cancelling 21
4.1 Filtering . . . 21
4.2 Translating . . . 22
4.3 Tokenization & stemming . . . 22
4.4 Besluit . . . 23

5 Ontwerp van de modellen 24
5.1 Het Rule Based Learning model . . . 24
5.1.1 Het lexicon . . . 25
5.1.2 De regels . . . 27
5.1.3 De evaluatie . . . 29
5.1.4 Gebruik van feedback . . . 30
5.1.5 Besluit . . . 31
5.2 Het Multinomial Naive Bayes model . . . 31
5.2.1 Vectorisatie . . . 31
5.2.2 Cross Validation . . . 31
5.2.3 Labels . . . 32
5.2.4 Realisatie . . . 32
5.2.5 Besluit . . . 32

6 Resultaten 33
6.1 Het Rule Based Learning model . . . 33
6.1.1 Evolutie van de accuraatheid . . . 33
6.1.2 De Evaluatiefunctie . . . 34
6.2 Multinomial Naive Bayes . . . 35
6.2.1 De vectorisatiemethodes . . . 35
6.2.2 Evolutie van de accuraatheid . . . 36
6.3 Variantieanalyse . . . 39
6.4 Anderstalige commentaren . . . 39
6.5 Rule Based Learning vs Multinomial Naive Bayes . . . 41
6.5.1 Normale verdeling van de modellen . . . 42

6.5.2 Vergelijking van het Rule Based Learning model ten opzichte van het Multinomial Naive Bayes model. 43
6.6 Besluit 44

7 Conclusie **46**
7.1 Toekomstig werk 47

A Adjectieven en Bijwoorden aanwezig in het Rule Based model **49**
A.1 Adjectieven 51
A.2 Bijwoorden 53

B Top 200 meest voorkomende woorden **54**
B.1 Niet gestemde woorden 55
B.2 Gestemde woorden 59

C Extended abstract **62**

D Poster **69**

Lijst van figuren

2.1 Voorbeeld van een SVM Afbeelding te vinden op: https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python 10

2.2 Voorbeeld van een Neuraal Netwerk Afbeelding te vinden op: https://medium.com/@curiousily/tensorflow-for-hackers-part-iv-neural-network-from-scratch-1a4f504dfa8 11

2.3 Voorbeeld van een Neuron Afbeelding te vinden op: http://cs231n.github.io/neural-networks-1/ 12

2.4 Voorbeeld van een Diep Neuraal Netwerk Afbeelding te vinden op: https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351 12

2.5 Voorbeeld van de constructie van een Decision Tree via RapidMiner 16

3.1 Aantal commentaren geplaatst per gebruiker. 18

3.2 Aantal commentaren met een bepaalde lengte. 19

3.3 Percentage commentaren per taal in de dataset 20

4.1 Accuraatheden voor verschillende (combinaties van) stemming algoritmen. 23

5.1 Schematische voorstelling van de workflow van beide modellen 24

5.2 Voorbeelden van de drie aanwezige dictionaries. 27

5.3 Werking van het Rule Based model voor een voorbeeldzin 28

5.4 Visualisatie van de neutrale zone 30

5.5 Werking 10 fold cross validation. 32

6.1 Evolutie van de accuraatheid van het Rule Based model in functie van het aantal iteraties. 33

6.2 Resultaten voor verschillende grenswaarden van de evaluatiefunctie. 35

6.3 Resultaten voor accuraatheden van de verschillende vectorisatiemethodes. 36

6.4 Accuraatheden ifv aantal commentaren voor elke vectorisatiemethode. 37

6.5 The average and standard deviation of critical parameters 38

6.6 Vergelijking tussen resultaten van Nederlandstalige en niet-Nederlandstalige commentaren. 41
6.7 Normaal-waarschijnlijkheidsplot voor het Rule Based model 43
6.8 Normaal-waarschijnlijkheidsplot voor het Naive Bayes model 43
6.9 Variantieanalyse tussen het Rule Based Model en het Multinomial Naive Bayes model. 44

Lijst van tabellen

6.1 Verschillende waarden voor de verschillende iteraties van de 10 fold cross validation uitgevoerd door beide modellen en voor beide datasets. 40
6.2 Resultaten voor de 10 fold cross validation voor de routedataset voor beide modellen. 42

Hoofdstuk 1

Inleiding

Commentaren van gebruikers zijn enorm handig voor een bedrijf om te weten of nieuwe producten aanslaan bij het doelpubliek, om na te gaan waar verbeteringen mogelijk zijn, om bepaalde producten in de verf te zetten... In deze thesis wordt er commentaar gegeven op fiets-, wandel- en motorroutes. Deze commentaren kunnen vervolgens gebruikt worden om onderscheid te kunnen maken tussen goede en slechte routes. Bij het zoeken naar routes zullen vervolgens ook eerder goede routes getoond worden in plaats van de slechtere routes.

Momenteel worden de commentaren die op de routes verschijnen nog manueel door één iemand beoordeeld. Dit neemt elke week een ochtend in beslag, waardoor er in een jaar 208 werkuren verloren gaan. Het zou dus interessanter zijn indien deze commentaren automatisch beoordeeld kunnen worden. Hiervoor wordt er op zoek gegaan naar een gratis oplossing voor dit probleem. Verder moeten er niet enkel Nederlandstalige commentaren beoordeeld kunnen worden, maar ook commentaren in andere talen zoals het Frans, Engels of Duits. De dataset met commentaren die momenteel beschikbaar is, is beperkt in grootte (5380 commentaren) en kan dus ook voor problemen zorgen bij het gebruiken van machine learning technieken. Tot slot zou het zeer voordelig zijn moest de Sentiment Analysis software direct kunnen communiceren met de reeds bestaande software van het bedrijf. Hierdoor zouden commentaren makkelijk automatisch beoordeeld kunnen worden.

Deze masterproef bestudeert op welke manier commentaren automatisch geanalyseerd kunnen worden en hoe hieruit gehaald kan worden of deze al dan niet positief, neutraal of negatief zijn. Er wordt dus een model ontwikkeld dat in staat is om deze taak te volbrengen. Om dit model te ontwerpen dient ook de juiste techniek gekozen te worden. Hiervoor dient men ook rekening te houden met de beschikbare hoeveelheid data. In totaal zijn er ongeveer 5830 commentaren over de verschillende routes beschikbaar. Er wordt dan ook gekozen voor een techniek die goed kan presteren bij geringe datasets. Twee technieken komen naar voren die hieraan voldoen: een Rule Based aanpak en een Multinomial Naive Bayes Classifier. Waarom deze twee technieken goed presteren bij kleine datasets wordt verder uitgelegd in de literatuurstudie in Hoofdstuk 2. Verder in de masterproef wordt vervolgens een vergelijkende studie uitgevoerd tussen deze twee technieken,

waarin bestudeerd wordt welke van de twee het best geschikt is voor de huidige situatie. Bijkomend wordt er ook gekeken hoe anderstalige commentaren in dit model geïntegreerd kunnen worden. Is het vertalen naar het Nederlands voldoende, of dienen er extra stappen genomen te worden, zoals een apart model per taal?

Het proces waarbij uit stukken tekst een bepaald gevoel gehaald wordt, wordt ook wel sentiment analysis genoemd. Een inleiding tot Sentiment Analysis is te vinden in Hoofdstuk 2. In dit hoofdstuk worden ook de verschillende technieken om aan Sentiment Analysis te doen uitvoerig beschreven, alsook de reeds bestaande Sentiment Analysis software.
In Hoofdstuk 3 wordt een exploratie en analyse van de gebruikte dataset uiteengezet. Concreet worden hier de verwerving van de data, algemene statistieken en de verschillende soorten commentaren toegelicht.
In Hoofdstuk 4 worden de verschillende preprocessing stappen besproken. Hierin komen de filtering, vertaling, stemming en tokenization van commentaren aan bod. Op elk van deze stappen zal vervolgens ingegaan worden.

Hoofdstuk 5 legt hierna de twee verschillende modellen waarmee gewerkt werd in deze studie uit. Hierin wordt ingegaan op de verschillende componenten van deze modellen, alsook op de realisatie en samenwerking van deze componenten.
De resultaten van beide modellen en de vergelijking tussen deze modellen komt aan bod in Hoofdstuk 6. Hierbij wordt een vergelijking gemaakt via variantieanalyse. Verder wordt er ook getoond hoe de anderstalige commentaren scoren ten opzichte van de Nederlandstalige commentaren.
In Hoofdstuk 7 wordt een besluit van de thesis geformuleerd en worden mogelijkheden voor verder onderzoek besproken.

Hoofdstuk 2

Literatuurstudie

In dit hoofdstuk wordt eerst een korte inleiding gegeven tot Sentiment Analysis, waarvoor het gebruikt kan worden en waarom het interessant is voor bedrijven.
Vervolgens worden de verschillende technieken onderzocht die reeds aangetoond hebben goede resultaten te leveren wanneer er Sentiment Analysis software dient ontwikkeld te worden.
Tot slot wordt een overzicht van bestaande third-party software gegeven, alsook de redenen waarom ze in dit scenario niet gunstig zijn.

2.1 Sentiment Analysis

"Sentiment analysis and opinion mining is the field of study that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions from written language." - (7)

Sentiment Analysis (SA) is het proces waarbij onderzocht wordt hoe sentimenten en emoties automatisch uit stukken tekst kunnen worden gehaald. Het analyseert dus het gevoel van gebruikers via de tekst die deze hebben geschreven. Concreet houdt dit in dat er uit tekst wordt gehaald of de gebruiker een positieve, neutrale of negatieve commentaar levert. SA bestaat uit verschillende stappen. Zo dient een tekst eerst omgezet te worden in een set van features. Methoden hiervoor zijn beschreven in Sectie 2.2.1. Deze features worden gebruikt worden om via Natural Language Processing, Machine Learning, of een combinatie van beide tot een bepaalde klasse te komen, die aangeeft of de oorspronkelijke tekst positief, neutraal of negatief is.
Natural Language Processing (NLP) is een deelveld van de artificiële intelligentie (AI) die zich toespitst op de interactie tussen computers en de menselijke taal. Hierbij ligt de focus op het creëren van programma's die de menselijke (natuurlijke) taal kunnen verwerken. Voorbeelden van NLP taken zijn het stemmen van woorden, het opdelen van een tekst in al zijn woorden en part-of-speech tagging.
Machine Learning, vaak gebruikt om aan SA te doen, is ook een deelveld van de AI waarbij onderzocht wordt hoe computersystemen zelf kunnen leren, en taken kunnen uitvoeren zonder hiervoor

expliciete instructies gekregen te hebben. Via machine learning kunnen computersystemen leren van data, zichzelf verbeteren en veranderen zonder tussenkomst van een mens. In Sectie 2.2 worden de voornaamste machine learning technieken om aan SA te doen besproken.

SA wordt tegenwoordig enorm veel gebruikt en is van cruciaal belang voor bedrijven. Via SA kunnen bedrijven namelijk een zeer goed beeld krijgen van de tevredenheid van hun klanten. Zo kunnen bedrijven te weten komen wat het grote publiek vindt van een nieuw product of een wijziging aan een bepaald product. Door vervolgens de negatieve reviews in acht te nemen kan het bedrijf vervolgens inspelen op de klachten van de klanten om betere producten aan te bieden.
Een bedrijf kan SA ook gebruiken om automatisch een kwaliteitslabel aan hun producten te geven, door het aantal negatieve en positieve beoordelingen te gebruiken.

SA is dus een enorm handige tool voor elk bedrijf om een betere klantenservice aan te bieden en een beter beeld te krijgen van hoe het grote publiek over hun producten denkt.

2.2 Technieken om aan Sentiment Analysis te doen

In deze sectie worden de verschillende technieken besproken om aan SA te doen. Deze hebben reeds in vorige onderzoeken aangetoond dat ze zeer geschikt zijn om SA modellen mee te creëren. Eerst volgt een kort overzicht van de manieren waarop tekst gerepresenteerd kan worden. Om alle verschillende ideeën te illustreren zal er met volgende running example gewerkt worden, bestaande uit drie zinnen:

'De fietsroute mooi. Ook het landschap was mooi'
'Leuke paden, met een mooi uitzicht op het landschap!'
'De regen maakte de route wat glad.'

2.2.1 Representatie

Om tekst voor te stellen bestaan reeds verschillende representaties. In wat volgt worden de voornaamste representaties kort besproken.

2.2.1.1 N-gram Model

Dit model groepeert verschillende opeenvolgende woorden of karakters waaruit een zin bestaat in groepen bestaande uit respectievelijk n woorden of karakters. Deze n kan gaan van 1, voor groepen van individuele woorden, tot oneindig. Als voorbeeld werden de drie voorbeeldzinnen omgezet tot een set van features via bi-grams, waarbij steeds groepen van twee woorden werden gevormd.

['De fietsroute','fietsroute was','was aangenaam','aangenaam en','en mooi','mooi Ook','Ook het','het

landschap','landschap was','was mooi']
['Heel leuke','leuke paden','paden met','met een','een mooi','mooi uitzicht','uitzicht op','op het','het landschap']
['De regen','regen maakte','maakte de','de route','route wat','wat glad']

Dit staat bekend als een word level n-gram model. Het is ook mogelijk om een character level n-gram model te realiseren, door telkens n opeenvolgende karakters samen te nemen. Als voorbeeld wordt hier gewerkt met de zin *'Dit is mooi'*. Dit wordt vervolgens omgevormd via een character-level n-gram model naar:

[' i', ' m', 'di', 'is', 'it', 'mo', 'oi', 'oo', 's ', 't ']

Hierbij valt op dat de spaties gezien worden als een karakter en dus mee verwerkt worden. Dit kan eventueel vermeden worden door spaties eruit te filteren. Deze woorden kunnen direct gebruikt worden als features, maar ook als termen voor de vocabulary in de twee volgende vectorvoorstellingen.

2.2.1.2 Count Vectors

Count vectors zijn vectoren waarbij elke waarde in deze vector staat voor het aantal keer dat een term uit het vocabulary voorkomt in deze tekst. De termen uit het vocabulary worden gegenereerd via een n-gram model. Er kan dus gekozen worden voor een vocabulary bestaande uit individuele woorden, uit groepen van n opeenvolgende woorden, of uit groepen bestaande uit minstens n_1 en maximum n_2 woorden.

Count vectors houde enkel bij hoeveel keer een woord voorkomt in een tekst. Indien we de drie voorbeeldzinnen omzetten tot count vectors, werkende met een unigram model, krijgen we als vocabulary:
['de', 'fietsroute', 'mooi', 'ook', 'het', 'landschap', 'was', 'leuke', 'paden,', 'met', 'een', 'uitzicht', 'op', 'regen', 'maakte', 'route', 'wat', 'glad']

En als count vectors:
['1', '1', '2', '1', '1', '1', '1', '0', '0,', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0']
['0', '0', '1', '0', '1', '1', '0', '1', '1,', '1', '1', '1', '1', '0', '0', '0', '0', '0']
['1', '0', '0', '0', '0', '0', '0', '0', '0,', '0', '0', '0', '0', '1', '1', '1', '1', '1']

2.2.1.3 TF-IDF Vectors

TF-IDF staat voor Term Frequency-Inverse Document Frequency. Hierbij wordt de vectorwaarde van een woord w in een tekst t berekend door eerst de frequentie na te gaan waarin het voorkomt

in de tekst zelf. Dit is hetzelfde als hetgeen count vectors doen. Maar in TF-IDF wordt er vervolgens ook rekening gehouden met het aantal teksten van de totale corpus waar dit woord voorkomt, door de frequentie waarmee het woord voorkomt in een enkele tekst te vermenigvuldigen met de waarde bekomen via onderstaande formule:

$$\log \frac{1+|D|}{1+Freq(w,t)} \tag{2.1}$$

waarin D de volledige corpus bestaande uit alle teksten voorstelt. Wanneer we deze TF-IDF vectorisatie toepassen op de drie voorbeeldzinnen bekomen we als vectoren:

[0.3698,0.3698,0.3698,0.3698,0.3698,0.5624]
[0.3596,0.3596,0.3596,0.2735,0.3596,0.3596,0.2735,0.2735,0.3596]
[0.3767,0.3767,0.5730,0.2865,0.2865,0.3767,0.2865]

Hierin moet opgemerkt worden dat woorden waarvoor de frequentie nul was niet werden opgenomen in deze vectoren. We zien dat de waarden voor woorden die meerdere keren voorkomen in de tekst beduidend hoger liggen dan woorden die slechts één keer voorkwamen in de tekst. Hier werd als vocabulary gebruik gemaakt van individuele woorden, maar dit kan ook variëren. Zo kan er ook gewerkt worden met een vocabulary bestaande uit groepen van woorden, of een vocabulary bestaande uit groepen van karakters. In wat volgt zullen de verschillende technieken besproken worden waarmee aan SA gedaan kan worden. Deze technieken gebruiken de representaties hier beschreven om teksten om te zetten tot features.

2.2.2 Rule Based Learning

Rule Based Learning is een machineleertechniek waarbij modellen beslissingen nemen op basis van verschillende regels. Deze regels kunnen manueel aangemaakt worden, of aangeleerd worden met behulp van de dataset. Zo werd reeds gebruikt gemaakt van regression decision trees om verschillende regels via deze bomen te kunnen afleiden (10). Hierbij werden decision trees gebruikt om verschillende regels te kunnen afleiden. De decision trees werden opgesteld aan de hand van voorbeelddata, waarbij steeds positieve en negatieve voorbeelden gegeven werden waarmee de bomen werden opgesteld.

Rule Based Machine Learning is een veelgebruikte techniek voor SA die bestaat uit verschillende componenten. Ten eerste moet er in SA een lexicon aanwezig zijn. Dit lexicon bevat alle woorden die het algoritme moet herkennen, gevolgd door een waarde die het sentiment van dit woord weergeeft. Het kan dus gezien worden als een verzameling van (word,value)-pairs. Er zijn reeds verschillende studies waarbij het gebruik van dit soort lexicons goede resultaten heeft opgeleverd (3) en (8). Dit lexicon kan verder niet enkel woorden bevatten, maar kan ook emoticons bevatten, wat tot grote verbeteringen heeft geleid bij het beoordelen van commentaren op online discussiefora (8). Een lexicon voor de drie voorbeeldzinnen is bijvoorbeeld:

['mooi', 'leuke','glad']

Vervolgens moet er ook een scoring algoritme aanwezig zijn, dat dit lexicon kan gebruiken om er vervolgens tekst zo juist mogelijk mee te beoordelen. Dit scoring algoritme bevat dan ook de eigenlijke regels die gebruikt worden om het sentiment van een tekst te achterhalen. Voor dit scoring algoritme zijn er tal van verschillende mogelijkheden. Er werden al resultaten behaald in het beoordelen van twitterberichten van boven de 80% door het optellen van individuele woorden, gebruik makend van een zeer breed lexicon (3). Toegepast op het running example zou dit neerkomen op volgende score voor de drie zinnen:

Zin1: score = 2 (bevat 2 keer het woord 'mooi').
Zin2: score = 2 (bevat het woord 'mooi' en het woord 'leuke').
Zin3: score = -1 (bevat het woord 'glad').

Een andere mogelijkheid is het gebruik maken van modifiers. Dit zijn versterkende of verzwakkende woorden die de gewichten van bepaalde woorden vergroten of verkleinen. Voorbeelden van modifiers zijn woorden als *'Heel'* of *'Enorm'*. Modifiers worden onder andere gebruikt in Adverb-Adjective Combination (AAC). Er werden goede resultaten behaald via AAC voor het beoordelen van twitterberichten (9). Een andere studie die het succes van het gebruik van zogenaamde modifiers aantoonde kwam tot resultaten met accuraatheden boven de 80% (8).

Een groot voordeel van een Rule Based algoritme is dat de accuraatheid niet per se afhangt van de grootte van de dataset, omdat het in het geval van SA niet gaat om een zelf lerend algoritme. Deze techniek presteert dus zeer goed voor kleine datasets, en zal evengoed blijven presteren naarmate de data in grootte groeit. Wanneer de dataset echter zeer groot wordt zijn andere zelflerende technieken interessanter aangezien deze betere resultaten zullen leveren.

SA via Rule Based Learning heeft reeds goede resultaten opgeleverd in verschillende vorige studies. Verder is deze techniek zeer betrouwbaar wanneer er aan SA gedaan moet worden waarbij slechts een kleine dataset beschikbaar is. Hierdoor is het werken via Rule Based Learning dus een aan te raden techniek voor SA.

2.2.3 Naïve Bayes

De Naïve Bayes methodes zijn een verzameling van supervised learning algoritmen gebaseerd op het theorema van Bayes'. Ze worden ook naïef genoemd omdat er vanuit gegaan wordt dat er een conditionele onafhankelijkheid bestaat tussen elk paar features voor een gegeven label[1]. Dit is uiteraard niet helemaal het geval, aangezien in een tekst sommige woorden in verband staan met

[1] Uitleg Naive Bayes gebaseerd op: `https://scikit-learn.org/stable/modules/naive_bayes.html`

elkaar, en dus vaker samen voorkomen. Het theorema van Bayes ziet er als volgt uit:

$$P(Klasse|Features) = \frac{P(Features|Klasse)P(Klasse)}{P(Features)} \tag{2.2}$$

Via deze formule kan dus de kans op een klasse A berekend worden, gegeven dat kenmerk B aanwezig is, gebruik makend van drie probabiliteiten: de kans dat kenmerk B aanwezig is indien de klasse gelijk is aan A, de kans dat kenmerk B aanwezig is ongeacht de waarde van de klasse en de kans dat de klasse gelijk is aan A.

Deze formule kan dus gebruikt worden indien er slechts 1 feature (B in dit geval) aanwezig is. De formule kan vervolgens ook uitgebreid worden voor n features, en ziet er als volgt uit:

$$P(y|x_1,...x_n) = \frac{P(x_1,...x_n|y)P(y)}{P(x_1,...x_n)} \tag{2.3}$$

Aangezien we veronderstellen dat alle features onafhankelijk van mekaar zijn, kan deze formule vervolgens versimpeld worden tot:

$$P(y|x_1,...x_n) = \frac{\prod_{i=1}^{n} P(x_i|y)P(y)}{P(x_1,...x_n)} \tag{2.4}$$

Via deze formule is het dus mogelijk om de kans dat een tekst tot klasse y behoort, gegeven dat deze tekst bepaalde woorden bevat, te kunnen berekenen. Het is dan ook op deze manier dat Naive Bayes een label geeft aan een bepaald stuk tekst. Dit door het label te kiezen waarvan de probabiliteit het grootst is. Om deze kans te berekenen moet eerst $P(x_i|y)$ gekend zijn. Hiervoor wordt er vanuit gegaan van een bepaalde verdeling voor $P(x_i|y)$. Hierin zit dan ook het grote verschil tussen de Naive Bayes methoden. In wat volgt worden de drie voornaamste methoden kort besproken.

2.2.3.1 Gaussian Naive Bayes

Guassian Naive Bayes is een variatie van Naive Bayes die aanneemt dat de waarde van een term x_i in een tekst Gaussiaans verloopt. Deze techniek is dus vooral geschikt wanneer de verschillende features x_i continue variabelen zijn, wat niet het geval is voor tekstclassificatie. Hierdoor zal Gaussian Naive Bayes zelden gebruikt worden wanneer er aan tekstclassificatie gedaan dient te worden.

2.2.3.2 Multinomial Naive Bayes

Multinomial Naive Bayes is een variatie van Naive Bayes die zeer goed presteert in het analyseren van tekst, omdat deze variatie rekening houdt met de frequentie waarmee woorden voorkomen. De probabiliteit $P(x_i|y)$ wordt als volgt berekent:

$$P(x_i|y) = \frac{T_{x_i y}}{\sum_{x'_i \in V} T'_{x'_i y}}$$

Hierin is V de vocabulary, de verzameling van alle woorden en features waarmee rekening gehouden zal worden. $T_{x_i y}$ is vervolgens het aantal keer dat de term x_i voorkomt in alle zinnen met y als waarde van klasse Y. $\sum_{x'_i \in V} T'_{x'_i y}$ is vervolgens het totale aantal features te vinden in alle samples met y als waarde van klasse Y. Zo kunnen we in het running example $P('mooi'|positief')$ berekenen. De drie zinnen zullen dus als trainingsdata gebruikt worden. De eerste twee zinnen zijn positief, en de laatste negatief. $P('mooi'|positief')$ wordt dan berekend als volgt:

$$P('mooi'|positief') = \frac{3}{4} = 0.75$$

Deze probabiliteiten worden ook voor de andere features berekend, en wanneer ze voor alle features gekend zijn, worden ze gebruikt om nieuwe zinnen te classificeren via Formule 2.4.

2.2.3.3 Bernouilli Naive Bayes

Een derde bekende techniek is Bernouilli Naive Bayes. Deze genereert voor elke tekst voor elke term een waarde die gelijk is aan 0 of 1. 0 indien de term niet voorkomt in de tekst, en 1 indien de term wel voorkomt in deze tekst. Deze techniek houdt dus geen rekening met het aantal keer dat een term voorkomt in een tekst. Verder houdt deze techniek wel rekening met het ontbreken van een bepaald woord in een bepaalde tekst. Het model zal vervolgens $P(x_i|y)$ schatten via de fractie van teksten waarin x_i aanwezig is[2].

2.2.3.4 Besluit

Omdat Multinomial Naive Bayes rekening houdt met het aantal keer dat een woord voorkomt in een bepaalde tekst, is deze de meest aangeraden techniek om via Naive Bayes aan tekst classificatie te doen. Deze methode heeft verder reeds veelbelovende resultaten kunnen behalen bij het automatisch classificeren van berichten (11), waarbij als representatie gekozen werd voor een word count vectorisatie (en dus geen gebruik gemaakt werd van tf-idf vectorisatie).
Doordat Naive Bayes gebruik maakt van een relatief simpel principe, het theorema van Bayes, kan deze techniek ook succesvol gebruikt worden voor gevallen waar er slechts een kleine dataset beschikbaar is, in tegenstelling tot enkele technieken die later in dit hoofdstuk besproken worden. Verder belooft Multinomial Naive Bayes de beste resultaten op te leveren in het geval van SA.

2.2.4 Support Vector Machines

De bedoeling van een Support Vector Machine is het vinden van een hyperplane in een N-dimensionele hyperspace (met N het aantal features) zodat via deze hyperplane de verschillende klassen goed gescheiden zijn. Hierin is het doel het maximaliseren van de marge tussen beide klassen in deze hyperspace. In Fig. 2.1 is een voorbeeld te vinden van een SVM.

Soms wordt toegelaten enkele datapunten verkeerd te classificeren om zo de performantie van

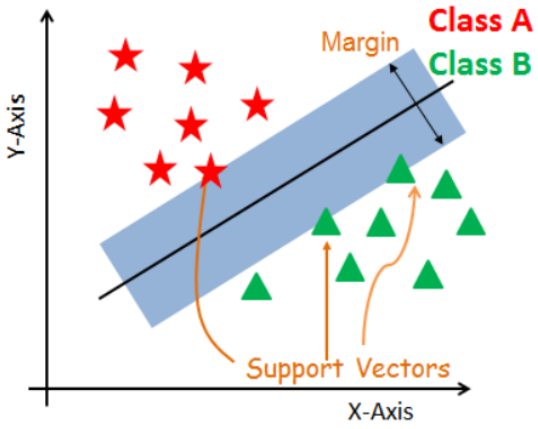
[2] https://nlp.stanford.edu/IR-book/html/htmledition/the-bernoulli-model-1.html

de SVM te verbeteren. De datapunten aan de randen van de marge worden ook wel de Support Vectors genoemd. Deze datapunten beslissen hoe breed de marge wordt en beïnvloeden dan ook waar de hyperplane zal komen te liggen.

Er werd reeds theoretisch aangetoond dat SVM's zeer goed presteren in het classificeren van teksten (4). Dit voornamelijk omdat SVM's goed kunnen omgaan met data met zeer veel features. Verder wordt ook aangetoond dat SVM's goed geschikt zijn voor problemen waarbij de document vectors zeer weinig niet-nul waarden bevatten, wat bij tekstclassificatie vaak voorkomt.

Buiten de theoretische geschiktheid van SVM's voor tekst classificatie zijn er ook verschillende studies en experimenten waarbij SVM's gebruikt werden om tekst te classificeren met goede resultaten. (4) behaalde uitstekende resultaten voor een SVM classifier op basis van twee verschillende datasets.
Een andere studie gebruikte een SVM om preferenties van gebruikers te kunnen achterhalen via hun commentaren (5). Ook hier werden zeer succesvolle resultaten behaald.

SA kan dus succesvol gedaan worden via een Support Vector Machine. In de verschillende aangehaalde studies werd echter steeds een SVM geconfigureerd via relatief grote datasets (minstens 10.000 documenten), iets wat in dit onderzoek helaas niet beschikbaar is. In een SVM dienen nog enkele parameters afgestemd te worden. Een te kleine trainingset is dus niet aangewezen, aangezien deze geen optimale hyperplane zal opleveren. Er kan dus aangenomen worden dat werken via een SVM, wat complexer is dan werken via bijvoorbeeld Naive Bayes, minder aangewezen is wanneer er gewerkt wordt met kleinere datasets.

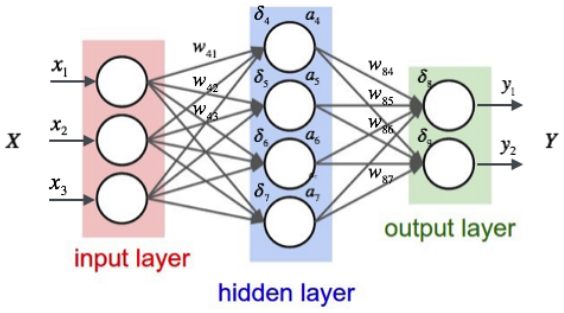


Figuur 2.1: Voorbeeld van een SVM
Afbeelding te vinden op: https://www.datacamp.com/community/tutorials/svm-classification-scikit-learn-python

2.2.5 Neurale Netwerken

Een neuraal netwerk is een netwerk van artificiële neuronen. Deze machineleertechniek is gebaseerd op het biologische neurale net dat aanwezig is in het menselijke brein. Net zoals in het menselijke brein verschillende delen verantwoordelijk zijn voor verschillende zaken, is een neuraal netwerk ook opgedeeld in verschillende lagen. Elke laag geeft informatie door aan de volgende laag, waarop deze deze informatie kan gebruiken om vervolgens weer informatie door te geven aan de volgende laag.

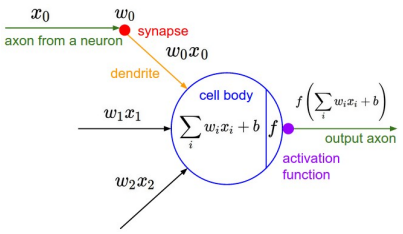
Een typisch neuraal netwerk bestaat uit tenminste drie lagen. Ten eerste een inputlaag, waarin de verschillende inputs aan de verschillende neuronen in deze laag worden gegeven. Vervolgens is er ook een ouputlaag, waarin de output terecht zal komen en via welke uiteindelijk besloten kan worden of document X bijvoorbeeld tot klasse A, B of C behoort. Tot slot is er de hidden layer, die de waarden van de inputlaag ontvangt en deze gebruikt om volgende waarden door te geven aan de outputlaag. Deze wordt de hidden layer genoemd omdat de output van deze laag niet direct zichtbaar is als een netwerk output.



Figuur 2.2: Voorbeeld van een Neuraal Netwerk
Afbeelding te vinden op: https://medium.com/@curiousily/tensorflow-for-hackers-part-iv-neural-network-from-scratch-1a4f504dfa8

Tussen alle neuronen van aanliggende lagen kunnen er ook gewichten te vinden zijn. Deze beïnvloeden hoeveel de voorgaande neuronen elk doorwegen voor het bepalen van de waarde van het volgende neuron. Figuur 2.3 toont een model van een enkele neuron. Het neuron krijgt verschillende inputs binnen, met gewichten hieraan gekoppeld. Ook wordt in de berekening een zekere bias b meegegeven. Vervolgens worden deze inputs opgeteld en via een activatiefunctie wordt bepaald wat de waarde is die doorgegeven zal worden aan de neuronen van de volgende laag. Tegenwoordig wordt als activatiefunctie vaak de ReLu-functie[3] genomen.

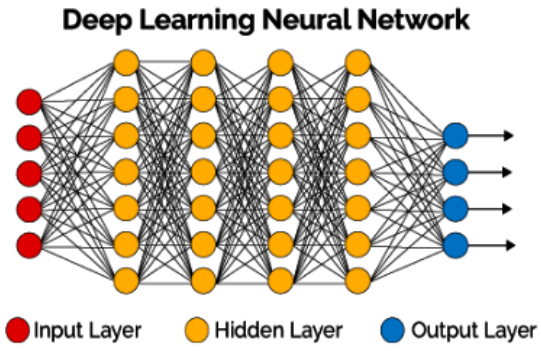
[3]Meer info: `https://en.wikipedia.org/wiki/Rectifier_(neural_networks)`



Figuur 2.3: Voorbeeld van een Neuron
Afbeelding te vinden op: http://cs231n.github.io/neural-networks-1/

De verschillende gewichten in het neuraal netwerk kunnen via trainingsdata fijngesteld worden. Zo zullen de gewichten na elk voorbeeld aangepast worden, gebruik makende van de error-rate.

Voor zeer complexe taken, zoals het herkennen van figuren op afbeeldingen, is een neuraal netwerk met slechts een hidden layer vaak niet genoeg. Vandaar dat er voor dit soort taken neurale netwerken worden opgesteld met meerdere hidden layers. Zo kan een laag instaan voor het herkennen van primaire vormen: een rechte lijn, een hoek, een boog,... Vervolgens kan de output van deze laag gebruikt worden door een volgende laag om complexere zaken te gaan herkennen: een een ronde, een haak,... Deze output kan dan weer gebruikt worden om nog complexere zaken te herkennen. Op deze manier kan een neuraal netwerk via verschillende lagen toch aan gezichtsherkenning doen. Wanneer er sprake is van verschillende hidden layers wordt dit ook wel Deep Learning Neural Network genoemd.



Figuur 2.4: Voorbeeld van een Diep Neuraal Netwerk
Afbeelding te vinden op: https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351

Als features voor het trainen van een Sentiment Analysis model kunnen verschillende vectorisaties gekozen worden. Zo kan een neuraal netwerk getraind worden met word count vectors, maar even goed ook met tf-idf vectoren waarbij n-grams gebruikt worden als vocabulary.

Er werden reeds goede resultaten behaald met Neurale Netwerken Modellen voor SA. Zo toonden (2) aan dat het achterhalen van het sentiment van twitterberichten via Neurale Netwerken een betere accuraatheid opleverde dan de modellen gecreëerd via SVM of Naive Bayes. Ook in (6) leverde

het werken met Neurale Netwerken goede resultaten op.

In dit soort neurale netwerken zijn er altijd zeer veel neuronen aanwezig, omdat tekstverwerking een complexe taak is. Omdat er verschillende lagen zijn met telkens een groot aantal neuronen, zijn er ook zeer veel verbindingen die gemaakt worden tussen de verschillende neuronen. Op al deze verbindingen staat vervolgens ook telkens een gewicht, en deze moeten ingesteld worden via trainingsdata. Om al deze gewichten goed in te stellen is een enorm grote hoeveelheid aan data nodig.
Dit is de reden waarom het niet aangeraden wordt om neurale netwerken te gebruiken bij problemen waarbij een relatief kleine hoeveelheid trainingsdata beschikbaar is. Dit is onder andere het geval in dit onderzoek. Neurale netwerken zijn dus niet interessant om in hier verder te gebruiken.

2.2.6 Besluit

Er zijn reeds veel verschillende technieken mogelijk om aan SA te doen, die allen reeds goede resultaten hebben behaald. Wegens de kleine dataset zijn er slechts twee hiervan die in aanmerking komen om te gebruiken in dit onderzoek, namelijk Rule Based Machine Learning en Multinomial Naive Bayes.

2.3 Bestaande software

In deze sectie zullen enkele alternatieven alsook software voor het verwerken van tekst uit de doeken gedaan worden. Deze zijn enerzijds alternatieven voor het ontwikkelen van eigen SA software, en anderzijds handige tools die kunnen helpen bij het ontwikkelen ervan. In elke subsectie zal een goed beeld gegeven worden van wat de software precies inhoudt, eventueel gevolgd door de verschillende beperkingen van dit ervan.

2.3.1 Natural Language Toolkit (NLTK)

NLTK is software beschikbaar in Python waarmee tekst verwerkt kan worden. Ze voorziet verschillende preprocessing technieken, waaronder methoden om tekst te tokenizen, om aan part-of-speech tagging te doen, om stopwoorden te filteren,... Deze technieken zijn beschikbaar in verschillende talen, waaronder ook het Nederlands. NLTK bezit geen directe methode om aan Sentiment Analysis te doen. Doordat deze bibliotheek verschillende goede methoden voorziet om Nederlandse teksten te preprocessen, zal NLTK gebruikt worden om tekst te tokenizen, stopwoorden te filteren, teksten en te stemmen. NLTK is dus een handige tool wanneer er met teksten gewerkt dient te worden.

2.3.2 Textblob library

Textblob is een library beschikbaar in Python. Deze bibliotheek biedt een simpele API aan waarmee verschillende Natural Language Processing technieken kunnen worden uitgevoerd. Textblob is een goede bibliotheek voor personen die willen starten met Natural Language Processing. Zo is het zeer makkelijk om stukken tekst te gaan manipuleren (bijvoorbeeld de stam van een woord nemen). Verder kan er via deze bibliotheek zeer snel aan Part-of-Speech tagging gedaan worden, het automatisch laten toekennen van zinsdelen aan een gegeven zin.

Textblob bevat een methode om van een gegeven stuk tekst het sentiment terug te geven. Concreet houdt dit in dat er een polariteitsscore wordt teruggegeven tussen 1 en -1. Deze geeft weer hoe negatief of positief de tekst is. Verder kan ook de subjectiviteit teruggegeven worden. Dit is een score tussen 0 en 1 waarbij 0 staat voor een zeer objectieve tekst, en 1 voor een zeer subjectieve tekst.

Via Textblob is het dus mogelijk om zeer snel en simpel aan SA te doen. Er zijn echter verschillende tekortkomingen aan deze bibliotheek waardoor deze niet geschikt is om te gebruiken voor deze case. Dit is ook de reden waarom deze software vooral aangeraden wordt aan beginners, die de verschillende mogelijkheden van Natural Language Processing willen verkennen en uittesten. De software wordt echter niet verder gebruikt door onderzoekers of in professionele contexten.

Een eerste beperking is het feit dat deze software niet werkt voor Nederlandstalige teksten. Dit wil zeggen dat deze teksten vertaald dienen te worden, wat ervoor kan zorgen dat deze een stuk van hun sentiment verliezen. Het zou dus interessanter zijn moest deze software werken voor Nederlandstalige teksten. Verder is deze software ook niet geschreven voor het beoordelen van commentaren op routes. Aangezien het deze context niet mee heeft zal het dus ook vaak fouten maken tijdens het beoordelen van deze commentaren.
Tot slot is ook de accuraatheid van deze software zeer laag wanneer we deze uittesten op de commentaren gegeven op verschillende routes. Dit ligt wellicht ook deels aan het missen van de context, namelijk het beoordelen van routes.

We concluderen dat deze software niet voldoet aan de eisen voor deze thesis, wegens te lage accuraatheid.

2.3.3 RapidMiner

"RapidMiner is a data science platform for teams that unites data prep, machine learning, and predictive model deployment." - (Rap)

RapidMiner is software speciaal ontwikkeld om aan data science te doen. Volgens de ontwikkleraars is er geen programmeerkennis nodig is om ze te kunnen gebruiken. Via RapidMiner kunnen dus ook personen zonder kennis van programmeren gebruik maken van data science tools.

RapidMiner voorziet verschillende handige operaties die men op de data kan uitvoeren via 'drag and drop' aanpak. De verschillende componenten kunnen ook met elkaar verbonden worden, dit is te zien op Figuur 2.5, waarbij als voorbeeld een decision tree werd opgesteld. Data manipuleren gaat dus snel en gemakkelijk. Via RapidMiner is het mogelijk om verschillende machine learning modellen op te bouwen (bijvoorbeeld een Neural Network of een Decision Tree). Verder is ook functionaliteit voorzien voor toegang tot data, het preprocessen van data en de validatie van modellen.

Wat RapidMiner in dit specifiek geval echter interessant maakt is de mogelijkheid om SA extensions toe te voegen. Er zijn vervolgens verschillende providers die dit soort extensions aanbieden, in het bijzonder Aylien, Twinword en Rosette.

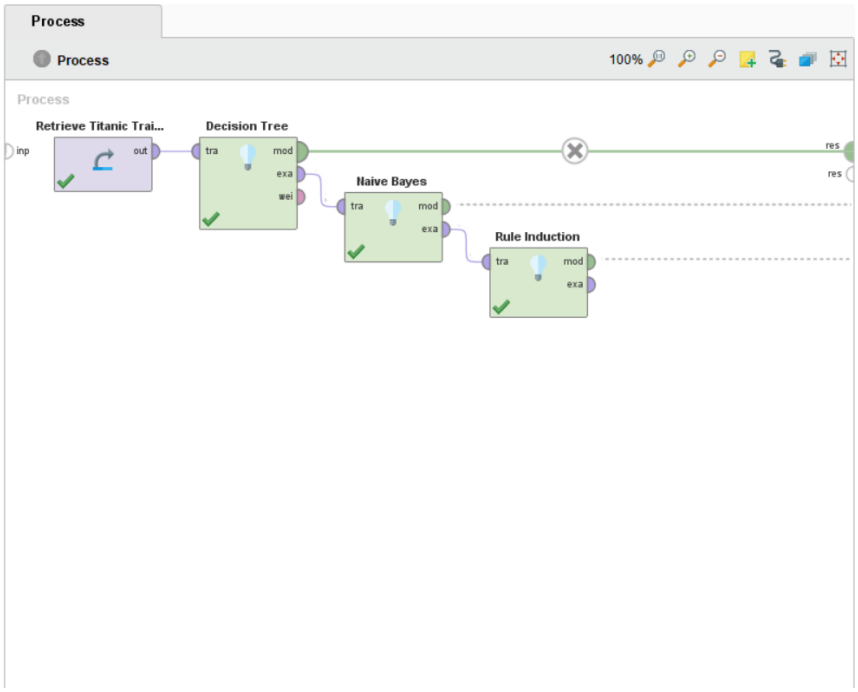
Er zijn echter ook aan deze software verschillende beperkingen. Een eerste beperking en wellicht de grootste is het feit dat de extensions om aan SA te doen niet gratis zijn. Zo voorzien alle providers een gratis demo om enkele voorbeelden te testen, maar zal vanaf een bepaalde threshold geld gevraagd worden om aan SA te doen.
Een andere beperking van deze extensions is dat ze SA API's voorzien voor teksten die in het Engels geschreven staan, maar geen ondersteuning bieden voor Nederlandstalige teksten.
Verder is de software ook niet direct integreerbaar met een apart geschreven programma, wat het moeilijker zal maken om automatische beoordelingen van commentaren te voorzien.

De software voorziet dus mogelijkheden om aan Data Science en in het bijzonder aan SA te doen, maar is gezien zijn prijs en het feit dat het aparte software is uiteindelijk toch minder geschikt om te gebruiken in deze case.

2.3.4 Repustate

Repustate is een text analytics engine die SA, entity extraction en topic detection voorziet in verschillende talen, waaronder het Nederlands, Engels, Duits en Frans. De tekst analyse is speciaal ontworpen voor sociale media, en zou dus ook uitstekend kunnen werken voor het verwerken en beoordelen van commentaren op routes.
Aangezien de context waarin aan SA wordt gedaan zeer belangrijk is, en elke verschillende industrie zijn eigen jargon kan hebben, zorgt Repustate ook dat er eigen regels kunnen toegevoegd worden aan de API.
Repustate beweert op hun website een accuraatheid te kunnen geven van 82% tot wel 90%, wat een zeer goede accuraatheid voor de case van deze thesis.

Een grote beperking aan deze software is dat deze niet gratis is. Zo voorziet Repustate via zijn website een demo om de software te testen, maar eenmaal men deze in bedrijfscontext wil gebruiken, dient er betaald te worden.
Een verdere beperking is dat de API van Repustate een RESTful API is. De communicatie tussen



Figuur 2.5: Voorbeeld van de constructie van een Decision Tree via RapidMiner

de bestaande software en de API zal dus moeten verlopen via http-requests, met de beoordelingen van de commentaren in JSON formaat. Dit maakt het beoordelen van commentaren iets ingewikkelder dan software die toegevoegd kan worden aan de reeds bestaande software van RouteYou, waarbij communicatie intern in de software gevoerd kan worden. Dit is echter slechts een kleinere beperking.

Ondanks de belofte van de goede accuraatheid is het ontwikkelen van een eigen systeem dus toch aangeraden, aangezien dit een gratis alternatief is en het makkelijk kan toegevoegd worden aan de reeds bestaande software van RouteYou.

2.4 Besluit

De alternatieven voor het ontwerpen van een eigen systeem zijn financieel gezien niet interessant, omdat deze alternatieven betalend zijn, ofwel in het geval van de gratis alternatieven geen goede resultaten opleveren. Het is dus aangeraden om zelf een model te gaan ontwikkelen.

Voor het ontwikkelen van een model werden verschillende technieken bestudeerd die in het verleden reeds tot goede resultaten geleid hadden bij het analyseren van tekst. Hieruit vielen specifiek de Rule Based approach alsook Naive Bayes op, omdat deze als groot voordeel hebben dat ze zeer goed blijken te werken wanneer er slechts een kleine hoeveelheid data ter beschikking is. Er wordt dan ook besloten om verder in deze studie te werken met deze twee modellen.

Hoofdstuk 3

Data exploratie

In dit hoofdstuk word de dataset waarmee gewerkt werd in deze studie toegelicht. Eerst wordt wat uitleg gegeven over de verwerving van de data. Hierna volgen er enkele statistieken over de data, om zo meer inzicht te krijgen over de inhoud van de dataset. Tot slot worden ook de verschillende soorten commentaren toegelicht, zoals de verschillende talen waarin deze voorkomen.

3.1 Verwerving

In deze studie werd gewerkt met een dataset aangeboden door RouteYou. Deze dataset bevat alle commentaren gegeven op alle routes van RouteYou vanaf 2 februari 2007 tot aan 13 maart 2019. Voor elke commentaar worden de volgende attributen bijgehouden:

a) Id van de gebruiker

b) Tekst

c) Datum van plaatsing

d) Id van route waarop de commentaar geplaatst werd

Hieronder wordt een voorbeeld gegeven van een commentaar uit de dataset.

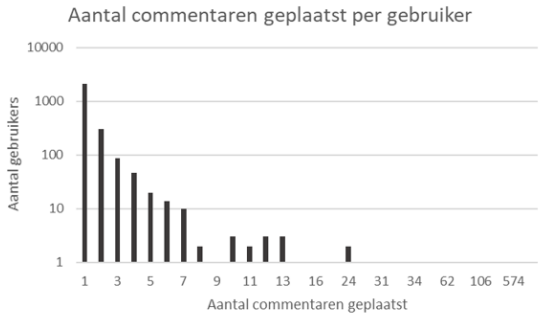
Id	Text	Owner_Id	Route_id	Created_Date
26287	<p>Ziet er goed uit<p />	3	1328	2007-04-23 11:20:37

Aangezien er in deze studie gewerkt werd via supervised machine learning, waarbij er vanuit wordt gegaan dat de data reeds gelabeled is, is de gekregen data van RouteYou vervolgens zelf manueel beoordeeld door aan elke commentaar mee te geven of deze positief, neutraal of negatief is. Dit werd gedaan door elke commentaar een extra waarde toe te kennen. Deze waarde schommelde tussen -2 en 2. -2 voor zeer negatieve commentaren en 2 voor zeer positieve commentaren.

Deze manuele beoordeling werd niet gedaan door personen van RouteYou zelf. Een kleinere subset met beoordeelde commentaren werd vervolgens voorgelegd om te zorgen dat deze beoordeling overeenkwam met de beoordelingen die vanuit RouteYou zouden gegeven worden.

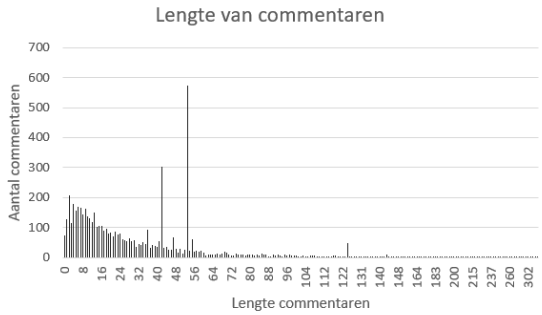
3.2 Algemene statistieken

De dataset waarmee gewerkt wordt bestaat uit 5830 commentaren. Van deze commentaren waren er 2067 neutraal, 3013 positief en 750 negatief. Deze commentaren werden geplaatst door 2602 unieke gebruikers. In Figuur 3.1 is terug te vinden hoeveel commentaren deze gebruikers elk plaatsten. Het grootste deel van deze gebruikers plaatsten slechts één enkele commentaar. Enkele gebruikers plaatsten meer dan 100 commentaren. Deze gebruikers waren echter steeds beheerders van de website, en plaatsten eerder informatieve commentaren.



Figuur 3.1: Aantal commentaren geplaatst per gebruiker.

De gemiddelde commentaar telde 33,35 woorden, met als standaardafwijking hierop 35,90. Dit is een vrij grote standaardafwijking. Dit kan liggen aan het feit dat vele gebruikers zeer korte commentaren leveren, en er ook vele langere commentaren gegenereerd worden door de beheerders van de website, om mee te delen dat de route is opgenomen in een verzameling van mooiste routes, om mee te delen dat de route rolstoelvriendelijk is,... De uiteenzetting van alle commentaren met hun lengte is terug te vinden op Figuur 3.2. Hierop is te zien dat er voor enkele grote lengtes enorm veel commentaren te vinden zijn. Dit zijn wellicht automatisch gegenereerde commentaren die terug te vinden zijn op verschillende routes, en eerder algemene info meedelen. Een voorbeeld is de commentaar geplaatst op routes die rolstoelvriendelijk zijn. Deze is op meer dan 500 routes geplaatst, en zou dan ook de piek kunnen verklaren te zien op Figuur 3.2.



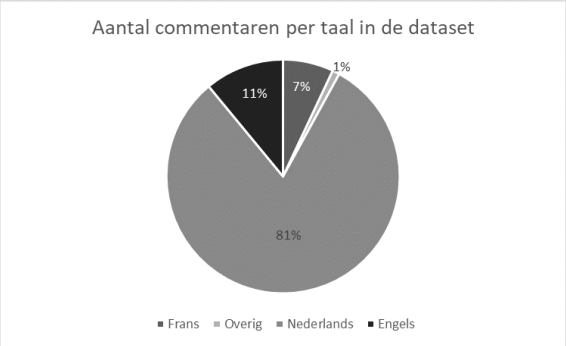
Figuur 3.2: Aantal commentaren met een bepaalde lengte.

In alle commentaren waren in totaal 194.413 woorden te vinden. Hiervan waren er uiteindelijk 17.328 uniek. In bijlage B is verder een lijst terug te vinden met de top 200 meest voorkomende woorden. Dit voor zowel gestemde als ongestemde data. Uit deze lijsten zijn de stopwoorden reeds gefilterd.

3.3 Soorten commentaren

De inhoud van de commentaren die aanwezig zijn in de dataset varieert sterk. Zeer vaak gaan de commentaren effectief over de routes, waarbij de gebruiker meedeelt waarom die de route al dan niet goed of slecht vindt. Een voorbeeld:
"Geweldige route, vooral bij mooi weer. Lekker uitleven tegen de redelijke klimmetjes."
Er zijn ook commentaren die geen sentiment uitdrukken, zoals een vraag, extra info over de route voorzien door de beheerders of een link naar relevante informatie met betrekking tot de route. Een voorbeeld: *"Ik denk deze route zelf nog eens te gaan rijden een dezer dagen.. Wie heeft er zin en is vrij?"*

Tot slot komen de commentaren ook voor in verschillende talen. Op Figuur 3.3 is de het percentage commentaren per taal in de dataset uitgezet. Zo'n 81% van alle commentaren zijn Nederlandse commentaren. Ongeveer 11% van de commentaren waren vervolgens in het Engels geschreven, en 7% van de commentaren waren Frans. Verder kwamen er nog commentaren voor in het Italiaans, Duits en Spaans. Deze maakten slechts ongeveer 1% van alle commentaren uit. Aangezien het aantal commentaren steeds toeneemt en de diensten van RouteYou steeds meer in het buitenland gebruikt worden, kan er ook aangenomen worden dat het aantal anderstalige commentaren in de toekomst nog zal toenemen.



Figuur 3.3: Percentage commentaren per taal in de dataset

3.4 Besluit

De beschikbare data varieert sterk in lengte, alsook in inhoud. Verder is de dataset van beperkte grootte, wat het werken met machineleertechnieken zoals Neurale Netwerken uitsluit. Het is dus nodig om een model te ontwikkelen dat met deze variaties weet om te gaan en goed kan presteren wanneer er slechts een kleine dataset ter beschikking is.

Hoofdstuk 4

Preprocessing and Noise Cancelling

De data aangeboden door RouteYou kan niet direct gebruikt worden voor Sentiment Analysis, omdat er nog enorm veel ruis zit op deze data, die de performantie nadelig zou beïnvloeden. Zo bevatten deze commentaren bijvoorbeeld nog veel html syntax en spelfouten. Verder is er stemming nodig om afgeleiden van dezelfde woorden te kunnen herleiden naar eenzelfde stam, waardoor het lexicon voor al de afgeleiden slechts één woord hoeft te hebben. Het is dus aangeraden om deze data eerst te preprocessen om de ruis klein mogelijk te maken en de performantie te optimaliseren.

4.1 Filtering

In de meeste commentaren zit nog enorm veel tekst die niet bijdraagt tot het sentiment van deze commentaar. Voorbeelden zijn vragen zoals *"Is dit een goede route?"* of *"Kan deze route ook gereden worden met een mountainbike?"*
In dit voorbeeld is het duidelijk dat deze vragen geen extra bijdrage leveren aan het sentiment. Vandaar dat er besloten werd om vragen uit de commentaren te filteren.

Buiten vragen werden ook de stopwoorden uit de commentaren gefilterd, omdat stopwoorden kleine, vaak voorkomende woorden zijn die geen sentiment meedragen. Dit zal dus de ruis verminderen, alsook de performantie verhogen. Stopwoorden die mogelijk wel sentiment hebben werden eerst uit de stopwoorden lijst gefilterd. Dit waren drie woorden, namelijk *'Niet'*, *'Geen'* en *'Veel'*Woorden zoals 'niet' zijn immers zeer belangrijk voor Sentiment Analysis.

Tot slot werd ook de html syntax uit alle commentaren verwijderd. Dit ging van html-tags zoals <p />tot het verwijderen van hyperlinks, omdat deze tekens niet bijdragen tot sentiment en enkel kunnen bijdragen tot de ruis.

4.2 Translating

Commentaren komen voor in verschillende talen. In deze studie zal onder andere getest worden of het vertalen van anderstalige commentaren naar het Nederlands een grote invloed heeft op het sentiment dat in deze commentaren aanwezig was. Er wordt de hypothese gesteld dat het sentiment grotendeels behouden blijft wanneer een tekst vertaald wordt naar een andere taal. Aangezien er gewerkt wordt met een kleine dataset, en slechts 20% van deze dataset niet-Nederlands is, is het interessant om na te gaan of vertaalde commentaren gelijkaardig verwerkt kunnen worden als Nederlandse commentaren door een model gemaakt voor Nederlandse commentaren. Vandaar dat de commentaren ook eerst vertaald moeten worden vooraleer ze worden doorgegeven naar het model dat zorgt voor de Sentiment Analysis.

Vooraleer een commentaar vertaald wordt naar het Nederlands, zal eerst nagegaan worden in welke taal de commentaar geschreven is via een vertaal-bibliotheek die steunt op de vertaling van Google Translate[1]. Indien gedetecteerd wordt dat deze commentaar niet in het Nederlands is geschreven zal deze vervolgens automatisch naar het Nederlands vertaald worden.

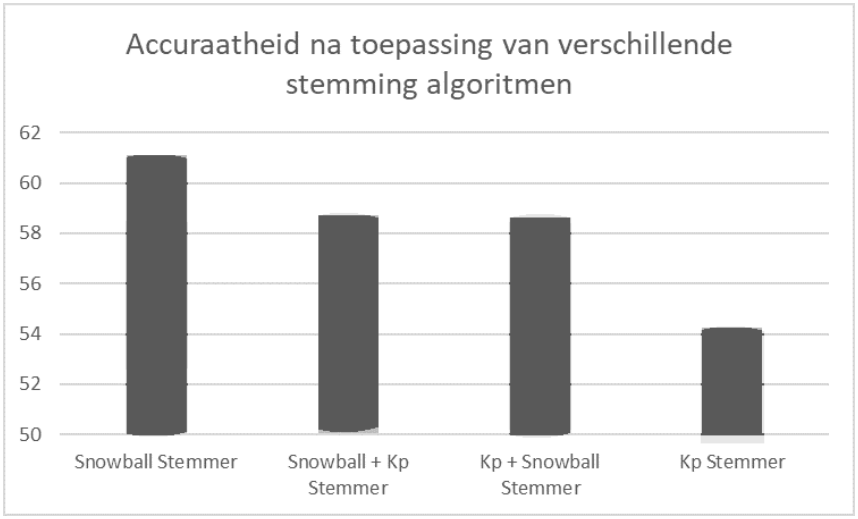
4.3 Tokenization & stemming

Om aan tekst classificatie te doen is het belangrijk dat zinnen worden omgezet naar een lijst van woorden. Dit omdat het de aparte woorden zijn die doorgegeven worden aan machine learning modellen. Deze gebruiken ze als features om op te trainen. Het proces waarbij een zin wordt opgedeeld in een lijst van zijn woorden wordt tokenization genoemd, en is dus belangrijk wanneer tekst dient geclassificeerd te worden.

Een laatste stap die dient te gebeuren is stemming, oftewel het herleiden van de woorden naar hun stam, om enorm veel ruis uit de data te krijgen. Zo worden woorden zoals *leuk*, *leuker* of *leukst* allemaal naar dezelfde stam herleid, namelijk 'leuk'. Dit zorgt ervoor dat er veel minder features zijn waarop machine learning technieken moeten trainen. Wel gaat mogelijks een deel van de betekenis van het woord verloren. *'Leuk'* is namelijk niet helemaal gelijk aan *'Leukst'*. Wel drukken beide woorden een positief sentiment uit. Hierdoor is er uiteindelijk toch besloten deze stemming door te voeren. Waar het model normaal drie aparte features zou hebben, moet het nu slechts één feature gebruiken om alle afleidingen van het woord 'leuk' hetzelfde te behandelen. In deze studie werd vergeleken welke stemmer het beste gebruikt wordt voor de Nederlandse dataset. Er werd geëxperimenteerd met twee stemmers die goed werken voor Nederlandse teksten, namelijk de Snowballstemmer[2] en het Kraaij-Pohlmann stemming algoritme[3]. Hieruit bleek dat de beste resultaten werden behaald wanneer er gewerkt werd met enkel de Snowballstemmer. Dit is te zien op Figuur 4.1. Dit experiment werd uitgevoerd bij het begin van de studie, om een stemming

[1] Meer informatie over deze bibliotheek is te vinden op `https://pypi.org/project/googletrans/`
[2] Informatie over Snowball stemmer beschikbaar op `http://www.nltk.org/howto/stem.html`
[3] Informatie over Kp-stemmer beschikbaar op `https://github.com/thomasbrockmeier/kpss_py3`

algoritme te kunnen kiezen waarmee verder gewerkt kon worden gedurende de studie.



Figuur 4.1: Accuraatheden voor verschillende (combinaties van) stemming algoritmen.

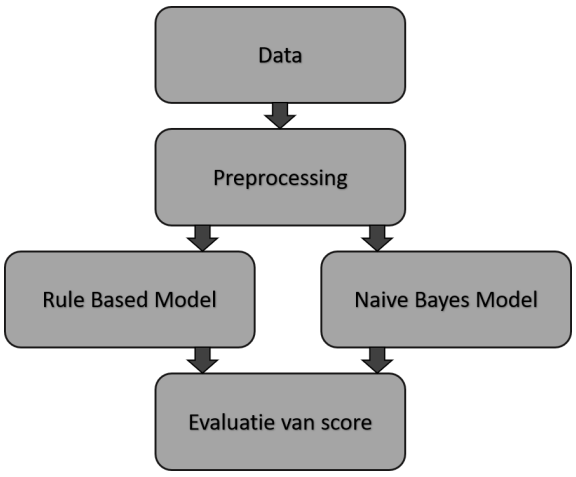
4.4 Besluit

Na het uitvoeren van deze verschillende stappen kan de herwerkte data vervolgens doorgegeven worden aan de verschillende modellen. Deze zullen de data vervolgens gebruiken om er een sentimentsscore uit te halen. De werking van deze modellen wordt verder besproken in het volgende hoofdstuk. Door het wegwerken van de ruis op deze data zullen de modellen in staat zijn veel betere voorspellingen te doen over de testdata, en zodoende de accuraatheid aanzienlijk verhogen in vergelijking met het werken met de ruwe data.

Hoofdstuk 5

Ontwerp van de modellen

In dit hoofdstuk worden de twee gerealiseerde modellen beschreven. Hierin wordt uitgelegd hoe de modellen zijn opgebouwd en uit welke componenten ze bestaan, alsook de inhoud van al deze componenten. In Figuur 5.1 is een schematische voorstelling gegeven van de workflow van beide modellen.



Figuur 5.1: Schematische voorstelling van de workflow van beide modellen

Vervolgens genereren beide modellen een score voor de gegeven data. Hoe de score berekend wordt en hoe de modellen getraind werden is terug te vinden verder in dit hoofdstuk. Tot slot wordt de gegenereerde score geëvalueerd. Deze evaluatie is voor beide modellen verschillend. Deze evaluatie wordt diepgaand besproken in de uiteenzetting van beide modellen.

5.1 Het Rule Based Learning model

Een eerste gerealiseerde model is een model op basis van Rule Based Learning. Dit bestaat uit een paar verschillende componenten. Enerzijds is er het lexicon, waarin bijgehouden wordt welke

woorden herkend moeten worden en welk gewicht ze meekrijgen. Verder zijn er ook de regels, die met behulp van het lexicon een score zullen berekenen voor een bepaalde tekst. Tot slot is er ook een evaluatiefunctie die de score zal interpreteren om vervolgens een label aan de tekst te geven.

5.1.1 Het lexicon

Het lexicon is een verzameling van alle woorden die bijdragen aan het sentiment van een commentaar. Dit lexicon bestaat in deze studie uit twee grote delen. Ten eerste de bijvoeglijke naamwoorden (verder adjectieven genoemd), en ten tweede de bijwoorden. Om te beslissen welke woorden allemaal herkend moesten worden door het initiële model werden eerst de top 200 meest voorkomende woorden van de commentaren bekeken. Deze top 200 meest voorkomende termen werden eens berekend voor de gewone commentaren, alsook voor de commentaren die eerst aan preprocessing werden onderworpen. De lijsten met deze 200 termen zijn terug te vinden in bijlage B.

Vervolgens worden deze lijsten manueel overlopen, en wordt voor elk woord nagegaan of dit een gevoelslading -sentiment- draagt. Voor de meeste woorden is dit niet het geval, aangezien het vaak gaat om zelfstandige naamwoorden zoals *'Route'* of *'Wandeling'*. In de lijst van ongestemde woorden zijn er van de 200 woorden 25 te vinden met eventueel sentiment erin. In de lijst van gestemde woorden zijn er vervolgens 26 te vinden. Aangezien vele woorden in beide lijsten voorkomen, is het uiteindelijke aantal unieke woorden dat gevoelslading draagt gelijk aan 28. Wanneer er sprake was van een term die sentiment met zich meedraagt wordt er een gewicht aan deze term meegegeven en bijgehouden in het lexicon. Aangezien dit gewicht ook afhangt van de rol van het woord, een adjectief of een bijwoord, zal de toekenning van de gewichten verder besproken worder in de subsecties over deze adjectieven en bijwoorden.

Het lexicon wordt verder aangevuld met synoniemen van deze woorden, gebruik makend van online websites[1]. Ook werd de kennis vergaard door het manueel beoordelen van alle commentaren gebruikt om het lexicon hierna verder aan te vullen met voor de hand liggende termen.

5.1.1.1 De adjectieven

Deze groep bestaat uit alle woorden die duidelijk een sentiment uitdrukken. Zo bevat deze groep woorden zoals *'goed'*, *'mooi'* of *'rustig'*. Een volledige lijst van alle adjectieven aanwezig in het recentste model is te vinden in bijlage A. Voor het meegeven van initiëel gewicht aan al deze woorden werd gekozen voor het starten met één woord. In dit geval werd er gestart met het woord *'Goed'*. Hieraan werd vervolgens de waarde 1 gegeven. Dit is een positieve waarde omdat het woord *'Goed'* een positief sentiment met zich meedraagt. Vervolgens werden alle andere woorden overlopen, en vergeleken met het woord *'Goed'*. Indien het sentiment gelijkaardig was

[1] `https://synoniemen.net/`
`https://www.mijnwoordenboek.nl/`
`https://nl.wiktionary.org/wiki/Hoofdpagina`

aan het sentiment van *'Goed'*, werd het gewicht ook gelijkgesteld aan 1. Indien het beduidend positiever was werd er aan dit woord de waarde 2 mee gegeven. Indien het een gelijkaardige maar tegengesteld sentiment bevatte werd de waarde -1 mee gegeven, en indien het een enorm negatief sentiment met zich meedroeg werd de waarde op -2 geplaatst. Indien het sentiment slechts lichtjes positief of negatief was, werd de waarde respectievelijk op 0,5 of -0,5 geplaatst.

5.1.1.2 De bijwoorden

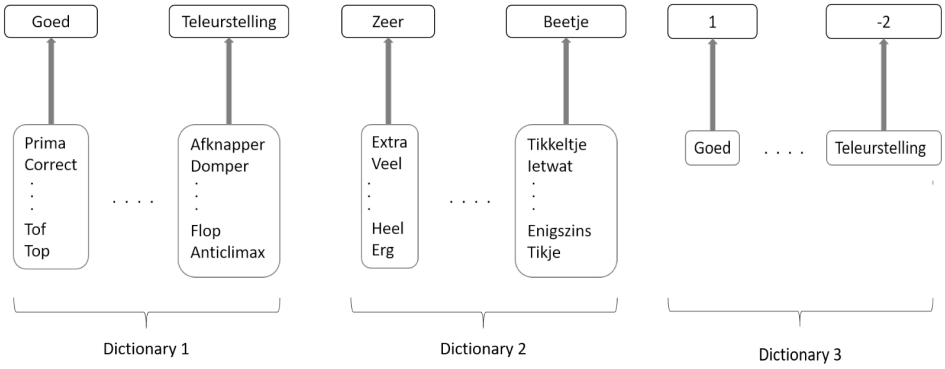
De bijwoorden zijn woorden die wanneer ze alleen staan geen sentiment uitdrukken, maar wanneer samengevoegd met een adjectief of een ander bijwoord het totale sentiment hiervan sterk kunnen beïnvloeden. Als voorbeeld kan het woord *'Heel'* gegeven worden. Het woord alleen zegt zeer weinig, maar wanneer samengevoegd met een adjectief zoals *'Goed'* valt duidelijk op dat het sentiment van *'Heel goed'* beduidend positiever is dan het sentiment van louter *'Goed'*.

Als waarde voor elke bijwoord werd het percentage genomen waarmee het bijwoord het gewicht van een volgend woord zal versterken. Een bijwoord met gewicht 1 zal het gewicht van het volgende woord met 100% doen toenemen. In bovenstaand voorbeeld is de score van de zin *'Goed.'* gelijk aan 1, terwijl de score van de zin *'Heel goed.'* gelijk zal zijn aan 2.

Bijwoorden kunnen woorden ook negatief beïnvloeden. Wanneer er een bijwoord met negatieve waarde gekoppeld wordt aan een adjectief het gewicht van dit adjectief dalen indien het sentiment hiervan positief was, en stijgen indien het sentiment negatief was. Dit kan er zelfs voor zorgen dat het sentiment van het adjectief zal veranderen van teken. Zo zorgt het bijwoord *'Niet'* door zijn waarde van -2 ervoor dat positieve woorden negatief worden, en negatieve woorden positief worden. Verdere uitleg over de samenwerking van bijwoorden en adjectieven wordt gegeven in de subsectie over de regels van het Rule Based Machine Learning model.

5.1.1.3 Realisatie

Het lexicon werd concreet gerealiseerd door het werken met dictionaries. Dit zijn structuren waarbij één sleutel kan verwijzen naar een lijst van waarden, horende bij deze sleutel. Een eerste dictionary hield alle adjectieven bij. Hierbij werden alle gelijkaardige woorden die dus ook eenzelfde gewicht hadden samen bijgehouden, met 1 van deze woorden als sleutel, en de andere als waarden horende bij deze sleutel. Dit werd vervolgens ook gedaan voor alle bijwoorden. Tot slot werden alle waarden voor elk woord bijgehouden in een derde dictionary. Deze bevatte telkens een sleutel uit de vorige twee dictionaries, gevolgd door een gewicht. Hierdoor werd aan elk woord horende tot deze sleutel dezelfde waarde meegegeven. Voorbeelden van deze drie dictionaries zijn te zien op Figuur 5.2.



Figuur 5.2: Voorbeelden van de drie aanwezige dictionaries.

5.1.2 De regels

De basis van het Rule Based model is de Adverb-Adjective Combination (AAC). In Algoritme 1 is toegelicht hoe het scoringsalgoritme werkt. Het algoritme zal altijd op zoek gaan naar adjectieven en bijwoorden. Dit doet het door de commentaar, die binnenkomt als een lijst van woorden, woord per woord te overlopen. Hierbij zal het voor elk woord nagaan of dit woord een adjectief is, een bijwoord, of geen van beide. Dit kan zeer snel gebeuren door het gebruik van de reeds uitgelegde dictionaries.

Het komt vaak voor dat er in een commentaar meerdere bijwoorden na elkaar worden gebruikt. Een voorbeeld hiervan is *'Heel erg goed'*. Hierbij ziet men dat de eerste van de twee bijwoorden een invloed heeft op de tweede. Vandaar dat het algoritme ook een *TotaleModifier* zal bijhouden. Hierin zal de uiteindelijke waarde terechtkomen die gebruikt moet worden om de waarde van het adjectief aan te passen. De *TotaleModifier* van n opeenvolgende bijwoorden wordt als volgt berekend:

$$TotaleModifier = Gewicht(Bijwoord_1) + ... + Gewicht(Bijwoord_n) \tag{5.1}$$

Wanneer het algoritme een woord met gewicht tegenkomt, zal het nagaan of het een adjectief is. Indien dit het geval is en er geen bijwoorden voor dit woord komen kan de score simpelweg verhoogd worden met het gewicht van dit adjectief. Wanneer er wel een reeks van bijwoorden voor

het adjectief komen, zal de score van het commentaar aangepast worden via volgende formule:

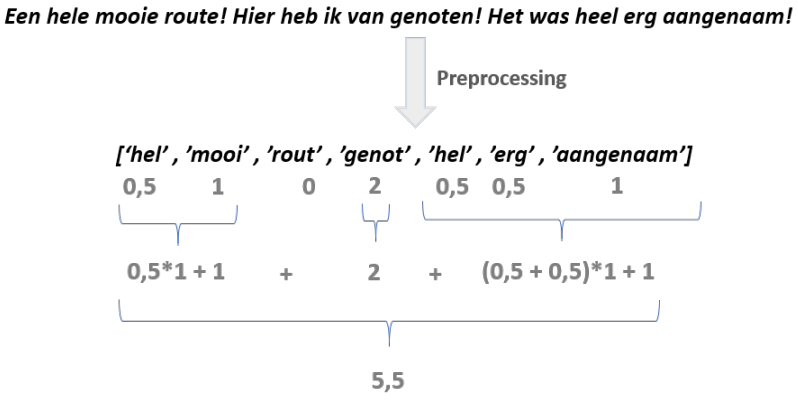
$$TotaleScore+ = gewicht(Adjectief) + TotaleModifier * gewicht(Adjectief) \quad (5.2)$$

Op deze manier loopt het algoritme door de gehele tekst, waarna het op het einde een totale score overhoudt. Deze score is de uiteindelijke score voor de tekst en kan vervolgens naar de evaluatiefunctie gestuurd worden. Om het algoritme nog wat duidelijker te maken is in Figuur 5.3 een voorbeeld weergegeven.

De eerste stap is het uitvoeren van de preprocessing. Vervolgens kan de zin gebruikt worden door het Rule Based model. Het model zal eerst het woord *'hel'* herkennen. Hierop haalt het de waarde op vanuit het lexicon, waarna de waarde van de TotaleModifier op 0,5 gezet wordt. Het volgende woord wordt weeral herkend, deze keer als een adjectief. Hierop zal het Rule Based model de totale score van de commentaar aanpassen via Formule 5.2. Ḣierdoor komt de totale score op 1,5 te staan.

Het volgende woord is het woord *'route'*. Dit draagt echter geen gewicht met zich mee en zal dus ook niet gebruikt worden door het Rule Based model. Dit model zal gewoon verder kijken naar het volgende woord in de lijst. Dit is het woord *'genot'*. Dit wordt weer herkend als een adjectief, waarna de totale score verhoogd wordt met de waarde 2. Aangezien er geen bijwoord voor dit woord komt, zal de TotaleModifier niet in rekening worden gebracht en wordt de totale score slechts verhoogd met het gewicht van *'genot'*.

Het volgende woord wordt weer herkend als een bijwoord. Hierdoor zal de TotaleModifier gelijk gesteld worden aan de waarde van dit bijwoord. Het woord na het bijwoord is weer een bijwoord, waardoor de waarde van de TotaleModifier verhoogd wordt met de waarde van het huidige bijwoord. Hierdoor komt de waarde van de TotaleModifier op $0,5+0,5=1$ te staan. Tot slot herkent het model het laatste woord als een adjectief, waarna het de totale score weer verhoogt via formule 5.2. De uiteindelijke score komt zo op 5,5 te staan.



Figuur 5.3: Werking van het Rule Based model voor een voorbeeldzin

Het scoringsalgoritme, te zien in Algoritme 1, houdt dus rekening met verschillende opeenvolgende bijwoorden, die samen het sentiment van een adjectief kunnen beïnvloeden. Door het werken via AAC kunnen negaties (*'niet goed'*), versterkingen (*'zeer slecht'*) en verzwakkingen (*'beetje goed'*) succesvol verwerkt worden. Omdat voor elke extra adjectief-bijwoord groep die voorkomt in de commentaar de totale score wordt aangepast, is het bereik waarin scores kunnen voorkomen groot, en zou deze in theorie kunnen reiken van $-\infty$ tot ∞. Vervolgens worden deze scores naar de evaluatiefunctie gestuurd, die deze scores zal omzetten tot een label. Dit label zal aangeven of de commentaar positief, negatief of neutraal is.

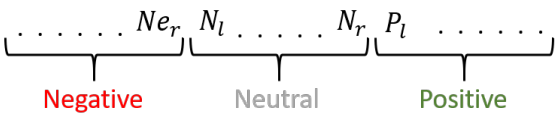
```
Data: Lijst van woorden waaruit de commentaar bestaat
Result: Score die het sentiment van de commentaar weergeeft
Initialiseren van het lexicon
TotaleScore=0;
TotaleModifier=0;
for Elk woord van de lijst do
    CheckIfInLexicon();
    if Aanwezig in lexicon then
        if Woord is geen adjectief then
            while Volgend woord is geen adjectief do
                TotaleModifier = TotaleModifier +Gewicht(Volgend woord);
                Huidig woord = Volgend woord;
                Volgend woord = Woord na huidig woord;
            end
            TotaleScore = Gewicht(Huidig woord)*TotaleModifier;
        else
            TotaleScore =+ Gewicht(woord);
        end
    else
        Einde();
    end
end
```

Algorithm 1: Algoritme voor het toekennen van scores

5.1.3 De evaluatie

De evaluatiefunctie is een simpele functie die ervoor zorgt dat elke score wordt omgezet naar een label. Deze functie werd gerealiseerd door het aanmaken van een neutrale zone, te zien op figuur 5.4. Er worden twee waarden ingesteld, N_l en N_r, die de grenzen van de neutrale zone zullen bepalen. De neutrale zone kan dus voorgesteld worden als het interval $[N_l, N_r]$. Elke commentaar waarvan de waarde binnen dit interval valt, zal als neutraal gelabeld worden. Elke commentaar waarvan de waarde groter is dan N_r zal als positief gelabeld worden, en indien de waarde kleiner

is dan N_l zal deze als negatief gelabeld worden.



Figuur 5.4: Visualisatie van de neutrale zone

Door het aanpassen van de grenzen van de neutrale zone kan gewijzigd worden hoe positief of negatief een commentaar dient te zijn vooraleer deze respectievelijk als negatief of positief wordt gelabeld. Bij zeer grote grenswaarden zullen commentaren al extreme scores moeten halen vooraleer hun sentiment gedetecteerd wordt door de evaluatiefunctie. Bij kleinere grenswaarden zal een commentaar al rapper als positief of negatief geclassificeerd worden. Door een grotere neutrale zone kan men dus zekerder zijn dat wanneer een commentaar als positief of negatief wordt geclassificeerd deze commentaar ook effectief positief of negatief is.

5.1.4 Gebruik van feedback

Het uitwerken van dit Rule Based model gebeurde in verschillende stappen. Er werd eerst een initiële implementatie voorzien, waarbij een initieel lexicon aanwezig was zoals beschreven in Sectie 5.1.1. Verder werden er initiële regels opgesteld waarbij nog geen rekening gehouden werd met de mogelijkheid van het voorkomen van verschillende bijwoorden na elkaar. Deze regels hielden enkel rekening met het voorkomen van een adjectief, of het voorkomen van een bijwoord gevolgd door een adjectief. Om het model te evalueren werden alle commentaren op voorhand opgesplitst in een trainingsset en een testset. 80% van de commentaren werd als trainingsset gebruikt, en 20% als testset. Hierna werd de initiële implementatie gebruikt om zowel de trainingsset als testset te evalueren. Via de trainingsset kon snel nagegaan worden waar de meeste fouten zich voordeden, omdat alle fout geclassificeerde commentaren als output werden meegegeven. Naast enkel de tekst werd ook de gegeven score samen met de gewenste score (het manueel gegeven label) mee als output gegeven. Hieruit konden vervolgens verschillende adjectieven alsook verschillende aanpassingen van de regels gehaald worden, door na te gaan wat de meeste fout geclassificeerde commentaren gemeenschappelijk hadden. Telkens het model aangepast werd, werden de datasets weer geherevalueerd, waarna het model weer verbeterd kon worden. De testset werd enkel gebruikt om de accuraatheid van het model in beeld te kunnen brengen. De commentaren in de testset werden niet bekeken om overfitting van het model tegen te gaan. Bij overfitting zou het model zeer goed presteren op alle zinnen van de dataset, maar wanneer nieuwe zinnen beoordeeld zouden worden, waarbij de zinsbouw licht afwijkt van de zinnen in de dataset, zou het model deze niet goed kunnen classificeren.

5.1.5 Besluit

Om dit Rule Based Machine Learning model te realiseren werd gewerkt via AAC, een goed gekozen evaluatiefunctie en een procedure waarbij het model via trainings - en testset aangepast en verbeterd werd. Doordat elke component aangepast kon worden zonder dat de andere componenten ook aangepast moesten worden kon het model vlot aangepast en verbeterd worden.

5.2 Het Multinomial Naive Bayes model

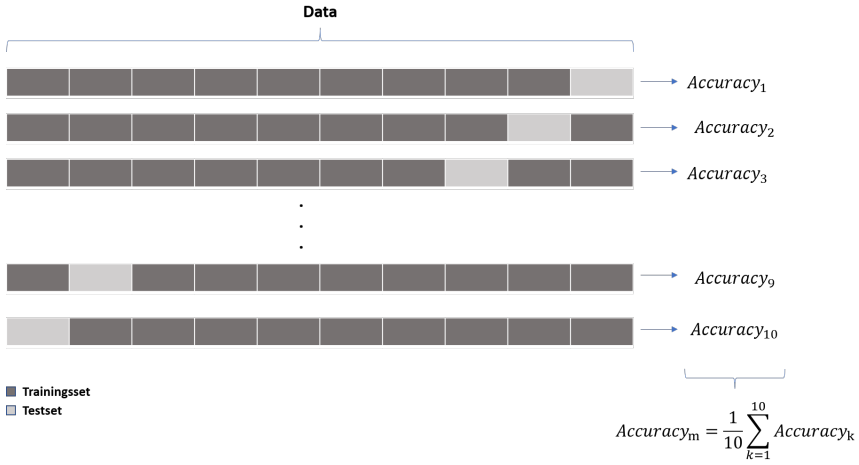
Het Multinomial Naive Bayes model bestaat ook uit verschillende componenten. Ten eerste moet er eerst een vectorisatie uitgevoerd te worden op de data, vervolgens worden deze vectors naar het model gestuurd, en tot slot wordt gewerkt met cross validation om een goed beeld te krijgen van de accuraatheid.

5.2.1 Vectorisatie

Nadat de ruis van de commentaren was verwijderd zoals beschreven in Hoofdstuk 4, werden de commentaren eerst omgezet in vectoren vooraleer deze gebruikt konden worden door het Naive Bayes model. Hiervoor werden 4 verschillende implementaties voorzien, gebaseerd op de representaties beschreven in Hoofdstuk 2. Een eerste vectorisatie werd gerealiseerd door te werken met word count vectors (n-grams met n=1). Een tweede maakt gebruik van tf-idf vectorisatie, met als vocabular alle individuele woorden. De derde vectorisatie maakte ook gebruik van tf-idf vectorisatie, maar nam als vocabulary word-level n-grams met n gaande van 2 tot 3. Tenslotte werd voor de laatste weer gebruik gemaakt van n-grams, maar in dit geval werd gekozen voor character-level n-grams. Weer met n gaande van 2 tot 3. Door het implementeren van alle mogelijke vectorvoorstellingen van de commentaren kon snel nagegaan worden welke implementatie voor dit geval het beste resultaat gaf. De vergelijk hiervan is te vinden in volgend hoofdstuk.

5.2.2 Cross Validation

Tenslotte wordt de gevectoriseerde data doorgegeven aan het Naive Bayes model. Om een goed beeld te krijgen van de accuraatheid van dit model wordt er gebruik gemaakt van cross validation. Figuur 5.5 illustreert de werking van 10 fold cross validation. De data wordt in 10 verschillende delen gedeeld, waarna elke van deze 10 delen eens gebruikt zal worden als testdata, en de overige 9 als trainingsdata. Voor elk van deze iteraties zal vervolgens een accuraatheid berekent worden via de testdata. Vervolgens kan de gemiddelde accuraatheid van het model achterhaald worden door het gemiddelde te nemen van de accuraatheden van de verschillende modellen.



Figuur 5.5: Werking 10 fold cross validation.

Zo werd uiteindelijk een Multinomial Naive Bayes model tienkeer getraind met verschillende training en testdata om een goed beeld te krijgen van de accuraatheid van dit model voor de routes-dataset.

5.2.3 Labels

In deze studie werd de data manueel gelabeled. Deze labels gingen van -2 tot 2. Voor het Multinomial Naive Bayes model werden deze scores echter vereenvoudigd tot enkel -1, 0 en 1, om te zorgen dat er van elke score genoeg voorbeelden beschikbaar. Zo waren er slechts een honderdtal commentaren die een score hadden van -2, wat zeer weinig is voor een Multinomial Naive Bayes model om hierop te kunnen trainen en testen. Met slechts drie mogelijke klassen is er minder data nodig om voor alle drie klassen, positief, neutraal en negatief, genoeg data te hebben.

5.2.4 Realisatie

Om dit model te creëren werd gebruik gemaakt van Scikit-Learn. Dit is een zeer populaire bibliotheek voor Python waarvan reeds hevig gebruik wordt gemaakt om aan machine learning te doen. Scikit-Learn voorziet ook goede implementaties van verschillende machine learning technieken, waaronder ook Multinomial Naive Bayes. Verder zijn ook de verschillende vectorisatiemethodes alsook de 10-fold cross validation gerealiseerd via deze bibliotheek.

5.2.5 Besluit

Via Scikit-Learn is het zeer makkelijk om aan machine learning te doen. Deze bibliotheek voorziet namelijk implementaties voor de meeste technieken gebruikt voor machinelearning. Er is dan ook een Multinomial Naive Bayes model opgesteld gebruik makende van Scikit-Learn. Vervolgens werd er een 10-fold cross validation op uitgevoerd om zo een goed beeld te krijgen van de gemiddelde accuraatheid van het model.

Hoofdstuk 6

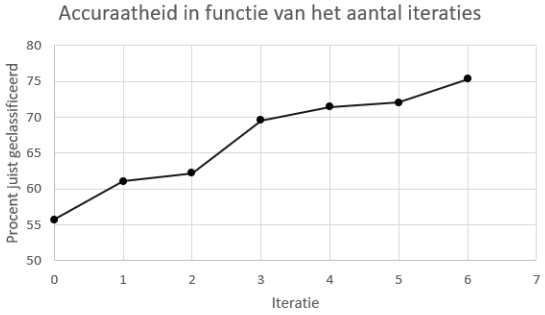
Resultaten

Dit hoofdstuk beschrijft de resultaten van de verschillende modellen. Er wordt gestart met model-speficieke resultaten. Hieronder vallen onder andere de evaluatiefunctie van het Rule Based model, alsook de resultaten van de verschillende vectorisatiemethodes van het Multinomial Naive Bayes model. Vervolgens wordt ook nagegaan wat de invloed is op het sentiment van een tekst wanneer deze vertaald wordt. Tot slot worden de twee modellen met elkaar vergeleken via variantieanalyse, waarbij wordt nagegaan of de resultaten van deze modellen voldoende van elkaar verschillen.

6.1 Het Rule Based Learning model

6.1.1 Evolutie van de accuraatheid

In figuur 6.1 is de evolutie van de accuraatheid van het Rule Based model in functie van het aantal iteraties weergegeven. Elke iteratie verschilt van zijn voorganger door het toevoegen van nieuwe regels, het uitbreiden van het lexicon, het wijzigen van regels, het toevoegen van vertalingen,... Er is te zien dat het model startte met een accuraatheid van iets meer dan 55%, en uiteindelijk is geëvolueerd naar een model met accuraatheid van 75,33%.

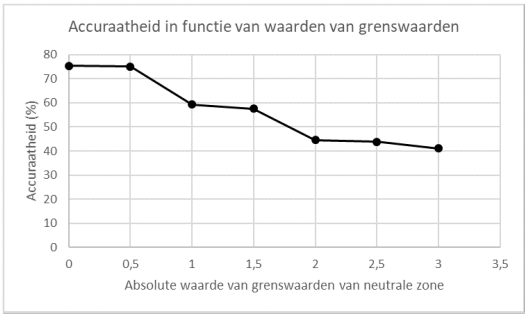


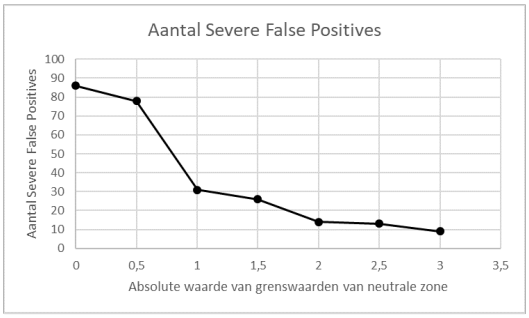
Figuur 6.1: Evolutie van de accuraatheid van het Rule Based model in functie van het aantal iteraties.

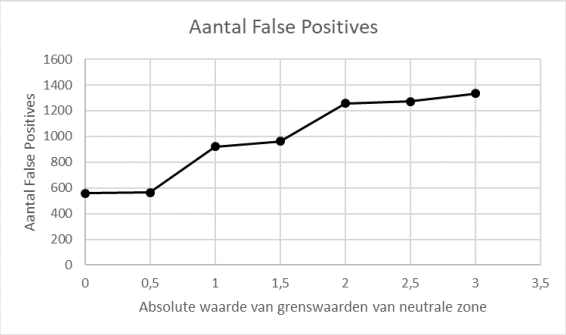
6.1.2 De Evaluatiefunctie

De invloed van de evaluatiefunctie op de resultaten van het Rule Based Learning model bleek zeer groot. In wat volgt zal er onderscheid gemaakt worden tussen gewone false positives, waarbij een commentaar geclassificeerd wordt in een verkeerde klasse, en severe false positives. Severe false positives komen voor wanneer een positieve commentaar als negatief wordt gezien, of als een negatieve commentaar als positief wordt gezien. Deze zijn een ergere misclassificatie dan wanneer er verkeerd geclassificeerd wordt tussen positief/negatief en neutraal. De waardes van de evaluatiefunctie bepalen hoeveel false positives er voorkomen, wat de totale accuraatheid is van het model, en hoeveel severe false positives er voorkomen. Naargelang de priortiteit kan er gekozen worden voor het minimaliseren van de severe false positives of de totale accuraatheid van het model.

In Figuur 6.2 zijn de resultaten van het rule based model terug te vinden voor verschillend grenswaarden van de evaluatiefunctie. Hierop is te zien dat de totale accuraatheid van het model daalt naargelang de neutrale zone groeit. Dit viel te verwachten, aangezien commentaren steeds duidelijker positief of negatief moeten zijn vooraleer herkend te worden als positief of negatief. Commentaren zullen dus al rapper gezien worden als neutraal. Doordat commentaren steeds extremer dienen te zijn zal het aantal negatief of positieve commentaren die respectievelijk als positief of negatief worden gezien ook verminderen. Dit is te zien in de daling van het aantal severe false positives.
Wanneer de neutrale zone groeit, neemt ook het aantal false positives toe omdat er steeds meer commentaren valselijk als neutraal zullen worden gelabeled.







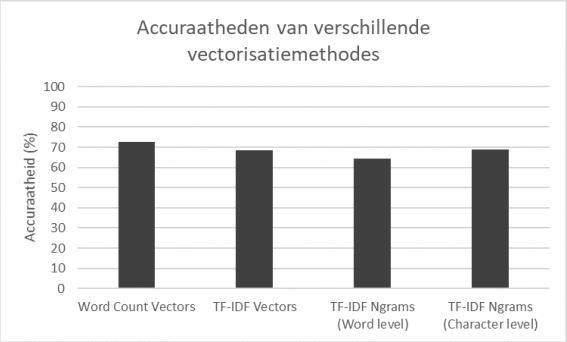
Figuur 6.2: Resultaten voor verschillende grenswaarden van de evaluatiefunctie.

Een grotere neutrale zone brengt dus een stijging van het aantal false positives met zich mee, alsook een dalende totale accuraatheid. Wel zorgt de grotere neutrale zone ervoor dat het aantal severe false positives daalt. Hierdoor kan men zekerder zijn dat wanneer een commentaar als positief of negatief wordt gelabeld, deze ook effectief respectievelijk positief of negatief is. Aangezien de twee modellen in deze studie vergeleken worden op basis van hun accuraatheid, werd er gekozen om te werken met een neutrale zone die enkel uit de waarde 0 bestaat. Deze zone leverde uiteindelijk de beste accuraatheid op. Hieruit blijkt dat wanneer een commentaar neutraal is, ze wellicht geen gevoelsgeladen woorden met zich meedraagt, waardoor de score op 0 blijft staan. De grenswaarden vergroten zorgt ervoor dat licht positieve/negatieve commentaren valselijk als neutraal worden geclassificeerd.

6.2 Multinomial Naive Bayes

6.2.1 De vectorisatiemethodes

In wat volgt worden de verschillende resultaten voor alle mogelijke vectorisaties beschreven. Er is voor elke vectorisatiemethode een 10-fold cross validation uitgevoerd, waarna er een gemiddelde accuraatheid alsook een standaardafwijking op deze accuraatheid werd bekomen. In figuur 6.3 zijn de resultaten hiervan weergegeven. Hierop is te zien dat het werken via *word count vectors* de beste accuraatheid oplevert, terwijl het werken via *word level n-grams* resulteert in een mindere accuraatheid. Aangezien voor het vergelijken van het Rule Based model en het Multinomial Naive Bayes model er een vergelijking gemaakt zal worden tussen de twee beste versies van deze modellen, zal er voor deze vergelijking gewerkt worden met een Multinomial Naive Bayes model werkend met count vectors.

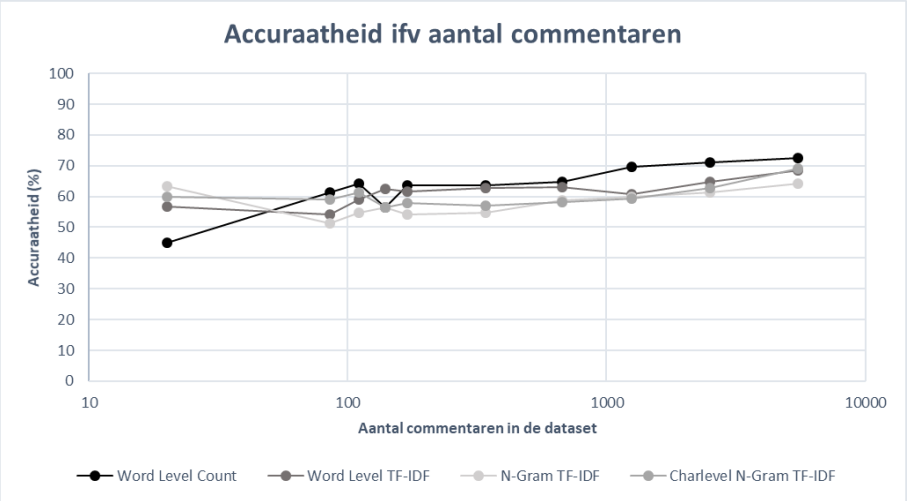


Figuur 6.3: Resultaten voor accuraatheden van de verschillende vectorisatiemethodes.

6.2.2 Evolutie van de accuraatheid

Naive Bayes is een machineleertechniek waarvan de performantie afhangt van hoeveel trainingsdata er beschikbaar is. Het model kan reeds relatief goed presteren met weinig trainingsdata in vergelijking met andere machineleertechnieken, maar ook hier zal de accuraatheid stijgen naarmate er meer trainingsdata beschikbaar is. In Figuur 6.4 zijn de verschillende accuraatheden uitgezet voor elke vectorisatiemethode, en voor verschillende groottes van beschikbare data. Deze accuraatheden zijn uitgezet op een logaritmische schaal. Er bleek namelijk een logaritmisch verband te zijn tussen de accuraatheid van dit model en het aantal beschikbare data. Aangezien er voor elk van de vier vectorisatiemethodes een stijgend verband te zien is kan aangenomen worden dat de accuraatheid van een Naive Bayes model logaritmisch toeneemt in functie van het aantal beschikbare trainings- en testdata.

Verder valt ook op te merken dat de accuraatheden bekomen voor zeer kleine datasets ($<$340) onbetrouwbaar zijn. Hierop zit namelijk een zeer grote standaardafwijking ($>$20%). Dit is wellicht te wijten aan de kleine testset, die ervoor zorgt dat de geschatte accuraatheid ofwel erg hoog ligt, ofwel erg laag. Dit is wellicht ook de reden waarom de resultaten voor de kleine datasets zeer veel schommelen. Eenmaal de dataset toeneemt in grootte is te zien dat de schommelingen in accuraatheid stabiliseren, de accuraatheden stijgen voor latere datasets lineair op de logaritmische schaal.

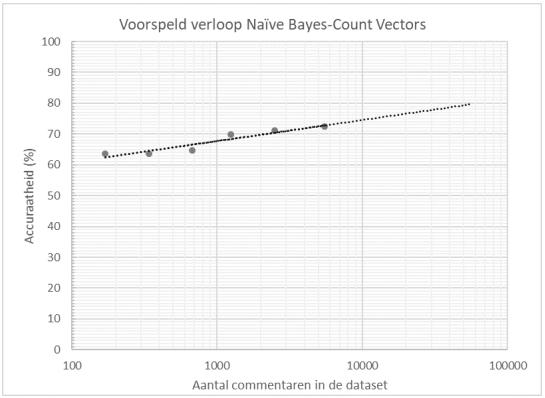


Figuur 6.4: Accuraatheden ifv aantal commentaren voor elke vectorisatiemethode.

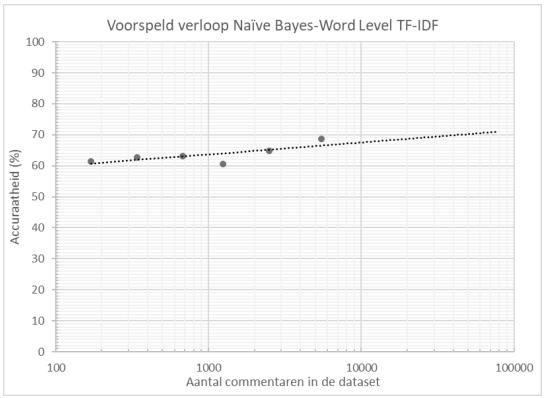
Om een beeld te krijgen van de evolutie van de accuraatheid van dit model wanneer het aantal commentaren toeneemt, werden de accuraatheden uitgezet op een logaritmische schaal, waarna er een trendlijn door getekend kon worden. Deze trendlijn kan ons een goede schatting geven van het verloop van de accuraatheid. In Figuur 6.5 is het voorspeld verloop van de accuraatheid van een Multinomial Naive Bayes model op basis van de vier vectorisatiemethoden weergegeven. Hierop is te zien dat voor het behalen van een accuraatheid van meer dan 75% er waarschijnlijk een dataset met grootte van 10.000 commentaren nodig zal zijn indien gewerkt wordt met *Word Count Vectors*. Aangezien we een logaritmisch verband vinden tussen de accuraatheid en het aantal commentaren, zal het aantal commentaren nodig om 100% te halen enorm groot zijn. Doordat we slechts 72% halen is een goede schatting van het benodigde aantal commentaren zeer moeilijk uit te voeren.

Deze trendlijn kan verder ook gebruikt worden om na te gaan wanneer het interessant zou worden om over te schakelen naar een Naive Bayes model. Indien het voorgestelde Rule Based model gemiddeld een accuraatheid van 75% haalt zou er gekozen kunnen worden over te schakelen op een Naive Bayes model wanneer de dataset is gegroeid tot een grootte van 10.000 commentaren, aangezien er aangenomen kan worden dat bij 10.000 commentaren de accuraatheid van het Multinomial Naive Bayes model groter of gelijk zal zijn aan 75%. Uiteraard is het aangeraden om dit eerst na te gaan vooraleer definitief over te schakelen. De trendlijn is uiteindelijk slechts een manier om het verloop van de accuraatheid te schatten. Verder valt ook op te merken dat er met een grotere dataset ook andere machineleertechnieken bruikbaar worden, die bij grotere datasets beter presteren dan Naive Bayes (bijvoorbeeld SVM). Het is dus aangeraden om bij een grotere dataset na te gaan of het nog zinvol is om Naive Bayes te gebruiken in plaats van een andere machineleertechniek zoals SVM.

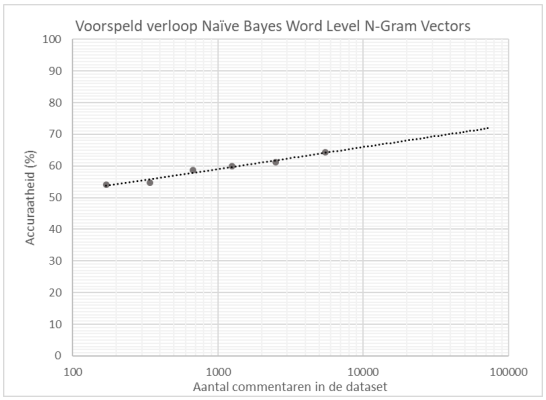
Verder werden er ook trendlijnen opgesteld voor word level-idf vectorisatie, n-gram tf-idf vectorisatie en character level n-gram tf-idf vectorisatie. Er werd reeds in dit hoofdstuk aangegeven dat



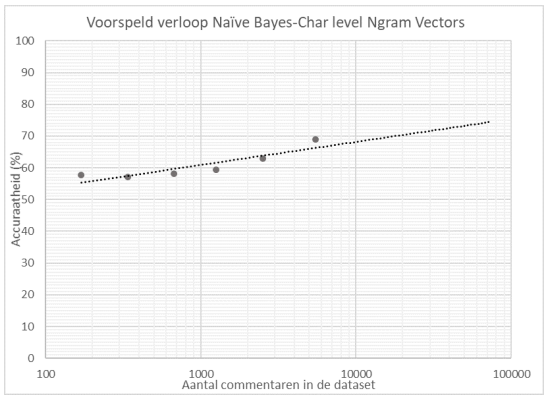
(a) Word Count Vectorisatie



(b) TF-IDF Vectorisatie



(c) Word Level N-gram Vectorisatie



(d) Character Level N-gram Vectorisatie

Figuur 6.5: Voorspeld verloop van de accuraatheid van Multinomial Naive Bayes voor de vier vectorisatie-methodes

deze vectorisatiemethoden minder goede resultaten opleverden dan de word count vectorisatie, en ook in de trendlijnen was dit te zien. Zo bleken er voor de eerste twee methodes meer dan 100.000 commentaren nodig te zijn om aan een accuraatheid van 75% te geraken, en voor de character level n-gram tf-idf vectorisatie bleken er ook nog 70.000+ commentaren nodig te zijn. Deze trendlijnen bleken dus oninteressant en zijn dus ook niet opgenomen in deze scriptie.

6.3 Variantieanalyse

Variantieanalyse (ook wel Anova genoemd) is een techniek uit de statistiek die gebruikt wordt om na te gaan of twee of meer groepen van elkaar verschillen. De gemiddelden van de afzonderlijke groepen worden vergeleken met het gemiddelde van alle groepen bij elkaar[1]. Als nulhypothese wordt gesteld dat de groepen gelijk zijn aan elkaar. Via de variantieanalyse kunnen hier vervolgens de p-waarde (overschrijdingskans) en de F-waarde van berekend worden. Bij een p-waarde $<$1% kunnen we spreken van een significant verschil. Ook indien de F-waarde groter is dan de kritische F-waarde kan er gesproken worden van een significant verschil[2]. Er bestaan zeer veel verschillende softwarepakketten waarmee deze toets zeer eenvoudig uitgevoerd kan worden. In deze studie is het Analysis Toolpak beschikbaar in Excel gebruikt om de variantieanalyse uit te voeren. Anova kan ook gebruikt worden om verschillende machine learning modellen met elkaar te vergelijken[3].

6.4 Anderstalige commentaren

In deze studie werd er vanuit gegaan dat het vertalen van anderstalige commentaren het sentiment van deze commentaren behoudt. Hierdoor zou het mogelijk zijn om een Rule Based Learning model te creëren waarbij de regels en het lexicon gebaseerd zijn op de Nederlandse taal. Verder hoeft het Naive Bayes model ook enkel vectors bij te houden waarbij gewerkt wordt met Nederlandstalige woorden.

Indien het vertalen van commentaren het sentiment niet zou bewaren, moet hiermee rekening gehouden worden. In het geval van het Rule Based model zou er dan ofwel voor elke taal een aparte module voorzien moeten worden, ofwel zouden de regels veralgemeend moeten worden om te kunnen werken voor verschillende talen. Zo wordt in dit model geen rekening gehouden met het feit dat in de Franse taal negatief uit twee delen kunnen bestaan. In het geval van het Naive Bayes model zou er ofwel een verschillend model per taal voorzien moeten worden, ofwel een algemeen model waarbij de vectoren alle woorden van alle talen bijhouden, wat zou leiden tot zeer grote vectoren.

[1] https://wetenschap.infonu.nl/wiskunde/162735-de-anova-of-variantie-analyse.html

[2] Zie theorema van Fisher: https://www.moaweb.nl/marktonderzoek-woordenboek/item/toetstheorema-van-fisher.html

[3] https://towardsdatascience.com/comparing-machine-learning-models-statistical-vs-practical-significance-de

Om na te gaan of er een verschil is tussen Nederlandstalige commentaren en vertaalde commentaren worden twee datasets door zowel het Naive Bayes model als het Rule Based model gestuurd. Beide modellen voeren vervolgens een 10-fold cross validation uit, waarna we een gemiddelde accuraatheid en standaardafwijking bekomen. De resultaten voor het Multinomial Naive Bayes model liggen hiervoor extreem laag. Dit ligt grotendeels aan het feit dat voor het uitvoeren van deze vergelijking de totale dataset verdeeld werd in Nederlandstalige commentaren en anderstalige commentaren. De dataset met anderstalige commentaren bestaat uit 1459 teksten. Om een eerlijke vergelijking te voorzien werden zowel de grootte van de dataset Nederlandse als die van anderstalige commentaren gelijk gesteld aan 1459. Dit is zeer weinig voor een model om te gebruiken als trainings- en testdata. Een goede vergelijking is dus moeilijk te maken, aangezien het gebrek aan voldoende anderstalige commentaren de accuraatheid van het Multinomial Naive Bayes model te sterk beïnvloedt.

Om een goede vergelijking te kunnen maken tussen Nederlandstalige en niet-Nederlandstalige commentaren is een model waar de accuraatheid niet afhangt van het aantal commentaren dus interessanter. Dit is het geval voor Rule Based Learning model. In Tabel 6.1 vinden we de resultaten terug voor beide modellen. Hier merken we dat beide gemiddelden voor de vertaalde commentaren beduidend lager liggen dan de gemiddelden voor de Nederlandstalige commentaren. In het geval van het Rule Based Learning model ligt dit niet aan het verschil in grootte van de datasets (Nederlandstalige dataset bevat 4372 commentaren en anderstalige dataset bevat 1460 commentaren).

	Nederlands-RB	Vertaald-RB	Nederlands-NB	Vertaald-NB
Iteratie 1	0,7044	0,7037	0.5646	0,5102
Iteratie 2	0,8146	0,6949	0.5986	0,4521
Iteratie 3	0,7232	0,6780	0.5306	0,3699
Iteratie 4	0,7627	0,5763	0.6438	0,7740
Iteratie 5	0,8114	0,7083	0.6027	0,6712
Iteratie 6	0,8047	0,7647	0.6712	0,4658
Iteratie 7	0,7643	0,6780	0.5822	0,4178
Iteratie 8	0,7766	0,6667	0.5616	0,4315
Iteratie 9	0,75325	0,7500	0.6027	0,5068
Iteratie 10	0,8072	0,7222	0.6549	0,4931
Gemiddelde	0,7722	0,6943	0.6013	0,5092
Variantie	0,001462	0,002706	0.001789	0.01355

Tabel 6.1 Verschillende waarden voor de verschillende iteraties van de 10 fold cross validation uitgevoerd door beide modellen en voor beide datasets.

Om zeker te zijn dat er verschil is tussen Nederlandstalige en niet-Nederlandstalige commentaren werd ook een variantieanalyse uitgevoerd. Hiervoor werd eerst nagegaan of de reeks van ver-

schillende accuraatheden normaal verdeeld is. Dit is een vereist om aan variantieanalyse te doen. Hieruit bleek deze reeks van accuraatheden normaal verdeeld te zijn. Een uitgewerkt voorbeeld van hoe dit precies werd nagegaan komt later aan bod in dit hoofdstuk. In Figuur 6.6 zijn de resultaten van de variantieanalyse zichtbaar. Hierin kan men onder andere het gemiddelde alsook de variantie in terug vinden. Verder is te zien dat er een p-waarde is van 0,001255981. Hieruit kunnen we concluderen dat er slechts 0.1255918% kans is dat deze twee accuraatheden werkelijk dezelfde zouden zijn. Ook de F-waarde ligt boven de kritische F-waarde. De hypothese dat het sentiment behouden blijft wanneer commentaren vertaald worden kan dus veilig verworpen worden. De gemiddelde accuraatheden verschillen te veel.

SAMENVATTING

Groepen	*Aantal*	*Som*	*Gemiddelde*	*Variantie*
Nederlandse commentaren	10	7,722449887	0,772244989	0,001461577
Vertaalde commentaren	10	6,942750452	0,694275045	0,002705873

Variantieanalyse

Bron van variatie	*Kwadratensom*	*Vrijheidsgraden*	*Gemiddelde kwadraten*	*F*	*P-waarde*	*Kritische gebied van F-toets*
Tussen groepen	0,03039656	1	0,03039656	14,58760471	0,001255981	4,413873
Binnen groepen	0,037507055	18	0,002083725			
Totaal	0,067903615	19				

Figuur 6.6: Vergelijking tussen resultaten van Nederlandstalige en niet-Nederlandstalige commentaren.

6.5 Rule Based Learning vs Multinomial Naive Bayes

In deze studie werden twee verschillende modellen gerealiseerd. Een model via Rule Based Learning, en een model via Multinomial Naive Bayes. In deze sectie zullen deze 2 modellen met elkaar vergeleken worden. Er zal nagegaan worden of er een verschil is tussen de resultaten van deze twee modellen, en er zal dus eventueel aangegeven worden welk model het beste werkt voor de routes-dataset.

De resultaten van het Naive Bayes model werden bekomen via 10-fold cross validation, reeds besproken in Sectie 5.2.2. Er werd ook een gelijkaardige procedure doorlopen voor het Rule Based Learning model. Het verschil zit echter in het feit dat dit model niet meer getraind dient te worden via trainingsdata. Om aan tien verschillende accuraatheden te komen werd de totale dataset opgesplitst in tien verschillende delen, waarna voor elk deel een accuraatheid werd bekomen. Deze tien waarden vormen vervolgens de waarden te vinden in tabel 6.2. Hiervoor dient opgemerkt te worden dat de dataset voor beide modellen willekeurig in tien delen verdeeld werd. Deze verdeling is dus niet voor beide modellen dezelfde. Voor elke iteratie in tabel 6.2 werd gewerkt met verschillende subdatasets voor het Rule Based model en het Naive Bayes model. Doordat na tien iteraties voor beide modellen de gehele dataset is overlopen, zullen beide modellen uiteindelijk een gemiddelde uitkomen dat de gemiddelde accuraatheid geeft voor de gehele dataset, die dezelfde is voor zowel het Rule Based model als het Naive Bayes model.

Om de twee modellen te vergelijken zal gebruikt gemaakt worden van variantieanalyse. Om twee resultaten te vergelijken via variantieanalyse zijn er wel verschillende voorwaarden waaraan de resultaten moeten voldoen. Een vereiste is dat voor iedere groep de variabele een normale verdeling heeft. Vooraleer de twee modellen dus vergeleken kunnen worden dient er eerst nagegaan te worden of aan deze vereiste voldaan is.

6.5.1 Normale verdeling van de modellen

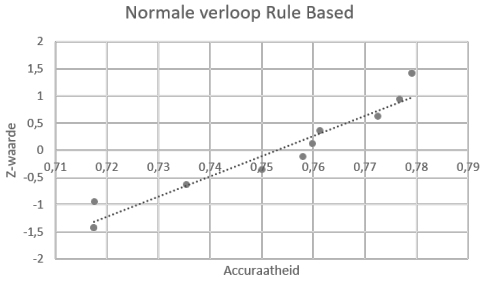
Een vuistregel van een normale verdeling is dat er gesteld wordt dat 68% van de waarnemingen binnen het gebied tussen $\mu \pm \sigma$ liggen. Verder liggen 95% van de waarnemingen tussen $\mu \pm 2\sigma$ en 99% tussen $\mu \pm 3\sigma$. In Tabel 6.2 zijn de resultaten weergegeven van de 10-fold cross validation voor beide modellen. Hierin vinden we ook het gemiddelde, alsook de variantie terug. Hierin zien we dat de vuistregels gelden voor het Rule Based model. Zo liggen 60% van de gevallen binnen $0.75279949 \pm \sqrt{0,00051106}$. Vervolgens liggen alle waarden zowel in $\mu \pm 2\sigma$ als binnen $\mu \pm 3\sigma$. Aangezien er slechts tien waarden zijn komen deze percentages dus enorm goed overeen met deze van een normale verdeling.
Voor het Multinomial Naive Bayes model vallen 70% van alle waarden binnen $0,72036999 \pm \sqrt{0,00027477}$. Vervolgens vallen alle gevallen zowel binnen $\mu \pm 2\sigma$ als $\mu \pm 3\sigma$.

	Rule Based	Multinomial Naive Bayes
Iteratie 1	0,735426009	0,72826087
Iteratie 2	0,717488789	0,71506352
Iteratie 3	0,772532189	0,68784029
Iteratie 4	0,757990868	0,72181818
Iteratie 5	0,717592593	0,72909091
Iteratie 6	0,776744186	0,72727273
Iteratie 7	0,761261261	0,71948998
Iteratie 8	0,779116466	0,69763206
Iteratie 9	0,750000000	0,74134791
Iteratie 10	0,75984252	0,73588342
Gemiddelde	0,75279949	0,72036999
Variantie	0,00051106	0,00027477

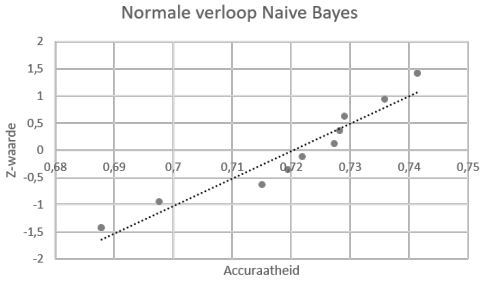
Tabel 6.2 Resultaten voor de 10 fold cross validation voor de routedataset voor beide modellen.

Aangezien beide groepen voldoen aan deze vuistregel, rekening houdend met het feit dat er slechts 10 waarden beschikbaar zijn, kan er dus vermoed worden dat deze inderdaad normaal verdeeld zijn.

Een andere manier om na te gaan of beide verdelingen normaal verdeeld zijn is door deze punten uit te zetten op een normaal-waarschijnlijkheidspapier. De schaalverdeling langs de verticale as is hierbij zodanig dat bij het uitzetten van kwantielen van een normale verdeling een rechte lijn ontstaat[4]. Dit is vervolgens ook gedaan voor de waarden van de beide modellen. Op de x-as vinden we de accuraatheid terug, op de y-as de Z-waarde. De Z-waarde geeft weer hoeveel standaardafwijkingen een bepaalde variabele X afwijkt van het gemiddelde[5]. Voor beide plots geldt dat de waarden zich in een rechte lijn bevinden. Dit wijst er dus op dat beide modellen een verband met normale verdeling vertonen.



Figuur 6.7: Normaal-waarschijnlijkheidsplot voor het Rule Based model



Figuur 6.8: Normaal-waarschijnlijkheidsplot voor het Naive Bayes model

Het aanwezig zijn van een normale verdeling voor beide groepen is nagegaan via verschillende methoden, die telkens aangaven dat deze groepen wellicht normaal verdeeld zijn. Als besluit kan er dus getrokken worden dat beide groepen een normale verdeling hebben en dus geschikt zijn om te vergelijken via variantieanalyse.

6.5.2 Vergelijking van het Rule Based Learning model ten opzichte van het Multinomial Naive Bayes model.

In wat volgt zullen het Rule Based Learning model en het Multinomial Naive Bayes model met elkaar vergeleken worden via dezelfde methode toegepast om Nederlandstalige en niet-Nederlandstalige commentaren te vergelijken, via variantieanalyse. In tabel 6.2 zijn de waarden terug te vinden waarmee gewerkt werd. Tot slot zijn in figuur 6.9 de resultaten terug te vinden van de variantieanalyse. Hierop is een p-waarde terug te vinden van 0,001798191. Dit komt erop neer dat er we met meer dan 99% zekerheid kunnen zeggen dat er wel degelijk een verschil zit tussen beide modellen.

Het Rule Based Learning model verschilt dus van het Multinomial Naive Bayes model. Waar het verschil precies zit, valt niet te zeggen via variantieanalyse. Wel weten we door de variantieanalyse dat het Rule Based Learning model met meer dan 99% zekerheid beter presteert dan het Multinomial Naive Bayes model voor de gegeven dataset.

[4] Bron: `https://nl.wikipedia.org/wiki/Normaal-waarschijnlijkheidspapier`
[5] https://statistics.laerd.com/statistical-guides/standard-score.php

SAMENVATTING

Groepen	*Aantal*	*Som*	*Gemiddelde*	*Variantie*
RB	10	7,52799488	0,752799488	0,000511057
NB	10	7,20369987	0,720369987	0,00027477

Variantie-analyse

Bron van variatie	*Kwadratensom*	*Vrijheidsgraden*	*Gemiddelde kwadraten*	*F*	*P-waarde*	*Kritische gebied van F-toets*
Tussen groepen	0,005258363	1	0,005258363	13,38299659	0,001798191	4,413873
Binnen groepen	0,007072447	18	0,000392914			
Totaal	0,012330809	19				

Figuur 6.9: Variantieanalyse tussen het Rule Based Model en het Multinomial Naive Bayes model.

Het is aan te raden om het Rule Based Learning model verder te gebruiken om aan Sentiment Analysis te doen, aangezien deze beter presteert op deze data. Dit is voornamelijk het geval omdat er slechts een kleine hoeveelheid trainingsdata beschikbaar is, en technieken zoals Naive Bayes, die trainen aan de hand van deze trainigsdata, minder goed presteren naarmate er minder data beschikbaar is.

Een ander voordeel van het Rule Based Learning model is het feit dat deze over een evaluatiefunctie beschikt. Dit zorgt ervoor dat het aantal severe false positives gecontroleerd kan worden door de grenswaarden van de evaluatiefunctie aan te passen. Het Multinomial Naive Bayes model beschikt niet over zo'n evaluatiefunctie, omdat de scores die door dit model gegenereerd worden slechts gelijk kunnen zijn aan scores waarmee de commentaren manueel gelabeled werden, namelijk in het geval van MNB -1,0 en 1.

6.6 Besluit

Beide modellen zijn vergeleken via variantieanalyse. Het Rule Based Learning model bleek veel te verschillen van het Multinomial Naive Bayes model. Er kan met 99% zekerheid gezegd kan worden dat de resultaten van deze modellen van elkaar verschillen. Het Rule Based Learning model is met een gemiddelde accuraatheid van 75,28% het betere van de twee modellen. Verder is dit model ook in staat door via de grenswaarden van zijn neutrale zone de hoeveelheid severe false positives te verminderen, al moet hierbij vermeld worden dat dit ten koste gaat van de accuraatheid.

Verder bleek de accuraatheid van het Multinomial Naive Bayes model toe te nemen naarmate er meer trainingsdata beschikbaar is. Het zou dus relevant zijn de twee modellen opnieuw te vergelijken wanneer de grootte van de dataset sterk is toegenomen.

Tot slot blijkt er ook een verschil te zijn tussen Nederlandse commentaren en niet-Nederlandse commentaren die vertaald werden naar het Nederlands. De hypothese dat het sentiment helemaal behouden blijft bij vertaalde commentaren kan dus verworpen worden. Het sentiment verzwakt wanneer commentaren automatisch vertaald worden, wellicht omdat er nog steeds foute vertaling gebeuren, waardoor de betekenis van de oorspronkelijke zin deels verloren gaat. Een andere

mogelijke verklaring is het feit dat andere talen, zoals het Frans (waar adjectieven vaak na een zelfstandig naamwoord worden gezet), een andere zinsbouw hebben, waar het Rule Based Learning model niet op is ingesteld, aangezien dit model is opgesteld voor zinnen met ook een Nederlandse zinsbouw. Er wordt bijvoorbeeld aangenomen dat bijwoorden altijd voor adjectieven komen.

Om de niet-Nederlandse commentaren met een even goede accuraatheid te kunnen classificeren als de Nederlandse commentaren is het dus aangewezen om hiervoor een aparte module te voorzien in het Rule Based Learning model. Deze module zou kunnen bestaan uit een lexicon met taal-specifieke woorden, en aparte taal-specifieke regels.

Hoofdstuk 7

Conclusie

In deze studie is onderzocht hoe er aan Sentiment Analysis gedaan kan worden, en concreet welke technieken hier het meest voor in aanmerking kwamen. Er werden twee verschillende technieken gebruikt, namelijk Rule Based Learning en Multinomial Naive Bayes. Vervolgens werd via beide methodes een model opgesteld, en werd er aangetoond dat beide modellen in staat zijn om een accuraatheid van meer dan 70% te behalen voor het classificeren van commentaren in de juiste gevoelsklasse. Beide technieken presteren goed wanneer er slechts een kleine dataset beschikbaar is en kunnen dus effectief gebruikt worden om commentaren over routes automatisch te beoordelen.

Van de twee modellen bleek het Rule Based Learning model het effectiefst te zijn. Deze behaalde een duidelijk betere accuraatheid dan het Multinomial Naive Bayes model. Wel bleek uit verder onderzoek dat naarmate het aantal commentaren toeneemt, de accuraatheid van het Multinomial Naive Bayes model ook zal toenemen, en na verloop van tijd de accuraatheid van het Rule Based Learning model waarschijnlijk zal overschrijden. Verder bleek dit model het best te werken wanneer er gewerkt werd met Word Count vectorisatie.

Verder werd ook onderzocht of er voor het verwerken van anderstalige commentaren een apart model opgesteld dient te worden. Hieruit bleek dat wanneer een commentaar vertaald wordt naar het Nederlands het sentiment van deze commentaar niet volledig behouden blijft. Er kan dus geconcludeerd worden dat voor het beoordelen van de anderstalige commentaren er een apart model gerealiseerd dient te worden. Het vertalen van de anderstalige commentaren naar het Nederlands biedt geen perfecte oplossing.

Het Rule Based model kan dus succesvol gebruikt worden om commentaren automatisch te beoordelen, om zo de betere routes te laten opvallen. Wel zal er rekening gehouden moeten worden met de lagere accuraatheid wanneer anderstalige commentaren via dit model beoordeeld worden. Hiervoor biedt dit onderzoek geen goede oplossing.

7.1 Toekomstig werk

Toekomstig werk kan zich toespitsen op verschillende zaken. Zo kan er enerzijds nagegaan worden hoe anderstalige commentaren goed verwerkt kunnen worden, aangezien het vertalen van deze commentaren geen oplossing biedt. Dit kan eventueel gedaan worden door het aanmaken van een aparte module voor elke taal, of door het veralgemenen van het model zodat dit kan werken voor eender welke taal.

Verder kan ook het Rule Based Learning model nog verbeterd worden. Dit door het lexicon hiervan uit te breiden met alle mogelijke woorden die in de context van commentaren op routes eventueel een sentiment zouden kunnen meedragen. Ook kan er met behulp van een taalspecialist opnieuw naar de toegekende gewichten aan de woorden in het lexicon gekeken worden, om deze eventueel nog bij te stellen.

Een andere interessante studie die aansluit bij het beoordelen van commentaren is het automatisch bepalen van het onderwerp van de commentaar. Door na te gaan waarover de commentaar gaat, kan men dit onderwerp samen met de sentimentscore van de commentaar gebruiken om de kwaliteit van de commentaar nog beter in te schatten. Wanneer er bijvoorbeeld negatief wordt gesproken over het weer dient de schatting niet te veranderen. Wanneer er negatief gesproken wordt over de weg, dient de schatting wel te veranderen. Het nagaan van het onderwerp zou dus een positieve bijdrage kunnen leveren aan het automatisch beoordelen van de kwaliteit van een route.

Bibliografie

[Rap] Rapidminer. https://rapidminer.com/. Accessed: 2019-04-09.

[2] Ahmed Sulaiman M. Alharbi, E. D. (2017). Enhance a deep neural network model for twitter sentiment analysis by incorporating user behavioral information.

[3] Fazal Masud Kundi, Aurangzeb Khan, S. A. M. Z. A. (2014). Lexicon-based sentiment analysis in the social web. *Journal of Basic and Applied Scientific Research*.

[4] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features.

[5] K. Chidambarathanu, K. L. S. (2017). Predicting user preferences on changing trends and innovations using svm based sentiment analysis. *Cluster Computing*.

[6] Kim, Y. (2014). Convolutional neural networks for sentence classification.

[7] Liu, B. (2012). Sentiment analysis and opinion mining.

[8] Muhammad Zubair Asghar, Aurangzeb Khan, S. A. M. Q. I. A. K. (2017). Lexicon-enhanced sentiment analysis framework using rule-based classification scheme. *PLoS ONE*.

[9] Rani, S. (2014). Rule based sentiment analysis system. Master's thesis, Computer science and engineering department thapar university.

[10] Sholom M. Weiss, N. I. (1995). Rule-based machine learning methods for functional prediction. *Journal of Articial Intelligence Research 3 (1995) 383-403*.

[11] Thársis Salathiel de Souza Viana, Marcos de Oliveira, T. L. C. d. S. M. S. R. F. a. . E. J. T. G. (2018). A message classifier based on multinomial naive bayes for online social contexts. *Journal of Management Analytics*.

Bijlage A

Adjectieven en Bijwoorden aanwezig in het Rule Based model

In onderstaande tabellen zijn alle aanwezige adjectieven en bijwoorden terug te vinden die aanwezig zijn in het rule based model. Hiernaast is ook telkens het meegegeven gewicht opgelijst.

A.1 Adjectieven

Adjectief	Gewicht
Goed	1
Nice	1
Betrouwbaar	1
Bruikbaar	1
Degelijk	1
Tevredenstellend	1
Juist	1
Correct	1
Prima	1
Interessant	1
Top	1
Plezant	1
Tof	1
Dank	1
Danku	1
Dankje	1
Bedankt	1
Merci	1
Thanks	1
Fijn	1
Aangenaam	1
Behaaglijk	1
Gezellig	1
Geschikt	1
Adequaat	1
Gelegen	1
Passend	1
Toegankelijk	1
Bereikbaar	1
Openbaar	1
Mooi	1
Bevallig	1
Fraai	1
Knap	1
Schoon	1
Aantrekkelijk	1
Aardig	1
Fietspad	1
Rustig	1
Merkwaardig	1

Adjectief	Gewicht
Bijzonder	1
Curieus	1
Notabel	1
Wonderlijk	1
Schitterend	2
Blinkend	2
Briljant	2
Glansrijk	2
Stralend	2
Grandioos	2
Briljant	2
Magnifiek	2
Subliem	2
Oogverblindend	2
Fascinerend	2
Fantastisch	2
Buitengewoon	2
Ongelooflijk	2
Prachtig	2
Toverachtig	2
Geweldig	2
Genot	2
Heerlijkheid	2
Plezier	2
Vreugde	2
Wellust	2
Leuk	2
Perfect	2
Ideaal	2
Uitmuntend	2
Uitstekend	2
Best	2
Excellent	2
Parel	2
Spectaculair	2
Adembenemend	2
Sensationeel	2
Opzienbarend	2
Fenomenaal	2
Aanrader	2

Adjectief	Gewicht
Heerlijk	2
Saai	-1
Doods	-1
Doorsnee	-1
Duf	-1
Eentonig	-1
Oninteressant	-1
Slaapverwekkend	-1
Vervelend	-1
Privé	-1
Privéterrein	-1
Privédomein	-1
Verboden	-1
Helaas	-1
Jammer	-1
Ongelukkigerwijs	-1
Onmogelijk	-1
Ondoenbaar	-1
Onberijdbaar	-1
Afwijken	-1
Verschillen	-1
Herbekijken	-1
Aanpassen	-1
Herevalueren	-1
Herbekeken	-1
Vervanging	-1
Vergoeding	-1
Vernieuwing	-1
Snelweg	-1
Verdwijn	-1
Lelijk	-1
Druk	-1
Fout	-1
Verkeerd	-1
Incorrect	-1
Onaanvaardbaar	-1
Slecht	-1
Gebrekkig	-1
Inferieur	-1
Onvoldoende	-1

Adjectief	Gewicht
Onplezierig	-1
Zwak	-1
Dom	-1
Stom	-1
Kut	-2
Shit	-2
Klote	-2
Verdomme	-2
Teleurstelling	-2
Afknapper	-2
Anticlimax	-2
Domper	-2
Flop	-2
Ontgoocheling	-2
Tegenvaller	-2
Teleurstellende	-2
Tijdsverspilling	-2
Weghalen	-2
Ramp	-2
Drama	-2
Tragedie	-2
Schande	-2
Afrader	-2
Doodlopen	-2
Ok	0,5
Oké	0,5
Verwarrend	-0,5

A.2 Bijwoorden

Bijwoord	Gewicht
Enorm	1
Super	1
Reuze	1
Uiterst	1
Ultra	1
Helemaal	0,7
Volkomen	0,7
Totaal	0,7
Zeer	0,5
Extra	0,5
Veel	0,5
Erg	0,5
Heel	0,5
Zo	0
Beetje	-0,25
Enigszins	-0,25
Ietwat	-0,25
Tikkeltje	-0,25
Tikje	-0,25
Weinig	-0,5
Amper	-0,5
Gering	-0,5
Minder	-0,5
Zelden	-0,8
Niet	-2
Geen	-2

Bijlage B

Top 200 meest voorkomende woorden

B.1 Niet gestemde woorden

Term	Aantal keer voorgekomen
route	5714
niet	2159
jouw	1637
criteria	1491
gt	1405
p	1333
voldoet	1106
rolstoelvriendelijk	1103
we	1038
â	1038
km	866
mooie	821
score	798
routeyou	673
dank	658
commentaar	593
wij	592
prachtige	586
indien	586
verwijderen	584
hartelijk	568
drie	563
gemarkeerd	556
ooh	556
bevestigen	555
bijdrage	555
reageren	555
connect	554
community	552
kenmerk	552

Term	Aantal keer voorgekomen
wandeling	542
gebruiken	511
s	493
wel	463
lees	460
langs	455
maken	445
mee	436
beter	434
geen	419
aangepast	411
2016	401
weg	396
vind	394
tips	392
kreeg	387
link	371
zeer	351
2	336
wegen	334
1	312
min	310
weer	295
5	291
goed	273
mooi	271
harer	269
heel	263
3	255
tijd	250

Term	Aantal keer voorgekomen
waar	247
gereden	242
4	238
routes	236
Ã£	236
bedankt	235
echt	232
gedaan	229
zeker	228
aanrader	226
leuke	216
privÃ£	204
mag	201
via	197
verder	197
rijden	194
rit	187
totale	186
staat	182
terug	178
be	173
tussen	173
stuk	171
deel	169
6	168
lopen	162
reis	161
8	159
gaan	156
7	154
fietsen	150
uur	148
erg	147
gaat	146
afstand	145
pieterpad	142
eer	140
groeten	139
richting	139
dag	138
gps	137
tocht	136

Term	Aantal keer voorgekomen
bos	135
onze	135
naam	134
pad	134
volgt	131
info	127
middelengels	127
leuk	127
kregen	124
zie	123
vandaag	123
af	122
enkele	122
ongeveer	122
gelopen	122
9	121
steeds	121
alleen	120
gebruikt	120
n	119
rivier	119
loopt	118
hallo	117
vanaf	117
fiets	116
e	116
volgen	116
gemiddeld	116
10	115
laat	115
gegevens	115
stukken	114
c	114
moving	114
even	114
grote	113
vriendelijke	113
melding	113
rond	111
goede	109
snelheid	109
gestopt	109

Term	Aantal keer voorgekomen
overall	107
kleine	105
ter	103
staan	102
maximale	102
beschrijving	102
komt	101
wind	101
paden	100
augustus	100
weinig	99
geschikt	98
meest	97
18	96
natuur	96
11	95
amp	95
plaatsen	95
plezier	95
plaats	95
platform	95
juli	94
zoals	94
nivon	94
stichting	94
œpieterpad â	94
keer	93
merknaam	92

Term	Aantal keer voorgekomen
water	91
wandelen	91
30	91
bee	91
loop	90
auto	90
racefiets	90
2018	90
vanuit	89
ga	89
stukje	88
lijkt	88
track	88
vooral	88
genoten	87
alle	86
verkeer	86
twee	86
probleem	86
moeten	85
mtb	85
groot	84
buurt	84
daarom	84
maart	84
parcours	82
publiek	82

B.2 Gestemde woorden

Term	Aantal keer voorgekomen
rout	5715
niet	2159
jouw	1637
criteria	1491
gt	1405
voldoet	1106
rolstoelvriend	1104
mooi	1092
we	1039
km	866
scor	798
weg	744
wandel	687
routeyou	673
prachtig	667
dank	659
commentar	595
wij	592
gebruik	590
verwijder	587
indien	586
hartelijk	569
lang	563
drie	563
reager	558
bevest	557
gemarkeerd	556
ooh	556
bijdrag	555
kenmerk	554
connect	554
community	552
kreg	511
les	486
vind	465
wel	465
mak	458
beter	448
link	437
me	436
gen	422
tip	418

Term	Aantal keer voorgekomen
aangepast	411
2016	401
goed	383
zer	352
leuk	343
hel	326
min	310
wer	295
stuk	286
echt	282
del	276
harer	269
be	264
war	254
lop	253
tijd	253
gered	243
routes	236
bedankt	235
pad	234
zeker	230
gedan	229
aanrader	226
total	226
rijd	208
verder	205
privÃ£	204
mag	201
grot	198
via	197
enkel	194
rit	193
groet	193
dag	190
stat	184
bos	181
terug	178
nam	177
tus	173
10	171
lat	167
gemiddeld	164

Term	Aantal keer voorgekomen
gan	164
problem	163
reis	161
klein	160
gat	156
natur	155
kom	150
fiets	150
uur	148
tocht	148
erg	147
afstand	146
richting	144
pieterpad	142
eer	140
gps	137
best	136
onz	136
rond	136
volg	135
wet	133
volgt	131
ker	130
overal	130
sted	129
plat	129
onverhard	128
info	127
middelengel	127
klim	125
gebruikt	125
rivier	124
zie	123
vandag	123
af	122
ongever	122
gelop	122
slecht	121
fiet	120
allen	120
ander	120
fietspad	118
vriendelijk	118
loopt	118

Term	Aantal keer voorgekomen
hallo	117
vanaf	117
technisch	117
gegeven	115
moving	114
even	114
melding	114
recht	113
mogelijk	112
snelheid	109
maximal	109
gestopt	109
beschrijv	109
zit	108
nem	107
verhard	105
ter	105
rustig	104
30	103
stan	102
wind	102
eind	102
komt	101
jar	101
weinig	100
track	100
augustus	100
geschikt	98
meest	97
punt	97
nieuw	97
18	96
zoal	96
plezier	96
1	95
brug	95
druk	95
amp	95
plaats	95
platform	95
al	94
juli	94
water	94
nivon	94

Term	Aantal keer voorgekomen
stichting	94
œpieterpadâ	94
parcour	93
merknam	92
begin	91
gewon	90
volled	90
genot	90
auto	90
lus	90
racefiet	90
2018	90
vanuit	89
ga	89

Term	Aantal keer voorgekomen
stukj	88
lijkt	88
vooral	88
ban	88
tof	87
verker	87
maart	87
twe	86
kort	86
moet	85
mtb	85
hen	84
buurt	84

Bijlage C

Extended abstract

Sentiment Analysis of User Commentaries on Scenic Routes*

Joran De Wilde

KU Leuven, Technologiecampus Gent, Belgium

Abstract. Many platforms give their users the opportunity to leave comments. Users post comments to inform other users about the quality of a product, or to give a company a review of their products. These companies can then use these comments to improve their products or services. Sentiment Analysis is the process by which it is investigated how sentiment and emotions can be automatically extracted from a text. In this study we work with a dataset of comments on scenic routes. We compare two approaches to Sentiment Analysis, using machine learning techniques that are applicable to small datasets: Rule Based Learning and Multinomial Naive Bayes. The performance of these models is evaluated and we find that the Rule Based Learning model can perform Sentiment Analysis with an accuracy of 75%, while the Multinomial Naive Bayes only gets an accuracy of 72%. By using variance analysis we find that the results from both these models differ from each other, and that the Rule Based Learning model is indeed the better of the two. Furthermore we investigate if all comments can be translated to a single language without losing their sentiment. The results in this study show that the accuracy drops significantly when Sentiment Analysis is performed on comments only existing of non-Dutch comments translated to Dutch.

Keywords: Sentiment Analysis · Naive Bayes · Rule Based Learning.

1 Introduction

In this case, users from all over the world can leave comments on cycle, walking and motorcycle routes. These comments are written especially in Dutch, but comments also appear in other languages like English and French. The comments are used to get info about the quality of the routes. This estimated quality can then be used to make the better routes stand out. Currently, rating these comments is done manually and takes a couple of hours each week. Via machine learning it is possible to automate the process of rating these comments. The dataset that will be used for this task is small in size, and the comments in this dataset are written in different languages, including Dutch, French and English. There are currently already different Sentiment Analysis softwares available. These softwares are however not written for the Dutch language and can't be used in this case. There is a need for software that is specialized in processing

* Supported by RouteYou

Dutch comments. In section 2, a brief literature review can be found. In the next section the methods to create both the models are described. This section is followed by the results of both these models. Finally, the last sections provide a discussion and the conclusion of this study.

2 Literature Review

Neural Networks have already shown to be excellent in processing texts and deduce sentiment from these texts [1, 2]. However, to train such a network a lot of data is required, which is not available in this study. Rule Based Systems (RBS) have already proven to produce good results when classifying English texts based on their expressed sentiment [3, 4]. Multinomial Naive Bayes (MNB) is also a technique used when classifying texts with good results [5]. Both these models don't need a lot of data to produce good results. In this work, a Rule Based Learning model is created, as well as a Multinomial Naive Bayes model. By comparing these models it is possible to check which of these models produces the best results. This study also investigates the effect of translating text on the sentiment of this text. This by comparing how well both models perform on a Dutch-only dataset versus a dataset consisting of only comments that had to be translated to Dutch.

3 Methods

In this section the two created models are described. For both these models, the components and their functions are explained. Before this, the acquisition of the data is shown.

In order to use the comments, all have been manually labeled with a score, ranging from -2 to 2. These scores represent how positive or negative the emotion expressed in this comment is. In Figure 1 the design for the Sentiment Analysis models is shown. The data will first be preprocessed before being handed to a machine learning algorithm. In the preprocessing module, text will be translated, filtered, stemmed and tokenized. This text can then be used by both models for Sentiment Analysis.

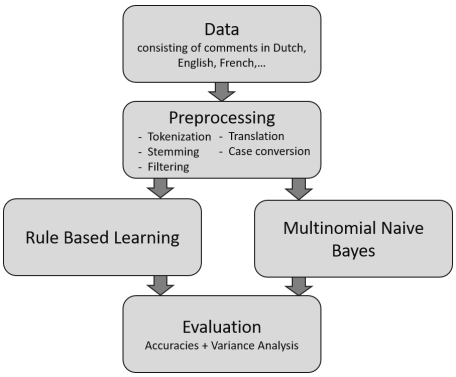


Fig. 1. Design of both Sentiment Analysis models.

3.1 Rule Based Machine Learning model

A RBS is a system that will classify data using rules. These rules are generally if-then clauses that determine what classification the system will make. In the case of text classification, a lexicon is always used to determine the sentiment of all the words. For this model we work with a lexicon consisting of subjectives and modifiers. Subjectives are words that contain sentiment. Modifiers are words that can reduce or enhance the polarity of words in a sentence. A score representing the sentiment of a comment C consisting of n groups G_i, where each group is a subjective S and m modifiers M, is calculated as follows:

$$TotalScore = \sum_{i=1}^{n} Score_{Gi} \tag{1}$$

Where the score of each individual group is calculated as:

$$Score = Weight(S) * \sum_{i=1}^{m} Weight(M_i) \tag{2}$$

This score is then passed to an evaluation function, which will map this score to a label being either positive, neutral or negative. This evaluation function works by using a neutral zone, an range going from a bordervalue B_1 to another bordervalue B_2. If the score of the text falls between these two values, this score will map to the label 'neutral'. If it's bigger it will map to the label 'positive' and if it's smaller it will map to the label 'negative'.

3.2 Multinomial Naive Bayes model

The MNB model uses the occurrences of certain words in a text to calculate for each class the probability that the text belongs this class. It learns which combination of words that give a high probability for a text being a certain class

by going over training data. For text classification, text needs to be converted to a vector, which can then be used by the MNB model. We compare four different comment representations, these four being word count vectors, TF-IDF vectors, TF-IDF Word level Ngram Vectors and TF-IDF Character Level Ngram Vectors. Word count vectors count for all the words in the corpus how many time each word occurs in each text. Tf-Idf also takes in account the amount of texts each word occurs in. Tf-Idf can also be applied to not standalone words, but to groups of 2 or more words, as well as groups of 2 or more characters. These groups are called ngrams. These vector representations are then given as input to the Multinomial Naive Bayes model. It uses the values of this vector representation as features, and the Bayes theorem to predict the class of a comment. Because of the small size of the dataset, only three classes are used instead of five. All negative values are combined to a class, '-1'. All positive values are also combined to one class, this being '1'. Comments that receive the class '1' are labeled as positive, comments that receive the class '0' are labeled neutral, and comments that are labeled '-1' are labeled negative.

4 Results

After checking the performance of all four vector representations, we find that using word count vectors gives us the best accuracy of all the four vectors. The accuracies of the different vectorisations can be found in Figure 2.

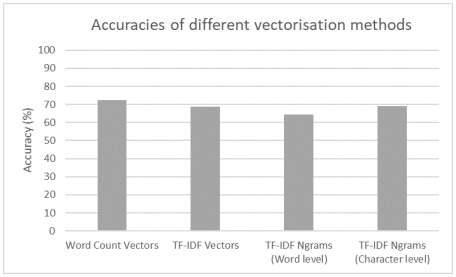


Fig. 2. Accuracy for each of the four vectorisations.

Because of their better performance, the word count vectorisation is used in the Multinomial Naive Bayes model that will be compared to the Rule Based Machine Learning model. To measure the performance of the RBS, the dataset was split randomly into ten, after which an accuracy was calculated for each of these ten subdatasets. This to get a mean accuracy, as well as a standard deviation on this accuracy. To check if the results of both models differ from eachother, variance analysis is performed on both these models. This variance analysis has as null hypothesis that the two models produce the same results. For the MNB model, the standard 10-fold cross validation was used. After the

10-fold cross validation, the MNB model achieves an average accuracy of 72,04%, with a standard deviation of 1,66%. The RBS achieves an average accuracy of 75,28%, with a standard deviation of 2,26%. Next, variance analysis is used to compare both models, and to check wether they differ enough from one another. A summary of the results of the variance analysis can be found in Figure 3.

SUMMARY

Groups	*Count*	*Sum*	*Mean*	*Variance*
Rule Based model	10	7,52799488	0,752799488	0,000511057
Naive Bayes model	10	7,20369987	0,720369987	0,00027477

Variance-Analysis

Source of variance	*Squared Sum*	*Degrees of freedom*	*Mean Squares*	*F*	*P-value*	*Critical value of F-test*
Between groups	0,005258363	1	0,005258363	13,38299659	0,001798191	4,413873
Within groups	0,007072447	18	0,000392914			
Total	0,012330809	19				

Fig. 3. Summary of the variance analysis between the Rule Based Machine Learning model and the Multinomial Naive Bayes model.

Because of a p-value of 0.001799 and a F value bigger than the critical value of the F-test, we can safely (certainty $> 99\%$) discard the null hypothesis.

Next, variance analysis is used again to evaluate wether translated comments lose sentiment when being translated or not. Here we find a difference of 8% between the accuracies of Dutch comments and translated comments. Because only 20% of the comments were non-Dutch, the MNB model trained on only these comments would already have a decreased accuracy, and thus be less accurate. This is why this comparison was done using the RBS.

5 Discussion

Both Rule Based Machine Learning and Multinomial Naive Bayes have shown to perform well when working with a small dataset. Both have shown to get accuracies higher than 70% when classifying texts, and can therefore be used to automatically rate comments on scenic routes. A big weakness of both of these models is the fact that they are both designed to work with Dutch comments. The RBS uses a lexicon consisting of only Dutch words, and the word count vector of the MNB model only counts Dutch words. We also found that translating texts to another language, in this case Dutch, can cause the sentiment of these texts to be lost, shown by the lowered accuracy of the Rule Based Machine Learning model for non-Dutch comments. To be able to succesfully process not only Dutch comments, the Naive Bayes model would need more trainingdata containing non-Dutch comments. For the Rule Based Learning Model, it would be necessary to generalize the rules, so they would be applicable to more languages than only Dutch, or to design a separate module in the Rule Based model for each language. It can also be noted that as the amount of comments on routes increases, the

MNB will perform better. This should be kept in mind for the future, given that each day, new comments on the scenic routes appear.

Other future work could be focused on using part-of-speech tagging to improve the rating of a route using its comments. By automatically knowing what the topic of the comment is, weight that is given to the sentiment of this comment could be enforced or reduced. This to make sure that comments that express a sentiment about a topic not related to the route do not influence the rated quality of this route. Combining Sentiment Analysis and automatic topic detection could produce a very effective way to process comments and automatically estimate the quality of a route.

6 Conclusion

Both Rule Based Machine Learning and Multinomial Naive Bayes are techniques that have shown to be able to successfully classify texts. They also work very well when there is only a small amount of data. In this case, a RBS achieved the best results and therefor should be used when classifying the user comments. This research also showed that to process non-Dutch comments, translating these comments to Dutch will not suffice.

References

1. A. Sulaiman, M. Alharbi, E. DeDoncker: Enhance a Deep Neural Network Model for Twitter Sentiment Analysis by Incorporating User Behavioral Information. (2017)
2. Y. Kim: Convolutional Neural Networks for Sentence Classification. (2014)
3. F. Kundi , A. Khan , S. Ahmad, M. Asghar: Lexicon-based sentiment analysis in the social web. Journal of Basic and Applied Scientific Research (2014)
4. M. Asghar, A. Khan, S. Ahmad, M. Qasim, I. Khan: Lexicon-enhanced sentiment analysis framework using rule-based classification scheme. PLoS ONE (2017)
5. T. de Souza Viana, M. de Oliveira, T. da Silva, M. Falcao, E. Gonalves: A message classifier based on multinomial Naive Bayes for online social contexts. Journal of Management Analytics (2018)

Bijlage D

Poster

KU LEUVEN

Technologiecampus Gent

Joran De Wilde

Sentiment Analysis of User Commentaries on Scenic Routes

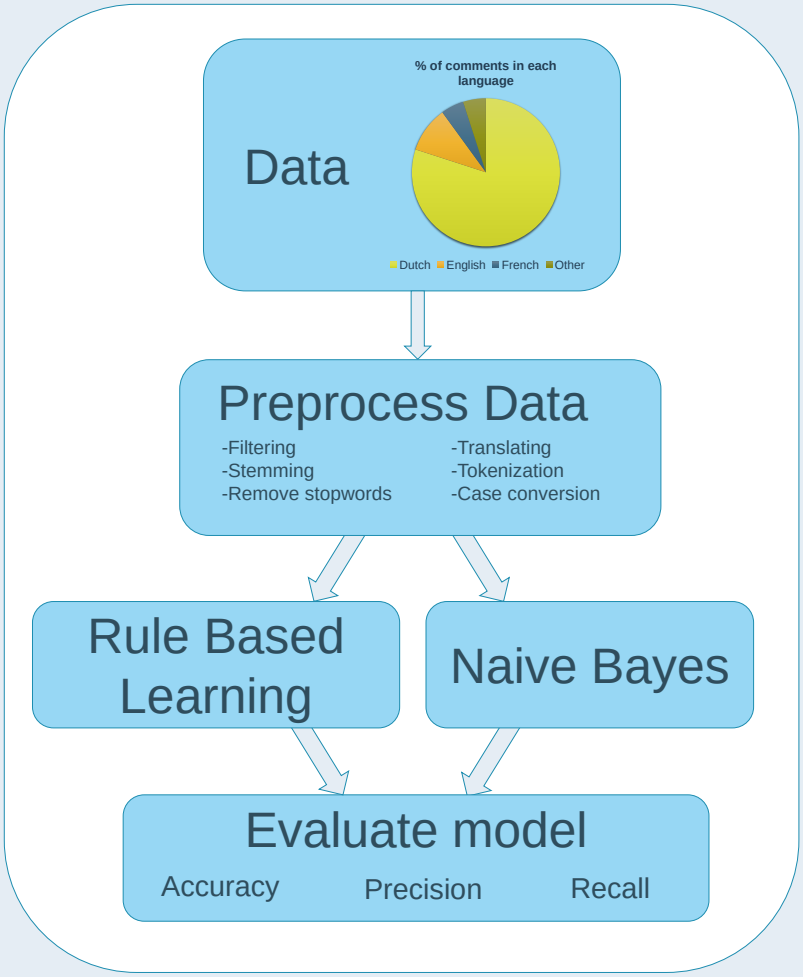
Many platforms today give their users the opportunity to leave comments. In this case, users can leave comments on bicycle, walking or motorcycle routes. These comments are used to get info about the quality of the routes. The rating of these comments is done manually and takes a couple of hours each week. These ratings are important to be able to estimate the quality of a route, and to use this quality to make the better routes stand out. In this thesis the aim is to investigate how Rule Based Learning and Naïve Bayes compare in processing these comments automatically. This processing means that the emotional charge of a comment is recognized.

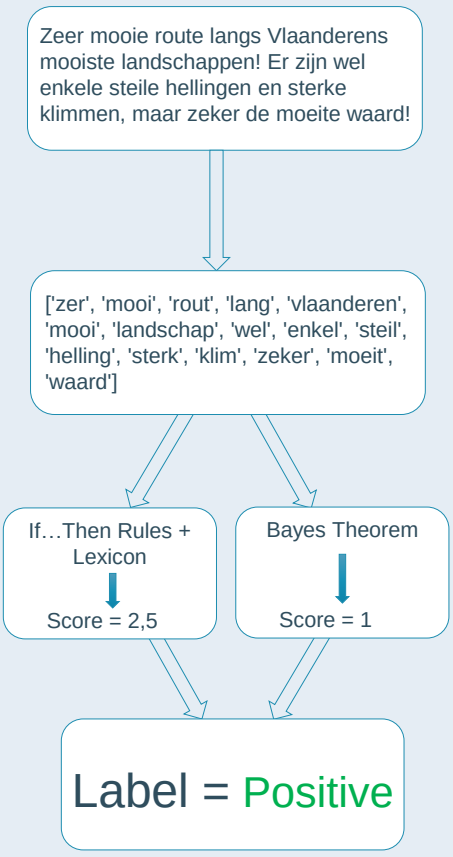
Goal: comparing the performance of 2 models trained to recognize the sentiment of a comment

'Toffe route, hier heb ik zeer van genoten' → Positive

'Deze route is 25km lang, met enkele steile klimmen' → Neutral

'Zeer lelijke weg, loopt bijna enkel langs snelwegen! → Negative





KU Leuven Technologiecampus Gent
Department of Computer Science
Gebroeders de Smetstraat 1
9000 Gent

Supervisor: G. Vanden Berghe
Co-Supervisor: M. Van Lancker
RouteYou: P. Brackman

Master of Electronics and ICT
Engineering Technology

FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
TECHNOLOGIECAMPUS GENT
Gebroeders De Smetstraat 1
8200 GENT, België
tel. + 32 50 66 48 00
iiw.gent@kuleuven.be
www.iiw.kuleuven.be