Afstudeerverslag

Ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *10-01-2014*

Versie: *1.0*

01. VOORWOORD

Voor het afronden van mijn HBO studie informatica op De Haagse Hogeschool heb ik een afstudeeropdracht uitgevoerd bij het bedrijf Liones. Voor het verkrijgen van deze opdracht wil ik mijn dank uitspreken aan Martin Middel die mij bij Liones heeft aanbevolen als afstudeerder.

Tijdens het uitvoeren van de afstudeeropdracht heb ik zeer naar mijn zin gehad binnen het bedrijf Liones en wil alle collega's bedanken voor het creëren van een fijne werksfeer. En de bereidheid van alle collega's om te helpen en vragen te beantwoorden wanneer het nodig was. Ook wil ik Tom Hendriks van Kluwer bedanken voor de goede samenwerking voor het tot stand komen van een mooi en bruikbaar product.

Bij het schrijven van dit verslag heb ik van veel mensen hulp gekregen in vorm van nakijken op spelling, opmerkingen over de structuur van het verslag en algemeen feedback. Hiervoor wil ik Ed van Doorn en Ben Kuiper van De Haagse Hogeschool bedanken voor het doorlezen en het leveren van feedback. En ook Gjalt Wijma en Martin Middel van Liones voor de feedback op mijn verslag en mijn moeder Els Lokker die heeft geholpen met het controleren op spelling.

INHOUDSOPGAVE

01. Voorwoord ii
02. Referaat 1
03. Inleiding 2
04. Organisatie omschrijving 3
05. Opdracht 5
05.1 Klanten en producten 5
05.2 Aanleiding opdracht 6
05.3 Opdracht omschrijving 7
06. Uitleg algoritmes 8
06.1 KNN 8
06.2 TextRank 10
07. Aanpak 14
07.1 Ontwikkel methode 14
07.2 Planning 16
07.3 Risico's 18
08. Inlezen 20
08.1 Taxonomieën 20
08.2 Semantisch web 22
08.3 KNN & CNN 23
08.4 Elasticsearch 24
08.5 TextRank 24
09. Requirements 25
10. Modellering 26
10.1 Lynkx framework 27
10.2 Importeerfuncties klassendiagram 30
10.3 Enquête-onderdeel klassendiagram 32
10.4 Database klassendiagram 34
10.5 Sequencediagrammen 36
11. Ontwikkeling 38
11.1 Importeerfuncties 38
11.2 Opzet tool 39

11.3 Implementatie KNN 40
11.4 Implementatie TextRank 41
11.5 Resultaten 42
11.6 Gebruiker trefwoord-suggesties + overig commentaar 42
11.7 Opstellen enquête 43
11.8 Refactoring importeerfuncties 44
11.9 Bug fixes en kleine features 44
12. Testen 47
12.1 Module test importeerfuncties 47
12.2 Acceptatietest 52
13. Analyse 56
13.1 Document analyse 56
13.2 KNN analyse 58
13.3 Redacteur analyse 60
13.4 Conclusie 63
14. Evaluatie 64
14.1 Proces 64
14.2 Product 65
14.3 Beroepstaken 66
15. Bibliografie 68
Bijlage A: Plan van aanpak .....
Bijlage B: Requirements rapport .....
Bijlage C: Module test .....
Bijlage D: Acceptatietest .....
Bijlage E: Analyse rapport .....

02. REFERAAT

Dit verslag houdt in het opstellen en ontwikkelen van een applicatie om algoritmes te beoordelen. De algoritmes waarop is gericht werken op basis van tekst om hierbij bijvoorbeeld: (trefwoorden te suggereren als metadata of automatisch samenvattingen opstellen). Ook staat de uitgevoerde analyse beschreven. Deze analyse richt zich op de kwaliteit van de aangeleverde documenten van Kluwer omtrent de metadatering. En hoe goed het algoritme KNN bij deze documenten trefwoorden heeft kunnen suggereren als metadata.

Steekwoorden

- Applicatie ontwikkeling
- Algoritme analyse
- Metadata suggesties
- KNN
- TextRank
- Information retrieval

03. INLEIDING

Dit afstudeerverslag gaat over de afstudeeropdracht “Ontwikkeling document trefwoordsuggesties” (hierna te noemen: de afstudeeropdracht), in de periode 26-08-2013 t/m 10-01-2014 (hierna te noemen: de afstudeerperiode).

De afstudeeropdracht richt zich op een applicatie om verschillende algoritmes te kunnen testen. Wat voor algoritmes dit zijn, staat in het verslag beschreven. Dit verslag zal inhouden wat is voorbereid voor het ontwikkelen van de applicatie, hoe de applicatie is ontwikkeld en hoe deze is getest.

Na het ontwikkelen is de applicatie ook nog gebruikt in de afstudeerperiode en is een analyse uitgevoerd om te bepalen hoe goed het algoritme KNN presteert op de documentenset van Kluwer.

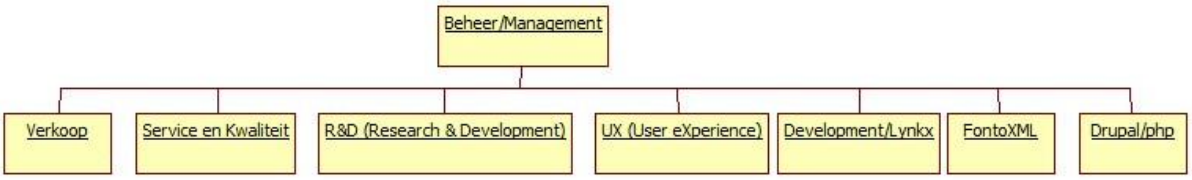
Voor verwijzingen en de bibliografie is gebruik gemaakt van de standaard APA 6^{e} editie.

04. ORGANISATIE OMSCHRIJVING

De afstudeeropdracht is uitgevoerd bij het bedrijf Liones. Liones is een internetbureau en richt zich op het maken en onderhouden van websites en webapplicaties. Veel klanten van Liones zijn uitgevers. Voor deze uitgevers biedt Liones een CMS (Content Managent System) dat is gebouwd met het framework Lynkx. Dit framework/CMS is geschreven in C# .NET en zal later in het verslag verder worden beschreven. Ook is Liones bezig met producten waarmee het voor redacteuren gemakkelijker wordt om artikelen te schrijven en te plaatsen. Naast Lynkx projecten worden ook mobiele applicaties ontwikkeld voor Android en iOS. Bij Liones zijn momenteel ongeveer dertig werknemers in dienst.

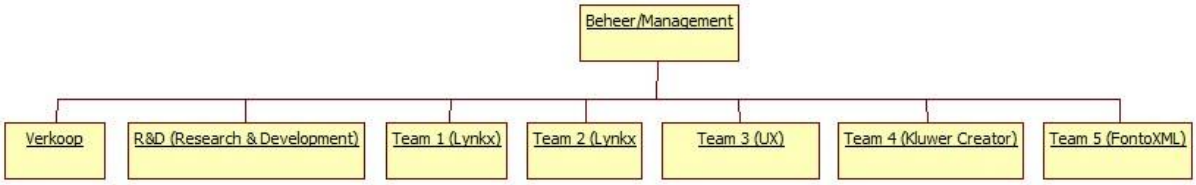
Omgang
Binnen Liones is een informele omgang tussen werknemers waarbij werknemers gemakkelijk naar andere werknemers kunnen toestappen voor vragen. Voor langere gesprekken wordt vaak een afspraak ingepland, maar ook deze afspraken worden door de werknemers zelf gepland en dit hoeft dus niet via een manager.

Structuur
De bedrijfsstructuur van Liones is begin 2014 gewijzigd, waarbij het meer is ingericht op de ontwikkelmethode SCRUM. De situatie van 2013 is weergeven in het eerste organogram en de situatie van 2014 is weergegeven in het tweede organogram.



Figuur 1 Organogram 2013

Liones heeft een platte structuur waarbij het bedrijf is onderverdeeld in verschillende afdelingen. Elke afdeling heeft andere verantwoordelijkheden. Boven de afdelingen staat de beheerafdeling die de andere afdelingen aanstuurt.



Figuur 2 Organogram 2014

In de nieuwe situatie zijn verschillende teams gevormd die elk in sprints werken. Hierbij is ervoor gezorgd dat binnen één team alle expertises aanwezig zijn, zodat dit team op zichzelf kan functioneren. Wel heeft elk team nog steeds zijn eigen specialiteit. Teams richten zich op Lynkx projecten, de Kluwer Creator of FontoXML. Naast deze teams zijn er nog de afdelingen Verkoop en R&D. De afdelingen

Verkoop en R&D zullen hetzelfde blijven functioneren als in 2013. Ook deze teams zullen nog worden aangestuurd door de beheerafdeling.

De producten en klanten die van belang zijn voor dit project worden in hoofdstuk 5.1 verder toegelicht.

Gedurende de afstudeerperiode ben ik werkzaam geweest binnen de afdeling R&D.

Website: www.liones.nl

05. OPDRACHT

Voor het omschrijven van de opdracht zijn eerst een paar klanten en producten van Liones toegelicht. Bij het uitvoeren van de opdracht is niet direct iets gedaan met de producten maar deze zijn van belang bij de aanleiding tot de opdracht.

Na de klanten en de producten is de opdracht zelf beschreven. Dit is opgedeeld in de aanleiding tot de opdracht en de omschrijving van de opdracht.

05.1 KLANTEN EN PRODUCTEN

Hieronder worden de in dit verslag genoemde klanten en producten omschreven.

Kluwer
De klant Kluwer is een uitgever die zich richt op vakliteratuur dat gericht is op bijvoorbeeld: “juridisch fiscaal”, “human resources” en “accounting”. Tijdens de opdracht is vooral gewerkt met juridische documenten.

Aangezien Kluwer een grote hoeveelheid content aanbiedt is het belangrijk dat de content goed is gemetadateerd zodat de klanten van Kluwer de content kunnen vinden waar ze naar op zoek zijn.

Voor het toevoegen van metadata kunnen trefwoorden uit de KBT (Kluwer Brede Thesaurus) aan documenten worden toegewezen. De KBT is een grote thesaurus van ongeveer 17000 trefwoorden. Hierdoor moeten de auteurs van de content voor Kluwer veel door de structuur van de KBT klikken voordat ze de correcte termen voor de content hebben gevonden.

Voor Kluwer is het dus belangrijk dat het voor de auteurs zo gemakkelijk mogelijk is om content te schrijven en deze juist te metadateren.

Kluwer is een belangrijke klant voor dit project, aangezien voor hen al eerder onderzoek is gedaan. Dit was dan ook een belangrijke aanleiding tot dit project. Verder is Kluwer nog zeer nauw betrokken bij het project ook omdat zij de tool al gedurende het project in gebruik zullen nemen. Dit onderzoek staat ook vermeld in paragraaf 5.2.

Bron: (Kluwer, 2013)

Bindinc
De klant Bindinc is een uitgever gericht op tv-gidsen. Zij hebben veel te maken met samenvatten wat er in een tv-programma of film gebeurt. Om het samenvattingsproces te automatiseren of te versnellen was er vanuit deze klant interesse in een samenvattingsalgoritme.

Bron: (Bindinc, 2013)

Kluwer Creator
De Kluwer Creator is door Liones specifiek voor Kluwer ontwikkeld. Dit is een web applicatie waarin auteurs voor Kluwer content kunnen schrijven. Dit product is al in gebruik genomen. Ook is een team van Liones nog steeds voltijd bezig met verder ontwikkelen. Ook de KBT staat in de Kluwer Creator en kan worden geraadpleegd door middel van een boomstructuur.

FontoXML
Dit is een product dat wordt ontwikkeld door Liones. In het kort is FontoXML een WYSIWYG (What You See Is What You Get) XML editor. Hiermee kunnen gebruikers online documenten maken die worden opgeslagen als XML.

Bron: (Liones, 2013)

05.2 AANLEIDING OPDRACHT

De klant Kluwer heeft interesse in een algoritme om auteurs en redacteuren te helpen bij het metadateren van artikelen. Het algoritme moet een aantal trefwoorden suggereren waaruit de auteur/redacteur kan kiezen. Als deze selectie goed is dan zal de auteur/redacteur de trefwoorden voor het document uit deze selectie kunnen nemen. En hoeft niet door alle trefwoorden uit de KBT te zoeken. Het zoeken naar de trefwoorden is momenteel veel werk omdat de trefwoorden zijn onderverdeeld in een boomstructuur en het zijn zeer veel trefwoorden (meer dan 17.000).

Hiervoor is al eerder een onderzoek gedaan naar verschillende algoritmes die hiervoor gebruikt kunnen worden (Middel, 2013). Uit dit onderzoek is gebleken dat het algoritme KNN het beste presteert en hierbij is K=25 als beste waarde gevonden met het model BM25. KNN is een algoritme dat naar de K meest lijkende documenten zoekt, van de gevonden documenten kunnen de toegewezen trefwoorden gebruikt worden om trefwoorden voor het nieuwe document te suggereren. Voor een uitgebreidere uitleg van het algoritme KNN zie hoofdstuk 06.1. Tijdens dit onderzoek hebben de redacteuren van Kluwer ook al een voor een aantal documenten de KNN trefwoord suggesties beoordeeld. Maar dit was een zeer tijdrovend proces omdat de resultaten werden aangeleverd in een word document. Hierdoor moesten de redacteuren de documenten in een ander systeem opzoeken en ook voor het geven van de beoordelingen was het geen goede manier.

Voor Kluwer is het belangrijk dat het voor de redacteuren gemakkelijker en sneller is om de trefwoord suggesties van KNN te beoordelen en deze feedback aan Liones te leveren. Want hierdoor zal Liones het nodige inzicht krijgen om nog verder te werken aan het algoritme of te zoeken naar andere oplossingen. Ook is het belangrijk om de testen uit te voeren op een grotere documentenset van Kluwer om uit te sluiten dat de resultaten alleen gelden voor slechts een deel van de documenten.

Liones is zelf bezig met de ontwikkeling van FontoXML. Hierbij wil Liones ook functies kunnen aanbieden zoals het suggereren van trefwoorden en/of het automatisch maken van samenvattingen. Hiervoor is het belangrijk dat er een manier is om de klanten de prestaties van het algoritme te laten zien en deze te laten beoordelen voordat deze wordt geïmplementeerd in de productie-omgeving. Ook deze beoordelingen zullen Liones meer inzicht geven over de kwaliteit van de algoritmes en of deze gebruikt kunnen worden of dat naar een andere oplossing gezocht moet worden.

05.3 OPDRACHT OMSCHRIJVING

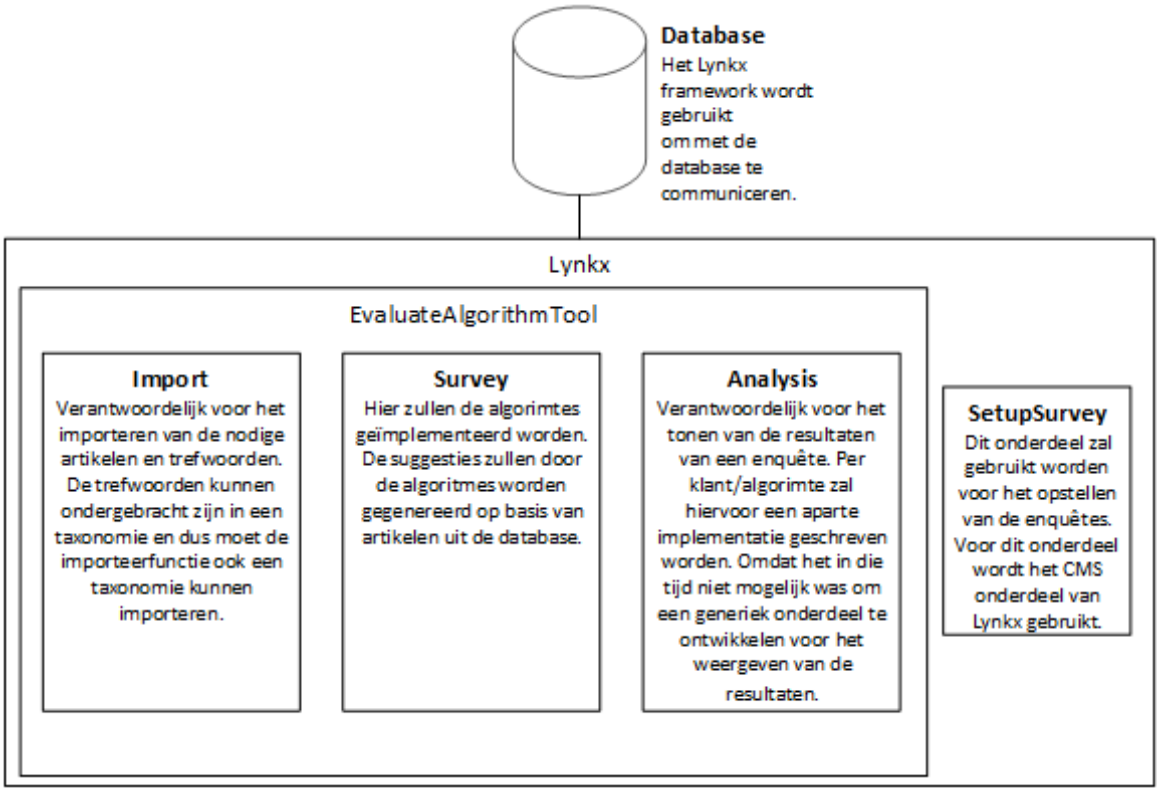
Om de wensen van Kluwer en Liones te combineren moet er één applicatie worden ontwikkeld waarmee verschillende algoritmes getest kunnen worden. De applicatie zal ontwikkeld worden in C# .NET gebruikmakend van het framework Lynkx. Het Lynkx framework biedt de mogelijkheid om snel beheer user interfaces te ontwikkelen voor database content applicaties.

Bij de applicatie moeten verschillende algoritmes gebruikt kunnen worden. Hierbij zal gericht worden op algoritmes die op basis van een tekst suggesties doen. Deze suggesties kunnen in de vorm zijn van bijvoorbeeld: trefwoorden voor metadatering of samengestelde samenvattingen.
Voor het beoordelen van de algoritmes zullen de suggesties worden beoordeeld door een gebruiker door middel van een score. Hierdoor zal duidelijk worden hoe goed suggesties van het algoritme zijn.

Omdat de trefwoorden van Kluwer zijn ondergebracht in een taxonomie[1] en dit ook mogelijk kan zijn bij andere klanten, moet de applicatie met verschillende taxonomieën om kunnen gaan.

Voor Kluwer zal de applicatie ook gebruikt worden met een implementatie van KNN met de waarde K=25 en met het model BM25. De resultaten die hieruit komen zullen ook nog geanalyseerd worden. Naast het analyseren van de resultaten uit de applicatie, zullen ook verschillende analyses uit het vorige onderzoek herhaald worden maar dan gebruikmakend van een grotere documentenset van Kluwer.

In figuur 3 is te zien welke delen de applicatie zal omvatten en hoe deze communiceert met de database. De delen uit het diagram zullen elk later in het verslag aan bod komen.



Figuur 3 Architectuur applicatie

[1] Zie hoofdstuk 8.1 voor uitleg taxonomie.

06. UITLEG ALGORITMES

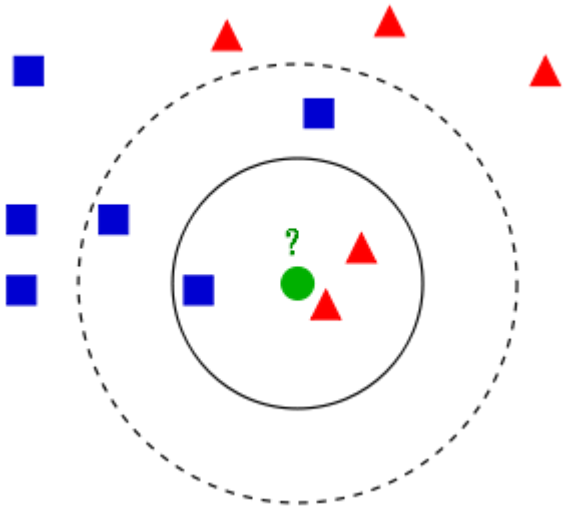
Tijdens de afstudeeropdracht zijn twee algoritmes geïmplementeerd en gebruikt. KNN is gebruikt voor het suggereren van trefwoorden als metadata en TextRank is gebruikt voor het automatisch samenvatten van tekst. In het verslag zullen deze algoritmes genoemd worden, hierom zijn deze in dit hoofdstuk uitgelegd. De implementatie van de algoritmes zal in een later hoofdstuk worden toegelicht.

06.1 KNN

KNN (K-Nearest Neighbors) is een classificatie algoritme. Om dit algoritme uit te leggen wordt gebruik gemaakt van figuur 4.

Met het algoritme KNN kan voor een nieuw punt bepaald worden wat voor een classificatie het moet krijgen aan de hand van andere al geclassificeerde punten.

In het voorbeeldfiguur stelt de groene cirkel het nieuwe punt voor dat geclassificeerd moet worden. De mogelijke classificaties zijn een rode driehoek of een blauw vierkant. De blauwe vierkanten en rode driehoeken in het figuur stellen al geclassificeerde punten voor.



Figuur 4 KNN uitleg (http://en.wikipedia.org/wiki/File:KnnClassification.svg)

De basis van KNN is het zoeken naar K dichtstbij liggende punten. De waarde K is instelbaar en heeft effect op hoe het algoritme functioneert. KNN zal de K dichtstbij liggende punten zoeken en de classificatie die het meeste voorkomt binnen de gevonden punten zal de classificatie zijn voor het nieuwe punt.

Om het effect van de waarde K duidelijk te laten zien zijn twee cirkels getrokken in figuur 4. De onderbroken cirkel wordt gebruikt bij de waarde K=5 en de ononderbroken cirkel wordt gebruikt bij de waarde K=3.
Bij de waarde K=3 is te zien dat de 3 dichtstbij liggende punten bij de groene cirkel, 2 rode driehoeken en 1 blauw vierkant zijn. Omdat van deze 3 punten de rode driehoeken het meest voorkomen zal het nieuwe punt als classificatie een rode driehoek krijgen.
Bij de waarde K=5 zijn de 5 dichtstbij liggende punten bij de groene cirkel, 2 rode driehoeken en 3 blauwe vierkanten. In dit geval zijn de meeste punten een blauw vierkant en dus zal het nieuwe punt geclassificeerd worden als een blauw vierkant.

Voor dit project is KNN gebruikt om artikelen te classificeren, hiervoor is gebruik gemaakt van Elasticsearch. Voor het vinden van de documenten biedt Elasticsearch een more like this (vind vergelijkbare documenten) query waarmee de meest lijkende documenten opgevraagd worden. Bij deze query wordt het artikel meegegeven waarvoor lijkende documenten gezocht moeten worden en hoeveel lijkende documenten gezocht moeten worden. Het aantal documenten dat gezocht moet worden is de waarde K. Elasticsearch zal dus de K meest lijkende documenten tonen. Van deze

documenten kan de metadata worden opgevraagd, deze metadata bestaat uit toegewezen trefwoorden. Uit de selectie trefwoorden worden vervolgens de 10 meest voorkomende trefwoorden geselecteerd en deze worden getoond als trefwoord suggesties.

Elasticsearch

Elasticsearch is een distributed searchengine die gebruik maakt van Lucene (een tekst gebaseerde zoekmachine in de vorm van een library). Hiermee kunnen veel geavanceerdere queries worden uitgevoerd wat ook sneller zal gaan dan met een relationele database zoals MSSQL.

Voor het gebruiken van Elasticsearch is geen kennis nodig hoe het werkt omdat met een query de meest lijkende documenten worden teruggegeven. De documenten uit de set zijn de K dichtsbij liggende punten.

Wel is onderzocht hoe Elasticsearch werkt om inzicht te geven hoe de documenten gevonden worden. Voor het zoeken naar de documenten biedt Elasticsearch de modellen tf/Idf en BM25. Bij het project is BM25 gebruikt want uit een voorgaand onderzoek is gebleken dat deze beter presteert dan tf/Idf. Beide modellen werken door de nieuwe tekst om te zetten naar een vector. Hierbij stelt elk woord een dimensie voor met daarbij hoe vaak dat woord in de tekst voorkomt. Vervolgens worden voor de al geclassificeerde documenten ook vectoren opgesteld waarbij dezelfde vector zal worden opgesteld als die van het nieuwe document maar hierbij zullen de keren dat een woord voorkomt verschillen.

tf/Idf gebruikt het natuurlijke logaritme bij het opstellen van de vectoren om ervoor te zorgen dat documenten die met zoveel mogelijk woorden overeenkomen ook beter scoren dan documenten die met slechts een paar woorden hoog scoren. Verder zorgt tf/Idf ervoor dat woorden die veel voorkomen in een taal minder hoog scoren. Dit soort woorden worden ook vaak benoemd tot ruis.

BM25 werkt voor een groot deel volgens hetzelfde principe als tf/Idf maar voert andere compensaties uit om er ook voor te zorgen dat de lengte van tekst minder uitmaakt. De formules van BM25 zijn echter een stuk ingewikkelder dan die van tf/Idf.

Bronnen: (k-nearest neighbors algortihm - Wikipedia, the free encyclopedia, 2013), (Sayad, 2013), (Sutton, 2012), (Cheng, Tan, & Jin, 2007), (Elasticsearch, 2013), (Elasticsearch - Wikipedia, the free encyclopedia, 2013), (Middel, 2013) , (The Apache Software Foundation, 2013), (Beiske, 2013)

06.2 TextRank

TextRank is een ranking algoritme voor tekst. TextRank biedt de mogelijkheid om de woorden uit de tekst te ranken en zo de kernwoorden uit de tekst te halen of om de volledige zinnen van de tekst te ranken en kan zo van de belangrijkste zinnen een samenvatting maken. In deze uitleg van TextRank zal ik het enkel hebben over het ranken van de zinnen en het opstellen van een samenvatting omdat enkel deze functie van TextRank in het project is gebruikt.

Voor het uitleggen van TextRank zal ik gebruik maken van de drie figuren figuur 5, figuur 6 & figuur 8 uit (Mihalcea & Tarau, 2004). Figuur 5 is de tekst waarvan een samenvatting zal worden gemaakt. De zinnen zijn in het figuur genummerd zodat deze terug te vinden zijn in het tweede figuur. Figuur 6 is het diagram dat is opgesteld door TextRank. En in Figuur 8 staan drie samenvattingen, de eerste samenvatting is opgesteld door TextRank en de andere twee samenvattingen zijn handmatig geschreven en staan erbij als vergelijkingsmateriaal.

In het kort

TextRank zal de tekst opknippen in zinnen en deze zinnen nummeren zoals te zien in figuur 5. Voor elke zin wordt de representatieve waarde bepaald, door te bepalen met hoeveel andere zinnen uit de tekst de zin een overeenkomst heeft, zie figuur 6. Vervolgens wordt de samenvatting opgesteld door de zinnen met de hoogste waarden achter elkaar te plakken, zie figuur 8.

Uitgebreid

De uitleg is opgedeeld in drie delen. Elk deel is een stap die door TextRank wordt uitgevoerd voor het opstellen van de samenvatting.

Stap 1 (Figuur 5)

De tekst moet eerst worden verdeeld in zinnen. Om te bepalen waar een zin begint en waar een zin eindigt maakt TextRank gebruik van de Sentence Detector van OpenNLP. OpenNLP is een library voor het verwerken van tekst en wordt in verschillende talen aangeboden waaronder Nederlands. De Sentence Detector kan bepalen of een punt het einde van een zin markeert of dat de punt als een ander leesteken dient. In het Engelse voorbeeld hieronder is duidelijk te zien dat het einde van een zin goed kan worden bepaald. Vervolgens geeft TextRank elke zin een nummer zodat deze terug te vinden zijn in de volgende stap.

Voorbeeld OpenNLP

Originele tekst:

```
1. Pierre Vinken, 61 years old, will join the board as a nonexecutive
   director Nov. 29. Mr. Vinken is chairman of Elsevier N.V., the Dutch
   publishing group. Rudolph Agnew, 55 years old and former chairman of
   Consolidated Gold Fields PLC, was named a director of this British
   industrial conglomerate.
```

Opgesplitste tekst, elke zin staat op een aparte regel:

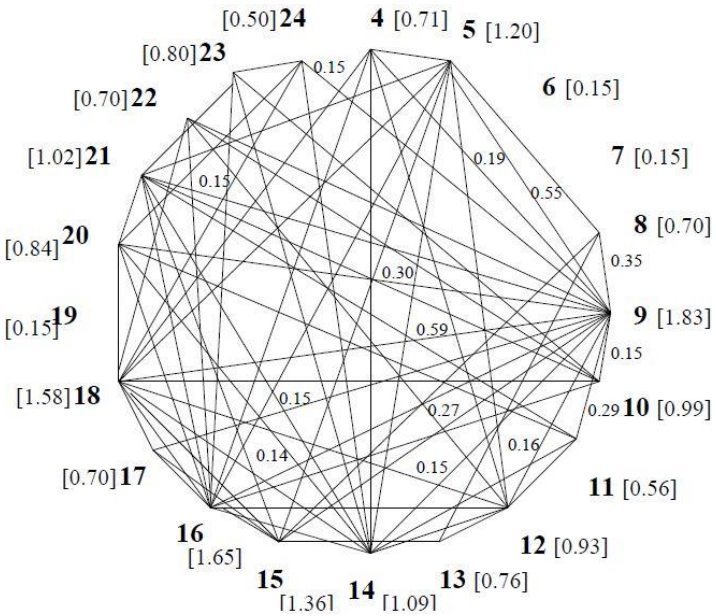
```
1. Pierre Vinken, 61 years old, will join the board as a nonexecutive
   director Nov. 29.
2. Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.
3. Rudolph Agnew, 55 years old and former chairman of Consolidated Gold
   Fields PLC, was named a director of this British industrial
   conglomerate.
```

3: BC−HurricaineGilbert, 09−11 339
4: BC−Hurricaine Gilbert, 0348
5: Hurricaine Gilbert heads toward Dominican Coast
6: By Ruddy Gonzalez
7: Associated Press Writer
8: Santo Domingo, Dominican Republic (AP)
9: Hurricaine Gilbert Swept towrd the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains, and high seas.
10: The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph.
11: "There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly after midnight Saturday.
12: Cabral said residents of the province of Barahona should closely follow Gilbert's movement.
13: An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo.
14. Tropical storm Gilbert formed in the eastern Carribean and strenghtened into a hurricaine Saturday night.
15: The National Hurricaine Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo.
16: The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westard at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm.
17. The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.
18: Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds, and up to 12 feet to Puerto Rico's south coast.
19: There were no reports on casualties.
20: San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night.
21: On Saturday, Hurricane Florence was downgraded to a tropical storm, and its remnants pushed inland from the U.S. Gulf Coast.
22: Residents returned home, happy to find little damage from 90 mph winds and sheets of rain.
23: Florence, the sixth named storm of the 1988 Atlantic storm season, was the second hurricane.
24: The first, Debby, reached minimal hurricane strength briefly before hitting the Mexican coast last month.

Figuur 5 TextRankuitleg 1 (http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf)

Stap 2 (figuur 6)
Van de zinnen die in de eerste stap zijn bepaald wordt vervolgens een diagram opgesteld. Het opstellen gebeurt door alle zinnen te doorlopen en voor elke zin de score te bepalen. Deze score geeft weer hoe belangrijk de zin is in de tekst. De zin met de hoogste score zal dan ook het meeste zeggen over de tekst.

De score van een zin wordt bepaald door alle andere zinnen te doorlopen en te bepalen of de twee zinnen een relatie met elkaar hebben. In figuur 6 zijn de gevonden relaties aangegeven door middel van lijnen. En bij elke zin (nummer) staat de score weergegeven binnen blokhaken.



Figuur 6 TextRank uitleg 2 (http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf)

Voor het bepalen van de overeenkomst tussen twee zinnen wordt de formule in figuur 7 gebruikt. Met de formule wordt de overlap tussen twee zinnen bepaald. Een grotere overlap zal een grotere score opleveren dan een kleinere overlap. Bij de formule staat S_I voor één zin, S_J voor een tweede zin en w_K voor woord K uit een zin. Het aantal overeenkomende woorden wordt gedeeld door de lengte van de zinnen om ervoor te zorgen dat niet alleen lange zinnen een hoge score krijgen maar dat ook korte zinnen een hoge score kunnen krijgen.

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)}$$

Figuur 7 TextRank formule (http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf)

Stap 3 (figuur 8)

Als laatste stap gebruikt TextRank het opgestelde diagram voor het samenstellen van de samenvatting. Hiervoor kan worden opgegeven uit hoeveel woorden de samenvatting ongeveer moet bestaan. Vervolgens zal TextRank zinnen uit het diagram nemen beginnend bij de zin met de hoogste score naar zinnen met daarna de hoogste score. TextRank zal hieruit zinnen nemen totdat het gewenste aantal woorden is gehaald of als er niet meer zinnen zijn. Vervolgens zal TextRank de zinnen in chronologische volgorde aan elkaar plakken om ervoor te zorgen dat samenvatting goed leesbaar blijft.

TextRank extractive summary

Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil De–fense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo. The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm. Strong winds associated with Gilbert brought coastal flooding, strong southeast winds and up to 12 feet to Puerto Rico's south coast.

Manual abstract I

Hurricane Gilbert is moving toward the Dominican Republic, where the residents of the south coast, especially the Barahona Province, have been alerted to prepare for heavy rains, and high wind and seas. Tropical storm Gilbert formed in the eastern Carribean and became a hurricane on Saturday night. By 2 a.m. Sunday it was about 200 miles southeast of Santo Domingo and moving westward at 15 mph with winds of 75 mph. Flooding is expected in Puerto Rico and in the Virgin Islands. The second hurricane of the season, Florence, is now over the southern United States and down–graded to a tropical storm.

Manual abstract II

Tropical storm Gilbert in the eastern Carribean strenghtened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday to be about 140 miles south of Puerto Rico and 200 miles southeast of Santo Domingo. It is moving westward at 15 mph with a broad area of cloudiness and heavy weather with sustained winds of 75 mph gusting to 92 mph. The Dominican Republic's Civil Defense alerted that country's heavily populated south coast and the National Weather Service in San Juan, Puerto Rico issued a flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.

Figuur 8TextRank uitleg 3 (http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf)

Bronnen: (Mihalcea & Tarau, 2004), (Storojev, 2010), (The Apache Software Foundation, 2013)

07. AANPAK

In dit hoofdstuk staat de aanpak beschreven voor de afstudeeropdracht “Ontwikkeling document trefwoordsuggesties” bij het bedrijf Liones. Hierbij zal de aanpak worden besproken met daarbij de planning en welke risico’s hierbij zijn voorzien.

07.1 ONTWIKKEL METHODE

In dit project is gewerkt met de ontwikkel methode SCRUM. Het project is uitgevoerd binnen de afdeling R&D (Research & Development). Typerend voor projecten binnen de R&D afdeling is dat de projecten en te ontwikkelen producten in het begin van het project nog niet helemaal duidelijk zijn. Voor dit soort projecten is SCRUM een geschikte methode omdat hierbij het te ontwikkelen product duidelijker wordt gedurende het project.
Delen van SCRUM die niet goed toepasbaar zijn voor dit project zijn niet gebruikt. Zo is de stand up meeting achterwege gelaten omdat het ontwikkelteam uit één persoon bestaat.

Sommige rollen binnen SCRUM zijn door dezelfde persoon uitgevoerd doordat er slechts één persoon was voor het ontwikkelen. In tabel 1 staan de rollen van SCRUM en door wie de rol is vervuld.

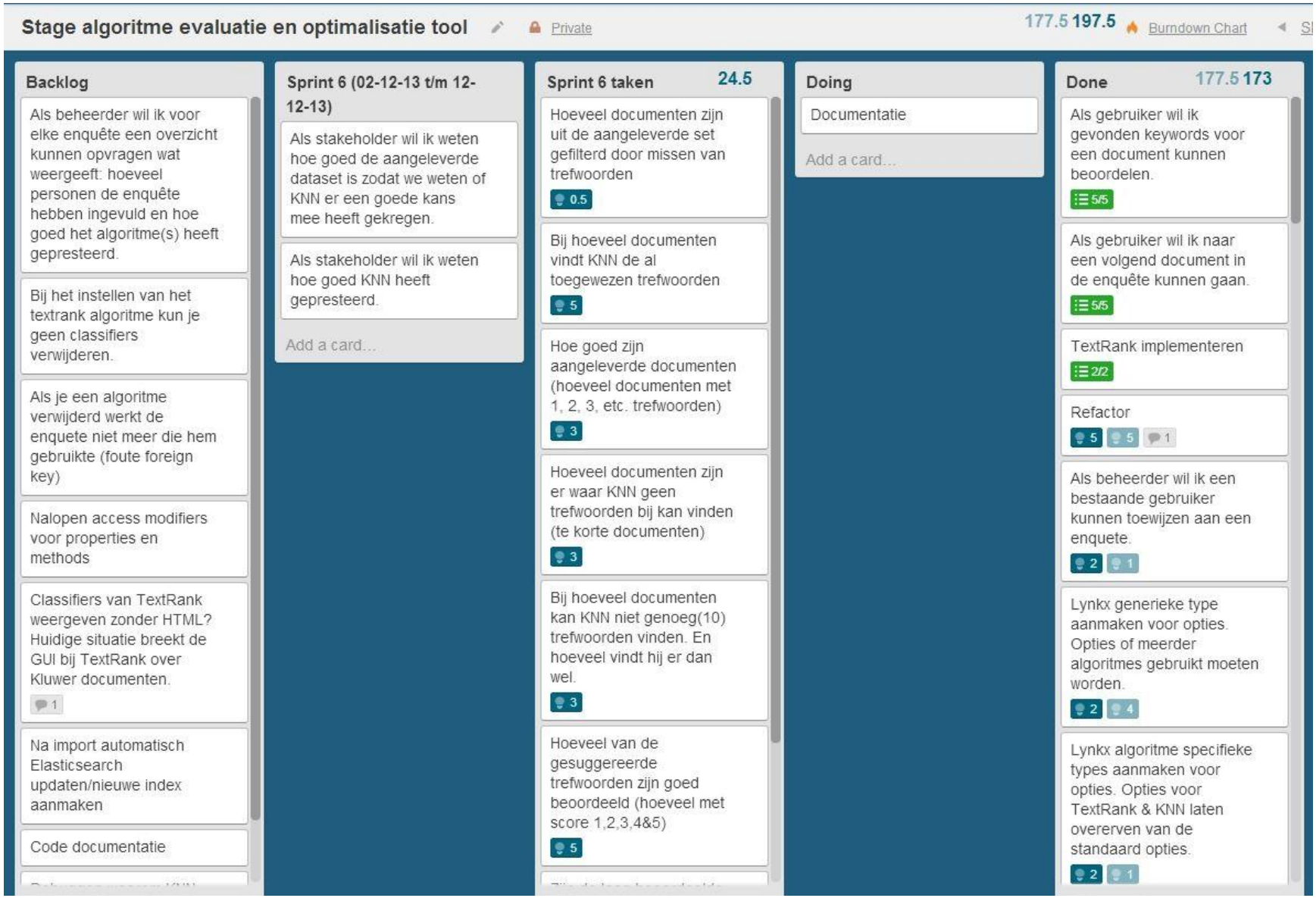
Tabel 1 SCRUM rollen

Rol	**Persoon**
Product owner	Software architect binnen Liones.
Ontwikkelteam	Student (ik)
Scrummaster	Student (ik)
Stakeholder	Kluwer contactpersoon.

De SCRUM backlog is in eerste instantie gevuld met user stories uit het eisen document. Daarna is de backlog tijdens het ontwikkelen bijgevuld wanneer bugs zich voordeden of wanneer er eisen bij kwamen. Wanneer een bug in de backlog werd geplaatst is dat niet altijd gedaan in de vorm van een user story. Op deze manier fungeerde de backlog ook als een algemene to-do lijst voor het project. Een voorbeeld van een eis er die tijdens het ontwikkelen bij is gekomen is: “Als gebruiker wil ik overig commentaar kunnen invullen voor een document uit de enquête.”.

Voor het werken met SCRUM is gebruik gemaakt van Trello. Trello is een online project management applicatie. Deze is niet speciaal ontwikkel om voor SCRUM te gebruiken maar deze is er wel goed geschikt voor. In Figuur 9 is te zien hoe Trello gebruikt is tijdens dit project. Hierbij worden de user stories voor de komende sprint gesleept naar het blok voor deze sprint. Daarna wordt elke user story onderverdeeld in taken en worden de verwachte uren toegewezen. Deze taken worden weer in een apart blok geplaatst. Wanneer er aan een taak wordt gewerkt wordt deze verplaatst naar het blok “Doing”. Hierbij kan er ook nog een persoon aan toegewezen worden maar in dit project is dat niet nodig omdat het ontwikkelteam uit één persoon bestaat. Het testen wordt ook gedaan terwijl de taak in het blok “Doing” staat. Als de taak klaar is wordt deze verplaatst naar het blok “Done” en wordt ingevuld hoeveel uur eraan is gewerkt. Indien gewenst kan er nog commentaar of een document toegevoegd worden aan de taak.

Figuur 9 dient enkel als een voorbeeld hoe met het programma Trello is gewerkt. Hier kunnen dus termen in staan die verder niet uitgelegd zijn.



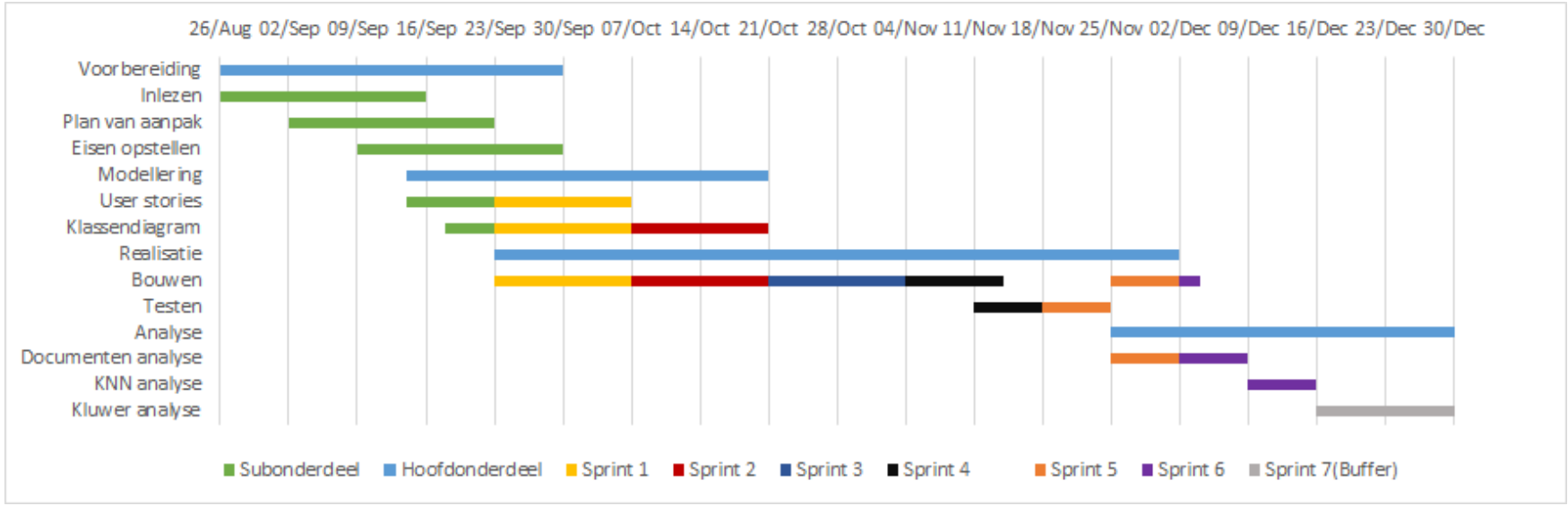
Figuur 9 Voorbeeld Trello

07.2 Planning

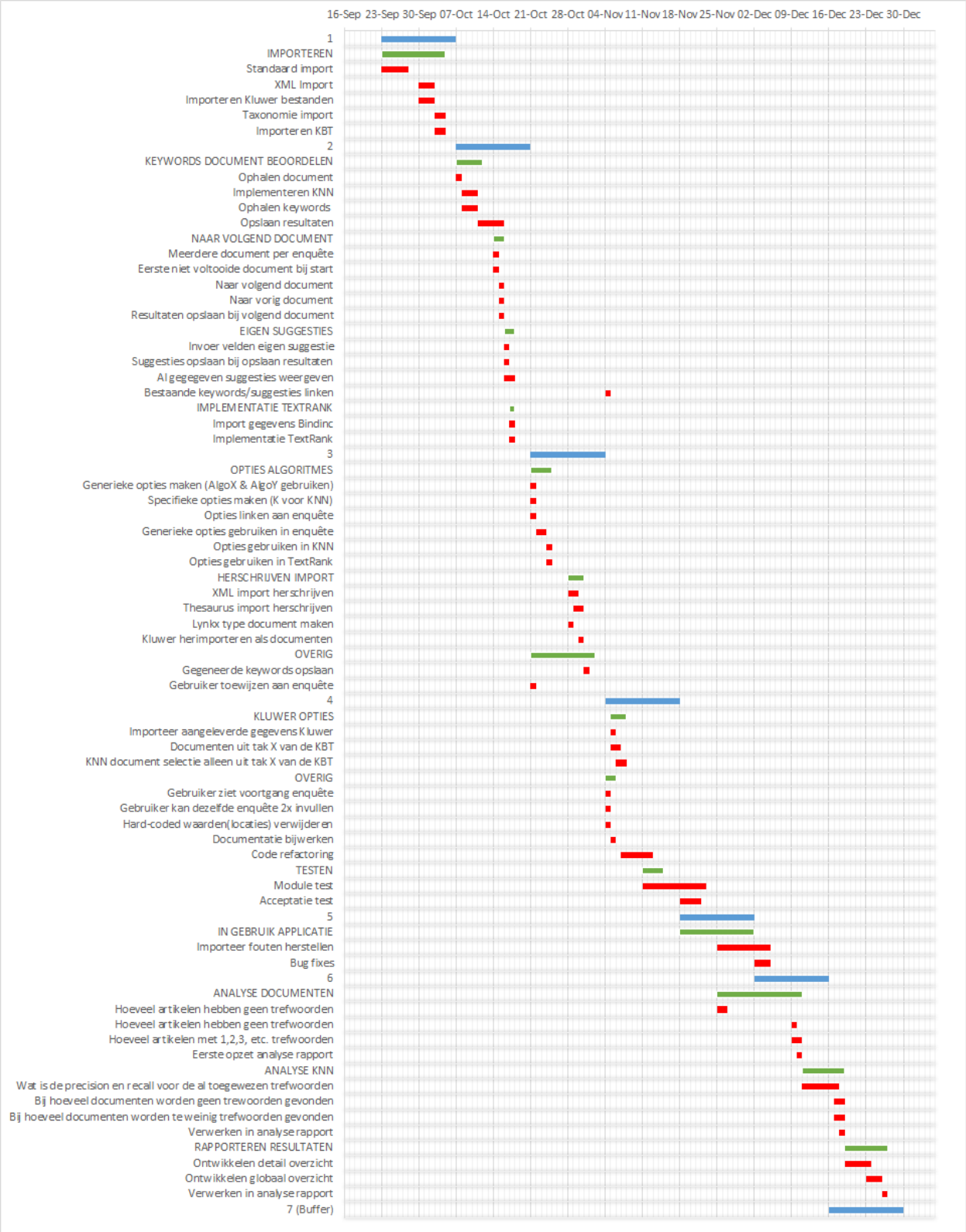
Het project is onderverdeeld in vier hoofdonderdelen, elk hoofdonderdeel is op zijn beurt weer onderverdeeld in subonderdelen. Vanaf het hoofdonderdeel realisatie zijn de subonderdelen verdeeld in sprints.

In het eerste hoofdonderdeel is de voorbereiding gedaan voor het project. In deze periode is ook bepaald welke ontwikkelmethode gebruikt zal worden voor de rest van het project. Omdat tijdens dit onderdeel pas is bepaald dat met SCRUM zal worden gewerkt, zijn de eerste vier weken niet verdeeld in sprints. De subonderdelen die onder sprints zijn verdeeld zijn gemarkeerd met de verschillende kleuren van de sprints. Deze planning is te zien en een Gantt chart in figuur 10.

Aan het begin van een sprint is de sprint zelf verder ingepland. Bij het inplannen zijn user stories van de backlog verplaatst naar de sprint en hierbij is bepaald hoeveel uur elke user story ongeveer zal kosten. De planning van de sprints is weergegeven in een Gantt chart in figuur 11. Hierbij zijn de sprints aangegeven met de cijfers, in hoofdletters staat het doel voor de sprint en de rest zijn de user stories. De user stories zijn in de Gantt chart verkort om het leesbaar te houden. De onderdelen voorbereiding, modellering en realisatie spreken voor zich. De analyse houdt in dat de prestaties van KNN voor de artikelen van Kluwer worden geanalyseerd.



Figuur 10 Planning



Figuur 11 SCRUM planning

07.3 Risico's

Tijdens het opstellen van het plan van aanpak is er ook gekeken naar de mogelijke risico's die zich kunnen voordoen tijdens het project. Hieruit is slechts één risico gekomen wat concreet omschreven kon worden en waar ook maatregelen tegen zijn genomen. De standaard risico's zoals ziekte zijn niet opgenomen.

Het risico uit het plan van aanpak is het falen van hardware. Hierbij is de kans als laag ingeschat omdat er is gewerkt met nieuwe hardware en deze hardware normaal jaren meegaat. Hiernaast is de impact echter op hoog ingeschat omdat verlies van data het project in gevaar zou brengen. Hiervoor is de volgende maatregel getroffen zoals omschreven in het plan van aanpak.

Maatregel voorkomen: Er zal regelmatig (minimaal wekelijks) een back-up worden gemaakt. Hierbij wordt de code geüpload naar een SVN en documenten naar Dropbox. Hierdoor zal bij het falen van hardware niet zoveel voortgang verloren gaan dat kan leiden tot het falen van het project. Deze maatregel zal de impact van dit risico verlagen van hoog – laag naar laag.
Maatregel wanneer het voorkomt: Wanneer de hardware faalt zal een back-up worden terug gezet naar een tijdelijke machine totdat er vervangende hardware is. Hierdoor zal er geen werktijd verloren gaan.

Tekstvak 1 Risico hardware falen

Dit risico is gedurende het project niet opgetreden.

Als ik terugkijk op dit project zie ik wel dat er risico's zijn voorgekomen die ik niet van tevoren had voorzien en een volgende keer ook zou opnemen in het plan van aanpak.

Het grootste risico wat is voorgekomen wat niet was voorzien is scopecreep. Dit houdt in dat de opdracht meer is gaan inhouden dan vooraf was bepaald. Zo is bij dit project het onderdeel analyse erbij gekomen terwijl dit niet was meegenomen bij het opstellen van de aanpak en de planning.

Deze zou ik als volgt opnemen:

Risico scopecreep
Kans: Medium. Het gaat bij dit project om een research & development project hierbij is het lastig om aan het begin van het project duidelijk te krijgen wat de verwachtingen zijn van de betrokken partijen.
Impact: Medium.
Maatregel voorkomen: Om dit te voorkomen moet de opdracht in het begin zo goed mogelijk geformuleerd worden. Standaard wordt hier ook naar gestreefd en dit zal de kans van voorkomen niet verder verlagen. Wel is belangrijk dat de betrokken partijen akkoord gaan met de maatregel wanneer het voorkomt.
Maatregel wanneer het voorkomt: Wanneer blijkt dat de opdracht groter wordt dan van te voren is verwacht zal eerst worden gekeken of de planning zo kan worden aangepast zodat dit ook kan worden uitgevoerd. (bijvoorbeeld door onderdelen die voorspoediger zijn verlopen dan ingeschat).
Kan de planning niet (genoeg) worden aangepast dan zal er met de betrokken partijen worden overlegd of het nieuwe onderdeel belangrijk genoeg is om andere delen te laten vervallen. Mocht dit niet mogelijk zijn dan zal het extra onderdeel niet worden uitgevoerd omdat het niet mogelijk is binnen de projecttijd.

Gedurende het project is de opdracht groter geworden doordat het analyse onderdeel erbij kwam. In dit geval kon de planning genoeg worden geschoven dankzij buffer tijd aan het eind van het project en door het test-proces meer tegelijk met het ontwikkelen uit te voeren.

08. INLEZEN

Omdat dit project voort bouwt op een eerder gedaan onderzoek (Middel, 2013) en nieuwe gebieden worden behandeld zoals information retrieval, heeft de bedrijfsbegeleider aangeraden om het project te beginnen met inlezen. De eerste twee onderwerpen (taxonomieën & semantisch web) zijn aangeraden door de bedrijfsbegeleider. Doordat ik het verslag van het voorgaande onderzoek al had doorgelezen, was mij bekend dat ik zou gaan werken met het KNN en Elasticsearch. Aangezien ik slechts vaag wist wat deze onderwerpen inhielden heb ik ook deze onderwerpen behandeld tijdens het inlezen. Ook kwam de interesse vanuit Bindinc in een algoritme voor het automatisch genereren van samenvattingen. De bedrijfsbegeleider had al eerder wat onderzoek naar dit onderwerp gedaan en was uitgekomen op het algoritme TextRank. Omdat ik dus ook dit algoritme zou gaan gebruiken in het project heb ik ook over dit onderwerp gelezen om zo te weten hoe het algoritme werkt. Deze onderwerpen zullen in de paragrafen van dit hoofdstuk behandeld worden.

Voor het inlezen is gebruik gemaakt van het internet waarmee verschillende websites en papers zijn doorgenomen. Ook zijn boeken doorgelezen voor bepaalde onderwerpen. Bij elk onderwerp dat hieronder wordt behandeld zal ook aangegeven worden welke literatuur hiervoor is doorgenomen. Dit zal enkel de belangrijkste literatuur inhouden, overige doorgenomen stukken zullen niet genoemd worden wanneer deze niet belangrijk genoeg zijn.

Alle literatuur die in dit hoofdstuk wordt genoemd is ook opgenomen in hoofdstuk 15.

08.1 TAXONOMIEËN

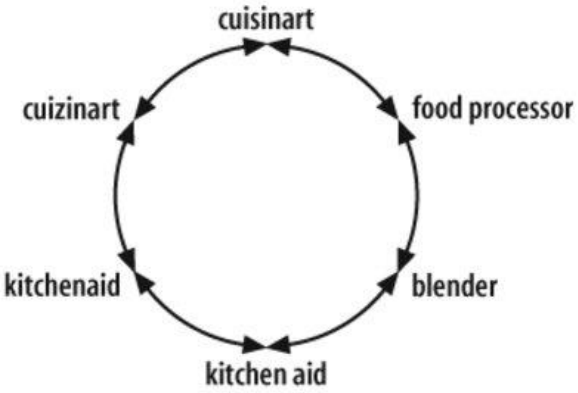
Taxonomieën is het eerste onderwerp waarover is ingelezen. Voor dit onderwerp is (Hedden, 2012) doorgelezen. Dit boek heeft mij de basisinformatie van taxonomieën duidelijk gemaakt. Deze basis staat hieronder samengevat zoals deze staat beschreven in (Hedden, 2012). Verder is ook nog meer gelezen over taxonomieën maar dit is veel samen gegaan met het onderwerp semantisch web over de bruikbare standaarden. Dit zal dan ook in paragraaf 08.2 worden behandeld.

Als er wordt gezocht naar de definitie van taxonomieën en de soorten die hierin bestaan dan kunnen andere definities gevonden worden. Dit is bekend en dus zal dit rapport de gegeven definities volgen van (Hedden, 2012).

Taxonomieën uitleg
Taxonomie is het classificeren van objecten in groepen. Er zijn vier soorten taxonomieën die gebruikt worden: “Controlled Vocabularies”, “Hierarchical Taxonomies”, “Thesauri”, “Ontologies”. Elke soort heeft zijn eigen kenmerken wat het ook voor bepaalde situaties beter geschikt maakt dan de anderen. Wat ze allemaal wel gemeen hebben is dat ze zijn opgebouwd uit termen. Hierbij is een term een woord of definitie binnen de taxonomie.

Controlled Vocabularies

Een Controlled Vocabulary (vertaald gecontroleerde vocabulaire) is een woordenlijst waarbij synoniemen zijn vastgesteld. Hierbij kunnen alle termen in de Controlled Vocabulary aan elkaar gelijk staan en kan dus elke term gekozen worden. Dit staat bekend als een synoniemring of synset want de termen kunnen weergeven worden als een ring waarbij elke term gelijk staat aan de volgende. Dit is te zien in Figuur 12. Ook kunnen de termen binnen een Controlled Vocabulary gedefinieerd worden met een “see” en “use” type. Hiermee verwijzen meerdere niet voorkeurstermen naar één voorkeursterm. Hierbij kunnen enkel de voorkeurstermen gebruikt worden om iets te classificeren. Dit wordt veel gebruikt bij zoekmachines om zo maar één term aan een onderwerp te koppelen en vervolgens alle synoniemen (niet voorkeurstermen) naar die term te herleiden. Dit wordt ook vaak gebruikt in combinatie met een andere taxonomie zoals een Thesaurus.



Figuur 12 Uitleg controller vocabualries (http://seanconnolly.ca/web/0596527349/I_0596527349_CHP_9_SECT_2.html)

Hierarchical Taxonomies

De Hierarchical Taxonomies (vertaald: Hiërarchische Taxonomieën) zijn het meest bekend als een boomstructuur. Kenmerkend van een Hierarchical Taxonomy is dat elke term meerdere onderliggende termen mag hebben maar slechts één bovenliggende term moet hebben tenzij de term de top-level term is. De structuur wordt vaak vastgelegd door bij elke term zijn broader en narrower terms bij te houden.

Thesauri

Een Thesaurus is een uitgebreidere versie van Hierarchical Taxonomy. Wat hierbij verschilt is dat termen niet beperkt zijn tot slechts één broader term. Wel is het een vereiste dat als het meerdere broader termen heeft, dat de broader termen niet met elkaar nog een broader-narrower relatie hebben. Ook kent een thesaurus related termen die een gelijke relatie aangeven. In Tabel 2 staat een voorbeeld hoe een Thesaurus term eruit kan zien.

In het voorbeeld gaat het om de term hond. Deze heeft als broader termen zoogdier en viervoetige omdat een hond onder beide valt. Als narrower termen heeft deze verschillende honden rassen. En een related term kan bijvoorbeeld een ander dier zijn zoals kat maar ook iets wat een hond te maken heeft zoals hondenvoer.

Tabel 2 Uitleg thesaurus

Term	**Hond**
Broader termen	Zoogdier, Viervoetige
Narrower termen	Pitbull, Dobermann
Related termen	Kat, Hondenvoer

Ontologies

Een Ontology (vertaald: ontologie) is een uitgebreidere versie van een Thesaurus. Het biedt de optie om niet alleen bij te houden met welke andere termen een term relaties heeft maar ook nog wat voor een relatie het is zoals: “broader-narrower”, “heeft-is van”, “is moeder van”, etc.

Software
Om goed te begrijpen hoe bedrijven omgaan met hun taxonomie zijn ook nog verschillende software pakketten hiervoor uitgeprobeerd. De pakketten die zijn uitgeprobeerd zijn MultiTes (Multisystems, 2013) en Ontology Manager van Smartlogic (Smartlogic, 2013). Bij beide pakketten ligt de nadruk op het maken en onderhouden van een Thesaurus waarbij termen gemaakt kunnen worden en hierbij kunnen broader, narrower en related termen worden toegekend. Ook beidt het de opties van een Controlled Vocabulary om voorkeurs- en niet-voorkeurstermen te maken en deze met elkaar te verbinden.

De opgedane kennis over taxonomieën heeft geholpen bij het begrijpen hoe de KBT in elkaar zit en ook was deze kennis zeer belangrijk bij het ontwikkelen van de importeerfunctie voor een thesaurus. Voor het importeren is gekozen om uit te gaan van een thesaurus. Omdat de betekenissen van relaties zoals bij een ontologie niet gebruikt worden door de algoritmes en alle taxonomieën kunnen worden omgezet naar een thesaurus wanneer de relaties van een ontologie worden vertaald naar een broader, narrower of related term.

Bronnen: (Hedden, 2012), (Multisystems, 2013), (Smartlogic, 2013)

08.2 Semantisch web

Bij het lezen over semantisch web is voornamelijk aandacht besteed aan het gebruik van taxonomieën in het web. Hierbij zijn veel verschillende standaarden gevonden zoals: RDF (Resource Description Framework), OWL (Web Ontology Language), SKOS (Simple Knowledge Organization System), FOAF (Friend Of A Friend).

RDF is als eerste opgesteld en de overige standaarden zijn op RDF gebaseerd. RDF zelf is niet speciaal gericht op taxonomieën maar op relaties aangeven tussen verschillende entiteiten. RDF doet dit door middel van expressies bekend als “triples”. Bij een triple wordt iets gezegd over het subject (onderwerp), de eigenschap die wordt benoemd heet de predicate (eigenschap) en wat de waarde is van de eigenschap is het object (onderwerp). Veel van de metadata termen worden beheerd door de Dublin Core.

Voorbeeld:
“De lucht heeft de kleur blauw.”
Het subject is *“de lucht”*, want hierover wordt iets gezegd.
Het predicate is *“de kleur”*, want deze eigenschap wordt benoemd door deze triple.
Het object is *“blauw”*, want dit is de waarde die is toegekend aan de eigenschap.

RDFs is ontworpen als een uitbreiding op RDF waarbij het klassen toevoegt. Een voorbeeld van zo een klasse is “foaf:person”. Hierdoor kan met RDFs een klassen worden gedefinieerd waarbij alle attributen gezet kunnen worden door middel van triples. Als uitbreiding op RDFs kan OWL of OWL2 gebruikt worden. Deze voegen klassen toe zoals “Example:Parent” & “Example:Woman”. Hiermee kunnen objecten gedetailleerder weergegeven worden.
FOAF is een uitbreiding die gebruik maakt van RDFs en OWL. FOAF richt zich op het omschrijven van personen en hun relaties met andere personen. Wanneer hiervan gebruik gemaakt wordt kun je je vrienden vinden maar ook de vrienden van je vrienden. Hier komt ook de naam vandaan.
SKOS is een andere uitbreiding op RDFs voor het weergeven van een taxonomie. Hiermee lijkt het veel op OWL maar is toch met een ander doel ontwikkeld. Want OWL is een stuk uitgebreider maar daarmee

ook een stuk formeler wat het gemakkelijker maakt om een taxonomie weer te geven door middel van SKOS. SKOS volgt overigens de standaarden van ISO 25964-1 wat de voorgaande thesaurus standaarden ISO 2788 en ISO 5964 vervangt.

Voor het bevragen van RDF en de daarop gebouwde standaarden is SPARQL (SPARQL Protocol And RDF Query Langauge) ontworpen. Hiermee kunnen de triples uit een RDF opgevraagd worden maar het biedt hiernaast ook de mogelijkheid om queries uit te voeren op de standaarden.

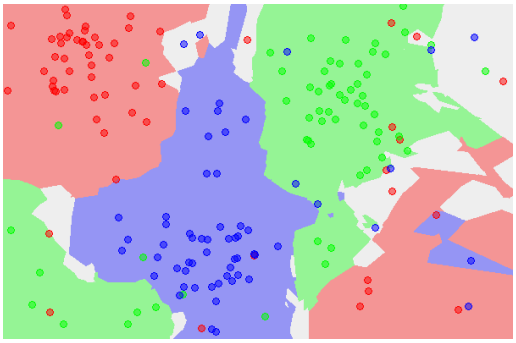
Er is voor gekozen om de standaarden direct te ondersteunen omdat bleek dat er veel verschillende standaarden zijn. In deze paragraaf zijn niet alle standaarden genoemd, zo bleek dat niet alle geteste software beschreven in paragraaf 08.1 kan exporteren naar RDF en de aangeleverde KBT is niet in RDF-formaat. In plaats daarvan is ervoor gekozen om een importeerfunctie te schrijven voor een thesaurus.

Bronnen: (Allemang & Hendler, 2011), (Metadata standards - Wikipedia, 2013), (Resource Description Framework - Wikipedia, 2013), (RDF Schema - Wikipedia, 2013), (Web Ontology Language - Wikipedia, 2013), (Isaac, 2011), (Tester, 2013), (LinkedDataTools, 2013), (EBU, 2013), (W3C, 2013)

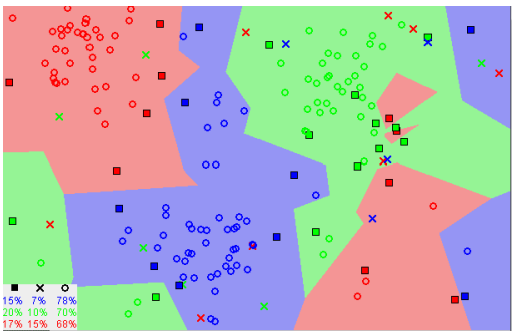
08.3 KNN & CNN

Omdat het al zeker was dat KNN gebruikt ging worden is dit onderwerp ook geselecteerd voor het inlezen. Tijdens het lezen over KNN kwam ik het onderwerp CNN tegen omdat de uitleg van KNN vaak wordt gecombineerd met een uitleg over CNN. Voor de uitleg van KNN zie hoofdstuk 06.1.

CNN (Condensed Nearest Neighbors) is een manier om de performance van KNN te verbeteren. Bij CNN wordt KNN ingesteld op K = 1 en worden alle punten geëlimineerd die niet van belang zijn voor de classificatie. Bij Figuur 13 is te zien hoe de classificaties worden verdeeld over gebieden wanneer KNN gebruikt wordt met K=5. Hierbij is te zien dat veel punten met dezelfde classificatie dicht bij elkaar liggen. De middelste groep hiervan is dus niet nodig wanneer K=1 wordt gebruikt want dan zal één van de buitenste het dichtst bij zijn. In Figuur 14 is te zien hoe de velden verdeeld worden wanneer CNN gebruikt wordt. Hierbij zijn de vierkanten de punten die blijven staan en die nieuwe punten zullen classificeren. De kruisjes stellen ruis voor die ook door CNN geëlimineerd zijn. En de rondjes zijn alle punten die niet meer nodig zijn dankzij CNN. Door CNN te gebruiken zijn nog 15% tot 20% van de originele punten nodig.



Figuur 13 CNN uitleg 1 (http://en.wikipedia.org/wiki/File:Map5NN.png)



Figuur 14 CNN uitleg 2 (http://en.wikipedia.org/wiki/File:Map1NNReducedDataSet.png)

CNN was een interessant onderwerp omdat het al bekend was dat er gewerkt zal worden met een grotere dataset dan bij het voorgaande onderzoek is gedaan. Hierdoor zou de performance van KNN een issue kunnen worden. CNN is hiervoor een mogelijke oplossing. Uiteindelijk bleek KNN al goed te presteren ook al wordt een grote dataset gebruikt en dus is CNN niet toegepast in het project.

Bronnen: (k-nearest neighbors algortihm - Wikipedia, the free encyclopedia, 2013), (Sutton, 2012), (Cheng, Tan, & Jin, 2007), (Mirkes, 2013), (He & Niyogi, 2003), (Chou,, Kuo, & Chang)

08.4 ELASTICSEARCH

Aan de hand van het voorgaande onderzoek was bekend dat met Elasticsearch gewerkt gaat worden. Dit zal gebruikt worden bij het KNN algoritme. De uitleg van Elasticsearch is al behandeld in hoofdstuk 06.1. Bij het inlezen over Elasticsearch is in eerste instantie gericht op het gebruik van Elasticserach. Hiervoor kunnen queries worden opgebouwd in JSon en zullen de resultaten ook geretourneerd worden in een JSon object. Later is ook gelezen over de werking van Elasticsearch zoals het gebruikt wordt voor KNN.

Bronnen: (Elasticsearch - Wikipedia, the free encyclopedia, 2013), (The Apache Software Foundation, 2013)

08.5 TEXTRANK

Zodra bekend was dat TextRank ook gebruikt zou worden in dit project heb ik hierover ook ingelezen. Doordat ik weet hoe TextRank werkt is het gemakkelijker uit te leggen wanneer hierover gesproken wordt met collega's en ook is beter in te schatten hoe het zal werken met een bepaald soort documenten. Zo kan het voorkomen dat TextRank geen samenvatting kan maken van minder dan 50 woorden wanneer de belangrijkste zin (de zin met de hoogste score) langer is dan dat aantal. Voor de uitleg van het algoritme TextRank zie hoofdstuk 06.2.

Bronnen: (Mihalcea & Tarau, 2004), (Storojev, 2010)

09. REQUIREMENTS

Na het opstellen van de eerste versie van het plan van aanpak was de opdracht een stuk duidelijker geworden en ben ik begonnen aan het opstellen van de requirements. Dit omvat de eisen waaraan de applicatie zal voldoen, de prioritering van de eisen, de afbakening van de eisen en de user stories.

Op basis van de opdrachtomschrijving uit het plan van aanpak is een eerste set aan eisen opgesteld. Deze eisen zijn eerst zonder prioritering opgesteld en onderverdeeld in functionele en non-functionele eisen. Hieronder staan een aantal voorbeelden van opgestelde functionele eisen. De codes zoals F16 & F07 zijn een nummering voor Functionele eis 16 en Functionele eis 07. Deze codes dienen enkel om eenvoudig naar een eis te kunnen verwijzen. Dit is bijvoorbeeld gedaan bij de acceptatietest.

- F16 Het systeem moet de keywords[2] en documenten kunnen importeren.
- F07 Bij elk document zal de gebruiker X keywords zien die door een geïmplementeerd algoritme worden gesuggereerd. Per enquête kan de X verschillen.
  Voor de gehele enquête zal de X hetzelfde zijn.
- F08 Bij elk keyword kan de gebruiker één van de volgende scores aanklikken: “Zeer goed, Goed, Matig, Slecht, Zeer slecht”.
- F11 De antwoorden van de gebruiker zullen in een database worden opgeslagen.

De opgestelde eisen zijn besproken met de opdracht met de opdrachtgever waarna deze zijn bijgewerkt en geprioriteerd. Voor het prioriteren van de requirements zijn de volgende scores gebruikt:

- (C) Critical
- (H) High
- (M) Medium
- (L) Low

Vervolgens zijn user stories opgesteld zodat hiermee met SCRUM gewerkt kan worden. Hieronder zijn een aantal voorbeelden weergegeven van de opgestelde user stories:

- Als gebruiker wil ik gevonden keywords voor een document kunnen beoordelen.
- Als gebruiker wil ik een eigen suggesties voor keywords kunnen geven.

Nogmaals zijn de eisen en user stories met de opdrachtgever doorgenomen waarna een contactmoment was ingepland met een stakeholder. De stakeholder is een contactpersoon van de klant Kluwer.
Het belangrijkste voor deze stakeholder is een advies of KNN ingebouwd kan worden in de Kluwer Creator om trefwoorden te suggereren. Hiervoor zullen een aantal redacteuren van Kluwer de te ontwikkelen applicatie gebruiken om KNN te beoordelen. Hierom zal het voor de stakeholder ook belangrijk zijn dat de te ontwikkelen tool eenvoudig te gebruiken is voor de redacteuren van Kluwer.

Na het gesprek met de stakeholder zijn de eisen aangescherpt voor zover nodig en zijn onderdelen verduidelijkt of uitgelegd. Na de verbeteringen te hebben doorgevoerd zijn deze wederom eerst besproken met de opdrachtgever waarna ze ook naar de stakeholder gestuurd zijn via mail. Na mailcontact waren de opgestelde eisen redelijk snel duidelijk voor de betrokkenen en goedgekeurd.

[2] Een keyword is een suggestie van een algoritme bijvoorbeeld een trefwoord voor metadatering of een samenvatting

10. MODELLERING

Bij het modelleren is voornamelijk gericht op klassendiagrammen. Ook al schrijft SCRUM geen modellering voor, er is hier toch voor gekozen, omdat het belangrijk is dat met de applicatie verschillende algoritmes getest kunnen worden. Hiervoor moet het dan ook mogelijk zijn om verschillende algoritmes te implementeren. Zonder eerst een ontwerp te maken voor de applicatie zal dit niet lukken.

Bij de klassendiagrammen zijn alleen de belangrijke attributen en/of methoden weergeven om het diagram leesbaar te houden. Zo zijn de constructors van klassen niet weergegeven wanneer deze geen parameters vraagt. En niet alle attributen van de Nodes zijn opgenomen.

Er zijn drie klassendiagrammen opgesteld, één voor de importeerfuncties, één voor het enquête-onderdeel en één voor de database. De importeerfuncties en het enquête-onderdeel zijn geschreven in C# .NET gebruikmakend van het framework Lynkx. Door het Lynkx framework te gebruiken is het eenvoudig om de content te beheren via snel op te stellen user interfaces en ook wordt communiceren met de database makkelijker gemaakt. Omdat het Lynkx framework een belangrijk onderdeel is, zal dit framework in paragraaf 10.1 worden uitgelegd. De database is een MSSQL database, ook voor de database is het Lynkx framework belangrijk omdat veel informatie wordt opgeslagen in Nodes. Wat deze Nodes inhouden zal ook worden uitgelegd in paragraaf 10.1.

De importeerfuncties zijn verantwoordelijk voor het importeren van de documenten en mogelijk andere gegevens zoals de trefwoorden uit de KBT. Deze importeerfuncties zijn belangrijk omdat de algoritmes die getest worden, werken aan de hand van een tekst. Om de teksten te kunnen gebruiken worden deze geïmporteerd naar een database. Bij dit importeren kan ook meer geïmporteerd worden dan alleen de tekst van het artikel. Zo zijn voor de artikelen van Kluwer ook metadata geïmporteerd. Voor het importeren van de metadata is eerst de KBT geïmporteerd. Deze KBT is een thesaurus waarvoor de optie gebruikt is voor het importeren van een taxonomie. De artikelen van Kluwer die geïmporteerd zijn hebben verwijzingen naar trefwoorden uit de KBT. Deze verwijzingen naar de trefwoorden dienen als classificatie van de documenten en zijn ook geïmporteerd. De trefwoorden van documenten worden door KNN gebruikt om voor een ander document de trefwoord-suggesties te doen.
Echter moet het mogelijk zijn om meerdere soorten taxonomieën te importeren en niet alleen gericht op de KBT. Ook de documenten zelf kunnen in verschillende formaten zijn zoals platte tekst of XML. Hoe dit is aangepakt staat verder beschreven in paragraaf 10.2.

Het enquête-onderdeel communiceert met de database via het Lynkx framework. Uit de database wordt de informatie opgehaald voor de enquête. Dit houdt in welke documenten in de enquête gebruikt worden, welk algoritme gebruikt wordt met mogelijke instellingen van het algoritme en de al gegeven beoordelingen van de gebruiker. Ook zijn de algoritmes in dit onderdeel geïmplementeerd en worden de suggesties van het geselecteerde algoritme opgevraagd van een document. De suggesties worden getoond waarna de gebruiker deze kan beoordelen, de beoordelingen van de gebruiker zullen ook weer worden opgeslagen.

Bij het klassendiagram van de database zal te zien zijn dat de Nodes van Lynkx framework veel gebruikt worden. Hoe deze Nodes werken is uitgelegd in paragraaf 10.1.

Naast de klassendiagrammen zijn ook twee sequencediagrammen opgesteld. Deze waren niet van tevoren gepland maar zijn opgesteld wanneer de structuur voor een te ontwikkelen onderdeel nog niet duidelijk was. Deze zijn later niet bijgewerkt omdat deze er enkel voor dienden om tot een idee te komen voor de structuur van een onderdeel van de applicatie.

Aan het einde van het project zijn voor de testen procesdiagrammen opgesteld. Deze zijn opgesteld aan de hand van de al ontwikkelde code. Deze diagrammen volgen de code niet letterlijk maar alleen de delen waarop de testen gericht zijn. Deze diagrammen zullen worden besproken in hoofdstuk 12.

10.1 LYNKX FRAMEWORK

Lynkx is een framework met ingebouwd CMS ontwikkeld door Liones. Dit framework is onder te verdelen in drie onderdelen. Het script onderdeel, de database connectie met de Nodes en het CMS onderdeel. Deze onderdelen zullen in deze paragraaf worden behandeld. Hierbij zal gericht worden op wat is gebruikt tijdens het project.

Lynkx script

Met het Lynkx framework kan gescript worden in een XML-achtige structuur wat wordt opgeslagen in een .lynkx bestand, een voorbeeld hiervan is te zien in tekstvak 2. Door middel van de tags kunnen een hoop standaardfuncties eenvoudig worden uitgevoerd zoals het ophalen van informatie uit de database en over deze informatie loopen. Aan de tags kunnen attributen worden meegegeven om bijvoorbeeld bij het ophalen aan te geven welk type de Nodes moeten hebben die worden opgehaald. De opgehaalde informatie is vervolgens beschikbaar tussen de tags, dus vanaf <retrieveNodeset type="SurveyResult"> tot en met </retrieveNodeset>.

```
<retrieveNodeset level="all" type="SurveyResult">
  <loopNodeset>
    <callProcedure name="renderDetails" />
  </loopNodeset>
</retrieveChildNodeset>
```

Tekstvak 2 Lynkx tags voorbeeld

Wanneer een optie nodig is die niet beschikbaar is als tag dan kan een ScriptTag worden aangemaakt. Als voorbeeld zal ik de tag <test> nemen. Hiervoor maken we een klasse TestTag die overerft van ScriptTag en zich bevindt in de namespace Lynbkx.Application.Tags. De klasse wordt geschreven in C# en overschrijft de methode Execute(ScriptContext context), deze methode wordt uitgevoerd wanneer de tag wordt gebruikt en zal de context meegegeven. In de context bevindt zich wat ervoor is opgehaald zoals een Nodeset uit de database. Wanneer de tag <test> onder de tag <loopNodeset> staat dan zal de loopNodeset door de nodeset heen loopen. Vanuit de tag <test> zijn dan de nodes uit de nodeset beschikbaar.

```
<if condition="{IsFalse(IsEmpty(Get.id))}">
  <retrieveNode type="SurveyResult" guid="{Get.id}">
    <callProcedure name="renderDetails" />
  </retrieveNode>
</if>
```

Tekstvak 3 Lynkx function voorbeeld

Naast de tags zijn ook nog functions beschikbaar. Een voorbeeld van zo'n function is IsFalse() zoals te zien in tekstvak 3. Deze functie wordt uitgevoerd en het resultaat is beschikbaar in de tag. Ook deze

functions kunnen worden toegevoegd door in C# een klasse aan te maken binnen de namespace Lynkx.Application.Functions. Wanneer de function wordt gebruikt dan zal de methode Evaluate worden aangeroepen en alle aanwezige attributen zullen worden meegegeven.

```
<declareProcedure name="renderDetails"><!--Survey result-->
  <DetailAnalysis resultId="{Node.id}" target="surveyResult">
    <renderSection name="detailsHead">
      <loopNodeset source="surveyResult.documentResults" target="documentResult">
        <callProcedure name="renderDocumentResult" />
      </loopNodeset>
    </renderSection>
  </DetailAnalysis>
</declareProcedure>
```

Tekstvak 4 Lynkx procedure voorbeeld

```
<template:section name="detailsHead" field="content">
    <fieldset>
        <legend>{surveyResult.name}</legend>
        {detailDocumentResult}
    </fieldset>
</template:section>
```

Tekstvak 5 Lynkx Section voorbeeld

Ook kunnen een soort methodes worden aangemaakt door middel van <declareProcedure name=”renderDetails”> zoals te zien in tekstvak 4. En op een zelfde soort manier wordt de link gelegd naar de grafische weergave door middel van sections zoals te zien in tekstvak 5. De weergave is weergegeven in HTML, hierin kunnen placeholders geplaatst worden door middel van {surveyResult.name}. In de placeholder zal de informatie worden geplaatst wanneer deze aanwezig is. Ook is het mogelijk om op de plaats van de placeholder een volgende HTML section te plaatsen.

Voor de beschikbare tags en functions is de documentatie van Lynkx gebruikt (Liones, 2013).

Nodes

Lynkx werkt veel met Nodes om informatie op te slaan in de database. Hierbij wordt alles wat wordt opgeslagen als een Node in de tabel Nodes opgeslagen. Een voorbeeld van een record uit deze tabel is te vinden in tabel 3. Een Node heeft een aantal standaard attributen zoals id, name, type, body & description. Van deze velden is name en type verplicht om in te vullen en het id en guid worden automatisch gevuld. Het id is een oplopende integer en de guid is een unieke string. De guid wordt gebruikt wanneer data van database kan wisselen, wanneer dit gebeurt kunnen nieuwe id’s nodig zijn omdat deze niet meer uniek maar de guid zal wel uniek blijven. Ook moet een parent Node worden opgegeven bij het aanmaken van een Node. Hierdoor zal een node (met uitzondering van de eerste) altijd een parent hebben en hiermee worden de Nodes ook weergegeven in een boomstructuur in het CMS.

Wanneer een Node meer informatie moet bevatten dan kan dit bij het opslaan van de node worden meegegeven in de vorm van key value waarna het geserialiseerd wordt naar het veld extendedProperties en wanneer de Node weer wordt opgehaald dan zal de informatie van het extendedProperties veld ook weer automatisch gedeserialiseerd worden. Voor het ophalen kunnen queries worden uitgevoerd door middel van een NodesQuery(). Hieraan kunnen meerdere attributen meegegeven worden om eenvoudig informatie op te halen uit de database zonder SQL te hoeven

schrijven. Wanneer de attributen niet genoeg optie bieden dan kan ook een string SQL als attribuut worden meegegeven als query.

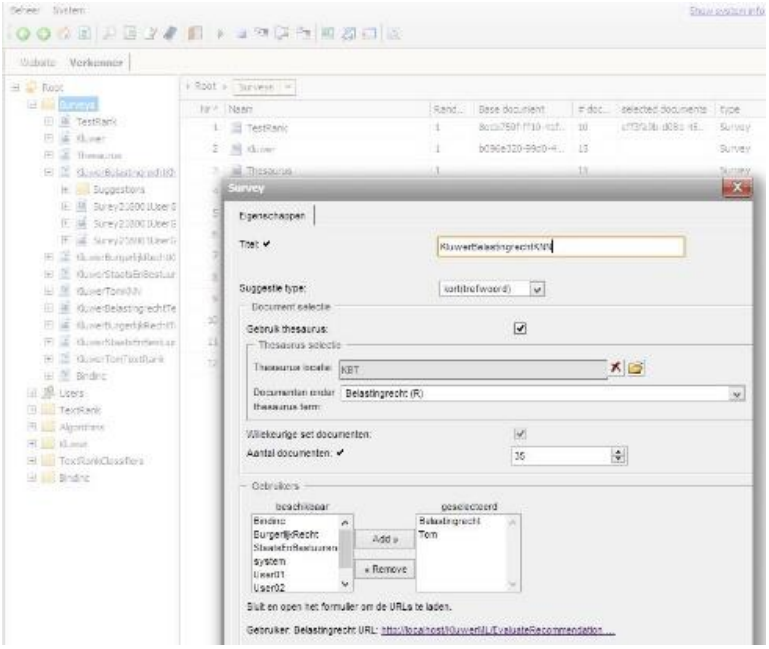
Veld	**waarde**
id	62285
guid	BA7DA5FB-0521-4B86-BC36-B2054DA7490A
key	101#12877
name	180-dagenregeling
type	ThesaurusTerm
subType	NULL
description	
body	180-dagenregeling
extendedProperties	0x120000000300000100020003030004000500060272740 96B6C757765724B65790262740536363037380531323837370537323330300 0

Tabel 3 Voorbeeld Node record

Om te zoeken op de velden binnen de extendedProperties kunnen types worden aangemaakt. Een type is een vereist veld voor elke Node en aan de hand van het type wordt herkend of het bepaalde attributen heeft.

CMS

Het CMS onderdeel is zo ontwikkeld dat klanten van Liones zelf de content van hun website/webapplicatie kunnen bijhouden. In figuur 15 is te zien hoe het CMS onderdeel van Lynkx eruit ziet. Bij dit project is het echter gebruikt om tijdens het ontwikkelen te controleren of gegevens correct werden opgeslagen en voor het aanmaken van een enquête. Het aanmaken van een enquête zal verder worden besproken in hoofdstuk 11.7. Zoals in figuur 15 is te zien kunnen gegevens worden aangepast door middel van formulieren. Deze formulieren kunnen worden aangemaakt voor een aangemaakt type van Node. Hierbij kan dan worden aangegeven welke attributen beschikbaar moeten zijn in het formulier en in wat voor een soort veld de informatie getoond moet worden.



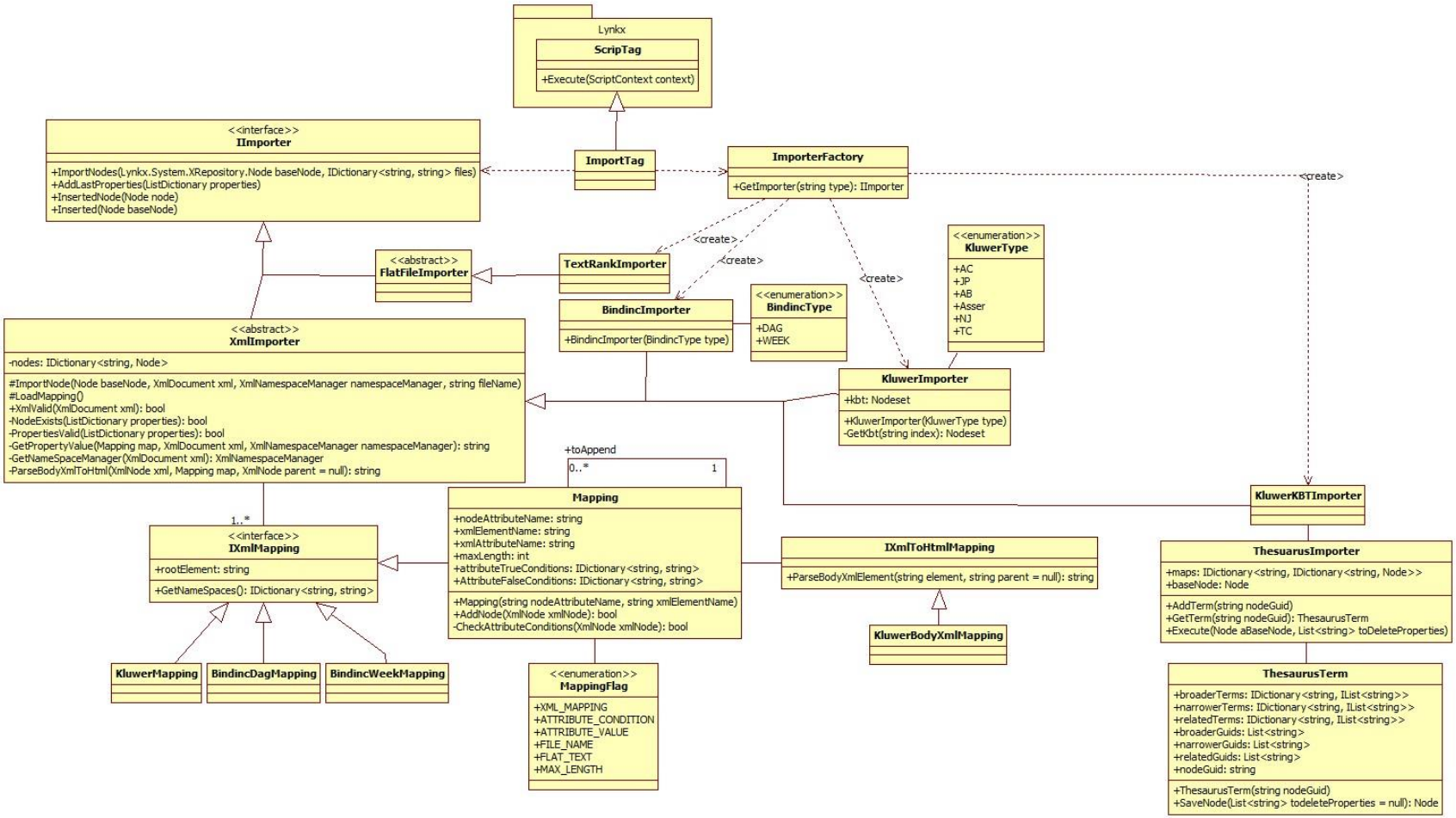
Figuur 15 CMS Lynkx

10.2 IMPORTEERFUNCTIES KLASSENDIAGRAM

In figuur 16 is het klassendiagram te zien van de importeerfuncties. Voor de importeerfuncties is een factory pattern gebruikt om zo de gewenste implementatie aan te roepen vanaf de ImportTag. De ImportTag krijgt als attributen mee welk type bestanden geïmporteerd moet worden, in welke map de bestanden zich bevinden en het type Importer dat gebruikt moet worden. Een IImporter implementatie zoals de KluwerImporter maakt gebruik van de FlatFileImporter of de XmlImporter. De FlatFileImporter is zeer simpel en importeert de inhoud van het document en gebruikt hierbij de naam van bestand als naam voor de Node. Bij de XMLImporter is een mapping waarmee wordt bepaald welke XML elementen worden geïmporteerd naar welke velden van de Node. Door middel van vlaggen kunnen verschillende opties worden ingesteld bij een mapping.

Voor het importeren van een taxonomie zoals de KBT is de ThesaurusImporter toegevoegd. Deze kan gebruikt worden na een andere IImporter om de geïmporteerde content om te zetten naar een thesaurus waarbij de broader, narrower en related termen refereren naar de id's van andere content.

Achteraf bekeken had ik het liever ontwikkeld met het pipes and filters pattern. Hiermee zou er een apart stuk zijn voor uitlezen van de bestanden zelf wat nu ook wordt gedaan door de FlatFileImporter en de XmlImporter. Deze klassen zouden de content van de bestanden omzetten naar objecten waarna hiermee verdere mutaties gedaan kunnen worden zoals wat de ThesaurusImporter doet. En als laatste zullen de objecten worden weggeschreven naar de database. Hierdoor zou er minder connectie met de database nodig zijn want nu wordt voor een thesaurus importeren vaker connectie gelegd met de database dan nodig.
Dit is uiteindelijk niet alsnog hiernaar omgebouwd omdat dit idee pas later in het project kwam en dit paste niet in de planning.



Figuur 16 Klassendiagram importeerfuncties

10.3 ENQUÊTE-ONDERDEEL KLASSENDIAGRAM

In figuur 17 staat het klassendiagram voor het enquête-onderdeel.

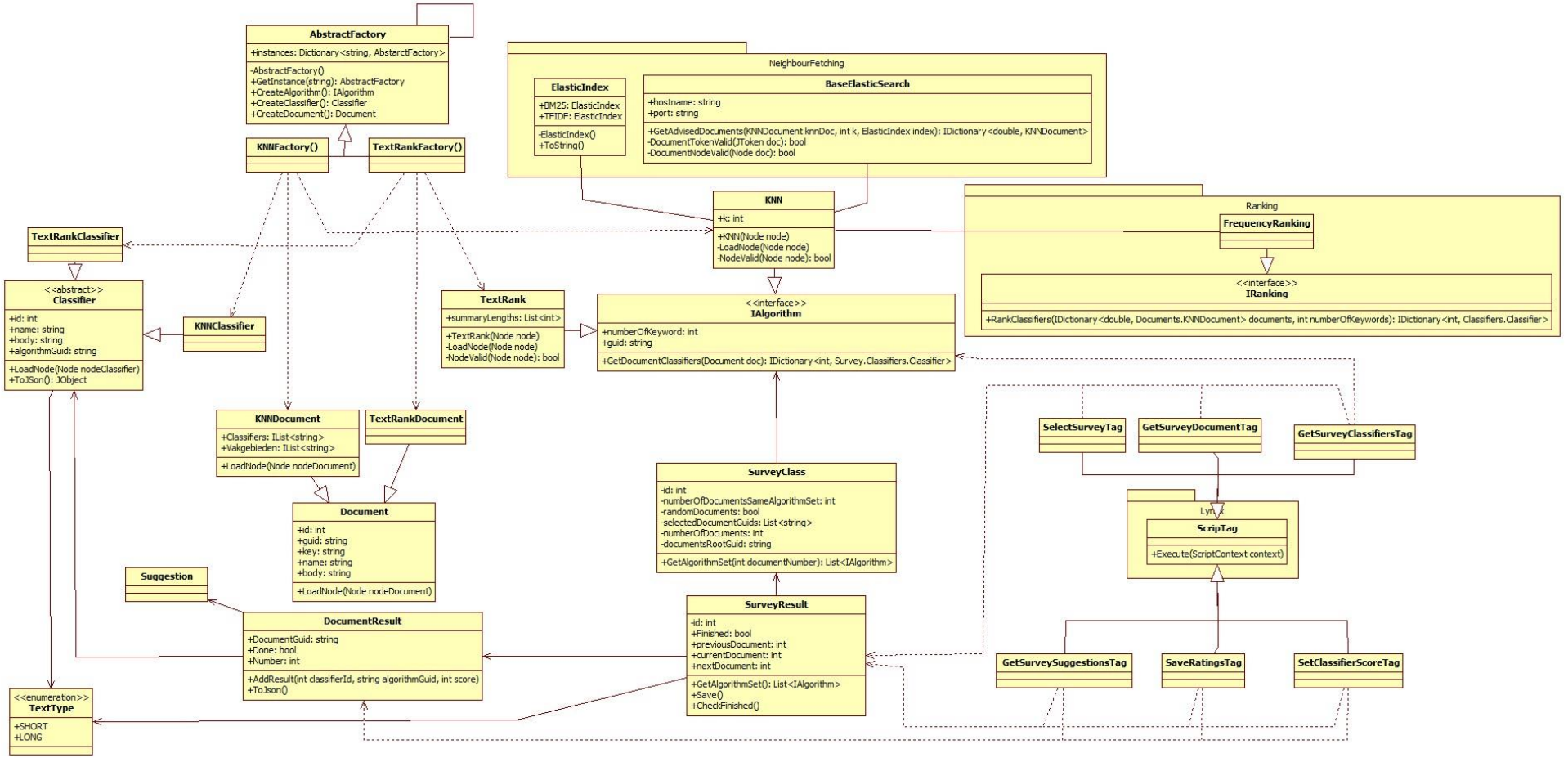
Om verschillende algoritmes te kunnen gebruiken is gebruik gemaakt van een abstract factory pattern. Een extra algoritme kan worden toegevoegd door nog een implementatie van AbstractFactory toe te voegen. Achter de implementatie van het IAlgorithm zit het algoritme zelf. In het diagram is te zien dat KNN een deel NeighbourFetching en Ranking heeft. Het NeighbourFetching is verantwoordelijk voor het zoeken van de lijkende documenten en het Ranking is verantwoordelijk voor het ophalen van de trefwoorden. Verder heeft elke implementatie een versie van Document en Classifier. In een aantal gevallen zullen deze niets overschrijven zoals TextRank, want dit algoritme heeft een eenvoudig document nodig. En de Classifier is voor TextRank de samenvatting waarvoor de body wordt gebruikt van een Classifier. Voor KNN is wel het Document overschreven omdat hierbij de al toegewezen trefwoorden (Classifiers) nodig zijn.

Vanaf de ScriptTags wordt de informatie voor de gebruiker opgehaald en ook worden deze gebruikt om de resultaten op te slaan. Hiervoor hebben de ScriptTags zelf weinig intelligentie en gebeurt het meeste bij de klassen: SurveyClass, SurveyResult en DocumentResult.

SurveyClass houdt de informatie bij van de enquête. Dus welke documenten voor de enquête zijn geselecteerd of de opties waarmee de documentenset opgesteld kan worden en welke algoritmes gebruikt worden.

SurveyResult is het resultaat van de enquête en heeft meerdere DocumentResults waarin de resultaten staan voor een document. De lijst met DocumentResults wordt ook gebruikt om het vorige, volgende en huidige document op te halen. SurveyResult maakt gebruik van de SurveyClass om de lijst van DocumentResults op te stellen wanneer deze nog niet gevuld is en om de te gebruiken algoritmes op te halen.

Een DocumentResult is het resultaat van een document en houdt bij welke classifiers zijn gesuggereerd, welke suggesties de gebruiker heeft toegevoegd en overig commentaar dat de gebruiker heeft toegevoegd. De Classifiers zelf houden bij welke score de gebruiker heeft gegeven en door welk algoritme de Classifier is gesuggereerd.



Figuur 17 Klassendiagram enquête-onderdeel

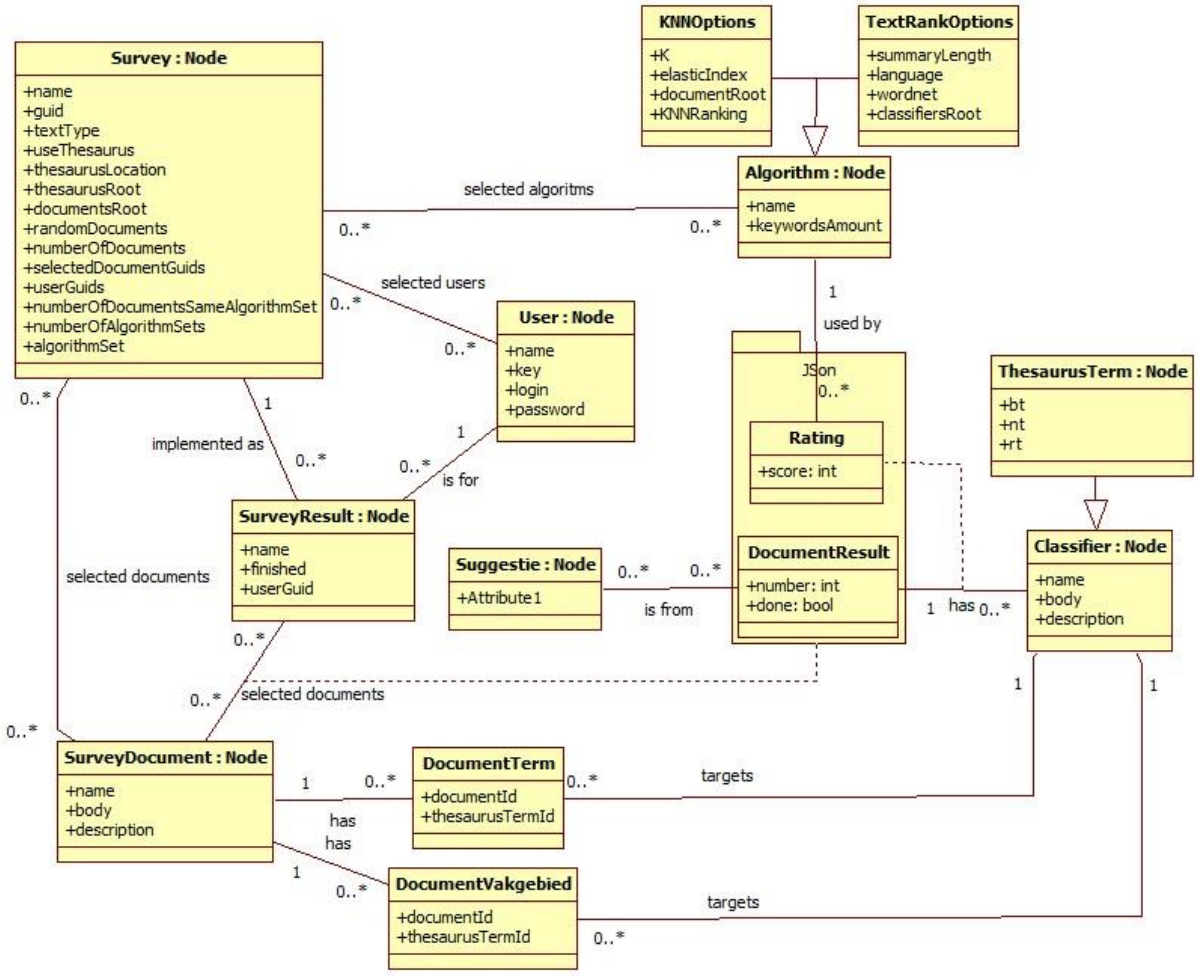
10.4 DATABASE KLASSENDIAGRAM

In figuur 18 is het klassendiagram van de database te zien. De klassen die zijn ondergebracht in een Node zijn gemarkeerd met ":Node". Hieronder zullen de belangrijkste klassen worden toegelicht.

Survey
Om een enquête aan te maken wordt een Survey aangemaakt. Deze omvat de data wat de enquête in zal houden, dit zijn drie onderdelen: welke documenten gebruikt worden, voor welke gebruikers de enquête is en welke algoritmes gebruikt zullen worden.
Voor het bepalen van de documenten wordt bepaald of de documenten geselecteerd moeten worden op basis van een thesaurus (useThesaurus). Als dit het geval is zal de locatie van thesaurus worden gevraagd onder welke node bevindt de thesaurus zich (thesaurusLocation). Vervolgens zullen alle termen getoond worden van de thesaurus waarna de gebruiker de gewenste term moet selecteren. Bij het gebruik van een thesaurus zullen de documenten willekeurig geselecteerd worden omdat het niet goed mogelijk bleek om alle documenten te tonen wegens performance problemen. De gebruiker zal kunnen ingeven hoeveel documenten geselecteerd moeten worden waarna dat aantal documenten geselecteerd zal worden die zich onder de geselecteerde thesaurusterm bevinden.
Wanneer de thesaurus niet gebruikt wordt moet de gebruiker de node opgeven waaronder de SurveyDocument nodes liggen die gebruikt moet worden. Hierbij kan gekozen worden of een willekeurig aantal documenten gebruikt moet worden of de gebruiker kan de te gebruiken documenten zelf selecteren.
De gebruikers selecteren is een stuk eenvoudiger. Hierbij zullen alle users getoond worden waarna de gebruiker de users kan selecteren die de enquête gaan invullen. Voor elke geselecteerde user zal daarna een URL gemaakt worden waarmee de enquête voor die user geopend kan worden.
Voor de selectie van algoritmes wordt bijgehouden hoeveel verschillende algoritme sets gebruikt moeten worden en om de hoeveel documenten er van set gewisseld moet worden. Hierdoor kan een enquête zo worden opgesteld dat bij de eerste twee documenten KNN gebruikt, de volgende twee TextRank en daarna weer KNN.

SurveyResult
Dit is de invulling van de Survey voor een specifiek user. Hierin zal bijgehouden worden om welke enquête het gaat, welke user deze heeft ingevuld, welke documenten in de enquête zitten en of de gebruiker klaar is met invullen en de resultaten. Een SurveyResult heeft een JSon object waarin de resultaten van de enquête zijn opgeslagen waarin wordt bijgehouden welke Classifiers zijn gesuggereerd door welk algoritme, voor welk document de classifiers zijn gevonden en welke score de gebruiker heeft gegeven aan de classifiers.

SurveyDocument
Dit is een document dat gebruikt kan worden in de enquête. De algoritmes zullen eerst proberen om de description te gebruiken en als deze niet is ingevuld zullen ze terug vallen op de body. Hiervoor is gekozen omdat bij het importeren de description is gevuld met content van het document zonder opmaak en de body is gevuld met de content maar hier zit ook HTML opmaak bij. Een document kan ook nog gekoppeld zijn aan termen door middel van DocumentTerm en DocumentVakgebied.
DocumentTerm wordt gebruikt door het KNN algoritme om de trefwoorden van een document op te halen. DocumentVakgebied wordt gebruikt bij het selecteren van documenten vanuit een thesaurus.



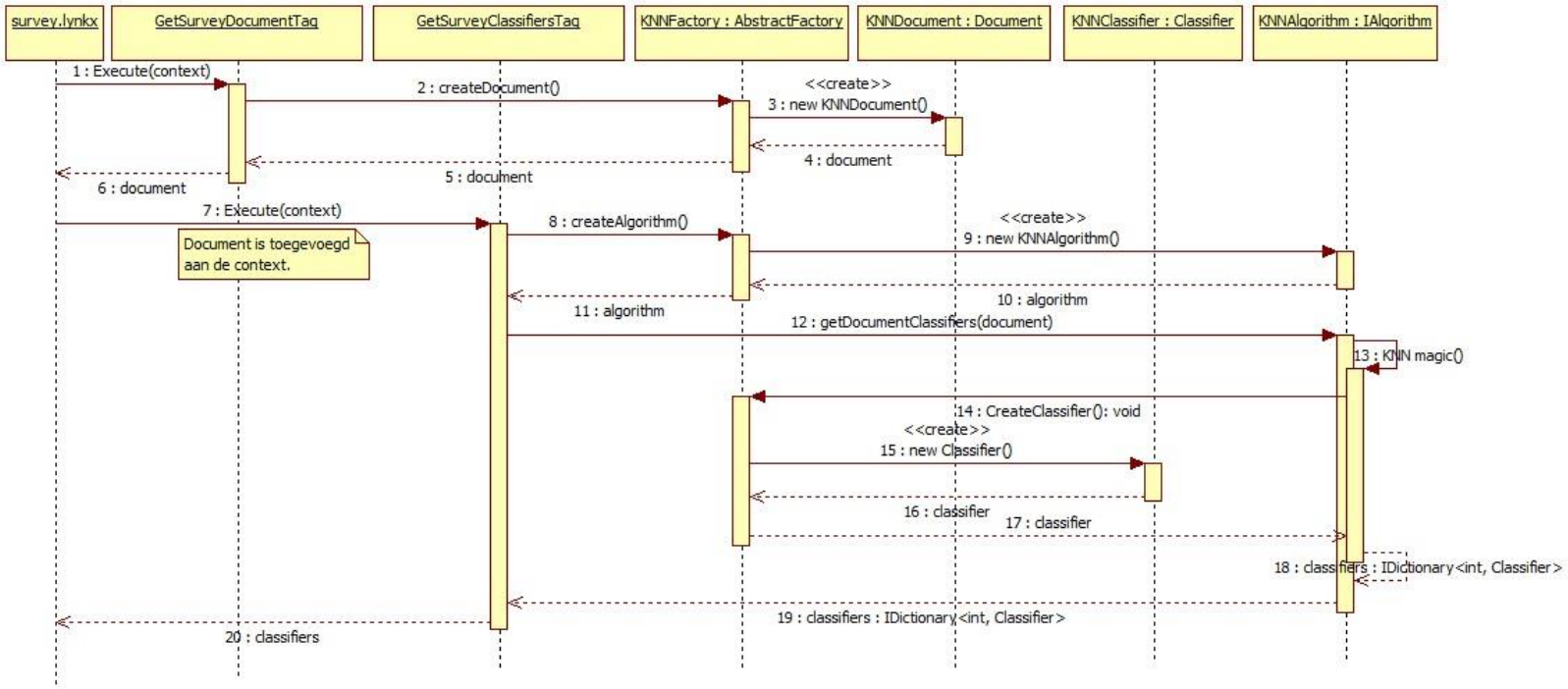
Figuur 18 Database klassendiagram

10.5 SEQUENCEDIAGRAMMEN

Tijdens het project zijn twee sequencediagrammen opgesteld. Deze zijn opgesteld als hulpmiddel tijdens het uitdenken van deze applicatie-onderdelen.

KNN sequencediagram

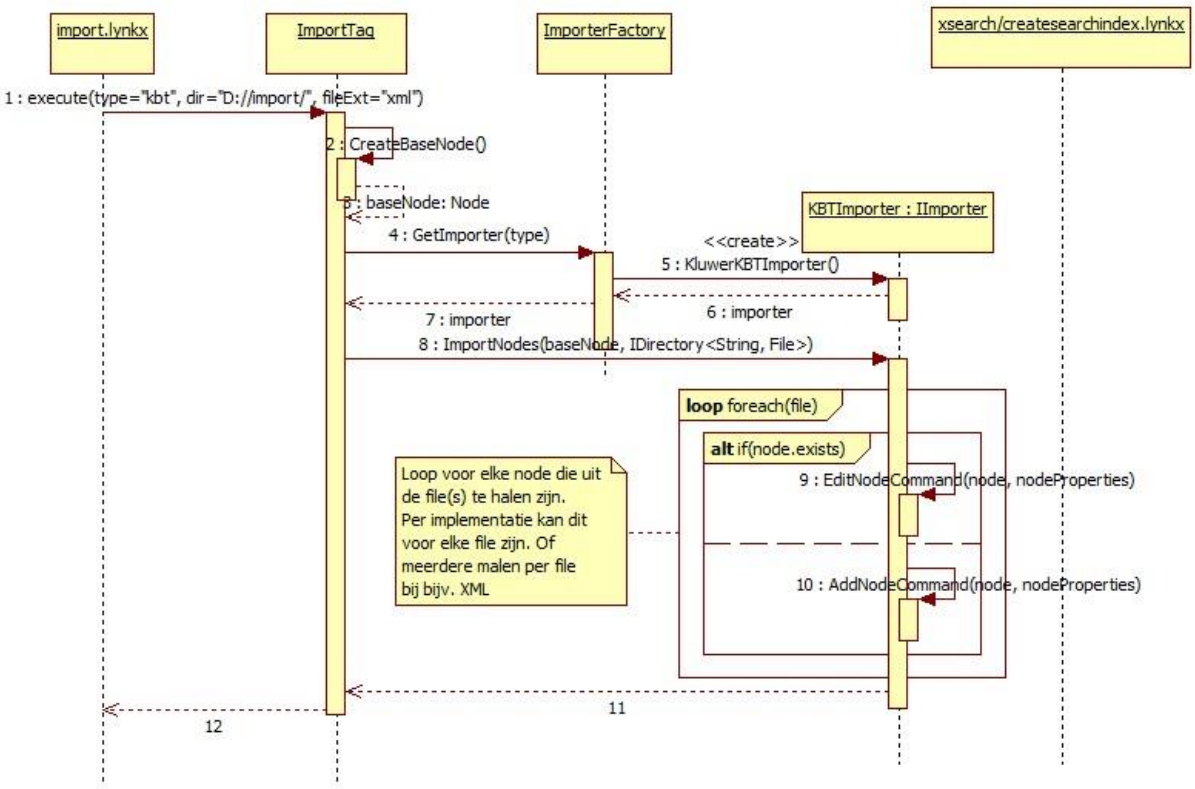
Voor het gebruiken van algoritmes is een sequencediagram opgesteld. Hierbij het is in het diagram het algoritme KNN als voorbeeld genomen. Het gaat hierbij enkel op de manier waarop een algoritme aangeroepen zal worden en niet op de werking van het algoritme. De documenten, classifiers en het algoritme zullen worden aangemaakt door de KNNFactory. Hierbij zal het document worden aangemaakt vanuit de GetSurveyDocumentTag wat aangeroepen wordt vanuit de GUI. Het algoritme zal vervolgens worden aangeroepen door de GetSurveyClassifiersTag en het algoritme zal objecten laten aanmaken voor de gevonden classifiers. Omdat het bij dit sequencediagram niet gaat om de werking van KNN is dit aangeduid met “KNN magic()”. Wel zal het een Dictionary retourneren met de classifiers en de volgorde waarin deze getoond moeten worden, net zoals elk algoritme zal doen.



Figuur 19 KNN sequencediagram

Import sequencediagram

Voor het importeren is de KBTImporter genomen als voorbeeld. Eerst zal de basis node worden opgehaald of aangemaakt wanneer deze nog niet bestaat. Onder de basis node zullen alle nieuwe nodes geplaatst worden. Vervolgens wordt de juiste importer opgehaald door middel van de ImporterFactory. Wat niet in het diagram staat is het uitlezen van de bestanden. Voor het uitlezen en importeren van de bestanden is later batch verwerking ingebouwd. De KBTImporter zal vervolgens de bestanden importeren die worden meegegeven met de methode ImportNodes. Bij het importeren zullen de bestaande nodes zullen worden ge-update en nieuwe zullen worden aangemaakt. Verdere specificaties voor de KBTImporter zoals het verwerken van de thesaurus staan hier niet in aangegeven omdat de focus niet ligt op de specifieke KBTImporter maar het algemeen gebruik van importers.



Figuur 20 Import sequencediagram

11. ONTWIKKELING

In dit hoofdstuk is de ontwikkeling opgesplitst in verschillende onderdelen die met elkaar te maken hebben. Omdat is gewerkt met de ontwikkelmethode SCRUM is bij elk onderdeel aangegeven in welke sprint dit is ontwikkeld. Wel staan de paragraven zoveel mogelijk in chronologische volgorde.

11.1 IMPORTEERFUNCTIES

In de eerste sprint is begonnen met het importeren van de documenten en de KBT. Dit onderdeel is als eerste aangepakt omdat het werken met de aangeleverde documenten een zeer belangrijk onderdeel van de applicatie is. Want zonder document zou het niet mogelijk zijn om documenten in de applicatie te tonen en hier suggesties bij doen.

De documenten en de KBT die als voorbeeld zijn genomen in deze sprint zijn degenen die de vorige student tijdens het onderzoek gebruikt heeft. Later in het project heeft Kluwer een nieuwe set aangeleverd maar omdat deze nog niet beschikbaar was kon de beschikbare set gebruikt worden als voorbeeld voor het ontwikkelen van de importeerfuncties. De later aangeleverde set is in sprint 4 geïmporteerd.

De bestanden van Kluwer die als voorbeeld zijn genomen zijn XML bestanden. Hierbij bestaat de KBT uit één XML bestand en zijn de termen onderverdeeld in XML elementen. In tekstvak 6 staat een voorbeeld van een KBT term uit het XML document. Elk document staat in zijn eigen XML bestand met daarin verschillende elementen die geïmporteerd moeten worden zoals de content, naam, vakgebieden en trefwoorden.

```
<concept>
    <descriptor>12 jaarsgrens</descriptor>
    <niveau>TH Thesaurusterm</niveau>
    <sc>824-444 Bijzonder strafrecht / Jeugdstrafrecht</sc>
    <id>101#11414</id>
    <btg>vervolgingsbeletsel</btg>
    <flg>A</flg>
</concept>
```

Tekstvak 6 Voorbeeld KBT term

Bij deze eerste versie van de importeerfuncties was er al de factory en de IImporter maar de XmlImporter was in deze versie nog niet gemaakt. Hierdoor erfde de KluwerKBTImporter direct de IImporter over wat hem zeer veel verantwoordelijkheid gaf. Ook voor was er geen aparte thesaurus importfunctie wat dus ook werd gedaan binnen de KluwerKBTImporter.

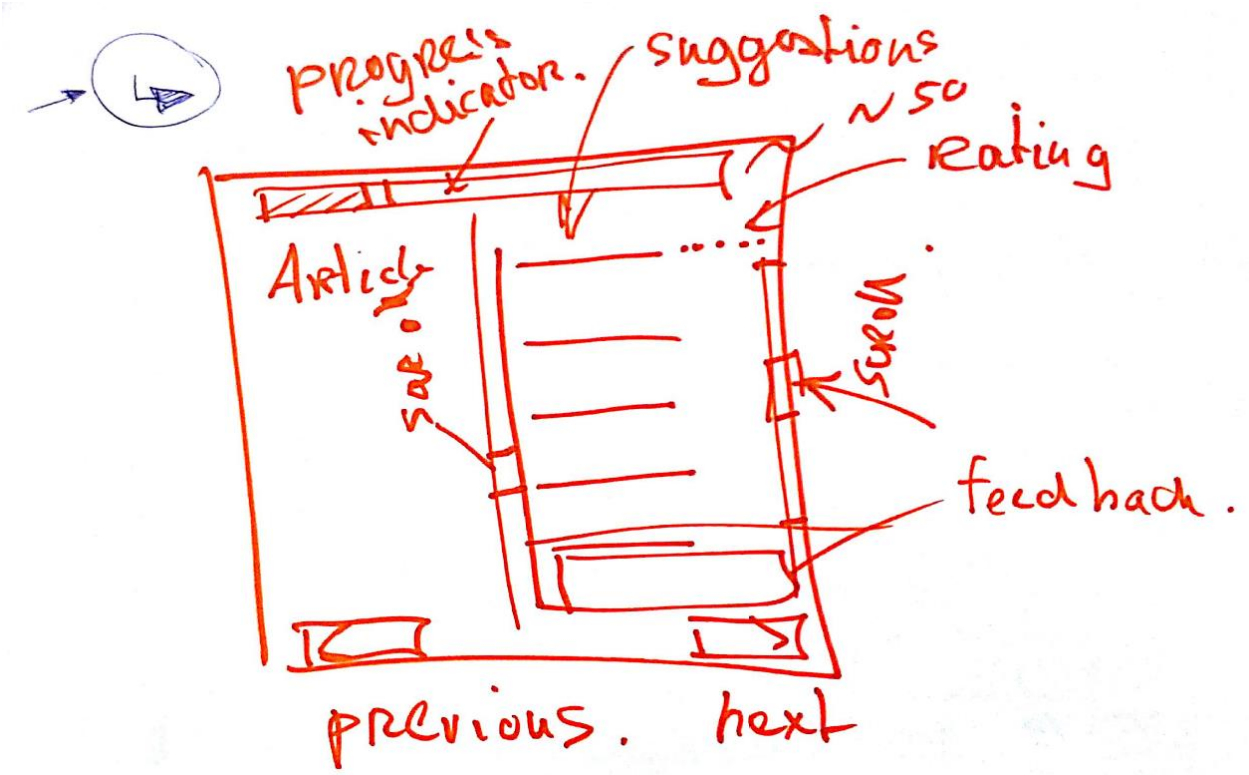
Ook zijn testbestanden van TextRank gebruikt als test documenten om te importeren. Hiervoor was de TextRankImporter gemaakt die overerft van de IImporter. Dit was een zeer eenvoudige klasse omdat het enkel ging om het importeren van eenvoudig tekst documenten.

Dit werkte wel voor de testbestanden die zijn gebruikt, maar deze structuur leverde veel dubbele code op en kon beter. Dit is op een later tijdstip gedaan omdat het voorlopig werkte. De verbetering is uitgevoerd in sprint 3 en staat beschreven in paragraaf 11.8.

11.2 OPZET TOOL

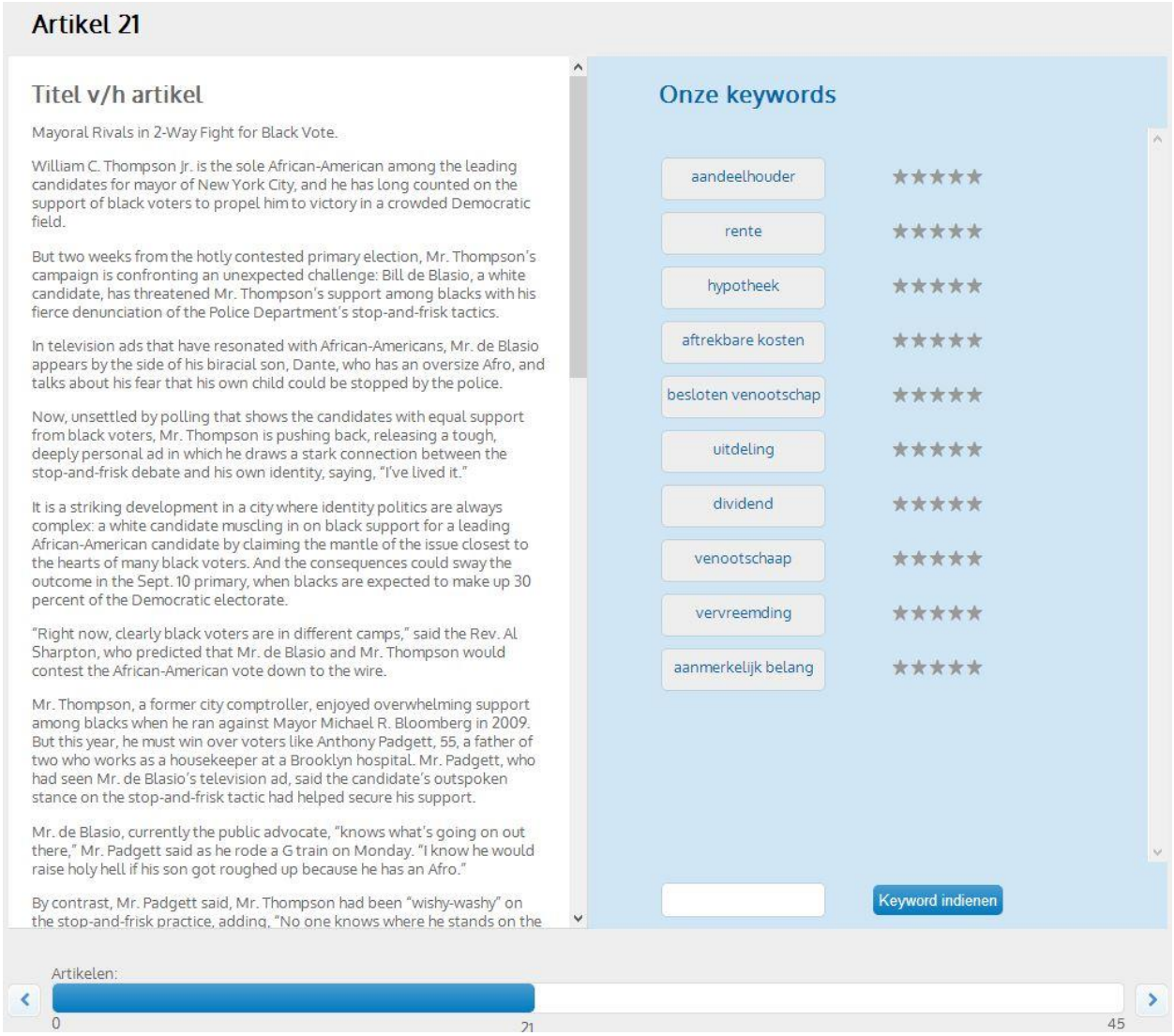
In de tweede sprint is voor de applicatie eerst de GUI (Graphical User Interface) ontwikkeld. Dit is gedaan door andere stagairs van het UX (User eXperience) team van Liones voor deze sprint, waardoor ik er al aan het begin van sprint 2 mee bezig kon.

Het opstellen van de GUI hebben zij gedaan aan de hand van een opzet van de bedrijfsbegeleider van het UX team, deze opzet is te zien in figuur 21. Dit hebben zij aangeleverd in een folder met daarin de benodigde HTML, CSS en Javascript bestanden. De aangeleverde GUI is te zien in Figuur 22. De HTML is omgezet naar Lynkx code, de CSS is direct gebruikt en de Javascript is aangepast zodat deze naar wens werkt. De GUI is later nog gewijzigd nadat de bedrijfsbegeleider van het UX team ernaar heeft gekeken, deze staat in paragraaf 11.9.



Figuur 21 GUI design

Achter de GUI is vervolgens de applicatie ontwikkeld. Hiervoor is eerst het geraamte opgezet zoals het abstract factory pattern. Ook is het stuk ontwikkeld voor het ophalen en tonen van het document. Als eerste was het alleen nog maar mogelijk om één document te bekijken, de gebruiker kon dus niet terug of verder naar een ander document.



Figuur 22 KNN eerste GUI

11.3 Implementatie KNN

In sprint 2 is het algoritme KNN geïmplementeerd voor het suggereren van trefwoorden. Met de implementatie van een algoritme is al goed te zien hoe de applicatie zal werken en vormt zo de basis. Dit was een relatief eenvoudig proces omdat goed werd begrepen hoe het KNN algoritme functioneert. En in het voorgaande onderzoek was al een implementatie van KNN geschreven die gebruik maakt van Elasticsearch. Na de bestaande implementatie aan te passen zodat het IAlgorithm hierbij geïmplementeerd was, kon het al gebruikt worden.

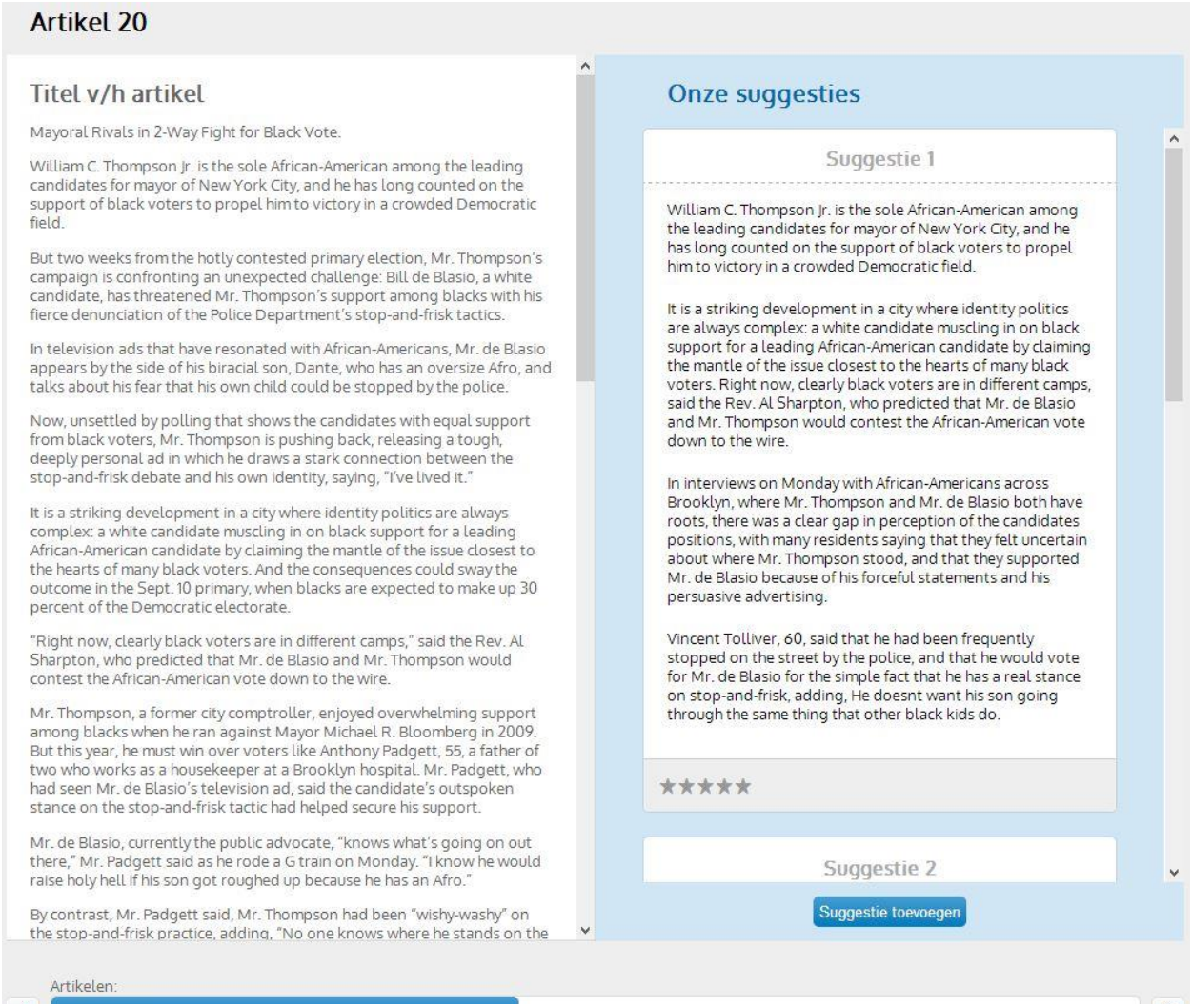
Hierna is het algoritme nog verder opgeschoond en herschreven omdat er nog delen aanwezig waren die gebruikt werden gedurende het onderzoek. Maar deze zijn in de applicatie niet meer nodig.

Tegelijk met het implementeren van KNN is verder gewerkt aan de basis van de applicatie. Hierbij is ervoor gezorgd dat de gevonden trefwoorden van KNN getoond worden in de applicatie.

11.4 Implementatie TextRank

Na de implementatie van KNN is TextRank ook nog in de tweede sprint geïmplementeerd. Wanneer naast KNN ook TextRank geïmplementeerd is, is al te zien dat de applicatie met verschillende algoritmes kan omgaan wat een belangrijke eis is voor de applicatie. Oorspronkelijk is TextRank ontwikkeld in Java, de versie die bij dit project is gebruikt is een versie van TextRank die is omgeschreven naar C#. Naar mijn inzicht was dit echter niet gedaan met een API in gedachten en meer als een POC. Zo was het bijvoorbeeld standaard een C# console applicatie. Voor het gebruiken zijn wat aanpassingen gedaan zodat de belangrijkste methodes aangeroepen konden worden. Vervolgens is het gebouwd als een library in het formaat .dll. Na deze te gebruiken bij het project naast een groot aantal andere libraries die TextRank nodig heeft, was TextRank geïmplementeerd.

De C# versie van TextRank is later in het project ook nog gebruikt door een andere collega. Deze kwam ook nog achter een aantal bugs in TextRank. De bug fixes zijn nog doorgevoerd maar als dit later echt gebruikt wordt in een productie-omgeving zal het waarschijnlijk herschreven moeten worden om het echt goed te kunnen gebruiken. Want het blijkt momenteel niet betrouwbaar te zijn gezien de gevonden



Figuur 23 TextRank eerste GUI

bugs. Ook worden zeer veel libraries gebruikt die waarschijnlijk niet allemaal nodig zijn. Tevens is de huidige implementatie redelijk langzaam, zo duurt het wel 5 seconden voor de samenvattingen zijn opgesteld van een lang document.

Voor de GUI zijn aparte delen gebouwd voor TextRank omdat de samenvattingen niet goed weergegeven worden wanneer dit op exact dezelfde manier wordt gedaan als de trefwoorden van KNN. In Figuur 23 is de GUI te zien die gebruikt is voor TextRank. Ook deze is later nog aangepast en te zien in paragraaf 11.9.

11.5 Resultaten

Na het implementeren van de algoritmes is gewerkt om de resultaten op te slaan. Hierin staan de scores die gebruiker heeft toegekend aan de trefwoorden of samenvattingen door middel van de sterren. Deze zijn in eerste instantie opgeslagen in een aparte tabel in de database. Later is hiervan afgezien en worden de resultaten opgeslagen in JSon. Dit onderdeel is in sprint 2 ontwikkeld.

Zoals te zien is in de Figuur 22 en Figuur 23 geeft de gebruiker scores door middel van sterren. Door middel van javascript wordt het aantal sterren ingevuld in een verborgen invulveld. Hier was ervoor gekozen om een extra knop te plaatsen voor het opslaan van de resultaten omdat het nog niet mogelijk was om naar een volgend of vorig document te gaan. Bij het opslaan wordt de informatie verstuurd met een POST. Vervolgens worden alle gegevens van de SurveyResult verwerkt tot JSon waarna de JSon wordt opgeslagen. Voor het lezen en schrijven naar Json is gebruik gemaakt van JSon.NET.

Na het opslaan is gewerkt aan het ophalen van de gegevens zodat de juiste gegevens staan ingevuld wanneer de gebruiker teruggaat naar een al ingevuld document. Hiervoor wordt bij het ophalen van de trefwoorden of samenvattingen ook gekeken of er al resultaten zijn. Als dit het geval is, dan worden de resultaten uit de JSon gelezen en gekoppeld aan de juist gevonden trefwoorden of samenvattingen.

Er is voor gekozen om altijd eerst de trefwoorden of samenvattingen te genereren door het algoritme en daarna pas te kijken naar de resultaten zodat mogelijke wijzigingen die impact hebben op de resultaten van algoritmes zullen werken.

11.6 Gebruiker trefwoord-suggesties + overig commentaar

Om ervoor te zorgen dat een gebruiker kan aangeven welke trefwoorden er in de gesuggereerde lijst misten, is er de optie gemaakt om eigen suggesties toe te voegen. Hiermee kunnen we zien welke trefwoorden er in de lijst werden verwacht voor het document maar hierin niet aanwezig waren. De mogelijkheid om suggesties toe te voegen is het laatste onderdeel dat in de tweede sprint is ontwikkeld.

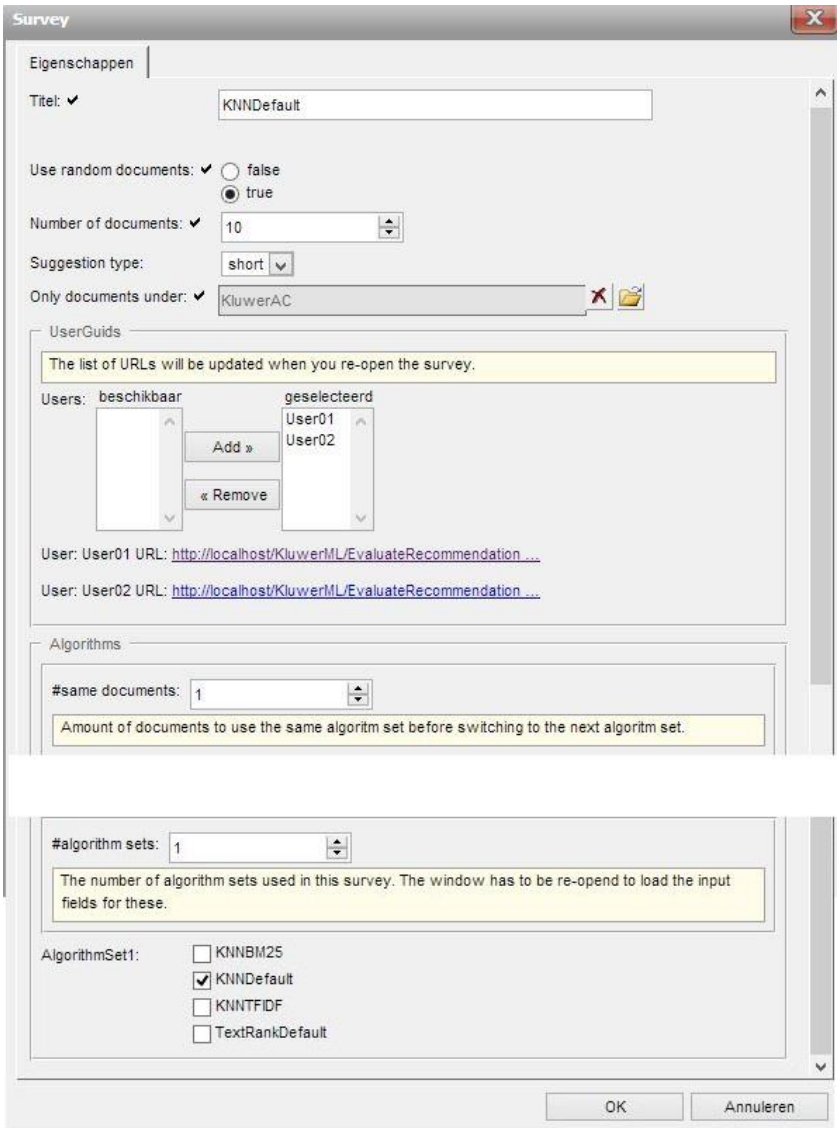
Voor het toevoegen van trefwoorden is een normaal invoerveld toegevoegd. Het is ook mogelijk om bij samenvattingen eigen suggesties toe te voegen, in dit geval is er textarea in plaats van een invoerveld, zodat er meer ruimte is voor tekst.

De suggesties worden als aparte nodes opgeslagen, waarna daarnaartoe wordt verwezen door middel van een foreign key. Wanneer bij een enquête meerdere malen dezelfde suggestie wordt gebruikt dan zal deze niet meerdere malen worden aangemaakt maar zal er naar de al bestaande suggestie een link worden gelegd.

Ook is de optie ingebouwd voor gebruikers om overig commentaar in te voeren, dit is ontwikkeld in sprint 4. Dit was een vraag van de stakeholder (contactpersoon van Kluwer) om de gebruikers de mogelijkheid te geven om verder commentaar te geven. Dit bleek ook echt gebruikt te worden, bijvoorbeeld: Een redacteur van Kluwer komt een document tegen uit een verkeerde set, hij/zij geeft aan in het extra commentaar veld dat het document uit de verkeerde set komt en geeft alle trefwoorden vervolgens de laagste beoordeling.

11.7 Opstellen enquête

In de derde sprint is eerst gewerkt aan het opstellen van een enquête. Hierbij kan worden opgegeven welke documenten gebruikt moeten worden, voor welke gebruikers de enquête is en welke algoritmes gebruikt moeten worden. Voor het opstellen van de enquête wordt gebruik gemaakt van het CMS deel van Lynkx genaamd de backoffice. In de backoffice kunnen nodes worden aangemaakt van de opgestelde types zoals beschreven in hoofdstuk 10.1. In Figuur 24 is het formulier te zien voor een SurveyNode, dit is het type wat gebruikt wordt voor het aanmaken van een enquête.



Figuur 24 Backoffice enquête formulier

Het aanmaken van het type is SurveyNode is gedaan door een ander al bestaand type te kopieëren en deze vervolgens aan te passen. Naast het type voor de enquête is er ook een type gemaakt voor een algoritme, hierbij zijn de velden gedefnieerd voor de naam van het algoritme en het aantal suggesties dat gegenereerd moet worden. Voor de algorimtes KNN en TextRank zijn vervolgens aparte types aangemaakt die van het algorimte type overerven. Hierdoor hebben zij al de velden van het algorimte en zijn extra velden aangemaakt. Zoals de waarde K voor KNN en de lengte van de samenvattingen voor TextRank.

11.8 Refactoring importeerfuncties.

Omdat de eerste versie van de importeerfuncties niet voldeden zijn deze in de derde sprint herschreven. Bij het herschrijven is ervoor gezorgd dat extra importeerfuncties gemakkelijker toegevoegd kunnen worden. De structuur zoals weergegeven in het klassendiagram voor importeerfuncties in paragraaf 10.2 is hier ontwikkeld. Bij het herschrijven kan een XML nu geïmporteerd worden met behulp van een mapping.

Bij een mapping kan aangegeven worden welke elementen van de XML moeten worden opgeslagen in welke velden van een node. Hierbij kunnen ook Flags worden toegevoegd als er iets anders moet gebeuren dan standaard een XML element naar een veld. Zo kan de bestandsnaam geselecteerd worden, kunnen XML elementen worden overgeslagen wanneer deze bepaalde attributen wel of niet hebben. Ook is het mogelijk om juist attributen van een XML element te selecteren in plaats van de inhoud van het element. Verder is het mogelijk om meerdere elementen aan elkaar te plakken en XML elementen om te zetten naar HTML elementen om zo de opmaak te regelen.

Na deze refactoring is een code review gedaan met een collega waarbij nieuwe ideeën waren, maar deze zijn uiteindelijk niet doorgevoerd wegens een gebrek aan tijd. Deze ideeën staan beschreven aan het einde van paragraaf 10.2.

11.9 Bug fixes en kleine features

In de vierde sprint zijn voornamelijk nog wat kleine features ingebouwd en is veel gewerkt aan bug fixes.

Er bleken nog een aantal bugs te zitten bij het importeren, waardoor niet alle onderdelen van thesaurus aan elkaar gelinkt waren. Hierdoor werden er geen documenten gevonden wanneer een term hoog in thesaurus geselecteerd werd. Na deze bug fix bleken er veel documenten gevonden te worden, waardoor een performance issue opkwam. Hierdoor duurde het lang voordat een selectie aan documenten gemaakt werd (meer dan 15 seconden). Dit was niet acceptabel omdat deze selectie wordt gedaan wanneer een gebruiker een enquête voor de eerste keer opent. Om dit op te lossen is een nieuwe query opgesteld om direct uit de database een willekeurige set op te vragen in plaats van alle documenten op te halen en daaruit een selectie willekeurige documenten maken.

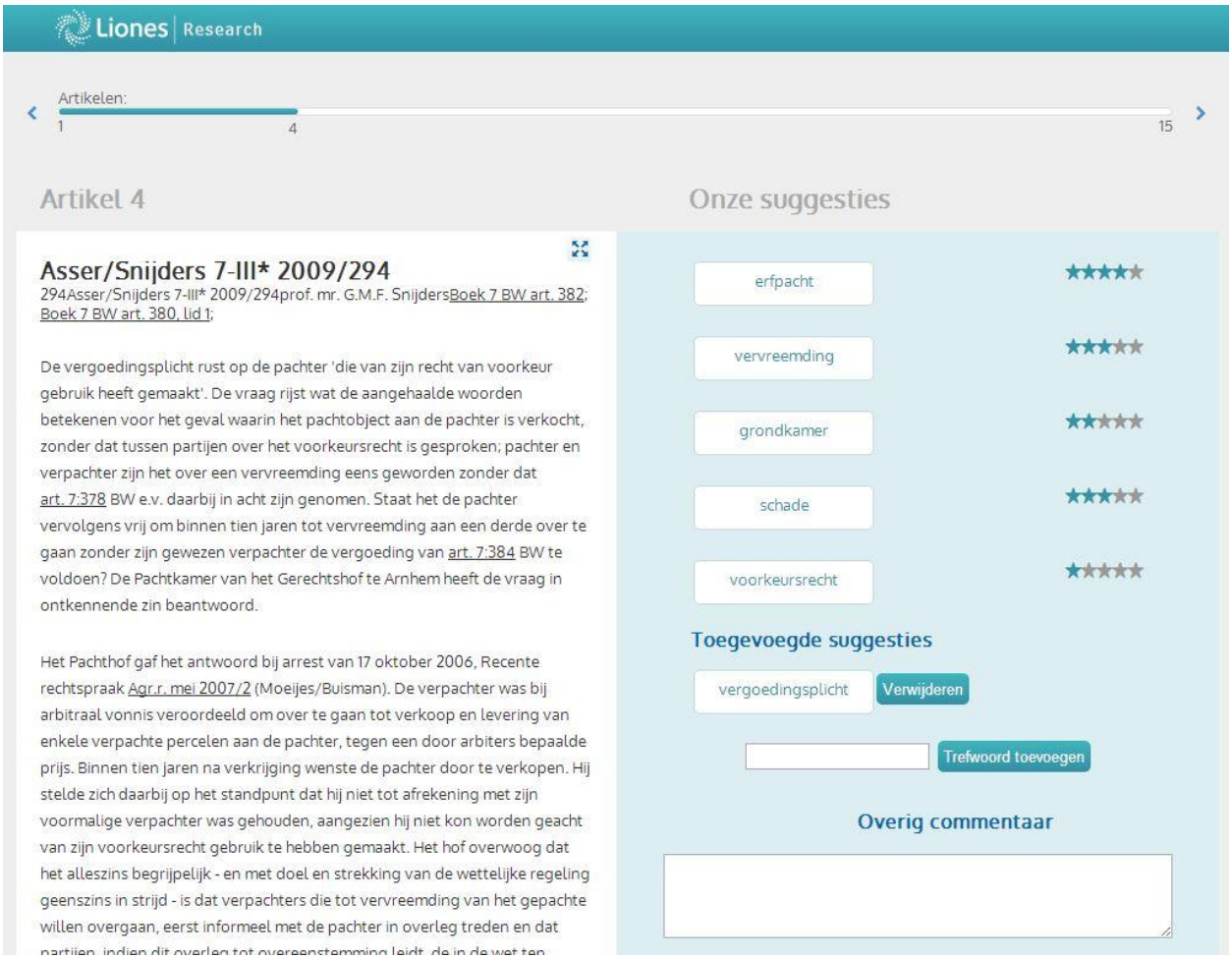
Ook werd bij de documentselectie op basis van een thesaurus gekeken naar de trefwoorden die aan een document waren toegewezen. Dit bleek niet te moeten omdat Kluwer in de XML van de documenten aparte velden had, waarop de documenten geselecteerd moesten worden. Deze bugs zelf waren snel op te lossen maar omdat de bugs in de importeerfuncties zaten, moesten alle documenten weer opnieuw geïmporteerd worden. Omdat met een grote documentenset gewerkt wordt, was dit ook een langdurig proces.

Verder waren er ook nog meerdere kleine bugs die snel op te lossen waren. Zo bleek de weergave niet goed te werken in Internet Explorer 8 en was niet bekend waarom KNN bij sommige documenten minder dan 10 trefwoorden vond.
De bug in Internet Explorer 8 was eenvoudig op te lossen door terug te stappen naar een oudere versie van JQuery en gebruik te maken van een Javascript library wat Internet Explorer 8 laat omgaan met HTML 5.
Dat KNN niet bij alle documenten de gevraagde 10 trefwoorden kon vinden kon niet worden opgelost

maar wel is gevonden waarom dit het geval was. Want KNN zoekt de 25 meest lijkende documenten, van deze documenten worden alle trefwoorden opgehaald en hieruit worden de 10 meest voorkomende getoond. Echter bleek het wel eens te gebeuren dat de 25 gevonden documenten niet in totaal 10 verschillende trefwoorden te hebben.

Als laatst zijn er nog twee features toegevoegd. De eerste controleert of de gebruiker alle suggesties een beoordeling heeft gegeven voordat hij/zij verder kan naar het volgende of vorige document. En de tweede feature zorgt ervoor dat de vorige knoppen niet worden getoond wanneer de gebruiker bij het eerste document is. En bij het laatste document staat er geen knop volgende maar voltooien. Met deze knop krijgt de gebruiker een pagina waar hij/zij bedankt wordt voor het invullen.

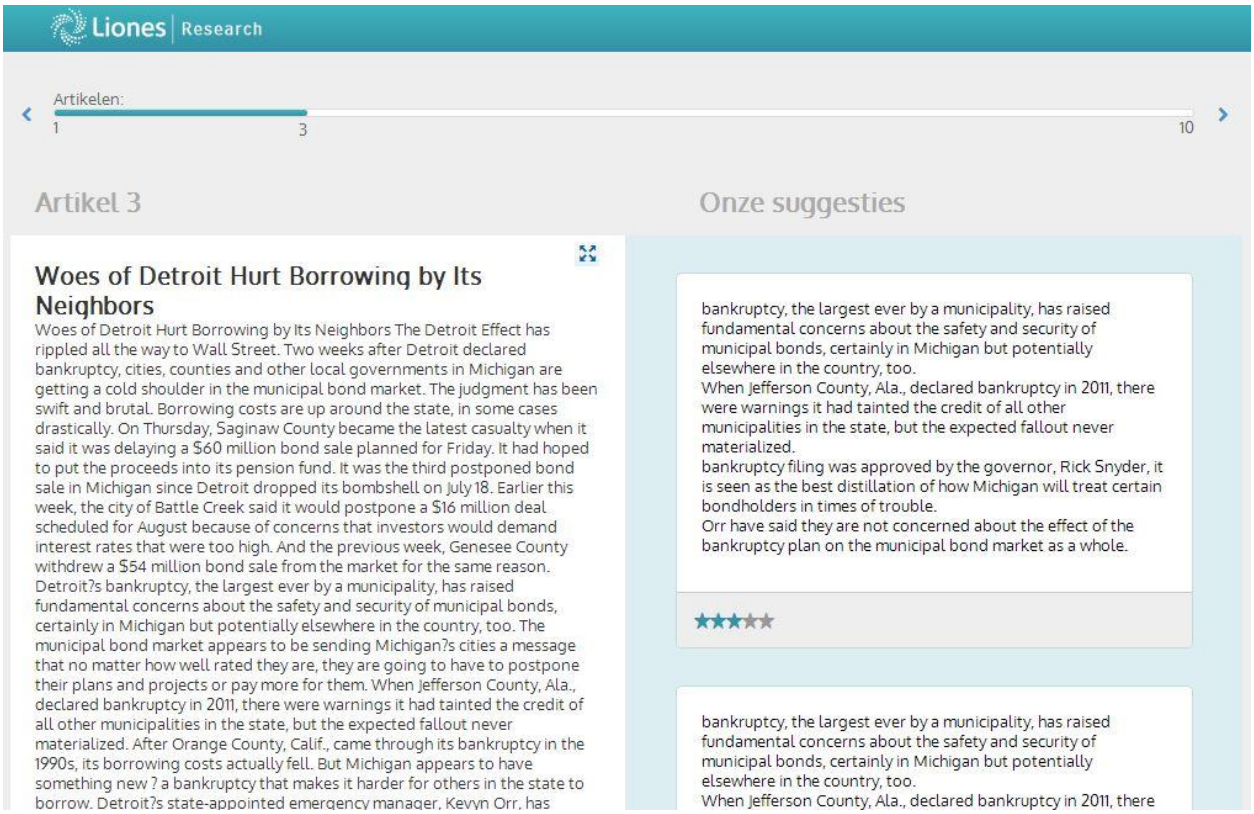
De rest van vierde sprint is gebruikt voor het uitvoeren van testen.



Figuur 25 Uiteindelijk KNN

In de vijfde sprint is de applicatie gebruikt door de stakeholder van Kluwer. Hierbij is gemerkt de selectie van documenten niet correct verliep. Want voor Kluwer wordt de document voor een enquête geselecteerd aan de hand van de KBT. Hierbij wordt een willekeurig aantal documenten genomen die allemaal onder dezelfde term in de KBT zitten. Om te bepalen of een document onder een term valt werd gekeken of het document die term of een onderliggende term als trefwoord heeft. Maar hiervoor moest gekeken worden naar het “vakgebied” van het document. Dit is een veld uit de XML maar deze was niet geïmporteerd. En dus is dit veld opgenomen in de mapping voor het importeren waarna de import nogmaals is uitgevoerd.

In figuur 25 is de uiteindelijke applicatie zien met gebruik van KNN en in figuur 26 is de applicatie te zien met gebruik van TextRank.



Figuur 26 Uiteindelijk TextRank

12. TESTEN

Voor dit project zijn twee testen uitgevoerd en gerapporteerd. Voor het importeren van een thesaurus is een moduletest opgesteld en voor het testen van de applicatie is een acceptatietest opgesteld. Verder is de applicatie meerdere malen doorlopen en is tijdens het ontwikkelen getest door middel van error guessing en exploratory testing.

12.1 MODULE TEST IMPORTEERFUNCTIES

Voor het importeren is een module test opgesteld. Hierbij worden alle belangrijke paden voor het importeren van een thesaurus doorlopen. Het importeren is verdeeld over vier onderdelen “ImportTag”, “KbtXmlImporter”, “ThesaurusImporter” & “LoadTermIds”. De voornaamste reden dat het is onderverdeeld is om ervoor te zorgen dat het overzichtelijk blijft. Hieronder staat van elk onderdeel een korte uitleg waarvoor het verantwoordelijk is.

ImportTag
In dit onderdeel wordt de hoofd node aangemaakt. Zoals te lezen in paragraaf 10.1 moet elke node worden aangemaakt onder een andere node als parent. De hoofd node is de node waaronder de nieuwe nodes aangemaakt zullen worden. Hiernaast zullen de bestanden worden uitgelezen en doorlopen. Bij het ophalen van de bestanden is gebruik gemaakt van batch verwerking om ervoor te zorgen dat het geheugen niet vol loopt.

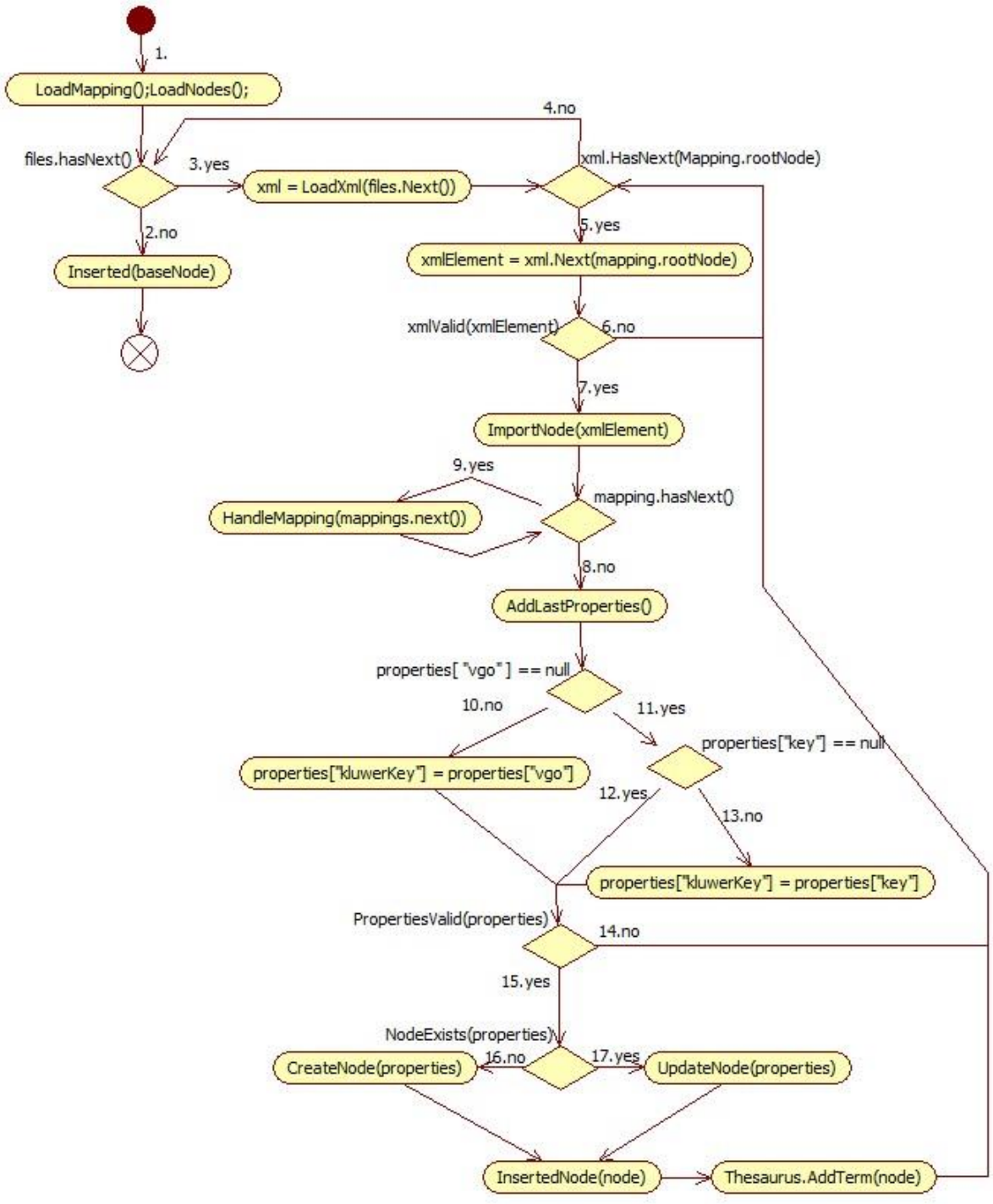
KbtXmlImporter
De KbtXmlImporter vertaald het XML bestand van de KBT naar Nodes. Hiervoor worden de termen in de XML doorlopen, er wordt gekeken welke termen geïmporteerd mogen worden en nodige conversies worden uitgevoerd.

ThesaurusImporter
Bij het aanmaken van de Nodes zijn de velden die verwijzen naar andere termen wel al meegenomen. Deze verwijzingen kunne verwijzen naar verschillende velden van andere Nodes zoals de naam. In dit onderdeel zullen deze verwijzingen worden opgeslagen in de velden “bt”, “nt” & “rt” en deze zullen verwijzen naar de id’s van andere Nodes. Om het overzichtelijk te houden is het onderdeel dat de veld (bijv. naam) vertaald naar id’s afgesplitst naar een apart procesdiagram en dus ook een aparte test.

LoadTermIds
Zoals beschreven in de ThesaurusImporter zal dit onderdeel verwijzingen vertalen naar id’s. Hierbij kan de originele verwijzing gaan naar de naam van een Node of een ander veld dat is aangemaakt en staat opgeslagen in de extraProperties.

Elk onderdeel is getest door middel van een algoritme test. Hierbij is eerst een procesdiagram opgesteld waarna het logische testontwerp en fysiek testontwerp en als laatste de Unit Test. Dit is gedaan op testmaat 1 omdat er al een datum met de stakeholder was afgesproken waarop hij het zou gaan gebruiken. Om na het testen nog de nodige bug-fixes te doen is testmaat 1 gekozen om zo tijd te besparen.

In deze paragraaf zal de KbtXmlImporter als voorbeeld genomen worden. Want dit is een interessanter onderdeel dan de ImportTag maar staat wel op zichzelf wat niet het geval is bij de ThesaurusImporter en LoadTermsIds.



Figuur 27 Procesdiagram KbtXmlImport

Procesdiagram
In figuur 27 is het procesdiagram te zien van de KbtXmlImporter. Het diagram is opgesteld aan de hand van de al geschreven code.

Het procesdiagram begint wanneer de methode ImportNodes wordt aangeroepen. Hierbij wordt de hoofd node meegegeven en een dictionary files. Alle bestanden die geïmporteerd worden zullen worden geplaatst onder de hoofd node. De dictionary files heeft als sleutels de namen van de bestanden en als waarden de inhoud van de bestanden.

De methode begint met het inladen van de correcte mapping. De mapping zal gebruikt worden om te bepalen welke XML elementen worden gebruikt voor welke velden van een node. Om de termen te vinden wordt door elke bestand geloopt en door elke rootNode. De rootNode is een waarde die is ingesteld in de mapping. Dit houdt in welk XML element geldt als basis voor een term zodat meerdere termen in één XML bestand doorlopen kunnen worden. Er wordt gekeken of de XML valide is, voor de KBT is het bijvoorbeeld een eis dat het XML element <flg> niet gevuld is met een B want dan is de term niet toewijsbaar. Wanneer de XML valide is zullen de mappings worden uitgevoerd, hierbij worden de XML elementen omgezet naar velden in een dictionary. Dit onderdeel is niet opgenomen in de test want dit heeft niet direct te maken met het importeren van een thesaurus maar het omzetten van XML elementen naar velden in een dictionary. Hierna wordt een methode aangeroepen waarbij laatste wijzigingen gedaan kunnen worden voordat het wordt opgeslagen als een node. Zo kan een veld worden ingevuld of het geïmporteerd mag worden als node. Bij de KBT wordt het veld kluwerKey ingevuld een substring van de key of de waarde “vgo”. Dit kon niet gedaan worden door middel van de mapping omdat daarbij geen if statements gedefinieerd kunnen worden. De kluwerKey is de waarde waar documenten naartoe verwijzen. Wanneer een term hoger in de thesaurus staat wordt hiervoor het XML element vgo voor gebruikt en anders een deel van het XML element id. In tekstvak 7 staat een voorbeeld van een term laag in de thesaurus en een term hoog in de thesaurus.

```
<concept>
 <descriptor>12 jaarsgrens</descriptor>
 <niveau>TH Thesaurusterm</niveau>
 <sc>000-999 Onbekend (V)</sc>
 <sc>824-444 Bijzonder strafrecht / Jeugdstrafrecht</sc>
 <id>101#11414</id>
 <btg>vervolgingsbeletsel</btg>
 <flg>A</flg>
</concept>

<concept>
 <descriptor>Bijzonder strafrecht / Jeugdstrafrecht</descriptor>
 <niveau>HO Hoofdonderwerp</niveau>
 <rub>STR Strafrecht</rub>
 <id>100#333</id>
 <vgo>824-444</vgo>
 <bt>Bijzonder strafrecht (V)</bt>
 <flg>A</flg>
</concept>
```

Tesktvak 7 Voorbeeld KBT termen

Hierna zal gekeken worden of de term valide is om te importeren. Wanneer de term valide is, zal gekeken worden of deze al bestaat als node en zal de term of toegevoegd worden of de bestaande node

zal gewijzigd worden. Als laatste wordt de term toegevoegd aan een lijst zodat de termen kunnen worden verwerkt door de thesaurusImporter zonder dat deze ze weer moet ophalen uit de database.

Logisch testontwerp

Voor het opstellen van het logische testontwerp zijn alle paden van het procesdiagram doorlopen. Omdat is gekozen voor testmaat 1 hoeft geen rekening gehouden te worden met de verschillende paden combinaties. Wel zijn de onmogelijke paden ontweken bij het opstellen van het logische testplan. Bijvoorbeeld: Omdat voor alle termen dezelfde mapping gebruikt wordt moeten alle termen ook hetzelfde aantal mappings (pad 9) hebben.

Het procesdiagram kan in één keer doorlopen worden waarbij ook alle paden gebruikt worden. Vervolgens is het gevonden pad opgedeeld in het aantal benodigde termen. Omdat pad 5 kenmerkt of er een volgende term aanwezig is binnen het XML bestand is pad opgedeeld bij elke 5. Vervolgens is elk stuk pad doorlopen en hierbij is opgeschreven waaraan het moet voldoen om de bepaalde paden te volgen. In tabel 4 is het uiteindelijke logische testontwerp te zien.

Logisch testontwerp 01	
Procesdiagram pad	1-3-5-6-5-7-9-8-10-14-5-7-9-8-11-12-15-16-5-7-9-8-11-13-15-17-4-2
Aantal termen	4
Term1 (5-6)	XML is NIET valid.
Term2 (5-7-9-8-10-14)	XML is valid. Heeft 1 mapping. Properties[“vgo”] != null. Properties is NIET valid.
Term3 (5-7-9-8-11-12-15-16)	XML is valid. Heeft 1 mapping. Properties[“vgo”] = null. Properties[“key”] = null. Properties is valid Node bestaat nog niet.
Term4 (5-7-9-8-11-13-15-17)	XML is valid Heeft 1 mapping. Properties[“vgo”] = null. Properteis[“key”] != null. Properties is valid. Node bestaat al.

Tabel 4 KbtXmlImporter Logisch testontwerp

Fysiek testontwerp

Aan de hand van het logische testontwerp is het fysieke testontwerp opgesteld zoals te zien in tabel 5. Hiervoor is de XML opgesteld zodat de verwachte termen aanwezig zijn met de correcte waarden. Zo is bij de eerste term <flg>B</flg> toegevoegd waardoor hij als niet valide wordt beschouwd bij xmlValid(xmlElement).

Verder is opgenomen waaraan de database moet voldoen. Zo moeten twee termen al aanwezig zijn zodat deze gewijzigd zullen worden in plaats van aangemaakt. En als laatste is opgeschreven wat het verwachte resultaat is. Met deze informatie kan vervolgens een Unit Test opgesteld worden.

Fysiek testontwerp 01	

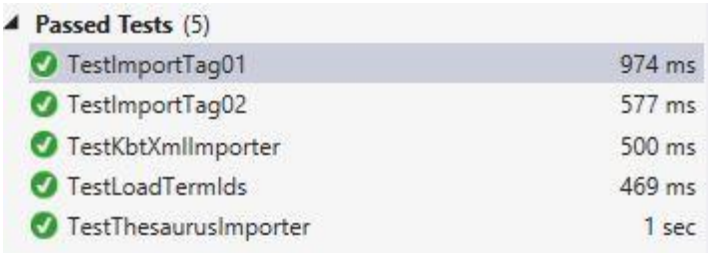
XML	`<?xml version="1.0" encoding="utf-8" ?>`
	`<kbt>`
	`<thesaurus>`
	`<concept>`
	`<descriptor>TestTerm01</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#01</id>`
	`<flg>B</flg>`
	`</concept>`
	`<concept>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#2</id>`
	`<vgo>vgo01</vgo>`
	`<flg>A</flg>`
	`</concept>`
	`<concept>`
	`<descriptor>TestTerm03</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#</id>`
	`<flg>A</flg>`
	`</concept>`
	`<concept>`
	`<descriptor>TestTerm04</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#04</id>`
	`<flg>A</flg>`
	`</concept>`
	`</thesaurus>`
	`</kbt>`

Database	Node met naam “TestTerm03” en key “101#” mag niet aanwezig zijn in de database. Node met naam “TestTerm04” en key “101#04” moet wel aanwezig zijn inde database.
Verwacht resultaat	Onder de node “TestKbtXmlImporter” zijn de volgende nodes aangemaakt/gewijzigd. **Aangemaakt:** Naam: “TestTerm03”, Key: “101#”, Description: “” **Gewijzigd:** Naam: “TestTerm04”, Key: “101#4”, Description: “04” De eerste XML term is niet valide want bij flg staat B. De tweede XML is wel valide maar zijn properties zullen invalide zijn omdat deze geen naam heeft.

Tabel 5 KbtXmlImporter fysiek testontwerp

Unit Test
Voor het opstellen van de Unit Test is gebruik gemaakt van het NUnit framework. Het opstellen van de Unit Test bleek echter wel een stuk lastiger te zijn dan van tevoren was verwacht. Want er waren problemen met het simuleren van de database door dat de connectie hiermee wordt geregeld door het Lynkx framework. En het Lynkx framework werkt door veel bij te houden in de context maar deze wordt niet correct gevuld wanneer het niet wordt aangeroepen vanuit een browser. Uiteindelijk is de Setup() methode uit de Unit Test van een ander project als voorbeeld gebruikt om hiermee een simulatie van Lynkx te starten. Ook met dit voorbeeld kost het veel moeite en tijd om het werkend te krijgen. Ook is voor het testen een aparte database gebruikt om er zeker van te zijn welke informatie wel en niet aanwezig is in de database.

Voor elke testmethode wordt de Setup() en de Teardown() uitgevoerd om de simulatie van Lynkx te starten en op te ruimen. Ook worden de benodigde tabellen hiermee in de testdatabase aangemaakt en deze worden ook weer verwijderd na het draaien van de test. Wanneer bepaalde data in de database moet zijn voor een bepaalde test bijvoorbeeld: Er moet een node aanwezig zijn met naam “TestTerm04” en key “101#04”. Dan wordt deze data toegevoegd aan het begin van de testmethode na de Setup(). Het toevoegen wordt gedaan door middel van een nodesQuery().



Figuur 28 Unit Test resultaten

Uiteindelijk zijn de testen succesvol opgesteld en uitgevoerd zie figuur 28.

12.2 Acceptatietest
Voor de acceptatietest is eerst een testplan opgesteld. Hierbij is bepaald welke personen de test zullen uitvoeren. Hiervoor zijn in totaal vier personen geselecteerd: de ontwikkelaar, een collega, een stakeholder en de opdrachtgever. Voor deze personen is om de volgende redenen gekozen.

De ontwikkelaar
Dit ben ikzelf en door het doorlopen van de acceptatietest zullen de meeste fouten al worden opgemerkt zodat deze nog verholpen kunnen worden voordat anderen de test doorlopen.

Een collega
De collega is een mede stagiair bij de afdeling R&D. Hij heeft de ontwikkelde applicatie nog niet eerder gezien en is hierom ook een geschikt persoon om de gebruiksvriendelijkheid te testen. Hiervoor was het belangrijk omdat de gebruiksvriendelijkheid wordt getest door te kijken hoe snel iemand met de applicatie om kan gaan zonder hier een uitleg van gekregen te hebben. De test voor gebruiksvriendelijkheid is te vinden in tabel 7.

Een stakeholder
De stakeholder is de contactpersoon van Kluwer. De stakeholder kent de gebruikers (de redacteuren van Kluwer) beter en kan dus ook beter inschatten wat zij verwachten van de applicatie.

De opdrachtgever
De opdrachtgever heeft als laatste de acceptatietest doorlopen. Op deze manier is getoond welke eisen zijn verwerkt in de applicatie en wat er (nog) niet in zit. Dit zal verder worden besproken bij de resultaten.

Test scenario's
Voor de user stories uit de requirements zijn test scenario's opgesteld. En ook voor de non-functionele eisen zijn test scenario's opgesteld. Bij de test scenario's voor de user stories en non-functionele eisen is de code van user story of eis aangegeven met daarbij de eis. Verder zijn er ook nog test scenario's opgesteld voor eisen die tijdens het project erbij kwamen.

De scenario's hebben een code gekregen om het scenario te identificeren. De scenario's voor de user stories en non-functionele eisen zijn gemarkeerd met SC[nr] en de scenario's voor de eisen die tijdens het project erbij kwamen zijn gemarkeerd met SCX[nr]. Hieronder zijn de volgende voorbeelden gegeven van opgestelde scenario's. Tabel 6 voorbeeld scenario voor een user story, tabel 7 voorbeeld scenario voor een non-functionele eis en tabel 8 voorbeeld scenario voor later verkregen eis.

Code	*SC02*
User story/eis	US02. Als gebruiker wil ik gevonden keywords voor een document kunnen beoordelen.
Scenario	1. De gebruiker heeft de enquête geopend. 2. De gebruiker klikt voor elk gevonden keyword op een score/ster. 3. De gebruiker gaat naar het volgende document en keer vervolgens terug. 4. De gegeven scores zijn opgeslagen en zullen nog steeds hetzelfde zijn.

Tabel 6 Test scenario user story

Code	*SC10*
User story/eis	N02. Gebruiksvriendelijk voor de gebruikers. Zij moeten de tool zonder uitleg binnen 1 minuut begrijpen en kunnen gebruiken.
Scenario	1. Wanneer de gebruiker de enquête tool al eens heeft gebruikt zal het resultaat minder betrouwbaar zijn. Want hij zal dan al weten hoe het werkt. 2. De gebruiker opent de enquête via de link. 3. Na 1 minuut wordt de gebruiker het volgende gevraagd te doen/beantwoorden en kan elk uitvoeren binnen 5 seconden. 4. Beoordeel 3 keywords/trefwoorden/samenvattingen (hoeft niet accuraat). 5. Voeg 2 eigen suggesties toe. 6. Ga naar een volgend document. 7. Hoeveel document zitten er in deze enquête.

Tabel 7 Test scenario non-functionele eis

Code	*SCX02*
User story/eis	Een gebruiker kan overig commentaar invullen per document.
Scenario	1. De gebruiker heeft een enquête geopend. 2. De gebruiker vult het commentaar in. (onder de keywords) 3. De gebruiker gaat naar het volgende document. 4. Het commentaar is opgeslagen. (Wanneer de gebruiker terug gaat naar het document zal het commentaar nog steeds ingevuld staan.

Tabel 8 Test scenario later verkregen eis

Resultaten
Voor het uitvoeren van de testen heeft de testpersoon de scenario's doorlopen. Wanneer de collega de test heeft doorlopen is begonnen bij SC10 (tabel 7) omdat het geen eerlijke test meer zou zijn wanneer eerst andere scenario's worden doorlopen waardoor hij de applicatie al leert kennen. Bij het doorlopen heb ik ernaast gezeten om de resultaten te noteren. Wanneer een scenario was doorlopen heb ik gevraagd of ik hem kan afvinken en eventueel commentaar bij geschreven.

De eerste keer heb ik als ontwikkelaar zelf de acceptatietest doorlopen. De enige twee punten die hieruit kwamen waren dat de resultaten van de enquête nog niet goed in te zien waren en dat TextRank niet altijd de resultaten heeft binnen 5 seconden.
Dat de resultaten niet goed in te zien waren was bekend omdat dit pas voor later in project gepland staat tijdens de analyse, ook was hiervoor nog niet bekend hoe dit gedaan zou worden. Want terwijl de applicatie gebruikt werd door de redacteuren van Kluwer zou ik bezig gaan met het voorbereiden om de resultaten te analyseren.
En dat TextRank het niet altijd haalde binnen 5 seconden was ook een bekend probleem. Echter is dit zo gelaten omdat het binnen de tijd niet haalbaar was om TextRank te herschrijven.

De tweede persoon die heeft doorlopen is de collega. Het interessantste wat hieruit is gekomen zijn de resultaten voor de gebruiksvriendelijkheid. De resultaten die hierboven beschreven staan zijn ook geconstateerd deze zullen niet nogmaals beschreven worden omdat hetzelfde hieruit kwam.
Voor het uitvoeren van de test is wel uitgelegd dat het om een enquête applicatie gaat waarbij suggesties worden beoordeeld. Bij het uitvoeren van de test waren de meeste onderdelen wel duidelijk en waren het voornamelijk kleine delen die niet geheel duidelijk waren of anders verwacht. Zo was het niet duidelijk of één ster betekend dat het geheel fout is of gedeeltelijk. En leken de gevonden trefwoorden van KNN op knoppen waarmee sterren worden toegevoegd.
Een aantal van de opmerkingen zijn nog verwerkt maar niet allemaal. Zo zijn de sterren gelaten zoals ze zijn maar kunnen nu inderdaad sterren worden toegevoegd door op het gevonden trefwoord te klikken.

Na de punten verwerkt te hebben van de test met de collega is het getest door de stakeholder. Bij de test werd net als bij de voorgaande testen geconstateerd dat de resultaten van een enquête nog niet goed in te zien zijn. Dit werd hier ook niet als een probleem gezien omdat hier nog aan gewerkt zal worden.
Dat TextRank meer dan 5 seconden nodig heeft voor lange documenten is ook gemerkt maar niet als probleem gezien. Want de redacteuren van Kluwer zullen eerst bezig gaan met het testen van KNN die wel snel is. Overigens wil Kluwer later misschien alsnog TextRank testen maar hiervoor werd de tijd die TextRank nodig heeft niet als te lang beschouwd.

Als laatste is de test doorlopen met de opdrachtgever (Software architect binnen Liones). Bij het doorlopen van de scenario's is ook soms getoond waar de informatie wordt opgeslagen wanneer hiernaar werd gevraagd. Zo is getoond waar de resultaten, suggesties & overig commentaar wordt opgeslagen. Verder is ook hier gemerkt dat TextRank niet altijd klaar is binnen 5 seconden. Bij de opdrachtgever is het bekend dat de gebruikte versie van TextRank niet van productie kwaliteit is en dus is het geaccepteerd dat deze er soms langer over doet. Overigens is ook de fout ontdekt dat de ingevulde resultaten niet worden getoond wanneer de gebruiker naar een vorig document gaat via de browser-terug-knop.
Dat de resultaten niet overzichtelijk in te zien zijn is ook gemerkt en in overleg is ervoor gekozen om hiervoor niet één generieke oplossing te ontwikkelen maar dit specifiek te doen per klant. Hiervoor is gekozen omdat het binnen de resterende tijd niet haalbaar is om nog een onderdeel te ontwikkelen waarmee de resultaten voor verschillende algoritmes getoond kunnen worden. In plaats daarvan zal per algoritme voor een klant een aparte analyse opgesteld worden. Hoe dit is gedaan voor het algoritme KNN voor de klant Kluwer is te lezen in paragraaf 13.3.

13. ANALYSE

Om te bepalen hoe goed KNN functioneert voor Kluwer zijn drie onderdelen geanalyseerd. Als eerste is de set aangeleverde XML documenten geanalyseerd om te bepalen hoeveel hiervan bruikbaar zijn en hoeveel trefwoorden de meeste documenten toegekend hebben. Hierna is geanalyseerd hoe goed KNN de al toegekende trefwoorden kan bepalen. Bij het vorige onderzoek (Middel, 2013) is dit ook gedaan maar ditmaal is het uitgevoerd met een grotere set documenten. En als laatste zijn de resultaten van de redacteurs geanalyseerd. Aan de hand van de drie onderdelen is een conclusie opgesteld.

13.1 DOCUMENT ANALYSE

De documenten zijn aangeleverd in verschillende mappen. Deze mappen zijn geïmporteerd onder verschillende nodes. Welke map onder welke node terecht is gekomen is aangegeven in tabel 9.

Het analyseren van de aangeleverde documenten is uitgevoerd per map. Per map zijn alle documenten doorlopen en hierbij is gekeken of deze geïmporteerd mag worden en zo nee waarom niet. Ook is gedurende dit proces gecontroleerd of de documenten die geïmporteerd mogen worden ook daadwerkelijk geïmporteerd zijn.

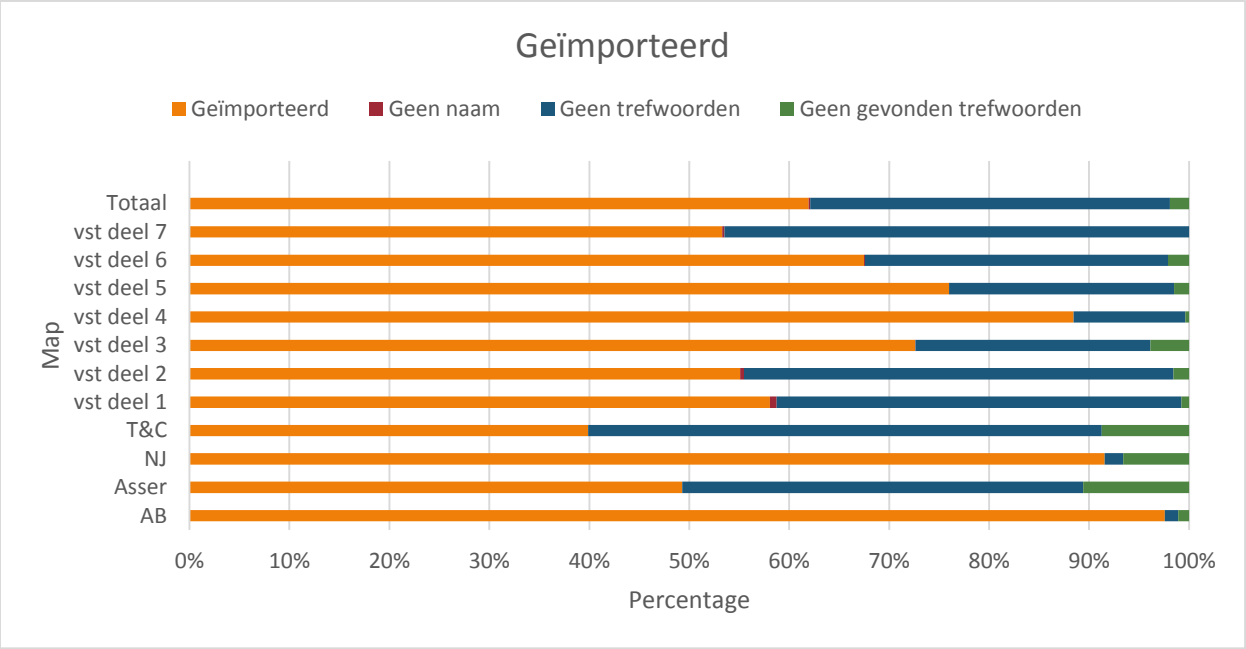
MAP NAAM	NODE
AB	JurisprudentieBewerkt
ASSER	Literatuur
NJ	JurisprudentieBewerkt
T&C	AangehaaktCommentaar
VST DEEL 1	VakstudieAC
VST DEEL 2	VakstudieAC
VST DEEL 3	VakstudieAC
VST DEEL 4	VakstudieAC
VST DEEL 5	VakstudieAC
VST DEEL 6	VakstudieAC
VST DEEL 7	VakstudieAC

Tabel 9 Importeer maping

Wanneer een document niet geïmporteerd mag worden kan dat komen door de volgende redenen:

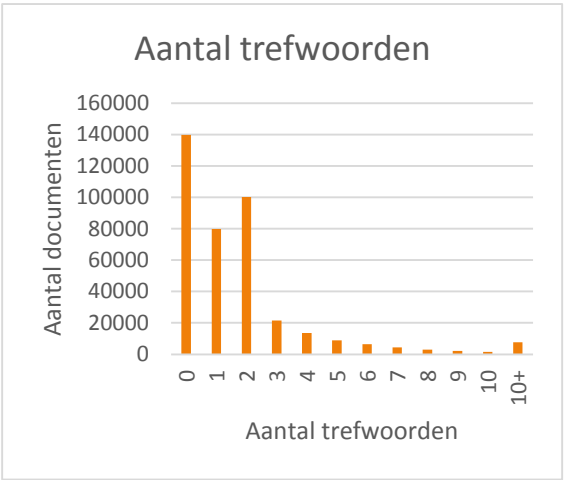
- **Geen naam is gevonden.** Elke node moet een naam hebben wanneer deze wordt aangemaakt. De waarde van het XML element md:vindplaats is gebruikt voor de naam. Wanneer dit element niet aanwezig is dan is het document niet geïmporteerd.
- **Geen trefwoorden.** Wanneer een document geen toegewezen trefwoorden heeft is het geen bruikbaar document om te gebruiken. Want KNN suggereert de trefwoorden aan de hand van de toegekende trefwoorden van lijkende documenten. Wanneer een lijkend document geen trefwoorden heeft dan kunnen er ook geen trefwoorden gesuggereerd worden.
- **Geen gevonden trefwoorden.** De trefwoorden die aan documenten zijn toegewezen komen uit de KBT. Wanneer voor een document wel trefwoord(en) zijn gevonden in diens XML maar de trefwoord(en) zijn niet te vinden in de KBT dan is het document ook niet geïmporteerd. Want uit steekproeven is gebleken dat het document dan trefwoorden heeft die niet toewijsbaar zijn en wat dus eigenlijk niet mag.

In figuur 29 is per map en voor het totaal aangegeven hoeveel procent van de documenten geïmporteerd zijn. En zo niet welke reden dit dat heeft.



Figuur 29 Geïmporteerde documenten

Ook is gekeken hoeveel trefwoorden elk document heeft. Hierbij is enkel gekeken hoeveel trefwoorden in de XML voorkomen en niet hoeveel daadwerkelijk zijn geïmporteerd. Want het aantal trefwoorden per document dat is geïmporteerd is bij het volgende onderdeel gemeten. In figuur 30 is het diagram weergegeven waarin staat hoeveel documenten 0, 1, 2, 3, etc. trefwoorden hebben.



Figuur 30 Trefwoorden per document

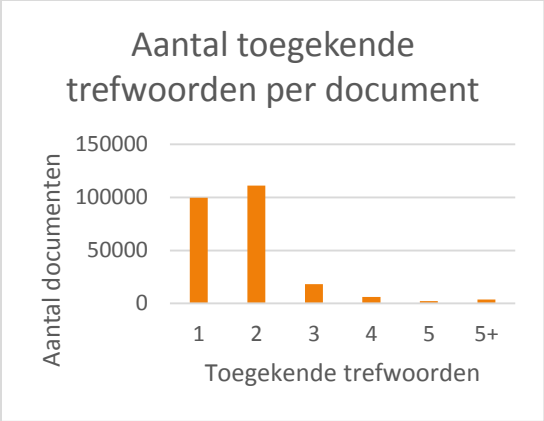
13.2 KNN ANALYSE

Bij de KNN analyse is geanalyseerd hoeveel al toegekende trefwoorden door KNN gesuggereerd worden. Dit is uitgevoerd voor de geïmporteerde documenten per set. De sets zijn bepaald door alle documenten te selecteren onder een bepaalde node. De sets staan onder de nodes waarnaar is geïmporteerd en zijn: “AangehaaktCommentaar”, “JurisprudentieBewerkt”, “Literatuur” & “VakstudieAC”.

Set	Aantal documenten
JurisprudentieBewerkt	5167
Literatuur	6377
AangehaaktCommentaar	9751
VakstudieAC	219445

Tabel 10 Aantal documenten per set

Bij doorlopen van de documenten is ook direct gemeten hoeveel trefwoorden elk document heeft toegekend. Dit is weergegeven in figuur 31 en hierbij is te zien dat de verdeling wel redelijk gelijk is als bij figuur 30 uit paragraaf 13.1. Maar het aantal documenten met meer 4 trefwoorden is in verhouding een stuk minder.



Figuur 31 Trefwoorden per geïmporteerd document

Voor figuur 32 en tabel 11 zijn niet alle documenten gebruikt want zoals te zien in tabel 10 is de set Vakstudie AC velen malen groter dan de andere sets. En om te zorgen dat de conclusie niet te sterk wordt gevormd naar één set zijn ‘slechts’ 10.000 documenten van VakstudieAC gebruikt.

Voor het analyseren zijn de documentsets twee keer doorlopen. Bij de eerste keer “Specifiek” heeft KNN alleen naar gelijkende document gezocht binnen dezelfde set. Dus voor een document uit VakstudieAC wordt alleen naar lijkende documenten gezocht binnen de set VakstudieAC. En bij de tweede keer “Kluwer” heeft KNN naar gelijkende documenten gezocht binnen de gehele documentenset van Kluwer.

Aantal gevonden trefwoorden	Specifiek aantal documenten	Kluwer aantal documenten
0	2317	2317
1	241	239
2	240	230
3	327	297
4	620	590
5	381	325
6	363	342
7	410	367
8	411	358
9	485	444
10	25500	25786

Tabel 11 aantal gevonden trefwoorden

Ook al is voor elk document geprobeerd om 10 trefwoorden te vinden lukt dit niet altijd. Dit komt doordat KNN zoekt naar de 25 meest lijkende documenten. En van de 25 gevonden documenten de meest voorkomende trefwoorden suggereert. Echter kan het zo zijn dat binnen de 25 gevonden document minder dan 10 verschillende trefwoorden toegewezen zijn. In dat geval worden de trefwoorden gesuggereerd die gevonden zijn ook al zijn dit er minder dan 10. Ook komt het voor dat KNN geen lijkende documenten kan vinden, in dat geval worden geen trefwoorden gesuggereerd. In tabel 11 staat weergeven hoe vaak het voorkwam dat 0, 1, 2, etc. trefwoorden gevonden zijn.

Om weer te geven hoeveel van de toegekende trefwoorden van een document zijn gesuggereerd door KNN. Is gebruik gemaakt van de waarden precision, recall en F-score. Voor het berekenen van deze waarden wordt het volgende gebruikt:

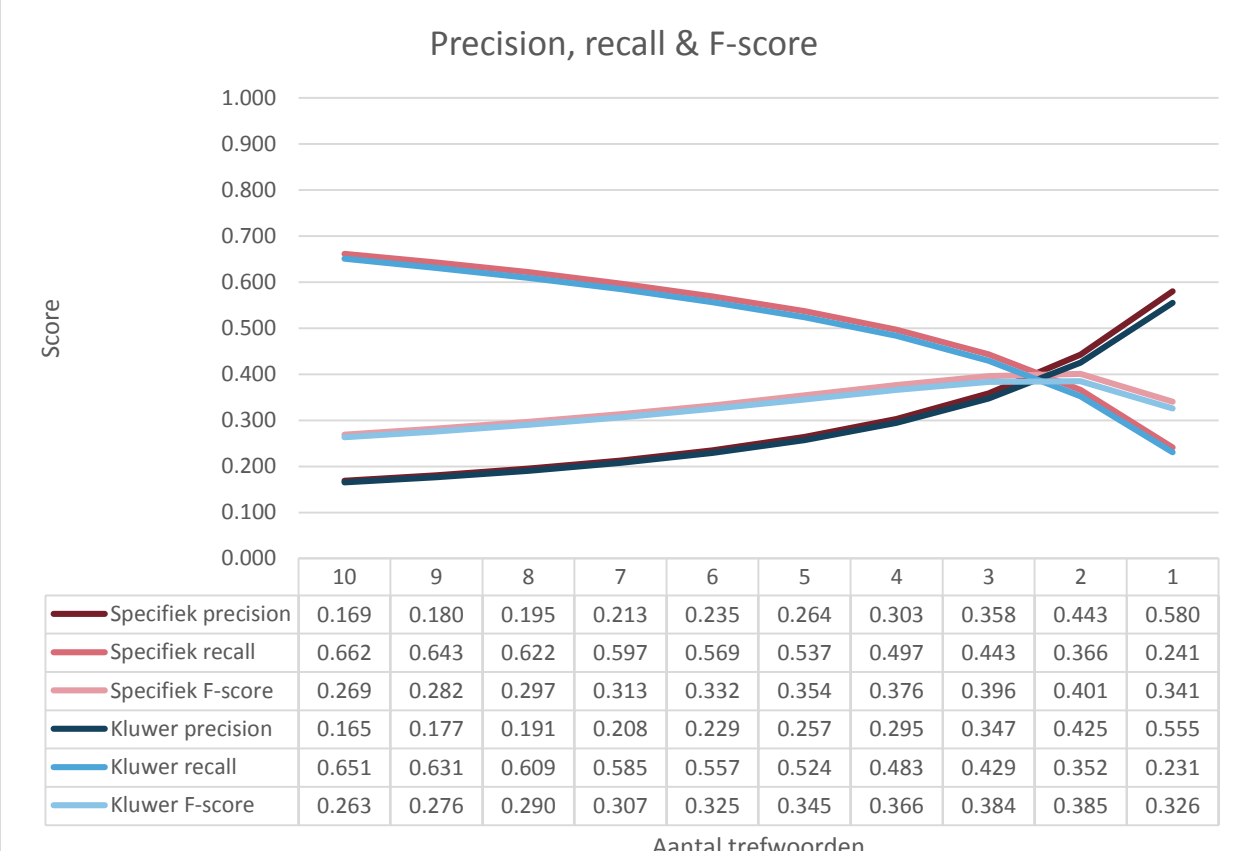
- TP (True positive) Het aantal trefwoorden die wel zijn toegekend en ook gesuggereerd.
- FP (False positive) Het aantal trefwoorden die niet zijn toegekend maar wel gesuggereerd.
- FN (False negative) Het aantal trefwoorden die wel zijn toegekend maar niet gesuggereerd.

De precision geeft weer hoeveel van de gesuggereerde trefwoorden ook toegekend zijn. Bij 1 zijn alle suggesties toegekend en bij 0 geen één. De precision wordt berekend door TP / (TP + FP).

De recall geeft weer hoeveel van de toegekende trefwoorden zijn gesuggereerd. Bij 1 zijn alle toegekende trefwoorden gesuggereerd bij 0 zijn geen van de toegekende trefwoorden gesuggereerd. De recall wordt berekend door TP / (TP + FN).

De F-score is een combinatie van de precision en recall om te voorkomen dat continue twee getallen vergeleken moeten worden.
De F-score wordt berekend door 2 * ((precision * recall) / (precision + recall))

In figuur 32 zijn de waarden(precision, recall & F-score) in een grafiek gezet. Hierbij worden de waarden weergegeven op de y-as. En de x-as geeft weer hoeveel trefwoorden gesuggereerd worden. Dus bij 10 op de x-as staan de waarden als 10 trefwoorden gesuggereerd worden en bij 1 op de x-as de waarden als 1 trefwoord gesuggereerd wordt.

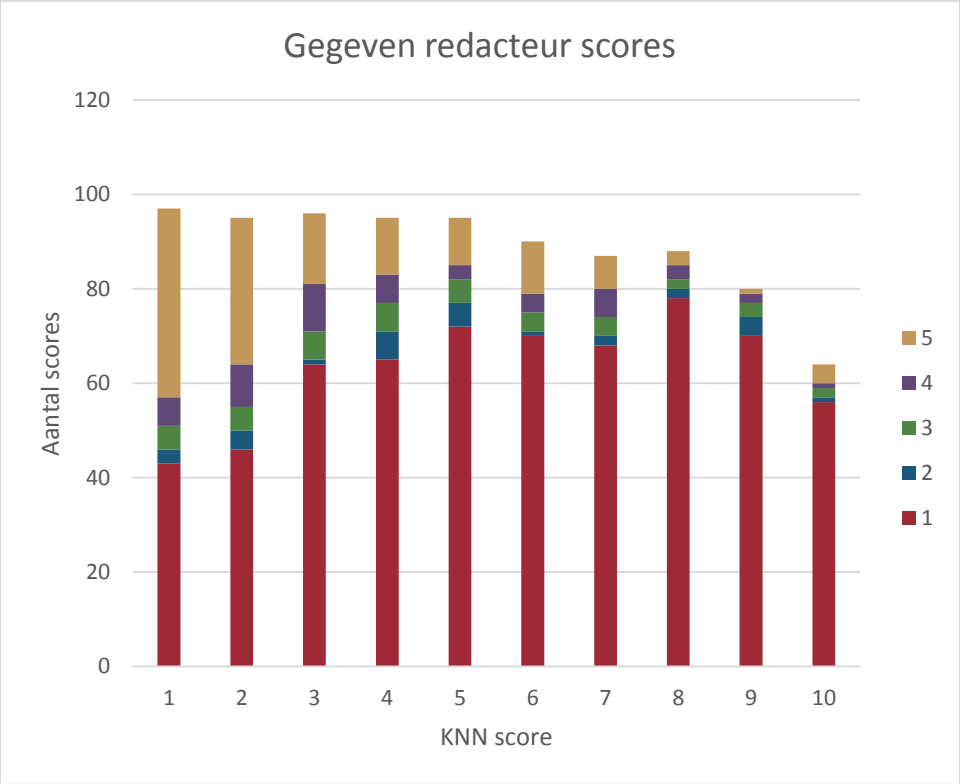


	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.169	0.180	0.195	0.213	0.235	0.264	0.303	0.358	0.443	0.580
Specifiek recall	0.662	0.643	0.622	0.597	0.569	0.537	0.497	0.443	0.366	0.241
Specifiek F-score	0.269	0.282	0.297	0.313	0.332	0.354	0.376	0.396	0.401	0.341
Kluwer precision	0.165	0.177	0.191	0.208	0.229	0.257	0.295	0.347	0.425	0.555
Kluwer recall	0.651	0.631	0.609	0.585	0.557	0.524	0.483	0.429	0.352	0.231
Kluwer F-score	0.263	0.276	0.290	0.307	0.325	0.345	0.366	0.384	0.385	0.326

Figuur 32 precision, recall & F-score

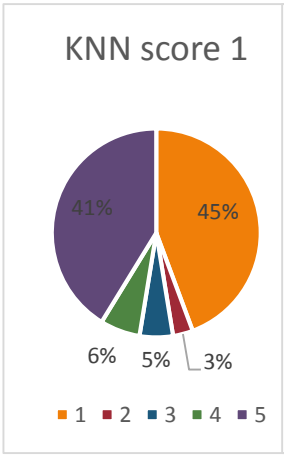
13.3 REDACTEUR ANALYSE

Van Kluwer hebben drie verschillende redacteuren elk 35 documenten beoordeeld. Voor elk document heeft KNN 10 trefwoorden gesuggereerd en de auteur heeft de trefwoorden beoordeeld met een score van 1 tot 5. De resultaten hiervan zijn weergegeven in figuur 33. De KNN score op de x-as geeft aan op welke positie KNN het trefwoord suggereerde, hierbij is de eerste positie het beste trefwoord en de tiende positie het minste van de 10.

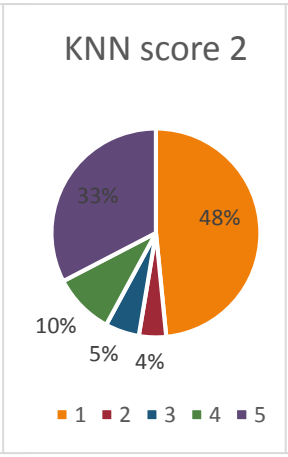


Figuur 33 Redacteuren resultaat

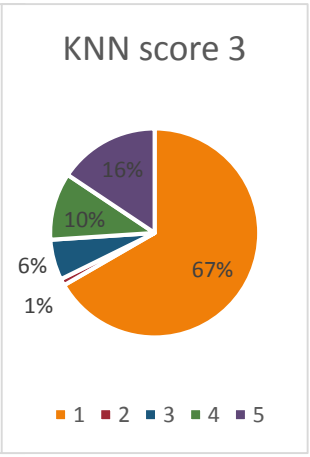
Het totaal aantal loopt lichtelijk af doordat KNN niet voor elk document 10 trefwoorden kan vinden. Het is te zien dat bij de eerst twee KNN scores het merendeel van de trefwoorden een score hoger dan 1 heeft gekregen. Dit duidt erop dat in meer dan de helft van de gevallen de trefwoorden op de eerste en tweede positie bruikbaar zijn. Vanaf de derde score loopt het aantal keer dat de score 1 is toegekend hard op. Om het verschil tussen de eerste twee en de derde positie goed aan te geven, zijn de gegeven scores hiervoor ook in een taart diagram weergegeven (figuur 34, 35 & 36).



Figuur 34 KNN score 1

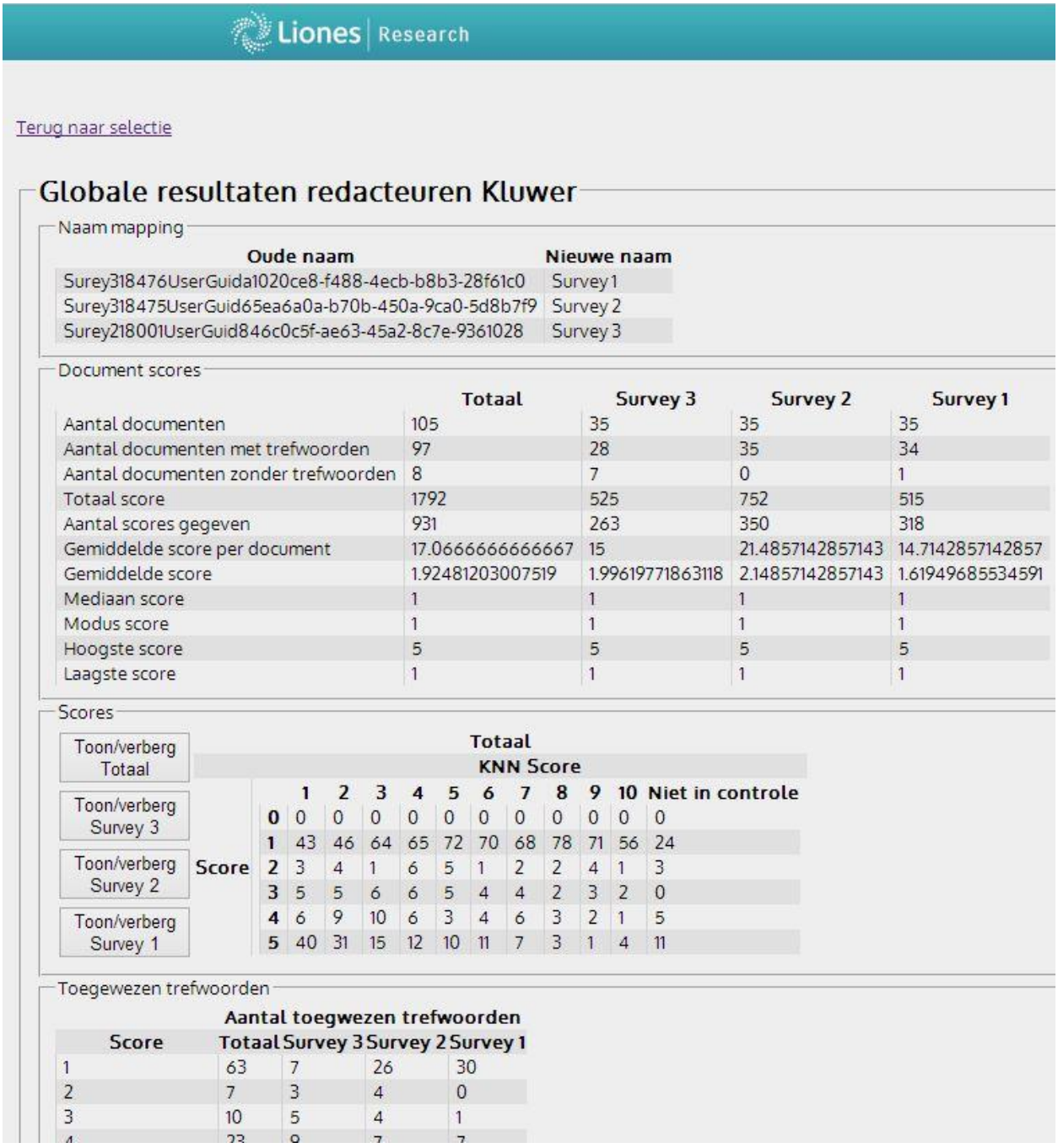


Figuur 35 KNN score 2



Figuur 36 KNN score 3

Ook is voor het analyseren van de resultaten van de redacteuren een tool ontwikkeld. Deze zal gebruikt worden door de contactpersoon van Kluwer om gedetailleerder te analyseren waarom deze scores zijn gehaald. De tool bestaat uit twee onderdelen, één onderdeel voor de globale analyse en één onderdeel voor de detail analyse. De globale analyse te zien in figuur 37 geeft de totaal informatie van welke scores zijn gegeven, hoe de reeds toegekende trefwoorden door de redacteuren zijn beoordeeld en hoeveel suggesties door de redacteuren hebben toegevoegd.



Figuur 37 Globale analyse tool

De detail analyse is te zien in figuur 38 geeft weer het document waarbij de trefwoorden zijn gezocht, welke trefwoorden hierbij zijn gevonden en welke KNN score deze hebben en de 25 gelijkende documenten die zijn gevonden met de similarity score. De similarity score is de score waarmee Elasticsearch aangeeft hoe goed de document overeen komen. Zoals te zien in figuur 38 kunnen de documenten ook in deze tool geopend worden om zo de documenten met elkaar te kunnen vergelijken of deze inderdaad met elkaar overeen komen.

Figuur 38 Detail analyse tool

13.4 Conclusie

Om ervoor te zorgen dat alleen goede trefwoorden gesuggereerd worden. Moeten alleen de eerste twee trefwoorden gebruikt worden. Dit is te zien aan de KNN analyse waarbij de F-score het hoogste is bij twee trefwoorden. En ook bij de redacteuren analyse scoren de eerste twee trefwoorden veruit het beste.

Van de KNN analyse was al te verwachten dat de beste score gehaald zou worden bij twee trefwoorden. Want de meeste documenten hebben twee trefwoorden toegekend. En wanneer een document maar 2 trefwoorden heeft toegekend en 10 trefwoorden worden gesuggereerd dan is de hoogste precision die gehaald kan worden 0,2. En ook al is de recall wel het hoogste bij 10 trefwoorden, de F-score wordt het sterkste beïnvloed door de laagste score (precision of recall).

Wel is het opvallend dat de analyse van de redacteuren ook uitkomt op twee trefwoorden. Hiervoor is geen directe link te leggen met het aantal toegewezen trefwoorden.

Overigens is in de KNN analyse te zien dat de gesuggereerde trefwoorden beter zijn wanneer alleen documenten uit dezelfde set gebruikt worden. Hierbij komt het wel iets vaker voor dat er minder dan 10 trefwoorden gevonden worden. Maar omdat dat het belangrijk is dat alleen goede suggesties gebruik worden maakt dit uit. Verder is te zien dat KNN niet veel beter presteert bij gebruik van documenten uit dezelfde set. Dit is slechts werk van een paar procent.

14. EVALUATIE

In dit hoofdstuk is het proces en de producten geëvalueerd. Bij het proces zijn de uitgevoerde onderdelen behandel. Bij de producten zijn de producten behandeld die gedurende de afstudeerperiode zijn gemaakt. En er is aangegeven welke beroepstaken zijn uitgevoerd gedurende de afstudeerperiode.

14.1 PROCES

Over het gehele proces ben ik tevreden. In deze paragraaf zal ik de onderdelen bespreken die tijdens de afstudeerperiode zijn uitgevoerd.

Inlezen
Het beginnen met inlezen bleek goed te werken omdat hierdoor veel inzicht is verkregen in de te gebruiken algoritmes en taxonomieën. Hierdoor was het tijdens de rest van het project niet nodig om nogmaals veel te besteden aan informatie opzoeken.

Aanpak
Het opstellen van het plan van aanpak is goed verlopen. Hierbij is de opdracht ook een stuk duidelijker geworden waardoor het ook makkelijker werd om de requirements op te stellen.
Over de gekozen ontwikkelmethode ben ik ook zeer tevreden. Want door met SCRUM te werken kon goed omgegaan worden met een opdracht die in het begin nog niet geheel duidelijk was. En ook konden zo extra bijkomende eisen goed verwerkt worden zoals het algoritme TextRank en het uitvoeren van de analyses.

Requirements
Bij het opstellen van de requirements heb ik veel zelf gedaan en wellicht iets te veel. Zo was het beter geweest wanneer ik de opdrachtgever en stakeholder meer had betrokken bij het prioriteren van de requirements.

Modellering
Het ontwerpen van de applicatie en het ontwikkelen hiervan is goed verlopen. Wel was het beter geweest als ik eerder met een collega hiernaar had gekeken want zoals te lezen in hoofdstuk 10.2 zijn de importeerfuncties nog niet geheel naar mijn wens.

Ontwikkeling
Het ontwikkelen is goed verlopen. In het begin ging het wel aanzienlijk langzamer dan later in het project doordat het gebruikte Lynkx framework nog onbekend was. Maar zeker toen ik beter bekend was met het framework bleek het zeer prettig om mee te werken.

Testen
Het testen is ook goed verlopen al had ik hier wel liever meer tijd aan besteed dan nu mogelijk was. Ook heeft hierbij meegespeeld dat het ontwikkelen van de Unit Testen meer tijd heeft gekost dan verwacht door het gebruik van het Lynkx framework.

Analyse
Het analyseren is goed verlopen waarbij nuttige resultaten zijn behaald. Het analyseren van de documenten was het gemakkelijkste omdat hiervoor alleen de documenten doorlopen hoeven worden waarbij bepaalde velden worden geteld. De KNN analyse was een stuk lastiger en duurde ook een stuk

langer om uit te voeren. Tijdens het uitvoeren heb ik nog wat problemen gehad met Elasticsearch waardoor ik deze opnieuw moest installeren wat een hoop tijd kostte. De redacteuren analyse is ook goed verlopen al was het niet mogelijk om ook nog de gedetailleerde analyse op te nemen in het rapport. Want de contactpersoon van Kluwer zal hier de komende weken mee bezig gaan.

14.2 PRODUCT

Voor de evaluatie van de producten zal ik de opgestelde documenten, applicatie en analyse bespreken.

Documenten
Tijdens dit project zijn de volgende documenten opgesteld: “plan van aanpak”, “requirements rapport”, “Module test rapport”, “Acceptatietest rapport” & “Analyse rapport”. Over het geheel ben ik wel tevreden over de opgestelde documenten.
Maar bij het plan van aanpak ben ik vergeten om daadwerkelijk op te schrijven welke ontwikkelmethode gebruikt wordt. Dit heeft echter geen negatieve invloed gehad omdat wel bekend was welke ontwikkelmethode ik zou gebruiken.
Het requirements rapport is in het begin opgesteld en was hierbij zeker nuttig om duidelijk te krijgen wat de te ontwikkelen applicatie moest kunnen. Wensen die later in het project erbij zijn gekomen, zijn hier niet alsnog in bijgeschreven maar in plaats daarvan zijn deze eisen bijgehouden in de backlog.
De test rapporten zijn voornamelijk gebruikt om de testen op te stellen. De resultaten die hierin staan hebben niet direct iets toegevoegd aan het project en zijn er enkel om aan te tonen welke resultaten uit de testen zijn gekomen.
Het analyse rapport is goed ontvangen door de opdrachtgever en stakeholder en ikzelf ben hier ook tevreden over. Dit rapport geeft weer hoe goed de documentenset van Kluwer is gemetadateerd en hoe goed KNN heeft gepresteerd met deze documentenset.

Applicatie
De ontwikkelde applicatie voldoet aan alle van tevoren opgestelde eisen. Het enige wat ik hieraan nog zou willen wijzigen zijn de importeerfuncties waarvoor ik nu een beter idee heb zoals al eerder is beschreven in hoofdstuk 10.2. Ook de opdrachtgever is ook tevreden met de ontwikkelde applicatie. En wil deze waarschijnlijk ook later weer gebruiken om een aangepaste versie van KNN of een soortgelijk algoritme te testen.

Ook de stakeholder (contactpersoon van Kluwer) is tevreden met de applicatie. De enige aanmerking is dat hij graag de optie had gezien om een trefwoord als compleet fout aan te geven in plaats van deze één ster te geven.

Analyse
Ook het analyseren is goed verlopen en hier is een nuttig resultaat uit gekomen. De stakeholder was ook zeer tevreden met de ontwikkelde tool voor het inzien van de details en zal hier de komende tijd nog mee bezig gaan. Wel is er de wens dat bij de gelijkende documenten ook wordt aangegeven welke trefwoorden deze hebben toegewezen. Dit zal alsnog worden ingebouwd in de komende week net na de afstudeerperiode. Want de informatie hiervoor is zeer eenvoudig op te halen en dit kan dus snel worden ingebouwd.

14.3 BEROEPSTAKEN

Voor de afstudeerperiode is aangegeven welke beroepstaken uitgevoerd zullen worden. Hierbij is ook voor elke beroepstaak aangeven op welk niveau deze is uitgevoerd. De niveaus hiervoor worden bepaald aan de hand van tabel 12. Alle beroepstaken zijn zelfstandig uitgevoerd en bij het bespreken van de beroepstaak is aangegeven hoe de complexiteit is bepaald.

Context	*Taakrol*		
	Geleid	Zelfstandig	Sturend
Simpel	1	2	3
Lastig	2	3	4
Complex	3	4	5

Tabel 12 beroepstaken niveaus

1.4 Uitvoeren analyse door definitie requirements

Deze beroepstaak is zelfstandig uitgevoerd met de complexiteit lastig. De definitie lastig voor deze beroepstaak zoals beschreven in (De Haagse Hogeschool, 2009) is als volgt.

"Het betreft één applicatie met een groot aantal requirements. Er is sprake van een beperkt aantal stakeholders met tegengestelde wensen en eisen t.o.v. de applicatie. Er is één set aan requirements, die gedurende de analyse kan veranderen."

Bij het opstellen van de requirements zijn een aantal requirements gekomen van de opdrachtgever (software architect binnen Liones) en een aantal van de stakeholder (contactpersoon van Kluwer). Voor de opdrachtgever is het belangrijk dat de ontwikkelde applicatie verschillende algoritmen kan gebruiken zodat het voor meerdere klanten gebruikt kan worden. Voor de stakeholder is dit niet van belang en gaat het er voornamelijk om dat er een applicatie is waarmee de redacteuren gemakkelijk het algoritme KNN kunnen beoordelen. En dat geanalyseerd wordt hoe goed KNN werkt voor de documentenset van Kluwer.

Ook kunnen er requirements op een later moment bij komen. Dit is ook gebeurd. Een voorbeeld hiervan is de requirement: "Als gebruiker wil ik overig commentaar kunnen invullen voor een document uit de enquête.".

Deze beroepstaak is uitgevoerd op niveau 3 zoals ook is aangegeven in het afstudeerplan.

3.2 Ontwerpen systeemdeel

Deze beroepstaak is zelfstandig uitgevoerd met de complexiteit lastig tot complex.

De definitie van lastig zoals beschreven in (De Haagse Hogeschool, 2009) is als volgt.

"Het betreft het ontwerpen (structuur en gedrag) van een objectgeoriënteerde applicatie, met behulp van een ontwerpmethodiek en een tool. Hierbij wordt rekening gehouden met toekomstige wijzigingen, testbaarheid en hergebruik."

De definitie van complex zoals beschreven in (De Haagse Hogeschool, 2009) is als volgt.

"Het betreft het ontwerpen van een gedistribueerde applicatie of een applicatie met complexe algoritmiek. Er wordt rekening gehouden met niet-functionele kwaliteitsattributen en beveiligingsaspecten en gebruik gemaakt van design patterns."

Bij het ontwerpen van het systeemdeel is veel rekening gehouden met herbruikbaarheid en uitbreidbaarheid. Ook zijn meerdere design patterns gebruikt en is rekening gehouden met non-functionele requirements zoals de gebruiksvriendelijkheid en de performance. Ook bevat de applicatie complexe algoritmiek in de vorm van de algoritmes KNN en TextRank. Omdat het een web applicatie

betreft is deze ontwikkeld op een lokaal systeem waarna deze naar een server is geüpload om gebruikt te worden. Het enige onderdeel uit complex waar geen rekening mee is gehouden zijn beveiligingsaspecten.

In het afstudeerplan staat dat deze beroepstaak zal worden uitgevoerd op niveau 3. Dit is uiteindelijk uitgevoerd op niveau 3-4.

3.3 Bouwen applicatie

Deze beroepstaak is zelfstandig uitgevoerd met de complexiteit complex. De definitie complex voor deze beroepstaak zoals beschreven in (De Haagse Hogeschool, 2009) is als volgt.

"De applicatie sluit aan op bestaande software of maakt gebruik van een bestaand Framework. Het bouwen gebeurt in een geavanceerde ontwikkelomgeving inclusief testomgeving en versiebeheertool."

Bij het ontwikkelen van de applicatie is veel aandacht besteed aan de herbruikbaarheid en uitbreidbaarheid. Ook is gebruik gemaakt van het framework Lynkx en het ontwikkelen is gedaan op een lokaal systeem. De unit testen zijn ook uitgevoerd op het lokale systeem maar hiervoor is wel een aparte database opgezet. En de acceptatietesten zijn uitgevoerd op een staging omgeving. De staging omgeving is gelijk aan de productieomgeving want na het testen is de applicatie ook gebruikt op de staging omgeving gezien het geringe aantal gebruikers. Voor versiebeheer is gebruik gemaakt van SVN.

Deze beroepstaak is uitgevoerd op niveau 4 zoals ook is aangegeven in het afstudeerplan.

3.5 Testen

Deze beroepstaak is zelfstandig uitgevoerd met de complexiteit lastig. De definitie lastig voor deze beroepstaak zoals beschreven in (De Haagse Hogeschool, 2009) is als volgt.

"Bij het opstellen van het logisch testontwerp wordt gebruik gemaakt van een testontwerptechniek. Er is aandacht voor herhaalbaarheid van de testen. Het betreft hoofdzakelijk het testen van de functionaliteit. Testrapportage betreft het volledige systeem."

Voor het testen van de importeerfuncties is gebruik gemaakt van een algoritmetest. Hiervoor zijn procesdiagrammen, logische testontwerpen en fysieke testontwerpen opgesteld. Aan de hand van de testontwerpen zijn Unit Testen geschreven zodat deze eenvoudig vaker kunnen worden uitgevoerd.

Deze beroepstaak is uitgevoerd op niveau 3 zoals ook is aangegeven in het afstudeerplan.

Overig

De ontwikkelde analyse tool is niet meegenomen in het beschrijven van de beroepstaken. En ook het analyseren van het algoritme KNN staat niet bij de beroepstaken omdat hiervoor geen beroepstaak staat gedefinieerd in (De Haagse Hogeschool, 2009).

15. BIBLIOGRAFIE

Allemang, D., & Hendler, J. (2011). *Semantic Web for the Working Ontologist.* Morgan Kaufmann.

Beiske, K. G. (2013, 11). *Similarity in Elasticsearch*. Opgehaald van Hosted Elasticsearch by Found: http://www.found.no/foundation/similarity/

Bindinc. (2013, 11 28). *Over Bindinc.* Opgehaald van Bindinc: http://www.bindinc.nl/pagina/over-ons

Bishop, J. (2008). *C# 3.0 Design Patterns.* O'Reilly.

Cheng, H., Tan, P.-N., & Jin, R. (2007). Opgehaald van SIAM: Data Mining Proceedings: http://www.siam.org/proceedings/datamining/2007/dm07_045cheng.pdf

Chou,, C.-H., Kuo, B.-H., & Chang, F. (sd). Opgehaald van Institute of Information Science: http://www.iis.sinica.edu.tw/papers/fchang/2248-F.pdf

De Haagse Hogeschool. (2009). *Beroepstaken van de opleiding Informatica.* Den Haag.

EBU. (2013, February). *tech3293v1_4.pdf.* Opgehaald van EBU Technology & Innovation - EBU CORE: http://tech.ebu.ch/docs/tech/tech3293v1_4.pdf

Elasticsearch - Wikipedia, the free encyclopedia. (2013, 11 28). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Elasticsearch

Elasticsearch. (2013, 09). *More Like This API*. Opgehaald van Elasticsearch: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/search-more-like-this.html

He, X., & Niyogi, P. (2003). *Machinelearning.* Opgehaald van Machine Learning: http://machinelearning.wustl.edu/mlpapers/paper_files/NIPS2003_AA20.pdf

Hedden, H. (2012). *The Accidental Taxonomist.* Information Today, Inc.

Isaac, A. (2011, September 21). *Isaac.* Opgehaald van DublinCore: http://dcevents.dublincore.org/IntConf/dc-2011/paper/view/69/36

Kluwer. (2013, 11 28). *Organisatie - Kluwer*. Opgehaald van Kluwer: overkluwer.kluwer.nl/organisatie

k-nearest neighbors algortihm - Wikipedia, the free encyclopedia. (2013, 09). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/KNN#Algorithm

LinkedDataTools. (2013, 09). *Introducing Linked Data And The Semantic Web*. Opgehaald van LinkedDataTool: http://www.linkeddatatools.com/semantic-web-basics

Liones. (2013, 11 28). *Fonto XML*. Opgehaald van Fonto XML: http://www.fontoxml.com/

Liones. (2013, 10). *Lynkx Documentatie - Lynkx developers documentatie*. Opgehaald van Lynkx Documentatie: http://documentatie.lynkx-staging.nl/Default.lynkx?id=162

Metadata standards - Wikipedia. (2013, 09). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Metadata_standards

Middel, M. (2013). *Automatisch suggereren van trefwoorden.*

Mihalcea, R., & Tarau, P. (2004). *TextRank: Bringing Order into Texts.* Opgehaald van ACL Anthology: http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf

Mirkes, E. (2013, 09). *KNN and Potential Energy: applet.* Opgehaald van University of Leicester: http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html

Multisystems. (2013, 09). *multites.com - Home*. Opgehaald van multites.com: http://www.multites.com/index.htm

RDF Schema - Wikipedia. (2013, 09). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/RDFS

Resource Description Framework - Wikipedia. (2013, 09). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Resource_Description_Framework

Sayad, D. S. (2013, 09). *KNN Classification*. Opgehaald van Data Mining Map: http://www.saedsayad.com/k_nearest_neighbors.htm

Smartlogic. (2013, 09). *Semaphore's Ontology Manager for Building, Enchancin and Browsing Semantic Models*. Opgehaald van Smartlogic: http://www.smartlogic.com/home/products/semaphore-modules/ontology-manager/ontology-manager-overview

Storojev, A. (2010, July). *Extraction of Topic Cloud Sets from Collections of Text Documents and Spreadsheets.* Opgehaald van UvA DARE: http://dare.uva.nl/document/199328

Sutton, O. (2012, Februari). *Introduction to kNN Classification and CNN Data Reduction.* Opgehaald van University of Leicester/Alexander N. Gorban's Page: http://www.math.le.ac.uk/people/ag153/homepage/KNN/OliverKNN_Presentation.pdf

Tester, D. (2013, 09). *RDFS vs. OWL - Cambridge Semantics*. Opgehaald van Cambridge Semantics: http://www.cambridgesemantics.com/nl/semantic-university/rdfs-vs.-owl

The Apache Software Foundation. (2013, 11 29). *Apache Lucene - Apache Lucene Core*. Opgehaald van Apache Lucene: http://lucene.apache.org/core/

The Apache Software Foundation. (2013, 11 29). *Apache OpenNLP Developer Documentation*. Opgehaald van Apache OpenNLP: http://opennlp.apache.org/documentation/1.5.3/manual/opennlp.html#tools.sentdetect

W3C. (2013, 09). *SKOS Simple Knowledge Organization System*. Opgehaald van World Wide Web Consortium (W3C): http://www.w3.org/2004/02/skos/

Web Ontology Language - Wikipedia. (2013, 09). Opgehaald van Wikipedia: http://en.wikipedia.org/wiki/Web_Ontology_Language

Plan van aanpak

Afstudeeropdracht ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *20-09-2013*

Versie: *1.0*

01. INHOUDSOPGAVE

2 Inleiding 1
3 Aanleiding 1
4 Opdracht 2
5 Aanpak 3
5.1 Voorbereiding 3
5.2 Modelering 3
5.3 Realisatie 3
6 Producten 4
7 Risico's 4
8 Planning 5

02. INLEIDING

Dit plan van aanpak is geschreven voor de afstudeeropdracht “Ontwikkeling document trefwoordsuggesties” bij Liones. Hierin zal de aanleiding voor de afstudeeropdracht worden toegelicht, wat de opdracht zal inhouden, wat de aanpak zal zijn voor dit project, welke producten hiervoor zullen worden opgeleverd, welke risico’s er bij dit project zijn en wat de planning voor dit project is.

03. AANLEIDING

Het bedrijf Kluwer is een uitgever voor juridische documenten. Hierbij worden er trefwoorden toegekend aan de documenten zodat deze relatief eenvoudig terug te vinden zijn. Het toewijzen van trefwoorden wordt momenteel nog gedaan door het handmatig selecteren van de trefwoorden uit de thesaurus.

De thesaurus met trefwoorden wordt bijgehouden door Kluwer en heet de Kluwer Brede Thesaurus (KBT). Deze thesaurus is bestaat momenteel uit ongeveer zeventienduizend trefwoorden. Deze thesaurus bestaat uit: rubrieken, vakgebieden, hoofdonderwerpen en trefwoorden. Hierbij kunnen de trefwoorden ook weer relaties met elkaar hebben wat dit een ingewikkelde thesaurus maakt.

Om het toekennen van de trefwoorden aan documenten gemakkelijker te maken voor de auteurs is er onderzoek gedaan naar verschillende algoritmes voor het classificeren van documenten. Aan de hand van dit onderzoek is de keuze gemaakt om KNN (K-Nearest Neighbors) hiervoor te gaan gebruiken. Echter is het nodig om de resultaten van het algoritme te laten beoordelen door vak specialisten om zo beter inzicht te krijgen in hoe goed het algoritme werkt en hoe dit verbeterd kan worden.

De ontwikkelen van een dergelijke tool past ook goed binnen de roadmap van FontoXML. Daarbij zijn services die de auteur helpen om de content juist te meta dateren van groot belang. De te ontwikkelen tool zal Liones helpen om te onderzoeken of verschillende algoritmes goed weren in de praktijk of dat er nog verbeteringen en/of tuning nodig is.

04. OPDRACHT

Voor het beoordelen van de algoritmes zal een enquête systeem ontwikkeld worden waarmee kan worden bepaald hoe goed algoritmes in de praktijk werken. Hierbij is het enquête systeem gericht op algoritmes die suggesties kunnen geven op basis van een stuk tekst/document. Deze algoritmes zullen als invoer dus tekst/documenten nodig hebben maar kunnen ook nog verdere invoer nodig hebben voor het geven van de suggesties.
Voorbeeld:
KNN(wordt gebruikt om trefwoorden te suggereren voor Kluwer) heeft naast het stuk tekst/document een andere set documenten nodig waar al trefwoorden zijn toegewezen. Want op basis van deze set zal het de suggesties bepalen.

Voor het bepalen hoe goed algoritmes werken zal een set documenten worden samengesteld en voor elk document zullen suggesties worden gegeven. De gebruiker die de enquête invult kan bij elke suggestie aangeven hoe goed deze past bij het document. De suggesties moeten door verschillende algoritmes gegenereerd kunnen worden.
Voorbeeld:
Voor Kluwer zal het systeem een document analyseren en hiervoor tien trefwoord suggereren door middel van KNN.
En voor een andere klant zal het systeem een document analyseren en hiervoor vijf samenvattingen suggereren door middel van een algoritme.

De resultaten zullen worden opgeslagen in een database waarna deze gebruikt kunnen worden voor het genereren van een rapport waarin de algoritme(s) beoordeeld worden. Maar het zal ook mogelijk genoeg inzicht geven om de algoritme(s) verder te tunen zodat deze beter zullen presteren. En wanneer meerdere algoritmes in een enquête gebruikt zijn kunnen deze met elkaar vergeleken worden om zo te bepalen welke hiervan het beste presteert.

05. AANPAK

Er zal één systeem worden ontwikkeld dit systeem zal voor meerdere klanten gebruikt kunnen worden om algoritmes te beoordelen. Voor het tot stand komen van dit systeem zal de volgende aanpak gevolgd worden.

05.1 VOORBEREIDING

Inlezen
De uitvoerende zal beginnen met inlezen om een goed idee te krijgen wat voor taxonomieën er zijn en hoe deze werken. Verder zal de uitvoerende ook het rapport doornemen van het onderzoek dat is uitgevoerd voor Kluwer waarbij het algoritme KNN is gekozen.

Plan van aanpak
Het plan van aanpak zal worden opgesteld zodat er duidelijk wordt wat er gedaan moet worden en hoeveel tijd hier per onderdeel ongeveer beschikbaar zal zijn.

Eisen opstellen
De (functionele en non-functionele) eisen zullen worden opgesteld aan de hand van eisen die worden aangeleverd. Hiernaast zal er nog worden onderzocht of er nog meer eisen zijn die nog onbekend of onbenoemd zijn.
Tijdens dit proces zullen ook user stories worden opgesteld van de eisen.

05.2 MODELERING

Use case(s)
Voor de user stories die verduidelijking kunnen gebruiken zullen use case beschrijvingen worden opgesteld. Wanneer dit veel use cases worden zal er ook een use case diagram worden opgesteld.

Klassendiagram
Er zal een klassendiagram op analyse niveau worden opgesteld. Hierin zullen de klassen worden genoemd en de belangrijkste functies en attributen die zij hebben. Gedurende het project zal dit klassendiagram worden bijgewerkt zodat het een goede representatie blijft van het systeem.

05.3 REALISATIE

Bouwen
Het systeem zal worden ontwikkeld in C# en zal gebruik maken van het CMS Framework Lynkx.

Testen
Voor het ontwikkelde systeem zal een testplan worden opgesteld waarna deze zal worden uitgevoerd. Aan de hand van de testresultaten zal een testrapport worden opgesteld.
Hierna zal er nog een acceptatietest worden uitgevoerd om te bepalen of alle eisen zijn verwerkt in het ontwikkelde systeem.

06. PRODUCTEN

De stappen die zijn beschreven in hoofdstuk 5 Aanpak zullen verschillende producten produceren. Deze producten zijn hieronder opgesomd.

- Plan van aanpak
- Eisen (functionele en non-functionele)
- User stories
- Use case beschrijving(en)
- (Optioneel) use case diagram
- Klassendiagram
- Code (systeem)
- Testplan
- Testrapport

07. RISICO'S

De risico's die hieronder zijn beschreven kunnen zich mogelijk voordoen bij gedurende dit project. Hierbij is per risico beschreven hoe groot de kans is dat het zich zal voordoen, hoe groot de impact is wanneer dit zich voordoet, welke maatregelen er genomen worden om deze te voorkomen en wat er gedaan zal worden wanneer deze zich toch voordoen.

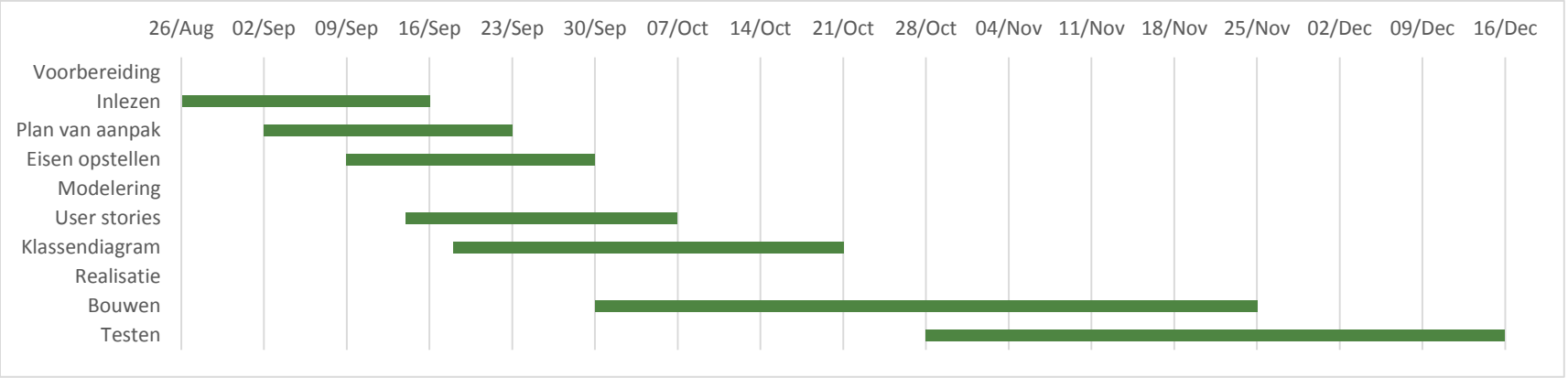
Deze lijst met risico's zal wanneer nodig gedurende het project worden uitgebreid.

Falen van hardware
Kans: Laag. De kans dat de hardware faalt is zeer laag omdat de hardware zeer nieuw is (op moment van schrijven ongeveer één maand) en hardware is erop berekend om jaren mee te gaan.
Impact: Hoog - laag. Wanneer de data schijf faalt kan de gemaakte progressie verloren gaan waardoor een hoop tijd verloren gaat. Als Hierdoor teveel progressie verloren gaat kan dit leiden tot het falen van het project. Wanneer andere onderdelen falen zal dit minder impact hebben omdat dit slechts vertraging oplevert doordat er vervanging van hardware nodig is maar er gaat geen progressie verloren.
Maatregel voorkomen: Er zal regelmatig (minimaal wekelijks) een back-up hierbij wordt de code geüpload naar een SVN en documenten naar Dropbox. Hierdoor zal bij het falen van hardware niet zoveel progressie verloren gaan dat kan leiden tot het falen van het project. Deze maatregel zal de impact van dit risico verlagen van hoog – laag naar laag.
Maatregel wanneer het voorkomt: Wanneer de hardware faalt zal een back-up worden terug gezet naar een tijdelijke machine totdat er vervangende hardware is. Hierdoor zal er geen werktijd verloren gaan.

08. PLANNING

Hieronder volgt een globale planning voor dit project. Hierbij zijn de onderdelen genomen die eerder zijn benoemd in hoofdstuk 5 Aanpak en hiervoor zijn tijden bepaald wanneer hieraan gewerkt zal worden.

	Start date	***Duration (days)***	***End date***
Voorbereiding			
Inlezen	26-08-13	21	16-09-13
Plan van aanpak	02-09-13	21	23-09-13
Eisen opstellen	09-09-13	21	30-09-13
Modelering			
User stories	14-09-13	23	07-10-13
Klassendiagram	18-09-13	33	21-10-13
Realisatie			
Bouwen	30-09-13	56	25-11-13
Testen	28-10-13	49	16-12-13



Eisen

Afstudeeropdracht ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *20-09-2013*

Versie: *1.0*

01. INHOUDSOPGAVE

2 Inleiding 1
3 Omschrijving opdracht 1
4 Functionele eisen 2
5 Non-functionele eisen 2
6 Afbakening 4
6.1 Algoritme types 4
7 User stories 5
8 Use cases 5

02. INLEIDING

Dit document beschrijft aan welke eisen het systeem moet voldoen dat ontwikkeld zal worden tijdens de afstudeeropdracht “ontwikkeling document classificatie” bij Liones. Hierbij zijn de eisen onderverdeeld tussen de functionele en non-functionele eisen. Alle eisen hebben een code gekregen zodat deze eenvoudig terug te vinden zijn wanneer hiernaar wordt gerefereerd.

De functionele eisen hebben een code gekregen van F[x] waarbij [x] een oplopend nummer is.
En de non-functionele eisen hebben een code gekregen van N[x] waarbij [x] een oplopend nummer is.

03. OMSCHRIJVING OPDRACHT

Er zal een applicatie worden ontwikkeld waarmee kan worden bepaald hoe goed verschillende algoritmes presteren. Dit zal worden gedaan door middel van een enquête waarbij gebruiker kunnen aangeven hoe goed de gegenereerde gesuggereerde passen bij documenten en welke suggesties zij in de selectie missen. Deze tool moet inzetbaar zijn voor meerdere klanten en moet dus van verschillende algoritme implementaties gebruik kunnen maken. Zo moet het voor Kluwer door middel van KNN trefwoorden bij documenten suggereren en voor Bindinc moet het voor een document samenvattingen genereren.

De gebruikers krijgen in het systeem een set documenten die zij kunnen beoordelen. Voor de gebruikers van Kluwer is het ook van belang dat de gebruiker alleen documenten krijgt waar hij/zij vanaf weet en zal dus niet documenten krijgen van allerlei verschillende rubrieken. Van de set documenten zal er maar één tegelijk getoond worden beginnend bij de eerste. Wanneer de gebruiker de gesuggereerde trefwoorden of samenvattingen heeft beoordeeld kan hij/zij doorgaan naar het volgende document. Wanneer de gebruiker alle documenten heeft afgewerkt is de enquête klaar.

In de volgende eisen zal er worden gesproken over keywords. Hiermee wordt er gesproken over de trefwoorden die voor Kluwer worden gegenereerd maar óók over andere mogelijke stukken die door een algoritme kunnen worden gezocht of gegenereerd zoals een samenvatting.
Verder zal er gesproken worden over gebruikers. Hiermee worden de gebruikers van de enquête bedoeld, dus de personen die de enquête zullen invullen.
Een beheerder is een persoon die een enquête kan opstellen en de resultaten hiervan mag inzien.

04. FUNCTIONELE EISEN

De volgende functionele eisen worden gesteld aan het te ontwikkelen systeem tijden de afstudeeropdracht “ontwikkeling document classificatie”.
De eisen zijn geprioriteerd om duidelijk te maken wat er echt moet gebeuren en welke eisen een lagere prioriteit hebben. Dit zal ook bepalen aan welke onderdelen eerst gewerkt zal worden.

De eisen hebben één van de volgende scores gekregen:
[C] Cruciaal – *Als deze eis niet is verwerkt dan kan het project niet succesvol worden afgerond.*

[H] Hoog – *Deze eisen zullen veel invloed hebben als deze wel of niet zijn verwerkt. Echter is het systeem nog bruikbaar wanneer deze eisen niet zijn verwerkt.*

[M] Medium – *Wanneer deze eisen niet zijn verwerkt dan zal dit de bruikbaarheid van het systeem schaden. Echter zal het systeem nog steeds bruikbaar zijn.*

[L] Laag – *Wanneer deze eisen zijn verwerkt voegen ze wat toe aan het systeem. Echter zal bij het ontbreken van deze eisen het systeem nog steeds goed bruikbaar zijn. De gebruiker zal geen ongemak ondervinden door het ontbreken van de eisen.*

Eis	**Prioriteit**	**#**
F16 Het systeem moet de keywords en documenten kunnen importeren.	C	01
F07 Bij elk document zal de gebruiker X keywords zien die door een geïmplementeerd algoritme worden gesuggereerd. Per enquête kan de X verschillen. *Voor de gehele enquête zal de X hetzelfde zijn.*	C	02
F08 Bij elk keyword kan de gebruiker één van de volgende scores aanklikken: “Zeer goed, Goed, Matig, Slecht, Zeer slecht”.	C	03
F11 De antwoorden van de gebruiker zullen in een database worden opgeslagen.	C	04
F04 Een gebruiker krijgt een set van X documenten waar X per enquête instelbaar is. *Als de documenten in een hiërarchische taxonomie geplaatst zijn dan kan de set worden beperkt tot document uit een bepaalde tak.*	C	05
F09 De gebruiker kan per document in een tekstveld invullen welk(e) keyword(s) er bij de suggesties mist(en).	H	06
F14 Een beheerder kan van een enquête een overzicht opvragen. *Voor meer details zie hoofdstuk 7 Afbakening.*	H	07
F15 Een beheerder kan van een enquête de details opvragen. *Voor meer details zie hoofdstuk 7 Afbakening.*	H	08
F02 Het systeem kan gebruik maken van verschillende algoritmes voor het suggereren van keywords.	H	09
F01 Het systeem moet om kunnen gaan met verschillende taxonomie standaarden. Welke standaarden hiervoor worden gekozen zal later worden bepaald. *Dit is nodig zodat het systeem met documenten en/of keywords kan omgaan die in een taxonomie zijn ondergebracht of aan een taxonomie zijn gekoppeld.*	H	10
F05 De set documenten zal een willekeurige set documenten zijn. *Als de documenten in een hiërarchische taxonomie geplaatst zijn dan kan de set worden beperkt tot document uit een bepaalde tak.*	M	11
F03 Verschillende algoritmes kunnen voor één document worden gebruikt. *Voorbeeld: [gebruiker krijgt bij document X 10 keywords getoond. Hiervan zijn 5*	M	12

bepaald door algoritme A en 5 zijn bepaald door algoritme B]. Deze aantallen zullen per enquête instelbaar zijn.		
F06 De gebruiker zal maar één document uit de set tegelijk zien.	M	13
F10 De gebruiker kan alleen naar het volgende document d.m.v. de knop “Volgende” wanneer hij/zij alle keywords heeft beoordeeld. (Bij elk keyword een score heeft aangeklikt.)	M	14
F12 Wanneer een enquête gedeeltelijk wordt ingevuld dan zullen de ingevulde antwoorden worden opgeslagen.	M	15
F13 Meerdere gebruikers kunnen dezelfde enquête invullen. Echter zal elke gebruiker een willekeurige set documenten krijgen zoals genoemd in eis F05. *Ook wanneer dezelfde gebruiker een enquête opnieuw invult zal hij/zij een nieuwe willekeurige set documenten krijgen.*	L	16

05. NON-FUNCTIONELE EISEN

De volgende non-functionele eisen worden gesteld aan het te ontwikkelen systeem tijden de afstudeeropdracht “ontwikkeling document classificatie”.
Deze zijn op dezelfde manier geprioriteerd als de functionele eisen.

Eis	**Prioriteit**	**#**
N01 Wanneer een gebruiker een document opent moet de lijst keywords hiervoor binnen vijf seconden geladen zijn.	M	1
N02 Gebruiksvriendelijk voor de gebruikers. Zij moeten de tool zonder uitleg binnen 1 minuut begrijpen en kunnen gebruiken.	M	2

06. AFBAKENING

Voor een aantal eisen is een afbakening opgesteld om duidelijker te maken wat deze eis wel en niet inhoudt. Hierbij wijzen de nummers terug naar degene gebruikt in hoofdstuk 5 Functionele eisen en in hoofdstuk 6 Non-functionele eisen.

F02 – Het systeem kan met de algoritme types omgaan die staan genoemd in hoofdstuk 7.1 Algoritme types. Voor elk algoritme uit één van deze types zal een implementatie geschreven moeten worden.

F03 – Per enquête zal maar één algoritme type gebruikt worden. Het is dus niet mogelijk om in één enquête keywords te suggereren uit een taxonomie en op basis van tekst.

F14 – Het overzicht zal de volgende onderdelen weergeven:
- Door hoeveel gebruikers de enquête is ingevuld.
- Hoeveel algoritmes zijn gebruikt en welke.
- Hoeveel keywords elk algoritme heeft bepaald.
- Per algoritme hoeveel keer elke score uit eis F08 is gegeven.
- Hoeveel missende keywords zijn ingevuld.

F15 – De details zullen per document worden weergegeven en de volgende onderdelen weergeven:
- Het document.
- Welke/of er keywords al aan het document waren toegekend.
- Per algoritme welke keywords zijn gegenereerd.
- Welke score elk keyword heeft gekregen.
- Welke missende keywords zijn ingevuld.

F16 – Wanneer er geïmporteerd moet worden dan moet hiervoor in code een mapping geschreven worden. Het systeem kan de volgende standaarden importeren, dit kan later worden uitgebreid met meer standaarden.
- XML

06.1 ALGORITME TYPES

- Keyword suggesties uit een taxonomie.
- Keyword suggestie op basis van tekst.
- Selectie met volgorde van zinnen ter behoeve van een samenvatting.
- Extracted entities.
- Related content.
- Taal classificatie.
- Sentiment.

07. USER STORIES

Voor de user stories zijn twee verschillende actoren. De gebruiker is de persoon die de enquête zal invullen en de beheerder is de persoon die een enquête kan opstellen en de resultaten kan inzien.

US01. Als gebruiker die de enquête start wil ik zien voor hoeveel documenten ik keywords kan beoordelen.
US02. Als gebruiker wil ik gevonden keywords voor een document kunnen beoordelen.
US03. Als gebruiker wil ik een eigen suggesties voor keywords kunnen geven.
US04. Als gebruiker wil ik naar een volgend document in de enquête kunnen gaan.
US05. Als gebruiker wil ik gedurende de enquête kunnen zien hoever ik in de enquête nog moet.
US06. Als gebruiker wil ik de enquête vroegtijdig kunnen beëindigen.
US07. Als beheerder wil ik eenvoudig een enquête kunnen opstellen door aan te geven welk algoritme(s) gebruikt moet worden, uit hoeveel documenten de enquête bestaat en hoeveel keywords er bij elk document worden gezocht.
US08. Als beheerder wil ik voor elke enquête een overzicht kunnen opvragen wat weergeeft: hoeveel personen de enquête hebben ingevuld en hoe goed het algoritme(s) heeft gepresteerd.

08. USE CASES

Enkel voor de user stories waarbij verduidelijking nodig is zijn use case beschrijvingen opgesteld.

NAAM	**BEOORDEEL KEYWORDS**
DOEL	De gebruiker beoordeeld keywords
UITVOERENDE	Gebruiker
OMSCHRIJVING	- De gebruiker leest het document door (linkerkant van het scherm) - De gebruiker selecteert voor elk keyword (rechterkant van het scherm) hoe goed deze bij het document past door een radio button aan te klikken. - Wanneer de gebruiker voor alle keywords een radio button heeft aangeklikt klikt hij/zij op de knop “Volgende”. - De ingevulde waarden worden opgeslagen in een database.
RESULTAAT	Voor een document zijn X keywords beoordeeld.

Detail testplan importeerfuncties

Afstudeeropdracht ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *22-11-2013*

Versie: *1.0*

01. INHOUDSOPGAVE

2 Inleiding 1
3 Doel 1
4 Testbasis 1
5 Teststrategie 1
5.1 ImportTag 1
5.2 KbtXmlImporter 4
5.3 ThesaurusImporter 8
5.4 LoadTermIds 11
6 Testuitvoering 15
7 Conclusie 15
Appendix A: Testresultaten 15

02. INLEIDING

Dit document beschrijft de moduletest voor het importeren van een thesaurus in het project "Ontwikkeling document trefwoordsuggesties". Hierbij zal worden toegelicht wat er wordt getest, hoe dit wordt getest, de resultaten van het testen en de conclusie die daaruit getrokken kan worden.

03. DOEL

Deze moduletest is ervoor om te bepalen of dat de importeerfuncties voor een thesaurus correct werken.

04. TESTBASIS

Voor deze test zal de KBT (Kluwer Brede Thesaurus) als voorbeeld genomen worden. Hierbij zal de mapping en de KluwerKbtImporter gebruikt worden. Maar de daadwerkelijk KBT zal niet gebruikt worden. In plaats daarvan zal XML worden opgesteld die de delen van de KBT zal simuleren.

05. TESTSTRATEGIE

In deze test zal de importeerfunctie voor de KBT getest worden. Het zal getest worden door middel van een algoritmetest met testmaat-1 ('branch coverage').

Het importeerproces voor de KBT is opgedeeld in 4 onderdelen om te voorkomen dat 1 moduletest te groot wordt waardoor het lastiger wordt om te overzien. Voor elke onderdeel is een procesdiagram opgesteld en aan de hand daarvan zijn de te doorlopen paden bepaald waarvan weer het fysiek testontwerp is opgesteld.

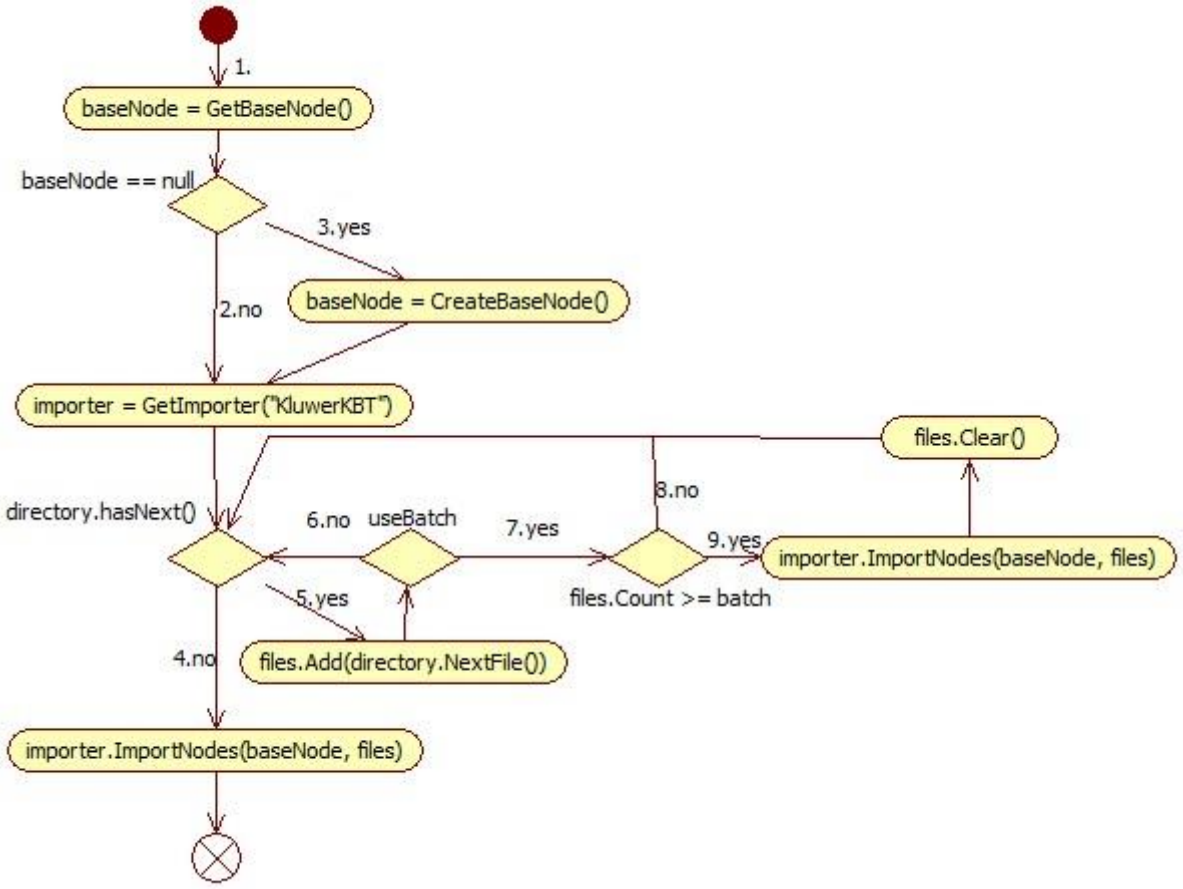
05.1 IMPORTTAG

Het eerste deel uit de importeerfunctie is verantwoordelijk voor het ophalen of aanmaken van de hoofdnode en het uitlezen van de files.

De hoofdnode is de node waaronder de nodes die worden aangemaakt geplaats zullen worden.

Als input krijgt dit onderdeel de gewenste naam van de hoofdnode. Naar deze naam zal gezocht worden om op te halen wanneer deze bestaat of een nieuwe zal met deze naam worden aangemaakt.
En als input heeft dit onderdeel ook het padnaam waar de te importeren bestand(en) zich bevind(en).

Procesdiagram



Logische testontwerpen

Logisch testontwerp 01	
Procesdiagram pad	1-2-5-7-8-5-7-9-4
Aantal XML bestanden	2
Aantal termen in XML1	1
baseNode	Bestaat nog niet.
File1-Term1	Valide
File2-Term1	Valide

Logisch testontwerp 02	
Procesdiagram pad	1-3-5-6-4
Aantal XML bestanden	1
Aantal termen in XML1	1
baseNode	Bestaat wel.
File1-Term1	Valide

Fysieke testontwerpen

Fysiek testontwerp 01	
Lynkx call	`<import dir="C:\\TestData\\ImportTag01" baseNode="TestImportTag01" fileExt="xml" type="KluwerKBT" batch="2" />`

XML 01	`<?xml version="1.0" encoding="utf-8" ?>`
	`<kbt>`
	`<thesaurus>`
	`<concept>`
	`<descriptor>TestTerm01</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#01</id>`
	`<flg>A</flg>`
	`</concept>`
	`</thesaurus>`
	`</kbt>`

XML 02	`<?xml version="1.0" encoding="utf-8" ?>`
	`<kbt>`
	`<thesaurus>`
	`<concept>`
	`<descriptor>TestTerm02</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#02</id>`
	`<flg>A</flg>`
	`</concept>`
	`</thesaurus>`
	`</kbt>`

Database	Node TestImportTag01 mag niet aanwezig zijn in de database.
Verwacht resultaat	De node “TestImportTag01” is aangemaakt. Onder de node “TestImportTag01” zijn 2 nodes met type “ThesaurusTerm” aangemaakt. De eerste node heeft de naam “TestTerm01”. De tweede node heeft de naam “TestTerm02”.

Fysiek testontwerp 02	
Lynkx call	`<import dir="C:\\TestData\\ImportTag02" baseNode="TestImportTag02" fileExt="xml" type="KluwerKBT" batch="0" />`

XML	`<?xml version="1.0" encoding="utf-8" ?>`
	`<kbt>`
	`<thesaurus>`
	`<concept>`
	`<descriptor>TestTerm01</descriptor>`
	`<niveau>TH Thesaurusterm</niveau>`
	`<id>101#01</id>`
	`<flg>A</flg>`
	`</concept>`
	`</thesaurus>`
	`</kbt>`

Database	Node TestImportTag02 moet wel aanwezig zijn in de database.
Verwacht resultaat	Onder de al bestaande node “TestImportTag02” is 1 node met type “TestImportTag02” aangemaakt. De node heeft de naam “TestTerm01”

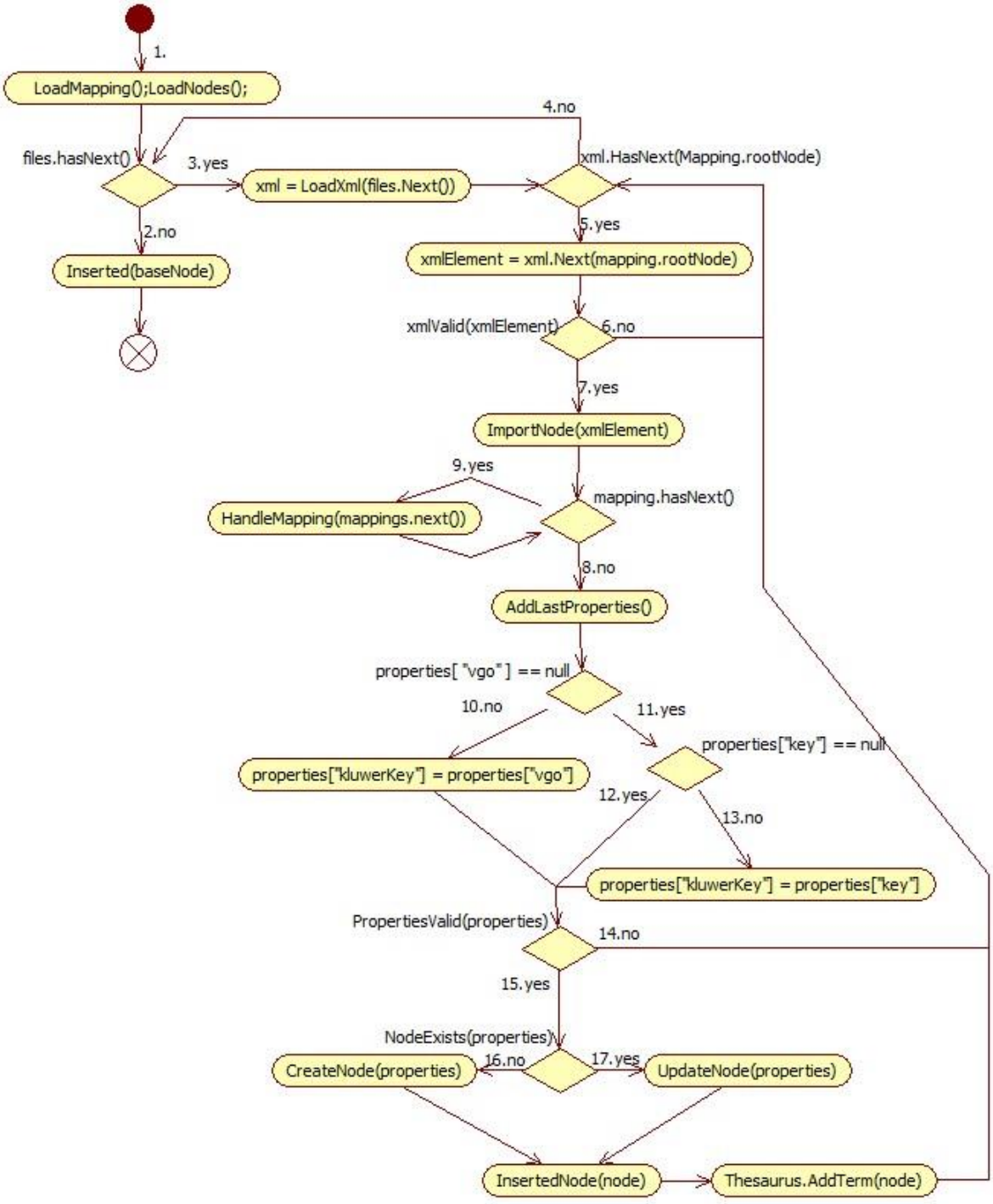
05.2 KBTXMLIMPORTER

Dit tweede onderdeel is specifieker per import. Hier wordt de XmlImporter en de KluwerKbtImporter gebruikt. De XmlImorter zal verantwoordelijk zijn voor het uitlezen van de XML en hiervan een node maken. De KluwerKbtImporter zal verantwoordelijk zijn voor het aanmaken van ThesaurusTerms.

De XmlImporter werkt met behulp van een mapping welke aan het begin wordt geladen. Bij deze mapping staat welke XML elementen vertaald moeten worden naar welke attributen van de node.

Voor het aanmaken van de ThesaurusTerm gebruikt de KluwerKbtImorter de aangemaakte node. Dankzij de mapping bevat deze de benodigde informatie om zijn broader, narrower en related termen te bepalen. In de mapping is dit opgeslagen als een CSV en kan dus eenvoudig vertaald worden naar een lijst die de ThesaurusTerm kan gebruiken.

Procesdiagram



Logisch testontwerp

Logisch testontwerp 01	
Procesdiagram pad	1-3-5-6-5-7-9-8-10-14-5-7-9-8-11-12-15-16-5-7-9-8-11-13-15-17-4-2
Aantal termen	4
Term1 (5-6)	XML is NIET valid.
Term2 (5-7-9-8-10-14)	XML is valid. Heeft 1 mapping. Properties[“vgo”] != null. Properties is NIET valid.
Term3 (5-7-9-8-11-12-15-16)	XML is valid. Heeft 1 mapping. Properties[“vgo”] = null. Properties[“key”] = null. Properties is valid Node bestaat nog niet.
Term4 (5-7-9-8-11-13-15-17)	XML is valid Heeft 1 mapping. Properties[“vgo”] = null. Properteis[“key”] != null. Properties is valid. Node bestaat al.

Fysiek testontwerp

Dit onderdeel kan getest worden door middel van een Unit test. Hierbij moet het deel van de ImportTag overgenomen worden om zo de importer aan te maken, een dictionary met de files vullen en de importeerNodes functie aanroepen van de importer.

<table>
<tr><td colspan="2">Fysiek testontwerp 01</td></tr>
<tr><td>XML</td><td><pre>
<?xml version="1.0" encoding="utf-8" ?>
<kbt>
<thesaurus>
<concept>
 <descriptor>TestTerm01</descriptor>
 <niveau>TH Thesaurusterm</niveau>
 <id>101#01</id>
 <flg>B</flg>
</concept>
<concept>
 <niveau>TH Thesaurusterm</niveau>
 <id>101#2</id>
 <vgo>vgo01</vgo>
 <flg>A</flg>
</concept>
<concept>
 <descriptor>TestTerm03</descriptor>
 <niveau>TH Thesaurusterm</niveau>
 <id>101#</id>
 <flg>A</flg>
</concept>
<concept>
 <descriptor>TestTerm04</descriptor>
 <niveau>TH Thesaurusterm</niveau>
 <id>101#04</id>
 <flg>A</flg>
</concept>
</thesaurus>
</kbt>
</pre></td></tr>
<tr><td>Database</td><td>Node met naam “TestTerm03” en key “101#” mag niet aanwezig zijn in de database.
Node met naam “TestTerm04” en key “101#04” moet wel aanwezig zijn inde database.</td></tr>
<tr><td>Verwacht resultaat</td><td>Onder de node “TestKbtXmlImporter” zijn de volgende nodes aangemaakt/gewijzigd.
Aangemaakt
Naam: “TestTerm03”, Key: “101#”, Description: “”
Gewijzigd
Naam: “TestTerm04”, Key: “101#4”, Description: “04”

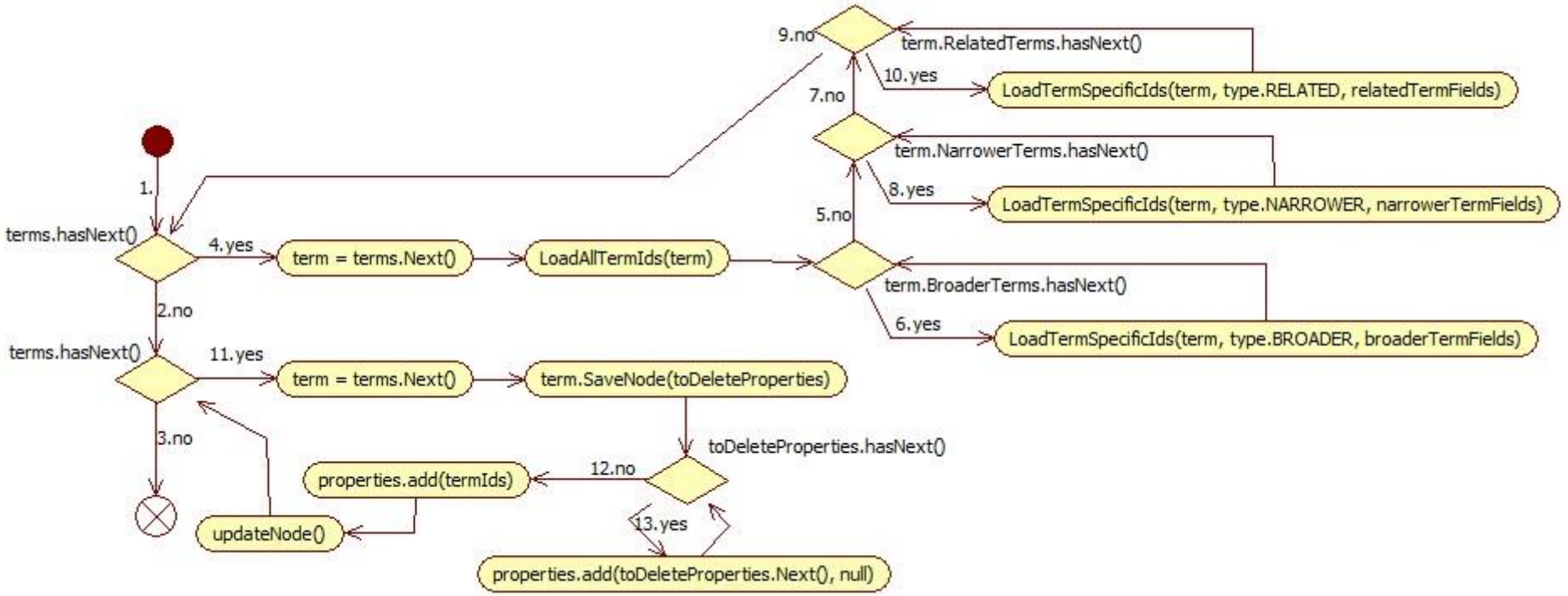
De eerste XML term is niet valide want bij flg staat B.
De tweede XML is wel valide maar zijn properties zullen invalide zijn omdat deze geen naam heeft.</td></tr>
</table>

05.3 THESAURUSIMPORTER

Dit onderdeel ervoor verantwoordelijk dat elke ThesaurusTerm zijn verwijzingen naar andere ThesaurusTerms omzet naar een verwijzing naar het id. En dat alle nodes van de ThesaurusTerms gewijzigd worden zodat de nieuwe verwijzingen naar de ids worden opgeslagen en de oude verwijderd worden wanneer deze zijn opgegeven in de toDeleteProperties.

Als input heeft de ThesaurusImporter een lijst van ThesaurusTerms die is gevuld door de KluwerKbtImporter tijdens het importeren vanaf XML.

Procesdiagram



Logisch testontwerp

Logisch testontwerp 01	
Procesdiagram pad	1-4-6-5-8-7-10-9-2-11-13-12-3
Aantal termen	1
Term1 (4-6-5-8-7-10-9) (2-11-13-12)	term heeft 1 BroaderTerm(s) term heeft 1 NarrowerTerm(s) term heeft 1 RelatedTerm(s) 1 in toDeleteProperties

Fysiek testontwerp

De andere 3 nodes moeten ook worden toegevoegd omdat de bt, nt & rt daarnaartoe moeten verwijzen.

ThesaurusImporter test pad 01	
XML	<?xml version="1.0" encoding="utf-8" ?> <kbt> <thesaurus> <concept> <descriptor>TestTerm01</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#01</id> <flg>A</flg> <bt>TestBroader01</bt> <nt>TestNarrower01</nt> <rt>TestRelated01</rt> </concept> <concept> <descriptor>TestBroader01</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#101</id> <flg>A</flg> </concept> <concept> <descriptor>TestNarrower01</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#201</id> <flg>A</flg> </concept> <concept> <descriptor>TestRelated01</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#301</id> <flg>A</flg> </concept> </thesaurus> </kbt>
Database	
Verwacht resultaat	Onder de node “TestThesaurusImporter” zijn de volgende nodes aangemaakt/gewijzigd. **Aangemaakt** Naam: “TestTerm01”, Key: “101#01”, bt: id van “TestBroader01”, nt: id van “TestNarrower01”, rt: id van “TestRelated01” Naam: “TestBroader01”, Key: “101#101”, nt: id van “TestTerm01” Naam: “TestNarrower01”, Key: “101#201”, bt: id van “TestTerm01” Naam: “TestRelated01”, Key: “101#301”, rt: id van “TestTerm01”

05.4 LoadTermIds

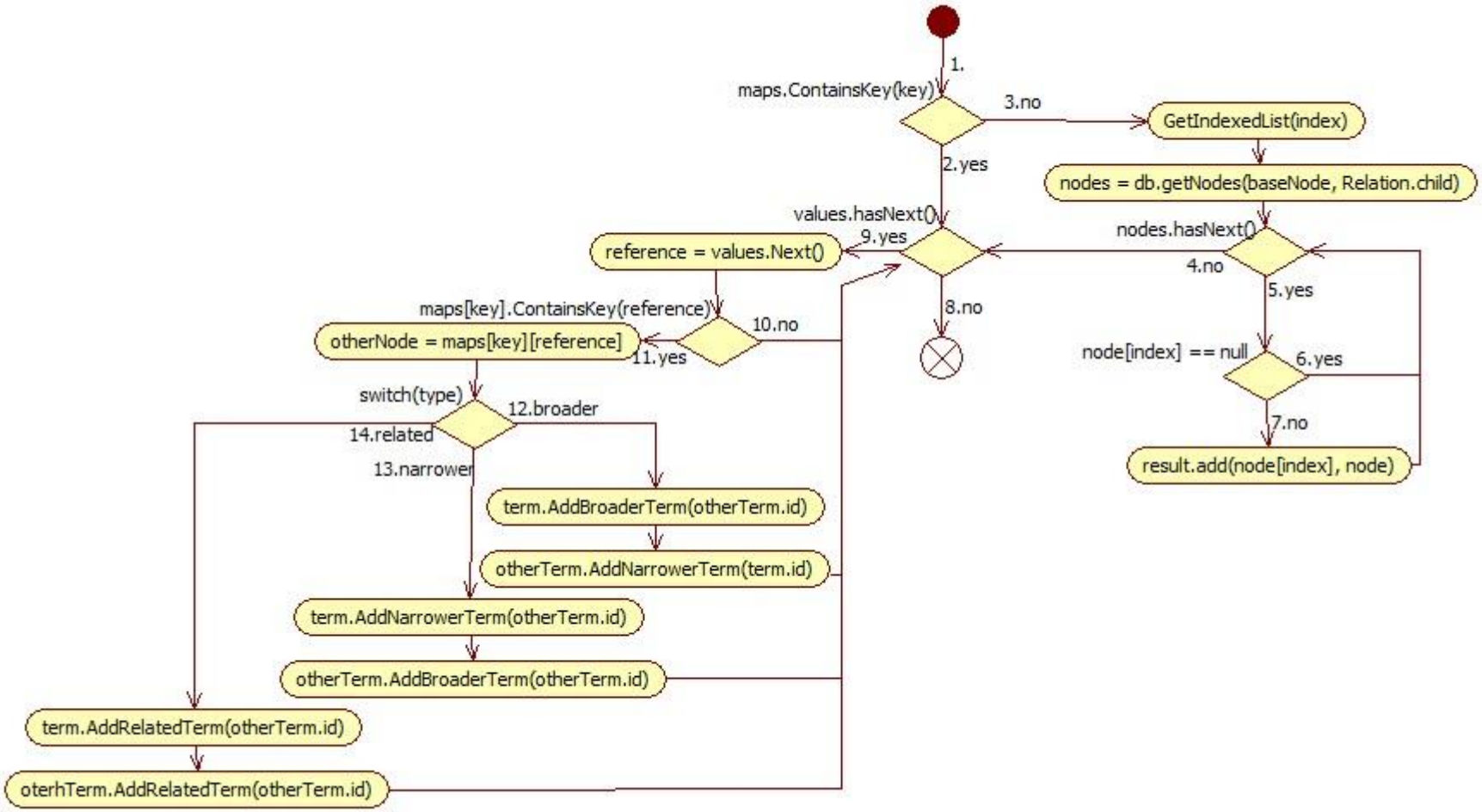
Dit laatste onderdeel is ervoor verantwoordelijk dat een ThesaurusTerm zijn verwijzingen naar andere ThesaurusTerms bijhoudt met een verwijzing naar het id van de ander ThesaurusTerm.

Als input heeft dit onderdeel de term waarvan de relaties gezet moeten worden, het type relaties (broader, narrower, related), de naam van het veld waar de waarden heen wijzen en de waarden die verwijzen naar de velden van andere ThesaurusTerms.

Als eerste wordt gekeken of er al een map is aangemaakt voor het betreffende veld.
Bijvoorbeeld: ThesaurusTerm X heeft verwijzingen naar broaderTerms op het veld “name”. Dan zal een map worden aangemaakt met alle ThesaurTerms met als index “name”. Hierdoor kan de ThesaurusTerm met “name” Y zeer snel gevonden worden.

Vervolgens worden de meegegeven waarden doorlopen en wordt elke ThesaurusTerm in de map gezocht. Vervolgens wordt de relatie voor beide ThesaursTerms gezet.

Procesdiagram



Logisch testontwerp

Het type dat binnenkomt blijft hetzelfde. De paden 12, 13 & 14 kunnen dus niet gecombineerd worden binnen 1 doorloop.

Logisch testontwerp 01	
Procesdiagram paden	1-3-5-7-5-6-9-11-12-9-10-8 1-2-9-11-13-8 1-2-9-11-14-8
Aantal termen in XML1	3
Term1 (1-3-5-7-5-6-4) (9-11-12) (9-10-8)	Map bestaat nog niet Twee nodes worden opgehaald onder de baseNode 1ste opgehaalde node bestaat niet in de map 2de opgehaalde node bestaat wel in de map (zelfde naam als 1ste) Term heeft 2 broaderTerms 1 broaderTerm bestaat in de map 1 broaderTerm bestaat niet in de map
Term2 (1-2-9-11-13-8)	Map bestaat al Term heeft 1 narrowerTerm narrowerTerm bestaat in de map
Term3 (1-2-9-11-14-8)	Map bestaat al Term heeft 1 relatedTerm relatedTerm bestaat in de map

Fysiek testontwerp

LoadTermIds test pad 01, 02 & 03	
Lynkx call	Niet nodig. Met code kan de xml opgebouwd worden.
XML file(s)	Niet nodig. Met code kan de xml opgebouwd worden.
LoadTermIds.xml inhoud	<?xml version="1.0" encoding="utf-8" ?> <kbt> <thesaurus> <concept> <descriptor>TestTerm01</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#01</id> <flg>A</flg> <bt>node01</bt> </concept> <concept> <descriptor>TestTerm02</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#02</id> <flg>A</flg> <nt>node01</nt> </concept> <concept> <descriptor>TestTerm03</descriptor> <niveau>TH Thesaurusterm</niveau> <id>101#03</id> <flg>A</flg> <rt>node01</rt> </concept> </thesaurus> </kbt>
Database	2 Nodes in de database: 1: Name: “node01”, key: “node01”, type: “ThesaurusTerm” 1: Name: “node01”, key: “node02”, type: “ThesaurusTerm”
Verwacht resultaat	De volgende drie nodes zijn aangemaakt: Node1: name: “TestTerm01”, key: “101#01”, type: “ThesaurusTerm”, bt: id van (key)“node01” Node2: name: “TestTerm02”, key: “101#02”, type: “ThesaurusTerm”, nt: id van (key)“node01” Node3: name: “TestTerm03”, key: “101#03”, type: “ThesaurusTerm”, rt: id van (key)“node01” De nodes “node01” zij ongewijzigd omdat deze niet zijn meegenomen door de ThesaurusImporter doordat ze niet via de importeerfunctie zijn geïmporteerd maar direct in de database gezet.

06. TESTUITVOERING

Voor het test zijn Unit testen geschreven. Deze zijn ontwikkeld aan de hand van de Fysieke testgevallen. Wanneer mogelijk , is de XML in de testmethode opgesteld om zo geen echte XML bestanden aan te hoeven maken. Wanneer dit toch nodig was zoals bij ImportTag dan is een map aangemaakt waarin de XML bestanden aan zijn gemaakt. Aan het einde van de test worden de aangemaakte XML bestanden en de aangemaakte mappen weer verwijderd.

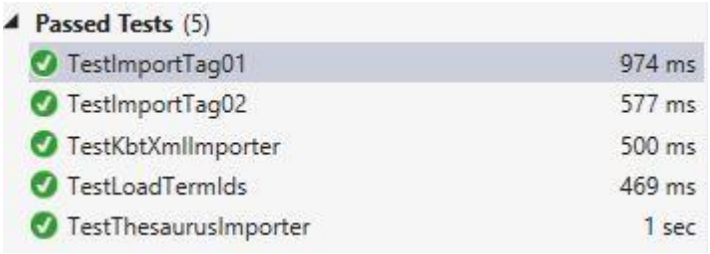
Voor de unit testen is een aparte test database aangemaakt en wordt een deel van het Lynkx framework gesimuleerd. De aparte database is aangemaakt om ervoor te zorgen dat de testen de data die gebruikt wordt niet kunnen manipuleren. En een deel van het framework moest gesimuleerd worden om het missen van de browser op te vangen.

07. CONCLUSIE

Tijdens het ontwikkel van de unit testen zijn een klein aantal (3-5) minor bugs ontdekt. Deze zijn direct verholpen waardoor alle testen slagen.

Alle paden zijn doorlopen en hierbij treden geen fouten op. Hieruit is geconcludeerd dat het importeren van een thesaurus correct functioneert.

APPENDIX A: TESTRESULTATEN



Detail testplan acceptatietest

Afstudeeropdracht ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *25-11-2013*

Versie: *1.0*

01. INHOUDSOPGAVE

2 Inleiding 1
3 Doel 1
4 Testbasis 1
5 Teststrategie 1
6 Testuitvoering 5
7 Conclusie 5
Appendix A: Testresultaten 6

02. INLEIDING

Dit document beschrijft de acceptatietest voor het project “Ontwikkeling document trefwoordsuggesties”. Hierbij zal worden toegelicht wat er wordt getest, hoe dit wordt getest, de resultaten van het testen en de conclusie die daaruit getrokken kan worden.

03. DOEL

Deze acceptatietest is ervoor om te testen of alle gewenste functionaliteiten uit het document Eisen 1_0 ook daadwerkelijk in de applicatie zijn. De functionaliteiten kunnen zijn omschreven in de vorm van functioneel & non-functionele eisen en user stories. Wanneer deze test is voltooid zal bekend zijn of er nog onderdelen in de applicatie missen.

04. TESTBASIS

Deze acceptatietest zullen de functionele eisen, non-functionele eisen en user stories getest worden die te vinden zijn in het document Eisen 1_0 van 20-09-2013.

05. TESTSTRATEGIE

Voor elke user story en non-functionele eis is een scenario uitgeschreven. Elk scenario zal eerst door de ontwikkelaar doorlopen worden om te zien of het naar behoren functioneert. Hierna zal ook nog elk scenario doorlopen worden door een collega die de applicatie nog niet heeft gezien, de opdrachtgever en een vertegenwoordiger van de gebruikers.

De collega is een mede stagiair die aan een ander project werkt en hierom de applicatie nog niet heeft gezien. Hij is geselecteerd als testpersoon omdat hij de applicatie nog niet heeft gezien zal zo de gebruiksvriendelijkheid getest kunnen worden. Dit zal uiteraard ook getest worden door de andere personen maar de andere personen hebben de applicatie al reeds gezien en zullen dus eenvoudiger begrijpen hoe deze werkt. Dit zal dan geen werkelijk resultaat betreffende de gebruiksvriendelijkheid.

Hieronder zijn de scenario’s omschreven. Hierbij is ook aangegeven voor welke eis of user story het scenario is opgesteld. En hieraan is ook een code toegewezen, bij de resultaten zal enkel naar de code verwezen worden om te voorkomen dat de scenario’s te vaak beschreven staan. Deze codes beginnen allemaal met SC.

Tijdens het ontwikkelen zijn er ook nog overige wensen naar voren gekomen. Deze zijn ook opgenomen in de test als scenario’s maar deze codes beginnen met SCX (SCenario eXtra). Als deze niet slagen dan zal dit minder zwaar meewegen in de conclusie.

Code	*SC01*
User story/eis	US01. Als gebruiker die de enquête start wil ik zien voor hoeveel documenten ik keywords kan beoordelen.
Scenario	1. De gebruiker klikt op de link waarmee hij/zij naar de enquête gaat. 2. De gebruiker ziet rechts bovenaan hoeveel documenten deze enquête omvat.

Code	*SC02*
User story/eis	US02. Als gebruiker wil ik gevonden keywords voor een document kunnen beoordelen.
Scenario	5. De gebruiker heeft de enquête geopend. 6. De gebruiker klikt voor elk gevonden keyword op een score/ster. 7. De gebruiker gaat naar het volgende document en keer vervolgens terug. 8. De gegeven scores zijn opgeslagen en zullen nog steeds hetzelfde zijn.

Code	*SC03*
User story/eis	US03. Als gebruiker wil ik een eigen suggestie(s) voor keywords kunnen geven.
Scenario	1. De gebruiker heeft de enquête geopend. 2. De gebruiker voert in het invoerveld onder de keywords zijn eigen suggestie in. 3. De gebruiker drukt op enter of klikt op “Keyword toevoegen”. 4. Het ingevoerde keyword verschijnt onder de al gegeven keywords onder het kopje “Toegevoegde suggesties”.

Code	*SC04*
User story/eis	US04. Als gebruiker wil ik naar een volgend document in de enquête kunnen gaan.
Scenario	1. De gebruiker heeft de enquête geopend. 2. De gebruiker klikt op “>” in de rechter bovenhoek. 3. De gebruiker ziet het volgende document. 4. De gebruiker klikt op “Next >” in de rechter onderhoek. 5. De gebruiker ziet het volgende document.

Code	*SC05*
User story/eis	US05. Als gebruiker ik wil gedurende de enquête kunnen zien hoever ik in de enquête nog moet.
Scenario	1. De gebruiker heeft de enquête geopend op het eerste document. 2. De progressiebalk is leeg. Rechts van de progressiebalk staat het totaal aantal documenten. 3. De gebruiker gaat naar het volgende document. 4. De gebruiker ziet dat hij bij het tweede document is en de progressiebalk is voor een deel ingekleurd. 5. De gebruiker gaat naar het laatste document. 6. De progressiebalk is geheel gevuld.

Code	*SC06*
User story/eis	US06. Als gebruiker wil ik de enquête vroegtijdig kunnen beëindigen.
Scenario	1. De gebruiker heeft de enquête geopend. 2. De gebruiker vult voor het eerste document de scores in voor de trefwoorden. 3. De gebruiker sluit de pagina en de browser. 4. De gebruiker gaat terug naar de enquête door middel van de link. 5. De gebruiker ziet het eerstvolgende nog niet ingevulde document. 6. De gebruiker gaat terug naar het eerste document. 7. De gegeven scores zijn nog steeds ingevuld.

Code	*SC07*
User story/eis	US07. Als beheerder wil ik eenvoudig een enquête kunnen opstellen door aan te geven welk algoritme(s) gebruikt moet worden, uit hoeveel documenten de enquête bestaat en hoeveel keywords er bij elk document worden gezocht.
Scenario	1. De gebruiker gaat naar de backoffice. 2. De gebruiker maakt een algoritme aan. 3. De gebruiker krijgt een formulier waarbij hij/zij de naam van het algoritme kan invullen en hoeveel trefwoorden het algoritme moet geven bij een document. 4. De gebruiker vult de gegevens in en slaat het formulier op en maakt een enquête(survey) aan. 5. De gebruiker krijgt een formulier waarbij hij/zij de naam, te gebruiken algoritme(s), gebruikers en documenten kan invullen. 6. De gebruiker vult de gegevens in en slaat het formulier op. 7. De gebruiker opent de enquête door middel van een gegenereerde link. 8. Controle of de enquête functioneert als verwacht.

Code	*SC08*
User story/eis	US08. Als beheerder wil ik voor elke enquête een overzicht kunnen opvragen wat weergeeft: hoeveel personen de enquête hebben ingevuld en hoe goed het algoritme(s) heeft gepresteerd.
Scenario	1. De gebruiker gaat naar de backoffice. 2. De gebruiker dubbel klikt op een enquête. 3. De gebruiker ziet hoeveel resultaten er zijn met een uniek UserGuid. 4. De gebruiker klikt rechts op een resultaat. 5. De gebruiker ziet per document de gegeven keywords, met welke algoritme elk keyword is gegeven en de score die is ingevuld door de gebruiker.

Code	*SC09*
User story/eis	N01. Wanneer een een gebruiker een document opent moet de lijst keywords hiervoor binnen vijf seconden geladen zijn.
Scenario	1. De gebruiker opent een enquête ingesteld met het algorimte KNN. 2. De pagina is geladen binnen 5 seconden. 3. De gebruiker gaat 25 keer naar de volgende pagina. 4. Elke pagina is geladen binnen 5 seconden. 5. De gebruiker opent een enquête ingesteld met het algorimte TextRank. 6. De pagina is geladen binnen 5 seconden. 7. De gebruiker gaat 25 keer naar de volgende pagina. 8. Elke pagina is geladen binnen 5 seconden.

Code	*SC10*
User story/eis	N02. Gebruiksvriendelijk voor de gebruikers. Zij moeten de tool zonder uitleg binnen 1 minuut begrijpen en kunnen gebruiken.
Scenario	8. Wanneer de gebruiker de enquête tool al eens heeft gebruikt zal het resultaat minder betrouwbaar zijn. Want hij zal dan al weten hoe het werkt. 9. De gebruiker opent de enquête via de link. 10. Na 1 minuut wordt de gebruiker het volgende gevraagd te doen/beantwoorden en kan elk uitvoeren binnen 5 seconden. 11. Beoordeel 3 keywords/trefwoorden/samenvattingen (hoeft niet accuraat). 12. Voeg 2 eigen suggesties toe. 13. Ga naar een volgend document. 14. Hoeveel document zitten er in deze enquête.

Code	*SCX01*
User story/eis	Wanneer een gebruiker de enquête heeft voltooid (alles documenten langsgegaan) en de enquête weer opent moet een nieuw SurveyResutaat worden aangemaakt.
Scenario	1. De gebruiker opent een enquête en vult deze in. (klikt door alle documenten heen) 2. De gebruiker sluit de enquête en opent deze opnieuw via de link. 3. De gebruiker krijgt een nieuwe nog niet ingevulde enquête. 4. Er zijn twee resultaten voor deze gebruiker aangemaakt voor de enquête.

Code	*SCX02*
User story/eis	Een gebruiker kan overig commentaar invullen per document.
Scenario	5. De gebruiker heeft een enquête geopend. 6. De gebruiker vult het commentaar in. (onder de keywords) 7. De gebruiker gaat naar het volgende document. 8. Het commentaar is opgeslagen. (Wanneer de gebruiker terug gaat naar het document zal het commentaar nog steeds ingevuld staan.

06. TESTUITVOERING

Zoals eerder omschreven is er getest door de ontwikkelaar, de opdrachtgever, een collega die de applicatie nog niet eerder heeft gezien en een vertegenwoordiger van de gebruikers.

Eerst heeft de ontwikkelaar zelf getest om zo de meeste fouten al op te sporen.

Bij de volgende testen heeft de ontwikkelaar ernaast gezeten om de testresultaten te noteren.

Hierna is het getest met de collega. Hierbij is eerst SC10 beantwoord omdat het resultaat voor dit scenario beter is wanneer de gebruiker nog geen kennis heeft van het systeem.

Als laatste is getest met de opdrachtgever en de vertegenwoordiger van de gebruikers.

De resultaten hiervan zijn te vinden in Appendix A: Testresultaten.

07. CONCLUSIE

Bijna alle user stories en eisen zijn verwerkt in de applicatie. Gezien de test met de collega moeten er nog wel wat kleine aanpassingen gedaan worden aan de user interface. De nodige wijzigingen zullen in een volgende sprint worden ingepland.

Scenario SC09 is niet in alle gevallen gehaald omdat het algoritme TextRank soms meer dan 5 seconden nodig heeft. Dit wordt echter geaccepteerd omdat de gebruikte implementatie van TextRank niet van productie kwaliteit is. Wanneer TextRank in een product gebruikt wordt zal deze herschreven moeten worden.

Scenario SC08 met de bijhorende eis komt te vervallen na overleg met de opdrachtgever. Hierbij is de keuze gemaakt om niet een generiek onderdeel te schrijven voor het analyseren van alle resultaten maar dit specifiek te doen per klant.

Met de correcties hierboven op SC08 & SC09 zijn alle testen geslaagd en zijn alle eisen verwerkt in de applicatie.

APPENDIX A: TESTRESULTATEN

De volgende resultaten zijn ingevuld door de ontwikkelaar van het systeem.

Scenario code	*SC01*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC02*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC03*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC04*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC05*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC06*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC07*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC08*
Uitvoerende	Ontwikkelaar
Resultaat	Dit onderdeel is niet ontwikkeld.
Opmerking/reden.	

Scenario code	*SC09*
Uitvoerende	Ontwikkelaar
Resultaat	Bij KNN gaat dit wel altijd goed. Echter is het algoritme TextRank een stuk langzamer waardoor het niet altijd wordt gehaald.
Opmerking/reden.	Bij TextRank is het soms dat het snel genoeg gaat en soms niet. Dit licht heel erg aan de lengte van de tekst.

Scenario code	*SC10*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	Als ontwikkelaar is dit niet te testen omdat de ontwikkelaar precies weet welke knoppen er zijn en wat ze doen omdat hij zelf het systeem gebouwd heeft.

Scenario code	*SCX01*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SCX02*
Uitvoerende	Ontwikkelaar
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

De volgende resultaten zijn ingevuld met de collega die de applicatie nog niet eerder heeft gezien. Hierbij is eerst SC10 doorlopen en daarna de rest van de scenario's.

Scenario code	*SC01*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC02*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC03*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC04*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC05*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC06*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC07*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC08*
Uitvoerende	Collega
Resultaat	De resultaten konden wel ingezien worden maar dit was niet overzichtelijk of duidelijk.
Opmerking/reden.	De resultaten konden ingezien worden in JSon.

Scenario code	*SC09*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC10*
Uitvoerende	Collega
Resultaat	Het doorlopen ging wat moeizaam.
Opmerking/reden.	De collega wist in eerste instantie niet geheel wat voor een applicatie het was. Hiervoor was dus wel wat uitleg nodig. Verder is de collega buitenlands waardoor hij moeite heeft met het lezen van Nederlands. Aangezien de applicatie in het Nederlands is zorgde dit ook voor extra moeilijkheid. Commentaar van collega: • De keywords lijken op knoppen waarbij hij verwachte sterren toe te voegen. • Het is niet direct duidelijk wat de sterren beteken. • Niet duidelijk of 0 sterren een slechte score is of geen score. • Bij het commentaar veld is een submit knop verwacht. • Een melding verwacht wanneer commentaar is opgeslagen (hiervoor is de submit knop nodig). • Delete knop bij toegevoegde keywords heeft niet dezelfde opmaak als de andere knoppen. • Toegevoegde keywords hebben geen sterren, hij verwachte 5 ingevulde of lege sterren. • Wanneer een keyword wordt toegevoegd door middel van de toets “enter” wordt het invoerveld leeg gemaakt. Dit gebeurt niet wanneer op de knop toevoegen wordt geklikt. • In de tekst van Kluwer is sommige tekst onderstreept, dit geeft de indruk dat het klikbaar is. • Bij het laatste document is een knop voltooien verwacht een eind pagina. • Voor de beheerder zou een handleiding nuttig zijn omdat het opzetten van een enquête relatief ingewikkeld is. • Het extra keyword veld voor samenvattingen moet groter getoond worden. • Bij de samenvattingen is duidelijker wat de sterren betekenen en de samenvattingen lijken niet op knoppen. • Ervoor zorgen dat of alles in het Engels is of alles in het Nederlands. • Bij het opzetten van een enquête is niet duidelijk wat bedoelt wordt met long en short bij “suggestion type”.

Scenario code	*SCX01*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SCX02*
Uitvoerende	Collega
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

De volgende resultaten zijn ingevuld met de vertegenwoordiger van de gebruikers. Dit is tevens de contactpersoon van Kluwer.

Scenario code	*SC01*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC02*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC03*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC04*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC05*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC06*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC07*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC08*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	De resultaten waren niet goed in te zien.
Opmerking/reden.	Er moet nog worden ontwikkeld zodat de resultaten ingezien kunnen worden en hiervan een conclusie getrokken kan worden.

Scenario code	*SC09*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC10*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	Aangezien de gebruikers vertegenwoordiger al wist hoe de applicatie werkt kon de gebruikersvriendelijkheid niet goed getest worden.

Scenario code	*SCX01*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SCX02*
Uitvoerende	Gebruikers vertegenwoordiger
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

DTP acceptatietest
Ontwikkeling document trefwoordsuggesties

De volgende resultaten zijn ingevuld met de opdrachtgever.

Scenario code	*SC01*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC02*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	Ook laten zien waar de resultaten opgeslagen worden.

Scenario code	*SC03*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	Ook laten zien waar de suggesties opgeslagen worden.

Scenario code	*SC04*
Uitvoerende	Opdrachtgever
Resultaat	Bij het teruggaan naar een al ingevuld document door middel van de terug knop van de browser worden de resultaten niet getoond.
Opmerking/reden.	

Scenario code	*SC05*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC06*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SC07*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	Ook laten zien waar een optie opgeslagen wordt. En hoe de URL's gegeneerd en afgehandeld worden.

Scenario code	*SC08*
Uitvoerende	Opdrachtgever
Resultaat	Niet ontwikkeld.
Opmerking/reden.	Na overleg met de opdrachtgever is besloten om analyses per klant op te stellen omdat deze te specifiek zijn en hiervoor niet een generiek onderdeel ontwikkeld kan worden.

Scenario code	*SC09*
Uitvoerende	Opdrachtgever
Resultaat	Alleen TextRank haalt de tijd soms niet.
Opmerking/reden.	De gebruikte versie van TextRank is niet van productie kwaliteit en het is geaccepteerd dat deze niet altijd binnen 5 seconden klaar is. Overigens haalt deze het meestal wel binnen de tijd en het duurt niet veel langer wanneer deze het niet haalt.

Scenario code	*SC10*
Uitvoerende	Opdrachtgever
Resultaat	De GUI was al bekend maar de opdrachtgever is akkoord gegaan.
Opmerking/reden.	Verteld over de eerdere test met de college en welke zwakke punten hieruit naar voren kwamen.

Scenario code	*SCX01*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Scenario code	*SCX02*
Uitvoerende	Opdrachtgever
Resultaat	Zonder problemen doorlopen.
Opmerking/reden.	

Analyse

Afstudeeropdracht ontwikkeling document trefwoordsuggesties

Door: *Jos Verburg*

Datum: *27-12-2013*

Versie: *1.0*

01. INHOUDSOPGAVE

2 Inleiding 1
3 Gebruikte middelen 1
4 Documenten set analyse 2
5 Documenten KNN analyse 9
5.1 Gehele set 10
5.2 AangehaaktCommentaar 11
5.3 JurisprudentieBewerkt 12
5.4 Literatuur 13
5.5 VakstudieAC 14
6 Redacteuren analyse 15
6.1 Belasting recht 15
6.2 Burgerlijk recht 16
6.3 Staats- en bestuursrecht 16
7 Conclusie 17
7.1 Conclusie documenten set 17
7.2 Conclusie documenten KNN 18
7.3 Conclusie redacteuren 19
7.4 Conclusie totaal 20
8 Advies 20

02. INLEIDING

Dit analyse rapport zal zich richten op het algoritme KNN dat is getest voor Kluwer. Hiervoor is toegelicht welke middelen zijn gebruikt voor het analyseren van KNN. En daarna volgen de verschillende analyses die zijn uitgevoerd.

Voor elke analyse is een conclusie opgesteld in hoofdstuk 7 en ook is daar een totaal conclusie opgesteld betreffende de inzetbaarheid van KNN voor het suggereren van trefwoorden voor de documenten van Kluwer.

03. GEBRUIKTE MIDDELEN

Voor het analyseren zijn documenten van Kluwer gebruikt. Deze zijn aangeleverd door Kluwer via FTP en zijn binnengehaald op 04-11-2013 en 08-11-2013. De documenten zijn aangeleverd elf mappen met elk daarin een aantal .zip bestanden. In de tabel is te zien welke mappen zijn geïmporteerd en onder welke node deze zijn ondergebracht. Hiernaast is ook de KBT gebruikt die op dezelfde manier is aangeleverd. Dit was slechts één bestand en is niet opgenomen in de tabel. De KBT is ook al voor het analyseren geïmporteerd. Hoeveel documenten uit elke map zijn geïmporteerd en waarom worden behandeld in hoofdstuk 4.

MAP NAAM	NODE
AB	JurisprudentieBewerkt
ASSER	Literatuur
NJ	JurisprudentieBewerkt
T&C	AangehaaktCommentaar
VST DEEL 1	VakstudieAC
VST DEEL 2	VakstudieAC
VST DEEL 3	VakstudieAC
VST DEEL 4	VakstudieAC
VST DEEL 5	VakstudieAC
VST DEEL 6	VakstudieAC
VST DEEL 7	VakstudieAC

Tabel 1 Importeer mapping

04. DOCUMENTEN SET ANALYSE

Eerst zijn de aangeleverde documenten geanalyseerd hierbij is gekeken naar hoeveel documenten er van de set geïmporteerd moeten worden. Om een document te importeren moet het voldoen aan de volgende eisen:

- Het moet een naam hebben “md:vindplaats”
- Het moet minimaal één gevonden trefwoord hebben “md:trefwoord”. Dit trefwoord zal gezocht worden in de al geïmporteerde KBT wanneer het trefwoord hier niet in wordt gevonden dan geldt het niet als een gevonden trefwoord.

De analyse is apart per aangeleverde map uitgevoerd. Hierbij zijn alle documenten nagelopen en worden verschillende onderdelen geteld. Voor elke map is in dit hoofdstuk een tabel opgesteld met daarin de belangrijke resultaten van de analyse. Ook is geanalyseerd welke trefwoorden wel aanwezig waren maar niet gevonden werden, deze resultaten zijn bewaard en kunnen worden opgevraagd.

De bestanden zonder naam worden alleen geteld wanneer deze wel gevonden trefwoorden hebben.

Map	Totaal geïmporteerd	Totaal niet geïmporteerd	Totaal
AB	1812	45	1857
Asser	6377	6561	12938
NJ	3355	309	3664
T&C	9751	14690	24441
vst deel 1	17189	12411	29600
vst deel 2	18021	14697	32718
vst deel 3	27816	10495	38311
vst deel 4	35523	4634	40157
vst deel 5	18349	5797	24146
vst deel 6	29277	14110	43387
vst deel 7	73270	64178	137448
TOTAAL	240740	147927	388667

Tabel 2 Map importeer aantallen

In Tabel 2 is het totaal weergegeven hoeveel documenten uit elke map zijn geïmporteerd. En in Tabel 3 is weergeven hoeveel documenten zijn geïmporteerd per node.

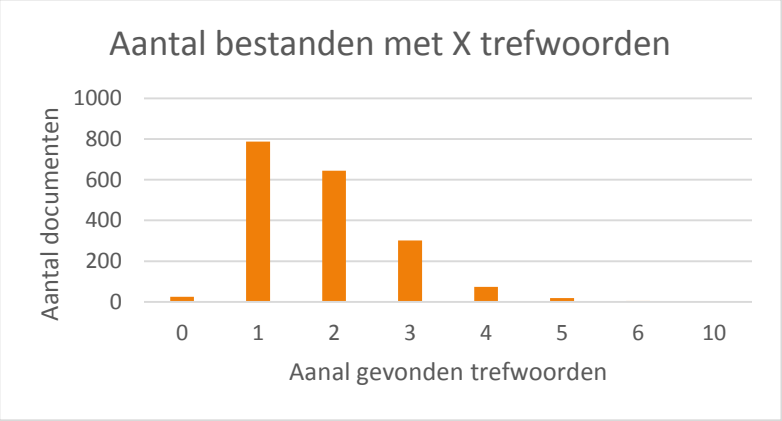
De grafieken in de tabellen geven het volgende weer: Voor elk aantal gevonden trefwoorden, bij hoeveel documenten dit voorkwam. Voorbeeld van de set AB: “Dus er zijn 787 documenten met 1 trefwoord en 645 documenten met 2 trefwoorden, etc.”

Node	Totaal geïmporteerd
JurisprudentieBewerkt	5167
Literatuur	6377
AangehaaktCommentaar	9751
VakstudieAC	219445

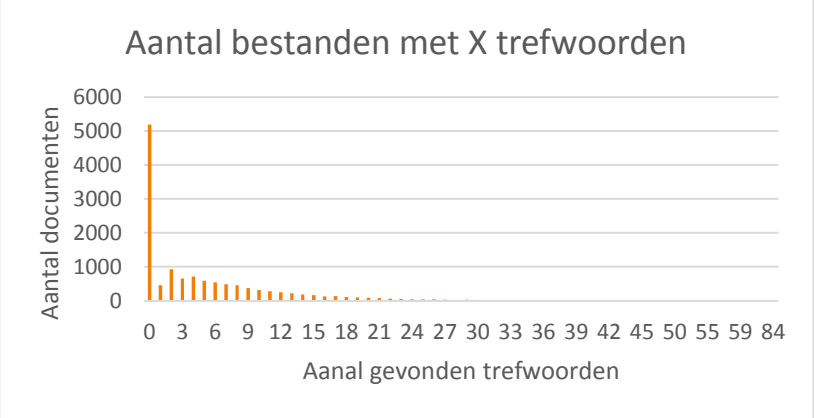
Tabel 3 Node importeer aantallen

Gezien het aantal niet gevonden trefwoorden zijn een aantal van de niet gevonden trefwoorden nagekeken. In deze gevallen verwees het document naar een trefwoord met een flg (vlag) B. Dit kenmerkt een trefwoord waar een document niet naar mag verwijzen en deze zijn dus ook niet geïmporteerd.

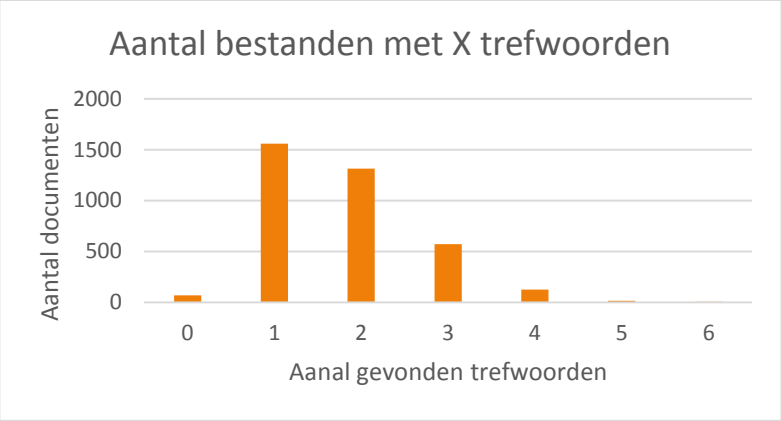
Map: AB	*Node: JurisprudentieBewerkt*
Totaal aantal bestanden	1857
Aantal bestanden met trefwoorden	1832
Aantal bestanden zonder trefwoorden	25
Aantal bestanden zonder gevonden trefwoorden	20
Aantal bestanden zonder naam	0
Aantal bestanden dat geïmporteerd moet zijn	1812
Aantal bestanden dat daadwerkelijk geïmporteerd is.	1812
Aantal verschillende trefwoorden	885
Aantal trefwoorden gevonden	798
Aantal niet gevonden trefwoorden	87



Map: Asser	*Node: Literatuur*
Totaal aantal bestanden	12938
Aantal bestanden met trefwoorden	7748
Aantal bestanden zonder trefwoorden	5190
Aantal bestanden zonder gevonden trefwoorden	1371
Aantal bestanden zonder naam	0
Aantal bestanden dat geïmporteerd moet zijn	6377
Aantal bestanden dat daadwerkelijk geïmporteerd is.	6377
Aantal verschillende trefwoorden	16451
Aantal trefwoorden gevonden	1735
Aantal niet gevonden trefwoorden	14716



Map: NJ	*Node: JurisprudentieBewerkt*
Totaal aantal bestanden	3664
Aantal bestanden met trefwoorden	3596
Aantal bestanden zonder trefwoorden	68
Aantal bestanden zonder gevonden trefwoorden	241
Aantal bestanden zonder naam	0
Aantal bestanden dat geïmporteerd moet zijn	3355
Aantal bestanden dat daadwerkelijk geïmporteerd is.	3355
Aantal verschillende trefwoorden	1623
Aantal trefwoorden gevonden	1411
Aantal niet gevonden trefwoorden	212

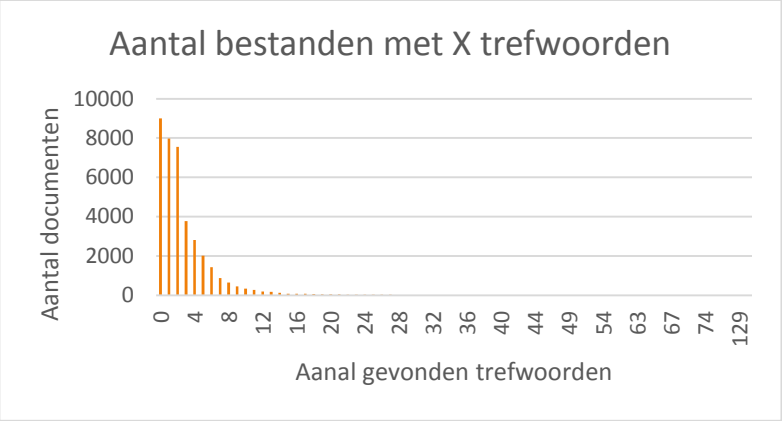


Map: T&C	*Node: AangehaaktCommentaar*
Totaal aantal bestanden	24441
Aantal bestanden met trefwoorden	11889
Aantal bestanden zonder trefwoorden	12552
Aantal bestanden zonder gevonden trefwoorden	2138
Aantal bestanden zonder naam	0
Aantal bestanden dat geïmporteerd moet zijn	9751
Aantal bestanden dat daadwerkelijk geïmporteerd is.	9751
Aantal verschillende trefwoorden	20203
Aantal trefwoorden gevonden	1792
Aantal niet gevonden trefwoorden	18411

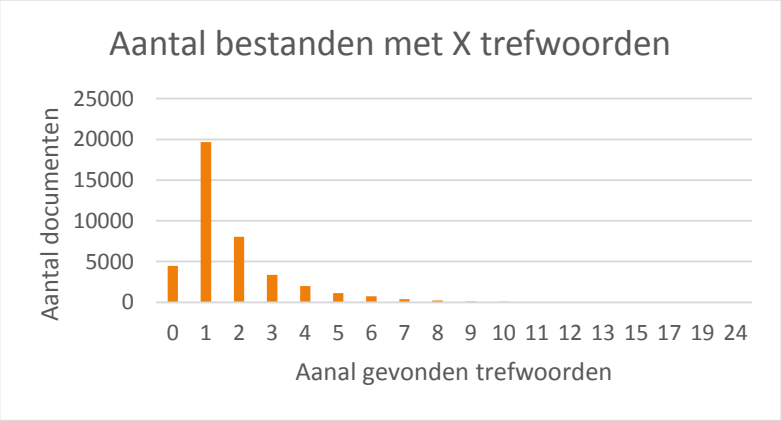
Map: vst deel 1	*Node: VakstudieAC*
Totaal aantal bestanden	29600
Aantal bestanden met trefwoorden	17611
Aantal bestanden zonder trefwoorden	11989
Aantal bestanden zonder gevonden trefwoorden	224
Aantal bestanden zonder naam	198
Aantal bestanden dat geïmporteerd moet zijn	17189
Aantal bestanden dat daadwerkelijk geïmporteerd is.	17189
Aantal verschillende trefwoorden	6865
Aantal trefwoorden gevonden	752
Aantal niet gevonden trefwoorden	6113

Map: vst deel 2	*Node: VakstudieAC*
Totaal aantal bestanden	32718
Aantal bestanden met trefwoorden	18671
Aantal bestanden zonder trefwoorden	14047
Aantal bestanden zonder gevonden trefwoorden	516
Aantal bestanden zonder naam	134
Aantal bestanden dat geïmporteerd moet zijn	18021
Aantal bestanden dat daadwerkelijk geïmporteerd is.	18021
Aantal verschillende trefwoorden	5337
Aantal trefwoorden gevonden	768
Aantal niet gevonden trefwoorden	4569

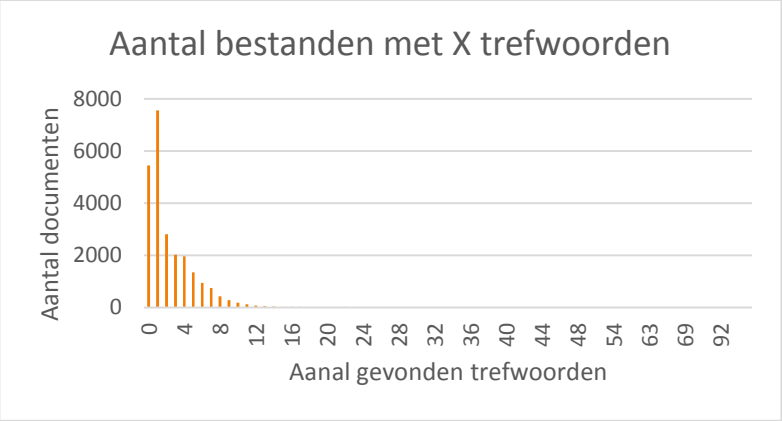
Map: vst deel 3	*Node: VakstudieAC*
Totaal aantal bestanden	38311
Aantal bestanden met trefwoorden	29318
Aantal bestanden zonder trefwoorden	8993
Aantal bestanden zonder gevonden trefwoorden	1486
Aantal bestanden zonder naam	16
Aantal bestanden dat geïmporteerd moet zijn	27816
Aantal bestanden dat daadwerkelijk geïmporteerd is.	27816
Aantal verschillende trefwoorden	28904
Aantal trefwoorden gevonden	1575
Aantal niet gevonden trefwoorden	27329



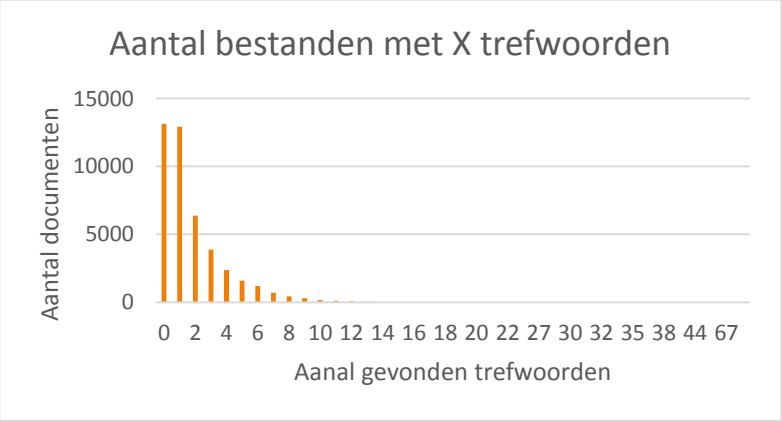
Map: vst deel 4	*Node: VakstudieAC*
Totaal aantal bestanden	40157
Aantal bestanden met trefwoorden	35680
Aantal bestanden zonder trefwoorden	4477
Aantal bestanden zonder gevonden trefwoorden	157
Aantal bestanden zonder naam	0
Aantal bestanden dat geïmporteerd moet zijn	35523
Aantal bestanden dat daadwerkelijk geïmporteerd is.	35523
Aantal verschillende trefwoorden	6543
Aantal trefwoorden gevonden	591
Aantal niet gevonden trefwoorden	5952



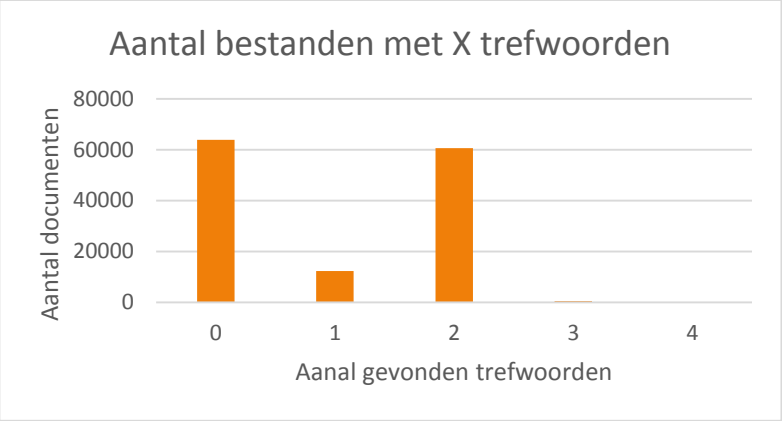
Map: vst deel 5	*Node: VakstudieAC*
Totaal aantal bestanden	24146
Aantal bestanden met trefwoorden	18708
Aantal bestanden zonder trefwoorden	5438
Aantal bestanden zonder gevonden trefwoorden	358
Aantal bestanden zonder naam	1
Aantal bestanden dat geïmporteerd moet zijn	18349
Aantal bestanden dat daadwerkelijk geïmporteerd is.	18349
Aantal verschillende trefwoorden	14443
Aantal trefwoorden gevonden	982
Aantal niet gevonden trefwoorden	13461



Map: vst deel 6	*Node: VakstudieAC*
Totaal aantal bestanden	43387
Aantal bestanden met trefwoorden	30234
Aantal bestanden zonder trefwoorden	13153
Aantal bestanden zonder gevonden trefwoorden	919
Aantal bestanden zonder naam	38
Aantal bestanden dat geïmporteerd moet zijn	29277
Aantal bestanden dat daadwerkelijk geïmporteerd is.	29277
Aantal verschillende trefwoorden	11193
Aantal trefwoorden gevonden	1285
Aantal niet gevonden trefwoorden	9908



Map: vst deel 7	*Node: VakstudieAC*
Totaal aantal bestanden	137448
Aantal bestanden met trefwoorden	73569
Aantal bestanden zonder trefwoorden	63879
Aantal bestanden zonder gevonden trefwoorden	0
Aantal bestanden zonder naam	299
Aantal bestanden dat geïmporteerd moet zijn	73270
Aantal bestanden dat daadwerkelijk geïmporteerd is.	73270
Aantal verschillende trefwoorden	209
Aantal trefwoorden gevonden	205
Aantal niet gevonden trefwoorden	4



05. DOCUMENTEN KNN ANALYSE

Bij dit onderdeel is KNN gebruikt op de complete documenten set. Hierbij is gekeken of de al toegekende trefwoorden achterhaald konden worden door KNN.

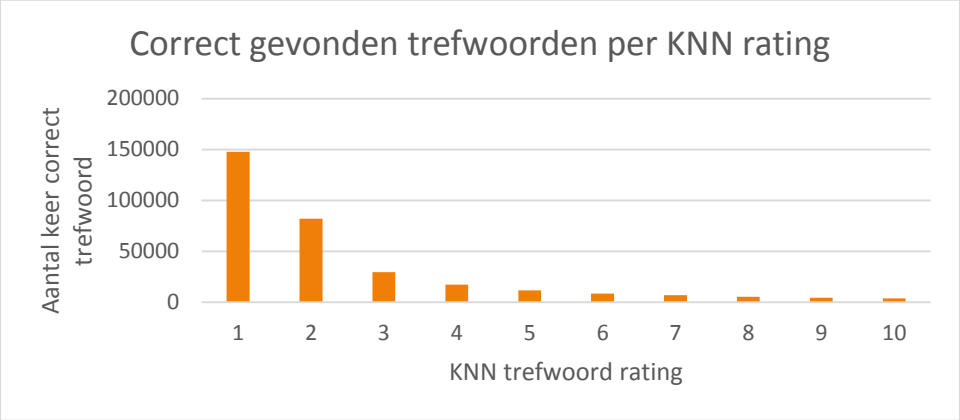
Bij KNN is het mogelijk om te definiëren onder welke Node gezocht mag worden voor lijkende documenten. Zoals in een eerder hoofdstuk staat zijn de documenten onderverdeeld onder vier Nodes. Er is één analyse voor de gehele documenten set waarbij KNN ook over alle documenten zoekt. Verder is er per Node één analyse waarbij KNN over alle documenten heeft gezocht en alleen over de documenten binnen die Node. De waarden waarbij over de alle documenten is gezocht zijn aangegeven door “Kluwer” wat een hoger liggende Node aangeeft. De waarden waarbij alleen over documenten is gezocht binnen de Node van de documenten zijn aangegeven met “Specifiek”.

In elke paragraaf staan delfde soort grafieken en tabel. Hieronder staat uitgelegd wat elke grafiek of de tabel betekend. Verder betekend in dit hoofdstuk een correct trefwoord: “Een trefwoord dat door KNN is gesuggereerd en al aan het document is toegewezen.”.

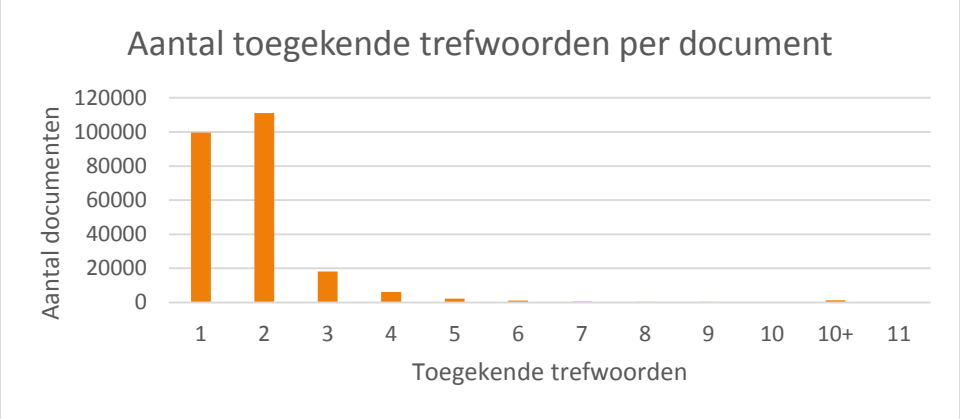
Tabel, grafiek	Uitleg
Correct gevonden trefwoorden per KNN rating	KNN geeft trefwoorden terug. Deze trefwoorden staan in een lijst van 1 t/m 10. Hierbij vindt KNN de eerste de beste en laatste het slechtste van de geselecteerde. In de grafiek staan de plaatsen 1 t/m 10 op de X-as en het aantal keer dat een trefwoord op deze positie correct was op de Y-as.
De tabel	Doordat KNN de 25 meest lijkende documenten zoekt en daarvan weer de trefwoorden kan het voorkomen dat alle 25 documenten veel overeenkomende trefwoorden hebben waardoor er minder dan 10 verschillende trefwoorden gevonden worden. In de tabel staat het aantal gevonden trefwoorden weergegeven bij hoe vaak dit aantal is gevonden.
Aantal toegekende trefwoorden per document	Deze grafiek geeft dezelfde soort informatie als in het vorige hoofdstuk maar dit gaat enkel over de documenten binnen de documenten set van de paragraaf. Op de X-as dus hoeveel trefwoorden het document heeft en op de Y-as bij hoeveel documenten dit het geval is.
De lijngrafiek	Hierin zijn de waarden precision, recall en f-score opgenomen. Hierbij staat de waarde op de Y-as en op de X-as bij hoeveel trefwoorden dit geld. Bij op de X-as 10 staan dus de waarden als 10 trefwoorden worden gesuggereerd en bij 1 op de X-as staan de waarden als maar 1 trefwoord wordt gesuggereerd. **Precision** De precision geeft weer hoeveel van de gesuggereerde trefwoorden ook toegekend zijn. Bij 1 zijn alle suggesties toegekend en bij 0 geen één. **Recall** De recall geeft weer hoeveel van de al toegekende trefwoorden voorkomen in de trefwoorden suggesties. Bij 1 zijn alle toegekende trefwoorden gesuggereerd bij 0 is geen van de toegekende trefwoorden gesuggereerd. **F-score** De f-score is een combinatie van de precision en recall om te voorkomen dat continue twee getallen vergeleken moeten worden. De formule voor de f-score is 2 x ((precision x recall) / (precision + recall))

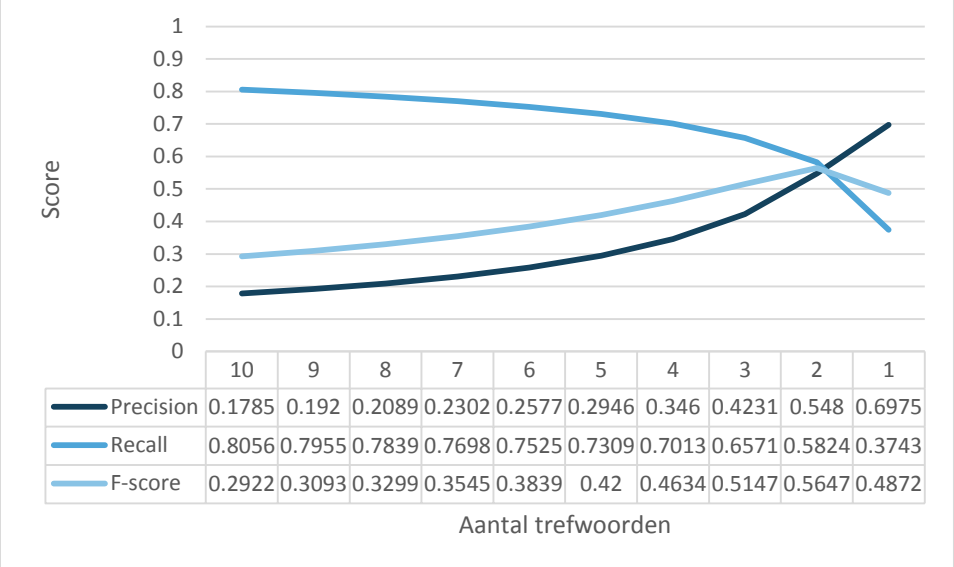
05.1 Gehele set

De gehele documenten set is gebruikt en KNN heeft over alle documenten gezocht voor lijkende documenten.



Aantal trefwoorden	Aantal documenten
0	28842
1	4160
2	16061
3	6801
4	7559
5	5939
6	5966
7	6145
8	6494
9	8709
10	146064

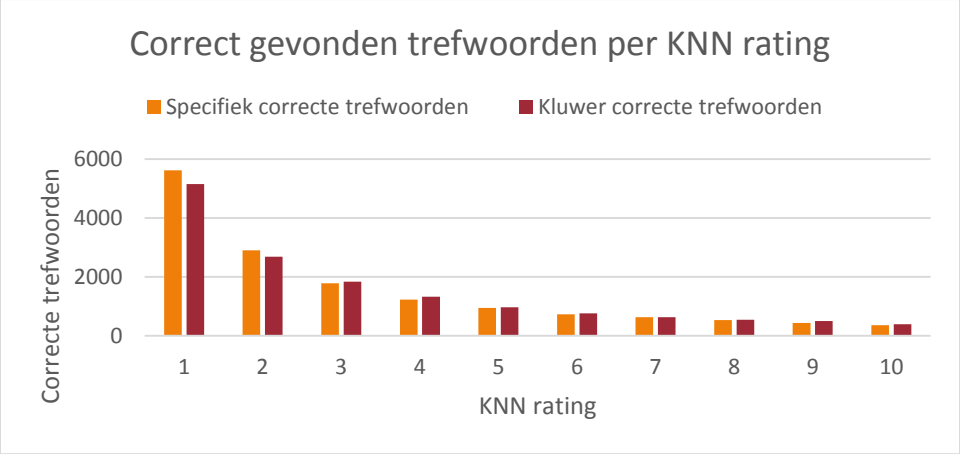


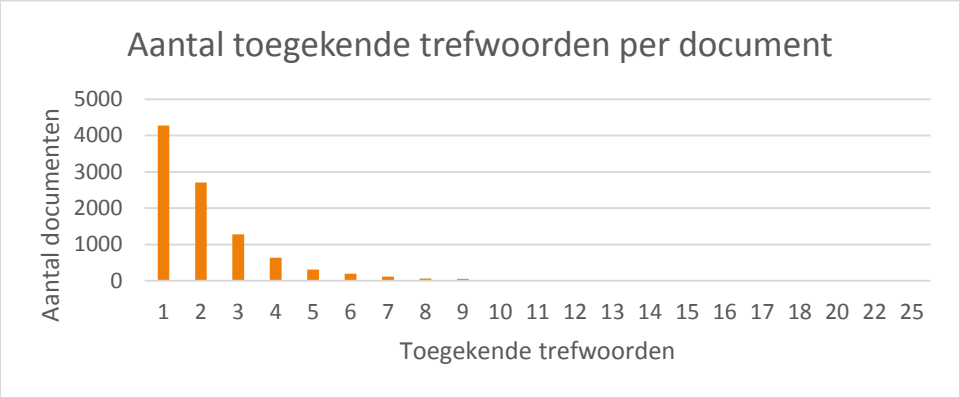


	10	9	8	7	6	5	4	3	2	1
Precision	0.1785	0.192	0.2089	0.2302	0.2577	0.2946	0.346	0.4231	0.548	0.6975
Recall	0.8056	0.7955	0.7839	0.7698	0.7525	0.7309	0.7013	0.6571	0.5824	0.3743
F-score	0.2922	0.3093	0.3299	0.3545	0.3839	0.42	0.4634	0.5147	0.5647	0.4872

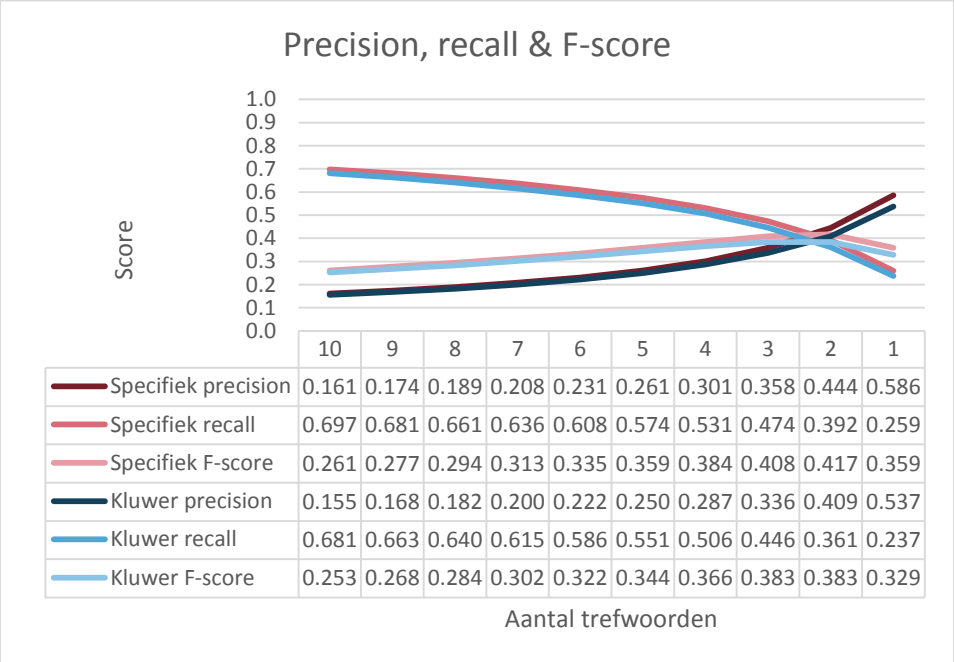
05.2 AANGEHAAKTCOMMENTAAR

Alleen de documenten onder AangehaaktCommentaar zijn gebruikt.





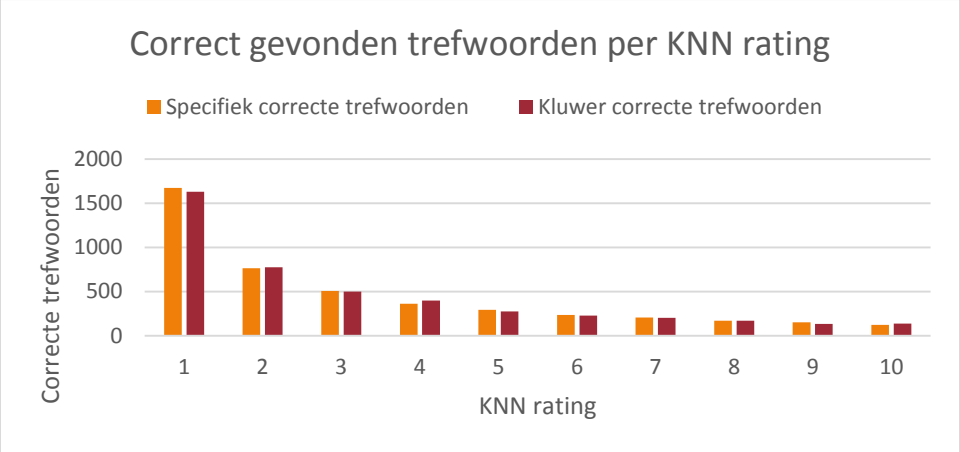
Aantal tefwoorden	Specifiek aantal	Kluwer aantal documenten
0	169	169
1	5	4
2	21	15
3	44	20
4	46	17
5	50	11
6	43	21
7	71	20
8	74	21
9	72	35
10	9156	9418



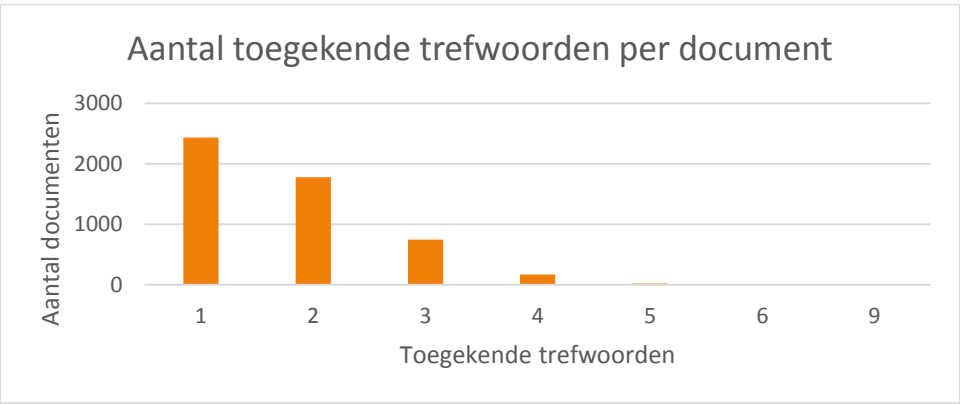
	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.161	0.174	0.189	0.208	0.231	0.261	0.301	0.358	0.444	0.586
Specifiek recall	0.697	0.681	0.661	0.636	0.608	0.574	0.531	0.474	0.392	0.259
Specifiek F-score	0.261	0.277	0.294	0.313	0.335	0.359	0.384	0.408	0.417	0.359
Kluwer precision	0.155	0.168	0.182	0.200	0.222	0.250	0.287	0.336	0.409	0.537
Kluwer recall	0.681	0.663	0.640	0.615	0.586	0.551	0.506	0.446	0.361	0.237
Kluwer F-score	0.253	0.268	0.284	0.302	0.322	0.344	0.366	0.383	0.383	0.329

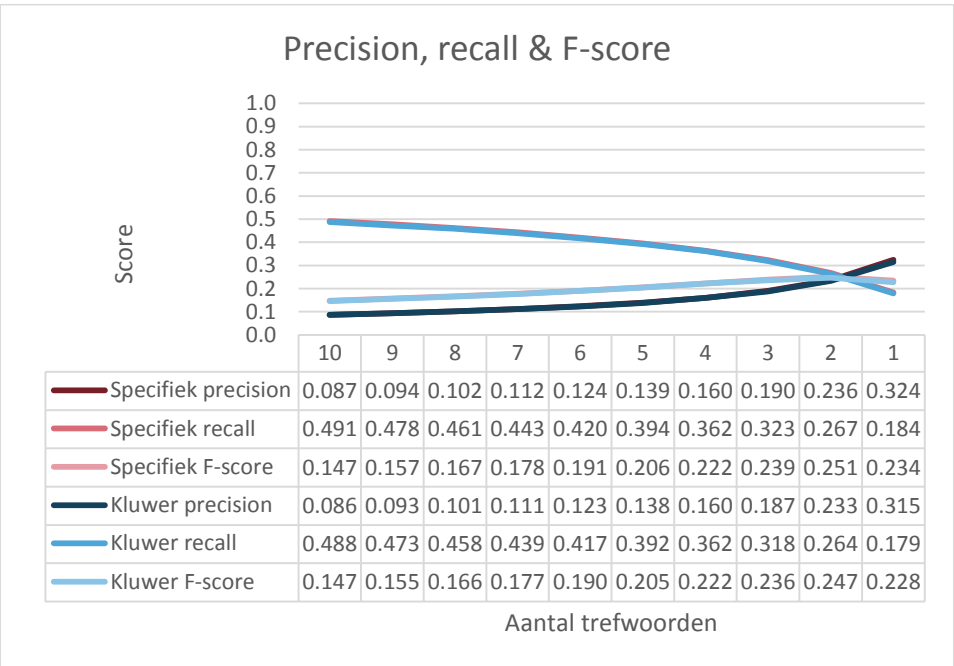
05.3 JurisprudentieBewerkt

Alleen de documenten onder JurisprudentieBewerkt zijn gebruikt.



Aantal tefwoorden	Specifiek aantal documenten	Kluwer aantal documenten
2	0	4
4	0	3
6	0	3
7	1	3
8	1	6
9	6	8
10	5159	5140

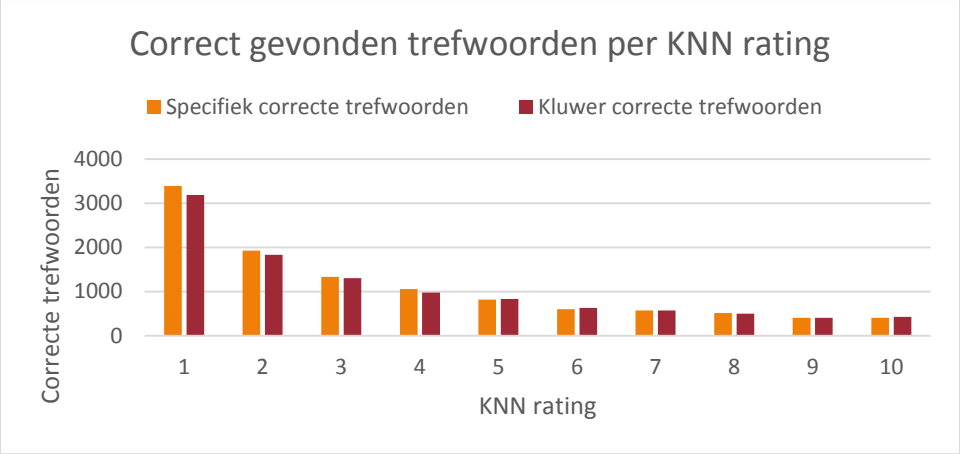


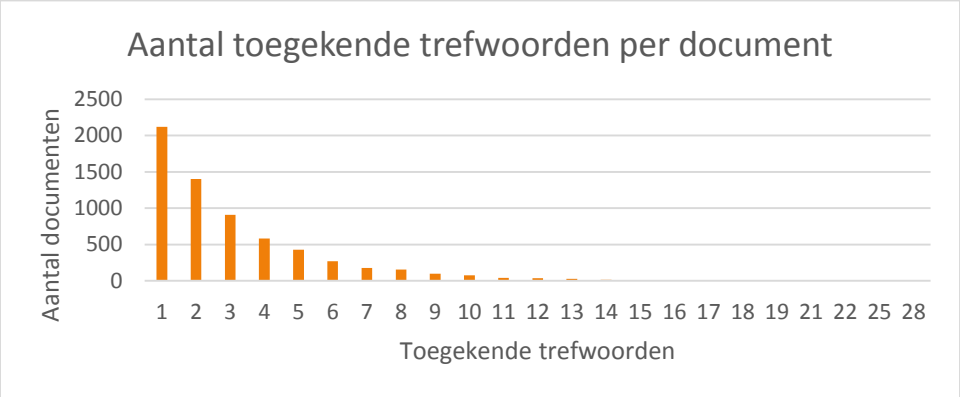


	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.087	0.094	0.102	0.112	0.124	0.139	0.160	0.190	0.236	0.324
Specifiek recall	0.491	0.478	0.461	0.443	0.420	0.394	0.362	0.323	0.267	0.184
Specifiek F-score	0.147	0.157	0.167	0.178	0.191	0.206	0.222	0.239	0.251	0.234
Kluwer precision	0.086	0.093	0.101	0.111	0.123	0.138	0.160	0.187	0.233	0.315
Kluwer recall	0.488	0.473	0.458	0.439	0.417	0.392	0.362	0.318	0.264	0.179
Kluwer F-score	0.147	0.155	0.166	0.177	0.190	0.205	0.222	0.236	0.247	0.228

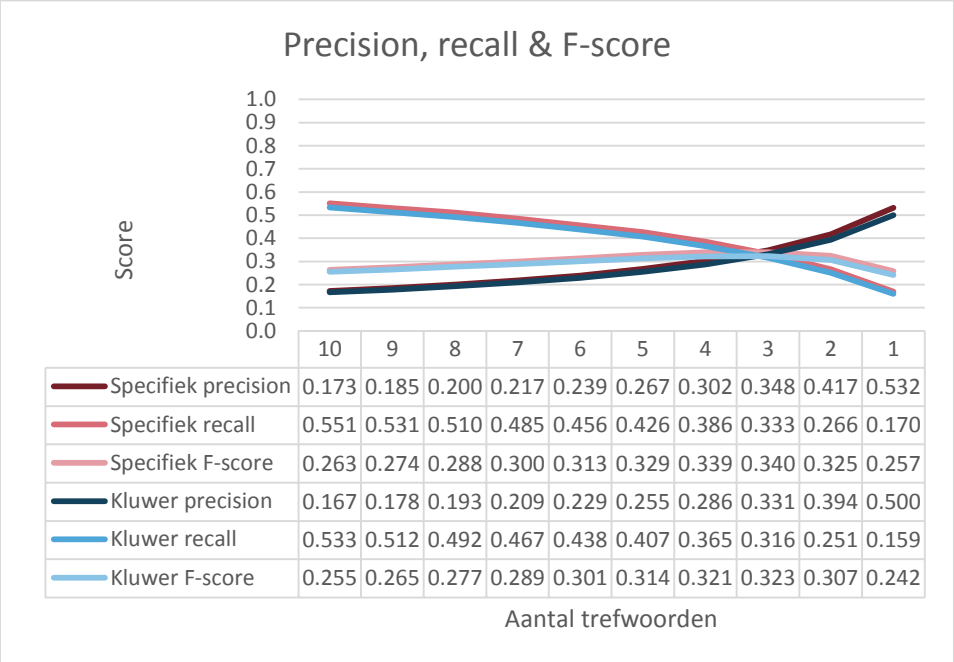
05.4 LITERATUUR

Alleen de documenten onder Literatuur zijn gebruikt.





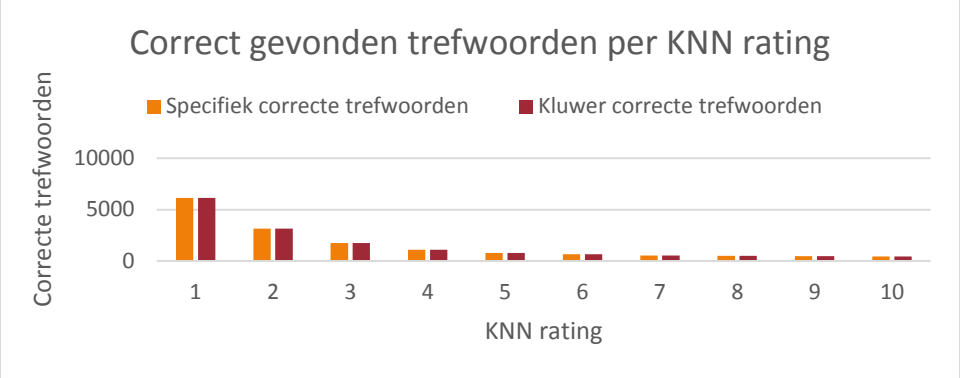
Aantal tefwoorden	Specifiek aantal documenten	Kluwer aantal documenten
0	3	3
2	0	1
3	0	1
6	2	0
7	2	2
8	1	1
9	5	2
10	6364	6367

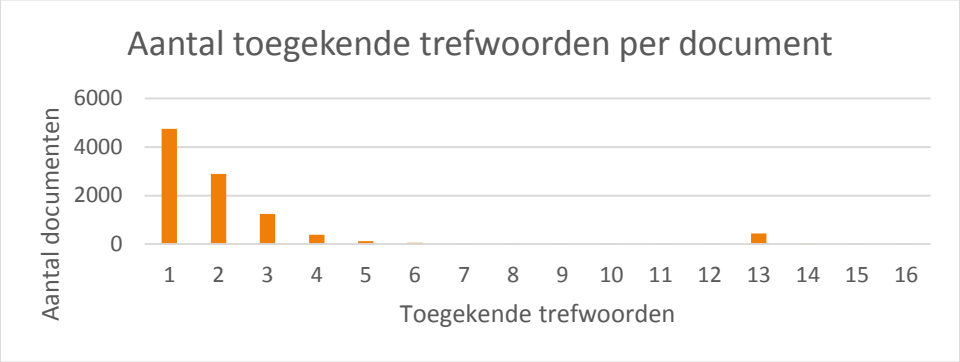


	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.173	0.185	0.200	0.217	0.239	0.267	0.302	0.348	0.417	0.532
Specifiek recall	0.551	0.531	0.510	0.485	0.456	0.426	0.386	0.333	0.266	0.170
Specifiek F-score	0.263	0.274	0.288	0.300	0.313	0.329	0.339	0.340	0.325	0.257
Kluwer precision	0.167	0.178	0.193	0.209	0.229	0.255	0.286	0.331	0.394	0.500
Kluwer recall	0.533	0.512	0.492	0.467	0.438	0.407	0.365	0.316	0.251	0.159
Kluwer F-score	0.255	0.265	0.277	0.289	0.301	0.314	0.321	0.323	0.307	0.242

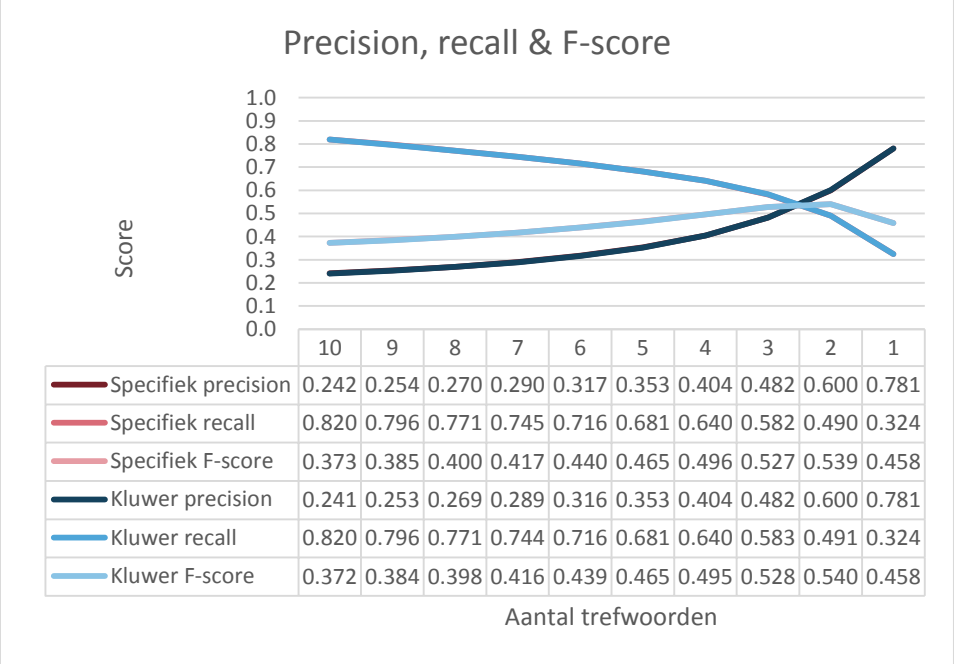
05.5 VAKSTUDIEAC

Alleen de documenten onder VakstudieAC zijn gebruikt. Door de ontdekte fout die is uitgelegd in hoofdstuk 6 moest dit onderdeel opnieuw worden uitgevoerd. Wegens tijdgebrek zijn alleen de eerste 10000 documenten gebruikt. Later zal dit nogmaals worden uitgevoerd voor alle documenten.





Aantal tefwoorden	Specifiek aantal documenten	Kluwer aantal documenten
0	2145	2145
1	236	235
2	219	210
3	283	276
4	574	570
5	331	314
6	318	318
7	336	342
8	335	330
9	402	399
10	4821	4861



	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.242	0.254	0.270	0.290	0.317	0.353	0.404	0.482	0.600	0.781
Specifiek recall	0.820	0.796	0.771	0.745	0.716	0.681	0.640	0.582	0.490	0.324
Specifiek F-score	0.373	0.385	0.400	0.417	0.440	0.465	0.496	0.527	0.539	0.458
Kluwer precision	0.241	0.253	0.269	0.289	0.316	0.353	0.404	0.482	0.600	0.781
Kluwer recall	0.820	0.796	0.771	0.744	0.716	0.681	0.640	0.583	0.491	0.324
Kluwer F-score	0.372	0.384	0.398	0.416	0.439	0.465	0.495	0.528	0.540	0.458

06. REDACTEUREN ANALYSE

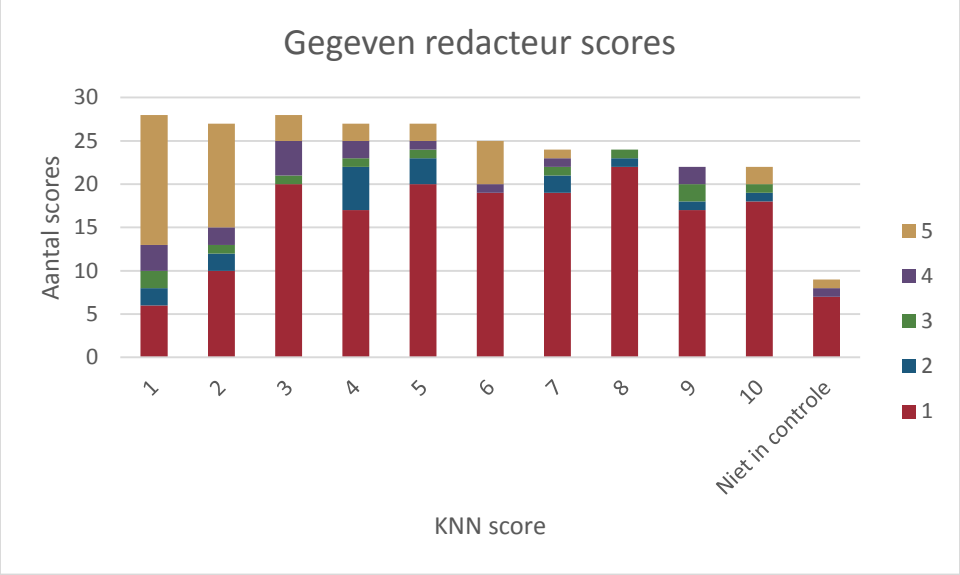
Drie redacteuren hebben elk voor 35 documenten de door KNN gesuggereerde trefwoorden beoordeeld. Hiervoor hebben zij elk trefwoord een score gegeven tussen de 1 en de 5. Tijdens het analyseren van de resultaten is nog een fout ontdekt. Het bleek dat bij de selectie van 25 lijkende documenten ook het document zelf zich bevond. Hierdoor hadden trefwoorden die al aan het document zijn toegewezen een grotere kans om gesuggereerd te worden. Om deze fout te herstellen is bij het analyseren nogmaals KNN gebruikt bij de documenten om de trefwoorden te suggereren maar ditmaal zonder de fout. De trefwoorden die wel bij de redacteur zijn voorgesteld maar niet bij de analyse zijn aangegeven met “niet in controle“.

De documenten die door de redacteuren zijn beoordeeld komen uit een bepaalde rubriek van de KBT (Kluwer Brede Thesaurus). De paragraaf titel zal de rubriek inhouden.

De grafiek geeft weer hoe vaak de redacteur scores heeft gegeven aan trefwoorden en op welke positie KNN de trefwoorden zet. Daaronder staat aangegeven hoeveel suggesties de redacteur heeft toegevoegd. Deze suggesties zijn trefwoorden die de redacteur wel verwacht voor het document maar deze niet zag. In de tabel staan de al toegewezen trefwoorden. Hierbij is aangegeven welke score de redacteur aan deze trefwoorden heeft gegeven. Wanneer een al toegevoegd trefwoord niet is gevonden dan staat deze bij “niet gevonden”.

De resultaten zijn in te zien met volgende link: http://rae.lynkx-staging.nl/analysis/KluwerAnalysis.lynkx

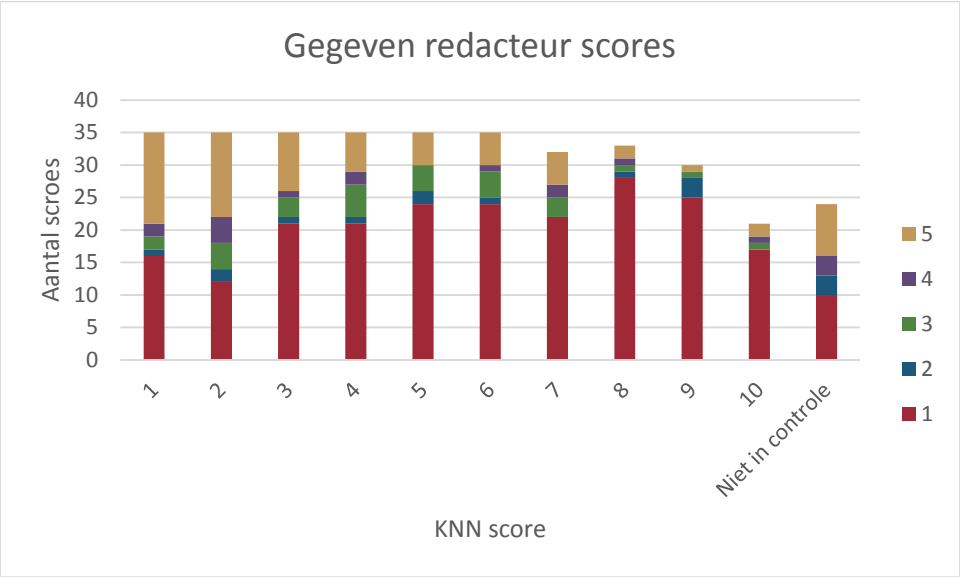
06.1 BELASTING RECHT



Score	Survey 0
1	7
2	3
3	5
4	9
5	32
Alleen in controle	0
Niet gevonden	24

Toegevoegde suggesties: 10

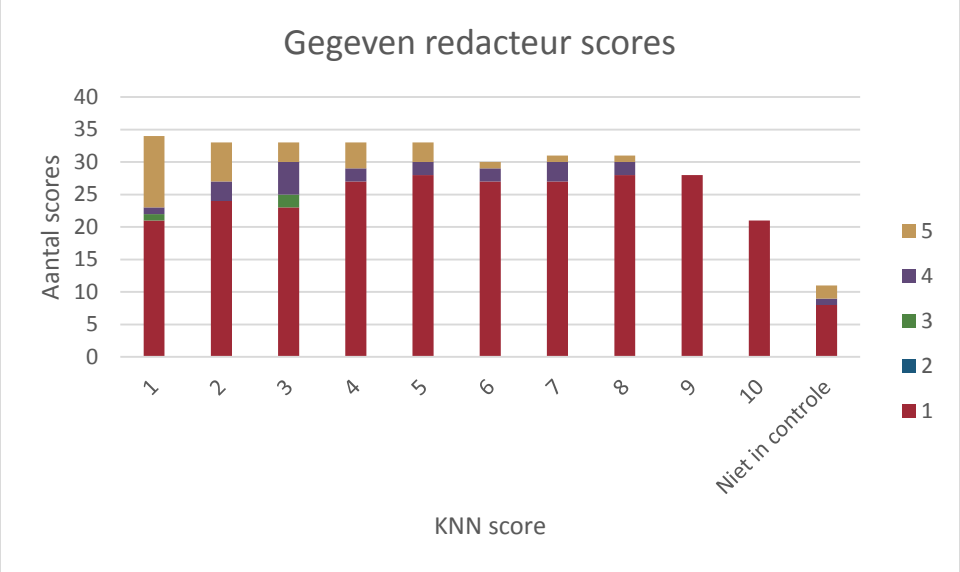
06.2 BURGERLIJK RECHT



Score	Survey 0
1	26
2	4
3	4
4	7
5	35
Alleen in controle	0
Niet gevonden	23

Toegevoegde suggesties: 27

06.3 STAATS- EN BESTUURSRECHT



Score	Survey 0
1	30
2	0
3	1
4	7
5	22
Alleen in controle	0
Niet gevonden	5

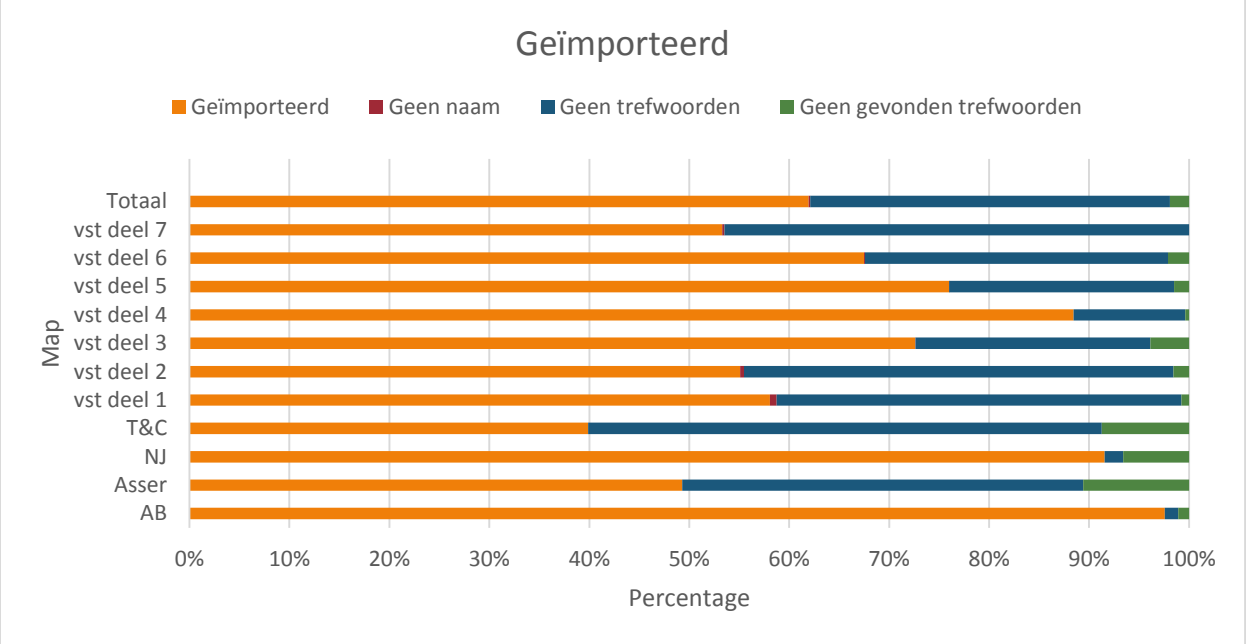
Toegevoegde suggesties: 7

07. CONCLUSIE

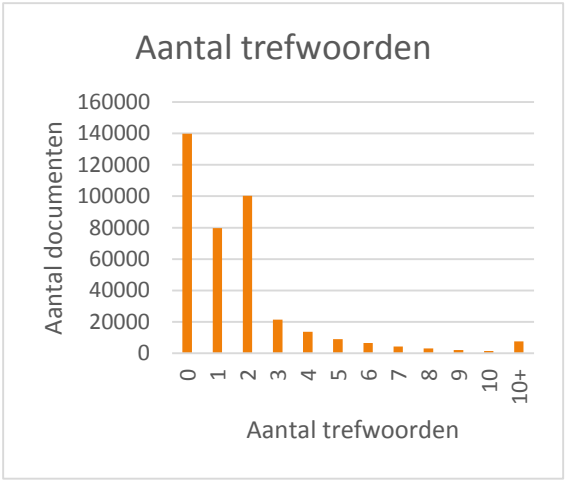
Voor elke analyse is eerst een aparte conclusie opgeteld. Van deze drie is vervolgens weer een totaal conclusie getrokken betreffende de inzetbaarheid van KNN voor het automatisch suggereren van trefwoorden.

07.1 CONCLUSIE DOCUMENTEN SET

De resultaten uit hoofdstuk zijn in de grafiek Geïmporteerd samengevat. Hierin is te zien dat het merendeel (62%) van de documenten wel is geïmporteerd.



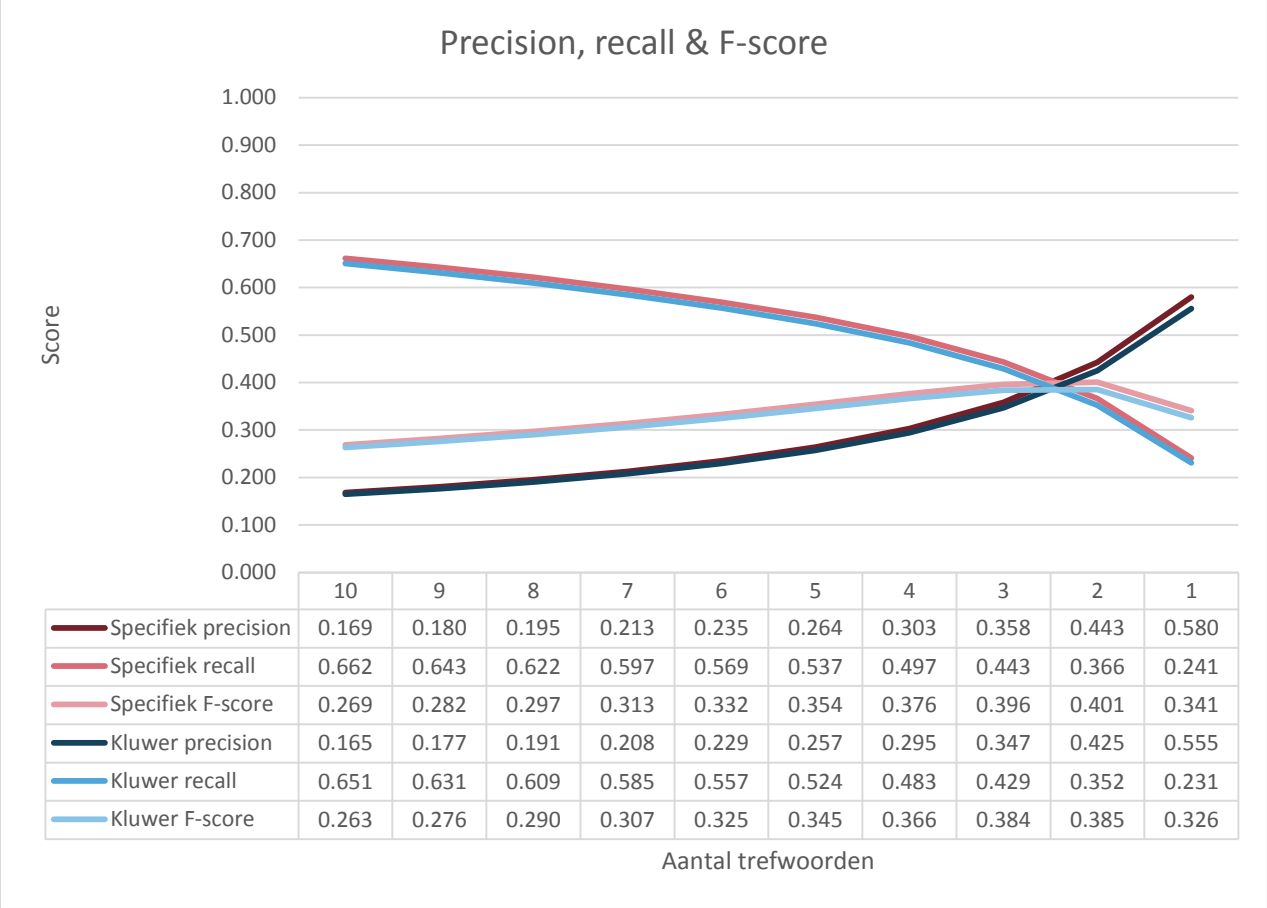
Van de keren dat een bestand niet geïmporteerd kon worden kwam in dit 94,51% van de gevallen door het ontbreken van trefwoorden, in 5,02% van de gevallen kwam het door het niet kunnen vinden van trefwoorden en in 0,46% van de gevallen kwam het door het ontbreken van de naam.



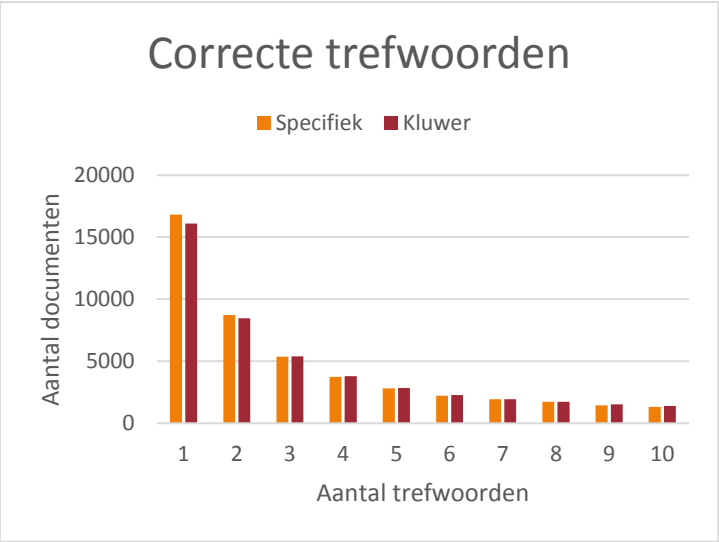
Bij de documenten zijn ook redelijk veel verschillen te zien in het aantal trefwoorden dat een document heeft. Van de documenten met trefwoorden zijn er veel documenten met één of twee trefwoorden. Documenten met meer dan twee trefwoorden zijn er relatief weinig.

07.2 Conclusie documenten KNN

Wat opvalt is dat er bijna geen verschil zit wanneer KNN zoekt over alle documenten of alleen bij de documenten binnen de set. Omdat het aantal document van VakstudieAC velen malen groter is zijn bij deze grafiek slechts 10.000 documenten van VakstudieAC gebruikt zodat de resultaten van VakstudieAC niet de andere sets overschaduwd.



	10	9	8	7	6	5	4	3	2	1
Specifiek precision	0.169	0.180	0.195	0.213	0.235	0.264	0.303	0.358	0.443	0.580
Specifiek recall	0.662	0.643	0.622	0.597	0.569	0.537	0.497	0.443	0.366	0.241
Specifiek F-score	0.269	0.282	0.297	0.313	0.332	0.354	0.376	0.396	0.401	0.341
Kluwer precision	0.165	0.177	0.191	0.208	0.229	0.257	0.295	0.347	0.425	0.555
Kluwer recall	0.651	0.631	0.609	0.585	0.557	0.524	0.483	0.429	0.352	0.231
Kluwer F-score	0.263	0.276	0.290	0.307	0.325	0.345	0.366	0.384	0.385	0.326

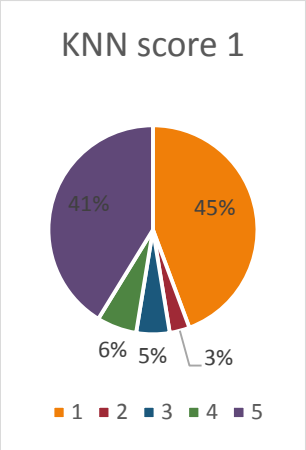


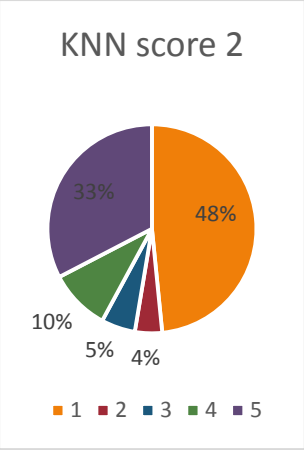
Aantal trefwoorden	Specifiek aantal documenten	Kluwer aantal documenten
0	2317	2317
1	241	239
2	240	230
3	327	297
4	620	590
5	381	325
6	363	342
7	410	367
8	411	358
9	485	444
10	25500	25786

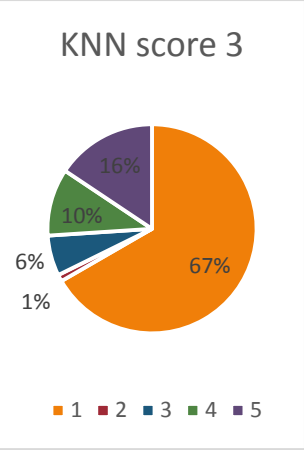
07.3 CONCLUSIE REDACTEUREN

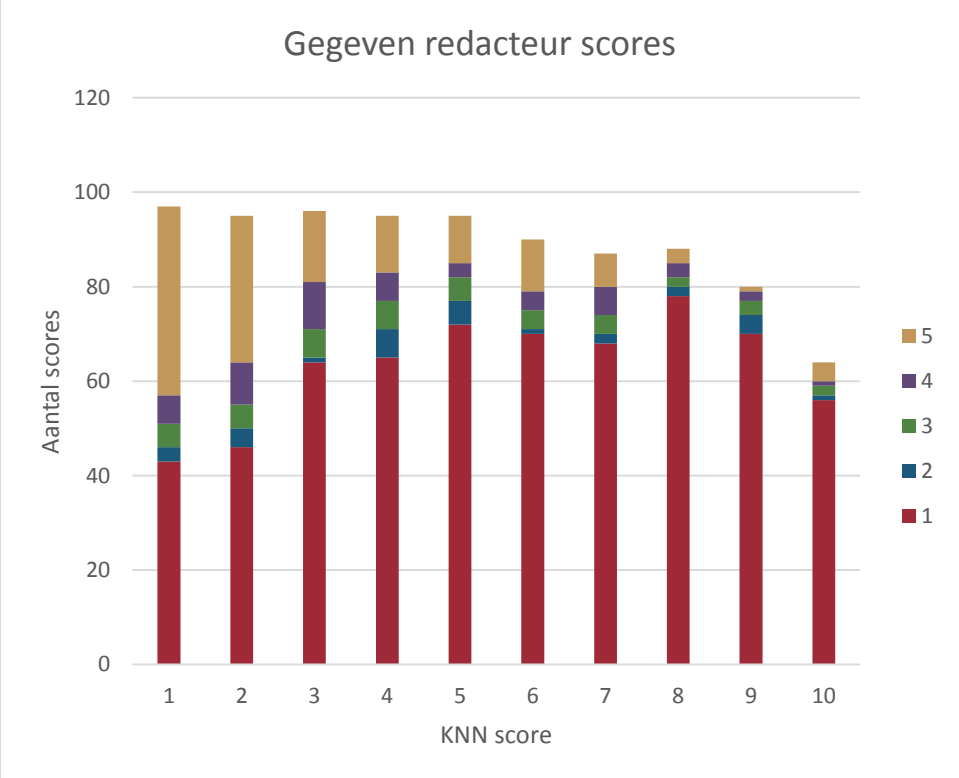
Te zien is dat de scores 1 & 5 het vaakste gegeven worden door de redacteuren en de scores daartussen (2,3&4) worden niet zo veel gebruikt.

De eerste twee trefwoorden scoren goed waarbij de laagste score minder dan de helft van de keren is toegekend. Bij het derde trefwoord is de laagste score bij 67% toegekend. Bij de trefwoorden hierna loopt dit aantal alleen nog maar verder op.









07.4 CONCLUSIE TOTAAL

Uit de documenten analyse is gebleken dat de meeste documenten die geïmporteerd zijn één of twee trefwoorden hebben. Uit de KNN analyse is de beste F-score bereikt bij twee trefwoorden want de F-score wordt het meeste beïnvloed door de laagste score of dit nou de precision is of de recall. Deze twee analyses komen met elkaar overeen omdat de precision het hoogste zal zijn bij één of twee trefwoorden omdat de meeste documenten zoveel trefwoorden toegewezen hebben.

Verder scoort KNN wat beter wanneer alleen binnen de correcte documenten set wordt gezocht naar lijkende documenten. Dit was niet te zien bij VakstudieAC maar dit komt doordat de set van VakstudieAC het merendeel is.

Uit de redacteuren analyse is gebleken dat eerste twee trefwoorden ook aanzienlijk het beste scoren. Of dit ook een verband heeft met het gemiddeld aantal toegewezen trefwoorden is niet zeker.

08. ADVIES

Om alleen goede suggesties te geven kan het beste de top twee trefwoorden gebruikt worden.

Ook is het mogelijk om met een vervolgonderzoek te zoeken naar een mogelijkheid om de suggesties van KNN te verbeteren.