Recommender Systems voor het realtime aanbieden van nieuwssecties

Thomas Janssen

23 januari 2007

Voorwoord

Deze scriptie is geschreven ter afsluiting van mijn Bachelor voor de studie Informatica aan de Radboud Universiteit Nijmegen. Ik wil allereerst bij deze een aantal mensen bedanken voor hun bijdrage. Als eerste wil ik Splendense bedanken voor het beschikbaar stellen van de data van Headliner, welke voor deze scriptie is gebruikt. Zonder deze data zou deze scriptie niet mogelijk zijn geweest.

Daarnaast zou ik mijn begeleider vanuit de universiteit, Tom Heskes, willen bedanken voor het sturen tijdens de beginfase en het kritisch bekijken van de scriptie gedurende de ontwikkeling ervan. Mede dankzij hem is het resultaat geworden tot wat het is.

Inhoudsopgave

1 Introductie

Het Internet wordt steeds vaker gebruikt om informatie op te zoeken. Steeds meer en meer informatie komt beschikbaar op deze manier. Het biedt niet alleen een gemakkelijke en snelle manier om informatie in te winnen, het biedt tevens de mogelijkheid om zoveel mogelijk mensen te bereiken op een relatief goedkope manier. Er ontstaan steeds meer websites waarop content wordt aangeboden. Denk hierbij aan e-commerce en hubs, sites die linken naar een veelvoud van andere sites. Voorbeelden van deze laatste zijn sites die nieuws verzamelen en aanbieden van een groot aantal websites. Deze scriptie zal zich hierop richten.

Veel gebruikers vragen veel van de server waarop een website draait en daardoor zijn optimalisaties nodig. Deze scriptie zal zich richten op het onderzoek hoe Data Mining, en daarin Recommender Systems en Clustering specifiek, kan bijdragen aan het aanbieden van nieuws die specifiek gericht is op de bezoeker en hoe deze optimalisaties daardoor mogelijk zijn. Het zal als volgt ingedeeld worden. Hoofdstuk 2 zal beschrijven wat Data Mining is en welke technieken ervoor bestaan. Hoofdstuk 3 zal beschrijven wat Recommender Systems zijn, waarvoor ze bedoeld zijn en wat ze kunnen. Hoofdstuk 4 zal Clustering beschrijven, welke gebruikt zal worden voor het efficiënter aanbieden van nieuwsitems. Hoofdstuk 5 zal het onderzoek beschrijven en welke middelen daarvoor gebruikt worden. De resultaten zullen gepresenteerd worden in hoofdstuk 6. Een discussie van deze resultaten en aanverwante zaken zullen daarna besproken worden in hoofdstuk 7. Deze scriptie zal eindigen met een conclusie in hoofdstuk 8.

2 Data Mining

Er ontstaat steeds meer informatie in de wereld. Deze informatie groeit zodanig dat het vaak voor mensen niet meer te doen is om in redelijke tijd deze te analyseren en daaruit waardevolle informatie af te leiden. Een oplossing voor dit probleem is Data Mining. Data Mining betreft het probleem van het vinden van patronen en nuttige informatie uit een grote hoeveelheid beschikbare data. Patronen die voor mensen niet of nauwelijks meer herkenbaar zijn, kunnen door middel van technieken uit Data Mining gevonden worden. Zo kunnen verschillende technieken voorspellen hoe het klimaat van de aarde zich zal gedragen gedurende een bepaalde periode, gebaseerd op informatie uit het verleden. Data Mining omvat zaken uit verschillende gebieden, zoals Machine Learning en Information Retrieval.

De verschillende technieken binnen Data Mining zijn op te delen in supervised en unsupervised technieken. In de volgende sectie zal worden uitgelegd wat het verschil is.

2.1 Supervised vs Unsupervised technieken

Data Mining technieken zijn op te delen in supervised en unsupervised technieken. Deze sectie zal beschrijven wat deze twee groepen inhouden en welke technieken daaronder vallen.

Supervised technieken proberen aan de hand van bekende data en bekende classificaties van deze data, nieuwe data te categoriseren binnen deze verschillende classificaties. Het is daarbij dan ook belangrijk dat van elk object uit de dataset bekend is tot welke categorie deze behoort. Technieken die supervised zijn, zijn onder andere Decision Trees en Bayesian Classifiers.

Unsupervised technieken proberen aan de hand van de beschikbare data te onderzoeken welke categorieën er gecreëerd kunnen worden en nieuwe data in te delen bij één van deze categorieën. Daarvoor hoeft op voorhand niet het aantal categorieën bekend te zijn. Het is aan het algoritme of de ontwerper om het aantal categorieën te bepalen. Een techniek die valt onder de unsupervised technieken, is Clustering Analysis. Deze scriptie zal zich op deze techniek richten. Hoofdstuk 4 zal deze techniek bespreken.

Information Retrieval omhelst het zoeken naar informatie binnen documenten. Een Recommender System is een voorbeeld van een techniek die zich schaart binnen de Information Retrieval. In het volgende hoofdstuk zal worden ingegaan op de rol van Information Retrieval binnen het aanbevelen van items.

3 Recommender Systems

Doordat het aanbieden van specifieke items aan een gebruiker zorgt voor een persoonlijkere benadering en service, maken meer en meer websites gebruik van Recommender Systems die dit mogelijk maken. Deze sectie zal beschrijven wat Recommender Systems zijn [4, 10, 12, 13].

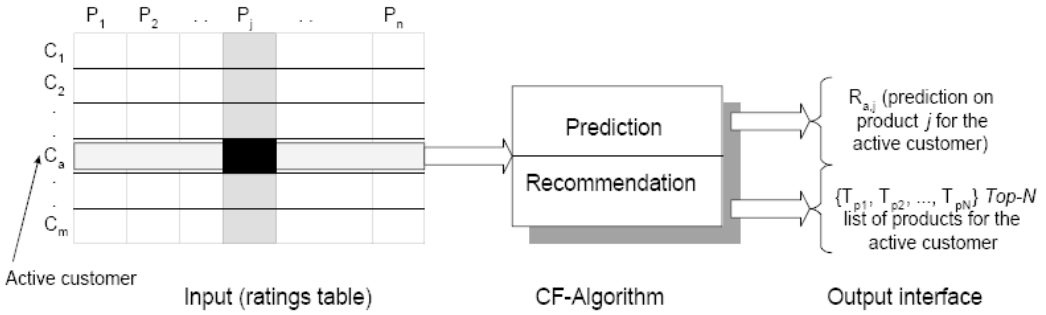
Grote sites die bezoekers het gemak bieden om vanuit de luie stoel te kopen, zoals Amazon.com, bieden duizenden, zo niet miljoenen producten aan. Voor de bezoeker is het dan ook niet mogelijk om alle producten waar te nemen. Het is voor de bezoeker dan ook uitdagend om toch de producten te vinden die zijn of haar interesse hebben en zodoende deze alsnog te kopen. Voor de webwinkel is het belangrijk om zoveel mogelijk producten te verkopen. Voor dit soort problemen zijn Recommender Systems ontwikkeld.

Recommender Systems zijn systemen die gebruikt worden om een persoonlijk aanbod te genereren voor de gebruiker. Het is een Informatie Filtering techniek, welke twee problemen probeert op te lossen; het "prediction problem" en het "top-N Recommendation problem" [12]. In het eerste geval wordt op basis van andere items de waarde van een specifiek item voorspeld. Hieruit kan dan afgeleid worden of een item interessant is of niet. De tweede techniek probeert te bepalen welke items het meest van interesse kunnen zijn voor de gebruiker en zal deze aanbieden als aanbeveling. Ze betreffen een techniek binnen de Information Retrieval, waarbij getracht wordt om informatie te achterhalen binnen een grote hoeveelheid gegeven data.

Recommender Systems zijn gebaseerd op een bepaalde filtering techniek, waarbij verschillende methoden gebruikt kunnen worden. Deze zijn Collaborative, Content-based, Demographic, Utility-based en Knowledge-based Filtering methoden [3]. Hiervan zal Collaborative Filtering worden besproken in sectie 3.1. In sectie 3.2 zal besproken worden welke twee technieken er bestaan om van gebruikers de interesse voor bepaalde items te ontdekken.

3.1 Collaborative Filtering

Collaborative filtering is één van de meest gebruikte technieken die gebruikt worden binnen Recommender Systems en waarschijnlijk ook één van de meest succesvolle [3, 4, 12]. In 1992 kwam het voor het eerst ter sprake in [6]. Deze techniek baseert zich op informatie van verschillende gebruikers en probeert te ontdekken welke relaties er bestaan tussen verschillende gebruikers. Op basis daarvan probeert het aanbevelingen op te stellen die bij de gebruiker passen. Het GroupLens systeem [7] was het eerste systeem dat gebruik maakte van deze techniek om automatisch aanbevelingen te genereren van Usenet berichten. Het idee achter dit systeem was dat er een omgeving van gebruikers met dezelfde interesses wordt opgesteld en gebaseerd op deze omgeving kunnen gebruikers interessante berichten aangeboden worden die anderen binnen deze omgeving ook interessant vinden.



Figuur 1: Van gebruikers naar voorspellingen en aanbevelingen [12]

User-Based Collaborative Filtering en Item-Based Collaborative Filtering [16] zijn twee technieken om Top-N Recommendation aan te pakken. Deze scriptie zal zich richten op de eerste mogelijkheid en zal besproken worden in sectie 3.1.1.

3.1.1 User-based Collaborative Filtering

User-Based Collaborative Filtering [11, 16] wordt het meest gebruikt voor het aanbevelen van items. Deze populariteit komt voort uit de lage complexiteit en de hoge kwaliteit die het oplevert. De aanpak maakt direct gebruik van de aanname dat een gebruiker toebehoort aan een groep van gebruikers die dezelfde soort interesses hebben. Dit maakt het mogelijk om aanbevelingen te doen op basis van deze groep en deze gelijkheid.

User-based Collaborative Filtering maakt gebruik van een set van m gebruikers $C = \{c_1, ..., c_m\}$ en een set van n producten $P = \{p_1, ..., p_n\}$ (zie Figuur 1). Daarnaast maakt het gebruik van waarden die gegeven worden aan de verschillende items, welke een mate van interesse bepalen (de ratings). Het proces valt op te splitsen in twee delen, namelijk het formeren van een neighborhood en het voorspellen van interesse of het aanbevelen van de top-N items. We definiëren c_a als de gebruiker waarvoor de recommendation gedaan dient te worden (het CF-algorithm in Figuur 1). Om de neighborhood te formeren, wordt allereerst de overeenkomst berekend tussen alle $c_j \in C \setminus \{c_a\}$ en de huidige gebruiker c_a. Vaak wordt hier Pearson correlation of cosine relation [11] voor gekozen (zie vergelijking 2 in sectie 5.2 voor meer informatie over de cosine relatie). De x best scorende gebruikers vormen de neighborhood. Vanuit deze neighborhood wordt aan de hand van de ratings voor de items van elke gebruiker c_j de voorspelling bepaald of de top-N recommendation gemaakt. Een veel gebruikte methode om de Top-N te bepalen is om de frequentie te nemen van een attribuut over alle gebruikers in de neighborhood. Algoritme 1 beschrijft het proces om tot een Top-N Recommendation te komen.

Algorithm 1 Collaborative Filtering, Top-N Recommendation [11]

Input: een gebruiker u, een dataset D en het aantal items op te leveren N.
Output: lijst van top-N items.

size := aantal gebruikers in D
attributen := aantal attributen van een gebruiker in D.
for i=1 t/m *size*
    Bereken de similarity tussen u en D_i.
end

Neighborhood *hood* := x meest gelijkende gebruikers.
for j=1 t/m *attributen*
    Bereken de frequentie van attribuut j over alle gebruikers in *hood*.
end

Lever de N items op met de hoogste frequentie.

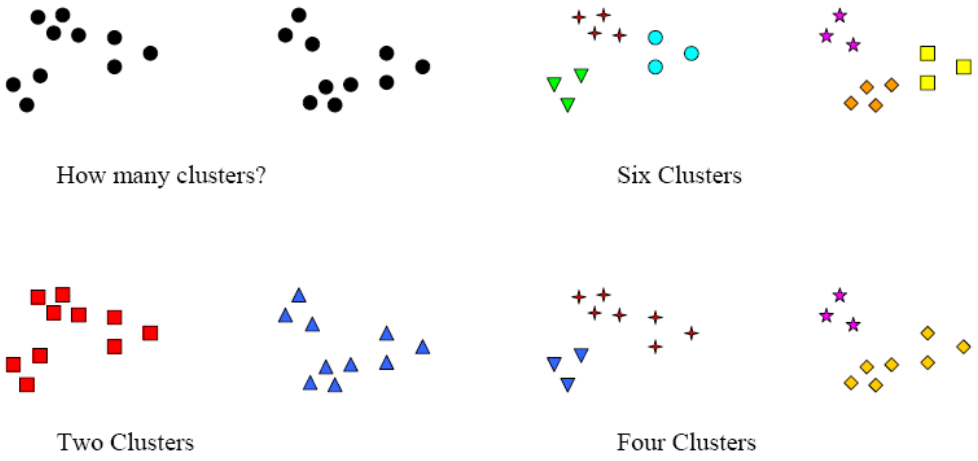
3.2 Feedback

Gebruikers hebben bepaalde interesses voor bepaalde items. Het is alleen vaak lastig om achter deze interesses te komen om zodoende daarmee de input te genereren voor Recommender Systems. In deze sectie wordt besproken hoe men achter deze interesses kan komen.

Expliciete Feedback en Impliciete Feedback zijn twee strategieën om achter de interesses te komen van bezoekers [9]. Expliciete feedback houdt in dat gebruikers expliciet kunnen aangeven wat ze interessant vinden. Hier kan gedacht worden aan het waarderen van films of boeken door middel van een cijfer of het aangeven of een bepaald restaurant bevalt of niet. Aan de hand van deze expliciete waarderingen kan een Recommender System zijn werk doen. Het voordeel van een dergelijke manier is dat een gebruiker direct kan aangeven waar zijn interesses liggen. Het nadeel is echter dat veel gebruikers niet de moeite nemen om waarderingen uit te spreken. Wanneer dit niet gebeurt, of in beperkte mate, zal het systeem zich niet goed kunnen richten op deze gebruiker en wordt bruikbare informatie gemist. In bepaalde systemen kan dit niet wenselijk zijn.

De andere aanpak is het gebruik van Impliciete Feedback, die op Expliciete Feedback een aanvulling kan bieden of als alternatief gebruikt kan worden. Deze techniek probeert niet aan de hand van door gebruikers gegeven waarderingen de interesses af te leiden, maar juist uit onderliggende patronen. Gebruikers laten vaak allerlei sporen achter. Het is bijvoorbeeld mogelijk om aan de hand van het surfgedrag van een persoon af te leiden waar zijn of haar interesses liggen. Stel dat er veel websites worden bezocht die gaan over voetbal, dan is de kans groot dat deze sport tot de interesse behoort. Onzichtbaar van de gebruiker wordt dus een model opgesteld wat de interesses bepaalt. Het voordeel hiervan

is dat er redelijk eenvoudig van praktisch elke gebruiker de interesses gevonden kunnen worden. De gebruiker is hier dan onbewust mee bezig en hoeft zich daar dan ook niet druk om te maken. Tevens worden de waarderingen realtime bijgewerkt, in tegenstelling tot expliciete feedback. Een nadeel is wel dat het niet altijd eenduidig is welke waarderingen uit de data gegenereerd kan worden. Dit heeft tot gevolg dat sommige gebruikers geen geschikte data opleveren om iets waardevols mee te doen, doordat er teveel ruis in voor kan komen. Binnen deze scriptie zal gebruik gemaakt worden van Impliciete Feedback.



Figuur 2: Clustering algoritme kan leiden tot een verschillend aantal clusters [15]

4 Clustering

Er zijn verschillende technieken binnen Data Mining. Eén van deze technieken is Clustering. Deze sectie zal beschrijven wat Clustering is en op welke manier deze toegepast kan worden.

Clustering [15] is een techniek om een grote hoeveelheid data in te delen in groepen die een bepaalde betekenis hebben. De datapunten binnen een cluster hebben een zo groot mogelijke gelijkheid met elkaar en een zo groot mogelijke ongelijkheid met andere datapunten binnen andere clusters. Het leren van een model vindt plaats aan de hand van observatie van de gegeven data. Figuur 2 geeft grafisch weer welk resultaat een cluster algoritme kan hebben. Het doel van het algoritme is om een groepering van datapunten te genereren, welke de clusters vormen. Deze clusters kunnen gebruikt worden om bepaalde voorspellingen te maken of om informatie uit te genereren.

Clustering kan bij Recommender Systems ingezet worden als voorbewerking van de data. Het doel is daarmee om de snelheid van het aanbevelen te verhogen, doordat er niet meer gebruik gemaakt wordt van alle gebruikers, maar een subset daarvan. De verwachting is dat de snelheid aardig verhoogd kan worden. Een nadeel hiervan is echter dat de kwaliteit zal dalen [2].

4.1 K-Means

Er zijn verschillende technieken om clustering toe te passen. Eén van deze technieken is K-Means. K-Means [8] is een veelgebruikte methode om data te clusteren (zie algoritme 2). Elk cluster wordt gerepresenteerd door de mean (centroid). Het algoritme verlangt een dataset en een input k, welke de hoeveelheid aan te maken clusters representeert. Het aantal clusters ligt vooraf dan ook al vast. Eerst worden er k random partities aangemaakt. Een veel gebruikte methode is om k willekeurige datapunten aan te wijzen als de mean van de k clusters. Voor elk datapunt uit de dataset wordt bepaald tot welke centroid van de k clusters deze zich het dichtst bevindt en zal daar aan worden toegewezen. Als alle datapunten zijn ingedeeld, wordt per cluster de mean opnieuw bepaald uit alle datapunten behorende tot de cluster. Dit proces herhaalt zich totdat aan een stopcriteria wordt voldaan. Deze kan bijvoorbeeld zijn wanneer de bezetting van de clusters niet meer (significant) verandert.

Algorithm 2 K-Means [8, 15]

1: **Input:** een dataset D, k het aantal clusters dat gevormd dient te worden.
2: **Output:** k clusters.
3:
4: Kies k random punten uit D als de centroids van de k clusters.
5: **while** Clusterbezetting niet stabiel
6: Wijs elk punt uit D toe aan de cluster met de dichtstbijzijnde centroid.
7: Bereken de centroid van elk cluster opnieuw.
8: **end**

Om te bepalen hoe dicht een datapunt zich bij de mean van een cluster bevindt, wordt vaak de afstand gemeten tussen het punt en de mean. Bekende maten zijn de Euclidean distance en Manhatten distance [15]. Ook andere gelijkheidsmaten, zoals de cosine similarity, kunnen gebruikt worden om de gelijkheid met de mean te bepalen. Waar bij de Euclidean distance wordt getracht om de afstand zo klein mogelijk te krijgen, wordt geprobeerd bij de cosine similarity een zo hoog mogelijke waarde te behalen.

Een probleem van dit algoritme is echter dat het kan eindigen in een lokaal minimum waar het niet meer uit kan komen. Door random een aantal datapunten aan te wijzen als initiële centroids, zal elke run ook een andere verdeling van de clusters opleveren. Een methode die dit probleem gedeeltelijk aanpakt, is Bisecting K-Means.

4.2 Bisecting K-Mean

Bisecting K-Means is gebaseerd op K-Means. Echter, er wordt niet vanaf het begin af aan uitgegaan van k verschillende clusters, maar van maar één cluster met daarin alle datapunten uit de dataset. Vanuit deze cluster wordt toegewerkt naar k clusters, door telkens een cluster in tweeën te splitsen (zie algoritme 3).

Algorithm 3 Bisecting K-Means [5, 15]

1: **Input:** een dataset D, k het aantal clusters dat gevormd dient te worden.
2: **Output:** k clusters.
3:
4: Initialiseer een lijst C met clusters met als inhoud een cluster bestaande uit alle datapunten.
5: **while** k clusters nog niet bereikt
6: Kies een cluster $c \in C$ en verwijder c uit C.
7: **for** i=1 t/m trials
8: Bisect c door middel van het K-Means algoritme.
9: **end**
10: Selecteer de twee beste clusters en voeg deze toe aan C.
11: **end**

Er zijn verschillende manieren om een cluster te kiezen uit de set van clusters. Er kan random een cluster gekozen worden, een cluster die de meeste winst oplevert of de grootste cluster. Steinbach et al.[14] heeft onderzocht dat de verschillen tussen de verschillende manieren minimaal zijn. Een populaire manier is het kiezen van de grootste cluster.

De clusters die voortkomen uit algoritme 3 zijn in eerste instantie nog niet geheel optimaal. Elke cluster dat gesplitst wordt, is wel lokaal optimaal door middel van het K-Means algoritme, maar dat betekent niet dat het totale resultaat een lokaal minimum heeft bereikt. Vaak worden de resulterende clusters gebruikt als initiële clusters voor het gewone K-Means algoritme. Het vervangt daarmee de random gekozen initiële clusters. Zodoende kunnen de clusters worden verfijnd tot een lokaal minimum. Het voordeel van dit algoritme is dat het clusters van relatief uniforme grootte produceert [14].

Nu we hebben vastgesteld wat Recommender Systems zijn en wat Clustering inhoudt, kan het onderzoek besproken worden dat is uitgevoerd. Beide technieken zullen hierin gecombineerd worden en de resultaten zullen worden besproken in hoofdstuk 6.

5 Onderzoek

In deze sectie zal het onderzoek besproken worden wat wordt uitgevoerd. Recommender Systems en Clustering zullen in dit onderzoek worden gecombineerd om te kijken welk effect Clustering heeft op het aanbevelen van nieuwssecties.

Tienduizenden mensen per dag gebruiken een website om zich te voorzien van nieuwsfeiten uit de wereld. Niet elke bezoeker is geïnteresseerd in hetzelfde soort nieuws. Omdat een dergelijke site vrij groot kan zijn wanneer het gebruik maakt van veel nieuwsbronnen, is het voor de bezoeker niet gemakkelijk om altijd het nieuws te vinden dat voor hem of haar geschikt is. Om de gebruiker te helpen in het vinden van interessante onderwerpen, kan gebruik gemaakt worden van Recommender Systems. Deze systemen zijn in staat om uit de grote hoeveelheid gegevens die bezoekers achterlaten patronen te ontdekken en daardoor aanbevelingen te maken. Uit literatuur is gebleken dat normale Collaborative Filtering technieken het werk goed doen. Het probleem is echter dat de performance er aardig onder kan leiden wanneer de website groot is en het aantal bezoekers hoog is. Veel implementaties zijn daarop niet berekend. Er zijn verschillende mogelijkheden om hier verandering in te brengen. Eén van de mogelijkheden is het toepassen van Clustering technieken. Deze moeten de performance sterk verbeteren.

Het doel van het inzetten van Clustering is dat er kleine neighborhoods worden gevormd, waardoor niet de gehele dataset wordt gebruikt bij het genereren van de aanbevelingen. Een kleine subset van het aantal gebruikers zal als input dienen voor de Recommender System, waarna de normale stappen gelden van uitvoer.

5.1 Dataset

Om de algoritmes te testen en te onderzoeken wat de invloed is van de verschillende implementaties, is het van belang om een geschikte dataset te gebruiken. In deze sectie zal beschreven worden waar deze dataset vandaan komt en hoe deze eruit ziet.

Eén van de grotere websites in Nederland die nieuws aanbieden van een groot aantal verschillende nieuwsbronnen is Headliner[1]. Deze site trekt per dag duizenden unieke bezoekers, waarvan er gemiddeld per bezoek tien nieuwsitems worden aangeklikt. Elk van deze nieuwsitems zijn ondergebracht in één of meerdere secties, zoals binnenlands nieuws, buitenlands nieuws en sport nieuws. In totaal zijn er 147 verschillende secties. Sommige secties hebben enige overlap. Zo zal de sectie sport onder andere nieuwsitems bevatten die ook voorkomen in de sectie voetbal. Echter, ze zullen als verschillende secties worden beschouwd, omdat elke sectie een mogelijk andere doelgroep aantrekt. De dataset waarvan gebruik gemaakt wordt in deze scriptie, bevat als attributen de secties die aanwezig zijn op de website, in totaal dus 147 attributen. Elk object wordt beschreven aan de hand van deze secties, waar-

[1] www.headliner.nl

in voor elke sectie staat aangegeven hoe vaak een nieuwsitem uit de desbetreffende sectie de interesse heeft getrokken van de bezoeker. Dat betekent dat voor elke unieke bezoeker wordt bijgehouden het aantal keer dat een bepaalde sectie heeft geleid tot een bezoek aan de bron van het artikel. De gebruiker wordt geïdentificeerd aan de hand van het unieke IP-adres.

De dataset die gebruikt zal worden, zal dus bestaan uit 147 attributen (de secties) en 10.221 gebruikers, waarbij elke gebruiker minimaal twee secties heeft bezocht en beslaat een tijdsspanne van twee weken. De aangeleverde dataset bevatte oorspronkelijk meer gebruikers. Veel van deze gebruikers bevatten echter maar één attribuutveld met een waarde groter dan nul. Twee gebruikers hebben pas enigszins een overeenkomst wanneer bij beide gebruikers minimaal één overeenkomstig attribuutveld de waarde groter dan nul is. Een gebruiker met maar één attribuutveld dat groter is dan nul kan alleen die sectie aanbevelen dat een andere overeenkomstige gebruiker al interessant vindt. Een dergelijke gebruiker bezit daardoor geen relevante waarde voor het systeem. De oorspronkelijke dataset is dan ook bewerkt om deze gebruikers te verwijderen. Er zullen geen missende waarden voorkomen, vanwege het feit dat elke sectie bekend is. Wanneer deze niet bezocht is, met andere woorden geen interessante link voor de bezoeker bevat of niet door de bezoeker is opgemerkt, zal deze de waarde nul bevatten. Doordat weinig bezoekers elke sectie interessant zullen vinden, zal deze dataset ook erg sparse zijn, dat wil zeggen uit veel nulvelden bestaan. Dit wordt uitgedrukt in de sparsity level [11]:

$$sparsity\ level = \frac{aantal\ nulvelden}{totaal\ aantal\ velden} = \frac{1.469.121}{1.502.487} = 0.9778 \tag{1}$$

5.2 Implementatie

In deze sectie zullen relevante gegevens besproken worden die gebruikt zijn voor het genereren van de resultaten.

De Top-N Recommendation voor Collaborative Filtering zal gevormd worden aan de hand van algoritme 1. Voor de overeenkomst is gekozen voor de cosine similarity. Deze wordt gedefinieerd als:

$$cosine(a, b) = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|_2 * \|\vec{b}\|_2} \tag{2}$$

Het voordeel van de cosine similarity is dat deze uitgaat van de richting van een vector en niet van de grootte. Dat is vooral binnen deze dataset belangrijk, omdat het niet belangrijk is hoe vaak een gebruiker bepaalde items bezoekt, maar juist de relatie tussen verschillende secties en verschillende gebruikers. Gebruikers die dezelfde secties hebben bezocht maar in verschillende mate, worden met de cosine similarity als gelijken gezien.

Om een vergelijking te maken tussen de kwaliteit en de snelheid van klassieke Collaborative Filtering en Clustering, wordt er gebruik gemaakt van Bisecting K-Means (zie algoritme 3) met de cosine gelijkheid, voorgesteld in [5]. Deze levert over het algemeen een beter resultaat dan de gewone K-Means algoritme [14]. De kwaliteit van de clusters zal gemeten worden door de overeenkomst van elk punt uit het cluster met de centroid bij elkaar op te tellen. De bedoeling is het maximaliseren van de kwaliteit van alle clusters [5]. Laat π_j cluster j zijn, X een gebruiker gezien als vector en c_j de centroid van cluster j. Dan kan de kwaliteit gemeten worden als:

$$Q(\{\pi_j\}_{j=1}^{k}) = \sum_{j=1}^{k} \sum_{X \in \pi_j} X^T c_j \tag{3}$$

Het stopcriteria met t als iteratie in de berekening van K-Means (zie algoritme 2 en 3) en $\epsilon = 0.01$ is als volgt gedefinieerd:

$$\mid Q(\{\pi_j^t\}_{j=1}^{k}) - Q(\{\pi_j^{t+1}\}_{j=1}^{k}) \mid \leq \epsilon \tag{4}$$

Vooraf zal elke gebruiker genormeerd worden naar unit-length, dat wil zeggen dat elke gebruiker als vector lengte 1 bevat. Dit is gedaan vanwege efficiëntie maatregelen. Merk op dat de cosine relatie hierdoor dezelfde ordening oplevert als de Euclidean Distance. Tevens heeft dit tot gevolg dat de centroid berekend kan worden als het gemiddelde van elke gebruiker in de cluster [5]. Het gemiddelde wordt als volgt berekend:

$$m_j = \frac{1}{n_j} \sum_{X \in \pi_j} X \tag{5}$$

met n_j het aantal gebruikers in cluster π_j. Het resulterende gemiddelde hoeft door deze berekening niet unit-length te bevatten. Om dit te bereiken kunnen we de centroid als volgt berekenen:

$$c_j = \frac{m_j}{\|m_j\|} \tag{6}$$

Deze centroid ligt gemiddeld het dichtst bij alle gebruikers binnen de cluster.

Om de resultaten te genereren, is gebruik gemaakt van een implementatie in Java. Deze implementatie is gedraaid op een Pentium-M 1.5 GHz machine met 1280 MB intern geheugen.

5.3 Evaluatie

Om de effecten van het clusteren na te gaan, is het nodig om een aantal zaken te evalueren. Deze sectie zal een aantal zaken bespreken die geëvalueerd dienen te worden.

Grote websites trekken veel bezoekers. Veel bezoekers betekenen vaak een grote belasting voor het systeem waarop de website draait. En veel bezoekers betekent ook veel informatie

die bezoekers achterlaten. Omdat de aanbevelingen real-time moeten worden aangegeven, is het nodig dat dit snel en efficiënt kan gebeuren. Daar zijn een aantal zaken die in de gaten gehouden dienen te worden:

- Kwaliteit
- Snelheid

5.3.1 Evaluatie Kwaliteit

Het meten van de kwaliteit van een Recommender System is niet eenvoudig. Elke gebruiker heeft een eigen smaak. De kans dat twee gebruikers één en dezelfde smaak hebben, is erg klein. Daardoor is het lastig te meten in hoeverre een aanbeveling juist is voor de gebruiker. Want wat voor de één goed is, is juist voor een ander niet relevant, zelfs in gevallen met bijna gelijke interesses.

Er zijn verschillende manieren om de kwaliteit te kunnen meten van Recommender Systems. Sarwar [11] geeft voor het meten van de kwaliteit een aangepaste methode uit de Information Retrieval, Precision en Recall. Deze methoden zijn licht aangepast, omdat er binnen Recommender Systems gebruik wordt gemaakt van een vast aantal items dat aanbevolen wordt, de Top-N. De manier om te evalueren bestaat allereerst uit het aanmaken van een trainingset en een testset uit de dataset [13]. De dataset wordt random verdeeld als 20% testset en 80% trainingset. Van elke bezoeker in de testset wordt random één non-zero item gekozen en verborgen in een aparte set, genoemd de hiddenset. De rest zal fungeren als input-items. Voor elke gebruiker uit deze testset zal de neighborhood gevormd worden aan de hand van de trainingset en de input-items van de gebruiker. Dat betekent dat de neighborhood alleen wordt gevormd aan de hand van de trainingset. Aan de hand van deze neighborhood worden de Top-N items bepaald van items die nog niet bekeken zijn door de bezoeker. Het systeem dient te achterhalen of het de hidden-item uit de hiddenset kan aanbevelen. De lijst die wordt gevormd, wordt de top-N list genoemd. Om het aantal hits te bepalen, wordt er voor elke gebruiker bekeken of het hidden-item voorkomt in de top-N list. Items die voorkomen in zowel de Top-N set en in de hiddenset, worden toegerekend aan de hitset.

De evaluatie vindt nu plaats als volgt: zoals gezegd zal er gebruik gemaakt worden van methoden uit de Information Retrieval, namelijk Precision en Recall. Precision is het percentage van het aantal relevante items dat is gevonden over het totale aantal items dat mogelijk is. Binnen Recommender Systems betekent dat, dat het percentage wordt aangegeven door het aantal hits over het totaal aantal elementen in de top-N list. Dit uit zich in vergelijking 7:

$$Precision = \frac{|hitset|}{N} \tag{7}$$

Recall is het percentage van relevante items op het totaal aantal relevante items dat bestaat. Nu is het vrijwel niet mogelijk om alle relevante items te markeren, daarom is ook hier

een aanpassing aangebracht. In plaats van uit te gaan van alle mogelijke relevante items, wordt er uitgegaan van alle bekende relevante items. Dit uit zich in vergelijking 8:

$$Recall = \frac{|hitset|}{|hiddenset|} \tag{8}$$

Het probleem van Precision en Recall is dat ze van nature met elkaar conflicteren. Wanneer de Precision vergroot dient te worden, zal de Recall zich verlagen. Wanneer de Recall vergroot dient te worden, zal dit ten koste gaan van de Precision. Vanwege dit conflicterende verschijnsel, is er een andere metric ontwikkeld, F1 metric. Precision en Recall zijn beide van belang bij het beoordelen van de kwaliteit van de Recommender System en F1 zoekt een balans tussen beide metrics om zodoende tot een kwaliteitsoordeel te komen. Deze metric uit zich in vergelijking 9:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{9}$$

Voor elk van de gebruikers uit de testset zal de Precision en Recall en daarmee de F1 metric worden berekend. Het gemiddelde hiervan zal dienen als de waarde voor de gehele testset. Om tot een betrouwbare uitslag te komen, zal deze evaluatie tien keer uitgevoerd worden en daarvan het gemiddelde worden berekend. Dit verkleint de kans op een toevalligheid resulterend in een extreem hoge of extreem lage score. Dit geeft een betrouwbaarder beeld van de kwaliteit van het Recommender System.

Om het klassieke Collaborative Filtering algoritme eerlijk te vergelijken met het Bisecting K-Means algoritme, zal eerst het Bisecting K-Means algoritme uitgevoerd worden. Er zal worden bijgehouden voor elke gebruiker uit de testset van hoeveel buren deze gebruik heeft gemaakt. Voor het klassieke Collaborative Filtering algoritme zal dan dezelfde hoeveelheid buren gebruikt worden voor het aanbevelen van de Top-N items. Deze methode geeft een eerlijkere en overzichtelijkere manier van vergelijken tussen beide algoritmen. Beide algoritmen maken dus gebruik van dezelfde hoeveelheid buren per gebruiker uit de testset.

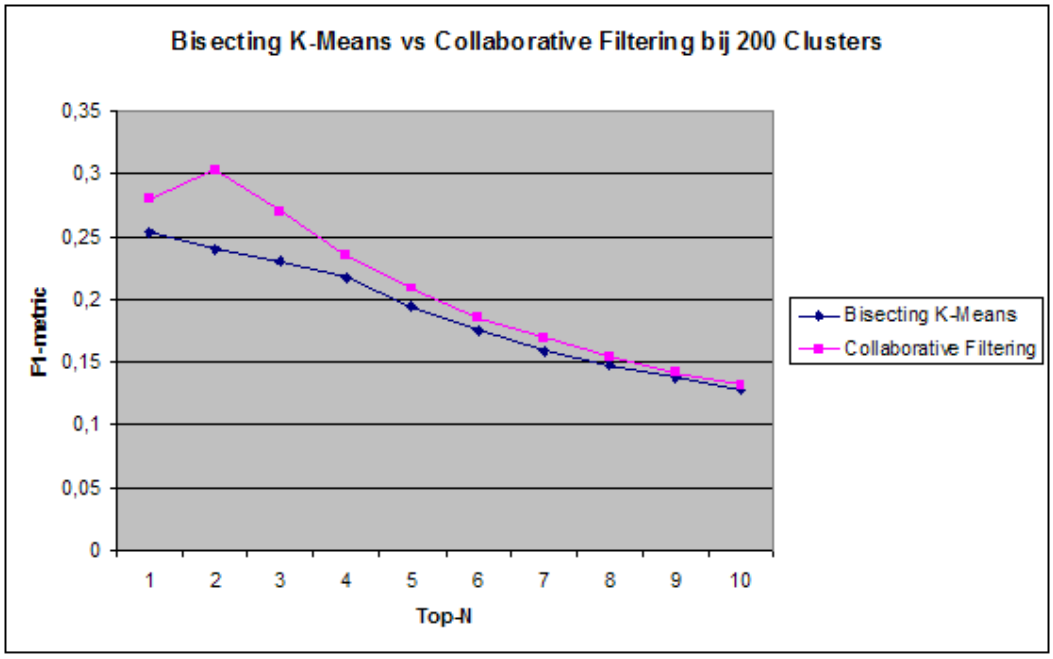
Nu de evaluatie van de kwaliteit is beschreven, wordt in de volgende sectie de evaluatie van de snelheid van het Recommender System beschreven.

5.3.2 Evaluatie Snelheid

Niet alleen de kwaliteit van de Recommender System is van belang, ook de snelheid waarmee het systeem zijn werk uitvoert is belangrijk. Vooral in drukke tijden zijn er duizenden bezoekers tegelijk op de website aanwezig. Het doel van een Recommender System is om real-time aanbevelingen te genereren voor de desbetreffende bezoeker. Het is dan ook van belang om te kijken naar de snelheid waarmee het systeem haar taken uitvoert.

De snelheid zal berekend worden aan de hand van het aantal aanbevelingen dat per seconde afgegeven kan worden. Dit geeft een indicatie van de snelheid van het algoritme. Net als

bij de evaluatie van de kwaliteit zal ook hier elke test tien keer uitgevoerd worden om tot een betrouwbaar gemiddelde te komen. Belangrijk is om te realiseren dat de snelheden alleen betrekking hebben op de aanbevelingen. Voorbewerking van data en het clusteren ervan wordt niet meegenomen in deze evaluatie. Dit zijn vaste tijden die niet samenhangen met het specifiek aanbevelen van items.



Figuur 3: Bisecting K-Means vs Collaborative Filtering

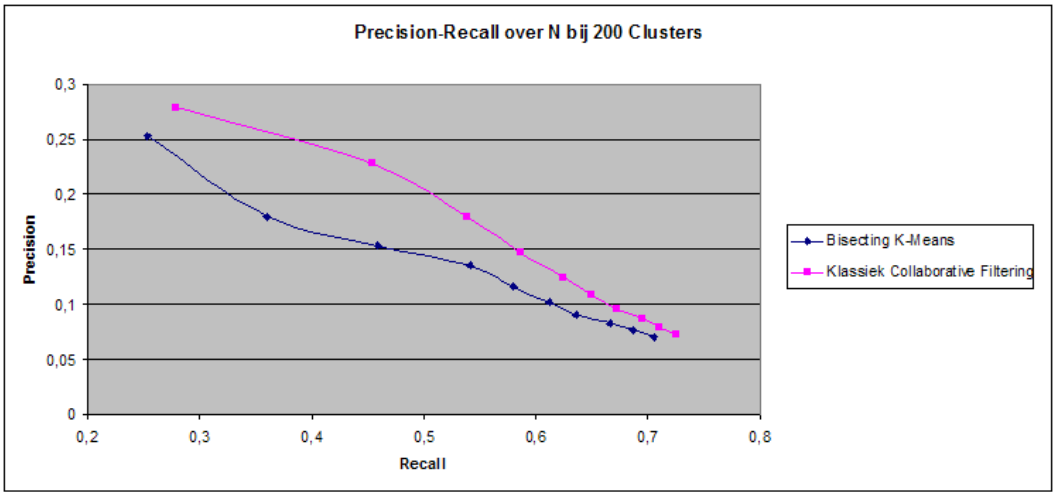
6 Resultaten

Dit hoofdstuk zal de resultaten presenteren die verkregen zijn uit het uitvoeren van de implementatie. Belangrijk is het om op te merken dat de resultaten specifiek voor de gebruikte dataset gelden en kunnen afwijken van resultaten verkregen met behulp van een andere dataset.

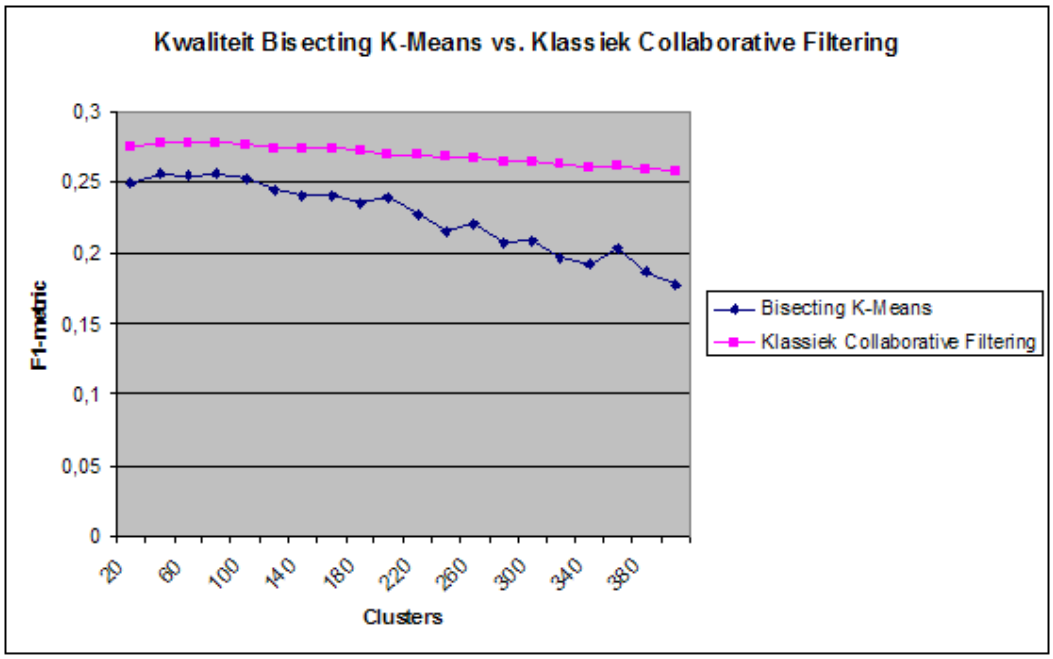
6.1 Kwaliteit

In figuur 3 staan zowel de Collaborative Filtering als de Bisecting K-Means methode uiteengezet wat betreft de F1 kwaliteit ten opzichte van het aantal items dat aanbevolen kan worden. Er kan worden afgeleid dat de Collaborative Filtering methode voor elke N een betere kwaliteit aflevert. Het verschil tussen beiden is echter niet groot. Naarmate N stijgt, wordt het verschil tussen beide kleiner. Een keerzijde daarvan is dat een te hoge waarde voor N tot een kwaliteit leidt die niet voldoende is.

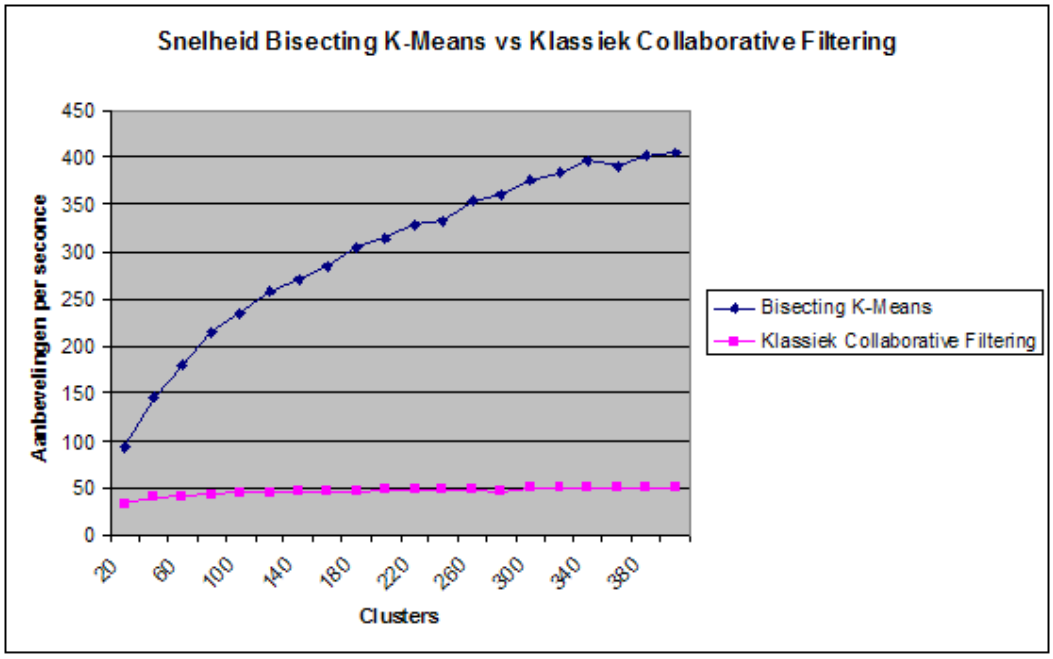
Als we specifiek naar het verloop van Precision en Recall kijken, dan kunnen we zien dat beide maten op elkaar moeten inleveren. Figuur 4 (met 200 clusters) laat zien dat wanneer N wordt gevarieerd, de Precision en Recall elkaar beïnvloeden. Wanneer de Recall wordt verhoogd door N te verhogen, neemt de Precision af. Dit volgt uit de definitie van Precision en Recall (sectie 5.3.1).



Figuur 4: Verhouding tussen Precision en Recall voor Bisecting K-Means met 200 Clusters



Figuur 5: Verloop van F1-metric over verschillend aantal clusters met N=3



Figuur 6: Aantal aanbevelingen per seconde over het aantal clusters.

Als we kijken naar de kwaliteit van het systeem over verschillend aantal clusters (figuur 5) met $N = 3$, zien we dat naarmate het aantal clusters wordt vergroot, de kwaliteit afneemt. In eerste instantie neemt de kwaliteit weinig af. Pas wanneer het aantal clusters oploopt tot boven de 200 zien we een sterke daling bij Bisecting K-Means. Het klassieke Collaborative Filtering proces zien we minder sterk in kwaliteit dalen. Dit kan verklaard worden doordat het aantal gebruikers dat van invloed kan zijn op de aanbevelingen niet groot hoeft te zijn. Het clusteringsalgoritme is daar meer vatbaar voor, omdat gebruikers die van belang kunnen zijn, kunnen eindigen in een ander cluster en daardoor niet de belangrijke informatie kunnen meegeven.

6.2 Snelheid

Figuur 6 geeft het aantal aanbevelingen per seconde weer dat het systeem kan genereren. Naarmate er meer clusters gevormd worden, kunnen er meer aanbevelingen gegenereerd worden. Dit komt doordat het aantal gelijke gebruikers in een cluster afneemt, waardoor er minder buren worden bekeken. Naarmate het aantal clusters groter wordt, vlakt de snelheidswinst wel af. Dit is te verklaren doordat het aantal buren in een cluster dermate klein wordt, dat een aantal clusters meer of minder niet veel meer uitmaakt voor de grote van de cluster. Voor de klassieke Collaborative Filtering methode geldt dat het aantal buren gelijk is aan het aantal gebruikers in een cluster. De snelheid zal daardoor vrij constant blijven. Het verschil tussen beide methoden kan oplopen tot boven de 800%.

Uit deze resultaten kunnen we constateren dat Clustering een grote snelheidswinst kan opleveren onder een klein kwaliteitsverlies. Het zal op elk moment een mindere kwaliteit leveren dan het klassieke Collaborative Filtering proces, maar het profijt dat gehaald kan worden uit de snelheidswinst is dermate groot dat er serieus overwogen kan worden om Clustering in te zetten bij grote websites.

7 Discussie

In dit hoofdstuk worden de resultaten wat dieper besproken en zullen relevante zaken aan de orde komen.

7.1 Kwaliteit

Zoals in hoofdstuk 6 wordt weergegeven, varieert de kwaliteit over de snelheid en de methode die gebruikt is. Collaborative Filtering is in alle gevallen in staat een hogere kwaliteit af te leveren. Dit komt mede doordat alle gebruikers worden gebruikt om de omgeving te bepalen en aanbevelingen te doen. Vanuit alle gebruikers is het mogelijk om meer vertrouwen op te bouwen in secties die aanbevolen worden. Het clusteringsalgoritme produceert een iets mindere kwaliteit. Dit komt doordat tijdens het aanbevelen van secties alleen de gebruikers uit het cluster tot de beschikking staan. Dit leidt ertoe dat niet de gebruikers die het dichtst bij de actieve gebruiker staan de buren vormen, maar juist de gebruikers die gemiddeld het dichtst bij deze gebruiker ligt. Een cluster genereert een omgeving die voor de desbetreffende gebruiker gemiddeld de hoogste overeenkomst oplevert. Het kan dus zijn dat een gebruiker uit een ander cluster veel relevante informatie voor de actieve gebruiker bevat, maar dus niet kan bijdragen aan de aanbeveling. Het klassieke Collaborative Filtering algoritme maakt wel gebruik van deze gebruiker. Het verschil in kwaliteit is echter niet groot. Hierdoor is het mogelijk om niet alleen naar de kwaliteit te kijken, maar ook naar andere zaken, zoals snelheid.

7.2 Snelheid

Zoals te zien is in figuur 6 neemt de snelheid toe naarmate het aantal clusters toeneemt. Dit komt doordat er minder buren geëvalueerd hoeven te worden. Minder buren leveren meer aanbevelingen per seconde op. Collaborative Filtering levert gemiddeld een snelheid op van 45 recommendations per seconde. Het gebruik van Clustering laat zien dat er een zeer grote verbetering in snelheid te behalen is. Als er gebruik gemaakt wordt van 200 clusters komt het algoritme tot boven de 300 aanbevelingen uit. Trekken we dat door naar de 400 clusters, dan komen we zelfs op een snelheid van ruim 400 aanbevelingen per seconde uit. Dat levert een snelheidswinst op van ruim 800%. Bij grote sites met veel gebruikers is het van belang om veel gebruikers tegelijk te kunnen bedienen, zonder dat de kwaliteit sterk achteruit loopt. Combineren we figuur 5 met figuur 6, dan kunnen we afwegen welke snelheid we dienen te behalen en wat voor kwaliteit we daarvoor terugkrijgen.

7.3 Nieuwe secties

Een website zal nooit hetzelfde blijven. Er zal dan ook bij deze dataset in de loop der tijd aanpassingen verricht worden, waarbij er secties kunnen worden toegevoegd. Voor het klassieke Collaborative Filtering levert dit niet veel problemen op. Een sectie wordt toegevoegd en elke gebruiker krijgt er een extra attribuut bij. De sectie zal echter wel eerst

gevonden dienen te worden door een bezoeker, voordat het ooit een keer aanbevolen kan worden. Dit is echter bij elk attribuut het geval. Voor Clustering brengt het een iets grotere verandering met zich mee. Op het moment dat er een nieuwe sectie wordt toegevoegd, zijn de oude clusters direct niet meer geldig. De clusters zijn gebaseerd op de centroid van alle gebruikers. Wanneer er een nieuwe sectie wordt toegevoegd, moeten de gebruikers als eerst bijgewerkt worden. Daarnaast moet elke centroid nog bijgewerkt worden dat het ook de nieuwe sectie bevat. Feitelijk betekent dat dat het uitgebreid moet worden met een extra attribuut waarvan de waarde nog nul is. De clusters hoeven niet opnieuw gegenereerd te worden, omdat er nog geen bezoeken aan die sectie zijn verricht. Het probleem van nieuwe secties kan daarmee vrij snel afgehandeld worden.

7.4 New user problem

Een bekend probleem binnen Recommender Systems is het probleem van nieuwe gebruikers. Nieuwe gebruikers hebben nog geen enkele sectie bekeken. Daardoor is het niet mogelijk om aan de hand van de gegevens uit het verleden een vergelijking te maken met andere gebruikers van het systeem. Voor dit type gebruikers kan er dan ook geen aanbevelingen gemaakt worden. Om toch een aanbeveling te genereren, kan het systeem ervoor kiezen om de meest bezochte secties te genereren door middel van een frequency count. Op deze manier geeft het systeem alsnog een gevoel van aanbeveling, zij het dat deze niet persoonlijk is. Het geeft echter wel een inzicht in secties die populair zijn bij het merendeel van de bezoekers. Daardoor is de kans groter dat één of meerdere van deze secties tot de interesse behoren van de nieuwe gebruiker.

7.5 Tijdsspanne van data

Het is van belang om een juiste tijdsspanne te kiezen waarin de data wordt verzameld. De gebruikte dataset (zie sectie 5.1) heeft een tijdsspanne van twee weken. In deze twee weken is van elke bezoeker bijgehouden welke sectie de interesse heeft gewekt. Het probleem van nieuwssecties is dat er niet een onbeperkte hoeveelheid data gegenereerd kan worden. Er dient een bepaalde maximum tijd te zitten aan de houdbaarheid van data. Wanneer er alsmaar data verzameld wordt, dan krijgen secties die altijd nieuws aanbieden, zoals Binnenlands nieuws, een grotere kans op aanbeveling dan secties die "seizoensgebonden"zijn. Met seizoensgebonden wordt bedoeld dat bepaalde secties alleen in een bepaalde periode van belang zijn. Een sectie die nieuws beschrijft over schaatsen heeft vaak alleen de interesse van bezoekers in het schaatsseizoen. Daarnaast is het niet interessant om schaatsnieuws aan te bevelen midden in de zomer, wanneer de meeste schaatsers niet actief bezig zijn.

Een te korte tijdspanne leidt tot het probleem dat veel gebruikers niet voldoende de site hebben bezocht om een goede aanbeveling te genereren. Een Recommender System gebaseerd op user-user relaties staat of valt met het aantal gebruikers dat de site heeft bezocht en met het aantal secties dat zij bezoeken. Wanneer deze te laag is, zullen aanbevelingen

niet goed op maat gegenereerd worden.

Door de juiste tijdsspanne te kiezen, stel je het systeem in staat om een zo goed mogelijke aanbeveling te genereren. Het is van belang om de tijd niet te kort te kiezen, omdat dan de aanbevelingen kwalitatief te kort kunnen schieten door een geringe hoeveelheid gelijkwaardige gebruikers. Een te lange tijd leidt tot mogelijke verkeerde aanbevelingen, zoals aanbevelingen die niet meer relevant hoeven te zijn vanwege de seizoensgebondenheid.

8 Conclusie

Recommender Systems zijn in staat om aanbevelingen te doen voor gebruikers om zodoende een extra service te verlenen of om extra verdiensten te kunnen genereren. Belangrijk daarbij is dat dit real-time gebeurt. Omdat grote sites, zoals Amazon en Headliner, veel bezoekers tegelijk krijgen, zullen systemen in staat moeten zijn om veel aanbevelingen tegelijk te kunnen genereren. Deze scriptie heeft laten zien dat het klassieke Collaborative Filtering algoritme te kampen heeft met het probleem dat het niet erg schaalbaar is. Als alternatief is bekeken of Clustering een voordeel oplevert wat betreft deze schaalbaarheid, waarbij wordt getracht het kwaliteitsverlies te beperken. Aan de hand van data van Headliner is aangetoond dat Clustering een grote snelheidswinst kan opleveren, zonder veel kwaliteit in te leveren. Voor grote sites is het dus interessant om te kijken of Clustering voor hen geschikt is, wanneer zij gebruik (willen) maken van Recommender Systems.

De resultaten die hier gepubliceerd zijn, zijn specifiek gericht op de dataset die is gebruikt. Het is belangrijk om op te merken dat de winst per dataset kan verschillen. Voordat tot een methode als Clustering wordt overgegaan, is het verstandig om te kijken welke voordelen er precies behaald kunnen worden. Deze scriptie heeft echter een inzicht gegeven in de mogelijke winst die behaald kan worden.

Referenties

[1] Basu, C., Hirsh, H. & Cohen, W., (1998) Recommendation as Classification: Using Social and Content-Based Information in Recommendation, *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 714–720

[2] Breese, J. S., Heckerman, D. & Kadie, C., (1998), Empirical Analysis of Predictive Algorithms for Collaborative Filtering, *In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 43–52.

[3] Burke, R., (2002), Hybrid Recommender Systems: Survey and Experiment, *User Modeling and User Adapted Interaction*, 12(4), 331–370

[4] Deshpande, M. & Karypis, G., (2004), Item-Based Top-N Recommendation Algoritms, *ACM Transactions on Information Systems*, 22(1), 143–177

[5] Dhillon, I. & Modha, D., (2000), Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1–2), 143–175

[6] Goldberg, D., Nichols, D., Oki, B., & Terry, D., (1992), Using Collaborative Filtering to weave an information tapestry, *Communications of the ACM*, 35(12), 61–70

[7] Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R. & Riedl, J., (1997), GroupLens: Applying Collaborative Filtering to Usenet News, *Communications of the ACM*, 40(3), 77–87

[8] MacQueen, J. B., 1967, Some Methods for classification and Analysis of Multivariate Observations, *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1, 281–297

[9] Oard, D. & Kim, J., (1998), Implicit Feedback for Recommender Systems, *In Proceedings of the AAAI Workshop on Recommender Systems*

[10] Resnick, P. & Varian, H. R., (1997), Recommender Systems, *Communications of the ACM*, 40(3), 56–58

[11] Sarwar, B. M., Karypis, G., Konstan, J. & Riedl, J., (2000), Analysis of Recommendation Algorithms for E-Commerce, *Proceedings of the 2nd ACM Conference on Electronic Commerce*, 285–295

[12] Sarwar, B. M., Karypis, G., Konstan, J. & Riedl, J., (2002), Recommender Systems for Large-scale E-Commerce: Scalable Neighborhood Formation Using Clustering, *Proceedings of the 5th International Conference on Computer and Information Technology*

[13] Srikumar, K. & Bhasker, B. (2005), Personalised recommendation in e-commerce, *International Journal Electronic Business*, 3(1), 4–27

[14] Steinbach, M., Karypis, G. & Kumar, V., (2000), A comparison of document clustering techniques. *In KDD Workshop on Text Mining*

[15] Tan, P-N., Steinbach, M. & Kumar, V., (2006), Introduction to Data Mining.

[16] Ziegler, C-N., McNee, S. M., Konstan, J. A. & Lausen, G., (2005), Improving Recommendation Lists Through Topic Diversification, *Proceedings of the 14th International World Wide Web Conference*, 22–32