X-means: clusteren met een onbekend aantal clusters

Peter Hendrikson

BWI-werkstuk

Vrije Universiteit
Faculteit der Exacte Wetenschappen
Studierichting Bedrijfswiskunde & Informatica
De Boelelaan 1081a
1081 HV Amsterdam

september 2004

Voorwoord

Ter afsluiting van de verkorte opleiding Bedrijfswiskunde & Informatica dient er een werkstuk te worden geschreven. Dat is een literatuurstudie ter waarde van zes ECTS en moet gaan over een onderwerp met de raakvlakken B(edrijfskunde), W(iskunde) en I(nformatica).

Normaal gesproken gaat een student naar een docent voor een onderwerp, maar door mijn werk kwam ik in aanraking met het volgende probleem: *Is er een methode om in een berg data de onderliggende clusters te vinden, zonder dat van tevoren het aantal clusters bekend is?*

In dit verslag leest u het antwoord op deze vraag. Ik heb het op een manier proberen te schrijven zodat het praktisch bruikbaar is voor een econoom. Dat houdt onder andere in dat zijn behoefte naar een theoretische achtergrond vervuld is.

Een financieel-economisch paper dat clusteren gebruikt voor de analyse van concurrentie is in voorbereiding en het is de bedoeling dat dit paper input zal leveren voor het onderdeel clusteren.

Via deze weg wil ik in degene bedanken met het idee voor dit paper: Jaap Bos. Daarnaast natuurlijk ook mijn begeleider van de VU, Elena Marchiori.

Samenvatting

Dit werkstuk bevat een uitdieping van het paper X-means: Extending K-means with Efficient Estimation of the Number of Clusters van Dan Pelleg en Andrew Moore (200X).

Het onderwerp is naar boven gekomen doordat iemand in het kader van concurrentie-onderzoek de spelers op de bankmarkt op wil delen in clusters en de vraag stelde of er een methode is om in een berg data de onderliggende clusters te vinden, zonder dat van tevoren het aantal clusters bekend is.

X-means is een algoritme dat voortborduurt op *k*-means. De gebruiker geeft in dit geval niet de waarde *k* op, maar een interval waarbinnen *k* moet vallen. *X*-means begint met de ondergrens van *k* *k*-means uit te voeren en voegt vervolgens lokaal clusters toe. De beslissing om punten toe te voegen wordt genomen op basis van de BIC-score. Wanneer de bovengrens van *k* is bereikt, kiest *x*-means de output *k* met de hoogste BIC-score.

De BIC-score is een benadering van de Bayes factor en wordt in Pelleg & Moore (200X) gedefinieerd als

$$BIC\left(M_j\right) = \hat{l}_j(D) - \frac{p_j}{2} \cdot \log R .$$

Dit is een te maximaliseren functie en is een trade-off tussen *fit* (de eerste term) en *complexity* (de tweede term).

Door de implementatie van kd-trees en het gebruik van blacklisting wint het algoritme aan snelheid zonder dat er benaderingen aan te pas komen.

Inhoudsopgave

1. Inleiding

Naast mijn studie werk ik twintig uur per week als statistisch analist bij de afdeling Toezichtstrategie van De Nederlandsche Bank. Voor het toezichthoudend proces is inzicht in de concurrentie op de bankenmarkt een belangrijk gegeven. Maar in tegenstelling tot bijvoorbeeld de concurrentie in de supermarktbranche kan dit op de bankenmarkt niet direct worden waargenomen vanwege het homogene karakter van geld.

Door de spelers in de markt op te delen in clusters is analyse van concurrentie mogelijk. Door op basis van bepaalde kenmerken van banken te clusteren proberen we de data voor zichzelf te laten spreken en natuurlijke clusters te ontdekken. Pelleg & Moore (200X) hebben hiervoor een algoritme ontwikkeld, door hen *x*-means genoemd, waarvan in dit werkstuk de belangrijkste dimensies beschreven en uitgediept worden.

In paragraaf 2 staat een introductie tot het meten van concurrentie. Dan volgt in paragraaf 3 een beschrijving van de standaard methode voor clusteren, *k*-means. Een vernieuwde aanpak waarbij grenzen aan *k* opgegeven worden en het algoritme op basis van een heuristiek de optimale *k* bepaalt, staat in paragraaf 4. Dit is het x-means algoritme. Een belangrijk onderdeel van *x*-means is de beoordeling van een model, waarbij model in dit geval gelijk is aan een *k*-means oplossing voor één bepaalde waarde voor *k*. Hiervoor gebruikt het algoritme het Bayesian Information Criterion (BIC). Dit criterium wordt nader bestudeerd in paragraaf 5. In paragraaf 6 wordt ingegaan op de gebruikte datastructuren, kd-trees en blacklisting. Tot slot volgen in paragraaf 7 de conclusies.

2. Meten van concurrentie

Om kwantitatief inzicht in de markt te krijgen zijn verschillende indicatoren ontwikkeld, die bijvoorbeeld de verhouding van de markt en de mate van concentratie weergeven. Twee indicatoren zullen kort besproken worden, de concentratieratio CR_k en Herfindahl-Hirschman index. Deze paragraaf is gebaseerd op hoofdstuk 3 van Bikker (2004), waarin ook andere traditionele concentratie- en concurrentie-indicatoren voor het bankwezen te vinden zijn.

2.1 De concentratieratio CR_k

De *k* bank concentratieratio CR_k is gedefinieerd als

$$CR_k = \sum_{i=1}^{k} s_i \ . \qquad (1)$$

Hierbij wordt gesommeerd over de *marktaandelen s* van de grootste *k* banken. CR_k heeft dus per definitie een bereik tussen 0 en 1. Verschillende grootheden kunnen dienen als marktaandeel, maar het meest gangbaar is een CR_k op basis van totale activa, totale leningen of totale deposito´s. *k* Is in de meeste gevallen gelijk aan 3 of 5; CR_3 en CR_5 zijn in bankenland bekende begrippen.

Het nadeel van CR_3 en CR_5 is dat ze eigenschappen hebben die een vertekend beeld van de markt kunnen laten ontstaan. Het deterministische karakter van de waarden 3 en 5 is de oorzaak van de belangrijkste vertekening. In Nederland bijvoorbeeld zullen CR3 en CR5 niet zo veel verschillen, omdat Nederland met ABN Amro, Rabobank en ING bank drie banken heeft die relatief veel groter zijn dan de overige banken in Nederland. Op basis van CR5 zou dus geconcludeerd kunnen worden dat in Nederland de **5** grootste banken een groot deel van de markt in handen hebben, wat op zich natuurlijk een juiste conclusie is, maar niet de gewenste, namelijk dat **3** banken de grootste spelers zijn.

2.2 Herfindahl-Hirschman index

De Herfindahl-Hirschman index is de meest gebruikte index voor concentratie in de theoretische literatuur. De HHI is gedefinieerd als

$$HHI = \sum_{i=1}^{n} s_i^2 \ . \qquad (2)$$

Hierbij is *s* wederom het marktaandeel, en resulteert HHI in een getal tussen 1/n en 1. De laagste waarde, 1/n, wordt bereikt als alle banken in de markt een even groot marktaandeel hebben. In het geval van een monopolie is HHI gelijk aan 1.

Een nadeel van de HHI is dat naarmate er meer banken op de markt actief zijn, de index minder gevoelig is voor veranderingen in het aantal banken. Tevens is op de HHI dezelfde kritiek geuit als op de CR_k, namelijk dat deze maatstaven zich richten op de grootste banken en de veranderingen in de rest van de markt negeren.

2.3 Concurrentie door clustering

Het belangrijkste nadeel dat hierboven geschetst is, is dat in feite alleen de grote banken in aanmerking genomen worden. Door gebruik te maken van clusters is de hele markt in kaart te brengen. Voor zover bekend worden er in financieel-economische papers die op één of andere manier gebruik maken van clusters, meestal vaste, beredeneerde grenzen gebruikt. De nieuwe manier die wij proberen toe te passen laat de data voor zich spreken en ondekt zelf clusters banken.

3. K-means clustering

"Clustering methods aim at partitioning a set of data-points in classes such that points that belong to the same class are more similar than points belonging to different classes" – Stramaglia (2004)

Het bekendste clusteralgoritme is wellicht het klassieke *k*-means. In deze paragraaf volgt een beschrijving gebaseerd op Witten en Frank (2000). We introduceren de volgende notatie:
D = de verzameling datapunten waarover geclusterd wordt.

Het *k*-means algoritme kent de volgende stappen:

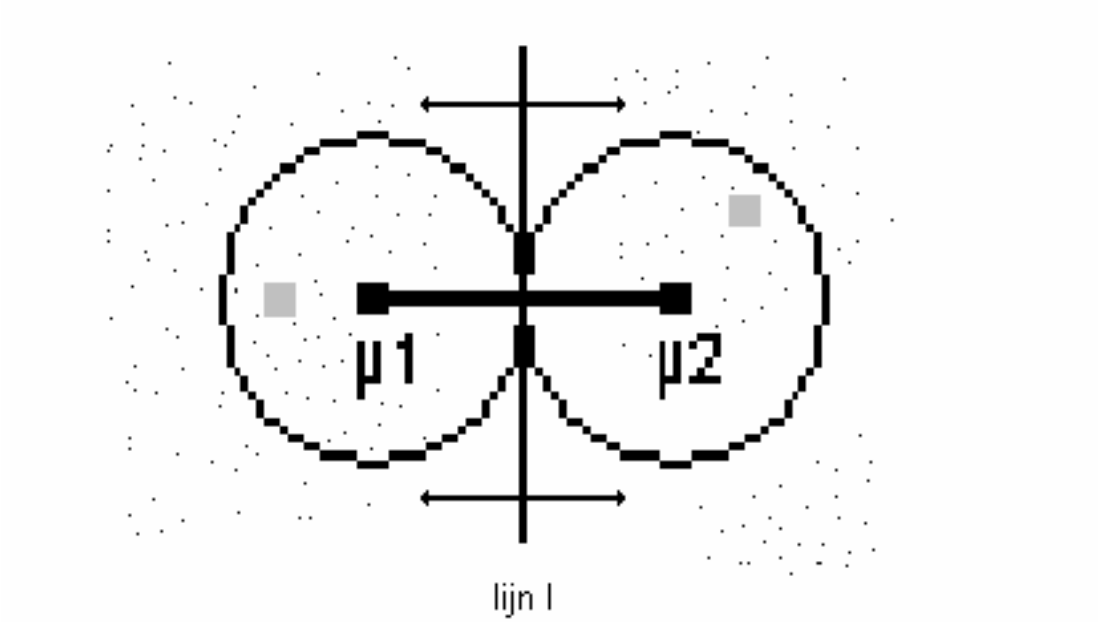
1. Opgeven van het aantal clusters *k*.
2. Generatie van *k* random punten, de zogeheten clustercentra.
3. Koppelen van elk punt in de verzameling D aan het dichtstbijzijnde clustercentrum.
4. Per clustercentrum-verzameling het nieuwe clustercentrum berekenen.
5. Herhalen van de punten 3 en 4 totdat de verzameling punten behorend bij de clustercentra niet meer veranderden of dat de verandering van een totale fout < ε is.

De *k* random gekozen clustercentra komen uit het bereik $[D_{min} ; D_{max}]$. Toekenning van de punten uit D aan de clustercentra vindt plaats op basis van de Euclidische afstand[1]. De nieuwe clustercentra worden berekend door in elk cluster het gemiddelde van alle punten in dat cluster te berekenen. Dit gemiddelde vervangt het oude clustercentrum. Vervolgens worden de punten uit D toegekend aan deze nieuwe clustercentra. Dit proces wordt herhaald tot de verzamelingen clusterpunten niet meer veranderen, of de verandering van de som van de Euclidische afstanden kleiner is dan ε, een willekeurig klein getal. Pelleg & Moore gebruiken in hun papers niet de som van Euclidische afstanden als criterium, maar meten de kwaliteit van een oplossing door de *gekwadrateerde* Euclidische afstanden tussen de datapunten en bijbehorende clustercentra te sommeren en vervolgens te delen door R = |D|, het aantal datapunten. Zij noemen dit de distortion.

Figuur 2 is een illustratie van het *k*-means algoritme, in dit geval een sferische, 2-dimensionale, 2-means clustering. De kleine puntjes stellen de datapunten voor. De punten μ1 en μ2 zijn de

[1] Euclidische afstand tussen twee n-dimensionale punten **a** en **b** is gedefinieerd als:
$\text{wortel}[(b_1-a_1)^2 + (b_2-a_2)^2 + \ldots + (b_n-a_n)^2]$

random-gekozen clusterscentra uit stap 2. Lijn l is de *decision-line* tussen de twee clustercentra. Deze staat loodrecht op de verbindingslijn tussen μ1 en μ2. Links van de lijn behoren datapunten op basis van de Euclidische afstand bij cluster μ1, rechts bij μ2. De grijze punten zijn de nieuwe clustercentra voor μ1 en μ2, berekend door de het gemiddelde van de punten links en rechts van de lijn l te nemen. Uit de figuur blijkt dat er waarschijnlijk drie clusters te onderscheiden zijn. Dit gegeven is niet met behulp van *k*-means uit de data te filteren, en is één van de grootste tekortkomingen van het algoritme. Er zijn oplossingen om het nadeel van de vaste *k* te omzeilen, wat meestal betekent dat k-means uitgevoerd wordt met $k = 2$, waarna per cluster recursief verder geclusterd wordt. Pelleg & Moore (200X) hebben een uitbreiding van *k*-means ontwikkeld waarbij ook zij lokaal verder clusteren. Dit is het *x*-means algoritme en komt in de volgende paragraaf aan de orde.



Figuur 1. Een *k*-means voorbeeld

Een ander nadeel van *k*-means is dat de uiteindelijke clusters niet de globale oplossing vormen, maar een lokale oplossing zijn van de minimalisatie van de som van de Euclidische afstanden. Andere random-punten bij initialisatie kunnen leiden tot een andere, wellicht betere oplossing. Het randomness effect doet zich alleen bij de initialisatie voor, daarna is *K*-means volledig deterministisch.

Het algoritme staat bekend om zijn traagheid bij grotere datasets. Er zijn verschillende verbeteringen voorgesteld om het algoritme te versnellen, maar meestal gaat het hierbij om benaderingen. Pelleg & Moore (2000) hebben een verbetering voorgesteld die niet benaderend is, maar deterministisch. Hiervoor gebruiken zij de technieken kd-trees en blacklisting, die ook in hun *x*-means algoritme terugkomen. Meer hierover in paragraaf 6.

Ter afsluiting van deze paragraaf willen we benadrukken dat buiten de nadelen die hierboven genoemd zijn, *k*-means natuurlijk een gemakkelijk en redelijk effectief algoritme blijft om te clusteren.

4. X-means

X-means is een algoritme waarbij ten opzichte van *k*-means niet de waarde *k* opgegeven wordt, maar een *interval* waarbinnen *k* moet vallen. *X*-means produceert vervolgens op basis van een heuristiek, de BIC-score, de beste waarde van *k*.

In Pelleg & Moore (200X) is het algoritme als volgt weergegeven:

1. *Verbeter-parameters*
2. *Verbeter-structuur*
3. Als $k > k_{max}$ stop en geef het model met de beste score.
   Anders: ga naar 1.

Het algoritme start met *k* gelijk aan de ondergrens, wat in het stappenplan hierboven als stap 0 toegevoegd zou kunnen worden. De stap *Verbeter-parameters* bestaat uit het runnen van *k*-means met de huidige *k* tot aan het eindcriterium van k-means voldaan is. Vervolgens wordt in de stap *Verbeter-structuur* bepaald of en waar nieuwe clustercentra moeten komen. Dit proces herhaalt zicht totdat de bovengrens van *k* bereikt is. Het model met de hoogste BIC-score, hierover in paragraaf 5 meer, wordt tenslotte gekozen als de beste representant van de clusters in de data.

Verbeter-structuur heeft als beginsituatie een stabiele *k*-means oplossing met *k* clustercentra en bijbehorende *ouder-regio's*, zoals we die vanaf nu zullen noemen. Lokaal, dus per cluster, valt de beslissing om het cluster al dan niet te splitsen, door om te beginnen per ouder-regio twee nieuwe centra te initieren, de *kinderen*. Deze worden binnen het ouder-regio volgens een random gekozen vector verplaatst. Per ouder-regio wordt nu een lokale *k*-means met de 2 kinderen uitgevoerd. Op basis van de BIC-score wordt bepaald of de 2 kinderen de clusters in de data beter representeren dan het ene ouder. Als dat zo is, vervangen de kinderen de ouder en komt er dus een cluster bij.

Het *x*-means algoritme maakt twee keer gebruikt van de BIC-score:

1. Aan het einde van het algoritme om het model te kiezen dat de clusters in de data het beste representeert.
2. Om te beoordelen of tijdens het proces van het lokale splitsen kinderen toegevoegd moeten worden.

Een aantrekkelijke eigenschap van de implementatie van *x*-means is het gebruik van *kd-trees* en *blacklisting*. Door deze technieken toe te passen hoeven niet telkens alle punten langsgelopen te worden, maar doorloopt het algoritme op een slimme manier de boom. In paragraaf 6 staat een informele beschrijving van deze technieken.

Naast het gebruik van *x*-means is het mogelijk herhaaldelijk *k*-means toe te passen en daarna de clusterverdeling te nemen met de optimale waarde voor de criteriumfunctie, wat bijvoorbeeld de BIC-score of de distortion kan zijn. De criteriumfunctie is dan niet helemaal eerlijk, in het eerste geval omdat *k*-means de som van de Euclidische afstanden tussen de datapunten en de bijbehorende clusterafstanden probeert te minimaliseren en *x*-means de BIC-score als criterium gebruikt. In het tweede geval geldt de omgekeerde redenatie.

Pelleg & Moore (200X) laten empirisch zien dat gemeten naar distortion *x*-means beter presteert dan *k*-means. Bij deze test werd een Gaussian gegenereerde dataset gebruikt en werd *k*-means het juiste aantal klassen opgegeven en *x*-means het interval [2, *k*]. De auteurs verklaren dit door het effect dat x-means alleen clusters toevoegt waar ze nodig zijn, in tegenstelling tot de eenmalige random plaatsing van de initiële clustercentra bij k-means.

Een andere vraag is hoe goed x-means is in het vinden van het juiste aantal klassen. Om te kunnen vergelijken, gebruikten de auteurs een variant van k-means die verschillende waarden voor *k* probeert en de beste resultaten op basis van de BIC-score weergeeft. De range waarbinnen de *k* in x-means viel, was [2, 2k]. *K*-means werd in 20 stappen uitgevoerd tot en met 2*k*.
X-means resulteerde in een configuratie die minder dan 15% van het werkelijke aantal klassen afligt. Voor *k*-means was dat 6%, zodat de conclusie is dat *k*-means het op dit gebied beter doet.

Wanneer de *k*-means oplossing met de beste configuratie gegeven de BIC-score vergeleken wordt met de *x*-means oplossing, blijkt de laatste beter te scoren. *X*-means scoort op basis van de BIC zelfs beter dan de onderliggende verdeling. Dit zou verklaard kunnen worden doordat de data beter weerspiegeld wordt door minder klassen dan er werkelijk zijn, bijvoorbeeld omdat twee of meer klassecentra zo dichtbij liggen dat ze eigenlijk één klasse zijn.

Een kwalitatief nadeel dat we nog willen noemen is dat in tegenstelling tot *k*-means, dat na de keuze van de random punten volledig deterministisch is, er bij x-means nog wel een fase is met een random punten keuze, namelijk bij de fase van het al dan niet toevoegen van clusters.

5. Bayesian Model Selection

Het *x*-means algoritme gebruikt de BIC-score om de uiteindelijke modellen te beoordelen en te beslissen of er clustercentra toegevoegd moeten worden. BIC-scoring is een vorm van Bayesiaanse statistiek, waarbij de posterior kansen Pr(Model$_j$|Data) gebruikt worden om model j te beoordelen. Op deze manier laten we "de data voor zichzelf spreken", een integrale eis uit de probleemstelling.

De basis van BIC-scoring is de Bayes factor. Kass en Raftery (1995) geven een goed te lezen introductie tot Bayes Factors, waarop is deze paragraaf gebaseerd.

5.1 Bayes Factor

We beginnen met data D en twee modellen, ofwel hypotheses, H_1 en H_2. Gegeven de a priori kansen pr(H_1) en pr(H_2) = 1 - pr(H_1) levert data D de a posteriori kansen pr(H_1|D) en pr(H_2|D) = 1 - pr(H_1|D). In formulevorm:

$$pr(H_k|D) = \frac{pr(D|H_k)pr(H_k)}{pr(D|H_1)pr(H_1) + pr(D|H_2)pr(H_2)} \quad (k = 1,2) \qquad (3)$$

Door de kansen te converteren naar de *odds*-schaal (odds = p / (1-p)) krijgen we de a posteriori odds

$$\frac{pr(H_1|D)}{pr(H_2|D)} = \frac{pr(D|H_1)}{pr(D|H_2)}\frac{pr(H_1)}{pr(H_2)}. \qquad (4)$$

Het quotiënt pr(H_1|D) / pr(H_2|D) geeft het op basis van de data verkregen bewijs weer van de voorkeur van H_1 ten opzichte van H_2. Uit (4) blijkt dat dit bewijs wordt verkregen door de a priori odds te vermenigvuldigen met de *Bayes factor*: pr(D|H_1) / pr(D|H_2). In woorden

posterior odds = Bayes factor * prior odds, (5)

Een andere intuïtieve benadering kan worden verkregen door de Bayes factor, kortweg B_{12} te beschouwen als de ratio van de posterior odds van H_1 ten opzichte zijn prior odds. In (6) is een en ander samengevat.

$$B_{12} = \frac{pr(D|H_1)}{pr(D|H_2)} = \frac{pr(H_1|D)}{pr(H_2|D)} \Big/ \frac{pr(H_1)}{pr(H_2)}, \quad (6)$$

Wanneer de hypotheses H_1 en H_2 een gelijke a priori kans hebben, dat wil zeggen $pr(H_1) = pr(H_2) = .5$, is de Bayes factor gelijk aan de a posteriori odds ten faveure van H_1. Het is echter goed mogelijk dat de twee hypotheses niet a priori even aannemelijk zijn. In het simpelste geval, wanneer de twee hypotheses enkelvoudige verdelingen zijn zonder vrije parameters (het zogeheten "simple versus simple" testen), is B_{12} de likelihood ratio. In andere gevallen, wanneer er onbekende parameters zijn onder één van de hypotheses, worden de dichtheden $pr(D|H_k)$ (k = 1, 2) verkregen door te integreren over de parameter ruimte, zodat in (4) de $pr(D|H_k)$ vervangen kan worden door

$$pr(D|H_k) = \int pr(D|\theta_k, H_k)\pi(\theta_k|H_k)d\theta_k \text{, met} \quad (7)$$

- θ_k de parameter onder H_k.
- $\pi(\theta_k|H_k)$ de a priori dichtheid van θ_k.
- $pr(D|\theta_k,H_k)$ de kansdichtheid van D gegeven de waarde van θ_k, ofwel de likelihoodfunctie van θ.
- dimensie d_k van vector θ_k.

Kortom, de Bayes factor is een weergave van het bewijs geleverd door de data ten voordele van de ene hypothese ten opzichte van de andere. Jeffreys, die in 1935 een paper schreef en later het boek *Theory of Probability*, en daarmee de grondlegger is van de Bayes factor, gaf de vuistregels in tabel 1. B_{10} betekent hierbij de Bayes factor ten faveure van H_1 en tegen H_0, in lijn met de notatie en gebruiken bij het testen van hypotheses.

Tabel 1: Jeffrey's schaal van bewijs voor Bayes factors

B_{10}	Bewijs tegen H_0
1 tot 3.2	Niet meer dan een kleine vermelding
3.2 tot 10	Positief
10 tot 100	Sterk
> 100	Beslissend

5.2 Het Schwarz Criterium

In (7) staat de berekening van pr(D|H_k). Het is mogelijk het gebruik van de a priori dichtheden $\pi(\theta_k|H_k)$ in (7) te vermijden door de logaritme van de Bayes factor te beanderen met

$$S = \log pr\left(D|\hat{\theta}_1, H_1\right) - \log pr\left(D|\hat{\theta}_2, H_2\right) - \frac{1}{2}\left(d_1 - d_2\right)\log(n) \qquad (8)$$

$\hat{\theta}_k$ is hierbij de maximum likelihood schatter onder H_k, d_k is de dimensie van θ_k en n is de sample grootte. In Kass & Raftery (1995) is S zonder afleiding gegeven. Als we echter de eerste twee termen samenvoegen dan zien we een bekendere uitdrukking

$$S = \log\left(\frac{pr\left(D \mid \hat{\theta}_1, H_1\right)}{pr\left(D \mid \hat{\theta}_2, H_2\right)}\right) - \frac{1}{2}\left(d_1 - d_2\right)\log(n) \qquad (9)$$

Als $n \to \infty$, voldoet S aan

$$\frac{S - \log B_{12}}{\log B_{12}} \to 0 \qquad (10)$$

en mag S beschouwd worden als een ruwe benadering van de logaritme van de Bayes factor. S wordt het Schwarz criterium genoemd; minus twee keer het Schwarz criterium het Bayesian Information Criterion (BIC).

De relatieve fout van exp (S) bij het benaderen van B_{12} is over het algemeen O(1). Zelfs voor grote datasets blijft er een fout. Tabel 1 geeft echter een indruk van deze fout en hieruit blijkt dat het Schwarz criterium een redelijke indicatie van het bewijs geeft. Het gebruik van BIC is aantrekkelijk vanwege het feit dat het gebruikt kan worden als een standaardprocedure, omdat de priors $\pi(\theta_k|H_k)$ niet bekend hoeven te zijn. Daarom wordt het vaak gebruikt als referentie in wetenschappelijke publicaties.

5.3 BIC vs AIC[2]

Bij het beoordelen van modellen wordt een evenwicht tussen de mate van *fit* en de *complexiteit* (het aantal vrije parameters) van het model berekend. Omdat toenemende complexiteit over het

[2] Deze sub-paragraaf is mede gebaseerd op Spiegelhalter (2002) en Wasserman (1997)

algemeen leidt tot een betere *fit*, maar ook tot *overfitting* kan leiden, worden modellen meestal beoordeeld door deze twee factoren tegen elkaar af te wegen. Akaike (1973) was een van de eersten die modellen op deze manier beoordeelde. Schwarz (1978) introduceerde de BIC en Spiegelhalter (2002) de DIC.

Akaike (1973) betoogde voor de keuze van het model dat gegeven een klasse van modellen voor een dataset de AIC minimaliseert.

$$AIC\ (M_j) = -2\ (\log \text{maximized likelihood(Data)}) + 2\ (\text{aantal parameters}) \qquad (11)$$

AIC is ontworpen om het volgende probleem op te lossen: stel dat, gegeven de huidige data en een verzameling mogelijke modellen, we een voorspellende verdeling willen voor een toekomstig datapunt. Als de voorspellende verdeling conditioneel is op de afzonderlijke modellen en hun geschatte parameters, dan kiest de AIC het model dat de beste benadering geeft op basis van de Kullback-Leibler afstand[3]. Dit wordt ook wel het "Akaike voorspel probleem"genoemd.

In dezelfde notatie als de AIC, is de BIC

$$BIC\ (M_j) = -2\ (\log \text{maximized likelihood(Data)}) + (\log N) * (\text{aantal parameters}) \qquad (12)$$

Om AIC en BIC te kunnen vergelijken moet gezegd worden dat ze voor verschillende doeleinden ontworpen zijn. AIC om het Akaike voorspel probleem op te lossen, terwijl BIC is ontworpen om het meest waarschijnlijke model te vinden, gegeven de data. Tevens blijkt uit de formules al dat de AIC de voorkeur geeft het aantal parameters te overschatten.

Een uitgebreidere discussie staat in Kass & Raftery (1995) en Wasserman (1997).

5.4 BIC-scoring in *x*-means

Pelleg & Moore (200X) gebruiken een aangepaste BIC-formule, in hun eigen paper genoteerd als

$$BIC\left(M_j\right) = \hat{l}_j(D) - \frac{p_j}{2} \cdot \log R \qquad (13)$$

$\hat{l}_j(D)$ is hierbij de log-likelihood van de data overeenstemmend met het j^{de} model, genomen bij het maximum likelihood punt en p_j is het aantal parameters in M_j. In plaats van BIC te minimaliseren wordt de aangepaste BIC uit Pelleg & Moore (200X) gemaximaliseerd. De notaties die in het paper gebruikt worden zijn:

μ_j = coördinaten van het j-de clustercentrum.

[3] De Kullback-Leibler afstand is gedefinieerd als $K(f,g) = \int f(y) \log [\ f(y) / g(y)\ ]\ dy$

(i) = de index van het clustercentrum dat het dichtst bij het i-de datapunt is.

D = de verzameling datapunten waarover geclusterd wordt.

D_i = de verzameling punten dat μ_j als dichtstbijzijnd clustercentrum heeft

R = |D|

R_i = $|D_i|$

M = het aantal dimensies

Voorbeeld: $\mu_{(i)}$ is het clustercentrum dat geassocieerd is met datapunt i.

De maximum likelihood schatting voor de variantie onder identieke sferische Gaussian aanname is

$$\hat{\sigma}^2 = \frac{1}{R-K}\sum_i \left(x_i - \mu_{(i)}\right)^2 . \qquad (14)$$

Deze formule staat in Pelleg & Moore zonder afleiding gegeven en het is ons niet duidelijk geworden waarom gedeeld wordt door 1/(R-K) in plaats van door R.

De kansverdeling van de punten is

$$\hat{P}(x_i) = \frac{R_{(i)}}{R} \cdot \frac{1}{\sqrt{2\pi}\hat{\sigma}^M} \exp\left(-\frac{1}{2\hat{\sigma}^2}\left\|x_i - \mu_{(i)}\right\|^2\right). \qquad (15)$$

Dit is een normale verdeling, waarbij rekening is gehouden met de dimensie van de punten. De loglikelihood van de data is

$$l(D) = \log\prod_i P(x_i) = \sum_i \left(\log\frac{1}{\sqrt{2\pi}\sigma^M} - \frac{1}{2\sigma^2}\left\|x_i - \mu_{(i)}\right\|^2 + \log\frac{R_{(i)}}{R}\right). \qquad (16)$$

Als we alleen focussen op de verzameling D_n van punten die behoren bij clustercentrum n, dus $1 \le n \le K$, en deze in de maximum likelihood schatting substitueren, evenals de schatting voor de variantie, krijgen we

$$\hat{l}(D_n) = -\frac{R_n}{2}\log(2\pi) - \frac{R_n \cdot M}{2}\log(\hat{\sigma}^2) - \frac{R_n - K}{2} + R_n \log R_n - R_n \log R . \qquad (17)$$

Het aantal vrije parameters p_j is de som van K – 1 clustercentra, M · K (aantal dimensie maal het aantal clustercentra) en 1 variantie schatting. Om deze formule uit te breiden naar alle clustercentra in plaats van deze ene, gebruiken we het feit dat de log-likelihood van de punten die tot alle clustercentra behoren de som is van de log-likelihood van de individuele clustercentra. R wordt dan vervangen door het aantal punten dat bij de clustercentra behoort.

6. KD-trees en blacklisting

Het *x*-means algoritme maakt gebruik van de *kd*-tree datastructuur en de blacklisting methode om het algoritme sneller uit te kunnen voeren. Dit is al eerder door Pelleg & Moore (2000) toegepast op *k*-means en deze paragraaf is daarop gebaseerd. Voor een verdere verdieping wordt verwezen naar Moore (1991).

6.1 De kd-tree

Om de datapunten op te slaan gebruikt *x*-means een mrkd-tree, ofwel een "multi-resolution kd-tree". Deze heeft de volgende eigenschappen:

- Het is een binaire boom.
- Elke node bevat informatie over alle punten die bevat zijn in een hyper-rechthoek *h*, die is opgeslagen als twee grensvectoren met lengte M: h_{min} en h_{max}. In de node staan tevens het aantal, centrum-massa en som van de Euclidische normen van alle punten in *h*. De kinderen van de node bevatten hyper-rechthoeken die zelf weer in *h* bevat zijn.
- Elke node heeft een split-dimensie *d* en een split waarde *v*. De kinderen *l* (respectievelijk *r*) representeren de hyper-rechthoeken h_l (h_r), beide binnen *h*, zodat alle punten in h_l (h_r) hun *d* coördinaat kleiner (groter) hebben dan *v*.
- De rootnode herbergt alle punten.
- De bladeren van de boom bevatten de datapunten.

Voor twee punten x, y definiëren we d(x,y) als de Euclidische afstand. Voor een punt x en een hyper-rechthoek h is closest(x,h) het punt in *h* dat het dichtst bij x ligt.

6.2 Blacklisting

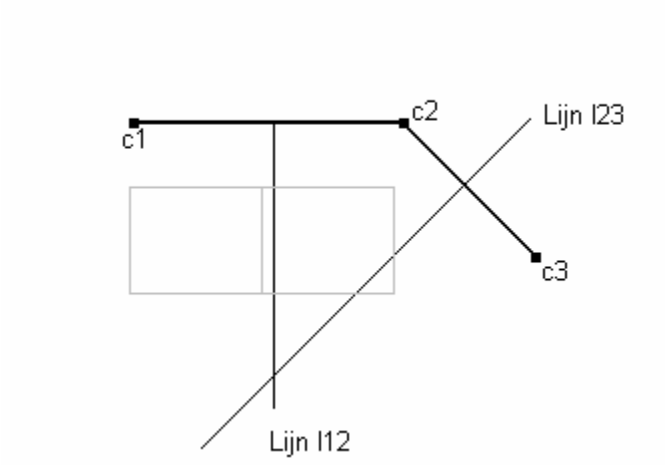
Het idee achter Blacklisting is om de clustercentra te identificeren die zeker *niet* eigenaar van de hyper-rechthoek *h* kunnen zijn. Als dit geldt voor een clustercentrum *c* hoeven we *c* niet meer te checken voor alle kinderen van *h*, vandaar de term blacklisting: we zetten de kinderen van h op een zwarte lijst.

We introduceren de volgende definities:

1. Gegeven een verzameling clustercentra C en een hyper-rechthoek *h*, definiëren we $\text{eigenaar}_c(h)$ als het clustercentrum c ϵ C, wanneer elk punt in *h* dichterbij c ligt dan bij elk ander clustercentrum c ϵ C, indien die bestaat.

2. Gegeven een hyper-rechthoek *h* en twee clustercentra c1 en c2, zodat d(c1,*h*) < d(c2,*h*) zeggen we dat c1 domineert over c2 met betrekking tot *h*, als elk punt in *h* dichterbij c1 is dan bij c2. (Om na te gaan of het ene clustercentrum een ander domineert, lopen we natuurlijk niet alle data na, maar alleen de hoekpunten van de rechthoek.)

Stel dat c1 een clustercentrum is op minimale afstand van *h*. c2 Is een ander clustercentrum, zodat d(c1,*h*) < d(c2,*h*). Als c1 over c2 domineert met betrekking tot *h*, zijn er twee mogelijkheden. De eerste is dat c1 = eigenaar (*h*). Dit is mooi, omdat dan geen berekeningen meer nodig zijn. De andere optie is dat we nog geen eigenaar voor deze node hebben, omdat er nog te weinig informatie is om hier iets over te mogen zeggen. Het blacklisting algoritme noteert dan dat c1 domineert over c2 met betrekking tot *h* en alle kinderen van *h* die nog onder *h* aan de boom hangen, en schrapt c2 van de lijst van mogelijke clustercentra voor *h* en al zijn nakomelingen. De lijst met potentiële eigenaren zal afnenem tot het 1 is. Het overgebleven clustercentrum wordt dan eigenaar van *h* verklaard. We hopen dat dit hoog in de boom gebeurd, zodat er tijd bespaard wordt. Als het algoritme toch tot op "blad"-niveau de boom in moet, kost het alleen meer tijd. Pelleg & Moore (2000) melden dat voor een standaard run met 30000 punten en 100 clustercentra het algoritme ongeveer 270000 berekeningen nodig is per iteratie. Dit, plus de "overhead", dient vergeleken te worden met de 3 miljoen (=30000 * 100) afstanden die het naïeve algoritme moet berekenen.



Figuur 2. Een blacklisting voorbeeld

In figuur 2 is een voorbeeld gegeven van blacklisting. Het grote grijs omkaderde vlak noemen we *h*. Dit is niet de *h* van de rootnode, omdat die alle punten van dataset D omvat, maar een *h* die is ontstaan na enkele iteraties. De twee lijnen *l12* en *l23* zijn de decision-lines, die in paragraaf 3 aan de orde zijn geweest, alle punten links van *l12* behoren bijvoorbeeld bij clustercentrum c1. Te zien is dat niet te zeggen is bij welk cluster de punten uit *h* horen. Door nu 1 stap verder in de boom te gaan en h te splitsen kunnen we zeggen dat c1 over c2 en c3 domineert wat betreft *hl*. Als we aannemen dat we 3-means doen, zijn we klaar voor de hyper-rechthoek *hl* en zijn kinderen. *hr* Zal nog 1 één of twee splitsingen nodig zijn voordat duidelijk is welk gedeelte van *hr* bij c1, c2 of c3 hoort.

7. Conclusies

Hoewel onze probleemstelling "*Is er een methode om in een berg data de onderliggende clusters te vinden, zonder dat van tevoren het aantal clusters bekend is?*" simpel onder woorden te brengen is, is het antwoord zeker niet triviaal.

Pelleg & Moore (200X) hebben een interessant idee om dit met behulp van een heuristiek op te lossen, zij breiden *k*-means uit tot *x*-means. Door op een slimme manier *k*-means te runnen en een benadering voor de score van het model, pr(Mj|D) toe te passen is het hun gelukt een cluster algoritme te ontwikkelen waarbij van tevoren het bereik van het aantal clusters *k* opgegeven wordt en het algoritme vervolgens op basis van de BIC-score de optimale *k* en de clusterverdeling geeft.

In dit paper zijn de verschillende dimensies van dat algoritme behandeld. Verreweg de meeste tijd ging zitten in het onder de knie krijgen van Bayes factors en de daaruit vloeiende BIC-scoring. Wat wel in het plan stond, maar waar ik niet aan toe ben gekomen is het uitzoeken van het Deviance Information Criterion (DIC) en om die aan het programma toe te voegen. Dat is wellicht nog interessant om te onderzoeken.

Kortom, na het bestuderen van de x-means algoritme denk ik dat het geschikt is om concurrentie te analyseren. Een minpunt is de random-initialisatie van de clustercentra, maar deze zouden bijvoorbeeld op economische gronden kunnen worden gekozen. X-means is mijns inziens duidelijk theoretisch onderbouwd, behoudens de schatting van de variantie in de BIC formule, en het is aan de economische wereld om deze vorm van wiskunde en informatica te accepteren.

Literatuur

Bikker, J.A. (2004), *Competition and Efficiency in a Unified European Banking Market*, Edward Elgar.

Kass, R.E. & Raftery, A.E. (1995), *Bayes Factors*, Journal of the American Statistical Association, Vol.90, No. 430, blz. 773-795.

Moore, A.W. (1991), *An introductory tutorial on kd-trees*. Extract from PhD Thesis: *Efficient Memory-based Learning for Robot Control*. Technical Report 209, Computer Laboratory, University of Cambridge.

Pelleg, D. & Moore, A. (2000). *Accelerating Exact k-means Algorithms with Geometric Reasoning*. Carnegie Mellon University, Pittsburgh, PA. (http://www.cs.cmu.edu/~dpelleg).

Pelleg, D. & Moore, A. (200X). *X-means: Extending K-means with Efficient Estimation of the Number of Clusters*. Carnegie Mellon University, Pittsburgh, PA. (http://www.cs.cmu.edu/~dpelleg).

Spiegelhalter, D. J., et al. (2002), *Bayesian measures of model complexity and fit*, Journal of the Royal Statistical Society Series B, blz. 583-639.

Stramaglia, S., et al (2004), *Statistical Physics and the Clustering Problem*, in: Wille, redactie, *New directions in Statistical Physics*, Springer-Verlag, blz. 253-272.

Wassermann L. (1997). *Bayesian Model Selection and Model Averaging*. Carnegie Mellon University, Pittsburgh, PA. (http://www.cs.cmu.edu/~dpelleg).

Wille, L. T. (redactie) (2004), *New Directions in Statistical Physics*, Springer-Verlag.

Witten, I. H. & Frank, E. (2000), *Datamining*, Morgan Kaufmann Publishers.