Market Basket Analysis

In dit hoofdstuk bespreken we *market basket* analyse. Dit betekent letterlijk “het analyseren van het boodschappenmandje”. De techniek ontleent zijn naam aan de veelvuldige toepassing van deze techniek op het koopgedrag van consumenten in de supermarkt. Maar zoals we zullen laten zien in ons voorbeeld, kan de techniek worden toegepast in alle gevallen waarin we op zoek zijn naar patronen, of verbanden tussen kenmerken van objecten of personen (aankopen in supermarkten; gedragskenmerken van criminelen; DNA-patronen bij bepaalde ziektes; enzovoorts).

De populariteit van de techniek heeft alles te maken met de enorme hoeveelheid gegevens die grote organisaties (supermarkten; online verkopers, zoals Amazon en Bol.com) hebben over personen (klanten; leden; enzovoorts) en hun transacties.

Traditioneel vertrouwen marketingspecialisten op hun intuïtieve kennis van het koopgedrag, soms ondersteund door informatie uit marktonderzoek. Belangrijke vragen zijn bijvoorbeeld: wat zijn de verwachte verkoopaantallen als de prijs voor een product met 5% wordt verlaagd? Of, voor supermarkten, wat is de optimale winkelindeling (volgorde van producten en schappen; plaats in het schap)?

Voorheen was het natuurlijk wel mogelijk hiermee te experimenteren, maar het frequent veranderen van bijvoorbeeld de looproute in de winkel zal gemakkelijk leiden tot verwarring en ergernis bij de klant. Tegenwoordig, met de opkomst van barcodes en scanningsystemen die gekoppeld zijn aan klantenkaarten, kunnen bedrijven veel beter in kaart brengen wie, wat, wanneer en in welke hoeveelheden koopt.

Online boekwinkels zoals Amazon en Bol.com hebben ten opzichte van traditionele boekhandels een groot concurrentievoordeel. Voor traditionele boekhandels zijn de meeste klanten min of meer anoniem. Maar online boekwinkels kennen het profiel van de klanten en hun koopgedrag. Zij hebben een beeld van het genre boeken dat hun klanten graag lezen, en kunnen gericht producten promoten, en boeken van andere auteurs aanbevelen (zie ter illustratie, figuur 1). De informatie “anderen bekeken ook” is een directe uitkomst van een *market basket* analyse!

Figuur 1. Market Basket Analysis bij Bol.com. *Bron: Bol.com.*

Het algoritme

Net als clusteranalyse via K-means clustering in hoofdstuk 2, is *market basket* analyse een voorbeeld van *unsupervised learning*. Dat betekent dat het niet nodig is om het algoritme eerst te trainen, en vervolgens te kijken of het algoritme goede voorspellingen oplevert in een test set van gegevens die niet zijn gebruikt in de training.

Waar we naar op zoek zijn is een set van regels die kenmerken met elkaar in verband brengen. Vervolgens kunnen we de vastgestelde verbanden tussen de kenmerken inzetten voor beleidsbeslissingen. In de setting van de supermarkten zijn de kenmerken de boodschappen (*items*) in het boodschappenmandje van de klant.

Items, item sets en verbindingsregels

Het verband tussen *items*, is het gezamenlijk voorkomen van *items* in het boodschappenmandje.

In het jargon van *market basket* analyse spreken we van *item sets*, en van verbindingsregels (*association rules*). Een *item set* is simpelweg een verzameling van een of meer *items*. We geven deze verzamelingen weer met accolades.

Een voorbeeld van een *item set* in het boodschappenmandje van een klant in de supermarkt, is de combinatie van brood en pindakaas, weergegeven als:

{brood, pindakaas}

Maar ook

{brood}

en

{brood, pindakaas, toiletreiniger}

zijn voorbeelden van *item sets*.

Het aantal mogelijke *item sets* neemt al snel enorme vormen aan.

Stel, een heel kleine winkel verkoopt 10 soorten producten (of *items*). Een klant kan elk *item* wel of niet kopen, en dat leidt tot 2^{10} (2 tot de macht 10) is 1,024 mogelijke *item sets*! Voor een iets grotere supermarkt met 100 verschillende *items* is het aantal mogelijkheden $1.27*10^{30}$ (meer dan een 1 met 30 nullen!). Het is in die situaties onmogelijk om alle mogelijke combinaties te evalueren. We gaan op zoek naar een slim en efficiënt algoritme om regels (*rules*) te genereren die *item sets* met elkaar verbinden.

Een voorbeeld van zo'n regel is:

{brood} → {pindakaas, hagelslag}

In deze regel wordt een verband gelegd tussen een *item* of *item set* aan de linkerkant van de pijl (aangeduid als LHS, of *left hand side*) en een *item* of *item set* aan de rechterkant van de pijl (RHS, of *right hand side*). De linkerkant stelt de voorwaarde, en de rechterkant is het gevolg. In woorden zegt deze regel dat de aanschaf van brood het kopen van pindakaas en hagelslag tot gevolg heeft.

In het *market basket* analyse algoritme zoeken we naar een slimme methode om "interessante" verbindingsregels tussen *items* op te sporen. Wat interessant is, hangt sterk af van het type toepassing. In een grote supermarkt met duizend producten en vele duizenden klanten en miljoenen transacties per jaar, kunnen bepaalde combinaties die op het geheel genomen een klein deel uitmaken van het geheel al belangrijke informatie opleveren. In medische toepassingen of kleinere data sets zullen andere criteria gelden.

Data

De naam van de techniek, *market basket analysis*, lijkt de toepassing ervan te beperken. Maar de principes van het analyseren van verbindingsregels, met de kernbegrippen **support, confidence** en **lift**, zijn breed inzetbaar, zoals u zult zien in het onderstaande voorbeeld.

Als voorbeeld van *market basket* analyse zullen we gebruikmaken van een kleine dataset met kenmerken van 10 personen die crimineel gedrag hebben vertoond.

> *Psychologen hebben deze personen geïnterviewd, en de informatie uit de gesprekken gecodeerd aan de hand van steekwoorden (zoals "drugs" als er sprake was van een drugsverslaving; "gescheiden", enzovoorts).*
>
> *De vraag is nu of deze kenmerken (items) aan elkaar gerelateerd zijn. Zijn, bijvoorbeeld, drugsgebruikers vaker gescheiden; of omgekeerd, gebruiken criminelen die gescheiden zijn vaker drugs?*

We kunnen de informatie uit de interviews vastleggen in een databestand. Zo'n databestand (in Excel) kan er als volgt uitzien:

	A	B	C	D	E
1	ouders gescheiden	winkeldiefstal			
2	drugs	gescheiden	geweldsdelict		
3	ouders gescheiden	baan kwijt	drugs	geweldsdelict	belastingschuld
4	adhd	ouders gescheiden			
5	drugs	gescheiden			
6	ouders gescheiden	belastingschuld	gescheiden		
7	drugs	baan kwijt			
8	baan kwijt	belastingschuld			
9	ouders gescheiden	gescheiden	drugs		
10	ouders gescheiden	geweldsdelict	gescheiden		

Figuur 2. Databestand in Excel. Bron: OGN.

In het eerste interview, bijvoorbeeld, bleek dat de geïnterviewde gescheiden ouders heeft, en zich ooit schuldig heeft gemaakt aan winkeldiefstal. In geen van de andere interviews komt winkeldiefstal voor. Een conclusie is dat winkeldiefstal altijd samengaat met gescheiden ouders (ook al is het bewijs voor deze conclusie, met slechts een geval van winkeldiefstal, erg dun). Omgekeerd zijn er diverse geïnterviewden met gescheiden ouders die zich niet aan winkeldiefstal schuldig hebben gemaakt.

Je ziet dat het zelfs in een kleine dataset niet gemakkelijk is de relaties te duiden, door het gebrek aan structuur.

In het databestand valt een aantal zaken op.

- Ten eerste is de structuur anders dan we gewend zijn. In de vorige hoofdstukken hadden de kolommen altijd een vaste betekenis. Elke kolom bevatte een variabele met een eenduidige betekenis (bijvoorbeeld leeftijd of geslacht).

  De kolommen in dit bestand zijn de "losse" opmerkingen die de psycholoog tijdens het gesprek heeft gemaakt. De opmerking drugs kan de eerste opmerking zijn (zoals voor persoon 2) of de derde opmerking (persoon 3). Het is in ons voorbeeld niet aan de orde, maar in principe kan de interviewer gedurende het gesprek twee of meer opmerkingen hetzelfde coderen (bijvoorbeeld "drugs" komt dan twee of meer keer voor).

- Ten tweede is het aantal gevulde kolommen verschillend per persoon. Bij persoon 3 staan 5 opmerkingen, en bij persoon 1 slechts 2 opmerkingen.

Deze manier van het opslaan van gegevens heeft grote voordelen als het gaat om de duizenden producten in supermarkten waarvan er in iedere transactie maar een zeer klein percentage wordt gekocht. Als iedere kolom een product (*item*) zou voorstellen, dan zou ieder record in het databestand vooral uit nullen of lege cellen bestaan! Een webwinkel zou een regel moeten aanmaken met evenveel kolommen als artikelen in het assortiment, ook al omvat elke transactie slechts één of enkele artikelen.

Ook in ons voorbeeld heeft deze manier van dataopslag voordelen: de interviewer kan de opmerkingen noteren gedurende het interview, zonder zich te bekommeren over de volgorde en over het aantal mogelijke opmerkingen. En de caissière in de supermarkt scant de producten ook in de "toevallige" volgorde waarmee ze op de band worden gelegd!

Zelftest

Probeer voor jezelf na te gaan of je een patroon ziet in de opmerkingen. Hoewel er maar 10 personen in het bestand zitten en het aantal opmerkingen maximaal 5 is, is het nog niet zo gemakkelijk patronen te detecteren! Een algoritme kan uitkomst bieden!

Het algoritme

Als we op zoek gaan naar "interessante" verbindingsregels in een enorme set van mogelijke regels is de uitdaging een efficiënt algoritme te ontwikkelen dat het aantal mogelijkheden zoveel mogelijk indikt. Zo'n algoritme kent twee stappen.

- In de eerste stap beperken we ons tot die *items* en *item sets* die een minimaal aantal voorkomen. In ons voorbeeld zouden we ons kunnen beperken tot die opmerkingen (*items*) en combinaties van opmerkingen die minimaal in 2 op de 10 gevallen worden genoemd.

  De opmerking "adhd" valt dan af, want die komt slechts een keer voor. Maar als de opmerking "adhd" maar een keer voorkomt, dan kunnen alle combinaties van andere opmerkingen met "adhd" ook niet 2 of meer keer voorkomen! Anders gezegd: met het uitsluiten van een *item* ("adhd") vervallen heel veel *item sets*. Voorbeelden van *item sets* die vervallen:

  {adhd; drugs}; **{adhd; gescheiden}**; **{adhd; drugs; gescheiden}**.

Het aantal voorkomens van *items* en *item sets* duiden we aan als **support**.

$$Support = \frac{Aantal\ keer\ dat\ een\ item\ set\ voorkomt}{Totaal\ aantal\ transacties}$$

De **support** voor de opmerking "adhd" is 1 op de 10 (0.10, of 10%). Onthoud dat een *item set* kan bestaan uit 1 of meer *items*. Een opmerking op zich is dus ook een *item set*, ook als bevat de set maar een *item*!

- Als we eenmaal *item sets* hebben bepaald met voldoende **support**, kunnen we doorgaan naar stap 2. Daar kijken we naar verbindingsregels tussen sets.

  Verbindingsregels beoordelen we aan de hand van hun **confidence** (vertrouwen). Dit is een lastig begrip dat vaak leidt tot verwarring. We zullen het toelichten aan de hand van ons voorbeeld.

  De formule voor **confidence** is:

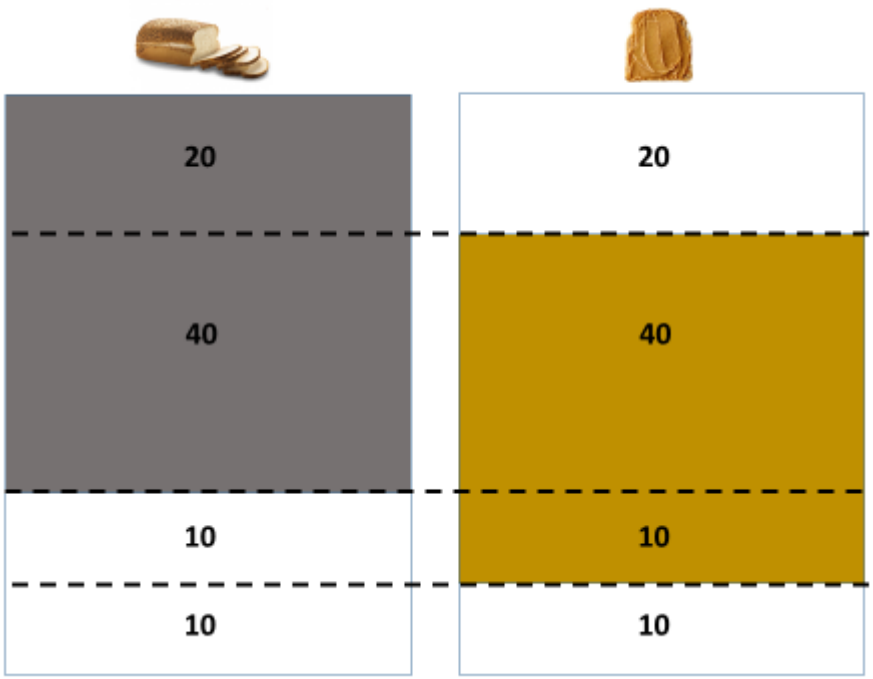
$$Confidence(X \rightarrow Y) = \frac{Support(X,Y)}{Support(X)}$$

  In woorden geeft deze formule de kans weer dat *item set* X (als voorwaarde) leidt tot *item set* Y (het gevolg). In het voorbeeld van de supermarkt: de regel dat het kopen van brood leidt tot het kopen van pindakaas, heeft een kans die gelijk is aan het aantal keren dat pindakaas én brood tegelijk worden gekocht, gedeeld door het aantal keren dat brood wordt gekocht.

  Met een cijfervoorbeeld: stel je voor dat een supermarkt 80 boodschappenmandjes analyseert. In 60 gevallen bevat het boodschappenmandje brood (het grijs gearceerde deel, in de linker kolom van figuur 3). In 50 gevallen bevat het boodschappenmandje pindakaas (het geel gearceerde gebied in de rechterkolom).

  De **support** voor brood bedraagt 60/80=0.75.

  De **support** voor pindakaas bedraagt 50/80=0.625. We kunnen ook de **support** voor het *item set* **{brood, pindakaas}** berekenen: 40/80 = 0.50.



*Figuur 3. Illustratie van het begrip **confidence.** Bron: OGN.*

We kunnen nu de **confidence** berekenen voor de verbindingsregels.

{brood} → {pindakaas}

en

{pindakaas} → {brood}

Voor de eerste verbindingsregel **{brood} → {pindakaas}** is de **confidence** te berekenen als:

$$Confidence(brood \rightarrow pindakaas) = \frac{0.50}{0.75} = 0.67$$

In 2 op de 3 gevallen (67%) leidt het aanschaffen van brood tot het aanschaffen van pindakaas.

Belangrijk is de "waarde" van deze informatie! De **confidence** van 67% voor deze regel, is alleen informatief als die afwijkt van de **support** voor pindakaas! Zonder informatie over het kopen van brood, schatten we de kans op het kopen van pindakaas in op 62.5%; maar als we weten dat er ook brood in het boodschappenmandje zit is de kans op pindakaas in het boodschappenmandje iets hoger (67%).

We hebben daarom naast **support** en **confidence** nog een derde maatstaf nodig, die we **lift** noemen. We komen daar later in het hoofdstuk op terug.

Confidence voor **{brood} → {pindakaas}** is niet hetzelfde als **confidence** voor **{pindakaas} → {brood}**!

$$Confidence(pindakaas \rightarrow brood) = \frac{0.50}{0.625} = 0.80$$

We kunnen de verschillen als volgt interpreteren.

- Als er pindakaas wordt gekocht dan is de kans heel groot (80%) dat ook brood wordt gekocht.
- Het komt vaker voor dat er "wel brood" wordt gekocht maar "geen pindakaas", dan "wel pindakaas" maar "geen brood" (niet iedereen houdt van pindakaas op zijn brood; pindakaas kan voor niet veel andere dingen worden gebruikt dan voor op het brood).

Laten we dit eens toepassen op ons voorbeeld. Als voorbeeld kunnen we kijken naar de relatie tussen **{ouders gescheiden}** en **{gescheiden}**. Het kan in incidentele gevallen natuurlijk wel voorkomen dat je ouders van elkaar scheiden nadat je zelf gescheiden bent, maar veel belangrijker in de context van verklaringen voor crimineel gedrag, zijn de gevolgen van gescheiden ouders op het leven van hun kinderen. Een scheiding kan gerelateerd zijn aan het verlies van een baan; het verlies van een baan kan leiden tot drugsgebruik en geldgebrek; enzovoorts.

De voor ons interessante relatie is dus:

{ouders gescheiden} → {gescheiden}

In de formule voor **confidence** moeten we berekenen wat de **support** is voor het *item* **{ouders gescheiden}** en voor het *item set* **{ouders gescheiden; gescheiden}**. De **support** voor **{gescheiden}** is in deze relatie niet van belang. Omdat het databestand klein is kunnen we het gemakkelijk berekenen, bijvoorbeeld in Excel.

					Ouders gescheiden?	Ouders gescheiden; gescheiden
ouders gescheiden	winkeldiefstal				1	0
drugs	gescheiden	geweldsdelict				x
ouders gescheiden	baan kwijt	drugs	geweldsdelict	belastingschuld	1	0
adhd	ouders gescheiden				1	0
drugs	gescheiden					x
ouders gescheiden	belastingschuld	gescheiden			1	1
drugs	baan kwijt					x
baan kwijt	belastingschuld					x
ouders gescheiden	gescheiden	drugs			1	1
ouders gescheiden	geweldsdelict	gescheiden			1	1
				Totaal	**6**	**3**

*Figuur 4. Illustratie van het begrip **confidence** in Excel.* *Bron: OGN.*

In de voorlaatste kolom hebben we met een 1 aangegeven of de opmerking "ouders gescheiden" van toepassing is op ieder van de 10 personen in ons onderzoek. Dat blijkt 6 keer het geval te zijn. Of, anders gezegd, de **support** voor **{ouders gescheiden}** is 6/10=0.60.

In de laatste kolom kijken we naar het *item paar* **{ouders gescheiden; gescheiden}**. We hebben een "x" ingevuld als de ouders niet gescheiden zijn. Immers als de ouders niet gescheiden zijn, dan is het *item paar* **{ouders gescheiden; gescheiden}** zeker niet aan de orde! We hebben met een "1" of 0"" aangegeven of de persoon wel of niet gescheiden is, *gegeven dat de ouders gescheiden zijn*. Dit komt voor in 3 van de 6 gevallen. De **confidence** van de relatie is dus 3/6=0.50.

Zelftest

In ons voorbeeldbestand bent u op zoek naar de relatie tussen "baan kwijt" en "drugs". Schrijf voor uzelf de verbindingsregel uit. Bereken vervolgens:

- *de **support** voor deze twee items afzonderlijk*
- *de **support** voor het itempaar, en*
- *de **confidence** die bij de verbindingsregel hoort!*

Stap 1: inlezen van de data

In een groot bestand met veel personen en veel *items* is het ondoenlijk deze berekeningen handmatig te maken. We gaan daarom gebruik maken van **R**.

Een krachtig *package* voor *market basket* analyse is **arules**.

Als we **arules** voor de eerste keer gaan gebruiken moeten we het eerst installeren op onze computer met **`install.packages()`**. Als we de functies van **arules** willen gebruiken in onze **R**-sessie roepen we het *package* aan met het **`library()`** commando.

```
install.packages("arules")
library(arules)
```

Noot:

In de console – ten tijde van het schrijven van dit script - verscheen een waarschuwing dat het package is geschreven in versie 3.2.5 van R.

```
Warning message:
package 'arules' was built under R version 3.2.5
```

*Packages worden regelmatig aangepast (bug fixes; nieuwe functies). Het is alleen daarom al van belang regelmatig de nieuwste versie van **R** te installeren: in de nieuwste versie van **R** werken packages gebouwd in oudere versies wel, maar het omgekeerde is niet altijd het geval. We noemen dat "backward compatibility" (achterwaartse compatibiliteit). **R** geeft "slechts" een waarschuwing omdat het best kan zijn dat de functies in het package wel werken onder de oude **R**-versie. In dit hoofdstuk maakten we gebruik van **R**-versie 3.2.3, en het package lijkt prima te werken.*

In de vorige hoofstukken lazen we de data in als een rechthoekig bestand, waarin de rijen en kolommen een vaste betekenis hadden: elke rij is een object (bijvoorbeeld een persoon of een transactie), en elke kolom is een variabele. We kunnen dat ook doen met onze data. De in Excel opgeslagen data uit figuur 2, zijn bewaard als een *csv*-bestand en vervolgens ingelezen in **R.** Afgezien van ontbrekende data voor sommige variabelen personen (*missing values*), is de rechthoek goed gevuld.

Als we het *comma separated values* bestand **crimi.txt** inlezen, met **`read.csv()`**, dan zien we het volgende resultaat.

```
> crimitest <- read.csv("ML4_OEFEN1.txt", header=F)
> crimitest
                 V1                V2            V3            V4               V5
1  ouders gescheiden    winkeldiefstal
2             drugs        gescheiden geweldsdelict
3  ouders gescheiden        baan kwijt         drugs geweldsdelict belastingschuld
4              adhd ouders gescheiden
5             drugs        gescheiden
6  ouders gescheiden   belastingschuld    gescheiden
7             drugs        baan kwijt
8        baan kwijt   belastingschuld
9  ouders gescheiden        gescheiden         drugs
10 ouders gescheiden     geweldsdelict    gescheiden
```

Het bestand bevat geen namen voor de variabelen in de eerste rij, en daarom hebben we opgegeven dat er geen header in het bestand zit (**`header=F`**; **`F`** is een korte schrijfwijze voor **`False`**). **R** genereert zelf namen voor de variabelen (**V1** t/m **V5**).

Het aantal variabelen (5) is bepaald door het maximale aantal opmerkingen voor een persoon, in dit geval persoon 3. Maar je ziet dat een bepaalde opmerking, bijvoorbeeld "drugs", kan voorkomen als waarde van **V1** (personen 2, 5 en 7), of als **V3** (personen 3 en 9). Dat is niet handig voor het analyseren, omdat we voor elke mogelijke opmerking willen weten of die wel of niet voorkomt voor een bepaalde persoon!

Stap 2: het prepareren van de data

Het **arules** package bevat een uiterst handige functie om dergelijke data in te lezen op een efficiënte manier: de **`read.transactions()`** functie.

De data worden nu weggeschreven als een zogenoemde *sparse matrix*, een rechthoek die spaarzaam omgaat met de ruimte. Het databestand heeft niet de vorm van een dataframe, en moet dan ook op een andere manier worden benaderd. Het voorbeeld zal het duidelijk maken. Omdat we het hierboven aangemaakte rechthoekige databestand **crimitest** niet gaan gebruiken, verwijderen we het met **`rm()`**.

```
rm(crimitest)

crimi <- read.transactions("ML4_OEFEN1.txt",sep=",")
summary(crimi)
```

```
> summary(crimi)
transactions as itemMatrix in sparse format with
 10 rows (elements/itemsets/transactions) and
 8 columns (items) and a density of 0.3375

most frequent items:
ouders gescheiden             drugs        gescheiden        baan kwijt   belastingschuld           (Other)
                6                 5                 5                 3                 3                 5

element (itemset/transaction) length distribution:
sizes
2 3 5
5 4 1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0     2.0     2.5     2.7     3.0     5.0

includes extended item information - examples:
           labels
1            adhd
2      baan kwijt
3 belastingschuld
```

*Figuur 5. Illustratie van de **summary()** functie in **R.** Bron: OGN.*

Je ziet weer dat de **`summary()`** functie in **R** een slimme functie is. Je kunt de functie toepassen op een *dataframe*, zoals in eerdere hoofdstukken. Maar je kan dezelfde functie ook gebruiken om het object **crimi** dat geen *dataframe* is, samen te vatten! Afhankelijk van het soort object (in **R**: de *object class*) geeft **`summary()`** een andere uitkomst.

Laten we het resultaat stap voor stap bekijken.

- In de eerste regel staat dat het inderdaad gaat om een matrix in een spaarzaam (*sparse*) format. Spaarzaam betekent hier dat elke regel alleen de wel gemaakte opmerkingen bevat (de "eentjes" in het bestand; en niet de "nullen" voor alle niet gemaakte opmerkingen).
- De matrix heeft 10 rijen, en 8 kolommen.

  De 10 rijen zijn de 10 personen met crimineel gedrag waarover de psychologen opmerkingen hebben gemaakt.

  De 8 kolommen representeren alle mogelijke opmerkingen (de *items*!) die zijn gemaakt. Je kan voor jezelf, in dit kleine voorbeeldbestandje gemakkelijk nagaan dat er inderdaad 8 verschillende opmerkingen zijn gemaakt!

  1. ADHD
  2. Baan kwijt
  3. Belastingschuld
  4. Drugs
  5. Gescheiden
  6. Geweldsdelict
  7. Ouders gescheiden
  8. Winkeldiefstal
- De 5 meestgemaakte opmerkingen zijn weergegeven, en ook hoe vaak ze voorkomen. De opmerking "ouders gescheiden" komt (zoals we hierboven al hebben berekend) 6 keer voor. De **support** kan worden berekend als 6, gedeeld door het aantal rijen (10) is 0.60.
- Alle andere opmerkingen (3 stuks) zijn samengevoegd onder "other". Ga voor jezelf na welke drie opmerkingen dit zijn!
- Een interessante statistiek – ook voor de latere analyses – is de dichtheid (*density*) van onze matrix. Als we de spaarzame matrix zien als een rechthoek van 10 personen maal 8 opmerkingen gevuld met nullen en enen, hoeveel van de 10*8=80 cellen van de matrix bevat dan een 1? We kunnen dat zelf uitrekenen in ons eenvoudige voorbeeld:

					Aantal?
ouders gescheiden	winkeldiefstal				2
drugs	gescheiden	geweldsdelict			3
ouders gescheiden	baan kwijt	drugs	geweldsdelict	belastingschuld	5
adhd	ouders gescheiden				2
drugs	gescheiden				2
ouders gescheiden	belastingschuld	gescheiden			3
drugs	baan kwijt				2
baan kwijt	belastingschuld				2
ouders gescheiden	gescheiden	drugs			3
ouders gescheiden	geweldsdelict	gescheiden			3
				Totaal	**27**

Figuur 6. Illustratie van de dichtheid. *Bron: Excel/OGN.*

In totaal zijn er 27 opmerkingen gemaakt. In een matrix met 80 cellen betekent dat een dichtheid van 27/80=0.3375. De output geeft dat weer.

De *density* in de boodschappenmandjes bij een supermarkt zal natuurlijk erg klein zijn!

- De volgende regel van de output geeft de omvang weer van de boodschappenmandjes, of in ons voorbeeld, de verdeling van het aantal opmerkingen. In de meeste gevallen (5 personen) is het aantal opmerkingen beperkt tot twee. In vier gevallen worden drie opmerkingen gemaakt, en in een geval vijf opmerkingen.
- Die verdeling wordt nog eens statistisch weergegeven, met onder andere het minimum (2), het gemiddelde (27/10=2.7) en het maximum (5).
- Ten slotte wordt er nog een aantal voorbeelden gegeven van de aan de kolommen van de matrix toegekende labels. De labels zijn in feite de opmerkingen uit ons bestand, en worden in alfabetische volgorde gezet: kolom 1 uit de matrix, geeft een vector van nullen en enen voor de opmerking “adhd”. We zullen dergelijke matrixen echter zelden gebruiken (voor een supermarkt met veel *items* in het assortiment, is zo’n matrix ook veel te groot).

Je kunt je data (de 10 regels van se *sparse matrix*) bekijken met het **`inspect()`** commando. Om de eerste drie rijen te laten zien gebruiken we:

```
> inspect(crimi[1:3])
    items
[1] {ouders gescheiden,winkeldiefstal}
[2] {drugs,gescheiden,geweldsdelict}
[3] {baan kwijt,belastingschuld,drugs,geweldsdelict,ouders gescheiden}
```

*Figuur .7. Illustratie van het **inspect()** commando.* *Bron: OGN.*

Vergelijk dit met de data in het Excelbestand!

Stap 3: Het verkennen en visualiseren van de data

We kunnen ook een tabel maken van de relatieve frequentie van de verschillende *items*, met de functie **`itemFrequency()`**.

In het *market basket analysis* jargon, geeft deze functie de **support** weer van de diverse *items*. Zo komt “adhd” in een van de 10 rijen voor, en is de **support** dus 1/10=0.1 (of 10%). De **support** voor drugs is 0.5, enzovoorts.

```
> itemFrequency(crimi)
             adhd        baan kwijt   belastingschuld             drugs        gescheiden
              0.1               0.3               0.3               0.5               0.5
    geweldsdelict ouders gescheiden    winkeldiefstal
              0.3               0.6               0.1
```

Figuur 8. Illustratie van support. *Bron: OGN.*

Je ziet dat de *items* gerangschikt zijn in alfabetische volgorde. Je kunt je voorstellen dat dit niet handig is als we heel erg veel *items* (producten; opmerkingen; enzovoorts) in ons bestand hebben. De output wordt dan erg omvangrijk. Je kunt de output wel beperken tot, zeg, de eerste 3 *items*, maar dat zijn waarschijnlijk niet de belangrijkste (want meest genoemde) *items*!

De output van **`itemFrequency()`** is een vector die we kunnen sorteren. Omdat we geïnteresseerd zijn in de *items* met de hoogste **support**, sorteren we aflopend met de optie **`decreasing=TRUE`** (de *default* is oplopend sorteren, dus **`decreasing=FALSE`**). In het geval van een groot aantal *items* is het raadzaam alleen de items met de hoogste **support** weer te geven.

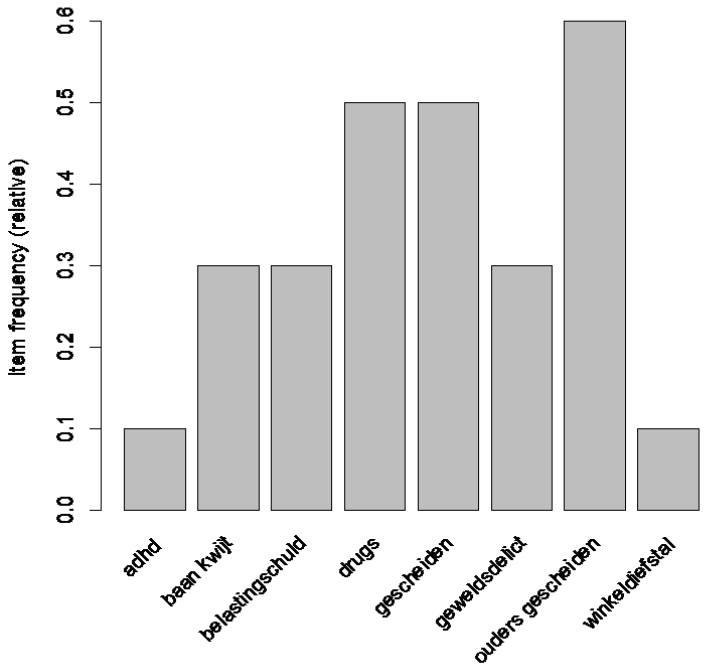
Om de commando's kort en overzichtelijke te houden maken we hieronder eerst een gesorteerde vector aan (**xs**). In het eerste commando drukken we de vector direct af door hem tussen haakjes "( )" te plaatsen. In het tweede commando drukken we alleen de eerste drie *items* af.

```
> (xs <- sort(itemFrequency(crimi), decreasing=TRUE))
ouders gescheiden             drugs        gescheiden        baan kwijt   belastingschuld
              0.6               0.5               0.5               0.3               0.3
    geweldsdelict              adhd    winkeldiefstal
              0.3               0.1               0.1
> xs[1:3]
ouders gescheiden             drugs        gescheiden
              0.6               0.5               0.5
```

Figuur 9. Illustratie van gesorteerde vectoren. *Bron: OGN.*

Veel handiger is de functie **`itemFrequencyPlot()`** omdat we daarin direct kunnen aangeven wat de minimale **support** moet zijn voor het weergeven van *items*, of de **`topN`** (bijvoorbeeld de top-drie) *items* met de hoogste **support**.

Omdat we maar 8 *items* hebben in ons eenvoudige voorbeeld, is een grafische weergave goed mogelijk (zie figuur 4).

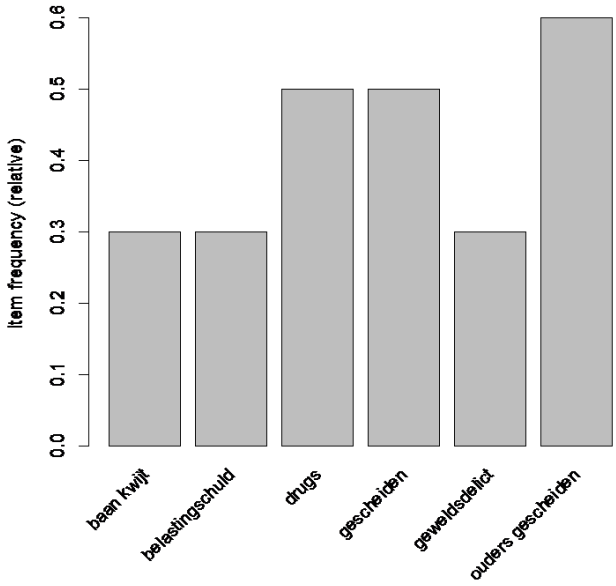


*Figuur 10. Grafische weergave van items met hun **support.** Bron: OGN.*

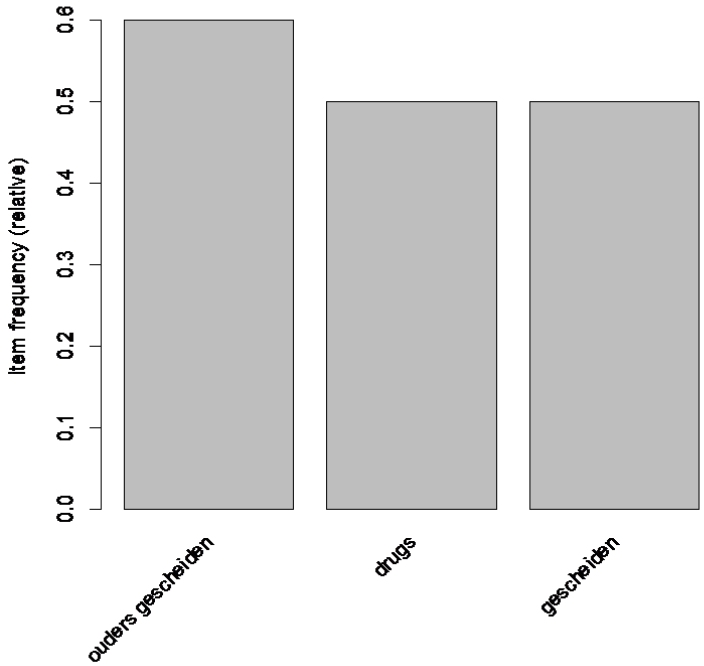
Voor het weergeven van de *items* met een **support** groter of gelijk aan 0.30, of voor de drie *items* met de hoogste **support**, gebruiken we de volgende opties. De optie **`topN`** is de meest gebruikte omdat (i) we het aantal *items* kunnen controleren, en (ii) de *items* worden geselecteerd op aflopende **support**.

```
itemFrequencyPlot(crimi, support=.3)
itemFrequencyPlot(crimi, topN=3)
```

De grafieken zien er als volgt uit (zien figuren 5 en 6).



*Figuur 11. Items met **support** ≥ .30.* *Bron: OGN.*

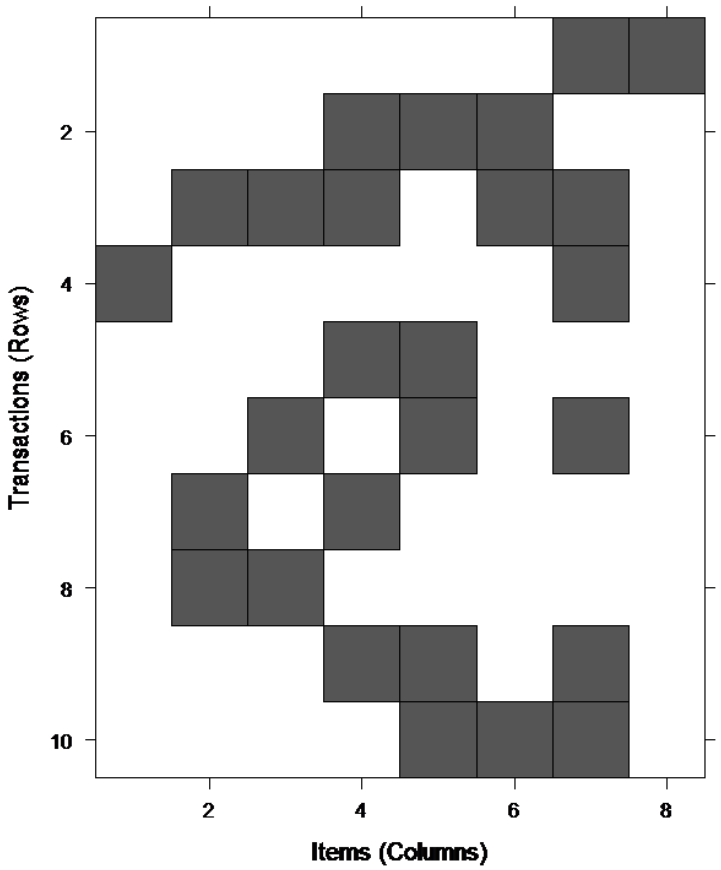


*Figuur 12. De 3 items met de hoogste **support.*** *Bron: OGN.*

De patronen in de data kunnen inzichtelijk worden gemaakt met de **`image()`** functie. Deze functie geeft – toegepast op boodschappenmandjes - in een matrix diagram (een grafiek met blokjes), weer welke consumenten welke producten kopen. Dit heeft eigenlijk alleen zin als het aantal rijen (in ons geval, criminelen) en het aantal *items* (opmerkingen over deze criminelen) klein is. Bij grote databestanden geeft de **`image()`** functie een enorme brei aan puntjes in een rechthoek die maar moeilijk te interpreteren is.

In ons voorbeeld:

```
image(crimi)          # alle data
image(crimi[1:3,])    # alleen de eerste 3 cases (rijen)
image(crimi[,6:8])    # items 6 t/m 8
```



Figuur 13. Matrixdiagram van items en cases. *Bron: OGN.*

Probeer zelf aan de hand van figuur 13 patronen te ontdekken. Welke *items* komen vaak voor? En welke *items* komen vaak voor in combinatie met andere *items*?

Stap 4: Het trainen van het model

Nu we de data hebben ingelezen en verkend, zijn we toe aan de volgende stap: het analyseren van de verbindingen tussen de *items*. In de **`image()`** functie in stap 3 zijn we daarmee al begonnen, maar het zal duidelijk zijn dat die grafiek niet gemakkelijk te interpreteren is – zeker niet als we heel veel data hebben.

Het efficiënt opsporen van relaties (of *associations*) in het bestand doen we met de **`apriori()`** functie. Deze functie gebruikt als argumenten de *sparse matrix*; de minimale **support**; de minimale **confidence**; en het minimale aantal *items* (*minlen*) in een verbindingsregel (of *association rule*).

Door het aangeven van een minimale **support** kunnen we *item sets* uitfilteren die weinig voorkomen. Hetzelfde geldt voor de minimale **confidence**: we zijn geïnteresseerd in de relatie tussen *items*, en in de kans dat de aanwezigheid in een *item* de aanwezigheid van andere *items* voorspelt.

Een aantal voorbeelden:

```
cr1 <- apriori(crimi)
inspect(cr1)

cr2 <- apriori(crimi, parameter = list(support=.1, confidence=.8, minlen=1))
inspect(cr2)

cr3 <- apriori(crimi, parameter = list(support=.3, confidence=.5))
inspect(cr3)

cr4 <- apriori(crimi, parameter = list(support=.3, confidence=.4, minlen=2))
inspect(cr4)

cr5 <- apriori(crimi, parameter = list(support=.2, confidence=.3, minlen=2))
inspect(cr5)
```

De eerste specificatie gebruikt de *default* waardes en resulteert hier in een groot aantal regels. Als de optionele parameters worden weggelaten, dan zijn de *default* waardes voor **support**, **confidence**, en *minlen* .1; .8 en 1. De verbindingsregels worden geschreven in een nieuw object (**cr1**).

Het is meestal niet verstandig de *default* waardes te gebruiken. De optimale waarden verschillen van geval tot geval. In ons geval gebruiken we voor **support** een vrij hoge waarde, van bijvoorbeeld 0.2 of 0.3 omdat een *item* of *itempaar* toch wel minimaal 2 op de 10 keer moet voorkomen. In een supermarkt met duizenden items zal de lat waarschijnlijk veel lager liggen.

- Sowieso willen we alleen relaties tussen *items* zien, en dan moeten we minimaal 2 *items* hebben (*minlen*=2).
- We willen een behapbare set van veelzeggende relaties overhouden.

  Als we de drempels te laag zetten dan krijgen we heel veel regels die moeilijk te interpreteren zijn. Maken we de drempels erg hoog dan is het aantal regels klein, of zelfs nul. Zeker in grote databestanden zullen we dus met *trial and error* (het gebruiken van verschillende parameterwaardes) moeten kijken wat het beste resultaat geeft.

  Je kunt dit vergelijken met het gebruik van Google. Als je op zoek bent naar informatie over de Zwarte-Pietdiscussie, en vooral in je eigen woonplaats, dan levert de zoekstring "Piet" meer dan 30 miljoen hits op (met veel hits over "piet" die niet in combinatie met "zwarte" voorkomen. Met "zwarte piet" is het aantal hits nog steeds 1.7 miljoen. De zoekstring "zwarte piet tietsjerkeradeel" levert maar 6 hits op (maar nu mis je misschien relevante links over de Zwarte-Pietdiscussie!

Laten we maar kijken wat deze functies voor resultaten geven, De volgende twee (sets) van commando's doen hetzelfde. In het eerste geval we geen lijst van parameters op, en in het tweede geval geven we expliciet de lijst met *default* waardes. Run beide sets van commando's en verifieer dat het resultaat inderdaad hetzelfde is!

De **`inspect()`** functie met als argument het object met *association rules* geeft een overzicht van de regels. De regels bestaan uit een linkerkant (LHS, voor left hand side) en een rechterkant (RHS, voor right hand side). De linkerkant "leidt tot", of voorspelt de rechterkant. De regels worden gevolgd door drie kenmerken: ***support***, ***confidence***, en ***lift***.

```
cr1 <- apriori(crimi)
inspect(cr1)

cr2 <- apriori(crimi, parameter = list(support=.1, confidence=.8,
minlen=1))
inspect(cr2)
```

```
> inspect(cr2)
     lhs                      rhs                  support confidence     lift
[1]  {winkeldiefstal}      => {ouders gescheiden}      0.1          1 1.666667
[2]  {adhd}                => {ouders gescheiden}      0.1          1 1.666667
[3]  {baan kwijt,
      geweldsdelict}       => {belastingschuld}        0.1          1 3.333333
```

[output weggelaten]

```
[39] {baan kwijt,
      drugs,
      geweldsdelict,
      ouders gescheiden} => {belastingschuld}        0.1          1 3.333333
[40] {belastingschuld,
      drugs,
      geweldsdelict,
      ouders gescheiden} => {baan kwijt}             0.1          1 3.333333
```

Deze informatie is compact, en waardevol! Maar je moet wel goed weten hoe de informatie te interpreteren.

Laten we de eerste regel nemen. De termen “lhs” en “rhs” geven de *left hand side* (linkerzijde) en *right hand side* (rechterzijde) van de regel aan; de voorwaarde aan de linkerkant leidt tot de rechterkant. Echter, het algoritme is louter gebaseerd op de uitkomsten van **support** en **confidence**, en of er daadwerkelijk een oorzakelijk verband is, is aan de analist!

{winkeldiefstal} → {ouders gescheiden}

- Laten we voorop stellen dat deze relatie “theoretisch” niet erg logisch is. Waarom immers zou een winkeldiefstal door een persoon, leiden tot het scheiden van de ouders?

  De output heeft zes kolommen. De eerste kolom geeft het volgnummer van de verbindingsregel; de tweede en derde kolom geven de **LHS** en de **RHS** van de verbindingsregel. Dan volgen in de kolommen 4 t/m 6 de belangrijkste statistieken: **support**, **confidence**, en **lift**.

  De **support** is die van het *itempaar*. De combinatie **{winkeldiefstal; ouders gescheiden}** komt een keer voor, in de 10 gevallen, en is dus 0.10. Dit is het getal in kolom 4.

  Voor het bereken van de **confidence** delen we de **support** van het *itempaar* (0.10) door de **support** voor de voorwaarde (de LHS). De **support** voor **{winkeldiefstal}** is ook 0.10, want **{winkeldiefstal}** komt maar een keer voor. De **confidence** is dus 0.10/0.10 = 1. Dit is weergegeven in kolom 5.

  Zoals we al eerder hebben aangegeven hebben we nog een derde maatstaf nodig die de informatiewaarde aangeeft.

  De redenering is als volgt. Als de kans op gescheiden ouders vaker voorkomt met de voorinformatie over winkeldiefstal, dan is er sprake van (mogelijk) nuttige informatie. In ons voorbeeld gaat winkeldiefstal altijd gepaard met gescheiden ouders (ook al hebben we dan maar een geval): de kans is dus 100%. Zonder de voorinformatie over winkeldiefstal, is de kans op gescheiden ouders 6/10 = 60% (de **support** voor **{ouders gescheiden}**). De verhouding 100% ten opzichte van 60% noemen we **lift**: 100%/60% = 1.67.

In een formule:

$$Lift(X \rightarrow Y) = \frac{Confidence(X \rightarrow Y)}{Support(Y)}$$

Merk op dat *items* met een **support** van 100% nooit informatief kunnen zijn! Dit geldt zowel voor (combinaties van) *items* aan de linkerkant (LHS) als de rechterkant (RHS)!

- Als de LHS (de voorwaarde; hier **{winkeldiefstal}**) in alle gevallen geldt, dan is de **confidence** gelijk aan de **support** voor *item* **{ouders gescheiden}**. De **lift** is dan gelijk aan 1.

- Ook als de RHS altijd opgaat, dan is de **lift** gelijk aan de **confidence** voor het *itempaar* gelijk aan 1

In ons voorbeeld **{brood}→{pindakaas}**.

- Als 100% van alle klanten brood koopt en 60% koopt pindakaas, dan is de voorinformatie over brood kopen niet informatief. Het percentage klanten dat brood en pindakaas koopt, is precies hetzelfde (60%) als het percentage klanten dat pindakaas koopt. In de formules: de **confidence** is gelijk aan 60%/100% = 0.60. De **lift** is gelijk aan de **confidence** (60%) gedeeld door de **support** voor pindakaas (60%): 60%/60% = 1.

- Als alle klanten pindakaas kopen en 40% koopt brood, dan is het percentage klanten dat brood en pindakaas koopt, gelijk aan de **support** voor brood (40%). De **confidence** is 40% gedeeld door de **support** voor brood (40%): 40%/40% = 1. De **lift** is dan **confidence** (1) gedeeld door de **support** voor pindakaas (1): 1/1 = 1.

Samenvattend, is de **lift** dus een zeer belangrijke maatstaf! Waar de bruikbare waardes voor **support** nog verschillen per situatie, en **confidence** niet altijd waardevolle informatie geeft, is de **lift** altijd informatief.

Tegen die achtergrond kunnen we de output interpreteren. Nuttige functies zijn, naast de **`summary()`** functie, vooral ook de **`inspect()`** functie. In die laatste functie kunnen we ook de regels sorteren op **lift**.

In de laatste onderstaande commando's schrijven we de op **lift** gesorteerde regels naar een matrix met de naam **short**. Vervolgens drukken we de rijen van de matrix met **lift**>1.2 af in de console.

```
> cr5 <- apriori(crimi, parameter = list(support=.2, confidence=.3, minlen=2))

[output weggelaten]

> inspect(cr5)
     lhs                    rhs                 support confidence lift
[1]  {baan kwijt}        => {belastingschuld}   0.2     0.6666667  2.2222222
[2]  {belastingschuld}   => {baan kwijt}        0.2     0.6666667  2.2222222
[3]  {baan kwijt}        => {drugs}             0.2     0.6666667  1.3333333

[output weggelaten]

[16] {ouders gescheiden} => {gescheiden}        0.3     0.5000000  1.0000000
[17] {drugs}             => {ouders gescheiden} 0.2     0.4000000  0.6666667
[18] {ouders gescheiden} => {drugs}             0.2     0.3333333  0.6666667

> summary(cr5)
set of 18 rules
```

```
rule length distribution (lhs + rhs):sizes
 2
18

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      2       2       2       2       2       2

summary of quality measures:
    support         confidence          lift
 Min.   :0.2000   Min.   :0.3333   Min.   :0.6667
 1st Qu.:0.2000   1st Qu.:0.4000   1st Qu.:1.1111
 Median :0.2000   Median :0.6000   Median :1.2000
 Mean   :0.2222   Mean   :0.5315   Mean   :1.2568
 3rd Qu.:0.2000   3rd Qu.:0.6667   3rd Qu.:1.3333
 Max.   :0.3000   Max.   :0.6667   Max.   :2.2222

mining info:
  data ntransactions support confidence
 crimi            10     0.2        0.3
```

```
> inspect(sort(cr5, by="lift"))
     lhs                    rhs                  support confidence lift
[1]  {baan kwijt}        => {belastingschuld}    0.2     0.6666667  2.2222222
[2]  {belastingschuld}   => {baan kwijt}        0.2     0.6666667  2.2222222
[3]  {baan kwijt}        => {drugs}             0.2     0.6666667  1.3333333
```

[output weggelaten]

```
[16] {ouders gescheiden} => {gescheiden}        0.3     0.5000000  1.0000000
[17] {drugs}             => {ouders gescheiden} 0.2     0.4000000  0.6666667
[18] {ouders gescheiden} => {drugs}             0.2     0.3333333  0.6666667
```

```
> short <- inspect(sort(cr5, by="lift"))
```

[output weggelaten]

```
> short[short$lift>1.2,]
                  lhs                    rhs support confidence     lift
[1]       {baan kwijt} => {belastingschuld}     0.2  0.6666667 2.222222
[2] {belastingschuld} =>       {baan kwijt}     0.2  0.6666667 2.222222
[3]       {baan kwijt} =>            {drugs}     0.2  0.6666667 1.333333
[4]            {drugs} =>       {baan kwijt}     0.2  0.4000000 1.333333
[5]    {geweldsdelict} =>       {gescheiden}     0.2  0.6666667 1.333333
[6]       {gescheiden} =>   {geweldsdelict}     0.2  0.4000000 1.333333
[7]    {geweldsdelict} =>            {drugs}     0.2  0.6666667 1.333333
[8]            {drugs} =>   {geweldsdelict}     0.2  0.4000000 1.333333
```

Natuurlijk is het belangrijk de verbindingsregels juist te interpreteren. Er zijn twee regels met een vrij grote **lift**, hoger dan 2. Beide hebben betrekking op de items **{baan kwijt}** en **{belastingschuld**}. Het is meer dan twee keer zo waarschijnlijk, voor de 10 personen in ons databestand, om een belastingschuld te hebben wetend dat ze hun baan zijn kwijt geraakt, dan voor de groep als geheel. Het omgekeerde geldt ook: het is meer dan twee keer zo waarschijnlijk je baan kwijt te zijn wetend dat er sprake is van een belastingschuld, dan voor de groep als geheel.

> *Noot:*
>
> *Dat de **lift** hetzelfde is voor de twee verbindingsregels is geen toeval. Het is eenvoudig aan te tonen dat **lift(x→y) = lift(y→x)**. Voor diegenen die wiskundig zijn ingesteld, probeer dit zelf aan te tonen door de formules uit te schrijven! De oplossing staat in bijlage 2.*

```
> short <- inspect(sort(cr5, by="lift"))
     lhs                     rhs                    support confidence lift
[1]  {baan kwijt}         => {belastingschuld}      0.2     0.6666667  2.2222222
[2]  {belastingschuld}    => {baan kwijt}           0.2     0.6666667  2.2222222
[3]  {baan kwijt}         => {drugs}                0.2     0.6666667  1.3333333
[4]  {drugs}              => {baan kwijt}           0.2     0.4000000  1.3333333
[5]  {geweldsdelict}      => {gescheiden}           0.2     0.6666667  1.3333333
[6]  {gescheiden}         => {geweldsdelict}        0.2     0.4000000  1.3333333
[7]  {geweldsdelict}      => {drugs}                0.2     0.6666667  1.3333333
[8]  {drugs}              => {geweldsdelict}        0.2     0.4000000  1.3333333
[9]  {gescheiden}         => {drugs}                0.3     0.6000000  1.2000000
[10] {drugs}              => {gescheiden}           0.3     0.6000000  1.2000000
[11] {belastingschuld}    => {ouders gescheiden}    0.2     0.6666667  1.1111111
[12] {ouders gescheiden}  => {belastingschuld}      0.2     0.3333333  1.1111111
[13] {geweldsdelict}      => {ouders gescheiden}    0.2     0.6666667  1.1111111
[14] {ouders gescheiden}  => {geweldsdelict}        0.2     0.3333333  1.1111111
[15] {gescheiden}         => {ouders gescheiden}    0.3     0.6000000  1.0000000
[16] {ouders gescheiden}  => {gescheiden}           0.3     0.5000000  1.0000000
[17] {drugs}              => {ouders gescheiden}    0.2     0.4000000  0.6666667
[18] {ouders gescheiden}  => {drugs}                0.2     0.3333333  0.6666667
```

Misschien ben je als analist vooral geïnteresseerd in bepaalde aspecten uit de interviews, zoals bijvoorbeeld in de samenhangen tussen (opmerkingen over) drugsgebruik, en de overige items.

We kunnen regels uit bijvoorbeeld het hierboven aangemaakte object **cr5** selecteren met het **`subset()`** commando. Na de naam van het object met de verbindingsregels, moet je dan aangeven welke *items* je wilt selecteren. Voor het selecteren van alle verbindingsregels met "drugs" (zowel als LHS of als RHS) gebruik je het volgende commando. Het is handig om de op **lift** gesorteerde regels als object (**x**) op te slaan. Het object is een dataframe; we kunnen vervolgens selecteren op een waarde voor de variabele **lift** (**x$lift**) groter dan 1.

```
> drugsRules <- subset(cr5, items %in% "drugs")
> x<-inspect(sort(drugsRules, by="lift"))
    lhs                     rhs                    support confidence lift
[1] {baan kwijt}         => {drugs}                0.2     0.6666667  1.3333333
[2] {drugs}              => {baan kwijt}           0.2     0.4000000  1.3333333
[3] {geweldsdelict}      => {drugs}                0.2     0.6666667  1.3333333
[4] {drugs}              => {geweldsdelict}        0.2     0.4000000  1.3333333
[5] {gescheiden}         => {drugs}                0.3     0.6000000  1.2000000
[6] {drugs}              => {gescheiden}           0.3     0.6000000  1.2000000
[7] {drugs}              => {ouders gescheiden}    0.2     0.4000000  0.6666667
[8] {ouders gescheiden}  => {drugs}                0.2     0.3333333  0.6666667

> x[x$lift>1,]
                lhs                rhs support confidence     lift
[1]    {baan kwijt} =>          {drugs}     0.2  0.6666667 1.333333
[2]          {drugs} =>    {baan kwijt}     0.2  0.4000000 1.333333
[3] {geweldsdelict} =>          {drugs}     0.2  0.6666667 1.333333
[4]          {drugs} => {geweldsdelict}     0.2  0.4000000 1.333333
[5]    {gescheiden} =>          {drugs}     0.3  0.6000000 1.200000
[6]          {drugs} =>    {gescheiden}     0.3  0.6000000 1.200000
```

Uit het overzicht lijkt drugsgebruik gerelateerd te zijn aan het kwijtraken van een baan, het plegen van geweldsdelicten, en scheiden.

Dubbele data

In de boodschappenmandjes (en de vertaling ervan in de *sparse matrix*) kan het voorkomen dat bepaalde *items* dubbel voorkomen. Bijvoorbeeld bij het afrekenen scant de caissière twee pakken yoghurt, een voor een. Bij de analyse van welke producten samen worden gekocht is het minder van belang welke hoeveelheden worden gekocht, en één entry voor het *item* yoghurt volstaat.

Hetzelfde geldt voor de interviews in ons voorbeeld: misschien komt "drugs" twee of meer keer ter sprake, maar een *item* voor drugs volstaat. Natuurlijk is het mogelijk om de duplicaties

handmatig of met een slim algoritme te verwijderen, maar gelukkig voorziet de **`read.transactions()`** hierin. Met de **`rm.duplicates`** optie worden dubbele items verwijderd.

Laten we dit toepassen op ons voorbeeld. We hebben voor de records 5 en 7 het *item* drugs toegevoegd, zodat het *item* in die records twee keer voorkomt (zie de gele markeringen, hieronder). Het totaal aantal *items* (opmerkingen) stijgt daardoor van 27 naar 29, waarvan twee duplicaties (items die binnen een record twee of meer keer voorkomen).

```
1:  ouders gescheiden, winkeldiefstal
2:  drugs, gescheiden, geweldsdelict
3:  ouders gescheiden, baan kwijt, drugs, geweldsdelict, belastingschuld
4:  adhd, ouders gescheiden
5:  drugs, gescheiden, drugs
6:  ouders gescheiden, belastingschuld, gescheiden
7:  drugs, baan kwijt, drugs
8:  baan kwijt, belastingschuld
9:  ouders gescheiden, gescheiden, drugs
10: ouders gescheiden, geweldsdelict, gescheiden
```

We lezen het bovenstaande csv-bestand **ML4_OEFEN1D.txt** (het zelfde als **ML4_OEFEN1.txt** maar met duplicaties) in, en vragen om eventuele duplicaties te verwijderen. Uit de output blijkt dat inderdaad 1 item 2 duplicaties heeft. Uit de vergelijking (met het **`summary()`** commando) met het oorspronkelijke (juiste) bestand blijkt dat de gecorrigeerde *sparse matrix* identiek is. Zo komt het *item* "drugs" weer 5 keer voor, en is de *density*, en dus het totale aantal items, hetzelfde!

```
> crimid <- read.transactions("ML4_OEFEN1D.txt",sep=",", rm.duplicates=TRUE)
distribution of transactions with duplicates:
1
2
> summary(crimid)
transactions as itemMatrix in sparse format with
 10 rows (elements/itemsets/transactions) and
 8 columns (items) and a density of 0.3375

most frequent items:
ouders gescheiden             drugs        gescheiden        baan kwijt   belastingschuld
                6                 5                 5                 3                 3
          (Other)
                5

element (itemset/transaction) length distribution:
sizes
2 3 5
5 4 1
```

Visualiseren van de regels

Er zijn diverse manieren om de gevonden verbanden te visualiseren.

Het *package* **arulesViz** biedt een groot scala aan mogelijkheden, waarvan we slechts twee populaire soorten grafieken kort zullen bespreken. Voor een overzicht raden we zeker aan om het *vignette* van het *package* te bekijken[1].

Allereerst, als u dat al niet gedaan hebt, moet u **arulesViz** installeren. En om de functies van het *package* te kunnen gebruiken, gebruikt u het **`library()`** commando.

[1] Het vignette is te vinden op: https://cran.r-project.org/web/packages/arulesViz/vignettes/arulesViz.pdf

```
install.packages("arulesViz")
library(arulesViz)
```

Het eerste type grafiek brengt alle verbindingsregels in kaart. Het is aan te raden alleen de belangrijkste regels in kaart te brengen, bijvoorbeeld door met **`subset()`** een selectie te maken van regels met een **lift** die groter is dan een bepaalde waarde. Wij gebruiken de set van regels **cr5**, omdat die al gebaseerd is op minimale waarden voor **support** en **confidence**.

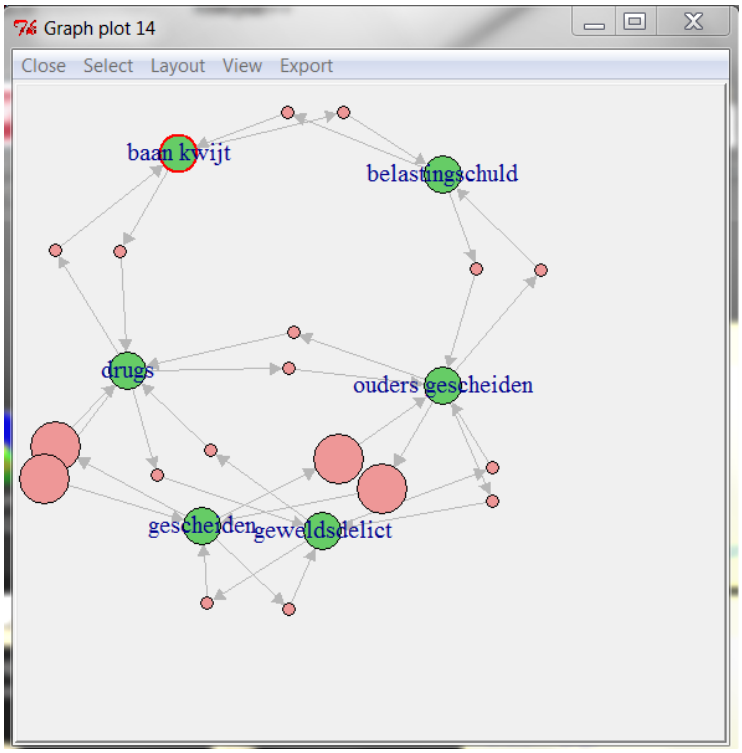
We gebruiken het volgende commando, om een eerste indruk te krijgen van de grafiek.

```
plot(cr5,method="graph",interactive=T,shading=NA)
```

De optie **`interactive=T`**, biedt de mogelijkheid om de grafiek interactief te bewerken. Met de muis kunt u dan de *items* en de verbindingen tussen de *items* zodanig verplaatsen dat de grafiek overzichtelijker wordt.

Ook is er de mogelijkheid bepaalde standaardmethodes voor de lay-out van de grafiek te gebruiken, en ook die kunnen interactief in de plot worden aangegeven.

De grafiek ziet er aanvankelijk als volgt uit:



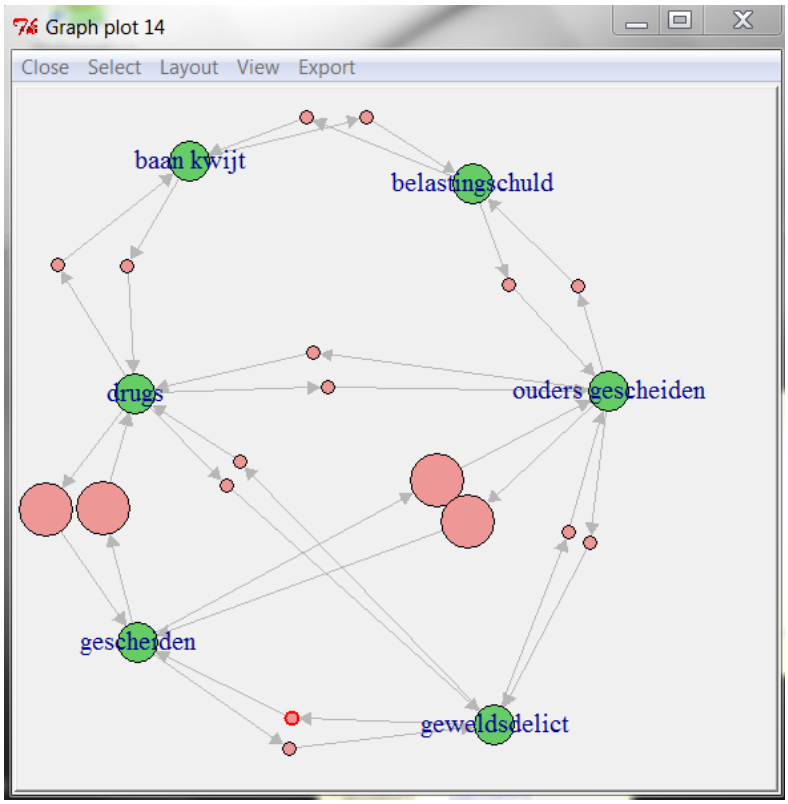
Figuur 14(a). Grafische weergave van de verbindingsregels. *Bron: OGN.*

De roze cirkels en de pijlen, stellen de verbindingen voor. De omvang van de roze cirkels komt overeen met de **support**: hoe groter de cirkel, des te groter de **support** van het *itempaar*.

Omdat, zoals eerder gesteld, de **lift** van, bijvoorbeeld, **{drugs}→{gescheiden}** hetzelfde is als die van **{gescheiden}→{drugs}**, zijn er twee paden tussen deze items, met gelijke **lift**.

Onder aan de grafiek lopen de pijlen door elkaar aan, wat het geheel erg onoverzichtelijk maakt. Maar u kunt alle cirkels met de muis naar een andere positie slepen. Probeer het zelf uit!

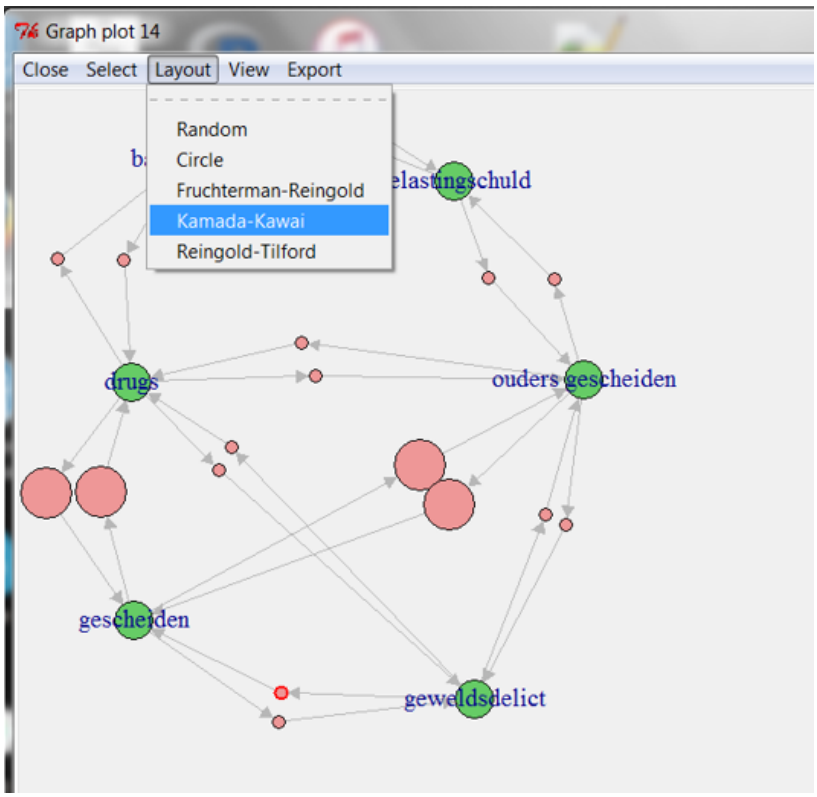
Onderstaand hebben we enkele aanpassingen gemaakt aan de onderkant van de grafiek.

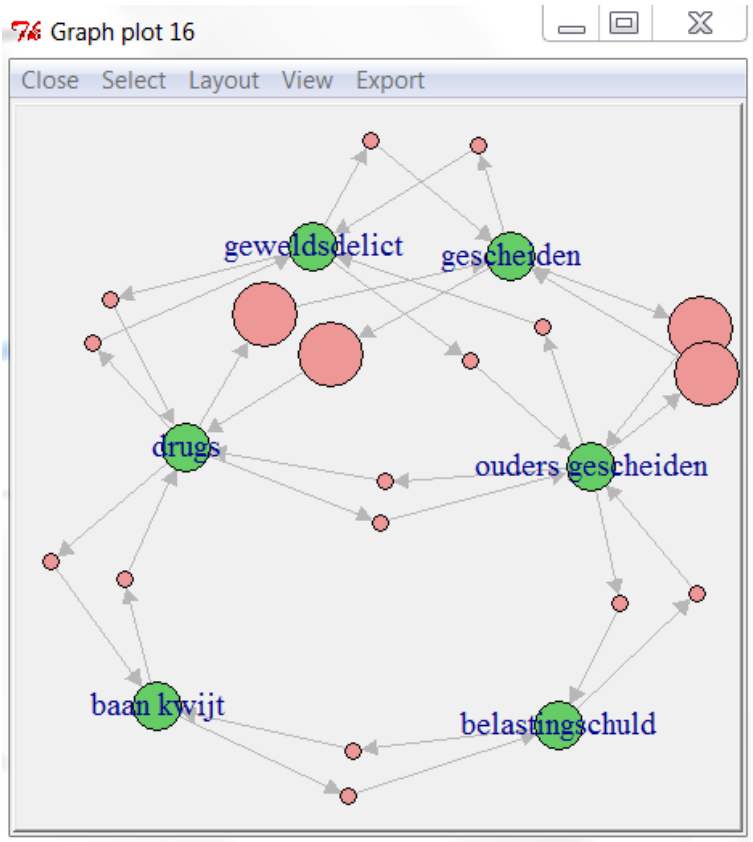


Figuur 14 (b). Grafische weergave van de verbindingsregels. *Bron: OGN.*

De standaardmethodes staan in de **lay-out** tab van het menu. Het beste is enkele van de mogelijkheden uit te proberen, en te kijken wat het beste resultaat geeft. Binnen sommige standaardmethodes zijn parameters aan te geven. Maar het is ook mogelijk de standaardwaarden te accepteren, en daarna de cirkels interactief te verslepen.

Bij wijze van voorbeeld hebben we het Kamada-Kawai algoritme toegepast.





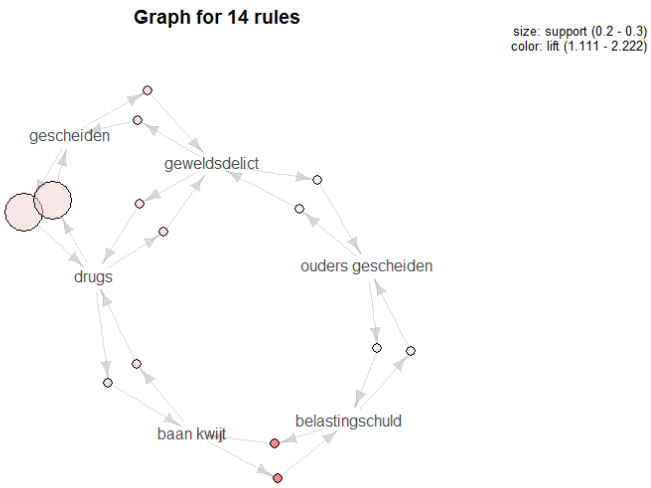
Figuur 14 (c en d). Grafische weergave van de verbindingsregels. *Bron: OGN.*

In de meeste gevallen is het beter om niet alle verbindingsregels klakkeloos in de grafiek op te nemen, maar een beredeneerde selectie te maken. We zullen twee selecties aanbrengen:

- Allereerst beperken we ons tot de verbindingsregels met een **lift**>1.
- En vervolgens kiezen we alleen de verbindingsregels die in een logische richting lopen. Het is bijvoorbeeld logischer dat het kwijtraken van een baan leidt tot een belastingschuld, dan andersom. Of de kans op scheiden is groter als je ouders gescheiden zijn, terwijl het minder aannemelijk is dat je ouders scheiden ten gevolge van jouw scheiding.

Allereerst bepalen we een subset met **lift**>1, en drukken die af. We gebruiken nu niet de optie voor interactief bewerken van de grafiek. Dat heeft twee voordelen. Ten eerste is het resultaat erg overzichtelijk. Ten tweede geeft de grafiek informatie over zowel de **support** (de omvang van de cirkels) als over de **lift** (de kleur van de cirkels; hoe donkerder de kleur des te groter de **lift**).

```
cr5s <- subset(cr5, lift>1)
plot(cr5s, method="graph")
```



Figuur 15(a). Grafische weergave van de verbindingsregels (niet interactief). *Bron: OGN.*

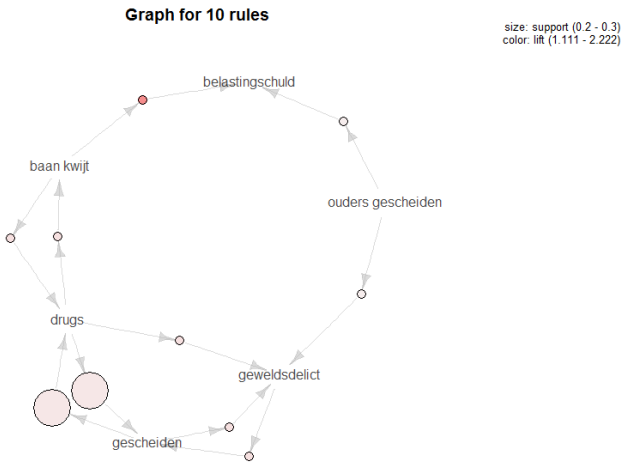
Met het **`inspect()`** commando drukken we de 14 verbindingsregels af. We hebben die verbindingen geel gemarkeerd die ons het meest logisch lijken.

Bijvoorbeeld: de relatie tussen drugs en gescheiden, kan twee kanten op. Maar het lijkt niet logisch dat een belastingschuld leidt tot het scheiden van iemands ouders. Natuurlijk zijn dit min of meer subjectieve beslissingen, waarover we het niet eens hoeven te zijn: de beslissing is niet aan de software maar aan de analist!

```
> inspect(cr5s)
     lhs                     rhs                     support confidence lift
[1]  {baan kwijt}         => {belastingschuld}       0.2     0.6666667  2.222222
[2]  {belastingschuld}    => {baan kwijt}            0.2     0.6666667  2.222222
[3]  {baan kwijt}         => {drugs}                 0.2     0.6666667  1.333333
[4]  {drugs}              => {baan kwijt}            0.2     0.4000000  1.333333
[5]  {belastingschuld}    => {ouders gescheiden}     0.2     0.6666667  1.111111
[6]  {ouders gescheiden}  => {belastingschuld}       0.2     0.3333333  1.111111
[7]  {geweldsdelict}      => {gescheiden}            0.2     0.6666667  1.333333
[8]  {gescheiden}         => {geweldsdelict}         0.2     0.4000000  1.333333
[9]  {geweldsdelict}      => {drugs}                 0.2     0.6666667  1.333333
[10] {drugs}              => {geweldsdelict}         0.2     0.4000000  1.333333
[11] {geweldsdelict}      => {ouders gescheiden}     0.2     0.6666667  1.111111
[12] {ouders gescheiden}  => {geweldsdelict}         0.2     0.3333333  1.111111
[13] {gescheiden}         => {drugs}                 0.3     0.6000000  1.200000
[14] {drugs}              => {gescheiden}            0.3     0.6000000  1.200000
```

We laten vier van de 14 verbindingsregels achterwege. We maken een nieuwe subset met het onderstaande commando, en geven de grafiek weer.

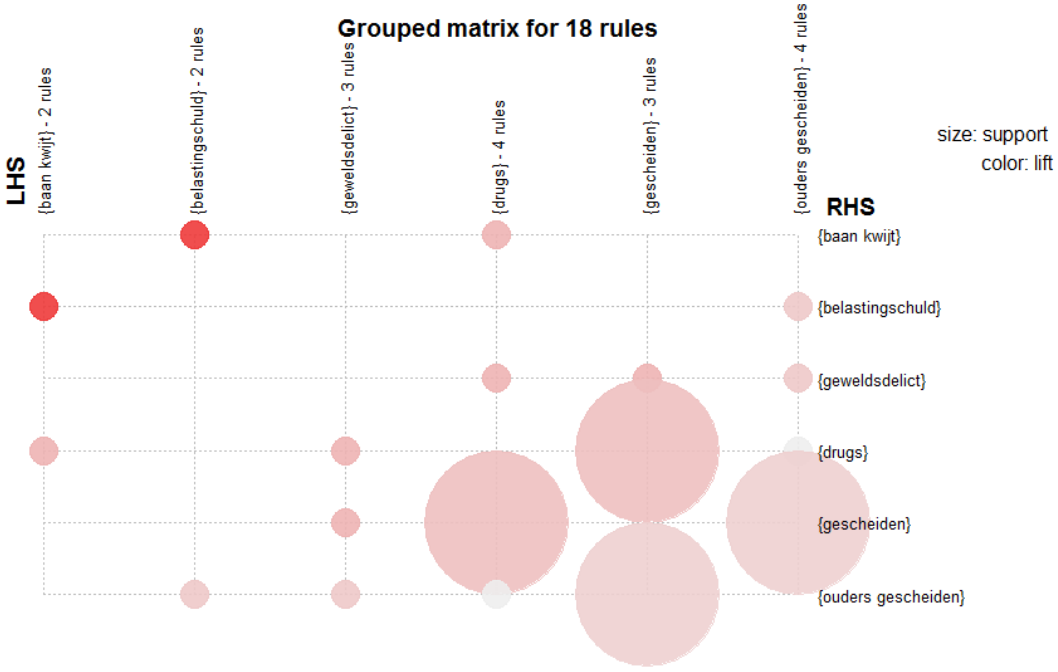
```
cr5s2 <- subset(cr5s, c(1,3,4,6,7,8,10,12,13,14)); inspect(cr5s2)
plot(cr5s2,method="graph")
```



Figuur 15(b). Grafische weergave van selectie van verbindingsregels (niet interactief). *Bron: OGN.*

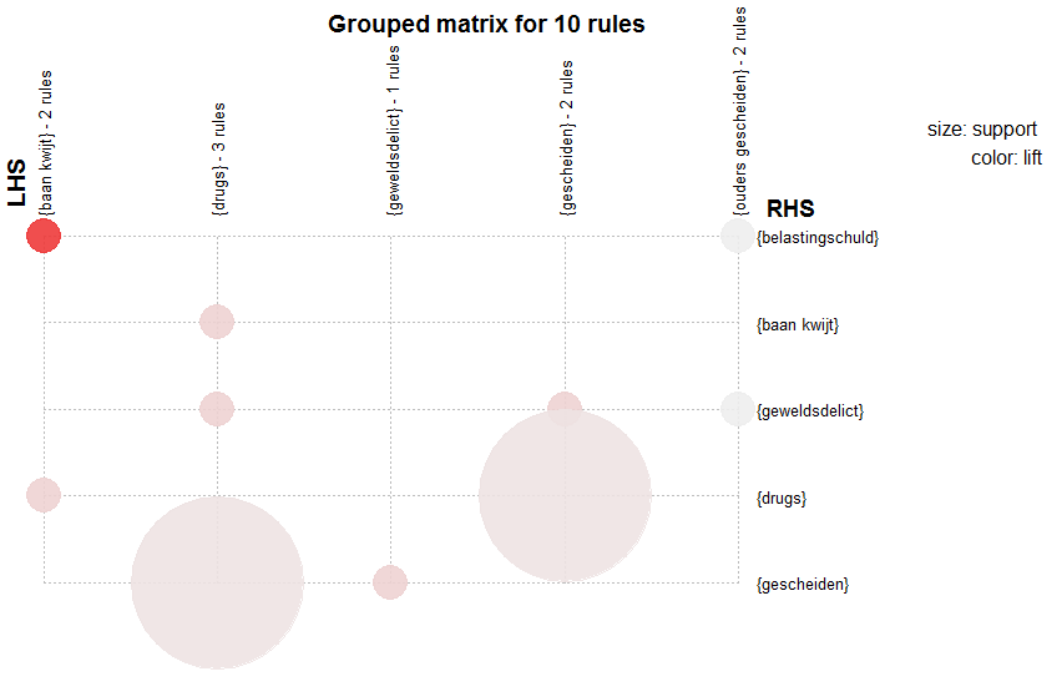
Dezelfde informatie kan worden weergegeven in een zogeheten *grouped matrix*. De matrix geeft alle items weer aan de LHS en alle items aan de RHS, en geeft voor elk *itempaar* de **support** en de **lift** weer, met de omvang en de kleur van de cirkels.

Uit de grafiek is snel te zien dat drugs vaak samengaat met scheiden, en dat het kwijtraken van een baan een goede voorspeller is van belastingschuld (of andersom). Merk op dat de grafiek symmetrisch is: de **lift** voor **{baan kwijt}→{belastingschuld}** is hetzelfde als voor **{belastingschuld}→{baan kwijt}**, en de **support** voor het *itempaar* ligt vast. Of anders gezegd: de twee cirkels het meest linksboven zijn even groot (**support**), en hebben dezelfde donkerrode kleur (**lift**).



Figuur 16(a). Grafische weergave van de verbindingsregels: grouped matrix. *Bron: OGN.*

We kunnen ons, net als eerder, beperken tot de 10 regels die ons het meest logisch lijken. We passen dan het commando toe op het object **cr5s2**. Dat heeft als nadeel dat de LHS en RHS items niet meer in dezelfde volgorde staan, wat het lezen van de grafiek bemoeilijkt. Het voordeel is dat onlogische relaties niet meer zijn weergegeven wat leidt tot minder (namelijk alleen relevante!) informatie. Een ideaal design is om in de selectie zo min mogelijk items te hebben die zowel aan de LHS als de RHS voorkomen.



Figuur 16(b). Grafische weergave van de verbindingsregels: grouped matrix. *Bron: OGN.*