Informatie Uitwisseling

Matt ter Steege
matttersteege@gmail.com
Universiteit Utrecht
Utrecht, Nederland

Contents

1 Semiotiek

Semiotiek is het zelfde als 'tekenleer'. Denk hierbij aan:

- letters (A-Z)
- Karakters (chinees alfabet)
- Woorden
- Morsetekens/braille
- verkeerdborden, pictogrammern
- gebaren
- voorwerpen (witte vlag, e.d)

Semiotiek houd zich bezeig met elke activiteit, handeling of preces waarbij tekens worden gebruitk. Een teken defineren we als: "Alles dat een booschap communiceert van de zender naar de ontvanger"

1.1 De semiotische ladder

1.2 Fysieke tekens (Technische laag)

Dit houdt alles in van Klanken die je met je mond maakt tot gebaren, letters, Geuren e.d. Tekens kunnen op zichzelf al een betekenis hebben, maar kunnen ook met elkaar een betekenis hebben. Bijvoorbeeld: Het naam "Roderick" is een reeks van tekens (letters) die samen een betekenis hebben. Ofwel, de drager van de boodschap.

1.3 Het empirische niveau (Technische laag)

Dit niveau houd zich bezig met de waarneembare eigenschappen van tekens. Bijvoorbeeld: De letters in het woord "Roderick" hebben een bepaalde vorm, kleur, grootte e.d. Ook de klanken die je maakt als je het woord uitspreekt hebben bepaalde eigenschappen zoals toonhoogte, volume e.d. Dit niveau is vooral belangrijk voor de technische verwerking van tekens, zoals bij spraakherkenning of beeldherkenning. Ofwel, hoe de boodschap wordt overgebracht.

1.4 Syntax

Syntax is de studie van de regels die bepalen hoe tekens gecombineerd kunnen worden om grotere eenheden te vormen. Bijvoorbeeld: In het Nederlands is de volgorde van woorden in een zin belangrijk voor de betekenis. "De kat zit op de mat" heeft een andere betekenis dan "Op de mat zit de kat". Syntax is dus de structuur van tekens en hoe ze samenhangen. Ofwel, hoe de boodschap is opgebouwd.

1.5 Semantiek

Semantiek is de studie van de betekenis van tekens en hoe ze worden geïnterpreteerd. Bijvoorbeeld: Het woord "kat" verwijst naar een bepaald dier, maar in een andere context kan het ook een metafoor zijn voor iets anders. Semantiek gaat dus over de relatie tussen tekens en hun betekenis. Ofwel, wat de boodschap inhoudt.

Objectivisme: Beteknis is vast en objectief. Woorden hebben een vaste betekenis die niet verandert. Denk aan woordenboeken die de betekenis van woorden definiëren.

Constructivisme: Betekenis is subjectief en contextafhankelijk. Woorden kunnen verschillende betekenissen hebben afhankelijk van de context en de interpretatie van de ontvanger. Denk aan hoe straattaal of jargon verschillende betekenissen kunnen hebben in verschillende groepen. (bijv. "cool" kan zowel "koud" als "gaaf" betekenen)

Pragmatisme: Betekenis ontstaat in de interactie tussen zender en ontvanger. Woorden krijgen betekenis door het gebruik ervan in communicatie. Denk aan hoe de betekenis van een woord kan veranderen afhankelijk van hoe het wordt gebruikt in een gesprek.

Determinisme: Betekenis wordt bepaald door externe factoren zoals cultuur, geschiedenis en sociale context. Woorden kunnen verschillende betekenissen hebben in verschillende culturen of tijdperken. Denk aan hoe bepaalde woorden in het verleden een andere betekenis hadden dan nu. (bijv. "gay" betekende vroeger "vrolijk" maar nu wordt het voornamelijk gebruikt om seksuele geaardheid aan te duiden)

1.6 Pragmatiek

Pragmatiek is de studie van hoe tekens worden gebruikt in communicatie en hoe de context de betekenis beïnvloedt. Bijvoorbeeld: Het woord "kat" kan verschillende betekenissen hebben afhankelijk van de situatie waarin het wordt gebruikt. Als iemand zegt "Ik heb een kat", kan dit betekenen dat ze een huisdier hebben, maar het kan ook een uitdrukking zijn van iets anders, afhankelijk van de context. Pragmatiek gaat dus over de praktische aspecten van communicatie en hoe tekens worden gebruikt in de echte wereld. Ofwel, de bedoeling achter de boodschap.

1.7 Samenvatting

De semiotische ladder bestaat uit vijf niveaus:

- Fysieke tekens (Technische laag): De drager van de boodschap
- Empirische niveau (Technische laag): Hoe de boodschap wordt overgebracht
- Syntax: Hoe de boodschap is opgebouwd
- Semantiek: Wat de boodschap inhoudt
- Pragmatiek: De bedoeling achter de boodschap

2 Kennisbanken

Graafstructuren bestaan uit een verzameling van punten en knopen die met elkaar verbonden zijn door lijnen (ongerichte graaf) of pijlen (gerichte graaf). In een kennisbank worden concepten voorgesteld als knopen en de relaties tussen deze concepten als lijnen of pijlen. Kennisbanken kunnen worden gebruikt om informatie te organiseren, te structureren en te analyseren.

2.1 relationele database

Een relationele database is een type database dat gegevens opslaat in tabellen die met elkaar verbonden zijn door relaties. Elke tabel bestaat uit rijen en kolommen, waarbij elke rij een record vertegenwoordigt en elke kolom een attribuut van dat record (denk aan een spreadsheet). Relaties tussen tabellen worden gemaakt door middel van primaire en vreemde sleutels. Relationele databases worden vaak gebruikt voor het opslaan van gestructureerde gegevens en het uitvoeren van complexe queries.

2.2 SQL

SQL (Structured Query Language) is een programmeertaal die wordt gebruikt voor het beheren en manipuleren van relationele databases. Met SQL kunnen gebruikers gegevens opvragen, invoegen, bijwerken en verwijderen uit de database. SQL biedt ook mogelijkheden voor het definiëren van de structuur van de database, zoals het maken van tabellen en het definiëren van relaties tussen tabellen. SQL is een gestandaardeerde taal en wordt ondersteund door de meeste relationele databasebeheersystemen.

2.3 NoSQL

NoSQL (Not Only SQL) is een type database dat niet gebaseerd is op het relationele model. In plaats daarvan gebruiken NoSQL-databases verschillende gegevensmodellen, zoals documenten, grafieken, kolomgeoriënteerde opslag en sleutel-waardeparen. NoSQL-databases zijn ontworpen om schaalbaarheid, flexibiliteit en prestaties te bieden voor het opslaan van grote hoeveelheden ongestructureerde of semi-gestructureerde gegevens. NoSQL-databases worden vaak gebruikt in big data-toepassingen en real-time webapplicaties.

2.4 Faceted Search

Faceted search is een zoektechniek die gebruikers in staat stelt om zoekresultaten te verfijnen door middel van verschillende categorieën. In een faceted search-systeem worden zoekresultaten georganiseerd op basis van verschillende categorieën, zoals prijs, merk, kleur, grootte, enzovoort. Gebruikers kunnen vervolgens filters toepassen op deze categorieën om de zoekresultaten te beperken tot diegene die aan hun specifieke criteria voldoen. Faceted search wordt vaak gebruikt in e-commerce websites en digitale bibliotheken om gebruikers te helpen snel de gewenste informatie te vinden.

2.5 Term Frequency enInverse Document Frequency ($TF \times IDF$)

Term Frequency (TF) is een maatstaf voor hoe vaak een term voorkomt in een document. Het wordt berekend door het aantal keren dat een term voorkomt in een document te delen door het totale aantal termen in dat document. (bijv. als het woord "kat" 3 keer voorkomt in een document met 100 woorden, is de TF van "kat" 3/100 = 0.03 = 3%) Inverse Document Frequency (IDF) is een maatstaf voor hoe belangrijk een term is in een verzameling documenten. Het wordt berekend door het totale aantal documenten te delen door het aantal documenten waarin de term voorkomt, en vervolgens de logaritme van dat quotiënt te nemen. (bijv. als het woord "kat" voorkomt in 10 van de 1000 documenten, is de IDF van "kat" log(1000/10) = log(100) = 2) De combinatie van TF en IDF, oftewel $TF \times IDF$, wordt gebruikt om de relevantie van een term in een document te bepalen binnen een verzameling documenten. Een hoge $TF \times IDF$-waarde geeft aan dat een term vaak voorkomt in een document, maar zelden in andere documenten, wat suggereert dat de term belangrijk is voor dat specifieke document.

3 RDF (Resource Description Framework)

RDF is een standaardmodel voor het uitwisselen van gegevens op het web. Het is ontworpen om gegevens te beschrijven in een gestructureerde en machine-leesbare manier. RDF maakt gebruik van een grafenmodel om gegevens te representeren, waarbij gegevens bestaan uit een subject, predicaat en object. Dit maakt het mogelijk om complexe relaties tussen gegevens te modelleren en te analyseren.

```
subject:  http://example.org/person/Alice
predicate: http://xmlns.com/foaf/0.1/knows
object:   http://example.org/person/Bob
```

In dit voorbeeld beschrijft de RDF-verklaring dat Alice Bob kent. RDF maakt gebruik van Uniform Resource Identifiers (URI's) om resources te identificeren, wat zorgt voor een unieke en consistente manier om gegevens te verwijzen.

FOAF (Friend Of A Friend) is ontwikkeld om mensen en hun relaties (met andere mensen of objecten) te beschrijven op het web.

Binnen het RDF heb je ok het RDF Schema (RDFS), dit is een verzameling van klassen en eigenschappen die worden gebruikt om RDF-gegevens te beschrijven. Bijvoorbeeld, je kunt een klasse "Persoon" definiëren en eigenschappen zoals "heeftNaam" en "heeftLeeftijd" om de attributen van een persoon te beschrijven. RDF en RDFS beschrijven alleen de structuur van de gegevens, maar niet de betekenis ervan.

3.1 linked data

Linked Data is een methode om gestructureerde gegevens op het web te publiceren en te verbinden. Het maakt gebruik van RDF om gegevens te beschrijven en URI's om resources te identificeren. Linked Data maakt het mogelijk om gegevens van verschillende bronnen te combineren

en te integreren, waardoor een rijker en meer verbonden web van gegevens ontstaat. Het concept van Linked Data is gebaseerd op vier principes:

- Gebruik URI's om resources te identificeren.
- Gebruik HTTP-URI's zodat resources kunnen worden opgezocht op het web.
- Gebruik RDF om gegevens te beschrijven.
- Verbind je gegevens met andere gegevensbronnen om een web van gegevens te creëren.

Zo is het dus mogelijk om af te leiden dat een persoon een studeert met een x aantal medestudenten:

- <http://example.org/person/MattTerSteege>
- rdf:type foaf:Person
- foaf:name "Matt ter Steege"
- foaf:studiesAt <http://dbpedia.org/resource/Universiteit_Utrecht>
- <http://dbpedia.org/resource/Universiteit_Utrecht>
- dbp:student_count "40000"
- dbp:location "Utrecht, Nederland"

3.2 SPARQL

SPARQL is een querytaal voor het opvragen en manipuleren van RDF-gegevens. Het stelt gebruikers in staat om complexe queries uit te voeren op RDF-datasets en specifieke informatie op te halen. Een query bestaat uit een prefix-declaratie, een select-verklaring en een waar-verklaring. Bijvoorbeeld, de volgende SPARQL-query haalt de namen op van alle personen die studeren aan de Universiteit Utrecht:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name
WHERE {
    ?person foaf:studiesAt <http://dbpedia.org/resource/Universiteit_Utrecht> .
    ?person foaf:name ?name .
}
```

* De "?name" betekendt hetzelde als "<http://xmlns.com/foaf/0.1/name>" maar dan korter geschreven.

4 Webnavigatie

4.1 Web usage mining

Dit gaat over het ontdekken van patronen in webdata, zoals gebruikersgedrag op websites. Het doel is om inzicht te krijgen in veel bezochte pagina's en websites of veel voorkomende paden van navigatie. Zo kan je begrijpen welke taken en behoeften gebruikers hebben, wat gebruiksonvriendelijke elementen zijn en hoe je de website kunt verbeteren (UI/UX of snelhied etc.). Het volgende zijn ook voorbeelden:

- Identificeren van advertentielocaties.
- Optimaliseren van menu-desing.
- Herkennen van bots en frauduleuze activiteiten.
- Personaliseren van content en aanbevelingen.
- Voorspellen van de volgende actie van een gebruiker

4.2 gebruikersdata

Gebruikersprofielen: zijn gegevens die door de gebruiker zelf zijn verstrekt, zoals naam, leeftijd, geslacht, locatie en interesses. **Gebruiksdata**: omvat informatie over hoe gebruikers omgaan met een website, zoals bezochte pagina's, klikgedrag, tijd besteed op pagina's en navigatiepaden.

Deze data kan je op verschillende manieren verzamelen, zoals:

- Webserver: Voornamelijk klikgedrag, bezochte pagina's en tijd op pagina.
- Webclient: Data van één gebruiker op verschillende sites, zoals muisbewegingen, scrollgedrag en interacties.
- Proxyservers: Data van meerdere gebruikers, zoals bezochte sites en algemene navigatiepatronen.

4.3 Webserverdata

: Als een gebruiker een website bezoekt, registreert de webserver automatisch verschillende gegevens, zoals het IP-adres van de gebruiker, de tijd van het bezoek, de bezochte pagina's en de duur van het bezoek. Deze gegevens worden opgeslagen in logbestanden die later kunnen worden geanalyseerd om inzicht te krijgen in het gedrag van gebruikers op de website.

Een voorbeeld van een webserverlogbestand:

```
• 213.6.31.68 - - [01/May/2004:22:38:32 +0200] "GET
  /forsale.html HTTP/1.1" 200 14956
  "http://www.fortepiano.nl/indexforsale.html" "Mozilla/4.0
  (compatible; MSIE 6.0; Windows NT 5.1)

• 213.6.31.68 - - [01/May/2004:22:38:34 +0200] "GET
  /pictures/forsale/bertsche1835-small.jpg HTTP/1.1" 200 5753
  "http://www.fortepiano.nl/forsale.html" "Mozilla/4.0
```

```
(compatible; MSIE 6.0; Windows NT 5.1)
```

Zo'n logbestand bestaat uit alle requests aan een webserver/website. Als je hier data uit wilt halen, dan moet je deze eerst voorbewerken (cleanen).

(1) **Data cleaning**: Verwijder onnodige gegevens, zoals requests voor afbeeldingen, scripts en stylesheets.
(2) **User identification**: Identificeer unieke gebruikers op basis van IP-adressen en user-agent strings. Je kunt dit doen door bijvoorbeeld cookies te gebruiken, maar accounts of headers kunnen ook helpen.
(3) **Session identification**: Groepeer requests van dezelfde gebruiker binnen een bepaalde tijdsperiode tot één sessie. Je kunt hiervoor een tijdslimiet instellen (bijv. 30 minuten van inactiviteit betekent het einde van een sessie).
(4) **Path completion**: Vul ontbrekende pagina's in die mogelijk zijn overgeslagen door caching of directe toegang.
(5) **Robot identification**: Identificeer en verwijder requests van webcrawlers en bots. Dit kan gedaan worden door user-agent strings te analyseren of door bekende IP-adressen van bots te blokkeren. Maar ook door het gedrag te analyseren (bijv. als een gebruiker in een paar seconden 100 pagina's bezoekt, is het waarschijnlijk een bot).

4.4 webclientdata

Webclientdata wordt verzameld via scripts die op de client-side (de browser van de gebruiker) worden uitgevoerd. Deze scripts kunnen informatie verzamelen over het gedrag van de gebruiker op de website, zoals muisbewegingen, scrollgedrag, klikgedrag en tijd besteed op pagina's. Deze gegevens worden vervolgens naar de server gestuurd voor analyse. Populaire tools voor het verzamelen van webclientdata zijn Google Analytics, Hotjar en Mixpanel. Dit is door de GDPR wet aardig in onbruik geraakt.

Conversion tracking: Dit houdt bij hoeveel gebruikers een specifieke actie voltooien, zoals het invullen van een formulier of het doen van een aankoop. Dit werd vaak gedaan met een "spy pixel", een onzichtbaar beeld dat wordt geladen wanneer een gebruiker een bepaalde pagina bezoekt. Om te checken hoe goed emailcampagnes werken of advertenties.

5 Voorspellende modellen

Wat zijn voorspellende modellen

Voorspellende (of probabilistische) modellen proberen niet *zeker* te weten wat een gebruiker gaat doen, maar schatten de kans dat een bepaalde handeling zal plaatsvinden. Alles draait om waarschijnlijkheden. Je kijkt naar eerder gedrag en zegt: *op basis hiervan is dit de meest waarschijnlijke volgende stap.*

Een handeling noemen we (a). De kans dat deze handeling voorkomt ligt altijd tussen 0 en 1:

$$0 \leq P(a) \leq 1$$

Deze kans heet de a-priori kans. Die staat los van context. Sommige handelingen gebeuren vaak (homepage bezoeken), andere zelden (voorwaardenpagina openen).

Voorwaardelijke kansen

Zodra je context meeneemt, verandert het spel. Als handeling (b) vaak gevolgd wordt door handeling (a), dan wordt (a) waarschijnlijker zodra (b) is waargenomen.

Dat schrijven we als:

$$P(a \mid b)$$

Dit lees je als: *de kans op a, gegeven dat b heeft plaatsgevonden.*

Concreet:

$$P(a \mid b) = \frac{\text{aantal keren dat b gevolgd wordt door a}}{\text{aantal keren dat b voorkomt}}$$

Dit is nog steeds simpel tellen, maar nu mét context.

Theorema van Bayes

Bayes generaliseert dit idee. Niet één vorige handeling telt mee, maar een hele set bewijsstukken (E) (een keten van acties).

De formule:

$$P(a \mid E) = \frac{P(E \mid a) \cdot P(a)}{P(E)}$$

Wat hier gebeurt:

- (P(a)): hoe waarschijnlijk was actie a sowieso al
- $(P(E \mid a))$: hoe goed past het waargenomen gedrag bij actie a
- (P(E)): normalisatie, zodat de kans geldig blijft

In praktijk: gebruikersgeschiedenis weegt mee, maar nooit zonder de basispopulariteit van een actie te corrigeren.

Rankingmethoden op basis van a-priori kansen

Niet alle pagina's zijn gelijk. Sommige pagina's worden structureel vaker bezocht dan andere. Rankingmethoden proberen pagina's te ordenen op basis van herbezoekkans.

Typische observaties:

- Een klein aantal pagina's trekt het merendeel van het verkeer
- Gebruikers keren vaak terug naar recent bezochte pagina's

Dit leidt tot eenvoudige, maar verrassend effectieve methoden.

Last Recently Used (LRU)

LRU kijkt alleen naar *recentheid*.

$$LRU(m_i, I_{m_i}, i_n) = \frac{1}{i_n - i_k + 1}$$

Waar:

- (i_n) de index is van het laatste paginabezoek
- (i_k) de index is van het laatste bezoek aan pagina (m_i)

Hoe recenter het bezoek, hoe hoger de score.

Most Frequently Used (MFU)

MFU kijkt alleen naar *frequentie*.

$$MFU(m_i, I_{m_i}, i_n) = \frac{|I_{m_i}|}{i_n}$$

Pagina's die vaak bezocht zijn in het verleden krijgen een hogere kans, ongeacht hoe lang geleden dat was.

Polynomial Decay (PD)

Polynomial Decay combineert frequentie en recentheid.

$$PD(m_i, I_{mi}, i_n) = \sum_{j=1}^{|I_{m_i}|} \frac{1}{1 + (i_n - i_j)^\alpha}, \quad 0 < \alpha \leq 1$$

Recente bezoeken tellen zwaarder dan oude. De parameter (α) is bepalend:

- hoge (α): snelle afname, focus op recentheid
- lage (α): langzamere afname, focus op frequentie

Markov-modellen

Markov-modellen veronderstellen dat de volgende actie afhangt van eerdere acties.

Een eerste-orde Markov-model gaat ervan uit dat alleen de *laatste* actie relevant is.

$$P(s_n) = P(s_1) \prod_{t=2}^{n} P(s_t \mid s_{t-1})$$

Dit model kan worden weergegeven als een transitiematrix:

From/To	A	B	C	D	Totaal
A		3	5	8	16
B	3		7	4	14
C	2	4		6	12
D	1	6	2		9

Elke cel geeft aan hoe vaak een overgang is waargenomen.

Hogere-orde Markov-modellen

Bij hogere-orde modellen hangt de volgende actie af van de laatste (k) acties. Dit is realistischer, maar schaalt slecht. Daarom worden pagina's of gebruikers vaak geclusterd om matrices hanteerbaar te houden.

Associatieregels

Associatieregels kijken niet naar volgorde, maar naar *samen voorkomen*.

Een regel heeft de vorm:

$$X \Rightarrow Y$$

Waarbij (X) en (Y) verzamelingen van items zijn zonder overlap.

Support en Confidence

Support:

$$support(X \Rightarrow Y) = \frac{|t_i \in D : X \cup Y \subset t_i|}{|D|}$$

Confidence:

$$confidence(X \Rightarrow Y) = \frac{|t_i \in D : X \cup Y \subset t_i|}{|t_i \in D : X \subset t_i|}$$

Support is vaak laag, confidence moet hoog zijn om de regel bruikbaar te maken.

A-priori algoritme

Het A-priori algoritme reduceert de zoekruimte drastisch:

- Als een itemset niet frequent is, kan geen enkele superset frequent zijn
- Frequent-itemsets worden iteratief opgebouwd

Dit maakt het vinden van associatieregels praktisch uitvoerbaar.

Praktische realiteit

Niet elk product of elke pagina volgt hetzelfde patroon. Wat werkt voor media en smaakgevoelige producten, faalt bij seizoensgebonden of functionele aankopen. Voorspellende modellen zijn krachtig, maar alleen zolang je accepteert dat gebruikersgedrag niet netjes, stabiel of symmetrisch is.

6 Kennis zoeken

6.1 Interfaces ontwerpen

Waarom ontwerpen gebruikers geen interfaces? Nou, dat komt door een paar dingen, zoals:

- Gebrek aan technische kennis, het idee hebben ze misschien wel, maar niet de vaardigheden om het uit te voeren.
- Gebruikers zien vaak niet of er andere interfaces zijn die beter bij hun behoeften passen. Zo heb je in het begin van het gebruik van een interface hulp nodig met alles vinden, leren en begrijpen (dmv wizards, menu's etc.). Maar als je het eenmaal onder de knie hebt, dan wil je graag sneltoetsen en shortcuts gebruiken, omdat dat nou eenmaal sneller werkt.
- Gebruikers weten simpelweg niet of ze de interface fijn vinden totdat het af is.

6.2 verschillende categorieën van informatie zoeken

- **Serendipitous browsing**: Je bent niet echt op zoek, je kijkt gewoon rond en hoopt iets interessants te vinden.
- **Exploratory seeking**: Je hebt een vaag doel, maar je moet alleen nog meer informatie vinden om dat doel te bereiken.
- **Semi-directed browsing**: Je hebt een specifiek doel, maar je weet niet precies waar je moet zoeken.
- **Known-item searching**: Je weet precies wat je zoekt en waar je het kunt vinden.
- **Re-finding**: Je hebt eerder iets gevonden, maar je moet het opnieuw vinden.

6.3 Information Foraging

De information foraging theorie is gebaseerd op het idee dat mensen die op zoek zijn naar informatie zich net zo gedragen als dieren op zoek naar eten. We hebben ingebouwde zoek- en jachtstrategieën die we automatisch toepassen. Deze strategieën zijn het resultaat van millennia aan evolutie. De theorie gaat ervan uit dat mensen (en dieren), binnen hun mogelijkheden, de beste strategie kiezen om met zo weinig mogelijk moeite (verbruikte energie) het beste resultaat krijgen (een volle maag, de benodigde kennis).

6.4 Modellen van information foraging

Information patch model: Dit model stelt dat informatie zich bevindt in "patches'' of clusters, vergelijkbaar met voedselbronnen in de natuur. Gebruikers moeten beslissen wanneer ze een patch moeten verlaten en naar een nieuwe moeten gaan, gebaseerd op de hoeveelheid beschikbare informatie en de moeite die het kost om deze te verkrijgen.
Information scent model: Dit model richt zich op de signalen of "scent" die gebruikers volgen om informatie te vinden. Deze signalen kunnen bestaan uit hyperlinks, zoekresultaten of andere aanwijzingen die gebruikers helpen te navigeren naar de gewenste informatie.
Information diet model: Dit model vergelijkt informatieconsumptie met voedselinname. Het suggereert dat gebruikers een "dieet" van informatie kiezen op basis van hun behoeften, voorkeuren en beschikbare tijd.

Information Foraging gaat er niet van uit dat we ons volledig rationeel en logisch gedragen. We maken fouten, omdat we incomplete informatie hebben. We wegen de benodigde inspanning af tegen de verwachte opbrengst. Dit kan leiden tot suboptimale keuzes, maar het is vaak goed genoeg in de praktijk.

Stel je voor, je bent een roofvogel die op zoek is naar voedsel.

- Hoeveel prooi kun ik vinden in een bepaald gebied?
- Zijn de dieren makkelijk of moeilijk te vangen? En zit er genoeg vlees aan?

- Hoeveel energie kost het om te jagen in dit gebied?

6.4.1 Information patches Informatie is vaak verdeeld is clusters of "patches". Denk aan een bibliotheek met verschillende secties (geschiedenis, wetenschap, kunst, enzovoort). Elke sectie is een informatiepatch met gerelateerde informatie. Gebruikers moeten beslissen wanneer ze een patch moeten verlaten en naar een nieuwe moeten gaan. Dit hangt af van de hoeveelheid beschikbare informatie en de moeite die het kost om deze te verkrijgen.

Dit patch-model heeft een wiskundige formule centraal staan:

$$R = \frac{G}{T_B + T_W}$$

Waarbij:

- R is de opbrengst (reward rate)
- G is de hoeveelheid verkregen informatie (gain)
- T_B is de tijd besteed aan het vinden van een patch
- T_W is de tijd besteed aan het verwerken van informatie binnen een patch

Het doel is om R te maximaliseren door een optimale balans te vinden tussen G, T_B en T_W.

Deze winstmaximalisatie kan je dus op 2(of 3) manieren bereiken:

- De tijd tussen de bossen verkleinen (sneller nieuwe patches vinden) Aks je dit doet dan gaat de winstverwachting omhoog.
- De tijd in de bossen verkleinen (sneller informatie verwerken) Als je dit doet, dan kan je meer patches bezoeken in dezelfde tijd, wat de totale informatieopbrengst verhoogt. De optimale "within-patch" tijd gaat hierdoor ook omlaag, want stel 75% van de informatie is optimaal, dan ben je daar sneller als je sneller door de patch heen gaat.
- De hoeveelheid informatie per patch verhogen (betere kwaliteit informatie, maar kan je van tevoren niet weten) Als je dit doet, dan verhoog je de totale informatieopbrengst per patch, wat de winstverwachting verhoogt.

6.4.2 Informatiezoekers Informatiezoekers zijn als jagers die op zoek zijn naar informatie. Denk aan studenten, onderzoekers, kantoorpersoneel, enzovoort. Deze hebben informatie nodig om hun taken uit te voeren, beslissingen te nemen of problemen op te lossen.

- De informatie is aanwezig in boeken, emails, chatlogs, websites, databases, enzovoort.
- Sommige zijn uitgebreid en complex, andere zijn simpel en rechttoe rechtaan.
- Online bronnen zijn sneller toegankelijk, maar een boek lenen kost tijd.

7 Afstemming in Communicatie

Communicatie lijkt vaak simpel: iemand zegt iets, iemand anders begrijpt het. In de praktijk is het zelden zo rechtlijnig. Woorden en symbolen krijgen pas betekenis binnen een gedeelde context en binnen het sociale spel dat gesprekspartners samen spelen. Afstemming is het geheel aan processen dat ervoor zorgt dat dit spel soepel verloopt.

7.1 Beperkingen van klassieke modellen

Het klassieke zender-ontvanger model is nuttig maar beperkt. Het verklaart niet waarom gesprekken met correcte woorden toch mislopen, of waarom minimale uitwisselingen soms perfect werken. Het verschil zit in de mate waarin gesprekspartners op elkaar zijn afgestemd.

7.2 Drie pijlers van afstemming

1. Achtergrondinformatie

- Gesprekspartners hebben kennis nodig van relevante objecten, relaties, aannames en spelregels
- Deze informatie is grotendeels impliciet en blijft onuitgesproken
- Veel wordt verondersteld: niemand hoeft uit te leggen dat je met griep thuisblijft

2. Activatie

- Achtergrondinformatie moet op het juiste moment beschikbaar zijn
- Dit gebeurt via aandacht, taalkeuze, framing en focussering
- Voorbeeld: een instructietekst wordt pas begrijpelijk als je weet dat het over vliegeren gaat

3. Common ground & coördinatie

- *Common ground*: informatie waarvan beide partijen weten dat ze die delen, en weten dat de ander dat ook weet
- Dit is recursief, dynamisch en altijd onvolledig
- Coördinatie via bevestigen, verifiëren en corrigeren houdt de gemeenschappelijke basis op orde
- Uitingen als "bedoel je dat?" zijn geen beleefdheden maar cruciale mechanismen

7.3 Wanneer het misgaat

Verkeerde aannames over wat de ander weet kunnen ernstige gevolgen hebben, bijvoorbeeld in:

- Luchtvaart, zorg en helpdesks
- Ontwerp van interfaces (stereotiepe aannames over gebruikers)
- Hiërarchische situaties (niet durven vragen of corrigeren)

Alles hierboven is voor de midterm, alles hieronder is voor de final

8 visualisatietechnieken

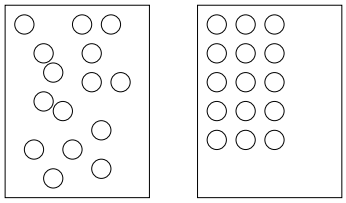
8.1 Definitie

“Een visuele representatie van gegevens om mensen sneller taken uit te kunnen laten voeren” & “punten, lijnen, coordinaten, kleuren, vormen en texturen gebruiken om data te representeren”

Je kan denken aan bijvoorbeeld (lijn-, staaf-, cirkel- en spreidings-)diagrammen, maar ook aan meer complexe visualisaties zoals heatmaps, boomstructuren en netwerkdiagrammen. Dit kan je dan allemaal gebruiken om dashboards, rapporten en presentaties (overzichtelijker) te maken.

Informatievisualisatie is **niet** gestructureerde data. Wetenschapelijke visualisatie is **wel** gestructureerde data. Dit zijn allebij onderdelen van het grote Data Visualisatie veld.

Het is een belangrijk, omdat het helpt om patronen, trends en inzichten te ontdekken die anders moeilijk te zien zouden zijn in grote datasets. Ook kan je door middel van verschillende visualisaties van dezelfde data verschillende inzichten krijgen.



Figuur 1: Slechte en goede manier om cirkel hoeveelheid te visualiseren

Zo is het makkelijker om bij figuur 2 te zien hoeveel cirkels er zijn, terwijl je bij figuur 1 veel langer bezig bent (en makkelijker telfouten maakt).

8.2 geschiendenis

(Geen idee of het nuttig is)
In 1786 maakte William Playfair de eerste grafieken. In 1854 gebruikte John Snow een kaart om een cholera-uitbraak in Londen te analyseren. In 1958 ontwikkelde Florence Nightingale de coxcomb chart om sterfteoorzaken in het leger te visualiseren. In 1869 introduceerde Charles Minard de beroemde kaart van Napoleon’s Russische veldtocht. In 1933 maakte Harry Beck de eerste metrokaart van Londen.

8.3 Algemene visualisatie principes

Er zijn een hele boel dingen die je kan doen om je visualisaties beter te maken

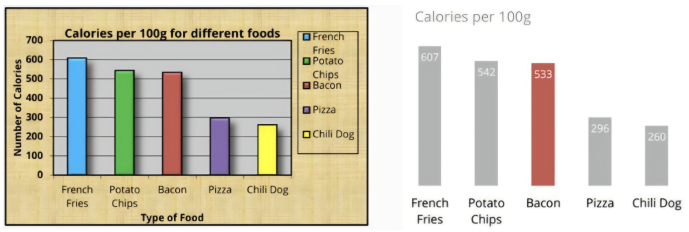
Vermijd onnodige 3D-effecten. Dit maakt het moeilijker om data te interpreteren én door diepte printen gaat er informatie verloren/worden de verhoudingen vertekend. **Pie charts werken vaak niet goed**. Het is lastig om oppervlakte te vergelijken, vooral bij meerdere segmenten. Om deze reden moet je **oppervlakte vergelijken vermijden**.

Edward Tufte heeft een aantal principes opgesteld voor effectieve informatievisualisatie:

- **Grafische integriteit**: Zorg ervoor dat de visualisatie de data nauwkeurig en eerlijk weergeeft, zonder misleiding.
- **Designesthetiek**: Maak de visualisatie visueel aantrekkelijk en gemakkelijk te begrijpen.

Hij zegt bij Designesthetiek:

(1) Laat vooral de data zien, geen poespas.
(2) Maximaliseer redelijkerwijs de **data-ink ratio** (de hoeveelheid inkt die daadwerkelijk data representeert versus de totale hoeveelheid inkt gebruikt in de visualisatie). $Data\text{-}ink\ ratio = \frac{Data\ ink}{Total\ ink}$
(3) Verwijder inkt die geen data representeert (zoals overbodige lijnen, kaders en decoraties).
(4) Kijk en doe opnieuw (iteratief proces).



Figuur 2: Slechte en goede visualisatie

Hier zie je een voorbeeld van een slechte en een goede visualisatie. De slechte visualisatie heeft veel onnodige elementen die de data verbergen en heel veel poespas toevoegen. De goede visualisatie is veel eenvoudiger en laat de data duidelijker zien. Bij de linker is de data-ink ratio heel laag, terwijl die bij de rechter hoog is.

8.4 Mark en channels

8.4.1 *Marks* zijn de basisvormen die worden gebruikt om data te representeren in een visualisatie. De meest voorkomende marks zijn punten, lijnen en vlakken. Deze marks kunnen worden gecombineerd en aangepast om verschillende soorten visualisaties te creëren, zoals scatterplots, lijndiagrammen en staafdiagrammen.

Marks kunnen punten zijn (zoals in een scatterplot), lijnen (zoals in een lijndiagram) of vlakken (zoals in een staafdiagram). Ook kan je Marks as Links gebruiken, dit zijn lijnen die twee of meer punten verbinden om relaties tussen data-elementen te tonen (zoals in een netwerkdiagram).

Marktypes zijn voor hoeveel Dimensies:

- 0D: punten (groote kan variëren, alleen locatie is relevant)
- 1D: lijnen (lengte kan variëren, locatie en oriëntatie zijn relevant)
- 2D: vlakken (oppervlakte kan variëren, locatie, oriëntatie en vorm zijn relevant)
- 3D: volumes (volume kan variëren, locatie, oriëntatie en vorm zijn relevant)

8.4.2 *Channels* zijn de visuele eigenschappen die worden gebruikt om marks te coderen en informatie over te brengen. Veelvoorkomende channels zijn positie, kleur, vorm, draaiing, grootte, oppervlakte, volume en textuur. Deze channels kunnen worden aangepast om verschillende aspecten van de data te benadrukken en om patronen en trends te identificeren. Bijvoorbeeld, in een scatterplot kan de positie van de punten de waarden van twee variabelen representeren, terwijl de kleur van de punten een derde variabele kan aangeven.

Dit is de lijst van beste naar slechtste channels (voor georganiseerde data):

(1) Positie op een gemeenschappelijke schaal
(2) Positie op niet-gemeenschappelijke schalen
(3) Lengte
(4) Hoek/oriëntatie
(5) Oppervlakte (2D)
(6) diepte (3D)
(7) Kleurintensiteit
(8) kleursaturatie
(9) ronding
(10) volume

Dit is de lijst voor categorische data:

(1) relatieve positie
(2) kleur/hue
(3) beweging
(4) vorm

9 Theoretische fundamenten

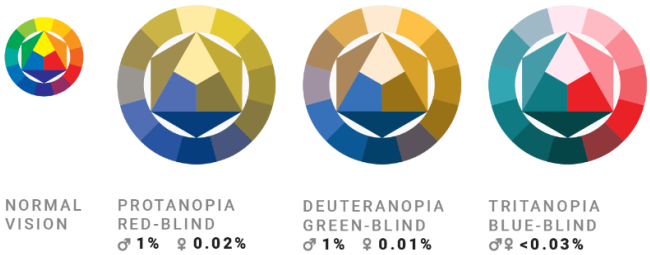
9.1 Visuele perceptie

Mensen zijn in sommige soorten perceptie erg goed, zoals het herkennen van patronen, kleuren en bewegingen. Maar op sommige erg slecht, zo zijn we niet goed in het inschatten van oppervlakten, volumes en hoeken.

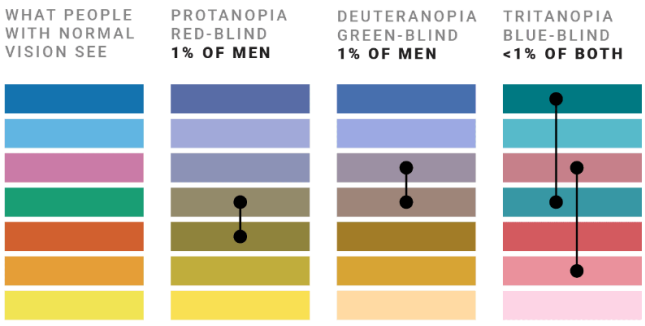
Dit kan je ook in je voordeel gebruiken. Zo kan je gebruik maken van “popout” Dit is wanneer een object zich onderscheidt van de rest door een uniek kenmerk, zoals kleur, vorm of grootte. Hierdoor valt het object direct op en trekt het de aandacht van de kijker. Bijvoorbeeld, in een afbeelding met allemaal blauwe cirkels, zal een rode cirkel direct opvallen en de aandacht trekken.

9.2 kleurgebruik

Kleurgebruik is belangrijk om voor iedereen de visualisatie goed te kunnen begrijpen.



Figuur 3



Figuur 4

Hier zie je heel goed dat er mensen zijn die rood en groen niet goed kunnen onderscheiden. Daarom is het belangrijk om kleuren te kiezen die voor iedereen goed te onderscheiden zijn. Hiervoor zijn verschillende kleurpalleten voor om goede kleuren te kiezen.

Je kan verschillende soorten oplossingen gebruiken om kleurenblindheid te omzeilen, zoals:

- Gebruik patronen of texturen in plaats van alleen kleuren.

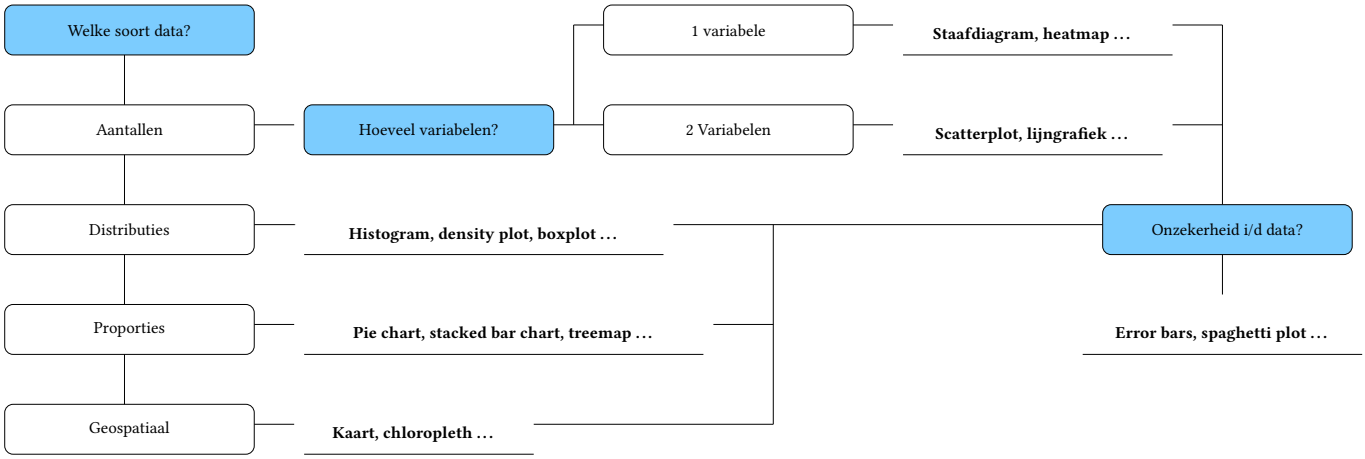
- Gebruik symbolen of vormen om verschillende categorieën aan te duiden.
- Gebruik kleuren die goed te onderscheiden zijn voor mensen met kleurenblindheid, zoals blauw en geel.

Simultaan contrast: Dit is een optische illusie waarbij de kleur van een object wordt beïnvloed door de kleuren eromheen. Bijvoorbeeld, een grijs vlak kan er lichter of donkerder uitzien afhankelijk van de achtergrondkleur. Dit kan leiden tot misinterpretaties van de data als de kleuren niet goed gekozen zijn.

Chromostereopsis: Dit is een optische illusie waarbij kleuren op verschillende afstanden lijken te liggen. Bijvoorbeeld, rode objecten lijken dichterbij te zijn dan blauwe objecten.

9.3 Klassieke visualisatie-types

Welke visualisatie moet je gebruiken?



10 Het geneste model

Tamara Munzner heeft een model om visualisaties te ontwerpen én te valideren. Het komt er op neer dat je vier niveaus hebt waarop je fouten kan maken, en dat je op elk niveau moet valideren of je ontwerp klopt. Dit moet je doen voordat je verder gaat naar het volgende niveau. Doe je dit niet, dan gaat het altijd mis op volgende niveaus.

Het model bestaat uit vier geneste niveaus. Elk niveau levert input aan het volgende, en elk niveau kent zijn eigen typische fouten en bijbehorende manieren om die fouten te toetsen.

10.1 De vier niveaus

(1) **Domeinprobleem en data** Eerst moet je begrijpen wat gebruikers daadwerkelijk doen en welke data daarbij hoort, in hun eigen taal en context. Het risico hier is simpel: je lost een probleem op dat zij helemaal niet hebben. Validatie gebeurt door observatie, interviews en veldstudies.

(2) **Abstractie naar operaties en datatypen** Vervolgens vertaal je het domeinprobleem naar generieke operaties (zoals vergelijken, filteren, clusteren) en datatypen (tabellen, grafen, tijdreeksen). Dit is vaak het zwakste punt in visualisatie-ontwerp. De klassieke fout: je kiest een verkeerde abstractie, waardoor de visualisatie per definitie niet helpt. Dit kun je alleen valideren door echte gebruikers hun eigen werk te laten doen met het systeem.

(3) **Visuele encodering en interactie** Pas hier beslis je hoe iets eruitziet en hoe gebruikers ermee werken. De vraag is niet of het mooi is, maar of het de gekozen abstractie effectief overbrengt. Validatie kan via ontwerp-argumentatie, perceptuele principes, labstudies (tijd en fouten), en analyse van resultaatbeelden.

(4) **Algoritmes** Het binnenste niveau gaat over performance en correctheid: is het snel genoeg, schaalbaar, en technisch juist? Validatie gebeurt via complexiteitsanalyse en metingen van tijd en geheugen.

10.2 Kernidee: fouten werken door

De niveaus zijn genest, niet onafhankelijk. Een fout hogerop besmet alles eronder:

- een verkeerd probleem → niemand gebruikt je tool;
- een verkeerde abstractie → je laat gebruikers het verkeerde zien;
- een slechte encodering → mensen begrijpen het niet;
- een traag algoritme → het systeem is onbruikbaar.

Daarom is validatie op één niveau nooit genoeg. Veel vormen van validatie zijn per definitie *downstream*: je kunt een hoger niveau pas echt toetsen als de lagere niveaus al werken.

10.3 Belangrijkste aanbevelingen

Munzner doet drie expliciete aanbevelingen:

(1) Maak duidelijk *op welk niveau* je bijdrage zit, zeker als een paper meerdere niveaus raakt.
(2) Benoem expliciet welke aannames je maakt over hogere niveaus die je niet onderzoekt.
(3) Accepteer en waardeer meer papers die zich uitsluitend richten op domein- en probleemkarakterisatie.

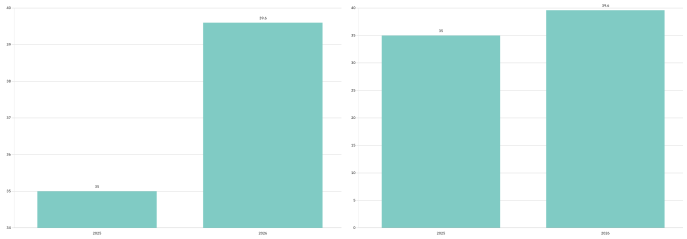
De onderliggende boodschap is streng maar terecht: veel zwakke visualisatiepapers falen niet omdat de techniek slecht is, maar omdat nooit scherp is vastgesteld welk probleem eigenlijk wordt opgelost en waarom de gekozen abstractie klopt.

11 misleiden met visualisaties

Het is heel makkelijk om mensen te misleiden met visualisaties. Bijvoorbeeld door de y-as niet bij 0 te laten beginnen, of door 3D-effecten te gebruiken die de verhoudingen vertekend weergeven. Ook kan je door kleuren en vormen te gebruiken die de aandacht afleiden van de data zelf.

11.1 Slecht ontwerp

Als een grafiek slecht ontworpen is, dan kan het (onbewust) misleidend zijn. Bijvoorbeeld door onduidelijke labels, onjuiste schalen of onlogische indelingen. 3D-effecten kunnen ook de verhoudingen vertekend weergeven, waardoor de data anders lijkt dan het is.



Figuur 5

Links ziet er veel dramatischer uit dan rechts, terwijl de data hetzelfde is. Dit komt doordat de y-as bij links niet bij 0 begint, waardoor de verschillen groter lijken dan ze zijn.

11.2 Dubieuze data

Als de data zelf al dubieus is, dan kan de visualisatie ook misleidend zijn. Bijvoorbeeld door selectieve dataweergave, waarbij alleen de data wordt getoond die het gewenste verhaal ondersteunt. Of door het gebruik van onjuiste statistische methoden die de data verkeerd interpreteren.

11.3 Te weinig data

Als er te weinig data is, dan kan de visualisatie ook misleidend zijn. Bijvoorbeeld door het gebruik van kleine steekproeven die niet representatief zijn voor de populatie. Of door het gebruik van gemiddelden die de variabiliteit in de data verbergen.

11.4 verborgen of onduidelijke onzekerheid

Als de onzekerheid in de data niet duidelijk wordt weergegeven, dan kan de visualisatie ook misleidend zijn. Bijvoorbeeld door het weglaten van foutenmarges of betrouwbaarheidsintervallen. Of door het gebruik van grafieken die de onzekerheid verbergen, zoals lijngrafieken zonder foutbalken.

11.5 suggestieve patronen

Data kan soms patronen lijken te tonen die er in werkelijkheid niet zijn. Bijvoorbeeld door een grafiek te tonen die levensverwachting tegenover sigarettenconsumptie per land weergeeft. Dit kan de suggestie wekken dat er een verband is, terwijl er in werkelijkheid veel andere factoren meespelen. De visualisatie toont dan dat je langer leeft als je meer rookt. Dit komt, omdat rijke landen zowel een hogere levensverwachting als een hogere sigarettenconsumptie hebben.

Ook kan je door toeval patronen zien die er niet zijn. Grappige voorbeelden hiervan kan je op de website https://www.tylervigen.com/spurious-correlations vinden.

12 interactieve visualisaties

Interactieve visualisaties zijn in sommige gevallen betern om:

- Grote datasets te verkennen
- Visueel verhaal te vertellen
- Data te verkennen zonder verhaal
- Mensen betrokkener maken bij de data

Er zijn verschillende soorten interacties die je kan gebruiken in interactieve visualisaties:

- **Exploreren** (zoomen, pannen, filteren) maakt het mogelijk om door de data te navigeren. Dit kan bijvoorbeeld door in- en uit te zoomen op een kaart, of door te pannen door een grote dataset.
- **Connecteren** (selecteren, highlighten) maakt het mogelijk om relaties tussen verschillende data-elementen te zien. Bijvoorbeeld door het highlighten van gerelateerde punten in een scatterplot als je een punt selecteert.
- **Filteren** (data subsetten) stelt gebruikers in staat om alleen de data te zien die relevant is voor hun analyse. Dit kan bijvoorbeeld door middel van dropdown-menu's of schuifregelaars. Je kan dan in een dataset of leeftijd van inwoners in Utrecht filteren op mannen of vrouwen, of mensen tussen de 20 en 30 jaar.
- **Configureren** (visualisatie aanpassen) laat gebruikers de visualisatie aanpassen aan hun voorkeuren. Dit kan bijvoorbeeld door het wijzigen van kleuren, vormen of groottes van de marks. kort: je kan de visualisatie personaliseren.
- **Encoderen** (data representatie aanpassen) maakt het mogelijk om de manier waarop data wordt weergegeven te veranderen. Bijvoorbeeld door het wisselen tussen een staafdiagram en een lijngrafiek. Of een scatterplot naar een dotplot.
- **Samenvatten/detailleren** (niveau van detail aanpassen) stelt gebruikers in staat om het niveau van detail in de visualisatie aan te passen. Dit kan bijvoorbeeld door het in- en uitzoomen op een kaart, of door het tonen van meer of minder gegevenspunten.
- **Selecteren** (items kiezen) maakt het mogelijk om specifieke data-elementen te kiezen voor verdere analyse. Dit kan bijvoorbeeld door het klikken op een punt in een scatterplot om meer informatie te zien over dat punt.