Kris Merckx

CONCEPTS OF DATA & ANALYTICS

Small data, Big data, Smart devices en AI

Kris Merckx 2019

Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International

In dit boek leer je over data. Hoe bewaren en organiseren we digitale data? Hoe doorzoek je data? Hoe kan je data verzamelen?

Meer nog, hoe kan je dit werk automatiseren eventueel met de hulp van kunstmatige intelligentie. Uiteraard bekijken we ook hoe kunstmatige intelligentie werkt en in staat is tot voorspellingen.

In andere vakken leer je hoe je data visualiseert… Maar eerst probeer ik je wat wegwijs de maken in de verwarrende wereld van (big) data en kunstmatige intelligentie.

Kris Merckx

1995. II
1994. II
1992 II
2003 II

Hoofdstuk 1

TEKSTBESTANDEN & BINAIRE BESTANDEN

- **Wat zijn tekstbestanden?**
- **Wat is het verschil tussen tekstbestanden en binaire bestanden?**
- **Waarom kunnen we zeggen dat de meeste bestandstypes ongestructureerde data bevatten?**

Sorry, je begrijpt het verkeerd. Met tekstbestanden bedoelen we in dit geval niet de documenten die je intikt in MS Word of LibreOffice. Uiteraard hebben ze wel wat elkaar gemeen. Een bestand uit MS Word is een tekstverwerkingsbestand. Als je het bewaart op je harde schijf, wordt het een binair en/of een gecomprimeerd bestand.

Als we het over tekstbestanden hebben in de titel, dan doelen we op “platte tekst”-bestanden. Bestanden die enkel karakters bevatten die je op een toetsenbord kan intikken, maar geen opmaak of afbeeldingen.

Als data-expert in spe is het van bijzonder groot belang dat je weet wat we met platte tekst bedoelen. Je zal als data-analist sowieso in teksteditors met platte tekstbestanden werken. Wat dit precies is, lees je zo meteen…

ASCII

De computer bewaart alle bestanden binair, als een reeks nullen en enen, want dat is de enige code die de meeste moderne computers verstaan. Elk bestand bestaat uit een vaak erg lange, maar toch eindige reeks nullen en enen. De computer bewaart ze op een opslagmedium (harde schijf, USB-stick…).

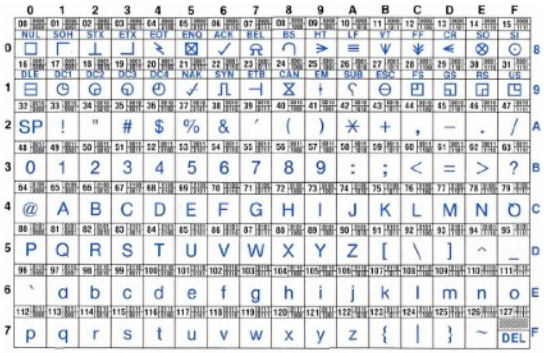
Terwijl ik deze zin intik op mijn toetsenbord, vertaalt de computer de toetsaanslagen in nullen en enen, in bits die op hun beurt gegroepeerd zijn in bytes. Eén byte is een "groepje" van acht bits, de letter a vertaalt de computer bijvoorbeeld als de byte 01100001. In decimale getallen is dit 97. Elke toets en ook combinatie van toetsen (bijvoorbeeld SHIFT + A voor de hoofdletter A) krijgt zo een byte toegewezen. Vermits één byte uit 8 bits bestaat, beschikken we over 2^8 combinaties (2 tot de achtste macht), want elke bit kan ofwel 1 ofwel 0 zijn.

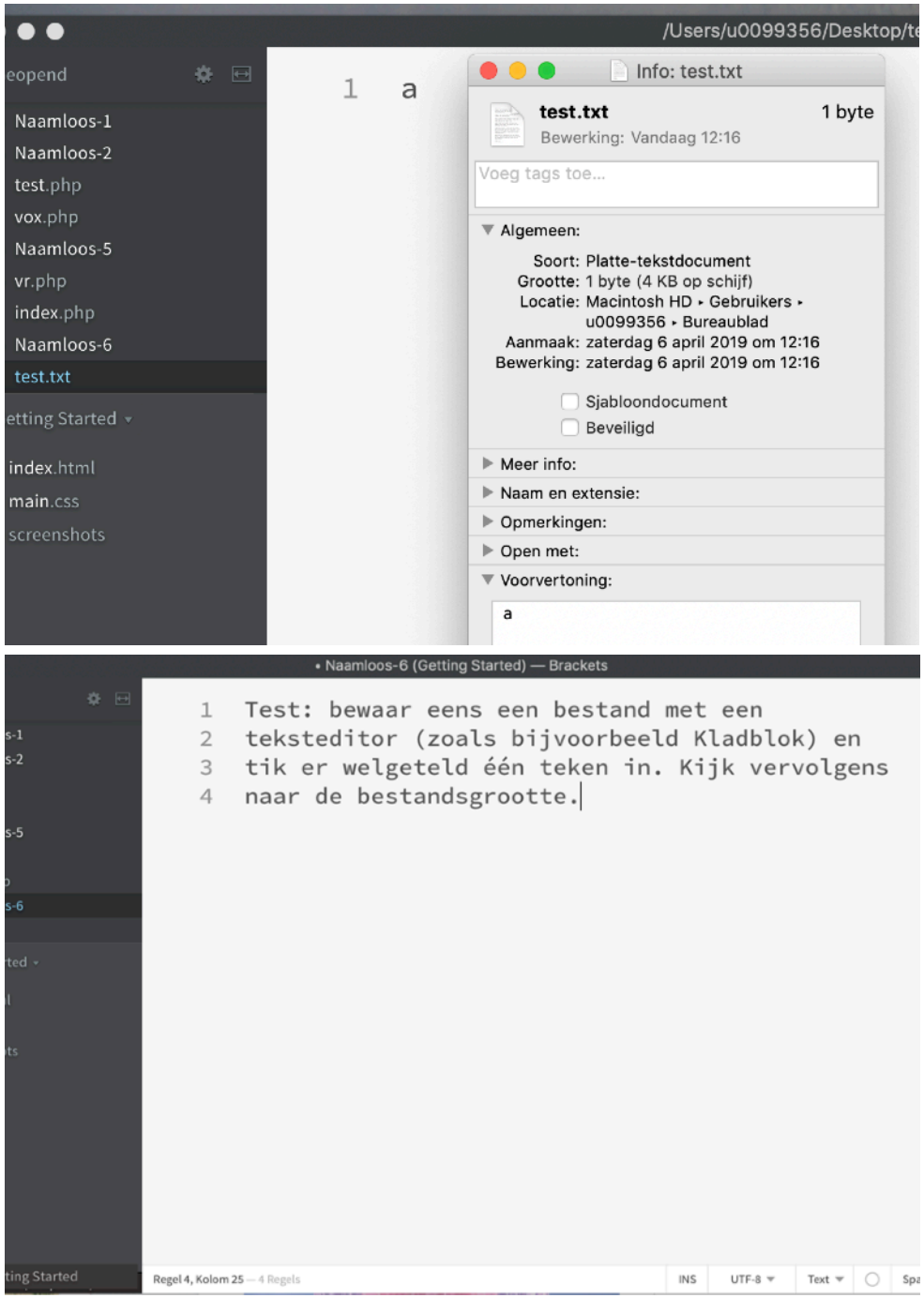
In totaal kan je op die manier 256 combinaties vormen, of alle combinaties tussen

```
00000000 = 0 en 11111111 = 255
```

Hiervoor zijn natuurlijk afspraken gemaakt. In principe kan je om het even welk toetsenbord op om het even welk digitaal systeem dat tekstinvoer toelaat, aansluiten. De toetsaanslagen worden automatisch herkend. Dit komt omdat ze zo goed als allemaal ASCII ondersteunen. **ASCII** (American Standard Code for Information Interchange) is een standaard om Latijnse letters, cijfers, leestekens en een aantal andere tekens en een reeks stuurcodes te koppelen aan een getal.

De ASCII-tabel is een tweedimensionale rij of tabel. Zoek het gewenste teken en neem de binaire code die in de kop van de rij is aangeduid. Plak daarachter vervolgens de inhoud van de kolomkop. Een spatie bestaat dus uit 0010 gevolgd door 0000 of samengevoegd 00100000, wat overeenkomt met het decimale getal 32.





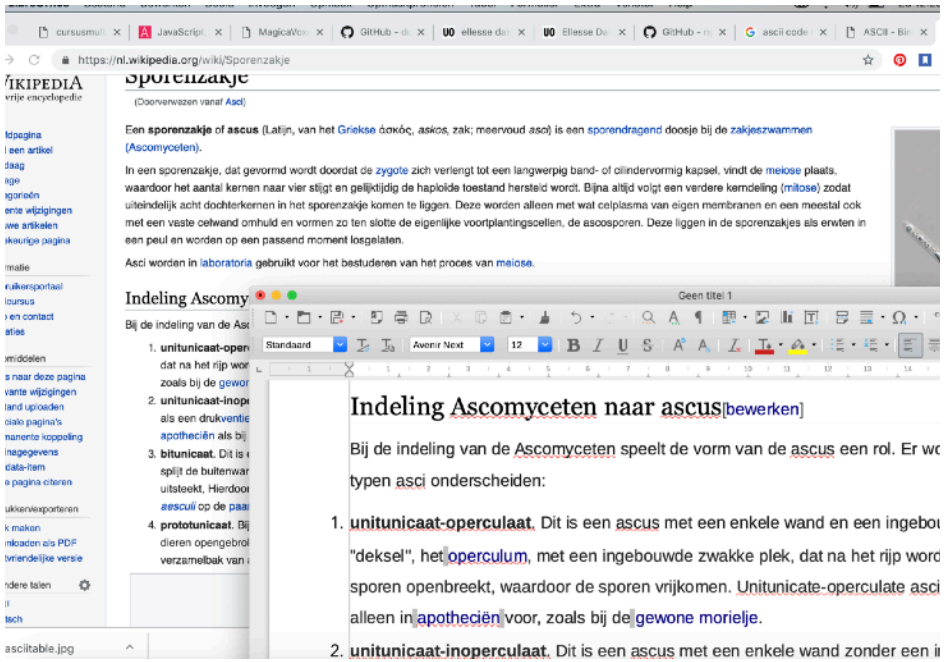
Test: bewaar eens een bestand met een teksteditor (zoals bijvoorbeeld Kladblok) en tik er welgeteld één teken in. Kijk vervolgens naar de bestandsgrootte (zie afbeelding links).

In een teksteditor kan je enkel ASCII-tekens (of UTF-8-karakters, waarover later meer) invoeren. Je kan er je tekst niet mee opmaken of er bijvoorbeeld foto's aan toevoegen. Ik hoor je al denken: *wat is daarvan het nut?*

Het grote voordeel is dat dit soort bestanden perfect uitwisselbaar zijn. Je hebt er geen speciale software voor nodig om de inhoud van die documenten te kunnen lezen. Een eenvoudige editor zoals Windows Kladblok of Adobe Brackets volstaan om het bestand te openen. Programmeercode of webpagina's kan je perfect in een editor openen, omdat ze enkel karakters kunnen opnemen die ondersteund worden door de ASCII- of UTF-standaarden.

Afbeelding: een tekst in de editor Adobe Brackets.

Je hebt het misschien al meegemaakt. Je kopieert een tekst van Wikipedia en je plakt het in een MS Worddocument. Ook alle links die in de webpagina zaten en de opmaak, zijn mee gekopieerd. Wil je dit vermijden, dan plak je de tekst eerst in een teksteditor. Op die manier heb je enkel de “platte tekst”.



Als je een tekst van een webpagina kopieert en plakt in een tekstverwerker, kopieer je ook heel de opmaak mee.

UNICODE en UTF-8

ASCII werkt zoals je merkt perfect, maar pakweg Arabieren, Chinezen en Japanners, kortom iedereen die geen standaard (indo)Europees georiënteerde taal schrijft, bleef met ASCII wel aardig op zijn honger zitten. Er bestonden geen binaire codes voor het Arabisch of Chinees (een Chinese Jan met de pet kent ongeveer 7000 verschillende karakters). Het recente Hanyu Cidian-Chinees beslaat zelfs 56.000 karakters! Uitbreidingen op de de ASCII-tekenset waren noodzakelijk. De noodzaak om alle andere taalsystemen en codes op te nemen leidde tot UNICODE.

De **UNICODE**-tekenset maakt het gebruik van alle mogelijke denkbare alfabetten en schriftsystemen in één document mogelijk. Niet dat je dit vaak nodig hebt, want je schrijft of tikt niet alle dagen in het Pau Cin Hau, het Bamum of Sharada, maar het gebruik van dingbats, wiskundige symbolen enz. kom je al wat vaker tegen. UNICODE is een ISO-standaard.

De meest bekende tekenset van UNICODE is **UTF-8** (8 bit Unicode Transformation Format). Moderne webpagina's zijn volgens deze norm gecodeerd waardoor ze om het even welk symbool standaard kunnen weergeven.

Binaire bestanden

Ook al bieden ASCII-bestanden heel wat voordelen, toch is het niet altijd de beste manier om bestanden te bewaren.

Wanneer men het over BINAIRE bestanden heeft, dan gaat het over bestanden die niet volgens de ASCII-tekenset zijn gecodeerd. *Er bestaan meerdere goede redenen om dit niet te doen.*

Voor het bewaren van de meeste bestanden zoals foto's, documenten uit een tekstverwerker, presentaties, films… gebruikt men een "binaire codering". ASCII zou in dat geval leiden tot te grote bestanden.

Heel vaak gebruikt men bovendien compressietechnieken om bestanden een stuk kleiner te maken zodat ze minder ruimte in beslag nemen. Denk maar eens hoe snel je smartphone vol is, als je alle dagen foto's of filmpjes maakt.

Waarom geen ASCII?

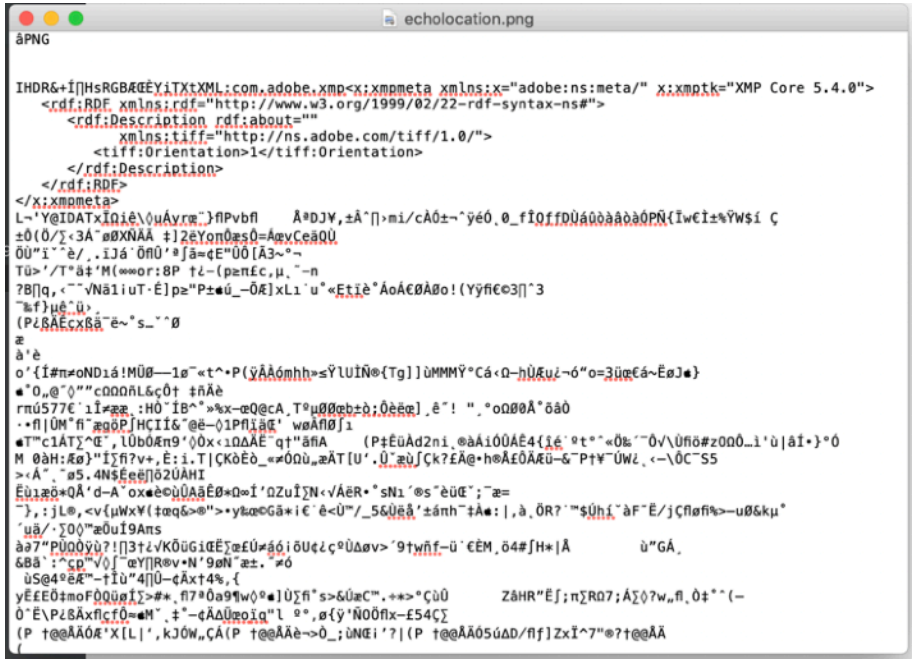
Een voorbeeld: een letter “a” (niet de hoofdletter) op je toetsenbord vertaalt zich als de byte 01100001. Eén teken wordt vervangen door een reeks van 8 nullen en/of enen. Een computer gebruikt in dit geval dus 8 bits waar een mens slechts één teken nodig heeft. Dit geldt ook voor de cijfers op je toetsenbord. Tik je een 0, dan wordt dit volgens de ASCII-regels 00110000 en een 1 wordt 00110001. Het getal 256 behandelt hij volgens de ASCII-tekenset als 3 afzonderlijke ASCII-karakters of 3 bytes en ziet er dan als volgt uit: 00110010 00110101 00110110.

Een nogal gekke manier van doen als we enkel getallen willen bewaren, want intern maakt ASCII reeds de vertaalslag naar decimale cijfers door aan elke toets of karakter een decimaal getal te koppelen.

Getallen kunnen veel eenvoudiger en compacter bewaard worden door ze niet als een reeks afzonderlijke karakters, maar als één geheel, als een echt getal te bewaren. 255 zal er dan uitzien als 1111 1111.

Eén enkele byte 256 kan verschillende waardes bevatten. Gaan we naar 4 bytes dan krijgen we 4 miljard mogelijke combinaties. Het getal 4000000000 (4 miljard) zou op die manier slechts 4 bytes in beslag nemen. Bewaren we het als ASCII, dan zou het 10 bytes aan ruimte oppeuzelen.

Om die reden gebruikt men heel wat vormen van codering. Het zou ronduit dom zijn om bijvoorbeeld een afbeelding als een reeks ASCII-waardes (weliswaar binair gecodeerd) te bewaren. Een afbeelding van 1024 bij 768 pixels telt in totaal 786432 pixels. Vermits elke pixel bestaat uit een mix van 256 waardes rood, 256 waardes groen en 256 waardes blauw, dan zou het gecodeerd volgens de ASCII-tekenset onwaarschijnlijk veel ruimte in beslag nemen.



Een binair bestand (PNG-afbeelding) in een teksteditor.

Kennen, begrijpen

1. Beschrijf kort en duidelijk wat ASCII-bestanden zijn.
2. Leg de noodzaak van UNICODE uit.
3. Leg de begrippen ASCII, UNICODE, UTF-8 uit.
4. Leg uit waarom tekstcodering niet altijd de meest geschikte opslagtechniek is.
5. Wanneer spreekt men van binaire bestanden?

Kunnen

1. Bestanden kunnen indelen in binaire of tekstbestanden

YOUTUBE
POPULAR
FRIENDS
WORDPRESS
COMMUNICATION
GOOGLE+
FACEBOOK
MEDIA
WEB
MOBILE
FLICKR
SEARCH
GLOBAL
COMMUNITY
INFORMATION
PEOPLE
LINKEDIN
PINTEREST
SHARE
CONTENT
INTERNET
MONEY
NETWORK
ONLINE
LIKE
RADIO
SOCIAL
DATA
NEWS

Hoofdstuk 2

GESTRUCTUREERDE EN ONGESTRUCTUREERDE DATA

- **Wat is het verschil tussen gestructureerde en ongestructureerde data?**
- **Hoe kan je data structureren?**
- **Wanneer zijn data semantisch?**

De meeste data en bestanden bevatten voor computers enigszins ongestructureerde data. Dat klinkt vreemd als je beseft dat de digitale data nu eenmaal voor en door computers zijn opgesteld.

Dit betekent echter nog niet dat de computer ook de inhoud van die data snapt. Een foto bijvoorbeeld wordt door een computer opgebouwd door een raster van kolommen en rijen te vullen met kleurinformatie. Een pixel (picture element) is een vakje in zo'n raster en bevat voor een computer drie cijfergegevens: een kleurwaarde voor rood, een waarde voor groen en een waarde voor blauw. Een computer zal daarnaast aan elke pixel een x- en y-positie in het raster toekennen en onthoudt aldus voor die pixel vijf cijfergegevens.

Dat is dus in zekere zin gestructureerd, maar dat vertelt nog niets over de inhoud van de afbeelding. Staat er een koe, een appel of een flesje bier op de foto? Dat vertellen de data niet in dit geval. Je zou die informatie kunnen toevoegen waardoor de "afbeeldingsbron" niet meer enkel pixelwaarden en posities bevat, maar ook "labels".

Ook voor teksten is dit het geval. Elke letter in een tekst wordt bewaard als een cijfercode, maar de computer weet niets over de inhoud van de tekst.

Gestructureerde data

Wat bedoelen we dan eigenlijk met gestructureerde gegevens? Het gaat om

- gegevens met heldere patronen die een machine/ computer makkelijk kan herkennen. Bijvoorbeeld: *een lijst met rijen en kolommen, creditcardnummers, postcodes,*
- gestructureerde mededelingen op overschrijvingen… Bijvoorbeeld: gegevens met sleutel en waarde-paren.
- gegevens met sleutel en waarde-paren. Bijvoorbeeld een JSON-bestand

Ongestructureerde data

Ongestructureerde data bieden weinig of geen houvast over de inhoud van de data of de “semantiek”.

- Gegevens die niet in een tabel staan met een herkenbaar aantal rijen en kolommen of sleutels met hun waarde.
- Gegevens met minder heldere patronen voor computersystemen (dat wil niet zeggen dat er helemaal geen patronen in zitten)
- Bijvoorbeeld: tekstdocumenten, webpagina’s, presentaties…
- Bijvoorbeeld: Afbeeldingen, audio, video…

Types gestructureerde data

Doorheen de tijd hebben zich een aantal standaarden ontwikkeld voor het structureren van tekstdata. Markeertalen zoals HTML (hypertext markup language) en XML (extensible markup language) markeren elementen in een tekstbestand door aan te duiden waar een onderdeel begint en waar één eindigt. Je markeert op die manier de diverse onderdelen van je data. Op die manier kan software sneller terugvinden waar een bepaald element staat.

Je kan tekstdata ook scheiden met leestekens, spaties of tabs. Een CSV-bestand scheidt rijen met regeleindes (harde returns), kolommen met een komma. YAML (YAML Ain't Markup Language) onderscheidt data met tabs en regeleindes. JSON (Javascript object notation) maakt gebruik van diverse leestekens (komma’s, aanhalingstekens…) en akkolades.

Een XML-bestand met informatie over boeken.

```xml
<?xml version="1.0"?>
<catalog>
   <book id="bk101">
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
      <price>44.95</price>
      <publish_date>2000-10-01</publish_date>
      <description>An in-depth look at creating applications
      with XML.</description>
   </book>
   <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <price>5.95</price>
      <publish_date>2000-12-16</publish_date>
      <description>A former architect battles corporate zombies,
      an evil sorceress, and her own childhood to become queen
      of the world.</description>
   </book>
</catalog>
```

Een YAML-bestand met gegevens over personeelsleden.

```yaml
# Employee records
-  martin:
    name: Martin D'vloper
    job: Developer
    skills:
      - python
      - perl
      - pascal
-  tabitha:
    name: Tabitha Bitumen
    job: Developer
    skills:
      - lisp
      - fortran
      - erlang
```

Een CSV-bestand met adresgegevens. De meeste rekenbladprogramma's zoals MS Excel kunnen CSV-data importeren en exporteren. Het is een veelgebruikt formaat voor data-uitwisseling.

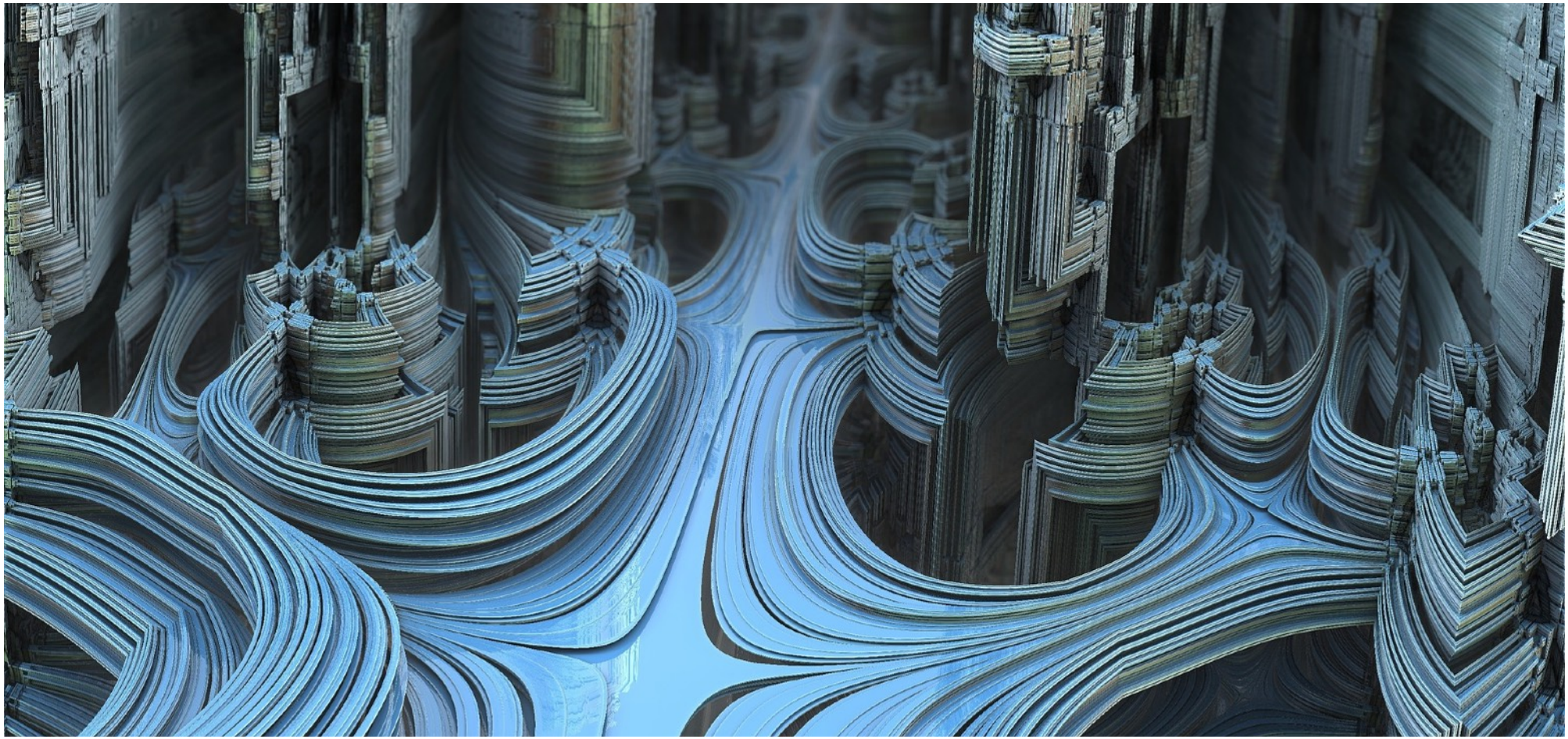
```
Naam,Functie ,Telnr.,Mob,E-mail,Web,Aantallen,,
Jeroen Trogh,Head Of Operations Academy,+32 (0) 16 22 85 08,+32 (0) 485 84
80 32,jeroen.trogh@ohl.be,www.ohleuven.com,40,,
Ben Geukens,Head Of Operations Academy,+32 (0) 16 22 85 08,+32 (0) 478 18
87 54,ben.geukens@ohl.be,www.ohleuven.com,40,,
Henk Mariman,Academy Manager,+32 (0) 16 22 85 08,+32 (0) 475 58 41
26,henk.mariman@ohl.be,www.ohleuven.com,40,,
Gauthier Vanderloock,Video & Data Analyst U18-U21,+32 (0) 16 22 85 08,+32
(0) 490 39 62 90,gauthier.vanderloock@ohl.be,www.ohleuven.com,40,,
```

Een ICAL-bestand voor het uitwisselen van kalendergegevens.

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ZContent.net//Zap Calendar 1.0//EN
CALSCALE:GREGORIAN
METHOD:PUBLISH
BEGIN:VEVENT
SUMMARY:Abraham Lincoln
UID:c7614cff-3549-4a00-9152-d25cc1fe077d
SEQUENCE:0
STATUS:CONFIRMED
TRANSP:TRANSPARENT
RRULE:FREQ=YEARLY;INTERVAL=1;BYMONTH=2;BYMONTHDAY=12
DTSTART:20080212
DTEND:20080213
DTSTAMP:20150421T141403
CATEGORIES:U.S. Presidents,Civil War People
LOCATION:Hodgenville\, Kentucky
GEO:37.5739497;-85.7399606
DESCRIPTION:Born February 12\, 1809\nSixteenth President (1861-1865)\n\n\n
 \nhttp://AmericanHistoryCalendar.com
URL:http://americanhistorycalendar.com/peoplecalendar/1,328-abraham-lincol
 n
END:VEVENT
END:VCALENDAR
```

Een JSON-bestand volgens afgesproken schema's (www.schema.org)

```
{
    "@context": "http://schema.org/",
    "@type": "TravelAgency",
    "address": {
        "@type": "PostalAddress",
        "addressCountry": "Belgium",
        "addressLocality": "Leuven",
        "postalCode": "3000",
        "streetAddress": "Quinten Metsysplein 12",
        "email": "cultuurreizen@davidsfonds.be",
        "telephone": "003216310608"
    },
    "email": "cultuurreizen@davidsfonds.be"
}
```



Kennen, begrijpen

1. De belangrijkste kenmerken van gestructureerde gegevens kunnen uitleggen.

2. De belangrijkste kenmerken van ongestructureerde gegevens kunnen uitleggen.

3. De belangrijkste types gestructureerde data opsommen.

Kunnen

1. Bestandstypes herkennen: JSON, XML, HTML, YAML, CSV, ICAL…

2. Gestructureerde en ongestructureerde bestanden kunnen onderscheiden.

3. Wanneer je een bestand krijgt, het kunnen indelen volgens bovengenoemde categorieën.

Hoofdstuk 3

DATABANKEN

- **Wat zijn databanken?**
- **Wat is het verschil tussen databanken en andere computerbestanden?**
- **Hoe kan je informatie opslaan in databanken?**
- **Welke soorten databanken bestaan er?**

Een databank is een manier om data op een gestructureerde manier te bewaren. Dit maakt het mogelijk om gegevens snel terug te kunnen vinden.

Hoe werken databanken? Welke soorten databanken bestaan er? Hoe kan je met software databanken doorzoeken en gegevens?

Tabellen met relaties

Sorry als je dit altijd hebt gedacht, maar een spreadsheetprogramma zoals MS Excel is geen databanktoepassing, ook al gebruiken heel wat mensen zo'n software nu net met dat doel. In een databank bewaar je gestructureerde informatie door ze te bewaren in een of meerdere tabellen.

Ga je regelmatig winkelen in een zaak waar men ook klantenkaarten uitdeelt? Heel vaak zal men je bij een aankoop eerst vragen naar je postcode. Hieronder kom je te weten waarom dit zo is.

Een eenvoudige adressenlijst zouden de meeste mensen meteen in één tabel bewaren. Die zou er dan als volgt kunnen uitzien:

Naam	Voornaam	Postcode	Gemeente	Straat
Jansens	Jan	3300	Tienen	Bostsestraat 2
Pieters	Peter	3000	Leuven	Naamsestraat 13
Anders	Ann	3001	Heverlee	Leo Dartelaan 10
...	...	...	...	...

Vaak volstaat zo'n simpele databank niet. Ook in het geval van het bovenstaande voorbeeld kan het beter. Immers: als je adressenlijst aangroeit, moet je wellicht vaak dezelfde gemeente en postcode en dezelfde straatnaam invoeren. We zouden het bovenstaande voorbeeld op die manier in 3 tabellen kunnen onderbrengen.

1 Een tabel voor de gemeentes

ID	Gemeente	Postcode
1	Tienen	3300
2	Leuven	3000
3	Heverlee	3001
...	...	...

Je merkt dat één kolom de naam “ID” krijgt. Een “id” is een uniek nummer. Je kan ervoor zorgen dat de databank dit getal automatisch verhoogt als je een rij toevoegt. Als je een rij verwijdert, wijzigen niet alle andere nummers. Dat klinkt misschien raar, maar is vrij normaal. Je moet het zien als een “volgnummer”. Bij het wielrennen, moeten ook niet alle renners hun volgnummer wijzigen als er iemand onderweg afhaakt.

2 Een tabel voor de straten

Daarnaast maken we een afzonderlijke tabel voor de straatnamen. Elke gemeente telt meerdere straten, maar één straat ligt (over het algemeen) in één gemeente (dat klopt natuurlijk niet altijd).

De tabel bevat in dit geval 3 kolommen: een ID, het ID van de gemeente in de andere tabel, de straatnaam.

ID	Straatnaam	IDgemeente
1	Bostsestraat	1
2	Naamsestraat	2
3	Leo Dartelaan	3
4	Putstraat	1
5	Houbaertstraat	1
6	Bondgenotenlaan	2
…	…	…

De tabel van de gemeentes en de tabel van de straten hebben een innige relatie. Je kan nu snel data "filteren" en zoeken. Bijvoorbeeld: *geef me alle straten van "Tienen"*. In databanktermen komt dit op het volgende neer: selecteer alle velden in de tabel "straten" waar IDgemeente = 1.

Iets geavanceerder kan je nu zeggen: geef me alle straten van de gemeente met postcode 3300. Dat is de postcode van Tienen.

Naam	Voornaam	Huisnummer	IDstraat
Jansens	Jan	2	1
Pieters	Peter	13	2
Anders	Ann	10	3
…	…	…	…

3 Een tabel voor de contacten

De tabel voor de contacten kan nu wat vereenvoudigd worden.

Door de 3 tabellen aan elkaar te verbinden, kan je te weten komen in welke gemeente en in welke straat iemand woont. Immers: Jan Janssen woont in IDstraat 1. Dat ID vind je terug in de tabel met de straatnamen. Die tabel bevat het ID van de gemeente in de tabel "gemeentes".

Dit lijkt nogal omslachtig, maar zorgt er uiteindelijk voor dat je gegevens veel **gestructureerder** en **compacter** kan bewaren. Bovendien zijn op die manier alle gegevens **conform** en voorkom je verschillende notaties en fouten.

Databanken bewerken met CRUD

Een databank moet je uiteraard ook makkelijk kunnen bewerken. Je moet elementen aan de tabellen kunnen toevoegen, gegevens kunnen verwijderen, informatie kunnen aanpassen. In de meeste gevallen wil je de informatie uit de databank gewoon kunnen doorzoeken, filteren en ze "lezen".

C	Create	Duurzame opslag van gegevens. Gegevens mogen niet dubbel opgeslagen worden. Een ID moet bijvoorbeeld altijd behouden blijven. Als je een rij verwijdert, behouden de andere rijen hun oorspronkelijke ID. Anders zijn de "relaties" om zeep.
R	Read	Eenvoudig opzoeken en filteren van gegevens.
U	Update	Gegevens aanpassen in de databank.
D	Delete	Gegevens verwijderen uit de databank

Het begrip **CRUD** staat voor: Create, Read, Update, Delete.

Uiteraard moeten er op een databank **gebruiksrechten** worden ingesteld. Niet iedereen mag het recht hebben om gegevens te verwijderen of aan te passen, of zelfs toe te voegen. Toepassingen waar dit wel kan, zoals Wikipedia, bewaren nog steeds een bewerkingsgeschiedenis en hebben een controlemechanisme ingebouwd om misbruik of fouten te voorkomen.

Databanken spreken SQL

Om CRUD-commando's uit te voeren, ondersteunen de meeste databanken de taal **SQL (structured query language = gestructureerde opvraagtaal).**

Indien we de gegevens uit onze adressenlijst zouden filteren op straat, dan zou dit in SQL kunnen klinken als

*SELECT * FROM contacten WHERE IDstraat=1*

*DELETE * FROM straten WHERE IDgemeente=2*

Gegevenstypes

Last but not least… de kolommen of "velden" (een vakje in een tabel) in een databanktabel kunnen een bepaald soort data bevatten: sommige mogen enkel een geheel getal bevatten (een zogenaamde "int" of "integer"), andere een tekst ("string" met een in te stellen lengte), een datum…Die gegevenstypes zijn eveneens belangrijk bij het conform houden van de gegevens.

Relationele databanken

In het bovenstaande voorbeeld gaan we ervan uit dat één persoon slechts op één huisnummer woont. De straatnaam hoeven we niet in te vullen. We kunnen enkel verwijzen naar een unieke "id" (identiteit) die de straat krijgt in de tabel "straten". Elke straat kan slechts in één gemeente liggen. Ik hoor je al denken dat meerdere gemeentes een "Dorpsplein" hebben, maar dat doet in deze databank niet terzake. Vanaf het moment dat je Straat_ID=1 kiest, weet de databank dat Straat_ID=1 in Gemeente_ID=1 ligt. Op die manier kom je automatisch bij de juiste gemeente terecht.

In de gebruikersinterface vraagt men eerst om bijvoorbeeld de postcode. Eens je die heb ingevoerd of geselecteerd, kies je de gemeente. Vervolgens krijg je een lijst van alle straten in die gemeente. Wanneer je de straat kiest, krijg je een overzicht van al je adressen in die straat. Een bijkomend voordeel is dat wanneer een straat van naam verandert, de straatnaam automatisch een update krijgt in alle adressen.

Omdat databanken beschikken over technieken om verschillende tabellen aan elkaar te koppelen of in relatie te brengen (denk aan IDgemeente, IDstraat...) spreekt men in dit geval over **RELATIONELE DATABANKEN**.

Relationele databanken worden in bedrijven, webshops, diensten, websites, elektronische leeromgevingen... massaal gebruikt. Zulke databanken maken het mogelijk om de inhouden die voor de gebruikers worden gepresenteerd, zinvol te bewaren.

- Lijsten met studenten.
- Lijsten met punten in een digitale leeromgeving.
- Producten, merknamen, kleuren en maten in een webshop.
- Facturen, betalingen en offertes in een boekhoudprogramma...
- Sociale netwerksites zoals Facebook en Twitter en zoekmachines zoals Google bewaren hun resultaten en index in grote databanken.
- Stockbeheer
- …

In heel veel gevallen gebruikt software nog steeds klassieke **relationele databasemanagementsystemen** of **RDBMS**. Maar niet in alle situaties zijn zulke relationele databanken even bruikbaar.

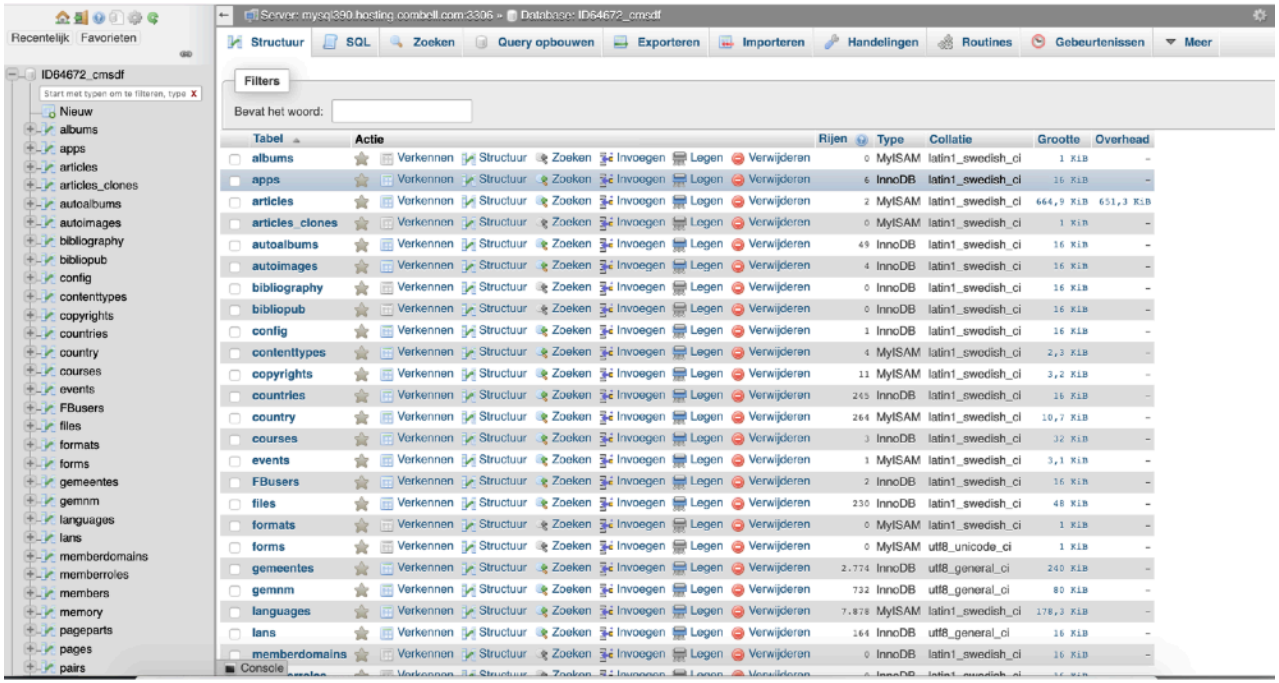
Databanken in maten en gewichten

Databanksoftware voor huis-, tuin- en keukengebruik zoals bijvoorbeeld MS Access of Filemaker Pro, bieden een gebruiksvriendelijke interface voor CRUD-commando's. MS Access-databanken kunnen echter ook gebruikt worden voor databankopslag achter bedrijfssoftware die bijvoorbeeld geprogrammeerd is in Visual Basic of Visual C++.

Grote toepassingen die heel veel data moeten kunnen verwerken, vereisen natuurlijk navenante databanksystemen. Op websites is de databank MySQL de grote nummer één. Het databanksysteem is open source en gratis te gebruiken. Zeer grote toepassingen zoals ERP-software kunnen gebruik maken van MySQL of een ORACLE DATABASE.

Databanken kunnen eveneens geautomatiseerd informatie verzamelen. In een aantal gevallen roept dit ernstige vragen op over schendingen van de privacy. De USA, Groot-Brittannië, Canada, Australië en Nieuw-Zeeland luisteren wereldwijd communicatiekanalen af via het spionagenetwerk ECHELON. Het marketingbedrijf Doubleclick probeert te achterhalen welke e-mailadressen achter IP-adressen verborgen zitten. Voor bedrijven zijn zulke data veel geld waard.

(Afbeelding: PHPMyAdmin, de frontend voor MySQL)



ACID om zeker te zijn

Een databank moet in de meeste gevallen heel consistent zijn. Een financiële databank achter banksoftware, bijvoorbeeld, kan zich niet permitteren om een paar overschrijvingen te missen. Net zo bij databanken achter ERP- of CRM-software. Je personeels- en stockbeheer moet consistent en correct blijven, anders loopt ondermeer je boekhouding in het honderd. Daarom moeten dit soort databanken aan een viertal regels voldoen.

Atomic	Atomair	Een CRUD-transactie moet consistent OF nietig zijn. Niets iets tussen de twee in.
Consistent	Consistent	Een CRUD-transactie moet een nieuwe "status" opleveren of bij de oude status blijven.
Isolated	Geïsoleerd	Als 2 CRUD-transacties tegelijkertijd worden uitgevoerd, mogen ze elkaar niet beïnvloeden.
Durable	Duurzaam	Een transactie blijft geldig. Als je een aanpassing doet, dan blijft die ook in de toekomst van toepassing.

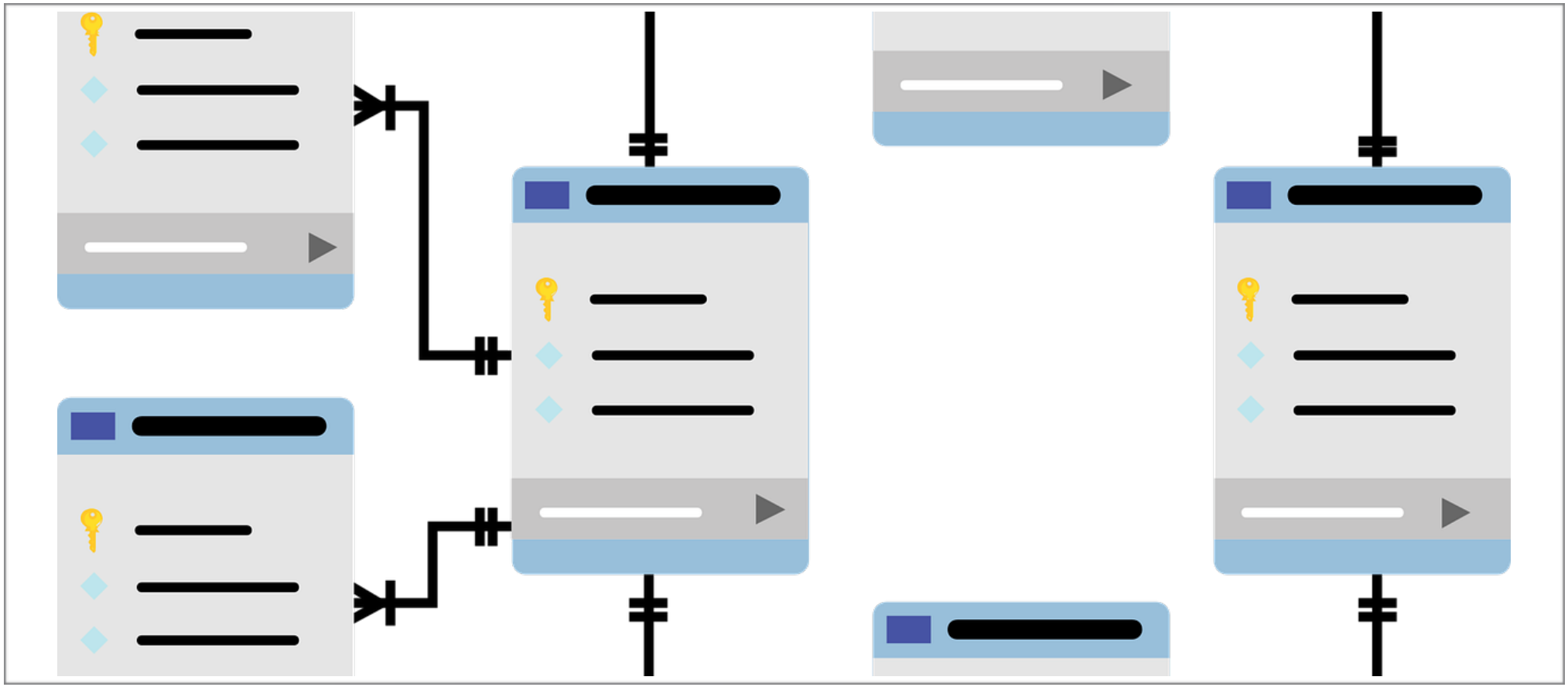
Databanken zonder relaties

Niet alle databanken hebben baat bij een relationele structuur. Immers, de vaste tabellen- en kolommenstructuur maakt het moeilijk om een databank te "schalen".

Een voorbeeld: stel dat je een databank maakt met een overzicht van alle ingrediënten in alle mogelijke voedingswaren. Hoe begin je daaraan met een klassieke relationele databank? Of je wil een analyse van clickstreams op een website. Niet elke webpagina telt even veel links waarop de gebruiker kan klikken en niet elke gebruiker gedraagt zich op identiek dezelfde manier. Soms is de vaste structuur van de databank niet handig en voel je de nood om meer velden of kolommen toe te voegen. Bovendien is "ACID" in zulke gevallen minder relevant. Bij het voorbeeld van de clickstreams: als je een paar kliks mist, heeft dit geen zware gevolgen.

In dit soort gevallen gebruiken bedrijven steeds vaker zogenaamde **NoSQL**-databanken. Begrijp het niet verkeerd. De "No" wil niet zeggen dat dit soort databanken "anti-sql" is. "No" staat voor "Not Only SQL". *"Bekende voorbeelden zijn: digg, Twitter (beiden van MySQL naar Cassandra) LinkedIn (van Oracle en MySQL naar Voldemort), Craigslist (van MySQL naar MongoDB) en natuurlijk Facebook (van MySQL naar HBase)."*[1]

(1. Bron:https://computerworld.nl/development/74798-wat-is-nosql)



Kennen, begrijpen

1. De begrippen RDBMS, CRUD, ACID, SQL en NoSQL volledig uitleggen.
2. Wat zijn gegevenstypes? Wat is het nut daarvan?
3. Uitleggen wat het nut is van databanken.
4. Het verschil tussen relationele databanken en NoSQL-databanken uitleggen.
5. Waarom is ACID belangrijk?
6. Wat zijn de voordelen van RDBMS?
7. Uitleggen hoe een relationele databank je gegevens compacter en consistenter kan bewaren.

Kunnen

8. Een zinvolle indeling van een databank in tabellen kunnen bedenken op basis van gegeven data.
9. Geef een voorbeeld van data waarbij een RDBMS niet zinvol is.

Hoofdstuk 4

DATA VERZAMELEN

- **Welke digitale sporen laat ieder van ons achter?**
- **Hoe verzamelt "men" data?**
- **Hoe kunnen we zelf data verzamelen?**

Welke sporen laat ieder van ons achter? We mailen, posten op sociale media, maken bestanden op onze computer. Al die digitale activiteit laat sporen achter. Van sommige sporen zijn we ons bewust, van andere heel wat minder.

In winkelstraten hangen, al dan niet slimme, camera's die onze bewegingen registreren op zoek naar verdacht gedrag. Ze registreren niet alleen, maar bewaren hun gegevens ook voor eventuele latere analyse.

Langs steenwegen en snelwegen installeert de overheid steeds meer camera's voor trajectcontrole. Iedereen is een potentiële verkeersovertreder en dat zullen we geweten hebben.

Hoe worden data verzameld? Hoeveel data produceren we op één enkele dag? Waarom neemt de hoeveelheid geproduceerde digitale data exponentieel toe?

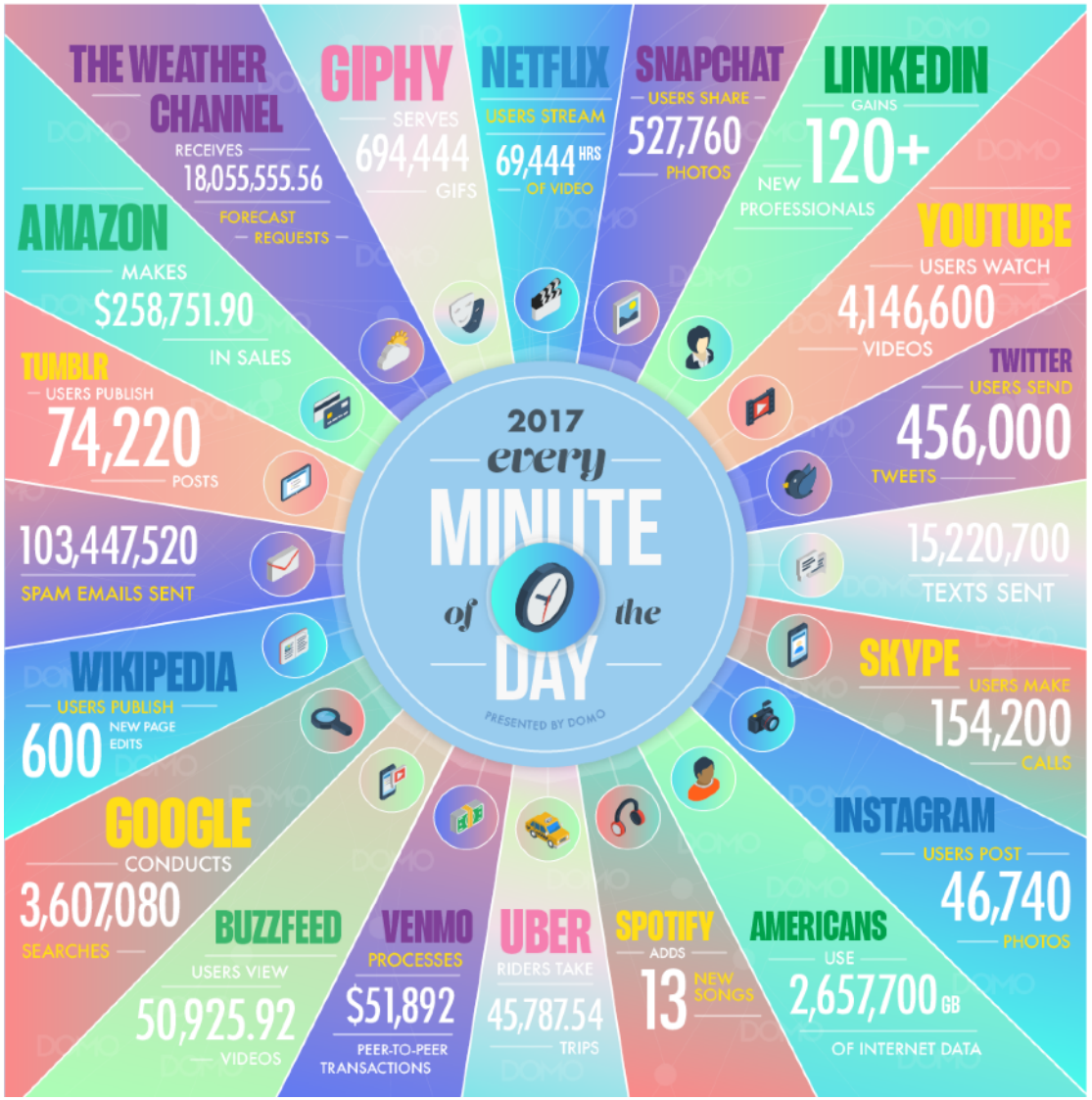
Datarevolutie

De hoeveelheid digitale data die de mensheid dagelijks genereert, tart elke verbeelding. De opkomst van het Internet of Things, versnelt dit nog aan een ongezien tempo.

Het Internet of Things, “smart” toestellen die met het internet zijn verbonden, verzamelen massa’s gegevens. In 2006 ging dit om ongeveer 2 miljard toestellen. Op dit moment is dit aantal al opgelopen tot 200 miljard en het neemt op elk moment nog toe.

90% van alle beschikbare data ter wereld is gedurende de laatste paar jaar tot stand gekomen.

De *smartrevolutie* bracht ons IoT, smartphones, digital assistants (Google Home, Amazon Alexa…), smartwatches, beacons, slimme auto’s, smarthomes… alle dragen bij tot het genereren van data.



"By 2025, it's estimated that 463 exabytes of data will be created each day globally – that's the equivalent of 212,765,957 DVDs per day!"

(Bron: https://www.visualcapitalist.com/how-much-data-is-generated-each-day/)

Oorzaken groei en doel

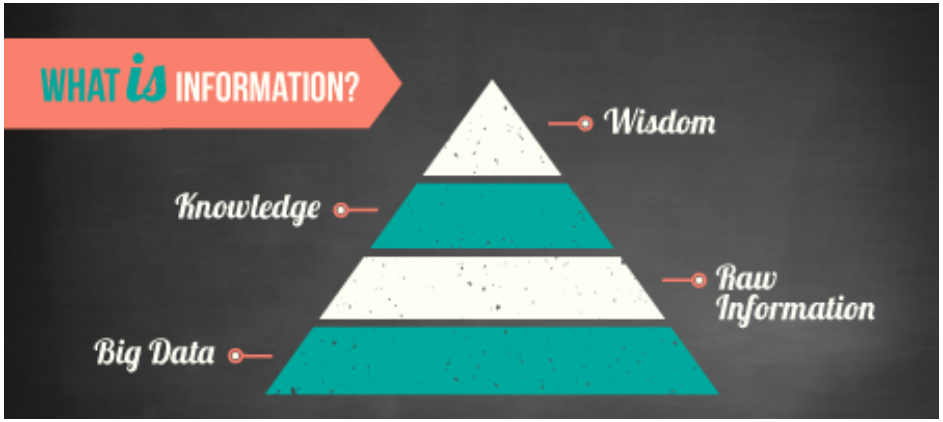
Data wordt niet altijd bewust of met een reden geproduceerd. Vaak dragen nieuwe technologieën bij tot een toename van dataproductie. Anderzijds hebben data op zichzelf geen doel.

Dataproductie en technologie lijken wel een bloem en een bij die elkaar voortdurend bestuiven. De miniaturisering van opslagmedia (harde schijven, SSD…) maakt het mogelijk om steeds meer data te verzamelen en te bewaren. De toenemende rekenkracht van computersystemen, maakt analyse van grote hoeveelheden data mogelijk. Concepten uit kunstmatige intelligentie, zoals neurale netwerken, zijn reeds bedacht op het einde van de jaren 1950 (John Von Neumann), maar worden nu pas echt bruikbaar.

Data kunnen ook doelbewust verzameld worden. Dit kan door nieuwe data te registreren of door bestaande data te filteren. De verzamelde data (data collection) dienen in dat geval een onderzoeksdoel. Dat doel kan van zeer uiteenlopende aard zijn:

- Wetenschappelijk
- Academisch
- Marketing gerelateerd
- Predictief
- Politiek
- Juridisch
- …

Het mag duidelijk zijn dat de Chinese overheid datacollectie en -analyse voor andere doeleinden inzet, dan de meeste universiteiten. Bewuste datacollectie leidt tot analyses. Die analyses leiden tot informatievorming en "kennis".

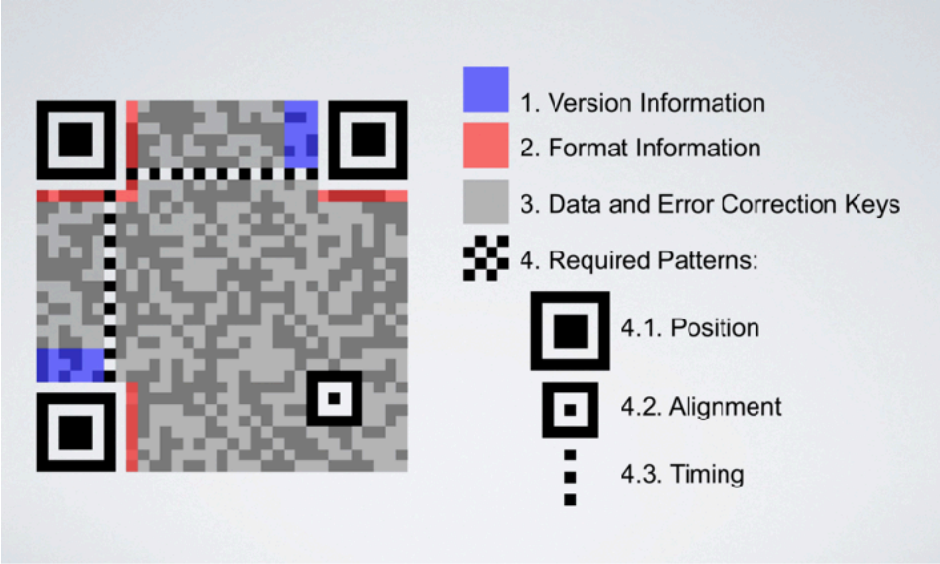


Kennen, begrijpen

1. Waarom groeit de hoeveelheid data exponentieel?
2. Som een aantal voorname oorzaken van de toenemende hoeveelheid data op.
3. Wat zijn de voornaamste redenen van datacollectie?

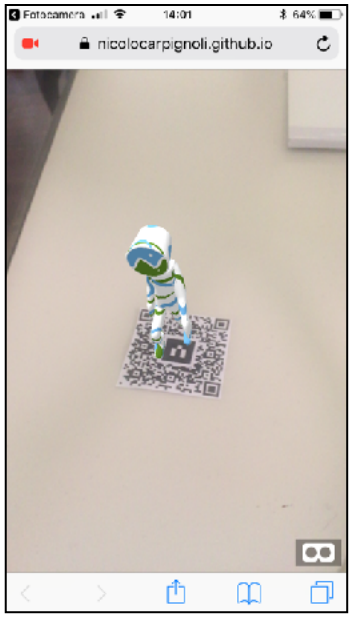
Kunnen

1. Voorbeelden geven van het doel en het nut van datacollectie.
2. Een dataset kunnen vinden en openen met DBeaver.



Bij Augmented Reality wordt digitale informatie toegevoegd aan de analoge werkelijkheid. Digitale data "verschijnen" in onze natuurlijke omgeving. Dit soort immersieve interfaces zal de normale gang van zaken worden bij het bedienen van computersystemen.

(Bron: https://medium.com/chialab-open-source/how-to-deliver-ar-on-the-web-only-with-a-qr-code-e24b7b61f8cb)

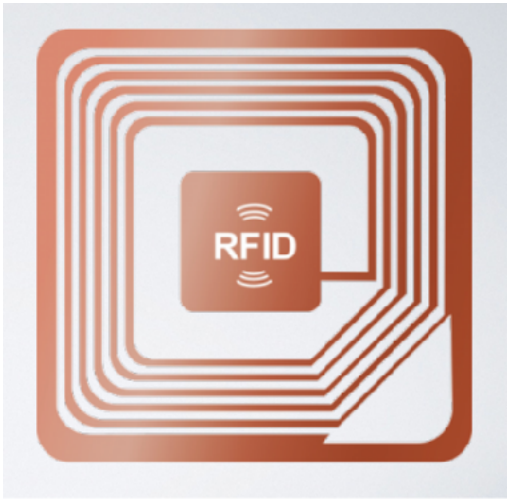


Hoe wisselen wij data uit?

De opkomst van de smartphone heeft ervoor gezorgd dat zo goed als iedereen een digitale assistent op zak heeft. Smartphones zijn doorgaans uitgerust met een hele resem **sensoren** (gps, accelerometer, gyroscoop…) en **communicatietechnologieën** (4G, Bluetooth, NFC…) die het **uitwisselen van data zonder directe menselijke tussenkomst mogelijk maken.**

Quick response

De camera van je smartphone is eveneens een scanner waarmee je documenten kan fotograferen. Met een gepaste app kan je de documenten via **OCR (optical character recognition)** zelfs weer omzetten naar bewerkbare tekst. Bij heel wat smartphones herkent de camera al automatische QR-codes. De QR (quick response) - code lijkt de opvolger van de streepjescode. Ze kan een prijs, maar ook tekstinformatie of een link bevatten. Banken gebruiken het bij hun mobiele apps om betalingen goed te keuren. Als een uitgever een QR-code toevoegt aan een boek, krijgt de lezer via de link in de QR-code toegang tot extra online informatie over het boek.



RFID-tags.

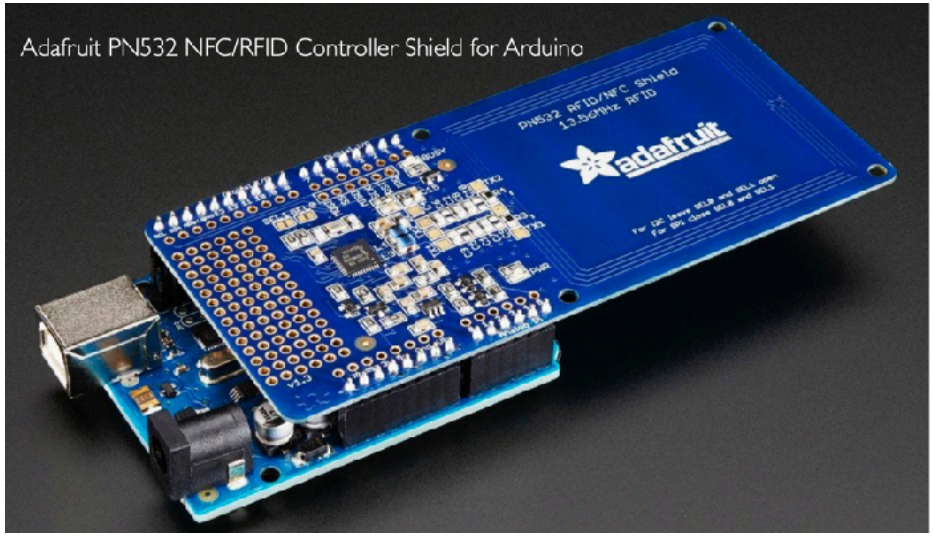
QR-code

QR is een relatief eenvoudig te integreren techniek. Je hebt een stuk software nodig om de codes te genereren. De eindgebruiker heeft een camera-app nodig op zijn smartphone en de nodige software om de code uit te lezen.

Slimme markeringen: RFID, NFC

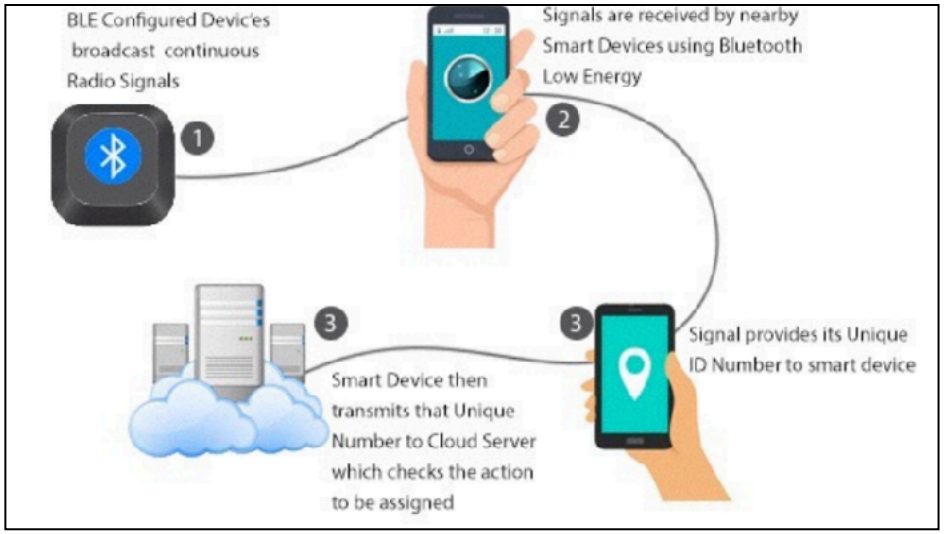
RFID, NFC en BLE daarentegen maken gebruik van radiogolven om informatie uit te wisselen.

RFID (radio frequency identification) is een techniek om fysieke objecten of zelfs levende wezens "uniek" te identificeren. Een volledig RFID-systeem omvat **een tag, een lezer (reader) en een antenne**. De lezer ('reader') stuurt een signaal uit via de antenne. De 'tag' antwoordt daarop met de unieke informatie (bijvoorbeeld een productcode, een prijs… Actieve RFID-tags beschikken over een eigen "energievoorziening" en kunnen zelf radiosignalen uitzenden tot ongeveer 100 m. Passieve RFID-tags krijgen hun energie van de elektromagnetische golven die de RFID-lezer uitzendt. Omdat de radiogolven sterk genoeg moeten zijn, werken passieve tags enkel goed op beperkte afstand (direct contact tot ongeveer 25 meter).



NFC (near field communication) is een gespecialiseerd "familielid" van de RFID-technologie. Een NFC-toestel kan zowel een reader als een tag zijn. NFC-apparaten moeten zich (zoals de naam doet vermoeden: 'near') dicht bij elkaar, op een paar centimeter afstand, bevinden. Daarom gebruiken winkels het graag voor contactloos betalen. Je moet je bankkaart enkel nog tegen de betaalautomaat houden. Veilige uitwisseling van informatie is het hoofddoel van NFC.

NFC vindt ondertussen zijn weg naar de nieuwe generatie smartphones. Als je twee NFC-smartphones in elkaars buurt houdt, kunnen ze foto's uitwisselen (door bijvoorbeeld te 'swipen'). Maar het is ook mogelijk om bijvoorbeeld affiches of posters uit te rusten met een NFC-tag om informatie uit te wisselen met de smartphones van de consumenten. Audioapparatuur uitgerust met NFC kan muziek afspelen door er simpelweg je smartphone op te leggen. *Samengevat*: NFC (Near Field Communication) is een draadloze manier om kleine hoeveelheden informatie uit te wisselen binnen een straal van 10 centimeter. De chip in je smartphone communiceert met een ander NFC-apparaat, zoals betaalsystemen, luidsprekers of andere smartphones,



Indoor plaatsbepaling

Beacons of iBeacons zijn kleine bluetoothzenders (bluetooth maakt gebruik van radiogolven). Een beacon ziet eruit als een klein doosje en bevat een **Bluetooth Low Energy (BLE)-chip** en een batterij. Elke beacon zendt onophoudelijk een uniek nummer uit. Een smartphone met BLE-ondersteuning kan dit signaal opvangen. Als je weet waar welke beacon staat, kan je binnen een gebouw of winkel te weten komen waar iemand zich bevindt. Beacons zijn dus een ideale manier van indoor positiebepaling, want daar is GPS minder bruikbaar.

Apple omarmt BLE als alternatief voor NFC. Terwijl NFC slechts op enkele centimeters werkt, reikt BLE tot zelfs 70 meter. Belangrijk voordeel is dat er geen ingewikkelde installatie bij komt kijken. Je kan ze plaatsen waar je wil. Een iOS- of Android-toestel weet wie jij bent. In combinatie met beacons weet de smartphone-app waar je je bevindt. Als we even niet stilstaan met de gevolgen voor de privacy, dan schept deze technologie heel wat **nieuwe mogelijkheden**: *Indoor plaatsbepaling en navigatie, sales promotion (in-store promotie), loyalty (klantenkaarten), toegangscontrole (alternatief voor toegangspassen), transacties (contactloos betalen)…*

(Bron: https://www.konstantinfo.com/blog/wearables-become-desirables/)

Wearables

Wearables passen vooral in de gezondheidshype. Een wearable is een gadget dat op het lichaam wordt gedragen en informatie over je lichaam verzamelt. De meeste van die toestellen verzamzlen persoonlijke data over de gebruiker. Ze geven advies over en voor je gezondheid. Activity trackers zijn slimme toestellen die bijhouden hoeveel je beweegt en hoeveel calorieën je verbrandt. Sportwatches hebben nog extra functies: zoals het meten van je hartslag of je slaapritme,..

Maar de fabrikanten zijn vooral geïnteresseerd in de data die ze over de gebruikers kunnen varzamelen. Ze vormen een onderdeel van de big data die grote bedrijven als Google en Apple verzamelen en verkopen aan verzekeringsmaatschappijen en andere afnemers.

Met de **LilyPad Arduino** beschik je over een flexibele microcontroller om je kleding of textiel om te vormen tot slimme kleren (**e-textile**). Zelf spreken de ontwikkelaars in dit geval van **softwear**.

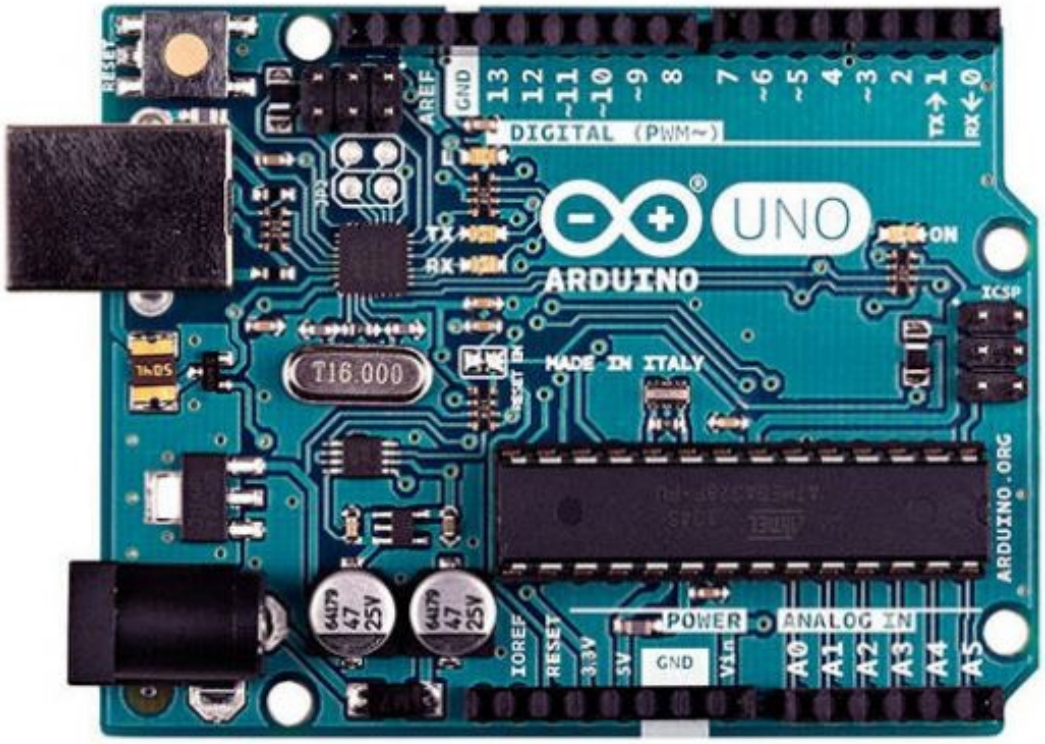
IoT en miniaturisering

Onze computer, laptop, tablet en smartphone zijn niet de enige toestellen in onze omgeving die data over ieder van ons verzamelen.

Er bestaan ook andere verschijningsvormen die men gemakshalve eveneens aanduidt met de naam “computer”, maar in wezen niet helemaal voldoen aan de definitie hier boven. Een microcontroller is zo’n “computer”. Men gebruikt een **microcontroller** of **microprocessor** om elektronische apparatuur te besturen.

Heel wat moderne apparaten bevatten zo'n microcontroller: een magnetron, een auto, een wasmachine, sommige telefoons... Ze bevatten vaak sensoren die data uit hun omgeving verzamelen. Vaak zijn ze via internet (dus op afstand) te bedienen.

Embedded systemen en microcontrollers zijn aan een ware veroveringstocht bezig. Wellicht hoorde je de term “internet of things” al eens vallen. Ruwweg houdt dit in dat stilaan elk huishoudelijk toestel, maar ook auto's, bewakingscamera's en domoticasystemen zulke embedded systemen aan boord hebben. Wanneer je al die dingen (things) aansluit op het internet, krijg je een allesomvattend netwerk van digitaal verbonden toestellen: het **internet of things (IoT)**.



De Arduino is een open source ontwikkelplatform met microcontroller. Je kan er snel prototypes van digitale/elektronische toestellen mee bouwen.

We stellen ons hier even geen vragen over de gevolgen en gevaren voor de privacy en de ethiek.

Kortom, embedded systemen en microcontrollers zijn niet meer weg te denken uit de wereld van vandaag en morgen.

Embedded systemen bieden tal van **voordelen** in vergelijking met een “computer”.

1. De verwerking gebeurt real time (beeld je maar eens in dat je auto zou werken op Windows),

2. snel

3. en verbruikt zeer weinig energie.

4. Een embedded systeem is ook veel goedkoper.

Nadeel is dat de functionaliteit heel beperkt is. Je kan bijvoorbeeld geen tekstverwerker installeren in je koelkast of videomontagesoftware in je auto.

Afbeelding:

Het Leuvense IT-bedrijf EASICS ontwikkelde de beeldherkenningshardware waarmee de sorteermachines van de Belgische firma BEST zijn uitgerust. De hardware van EASICS herkent tegen een onwaarschijnlijk hoog tempo "ongewenste" elementen tussen bijvoorbeeld razendsnel voorbij rollende frieten, krenten, garnalen, spijkers of wat dan ook en activeert een luchtdrukstraal die de ongewenste element wegspuit. Een "multifunctionele" computer, hoe krachtig en snel ook, zou er niet in slagen om die taak zo snel af te handelen. Omdat de programmacode in de chip is gecodeerd, verloopt de verwerking hier onwaarschijnlijk snel.

Naast de term "microcontroller" en "embedded system" duikt ook wel eens de term "**system on a chip" (SoC)** op. Een SoC (met een C wel te begrijpen), integreert alle componenten van een computer en/of elektronisch systeem op één enkele chip.

SoC's worden veelvuldig toegepast in de markt van mobiele consumentenelektronica. Een "embedded systeem" bevat vaak een SoC.

Afbeelding:
Diverse SoC's in een smartphone.

Kennen, begrijpen

1. Noem minstens 4 jonge technologieën voor draadloze data-uitwisseling op en leg uit.
2. Wat zijn of betekenen QR, RFID, NFC, BLE, Beacons, wearables, IoT, microcontrollers, SoC…
3. Hoe kan je binnen een gebouw data verzamelen van de bezoekers?
4. Welke invloed hebben deze technologieën op de exponentiële groei van data.

Kunnen

5. Situaties bedenken waarin je deze technologieën zinvol zou kunnen inzetten voor data-collectie.

Data verzamelen

Als je gegevens wil verzamelen beschik je over een massa bronnen. De veelheid van data maakt het echter niet eenvoudiger om zinvolle informatie uit te halen. Zoals eerder al gezegd heeft data-collectie op zich geen zin. Je wil de data analyseren en er vervolgens conclusies uit trekken.

Soms bestaan de gegevens al die je nodig hebt. Een verzameling van gegevens (die je bijvoorbeeld als trainingsset aan een AI-algoritme gaat voeren, zie hoofdstuk 6) noemt men vaak een **dataset**.

Het lijkt vrij logisch om je informatie te zoeken op het internet of het wereldwijde web. Zo goed als alle webpagina's zijn gebouwd met de taal HTML die weliswaar gestructureerde informatie bevat, maar niet altijd uitblinkt op het vlak van semantiek.

Daarnaast vind je op het web heel wat informatiebronnen die hun data ook nog op een andere manier toegankelijk maken, via een REST-api.

HTML en online data

Ieder van ons bezoekt websites, social media en gebruikt applicaties op het wereldwijde web. De inhouden die we gepresenteerd krijgen, zijn een combinatie van **HTML** (inhoud, structuur, links), CSS (opmaak, animaties) en javascript (interactie). Die drie "talen" vormen de bouwblokken van wat we in onze browsers te zien krijgen.

Een groot deel van de informatie en de inhouden achter al die websites en webservices wordt echter bewaard in online databanken (heel vaak MySQL).

Met behulp van "backendtechnologieën" zoals PHP, Node.JS, Java... worden die databankrecords omgetoverd in webpagina's.

REST-API's

Een aantal grotere online platformen bieden ook nog andere outputformaten aan dan enkel HTML. Zo kan je bijvoorbeeld gegevens uit Wikipedia plukken in de vorm van JSON-data. Dat is vooral handig als je je eigen software wil ontwikkelen die gegevens uit andere websites wil gebruiken. In sommige gevallen mag dit gratis, in andere gevallen moet je voor zo'n dienst betalen. Na betaling krijg je een authenticatiesleutel die je in je programmeercode moet opnemen.

Om de gegevens op te vragen, volstaat het vaak om een URL op de correcte manier op te bouwen. De techniek om informatie op te vragen in gestandaardiseerde en/of gestructureerde formaten, heet **REST (Representational state transfer)**. De mogelijkheid die een online service biedt en de beschrijving van hoe je de data kan opvragen, heet een **API (een Application programmering interface)**.

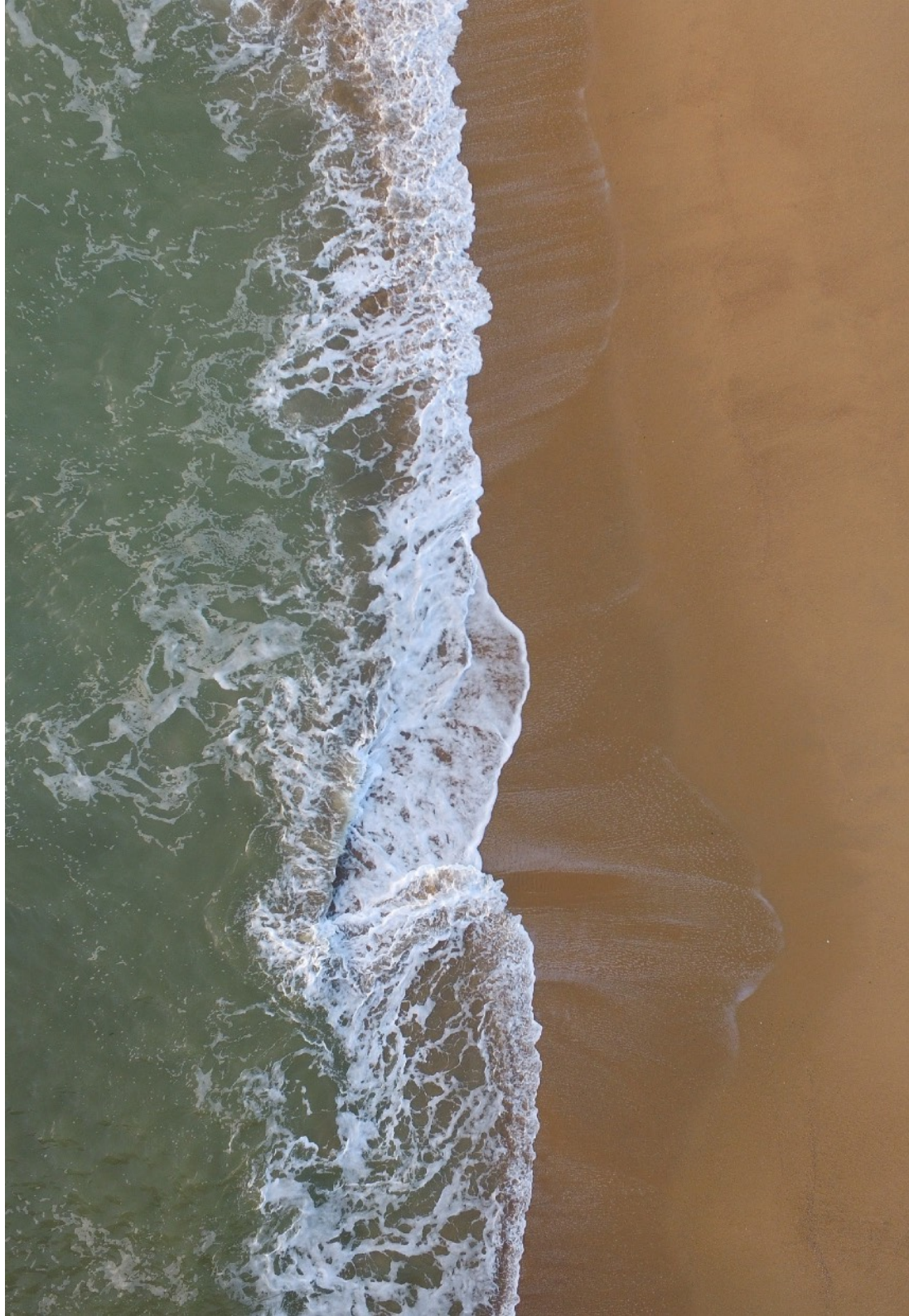
Webscraping

Webscraping is een techniek waarbij je geautomatiseerd inhouden van webpagina's of REST-api's kopieert naar een computer of naar een andere website of webservice. Over het algemeen moet je hiervoor aan de slag met programmeercode.

Data clearance

Gegevens opschonen of data cleansing is het proces van het detecteren en corrigeren (of verwijderen) van onnauwkeurige of foute records uit een recordset, tabel, database of dataset. Na het opschonen moet een gegevensset consistent zijn met andere vergelijkbare gegevenssets in het systeem. De gedetecteerde of verwijderde inconsistenties kunnen oorspronkelijk zijn veroorzaakt door fouten bij het invoeren van gebruikers, door fouten bij verzending of opslag.

De validatie kan streng zijn (zoals het weigeren van elk adres dat geen geldige postcode heeft) of vaag ("fuzzy", zoals het corrigeren van records die gedeeltelijk overeenkomen met bestaande, bekende records).



Kennen, begrijpen

1. Was zijn datasets?
2. Wat is een REST-api?
3. Leg het nut en het doel van webscraping uit.
4. Op welke manieren kan je geautomatiseerd data verzamelen uit bronnen op het internet?
5. Wat versta je onder "data clearance" of "data cleansing”?

Kunnen

6. Zelf tips kunnen geven om gegevens op te schonen (data cleansing).
7. Tijdens een gerichte opdracht via webscraping informatie van websites kunnen halen (tijdens de les).
8. Tijdens een gerichte opdracht via REST-api's informatie kunnen ophalen uit webservices (tijdens de les).

Hoofdstuk 5

VAN ALGORITME TOT SLIMME COMPUTER

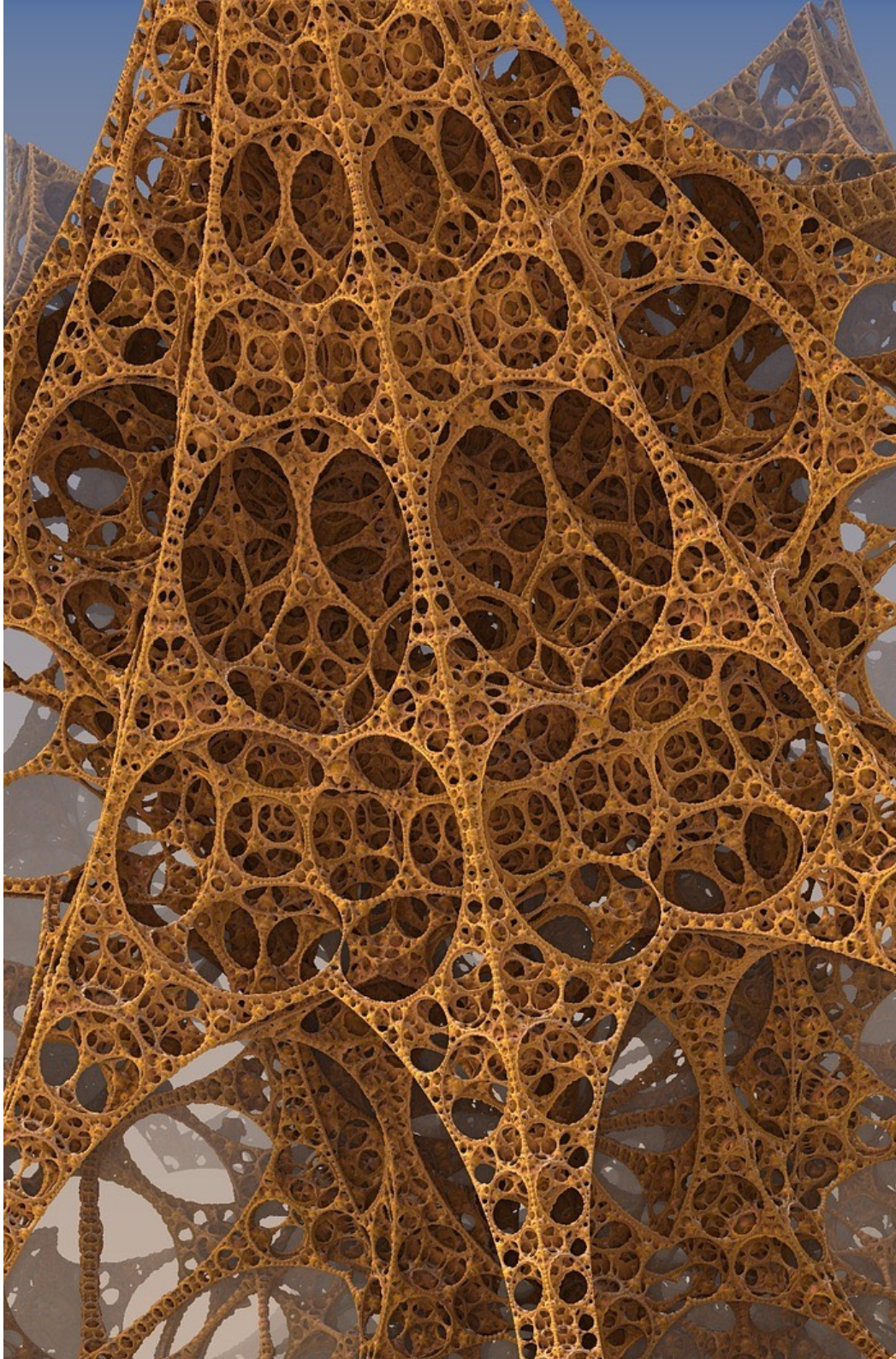
- Wat zijn algoritmes?
- Waarom spelen algoritmes zo'n belangrijke rol in de wereld van vandaag?
- Wat is automatisering?

Algoritmes

Ongetwijfeld heb je het woord al eens gehoord: een algoritme. Het heeft niks met ritmische muziek te maken alhoewel Shazam wel algoritmes gebruikt voor patroonherkenning in muziek. Google gebruikt het PageRank-zoekalgoritme en wanneer je naar een MP3-muziekbestand luistert dan is dit eerst met een compressiealgoritme tot een relatief klein bestand herleid vooraleer het op je mediaspeler is beland. Algoritmes spelen een cruciale rol in de verwerking van big data.

Algoritmes zijn in de digitale wereld alomtegenwoordig, maar wat zijn het eigenlijk?

Algoritmes zijn in dit geval stukjes programmeercode die ervoor zorgen dat een computer bepaalde taken in de correcte volgorde kan uitvoeren. We sommen de belangrijkste software-algoritmes op.



Dijkstra: de kortste weg

Het Dijkstra-algoritme is vooral bekend als het kortste pad-algoritme. Het beschrijft de kortste afstand tussen twee punten.

"Het algoritme van Dijkstra wordt gebruikt door verschillende internetprotocollen, voor het vinden van de kortste route tussen computers of routers. Het algoritme van Dijkstra geeft gegarandeerd het kortste pad, als er überhaupt een pad tussen A en B bestaat, maar kan hier wel lang over doen. De snelheid van het algoritme is afhankelijk van de manier waarop de punten en lijnen opgeslagen zijn in het geheugen van de computer."

(BRON; GUNNINK, M., "Padvinder vindt pad: over Pathfinding", (http://kninnug.nl/padvinder/index.html), Geraadpleegd op 20 oktober 2015.)

Dobbelen

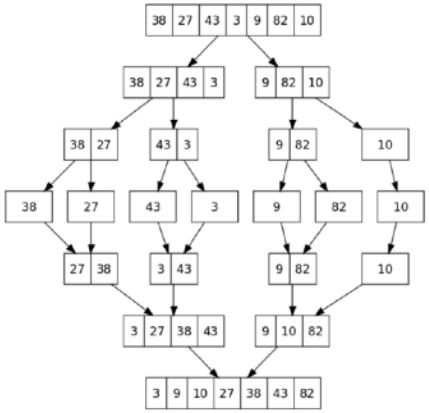
Als je dobbelt met een dobbelsteen of de lotto speelt, speel je letterlijk met "toeval". Een computer kan eveneens een lukraak getal genereren.

In de meeste programmeertalen kan je de functie om een random getal te genereren eenvoudig oproepen. Random-getalgenerators zijn belangrijk in beveiliging en cryptografie, games, AI enz.

Sorteeralgoritmes

Op school leer je woordjes alfabetisch rangschikken. Dat doe je in eerste instantie door naar de eerste letter te kijken. Bevatten veel woorden dezelfde beginletters, dan rangschik je die woorden door naar de tweede letter en vervolgens naar de volgende letters te kijken.

Rangschikking vereist dus een algoritme, een strikt stappenplan om tot het beoogde eindresultaat (een alfabetische volgorde) te komen. Een **sorteeralgoritme** zet elementen van een lijst in een bepaalde volgorde: alfabetische, aflopend, oplopend… Als je in bijvoorbeeld MS Excel getallen naar grootte rangschikt, dan gebruik je in feite een sorteeralgoritme.



Het alfabetisch rangschikken van woorden en het ordenen volgens grootte van getallen lijkt voor ons erg makkelijk. Maar het vraagt wel een stappenplan in ons hoofd, ook al doe je het na een tijdje zonder na te denken. Hoe beter het algoritme, hoe sneller de procedure. Dit soort algoritmes wordt reeds van in de beginjaren van de computer gebruikt in de computerwetenschappen. John von Neumann bedacht het "sort merge"-algoritme in 1945. Het sorteert gegevens door het splitsen, ordenen en weer samenvoegen (merge)van data. Tony Hoare bedacht in 1959 het Quicksort-algoritme, dat zoals de naam al doet vermoeden, sneller werkt dan "sort merge".

In 1964 bedacht J.W.J. Williams het "heapsort"-sorteeralgoritme. Zonder dit soort "sorteeralgoritmes" zouden heel wat moderne computertechnieken zoals data mining, AI, linkanalyse... ondenkbaar zijn.

Linkanalyse

Welke links bovenaan verschijnen in Googles zoekresultaten wordt bepaald door het PageRank-algoritme. Facebook toont bij iedere gebruiker een andere newsfeed gebaseerd op de "vrienden" met wie je het meest contact hebt (op Facebook dan toch), berichten die de meeste "likes" hebben gekregen enz. Wat je te zien krijgt, hangt dus af van een heleboel factoren en wordt bepaald door een linkanalyse-algoritme. Uiteraard verschillen de algoritmen maar de basis voor linkanalyse werd gelegd in 1976 door Gabriel Pinski en Francis Narin.

"Who uses this algorithm? Google in its Page Rank, Facebook when it shows you your news feed (this is the reason why Facebook news feed is not an algorithm but the result of one), Google+ and Facebook friend suggestion, LinkedIn suggestions for jobs and contacts, Netflix and Hulu for movies, YouTube for videos, etc. Each one has a different objective and different parameters, but the math behind each remains the same. Finally, I'd like to say that even thought it seems like Google was the first company to work with this type of algorithms, in 1996 (two years before Google) a little search engine called "RankDex" , founded by Robin Li, was already using this idea for page ranking. Finally Massimo Marchiori, the founder of "HyperSearch", used an algorithm of page rank based on the relations between single pages. (The two founders are mentioned in the patents of Google)."

Bron: OTERO, M., "The real 10 algorithms that dominate our world" (https://medium.com/@_marcos_otero/the-real-10-algorithms-that-dominateour-world-e95fa9f16c04), 2014, Geraadpleegd op 19 oktober 2015.

Suggesties

Wanneer je op twee verschillende computers van twee verschillende “mensen” dezelfde zoekterm invoert in Google krijg je vaak andere zoekresultaten. Googles **PageRank** baseert de zoekresultaten immers ook op uw zoekgeschiedenis en profiel en toont op basis daarvan ook andere advertenties via het **Google Adwords-algoritme**.

Wanneer je een zoekterm begint in te voeren toont **Google Suggest** reeds een reeks mogelijkheden waaruit je een optie kan selecteren. Ook Amazon en Netflix tonen op basis van uw zoekgeschiedenis en aankopen andere producten die u wellicht interesseren. Eli Pariser noemt dit “informatie determinisme”. De gebruiker zit in een soort van “filter bubbel” waar hij zelf voor een deel de controle over verliest. Je kan dus niet zelf meer kiezen wat je te zien krijgt, algoritmes bepalen dit voor u. Vraag is of je, puur uit marketingoogpunt, op die manier geen potentiële kopers van bepaalde producten mist.

Online dating is eveneens een voorbeeld van een veelgebruikte dienst die op basis van suggestie-algoritmes functioneert. Het matching-algoritme van OKCupid (een online datingdienst) werd afgeleverd door de Harvardwiskundige Christian Rudder. Het maakt een “ruwe” match op basis van gemeenschappelijke interesses. Vraag is natuurlijk of iemand met gelijkaardige interesses sowieso de beste partner is. Daarom berekent het algoritme eveneens hoe belangrijk elke vraag is voor de andere partij.

Veiligheid

Steeds meer computergebruikers plakken hun webcam af met een stukje plakband, iedereen weet dat je privacy wel eens kan geschonden worden door al je internetavonturen. Toch rekenen we wel op veiligheid als we online bankieren of wanneer we een reis boeken via Booking.com en onze creditcardgegevens invullen in een App Store of in PayPal. Zonder dit gevoel van zekerheid en veiligheid zou je wel gek moeten zijn om je bankkaartgegevens ergens achter te laten.

Het **RSA-algoritme** (ooit geschreven door de firma RSA) en het **Secure hash-algoritme** zorgen ervoor dat onze data veilig en versleuteld worden uitgewisseld tussen onze thuiscomputer en de servers van banken en online winkels. Een ander belangrijk algoritme in de wereld van de cryptografie is het **Integer factorization-algoritme**.

Foutcorrectie en autotuning

Het klinkt in de eerste plaats al indrukwekkend, het "Proportional Integral Derivative Algorithm" maar wat het doet is al even tot de verbeelding sprekend. Een **PID-controller** berekent constant foutwaardes als het verschil tussen de gemeten signaalwaarde en het gewenste resultaat en werkt vervolgens de fout weg. PIDcontrollers worden in de industrie toegepast, maar ook in meer populaire toepassingen. Zangers zingen in een opnamestudio niet altijd netjes op de toon. Doorheen een zangpartij zitten ze vaak wel eens net boven of net onder de juiste toon. Dat wil nog niet zeggen dat ze ronduit vals zingen, maar ze zitten zelden loepzuiver op de toon.

Autotuning zorgt ervoor dat al die foutjes verdwijnen in de uiteindelijke gemixte opname. Het lied "Believe" van de zangeres Cher zou de eerste popsong zijn die van autotuning gebruik heeft gemaakt. Je merkt meteen het verschil als je naar opnames uit de jaren 1960 of 1970 luistert en die vergelijkt met opnames uit de digitale periode vanaf 1984. Beluister bijvoorbeeld eens het album "Communique" van Dire Straits en het loepzuivere digitale geluid van "Brothers in Arms" van diezelfde groep. Niet dat we Dire Straits en Mark Knopfler "verdenken" van autotuning, maar de digitaliseringsalgoritmes hebben een blijvende "stempel" gedrukt op alle opnames vanaf de komst van de CD.

Conversie en compressie

Het Fourier-transformatie-algoritme en de **Fast Fourier-transformatie of het FFT-algoritme** liggen aan de basis van heel wat compressie-algoritmen zoals JPG en MP3. Een **Fourier-transformatie** zet een geluidssignaal of lichtintensiteit om in een golffunctie.

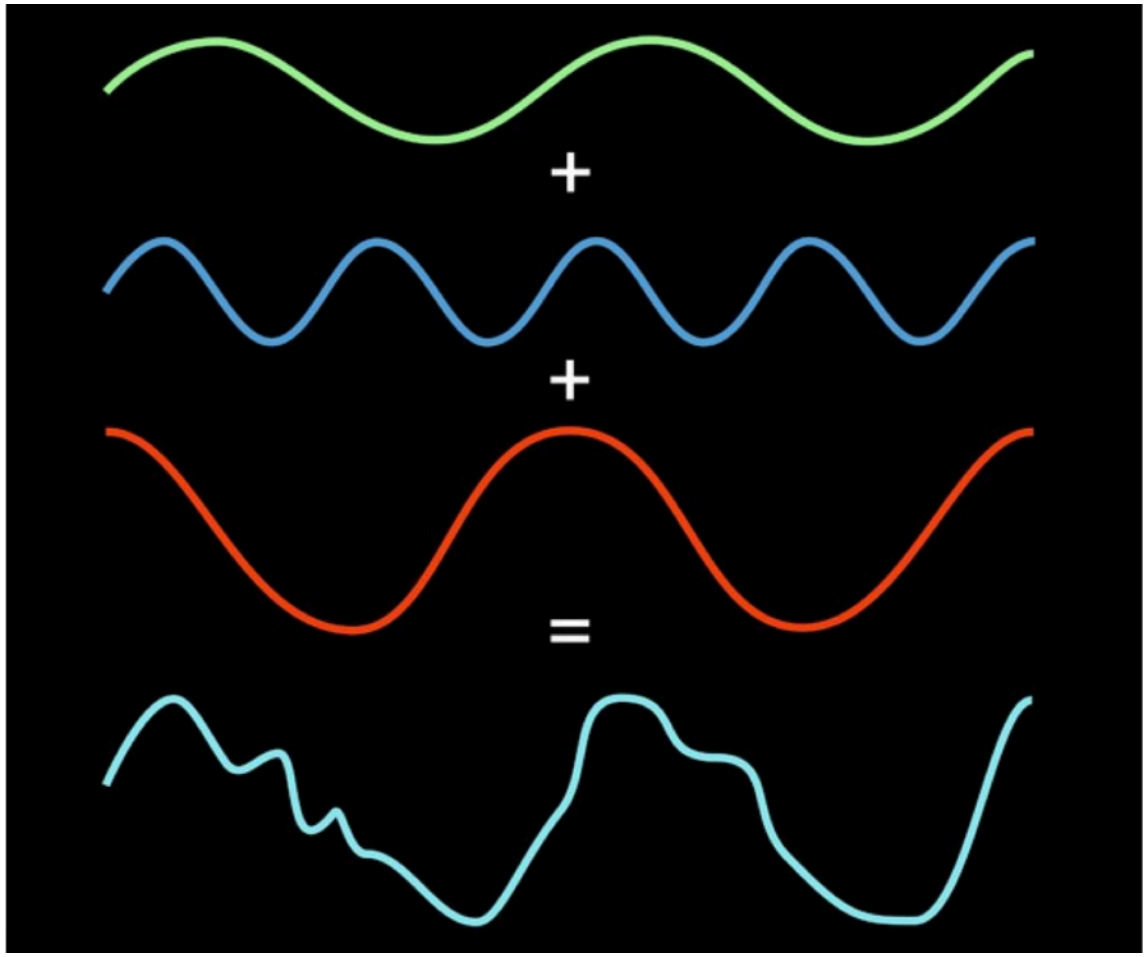
Het internet, WiFi, een smartphone, computer, router, communicatiesatellieten enz., zo wat alle toestellen waarin een soort computer aanwezig is, gebruikt deze algoritmen. Het meet analoge signalen doorheen de tijd en tekent die uit als een golf op basis van de gemeten frequenties. Door signalen op te zetten in (golf)functies, kan de intensiteit van de golf eveneens getransformeerd (vervormd) worden.

Het FFT-algoritme wordt bijvoorbeeld gebruikt bij afbeeldingsfilters waarbij men de intensiteit van kleuren of belichting 'transformeert'. Bij compressie worden lage intensiteitsverschillen weggegooid waardoor de bestandsgrootte erg verkleind.

"Fourier-transformaties worden gebruikt bij het omzetten van analoge signalen in digitale data en worden onder meer toegepast bij signaalverwerking, maar spelen ook een rol bij de compressie van beelden. Een complex signaal wordt met fft gereduceerd tot een aantal eenvoudig digitaal te representeren componenten. Daarvoor wordt een analoog sample opgedeeld in kleine stukjes, waarna de stukjes worden geanalyseerd en geconverteerd naar een digitaal signaal."

Bron: DE MOOR, W., "Sneller algoritme voor Fourier-transformaties ontwikkeld", (http://tweakers.net/nieuws/79444/sneller-algoritme-voor-fouriertransformaties-ontwikkeld.html), 2012, Geraadpleegd op 19 oktober 2015.

Compressie-algoritmes spelen een bijzonder belangrijke rol. *Ze zorgen ervoor dat data kleiner worden, maar vaak gaat dit gepaard met kwaliteitsverlies.* De algoritmes maken een afweging tussen kwaliteit en kwantiteit.



Controle en voorspelling

In 1948 schreef George Orwell het boek "1984" waarin hij scherpe kritiek leverde aan het adres van de USSR.

Hij schetste *"een onmenselijke dictatoriale eenpartijstaat die in alle opzichten volledig beheerst wordt door de Partij. De alom aanwezige leider van de Partij en het land wordt Big Brother genoemd. Iedere bewoner van het land wordt continu in de gaten gehouden via camera's die zelfs in de huizen zijn geïnstalleerd. Dit gebeurt onder de slogan 'Big Brother is watching you' ('Grote Broer houdt je in de gaten')."*

Ondertussen leven wij in het westen niet in dictatoriale eenpartijstaten, maar Big Brother lijkt heel reëel. Ieder van ons wordt in de gaten gehouden, zij het niet door mensen, maar door algoritmes. Vanuit de USA houdt het National Security Agency (NSA) en zijn internationale partners (Five Eyes: US, Australië, Canada, Nieuw-Zeeland, Groot-Brittannië) wereldwijd miljoenen mensen in de gaten. Ze monitoren telefoongesprekken, SMS-berichten, e-mails, webcambeelden, GPS-locaties enz. De hoeveelheid verzamelde informatie is veel te groot om te laten analyseren en interpreteren door mensen. De analyse gebeurt automatisch met behulp van krachtige algoritmes.

Sommige algoritmes zoals IBM's CRUSH, gaan nog een stukje verder. CRUSH of "Criminal Reduction Utilizing Statistical History" zorgt voor "predictive analysis" en is in hoofdzaak bedoeld om misdaten te voorkomen. De politiediensten van Memphis konden de misdaadcijfers met 30% laten teruglopen dankzij CRUSH. Het aantal gewelddadige misdrijven liep sinds 2006 terug met 15%. Op basis van statistische gegevens, data-aggregatie en algoritmes, toont de software criminele hot spots op een kaart.

Politie-eenheden kunnen op de manier pro-actief ingeschakeld worden en bij wijze van spreken aankomen voor een misdaad plaatsvindt. In de toekomst zullen criminelen heel snel opgespoord kunnen worden dankzij internetactiviteit, GPS en biosignaturen, verdacht gedrag.

"In the future, these systems will largely take over the work of analysts. Criminals will be tracked by sophisticated algorithms that monitor internet activity, GPS, personal digital assistants, biosignatures, and all communications in real time. Unmanned aerial vehicles will increasingly be used to track potential offenders to predict intent through their body movements and other visual clues."

Bron: DVORSKY, G., "The 10 algorithms that dominate our world", (http://io9.com/the-10-algorithms-that-dominate-our-world-1580110464), 2014, Geraadpleegd op 20 oktober 2015.

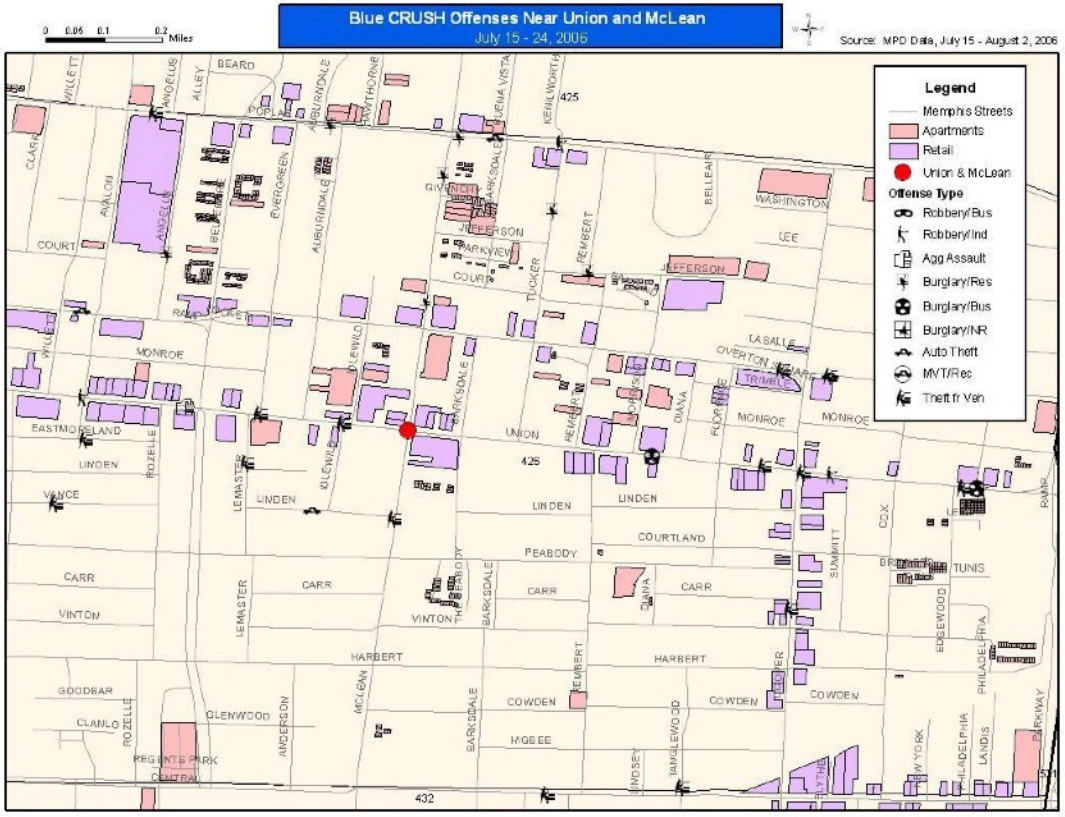
De film Minority Report van Steven Spielberg uit 2002 toont al een glimp van waartoe dit in de toekomst kan leiden.

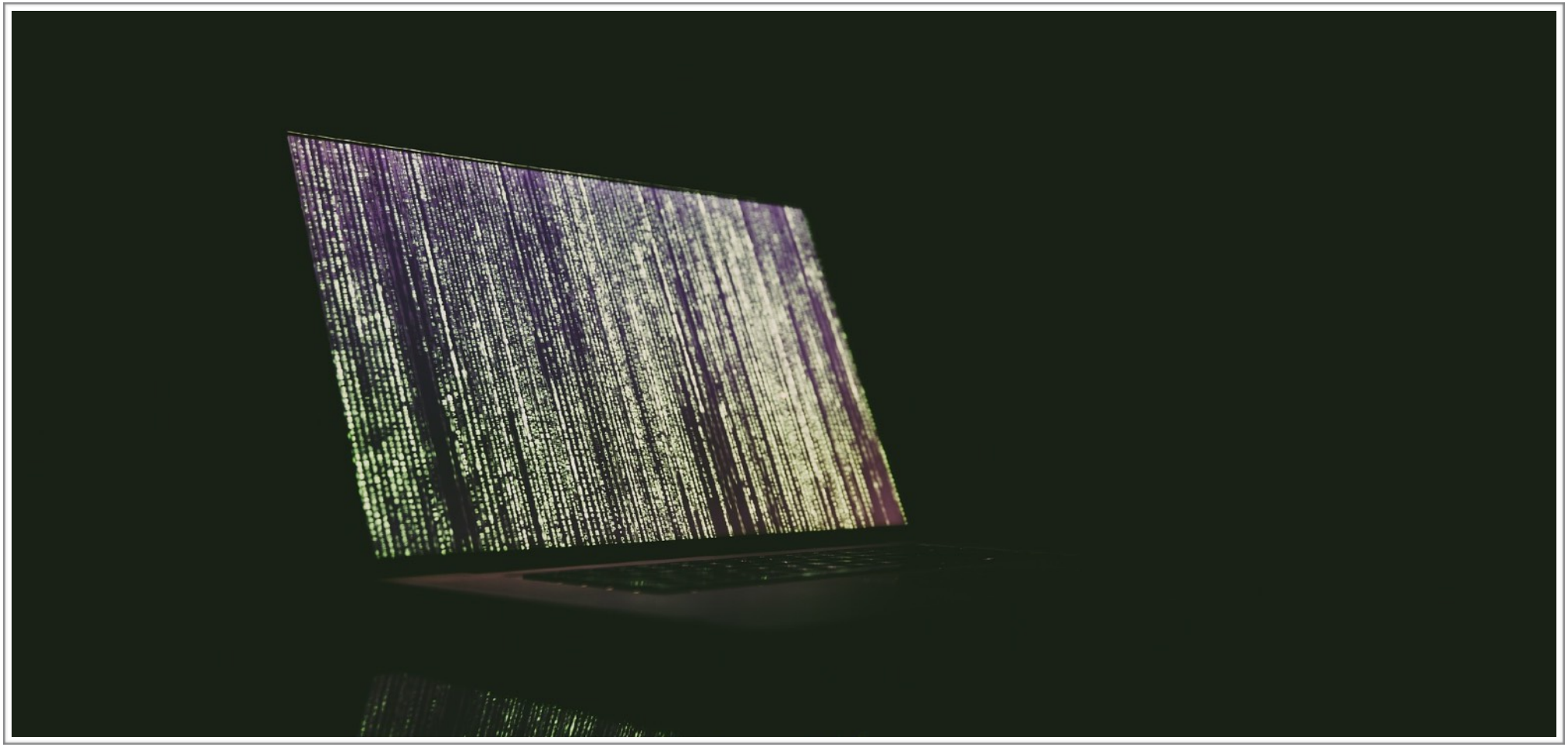
Algoritmes voor predictieve analyse zijn reeds lang in gebruik in de beurswereld. De analyse van razendsnel voorbijvliegende transacties gebeurt met behulp van slimme algoritmes. Soms kloppen de predicties niet, zoals bij de “Flash Crash” van 2010.

De opkomst van AI en slimme camera’s, maakt het volgen en controleren van burgers nog makkelijker.

In China is Big Brother niet langer fictie: *“This fictional scenario is now a daily reality in the People’s Republic of China, thanks to a massive police surveillance apparatus powered by Big Data and artificial intelligence. For example, trains now require national ID’s to buy tickets, which allows the government to block human rights activists or anti-corruption journalists from traveling. In Xinjiang province, home of China’s Uighur Moslem minority, the government uses AI-sifted Big Data to scrutinize anyone entering a mosque or even a shopping mall thanks to thousands of checkpoints requiring a national ID check-in—and which can collate real-time data with other personal information on everything from bank accounts to family planning.”*

Bron: HERMAN, A. “China's Brave New World Of AI”, (https://www.forbes.com/sites/arthurherman/2018/08/30/chinas-brave-new-world-of-ai/), 2018, Geraadpleegd op 5 april 2019





Kennen, begrijpen

1. Uitleggen wat een algoritme is.
2. Noem minstens 5 belangrijke algoritmes op en leg uit.
3. Voorbeelden geven van praktische toepassingen van de algoritmes die je opsomt.
4. Uitleggen waarom algoritmes zeer belangrijk zijn bij dataverwerking.

Kunnen

5. Bij een gegeven dataset kunnen definiëren welke algoritmes zouden kunnen helpen bij het verwerken van de data.

De evolutie van automatisering

Heel wat mensen vrezen dat de opkomst van kunstmatige intelligentie zal leiden tot een gigantisch banenverlies. Wat moeten onze kinderen en kleinkinderen nog gaan doen om een inkomen te vergaren? Waarom maken mensen machines en technologie die hen overbodig maakt?

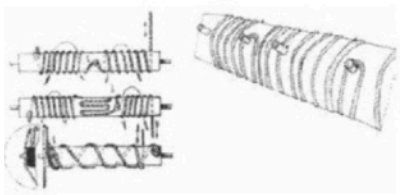
Welke evolutie kennen machines? In eerste instantie hebben mensen gereedschappen ontwikkeld om het werk lichter te maken, zodat ze minder spierkracht moesten gebruiken. Het is nu eenmaal eenvoudiger om een spijker met een hamer in een muur te slaan dan met je blote vuist.

De mens heeft m.a.w. gereedschappen ontwikkeld die het werk, maar ook het denkwerk of onthoudwerk lichter maken. De uitvinding van het schrift bijvoorbeeld zorgde er voor dat je niet langer alles moest onthouden. Door het gebruik van trekdieren, maar ook windmolens en watermolens, kon heel wat menselijke spierkracht uitgespaard worden.

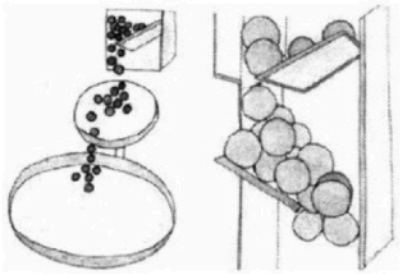
Maar het aansturen van taken diende over het algemeen nog door mensen te gebeuren. Hoe je een bepaalde taak tot een goed einde moest brengen, hoe je gereedschap moest bedienen… dat diende nog aangeleerd te worden. Mensen moesten de diverse stappen van een taak, job of functie nog aanleren.

Elke taak bestaat immers uit een aantal te volgen stappen. Ze volgt een soort van recept. In de wiskunde en informatica noemt men zo'n proces of stappenplan een **algoritme**. De mens ontwikkelt al meer dan 2000 jaar automaten waarin zo'n algoritmes konden worden geprogrammeerd. Een klok is daar een mooi voorbeeld van. Een klok bevat een ingewikkeld mechanisme waarmee de tijd kan worden berekend. Tot de meest bekende voorbeelden van automatisering, behoort het assemblageproces van auto's aan een "lopende band" in autofabrieken.

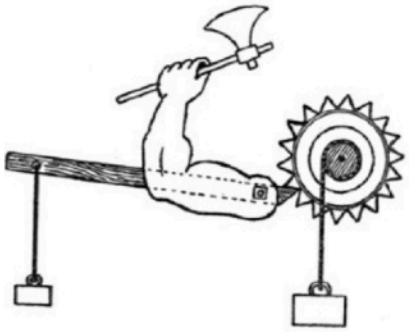
Afbeeldingen: De Computer van Antikythera, 2e eeuw v.Chr.

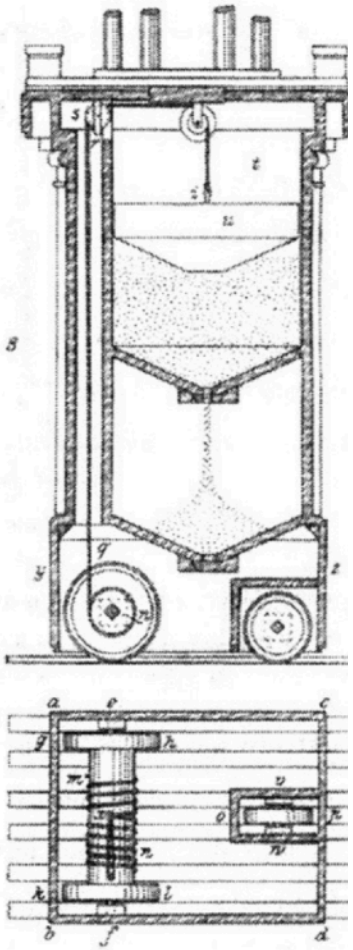


De touwen en de pinnen bepalen de beweging van de assen. (Bron: Manuela Rausch, *Die Programmgesteuerten Automaten Des Heron Von Alexandria*, 2012)



Geluidseffecten in het automatische theater. (Bron: Manuela Rausch, *Die Programmgesteuerten Automaten Des Heron Von Alexandria*, 2012)





(Bron: Beck T., *Herons des älteren Automatentheater*, Jahrbuch des Vereines deutscher Ingenieure, 1909)

Programmeerbare automaat, 1e eeuw na Chr. Ingenieur: Heron van Alexandrië

De eerste golven van automatisering vanaf midden 19e eeuw kostten het werk aan veel arbeiders. Zeker vanaf het moment dat ook robots in het productieproces werden ingezet. Een robot is zodanig afgesteld en geprogrammeerd dat hij een bepaalde taak telkens stapsgewijs kan uitvoeren, zonder moe of ziek te worden of zonder vakantie te eisen. Robots voeren m.a.w. voorgeprogrammeerde algoritmes uit.

De volgende stap in de evolutie van technologie gaat nog een stuk verder. We bouwen machines die zichzelf kunnen programmeren. Meer nog, slimme algoritmes zijn in staat om zelf taken te leren. Shazam kan bijvoorbeeld liedjes herkennen en wordt daar steeds beter in. Google Translate leert net als een kind stapsgewijs alle talen ter wereld.

Wanneer je bepaalde werk- of bedrijfsprocessen automatiseert, spreek je van **RPA (robotic process automation)**. Dit betekent niet noodzakelijk dat je “robots” inzet, maar dat je manueel (denk)werk vervangt door software of andere geautomatiseerde processen.

Doemscenario’s duiken op waarbij toekomstige software in staat zal zijn elke denkbare taak aan te leren om in sneltempo slimmer te worden dan de mens.

Kennen, begrijpen

1. Kort omschrijven welke de belangrijkste “sprongen” zijn in de automatisering.
2. Welke belangrijke evoluties komen eraan op het vlak van automatisering?

Kunnen

3. Bij een gegeven werksituatie tips voor automatisering kunnen geven.

Hoofdstuk 6

LERENDE MACHINES

- **Zijn computers echt slim of voeren ze enkel geprogrammeerde instructies uit?**
- **Hoe slim is kunstmatige intelligentie?**
- **Hoe leren computers?**

In de vorige les kwam je al te weten wat algoritmes zijn en hoe ze werken. Je maakte ook kennis met een aantal veel voorkomende algoritmes.

Kunstmatige intelligentie. Het klinkt velen angstaanjagend in de oren. Zullen computers en robots ons leven gaan domineren? Is er nog wel een toekomst weggelegd voor ons? Heeft het nog wel zin om te studeren als computers en robots ons werk overnemen? Zal AI straks slimmer zijn dan de mens of is dit nu al het geval?

In deze module kom je te weten hoe ver het staat met de ontwikkeling van kunstmatige intelligentie. Hoe werkt AI nu eigenlijk? Kan je kunstmatige intelligentie nu eigenlijk wel vergelijken met menselijke intelligentie? Werken die beide vormen van intelligentie op dezelfde manier?

Zijn er terreinen waarop AI de mens nooit zal voorbijsteken?

Hoe leren mensen en computers?

Computers volgen stappenplannen (**algoritmes**) om invoer tot de gewenste uitvoer te verwerken. Hoe slim die stappenplannen soms ook wel niet zijn, toch kunnen ze niet tippen aan menselijke intelligentie. Bovendien zijn al die algoritmes eerst door mensen bedacht. Computers slagen er namelijk nog niet in om zelf alle stappenplannen te bedenken of om de best mogelijke oplossingen te bedenken voor problemen.

Mensen kunnen dit wel. Ook al maak je in je leven behoorlijk wat foute keuzes, toch leer je in veel gevallen uit je fouten. We leren uiteraard niet alles door eerst fouten te maken. De meeste kennis en vaardigheden leren we door ervaring of door naar voorbeelden te kijken. Herhaling helpt daarbij, dat weten we allemaal.

Kunstmatig leren

Kunstmatige intelligentie betekent dat computersystemen zelf leren tot oplossingen te komen. We programmeren software om zelf te leren door ervaring of aan de hand van voorbeelden, net zoals mensen en zelfs dieren dit doen.

Computerwetenschappers gebruiken hiervoor allerlei **leeralgoritmes**. Ze proberen bijvoorbeeld de werking van neuronen in de menselijke hersenen te simuleren met software. We spreken in dit geval van een **neuraal netwerk**.

Leren uit voorbeelden

De meeste dingen in ons leven leren we door voorbeelden van anderen. Zelfs op school moeten we leren uit onze fouten.

Stel dat je een kat als huisdier hebt en je zegt telkens tegen je zoontje van twee: “Dit is de kat”. Je zoontje heeft echter nog nooit een hond gezien. De kans bestaat dat hij bij zijn eerste ontmoeting met een hond zegt: “Kijk mama, kat!” Jij zal dan zeggen: “Nee hoor, dat is een hond”.

Door dit vaak te herhalen, zal je zoontje snel het verschil leren tussen een hond en een kat. Hij zal geleidelijk beide leren onderscheiden en even zeer begrijpen dat de schapen van de buren al evenmin honden of katten zijn. Kinderen (ook op school) leren door “supervised learning”. Ze leren uit hun fouten en uit voorbeelden omdat iemand hen daar op wijst.

Door neurale netwerken in software te programmeren, kunnen computers zelf leren (machine learning). We voeren de AI-software eerst met **trainingssets**. Dat kunnen bijvoorbeeld een reeks foto's van honden zijn. Door een massa foto's van honden als trainingssets aan de software te voeren, kan het neuraal netwerk stilaan zelf honden op andere foto's leren herkennen.

Gewicht geven

In ons leven moeten we voortdurend beslissingen nemen. Hierbij wegen we tal van factoren af. Welke kleren zullen we vandaag dragen? Welk geschenk ga ik kopen? Wat ga ik vanavond eten? Voor wie zal ik stemmen bij de verkiezingen? Hierbij maak je heel wat afwegingen en sommige factoren wegen sterker door in je eindbeslissing.

Uiteindelijk tel je al die factoren met elkaar op. Sommige zullen zwaarder doorwegen. Een neuraal netwerk (of het nu gaat om een stel neuronen in onze hersenen of om een kunstmatig neuraal netwerk) krijgt op die manier heel wat "inputs" (afwegingen) die elk hun eigen gewicht krijgen. Het netwerk "berekent" een beslissing en stuurt die uit als "uitvoer". **Opgelet**: neurale netwerken zijn niet de enige methode voor kunstmatig leren (zie hoofdstuk 8*)*

Als je twijfelt of je de fiets of de auto zou nemen om naar je werk te gaan, dan weeg je heel wat factoren af: het weer, de afstand, vermoeidheid, kostprijs, of je nog naar de winkel moet… De ene dag neem je de auto, de andere dag weer niet. Je uiteindelijke beslissing (output) hangt van heel veel inputs af. De neuronen in je hersenen maken heel wat (statistische) afwegingen. Misschien neemt je collega wel de fiets, terwijl jij de auto neemt. Hij/zij heeft misschien een andere afweging gemaakt. Als je echter de ene dag strontnat werd door een fikse regenbui, leer je in de toekomst eerst het weerbericht te bekijken voor je op je fiets stapt.

1. Scan de QR-code
2. Bekijk de films
3. Maak de bijhorende opdracht.

Leren onder toezicht

Het is niet omdat je een uitgebreide dataset/trainingsset aan je AI-algoritme voedt, dat het vanaf dat moment foutloos werkt. Een AI-algoritme kan leren uit haar fouten. In deze fase is menselijke tussenkomst nog van belang. De mens kan zelf nog "controleren" of de beslissing van het algoritme correct is. We spreken van **gecontroleerd leren** of **supervised learning**.

Een spamfilter is een bekend voorbeeld van AI-software op basis van een neuraal netwerk. Heel wat mails die het woord "casino" en "penisverlenging" bevatten, zullen spamberichten zijn. Maar stel dat je vriend je een mail stuurt met de melding "Ga je vanavond mee naar het casino?" dan wil je niet dat dit bericht in de spam terechtkomt. Of als je echt problemen hebt en je huisdokter stuurt je een mailtje over je voortplantingsorgaan met daarin het zinnetje "we gaan je ziekteverlof verlengen", dan wil je niet dat de combinatie van "verlengen" en "penis" in hetzelfde bericht tot "spam" leidt. Het neuraal netwerk moet heel wat afwegingen maken om te besluiten of een bericht al dan niet als spam gemarkeerd moet worden.

Zo'n spamfilter bouw je door een reeks van verdachte woorden samen te stellen. Berichten waarin een of meer van die woorden voorkomen, moeten al dan niet als spam worden herkend. Die verdachte woorden krijgen in het ene bericht de classificatie "true negative" (geen spam), in het andere "true positive" (wel degelijk spam).

Andere termen kunnen leiden tot de classificatie "false positive" of "false negative". Dit gebeurt wanneer een spambericht onverwacht toch in je gewone mailbox terechtkomt of een goed bericht in de spam. In dat geval kan je een bericht zelf classificeren. Op dat moment leert de spamfilter dat hij een fout heeft gemaakt. Daar zal hij in de toekomst rekening mee houden. Spamfilters worden op die manier steeds beter en maken minder en minder fouten.

Daarom kan je zeggen dat AI-software letterlijk voorspellingen doet op basis van wat de software eerder is geleerd. Die voorspellingen kunnen ok zijn (true positive, true negative), maar even zeer fout (false positive of true negative). In dit laatste geval moet het algoritme bijgestuurd worden.

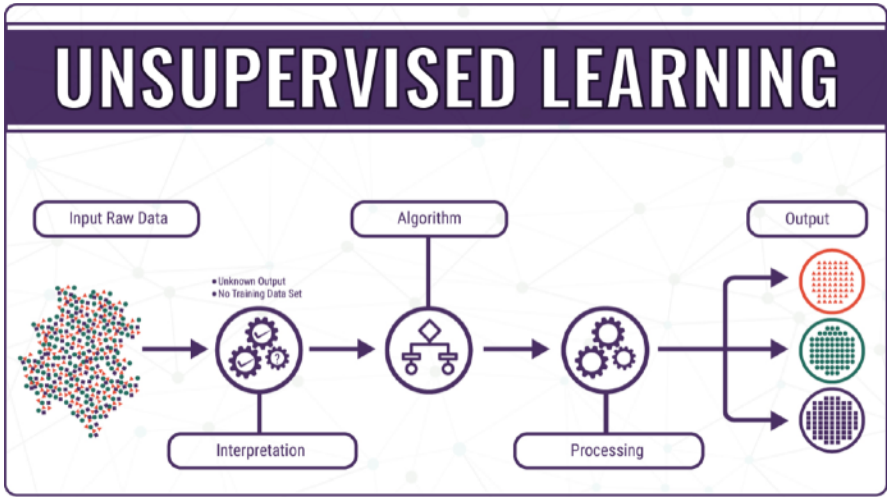
Leren ZONDER toezicht

Bij **supervised learning** leert het algoritme uit voorbeelden. Bij **bekrachtigingsleren (reinforcement learning)** bekrachtig je het algoritme door het te belonen of te bestraffen. Bij ongecontroleerd leren (unsupervised learning) krijgt het algoritme geen voorbeelden, maar moet het zelf (bijvoorbeeld) elementen indelen (categoriseren) op basis van gemeenschappelijke kenmerken.

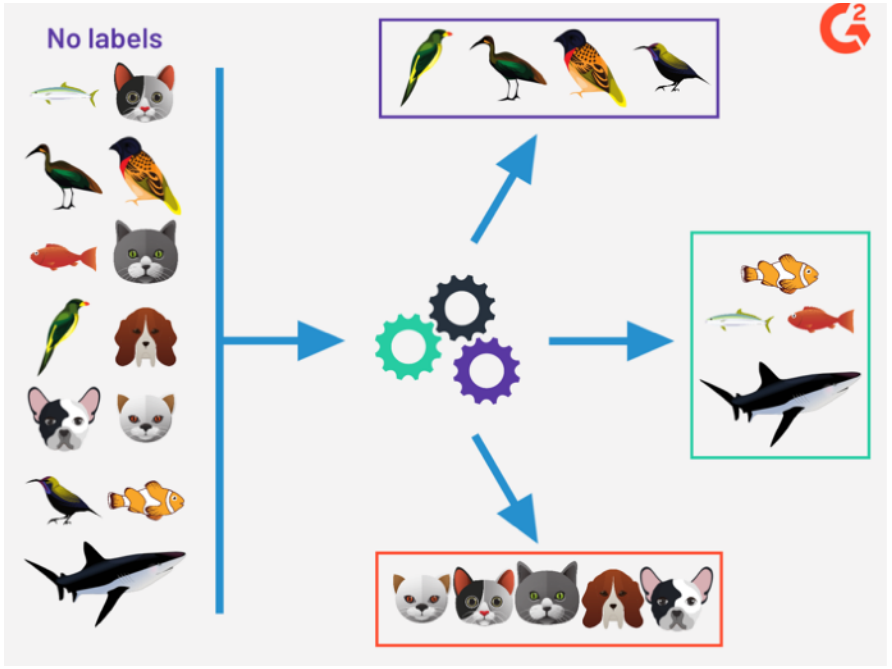
Stel dat je voor het eerst in je leven de was moet doen. Je ziet een berg ongewassen kleren voor je liggen. Je moet de was nu indelen op basis van een aantal kenmerken. Indelen op basis van kleur en textielsoort zou de gewenste output zijn. Maar dat is wat het "algoritme" nu dient te ontdekken. Als je het K-means-algoritme gebruikt, kan je dit vooraf enkel vertellen dat je tot drie groepen wil komen. De rest moet het algoritme zelf zien te vinden.

Bij unsupervised learning krijgt het algoritme vooraf dus geen voorbeelden of labels. Het algoritme moet de informatie zelf indelen op basis van kernmerken.

In hoofdstuk 8 (AI en data-analyse) leer je meer over de diverse AI-algoritmes.



Bron: https://medium.com/@chisoftware/supervised-vs-unsupervised-machine-learning-7f26118d5ee6



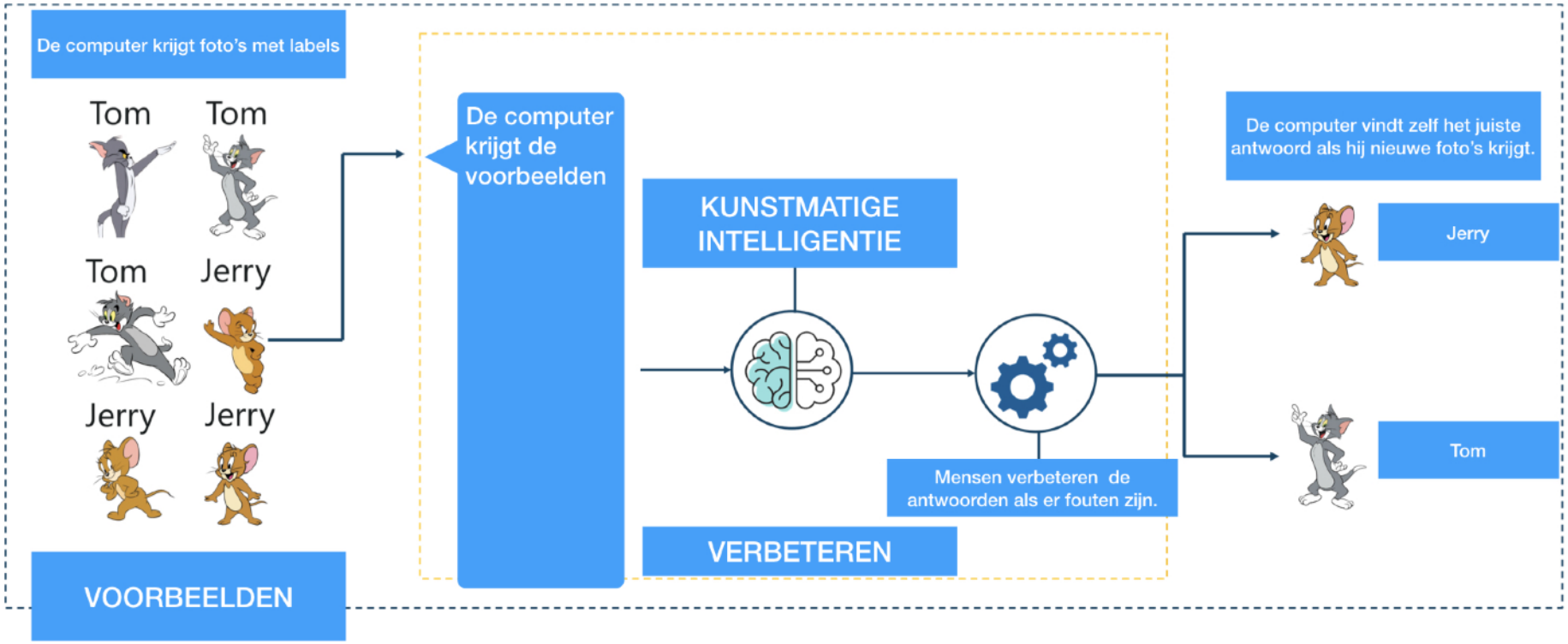
Bron: https://learn.g2.com/supervised-vs-unsupervised-learning

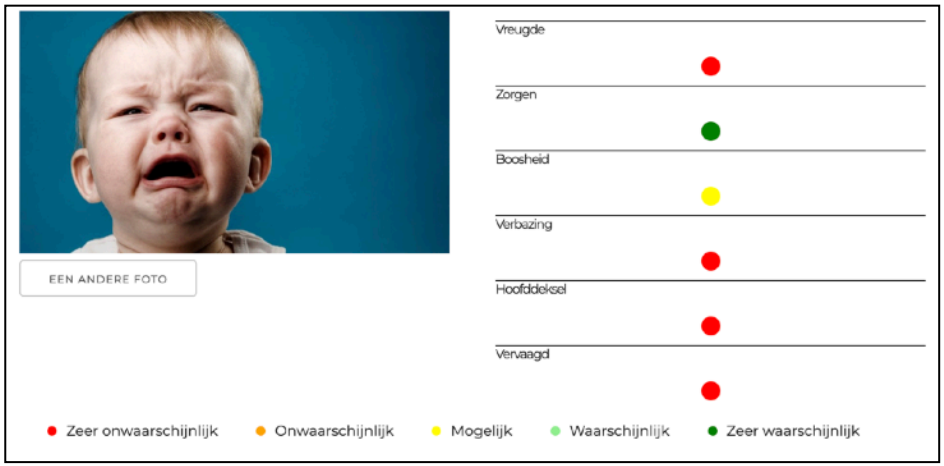
Hoe kunnen machines leren?

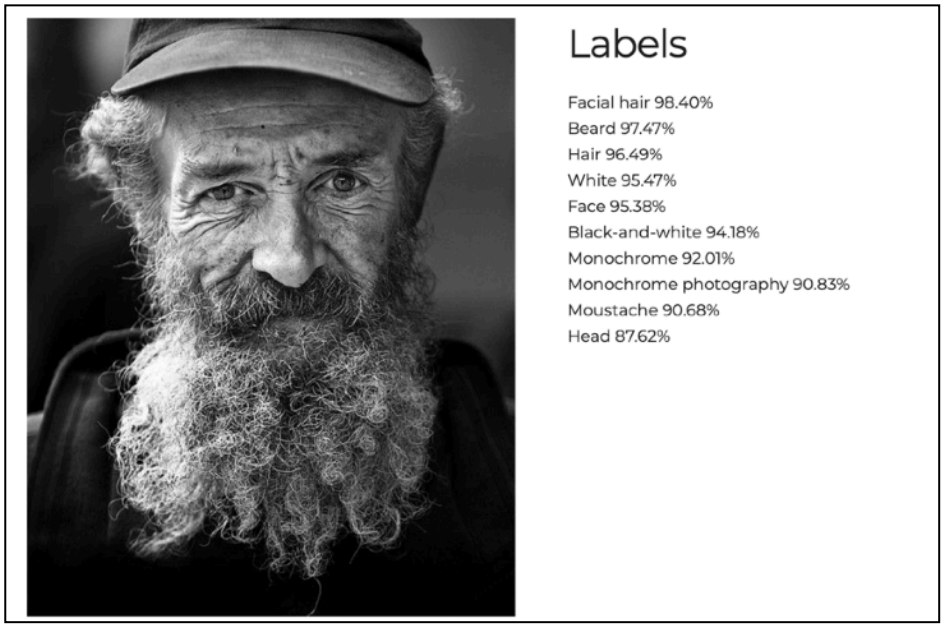
Dat is niet gemakkelijk uit te leggen. Vaak proberen programmeurs een stukje van de menselijke hersenen na te bouwen. Ze bouwen met codes een hersencel (neuron) na. Zoiets noemen ze een **neuraal netwerk**.

Soms leert kunstmatige intelligentie door te **vergelijken**. Als je naar een serie kijkt op Netflix, vergelijkt Netflix jouw voorkeuren en selecties met die van andere mensen die naar dezelfde serie keken. Netflix suggereert je series waar die andere mensen ook naar gekeken hebben.

Kunstmatige intelligentie leert door **kansen** te **berekenen**. Bijvoorbeeld: je hebt 50% kan dat de eerste leerling die je op school tegenkomt een jongen is en 50% dat het een meisje is. Maar in een nonnenklooster is er vrijwel 100% kans dat het een vrouw is. Een computer zou dat niet weten omdat hij die **context** niet weet. Als je die informatie aan de AI-software geeft, zal hij een correctere voorspelling doen. In hoofdstuk 8 leer je meer over diverse AI-algoritmes.







Een AI-algoritme op basis van de Google Vision API herkent de emoties. Het doet dit met "procentuele" waardes, in dit geval uitgedrukt met kleurwaardes.
(Bron: www.schoolvoorbeeld.be/nodig/cloudvision - Kris Merckx)

Een neuraal netwerk berekent statistische kansen.
(Bron: www.schoolvoorbeeld.be/nodig/cloudvision - Kris Merckx)

AI is beperkt

Toch moet elk stukje AI-software, elk neuraal netwerk nog door mensen worden geprogrammeerd. We moeten ze nog voeden met trainingssets: **supervised learning**. Een spamfilter is sterk in het herkennen van spam, maar kan geen zelfrijzende auto besturen. De software van zo'n auto kan op tijd stoppen voor een overstekend ree, maar kan je niet gebruiken voor een vertaalopdracht. **Elk stukje AI-software heeft namelijk nog zijn eigen specialiteit.**

Daarom kan je gerust stellen dat de meeste AI beperkt is, en bestempelt men de meeste AI-algoritmes als "**Narrow AI**".

De veroveringstocht van AI

Toch is AI aan een niet te stuiten opmars begonnen. Amazon, Netflix en Spotify geven hun gebruikers suggesties (**recommendations**) op basis van het gedrag van andere bezoekers. Zelfrijdende auto's voorspellen het rijgedrag van de auto's in hun buurt om ongelukken te voorkomen. Weersvoorspellingen op basis van AI zijn accurater dan ooit tevoren. Digitale assistenten zoals Google Home en Amazon Alexa begrijpen (sommige) gewone vragen en proberen daarop zinvol te antwoorden. Google Translate vertaalt elke dag een beetje beter.

Algemene kunstmatige intelligentie

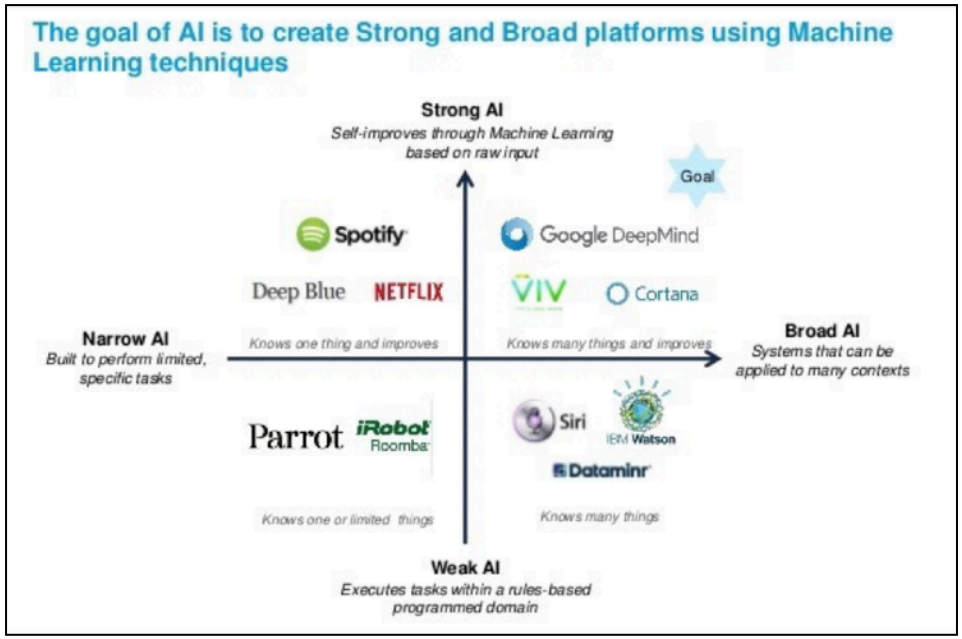
Wie bang is dat AI de mens is voorbijgestoken, kunnen we (voorlopig) gerust stellen. Ondanks het buitensporig enthousiasme en de talloze persberichten, zijn we nog een eind verwijderd van een AI-algoritme dat zonder toezicht **(unsupervised learning)** elk probleem kan oplossen (**broad AI of Algemene Kunstmatige Intelligentie**).

Toch vordert de ontwikkeling van unsupervised learning en **deep learning**: algoritmes die zelf problemen leren oplossen, zonder hiervoor eerst trainingssets te doorspitten. Een van de rijzende sterren achter deep learning is **Demis Hassabis** en zijn bedrijf **Deepmind** dat werd ingelijfd door Google. Deepmind scheerde hoge toppen toen het AI-programma **AlphaGo** de grootmeester in het Chinese Go-spel versloeg.

> Om je een idee te geven: bij Go kan een speler meer zetten doen dan er atomen in het heelal zijn. De overwinnende zet die AlphaGo deed, verbaasde Go-spelers wereldwijd. Geen mens zou ooit op zo'n creatief bedachte zet zijn gekomen.

Wordt de mens overbodig?

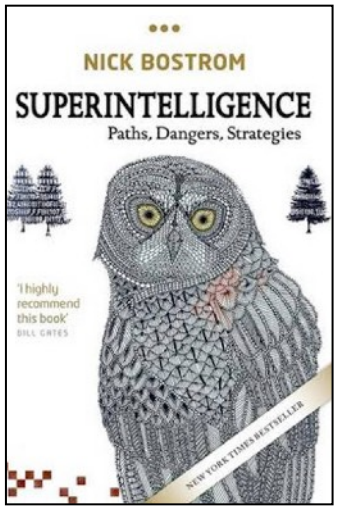
Automatisering maakte in het verleden handenarbeid overbodig. AI kan taken waar voordien denkwerk aan te pas kwam, automatiseren. Sowieso zal AI-automatisering heel wat jobs en



opleidingen overbodig maken. Over welke jobs het eerst zullen verdwijnen, hoor je tegenstrijdige berichten. Over het algemeen gelooft men dat jobs die meer expertise of creativiteit vragen, langer zullen overleven, evenals gespecialiseerde handenarbeid.

(Bron: AMP new ventures)

Het boek "Superintelligence" van de Britse computerwetenschapper Nick Bostrom



Februari 2018. Volgens The Economic Times van India vertelde Dr. David Hanson van Hanson Robotics, op het World Congress on Information Technology in Hyderabad, dat robots "would be alive and have full consciousness in five years".

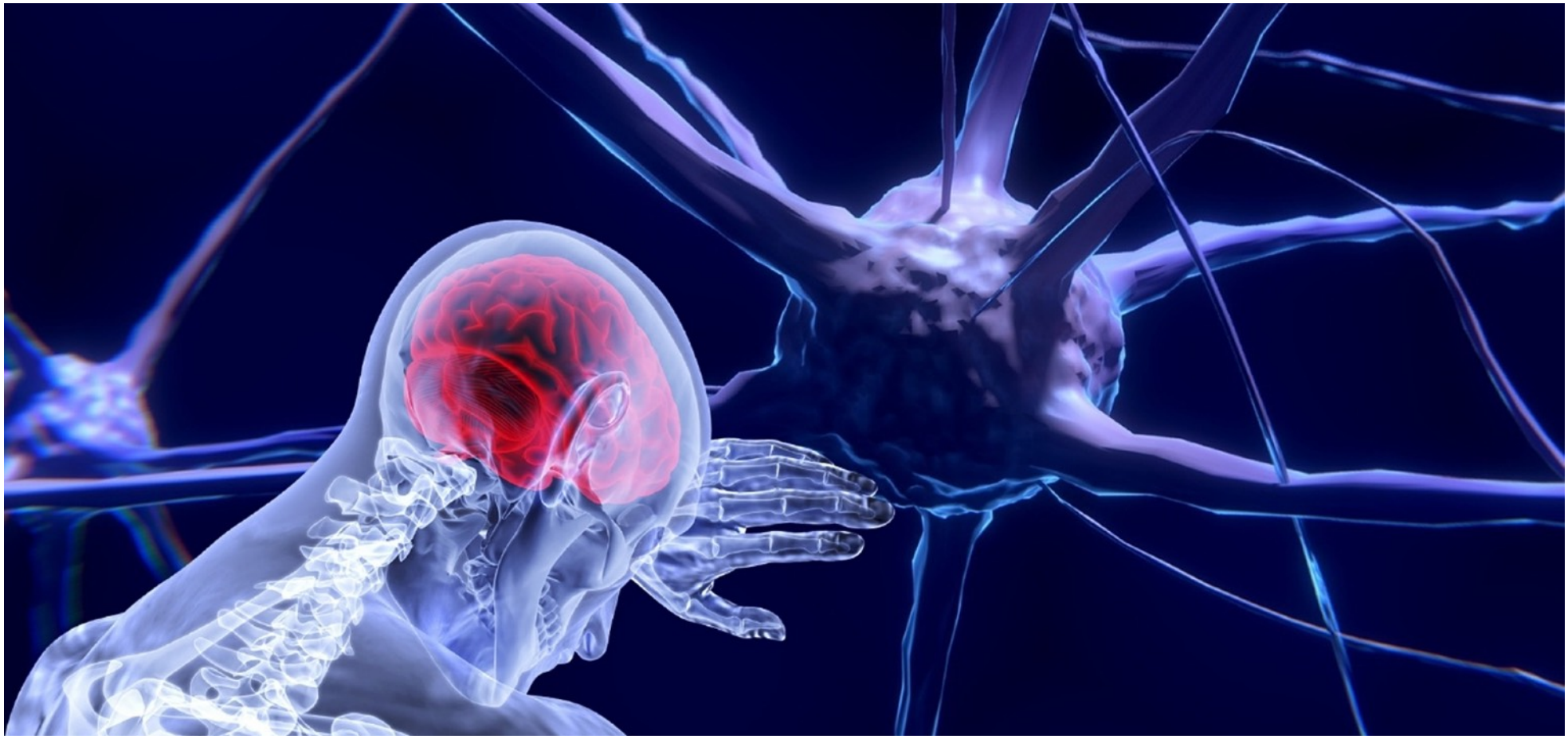
De singulariteit

Wanneer een computer menselijke intelligentie bereikt of het niveau waartoe mensen na miljoenen jaren biologische evolutie zijn aanbeland, dan stopt het voor AI niet. Nick Bostrom, Ray Kurzweil, Bill Gates, Elon Musk... waarschuwen voor dit schakelmoment: de **singulariteit**.

Kunstmatige intelligentie zal zich, volgens sommigen, vanaf dat moment in razend tempo blijven verder ontwikkelen. Menselijke intelligentie is volgens hen geen eindpunt, maar een kantelmoment waarna menselijke intelligentie snel overbodig lijkt te worden, als we hen mogen geloven.

Voor Ray Kurzweill is die "singulariteit" het moment waarop mens en machine in elkaar zullen opgaan en er hybride overgangsvormen zoals humanoïden of cyborgs zullen ontstaan.

Bostrom, Gates en Musk roepen overheden op om goed na te denken en nu al "regels" en "beperkingen" in te stellen. Eén van de eerste resultaten van die oproep is de "internationale" bereidheid om een halt toe te roepen aan de ontwikkeling van autonome robots die op het "slagveld" en bij oorlogvoering zullen ingeschakeld worden.



Kennen

1. Hoe kunnen machines leren?
2. De verschillen tussen supervised en unsupervised learning uitleggen.
3. Wat is het nut van trainingssets?
4. Waarvoor dienen kunstmatige neurale netwerken?
5. Kort uitleggen hoe neurale netwerken functioneren.

Kunnen

6. Bepalen wanneer het zinvol zou kunnen zijn om een kunstmatig neuraal netwerk in te zetten.
7. Zelf een planmatige aanpak bedenken om een supervised neuraal network uit te bouwen.

Hoofdstuk 7

AI EN HET BEGRIP VAN BEELD EN TAAL

- **Waarom hebben computers en zelfs AI het moeilijk met menselijke taal?**
- **Waarom is het begrijpen van de inhoud van afbeeldingen niet evident voor AI?**

Ook al kan een computer iedere denkbare afbeelding weergeven, begrijpen wat op een afbeelding staat, is nog een ander paar mouwen.

Maar ook het begrijpen van menselijke taal, loopt niet van een leien dakje. Computers hebben het moeilijk met menselijke taal. Zeggen dat dit komt omdat computers enkel binaire getallen kennen, is wat te simplistisch. Het klopt slechts voor een stuk. Als je een tekst invoert op je toetsenbord, vertaalt de computer elke toetsaanslag naar een code. Elke toets op een toetsenbord krijgt intern een cijfer toegewezen. Een computer begrijpt dus niet wat je invoert.

Maar het probleem is veel ruimer. Wereldwijd bestaan er ongeveer 6500 menselijke talen, elk met hun eigen woordenschat, spelling en spraakkunst. Elke taal kent in de vorm van streekdialecten nog diverse varianten. Ook al zijn heel wat talen verwant met elkaar (bijvoorbeeld de Indo-Europese talen), dan nog is het niet eenvoudig om software al die talen te leren begrijpen. Naast het westerse alfabel bestaan nog andere lettersystemen.

Bovendien verschillen de geschreven en gesproken versies van één taal. Naast het begrijpen van een taal, is het genereren van taal nog een ander heikel punt.

Wanneer een docent je op een examen een vraag stelt, dan kan je daar in het beste geval vlot op antwoorden. Je begrijpt de vraag. Je maakt in je hoofd de juiste analogie en vergelijkingen. Je kan de inhoud simultaan *vertalen* in je eigen woorden. Je leert het niet uit je hoofd en je rammelt het niet woord voor woord betekenisloos af. Dat zou je natuurlijk kunnen doen, maar een docent verwacht dat je intrinsiek verbanden kan leggen en de betekenis snapt.

Sommige woorden hebben meerdere betekenissen ('haar' of 'bal' bijvoorbeeld). Soms kan je de betekenis van een woord pas afleiden uit de context (de rest van de zin of tekst).

> De hond slaapt in zijn mandje. Jacky is heel erg lief. Als ik thuis kom, loopt hij al kwispelstaartend naar me toe. Waar blijft hij vandaag? Weet jij het? Ik ben zo bang dat hij weggelopen is.

Lees het kleine verhaaltje hier boven: Als menselijke lezer snap je meteen waarover dit korte tekstje gaat. Hoe kunnen we een machine leren om deze tekst te begrijpen? Tussen mens en machine is er in dit geval al meteen een immens verschil. Een kind leert al snel een visueel beeld van een hond te koppelen aan het begrip "hond" en de bijhorende geluiden. Denk maar aan de mama's en de papa's die vragen: "En, hoe doet de hond? Nee, niet miauw."

Het taalmechanisme in het hoofd van het kind koppelt aan de hond bepaalde acties zoals blaffen, lopen, kwispelstaarten, maar ook onderdelen zoals staart, poten, tong. Een heel kader, onderdeel van de "wereldontologie" in het hoofd van het kind. Maar hoe vertalen we die taalverwerking naar een computer? Oeps, dat is wat anders.

Laat ons eerst eens de tekst scheiden in zijn onderdelen: zinnen (**sentence breaking**). Dat kan eenvoudig door te zoeken naar leestekens zoals punten, vraagtekens of uitroeptekens. Dit werkt perfect voor heel wat talen, maar niet voor pakweg Arabisch, Chinees of Japans, want daar is het gebruik van leestekens onbekend. Vervolgens kunnen we de zinnen splitsen in hun diverse woorden (**word segmentation**), maar dat werkt al evenmin voor de genoemde talen: Arabisch kent bijvoorbeeld geen door spaties onderscheiden woorden. Daar bepaalt het uitzicht van een bepaald karakter (letter) of het gaat om het begin, het midden of het einde van een woord.

Zinnen bepalen of ze iets meedelen of eerder iets vragen (het vraagteken), en vragen op hun beurt kunnen open of gesloten (ja of neen, if of else, 1 of 0) zijn. De soort zinnen wordt bepaald door de "discourse analysis".

De machine kijkt vervolgens naar de individuele woorden en probeert die te herleiden tot hun basis of stam in een proces dat "**stemming**" en "**morphological segmentation**" heet. In een taal als het Engels is dit een relatief eenvoudig proces:

open is de "stam" van open, opens, opening, opened…

In talen zoals het Turks is dit een uiterst moeizame techniek. De stam is niet noodzakelijk hetzelfde als de morfologische "wortel" van het woord. Meestal volstaat het dat verwante woorden dezelfde stam bevatten. Heel wat zoekmachines behandelen woorden met dezelfde stam als synoniemen om de zoekterm enigszins uit te breiden (conflatie). Google gebruikt "stemming" sinds 2003. Voordien zou de zoekterm "vis" niet "vissen" of "gevist" opgeleverd hebben.

Dit leidt soms tot foute resultaten. De machine kan teveel resultaten leveren (overstemming) zoals wanneer je "universiteit" zoekt en ook "universum" krijgt. In dit geval behandelt de machine beide woorden door hun gemeenschappelijke stam als synoniemen. In andere gevallen krijg je te weinig zoekresultaten (understemming).

Bij woorden zoekt het NLP-systeem niet alleen naar de stam, maar bovendien ook naar de woordsoort: gaat het om een adjectief of een zelfstandig naamwoord? Bovendien kunnen zelfstandige naamwoorden ook eigennamen zijn.

Jacky *is een hond.*

Deze techniek van "**named entity recognition**" werkt simpelweg door naar hoofdletters te kijken. Maar ook dit levert problemen op. Bij het begin van een zin schrijf je steevast een hoofdletter. In het Duits krijgen dan weer alle naamwoorden een hoofdletter.

Jacky *ist ein* ***Hund****.*

En in het Engels schrijf je ook een hoofdletter als je het over jezelf hebt:

I*,* ***I*** *wanna be your dog.*

De woordsoort wordt vaak bijgehouden in een soort lexicon, een soort lijst waarin op basis van de stam en eventuele voor- en achtervoegsels de woordsoort wordt bepaald (**Part-of-speech tagging**).

Maar eenvoudig is dit niet. Immers, afhankelijk van de zin kan een woord iets compleet anders betekenen:

Ik speel met de bal op het bal.

Je merkt het al. De betekenis zit niet in het woord alleen, maar ook in zijn relatie met andere woorden: homoniemen (**word sense disambiguation**). De volgende zin klinkt afhankelijk van tegen wie (je docent) of wat (een eik) je het zegt helemaal anders, maar een NLP-systeem heeft het er moeilijk mee:

Je bent een eikel.

Via de techniek van "**natural language understanding**" probeert een NLP-systeem logica te vinden in korte fragmenten tekst. Welke adjectieven horen bij zelfstandige naamwoorden (conference resolution)? Wat zijn de onderlinge relaties tussen objecten in een zin (relationship extraction)? Op die manier probeert een NLP-systeem stilaan de zin te "parsen", een stamboom te maken van een zin en tenslotte van het hele verhaal.

> "Je bent een eikel," zei ik tegen de directeur. Ik was woedend. Hij werd door mijn uitspraak eveneens razend.

Deze zin verklapt hoe beide partijen zich voelen. Een NLP-systeem kan op basis van de betekenis van woorden en de frequentie waarmee ze voorkomen, raden wat het gevoel is dat uit een tekst spreekt: **sentiment analysis**.

Spelling- en spraakkunstcorrectie

In heel wat tekstverwerkers zit spelling- en spraakkunstcorrectie. MS Word bijvoorbeeld duidt met een rood lijntje aan waar je woorden fout hebt gespeld. De programmeurs hebben de spellingregels in de software geprogrammeerd. Open bijvoorbeeld eens een Duitse of Franse tekst in een tekstverwerker die in het Nederlands is ingesteld. Dan lijkt het alsof je alles fout hebt geschreven. Dit komt omdat op dat moment enkel de Nederlandstalige spellingcorrectie is geladen.

Dankzij de evolutie van **AI (artificiële intelligentie)** heeft **NLP (natural Language processing)** grote stappen voorwaarts genomen. Spelling- en spraakkunstregels worden niet langer per taal gecodeerd (rule based). De software leert uit andere teksten zelf hoe de regels in elkaar zitten. Het helft de computer ook om de inhoud van de teksten **semantisch** te begrijpen.

Robotjournalisme

Tekstbegrip kan dan voor digitale systemen nog relatief moeilijk zijn, toch zijn steeds meer artikels niet meer door mensen geschreven. Google kocht Jetpac, een app die op basis van image recognition-algoritmes automatisch stadsgidsen genereert. Associated Press laat duizenden artikels schrijven door robotschrijvers.

Betekent dit dat het einde van de "menselijke journalist" in zicht is? Het onderstaande artikel werd door een robotjournalist geschreven:

Aerie Pharmaceuticals Inc. (AERI) on Tuesday reported a loss of $13.1 million in its third quarter. The Research Triangle Park, North Carolinabased company said it had a loss of 54 cents per share. Losses, adjusted for stock option expense, came to 44 cents per share.

Het artikel voldoet aan de vereisten voor dit soort artikels: het is correct en gedetailleerd. Uiteraard moet je je bij een robotjournalist niet een humanoïde robot voorstellen die op een toetsenbord teksten zit in te tikken. Het gaat om een stuk software dat "natuurlijke taal" produceert. Auteur van dienst is in dit geval WordSmith van de softwareproducent Automated Insights. In 2013 produceerde WordSmith wereldwijd 300 miljoen nieuwsverslagen voor diverse klanten, meer dan alle "menselijke" journalisten samen en vooral veel goedkoper...

Volgens Automated Insights komen de jobs van echte journalisten niet in gevaar:

"We're producing articles that never would have existed in the first place," he says. "AP was doing 300 corporate-earnings stories per quarter; they're now doing about 4,440. So 4,100 of these stories would not exist without Wordsmith." "The computer handles the who, what, where and when, and humans are freed up to ask why and how."

De LA Times zet Quakebot in om geautomatiseerd artikels af te leveren op basis van data van de US Geological Survey (aarbevingen). Inleving, gevoel, medeleven.... zijn vreemd bij dit soort artikels, maar dat is voor "objectieve" verslaggeving ook niet nodig. "Creatief schrijven" ligt voorlopig nog buiten de mogelijkheden, want de algoritmes voeden zich met gestandaardiseerde informatie, een beetje te vergelijken met gestructureerde data in een rekenbladprogramma. Een van de mogelijke pistes om meer menselijke "warmte" in een tekst te steken is door de data aan te vullen met informatie geleverd door sensoren.

Vaak hoor je dat Wikipedia-informatie onbetrouwbaar is. Nochtans zijn een massa artikels niet meer door menselijke auteurs afgeleverd maar door software-algoritmes. De meest bekende Wikibot luistert naar de naam Lsjbot, geprogrammeerd door de Zweed Sverker Johansson. Lsjbot schraapt informatie van betrouwbare bronnen en schrijft korte artikels over onderwerpen gerelateerd met dieren. Per dag levert de bot ongeveer 10000 artikels af. In totaal zouden ongeveer 8.5% van de artikels van de (Zweedse) Wikipedia geschreven zijn door Lsjbot.

Wikipedia zette voor het eerst een bot in in 2002. Die "Rambot" leverde per dag duizenden artikels op over zo wat elk(e) dorp, stad of staat van de Verenigde Staten en ook van een aantal andere landen. Een robot onder de gepaste naam Asteroids schraapte data van de NASA en leverde artikels over... asteroïden. Op dit moment zijn er meer dan duizend verschillende Wikibots aan het werk: ze verbeteren teksten en tegenstrijdigheden... De meest actieve is Cydebot die meer dan 4.5 miljoen aanpassingen aanbracht.

Toch getuigt de techniek voor het geautomatiseerd schrijven van artikels niet altijd van “kunstmatige intelligentie”. Quakebot gebruikt bijvoorbeeld door mensen opgestelde sjablonen en vult die met vooraf verzamelde en gestructureerd bewaarde data over aardbevingen. Het eindresultaat klinkt als een door mensen geschreven artikel, en dat is het voor een groot deel natuurlijk ook. Maar de robot doet in wezen niet veel meer dan het vullen van sjablonen met gestructureerde data. Andere ontwikkelaars, zoals Narrative Science proberen robotjournalisme naar een ander niveau te tillen. Zij pogen de software zelf te laten uitmaken waarover de data gaan en op basis daarvan “content” te laten genereren. De software beslist in dit geval zelf hoe de inhoud moet klinken.

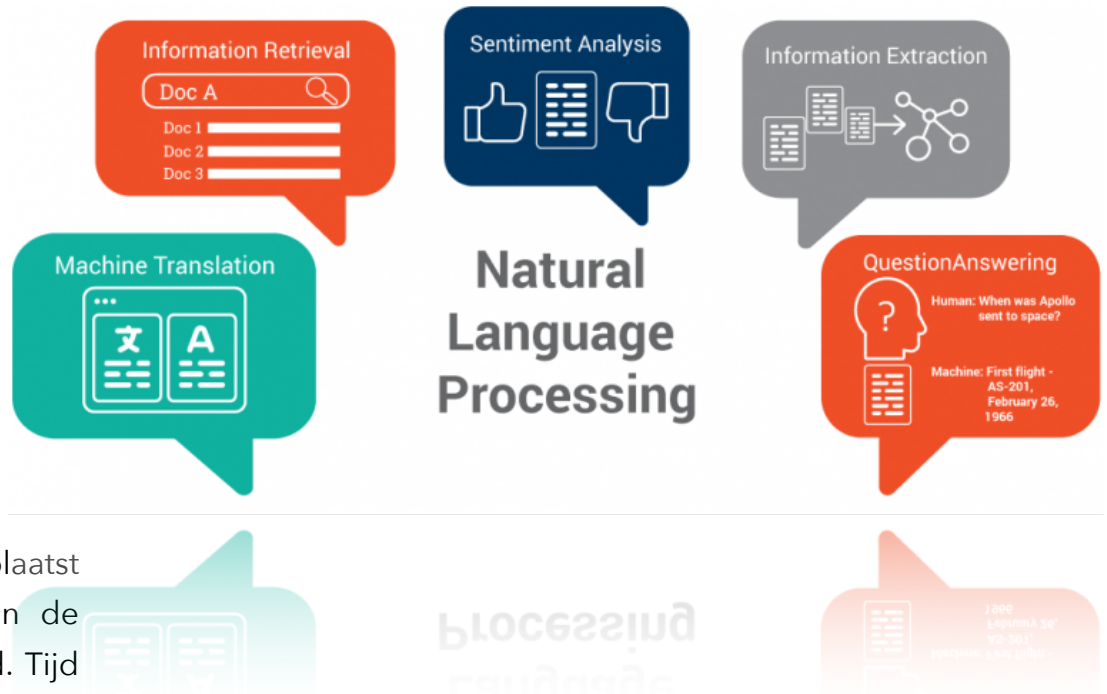
Gesproken tekst en spraaksynthese

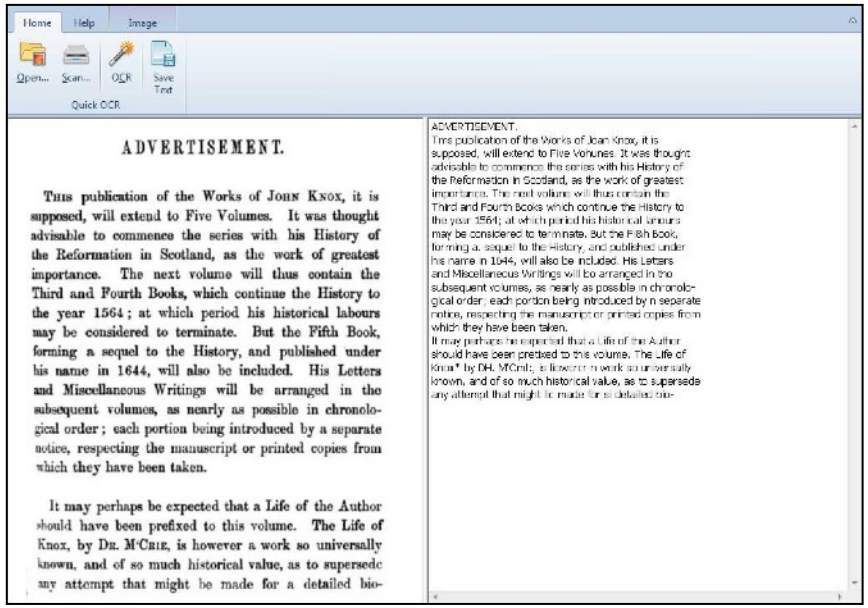
Vooraleer machines vlekkeloos in menselijke talen kunnen communiceren, is er nog een hele weg af te leggen. Het draait niet enkel om de semantiek. In de Studio 100-reeks ROX praat de gelijknamige auto vlekkeloos met de andere (menselijke) personages. Hij herkent en interpreteert hun vragen en antwoorden en reageert er vaak laconiek op. Onze vriend “ROX” combineert het beste op het vlak van NLP met speech recognition, speech segmentation en speech processing en als kers op de taart **natural language generation**.

Het verstaan van gesproken taal (speech recognition) is nog een aardig stuk moeilijker dan het lezen of interpreteren van gedrukte tekst omdat er in gesproken tekst nauwelijks pauzes zijn tussen woorden. Ook letters vloeien onhoorbaar in elkaar over. Het verdelen van een uitspraak in afzonderlijke woorden (speech segmentation) is vreselijk moeilijk te programmeren.

Natural language generation, het genereren van taal in geluidsvorm, is daarentegen al aardig ingeburgerd in bijvoorbeeld GPS-systemen. De software kan in dat geval gebruik maken van vooraf opgenomen geluid (afzonderlijke woorden of zinnen) dat het tot de gewenste zinnen samenvoegt. Dat is echter nog heel wat anders dan het kunstmatig "toon per toon" genereren van spraak (**speech synthesis**).

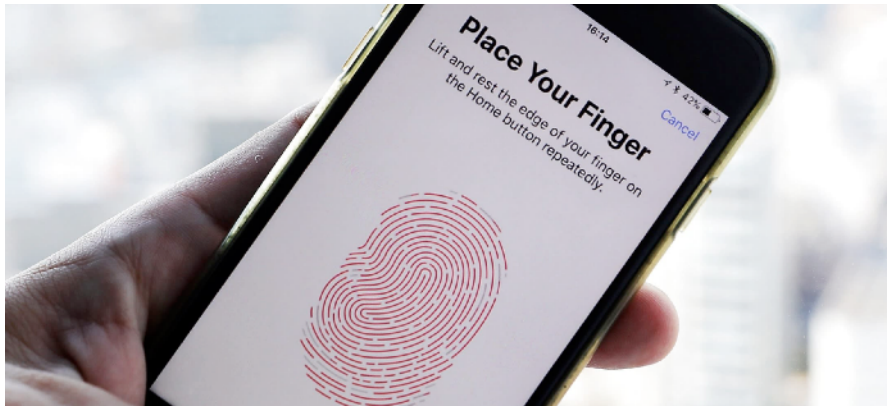
In machine translation komen alle toeters en bellen van NLP-systemen en natural language generation samen. Zinvolle vertalingen laten gebeuren door een machine is dus allesbehalve evident. Het vraagt niet alleen begrip van beide talen, maar ook het betekenisvol begrijpen en converteren van het ene bekenissysteem (taal) naar het andere (een andere taal). Een voorbeeld: In het Chinees plaatst een spreker geen "tijd" in zijn zin. Je kan dus niet aan de persoonsvorm zien wanneer de actie zich heeft afgespeeld. Tijd heeft in het Chinees geen belang wanneer je schrijft of spreekt, een volledig andere manier van "denken" dan sprekers van Indo-Europese talen. Allesbehalve eenvoudig dus om te vertalen van pakweg het Chinees naar het Nederlands of omgekeerd. Denk dus zeker nog eens na wanneer mensen spotten met de vertalingen van bijvoorbeeld Google Translate. Google Translate werkt bovendien met machine learning. De software leert dus voortdurend zelf bij en wordt alsmaar beter.





Sjabloon-gebaseerde OCR.

(Bron: https://www.malavida.com/en/soft/supergeek-free-document-ocr/#gref)



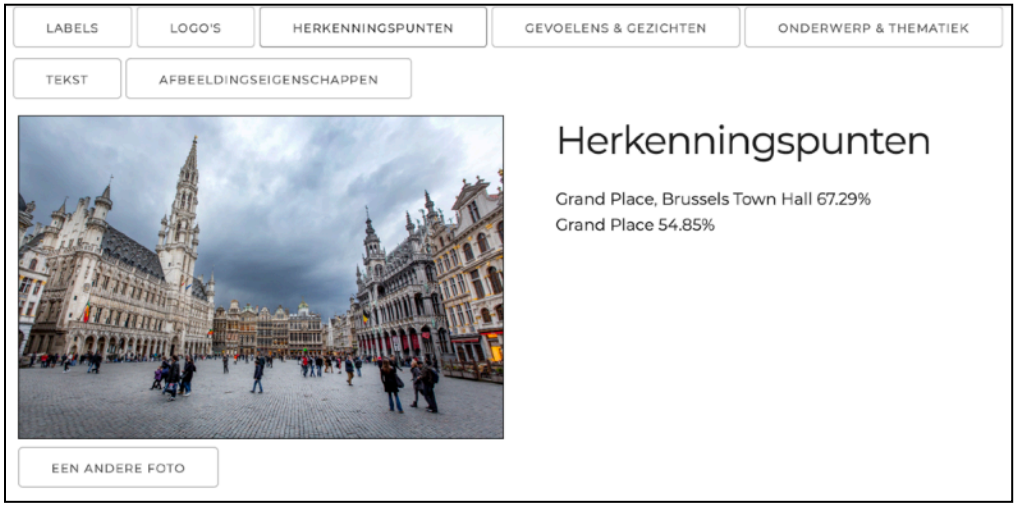
(Bron: https://asia.nikkei.com)

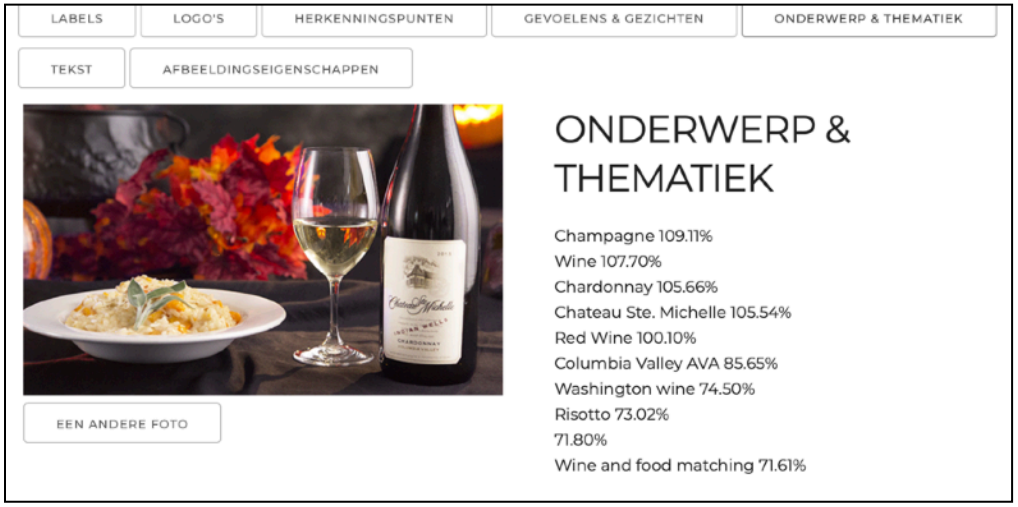
Image recognition

Je smartphone beschikt doorgaans over minstens één camera, veel laptops zijn uitgerust met een webcam. Maar dit betekent nog niet dat die toestellen kunnen zien zoals mensen of dieren. Met onze ogen kunnen we niet enkel zien, maar we herkennen ook objecten. Willen we een computer informatie in beelden leren herkennen, dan spreken we van **computer vision** en **image recognition**.

In de warenhuizen "herkennen" scanners streepjescodes. De meeste smartphones herkennen informatie in QR-codes. Smartphones herkennen het gezicht of de vingerafdruk van de gebruiker. Om de ruwe beeldinformatie te herkennen, is software nodig. Net zoals het bij mensen niet de ogen zijn, maar de hersenen die objecten en mensen herkennen.

Streepjescodes, QR-codes en tekstherkenning zijn reeds relatief oude technieken. De software herleidt het beeld tot een zwart-witafbeelding. Bij **OCR (optical character recognition** vergelijkt de software de rasters van zwarte pixels met bestaande letters. Op die manier kan OCR-software teksten herkennen. Dit betekent echter niet dat de software snapt waarover de tekst gaat





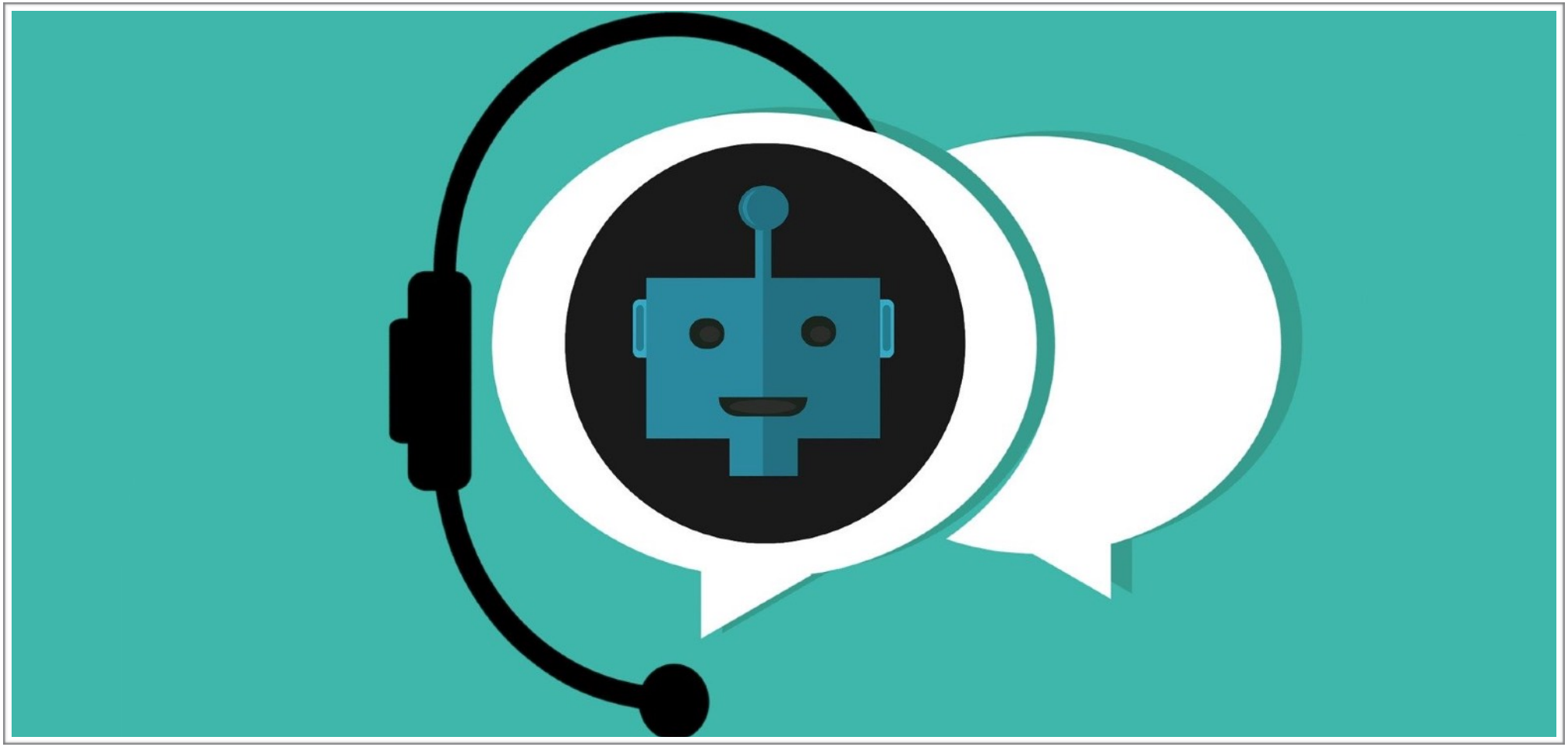
Om computers echt te leren herkennen wat op een afbeelding staat, gebruikt men **AI (kunstmatige intelligentie)**. Net zoals een klein kind de naam moet leren van alle objecten en levende wezens in de fysische wereld, moet AI-software stapsgewijs elementen leren herkennen. Als je een kind vaak genoeg bepaalde letters toont, zal het die letters leren kennen. Net zo gaat het met AI. Klassieke OCR kan geen letters herkennen die ondersteboven zijn afgedrukt, maar een kind of AI kan dit wel.

In het onderdeel over AI kom je te weten dat AI beelden kan herkennen door de werking van neuronen (in de hersenen) na te bootsen. Een "**neural network**" leert via vooraf gelabelde afbeeldingen, gelijkaardige afbeeldingen herkennen.

Als je een stuk software een hond wil leren herkennen, dan moet je de software eerst voeden met honderden of duizenden afbeeldingen van honden. Zo leert de software stapsgewijs dat zowel een Pekinees als een Pitbull honden zijn. Bovendien moet de software leren dat zowel Disney's Pluto als Lassie een hond is.

Enkel als een **robot** snel en accuraat objecten kan herkennen, kan hij (zij?) net zoals echte mensen "rondlopen". *Zelfrijdende auto's bijvoorbeeld gebruiken AI-gebaseerde computer vision en image recognition om voetgangers, verkeersborden, voertuigen… te herkennen.*

AI-gebaseerde computer vision. De software maakt gebruik van de Google Vision API.
(Bron: https://www.schoolvoorbeeld.be/nodig/cloudvision, Kris Merckx)



Kennen, begrijpen

1. Uitleggen waarom computersystemen het moeilijk hebben met het begrijpen van menselijke talen.
2. Waarom hebben "computers" het moeilijk met geschreven taal?
3. Wat is het verschil tussen spelling- en spraakkunstcorrectie enerzijds en NLP anderzijds?
4. Wanneer kan je spreken van "tekstbegrip" bij computersystemen?
5. Waarom is tekstbegrip heel belangrijk bij data-analyse?
6. Waarom is het begrijpen van gesproken tekst nog moeilijker?
7. Waarom hebben computersystemen het moeilijk met het begrijpen van afbeeldingen?
8. Waarom kan je zeggen dat image recognition een goed voorbeeld is van kunstmatige intelligentie?

Hoofdstuk 8

AI EN DATA-ANALYSE

- **Welke algoritmes gebruikt AI?**
- **Voor welke vormen van analyse kan je AI inzetten?**

Je weet al dat AI op verschillende manieren kan leren uit data. Maar welke algoritmes of technieken schuilen er (naast neurale netwerken) nog achter supervised, unsupervised en reinforcement learning?

Zonder te vervallen in uitgebreide wiskundige beschrijvingen, probeert dit hoofdstuk je de weg wat te wijzen.

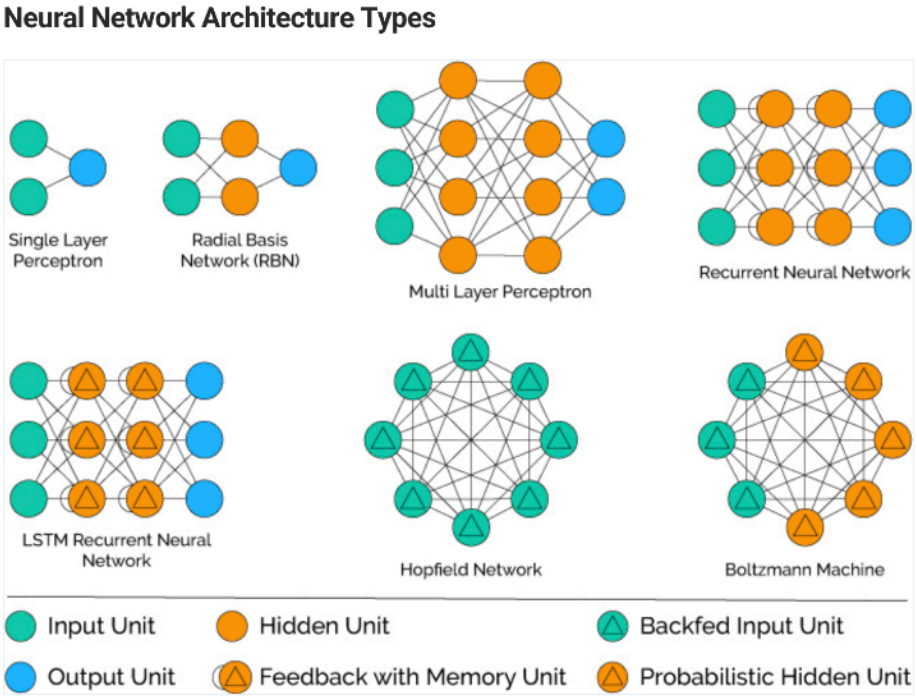
Zo kom je te weten welke ideeën er schuilgaan achter ronkende namen als K-means, linear regressie, decision trees, random forest en support vector machines.

Neurale netwerken

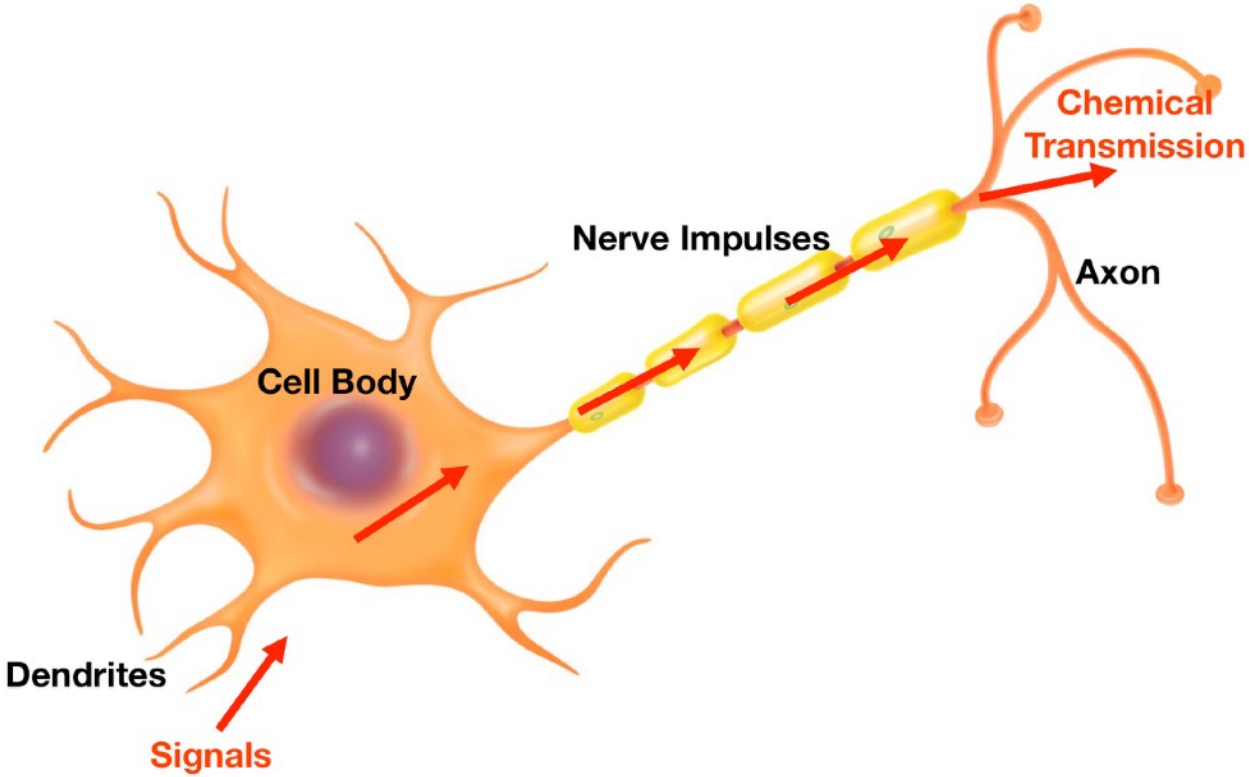
Artificial neural networks (ANN's) kwamen in een eerder hoofdstuk al aan bod. ANN's zijn statistische modellen, geïnspireerd en "gemodelleerd" naar biologische neurale netwerken. Menselijke hersenen bevatten naar schatting ongeveer 100 miljard neuronen. Die neuronen vormen netwerken die informatie uitwisselen en daaruit ook kunnen "leren". Hoe het precies allemaal werkt, is nog niet helemaal duidelijk, maar dingen "leren" zou gebeuren door het versterken van connecties tussen neuronen. Beslissingen (*van iemand een mep in het gezicht geven tot een keuze maken tussen friet en shoarma*) worden genomen op basis van de inputs die neuronen krijgen. De uitvoer van het "netwerk" is de definitieve beslissing.

ANN's zijn in staat om **niet-lineaire verbanden** te leggen tussen invoerdata en uitvoer (de beslissing). Ruwweg betekent dit dat het resultaat niet altijd een keuze voor of tegen is (een **binaire beslissing**), maar een gewogen schatting. De uiteindelijke beslissing is het gevolg is van het optellen van meerdere inputs en het gewicht dat het netwerk aan elke input geeft (zie het voorbeeld van de fiets en de auto). ANN's gebruiken diverse algoritmes en kunnen ook meerdere vormen (architectures) aannemen. Om er een paar op te sommen: *Feed-forward neural networks, Recurrent neural network, Multi-layer perceptrons (MLP), Convolutional neural networks, Recursive neural networks, Deep belief networks, Convolutional deep belief networks, Self-Organizing Maps, Deep Boltzmann machines, Stacked de-noising auto-encoders, Backpropagation, Gradient descent-algoritme, Hebbian learning Rule, Self – Organizing Kohonen Rule, Hopfield Network Law, LMS algorithm (Least Mean Square), Competitive Learning……*

Dieper duiken in ANN's? https://www.xenonstack.com/blog/artificial-neural-network-applications/



Bron: https://www.xenonstack.com/blog/artificial-neural-network-applications/



Een neuron in het menselijk brein. Naar schatting zou ons brein ongeveer 100 miljard van deze neuronen bevatten. De neuronen vormen netwerken en verwerken gezamenlijk informatie. (Bron afbeelding: thinglink.com). *"Een perceptron (of meerlaags perceptron) is een neuraal netwerk waarin de neuronen in verschillende lagen met elkaar verbonden zijn. Een eerste laag bestaat uit ingangsneuronen, waar de inputsignalen aangelegd worden. Vervolgens zijn er één of meerdere 'verborgen' lagen, die zorgen voor meer 'intelligentie' en ten slotte is er de uitgangslaag, die het resultaat van het perceptron weergeeft."* (Bron: https://nl.wikipedia.org/wiki/Perceptron)

HOW A DEEP NEURAL NETWORK SEES

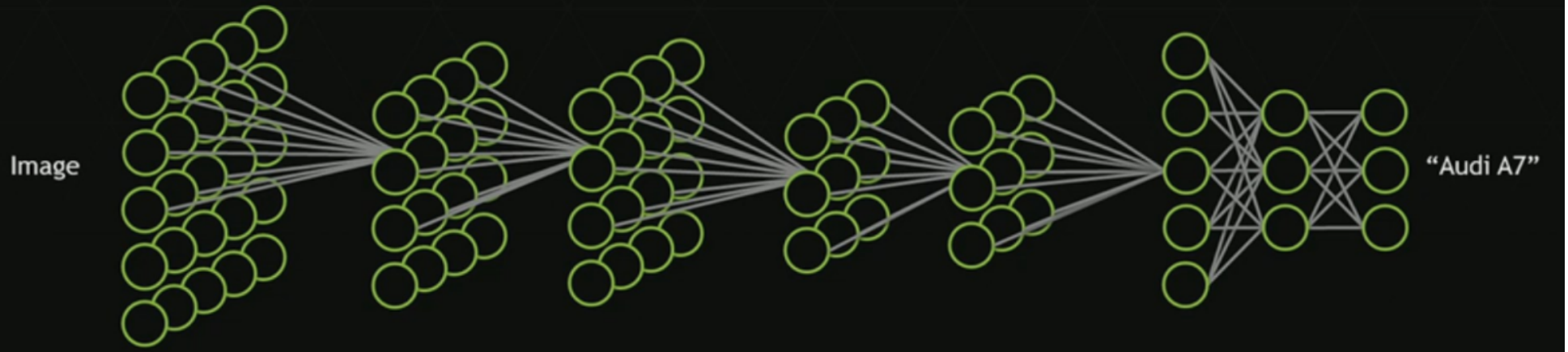
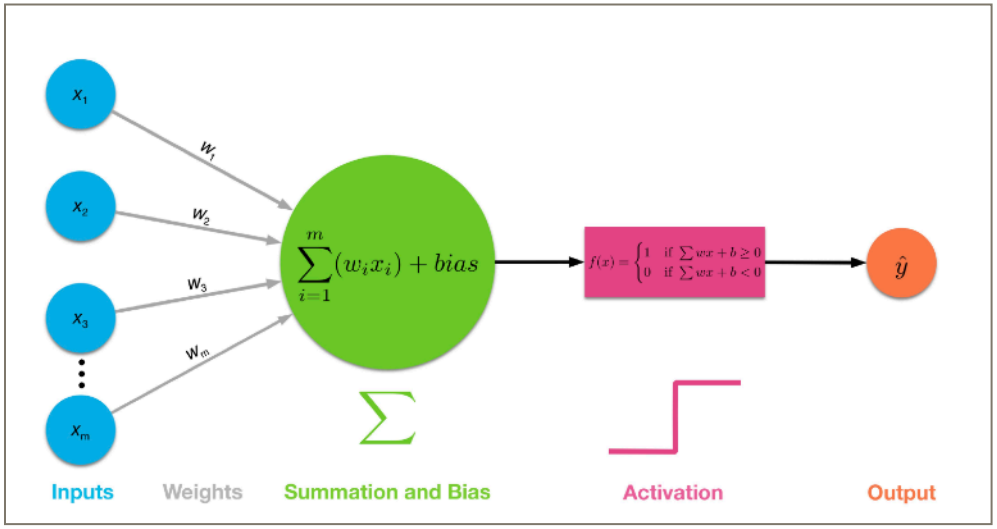


Image source: "Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks" ICML 2009 & Comm. ACM 2011.
Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Ng.



Een wiskundig model van een ANN, een Single-layer Perceptron Network.
Bron:https://towardsdatascience.com/

Het neurale netwerk in de afbeelding hier boven bevat één enkele laag, vandaar de benaming “single-layer perceptron”. Dit geeft de volgende sigmoidfunctie:
$(x_1*w_1)+(w_2*w_2)+\ldots+(x_n*w_n) + b = y$

Over de b=bias lees je in hoofdstuk 9.

Multilayer perceptrons en deep learning

Neurale netwerken kunnen meerdere lagen bevatten. We spreken dan van een "**multilayer perceptron**". Bekijk even de afbeelding op de vorige pagina. Elke laag in dat netwerk probeert patronen te herkennen in de afbeelding. Als een patroon is gevonden, activeert het algoritme de volgende "verborgen" laag. De eerste laag herkent bijvoorbeeld randen in de afbeelding. De volgende laag combineert de gevonden randen enz. Hoe meer lagen zo'n netwerk bevat (hoe dieper het netwerk), hoe beter de patroonherkenning. Elke laag in het ANN kent gewichten toe (of vergroot of verkleint die), elke keer als nieuwe data (in dit geval nieuwe foto's) het netwerk binnenkomen, en dat bepaalt mee te invoer van de volgende laag.

Architectuur	Voorbeeld	Leermethode
Convolutional Neural Networks	Herkennen afbeeldingen	Supervised
Recurrent neural networks en long-short term memory	Natural language processing	Supervised
Autoencoders	afwijkende data opsporen (bijvoorbeeld fraude)	Unsupervised
Generative Adversarial Networks	Computer vision	Unsupervised

Wanneer meerdere "multilayer perceptrons” worden gecombineerd, spreekt men van **deep learning**. Deep learning kent meerdere vormen of architecturen.

AI als oplossing voor analyse-problemen

AI-algoritmes doen in essentie allemaal hetzelfde. Ze voorspellen een uitvoer op basis van onbekende invoer. Het soort data bepaalt echter het meest geschikte algoritme. Je kan AI-algoritmes indelen op basis van het doel waarvoor je ze wil inzetten.

	Doel	Leermethode	Voorbeeld
1	Classificatie	Supervised	Spam of geen spam
2	Regressie	Supervised	Marktwaarde voorspellen
3	Clustering	Unsupervised	Frauduleuze transactie ontdekken

1 Classificatie-algoritmes

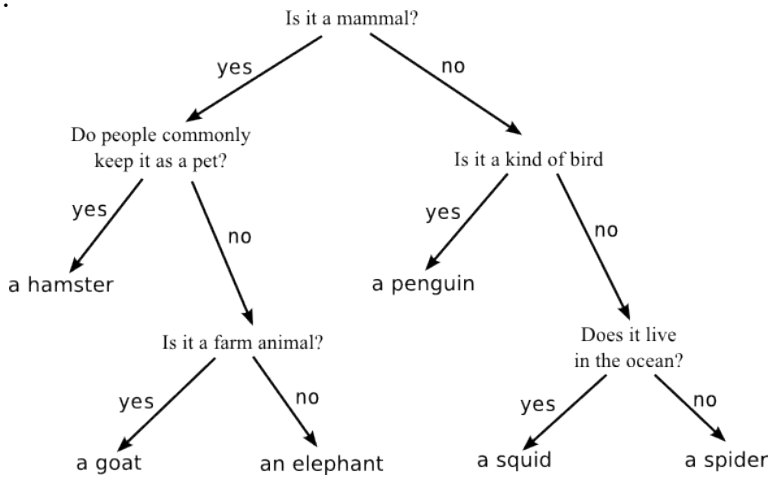
- Naive Bayes
- Decision Tree
- Random Forest
- Logistic Regression
- Support Vector Machines
- K Nearest Neighbours

Naive Bayes classificeert data door de statische waarschijnlijkheid te schatten dat data bij een bepaalde klasse of groep horen. Het kijkt niet naar verbanden, maar naar het aantal overeenkomsten en berekent op basis daarvan de waarschijnlijkheid. In de afbeeldingen zie je een voorbeeld van sentiment analysis (*links de dataset, rechts een voorspelling*)

TEXT	CATEGORY
I am so angry.	mood
I feel like superstar.	mood
It is going to rain.	not mood
I want to cry.	mood
They will come together.	not mood

I	mood
feel	mood
so	
terrible	

Een **decision tree** (beslissingsboom) kent een stamboomachtige structuur. Bij elke "tak" test je een bepaalde voorwaarde of attribuut dat de uitvoer naar de volgende tak bepaalt. Opgelet: het kan leiden tot "overfit" en de minste aanpassing aan de invoer (in dit geval trainingsdata), kan leiden tot drastische verschillen in de uitvoer.



Een **random forest** is een verzameling van beslissingsbomen, waarbij elke beslissingsboom een subset van de attributen van gegevens heeft gekregen en voorspelt op basis van die subset. Het gemiddelde van de stemmen van alle beslissingsbomen leidt tot het antwoord. Dit algoritme vermindert het probleem van overfitting en leidt tot een veel nauwkeuriger classificatie.

Simpel gezegd voorspelt **logistische regressie** in welke categorie data valt op basis van andere informatie. Bijvoorbeeld: *op basis van de eerdere resultaten van een student, kan je zijn vermoedelijke eindresultaat voorspellen.*

Een **support vector machine** classificeert binair, het verdeelt de ingevoerde data in twee klassen aan de hand van kenmerken die je vooraf geselecteerd hebt. Vooraleer de SVM hiertoe in staat is, moet je het model trainen met historische data. Men spreekt van *vector* omdat elk datapunt een x- en y-positie krijgt in een grafiek (of vectorruimte). SVM's werken goed bij kleine datasets waar weinig ruis in staat en zo goed als geen overlappende categorieën. SVM's zijn niet geschikt voor grotere datasets.

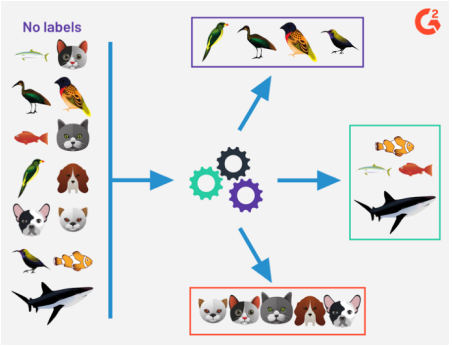
K-Nearest Neighbours (KNN) gaat uit van het idee dat de klasse waartoe een element behoort overeenkomt met de klassen van de meest nabijgelegen punten. Stel dat iedereen in je omgeving kiest voor Groen, dan is de kans groot dat jij dat ook doet. Bij K=5 kijk je naar 5 dichtstbij gelegen "buren" (weliswaar in een vectorruimte). Bij K=3 naar de 3 meest nabije buren enz.

2 Regressie-algoritmes

Lineaire regressie is het meest eenvoudige en effectieve regressie-algoritme. Het wordt gebruikt om twee variabelen (*bijvoorbeeld de kostprijs van een huis en de ligging van een huis*) tegen elkaar af te wegen. Dit kan je uittekenen op een grafiek en komen tot een regressielijn. Op basis van die lineaire regressie kan je prijzen voorspellen op basis van de ligging van een woning.

3 Clustering-algoritmes

Het reeds eerder vermelde **K-means**-algoritme verdeelt data in een door de gebruiker bepaalt aantal klassen (k=1, k=4…). Jij bepaalt m.a.w. in hoeveel groepen je de ingevoerde data wil indelen. Het algoritme is handig te gebruiken bij datasets waar duidelijk te onderscheiden elementen (punten) aanwezig zijn.



Kennen, begrijpen

1. Het verschil tussen "niet-lineaire verbanden" en "binaire beslissingen" uitleggen aan de hand van een voorbeeld.
2. De volgende begrippen begrijpelijk uitleggen: ANN's, multilayer perceptrons, deep learning...
3. Voor welke drie doelen kunnen AI-algoritmes worden ingezet?
4. Uitleggen welke leermethode AI-algoritmes hanteren naargelang het doel.
5. Zes belangrijke classificatie-algoritmes opsommen en uitleggen.
6. Uitleggen waarom clustering-algoritmes in de meeste gevallen gebruik maken van unsupervised learning.
7. Een praktische toepassing van regressie-algoritmes geven.

Kunnen

8. Bij een gegeven dataset en probleemstelling het meest geschikte algoritme selecteren. Argumenteren waarom je die keuze maakt.
9. Bij een gegeven dataset en probleemstelling de meest geschikte "AI-leermethode" selecteren. Argumenteren waarom je die keuze maakt.

Hoofdstuk 9

BEVOOROORDEELDE KUNSTMATIGE INTELLIGENTIE

- **Is predictie altijd objectief?**
- **Werkt een AI-algoritme altijd vlekkeloos en betrouwbaar omdat het gebaseerd is op wiskunde?**

Microsoft onthulde Tay, een Twitter-bot, een experiment in 'conversational understanding'. *"Hoe meer je met Tay chat"*, zei Microsoft, *"hoe slimmer het wordt, mensen leren betrekken door middel van een informeel en speels gesprek"*.

Helaas bleven de gesprekken niet lang speels. Vrij snel nadat Tay gelanceerd was, begonnen mensen de bot te tweeten met allerlei misogyne, racistische en Donald Trumpistische opmerkingen. In minder dan 24 uur veranderde Tay, gevoed met extremistische Twitterpraat, in een racistische bullebak.

Waar liep het mis? Het mag duidelijk zijn dat AI-modellen niet altijd het verhoopte resultaat opleveren.

Het black box-probleem en XAI

Op basis van de trainingsset en menselijke feedback heeft het algoritme bepaalde gewichten verhoogd en andere weer verminderd. Als het ontwikkelteam merkt dat de controledata het gewenste resultaat opleveren, kan het model worden ingezet in de praktijk. Hoe en waarom het AI-model tot een conclusie komt, is op dat moment niet meer de vraag. De predicties komen overeen met de gewenste resultaten. Maar net daar schuilt het probleem. Je kan niet met AI-modellen discussiëren en argumenteren waarom het tot een bepaalde conclusie is gekomen. De reden waarom lijkt een gesloten boek, een zogenaamde "**black box**". *IBM trachtte zijn Watson-supercomputer te promoten als een belangrijk hulpmiddel bij kankeropsporing, maar de (menselijke) oncologen vertrouwden het systeem simpelweg niet. Twee oncologen kunnen discussiëren, maar dat lukt niet met een AI-model.*

Meer en meer groeit de eis dat AI-algoritmes ook verklaarbaar moeten zijn. Het moet duidelijk zijn hoe en waarom een model tot een bepaald besluit komt. Ook AI-critici en mensen die geloven in een dystopische toekomst waarbij AI de wereld overneemt, pleiten voor **Explainable AI (XAI)**. Enkel door AI-algoritmes en modellen verklaarbaar te houden, kan de ontwikkeling van AI onder controle worden gehouden.

In de trilogie The Matrix heeft AI de wereld overgenomen. De mensen denken nog bewust te leven, maar zijn niet meer dan virtuele avatars in een "ingebeelde" game.

Subjectieve resultaten

Stel dat een kredietkaartbedrijf wil weten of iemand kredietwaardig is. Een ontwikkelaar kan een AI-algoritme inzetten om te bepalen of dit het geval is voor elke gegeven gebruiker. Maar kredietwaardigheid is een subjectief en vaag gegeven. Iemand is minder kredietwaardig als het risico op betaalachterstanden groter is (bijvoorbeeld bij een laag loon of hoog risico op werkloosheid). De geldverstrekker in kwestie kan er echter meer aan verdienen dan aan traditionele leningen.

Het bedrijf zal in eerste instantie zo veel mogelijk profijt willen behalen door zoveel mogelijk leningen toe te kennen die terugbetaald moeten worden aan een maximale rente. "Kredietwaardigheid" wordt op die manier geherformuleerd op basis van een zo hoog mogelijke winstmarge. Als het algoritme ontdekt dat het verlenen van leningen aan minder kredietwaardige mensen een effectieve manier is om de winst te maximaliseren, zal het uiteindelijk roofzuchtig gedrag vertonen, zelfs als dat niet de bedoeling van het bedrijf was.

Vooroordelen in de dataset

Vooroordelen, **bias** in het Engels, kunnen optreden wanneer de data die je verzamelt niet helemaal representatief is voor de werkelijke situatie, of bestaande vooroordelen weerspiegelt. Wat als je een stuk AI-software ontwikkelt voor het herkennen van gezichten, maar enkel gezichten van blanke mensen aan het systeem voedt als trainingsdata? Amazon ontwikkelde een aanwervingstool die heel duidelijk een voorkeur had voor mannelijke kandidaten, omdat het leerde uit historisch data van het bedrijf.

Trainingsdata voorbereiden

Welke attributen of eigenschappen je gaat gebruiken in de trainingsdata, bepalen mee het eindresultaat. Neem de Amazon-aanwervingstool als voorbeeld: je kan ook het aantal jaren ervaring of de opleiding als attribuut als invoer gebruiken. Bij kredietwaardigheid speelt de leeftijd en het inkomen een bepalende rol, niet zo zeer het te bepalen profijt.

AI-bias: moeilijk op te lossen

1 Onbekende onbekenden

Vaak stellen ontwikkelaars pas achteraf vast wat de effecten zijn van de gekozen attributen. In het geval van Amazon herprogrammeerden de ontwikkelaars de tool door alle *geslachtsgerelateerde* woorden uit hun data te weren. Maar ook dan pikte het aangepaste systeem werkwoorden op die eerder met mannen correleren dan met vrouwen.

2 Training en controle

Vaak splitsen ontwikkelaars vooraf hun data op in een trainingsset en een controleset. Dit houdt in dat de controledata dezelfde vooroordelen bevat dan de trainingsdata.

3 Gebrek aan sociale context

Computerwetenschappers of ontwikkelaars willen vooral systemen ontwikkelen die voor meerdere situaties inzetbaar zijn. Net doordat ze de sociale context buiten beschouwing laten, krijg je vooroordelen. Je kan geen aanwervingstool ontwikkelen voor Amazon en die eveneens gebruiken voor het aanwerven en selecteren van bouwvakkers.

4 Hoe meet je eerlijkheid?

Wanneer is er sprake van eerlijkheid en vooroordelen? Is Zwarte Piet racistisch? Vooroordelen komen altijd en overal voor, maar iedereen definieert ze anders. Ook in AI-data komen ze voor. Het verschil is echter dat ze voor AI mathematisch worden gedefinieerd. Hoe kan je zulke vooroordelen voorkomen? Betekent dit dat je in alle situaties groepen gelijk moet laten voorkomen? Moeten in alle situaties vrouwen en mannen, mensen met een blanke of gekleurde huid enz. even sterk vertegenwoordigd zijn? Moeten vrouwen even hoog of vaak scoren in aanwervingsprocedures, ook al voldoen de groepen niet aan andere eigenschappen die noodzakelijk zijn?

AI-onderzoekers doen hun best om het probleem van AI-bias aan te pakken. Ze trachten algoritmen te ontwikkelen die helpen bij het detecteren en verminderen van verborgen vooroordelen binnen trainingsgegevens of die de vooroordelen van het model verminderen, ongeacht de gegevenskwaliteit. Eenvoudig kan het niet worden opgelost. Het is een continu proces en we moeten er bij het ontwikkelen van AI-tools steeds bewust van zijn dat de resultaten of voorspellingen niet enkel het gevolg zijn van een mathematisch algoritme. Net zoals in alle andere aspecten van de samenleving is predictieve analyse nooit helemaal objectief.

Kris Merckx - september-november 2019

Kennen, begrijpen

1. Uitleggen wat het "black box"-probleem is bij AI.
2. Waarom groeit de eis voor "explainable AI"?
3. Geef een voorbeeld van een situatie waarbij de AI-predictie subjectieve resultaten op kan leveren.
4. Op welke manieren kan er "bias" in de dataset sluipen?
5. Geef 4 redenen (en leg uit) waarom bias in AI moeilijk op te lossen is.

Kunnen

6. Bij een gegeven dataset kunnen inschatten of er bias aanwezig kan zijn.

www.schoolvoorbeeld.be

Test zelf een image recognition tool uit;
www.schoolvoorbeeld.be/nodig/cloudvision

Kris Merckx 2019

Kris Merckx 2019

Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International

We horen er veel en vaak over: kunstmatige intelligentie. Wat is dat eigenlijk? Hoe werkt het? Hoe komt het dat we computersystemen plots “intelligent” of smart kunnen noemen? Moeten zij ook eerst leerstof doornemen en studeren, net zoals wij mensen dat doen?

Klopt het dat we bang moeten zijn voor kunstmatige intelligentie? Worden we als mens overbodig? Zullen machines onze plaats innemen?

Kent kunstmatige intelligentie beperkingen?

Zijn er dingen die kunstmatige intelligentie nooit zal kunnen?

Dit boek probeert een begrijpelijk antwoord te leveren op heel veel vragen over kunstmatige intelligentie en data-analyse.

Kris Merckx