

Advanced Programming & Maths

Tuur Vanhoutte

15 februari 2021

Inhoudsopgave

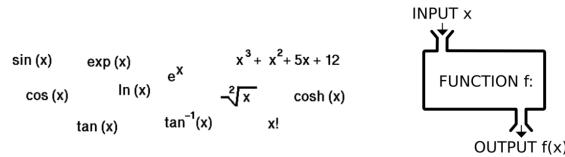
1 Basisfuncties in de wiskunde	1
1.1 Functies	1
1.2 Veelterm en veeltermfuncties	2
1.3 Bijzondere veeltermfuncties	2
1.3.1 Constante functie	3
1.3.2 Lineaire functie	3
1.3.3 Tweedegraadsfunctie	3
1.3.4 Derdegraadsfunctie	5
1.3.5 Exponentiële functie	5
2 Exponentiële verbanden in data	6
2.1 Lineaire groei	6
2.2 Exponentiële groei	7
2.3 Van groeipercentage naar groeifactor	7
2.3.1 Percentage naar factor	7
2.3.2 Factor naar percentage	8
2.4 Voorbeeld	8
2.5 Belangrijke maten voor exponentiële toename	9
2.5.1 Oefening: Combinatie van groeifactoren?	9
3 Belangrijke functies met betrekking tot machine learning	10
3.1 Logistische groei	10
3.1.1 Voorbeeld	10
3.1.2 De groei	10
3.1.3 Functievoorschrift	10
3.1.4 Voorbeeld	11
3.1.5 Algemene wiskundige notatie van een logistische functie	11
3.2 Regression analysis	12
3.2.1 Lineair regressiemodel	12
3.2.2 Logistisch regressiemodel	13
3.2.3 Lineair vs logistisch regressiemodel	14
3.2.4 Meerdere inputfactoren	14
3.3 Softmax functie	14
3.3.1 Kansen	15
3.3.2 Model	15
3.3.3 Wiskundig	16
3.4 Logistic regression cost function	16
3.4.1 Success meten	16
4 Pandas library	17
4.1 Inleiding	17
4.1.1 Welke data verwerken?	18
4.2 Pandas.core	18
4.3 Series	18
4.4 DataFrame	18
4.4.1 Select data from DataFrame	19
4.4.2 Veelgebruikte commandos bij dataframes	20
4.5 Loc vs iloc	21
4.5.1 iloc	21
4.5.2 loc	21

4.6	Plotten met pandas	22
4.6.1	Dataframe plotten	22
4.6.2	Series plotten	23
4.7	Demo: Iris Dataset	23
4.8	Complexe bewerkingen	24

1 Basisfuncties in de wiskunde

1.1 Functies

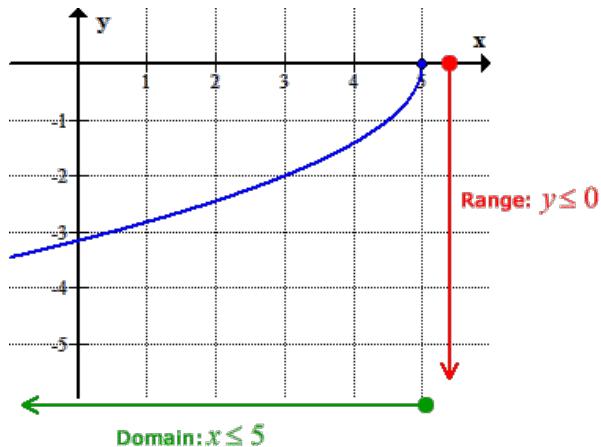
Definitie 1.1 (Reële functie) Een reële functie is een relatie in \mathbb{R} waarbij elke waarde x hoogstens één beeldwaarde $f(x)$ heeft



Figuur 1: Voorbeelden reële functies

Definitie 1.2 Voor elke functie geldt: er bestaat een ...

- (i) ... domein van de functie (domain)
- (ii) ... beeld van de functie (range)
- (iii) ... functievoorschrift van de functie

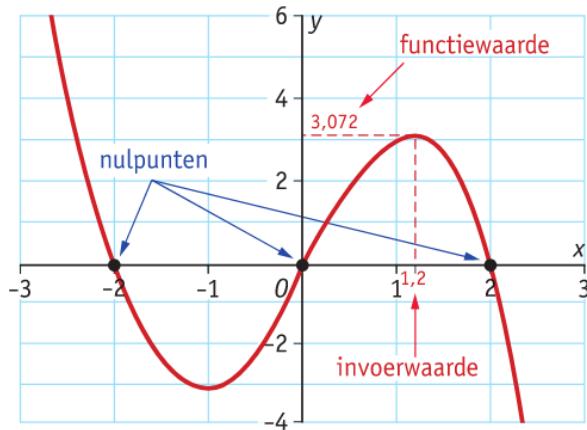


Figuur 2: Domein, bereik, functievoorschrift

$$f : \text{domein} \rightarrow \text{bereik} : x \rightarrow y = f(x)$$

$$f : \mathbb{R} \rightarrow \mathbb{R} : x \rightarrow y = x^3 - 4x$$

Definitie 1.3 Elke functie kan nulpunten hebben.



Figuur 3: $y = -x^3 + 4x$

Verloop van een functie wordt via een tekenschema verduidelijkt:

x		-2		0		2	
$f(x)$	+	0	-	0	+	0	-

Figuur 4: Tekenschema

1.2 Veelterm en veeltermfuncties

Definitie 1.4 (Veelterm)

$$A(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \dots + a_2 x^2 + a_1 x + a_0 \quad (a_n, a_{n-1}, \dots, a_2, a_1, a_0 \in \mathbb{R}) \quad (1)$$

Definitie 1.5 (Veeltermfunctie)

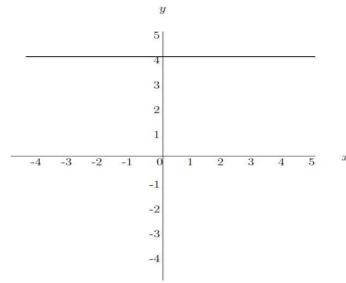
$$f(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \dots + a_2 x^2 + a_1 x + a_0 \quad (2)$$

Graad van veelterm = n als $a_n \neq 0$

1.3 Bijzondere veeltermfuncties

- Constante functie: $f(x) = 4$
- Lineaire functie: $f(x) = 4$
- Tweedegraadsfunctie: $f(x) = 3x^2 + 2x + 1$
- Derdegraadsfunctie: $f(x) = 5x^3 - 3x^2 + 2x - 1$
- Exponentiële functie: $f(x) = 2^x$
- Logaritmische functie: $(fx) = \log_2(x)$

1.3.1 Constante functie



Figuur 5: $y = 4$

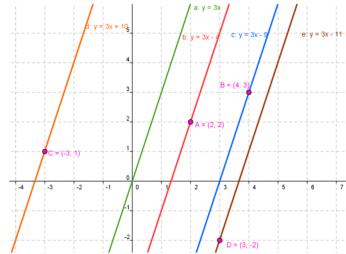
1.3.2 Lineaire functie

Definitie 1.6 (Lineaire functie)

$$f(x) = ax + b \quad (3)$$

Voorbeeld: $f(x) = 3x + 6$

- Betekenis van a : de richtingscoëfficiënt (rico)
- Betekenis van b : het snijpunt met de y -as
- Nulpunt: $f(x) = 0$
 $\Leftrightarrow 3x + 6 = 0$
 $\Leftrightarrow 3x = -6$
 $\Leftrightarrow x = -2$



Figuur 6: Meerdere evenwijdige lineaire functies

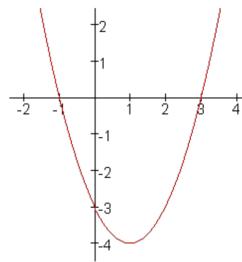
Evenwijdige rechten als: als $a_1 = a_2$

Loodrechte rechten als: als $a_1 \cdot a_2 = -1$

1.3.3 Tweedegraadsfunctie

Definitie 1.7

$$f(x) = ax^2 + bx + c, \quad (a \neq 0) \quad (4)$$



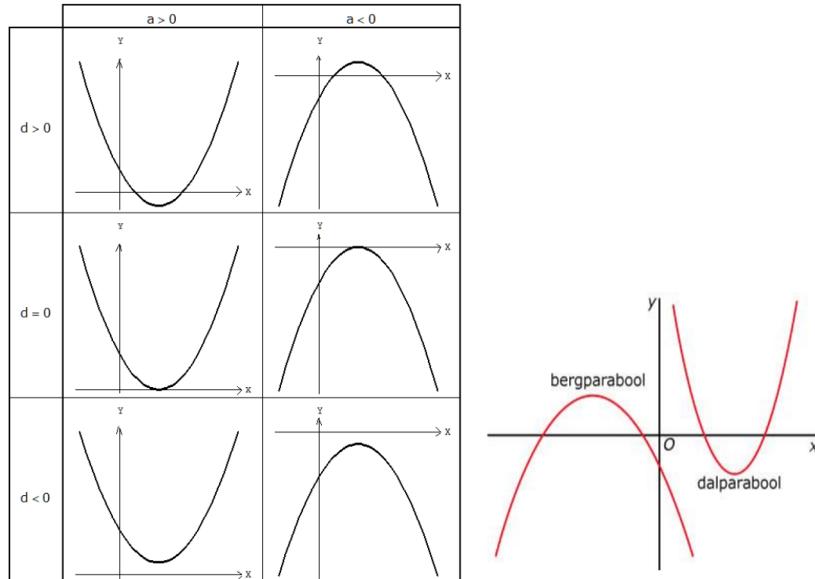
Figuur 7: $f(x) = x^2 - 2x - 3$

- Betekenis van a : positief \Rightarrow dalparabool, negatief \Rightarrow bergparabool
- Nulpunten: via de discriminant berekenen:

Definitie 1.8 (Discriminant) Bij een tweedegraadsvergelijking is de discriminant:

$$D = b^2 - 4ac \quad (5)$$

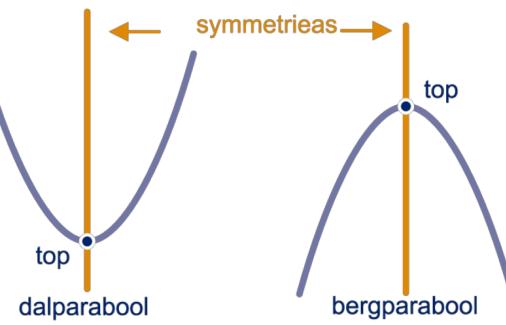
- Geval 1: $D > 0 \Rightarrow$ de functie heeft 2 nulpunten
- Geval 2: $D = 0 \Rightarrow$ de functie heeft 1 nulpunt
- Geval 3: $D < 0 \Rightarrow$ de functie heeft géén nulpunten



Figuur 8: De discriminant toont de nulpunten

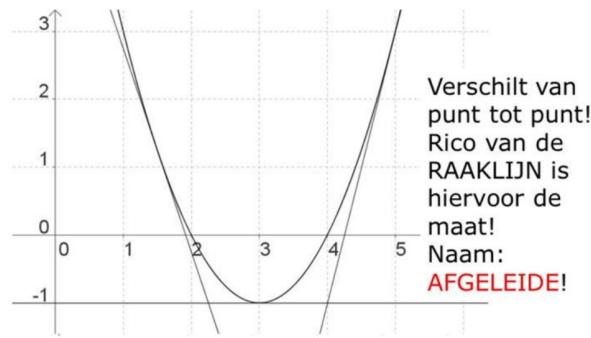
Nulpunten berekenen:

$$x_{1,2} = \frac{-b \pm \sqrt{D}}{2a} \quad (6)$$



Figuur 9: Symmetrieas: $x = \frac{-b}{2a}$

Voorbeeld:



Figuur 10: $y = x^2 - 6x + 8$

1.3.4 Derdegraadsfunctie

Definitie 1.9 (Derdegraadsfunctie)

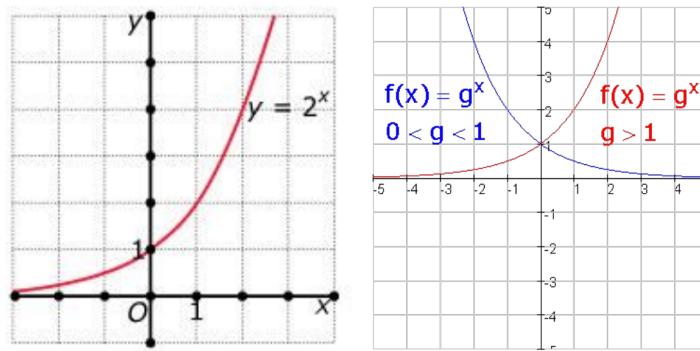
$$f(x) = ax^3 + bx^2 + cx + d (a \neq 0) \quad (7)$$

1.3.5 Exponentiële functie

Definitie 1.10 (Exponentiële functie)

$$f(x) = a^{g(x)} \quad (8)$$

Met grondtal $a \in \mathbb{R}_0^+ \setminus \{1\}$



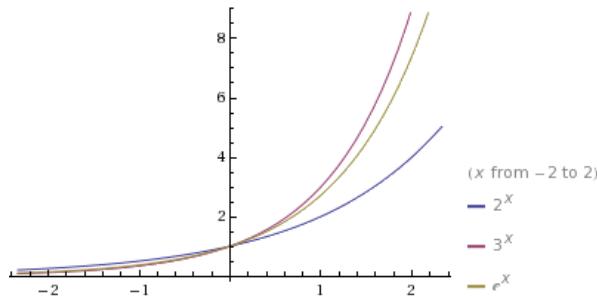
Figuur 11

- Betekenis van a: groefactor
- Wanneer stijgend?
- Wanneer dalend?
- Nulpunten:
- Vaststelling beeld functie

Definitie 1.11 (Constante van Euler)

$$e \approx 2.718281828 \dots \quad (9)$$

$f(x) = e^x$ is een bijzondere exponentiële functie



Figuur 12: Verschil tussen 2^x , 3^x en e^x

2 Exponentiële verbanden in data

2.1 Lineaire groei

Kenmerkend:

- Per tijdseenheid wordt hetzelfde getal **opgeteld**
- Grafiek is een rechte
- Algemene formule (N = aantal, t = tijd, b : beginhoeveelheid):

$$N = a \cdot t + b \quad (10)$$

t	0	1	2	3	4	5
N	750	780	810	840	870	900
	+30	+30	+30	+30	+30	+30

Figuur 13: Lineaire groei

2.2 Exponentiële groei

Kenmerkend:

- Per tijdseenheid wordt de hoeveelheid met hetzelfde getal **vermenigvuldigd**
- Grafiek is een exponentiële functie
- **Algemene formule:**

$$N = b \cdot g^t \quad (11)$$

t	0	1	2	3	4
N	1280	1600	2000	2500	3125
	x1,25	x1,25	x1,25	x1,25	x1,25

Figuur 14: Exponentiële groei

LENGTE FIETSPADEN IN NEDERLAND					
jaar	1998	2002	2006	2010	2014
aantal km	17 600	21 500	26 200	32 000	39 000

Figuur 15: Voorbeeld exponentiële groei met groeifactor ≈ 1.22

2.3 Van groeipercentage naar groeifactor

De toename/afname wordt vaak ook procentueel uitgedrukt

- Een jaarlijkse toename van 14.6%
- Een jaarlijkse afname van 14.6%

Definitie 2.1 (Groeifactor) *De groeifactor is de factor die per tijdseenheid wordt vermenigvuldigd met de vorige waarde.*

2.3.1 Percentage naar factor

$$g = \frac{p + 100}{100}\% \quad (12)$$

$$\begin{aligned} 100\% + 14,6\% &= 114,6\% = 1,146 \\ \times 1,146 \end{aligned} \qquad \begin{aligned} 100\% - 14,6\% &= 85,4\% = 0,854 \\ \times 0,854 \end{aligned}$$

Figuur 16: Van groeipercentage naar groeifactor

2.3.2 Factor naar percentage

$$0,765 = 76,5\% - 100\% = -23,5\%$$

Figuur 17: Van groeifactor naar groeipercentage



Voorbeeld

De groeifactor per uur is gelijk aan 3.

De groeifactor per 2 uur is gelijk aan $3^2 = 9$

De groeifactor per 20 minuten is gelijk aan $3^{\frac{1}{3}} = 1,44$.

Figuur 18: Let op: hier gebeuren vaak fouten bij het omrekenen

2.4 Voorbeeld

Een hoeveelheid groeit exponentieel. Na 5u is $N = 82$ en na 12u is $N = 246$.

Stel de formule van N op.

Oplossing

$$N = b \cdot g^t \quad (13)$$

Stap 1: groeifactor berekenen per tijdseenheid:

$$\left. \begin{array}{l} \text{Na 5u} \rightarrow N = 82 \\ \text{Na 12u} \rightarrow N = 246 \end{array} \right\} \Delta = 7u \rightarrow 164$$

Groeifactor voor 7 uren: $\frac{246}{82} = 3$

Groeifactor voor 1 uur: $3^{1/7} \approx 1.170$

Stap 2: 1 punt nemen waarvan we N weten:

Gekozen punt: (5, 82)

$$82 = b \cdot (1.170)^5$$

$$\Leftrightarrow b = \frac{82}{1.170}^5 \approx 37$$

$$\Leftrightarrow N = 37 \cdot 1.170^t$$

2.5 Belangrijke maten voor exponentiële toename

Definitie 2.2 (Verdubbelingstijd) *De verdubbelingstijd is de nodige tijd tot de hoeveelheid verdubbeld is.*

De verdubbelingstijd t kan je berekenen met:

$$g^t = 2 \quad (14)$$

Oefening

De populatie neemt toe met 8.3% per jaar. Bereken de verdubbelingstijd:

$$\begin{aligned} g^t &= 2 \\ \Leftrightarrow (1.083)^t &= 2 \\ \Leftrightarrow \log(1.083^t) &= \log(2) \\ \Leftrightarrow t \cdot \log(1.083) &= \log(2) \\ \Leftrightarrow t &= \frac{\log(2)}{\log(1.083)} \\ \Leftrightarrow t &= 8.69 \text{ jaar} \end{aligned}$$

Definitie 2.3 (Halveringstijd) *De halveringstijd is de nodige tijd tot de hoeveelheid gehalveerd is.*

De halveringstijd t kan je berekenen met:

$$g^t = 1/2 \quad (15)$$

2.5.1 Oefening: Combinatie van groeifactoren?

Een hoeveelheid neemt eerst 5 jaar lang met vast percentage (*) toe, om daarna nog 3 jaar met 10% per jaar toe te nemen. Na 8 jaar is de totale hoeveelheid verdubbeld.

(*) Bereken het jaarlijkse groeipercentage in de eerste 5 jaren.

Oplossing

We weten:

- Eerste 5 jaar: toename met vast percentage
- Volgende 3 jaar: toename met 10% (= factor van 1.1)
- Na 8 jaar: hoeveelheid verdubbeld (= factor van 2)

$$g^5 \cdot 1.1^3 = 2$$

We moeten g vinden:

$$\begin{aligned} \Leftrightarrow g^5 &= \frac{2}{1.1^3} \\ \Leftrightarrow g &= \sqrt[5]{\frac{2}{1.1^3}} \end{aligned}$$

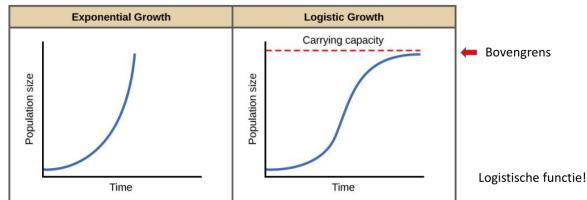
3 Belangrijke functies met betrekking tot machine learning

3.1 Logistische groei

3.1.1 Voorbeeld

Startsituatie: een bos (bv 10km²) waarin een konijnenepidemie uitbreekt. Boswachter houdt de populatie van de konijnen bij. Wat stelt hij vast?

De groei van de populatie verloopt volgens een typisch patroon (niet exponentieel):



Figuur 19: De rode lijn is de bovengrens

3.1.2 De groei

= de mate van toename

- Hangt af van hoeveel er al zijn tegenover hoeveel er nog bij kunnen
- Heel sterke verandering bij start, op het einde heel kleine verandering
- Hangt dus ook af van de tijd

Definitie 3.1 (De logistische groei) *De logistische groei is de mate van toename, afhankelijk van hoeveel er nog bij kan en hoeveel er al is*

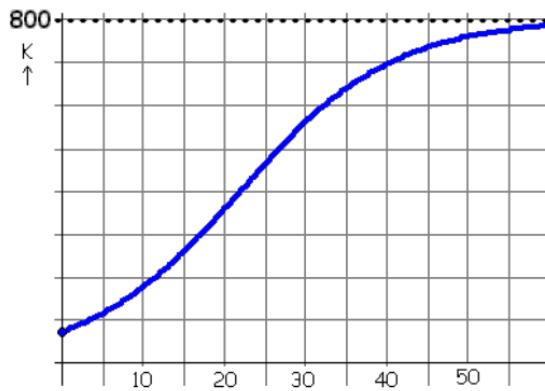
$$\frac{\text{Hoeveel er nog bij kan}}{\text{Hoeveel er al is}} = B \cdot g^t \quad (16)$$

- t = de tijd,
- B en g = constanten

3.1.3 Functievoorschrift

$$y = \frac{G}{1 + B \cdot g^t} \quad (17)$$

- t = de tijd
- B en constanten
- G = bovengrens



Figuur 20: Grafiek logistische groei met $G = 800$

3.1.4 Voorbeeld

Het aantal vissen in een meer is gegeven door:

$$N = \frac{2500}{1 + 5.5 \cdot 0.74^t}$$

waarbij N = aantal vissen, t = tijd

Beredeneer: Wanneer bereiken we het 'verzadigingsniveau'

Als t heel groot is:

- Dan wordt $0.74^t \approx 0$
- Dan wordt $5.5 \cdot 0.74^t \approx 0$
- Dan wordt $N \approx 2500$
- \Rightarrow Het meer is 'verzadigd'

3.1.5 Algemene wiskundige notatie van een logistische functie

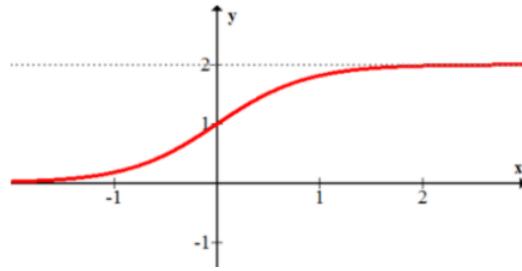
Definitie 3.2 (Logistische functie) De wiskundige notatie voor een logistische functie is:

$$f(x) = \frac{c}{1 + a \cdot b^x} \tag{18}$$

met a, b, c constanten waarbij de constante c de belangrijkste is:

c drukt uit wat de maximumwaarde kan zijn

$$f(x) = \frac{2}{1 + 0.1^x}$$



Figuur 21

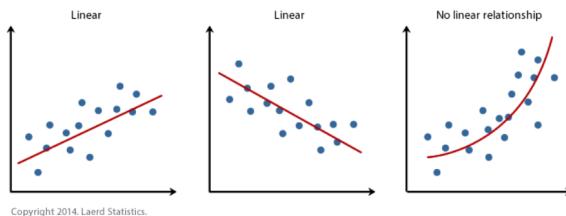
3.2 Regression analysis

Regressieanalyse:

- Is er een (voorspellend) verband tussen 2 variabelen
- Heeft de ene variabele een invloed op de andere variabele



Figuur 22: Regressieanalyse

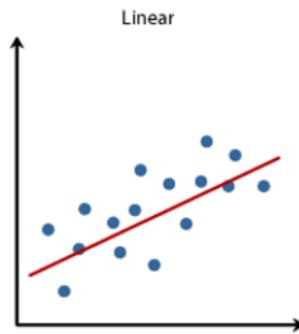


Figuur 23: Lineaire vs niet-lineaire samenhang

3.2.1 Lineair regressiemodel

Enkelvoudige vorm:

- 1 inputwaarde x
- via lineaire functie $h_\theta(x) = \theta_0 + \theta_1 x$



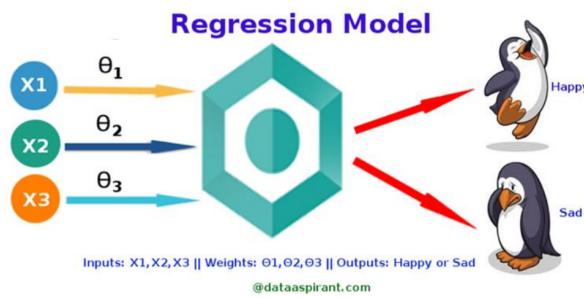
Figuur 24

- Aan de hand van de opgestelde functie doe je een voorspelling
- **Doel:** een zo goed mogelijke lineaire functie opstellen
- ⇒ zoektocht naar de beste θ_0 en θ_1

3.2.2 Logistisch regressiemodel

Logistische regressie = **Classificatie-algoritme**

Zoeken naar een model dat uitkomst (2 mogelijkheden) voorspelt mbv inputwaarden. Elke inputwaarde heeft een zeker belang (gewicht).



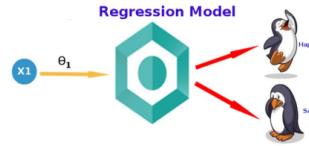
Figuur 25: 3 inputs met elk een bepaald gewicht, die een uitkomst zoekt (2 mogelijkheden)

Vereenvoudiging:

- 1 inputwaarde x
- Logistische functie $p = \frac{1}{1+e^{-(b_0+b_1x)}}$

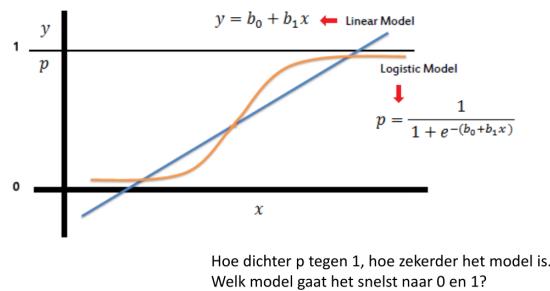
Uitkomst:

- de persoon slaagt als $h_\theta(x) \geq 0.5$
- de persoon slaagt niet als $h_\theta(x) < 0.5$



Figuur 26: 1 inputwaarde x , met twee uitkomsten

3.2.3 Lineair vs logistisch regressiemodel



Figuur 27: Hoe dichter p tegen 1, hoe zekerder het model is

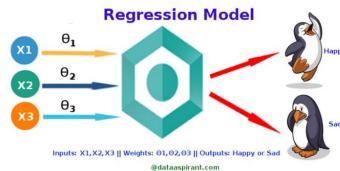
Welk model gaat het snelst naar 0 en 1?

- Het logistische model
- Daarom is het logistische model beter voor classificatie: je splitst de groep op in 2

3.2.4 Meerdere inputfactoren

Zelfde redenering:

- Meerdere inputwaarden x_1, x_2, \dots
- Gebruik $\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots$



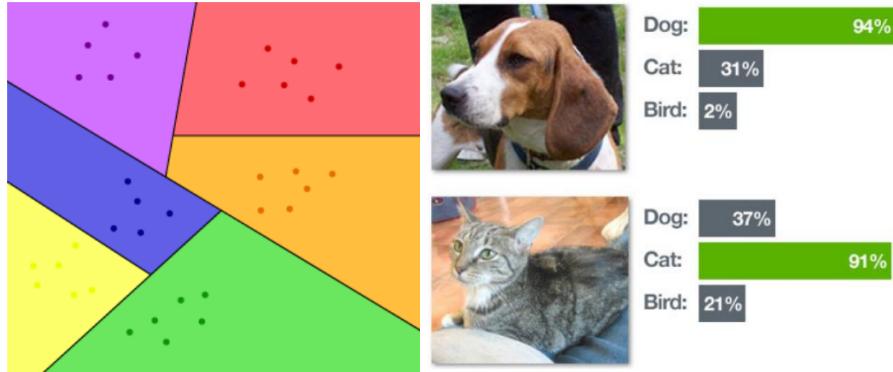
Figuur 28: Regressiemodel met meerdere inputfactoren

3.3 Softmax functie

Doelstelling:

- Model dat in staat is om data te gaan categoriseren
- Hoe?

- ⇒ Met behulp van verschillende inputvariabelen en bijhorende parameters



Figuur 29: Categoriseren met de softmax functie

3.3.1 Kansen

Kans dat de toestand tot groep A behoort:

- $\theta_{A,0} + \theta_{A,1}x_1 + \theta_{A,2}x_2$
- Voorbeeld: $0.01 + 0.1x_1 + 0.1x_2$

Kans dat de toestand tot groep B behoort:

- $\theta_{B,0} + \theta_{B,1}x_1 + \theta_{B,2}x_2$
- Voorbeeld: $0.1 + 0.2x_1 + 0.2x_2$

Kans dat de toestand tot groep C behoort:

- $\theta_{C,0} + \theta_{C,1}x_1 + \theta_{C,2}x_2$
- Voorbeeld: $0.1 + 0.3x_1 + 0.3x_2$

3.3.2 Model

Het softmax-model berekent de mate van zekerheid dat een toestand tot een bepaalde categorie behoort.

vb: volgende quotiënt drukt uit hoe zeker hij is dat (z_1, z_2) tot categorie A behoort:

$$\frac{e^{\theta_{A,0} + \theta_{A,1}z_1 + \theta_{A,2}z_2}}{e^{\theta_{A,0} + \theta_{A,1}z_1 + \theta_{A,2}z_2} + e^{\theta_{B,0} + \theta_{B,1}z_1 + \theta_{B,2}z_2} + e^{\theta_{C,0} + \theta_{C,1}z_1 + \theta_{C,2}z_2}}$$

(analoog voor categorie B en C: vervang de teller)

$$\frac{e^{0.01 + 0.1 \cdot 0.1 + 0.1 \cdot 0.5}}{e^{0.01 + 0.1 \cdot 0.1 + 0.1 \cdot 0.5} + e^{0.1 + 0.2 \cdot 0.1 + 0.2 \cdot 0.5} + e^{0.1 + 0.3 \cdot 0.1 + 0.3 \cdot 0.5}} = 0.2945$$

Figuur 30: Betekenis: het model is 29% zeker dat $(0.1, 0.5)$ tot categorie A behoort. Bereken zelf als oefening voor B en C

3.3.3 Wiskundig

Het gebruikte model wordt via volgende wiskundige formule algemeen beschreven:

$$\frac{e^{x_k}}{\sum_{i=1}^n e^{x_i}} \quad (19)$$

waarbij:

- $x_k = \theta_{k,0} + \theta_{k,1}x_1 + \theta_{k,2}x_2 + \dots + \theta_{k,m}x_m$
- n = aantal groepen
- m = het aantal meetcriteria

3.4 Logistic regression cost function

Het model:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^\top x}} \quad (20)$$

waarbij:

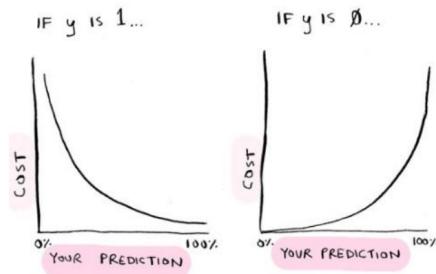
- $\theta^\top x = \theta_0 + \theta_1x_1 + \theta_2x_2$
- h_θ drukt uit wat de kans is dat voor opgegeven x_1 en x_2 de waarneming tot 1 groep behoort
- x_1 en x_2 zijn de inputwaardes
- θ_1 en θ_2 zijn gewichten (hoe belangrijk is de input)
- **Doel:** vinden van de beste gewichten zodat de voorspelling == de werkelijkheid

3.4.1 Success meten

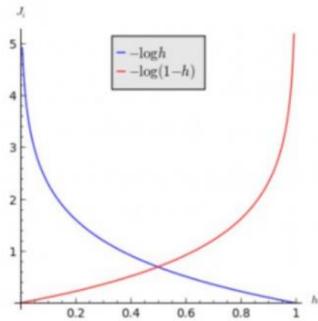
Stel: je maakt een logistisch regressiemodel die bepaalt of een object een groene appel of een tennisbal is.

- Bepalen van de kostenfunctie $J(\theta)$ met als doel deze zo laag mogelijk te brengen
- kost = afwijking tegenover de werkelijke situatie
- werkelijkheid kan 2 situaties zijn:
 - Indien de werkelijkheid een groene appel is $\Rightarrow y = 1$
 - Indien de werkelijkheid géén groene appel is $\Rightarrow y = 0$

Hoe ziet zo'n kostfunctie er dan uit?



Figuur 31: Als $y = 1$ en $y = 0$



Figuur 32

$$\begin{aligned} -\log(h_\theta(x)) & \quad \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \quad \text{if } y = 0 \end{aligned}$$

Figuur 33

Hoe brengen we 2 mogelijke situaties in 1 functie samen?

(TODO: slide 24 - 32)

4 Pandas library

4.1 Inleiding

- Doelstelling:
 - Nut van de pandas library kunnen situeren
 - Data-analyse: basisbewerkingen
- Pandas = ‘Python Data Analysis Library’
- Pandas bouwt op de NumPy library
- Officiële website: <https://pandas.pydata.org/>
- Goede start: <http://pandas.pydata.org/pandas-docs/stable/10min.html>

4.1.1 Welke data verwerken?

- csv-files
- txt-files
- Excel-files
- Databases

4.2 Pandas.core

Beschikbare datastructuren:

- Series (1D)
- DataFrame (2D)
- Panel (3D)

4.3 Series

Bestemd voor 1-dimensionale data:

'a one-dimension labeled array capable of holding any data'

- Subklasse van numpy-ndarray
- Data: elk soort datatype
- Geordende index
- Duplicaten mag (maar niet optimaal)

	index	values
A	→	5
B	→	6
C	→	12
D	→	-5
E	→	6.7

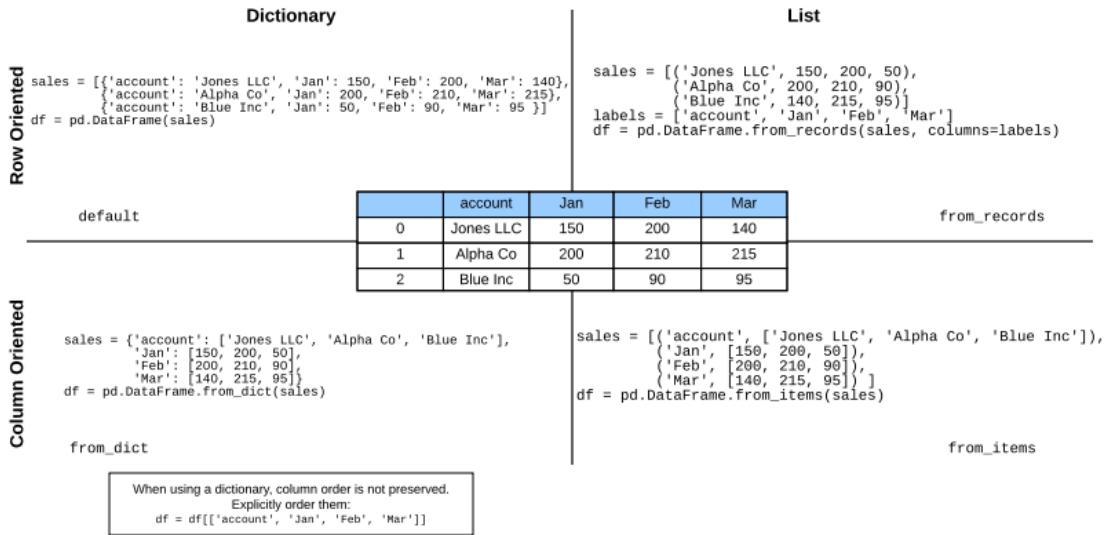
Figuur 34: Elk element heeft een index

4.4 DataFrame

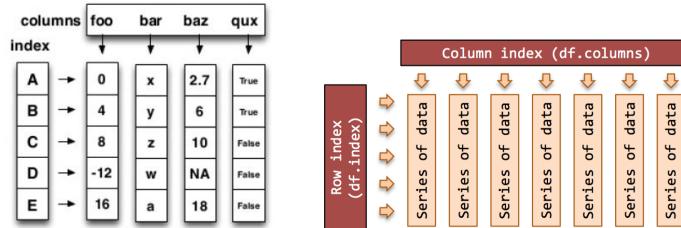
Bestemd voor meer-dimensionale data

- Subklasse van numpy-ndarray
- Elke kolom kan ander datatype hebben
- Rij en kolom index
- Grootte wijzigbaar (invoegen/verwijderen van rijen en kolommen)

Creating Pandas DataFrames from Python Lists and Dictionaries



Figuur 35: DataFrames maken uit Python lists en dictionaries



Figuur 36: Elk element heeft een rij en kolom

4.4.1 Select data from DataFrame

Via operator [] selecteer je een kolom:

	Feb	Jan	Mar	account
0	200	150	140	Jones LLC
1	210	200	215	Alpha Co
2	90	50	95	Blue Inc

Figuur 37: Voorbeeld DataFrame

```
# selecteer de kolom Jan uit dataframe
df['Jan']
```

```
# analoog: elke kolom is dus een attribuut van dataframe
df.Jan
```

```
# returnwaarde: Series-object
```

```

0    150
1    200
2     50
Name: Jan, dtype: int64

```

Via operator [] selecteer je een kolom & krijg je een dataframe terug:

```

>> df[['Jan']]
# returns:
   Jan
0  150
1  200
2   50

>> df[['Jan', 'Mar']]
# returns:
   Jan    Mar
0  150   140
1  200   215
2   50    95

```

Via de operator [] en met een conditie:

	Feb	Jan	Mar	account
0	200	150	140	Jones LLC
1	210	200	215	Alpha Co
2	90	50	95	Blue Inc

Figuur 38: Voorbeeld DataFrame

```

>> df[df.Jan > 60]
# returns:
   Feb    Jan    Mar    account
0  200   150   140  Jones LLC
1  210   200   215  Alpha Co

>> df[np.logical_and(df.Jan > 100, df.Feb <= 200)]
# returns:
   Feb    Jan    Mar    account
0  200   150   140  Jones LLC

>> df[df.account.str.startswith('Alpha')]
# test deze eens zelf uit als oefening :(

```

4.4.2 Veelgebruikte commandos bij dataframes

df.shape	# geeft de dimensie als een tuple terug
df.info()	# oplijsting van de aanwezige kolommen
df.head([aantal])	# eerste vijf/aantal rijen
df.tail([aantal])	# laatste vijf/aantal rijen
df.index	# geef de index-kolom weer
df.columns	# geef de kolomnamen weer
df.describe()	# geef snel overzicht van statistische data
df.T	# transponeer data (rij -> kol, kol -> rij)

```
df.sort_index()      # sorteert op basis van index
df.sort_values()    # sorteren op één of meerdere kolommen
```

4.5 Loc vs iloc

4.5.1 iloc

= Integer-location based indexing / selection by position

- Nut: selecteren van rijen en kolommen via rij/kolomnummer
- Syntax: data.iloc[<row>, <column>]
- Returnwaarde:
 - Indien 1 **rij** ⇒ series-object
 - Indien meerdere **rijen**: ⇒ dataframe-object
 - 1 of meerdere **kolommen**: ⇒ dataframe-object

iloc-voorbeelden:

```
# Rows:
data.iloc[0] # first row of data frame
data.iloc[1] # second row of data frame
data.iloc[-1] # last row of data frame

# Columns:
data.iloc[:,0] # first column of data frame
data.iloc[:,1] # second column of data frame
data.iloc[:, -1] # last column of data frame

data.iloc[0:5] # first five rows of dataframe

# first two columns of data frame with all rows
data.iloc[:, 0:2]

# 1st, 4th, 7th, 25th row + 1st 6th 7th columns.
data.iloc[[0,3,6,24], [0,5,6]]

# first 5 rows and 5th, 6th, 7th columns of data frame
data.iloc[0:5, 5:8]
```

4.5.2 loc

= label based indexing / selection

- Nut: selecteren van rijen en kolommen via label / via conditionele look-up
- Syntax: data.loc[<row>, <column>]
- Returnwaarde:
 - Indien 1 **rij/kol** ⇒ series-object
 - Indien meerdere **rijen**: ⇒ dataframe-object
 - 1 of meerdere **kolommen**: ⇒ dataframe-object

loc-voorbeelden:

	cars_per_cap	country	drives_right
US	809	United States	True
AUS	731	Australia	False
JAP	588	Japan	False
IN	18	India	False
RU	200	Russia	True
MOR	70	Morocco	True
EG	45	Egypt	True

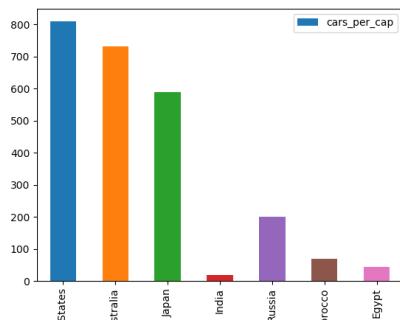
Figuur 39: Voorbeeld dataframe

```
reviews.loc[:2, "score"] # return type =  
  
reviews.loc[:2, ["score", "title"]] # return type =  
  
# select column "score" where value of index <= 5  
reviews.loc[:5, "score"]  
  
# select columns "country" and "cars_per_cap" where rowindex is "US" or "RU"  
cars.loc[ ["US","RU"] , ["country","cars_per_cap"]]  
  
# select columns "country" and "cars_per_cap" where rowindex is from "US" to "RU"  
cars.loc[ "US " : "RU " , ["country","cars_per_cap"]]  
  
# selectie rijen hoeft niet altijd op basis van row-index te zijn  
# select columns "country" and "drives_right", voor de landen 'Japan' en 'India'  
cars.loc[ cars.country.isin( ['Japan', 'India'] ) , ['country','drives_right']]
```

4.6 Plotten met pandas

4.6.1 Dataframe plotten

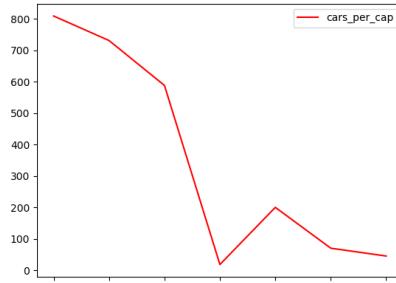
```
# print(cars[['country', 'cars_per_cap']])  
# werkwijze 1:  
cars[['country', 'cars_per_cap']].plot(kind='bar', legend=True)  
# werkwijze 2:  
cars.plot(x='country', y='cars_per_cap', kind='bar', legend=True)  
  
plt.show()
```



Figuur 40: Resultaat

4.6.2 Series plotten

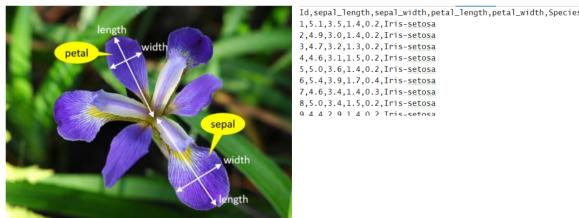
```
# werkwijsje 1:  
plt.plot(cars['cars_per_cap'])  
# werkwijsje 2:  
plt.plot(cars['cars_per_cap'].plot(color='r', legend=True))  
  
plt.show()
```



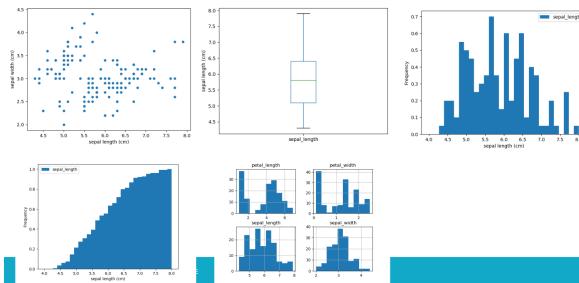
Figuur 41: Resultaat

4.7 Demo: Iris Dataset

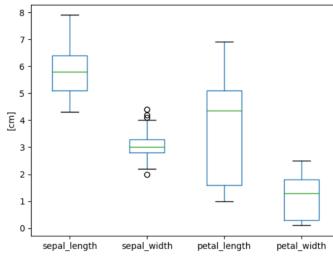
(Zie DemoPandas.zip op Leho voor de code)



Figuur 42: Een iris bestaat uit petals & sepals, met elk hun breedte en lengte

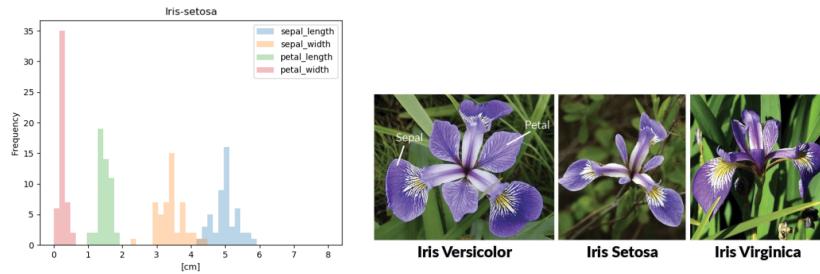


Figuur 43: Plotten van de beschikbare data (demo5.py)



Figuur 44: Vergelijken van de lengtes en breedtes adhv box-plots (demo6.py)

```
# filteren van data
result_check = iris['Species'] == 'Iris-setosa'
# print(type(result_check)) == TODO
filtered_setosa = iris.loc[result_check, :]
# of in 1 lijn:
filtered_setosa = iris.loc[iris['Species'] == 'Iris-setosa', :]
```



Figuur 45: Frequentiediagram voor de Iris-setosa soort (demo7.py)

4.8 Complexe bewerkingen

		<code>df.sort_values('mpg')</code> Order rows by values of a column (low to high).
<code>df.rename(columns = {'y':'year'})</code> Rename the columns of a DataFrame	<code>df.sort_index()</code> Sort the index of a DataFrame	<code>df.reset_index()</code> Reset index of DataFrame to row numbers, moving index to columns.
<code>pd.concat([df1,df2])</code> Append rows of DataFrames	<code>pd.concat([df1,df2], axis=1)</code> Append columns of DataFrames	<code>df.drop(['Length', 'Height'], axis=1)</code> Drop columns from DataFrame

Figuur 46: Reshaping data: change the layout of a data set

(komen we later nog op terug)