

# Supervised learning - regression

JENS BAETENS

# Lineaire regressie

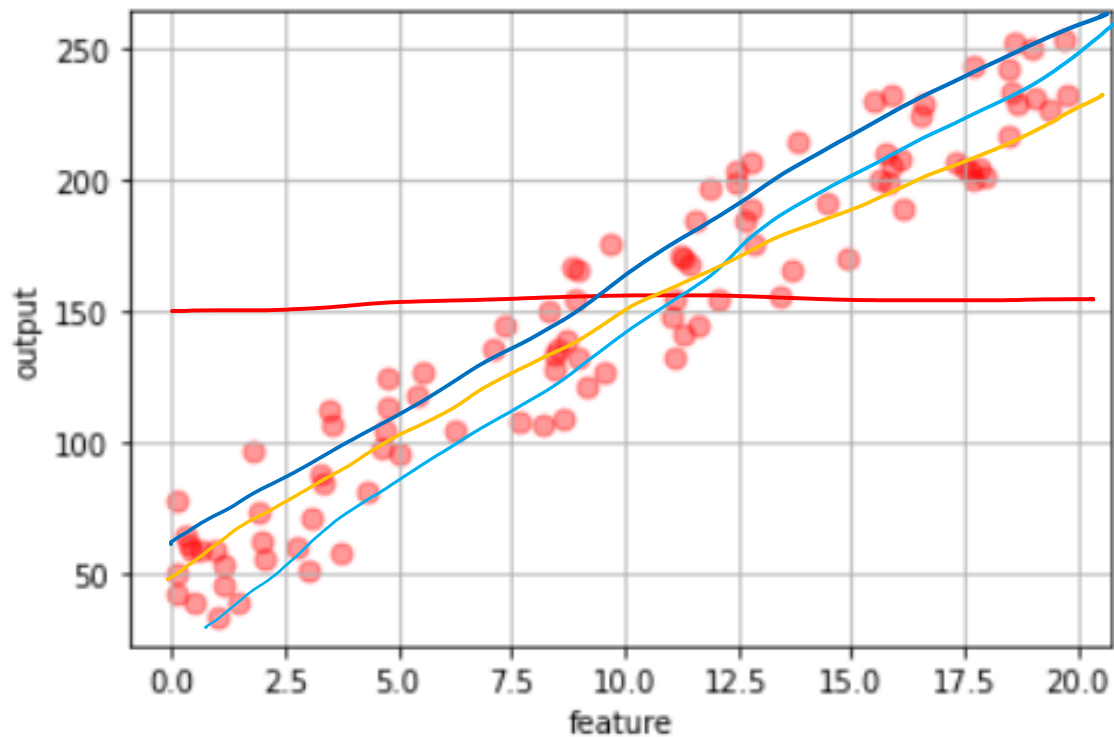
- Voorspel het resultaat in de output kolom op basis van de inputs (hier de feature kolom)
- Output wordt ook vaak target genoemd
- Trainingsset = 100 training examples
- Output is een (continue) variabele

Input      output  
example    label  
rij          target  
voorbeeld  
X          variabele

1	4.294681	81.283221
2	18.450278	217.219276
3	1.454430	39.722608
4	15.529496	230.239091
5	0.994415	33.785656
6	17.832737	204.194535
7	9.533831	127.256491

nieuw 25 → ?

lineaire regressie = rechte lijnen

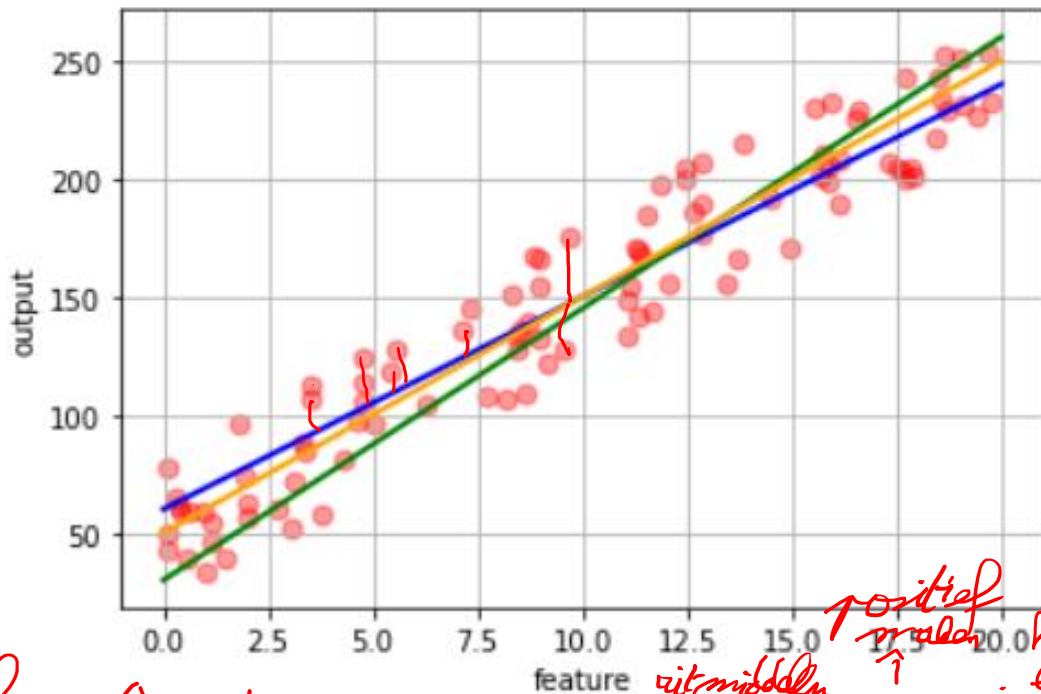


# Wat is het beste model?

↳ loss error bereken

via een loss/error functie

↓ MSE  
least mean square  
↓ MAE  
minimum error



positief  
negatief  
uitmiddelen  
↑  
square  
absolute  
↑  
hoe ver  
ernast  
↑  
error

# Enkelvoudige lineaire regressie

*1 input/feature*

Zoek verband feature en output

Lineaire trendlijn  $f(x)$

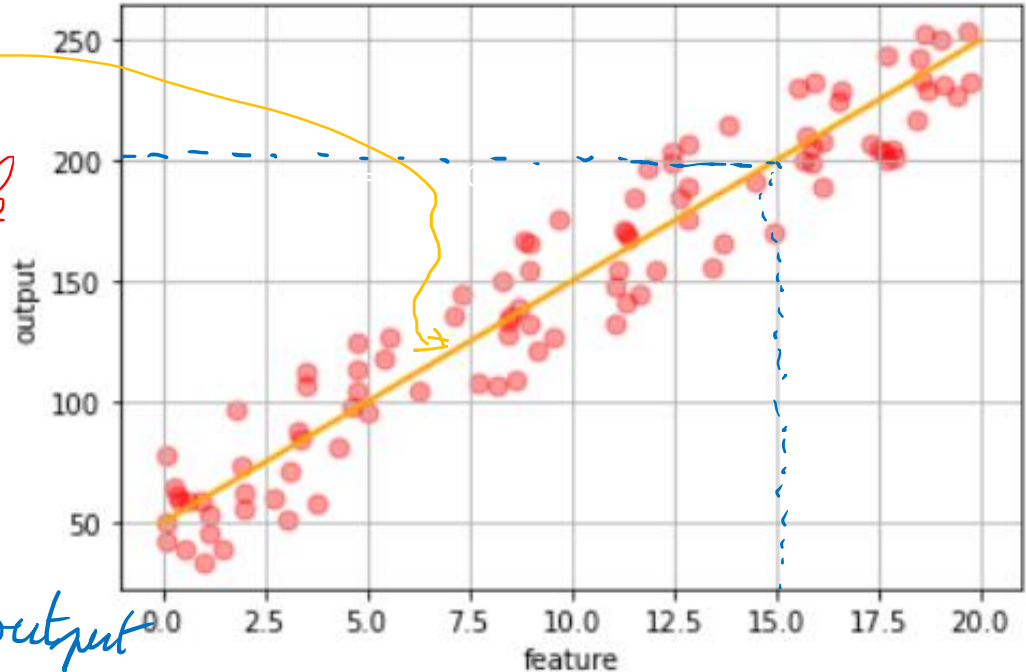
Enkelvoudig of univariate

*1 variable*

de optimale  
rechte lijn

input 15

$f(15) = 200 = \text{output}$



# Enkelvoudige lineaire regressie

De trendlijn = Het verband tussen twee waarden

*dit is in het net  
= vector*

$$f_{\mathbf{w}}(x) = w_0 + w_1 x = \text{target}$$

Regressie zoekt de optimale waarden voor  $w_0$  en  $w_1$

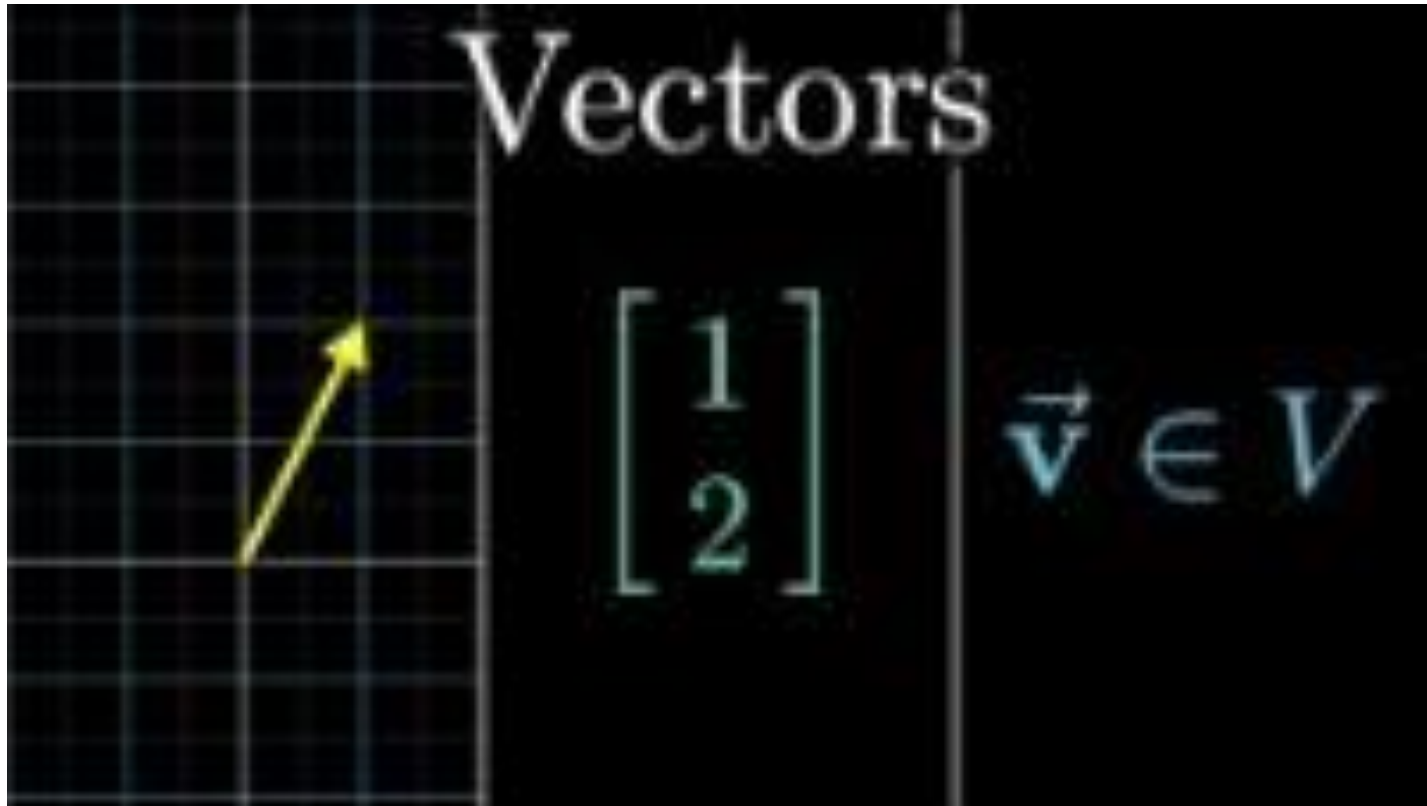
Deze waarden worden **gewichten** genoemd (**weights**) of de te trainen

**parameters**

*→ worden gebruikt door het model / ML-techniek*

- Gecombineerd voorgesteld als vector  $\mathbf{w} = [w_0, w_1]$

Het zoeken van het trendlijn / model / hypothese = training / learning



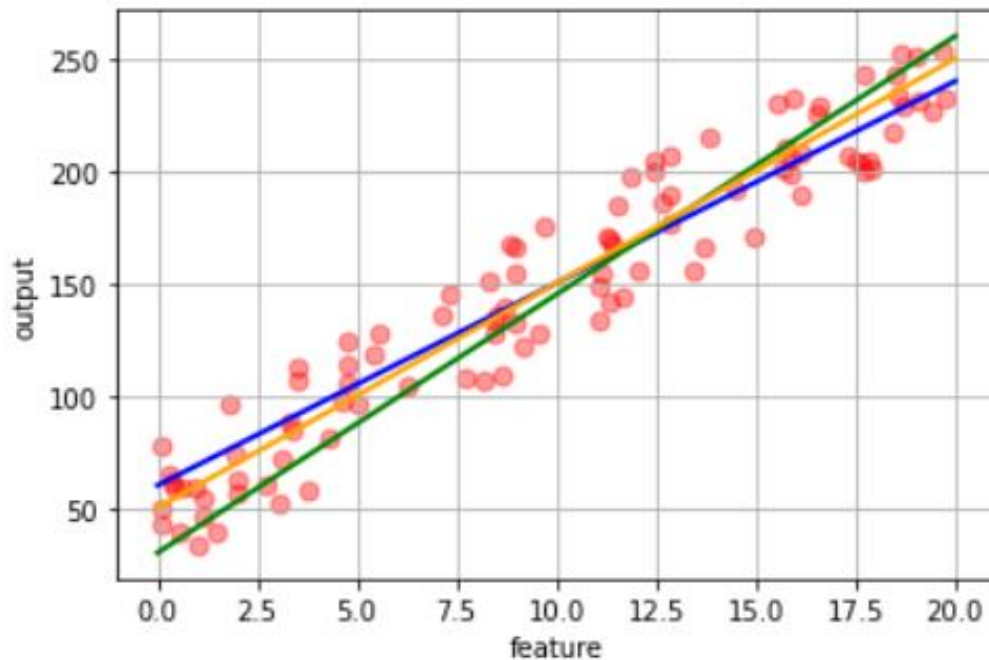
[https://youtu.be/fNk\\_zzaMoSs](https://youtu.be/fNk_zzaMoSs)

Wat is  
het beste  
model?

$$f(x) = w_0 + w_1 x$$
$$\Rightarrow [x^0 \ x^1] \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = ?$$

$x = 15$   
 $\downarrow$

$$\Rightarrow [1 \ 15] \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = 200?$$





# Wat is het beste model?

Beste model wordt gekozen door minimalisatie van een kostenfunctie.

Bvb: Least Mean Squares (LMS) voor  $N$  examples met input  $x^i$  en targets  $y^i$

$$L(\mathbf{w}) = \frac{1}{2N} \sum_{i=1}^N (f_{\mathbf{w}}(x^i) - y^i)^2 \rightarrow \text{positief}$$

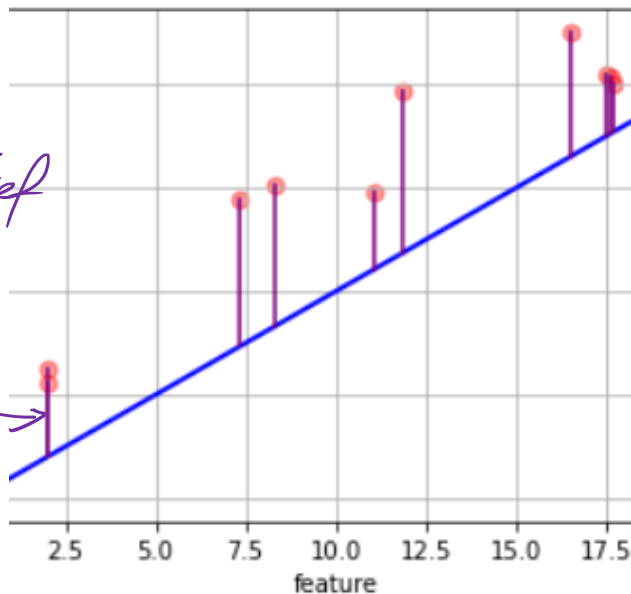
$N = \text{aantal rijen}$

$$\sum_{i=1}^N x[i] =$$

mean uitmiddelen

$\text{res}_{\text{old}} = 0$

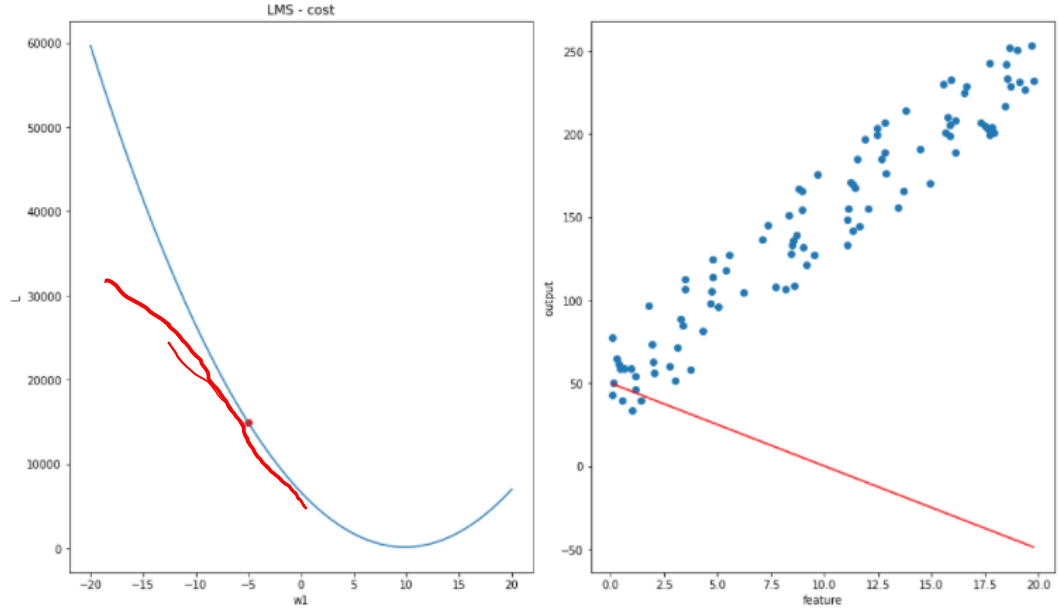
for  $x$  from 1 to  $N$   
 $\text{res} += x[i]$



# Least Mean squares

Afgeleide van de cost-  
functie

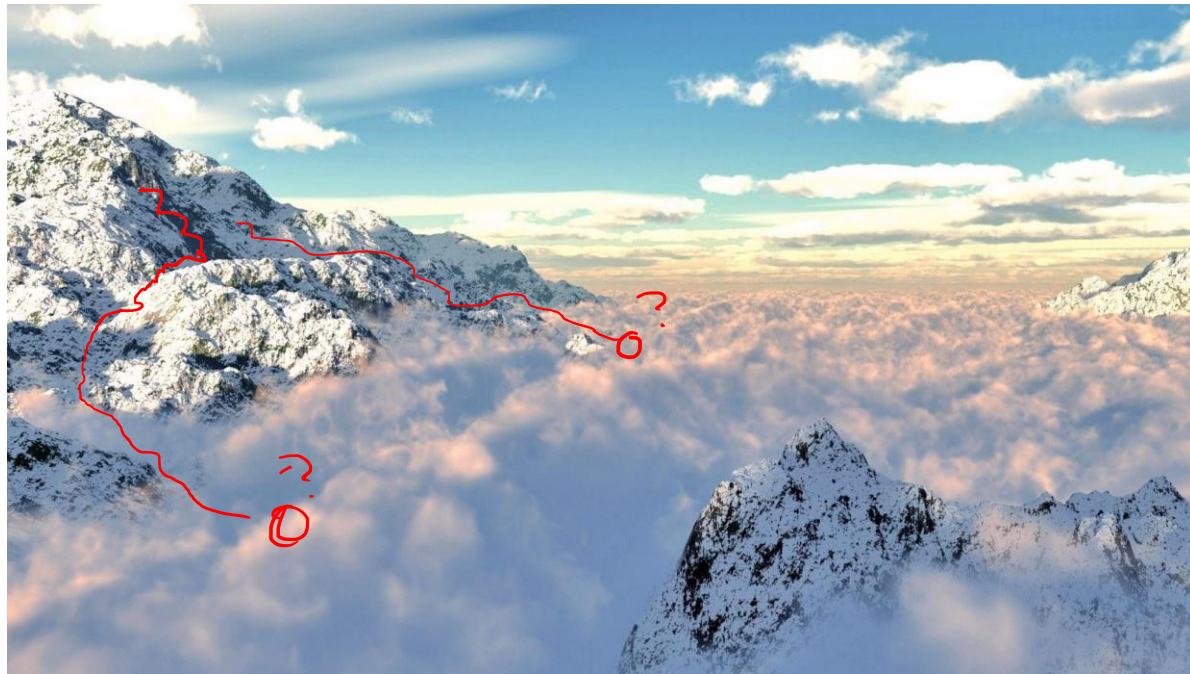
↳ geeft aan hoe de gewichten moeten veranderen





# Gradient descent

lokal minimum  
global minimum



# Gradient Descent – Lokaal Minimum?

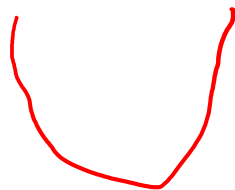
LMS-functie is convex

- Hierdoor altijd global minimum

Bij neurale netwerken kan het wel

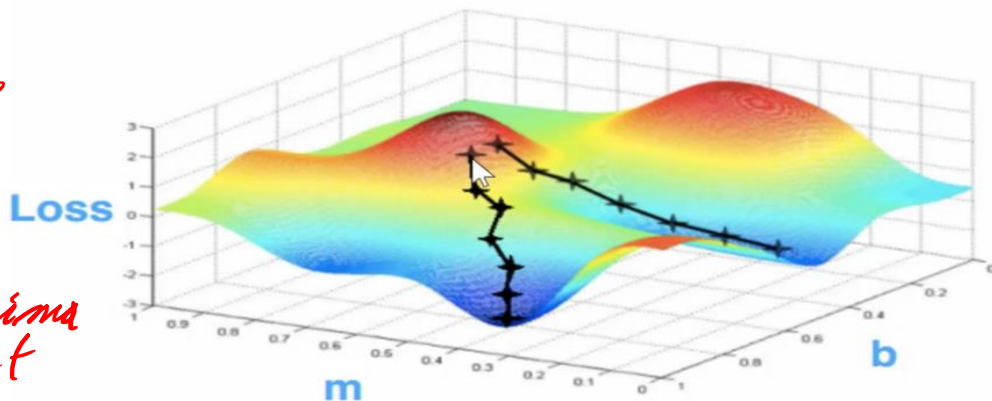
LMS

↳ parabolische functie



→ Convex

→ lokaal minima  
kan niet



Gemiddelde kwadratische fout

$$\underline{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

Gemiddelde absolute fout

$$\underline{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

$1-5 = 5$   
 $151 = 5$

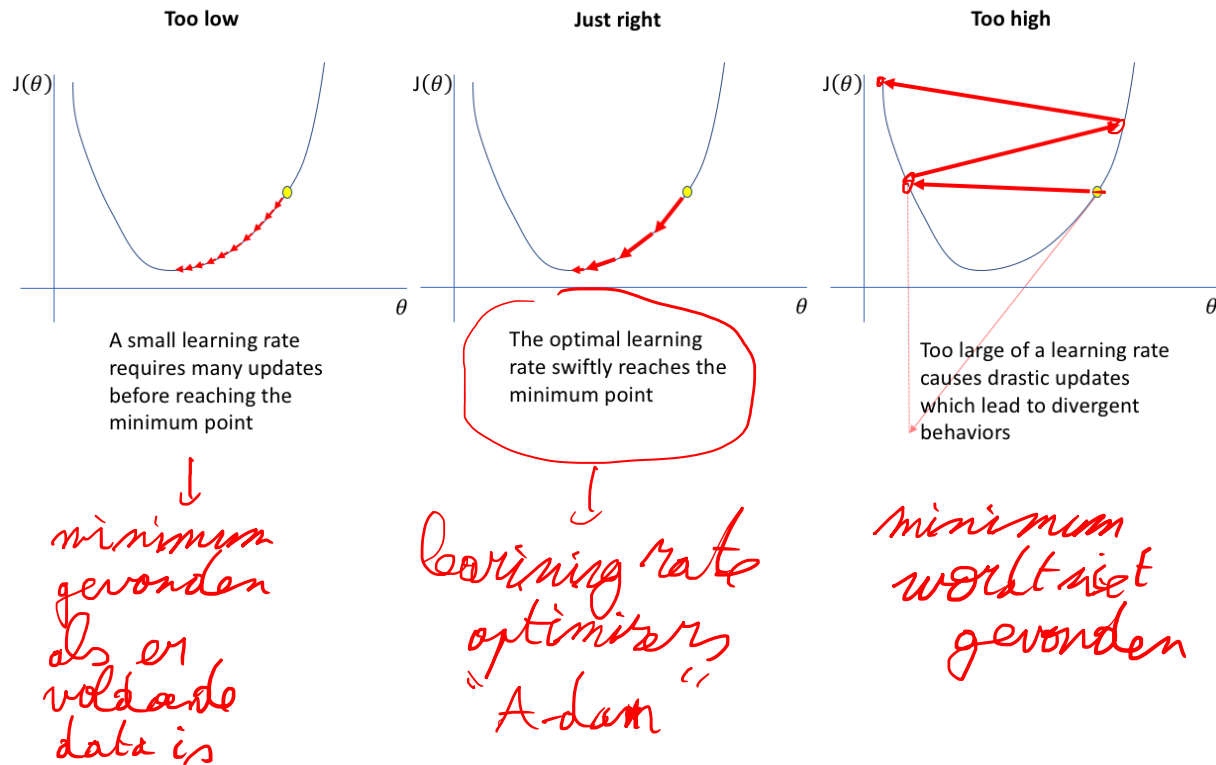
~~Determinatiecoëfficiënt~~

$$\underline{R^2} = \frac{\sum_{i=1}^N (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^N (y_i - \bar{y})^2}$$

# Gradient Descent – Learning rate

Bepaalt hoe snel je het optimum benaderd.

“De grootte van de stappen”



# Trainen van het model

*Voor de technische kant*

Zelf implementeren of gebruik maken van bestaande frameworks (sklearn)

Construct model => Fit model => Make predictions

Hyperparameters

↓  
kunen we kiezen

*trainen*

↓  
weights  
gaan bepalen  
↓  
gewichten/parameters  
niet te kiezen

↳ op nieuwe data  
↳ evalueren hoe goed het werkt

# Meerdere features

In de praktijk zijn er normaal meer features beschikbaar.

- Meervoudige of multiple regression

Bovenstaande formules aan te passen met meer gewichten.

Hoeveel extra gewichten per feature nodig?





# Meerdere features – aanpassingen?

1 feature met 2 gewichten  $\rightarrow$  ~~1~~  $M$  features met ? gewichten

- $f_w(x) = w_0 + w_1 x = \text{target}$

$$\mathbf{w} = [w_0, w_1]$$

$$L(\mathbf{w}) = \frac{1}{2N} \sum_{i=1}^N (f_w(x^i) - y^i)^2$$

kan we samenvoegen

$$w = [x_1^0 \ x_1^1 \ x_2^0 \ x_2^1 \ \dots \ x_M^0 \ x_M^1]$$

$$w = [w_0 \ w_1 \ \dots \ w_{M+1}]$$

$$f_w(x) = w_0 + w_1 x_1 + w_2 x_2 + \dots + w_M x_M$$

Hoe ziet dit er wiskundig uit?

3 features  $x_1, x_2, x_3$

$x_1$

$x_2$

$x_3$

$$\text{sample } \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} \begin{bmatrix} 1 & 15 & 20 & 25 \\ 1 & 5 & 6 & 8 \\ 1 & 20 & -5 & 10 \\ 1 & 100 & -50 & 60 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$N \times 4$

$4 \times 1$

$N \times 1$

aantal  
- samples  
- rijen

# features + 1

1 output waarde  
1 getal

# Feature engineering - Normalisation

→ op numerieke data

↳ dit mag je niet doen voor de output

↳ features op dezelfde grootte orde brengen

Herschaal elke kolom (behalve target) zodat

- Gemiddelde gelijk aan 0
- Standaardafwijking is 1

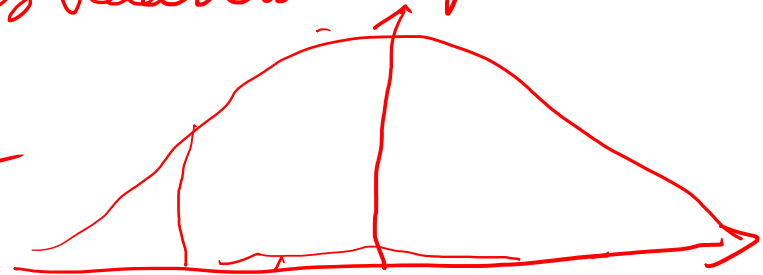
- Statistische normalisatie

```
1 scaler = StandardScaler().fit(X_train)
2 X_train = scaler.transform(X_train)
3 X_test = scaler.transform(X_test)
```

Andere vormen:

- Delen door het maximum
- Schalen naar het interval 0-1

→ bij beeldverwerking

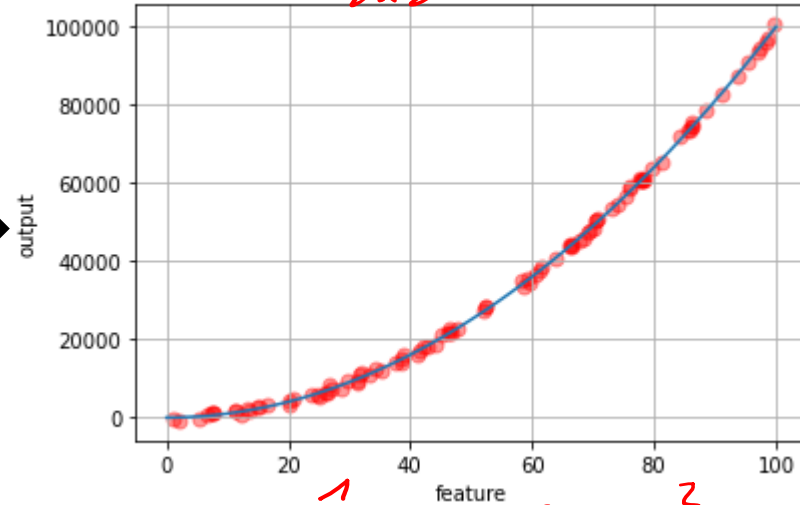
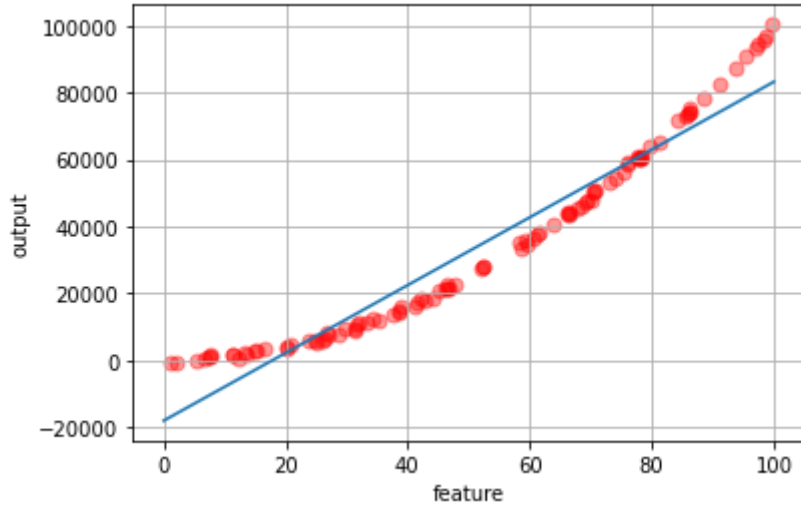


Belangrijk omdat er gewerkt wordt met afstanden om errors te berekenen

# Feature engineering – Higher order

$$f_w(x) = w_0 + w_1x_1 + w_2x_2 + \dots$$

$$\rightarrow f_w(x) = w_0 + w_1x_1 + w_2x_1^2 + w_3x_2 + w_4x_2^2 + \dots$$



Kwadratische features  
toevoegen

$$\begin{matrix} & & 1 & & \\ & 1 & & 1 & \\ & 1 & 2 & 1 & \\ & 1 & 3 & 3 & 1 \\ 1 & 4 & 6 & 4 & 1 \\ & & \vdots & & \end{matrix} \quad \begin{matrix} (x+y)^2 \\ x^2 + y^2 + 2xy \end{matrix}$$

# Feature engineering – extra features

Bedenken van nieuwe features ~~en~~

- Oppervlakte op basis van breedte en lengte  $x \cdot y$
- Uit start en eindpunt de afstand halen  ~~$\sqrt{x^2 + y^2}$~~   $\sqrt{x^2 - y^2}$
- Snelheid bereken op basis van afgelegde afstand en duur van de rit
- Dag van de week of welke maand het is uit de datum halen.
- ...

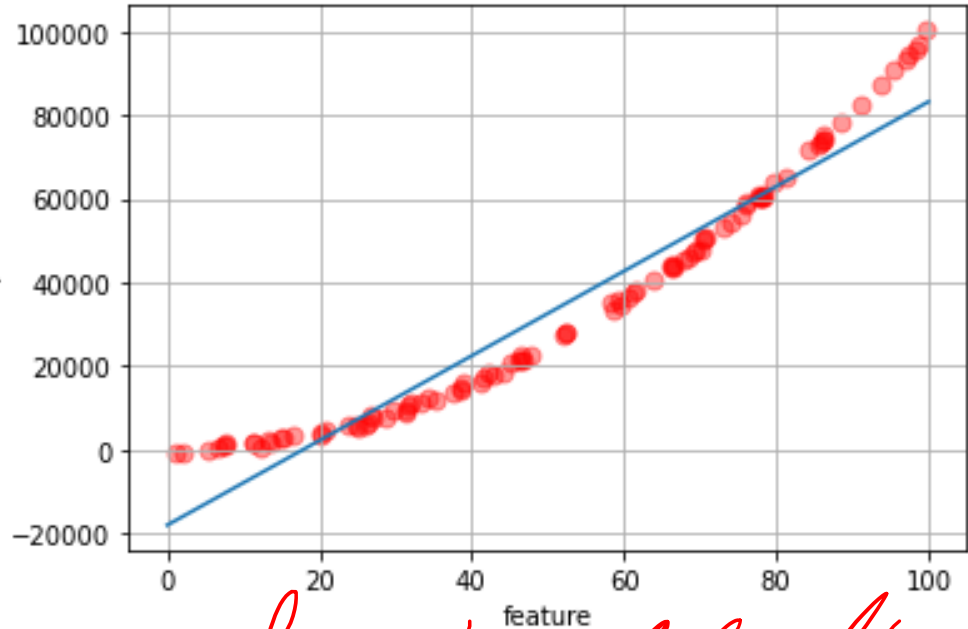
*Niet echt schadvaan  $\rightarrow$  automatiseren*

# Underfitting

Model is te eenvoudig om de data correct te modelleren

↳ train  
test

error/loss  
groot  
groot



oorzaken → dataset bevat niet voldoende informatie  
→ model niet complex genoeg

train      error/loss  
test      laag  
            hoog

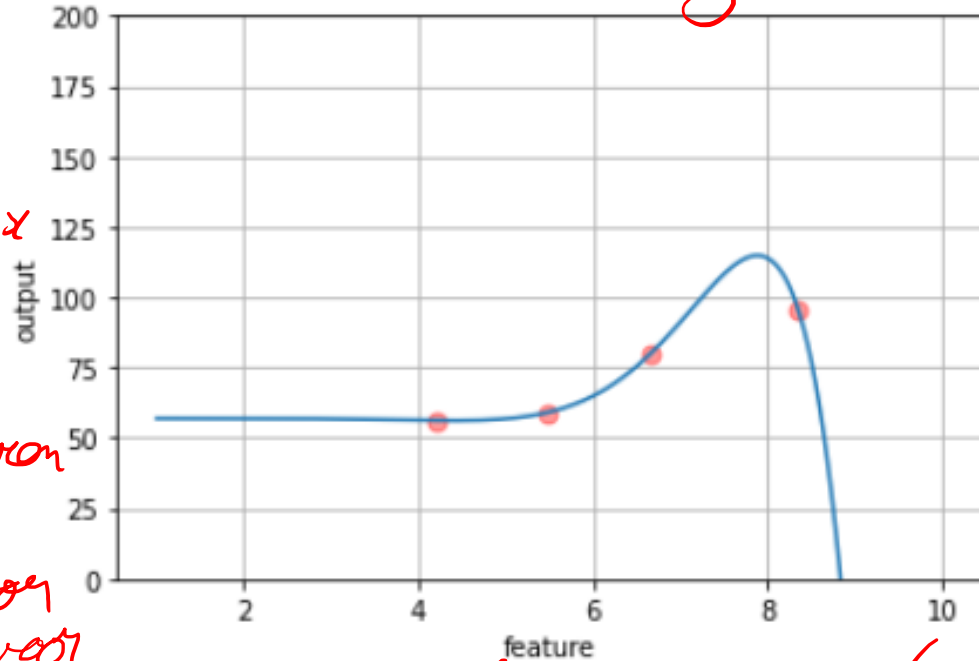
# Overfitting

↳ model te complex  
is

↳ de trainingsdata  
vanleiden aan het leren  
is

→ er kunnen hierdoor  
problemen zijn voor  
testdata

→ generaliseert niet voldoende



oplossingen meer data  
→ regularisatie

# Overfitting - regularisatie

Extra term in de kostenfunctie voor het gebruik van features te penaliseren

$$L(\mathbf{w}) = \frac{1}{2N} \sum_{i=1}^N (f_{\mathbf{w}}(x^i) - y^i)^2 + \lambda R(\mathbf{w})$$

*↳ alpha uit de cost*

De parameter  $\lambda$  is de mate waarin er regularisatie is

- 0 -> geen regularisatie
- $\infty$  -> alle gewichten zijn nul



# Overfitting – L2norm → Ridge by linear regression

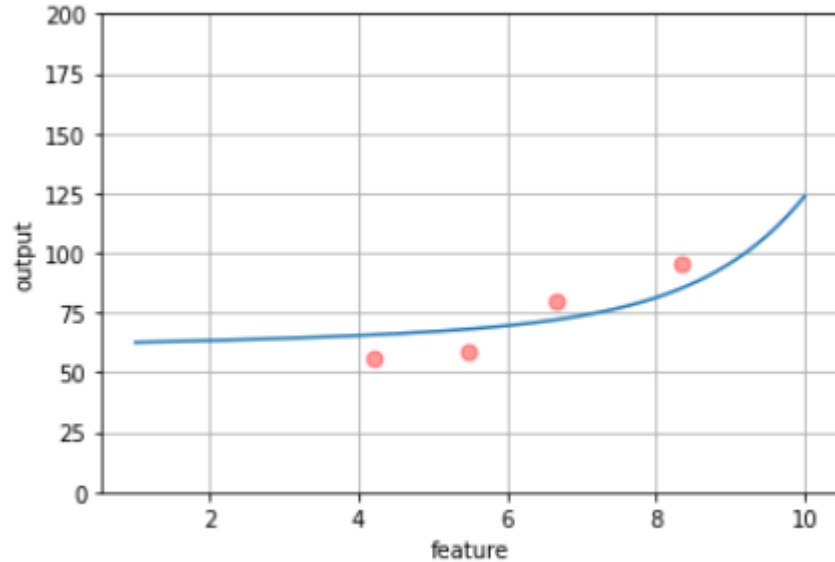
$$\text{Regularisatieterm} = \sum_{i=1}^N w_i^2$$

Merk op dat de som begint vanaf 1

De bias wordt niet in rekening gebracht

→ er is een parameter om deze mee in rekening te nemen

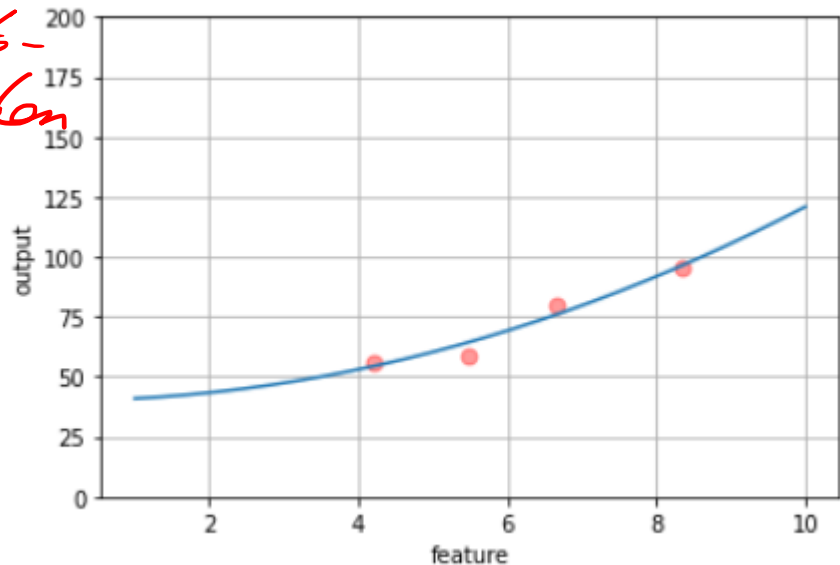
↳ vaak dit niet doen → onafh van de features



# Overfitting – L1norm $\rightarrow$ *lasso*

Regularisatieterm =  $\sum_{i=1}^N |w_i|$  *isolate-  
waarden*

Voordeel is dat gewichten op nul  
gezet kunnen worden



# Lexicon

- Supervised
- Unsupervised
- Reinforcement Learning
- Regression
- Overfitting
- Underfitting
- Learning Rate
- Loss Function
- Feature Engineering
- Normalisation
- Regularisation
- Trainen van een model

## 0. Connection slide



Click on the projected screen to start the question



[Copy participation link](#)

wooclap




100 %



12



 You cannot vote anymore



Overfitting kan gedetecteerd worden door

1

een hoge error op de trainings- en testdata

33%

3 

2

een hoge error op de trainingsdata en een lage error op de testdata

33%

3 



Click on the projected screen to start the question

3

een lage error op de trainingsdata en een hoge error op de testdata

33%

3 

4

een lage error op de trainings- en testdata

0%

0 



wooclap



100 %



33% correct

9 / 12



Go to **www.wooclap.com** and use the code **DGMOAR**



Wat is het verschil tussen parameters en hyperparameters?

parameters is van data en hyperparameters is handmatig ( zelf dus ) gedaan en word gebruikt in...



parameters worden geschat



hyperparameters niet



Click on the projected screen to start the question

hyper parameters zijn pre-set en reguliere parameters niet



voor de ene staat hyper en voor de andere niet



parameters automatisch uit data hyperparams



Go to **www.wooclap.com** and use the code **DGMOAR**



Welke technieken kan je gebruiken om overfitting tegen te gaan?

cross validatie



CrossValidation



grotere dataset & cross validation



cross-validatie



mehr data benutzen



**wooclap**



100 %



11



Go to **www.wooclap.com** and use the code **DGMOAR**



Wat is de loss function?

Een manier om te kijken hoe goed uw code uw data  
modeleert



Vertaald uit het Engels-In wiskundige optimalisatie- en  
beslissingstheorie is een verliesfunctie of kostenfuncti...



It's a method of evaluating how well your algorithm  
models your dataset



evalueert hoe goed je algoritme werkt



**wooclap**



100 %



9





# Opdracht

---

- Github classroom: <https://classroom.github.com/a/qESKmk8z>
- Een aantal groepen reeds aangemaakt voor een aantal verschillende ML-technieken
  - Logistic Regression, SVM, Naïve Bayes, Knearest Neighbour, Decision Trees, Random Forests
  - In groepjes van max 5
- Bereid een presentatie voor waarin je de techniek uitlegt
  - Hoe werkt de techniek?
  - Welke waarden kan je kiezen?
  - Wanneer kan het (niet) gebruikt worden?
  - Voor- en nadelen van deze techniek?
- Maak ook de oefening in de notebook met de aan je groep toegekende techniek
- Volgende les krijg je tijd om hieraan te werken en vragen te stellen