Wifi

- Naam: Mikrocentrum_gasten
- Wachtwoord: mikrowifi1115!

https://tinyurl.com/AIVEslides

mikrocentrum
opleiden • ontmoeten • ondernemen

mikrocentrum
opleiden • ontmoeten • ondernemen

Simcha van Helvoort
Dag 1

Artificial Intelligence voor Engineers

WIE BEN IK?

mikrocentrum
opleiden • ontmoeten • ondernemen

WIE ZIJN JULLIE?

- Wie ben je en wat is je achtergrond?
- Waar werk je en wat is je functie/dagelijkse bezigheden?
- Wat hoop je te leren?
- Wat is je ervaring met data en AI?

WAT WILLEN WE BEREIKEN?

mikrocentrum
opleiden • ontmoeten • ondernemen

Aan het eind…

- Kun jij zelf een data gedreven AI model maken
- Weten bij welke problemen je AI zou kunnen gebruiken
- Kan je van ruwe data informatie halen
- Kan je een AI model evalueren

Leerdoelen

- Python en libraries
- Crash course basis statistiek
- Wat is het verschil tussen AI en ML?
- Hoe werkt ML (globaal)
- Wat is het verschil tussen supervised en unsupervised learning?
- 3 verschillende ML technieken
- Hoe evalueer je een ML model?
- Hoe trek je conclusies uit een ML model?

Terminologie

- Begrippen en concepten zullen *schuin* gedrukt zijn
- Vaak direct uitgelegd
- Anders komt de uitleg later in de cursus
  of kan verder over gelezen worden in leesmateriaal

Waarmee gaan we werken

Programmeertaal: Python

Hoe: Jupyter Notebook op je eigen machine via Anaconda

Bestanden: https://github.com/SimchaGD/AIVE

De core ligt op deze packages:

- SciKit Learn - https://scikit-learn.org/
- Lifelines - https://lifelines.readthedocs.io/
- Numpy - https://numpy.org/
- Pandas - https://pandas.pydata.org/
- Matplotlib - https://matplotlib.org/
- Scipy - https://www.scipy.org/

Notebooks

- Hoe werkt het? (demo)
- Voordelen

jupyter Untitled Last Checkpoint: 05/31/2021 (unsaved changes)

Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
filename = '../3dprinter/data.csv'
df = pd.read_csv(filename)
```

In [3]:
```
df.head()
```

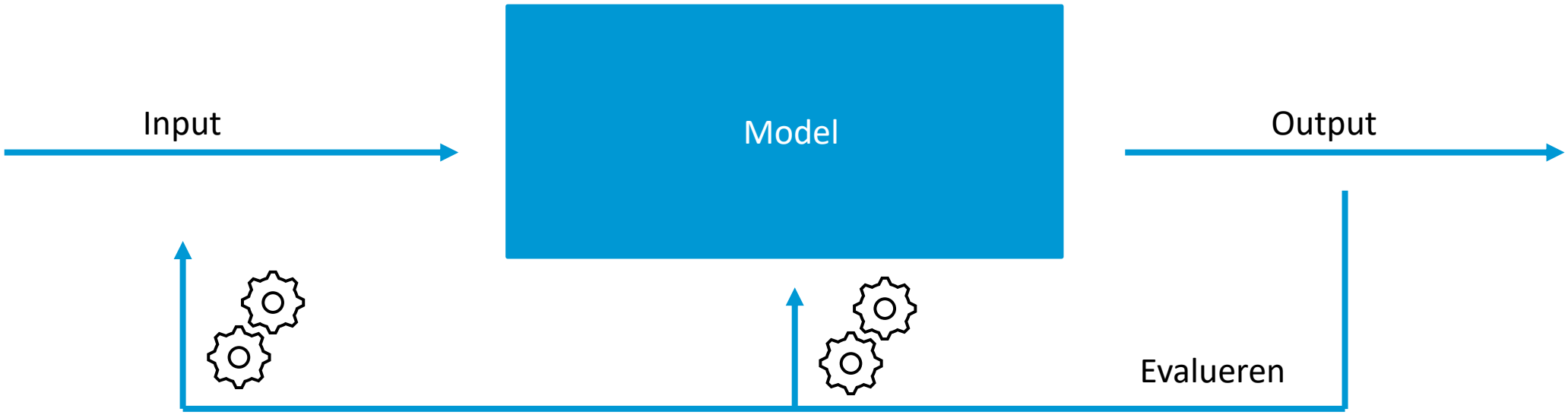
Out[3]:

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	material	fan_speed	roughness	tension_strenght
0	0.02	8	90	grid	220	60	40	abs	0	25	18
1	0.02	7	90	honeycomb	225	65	40	abs	25	32	16
2	0.02	1	80	grid	230	70	40	abs	50	40	8
3	0.02	4	70	honeycomb	240	75	40	abs	75	68	10
4	0.02	6	90	grid	250	80	40	abs	100	92	5

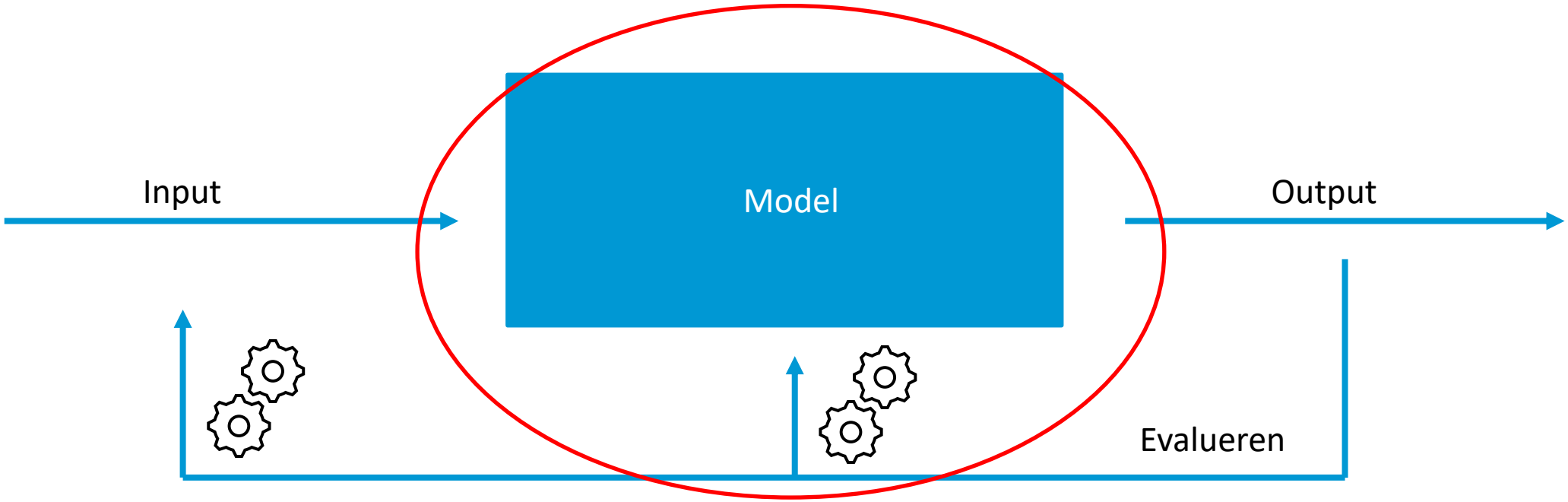
In [4]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
```

In [5]:
```
# Verander alle categorien naar cijfers. We zetten de infill_patern "grid" op 1 en "honeycomb" op 0.
df.infill_pattern = df.infill_pattern.apply(lambda x: 1 if x == "grid" else 0)

# We zetten de material "abs" op 1 en "pla" op 0.
df.material = df.material.apply(lambda x: 1 if x == "abs" else 0)

df.head() # Hier zien we dat de infill pattern 1 en 0 zijn geworden. En de kolom material staat er niet meer in
```

Pipeline
Waar gaan we naar toe werken



Vandaag



Maar eerst:

BASIS STATISTIEK

2 soorten data

Categorisch

- Benaderingen
- Groeperingen
- Beschrijvingen

Numeriek

- Exact
- Vaste ordening

Data type

Categorisch

Nominaal
Categorieën zonder volgorde
- Man/Vrouw
- Hond/Kat
- Groep A/ Group B/ Groep C

Ordinaal
Categorieën met volgorde
- Eens/Middel/Oneens
- Basisschool/ Middelbare school/ Universiteit

Numeriek

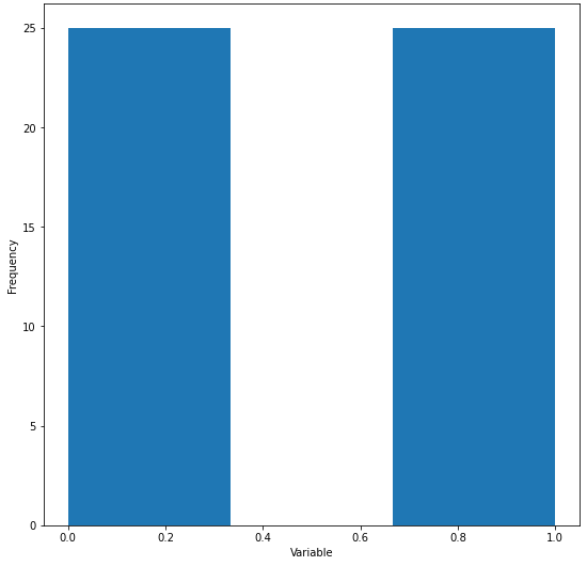
Interval
Numeriek zonder “echt nulpunt”. Gelijke afstand tussen waarden
- Temperatuur ˚C: -10/-5/0/5/10/20
- IQ
- Tijd op een klok

Ratio
Numeriek met nulpunt. Gelijke afstand
- Meter
- Gram
- Aantal presidenten van een land

Data types plotten

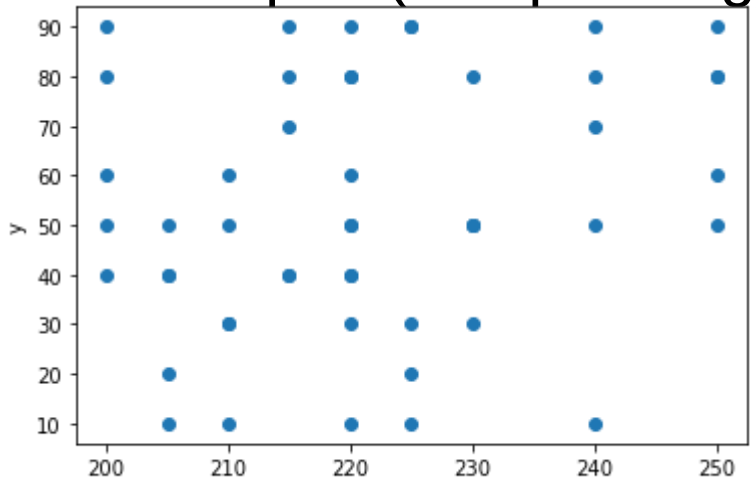
Categorisch

- Histogram (frequentie)

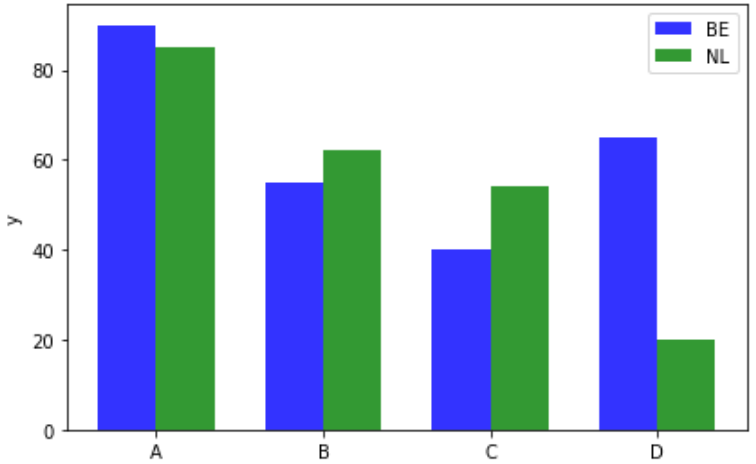


Numeriek

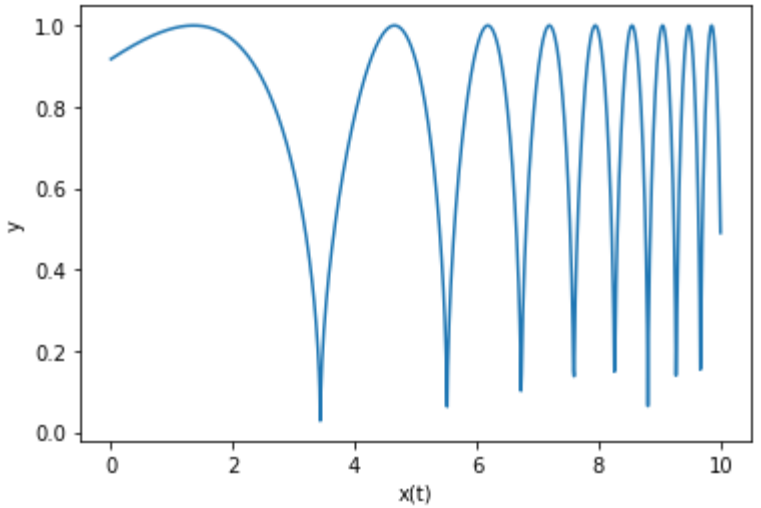
- Scatterplot (verspreiding)



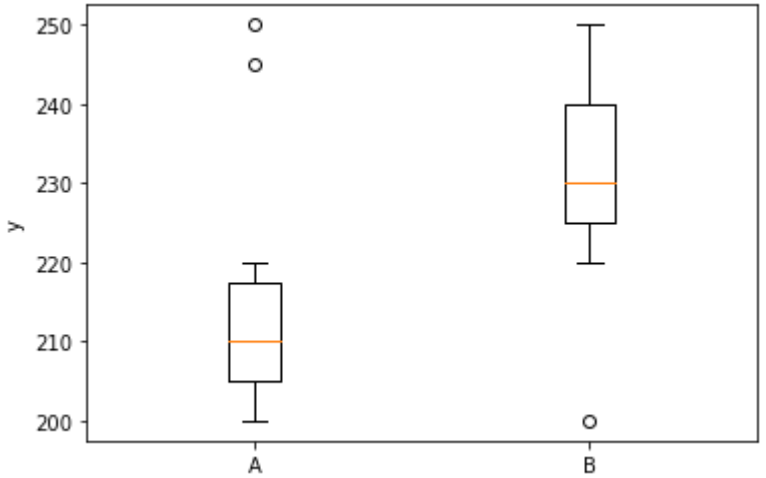
- Barchart (verhouding)



- Lineplot (verandering)



- Boxplot (verdeling)

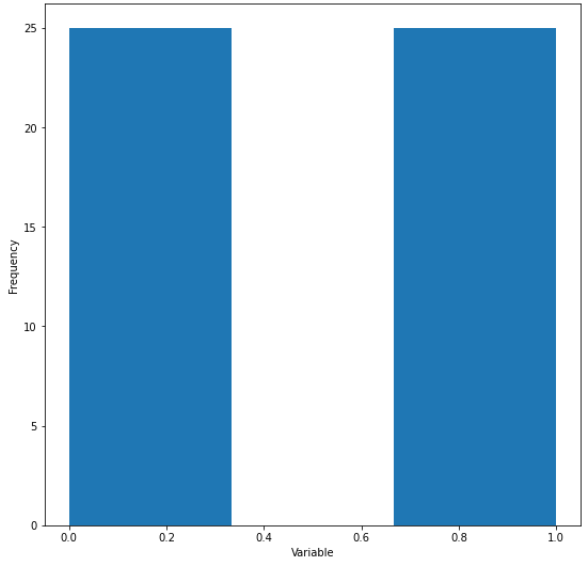


Calling Bullshit = Hoe te misleiden met grafieken en data

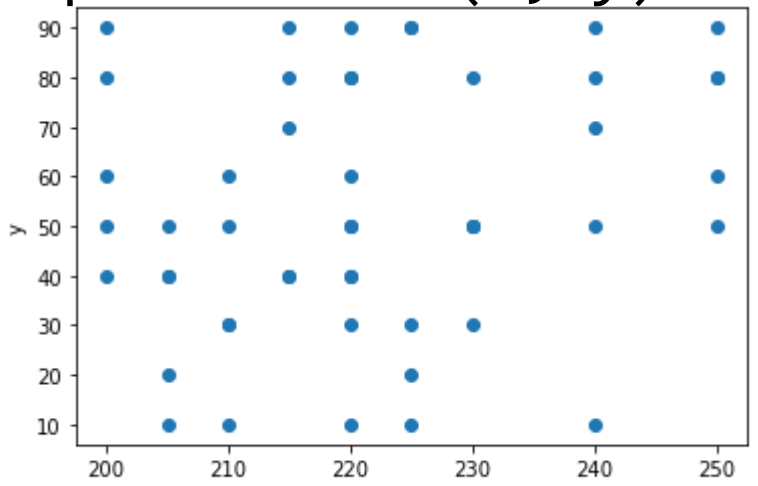
```
import matplotlib.pyplot as plt
```

Data types plotten

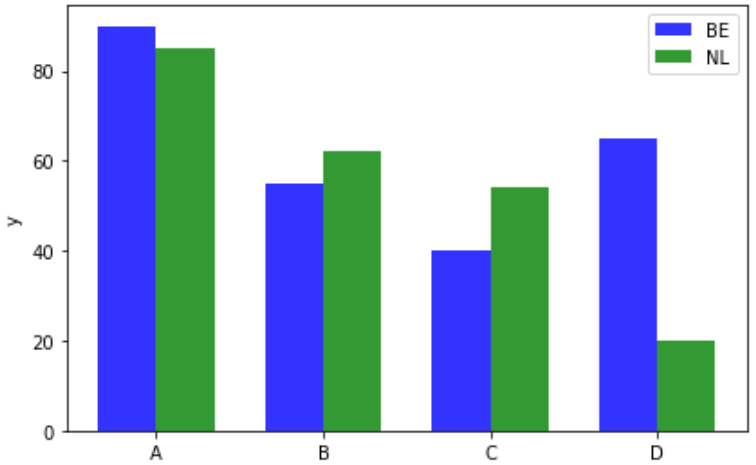
- `plt.hist(x)`



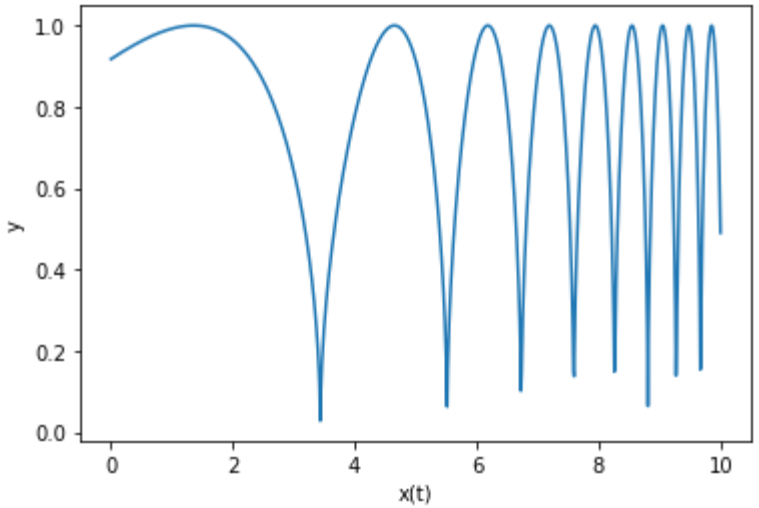
- `plt.scatter(x, y)`



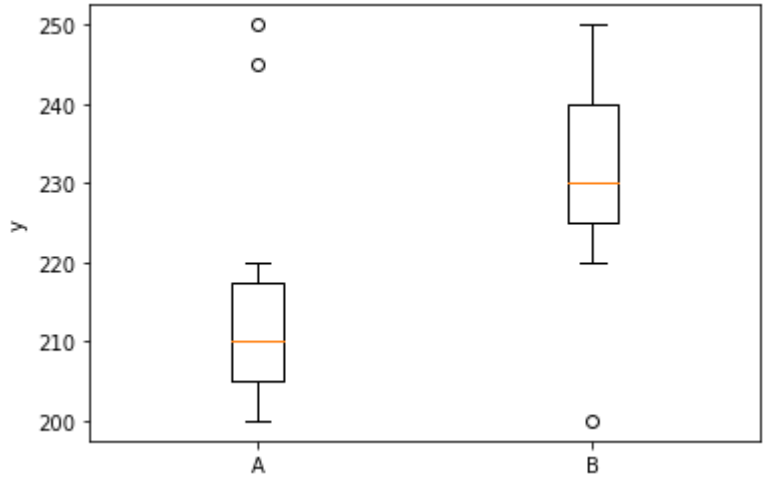
- `plt.bar(x, height)`



- `plt.plot(x, y)`



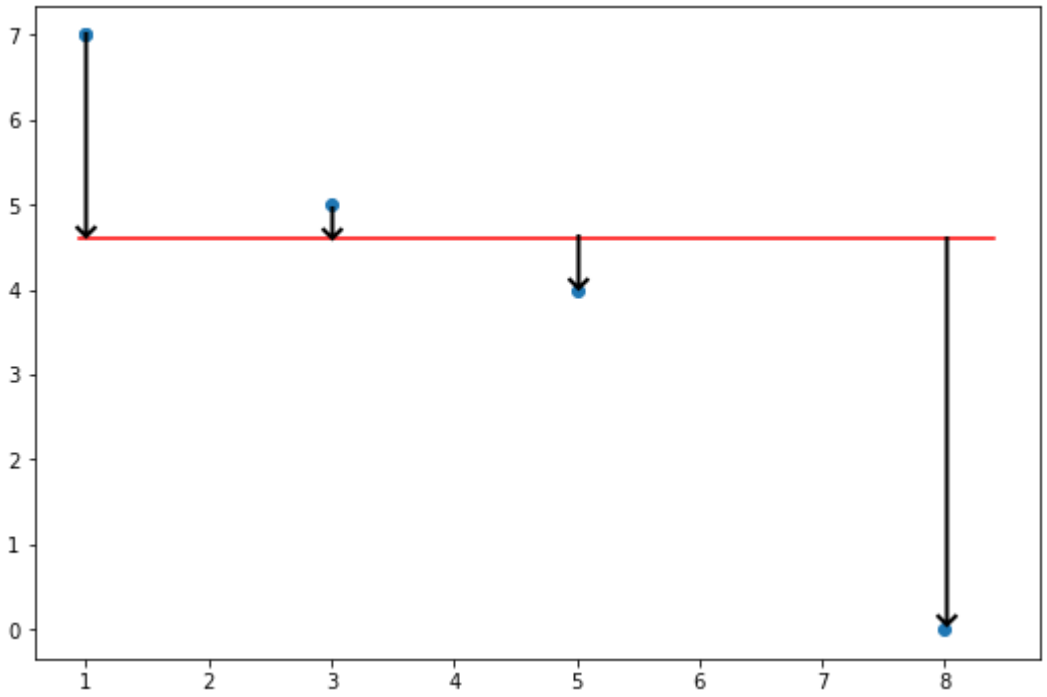
- `plt.boxplot(x)`



Calling Bullshit = Hoe te misleiden met grafieken en data

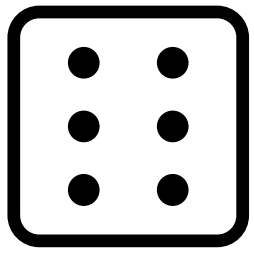
Statistieken

- Gemiddelde $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$
- Variantie
  Het kwadraat van de gemiddelde afstand van het gemiddelde
  $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$
- Standaarddeviatie
  De wortel van de variantie
  $\sigma = \sqrt{\sigma^2}$
- Modus
  De waarde die het meest voorkomt
- Mediaan
  De waarde die in het midden ligt
- 25%, 50%, 75% percentiel
  Onder waarde X heb je Y% van alle data



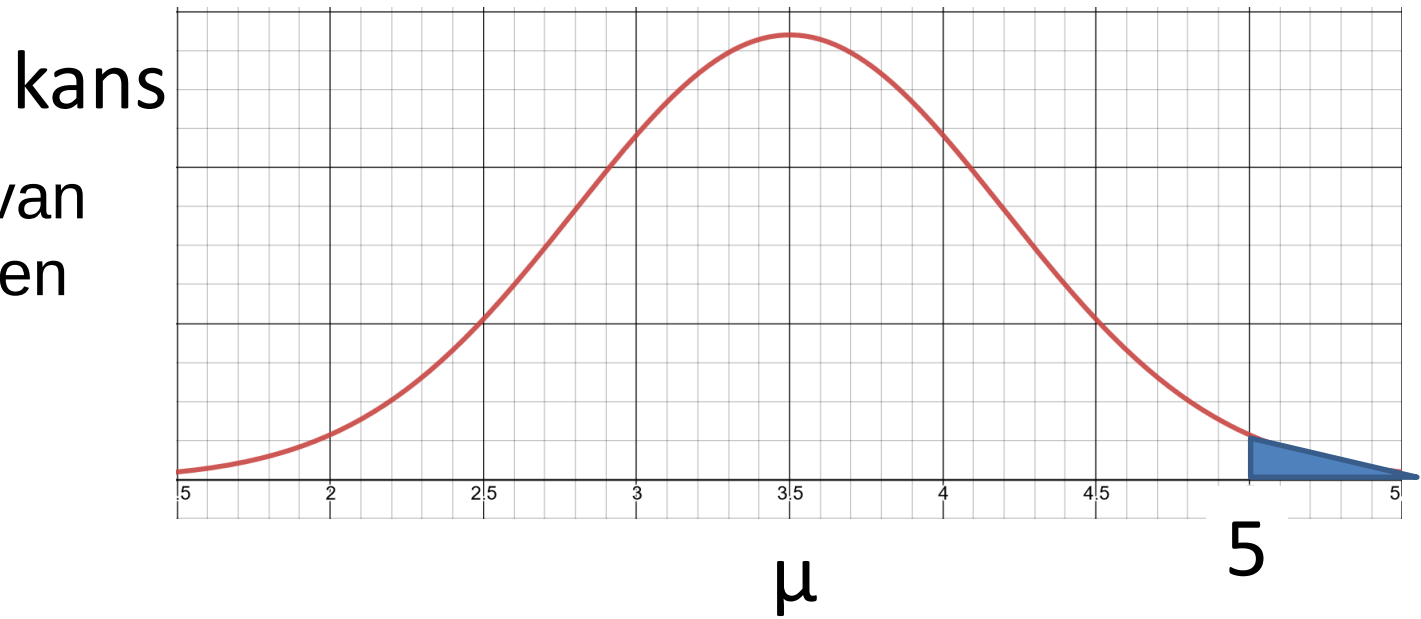
Hypothese toetsen

- Statistische significantie:
  De kans dat iets waar is, is groot
  OF
  De kans dat iets niet waar is, door toeval, is klein
- Hypothese toetsen
  Ik gooi 10 keer. Het gemiddelde van mijn worpen is 5. Is de dobbelsteen frauduleus?

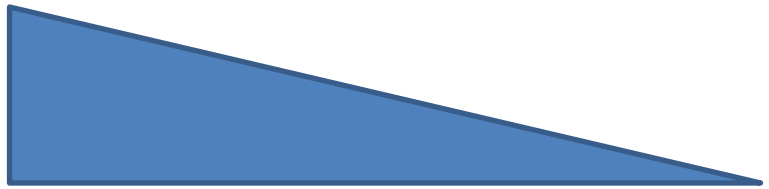


$$H_0 : \mu = 3.5$$ ← niet frauduleus

$$H_a : \mu > 3.5$$ ← wel frauduleus



Hypothese toetsen

OPP() = Probability (p-waarde)

p ≤ 0.05 → statistisch significant
(geen normale dobbelsteen)
p ≥ 0.05 → onvoldoende bewijs

Het verschil

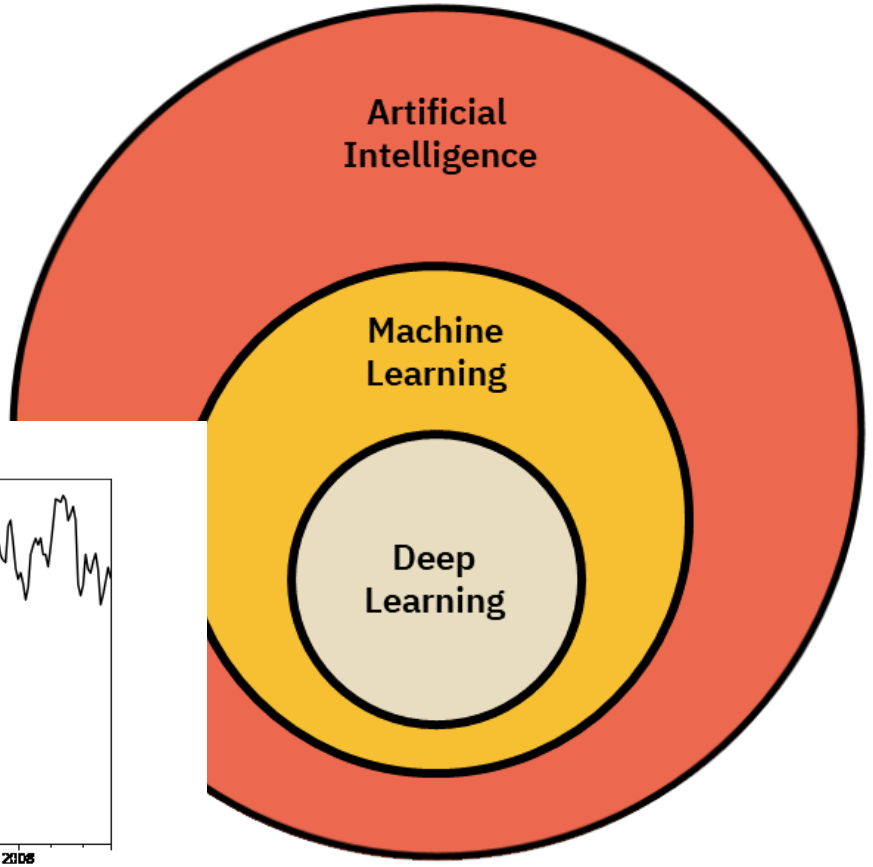
Artificial Intelligence

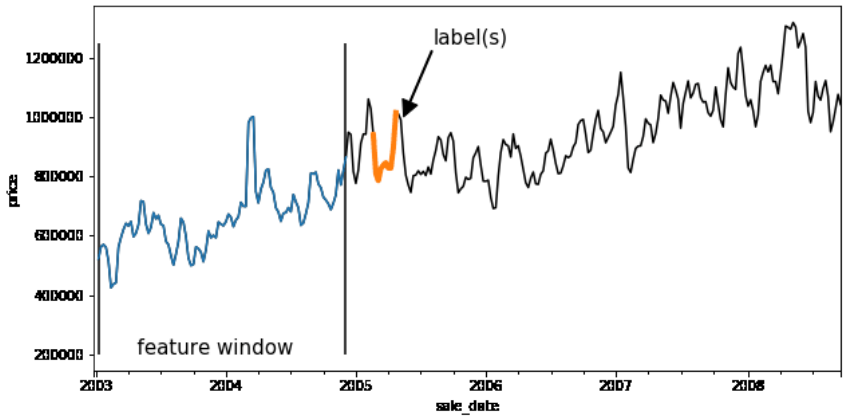
Namaken van intelligentie

Machine Learning

Namaken hoe we leren

Het verschil tussen AI en ML





https://www.youtube.com/watch?v=j4RWJTs0QCk&ab_channel=TOMRA

JE ZULT ME VEEL TERMEN HOREN GEBRUIKEN DIE EIGENLIJK VAN TOEPASSING ZIJN OP EEN MENS

Twee soorten machine learning

Supervised learning

Unsupervised learning

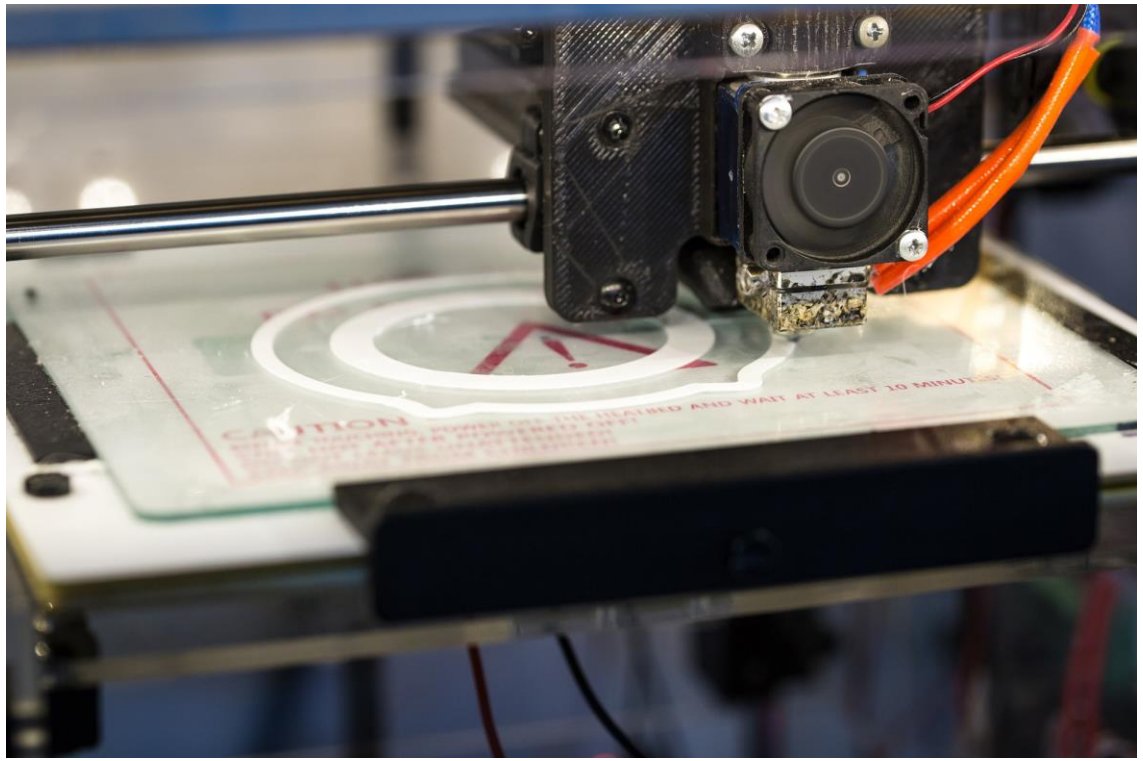
3 verschillende methoden

3 verschillende *classificatie* methoden[1]:

- Decision Tree/Random Forest
- Support Vector Machine
- Neural Networks

1: Er zijn meer methoden, maar deze komen het meest voor.

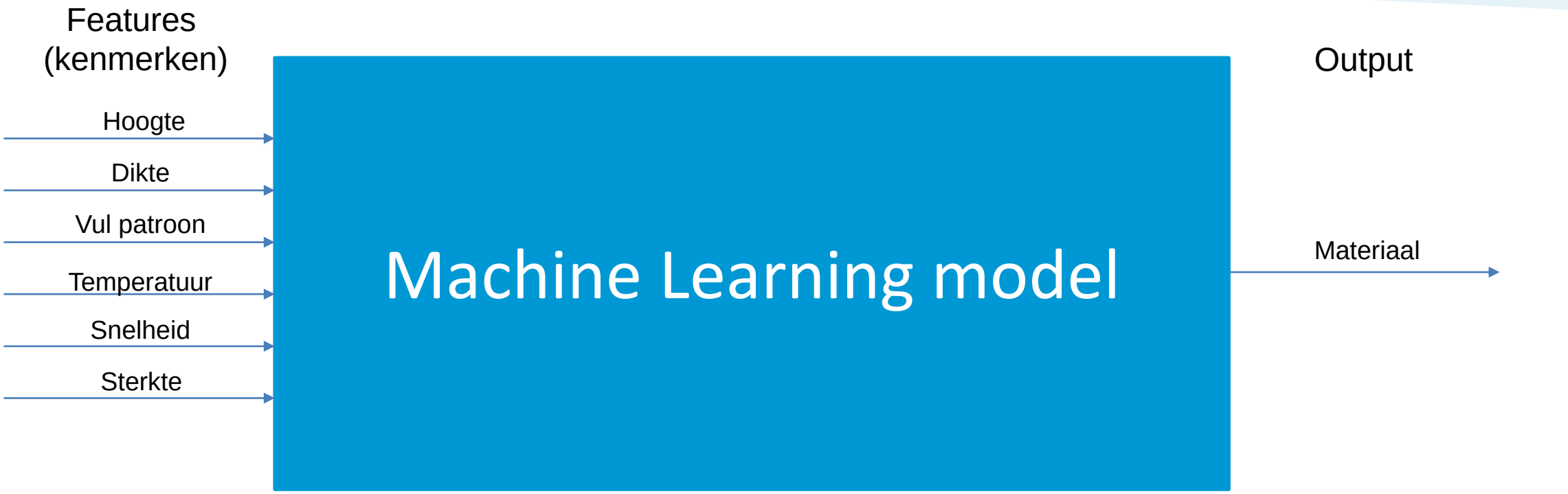
Voorbeeld casus



3D printer:

- https://www.kaggle.com/afumetto/3dprinter (staat ook op github)
- Eigenschappen van printopdracht:
  - Hoogte
  - Temperatuur
  - Snelheid
  - Etc.
- Kunnen we met Machine Learning raden met welk materiaal geprint is?

Het idee achter elk machine learning model



$$\mathbf{Y} = f(\mathbf{X})$$

3D printer dataset

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	fan_speed	roughness	tension_strenght	elongation
0	0.02	8	90	1	220	60	40	0	25	18	1.2
1	0.02	7	90	0	225	65	40	25	32	16	1.4
2	0.02	1	80	1	230	70	40	50	40	8	0.8
3	0.02	4	70	0	240	75	40	75	68	10	0.5
4	0.02	6	90	1	250	80	40	100	92	5	0.7

X

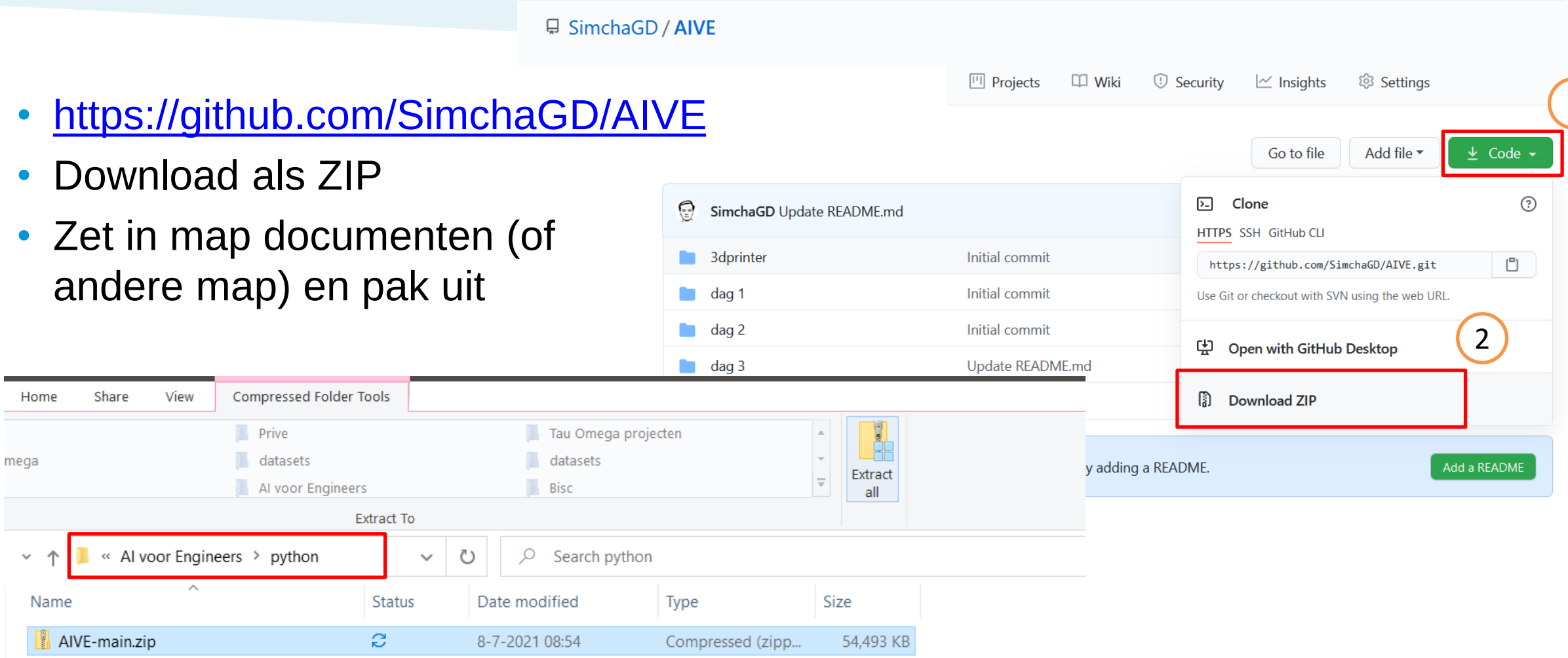
```
0    1
1    1
2    1
3    1
4    1
Name: material, dtype: int64
```

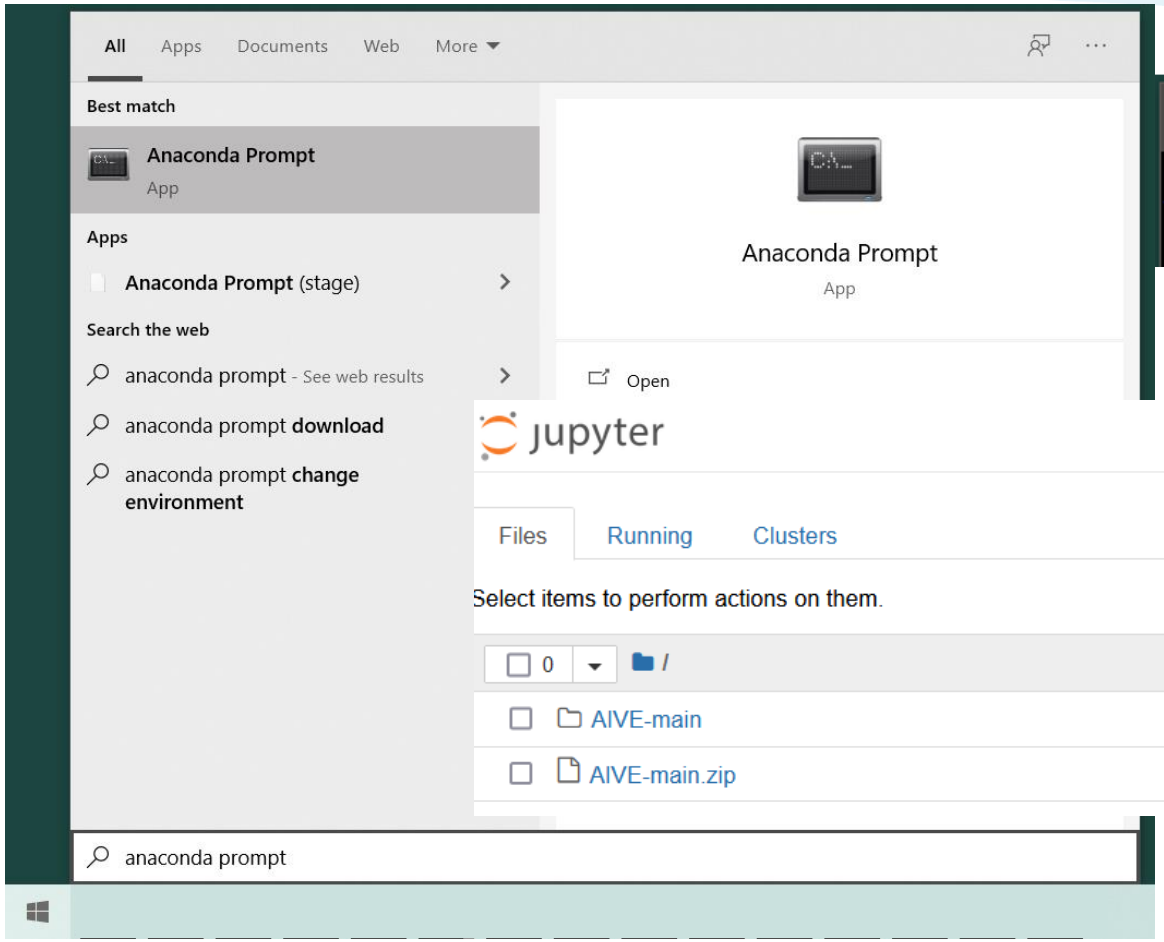
Y

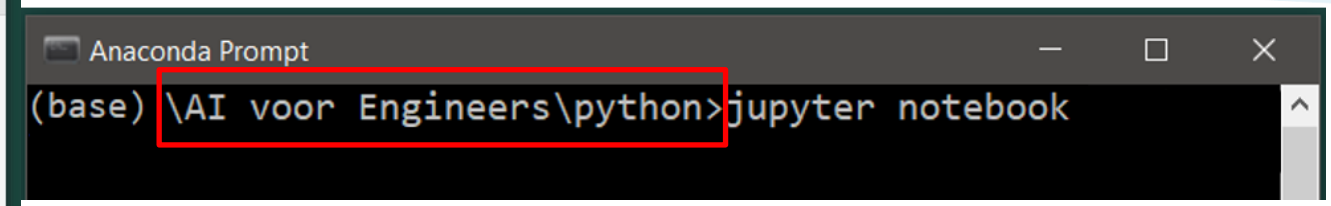
Kunnen we aan de hand van de laag hoogte, print dikte etc. voorspellen met welk materiaal is geprint?

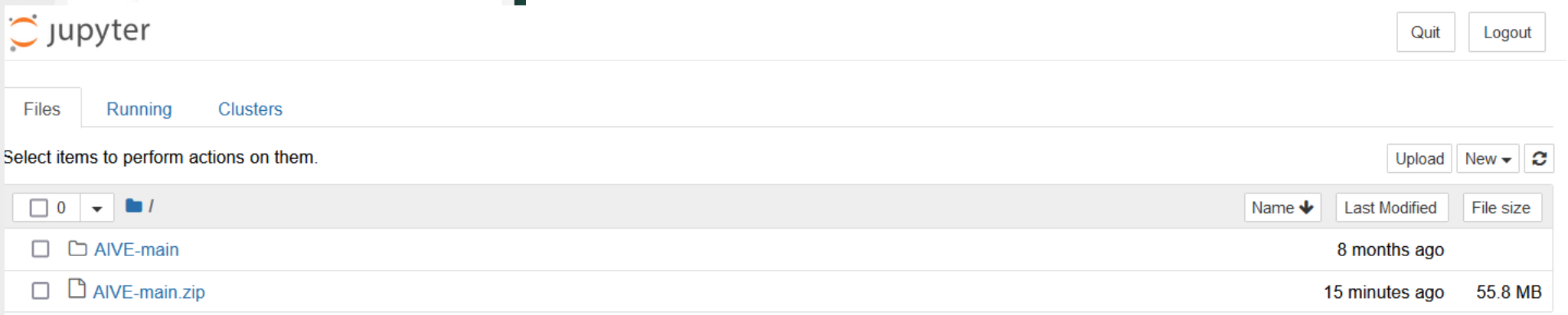
Materiaal
1 = ABS
0 = PLA

- https://github.com/SimchaGD/AIVE
- Download als ZIP
- Zet in map documenten (of andere map) en pak uit









mikrocentrum
opleiden • ontmoeten • ondernemen

Hoe laden we data in?

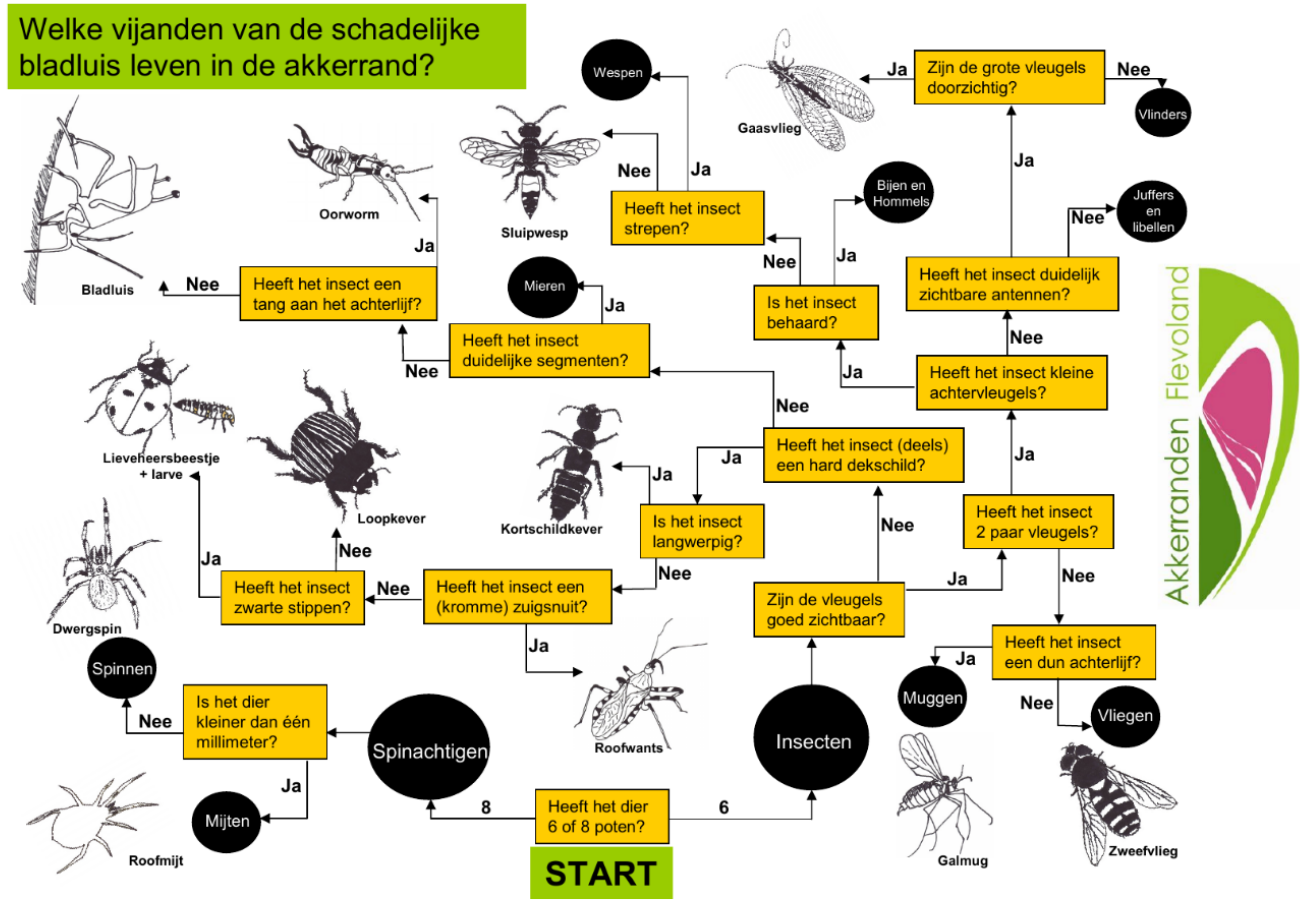
```
import pandas as pd
filename = "3dprinter/data.csv"
df = pd.read_csv(filename, sep = ';')
df.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	fan_speed	roughness	tension_strenght	elongation
0	0.02	8	90	1	220	60	40	0	25	18	1.2
1	0.02	7	90	0	225	65	40	25	32	16	1.4
2	0.02	1	80	1	230	70	40	50	40	8	0.8
3	0.02	4	70	0	240	75	40	75	68	10	0.5
4	0.02	6	90	1	250	80	40	100	92	5	0.7

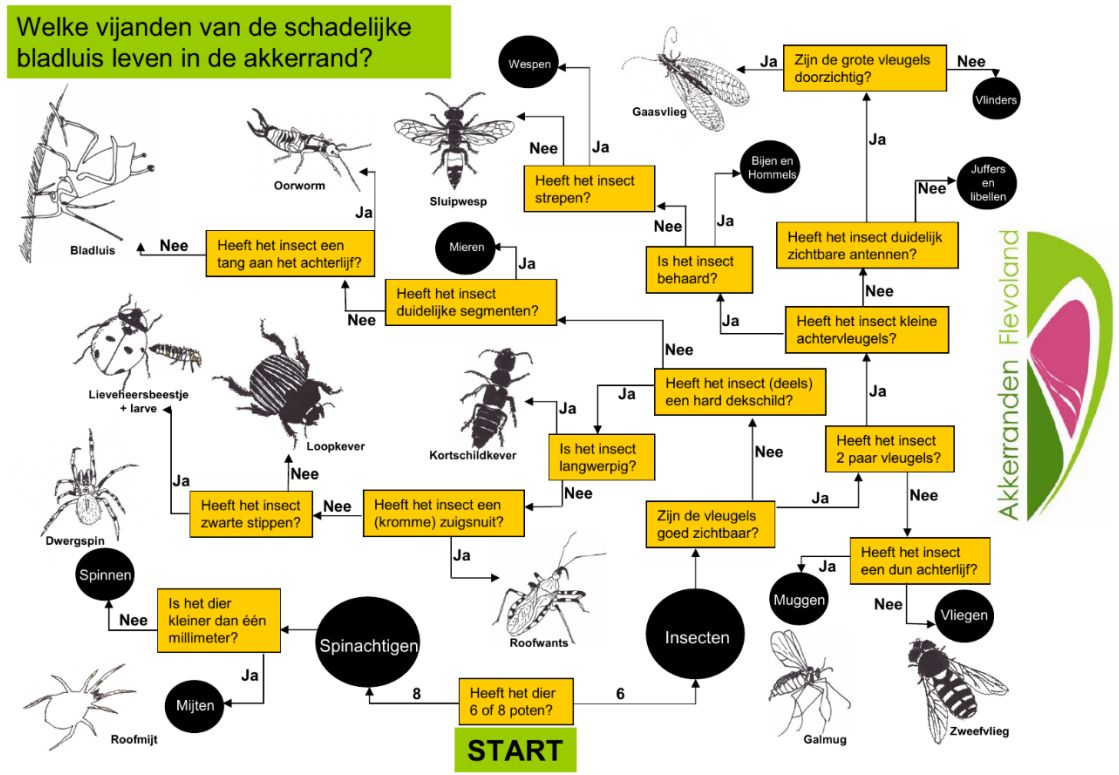
0 = grid
1 = honeycomb/hexagon

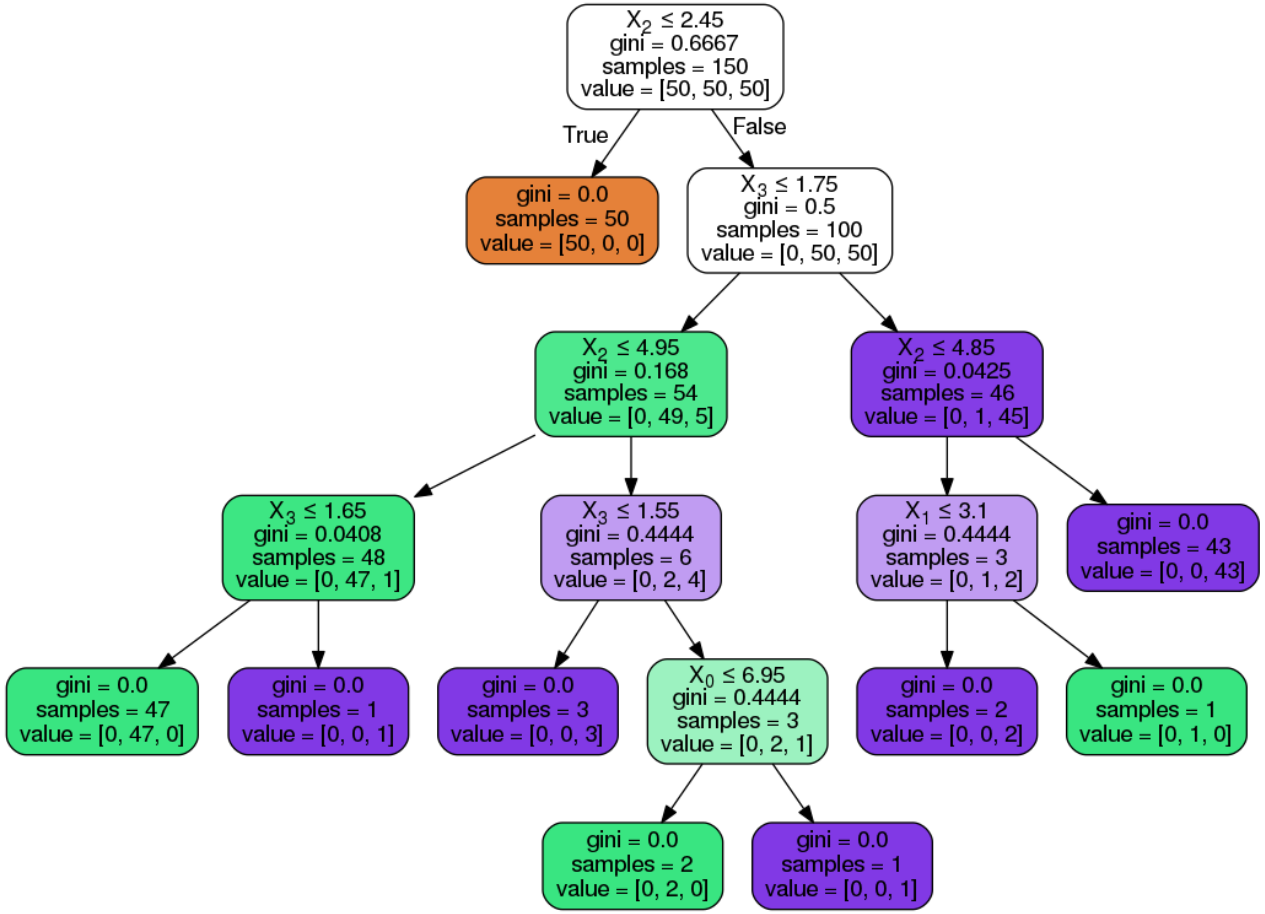
Model 1:

DECISION TREES

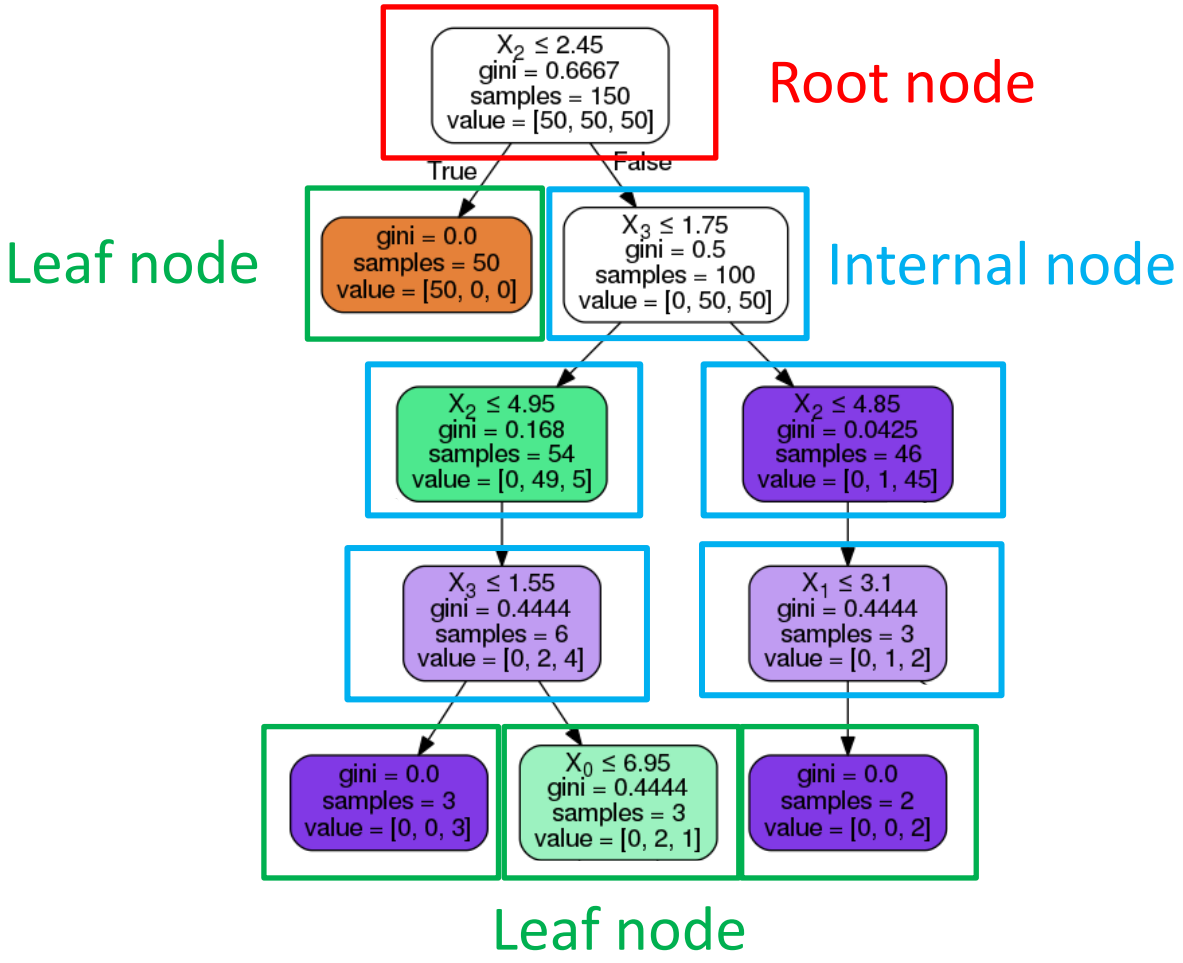


Decision Trees





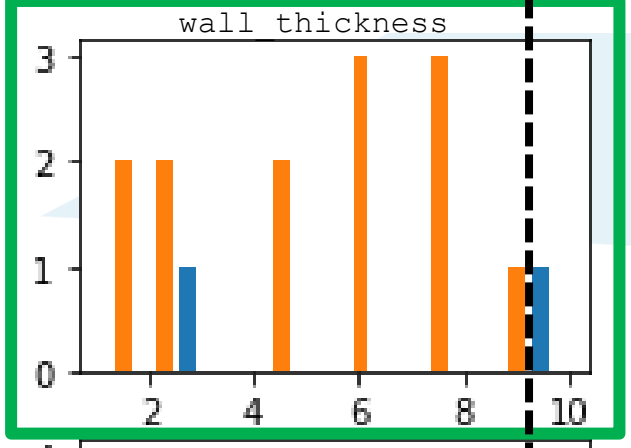
Terminologie

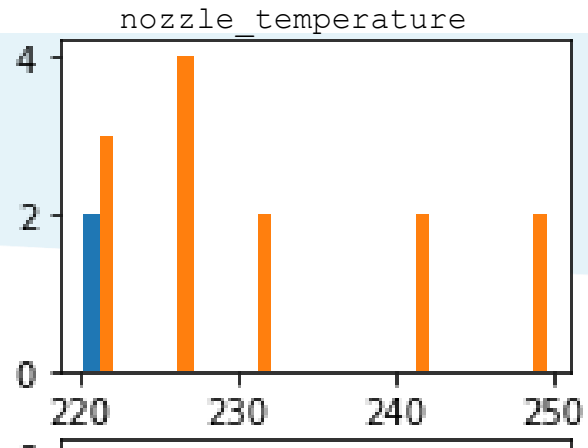


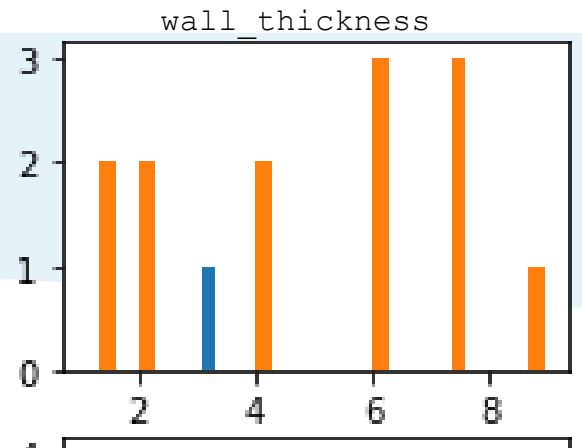
Met de hand

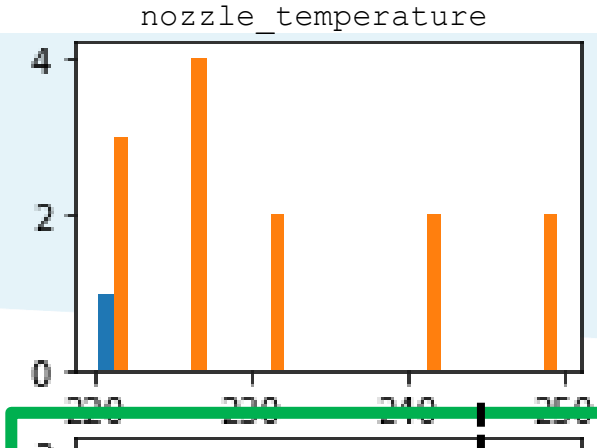
PLA

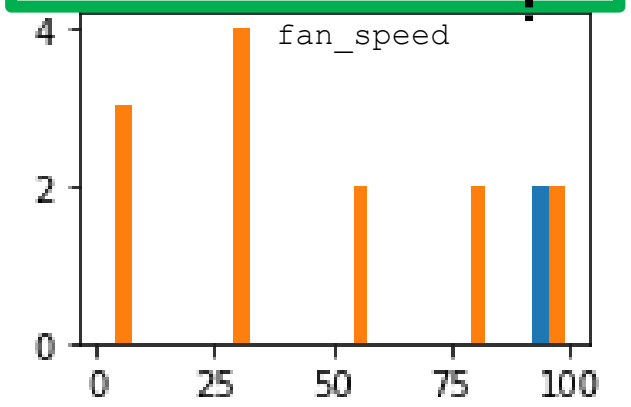
ABS

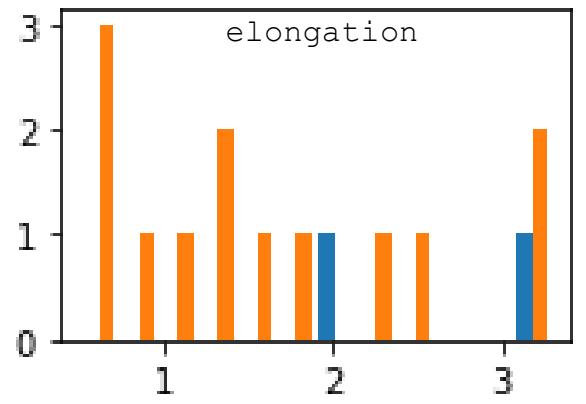

wall_thickness

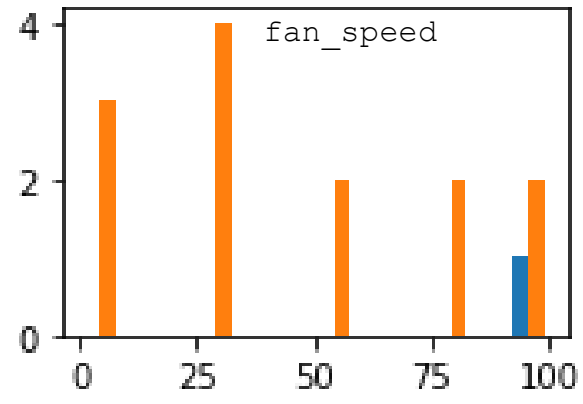

nozzle_temperature

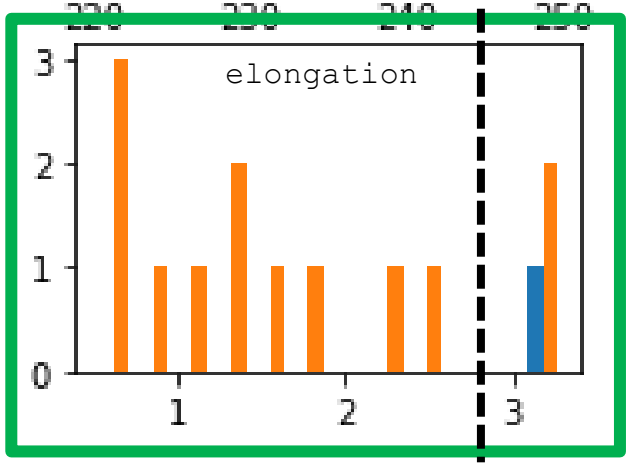

wall_thickness

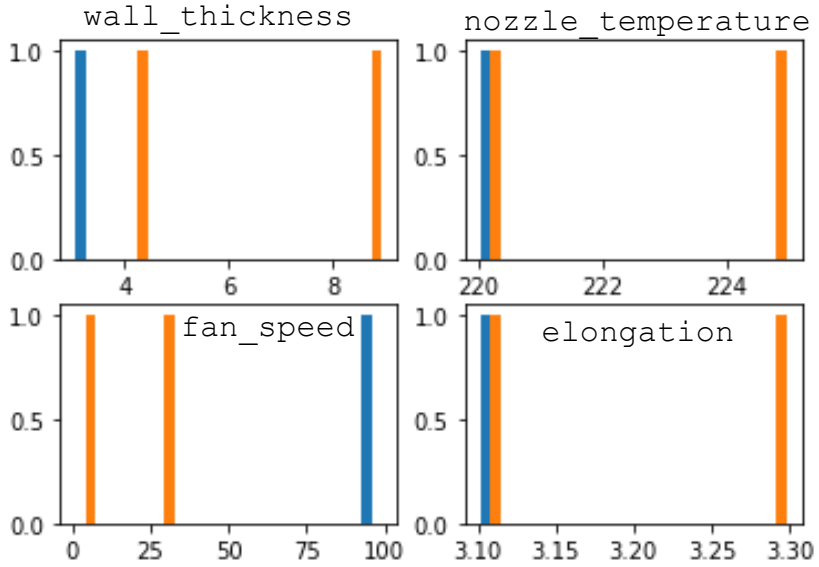

nozzle_temperature

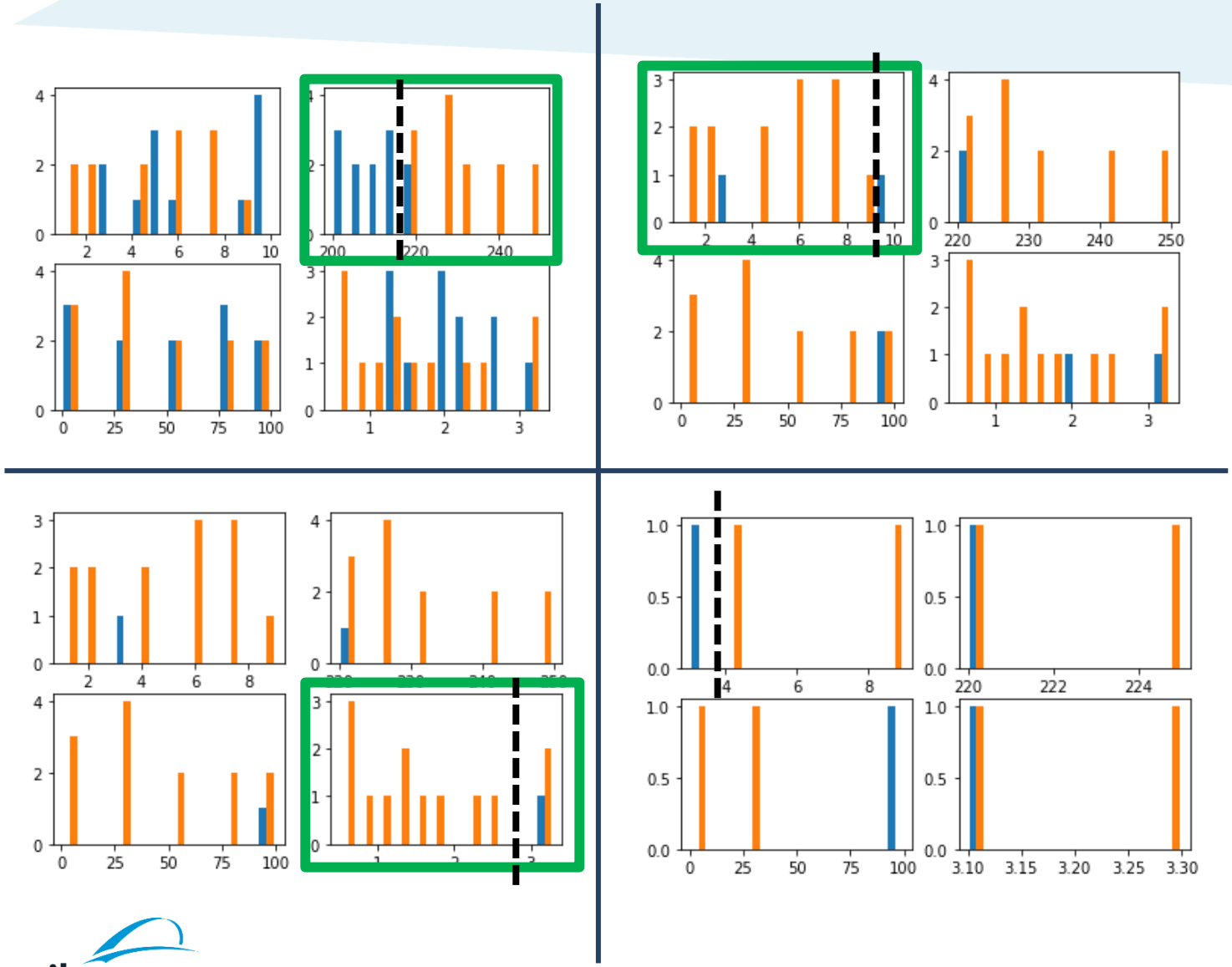

fan_speed

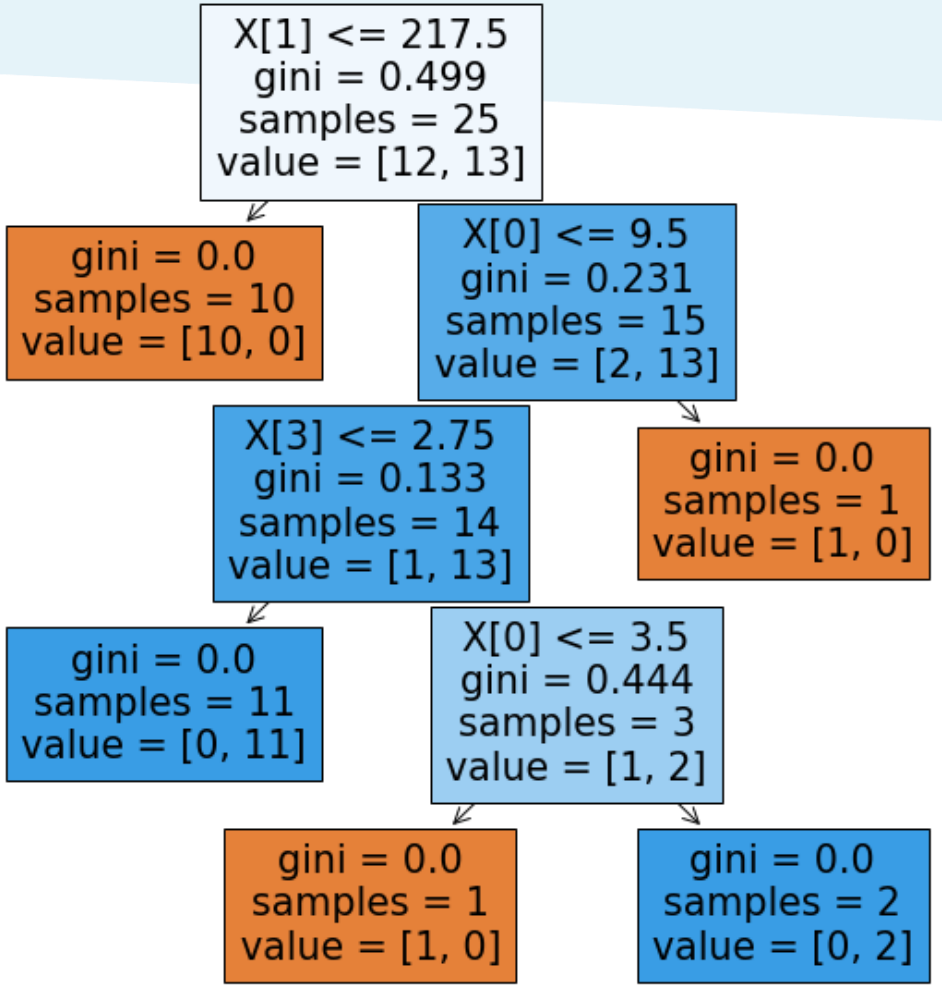

elongation

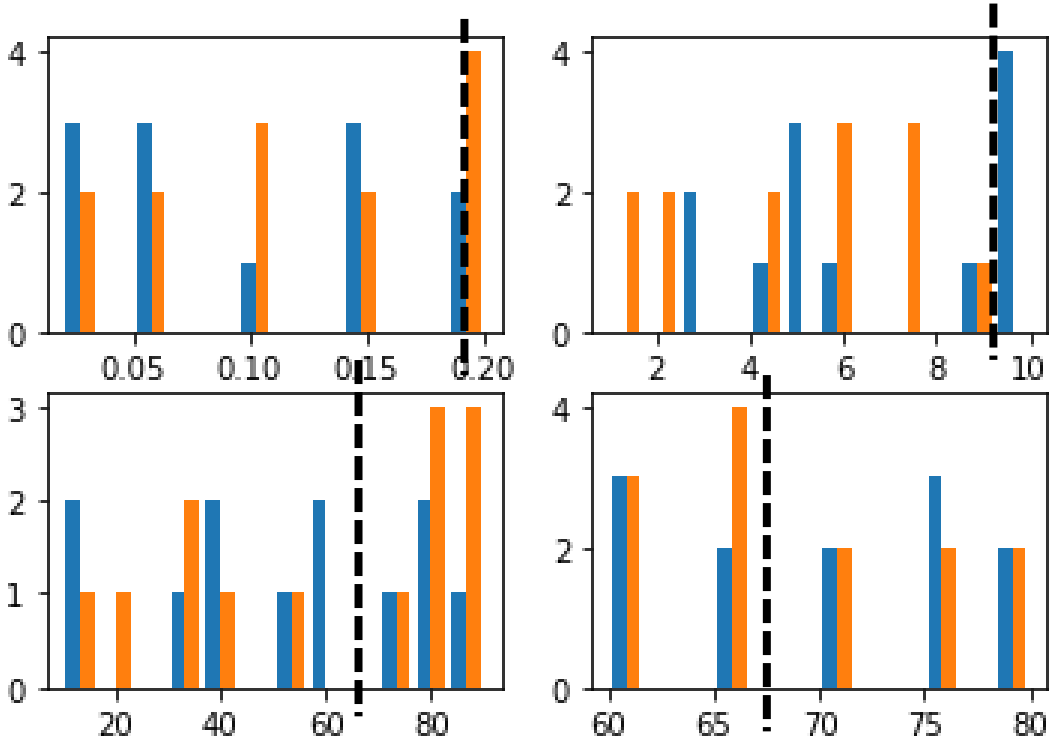

fan_speed


elongation


wall_thickness
nozzle_temperature
fan_speed
elongation



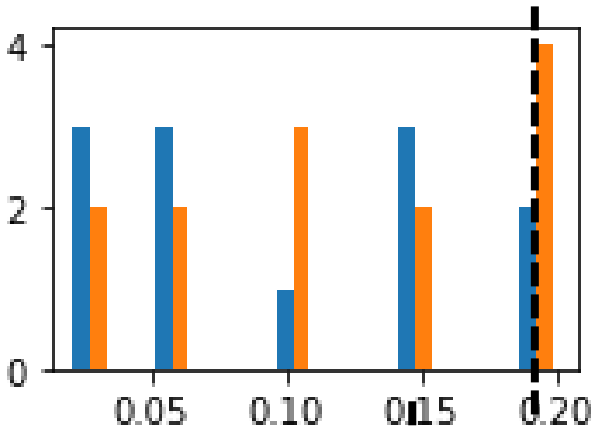

X[1] <= 217.5
gini = 0.499
samples = 25
value = [12, 13]
gini = 0.0
samples = 10
value = [10, 0]
X[0] <= 9.5
gini = 0.231
samples = 15
value = [2, 13]
X[3] <= 2.75
gini = 0.133
samples = 14
value = [1, 13]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 11
value = [0, 11]
X[0] <= 3.5
gini = 0.444
samples = 3
value = [1, 2]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 2
value = [0, 2]

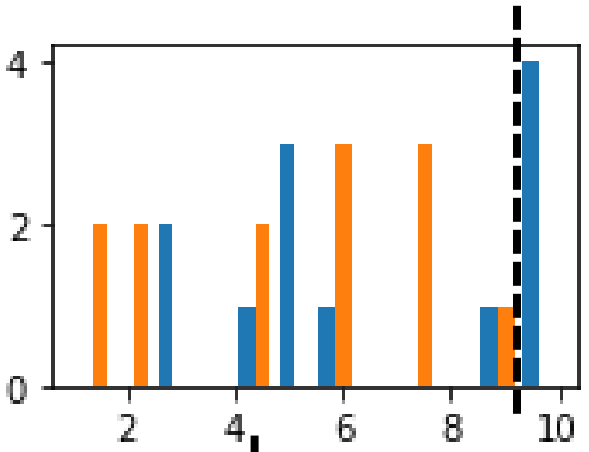


? $\text{Gini} = 1 - \sum_{i=1}^{n} p_i^2$

> 0.19

ABS	PLA	ABS	PLA
12	9	0	4



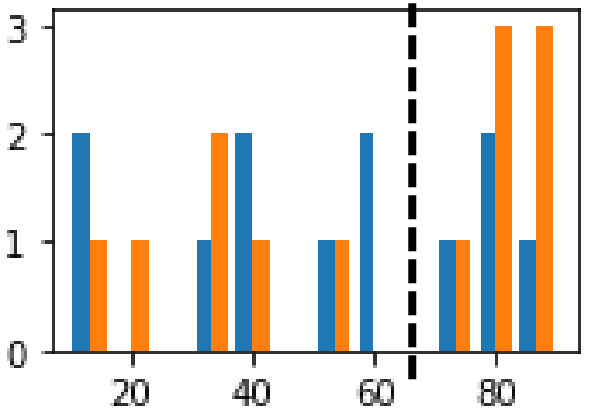


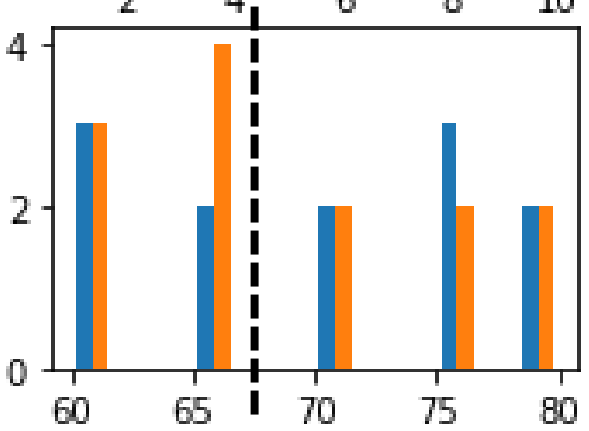
> 9.5

ABS	PLA	ABS	PLA
8	13	4	0

> 65

ABS	PLA	ABS	PLA
8	8	4	7





> 67.5

ABS	PLA	ABS	PLA
5	7	7	6

mikrocentrum
opleiden • ontmoeten • ondernemen

$$\text{Gini} = 1 - \sum_{i=1}^{n} p_i^2 = 1 - P(\text{ABS})^2 - P(\text{PLA})^2$$

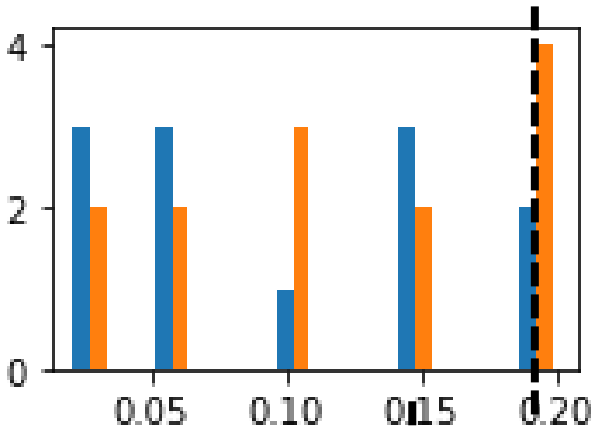
$$1 - \left(\frac{12}{12+9}\right)^2 - \left(\frac{9}{12+9}\right)^2 = 0.490 \qquad 1 - \left(\frac{4}{4+0}\right)^2 - \left(\frac{0}{4+0}\right)^2 = 0$$

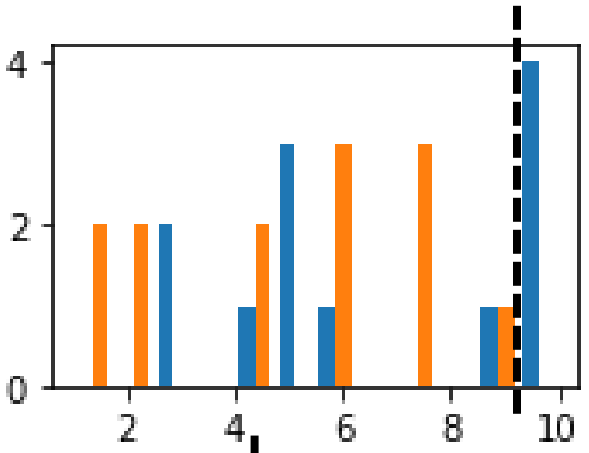
$$\left(\frac{21}{25}\right) \times 0.490 + \left(\frac{4}{25}\right) \times 0 = 0.412$$

> 0.19

ABS	PLA	ABS	PLA
12	9	0	4

$$\text{Gini} = 0.412$$





> 9.5

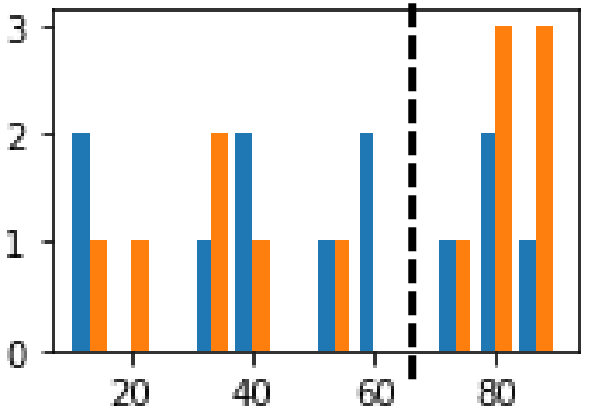
ABS	PLA	ABS	PLA
8	13	4	0

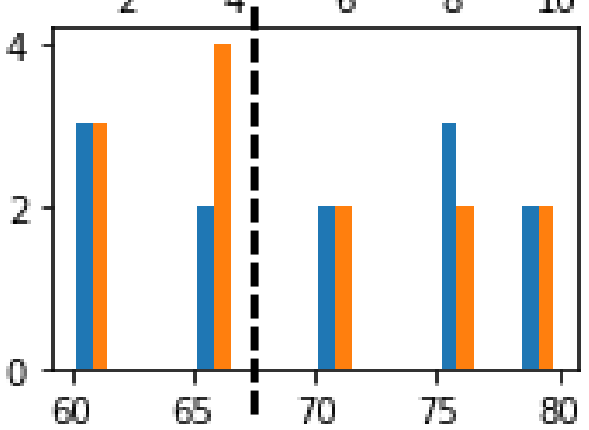
$$\text{Gini} = 0.396$$

> 65

ABS	PLA	ABS	PLA
8	6	4	7

$$\text{Gini} = 0.478$$





> 67.5

ABS	PLA	ABS	PLA
5	7	7	6

$$\text{Gini} = 0.492$$

mikrocentrum
opleiden • ontmoeten • ondernemen

```
In:   from sklearn.tree import DecisionTreeClassifier
      from sklearn.metrics import accuracy_score

      clf = DecisionTreeClassifier()
      y_pred = clf.fit(X, Y).predict(X)
      print(accuracy_score(y_pred, Y))
Out: 1.0
```

mikrocentrum
opleiden • ontmoeten • ondernemen

Oefening

Laad de 3D-printer data in met Pandas

Splits de voorspellers van de afhankelende variabelen.

Maak een Decision Tree met behulp van de SciKit-Learn library

~10 minuten

Hyperparameters

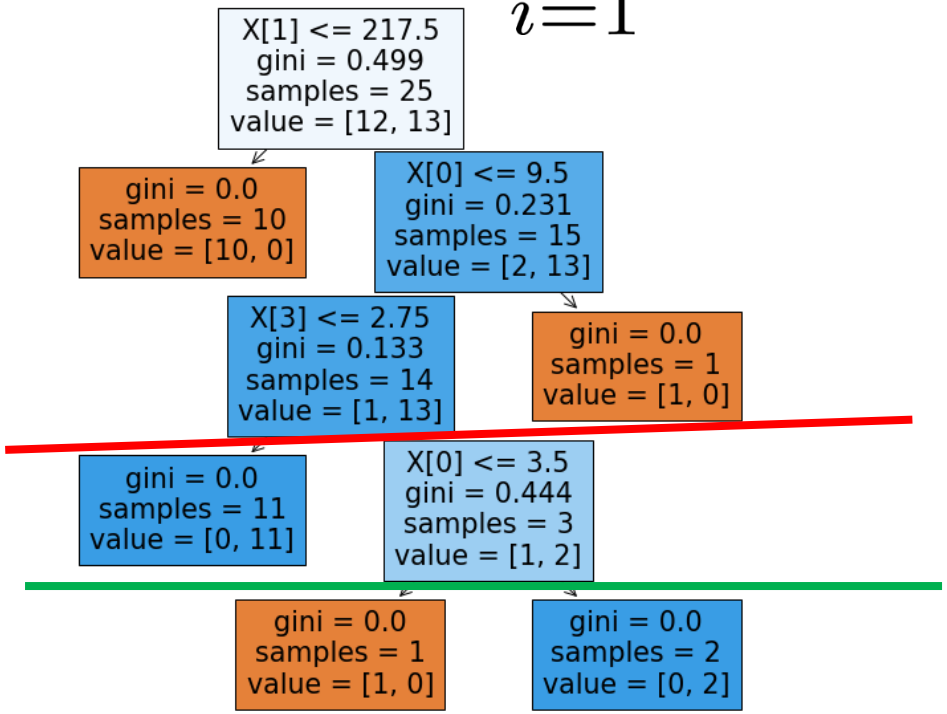
- criterion = ("gini", "entropy")

$$\text{Gini} = 1 - \sum_{i=1}^{n} p_i^2$$

$$\text{Entropy} = -\sum_{i=1}^{n} p_i \log(p_i)$$

- max_depth = None
  max_depth = 2

- min_samples_split = 2
  min_samples_split = 6



```
In: dt_clf = DecisionTreeClassifier(max_depth=4)
    y_pred = dt_clf.fit(x_train, y_train).predict(x_test)
    print(accuracy_score(y_pred, y_test))


Out: 0.8
```

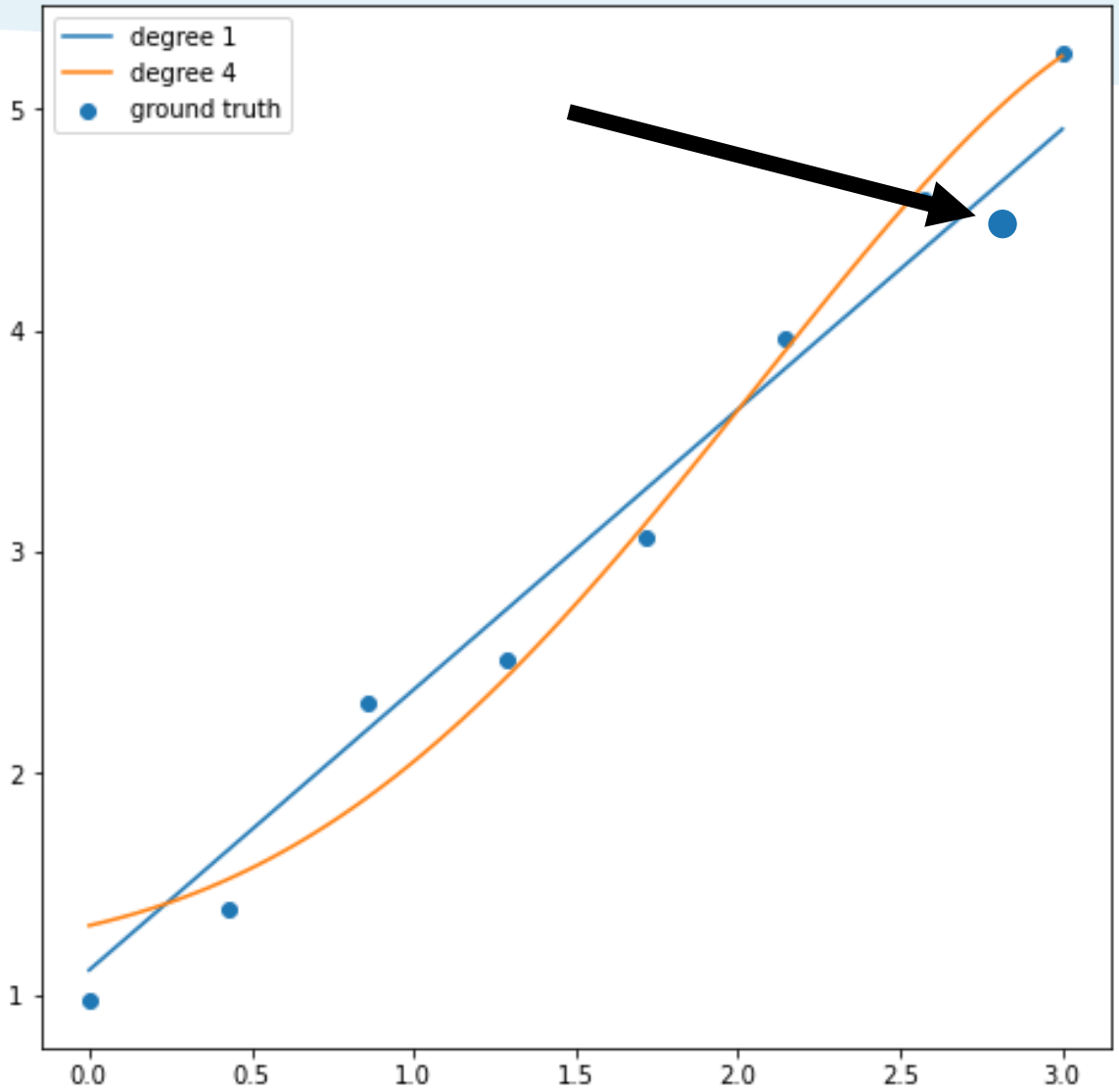
Train test split

Overfitting:

Creating over-complex models
that do not generalize well to new data

Het creëren van over-complexe modellen
die niet generaliseerbaar zijn op nieuwe data

https://en.wikipedia.org/wiki/Gambler%27s_fallacy



Train test split

Voorbeeld

Hele dataset

1
2
3
4
5
6
7
8
9
10

Train test split

Verhouding 70/30

Trainset

3
5
2
8
6
10
9

Test

4
1
7

Train test split

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X, Y,
                                                    test_size = 0.5,
                                                    random_state = 2)

clf = DecisionTreeClassifier()
y_pred = clf.fit(x_train, y_train).predict(x_test)
print(accuracy_score(y_test, y_pred))
```

Decision Trees

Voordelen

- Makkelijk/Simpel
- Goede resultaten met relatief weinig data
- Snel
- Goed met *numerieke en categorische data*
- *Glass box*

Nadelen

- *Niet robuust*
- Erg gevoelig voor overfitting
- Optimale oplossing niet altijd gegarandeerd
- Kan *biased* uitkomsten produceren

Oefening met een dataset

Maak een *decision tree classifier* die voorspelt met welk materiaal is geprint

Split je dataset in een *train-* en *testset*

Bereken de *accuracy*

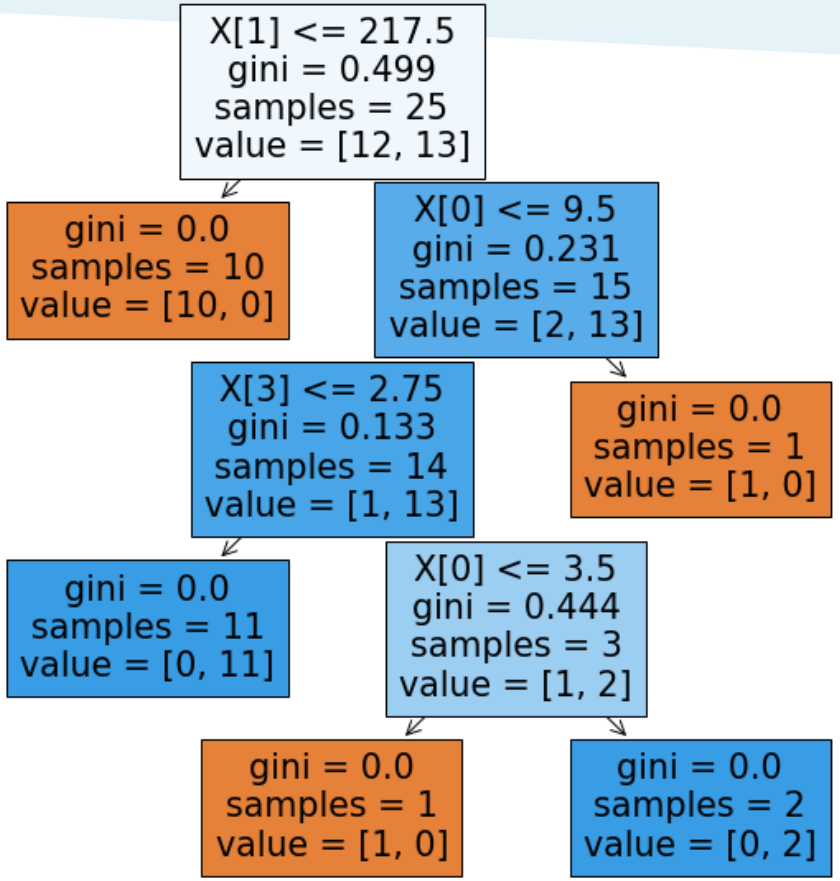
~ 10 minuten

Model opslaan en inladen

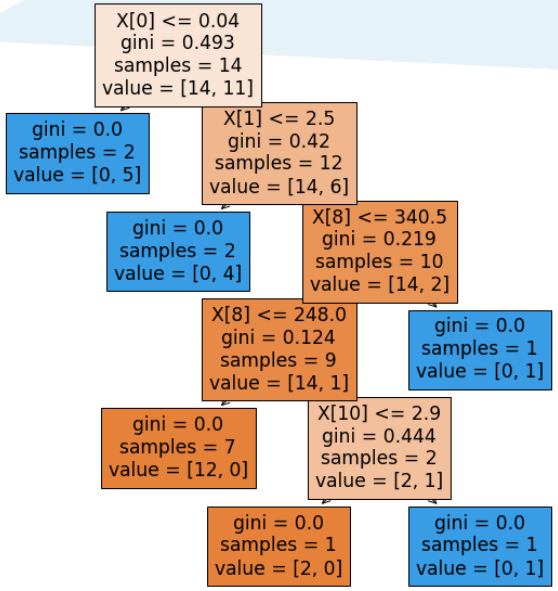
```
import pickle
s = pickle.dumps(clf)
with open("decision_tree.sav", "wb") as f:
    f.write(s)

import pickle
with open("decision_tree.sav", "rb") as f:
    clf = pickle.load(f)
```

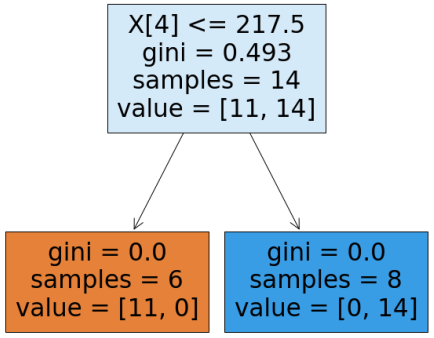
Model 1b:

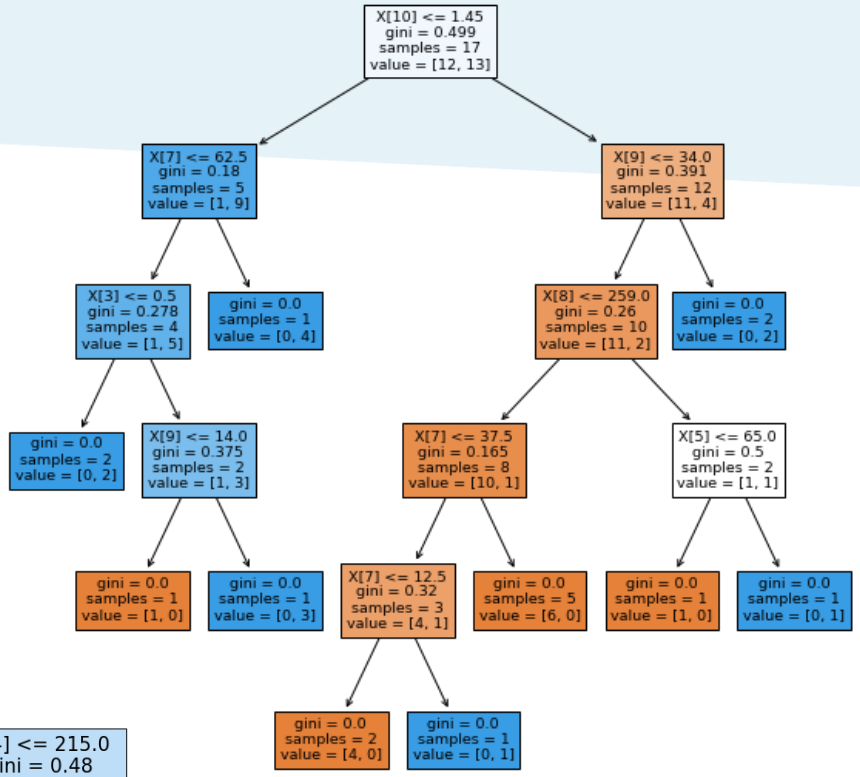
RANDOM FOREST

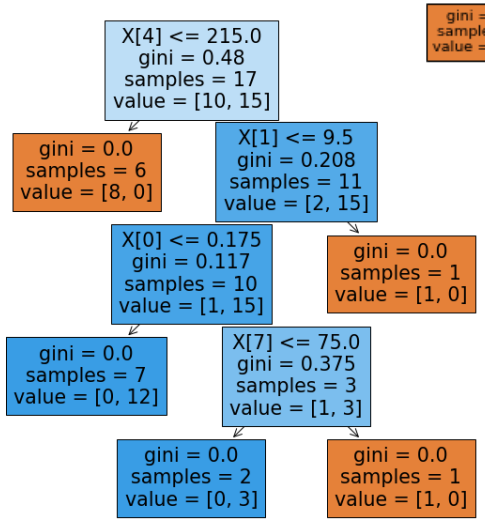


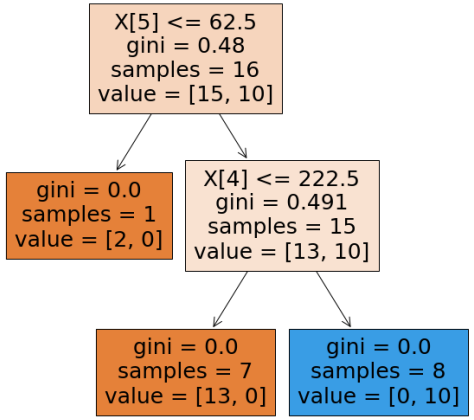
Random Forest

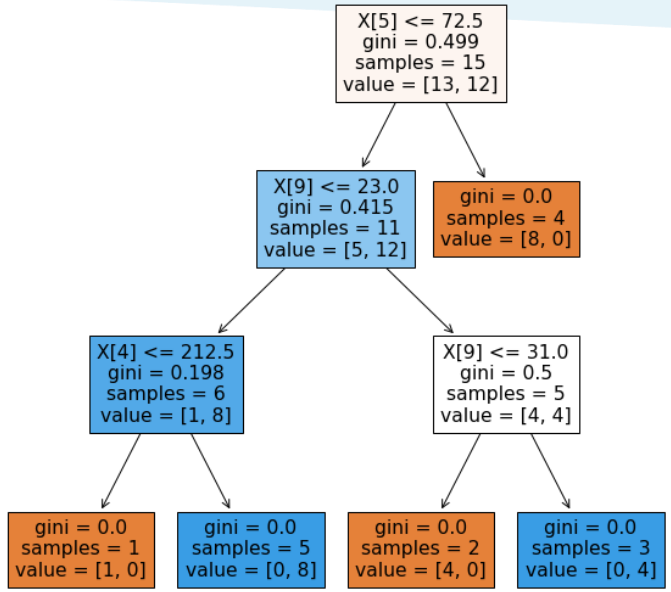


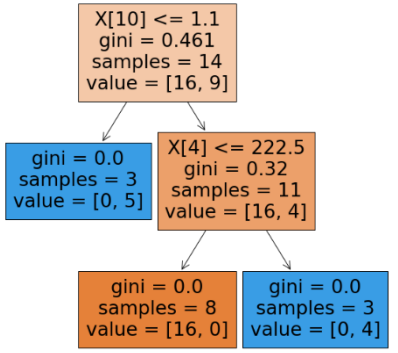












mikrocentrum
opleiden • ontmoeten • ondernemen

Stappenplan

1. Maak een ‘bootstrapped’ dataset
2. Maak een decision tree van deze dataset (met een paar kolommen)
3. Herhaal 1 en 2
4. Alle decision trees bij elkaar representeren de Random Forest

Stap 1. bootstrapped dataset

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	material	fan_speed	roughness	tension_strenght
0	0.02	8	90	0	220	60	40	0	0	25	18
1	0.02	7	90	1	225	65	40	0	25	32	16
2	0.02	1	80	0	230	70	40	0	50	40	8
3	0.02	4	70	1	240	75	40	0	75	68	10
4	0.02	6	90	0	250	80	40	0	100	92	5
					.						
					.						
					.						
45	0.2	5	60	1	200	60	40	1	0	321	28
46	0.2	4	20	0	205	65	40	1	25	265	14
47	0.2	5	60	1	210	70	40	1	50	278	30
48	0.2	7	40	0	215	75	40	1	75	244	29
49	0.2	3	60	1	220	80	40	1	100	220	27

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	fan_speed	roughness	tension_strenght	elongati
48	0.20	7	40	0	215	75	40	75	244	29	3
29	0.10	1	30	1	220	80	120	100	121	14	1
28	0.10	4	90	0	215	75	120	75	138	34	2
1	0.02	7	90	1	225	65	40	25	32	16	1
1	0.02	7	90	1	225	65	40	25	32	16	1
22	0.10	1	50	0	230	70	120	50	265	10	0
45	0.20	5	60	1	200	60	40	0	321	28	2
36	0.15	4	50	0	205	65	60	25	212	24	1
44	0.20	3	80	0	250	80	40	100	357	21	1
20	0.10	1	40	0	220	60	120	0	120	16	1

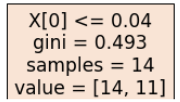
bootstrapped

Stap 2. decision tree maken

Bij elke stap selecteren we X random kolommen waarmee we een decision tree gaan vullen

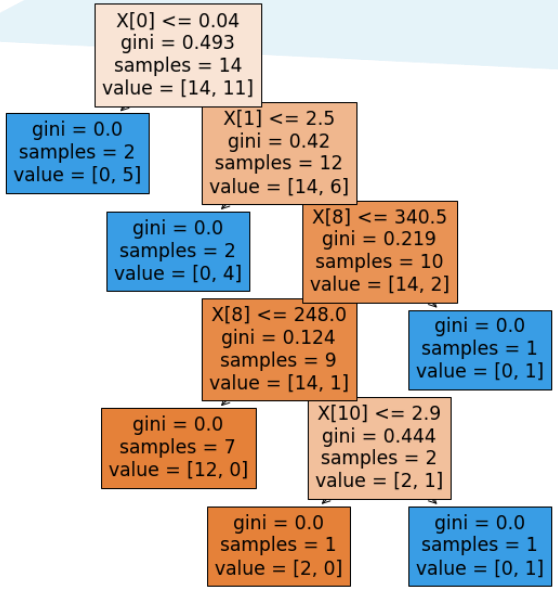
Stel, we selecteren elke keer 2 kolommen

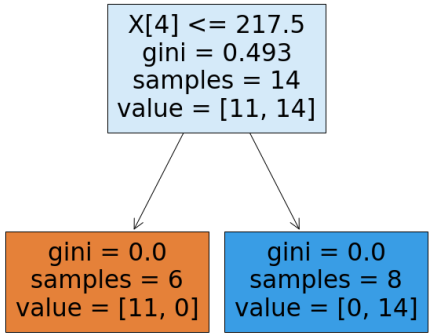
	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	fan_speed	roughness	tension_strenght	elongatic
48	0.20	7	40	0	215	75	40	75	244	29	3
29	0.10	1	30	1	220	80	120	100	121	14	1
28	0.10	4	90	0	215	75	120	75	138	34	2
1	0.02	7	90	1	225	65	40	25	32	16	1
1	0.02	7	90	1	225	65	40	25	32	16	1
22	0.10	1	50	0	230	70	120	50	265	10	0
45	0.20	5	60	1	200	60	40	0	321	28	2
36	0.15	4	50	0	205	65	60	25	212	24	1
44	0.20	3	80	0	250	80	40	100	357	21	1
20	0.10	1	40	0	220	60	120	0	120	16	1

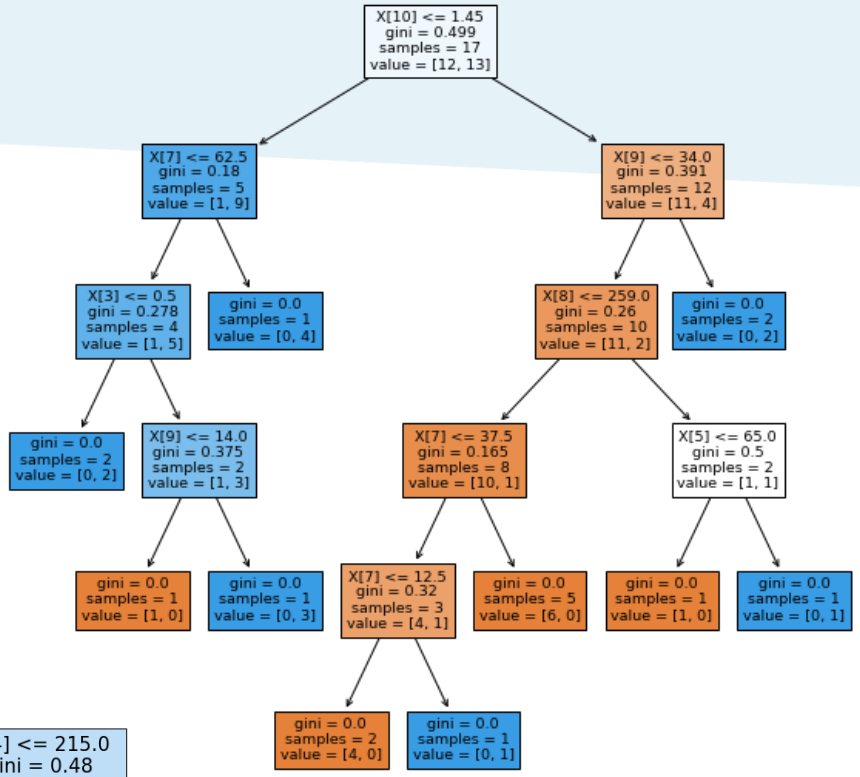


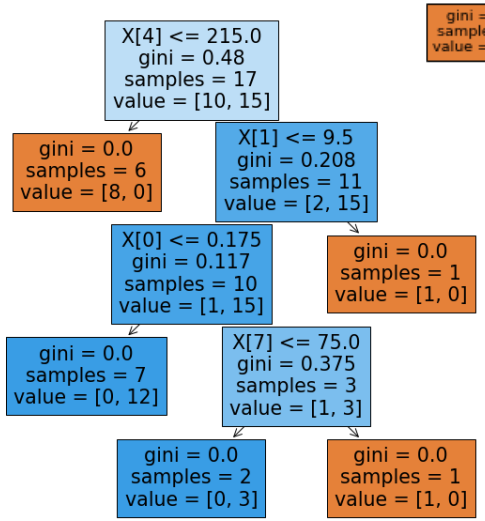
We zorgen hiermee voor grote variatie tussen alle decision trees. Dit maakt de classifier robuster.

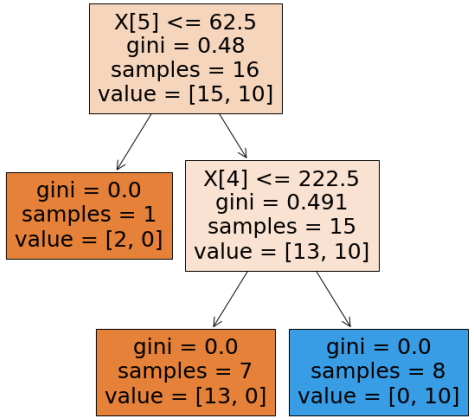
Herhaal stap 1 en 2

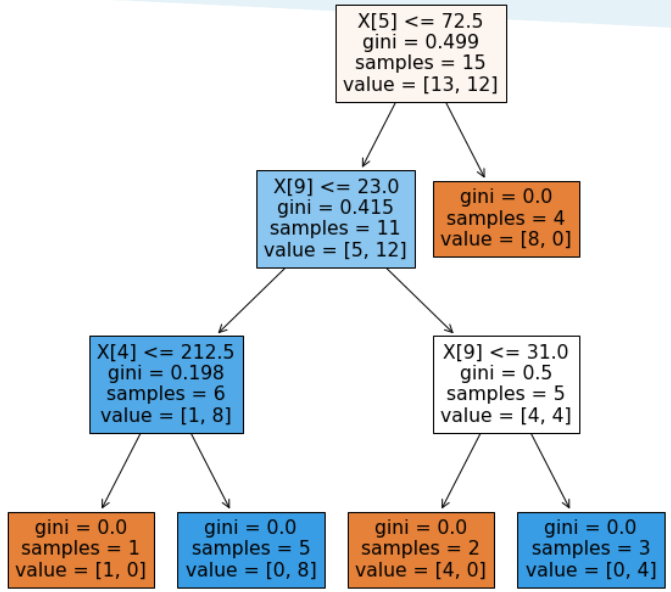


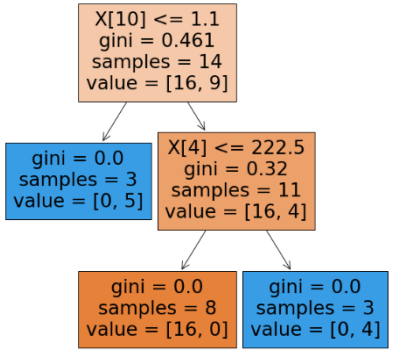






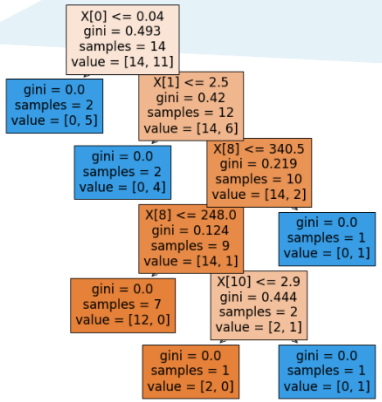




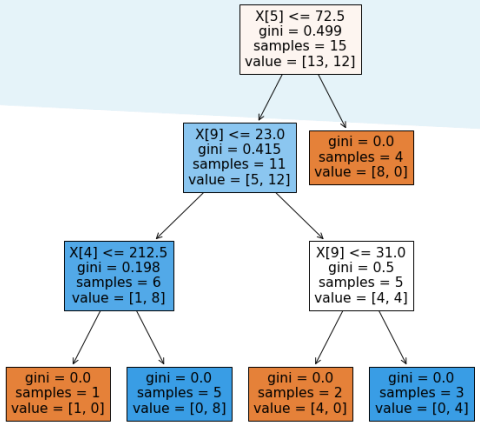


mikrocentrum
opleiden • ontmoeten • ondernemen

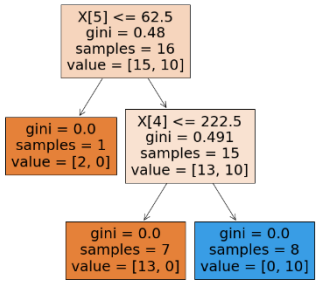
Classificeren



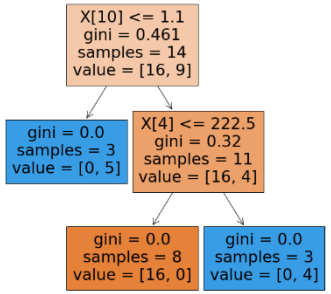
$$\mathbf{P} = \begin{bmatrix} 0.82 & 0.18 \end{bmatrix}$$



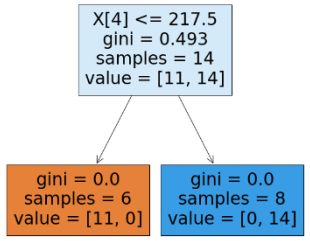
$$\mathbf{P} = \begin{bmatrix} 0.99 & 0.01 \end{bmatrix}$$



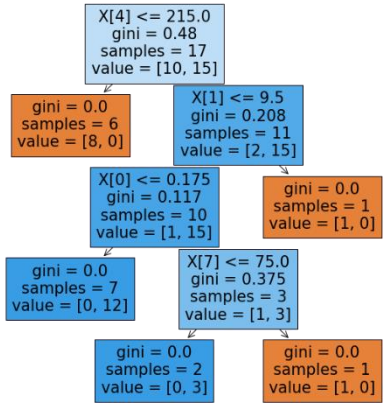
$$\mathbf{P} = \begin{bmatrix} 0.72 & 0.28 \end{bmatrix}$$



$$\mathbf{P} = \begin{bmatrix} 0.84 & 0.16 \end{bmatrix}$$



$$\mathbf{P} = \begin{bmatrix} 0.65 & 0.35 \end{bmatrix}$$



$$\mathbf{P} = \begin{bmatrix} 0.93 & 0.07 \end{bmatrix}$$

mikrocentrum
opleiden • ontmoeten • ondernemen

Aggregeren van de uitkomsten

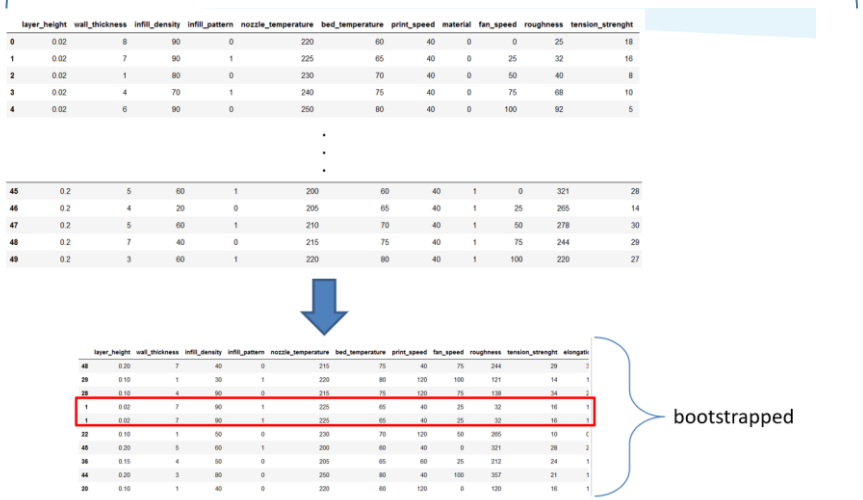
$$\mathbf{P} = \begin{bmatrix} 0.82 & 0.18 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.72 & 0.28 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.65 & 0.35 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.99 & 0.01 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.84 & 0.16 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.93 & 0.07 \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} 0.825 & 0.175 \end{bmatrix}$$

Aggregeren:
- Gemiddelde
- Meeste ‘stemmen’

Wordt ook wel eens Bagging trees genoemd

Bootstrapping + **Agg**regeren = **Bagging**



	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	bed_temperature	print_speed	material	fan_speed	roughness	tension_strenght
0	0.02	8	90	0	220	60	40	0	0	25	18
1	0.02	7	90	1	225	65	40	0	25	32	16
2	0.02	1	80	0	230	70	40	0	50	40	8
3	0.02	4	70	1	240	75	40	0	75	68	10
4	0.02	6	90	0	250	80	40	0	100	92	5
...											
45	0.2	5	60	1	200	60	40	1	0	321	28
46	0.2	4	20	0	205	65	40	1	25	265	14
47	0.2	5	60	1	210	70	40	1	50	278	30
48	0.2	7	40	0	215	75	40	1	75	244	29
49	0.2	3	60	1	220	80	40	1	100	220	27

bootstrapped

$$\mathbf{P} = \begin{bmatrix} 0.82 & 0.18 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.72 & 0.28 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.65 & 0.35 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.99 & 0.01 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.84 & 0.16 \end{bmatrix}$$
$$\mathbf{P} = \begin{bmatrix} 0.93 & 0.07 \end{bmatrix}$$

$$\mathbf{P} = \begin{bmatrix} 0.825 & 0.175 \end{bmatrix}$$

mikrocentrum
opleiden • ontmoeten • ondernemen

Random Forest

Voordelen

- Makkelijk/Simpel
- Goede resultaten met relatief weinig data
- Betrouwbaar/Robuust
- Minder gevoelig voor overfitting (kan nog steeds gebeuren)
- Goed met categorische en numerieke data

Nadelen

- Langzamer dan DT’s
- *Black box*
- Optimale oplossing niet altijd gegarandeerd

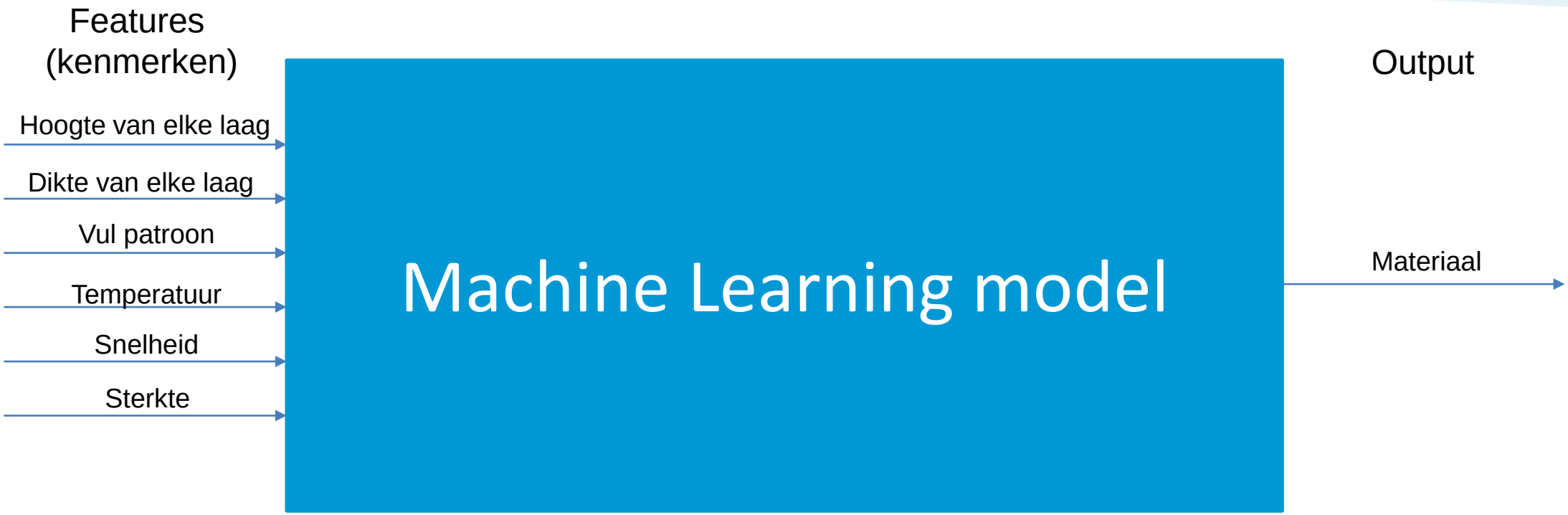
Oefening

Maak een *Random Forest* classifier die voorspelt met welk materiaal is geprint

```
from sklearn.ensemble import RandomForestClassifier
```

Welke werkt beter? Random Forest of Decision Tree?

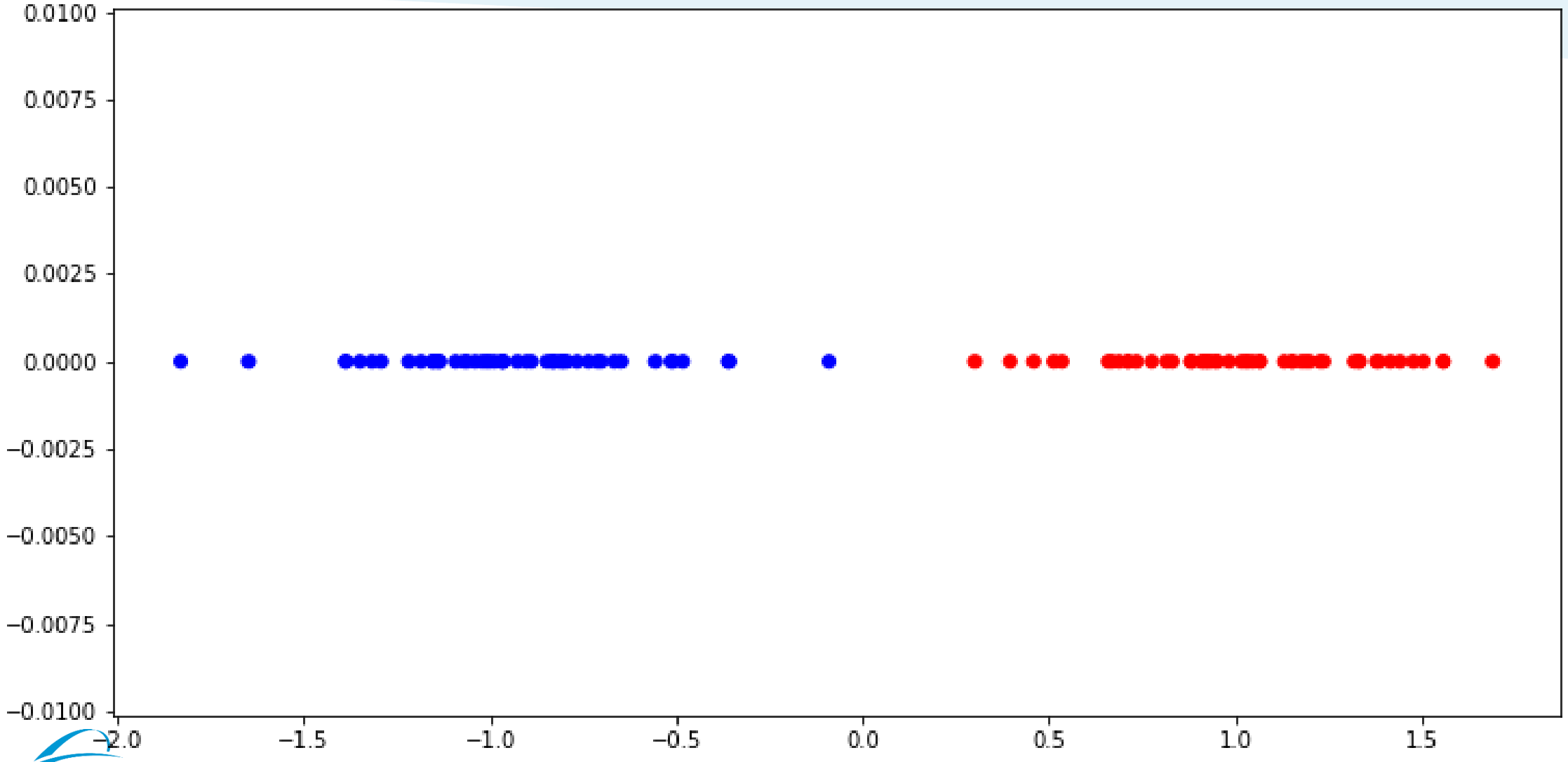
~ 5 minuten



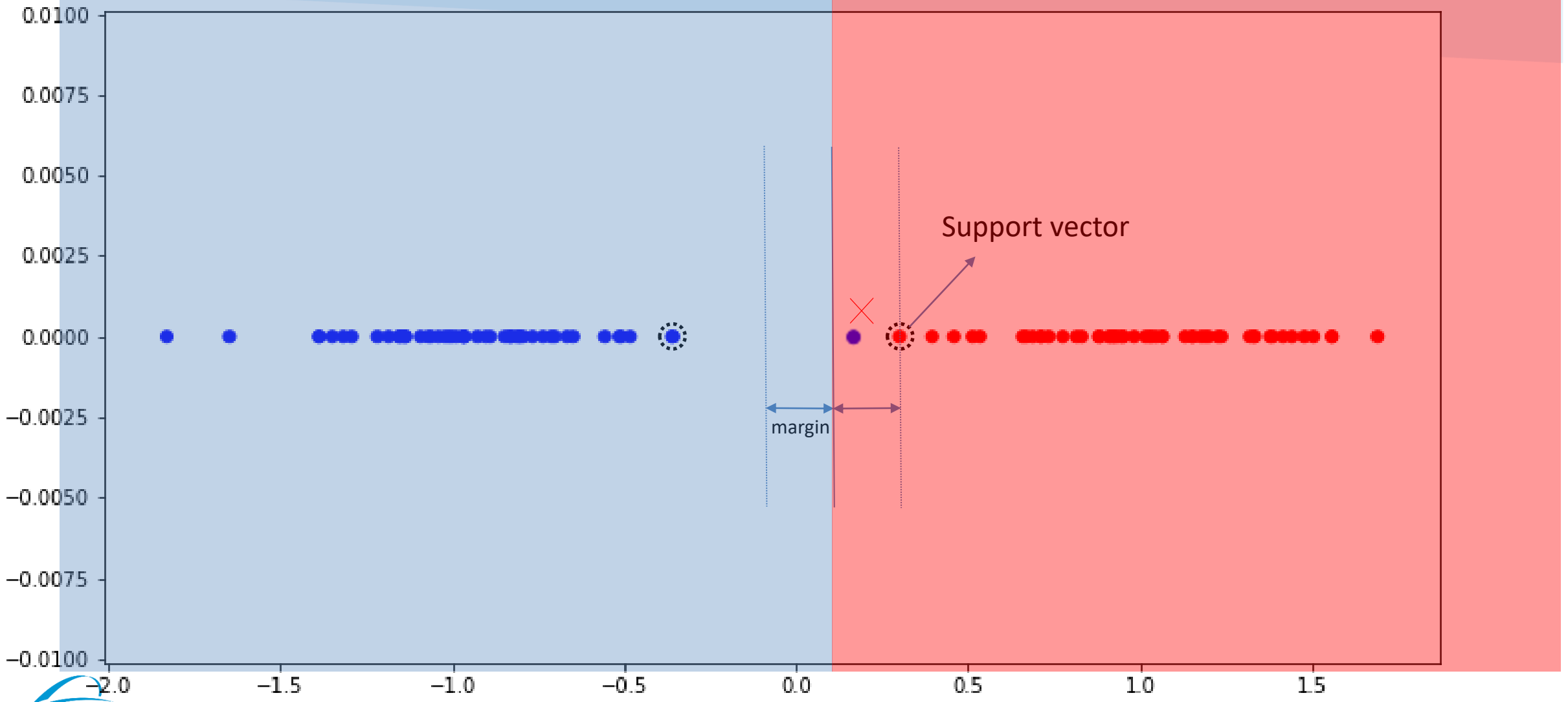
$$\mathbf{Y} = f(\mathbf{X})$$

Model 2:

SUPPORT VECTOR MACHINES

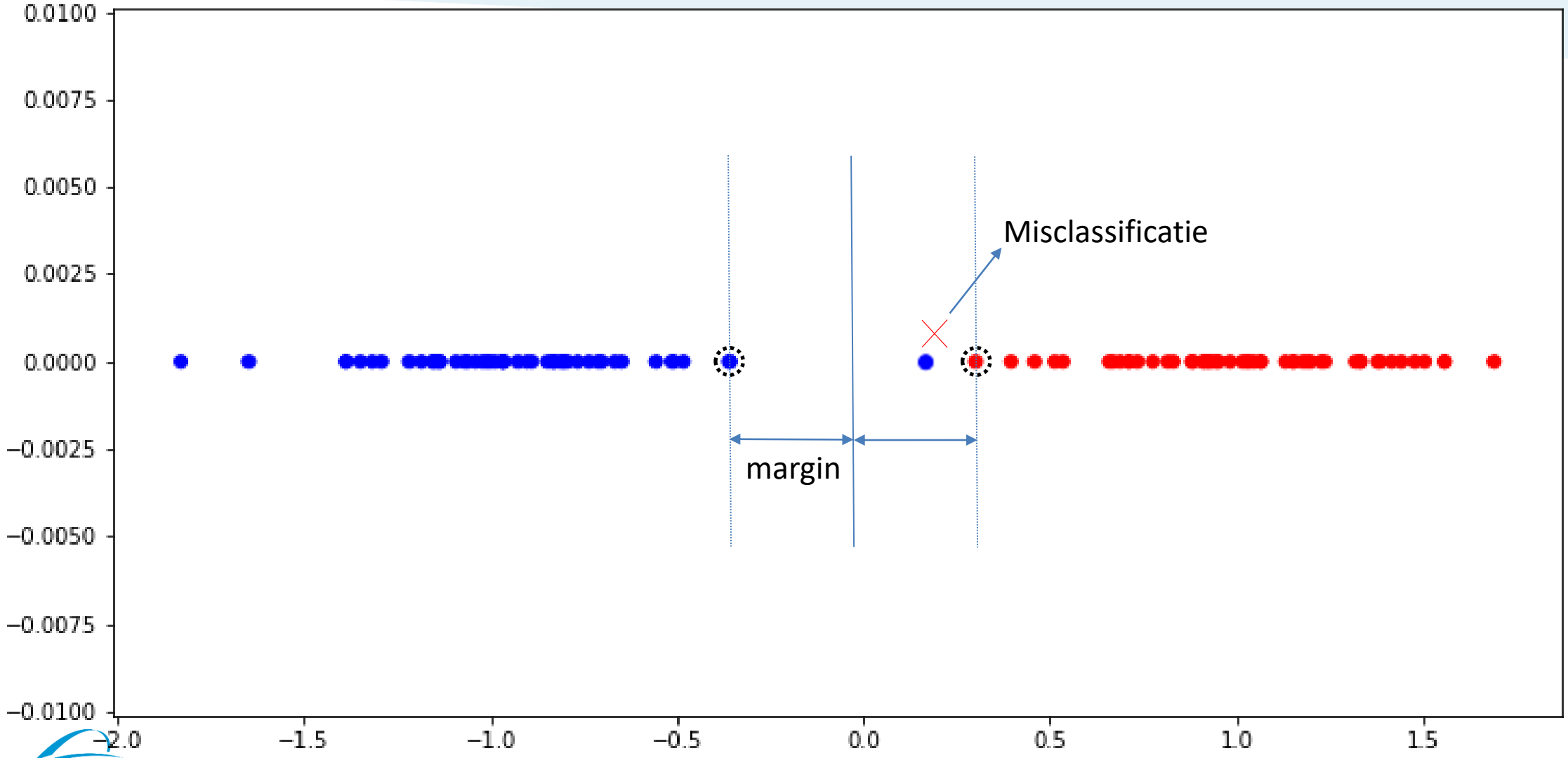

0.0100
0.0075
0.0050
0.0025
0.0000
−0.0025
−0.0050
−0.0075
−0.0100
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

SVM

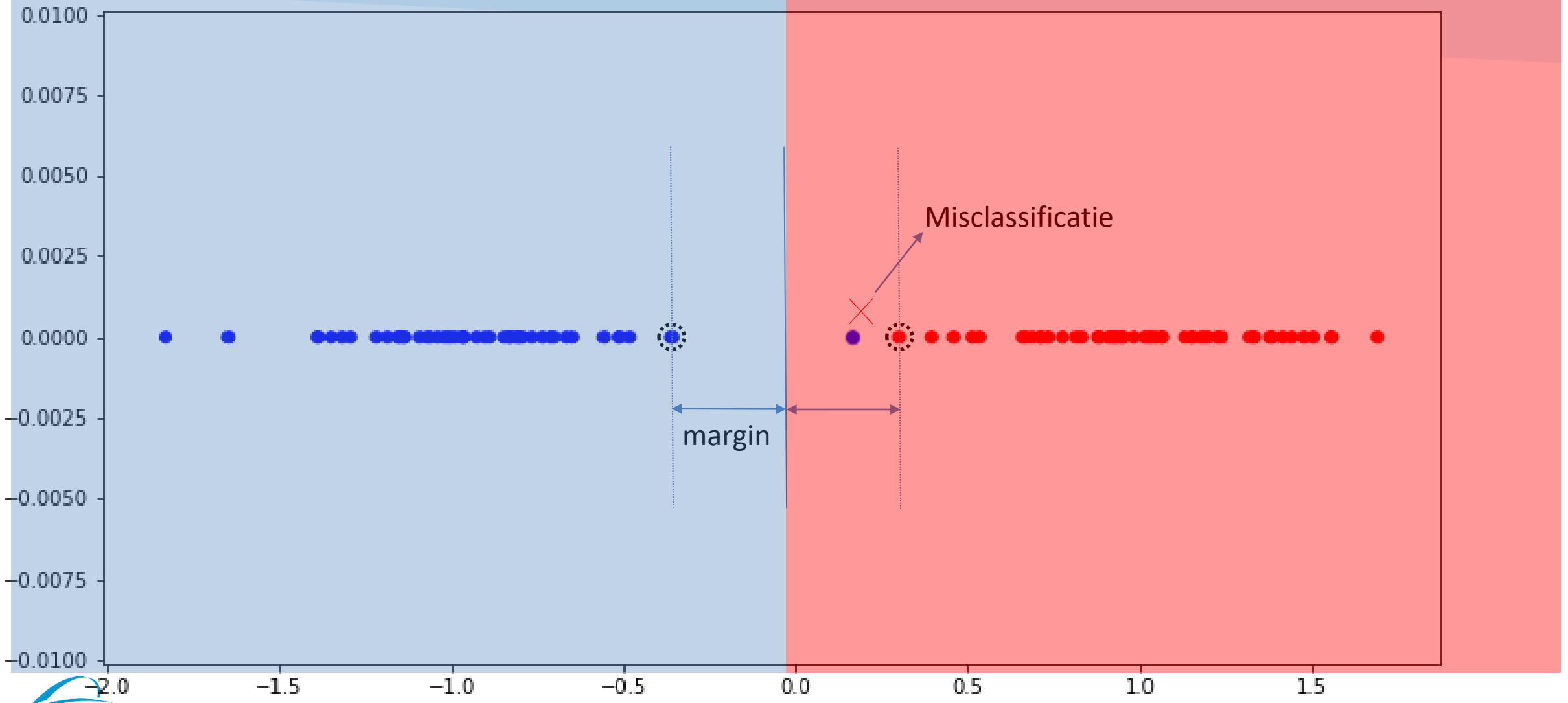


mikrocentrum
opleiden • ontmoeten • ondernemen

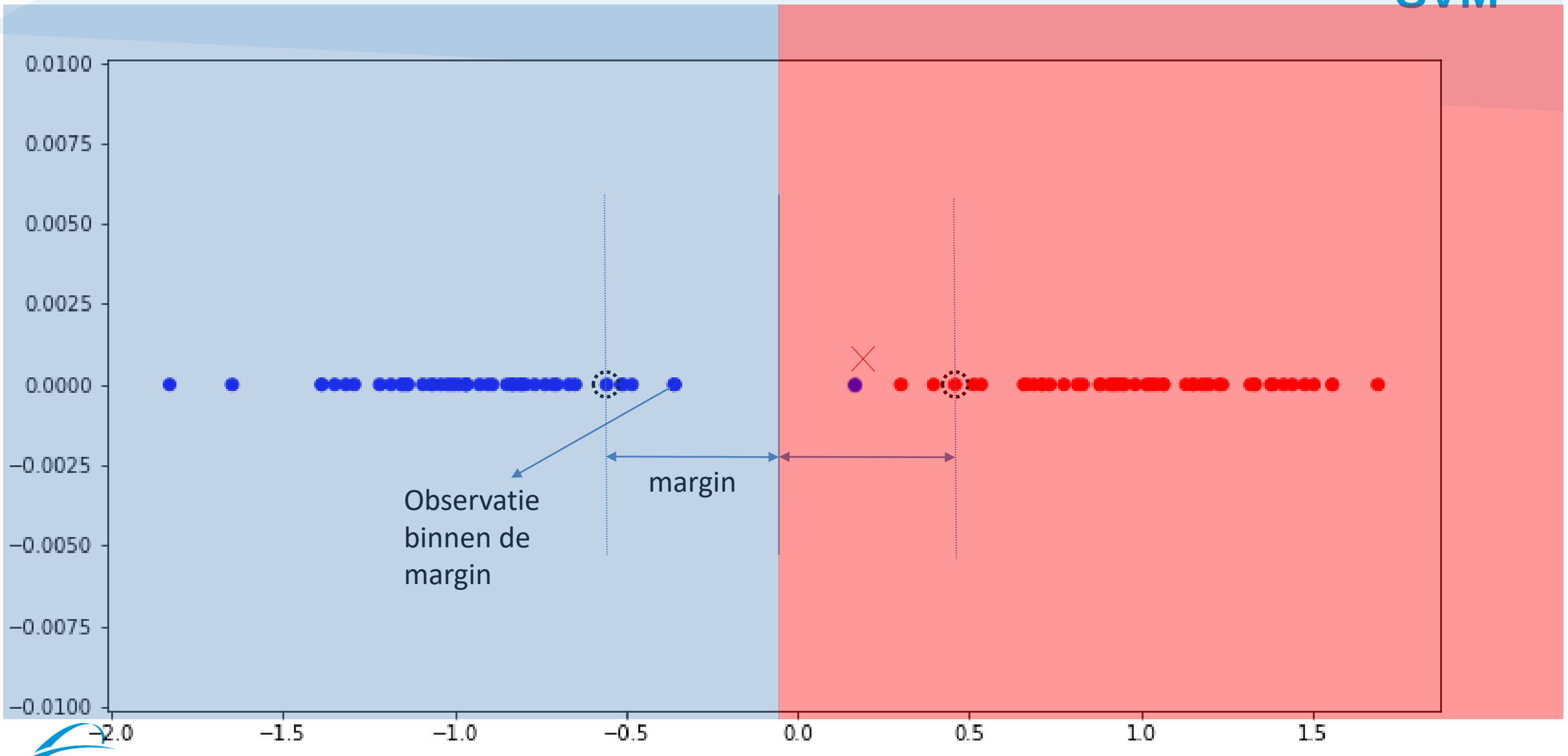
SVM


0.0100
0.0075
0.0050
0.0025
0.0000
−0.0025
−0.0050
−0.0075
−0.0100
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
Misclassificatie
margin

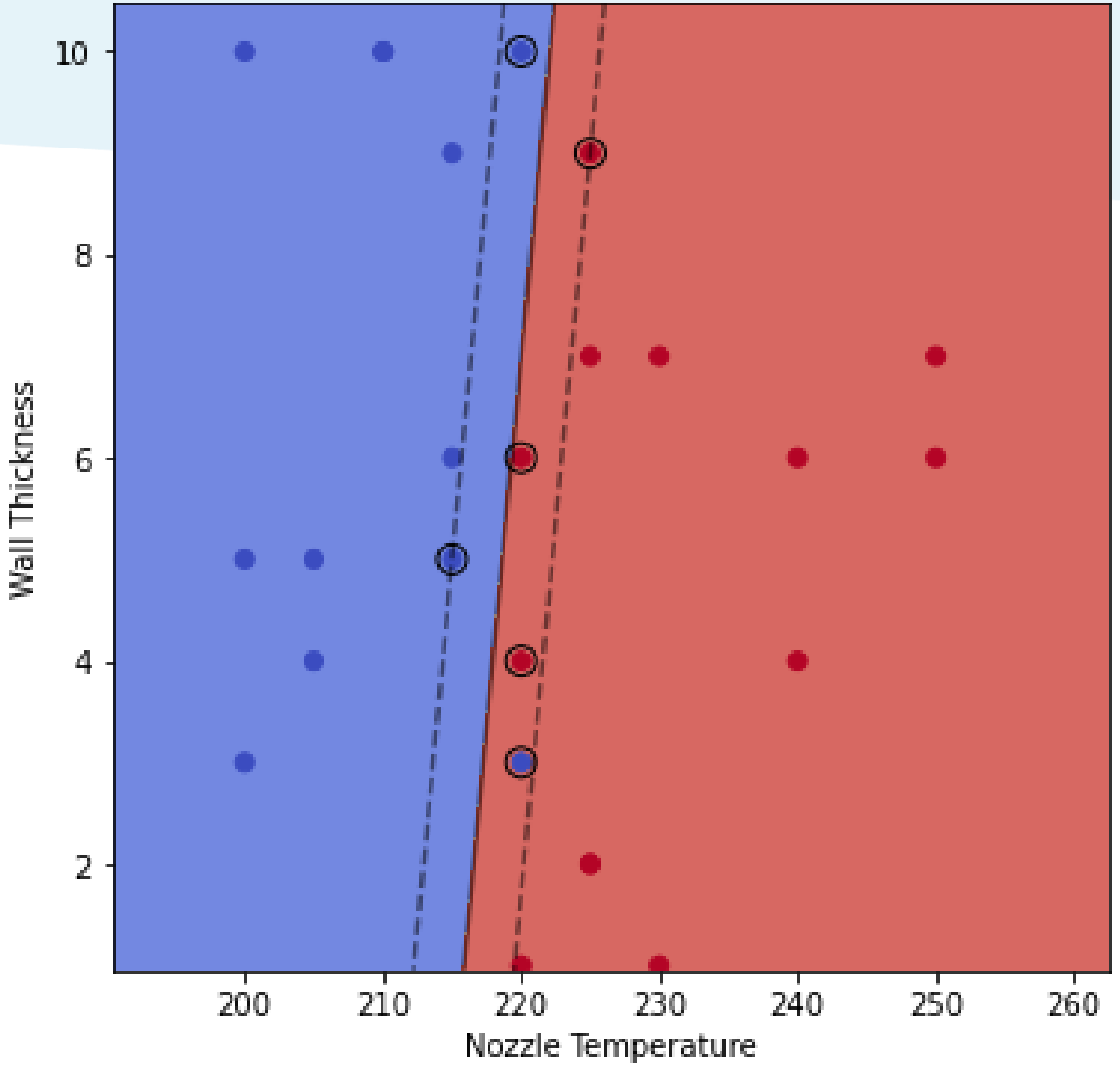
SVM


0.0100
0.0075
0.0050
0.0025
0.0000
−0.0025
−0.0050
−0.0075
−0.0100
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
Misclassificatie
margin

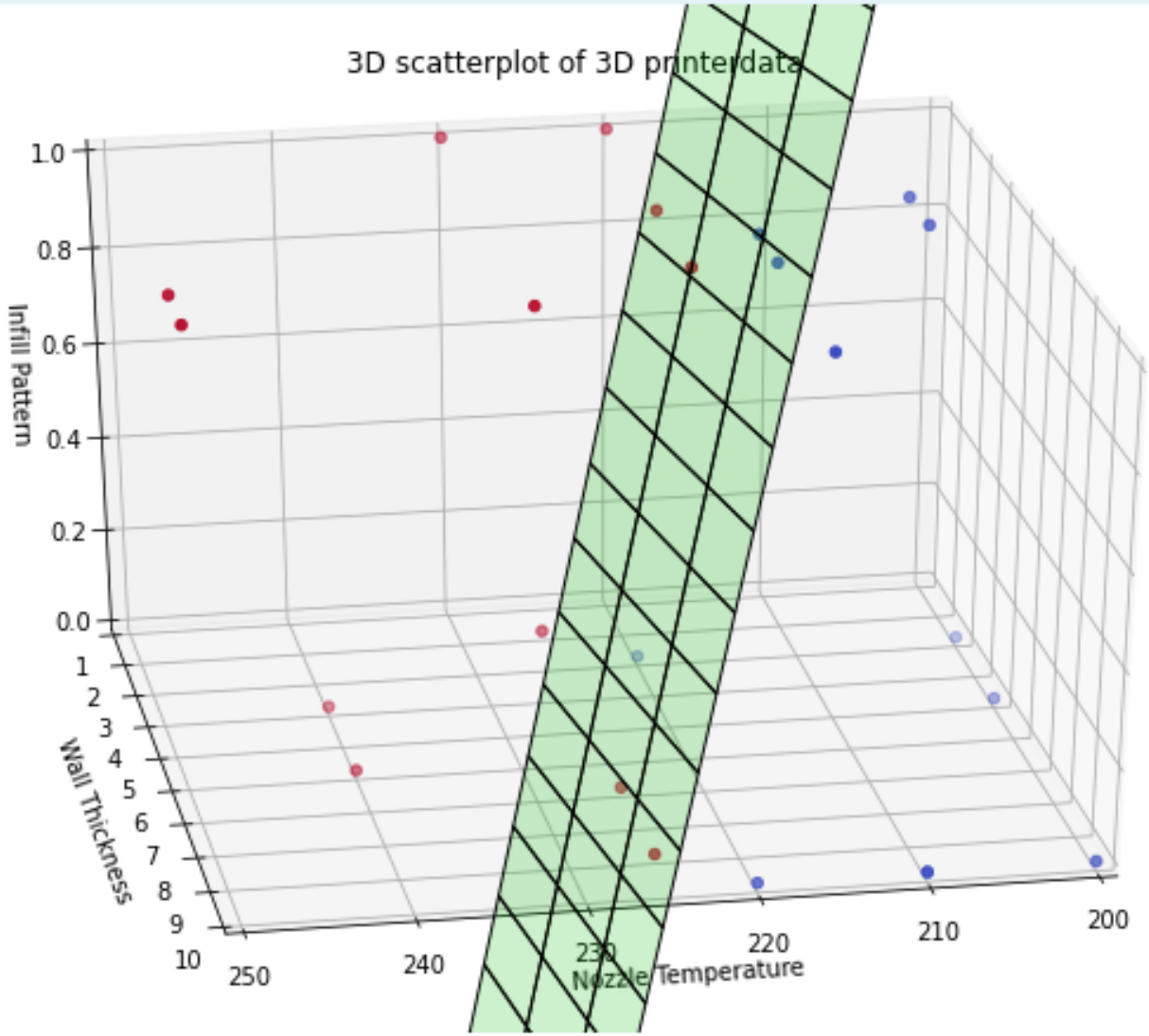
SVM



mikrocentrum
opleiden • ontmoeten • ondernemen

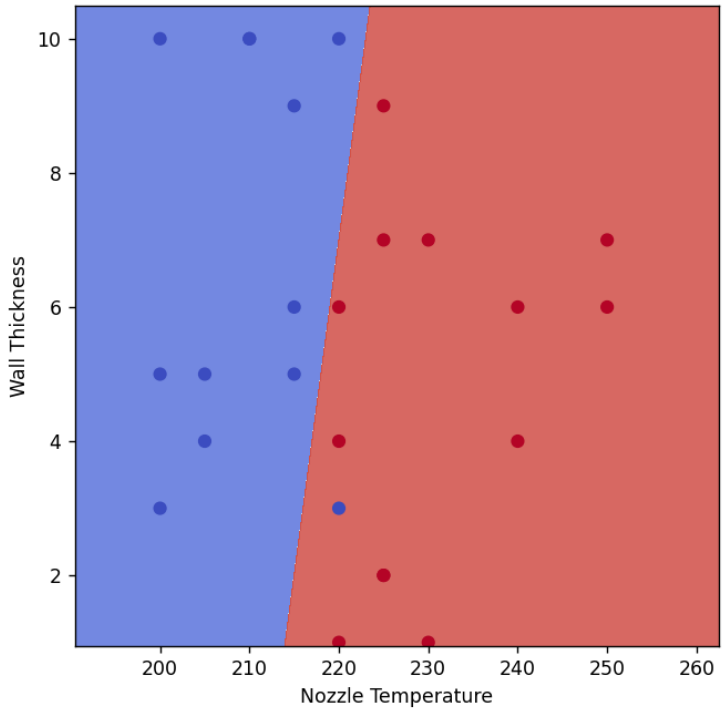

Wall Thickness
10
8
6
4
2
200
210
220
230
240
250
260
Nozzle Temperature

Scatterplot

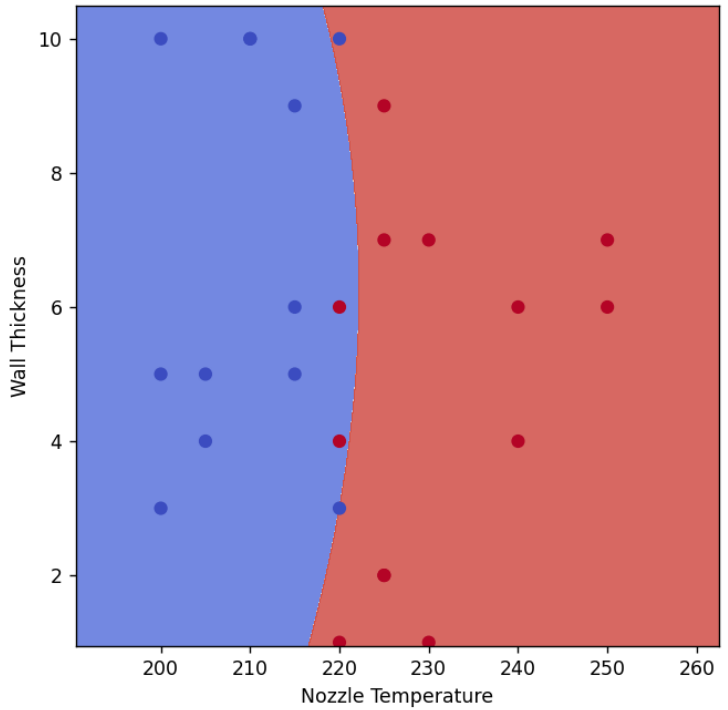


Kernel trick

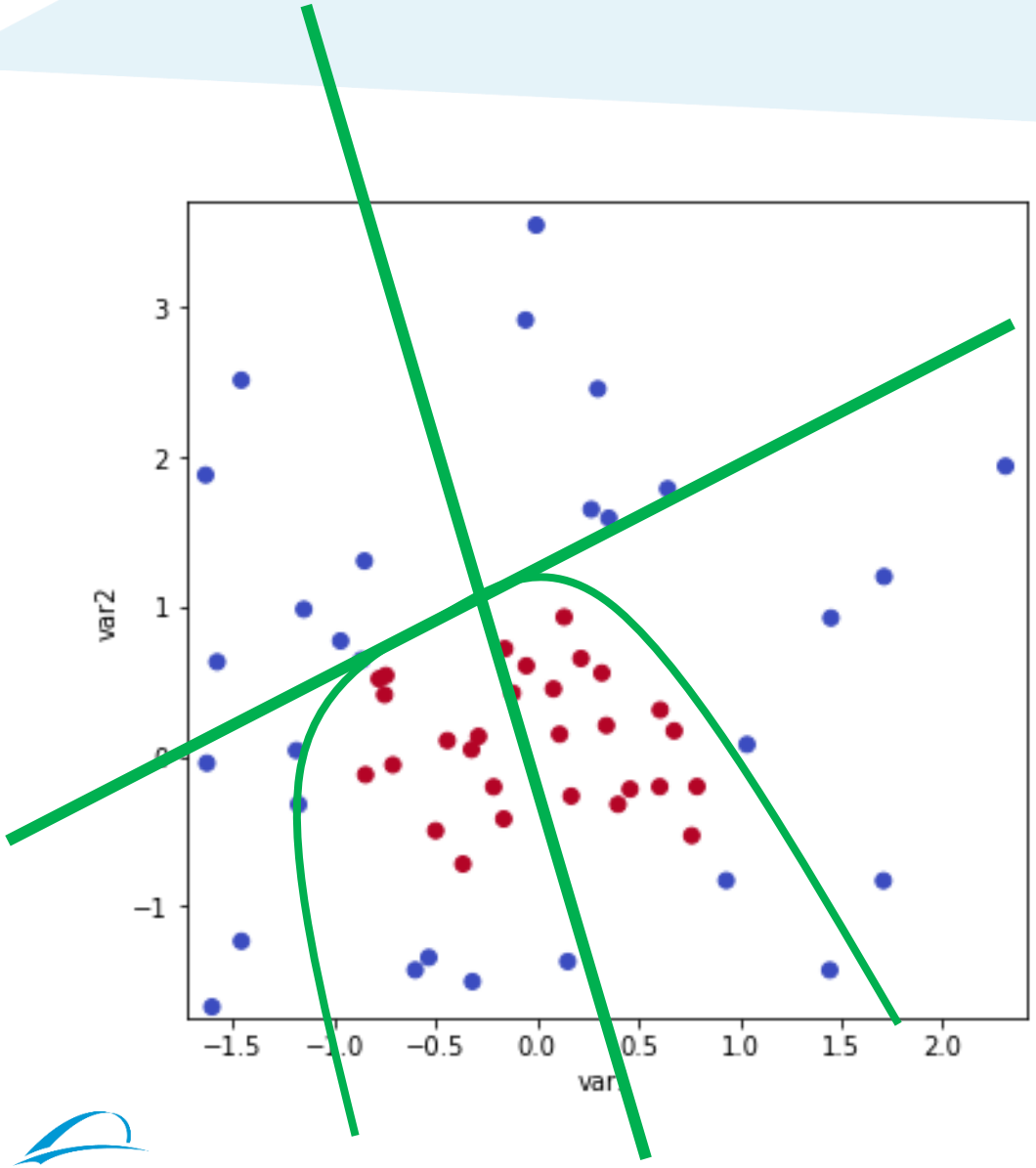
Lineair

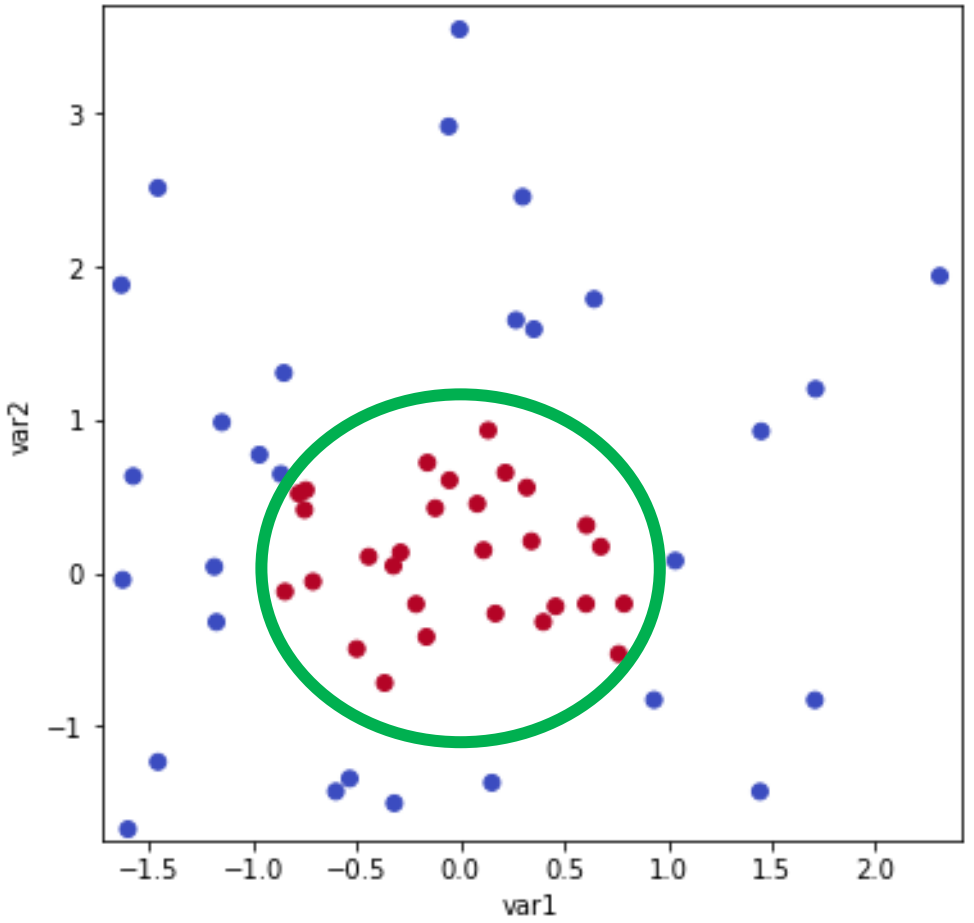


Polynoom



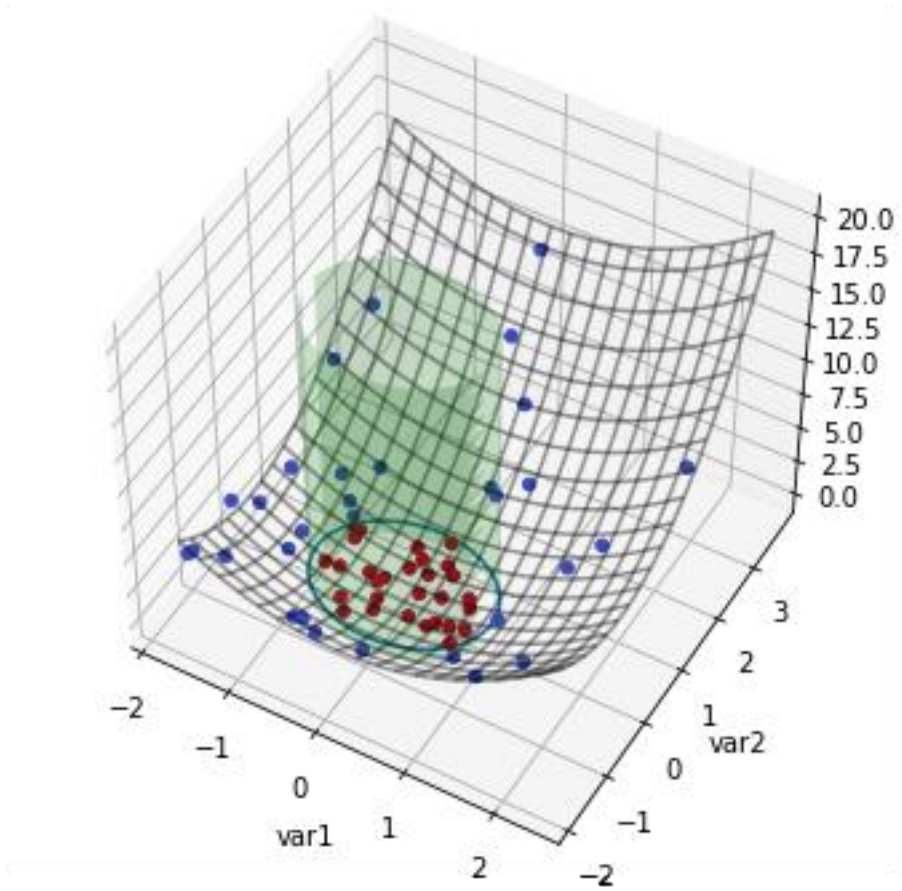
Radial Basis Function

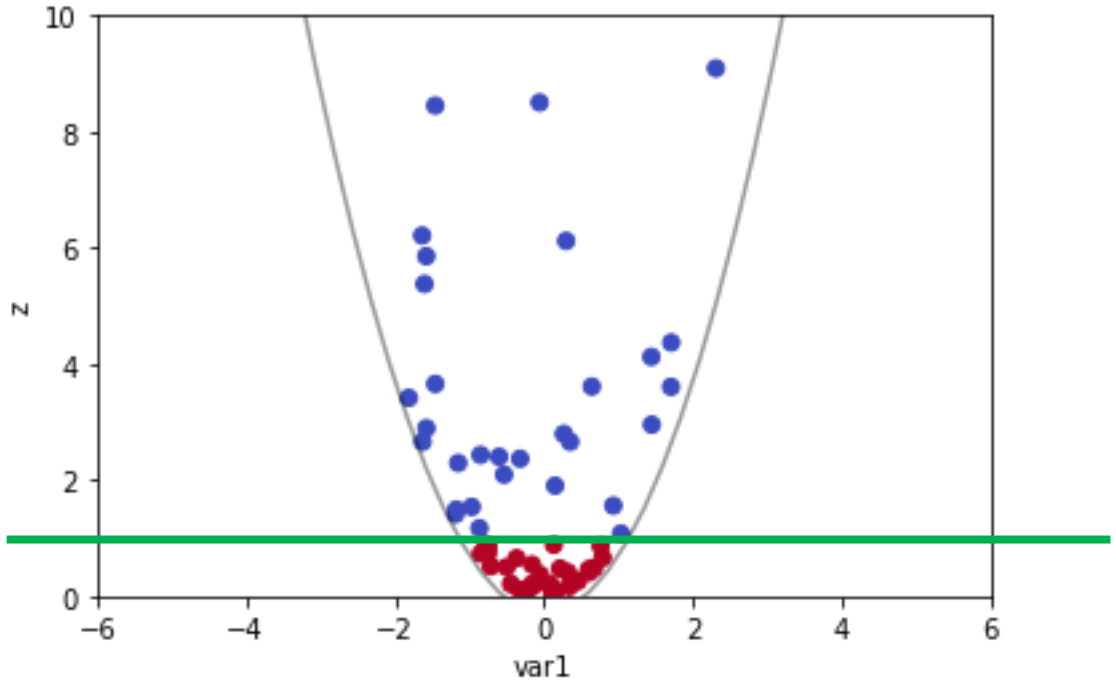


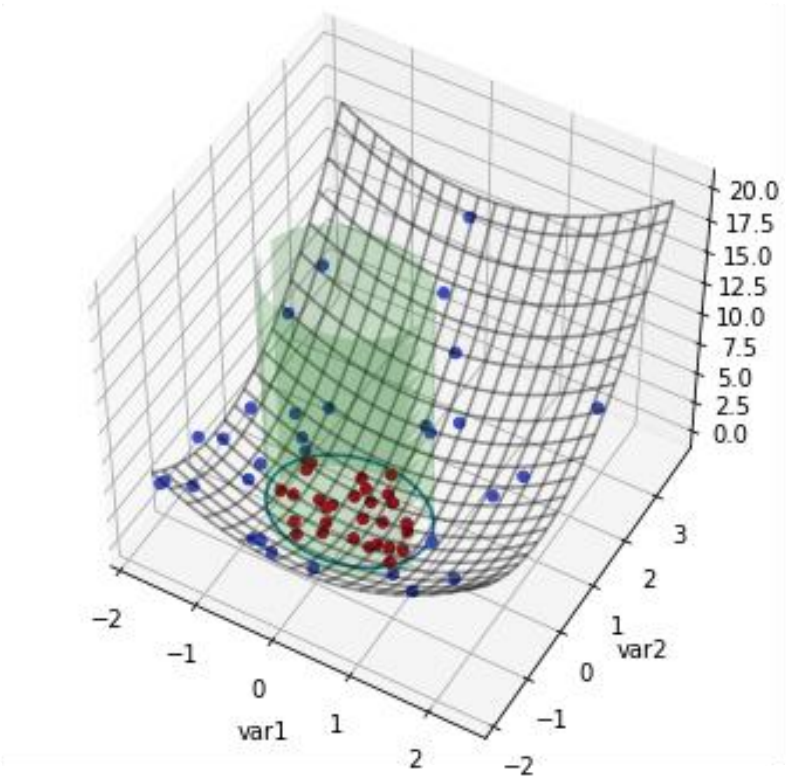


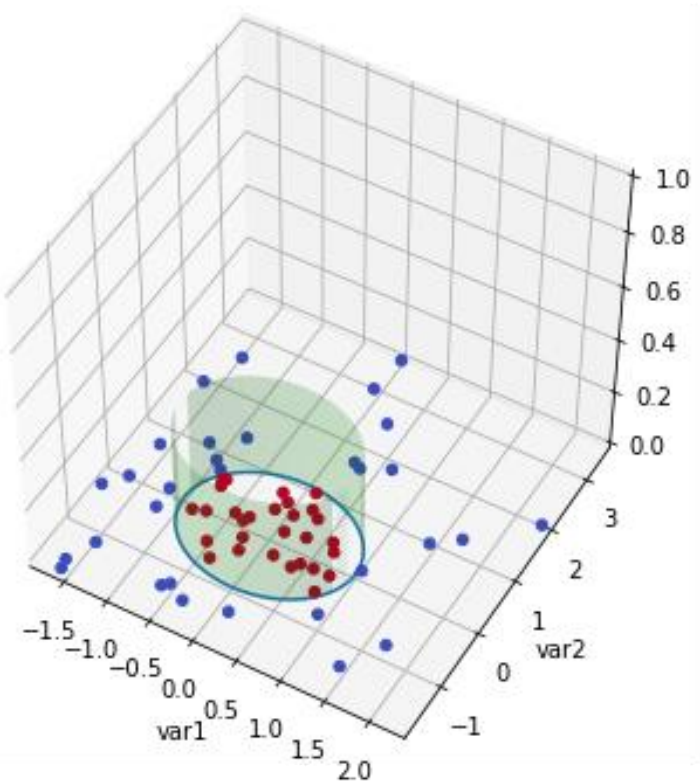
mikrocentrum
opleiden • ontmoeten • ondernemen

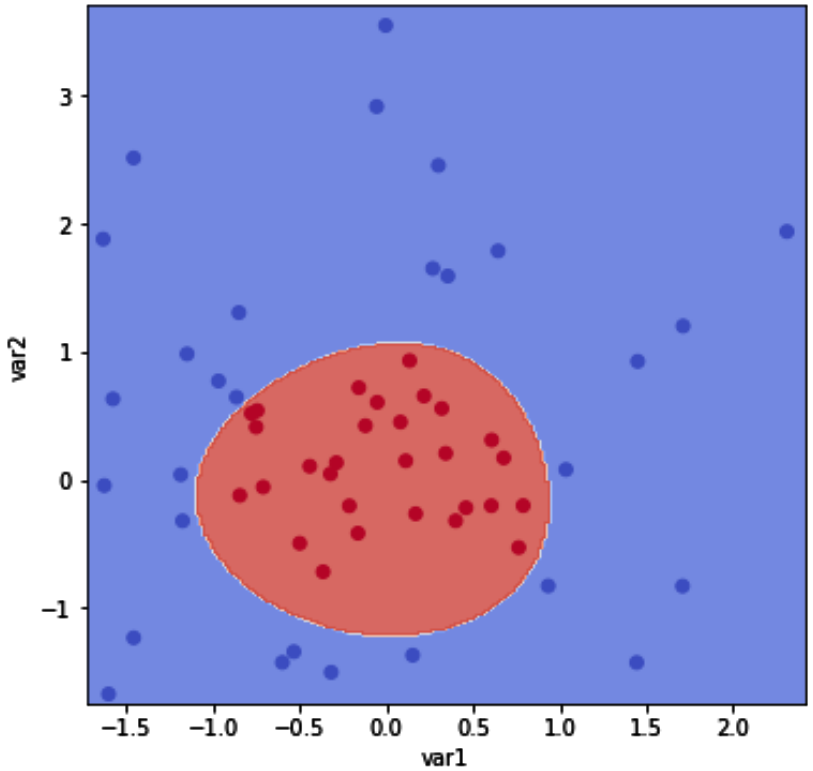
Kernel trick





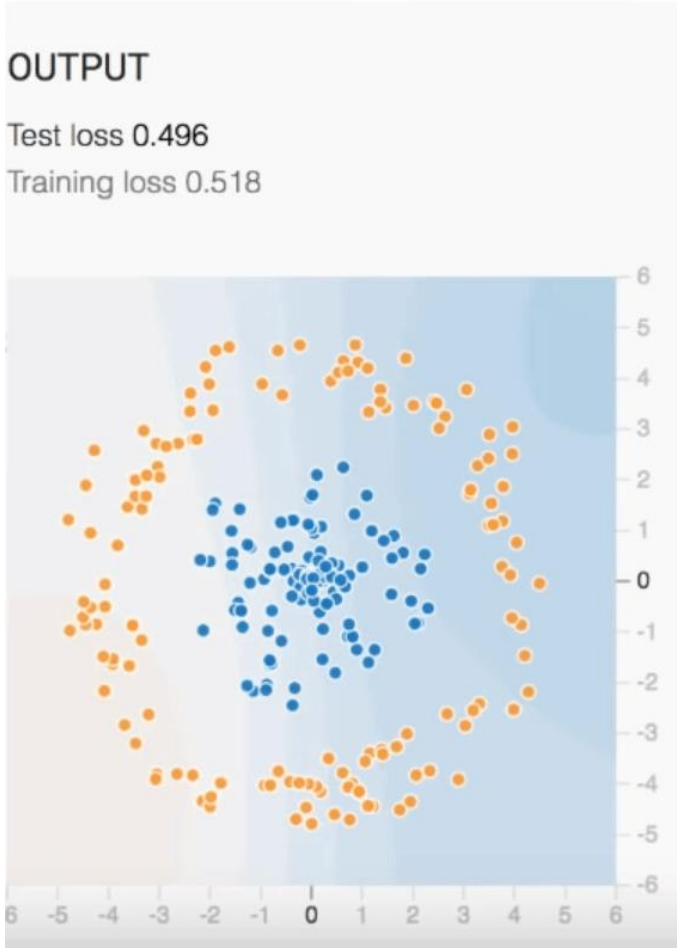

20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−2
−1
0
1
2
var1
−2
−1
0
1
2
3
var2


1.0
0.8
0.6
0.4
0.2
0.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
var1
−1
0
1
2
3
var2


3
2
1
0
−1
var2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
var1

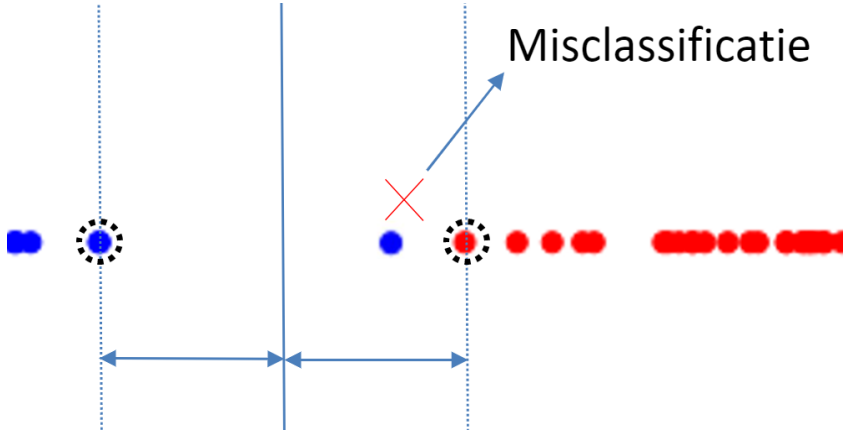
SVM optimization

- Kernel trick ook wiskundig handig
  - Makkelijk te optimaliseren
  - Geen informatie over de data nodig



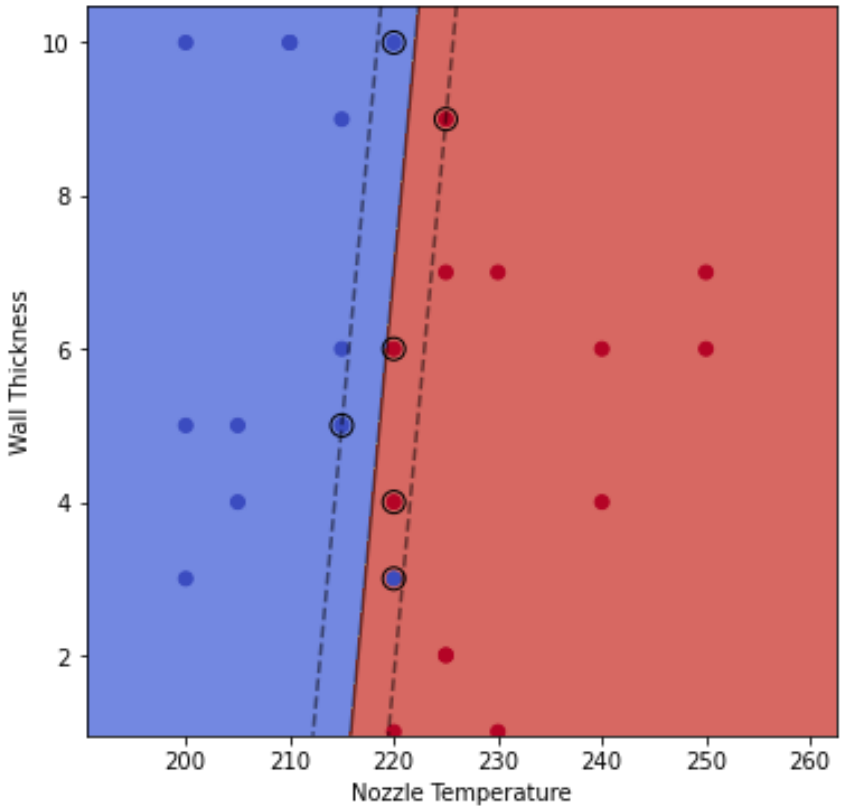
Hyperparameters

- Function/Kernel `['linear', 'poly', 'rbf']`
- C → penalty/*regularization* voor misclassificatie
- Gamma γ (alleen voor `'poly' & 'rbf'`) → De invloed van 1 datapunt.

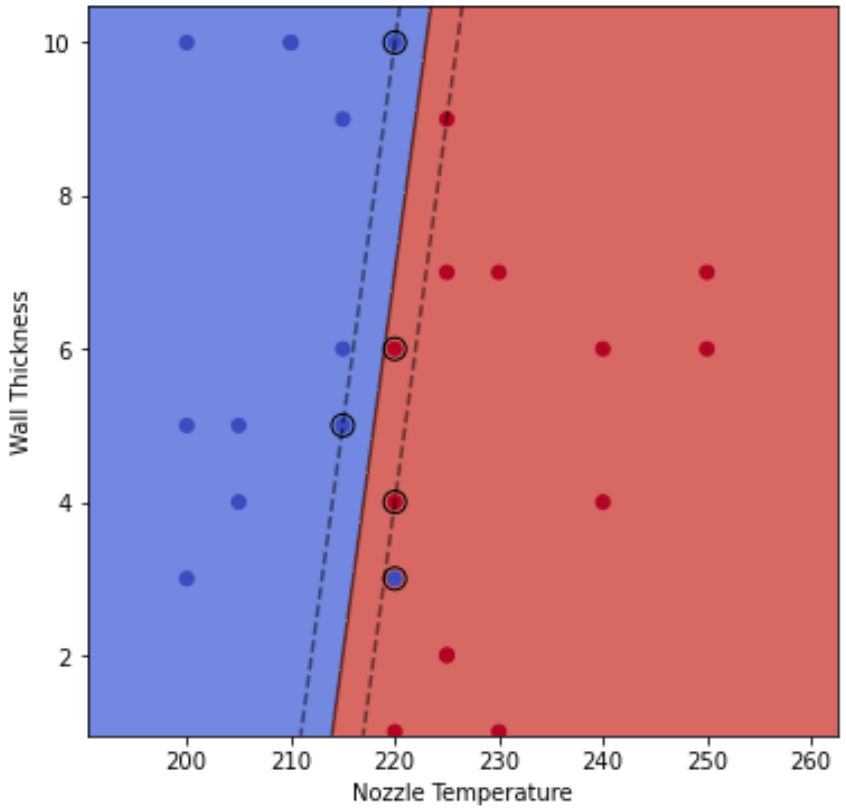


C (penalty/*regularization*)

C = 0.1

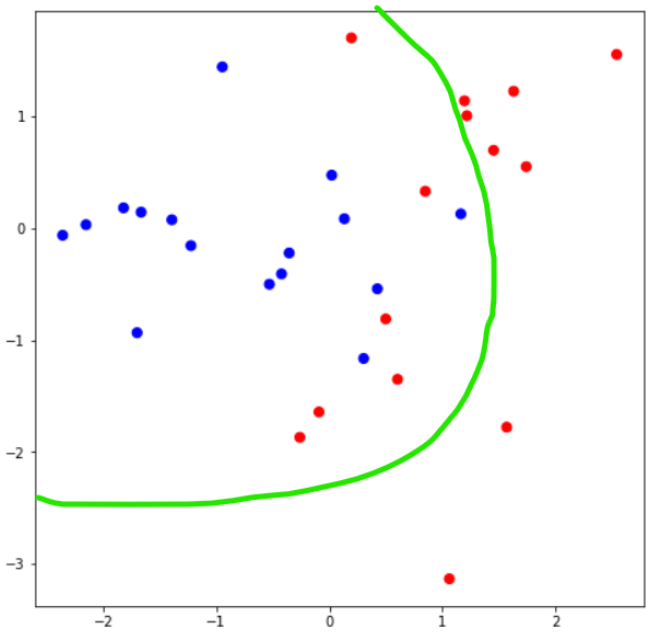


C = 10

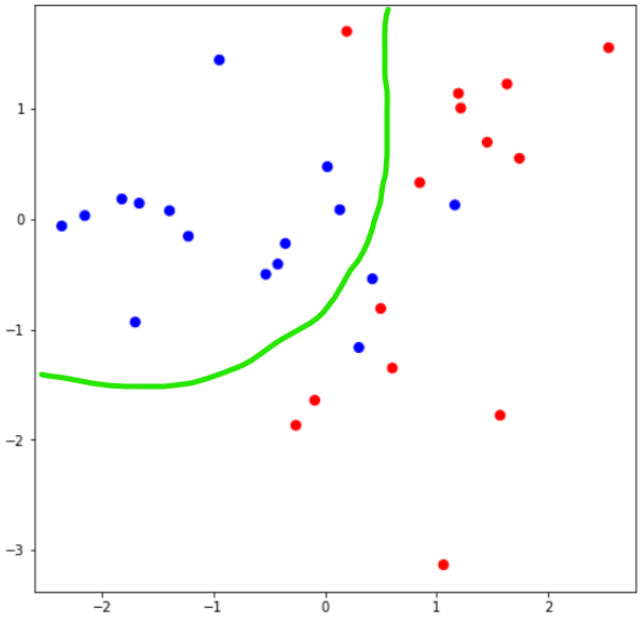


C (penalty/*regularization*)

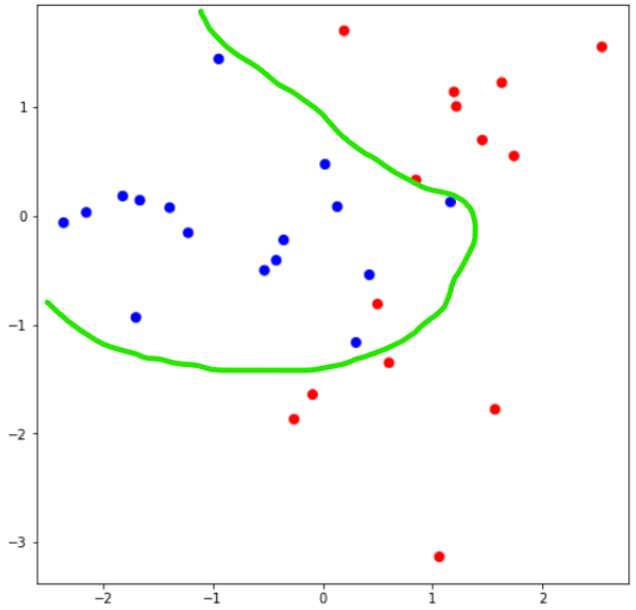
Lage C



Middel C

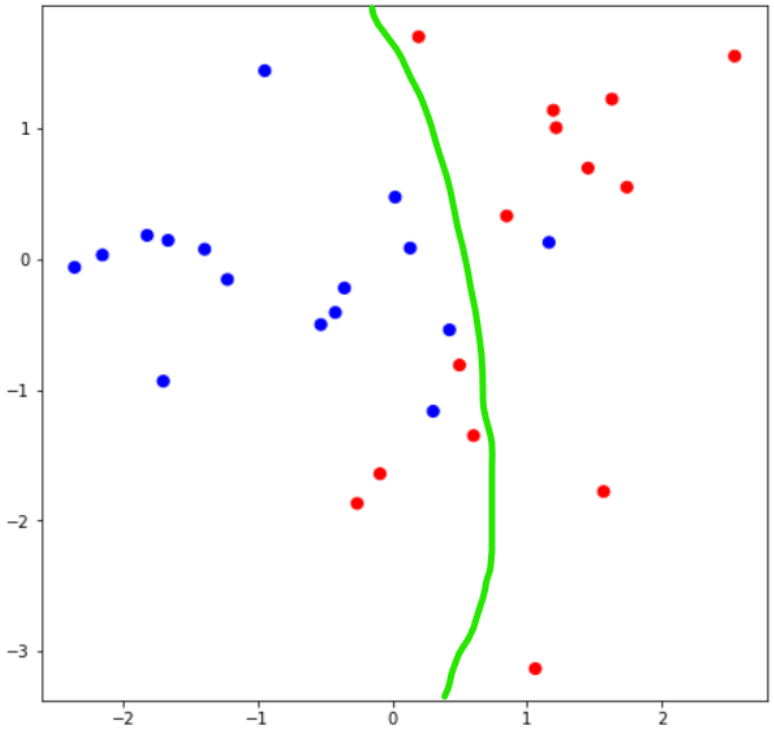


Hoge C

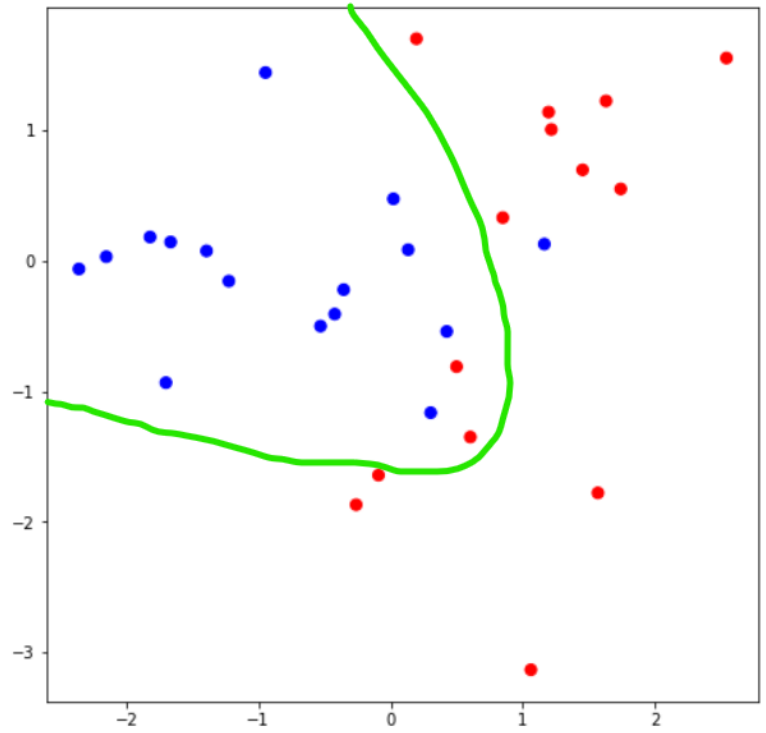


Gamma

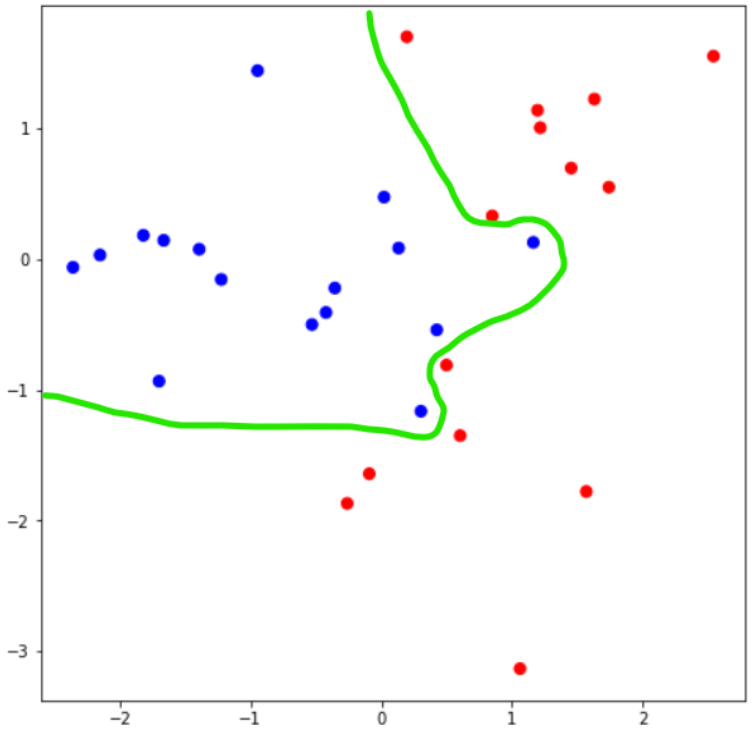
Lage Gamma



Middel Gamma



Hoge Gamma



$$\gamma = \texttt{"scale"} = \frac{1}{N_{\text{features}}\sigma}$$

$$\gamma = \texttt{"auto"} = \frac{1}{N_{\text{features}}}$$

Oefening

Maak een *Support Vector Machine* classifier die voorspelt met welk materiaal is geprint

```
from sklearn.svm import SVC
```

~ 5 minuten

Support Vector Machine

Voordelen

- Goed wanneer er een duidelijke scheidingslijn is tussen de classes
- Werkt goed met weinig data en veel features
- Veel mogelijkheden met kernels

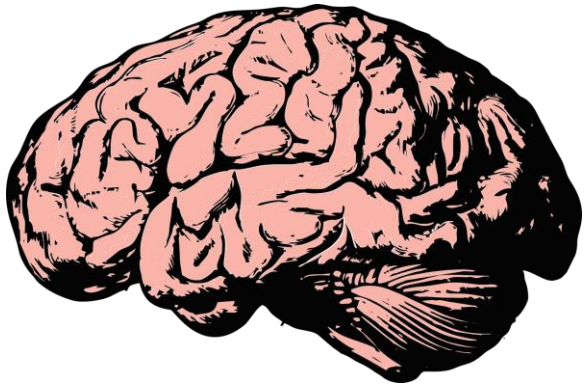
Nadelen

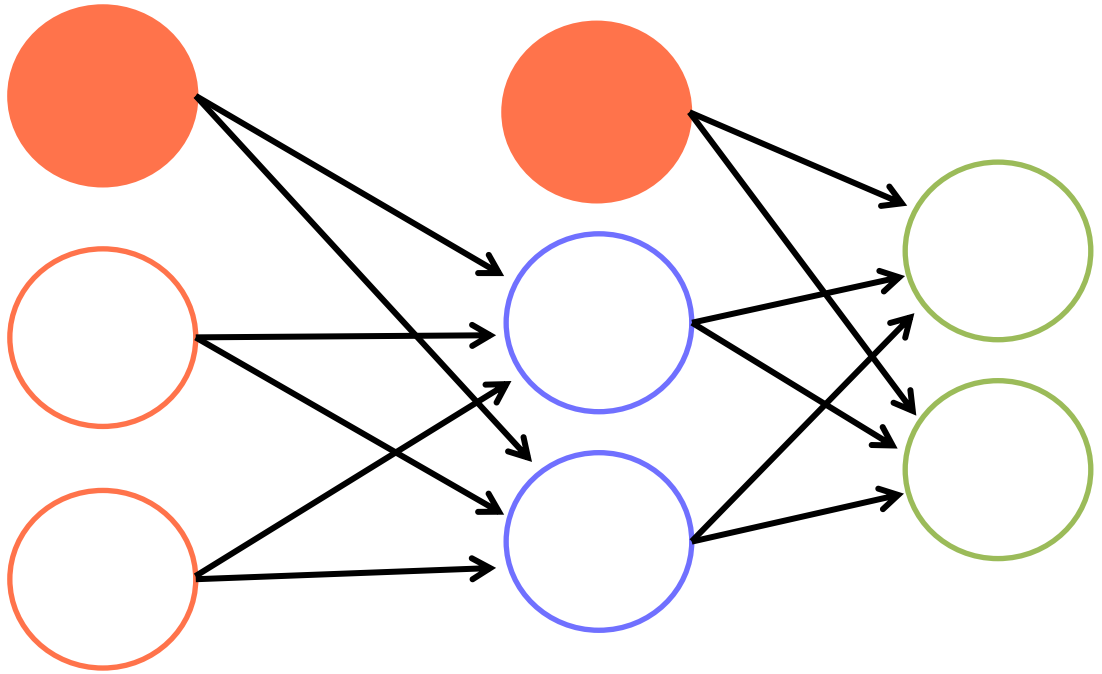
- Gevoelig voor ruis en overlappende classes
- Niet goed met grote datasets
- Veel mogelijkheden met kernels

Methode 3

NEURALE NETWERKEN

Neurale netwerken





$$\frac{\partial E}{\partial \mathbf{W}^{(L)}} = \frac{\partial E}{\partial \mathbf{y}} \frac{\partial f_L\left(\mathbf{W}^{(L)}\mathbf{x}^{(L-1)}\right)}{\partial \mathbf{W}^{(L)}}$$

$$= \frac{\partial E}{\partial \mathbf{y}} \odot \frac{\partial f_L\left(\mathbf{W}^{(L)}\mathbf{x}^{(L-1)}\right)}{\partial \mathbf{W}^{(L)}\mathbf{x}^{(L-1)}} \frac{\partial \mathbf{W}^{(L)}\mathbf{x}^{(L-1)}}{\partial \mathbf{W}^{(L)}}$$

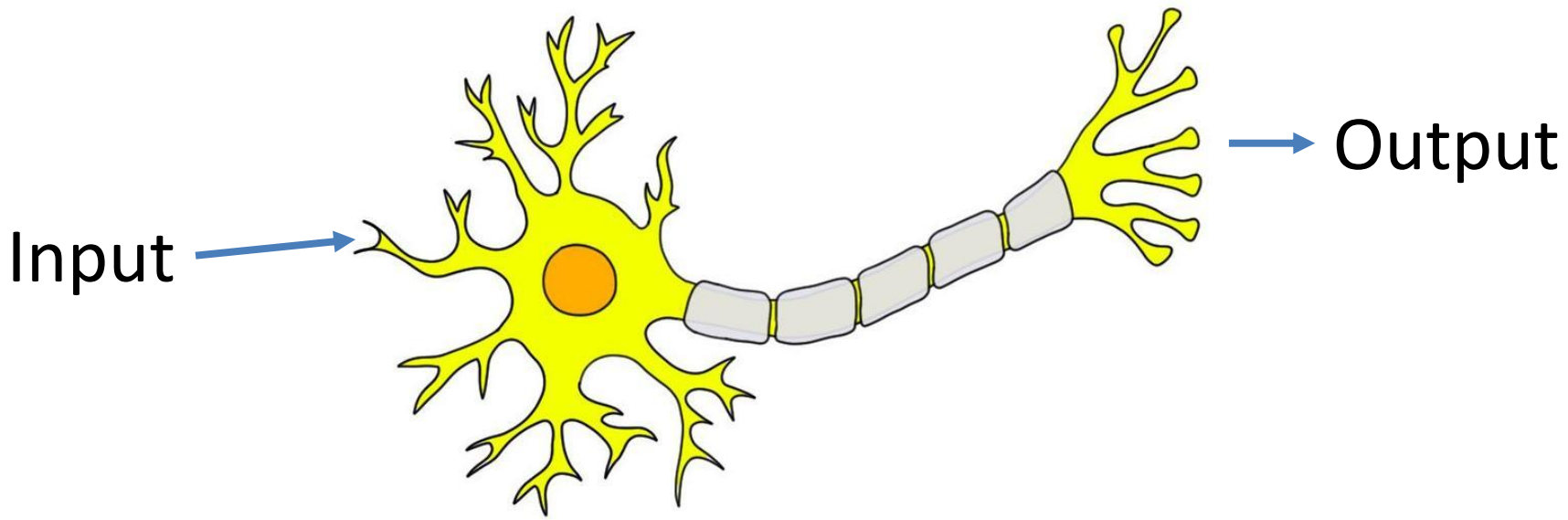
$$= \frac{\partial E}{\partial \mathbf{y}} \odot f_L'\left(\mathbf{W}^{(L)}\mathbf{x}^{(L-1)}\right)\left(\mathbf{x}^{(L-1)}\right)^{\mathrm{T}}$$

Substitute $\delta_L = \frac{\partial E}{\partial \mathbf{y}} \odot f_L'\left(\mathbf{W}^{(L)}\mathbf{x}^{(L-1)}\right)$

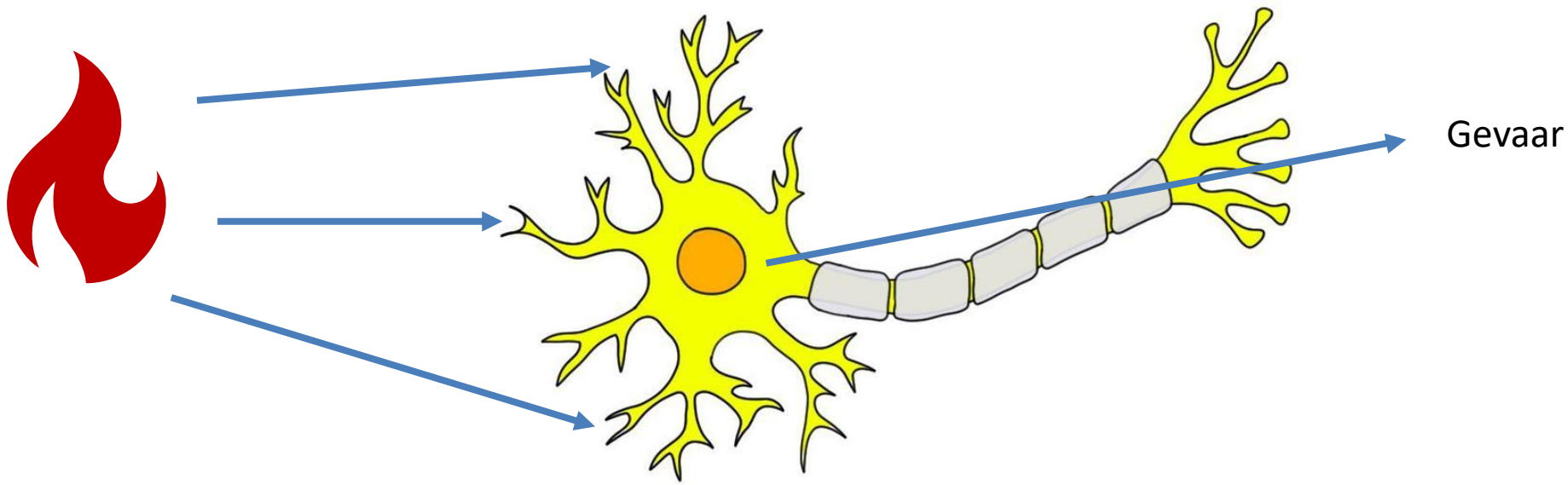
$$\frac{\partial E}{\partial \mathbf{W}^{(L)}} = \delta_L\left(\mathbf{x}^{(L-1)}\right)^{\mathrm{T}}$$

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), 115–133.

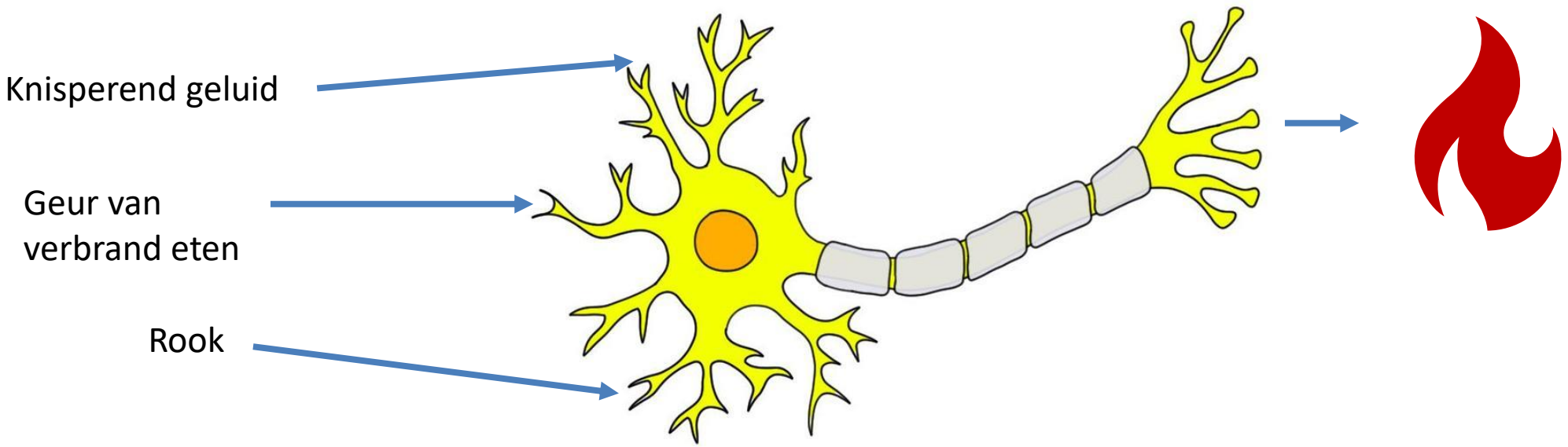
Neuroscience



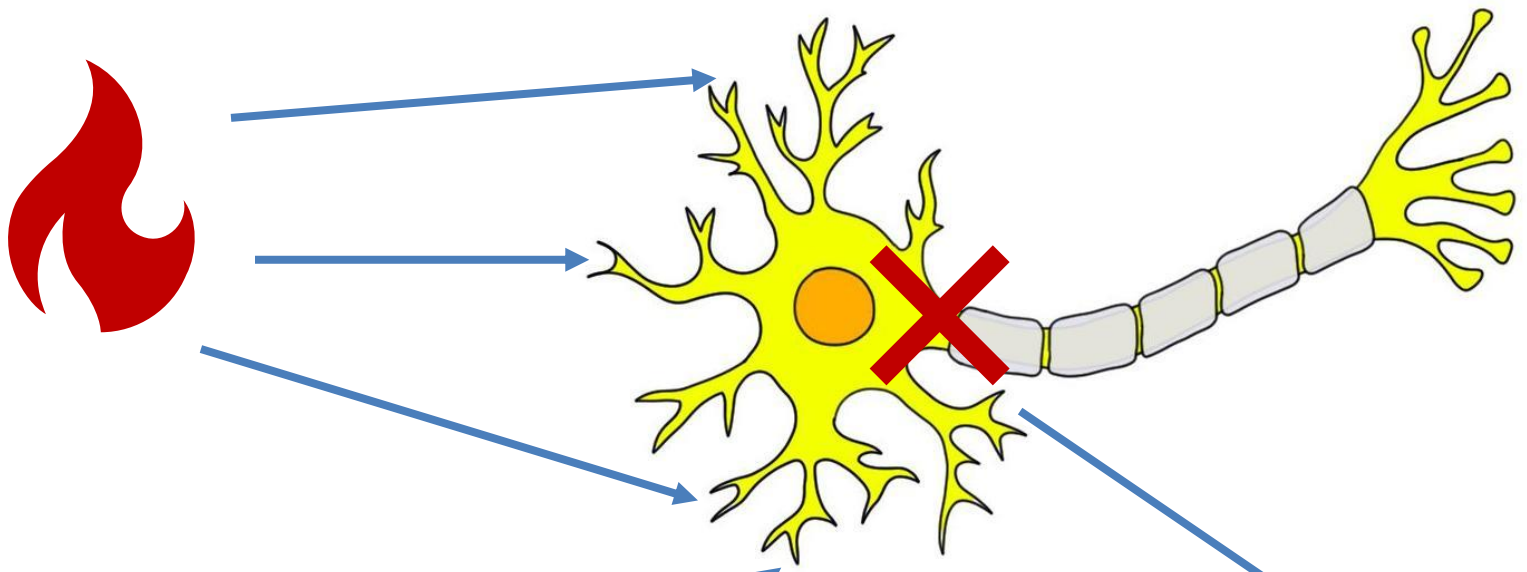
Neuron voor gevaar



Neuron voor vuur



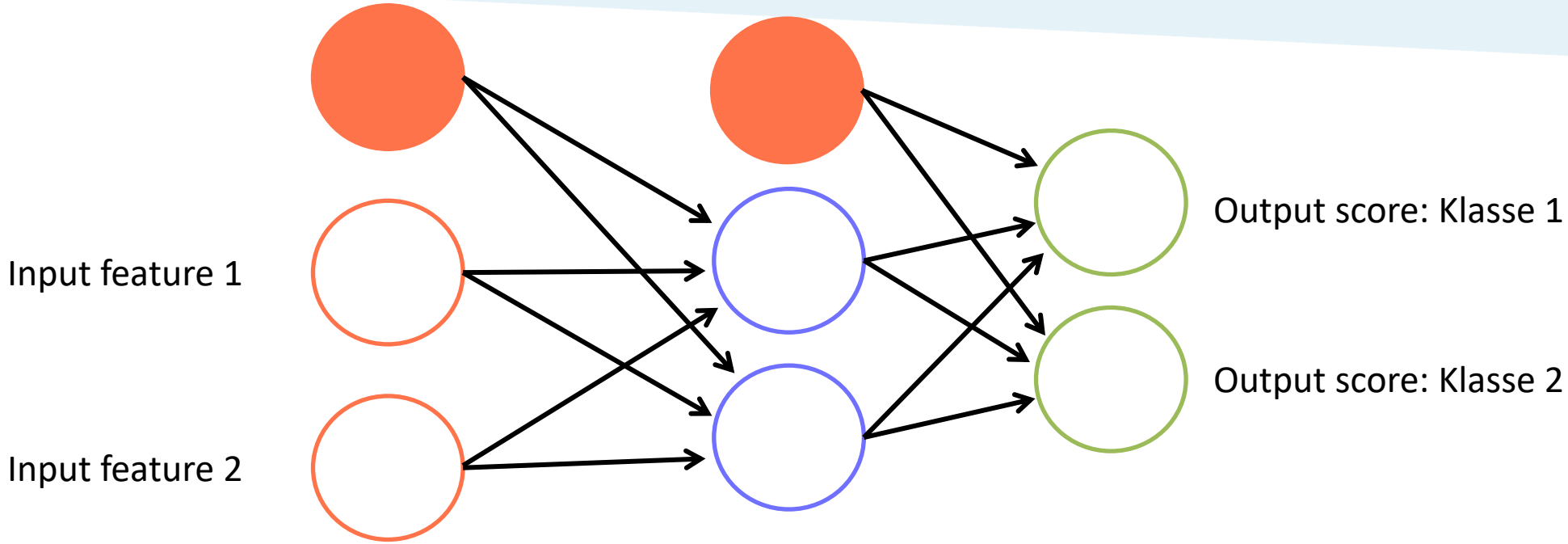
Neuron voor gevaar

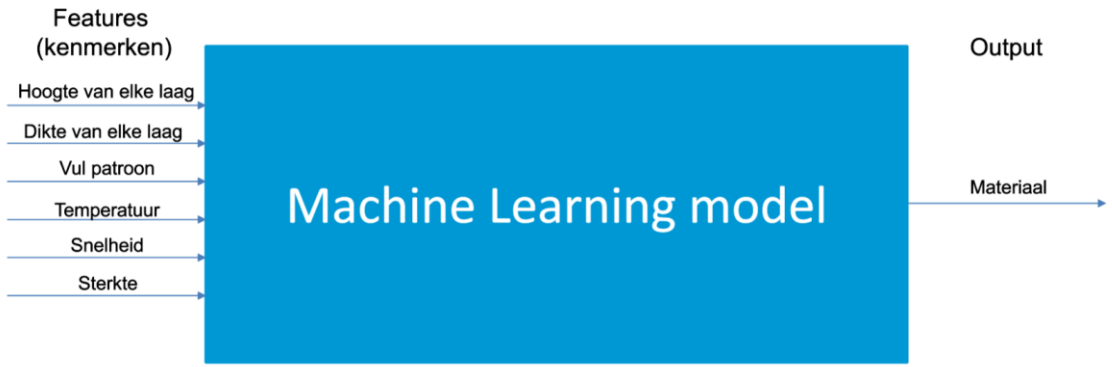


Kampvuur

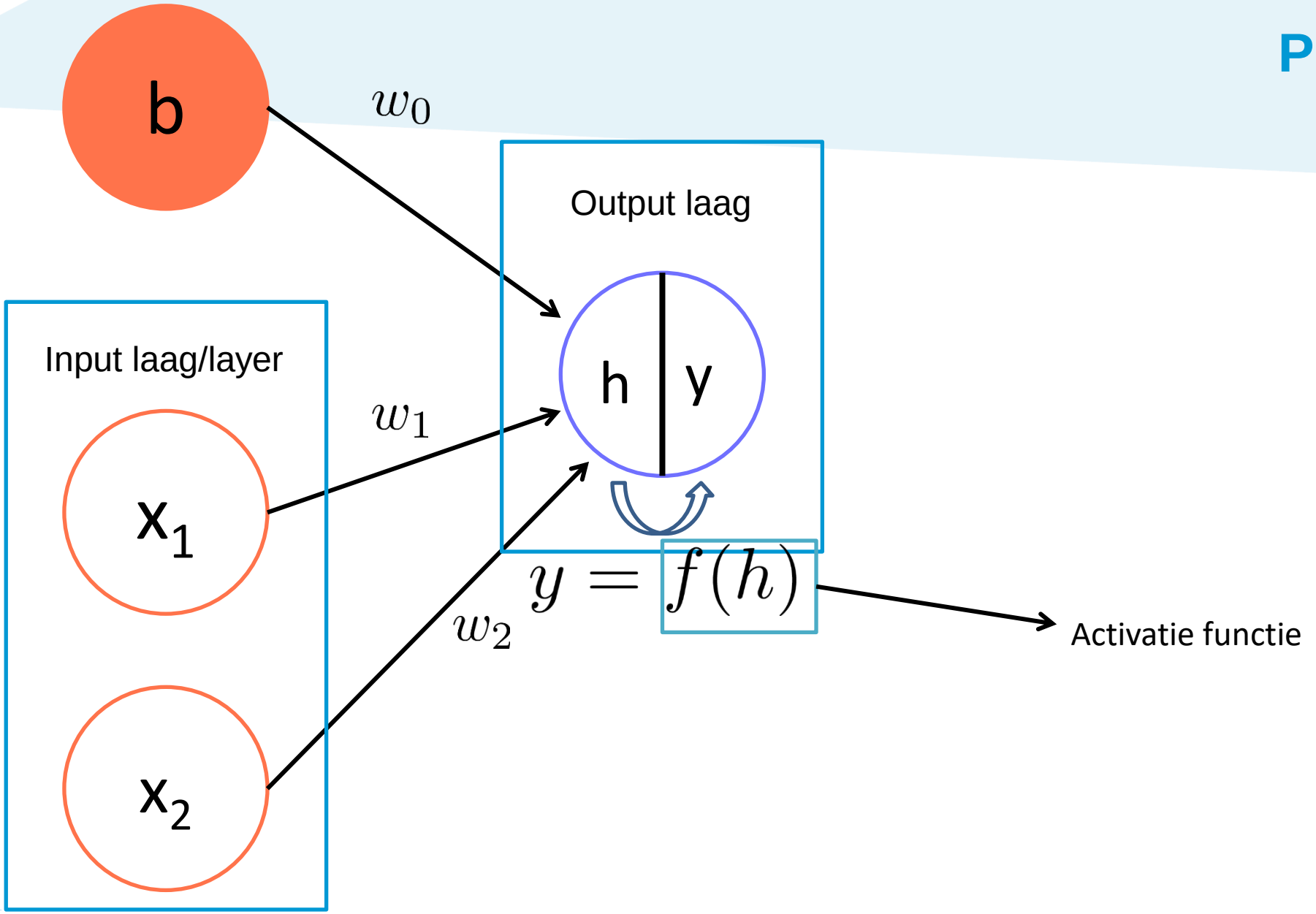
Neuron vuurt niet.
Dus (nog) geen gevaar

Neurale netwerk (basics)



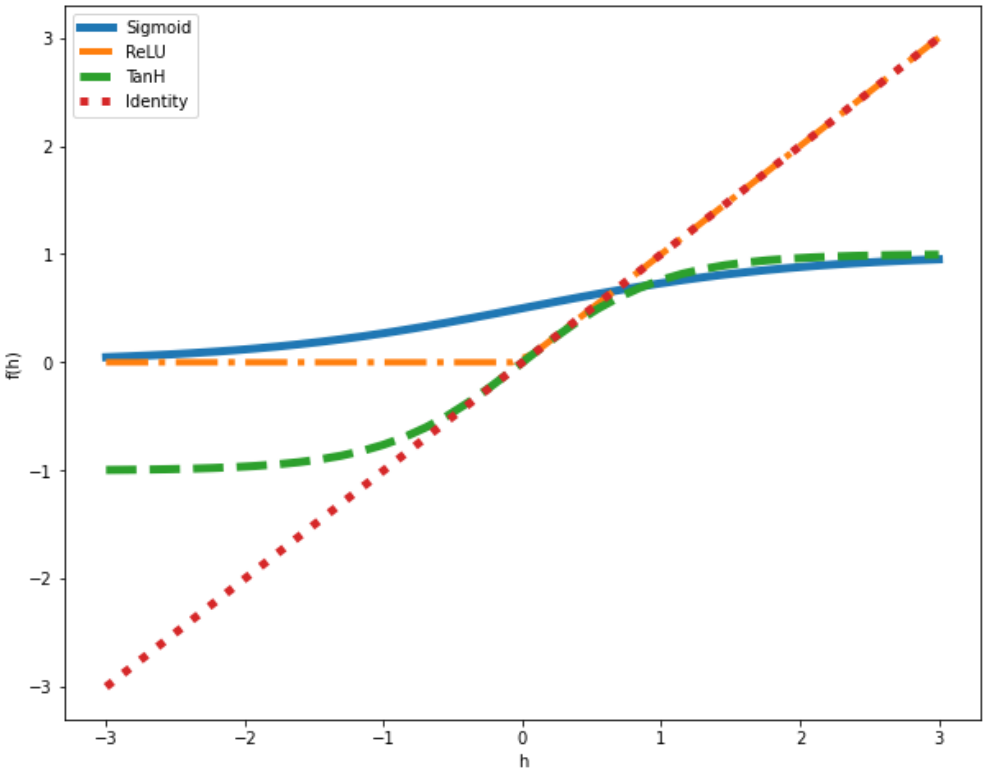


Perceptron

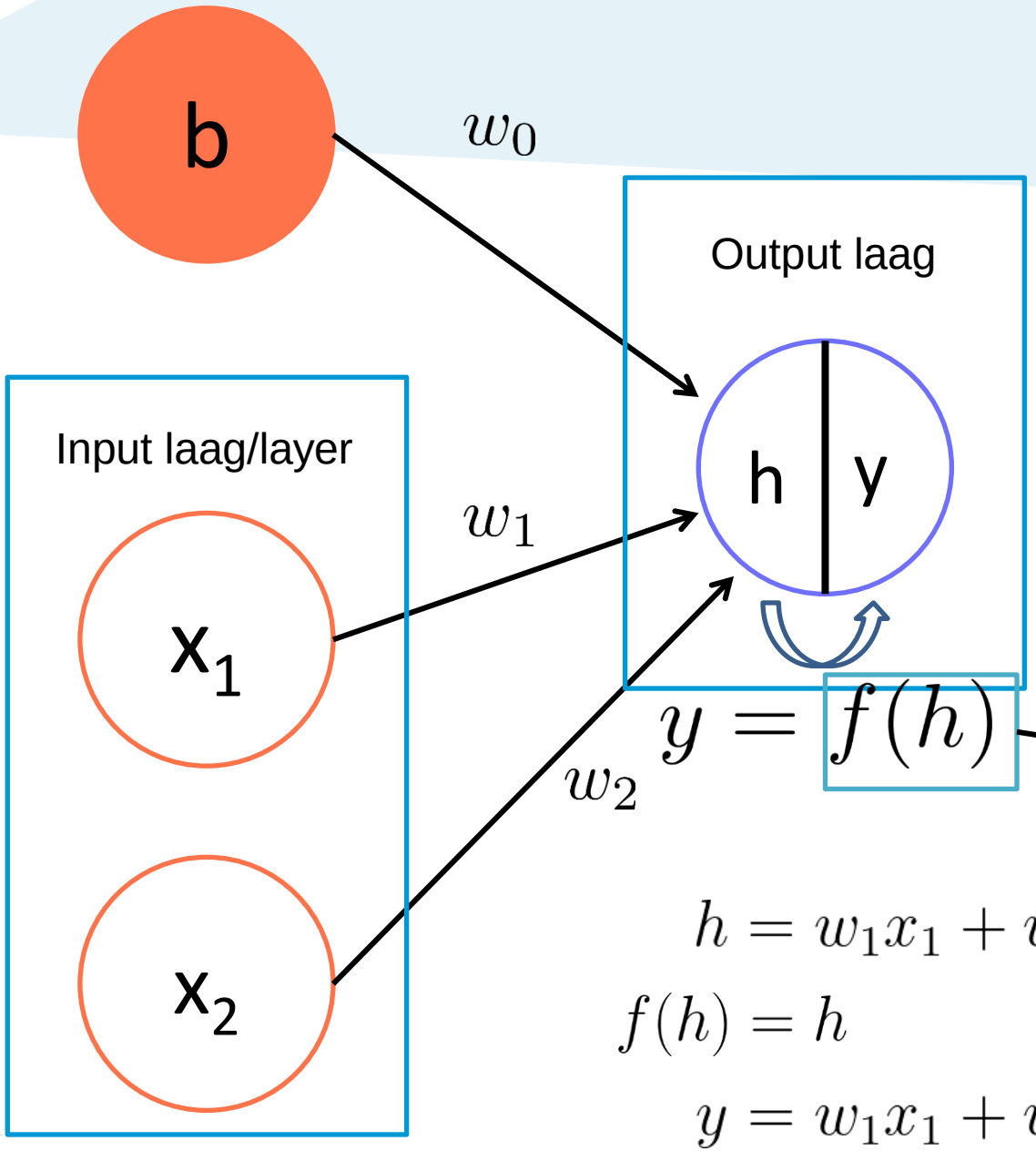

b
w_0
Output laag
h
y
Input laag/layer
w_1
x_1
x_2
w_2
$y = f(h)$
Activatie functie

Activatie functies

Function name	Function
Linear	$f(x) = x$
Sigmoid/Logistic	$f(x) = \frac{1}{1+e^{-x}}$
ReLU	$f(x) = \max(0, x)$
Hyperbolic tangent	$f(x) = \tanh x$
Softmax	$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$



Perceptron

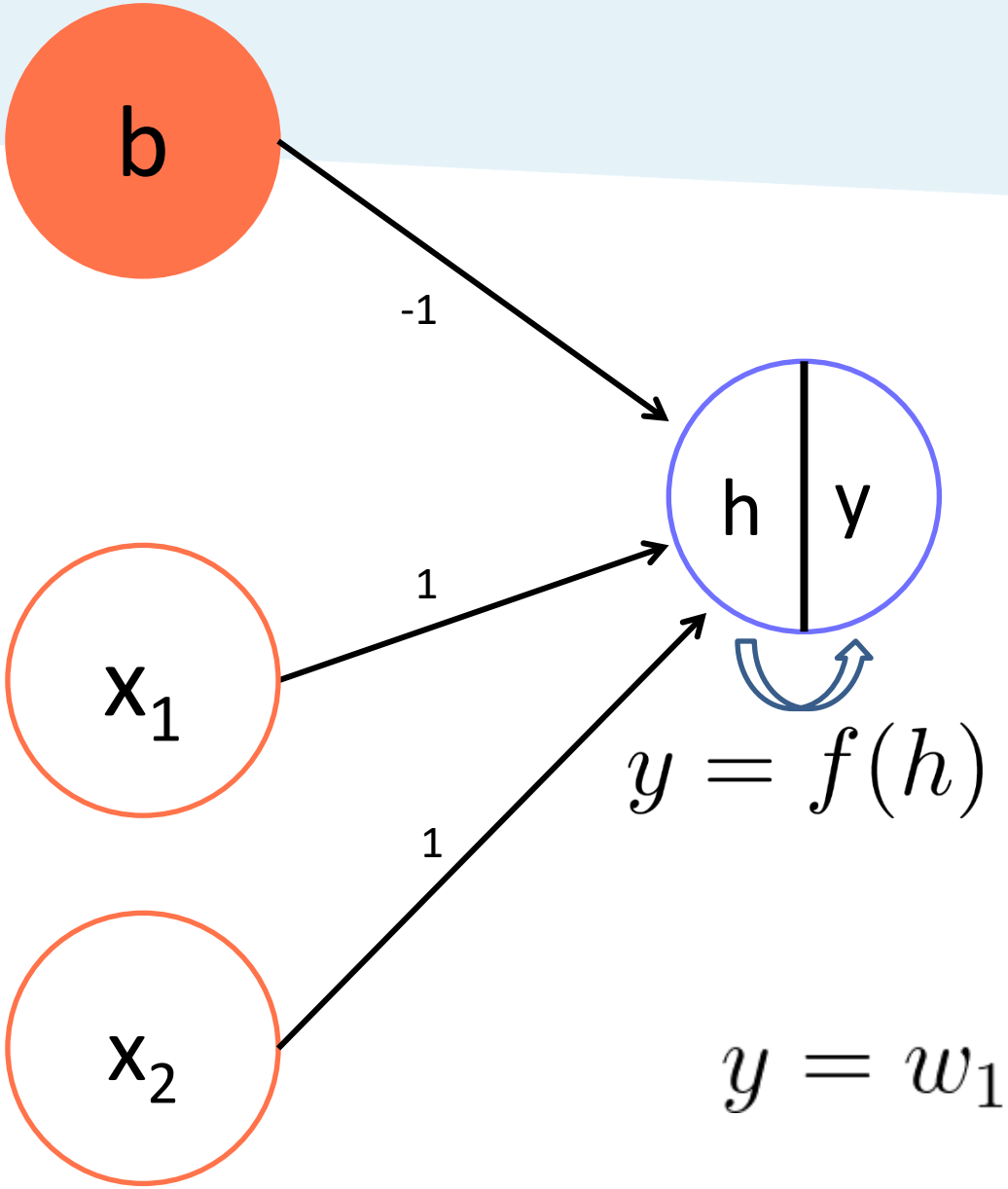


$$y = f\left(w_0 + \sum_{i=1}^{n} x_i w_i\right)$$

$$h = w_1 x_1 + w_2 x_2 + w_0 b$$

$$f(h) = h$$

$$y = w_1 x_1 + w_2 x_2 + w_0 b = w_1 x_1 + w_2 x_2 + w_0$$



$$y = f(h)$$

AND-operator

$$(x_1 = 1) \cap (x_2 = 1) = 1$$
$$(x_1 = 1) \cap (x_2 = 0) = 0$$
$$(x_1 = 0) \cap (x_2 = 1) = 0$$
$$(x_1 = 0) \cap (x_2 = 0) = 0$$

$$y = w_1 x_1 + w_2 x_2 + w_0 b$$

Voorbeeld

$$y = w_1 x_1 + w_2 x_2 + w_0 b$$

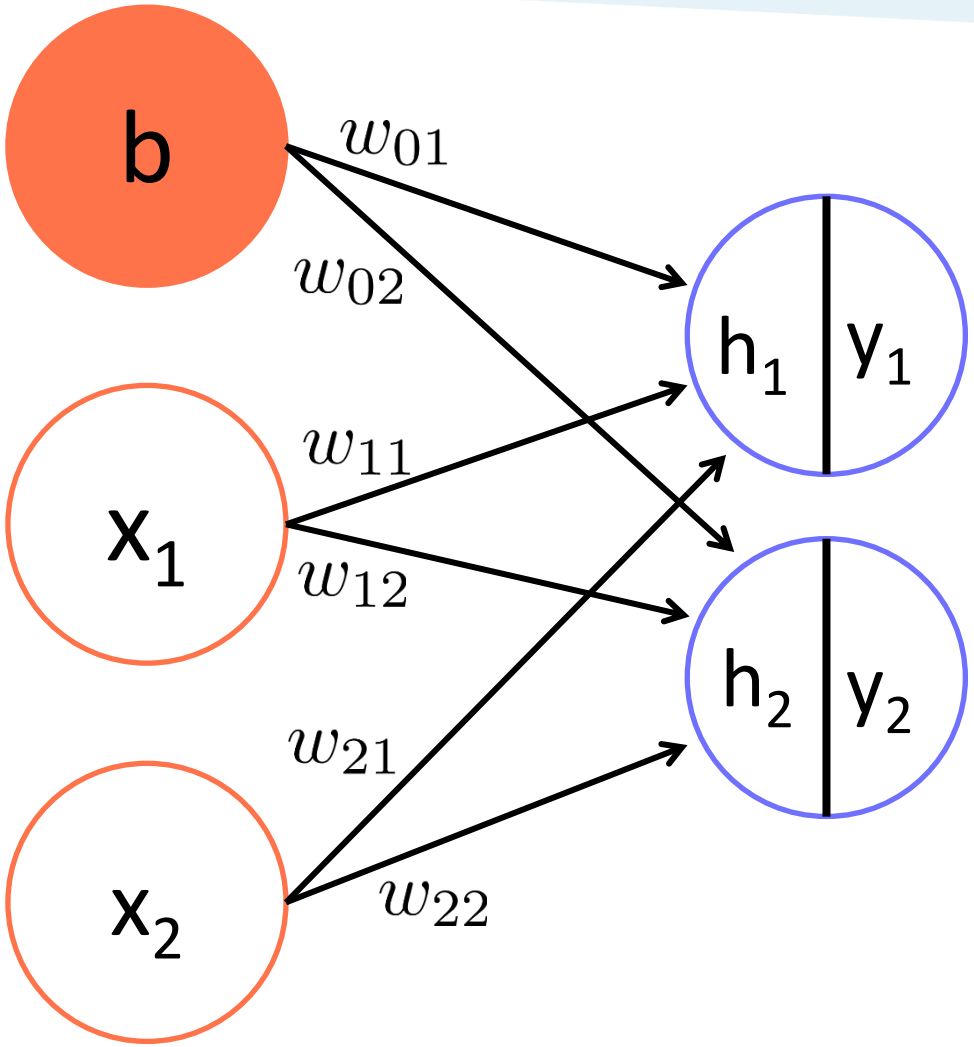
$$y = 1 \cdot x_1 + 1 \cdot x_2 - 1$$
$$1 = 1 \cdot 1 + 1 \cdot 1 - 1$$
$$0 = 1 \cdot 1 + 1 \cdot 0 - 1$$
$$0 = 1 \cdot 0 + 1 \cdot 1 - 1$$
$$-1 = 1 \cdot 0 + 1 \cdot 0 - 1 \sim$$

AND-operator

$$(x_1 = 1) \cap (x_2 = 1) = 1$$
$$(x_1 = 1) \cap (x_2 = 0) = 0$$
$$(x_1 = 0) \cap (x_2 = 0) = 0$$
$$(x_1 = 0) \cap (x_2 = 0) = 0$$

Function name	Function
Linear	$f(x) = x$
Sigmoid/Logistic	$f(x) = \frac{1}{1+e^{-x}}$
ReLU	$f(x) = \max(0, x)$
Hyperbolic tangent	$f(x) = \tanh x$
Softmax	$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$

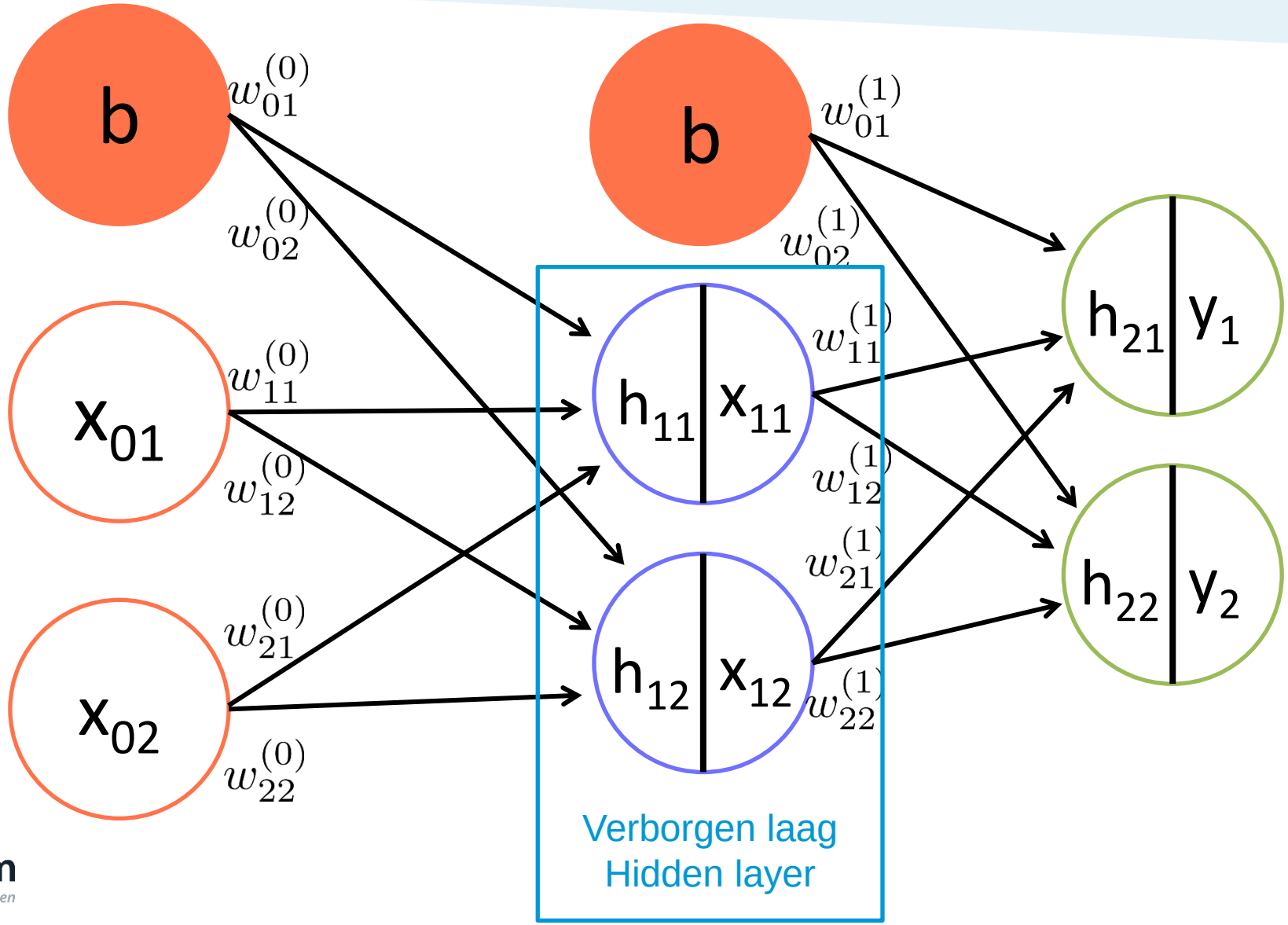
Perceptron



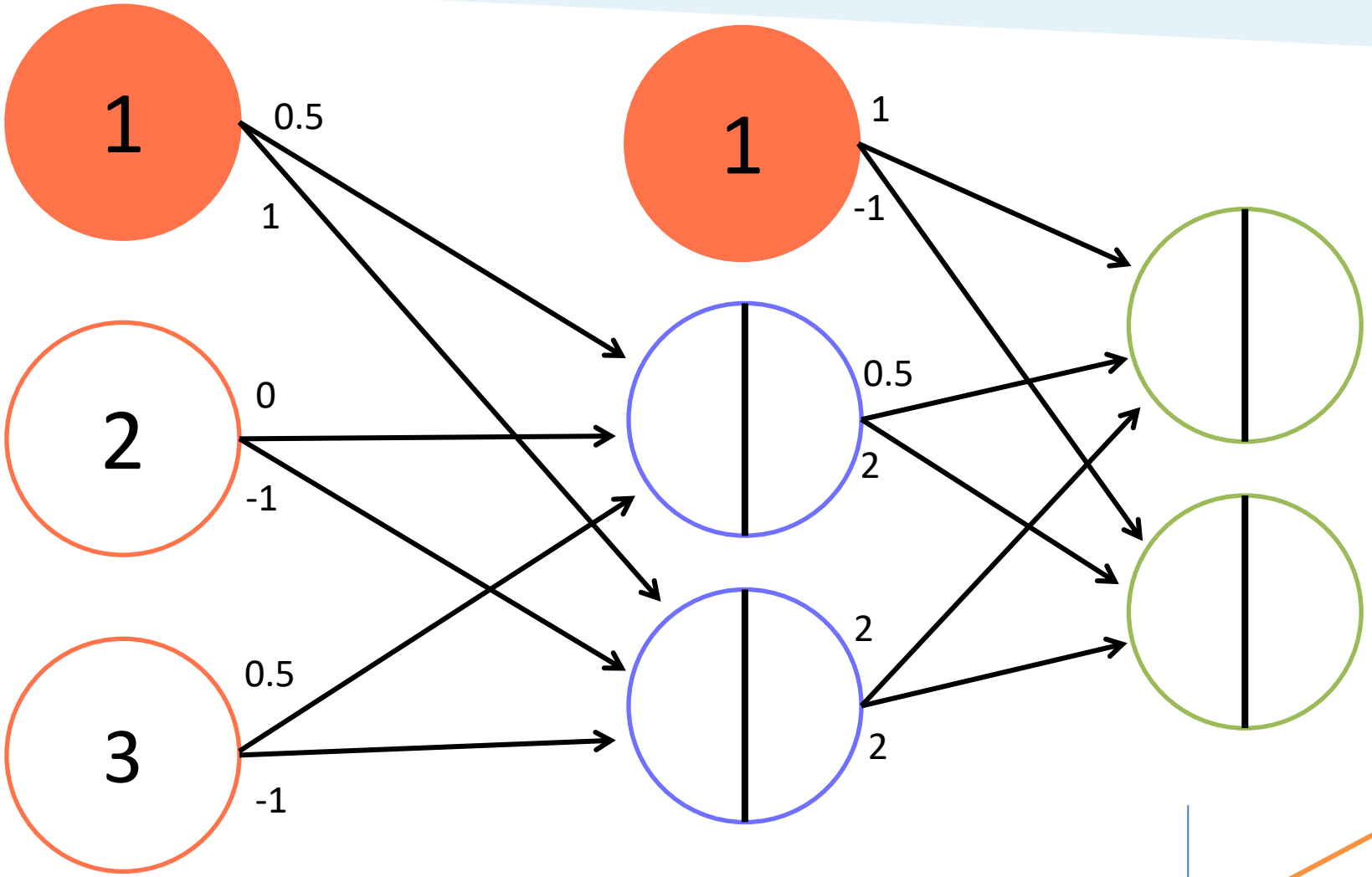
$$y_j = f\left(w_0 + \sum_{i=1}^{n} x_i w_{ij}\right)$$

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

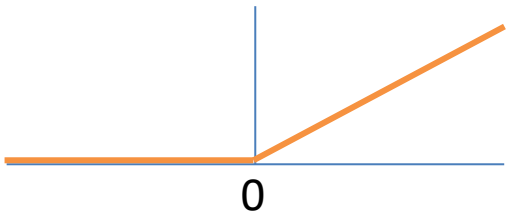
Multi-Layer Perceptron/Neural Network



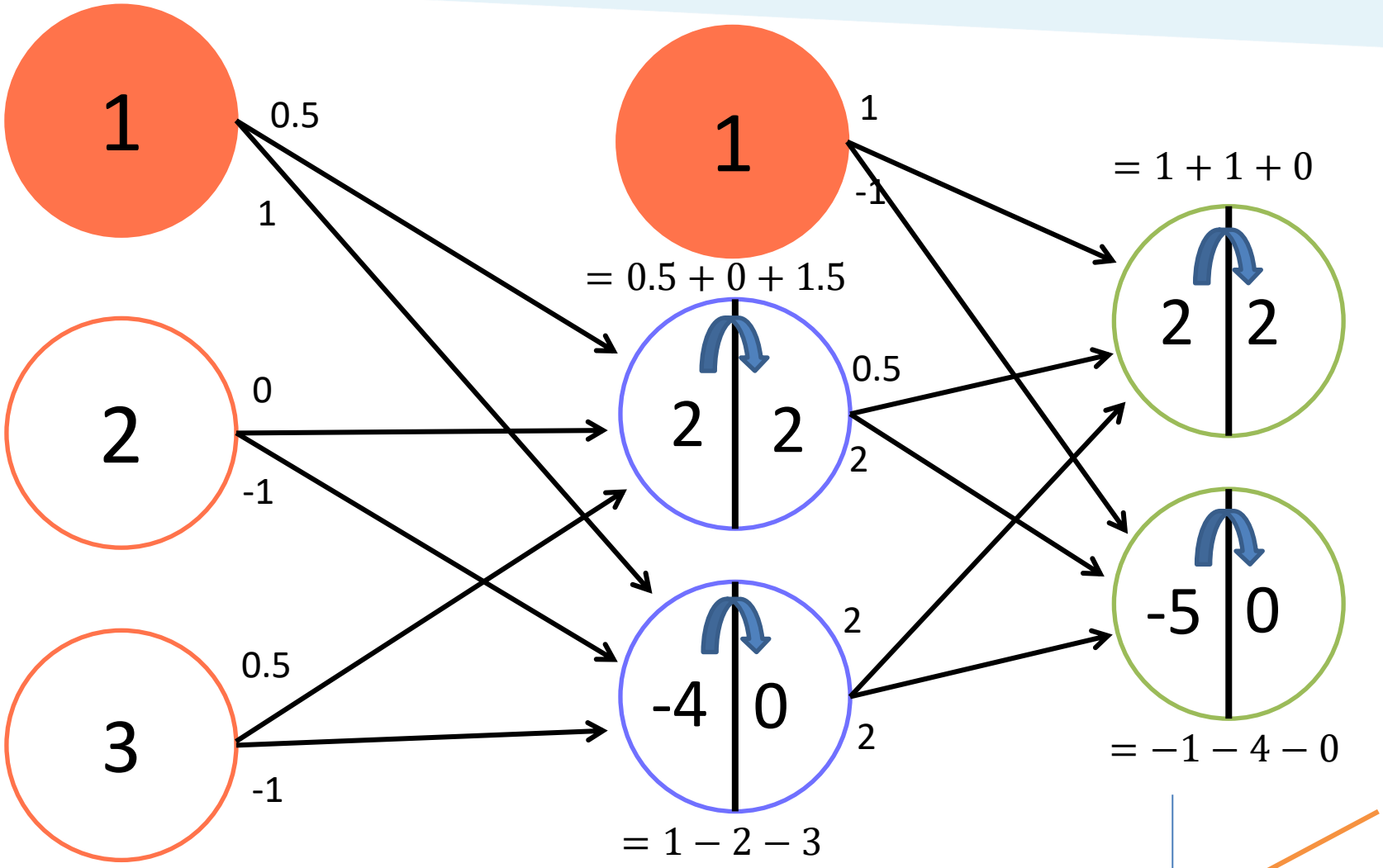
Bereken het neuraal netwerk



$$\text{ReLU} \quad f(x) = \max(0, x)$$



Multi-Layer Perceptron/Neural Network

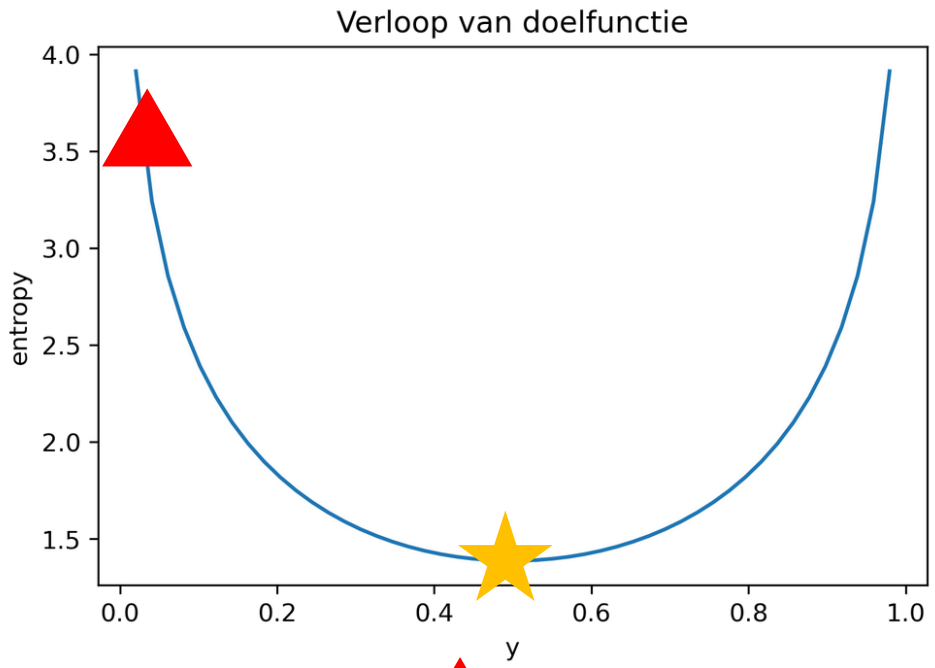


$$\text{ReLU} \quad f(x) = \max(0, x)$$

Hoe leert een Neural Network?

$$\text{MSE} = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

$$\text{Entropy} = -\sum_{i=1}^{n} y_i \log \hat{y}_i$$



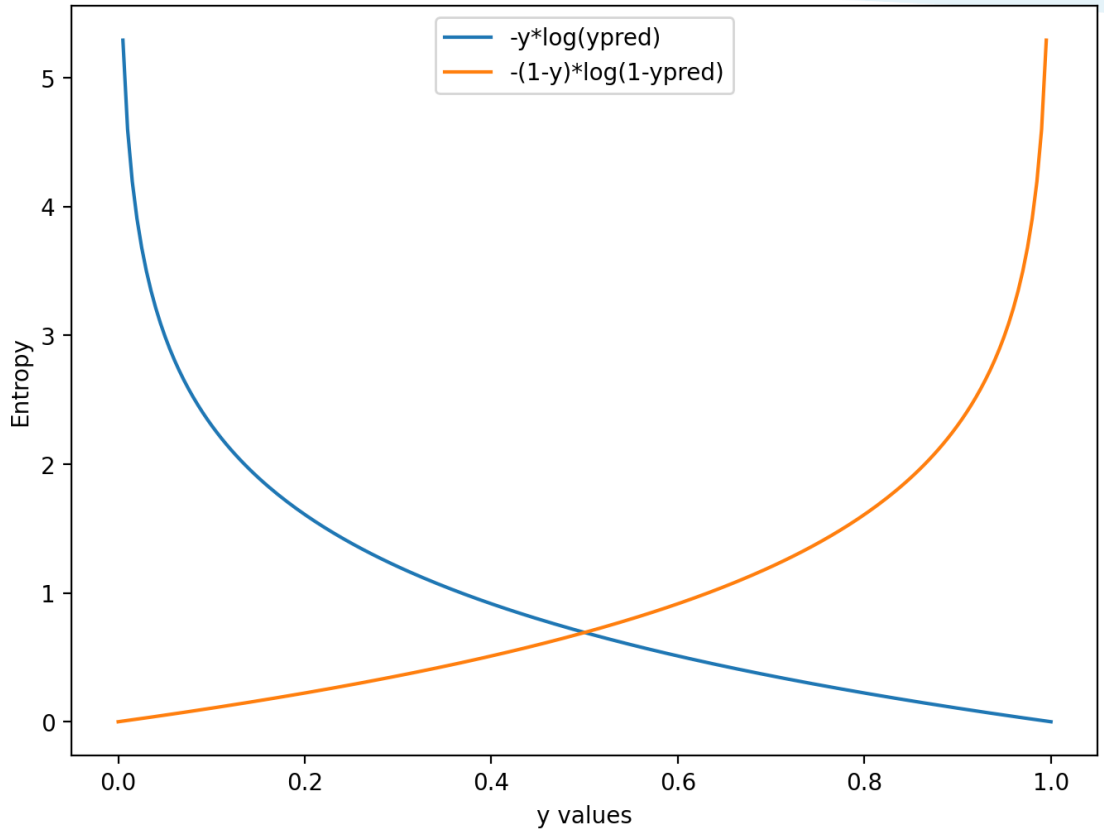
$$\hat{y}_j = f\left(w_0 + \sum_{i=1}^{n} x_i w_i\right)$$

Voor welke $\mathbf{W}$ is de entropy het kleinst?

Start punt

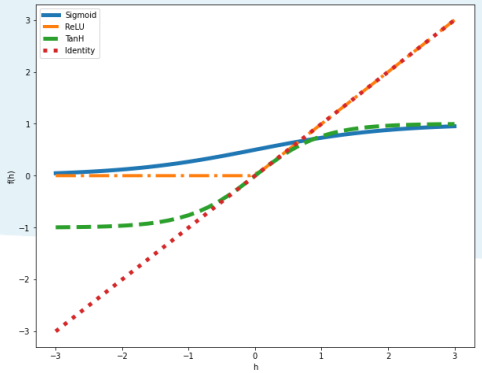
Optimum

Entropy function



$$\text{Entropy (binair)} = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

Hyperparameters



- `activation =` Activation function
- `hidden_layer_sizes = (128, 256, 512)`
  Dit maakt een neuraal netwerk met 3 hidden layers met respectievelijk 128, 256 en 512 neuronen in de laag
- `max_iter =` Maximale hoeveelheid iteraties in de training
- `tol =` Hoeveel het model moet verbeteren om door te gaan met training (tolerantie)
- `learning_rate= "adaptive"` of `learning_rate= "constant"`
  Dit bepaald of de learning rate aangepast wordt als het model minder verbeterd na een paar iteraties
- `initial_learning_rate = 0.001`
  Learning rate die aan het begin wordt gebruikt

Oefening

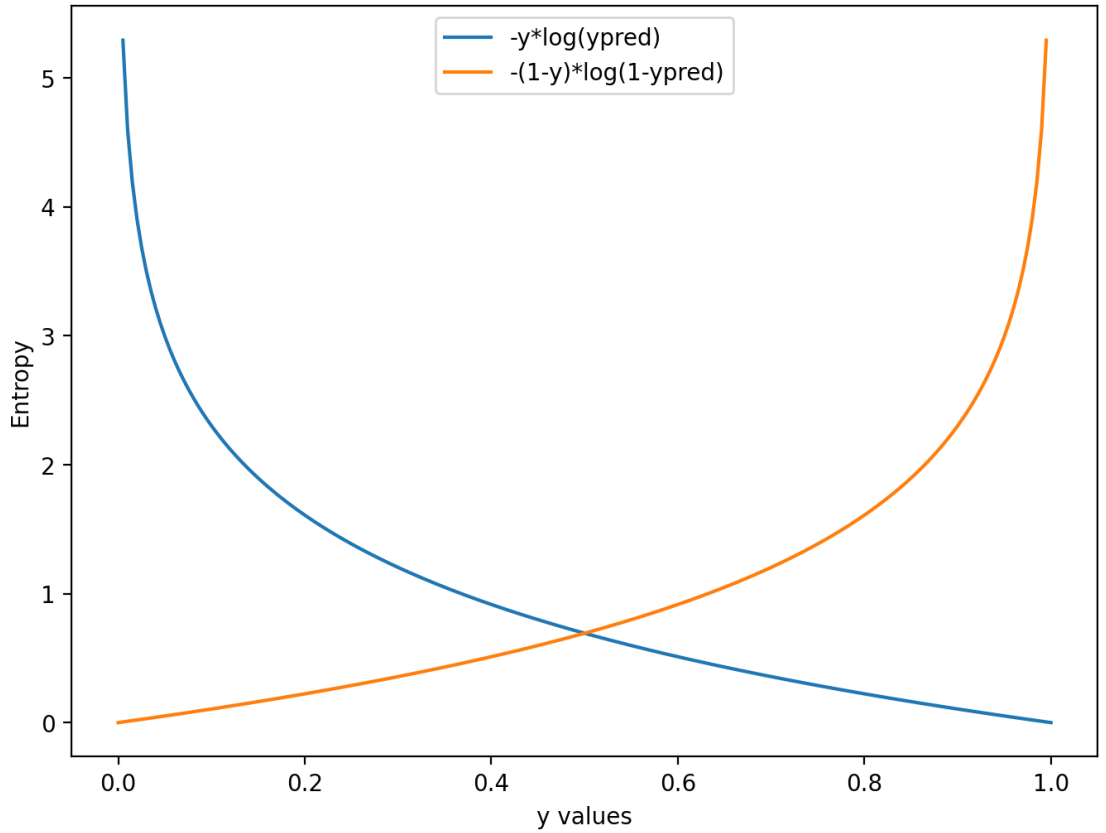
Maak een *Multi Layer Perceptron*/*Neural Network* classifier die voorspelt met welk materiaal is geprint

```
from sklearn.neural_network import MLPClassifier
```

~10 minuten

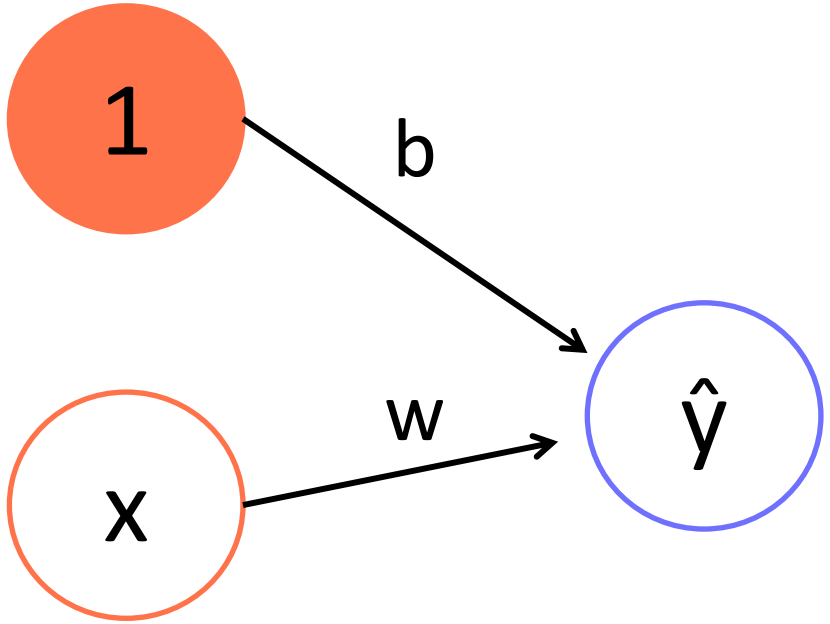
Wat zit achter een Neural Network?

- (Stochastic) Gradient descent
- Wees gewaarschuwd: wiskunde



$$\text{Entropy (binair)} = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

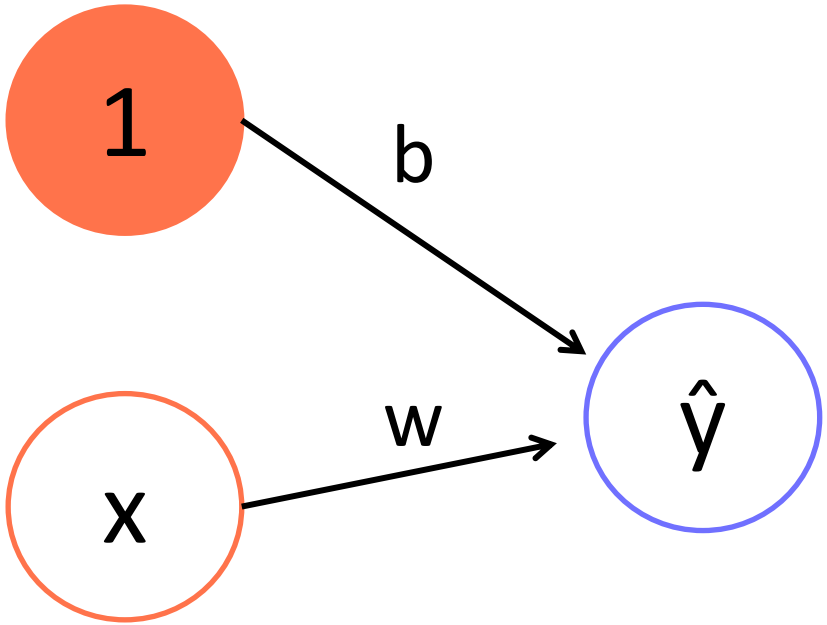
Gradient descend (basis)



$$E(w,b) = -y\log\left(\hat{y}\right) - (1-y)\log\left(1-\hat{y}\right)$$

$$\hat{y} = wx + b$$

Gradient descend (basis)



$$\min_w E(w,b)$$

$$\frac{\partial E(w,b)}{\partial w} = \frac{\partial E(w,b)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w}$$

$$\frac{\partial E(w,b)}{\partial b} = \frac{\partial E(w,b)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b}$$

$$E(w,b) = -y \log(\hat{y}) - (1-y) \log(1-\hat{y})$$

$$\hat{y} = wx + b$$

Gradient descent (basis)

$$E(w,b) = -y\log\left(\hat{y}\right) - (1-y)\log\left(1-\hat{y}\right)$$

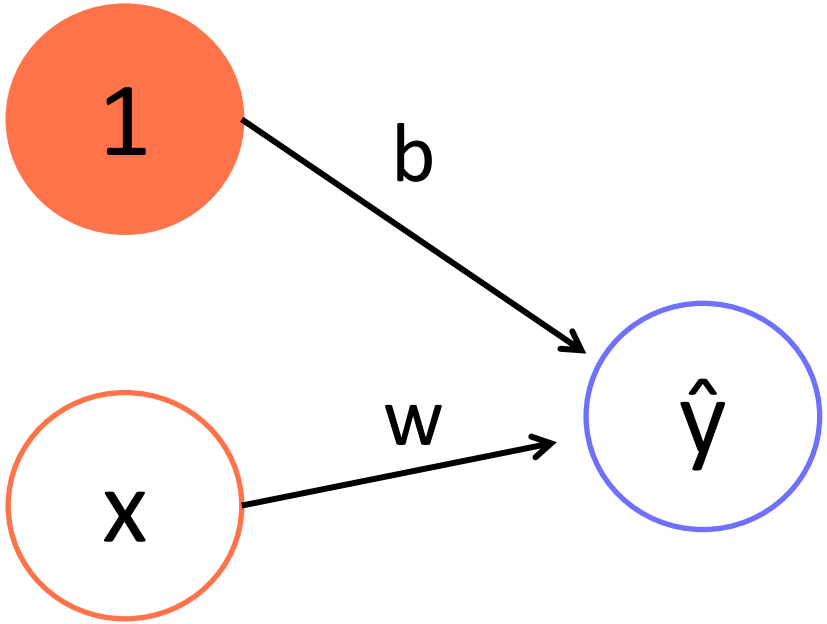
$$\hat{y} = wx + b$$

$$\frac{d\log x}{dx} = \frac{1}{x}$$

$$\frac{\partial E(w,b)}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$$

$$\frac{\partial \hat{y}}{\partial w} = ? \qquad \frac{\partial \hat{y}}{\partial b} = ?$$

Gradient descend (basis)



$$E(w,b) = -y\log\left(\hat{y}\right) - (1-y)\log\left(1-\hat{y}\right)$$

$$\hat{y} = wx + b$$

$$\min_{w} E(w,b)$$

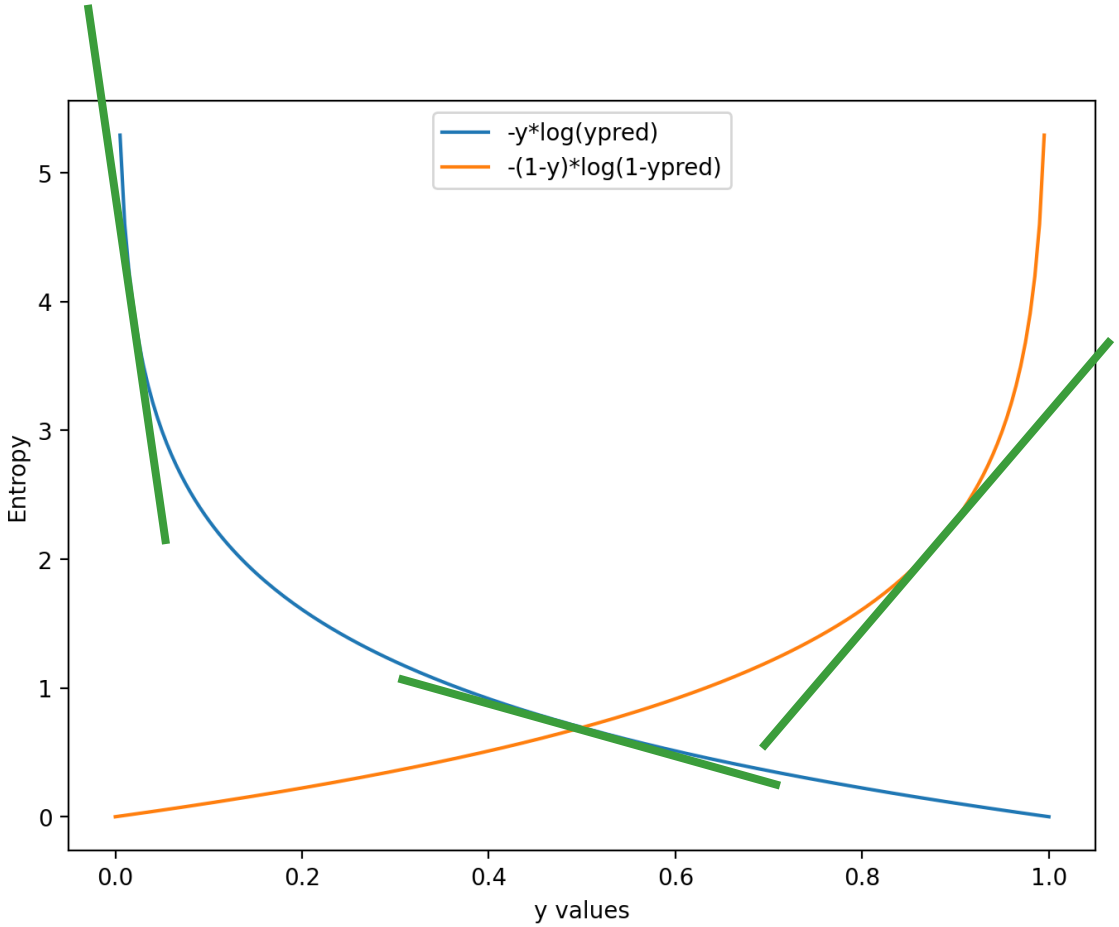
$$\frac{\partial E(w,b)}{\partial w} = \frac{\partial E(w,b)}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial w}$$

$$= \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right)x$$

$$\frac{\partial E(w,b)}{\partial b} = \frac{\partial E(w,b)}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial b}$$

$$= \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right)$$

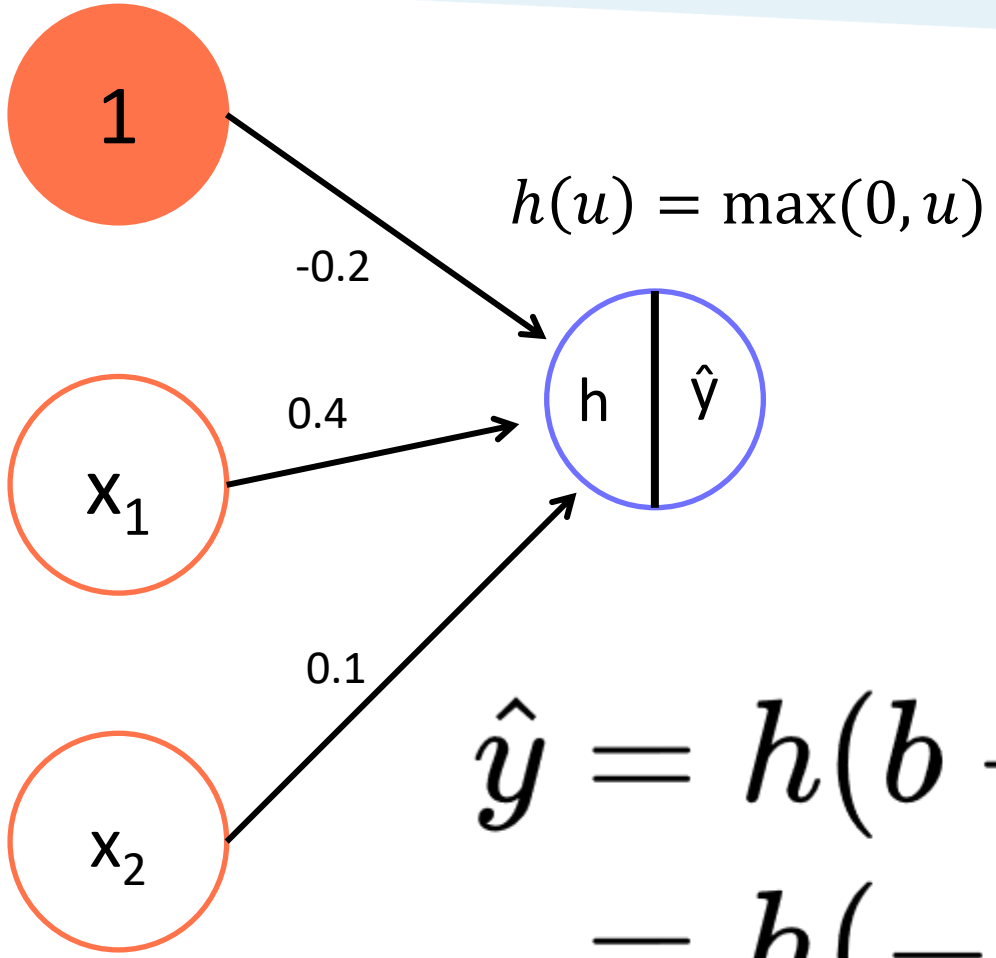
Update gradient descend



$$w^+ = w - \alpha \frac{\partial E(w,b)}{\partial w}$$

$$b^+ = b - \alpha \frac{\partial E(w,b)}{\partial b}$$

AND operator trainen



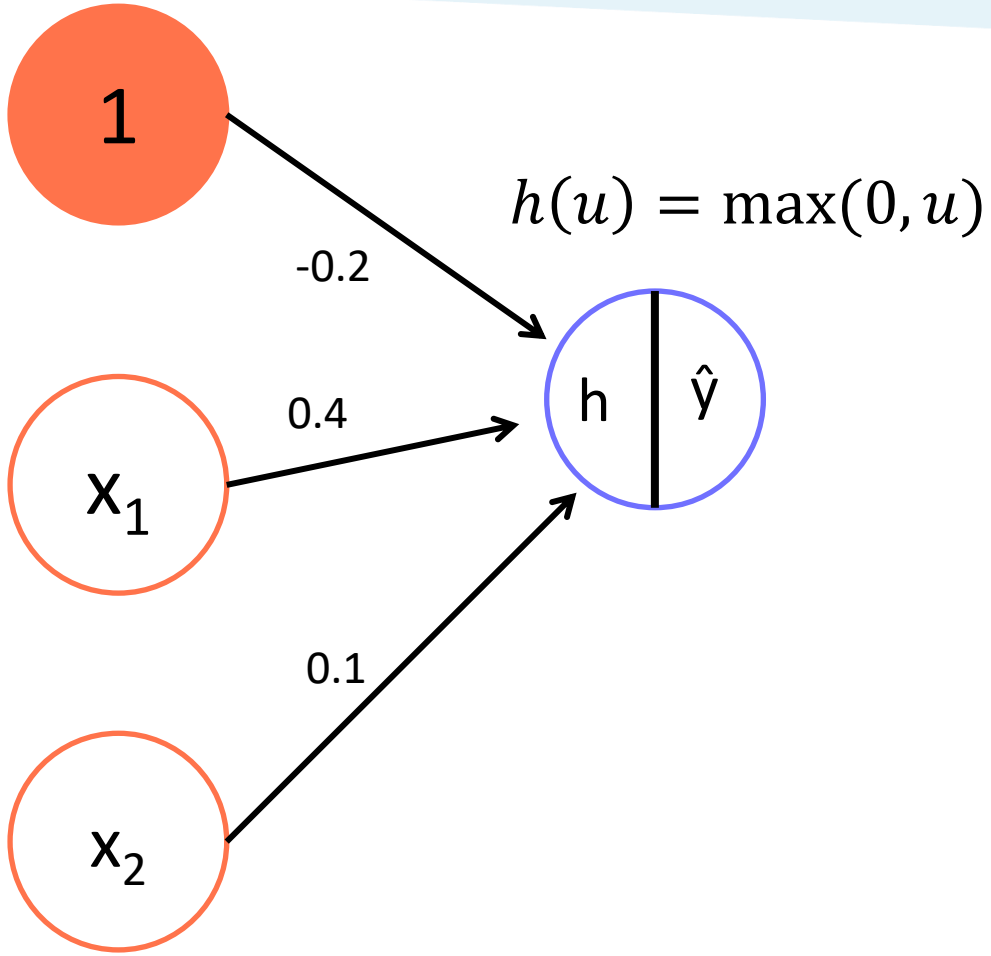
$$h(u) = \max(0, u)$$

AND-operator

$(x_1 = 1) \cap (x_2 = 1) = 1$
$(x_1 = 1) \cap (x_2 = 0) = 0$
$(x_1 = 0) \cap (x_2 = 1) = 0$
$(x_1 = 0) \cap (x_2 = 0) = 0$

$$\hat{y} = h(b + w_1 x_1 + w_2 x_2)$$
$$= h(-0.2 + 0.4x_1 + 0.1x_2)$$

AND operator trainen



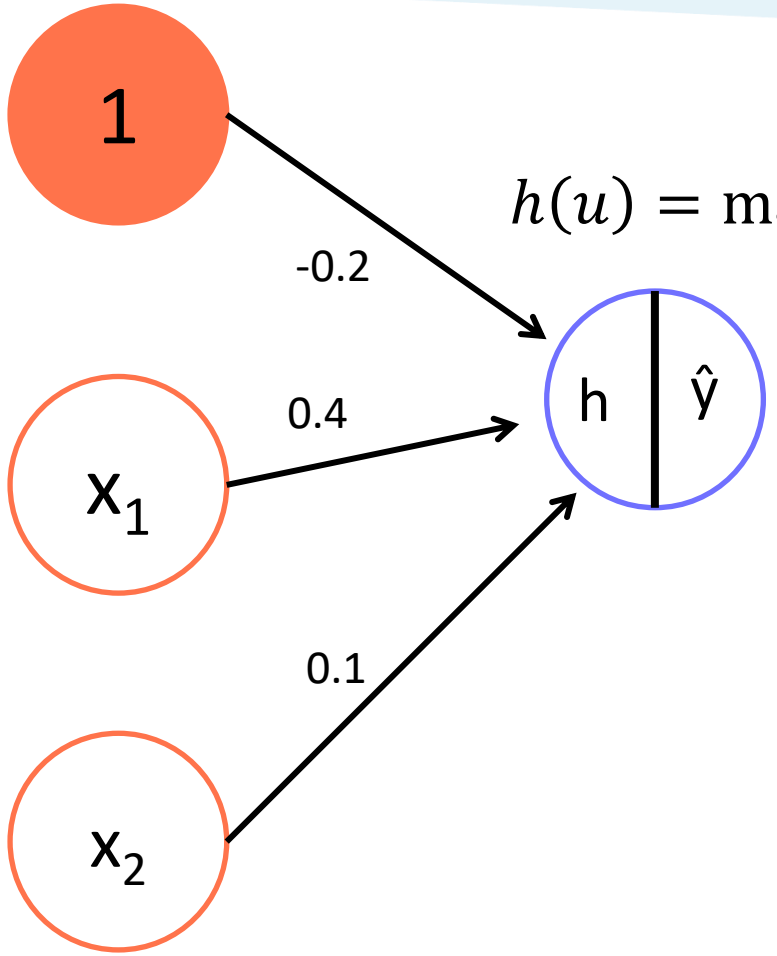
$$\hat{y} = h(b + w_1 x_1 + w_2 x_2)$$
$$= h(-0.2 + 0.4x_1 + 0.1x_2)$$

$$\min_w E(w, b)$$

$$\frac{\partial E(w,b)}{\partial w} = \frac{\partial E(w,b)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{\partial w}$$

$$\frac{\partial E(w,b)}{\partial b} = \frac{\partial E(w,b)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h} \frac{\partial h}{\partial b}$$

AND operator trainen



$$h(u) = \max(0, u)$$

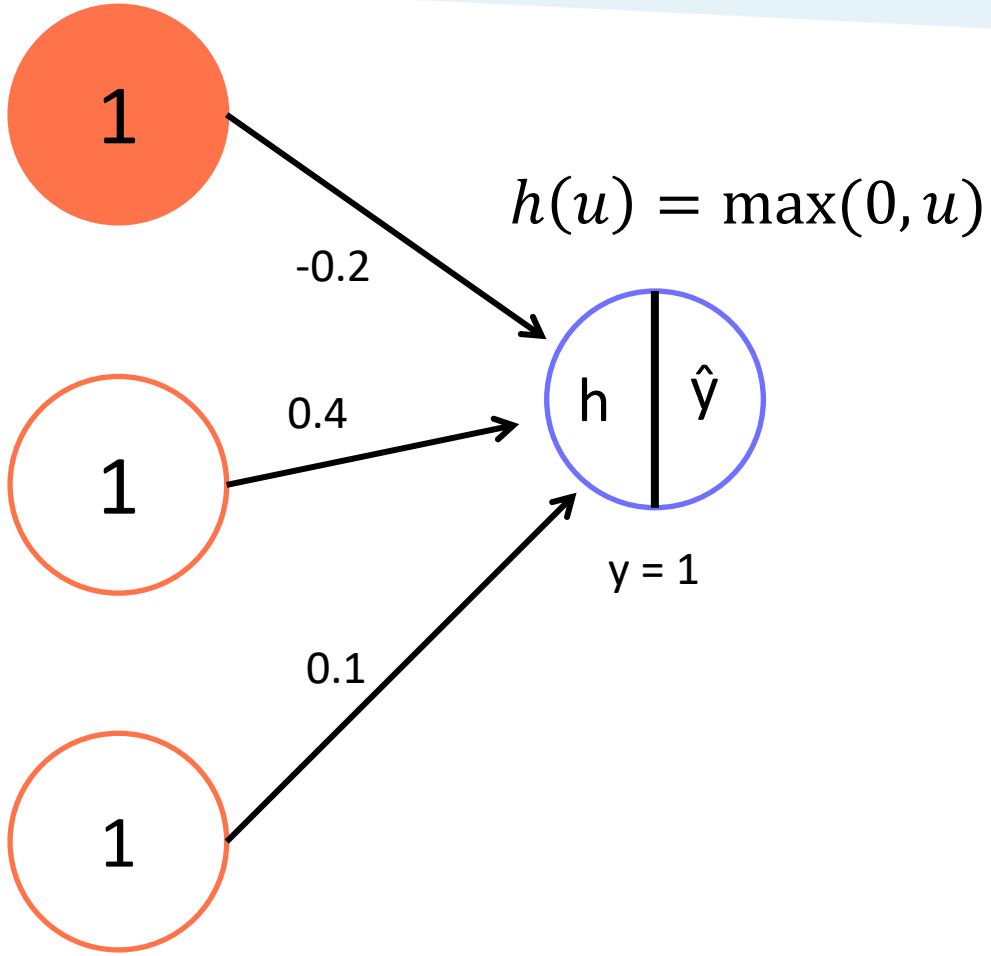
$$\begin{aligned}\hat{y} &= h(b + w_1 x_1 + w_2 x_2) \\ &= h(-0.2 + 0.4x_1 + 0.1x_2)\end{aligned}$$

$$\frac{\partial E(w,b)}{\partial w} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b)x$$

$$\frac{\partial E(w,b)}{\partial b} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b)$$

$$h'(x,w,b) = \begin{cases} 1 & \text{als } b + w_1 x_1 + w_2 x_2 > 0 \\ 0 & \text{anders} \end{cases}$$

AND operator trainen



$$h(u) = \max(0, u)$$

$$\begin{aligned}\hat{y} &= h(b + w_1 x_1 + w_2 x_2) \\ &= h(-0.2 + 0.4x_1 + 0.1x_2)\end{aligned}$$

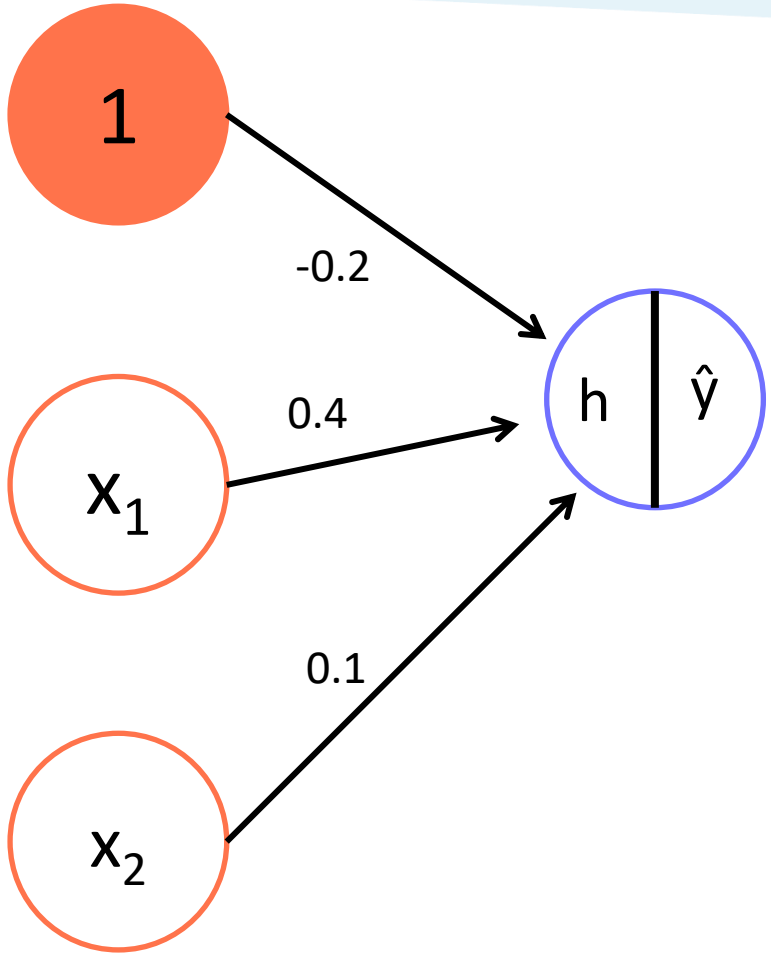
$$\frac{\partial E(w,b)}{\partial w} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b)x$$

$$\frac{\partial E(w,b)}{\partial b} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b)$$

$$h'(x,w,b) = \begin{cases} 1 & \text{als } b + w_1 x_1 + w_2 x_2 > 0 \\ 0 & \text{anders} \end{cases}$$

Opdracht: bereken de gradienten

AND operator trainen



$$h(u) = \max(0, u)$$

AND-operator

$$(x_1 = 1) \cap (x_2 = 1) = 1$$
$$(x_1 = 1) \cap (x_2 = 0) = 0$$
$$(x_1 = 0) \cap (x_2 = 1) = 0$$
$$(x_1 = 0) \cap (x_2 = 0) = 0$$

$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 1) = 0.3$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 0) = 0.2$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 1) = 0$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 0) = 0$$

AND operator trainen

$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 1) = 0.3$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 0) = 0.2$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 1) = 0$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 0) = 0$$

AND-operator

$(x_1 = 1) \cap (x_2 = 1) = 1$
$(x_1 = 1) \cap (x_2 = 0) = 0$
$(x_1 = 0) \cap (x_2 = 1) = 0$
$(x_1 = 0) \cap (x_2 = 0) = 0$

$$\frac{\partial E(w,b)}{\partial w_1} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) x_1 = \frac{-1}{0.3} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial w_2} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) x_2 = \frac{-1}{0.3} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial b} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) = \frac{-1}{0.3} \cdot 1$$

$$h'(x,w,b) = \begin{cases} 1 & \text{als } b + w_1 x_1 + w_2 x_2 > 0 \\ 0 & \text{anders} \end{cases}$$

mikrocentrum
opleiden • ontmoeten • ondernemen

AND operator trainen

$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 1) = 0.3$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 0) = 0.2$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 1) = 0$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 0) = 0$$

AND-operator

$(x_1 = 1) \cap (x_2 = 1) = 1$
$(x_1 = 1) \cap (x_2 = 0) = 0$
$(x_1 = 0) \cap (x_2 = 1) = 0$
$(x_1 = 0) \cap (x_2 = 0) = 0$

$$\frac{\partial E(w,b)}{\partial w_1} = \frac{1}{0.8} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial w_2} = \frac{1}{0.8} \cdot 1 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial b} = \frac{1}{0.8} \cdot 1$$

AND operator trainen

$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 1) = 0.3$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 1 + 0.1 \cdot 0) = 0.2$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 1) = 0$$
$$\hat{y} = h(-0.2 + 0.4 \cdot 0 + 0.1 \cdot 0) = 0$$

AND-operator

$$(x_1 = 1) \cap (x_2 = 1) = 1$$
$$(x_1 = 1) \cap (x_2 = 0) = 0$$
$$(x_1 = 0) \cap (x_2 = 1) = 0$$
$$(x_1 = 0) \cap (x_2 = 0) = 0$$

$$\frac{\partial E(w,b)}{\partial w_1} = \frac{1}{1} \cdot 0 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial w_2} = \frac{1}{1} \cdot 0 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial b} = \frac{1}{1} \cdot 0$$

$$\frac{\partial E(w,b)}{\partial w_1} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) x_1 = \frac{-1}{0.3} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial w_2} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) x_2 = \frac{-1}{0.3} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial b} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) h'(x,w,b) = \frac{-1}{0.3} \cdot 1$$

$$\frac{\partial E(w,b)}{\partial w_1} = \frac{1}{0.8} \cdot 1 \cdot 1$$
$$\frac{\partial E(w,b)}{\partial w_2} = \frac{1}{0.8} \cdot 1 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial b} = \frac{1}{0.8} \cdot 1$$

$$\frac{\partial E(w,b)}{\partial w_1} = \frac{1}{1} \cdot 0 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial w_2} = \frac{1}{1} \cdot 0 \cdot 0$$
$$\frac{\partial E(w,b)}{\partial b} = \frac{1}{1} \cdot 0$$

$$\alpha = 0.1$$

$$w_1^+ = w_1 - \alpha(-\frac{1}{0.3} + \frac{1}{0.8}) = 0.6083$$

$$w_2^+ = w_2 - \alpha(-\frac{1}{0.3}) = 0.4333$$

$$b^+ = b - \alpha(-\frac{1}{0.3} + \frac{1}{0.8}) = 0.0083$$

In python

```
epoch = 0
w1 = 0.4
w2 = 0.1
b = -0.2

X1 = np.array([1, 1, 0, 0])
X2 = np.array([1, 0, 1, 0])
Y = X1 & X2

dEdw1 = 0
dEdw2 = 0
dEdb = 0
E = 0
for x1, x2, y in zip(X1, X2, Y):
    h = b + w1*x1 + w2*x2
    yhat = max(0, h)
    E += -np.log(yhat) if y else - np.log(1-yhat)
    dEdy = -1/yhat if y else 1/(1-yhat)
    dEdw1 += dEdy*(h > 0) * x1
    dEdw2 += dEdy*(h > 0) * x2
    dEdb += dEdy*(h > 0)


alpha = 0.1
w1new = w1 - alpha * dEdw1
w2new = w2 - alpha * dEdw2
bnew = b - alpha * dEdb
print(f'Iteration {epoch}: loss = {E:.4f}')
print(w1, w1new)
print(w2, w2new)
print(b, bnew)
```

```
Iteration 0: loss = 1.4271
0.4 0.6083333333333334
0.1 0.43333333333333335
-0.2 0.008333333333333304
```

In python

```
epoch = 0
w1 = 0.4
w2 = 0.1
b = -0.2

X1 = np.array([1, 1, 0, 0])
X2 = np.array([1, 0, 1, 0])
Y = X1 & X2

dEdw1 = 0
dEdw2 = 0
dEdb = 0
E = 0
for x1, x2, y in zip(X1, X2, Y):
    h = b + w1*x1 + w2*x2
    yhat = max(0, h)
    E += -np.log(yhat) if y else - np.log(1-yhat)
    dEdy = -1/yhat if y else 1/(1-yhat)
    dEdw1 += dEdy*(h > 0) * x1
    dEdw2 += dEdy*(h > 0) * x2
    dEdb += dEdy*(h > 0)


alpha = 0.1
w1new = w1 - alpha * dEdw1
w2new = w2 - alpha * dEdw2
bnew = b - alpha * dEdb
print(f'Iteration {epoch}: loss = {E:.4f}')
print(w1, w1new)
print(w2, w2new)
print(b, bnew)
```

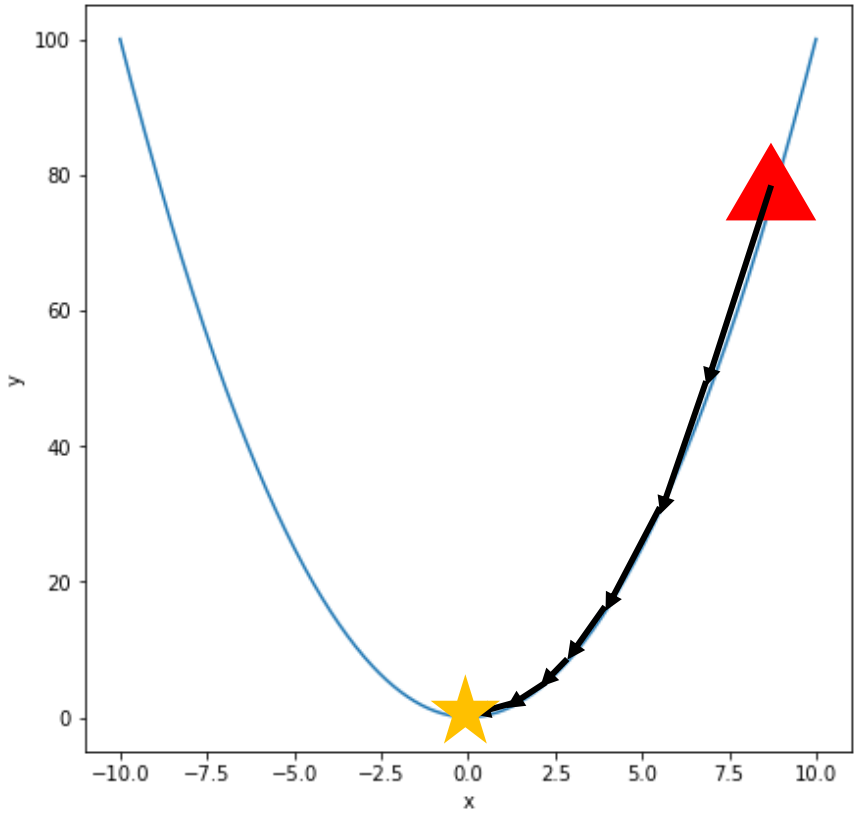
```
Iteration 0: loss = 1.4271
0.4 0.6083333333333334
0.1 0.43333333333333335
-0.2 0.008333333333333304
```

```
epoch += 1
w1 = w1new
w2 = w2new
b =  bnew

dEdw1 = 0
dEdw2 = 0
dEdb = 0
E = 0
for x1, x2, y in zip(X1, X2, Y):
    h = b + w1*x1 + w2*x2
    yhat = max(0, h)
    E += -np.log(yhat) if y else - np.log(1-yhat)
    dEdy = -1/yhat if y else 1/(1-yhat)
    dEdw1 += dEdy*(h > 0) * x1
    dEdw2 += dEdy*(h > 0) * x2
    dEdb += dEdy*(h > 0)


w1new = w1 - alpha * dEdw1
w2new = w2 - alpha * dEdw2
bnew = b - alpha * dEdb
print(f'Iteration {epoch}: loss = {E:.4f}')
print(w1, w1new)
print(w2, w2new)
print(b, bnew)
```

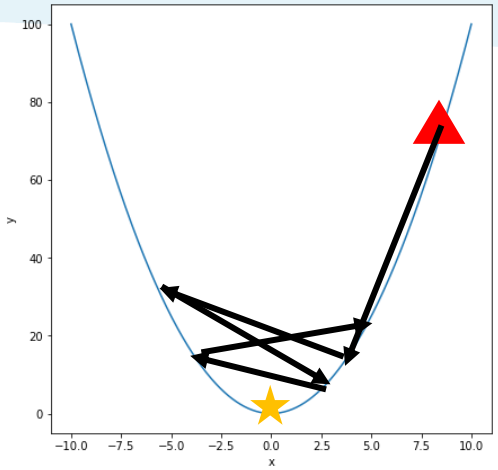
```
Iteration 1: loss = 1.5012
0.6083333333333334 0.4427018633540373
0.43333333333333335 0.34946695095948827
0.008333333333333304 -0.43724295039235694
```

Learning rate

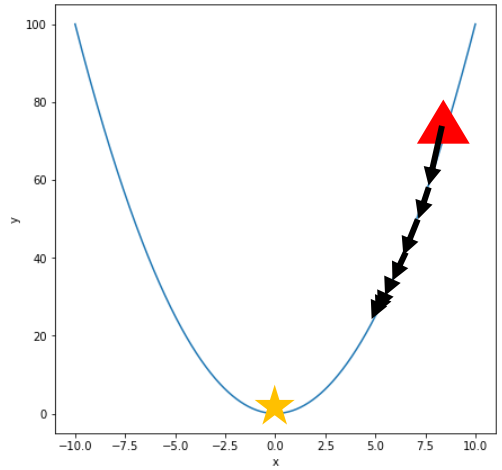
$$LR = 0.001$$



$$LR = 1$$



$$LR = 10^{-8}$$



Start punt

Optimum

Neurale Netwerken

Voordelen

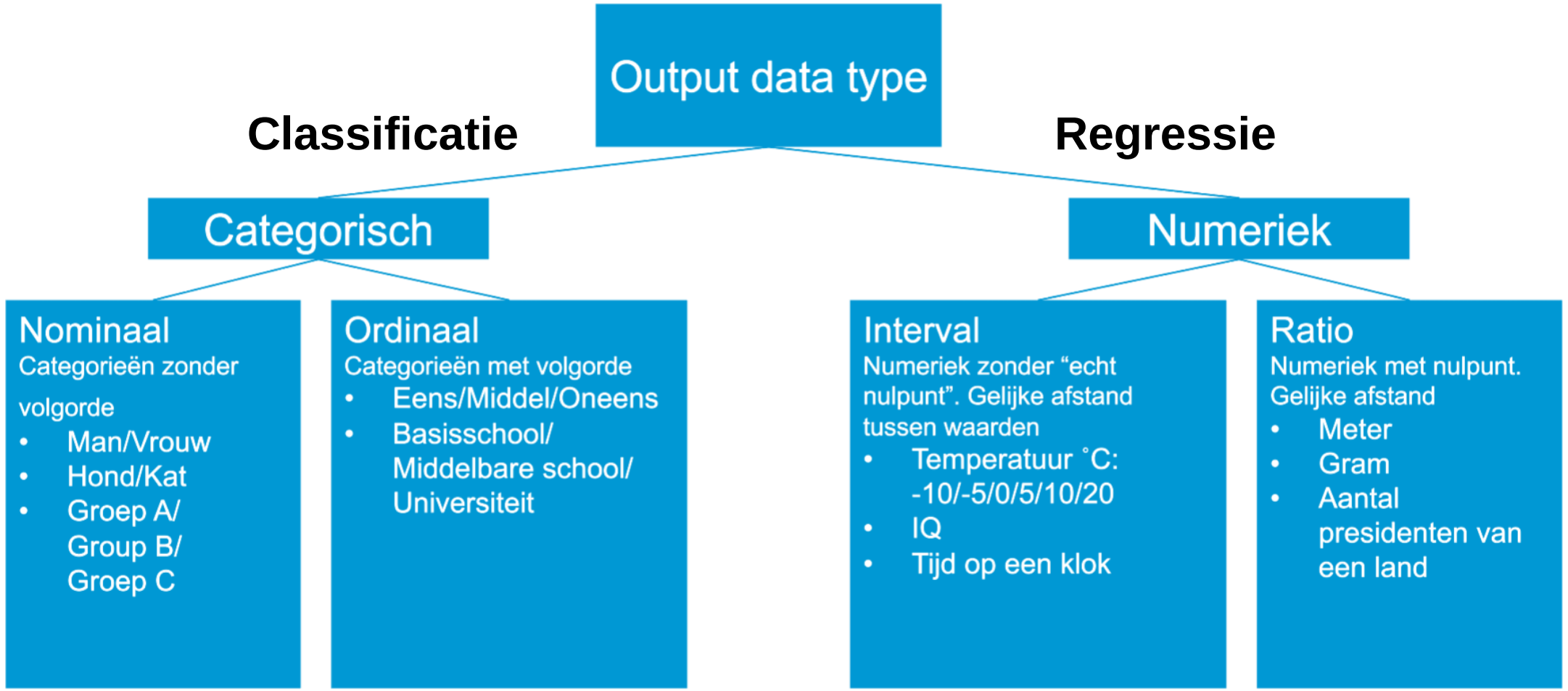
- Goed in het herkennen van complexe patronen
- Generiek voor nieuwe data
- Niet vatbaar voor de dataverdelingen

Nadelen

- Lastig om de hyperparameters te bepalen
- Trainen kan soms erg lang duren
- Gevoelig voor multicollineariteit (komt later in de cursus). Zorgt dat het netwerk minder snel tot een oplossing komt
- Mag door wetgeving niet gebruikt worden met persoonsgegevens

REGRESSIE

Verschil classificatie & regressie



Wat is het verschil tussen regressie en classificatie?

Classificatie

- Voorspellen van groepen/klassen
- Niet opzoek naar de relatie tussen de verschillende variabelen
- Welke variabelen zijn onderscheidend?
- Discreet

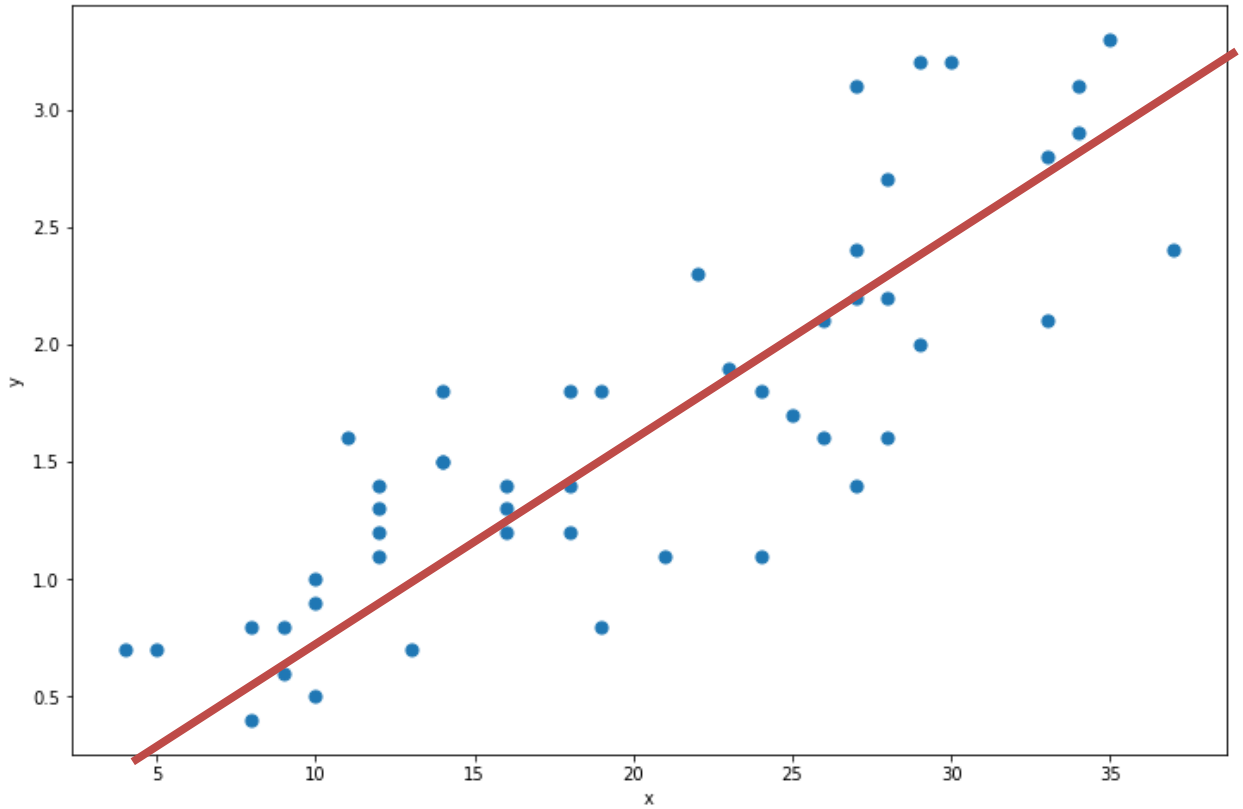
Regressie

- Voorspellen van waardes
- Opzoek naar een relatie tussen de afhankelijk variabele en de voorspeller.
- Welke variabelen zijn verklarend?
- Continue

Voorbeelden: Classificatie of regressie?

- Voor de kwaliteit van een product wordt een score aan gekoppeld van 0 tot 100. Voorspel de kwaliteit.
  Regressie of classificatie?
- Voor de kwaliteitscontrole wordt een product gescand met als uitkomst: Fout, Goed, laten controleren door een expert.
  Regressie of classificatie?
- Er wordt voorspeld of een robot snel vervangen moet worden.
  Regressie of classificatie?

Regressie



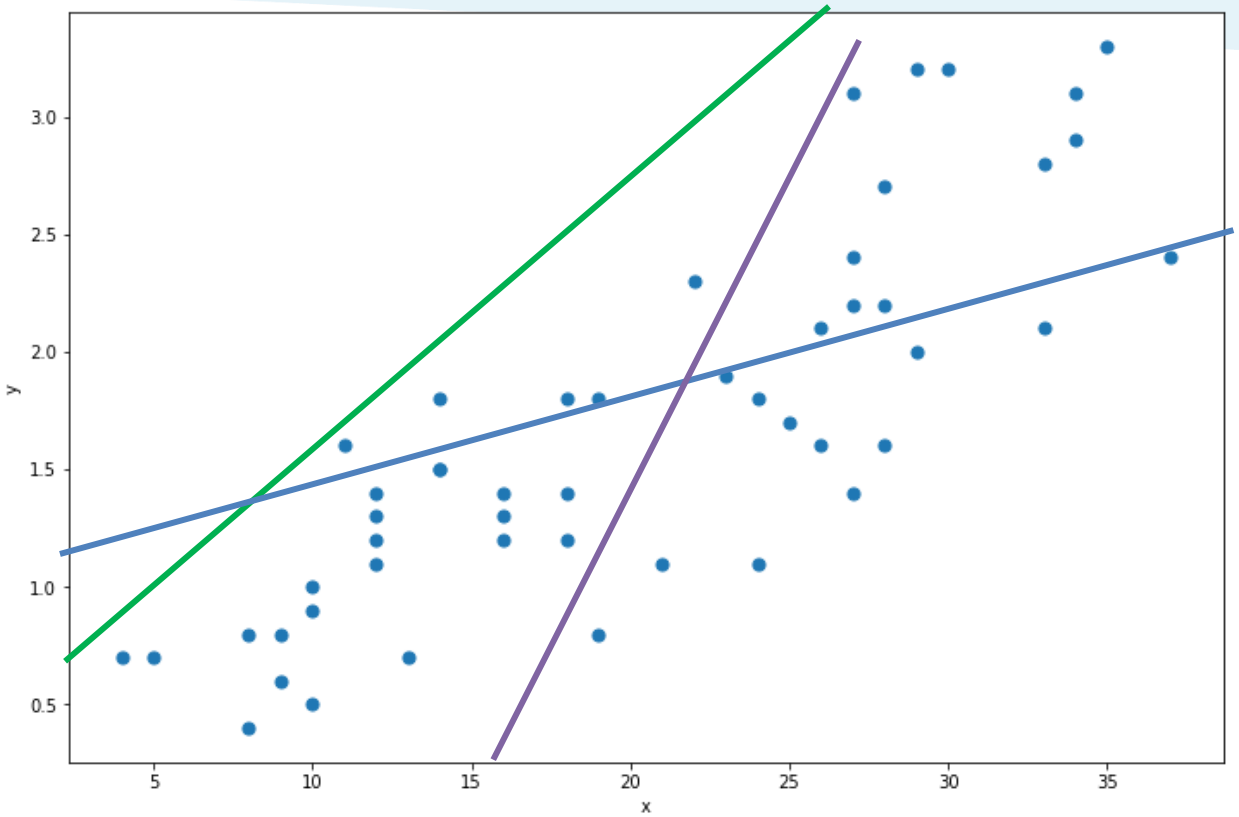
$$\hat{y} = \beta_1 x + \beta_0$$

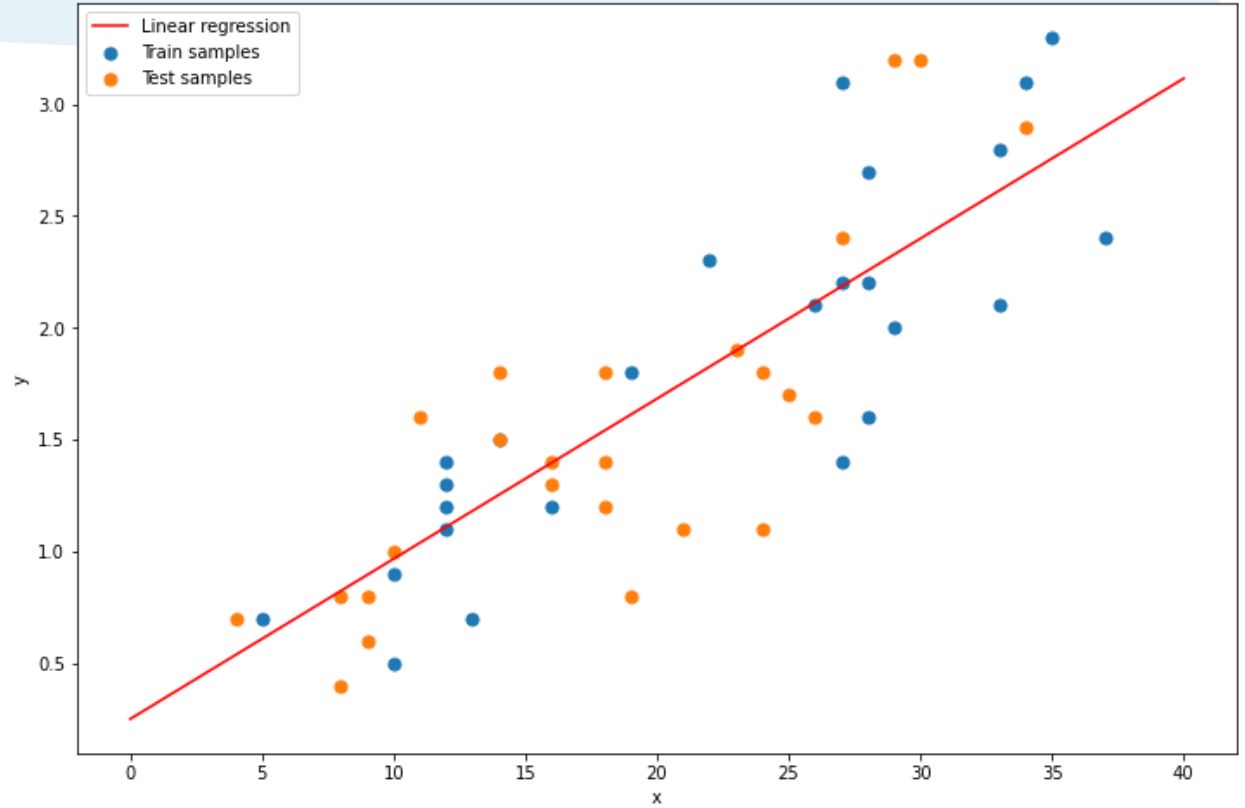
Lineaire Regressie

$$\hat{y} = \beta_2 x_2 + \beta_1 x_1 + \beta_0$$

Multi Lineaire Regressie

Lineaire Regressie



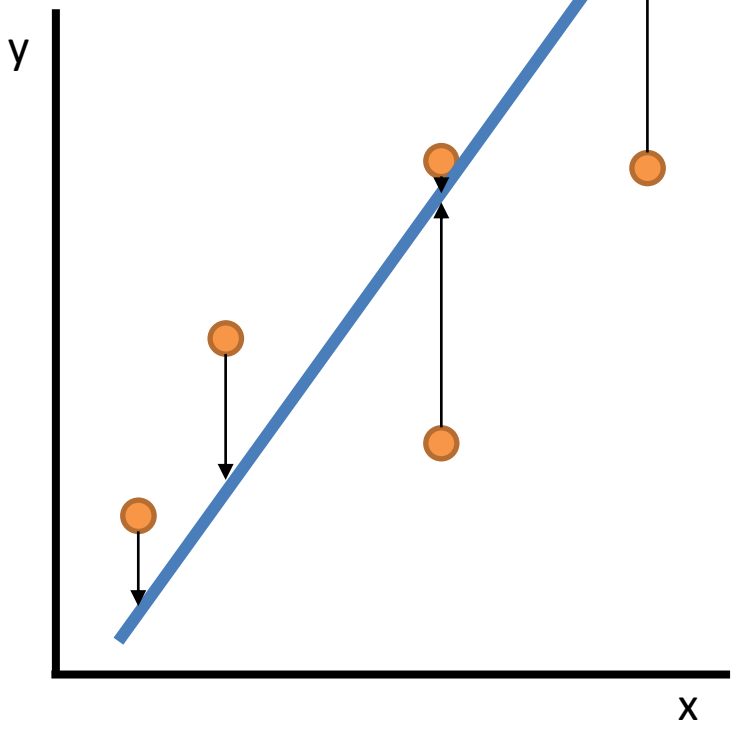


$$\hat{y} = \beta_1 x + \beta_0$$

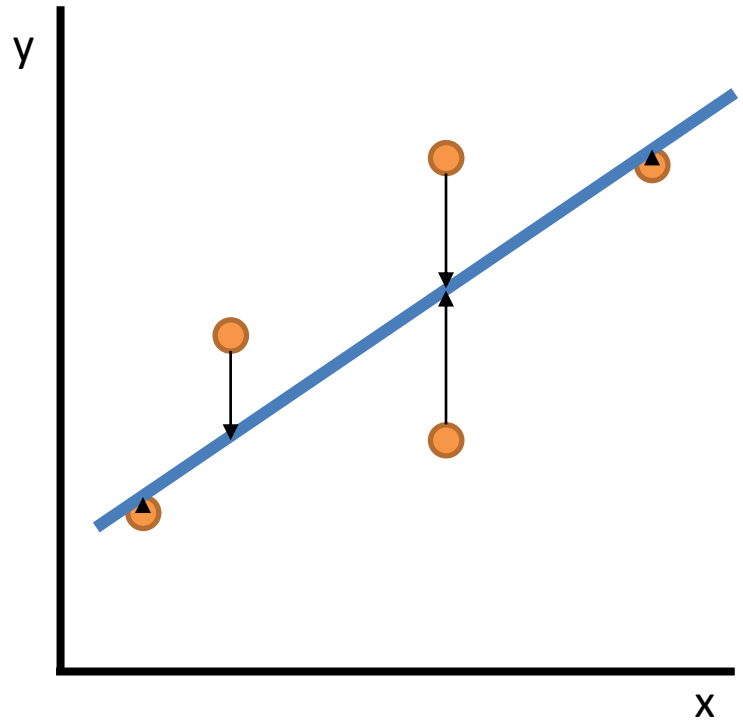
$$\min_{\beta} (y - \hat{y})^2$$

Least squares method

A



B



Welke is beter?

Metrics

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

Wat is de gemiddelde kwadratische afstand?
Range tussen 0 en oneindig

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

Hoeveel van de spreiding kan verklaard worden door het model?

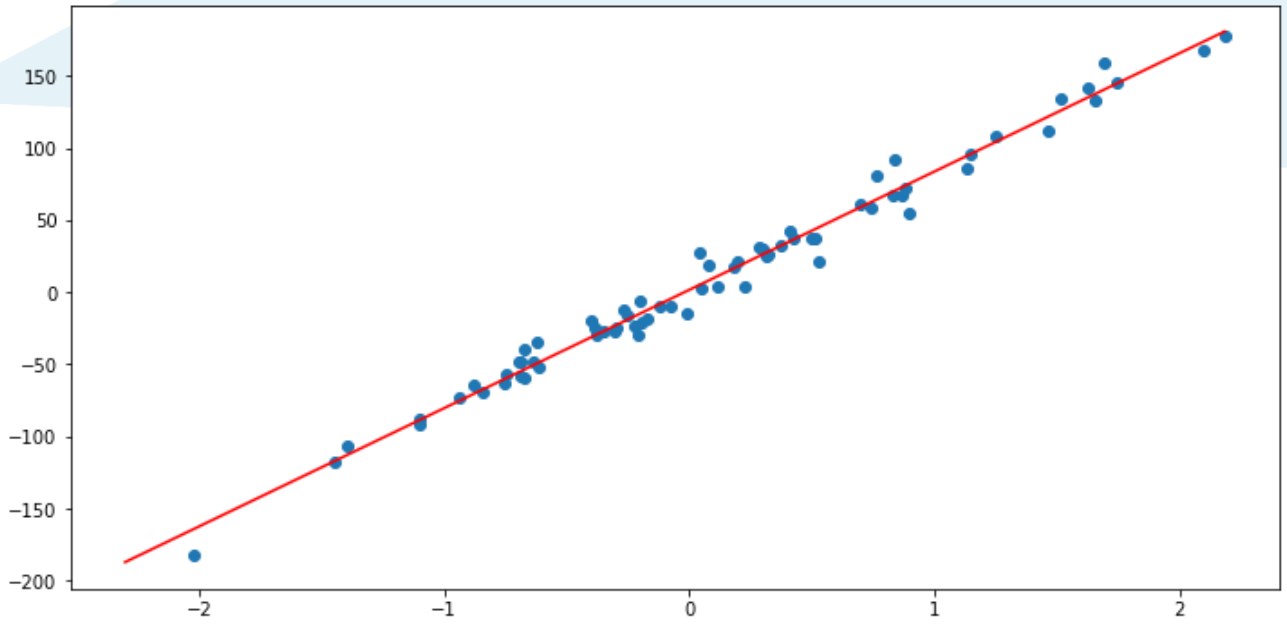
Range tussen 0 en 1

$$\sum_{i=1}^{5} i = 1 + 2 + 3 + 4 + 5$$

$\hat{y} =$ voorspelling van y

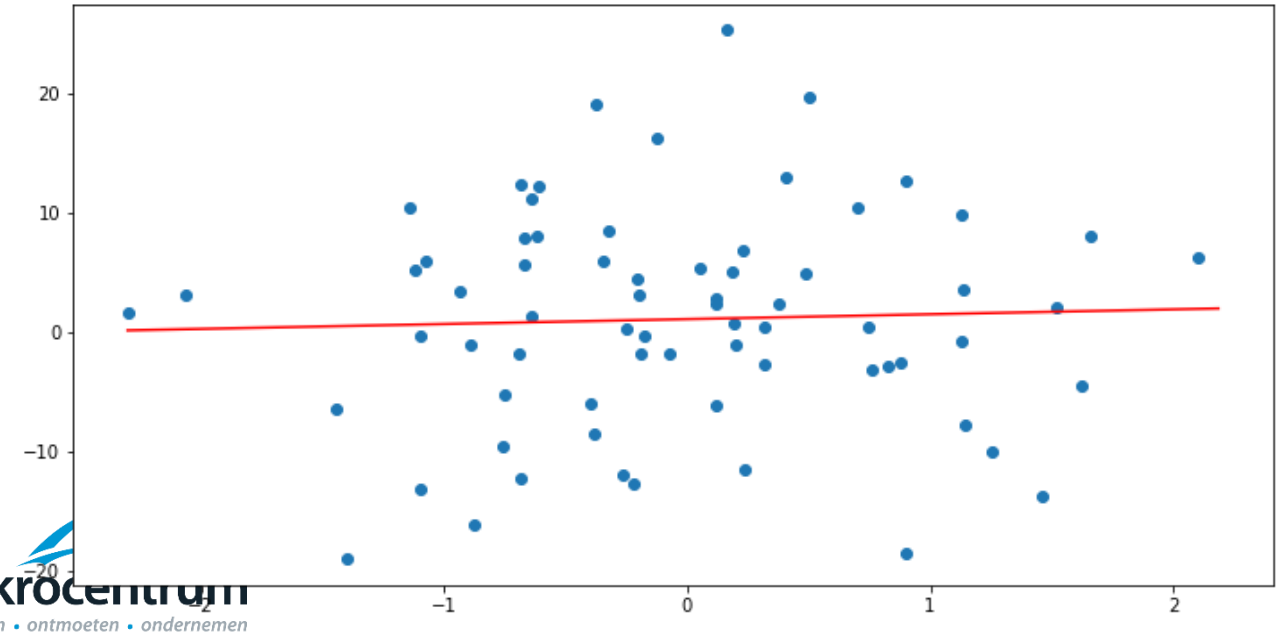
$\bar{y} =$ gemiddelde van y

Goede en slechte regressie



MSE = 94.4

R^2 = 0.98



MSE = 97.5

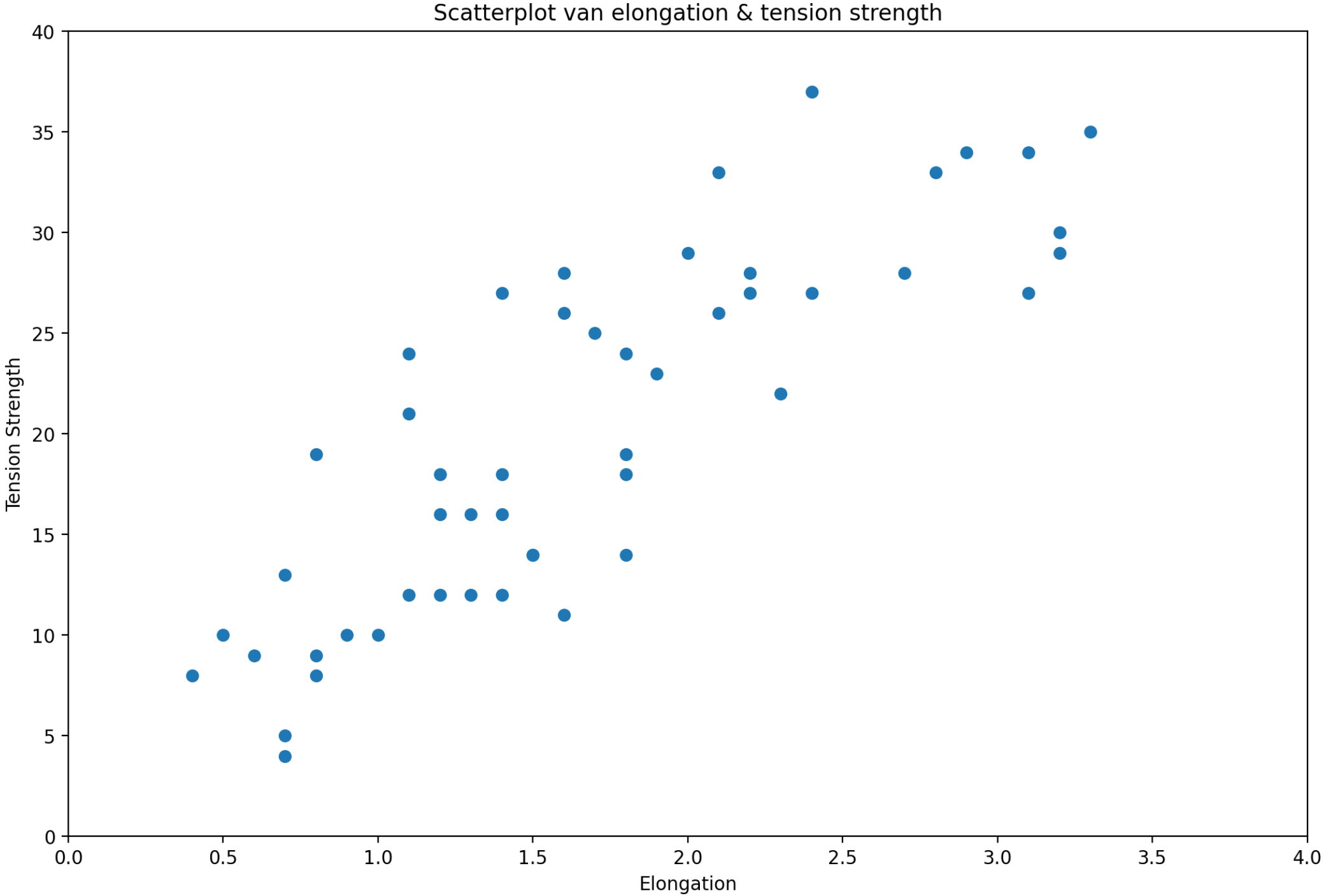
R^2 = -0.009

R2 kan negatief zijn als de voorspellingen slechter zijn als het gemiddelde

Oefening

- Met de 3d printer dataset, voorspel de waarde van “tension_strength” met de variabel "elongation“ (of gebruik de scatterplot op de volgende dia)
- Doe dit door waardes van β te schatten zonder sklearn en een zo laag mogelijke MSE te krijgen
- Maak een scatterplot van de voorspellingen en de daadwerkelijke waarden
- Maak een visualisatie

10 minuten de tijd


Scatterplot van elongation & tension strength
Tension Strength
Elongation
0
5
10
15
20
25
30
35
40
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

Correlatie is geen causatie

$$\hat{y} = \beta_1 x + \beta_0$$

β_1 = Hoeveel x
correleert met y

CORRELATIE IS GEEN CAUSATIE

Correlatie is geen causatie

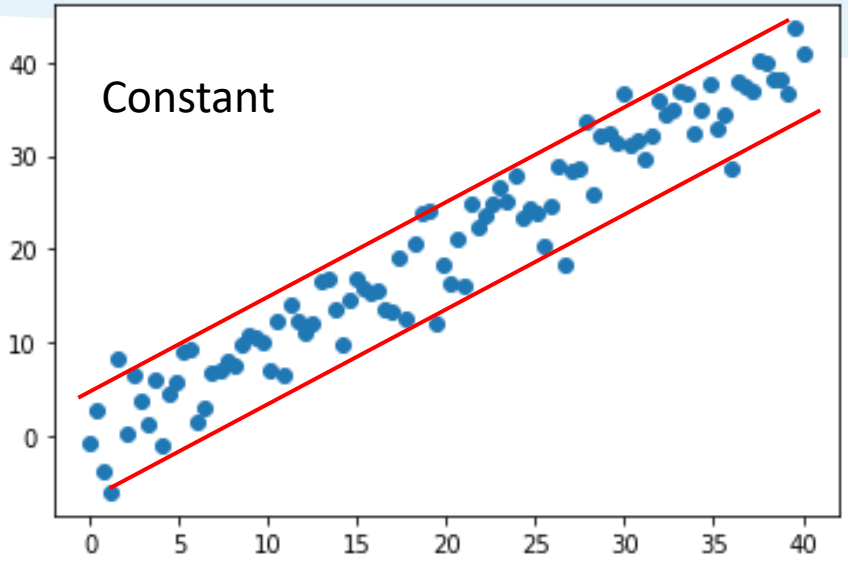
Ik zie veel paraplu's op straat wanneer het regent. Dus regen wordt veroorzaakt door paraplu's.
(tegenovergestelde causaliteit)

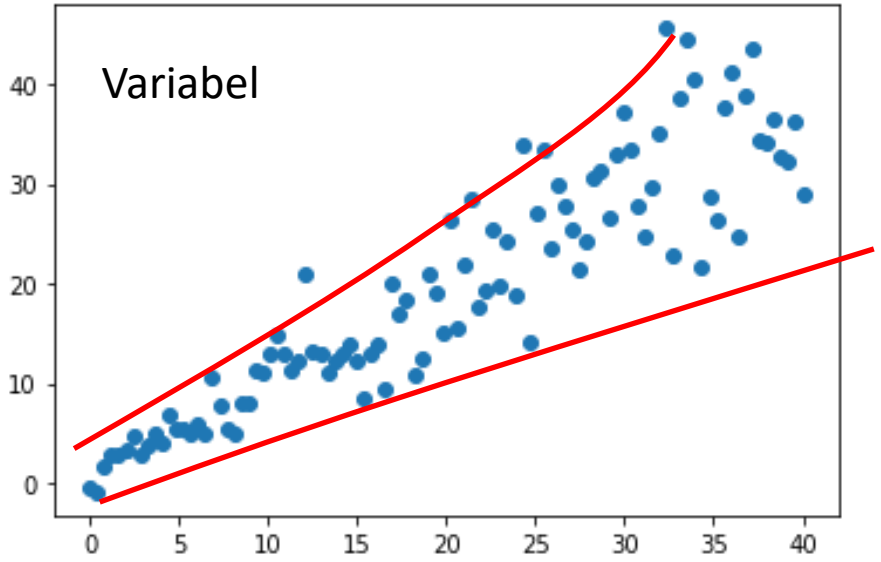
Slapen met je sokken aan zorgt voor wakker worden met hoofdpijn.
(causaliteit door een verbogen variabele)

De hoeveelheid verkochte staal voorspeld het aantal zetels van de Duitse socialistische partij (SPD)
(puur toeval)

Aannamens

- De relatie tussen x en y is lineair (duh)
- De variabel x bevat geen fouten (zoals meetfouten)
- Alle observaties zijn onafhankelijk van elkaar
- Constante variantie





Multi variabele lineaire regressie

$$\hat{y} = \beta_2 x_2 + \beta_1 x_1 + \beta_0$$

$$\min_{\beta} (y - \hat{y})^2$$

`sklearn.linear_model.LinearRegression`

Paar opmerkingen

- Simpel = vaak beter
- x1 en x2 moeten onafhankelijk van elkaar zijn (collineariteit, komen we nog op terug)
- Meer variabelen vereist meer data

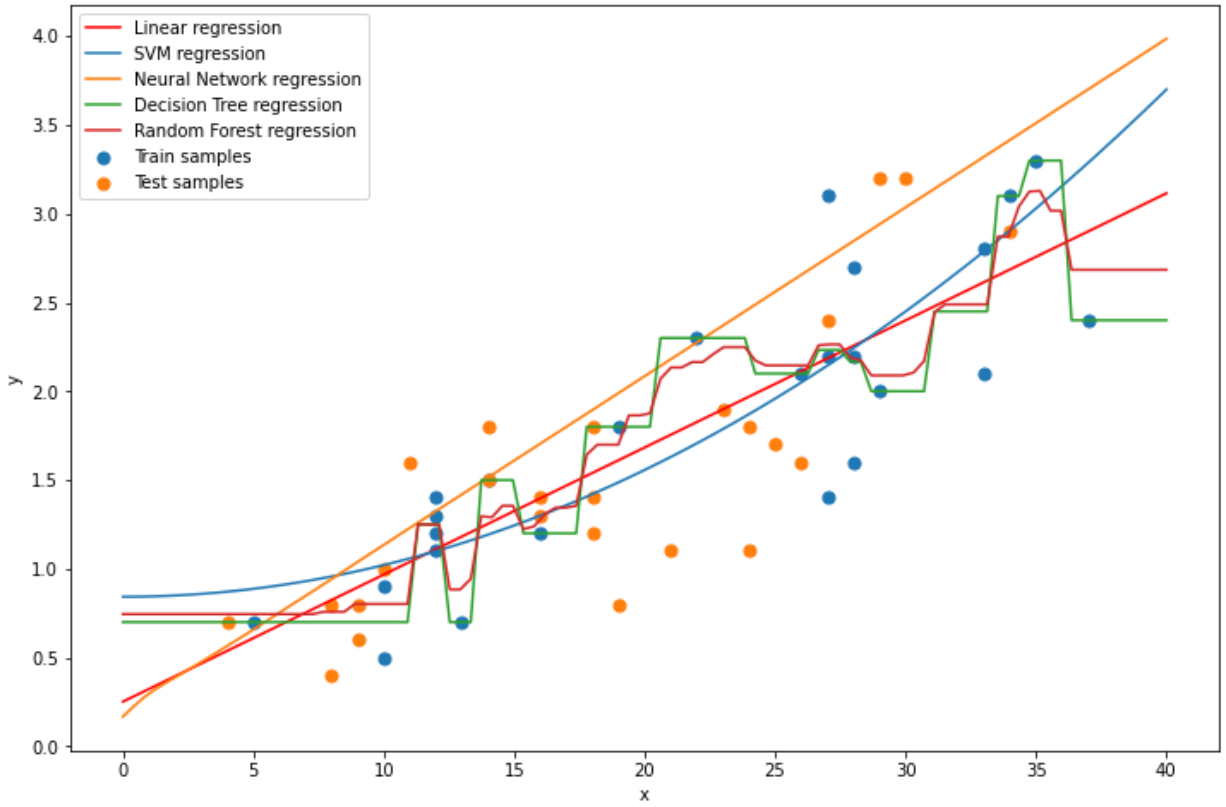
Oefening

- Met de 3d printer dataset, voorspel de waarde van “`tension_strength`”, nu ook met andere variabele
- Gebruik de sklearn library
- Maak een scatterplot van de voorspellingen en de daadwerkelijke waarden. `plt.scatter(y, y_pred)`
- Bereken de MSE en de R2

15 minuten de tijd

Andere regressie methodes

- Support Vector Machine Regress
- Decision Tree/Random Forest regressie
- Neural Network regressie



Meer regressie!

- Ridge regression (`sklearn.linear_model.Ridge`)

$$\min_{\beta} (y - \hat{y})^2 + \lambda \sum_{i=1}^{p} \beta_i^2$$

- Lasso regression (`sklearn.linear_model.Lasso`)

$$\min_{\beta} (y - \hat{y})^2 + \lambda \sum_{i=1}^{p} |\beta_i|$$

Regularization:

Het toevoegen van informatie aan het model om overfitting te voorkomen

Oefening

- Met de 3d printer dataset, voorspel de waarde van “`tension_strength`”, nu met Ridge regression
- Bereken de MSE en de R2
- Visualiseer de *β's* in een barplot van lineare regressie en ridge regressie: `plt.barh(y = x_train.columns, width = linreg.coef_)`

10 minuten

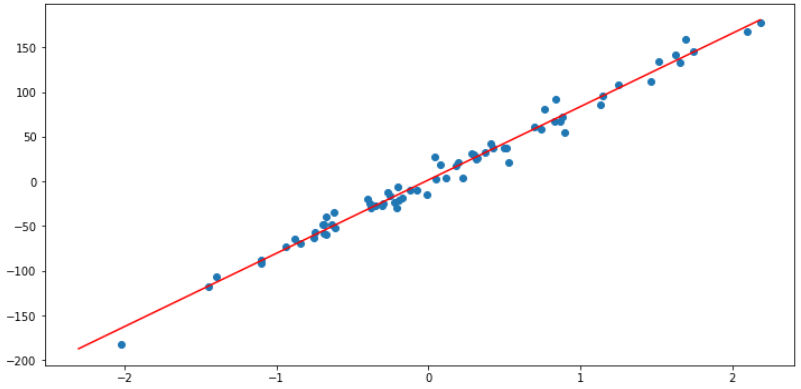
Samenvatting

Lineair

- (Multi Variable) Linear regression
- Ridge Regression
- Lasso Regression

Non-lineair

- SVM Regression (SVR)
- Random Forest regression
- Neural Networks regression



METRICS

Performance measures

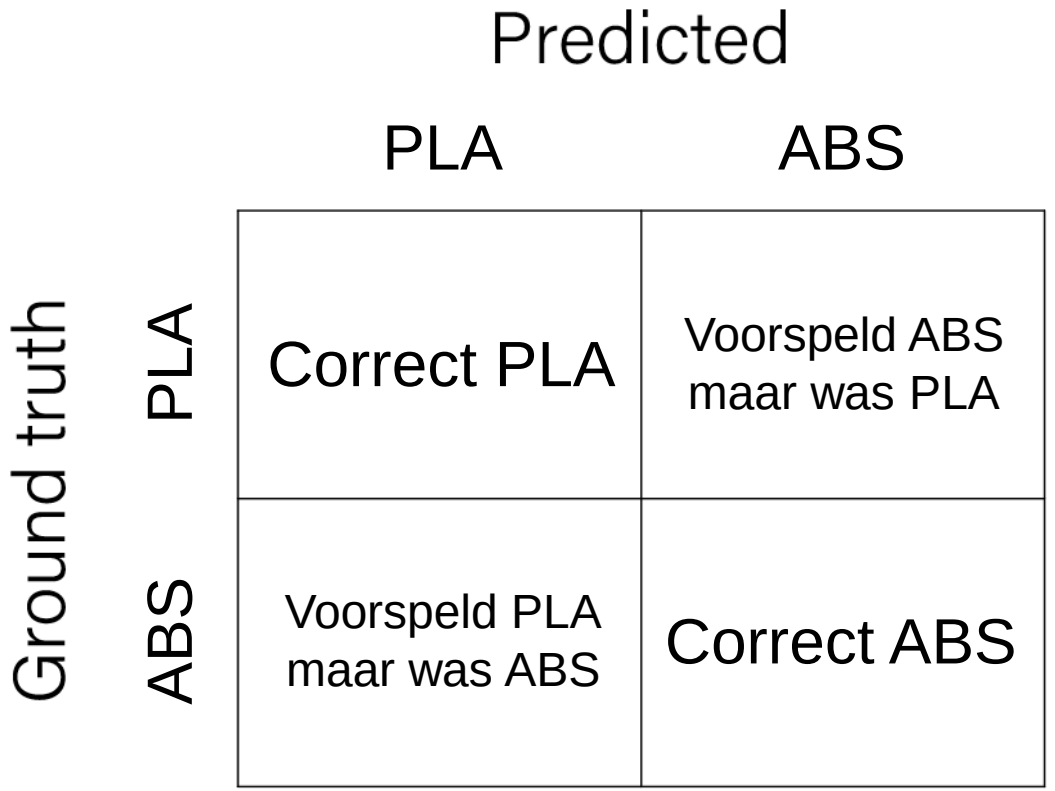
Accuracy

Hoeveel procent is correct geclassificeerd?

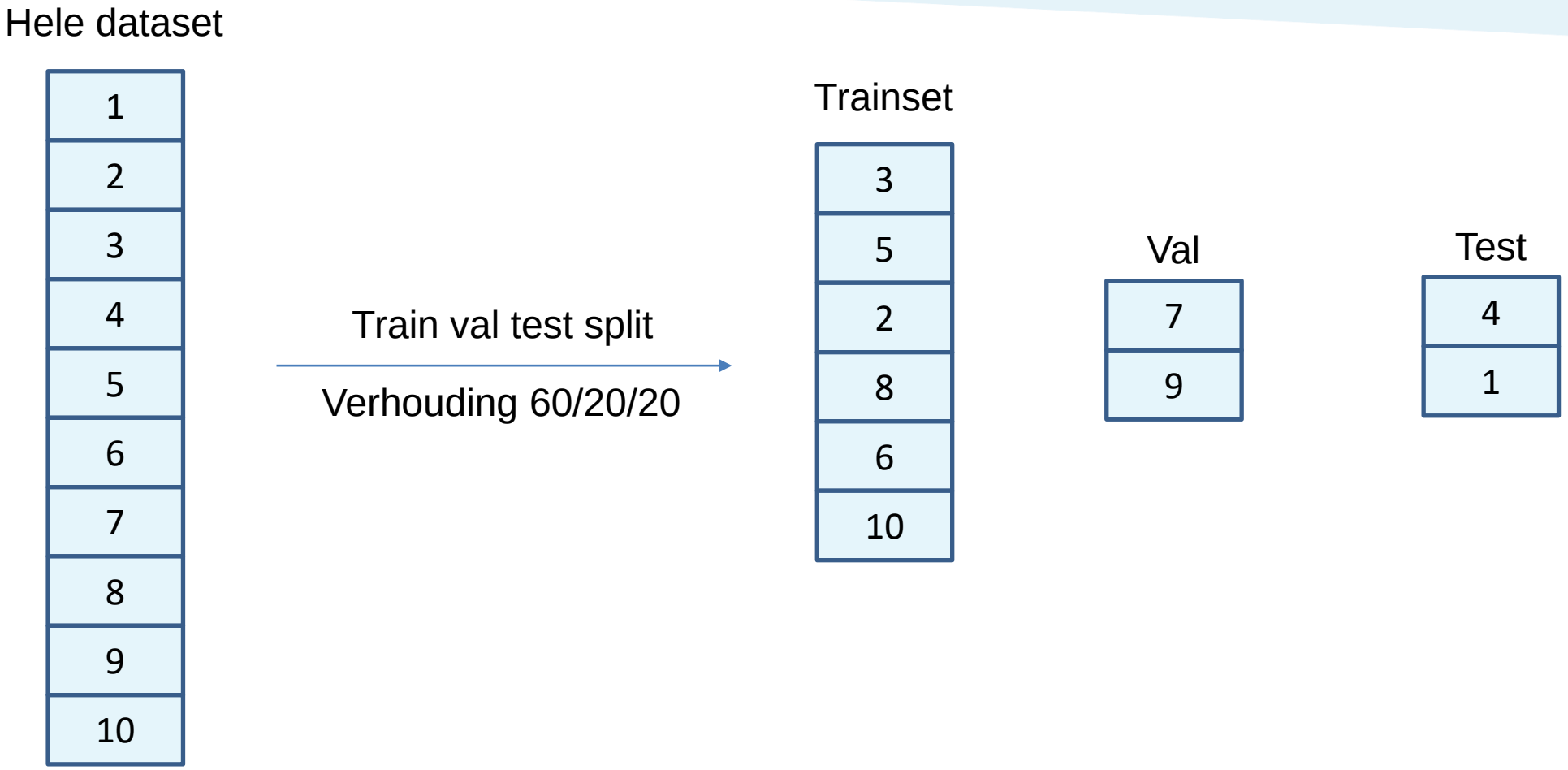
$$\text{Accuracy} = \frac{\text{aantal correct}}{\text{aantal voorspeld}}$$

Ground truth \ Predicted	A	B
A	True A	False B
B	False A	True B

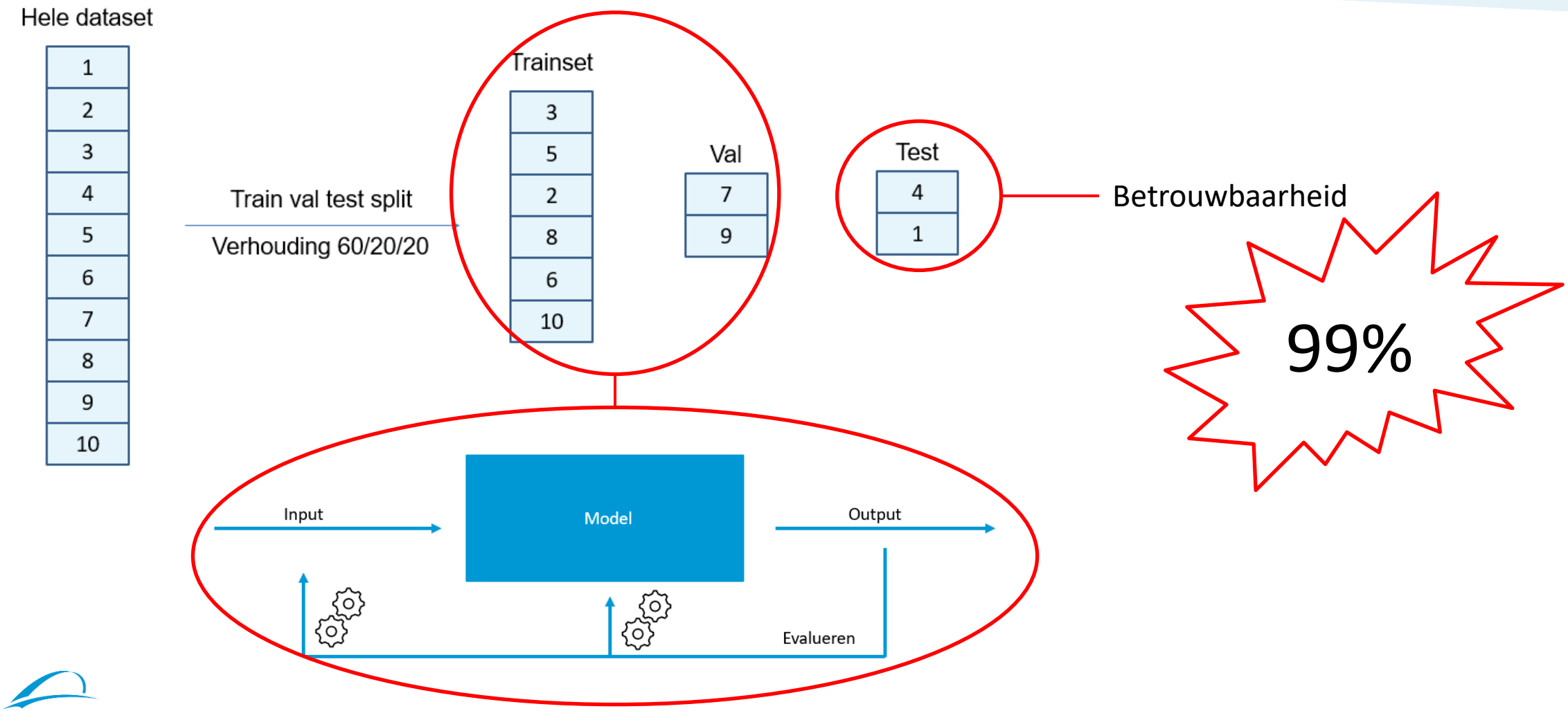
$$\text{Accuracy} = \frac{\text{True A} + \text{True B}}{\text{True A} + \text{False A} + \text{True B} + \text{False B}}$$


Predicted
PLA
ABS
Ground truth
PLA
ABS
Correct PLA
Voorspeld ABS maar was PLA
Voorspeld PLA maar was ABS
Correct ABS

Train-validatie-test split



Voorbeeld


Hele dataset
1
2
3
4
5
6
7
8
9
10
Train val test split
Verhouding 60/20/20
Trainset
3
5
2
8
6
10
Val
7
9
Test
4
1
Betrouwbaarheid
99%
Input
Model
Output
Evalueren

Oefening

Vergelijk alle modellen die je hebt gemaakt. Welk model is het beste om materiaal te voorspellen van de 3D-Printer dataset?

Welk model voorspelt het minste PLA terwijl dat eigenlijk ABS zou moeten zijn?

~15 minuten

Extra lesmateriaal

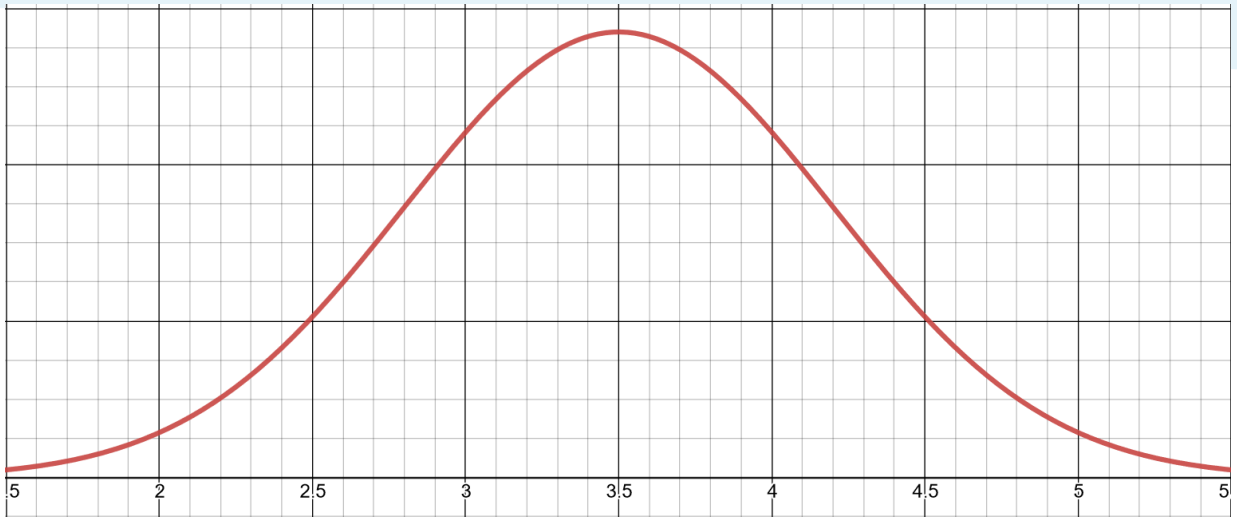
- Oefen met Pandas. Volg deze cursus (voor volgende week): https://www.kaggle.com/learn/pandas
- Bias/variance tradeoff: https://www.youtube.com/watch?v=SjQyLhQIXSM
- SVM kernel trick: https://www.youtube.com/watch?v=3liCbRZPrZA&ab_channel=udiprod
- Decision Trees: https://scikit-learn.org/stable/modules/tree.html#classification
- Random Forest: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm
- Hoe leert een machine (wiskundig): https://www.youtube.com/watch?v=IHZwWFHWa-w&t=5s&ab_channel=3Blue1Brown
- Calling Bullshit (https://www.callingbullshit.org/tools/tools_misleading_axes.html): Hoe te misleiden met grafieken en data

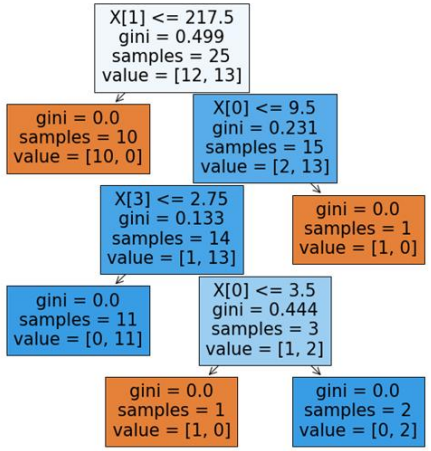
Dagcasus (huiswerk)

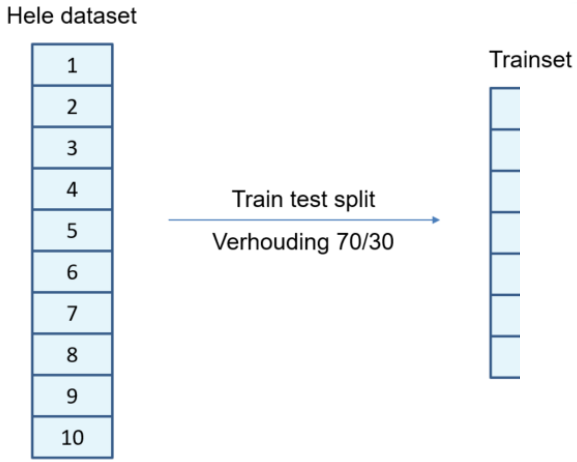
Maak de opdrachten in de ‘dagcasus’ map.

Dag samenvatting

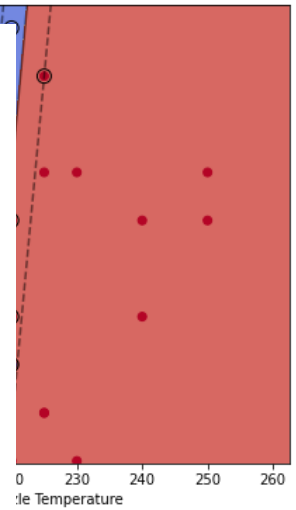
$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

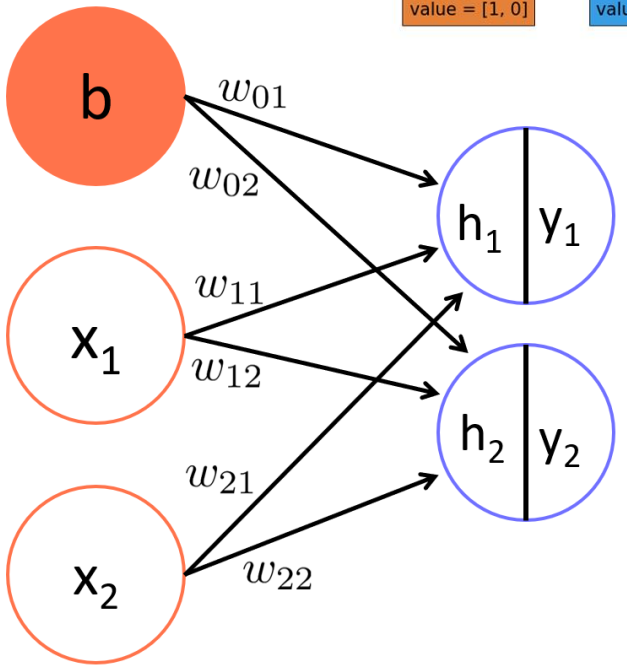






Ground truth	Predicted: PLA	Predicted: ABS
PLA	Correct PLA	Voorspeld ABS maar was PLA
ABS	Voorspeld PLA maar was ABS	Correct ABS





mikrocentrum
opleiden • ontmoeten • ondernemen