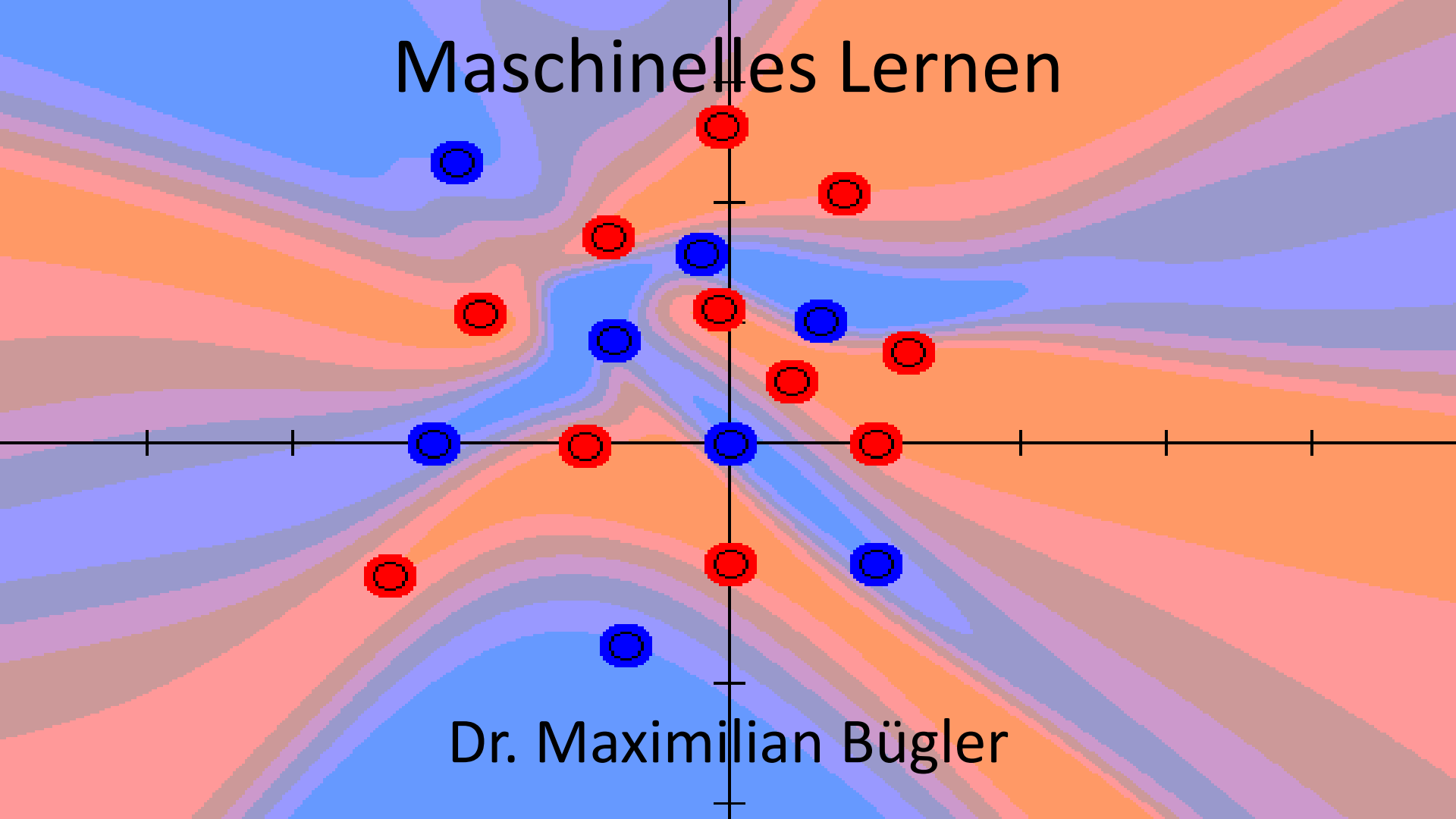
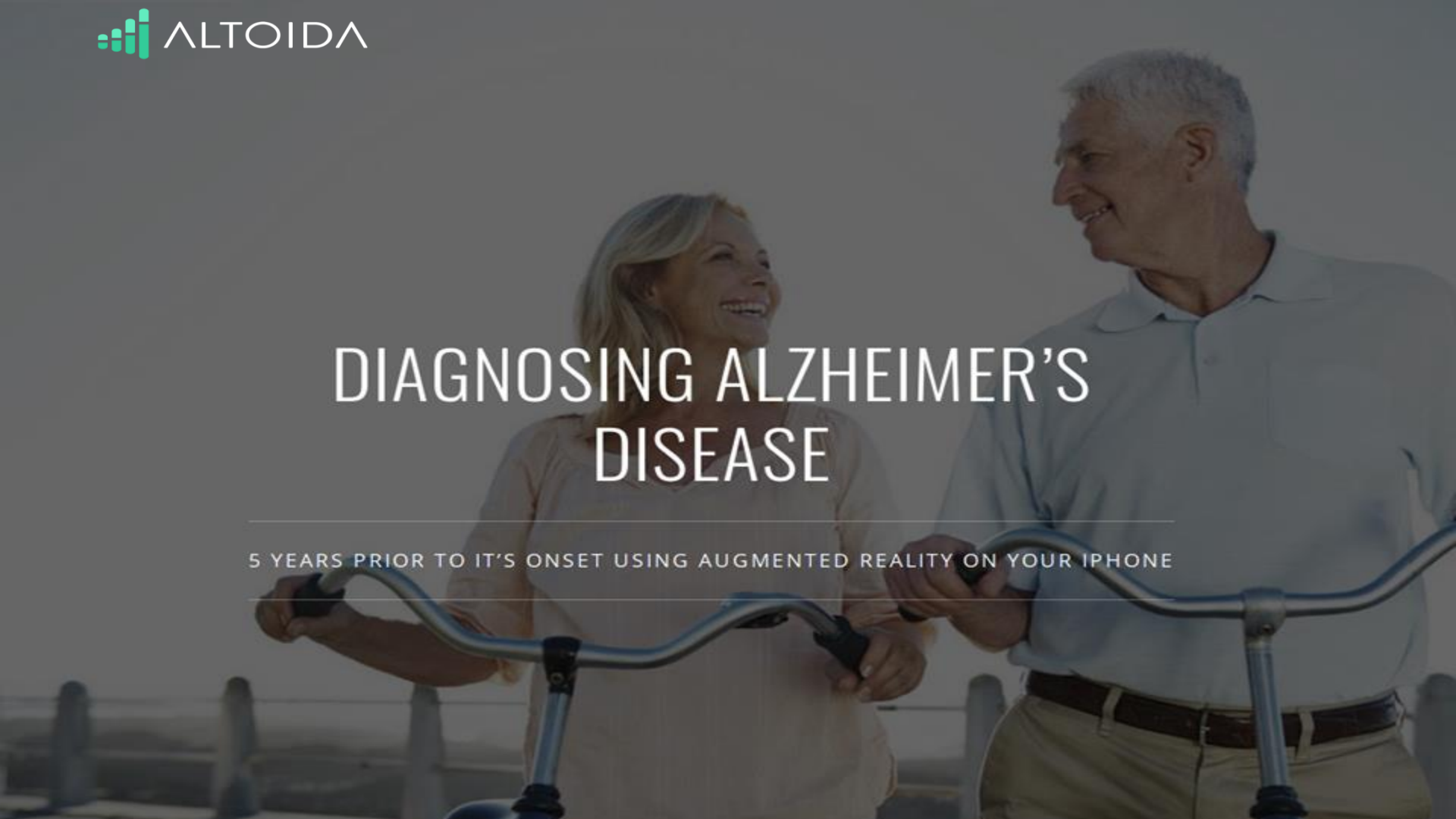
Maschinelles Lernen



Dr. Maximilian Bügler

ALTOIDA



DIAGNOSING ALZHEIMER'S DISEASE

5 YEARS PRIOR TO IT'S ONSET USING AUGMENTED REALITY ON YOUR IPHONE

75% OF ALL ALZHEIMER'S DIAGNOSIS ARE DONE TOO LATE.

Earlier diagnosis could help to dramatically increase the quality years before the actual onset of the disease. Based on 12 years of R&D and clinical trials with more than 2500 patients, we have developed complex everyday activities providing neuromotor biomarkers that can diagnose Alzheimer's disease 5 years prior to onset with 94% accuracy using smartphones and augmented reality. Our technology will power apps that target both consumer and professionals markets.

EARLIER

Discovering Alzheimer's Disease (AD) earlier, will have a heavy impact on treatment and therapy and thus can significantly prolong symptom-free years before the actual onset. Studies today already show that moderate physical exercise can add up to 10 quality years. Nutrition and general lifestyle are closely linked to the disease progression too. Using our patented technology, we are going to diagnose a new cohort of patients that today is undetected and therefore untreated.

CHEAPER

The current state-of-the-art clinical diagnosis of AD requires a specialty clinic and includes a medical examination, neuropsychological testing, neuroimaging (MRI), cerebrospinal fluid (CSF) analysis and blood examination. This process is very time-intense and costly. With an easier to use, more scalable and less expensive diagnosis method, that is even more accurate than todays gold-standard, we are going to change that paradigm.

PERSONALIZED

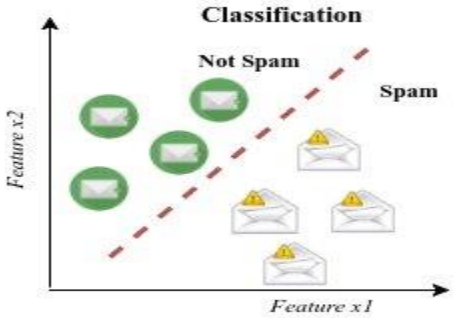
It is without debate, that the future of medicine is personalized. In the specific case of AD, amyloid proteins, APOE4, and tau all contribute to the individual progression of the disease. And so do other factors of lifestyle, nutrition and phyiscal exercise. In order to treat or prevent AD, personalized, on actual patient data based therapies and drugs are much more promising than one-fits-all approaches.

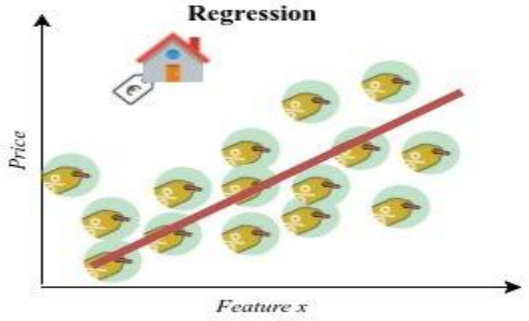
Source: http://www.altoida.com/

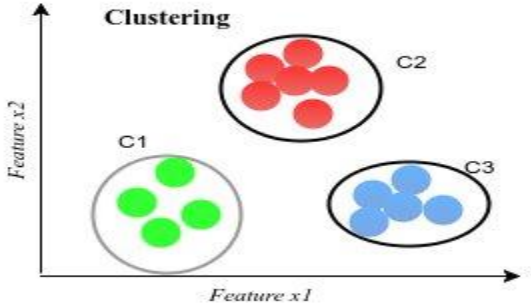
Was ist maschinelles Lernen?

- Erfahrung
  - Durch Wahrnehmung erworbene Kenntnisse
- Wissen
  - Fakten, Theorien, Regeln
- Maschinelles lernen
  - Erfahrung → Wissen

Arten von maschinellem Lernen



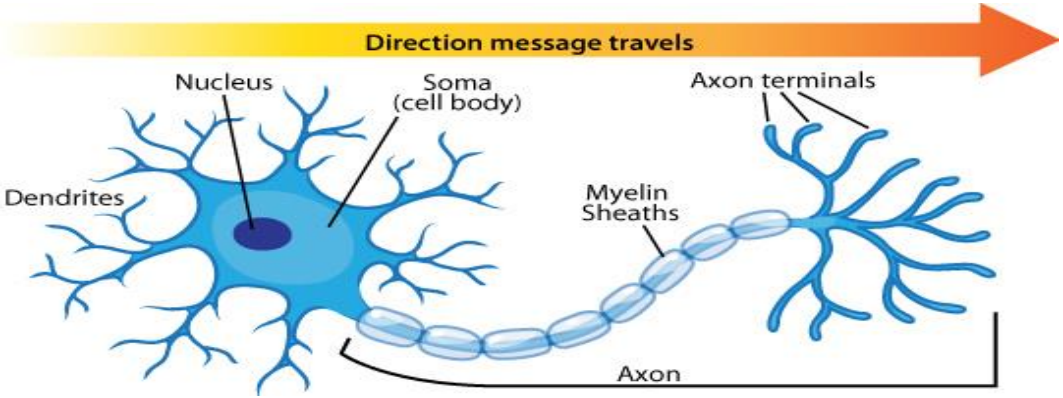


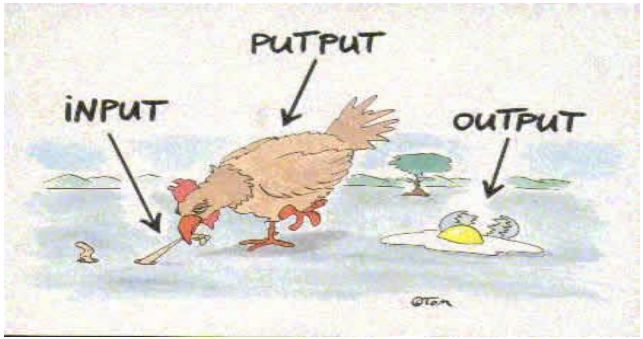


- Klassifizierung
- Regression
- Kategorisierung
- Aktives Lernen

Biologische neuronale Netze

- Was ist ein Neuron?
  - Eingang (Dendrit)
  - Neuron (Soma)
  - Ausgang (Axon)
  - Neuron „feuert“ unter bestimmten Umständen





Oberes Bild: https://askabiologist.asu.edu/neuron-anatomy Unteres Bild: http://schluroff.de/mmb/putput.htm

Biologische neuronale Netze

- Was ist ein neuronales Netz?
  - Neuron sind in einer Netzwerkstruktur verbunden
  - Eingänge von Neuronen sind mit Ausgängen anderer Neuronen oder Sensoren verbunden

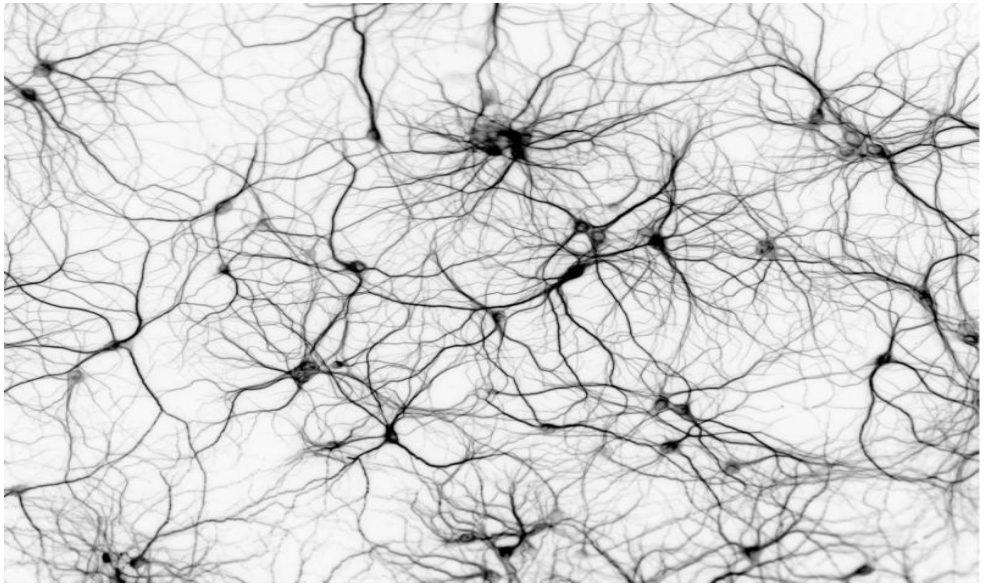


Bild von: http://www.wordlypost.in/artificial-neural-network-learns-to-communicate-with-humans/

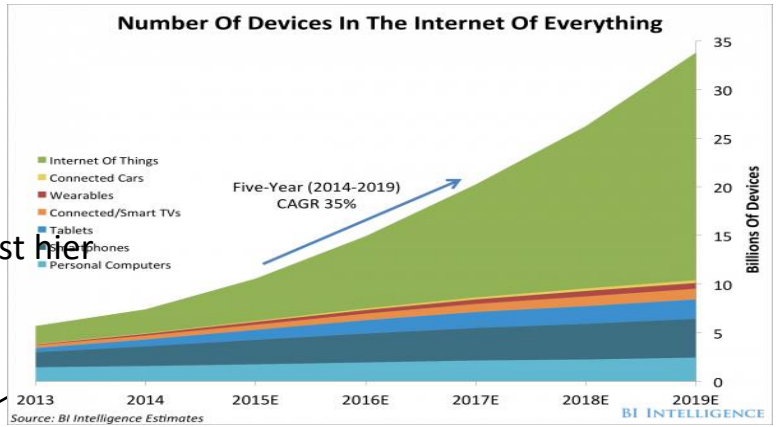
Biologische neuronale Netze

- Wie groß sind neuronale Netze?
  - 302 Neuronen: Fadenwurm
  - 10^5 Neuronen: Fliege
  - $1.5 \cdot 10^7$ Neuronen: Ratte
  - $1.6 \cdot 10^8$ Neuronen: Hund
  - $3.0 \cdot 10^8$ Neuronen: Katze
  - 10^{11} Neuronen: Mensch
  - $2 \cdot 10^{11}$ Neuronen: Elefant
    - Data from http://hagan.okstate.edu/NNDesign.pdf

http://www.openworm.org/

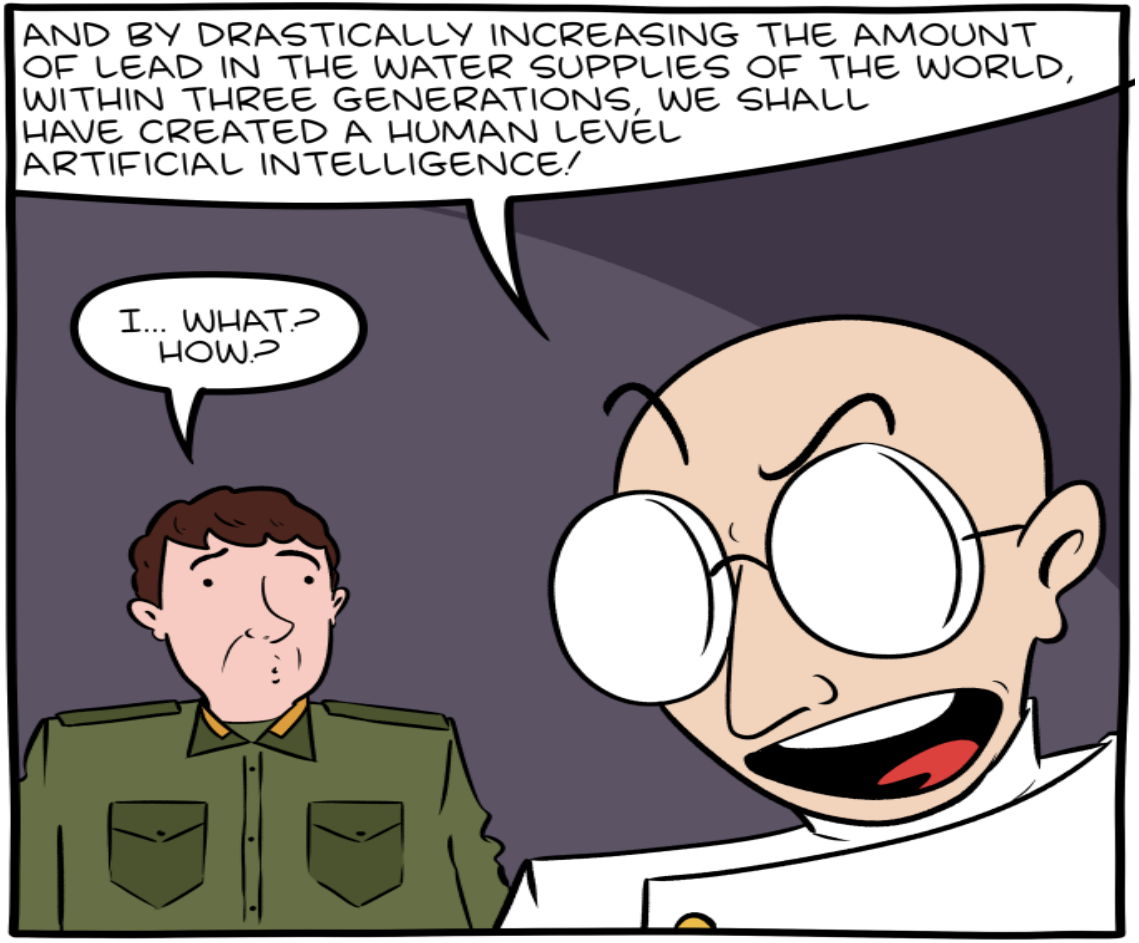
Unsere Computer sind hier (???)

Du bist hier



Was wäre wenn wir auf jedem Gerät im Internet 10 Neuronen simulieren würden?

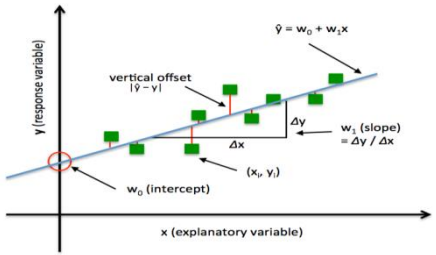
Bild von: http://www.ironpaper.com/webintel/articles/internet-things-market-statistics-2015/

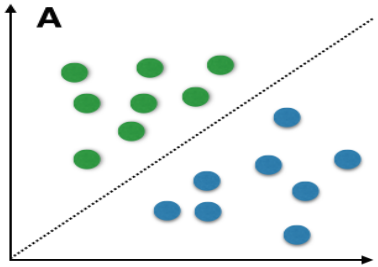


Any AI can be made "human level" by lowering all human intelligence.

Wie funktioniert ein künstliches neuronales Netz?

- Ein neuronales Netz kann sowohl für Regression, wie auch für Klassifikation verwendet werden.
- Für eine bestimmte Eingabe kann eine bestimmte Ausgabe „trainiert" werden.
- Klassifizierung erfolgt durch Überprüfung ob ein Neuron „feuert" oder nicht.





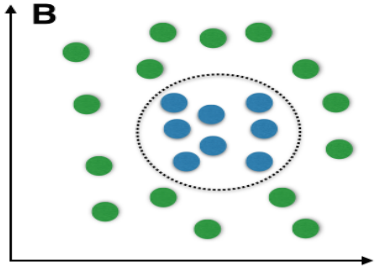


Bild: http://sebastianraschka.com/Articles/2014_naive_bayes_1.html and http://sebastianraschka.com/faq/docs/closed-form-vs-gd.html

Das Perzeptron

- Simples lineares Modell eines Neurons
- Es gibt **R** Eingangsdendrites
- Jeder Eingang hat ein Gewicht **w**
- Das Neuron hat einen Bias Wert **b**
- Es gibt eine Aktivierungsfunktion **f**
- Die Ausgabe des Neurons ist **a**

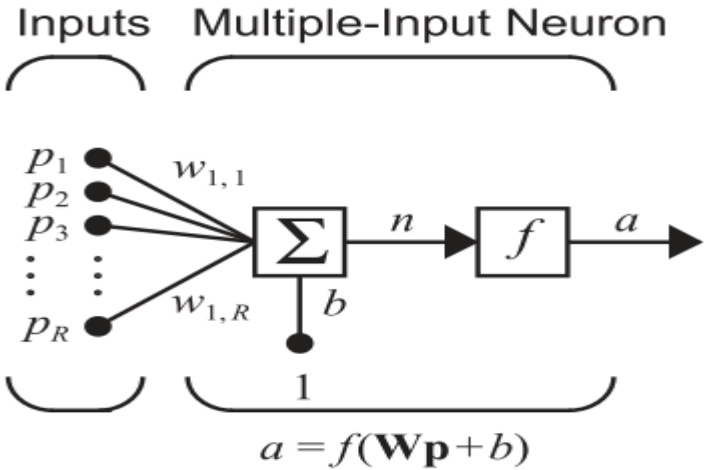
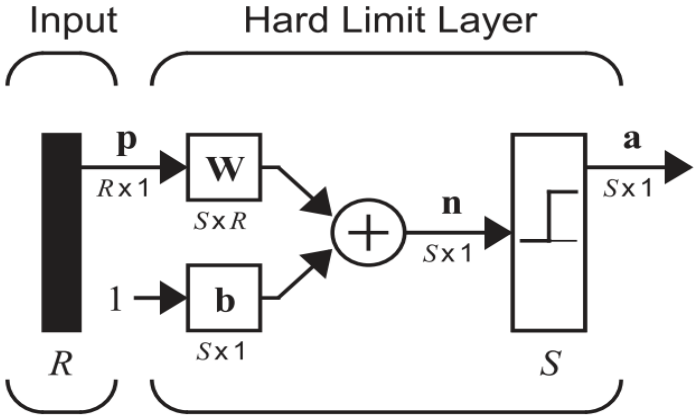


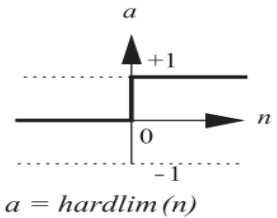
Image from: Hagan et al. - Neural Network Design - http://hagan.okstate.edu/NNDesign.pdf

Das Perzeptron – Matrix Form

- Es gibt nun **R** Eingänge und **S** Ausgänge
- Eingaben werden repräsentiert durch einen Vektor **p** der Größe **R**x1.
- Die Gewichtungen der Eingänge werden in der Gewichtsmatrix **w** der Größe **SxR** und dem Bias Vektor **b** der Größe **S**x1 gespeichert.
- Als Aktivierungsfunktion **f** wird die **hardlim** Funktion verwendet, welche 1 ausgibt sobald der Parameter 0 erreicht.
- Ob ein Neuron feuert wird errechnet indem man die Gewichtsmatrix mit dem Eingabevektor multipliziert, den Biasvektor addiert und das Ergebnis in die Aktivierungsfunktion einsetzt.



$$\mathbf{a} = \mathbf{hardlim}(\mathbf{Wp}+\mathbf{b})$$

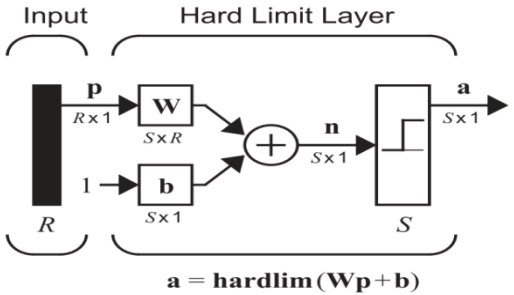


$$a = hardlim(n)$$

Image from: Hagan et al. - Neural Network Design - http://hagan.okstate.edu/NNDesign.pdf

Das Perzeptron – Ein Beispiel

- 2 Eingänge, 1 Ausgang
- Gewichtsmatrix W = [-1 1]
- Biasvektor b = -1



- Was passiert wenn wir dem Netzwerk den Vektor p = [1 0] präsentieren?
- a = hardlim (W p + b) = hardlim (-1 · 1 + 1 · 0 – 1) = hardlim (-2) = 0
- Und bei p = [0 1]?
- a = hardlim (W p + b) = hardlim (-1 · 0 + 1 · 1 – 1) = hardlim (0) = 1

Das Perzeptron – Ein Beispiel

- 2 Eingänge, 1 Ausgang
- Gewichtsmatrix **W** = [-1 1]
- Biasvektor **b** = -1

Geometrische Interpretation:

- **W** → Richtung der Entscheidungsgrenze
- **b** → Verschiebung

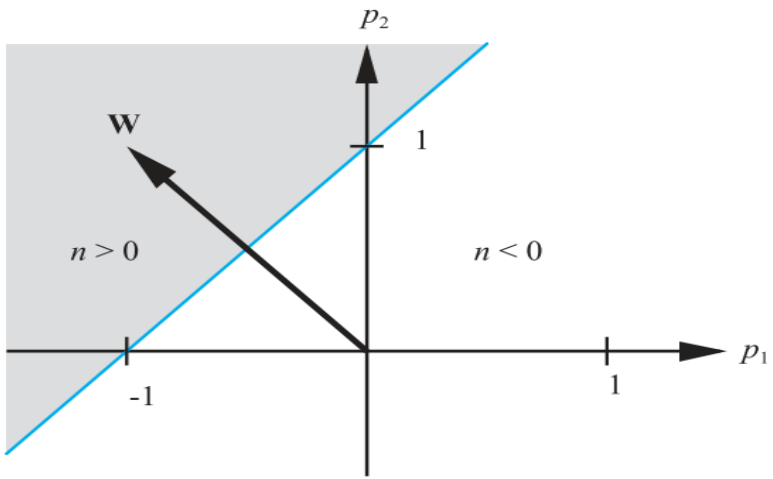


Image from: Hagan et al. - Neural Network Design - http://hagan.okstate.edu/NNDesign.pdf

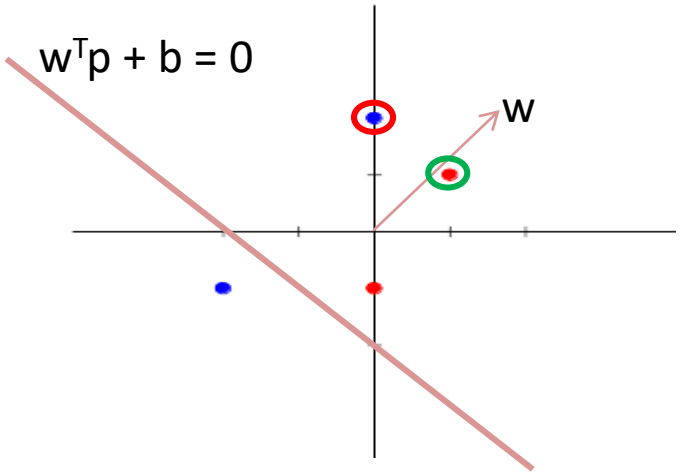
Die Perzeptron-Lernregel

- Wir starten mit W = [1 1] und b = [2]

- Entscheidungsgrenze: $W^T p + b = 0 \Leftrightarrow 1x + 1y + 2 = 0 \Leftrightarrow y = - x - 2$

- $p_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $t_1 = [1]$
- a = hardlim (W p + b) = hardlim (1 + 1 + 2) = 1
- e = t –a = 1 – 1 = 0

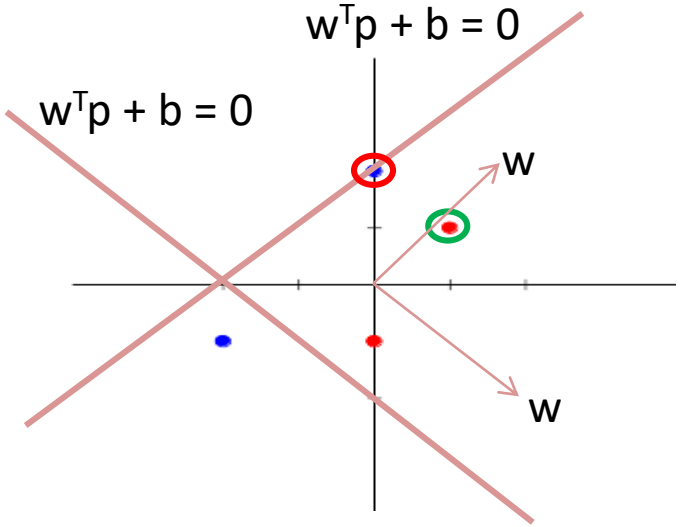
- $p_2 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$ $t_2 = [0]$
- a = hardlim (W p + b) = hardlim (0 + 2 + 2) = 1
- e = t –a = 0 – 1 = -1



Die Perzeptron-Lernregel

- $p_2=\begin{bmatrix}0\\2\end{bmatrix}$ $t_2 = [\ 0\ ]$
- a = hardlim (W p + b) = hardlim (0 + 2 + 2) = 1
- e = t –a = 0 – 1 = -1
  - ➔ Ergebnis zu groß
  - ➔ Wo liegt die Ursache des Fehlers?
  - ➔ Wie korrigieren wir den Fehler?
  - ➔ $W_{new}=W_{old} - p^T$
  - ➔ $W_{new}=[\ 1\ 1\ ] - [\ 0\ 2\ ] = [\ 1\ -1\ ]$

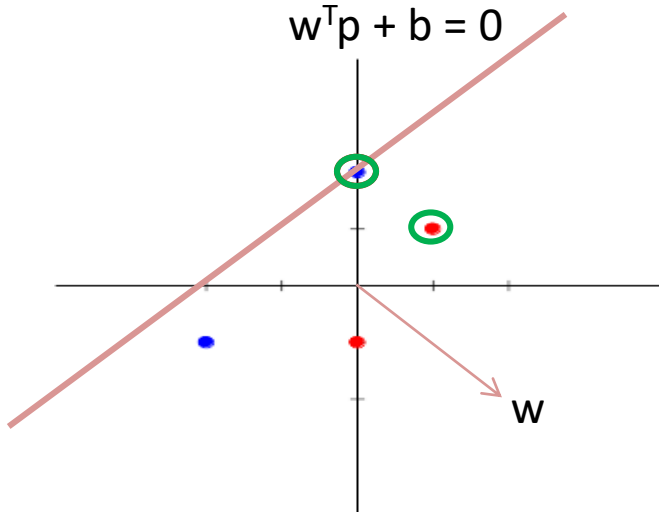
- Entscheidungsgrenze:
- $W^Tp + b = 0$
  $\Leftrightarrow$ 1x - 1y + 2 = 0
  $\Leftrightarrow$ y = x + 2



Die Perzeptron-Lernregel

- $p_2=\begin{bmatrix}0\\2\end{bmatrix}$ $t_2 = [\ 0\ ]$
- a = hardlim (W p + b) = hardlim (0 - 2 + 2) = 1
- e = t –a = 0 – 1 = -1
  - ➔ Ergebnis immer noch zu groß
  - ➔ Zu großes Ergebnis durch Biasvektor
  - ➔ Biasvektor muss angepasst werden
  - ➔ $b_{new} = b_{old} + e = 2 – 1 = 1$

- Entscheidungsgrenze:
  $W^Tp + b = 0$
  $\Leftrightarrow$ 1x - 1y + 1 = 0
  $\Leftrightarrow$ y = x + 1



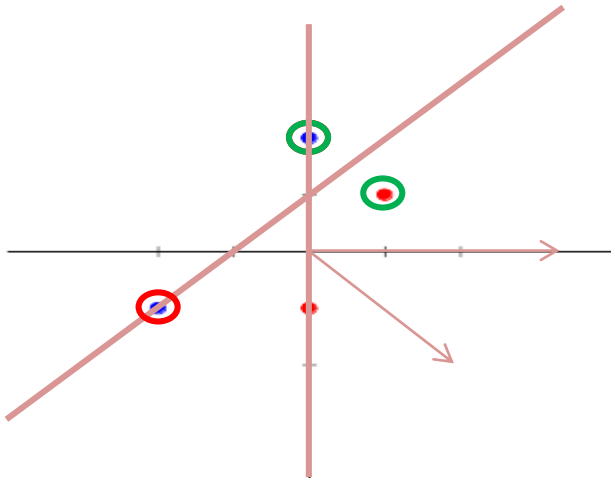
Die Perzeptron-Lernregel

- $p_3=\begin{bmatrix}-2\\-1\end{bmatrix}$ $t_3 = [\ 0\ ]$
- a = hardlim (W p + b) = hardlim (-2 + 1 + 1) = 1
- e = t –a = 0 – 1 = -1

– $W_{new}=W_{old} - p^T$

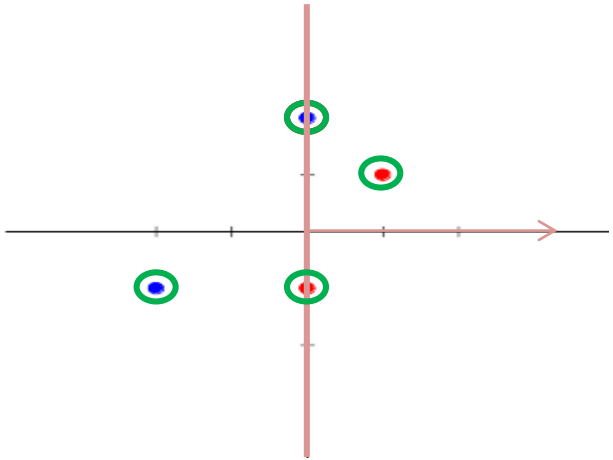
– $W_{new}=[\ 1\ -1\ ] - [\ -2\ -1\ ] = [\ 3\ 0\ ]$ ➔
Entscheidungsgrenze: x = -1/3

– $b_{new} = b_{old} + e = 1 - 1 = 0$ ➔
Entscheidungsgrenze: x = 0



Die Perzeptron-Lernregel

- $p_4=\begin{bmatrix} 0 \\ -1 \end{bmatrix}$ $t_4 = [\ 1\ ]$
- a = hardlim (W p + b) = hardlim (0 + 1 + 0) = 1
- e = t –a = 1 – 1 = 0



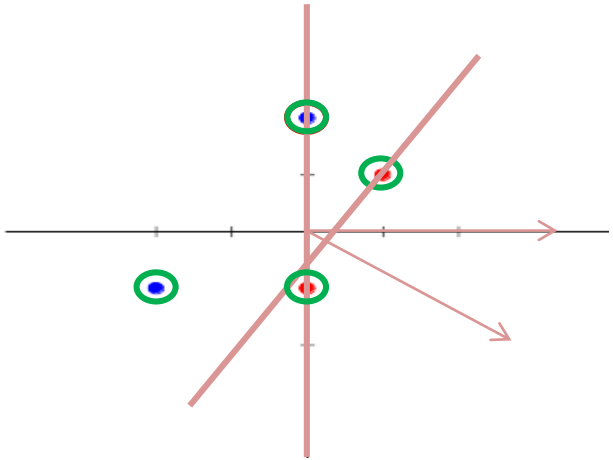
Die Perzeptron-Lernregel

- $p_2=\begin{bmatrix}0\\2\end{bmatrix}$ $t_2 = [\ 0\ ]$
- a = hardlim (W p + b) = hardlim (0 + 0 + 0) = 1
- e = t –a = 0 – 1 = -1

– $W_{new}=W_{old} - p^T$

– $W_{new}=[\ 3\ 0\ ] - [\ 0\ 2\ ] = [\ 3\ -2\ ]$
– $b_{new} = b_{old} + e = 0 - 1 = -1$

– Entscheindungsgrenze:
$W^Tp + b = 0$
$\Leftrightarrow$ 3x - 2y - 1 = 0
$\Leftrightarrow$ y = 3/2x - 1/2

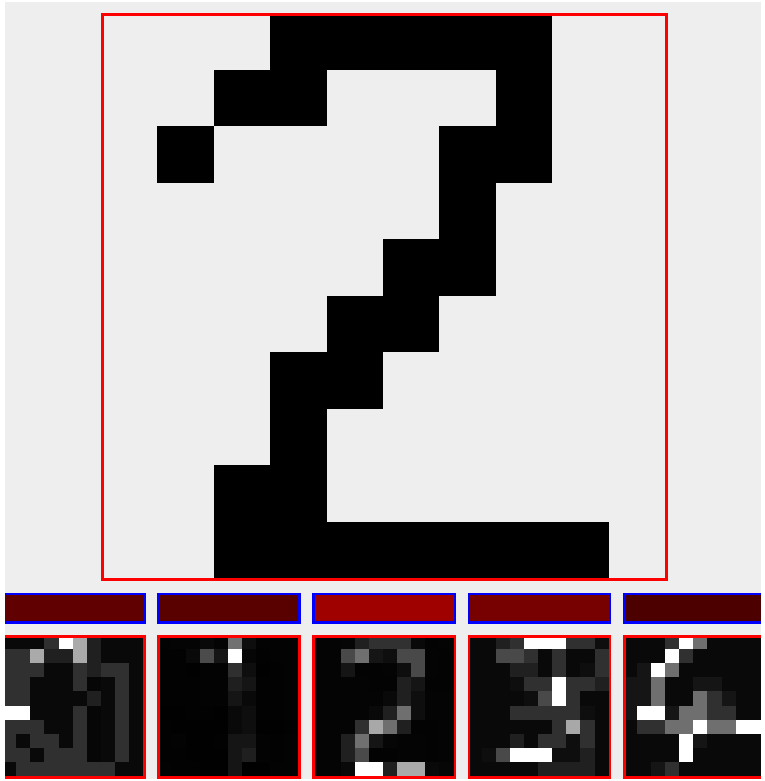


Die Vereinheitlichte Perzeptron-Lernregel

- $W_{new} = W_{old} + e\, p^T = W_{old} + (t - a)\, p^T$

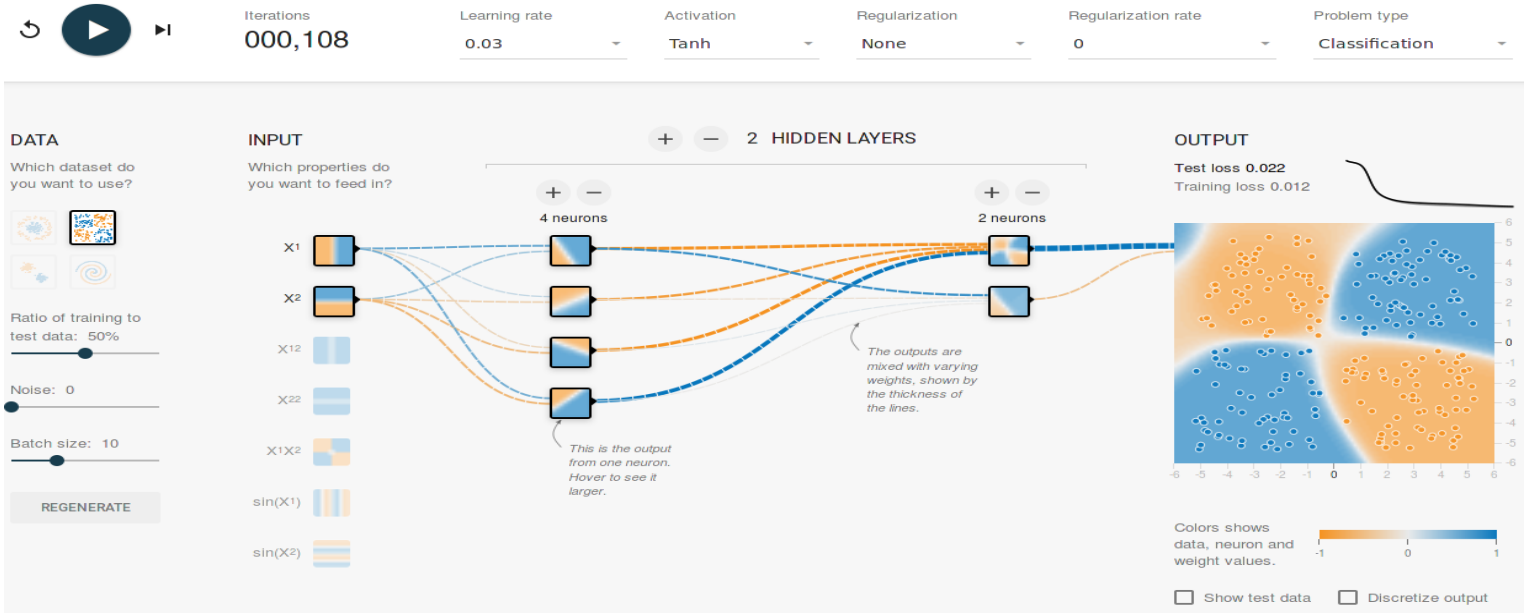
- $b_{new} = b_{old} + e = b_{old} + t - a$

Anwendungsbeispiel



Tensorflow Playground

- http://playground.tensorflow.org/



Describes without errors	Describes with minor errors	Somewhat related to the image	Unrelated to the image
 A person riding a motorcycle on a dirt road.	 Two dogs play in the grass.	 A skateboarder does a trick on a ramp.	 A dog is jumping to catch a frisbee.
 A group of young people playing a game of frisbee.	 Two hockey players are fighting over the puck.	 A little girl in a pink hat is blowing bubbles.	 A refrigerator filled with lots of food and drinks.
 A herd of elephants walking across a dry grass field.	 A close up of a cat laying on a couch.	 A red motorcycle parked on the side of the road.	 A yellow school bus parked in a parking lot.

Source: https://research.googleblog.com/2014/11/a-picture-is-worth-thousand-coherent.html

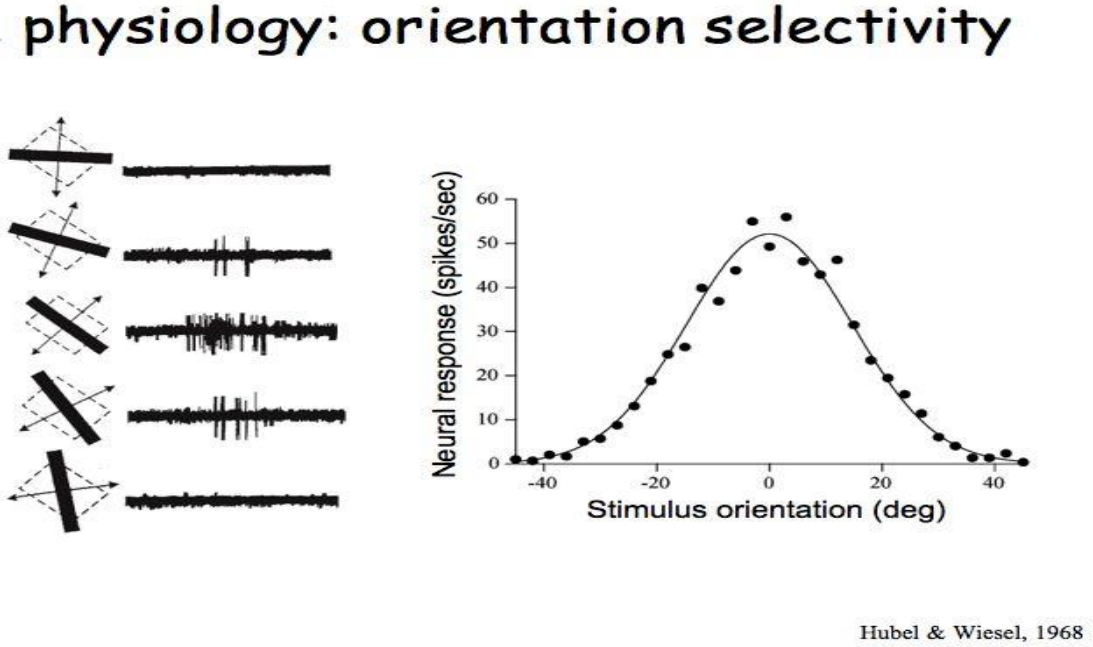
Was ist deep learning?

Deep learning ist eine Klasse künstlicher neuronaler Netze, die zahlreiche Zwischenlagen zwischen Eingabeschicht und Ausgabeschicht haben und dadurch eine umfangreiche innere Struktur aufweisen.

Source: https://en.wikipedia.org/wiki/Deep_learning#Definitions

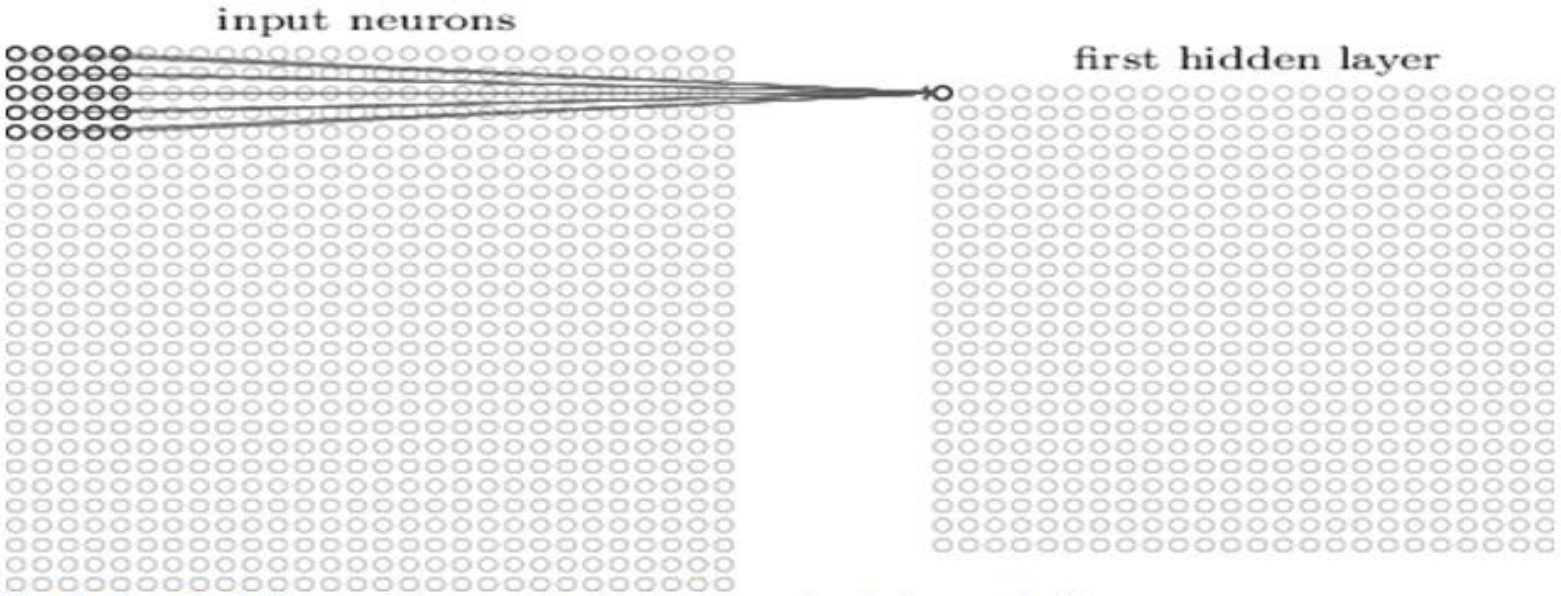
Convolutional Neural Networks

- Inspiriert vom visuellen Kortex



Source: http://www.cns.nyu.edu/~david/courses/perception/lecturenotes/V1/lgn-V1.html

Convolutional Neural Networks



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

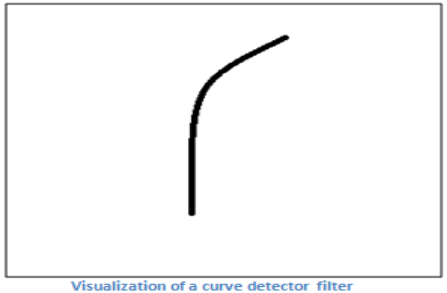
Source: http://neuralnetworksanddeeplearning.com/

Convolutional Neural Networks

- Verschiedene Filter werden für verschiedene Arten von Merkmaler verwendet.
- Filter werden ähnlich wie Gewichte in neuronalen Netzen trainiert.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



Visualization of a curve detector filter

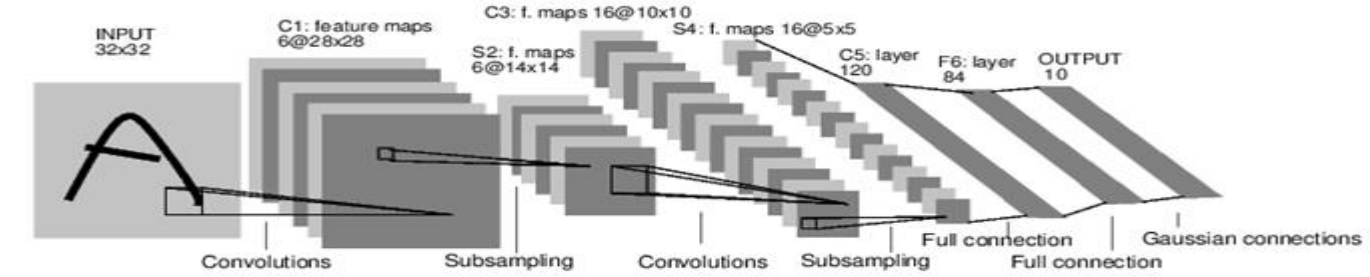
Original image

Visualization of the filter on the image

Source: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

Convolutional Neural Networks

- Mehrere Schichten neuronaler Netze und Convolution Networks sind meist in Reihe geschaltet und werden meist zum Schluss mit einem neuronalen Netzwerk zusammengeführt.



A Full Convolutional Neural Network (LeNet)

Source: https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

Select all images with an **Orange**.

Report a problem

Verify

Aktives Lernen

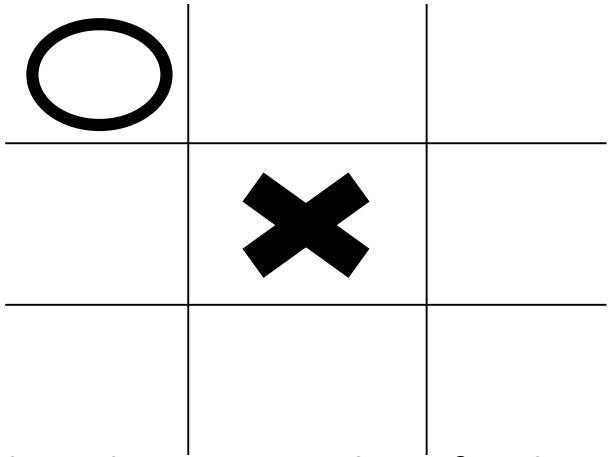
- https://www.youtube.com/watch?v=cNZPRsrwumQ

Aktives Lernen / Bestärkendes Lernen

- Eine Vielzahl von Entscheidungen oder Aktionen müssen getroffen werden um den Erfolg des Handelns zu bestimmen.
- Keine simplen Eingabe/Ausgabe Paare zum Training verfügbar.

Bestärkendes Lernen

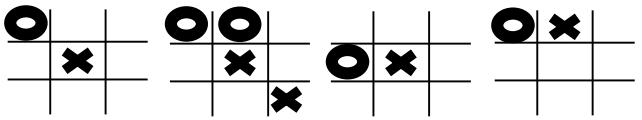
- Spielen wir eine Runde Tic Tac Toe



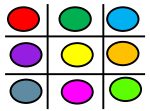
- Wie gut war der Zug?
  - Wir können die Qualität des Zuges nicht sofort beurteilen
  - (es sei denn wir analysieren jeden möglichen Ausgang des Spiels)

Bestärkendes Lernen

- Tic Tac Toe hat eine endliche Menge an möglichen Zuständen.
  - Stellen wir uns pro Zustand eine Schachtel vor.

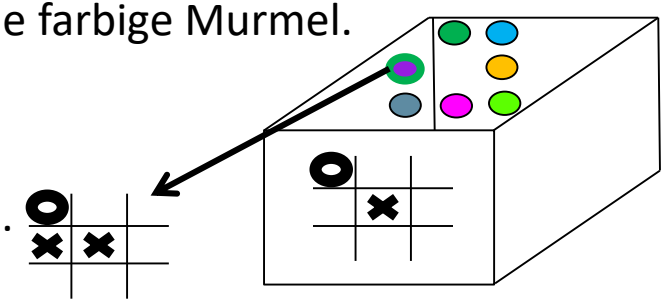


- Für jeden Zustand gibt es eine endliche Anzahl möglicher Aktionen
  - Stellen wir uns pro möglicher Aktion eine farbige Murmel vor.



- In jeder Schachtel liegt pro möglicher Aktion eine farbige Murmel.

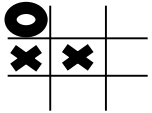
- Erreichen wir einen Zustand, ziehen wir zufällig eine Murmel aus der entsprechenden Schachtel.



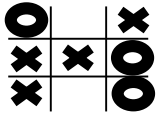
Example adapted from: Michie, D. (1961)

Bestärkendes Lernen

- War das nun ein guter Zug??
  - Wir wissen es immer noch nicht.



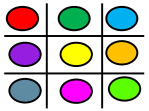
- Wir machen einfach weiter bis das Spiel zu Ende ist.
- Stellen wir uns vor wir haben gewonnen.



- Welcher Zug sicherte uns diesen Gewinn?
  - Jeder Zug den wir gemacht haben steuerte seinen Teil zu unserem Sieg bei.
  - Wir bestärken daher jeden ausgeführten Zug.

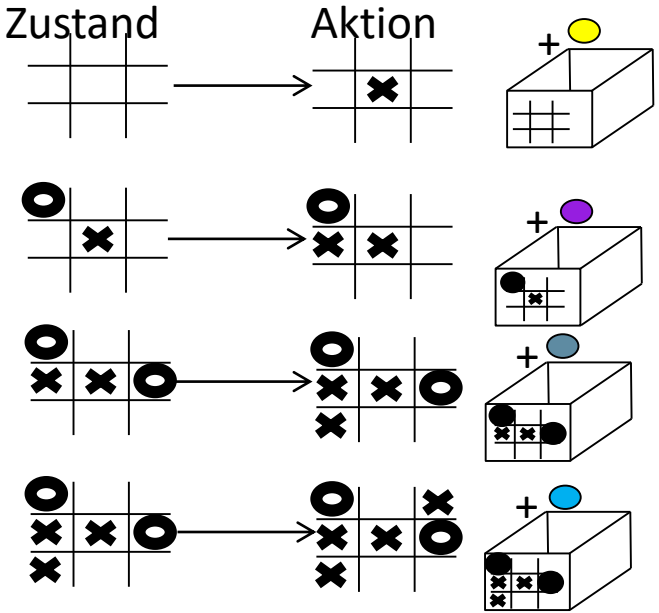
Bestärkendes Lernen

- Wie bestärken wir die ausgeführten Züge?
  - Jeder Zustand der im Spiel vorkam bekommt eine zusätzliche Murmel für die ausgeführte Aktion



- Je öfter wir spielen, je höher ist die Wahrscheinlichkeit gute Züge zu ziehen.

- Wir werden somit besser ohne Wissen darüber zu haben wie gut ein einzelner Zug ist.



Bestärkendes Lernen

- Wir können Züge bei verlorenen Spielen auch bestrafen.
  - Wir entnehmen eine Murmel für jede Aktion eines verlorenen Spiels.

- Nach vielen Spielen werden schlechtere Züge mit geringerer Wahrscheinlichkeit gezogen.

- Jedoch sollten wir uns gut überlegen ob wir die letzte Murmel einer Aktion aus der Schachtel entfernen, da dies die Aktion für immer aus dem Spiel nehmen würde.

Reinforcement Learning

- MENACE wurde wirklich praktisch umgesetzt

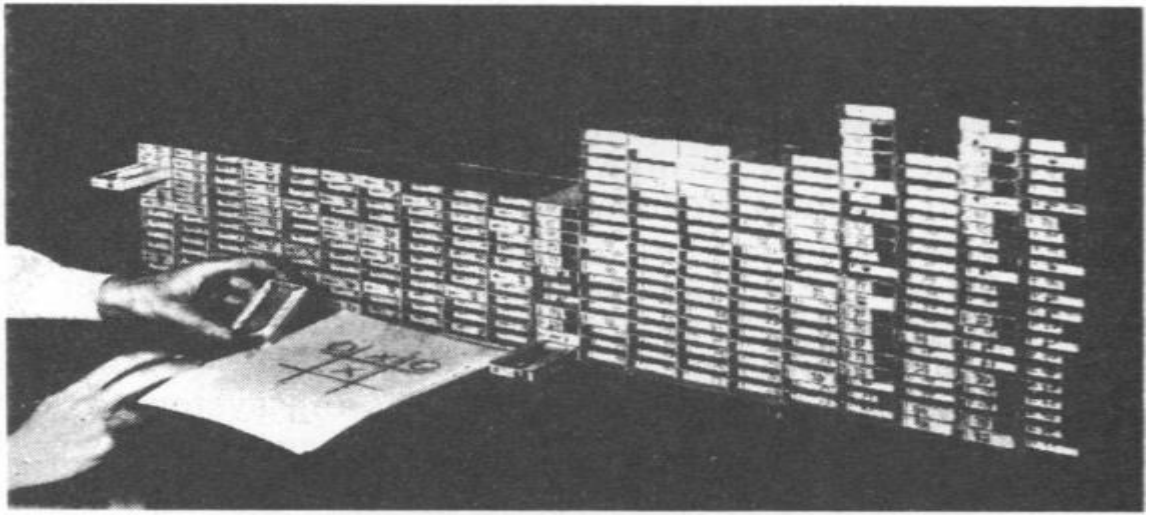
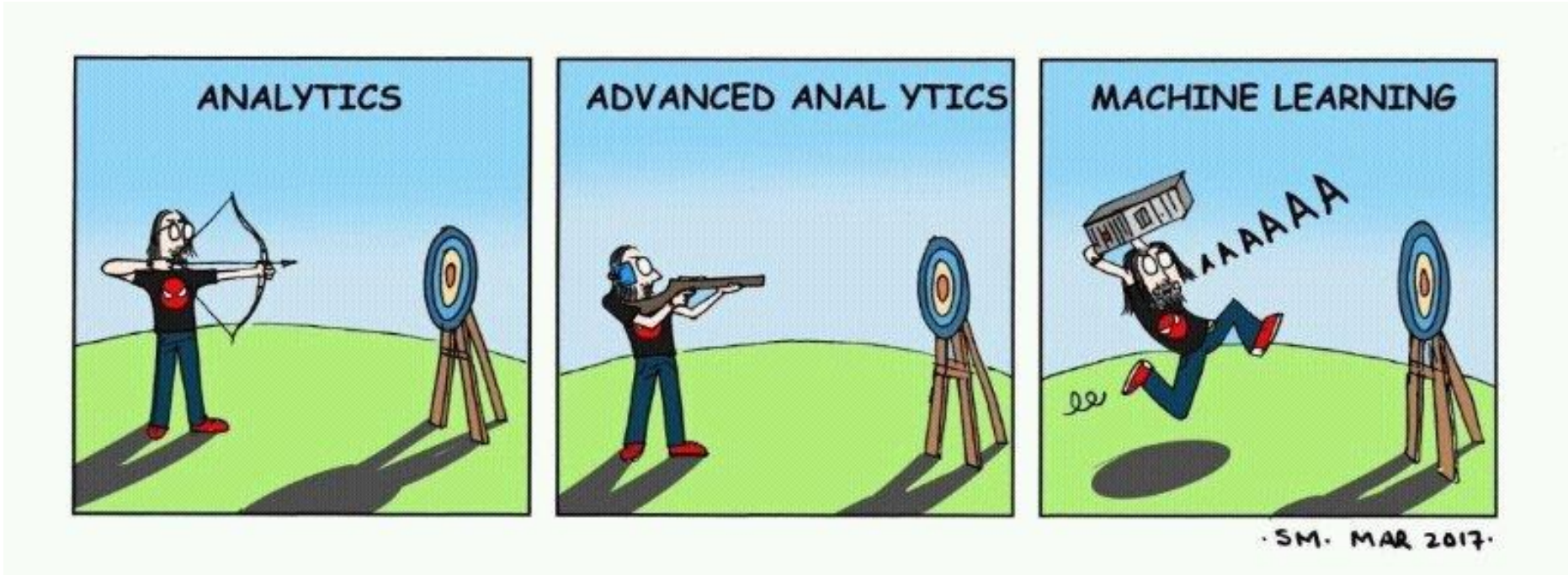


Fig. 2.—The matchbox machine—MENACE

Image from: Michie, D. (1961)

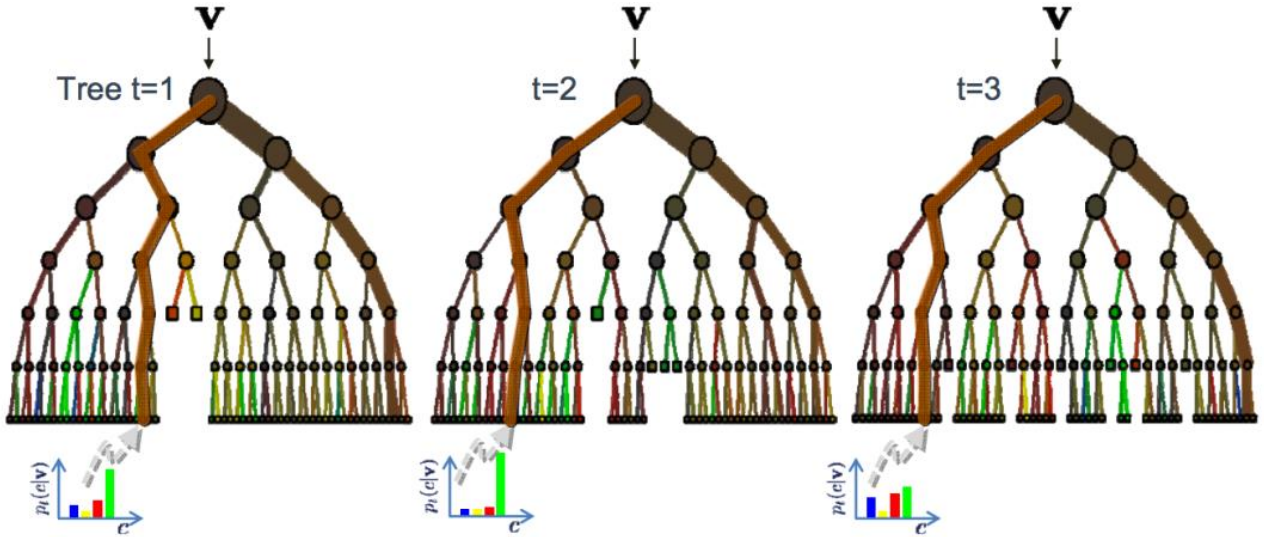
Ensembles

Maschinelles lernen im Zeitalter der Cloud



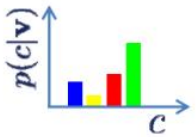
Ensembles

Maschinelles lernen im Zeitalter der Cloud



The ensemble model

Forest output probability $p(c|\mathbf{v}) = \frac{1}{T}\sum_{t}^{T} p_t(c|\mathbf{v})$



Buchempfehlungen

Russell, S., Norvig, P., Canny, J.,
Malik, J., & Edwards, D. (1995).
Artificial Intelligence: A Modern Approach.

Hagan, M. T., Demuth, H. B., &
Beale, M. H. (1995). **Neural Network Design**.
http://hagan.okstate.edu/NNDesign.pdf