Supervised Learning mit Python

Damian Murezzan

DIGICOMP

canoo
[delivering end-user happiness]

{ DEVDAY }
by DIGICOMP

Machine Learning

{ DEVDAY } by DIGICOMP

Scikit-Learn

Neuronale Netzwerke

Bilderkennung

Support Vector Machines

Klassifizierung

Overfitting

Tensorflow

Regression

Features

Regularisierung

DIGICOMP

canoo [delivering end-user happiness]

Machine Learning

{ DEVDAY } by DIGICOMP

Modelle:

Regressionsmodelle
Neuronale Netzwerke

Frameworks:

Scikit Learn
Tensorflow

Methoden:

Feature Auswahl
Regularisierung
Training/Testing/Overfitting

Anwendungen:

Regression / Klassifizierung
Bilderkennung

DIGICOMP canoo [delivering end-user happiness]

Daten

	Alter	# Bestellungen	…	Umsatz
Kunde 1	33	4	…	230
Kunde 2	26	1	…	60
Kunde 3	43	12	…	450
….	…	…	…	…

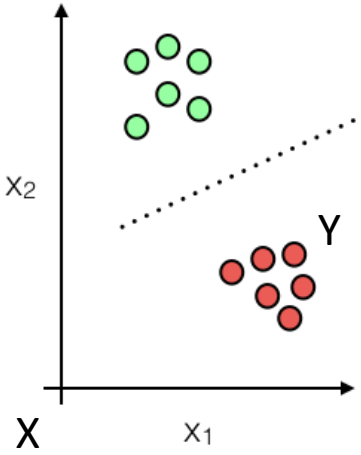
$$\begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,1} & x_{3,2} & x_{3,3} \\ x_{4,1} & x_{4,2} & x_{4,3} \\ x_{5,1} & x_{5,2} & x_{5,3} \end{bmatrix}$$

Daten X

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix}$$

Label/Target y

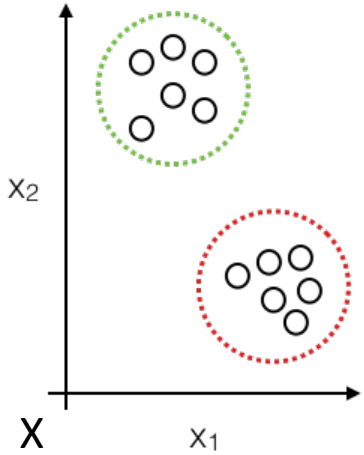
Arten von Machine Learning



Supervised

Daten und Labels vorhanden

Labels vorhersagen

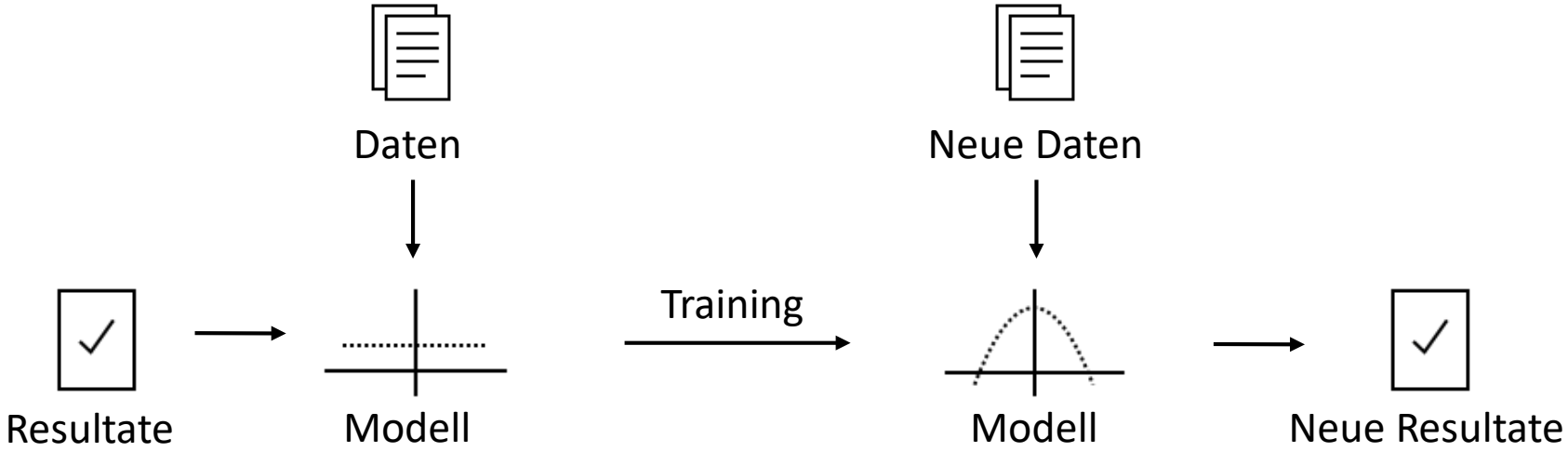


Unsupervised

Nur Daten vorhanden

Strukturerkennung

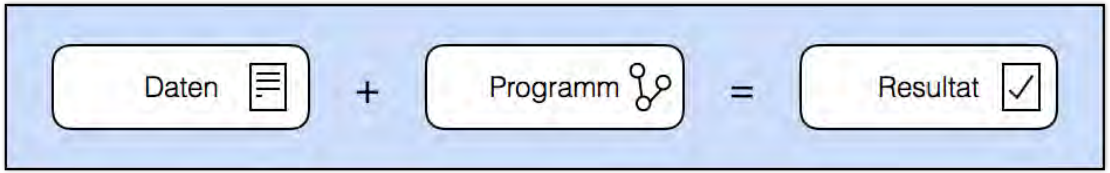
Supervised Learning



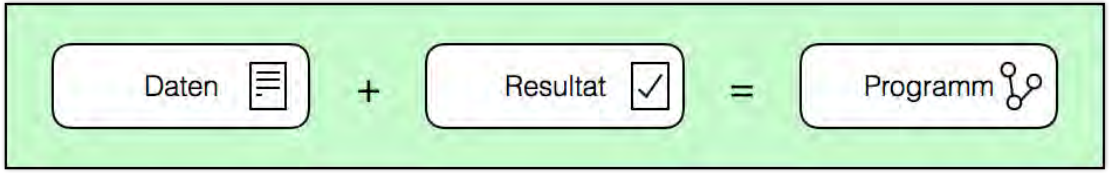
Machine Learning

{ DEVDAY } by DIGICOMP

Klassische Programmierung

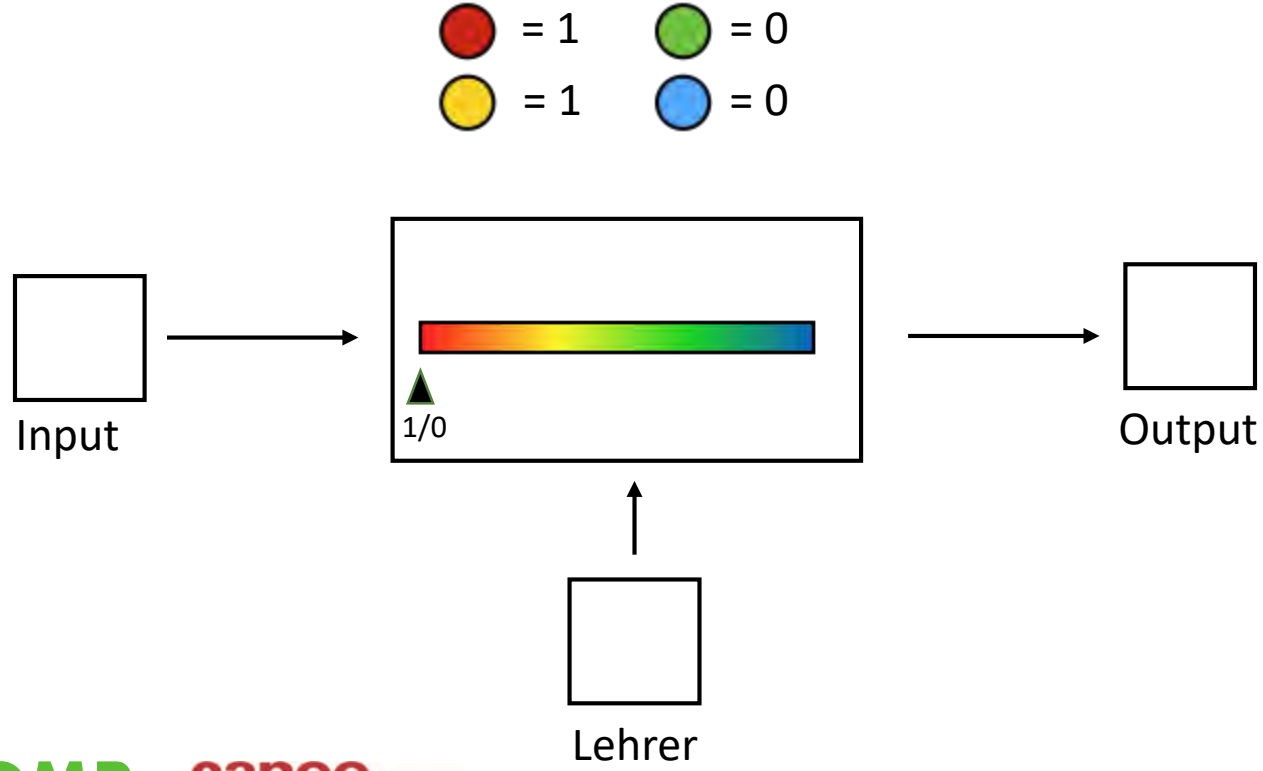


Machine Learning Paradigma

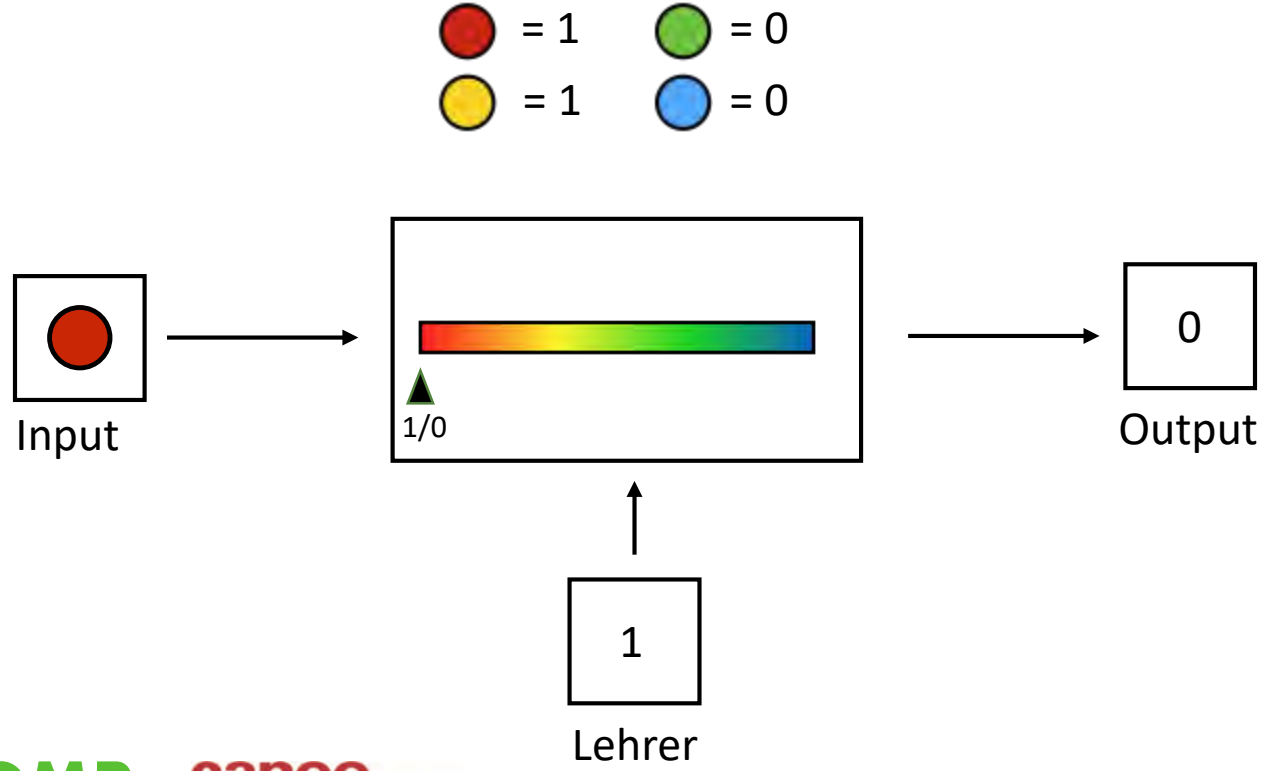


DIGICOMP canoo [delivering end-user happiness]

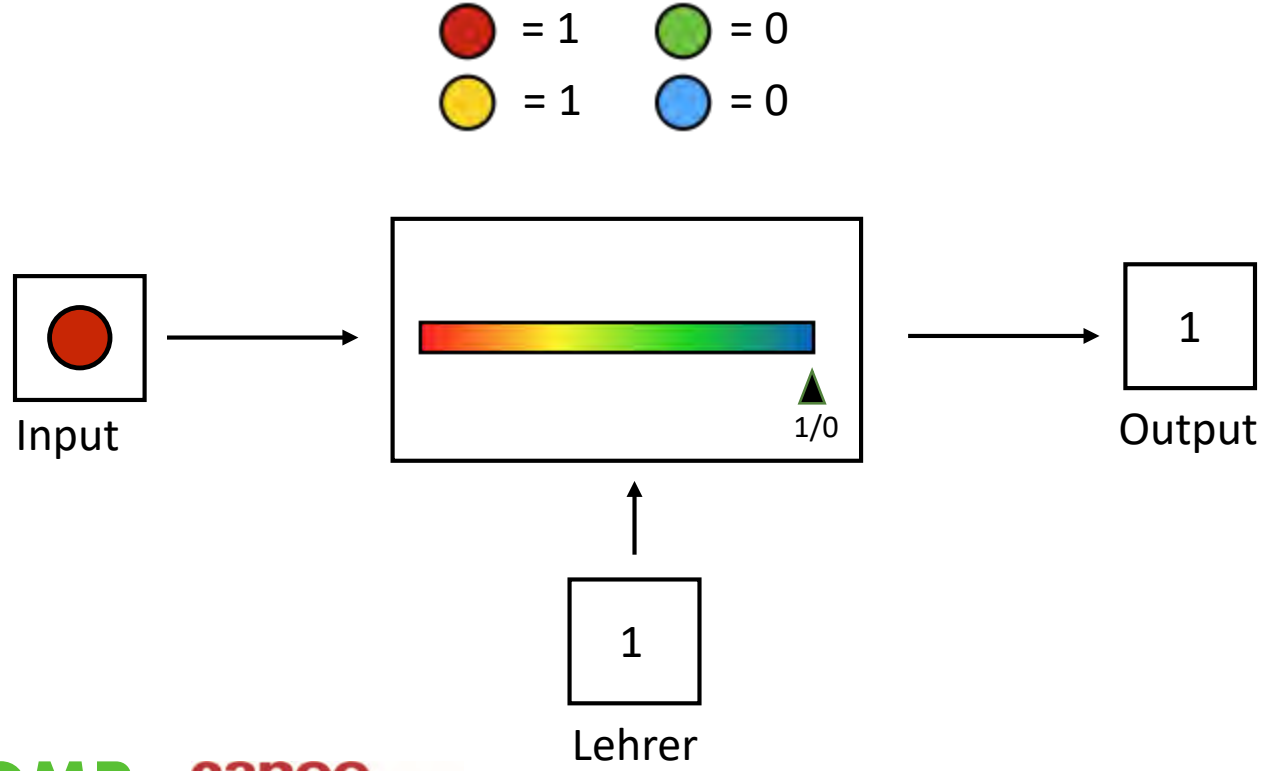
Wie trainiert man ein Machine Learning Modell?



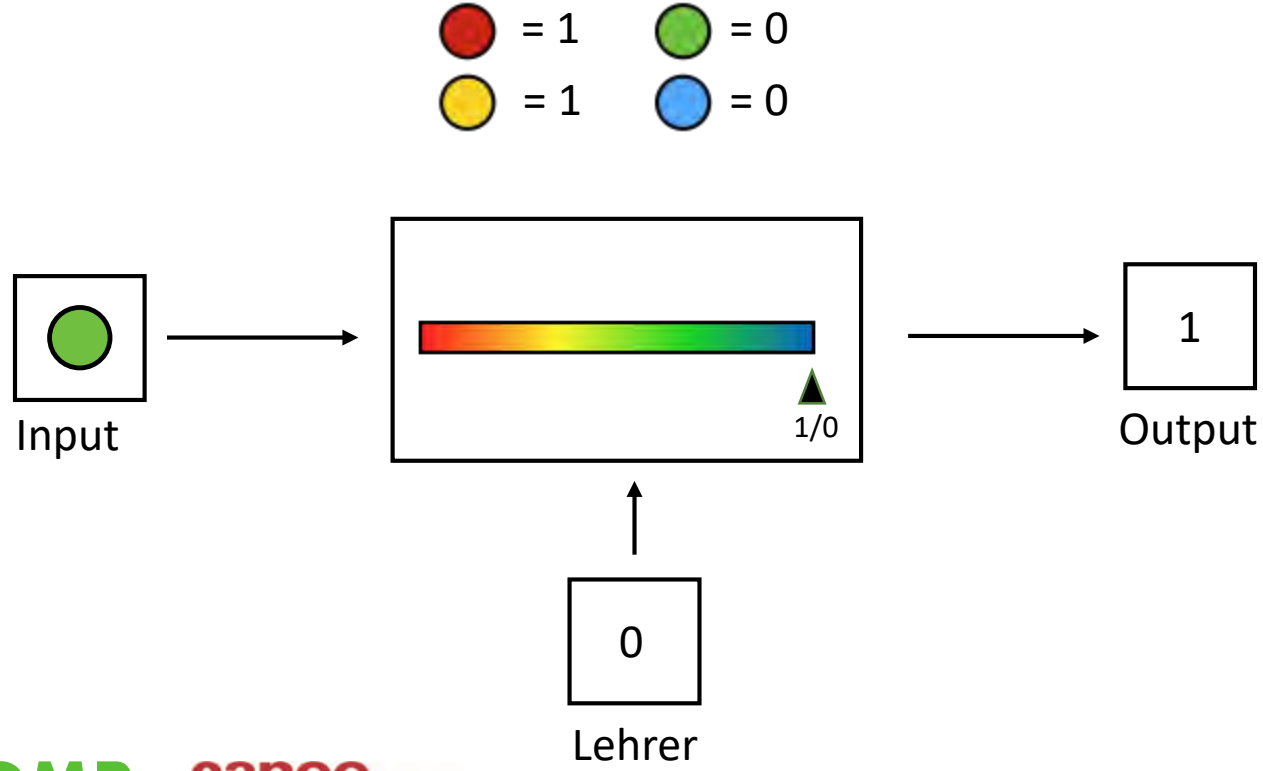
Wie trainiert man ein Machine Learning Modell?



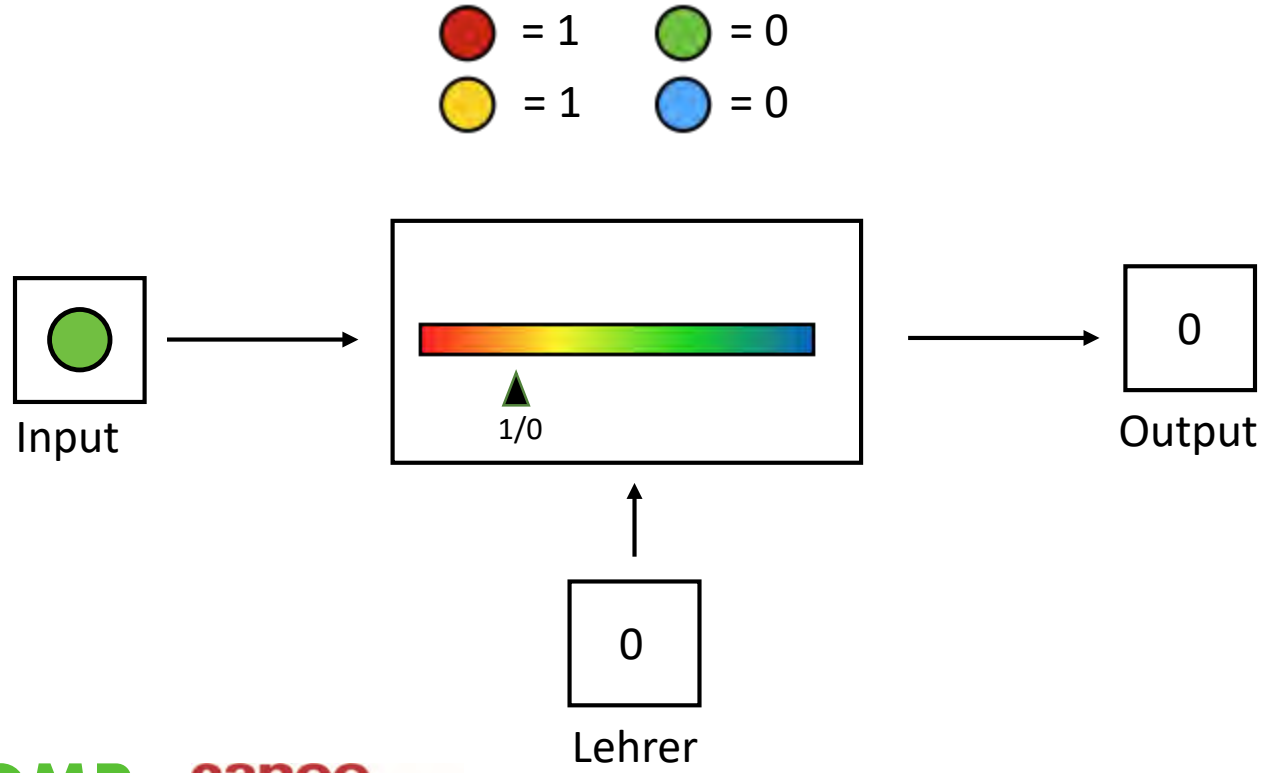
Wie trainiert man ein Machine Learning Modell?



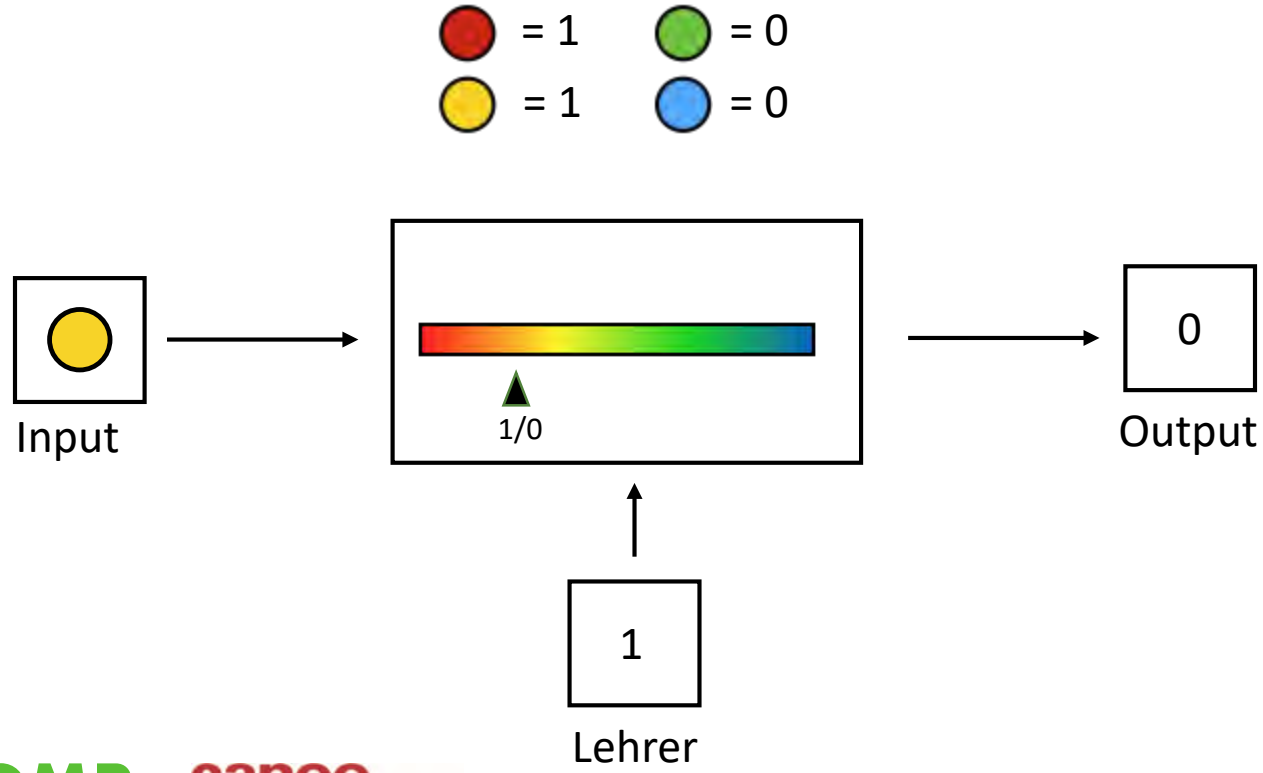
Wie trainiert man ein Machine Learning Modell?



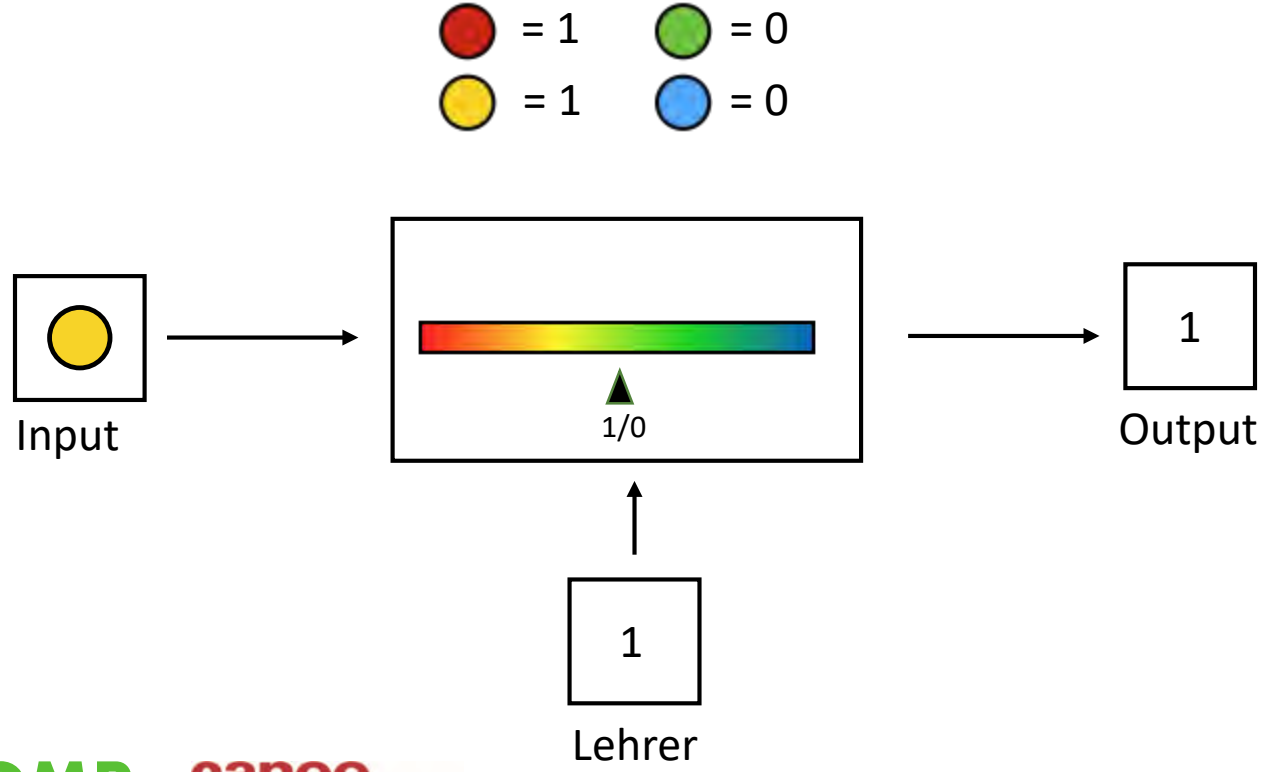
Wie trainiert man ein Machine Learning Modell?



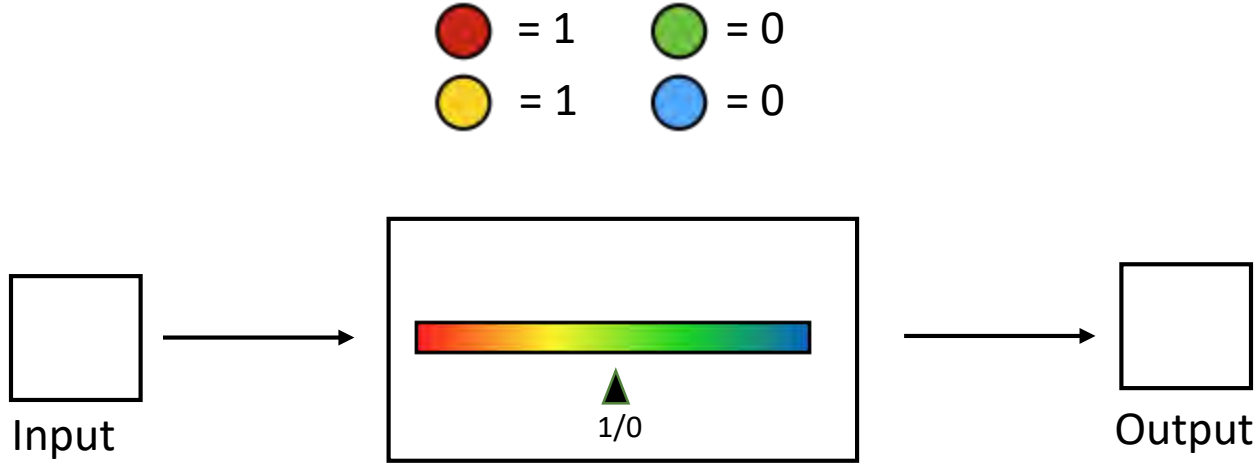
Wie trainiert man ein Machine Learning Modell?



Wie trainiert man ein Machine Learning Modell?

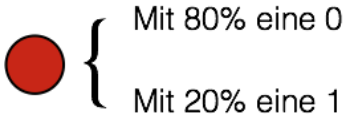


Wie trainiert man ein Machine Learning Modell?



Schwierigkeiten des Trainings

- Unsicherheit in den Daten/Labels



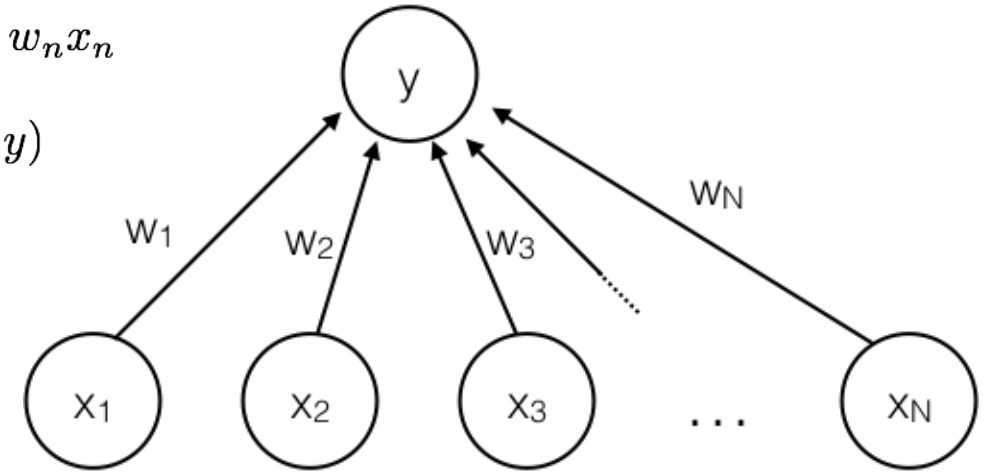
- Auswahl der Features

- Generalisierbarkeit

Lineare Regression

$$\hat{y} = w_1 x_1 + w_2 x_2 + ... + w_n x_n$$

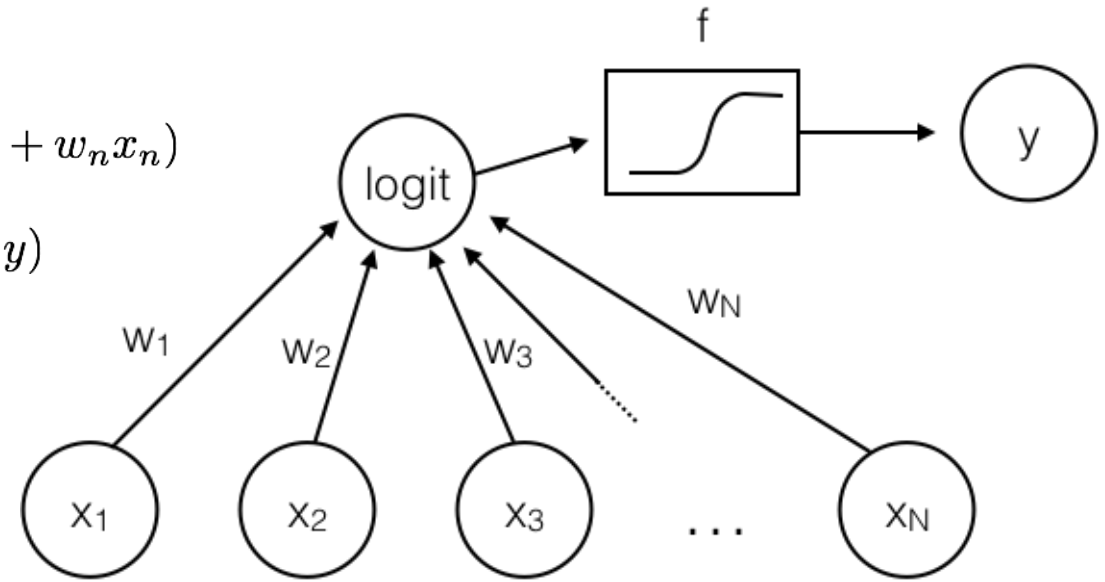
$$w^* = \text{argmin}_w \, \ell(\hat{y} - y)$$



Logistische Regression

$$\hat{y} = f(w_1 x_1 + w_2 x_2 + ... + w_n x_n)$$

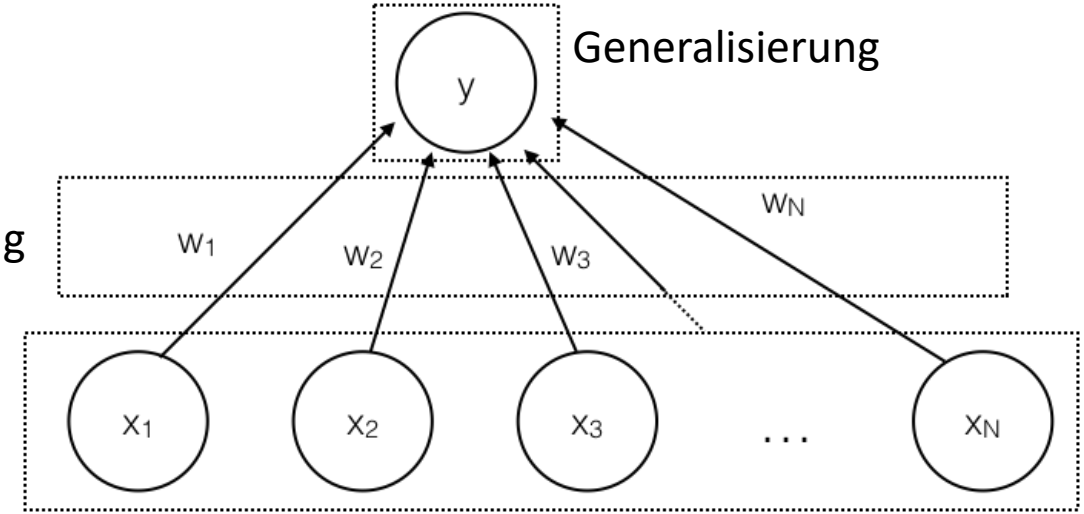
$$w^* = \text{argmin}_w \, \ell(\hat{y} - y)$$



Scikit-Learn

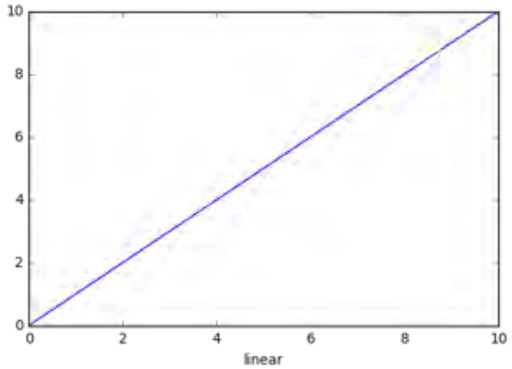
- Verschiedene Modelle vorhanden
  - Regressionsmodelle
  - Decision Trees
  - Support Vector Machine
  - …
- Erstellen eines Modells, fitten der Daten, vorhersage von neuen Daten
  - model = Model(Parameter)
  - model.fit(data, labels)
  - model.predict(new data)

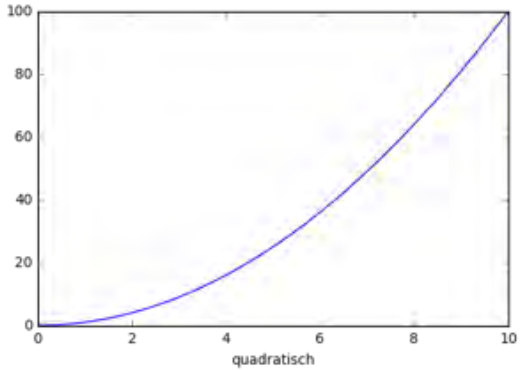
Modell Validieren und Erweitern



Feature Design

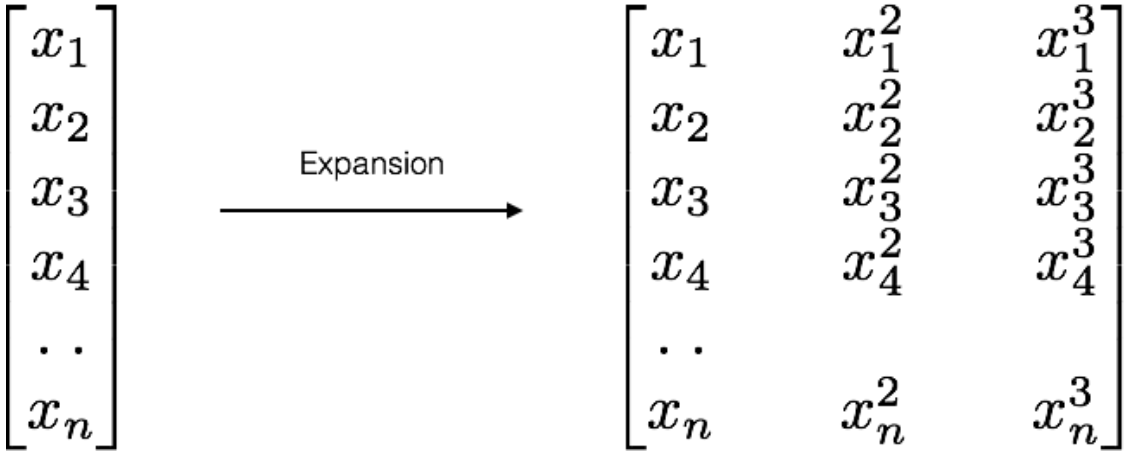
- Modell kann nur lernen, wenn die Zielvariable vom Input abhängig ist
- Abhängigkeit muss nicht immer linear sein





Feature Design

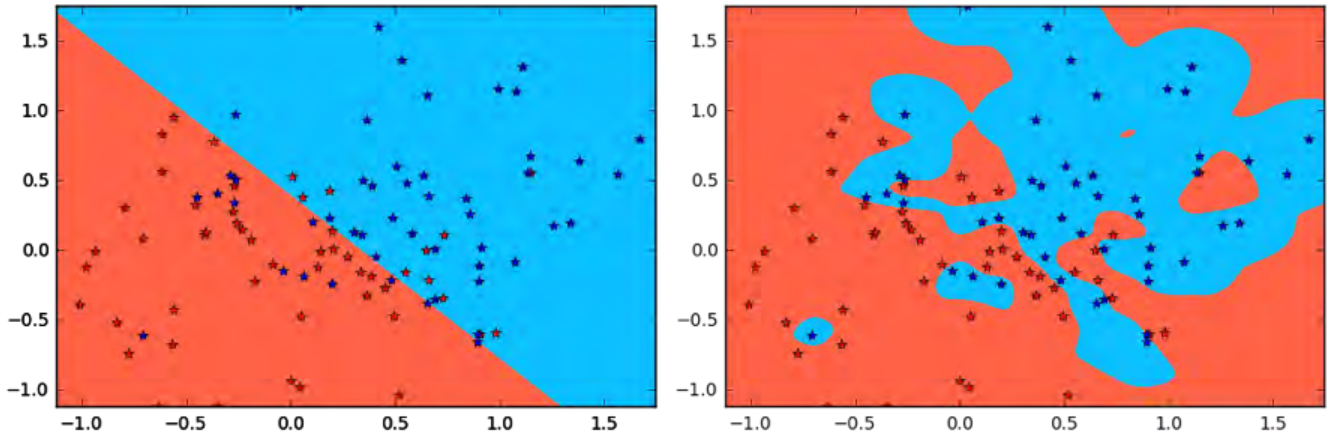
- Einfachster Fall: Feature Expansion



$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \cdot\ \cdot \\ x_n \end{bmatrix} \xrightarrow{\text{Expansion}} \begin{bmatrix} x_1 & x_1^2 & x_1^3 \\ x_2 & x_2^2 & x_2^3 \\ x_3 & x_3^2 & x_3^3 \\ x_4 & x_4^2 & x_4^3 \\ \cdot\ \cdot & & \\ x_n & x_n^2 & x_n^3 \end{bmatrix}$$

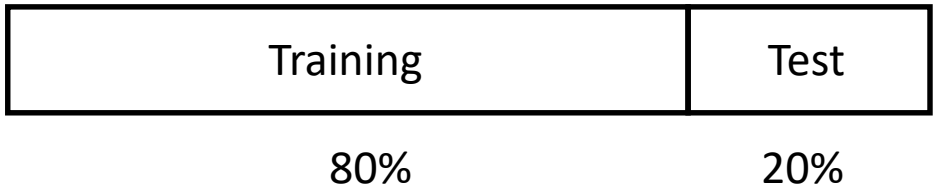
Training und Overfitting

- Je besser ein Modell die Trainingsdaten fitten kann, desto anfälliger ist es für overfitting
- Wenn ein Modell overfitted, wird es nicht in der Lage sein, neue Daten korrekt einzuordnen



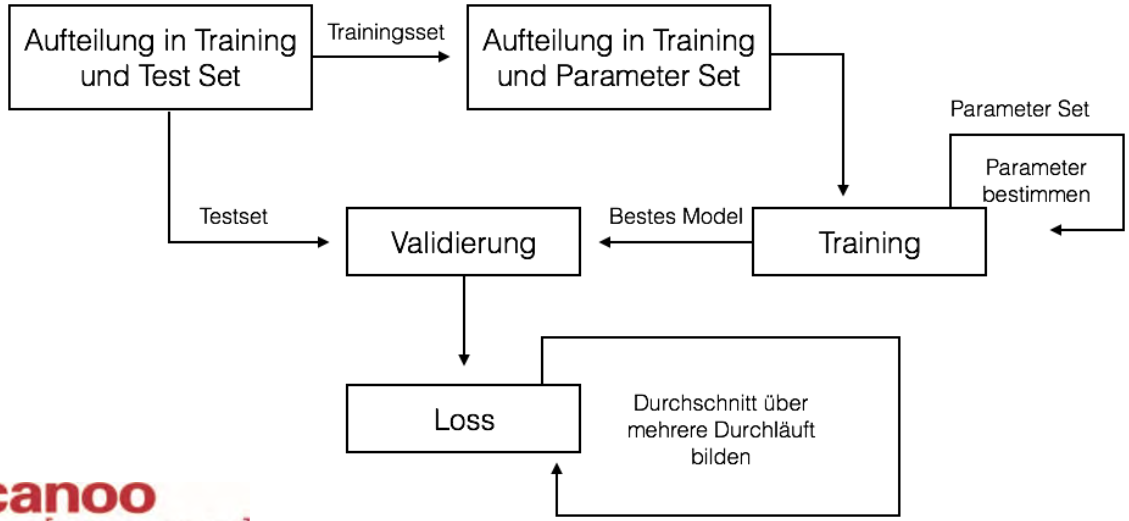
Training und Overfitting

- 1.) Datenset in Training und Testset aufteilen
- 2.) Modell auf Trainingsdaten trainieren
- 3.) Modell auf Testdaten validieren



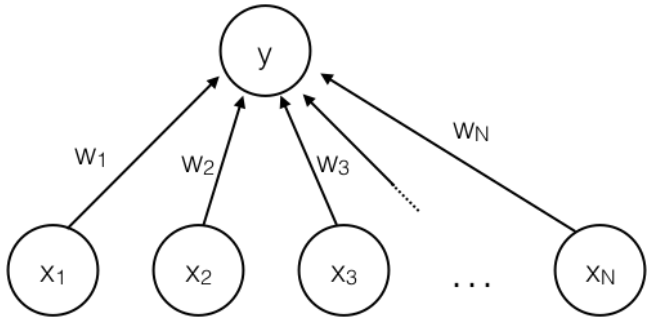
Cross Validation

- Ideal wenn nur begrenzte Menge an Daten vorhanden sind
- Modell wird mehrmals trainiert auf veschiedenen Splits von Training und Test Set
- Durchschnitt über mehrere Durchläufe



Regularisierung

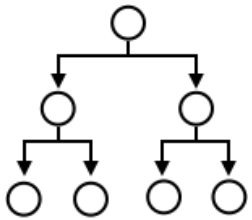
- Regularisierung hilft, Flexibilität der Modelle einzustellen
- Kann helfen, Struktur in den Daten zu nutzen
- Ridge Regression: Führt zu Modellen, die weniger Overfitten
- Lasso Regression: Führt dazu, dass unwichtige Gewichte auf 0 gesetzt werden (Selektion)



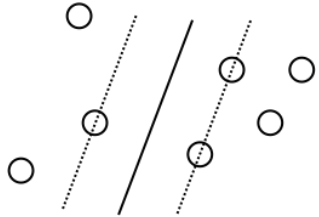
Gültig für alle ML Modelle

{ DEVDAY } by DIGICOMP

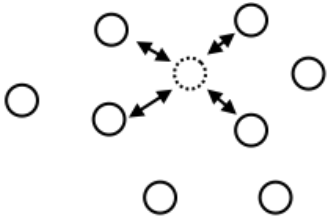
Decision Trees



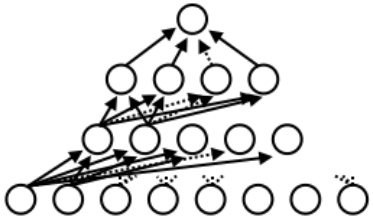
Support Vector Machines



Nearest Neighbours

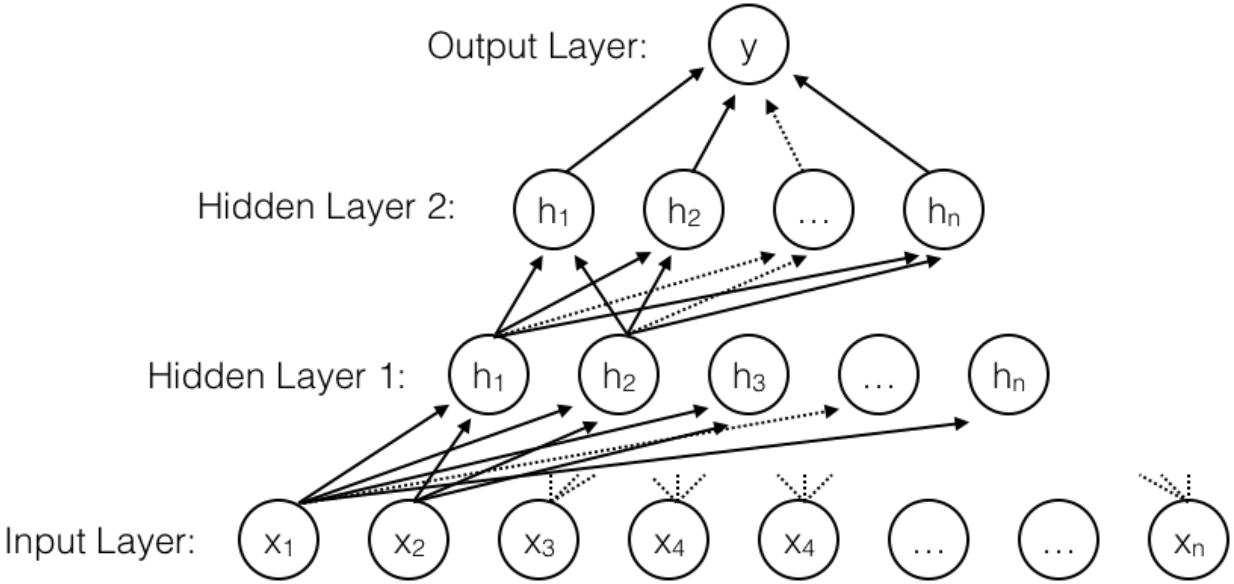


Artificial Neural Networks

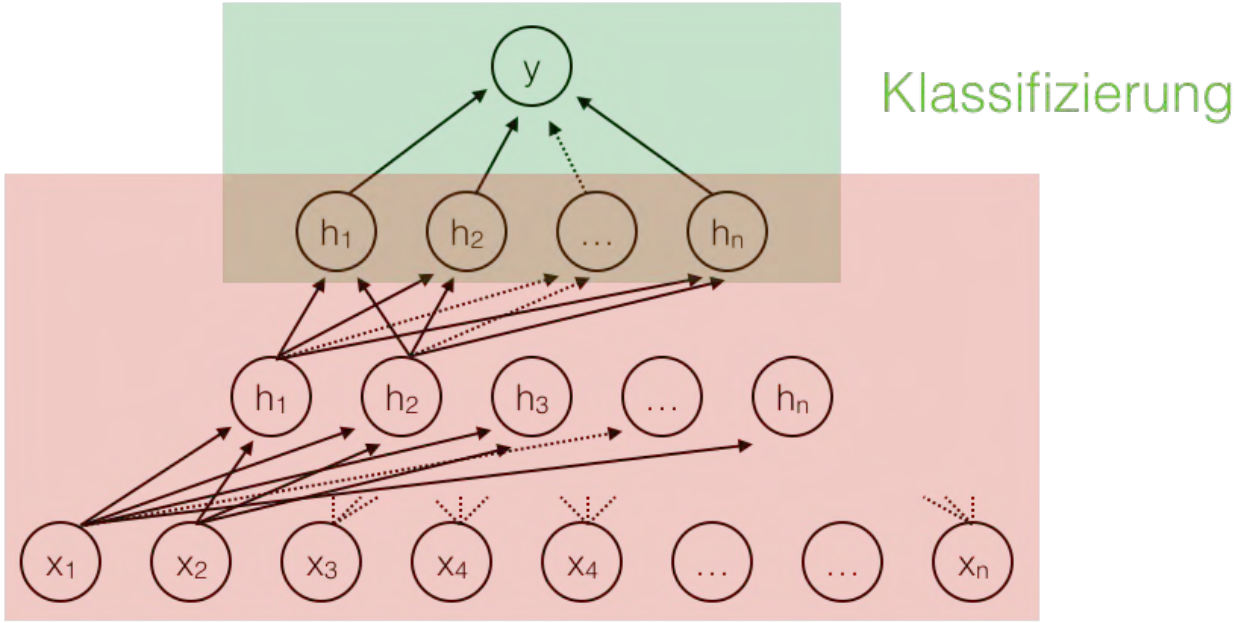


DIGICOMP canoo [delivering end-user happiness]

Artificial Neural Networks



Artificial Neural Networks



Artificial Neural Networks

{ DEVDAY } by DIGICOMP

- Gestapelte Regressionen
- Grosse Modelle mit vielen Parametern
- Sehr gute Ergebnisse auf traditionell schwierigen Daten, e.g. Bildern

- Schwierig zu Interpretieren: „Black Box“ Ansatz
- Training ist nicht einfach, viele Daten vonnöten
- Bereits Trainierte Netzwerke sind meistens erhältlich!

Wann braucht man ANNs?

{ DEVDAY } by DIGICOMP

- Schwierige Daten:
  - Einzelne Datenpunkte wenig Informativ
  - Effiziente Features sind nicht einfach zu erstellen
- Beispiel Bilder: Convnets sind optimierte Neuronale Netzwerke, die Gewichte räumlich teilen können
- Beispiel Zeitreihen: Recurrent Neural Networks teilen Gewichte über die Zeit

Tensorflow

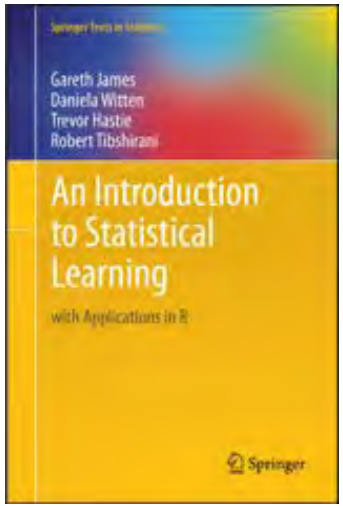
- Framework für das erstellen von Artifical Neural Networks
- Einfaches Training
- Bereits Trainierte Netzwerke vorhanden
- Intuitive API mit Keras

Zusammenfassung

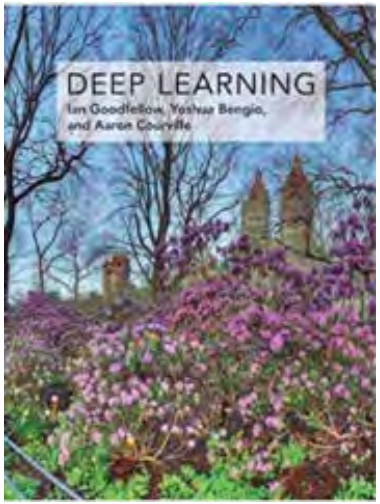
{ DEVDAY } by DIGICOMP

- Für Supervised Learning sind bereits gute Frameworks verfügbar
- Training und testen der Modelle sind essenziell
  - Feature Design
  - Overfitting
  - Regularisierung
- Neuronale Netzwerke sind bereits vortrainiert erhältlich und erlauben das Klassifizieren von schwierigen Daten wie Bildern

Bücher



http://www-bcf.usc.edu/~gareth/ISL/



http://www.deeplearningbook.org

Fragen?