INF FAKULTÄT FÜR INFORMATIK

Intelligente Systeme

Maschinelles Lernen

Prof. Dr. R. Kruse **C. Braune**

{rudolf.kruse,christian.braune}@ovgu.de

Institut für Intelligente Kooperierende Systeme
Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg

Übersicht

Most people think computers will never be able to think.
Marvin Minsky, 1982

Definitionen des Lernens (1)

Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time.[Simon, 1983]

Umfasst allerdings auch Veränderungen, die nichts mit Lernen zu tun haben

Beispiel einer Lernleistung: Verwendung eines schneller getakteten Prozessors als schnellere Abarbeitung einer arithmetischen Berechnung

Definitionen des Lernens (2)

The study and computer modeling von learning processes in their multiple manifestations constitutes the subject matter von machine learning.[Michalski et al., 1986]

Direkte Anspielung auf „Lernprozesse in ihren verschiedenen Erscheinungsformen“

Definitionen des Lernens (3)

Learning is constructing or modifying representations von what is being experienced. [Michalski, 1986]

Zentraler Aspekt: Konstruktion einer Repräsentation

Definitionen des Lernens (4)

Research in machine learning has been concerned with building computer programs able to construct new knowledge or to improve already possessed knowledge by using input information.[Michalski and Kodratoff, 1990]

Ziel des ML: Computerprogramme sollen durch Erfahrung ihr eigenes Handeln verbessern können

Lernmodell

Performanzelement: interagiert mit der Umgebung, wird durch vorhandenes Wissen gesteuert

Lernelement: nimmt Erfahrungen und Beobachtungen aus der Umgebung auf, erzeugt/modifiziert Wissen

Zusätzlich meist:

Kritikelement: teilt dem Lernelement mit, wie erfolgreich es ist

Problemgenerator: erzeugt Aufgaben, die zu neuen und informativen Erfahrungen führen sollen

Klassifikation der Ansätze

Klassifikation gemäß der zugrundeliegenden *Lernstrategie:*
Unterscheidung wie viel Information bereits vorgegeben wird und in welchem Maße das Lernsystem eigene Inferenzen durchführt

Klassifikation gemäß der benutzten *Repräsentation von Wissen*, welches das System erlernt

Klassifikation gemäß dem *Anwendungsbereich* des Lernsystems

Klassifikation gemäß der benutzten Lernstrategie

Direkte Eingabe neuen Wissens und Auswendiglernen:

Keinerlei Inferenz oder andere Art der Wissenstransformation erforderlich

z.B. Speichern von Daten/Fakten, Lernen durch direkte Programmierung

Lernen durch Anweisungen:

Aufbereitetes Wissen wird vorgegeben, was intern verarbeitet werden muss

Wissen soll effektiv verwendet werden

Anweisungen werden durch den Lehrenden aufgearbeitet, so dass Wissen des Lernenden schrittweise erweitert werden kann

Klassifikation gemäß der benutzten Lernstrategie

Lernen durch Deduktion:

Leitet aus vorhandenem Wissen mittels deduktiver Schlussweisen neues Wissen ab

Neues Wissen kann zur Effizienz- oder Effektivitätssteigerung verwendet werden

Lernen durch Analogie:

Erlernen neuer Fakten und Fähigkeiten durch Anpassung vorhandenen Wissens an neue Situationen

Klassifikation gemäß der benutzten Lernstrategie

Lernen aus Beispielen:

Allgemeine Konzeptbeschreibung soll erstellt werden, die alle vorher gegebenen Beispiele umfasst und evtl. vorhandene Gegenbeispiele ausschließt

- *Beispiele vom Lehrenden:* Konzept ist dem Lehrer bekannt; Beispiele können entsprechend ausgewählt werden; schneller Lernerfolg möglich
- *Beispiele vom Lernenden:* Lernender hat Hypothese für das zu lernende Konzept und generiert Beispiele; von außerhalb kommt Feedback zu den Beispielen (positive oder negative Beispiele)
- *Beispiele aus der Umgebung:* Zufallsbeobachtungen; Notwendigkeit, dem Lernenden mitzuteilen, ob die Beobachtung ein positives oder ein Gegenbeispiel ist

Klassifikation gemäß der benutzten Lernstrategie

Alternative Klassifizierung des „Lernens durch Beispiele“:

Nur positive Beispiele verfügbar: keine Informationen darüber verfügbar, ob abgeleitetes Konzept zu allgemein ist; dem wird oft durch Minimalitätskriterien entgegenzuwirken versucht

Positive und negative Beispiele verfügbar: üblichste Situation beim Lernen; positive Beispiele sorgen dafür, dass abgeleitetes Konzept allgemein genug ist; negative Beispiele verhindern, dass das Konzept zu allgemein wird

Klassifikation gemäß der benutzten Lernstrategie

Weitere alternative Klassifizierung des „Lernens durch Beispiele“:

Alle Beispiele gleichzeitig: alle Informationen stehen in jedem Fall am Anfang zur Verfügung; Hypothesen können sofort auf Richtigkeit überprüft werden

Beispiele sind inkrementell gegeben: Hypothese in Konsistenz mit den bisherigen Beispielen wird erstellt, die keines der Gegenbeispiele erfasst; anhand nachfolgender Beispiele wird Hypothese überprüft und ggf. verfeinert

Klassifikation gemäß der benutzten Lernstrategie

Lernen aus Beobachtungen und durch Entdeckungen: generelle Ausprägung des induktiven Lernens; keinerlei Steuerung durch Lehrenden; verschiedene Konzepte sind gleichzeitig zu erlernen

- *Passive Beobachtungen:* Konzepte, die aufgrund der Beobachtungen der Umgebung durch den Lernenden entwickelt werden;
- *Aktive Experimente:* Umgebung wird gezielt beeinflusst, um die Auswirkungen der Experimente beobachten zu können; Steuerung der Experimente per Zufall, nach allgemeinen Gesichtspunkten oder durch theoretische Überlegungen.

Klassifikation gemäß dem gelernten Typ von Wissen

Parameter in algebraischen Ausdrücken: gegeben ist ein algebraischer Ausdruck; numerische Parameter oder Koeffizienten sind so zu optimieren, dass ein gewünschtes Verhalten erreicht wird

Entscheidungsbäume: zur Unterscheidung zwischen Elementen einer Klasse; Knoten: Attribute der Objekte; Blätter: Menge der Objekte, die der gleichen Unterklasse zugeordnet werden

Formale Grammatiken: zur Beschreibung einer formalen Sprache; ausgehend von Beispielausdrücken wird eine formale Grammatik erlernt

Klassifikation gemäß dem gelernten Typ von Wissen

Regeln: **if** C **then** A; C ist Menge von Bedingungen, A ist eine Aussage

Ausdrücke basierend auf formaler Logik: für die Beschreibung einzelner Objekte als auch für die Bildung des zu erlernden Konzepts; Aussagen, Prädikate, Variablen, logische Ausdrücke

Begriffshierarchien: Begriffe, die in Beziehung zueinander stehen, werden hierarchischen Begriffskategorien zugeordnet; Begriffshierarchien bzw. Taxonomien sind zu lernen

Entscheidungsbäume

Hier: vereinfachte Entscheidungsbäume, die nur ja/nein-Knoten beinhalten.

- Die Blätter sind mit dem Wahrheitswert markiert, der als Ergebnis der Funktion zurückgeliefert werden soll, wenn das Blatt erreicht wird.
- Die inneren Knoten sind mit einem Attribut markiert. Eine solche Markierung *a* repräsentiert eine Abfrage, welchen Wert das betrachtete Objekt für das Attribut *a* hat.
- Die von einem mit *a* markierten Knoten ausgehenden Kanten sind mit den zu *a* möglichen Attributwerten markiert.

Entscheidungsbäume: Beispiel

Beispiel: Kinobesuch, Definition der Attribute

Attraktivität des Films: hoch, mittel, gering

Preis: normal (n), mit Zuschlag (z)

Loge: noch verfügbar (ja), nicht verfügbar (nein)

Wetter: schön, mittel, schlecht

Warten: ja, nein

Besetzung: top, mittel

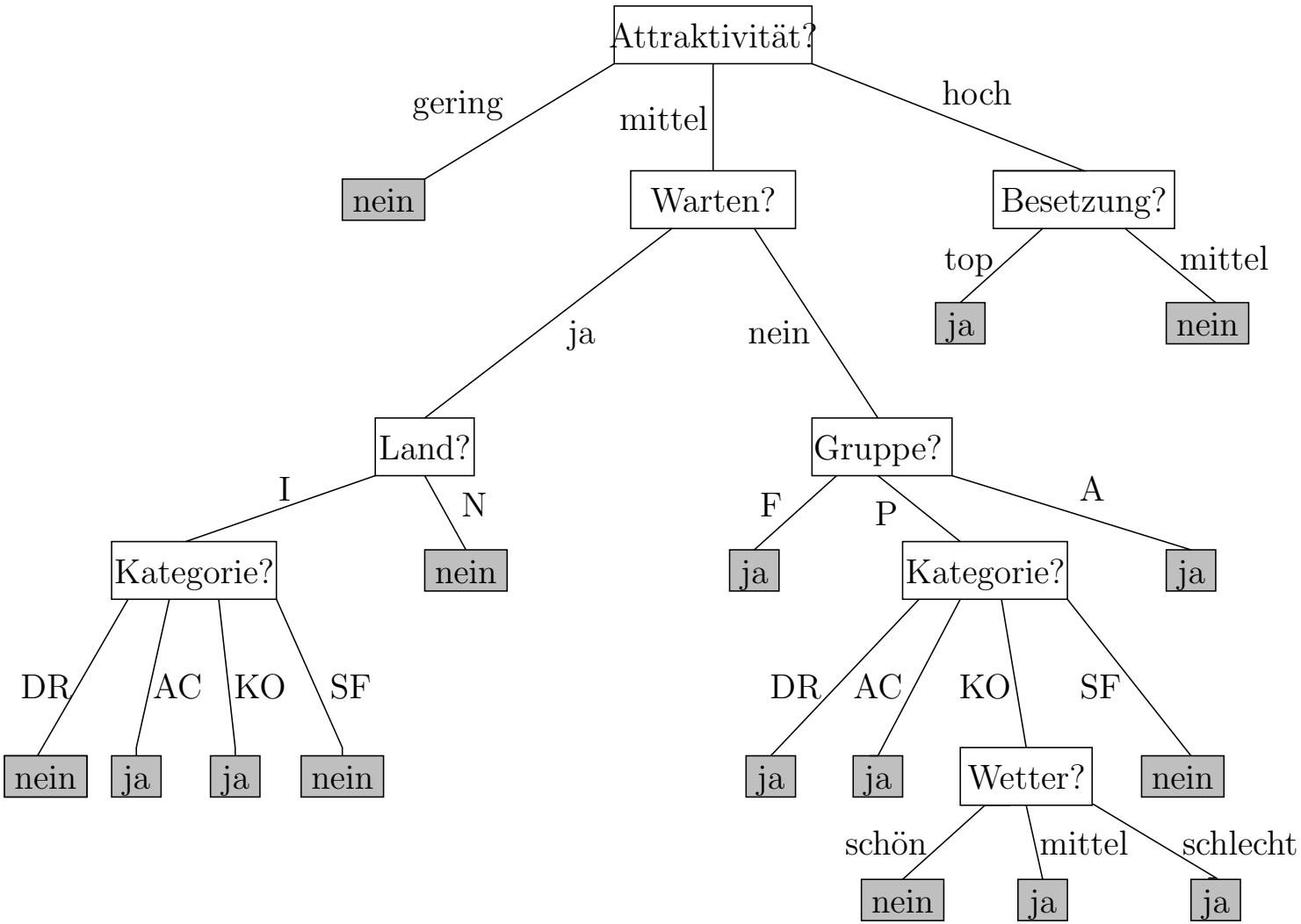
Kategorie: Action (AC), Komödie (KO), Drama (DR), Science Fiction (SF)

Reservierung: ja, nein

Land: national (N), international (I)

Gruppe: Freunde (F), Paar (P), allein (A)

Entscheidungsbäume: Beispiel



Entscheidungsbäume: Regeln

Aus Entscheidungsbäumen können sehr einfach Regeln abgelesen werden:

if Attraktivität = hoch **and** Besetzung = top **then** Kinobesuch = ja.

if Attraktivität = mittel **and** Warten = ja **and** Land = national **then** Kinobesuch = nein.

Ein Lernverfahren für Entscheidungsbäume generiert aus einer Menge von Beispielen (der *Trainingsmenge*) einen Entscheidungsbaum. Ein *Trainingsbeispiel* ist dabei eine Menge von Attribut/Wert-Paaren zusammen mit der Klassifikation.

Entscheidungsbäume: Generierung

Für jedes Beispiel steht am Ende genau ein Pfad im Baum von der Wurzel zum Blattknoten.

Diese Vorgehensweise liefert keine sinnvolle Generalisierung, der Baum passt nur auf die vorhandenen Trainingsdaten, aber nicht auf neue Daten.

Occam's Razor: Bevorzuge die einfachste Hypothese, die konsistent mit allen Beobachtungen ist.

Problem: Welches Attribut wird ausgewählt, um in einem Knoten die Beispieldaten aufzuteilen? Welches Attribut ist das *wichtigste*?

Beispiel: Induktion eines Entscheidungsbaums

Patienten-Datenbank

12 Beispielfälle

3 beschreibende Attribute

1 Klassenattribut

Bestimmung des Medikaments M

(ohne Patientenattribute)
immer Medikament A oder immer Medikament B:
50% korrekt (in 6 v. 12 Fällen)

Nr	Geschl.	Alt.	Blutdr.	M.
1	männlich	20	normal	A
2	weiblich	73	normal	B
3	weiblich	37	hoch	A
4	männlich	33	niedrig	B
5	weiblich	48	hoch	A
6	männlich	29	normal	A
7	weiblich	52	normal	B
8	männlich	42	niedrig	B
9	männlich	61	normal	B
10	weiblich	30	normal	A
11	weiblich	26	niedrig	B
12	männlich	54	hoch	A

Beispiel: Induktion eines Entscheidungsbaums

Geschlecht eines Patienten

Teilung bzgl. männlich/weiblich

Bestimmung des Medikaments

männlich:	50% korrekt	(in 3 von 6 Fällen)
weiblich:	50% korrekt	(in 3 von 6 Fällen)
gesamt:	**50% korrekt**	(in 6 von 12 Fällen)

Nr	Geschl.	M.
1	männlich	A
6	männlich	A
12	männlich	A
4	männlich	B
8	männlich	B
9	männlich	B
3	weiblich	A
5	weiblich	A
10	weiblich	A
2	weiblich	B
7	weiblich	B
11	weiblich	B

Beispiel: Induktion eines Entscheidungsbaums

Alter des Patienten

Sortieren anhand des Alters

beste Teilung finden, hier: ca. 40 Jahre

Bestimmung des Medikaments

≤ 40:	A	67% korrekt	(in 4 von 6 Fällen)
> 40:	B	67% korrekt	(in 4 von 6 Fällen)
gesamt:		**67% korrekt**	(in 8 von 12 Fällen)

Nr	Alt.	M.
1	20	A
11	26	B
6	29	A
10	30	A
4	33	B
3	37	A
8	42	B
5	48	A
7	52	B
12	54	A
9	61	B
2	73	B

Beispiel: Induktion eines Entscheidungsbaums

Blutdruck des Patienten

Teilung bzgl. hoch/normal/niedrig

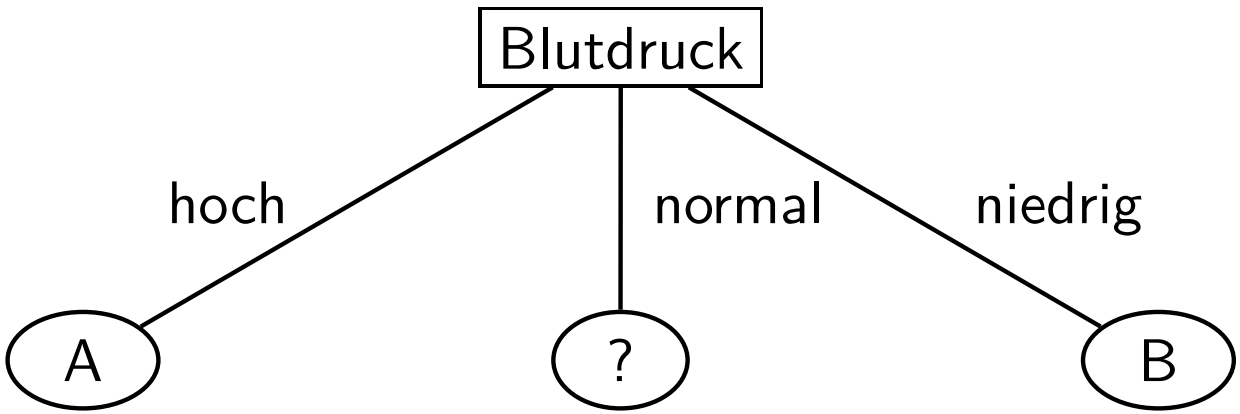
Bestimmung des Medikaments

hoch:	A	100% korrekt	(3 von 3)
normal:		50% korrekt	(3 von 6)
niedrig:	B	100% korrekt	(3 von 3)
gesamt:		**75% korrekt**	(9 von 12)

Nr	Blutdr.	M.
3	hoch	A
5	hoch	A
12	hoch	A
1	normal	A
6	normal	A
10	normal	A
2	normal	B
7	normal	B
9	normal	B
4	niedrig	B
8	niedrig	B
11	niedrig	B

Beispiel: Induktion eines Entscheidungsbaums

Momentaner Entscheidungsbaum:



Beispiel: Induktion eines Entscheidungsbaums

Blutdruck und Geschlecht

nur Patienten mit normalem Blutdruck

Zerlegung bzgl. männlich/weiblich

Bestimmung des Medikaments

männlich:	A	67% korrekt	(2 von 3)
weiblich:	B	67% korrekt	(2 von 3)
gesamt:		**67% korrekt**	(4 von 6)

Nr	Blutdr.	Geschl.	M.
3	hoch		A
5	hoch		A
12	hoch		A
1	normal	männlich	A
6	normal	männlich	A
9	normal	männlich	B
2	normal	weiblich	B
7	normal	weiblich	B
10	normal	weiblich	A
4	niedrig		B
8	niedrig		B
11	niedrig		B

Beispiel: Induktion eines Entscheidungsbaums

Blutdruck und Alter

nur Patienten mit normalem Blutdruck

Sortieren anhand des Alters

beste Teilung finden, hier: ca. 40 Jahre

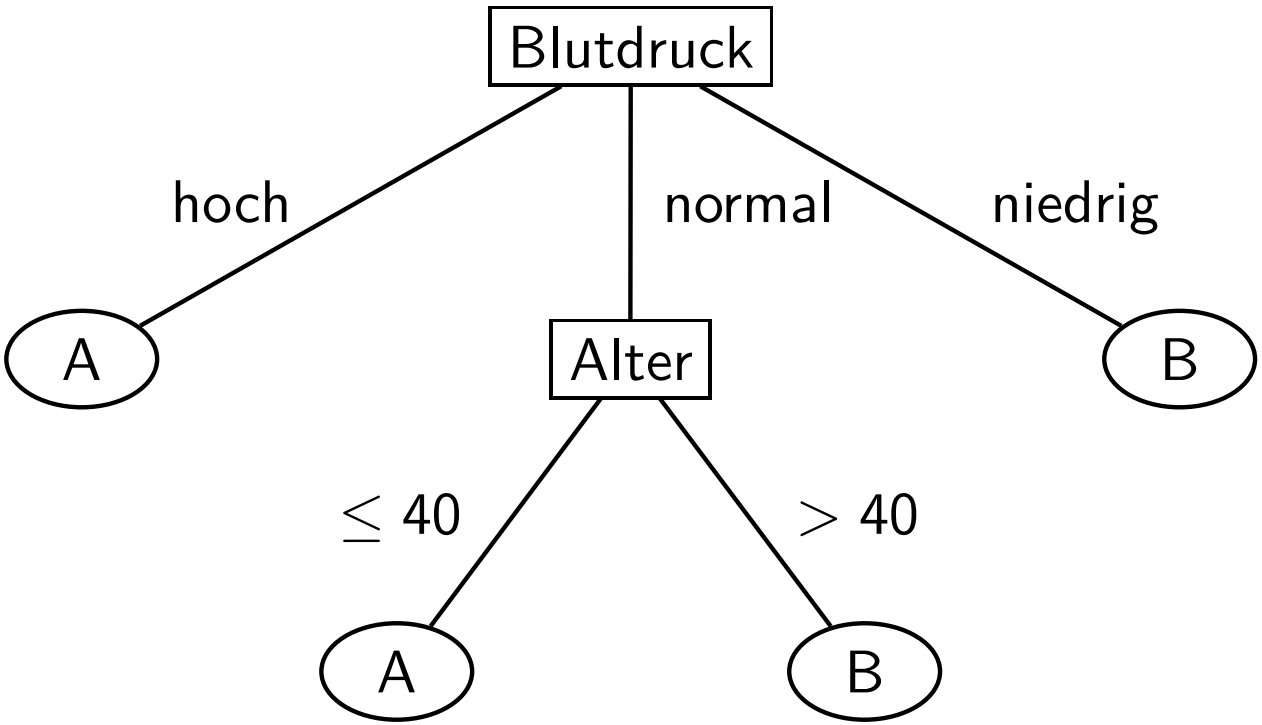
Bestimmung des Medikaments

≤ 40:	A	100% korrekt	(3 von 3)
> 40:	B	100% korrekt	(3 von 3)
gesamt:		**100% korrekt**	(6 von 6)

Nr.	Blutdr.	Alt.	M.
3	hoch		A
5	hoch		A
12	hoch		A
1	normal	20	A
6	normal	29	A
10	normal	30	A
7	normal	52	B
9	normal	61	B
2	normal	73	B
11	niedrig		B
4	niedrig		B
8	niedrig		B

Ergebnis nach Lernen des Entscheidungsbaums

Bestimmung eines Medikaments für einem Patienten:



Bewertungsmaße

im vorherigen Beispiel:
Rate der korrekt klassifizierten Beispielfälle

- Vorteil: durch Auszählen berechenbar, leicht nachvollziehbar
- Nachteil: bei mehr als zwei Klassen, wird die Verteilung der Fehler ignoriert:

  nur Mehrheitsklasse (d.h. Klasse, mit den meisten Vorkommen im aktuellen Schritt) wird berücksichtigt

  Verteilung der anderen Klassen hat keinen Einfluss

  Gute Attributswahl ist wichtig für tiefere Ebenen des Entscheidungsbaums
- **darum:** auch andere Bewertungsmaße betrachten, hier:

  Informationsgewinn und seine verschiedenen Normierungen, χ^2-**Maß** (sehr bekannt in der Statistik)

Induktion eines Entscheidungsbaums: Notation

S	Menge von Fällen oder Objektbeschreibungen
C	Klassenattribut
$A^{(1)}, \ldots, A^{(m)}$	beschreibende Attribute (ohne Inidices im Folgenden)
$\mathrm{dom}(C)$	$= \{c_1, \ldots, c_{n_C}\}$, n_C: Anzahl der Klassen
$\mathrm{dom}(A)$	$= \{a_1, \ldots, a_{n_A}\}$, n_A: Anzahl der Attribute
$N_{\cdot\cdot}$	Gesamtanzahl der Fälle, d.h. $N_{\cdot\cdot} = \lvert S \rvert$
$N_{i\cdot}$	absolute Häufigkeit der Klasse c_i
$N_{\cdot j}$	absolute Häufigkeit des Attributwerts a_j
N_{ij}	absolute Häufigkeit der Kombination aus c_i und a_j
	wobei $N_{i\cdot} = \sum_{j=1}^{n_A} N_{ij}$ und $N_{\cdot j} = \sum_{i=1}^{n_C} N_{ij}$
$p_{i\cdot}$	relative Häufigkeit der Klasse c_i, $p_{i\cdot} = \frac{N_{i\cdot}}{N_{\cdot\cdot}}$
$p_{\cdot j}$	relative Häufigkeit des Attributwerts a_j, $p_{\cdot j} = \frac{N_{\cdot j}}{N_{\cdot\cdot}}$
p_{ij}	relative Häufigkeit der Kombination aus c_i und a_j, $p_{ij} = \frac{N_{ij}}{N_{\cdot\cdot}}$
$p_{i\lvert j}$	relative Häufigkeit von c_i für Fälle mit a_j, $p_{i\lvert j} = \frac{N_{ij}}{N_{\cdot j}} = \frac{p_{ij}}{p_{\cdot j}}$

Ein informationstheoretisches Bewertungsmaß

Informationsgewinn (Kullback und Leibler 1951, Quinlan 1986)
basiert auf Shannon-Entropie $H = -\sum_{i=1}^{n} p_i \log_2 p_i$ (Shannon 1948)

$$
\begin{aligned}
I_{\text{gain}}(C, A) &= \overbrace{H(C)}^{} - \overbrace{H(C \mid A)}^{} \\
&= \overbrace{-\sum_{i=1}^{n_C} p_{i\cdot} \log_2 p_{i\cdot}}^{} - \overbrace{\sum_{j=1}^{n_A} p_{\cdot j} \left(-\sum_{i=1}^{n_C} p_{i|j} \log_2 p_{i|j} \right)}^{}
\end{aligned}
$$

$H(C)$	Entropie von der Klassenverteilung (C: Klassenattribut)
$H(C \mid A)$	*erwartete Entropie* von der Klassenverteilungfalls der Wert vom Attribut A bekannt ist
$H(C) - H(C \mid A)$	erwartete Entropie der Verminderung oder des *Informationsgewinns*

Induktion des Entscheidungsbaums

Informationsgewinn von Medikament und Geschlecht:

$$H(\text{Medi.}) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(\text{Medi.} \mid \text{Geschl.}) = \frac{1}{2}\underbrace{\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right)}_{H(\text{Medi.}\mid\text{Geschl.}=\text{männlich})} + \frac{1}{2}\underbrace{\left(-\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2}\right)}_{H(\text{Medi.}\mid\text{Geschl.}=\text{weiblich})} = 1$$

$$I_{\text{gain}}(\text{Medi.}, \text{Geschl.}) = 1 - 1 = 0$$

überhaupt kein Informationsgewinn, weil ursprüngliche Gleichverteilung des Medikaments in zwei Gleichverteilungen geteilt wird

Induktion des Entscheidungsbaums

Informationsgewinn von Medikament und Alter:

$$H(\text{Medi.}) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(\text{Medi.} \mid \text{Alt.}) = \frac{1}{2}\underbrace{\left(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}\right)}_{H(\text{Medi.}|\text{Alt.}\leq 40)} + \frac{1}{2}\underbrace{\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right)}_{H(\text{Medi.}|\text{Alt.}>40)} \approx 0.9183$$

$$I_{\text{gain}}(\text{Medi.}, \text{Alt.}) = 1 - 0.9183 = 0.0817$$

Teilung bzgl. Alter kann gesamte Entropie reduzieren

Induktion des Entscheidungsbaums

Informationsgewinn von Medikament und Blutdruck:

$$H(\text{Medi.}) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(\text{Medi.} \mid \text{Blutdr.}) = \frac{1}{4}\cdot 0 + \frac{1}{2}\underbrace{\left(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}\right)}_{H(\text{Medi.}\mid\text{Blutdr.}=\text{normal})} + \frac{1}{4}\cdot 0 = 0.5$$

$$I_{\text{gain}}(\text{Medi.}, \text{Blutdr.}) = 1 - 0.5 = 0.5$$

größter Informationsgewinn, also wird zuerst bzgl. Blutdruck aufgeteilt (genauso wie im Beispiel mit Fehlklassifikationsrate)

Induktion des Entscheidungsbaums

nächte Ebene: Teilbaum „Blutdruck ist normal“

Informationsgewinn für Medikament und Geschlecht:

$$H(\text{Medi.}) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(\text{Medi.} \mid \text{Geschl.}) = \frac{1}{2}\underbrace{\left(-\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3}\right)}_{H(\text{Medi.}|\text{Geschl.}=\text{männlich})} + \frac{1}{2}\underbrace{\left(-\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3}\right)}_{H(\text{Medi.}|\text{Geschl.}=\text{weiblich})} = 0.9183$$

$$I_{\text{gain}}(\text{Medi.}, \text{Geschl.}) = 0.0817$$

Entropie kann reduziert werden

Induktion des Entscheidungsbaums

nächste Ebene: Teilbaum „Blutdruck ist normal“

Informationsgewinn für Medikament und Alter:

$$H(\text{Medi.}) = -\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1$$

$$H(\text{Medi.} \mid \text{Alt.}) = \frac{1}{2}\cdot 0 + \frac{1}{2}\cdot 0 = 0$$

$$I_{\text{gain}}(\text{Medi.}, \text{Alt.}) = 1$$

maximaler Informationsgewinn, d.h. perfekte Klassifikation

ID3: Induktion von Entscheidungsbäumen

ID3 ist mit dieser Heuristik, das Attribut mit dem höchsten Informationsgewinn als „Split-Attribut“ zu verwenden, sehr erfolgreich

Werte mit sehr vielen Attributen durch ID3 bevorzugt: Beispiel: bei einer Einkommensteuererklärung die jedem Bürger zugeordnete eineindeutige Steuernummer.

- Genausoviele Ausprägungen, wie es Bürger (n) gibt
- Partitionierung der Beispielmenge E in n Teilmengen
- bedingte mittlere Information

$$I(E \mid StNr\text{bekannt}) = \sum_{i=1}^{n} \frac{1}{n} H(0;1) = 0bit$$

- Informationsgewinn maximal, allerdings Attribut nutzlos.

C4.5: Induktion von Entscheidungsbäumen

Verbesserung: C4.5

Statt des absoluten Informationsgewinns wird ein normierter Informationsgewinn genutzt.

$$gain\ ratio(a) = \frac{gain(a)}{split\ info(a)}$$

split info(a) ist hierbei die Entropie des Attributes a:

$$split\ info(a) = H(a) = -\sum_{i=1}^{k} P(a = w_i) log_2 P(a = w_i)$$

Beispiel Steuernummer: Induktion einer Gleichverteilung, Normierungsfaktor maximal; nächstes Attribut: dasjenige mit maximalem *gain ratio*

Data Mining und Wissensfindung in Daten

Oberbegriffe für die Automatisierung der Analyse von Daten, Knowledge Discovery in Databases (KDD)

zentrales Forschungsthema in der Künstlichen Intelligenz

KDD: Prozess, neues, nützliches und interessantes Wissen aus Daten herauszufiltern und in verständlicher Form zu präsentieren

KDD-Prozess

Hintergrundwissen und Zielsetzung: Relevantes, bereichsspezifisches Wissen wird zur Verfügung gestellt. Die Ziele des durchzuführenden KDD sollten definiert werden.

Datenauswahl: Eine Menge von Daten wird als Untersuchungsobjekt festgelegt. Darüberhinaus erfolgt gegebenenfalls eine Vorauswahl der betrachteten Variablen.

Datenbereinigung: Ausreißer müssen aus der Datenbasis entfernt, Rauscheffekte herausgefiltert werden. Datentypen werden festgelegt und die Behandlung fehlender Daten wird geklärt.

Datenreduktion und -projektion: Die vorbehandelte Datenmenge wird noch einmal komprimiert durch Reduktion oder Transformation der behandelten Variablen.

KDD-Prozess

Modellfunktionalität: Welchem Zweck dient das Data Mining? U.a. gibt es Klassifikation, Clustering, Regressionsanalyse.

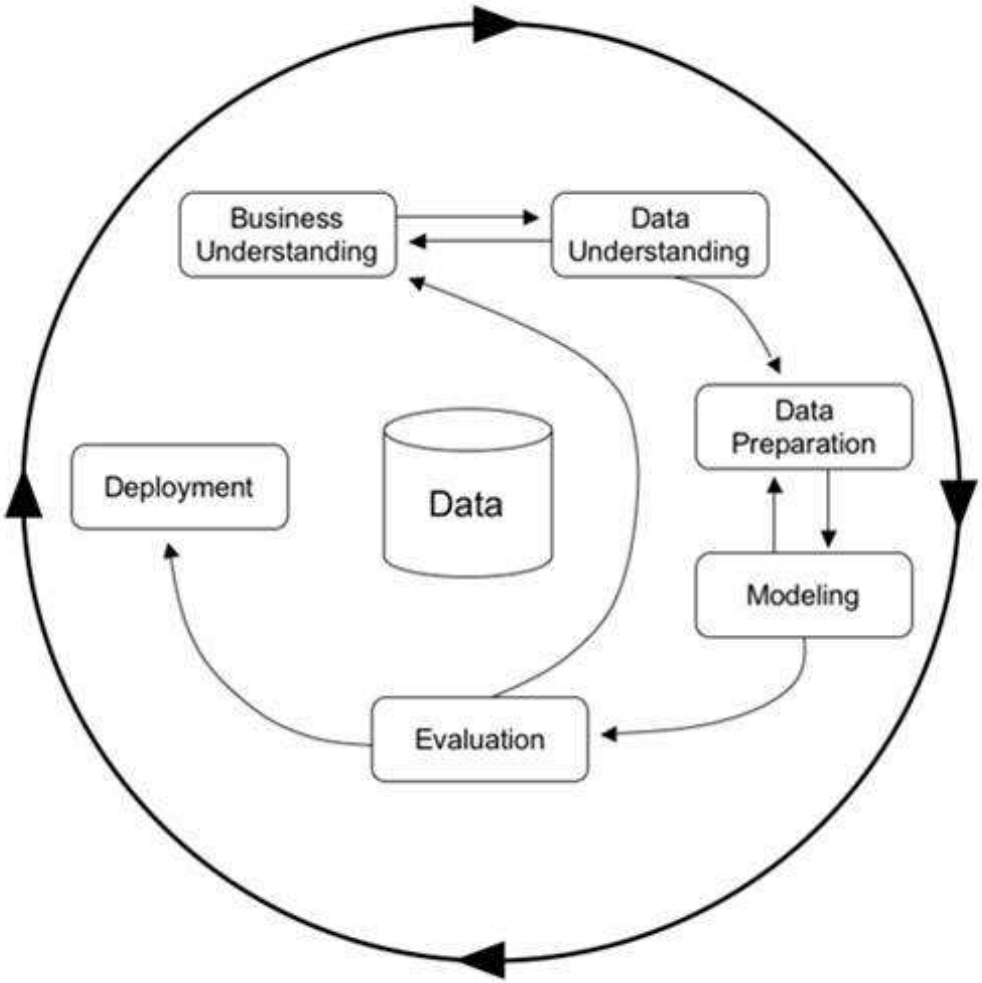
Verfahrenswahl: Bestimmung eines Data-Mining-Verfahrens, das zu den untersuchten Daten und der Zielvorgabe des gesamten KDD-Prozesses passt.

Data Mining: der eigentliche Data-Mining-Prozess, bei dem das ausgewählte Verfahren auf die behandelte Datenmenge angewandt wird, um interessante Informationen z.B. in Form von Klassifikationsregeln oder Clustern zu extrahieren

Interpretation: Die im Data-Mining-Schritt gewonnene Information wird aufbereitet, indem z.B. redundante Information entfernt wird, und schließlich dem Benutzer in verständlicher Form (Visualisierung!) präsentiert.

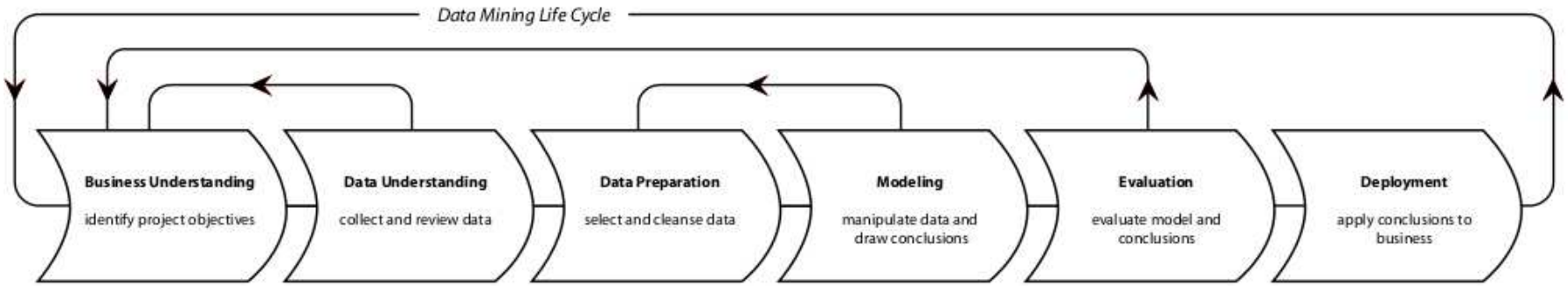
CRross Industry Standard Process for Data Mining

ftp://ftp.software.ibm.com/software/analytics/spss/support/Modeler/Documentation/14/UserManual/CRISP-DM.pdf



CRISP-DM im Detail

http://exde.wordpress.com/2009/03/13/a-visual-guide-to-crisp-dm-methodology/



Determine Business Objectives
Background
Business Objectives
Business Success Criteria
(Log and Report Process)

Assess Situation
Inventory of Resources, Requirements, Assumptions, and Constraints
Risks and Contingencies
Terminology
Costs and Benefits
(Log and Report Process)

Determine Data Mining Goals
Data Mining Goals
Data Mining Success Criteria
(Log and Report Process)

Produce Project Plan
Project Plan
Initial Assessment of Tools and Techniques
(Log and Report Process)

Collect Initial Data
Initial Data Collection Report
(Log and Report Process)

Describe Data
Data Description Report
(Log and Report Process)

Explore Data
Data Exploration Report
(Log and Report Process)

Verify Data Quality
Data Quality Report
(Log and Report Process)

Data Set
Data Set Description
(Log and Report Process)

Select Data
Rationale for Inclusion/ Exclusion
(Log and Report Process)

Clean Data
Data Cleaning Report
(Log and Report Process)

Construct Data
Derived Attributes
Generated Records
(Log and Report Process)

Integrate Data
Merged Data
(Log and Report Process)

Format Data
Reformatted Data
(Log and Report Process)

Select Modeling Technique
Modeling Technique
Modeling Assumptions
(Log and Report Process)

Generate Test Design
Test Design
(Log and Report Process)

Build Model Parameter Settings
Models
Model Description
(Log and Report Process)

Assess Model
Model Assessment
Revised Parameter
(Log and Report Process)

Evaluate Results
Align Assessment of Data Mining Results with Business Success Criteria
(Log and Report Process)

Approved Models
Review Process
Review of Process
(Log and Report Process)

Determine Next Steps
List of Possible Actions
Decision
(Log and Report Process)

Plan Deployment
Deployment Plan
(Log and Report Process)

Plan Monitoring and Maintenance
Monitoring and Maintenance Plan
(Log and Report Process)

Produce Final Report
Final Report
Final Presentation
(Log and Report Process)

Review Project
Experience Documentation
(Log and Report Process)

Data Mining

Einsatzgebiete für Data Mining:

Klassifikation: Ein Objekt wird einer oder mehreren vordefinierten Kategorien zugeordnet

Clustering: Ein Objekt wird einer oder mehreren Klassen bzw. Clustern zugeordnet, wobei diese im Unterschied zur Klassifikation nicht vorgegeben sind, sondern erst bestimmt werden müssen. Natürliche Gruppierungen von Clustern sollen gefunden werden.

Modellierung von Abhängigkeiten: Lokale Abhängigkeiten zwischen Variablen werden etabliert. Die Stärke der Abhängigkeiten wird bei quantitativen Methoden numerisch angegeben.

Data Mining

Einsatzgebiete für Data Mining:

Sequenzanalyse: beschreibt Muster in sequentiellen Daten, um Regelmäßigkeiten und Trends transparent zu machen, beispielsweise in der Zeitreihenanalyse

Assoziationen: sind Zusammenhänge zwischen mehreren Merkmalen und werden meist durch *Assoziationsregeln* repräsentiert.

Im Folgenden werden Assoziationen in Form von Assoziationsregeln eingehender behandelt.

Assoziationsregeln

beschreiben gewisse Zusammenhänge und Regelmäßigkeiten zwischen verschiedenen Dingen wie z.B. den Artikeln eines Warenhauses oder sozio-ökonomischen Merkmalen

Zusammenhänge sind allgemeiner Art, nicht notwendigerweise kausaler Natur

Annahme: in diesen Assoziationen manifestieren sich implizite strukturelle Abhängigkeiten

Beispiel: Warenkorbanalyse

Label	Artikel	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	support
A	Seife	•				•		•		•		0,4
B	Shampoo	•	•	•	•		•		•	•	•	0,8
C	Haarspülung		•	•	•		•		•	•		0,6
D	Duschgel	•			•		•	•		•	•	0,6
E	Zahnpasta	•		•		•		•				0,4
F	Zahnbürste			•		•						0,2
G	Haarfärbung		•		•				•			0,3
H	Haargel		•									0,1
J	Deodorant			•	•	•	•	•	•			0,6
K	Parfüm						•		•			0,2
L	Kosmetikartikel		•		•		•		•		•	0,5

Einkaufstransaktionen in einem Drogeriemarkt

Assoziationsregeln, Formales

behandelte Dinge: *Items*, $\mathcal{I} = \{i_1, i_2, \ldots\}$

$\mathcal{X} \subseteq \mathcal{I}$: Itemmenge

k-Itemmenge: Itemmenge mit k Elementen

Transaktion $t \subseteq \mathcal{I}$ ist eine Itemmenge

$\mathcal{D} = \{t_1, t_2, \ldots\}$ Menge von Transaktionen als Datenbasis

Relativer Anteil aller Transaktionen, die X enthalten:

$$support(X) = \frac{|\{t \in \mathcal{D} \mid X \subseteq t\}|}{|\mathcal{D}|}$$

Assoziationsregeln, Formales

Assoziationsregel: $X \rightarrow Y$

- $X, Y \subseteq \mathcal{I}$
- $X \cap Y = \emptyset$

$support(X \rightarrow Y) = support(X \cup Y)$

Relativer Anteil derjenigen X enthaltenden Transaktionen, die auch Y enthalten:

$$confidence(X \rightarrow Y) = \frac{|\{t \in \mathcal{D} \mid (X \cup Y) \subseteq t\}|}{|\{t \in \mathcal{D} \mid X \subseteq t\}|} \quad (1)$$

$$= \frac{support(X \rightarrow Y)}{support(X)} \quad (2)$$

Assoziationsregeln, Algorithmus

Aufgabe: Finde alle Assoziationsregeln, die in der betrachteten Datenbasis mit einem Support von mindestens *minsupp* und einer Konfidenz von mindestens *minconf* gelten, wobei *minsupp* und *minconf* benutzerdefinierte Werte sind.

Teilaufgabe 1: Finde alle Itemmengen, deren Support über der *minsupp*-Schwelle liegt. Diese Mengen werden *häufige Itemmengen* (frequent itemsets) genannt.

Teilaufgabe 2: Finde in jeder häufigen Itemmenge I alle Assoziationsregeln $I' \rightarrow (I - I')$ mit $I' \subset I$, deren Konfidenz mindestens *minconf* beträgt.

Nützliche Tatsache für den folgenden *apriori*-Algorithmus: Alle Teilmengen einer häufigen Itemmenge sind ebenfalls häufig. Alle Obermengen einer nicht häufigen Itemmenge sind ebenfalls nicht häufig.

Apriori-Algorithmus

Algorithmus: Apriori
Eingabe: Datenbasis $\mathcal{D}$
Ausgabe: Menge häufiger Itemmengen

$L_1 := \{\text{häufige 1-Itemmengen}\}$
$k := 2$
while $L_{k-1} \neq \emptyset$ **do**
    $C_k := \text{AprioriGen}(L_{k-1})$
    for all Transaktionen $t \in \mathcal{D}$ **do**
        $C_t := \{c \in C_K \mid c \subseteq t\}$
        for all Kandidaten $c \in C_t$ **do**
            c.count := c.count + 1
        end for
    end for
    $L_k := \{c \in C_k \mid c.count \geq |\mathcal{D}| \cdot minsupp\}$
    $k := k + 1$
end while
return $\bigcup_k L_k$

Apriori-Algorithmus

Algorithmus: AprioriGen(L_{k-1})
Eingabe: Menge häufiger (k-1)-Itemmengen L_{k-1}

Ausgabe: Obermenge der Menge häufiger k-Itemmengen

$C_k := \emptyset$
for all $p, q \in L_{k-1}$ mit $p \neq q$ **do**
    if p und q haben $k-2$ gleiche Elemente
        $p = \{e_1, \ldots, e_{k-2}, e_p\}$
        $q = \{e_1, \ldots, e_{k-2}, e_q\}$
        und $e_p < e_q$ **then**
        $C_k := C_K \cup \{\{e_1, \ldots, e_{k-2}, e_p, e_q\}\}$
    end if
end for
for all $c \in C_k$ **do**
    for all $(k-1)$-Teilmengen s von c **do**
        if $s \notin L_{k-1}$ **then**
            $C_k := C_K \setminus \{c\}$
        end if
    end for
end for
return C_k

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

a: 7	b: 3	c: 7	d: 6	e: 7

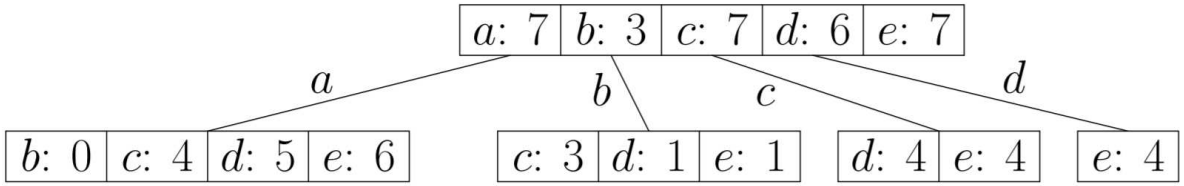
Beispiel Transaktionsdatenbank mit 5 Items und 10 Transaktionen.

Minimum support: 30%, d.h., mindestens 3 Transaktionen müssen das Itemset enthalten.

Alle einelementigen Itemsets sind häufig $\rightarrow$ komplette zweite Ebene wird benötigt.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

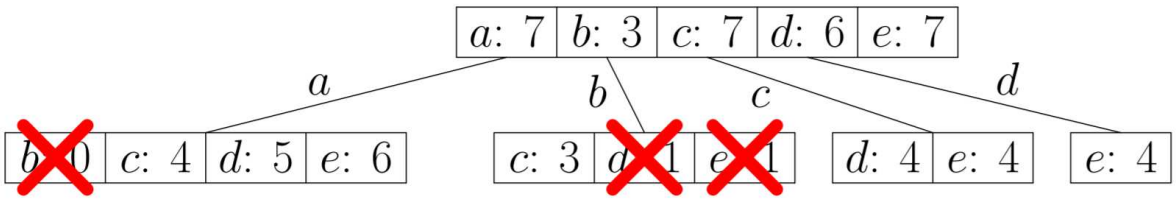


Berechnung des Supports: Für jedes Itemset durchlaufe die Datenbank und zähle die Transaktionen, die das Itemset enthalten (hochgradig ineffizient).

Besser: Traversiere den Baum für jede Transaktion und finde die Itemsets die sie enthält. Dies kann effizient durch eine doppelt rekursive Funktion implementiert werden.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



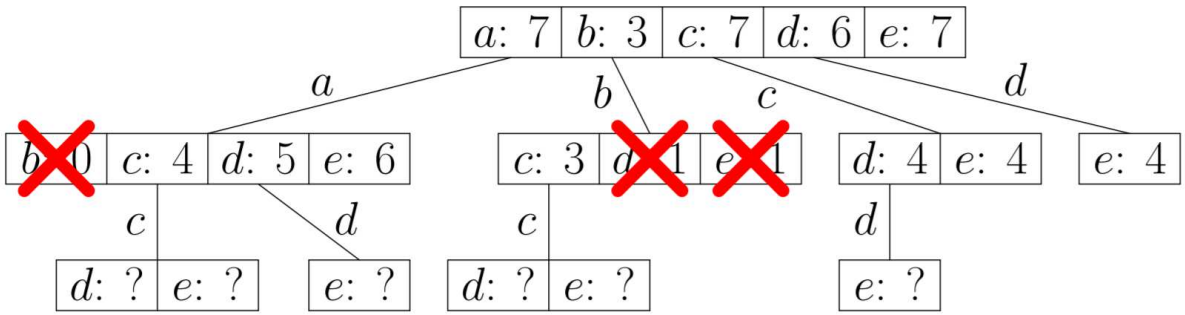
Minimum support: 30%, d.h., mindestens 3 Transaktionen müssen das Itemset enthalten

Nicht-häufige Itemsets: $\{a, b\}$, $\{b, d\}$, $\{b, e\}$.

Die Teilbäme, die an diesen Itemsets starten, müssen nicht mehr durchlaufen werden.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



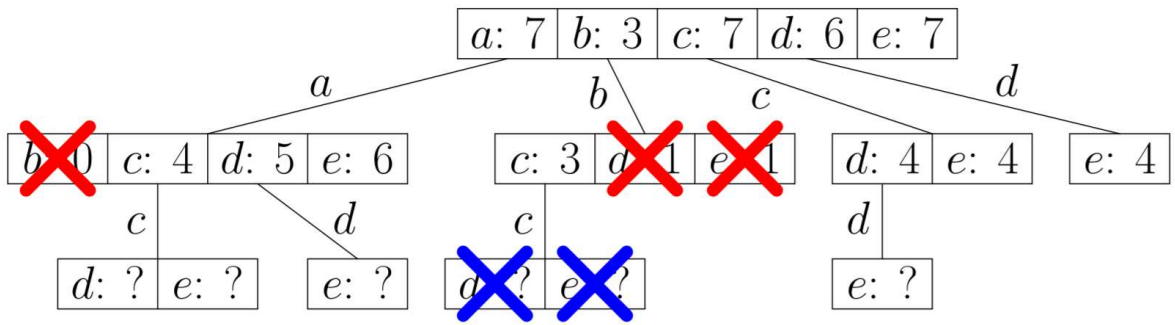
Generiere Kandidatenitemsets mit 3 Items (Eltern müssen häufig sein).

VOR dem Auszählen: Prüfe ob die Kandidaten ein nicht-häufiges Itemset beinhalten.

- Ein Itemset mit k Items hat k Teilmengen mit jeweils $k-1$ Items.
- Der Elter ist nur eine dieser Mengen.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

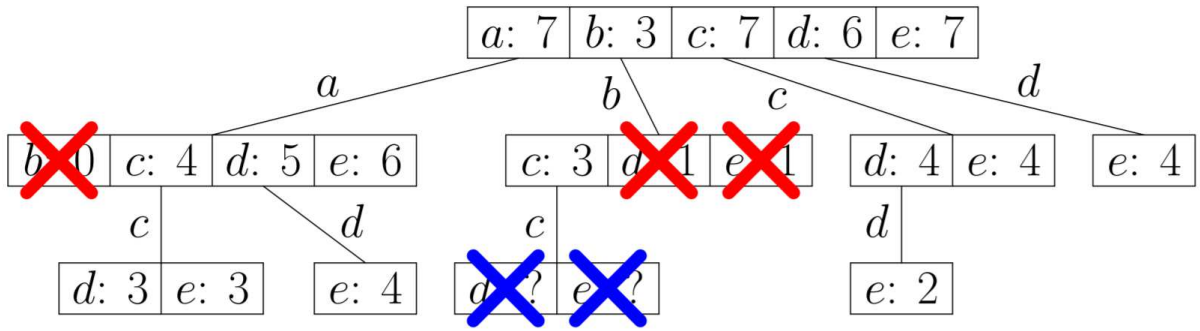


Die Itemsets $\{b, c, d\}$ und $\{b, c, e\}$ können verworfen werden, da

- $\{b, c, d\}$ die nicht-häufige Itemmenge $\{b, d\}$ enthält, und
- $\{b, c, e\}$ die nicht-häufige Itemmenge $\{b, e\}$ enthält.

Apriori: Ebenenweise Suche

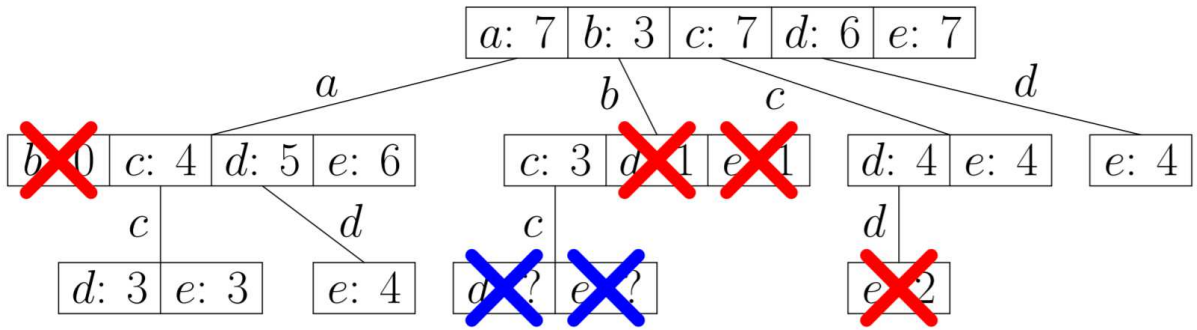
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



Nur die verbleibenden vier Itemsets der Größe 3 werden ausgewertet.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

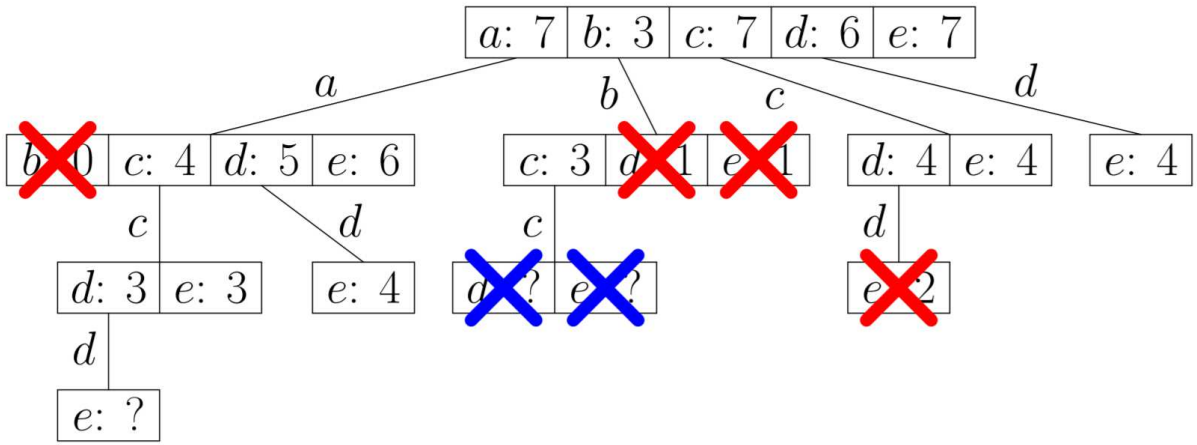


Minimum support: 30%, d.h., mindestens 3 Transaktionen müssen das Itemset enthalten

Nicht-häufiges Itemset: $\{c, d, e\}$.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

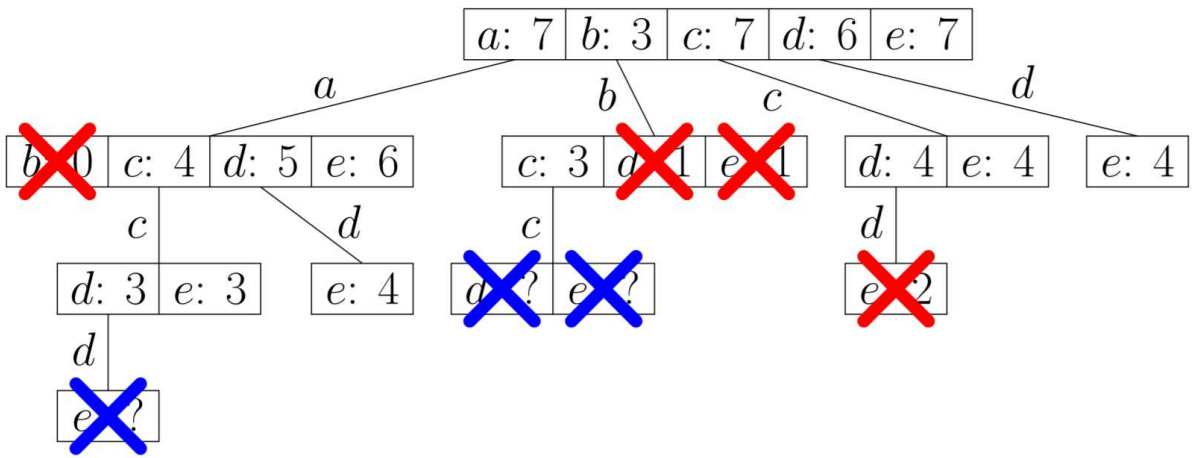


Generiere Kandidatenitemsets mit 4 Items (Eltern müssen häufig sein).

VOR dem Auszählen: Prüfe ob die Kandidaten ein nicht-häufiges Itemset beinhalten.

Apriori: Ebenenweise Suche

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



Das Itemset $\{a, c, d, e\}$ kann verworfen werden,
da es das nicht-häufige Itemset $\{c, d, e\}$ enthält.

Konsequenz: Keine Kandidatenitemsets mit 4 Items.

Vierter Zugriff auf die Transaktionsdatenbank nicht notwendig.

Beispiel: Warenkorbanalyse

ideales Einsatzszenario für Assoziationsregeln

- Modellbildung ist nicht nötig
- Regeln können isoliert betrachtet werden
- Daten stehen in der Regel bereits zur Verfügung

Beispiel: Warenkorbanalyse

Label	Artikel	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	support
A	Seife	•				•		•		•		0,4
B	Shampoo	•	•	•	•		•		•	•	•	0,8
C	Haarspülung		•	•	•		•		•	•		0,6
D	Duschgel	•			•		•	•		•	•	0,6
E	Zahnpasta	•		•		•		•				0,4
F	Zahnbürste			•		•						0,2
G	Haarfärbung		•		•				•			0,3
H	Haargel		•									0,1
J	Deodorant			•	•	•	•	•	•			0,6
K	Parfüm						•		•			0,2
L	Kosmetikartikel		•		•		•		•		•	0,5

Einkaufstransaktionen in einem Drogeriemarkt

Beispiel: Warenkorbanalyse

gesucht: alle Assoziationsregeln mit:

- minsupp $= 0{,}4$
- minconf $= 0{,}7$

in realen Anwendung wird *minsupp* in der Regel sehr viel kleiner gewählt ($< 0,01$)

häufige 1-Itemmengen:

$$L_1 = \{\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{J\}, \{L\}\}$$

Beispiel: Warenkorbanalyse

Berechnung der Menge C_2: alle paarweisen Kombinationen von Mengen in L_1 bilden und deren Support bestimmen.

C_2-Menge	Support	C_2-Menge	Support	C_2-Menge	Support
{A,B}	0,2	{B,D}	0,5	{C,L}	0,4
{A,C}	0,1	{B,E}	0,2	{D,E}	0,2
{A,D}	0,2	{B,J}	0,4	{D,J}	0,3
{A,E}	0,3	{B,L}	0,5	{D,L}	0,3
{A,J}	0,2	{C,D}	0,3	{E,J}	0,3
{A,L}	0,0	{C,E}	0,1	{E,L}	0,0
{B,C}	0,6	{C,J}	0,4	{J,L}	0,3

Beispiel: Warenkorbanalyse

häufigste 2-Itemmengen:

$$L_2 = \{\{B, C\}, \{B, D\}, \{B, J\}, \{B, L\}, \{C, J\}, \{C, L\}\}$$

Berechnung von C_3:

C_3 vor Teilmengencheck	C_3 nach Teilmengencheck	Support
{B,C,D}	{B,C,J}	0,4
{B,C,J}	{B,C,L}	0,4
{B,C,L}		
{B,D,J}		
{B,D,L}		
{B,J,L}		
{C,J,L}		

Beispiel: Warenkorbanalyse

Damit ist

$$L_3 = \{\{B, C, J\}, \{B, C, L\}\}$$

einzig mögliche weitere Kombination: $\{B, C, J, L\}$

allerdings nicht häufig, daher ist $C_4 = L_4 = \emptyset$

Beispiel: Warenkorbanalyse

Bildung der Assoziationsregeln aus den häufigen Itemmengen:

Regel	Konfidenz	Regel	Konfidenz
B→C	0,75	C→B	1,00
B→D	0,63	D→B	0,83
B→J	0,50	J→B	0,67
B→L	0,63	L→B	1,00
C→J	0,67	J→C	0,67
C→L	0,67	L→C	0,80

fünf der Regeln erfüllen die Konfidenzbedingung (minconf$= 0,7$)

Beispiel: Warenkorbanalyse

L_3 enthält $l_{3.1} = \{B, C, J\}$ und $l_{3.2} = \{B, C, L\}$

$l_{3.1}$ (in [] die Konfidenz der Regel)

- $H_1 = \{B, C, J\}$
- Regeln: BC→J [0,67], BJ→C [1,00], CJ→B[1,00]
- $H_2 = \text{AprioriGen}(H_1) = \{B, C\}$
- Regel: J→BC [0,67]

$l_{3.2}$

- Regeln: BC→L [0,67], BL→C [0,8], CL→B [1,00]
- durch Erweiterung der Konklusion noch: L→BC [0,8]

Beispiel: Warenkorbanalyse

Regel			Support	Konfidenz
Shampoo	$\rightarrow$	Haarspülung	0,6	0,75
Haarspülung	$\rightarrow$	Shampoo	0,6	1,00
Duschgel	$\rightarrow$	Shampoo	0,5	0,83
Kosmetik	$\rightarrow$	Shampoo	0,5	1,00
Kosmetik	$\rightarrow$	Haarspülung	0,4	0,80
Shampoo, Deodorant	$\rightarrow$	Haarspülung	0,4	1,00
Haarspülung, Deodorant	$\rightarrow$	Shampoo	0,4	1,00
Shampoo, Kosmetik	$\rightarrow$	Haarspülung	0,4	0,80
Haarspülung, Kosmetik	$\rightarrow$	Shampoo	0,4	1,00
Kosmetik	$\rightarrow$	Shampoo, Haarspülung	0,4	0,80

FPM – Frequent Pattern Mining

Werbung in eigener Sache: während der prüfungsfreien Zeit wird die Blockveranstaltung **„Frequent Pattern Mining“** stattfinden, die von PD Christian Borgelt gehalten wird. In dieser Veranstaltung geht es um das Finden häufiger Muster verschiedenster Formen in Daten, u.a. Assoziationsregeln. Verschiedene Algorithmen zum Thema werden vorgestellt und eingehend behandelt.

Weitere Informationen sind verfügbar unter:

`http://www.borgelt.net/teach/fpm`

Übersicht

Probleme bei großen Neuronalen Netzen

Probleme:

Gewichtsänderung nimmt in vorderen Schichten exponentiell ab

Lernen dauert zu lang

Zuwenige (gelabelte) Lernbeispiele vorhanden

Konvergenz zu lokalen Minima

Lösungsansätze:

Initialisiere Gewichte nicht zufällig, sondern abhängig vom Datensatz

Verwende schnellere Computer
Lastverteilung auf GPUs

Sammle mehr gelabelte Lernbeispiele

Kann nicht vollständig verhindert werden

Rectified Linear Unit (ReLU)

Wähle statt Neuron
Rectified Linear Unit

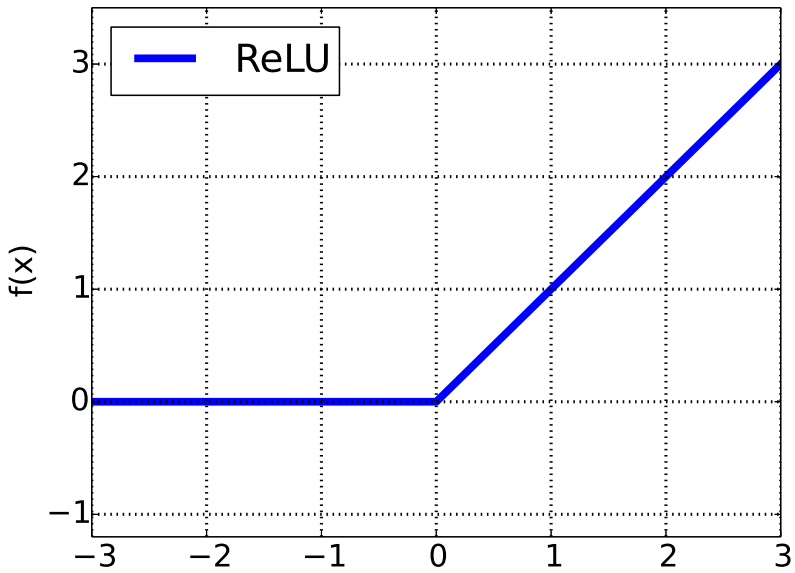
ReLU: $f(x) = max(0, x)$

Vorteile:

- sehr einfache Berechnung
- Ableitung ist leicht zu bilden
- 0-Werte vereinfachen Lernen

Nachteile:

- kein Lernen links der 0
- mathematisch eher unschön
- Nicht-differenzierbarer „Knick“ bei 0



[ReLu nach Glorot et. al 2011]

ReLU: Berechnungsvarianten

Softplus:

$f(x) = ln(1 + e^x)$

- „Knick“ wurde beseitigt
- Einige Vorteile auch

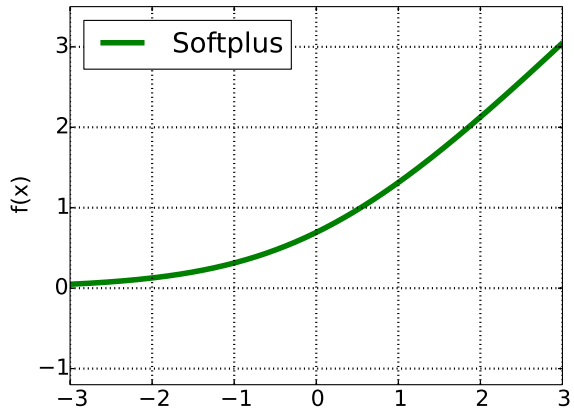
Noisy ReLU:

$f(x) = max(0, x + \mathcal{N}(0, \sigma(x)))$

- Addiert Gaussches Rauschen

Leaky ReLU

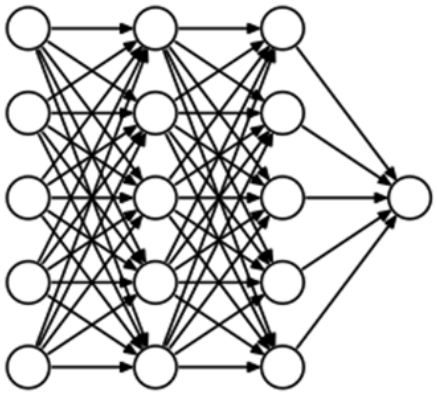
$$f(x) = \begin{cases} x, & \text{falls } x > 0, \\ 0.01x, & \text{sonst.} \end{cases}$$

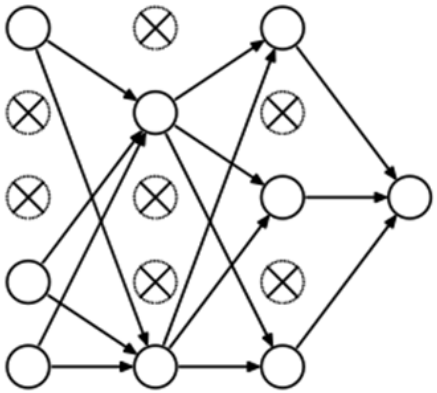


[Softplus nach Glorot et. al 2011]

Dropout

ohne Dropout





mit Dropout

Gewünschte Eigenschaft:
Robustheit bei Ausfall von Neuronen

Ansatz beim Lernen:

- Nutze nur 50% der Neuronen
- Wähle diese zufällig

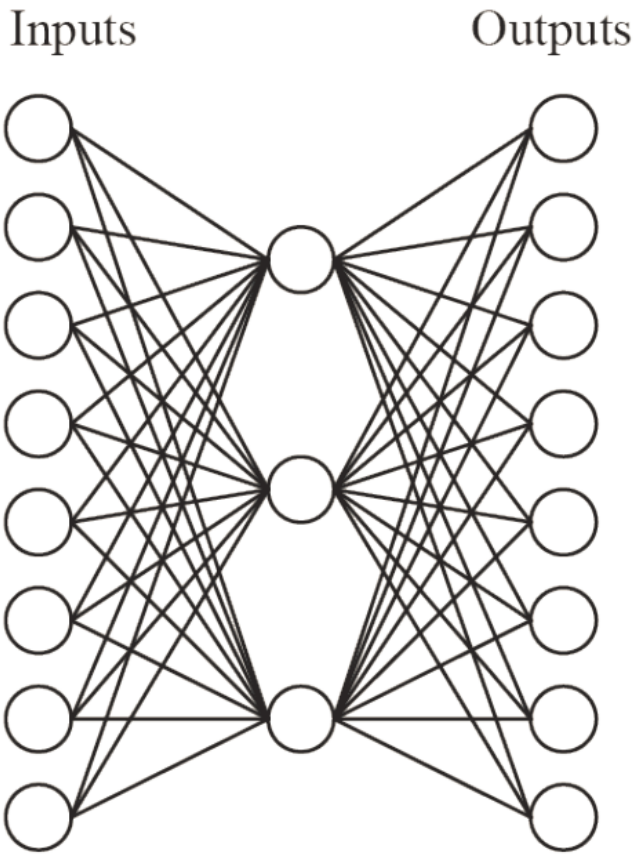
Ansatz beim Anwenden

- Nutze 100% der Neuronen
- Halbiere alle Gewichte

Ergebnis:

- Robustere Repräsentation
- Verbesserte Generalisierung

Autoencoder



Erstellt eine Kodierung der Daten

Lernt Gewichte mit Rückpropagation

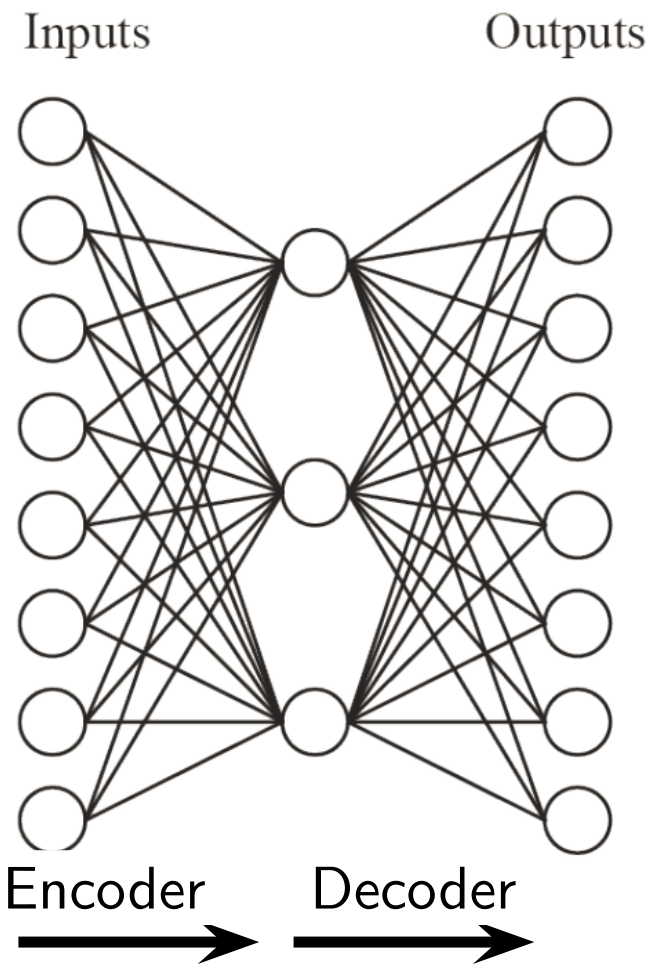
Durch **unüberwachtes** Lernen

Fehler ist $|out - in|^2$

Inputs		Versteckte Gewichte		Outputs
10000000	→	?	→	10000000
01000000	→		→	01000000
00100000	→		→	00100000
00010000	→		→	00010000
00001000	→		→	00001000
00000100	→		→	00000100
00000010	→		→	00000010
00000001	→		→	00000001

[Grafiken nach T. Mitchell, Machine Learning, McGraw Hill, 1997]

Autoencoder



Nutze für Dekodierung die transponierte Gewichtsmatrix der Encodierung

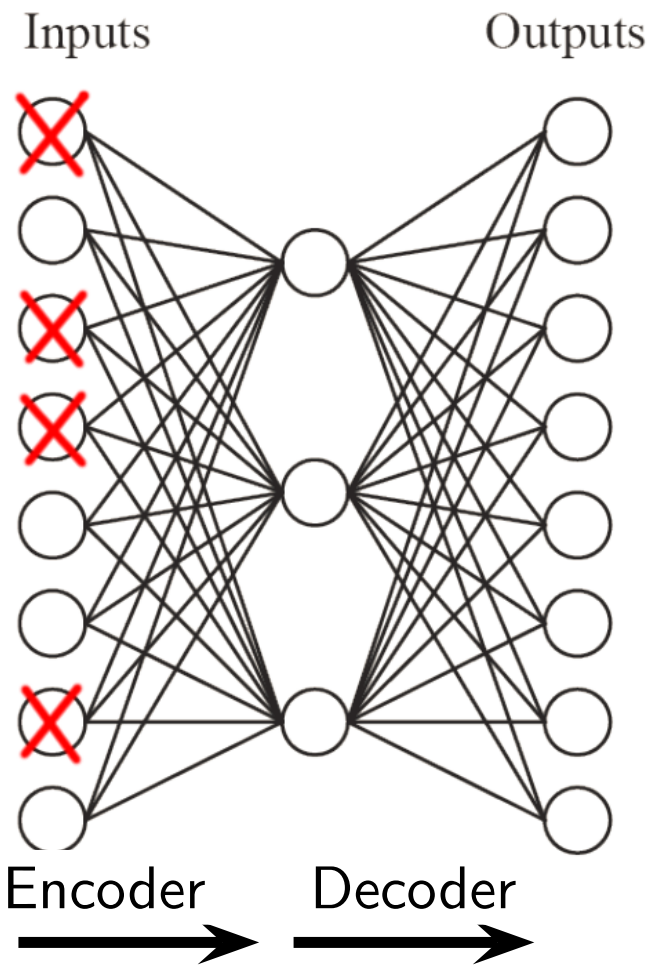
Ergebnis nach 5000 Iterationen:

Binäre Kodierung fast erreicht

Inputs		Versteckte Gewichte				Outputs
10000000	→	.89	.04	.08	→	10000000
01000000	→	.15	.99	.99	→	01000000
00100000	→	.01	.97	.27.	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.01	.11	.88	→	00000100
00000010	→	.80	.01.	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

[Grafiken nach T. Mitchell, Machine Learning, McGraw Hill, 1997]

Rauschreduzierender (Denoising) Autoencoder



Gegeben:
eine dünne (sparse) Repräsentation

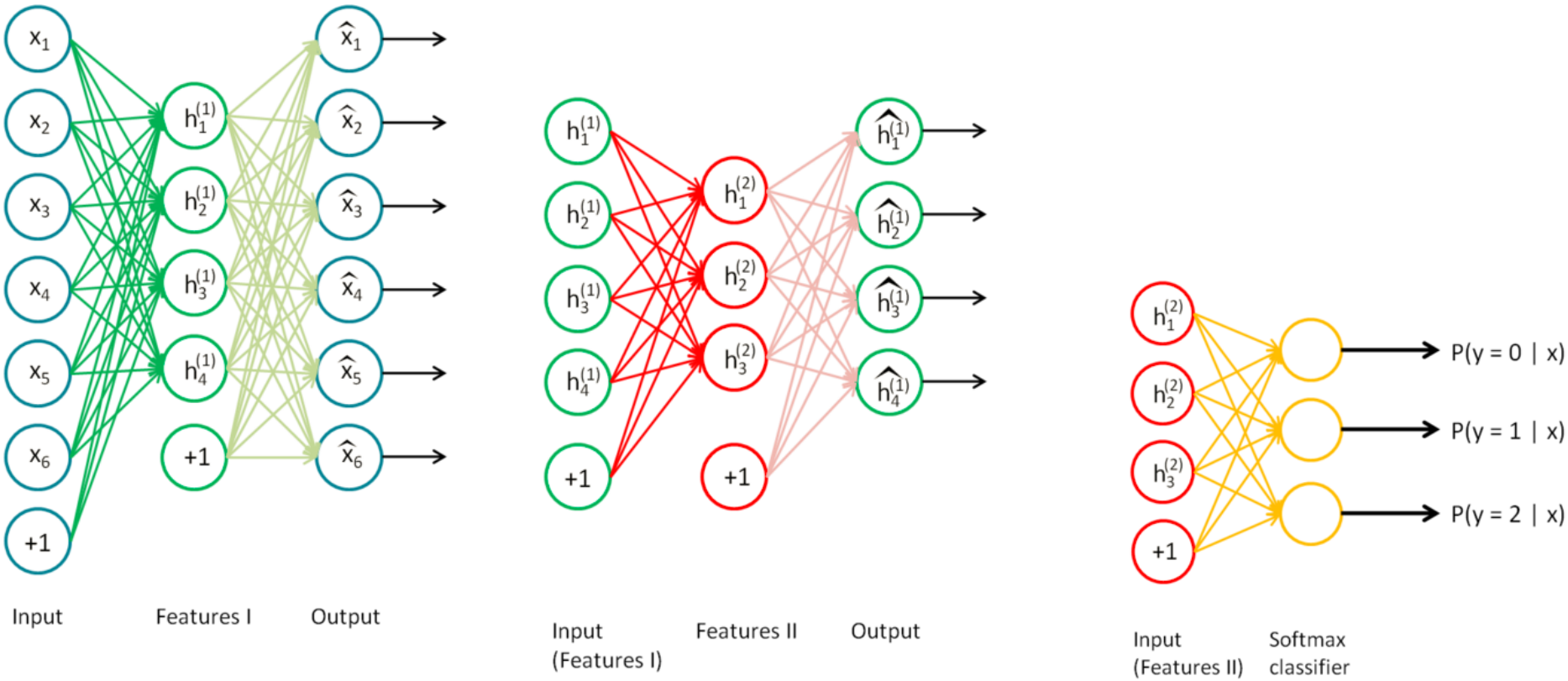
Gewünscht:
eine volle Repräsentation

Ansatz:
Kombiniere Autoencoder mit Dropout

Ergebnis:
komprimierte Darstellung
dynamisch auf Lernbeispiele zugeschnitten
Features für andere Algorithmen

Stapeln von Autoencodern

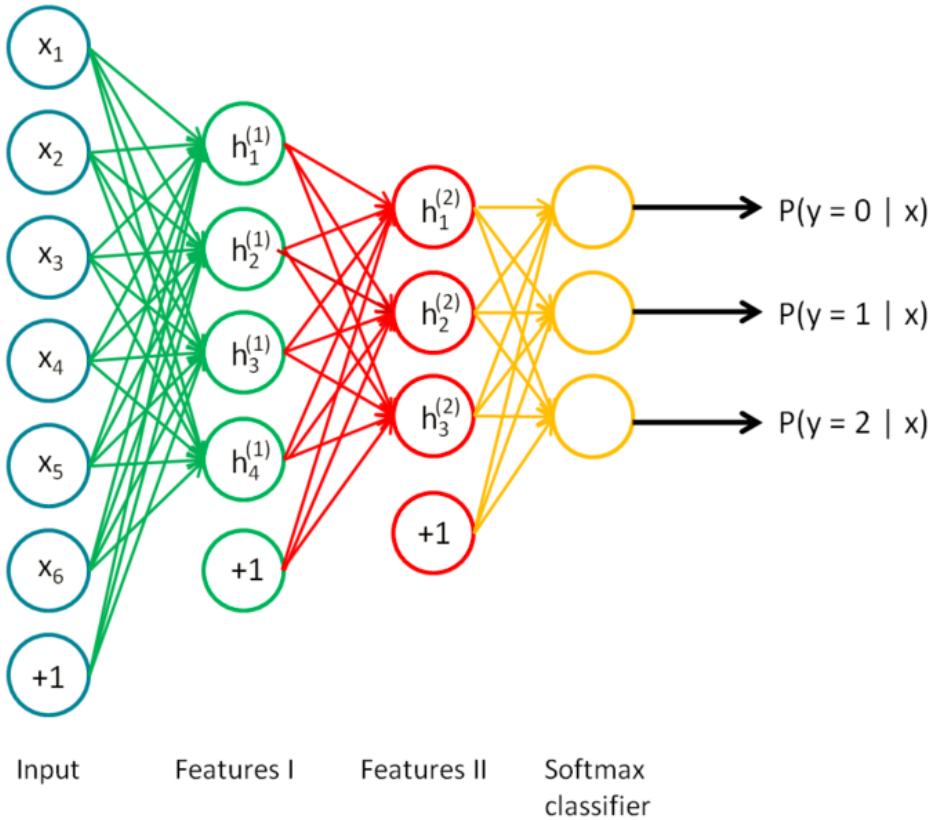
Staple Autoencoder, um die besten Features zu erhalten



[http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders]

Stapeln von Autoencodern

Nutze die (vor)gelernten Features zur Klassifikation



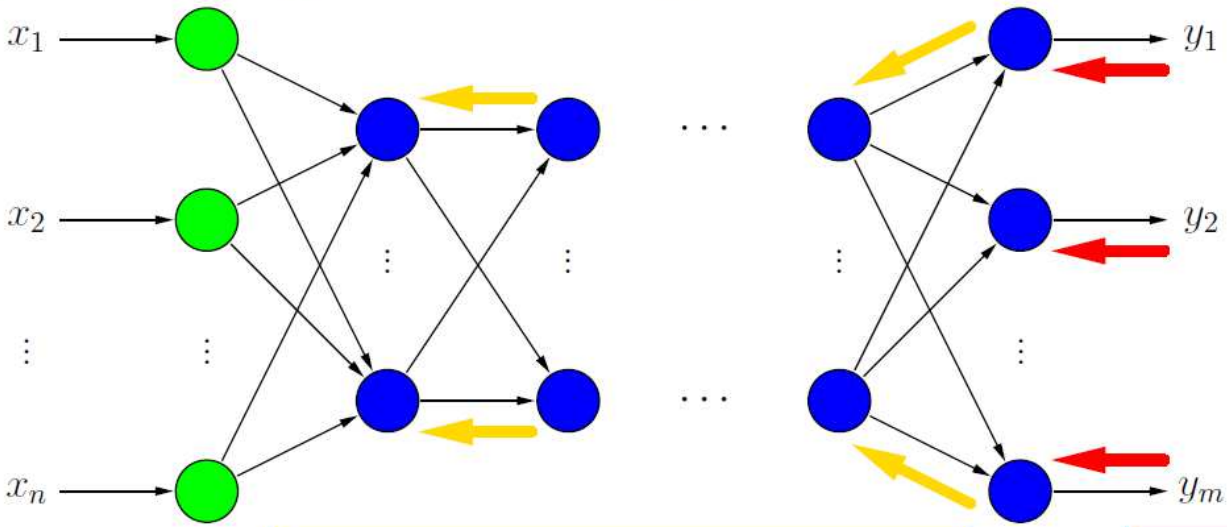
[http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders]

Wiederholung: Fehlerrückpropagation

$$\forall u \in U_{\text{in}}: \quad \text{out}_u^{(l)} = \text{ex}_u^{(l)}$$

Vorwärts-propagation:

$$\forall u \in U_{\text{hidden}} \cup U_{\text{out}}: \quad \text{out}_u^{(l)} = \left(1 + \exp\left(-\sum_{p \in \text{pred}(u)} w_{up}\, \text{out}_p^{(l)}\right)\right)^{-1}$$



logistische Aktivierungs-funktion

impliziter Biaswert

Fehlerfaktor:

Rückwärts-propagation:

$$\forall u \in U_{\text{hidden}}: \quad \delta_u^{(l)} = \left(\sum_{s \in \text{succ}(u)} \delta_s^{(l)} w_{su}\right) \lambda_u^{(l)}$$

$$\forall u \in U_{\text{out}}: \quad \delta_u^{(l)} = \left(o_u^{(l)} - \text{out}_u^{(l)}\right) \lambda_u^{(l)}$$

Aktivierungs-ableitung:

$$\lambda_u^{(l)} = \text{out}_u^{(l)} \left(1 - \text{out}_u^{(l)}\right)$$

Gewichts-änderung:

$$\Delta w_{up}^{(l)} = \eta\, \delta_u^{(l)}\, \text{out}_p^{(l)}$$

Hybrider Deep Learning Algorithmus

Definiere für die Lernaufgabe geeignete Netzstruktur

Erstelle entsprechend der Struktur Autoencoder und lasse sie mit Rückpropagation einzeln lernen

Verwende nur die Encoder, ihre Gewichte und eine weitere vollständig vernetzte, zufällig initialisierte Schicht zur Klassifikation

Lasse das so vortrainierte Netz mit Rückpropagation lernen

Problem: Objekterkennung in Bildern

Imagenet Large Scale Visual Recognition Challenge (LSVRC) seit 2010

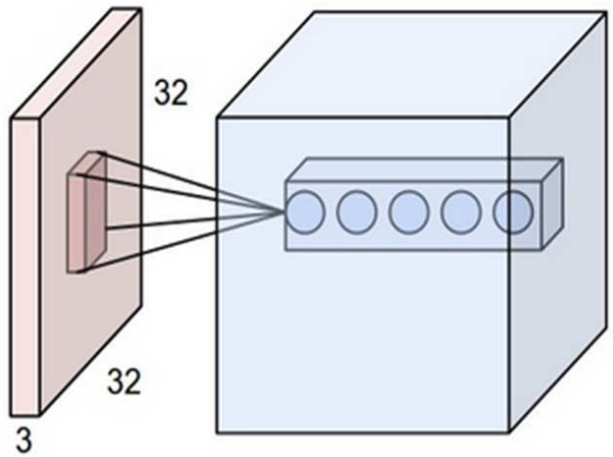
Finde 200 Objektklassen (Stuhl, Tisch, Person, Fahrrad,...)

in Bildern mit ca. 500 x 400 Pixeln, 3 Farbkanälen

Neuronales Netz mit ca. 600.000 Neuronen in der ersten Schicht

200 Neuronen in der Ausgabeschicht

Faltung (Convolution)



Motivation: Egal wo auf dem Bild ein Objekt ist, soll es erkannt werden

Idee: Verwende die selben Features auf dem gesamten Bild

Umsetzung: Filter / Kernel werden auf jedem Teil des Bildes angewandt und teilen sich die Gewichte

Parameter:
Anzahl der Filter
Stärke der Überlappung

Faltung (Convolution)

Image

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Filter

1	0	1
0	1	0
1	0	1

Convolved Feature

4	3	4
2	4	3
2	3	4

[http://ufldl.stanford.edu/wiki/index.php/Feature_extraction_using_convolution]

Featuretransformation

Schiebe einen „Filter“ über die Features und betrachte die „gefilterten“ Features

Multipliziere Originalfeature mit Filter und Summiere

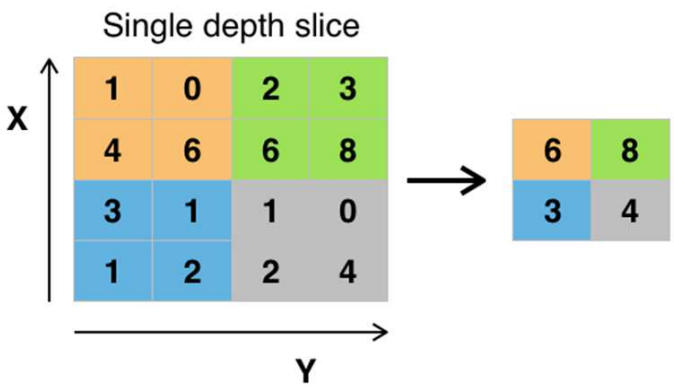
Originalraum: 5x5

Filtergröße: 3x3

Neue Featuregröße: 3x3

Featureraum wird kleiner

Pooling



Featuretransformation

Schiebe einen „Filter“ über die Features und betrachte die „gefilterten“ Features

Betrachte den Bereich entsprechend der Filtergröße

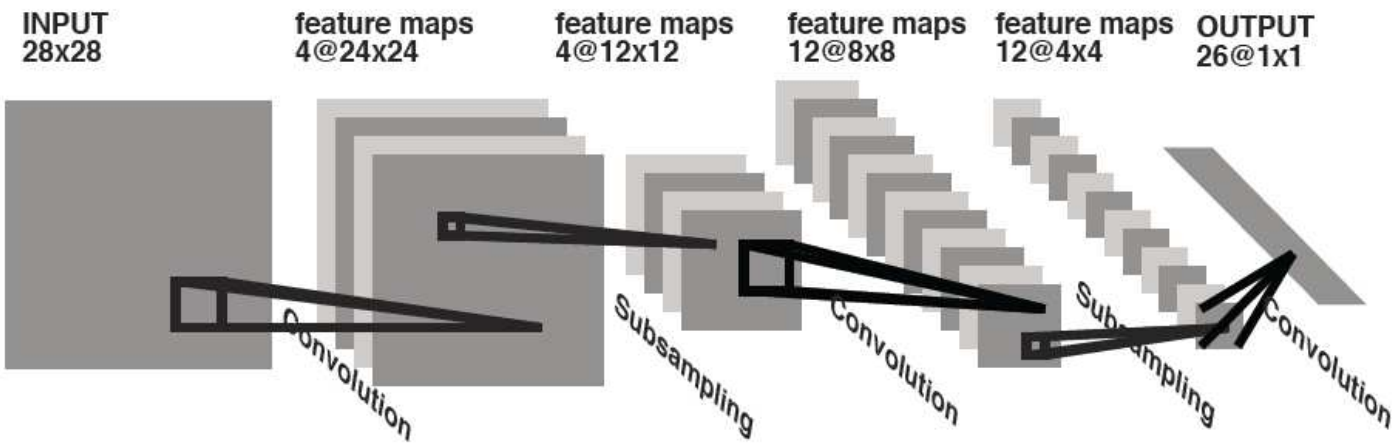
Max Pooling: Nimm maximalen Wert

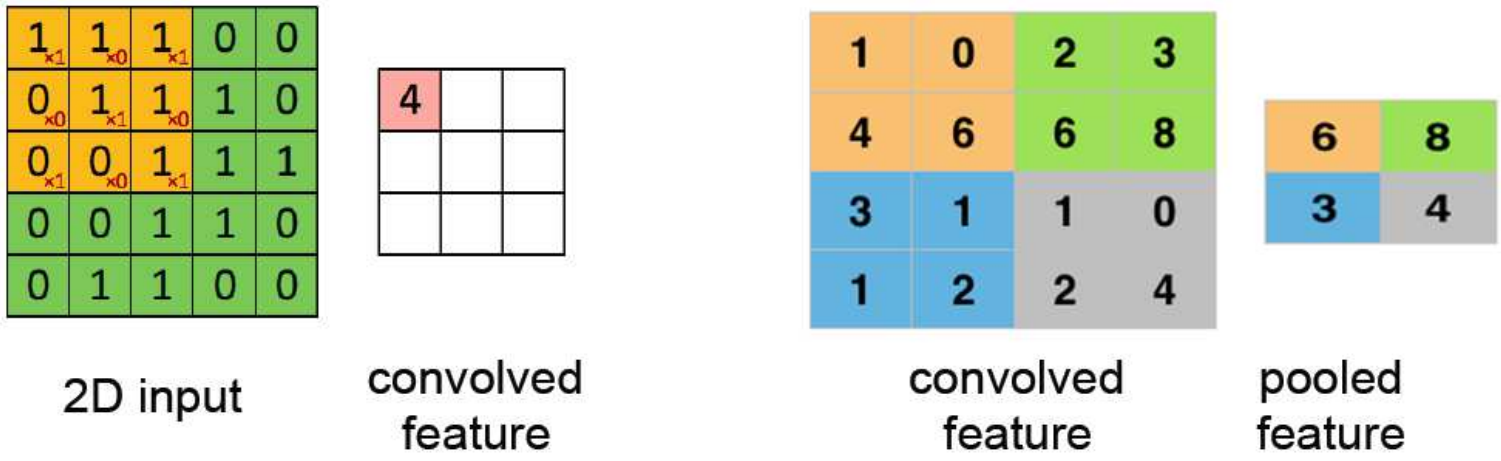
Mean Pooling: Nimm Mittelwert

Featureraum wird kleiner

Faltendende (Convolutional) Neuronale Netze

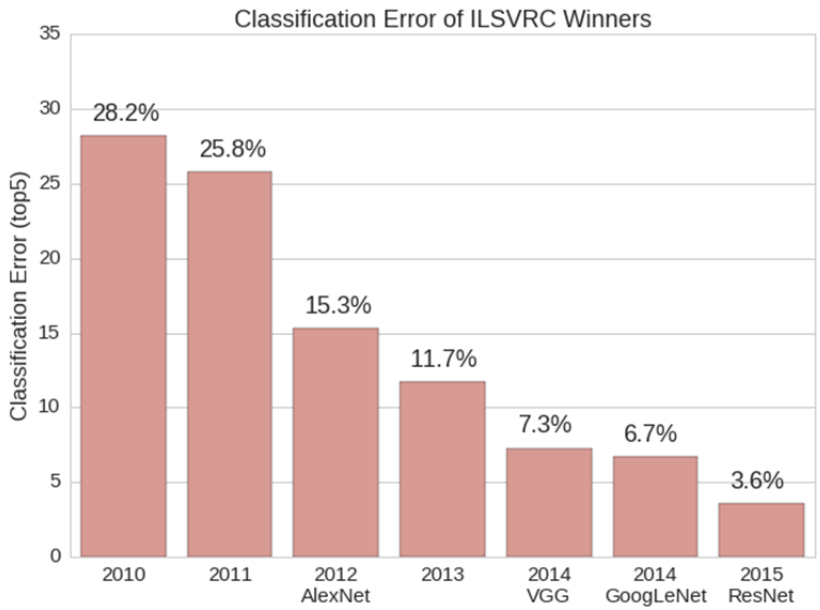
[Y. Bengio and Y. Lecun, 1995]





[http://ufldl.stanford.edu/wiki/index.php/Feature_extraction_using_convolution]

Resultate im Bereich Bildklassifizierung



Grafik: William Beluch, ImageNet Classification with Deep Convolutional Neural Networks

Noch vor 10 Jahren: unmöglich

Rasante Entwicklung in den letzten Jahren

Oft verwendet: Ensembles von Netzen

Netze werden tiefer: ResNet (2015) mehr als 150 Schichten

Anwendung: IQ-Test

Lösen von verbalen Verständnisfragen in IQ-Tests

Verbale IQ-Tests beinhalten hier 5 Arten von Fragen:
Analogie 1, Analogie 2, Klassifikation, Synonym, Antonym

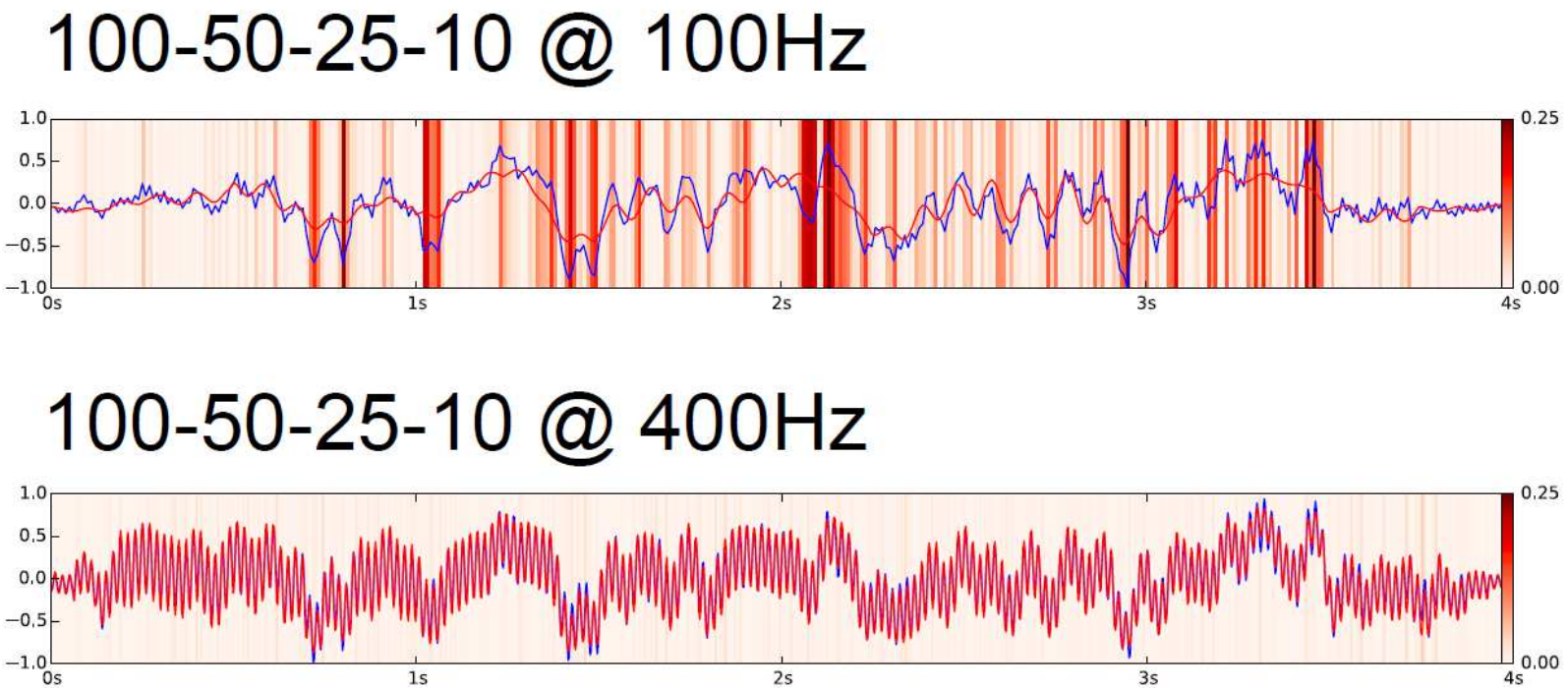
Beispielfrage(Analogie 1): Isotherm verhält sich zu Temperatur wie isobar zu?
(i) Atmosphäre, (ii) Wind, (iii) Druck, (iv) Ausdehnung, (v) Strömung

Ansatz:

- Klassifiziere den Fragentyp mit Hilfe einer SVM
- Benutze für jeden Fragentyp einen dafür erstelltes Tiefes Neuronales Netz
- Nutze zum Lernen von zusammengehörenden Wörter eine große Datenbasis (Wiki2014)

Ergebnis: DeepLearning etwas besser als Bachelor-Absolventen

Rhythmus-Rekonstruktion durch EEG-Analyse



[Quelle: Sebastian Stober, DKE-Kolloquium 03.06.2014]

German Traffic Sign Recognition Benchmark

Wurde analysiert bei der International Joint Conference on Neural Networks (IJCNN) 2011

Problemstellung:

Ein Bild, mehrere Klassen Klassifikationsproblem

Mehr als 40 Klassen

Mehr als 50.000 Bilder

Ergebnis:

Erste übermenschliche visuelle Mustererkennung

Fehlerraten:

Mensch: 1.16%, NN:0.56%

Stallkamp et al. 2012

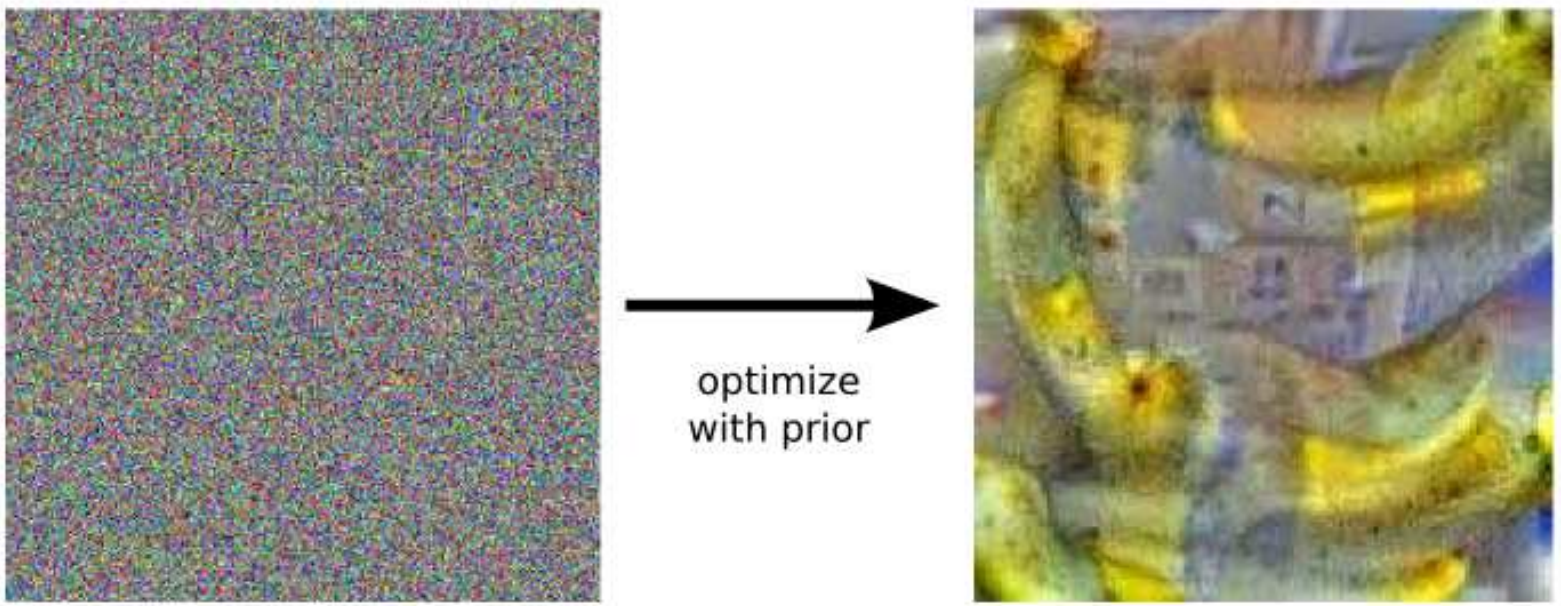
Verwendetes Netz:
Input, Conv., Max., Conv., Max., Conv., Max, Full, Full

Details zu den Gewinnern

Visualisierung von gelernten Neuronalen Netzen

Neuronale Netze zur Objekterkennung in Bildern

Was erkennt ein Neuronales Netz in Rauschen, wenn es Bananen gelernt hat?



Mehr Beispiele

Quelle: Heise: Wovon träumen neuronale Netze?

Deep Learning Libraries

Theano http://deeplearning.net/software/theano/
Python Implementierung für GPU-Verarbeitung von math. Ausdrücken

Tensorflow https://www.tensorflow.org/
Verwendet von Googles DeepMind

Keras http://keras.io
Python Implementierung, basierend auf Theano oder Tensorflow

Torch http://torch.ch/
LuaJIT und C/CUDA, verwendet bei Facebook, Google, Twitter

DL4J http://deeplearning4j.org/
Plattformunabhängige Java Implementierung, kompatibel mit Spark, Hadoop

Caffe http://caffe.berkeleyvision.org/
C++, CUDA Implementierung mit Python und MATLAB Schnittstelle
Sehr schnell, viel verwendet für Bildanalyse z.B. bei Facebook

Weiterführende Literatur I


Bishop, C. M. (2006).
Pattern Recognition and Machine Learning.
Springer, 2. edition.


Hastie, T., Tibshirani, R., and Friedman, J. (2009).
The Elements of Statistical Learning: Data Mining, Inference, and Prediction.
Springer, 2. edition.


Michalski, R. S. (1986).
Understanding the nature of learning: Issues and research directions.
In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine Learning: An Artificial Intelligence Approach (Volume 2)*, pages 3–26. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.


Michalski, R. S. and Kodratoff, Y. (1990).
Research in machine learning: Recent progress, classification of methods, and future directions.
In Kodratoff, Y. and Michalski, R. S., editors, *Machine Learning*, pages 3–30. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Weiterführende Literatur II


Michalski, S. R., Carbonell, G. J., and Mitchell, M. T., editors (1986).
Machine learning an artificial intelligence approach volume II.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.


Mitchell, T. (1997).
Machine Learning.
McGraw Hill.


Simon, H. A. (1983).
Why should machines learn?
In *Machine Learning, An Artificial Intelligence Approach*. Tioga, Palo Alto, California.