Karlsruhe Institute of Technology

AIFB

Teil 5

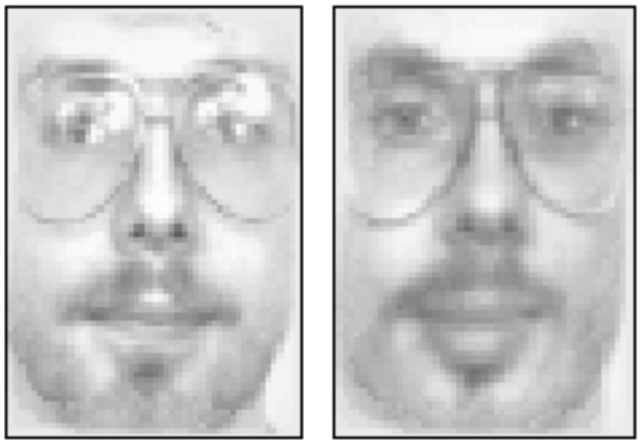
Maschinelles Lernen

Definitionen und Abgrenzungen

- Was ist Lernen?
  - Zentrale Fähigkeit von „intelligenten Systemen“ in Natur und KI
  - Zielgerichtete Veränderung von Wissen und/oder Verhaltensweisen durch Erfahrung
  - Verallgemeinerbarkeit auf neue Situationen
  - Viele Formen des Lernens:
    - motorisches Lernen, Regellernen, Sprachlernen, Lernen mit Lehrern, Lernen in der Entwicklung, ...
- Was ist kein Lernen?
  - festes Programmieren von Lösungen (keine Erfahrung)
  - Einfaches Speichern von Daten (keine Verallgemeinerung)

Beispiel: Gesichtserkennung

- Erkennung anhand von Gesichtern:
  - Person
  - Geschlecht
  - Stimmung



Bsp.: Wiskott et al.

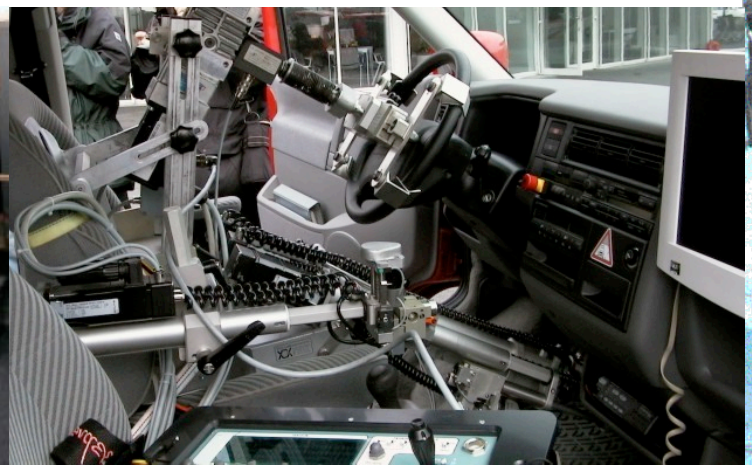
m	m	m	m		g	g	g	g
m	m	m	m		.	g	g	g
m	m	m	m		g	g	g	g
m	m	m	m		g	g	g	g
m	m	m	m		b	b	b	b
m	m	m	m		b	b	b	b
m	m	m	m		.	b	b	.

attributes determined: person is male, has glasses, and is bearded

Bsp.: SmartKom System

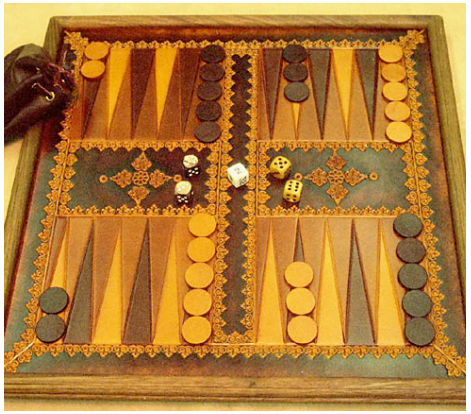
Beispiel: automatische Fahrzeugnavigation

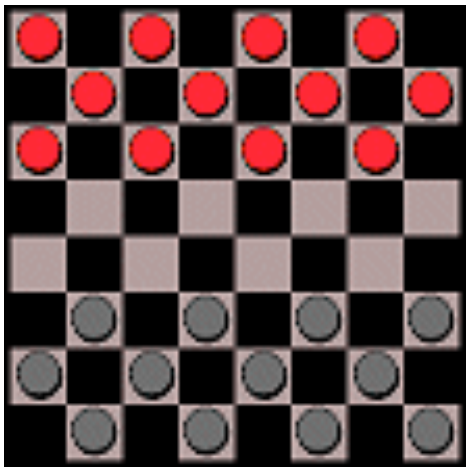
- Ziel selbständige Fahrzeuge in z.T. unbekannten Umgebungen
- Viele zu lernende Teilprobleme
  - ⇨ Steuerung und Aktorik
  - ⇨ Situationserkennung
  - ⇨ Planung und Navigation
  - ⇨ ...



Beispiel: Spiele

- Backgammon, Schach, Mühle etc.
- Gutes „Spielfeld“ für neue Verfahren
- Typische Domäne von Menschen
- Beispiel: TD-Gammon nutzt temporal difference (TD) Algorithmen und trainiert, während es mit sich selbst spielt
- Auf dem Treffen der American Association of Artificial Intelligence wurde 1998 ein Großmeister (der damalige Weltmeister) in 99 von 100 Spielen von einer Maschine besiegt.





Beispiel: Handschriftenerkennung

- Erkennen von handgeschriebenen Zeichen hat bis 1995 sehr schlecht funktioniert.
- Künstliche Neuronale Netze haben dies im Apple Newton wesentlich verbessert.
  "...vastly improved hand-writing recognition..." (BYTE May 1996)
- 2003: Die aktuelle Version von Mac OS besitzt eine integrierte Unterstützung für Stifteingabe, die auf der Schrifterkennung des Newton basiert.

Warum maschinelles Lernen?

- Neue Möglichkeiten mit Computern
  - Große Datenmengen können gesammelt, gespeichert und verarbeitet werden
- Neue Anwendungen
  - Data Mining: Extraktion von Wissen aus Daten
  - Selbst-adaptierende Programme/Filter: Anpassung an Nutzer und Situationen
  - Aktionslernen: Robotik, Steuerungen, Unterstützung bei Entscheidungen
  - Anwendungen, die schwer programmierbar sind (explizites Modell nicht verfügbar oder zu teuer), z.B.: Sprachverarbeitung, Fahrzeugsteuerung
- Besseres Verständnis des menschlichen Lernens und Lehrens
  - Kognitionswissenschaften: Theorien des Wissenserwerbs (z.B. durch Übung)
  - Technische Umsetzung: in Lernsysteme, Recommender Systeme etc.
- Maschinelles Lernen ist im Trend
  - Gute Fortschritte bei Algorithmen und Theorie
  - Wachsende Datenmengen, die automatisch verarbeitet werden müssen
  - Verfügbare Rechenleistung
  - Wachsender Markt und Industrie für Nutzung des maschinellen Lernens (z.B. Data Mining, Sprachsysteme, Bioinformatik)

Wozu maschinelles Lernen?

- Anwendungsfelder:
  - Data Mining:
    Extraktion von Wissen aus Daten
  - Selbst-adaptierende Programme/Filter:
    dynamische Anpassung an Nutzer und Situationen
  - Aktionslernen:
    Robotik, Steuerungen, Unterstützung bei Entscheidungen
  - Anwendungen, die schwer programmierbar sind (explizites Modell nicht verfügbar oder zu teuer):
    Sprachverarbeitung, Fahrzeugsteuerung
- Besseres Verständnis des menschlichen Lernens und Lehrens
  - Kognitionswissenschaften:
    Theorien des Wissenserwerbs (z.B. durch Übung)
  - Technische Umsetzung:
    in Lernsysteme, Recommender Systeme etc.

Spezifikation von Lernproblemen

- Lernen = Verbesserung der Leistung eines Systems
  - bei einer Aufgabe A,
  - in Bezug auf ein Leistungsmaß/eine Zielfunktion Z,
  - basierend auf der vorhergehenden Erfahrung E.
- Beispiel: Schach spielen
  - A: spiele Schach und gewinne
  - Z: Prozentsatz der gewonnenen Spiele in der ersten Liga
  - E: Menge bekannter Spiele
- Zu lösende Fragen (Modellierungsentscheidungen):
  - Wie genau soll die Erfahrung gesammelt werden?
  - Was genau soll eigentlich gelernt werden?
  - Wie wird das Gelernte repräsentiert?
  - Welcher Lernalgorithmus soll angewandt werden?

Beispiel: Schach spielen

- Typ der Erfahrung E im Training
  - Vorgegebene Situationen oder selbst erzeugte?
  - Bewertung mit oder ohne Lehrer?
  - Hintergrundwissen über das Spiel (z.B. Eröffnungen oder Endspiel)?
- Problem: Wie repräsentativ ist die Trainingserfahrung in Bezug auf das Lernziel?

- zu Lernendes wird i.A. repräsentiert als Funktion V
- Möglichkeiten für V:
  - Aktionsauswahl V: Brett → Zug
  - Brettauswertung V: Brett → Bewertung
  - Optionsvergleich V: Brett x Brett → {0,1}
- gelernt wird angenäherte Funktion V‘
  - Ziel des Lernprozesses: Annäherung von V‘ an das optimale V

Beispiel: Schach spielen - Zielfunktion

- Was ist die beste Zielfunktion?
  - ⇨ bestimmbar durch αβ-Suche
  - ⇨ liefert korrekte und optimale Werte, aber praktisch nicht berechenbar, da zu aufwendig.
- Was sind praktikable Alternativen?
  - ⇨ Sammlung von Regeln?
  - ⇨ Neuronales Netz?
  - ⇨ Polynome (z.B. linear, quadratisch) der Brettmerkmale?
  - ⇨ andere?
- Beispiel:

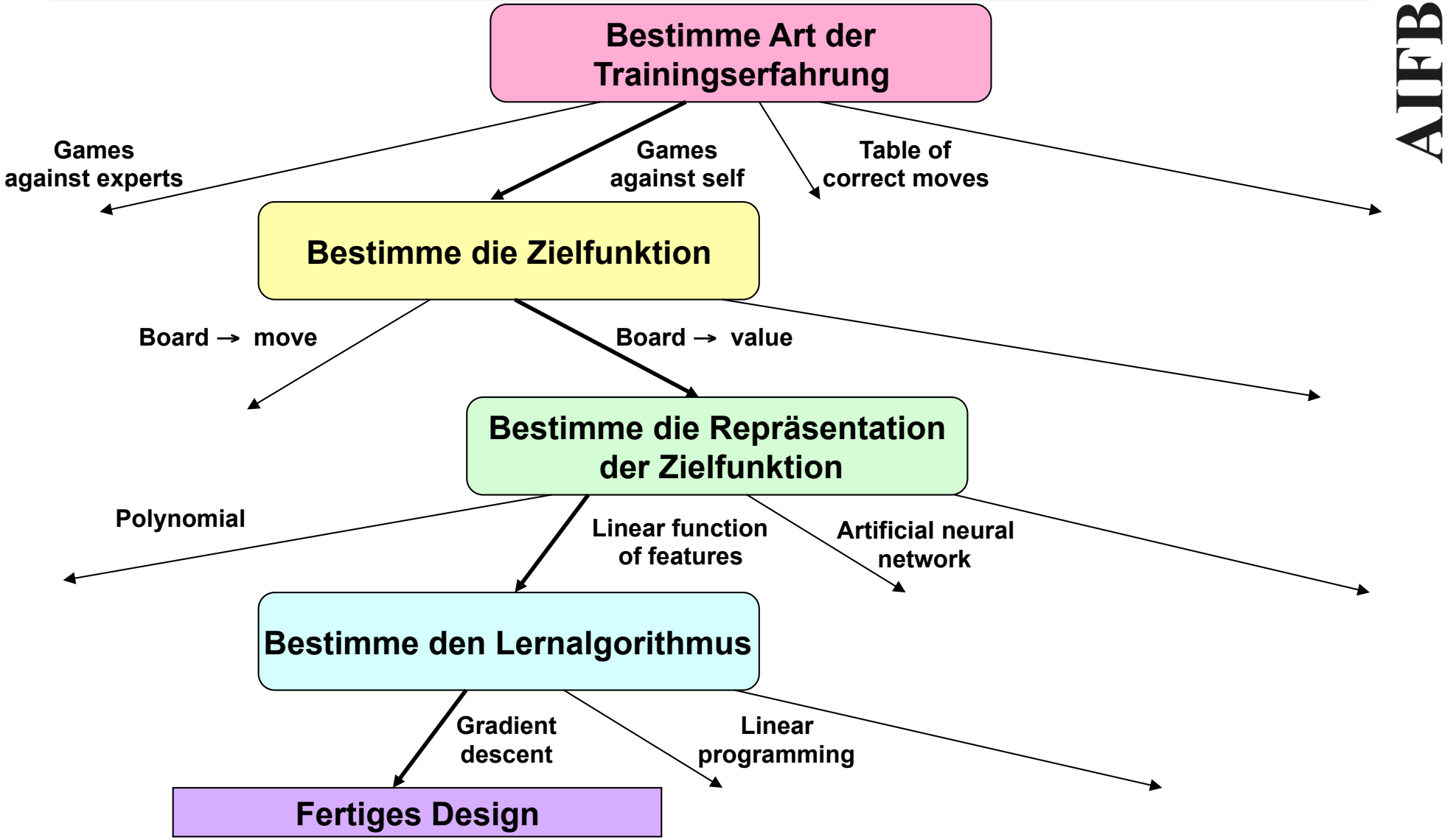
$$V'(b) = w_0 + w_1 x_1(b) + w_2 x_2(b) + w_3 x_3(b) + w_4 x_4(b) + w_5 x_5(b) + w_6 x_6(b)$$

  - ⇨ $x_{1/2}$ vorhandene weiße/schwarze Steine, $x_{3/4}$ bedrohte weiße/schwarze Steine, $x_{5/6}$ weiße/schwarze Offiziere

Beispiel: Schach spielen - Lernen

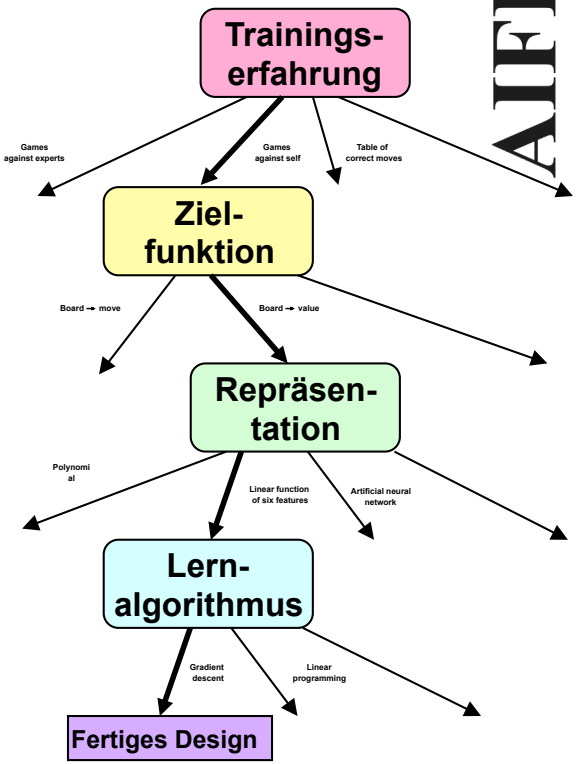
- Wie kann man trainieren?
  - ⇨ V zu lernende Funktion
  - ⇨ V‘ bisher gelernte Funktion
  - ⇨ (b, V_b) Trainingsbeispiel
- Möglichkeit, Trainingsbeispiel festzulegen:
  - ⇨ V_b := V‘($b_{Nachfolger}$)
- Lernregel
  - ⇨ Methode der kleinsten Quadrate (Least Mean Square,LMS):
    wiederhole
    - Zufällige Auswahl einer Brettsituation b mit bekanntem V_b
    - Fehlerberechnung für die Situation
      $error(b) := V_b - V‘(b)$
    - Für jedes Feature werden die Gewichte angepaßt:
      $w_i := w_i + \eta \cdot x_i \cdot error(b)$
    - η ist eine kleine Lernrate (konstant)

Problemlösung mit maschinellem Lernen



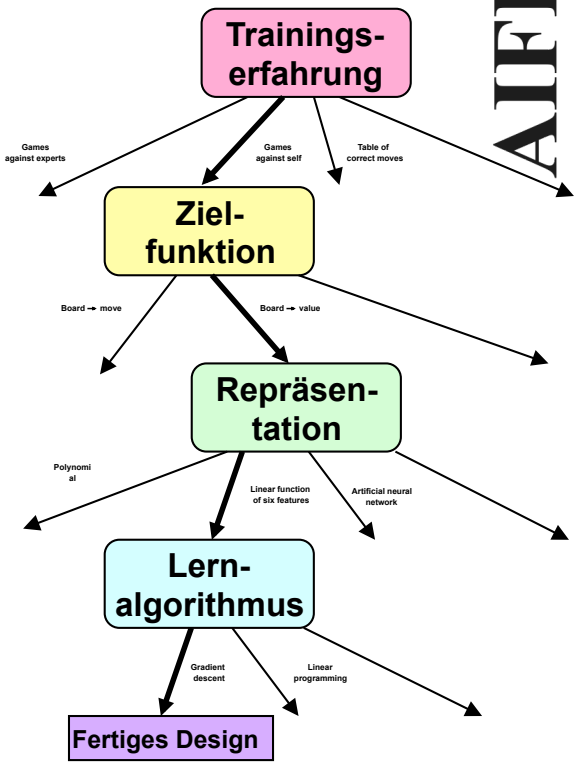
Problemlösung mit maschinellem Lernen

- Bemerkungen:
  - Dieses Vorgehen ist sehr allgemein.
  - Oft wird unter maschinellem Lernen nur der Teil „Lernalgorithmus“ verstanden, es geht aber um die ganze Pipeline.
  - Nur wenn alle Teile zusammenpassen ist ein gutes Resultat zu erwarten.
  - Dieses Modell ist modular und Verfahren auf den verschiedenen Ebenen können miteinander kombiniert werden.
  - Je nach zu lösendem Problem, kann es sein, dass es in mehrere Lernprobleme zerlegt werden muss, die durch unterschiedliche Ansätze gelöst werden.



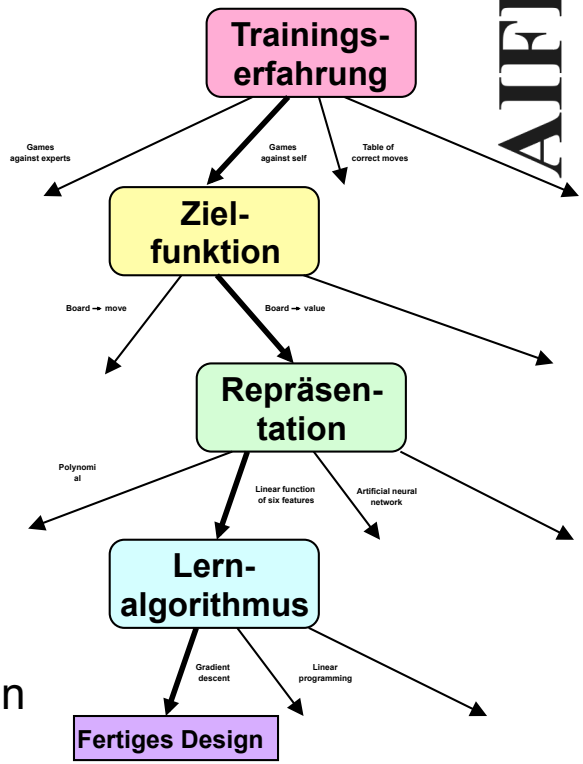
Trainingserfahrung

- Designentscheidungen:
  - Erzeugung von Beispielen
    - Beispiele aus Datenbanken/-sammlung
    - Erzeugung durch das System
    - Vorgabe durch Experten
  - Welche Vorgabe von außen?
    - Überwachtes Lernen (Bewertung durch Lehrer)
    - Unüberwachtes Lernen (keine Vorgabe)
    - Reinforcement Learning (Bewertung über Erfolg/ Misserfolg einer Serie von Entscheidungen)
  - Wann gibt es Vorgaben?
    - Getrennte Lern- und Kannphase
    - Kontinuierliches Lernen
    - Lernen mit Vergessen
  - Wann wird gelernt?
    - In jedem Schritt (Learning by Pattern)
    - Nach einigen Schritten (Learning by Block)
    - Nach einem kompletten Satz von Eingaben (Learning by Epoch)



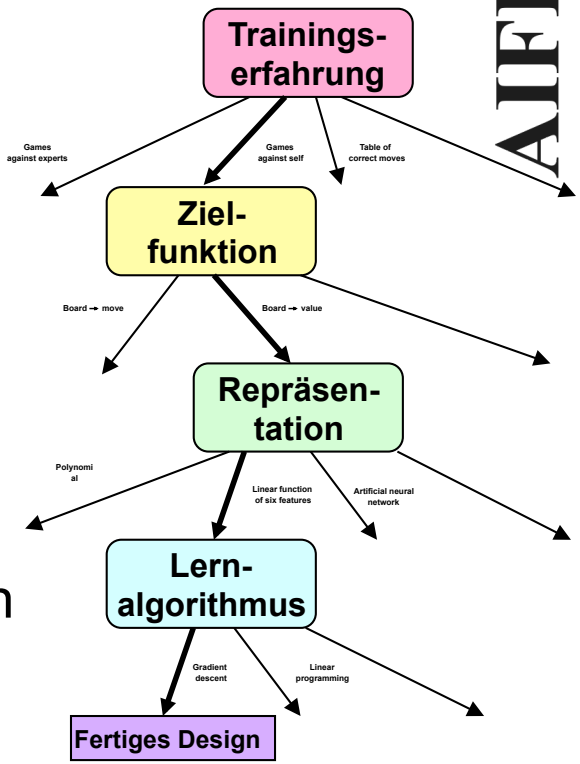
Zielfunktion

- oft „schwarze Kunst“, welche Kodierung für konkreten Fall am besten geeignet ist
- meist wird hier das Problem als Problem der Mustererkennung aufgefasst, also
  - Klassifikation von Mustern
    - Ist ein Produkt in Ordnung oder nicht?
    - Ist ein Patient krank oder gesund?
    - Ist eine handgeschriebene Ziffer eine 1,2,3 … ?
  - Bewertung von Mustern
    - Oft: Generalisierung (Schätzen von Werten für bestimmte Beobachtungen bei vorgegebenen Mustern. Rekonstruktion einer Funktion)
    - Was ist ein vernünftiger Preis für dieses Haus / Auto?
    - Wie gesund/krank ist ein Patient?
    - In der klassischen Statistik sind das Regression, Inter- sowie Extrapolation. Für hochdimensionale Probleme tritt in der Praxis meist Extrapolation auf
- Die Güte der Zielfunktion ist nicht unabhängig von der Wahl des Lernalgorithmus



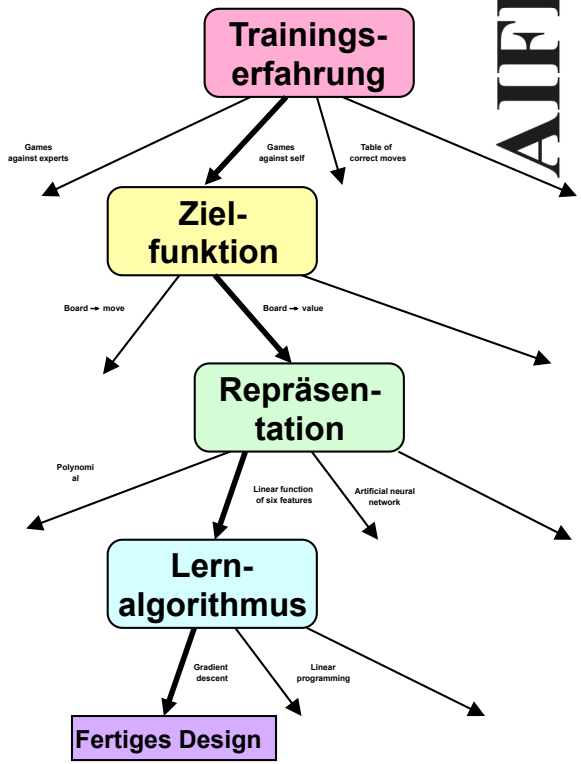
Repräsentation

- Zwei Lager: symbolisch und subsymbolisch
- Symbolisch
  - ⇨ Beispiele: Regeln, Entscheidungsbäume, logische Formeln, Beispielfälle
  - ⇨ Vorteil: Erklärungsmöglichkeit
- Subsymbolisch
  - ⇨ Beispiele: Statistik, Polynome, neuronale Netze, „Genstrings“
  - ⇨ Vorteil: Leistungsfähigkeit
- Repräsentation schränkt Auswahl der Lernalgorithmen stark ein
- Neben der Repräsentation ist auch die Kodierung der Eingangsdaten entscheidend
  - ⇨ Beispiel: Kodierung des Alters von Patienten als Integer, als reelle Werte, als binäre Klassenvariablen („Alter 0-10“, „Alter 11-30“, „über 30“)
  - ⇨ Ungeeignete Kodierungen können aus einfachen Lernproblemen schwierige machen!



Lernalgorithmus

- Eigentlicher Kern des Maschinellen Lernens
- Viele Verfahren aus verschiedenen Bereichen (Statistik, Logik, Neuro-wissenschaften, Optimierung, …)
  - ⇨ Schätzverfahren
  - ⇨ Induktives Schließen
  - ⇨ Case-Based Reasoning
  - ⇨ Backpropagation
  - ⇨ LMS-Verfahren
  - ⇨ Genetische und evolutionäre Algorithmen
  - ⇨ Support-Vektor-Maschinen
  - ⇨ Selbstorganisierende Karten

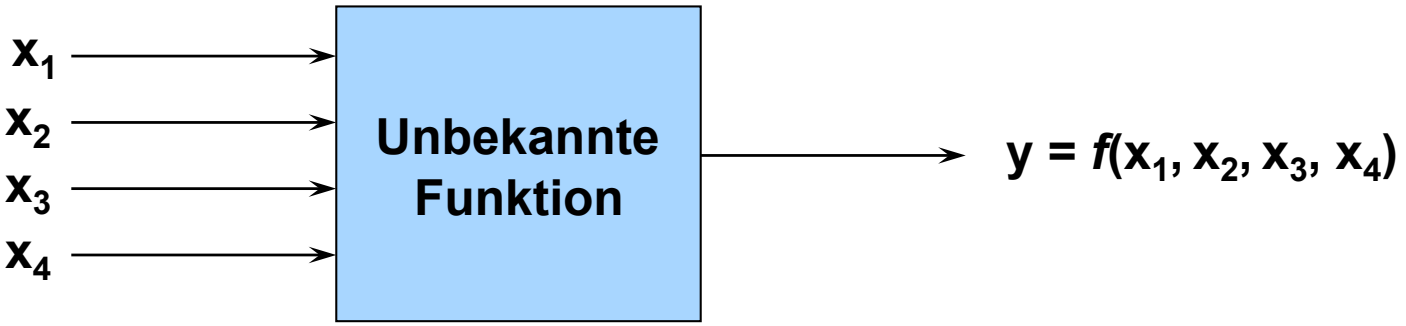


Herausforderungen beim Maschinellen Lernen

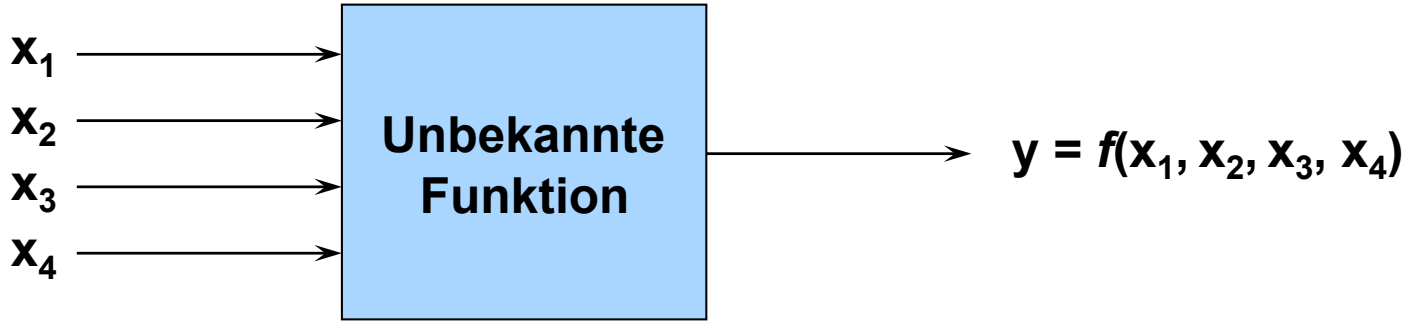
- Welche Verfahren eignen sich wann, um Funktionen anzunähern?
- Entscheidungen beim Design einer Lösung...
  - ⇨ Anzahl der Beispiele?
  - ⇨ Komplexität des Problems
- Wie wird die Lösung des Problems durch Umwelt beeinflusst?
  - ⇨ Verrauschte Daten
  - ⇨ Mehrere Datenquellen
- Was sind die theoretischen Grenzen der Lernbarkeit?
- Wie kann Vorwissen genutzt werden?
- Welche Anregungen können wir aus der Natur abschauen?
- Wie können Systeme selbstständig Repräsentationen finden?

Lernen aus Beispielen

- Gegeben:
  Trainingsbeispiele <x, *f*(x)> einer unbekannten Funktion *f*(x)
- Gesucht: Gute Approximation von *f*
- Einordnung: überwachtes Lernen/Mustererkennung
- Anwendungen:
  - Diagnose von Krankheiten
    - x = Patientendaten (Geschlecht, Alter, Erkrankung, …)
    - *f* = Krankheit (oder vorgeschlagene Therapie)
  - Risikobewertung
    - x = Eigenschaften von Kunden (Demographie, Unfälle, Vorgeschichte, …)
    - *f* = Risikoeinschätzung
  - Part-of-speech Tagging
  - …



Ein Lernproblem



Example	x_1	x_2	x_3	x_4	y
0	0	1	1	0	0
1	0	0	0	0	0
2	0	0	1	1	1
3	1	0	0	1	1
4	0	1	1	0	0
5	1	1	0	0	0
6	0	1	0	1	0

- Wir schreiben A →B für die Menge aller Funktionen, die Elemente von A auf Elemente von B abbilden
- $x_i \in X_i$, $y \in Y$, $f \in (X_1 \times X_2 \times X_3 \times X_4) \rightarrow Y$
- Definitionsbereich von $x=(x_1, x_2, x_3, x_4)$ ist $X=(X_1 \times X_2 \times X_3 \times X_4)$, d.h. $f \in X \rightarrow Y$
- Beispiel: $X_i = Y = \{0,1\}$

Hypothesen-Raum (unbeschränkter Fall)

- Wieviel mögliche Funktionen *f* (Hypothesen) gibt es?
- $| A \rightarrow B | = | B |^{| A |}$
- $| \{0,1\} \times \{0,1\} \times \{0,1\} \times \{0,1\} \rightarrow \{0,1\} | = 2^{16} = 65536$ mögliche *f*
- Naiver Ansatz: Streiche mit jedem Beispiel alle nicht passenden *f*
  - ⇨ Dazu müssen alle möglichen Eingaben betrachtet werden
  - ⇨ Nach 7 Beispielen bleiben noch $2^9 = 512$ Hypothesen für *f* (von 65536)
  - ⇨ Keine Vorhersage für unbekannte Beispiele
  - ⇨ kein echtes (verallgemeinerndes) Lernen
  - ⇨ eher: Look-Up-Table

Example	x_1	x_2	x_3	x_4	y
0	0	0	0	0	?
1	0	0	0	1	?
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	?
8	1	0	0	0	?
9	1	0	0	1	1
10	1	0	1	0	?
11	1	0	1	1	?
12	1	1	0	0	0
13	1	1	0	1	?
14	1	1	1	0	?
15	1	1	1	1	?

Spezialfall für Lernen aus Beispielen: Begriffslernen

- Ein **Begriff** (engl.: concept) bezeichnet eine Menge von Entitäten (Begriffsumfang) mit gemeinsamen Eigenschaften (Begriffsinhalt).
- Beim **Begriffslernen** (concept learning) enthält der Wertebereich der zu lernenden Funktion f nur die Werte 1 (wahr, gehört zum Begriff) und 0 (falsch, gehört nicht zum Begriff).

Lernen eines Begriffs, Beispiel EnjoySport

- Begriff EnjoySport umfasst die Tage, an denen Harry gern Sport treibt
- Beispiele
  - ⇨ Ähnlich zur Definition von Datentypen, oft Aufzählungs-Datentypen
  - ⇨ Hier: ∈ 6 Attribute:

    Sky∈{Rainy, Sunny} Temp∈{Warm, Cold}

    Humidity∈{Normal, High} Wind∈{None, Mild, Strong}

    Water∈{Cool, Warm} Forecast∈{Same, Change}

Example	Sky	Air Temp	Humidity	Wind	Water	Forecast	Enjoy Sport
0	Sunny	Warm	Normal	Strong	Warm	Same	**Yes**
1	Sunny	Warm	High	Strong	Warm	Same	**Yes**
2	Rainy	Cold	High	Strong	Warm	Change	**No**
3	Sunny	Warm	High	Strong	Cool	Change	**Yes**

- Ziel: Finde Beschreibung des Begriffes, generalisiere für unbekannte Daten

Repräsentierung der Hypothesen

- Viele Möglichkeiten
- Vorschlag: Jede Hypothese ist eine Konjunktion von Attributbeschreibungen:
  - ⇨ Für jedes Attribut werden Constraints/Bedingungen angegeben:
    - ✧ Spezifische Werte: z.B. Water = Warm
    - ✧ oder: Wert ist egal : z.B. Water = ?
    - ✧ oder: gar kein Wert erlaubt : z.B. Water = Ø
- Beispiel-Hypothese für *EnjoySport*

Sky	AirTemp	Humidity	Wind	Water	Forecast
<Sunny	?	?	Strong	?	Same>

  - ⇨ Ist diese Hypothese konsistent mit den Trainingsbeispielen?
  - ⇨ Wie sehen Hypothesen aus, die konsistent mit den Beispielen sind?
- Hypothese $h \in H$
- H ist die Menge aller möglichen Hypothesen $H \subseteq X \rightarrow \{0,1\}$
- Bemerkung: <Ø,Ø,Ø,Ø,Ø> = <?,?,?,?,Ø>, …
  sobald ein Ø vorkommt entspricht die Hypothese der Funktion $f(x)=0$
- Anzahl der möglichen Hypothesen: $(3 \cdot 3 \cdot 3 \cdot 3 \cdot 4 \cdot 3) + 1 = 973 = |H|$
- Bemerkung: $|H|=973 << |X \rightarrow \{0,1\}| = 2^{(2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 2)} = 2^{96} \approx 7{,}9 \cdot 10^{27}$

Lernen des Konzepts *EnjoySports*

- Gegeben:
  Menge D von Trainingsbeispielen <x, *f*(x)>,
  - ⇨ die gegebene Tage mit den Attributen Sky, AirTemp, Humidity, Wind, Water, Forecast beschreiben (x)
  - ⇨ X = {{Rainy, Sunny} × {Warm, Cold} × {Normal, High} × {None, Mild, Strong} × {Cool, Warm} × {Same, Change}}
  - ⇨ Angabe, ob ein Tag mit solchen Attributen zu dem Zielkonzept gehören (*f*(x)) also **positive** und **negative Beispiele** für Zielfunktion (*f*(x)=0 oder *f*(x)=1) $<x^1,f(x^1)>$, $<x^2,f(x^2)>$, $<x^3,f(x^3)>$, $<x^4,f(x^4)>$, $<x^5,f(x^5)>$, …

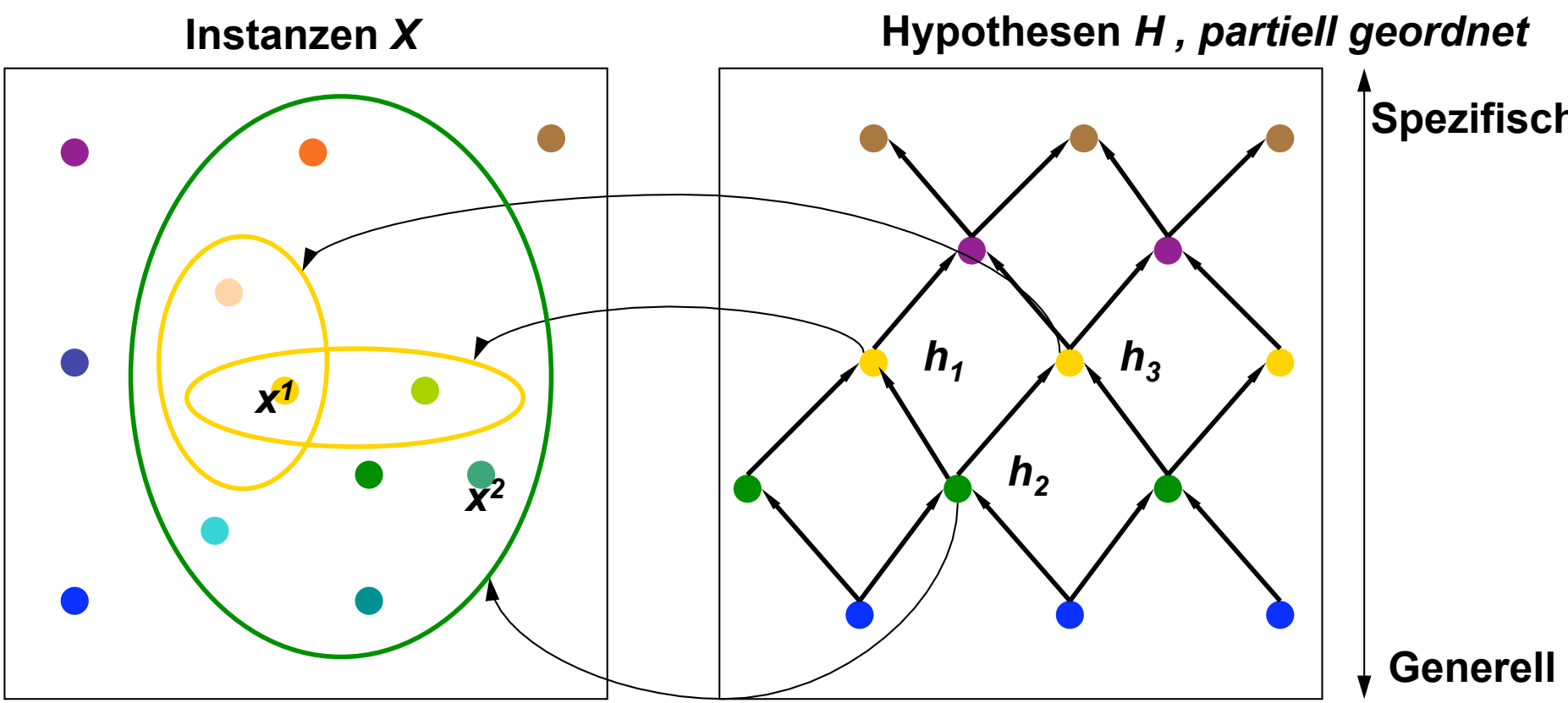
- Gesucht:
  - ⇨ Hypothese *h* als Konjunktion von Attributen (z.B. <?, Cold, High, ?, ?, ?>)
  - ⇨ *h* ist Annäherung für „verborgene“ Zielfunktion $f \equiv$ EnjoySport: $X \rightarrow \{0,1\}$
  - ⇨ d.h.: Hypothese $h \in H$ so dass $h(x) = f(x)$ für alle $x \in D$
  - ⇨ Solche *h* nennt man **konsistent** mit der Trainingsmenge D

- Trainingsannahmen:
  - ⇨ es fehlen keine Werte
  - ⇨ kein Rauschen in den Daten (widersprüchliche Daten)
  - ⇨ es gibt eine Hypothese, die konsistent mit D ist (die *f* annähert)

Grundidee des induktiven Lernens

- Induktion vs. Deduktion
  - Deduktion: aus gegebenen Fakten neue Fakten ableiten (Beweis)
    - Gegeben: A⇒B, B⇒C, Folgerung: A⇒C
    - "Vom Allgemeinen zum Speziellen"
  - Induktion: aus gegebenen Fakten neue Fakten mutmaßen (Hypothese)
    - Gegeben: Fliegt(Amsel), Fliegt(Storch), Folgerung: Fliegt(Vogel)
    - "Vom Speziellen zum Allgemeinen"
- Idee des induktiven Lernens:
  - Lerne eine Funktion aus Beispielen durch (schrittweise) Verallgemeinerung
- Annahme beim induktiven Lernen
  - Eine Hypothese, die für eine hinreichend große Beispielmenge die Zielfunktion gut annähert, wird die Funktion auch gut für unbekannte Beispiele annähern.
- Aber zunächst: Wie kann ein solches Lernverfahren konkret aussehen?

Instanzen- und Hypothesenraum



x_1 = <Sunny, Warm, High, Strong, Cool, Same>
x_2 = <Sunny, Warm, High, Light, Warm, Same>

h_1 = <Sunny, ?, ?, Strong, ?, ?>
h_2 = <Sunny, ?, ?, ?, ?, ?>
h_3 = <Sunny, ?, ?, ?, Cool, ?>

$\geq_g$ ≡ More-General-Than-Or-Equal
$>_g$ ≡ More-General-Than

$\geq_g$ ist partielle Ordnung:
transitiv, reflexiv, antisymmetrisch

$h_2 \geq_g h_1$, $h_2 \geq_g h_3$

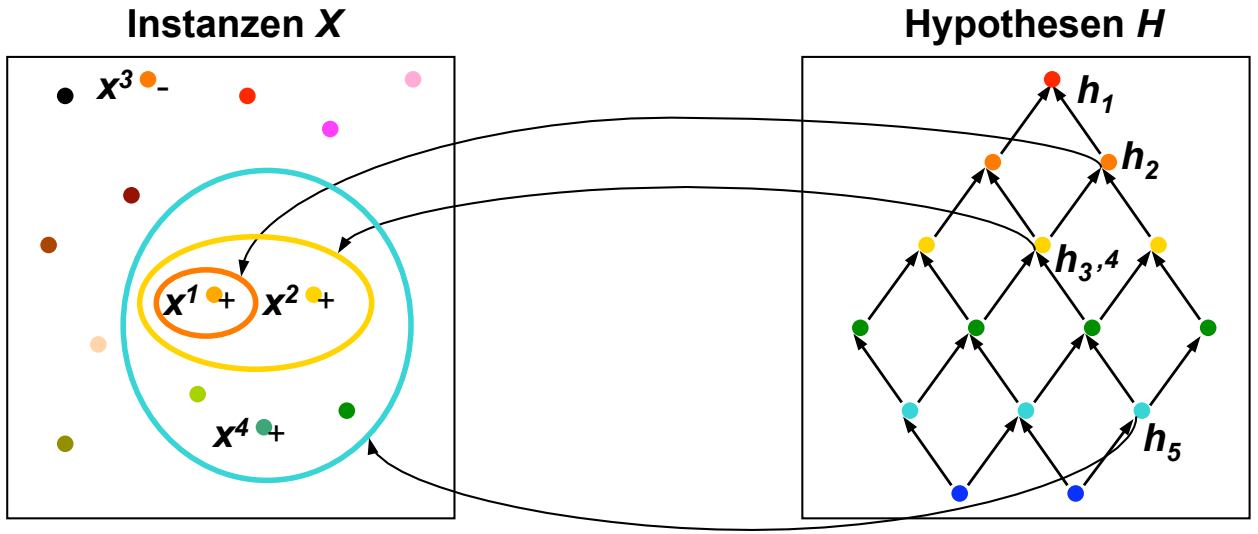
Aber weder $h_1 \geq_g h_3$ noch $h_3 \geq_g h_1$

Der Find-S Algorithmus

1. Initialisiere h als spezifischste mögliche Hypothese aus H
   (In unserem Fall ist das die Hypothese <Ø,Ø,Ø,Ø,Ø>)
2. ***FÜRALLE*** positive Instanzen x aus D, d.h. (*f*(x)=1)
   FÜRALLE Attribut mit Wert a_i in *h*
   WENN a_i aus *h* wird durch x erfüllt
   DANN keine Änderung
   SONST ersetze a_i durch den nächst allgemeineren Wert, der durch x erfüllt wird
3. Ausgabe von Hypothese *h*

- Bemerkungen:
  - negative Beispiele werden nicht betrachtet
  - Verfahren sucht “minimale” bzw. spezifischste Hypothese aus H, die alle positiven Beispiele umfaßt
  - Wenn es eine Lösung gibt, dann wird sie auch gefunden
  - Was ist, wenn es keine Lösung gibt?

Find-S: Suche im Hypothesenraum



x_1 = <Sunny, Warm, Normal, Strong, Warm, Same>, +
x_2 = <Sunny, Warm, High, Strong, Warm, Same>, +
x_3 = <Rainy, Cold, High, Strong, Warm, Change>, -
x_4 = <Sunny, Warm, High, Strong, Cool, Change>, +

h_1 = <Ø, Ø, Ø, Ø, Ø, Ø>
h_2 = <Sunny, Warm, Normal, Strong, Warm, Same>
h_3 = <Sunny, Warm, ?, Strong, Warm, Same>
h_4 = <Sunny, Warm, ?, Strong, Warm, Same>
h_5 = <Sunny, Warm, ?, Strong, ?, ?>

- Probleme von Find-S
  - ⇨ Sagt nicht, ob der Begriff tatsächlich gelernt wurde
  - ⇨ Keine Aussage, ob Daten inkonsistent sind
  - ⇨ Spezifischste Hypothese muss nicht die einzige Lösung sein
  - ⇨ Spezifischste Hypothese muss nicht die beste Lösung sein

Versionsraum

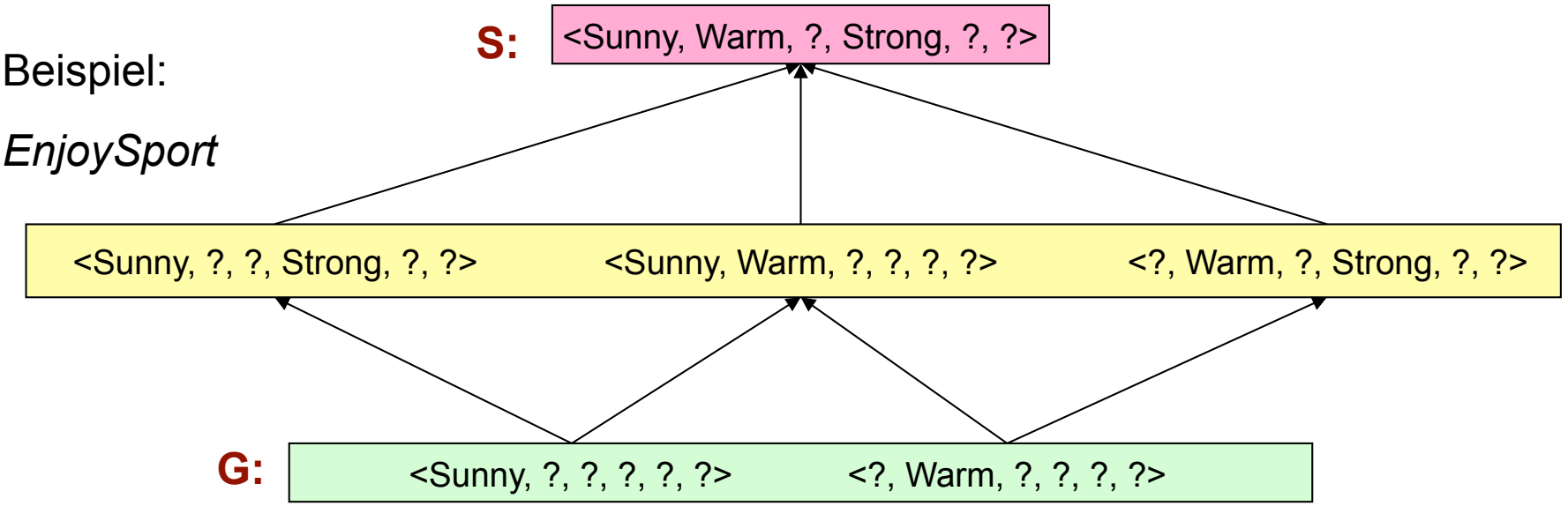
- Definition: konsistente Hypothesen
  - Eine Hypothese *h* ist **konsistent** mit einer Menge von Trainingsdaten D eines Zielbegriffs *f* genau dann wenn *h*(x) = *f*(x) für alle Beispiele <x, *f*(x)> aus D gilt
  - konsistent(h, D) :≡ ∀ <x, *f*(x)> ∈ D: *h*(x) = *f*(x)
- Definition: Versionsraum
  - Der **Versionsraum** $VS_{H,D}$, zu einem Hypothesenraum H und Trainingsbeispielen D ist die Teilmenge der Hypothesen aus H, die konsistent mit allen Trainingsdaten aus D sind
  - $VS_{H,D}$:≡ { *h* ∈ H | konsistent(*h*, D) }
- Bemerkung:
  - Wenn wir nicht nur an einer gültigen Lösung interessiert sind, dann ist der Versionsraum interessant.

Der List-Then-Eliminate Algorithmus

1. Initialisierung: ***SETZE*** VS := H,
   also ist VS die Liste aller Hypothesen
2. ***FÜRALLE*** Trainingsbeispiele <x, *f*(x)>
   FÜRALLE Versionen *h*∈VS
   FALLS *h*(x) ≠ *f*(x)
   SETZE VS := VS\{*h*}
3. Ausgabe des Versionsraums VS

Beispiel:

EnjoySport



Repräsentierung des Versionsraumes

- Bemerkung zum Versionsraum
  - ⇨ Offensichtlich gibt es am meisten spezifische und am meisten generelle Versionen
  - ⇨ Jede Hypothese, die “dazwischen” liegt, muss auch eine Lösung sein

- Definition: Generelle Schranke
  - ⇨ Die generelle Schranke G des Versionsraumes $VS_{H,D}$ ist die Menge der generellsten Hypothesen
  - ⇨ $G = \{g \in H \mid \text{konsistent}(g,D) \text{ und } \forall h \in H: (h >_g g \Rightarrow \neg\text{konsistent}(h,D))\}$
  - ⇨ Das heißt, jedes *h*, das mehr Elemente zum Begriff dazunimmt als ein *g*, nimmt auch Fehler auf

- Definition: Spezifische Schranke
  - ⇨ Die spezifische Schranke S des Versionsraumes $VS_{H,D}$ ist die Menge der spezifischsten Hypothesen
  - ⇨ $G = \{g \in H \mid \text{konsistent}(g,D) \text{ und } \forall h \in H: (g >_g h \Rightarrow \neg\text{konsistent}(h,D))\}$
  - ⇨ Das heißt, jedes *h*, das weniger Elemente im Begriff hat als ein *g*, dem fehlen gültige Beispiele

Versionsraum-Theorem

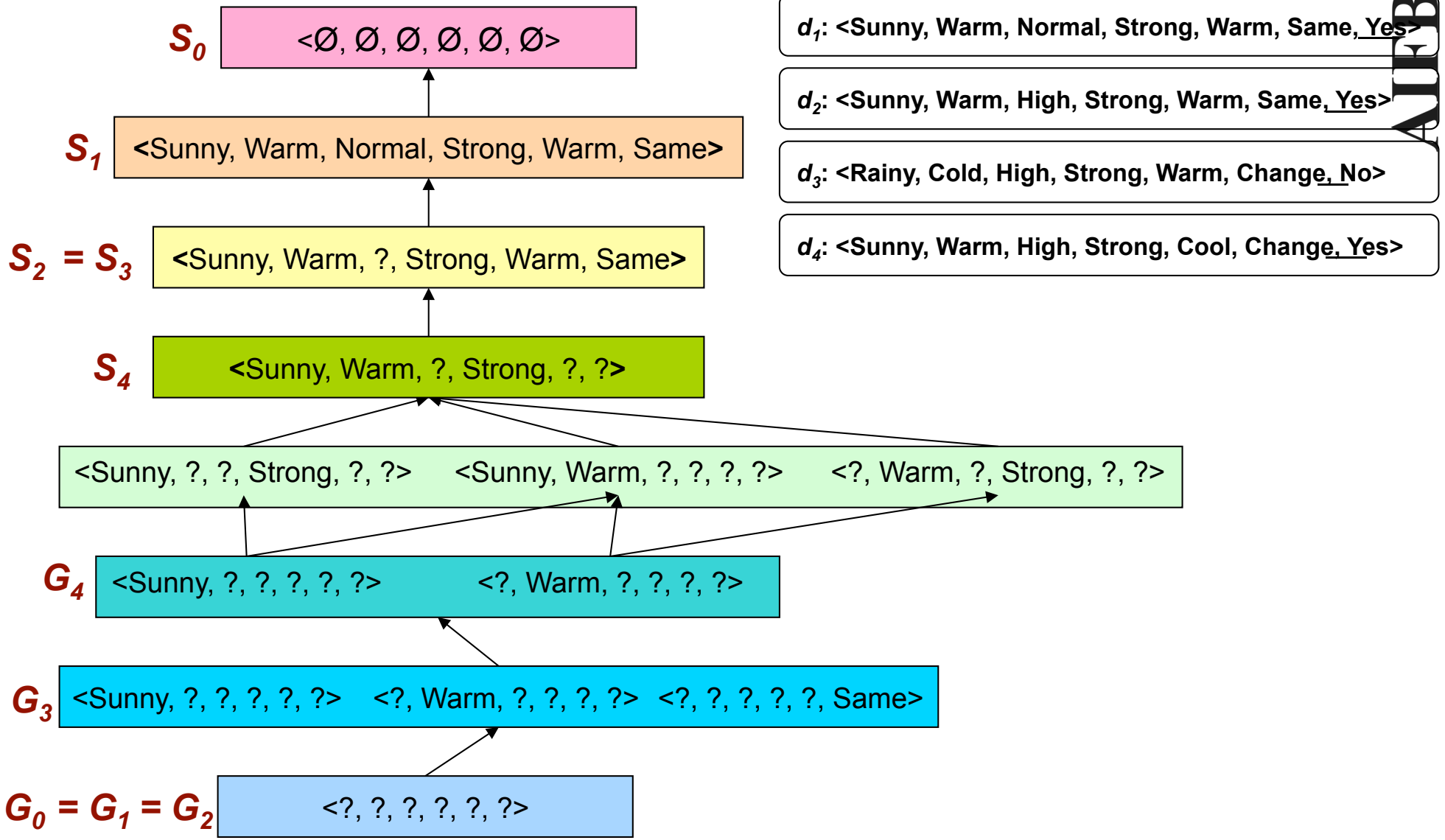
- Versionsraumtheorem
  - Jede Hypothese des Versionsraums liegt zwischen S und G
  - $VS_{H,D} = \{ h \in H \mid \exists s \in S: \exists g \in G: g \geq_g h \geq_g s \}$
- Bemerkung
  - Beweis: Übung und/oder bei Mitchell nachschauen
  - Hinweis:
    - Zeige erst: Wenn $g, s \in VS_{H,D}$ und $g \geq_g h \geq_g s$, dann ist auch $h \in VS_{H,D}$
    - Zeige dann: Wenn $h \in VS_{H,D}$, dann gibt es ein $g \in G$, mit $g \geq_g h$
      - Beweisidee:
        - Aus Definition von G folgt:

          Es gilt entweder $h \in G$ oder es gibt ein $h' >_g h$, $h' \in VS_{H,D}$
        - Dann kann man das gleiche mit h' fortführen.
        - Da H endlich ist, muß man irgendwann an ein h'' kommen mit $h'' \in G$
    - Und analog: Wenn $h \in VS_{H,D}$, dann gibt es ein $s \in S$, mit $h \geq_g s$

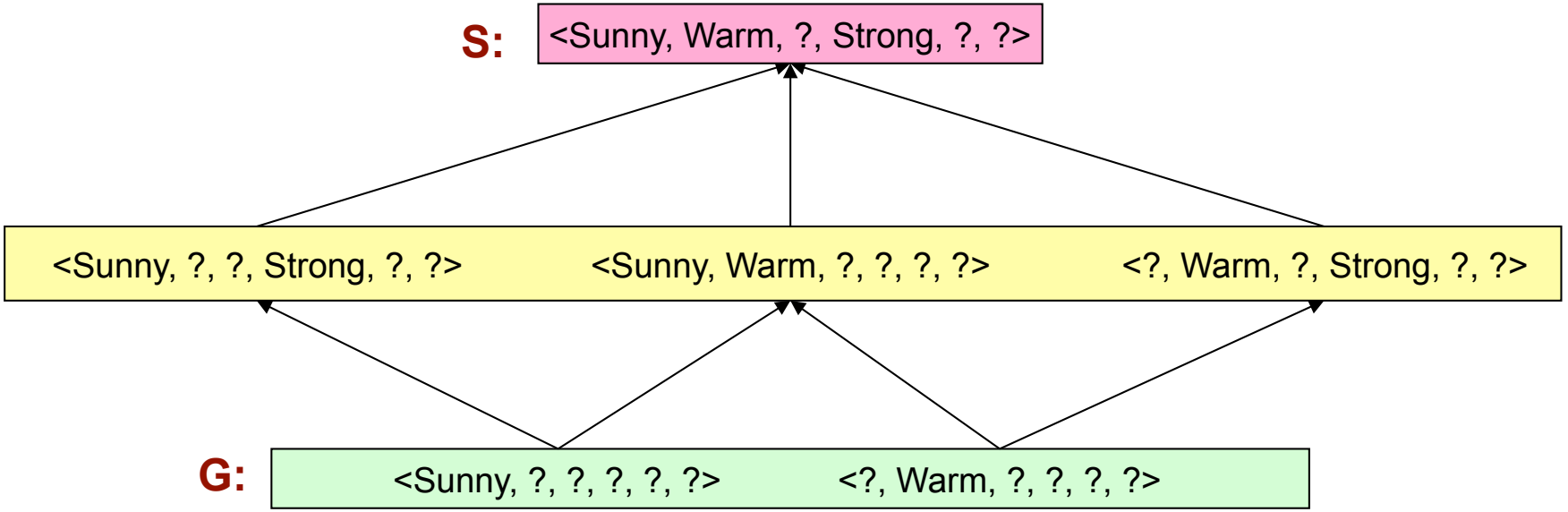
Kandidateneliminationsalgorithmus

1. Initialisierung
   G = {<?, … , ?>}, *Menge mit generellstem Element*
   S = {<Ø, … , Ø>}, *Menge mit speziellstem Element*
2. ***FÜRALLE*** Trainingsbeispiele d=<x, *f*(x)>∈D
   WENN *f*(x)=1, *d.h. für positive Beispiele*
      FÜRALLE *g*∈G mit *g*(x)=0 **SETZE** G:=G\{*g*} *(d.h. lösche alle g∈G, die schon zu klein sind)*
      FÜRALLE *s*∈S mit *s*(x)=0
         SETZE S:=S\{s}
         Erweitere S um alle minimalen Generalisierungen *h* von *s*, so dass
            - *h*(x) = 1, d.h., *h* ist konsistent mit d
            - ∃*g*∈G: $g \geq_g h$, d.h., *h* liegt noch unter der generellen Schranke
         Lösche alle *s*∈S, die genereller sind als andere Elemente aus S
   WENN *f*(x)=0, *d.h. für negative Beispiele*
      FÜRALLE *s*∈S mit *s*(x)=1 ***SETZE*** S:=S\{*s*} *(d.h. lösche alle s∈S, die schon zu groß sind)*
      FÜRALLE *g*∈G mit *g*(x)=1
         SETZE G:=G\{*g*}
         Erweitere G um alle minimalen Spezialisierungen *h* von *g*, so dass
            h(x) = 0, d.h., *h* ist konsistent mit d
            ∃ *s*∈S: $h \geq_g s$, d.h., *h* liegt noch über der speziellen Schranke
         Lösche alle *g*∈G, die spezieller sind als andere Elemente aus G

Beispiellauf



Was macht man mit dem Versionsraum?



- Bewertung von neuen Beispielen
  - <Sunny, Warm, Normal, Strong, Cool, Change>
  - <Rainy, Cold, Normal, Light, Warm, Same>
  - <Sunny, Warm, Normal, Light, Warm, Same>
  - <Sunny, Cold, Normal, Light, Warm, Same>
- Möglichkeiten:
  - Klare Entscheidungen, Mehrheiten, Unentschieden
- Auswahl neuer Trainingsbeispiele
  - Ideal: solche, die den Versionsraum in zwei gleich große Teile teilen

Generalisierung beim induktiven Lernen

- Beispiel für induktive Generalisierung
  - Positive Beispiele:
    <Sunny, Warm, Normal, Strong, Cool, Change, Yes>
    <Sunny, Warm, Normal, Light, Warm, Same, Yes>
  - Gefolgertes S:
    <Sunny, Warm, Normal, ?, ?, ?>
- Fragen:
  - Was rechtfertigt eine Generalisierung?
    - z.B. <Sunny, Warm, Normal, Strong, Warm, Same>
  - Wann gibt es ausreichende Informationen für eine Generalisierung?

Der induktive Bias

- Bias (engl.) – Vorliebe, Voreingenommenheit, Befangenheit, systematischer Fehler, …
- Induktiver Bias:
  - ⇨ Menge der möglichen Hypothesen beschränkt die möglichen Lösungen *h*, die *f* annähern können
  - ⇨ Der induktive Bias beschreibt alle Grundannahmen, die in dem Lern- und Klassifikationsverfahren stecken.
  - ⇨ Ohne induktiven Bias gibt es keine Generalisierung!

Lernen ohne induktiven Bias?

- Bisher hatte H einen induktiven Bias
  - Nur Konjunktionen (Und-Verknüpfung) und “Egal “ = “?”
  - Welche Begriffe können damit nicht erfasst werden?
- Hypothesenraum ohne Bias
  - Wähle ein H’, das alle möglichen Begriffe erfasst
  - das heißt, H’ ist die Potenzmenge von X
  - H’ erlaubt neben Konjunktionen auch Disjunktionen (Oder) und Negationen
  - Erinnerung:
    - $| A \rightarrow B | = | B |^{| A |}$, also
      $|H'|=|X \rightarrow \{0,1\}|=2^{(2\cdot2\cdot2\cdot2\cdot3\cdot2)} = 2^{96} \approx 7{,}9\cdot10^{27}$
    - Wogegen: $|H| = (3\cdot3\cdot3\cdot3\cdot4\cdot3) +1 = 973$
- Wie sehen Schranken S, G und der Versionsraum bei H’ aus?
  - Bemerkung: Die Schranken und der Versionsraum beschreiben die verbliebene Unsicherheit des Lernverfahrens
  - S’ ist die Disjunktion aller positiven Beispiele
    - Das spezifischste Konzept besteht aus gerade den beobachteten bisherigen positiven Beispielen
    - S’ hat also genau ein Element: *s*’
  - G’ ist die Negation der Konjunktion aller negativen Beispiele
    - Das generellste Konzept hat keines der negativen Beispiele drin, aber alle anderen
    - G’ hat also genau ein Element: *g*’

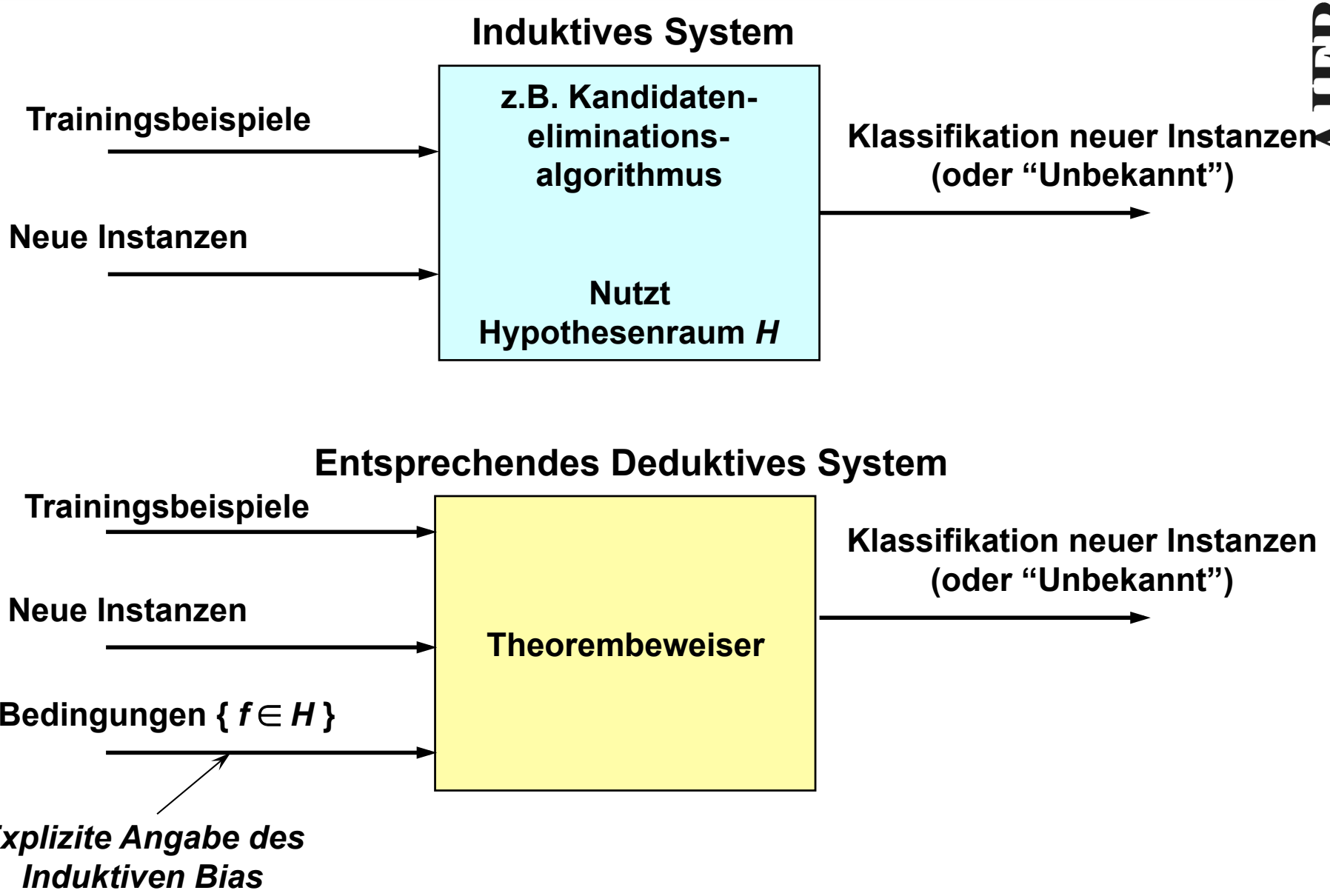
Lernen ohne induktiven Bias?

- In H' entspricht jedes *h* einer Teilmenge von X
- Der Versionsraum $VS_{H',D}$ besteht aus allen Teilmengen von X, die zwischen *s*' und *g*' liegen: $VS_{H',D}=\{h \mid s' \subseteq h \subseteq g'\}$
- Wie kann man mit S' und G' ein neues x bewerten?
  - ⇨ Versionsraum spannt alle mit den Trainingsbeispielen konsistente Konzepte zwischen S' und G' auf.
- Mehrheitsentscheidung?
  - ⇨ Wenn x neu ist, dann ist $x \notin s'$ und $x \in g'$
  - ⇨ Sei *h* eine konsistente Hypothese, $h \in VS_{H',D}$, d.h. $s' \subseteq h \subseteq g'$
  - ⇨ Wenn $x \in h$ ist, dann ist auch $h'=h \setminus \{x\} \in VS_{H',D}$
  - ⇨ Wenn $x \notin h$ ist, dann ist auch $h'=h \cup \{x\} \in VS_{H',D}$
  - ⇨ Das heißt: Es gibt genauso viele konsistente Hypothesen dafür, dass x positiv ist, wie dafür dass x negativ ist.
  - ⇨ Also: immer unentschieden!
- Fazit:
  - ⇨ ohne Bias/Annahmen kann man überhaupt nicht generalisieren,
  - ⇨ sondern nur bekannte Beispiele bewerten
  - ⇨ Also nur Speichern und nicht Lernen!

Induktiver Bias – formale Sicht

- Komponenten einer formalen Definition für den induktiven Bias
  - Algorithmus zum Lernen von Begriffen L
  - Instanzen X und Zielfunktion *f*
  - Trainingsbeispiele D = {<x, *f*(x)>}
  - L(x, D) = Klassifikation von x durch den Lerner L nach Training mit D
- Definition
  - Der induktive Bias des Lerners L ist jede minimale Menge von Bedingungen B über die Zielfunktion, so dass für jeder Begriff *f* mit zugehöriger Trainingsmenge D gilt:
  - $\forall x \in X: ((B \wedge D \wedge x) \vdash L(x, D))$
  - wobei $A \vdash B$ bedeutet, B aus A logisch folgerbar ist
  - Das heißt, man bevorzugt bestimmte Hypothesen durch strukturelle Einschränkungen
- Also
  - Vorgegebene Annahmen über den zu lernenden Begriff
  - Dadurch Ermöglichung von Generalisierung

Induktion vs. Deduktion



Lerner mit unterschiedlichem Bias

- Lerner ohne Bias, “Auswendiglernen”
  - Nur Klassifizierung von vorher Gesehenem
  - Speichert Beispiele
  - Kann nur solche x klassifizieren, die schon gesehen wurden
- Versionsraum und Kandidateneliminationsalgorithmus
  - Stärkerer Bias: Konzepte lassen sich als *h* aus H beschreiben
  - Speichert Schranken für Generalisierungen und Spezialisierungen
  - Klassifikation von x genau dann, wenn es im Versionsraum liegt und alle Versionen in der Bewertung übereinstimmen
- Kandidateneliminationsalgorithmus mit Mehrheitsentscheidung
  - Noch stärkerer Bias: Konzepte lassen sich als *h* aus H beschreiben
    Mehrheit der Konzepte in H wird recht haben
  - Klassifikation von x, wenn es eine mehrheitliche Klassifikation der Versionen gibt (z.B. mehr als 70%)
  - Hier sind implizit statistische Annahmen vorhanden, wie repräsentativ die Trainingsbeispiele sind
- Find-S
  - Noch stärkerer Bias: die spezifischste Hypothese gilt
  - Implizite Annahme: alle noch nicht bekannten Beispiele sind negativ
  - Klassifiziert x anhand von S

Zusammenfassung einiger Begriffe

- Überwachtes Lernen
  - Begriff – teilt X in zugehörige und nicht zugehörige (also +/-) Beispiele
  - Zielfunktion – die Funktion *f*(x) die jeder Eingabe eine Bewertung zuordnet (z.B. Begriffszugehörigkeit)
  - Hypothese – Vorschlag für eine Funktion, die mutmaßlich ähnlich zu *f* ist
  - Hypothesenraum – Menge aller möglichen Hypothesen, die das Lernsystem erlaubt
  - Trainingsbeispiele – Paare der Form <x, *f*(x)>
  - Klassifikation – Funktionen mit diskreten Werten, die Klassen bezeichnen
- Versionsraum-Algorithmen
  - Algorithmen: Find-S, List-Then-Eliminate, Kandidatenelimination
  - Konsistente Hypothesen – solche, die zu allen beobachteten Beispielen passen
  - Versionsraum – Menge aller aktuell konsistenten Hypothesen
- Induktives Lernen
  - Induktive Generalisierung – Verfahren, welches Hypothesen generiert, die auch Fälle bewerten, die noch nicht bekannt sind
- Annahmen beim induktiven Lernen
  - Keine widersprüchlichen Daten, kein Rauschen, keine Fehler
  - Es gibt ein *h* im Hypothesenraum, das *f* ausreichend annähhert

Zusammenfassung

- Begriffslernen ist eine Suche in H
  - Hypothesenraum H ist der Suchraum
  - Lernen entspricht Suchen und Finden der richtigen Hypothese
- Hypothesenraum kann geordnet werden (generell-spezifisch)
  - Die more-general-than Relation ist eine partielle Ordnung
  - H hat eine obere und untere Schranke
- Versionsraum und Kandidateneliminationsalgorithmus
  - S und G Schranken beschreiben die Unsicherheit der Lerners
  - Versionsraum kann genutzt werden, um Vorhersagen für neue Beispiele zu machen
- Verfahren kann genutzt werden, um Beispiele auszuwählen
- Jedes Beispiel muss nur einmal gezeigt werden
- Induktiver Bias
  - zusätzliche Annahmen über Struktur der Hypothesen erhöhen die Generalisierungsfähigkeit
  - Lernen mit unterschiedlich hohem Bias

Literatur

- Machine Learning, Tom Mitchell, McGraw Hill, 1997
- "Neural Networks - A Systematic Introduction", Raul Rojas, Springer-Verlag, Berlin, 1996.
- "Theorie der neuronalen Netze", Raul Rojas, Springer-Verlag, Berlin, 1993/99.
- ...

Weitere Lernverfahren

- Symbolisch
  - Entscheidungsbäume
  - Fallbasiertes Schließen (CBR)
  - ...
- Subsymbolisch
  - Neuronale Netze
  - Support Vector Machines
  - Genetische Algorithmen
  - …