

Maschinelles Lernen 1 - Instanzbasiertes Lernen

Lazy learning vs. Eager learning

- Instanzbasiertes Lernen ("Memory-based learning") vergleicht neue Probleminstanzen mit den Instanzen aus dem Training, die im Speicher gehalten werden
=> „lazy learning“
- Eager learning: Modellbildung findet einmalig auf Basis der Trainingsdaten statt (Offline) anstatt Online zur Anfragezeit (=lazy)
- Lernen ist einfacher Abspeichern der Instanzen
- Folge: schneller Training, hohe Testlaufzeit bei vielen Instanzen
 - jede Anfrage birgt unterschiedliche Hypothesen (lokale Approx. der Zielfunktion)
 - „Generalisierung“ wird bis zur Anfrage aufgeschoben

Beurteilung von Instanzbasiertem Lernen

- ⊕ Information aus Trainingsbeispielen geht nicht verloren
- ⊕ auf einfache Weise komplexe Zielfunktion modellieren
- ⊖ evtl. rechenintensiv, bei neuer Anfrage und vielen Instanzen
- ⊖ Definition von „Ähnlichkeit“ => welche Instanzen sind ähnlich?

k-Nearest-Neighbor

- Instanzen: $x = \langle a_1(x), a_2(x), \dots, a_n(x) \rangle$ korrespondieren mit Punkten im n-dimensionalen Raum \mathbb{R}^n
- Distanz von Nachbarn berechnet mit euklidischer Distanz:
$$d(x_i, x_j) = \sqrt{\sum_{v=1}^n (a_v(x_i) - a_v(x_j))^2}$$
- Lernen einer Funktion $f: \mathbb{R}^n \rightarrow V$ aus einer Menge von Trainingsbeispielen $\langle x_i, c(x_i) \rangle$ wobei $c(x)$ eine Zielkonzeptfunktion und V endliche (diskrete) Menge ist

Trainingsphase

- für jedes Trainingsbeispiel $\langle x_i, c(x_i) \rangle$ mit $c(x_i) \in V$: füge jedes Trainingsbeispiel zur Liste „training-list“ hinzu

Testphase (Anfrage: x_q)

- x_1, \dots, x_k seien die k Instanzen von training-list, die am nächsten zu x_q liegen
- $f(x_q) \leftarrow \arg \max_{v \in V} \sum_{i=1}^k \delta(v, c(x_i))$
mit $\delta(a, b) = \begin{cases} 1, & \text{falls } a = b \\ 0, & \text{sonst} \end{cases}$

→ kNN mit $k=1$ entspricht Voronoi-Diagramm

Induktiver Bias: Klassifikation einer Instanz x_q ähnlich zu Klassifikation anderer benachbarter Instanzen

kNN-Erweiterungen

(2)

- Normalisierung der Eingabevektoren
→ da Verzerrung bei stark ungleichen Eingabedimensionen
- Abstandsgewichtung
→ nahe Nachbarn haben evtl. viel größere Ähnlichkeit mit neuer Instanz aber weit entfernte Samples fließen genauso stark ein
→ Distanz-basierte Gewichtung:

$$f(x_q) \leftarrow \arg \max_{v \in V} \sum_{i=1}^n w_i S(v, c(x_i)) \text{ mit } w_i = \frac{1}{d(x_q, x_i)^2}$$

kNN-Bevorteilung

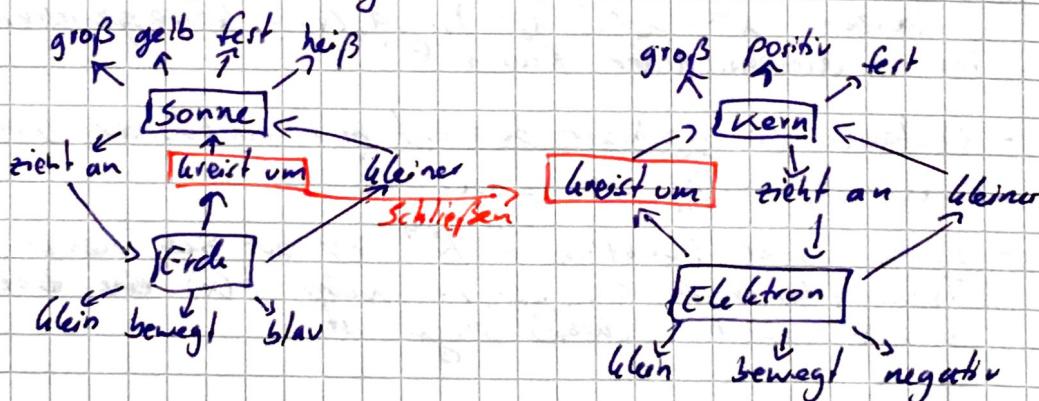
- ⊕ k ist Werbung, wenn Trainingsdaten voraussetzt (insbesondere mit Abstandsgewichtung)

- ⊖ „Curse of dimensionality“ → Distanzmaß basiert auf allen Attributen

- ⊖ Speicheraufwand
- ⊖ kurze Trainingszeit, lange Testzeit

Case-based Reasoning

- Analoges Schließen (induktiv/nicht deduktiv): Ähnlichkeit von Größen hinsichtlich mehrerer Eigenschaften (Relationen) erlaubt Schluss auf Ähnlichkeit weiterer Eigenschaften



„A case-based reasoner solves new problems by adapting solutions that were used to solve old problems“ (Schank)

Überblick

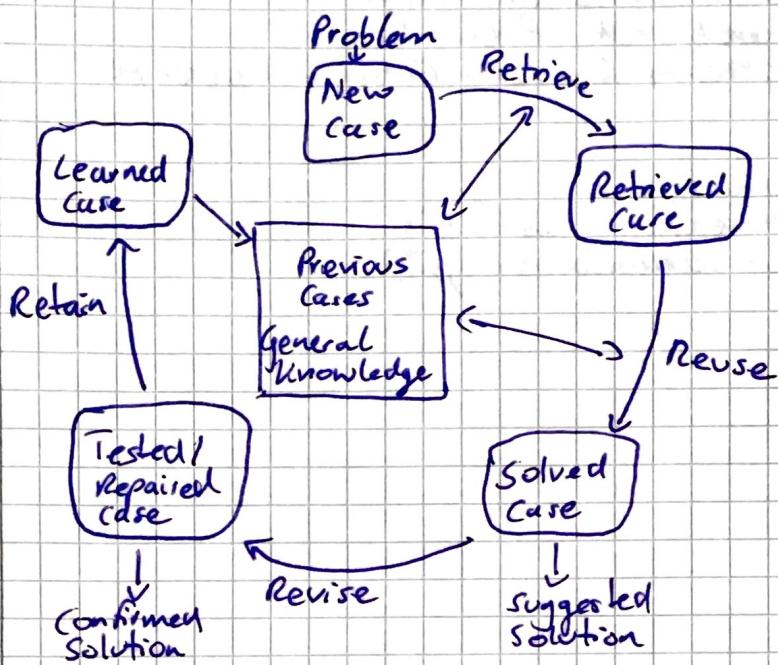
- kein direkt anwendbarer Algorithmus
- Wiederverwendung alter Fälle
- Suche nach Lösungen bei ähnlichen Problemen

Was ist ein Fall?

Ein Fall ist die Beschreibung einer bereits real aufgetretenen Problemsituation zusammen mit den Erfahrungen, die während der Bearbeitung des Problems gewonnen werden konnten

③ Maschinelles Lernen 1 - Instanzbasierter Lernen

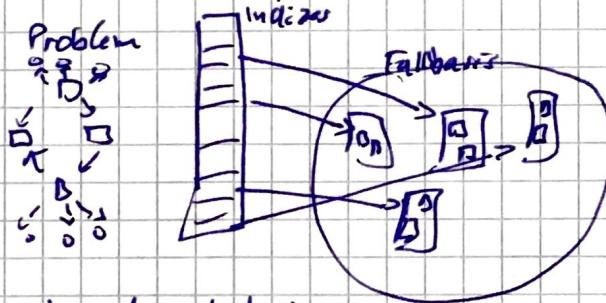
CBR-Zyklus



[Amold & Plaza, 1994]

Retrieve

- finden ähnlicher Fälle
- Ähnlichkeitsmaß z.B. eukl. Distanz oder syntaktische/semantische Ähnlichkeit
- Fallbasis kann Liste, Baum etc. sein



Probleme bei der Indizierung:

- Indizierung muss Umstände antizipieren, unter denen ein Fall wieder verwendet werden soll, z.B. „vegetarisches Hauptgericht“, „laktoseintolerant“ etc.

Reuse: z.B. einfacher übertragen (Copy) oder durch Benutzer (interaktiv)

Revise: Evaluierung / Verbessern (z.B. Fehler erkennen) der Lösung
(iterativ)

führt dies nicht dazu, dass bei Fallnachreihen schlechtere Ergebnisse erzielt werden

Retain: neue Erfahrung speichern, alten Fall generalisieren
(induktiv/deduktive Lernverfahren oder Auswendiglernen / einfaches Speichern des Falls)

Beispiele

- Lockheed (Flugzeugbau) -> „Clavier“
-> unterschiedliche Bauteile für Otter zusammensetzen
-> Suche nach „ähnlichster“ bekannte Zusammenstellung
-> Adaption durch Ersetzen möglichst weniger Elemente
- Kogni Mobil
-> Situationserkennung -> ähnlichster Fall aus Fahr/Verkehrssituationen -> Falldatenbank (hierarchisch) entnommen
-> Verhalten von diesem Fall ableiten

CBR Beurteilung

- ⊕ es genügen bereits wenige Informationen
- ⊕ Analogie zu menschlichen Problemlösen
- ⊕ einfach und trotzdem komplexe Entscheidungsgruppen möglich
- ⊖ hohe Speicher Kosten (ggf.)
- ⊖ stark von Repräsentation abhängig
- ⊖ hohe Laufzeit für Klassifikation (ggf.)