
Data Mining und Maschinelles Lernen

Lösungsvorschlag für das 2. Übungsblatt



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Aufgabe 1: Konzepte, Regeln und Hypothesenräume

Gegeben sei ein Beispielraum, der durch n binäre Attribute aufgespannt wird. Hypothesenraum sind alle möglichen Regeln, die sich durch Konjunktionen von Tests der Form *Attribut = Wert* ergeben.

a) Wie viele mögliche Konzepte gibt es?

Lösung: Ein Konzept ist definiert als eine Untermenge aller möglichen Objekte (Folie 2). Bei bool'schen Attributen sind die Objekte boolesche Vektoren. Gehen wir die ersten zwei Fälle $n = 1$ und $n = 2$ durch.

Fall $n = 1$

Es gibt genau ein bool'sches Attribut A_1 , also zwei mögliche Objekte.

- ▶ 1 = im Konzept C_i enthalten
- ▶ 0 = nicht enthalten.

Wert von A_1	In C_1 ?	C_2	C_3	C_4
false	0	1	0	1
true	1	0	0	1

$n = 1 \rightarrow 4$ mögliche Konzepte

Fall $n = 2$

Es gibt genau zwei bool'sche Attribut A_1 und A_2 , also $2^2 = 4$ mögliche Objekte.

A_1	A_2	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}
false	false	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
false	true	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
true	false	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
true	true	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$n = 2 \rightarrow 16$ mögliche Konzepte C_i

Hat man n binäre Attribute, so ergeben sich 2^n verschiedene mögliche Objekte.

Da jedes Objekt in der Untermenge enthalten oder nicht enthalten sein kann erhält man 2^{2^n} mögliche Konzepte.

b) Wie viele mögliche Regeln gibt es?

Lösung: Ein Attribut kann in einer Regel

- ▶ den Wert „false“ oder
- ▶ den Wert „true“ annehmen oder
- ▶ in der Regel überhaupt nicht vorkommen!

⇒ $3^n + 1$ mögliche Regeln.

Dazu kommt noch die Regel mit dem Body "false" (deswegen +1).

In der Tupel-Notation aus der Vorlesung gibt es für jede Stelle die Möglichkeiten *true*, *false* und *?*, sowie die Theorie, die an allen Stellen \emptyset ist.

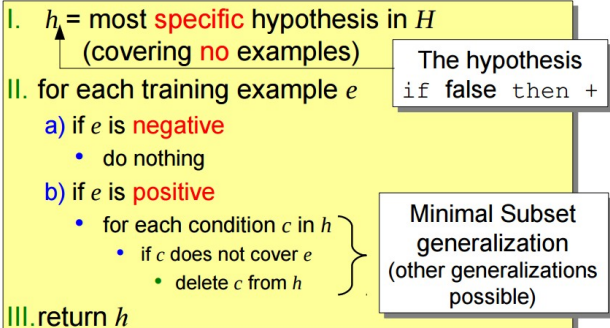
Diese Lösung setzt allerdings voraus, dass man nur "sinnvolle" Regeln zulässt. Wären auch beide Attributwerte innerhalb einer Regel möglich (*Attribut₁ = false* AND *Attribut₁ = true*), so würde es mehr mögliche Regeln geben.

→ Es sollte klar definiert sein, was zulässig ist!



c) Unter der Voraussetzung, daß das zu lernende Konzept im Hypothesenraum darstellbar ist, wie viele Fehler kann der Algorithmus Find-S beim Lernen dieses Konzepts maximal machen?

(Anm: Ein Fehler ist hier ein (Trainings-)Beispiel, das von der momentanen Theorie falsch klassifiziert wird.)



Note: when the first positive example is encountered, step II.b) amounts to converting the example into a rule (The most specific hypothesis can be written as a conjunction of all possible values of each attribute.)



Lösung: Wenn das zu lernende Konzept im Hypothesenraum enthalten ist, gibt es also eine Hypothese, die konsistent (deckt also keine negativen Beispiele ab) und komplett (deckt also alle positiven Beispiele ab) ist.

Der Algorithmus FINDS beginnt mit der **speziellsten Hypothese** (der leeren Menge {}). Auf dem Weg zum Zielkonzept können **keine** negativen Beispiele abgedeckt werden.

Da der Algorithmus seine Hypothese nur bei einem positiven Beispiel verändert, kann er während des Lernprozesses **maximal so viele Fehler machen, wie es positive Beispiele gibt.**



Mehr positive Beispiele wie Attribute:

- ▶ Unterscheiden sich nun die Attributwerte der positiven Beispiele immer **genau an einer Stelle**
- ▶ und gibt es mehr positive Beispiele als Attribute in der Trainingsmenge

dann macht der Algorithmus **nur so viele Fehler, wie es Attribute gibt**, da er nach der letzten Änderung bereits die generellste Hypothese gefunden hat. Da er beim ersten Beispiel immer falsch liegt, *kommt noch ein Fehler* hinzu.

Mögliche Fehler:

$$\Rightarrow \min\{|Attribute| + 1, |positiveBeispiele|\}$$

Beispiel 1

Fall 1: Der maximale Fehler entspricht der Anzahl der positiven Beispiele

A_1	A_2	A_3	A_4	Klasse	Hypothese	Fehler?
-	-	-	-	-	{}	-
false	true	true	false	+	(false,true,true,false)	ja
true	false	true	false	+	(?,?,true,false)	ja
false	false	false	true	+	(?,?,?,?)	ja

$$\min\{|Attribute| + 1, |positiveBeispiele|\} = \min\{5, 3\} = 3$$

Der Algorithmus macht hier auch 3 Fehler!

Beispiel 2

Fall 2: Der maximale Fehler entspricht der Anzahl der Attribute +1

A_1	A_2	A_3	Klasse	Hypothese	Fehler?
-	-	-	-	{}	-
false	false	false	+	(false,false,false)	ja
true	false	false	+	(?,false,false)	ja
true	true	false	+	(?,?,false)	ja
false	false	true	+	(?,?,?)	ja
true	true	true	+	(?,?,?)	nein
false	true	true	+	(?,?,?)	nein

$$\min\{|Attribute| + 1, |positiveBeispiele|\} = \min\{4, 6\} = 4$$



d) Gegeben sei ein Hypothesenraum, der nur *Disjunktionen* von binären Attribut-Wert-Paaren erlaubt. Eine gültige Regel wäre also z.B.

if ($att_i = t$) **or** ($att_j = f$) **or** ($att_k = f$) **then** +

Überlegen Sie sich Verallgemeinerungs- und Spezialisierungsvorschriften für diesen Hypothesenraum und geben Sie einen geeigneten Lernalgorithmus an.

Lösung:

- ▶ Als Verallgemeinerungsvorschrift fügt man Bedingungen hinzu
- ▶ Als Spezialisierungsvorschrift entfernt man Bedingungen.

Lernalgorithmen:

Die Rollen der beiden Algorithmen FindS und FindG vertauschen sich damit. Da FINDG Bedingungen hinzufügt (spezialisiert), sollte man zum Lernen den Algorithmus FINDS verwenden. Da die Rollen aber in diesem Hypothesenraum vertauscht sind, ist der geeignete Lernalgorithmus FINDG.

Aufgabe 2: Find-S und Find-GSet



Sie wollen den Find-S Algorithmus auf numerische Daten anwenden, indem Sie Intervalle definieren.

a) Wie sieht das spezifischste und generellste Element der Sprache aus, wenn Sie offene Intervalle verwenden?

Lösung:

- ▶ Generellstes Element: $(-\infty, \infty)$
- ▶ Spezifischstes Element: $(x, x) = \{x\}$ mit $x \in \mathbb{R}$

Aufgabe 2: Find-S und Find-GSet



TECHNISCHE
UNIVERSITÄT
DARMSTADT

b) Wie sieht das spezifischste und generellste Element der Sprache aus, wenn Sie geschlossene Intervalle verwenden?

Lösung:

Es gibt **kein** eindeutiges spezifischstes und auch **kein** generellstes Element.
→ Hier muss man sich virtuelle Elemente vorstellen, die einerseits der leeren Menge entsprechen und andererseits alle möglichen Werte abdecken.

Aufgabe 2: Find-S und Find-GSet

c) Finden Sie eine passende Generalisierungsvorschrift und simulieren den Algorithmus Find-S auf folgenden Beispielen:

A1	Klasse
0.5	—
1.0	+
2.1	—
0.8	—
1.5	+
1.8	+

Aufgabe 2: Find-S und Find-GSet



Lösung: Als Intervall verwenden wir **geschlossene Intervalle**.

Als Generalisierungsvorschrift verwenden wir die **Aktualisierung (Erweiterung)** einer der beiden Intervallgrenzen (abhängig vom Wert des Beispiels).

A1	Klasse	Hyp $S_0 = \{ \}$
0.5	—	$\{ \}$ (IIa)
1.0	+	$[1.0, 1.0]$ (IIb, siehe Note!)
2.1	—	$[1.0, 1.0]$ (IIa)
0.8	—	$[1.0, 1.0]$ (IIa)
1.5	+	$[1.0, 1.5]$ (IIb)
1.8	+	$[1.0, 1.8]$ (IIb)

Zusatzfrage: Wieso wurden keine offenen Intervalle genommen?

- I. h = most **general** hypothesis in H (covering **all** examples)
- II. $G = \{ h \}$
- III. for each training example e
 - a) if e is **positive**
 - remove all $h \in G$ that do not cover e
 - b) if e is **negative**
 - for all hypotheses $h \in G$ that cover e
 - $G = G \setminus \{h\}$
 - for **every** condition c in e that is not part of h
 - for all conditions c' that negate c
 - $h' = h \cup \{c'\}$
 - if h' covers all previous positive examples
 - $G = G \cup \{h'\}$
- IV. return G

d) Finden Sie eine passende Spezialisierungsvorschrift und simulieren Sie den Algorithmus Find-GSet auf denselben Beispielen.

Kann der Intervalltyp aus vorheriger Aufgabe genutzt werden, warum?

→ Als Intervall verwenden wir **offene Intervalle**. Als Spezialisierungsvorschrift verwenden wir eine **Aufspaltung des aktuellen Intervalls** in ein linkes und ein rechtes Intervall.

A1	Klasse	Hyp $G_0 = (-\infty, \infty)$	(generellstes Intervall)
0.5	-	$G_1 = (-\infty; 0.5), (0.5; \infty)$	spalte das Intervall auf
1.0	+	$G_2 = (0.5, \infty)$	entferne $(-\infty, 0.5)$, da das positive Beispiel nicht abgedeckt ist ($1.0 \notin (-\infty, 0.5)$)
2.1	-	$G_3 = (0.5, 2.1)$	1. aufspalten: $(0.5, 2.1), (2.1, \infty)$ 2. nur die hinzufügen, die alle vorigen positiven Beispiele abdecken (letzter Schritt des Algorithmus), da $1.0 \notin (2.1, \infty)$ nur $(0.5, 2.1)$
0.8	-	$G_4 = (0.8, 2.1)$	1. aufspalten: $(0.5, 0.8), (0.8, 2.1)$ 2. $(0.5, 0.8)$ entfernen, da $1.0 \notin (0.5, 0.8)$
1.5	+	nichts passiert	
1.8	+	nichts passiert	

Aufgabe 2: Find-S und Find-GSet

e) Skizzieren Sie beide Lösungen und vergleichen Sie die Allgemeinheit.



Lösung: Das Ergebnis von FINDS ist die speziellste vollständige und konsistente Theorie (die Theorie, die gerade noch kein positives Beispiel ausschließt). In der Grafik ist diese durch das rot markierte, geschlossene Intervall gekennzeichnet.

Das Ergebnis von FINDGSET ist die allgemeinste vollständige und konsistente Theorie (die Theorie, die gerade noch kein negatives Beispiel einschließt), welche in der Grafik durch das blau markierte, offene Intervall repräsentiert ist.