

Klassifikation I

Praktikum Data Warehousing und Data Mining

Klassifikationsprobleme

- Idee
 - Bestimmung eines unbekanntes *kategorischen* Attributwertes (*ordinal* mit Einschränkung)
 - Unter Benutzung beliebiger bekannter Attributwerte
- Beispiele:
 - Klassifikation von Spam
 - Vorhersage von Kundenverhalten wie Kündigungen
 - Medizinische Diagnosen
 - Vorhersage von Kreditwürdigkeit
 - Beurteilung von industriellen Gütern
 - ...

Aufgabenstellung Data-Mining-Cup 2010

Aufgabenstellung

- Cross-Selling bzw. Up-Selling für einen Online-Händler
- Konkret:
 - Gewinnmaximierung durch Intelligentes Couponing
 - Welche Neukunden sollen einen 5 EUR Gutschein bekommen?
 - Kunden, die sowieso wieder kommen würden, kosten unnötig Geld.
- Data-Mining-Problem:
 - Binäre Entscheidung (Klassifikation), welche Kunden einen Gutschein bekommen sollen.
- Gegebene Daten:
 - Standard-Kundendaten
 - Kauf in den nächsten drei Monaten?



Klassifikation I – Wie baue ich gute Klassifikatoren im Allgemeinen?

Zunächst: Ein sehr einfacher
Klassifikator...

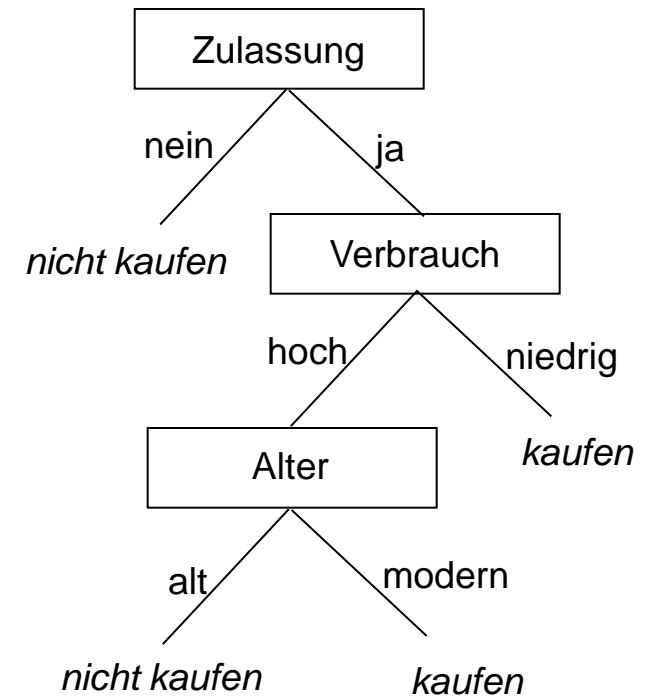
k -Nearest Neighbour

- Gegeben:
 - Lerndatensatz L
- Vorgehen zur Klassifikation eines Tupels t :
 - Menge S : k nächsten Nachbarn von t in L
 - Klassifikation von t durch Klasse mit meisten Elementen in S
- Anmerkungen:
 - „Nähe“ über Distanzmaß (z.B. euklidische Distanz)
 - Kein Lernen im engeren Sinn
 - Klassifikation rechenaufwändig für große L
 - Ggf. Einsatz einer (repräsentativen) Stichprobe von L
 - Einsatz sinnvoll bei wenigen, numerischen Attributen
 - Ggf. auch bei DMC-Aufgabe geeignet, aber...

Entscheidungsbäume

Klassifikation - Entscheidungsbäume

- Vorgehen
 - Aufbau eines Baums
 - Knoten entspricht Entscheidungskriterium
 - Blatt entspricht Entscheidung
- Vorteile
 - Ergebnis leicht interpretierbar
 - Übersetzbar in Regelsystem
- Diverse Verfahren
 - ID3 / C4.5 / C5.0 / J48
 - C&R Tree
 - Quest
 - Chaid
 - ...



Eine Regel (von mehreren):
Wenn
Zulassung vorhanden und
Verbrauch niedrig,
dann kaufen.

Vorgehen bei der Konstruktion

- 01 Alle Datentupel im Wurzelknoten
- 02 IF Stoppkriterium erreicht THEN
- 03 Aktueller Knoten wird Blattknoten mit Majoritätsklasse
- 04 ELSE
- 05 Suche geeignetes Entscheidungsattribut (Splitattribut)
- 06 Wende Algorithmus rekursiv auf Teilmengen an

Anschließend: Beschneide Baum (Pruning)

- Existierende Algorithmen unterscheiden sich in
 - ... der Wahl des Stoppkriteriums
 - ... der Art des Splits
 - ... dem Vorgehen zur Wahl des Splitattributs
 - ... der Wahl des Pruningverfahrens

Wahl des Stoppkriteriums

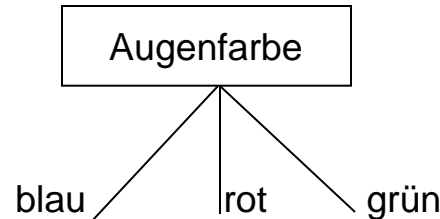
- Natürliche Stoppkriterien
 - Knoten enthält (fast) nur Tupel einer Klasse
 - Alle Klassifikationsattribute ausgeschöpft
- Weitere Kriterien
 - Minimale Tupelzahl je Knoten
 - Minimaler Anteil falsch klassifizierter Tupel
 - Maximale Baumtiefe
 - Maximale Knotenanzahl

Art des Splits

- Diskrete vs. kontinuierliche Attribute

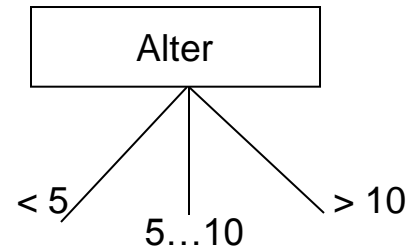
- Diskret

- Ein Knoten pro
Attributwert



- Kontinuierlich:

- Ein Knoten pro
Attributintervall



- Binäre vs. n-äre Bäume

- Zwei oder mehrer Ausgangskanten

- Frage: Wie erreicht man binäre Bäume mit diskreten Attributen?

Wahl der Splitattribute

- Beispiel
 - Objektmenge: 1 0 0 0 0 0 1 0 0 1 0 1
 - Verschiedene Splits möglich:
 - Split A: 1 0 0 0 0 0 | 1 0 0 1 0 1
 - „Linke Seite“: 17% Fehler
 - „Rechte Seite“: 50% Fehler
 - Split B: 1 0 0 0 0 0 1 0 0 | 1 0 1
 - „Linke Seite“: 22% Fehler
 - „Rechte Seite“: 33% Fehler
 - Split C ...
- Welcher Split ist vorzuziehen?
 - Festlegung eines Fehlermaßes

Informationsgewinn

- Prinzip
 - Maximierung des Informationsgewinns
- Vorgehen
 - Basiert auf Shannon-Entropie
 - $H = - \sum_{i=1}^n p_i \log_2 p_i$
 - Informationsgewinn
 - $I_{\text{gain}}(C,A) = H(C) - H(C|A)$
 - $I_{\text{gain}}(C,A) = - \sum_{i=1}^{|C|} p_i \log_2 p_i - \sum_{j=1}^{|A|} p_j (- \sum_{i=1}^{|C|} p_{ij} \log_2 p_{ij})$
 - $H(C)$ – Entropie der Klassenverteilung (mit C – Klassenattribut)
 - $H(C|A)$ – Erwartete Entropie der Klassenverteilung, wenn Attribut A bekannt

Informationsgewinn - Beispiel

- Berechnung des Informationsgewinns
 - Allgemein:

$$I_{\text{gain}}(C,A) = - \sum_{i=1}^{|C|} p_i \log_2 p_i - \sum_{j=1}^{|A|} p_j (- \sum_{i=1}^{|C|} p_{ij} \log_2 p_{ij})$$
 - $H(C) = - (4/12 * \log_2(4/12) + 8/12 * \log_2(8/12)) = 0,918$
 - Split A:
 - 1 0 0 0 0 0 | 1 0 0 1 0 1
 - $I_{\text{gain}}(C,A) = 0,918 - (6/12 * (-1) * (1/6 * \log_2(1/6) + 5/6 * \log_2(5/6)) + 6/12 * (-1) * (3/6 * \log_2(3/6) + 3/6 * \log_2(3/6)))$

$$= 0,918 - 0,325 - 0,500$$

$$= 0,918 - 0,825 = 0,093$$
 - Split B:
 - 1 0 0 0 0 0 1 0 0 | 1 0 1
 - $I_{\text{gain}}(C,B) = 0,918 - (9/12 * (-1) * (2/9 * \log_2(2/9) + 7/9 * \log_2(7/9)) + 3/12 * (-1) * (1/3 * \log_2(1/3) + 2/3 * \log_2(2/3)))$

$$= 0,918 - 0,573 - 0,230$$

$$= 0,918 - 0,803 = 0,115$$
- Hier würde B bevorzugt

Gini Index

- Prinzip
 - Minimierung der Heterogenität
- Vorgehen
 - Wahrscheinlichkeitsmaß, bei Stichprobe Datentupel aus 2 unterschiedlichen Klassen zu erhalten:
 - $Gini = 1 - p(0)^2 - p(1)^2$
 - Minimum = 0,0
 - alle Objekte aus einer Klasse
 - Maximale Homogenität
 - Maximum = 0,5
 - Objekte zweier Klassen gleich häufig
 - Maximale Heterogenität

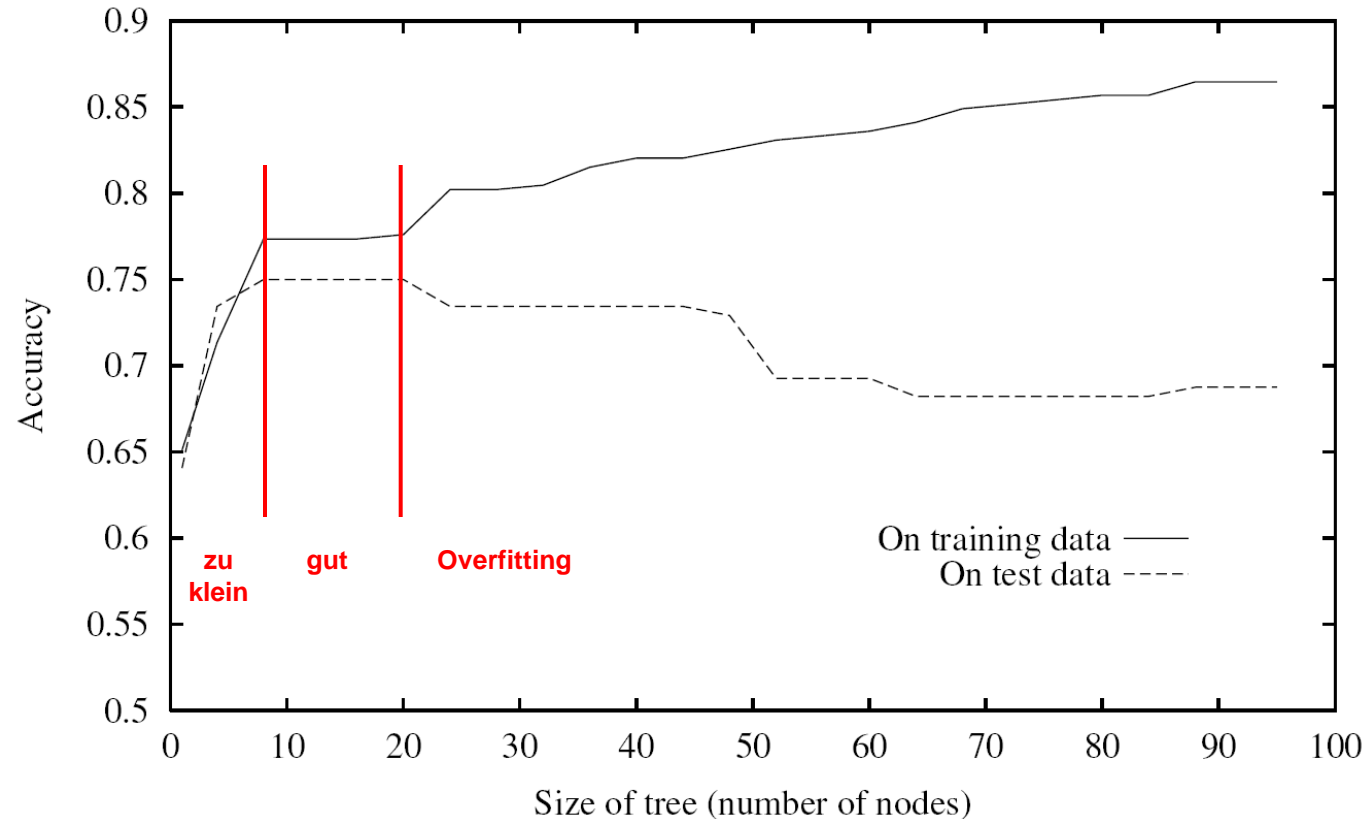
Gini Index - Beispiel

- Berechnung der Heterogenität
 - Split A:
 - 1 0 0 0 0 0 | 1 0 0 1 0 1
 - Linke Seite = $1 - (1/6)^2 - (5/6)^2 = 0,278$
 - Rechte Seite = $1 - (3/6)^2 - (3/6)^2 = 0,500$
 - Split B:
 - 1 0 0 0 0 0 1 0 0 | 1 0 1
 - Linke Seite = $1 - (2/9)^2 - (7/9)^2 = 0,346$
 - Rechte Seite = $1 - (1/3)^2 - (2/3)^2 = 0,444$
 - Einfacher Durchschnitt
 - A: $(0,278 + 0,500) / 2 = 0,389$
 - B: $(0,346 + 0,444) / 2 = 0,395$
- Hier würde A bevorzugt

Weitere Fehlermaße

- Chi-Quadrat-Test
 - Maß für die Abhängigkeit zwischen Merkmal und Zielgröße
 - Auswahl des Merkmals mit dem höchsten Chi-Quadrat-Wert (= stärkste Abhängigkeit)
- Minimale Beschreibungslänge (MDL)
 - Ähnlich Informationsgewinn
 - Zusätzlich „Strafe“ für zunehmende Komplexität des Baums

Pruning - Motivation



- **Achtung: Die optimale Baumgröße ist bei jedem Datensatz unterschiedlich!**

Pruningverfahren

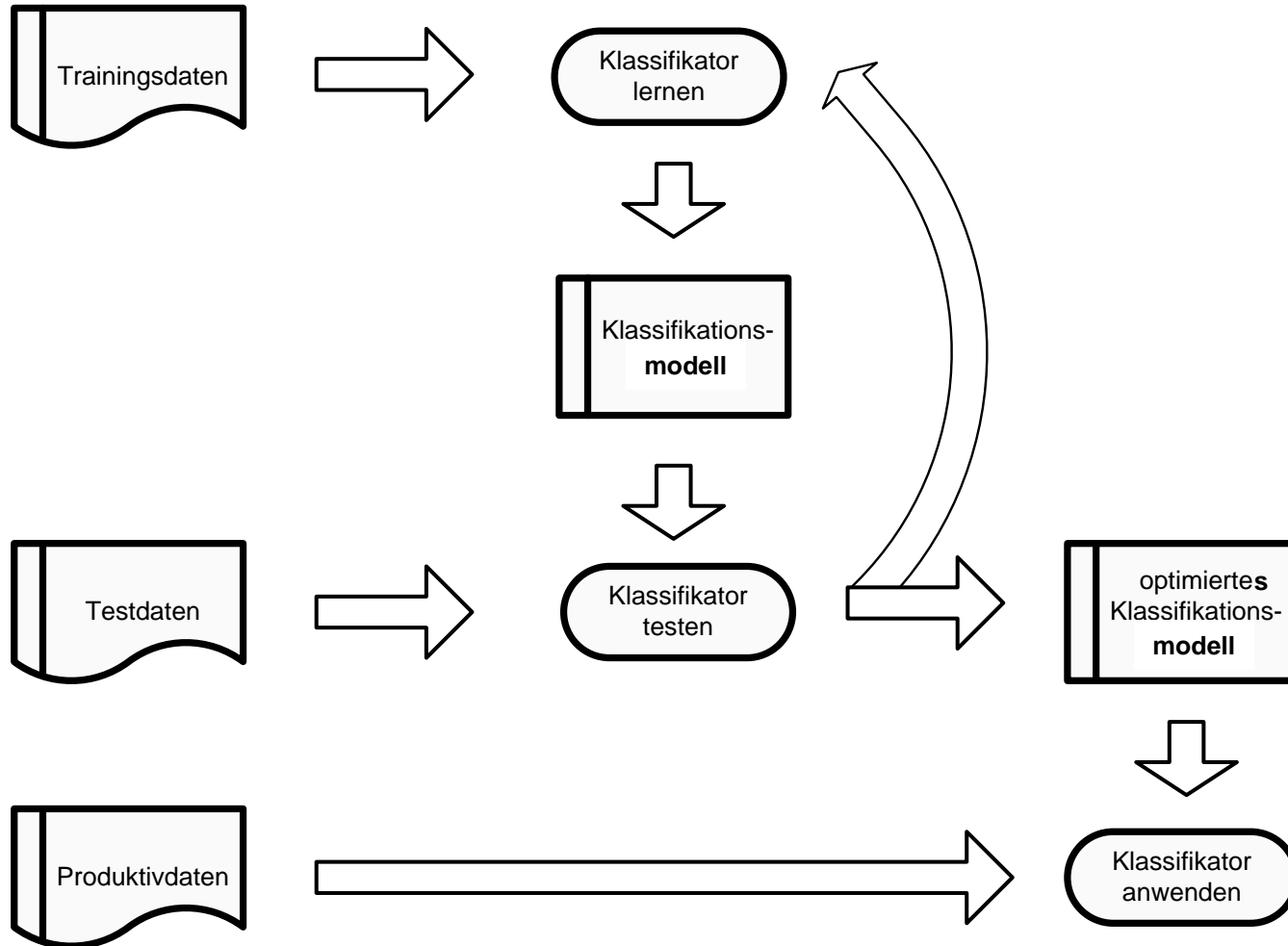
- Gründe für Pruning komplexer Bäume
 - Einfachheit / Verständlichkeit
 - Verhinderung von Overfitting / Generalisierungsfähigkeit
- Pre-Pruning: Stopkriterien bei Baumerstellung
- Post-Pruning: Nachträgliches Stutzen
 - Subtree Replacement
 - Ersetzen von Entscheidungsknoten durch Blattknoten wenn Klassifikationsbeitrag gering
 - Optimal: Entscheidung zum Ersetzen mit „frischen“ Daten evaluieren
 - Subtree Raising
 - Verschiebung von Teilbäumen nach oben
 - Insbesondere dann, wenn es Attribute gibt, die einzeln wenig, aber in Kombination sehr stark zur Klassifikation beitragen.
 - Solche Attribute rutschen sonst sehr leicht weit nach unten.

Evaluation von Klassifikatoren

Sampling bzw. Holdout

- Die Leistung eines Klassifikators kann nicht mit dem Lerndatensatz beurteilt werden!
 - Overfitting! Vgl. Motivation Pruning.
- Deshalb: Unterteilung der Ausgangsdaten in
 - Training Set zum Lernen des Klassifikators (oft zwei Drittel)
 - Test Set zur Evaluation des Klassifikators (oft ein Drittel)
- Beide Mengen sollten möglichst repräsentativ sein:
 - **Stratifikation:** Aus jeder Klasse wird ein proportionaler Anteil in das Training- und Test Set übernommen.
- Eine Unterteilung in Training- und Test Set ist oft nicht möglich, wenn nicht genug Daten zur Verfügung stehen:
 - Ein kleines Test Set ist ggf. nicht mehr repräsentativ.
 - Ein kleines Training Set bietet ggf. zu wenig zum Lernen.

Klassifikation - Vorgehen



Cross-Validation

- Unterteilung der Ausgangsdaten in k Partitionen
 - Typischerweise wird $k=10$ gewählt
 - Eine Partition bildet Test Set
 - $k-1$ Partitionen bilden Training Set
- Berechnung und Evaluation von k Klassifikatoren:
 - In k Runden wird jedes Datentupel $k-1$ mal zum lernen verwendet und genau ein mal klassifiziert.
- Stratifizierte Cross-Validation ist in vielen Fällen die zu empfehlende Evaluationstechnik, besonders aber bei kleinen Datensätzen.
- Achtung: Cross-Validation ist sehr Rechenaufwändig
- „Leave-One-Out“ ist Spezialfall für $k=n$

Evaluationsmasse für Klassifikatoren

- Konfusions-Matrix

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	True Positives (TP)	False Negatives (FN)
	Nein	False Positives (FP)	True Negatives (TN)

- $accuracy = (TP + TN) / (TP + FN + FP + TN)$
- $sensitivity = TP / (TP + FN)$
- $specificity = TN / (FP + TN)$
- $precision = TP / (TP + FP)$
- $lift = precision / P(ja); P(ja) = (TP + FN) / (TP + FN + FP + TN)$

Evaluationsmasse – Beispiel 1 („Traum“)

- Konfusions-Matrix

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	490 (TP)	10 (FN)
	Nein	10 (FP)	490 (TN)

- accuracy = 0,98
- sensitivity = 0,98
- specificity = 0,98
- precision = 0,98
- lift = 1,96; P(ja) = 0,50

Evaluationsmasse – Beispiel 2 („schlecht“)

- Konfusions-Matrix

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	10 (TP)	90 (FN)
	Nein	95 (FP)	805 (TN)

- accuracy = 0,82
- sensitivity = 0,10
- specificity = 0,89
- precision = 0,10
- lift = 0,95; $P(\text{ja}) = 0,10$

Evaluationsmasse – Beispiel 2a („besser“)

- Konfusions-Matrix

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	0 (TP)	100 (FN)
	Nein	0 (FP)	900 (TN)

- accuracy = 0,90 (besser!)
- sensitivity = 0,00 (schlechter)
- specificity = 1,00 (besser!)
- precision = undef.
- lift = undef. bzw. „1,00“ (besser!); $P(\text{ja}) = 0,10$

Evaluationsmasse – Beispiel 3 („brauchbar“)

- Konfusions-Matrix

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	259 (TP)	1,077 (FN)
	Nein	578 (FP)	21,664 (TN)

- accuracy = 0,93
- sensitivity = 0,19
- specificity = 0,97
- precision = 0,31
- lift = 5,46; $P(\text{ja}) = 0,06$

Kostenbasierte Fehlermaße

- „Falsch ist nicht gleich falsch!“
- Kosten-Matrix wenn keine Kosten vorgegeben:

		Vorhersage	
		Ja	Nein
Tatsächliche Klasse	Ja	0 (TP)	1 (FN)
	Nein	1 (FP)	0 (TN)

- Die Einträge in der Matrix beschreiben die Kosten die bei FP und FN entstehen.
- Beispiel Geldscheinprüfer:
 - FP=50,00 EUR; FN=0,01 EUR

Evaluationsverfahren beim DMC 2010

Assessing the results

If a customer is sent a voucher but would have repurchased anyway, this results in a € 5.00 loss in revenue.

In contrast if a customer who would not have repurchased is sent a voucher this increases revenue on average by € 1.50.

This results in the following cost matrix:

		Real	
		Non-repurchasers (0)	Repurchasers (1)
Forecast	No voucher (0)	0	0
	Voucher (1)	1.5	-5

The number of points is therefore calculated as follows:

i ... Customer number

g_i ... Voucher decision

k_i ... Purchase

$$x_i = \begin{cases} 0, & g_i = 0 \\ -5, & g_i = 1 \wedge k_i = 1 \\ 1.5, & g_i = 1 \wedge k_i = 0 \end{cases}$$

$$\text{Number of points} = \sum_{i \in \text{customer numbers}} x_i$$

Bayes Klassifikator

Bayes Klassifikation - Idee

- Gegeben sei
 - Ereignis X
 - Hypothese H : Data Tupel X gehört zu Klasse C
- Ziel der Klassifikation
 - Maximiere $p(H | X)$
 - $p(H | X)$: Bedingte Wahrscheinlichkeit, dass H stimmt, gegeben Tupel X
 - Problem:
 - $P(H | X)$ lässt sich nicht aus Daten bestimmen
- Bayes Theorem
 - $$p(H | X) = \frac{p(X | H) p(H)}{p(X)}$$
 - Vorteil: $p(X)$, $p(H)$ und $p(X | H)$ lassen sich hier bestimmen

Naive Bayes Klassifikator I

- Geben sei
 - Trainingsmenge D
 - Attribute A_i mit $i = \{1, 2, \dots, n\}$
 - m Klassen C_j mit $j = \{1, 2, \dots, m\}$
- Tupel gehört zu C_i , wenn $p(C_i | X) > p(C_j | X)$ für $1 \leq j \leq m, j \neq i$
- Ziel also
 - Maximierung von $p(C_i | X) = \frac{p(X | C_i) p(C_i)}{p(X)}$
 - $p(X)$ ist konstant für alle Klassen, also
 - Maximierung von $p(X | C_i) p(C_i)$

Naive Bayes Klassifikator II

- Vereinfachungen für die Berechnung
 - Bestimmung von $p(C_i)$
 - Abschätzung:
 - $p(C_i) = |C_{i,D}| / |D|$
 - $|C_{i,D}|$ ist Anzahl der Trainingstupel von Klasse C_i in D
 - Bestimmung von $p(X | C_i)$:
 - Unabhängigkeit der Klassen angenommen, dann gilt:
 $p(X | C_i) = \prod_{k=1}^n p(x_k | C_i)$
 - $p(x_k | C_i) = z / |C_{i,D}|$
 - z ist die Anzahl der Tupel in Klasse C_i mit Attributwert x_k
 - $|C_{i,D}|$ ist die Anzahl der Trainingstupel von Klasse C_i in D
- Klassifikation
 - Klasse von X bestimmt durch Berechnung von $p(X | C_i)$
 $p(C_i)$ für alle Klassen

Naiver Bayes Klassifikator - Beispiel

Gesucht: Klassifikation für:

$X = \{\text{jung, mittel, ja, schlecht}\}$

$p(\text{ja}) = 9/14$

$p(\text{nein}) = 5/14$

$p(\text{jung} | \text{ja}) = 2/9$

$p(\text{jung} | \text{nein}) = 3/5$

$p(\text{mittel} | \text{ja}) = 4/9$

$p(\text{mittel} | \text{nein}) = 2/5$

$p(\text{ja} | \text{ja}) = 6/9$

$p(\text{ja} | \text{nein}) = 1/5$

$p(\text{schlecht} | \text{ja}) = 6/9$

$p(\text{schlecht} | \text{nein}) = 2/5$

$P(X | C_1) = 2/9 * 4/9 * 6/9 * 6/9 * 9/14 =$
 $0,0282$

$P(X | C_2) = 3/5 * 2/5 * 1/5 * 2/5 * 5/14 =$
 $0,0069$

Vorhersage: ja (kauft PC)

Alter	Einkommen	Student	Kreditwürdigkeit	Klasse: Kauft PC
Jung	Hoch	Nein	Schlecht	Nein
Jung	Hoch	Nein	Gut	Nein
Mittelalt	Hoch	Nein	Schlecht	Ja
Senior	Mittel	Nein	Schlecht	Ja
Senior	Niedrig	Ja	Schlecht	Ja
Senior	Niedrig	Ja	Gut	Nein
Mittelalt	Niedrig	Ja	Gut	Ja
Jung	Mittel	Nein	Schlecht	Nein
Jung	Niedrig	Ja	Schlecht	Ja
Senior	Mittel	Ja	Schlecht	Ja
Jung	Mittel	Ja	Gut	Ja
Mittelalt	Mittel	Nein	Gut	Ja
Mittelalt	Hoch	Ja	Schlecht	Ja
Senior	Mittel	Nein	Gut	Nein

Weitere Klassifikationstechniken

Regelbasierte Klassifikatoren

- Klassifikation durch Regelsatz
 - Beispiel:
 1. $\text{petalwidth} \leq 0.6$: Iris-setosa
 2. $\text{petalwidth} \leq 1.7$ AND $\text{petallength} \leq 4.9$: Iris-versicolor
 3. Sonst: Iris-virginica
- Übliches Vorgehen:
 - Entscheidungsbaum lernen
 - Deduktion der wichtigsten Regeln aus Baum
 - Nicht alle Tupel klassifiziert:
 - Default-Regel klassifiziert einige Tupel
 - Im Beispiel: Default-Regel: Iris-virginica
- Regelsätze oft einfacher als Entscheidungsbäume
⇒ Generalisierung

Assoziationsregeln zur Klassifikation - Beispiel

- Gegeben:
Folgende Assoziationsregeln
 - Saft -> Cola; conf: 80%
 - Cola -> Saft; conf: 100%
 - Cola -> Bier; conf: 75%
 - Bier -> Cola; conf: 100%
- Vorhersageattribut:
 - Kauft Kunde Cola?
- Beispieletupel:
 - Kunde kauft Bier
⇒ Kunde kauft Cola (4. Regel)

Assoziationsregeln zur Klassifikation -Vorgehen

- Eine Regel passt:
⇒ Klassifikation eindeutig (mit Konfidenz der Regel)
- Keine Regel passt:
⇒ Mehrheits-Klasse bzw. unklassifiziert
- Mehrere Regeln passen:
 - Berücksichtigung der Regel mit höchster Konfidenz
 - Regel entscheidet
 - Berücksichtigung der k Regeln mit höchster Konfidenz (oder auch aller Regeln)
 - Häufigste auftretende Klasse
 - Klasse mit höchster durchschnittlicher Konfidenz der Regeln
 - ...
- Hinweis:
Verfahren eignet sich auch für sequentielle Regeln.

Quellen

- J. Han und M. Kamber: „Data Mining: Concepts and Techniques“, Morgan Kaufmann, 2006.
- T. M. Mitchell: „Machine Learning“, Mc Graw Hill, 1997
- C. Borgelt: Folien zur Vorlesung „Intelligent Data Analysis“, 2004.
- F. Klawonn: Folien zur Vorlesung „Data Mining“, 2006.
- M. Spiliopoulou: Vorlesung „Data Mining for Business Applications“, 2003.