Seminar Business Intelligence Teil II: Data-Mining und Knowledge-Discovery

Thema 3: Klassifikation und Prädiktion

Vortrag von Philipp Breitbach

Übersicht

1. Motivation
2. Grundlagen
3. Entscheidungsbauminduktion
4. Bayes‘sche Klassifikation
5. Regression
6. Zusammenfassung und Ausblick

1. Motivation

Data-Mining: Auffinden von bisher unbekannten Beziehungen innerhalb einer Datenmenge

Klassifikation: Einteilung von Datentupeln in endlich viele Kategorien, genannt Klassen

→ Ist der neue Kunde einer Bank kreditwürdig – Ja oder Nein?

Prädiktion: Vorhersage unbekannter numerischer Attributwerte von Datentupeln

→ Wie hoch ist das zukünftige Gehalt eines Studien-absolventen?

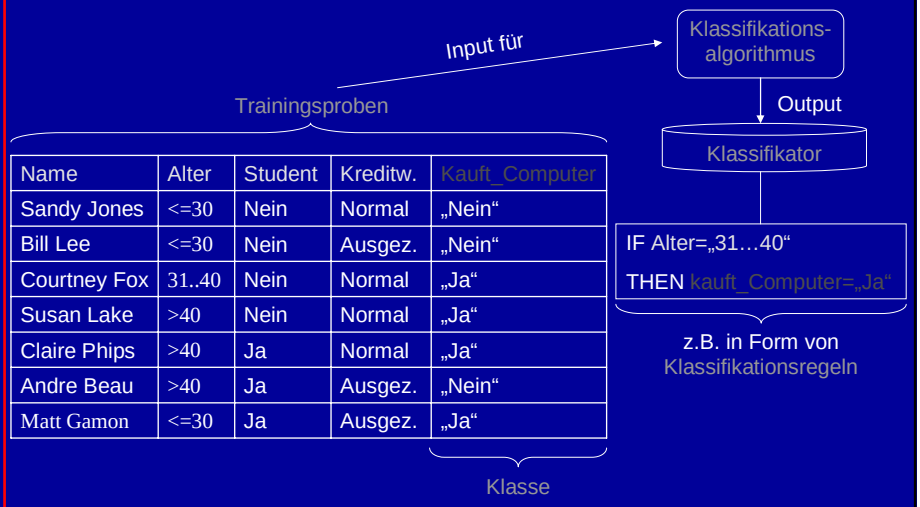
2. Grundlagen

Zwei Phasen der Klassifikation:

Phase 1: Bilden eines Modells, des so genannten Klassifikators, aus einer Menge von Trainingsdaten, den Trainingsproben

Phase 2: Klassifikation unbekannter Proben mithilfe des Klassifikators aus Phase 1, falls die Genauigkeit des Klassifikators akzeptabel ist

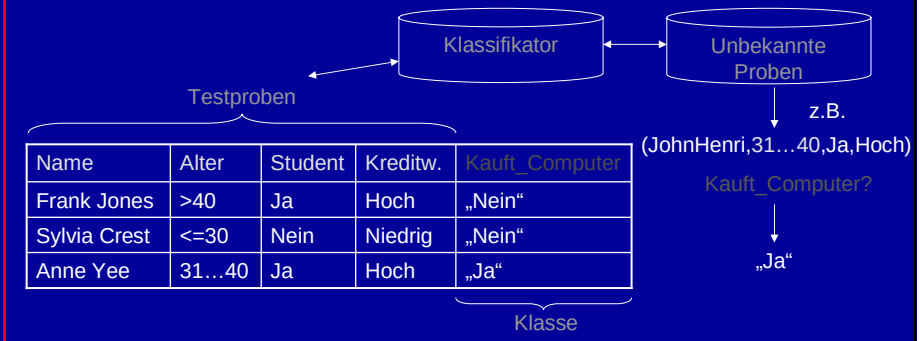
Phase 1



Name	Alter	Student	Kreditw.	Kauft_Computer
Sandy Jones	<=30	Nein	Normal	„Nein“
Bill Lee	<=30	Nein	Ausgez.	„Nein“
Courtney Fox	31..40	Nein	Normal	„Ja“
Susan Lake	>40	Nein	Normal	„Ja“
Claire Phips	>40	Ja	Normal	„Ja“
Andre Beau	>40	Ja	Ausgez.	„Nein“
Matt Gamon	<=30	Ja	Ausgez.	„Ja“

Klasse

Phase 2



Name	Alter	Student	Kreditw.	Kauft_Computer
Frank Jones	>40	Ja	Hoch	„Nein“
Sylvia Crest	<=30	Nein	Niedrig	„Nein“
Anne Yee	31…40	Ja	Hoch	„Ja“

Klasse

$$\text{Genauigkeit} = \frac{\#\text{ korrekt klassifizierte Testproben}}{\#\text{ Testproben}}$$

Genauigkeit > α: Klassifikationsregeln können zur Klassifikation neuer Daten genutzt werden

($z.B.\ \alpha = 0{,}9$)

3. Entscheidungsbauminduktion

Entscheidungsbaum wird als Klassifikator verwendet

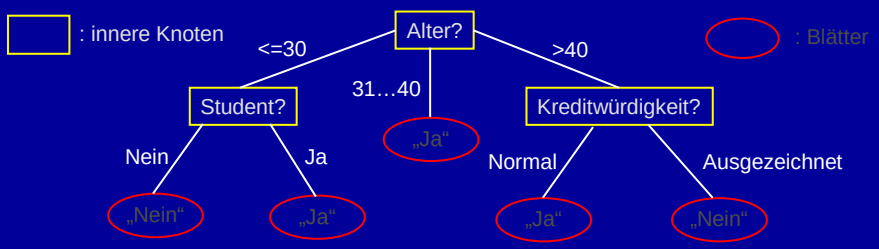
Innerer Knoten eines Entscheidungsbaums enthält Test auf ein Attribut, das Testattribut dieses Knotens

Blatt eines Entscheidungsbaums enthält die Klasse, die einer unbekannten Probe zugeordnet wird

Klassifikation einer unbekannten Probe:

Baum wird entsprechend den Attributwerten der Probe durchlaufen und ordnet die Probe derjenigen Klasse zu, die im so erreichten Blatt enthalten ist

Beispiel: Entscheidungsbaum



Einteilung in die beiden Klassen kauft_Computer=„Ja“ und kauft_Computer=„Nein“ anhand der Attribute Alter, Student, und Kreditwürdigkeit

Klassifikation: Dem Wert des im inneren Knoten angegeben Attributs einer Probe entsprechend dem Pfad des Baums folgen, bis Blatt erreicht

→ Blatt enthält die Klasse, der die Probe zugeordnet wird

Basisalgorithmus(1)

Algorithmus: Generiere_Entscheidungsbaum
Input: Menge der Trainingsproben, Proben; Menge der Testattributkandidaten, Attributliste
Output: Entscheidungsbaum

(1) Bilde Knoten K
(2) If Proben gehören alle zu Klasse C Then
(3) Return K als Blatt der Klasse C
(4) If Attributliste ist leer Then
(5) Return K als Blatt der Klasse, die in Proben am häufigsten vorkommt
(6) Wähle Testattribut als das Attribut aus Attributliste mit dem höchsten Informationsgewinn
(7) Kennzeichne Knoten K mit Testattribut

Basisalgorithmus(2)

(8) For Each bekannten Wert a_i von Testattribut
(9) Füge Ast ausgehend von Knoten K mit Bedingung Testattribut = a_i hinzu
(10) Setze S_i gleich der Menge der Proben aus Proben mit Testattribut = a_i
(11) If S_i ist leer Then
(12) Füge ein Blatt gekennzeichnet mit der häufigsten Klasse in Proben hinzu
(13) Else Füge den von Generiere_Entscheidungsbaum(S_i, Attributliste - Testattribut) zurückgegebenen Teilbaum hinzu

Informationsbedarf

Seien:

S Menge von s Proben;

$C_1,\ldots,C_n$ Klassen;

s_i Anzahl der Proben von S aus C_i;

p_i relative Häufigkeit der Klasse C_i in S;

C_1: kauft_Computer=„Ja“ mit s_1=4 und p_1=4/7

C_2: kauft_Computer=„Nein“ mit s_2=3 und p_2=3/7

RID	Alter	Student	Kreditw.	Klasse
1	<=30	Nein	Normal	Nein
2	<=30	Nein	Ausgez.	Nein
3	31..40	Nein	Normal	Ja
4	>40	Nein	Normal	Ja
5	>40	Ja	Normal	Ja
6	>40	Ja	Ausgez.	Nein
7	<=30	Ja	Ausgez.	Ja

S mit s=7

Informationsbedarf: $I(s_1,\ldots,s_n) = -\sum_{i=1}^{n} p_i \log_2(p_i)$

➡ $I(s_1,s_2) = I(4,3) = -4/7\log_2(4/7) - 3/7\log_2(3/7) = 0{,}985$

Entropie

Seien:

A Attribut mit v verschiedenen Werten;

S_j Menge der Proben aus S mit A = a_j;

s_{ij} Anzahl der Proben aus S_j, die zu C_i gehören;

RID	Alter	Student	Kreditw.	Klasse
1	<=30	Nein	Normal	Nein
2	<=30	Nein	Ausgez.	Nein
3	31..40	Nein	Normal	Ja
4	>40	Nein	Normal	Ja
5	>40	Ja	Normal	Ja
6	>40	Ja	Ausgez.	Nein
7	<=30	Ja	Ausgez.	Ja

Entropie:

$$E(A) = \sum_{j=1}^{v} \underbrace{\frac{s_{1j}+\ldots+s_{nj}}{s}}_{\text{Gewichtung}} \underbrace{I(s_{1j},\ldots,s_{nj})}_{\text{Informations-bedarf}}$$

Alter = „<=30“: s_{11} = 1, s_{21} = 2;

$I(s_{11},s_{21}) = -1/3\log_2(1/3) - 2/3\log_2(2/3) = 0{,}918$

Alter = „31…40“: s_{12} = 1, s_{22} = 0;

$I(s_{12},s_{22}) = -1\log_2(1) - 0\log_2(0) = 0$

Alter = „>40“: s_{13} = 2, s_{23} = 1;

$I(s_{13},s_{23}) = -2/3\log_2(2/3) - 1/3\log_2(1/3) = 0{,}918$

⬇

E(Alter) = 3/7*0,918 + 1/7*0 + 3/7*0,918 = 0,787

Informationsgewinn

Informationsgewinn:

$$Gewinn(A) = \underbrace{I(s_1,...,s_n)}_{\text{Informations-bedarf}} - \underbrace{E(A)}_{\text{Entropie}}$$

Also: Gewinn(Alter) = I(4,3) – E(Alter) = 0,985 – 0,787 = 0,198

Analog: Gewinn(Student) = 0,020,
Gewinn(Kreditwürdigkeit) = 0,128

➡ Aufgrund des höchsten Informationsgewinns Auswahl von Alter als Testattribut

Beispiel

Alter?

<=30

Alter	Student	Kreditw.	Klasse
<=30	Nein	Normal	Nein
<=30	Nein	Ausgez.	Nein
<=30	Ja	Ausgez.	Ja

31…40

Alter	Student	Kreditw.	Klasse
31..40	Nein	Normal	Ja

>40

Alter	Student	Kreditw.	Klasse
>40	Nein	Normal	Ja
>40	Ja	Normal	Ja
>40	Ja	Ausgez.	Nein

Baumbeschneidung

Widerspiegelung von Datenanomalien in vielen Ästen

Baumbeschneidung: Entfernen der unzuverlässigsten Äste

Zwei Arten von Baumbeschneidung:

Prepruning: Vorzeitiges Beenden des Aufbaus von Ästen, die nicht zuverlässig genug sind

Postpruning: Entfernen von Ästen eines voll ausgebildeten Baums

Skalierbarkeitsbetrachtung

Skalierbarkeit der Algorithmen wichtig für das Data-Mining, da große Datenmengen klassifiziert werden müssen

Basisalgorithmus (u.a. übliche Algorithmen) nicht skalierbar, denn sobald die Trainingsmenge größer dem verfügbaren Hauptspeicher ist, wird die Leistung durch ständige Ein- und Auslagerung beeinträchtigt

Deshalb: Entwicklung von skalierbaren Algorithmen für das Data-Mining

Hier: Betrachtung der Algorithmen SLIQ und SPRINT

SLIQ + SPRINT: Gemeinsamkeiten

Aufteilen aller Proben eines Knotens auf zwei Söhne (Schnitt)

Numerische Attribute: Schnitt der Form $A \leq a$

Kategorische Attribute: Schnitt der Form $A \in A'$, $A' \subset \{a_1,\ldots,a_n\}$

Nutzung des gini-Index: $gini(S) = 1 - \sum p_j^2$

$$gini_{split}(S) = \frac{n_1}{n} gini(S_1) + \frac{n_2}{n} gini(S_2)$$

Auswerten aller möglichen Schnitte für einen Knoten und Auswahl des Schnittes mit dem niedrigsten Schnittindex

Verwendung von Attributlisten, die für numerische Attribute vorsortiert werden (Presorting) und auf dem Externspeicher gehalten werden

SLIQ

Attributlisten enthalten für alle Trainingsproben jeweils den entsprechenden Attributwert und RID (Record Identifier) und werden zu Beginn sortiert generiert (Presorting)

Zusätzlich Klassenliste mit RID, Klasse und Referenz auf zugehörigen Knoten des Baums

RID	Alter	Gehalt	Klasse
1	30	65	G
2	23	15	B
3	40	75	G
4	55	40	B
5	55	100	G
6	45	60	G

Trainingsproben

Alter	RID
23	2
30	1
40	3
45	6
55	5
55	4

Gehalt	RID
15	2
40	4
60	6
65	1
75	3
100	5

Vorsortierte Attributlisten

RID	Klasse	Knoten
1	G	N1
2	B	N1
3	G	N1
4	B	N1
5	G	N1
6	G	N1

Klassenliste

SLIQ: Vorgehensweise

Sukzessives Aufbauen der einzelnen Ebenen des Baums

Für jede Ebene:

- Durchlauf aller Attributlisten und Auswerten des Schnittindex mithilfe von für jeden Knoten gehaltenen Histogrammen an der aktuellen Position
- Für jeden Knoten der Ebene Auswahl des Schnitts mit dem kleinsten Schnittindex und Aufteilung der zugehörigen Proben auf linken und rechten Sohn entsprechend dem Schnitt
- Durchlauf der in Schnitten benutzten Attributlisten und Aktualisierung der Knotenreferenzen in der Klassenliste
- Lokale Terminierung falls ein Knoten nur noch Proben einer Klasse enthält

SLIQ: Beispiel

Schnitt →

Alter	RID
23	2
30	1
40	3
45	6
55	5
55	4

RID	Klasse	Knoten
1	G	N3
2	B	N2
3	G	N3
4	B	N1
5	G	N1
6	G	N1

	B	G
L	2	4
R	0	0

Histogramme L und R halten Klassenverteilung für die Söhne

actN1($gini_{split}$): 4/9

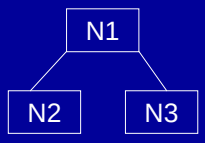
minN1($gini_{Split}$): 4/15

Für Schnitt: Alter ≤ 23

$gini_{Alter \le 22}(S) = 0/6 * gini(S_L) + 6/6 * gini(S_R)$

$= 0 * (1 - (0^2 + 0^2)) + 1 * (1 - ((2/6)^2 + (4/6)^2))$

$= 0 + 1 - (1/9 + 4/9) = 4/9$



$$gini_{split}(S) = \frac{n_L}{n} gini(S_L) + \frac{n_R}{n} gini(S_R)$$

$$gini(S) = 1 - \sum_{j \in \{B,G\}} p_j^2$$

SPRINT

Attributlisten enthalten für alle Trainingsproben jeweils den entsprechenden Attributwert, RID und die Klasse und werden auch zu Beginn sortiert generiert (Presorting)

Keine zusätzliche Klassenliste

Attributlisten werden bei jedem Schnitt partitioniert

→ Eigene Attributlisten für jeden Knoten

Alter	Klasse	RID
23	B	2
30	G	1
40	G	3
45	G	6
55	G	5
55	B	4

Gehalt	Klasse	RID
15	B	2
40	B	4
60	G	6
65	G	1
75	G	3
100	G	5

Vorsortierte Attributlisten für SPRINT

SPRINT: Vorgehensweise

Sukzessives Aufbauen der einzelnen Knoten des Baums

Für jeden Knoten:

- Durchlauf aller Attributlisten des Knotens und Auswerten des Schnittindex mithilfe der Histogramme L und R
- Auswahl des Schnitts mit dem kleinsten Schnittindex
- Partitionierung der Attributlisten des Knotens auf linken und rechten Sohn entsprechend dem ausgewählten Schnitt:
  - → Für die im Schnitt benutzte Attributliste AListe einfaches verschieben der einzelnen Einträge
  - → Für die anderen Attributlisten Probing der RID auf eine Hash-Tabelle, die bei der Partitionierung von AListe für alle Proben des linken Baums aufgebaut wird, um zu wissen, zu welchem Sohn der Eintrag verschoben werden muss
- Lokale Terminierung, falls ein Knoten nur Proben einer Klasse enthält

SLIQ + SPRINT: Performance

SLIQ bei „kleinen“ Datenmengen vergleichbar mit üblichen Algorithmen

Vergleich zwischen SLIQ und üblichen Algorithmen bei sehr großen Datenmengen nicht lohnenswert

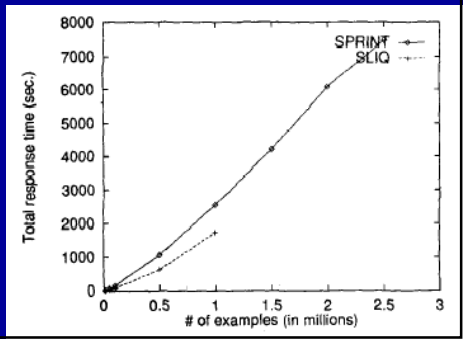
Deshalb: Vergleich von SLIQ und SPRINT

SLIQ schneller als SPRINT

Aber: Thrashing sobald Klassenliste nicht in Hauptspeicher passt

Bei SPRINT lineare Abhängigkeit der Antwortzeit von der Größe der Trainingsmenge

→ SPRINT vollständig skalierbar



4. Bayes‘sche Klassifikation

Anwendung des Satzes von Bayes zur Bestimmung der Wahrscheinlichkeiten p_i, dass eine Probe X zur Klasse C_i gehört

Klassifikation von X zu der Klasse mit höchstem p_i

Naiver Bayes‘scher Klassifikator:

Annahme der klassenbedingten Unabhängigkeit

Bayes‘sche Netze:

Modellierung von Abhängigkeiten

Stochastische Grundlagen

Seien: $X = (x_1,\ldots,x_n)$ unbekannte Probe;

H Hypothese, dass X zu Klasse C gehört;

P(H|X): A-Posteriori-Wahrscheinlichkeit von H

P(H): A-Priori-Wahrscheinlichkeit von H

P(X|H): A-Posteriori-Wahrscheinlichkeit von X

P(X): A-Priori-Wahrscheinlichkeit von X

Beispiel: X = („rot", „rund"); C = Apfel

Satz von Bayes: $P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)}$

Naiver Bayes'scher Klassifikator

Seien: $C_1,\ldots,C_m$ Klassen; $X = (x_1,\ldots,x_n)$ unbekannte Probe

Maximierung von $P(C_i|X)$ $\longleftrightarrow$ Maximierung von $\frac{P(X \mid C_i)P(C_i)}{P(X)}$

Satz von Bayes

P(X) konstant für alle C_i: Maximierung von $P(X|C_i)*P(C_i)$

Abschätzung von $P(C_i)$ durch relative Häufigkeit p_i der Klasse C_i in der Trainingsmenge

Annahme der klassenbedingten Unabhängigkeit:

$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$ → Deswegen naiv

Berechnung der $P(x_k|C_i)$

Sei A_k das x_k entsprechende Attribut

A_k kategorisch:

s_{ik} = # Proben aus C_i mit $A_k = x_k$

s_i = # Proben aus C_i

Abschätzung von $P(x_k|C_i)$ durch $P(x_k|C_i) = s_{ik} / s_i$

A_k numerisch:

$$P(x_k \mid C_i) = \frac{1}{\sqrt{2\pi}\sigma_{C_i}} e^{\frac{(x_k - \mu_{C_i})^2}{2\sigma_{C_i}^2}}$$

Gauß-Verteilung

Dabei sind μ_{c_i} und σ_{C_i} die aus den Trainingsproben der Klasse C_i ermittelten Werte für Mittelwert und Standardabweichung

Beispiel

C_1: kauft_Computer="Ja" mit s_1= 4 und p_1 = $P(C_1)$ = 4/7

C_2: kauft_Computer="Nein" mit s_2= 3 und p_2 = $P(C_2)$ = 3/7

Klassifikation von
X = („31…40", „Nein", „Normal")

Alter: s_{11} = 1; s_{21} = 0;

$P(\text{„31…40"}|C_1) = s_{11} / s_1 = 1 / 4$
$P(\text{„31…40"}|C_2) = s_{21} / s_2 = 0 / 3 = 0$

Student: s_{12} = 2, s_{22} = 2;

$P(\text{„Nein"}|C_1) = s_{12} / s_1 = 2 / 4 = 1 / 2$
$P(\text{„Nein"}|C_2) = s_{22} / s_2 = 2 / 3$

Kreditwürdigkeit: s_{13} = 3, s_{23} = 1;

$P(\text{„Normal"}|C_1) = s_{13} / s_1 = 3 / 4$
$P(\text{„Normal"}|C_2) = s_{23} / s_2 = 1 / 3$

RID	Alter	Student	Kreditw.	Klasse
1	<=30	Nein	Normal	Nein
2	<=30	Nein	Ausgez.	Nein
3	31..40	Nein	Normal	Ja
4	>40	Nein	Normal	Ja
5	>40	Ja	Normal	Ja
6	>40	Ja	Ausgez.	Nein
7	<=30	Ja	Ausgez.	Ja

$P(X|C_1)$ = $P(x_1|C_1)*P(x_2|C_1)*P(x_3|C_1)$
= 1 / 4 * 1 / 2 * 3 / 4 = 3 / 36

$P(X|C_2)$ = $P(x_1|C_2)*P(x_2|C_2)*P(x_3|C_2)$
= 0 * 2 / 3 * 1 / 3 = 0

$P(X|C_1)*P(C_1)$ = 3 / 36 * 4 / 7 = 1 / 21

$P(X|C_2)*P(C_2)$ = 0 * 3 / 7 = 0

Bayes'sche Netze

Graphische Modellierung von Abhängigkeiten zwischen Attributen durch azyklischen gerichteten Graph

Attribut bei gegebenen direkten Vorgängern unabhängig von Nicht-Nachfolgern

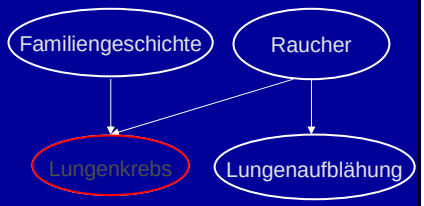
CPT für ein Attribut A speichert bedingte Wahrscheinlichkeiten P(A|Vorgänger(A)) für jede Wertekombination der Vorgänger

Zuordnung einer unbekannten Probe zu der Klasse mit höchstem $P(C_i|X)$

→ ≙ $P(C_i|\text{Vorgänger}(C_i))$

z.B. P(LK|FG,R) = 0,8 und P(~LK|FG,R) = 0,2

Wie erhält man die Werte $P(C_i|\text{Vorgänger}(C_i))$?



	FG,R	FG,~R	~FG,R	~FG,~R
LK	0,8	0,5	0,7	0,1
~LK	0,2	0,5	0,3	0,9

CPT: Conditional Probability Table

Trainieren Bayes'scher Netze

3 Szenarien:

- Netzstruktur gegeben, keine fehlenden Werte
  - → Berechnung der P(A|Vorgänger(A)) ähnlich wie bei naivem Bayes'schen Klassifikator
- Netzstruktur gegeben, aber fehlende Werte möglich
  - → Annäherung durch Gradientenabstieg
- Netzstruktur unbekannt
  - → Diskrete Optimierung

5. Regression

Verwendung zur Prädiktion, also der Vorhersage von numerischen Attributwerten

Rückführung eines Zielattributwertes einer unbekannten Probe auf die Verteilung einer analysierten Trainingsmenge

Hier:
- Lineare Regression
- Multiple Regression

Lineare Regression

Modellierung einer Antwortvariablen Y durch eine auf eine Schätzervariable X angewendete lineare Funktion

Regressionsgleichung: $Y = \underbrace{\alpha + \beta}_{\text{Regressionskoeffizienten}} * X$

Methode der kleinsten Quadrate zur Bestimmung der Regressionskoeffizienten:

$$\beta = \frac{\sum_{i=1}^{s}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{s}(x_i - \bar{x})^2} \qquad \alpha = \bar{y} - \beta\bar{x}$$

$\bar{x}$: Mittelwert der Werte der Schätzervariable X in den Trainingsproben

$\bar{y}$: Mittelwert der Werte der Antwortvariable Y in den Trainingsproben

x_i, y_i: Wert von X bzw. Y der i-ten Trainingsprobe

Beispiel

Mittelwert für X: 8,8

Mittelwert für Y: 51,2

$$\beta = \frac{(1-8{,}8)(33-51{,}2)+(3-8{,}8)(30-51{,}2)+\ldots+(13-8{,}8)(72-51{,}2)}{(1-8{,}8)^2+(3-8{,}8)^2+\ldots+(13-8{,}8)^2} = 3{,}1$$

$$\alpha = 51{,}2-(3{,}1)(8{,}8) = 23{,}9$$

Gehaltsvorhersage für Person mit X = 7 Jahren Berufserfahrung:

$Y = \alpha + \beta * X = 23{,}9 + 3{,}1 * 7 = 45{,}6$

X – Berufserfahrung (Jahre)	Y - Gehalt (in 1000$)
1	33
3	30
8	57
9	64
13	72

Trainingsproben

Multiple Regression

Erweiterung der linearen Regression auf mehrere Schätzervariablen $X_1,\ldots,X_n$

→ Modellierung von Y durch mehrdimensionalen Attributvektor

Multiple Regressionsgleichung:

$$Y = \alpha + \beta_1 * X_1 + \beta_2 * X_2 + \ldots + \beta_n * X_n$$

Berechnung der Regressionskoeffizienten α und $\beta_1,\ldots,\beta_n$ ebenfalls mit Methode der kleinsten Quadrate

6. Zusammenfassung und Ausblick

Vorstellung der Klassifikationskonzepte Entscheidungsbaum-induktion und Bayes'sche Klassifikation

Vorstellung des Prädiktionskonzeptes der linearen und multiplen Regression

Skalierbarkeitsbetrachtungen im Zusammenhang mit immer größeren Datenmengen (SLIQ und SPRINT)

Weiterhin großes Interesse an schnelleren skalierbaren Algorithmen aufgrund des immensen Datenwachstums

Forschungsschwerpunkte: Klassifikation von nicht-relationalen Daten wie z.B. Textdokumenten, räumlichen Daten oder Multimedia-Daten