Applied Analytical Data Science

Teil 5: Klassifikation und Regression

Dr. Jörg-Uwe Kietz,
Vorlesung an der Univ. Zürich,

Mittwoch, 14:00-15:45 Uhr Vorlesung,
16:00-17:30 Uhr Übung

http://www.kietz.ch/AADS/

Heutiges Programm

- Definition und Anwendungen von Klassifikation und Regression
- Entscheidungsbäume (Top-Down Induction of Decision Trees)
  - Ein einfacher Entscheidungsbaum
  - Attribute Selektion durch Informationsgewinn
  - Vorhersagequalität, Overfitting und Pruning
- Andere Klassifikationsmethoden
  - Regellernen: Geordnete und Ungeordnete Regelmengen
  - SVM: lineare Diskriminierung mit Attribut Konstruktion
  - IBL: Klassifikation durch Bestimmung nächster Nachbarn
  - Neuronale Netze: nicht lineare Diskriminierung
  - Kombination mehrerer Klassifikatoren
- Warum funktioniert Vorhersage

Definition Klassifikation und Regression

Gegeben:

Eine Menge von Beispielen, beschrieben durch

- Eingabeattribute, und
- ein Zielattribut: nominal (Klassifikation)
  numerisch (Regression)

Gesucht:

Eine Funktion (Baum, Regeln, Funktion, Neuronales Netz ...):

- zur Berechnung/Vorhersage des Zielattributes, für jedes neue, nur durch die Eingabeattribute beschriebene Beispiel.

Klassifikation und Regression: Anwendungen

Analyse alter Entscheidungen / Werte

- Wer kaufte ein Produkt?
- Bei wieviel Kredit gab es keine Ausfälle?
- Welche Benutzung war ein Missbrauch?

⇒ Vorhersage von zukünftigen Entscheidungen / Werten

- Wer wird ein Produkt kaufen?
- Bei wieviel Kredit wird es keine Ausfälle geben?
- Welche Benutzung wird ein Missbrauch sein?

Fehler

- D ist eine Wahrscheinlichkeitsverteilung über E
- E ist die Menge aller Beispiele
- h ist die gelernte Funktion
- f ist die wirkliche Funktion

Trainingsfehler:

$\text{fehler}_D(h) = \Sigma_{(x,y) \in E} \text{error}(h(x),y)$, mit

- $\text{error}(h(x),y) = (h(x) - y)^2$ für numerische Werte (quadrierter Fehler)
- $\text{error}(h(x),y) = 0$, wenn $h(x) = y$, sonst 1 für nominale Werte (0-1-loss)

Wahrer Fehler:

$\text{fehler}_D(h,f) = E_D[\text{error}(h(x),f(x))]$

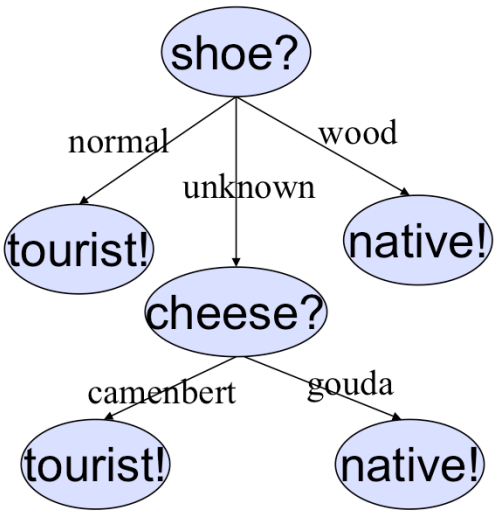
Heutiges Programm

- Definition und Anwendungen von Klassifikation und Regression
- Entscheidungsbäume (Top-Down Induction of Decision Trees)
  - Ein einfacher Entscheidungsbaum
  - Attribute Selektion mit Informationsgewinn
  - Vorhersagequalität, Overfitting und Pruning
- Andere Klassifikationsmethoden
  - Regellernen: Geordnete und Ungeordnete Regelmengen
  - SVM: lineare Diskriminierung mit Attribute Konstruktion
  - IBL: Klassifikation durch Bestimmung nächster Nachbarn
  - Neuronale Netze: nicht lineare Diskriminierung
  - Kombination mehrerer Klassifikatoren
- Warum funktioniert Vorhersage

Ein einfacher Entscheidungsbaum

How to recognize tourists in Netherlands ... (no warranty) ?

id	shoe	cheese	caravan	tourist
1	unknown	gouda	yes	no
2	unknown	camenbert	yes	yes
3	wood	gouda	yes	no
4	normal	gouda	no	yes
5	wood	camenbert	yes	no
6	unknown	gouda	no	no
7	normal	gouda	yes	yes
8	wood	gouda	no	no
9	unknown	camenbert	no	yes
10	normal	camenbert	yes	yes
11	unknown	gouda	no	???



Lernen von Entscheidungsbäumen

- Warum dieser Entscheidungsbaum und nicht ein anderer?
  - Globale Kriterien oder welchen Baum will ich?
  - Lokale Kriterien oder wie finde ich ihn?

=> Attributeauswahl durch Informationsgewinn

- Welcher Baum eignet sich am besten zur Vorhersage

=> Abschätzen der Vorhersagequalität

=> Overfitting und Pruning

Welcher Entscheidungsbaum

Globale Kriterien oder welchen Baum will ich?

- Die beste Vorhersagequalität
- Der kleinste (verständlichste) Baum

=> Finden des global besten Entscheidungsbaums ist zu aufwendig.

Lokale Kriterien oder wie finde ich einen akzeptablen Baum?

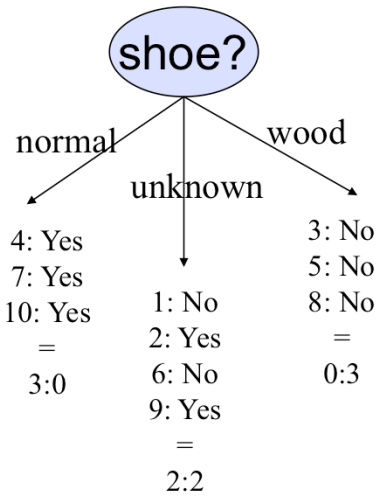
- Auf jeder Ebene das Attribut zur Entscheidung auswählen, das die Klassifikation am stärksten verbessert.

=> Attributeauswahl z.B. durch **Informationsgewinn**

=> Kurzsichtigkeit: Ein lokales Kriterium führt nicht zum globalen Optimum (z.B: XOR-Funktion)

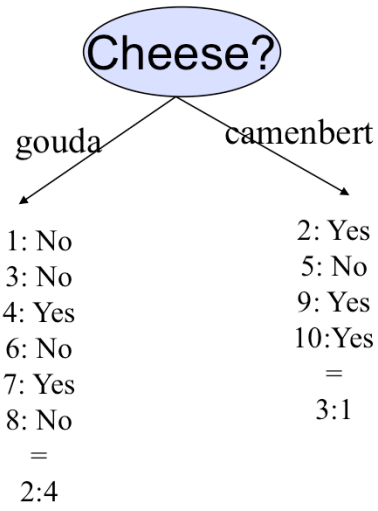
Attributeauswahl durch Informationsgewinn

id	shoe	cheese	caravan	tourist
1	unknown	gouda	yes	no
2	unknown	camenbert	yes	yes
3	wood	gouda	yes	no
4	normal	gouda	no	yes
5	wood	camenbert	yes	no
6	unknown	gouda	no	no
7	normal	gouda	yes	yes
8	wood	gouda	no	no
9	unknown	camenbert	no	yes
10	normal	camenbert	yes	yes



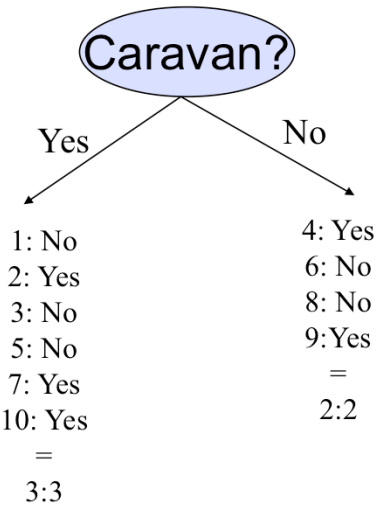
Attributeauswahl durch Informationsgewinn

id	shoe	cheese	caravan	tourist
1	unknown	gouda	yes	no
2	unknown	camenbert	yes	yes
3	wood	gouda	yes	no
4	normal	gouda	no	yes
5	wood	camenbert	yes	no
6	unknown	gouda	no	no
7	normal	gouda	yes	yes
8	wood	gouda	no	no
9	unknown	camenbert	no	yes
10	normal	camenbert	yes	yes



Attributeauswahl durch Informationsgewinn

id	shoe	cheese	caravan	tourist
1	unknown	gouda	yes	no
2	unknown	camenbert	yes	yes
3	wood	gouda	yes	no
4	normal	gouda	no	yes
5	wood	camenbert	yes	no
6	unknown	gouda	no	no
7	normal	gouda	yes	yes
8	wood	gouda	no	no
9	unknown	camenbert	no	yes
10	normal	camenbert	yes	yes



Informationsgehalt

- Botschaft, die mitteilt, dass beliebig ausgewähltes Beispiel aus Menge S in Klasse C_i liegt, hat den Informationsgehalt

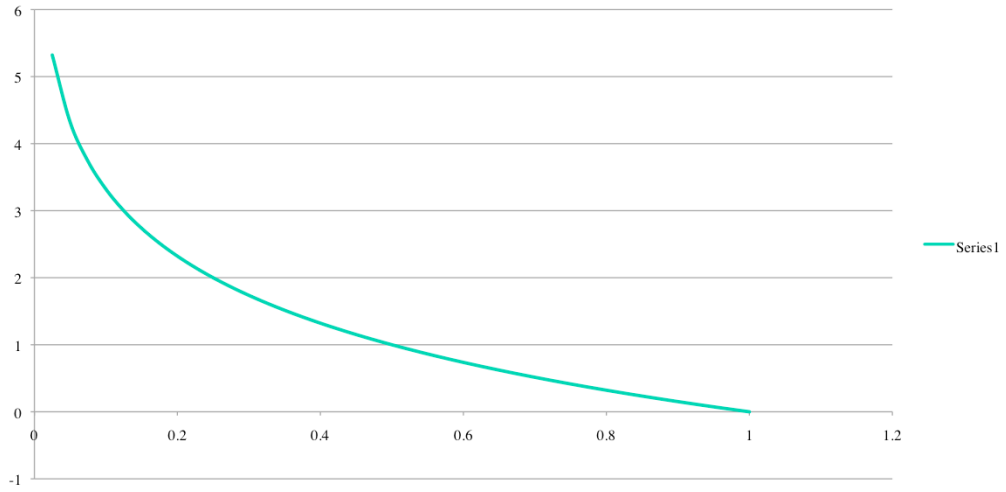
$$-\log_2\left(\frac{|C_i \cap S|}{|S|}\right) \text{ bits}$$

- Erwartungswert für den Informationsgehalt dieser Botschaften (bei *k* Klassen C_1 , ... ,C_k):

$$info(S) = -\sum_{j=1}^{k} \frac{|C_j \cap S|}{|S|} * \log_2\left(\frac{|C_j \cap S|}{|S|}\right) \text{ bits}$$

(Entropie von S)

-lg from 0.025 to 1



Informationsgehalt von Tests

- Ein Test partitioniert T in T_1, ..., T_n, dann ist der Erwartungswert für den Informationsgehalt einer Botschaft, die mitteilt, dass ein beliebig ausgewähltes Beispiel aus T_i in der Klasse C_i liegt (bei *k* Klassen C_1, ..., C_k):

$$info(T_i) = -\sum_{j=1}^{k} \frac{|C_j \cap T_i|}{|T_i|} * \log_2 \left(\frac{|C_j \cap T_i|}{|T_i|} \right) \text{ bits}$$

- Damit ist der Erwartungswert über alle T_1, ..., T_n :

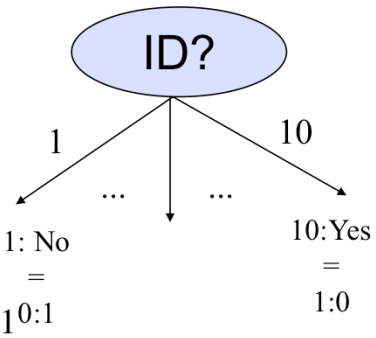
$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} * \text{info}(T_i)$$

Informationsgehalt der Beispiel Tests

- Top: info([5:5]) = - (5/10 * $\log_2$ 5/10 + 5/10 * $\log_2$ 5/10) = -1.0 bits
- Shoes: $info_X$([3:0, 2:2, 0:3]) =
  - (3/10 * (3/3 * $\log_2$ 3/3 + 0/3 * $\log_2$ 0/3) +
    4/10 * (2/4 * $\log_2$ 2/4 + 2/4 * $\log_2$ 2/4) +
    3/10 * (0/3 * $\log_2$ 0/3 + 3/3 * $\log_2$ 3/3)) = -0.4 bits
- Cheese: $info_X$([2:4, 3:1]) =
  - (6/10 * (2/6 * $\log_2$ 2/6 + 4/6 * $\log_2$ 4/6) +
    4/10 * (3/4 * $\log_2$ 3/4 + 1/4 * $\log_2$ 1/4)) = -0.875 bits
- Caravan: $info_X$([3:3, 2:2]) =
  - (6/10 * (3/6 * $\log_2$ 3/6 + 3/6 * $\log_2$ 3/6) +
    4/10 * (2/4 * $\log_2$ 2/4 + 2/4 * $\log_2$ 2/4)) = -1.0 bits

Aber: Informationsgehalt von Schlüsseln

- Informationsgewinn bevorzugt Attribute mit vielen verschiedenen Werten, z.B. die ID
- info([0:1, ... 1:0]) = 0 bits
- Deshalb:
  gain ratio$_X$(T) =
  (info(Top) - info$_X$(T)) / info(T)
  - gain-ratio(id) = (1 - 0) / 3.332 = 0.301
  - **gain-ratio(shoes) = (1 - 0.4) / 1.571 = 0.382**
  - gain-ration(cheese) = (1 - 0.875) / 0.971 = 0.128
  - gain-ration(caravan) = (1 - 1) / 1 = 0



Abschätzen der Vorhersagequalität

- Der Entscheidungsbaum wird solange verfeinert, bis alle Knoten nur noch eine Klasse enthalten.

  => Der Entscheidungsbaum macht auf den Daten auf denen er erstellt wird keinen Fehler.

- Das Ziel der Erzeugung von Entscheidungsbäumen ist ein möglichst geringer Fehler für zukünftige Daten.
  - Wie gross ist dieser Fehler?

  => Abschätzung des Fehlers durch Anwendung des Entscheidungsbaumes auf unabhängige **Testdaten**

Overfitting

Def.: Overfitting (Überspezialisierung) (Mitchell, 1997)
Eine Klassenbeschreibung L ist in Bezug auf eine Menge von Trainingsdaten überspezialisiert , wenn es eine alternative Klassenbeschreibung L‘ gibt, so dass L in Bezug auf die Trainingsdaten eine geringere Fehlerrate hat als L‘, aber in Bezug auf alle möglichen Beispiele (insbesondere bisher nicht bekannte Daten wie die Testdaten) eine grössere Fehlerrate hat als L‘.

Pruning

- Aufteilung der Beispielmenge in Trainings- und Pruningdaten
- Erstellung des Entscheidungsbaumes auf den Trainingsdaten
- Pruning: Die Anwendung des erstellten Entscheidungsbaumes auf die Pruningdaten ergibt eine Schätzung des Fehler auf neuen unabhängigen Daten!
  - **Alle Teilbäume die auf den Pruningdaten einen höheren Fehler haben, als der Wurzelknoten des Teilbaums werden gelöscht, d.h. zu einfachen Blättern**.
- Pruning reduziert den Fehler auf den Pruningdaten, aber es erhöht den Fehler auf den Trainingsdaten.

Heutiges Programm

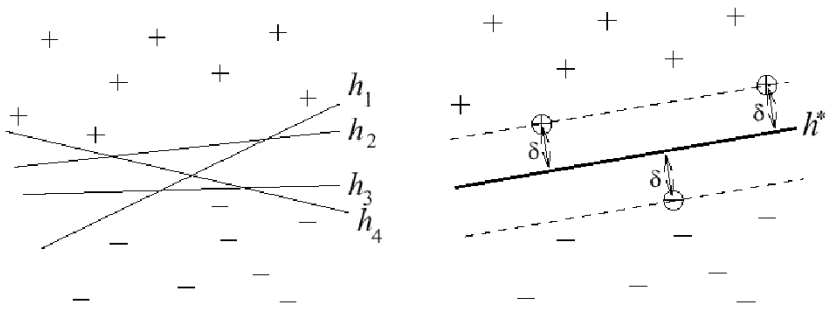
- Definition und Anwendungen von Klassifikation und Regression
- Entscheidungsbäume (Top-Down Induction of Decision Trees)
  - Ein einfacher Entscheidungsbaum
  - Attribute Selektion mit Informationsgewinn
  - Vorhersagequalität, Overfitting und Pruning
- Andere Klassifikationsmethoden
  - Regellernen: Geordnete und Ungeordnete Regelmengen
  - SVM: lineare Diskriminierung mit Attribute Konstruktion
  - IBL: Klassifikation durch Bestimmung nächster Nachbarn
  - Neuronale Netze: nicht lineare Diskriminierung
  - Kombination mehrerer Klassifikatoren
- Warum funktioniert Vorhersage

Regellernen

- Grosse Entscheidungsbäume werden oft unübersichtlich.
- Pruning kann nur Teilbäume ersetzen: Eine schlechte Auswahl der Top-Level Attribute ist nicht mehr zu ändern.
- Entscheidungsbäume können in Regelmengen übersetzt werden: Jeder Pfad von der Wurzel zu einem Blatt wird zu einer Regel.
  - ungeordnete Regelmengen (If-Then Rules): Es wird immer die Korrektheit gegenüber der Gesamtmenge getestet
  - geordnete Regelmengen (Entscheidungslisten, If-Then-Else Rules): Man testet immer nur noch die Korrektheit gegenüber den bisher nicht abgedeckten Beispielen.
- Direktes Regellernen mit inkrementellem Pruning ist besser

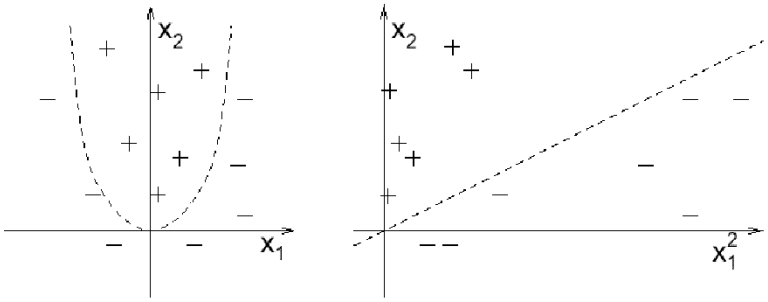
SVM: Support Vector Machine

- Finden der optimalen Hyperebene zur linearen Diskriminierung und ihre Stützvektoren (Support Vector)



SVM: Support Vector Machine

- Nicht-lineare Diskriminierung durch lineare Diskriminierung und die Konstruktion von nicht-linearen Attributen



SVM: Support Vector Machine

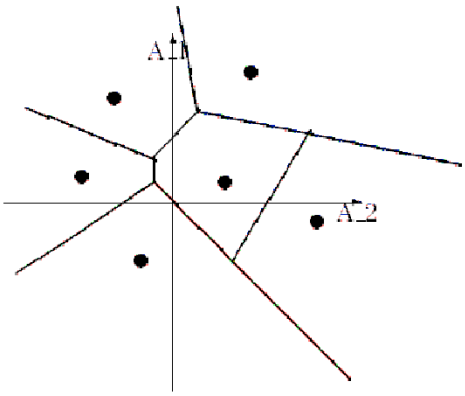
Eigenschaften der SVM:

- Nur auf skalare Attribute anwendbar.
- Reduktion von linearer Diskriminierung auf quadratische Optimierung.
- Nicht-lineare Klassifikation durch virtuelle Attributekonstruktion mit Hilfe von „Kernfunktionen“, wie z.B.
  - Polynome vom Grad *d*,
  - Sigmoide (entspricht zweilagigen Neuronalen Netzen)
- Die erwartete Vorhersagequalität hängt von der Anzahl der Stützvektoren (< |Attribute| + 2) und ihrem Abstand von der Hyperebene ab.

IBL: Instanzen basiertes Lernen

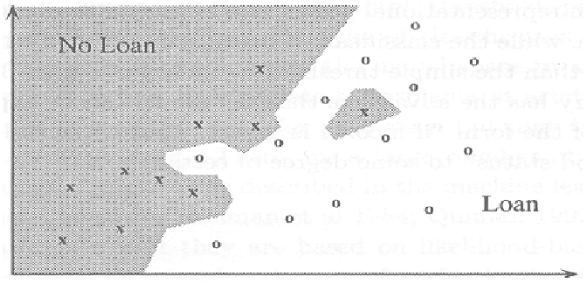
- Klassifikation (Regression) durch Bestimmung des ähnlichsten Beispiels.
- Ähnlichkeit sollte eine Metrik sein, z.B. der gewichtete (Skalierung!!) geometrischen Abstand (skalare oder binäre Attribute).
- Für Noise-Toleranz: Mehrheitsvotum (Mittelwert) der *k* nächsten Nachbarn.

Voronoi-Diagramm für Nachbarschaftsbeziehungen



IBL: Instanzen basiertes Lernen

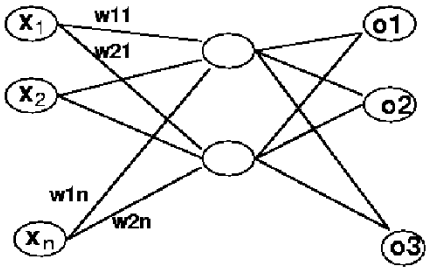
- IBL kann nicht-kontinuierliche, nicht-lineare Klassifikationen oder Regressionen lernen.



- Das Ergebnis hängt starkt von der Wahl der Gewichte (der Skalierung der Attribute) ab.
- Die Verständlichkeit des Ergebnisses sollte kein Ziel der KDD Anwendung sein.

Neuronale Netze

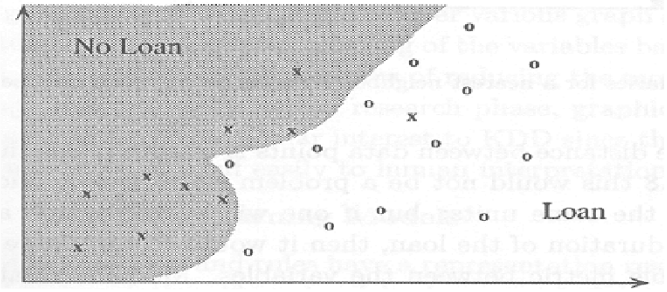
- Ein Menge von skalaren Eingabe-Attributen (x_1, ..., x_n).
- Eine festzulegende Anzahl von Neuronen in der Mittelschicht.
- Eine Menge von skalaren Ausgabe-Attributen (o_1, ..., o_n).



- Die Ausgabe eines Neuron berechnet sich durch die gewichtete Summe seiner Eingaben.
- Lernen/trainieren ist die Bestimmung der optimalen Gewichte z.B. Backpropagation der Ausgabefehler zur Minimierung des quadratischen Fehlers.

Neuronale Netze

- Neuronalen Netze können nicht-lineare Klassifikationen oder Regressionen lernen.



- Die Verständlichkeit des Ergebnisses (Gewichte an den Neuronen) sollte kein Ziel der KDD Anwendung sein.

Kombination mehrer Klassifikatoren

- **Bagging:** Alle Klassifikatoren werden über zufällig ausgewählten Teilmengen derselben Daten gelernt: Zur Klassifikation wird eine Mehrheitsentscheidung benutzt.
- **Boosting:** Weitere Klassifikatoren werden über den Daten gelernt, auf denen die bisherigen Klassifikatoren nicht gut waren. Bei der Vorhersage werden sie durch ihren Trainingsfehler gewichtet.
- **Stacking:** Mehrere Klassifikatoren werden gelernt, und es wird ein Meta-Klassifikator gelernt, der bestimmt, wann welcher Klassifikator anzuwenden ist.

=> Kombinierte Klassifikatoren funktionieren oft bedeutend besser. Boosting ist auch durch die PAC-Lerntheorie motiviert.

Heutiges Programm

- Definition und Anwendungen von Klassifikation und Regression
- Entscheidungsbäume (Top-Down Induction of Decision Trees)
  - Ein einfacher Entscheidungsbaum
  - Attribute Selektion mit Informationsgewinn
  - Vorhersagequalität, Overfitting und Pruning
- Andere Klassifikationsmethoden
  - Regellernen: Geordnete und Ungeordnete Regelmengen
  - SVM: lineare Diskriminierung mit Attribut Konstruktion
  - IBL: Klassifikation durch Bestimmung nächster Nachbarn
  - Neuronale Netze: nicht lineare Diskriminierung
  - Kombination mehrerer Klassifikatoren
- Warum funktioniert Vorhersage

Warum funktioniert Vorhersage

- **Im Allgemeinen kann Vorhersage nicht funktionieren.**
  Für alle Klassifikationslerner ist die Summe der Vorhersagequalitäten über allen möglichen Anwendungen genau die gleiche, insbesondere ist es die gleiche wie die von „zufälligem Raten“ (Wolpert 92, Schaffer 93).
- **Aber:** Die Wahrscheinlichkeit, das eine Klassifikator zufällig einen kleinen Fehler auf einer Beispielmenge macht, ist umso geringer, je grösser das Verhältnis der Grösse der Beispielmenge und der Kapazität der Menge, aus der er ausgewählt wurde, ist.
- **GIGO (Garbage In - Garbage Out):** Der Analyst sorgt dafür, dass Data Mining nur auf „gute“ Anwendungen angewendet wird.

Probably Approximately Correct Learning

Theorie des Maschinellen Lernens:

- Für welche **Klassifikationssysteme** sind **wahrscheinlich annähernd korrekte Ergebnisse** mit **polynomialer Anzahl von Beispielen** und **polynomialem Aufwand** zu erreichen.
- Liefert einen Rahmen um:
  - Bedingungen unter denen Vorhersage möglich ist zu bestimmen.
  - Die Kapazität von Klassifikationen zu untersuchen (VC-Dimension).
  - Die Anzahl der Beispiele die notwendig zum Lernen ist zu bestimmen.
  - Die Komplexität von Lernproblemen (im Unterschied zu speziellen Algorithmen) zu ermitteln.

Fazit

- Klassifikation & Regression sind die am häufigsten genutzten Data Mining Aufgaben.
- Klassifikation wird benutzt um kategoriale Attribute vorherzusagen.
- Es gibt viele verschiedene Methoden zur Klassifikation, die wichtigsten sind: TDIDT, Regel lernen, SVM, IBL und NN.
- Vorhersage kann nur funktionieren, wenn der Benutzer (Data Miner) sinnvolle Daten zum Lernen zur Verfügung stellt.
- Daten werden unterschieden in: Trainingsdaten, Prunningdaten, Testdaten, und schliesslich Anwendungsdaten.

Literatur

Data Mining Methoden zur Klassifikation und Regression:

- Witten, I.; Frank, E.: Practical Machine Learning Tools and Techniques with Java implementations, Morgan Kaufmann, 2000.

- K. Morik; S. Wrobel; T. Joachims: "Maschinelles Lernen und Data Mining" Beitrag zum »Handbuch KI«, G. Görz, J. Schneeberger und C.-R. Rollinger (Hrsg.), Oldenbourg Verlag, im erscheinen.
  PDF download: http://www-ai.informatik.uni-dortmund.de/LEHRE/VORLESUNGEN/MLRN/SKRIPT/handbuch_ki-ml.pdf

Literatur

Vertiefung Entscheidungsbäume:

- R. Quinlan: „C4.5: Programms for Machine Learning“, Morgan Kaufmann, 1993.
- L. Breiman; J.H. Friedman; R.A. Olshen; C.J. Stone: „Classification and regression trees“, Belmont, 1984.

Vertiefung Regellernen:

- W.W. Cohen: „Fast Effective Rule Induction“, Proceedings of the Twelfth International Conference on Machine Learning (ML95), 1995.

Literatur

Vertiefung Neuronale Netzte:

- C. M. Bishop: „Neuronal Networks for Pattern Recognition“ Clarendon Press, 1995.

Vertiefung Support Vector Machine:

- T. Joachims, „Making large-Scale SVM Learning Practical“, In: Advances in Kernel Methods - Support Vector Learning, B. Schölkopf; C. Burges; A. Smola (ed.), MIT-Press, 1999.
- V. N. Vapnik, „Statistical Learning Theory“, Wiley, 1998.