Data Mining für Technische Anwendungen – Grundlagen

PD Dr.-Ing. habil. Sven Tomforde

Universität Kassel
Fachbereich Elektrotechnik / Informatik
Fachgebiet „Intelligent Embedded Systems“

WS 2017/2018

Intelligent Embedded Systems

Agenda

Daten $\longrightarrow$ Wissen

Agenda

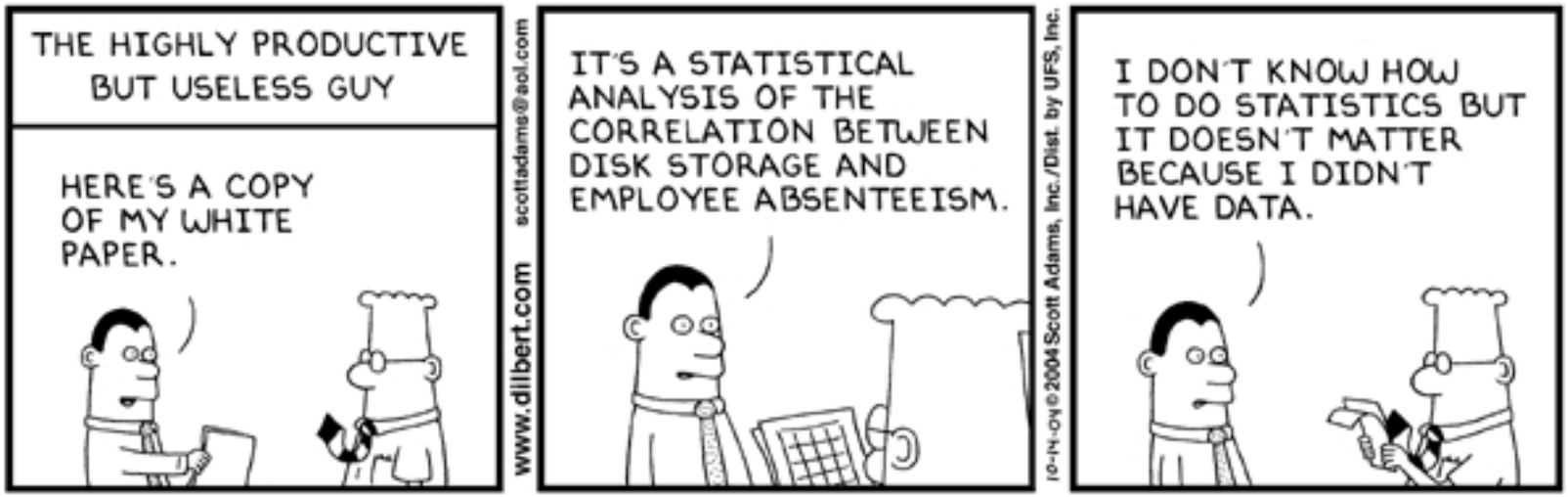
Daten $\longrightarrow$ Wissen

- Daten
- Wissen
- Daten $\longrightarrow$ Wissen
  - Begriffe KDD (Knowledge Discovery in Databases) und DM (Data Mining)
  - Aufgaben des DM
  - Komponenten des DM
  - DM-Prozessmodelle
  - Bewertung von Ergebnissen des DM
- Sonstiges

Intelligent Embedded Systems

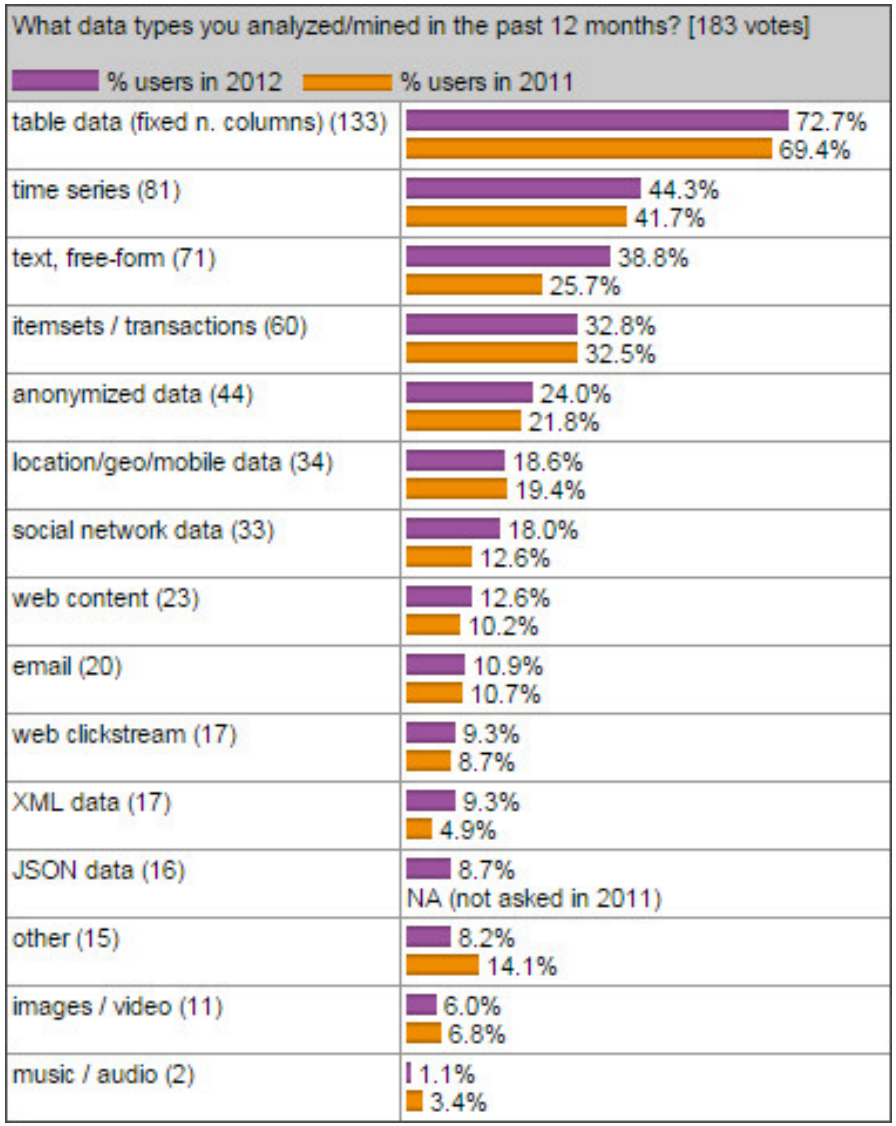
Daten

Daten



Intelligent Embedded Systems

Arten von Daten



Quelle: `http://www.kdnuggets.com/polls/2012/data-types-analyzed-data-mined.html` Stand: 21.10.2014

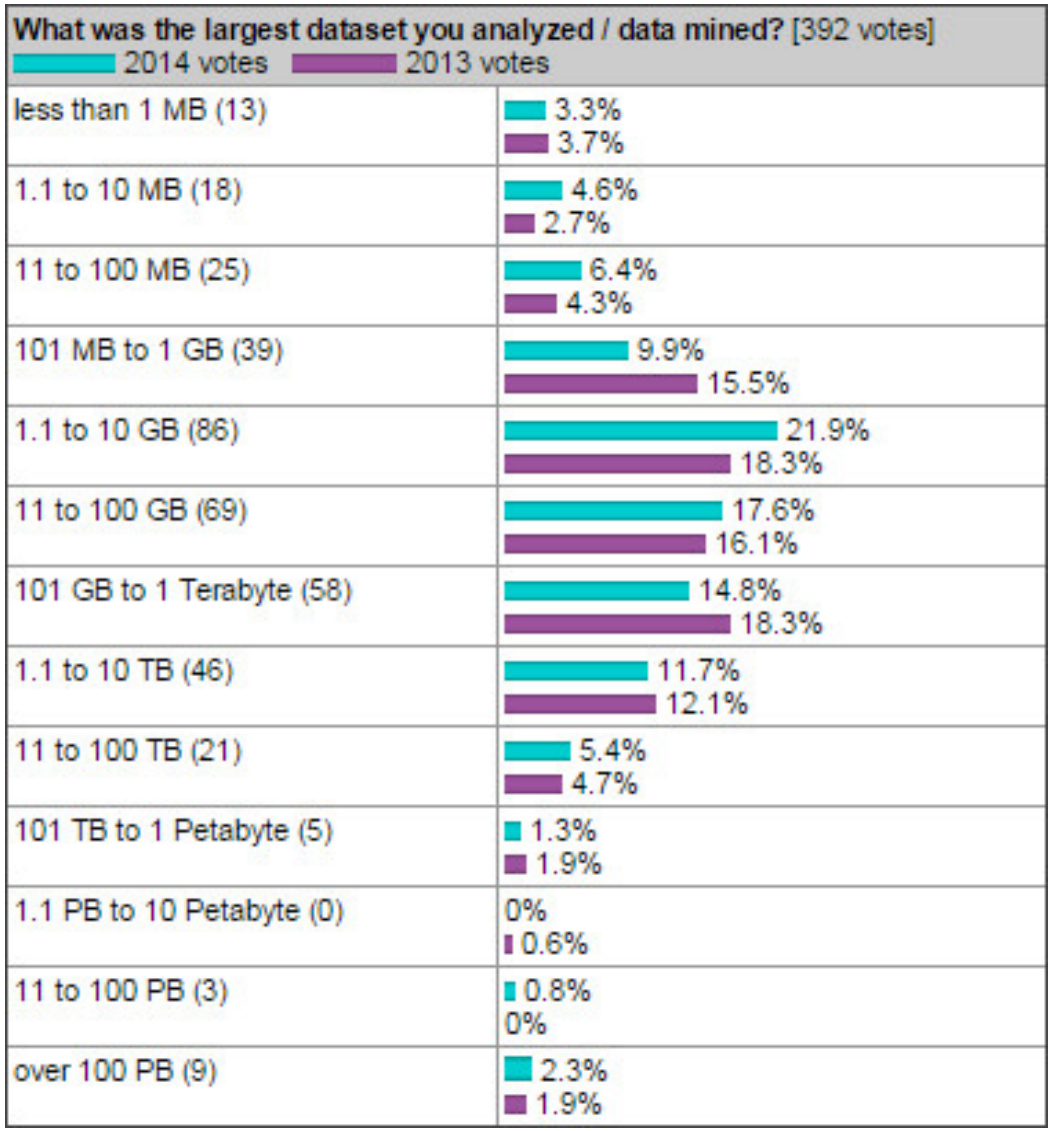
Intelligent Embedded Systems

Daten – Beispiel

Urliste:

ID	Alter	Geschlecht	verheiratet	Ausbildung	Einkommen
58	54	männl.	ja	Diplom	3000
59	?	weibl.	ja	Diplom	10000
60	29	männl.	ja	Abitur	2900
61	9	männl.	nein	Lehre	0
62	85	weibl.	nein	Diplom	5400
63	40	männl.	ja	Diplom	0
64	38	weibl.	nein	Promotion	7500
65	7	männl.	?	keine	630
66	49	männl.	ja	Lehre	4500
67	76	männl.	ja	Abitur	4200

Typische Datenmengen



Quelle: `http://www.kdnuggets.com/polls/2014/largest-dataset-analyzed-data-mined-2014.html` Stand: 21.10.2014

Intelligent Embedded Systems

Begriffe

- Matrix von Beispieldaten: **Datensatz**
- Einzelne Spalte des Datensatzes: **Attribut**, **Feature**, **Merkmal**, Variable oder Feld
- Einzelne Zeile des Datensatzes: **Muster**, **Sample**, Individuum, Instanz, Fall, Objekt, Datum, Record, ...

Vorsicht: Der Begriff „Muster“ wird auch anders verwendet, im Sinne interessanter „Strukturen“ in Datensätzen (vgl. „Mustererkennung“).

Intelligent Embedded Systems

Arten von Attributen – 1

Nominale Attribute haben einen diskreten endlichen Wertebereich ohne Ordnungs-/Präferenzstruktur.

Beispiele:

- Geschlecht (männlich, weiblich),
  d. h. **binäres / dichotomes Attribut**
- Studienfach (BWL, Informatik, Medizin, ...)
- Nationalität (deutsch, österreichisch, britisch, ...)

Intelligent Embedded Systems

Arten von Attributen – 2

Ordinale Attribute haben einen endlichen Wertebereich mit einer Ordnungs-/Präferenzstruktur.

Beispiele:

- Ausbildung (Lehre, Abitur, Diplom, Promotion, ...)
- Härte (Graphit, Kalkstein, Granit, Diamant, ...)

Intelligent Embedded Systems

Arten von Attributen – 3

Intervallgrößen haben eine feste Ordnung und werden auch in gleichen Einheiten gemessen.

Im Allgemeinen gibt es keinen spezifischen Nullpunkt, d. h., Differenzen ergeben Sinn, Vielfache oder Verhältnisse nicht.

Beispiele:

- Datum (Jahreszahl mit willkürlicher Festsetzung des Jahres 0)
- Temperatur (in Grad Celsius oder Fahrenheit)

Intelligent Embedded Systems

Arten von Attributen – 4

Ratiogrößen besitzen im Gegensatz zu Intervallgrößen einen spezifischen Nullpunkt.

Das Berechnen von Verhältnissen von Werten ist sinnvoll.

Beispiele:

- Abstand zweier Objekte
- Einkommen
- Anzahl der Kinder

Intelligent Embedded Systems

Arten von Attributen – 5

Ganzzahlige Attribute können nur ganzzahlige (Integer-)Werte annehmen.

Beispiele: Jahreszahl (Intervallgröße), Anzahl der Kinder (Ratiogröße)

Kontinuierliche Attribute können reelle Werte annehmen.

Beispiele: Temperatur (Intervallgröße), Abstand zweier Objekte (Ratiogröße)

Intelligent Embedded Systems

Missing Values – 1

Bei einigen Mustern können die Werte einzelner Attribute fehlen, sog. **Missing Values**.

Mögliche Ursachen:

- Ausfall eines Sensors bei Messung physikalischer Größen
- Verweigerung einer Auskunft
- irrelevantes Attribut für das betreffende Objekt
  (z. B. schwanger (ja/nein) bei Männern)
- Änderungen in einem Versuchsaufbau
- Zusammenfassen verschiedener Datensätze

Intelligent Embedded Systems

Missing Values – 2

Die Wahrscheinlichkeit, dass der Wert fehlt, kann vom wahren Wert abhängen oder nicht!

Beispiele:

- Ein Temperatursensor liefert keine Werte, weil seine Stromversorgung ausgefallen ist.
- Ein Temperatursensor liefert keine Werte unterhalb des Gefrierpunktes.

Intelligent Embedded Systems

Missing Values – 3

Möglichkeiten für die Behandlung von Missing Values:

- Muster mit Missing Values werden nicht verwendet (nur wenn wenige Muster betroffen, schlecht z. B. bei Zeitreihen).
- Missing Values werden durch das DM-Verfahren selbst berücksichtigt (verfahrensabhängig).
- Missing Values werden geschätzt, z. B. (vgl. Verfahren zur Datenvorverarbeitung!):
  - Verwendung des Mittelwerts
  - Verwendung des häufigsten Werts
  - Schätzung mit Hilfe der Werte anderer Attribute
  - Interpolation bei Zeitreihen
  - ...

Wichtig! Prüfen, ob Ergebnisse des DM verfälscht werden können!!!

Intelligent Embedded Systems

Wissen

Wissensrepräsentation

Wie kann Wissen repräsentiert sein?

Antwort:

Es gibt sehr viele verschiedene Formen der Wissensrepräsentation.

Die Form der Wissensrepräsentation hängt stark von dem verwendeten DM-Algorithmus und von den Zielen des DM-Prozesses ab!

Intelligent Embedded Systems

Beispiel zur Wissensrepräsentation – 1

Datensatz (Wetterdaten):

outlook	temperature	humidity	windy	play game
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Beispiel zur Wissensrepräsentation – 2

Attribute:

- outlook: sunny, overcast, rainy
- temperature: hot, mild, cool
- humidity: high, normal
- windy: false, true
- play game: no, yes

Abhängig von Wetterbedingungen wird ein Spiel im Freien gespielt oder nicht.

Intelligent Embedded Systems

Beispiel zur Wissensrepräsentation – 3

Aufgabe 1: Entscheidung über Spiel, abhängig vom Wetter

Lösung 1: Liste von Entscheidungsregeln

1. IF outlook = sunny AND humidity = high THEN play game = no
2. IF outlook = rainy AND windy = true THEN play game = no
3. IF outlook = overcast THEN play game = yes
4. IF humidity = normal THEN play game = yes
5. IF none of the above THEN play game = yes

Regeln sind in der angegebenen Reihenfolge anzuwenden.

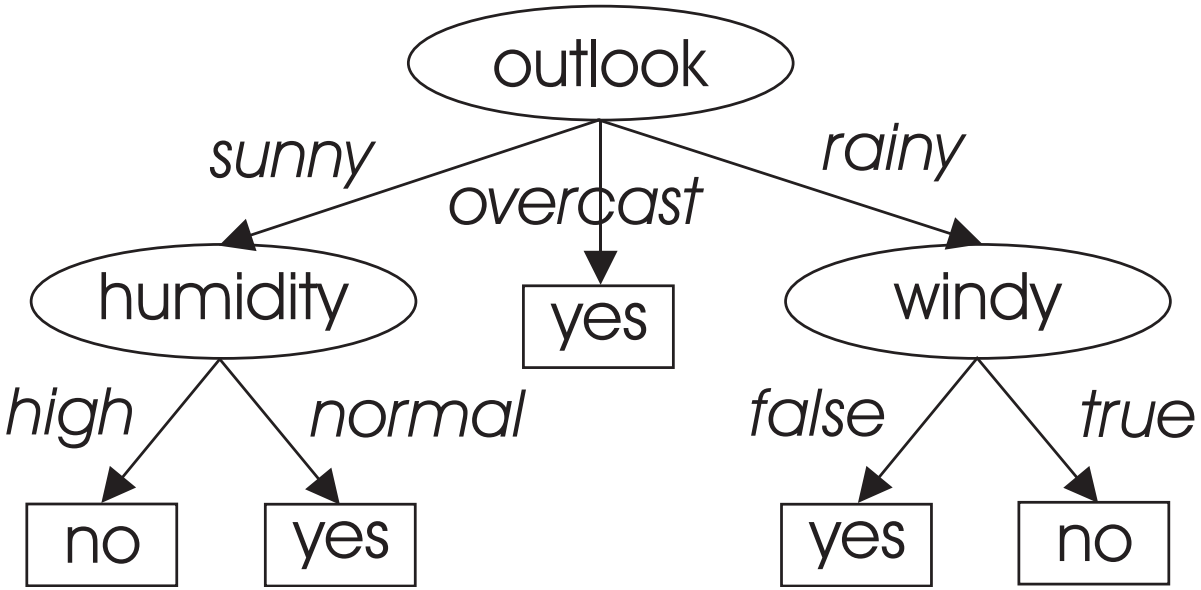
Wissensrepräsentation: Struktur und Reihenfolge der Regeln

Intelligent Embedded Systems

Beispiel zur Wissensrepräsentation – 3

Aufgabe 1: Entscheidung über Spiel, abhängig vom Wetter

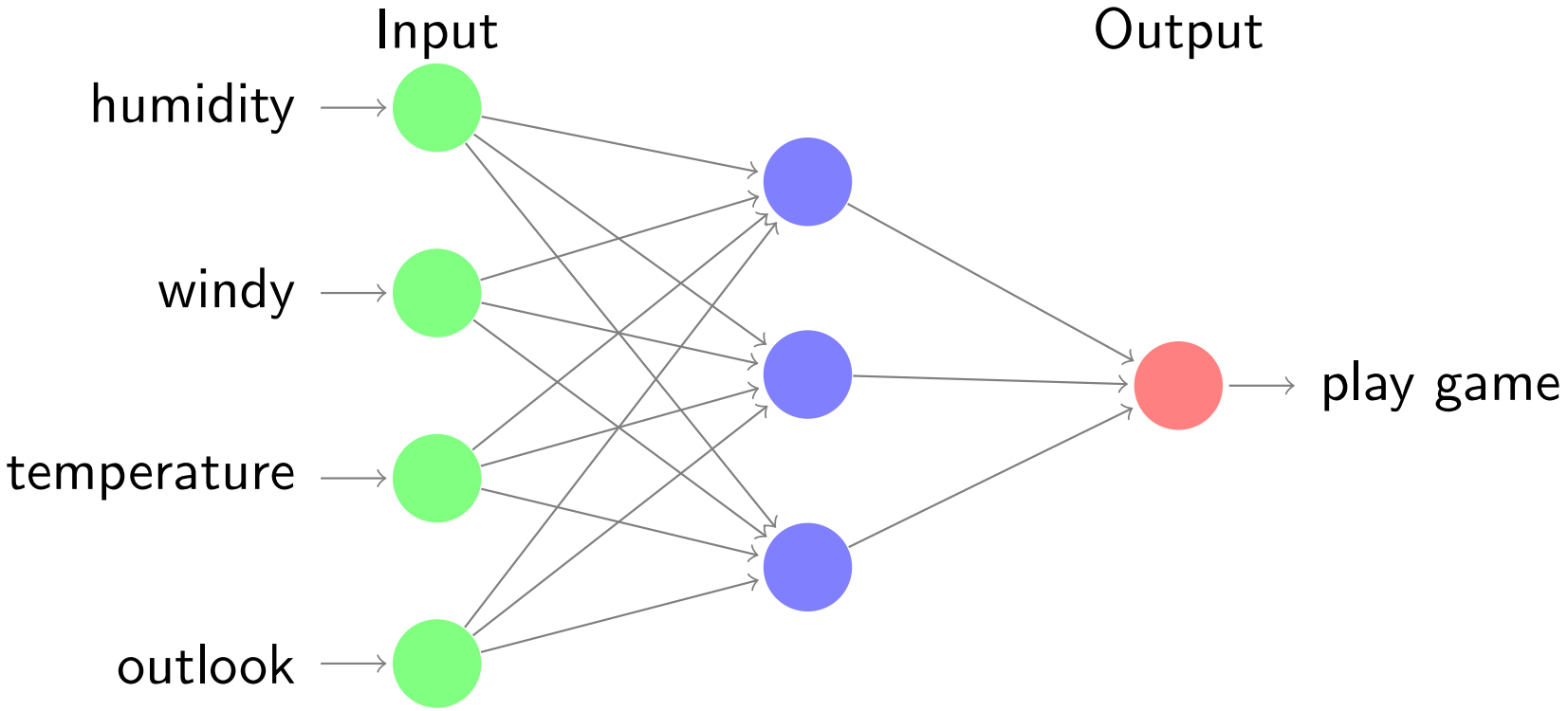
Lösung 2: Entscheidungsbaum



Wissensrepräsentation:
Struktur des Baums

Beispiel zur Wissensrepräsentation – 3

Aufgabe 1: Entscheidung über Spiel, abhängig vom Wetter



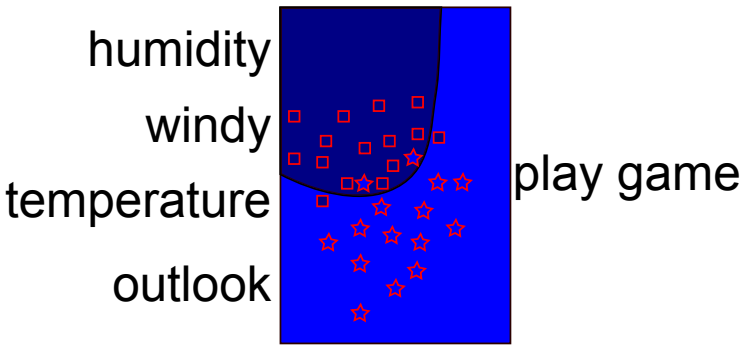
Wissensrepräsentation: Architektur des Netzes und Gewichte

Intelligent Embedded Systems

Beispiel zur Wissensrepräsentation – 4

Aufgabe 1: Entscheidung über Spiel, abhängig vom Wetter

Lösung 4: Support Vector Machine



Wissensrepräsentation: Stützvektoren (support vectors)

Intelligent Embedded Systems

Beispiel zur Wissensrepräsentation – 5

Aufgabe 2: Abhängigkeiten zwischen Attributen feststellen

Lösung: Assoziationsregeln

1. IF temperature = cool THEN humidity = normal
2. IF humidity = normal AND windy = false THEN play game = yes
3. IF outlook = sunny AND play game = no THEN humidity = high
4. IF windy = false AND play game = no THEN outlook = sunny AND humidity = high

Wissensrepräsentation: Struktur der Regeln

Intelligent Embedded Systems

Formen der Wissensrepräsentation

Beispiele:

- Liste von Entscheidungsregeln (Struktur und Reihenfolge der Regeln)
- Entscheidungsbaum (Struktur des Baums)
- Neuronales Netz (Architektur und Gewichte)
- Bayessches Netz (Graphstruktur und Wahrscheinlichkeitstabellen)
- Support Vector Machine (Kernelfunktionen und Support Vektoren)
- u. v. m.

Intelligent Embedded Systems

Bewertung von Wissen – 1

Laut Definition des DM soll das erworbene Wissen folgende Eigenschaften haben:

- **stichhaltig / gültig:** für neue, unbekannte Daten mit gewisser Wahrscheinlichkeit gültig, ist messbar z. B. über Klassifikationsraten
- **neu:** nicht nur für System, sondern auch für Anwender, ist feststellbar durch Vergleich mit *a priori* Wissen
- **nützlich:** natürlich von Aufgabenstellung abhängig, ist messbar z. B. über finanziellen Gewinn
- **verständlich:** zumindest nach Vorverarbeitung, ist schwer messbar (z. B. Anzahl verknüpfter Variablen in Regeln)

interessant: Summe gewichteter Einzelkriterien, je nach Anwendungsfall

Intelligent Embedded Systems

Von Daten zum Wissen

Daten $\longrightarrow$ Wissen

Intelligent Embedded Systems

Definition KDD

Definition KDD

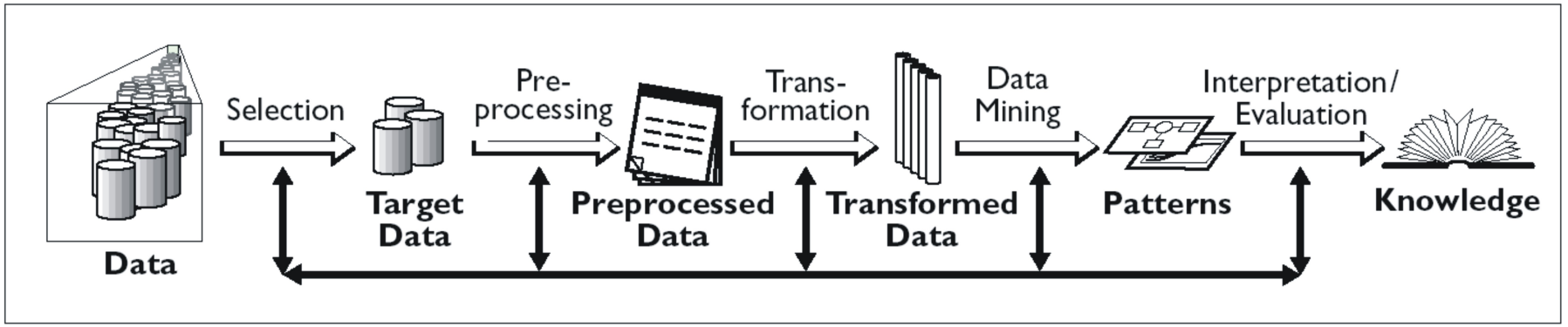
KDD (knowledge discovery in databases): Der gesamte Prozess der Identifikation von stichhaltigem, neuem, potentiell nützlichem und auch verständlichem Wissen aus Daten.

Definition KDD

Definition KDD

KDD (knowledge discovery in databases): Der gesamte Prozess der Identifikation von stichhaltigem, neuem, potentiell nützlichem und auch verständlichem Wissen aus Daten.

Schritte:



[Fayyad, Piatetsky-Shapiro und Smyth 1996]

Intelligent Embedded Systems

Data Mining – 2

Schritte des KDD:

- **Vorbereitung:** Verstehen des Anwendungsfelds und des Problems (Definition der Aufgabe, Festlegung der Erfolgskriterien, Einbeziehung von Vorwissen, ...);
  Bereitstellung von Rohdaten (experimentelle Untersuchungen, Datenbanken, Interviews, physikalische Messungen, ...);
  Verstehen der Daten (Überblick über alle zentral oder dezentral vorhandenen Daten, Klärung des Zugriffs auf die Daten, ...)
- **Datenselektion:** Auswahl einer Datenmenge (Teilmenge von Samples und / oder Attributen, auf denen KDD durchgeführt werden soll)

Intelligent Embedded Systems

Data Mining – 3

- **Vorverarbeitung der Daten:** Aufbereitung der Daten durch Sichtung und Behandlung fehlerbehafteten oder fehlenden Datenmaterials (Identifizierung und Eliminierung von Ausreißern bzw. Rauschen); Entscheidung über Datenrepräsentation (z. B. Variablentypen, Darstellung von fehlenden bzw. unbekannten Daten)
- **Datentransformation:** Datenreduktion und Datenprojektion mit dem Ziel der Verdichtung relevanter Informationen in einer geringeren Zahl von Variablen (Dimensionsreduktion) und Identifikation relevanter Attribute (Merkmalsselektion)

Intelligent Embedded Systems

Data Mining – 4

- **Data Mining:** Modellbildung durch Selektion eines Modells (Paradigmas), Parametrisierung des Modells, Anwendung eines geeigneten DM- Algorithmus, Wissensfindung entsprechend den Zielen des KDD Prozesses
- **Interpretation und Evaluation der Ergebnisse:** Soll-Ist-Vergleich mit kritischer Bewertung der Resultate des Data Mining, eventuell Hinzunahme weiterer Daten und / oder Verfeinerung des KDD Prozesses (Iteration), Visualisierung von Ergebnissen usw.
- **Anwendung des Wissens:** Umsetzung der Data Mining Ergebnisse in die Praxis, z. B. Integration in die täglichen Geschäftsabläufe, ggf. Entwicklung einer individuellen Lösung in Form von Spezialsoftware.

Intelligent Embedded Systems

Data Mining – 5

Definition DM

DM (data mining): Suche von interessantem Wissen (Modelle, Strukturen, Regeln, Mustern, ...) in einer gegebenen Datenmenge durch Anwendung von Algorithmen zur Datenanalyse.

Andere Begriffe für Data Mining: exploratory data analysis, data driven discovery, deductive learning, knowledge extraction, information discovery, information harvesting, data archeology, data pattern processing, ...

KDD und DM werden heute oft synonym verwendet!

Intelligent Embedded Systems

Aufgaben des Data Mining – 1

Deskriptives DM

- Beschreibung der Gesamtheit der Daten oder des Entstehungsprozesses der Daten
- Betrachtung / Schätzung der Wahrscheinlichkeitsverteilungen einzelner Attribute oder gemeinsamer Verteilungen der Attribute
- Gruppierung der Daten, z. B. durch Clusteranalyse

Aufgaben des Data Mining – 2

Exploratives DM

- Analyse oder Sichtung der Daten zur Gewinnung allgemeiner Erkenntnisse ohne vorgegebenes Ziel
- Häufig Anwendung von Visualisierungstechniken
- z. B. Berechnung von Kovarianzen oder Korrelationskoeffizienten von Attributen

Intelligent Embedded Systems

Aufgaben des Data Mining – 3

Prädiktives DM

- Vorhersage des Wertes eines Attributes aus den Werten anderer Attribute (Vorhersage hier nicht notwendigerweise zeitlich zu Verstehen, in diesem Fall spricht man im englischen von “Forecasting”)
- Klassifikation (vorherzusagendes Attribut ist kategorisch, z. B. dichotom)
- Regression (vorherzusagendes Attribut ist z. B. reellwertig)

Intelligent Embedded Systems

Aufgaben des Data Mining – 4

Weitere Beispielaufgaben:

- **Content Retrieval**: zu einem gegebenen, interessanten Muster sollen gleichartige bzw. ähnliche Muster gefunden werden
  - Dokumentensuche im Web nach Schlüsselwörtern
  - Auffinden von Bildern mit bestimmten Inhalten
  - Identifikation von Musikstücken
- **Ausreißererkennung**: Atypische Muster sollen erkannt werden
  - Indentifikation von Angriffen in Rechnernetzen
  - Erkennen von fehlerbehafteten Mustern
- **Regelerkennung**: Erkennung von Assoziationen zwischen Attributen
  - Warenkorbanalyse für Verkaufs- und Marketingstrategien

Intelligent Embedded Systems

Klassifikation – 1

- Datenpunkte in einer Beispielmenge $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ jedes $\mathbf{x}_n$ ist ein D-dimensionaler Vektor, oft wird einfach angenommen $\mathbf{x}_n \in \mathbb{R}^D$ (mit $D \in \mathbb{N}^+$).
- Bei einer Klassifikationsaufgabe muss ein Punkt im Eingaberaum des Klassifikators $\mathbf{x} \in \mathbb{R}^D$ einer Kategorie (Klasse) zugeordnet werden. Es gibt $\mathcal{C}$ Klassen ($\mathcal{C} \in \mathbb{N}^+, \mathcal{C} \geq 2$). Oft hat eine Klasse eine Nummer $c \in \{1, \ldots, \mathcal{C}\}$, was aber keine Ordnung auf der Klasse implizieren soll.
- Für viele Verfahren muss die Klassennummer geeignet numerisch codiert werden, z. B. bei binären Klassifikationsproblemen ($\mathcal{C} = 2$) durch $\{0, 1\}$ oder $\{-1, 1\}$.

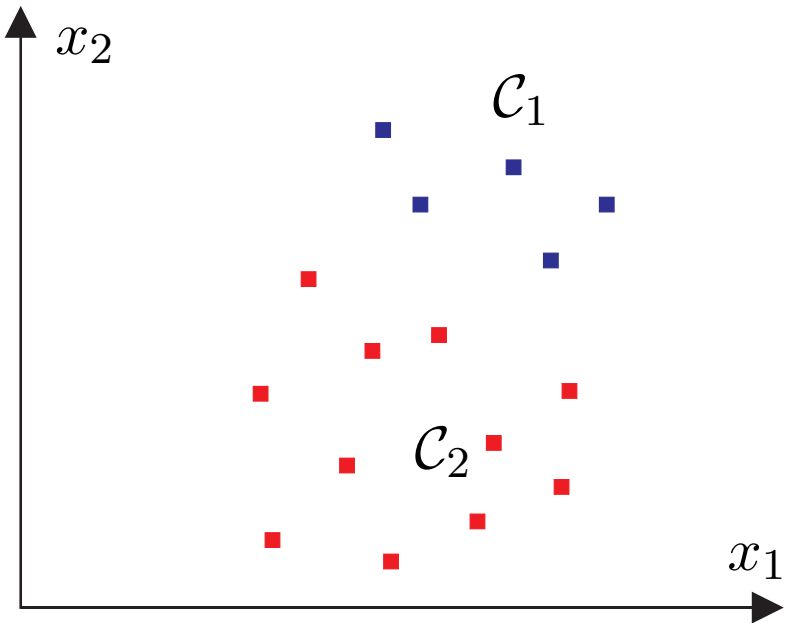
Intelligent Embedded Systems

Klassifikation – 2

- Ein Klassifikator wird häufig mit Hilfe einer Beispieldatenmenge gefunden ("trainiert"), bei der für jede Eingabe, z. B. für $\mathbf{x}_n \in \mathbb{R}^D$, eine korrekte Ausgabe, z. B. $t_n \in \{-1, 1\}$ vorliegt, d. h. insgesamt eine Beispieldatenmenge $\mathbf{X} = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$.
- Bei der Klassifikation werden oft nur binäre Klassifikationsprobleme betrachtet, da eine Lösung für ein Mehrklassenklassifikationsproblem auf Lösungen für mehrere binäre Klassifikationsprobleme zurückgeführt werden kann.

Beispiel: Problem der Klassifikation – 1

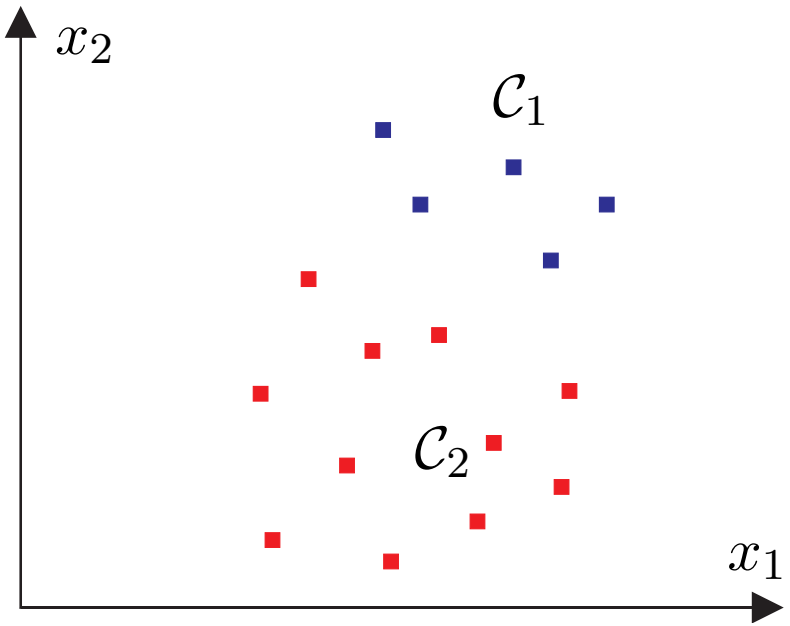
Beispiel: Problem der Klassifikation – 1



gegeben:
binäre Klassifikationsaufgabe mit Klassen $\mathcal{C}_1$ und $\mathcal{C}_2$

Intelligent Embedded Systems

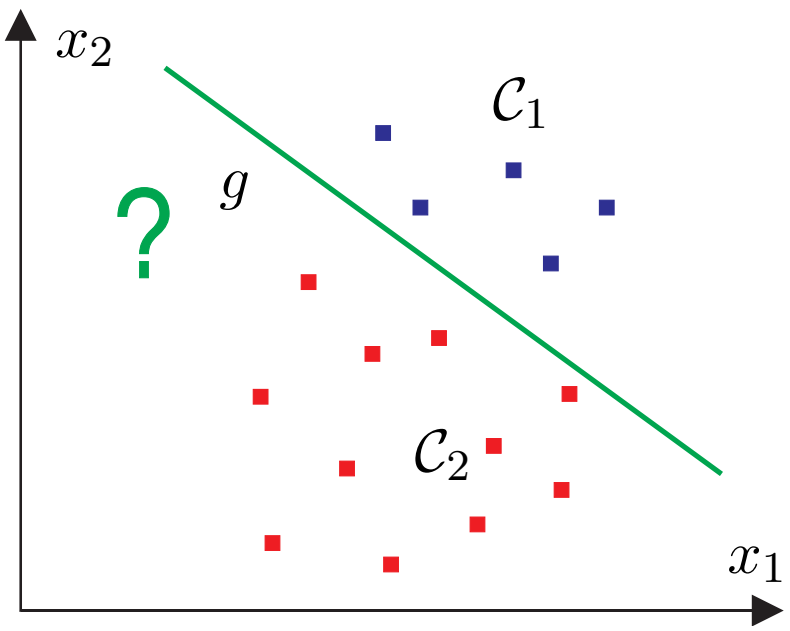
Beispiel: Problem der Klassifikation – 1



gegeben:
binäre Klassifikationsaufgabe mit Klassen $\mathcal{C}_1$ und $\mathcal{C}_2$
Datenpunkte (Muster) $\mathbf{x}_n \in \mathbb{R}^D$
$(n = 1, 2, \ldots, N$ und $D \in \mathbb{N}^+)$
mit Klassenzugehörigkeit (Label)
$t_n \in \{-1, +1\}$
$(+1$ für $\mathcal{C}_1$ und -1 für $\mathcal{C}_2)$.

Intelligent Embedded Systems

Beispiel: Problem der Klassifikation – 1



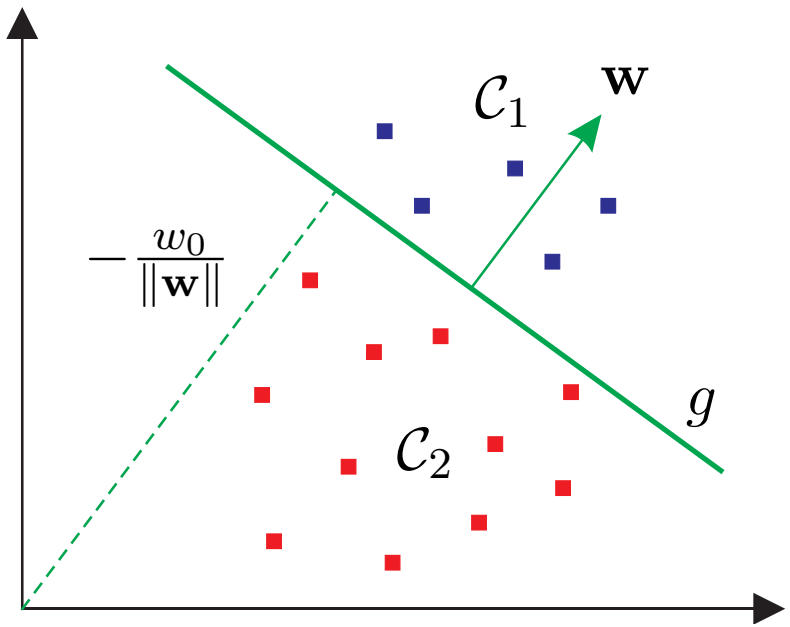
gegeben:
binäre Klassifikationsaufgabe mit Klassen $\mathcal{C}_1$ und $\mathcal{C}_2$
Datenpunkte (Muster) $\mathbf{x}_n \in \mathbb{R}^D$
($n = 1, 2, \ldots, N$ und $D \in \mathbb{N}^+$)
mit Klassenzugehörigkeit (Label)
$t_n \in \{-1, +1\}$
($+1$ für $\mathcal{C}_1$ und -1 für $\mathcal{C}_2$).

gesucht:
Hyperebene g (hier: Gerade)
zur **linearen Separierung**
der beiden Klassen.

Intelligent Embedded Systems

Beispiel: Problem der Klassifikation – 2

Beispiel: Problem der Klassifikation – 2



Beschreibung von g durch einen Normalenvektor $\mathbf{w} \in \mathbb{R}^D$ und einen Bias $w_0 \in \mathbb{R}$, so dass die Funktion

$$g(\mathbf{x}_n) = \mathbf{w}^T \mathbf{x}_n + w_0$$

zur Separierung verwendet werden kann:

$$\mathbf{x}_n \to \mathcal{C}_1 \quad \text{wenn} \quad g(\mathbf{x}_n) \geq 0$$
$$\mathbf{x}_n \to \mathcal{C}_2 \quad \text{wenn} \quad g(\mathbf{x}_n) < 0$$

Intelligent Embedded Systems

Beispiel: Problem der Klassifikation – 3

Wie wird g gefunden?

Verschiedene Ideen werden in der Vorlesung besprochen!

- Perzeptron-Lernen
- Lösen eines linearen Ausgleichsproblems
- Anwendung des Fisher-Kriteriums
- Support Vector Machine mit Standardskalarprodukt als Kernel-Funktion

Intelligent Embedded Systems

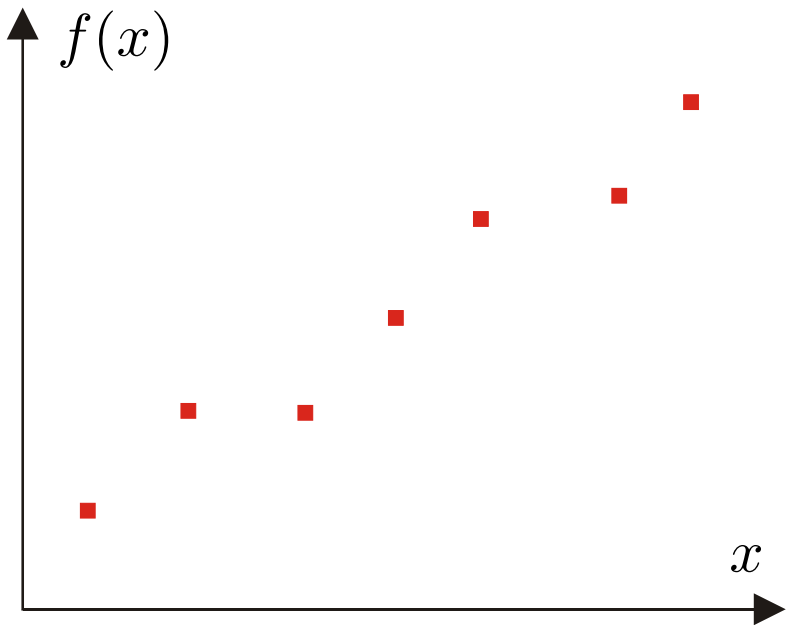
Regression

- Bei der Regression hat man im Gegensatz zur Klassifikation als Ausgabe keine Kategorie (Klasse), sondern einen numerischen Wert.
- Eine unbekannte Funktion f mit $f : \mathbb{R}^D \rightarrow \mathbb{R}^E$ wird durch eine Beispielmenge $\mathbf{X} = \{(\mathbf{x}_1, \mathbf{t}_1), \ldots, (\mathbf{x}_N, \mathbf{t}_N)\}$ mit $\mathbf{x}_n \in \mathbb{R}^D, \mathbf{t}_n \in \mathbb{R}^E (n \in 1, \ldots, N), D, E \in \mathbb{N}^+$ beschrieben.
- Aufgabe ist, für eine neue Eingabe $\mathbf{x} \in \mathbb{R}^D$ den korrekten Funktionswert zu bestimmen.
- Oft wird nur der Fall $E = 1$ betrachtet und der Fall $E \geq 2$ auf mehrere unabhängige, einfache Regressionsprobleme zurückgeführt.

Intelligent Embedded Systems

Beispiel: Problem der Regression – 1

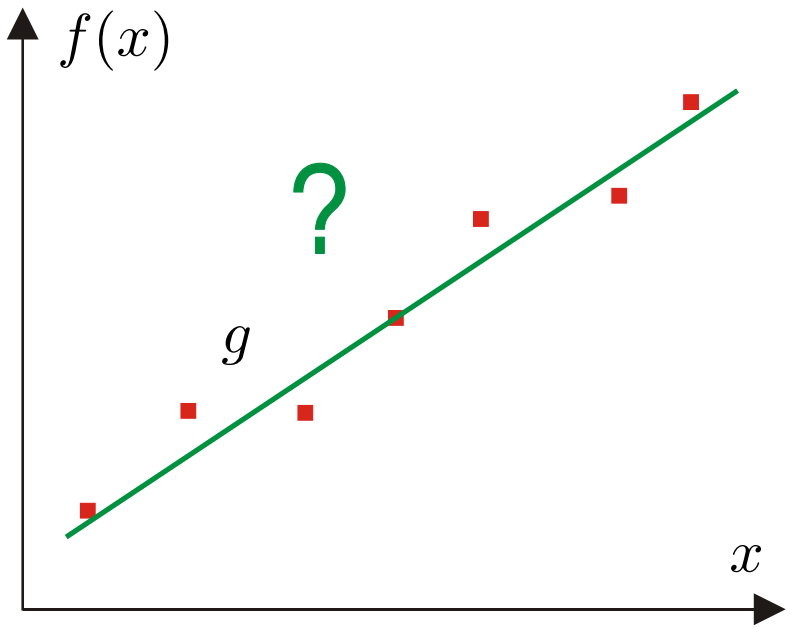
Beispiel: Problem der Regression – 1



gegeben:
Regressionsaufgabe mit Datenpunkten (Mustern)
$(\mathbf{x}_n, f(\mathbf{x}_n)) \in \mathbb{R}^2$
$(n = 1, 2, \ldots, N)$

Intelligent Embedded Systems

Beispiel: Problem der Regression – 1



gegeben:
Regressionsaufgabe mit Datenpunkten (Mustern)
$(\mathbf{x}_n, f(\mathbf{x}_n)) \in \mathbb{R}^2$
$(n = 1, 2, \ldots, N)$

gesucht:
Hyperebene g (hier: Gerade), welche die durch die Daten beschriebene Funktion gut modelliert.
$g = a \cdot x + b$

Intelligent Embedded Systems

Beispiel: Problem der Regression – 2

Wie wird g gefunden?

Beispiel: Problem der Regression – 2

Wie wird g gefunden?

Mögliche Lösung:
Mit $g(x) = a \cdot x + b$ $(a, b \in \mathbb{R})$:
Wähle a und b so, dass

$$\mathcal{E}(a, b) = \sum_{n=1}^{N} \left(g(\mathbf{x}_n) - f(\mathbf{x}_n)\right)^2$$

minimal wird. (sog. least-squares Fehler)

(Lösen eines linearen Ausgleichsproblems (Euklidisches Fehlermaß): quadratmitteloptimale Lösung, Methode der kleinsten Quadrate)

Intelligent Embedded Systems

Data Mining Algorithmen – 1

Gemeinsame Komponenten:

- **Modell (Paradigma)**: Data Mining Algorithmen finden eine Modellinstanz (bzw. die Parameter eines Modells), das grundlegende Charakteristika (Strukturen, Regeln, ...) der Daten beschreibt.
- **Bewertungsfunktion**: Kriterien und Testverfahren werden benötigt, um die Güte von Modellen zu bewerten.
- **Suchalgorithmus**: Data Mining Algorithmen durchsuchen die gegebenen Daten unter Verwendung der Bewertungsfunktion in geeigneter Weise, um Modelle / Parameter zu finden.

Ein Data Mining Algorithmus ist typischerweise eine bestimmte Instanz dieser drei Komponenten.

Intelligent Embedded Systems

Data Mining Algorithmen – 2

Beispiele für Modelle: wurden schon viele besprochen!

- Liste von Entscheidungsregeln (Struktur und Reihenfolge der Regeln)
- Entscheidungsbaum (Struktur des Baums)
- Neuronales Netz (Architektur und Gewichte)
- Assoziationsregeln (Struktur der Regeln)
- Bayessches Netz (Graphstruktur und Wahrscheinlichkeitstabellen)
- Support Vector Machine (Kernelfunktionen und Support Vektoren)
- ...

Intelligent Embedded Systems

Data Mining Algorithmen – 3

Beispiele für Bewertungsfunktionen:

- mittlere absolute Fehler
- mittlere quadratische Fehler
- Klassifikationsraten / Fehlklassifikationsraten
- False Positives, False Negatives, True Positives, True Negatives und Kombinationen daraus
- ...

... abhängig von Aufgabenstellung und Modell

Intelligent Embedded Systems

Data Mining Algorithmen – 4

Beispiele für Suchalgorithmen:

- Lineare Optimierungsverfahren, z. B. Lösen linearer Ausgleichsprobleme
- Nicht-lineare Optimierungsverfahren, z. B.
  - Methoden nullter Ordnung (benutzen Zielfunktion, aber nicht deren (partielle) Ableitung): Random Search, Hillclimbing, Simulated Annealing, Evolutionäre Algorithmen, ...
  - Methoden erster Ordnung (benutzen Zielfunktion und (partielle) Ableitung der Zielfunktion): Gradientenabstieg, ...
  - Methoden zweiter Ordnung (benutzen zusätzlich die zweite (partielle) Ableitung): Konjugierte Gradienten, Newton-Methode, ...

... abhängig von Aufgabenstellung, Modell und Bewertungsfunktion

Intelligent Embedded Systems

Verwandte Gebiete – Data Warehousing

A data warehouse is a subject oriented, integrated, non-volatile, and time-variant collection of data in support of management's decisions.

Inmon

Data Warehousing beinhaltet

- die Verwaltung großer, sich mit der Zeit ändernder Datenmengen,
- die aus verschiedenen Quellen/Datenbanken stammen können,
- sowie Techniken zur Aufbereitung und zum komfortablen Umgang eines Benutzers mit den Daten.

OLAP (online analytical processing): Tools (oft interaktiv) für multidimensionale Datenanalyse in Datenbanken

Intelligent Embedded Systems

Verwandte Gebiete – Anfragen in Datenbanken

Wesentliche Unterschiede beim Data Mining:

- Die gewünschten Anfragen können mit gängigen Sprachen (z. B. SQL) nicht gestellt werden.
- Oft ist bei Beginn eines Data Mining Prozesses nicht klar, welche Art von Ergebnis erwartet werden kann.
- Daten können oft erst nach geeigneter Vorverarbeitung bzw. Transformation für Data Mining verwendet werden.
- Ausgabe des Data Mining Prozesses ist meist nicht Teilmenge der Datenbank, sondern Ergebnis einer Analyse des Datenbankinhalts.

Intelligent Embedded Systems

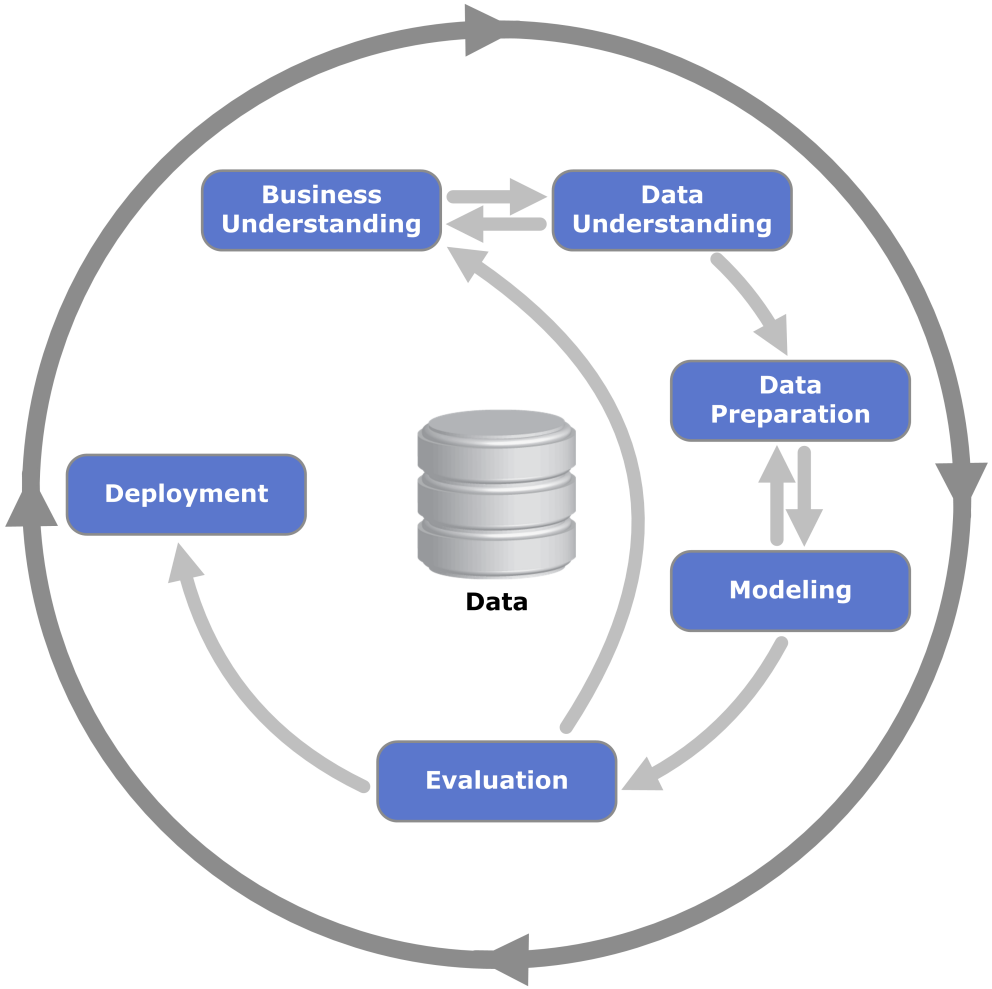
CRISP Data Mining Prozessmodell – 1

CRoss-Industry Standard Process for Data Mining:

- Prozessmodell für Data Mining, beschreibt Lebenszyklus eines Data Mining Projekts
- Web, siehe *www.crisp-dm.eu*
- Begonnen 1996 u. a. durch DaimlerChrysler und SPSS, wird aber wohl nicht weiter unterstützt
- Unabhängig von Firmen, Werkzeugen und Anwendern

Intelligent Embedded Systems

CRISP Data Mining Prozessmodell – 2



Quelle: http://upload.wikimedia.org/wikipedia/commons/b/b9/CRISP-DM_Process_Diagram.png Stand: 27.10.2014

Intelligent Embedded Systems

CRISP Data Mining Prozessmodell – 3

Aus der Beschreibung von CRISP:

- **Business Understanding:** This initial phase focuses on understanding the project objectives and requirements from a business perspective, and then converting this knowledge into a data mining problem definition, and a preliminary plan designed to achieve the objectives.
- **Data Understanding:** The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data, or to detect interesting subsets to form hypotheses for hidden information.

CRISP Data Mining Prozessmodell – 4

- **Data Preparation:** The data preparation phase covers all activities to construct the final dataset (data that will be fed into the modeling tool(s)) from the initial raw data. Data preparation tasks are likely to be performed multiple times, and not in any prescribed order. Tasks include table, record, and attribute selection as well as transformation and cleaning of data for modeling tools.
- **Modeling:** In this phase, various modeling techniques are selected and applied, and their parameters are calibrated to optimal values. Typically, there are several techniques for the same data mining problem type. Some techniques have specific requirements on the form of data. Therefore, stepping back to the data preparation phase is often needed.

Intelligent Embedded Systems

CRISP Data Mining Prozessmodell – 5

- **Evaluation:** At this stage in the project you have built a model (or models) that appears to have high quality, from a data analysis perspective. Before proceeding to final deployment of the model, it is important to more thoroughly evaluate the model, and review the steps executed to construct the model, to be certain it properly achieves the business objectives. A key objective is to determine if there is some important business issue that has not been sufficiently considered. At the end of this phase, a decision on the use of the data mining results should be reached.

CRISP Data Mining Prozessmodell – 6

- **Deployment:** Creation of the model is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it. Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process. In many cases it will be the customer, not the data analyst, who will carry out the deployment steps. However, even if the analyst will not carry out the deployment effort it is important for the customer to understand up front what actions will need to be carried out in order to actually make use of the created models.

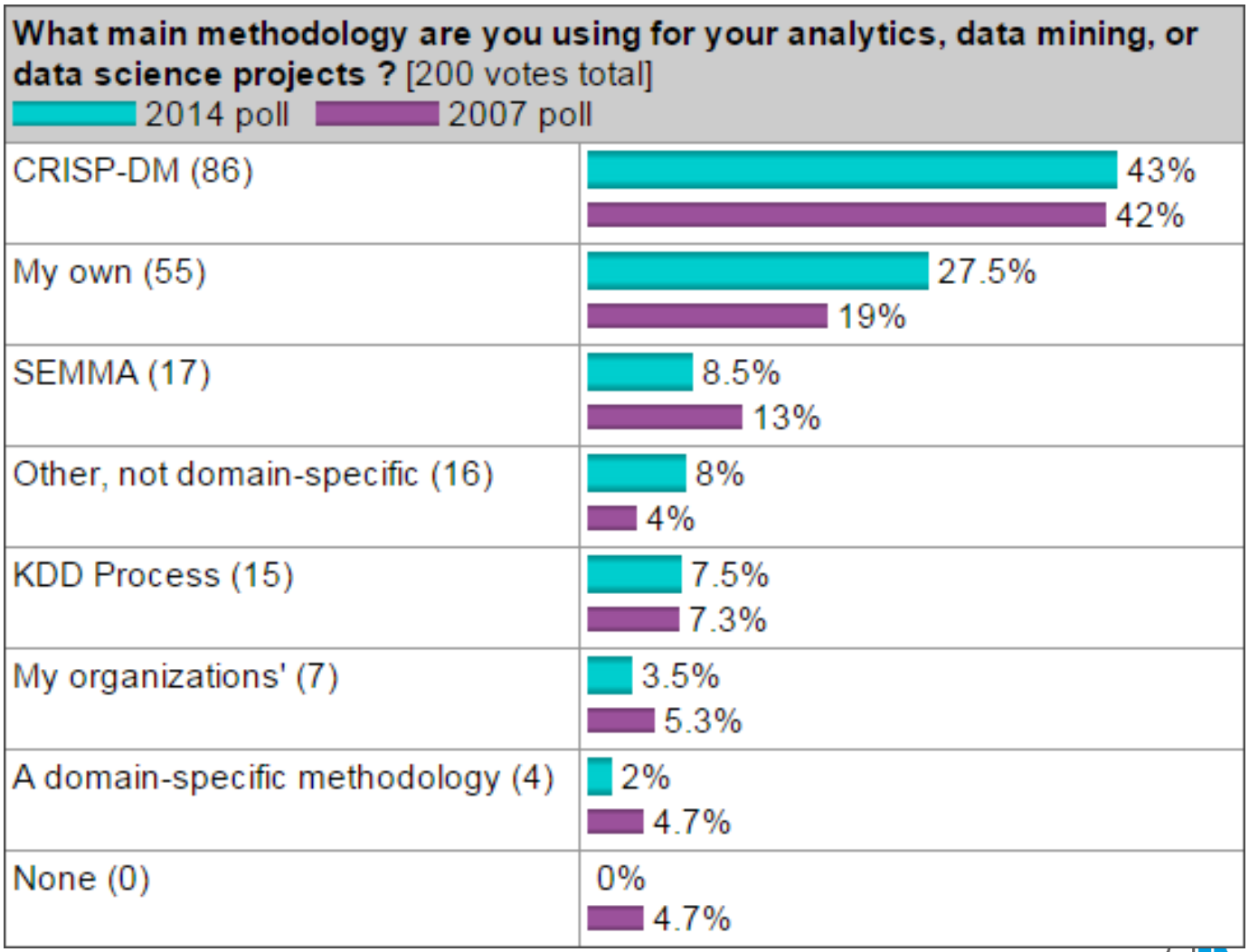
SEMMA Data Mining Prozessmodell

SEMMA (sample, explore, modify, model, assess):

- Prozessmodell für Data Mining, beschreibt ebenfalls Ablauf eines Data Mining Prozesses
- siehe *www.sas.com*
- von SAS speziell für den SAS Enterprise Miner

Intelligent Embedded Systems

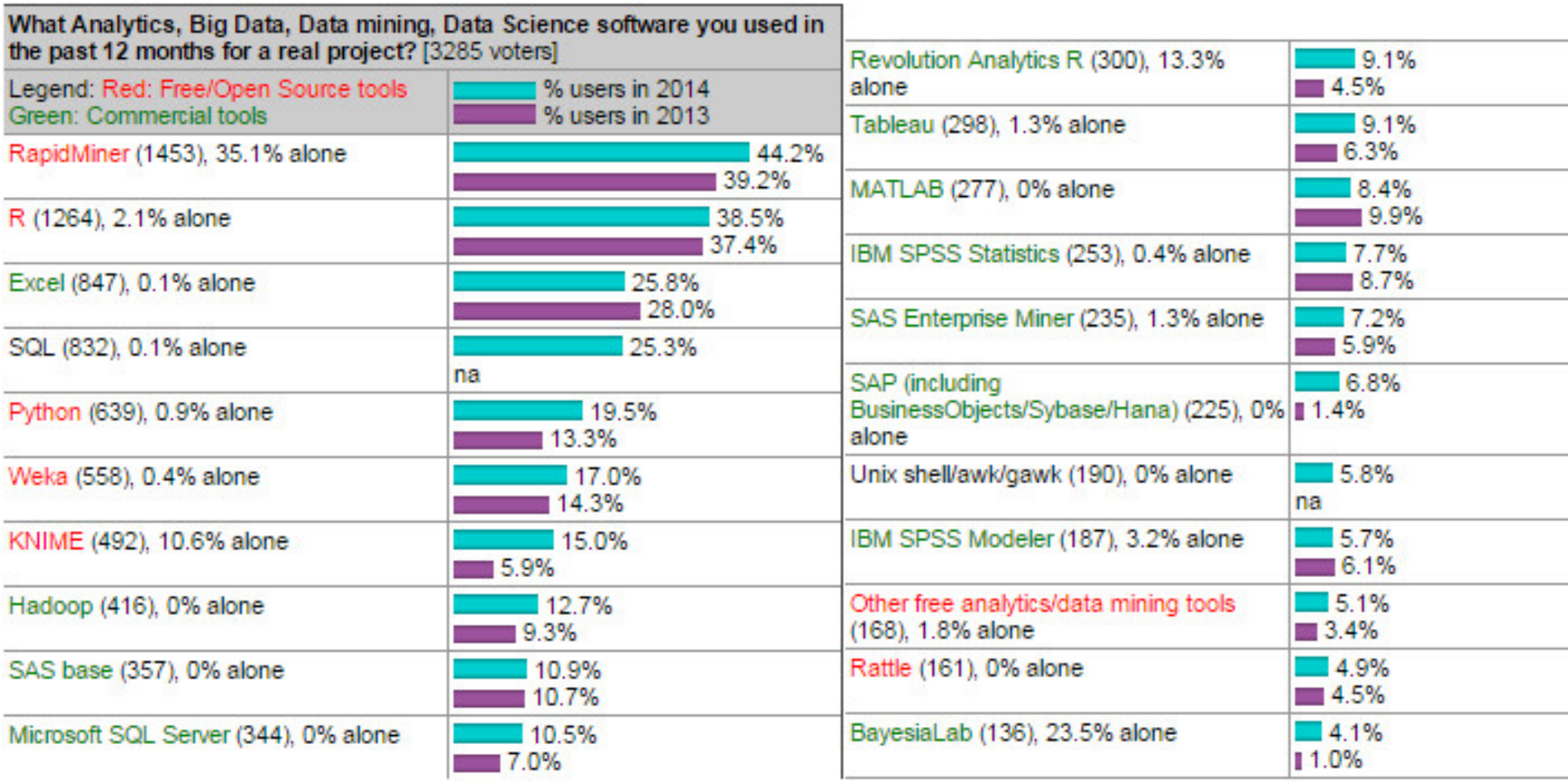
Prozessmodelle in der Praxis



Quelle: `http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-methodology.html` (Stand: 16.12.2014 – 8:50)

Intelligent Embedded Systems

DM-Tools in der Praxis



Quelle: `http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-software-used.html`

(Stand: 24.10.2014)

Intelligent Embedded Systems

Bewertung von Ergebnissen – 1

Generalisierungsfähigkeit:

Eine Modellinstanz, deren Parameter mit einem Suchalgorithmus und einer Bewertungsfunktion eingestellt wurden, soll nicht nur für den Datensatz, mit dem die Parameter eingestellt wurden, geringe Fehler liefern, sondern auch für (alle) anderen Datensätze, denen dieselbe zu modellierende Funktion (oder Aufgabenstellung) zugrunde liegt.

Ziel:

Eine Modellinstanz soll in einer späteren Anwendung auf neuen (sog. „unbekannten“ Daten) gute Ergebnisse liefern.

Intelligent Embedded Systems

Bewertung von Ergebnissen – 2

Begriffe:

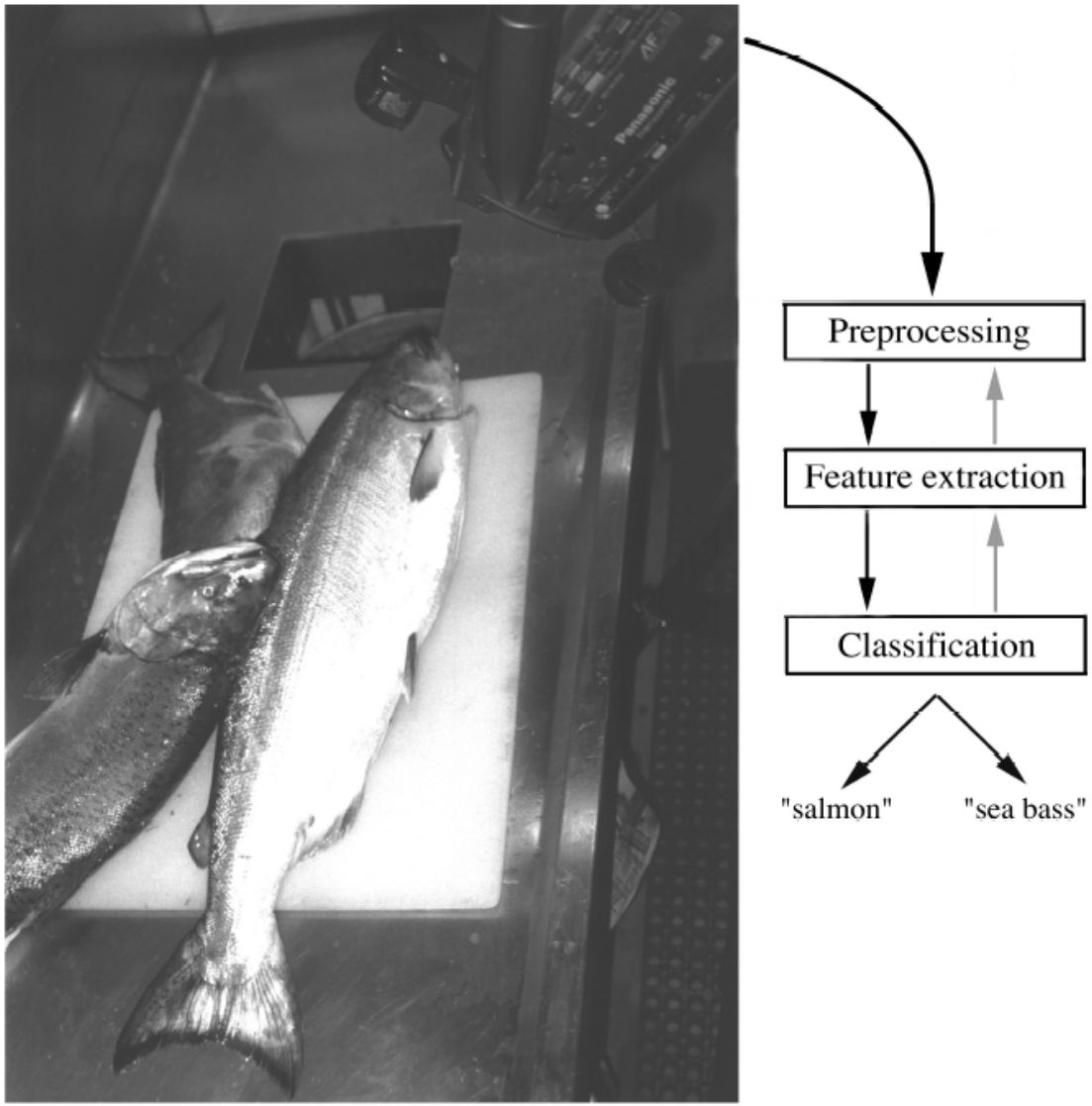
- **Trainingsdaten**: Daten, mit denen die Modellinstanz erstellt wird.
- **Testdaten**: unbekannte Daten, mit denen die Modellinstanz vor ihrer Anwendung getestet wird, um die Generalisierungsfähigkeit zu bewerten.

Beobachtungen:

1. Im Allgemeinen liefert eine Modellinstanz für Testdaten höhere Fehler als für Trainingsdaten.
2. Je komplexer ein Modell ist, d. h., je mehr Parameter (Freiheitsgrade) es hat, um so kleiner wird im Allgemeinen der Trainingsfehler und um so größer wird der Testfehler.

Intelligent Embedded Systems

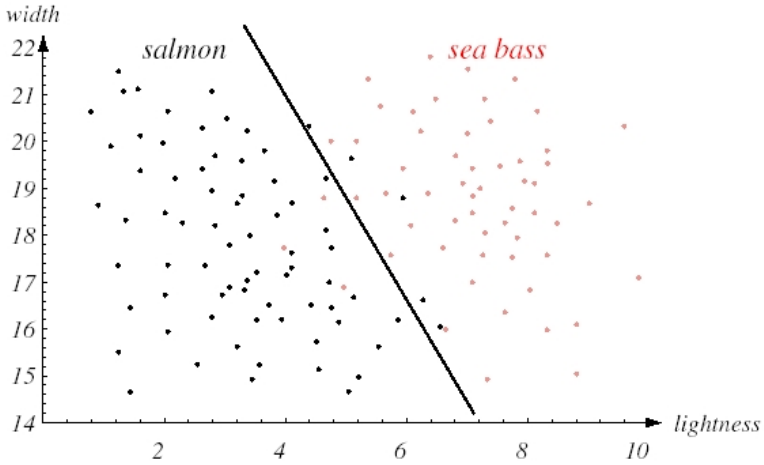
Bewertung von Ergebnissen – 3



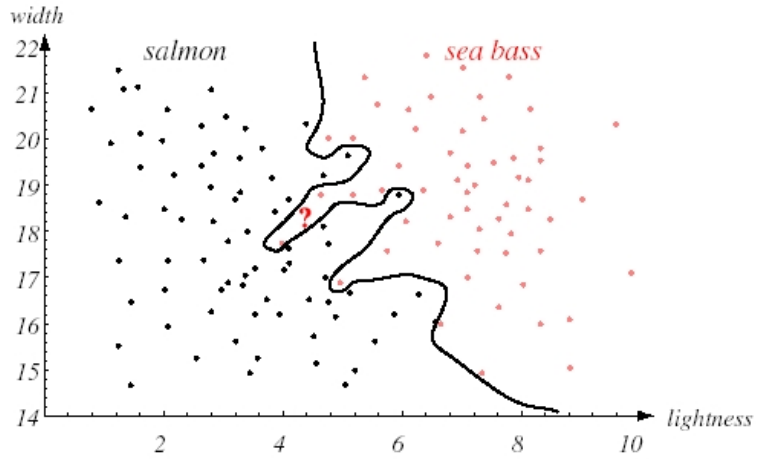
Quelle: [Duda, Hart, Stork, 2001]

Intelligent Embedded Systems

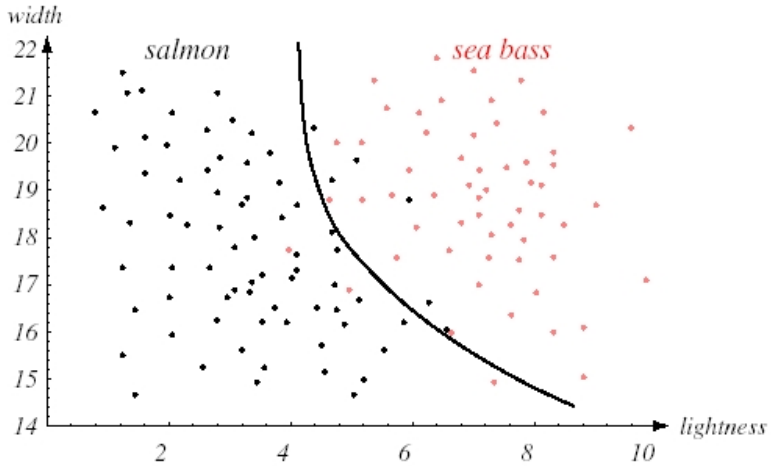
Bewertung von Ergebnissen – 4



Einfaches Modell, einfache Decision Boundary (Versuch der linearen Separierung)



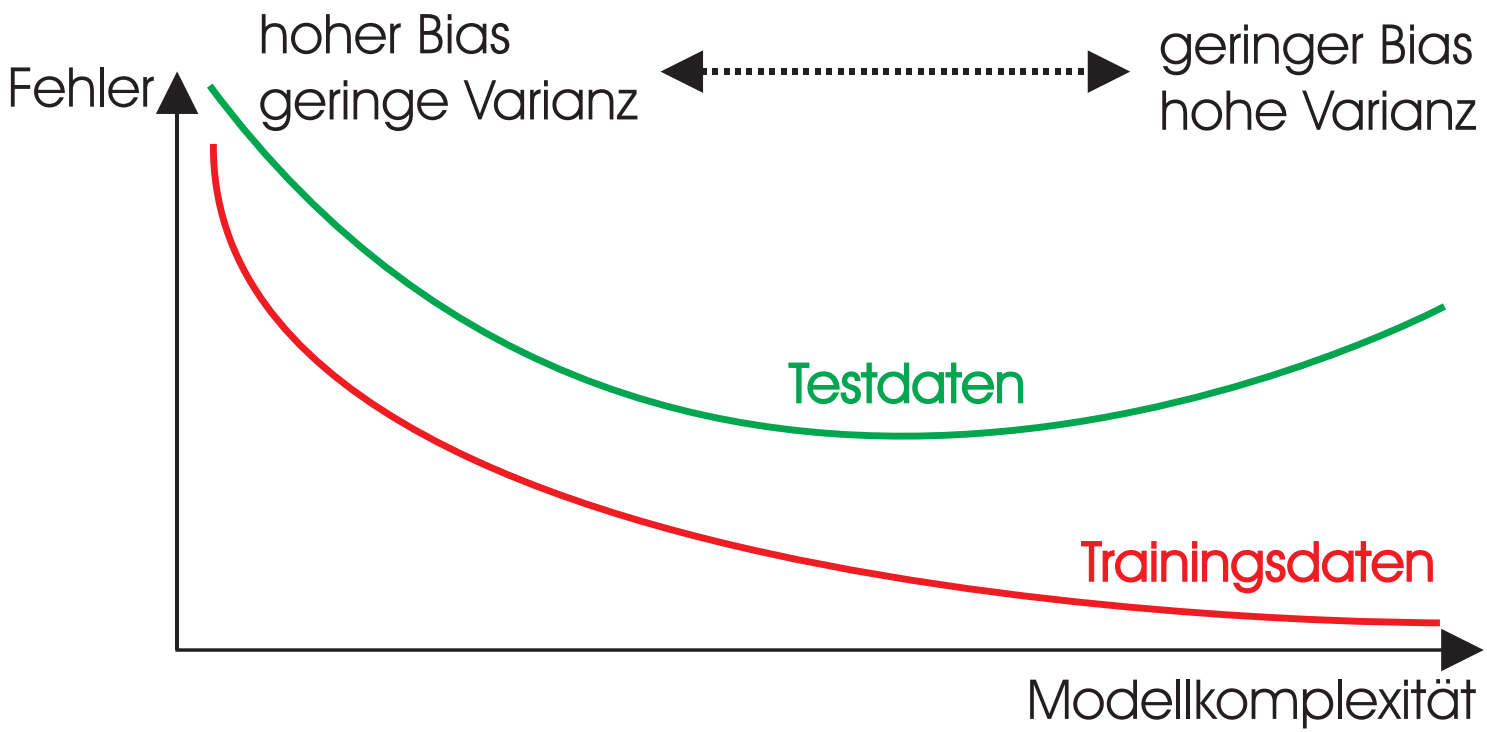
Komplexes Modell, komplexe Decision Boundary (perfekte Separierung der Trainingsdaten)



Kompromiss: hat möglicherweise optimale Generalisierungsfähigkeit

Intelligent Embedded Systems

Bewertung von Ergebnissen – 5



Quelle: [Hastie, Tibshitani, Friedman, 2001]

Intelligent Embedded Systems

Erwartungswert und Varianz – 1

Erwartungswert:

- Der Erwartungswert $\mathbb{E}[\mathbf{x}]$ eine Zufallsvariable $\mathbf{x}$ beschreibt einen Schätzer für den zu erwartenden Wert (durchschnittlichen Wert) eines Zufallsprozesses (hier: Datensatz).
- Um eine Verteilung zu beschreiben, benötigt man u. a. einen Lageparameter (Erwartungswert). Es kann beispielsweise entschieden werden, ob ein Spiel als fair anzusehen ist und im Mittel für alle Parteien die selbe Gewinnwahrscheinlichkeit vorliegt.
- Der Erwartungswert $\mathbb{E}[\mathbf{x}]$ wird definiert als $\mathbb{E}[\mathbf{x}] = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$ mit $N \in \mathbb{N}^+$ als Anzahl der Einzelereignisse.
- Beispiel: Der Erwartungswert für die Augenzahl eines idealen Würfels liegt bei 3.5.

Quelle: `http://link.springer.com/book/10.1007/978-3-8274-2760-1/page/1` Stand: 24.10.2014

Intelligent Embedded Systems

Erwartungswert und Varianz – 2

Varianz:

- Um eine Verteilung zu beschreiben, benötigt man neben einem Lageparameter (Erwartungswert) auch einen Streuungsparameter (z. B. die Varianz).
- Dieser Parameter beschreibt die zu erwartende Abweichung eines Ereignisses $\mathbf{x}$ vom Erwartungswert $\mathbb{E}[\mathbf{x}]$.
- Sei $\mathbf{x}$ eine diskrete Zufallsvariable mit dem Erwartungswert $\mathbb{E}[\mathbf{x}]$. Die Varianz von $\mathbf{x}$ ist definiert durch $\sigma^2 = \frac{1}{N-1} \sum_{n=1}^{N} (\mathbf{x}_n - \mathbb{E}[\mathbf{x}])^2$ mit $N \in \mathbb{N}^+$.
- Die Varianz ist das Quadrat der Standardabweichung σ.

Quelle: `http://link.springer.com/book/10.1007/978-3-8274-2760-1/page/1` Stand: 24.10.2014

Intelligent Embedded Systems

Bias und Varianz – 1

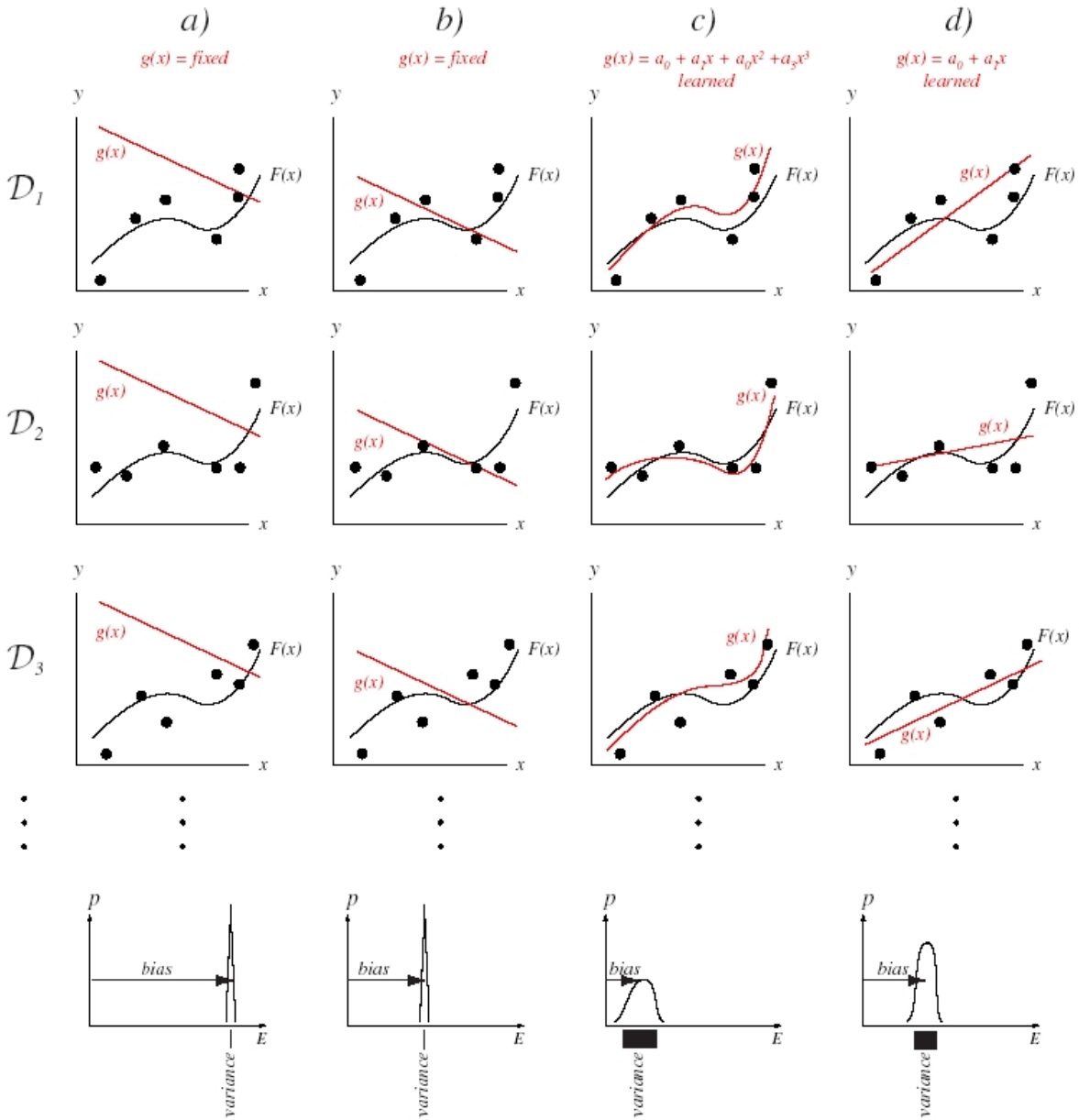
Begriffe:

- **Bias**: beschreibt die Genauigkeit von Modellen (hoher Bias – geringe Genauigkeit)
- **Varianz**: beschreibt die Spezifität von Modellen (hohe Varianz – geringe Spezifität)

Beispiele: Bias und Varianz bei Regressionsaufgaben

Anmerkung: Der Begriff des Bias hier hat nichts zu tun mit dem Begriff des Bias bei der Darstellung einer separierenden Hyperebene (s.o.).

Bias und Varianz – 2



$F(x)$: zu modellierende Funktion

$\mathcal{D}_i$: Mengen von Samples dieser Funktion

$g(x)$: Modell

E: Fehler eines Modells

p: Häufigkeit

Spalten: verschiedene Modelltypen

Zeilen: verschiedene Samplemengen

Quelle: [Duda, Hart, Stork, 2001]

Intelligent Embedded Systems

Bias und Varianz – 3

Sei $g(x)$ die Approximation von $f(x)$ basierend auf dem Datensatz $\mathcal{D}$.

Für manche Wahl des Datensatzes wird die Approximation gut sein, für andere schlecht. Uns interessiert als Maß der quadratische Abstand zwischen Modellinstanz (basierend auf dem jeweiligen Datensatz) und approximierter Funktion.

Mit der Notation $\mathbb{E}[.]$ für den Erwartungswert gilt:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}[(g(\mathbf{x}) - f(\mathbf{x}))^2] &= \\ &= \underbrace{(\mathbb{E}_{\mathcal{D}}[g(\mathbf{x}) - f(\mathbf{x})])^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathcal{D}}[(g(\mathbf{x}) - \mathbb{E}_D[g(\mathbf{x})])^2]}_{\text{Varianz}}\end{aligned}$$

Anmerkung: $g(\mathbf{x})$ basiert auf dem Datensatz $\mathcal{D}$, während $f(\mathbf{x})$ davon unabhängig ist.

Intelligent Embedded Systems

Bias und Varianz – 5

- niedriger Bias: die Funktion $f(\mathbf{x})$ wird durch $g(\mathbf{x})$mit Hilfe von $\mathcal{D}$ im Durchschnitt genau approximiert
- niedrige Varianz: die Approximation von $f(\mathbf{x})$ ändert sich nicht stark mit einem anderen Datensatz
- trotz eines Bias von Null (die Approximation ist „unbiased“) kann der Erwartungswert des quadratischen Fehlers hoch sein (wegen einer hohen Varianz)

Intelligent Embedded Systems

Bias und Varianz – 6

Bias-Varianz-Dilemma:

Es existiert ein Trade-Off zwischen Bias und Varianz: Modelle mit mehr freien Parametern haben meist geringeren Bias, aber höhere Varianz (und umgekehrt).

Ziel: niedriger Bias und niedrige Varianz

Intelligent Embedded Systems

Bias und Varianz – 7

- Bei Klassifikationsaufgaben ist die Zerlegung in Bias und Varianz schwieriger.
- Üblicherweise verwendet man ein 0/1-Fehlermaß (Sample wird entweder richtig oder falsch klassifiziert).
- Es zeigt sich, dass der Einfluss der Varianz bei Klassifikationsaufgaben unter der Annahme dieses Fehlermaßes deutlich höher ist als der des Bias.

Intelligent Embedded Systems

Test von Modellen – 1

(meint i. A. Test der Generalisierungsfähigkeit)

Methoden:

- Holdout-Methode
- Kreuzvalidierung
- Jackknife (Leave-One-Out)
- Bootstrap
- u. v. m.

Abschätzungen für Bias und Varianz: siehe z. B. [Duda, Hart, Stork, 2001]

Intelligent Embedded Systems

Test von Modellen – 2

Holdout-Methode:

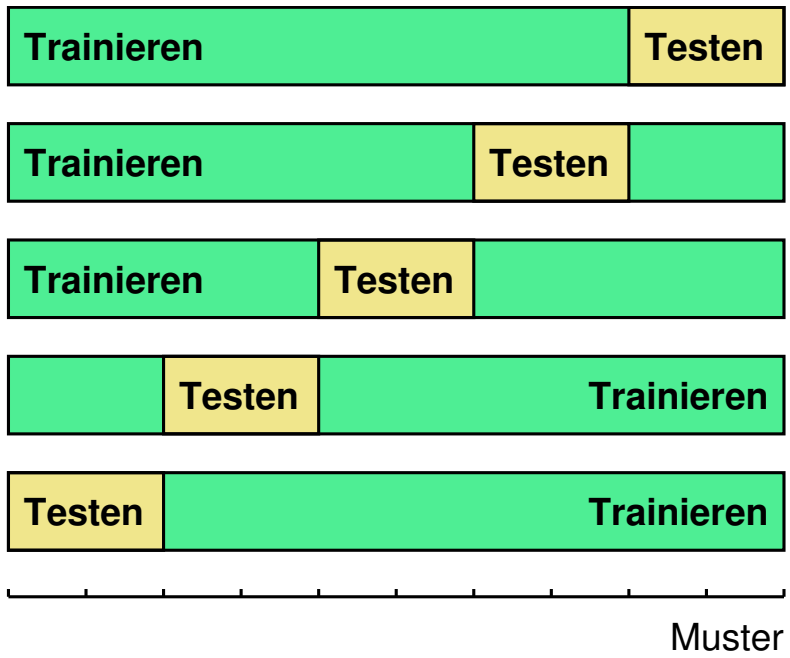
- Teil der verfügbaren Daten zur Modellbildung (Training), anderer Teil zum Test (unbekannte Daten)
- Oft: ein Drittel zum Testen, zwei Drittel zum Trainieren
- Problem: schlechte Schätzung der Generalisierungsleistung (starke Abhängigkeit von Datenaufteilung)

Intelligent Embedded Systems

Test von Modellen – 3

Kreuzvalidierung:

Wiederholung der Holdout-Methode mit unterschiedlichen Teildatenmengen:



(hier: 5-fache Kreuzvalidierung)

Intelligent Embedded Systems

Test von Modellen – 4

Jackknife (Leave-One-Out):

- Entspricht N-facher Kreuzvalidierung bei N Mustern in der Datenmenge
- In jedem Durchgang der Kreuzvalidierung wird also ein Muster zum Testen verwendet

Test von Modellen – 5

Bootstrap:

- Aus einer Datenmenge mit N Mustern werden N Muster durch Ziehen mit Zurücklegen ausgewählt.
- Die restlichen Muster werden zum Testen verwendet.
- Die Wahrscheinlichkeit, dass ein Muster nicht zum Trainieren ausgewählt wird, ist etwa 0.368.

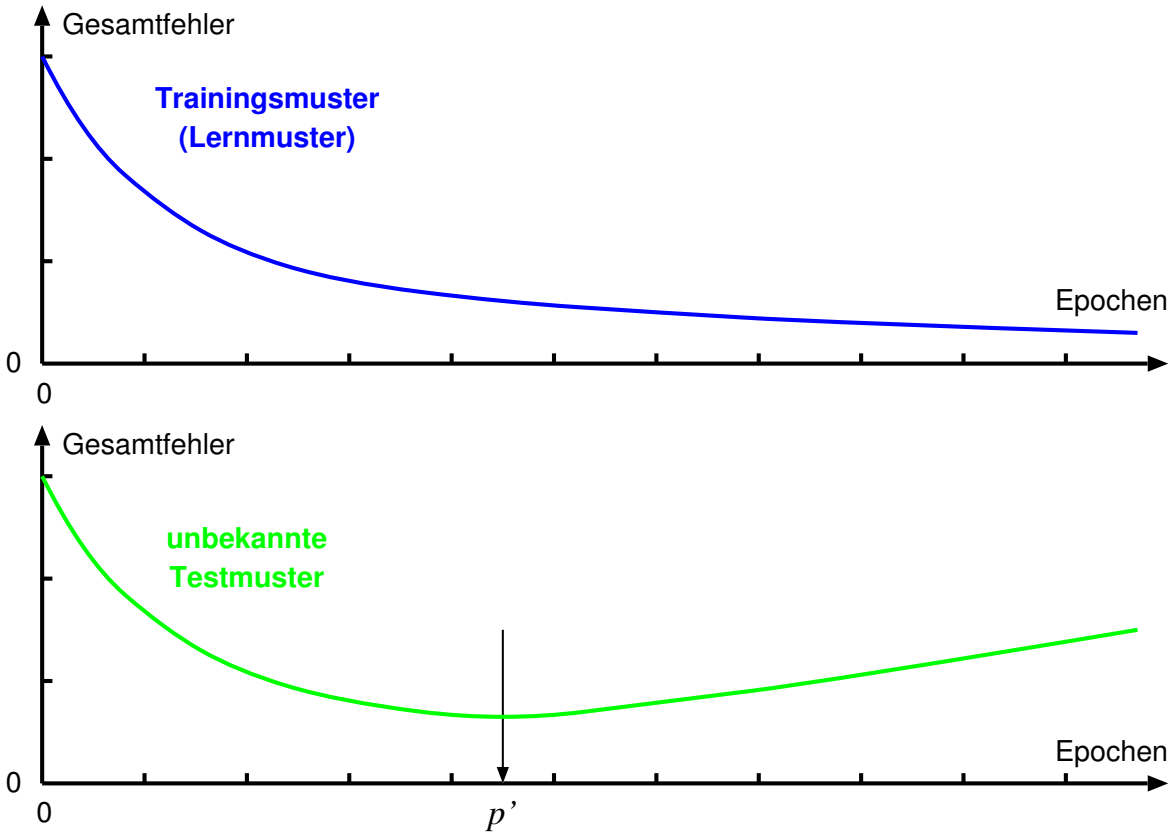
Überanpassung – 1

Eine Überanpassung (Overfitting) von Modellen an Trainingsdaten (schlechte Generalisierungsfähigkeit) ist möglich

- bei hoher Modellkomplexität (viele Freiheitsgrade)
- bei wenigen Trainingsmustern
- im Verlauf eines Parameteradaptionsvorgangs bei vielen iterativen Suchalgorithmen
- ...

Intelligent Embedded Systems

Überanpassung – 2



(z. B. beim Training eines Neuronalen Netzes, aber genauso bei anderen Paradigmen)

Intelligent Embedded Systems

Fazit

Bei der Bewertung der Generalisierungsleistung müssen zufällige und pseudo-zufällige Einflüsse berücksichtigt werden:

- Fehler bei der Ermittlung oder Messung von Mustern
- Beschreibung der zu modellierenden Funktion durch eine beschränkte, möglicherweise sehr kleine Menge von Mustern
- Einflüsse bei Suchalgorithmen, z. B. stochastische Optimierungsverfahren, wie
  - zufällige Startwerte bei iterativen Verfahren
  - Reihenfolge, in der Muster bei der Parameteradaption betrachtet werden
  - ...

Intelligent Embedded Systems

Sonstiges

Literatur zum Kapitel

- R. O. Duda, P. E. Hart, D. G. Stork: Pattern Classification

Ende

Intelligent Embedded Systems