Vorlesungsplan

- 17.10. Einleitung
- **24.10. Ein- und Ausgabe**
- 31.10. Reformationstag
- 7.11. Einfache Regeln, Naïve Bayes
- 14.11. Entscheidungsbäume, und Regeln
- 21.11. Assoziationsregeln
- 28.11. Lineare Modelle
- 5.12. Instanzbasiertes Lernen, Clustering
- 12.12. Evaluation I
- 19.12. Evaluation II
- 9.1. Entscheidungsbäume, Klassifikationsregeln
- 16.1. Lineare Modelle, Numerische Vorhersage
- 23.1. Clustering
- 30.1. Attribut-Selektion, Diskretisierung, Transformationen
- 6.2. Kombination von Modellen, Lernen von nicht-klassifizierten Beispielen

Eingabe: Konzepte, Instanzen, Attribute

- Terminologie
- Was ist ein Konzept?
  - Klassifikation, Assoziation, Clustering, Nummerische Vorhersage
- Was ist ein Beispiel?
  - Relationen, flache Dateien, Rekursion
- Was ist ein Attribut?
  - Nominal, Ordinal, Interval, Verhältnisse
- Vorbereitung der Eingabe
  - ARFF, Attributes, fehlende Werte, Daten kennenlernen

Terminologie

- Bestandteile der Eingabe:
  - Konzepte: Dinge die gelernt werden können
    - Ziel: verstehbare und operationale Konzeptbeschreibung
  - Instanzen: individuelle, unabhängige Beispiele eines Konzeptes
    - komplizierte Formen der Eingabe sind möglich
  - Attribut: beschreibt Aspekte einer Instanz
    - wir konzentrieren auf nominale und nummerische Attribute

Was ist ein Konzept?

- Verschiedene Arten des Lernens:
  - Klassifikationslernen:
    Vorhersage einer diskrete Klasse
  - Assoziationslernen:
    Assoziationen zw. Eigenschaften finden
  - Clustering:
    gruppiere ähnliche Instanzen in Cluster
  - Nummerische Vorhersage:
    Vorhersage einer nummerische Größe
- Konzept: Dinge, die gelernt werden können
- Konzeptbeschreibung:
  Ausgabe eines Lernalgorithmus

Klassifikationslernen

- Beispielprobleme: Wetterdaten, Kontaktlinsen, Irisdaten, Gewerkschaftsverhandlungen
- Klassifikationslernen ist überwacht
  - Algorithmus verfügt über Beispiele mit bekanntem Ergebnis
- Ergebnisse sind die *Klasse* der Beispiele
- Erfolg mißt sich auf neuen Daten für welche die Klassen bekannt sind (*Testdaten*)
- In der Praxis wird Erfolg oft subjektiv gemessen

Assoziationslernen

- Kann angewendet werden falls keine Klasse bekannt und jede Art von Struktur “interessant” ist
- Unterschied zum Klassifikationslernen:
  - Kann die Werte jedes Attributes vorhersagen, nicht nur die des Klassenattributes, auch mehre Attributewerte auf einmal sind möglich
  - weit mehr Assoziationsregeln als Klassifikationsregeln
  - Deshalb: Bedingungen sind notwendig
    - Minimaler Support und minimaler Konfidenz

Clustering

- Teile Objekte in Gruppen mittels Ähnlichkeit
  - Objekte derselben Gruppe sind ähnlich
  - Objekte verschiedener Gruppen sind unähnlich
- Clustering ist *unüberwacht*
  - Klasse der Beispiele sind nicht bekannt
- Erfolg wird oft subjektiv gemessen

	Sepal length	Sepal width	Petal length	Petal width	Type
1	5.1	3.5	1.4	0.2	Iris setosa
2	4.9	3.0	1.4	0.2	Iris setosa
…					
51	7.0	3.2	4.7	1.4	Iris versicolor
52	6.4	3.2	4.5	1.5	Iris versicolor
…					
101	6.3	3.3	6.0	2.5	Iris virginica
102	5.8	2.7	5.1	1.9	Iris virginica
…					

Nummerische Vorhersage

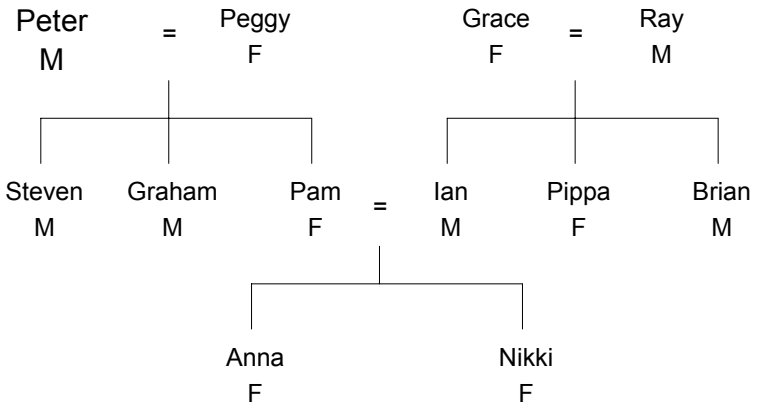
- Klassifikationslernen, aber “Klasse” ist nummerisch
- Lernen ist überwacht
  - Beispiele mit Zielwerten
- Erfolg wird auf Testdaten gemessen

Outlook	Temperature	Humidity	Windy	Play-time
Sunny	Hot	High	False	5
Sunny	Hot	High	True	0
Overcast	Hot	High	False	55
Rainy	Mild	Normal	False	40
…	…	…	…	…

Was ist ein Beispiel?

- Instanz: spezifischer Typ eines Beispieles
  - Obj. das klassifiziert, assoziiert oder gruppiert werden kann
  - Individuelles, unabhängiges Beispiel d. Zielkonzeptes
  - Charakterisiert durch vorgegebene Attributmenge
- Eingabe für den Lernalgorithmus: Menge von Instanzen/Datenmenge
  - Repräsentiert als einzelne Relation/Datei
- Relative restriktive Form der Eingabe
  - Keine Beziehungen zwischen Objekten
- Gebräuchlichste Form in der Praxis

Stammbaum



Stammbaum repräsentiert als Tabelle

Name	Gender	Parent1	parent2
Peter	Male	?	?
Peggy	Female	?	?
Steven	Male	Peter	Peggy
Graham	Male	Peter	Peggy
Pam	Female	Peter	Peggy
Ian	Male	Grace	Ray
Pippa	Female	Grace	Ray
Brian	Male	Grace	Ray
Anna	Female	Pam	Ian
Nikki	Female	Pam	Ian

Die “Schwester-von” Relation

First person	Second person	Sister of?
Peter	Peggy	No
Peter	Steven	No
…	…	…
Steven	Peter	No
Steven	Graham	No
Steven	Pam	Yes
…	…	…
Ian	Pippa	Yes
…	…	…
Anna	Nikki	Yes
…	…	…
Nikki	Anna	yes

First person	Second person	Sister of?
Steven	Pam	Yes
Graham	Pam	Yes
Ian	Pippa	Yes
Brian	Pippa	Yes
Anna	Nikki	Yes
Nikki	Anna	Yes
All the rest		No

Closed-world Annahme

Die volle Repräsentation in einer Tabelle

First person				Second person				Sister of?
Name	Gender	Parent1	Parent2	Name	Gender	Parent1	Parent2	
Steven	Male	Peter	Peggy	Pam	Female	Peter	Peggy	Yes
Graham	Male	Peter	Peggy	Pam	Female	Peter	Peggy	Yes
Ian	Male	Grace	Ray	Pippa	Female	Grace	Ray	Yes
Brian	Male	Grace	Ray	Pippa	Female	Grace	Ray	Yes
Anna	Female	Pam	Ian	Nikki	Female	Pam	Ian	Yes
Nikki	Female	Pam	Ian	Anna	Female	Pam	Ian	Yes
All the rest								No

```
If second person's gender = female
   and first person's parent = second person's parent
   then sister-of = yes
```


Erzeugung einer flachen Datei

- Denormalisierung
  - Mehrere Relationen werden zu einer verbunden
- Möglich mit jeder endlichen Anzahl von Relationen
- Problematisch: Beziehungen ohne feste Anzahl von Objekten
  - Beispiel: Konzept der *Familie (mit Vorfahren)*
- Denormalisierung kann scheinbare Regelmäßigkeiten erzeugen, welche die Struktur der Datenbank reflektieren
  - Beispiel: "Zulieferer" bestimmt "Zuliefereradresse"

Die "Vorfahre-von" Relation

First person				Second person				Ancestor of?
Name	Gender	Parent1	Parent2	Name	Gender	Parent1	Parent2	
Peter	Male	?	?	Steven	Male	Peter	Peggy	Yes
Peter	Male	?	?	Pam	Female	Peter	Peggy	Yes
Peter	Male	?	?	Anna	Female	Pam	Ian	Yes
Peter	Male	?	?	Nikki	Female	Pam	Ian	Yes
Pam	Female	Peter	Peggy	Nikki	Female	Pam	Ian	Yes
Grace	Female	?	?	Ian	Male	Grace	Ray	Yes
Grace	Female	?	?	Nikki	Female	Pam	Ian	Yes
Other positive examples here								Yes
All the rest								No

Rekursion

- Unendliche Beziehungen erfordern Rekursion

```
If person1 is a parent of person2
   then person1 is an ancestor of person2

If person1 is a parent of person2
   and person2 is an ancestor of person3
   then person1 is an ancestor of person3
```

- Passende Techniken sind "Inductive Logic Programming"
  - (z.B. Quinlan's FOIL)
  - Probleme: (a) Rauschen und (b) Berechnungskomplexität

Was ist ein Attribut?

- Jede Instanz wird durch feste vordefinierte Eigenschaftsmenge beschrieben
- Aber: Anzahl der Attribute kann in der Praxis variieren
  - Mögliche Lösung: “Irrelevanter Wert” Flag
- Problem: Vorhandensein eines Attributes kann vom Wert eines Anderen abhängen
- Mögliche Attributetypen:
  - *Nominal, Ordinal, Interval* und *Verhältnisse*

Nominale Attribute

- Werte sind unterschiede Symbole
  - Werte sind Bezeichnungen oder Namen
  - *Nominal* ist Latein für Name
- Beispiel: Attribut “outlook” der Wetterdaten
  - Werte: “sunny”,”overcast” und “rainy”
- Nominale Attributewerte implizieren keine Beziehungen untereinander (keine Reihenfolge oder Distanzmaß)
- Nur Gleichheitstest möglich

Ordinale Attribute

- Werte können geordnet werden
- Aber: Distanz zwischen den Werten ist nicht definiert
- Beispiel:
  Attribut “temperature” der Wetterdaten
  - Werte: “hot” > “mild” > “cool”
- Addition und Subtraktion sind nicht definiert
- Bespielregel:
  temperature < hot ⇨ play = yes
- Unterscheidung zwischen nominal und ordinal ist nicht immer klar (z.B. Attribut “outlook”)

Intervalattribute

- Intervalattribute sind nicht nur geordnet sondern werden auch in festen gleichmäßigen Einheiten gemessen
- Beispiel 1: Attribut “temperature” gemessen in Grad Celsius
- Beispiel 2: Attribut “year”
- Differenz von zwei Werten macht Sinn
- Summe oder Produkt macht keinen Sinn
  - Nullpunkt ist nicht definiert!

Verhältnisattribute

- Verhältnisattribute haben Maßschemata mit Nullpunkt
- Beispiel: Attribut “Distanz”
  - Abstand zwischen einem Objekt und sich selbst ist Null
- Verhältnisattribute werden in Gleitkommazahlen gemessen
  - Alle mathematischen Operationen sind erlaubt
- Aber: gibt es einen “inhärenten” definierten Nullpunkt?
  - Antwort hängt ab vom Stand der Wissenschaft (z.B. Celsius kannte keinen absoluten Nullpunkt)

Transformation ordinaler in boolsche Attribute

- Einfache Transformation erlaubt ordinale Attribute mit *n* Werten durch *n–1* boolsche Attribute zu kodieren
- Beispiel: Attribut “temperature”

Original Daten *Transformierte Daten*

Temperature
Cold
Medium
Hot

⇨

Temperature > cold	Temperature > medium
False	False
True	False
True	True

- Besser als es als nominales Attribut zu kodieren

Metadaten

- Informationen über die Daten, die Hintergrundwissen kodieren
- Kann genutzt werden um den Suchraum einzuschränken
- Beispiele:
  - Maßeinheiten (Ausdrücke müssen bezüglich der Einheiten korrekt sein)
  - Ringordnungen (z.B. Grad bei Winkeln (z.B. Torsionswinkel bei Aminosäuren)
  - Teilweise Ordnungen (z.B. Verallgemeinerung/Spezialisierung)

Vorbereitung der Eingabe

- Problem: verschiedene Datenquellen (z.B. Verkaufs-, Kundenabteilung, …)
  - Unterschiede: Arten der Aufzeichnung, Konventionen, Zeitperioden, Datenaggregation, Primärschlüssel, Fehler
  - Daten müssen zusammengefügt, Integriert und gereinigt werden
  - “Data warehouse”: konsistente Datenquelle
- Denormalisierung ist nicht das einzige Problem
- Externe Daten können gebraucht werden (“overlay data”)
- Typ und Grad der Daten Aggregation, Integration

Das ARFF Format

```
%
% ARFF file for weather data with some numeric features
%
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature numeric
@attribute humidity numeric
@attribute windy {true, false}
@attribute play? {yes, no}

@data
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
...
```


Attributtypen

- ARFF unterstützt nummerische und nominale Attribute
- Interpretation hängt vom Lernschema ab
  - Nummerische Attribute werden interpretiert als
    - ordinale Attribute wenn nur >, < genutzt werden
    - Verhältnisseattribute wenn Distanzberechnungen durchgeführt werden (Normalisierung/Standardisierung kann erforderlich sein)
  - Instanz-basierte Schemata definieren Distanz zwischen nominalen Werten (0 falls Werte gleich sind, 1 sonst)
- Integer: nominal, ordinal, oder Verhältnisattribut?

Nominal versus Ordinal

- Attribut “age” nominal

```
If age = young and astigmatic = no
    and tear production rate = normal
    then recommendation = soft
If age = pre-presbyopic and astigmatic = no
    and tear production rate = normal
    then recommendation = soft
```

- Attribut “age” ordinal (z.B. “young” < “pre-presbyopic” < “presbyopic”)

```
If age ≤ pre-presbyopic and astigmatic = no
   and tear production rate = normal
   then recommendation = soft
```


Fehlende Werte

- Oft makrkiert durch out-of-range Einträge
  - Typen: Unbekannt, nicht aufgezeichnet, nicht relevant
  - Gründe:
    - Fehlfunktion von Geräten
    - Änderung im Aufzeichnungdesign
    - Zusammenführung von verschiedenen Datenmengen
    - Aufzeichnung nicht möglich
- Fehlende Werte können selbst eine signifikante Bedeutung haben (z.B. fehlender Test bei einer medizinischen Untersuchung)
  - Die meisten Schemata nehmen an, dass dies nicht der Fall ist
    ⇨ “Fehlend” kann als zusätzlicher Wert kodiert werden

Fehlerhafte Werte

- Grund: Daten wurden nicht für die Analyse gesammelt
- Ergebnis: Fehler und Auslassungen, welche den ursprünglichen Zweck der Daten nicht beeinflussen (z.B. Alter eines Kunden)
- Schreibfehler in nominalen Attributen ⇒ Werten müssen auf Konsistenz geprüft werden
- Schreib- und Meßfehler in nummerischen Attributen ⇒ Ausreißer müssen gefunden werden
- Fehler können mutwillig falsch sein (z.B. falsche Postleitzahlen)
- Andere Probleme: Duplikate, veraltete Daten

Die Daten kennenlernen

- Einfache Visualisierungswerkzeuge sind sehr nützlich
  - Nominale Attribute: Histogramme (Ist die Verteilung konsistent mit dem Hintergrundwissen?)
  - Nummerische Attribute: Histogramme, Mittelwert, Varianz, Quantile, (offensichtliche Ausreißer?)
- 2-D und 3-D Visualisierungen zeigen Abhängigkeiten
- Gespräche mit Fachleuten
- Falls zu viele Daten zum explorieren? verwende eine Probe (zufällige Teilmenge)

Ausgabe: Wissensrepräsentation

- Entscheidungstabellen (Decision tables)
- Entscheidungsbäume (Decision trees)
- Entscheidungsregeln (Decision rules)
- Assoziationsregeln
- Regeln mit Beziehungen
- Instanz-basierte Repräsentation
- Cluster

Ausgabe: repräsentiert Strukturmuster

- Verschiedene Möglichkeiten Muster zu repräsentieren
  - Entscheidungsbäume, -regeln, Instanz-basiert
- Repräsentation bestimmt Algorithmus
- Verständnis der Ausgabe ist der Schlüssel zum Verständnis der zugrunde liegende Lernmethoden
- Verschiedene Ausgabetypen für verschiedene Lernprobleme (z.B. Klassifikation, Regression, …)

Entscheidungstabellen

- Einfachste Form um die Ausgabe zu repräsentieren:
  - Verwendet dasselbe Format wie die Eingabe
- Entscheidungstabelle für Wetterproblem:

Outlook	Humidity	Play
Sunny	High	No
Sunny	Normal	Yes
Overcast	High	Yes
Overcast	Normal	Yes
Rainy	High	No
Rainy	Normal	No

- Hauptproblem: Auswahl der richtigen Attribute

Entscheidungsbäume

- "Teile-und-Herrsche" Ansatz erzeugt Baum
- Knoten beinhaltet den Test einzelner Attribute
- Meistens: Attributwert wird mit Konstanten verglichen
- Andere Möglichkeiten:
  - Vergleich der Wert von zwei Attributen
  - Vergleich einer Funktion von einer oder mehreren Attributen
- Blätter legen die Klassifikation fest, o. Menge der Klassifikationen, oder Whr. Verteilung
- Neue Instanz wird durch einen Pfad klassifiziert

Nominale und nummerische Attribute

- Nominal:
  Anzahl der Kinder ist meist die der möglichen Attributwerte $\Rightarrow$ Attribut wird nur einmal getestet
  - Andere Möglichkeit: Aufspaltung in zwei Teilmengen
- Nummerisch:
  Test ob Wert > oder < einer Konstante ist
  $\Rightarrow$ Attribut kann mehrmals getestet werden
  - Andere Möglichkeit: Dreifach-Teilung (oder Mehrfach-Teilung)
    - Integer: <, =, >
    - Real: *unter, innerhalb, über*

Fehlende Werte

- Hat das Fehlen des Wertes Signifikanz?
- Ja $\Rightarrow$ "missing" ist ein seperater Wert
- Nein $\Rightarrow$ "missing" muss speziell behandelt werden
  - Lösung A: wähle für die Instanz den populärsten Pfad
  - Lösung B: teile Instanz in Stücke auf
    - Stücke bekommen Gewicht im Verhältnis der Trainingsinstanzen, die durch den jeweiligen Pfad bestimmt werden
    - Klassifikationen der Blätterknoten werden mittels der Gewichte kombiniert

Klassifikationsregeln

- Populäre Alternative zu Entscheidungsbäumen
- *Voraussetzung*: Testmenge (ähnlich der Bedingungen in den Knoten eines Entscheidungsbaums)
- Bedingungen sind üblicherweise UND verknüpft (allg. logische Ausdrücke sind aber auch möglich)
- *Schlussfolgerung*: Klassen, Klassenmengen oder Whr. Verteilung, die durch die Regel zugewiesen werden
- Individuelle Regln sind oft ODER verknüpft
  - Konflikte entstehen, falls verschiedene Schlussfolgerungen möglich sind

Von Bäumen zu Regeln

- Einfach: konvertiere Baum in Regelmenge
  - Eine Regel für jedes Blatt:
    - Voraussetzung enthält eine Bedingung für jeden Knoten auf dem Weg vom Blatt zur Wurzel
    - Schlussfolgerung ist die Klasse die durch das Blatt zugewiesen wird
- Erzeugte Regeln sind eindeutig
  - Reihenfolge der Ausführung spielt keine Rolle
- Aber: erzeugte Regeln sind kompliziert
  - Vereinfachungen um redundante Tests/Regeln zu vermeiden

Von Regeln zu Bäumen

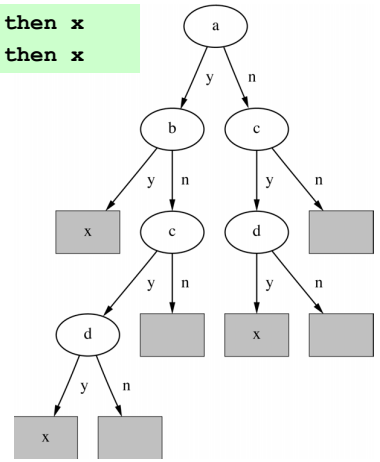
- Schwieriger: Transformation einer Regelmenge in einen Baum
  - Baum kann nicht leicht Disjunktion zwischen Regeln ausdrücken
- Beispiel: Regeln mit verschied. Testattributen

```
If a and b then x
If c and d then x
```

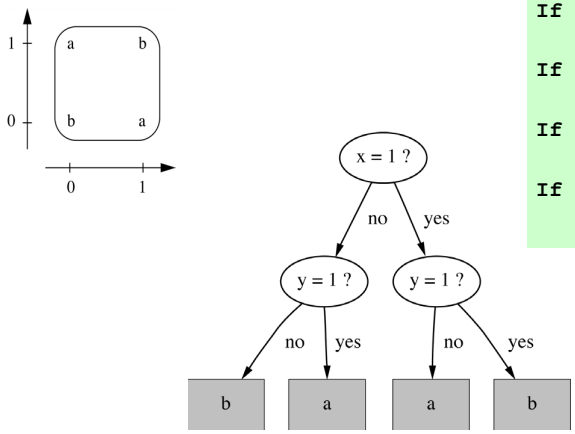
- Korrespondierender Baum enthält identische Teilbäume ($\Rightarrow$ “Redundanzproblem”)

Baum für eine einfache Disjunktion

```
If a and b then x
If c and d then x
```



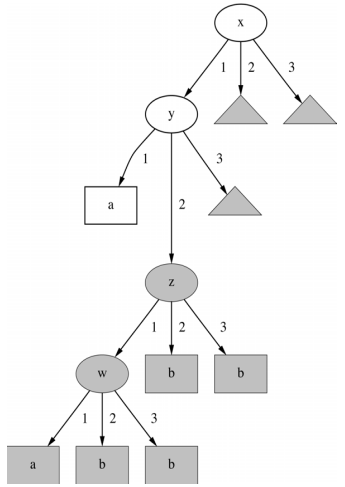
Exklusiv-oder Problem




```
If x = 1 and y = 0
   then class = a
If x = 0 and y = 1
   then class = a
If x = 0 and y = 0
   then class = b
If x = 1 and y = 1
   then class = b
```


Baum mit wiederholten Teilbäumen

```
If x = 1 and y = 1
   then class = a
If z = 1 and w = 1
   then class = a
Otherwise class = b
```



“Goldstücke” des Wissens

- Sind Regeln unabhängige Wissensstücke? (Es scheint leicht zu sein eine neue Regel zu einer Menge hinzuzufügen.)
- Problem: wie werden Regeln abgearbeitet?
- Zwei Möglichkeiten:
  - Geordnete Regelmenge (“Entscheidungsliste”)
    - Reihenfolge ist wichtig für die Interpretation
  - Ungeordnete Regelmenge
    - Regeln können sich widersprechen und zu verschiedenen Schlußfolgerungen bei einer Instanz führen

Interpretation der Regeln

- Was wenn zwei oder mehr Regeln anwendbar sind?
  - Keine Klassifikation?
  - Wahl der häufigsten Regel auf den Trainingsdaten?
  - …
- Was wenn keine Regel anwendbar ist?
  - Keine Klassifikation?
  - Wahl der häufigsten Klasse in den Trainingsdaten?
  - …

Spezialfall: boolsche Klasse

- Annahme: Falls Instanz nicht zu Klasse “yes” gehört, gilt Klasse “no”
- Trick: Lerne nur Regeln für Klasse “yes” und nutze Standardregel für “no” (Positivliste)

```
If x = 1 and y = 1 then class = a
If z = 1 and w = 1 then class = a
Otherwise class = b
```

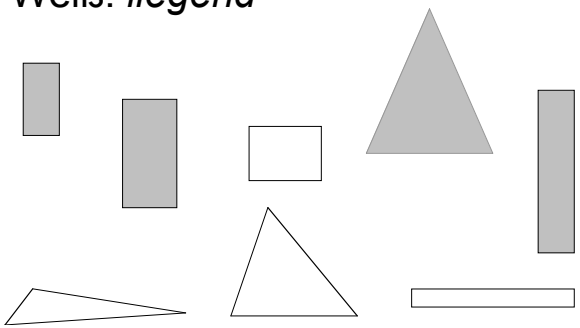
- Ordnung der Regeln nicht wichtig. keine Konflikte!
- Regeln können als eine in *disjunktiver Normalform* geschrieben werden

Regeln mit Beziehungen

- Bisher: alle Regeln vergleichen Attributwerte mit Konstanten (z.B. temperature < 45)
- Diese Regeln sind “aussagend”, weil sie die gleiche Ausdrucksmächtigkeit wie Aussagenlogik haben
- Was wenn das Lernproblem Beziehungen zwischen Beispielen umfaßt (z.B. Stammbaum-Problem)?
  - Kann nicht mit Aussageregeln erfaßt werden
  - Stärkere Regeln sind erforderlich

Formenproblem

- Zielkonzept: *aufrecht stehend*
- Grau: *stehend*
  Weiß: *liegend*



Aussagenbasierte Lösung

Width	Height	Sides	Class
2	4	4	Standing
3	6	4	Standing
4	3	4	Lying
7	8	3	Standing
7	6	3	Lying
2	9	4	Standing
9	1	4	Lying
10	2	3	Lying

```
If width ≥ 3.5 and height < 7.0
   then lying
If height ≥ 3.5 then standing
```

Lösung mit Beziehungen

- Vergleich von Attributen untereinander

```
If width > height then lying
If height > width then standing
```

- Trifft das Problem besser, d.h. neue Daten werden korrekter beschrieben
- Standard Relationen: =, <, >
- Aber: lernen von relationalen Regeln ist aufwändig
- Einfache Lösung: füge Extraattribute hinzu (z.B. ein binäres Attribut "*is width < height?*")

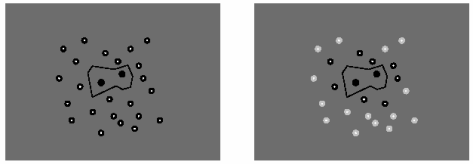
Instanz-basierte Repräsentation

- Einfachste Form des Lernens: auswendig Lernen
  - Trainingsinstanzen werden nach einer möglichst ähnlichen Instanz durchsucht
  - Instanzen repräsentieren das Wissen selbst
- Ähnlichkeitsfunktion definiert was "lernen" heißt
- Instanz-basiertes Lernen ist *faules* lernen
- Methoden: k-nächste Nachbarn, …

Distanzfunktion

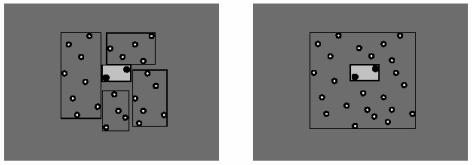
- Einfachster Fall: ein nummerisches Attribut
  - Distanz ist Differenz zwischen zwei Attributwerten (oder Funktion davon)
- Mehrere nummerische Attribute: oft Euklidische Distanz, Attribute sind normalisiert
- Nominales Attribut: Distanz ist 1 falls Werte verschieden, 0 falls sie gleich sind (Hamming)
- Sind alle Attribute gleichwichtig?
  - Gewichtung der Attribute möglich

Lernen von Prototypen



- Nur solche Instanzen, die Entscheidungen beeinflussen können, müssen gespeichert werden (Statische Trainingsmenge)
- Ausreißer sollten herausgefiltert werden
- Idee: nutze nur *prototypische* Beispiele

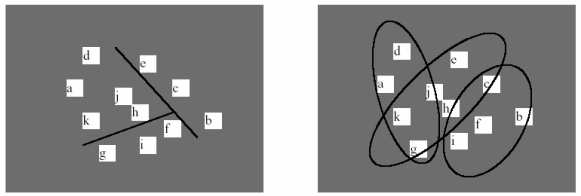
Verallgemeinerungen zu Rechtecken



- Nächste Nachbar Regel wird auf Rechtecke angewandt, falls neue Instanz außerhalb
- Rechtecke sind auch Regeln! (Aber konservativer als “normale” Regeln.)
- Verschachtelte Rechtecke sind Regeln mit Ausnahmen

Repräsentation von Clustern I

Geometrische Repräsentation *Mengen-Repräsentation*



Überlappende Cluster

Repräsentation von Clustern II

Probabilistisches Zuweisung *Dendrogram*

	1	2	3
a	0.4	0.1	0.5
b	0.1	0.8	0.1
c	0.3	0.3	0.4
d	0.1	0.1	0.8
e	0.4	0.2	0.4
f	0.1	0.4	0.5
g	0.7	0.2	0.1
h	0.5	0.4	0.1

Dendron ist Griechisch für Baum