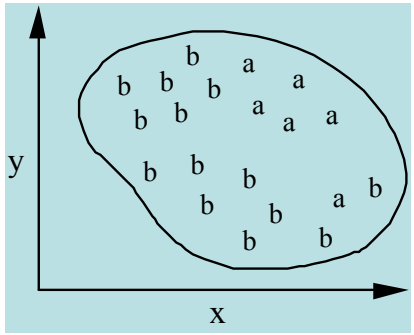
Vorlesungsplan

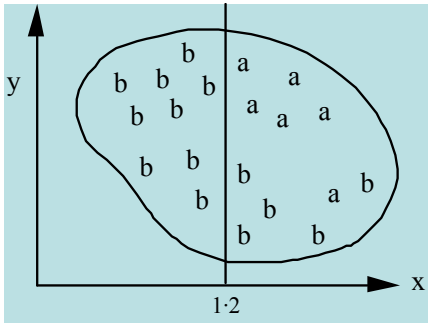
- 17.10. Einleitung
- 24.10. Ein- und Ausgabe
- 31.10. Reformationstag, Einfache Regeln
- 7.11. Naïve Bayes, Entscheidungsbäume
- **14.11. Entscheidungsregeln, Assoziationsregeln**
- 21.11. Lineare Modelle, Instanzbasiertes Lernen
- 28.11. Clustering I
- 5.12. Clustering II
- 12.12. Evaluation I
- 19.12. 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

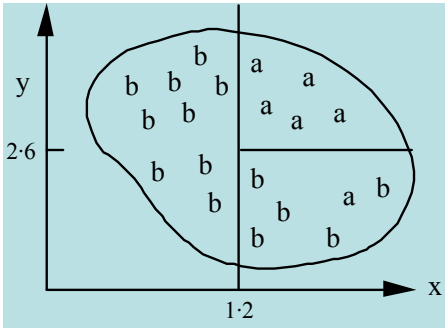
Überdeckende Algorithmen

- Konvertierung Entscheidungsbaum in Regelmenge
  - Direkt möglich, aber Regelmenge ist unnötig kompliziert
  - Effektivere Konvertierungen sind nicht trivial
- Anstelle, erzeuge Regelmenge direkt
  - für jede Klasse: finde Regelmenge, die alle Instanzen überdeckt (Instanzen die nicht in der Klasse sind, werden ausgeschlossen)
- Überdeckungsansatz:
  - in jedem Schritt finde Regel, die einige Instanzen überdeckt

Beispiel: Erzeugung einer Regel








```
If true
   then class = a
```

```
If x > 1.2
   then class = a
```

```
If x > 1.2 and y > 2.6
   then class = a
```

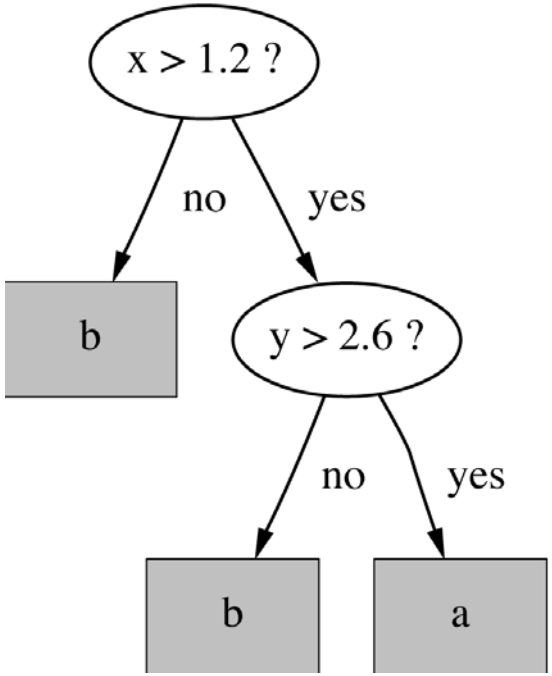
- Mögliche Regelmenge für Klasse “b”:

```
If x ≤ 1.2 then class = b
If x > 1.2 and y ≤ 2.6 then class = b
```

- Mehr Regeln “perfektionieren” Regelmenge

Regeln vs. Bäume

- Zugehöriger Entscheidungsbaum:
  (findet exakt dieselben Vorhersagen)



- Aber: Regelmenge kann verständlicher sein,
  wenn Entscheidungsbaum replizierte Teilbäume enthält
- Auch: Bei Mehrklassenproblem Überdeckungsalg.
  konzentriert sich nur auf eine Klass, Alg. zum Lernen eines
  Entscheidungsbaums betrachtet Verteilung aller Klassen

Einfacher Überdeckungsalgorithmus

- Erzeuge Regeln durch Hinzufügen von Tests, welche die Genauigkeit verbessern
- Ähnliche Situation wie Entscheidungsbäume: wähle ein Attribut zum Aufteilen
  - Aber: beim Lernen von Entscheidungsbäumen wird Gesamtreinheit maximiert
- Jeder neue Test reduziert die Überdeckung der Regel

Beispiel: Kontaktlinsen Daten

- Aktuelle Regel:

```
If ?
     then recommendation = hard
```

- Mögliche Tests:

```
Age = Young                                2/8
Age = Pre-presbyopic                       1/8
Age = Presbyopic                           1/8
Spectacle prescription = Myope             3/12
Spectacle prescription = Hypermetrope      1/12
Astigmatism = no                           0/12
Astigmatism = yes                          4/12
Tear production rate = Reduced             0/12
Tear production rate = Normal              4/12
```

Veränderte Regel & zugehörige Daten

- Regel mit bestem Test:

```
If astigmatism = yes
    then recommendation = hard
```

- Überdeckte Instanzen:

Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	Yes	Reduced	None
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	Yes	Reduced	None
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	Yes	Reduced	None
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	Yes	Reduced	None
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	Yes	Reduced	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	Yes	Reduced	None
Presbyopic	Hypermetrope	Yes	Normal	None

Weiterer Aufbau

- Aktuell:

```
If astigmatism = yes
    and ?
  then recommendation = hard
```

- Mögliche Tests:

```
Age = Young                              2/4
Age = Pre-presbyopic                     1/4
Age = Presbyopic                         1/4
Spectacle prescription = Myope           3/6
Spectacle prescription = Hypermetrope    1/6
Tear production rate = Reduced           0/6
Tear production rate = Normal            4/6
```

Veränderte Regel & zugehörige Daten

- Regel mit bestem Test :

```
If astigmatism = yes
    and tear production rate = normal
  then recommendation = hard
```

- Überdeckte Instanzen :

Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	Yes	Normal	None

Weiterer Aufbau

- Aktuell :

```
If astigmatism = yes
     and tear production rate = normal
     and ?
   then recommendation = hard
```

- Mögliche Tests :

`Age = Young`	`2/2`
`Age = Pre-presbyopic`	`1/2`
`Age = Presbyopic`	`1/2`
`Spectacle prescription = Myope`	`3/3`
`Spectacle prescription = Hypermetrope`	`1/3`

- Gleichstand zwischen erstem und viertem Test
  - Wähle Regel mit größerer Überdeckung

Ergebnis

- Regel:

```
If astigmatism = yes
   and tear production rate = normal
   and spectacle prescription = myope
   then recommendation = hard
```

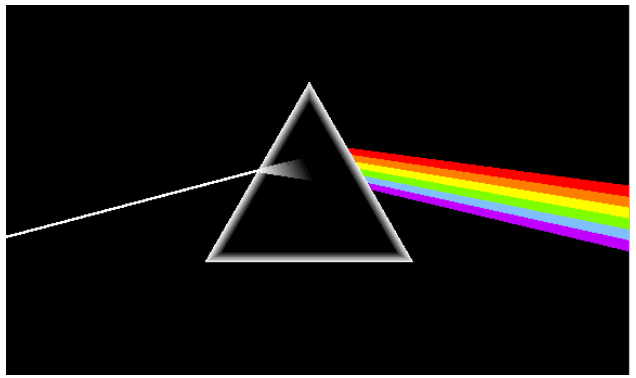
- Zweite Regel für “hard lenses”:
  (erzeugt auf Instanzen, die nicht von der ersten Regel überdeckt werden)

```
If age = young and astigmatism = yes
   and tear production rate = normal
   then recommendation = hard
```

- Diese zwei Regeln überdecken alle “hard lenses”:
  - Prozeß wird wiederholt mit den anderen beiden Klassen

Pseudo-Kode für PRISM

```
For each class C
  Initialize E to the instance set
  While E contains instances in class C
    Create a rule R with an empty left-hand side that predicts class C
    Until R is perfect (or there are no more attributes to use) do
      For each attribute A not mentioned in R, and each value v,
        Consider adding the condition A = v to the left-hand side of R
        Select A and v to maximize the accuracy p/t
          (break ties by choosing the condition with the largest p)
      Add A = v to R
    Remove the instances covered by R from E
```



Regeln vs. Entscheidungslisten

- PRISM ohne äußere Schleife erzeugt Entscheidungsliste für **eine Klasse**
  - Spätere Regeln sind für Instanzen, die nicht durch frühere Regeln überdeckt werden
  - Aber: Reihenfolge spielt keine Rolle, da alle Regeln dieselbe Klasse vorhersagen
- Äußere Schleife betrachtet die Klassen seperat
  - Keine Reihenfolge der Klassen wird angenommen
- Probleme: überlappende Regeln => Default-Regel erforderlich

Aussondern und Herrschen

- Methoden wie PRISM (zum Behandeln einer Klasse) sind *separate-and-conquer* Algorithmen:
  - (i) finde nützliche Regel
  - (ii) sondere die überdeckten Instanzen aus
  - (iii) “conquer” die übrigen Instanzen
- Unterschied zu divide-and-conquer Methoden:
  - Teilmengen, die von einer Regel überdeckt werden, brauchen nicht weiter bearbeitet zu werden

Assoziationsregeln

- Assoziationsregeln …
  - … können jedes Attribut und Attributkombinationen vorhersagen
  - … sind nicht dafür gedacht als Menge genutzt zu werden
- Problem: extrem große Anzahl von möglichen Assoziationsregeln
  - Ausgabe muß auf die besten Regeln beschränkt werden $\Rightarrow$ nur solche mit hoher Überdeckung (support) und hoher Konfidenz

Überdeckung und Konfidenz einer Regel

- Überdeckung (Support): Anzahl (absolut/relativ) der Instanzen, die korrekt vorhergesagt werden
- Konfidenz: Anzahl der korrekt Vorhersagen im Verhältnis zu allen Instanzen, auf die die Regel zutrifft
- Beispiel:

```
If temperature = cool then humidity = normal
```

$\Rightarrow$ Überdeckung = 4, Konfidenz = 100%

- Anwendung: minimale Überdeckung und Konfidenz wird vorgegeben (z.B. 58 Regeln mit Überdeckung ≥ 2 und Konfidenz $\geq 95\%$ für Wetterdaten)

Outlook	Temp	Humidity	Windy	Play
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

Interpretation von Assoziationsregeln

- Interpretation ist nicht offensichtlich:

```
If windy = false and play = no
   then outlook = sunny and humidity = high
```

ist *nicht* das gleiche wie

```
If windy = false and play = no
   then outlook = sunny
If windy = false and play = no
   then humidity = high
```

- Aber es bedeutet, daß folgendes auch zutrifft:

```
If humidity = high and windy = false and play = no
   then outlook = sunny
```

Finden von Assoziationsregeln

- Naïve Methode:
  - Nutze separate-and-conquer Methode
  - Behandele jede mögliche Kombination von Attributwerten als separate Klasse
  - Lassen Regeln weg, die nicht minimale Überdeckung und Konfidenz erfüllen
- Zwei Probleme:
  - Berechnungskomplexität
  - Anzahl der untersuchten Regeln
- Aber: Regeln mit hoher Überdeckung direkt finden!

Item-Mengen

- Überdeckung: Anzahl der korrekt überdeckten Instanzen
  - Entspricht der Anzahl der Instanzen die allen Tests in der Regel genügen (linke und rechte Seite)
- *Item*: ein Test/Attributwert Paar
- *Item-Menge* : alle Items, die in einer Regel auftreten
- Ziel: nur Regeln, die minimale Überdeckung erfüllen

  $\Rightarrow$ Finde Item-Mengen, die min. Überdeckung erfüllen und generiere Regeln von diesen!

Item-Mengen für Wetterdaten

One-item sets	Two-item sets	Three-item sets	Four-item sets
Outlook = Sunny (5)	Outlook = Sunny Temperature = Hot (2)	Outlook = Sunny Temperature = Hot Humidity = High (2)	Outlook = Sunny Temperature = Hot Humidity = High Play = No (2)
Temperature = Cool (4)	Outlook = Sunny Humidity = High (3)	Outlook = Sunny Humidity = High Windy = False (2)	Outlook = Rainy Temperature = Mild Windy = False Play = Yes (2)
…	…	…	…

- Gesamt: 12 one-Item-Mengen, 47 two-Item-Mengen, 39 three-Item-Mengen, 6 four-Item-Mengen und 0 five-Item-Mengen (mit minimaler Überdeckung 2)

Outlook	Temp	Humidity	Windy	Play
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

Erzeugen von Regeln aus Item-Mengen

- Item-Mengen mit minimaler Überdeckung können in Regeln gewandelt werden
- Beispiel:

```
Humidity = Normal, Windy = False, Play = Yes (4)
```

- Sieben (2^N-1) mögliche Regeln:

```
If Humidity = Normal and Windy = False then Play = Yes    4/4
If Humidity = Normal and Play = Yes then Windy = False    4/6
If Windy = False and Play = Yes then Humidity = Normal    4/6
If Humidity = Normal then Windy = False and Play = Yes    4/7
If Windy = False then Humidity = Normal and Play = Yes    4/8
If Play = Yes then Humidity = Normal and Windy = False    4/9
If True then Humidity = Normal and Windy = False
   and Play = Yes                                         4/12
```

Regeln für Wetterdaten

- Regeln mit Überdeckung > 1 und Konfidenz = 100%:

	Association rule		Sup.	Conf.
1	Humidity=Normal Windy=False	⇒ Play=Yes	4	100%
2	Temperature=Cool	⇒ Humidity=Normal	4	100%
3	Outlook=Overcast	⇒ Play=Yes	4	100%
4	Temperature=Cold Play=Yes	⇒ Humidity=Normal	3	100%
	...	...	...	...
58	Outlook=Sunny Temperature=Hot	⇒ Humidity=High	2	100%

- Gesamt:
  - 3 Regeln mit Überdeckung vier
  - 5 Regeln mit Überdeckung drei
  - 50 Regeln mit Überdeckung zwei

Beispiele für Regeln von einer Item-Menge

- Item-Menge :

```
Temperature = Cool, Humidity = Normal, Windy = False, Play = Yes (2)
```

- Ergebnis Regeln (alle mit 100% Konfidenz):

```
Temperature = Cool, Windy = False ⇒ Humidity = Normal, Play = Yes
Temperature = Cool, Windy = False, Humidity = Normal ⇒ Play = Yes
Temperature = Cool, Windy = False, Play = Yes ⇒ Humidity = Normal
```

wegen folgender “häufigen” Item-Mengen:

```
Temperature = Cool, Windy = False                      (2)
Temperature = Cool, Humidity = Normal, Windy = False   (2)
Temperature = Cool, Windy = False, Play = Yes          (2)
```

Effizientes Finden von Item-Mengen

- Problem: finde alle häufigen Item-Mengen
- Finden von Einer-Item-Mengen ist leicht
- Idee: nutze Einer-Item-Mengen um Zweier-Item-Mengen zu generieren, aus Zweier-Item-Mengen generiere Dreier-Item-Mengen, …
  - Falls (A B) häufig ist, dann müssen (A) und (B) häufig sein!
  - Allgemein: falls X eine häufige *k*-Item-Menge ist, dann müssen alle (*k*-1)-Item-Teilmengen von X häufig sein

  $\Rightarrow$ Berechne Kandidaten für *k*-Item-Mengen durch Zusammenfassen von (*k*-1)-Item-Mengen

Beispiel

- Gegeben: 5 Dreier-Item-Mengen

  (A B C), (A B D), (A C D), (A C E), (B C D)

- Lexikographisch geordnet!
- Kandidaten für Vierer-Item-Mengen:

  (A B C D) OK weil (B C D)

  (A C D E) Nicht OK weil (C D E) fehlt

- Kandidatentest durch Zählen der Instanzen in der Datenmenge!
- (k –1)-Item-Mengen werden in Hash gespeichert

Effiziente Erzeugung der Regeln

- Suche nach Regeln mit hoher Konfidenz
  - Überdeckung der Voraussetzung aus dem Hash
  - Aber: brute-force Methode kostet in $O(2^N-1)$
- Besserer Weg: erzeuge Regeln mit (c+1) Attributen in der Schlußfolgerung aus denen die nur c Attribute dort haben.
  - Beobachtung: (c + 1)-Schlußfolgerungsregel kann nur zutreffen, wenn alle korrespondierenden c-Schlußfolgerungsregel zutreffen
- Algorithmus ist ähnlich zu finden von häufigen Item-Mengen

Beispiel

- 1- Schlußfolgerungsregel:

```
If Outlook = Sunny and Windy = False and Play = No
   then Humidity = High (2/2)
```

```
If Humidity = High and Windy = False and Play = No
   then Outlook = Sunny (2/2)
```

- Korrespondierende 2- Schlußfolgerungsregel:

```
If Windy = False and Play = No
   then Outlook = Sunny and Humidity = High (2/2)
```

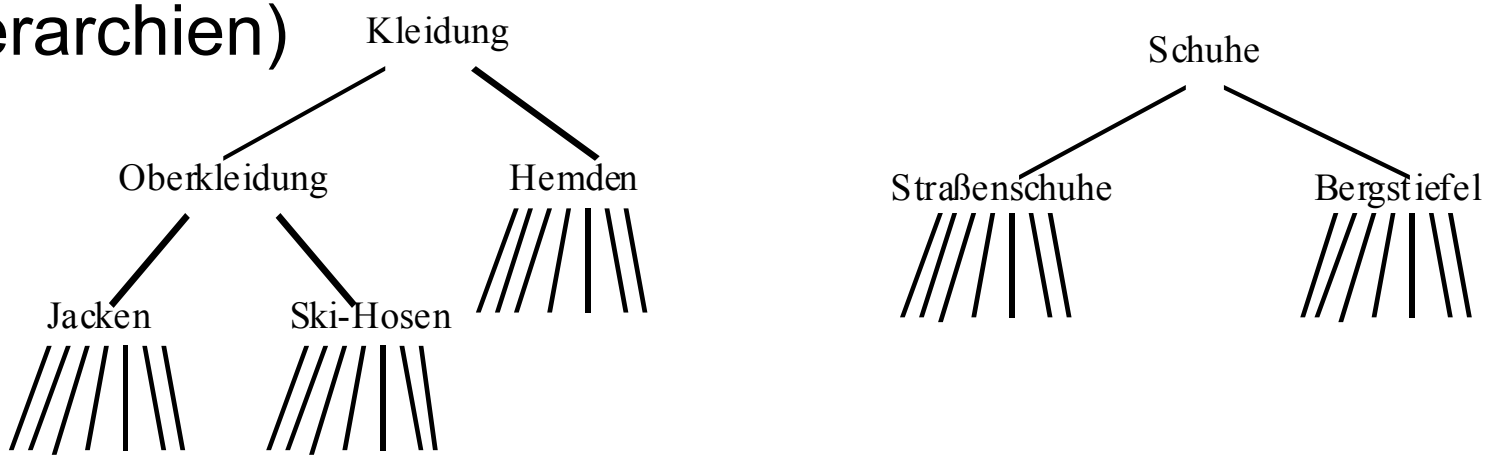
- Test der Voraussetzung gegen den Hash!

Erweiterungen (1)

- Problem: selbst hohe minimale Konfidenz garantiert nicht immer interessante Regeln
  - z.B. falls alle Instanzen Z enthalten,
    dann hat jede Regel I => Z Konfidenz 100%.
- Lift einer Regel *I* => *J* als anderes Maß zum Filtern
  - lift = Pr(J|I) / Pr(J)
  - Hinweis:
    - Pr(I) = (Support von I) / (Anzahl der Instanzen)
    - Pr(J|I) = (Support von I vereinigt mit J) / (Support von I)
  - Verhältnis von Konfidenz zu erwarteter Konfidenz
- Interpretation:
  - Falls lift > 1, dann I und J sind positiv korreliert
    lift < 1, dann I und J sind negativ korreliert
    lift = 1, dann I und J sind unkorreliert

Erweiterungen (2)

- in vielen Anwendungen: Item-Taxonomien (*is-a* Hierarchien)



- suche Assoziationsregeln zwischen abstrakten Items
  - z.B. zwischen Warengruppen
  - wesentlich höherer Support
  - viele triviale Regeln, durch die Taxonomie induziert, müssen ausgeschlossen werden

Assoziationsregeln: Diskussion

- Apriory Methode läuft für jede Größe der Kandidaten-Item-Mengen über die Daten
  - generiere (k+2)-Item-Mengen statt (k+1) => weniger Datendurchläufe
- Kandidaten Menge ist zu groß
  - nutze Depth-First Search statt BFS durch den Suchraum
- Item-Mengen sind redundant, A' ist Teil-Item-Mengen von A mit gleicher Überdeckung
  - geschlossene Item-Mengen
- Genaue Überdeckung von Item-Mengen ist nicht wichtig
  - maximale Item-Mengen, alle Teilmengen einer häufigen Item-Menge werden weggelassen

Assoziationsregeln: Diskussion

- Standard ARFF Format (feste Vektorlänge) sehr ineffizient für typische Supermarktdaten
  - Bit-Matrix mit vielen Nullen
  - Dünne Datenrepräsentation
- Einsen in Bitmatrix sind wichtiger als Nullen => keine Negation in den Regeln
- Instanzen werden auch *Transaktionen* genannt
- Konfidenz ist nicht notwendigerweise das beste Maß
  - Andere Maße mit ähnlichen Eigenschaften (Monotonie)