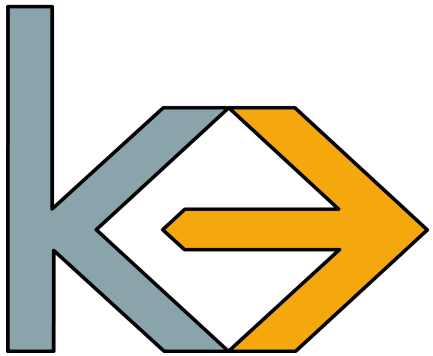
Probabilistisches Regellernen & Ranking mittels Klassifikations-regeln

Martin Schmitz
mschmitz@rbg.informatik.tu-darmstadt.de

TECHNISCHE UNIVERSITÄT DARMSTADT

Seminar aus maschinellem Lernen
Fachgebiet Knowledge Engineering
Fachbereich Informatik



Übersicht

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Probabilistisches Regellernen**
  - Motivation
  - Grundlagen
  - Problemdefinition
  - ProbFOIL
  - Experimente
  - Kritik
- **Ranking mittels Klassifikationsregeln**
  - Motivation
  - Methoden
  - Ranking mit mehreren Regeln
  - Experimente
  - Kritik
- Kombination
- Zusammenfassung

TECHNISCHE UNIVERSITÄT DARMSTADT

Probabilistisches Regellernen

Luc De Raedt und Ingo Thon

Motivation

Probabilistisches Lernen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Problem 1: Regellerner können häufig nur mit boolschen Fakten umgehen

`regen(t)` **entweder** `wahr` **oder** `falsch`

→ Unsicherheit der Daten wird nicht beachtet (z.B. die Regenwahrscheinlichkeit)

- Problem 2: rein statistische Verfahren lernen keine logischen Regeln

⟶ **Erweiterung eines Regellerners um Wahrscheinlichkeiten**

`0.2::regen(t)`

das Prädikat `regen(t)` ist mit einer Wahrscheinlichkeit von 20% wahr

Grundlagen (1)

ProbLog

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Prolog:** logische Programmiersprache
  - Ein Faktum ist entweder `wahr` oder `falsch`
  - Berechnet für ein Programm, ob eine Abfrage `wahr` oder `falsch` ist

- **ProbLog:** eine probabilistische Implementierung von Prolog
  - Ein Faktum ist mit Wahrscheinlichkeit `p` `wahr` ansonsten `falsch`
  - Berechnet die Erfolgswahrscheinlichkeit `P` einer Abfrage

Grundlagen (2)

ProbLog: Berechnung einer Erfolgswahrscheinlichkeit

TECHNISCHE UNIVERSITÄT DARMSTADT

Fakten: `0.2::regen(t)` `0.7::windOK(t)` `0.6::sonne(t)` (unabhängig verteilt)

Regeln:
```
surfen(X) :- not regen(X), windOK(X)
surfen(X) :- not regen(X), sonne(X)
```

Erfolgswahrscheinlichkeit:

$$P(\texttt{surfen(t)})$$
$$=$$
$$P(\ (\neg\texttt{regen(t)} \wedge \texttt{windOK(t)}) \vee (\neg\texttt{regen(t)} \wedge \texttt{sonne(t)})\ )$$
$$=$$
$$P((\neg\texttt{regen(t)} \wedge \texttt{windok(t)}) \vee (\neg\texttt{regen(t)} \wedge \texttt{sonne(t)} \wedge \neg\texttt{windOK(t)}))$$
$$=$$
$$(1 - 0.2) * 0.7 + (1 - 0.2) * 0.6 * (1 - 0.7)$$
$$=$$
$$0.704$$

→ **Ergebnis:** `surfen(X)` ist in 70,4% der Fälle `wahr`

Problemdefinition

Probabilistisches Regellernen (PRL)

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Gegeben:**
  1) Eine **Menge von Beispielen** $E = \{(x_i, p_i) | x_i$: Faktum für Zielprädikat `t`; $p_i \in [0, 1]$: die Wahrscheinlichkeit von $x_i\}$.

  2) Eine **Hintergrundtheorie** `B` in Form eines ProbLog-Programms.

  3) Eine **Loss-Funktion** `loss(H, B, E)`, welche den Loss einer Hypothese `H` (d.h. einer Menge von Klauseln) in Bezug auf `B` und `E` misst.

- **Gesucht:** $argmin_H loss(H, B, E) = argmin_H \sum_{e \in E} |P_s(B \cup H \models e) - p_i|$

  Erfolgswahrscheinlichkeit, dass für Hintergrund-theorie B und Hypothese H das Beispiel e `wahr` ist

→ Minimierung der absoluten Differenz zwischen Vorhersagen `P` und Beobachtungen `p`

ProbFOIL (1)

Algorithmus: äußere Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

Algorithm 1 The ProbFOIL algorithm

1: $h := t(X_1, \ldots, X_n)$ where t is the target predicate and the X_i distinct variables;
2: $H := \emptyset$; $c := (h \leftarrow b)$; $b := []$;
3: **repeat**
4:
5:
6:
7:
8:
9:
10:
11:
12:
13: **until** globalscore$(H) \geq$ globalscore$(H \cup \{c\})$
14: **return** H

Solange bis die neuste Regel die Hypothese nicht verbessert

ProbFOIL (2)

Erläuterung: äußere Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Schleifenbedingung: **until** $\text{globalscore}(H) \geq \text{globalscore}(H \cup \{c\})$

- Raedt und Thon nutzen hierfür:

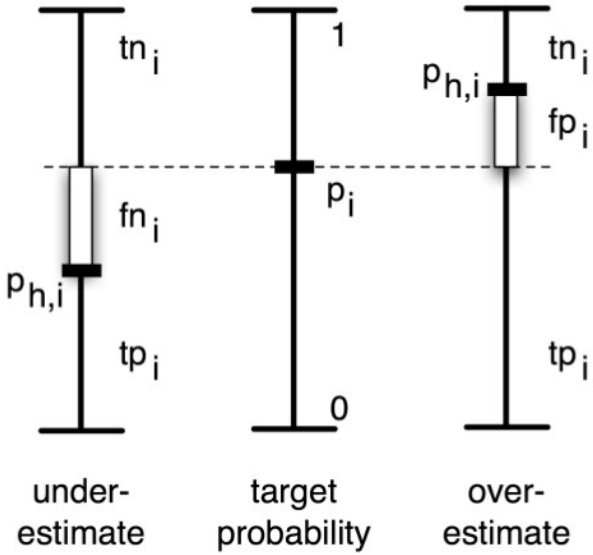
$$\text{globalscore}(H) = accuracy(H) = \frac{TP(H)+TN(H)}{TP(H)+TN(H)+FP(H)+FN(H)}$$

wobei $TP = \sum_i tp_i$ bzw. $TN = \sum_i tn_i$

$FP = \sum_i fp_i$ bzw. $FN = \sum_i fn_i$

mit $tp_i = min(p_i, p_{h,i})$ / $tn_i = min(n_i, n_{h,i})$

$fp_i = max(0, n_i - tn_i)$ / $fn_i = max(0, p_i - tp_i)$



p_i : Wahrscheinlichkeit des Faktums e_i („Positiv-Anteil“)

$n_i = 1 - p_i$: Gegenwahrscheinlichkeit des Faktums e_i („Negativ-Anteil“)

$p_{h,i}$: Vorhersage der Wahrscheinlichkeit eines Faktums e_i unter einer Hypothese h

$n_{h,i} = 1 - p_{h,i}$: Gegenwahrscheinlichkeit eines Faktums e_i unter einer Hypothese h

ProbFOIL (2)

Erläuterung: äußere Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Schleifenbedingung: $\textbf{until}\ \text{globalscore}(H) \geq \text{globalscore}(H \cup \{c\})$

- Raedt und Thon nutzen hierfür:

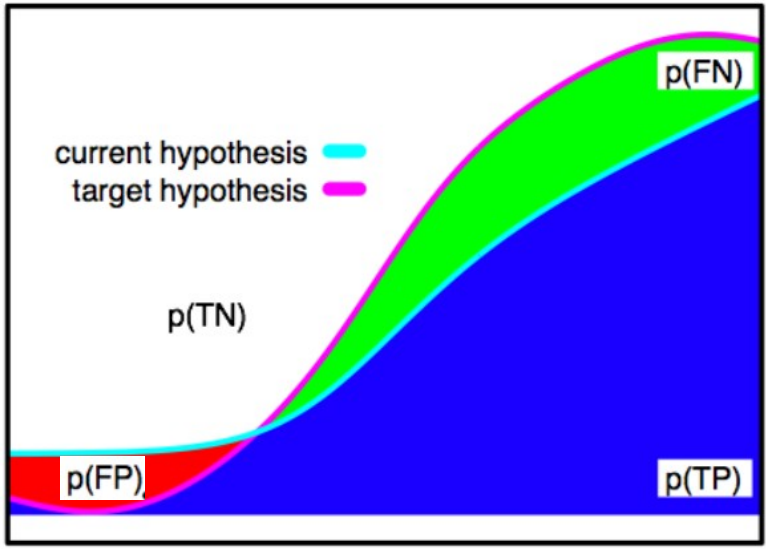
$$\text{globalscore}(H) = accuracy(H) = \frac{TP(H)+TN(H)}{TP(H)+TN(H)+FP(H)+FN(H)}$$

wobei $TP = \sum_i tp_i$ bzw. $TN = \sum_i tn_i$

$FP = \sum_i fp_i$ bzw. $FN = \sum_i fn_i$

mit $tp_i = min(p_i, p_{h,i})$ / $tn_i = min(n_i, n_{h,i})$

$fp_i = max(0, n_i - tn_i)$ / $fn_i = max(0, p_i - tp_i)$



p_i : Wahrscheinlichkeit des Faktums e_i („Positiv-Anteil“)

$n_i = 1 - p_i$: Gegenwahrscheinlichkeit des Faktums e_i („Negativ-Anteil“)

$p_{h,i}$: Vorhersage der Wahrscheinlichkeit eines Faktums e_i unter einer Hypothese h

$n_{h,i} = 1 - p_{h,i}$: Gegenwahrscheinlichkeit eines Faktums e_i unter einer Hypothese h

ProbFOIL (3)

Algorithmus: innere Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

Algorithm 1 The ProbFOIL algorithm

1: $h := t(X_1, \ldots, X_n)$ where t is the target predicate and the X_i distinct variables;
2: $H := \emptyset$; $c := (h \leftarrow b)$; $b := []$;
3: **repeat**
4: $\quad b := []$; initially the body of the rule is empty;
5: $\quad$ **while** $\neg$localstop$(H, h \leftarrow b)$ **do**
6: $\quad\quad l := \arg\max_{l \in \rho(h \leftarrow b)}$ localscore$(h \leftarrow [b, l])$
7: $\quad\quad b := [b, l]$
8:
9:
10:
11:
12:
13: **until** globalscore$(H) \geq$ globalscore$(H \cup \{c\})$
14: **return** H

Refinement → Suche neue Literale

Solange bis die neuste Regel die Hypothese nicht verbessert

ProbFOIL (4)

Erläuterung: innere Schleife

TECHNISCHE UNIVERSITÄT DARMSTADT

- Refinement-Operator ρ (Mapping einer Regel auf Menge von Spezialisierungen) [Rae08]
- Lokaler Score basiert auf m-Estimation [DCP93, Ces90]

$$\text{localscore}(H, c) = \text{m-estimate}(H \cup \{c\}) - \text{m-estimate}(H)$$

wobei $m - estimate(H) = \frac{TP(H)+m*\frac{P}{N+P}}{TP(H)+FP(H)+m}$ [sic!] [RT11]

mit $P = \sum_i p_i$, $N = \sum_i n_i$

m : Gewichtung zwischen Precision (für $m = 0$) und apriori-Verhältnis positiver zu negativer Beispiele

- Lokales Stoppkriterium:

$$\text{localstop}(H, c) = (TP(H \cup \{c\}) - TP(H) = 0) \vee (FP(\{c\}) = 0)$$

→ d.h. beende Schleife, wenn das neue Literal c

1) keinen neuen **positiven** Anteil eines Beispiels (TP) abdeckt
2) keinen neuen **negativen** Anteil eines Beispiels (FP) abdeckt

ProbFOIL (5)

Algorithmus: Pruning

TECHNISCHE UNIVERSITÄT DARMSTADT

Algorithm 1 The ProbFOIL algorithm

1: $h := t(X_1, \ldots, X_n)$ where t is the target predicate and the X_i distinct variables;
2: $H := \emptyset$; $c := (h \leftarrow b)$; $b := []$;
3: **repeat**
4: $\quad b := []$; initially the body of the rule is empty;
5: $\quad$ **while** $\neg$localstop$(H, h \leftarrow b)$ **do**
6: $\quad\quad l := \arg\max_{l \in \rho(h \leftarrow b)}$ localscore$(h \leftarrow [b, l])$
7: $\quad\quad b := [b, l]$
8: $\quad$ let $b = [l_1, \ldots, l_n]$
9: $\quad i := \arg\max_i$ localscore$(H, h \leftarrow l_1, ..., l_i)$;
10: $\quad c := p(X_1, \ldots, X_n) \leftarrow l_1, \ldots, l_i$;
11: $\quad$ **if** globalscore$(H) <$ globalscore$(H \cup \{c\})$ **then**
12: $\quad\quad H := H \cup \{c\}$
13: **until** globalscore$(H) \geq$ globalscore$(H \cup \{c\})$
14: **return** H

Refinement → Suche neue Literale

Pruning → Entferne Literale ab lokalem Maximum

Solange bis die neuste Regel die Hypothese nicht verbessert

Experimente (1)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Analyse mittels des Spiels „Eulisis“
  - **Dealer**: legt vor Spielbeginn Regeln für „korrekte Erweiterung“ fest
  - **Spieler**: der Dealer legt dem Spieler nacheinander Spielkarten vor und gibt an, ob diese Karte nach seinen (geheimen) Regeln eine „korrekte Erweiterung“ ist
  - **Ziel des Spiels**: Spieler muss die geheimen Regeln finden

- Erkennung der Spielkarten aus Bildern mittels SIFT / RANSAC
  - Absichtliche Wahl:
    1) Häufige Verwechslung bei Erkennung der Spielkarten
    2) Anzahl erkannter Features wird als proportional zur Wahrscheinlichkeit für den Spielkartentyp aufgefasst

```
0.24::sift(26, karte(Herz,9))
0.40::sift(26, karte(Pik,9))
```

Experimente (2)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Zu lernendes Konzept: Auf schwarze Spielkarten folgen ungerade Karten, auf rote Spielkarten folgen gerade Karten

  → Ergebnis: gefundene Regeln von ProbFOIL

```
trans(PrevPos,Curr) :- schwarz(Prev), ungerade(Curr)
trans(PrevPos,Curr) :- rot(PrevPos), gerade(Curr)

trans(PrevPos,Curr) :- ungerade(Curr), gerade(Curr), rot(Curr)
trans(PrevPos,Curr) :- schwarz(Curr), gerade(Curr), ungerade(Curr)
```

- Letzten beiden Regeln logisch inkonsistent
  - → Entstehung durch Rauschen in der Spielkartenerkennung
  - → Diese Regeln könnten durch Post-Pruning entfernt werden (da nur +0.2% Accuracy)

Kritik

Probabilistisches Lernen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Fehler in Schleifenbedingung der äußeren Schleife
  → Korrigiert in [RT11]

- Keine Betrachtung der Komplexität/des Laufzeitverhaltens
  → Nur Sekundenangaben ohne Angabe der Hardwarekonfiguration
  → Korrigiert in [RT11]: 3 GHz PC, 2GB Ram
  → Trotzdem: keine Angaben zur Komplexität (interessant, da bei ProbFOIL kein „Seperate“-Schritt existiert)

- Welcher m-Wert für m-estimation?
  → Entscheidend für die resultierende Regelmenge

- Experimente mit „Eulisis“ schwer vergleichbar
  → Weitergehende Analyse mit vergleichbaren Beispielmengen wünschenswert

Ranking mittels Klassifikationsregeln

Jianping Zhang, Jerzy W. Bala, Ali Hadjarian und Brent Han

Motivation

Ranking mittels Klassifikationsregeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- Problem 1: Regellerner können häufig nur klassifizieren

  `surfen(t)` **entweder** `wahr` **oder** `falsch`

  → kein weiteres Ranking von gleich-klassifizierten Beispielen möglich

- Problem 2: vorhandene Rankingverfahren weisen häufig den gleichen Score zu, wenn zwei Beispiele gleich gut zur Regelmenge passen

⟶ **Entwurf eines Regel-Ranking-Frameworks**

`surfen(heute) |= falsch` → **`Score: 10`**

`surfen(morgen) |= wahr` → **`Score: 90`**

`surfen(übermorgen) |= wahr` → **`Score: 75`**

→ hilft bei der Entscheidung, ob / wann man surfen gehen sollte

Methoden (1)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Zhang et al. schlagen drei Methoden zur Berechnung von Rankings vor:

1) Hybride Methoden

→ Integration von Lernalgorithmen mit Ranking-Fähigkeiten in Regellerner (z.B. Perceptron-Algorithmus, Naive-Bayes, Instance-based Learning)

2) Rule Ensembles und redundante Regeln

→ Nutzung von Ensemble Learning: Kombination vieler Klassifikatoren durch Mehrheitsvotum / Mittelwert (z.B. Bagging, Boosting)

→ Nutzung von redundanten Regeln (ermöglichen feineres Ranking)

3) Post-Analyse von Regeln

→ Nutzung eines normalen Regellerners und nachträgliche Berechnung eines Rankings

Methoden (2)

Post-Analyse von Regeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Gegeben:** vorhandene Regelmenge, Zusatzinformationen (z.B. Anzahl abgedeckter positiver/negativer Beispiel pro Regel)

→ **1) Wahrscheinlichkeitsschätzung**

(i) Precision

$$\frac{k}{n}$$

- (Anzahl abgedeckter Beispiele irrelevant)

(ii) Laplace-Korrektur

$$\frac{k+1}{n+C}$$

- (gleicher Score für gleich gut abgedeckte Beispiele)

mit k/n: Anzahl positiver/gesamt abgedeckter Beispiele
C : Anzahl Klassen

(iii) F-Measure

$$F-measure(r) = \frac{\beta^2+1}{\frac{\beta^2}{recall(r)} + \frac{1}{precision(r)}}$$

mit $\beta < 1$: mehr Precision
$\beta > 1$: mehr Recall (TP / (TP+FN))

[LG94]

→ großes β bevorzugt generellere aber weniger präzise Regeln

Methoden (3)

Post-Analyse von Regeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Gegeben:** vorhandene Regelmenge, Zusatzinformationen (z.B. Anzahl abgedeckter positiver/negativer Beispiel pro Regel)

→ **2) Geometrische Methoden**

- Problem: Bei kleiner Regelmenge ist ein feines Ranking schwer

> **Szenario**: 1) nur 0,1% der Fälle überhaupt interessant
> 2) alle Regeln in der Regelmenge decken jedoch mehr als 0,1% dieser Beispiele ab
>
> → Problem: Ranking der interessanten Fälle nicht möglich

→ abgedeckte Beispiele: je näher an der Entscheidungsgrenze (der Regel), desto **geringer** ist der Score

→ nicht abgedeckte Beispiele (Partial Matching): je näher an der Entscheidungsgrenze (der Regel), desto **höher** ist der Score

⟶ **Berechnung eines Übereinstimmungsgrades**

Ranking mit mehreren Regeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Gegeben:** Regelmenge `RS`, Ranking `score(e, R)` für ein Beispiel `e` und eine einzelne Regel `R`
- **Gesucht:** Ranking für ein Beispiel `e`, welches Menge von Regeln `RS` abgedeckt

→ drei Methoden:

1) Max $$score(e, RS) = \max_{i=1}^{l} score(e, R_i)$$

2) Mittelwert $$score(e, RS) = \frac{1}{l} \cdot \sum_{i=1}^{l} score(e, R_i)$$

3) Probabilistische Summe

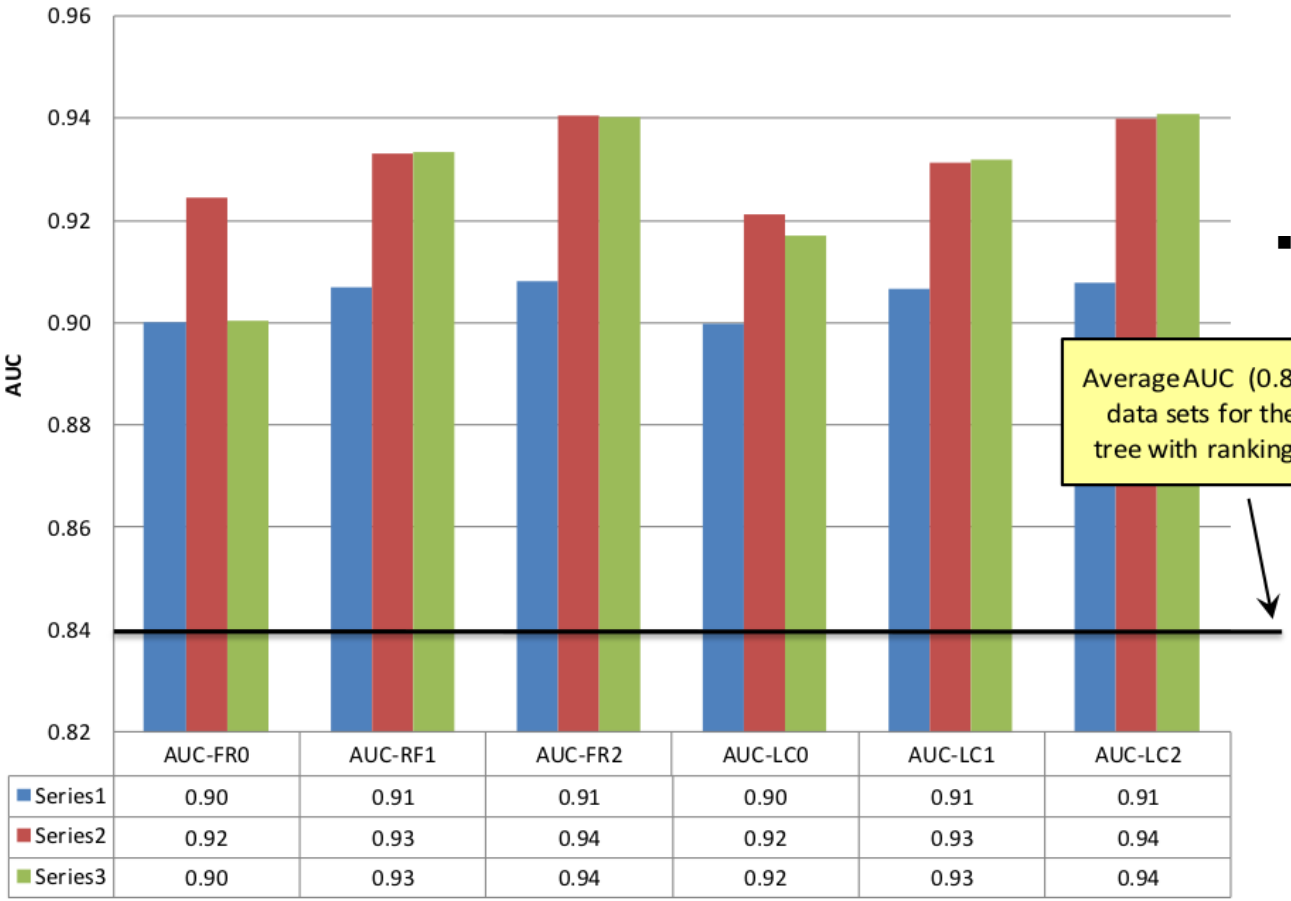
$$score(e, \{R_1\}) = score(e, R_1)$$

$$score(e, \{R_1, \ldots, R_n\} = score(e, \{R_1, \ldots, R_{n-1}\}) + score(e, R_n) - score(e, \{R_1, \ldots, R_{n-1}\}) \times score(e, R_n)$$

⟶ **Mittelwert / Probabilistische Summe produzieren feineres Ranking**

Experimente (1)

AUC (Area Under the Curve) von ROC-Kurven (FPR vs. TPR)



	AUC-FR0	AUC-RF1	AUC-FR2	AUC-LC0	AUC-LC1	AUC-LC2
Series1	0.90	0.91	0.91	0.90	0.91	0.91
Series2	0.92	0.93	0.94	0.92	0.93	0.94
Series3	0.90	0.93	0.94	0.92	0.93	0.94

- Serien:
  - 1) generelle Regeln
  - 2) gemischte Regeln
  - 3) spezifische Regeln
- Methoden:
  - 1) F-Measure
    - + Max → FR0
    - + Mittelwert → FR1
    - + Probabilistische Summe → FR2
  - 2) Laplace-Korrektur
    - + Max → LC0
    - + Mittelwert → LC1
    - + Probabilistische Summe → LC2

- Regellerner: RIPPER / Entscheidungsbaum-Lerner: ID3 mit Laplace-Korrektur

Experimente (1)

Details: AUC von ROC-Kurven (FPR vs. TPR)

Dataset	AUC-FR0	AUC-RF1	AUC-FR2	AUC-LC0	AUC-LC1	AUC-LC2
Series 1	R=0.19					
D1	0.98	0.98	0.98	0.98	0.98	0.98
D2	0.77	0.79	0.79	0.77	0.78	0.79
D3	0.92	0.92	0.92	0.92	0.92	0.92
D4	0.82	0.84	0.84	0.82	0.84	0.84
D5	1.00	1.00	1.00	1.00	1.00	1.00
D6	0.91	0.91	0.91	0.91	0.91	0.91
Averages	**0.90**	**0.91**	**0.91**	**0.90**	**0.91**	**0.91**
Series 2	R=0.25					
D1	0.99	0.99	0.99	0.99	0.99	0.99
D2	0.84	0.85	0.88	0.83	0.85	0.87
D3	0.95	0.95	0.96	0.95	0.95	0.96
D4	0.85	0.88	0.89	0.84	0.87	0.89
D5	1.00	1.00	1.00	1.00	1.00	1.00
D6	0.92	0.93	0.93	0.92	0.93	0.93
Averages	**0.92**	**0.93**	**0.94**	**0.92**	**0.93**	**0.94**
Series 3	R=0.28					
D1	0.92	0.99	0.99	0.95	0.99	1.00
D2	0.84	0.85	0.87	0.83	0.85	0.87
D3	0.96	0.96	0.96	0.95	0.95	0.96
D4	0.85	0.88	0.89	0.84	0.87	0.89
D5	0.90	1.00	1.00	1.00	1.00	1.00
D6	0.93	0.93	0.93	0.93	0.93	0.93
Averages	**0.90**	**0.93**	**0.94**	**0.92**	**0.93**	**0.94**

TECHNISCHE UNIVERSITÄT DARMSTADT

- F-Measure (FR)
- Laplace-Korrektur (LC)
  - + Max → 0
  - + Mittelwert → 1
  - + Probabilistische Summe → 2
- D1: Breast Cancer
- D2: Chess (2 class)
- D3: German Credit
- D4: Japanese Credit
- D5: Magic
- D6: Yeast

D1	0.948736
D2	0.779182
D3	0.891147
D4	0.722007
D5	0.999982
D6	0.705955
Averages	**0.84**

DCT-Ergebnisse

Experimente (2)

Ergebnisse

TECHNISCHE UNIVERSITÄT DARMSTADT

- Entscheidungsbaum-Ranking hat niedrigeren AUC-Wert als Regel-Ranking

- Spezifische Regeln resultieren in größerem AUC-Wert
- Spezifische Regeln generieren mehr Scores als generelle Regeln
- AUC-Wert korreliert zur Feinheit der Scores

- Probabilistische Summe erzielt den besten AUC-Wert
- Laplace-Korrektur für einfache Wahrscheinlichkeitsschätzung nicht signifikant

Kritik

Ranking mittels Klassifikationsregeln

TECHNISCHE UNIVERSITÄT DARMSTADT

- Paper schlecht strukturiert
  - Hinweise auf Abbildungen fehlen
  - Inkonsistente Terminologie
- Nur intuitive Begründungen
  - Wahl von F-Measure nur intuitiv begründet
  - Methoden für Kombination mehrerer Regeln (Max, Mittelwert, Prob. Summe) unbegründet
- Experimentelle Studie positiv
  - Klar strukturiert
  - Verschiedene Serien, um generelle/spezielle Regelmengen zu erhalten
- Genauere Analyse der geometrischen/hybriden Methoden wünschenswert

Kombination

Ranking mittels probabilistischem Regellernen?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Probabilistisches Regellernen für Ranking geeignet
  - Berechnung eines Scores mittels ProbLog
    (Score entspricht berechneter Erfolgswahrscheinlichkeit)
  - Auswertung beachtet außerdem alle relevanten Regeln für ein Beispiel

→ In der Terminologie von Zhang et al. wäre ProbFOIL eine hybride Methode
(Erweiterung eines Regellerners um Ranking-Fähigkeit)

Zusammenfassung

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Probabilistisches Regellernen** (PRL): Erweiterung von Regellernern um unsichere Fakten
  → **ProbFOIL**: Probabilistische Erweiterung von FOIL zur Berechnung von Regelmengen unter Berücksichtigung unsicherer Fakten
  → Gut geeignet für Arbeit mit rauschenden Daten (Bilder, Messwerte, ...)

- **Ranking**: Berechnung eines Scores pro Beispiel
  - Erweiterung auf Regellerner:
    - Nutzung der **F-Measure** (Recall vs. Precision) zur Score-Berechnung
    - Kombination des Scores von Beispielen, die durch mehrere Regeln abgedeckt werden (**Max, Mittelwert, Probabilistische Summe**)
    - Experimente: 1) Probabilistische Summe schneidet am besten ab
      2) Regel-basierter Ansatz besser als Ansatz mit Entscheidungsbäumen

Literaturverzeichnis

- [DCP93] Sašo Džeroski, Bojan Cestnik, and Igor Petrovski. 1993. Using the m-estimate in rule induction. J. Comput. Inf. Technol. 1, 1 (March 1993), 37-46.
- [Ces90] B. Cestnik. Estimating Probabilities: A Crucial Task in Machine Learning. In L. Aiello (ed.) Proceedings of the 9th European Conference on Artificial Intelligence (ECAI-90), pp. 147–150, Stockholm, Sweden, 1990.
- [Rae08] Luc De Raedt. 2008. Logical and Relational Learning: From ILP to MRDM (Cognitive Technologies). Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- [RT11] Luc De Raedt, Ingo Thon, Probabilistic Rule Learning. In Paolo Frasconi, Francesca A. Lisi (eds.). Inductive Logic Programming - 20th International Conference, ILP 2010, Florence, Italy, June 27-30, 2010. Revised Papers. Springer, pp. 47-58, 2011.
- [LG94] David D. Lewis and William Gale. Training text classifiers by uncertainty sampling. In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-94), pages 3-12, 1994.