

# TUD 2008 Poker Challenge

„Creating an SVM to play Strong Poker”

Blank, Soh, Scott

-

„Explaining Winning Poker - A Data Mining Approach”

Johansson, Sönströd, Niklasson

-

Benjamin Herbert

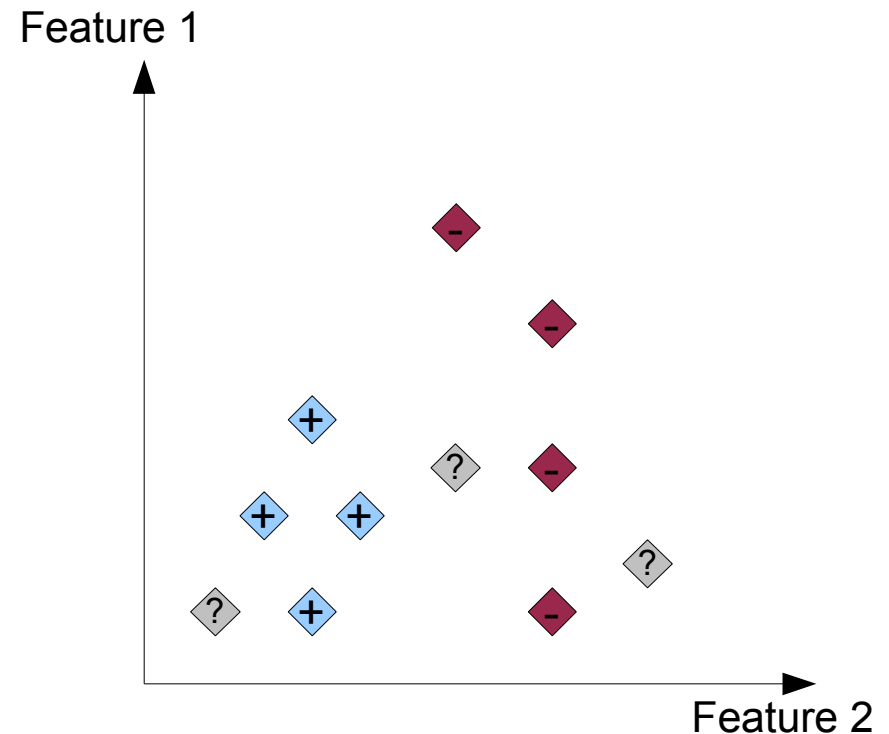
*bherbert@rbg.informatik.tu-darmstadt.de*

# Agenda

- Einführung
- Klassifikation
- Experimente 1
- Entscheidungsbäume
- Experimente 2
- Fazit/Ausblick
- Links

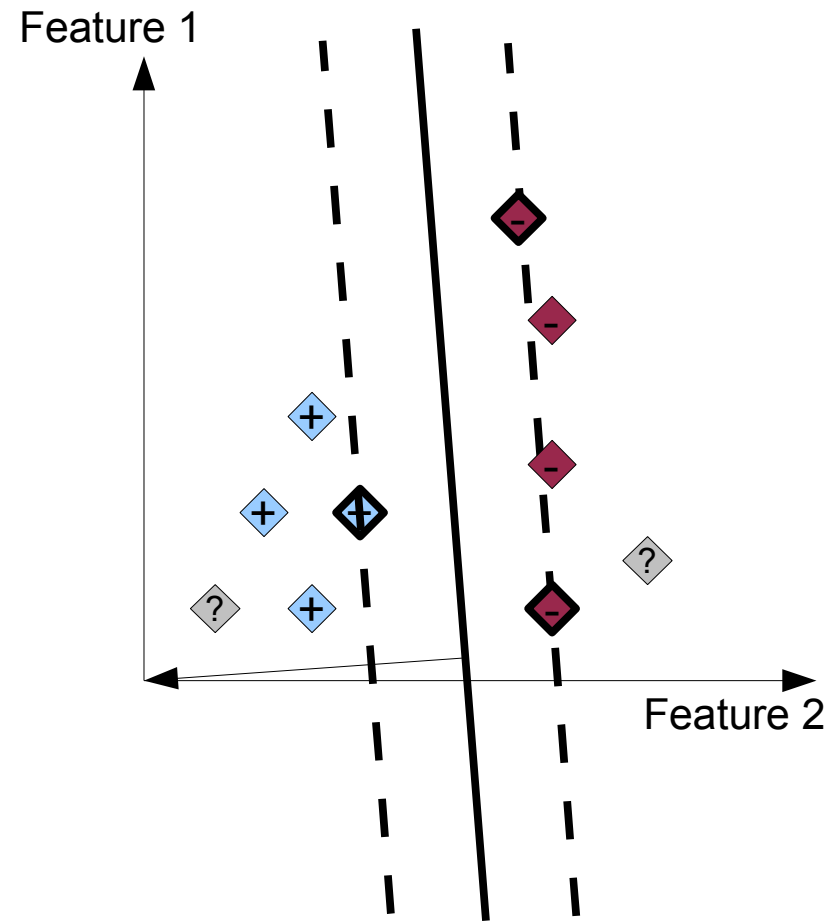
# Klassifikation

- Menge von Beispielen
- Beispiele gegeben durch Merkmale und zugehörige Klasse
- Tupel der Form (**Feature**, Klasse)
- Trainieren eines Klassifikators
- Eingabe **Feature**
- Ausgabe **Klasse**\*



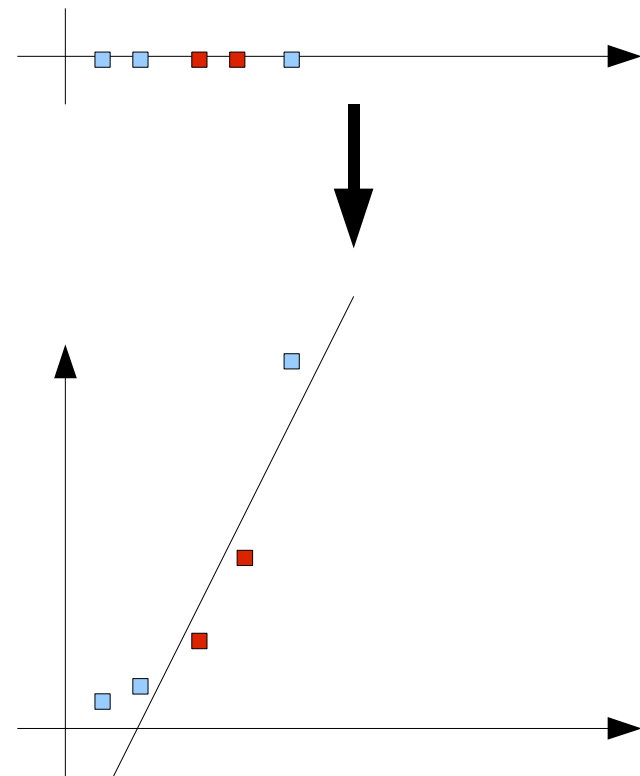
# Support Vector Machines

- Beispiele als Punkte im Featureraum
- Hyperebene finden, die die Punkte der zwei Klassen trennt
- Dabei Abstand zu den jeweils nächsten Punkten  $p$  maximieren
- “Maximum Margin”
- Sicherheitsbereich



# Support Vector Machines

- Quadratic Optimization Problem
- Nicht linear separierbare Probleme können durch Transformation in höherdimensionale Vektorräume linear separierbar werden
- Viele Implementierungen verfügbar
- Parameterwahl oft durch Cross-Validation



# SVM<sup>light</sup> *in Aktion*

- Ziel: Strategie eines Bots nachahmen
- Spieldaten
  - Neun Bots spielen gegeneinander in Poker Academy
    - ~5000 Hände
    - “Hari” – eine Simbot Version gewinnt
  - Spieldaten des besten Bots werden verwendet
    - Parsen des Logs von Poker Academy
    - Berechnung der Features
    - Welche Aktion wurde gewählt
- Beispiele
  - (**Feature**, Klassen)

# SVM<sup>light</sup> in Aktion

- Features
  - Hand Strength
  - Pot Odds
  - Positive Potential
    - Straight Draw
    - Flush Draw
    - Potential
- Klassen
  - Fold
  - Check/Call
  - Bet/Raise



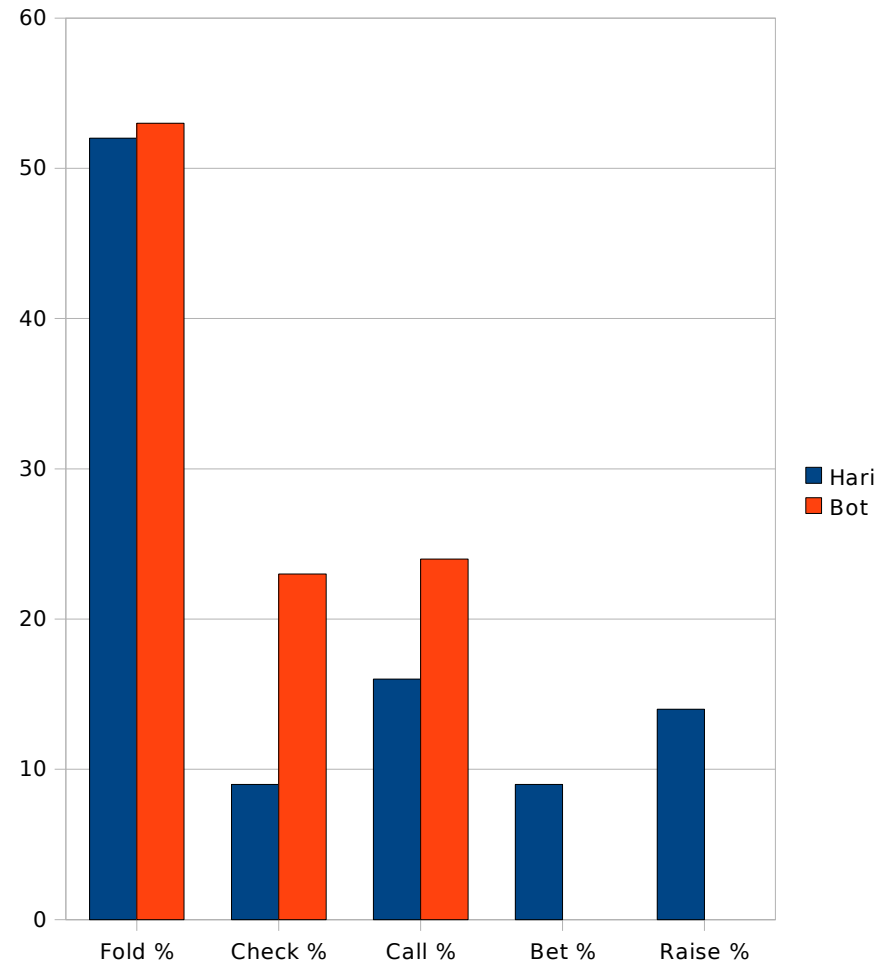
# SVM<sup>light</sup> in Aktion

- Erlernen von drei binären Klassifikatoren mit SVM<sup>light</sup>
  - Fold – Check/Call
  - Fold – Bet/Raise
  - Check/Call – Bet/Raise
- Parameterwahl für SVM<sup>light</sup>
  - 2194 Entscheidungen im Training Set
  - 366 Entscheidungen im Test Set
  - Parameter mit bester Accuracy wird benutzt
- Modelle werden durch MultiClass Klassifikator benutzt
  - Aktion Mehrheitsentscheidung
  - Bei Unentschieden wird sicherste Klassifikation gewählt



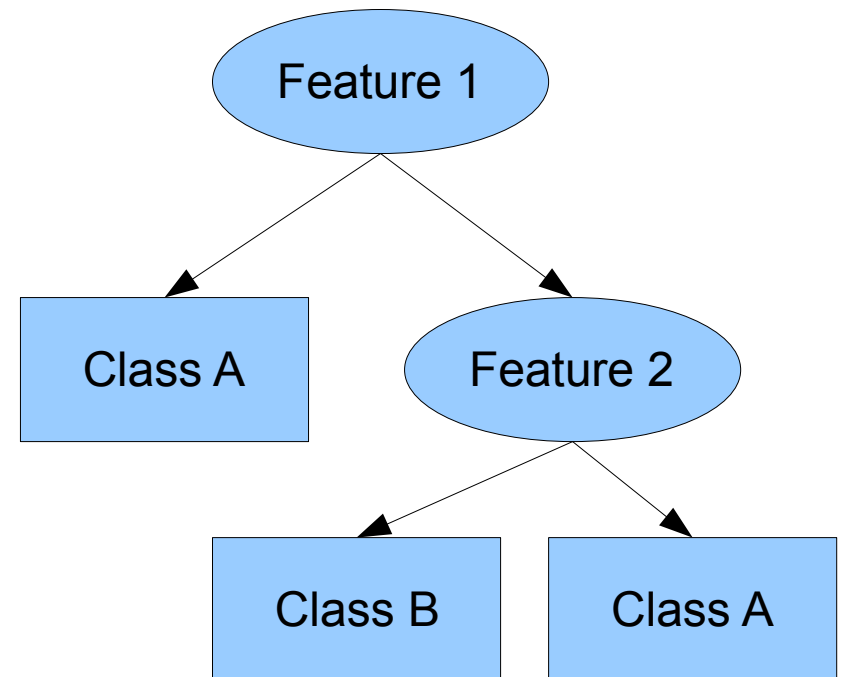
# SVM<sup>light</sup> in Aktion - Ergebnis

- Test gegen Poker Academy Bots
- „Hari” spielt auch mit
- Aktuelle Spielsituation wird erfasst
- Mit gelernten Modellen *berechnete* Aktion wird durchgeführt
- **0,53 Small Bets** pro Hand  
**Verlust**
- Hari gewinnt immerhin  
**0,103 Small Bets** pro Hand



# Entscheidungsbäume

- Entscheidungsknoten
- Klassifikationsergebnis als Blätter
- Verschiedene Heuristiken um Baum zu erlernen
  - “Beste” Features weiter oben
  - Information Gain
- Training mit Beispielen
- Klassifikation durch Traversieren



# Explaining Winning Poker

- Studie auf Spieldaten
  - Ladbroke's Online Poker
  - 105 Spieler
  - Mindestens 500 Hände pro Spieler (Ø 1126 Hände)
- Anwendung von Data Mining Methoden um Erklärungen für erfolgreiches Spielen zu finden
  - Regellerner G-REX
  - Entscheidungsbaumlerner J48 (Wekas Implementierung von C4.5)
- “Concept Description”

# Datenextraktion

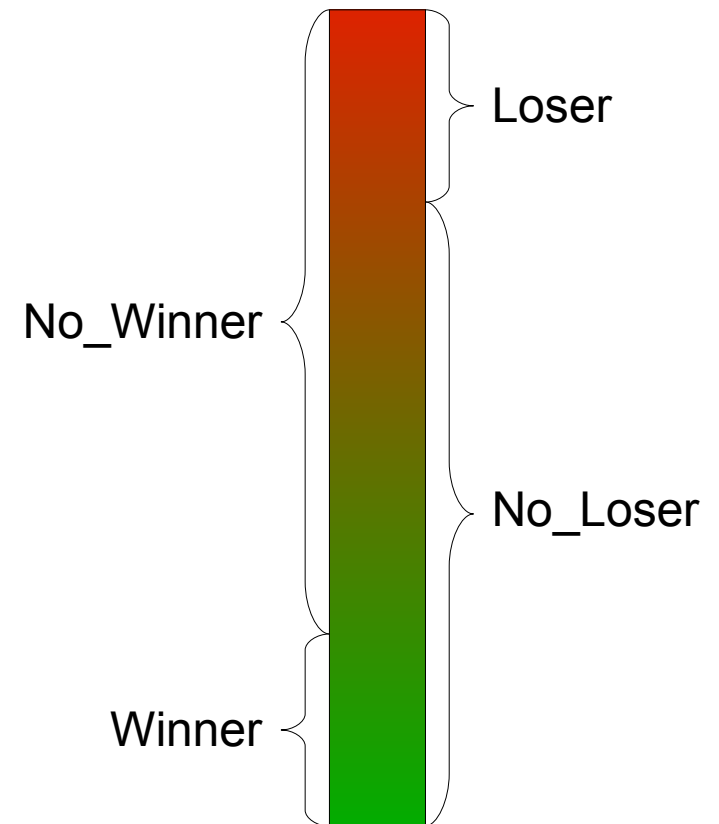
- Importieren der Daten aus Ladbrokes Poker mit PokerOffice
- Beobachtung von Spielen
  - März bis Mai 2006
  - **Alle** 6 Spieler Tische - rund um die Uhr
  - Limit \$0.5 / \$1
  - Bester Spieler **\$645** Dollar Gewinn
  - Schlechtester Spieler **\$357** Verlust
  - Ø **\$15,18** Verlust
- Speichern in MySQL Datenbank
- Berechnung von verschiedenen Features

# Attribute

- CPRE – CallPreFlop%
- AFPRE – AggressionFactorPreFlop
- AFPOST – AggressionFactorPostFlop
- SF – SawFlop%
- FT – FlopTurn%
- TR – Turn-River%
- SDN – Showdown%
- PFR – PreFlopRaise%
- FIR – FirstInRaise%
- CR – CheckRaisePerHand
- CC- ColdCallPerHand

# Experimente

- Klassen
  - Winner / No\_Winner
  - Loser / No\_Loser
- Verschiedene „Sortierungen“
  - WIN: Gesamtgewinn
  - WIN\_AVG: Gewinn pro Hand
  - LOSE: Gesamtverlust
  - LOSE\_AVG: Verlust pro Hand
- „Top“ 25 Spieler in Zielklasse
- Jeweils Modell erstellen mit/von allen Datensätzen



# Ergebnisse

	WIN		WIN_AVG		LOSE		LOSE_AVG	
	Acc.	Size	Acc.	Size	Acc.	Size	Acc.	Size
J48	88.6	15	89.5	11	76.2	1	78.1	3
G-REX	91.4	11	90.5	19	90.5	21	93.3	13

- In LOSE ordnet J48 alle Spieler als No\_Losers ein
- In LOSE\_AVG gibt es nur eine einzige Entscheidung

CPRE  $\leq$  0.4609: No\_Loser (65/4)

CPRE  $>$  0.4609: Loser (40/19)

- *Accuracy*:  $1 - (\text{falsch klassifiziert}/n)$
- *Size*: Anzahl Knoten/Blätter

# Ergebnisse

	WIN		WIN_AVG		LOSE		LOSE_AVG	
	Acc.	Size	Acc.	Size	Acc.	Size	Acc.	Size
J48	88.6	15	89.5	11	76.2	1	78.1	3
G-REX	91.4	11	90.5	19	90.5	21	93.3	13

- G-REX hat höhere Genauigkeit
- Dafür komplexere Regeln



# G-REX für WIN

FT > 0.7265: No\_Winner (41/3)

FT <= 0.7265

CR <= 0.0078 No\_Winner (26/3)

CR > 0.0078

TR > 0.7625

SDN <= 0.2374: Winner (5/0)

SDN > 0.2374: No\_Winner (2/0)

TR <= 0.7625

PFR <= 0.1010: No\_Winner (18/2)

PFR > 0.1010: Winner (13/1)

# G-REX für WIN

FT > 0.7265: No\_Winner (41/3)

FT <= 0.7265

CR <= 0.0078 No\_Winner (26/3)

CR > 0.0078

TR > 0.7625

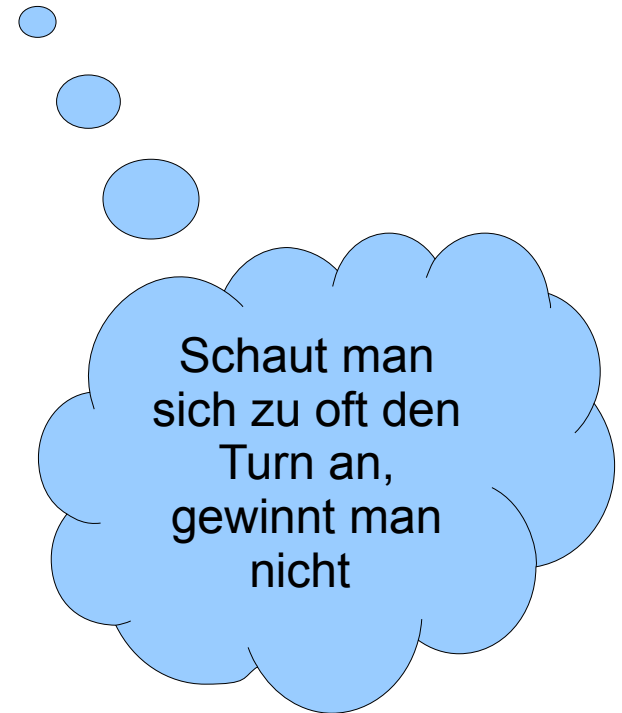
SDN <= 0.2374: Winner (5/0)

SDN > 0.2374: No\_Winner (2/0)

TR <= 0.7625

PFR <= 0.1010: No\_Winner (18/2)

PFR > 0.1010: Winner (13/1)



# G-REX für WIN

FT > 0.7265: No\_Winner (41/3)

FT <= 0.7265

CR <= 0.0078 No\_Winner (26/3)

CR > 0.0078

TR > 0.7625

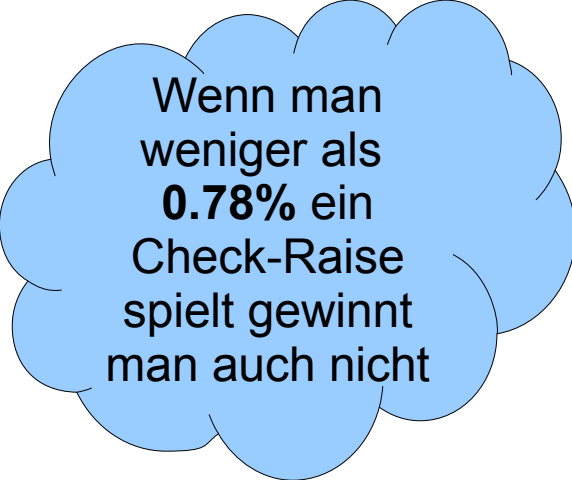
SDN <= 0.2374: Winner (5/0)

SDN > 0.2374: No\_Winner (2/0)

TR <= 0.7625

PFR <= 0.1010: No\_Winner (18/2)

PFR > 0.1010: Winner (13/1)



Wenn man  
weniger als  
**0.78%** ein  
Check-Raise  
spielt gewinnt  
man auch nicht

# Fazit

- Creating an SVM to Play Strong Poker
  - Spieler und seine Aktionen beobachten
  - Mit SVM<sup>light</sup> mehrere Klassifikatoren erstellen und Aktion vorhersagen lassen
  - Ergebnis war ein schlecht spielender Bot
  - Hat 0.53 Small Bets pro Hand verloren
- Explaining Winning Poker – A Data Mining Approach
  - J48 bildet kompakte Regeln (dank Pruning)
  - Relativ gute Konzeptbeschreibung durch G-REX
  - G-REX Regeln sind konform mit allgemeinem Pokerwissen
  - Kleine Datengrundlage

# Ausblick

- Mehr Features betrachten
  - Position ist elementar
- Gegner grob einschätzen
- Vielleicht sinnvolle Regeln findbar aus größeren Datenmengen
- SVM<sup>light</sup> und Weka nützlich für eigenen Bot
  - Siehe Handout



# Literatur

- <http://www.learning-with-kernels.org/>
- <http://paul.rutgers.edu/~dfradkin/papers/svm.pdf>
- <http://svmlight.joachims.org/>
- <http://www.cs.waikato.ac.nz/ml/weka/>

**Vielen Dank für die Aufmerksamkeit!**