Data-Mining Routinen in SAP BI 7.0

Norbert Ketterer
Fachbereich Angewandte Informatik
Hochschule Fulda
36039 Fulda
E-mail: norbert.ketterer@informatik.hs-fulda.de

STICHWORTE
Data-Mining, SAP-BI 7.0, Klassifikation, Neuronale Netzwerke, Benchmarking, RapidMiner.

ABSTRACT

In der vorliegenden Arbeit werden ausgewählte klassifizierende Data-Mining-Verfahren aus SAP BI 7.0 und RapidMiner an Hand etablierter Benchmarks analysiert. Schließlich wird eine Implementierung eines alternativen Data-Mining-Verfahrens in SAP-BI vorgestellt, welches auf neuronalen Netzen basiert.
Dieses Paper ist im Wesentlichen eine Wiedergabe der bereits 2012 im Rahmen der AKWI-Tagung in Pforzheim erschienenen Arbeit von Augustin und Ketterer durch einen der Autoren (Augustin& Ketterer 2012).

EINLEITUNG

Die SAP AG bietet bereits seit geraumer Zeit in deren Business-Intelligence-Produkt SAP-BI eine Auswahl von Funktionen an, die Standardmethoden des Data-Mining implementieren. Die Methode der „Klassifikation“ wird dabei in der aktuellen Version von SAP-BI durch eine Data-Mining-Funktion implementiert, die im Wesentlichen entropiebasiert einen Entscheidungsbaum erzeugt (SAP 2007) – die Funktionsweise ist ähnlich dem bekannten C 4.5 Algorithmus; alternative Klassifikationsverfahren – etwa neuronale Netze - sind dagegen nicht Teil des von SAP-BI bereitgestellten Funktionsumfangs. Weitere wesentliche Data-Mining-Verfahren – speziell in Anlehnung an die Verfahrensübersicht aus (Ester& Sander 2009)– sind in SAP-BI: Clusteranalyse über ein Verfahren ähnlich dem „K-Means-Algorithmus“, Assoziationsanalyse über ein Verfahren ähnlich dem „A-Priori-Algorithmus“ sowie eine Regressionsanalyse.
RapidMiner deckt dagegen eine größere Bandbreite von Klassifikationsverfahren ab; hier stellt etwa der C 4.5-Algorithmus nur eine Alternative zwischen einer Reihe von Verfahren dar, die u.a. auch auf neuronalen Netzen basieren.

Das Ziel dieser Arbeit besteht zum einen darin, die SAP-BI Data-Mining-Lösung zur Klassifikation auf Basis etablierter, veröffentlichter Benchmarks bezüglich seiner Klassifikationsgenauigkeit zu bewerten; eine ähnliche Analyse wird zudem für RapidMiner – hier für den C 4.5 Algorithmus sowie ein neuronales Netz mit einem verborgenen Layer – durchgeführt.
Zum anderen wird die Implementierung eines alternativen Verfahrens des Data-Mining in SAP-BI skizziert, welches ebenfalls auf neuronalen Netzen mit einem verborgenen Layer basiert. Das Benchmarking dieser Implementierung wird hier wiederum detailliert dokumentiert.

KLASSIFIZIERNDE DATA-MINING-VERFAHREN IN SAP-BI UND RAPIDMINER

Bei Anwendung eines Klassifikationsverfahrens sind, anders als etwa beim Clusterverfahren, die Klassen bereits bekannt. Es ist Aufgabe des Klassifikationsverfahrens, Objekte auf Basis von Attributwerten den Klassen zuzuordnen. Hierbei ist ein Klassifikationswissen aufzubauen, welches die Klassenstruktur beschreibt (Ester& Sander 2009).
Der Klassifikationsprozess besitzt anfangs eine Trainingsphase, die dem Aufbau des Klassifikationswissens dient; nach der Trainingsphase soll das Wissen ausreichen, neue Objekte selbständig in das Klassenschema einzuordnen. Das Training erfolgt dabei auf Basis historischer Daten für die die Klassenzuordnung bereits bekannt ist, die Trainingsdaten sollten dabei eine ähnliche Klassenverteilung besitzen wie die zu analysierenden Daten.

Bekannte Klassifikationsverfahren stellen der C4.5 Algorithmus von Quinlan (Quinlan 1986) sowie sein Vorgänger ID3 dar. Bei diesen Algorithmen wird versucht, die Ausgangsmenge schrittweise in Klassen aufzuteilen, wobei in jedem Schritt der maximale Informationsgewinn erzielt werden soll. Ein bei diesen Schritten generierter Entscheidungsbaum repräsentiert dann das Klassenwissen. Eine frühe Darstellung der Idee findet sich beispielsweise in Quinlans Veröffentlichung (Quinlan 1986) eine grundlegende Darstellung im Buch von Ester und Sander (Ester& Sander 2009) ein Vergleich von C4.5 mit dem Vorgänger ID3 und dem Nachfolger C5 in der Arbeit von Shahnaz (Shahnaz 2006).

Für die Entropie und den Informationsgewinn werden bei diesen Verfahren typischerweise Definitionen wie folgt verwendet (Ester& Sander 2009):

$$Entropie(T) = -\sum_{i=1}^{k} p_i log_2 p_i \quad (1)$$

$$Informationsgewinn\ (T,\ A) = Entropie(T) - \sum_{i=1}^{m} \frac{|T_i|}{|T|}\ Entropie(T_i) \quad (2)$$

Weitere Klassifikationsverfahren basieren neben Entscheidungsbäumen auf - siehe etwa (Maimon, 2010) - Bayes'schen Netzen, Support Vector Maschinen und Neuronalen Netzen.
Während in SAP BI 7.0 lediglich ein Verfahren zur Klassifikation – nämlich ein entropiebasiertes Entscheidungsbaumverfahren - implementiert wurde, finden sich in RapidMiner neben dem klassischen Entscheidungsbaum nach C4.5 unter anderem auch verschiedene entscheidungsbaumbasierte Modelle – neben C 4.5 und dem älteren ID3 auch CHAID, Bayes'sche Modelle, Support-Vektor-Modelle, Regelbasierte Modelle wie „Ripper" – siehe hier zu Cohen (Cohen 1995) - sowie Neuronale Netzwerke. Es kann hier neben einem einschichtigen Perzeptron auch ein Multi-Layer Netzwerk aufgebaut werden.

BENCHMARKS FÜR KLASSIFIKATIONSROUTINEN

Das UCI-Repository listet eine Reihe von Benchmarks, die sich bezüglich der Art der Daten (etwa „stetig", „diskret" und dem „Grad der Separierbarkeit") stark unterscheiden. Diese Benchmarks bilden die Referenz für eine Vielzahl von Veröffentlichungen, von denen pro Benchmark eine Reihe von Referenzen im Repository angegeben werden (UCI 2012).

Name in UCI (Name in Ref)	Elemente	Klassen
IRIS-Data-Set (IRIS) (Opitz 1999)	**150**	**3**
Congressional Voting Records Data Set (House-Votes-84) (Opitz 1999)]	**435**	**2**
Heart Disease Data Set (Heart-Cleveland) (Opitz 1999)]	**303**	**2**
Chess (King-Rook vs. King-Pawn) Data Set (kr-vs-kp) (Opitz 1999)]	**3092**	**2**
Wine Data Set (Wine) (Rokach 2008)	**178**	**3**
Census Income (Adult) (Kohavi 1996)	**32561**	**2**
Zoo (Zoo) (Rokach 2008)	**101**	**7**

Tabelle 1: Untersuchte Benchmarks des UCI-Repositories (UCI 2012) – Referenzname in Literatur in Klammern

Name in UCI (Name in Ref)	Bemerkung
IRIS-Data-Set (IRIS) (Opitz 1999)	Klassifikation der IRIS-Blumen basierend auf der Länge der Kelch/ Kronblätter. Eine Klasse ist linear separierbar von den übrigen, die anderen zueinander nicht. Daten sind stetig.
Congressional Voting Records Data Set (House-Votes-84) (Opitz 1999)]	Klassen unterscheiden "Republikaner" und "Demokraten" abhängig von 16 Einzelthemen[3], die verschiedenen Entscheidungsmöglichkeiten wurden dabei verdichtet. Daten sind diskret.
Heart Disease Data Set (Heart-Cleveland) (Opitz 1999)]	Der Datensatz besitzt 76 Parameter[4], die Literatur verwendet oft nur eine Teilmenge (etwa 13 oder 14) – (Opitz 1999) verwendet 13 Parameter. Klassen sind „krank" Ja/ Nein. Daten sind gemischt stetig/ diskret.
Chess (King-Rook vs. King-Pawn) Data Set (kr-vs-kp) (Opitz 1999)]	Entscheidung, ob der weiße Spieler ein Schachspiel auf Basis einer gegebenen Situation gewinnen kann – ein Bauer steht dabei bereits auf A7. Die Feldsituation wird durch 36 diskrete Merkmale beschrieben[5].
Wine Data Set (Wine) (Rokach 2008)	Unterscheiden von 3 Weinen an Hand von 13 Merkmalen, wie Phenolgehalt. Die Merkmale sind stetig.
Census Income (Adult) (Kohavi 1996)	Entscheidung an Hand einer Reihe von Kriterien, ob das Einkommen >50K ist. Die Merkmale sind diskret./ stetig.
Zoo (Zoo) (Rokach 2008)	Tiere werden anhand von 17 – größtenteils boolschen Merkmalen – in 7 Klassen eingeordnet[6]. Die Merkmale sind alle diskret.

Tabelle 2: Untersuchte Benchmarks des UCI-Repositories (UCI 2012) – Referenzname in Literatur in Klammern

BEWERTUNG DER EXISTIERENDEN VERFAHREN

Für sämtliche Benchmarks wurde ein Klassifikationslauf auf Basis von Entscheidungsbäumen in RapidMiner (C 4.5) sowie SAP-BI (Entscheidungsbäume) durchgeführt und mit den Benchmarks der in Tabelle 2 und Tabelle 2 referenzierten Quelle verglichen.

Methoden zum Vergleich von Klassifikatoren

Aus einer Trainingsmenge können sehr unterschiedliche Klassifikatoren gelernt werden - siehe hierzu auch wieder (Ester& Sander 2009). Ein typisches Verfahren zum Vergleich der Klassifikatoren bei einer vergleichsweise kleinen Anzahl von Objekten mit bekannter Klasse stellt die „Kreuzvalidierung" dar. Die k-fache Kreuzvalidierung teilt die Menge der Daten in k gleich große Teilmengen, von denen jeweils k-1 Teilmengen zum Training und die verbleibende Teilmenge zum Test verwendet wird.
Der Klassifikationsfehler wird in Anlehnung an die Methode in der Arbeit von Opitz und Maclin (Optiz&Maclin 1999) als Mittelwert einer 5-maligen Wiederholung einer 10-fachen Kreuzvalidierung gebildet (was einem k = 10 entspricht).

Klassifikationsalgorithmen auf den Datensätzen

Die Entscheidungsbäume sind gemäß Benchmark einem Pruning zu unterziehen – es ergeben sich die in Abbildung 1 dargestellen Abweichungen.

Datensatz	*Rapidminer C4.5*	*SAP-BI DT*	*Benchmark C4.5*
Iris	8% (+ 2.8%)	13.33% (+ 8.13%)	5.2%

Abbildung 1: IRIS Daten, klassifiziert via Entscheidungsbaum, Abweichung von RapidMiner und SAP-BI Entscheidungsbaum (DT) zum Benchmark

Ein Beispiel einer Fehlermatrix direkt aus dem Analyseprozessdesigners des SAP BI zeigt Abbildung 3; für diese spezifische Instanz aller Validierungsschritt wurde eine Genauigkeit von 86% erreicht, den zugeordneten Entscheidungsbaum zeigt Abbildung 2. Zu beachten ist, dass die Anzahl der Sätze aufgrund des Validierungsverfahrens mit k = 10 nur „15“ beträgt und das es sich hier nur um eine Instanz aller Validierungen handelt, wobei der Klassifikationsfehler in diesem Fall auch dem durchschnittlichen Fehler entspricht.

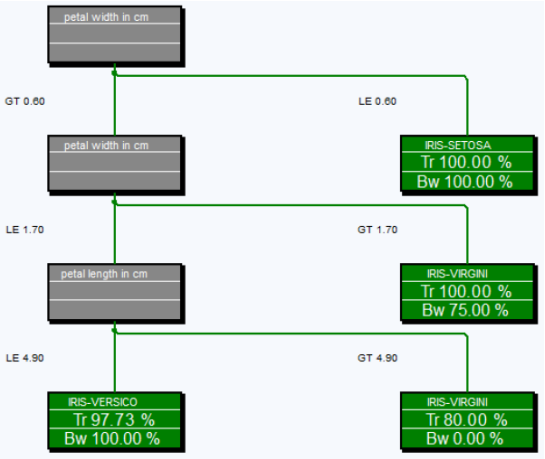

Abbildung 2: Entscheidungsbaum für IRIS-Datensatz (pruned) aus SAP-BI

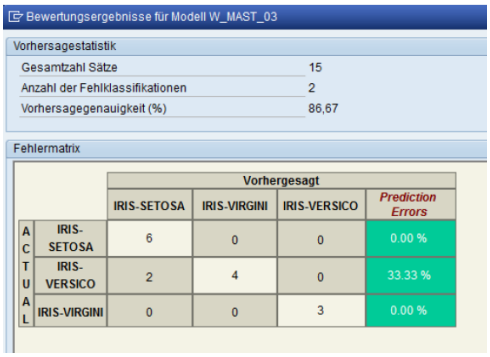

Abbildung 3: Ergebnis eines Validierungslaufes für IRIS

Gemäß der Vorgaben der Benchmark-Literatur ist der Entscheidungsbaum bei den „Voting-“, „Herzerkrankungs-“ und „Schach-Daten“ ebenfalls einem “Pruning” zu unterziehen. Der SAP-Entscheidungsbaum besitzt einen Entscheidungsfehler, der der Benchmark sehr nahe kommt.

Datensatz	*Rapidminer C4.5*	*SAP-BI DT*	*Benchmark C4.5*
House-Votes-84	6.45% (+ 2.85%)	4.6% (+ 1%)	3.6%
Heart-Cleveland	21.14% (- 3.16%)	25.81% (+ 1.51%)	24.3%
Kr-vs-Kp	0.31% (- 0.29%)	0.94% (+ 0.34%)	0.6%
Census-Income	16.51% (+ 1%)	15.38% (- 0.16%)	15.54%
Wine	6.14% (- 7.9%)	11.11% (- 2.93%)	14.04%
Zoo	3.91% (-3.02%)	10% (+ 3.07%)	6.93%

Abbildung 4: übrige Beispieldaten, klassifiziert via Entscheidungsbaum (DT)

ERWEITERUNG VON SAP-BI 7.0 UM EIN NEURONALES NETZ

Erweiterungsmöglichkeiten von SAP BI 7.0 für Hochschulen

Hochschulen und Universitäten betreiben ein SAP BI 7.0 typischerweise beim UCC gemäß Preislistenposition „F2“ oder „F3“ (UCC 2012) – dieses sind „Shared Systeme“, die keine Entwicklungsberechtigung besitzen, sondern lediglich ermöglichen, ein Data-Warehouse über die DW-Modelling-Workbench zu modellieren, sowie Data-Mining-Prozesse mit Hilfe des Analyse Prozess Designers (APD) aufzusetzen.
Innerhalb der Definition eines Analyseprozesses kann jedoch trotzdem eine einfache ABAP-Routine eingebunden werden, die mit Hilfe eines externen Remote Function Call (RFC) in ein anderes System – etwa einem SAP-Entwicklungssystem der Preislistenpostion „G“ – zeigt, welches ABAP-Entwicklungen erlaubt. Somit können komplexe Erweiterungen in das Entwicklungssystem ausgelagert und vom APD über einen RFC referenziert werden. Werden Daten des neuronalen Netzwerks direkt im Data-Warehouse – etwa in einem DSO - gespeichert und muss auf diese im Rahmen des Analyseprozesses zugegriffen werden, kann das mit Hilfe eines BAPIs (BAPI_ODSO_READ_DATA_UC) aus dem Entwicklungssystem erfolgen. Die Spezifikation des RFC kann als Codefragment direkt in die APD-Routine eingebunden werden, die zugehörigen ABAP-Objekte werden direkt von SAP-BI verwaltet; dies gilt nicht als Entwicklungstätigkeit (siehe Abbildung 5 und Abbildung 6).

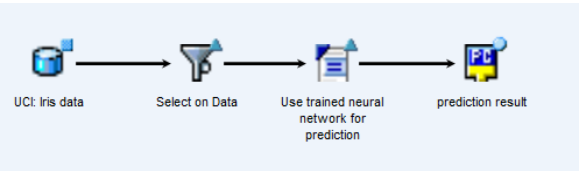

Abbildung 5: Einbindung einer Routine in den APD

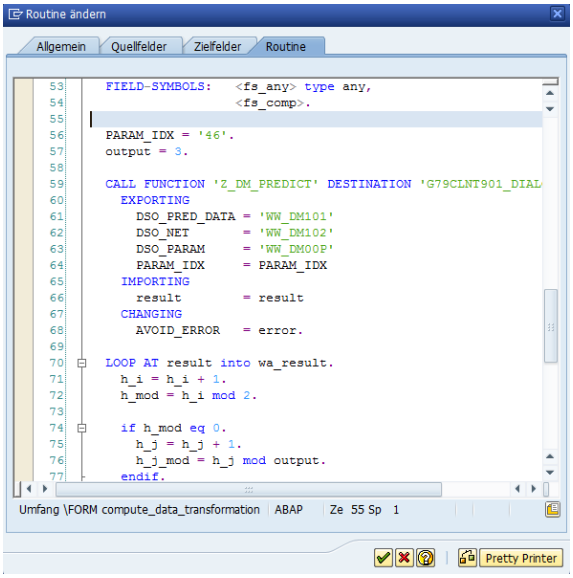

Abbildung 6: Eingebundene Routine in APD

Dieses Konzept der Einbindung erlaubt auch Hochschulen unter den gegebenen Restriktionen des Lizenzabkommens eine Einbindung beliebiger Data-Mining-Routinen; beispielsweise kann eine Data-Mining-Routine nun auch komplett außerhalb von SAP realisiert und mit dem Business-Connector oder Netweaver-Funktionen in das ABAP-System eingebunden und dann per RFC aus dem APD angesprochen werden.

Struktur des Netzwerks und verwendete Algorithmen

Ein neuronales Netz besteht aus einer Menge von Knoten, die über Kanten miteinander verbunden sind. Das Netz besteht aus einer Input-Schicht, einer Reihe verborgener Schichten und einer Output-Schicht, welche mit Kanten, denen Gewichte zugeordnet sind verbunden sind. Es existieren eine Reihe von Aktivierungs- und Ausgabefunktionen in Knoten, die typischerweise die Form einer „Squashing"-Funktion besitzen (Ester& Sander 2009)

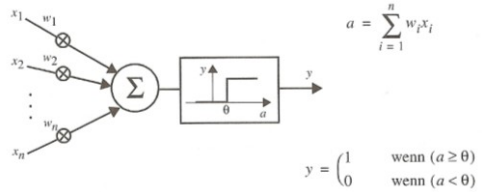

Abbildung 7: Gewichtete Eingangskanten eines Neurons aus (Ester& Sander 2009)

Ein Knoten kann mit Hilfe genau einer Hyperebene die Eingangsdaten separieren - siehe beispielsweise (Ester& Sander 2009) - durch weitere Schichten können weitere Hyperebenen gelernt werden.

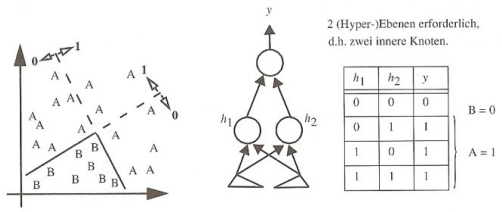

Abbildung 8: Skizze eine Neuronalen Netzwerks sowie der separierbaren Datenklassen - beides aus (Ester& Sander 2009)

Die Klassifikation wird durch die Belegung der Kantengewichte des Netzes gesteuert. Ist die Klassifikation anzupassen, müssen bei einem mehrstufigen Netzwerk eine Reihe von Kanten-gewichten angepasst werden, einen typischen Algorithmus hierfür stellt der „Backpropagation-Algorithmus" dar (Es handelt sich bei diesem Algorithmus i.W. um ein Gradientenabstiegsverfahren). Ein Backpropagation-Algorithmus wurde auch hier implementiert – er wurde dabei um eine Flat-Spot-Elimination und einen Momentum-Term ergänzt. Die Speicherung der Netzstruktur erfolgte in einer DSO – für Details bezüglich der Implementierung sei auf (Augustin 2011) verwiesen.

Das Netz ist – in Anlehnung an die Vergleichsbenchmarks einen Feed-Forward-Netz mit einem verdeckten Layer; es wurden die folgenden Parameter für das neuronales Netz/ den Lernalgorithmus gewählt:

Parameter	Belegung
Lernrate	$0.1 < \alpha < 0.6$
Momentum	$0.2 < \eta < 0.99$
Epochen	$20 < Epochen < 80$
Sigmoid. T.	0.2
FSE	$0.05 < \tau < 0.1$
Input	Anzahl Attribute in D ohne Klassen
Hidden	$5 < hidden < 25$
Output	Anzahl an Klassenattribute in D
Gewicht-Bias	$-0.5 < w_{bias} < 0.5$
Gewicht-Normal	$-0.5 < w_{ij} < 0.5$
Aktivierungsfunktion	$\frac{1}{1+e^{-x/T}}$
Ausgabefunktion	$O_i = id(f_{act}) = f_{act}(net_j)$

Abbildung 9: Parameter des neuronalen Netzes/ des Trainings

Der Grund für diese Parameter liegt in der Vergleichbarkeit mit den Benchmarks (Opitz & Maclin 1999) sowie genereller Vorgaben aus Zell (Zell 2003).

BEWERTUNG DER ERWEITERUNG
Ergebnisse der Literatur

Für die ausgewählten Benchmarks existieren Ergebnisse in der Literatur – speziell die Arbeit von Opitz& Maclin (Optiz&Maclin 1999) stellt für neuronale Netze Ergebnisse für die verwendeten Benchmarkdatensätze vor, in denen auch die Parameter der neuronalen Netze vorgegeben sind.
Für die übrigen drei Benchmarks liegen keine Details bezüglich neuronaler Netze aus der Literatur vor, so dass alleine die Ergebnisse des mehrlagigen neuronalen Netzes aus RapidMiner und den Entscheidungsbäumen der Literatur als Benchmarks verwendet wurden. Schließlich wird das Ergebnis mit dem durch Agustin implementierten (Augustin 2011) für SAP-BI entwickelten neuronalen Netz verglichen. Basis des Vergleiches ist wiederum eine 5 malige Kreuzvalidierung mit Faktor 10. Das Feed-Forward-Netz besaß in den Tests die folgenden Parameter:

Datensatz	Instanzen	Eingabeneuronen	Ausgabeneuronen	verdeckte N.	Epochen
Iris	150	4	3	5	80
House-Votes-84	435	16	1	5	40
Heart cleveland	303	12	1	5	40
Kr-vs-Kp	3196	74	1	15	20
Wine	178	13	3	15*	20*
Census-Income	32561	14	1	15*	20*
Zoo	101	17	7	15*	20*

Abbildung 10: Parameter des Benchmarks –
mit „*" gekennzeichnete Parameter waren nicht in der Literatur dokumentiert

Neuronale Netz auf den Datensätzen
Der Iris-Datensatz stellte sich mit dem gegebenen Parametersatz in RapidMiner als ähnlich gut lösbar durch das neuronale Netz dar, wie im Benchmark; unser („das Fuldaer") neuronale Netz lag dagegen bei vorgegebener verdeckten Neuronen Anzahl der Klassifikationsfehler weit über der Benchmark (> 20%).

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark NN
Iris	4.67% (- 0.37%)	25% (+ 20.7%)	4.3%

Abbildung 11: Vergleich „Fulda NN“ mit Benchmark für IRIS Daten

Die Klassifikationsgenauigkeit konnte nur durch das Einfügen weiterer verdeckten Neuronen erhöht werden – ein Hinzufügen zwei weiterer verdeckter Neuronen erhöht die Genauigkeit in einem Masse, dass das Ergebnis von RapidMiner ungefähr erreicht werden konnte. Im Fulda-NN trat somit ein klares „Underfitting“ aus, welches durch Erhöhen der verdeckten Neuronen behoben werden konnte.

Für das „Congressional Voting“ ergaben sich die Ergebnisse von Abbildung 12 – RapidMiner erreicht die Benchmark gut, die eigene Routine weicht um > 3% ab. In diesem Fall wurde durch die Erhöhung der Anzahl der Neuronen im eigenen Netz nur leichte Verbesserungen, teilweise sogar Verschlechterungen erreicht .

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark NN
House-Votes-84	5.53% (- 0.63%)	8.22% (+ 3.32%)	4.9%

Abbildung 12: Vergleich Rapidminer-NN, mit „Fulda-NN“ mit Benchmark für Voting-Daten

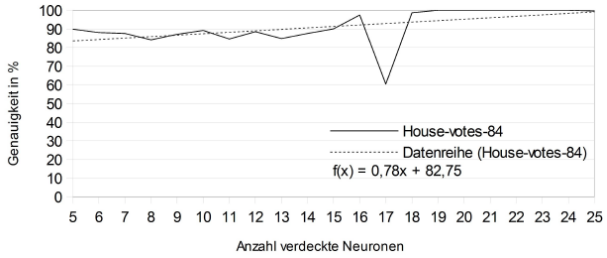

Abbildung 13: Abhängigkeit des Klassifizierungsfehlers von Anzahl der verdeckten Neuronen

Abbildung 14 zeigt die Ergebnisse für die Herz-Daten – es fällt die Nähe von RapidMiner zur Benchmark auf. Für vergleichbare Ergebnisse im eigenen neuronalen Netz war eine Erhöhung der verdeckten Neuronen notwendig. Eine Erhöhung von 5 auf 6 Neuronen erhöhte die Genauigkeit um > 5% (absolut). Wie im Falle des „Iris-Datensatzes“ lag ein „Underfitting“ vor.

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark NN
Heart-Cleveland	18.56% (- 0.04%)	36% (+ 17.4%)	18.6%

Abbildung 14: Vergleich Rapidminer-NN, mit „Fulda-NN“ mit Benchmark für Herz-Daten

Zur Abbildung des Schach-Problems wurde experimentell versucht, die 36+1 diskreten Attribute des Problems in nominelle Werte umzuwandeln. Aufgrund der im neuronalen Netz vorhandenen schlechten semantischen Trennung war die Klassifizierung unbefriedigend. Die bei Optiz und Maclin (Optiz&Maclin 1999) vorgeschlagene binäre Darstellung in 74 Attributen liefert das folgende sehr gute Ergebnis.

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark NN
Kr-vs-Kp	1.97% (- 0.33%)	1.95% (- 0.35%)	0.6%

Abbildung 15: Vergleich Rapidminer-NN, mit Fulda-NN mit Benchmark für Schach-Daten

Für die übrigen Benchmarks wurden keine Vergleichszahlen auf Basis neuronaler Netze gefunden, so dass als Benchmark das Klassifikationsverfahren nach C4.5 herangezogen wird; die Daten hierfür stammen von Rokach (Rokach 2008), es wurden 15 verdeckte Neuronen verwendet. Benchmark zur Bewertung des „Fulda-NN“ ist somit lediglich RapidMiner.

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark C4.5
Wine	3.95% (- 10.09%)	0.68% (- 13.36%)	14.04%
Census-Income	15.61% (+ 0.07%)	0.37 (- 15.17)	15.54%
Zoo	6.91% (+ 0.02%)	8.41% (- 1.48%)	6.93%

Abbildung 16: Vergleich Rapidminer-NN, mit Fulda-NN mit C4.5 Benchmark für übrige Daten

Eine Erhöhung der verdeckten Neuronen für die problematischen Modelle (etwa IRIS) auf 15 führt schließlich zu folgenden Ergebnissen:

Datensatz	Rapidminer NN	Fulda-NN @ SAP-BI	Benchmark NN	Benchmark C 4.5
Iris	4.67% (- 0.37%)	2.67% (- 1.63%)	4.3%	5.2%
House-Votes-84	5.53% (- 0.63%)	6.12% (- 1.22%)	4.9%	3.6%
Heart-Cleveland	18.56% (- 0.04%)	7.62% (- 17.4%)	18.6%	24.3%
Kr-vs-Kp	1.97% (- 0.33%)	1.95% (- 0.35%)	2.3%	2.3%
Datensatz	**Rapidminer NN**	**Fulda-NN @ SAP-BI**	**Benchmark C 4.5**	
Wine	3.95% (- 10.09%)	0.68% (- 13.36%)	14.04%	
Census-Income	15.61% (+ 0.07%)	0.37 (- 15.17)	15.54%	
Zoo	6.91% (+ 0.02%)	8.41% (- 1.48%)	6.93%	

Abbildung 17: Ergebnisse des Benchmarkings bei erhöhter Neuronenanzahl („SAP-NN“ bezeichnet dabei das Fuldaer-NN in SAP-BI)

RapidMiner besitzt – zumindest in der verwendeten Community-Edition – eine deutlich geringere Abhängigkeit von der Anzahl der verdeckten Neuronen, als das eigene Netz.

ZUSAMMENFASSUNG

Die Evaluation zeigt, dass die entscheidungsbaumbasierten Verfahren in SAP und RapidMiner vergleichbare Ergebnisse zum Benchmark liefern.

Es wurde ein Prototyp erstellt, in dem SAP-BI mit einfachen Routinen direkt im APD um neuartige Klassifikationsroutinen erweitert werden kann.

Diese Klassifikationsroutinen wurden mit verschiedenen etablierten Benchmarks evaluiert – aktuell basieren diese Routinen jedoch lediglich auf sehr grundlegenden Standardverfahren der Literatur; es besteht eine weite Bandbreite der Verbesserung dieser Verfahren.

Abhängig vom Datensatz (etwa bei „Wine“ oder „Heart Cleveland“) ergeben sich bereits jetzt deutliche Verbesserungen des Klassifikationsresultats gegenüber den entscheidungsbaumbasierten Verfahren.

Es wurde zudem ein Verfahren implementiert, welches auch Hochschulen erlaubt beliebige Data-Mining-

Routinen in SAP- BI zu implementieren – bis hin zu einer Einbindung externer Routinen.

LITERATUR

Y. Augustin, N. Ketterer: Data-Mining Routinen in SAP BI 7.0, *Management und IT, Tagungsband zur AKWI-Fachtagung 2012*, S.41-54, News & Media 2012

Y. Augustin: Design und Implementierung von Klassifikationsroutinen auf Basis neuronaler Netze in SAP BI 7.0, *Master-Thesis am FB-AI*, HS Fulda 2011

W. W. Cohen: Fast Effective Rule Induction, *Machine Learning 12*, 1995

M. Ester, J. Sander: Knowledge discovery in databases: Techniken und Anwendungen, Springer, 2009

R. Kohavi: A study of cross-validation and bootstrap for accuracy estimation and model selection, *International Joint Conference on Artificial Intelligence 14*: S.1137–1145, 1995

R. Kohavi: Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid., *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996

O. Maimon, L. Rokack: Data Mining and Knowledge Discovery Handbook, 2nd ed. Springer, 2010

D. Opitz, R. Maclin: Popular ensemble methods: an empirical study, *Journal of Artificial Intelligence Research* 11, S.169–198, 1999

J.R. Quinlan: Induction of Decision Trees, *Machine-Learing 1*, S. 81-106, Kluwer 1986

RapidMiner: Deutschsprachiger Internetauftritt unter: http://rapid-i.com/content/view/181/190/lang,de/

L. Rokach: Genetic algorithm-based feature set partitioning for classification problems, *Pattern Recognition*. 41, S.1676–1700, May 2008

SAP AG: Training-Material for Course BW380, 2007

F. Shahnaz: Decision Tree based Algorithms, in Lecture-Notes in Data-Mining von M.W. Berry, M. Browne, World Scientific, 2006

SAP UA EMEA Portal: Preisliste des SAP-UCC von 2012, https://portal.ucc.uni-magdeburg.de/irj/portal/anonymous

UCI Machine Learning Repository: Classification Data-Sets, retrieved 2012, http://archive.ics.uci.edu/ml/datasets.html (Default-Task = "Classification")

Zell: Simulation neuronaler Netze, Oldenbourg, 2003

KONTAKT

Norbert Ketterer lehrt seit Ende 2008 Wirtschaftsinformatik an der Hochschule Fulda. Sein Hauptinteresse liegt in dem Bereich „Betriebliche Anwendungssysteme", insbesondere deren Unterstützung von betrieblichen Geschäftsprozessen.