Maschinelle Erfassung von Problemlösestrategien bei algorithmischen Problemstellungen am Beispiel des Sortierens

Christian Wach

Technische Universität Darmstadt
Didaktik der Informatik
Hochschulstr. 10
64289 Darmstadt
wach@cs.tu-darmstadt.de

Abstract: Die Untersuchung von kognitiven Problemlöseprozessen mit Protokollanalysen ist sehr aufwändig. Mit dem Einsatz von maschinellen Lernverfahren verbindet sich die Hoffnung, aus leicht beobachtbaren Daten auf kognitive Prozesse schließen zu können.
Für Sortierprobleme wurden über 4000 unklassifizierte Nutzerinteraktionen und 490 klassifizierte Nutzerinteraktionen aufgezeichnet und ausgewertet. Verschiedene Klassifikationsverfahren wurden mit den klassifizierten Datensätzen evaluiert. Varianten der linearen Diskriminanzanalyse erreichten nicht nur eine geringe Fehlerrate bei der Klassifikation, sondern können auch zur Dimensionsreduktion der Datensätze genutzt werden.

1 Einleitung

Der Begriff des Problemlösens bzw. der des problemlösenden Denkens subsumiert in der kognitiven Psychologie die kognitiven Prozesse eines Lebewesens zur Lösung von Problemen. Folgende Definition von Dunker [Du35] hält sich in Varianten bis heute (vgl. [Fu03]):

„Ein *Problem* entsteht z. B. dann, wenn ein Lebewesen ein *Ziel* hat und *nicht weiß, wie es dieses Ziel erreichen soll*. Wo immer der gegebene *Zustand* sich nicht durch bloßes Handeln (Ausführen selbstverständlicher Operationen) in den erstrebten Zustand überführen lässt, wird das Denken auf den Plan gerufen. Ihm liegt es ob, ein vermittelndes Handeln allererst zu konzipieren“[Du35] zit. nach [Fu03].

Das Problemraummodell (vgl. [NS72]) formalisiert den Problemlöseprozess. Mögliche Handlungen werden durch Operatoren abgebildet. Ein Operator überführt einen vorliegenden Problemzustand in einen neuen Problemzustand. Eine Lösung enspricht einer Folge von Operatoren, die den Anfangszustand durch Hintereinanderausführung in den Zielzustand überführen.

Nicht alle Probleme sind einander gleich. Daher wurden im Laufe der Zeit verschiedene Klassifikationen und Taxonomien vorgeschlagen (s. [Fu03, Kap. 1.4]). Schüler lösen im Unterricht oft *einfache* Probleme. Bei einfachen Problemen sind die Operatoren bekannt, die Anzahl der jeweils anwendbaren Operatoren ist gering und Anfangs- und Zielzustand sind genau definiert. Charakteristisch für *komplexe* Probleme ist nicht nur eine größere *Komplexität* (im Sinne eines größeren Problemraumes), sondern auch Unklarheit im Bezug auf Problemzustände und Operatoren (vgl. [Fu03]).

In der Informatik erhält der Begriff des Problemlösens eine zusätzliche Dimension. Mit der fundamentalen Idee der Algorithmisierung „verbindet sich die Zielvorstellung (Zielkriterium), alle Probleme ließen sich durch maschinell nachvollziehbare Verfahren, deren Korrektheit jederzeit gesichert ist, effizient lösen“[SS04, Kap. 3.3.2]. Ein Algorithmus ist eine (formalisierte) Handlungsvorschrift zur Lösung eines Problems. Die Erarbeitung bzw. Herleitung einer solchen Handlungsvorschrift zur Lösung eines Problemes ist im Regelfall ein zusätzliches kognitives Problem für den Schüler. Beispielaufgaben für unterschiedliche Jahrgangsstufen finden sich in den Bildungsstandards [Pu07].

Die Untersuchung von kognitiven Prozessen ist recht aufwändig. Mit qualitativen Sprachprotokollanalysen und der Think-Aloud-Methode können Rückschlüsse auf die kognitiven Prozesse gezogen werden (vgl. u.a. [RJJ10]). Mit der Anwendung von maschinellen Lernverfahren ist die Hoffnung verbunden, Rückschlüsse auf kognitive Prozesse aus relativ einfach beobachtbaren Daten, wie den Interaktionen mit Computersystemen, zu ziehen. Kiesmüller et. al. [Ki10a, Ki10b] untersuchten die Interaktionen von Schülern mit der Programmierlernumgebung Kara. Mit einem Hidden Markov Modell konnten die Interaktionen vier definierten allgemeinen Problemlösestrategien zugeordnet werden. McQuiggan et. al. [MML08] nutzten Verfahren des maschinellen Lernens, um aus erhobenen Daten Rückschlüsse auf die Selbstwirksamkeitserwartung einer Person zu ziehen.

1.1 Las Vegas Cardsort

Las Vegas Cardsort [WG09] wurde an der Technischen Universität Darmstadt von Studierenden im Rahmen eines Praktikums implementiert. Ziel der Entwicklung war ein System, das Schülern unterschiedlicher Jahrgangsstufen einen intuitiven und handlungsorientierten Zugang in das Themenfeld Sortieralgorithmen bietet.

Für die vorliegende Arbeit wurde Las Vegas Cardsort in der *Nur Tausch* Variante verwendet. Die Aufgabe für die Schüler bestand darin, 10 Karten mit Zahlen von 1 bis 99 aufsteigend zu sortieren. Für jeden Versuch wurden die Kartenwerte und die Vorsortierung der Karten zufällig bestimmt. Die Karten liegen verdeckt auf dem Spielfeld und müssen aufgedeckt werden. Jedoch kann nur eine Karte gleichzeitig sichtbar sein. Die Karten können nur paarweise vertauscht werden, die in Las Vegas Cardsort möglichen Einfügeoperationen wurden deaktiviert. Bei der Tauschoperation sind bis zu zwei Karten aufgedeckt.

Las Vegas Cardsort protokolliert für jeden Versuch den Anfangszustand und jede Nutzeraktion (Tausch, Ansicht) mit Zeitstempel in ein XML Protokoll.

1.2 Zielsetzung

Für die o.g. Konfiguration von Las Vegas Cardsort konnten bisher über 4000 erfolgreiche Sortierversuche erfasst werden. Auf Basis dieser Daten sollen unter Verwendung von Verfahren des maschinellen Lernens die folgenden Forschungsfragen beantwortet werden:

1. Können Handlungsfolgen von Schülern maschinell einer konkreten Lösungsstrategie zugeordnet werden?
2. Welche Lösungsstrategien können identifiziert werden?
3. Wie entwickelt sich die Lösungsstrategie eines Schülers, wenn dieser mehrfach Sortierungsprobleme löst?

2 Vorverarbeitung der Protokolldaten

Bevor die Protokolldaten analysiert und in maschinellen Lernalgorithmen verarbeitet werden können, müssen diese zunächst in eine geeignete Form gebracht werden. Bei der Vorverarbeitung ist insbesondere darauf zu achten, dass die Datensätze trotz unterschiedlichem Anfangszustand, bedingt durch die unterschiedliche Vorsortierung, vergleichbar bleiben. Um die Vergleichbarkeit zu gewährleisten, wurden die Karten nach ihrer jeweiligen Position im sortierten Zielzustand indiziert. Folglich ist die 0-te Karte die Karte mit dem kleinstem Wert und die 9-te Karte die Karte mit dem größten Wert. Auf Basis dieser Indizierung wurden die beobachteten Variablen in Tabelle 1 ermittelt.

Variable	*Beschreibung*
$t_i, i \in [0,9]$	Zeitpunkt zu dem die i-te Karte zuletzt bewegt wurde, dividiert durch d.
$mc_i, i \in [0,9]$	Anzahl der Vertauschungen der i-ten Karte
$vc_i, i \in [0,9]$	Anzahl der Aufdeckungen der i-ten Karte
$m_i, i \in [0,9]$	Anzahl der Vertauschungen bis zum Zeitpunkt t_i
$v_i, i \in [0,9]$	Anzahl der Aufdeckungen bis zum Zeitpunkt t_i
p	längste Pause zwischen zwei Aktionen
l	Es seien i, j die Indizes zweier zu vertauschender Karten. l entspricht der Summe der Abstände $\|i-j\|$ für alle Tauschoperationen
l_s	Summe der quadrierten Abstände $\|i-j\|^2$ für alle Tauschoperationen
s_m	Gesamtanzahl der Vertauschungen
s_v	Gesamtanzahl der Aufdeckungen
d	Benötigte Zeit zum Sortieren der Folge

Tabelle 1: Beoachtete Variablen nach der Vorverarbeitung

Der erste Schritt der Vorverarbeitung erfolgte in Java. Hierzu wurden anhand der Protokolldateien die Handlungsfolgen nachgespielt und hierbei die Variablen aufgezeichnet. So können die Variablen nicht nur im Nachhinein aus den Logdateien berechnet, sondern auch in Las Vegas Cardsort direkt bestimmt werden. Die Variablen werden als CSV Datei exportiert und können so in anderen Umgebungen weiterverarbeitet werden.

Zur weiteren Verarbeitung der Daten wurde R [R10] verwendet. In R wurden die Variablen *zentriert* und *skaliert*, so dass für jede Variable der Mittelwert 0 und die Standardabweichung 1 ist.

3 Analyse der unklassifizierten Daten

Die 58 beobachteten Variablen sind stark korreliert. In Abb. 1 ist eine graphische Repräsentation der Korrelationsmatrix, ein *Corrgram* [Fr02], abgebildet. Die Reihenfolge der Variablen in der Korrelationsmatrix entspricht der Reihenfolge in Tabelle 1. Die Schattierung der Felder repräsentiert den Pearson Korrelationskoeffizienten der beiden Variablen (Schwarz = 1, Weiß = 0).

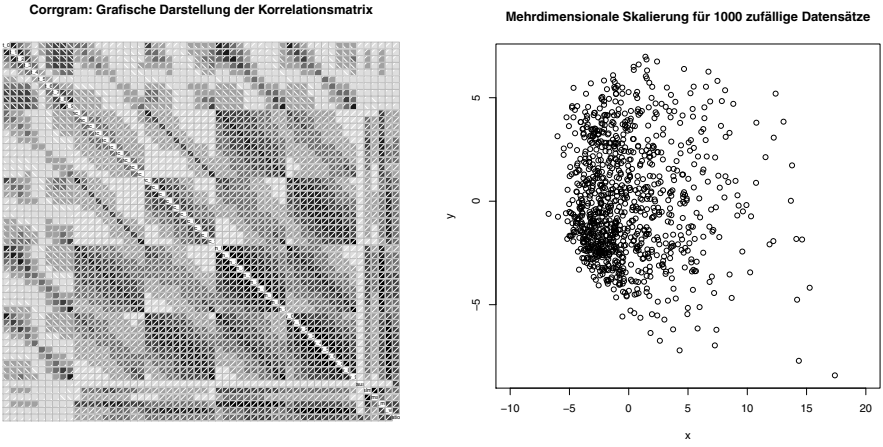

Abbildung 1: Links: Corrgram; Rechts: Multidimensionale Skalierung

Im Plot der multidimensionalen Skalierung für 1000 zufällig gewählte Datensätze ist keine Clusterstruktur erkennbar. Der *k-means* Algorithmus bestimmte für $k \in [2, 30]$ nur künstliche Cluster mit einer durchschnittlichen Silhouttenbreite (*average silhouette width* [Ro87]) kleiner 0,2. Auch ein dichtebasiertes Clustering mit DBSCAN [Es96] konnte keine natürliche Clusterstruktur in den Daten finden.

4 Erfassung von klassifizierten Daten

Zur weiteren Analyse wurden durch Studierende der TU Darmstadt Trainingsdatensätze erfasst. Hierzu sortierten die Studierenden unter den gleichen Bedingungen nach einem vorgegebenen Sortieralgorithmus. Insgesamt wurden 490 Trainingsdatensätze für die folgenden Sortieralgorithmen erfasst:

1. Selection sort von links nach rechts (Suche die Karte mit kleinstem Wert und tausche diese an den linken Rand des unsortierten Bereichs),
2. Selection sort von rechts nach links (Suche die Karte mit größtem Wert und tausche diese an den rechten Rand des unsortierten Bereichs),
3. Bubble sort von links nach rechts (Gehe das Feld von rechts nach links durch und tausche immer die kleinere Karte nach links),
4. Bubble sort von rechts nach links (Gehe das Feld von links nach rechts durch und tausche immer die größere Karte nach rechts),
5. Gnomesort von links nach rechts (Kann sowohl als Insertionsort, als auch als bidirektionaler Bubble sort betrachtet werden).

Die Trainingsdaten wurden zusammen mit den 4000 unklassifizierten Datensätzen auf Mittelwert 0 zentriert und auf Standardabweichung 1 skaliert.

Die Trainingsdaten sind innerhalb der Klassen nicht multivariat normal verteilt und haben unterschiedliche Dispersion.

5 Ergebnisse

Mit den klassifizierten Datensätzen wurden verschiedene Klassifikationsverfahren des maschinellen Lernens evaluiert. Hierzu wurden die Datensätze zunächst in eine Trainingsmenge (2/3) und eine Testmenge (1/3) aufgeteilt. Die jeweiligen Klassifikationsverfahren wurden ausschließlich mit den Daten der Trainingsmenge trainiert. Das so entstehende Vorhersagemodell wurde mit Hilfe der Testmenge auf Genauigkeit überprüft. Die Fehlerrate gibt für Trainings- und Testdaten den Anteil von falsch klassifizierten Datensätzen an.

Die Fehlerraten für die jeweiligen Verfahren wiesen in Abhängigkeit von der Verteilung der Datensätze auf beiden Mengen eine so starke Streuung auf, dass ein genauer Vergleich der Verfahren nicht möglich war. Um stabilere Fehlerraten zu erhalten wurden die Verfahren anschließend mit *10-facher stratifizierter Kreuzvalidierung* getestet. Hierzu wurden die Datensätze in 10 Mengen partitioniert, wobei der Anteil der Datensätze einer Klasse in einer Menge dem Anteil in der Gesamtmenge entspricht. Für jede dieser Mengen wurde jedes Verfahren mit den übrigen Mengen trainiert und mit den Daten dieser Menge evaluiert. Als Gesamtfehlerrate wurde das arithmetische Mittel über die 10 Einzelfehlerraten

Verfahren	*Fehlerrate Training*	*Fehlerrate Test*
Lineare Diskriminanzanalyse [VR02]	0,029	0,049
J48 Entscheidungsbaum [HBZ09, WF05] (Implementierung von Quinlans C4.5 Algorithmus[Qu93])	0,013	0,103
5 Nächste-Nachbarn-Klassifikation (knn) [We05]	0,057	0,098
Random Forest [LW02]	0,000	0,054
Multinomiale logistische Regression (multinom, nnet package [VR02])	0,000	0,141
L2 - Regularisierte multinomiale logistische Regression (LIBLINEAR) [Fan08, He10]	0,008	0,054
Support Vector Machine mit linearem Kernel[Di10]	0,022	0,060
Naiver Bayes-Klassifikator unter der Annahme multinomialer Normalverteilung [We05]	0,118	0,134
Naiver Bayes-Klassifikator mit Kerndichteschätzer	0,057	0,086
sparseLDA mit max. 35 Koeffizienten pro Diskriminante [CHE08, Cl08]	0,037	0,050
Regularized Discriminant Analysis [Fr89, We05]	0,000	0,023

Tabelle 2: Vergleich der Fehlerraten für verschiedene Klassifikationsverfahren 10-fold cv

gebildet. Die Fehlerraten für eine Auswahl der getesteten Klassifikationsverfahren sind in Tabelle 2 abgebildet.

Auffällig ist das gute Ergebnis der Linearen Diskriminanzanalyse (LDA) [VR02], obwohl die Vorraussetzungen, eine multivariate Normalverteilung mit gleicher Dispersion in den jeweiligen Klassen, nicht gegeben ist. Eine Quadratische Diskriminanzanalyse (QDA) [VR02] konnte wegen der Anzahl der Variablen nicht angewendet werden. Friedmans Regularized Discriminant Analysis (RDA) [Fr89, We05] erzielte die besten Ergebnisse.

Multinomiale logistische Regression erzeugte eine deutliche Überanpassung des Modells an die Trainingsdaten. Diese Überanpassung konnte mit einer L_2-Norm Regularisierung [Fan08, He10] vermieden werden. Ähnliche Überanpassungen konnten für andere Verfahren beobachtet werden, die komplexere, nicht-lineare Entscheidungsgrenzen bestimmen. Verursacht wurden diese Überanpassungen durch die vergleichsweise große Anzahl der Variablen.

Sehr deutlich wird dieses Problem der hohen Dimensionalität im direkten Vergleich von LDA und QDA. Es sei $\mathbf{X} \in \mathbb{R}^{n \times p}$ die Matrix aller Datensätze mit n Datensätzen und p beobachteten Variablen. Für jede der k-Klassen $C_1, \ldots, C_k$ sei n_i die Anzahl der Datensätze in dieser Klasse und $\mathbf{m}_i$ der Schwerpunkt (arithmetisches Mittel) der Klasse.

Die LDA ordnet einen neuen Datensatz $\mathbf{x}'$ derjenigen Klasse zu, deren Schwerpunkt $\mathbf{m}_i$ den geringsten Abstand zu $\mathbf{x}'$ hat. Der Abstand wird in der Mahalanobisdistanz

$$D_i(\mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{m_i})^t \mathbf{S_W}^{-1} (\mathbf{x} - \mathbf{m_i})}$$

gemessen. Wobei $\mathbf{S_W}$ die klassenübergreifende Kovarianzmatrix bezeichnet (vgl. [HTB94]).

$$\mathbf{S_W} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{m_i})(\mathbf{x} - \mathbf{m_i})^t$$

Die QDA unterscheidet sich von der LDA nur in der Distanzmessung. Statt einer klassenübergreifenden Kovarianzmatrix wird für jede Klasse eine klassenspezifische Kovarianzmatrix verwendet. Im vorliegenden Fall würde dann die 56×56 Kovarianzmatrix aus nur noch ca. 40 Datensätzen geschätzt werden. Es ist offensichtlich, dass eine so geschätzte Kovarianzmatrix keine zufriedenstellenden statistischen Eigenschaften besitzt. Probleme aufgrund der Dimensionalität können mit den folgenden Verfahren vermindert werden:

1. Vorherige Variablenauswahl (z. B. nach der Korrelation mit der Klassenzugehörigkeit),
2. Bestimmen von latenten Variablen mit Hilfe von Hauptkomponentenanalyse (Principal component analysis) [Ho33, Jo02],
3. Wahl von Klassifikationsverfahren, die die Variablen während der Trainingsphase auswählen (z. B. Entscheidungsbäume [Qu93]),
4. Wahl von regularisierten Klassifikationsverfahren. Erwähnenswert sind insbesondere die aus der Regression bekannten L_1[Ti96] und L_2 Norm Regularisierungen [ZH05], die in den letzten Jahren für verschiedene Klassifikationsverfahren genutzt wurden (s. u.a. [Yu10]).

Im vorliegenden Fall, haben sich insbesondere die regularisierten Verfahren bewährt, die auch bei wenigen Trainingsdatensätzen akkurat (Fehlerrate 5 – 7%) klassifizierten. Erwähnenswert ist, dass eine Regularisierung mit L_1 Norm einige Koeffizienten auf 0 schrumpft, während Regularisierung mit L_2 Norm die Koeffizienten des Modells zwar schrumpft, diese aber nicht 0 werden. Stattdessen schrumpft die Regularisierung mit L_2 Norm Koeffizienten korrellierter Variablen auf ähnliche Werte (vgl. [ZH05])

Friedmans RDA [Fr89] ist eine spezielle Form der Regularisierung, die einen Kompromiss zwischen QDA und LDA darstellt. Die Kovarianzmatrizen für die jeweiligen Klassen werden durch die klassenübergreifende Kovarianzmatrix und ein Vielfaches der Einheitsmatrix regularisiert. Die geringere Fehlerrate der RDA gegenüber der LDA erklärt sich aus der beobachteten unterschiedlichen Dispersion der Daten in den Klassen.

5.1 Dimensionsreduktion mit Fishers linearer Diskriminante

Eine alternative Formulierung der LDA geht auf Fisher [Fi36] zurück. Gesucht ist eine lineare Projektion $\mathbf{W}^t$, die die Datensätze in einen $k-1$-dimensionalen Raum (k ist die Anzahl der Klassen) projeziert, so dass die Schwerpunkte der Klassen möglichst weit auseinanderliegen und die klassenübergreifende Kovarianzmatrix im projezierten Raum der Einheitsmatrix entspricht. Diese Projektion kann durch Eigenwertzerlegung recht einfach bestimmt werden (vgl. [VR02]). Die Spalten von $\mathbf{W}$ nennt man Diskriminanten (discriminant directions).

Der durch diese Projektion entstehende $k-1$-dimensionale Raum enthält alle nötigen Abstandsinformationen der Datensätze. Der euklidische Abstand im Bild von $\mathbf{W^t}$ entspricht der Mahalanobisdistanz der einzelnen Datensätze. Die Dimension kann noch weiter reduziert werden, in dem nur die ersten $k' < k-1$ Diskriminanten verwendet werden. Dieses Verfahren ist als *Reduced rank LDA* bekannt und kann zu einer höheren Genauigkeit führen (vgl. [HTB94]).

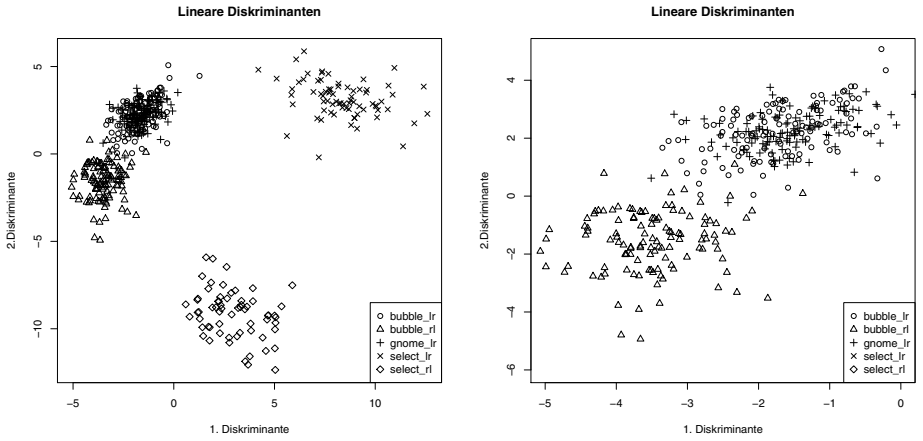

Abbildung 2: Projektion der klassifizierten Daten auf die ersten beiden Diskriminanten

Die so entstehende Projektion kann nicht nur für weitere Analysen der Daten verwendet, sondern auch für eine geeignete grafische Darstellung der Datensätze benutzt werden. So enthalten die ersten zwei Diskriminanten für die klassifizierten Daten über 90 % der zur Klassifikation nötigen Informationen. Die Projektion der Daten auf die ersten zwei Diskriminanten sind in Abb. 2 abgebildet. Die beiden Selectionsort Varianten werden optimal getrennt (s. Abb 2 links). Die Überlagerungen zwischen Gnomesort und Bubblesort (s. Abb. 2 rechts) sind durch die große Ähnlichkeit der Verfahren zu erklären. Mit den übrigen zwei Diskriminanten gelingt die Trennung besser, allerdings nicht perfekt. Die Fehlerrate (s. Tab. 2) der LDA und anderer Verfahren erklärt sich durch diese Überdeckungen.

6 Zusammenfassung und Ausblick

Handlungsfolgen beim Sortieren von Karten können mit Hilfe von verschiedenen Verfahren des maschinellen Lernens mit hoher Genauigkeit (>90 %) konkreten Sortierstrategien zugeordnet werden. Die Lineare Diskriminanzanalyse (LDA) erreichte eine gute Testfehlerrate von 5 %, obwohl die Vorraussetzung einer möglichst gleichen Dispersion in den Klassen nicht gegeben war. Die geringste Fehlerrate erreichte Friedmans Regularized Discriminant Analysis [Fr89] eine regularisierte Erweiterung der LDA.

LDA kann nicht nur zur Klassifizierung, sondern auch zur Dimensionsreduktion genutzt werden. Die Projektion in den niederdimensionalen Raum kann nicht nur zur grafischen Darstellung, sondern auch zur weiteren Analyse und Verarbeitung der Daten verwendet werden. Diese Projektion ist ein Kompromiss zwischen Bias und Varianz. Im projezierten Raum sind die unterschiedlichen Klassen durch lineare Entscheidungsgrenzen zu 95 % trennbar.

Gegenstand weiterer Untersuchungen sind die erfassten unklassifizierten Datensätze. Erste Analysen mit Clusteringverfahren konnten keine natürliche Clusterstrukur identifizieren. Mögliche Ursachen hierfür sind verrauschte Daten, überlappende Klassen und ein degenerierendes Distanzmaß in höheren Dimensionen.

In weiteren Untersuchungen sollen mit Verfahren der Novelty Detection bisher nicht klassifizierte Sortierstrategien entdeckt werden. Der Klassifikator soll im Anschluß, um diese neuen Sortierstrategien erweitert werden. Mit diesem Klassifikator können anschließend die Fragestellungen zur Veränderung der Strategien eines Schülers untersucht werden.

Literaturverzeichnis

[CHE08] Line Clemmensen, Trevor Hastie und Bjarne Ersbøll. Sparse discriminant analysis. Bericht, Technical University of Denmark, Lyngby, June 2008.

[Cl08] Line Clemmensen. Sparse discriminant analysis (sparseLDA) software in R, may 2008.

[Di10] Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, und Andreas Weingessel. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2010. R package version 1.5-24.

[Du35] Konrad Duncker. *Zur Psychologie des produktiven Denkens*. Julius Springer, 1935.

[Es96] Martin Ester, Hans P. Kriegel, Jorg Sander und Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Evangelos Simoudis, Jiawei Han und Usama Fayyad, Hrsg., *Second International Conference on Knowledge Discovery and Data Mining*, Seiten 226–231, Portland, Oregon, 1996. AAAI Press.

[Fan08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang und Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874, August 2008.

[Fi36] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugen.*, 7:179–188, 1936.

[Fr89] Jerome H. Friedman. Regularized Discriminant Analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.

[Fr02] Michael Friendly. Corrgrams: Exploratory Displays for Correlation Matrices. *The American Statistician*, 56(4):316–324, 2002.

[Fu03] Joachim Funke. *Problemlösendes Denken*. Kohlhammer, Stuttgart, 2003.

[HBZ09] Kurt Hornik, Christian Buchta und Achim Zeileis. Open-Source Machine Learning: R Meets Weka. *Computational Statistics*, 24(2):225–232, 2009.

[He10] Thibault Helleputte. *LiblineaR: Linear Predictive Models Based On The Liblinear C/C++ Library*, 2010. R package version 1.51-3.

[Ho33] Harold Hotelling. Analysis of complex statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, September 1933.

[HTB94] Trevor Hastie, Robert Tibshirani und Andreas Buja. Flexible Discriminant Analysis by Optimal Scoring. *Journal of the American Statistical Association*, 89:1255–1270, December 1994.

[Jo02] Ian T. Jolliffe. *Principal Component Analysis*. Springer, 2nd. Auflage, October 2002.

[Ki10a] Ulrich Kiesmüller. Automatisierte, prozessbegleitende Identifizierung der Problemlösestrategien Lernender unter Verwendung von Mustererkennungsmethoden. In *Didaktik der Informatik - Möglichkeiten empirischer Forschungsmethoden und Perspektiven der Fachdidaktik.*, Seiten 93–104, 2010.

[Ki10b] Ulrich Kiesmueller, Sebastian Sossalla, Torsten Brinda und Korbinian Riedhammer. Online identification of learner problem solving strategies using pattern recognition methods. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, ITiCSE '10, Seiten 274–278, New York, NY, USA, 2010. ACM.

[LW02] Andy Liaw und Matthew Wiener. Classification and Regression by randomForest. *R News*, 2(3):18–22, 2002.

[MML08] Scott McQuiggan, Bradford Mott und James Lester. Modeling self-efficacy in intelligent tutoring systems: An inductive approach. *User Modeling and User-Adapted Interaction*, 18(1):81–123, February 2008.

[NS72] Allen Newell und Herbert A. Simon. *Human Problem Solving*. Prentice-Hall, Inc., 1972.

[Pu07] Hermann Puhlmann et al. Grundsätze und Standards für die Informatik in der Schule. *LOG IN*, 146/147, 2007.

[Qu93] J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[R10] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.

[RJJ10] V. G. Renumol, Dharanipragada Janakiram und S. Jayaprakash. Identification of Cognitive Processes of Effective and Ineffective Students During Computer Programming. *Trans. Comput. Educ.*, 10, August 2010.

[Ro87] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, November 1987.

[SS04] Sigrid Schubert und Andreas Schwill. *Didaktik der Informatik*. Spektrum Akademischer Verlag, 1.. Auflage, 2004.

[Ti96] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[VR02] W. N. Venables und B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth. Auflage, 2002. ISBN 0-387-95457-0.

[WF05] Ian H. Witten und Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd. Auflage, 2005.

[WG09] Christian Wach und Jens Gallenbacher. Spielend Sortieren mit Las Vegas Cardsort. In Bernhard Koerber, Hrsg., *INFOS*, Jgg. 156 of *LNI*, Seiten 256–267. GI, 2009.

[We05] Claus Weihs, Uwe Ligges, Karsten Luebke und Nils Raabe. klaR Analyzing German Business Cycles. In L. Schmidt-Thieme, Hrsg., *Data Analysis and Decision Support*, Seiten 335–343, Berlin, 2005. Springer-Verlag.

[Yu10] Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh und Chih-Jen Lin. A Comparison of Optimization Methods and Software for Large-scale L1-regularized Linear Classification. *Journal of Machine Learning Research*, Seiten 3183–3234, 2010.

[ZH05] Hui Zou und Trevor Hastie. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.