Adaption und Vergleich evolutionärer mehrkriterieller Algorithmen mit Hilfe von Variablenwichtigkeitsmaßen

- Am Beispiel der kostensensitiven Klassifikation von Lungenkrebssubtypen -

Dissertation
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften

Der Fakultät Statistik
der Technischen Universität Dortmund
vorgelegt von

Swaantje Wiarda Casjens
aus Oldenburg

Dortmund, 2013

Tag der mündlichen Prüfung: 09. Juli 2013
Gutachter: Prof. Dr. Katja Ickstadt
JProf. Dr. Uwe Ligges

Danksagung

Mein Dank für die hilfreiche Unterstützung bei der Erstellung meiner Doktorarbeit geht vor allem an meine Doktormutter Frau Prof. Dr. Katja Ickstadt. Ihr kompetenter Rat und ihre Hilfe kam mir in zahlreichen Angelegenheiten sehr zu Gute.

Herrn JProf. Dr. Uwe Ligges danke ich insbesondere für seine Unterstützung bei der Erstellung des R-Pakets und Herrn Olaf Mersmann für das R-Paket `emoa`, das er mir vorzeitig zur Verfügung stellte. Dies ersparte mir viel Programmierarbeit.

Ferner bedanke ich mich bei Prof. Dr. Holger Schwender für seine anregenden Diskussionen und hilfreiche Unterstützung vor allem beim Aufbau der Simulationsstudie.

Anita Thieler danke ich ebenfalls für die zahlreichen Diskussionen, das akribische Korrektur lesen, sowie für den Beistand während des gesamten Studiums. An dieser Stelle seien ebenfalls Frau Dr. Esther Herberich und Frau Dr. Verena Hoffmann genannt.

Ein besonderer Dank gilt Frau PD Dr. Beate Pesch, die den Anstoß zur Themenfindung in Richtung der mehrkriteriellen Optimierung gab.

Für die zur Verfügung gestellten Daten bin ich dem Bundesamt für Strahlenschutz, dem Institut für Prävention und Arbeitsmedizin der DGUV (IPA) und dem Institut für Pathologie in Bochum dankbar.

Der größte Dank gilt jedoch Hendrik Blom, der mich auf die Fährten der Evolutionären Algorithmen brachte, mich stets unterstützte und ermunterte, weiter zu machen.

Inhaltsverzeichnis

1 Einleitung

In der Biostatistik sowie in verwandten Wissenschaften ist die Vorhersage kategorieller Zielvariablen, wie etwa der Krankheitsstatus eines Patienten oder dessen Therapiechancen, ein wesentliches Ziel. Neben der guten Vorhersage der Zielvariablen ist sowohl die Identifikation und Selektion verlässlicher Einflussvariablen als auch eine gute Interpretierbarkeit der Ergebnisse mit entsprechendem Erkenntnisgewinn wichtig. Ferner ist bei vorliegenden Einflussvariablen mit unterschiedlichen Kosten eine kostensensitive Klassifikation erstrebenswert, bei der ein Kompromiss aus hoher Vorhersagegüte und geringen Kosten getroffen werden muss.

Nach der frühen Arbeit von Morgan und Sonquist (1963) zur Ermittlung von Interaktionen in Umfragedaten durch baumähnliche Klassifikations- und Regressionsregeln enstanden diverse Entscheidungsbaumalgorithmen. Ihr nicht-parametrischer robuster Ansatz und die einfache Interpretierbarkeit der Ergebnisse, die dem menschlichen Denken ähnlich und somit leicht verständlich sind (vgl. Hastie et al., 2009), trug stark zur Popularität der Entscheidungsbäume für die Lösung von Klassifikationsproblemen bei. Vor allem in der Medizin finden Entscheidungsbäume und auf Entscheidungsbäumen basierende Ensemblemethoden großen Zuspruch (vgl. etwa Righi et al., 2012; Vendrame et al., 2012; Berney et al., 2010; Kim et al., 2010; Das und Sengur, 2010), sodass binäre Entscheidungsbäume (vgl. Kapitel 2) in dieser Arbeit als grundlegende Klassifikationsmethode betrachtet werden.

1.1 Problembeschreibung

Die Herleitung eines Klassifikationsmodells h besteht aus zwei Teilaufgaben: der Auswahl der bestmöglichen Parameter und der Auswahl der besten Variablenteilmenge für das zu lösende Problem. Die Qualität des Klassifikationsmodells wird üblicherweise anhand eines Maßes $\gamma(h)$, wie etwa der Fehlklassifikationsrate, bewertet. Sobald dieses Gütemaß spezifiziert ist, wird die Aufgabe zum Auffinden eines Klassifikationsmodells zu einem Optimierungsproblem im Raum der potentiellen Klassifikationsmodelle $\mathcal{H}$: $\arg \min_{h \in \mathcal{H}} \tilde{\gamma}(h)$ mit $\tilde{\gamma}(h) = \gamma(h)$, wenn $\gamma(h)$ minimiert, und $\tilde{\gamma}(h) = -\gamma(h)$, wenn $\gamma(h)$ maximiert werden soll. Das Optimierungsproblem wird besonders schwierig, wenn die Anzahl möglicher Einflussvariablen p und die daraus resultierende Anzahl der Variablenkombinationen 2^p hoch ist. Die vollständige Enumeration, d.h. die Analyse aller möglichen Variablenteilmengen, ist dann praktisch nicht möglich (vgl. Jain et al., 2000), sodass eine Selektion potentiell geeigneter Teilmengen mit den gewünschten Eigenschaften notwendig wird.

Falls eine kostensensitive Klassifikation erfolgen soll oder neben der Vorhersagegüte noch weitere Zielfunktionen in Form von Gütemaßen optimiert werden sollen, entsteht ein mehrkriterielles Optimierungsproblem (vgl. Definition 4 in Kapitel 5). Im Allgemeinen sind die verschiedenen Ziele in der mehrkriteriellen Optimierung konfliktär, sodass nicht eine einzige sondern eine Menge unvergleichbarer Lösungen existieren.

Die unvergleichbaren Lösungen sind Pareto-optimal, wenn keine weitere Lösung im Suchraum existiert, die bei Betrachtung aller Optimierungsziele diesen überlegen ist. Sie liefern einen besseren Einblick in das Optimierungsproblem und erlauben dem Anwender a posteriori die Auswahl aus der Menge der verschiedenen unvergleichbaren Lösungen. Diese Auswahlmöglichkeit ist intuitiv effektiver als a priori eine Gewichtung der konfliktären Ziele vorzunehmen und das mehrkriterielle in ein einkriterielles Optimierungsproblem zu überführen.

In den letzten Jahren wuchs das Interesse an der mehrkriteriellen Variablenselektion und Klassifikation stetig (vgl. Hamdani et al., 2007; Reynolds und de la Iglesia, 2007; Badran und Rockett, 2008; Garcia-Nieto et al., 2009; Coello Coello, 2009). Die Gründe hierfür liegen in der Tatsache, dass nicht nur die Vorhersagegüte ausschlaggebend für die Wahl eines überwachten Klassifikationsmodells ist, sondern auch eine gewisse Verringerung in der Vorhersage tragbar ist, wenn dies zu einem weniger komplexen Modell oder einer kostengünstigeren Vorhersage führt. Für viele Anwendungen ist der Kompromiss zwischen Vorhersagegüte und Vorhersagekosten a priori schwer bestimmbar und somit auch die Kombination mehrerer Optimierungsprobleme in einem Problem durch eine Gewichtungsfunktion schwierig. Wird ein mehrkriterielles Optimierungsproblem in ein skalares Optimierungsproblem überführt, existieren einige Nachteile, wie das Scheitern bei der Aufdeckung nicht-konvexer Teile der Pareto-Front oder die fehlende Diversität der Lösungen auf der Pareto-Front (vgl. Das und Dennis, 1997).

1.2 Lösungsansätze

Obwohl einige Methoden für die gleichzeitige Optimierung verschiedener Zielfunktionen existieren, ist es schwer, mittels eines deterministischen Verfahrens eine Menge Pareto-optimaler Lösungen in komplexen Problemen zu finden (vgl. Wegener, 2003). Bei unbekannten zu optimierenden Funktionen können keine Gradientenverfahren (vgl. Fletcher, 1987) zum Einsatz kommen, sodass in diesem Fall direkte Suchverfahren, wie etwa evolutionäre mehrkriterielle Optimierungsalgorithmen (EMOAs), unabdingbar sind.

Häufig werden EMOAs, die auf dem Prinzip der biologischen Evolution basieren, zur Approximation von Pareto-optimalen Lösungen herangezogen. In vielen Anwendungsbereichen führte ihr populationsbasierter Ansatz zu guten Lösungen mit einer hohen Konvergenz und Diversität (vgl. etwa Garcia-Nieto et al., 2009; Castillo Tapia und Coello Coello, 2007; Alba et al., 2007; Oliveira et al., 2003). EMOAs eignen sich für die Lösung mehrkriterieller Optimierungsprobleme, da sie nach verschiedenen Lösungen, den sogenannten Individuen, parallel suchen können, unabhängig von der zugrundeliegenden Datenverteilung sind, globale stochastische Sucheigenschaften besitzen und direkt Klassifikationsmodelle entwickeln können (vgl. De Jong, 2006; Emmanouilidis et al., 2000). Ferner können EMOAs nicht-konvexe und nichtstetige Funktionen optimieren (vgl. Kshetrapalapuram und Kirley, 2005). Ein guter Überblick über EMOAs wurde etwa von Coello Coello et al. (2007) publiziert.

EMOAs werden für die Lösung von Klassifikationsproblemen klassischerweise in Form von mehrkriteriellen *Wrapper*-Ansätzen verwendet. Der *Wrapper* (vgl. Kohavi und John, 1997) umhüllt dabei einen Klassifikationsalgorithmus und übersetzt sowohl die EMOA-Individuen in das Eingabeformat des Klassifikationsalgorithmus als auch das durch den Klassifikationsalgorithmus berechnete Gütemaß in den Fitnesswert der EMOA-Individuen, durch den die einzelnen Individuen untereinander vergleichbar werden. Im Falle der Variablenselektion werden die Individuen eines EMOAs als binäre Zeichenketten (Bitstrings) codiert, bei der die Binärzeichen (Bits) die Verfügbarkeit der entsprechenden Einflussvariable beschreiben, sodass jedes Individuum in der Population eine Variablenteilmenge darstellt. Basierend auf diesen Teilmengen und gegebenen Daten erstellt der umhüllte Klassifikationsalgorithmus ein Klassifikationsmodell, das durch ein Gütemaß bewertet werden kann. Die Optimierung dieser Güte stellt ein externes Zielkriterium dar. Das interne Optimierungskriterium des Klassifikationsalgorithmus, wie die Gini-Wichtigkeit in *Classification and Regression Trees* (CART) (vgl. Breiman et al., 1998) oder die *log-Likelihood* im logistischen Regressionsmodell (vgl. Hastie et al., 2009), zielt nur auf ein externes Zielkriterium, nämlich die Vorhersagegüte des Modells ab. Erst nach der Konstruktion des Klassifikationsmodells können weitere externe Zielkriterien, wie etwa die Kosten der selektierten Variablen, ausgewertet werden. Damit entsteht eine eindeutige Hierarchie der externen, zu optimierenden Zielkriterien mit Vorteil für die Vorhersagegüte. Offensichtlich werden in einem mehrkriteriellen Optimierungsproblem durch einen mehrkriteriellen *Wrapper*-Ansatz somit keine nicht-hierarchischen Lösungen gefunden. Diese Hierarchie der Zielfunktionen wird erstmals in Rahmen dieser Arbeit beschrieben und untersucht (vgl. Abschnitt 6.1).

Als Alternative zum mehrkriteriellen *Wrapper*-Ansatz wird hier ein nicht-hierarchischer evolutionärer mehrkriterieller Optimierungsalgorithmus mit Baum-Reprä-

sentation (NHEMOtree) entwickelt, um mehrkriterielle Optimierungsprobleme mit gleichberechtigten Optimierungszielen zu lösen. Dieser neue Algorithmus basiert auf einem EMOA mit Baum-Repräsentation und behandelt die Modellierung des Klassifikationsproblems und der Kostenreduktion in einem einzigen Optimierungsproblem. Banzhaf et al. (1998) und Loveard und Ciesielski (2001) zeigten bereits, dass mittels EMOAs mit Baum-Repräsentation gute Entscheidungsbäume in einer angemessenen Zeit generiert werden können. In NHEMOtree erfolgt die Variablenselektion und die Erstellung der binären Entscheidungsbäume ausschließlich auf Basis des EMOAs, sodass kein interner Klassifikationsalgorithmus benötigt wird (vgl. Jabeen und Baig, 2010b). Somit kann NHEMOtree im Gegensatz zum mehrkriteriellen *Wrapper*-Ansatz eigenständig und ohne Hierarchie in den Zielfunktionen basierend auf den ausgewählten Variablen mehrkriteriell optimierte binäre Entscheidungsbäume erstellen (vgl. Abschnitt 6.2).

Im Rahmen dieser Arbeit erfolgt erstmalig ein Vergleich der mehrkriteriellen Optimierung durch einen mehrkriteriellen *Wrapper*-Ansatz bestehend aus einem der EMOAs NSGA-II, *Steady-State* NSGA-II oder SMS-EMOA (vgl. Srinivas und Deb, 1994; Nebro und Durillo, 2009; Emmerich et al., 2005) mit CART als binärem Entscheidungsbaumalgorithmus und durch einen EMOA mit Baum-Repräsentation (NHEMOtree). Die Algorithmen werden dabei anhand der gefundenen Lösungen miteinander verglichen. Die Bewertung der Lösungen erfolgt dabei zum einen mittels der bekannten S-Metrik (vgl. Zitzler und Thiele, 1998) und zum anderen mit dem hier entwickelten Dominanzquotienten γ^D (vgl. Abschnitt 5.2). γ^D setzt die nicht-dominierten Individuen der gemeinsamen Pareto-Front beider Lösungsmengen ins Verhältnis, so dass ein Quotient größer Eins einen Vorteil für die Lösungemenge im Zähler und ein Quotient kleiner Eins einen Vorteil for die Lösungesmenge im Nenner beschreibt. Im Vergleich zur S-Metrik honoriert γ^D die Anzahl der nicht-dominierten Lösungen stärker als die S-Metrik. Dies ist vorteilhaft, da der Anwender aus einer größeren Menge Pareto-optimaler Lösungen auswählen kann.

Des Weiteren wird die Variablenwichtigkeit bekannter Ensemblemethoden (vgl. Breiman, 2001; Strobl et al., 2007b) für mehrkriterielle Optimierungsprobleme angepasst (vgl. Abschnitt 6.3). Diese neuen mehrkriteriellen Variablenwichtigkeitsmaße (engl. *variable importance measures*, VIMs) dienen dann der verbesserten Selektion von Baumknoten für den in dieser Arbeit entwickelten VIM-basierten Rekombinationsoperator X_{VIM} in NHEMOtree (vgl. Abschnitt 6.4).

1.3 Anwendungsbeispiel und Simulationsstudie

Der Vergleich der beiden mehrkriteriellen Optimierungsansätze erfolgt auf einem medizinischen Datensatz von 143 Lungengewebeproben mit und ohne Krebs. Lungenkrebs stellt die häufigste Ursache für krebsbezogene Todesfälle in Deutschland und den USA dar (vgl. RKI, 2012; Jemal et al., 2009). Die Histopathologie der drei Lungenkrebssubtypen (Kleinzelliger Lungenkrebs, Adenokarzinom, Plattenepithelkarzinom) ist nicht immer offensichtlich, deren korrekte Klassifikation für die Behandlung aber entscheidend. Die Heterogenität der Lungenkrebssubtypen motivierte bereits Garber et al. (2001), Lungenkrebs anhand seines molekularen Profils zu klassifizieren, wodurch eine frühere Typisierung des Lungenkrebs und eine Reduzierung der Krebsmortalität ermöglicht werden könnte.

In dieser Arbeit wird die kostensensitive Klassifikation der Lungenkrebssubtypen und von gesundem Lungengewebe als mehrkriterielles Optimierungsproblem formuliert und mittels der genannten EMOAs gelöst. Auf Basis der immunhistochemischen Färbungen verschiedener Proteine, die sich in ihren Anschaffungskosten unterscheiden (vgl. Abschnitt B im Anhang), werden Pareto-optimale Variablenteilmengen für eine gute und zugleich kostengünstige Klassifikation gesucht. Die Güte der Klassifikation wird durch die Fehlklassifikationsrate und die Kosten durch die Variablenanzahl oder die finanziellen Kosten der Variablenmessungen beschrieben.

Zur Untermauerung der Ergebnisse bezüglich des VIM-basierten Rekombinationsoperators aus dem medizinischen Anwendungsbeispiel wird eine Simulationsstudie durchgeführt. Die Simulationsstudie basiert auf der Hypothese, dass NHEMOtree mit dem VIM-basierten Rekombinationsoperator X_{VIM} schneller konvergiert als mit dem Standard-Rekombinationsoperator.

1.4 Aufbau der Arbeit

Das nachfolgende Kapitel 2 beschreibt die Hauptkonzepte der Entscheidungsbäume, die als umhüllte Klassifikationsalgorithmen im mehrkriteriellen *Wrapper*-Ansatz angewendet werden. Da die Güte eines Klassifikationsmodells stets von der Auswahl geeigneter Variablen abhängt, geht mit der Konstruktion eines guten Klassifikationsmodells eine Variablenselektion einher. Verschiedene Variablenselektionsmethoden und einkriterielle VIMs zur Beschreibung der globalen Wichtigkeit einzelner Variablen werden daher in Kapitel 3 vorgestellt. In Kapitel 4 werden die Grundlagen der Evolutionären Algorithmen (EAs) geschaffen und in den Abschnitten 4.2 und 4.3 die Besonderheiten der EAs mit den hier verwendeten Repräsentationsformen erläutert. In Kapitel 5 werden verschiedene mehrkriterielle Optimierungsansätze, sowie

der entwickelte Dominanzquotient zum Vergleich zweier Pareto-Fronten vorgestellt. Ferner werden die wichtigsten Vertreter der EMOAs (NSGA-II und SMS-EMOA), das Problem der Konvergenzanalyse in EMOAs und verschiedene Abbruchkriterien beschrieben.

Kapitel 6 beinhaltet die in dieser Arbeit entwickelten Verfahren und Operatoren zur Lösung mehrkriterieller Optimierungsprobleme. Zunächst wird der verwendete mehrkriterielle *Wrapper*-Ansatz in Abschnitt 6.1 erläutert. Anschließend wird NHEMOtree in Abschnitt 6.2 beschrieben und in Abschnitt 6.3 die mehrkriteriellen VIMs für X_{VIM} (vgl. Abschnitt 6.4) entwickelt.

Ausführungen zur Implementierung der in dieser Arbeit entwickelten Verfahren finden sich in Kapitel 7. Die Anwendung des NHEMOtrees und des mehrkriteriellen *Wrapper*-Ansatz auf echte und simulierte Daten, der Vergleich beider Ansätze zur Lösung mehrkriterieller Optimierungsprobleme, sowie die Bewertung der verschiedenen Operatoren des NHEMOtrees erfolgt in den Kapiteln 8 und 9. Zuletzt werden in Kapitel 10 die wichtigsten Resultate zusammengefasst und ein Ausblick gegeben.

2 Entscheidungsbäume

In vielen Anwendungen der Statistik und des Data Minings ist die Klassifikation eine zentrale Aufgabe (vgl. Espejo et al., 2010; Jabeen und Baig, 2010b; Alba et al., 2007). Klassifikationsverfahren werden in überwachte und nicht-überwachte Algorithmen je nach Vorliegen von Klassenzugehörigkeiten unterteilt (vgl. Han und Kamber, 2006), wobei in dieser Arbeit lediglich überwachte Klassifikationsverfahren zum Einsatz kommen.

Die überwachte Klassifikation besteht aus der Herleitung eines Modells h, das den Zusammenhang zwischen den Einflussvariablen und den vorliegenden Klassen beschreibt, sodass Strukturen in den Daten erklärt und zukünftige Beobachtungen ohne bekannte Klassenzugehörigkeit auf Basis der gelernten Beziehungen einer Klasse zugeordnet werden können. Die Beschreibung eines überwachten Klassifikationsproblems liefert Definition 1.

Definition 1 (Überwachtes Klassifikationsproblem)
Sei $D = \{(x_1,k_1),\ldots,(x_n,k_n)\} \subseteq \mathcal{X} \times \mathcal{K}$ *eine Menge von Beobachtungspaaren* (x_i,k_i), $i = 1,\ldots,n$. *Sei* $\mathcal{X} = \{X_1,\ldots,X_p\}$ *eine endliche Menge von Zufallsvariablen und* $\mathcal{K}$ *eine endliche Menge von Klassen. Dann beschreibt*

$$h : \mathcal{X} \rightarrow \mathcal{K}$$

ein überwachtes Klassifikationsmodell.
Das Auffinden eines möglichst guten Modells h wird als überwachtes Klassifikationsproblem bezeichnet.

Überwachte Klassifikationsprobleme können mit Hilfe von Klassifikationsalgorithmen (vgl. Definition 2), wie etwa der linearen Diskriminanzanalyse (vgl. Fisher, 1936), Naive Bayes-Methoden (vgl. Rish, 2001), Support Vector Machines (vgl. Cortes und Vapnik, 1995) oder Entscheidungsbäumen (vgl. Hastie et al., 2009; Breiman et al., 1998) gelöst werden.

Definition 2 (Algorithmus für ein überwachtes Klassifikationsproblem)
Sei $h : \mathcal{X} \rightarrow \mathcal{K}$ *ein überwachtes Klassifikationsmodell, das den Zusammenhang zwischen den Einflussvariablen* $\mathcal{X} = \{X_1,\ldots,X_p\}$ *und den Klassen* $\mathcal{K}$ *beschreibt.*
Ein Algorithmus

$$A(\mathcal{X},D) = \{s, h(\mathcal{X}_s,D)\}$$

mit Variablenselektion $s \in \{0,1\}^p$*,* $\mathcal{X}_s \subset \mathcal{X}$*, und überwachtem Klassifikationsmodell* $h(\mathcal{X}_s,D)$ *beschreibt einen Klassifikationsalgorithmus auf der Menge* $\mathcal{X}$ *und den Beobachtungen* $D = \{(x_1,k_1),\ldots,(x_n,k_n)\} \subseteq \mathcal{X} \times \mathcal{K}$*.*

In dieser Arbeit werden überwachte Klassifikationsprobleme durch Entscheidungsbäume gelöst. Der allgemeine Aufbau von Entscheidungsbäumen und Unterschiede zwischen verschiedenen Entscheidungsbaumarten werden in Abschnitt 2.1 erläutert. Ferner wird die Gini-Wichtigkeit als gängiges Gütekriterium zur Splitauswahl des bekannten Entscheidungsbaumalgorithmus CART (vgl. Breiman et al., 1998) beschrieben. Die Gini-Wichtigkeit wird auch später in NHEMOtree zur lokalen Cutoff-Optimierung verwendet (vgl. Abschnitt 6.5). Die Schwächen der Entscheidungsbäume werden in Abschnitt 2.2 erläutert und in Abschnitt 2.3 folgt ein kurzes Fazit.

2.1 Konstruktion von Entscheidungsbäumen

Entscheidungsbäume stellen Entscheidungsregeln durch eine baumartige Struktur bestehend aus einem Wurzelknoten, Kanten, internen Knoten und Blättern dar (vgl. Abbildung 1). Jedes Blatt ist einer bestimmten Klasse zugeordnet, wobei zu einer Klasse mehrere Blätter existieren können (vgl. Han und Kamber, 2006). Für die Klassifikation einer Beobachtung wird beginnend am Wurzelknoten der Entscheidungsbaum durchlaufen bis ein Blatt erreicht ist und damit der Beobachtung eine Klasse zugeordnet werden kann. Durch ihre Struktur sind Entscheidungsbäume gut verständlich, einfach zu interpretieren und damit auch populär für die Lösung von Klassifikationsproblemen (vgl. Hastie et al., 2009; Zhao, 2007; Berney et al., 2010; Kim et al., 2010; Righi et al., 2012; Vendrame et al., 2012).

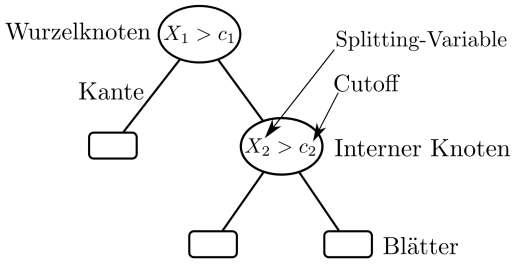


Abbildung 1: Schematische Darstellung eines binären Entscheidungsbaums

Entscheidungsbäume verbinden eine Folge einfacher Abfragen logisch miteinander, wobei jede Abfrage eine oder mehrere numerische Variablen mit einem Schwellwert (Cutoff) bzw. nominale Variablen mit einer Menge möglicher Werte vergleicht. Jede Kante in einem Baum kennzeichnet das Ergebnis einer solchen Abfrage. Diese Abfragen werden als Splittingkriterien bezeichnet, da sie die Beobachtungen D in disjunkte Teilmengen $D_1,\ldots,D_d$ partitionieren.

Die meisten Entscheidungsbaumlgorithmen beginnen mit einem Wurzelknoten, der zunächst alle Beobachtungen D enthält. Ein neuer Knoten wird für jede disjunkte Teilmenge $D_i, i = 1, \ldots, d$, an den Entscheidungsbaum angehängt, indem ein Splittingkriterium zur weiteren Partitionierung von D bestimmt wird. Hierzu wird an jedem Knoten eines Entscheidungsbaumes die Variable mit entsprechendem Splittingkriterium ausgewählt, die zur Unterteilung der Beobachtungen an diesem Knoten am nützlichsten ist und ein Gütekriterium maximiert. Neue Knoten werden dem Entscheidungsbaum als Tochterknoten angehängt. Danach wird die Partitionierung rekursiv fortgesetzt. Eine disjunkte Teilmenge, in der alle Tupel einer identischen Klasse angehören, wird nicht weiter aufgeteilt und das Blatt entsprechend seiner Klassenzugehörigkeit benannt. Auf diese Weise entsteht ein Entscheidungsbaum, der jeder Beobachtung der Trainingsdaten perfekt eine Klasse zuordnet (vgl. Algorithmus 1).

Anhand der Kantenanzahl pro Knoten wird zwischen binären und nicht-binären Entscheidungsbäumen unterschieden. Binäre Entscheidungsbäume besitzt je Knoten genau zwei Kanten mit einem rechten und einem linken Tochterknoten. Bei nicht-binären Bäumen sind es mehr als zwei Kanten, sodass die Anzahl der Splittingebenen im Vergleich zum binären Entscheidungsbaum häufig reduziert ist. Da mehrwegige Splits durch eine Folge von Binärsplits dargestellt werden können, werden häufig in der Praxis und so auch hier letztere bevorzugt (vgl. Hastie et al., 2009).

Außerdem werden in dieser Arbeit univariate Entscheidungsbäume verwendet. Diese nutzen genau eine Variable je Splittingkriterium und unterteilen den Variablenraum durch achsenparallele Hyperebenen. Alternativ können auch Linearkombinationen oder Nicht-Linearkombinationen aus mehreren Variablen getestet werden. Auf diese Weise wird der Variablenraum durch komplexere Entscheidungsfunktionen partitioniert und der Suchraum stark vergrößert.

Der in der statistischen Praxis sehr häufig verwendete Entscheidungsbaum-Algorithmus ist CART mit seinem geringen Rechenaufwand und den wenigen notwendigen Parametereinstellungen. CART ist ein rekursiver Partitionierungsalgorithmus, der sowohl Klassifikations- als auch Regressionsprobleme lösen kann, wobei in dieser Arbeit lediglich Klassifikationsprobleme von Interesse sind. CART verwendet zur Erstellung der binären Entscheidungsbäume mit univariater Partitionierung des Variablenraums ein Zweistufenmodell.

In der ersten Stufe wächst der Baum t_{max} ausgehend vom Wurzelknoten bis er perfekt die Trainingsdaten klassifiziert oder ein weiteres Wachstum nicht mehr möglich ist. CART verwendet in jedem Knoten eine zufällige Auswahl an Variablen, für die der beste Split, d.h. die Variable mit dem besten Cutoff, in dem entsprechenden Knoten bestimmt wird. Die Bestimmung der besten Partitionierung der Beobach-

tungen D in einem Knoten ν des Baums t in die möglichen Tochterknoten ν_{l,X_j,c_i} und ν_{r,X_j,c_i} durch Teilen bezüglich der Variablen X_j im Cutoff c_i erfolgt dabei bezüglich eines Gütekriteriums $\gamma(\nu, D)$.

Das vorherrschende Gütekriterium zur Auswahl der Splits in CART ist die Gini-Wichtigkeit γ^G (vgl. Gleichung (1)). Diese wird unter anderem auch in NHEMOtree zur Optimierung der Cutoffs verwendet (vgl. Abschnitt 6.5). Ein weiteres weit verbreitetes Gütekriterium ist etwa die Shannon-Entropie (vgl. Shannon, 1948).

Die Gini-Wichtigkeit γ^G beschreibt die maximale Verringerung des Gini-Index γ^g, die aus dem optimalen Split der Beobachtungen D im Knoten ν bezüglich der entsprechenden Variable X^* mit optimalen Cutoff c^* folgt. Es gilt

$$\begin{aligned}\gamma^G(\nu; D) &= \max_{X_j,c_i} \left\{\Delta\gamma^g(\nu_{X_j,c_i}; D)\right\} \\ &= \max_{X_j,c_i} \left\{\gamma^g(\nu_{X_j,c_i}; D) - \gamma^g(\nu_{l,X_j,c_i}; D) - \gamma^g(\nu_{r,X_j,c_i}; D)\right\}, \qquad (1) \\ &\quad \forall i = 1,...,q_j,\ j = 1,...,p.\end{aligned}$$

Dabei misst der Gini-Index die Unreinheit einer Datenpartition D in einem Knoten ν eines Baums t als

$$\gamma^g(\nu; D) = 1 - \sum_{k=1}^{K} p_k^2 \qquad \in [0, 1]$$

mit der relativen Häufigkeit p_k, dass eine Beobachtung in ν zur Klasse $k \in \{1, \ldots, K\}$ gehört. Wenn alle Klassen im Knoten ν gleich häufig vertreten sind, ist $\gamma^g(\nu; D)$ maximal. Die Unreinheit beträgt Null, wenn der Knoten aus nur einer Klasse besteht. Die Variable mit optimalem Cutoff, die zur Maximierung der Gini-Wichtigkeit im Knoten ν führt und somit den größten Informationsgewinn erzielt, wird für den Split gewählt und teilt die Beobachtungen in den linken und rechten Tochterknoten (vgl. Breiman et al., 1998).

Bei der Generierung von Entscheidungsbäumen besteht eine wesentliche Aufgabe darin, den Entscheidungsbaum möglichst klein zu halten, sowie Überanpassung und unnötige Blätter zu vermeiden. In vielen Entscheidungsbaumalgorithmen, so auch in CART, erfolgt dies mittels *Pruning* (vgl. Definition 3), das sowohl während als auch nach der Baumentwicklung stattfinden kann (vgl. Petersohn, 2005).

Definition 3 (Pruning)
Für einen Baum t mit innerem Knoten ν besteht das Pruning des Subbaums t_ν aus der Entfernung aller nachfolgenden Knoten von ν und den entsprechenden Kanten. Der Knoten ν ist dann ein Blatt.

Im zweiten Schritt des Zweistufenmodells wird in CART der maximale Baum t_{max} mit Hilfe des minimalen Kosten-Komplexitäts-Pruning (vgl. Breiman et al., 1998) schrittweise verkleinert. Durch die Verkleinerung des Entscheidungsbaums können Einflussvariablen aus der Menge der potentiellen Einflussvariablen entfernt werden, sodass eine zusätzliche Variablenselektion durchgeführt wird.

Weitere bekannte Entscheidungsbaumalgorithmen neben CART sind etwa der Iterative Dichotomiser 3 (ID3) von Quinlan (1979) als Vorgänger des vor allem in der Informatik weit verbreitetem C4.5-Algorithmus von Quinlan (1993). Algorithmus 1 beschreibt einen Meta-Algorithmus zur Erstellung binärer Entscheidungsbäume.

Algorithmus 1: Binärer Entscheidungsbaum

Definition:

t	Binärer Entscheidungsbaum
ν	Knoten mit $\nu = \{X_i, c_i, \nu_l, \nu_r\}$

Eingabe:

$D \subseteq \mathcal{X} \times \mathcal{K}$	Beobachtungen mit p Variablen $X_i \subset \mathcal{X}$ und Klassen $\mathcal{K}$
$\gamma(\nu; D)$	Gütemaß der Beobachtungen D im Knoten ν zur Bewertung der Knotenauswahl
ψ	Schwellwert

Ausgabe: Binärer Entscheidungsbaum t

Initialisierung:

$V = \varnothing$ `/* Menge potentieller Blätter */`
Erstelle Wurzelknoten ν
$V = V \cup \nu$
$D = D_\nu$ `/* Zuordnung aller Daten zum Wurzelknoten */`
$t \leftarrow \nu$ `/* Zuordnung des Wurzelknotens zu Baum` t `*/`

Binärer Entscheidungsbaum:

for $\nu \in V$ **do**
  if $\gamma(\nu; D_\nu) \geq \psi$ **then**
    ν ist ein Blatt
  else
    Berechne neue Tochterknoten ν_l und ν_r für ν und D_ν
    $\nu \leftarrow \{\nu_l, \nu_r\}$ `/* Zuordnung der Tochterknoten zu` ν `*/`
    $V = V \cup \{\nu_l, \nu_r\}$
    $D_\nu = D_{\nu_l} \cup D_{\nu_r}, D_{\nu_l} \cap D_{\nu_r} = \varnothing$ `/* Aufteilung von` D_ν `*/`
  end
end
Ausgabe von t

2.2 Schwächen von Entscheidungsbäumen

Neben den Vorteilen der guten Interpretierbarkeit und der breiten Anwendungsmöglichkeiten auf diverse Datenarten besitzen Entscheidungsbäume jedoch auch Schwächen. Breiman (1996) und Breiman (2001) diskutieren etwa das Problem der Instabilität von Entscheidungsbaum-Lösungen, bei denen bereits kleine Änderungen in den Beispieldaten zu sehr unterschiedlichen Baumstrukturen führen können. Da in binären Entscheidungsbäumen der beste Cutoff innerhalb einer Einflussvariable sowohl die Variable zum Splitting bestimmt als auch die Beobachtungen in zwei neue Knoten aufteilt, kann sich die gesamte Baumstruktur bei einer anderen Wahl des ersten Cutoffs ändern.

Ferner können aufgrund starker Korrelationsstrukturen unwichtige Variablen als wichtig erachtet werden. Denn der beste Cutoff des aktuellen Knotens hängt stets von den vorangegangenen Knoten und Datenpartitionen ab. Somit ist es möglich, dass eine unwichtige, die Klassenzugehörigkeit nur schwach beeinflussende Variable, die aber stark mit einer wichtigen Einflussvariable korreliert ist, in etwa gleich oft zur Aufteilung des Variablenraums herangezogen wird, wie eine tatsächlich wichtige Einflussvariable.

Ähnlich zur Vorwärtsselektion bei Regressionsmodellen garantiert die lokale Optimierung in jedem Splittingschritt keine globale Optimierung der gesamten Baumstruktur. Rekursiv partitionierte Entscheidungsbäume, wie CART, zählen zu den Greedy-Algorithmen (vgl. Cormen et al., 2001), da sie in jedem Splittingschritt die Partitionierung auswählen, die zum Zeitpunkt der Wahl, d.h. lokal, das beste Ergebnis verspricht. Somit kann eine Variable, die bei der Verwendung von Greedy-Algorithmen als nützlich erscheint, dies jedoch nicht mehr bei gemeinsamer Betrachtung mit anderen Variablen sein (vgl. Turney, 1995).

Eine weitere Schwäche der Entscheidungsbäume basiert auf der Verzerrung der Gini-Wichtigkeit und der daraus resultierenden häufigeren Selektion von Variablen mit mehr Kategorien (vgl. Strobl et al., 2007b,a; Kim und Loh, 2001). Da die Anzahl potentieller Cutoffs exponentiell mit den Kategorien ungeordneter Einflussvariablen steigt, führen diese Variablen mit einer höheren Wahrscheinlichkeit bereits zufällig zu guten Kriterienwerten und werden in CART-ähnlichen Entscheidungsbäumen präferiert. Es existieren verschiedene Ansätze zur Verminderung des Variablenselektionsbias, wie etwa das unverzerrte p-Wert-Kriterium von Dobra und Gehrke (2001), die Trennung von Variablen- und Cutoff-Auswahl (vgl. Loh und Shih, 1997) oder unverzerrte Entscheidungsbaumalgorithmen basierend auf bedingten Inferenztests (vgl. Hothorn et al., 2006).

Eine Alternative zu Einzelbaumalgorithmen stellen Ensemblemethoden dar, die einzelne Klassifikationsmodelle, wie etwa Entscheidungsbäume, in einem Modell aggregieren und somit stabilere Vorhersagen mit einer höheren Vorhersagegenauigkeit liefern (vgl. Skurichina und Duin, 2002). Vorschläge hierfür sind etwa Bagging (vgl. Breiman, 1996), Random Forests (vgl. Breiman, 2001), Boosting (vgl. Freund, 1995) oder Stacking (vgl. Wolpert, 1992). Nachteile der Ensemblemethoden sind der deutlich erhöhte Rechenaufwand und die zusätzlichen Verzerrungsquellen in der Variablenselektion, die ebenfalls zugunsten von Variablen mit vielen Kategorien agieren (vgl. Strobl et al., 2007b, 2008). Des Weiteren führt die Aggregation der einzelnen Klassifikationsmodelle zu einem nicht mehr leicht zu interpretierenden Modell und der Einfluss einzelner Variablen auf die Zielvariable ist in einem Ensemble nur mit Hilfe von VIMs möglich. Es existieren verschiedene VIMs (vgl. Kapitel 3), die zur Unterscheidung von wichtigen und unwichtigen Variablen herangezogen werden können und die Variablenselektion erleichtern.

2.3 Fazit

Dieses Kapitel dokumentiert die weitverbreitete Anwendung von Entscheidungsbäumen bei überwachten Klassifikationsproblemen. Ursächlich für die Beliebtheit der Entscheidungsbäume ist vor allem ihre leichte Interpretierbarkeit. Entscheidungsbäume existieren in vielfältigen Darstellungsformen mit binären oder mehrwegien, univariaten oder multivariaten, sowie linearen und nicht-linearen Splits mit entsprechenden Partitionierungen des Variablenraums. Diese unterschiedlichen Baumstrukturen lassen sich leicht mit Hilfe von EAs erzeugen (vgl. Abschnitt 4.3). In dieser Arbeit dient CART als Referenz-Entscheidungsbaumalgorithmus, sodass aus Gründen der Vergleichbarkeit im weiteren Verlauf lediglich univariat binäre Entscheidungsbäume verwendet werden.

In Kapitel 6 wird der hier entwickelte NHEMOtree zur Konstruktion univariat binärer Entscheidungsbäume zur Lösung mehrkriterieller Optimierungsprobleme vorgestellt. In Rahmen des Algorithmus werden wie bei Ensemblemethoden VIMs berechnet. Diese werden dann für die Auswahl geeigneter Rekombinationsknoten im Entscheidungsbaum verwendet (vgl. Abschnitt 6.4). Außerdem erfolgt die von Loh und Shih (1997) propagierte Trennung der Variablen- und Cutoff-Auswahl zur Vermeidung des Variablenselektionsbias auf ähnliche Weise in NHEMOtree mit lokaler Cutoff-Optimierung. Dabei basiert die Variablenselektion und die Modellierung der Baumstruktur auf den Variationsoperatoren des NHEMOtrees und die Cutoff-Optimierung auf der allgemeinen Fehlklassifikationsrate bzw. der Gini-Wichtigkeit in dem jeweiligem Knoten (vgl. Abschnitt 6.5).

3 Variablenselektion und Wichtigkeitsmaße

Die Güte eines Klassifikationsmodells hängt stets von der Auswahl geeigneter Variablen aus der Menge aller verfügbaren Einflussvariablen ab. Einen Überblick über diverse Strategien zur Auswahl der besten Variablenteilmenge liefern etwa Liu und Yu (2005), Guyon und Elisseeff (2003) und Jain und Zongker (1997). Wie bereits in Kapitel 2 erwähnt, wird in dieser Arbeit das Problem der überwachten Klassifikation betrachtet und deshalb auch in diesem Kapitel nur die Variablenselektion für überwachte Klassifikationsprobleme erörtert. Spezielle Literatur für die Variablenselektion in nicht-überwachten Lernproblemen existiert etwa von Zhang et al. (2012), Varshavsky et al. (2006), Dy und Brodley (2004) und Hastie et al. (2000).

Die Ziele der Variablenselektion in überwachten Klassifikationsproblemen sind die Verbesserung des Klassifikationsmodells, das leichtere Verständnis des finalen Klassifikationsmodells und des zugrundeliegenden datengenerierenden Prozesses, sowie die schnellere Bereitstellung der Modelle (vgl. Guyon und Elisseeff, 2003). Außerdem sinken mit jeder entfernten Variablen der Aufwand der Datenerhebung, die damit zusammenhängenden Kosten, sowie die benötige Rechenleistung, da mit der Reduzierung der Variablenmenge die Anzahl nötiger Beobachtungen exponentiell abnimmt (vgl. Jain und Zongker, 1997).

Die Methoden der Variablenselektion lassen sich in *Filter*-, *Wrapper*- und *Embedded*-Methoden einteilen. Sie unterscheiden sich hinsichtlich der Bewertung der Variablenteilmenge und der möglichen Interaktion mit einem Klassifikationsalgorithmus (vgl. Guyon und Elisseeff, 2003). Jedoch haben alle Variablenselektionsverfahren gemein, dass ein Gütekriterium $\gamma(\mathcal{X}; D)$ basierend auf verschiedenen Kombinationen der Variablen $X_1, \ldots, X_p \subset \mathcal{X}$ und Beobachtungen D für die gegebenen Klassen maximiert werden soll. Das Problem der Variablenselektion ist, dass bereits für kleine p die Bewertung $\gamma(\mathcal{X}; D)$ für alle 2^p Variablenteilmengen lange Berechnungsdauern mit sich bringt. Alternativen sind schrittweise Methoden, die die sequentiellen Variablenteilmengen aller 2^p möglichen Modelle prüfen. Populäre Varianten dieser Methode sind die Vorwärts- und die Rückwärtssuche, sowie Kombinationen aus beiden (vgl. Liu und Yu, 2005). Kriterien zur Modellevaluierung sind in Regressionmodellen etwa das Bestimmtheitsmaß (vgl. Hartung et al., 2002), Akaikes Informationskriterium (AIC) (vgl. Akaike, 1974) oder das Bayes'sche Informationskriterium (BIC) (vgl. Schwarz, 1978).

Ferner sind Gütemaße zur Bewertung der Wichtigkeit einzelner Variablen für die Klassifikation und Regression unverzichtbar. Auch in Entscheidungsbaumalgorithmen basiert die Variablenauswahl auf VIMs, wie etwa die Gini-Wichtigkeit (vgl. Gleichung (1) in Abschnitt 2.1). Viele einfache Verfahren zur Variablenselektion

nehmen jedoch die Unabhängigkeit der Variablen an und wählen sie bezüglich ihrer individuellen Güte für die Prognose der Klassenzugehörigkeit aus. Unzureichende Annahmen über die Variableninteraktionen führen häufig zu nicht zufriedenstellenden Variablenteilmengen. Um dies zu verhindern, muss der Zusammenhang zwischen den Variablen bei der Selektion berücksichtigt werden, wie dies etwa bei der vollständigen, heuristischen oder randomisierten Suche geschieht (vgl. Dash und Liu, 1997). Direkte Suchverfahren wie etwa EAs (vgl. Kapitel 4) benutzen randomisierte Stichprobenprozesse und wurden bereits zur Variablenselektion erforscht (vgl. Siedlecki und Sklansky, 1989; Kudo und Sklansky, 2000). Ihre Verwendung in Verbindung mit der Variablenselektion ist insbesondere in dieser Arbeit nützlich, weil EAs auch im Rahmen der mehrkriteriellen Variablenselektion eingesetzt werden können (vgl. Abschnitt 5.3).

Nachfolgend werden in den Abschnitten 3.1, 3.2 und 3.3 *Filter*-, *Wrapper*- und *Embedded*-Methoden beschrieben. Danach werden verschiedene einkriterielle VIMs zur Unterscheidung von wichtigen und unwichtigen Variablen in Baum-basierten Ensembles vorgestellt (vgl. Abschnitt 3.4). Auf Basis der einkriteriellen VIMs werden später mehrkriterielle VIMs entwickelt, die dann zur Adaption des Standard-Rekombinationsoperators in NHEMOtree verwendet werden (vgl. Kapitel 6).

3.1 *Filter*-Methoden

Filter-Methoden bewerten die Relevanz von Variablen bezüglich der spezifischen Eigenschaften der Daten. In den meisten Fällen wird zunächst die Variablenwichtigkeit mit einem Maß $\gamma(\mathcal{X}; D)$ berechnet und danach die Variablen mit geringer Wichtigkeit entfernt. Die selektierten Variablen dienen dann als Eingabe für einen Klassifikationsalgorithmus, sodass *Filter*-Methoden eine Vorverarbeitung darstellen und das Problem der Variablenselektion unabhängig von dem Klassifikationsschritt durchführen. Der Meta-Algorithmus der *Filter*-Methoden ist in Algorithmus 2 dargestellt.

Vorteile der *Filter*-Methoden sind ihre geringe Rechenintensität und ihre Unabhängigkeit von dem Klassifikationsalgorithmus, sodass die Variablenselektion nur einmal durchgeführt werden muss und danach verschiedene Klassifikationsverfahren eingesetzt werden können (vgl. Saeys et al., 2007). Häufig werden *Filter*-Methoden zur Erstellung von Variablen-Ranglisten in der Microarray-Analyse zur Auswahl von Genen oder Proteinen verwendet (vgl. Guyon und Elisseeff, 2003; Varshavsky et al., 2006; Zhang et al., 2012). Allerdings sind die meisten *Filter*-Methoden univariat. Sie betrachten die Variablen einzeln und ignorieren Abhängigkeiten zwischen den Variablen, sodass im Vergleich zu anderen Variablenselektionsverfahren eine schlechtere

Klassifikationsgüte erreicht wird und auch redundante Variablen ausgewählt werden können (vgl. Ressom et al., 2007).

In Klassifikationsproblemen können die Variablen ferner nach ihrer individuellen Wichtigkeit für die Vorhersage in Form der Fehlklassifikationsrate, der Sensitivität oder Spezifität in eine Reihenfolge gebracht werden (vgl. Guyon und Elisseeff, 2003). Existieren jedoch verschiedene Variablen, die die Daten perfekt trennen, können *Filter*-Methoden nicht zwischen den besten Variablen unterscheiden. Um den Nachteil der missachteten Variablenabhängigkeiten in den univariaten *Filter*-Methoden zu beheben, wurden multivariate *Filter*-Methoden entwickelt (vgl. Yu und Liu, 2004). Beispiele diverser *Filter*-Algorithmen finden sich in Kohavi und John (1997).

Algorithmus 2: *Filter*-Methode

Eingabe:

$D \subseteq \mathcal{X} \times \mathcal{K}$	Beobachtungen mit p Variablen $X_i \subset \mathcal{X}$ und Klassen $\mathcal{K}$
$\gamma(X_i; D)$	Gütemaß zur Bewertung der Variable X_i
ψ	Schwellwert

Ausgabe:

Beste Variablenteilmenge $\mathcal{X}^*$

Initialisierung:

$\mathcal{X}^* = \varnothing$

***Filter*-Methode:**

for $i = 1 \dots p$ **do**
  if $\gamma(X_i; D)$ *besser als* ψ **then**
    $\mathcal{X}^* = \mathcal{X}^* \cup X_i$
  end
end

3.2 *Wrapper*-Methoden

Im Gegensatz zu *Filter*-Methoden verbinden *Wrapper*-Methoden die Variablensuche mit der Erstellung eines Klassifikationsmodells, indem eine Suchprozedur im Raum der möglichen Variablenteilmengen einen Klassifikationsalgorithmus umhüllt. Dabei werden verschiedene Teilmengen durch den *Wrapper* erstellt und durch das spezifische Klassifikationsmodell bewertet, sodass die Güte des Klassifikationsmodells bei der Variablenselektion berücksichtigt wird. Die Gütebewertung erfolgt üblicherweise mittels einer Validierungsmenge oder durch Kreuzvalidierung. Da jedoch die Anzahl der Variablenteilmengen exponentiell mit der Anzahl der Variablen wächst, kann lediglich bei kleinen Variablenanzahlen eine vollständige Suche zum Auffinden der optimalen Variablenteilmenge durchgeführt werden. Alternativen bieten laut Kohavi

und John (1997) etwa EAs (vgl. Kapitel 4), die sequentielle Suche (vgl. Xiong et al., 2001) oder Schwarmalgorithmen (vgl. Ressom et al., 2007).

Algorithmus 3: *Wrapper*-Methode

Eingabe:

$D \subseteq \mathcal{X} \times \mathcal{K}$ Beobachtungen mit p Variablen $X_i \subset \mathcal{X}$ und Klassen $\mathcal{K}$
$h(\mathcal{X}_s; D)$ Klassifikationsmodell basierend auf $\mathcal{X}_s \subset \mathcal{X}$
$\gamma(h(\mathcal{X}_s; D); D)$ Gütemaß des Modells h bzgl. $\mathcal{X}_s$ und D
δ Abbruchkriterium

Ausgabe:

Beste Variablenteilmenge $\mathcal{X}^*$ mit $h(\mathcal{X}^*; D)$

Initialisierung:

$\mathcal{X}^* = \varnothing$
$\psi^* = \gamma(h(\varnothing; D); D) := 0$ `/* Bewertung von` $\varnothing$ `durch Modell` h `auf` D `*/`

***Wrapper*-Methode:**

while *Abbruchkriterium* δ *nicht erfüllt* **do**
  $\mathcal{X}_s \subseteq \mathcal{X}$ `/* Bilden einer neuen Variablenteilmenge` $\mathcal{X}_s$ `*/`
  $\psi = \gamma(h(\mathcal{X}_s; D); D)$ `/* Bewertung von` $\mathcal{X}_s$ `durch Modell` h `auf` D `*/`
  if ψ *besser als* ψ^* **then**
    $\psi^* = \psi$
    $\mathcal{X}^* = \mathcal{X}_s$
  end
end

Vorteile der *Wrapper*-Methoden sind die Interaktionen zwischen Variablensuche und Modellerstellung, die häufig im Gegensatz zu *Filter*-Methoden mit der Erstellung genauerer Klassifikationsmodelle einhergeht, sowie die Möglichkeit, Abhängigkeiten zwischen den Variablen zu beachten. Allerdings ist das Risiko der Überanpassung bei *Wrapper*-Methoden erhöht und sie sind rechenintensiver, da sie für jede Teilmenge ein Modell erstellen (vgl. Saeys et al., 2007). Der Meta-Algorithmus der *Wrapper*-Methode ist in Algorithmus 3 beschrieben (vgl. Liu und Yu, 2005). Er unterscheidet sich durch die Art der Variablenteilmengenbewertung vom Meta-Algorithmus der *Filter*-Methode (vgl. Algorithmus 2).

Populäre Klassifikationsverfahren innerhalb eines *Wrappers* sind etwa Entscheidungsbäume (vgl. Cherkauer und Shavlik, 1996), k-Nächste-Nachbarn (vgl. Hamdani et al., 2007) oder Support Vector Machines (SVM) (vgl. Garcia-Nieto et al., 2009; Ressom et al., 2007). Aber auch die Kombination diverser Klassifikationsverfahren innerhalb eines *Wrappers* ist möglich (vgl. Chrysostomou et al., 2011).

3.3 *Embedded*-Methoden

Embedded-Methoden vollziehen eine Variablenselektion während der Optimierung und sind für einen spezifischen Lernalgorithmus ausgelegt (vgl. Guyon und Elisseeff, 2003). Im Gegensatz zu *Filter*- und *Wrapper*-Methoden können die Variablenselektion und -bewertung durch den Lernalgorithmus in *Embedded*-Methoden nicht getrennt werden. Dadurch haben sie den Vorteil, dass sie mit dem Klassifikationsmodell interagieren können und gleichzeitig weniger rechenintensiv als *Wrapper*-Methoden sind (vgl. Saeys et al., 2007). In Algorithmus 4 ist der Meta-Algorithmus der *Embedded*-Methode aufgeführt.

Beispielsweise die Variablenselektion in CART, aber auch der in dieser Arbeit entwickelte NHEMOtree (vgl. Abschnitt 6.2) stellt eine *Embedded*-Methode dar. Eine Übersicht über diverse *Embedded*-Methoden existiert etwa von Lal et al. (2006).

Algorithmus 4: *Embedded*-Methode

Eingabe:

$D \subseteq \mathcal{X}^p \times \mathcal{K}$	Beobachtungen mit p Variablen $X_i \subset \mathcal{X}^p$ und Klassen $\mathcal{K}$
$A(\mathcal{X}; D)$	Algorithmus zur Variablenselektion und Modellerstellung
$\gamma(A(\mathcal{X}; D); D)$	Gütemaß des Algorithmus A bzgl. $\mathcal{X}$ auf D
δ	Abbruchkriterium

Ausgabe:

Beste Variablenteilmenge $\mathcal{X}^*$ mit Algorithmus $A(\mathcal{X}^*; D)$

Initialisierung:

$\mathcal{X}^* = \varnothing$

$\psi^* = \gamma(A(\mathcal{X}^*; D); D)$ `/* Bewertung von` $\mathcal{X}^*$ `durch Algorithmus` A `auf` D `*/`

***Embedded*-Methode:**

while *Abbruchkriterium* δ *nicht erfüllt* **do**

$\{\mathcal{X}_s, A(\mathcal{X}_s; D)\} = A(\mathcal{X}, D)$ `/* Variablenselektion und Modellerstellung */`

$\psi = \gamma(A(\mathcal{X}_s; D))$ `/* Bewertung von` $\mathcal{X}_s$ `durch Algorithmus` A `auf` D `*/`

if ψ *besser als* ψ^* **then**

$\psi^* = \psi$

$\mathcal{X}^* = \mathcal{X}_s$

end

end

3.4 Einkriterielle Variablenwichtigkeitsmaße

Methoden zur Selektion potentieller Einflussvariablen $X_1, \ldots, X_p$ basieren häufig auf sogenannten Variablenwichtigkeitsmaßen (VIMs). Im Allgemeinen ist jedes Maß, das zur Bewertung der Wichtigkeit einer Variable für die Klassifikation oder Regression herangezogen wird, ein VIM.

Ein natürliches VIM in Baum-basierten Ensemblemethoden ist die Häufigkeit der Variablen in den Bäumen des Ensembles. Sei $N_t, t = 1, \ldots, T$, die Anzahl korrekt klassifizierter Beobachtungen für jede der *Out-of-bag*-Stichproben $B_1, \ldots, B_T$. Werden jeweils die einzelnen Variablen X_j aus den Bäumen entfernt und die Anzahl N_t^* korrekt klassifizierter Beobachtungen ermittelt, ergibt sich das entsprechende VIM

$$VIM(X_j) = \frac{1}{T} \sum_{t=1}^{T} (N_t - N_t^*).$$

Im Kontext der Logischen Regression (vgl. Ruczinski et al., 2003) ist neben der Wichtigkeit der Variablen auch die Wichtigkeit der Variableninteraktionen von großem Interesse. Diese wird auch in der Erweiterung der Logischen Regression, der sogenannten *LogicFS*, durch Zählen der Modelle, die spezifische Variablen oder Interaktionen beinhalten, ermittelt (vgl. Schwender und Ickstadt, 2008; Schwender et al., 2011).

Im Rahmen von Random Forests schlägt Breiman (2001) ein VIM basierend auf der Reduzierung der Vorhersagegenauigkeit nach der zufälligen Permutation der Werte jeweils einzelner Einflussvariablen vor. Diese Permutierte Fehlerfreiheit ist ein gängiges VIM in Random Forests. Sie misst nicht den Informationszuwachs, sondern vergleicht die Vorhersagegenauigkeit des Modells auf den ursprünglichen Daten mit der Vorhersagegenauigkeit auf den permutierten Daten. Der Einfluss von X_j auf die Klassifikation wird durch Permutation anstelle durch Löschung aufgehoben, da die hierarchische Struktur der Entscheidungsbäume durch das Entfernen einer Variable zerstört werden könnte. Sinkt die Vorhersagegenauigkeit auf den permutierten gegenüber den ursprünglichen Daten wesentlich, spricht dies für die Wichtigkeit von X_j.

Seien B_t wieder die sogenannte *Out-of-bag*-Stichproben für die Bäume $t \in \{1, \ldots, T\}$. Laut Strobl et al. (2008) wird die Wichtigkeit der Variablen X_j in Baum t berechnet als

$$pf(X_j,t) = \frac{\sum_{x_i \in B_t} \mathbb{1}\left(k_i = \hat{k}_i\right)}{|B_t|} - \frac{\sum_{x_i \in B_t} \mathbb{1}\left(k_i = \hat{k}_{i,\pi_j}\right)}{|B_t|}.$$

Für die Beobachtung $x_i \in B_t$ beschreibt dabei $\hat{k}_i$ die vorhergesagte Klasse vor und $\hat{k}_{i,\pi_j}$ die vorhergesagte Klasse nach der Permutation von X_j; $\mathbb{1}$ bezeichnet die Indikatorfunktion. Ist Variable X_j nicht im Baum t enthalten, so gilt $pf(X_j,t) = 0$. Die Permutierte Fehlerfreiheit (PF) beschreibt dann die mittlere Wichtigkeit von X_j über alle Bäume durch

$$PF(X_j) = \frac{\sum_{t=1}^{T} pf(X_j,t)}{T}. \tag{2}$$

Weitere VIMs für den Random Forest stellen etwa Ishwaran (2007) und Sandri und Zuccolotto (2008) vor. VIMs sind in Verbindung mit Random Forests in vielen praktischen Anwendungen bei der Variablenselektion erfolgreich (vgl. Lunetta et al., 2004; Bureau et al., 2005; Díaz-Uriarte und de Andrés, 2006; Menze et al., 2007; Riddick et al., 2011). Andere VIMs beinhalten ein gewichtetes Mittel über die Verbesserungen in den Splittingkriterien, die durch jede Variable in den einzelnen Bäumen erreicht wird (vgl. Friedman, 2001). Ein Beispiel für ein solches Maß ist etwa die Gini-Wichtigkeit (vgl. Gleichung (1) in Abschnitt 2.1). Untersuchungen und Anwendungen weiterer VIMs können unter anderem in den theoretischen Arbeiten von Breiman (2001) und Friedman (2001) gefunden werden.

3.5 Fazit

Dieses Kapitel zu Variablenselektion und Wichtigkeitsmaßen hat die Notwendigkeit der Auswahl geeigneter Variablen zur Erstellung eines guten Klassifikationsmodells erörtert, sowie gängige Variablenselektionsmethoden und einkriterielle VIMs vorgestellt.

Im Allgemeinen wird durch die Variablenselektion vor allem die Vorhersagegüte und die Verständlichkeit des Klassifikationsmodells verbessert. Allerdings existieren bisher keine VIMs für die Anwendung in Verfahren zur Lösung mehrkriterieller Optimierungsprobleme. Die direkte Übertragung einkriterieller VIMs auf mehrkriterielle Optimierungsprobleme (vgl. Kapitel 5), die abhängig von der Baumgröße optimiert werden sollen, funktioniert in der Regel nicht. Daher werden in Abschnitt 6.3 mehrkriterielle VIMs zur Anwendung in mehrkriteriellen Optimierungsalgorithmen entwickelt und in Kapitel 8 und 9 experimentell überprüft.

4 Evolutionäre Algorithmen

Das Prinzip der biologischen Evolution gilt als generelles Erklärungsmodell für die Entwicklung von Lebewesen (vgl. Darwin, 1859). Unter gegebenen Umweltbedingungen werden in der Evolution optimale Lösungen für vielfältige Konstruktionsprobleme gefunden, die sich auch in der Bionik als idealer Weg zur technischen Problemlösung behaupten. So dient etwa die Charakteristik von Blutgefäßsystemen als Basis für die Entwicklung von Auto-Klimaanlagen oder das aerodynamische Verhalten von Vogelflügeln für die Konstruktion von Tragflächen bei Flugzeugen (vgl. Nachtigall, 2010).

Abhängig von der natürlichen Umgebung stehen die verschiedenen Spezies unter einem Evolutionsdruck, der sie zur Anpassung an verschiedene Umweltbedingungen drängt, um so ihren Fortbestand zu sichern. Diese Anpassung basiert auf Populationen aus einzelnen sich zufällig verändernden Individuen. Die genetischen Anlagen der Individuen verbreiten sich aufgrund ihrer phänotypischen Überlegenheit, d.h. aufgrund ihrer besseren Eigenschaften, und durch ihren Fortpflanzungserfolg in der gesamten Population. Die Rekombination von Erbanlagen verschiedener Individuen und zufällige Mutationen spielen eine zentrale Rolle bei der Erzeugung von Innovation und Diversität im Phänotyp. Zugleich erhöht der Fortpflanzungserfolg eines Individuums die Chance, dass Neuerungen in der Population verbleiben und sich verbreiten.

Basierend auf dem Grundprinzip dieser Methodik mit sich wiederholenden Variationen (Mutation und Rekombination) und Selektionsvorgängen entstanden direkte Suchverfahren mit biologischen Vorbildern (vgl. Hooke und Jeeves, 1961), wie etwa die hier verwendeten Evolutionären Algorithmen (EAs) (vgl. Eiben und Smith, 2003), Schwarmalgorithmen (vgl. Dorigo und Di Caro, 1999; Kennedy und Eberhart, 1995) oder memetische Algorithmen (vgl. Radcliffe und Surry, 1994). Direkte Suchverfahren kommen zur Anwendung, wenn etwa Probleme NP-vollständig oder keine exakten analytischen Lösungen in praktikabler Zeit berechenbar sind (vgl. Wegener, 2003). Gradientenverfahren, wie etwa das Newton-Raphson-Verfahren (vgl. Deuflhard und Weiser, 2011), können im Gegensatz zu direkten Suchverfahren nur bei differenzierbaren Optimierungsfunktionen eingesetzt werden. Direkte Suchverfahren müssen hingegen die zu optimierende Funktionen nicht kennen. Es muss lediglich die Berechnung der Funktionswerte für gegebene Einflussvariablen möglich sein, sodass der Parameterraum effizient untersucht werden kann, um ein Optimum zu finden. Dieser Fall der unbekannten Optimierungsfunktionen mit berechenbaren Funktionswerten wird in dieser Arbeit angenommen.

EAs wurden bereits Mitte des letzten Jahrhunderts von Turing als „genetische oder evolutionäre Suche“ beschrieben (vgl. Turing, 1948). Bremermann (1962) führte die ersten Computerversuche zur *Optimierung durch Evolution und Rekombination* durch. Zur gleichen Zeit entstanden unabhängig voneinander drei weitere EAs. Fogel et al. (1966) führten die Evolutionäre Programmierung ein, Rechenberg (1973) und Schwefel (1977) schufen die Evolutionsstrategie und Holland (1975) entwickelte den Genetischen Algorithmus (GA). Anfang der 1990er Jahre wurden diese Algorithmen als Dialekte unter dem Begriff der EAs zusammengefasst (vgl. Eiben und Smith, 2003). Zu dieser Zeit führte Koza (1992) den vierten Dialekt, die Genetische Programmierung (GP), als eine Erweiterung der GAs zur Evolution von Computerprogrammen ein. Heute wird die GP als eine Variante der GAs mit einer komplexeren Repräsentation zur Codierung der Individuen aufgefasst (vgl. Espejo et al., 2010).

Die verschiedenen Dialekte der EAs folgen alle demselben Prinzip. In der natürlichen Evolution streben Individuen nach Überleben und Reproduktion. Ihre Fähigkeit, sich an ihre Umgebung anzupassen, drückt sich in ihrer Überlebenschance und Vermehrungswahrscheinlichkeit aus. In einer Umgebung mit begrenzten Ressourcen kommt es durch den Wettbewerb um diese Ressourcen zu einer natürlichen Auslese, sodass sich die Fitness der Population steigert und die Zahl der Individuen nicht unbegrenzt wächst. Im Kontext der Optimierung stellen die Individuen eine Menge möglicher Lösungen dar, deren Anpassungsfähigkeit durch eine oder mehrere Fitnessfunktionen (vgl. Abschnitt 4.1.1) beschrieben werden. Die Werte der Fitnessfunktionen beschreiben also die Qualität jeder einzelnen Lösung. Auf Basis dieser Fitnesswerte werden bei der Fitness-basierten Selektion die besseren Individuen durch Selektionsoperatoren (vgl. Abschnitt 4.1.4) ausgewählt, um nach Anwendung der Variationsoperatoren Rekombination und Mutation die nächste Populationsgeneration zu bilden (vgl. Abschnitt 4.1.5). Die Umweltselektion sorgt für die Deckelung der Populationsgröße und lässt nur eine bestimmte Anzahl der Individuen entweder aufgrund ihrer Fitness oder ihren Alters in die nächste Generation übergehen (vgl. Abschnitt 4.1.6). Tabelle 1 fasst die technischen Details aller Dialekte zusammen.

Nach der Umweltselektion wird überprüft, ob ein Abbruchkriterium erfüllt ist. Entsprechend ist entweder die finale Population ermittelt oder eine weitere Generation wird begonnen. Die beiden häufigsten Abbruchkriterien sind das Erreichen einer vorgegebenen Anzahl an Generationen oder das erfolgreiche Auffinden einer Lösung mit erwünschter Fitness. Alternativ können etwa die Berechnungen basierend auf der *Online Convergence Detection* (vgl. Wagner et al., 2009) bezüglich eines oder mehrerer Gütemaße abgebrochen werden (vgl. Kapitel 5). In diesem Fall stoppt der Algorithmus, wenn über einen zuvor bestimmten Zeitraum hinweg keine signifikante

Veränderung in der Lösungsgüte der Population stattfand. Weitere Abbruchkriterien für einkriterielle Optimierungen tragen etwa Trautmann et al. (2009) zusammen. Der generelle Ablauf eines EAs ist in Algorithmus 5 zusammengefasst.

Algorithmus 5: Evolutionärer Algorithmus

Definition:

$g \in \mathbb{N}$	Anzahl der Generationen
P_g	Population in Generation g

Eingabe:

I	Initialisierungsmethode
$\mu \in \mathbb{N}$	Größe der Population
$\lambda \in \mathbb{N}$	Größe der Nachkommenpopulation
E	Elternselektionsoperator
R	Rekombinationsoperator
$p_r \in [0,1]$	Rekombinationsrate
M	Mutationsoperator
$p_m \in [0,1]$	Mutationsrate
U	Umweltselektionsoperator
δ	Abbruchkriterium

Ausgabe:

Finale Lösungspopulation P_g

Initialisierung:

$g = 0$

Erzeuge Population P_0 der Größe μ durch I

Evolutionärer Algorithmus:

while *Abbruchkriterium δ nicht erfüllt* **do**
  for $j = 1 \ldots \lambda$ *Nachkommen* **do**
    Elternselektion durch E
    Elternrekombination mit Rekombinationsrate p_r durch R
    Mutation der Nachkommen mit Mutationsrate p_m durch M
  end
  Umweltselektion durch U: Auswahl der Individuen aus der Menge der Nachkommen- und/oder der Elternpopulation als neue Population P_{g+1}
  $g = g + 1$
end

4.1 Grundlagen und Prinzipien der Evolutionären Algorithmen

In diesem Abschnitt werden die allgemeinen Aspekte bezüglich Aufbau, Ablauf und Funktionsweise, die alle EAs gemein haben, beschrieben. Zunächst ist die Bestimmung der zu optimierenden Fitnessfunktionen (vgl. Abschnitt 4.1.1) und die Auswahl der passenden Repräsentation (vgl. Abschnitt 4.1.2) entscheidend für eine

	Genetischer Algorithmus	Evolutionsstrategien	Evolutionäre Programmierung	Genetische Programmierung
Typische Anwendung	Kombinatorische Optimierung	Stetige Optimierung	Optimierung	Modellierung
Typische Repräsentation	Zeichenketten über endliches Alphabet	Zeichenketten, Vektoren reeller Zahlen	Anwendungs-spezifisch	Bäume
Funktion der Rekombination	Primärer Variationsoperator	Wichtig, aber zweitrangig	Nicht angewandt	Primärer Variationsoperator
Funktion der Mutation	Sekundärer Variationsoperator	Wichtig, manchmal einziger Variationsoperator	Einziger Variationsoperator	Sekundärer Variationsoperator
Selektion der Eltern	Zufällig, Fitness-basiert	Zufällig, gleichverteilt	Jedes Individuum erzeugt einen Nachkommen	Zufällig, Fitness-basiert
Umweltselektion	Zufällig, Fitness-basiert	Deterministisch, Fitness-basiert	Zufällig, Fitness-basiert	Zufällig, Fitness-basiert
Autor	Holland (1975)	Rechenberg (1973), Schwefel (1977)	Fogel, Owens und Walsh (1966)	Koza (1992)

Tabelle 1: Hauptdialekte der Evolutionären Algorithmen (vgl. Eiben und Smith, 2003)

erfolgreiche Anwendung der EAs auf ein Optimierungsproblem. Nach der Initialisierung der ersten Population (vgl. Abschnitt 4.1.3) erfolgt unter Verwendung von Selektions- und Variationsoperatoren die Erstellung neuer Individuen (vgl. Abschnitte 4.1.4 bis 4.1.6). Ferner ist die Wahl der Kontrollparameter (vgl. Abschnitt 4.1.7) für den Erfolg eines EAs von großer Bedeutung.

4.1.1 Fitnessfunktionen

Mathematisch beschreibt eine Fitnessfunktion f die zu optimierende Funktion eines Optimierungsproblems (vgl. Defintion 4 in Kapitel 5), die den Abstand einer potentiellen Lösung von den gesetzten Zielen misst. Sie ordnet jeder potentiellen Lösung $t \in \mathbb{L}$, die durch jeweils ein Individuum dargestellt wird, einen Fitnesswert zu: $f : \mathbb{L} \to \mathbb{R}$. Dieser Fitnesswert beschreibt die Güte des Individuums, von dem das Überleben des Individuums in der Population und dessen Erfolg bei der Erstellung neuer Individuen abhängt.

Das Ziel der Fitnessfunktion ist somit die Bereitstellung eines aussagekräftigen, messbaren und vergleichbaren Wertes für jedes Individuum, um eine Ordnung der Individuen für die Selektion zu ermöglichen. Fitnessfunktionen sollten stets stetig sein und kleinere Verbesserungen eines Individuums sollten sich in kleineren Fitnessverbesserungen widerspiegeln, während starke Verbesserungen größere Änderungen im Fitnesswert mit sich bringen (vgl. Banzhaf et al., 1998). Eine gute Fitnessfunktion ist entscheidend für den Erfolg eines EAs (vgl. Koza, 1992). Mit schlechten

Fitnessfunktionen kann der Suchraum nicht erfolgreich abgesucht werden und der EA in lokalen Optima stagnieren (vgl. Fan et al., 2004).

Gängige Fitnessfunktionen zur Bewertung der Vorhersagegüte sind etwa die Fehlklassifikationsrate (vgl. Bhowan et al., 2012) oder die Sensitivität und Spezifität (vgl. Garcia-Nieto et al., 2009). Soll bei einem mehrkriteriellen Optimierungsproblem nicht eine aggregierende mehrkriterielle Optimierung (vgl. Abschnitt 5.1) durchgeführt sondern eine Pareto-Front (vgl. Abschnitt 5.2) ermittelt werden, wird für jedes Optimierungsziel eine eigene Fitnessfunktion erstellt und optimiert. Häufig verwendete Fitnessfunktionen in der mehrkriteriellen Optimierung sind Maßzahlen zur Vorhersagegüte, zur Modellkomplexität und zu Vorhersagekosten (vgl. Badran und Rockett, 2008; Reynolds und de la Iglesia, 2007; Muni et al., 2006; Oliveira et al., 2003).

4.1.2 Repräsentation

Mittels der Repräsentation stellt ein EA eine Verbindung zwischen der „realen Welt“ und der „Welt des EAs“ her. Dabei heißen Objekte, die Lösungen in der Umgebung des Ausgangsproblems darstellen, Phänotypen, ihre kodierten Gegenstücke heißen Genotypen, Chromosomen oder Individuen. Die Elemente eines Individuums heißen Variablen, Loki, Positionen, oder in biologischen Anwendungen auch Gene (vgl. Rothlauf, 2006). Demnach bildet die Repräsentation die Phänotypen auf die Menge der Genotypen ab. Die gesamte evolutionäre Suche findet im Genotypraum statt. Die Repräsentation sollte invertierbar sein, sodass für jeden Genotyp höchstens ein Phänotyp existiert, und sie sollte zur Implementierung der genetischen Operatoren passen.

Mögliche Repräsentationen der Chromosome können etwa Bitstrings, Zahlen, oder Matrizen sein. Die typischen Repräsentationen der verschiedenen Dialekte der EAs zeigt Tabelle 1. Rothlauf (2006) oder auch Jabeen und Baig (2010b) stellen zahlreiche Repräsentationsvarianten vor. Hier seien jedoch nur die beiden Ausprägungsformen erwähnt, die im weiteren Verlauf der Arbeit zur Anwendung kommen (vgl. Abschnitte 6.1 und 6.2). Diese sind Bitstrings und binäre Entscheidungsbäume. Insbesondere bei ganzzahligen Problemstellungen ist die häufigste Methode, ein Individuum durch einen Bitstring $\mathbb{B} = \{0,1\}^p$ der Länge p zu repräsentieren. In der GP werden unter der Vielzahl von Repräsentationsschemata am häufigsten Bäume verwendet (vgl. Banzhaf et al., 1998), die sich durch ihre nichtlineare und in ihrer Größe variable Struktur von den anderen Repräsentationsformen stark unterscheiden. Bei Individuen mit Baumstruktur stellen die internen Knoten Operatoren und Funktionen und die Blätter Klassen dar. Da die Individuen unter-

schiedlich groß sind, enthält nicht jedes Individuum alle Variablen, sodass mit der Entwicklung des Klassifikationsmodells eine implizite Variablenselektion als Teil des evolutionären Prozesses einhergeht. Aufgrund der großen Variabilität bei der Individuenrepräsentation besteht eine hohe Interpretationsgüte bei Verwendung der Baum-Repräsentation (vgl. Chaudhari et al., 2008).

4.1.3 Initialisierung

Die Initialisierung spielt eine wichtige Rolle für den Erfolg eines EAs. Eine schlechte initiale Population, zum Beispiel bestehend aus vielen, sehr ähnlichen Individuen, kann dazu führen, dass der Algorithmus in einem lokalen Minimum konvergiert (vgl. Jabeen und Baig, 2010b). Nichtsdestotrotz wird in den meisten EAs die Initialisierung einfach gehalten und die initiale Population aus zufälligen Individuen erstellt. Grundsätzlich können auch problemspezifische Heuristiken zur Erstellung einer initialen Population mit höheren Fitnesswerten genutzt werden, wobei der zusätzliche Rechenaufwand jedoch nicht für jedes Problem zu rechtfertigen ist (vgl. Eiben und Smith, 2003).

Im hier entwickelten NHEMOtree wird neben einer zufälligen initialen Population auch die Verwendung einer initialen Population, die durch einen *Wrapper*-Ansatz erstellt wurde, überprüft (vgl. Kapitel 6). Die beiden Ergebnisse werden in Kapitel 8 miteinander verglichen.

4.1.4 Elternselektion

Die Elternselektion beschreibt formal das Ziehen mit oder ohne Zurücklegen von Individuen aus der Gesamtpopulation, die zur Erstellung der Nachkommen verwendet werden. Die Selektion kann rein zufällig erfolgen oder die Individuen werden mit einer bestimmten Wahrscheinlichkeit basierend auf ihren Fitnesswerten ausgewählt. Die Fitness-basierte Selektion ist verantwortlich für die konkurrenzbetonte Populationsauswahl und findet sowohl während der Eltern- als auch während der Umweltselektion (vgl. Abschnitt 4.1.6) statt. Nachfolgend werden verschiedene Fitness-basierte Selektionsmechanismen vorgestellt. Diese sind stets unabhängig von der Problemrepräsentation.

Turnierselektion
Bei der Turnierselektion zur Auswahl eines Elternteils werden zufällig τ Individuen mit oder ohne Zurücklegen aus der Population gezogen und jenes der τ Individuen mit dem höchsten Fitnesswert wird mit Wahrscheinlichkeit p_τ als potentielles Elternteil ausgewählt. Dies wird μ-mal wiederholt bis μ potentielle Eltern gewählt sind.

Die Wahrscheinlichkeit, dass ein Individuum bei der Turnierselektion ausgewählt wird, hängt von vier Faktoren ab:

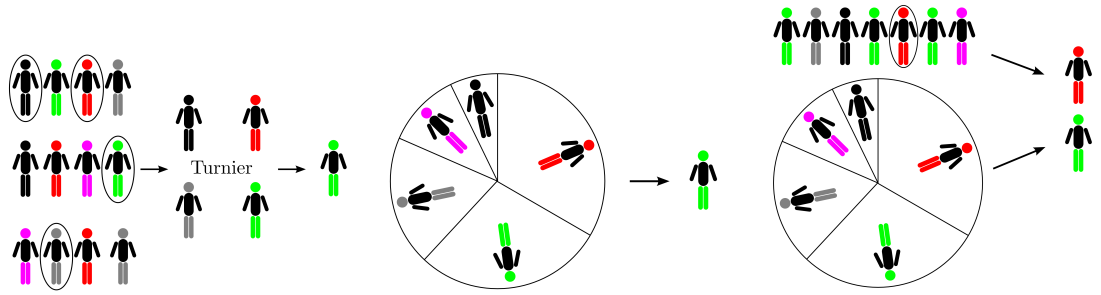
1. Rang des Individuums in der gesamten Population.
2. Turniergröße τ, d.h. Anzahl der Individuen, die pro Durchgang gegeneinander antreten.
3. Wahrscheinlichkeit p_τ, dass das Individuum mit dem höchsten Fitnesswert pro Turnier ausgewählt wird. In deterministischen Turnieren ist $p_\tau = 1$, sodass stets das fitteste Individuum ausgewählt wird. Aber auch stochastische Versionen sind möglich mit $p_\tau < 1$. In diesem Fall ist der Selektionsdruck reduziert.
4. Die Tatsache, ob Individuen mit oder ohne Zurücklegen ausgewählt werden. Im letzteren Fall fungieren die schwächsten $\tau - 1$ Individuen nie als Elternteil. Bei der Turnierselektion mit Zurücklegen können bis auf das schwächste Individuum alle ausgewählt werden.

Aufgrund der einfachen Kontrolle des Selektionsdrucks durch Veränderung der Turniergröße τ ist die Turnierselektion in EAs weit verbreitet (vgl. etwa Bhowan et al., 2012; Jabeen und Baig, 2010b; Alba et al., 2007; Zhao, 2007; Eiben und Smith, 2003; Emmanouilidis et al., 2000; Banzhaf et al., 1998). Abbildung 2(a) stellt schematisch die Turnierselektion mit $\tau = 4$ dar.

Roulette-Selektion

Bei der Roulette-Selektion (vgl. Abbildung 2(b)) beträgt die Wahrscheinlichkeit, dass ein Individuum t als Elternteil ausgewählt wird $f_t / \sum_{t=1}^{\mu} f_t$, wobei f_t die Fitness des t-ten Individuums und μ die Anzahl der Individuen in der Population beschreibt. Sollen μ Eltern ausgewählt werden, wird der Vorgang μ-mal wiederholt. Obwohl Individuen mit höherer Fitness mit größerer Wahrscheinlichkeit als Elternteil ausgewählt werden, besteht dennoch für Individuen mit geringer Fitness eine Auswahlchance. Dies ist vorteilhaft, da auch schwache Individuen nützliche Komponenten für den Rekombinationsprozess besitzen können.

Problematisch bei der Roulette-Selektion ist jedoch die schnelle Übernahme der gesamten Population durch sehr fitte Individuen, sodass es zu einer vorzeitigen Konvergenz kommt. Bei sehr ähnlichen Fitnesswerten liegt auf der anderen Seite kaum ein Selektionsdruck vor, sodass die Selektion nahezu gleichverteilt ist. Unter solchen Umständen steigert sich die Güte der Population nur sehr langsam (vgl. Coello Coello et al., 2007).



(a) Turnierselektion (b) Roulette-Selektion (c) Winkler-Selektion

Abbildung 2: Elternselektionsverfahren in Evolutionären Algorithmen

Selektion nach Winkler

Bei der Elternselektion nach Winkler (Winkler-Selektion) wird die Hälfte der Eltern zufällig und die andere Hälfte per Roulette-Selektion ausgewählt (vgl. Abbildung 2(c)). Laut Winkler et al. (2009) ist diese Art der Elternselektion besonders bei EAs mit Baum-Repräsentation geeignet.

Im NHEMOtree werden die Eltern für die Entwicklung der nächsten Population per Turnierselektion mit Zurücklegen oder per Winkler-Selektion ausgewählt. Die Ergebnisse bei Verwendung der beiden Elternselektionsverfahren werden in Abschnitt 8.3 miteinander verglichen.

4.1.5 Variationsoperatoren

Durch die Variationsoperatoren werden aus bestehenden Individuen veränderte Individuen erstellt, die neue, potentielle Lösungskandidaten repräsentieren. Im Allgemeinen unterscheiden sich die Variationsoperatoren in der Anzahl teilnehmender Individuen.

Rekombination

Die Rekombination wird auf zwei oder mehr ausgewählten Individuen, den Eltern, angewandt und produziert durch Kombination der verschiedenen Genotypen ein oder mehrere Nachkommen. Dies geschieht stochastisch, da die Auswahl der Eltern und die Art der Kombination zufällig erfolgt. Durch die Paarung zweier Individuen mit verschiedenen, aber wünschenswerten Merkmalen können Nachkommen produziert werden, die die Merkmale beider Eltern vereinigen. EAs produzieren eine Vielzahl von Nachkommen durch zufällige Rekombination in der Hoffnung, dass nur wenige Nachkommen schlechtere Eigenschaften als ihre Eltern und viele Nachkommen verbesserte Eigenschaften besitzen.

Mutation
Die Mutation wird hingegen nur auf einem Individuum vollzogen und resultiert in einem neuen Individuum. Im Allgemeinen erfolgt mit Hilfe der Mutation die Exploration des Suchraums. Üblicherweise wird die Mutation nach der Rekombination durchgeführt und sollte nur mit geringer Wahrscheinlichkeit, der sogenannten Mutationsrate p_m, stattfinden.

Die Art der Variationsoperatoren sowie die Wichtigkeit für die Nachkommenerstellung hängen sehr stark von dem Dialekt und der Repräsentation der Individuen ab (vgl. Tabelle 1). Die gängigsten Variationsoperatoren für Individuen mit Bitstring- und mit Baum-Repräsentation werden in den Abschnitten 4.2 und 4.3 vorgestellt.

4.1.6 Umweltselektion

Die Umweltselektion reduziert die Menge der μ Eltern und λ Nachkommen auf μ Individuen der neuen Generation basierend auf dem Alter oder der Fitness der Individuen. Bei der Alter-basierten Umweltselektion ist die Anzahl der Generationen, die ein Individuum bereits in der Population besteht, ausschlaggebend für dessen Überleben. Jedes Individuum verbleibt die gleiche Generationenanzahl in der Population, sodass die Fitness nur für das Eintreten in die Population entscheidend ist (vgl. Eiben und Smith, 2003).

Eine Vielzahl von Strategien existieren zur Fitness-basierten Auswahl der neuen μ Individuen. Bei der $(\mu + \lambda)$-Selektion (Plus-Selektion) treten die μ Eltern und λ Nachkommen gegeneinander an und die fittesten Individuen, d.h. die Individuen mit den höchsten Fitnesswerten, bilden die neue Population (vgl. Beyer und Schwefel, 2002). Da fittere Individuen viele Generationen überleben können, handelt es sich hierbei um eine elitäre Selektion. Elitärismus wird häufig in Verbindung mit Fitness-basierten Strategien verwendet, um den Verlust der aktuell fittesten Individuen zu vermeiden. Elitärismus ist vor allem ein wichtiger Faktor zur Verbesserung einer evolutionären mehrkriteriellen Suche (vgl. Zitzler et al., 2000), beschleunigt die Suche in Richtung der Pareto-Front und verhindert eine vorzeitige Konvergenz.

Bei der (μ, λ)-Selektion (Komma-Selektion) werden hingegen nur aus den λ Nachkommen die μ besten selektiert und die Eltern ignoriert, sodass bei dieser Selektionsform keine unsterblichen Individuen existieren. Jedes Individuum lebt nur für die Dauer einer Generation (vgl. Beyer und Schwefel, 2002).

Ferner wird zwischen generationsbasierten und *Steady-State*-Algorithmen unterschieden. Generationsbasierte Algorithmen bilden in jeder Generation eine vollständig neue Population. Wird nur ein Teil der Population pro Generation erstellt, handelt es sich um *Steady-State*-Algorithmen (vgl. Beyer und Schwefel, 2002) Die elitärere

Steady-State-Strategie wirkt der zerstörerischen Wirkung von hohen Rekombinations- und Mutationsraten entgegen. Bei der häufigsten Form der *Steady-State*-Strategie wird pro Generation lediglich ein Individuum ersetzt.

Im *Wrapper*-Ansatz sowie in NHEMOtree wird ausschließlich die Plus-Selektion benutzt. Allerdings werden die Ergebnisse verschiedener generationsbasierter und *Steady-State*-Umweltselektionsverfahren miteinander verglichen (vgl. Kapitel 8).

4.1.7 Kontrollparameter

Die richtige Wahl der Kontrollparameter trägt wie bei anderen Methoden wesentlich zu einer guten Leistung der EAs bei und hängt vor allem von den zugrunde liegenden Daten ab. Eine beste Parametereinstellung für alle Probleme gibt es auch für EAs nach dem No-Free-Lunch-Theorem nicht (vgl. Wolpert und Macready, 1997). Wichtige Kontrollparameter der EAs sind die Populationsgröße $\mu \in \mathbb{N}$, die Nachkommenanzahl $\lambda \in \mathbb{N}$, sowie die Rekombinations- und Mutationsraten $p_r, p_m \in [0,1]$.

Laut De Jong (2007) beschreibt die Populationsgröße μ den Grad der parallelen Suche in EAs. Sie beeinflusst stark dessen Konvergenzvermögen gegen die Pareto-Front (vgl. Zitzler et al., 2000). Sehr kleine Populationen stellen nicht genug Vielfalt unter den Individuen bereit und verhindern eine gute Abdeckung der Pareto-Front. Eine Vergrößerung der Population führt jedoch nicht automatisch zu verbesserten Ergebnissen. Das Gleiche gilt für die Anzahl simulierter Generationen $g \in \mathbb{N}$. Auf der anderen Seite verlängert sich bei einer größeren Population die Suchdauer, um im Suchraum in der optimalen Region zu konvergieren.

Vor allem bei komplexen Suchräumen mit vielen lokalen Optima ist eine große Population nötig, um mit einer realistischen Chance das globale Optimum zu finden. Im Laufe der Berechnung beschränkt sich der EA auf einen kleinen Teil des gesamten Suchraums, sodass eine Verringerung der Populationsgröße mit steigender Generationenanzahl g sinnvoll erscheint. Jedoch stellt sich die Frage, wie der Zeitpunkt und die Stärke der Populationsgrößenreduzierung a priori festgelegt werden soll. Eine dynamische Anpassung der Populationsgröße wäre geeignet. Sie ist jedoch in der Praxis wegen interagierender Faktoren, wie der Nachkommenanzahl oder des Selektionsdrucks, nur schwierig realisierbar, sodass eine feste Populationsgröße über den gesamten Analysezeitraum in der Praxis am verbreitesten ist (vgl. De Jong, 2007). Die Anzahl der Nachkommen λ beschreibt die Explorationsstärke je Elternpopulation. Analog zu den Ausführungen zur Populationsgröße der Eltern ist eine dynamische Nachkommenanzahl in der Praxis nicht etabliert.

Höhere Rekombinationsraten p_r steigern sowohl die Kombination von guten *Building Block*s (vgl. Abschnitt 4.2) als auch die Zerstörung guter Lösungen. Zu hohe Mutationsraten p_m machen aus der evolutionären eine zufällige Suche. Jedoch können durch die Mutation Bereiche des Lösungsraums, die nicht in der aktuellen Population vertreten sind und durch Rekombination nicht erreicht werden können, wieder in die Population integriert werden. Wird auf der anderen Seite eine zu schwache Variation gewählt, kann kein echter Fortschritt erzielt werden. Dies erhöht die Wahrscheinlichkeit, dass der EA vorzeitig in einem lokalen Optimum konvergiert.

Da der Lauf eines EAs ein dynamischer adaptiver Prozess ist, scheinen a priori festgelegte, statische Parameterwerte dem eigentlichen Geist zu widersprechen (vgl. Eiben et al., 1999). Die Abhängigkeiten zwischen den einzelnen Parametern führen ebenfalls zu Schwierigkeiten bei der Parameterfestlegung. Das systematische Testen aller Parameterkombinationen ist praktisch nicht möglich und auch nach aufwändigem Ausprobieren müssen die ausgewählten Parameter für das entsprechende Problem nicht optimal sein. Außerdem erscheinen verschiedene Parameterwerte zu verschiedenen Stadien eines evolutionären Prozesses sinnvoll. So sind etwa große Mutationsraten zu Beginn des Prozess hilfreich, um den Suchraum schneller zu erforschen. Später im Prozess können geringere Mutationsraten die Feinabstimmung der Individuen vollziehen.

Dynamische Parameter verändern sich dabei über die Generationen hinweg. Dabei kann zum einen der Fortschritt der Generationen zur Veränderung der Parameter führen oder aber es wird eine Rückmeldung des Suchprozesses in die Parameteranpassung integriert, wie etwa bei der 1/5-Erfolgsregel in den Evolutionsstrategien (vgl. Abschnitt 4.3 und Rechenberg, 1973). Eine weitere Variante ist die selbstadaptive Parameterkontrolle, bei der die anzupassenden Parameter im Chromosom kodiert sind und sich ebenfalls der Rekombination und Mutation unterziehen.

Allerdings ist die praktische Umsetzung flexibler Parameter schwierig. Hierzu müssen Kriterien bestehen, aufgrund derer Parameter verändert werden. Außerdem ist bisher die Verbesserung des Algorithmus durch eine dynamische Parameteranpassung nicht belegt (vgl. De Jong, 2007). Eine weitere Möglichkeit ist ein zweistufiger EA, bei dem die erste Stufe die Parameterwerte entwickelt, die für den eigentlichen EA verwendet werden. Jedoch bleibt auch dort die Frage, wie die Parameter des äußeren EAs gewählt werden sollten. Ferner wird durch diese zweifache Optimierung die Berechnungsdauer stark verlängert.

In dieser Arbeit werden für den medizinischen Anwendungsfall verschiedenen Parameterkombinationen durch ein Latin-Hypercube-Design (vgl. Abschnitt A im Anhang) erstellt und die Ergebnisse miteinander verglichen (vgl. Kapitel 8).

4.2 Evolutionäre Algorithmen mit Bitstring-Repräsentation

Die typische Repräsentation der Genetischen Algorithmen (GAs) sind Bitstrings $\mathbb{B} = \{0,1\}^p$ der Länge p, wobei jedes der p Bits $\in \{0,1\}$ eine der betrachteten Einflussvariablen X_i, $i = 1, \ldots, p$, darstellt. In dieser Arbeit werden EAs mit einer solchen Repräsentation als GAs bezeichnet, auch wenn GAs mit anderen Repräsentationsformen existieren (vgl. Tabelle 1). Positionen, die im Bitstring mit 1 codiert sind, zeigen an, dass die entsprechende Variable in der Lösung enthalten ist. Mit 0 codierte Variablen sind hingegen nicht enthalten. Bei $p = 10$ Einflussvariablen X_i ergibt sich für ein Individuum bestehend aus X_2, X_3, X_5 und X_{10} der Bitstring $\mathbb{B} = 0110100001$.

Im Folgenden werden zunächst Anwendungsmöglichkeiten und theoretische Aspekte der GAs beschrieben. Das gängigste Initialisierungsverfahren sowie die speziell auf die Repräsentationen mit Bitstrings zugeschnittene Variationsoperatoren werden erläutert und Vorschläge für die Wahl der Kontrollparameter zusammengetragen.

Anwendung

Durch Bitstrings können leicht Variablenteilmengen dargestellt werden, sodass GAs einen natürlich Ansatz zur Variablenselektion darstellen.Üblicherweise agieren GAs als *Wrapper* (vgl. Abschnitt 3.2), wobei die Vorhersage der ausgewählten Variablenteilmengen vom internen Klassifikationsalgorithmus bewertet wird (vgl. Kohavi und John, 1997). Durch diese reduzierte Variablenverfügbarkeit kann der interne Klassifikationsalgorithmus nun auch Variablen beurteilen, die zuvor von einer dominanten Variablen „unterdrückt“ wurden. Die Qualität der Lösungen wird unabhängig von der Repräsentation mittels der Fitnessfunktionen gemessen, die durch den GA optimiert werden müssen.

Verschiedene Klassifikationsalgorithmen wie etwa Entscheidungsbäume, Neuronale Netze, k-Nächste-Nachbarn, Suport Vector Machines oder Random Forests wurden bereits in GAs zur Evaluation der Individuen eingesetzt (vgl. Cherkauer und Shavlik, 1996; Siedlecki und Sklansky, 1989; Yang, 1998; Li et al., 2001; Alba et al., 2007; Gray und Fan, 2008).

Theoretische Betrachtung

Trotz ihrer erfolgreichen Anwendungen in vielen Wissenschaftsbereichen zur Lösung von Optimierungs- und Suchproblemen existiert bisher keine vollständige Theorie für die Funktionsfähigkeit der GAs. In diesem Abschnitt werden der Vollständigkeit halber einige theoretische Erklärungsversuche vorgestellt.

Die zentrale Frage bezüglich der robusten Funktionsfähigkeit der GAs und des häufigen Erfolgs bei der Generierung von Lösungen mit hoher Fitness behandelt Holland (1975) in seinem Schema-Theorem für GAs. In GAs beschreibt ein Schema einen Bitstring mit den Symbolen 0, 1 und einem Platzhalter #, sodass ein Schema mehrere Bitstrings repräsentiert. Das Schema #10#1 repräsentiert etwa die Bitstrings 01001, 01011, 11001, 11011. Der Abstand zwischen der ersten und letzten festen Bitstringposition heißt definierende Länge des Schemas, die Anzahl der festen Positionen heißt Ordnung. Besondere Schemata mit kurzer definierter Länge, wenigen definierten Positionen und überdurchschnittlicher Fitness heißen *Building Block*s (vgl. Koza, 1992). Laut Hollands Schema-Theorem verändert sich bei einer Fitness-proportionalen Selektion sukzessive die Wahrscheinlichkeit für das Auftreten einiger Schemata in der Population, um das allgemeine Fitnessoptimum zu erreichen.

Theorem *Schema-Theorem*
Kurze Schemata mit geringer Ordnung und überdurchschnittlicher Fitness erhalten exponentiell steigende Anzahl von Betrachtungen in aufeinanderfolgenden Generationen eines GA.

Nach dem Schema-Theorem verändert sich die Anzahl der Bitstrings θ in der Population, die einem Schema S zum Zeitpunkt g entsprechen, nach folgender Wachstumsgleichung

$$\theta(S,g+1) = \theta(S,g) \times \frac{eval(S,g)}{\overline{F(g)}}. \tag{3}$$

Die Fitness des Schemas S wird durch $eval(S,g) = \sum_{j=1}^{\mu} eval(\mathbb{B}_j/\mu)$ mit μ Bitstrings $\{\mathbb{B}_1, \ldots, \mathbb{B}_\mu\}$ in der Population, die dem Schema S zum Zeitpunkt g entsprechen, und der durchschnittlichen Fitness der Population zum Zeitpunkt g, $\overline{F(g)} = \frac{F(g)}{\mu}$, beschrieben. Gleichung (3) zeigt, dass die Selektion eines überdurchschnittlichen Schemas S erhöht ist und diese Auswahlwahrscheinlichkeit exponentiell mit laufender Zeit steigt (vgl. Michalewicz, 1992). Der Langzeiteffekt eines Schemas unter der Annahme, dass S stets mit ϵ% über dem Durchschnitt liegt und somit $eval(S,g) = \overline{F(g)} + \epsilon \times \overline{F(g)}$ gilt, kann beschrieben werden als

$$\begin{aligned}\theta(S,g+1) = \theta(S,g) \times \frac{eval(S,g)}{\overline{F(g)}} &= \theta(S,g) \times \frac{(\overline{F(g)} + \epsilon \times \overline{F(g)})}{\overline{F(g)}} \\ &= \theta(S,g) \times (1+\epsilon) = \ldots = \theta(S,0) \times (1+\epsilon)^{g+1}.\end{aligned}$$

Mit Hilfe der Variationsoperatoren werden neue Schemata eingeführt. Der Rekombinationsoperator ermöglicht einen zufälligen Informationsaustausch und durch den

Mutationsoperator wird eine größere Variabilität eingeführt. Michalewicz (1992) zeigt, dass der kombinierte störende Effekt dieser Operatoren auf das Schema nicht signifikant ist, wenn das Schema eine kurze definierende Länge besitzt und von geringer Ordnung ist.

Das Schema-Theorem beschreibt also die Dynamik der GAs und macht Annahmen, wie sich Schemata von Generation zu Generation weiterentwickeln, während Selektions- und Variationsoperatoren auf sie einwirken. Ein direktes Resultat des Schema-Theorems ist die *Building Block*-Hypothese. Diese besagt, dass GAs den Suchraum durch Schemata mit kurzer definierender Länge und niedriger Ordnung, sogenannte *Building Block*s, erkunden, die die nützlichen Eigenschaften eines Individuums zusammenfassen und zum Informationsaustausch während der Rekombination verwendet werden. So können aus partiellen Lösungen der *Building Block*s neue Bitstrings mit möglicherweise höherer Fitness entstehen.

Das Schema-Theorem und die *Building Block*-Hypothese wurde jedoch von vielen Autoren kritisiert, da sie weder die Funktionsweise der GAs als Optimierer erklären noch das Verhalten der GAs über mehrere Generationen hinweg gut vorhersagen können (vgl. Langdon und Poli, 2002; Beyer, 1997; Grefenstette, 1993; Michalewicz, 1992). Ferner gilt Gleichung (3) nur für unendliche Populationen, was jedoch in der Praxis nicht möglich ist.

Grefenstette (1993) etwa stellt klar, dass die *Building Block*-Hypothese nicht das tatsächliche Verhalten der GAs während der evolutionären Entwicklung beschreibt, jedoch eine grobe, aber zweckdienliche und deshalb häufig verwendete Erklärung der Funktionsweise von GAs ist. Andere Autoren messen dem Schema-Theorem für praktische Anwendungen mit endlichen Populationsgrößen in der herkömmlichen Interpretation nur eine geringe Bedeutung zu (vgl. Michalewicz, 1992).

Laut Radcliffe (1997) ist allerdings nicht das Schema-Theorem als solches, sondern dessen Überinterpretation das eigentliche Problem. Seiner Meinung nach ist das Schema-Theorem ein Ergebnis, das für viele GAs nachweisbar ist.

Beyer (1997) generiert aus der Theorie der Evolutionsstrategien eine alternative Hypothese für die Funktionsweise der GAs. Er überträgt das Prinzip des evolutionären Fortschritts, die mutationsinduzierte Artenbildung durch Rekombination, bei der durch das Zusammenspiel von Mutation und Rekombination eine verbesserte Population erstellt wird, sowie die Hypothese der genetischen Reparatur aus den Evolutionsstrategien auf die GAs. Das beherrschende Prinzip sei dabei das des evolutionären Fortschritts als Ergebnis der gegensätzlichen Fortschrittszu- und -abnahme.

Unbestritten hängt die neue Population eines GAs nur von dem Status der aktuellen Population ab. Diese Tatsache ist bekannt als Markov-Eigenschaft und zeigt, dass

Markovprozesse angemessene Modelle für das probabilistische Verhalten von GAs sind. So konnte in den 1990er Jahren mittels der Markovmodelle eine generelle Theorie entwickelt werden (vgl. Nix und Vose, 1992). Theoretische Resultate bezüglich der Konvergenzeigenschaften von GAs und ihrem dynamischen Verhalten ermöglichten deren direkte Simulation. Jedoch gelingt dies nur für Probleme mit ausreichend kleiner Dimension, da die assoziierten Übergangsmatrizen sehr groß sind (vgl. De Jong et al., 1995; Rudolph, 1998).

Weitere Ansätze für theoretische Erklärungen zur Funktionsweise von GAs tragen etwa Eiben und Rudolph (1999) zusammen. Außerdem kann das Verhalten von GAs bei bekannter Übergangsmatrix des zugehörigen Markovprozesses numerisch betrachtet werden (vgl. etwa De Jong et al., 1995).

Initialisierung

Beim klassischen GA mit Bitstring-Repräsentation erfolgt die Initialisierung der Ausgangspopulation zufällig. Die Initialisierung der Bits der Individuen in der Ausgangspopulation stellt ein Bernoulli-Experiment mit Erfolgswahrscheinlichkeit $\pi = P(\text{Bit} = 1)$ dar, sodass mit gleicher Wahrscheinlichkeit die einzelnen Bits des Bitstrings an- oder ausgeschaltet werden. Neben der am häufigsten verwendeten zufälligen Initialisierung des GAs werden die Individuen gelegentlich auch in Bereichen initialisiert, in denen optimale Lösungen vermutet werden (vgl. Wakabi-Waiswa und Baryamureeba, 2011; Freitas, 2002) oder so, dass die Individuen vorgegebene Kriterien erfüllen (vgl. Biles, 1994).

Variationsoperatoren

In diesem Abschnitt werden die Variationsoperatoren zur Rekombination und Mutation von Individuen mit Bitstring-Repräsentation vorgestellt.

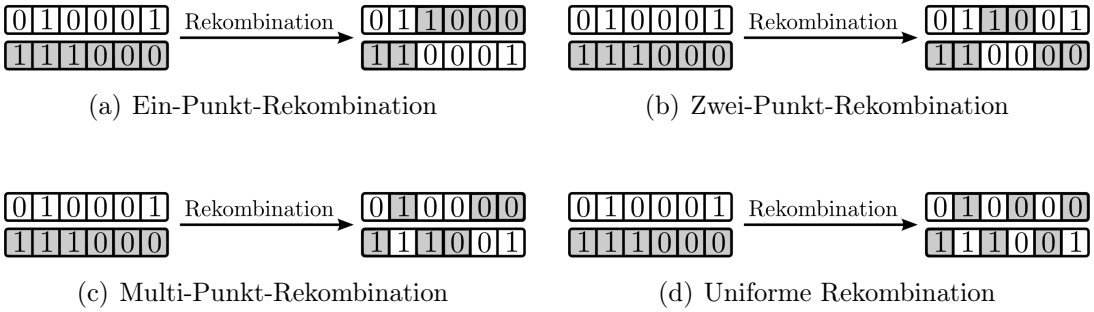
Rekombination

In GAs stellt die Rekombination den hauptsächlichen Variationsoperator dar, bei dem laut Srinivas und Patnaik (1994) traditionell bei zwei Eltern Rekombinationspunkte an ein oder zwei Stellen gesetzt werden. In der Ein-Punkt-Rekombination wird je Elternteil jeweils ein Rekombinationspunkt zufällig ausgewählt und die so erhaltenen Segmente der Bitstrings untereinander getauscht. Analog erfolgt die Rekombination im Zwei-Punkt-Rekombinationsschema (vgl. Abbildung 3(a) und 3(b)). Eine Erweiterung ist die Multi-Punkt-Rekombination, die jedes Bitstring in $k < p$ Segmente teilt, wobei die Abschnitte der Bits abwechselnd zwischen den zu rekombinierenden Bitstrings ausgetauscht werden (vgl. Abbildung 3(c)). Die unifor-

me Rekombination tauscht hingegen Bits anstelle von Segmenten zwischen den Bitstrings. So werden an jeder Position des Bitstrings die Bits mit einer festen Wahrscheinlichkeit zwischen den Eltern ausgetauscht (vgl. Abbildung 3(d)).

Die uniforme Rekombination besitzt eine stärkere, zerstörendere Natur als die anderen genannten Rekombinations-Schemata. Bei Verwendung von Ein- und Zwei-Punkt-Rekombinations-Operatoren werden *Building Block*s eher beibehalten (vgl. Srinivas und Patnaik, 1994), jedoch geht bei homogeneren Populationen dadurch der forschende Charakter des Algorithmus verloren. Laut Srinivas und Patnaik (1994) ist ein gutes Zusammenspiel von Populationsgröße und Rekombinationsart für den Erfolg eines GAs ausschlaggebend. Es existiert empirische Evidenz, dass die uniforme Rekombination besser für kleine Populationen geeignet ist und die Ein- bzw. Zwei-Punkt-Rekombination besser für große Populationen. Die uniforme Rekombination hilft, eine stark explorative Suche in einer kleinen Population aufrecht zu erhalten, während die Vielfältigkeit in großen Populationen die Notwendigkeit auf Erkundung reduziert und eine Punkt-Rekombination passender ist.

Die Nachkommen der GAs beinhalten Informationen von beiden Elternteilen. Die Ein-Punkt-Rekombination transferiert durchschnittlich 50% des genetischen Materials beider Eltern. Bei sehr verschiedenen Elternteilen sind die Nachkommen im genotypischen Raum weit von diesen entfernt. Dennoch ist etwa durch die zufällige Bevorzugung des genetischen Materials eines Elternteils oder bei sehr ähnlichen Eltern auch die Erstellung von Nachkommen möglich, die sehr ähnlich zu ihren Eltern sind. Somit besitzen die Rekombinationsoperatoren der GAs sowohl Eigenschaften von globalen als auch von lokalen Suchoperatoren, sodass die globale Suche zur Exploration des Suchraumes führt und ein Entkommen aus lokalen Optima ermöglicht.



(a) Ein-Punkt-Rekombination (b) Zwei-Punkt-Rekombination

(c) Multi-Punkt-Rekombination (d) Uniforme Rekombination

Abbildung 3: Rekombinationsoperatoren für Evolutionäre Algorithmen mit Bitstring-Repräsentation

Rekombinationsoperatoren mit mehr als zwei Eltern sind ebenfalls denkbar und einfach zu implementieren, haben jedoch keinen biologischen Hintergrund. In dieser Ar-

beit wird im *Wrapper*-Ansatz wegen der einfachen Vergleichbarkeit zum NHEMOtree die Ein-Punkt-Rekombination von jeweils zwei Eltern verwendet (vgl. Abschnitt 6.1).

Mutation

Der am häufigsten verwendete Mutationsoperator bei GAs ist die sogenannte Bit-Flip-Mutation, die jedes Bit separat mit einer geringen Mutationsrate p_m umdreht (vgl. Abbildung 4). Die Anzahl der Werte, die durch die Mutation verändert werden, ist somit zufällig (vgl. Eiben und Smith, 2003).

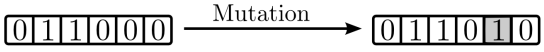


Abbildung 4: Bit-Flip-Mutationsoperator für Evolutionäre Algorithmen mit Bitstring-Repräsentation

Wahl der Kontrollparameter

Trotz vielfältiger Möglichkeiten zur Wahl der Kontrollparameter und der dynamischen Parameterkontrolle trugen Eiben et al. (1999) für die Rekombinationsrate p_r in GAs Werte zwischen 0,60 und 0,95 aus der Literatur zusammen. Vorschläge für die Mutationsrate p_m machten etwa De Jong (2007), Grefenstette (1986) und Schaffer (1985). In Vergleichsstudien schnitt dabei $p_m = \frac{1}{p}$, mit p als Länge des Bitstrings am besten ab.

Dieses Wissen über frühere, erfolgreiche Parametereinstellungen in GAs fließt in die Erstellung des Maximin-Latin-Hypercube-Design-Versuchsplans (vgl. Abschnitt A im Anhang) für die Parametereinstellungen des *Wrapper*-Ansatzes ein (vgl. Tabelle 6 in Abschnitt 8.2).

4.3 Evolutionäre Algorithmen mit Baum-Repräsentation

Neben den GAs kommen in dieser Arbeit EAs mit Baum-Repräsentation zum Einsatz. Die Strukturen, die einer Variation ausgesetzt sind, sind dabei keine Bitstrings sondern Bäume, sodass ein flexibler Suchraum zur Sondierung der besten Klassifikationsregel entsteht. Die Beschaffenheit dieser Individuen gibt ihnen die Möglichkeit, nicht nur Wissen darzustellen und eine Variablenselektion durchzuführen, sondern auch numerische Informationen an den inneren Knoten zu verarbeiten und in Boolesche Antworten zu transformieren. Als vergleichender Operator wird in dieser Arbeit $\{\leq, >\}$ herangezogen, um die Daten zu teilen. Somit kann die GP aufgrund ihrer Flexibilität, die eine Adaption an spezielle Probleme erlaubt, in nahezu allen Klassifikationsanwendungen zum Einsatz kommen, ohne einen internen Klassifikationsalgorithmus zur Variablenbewertung zu benötigen.

Die Baum-Repräsentation der Individuen erfordert eine spezielle Fitnessbewertung, die im Folgenden vorgestellt wird. Analog zur Beschreibung der GAs in Abschnitt 4.2 erfolgt ebenfalls die Beschreibung der theoretischen Aspekte von EAs mit Baum-Repräsentation. Danach wird das für EAs und andere Algorithmen mit flexibler Individuengröße spezifische Problem des Individuumwachstums ohne Fitnesssteigerung, das sogenannte *Bloating*, erläutert und Lösungsansätze diskutiert. Spezifische Initialisierungsverfahren und verschiedene Rekombinations- und Mutationsoperatoren für EAs mit Baum-Repräsentation werden ferner beschrieben. Da die Selektion lediglich auf Fitnessinformationen beruht und somit unabhängig von der Repräsentation ist, können die bereits vorgestellten Selektionsverfahren aus den Abschnitten 4.1.4 und 4.1.6 verwendet werden. Vorschläge für die Wahl der Kontrollparameter sind am Ende des Abschnittes zusammengetragen.

Fitnessbewertung und Anwendung

In den Entscheidungsbäumen der EAs wird analog zu CART einem Blatt durch eine einfache Mehrheitswahl die Klasse zugeteilt, die die meisten Beobachtungen in dem entsprechenden Blatt innehaben. Somit können EAs mit Baum-Repräsentation direkt auf Klassifikationsprobleme angewendet werden (vgl. Bot und Langdon, 2000). Die Fehlklassifikationsrate wird demnach berechnet als

$$1 - \frac{1}{n} \sum_{1=l}^{L} \max_{k} \{\nu_{lk}\}, \tag{4}$$

wobei ν_{lk} die Anzahl der Beobachtungen im Blatt $l \in \{1, \ldots, L\}$ mit Klassenzugehörigkeit $k \in \{1, \ldots, K\}$ beschreibt und $n = \sum_l \sum_k \nu_{lk}$ die Anzahl aller Beobachtungen.

Einen allgemeinen Überblick über Anwendungsmöglichkeiten der EAs mit Baum-Repräsentation geben beispielsweise Banzhaf et al. (1998), Espejo et al. (2010), Jabeen und Baig (2010b) oder Khan und Alam (2012). Frühere Arbeiten zur Entwicklung von Entscheidungsbäumen durch EAs mit Baum-Repräsentation stammen etwa von Winkler et al. (2009), Kumar et al. (2009), Gray und Fan (2008), Reynolds und de la Iglesia (2007), Muni et al. (2006), Papagelis und Kalles (2001), Bot und Langdon (2000) und Eggermont et al. (1999). Die Erstellung von Ensemble (vgl. unter anderem Hong und Cho, 2006) sowie Kombinationen von EAs mit Baum-Repräsentation und Klassifikationsalgorithmen existieren ebenfalls. Sherrah et al. (1996) verwenden dazu etwa ein generalisiertes lineares Modell oder Jabeen und Baig (2010a) einen Partikelschwarmansatz.

Allerdings wird in vielen Publikationen nicht das Problem des *Bloatings* (vgl. Seite 42) diskutiert und keine Gegenmaßnahmen in die Algorithmen integriert (vgl. Winkler et al., 2009; Gray und Fan, 2008; Muni et al., 2006; Eggermont et al., 1999; Papagelis und Kalles, 2001; Kumar et al., 2009; Hong und Cho, 2006; Sherrah et al., 1996). Ferner werden die verwendeten Selektions- und Variationsoperatoren nicht immer vorgestellt (vgl. Bot und Langdon, 2000; Jabeen und Baig, 2010a; Hong und Cho, 2006) oder nur Zwei-Klassen-Probleme sind lösbar (vgl. Reynolds und de la Iglesia, 2007; Jabeen und Baig, 2010a; Hong und Cho, 2006; Eggermont et al., 1999). Ein weiteres Problem bei den genannten Publikationen ist die Fokussierung auf nur einen oder wenige Rekombinations- und Mutationsoperatoren ohne Diskussion der anderen vielfältigen Möglichkeiten, die insbesondere bei EAs mit Baum-Repräsentation existieren (vgl. Seite 45).

Bei Anwendung der EAs mit Baum-Repräsentation auf mehrkriterielle Optimierungsprobleme (vgl. Kapitel 5) wird häufig die Fehlklassifikationsrate und die Baumgröße minimiert (vgl. Haruyama und Zhao, 2002; Badran und Rockett, 2008), aber auch die Optimierung anderer Zielfunktionen ist möglich (vgl. Zhao, 2007; Khoshgoftaar und Liu, 2007; Mugambi und Hunter, 2003). Haruyama und Zhao (2002) schlagen insgesamt vier verschiedene mehrkriterielle Optimierungsmethoden vor, wobei stets die Vorhersagegüte eine höhere Priorität als die Baumgröße für die Selektion der Individuen darstellt. Badran und Rockett (2008) betrachten die beiden Fitnessfunktionen, Knotenanzahl und Fehlklassifikationsrate bei der Klassifikation auf Basis kategorialer Einflussvariablen. Allerdings messen sie die finale Ergebnisgüte nur anhand der Fehlklassifikationsrate und erwähnen die Baumgröße bei ihrem Vergleich nicht. Daher ist fragwürdig, ob eine mehrkriterielle Optimierung in der Anwendung von Badran und Rockett (2008) tatsächlich sinnvoll ist. Zhao (2007) verwendet die mehrkriterielle GP zur gleichzeitigen Optimierung von verschiedenen Darstellungen der Vorhersagegüte, wie etwa Sensitivität und Spezifität. Andere, von der Vorhersagegüte unabhängige Zielfunktionen können mit dieser Methode jedoch nicht optimiert werden. Khoshgoftaar und Liu (2007) schlagen ein Verfahren speziell für die Klassifikation von Software-Qualität vor, mit dem zwischen fehleranfälligen und nicht-fehleranfälligen Softwaremodulen diskriminiert werden kann. Das mehrkriterielle Optimierungsproblem soll die erwarteten Kosten einer Fehlklassifikation minimieren, die Anzahl fehlerhafter Module vorhersagen und die Größe des Entscheidungsbaummodells minimieren. Zur Optimierung von Entscheidungsstrukturen verwenden Mugambi und Hunter (2003) einen EA mit Baum-Repräsentation und integrierter Quasi-Newton-Technik (vgl. Fletcher, 1987) zur Koeffizientenoptimierung. Als Zielfunktion betrachten sie die Klassifikationsgüte, die Baumgröße, sowie die Verständlichkeit der Lösungsbäume. Analog zu den Publikationen zur einkriteriellen Optimierung durch EAs mit Baum-Repräsentation

wird auch bei der mehrkriteriellen Optimierung häufig das Problem des *Bloatings* nicht bedacht. Ferner erfolgt nie ein Vergleich mit einem anderen mehrkriteriellen Optimierungsverfahren und selten werden ausgewählte evolutionäre Operatoren und Parametereinstellungen erwähnt und noch seltener begründet.

Im Gegensatz dazu wird in NHEMOtree keinem Zielkriterium eine höhere Priorität zugeordnet als einem anderen und es können alle Arten von Zielkriterien und nicht etwa nur verschiedene Funktionen bezüglich der Vorhersagegüte optimiert werden. In NHEMOtree wird dem Problem des *Bloatings* auf diverse Art und Weise entgegengewirkt, die Auswahl der Selektions- und Variationsoperatoren sowie der Parameter wird begründet (vgl. Abschnit 6.2).

Theoretische Betrachtung

Analog zu den GAs existiert als Erweiterung von Hollands Schema-Theorem (vgl. Holland, 1975) ein Schema-Theorem für Entscheidungsbäume, die in diesem Abschnitt kurz erörtert werden soll. Die Definition von Schemata für Entscheidungsbäume ist schwieriger als für Bitstrings. Eine Übersicht über die verschiedenen Bemühungen, das Schema-Theorem der GAs auf EAs mit Baum-Repräsentation zu übertragen, geben Langdon und Poli (2002). Sie führen exakte Schema-Theoreme ein und zeigen, dass Standard-Rekombinations-Schemata höherer Ordnung aus Schemata niedriger Ordnung zusammengesetzt sind. Die alternativen Definitionen der Schemata lassen sich in zwei Gruppen unterteilen.

Die erste Gruppe der Schema-Theorien für EAs mit Baum-Repräsentation verstehen Schemata hauptsächlich als Komponenten, die sich in der Population fortpflanzen können (vgl. Koza, 1992; Altenberg, 1994; O'Reilly und Oppacher, 1995b; Whigham, 1995). Sei $\theta(S,g)$ die relative Häufigkeit des Schemas S in Generation g, $\hat{F}(S,g)$ die beobachtete mittlere Fitness von S in Generation g, $\overline{F(g)}$ die mittlere Fitness der Population in Generation g, P_x die Wahrscheinlichkeit für die Rekombination und $P_d(S,g)$ der obere Grenzwert für die Wahrscheinlichkeit der Schema-Zerstörung in Generation g. Dann gilt nach dem Schema-Theorem von O'Reilly und Oppacher (1995b) für die erwartete Häufigkeit eines Schemas S in Generation $g+1$

$$E(\theta(S,g+1)) \geq \theta(S,g)\frac{\hat{F}(S,g)}{\overline{F(g)}}(1 - P_x P_d(S,g)).$$

Die erwarteten Häufigkeiten eines Schemas S in Generation $g+1$ ist demnach mindestens so groß wie die Differenz der erwarteten Häufigkeiten der reproduzierten Schemata $\theta(S,g)\frac{\hat{F}(S,g)}{\overline{F(g)}}$ und der durch die Rekombination zerstörten.

Die zweite Gruppe der Schema-Theorien betrachten Schemata als Untermengen des Suchraumes. Der Schwerpunkt liegt dabei auf der Modellierung, wie sich die Anzahl der Individuen in solchen Teilmengen über die Zeit verändern, wie etwa bei dem *Rooted-Tree-Schema* von Rosca (1997) oder dem *Fixed-Size-and-Shape-Schema* von Poli (1997) und Poli und Langdon (1998a). Nach der Meinung von Langdon und Poli (2002) ist der Auswahlprozess der verschiedenen Schemata jedoch nicht reproduzierbar und findet zufällig statt, wobei nicht notwendigerweise nur besonders kurze oder überdurchschnittlich fitte Schemata ausgewählt werden.

Ferner existiert die *Building Block*-Hypothese, die derselben Argumentationskette wie die der GAs (vgl. Abschnitt 4.2) folgt. Im Wesentlichen argumentierte Koza (1992), dass Populationen von EAs mit Baum-Repräsentation *Building Block*s enthalten. *Building Block*s sind in diesem Fall Bäume oder Subbäume, die in einem Teil der Population auftauchen. Gute *Building Block*s verbessern die Fitness der Individuen, sodass Individuen mit guten *Building Block*s mit größerer Wahrscheinlichkeit selektiert und sich stärker in der Population vermehren. Aufgrund dieser *Building Block*s ist es EAs mit Baum-Repräsentation möglich, schneller als rein mutationsbasierte Verfahren gute Problemlösungen zu erstellen. Denn gute *Building Block*s werden zu immer größeren und besseren *Building Block*s kombiniert, die bessere Individuen bilden (vgl. Banzhaf et al., 1998).

Die Nützlichkeit der Schema-Theorien und der *Building Block*-Hypothese für EAs mit Baum-Repräsentation wurden weithin kritisiert (vgl. Langdon und Poli, 2002; Poli, 2001a; O'Reilly und Oppacher, 1995b). So fassen etwa O'Reilly und Oppacher (1995b) zusammen, dass das Schema-Theorem wichtige Abhängigkeiten von der Rekursion ausspart und somit die Dynamik zu stark vereinfacht. Die Hauptschwächen des Schema-Theorems seien zum einen die Beschreibung der Rekombinations- und Selektionsentwicklung über nur eine Generation hinweg, obwohl die Annahmen der *Building Block*-Hypothese auf dem Langzeitverhalten von Rekombination und Selektion basieren. Zum anderen sei die Formulierung von nur unteren Grenzen für die erwartete Anzahl an Schemata unzureichend (vgl. O'Reilly und Oppacher, 1995b).

Neuere exakte Schema-Theoreme von Poli und McPhee (2003), Poli (2001a) und Poli (2001b) geben jedoch eine genaue Formulierung anstelle einer unteren Grenze für die erwartete Anzahl an Schemata in der nächsten Generation. Ferner zeigen experimentelle Ergebnisse, dass die stark zerstörerischen Effekte der Rekombination nicht für EAs mit Baum-Repräsentation und Ein-Punkt-Rekombination oder Subbaumtausch gelten und somit die *Building Block*-Hypothese immer noch als eine erste gute Approximation des Verhaltens EAs mit Baum-Repräsentation gilt (vgl.

Poli und McPhee, 2003; Langdon und Poli, 2002; Poli, 2001a,b; Poli und Langdon, 1997).

Bloating

Ein für Algorithmen mit flexibler Individuengröße spezifisches Problem wird durch sogenannte Introns gefördert (vgl. Nordin et al., 1995). Introns beschreiben durch den evolutionären Prozess erstellte Codeabschnitte in einem Individuum, die keinen Einfluss auf die Fitness des Individuums haben. Die Menge von nicht-funktionalem Code in einem Individuum wächst üblicherweise exponentiell mit fortlaufender Generationenanzahl an. Somit wachsen die Individuen ohne Fitnesssteigerung. Im Allgemeinen wird dieser Prozess als *Bloating* bezeichnet.

Begründungen für das *Bloating* liefern etwa Soule und Foster (1998) und De Jong und Pollack (2003). *Bloating* schützt vor destruktiven Operationseffekten. Je mehr nicht-funktionaler Code, der weder die Fitness noch die Funktionalität des Baums beeinflusst, in einem Individuum enthalten ist, desto unwahrscheinlicher ist die Zerstörung von guten *Building Block*s durch Rekombination oder Mutation. Des Weiteren führt die Tatsache, dass das Entfernen eines kleineren Subbaums im Gegensatz zum Löschen großer Subbäume einen geringeren Einfluss auf die Fitness hat, zu *Bloating*. Da keine solche Beziehung für das Hinzufügen von Subbäumen existiert, wachsen die Individuen tendenziell. Gustafson et al. (2004) wiesen einen Zusammenhang zwischen Problemschwierigkeit, Selektionsdruck und Diversitätsverlust auf. Je schwieriger ein Problem ist, desto höher ist der Selektionsdruck und desto niedriger die Diversität der Lösungen. Ein erhöhter Selektionsdruck und eine verringerte Diversität haben wiederum *Bloating* zur Folge. Nachfolgend werden Vorschläge zur Limitierung des Baumwachstums vorgestellt.

Zu den verschiedenen Mechanismen zur Linderung des unkontrollierten Code-Wachstums gehören die Größenrestriktion der Bäume, die Einführung von Größenbestrafungen in der Fitnessfunktion, die Verwendung nichtzerstörerischer Rekombinationsoperatoren und die Anwendung von Fitnessfunktionen, die sich über den evolutionären Prozess hinweg verändern (vgl. Koza, 1992).

Die Baumgröße kann entweder über die Beschränkung der Baumtiefe oder der Knotenanzahl in einem Baum reglementiert werden. Beides sind wichtige Einstellungen in EAs mit Baum-Repräsentation, da sie die Größe des Lösungsraums und damit die Lösungsqualität begrenzen. Je größer ein Baum ist, desto mehr potentielle Lösungen existieren und umso wahrscheinlicher ist das Auffinden einer für den Anwender optimalen Lösung. Die Baumgröße wirkt sich allerdings auch proportional zum Rechenaufwand aus und beeinflusst die Geschwindigkeit der Lösungsfindung. Opti-

male Einstellungen für die maximale Baumtiefe bzw. die Knotenanzahl existieren im Allgemeinen nicht, sondern müssen je Problemstellung individuell getroffen werden.

Durch die Beschränkung der Baumtiefe werden mit höherer Wahrscheinlichkeit balancierte Bäume mit ähnlicher Knotenanzahl in den jeweiligen Subbäumen erstellt, da sich die Bäume nach Erreichen der maximalen Baumtiefe nur noch in die Breite vergrößern können. Dies schränkt die Flexibilität des EAs bei der Suche nach Lösungen verschiedener Formen ein. Bei Limitierung der Knotenanzahl wird hingegen die Form der Bäume nicht verzerrt (vgl. De Jong und Pollack, 2003). Aber auch flexible Schranken zur Reglementierung der Baumgröße, die sich beispielsweise über die Generationen hinweg verändern, sind denkbar (vgl. Silva und Almeida, 2003).

Neben der reinen Größenbeschränkung können größere Individuen auch durch die Verminderung ihrer Fitnesswerte bestraft werden. Dieser zusätzliche Selektionsdruck begünstigt fittere und kleinere Individuen. Bei der Tarpeian-Methode (vgl. Poli, 2003) wird beispielsweise die Fitness von Subbäumen mit überdurchschnittlicher Größe auf Null gesetzt und somit das *Bloating* durch eine direkte Einwirkung auf die Selektionswahrscheinlichkeit kontrolliert. Weitere Methoden, komplexe Bäume in ihrer Fitnessfunktion zu bestrafen, beschreiben etwa Poli et al. (2008).

Ein anderes Mittel zur Reduzierung des *Bloatings* ist die Modifikation der Variationsoperatoren. So können etwa die Rekombinations- und Mutationsrate variiert werden, um dem evolutionären Druck entgegen zu wirken oder um destruktive Rekombinationsvorgänge zu verhindern. Die Wichtigkeit der Tiefenabhängigkeit bei der Rekombination zeigen etwa Poli und Langdon (1998a). Sie wählen bei ihrer Größen-gerechten Rekombination den ersten Rekombinationspunkt zufällig aus. Der zweite Punkt wird hingegen so gewählt, dass die rekombinierten Individuen nicht zu groß werden (vgl. Abbildung 6).

Alternativ können Fitness und Größe als gleichberechtigte Zielfunktionen mehrkriteriell optimiert werden. Im Allgemeinen gilt jedoch, dass die alleinige Nutzung von Individuengröße und -fitness als Zielfunktionen ohne ausgiebige Parametersuche zwar das Baumwachstum reduziert, aber auch gleichzeitig die Fitness verschlechtert (vgl. Deb et al., 2002).

Die verschiedenen vorgestellten Methoden zur Vermeidung von *Bloating* werden in der Praxis vielfach verwendet. So benutzen etwa Muni et al. (2004) und Liu und Xu (2009) eine Begrenzung der Baumgröße bzw. -tiefe. Badran und Rockett (2008) verwenden die tiefenabhängige Rekombination von Ito et al. (1998) und Jabeen und Baig (2011) nutzen die Ein-Punkt-Rekombination von Poli und Langdon (1998a). Im Vergleich zu anderen Rekombinationsarten schneiden sie ebenfalls gut ab, und führen zu Bäumen mit weniger Knoten.

Da die für ein Klassifikationsmodell wünschenswerten Eigenschaften Verständlichkeit und Interpretierbarkeit mit der Größe des Modells korreliert sind, ist eine Größen-Reglementierung unumgänglich (vgl. Otero et al., 2003). Übermäßig große Bäume bringen eine schlechtere Generalisierung als kleinere mit sich (vgl. De Jong und Pollack, 2003) und somit eine geringere Verständlichkeit für den Anwender. Ferner können sie zur Überanpassung und zu längeren Berechnungsdauern führen und sich nachteilig auf die gesamte Suche auswirken.

Zwar ist in NHEMOtree (vgl. Abschnitt 6.2) aufgrund der mehrkriteriellen Optimierung im Allgemeinen weniger *Bloating* zu erwarten. Dennoch werden aufgrund der negativen Aspekte des *Bloatings* auch in NHEMOtree Verfahren zur Reduzierung des unkontrollierten Code-Wachstums untersucht. Hierzu gehört under anderem die Größen-Reglementierung der Bäume durch die Begrenzung der maximalen Knotenanzahl je Baum, da diese die Flexibilität der Baumstruktur weniger stark einschränkt als die Begrenzung der Baumtiefe. Zum anderen werden diverse Variationsoperatoren mit Wachstum-begrenzendem Faktor in NHEMOtree integriert. Die Fitnessbestrafung zu großer Bäume wird in NHEMOtree hingegen nicht verwendet, da dafür a priori eine Gewichtung der Fitnessfunktionen bezüglich der Vorhersagegüte und der Baumgröße getroffen werden müsste. Dies widerspricht dem Prinzip der mehrkriterielle Optimierung, bei der der Anwender a posteriori eine Pareto-optimale Lösung auswählt.

Initialisierung

Die *ramped-half-and-half*-Methode von Koza (1992) ist die gängigste Methode zur Erstellung der initialen Population in EAs mit Baum-Repräsentation. Sie gewährleistet die gewünschte initiale Populationsvielfalt durch die kombinierte Anwendung der *Grow Method* auf die eine und der *Full Method* auf die andere Populationshälfte. Dabei werden verschiedene maximale Knotenanzahlen bzw. Baumtiefen herangezogen.

Die *Grow Method* stoppt das Baumwachstum zufällig, jedoch unter Einbehaltung der maximalen Baumtiefe oder Knotenanzahl. Dadurch entstehen unregelmäßige Baumstrukturen bis hin zur maximalen Größe. Die *Full Method* hingegen erzeugt so lange Knoten bis stets die maximale Baumtiefe bzw. Knotenanzahl erreicht ist. Auf diese Weise werden sehr ähnliche Baumstrukturen erzeugt.

Variationsoperatoren

Der vorherrschende Variationsoperator in EAs mit Baum-Repräsentation ist die Rekombination (vgl. Tabelle 1). Die Rekombination wird als Grund angesehen, warum diese effektiver sind als Systeme, die nur auf Mutationen basieren. Denn gute *Building Block*s kombinieren sich zu immer besseren und größeren *Building Block*s, die immer bessere Individuen bilden.

Neben dem Standard-Rekombinationsoperator X_S in den EAs mit Baum-Repräsentation werden nachfolgend verbesserte Rekombinationsoperatoren, nämlich die Brut-Rekombination X_B sowie die tiefenabhängige Ein-Punkt-Rekombination nach Poli und Langdon (1998a) X_P beschrieben. Ferner existieren auch spezielle Mutationsoperatoren für die Baum-Repräsentation. Sie mutieren entweder die Baumstruktur, die Variablen in den Baumknoten oder die Cutoffs.

Standard-Rekombination X_S

Als binärer Ein-Punkt-Rekombinationsoperator wählt der Standard-Rekombinationsoperator X_S die Rekombinationsknoten in beiden Elternbäumen zufällig aus, wobei der Wurzelknoten ausgenommen ist. Die beiden so ermittelten Subbäume (grau gekennzeichnet) werden zwischen den Eltern getauscht und an die Stelle des jeweils alten Subbaums eingefügt (vgl. Abbildung 5). So können mit X_S beliebige Subbäume getauscht und erstellte *Building Block*s wieder zerstört werden.

In der Natur wird viel Energie darauf verwendet, gute *Building Block*s vor den schädlichen Einflüssen der Rekombination zu schützen. So werden dort etwa nur selten zwei Gene mit vollkommen verschiedenem Zweck gegeneinander getauscht (vgl. Nordin et al., 1995). Für EAs zeigen O'Reilly und Oppacher (1994) sowie O'Reilly und Oppacher (1995a), dass durch X_S Individuen mit großen syntaktischen Unterschieden im Vergleich zu den Eltern erzeugt werden. Trotz der großen Diskrepanz zwischen X_S und der biologischen Rekombination funktioniert X_S jedoch laut Banzhaf et al. (1998) gut.

Neben den Unterschieden zur Biologie erscheint laut Poli und Langdon (1998b) X_S ferner ungeeignet, da er einen lokalen und verzerrten Suchoperator darstellt, der den Suchraum nur unvollständig durchsucht. Der Operator ist lokal in dem Sinne, dass die resultierenden Nachkommen hauptsächlich Informationen eines Elternteils besitzen. Außerdem existiert bei der Selektion des Rekombinationspunktes eine implizite Verzerrung zu den Blättern (vgl. Harries und Smith, 1997). Somit ist X_S kein idealer Suchoperator und kann zu *Bloating* führen, sodass bereits diverse Alternativen zu X_S entwickelt wurden.

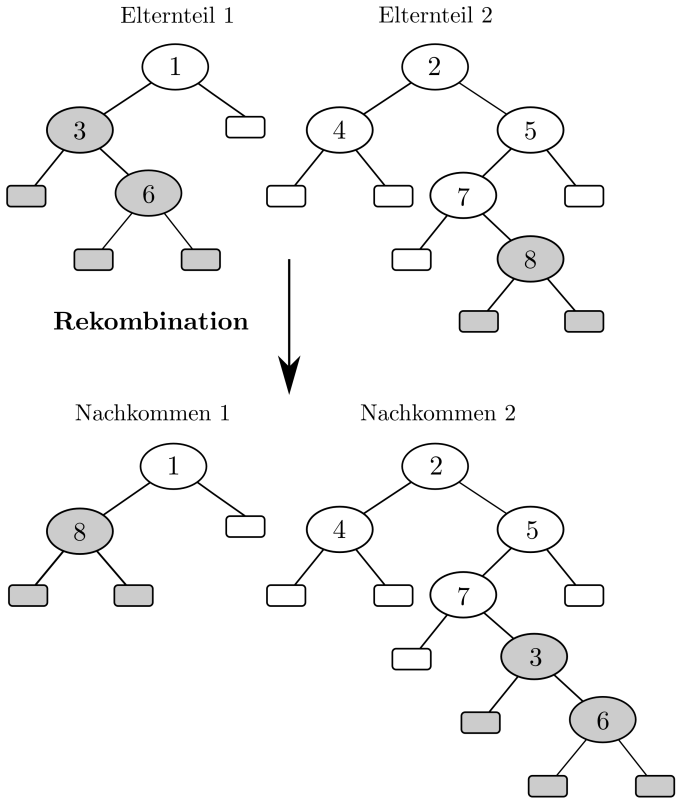


Abbildung 5: Standard-Rekombinationsoperator X_S für Evolutionäre Algorithmen mit Baum-Repräsentation

Alternative Rekombinationsoperatoren

Alternative Rekombinationsoperatoren versuchen, dem biologischen Vorbild ähnlicher zu werden. In der Vergangenheit zeigten sich bereits erfolgreiche Erweiterung von X_S, die durch die Adjustierung der Rekombinationstiefe die erwähnte Verzerrung des Rekombinationspunktes zu den Blättern aufheben und die Werte der Fitnessfunktionen verbessern (vgl. Xie und Zhang, 2011; Xie et al., 2007; Zhang et al., 2006; Majeed und Ryan, 2006; Banzhaf et al., 1998; Poli und Langdon, 1998a; Ito et al., 1998).

Bei der tiefenabhängigen Rekombination, bei der die Auswahlwahrscheinlichkeit eines Rekombinationsknotens lediglich von dessen Position im Baum mit exponentiell abfallenden Wahrscheinlichkeiten abhängt, postulieren Ito et al. (1998), dass große Subbäume mit einer höheren Wahrscheinlichkeit rekombiniert werden sollten. Der Rekombinationspunkt sollte also näher am Wurzelknoten sein, um der zerstörerischen Natur von X_S entgegen zu wirken. Ähnlich argumentieren Xie und Zhang (2011). Sie schlagen ebenfalls ungleiche Tiefenselektionswahrscheinlichkeiten vor, erwähnen jedoch, dass eine effektive Tiefenkontrolle abhängig vom Optimierungsproblem ist. Weitere tiefenkontrollierende Strategien für die Rekombination fassen etwa Xie und Zhang (2011) zusammen.

Aus der Vielzahl von bereits existierenden Rekombinationsoperatoren werden im Folgenden zwei viel versprechende Rekombinationsoperatoren für Individuen mit Baum-Repräsentation vorgestellt. Diese kommen neben X_S als Basis-Rekombinationsoperator in NHEMOtree zur Anwendung und werden miteinander verglichen.

Brut-Rekombination X_B

Der Brut-Rekombinationsoperator (X_B) stellt keine eigentliche Verbesserung von X_S dar, dämmt aber dessen negative Auswirkungen stark ein. Die Reduzierung des zerstörerischen Effekts der Rekombination erfolgt durch die Erstellung mehrerer Nachkommen, von denen wie bei vielen Tierarten nicht alle überleben. Anstelle nur ein oder zwei Nachkommen zu erzeugen, wird die Rekombination zweier ausgewählter Eltern ρ-mal durchgeführt, sodass 2ρ Nachkommen, die so genannte Brut, entstehen. Im Allgemeinen überleben die ein oder zwei fittesten Nachkommen aus der Brut, die restlichen Nachkommen werden verworfen. Da die Balance zwischen zerstörerischer und konstruktiver Rekombination verändert wird, können größere *Building Block*s erhalten bleiben. Mit ansteigender Größe der *Building Block*s vergrößert sich jedoch auch die Wahrscheinlichkeit für eine destruktive Rekombination. *Bloating* könnte daraufhin einsetzen, sodass die größeren *Building Block*s geschützt werden.

Mit zunehmender Brutgröße ρ kann das Optimierungsergebnis verbessert werden. Übersteigt die Brutgröße jedoch einen bestimmten Punkt, wird das System wieder weniger effizient. Zhang et al. (2006) untersuchen neben verschiedenen festen Brutgrößen auch dynamische Modelle, bei der die Brutgröße mit wachsender Generationenanzahl steigt. Für Klassifikationsprobleme verwendeten sie feste Brutgrößen zwischen zwei bis zehn, wobei solche von vier, sechs oder acht die besten Ergebnisse lieferten. Die Brutgröße, die zu den besten Ergebnissen für ein bestimmtes Problem führt, heißt *brood-diversity point*. Üblicherweise wird eine Brutgröße von $\rho = 4$ genutzt (vgl. Tackett, 1994). Jedoch ist die Brutgröße abhängig von dem zu lösenden Problem und den anderen evolutionären Parametern.

Da meistens die maximale Baumgröße limitiert ist, ist die Wahrscheinlichkeit, bei einer Rekombination derselben Eltern stets redundante Nachkommen zu erzeugen, hoch. Übersteigt die Brutgröße den *brood-diversity point*, produziert X_B nicht mehr unterschiedliche Nachkommen und benötigt längere Zeit für mehr Generationen. Demnach sollte bei der Festlegung der Brutgröße die Menge der möglichen Rekombinationspunkte beachten werden.

Tiefenabhängige Ein-Punkt-Rekombination X_P

In Hinblick auf die Vorteile der Rekombinationsoperatoren in GAs entwickelten Poli und Langdon (1998a) für EAs mit Baum-Repräsentation den tiefenabhängigen Ein-Punkt-Rekombinationsoperator (X_P). Hierbei wird ein Rekombinationspunkt mit gleichverteilter Wahrscheinlichkeit aus der Menge aller potentiellen Punkte ausge-

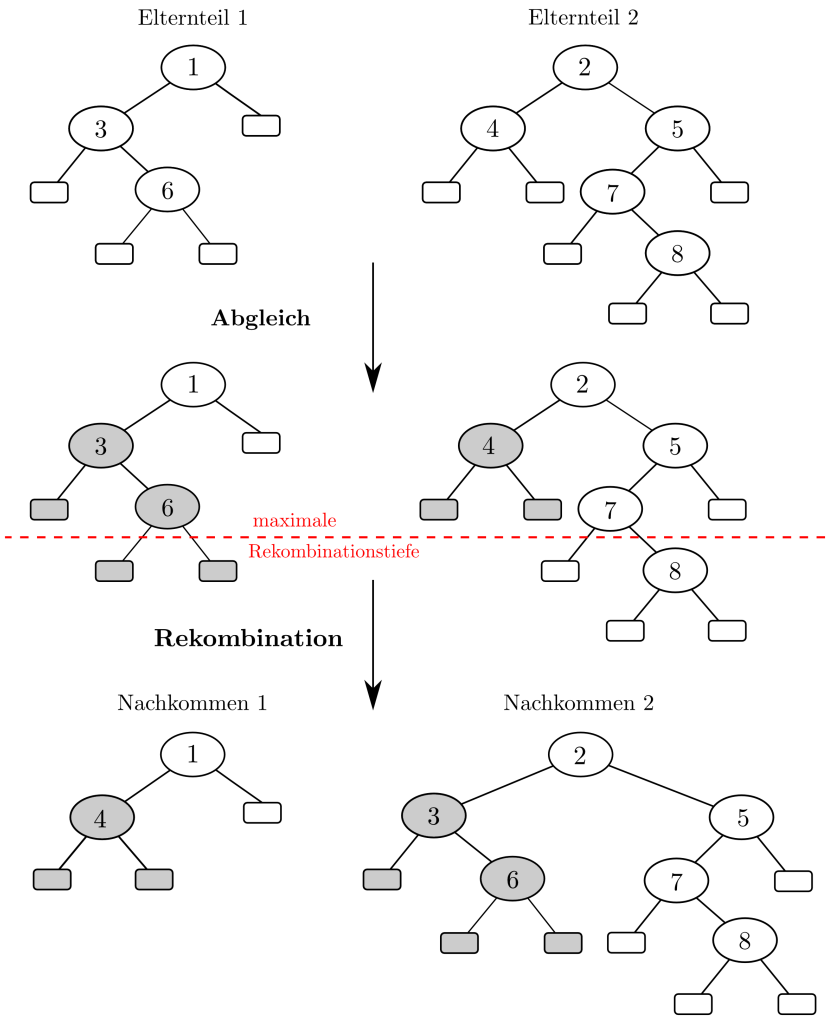


Abbildung 6: Tiefenabhängiger Ein-Punkt-Rekombinationsoperator X_P für Evolutionäre Algorithmen mit Baum-Repräsentation nach Poli und Langdon (1998a)

wählt. Jedoch muss sich der Rekombinationspunkt in beiden Elternteilen auf derselben Baumebene befinden. Dadurch ist die Wahrscheinlichkeit für die Auswahl von Punkten in tieferen Baumabschnitten gering, die Baumgröße variiert nicht so stark und beläuft sich im Rahmen der initialen Individuengrößen. Laut Poli und Langdon (1998a) führt X_P zu einer schnellen Konvergenz im oberen Baumabschnitt und ist besser als X_S, da zu Beginn eines Laufs eine globale Suche des Suchraums vollzogen wird. Mit steigender Generationenzahl und der damit verbundenen Konvergenz der Population wird die Suche immer lokaler und verzerrter, sodass das ersetzte Material durch den Rekombinationsoperator dem Ausgangsmaterial immer ähnlicher wird. Abbildung 6 zeigt das Vorgehen bei der tiefenabhängigen Ein-Punkt-Rekombination. Lediglich Knoten bis zu einer maximalen Tiefe von 3 (oberhalb der gestrichelten Linie) kommen in diesem Fall als Rekombinationsknoten in Frage. Jabeen und Baig (2011) zeigen, dass dieser Rekombinationsoperator zu einer Verringerung des *Bloatings* führt.

Typ	Mutation	Beschreibung	Abb.
Baum	Subbaum-Löschung	Löschen eines zufällig ausgewählten Subbaums	7(b)
	Subbaum-Austausch	Austausch eines zufällig ausgewählten Subbaums durch einen neuen Baum	7(c)
	Expansion	Baumerweiterung durch einen zusätzlichen Subbaum	7(d)
	Permutation	Tausch der Subbäume unterhalb eines zufällig ausgewählten Knotens	7(e)
	Hoist	Auswahl eines zufälligen Subbaums als neuen Baum	7(f)
Knoten	Punktmutation	Variablenaustausch in einem zufällig ausgewählten Knoten	7(g)
	Variablentausch	Tausch der Variablen zweier zufällig ausgewählten Knoten im Baum	7(h)
Cutoff	Gauß-Mutation	Adaption der Variablen-Cutoffs eines Baums durch die Gauß-Mutation mit Schrittweitenanpassung nach Rechenbergs 1/5-Erfolgsregel	

Tabelle 2: Mutationsoperatoren in Evolutionären Algorithmen mit Baum-Repräsentation und NHEMOtree

Mutation

Bei EAs mit Baum-Repräsentation sind drei verschiedene Mutationsarten denkbar (vgl. Tabelle 2 und Abbildung 7). Zum einen kann die Struktur des Ausgangsbaums mutiert werden. Dazu gehören laut Poli et al. (2008) das Löschen eines Subbaums *(Mutation b)*, der Austausch eines Subbaums durch einen zufällig erstellten Baum *(Mutation c)*, die Expansion des Ausgangsbaums durch Ersetzen eines Blattes durch einen zufällig erstellten Baum *(Mutation d)*, das Vertauschen der Subbäume unterhalb des entsprechenden Knotens *(Mutation e)* und die so genannte Hoist-Mutation von Kinnear Jr. (1993), bei der ein Subbaum des Ausgangsbaums zum neuen Individuum wird *(Mutation f)*. Wyns und Boullart (2009) entwickeln eine Fitness-basierte Hoist-Mutation, bei der die Zufallskomponente der Subbaum-Auswahl durch eine Fitness-orientierte Komponente ersetzt wird. Bei Auswahl des fittesten Subbaums als Nachkommen erzielen sie die besten Ergebnisse. Zum anderen können die Variablen in den Baumknoten mutiert werden. Dabei kommt zum einen die Punktmutation, bei der ein Knoten gegen einen anderen zufälligen Knoten getauscht wird *(Mutation g)*, oder der Variablentausch zwischen zwei Knoten *(Mutation h)* zum Tragen. Im Allgemeinen wird für die verschiedenen Mutationsoperatoren eine Mutationsrate p_m festgesetzt und jeweils zufällig entschieden, welche Art der Mutation ausgeführt wird.

Die Cutoff-Werte werden separat mit eigener Mutationsrate durch eine Gauß-Mutation mutiert (vgl. Rechenberg, 1973). Die Gauß-Mutation mutiert einen Lösungsvek-

tor $\boldsymbol{x} = (x_1, \ldots, x_n)$ durch Addition normalverteilter Zufallszahlen. Die Standardabweichung wird dabei als Schrittweite bezeichnet, die konstant aber auch individuell gewählt werden kann

$$\begin{aligned} &\boldsymbol{x}* = \boldsymbol{x} + \boldsymbol{z}, \boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 I_n) && \text{konstante Schrittweite} \\ &x_i* = x_i + z_i, z_i \sim \mathcal{N}(0, \sigma_i^2),\ i = 1, \ldots, n && \text{individuelle Schrittweite.} \end{aligned}$$

Die Größe der Schrittweite sollte im Laufe des Algorithmus angepasst werden. Die einfachste Möglichkeit hierfür ist die deterministische Steuerung über die aktuelle Generationenanzahl. Eine adaptive Steuerung kann etwa über das Verhältnis erfolgreicher gegenüber nicht erfolgreicher Mutationen erfolgen. Dabei wird die Schrittweite zunächst über eine gewisse Generationenanzahl konstant gehalten und die Erfolgswahrscheinlichkeit der Mutationen berechnet. Rechenbergs (1973) adaptive Schrittweitenanpassung beruht auf der theoretisch abgeleiteten 1/5-Erfolgsregel, die besagt, dass der Quotient aus den erfolgreichen Mutationen, die eine Verbesserung der Fitness bewirken, zu allen Mutationen etwa ein Fünftel betragen sollte. Bei einem größeren Quotienten sollte die Streuung der Mutationen erhöht, bei kleinerem verringert werden, sodass mit $\alpha \in [0{,}817;\ 1]$ gilt

$$\sigma = \begin{cases} \sigma/\alpha & \text{falls Erfolgswahrscheinlichkeit } > 1/5 \\ \sigma \cdot \alpha & \text{falls Erfolgswahrscheinlichkeit } < 1/5 \\ \sigma & \text{falls Erfolgswahrscheinlichkeit } = 1/5 \end{cases}.$$

Wahl der Kontrollparameter

Analog zu den GAs existieren auch in den EAs mit Baum-Repräsentation vielfältige Möglichkeiten zur Wahl der Kontrollparameter, jedoch arbeiten diese laut Banzhaf et al. (1998) gut über eine große Bandbreite an Einstellungen hinweg. Die Komplexität des Problems sollte allerdings stets ausschlaggebend für die Wahl der Populationsgröße μ sein (vgl. Koza, 1992). Je schwieriger das Problem, desto größer sollte μ gewählt werden, da große Populationen eine größere Diversität mit sich bringen und den Suchraum besser entdecken.

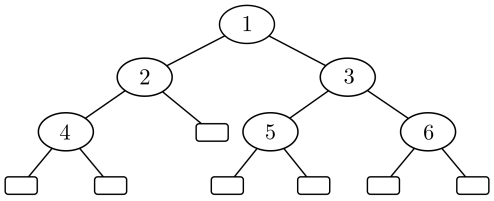
Aufgrund langer Berechnungsdauern sollte die Anzahl der Generationen nicht zu groß gewählt werden. Banzhaf et al. (1998) schlagen vor, mit geringen Generationsanzahl zu starten (50 bis 100 Generationen) und bei unzulänglichen Ergebnissen zunächst die Populationsgröße und dann die Generationenanzahl zu erhöhen.

Im Allgemeinen sind geringe Mutations- und hohe Rekombinationsraten sehr effektiv (vgl. Zhao, 2007). Sind die Ergebnisse nicht zufriedenstellend, sollte zunächst die Mutationsrate erhöht werden. Die schlechtesten Ergebnisse wurden erzielt, wenn einer der beiden Variationsoperatoren nicht verwendet wurde.

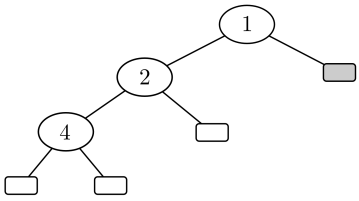
4.4 Fazit

EAs basieren auf dem Prinzip der biologischen Evolution und erfordern genau wie andere direkte Suchverfahren nur ein geringes Problemwissen und besitzen keine Voraussetzungen an die Datenverteilung, sodass sie zur Lösung vielfältiger Probleme eingesetzt werden können. Die verschiedene Dialekte der EAs folgen alle demselben Grundprinzip und unterscheiden sich hauptsächlich in der Repräsentation der Individuen. Zu Beginn des EAs wird eine zufällige Population potentieller Lösungen des Optimierungsproblems erzeugt. Diese werden anhand einer oder mehrerer Fitnessfunktionen, die das Optimierungsproblem beschreiben, bewertet. Individuen mit hohen Fitnesswerten werden als Eltern selektiert und aus ihnen mit Hilfe von Variationsoperatoren neue Individuen, die Nachkommen, generiert. Die Nachkommen werden wiederum bewertet und aus alten und neuen Individuen die neue Population gebildet. Dieser Kreislauf setzt sich fort, bis ein Abbruchkriterium erfüllt ist.

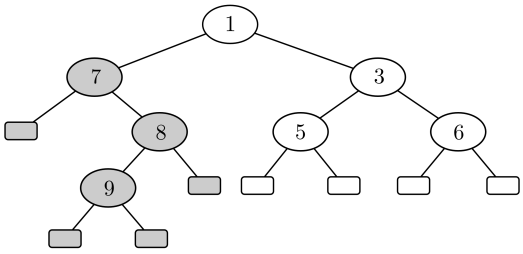
In dieser Arbeit werden EAs mit Bitstring-Repräsentation (GAs), die als *Wrapper* um einen Klassifikationsalgorithmus eingesetzt werden (vgl. Abschnitt 6.1), und EAs mit Baum-Repräsentation in Form des NHEMOtrees (vgl. Kapitel 8) untersucht. Letztere können direkt eine Variablenselektion und -bewertung durchführen und begünstigen so die direkte Beurteilung der betrachteten Variablenteilmenge. Dies macht einen internen Klassifikationsalgorithmus zur Variablenbewertung überflüssig. Die Anwendung von EAs mit Baum-Repräsentation auf Klassifikationsprobleme hat viele Vorteile, vor allem dessen Flexibilität, die eine Adaption an das spezielle Problem erlaubt. Auf der anderen Seite existiert bei EAs mit Baum-Repräsentation das Problem des *Bloatings*. Ansätze, das *Bloating* durch die Verwendung spezieller Variationsoperatoren, Größenbestrafungen in den Fitnessfunktionen oder Größenrestriktionen der Bäume zu verringern, wurden in diesem Kapitel erörtert und werden in NHEMOtree integriert.



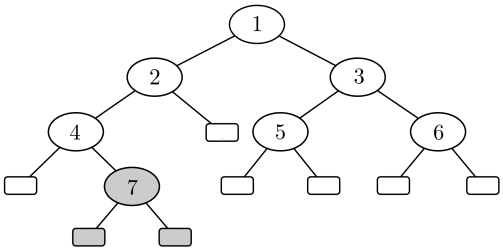
(a) Originalbaum



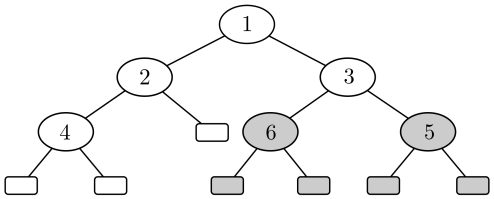
(b) Baum nach Subbaum-Löschung



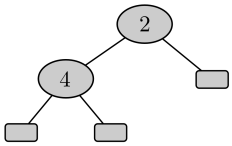
(c) Baum nach Subbaum-Austausch



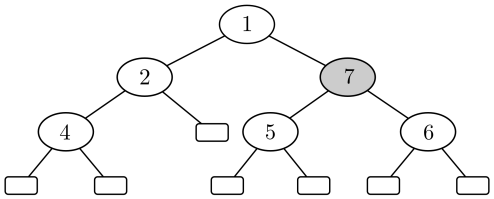
(d) Baum nach Expansionsmutation



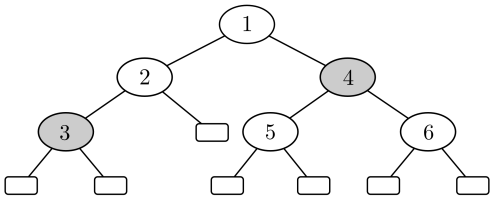
(e) Baum nach Permutation



(f) Baum nach Hoist-Mutation



(g) Baum nach Punktmutation



(h) Baum nach Vertauschungsmutation

Abbildung 7: Mutationsoperatoren in Evolutionären Algorithmen mit Baum-Repräsentation und in NHEMOtree

5 Mehrkriterielle Optimierung

Ziel der Optimierung ist es, basierend auf Optimalitätskriterien die beste Lösung für ein gegebenes Problem zu finden, wobei die beste Lösung als Optimum bezeichnet wird. Voraussetzungen für die Optimierung sind zum einen qualitativ messbare und unterscheidbare Lösungen. Zum anderen ist ein funktionaler Zusammenhang, der durch eine Zielfunktion zwischen den Parametern und den Optimalitätskriterien dargestellt wird, notwendig.

Im Gegensatz zu einkriteriellen Optimierungsproblemen besitzen mehrkriterielle Optimierungsprobleme zwei oder mehr sich widersprechende Zielfunktionen (vgl. Definition 4). Diese sollen gleichzeitig durch eine mehrkriterielle Funktion $\boldsymbol{f}$ optimiert werden, sodass Lösungen gefunden werden, die in allen Zielfunktionen möglichst optimal sind.

Definition 4 (Mehrkriterielles Optimierungsproblem)
Bei gegebenen Variablen $X_1,\ldots,X_p$ aus der Variablenmenge $\mathcal{X} \subset \mathbb{R}^p$ ergibt sich ein unbeschränktes mehrkriterielles Optimierungsproblem als

$$\min_{\boldsymbol{x}\in\mathcal{X}} (f_1(\boldsymbol{x}),\ldots,f_m(\boldsymbol{x}))$$

mit $\boldsymbol{f} = (f_1,\ldots,f_m) : \mathcal{X} \to \mathbb{R}^m$ und $m \geq 2$ Zielen.
O.B.d.A. werden wegen $\max(f) = -\min(-f)$ in dieser Arbeit ausschließlich Minimierungsprobleme betrachtet.

Im Mehrkriteriellen existiert im Gegensatz zum Einkriteriellen jedoch kein eindeutiges Optimum. Lösungen sind unvergleichbar, wenn jeder Lösungsvektor mindestens eine bessere Komponente enthält als der andere, wie etwa die Vektoren (1,2,1) und (1,1,3). Ein Vektor $\boldsymbol{x}$ dominiert einen anderen Vektor $\boldsymbol{x'}$ ($\boldsymbol{x} \prec \boldsymbol{x'}$), falls $\boldsymbol{x}$ in allen Kriterien mindestens genauso gut wie $\boldsymbol{x'}$ und in mindestens einem Kriterium überlegen ist:

$$\boldsymbol{x} \prec \boldsymbol{x'} \Leftrightarrow \forall i \in \{1,\ldots,m\} : f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x'}) \wedge \exists\, i : f_i(\boldsymbol{x}) < f_i(\boldsymbol{x'}), \quad \boldsymbol{x},\boldsymbol{x'} \in \mathcal{X}.$$

Ein Beispiel hierfür wären die Vektoren $\boldsymbol{x}$ = (1,1,3) und $\boldsymbol{x'}$ = (1,2,3). $\boldsymbol{x}$ heißt nicht-dominiert in einer Menge von Lösungen $\mathcal{L} \subseteq \mathcal{X}$ genau dann, wenn kein $\boldsymbol{x'}$ in $\mathcal{L}$ existiert, das $\boldsymbol{x}$ dominiert. Nicht-dominierte Lösungen im gesamten Suchraum heißen Pareto-optimal. Diese können nicht verbessert werden, ohne eine Verschlechterung in mindestens einem anderen Kriterium in Kauf nehmen zu müssen. Die Menge aller Pareto-optimalen Lösungen heißt Pareto-Menge $\mathcal{P} := \{\boldsymbol{x} \in \mathcal{X} \mid \nexists\, \boldsymbol{x'} \in \mathcal{X} : \boldsymbol{x'} \prec \boldsymbol{x}\}$,

die Menge der Funktionswerte dieser Lösungen heißt Pareto-Front

$$\mathcal{PF} := \{f(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{X} \wedge \nexists\, \boldsymbol{x'} \in \mathcal{X} \colon \boldsymbol{x'} \prec \boldsymbol{x}\}.$$

Das in dieser Arbeit zu lösende Optimierungsproblem ist die kostensensitive Klassifikation zur Vorhersage von Lungenkrebssubtypen durch wenige, günstige Biomarker. Dafür müssen folgende univariate Probleme gemeinsam optimiert werden:

- Finde für die Trainingsdaten D ein Modell $h(\mathcal{X}_s, D)$ mit minimaler Fehlklassifikationsrate der Lungenkrebssubytpen.
- Finde eine Teilmenge $\mathcal{X}_s \subset \mathcal{X}$ aus der Menge der Biomarker mit minimalen Kosten.

Dazu werden die Fitnessfunktionen $f_1(h(\mathcal{X}_s, D), V)$ als Fehlklassifikationsrate über alle mögliche Modelle $h(\mathcal{X}_s, D)$, $\mathcal{X}_s \subset \mathcal{X}$, auf den Testdaten V und $f_2(\mathcal{X}_s)$ als Kosten der ausgewählten Biomarker definiert und in der gemeinsamen Fitnessfunktion $\boldsymbol{f}(\mathcal{X}_s) = (f_1, f_2)$ zusammengefasst. Schließlich ist das mehrkriterielle Optimierungsproblem $\min\limits_{\mathcal{X}_s \in P(\mathcal{X})} \boldsymbol{f}(\mathcal{X}_s)$ zu lösen.

In diesem Kapitel werden verschiedene Ansätze und Verfahren zur Lösung mehrkriterieller Optimierungsprobleme vorgestellt. Diese lassen sich in aggregierende (vgl. Abschnitt 5.1) und Pareto-basierte Ansätze (vgl. Abschnitt 5.2) unterteilen (vgl. Zitzler et al., 2000). Evolutionäre mehrkriterielle Optimierungsalgorithmen (EMOAs), die sich gut für die Lösung mehrkriterieller Optimierungsprobleme eignen, werden in Abschnitt 5.3 erläutert. Abschnitt 5.4 fasst die wichtigsten Punkte des Kapitels zusammen.

5.1 Aggregierende mehrkriterielle Optimierung

Die aggregierende mehrkriterielle Optimierung stellt einen einfachen und häufig verwendeten Lösungsansatz für mehrkriterielle Optimierungsprobleme dar (vgl. Ephzibah, 2010; Alba et al., 2007; Bosin et al., 2007; Iswandy und Koenig, 2006; Juliusdottir et al., 2005; Bojarczuk et al., 2004; Papagelis und Kalles, 2001; Ishibuchi und Nakashima, 2000; Noda et al., 1999; Yang, 1998; Siedlecki und Sklansky, 1989). Dabei werden die Zielvektoren eines mehrkriteriellen Optimierungsproblems auf ein einziges Ziel aggregiert, um eine einzelne Lösung zu erstellen.

Eine klassische Möglichkeit hierfür ist die Gewichtungsmethode, die mehrere Zielfunktionen $f_i, i = 1, \ldots, m$, in einer Konvexkombination

$$F(\boldsymbol{x}) = \sum_{i=1}^{m} w_i f_i(\boldsymbol{x})$$

zusammenfügt. Dabei beschreibt $\boldsymbol{w} \in [0,1]^m$, $\sum_{i=1}^{m} w_i = 1$, den Gewichtungsvektor, der durch den Anwender a priori definiert werden muss, um das Verhältnis zwischen den Zielfunktionen auszudrücken. Die Präferenz eines Ziels f_i kann durch die Modifikation des entsprechenden Gewichts einfach verändert werden. Beispiele für die Reduktion eines mehrkriteriellen Optimierungsproblems auf ein einkriterielles zeigen etwa Noda et al. (1999) und Carreno et al. (2007).

Der Nachteil der Gewichtungsmethode ist das benötigte Vor- oder Expertenwissen für die Bestimmung von $\boldsymbol{w}$, um die Gewichtung der einzelnen Zielfunktionen sinnvoll zu wählen. Ferner ist die Lösung sensitiv gegenüber der Wahl von $\boldsymbol{w}$ (vgl. Sarker et al., 2002). Zusätzlich zeigen Das und Dennis (1997), dass selbst bei korrekter Wahl von $\boldsymbol{w}$, die Lösung nicht durch eine skalare Zielfunktion erreicht werden kann. Dieser Nachteil kann jedoch teilweise durch die dynamische Veränderung der Gewichte während der Optimierung ausgeglichen werden (vgl. Jin et al., 2001).

Weitere aggregierende mehrkriterielle Optimierungmethoden sind etwa die von Bowman (1976) vorgeschlagene *Tchebycheff-Programming-Methode* oder die *ϵ-Constraint-Methode* von Haimes et al. (1971).

Im Allgemeinen können die aggregierenden mehrkriteriellen Optimierungsmethoden zwar im Schnitt zu guten Ergebnissen führen, allerdings nicht in den einzelnen Zielgrößen. Auf diese Weise wird nur eine Lösung erreicht und es werden nur wenige Erkenntnisse über das zugrundeliegende Problem gewonnen (vgl. Horn et al., 1994).

Alternativ können mehrere Läufe zur Optimierung einzelner Ziele durchgeführt werden, während dabei die Einstellungen der anderen Ziele konstant gehalten werden. Dieses Vorgehen fördert allerdings Populationen von Lösungen, die sich zunehmend um einen Zielfunktionswert konzentrieren, sodass die Diversität der Population verloren geht. Weitere potentielle Probleme bei diesem Ansatz sind laut Deb (2001) die mehrfach benötigten Optimierungsläufe. Da die Läufe unabhängig voneinander erfolgen, können keine Synergieeffekte genutzt werden.

Die notwendige a priori Gewichtung der Zielkriterien durch den Anwender führt zwangsläufig zu einer Hierarchie (vgl. Definition 6 in Abschnitt 6.1) in den Zielkriterien. Folglich ist die aggegierende mehrkriterielle Optimierung für eine nichthierarchische Lösung ungeeignet und kommt demnach in dieser Arbeit nicht zur Anwendung.

5.2 Pareto-Optimierung

Im Gegensatz zu den aggregierenden mehrkriteriellen Optimierungsmethoden, die nur durch vielmalige Anwendung einkriterieller Verfahren mit diversen Parametrisierungen verschiedene Lösungen ermitteln können, wird bei der Pareto-Optimierung die Pareto-Front approximiert. Das Ziel der Pareto-Optimierung ist die Approximation der Pareto-Front bestehend aus den ermittelten Kompromisslösungen der mehrkriteriellen Optimierung, um den Zusammenhang zwischen den Zielfunktionen aufzuzeigen. Die Verwendung der Pareto-Optimierung zur Analyse mehrkriterieller Zielfunktionen wurde bei der Variablenselektion, Wissensentdeckung und Generierung von Ensemblen erfolgreicher als aggregierende Verfahren angewendet (vgl. Jin und Sendhoff, 2008). Ferner kann der Anwender auf Basis seiner aktuellen Anforderungen eine Lösung aus der Pareto-Front wählen. Diese a posteriori und interaktive Entscheidungsfindung kann ebenfalls effektiver sein als die a priori Festlegung von Gewichtungsfaktoren oder Randbedingungen bei den aggregierenden mehrkriteriellen Optimierungsmethoden, da dem Anwender bereits mögliche Kompromisslösungen vorgeschlagen werden. Ohne die einzelnen Zielekriterien nach Wichtigkeit a priori ordnen zu müssen, wird bei der Pareto-Optimierung im Gegensatz zur aggregierenden mehrkriteriellen Optimierung eine hierarchielose Optimierung möglich (vgl. Definition 6).

In der Praxis ist die globale Pareto-Front unbekannt, sodass nicht festgestellt werden kann, ob die nicht-dominierten Lösungen des Algorithmus tatsächlich Pareto-optimal sind. Dennoch werden nicht-dominierte Lösungen, die durch einen mehrkriteriellen Pareto-optimalen Optimierungsalgorithmus gefunden werden, häufig als Pareto-optimale Lösungen bezeichnet.

Qualitative Kriterien für die Güte einer Lösungsmenge stellen Konvergenz und Diversität dar. Zum einen sollte die Lösungsmenge auf der Pareto-Front bzw. in deren Nähe liegen (Konvergenz), zum anderen sollte die Lösungsmenge die gesamte Vielfalt der Pareto-Front und nicht nur wenige Bereiche abdecken (Diversität). Die quantitative Erfassung der Güte einer gefundenen Lösungsmenge wird häufig durch das dominierte Hypervolumen, die sogenannte S-Metrik (vgl. Zitzler und Thiele, 1998) beschrieben. Dadurch ist es möglich, sowohl die Lösungsmengen als auch die entsprechenden Algorithmen miteinander zu vergleichen.

Die S-Metrik γ^S misst in Abhängigkeit eines Referenzpunktes $\boldsymbol{r}$ (üblicherweise das Maximum aller Zielfunktionen) das dominierte Hypervolumen. Dieses beschreibt den Bereich des Zielraums, in dem sich Vektoren befinden, die schlechter sind als mindestens ein Mitglied der Lösungsmenge. Das maximale Hypervolumen wird von der Pareto-Front dominiert. Die S-Metrik honoriert neben der Konvergenz zur Pareto-

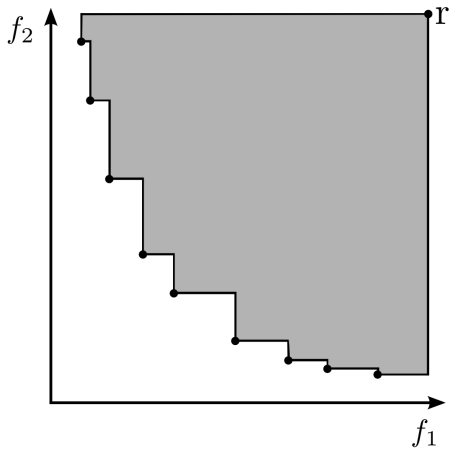


Abbildung 8: Pareto-Front (schwarze Punkte) und dominiertes Hypervolumen (graue Fläche) eines zweikriteriellen Optimierungsproblems mit Referenzpunkt $\boldsymbol{r}$

Front auch die Verteilung der Punkte auf der Front. Formal ist die S-Metrik der Lösungsmenge $\mathcal{M}$ definiert als

$$\gamma^S(\mathcal{M}, \boldsymbol{r}) := \Lambda \left(\bigcup_{\boldsymbol{f}(\boldsymbol{x}) \in \mathcal{M}} \{\boldsymbol{f'}(\boldsymbol{x}) | \boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f'}(\boldsymbol{x}) \prec \boldsymbol{r}\} \right), \qquad \mathcal{M} \subseteq \mathbb{R}^m, \boldsymbol{r} \in \mathbb{R}^m. \quad (5)$$

Dabei beschreibt Λ das Lebesgue-Maß, $\boldsymbol{r}$ den Referenzpunkt, der durch alle Pareto-optimalen Lösungen dominiert werden soll, und $\boldsymbol{f}(\boldsymbol{x})$ den Vektor der Fitnessfunktionen. Abbildung 8 zeigt die nicht-dominierten Lösungen und die Pareto-Front eines zweikriteriellen Optimierungsproblems mit Referenzpunkt $\boldsymbol{r}$. Die graue Fläche stellt den Bereich des Zielraums dar, der durch die Pareto-Front dominiert wird. Das Lebesgue-Maß der grauen Fläche ist die S-Metrik.

Neben der weit verbreiteten S-Metrik (vgl. Beume et al., 2007; Deb et al., 2003) kann ferner als Vergleich von Pareto-Fronten die Anzahl nicht-dominierter Individuen herangezogen werden. Werden aus der Vereiningung zweier Pareto-Fronten die nicht-dominierten Individuen bestimmt, kann die Anzahl der nicht-dominierten Individuen aus den zugrundeliegenden Pareto-Fronten miteinander verglichen und ins Verhältnis gesetzt werden. Dieses Verhältnis wird erstmals in dieser Arbeit definiert und als Dominanzquotient γ^D bezeichnet (vgl. Definition 5).

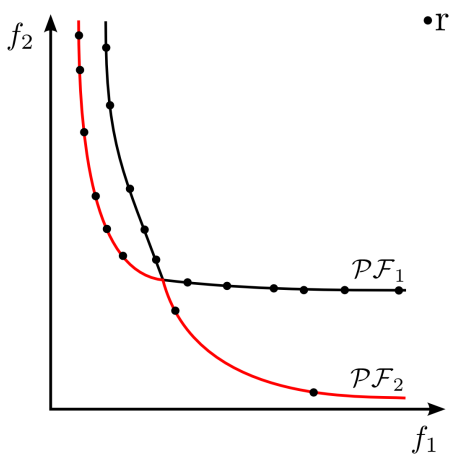


Abbildung 9: Vereinigte Pareto-Front von $\mathcal{PF}_1$ und $\mathcal{PF}_2$ für ein zweikriterielles Optimierungsproblem mit Referenzpunkt $\boldsymbol{r}$

Definition 5 (Dominanzquotient)
Seien $\mathcal{PF}_1$ und $\mathcal{PF}_2$ die Pareto-Fronten eines m-kriteriellen Optimierungsproblems $\boldsymbol{f} : \mathbb{R}^p \to \mathbb{R}^m$ mit $\mathcal{PF}_i = \bigcup_{\boldsymbol{f}(\boldsymbol{x}) \in \mathcal{M}_i} \{\boldsymbol{f'}(\boldsymbol{x}) | \boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f'}(\boldsymbol{x})\}$, $\mathcal{M}_i \subseteq \mathbb{R}^m$, $i = \{1,2\}$, und

$$\mathcal{PF} = \bigcup_{\boldsymbol{f}(\boldsymbol{x}) \in \{\mathcal{PF}_1 \cup \mathcal{PF}_2\}} \{\boldsymbol{f'}(\boldsymbol{x}) | \boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f'}(\boldsymbol{x})\}$$

die vereinigte Pareto-Front von $\mathcal{PF}_1$ und $\mathcal{PF}_2$.
Dann ist der Dominanzquotient γ^D von $\mathcal{PF}_1$ bezüglich $\mathcal{PF}_2$ definiert als

$$\gamma^D(\mathcal{PF}_1, \mathcal{PF}_2) := \frac{|\mathcal{PF}_1 \cap \mathcal{PF}|}{|\mathcal{PF}_2 \cap \mathcal{PF}|} \in \mathbb{R}_0^+.$$

Für $\mathcal{PF}_2 \cap \mathcal{PF} = \varnothing$ gilt $\gamma^D(\mathcal{PF}_1, \mathcal{PF}_2) := \infty$.

Sind etwa mehr Individuen aus der ersten Pareto-Front in der vereinigten Pareto-Front enthalten, macht dies den Vorteil der ersten gegenüber der zweiten Pareto-Front ersichtlich und γ^D ist größer Eins. Abbildung 9 zeigt zwei Pareto-Fronten $\mathcal{PF}_1$ und $\mathcal{PF}_2$ sowie die vereinigte Pareto-Front (rot markiert). Auf dieser sind sechs Beobachtungen von $\mathcal{PF}_1$ und zwei Beobachtungen von $\mathcal{PF}_2$ enthalten, sodass gilt $\gamma^D(\mathcal{PF}_1, \mathcal{PF}_2) = \frac{6}{2} = 3$. Somit besteht ein klarer Vorteil für $\mathcal{PF}_1$, auch wenn die S-Metrik offensichtlich geringer ist. γ^D honoriert also die Anzahl und die Diversität der Punkte auf der Pareto-Front stärker als die S-Metrik. Dies ist vorteilhaft, wenn ein Anwender a posteriori aus der Menge der Pareto-optimalen Lösungen eine auswählen möchte und mehr Möglichkeiten zur Verfügung stehen.

Sollen mehr als zwei Pareto-Fronten miteinander verglichen werden können die Anzahlen der nicht-dominierten Individuen in der gemeinsamen Pareto-Front gegenübergestellt werden. Die Pareto-Front, aus der die meisten Individuen der gemein-

samen Pareto-Front stammen, ist dann die beste. Alternativ kann eine der Pareto-Fronten als Vergleichsfront, also als $\mathcal{PF}_2$ in Definition 5, fungieren, so dass γ^D stets bezüglich dieser Front bestimmt wird. Dadurch sind die Dominanzquotienten auch bei mehr als zwei Pareto-Fronten untereinander vergleichbar. Weitere alternative Maßzahlen zur Quantifizierung der Lösungsgüte tragen etwa Deb (2001) zusammen.

Anstrengungen zur Lösung mehrkriterieller Probleme durch Pareto-basierte Optimierungsmethoden waren im letzten Jahrzehnt erfolgreich. Allerdings sind viele Verfahren für die Pareto-Optimierung empfindlich bezüglich der Form der Pareto-Front und funktionieren nicht bei konkaven oder unzusammenhängenden Pareto-Fronten (vgl. Coello Coello, 2006). Andere Algorithmen benötigen die Differenzierbarkeit der Zielfunktionen und der Nebenbedingungen. Im Gegensatz hierzu arbeiten EMOAs (vgl. Kapitel 5.3) simultan mit einer Menge möglicher Lösungen, sodass parallel mehrere Lösungen der Pareto-Menge pro Lauf generiert werden (vgl. Emmanouilidis et al., 2000). Außerdem können EMOAs mit konkaven und nicht zusammenhängenden Pareto-Fronten umgehen, sodass sie großen Zuspruch bei der mehrkriteriellen Optimierung finden (vgl. Zitzler et al., 2000) und auch in dieser Arbeit zur Anwendung kommen. Gute Übersichtsarbeiten hierzu existieren etwa von Coello Coello et al. (2007), Jin (2006) und Deb (2001).

5.3 Evolutionäre mehrkriterielle Optimierungsalgorithmen

Die Besonderheit der evolutionären mehrkriteriellen Optimierungsalgorithmen (EMOAs) gegenüber den einkriteriellen EAs (vgl. Abschnitt 4) liegt in der Fitnessbestimmung und Selektion der Individuen. Alle anderen Operatoren und die grundlegende Idee bleiben erhalten. Bei der Eltern- und bei der Umweltselektion müssen in einem EMOA die Individuen aufgrund aller Fitnessfunktionen ausgewählt werden. Hierfür bestehen diverse Ansätze, von denen zwei populäre Algorithmen mit Bitstring-Repräsentation zur Anwendung kommen. Dies ist zum einen der NSGA-II (vgl. Abschnitt 5.3.1), der bekannteste EMOA (vgl. Nebro und Durillo, 2009), und zum anderen der SMS-EMOA (vgl. Abschnitt 5.3.2). In Abschnitt 5.3.3 werden diverse Abbruchkriterien für EMOAs vorgestellt.

EMOAs mit Bitstring-Repräsentation wurden bereits in vielen Wissenschaftsbereichen in Form von mehrkriteriellen *Wrapper*-Ansätzen (vgl. Kohavi und John, 1997) zur Klassifikation angewandt (vgl. Vatolkin et al., 2012; Sanchez-Faddeev et al., 2012; Soyel et al., 2011; Mersmann et al., 2011; Garcia-Nieto et al., 2009; Castillo Tapia und Coello Coello, 2007; Hamdani et al., 2007; Oliveira et al., 2003). Der in dieser Arbeit entwickelte und untersuchte mehrkriteriellen *Wrapper*-Ansatz wird in Abschnitt 6.1 vorgestellt.

5.3.1 NSGA-II

Srinivas und Deb (1994) entwickelten basierend auf einer nicht-dominierten Rangfolgenprozedur den *Nondominated Sorting Genetic Algorithm* (NSGA) als einen der ersten GAs zur Lösung mehrkriterieller Optimierungsprobleme. Seine hohe Rechenkomplexität für die nicht-dominierte Sortierung von $\mathcal{O}(m\mu^3)$ mit m Zielfunktionen und einer Populationsgröße μ sowie der fehlende Elitärismus (vgl. Abschnitt 4.1.6), wurde unter anderem lange Zeit kritisiert. Als Verbesserung des NSGA präsentierten Deb et al. (2002) den NSGA-II mit einer schnellen nicht-dominierten Sortierungsprozedur und geringerer Rechenkomplexität, einer elitären Selektion und den parameterfreien *Crowding*-Distanz-Operator zur Diversitätserhaltung.

Der NSGA-II ist ein generationsbezogener EMOA, der aus der aktuellen Population P_g der Größe μ unter Verwendung verschiedener Operatoren eine Nachkommenpopulation Q_g der Größe λ erstellt. Danach werden die beiden Populationen zu einer Population R_g der Größe $\mu + \lambda$ kombiniert. Die Lösungen der neuen Population $P_{g+1} \subset R_g$ der Größe μ werden durch die nicht-dominierte Sortierprozedur aus R_g ermittelt.

Diese nicht-dominierte Sortierprozedur erstellt eine Rangfolge, bei der die nicht-dominierten Lösungen aus R_g den Rang 1 erhalten, der Front $\mathcal{F}_1$ zugeordnet und aus R_g entfernt werden. Die nicht-dominierten Lösungen der restlichen Population erhalten Rang 2, bilden die Front $\mathcal{F}_2$ und werden wiederum aus der Population R_g entfernt. Dies geschieht sukzessive bis jedes Individuum einer Front $\mathcal{F}_i$ zugeordnet wurde (vgl. Deb, 2001). Die neue Population P_{g+1} wird durch Auswahl der besten nicht-dominierten Fronten erstellt. Enthält die letzte erlaubte Front $\mathcal{F}_l$ mehr Lösungen als noch offene Positionen in P_{g+1}, werden die Lösungen aus $\mathcal{F}_l$ in die nächste Population übernommen, die die Diversität aller ausgewählten Lösungen maximiert. Hierfür wird für jede Lösung die *Crowding*-Distanz als euklidischer Abstand zu den benachbarten Lösungen im Zielraum betrachtet. Die *Crowding*-Distanz für einen Punkt p_j ist ein Schätzer für die Größe des maximalen Quarders um p_j ohne eine andere Lösungen der Population zu enthalten. Abbildung 10 verdeutlicht die Berechnung der *Crowding*-Distanz für eine gegebene Lösung p_j (vgl. Deb et al., 2002). Die wesentlichen Schritte des NSGA-II sind in Algorithmus 6 aufgeführt.

Nebro und Durillo (2009) schlagen eine *Steady-State*-Version des NSGA-II vor, bei der pro Generation nur ein Individuum erzeugt und sofort in den evolutionären Zyklus integriert wird. Da die Erstellung der Rangfolge sowie die *Crowding*-Prozedur nach jeder Generierung eines neuen Individuums erfolgt, verlängert sich die Berechnungsdauer des *Steady-State*-Ansatzes im Vergleich zum klassischen NSGA-II. Nebro und Durillo (2009) zeigen, dass mittels des *Steady-State* NSGA-II die Konvergenz zu

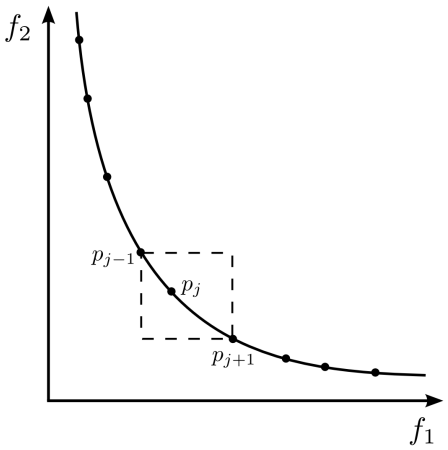


Abbildung 10: *Crowding*-Distanz für die Lösung p_j auf der Pareto-Front

der Pareto-Front im Vergleich zum NSGA-II in den meisten *Benchmark*-Problemen bessere Ergebnisse liefert und schneller konvergiert.

Beispiele für die Anwendung des NSGA-II liefern etwa Soyel et al. (2011), Garcia-Nieto et al. (2009), Hamdani et al. (2007) und Oliveira et al. (2003). Soyel et al. (2011) kombinieren den NSGA-II mit dem Fisher-Kriterium, einem Gütemaß für die Trennbarkeit der Klassen, das auf die Pareto-optimalen Variablenteilmengen angewendet wird. Der Vorteil dieses Verfahrens gegenüber der direkten Optimierung von Fehlklassifikationsrate und Variablenanzahl wird begründet mit der höheren Stabilität des Fisher-Kriteriums gegenüber der Fehlklassifikationsrate, vor allem wenn die Größe und die Anzahl der Validierungssets klein ist (vgl. Dubuisson et al., 2002). Garcia-Nieto et al. (2009) wenden einen NSGA-II-*Wrapper* mit interner *Support Vector Machine* in der Krebsdiagnose an. Als Fitnessfunktionen wählen sie entgegen der gängigen Anwendung der Fehlklassifikationsrate die Sensitivität und die Spezifität. Hamdani et al. (2007) verwenden einen *Wrapper*-Ansatz aus NSGA-II und 1-nächste-Nachbarn zur Optimierung der Fehlklassifikationsrate und der Variablenanzahl. Oliveira et al. (2003) verwenden einen *Wrapper*-Ansatz aus NSGA-II und Neuronalen Netzen. Da der NSGA-II bereits häufig erfolgreich auf Klassifikationsprobleme angewendet wurde, dient er auch in dieser Arbeit als EMOA im mehrkriteriellen *Wrapper*-Ansatz (vgl. Abschnitt 6.1) und dessen nicht-dominierte Sortierprozedur sowie die *Crowding*-Distanz zur Umweltselektion in NHEMOtree (vgl. Abschnitt 6.2).

5.3.2 SMS-EMOA

Als anerkanntes Qualitätsmaß für die Pareto-Front-Approximation ist es naheliegend, eine Maximierung der S-Metrik (vgl. Gleichung (5) in Abschnitt 5.2) bereits im Optimierungsprozess anzustreben. Dies realisieren Emmerich et al. (2005) mit

Algorithmus 6: NSGA-II

Definition:

P_g Population in Generation g

Q_g Nachkommenpopulation in Generation g

Eingabe:

$\mu \in \mathbb{N}$ Größe der Population P_g

$\lambda \in \mathbb{N}$ Größe der Nachkommenpopulation Q_g

δ Abbruchkriterium

Ausgabe:

Finale Lösungspopulation P_{g+1}

Initialisierung:

$g = 0$;

Erzeuge Population P_0 der Größe μ durch Initialisierungsmethode;

NSGA-II:

while *Abbruchkriterium δ nicht erfüllt* **do**

Generierung von Q_g basierend auf P_g unter Verwendung von Selektions-, Rekombinations- und Mutationsoperatoren;

Erstelle $R_g = P_g \cup Q_g$ der Größe $\mu + \lambda$; `/* Gemeinsame Population */`

Erstelle $P_{g+1} \subset R_g$ der Größe μ; `/* Durch nicht-dominierte Sortierung */`
`/* und Crowding-Distanz-Maximierung */`

$g = g + 1$;

end

dem *S-Metric Selection Evolutionary Multiobjective Optimization Algorithm* (SMS-EMOA). Im SMS-EMOA kommt zunächst die bekannte nicht-dominierte Sortierung des NSGA-II zum Tragen, bevor die S-Metrik als sekundäres Selektionskriterium angewendet wird. Das S-Metrik-basierte Selektionskriterium bildet die partiell geordneten Zielfunktionsvektoren auf skalare Fitnesswerte ab, sodass eine vollständige Ordnung der Lösungen und eine Selektion der besten möglich ist. Ein neues Individuum gelangt somit nur in die Population, falls es die S-Metrik erhöht.

Der Ablauf des SMS-EMOA ist in Algorithmus 7 dargestellt. Als initiale Population werden μ Individuen zufällig generiert. Aus dieser Population wird mit Hilfe randomisierter Variationsoperatoren (vgl. Abschnitt 4.2) genau ein Nachkomme ($\lambda = 1$) erzeugt. Im Vergleich zu einer $(\mu + \lambda)$-Selektion mit $\lambda > 1$ müssen weniger S-Metrik-Werte berechnet werden, um eine optimal zusammengesetzte Population zur Maximierung der S-Metriken zu erhalten. Aufgrund der *Steady-State*-Selektion (vgl. Abschnitt 4.1.6) ist der Selektionsdruck beim SMS-EMOA gering (vgl. Emmerich et al., 2005).

Aus der Menge der aktuellen Population und dem einen Nachkommen werden alle dominierten Individuen in der Menge $\mathcal{D}$ zusammengefasst. Falls $\mathcal{D}$ nicht leer ist, wird das primäre Selektionskriterium, das auf der Pareto-Dominanz basiert, angewendet.

Das Individuum mit der größten Dominanzzahl z, das von den meisten anderen Individuen dominiert wird, wird aussortiert. Existieren keine dominierten Individuen ($\mathcal{D} = \varnothing$), kommt das sekundäre Selektionskriterium zum Tragen, bei dem das Individuum mit der kleinsten S-Metrik entfernt wird.

Algorithmus 7: SMS-EMOA

Definition:

P_g Population in Generation g
z Dominanzzahl
γ^S S-Metrik mit Referenzpunkt $\boldsymbol{r}$

Eingabe:

$\mu \in \mathbb{N}$ Größe der Population P_g
δ Abbruchkriterium

Ausgabe:

Finale Lösungspopulation P_{g+1}

Initialisierung:

$g = 0$
Erzeuge Population P_0 der Größe μ durch Initialisierungsmethode

SMS-EMOA:

while *Abbruchkriterium δ nicht erfüllt* **do**
  Generierung eines Nachkommens λ basierend auf P_g unter Verwendung von Selektions-, Rekombinations- und Mutationsoperatoren
  Bestimme $\mathcal{D} \subseteq P_g \cup \lambda$ `/* Menge der dominierten Individuen */`
  if $\mathcal{D} \neq \varnothing$ `/* Primäres Selektionskriterium */`
  then
    $a^* = \arg\max_{a \in \mathcal{D}} [z(a, P_g \cup \lambda)]$ `/* Individuum mit max. Dominanzzahl z */`
  end
  else
    if $\mathcal{D} = \varnothing$ `/* Sekundäres Selektionskriterium */`
    then
      $a^* = \arg\min_{a \in \{P_g \cup \lambda\}} [\gamma^S(a, \boldsymbol{r})]$ `/* Individuum mit min. S-Metrik */`
    end
  end
  Erstelle $P_{g+1} = \{P_g \cup \lambda\} \backslash \{a^*\}$ `/* Neue Population ohne a* */`
  $g = g + 1$
end

Die Motivation dieser zweistufigen Selektion ist eine geringere Laufzeitkomplexität im Vergleich zum alleinigen Verwenden der S-Metrik. Außerdem wird so eine gute Verteilung der Lösungen auf der ersten nicht-dominierten Front $\mathcal{F}_1$ angestrebt. Das sekundäre Selektionskriterium dient als Diversitätsmaß, das die Verteilung der Punkte optimiert, den Fortschritt in Richtung der Pareto-Front jedoch kaum berücksichtigt. Da der SMS-EMOA immer die Population mit der höchsten S-Metrik

auswählt, ist die S-Metrik der Population über die Generationen monoton steigend, sodass die Population gegen die Pareto-Front strebt.

Der SMS-EMOA fand bereits in verschiedenen Wissenschaftsbereichen Anwendung, etwa zur Entwicklung neuer Proteinliganden (vgl. Sanchez-Faddeev et al., 2012), als Hybrid-Algorithmus mit Random Forests zur Charakterisierung von Problemklassen (vgl. Mersmann et al., 2011) oder in Verbindung mit verschiedenen Klassifikationsalgorithmen (C4.5, Random Forest, Naive Bayes und Support Vector Machines) zur Erkennung von Musikinstrumenten (vgl. Vatolkin et al., 2012).

Die Ähnlichkeit des NSGA-II und des SMS-EMOA ist leicht zu erkennen. Der Hauptunterschied beider Prozeduren ist die $\mu + \lambda$-Selektion des NSGA-II und die *Steady-State*-Selektion des SMS-EMOA, sowie das unterschiedliche Ranking der Lösungen auf derselben Front $\mathcal{F}_i$. Im NSGA-II sorgt die *Crowding*-Distanz für die gleichmäßige Verteilung der Lösungspunkte auf der Pareto-Front, sodass der Wert einer Lösung von dessen Nachbarn abhängt und nicht direkt von dessen eigener Position. Im Gegensatz dazu werden die Lösungspunkte im SMS-EMOA so verteilt, dass die S-Metrik maximiert wird. Beispiele zeigen, dass gute Kompromisslösungen im *Knick* einer konvexen Pareto-Front bei der Verwendung eines SMS-EMOA bessere Ränge zugeteilt bekommen als beim NSGA-II (vgl. Emmerich et al., 2005). Ferner funktioniert der SMS-EMOA auch bei hoch-dimensionalen Zielfunktionen. Der NSGA-II ist hingegen bereits bei fünf Optimierungskriterien nicht mehr einsetzbar, da durch die *Crowding*-Distanz Extremlösungen bevorzugt werden und keine sinnvolle Pareto-Front approximiert wird (vgl. Wagner et al., 2007).

5.3.3 Abbruchkriterien

Durch Abbruchkriterien wird versucht, eine Balance zwischen Rechenressourcen und der Qualität der Approximation zu finden. Zur Herleitung für geeignete Abbruchkriterien können empirische Richtlinien oder die mathematische Konvergenztheorie herangezogen werden (vgl. Wagner und Trautmann, 2010). Der hohen Komplexität der EMOAs ist es jedoch geschuldet, dass für sie nur wenige asymptotische Konvergenztheorien bestehen. So zeigen etwa Rudolph und Agapie (2000) basierend auf der Markovketten-Theorie, dass in speziellen Anwendungsfällen elitäre EMOAs in einer endlichen Anzahl von Funktionsevaluationen in endlichen Suchräumen zur wahren Pareto-Front konvergieren können.

Eine Übersicht verschiedener Ansätze zur Konvergenzanalyse in EMOAs tragen etwa Trautmann et al. (2009) zusammen. Demnach existieren neben der sehr rechenintensiven Offline-Konvergenzanalyse (vgl. Trautmann et al., 2008) und den häufig verwendeten Generationenanzahl- oder Laufzeit-abhängigen Abbruchkriterien

zwei Hauptansätze zur Online-Konvergenzanalyse. Diese sind zum einen statistisch inspirierte Techniken wie die *Online Convergence Detection* (OCD) von Wagner et al. (2009), das *Least Squares Stopping Criterion* von Guerrero et al. (2009), das *Dominance-based Stability Measure* von Bui et al. (2009) oder das Abbruchkriterium von Rudenko und Schoenauer (2004), das auf der Standardabweichung der *Crowding*-Distanz basiert. Zum anderen dienen auf Kalman-Filtern basierende Online-Methoden der Schätzung des Zustands eines dynamischen Systems (vgl. Kalman, 1960). Ein Beispiel hierfür ist etwa das MGBM-Kriterium von Martí et al. (2007).

Online-Methoden messen die Verbesserung eines oder mehrerer Gütemaße innerhalb eines Intervalls und stoppen den Lauf, falls die Verbesserung kleiner als ein vorgegebener Grenzwert ist. Problematisch hierbei ist jedoch die Auswahl des Grenzwertes, sowie die Entscheidung bei Funktionen mit sehr kleinen Änderungen. Ferner kann ein Algorithmus bei Erreichen einer bestimmten Lösungsqualität gestoppt werden, wobei das Festlegen eines geeigneten Qualitätslimits schwierig ist. Das allgemeine Problem der Abbruchkriterien jedoch ist, dass sie nicht die Konvergenz sondern die Stagnation eines EMOAs ermitteln und die Nähe der Lösung zum Optimum nicht sichergestellt ist (vgl. Rudenko und Schoenauer, 2004). In dieser Arbeit kommt neben der Beschränkung der Generationenanzahl das OCD-Abbruchkriterium in NHEMOtree zur Anwendung (vgl. Abschnitt 6.2), da für diesen Ansatz im Gegensatz zu den meisten anderen Verfahren keine visuelle Einschätzung des Konvergenzverhaltens notwendig ist und empirisch belegte Parametereinstellungen vorgeschlagen werden (vgl. Wagner et al., 2011).

Die *Online Convergence Detection* (OCD) von Wagner et al. (2009) zielt darauf ab, während des Laufes die Konvergenz des EMOAs zu erkennen und dann zu stoppen. Bei der OCD wird mit einem Varianz- und einem Trendkriterium die Konvergenz ermittelt. Das Varianz-Kriterium bricht den EMOA ab, wenn die Varianz eines Gütemaßes γ über v Generationen hinweg ($var(\gamma; v)$) unterhalb eines festgelegten Wertes L fällt. Dies wird mittels eines Ein-Stichproben χ^2-Tests (vgl. Hartung et al., 2002) mit der Nullhypothese $H_0 : \ var(\gamma; v) \geq L$ getestet. Beim Trendkriterium wird eine Kleinste-Quadrate-Anpassung für ein lineares Regressionsmodell mit Steigung β durchgeführt und mittels eines t-Tests (vgl. Hartung et al., 2002) getestet, ob kein signifikanter Trend eines Gütemaßes über die letzten Generationen hinweg besteht ($H_0 : \ \beta = 0$). Liegt der p-Wert über $\alpha = 0{,}05$, kann die Nullhypothese nicht abgelehnt werden und der EMOA bricht ab.

Die Grundstruktur der OCD ist in Algorithmus 8 dargestellt. Für die aktuelle Generation werden alle n Gütemaße ermittelt und einzeln gegen die Alternativhypothese $H_1 : var(\gamma_j; v) < L$ getestet. Aufgrund der multiplen Tests erfolgt eine Bonferroni-Korrektur (vgl. Bland und Altman, 1995) des Signifikanzniveaus α, sodass jeder Test

ein individuelles Signifikanzniveau von $\frac{\alpha}{n}$ besitzt und in jeder Generation das globale Signifikanzniveau eingehalten wird. Danach werden die linearen Regressionsmodelle angepasst und die p-Werte der Trendtests für alle n Gütemaße berechnet. Der EMOA bricht ab, wenn für alle n Gütemaße in zwei aufeinanderfolgenden Generationen entweder die p-Werte der Varianztests kleiner oder gleich $\frac{\alpha}{n}$ sind, die p-Werte der Trendtests größer α oder aber die maximale Generationenanzahl erreicht wurde.

Algorithmus 8: Online Convergence Detection

Definition:

$p\chi^2(\gamma_j; v)$ p-Wert des Varianztests mit $H_0 : \ var(\gamma_j; v) \geq L_j$ bzgl. γ_j über die letzten v Generationen

$pReg(\gamma_j; v)$ p-Werte des Trendtests mit $H_0 : \beta = 0$ bezüglich γ_j über die letzten v Generationen

Eingabe:

g	Aktuelle Generationenanzahl
G	Maximale Generationenanzahl
v	Anzahl vorangegangener Generationen für Vergleiche
$\gamma_1, \ldots, \gamma_n$	Vektoren n verschiedener Gütemaße der Länge v
L_j	Maximales Varianzlimit für den Varianztest bzgl. γ_j
α	Signifikanzniveau

Online Convergence Detection:

if $g \geq G$ **then**
  Ausgabe: Algorithmus ist konvergiert!
else
  for $j = 1 \ldots n$ **do**
    if $(p\chi^2(\gamma_j; v) > \frac{\alpha}{n} \vee p\chi^2(\gamma_j; v-1) > \frac{\alpha}{n})$
    $\wedge \ (pReg(\gamma_j; v) \leq \alpha \vee pReg(\gamma_j; v-1) \leq \alpha)$ **then**
      Ausgabe: Algorithmus ist nicht konvergiert!
    end
  end
end
Ausgabe: Algorithmus ist konvergiert!

Im Allgemeinen werden bei der OCD der additive ϵ-Indikator (vgl. Zitzler et al., 2003), der R2 Indikator (vgl. Hansen und Jaszkiewicz, 1998) und die S-Metrik (vgl. Gleichung (5) in Abschnitt 5.2) als Gütemaße betrachtet. Die OCD terminiert genau dann, wenn der Trend- oder der Varianztest gleichzeitig bezüglich aller Gütemaße auf Konvergenz hindeutet. Wagner und Trautmann (2010) entwickeln eine Erweiterung der OCD (OCD-HV), bei der als Metrik lediglich die S-Metrik zur Konvergenzentdeckung betrachtet wird. Aufgrund der monotonen Erhöhung der S-Metrik im $(\mu + 1)$-SMS-EMOA entdeckt der Trendtest eine Konvergenz, wenn keine Verbesserung in den letzten v Generationen erfolgte. In diesen Fällen führt auch der Varianztest zum Abbruch, sodass auf den Trendtest verzichtet werden kann. Demnach terminiert der OCD-HV, wenn laut χ^2-Varianztest die Varianz der

S-Metrik signifikant unterhalb der a priori festgelegten Varianzschranke liegt. Als Parameter für den OCD-HV schlagen Wagner und Trautmann (2010) ein Varianzlimit von $L = 10^{-10}$ mit $v = 14$ vorangegangenen Generationen für Vergleiche vor. Die OCD-HV mit den entsprechenden Parametern wird auch in dieser Arbeit zur Konvergenzanalyse eingesetzt.

5.4 Fazit

Dieses Kapitel erläutert die grundlegenden Konzepte der mehrkriteriellen Optimierung, bei der gleichzeitig zwei oder mehr sich widersprechende Zielfunktionen optimiert werden. Die Verfahren zur Lösung mehrkriterieller Optimierungsprobleme lassen sich in aggregierende und Pareto-basierte Ansätze unterteilen. Erstere reduzieren das mehrkriterielle Optimierungsproblem durch Transformation auf ein einkriterielles Optimierungsproblem. Hierfür ist a priori-Wissen erforderlich und nur eine Lösung wird gefunden. Dies ist insbesondere bei konfliktären Zielen, für die keine einzelne optimale Lösung für alle Zielfunktionen existiert, problematisch.

Im Gegensatz dazu wird bei der Pareto-Optimierung die Pareto-Front bestehend aus den Kompromisslösungen der mehrkriteriellen Optimierung ermittelt. Hieraus kann der Anwender a posteriori eine mögliche Kompromisslösung zur Realisierung auswählen und erhält bessere Erkenntnisse über das zugrundeliegende Optimierungsproblem. Insgesamt können Pareto-basierte mehrkriterielle Ansätze erfolgreicher zur Lösung von Optimierungsproblemen angewendet werden als aggregierende Ansätze (vgl. Jin und Sendhoff, 2008).

Für die Lösung mehrkriterieller Optimierungsprobleme stellen mehrkriterielle *Wrapper*-Ansätze basierend auf EMOAs mit Bitstring-Repräsentation das übliche Vorgehen bei einer mehrkriteriellen Optimierung mit integrierter Variablenselektion dar. In diesem Kapitel wurden die bekannten EMOAs NSGA-II, *Steady-State* NSGA-II und SMS-EMOA, die in dieser Arbeit zur Anwendung kommen, vorgestellt. Ferner wird das besonders für EMOAs geeignete Abbruchkriterium von Wagner et al. (2009), die OCD, beschrieben. Dies wird ebenfalls in NHEMOtree eingesetzt.

6 Entwicklung neuer Verfahren und Operatoren zur Lösung mehrkriterieller Optimierungsprobleme

Der naheliegende Ansatz zur Lösung mehrkriterieller Optimierungsprobleme insbesondere von kostensensitiven Klasifikationsproblemen ist ein mehrkriterieller *Wrapper*-Ansatz bestehend aus einem mehrkriteriellen EMOA mit Bitstring-Repräsentation und einem umhüllten Klassifikationsalgorithmus. Der in dieser Arbeit untersuchte mehrkriterielle *Wrapper*-Ansatz, der erstmals CART als umhüllten Klassifikationsalgorithmus verwendet, wird in Abschnitt 6.1 vorgestellt. Nachfolgend wird der nicht-hierarchische evolutionäre Optimierungsalgorithmus mit Baum-Repräsentation (NHEMOtree) entwickelt (vgl. Abschnitt 6.2). Dieser optimiert die verschiedenen Fitnessfunktionen im Vergleich zum mehrkriteriellen *Wrapper*-Ansatz gleichzeitig. NHEMOtree hat den Vorteil, dass er durch seine Repräsentation auf einen internen Klassifikationsalgorithmus verzichten kann und dadurch eine nicht-hierarchische unverzerrte mehrkriterielle Optimierung erfolgt. In Abschnitt 6.3 werden mehrkriterielle Variablenwichtigkeitsmaße (VIMs) für Entscheidungsbäume entwickelt, die unter Berücksichtigung aller zu optimierenden Zielfunktionen die Wichtigkeit der Variablen beschreiben. Diese werden in NHEMOtree für den hier entwickelten VIM-basierten Rekombinationsoperator verwendet (vgl. Abschnitt 6.4). In Abschnitt 6.5 wird NHEMOtree mit lokaler Optimierung der Cutoff-Werte anstelle der standardmäßigen Cutoff-Optimierung durch Gauß-Mutation vorgestellt. Abschließend werden in Abschnitt 6.6 die in dieser Arbeit entwickelten Verfahren und Operatoren zusammengefasst.

6.1 Mehrkriterieller *Wrapper*-Ansatz

Häufig werden EMOAs zur mehrkriteriellen Optimierung in Form von mehrkriteriellen *Wrapper*-Ansätzen verwendet (vgl. Kohavi und John, 1997). Die Individuen eines EMOAs werden dabei als Bitstring s codiert, sodass jedes Individuum in der Population eine Variablenteilmenge $\mathcal{X}_s \subset \mathcal{X}$ darstellt. Der EMOA dient der Variablenselektion und der umhüllte Klassifikationsalgorithmus, in dieser Arbeit CART, erstellt auf Basis der ausgewählten Variablenteilmenge $\mathcal{X}_s$ ein Klassifikationsmodell, das durch ein Gütemaß für die Vorhersagegenauigkeit beschrieben wird. Auf diese Weise wird zunächst nur ein externes Zielkriterium optimiert. Erst nach der Konstruktion des Klassifikationsmodells werden weitere Zielkriterien, wie etwa die finanziellen Kosten oder die Anzahl der verwendeten Variablen im Klassifikationsmodell, ermittelt. Damit besteht im mehrkriteriellen *Wrapper*-Ansatz eine eindeutige Hierarchie der zu optimierenden Zielkriterien mit Vorteil für die Vorhersagegüte

(vgl. Definition 6), sodass auf diese Weise keine nicht-hierarchischen Lösungen des mehrkriteriellen Optimierungsproblems gefunden werden. Die Bevorzugung eines Zielkriteriums führt zu einer Verschiebung der mehrkriteriellen Lösungen in Richtung des bevorzugten Kriteriums. Dadurch ist die Lösungsvielfalt beschränkt und die approximierte Pareto-Front verzerrt.

Definition 6 (Hierarchie in der mehrkriteriellen Optimierung)
Bei gegebenen Variablen $\mathcal{X} \subset \mathbb{R}^p$ *sei* $\min_{\boldsymbol{x} \in \mathcal{X}} \boldsymbol{f}(\boldsymbol{x})$ *mit* $\boldsymbol{f} = (f_1, \ldots, f_m) : \mathcal{X} \to \mathbb{R}^m$, $m \geq 2$, *ein mehrkriterielles Optimierungsproblem. Mögliche Lösungen für das Optimierungsproblem werden durch einen Algorithmus* $A(\mathcal{X}; \gamma_1, \ldots, \gamma_l)$ *mit internen Zielkriterien* γ_j, $j = 1, \ldots, l$, *erstellt. Die* γ_j *sind abhängig vom verwendeten Algorithmus* $A(\mathcal{X})$, *wie etwa die Gini-Wichtigkeit bei CART. Die möglichen Lösungen werden durch die externen Zielkriterien* f_i, $i = 1, \ldots, m$, *wie die Vorhersagegüte, bewertet.*

Steht nicht zu jedem f_i *ein internes Zielkriterium* γ_j *in Relation, haben die* f_i *ohne entsprechendes* γ_j *keinen Einfluss auf die Erstellung der Lösung und können nur nachrangig die Güte der Lösung bewerten.*

Damit besteht eine Hierarchie zwischen den externen Zielkriterien $f_1, \ldots, f_m$.

Um die Defizite dieser nicht-hierarchischen Lösung aufzuzeigen, werden in dieser Arbeit verschiedene mehrkriterielle *Wrapper* untersucht und dienen als Basis zum Vergleich des in Abschnitt 6.2 entwickelten NHEMOtrees. Die untersuchten mehrkriteriellen *Wrapper* zur mehrkriteriellen Optimierung von Vorhersagegüte und Kosten basieren auf verschiedenen EMOAs mit Bitstring-Repräsentation (NSGA-II, *Steady-State* NSGA-II oder SMS-EMOA), die CART als internen Klassifikationsalgorithmus umhüllen. Die mehrkriteriellen *Wrapper* liefern im Gegensatz zur alleinigen Anwendung von CART eine Fülle unvergleichbarer Lösungen und nicht nur die vermeintlich beste. So kann a posteriori je nach Fragestellung entschieden werden, ob eher günstigere Lösungen oder Lösungen mit einer geringeren Fehlklassifikationsrate von Interesse sind.

Abbildung 11 zeigt das Ablaufschema des entwickelten mehrkriteriellen *Wrapper*-Ansatzes. Zunächst wird eine initiale Population bestehens aus μ Individuen als Bitstrings codiert erstellt. Im mehrkriteriellen *Wrapper*-Ansatz wird die Ausgangspopulation zufällig initialisiert. Die Initialisierung der Bits der Individuen in der Ausgangspopulation stellt ein Bernoulli-Experiment mit Erfolgswahrscheinlichkeit $\pi = P(\text{Bit} = 1)$, der sogenannten Input-Rate, dar (vgl. Abschnitt 4.2). Die verschiedenen untersuchten Input-Raten π sowie die anderen Parameter wurden durch ein Latin-Hypercube-Design ermittelt und sind in Tabelle 6 in Abschnitt 8.2 dargestellt.

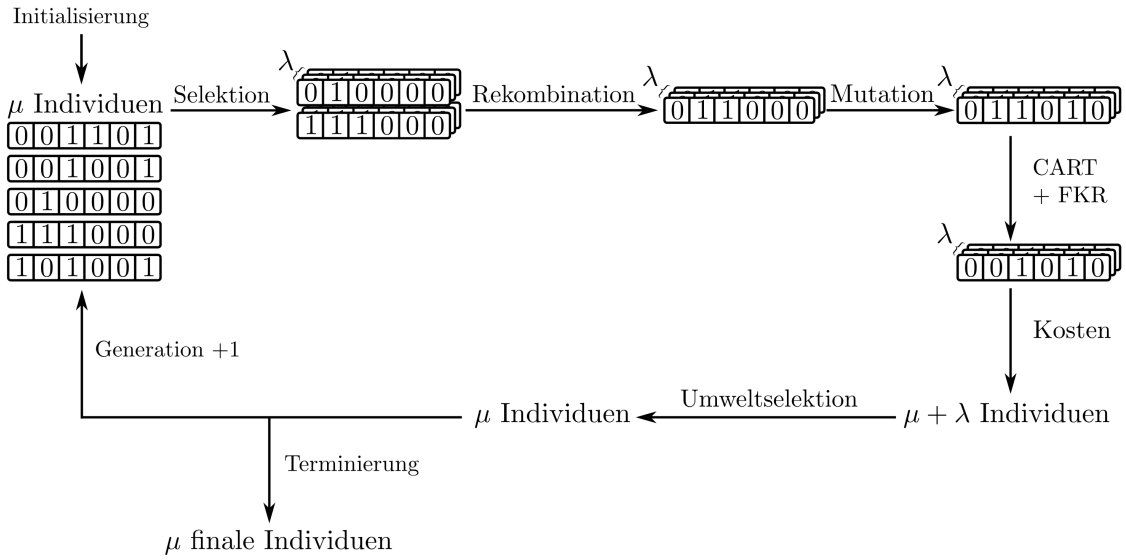


Abbildung 11: Ablaufschema des mehrkriteriellen *Wrapper*-Ansatzes

Zur Elternselektion wird die in EAs weit verbreitete Turnierselektion ohne Zurücklegen verwendet (vgl. Abschnitt 4.1.4). Jedes Elternteil entstammt aus einem Turnier der Größe $\tau = 2$. Bei identischen Eltern wird erneut aus einem Turnier selektiert.

Nach der Elternselektion werden die beiden ausgewählten Individuen einer Ein-Punkt-Rekombination unterzogen. Mit einer festen Rekombinationsrate p_r wird die Ein-Punkt-Rekombination für EAs mit Bitstring-Repräsentation basierend auf zwei Eltern, bei der einer der beiden Nachkommen überlebt, durchgeführt (vgl. Abschnitt 4.2). In Abbildung 11 erfolgt die Rekombination zwischen dem zweiten und dritten Bit. Das obere Individuum überlebt.

Bei der anschließenden Mutation wird der Standard-Mutationsoperator für EAs mit Bitstring-Repräsentation verwendet (vgl. Abschnitt 4.2). Dabei wird jedes Bit des Chromosoms mit einer festen Mutationsrate p_m mutiert. In Abbildung 11 erfolgt eine Mutation für das fünfte Bit. Die verwendeten Rekombinations- und Mutationsraten sind neben der maximal durchzuführenden Generationsanzahl und der Populationsgröße in Tabelle 6 in Abschnitt 8.2 dargestellt.

Auf der entsprechenden Variablenteilmenge der Nachkommen erstellt CART mit integriertem Pruning (vgl. Kapitel 2) den optimalen binären Entscheidungsbaum und liefert die fünffach kreuzvalidierte Fehlklassifikationsrate. In Abbildung 11 hat CART Variable 2 nicht in den Entscheidungsbaum aufgenommen. Folglich belaufen sich die Kosten des Individuums nur auf die Summe der Einzelkosten für Variable 3 und 5. Auf diese Weise werden λ Nachkommen generiert.

Danach erfolgt die $(\mu + \lambda)$-Umwelselektion auf Basis der μ Individuen der aktuellen Generation sowie der λ Nachkommen und liefert die neue Population der Größe μ.

Je EMOA (NSGA-II, *Steady-State* NSGA-II, SMS-EMOA) erfolgt die Umweltselektion leicht verschieden (vgl. Abschnitt 5.3). Danach beginnt entweder eine neue Generation oder der Algorithmus ist konvergiert und die finale Population gefunden.

In dieser Arbeit werden mit dem mehrkriteriellen *Wrapper*-Ansatz neben der Fehlklassifikationsrate verschiedene zweite Fitnessfunktionen untersucht. Entweder werden die finanziellen Kosten der Variablen oder die Anzahl der verschiedenen Variablen im Entscheidungsbaum als zweite Fitnessfunktion genutzt. Ebenfalls wird mit dem mehrkriteriellen *Wrapper*-Ansatz ein dreikriterielles Optimierungsproblem mit den Fitnessfunktionen Fehlklassifikationsrate, finanzielle Kosten und Variablenanzahl betrachtet.

6.2 NHEMOtree

NHEMOtree (nicht-hierarchischer evolutionärer mehrkriterieller Optimierungsalgorithmus mit Baum-Repräsentation) erstellt die Individuen direkt durch einen EMOA mit Baum-Repräsentation, sodass zur Optimierung der Individuen kein interner Klassifikationsalgorithmus benötigt wird. Auf diese Weise können die Individuen unmittelbar bezüglich aller Zielfunktionen optimiert und bewertet werden ohne eine Hierarchie oder Bevorzugung einer der Optimierungsfunktionen. Mit NHEMOtree werden also anders als beim mehrkriteriellen *Wrapper*-Ansatz nicht-hierarchische evolutionäre mehrkriteriell optimierte Entscheidungsbäume erstellt.

In NHEMOtree werden die Individuen als binäre Entscheidungsbäume (vgl. Kapitel 2) dargestellt. Jeder Knoten setzt sich aus einer Variable mit entsprechendem Cutoff, der die Daten im Knoten bezüglich der entsprechenden Variablen und des Cutoffs teilt, zusammen. Eine einfache Mehrheitswahl teilt die am häufigsten vertretene Klasse der Beobachtungen in einem Blatt dem entsprechenden Blatt zu. Die erste Fitnessfunktion beschreibt die Fehlklassifikationsrate, die entsprechend Gleichung (4) in Abschnitt 4.3 berechnet wird. Die zweite Fitnessfunktion beschreibt die Kosten aller Variablen des jeweiligen Entscheidungsbaums.

Der Ablauf des NHEMOtrees ist in Abbildung 12 dargestellt, wobei je Operator verschiedene Varianten zur Auswahl stehen. Zu Beginn wird die initiale Population P_0 durch die gängige *ramped-half-and-half*-Methode (vgl. Abschnitt 4.3) erstellt, wenn kein apriori-Wissen in die Optimierung einfließen soll. Als Alternative kann eine a priori optimierte initiale Population verwendet werden, wie etwa die finale Population einer mehrkriteriellen *Wrapper*-Lösung.

Für die Entwicklung der nächsten Population werden die Individuen per Turnier- oder Winkler-Selektion (vgl. Abschnitt 4.1.4) ausgewählt. Die bereits in vielen EAs

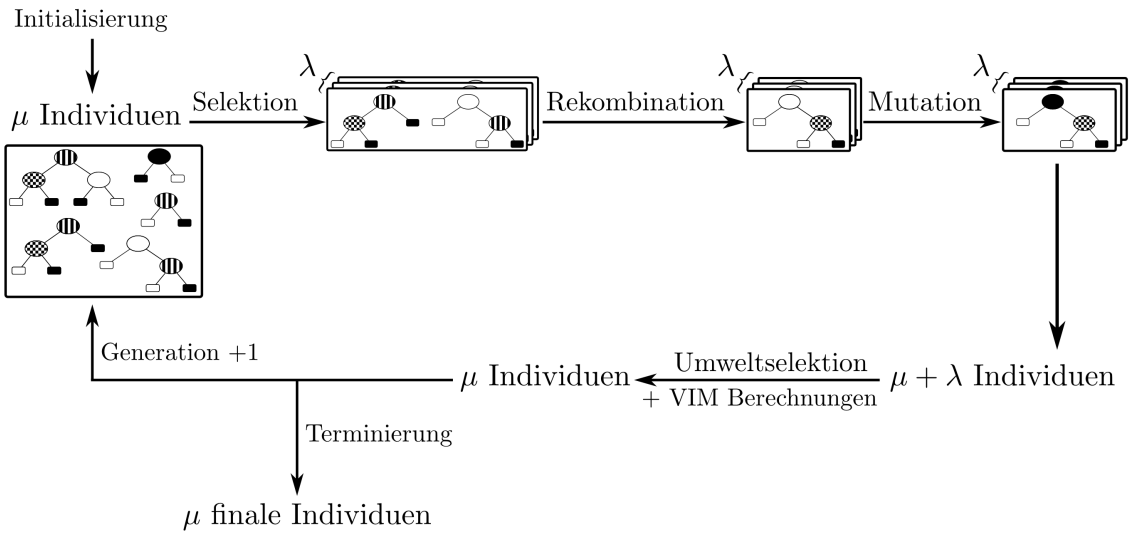


Abbildung 12: Ablaufschema des NHEMOtrees

erfolgreich angewendete Turnierselektion mit Zurücklegen wird mit der von Banzhaf et al. (1998) empfohlenen Turniergröße $\tau = 4$ verwendet. Die Winkler-Selektion wird aufgrund ihrer erfolgreichen Anwendung bei EAs mit Baum-Repräsentation untersucht (vgl. Winkler et al., 2009).

Je zwei Eltern werden mit einer festen Rekombinationsrate $p_r \in [0,1]$ rekombiniert, sodass zwei Nachkommen entstehen, von denen ein Nachkomme zufällig in die Nachkommenpopulation übergeht und sich mit Wahrscheinlichkeit $p_m \in [0,1]$ einer Mutation unterzieht. Erfolgt keine Rekombination der Eltern, überlebt zufällig ein Elternteil, das ebenfalls mit einer festen Mutationsrate p_m mutiert wird und einen Nachkommen darstellt. Auf diese Weise werden bei λ Durchführungen λ Nachkommen erstellt.

In NHEMOtree werden verschiedene Rekombinationsoperatoren untersucht, da die Rekombination als vorherrschender Variationsoperator in EAs mit Baum-Repräsentation als Grund für dessen Effektivität angesehen wird (vgl. Abschnitt 4.3). Mit verbesserten Rekombinationsoperatoren wird versucht, auch verbesserte Ergebnisse im Vergleich zum Standard-Rekombinationsoperator der EAs mit Baum-Repräsentation (X_S) zu erzielen. In NHEMOtree wird X_S mit der Brut-Rekombination (X_B) und der von Tackett (1994) propagierten Brutgröße $\rho = 4$, sowie mit der tiefenabhängigen Ein-Punkt-Rekombination nach Poli und Langdon (1998a) (X_P) verglichen. Außerdem wird ein VIM-basierter Rekombinationsoperator auf Grundlage von X_S und mehrkriteriellen Variablenwichtigkeitsmaßen (X_{VIM}) entwickelt (vgl. Abschnitt 6.4) und NHEMOtree dadurch nochmals verbessert (vgl. Abschnitt 8.3 und Kapitel 9). Die Rekombinationsoperatoren werden stets in unterschiedlichen Läufen ausgeführt.

Überschreiten die Nachkommen nach der Rekombination die maximale Knotenanzahl, erfolgt zunächst eine Fitness-basierte Hoist-Mutation (vgl. Abschnitt 4.3), die den Subbaum mit der höchsten Fitness als neuen Elternbaum auswählt. Ist die Fitness des besten Subbaums schlechter als die Fitness des Gesamtbaums erfolgt keine Hoist-Mutation. In diesem Fall erfolgt bis zur Einhaltung der maximalen Knotenanzahl sukzessive eine zufällige Subbaum-Löschung (vgl. Abschnitt 4.3).

In NHEMOtree existiert neben der allgemeinen Rekombinationsrate auch eine allgemeine Mutationsrate, die zu Beginn jedes Laufs festgelegt wird. Allerdings werden in NHEMOtree im Gegensatz zur Rekombination verschiedene Mutationsoperatoren in einem Lauf genutzt. Denn sowohl die Verwendung verschiedener Mutationsoperatoren (vgl. Kinnear Jr., 1993; Kraft et al., 1994; Angeline, 1996) als auch deren Kombination (vgl. Chellapilla, 1997; Harries und Smith, 1997) ist bei anderen EMOAs mit Baum-Repräsentation erfolgreich. Die Auswahl des Operators zur Mutation der Baumstruktur und der Knotenvariablen (vgl. Abschnitt 4.3) des jeweiligen Nachkommens erfolgt dabei zufällig (vgl. Poli et al., 2008). Soll jedoch ein Individuum, das aus nur einem Wurzelknoten besteht, mutiert werden, erfolgt in NHEMOtree keine Hoist-Mutation, Subbaum-Löschung oder Subbaum-Mutation. In diesem Fall wird eine der verbleibenden Mutationsarten verwendet.

Die Mutation der Cutoff-Werte erfolgt in NHEMOtree standardmäßig mit der Gauß-Mutation und mit $\alpha = 0{,}85$ analog zu Schwefel (1977) (vgl. Abschnitt 4.3). Eine andere Möglichkeit zur Adaption der Variablen-Cutoffs eines Baums durch die Gauß-Mutation ist die Mutation der Cutoffs durch die Verwendung von lokalen Optimierungsverfahren (vgl. Poli et al., 2008). In NHEMOtree wird hierzu die lokale Optimierung der Cutoff-Werte basierend sowohl auf der Fehlklassifikationsrate als auch auf der Gini-Wichtigkeit genutzt (vgl. Abschnitt 6.5).

Durch die Variationsoperatoren können Individuen mit redundanten Knoten entstehen. Nichtterminale Knoten sind dann redundant, wenn sie nur einen Nachkommen besitzen. Basierend auf den Ergebnissen von Haruyama und Zhao (2002) werden in NHEMOtree redundante Knoten während eines Laufs entfernt und durch ein Blatt ersetzt.

Nach der Rekombination und Mutation werden die Fitnesswerte (fünffach kreuzvalidierte Fehlklassifikationsrate und Vorhersagekosten) anhand der Entscheidungsbäume ermittelt. Auf Basis der Fitnesswerte erfolgt nach dem klassischen Vorbild der EMOAs mit Baum-Repräsentation eine generationsbasierte Umweltselektion (vgl. Banzhaf et al., 1998). Dabei wird mit der nicht-dominierten Sortierung und der *Crowding*-Distanz wie im NSGA-II (vgl. Abschnitt 5.3.1) aus der aktuellen Population und der Nachkommenpopulation die neue Population bestehend aus μ Individuen ausgewählt. Für die Individuen der neuen Population werden die

Variablenwichtigkeiten bestimmt (vgl. Abschnitt 6.3), anhand derer der Standard-Rekombinationsoperator X_S erweitert wird (vgl. Abschnitt 6.4). Danach beginnt entweder eine neue Generation oder der Algorithmus terminiert und die finale Population ist gefunden. Als Abbruchkriterium kann in NHEMOtree sowohl die maximale Generationenanzahl als auch die OCD-HV (vgl. Abschnitt 5.3.3) gewählt werden.

Die initialen Parametereinstellungen werden in Anlehnung an die Vorschläge von Koza (1992), Banzhaf et al. (1998), Zhao (2007) und Winkler et al. (2009) in NHEMOtree analog zum mehrkriteriellen *Wrapper*-Ansatz je Optimierungsproblem durch ein Latin-Hypercube-Design (vgl. Abschnitt A im Anhang) ermittelt und sind in Tabelle 11 in Abschnitt 8.3 dargestellt.

NHEMOtree leidet wie jeder andere EMOA mit Baum-Repräsentation unter *Bloating*. Jedoch verhindert die mehrkriterielle Optimierung der Fehlklassifikationsrate und der Vorhersagekosten, sowie eine Limitierung der maximalen Knotenanzahl im Baum das exzessive Wachstum der Entscheidungsbäume. Denn je größer der Baum wird und je mehr Knoten mit verschiedenen Variablen enthalten sind, desto „teurer“ wird die Klassifikation. Außerdem wird in NHEMOtree durch Anwendung der Hoist-Mutation und der Subbaum-Löschung (vgl. Abschnitt 4.3) die Baumgröße regelmäßig reduziert.

6.3 Mehrkriterielle Variablenwichtigkeitsmaße

In Kapitel 3 wurden gängige VIMs für einkriterielle, Baum-basierte Optimierungsverfahren vorgestellt. Diese einkriteriellen VIMs beschreiben häufig die Differenz der Güte eines Modells bei Betrachtung mit und ohne der entsprechenden Variable. Die Übertragung dieser VIMs auf mehrkriterielle Optimierungsprobleme mit einer oder mehreren weiteren Zielfunktionen, die etwa abhängig von der Baumgröße oder den Baumkosten optimiert werden sollen, funktioniert im Allgemeinen nicht. Der kleinere Baum hätte stets einen Vorteil gegenüber dem größeren. Denn der kleinere Baum ist in der entsprechenden Zielfunktion immer besser als der größere, sodass der größere Baum niemals echt besser als der kleinere sein kann. Diesen generellen Nachteil besitzen alle bisher vorgestellten einkriteriellen VIMs.

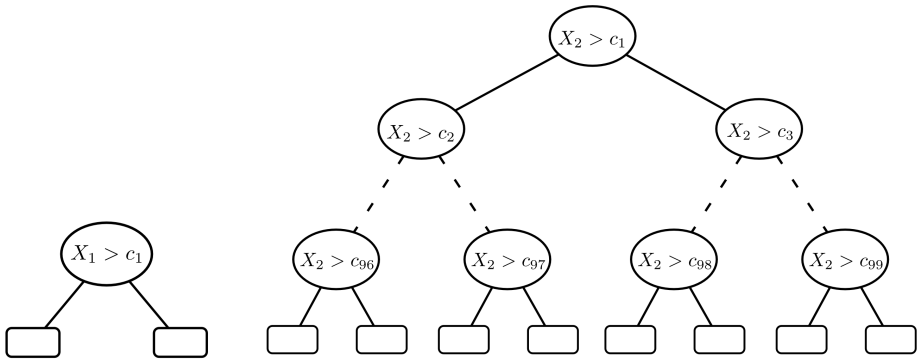
Mehrkriterielle VIMs sollten hingegen alle zu optimierenden Zielfunktionen berücksichtigen. Nach meinem Wissen existieren jedoch keine mehrkriteriellen VIMs für EMOAs mit Baum-Repräsentation. Daher werden in diesem Abschnitt entsprechende mehrkriterielle VIMs für Entscheidungsbäume entwickelt. Diese mehrkriteriellen VIMs basieren auf Gedankenexperimenten und werden in NHEMOtree für eine verbesserte Rekombination verwendet (vgl. Abschnitt 6.4).

Eine Möglichkeit für mehrkriterielle VIMs ist die Verwendung der Variablenhäufigkeit in den Individuen nach der Umweltselektion über alle Generationen des EMOAs hinweg. Die Variablenwichtigkeit wird jeweils an den Individuen, die die Umweltselektion überleben, ermittelt. Da die Umweltselektion auf allen Zielkriterien erfolgt, wird die Variablenwichtigkeit somit bezüglich aller Optimierungskriterien bestimmt.

Im Gegensatz zu den VIMs von Kooperberg und Ruczinski (2005) in der *Monte Carlo Logic Regression*, die zur Modellierung von Einzelnukleotid-Polymorphismen (engl. *single-nucleotide polymorphisms*, SNPs) entwickelt wurden und bei der je Modell ein SNP nur einmal enthalten sein kann, ist bei Entscheidungsbäumen das Vorkommen einer Variablen in mehreren Knoten mit verschiedenen Cutoff-Werten möglich. Damit stellt sich die Frage, ob für die Variablenwichtigkeit entscheidend ist, ob eine Variable einfach oder mehrfach in einem Entscheidungsbaum enthalten ist. Zur Beantwortung wird nachfolgendes Gedankenexperiment durchgeführt.

Es soll die Variablenwichtigkeit zweier Variablen X_1 und X_2 in 100 Entscheidungsbäumen mit identischer Fehlklassifikationsrate bestimmt werden. Bestünden 99 Bäume nur aus einem Knoten mit X_1 (vgl. Abbildung 13(a)) und bestünde ein Baum aus 99 Knoten stets mit X_2 aber immer anderen Cutoff-Werten $c_1, \ldots, c_{99}$ (vgl. Abbildung 13(b)), so liegt die einfache relative Häufigkeit, bei der lediglich überprüft wird, ob eine Variable im Modell enthalten ist, von X_1 über alle Bäume bei 0,99 und für X_2 bei 0,01. Die relative Häufigkeit unter Berücksichtigung der Variablenhäufigkeit je Baum wäre für X_1 und X_2 hingegen identisch mit

$$h(X_1) = h(X_2) = \frac{99 \text{ Variablen}}{(99 \times 1 + 1 \times 99) \text{ Gesamtvariablen in allen 100 Bäumen}} = 0{,}5.$$



(a) Baum 1 (b) Baum 2

Abbildung 13: Motivation des Variablenwichtigkeitsmaß *Einfache absolute Häufigkeit* (VIM_1)

Demnach ist bei Berechnung der Variablenwichtigkeit entscheidend, wie mit mehrmaligem Vorkommen einer Variablen in einem Baum umgegangen werden soll. Das Gedankenexperiment zeigt, dass die Berücksichtigung der Variablenhäufigkeit innerhalb eines Baums zu einer Überrepräsentation dieser Variablen im VIM führen kann und somit Situationen existieren, in denen auf die Mehrfachzählung der Variablen innerhalb eines Baums verzichtet werden sollte.

Einfache absolute Häufigkeit ($\mathbf{VIM_1}$)

Basierend auf dem Gedankenexperiment beschreibt das erste mehrkriterielle VIM als *Einfache absolute Häufigkeit* (VIM_1) die Variablenwichtigkeit für die Variable $X_j, j = 1, \ldots, p$, in Baum t mit

$$\mathrm{VIM}_1(X_j; t) = \begin{cases} 1 & \text{falls } X_j \in t \\ 0 & \text{falls } X_j \notin t \end{cases} . \tag{6}$$

Einfache relative Häufigkeit ($\mathbf{VIM_2}$)

Ferner sind wahrscheinlich Variablen, die gute Ergebnisse in kleineren Modellen erzielen, wichtiger als Variablen, die gleich gute Ergebnisse in großen Modellen erzielen. Denn im Allgemeinen werden für die bessere Interpretierbarkeit und Generalisierbarkeit kleine Modelle gegenüber großen bevorzugt. Diese Hypothese soll hier durch ein VIM berücksichtigt werden, das nicht das absolute Vorkommen einer Variable in einem Entscheidungsbaum betrachtet, sondern die Variablenwichtigkeit auf die Baumgröße bezieht. Deshalb gilt für die *Einfache relative Häufigkeit* (VIM_2)

$$\mathrm{VIM}_2(X_j; t) = \begin{cases} \frac{1}{n_t} & \text{falls } X_j \in t \\ 0 & \text{falls } X_j \notin t \end{cases} , \tag{7}$$

wobei n_t die Anzahl aller Variablen X_j, $j = 1, \ldots, p$, in Baum t beschreibt.

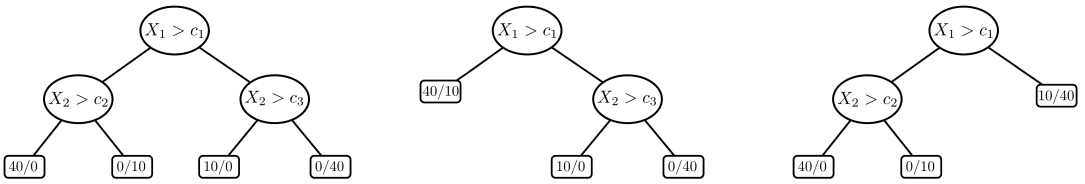
Relative Häufigkeit ($\mathbf{VIM_3}$)

Für den Baum in Abbildung 14(a) hingegen ist $\mathrm{VIM}_2(X_1) = \mathrm{VIM}_2(X_2) = \frac{1}{2}$ mit einer Fehlklassifikationsrate von 0%. Würde einer der Knoten mit Variable X_2 fehlen (vgl. Abbildung 14(b) und 14(c)), ergäben sich Fehlklassifikationsraten von 10% bei gleicher Variablenwichtigkeit. Die Kosten für die kleineren Bäume blieben konstant oder reduzierten sich je nach Art der Kostenberechnung. Im vorliegenden Fall ist es im Gegensatz zum Gedankenexperiment wichtig, dass beide X_2-Knoten im Baum vorhanden sind. Die *Relative Häufigkeit* (VIM_3) honoriert dementsprechend das

mehrfache Vorkommen der Variable X_2 im Baum. Es gilt

$$\text{VIM}_3(X_j;t) = \begin{cases} \frac{N_t(X_j)}{N_t} & \text{falls } X_j \in t \\ 0 & \text{falls } X_j \notin t \end{cases} \tag{8}$$

mit $N_t(X_j)$ als Anzahl der Knoten mit Variable X_j in Baum t und $N_t = \sum_{j=1}^{p} N_t(X_j)$ als Anzahl aller Knoten im Baum. Bei diesem Variablenwichtigkeitsmaß ist es somit im Gegensatz zu VIM_1 und VIM_2 ein Unterschied, ob eine Variable einfach oder mehrfach in einem Entscheidungsbaum auftritt.



(a) 0% FKR (b) 10% FKR (c) 10% FKR

Abbildung 14: Motivation des Variablenwichtigkeitsmaß *Relative Häufigkeit* (VIM_3)

Gewichtete relative Häufigkeiten (VIM_4 und VIM_5)

Neben der relativen Häufigkeit der Variablen in einem Baum ist auch die Position der Variablen im Baum entscheidend für deren Wichtigkeit (vgl. Strobl et al., 2007b). Variablen mit geringerer Baumtiefe, d.h. näher am Wurzelknoten, könnten wichtiger sein als solche kurz vor den Blättern, wenn nur noch wenige Beobachtungen aufgespaltet werden. Tiefe Knoten, vor allem in großen Bäumen, erhöhen ferner die Gefahr der Überanpassung. Somit erscheint eine Gewichtung der Variablenhäufigkeit bezüglich der Position im Baum als sinnvoll. Die Modellierung der Knotenposition erfolgt hier auf zwei Weisen.

Die *Linear gewichtete relative Häufigkeit* (VIM_4) gewichtet die Variablen bezüglich ihrer Position im Baum linear. Der Wurzelknoten erhält dabei das Gewicht $G_l(\vartheta_t(\nu);t) = 1$ und die weiteren Knoten ν bezüglich ihrer Tiefe $\vartheta_t(\nu)$ in Baum t das Gewicht $G_l(\vartheta_t(\nu);t) = 1 - \frac{\vartheta_t(\nu)-1}{\Theta_t}$, wobei Θ_t die maximale Baumtiefe des Baums t beschreibt.

Die *Exponentiell gewichtete relative Häufigkeit* (VIM_5) lässt die Variablenwichtigkeit mit größerer Baumtiefe exponentiell abfallen und gewichtet den Wurzelknoten ebenfalls mit $G_e(\vartheta_t(\nu);t) = 1$. Im Allgemeinen gilt für die Gewichte $G_e(\vartheta_t(\nu);t) =$

$\frac{1}{2^{\vartheta_t(\nu)-1}}$, wobei $\vartheta_t(\nu)$ der Knotentiefe im Baum t entspricht. Es gilt

$$\mathrm{VIM}_4(X_j;t) = \begin{cases} \sum\limits_{\nu \in \mathcal{N}_t(X_j)} \frac{G_l(\vartheta_t(\nu);t)}{N_t} & \text{falls } X_j \in t \\ 0 & \text{falls } X_j \notin t \end{cases} \tag{9}$$

und

$$\mathrm{VIM}_5(X_j;t) = \begin{cases} \sum\limits_{\nu \in \mathcal{N}_t(X_j)} \frac{G_e(\vartheta_t(\nu);t)}{N_t} & \text{falls } X_j \in t \\ 0 & \text{falls } X_j \notin t \end{cases} \tag{10}$$

mit $\mathcal{N}_t(X_j)$ als Menge der Knoten mit Variable X_j in Baum t, N_t als Anzahl aller Knoten in Baum t und den entsprechenden linearen und exponentiellen Gewichten G_l und G_e der Variable X_j je Knoten ν im Baum t.

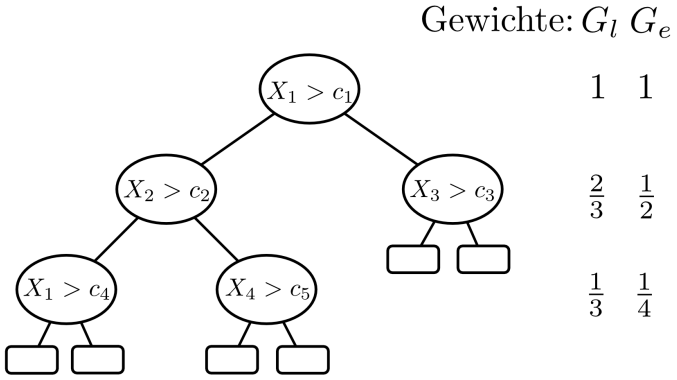


Abbildung 15: Beispiel zur Berechnung der mehrkriteriellen Variablenwichtigkeitsmaße

Zur besseren Verdeutlichung der fünf mehrkriteriellen VIMs ist in Abbildung 15 exemplarisch ein Baum mit maximaler Baumtiefe 3 und entsprechenden linearen Gewichten $G_l = (1, \frac{2}{3}, \frac{1}{3})$ und exponentiellen Gewichten $G_e = (1, \frac{1}{2}, \frac{1}{4})$ dargestellt. Die entsprechenden VIMs der einzelnen Variablen X_1 bis X_4 zeigt Tabelle 3.

VIM	X_1	X_2	X_3	X_4
VIM_1	1	1	1	1
VIM_2	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$
VIM_3	$\frac{2}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$
VIM_4	$\frac{1}{5} \times (1 + \frac{1}{3})$	$\frac{1}{5} \times \frac{2}{3}$	$\frac{1}{5} \times \frac{2}{3}$	$\frac{1}{5} \times \frac{1}{3}$
VIM_5	$\frac{1}{5} \times (1 + \frac{1}{4})$	$\frac{1}{5} \times \frac{1}{2}$	$\frac{1}{5} \times \frac{1}{2}$	$\frac{1}{5} \times \frac{1}{4}$

Tabelle 3: Variablenwichtigkeitsmaße (VIMs) entsprechend zu Abbildung 15

Permutierte Fehlerfreiheit (VIM_6)

Ferner wird die *Permutierte Fehlerfreiheit* (VIM_6) als populäres einkriterielles VIM in Random Forests nach der Umweltselektion berechnet (vgl. Gleichung (2) in Abschnitt 3.4). Im Vergleich zu den anderen VIMs liegt hierbei nicht die Variablenhäufigkeit der Wichtigkeit zugrunde, sondern die Änderung der Fehlklassifikationsrate bei Permutation der Variable im entsprechenden Knoten. Diese Fokussierung auf die Zielfunktion der Vorhersagegüte stellt eine besondere Gewichtung dieses einen Ziels dar und könnte zu einer Verzerrung der Ergebnisse führen.

Zur besseren Übersicht der verwendeten VIMs sind diese in Tabelle 4 zusammengefasst.

VIM	Bezeichnung	Beschreibung
1	Einfache absolute VH	Absolute Variablenhäufigkeit im Baum ohne Berücksichtigung der Baumgröße und des Mehrfachvorkommens einer Variable.
2	Einfache relative VH	Relative Variablenhäufigkeit im Baum bezüglich der Baumgröße ohne Berücksichtigung des Mehrfachvorkommens.
3	Relative VH	Relative Variablenhäufigkeit im Baum bezüglich der Baumgröße sowie des Mehrfachvorkommens.
4	Linear gewichtete VH	Gewichtete relative Variablenhäufigkeit mit linearer Abnahme der Variablenwichtigkeit mit zunehmender Entfernung vom Wurzelknoten.
5	Exponentiell gewichtete VH	Gewichtete relative Variablenhäufigkeit mit exponentieller Abnahme der Variablenwichtigkeit mit zunehmender Entfernung vom Wurzelknoten.
6	Permutierte Fehlerfreiheit	Vergleich des Variablen-Vorhersagewertes basierend auf veränderten und ursprünglichen Daten.

VH: Variablenhäufigkeit

Tabelle 4: Mehrkriterielle Variablenwichtigkeitsmaße in NHEMOtree

6.4 VIM-basierter Rekombinationsoperator in NHEMOtree

Der VIM-basierte Rekombinationsoperator (X_{VIM}) erweitert X_S unter Zuhilfenahme der mehrkriteriellen VIMs aus Abschnitt 6.3. Das Ziel der Adaption von X_S ist die Beschleunigung der Generierung guter Lösungen. Im klassischen Sinne werden bei der Adaption in EAs aus dem bisherigen Ablauf des Algorithmus Schlussfolgerungen auf die Parametereinstellungen für zukünftige Schritte gezogen. Hier werden jedoch nicht die Parametereinstellungen adaptiert, sondern die Auswahl der Rekombinati-

onsknoten wird auf Basis der VIMs optimiert. Nach dem Motto „Gutes weitergeben“ haben Knoten mit wichtigen Variablen höhere Chancen, bei der Auswahl als Rekombinationspunkt zu fungieren. Basierend auf dem VIM der Splittingvariable werden die Variablen sortiert und ihnen ihr Rang zugeordnet. Der Rang des VIMs bezüglich der Variable dividiert durch die Summe aller Ränge im Baum ergibt die Auswahlwahrscheinlichkeit $p_x \in [0,1]$ des oder der Knoten mit der entsprechenden Variable für die Rekombination. Für die Auswahlwahrscheinlichkeit der Variable X aus dem Baum t gilt mit jeweils einem der vorgestellten VIM_i, $i = 1,\ldots,6$,

$$p_x(\text{VIM}_i(X;t)) = \frac{rg(\text{VIM}_i(X;t))}{\sum_{X \in t} rg(\text{VIM}_i(X;t))}.$$

Existieren mehrere Knoten im Baum mit der entsprechenden Variable wird per Zufall einer der in Frage kommenden Knoten als Rekombinationspunkt ausgewählt. Tabelle 5 zeigt beispielhaft die Auswahlwahrscheinlichkeiten der Variablen für die Rekombinationsadaption des Baums aus Abbildung 15 mit VIM_4 (vgl. Tabelle 3 in Abschnitt 6.3).

Variable	X_1	X_2	X_3	X_4	
VIM_4	$\frac{4}{15}$	$\frac{2}{15}$	$\frac{2}{15}$	$\frac{1}{15}$	
Rang	4	2,5	2,5	1	Σ 10
Auswahlwsk.	0,4	0,25	0,25	0,1	

Tabelle 5: Auswahlwahrscheinlichkeiten der Variablen für den Rekombinationspunkt im Baum aus Abbildung 15 mit VIM_4

Abbildung 16 zeigt die Funktionsweise von X_{VIM}. Die Auswahlwahrscheinlichkeiten des linken Elternbaums wurden bereits berechnet (vgl. Tabelle 5). Beim rechten Elternbaum wird nur der interne Knoten rekombiniert, da per Definition Wurzelknoten nicht als Rekombinationspunkte zur Verfügung stehen. Mit einer Wahrscheinlichkeit von 40% wird dann der Knoten mit „$X_1 > c_4$“ im linken Elternbaum als Rekombinationsknoten ausgewählt, sodass mit einer Wahrscheinlichkeit von 40% das linke Nachkommenpaar erstellt wird. Das rechte Nachkommenpaar mit entsprechenden Rekombinationspunkten wird mit einer Wahrscheinlichkeit von 25% erstellt. Neben den aufgezeigten Nachkommenpaaren sind noch zwei weitere möglich, in dem der Knoten mit „$X_3 > c_3$“ oder der Knoten mit „$X_4 > c_5$“ des ersten Elternteils mit dem internen Knoten des zweiten Elternsteils getauscht werden.

6.5 NHEMOtree mit lokaler Optimierung der Cutoff-Werte

Während EAs mit Baum-Repräsentation gut die Baumstruktur optimieren, sind sie laut Hunter (2002) weniger gut für die Optimierung der Koeffizienten im Baum geeignet. Daher wird in dieser Arbeit untersucht, welchen Effekt eine lokale Cutoff-Optimierung nach dem Vorbild von CART in NHEMOtree im Vergleich zu der Cutoff-Mutation basierend auf der Gauß-Mutation mit sich bringt. Sukzessive ausgehend vom Wurzelknoten wird der optimale Cutoff je Knoten für den Gesamtbaum basierend auf einem Qualitätsmaß ermittelt.

Breiman et al. (1998) und Loh und Shih (1997) diskutieren, welches Qualitätsmaß zur Cutoff-Auswahl geeignet ist. Auch wenn die Fehlklassifikationsrate ein intuitives Maß für die Baumbewertung darstellt, ist es jedoch als Splittingkriterium nicht gut geeignet. Denn die Fehlklassifikationsrate kann für alle Splits Null sein, sodass keine Auswahl erfolgen kann oder aber zwei Splits gleich gut erscheinen, obwohl diese es in Wahrheit nicht sind. Die Gini-Wichtigkeit (vgl. Gleichung (1) in Abschnitt 2.1) ist hingegen besser geeignet, da sie reinere Knoten bevorzugt (vgl. Hastie et al., 2009). Dennoch werden hier beide Qualitätsmaße explorativ angewendet und miteinander verglichen. Die Wahl des optimalen Cutoffs je Knoten basiert entweder auf der Reduzierung der Fehlklassifikationsrate des Gesamtbaums oder auf der Gini-Wichtigkeit. Der Cutoff-Wahl liegt dabei stets eine fünffachen Kreuzvalidierung zu Grunde.

Die Variabilität des NHEMOtrees wird durch diese lokale Cutoff-Optimierung eingeschränkt und bringt eine gewisse *Greediness* zurück, da die Cutoff-Wahl stets von den oberen Knoten abhängt. Allerdings ist die lokale Optimierung nicht derart greedy wie CART, da neben den Cutoffs weder die Baumstruktur noch die Knotenvariablen lokal optimiert werden.

Außerdem widerspricht die integrierte lokale Optimierung der Idee der Metaheuristik. Metaheuristiken leiten die Optimierung mit dem Ziel, den Suchraum zu erkunden, um optimale Lösungen zu finden. Sie optimieren ein Problem durch iteratives Ausprobieren, um eine Kandidatenlösung zu verbessern (vgl. Mucherino und Seref, 2009; Blum und Roli, 2003). Im Gegensatz dazu ist die vollständige Iteration über alle Möglichkeiten wie bei der Cutoff-Wahl in CART wenig elegant und wahrscheinlich zeitaufwendiger als die Verwendung der Gauß-Mutation. Nichtsdestotrotz wird der Nutzen der lokalen Cutoff-Optimierung in Abschnitt 8.3.7 untersucht.

6.6 Fazit

In diesem Kapitel wurden ein mehrkriterieller *Wrapper*-Ansatz mit CART als umhüllten Klassifikationsalgoritmus und NHEMOtree zur mehrkriteriellen Optimierung vorgestellt. Beide Verfahren erstellen mehrkriterielle Entscheidungsbäume, wobei NHEMOtree im Gegensatz zum mehrkriteriellen *Wrapper*-Ansatz ohne Hierarchie in den Zielkriterien operiert. NHEMOtree erstellt direkt durch einen EMOA mit Baum-Repräsentation binäre Entscheidungsbäume und benötigt keinen internen Klassifikationsalgorithmus zur Bewertung der Individuen. Die Optimierung der verschiedenen Zielfunktionen erfolgt gleichzeitig, sodass keine Hierarchie der Optimierungsfunktionen besteht und tatsächlich eine nicht-hierarchische mehrkriterielle Optimierung erfolgt.

Ferner wurden erstmals mehrkriterielle VIMs für EMOAs mit Baum-Repräsentation entwickelt. Diese bewerten die Variablenhäufigkeit in den Individuen nach der Umweltselektion des EMOAs. Zwei mehrkriterielle VIMs (VIM_4 und VIM_5) berücksichtigen außerdem die Variablenposition im Baum. Da in EAs mit Baum-Repräsentation der Rekombinationsoperator als Ursache für die erfolgreiche Erstellung von *Building Block*s angesehen wird und sich darauf ihre Effektivität begründet (vgl. Abschnitt 4.3), wurde hier der Standard-Rekombinationsoperator X_S adaptiert, um bessere *Building Block*s und somit auch bessere Pareto-Front-Approximationen zu generieren. Dies resultierte in dem neu entwickelten VIM-basierten Rekombinationsoperator X_{VIM}, der X_S unter Zuhilfenahme der mehrkriteriellen VIMs erweitert. Die Auswahlwahrscheinlichkeit der Knoten für die Rekombination ist in X_{VIM} abhängig von dem VIM der Knotenvariable. Knoten mit wichtigen Variablen haben höhere Chancen, als Rekombinationspunkt zu fungieren.

Neben der standardmäßigen Cutoff-Wahl durch die Gauß-Mutation wurde in NHEMOtree auch die lokale Optimierung der Variablen-Cutoffs in den Knoten eingeführt, da EAs mit Baum-Repräsentation die Koeffizienten im Baum weniger gut optimieren können. Diese zusätzliche Optimierung erfolgt nach dem Vorbild von CART entweder auf Basis der Fehlklassifikationsrate oder der Gini-Wichtigkeit.

Mit einer Wahrscheinlichkeit von 40% ergibt sich nach Rekombination der Elterns das linke Nachkommenpaar. Das rechte Nachkommenpaar wird mit einer Wahrscheinlichkeit von 25% erstellt. Es existieren zwei weitere mögliche Nachkommenpaare, die jedoch nicht dargestellt sind.

Abbildung 16: Beispiel für die Knotenauswahl des VIM-basierten Rekombinationsoperators X_{VIM}

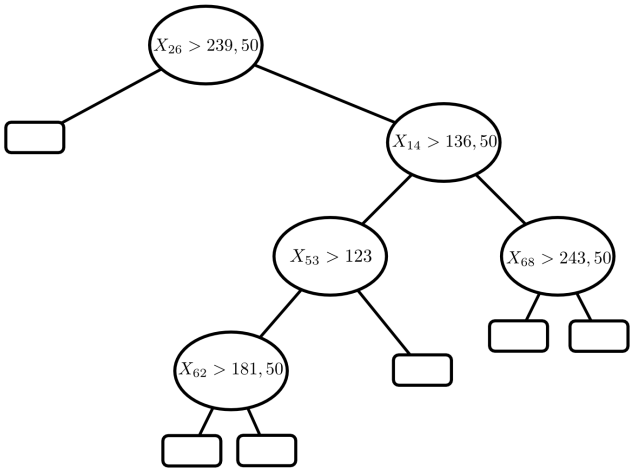
7 Implementierung

Alle Berechnungen werden in dieser Arbeit mit der Statistik-Software R (vgl. R Core Team, 2013) durchgeführt. Die CART-Bäume werden mit den Funktionen `rpart`, `prune` und `predict` aus dem Paket `rpart` (vgl. Therneau et al., 2012) erstellt und ausgewertet.

Für die Implementierung des *Wrapper*-Ansatzes bestehend aus einem EMOA mit Bitstring-Repräsentation und CART werden ebenfalls die CART-Funktionen aus dem Paket `rpart` zur Erstellung der Entscheidungsbäume und Bewertung der Variablenteilmengen verwendet. Die Funktionen zur Initialisierung der Startpopulation, die binäre Turnierselektion zur Auswahl der Eltern, die Variationsoperatoren Ein-Punkt-Rekombination und Bit-Flip-Mutation, sowie die Funktionen zur Kostenberechnung der Individuen im *Wrapper*-Ansatz mussten hingegen implementiert werden. Auch die Umweltselektionsoperatoren basierend auf den Prozeduren des NSGA-II mit entsprechender Berechnung der *Crowding*-Distanz und der SMS-EMOA mussten neu implementiert werden, da diese nicht aus den bereits bestehenden R-Paketen `mco` (vgl. Trautmann et al., 2010) oder `SPOT` (vgl. Bartz-Beielstein et al., 2012) in Verbindung mit CART verwendet werden konnten. Zur Ermittlung nichtdominierter Individuen und zur Berechnung der S-Metriken konnten jedoch die Funktionen `nds_rank` und `dominated_hypervolume` aus dem Paket `emoa` (vgl. Mersmann, 2011) benutzt werden. Abbildung 37 im Anhang stellt den Programmierungsablauf für den *Wrapper*-Ansatz dar.

Für NHEMOtree konnte ebenfalls nur auf wenig bestehenden Code zurückgegriffen werden. Die Implementierung der NHEMOtree-Individuen erfolgte selbstständig. Ein Individuum in NHEMOtree stellt einen binären Entscheidungsbaum dar, der im R-Code aus drei Komponenten besteht: der Zusammenhangs-, der Variablen- und der Cutoff-Tabelle. Die Zusammenhangstabelle ist eine Matrix, die die Baumstruktur beschreibt. Sie enthält drei Spalten und doppelt so viele Zeilen wie interne Knoten im Baum. In jeder Zeile wird die Zuordnung eines Tochterknotens zum entsprechenden Elternknoten beschrieben. Die erste Spalte enthält die Bezeichnung des Elternknotens als natürliche Zahl, die zweite Spalte die Position des Tochterknotens in Abhängigkeit des Elternknotens (0=links, 1=rechts) und die dritte Spalte die Bezeichnung des Tochterknotens als natürliche Zahl, wobei Blätter mit „-99“ codiert sind. Die Variablentabelle beinhaltet die Variablen bezüglich der internen Knoten aus der Zusammenhangstabelle und die Cutoff-Tabelle die entsprechenden Cutoff-Werte.Die Zusammenhangstabelle, die Variablentabelle und die Cutoff-Tabelle sind in einer Liste zusammengefasst und stellen gemeinsam ein NHEMOtree-Individuum dar.

Abbildung 17 zeigt den R-Code sowie die entsprechende graphische Darstellung eines NHEMOtree-Individuums. Per Definition werden stets die Knoten nach rechts abgebildet, die größer als der entsprechende Cutoff sind. Die Erstellung der Abbildungen erfolgt mit der Funktion `party` aus dem Paket `partykit` (vgl. Hothorn und Zeileis, 2012).




```
Beispielbaum
$Zusammenhangstabelle
  [,1] [,2] [,3]
     1    0  -99
Z    1    1    2
Z    2    0    3
Z    2    1    4
Z    3    0    5
Z    3    1  -99
Z    4    0  -99
Z    4    1  -99
Z    5    0  -99
Z    5    1  -99
```

```
$Varis
        K
[1,] 1 26
[2,] 2 14
[3,] 3 53
[4,] 4 68
[5,] 5 62
```

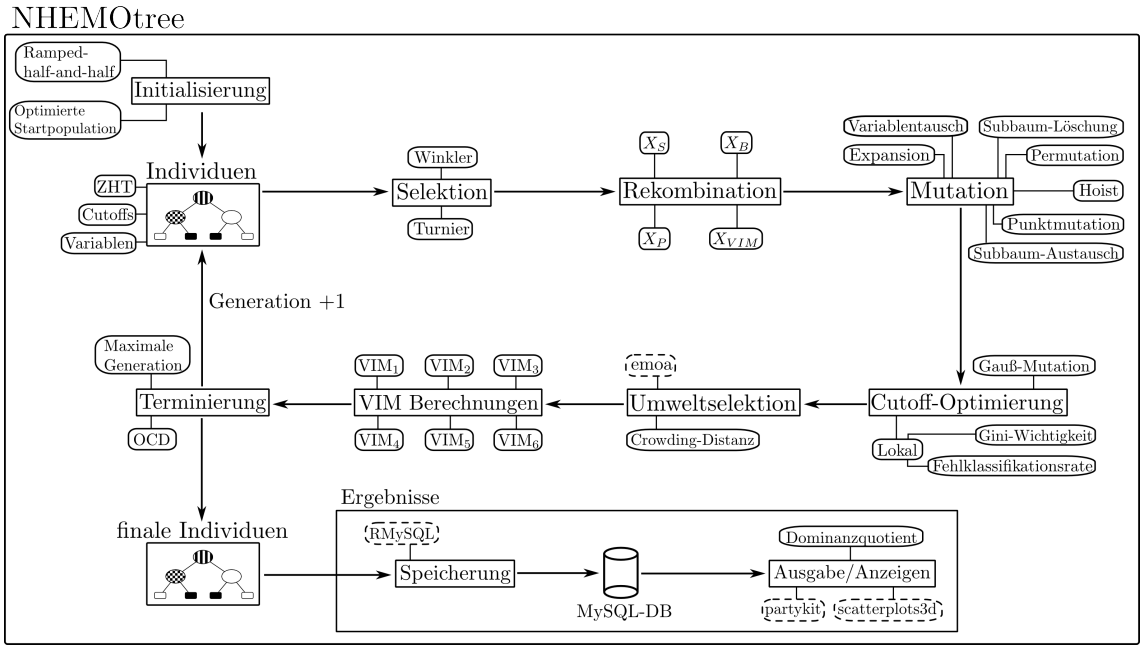
```
$CO
             CO
[1,] 1 239.50
[2,] 2 136.50
[3,] 4 123.50
[4,] 5 243.50
[5,] 6 181.50
```

Abbildung 17: Abbildung und R-Code eines NHEMOtree-Individuums

Für NHEMOtree mussten sowohl die *ramped-half-and-half*-Methode zur Initialisierung der Startpopulation, als auch die beiden Operatoren zur Elternselektion, die Turnierselektion und die Winkler-Selektion selbst implementiert werden. Ferner mussten die vier Rekombinationsoperatoren X_S, X_B, X_P und X_{VIM}, die mehrkriteriellen Variablenwichtigkeitsmaße und die Permutierte Fehlerfreiheit, die Mutationsoperatoren bezüglich der Baumstruktur und der Knotenvariablen, sowie die Gauß-Mutation mit Schrittweitenanpassung nach Rechenbergs 1/5-Erfolgsregel für die Mutation der Cutoff-Werte implementiert werden. Die Umweltselektionsoperatoren in NHEMOtree basieren auf den bereits für den *Wrapper*-Ansatz programmierten Operatoren. Das OCD-Abbruchkriterium wurde ebenfalls eigenständig implementiert, jedoch existiert ein guter Pseudocode von Trautmann et al. (2009). Der

Standard-NHEMOtree mit den entsprechenden Operatoren und mehrkriteriellem VIM ist als Pseudocode in Algorithmus 9 dargestellt.

Je Auswertestrategie (*Wrapper*-Ansatz und NHEMOtree) und Anwendung (Medizinischer Datensatz und Simulation) werden separate MySQL-Datenbanken, Version 5.1.11, erstellt. In diesen werden die finalen Populationen jedes Laufs gespeichert. Die Verbindung zwischen R und den MySQL-Datenbanken erfolgt über die entsprechenden Funktionen `dbConnect`, `dbDisconnect`, `dbGetQuery` und `dbSendQuery` aus dem Paket `RMySQL` (vgl. James und DebRoy, 2009). Weitere für die Programmierung verwendete Pakete sind `sets` und `scatterplot3d` (vgl. Meyer und Hornik, 2009; Ligges und Mächler, 2003). Abbildung 18 stellt den Programmierungsablauf für NHEMOtree dar. Die einzelnen Programmierungsschritte sind durch Rechtecke markiert. Zu jedem Programmierungsschritt sind die grundlegenden, selbstgeschriebenen Funktionen (abgerundete Rechtecke) und verwendete R-Pakete (gestrichelte Umrandung) aufgeführt.



Legende:

Rechtecke:	Einzelne Programmierungsschritte
Abgerundete Rechtecke:	Selbstgeschriebene Funktionen
Gestrichelte Umrandungen:	Verwendete R-Pakete

Abbildung 18: Programmierungsablauf für NHEMOtree

Alle Funktionen, der Wrapper- sowie der NHEMOtree-Algorithmus sind im gleichnamigen R-Paket `NHEMOtree` enthalten (vgl. Casjens, 2013).

Algorithmus 9: NHEMOtree

Definition:

P_g Population aus μ binären Entscheidungsbäumen in Generation g
Q_g Nachkommenpopulation in Generation g
$\mathrm{X}_{\mathrm{VIM}_5}$ VIM-basierter Rekombinationsoperator basierend auf VIM_5

Eingabe:

μ Größe der Population P_g
ν_{max} Maximal erlaubte Knotenanzahl je Baum
p_r Rekombinationsrate
p_m Mutationsrate
δ Abbruchkriterium

Ausgabe:

Finale Lösungspopulation P_{g+1}

Initialisierung:

$g = 0$
Erzeuge Population P_0 durch die *ramped-half-and-half*-Methode
Evaluierung der Individuen in P_0 anhand der Fitnessfunktionen (*):
1. Fünffach kreuzvalidierte Fehlklassifikationsrate
2. Vorhersagekosten als Variablenkosten je Baum

NHEMOtree:

while *Abbruchkriterium δ nicht erfüllt* **do**
  for $j = 1 \ldots \mu$ *Nachkommen* **do**
    Selektion zweier Eltern durch Turniere der Größe $\tau = 4$
    Nachkommenerstellung durch $\mathrm{X}_{\mathrm{VIM}_5}$ mit Wahrscheinlichkeit p_r
    Zufälliges Überleben eines der beiden Nachkommen
    Mutation des Nachkommens mit Wahrscheinlichkeit p_m:
    1. Mutation der Baumstruktur oder der Knotenvariablen durch:
      i. Subbaum-Löschung
      ii. Subbaum-Mutation
      iii. Expansionsmutation
      iv. Permutation
      v. Hoist-Mutation
      vi. Punktmutation oder
      vii. Vertauschung
    2. Gauß-Mutation der Cutoffs mit adaptiver Schrittweitenanpassung
    Löschen redundanter Knoten
    Evaluierung der Nachkommen analog zu (*)
  end
  $P_{g+1} \subset P_g \cup Q_g$ `/* Auswahl von` P_{g+1} `analog zum NSGA-II */`
  Berechnung von VIM_5 für die Individuen in P_{g+1}
  $g = g + 1$
end

8 Experimentelle Ausführungen am medizinischen Datensatz

In diesem Kapitel werden NHEMOtree, der mehrkriterielle *Wrapper*-Ansatz und CART auf medizinische Daten angewendet. In Abschnitt 8.1 werden zunächst die Daten vorgestellt und bereits veröffentlichte Ergebnisse basierend auf univariaten Analysen mit CART präsentiert. Anschließend werden die Ergebnisse des mehrkriteriellen *Wrapper*-Ansatzes (vgl. Abschnitt 8.2) und von NHEMOtree (vgl. Abschnitt 8.3) auf dem medizinischen Datensatz erläutert und miteinander verglichen.

Aufgrund der probabilistischen Natur der EMOAs ist nicht sichergestellt, dass die Ergebnisse zweier Läufe über dieselben Trainingsdaten identisch sind. Zur Gewährleistung valider Aussagen wird daher jede Methode mehrfach wiederholt und die über diese Läufe gemittelten Ergebnisse präsentiert.

Die Parameter für EMOAs sollten stets Problem-abhängig ausgewählt werden, da sie entscheidend für ein gutes Ergebnis sind (vgl. Parsons und Johnson, 1997). Verschiedene Ansätze existieren, die Parameter während des evolutionären Prozesses anzupassen (vgl. De Jong, 2007). In dieser Arbeit werden die Parameter nicht während eines Laufs adjustiert, sondern vorher angepasst, da hier der Fokus auf der Entwicklung eines neuen Klassifikationsalgorithmus für mehrkriterielle Optimierungsprobleme liegt. Verschiedene Parameterkombinationen je Algorithmus wurden durch Latin-Hypercube-Designs (vgl. McKay et al., 1979) ermittelt und die Parameterkombination mit den besten Ergebnissen für weitere Analysen ausgewählt. Eine Einführung in die Versuchsplanung von Computer-Experimenten und Latin-Hypercube-Designs befindet sich im Anhang in Abschnitt A.

8.1 Medizinischer Datensatz

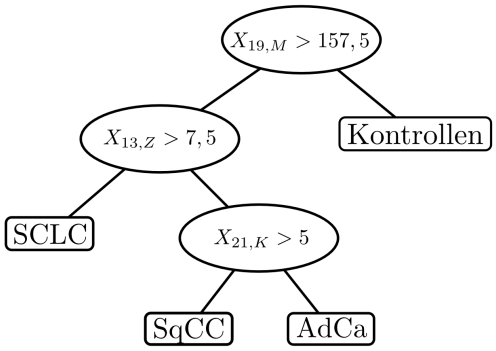
Im hier betrachteten medizinischen Anwendungsfall sollen die histologischen Subtypen des Lungenkrebs und gesundes Lungengewebe von 143 Probanden anhand der immunhistochemischen Färbung von 22 Proteinen in den drei Zellbestandteilen Zellmembran, Zytoplasma und Zellkern kostensensitiv klassifiziert werden. Die Proteine wurden auf Grundlage von Vorwissens aus der Literatur ausgewählt und mittels markierter Antikörper im Gewebe sichtbar gemacht (vgl. Abschnitt B.1 im Anhang). Die Färbung je Protein und Zellbestandteil wird durch je einen Score quantitativ erfasst (vgl. Abschnitt B.3 im Anhang), sodass insgesamt pro Gewebe 66 Farbintensitätsscores mit Ausprägungen zwischen 0 und 300 vorliegen.

Die Studienpopulation besteht aus männlichen Uranbergarbeitern, die zwischen 1957 und 1991 verstarben und am Institut für Pathologie in Stollberg (Sachsen) autopsiert wurden. Gesundes Lungengewebe stammt von männlichen Uranbergarbeiter ohne Krebserkrankung. Das balancierte Studiendesign sollte je 40 Probanden mit Adenokarzinom, Plattenepithelkarzinom und kleinzelligem Lungenkarzinom sowie 40 Personen ohne Lungenkrebs umfassen, wobei der Lungenkrebssubtyp von mindestens zwei von drei Referenzpathologen übereinstimmend klassifiziert sein musste. Aufgrund von fehlendem Probenmaterial bzw. mangelhafter Probenqualität mussten Gewebeproben nachträglich aus der Stichprobe entfernt werden, sodass für 143 Gewebeschnitte (35 Adenokarzinome, 33 Plattenepithelkarzinome, 30 kleinzellige Lungenkarzinome, 36 krebsfreies Lungengewebe) vollständige immunhistochemische Daten existieren. Eine detailierte Beschreibung der Studienpopulation, sowie zur einkriteriellen Analyse der Daten sind in Pesch et al. (2012) veröffentlicht.

Neben der Fehlklassifikationsrate dienen in dieser Arbeit als weitere Fitnessfunktionen die finanziellen Kosten für die verwendeten Antikörper zur Proteinfärbung (Finanzen) bzw. die Anzahl der verschiedenen Antikörper (Anzahl). In Abschnitt B.4 im Anhang werden die Kostenparameter für die kostensensitive Klassifikation im Detail vorgestellt.

Zum Vergleich der Pareto-Front-Approximationen wird unter anderem das dominierte Hypervolumen herangezogen, für dessen Referenzpunkt jeweils die maximal möglichen Fitnesswerte je Funktion angenommen werden. Bei der Fehlklassifikationsrate ist dies 100%, bei den Gesamtkosten der Antikörper 100 Kosteneinheiten und bei der Anzahl der Antikörper 22. Als Referenzpunkt wird somit bei der zweikriteriellen Optimierung von Fehlklassifikatonsrate und Finanzen (100; 100), bei der zweikriteriellen Optimierung von Fehlklassifikatonsrate und Anzahl (100; 22) und bei der dreikriteriellen Optimierung von Fehlklassifikatonsrate, Finanzen und Anzahl (100; 100; 22) genutzt. Ferner wird als Vergleichsmaß zwischen zwei Pareto-Fronten der in dieser Arbeit definierte Dominanzquotient (vgl. Definition 5 in Abschnitt 5.2) verwendet.

CART kann aufgrund seiner Konstruktion nur einkriterielle Klassifikationssprobleme lösen und erstellt genau einen Entscheidungsbaum (Standard-CART). In dieser Arbeit wird durch CART die Fehlklassifikationsrate ohne Berücksichtigung der Variablenkosten minimiert. Die einkriterielle Lösung ist in Abbildung 19 dargestellt und beschreibt einen binären Entscheidungsbaum bestehend aus den drei Knoten $\{X_{19,\mathrm{M}}\}$, $\{X_{13,\mathrm{Z}}\}$ und $\{X_{21,\mathrm{K}}\}$ mit einer LOOCV-Fehlklassifikationsrate (Fehlklassifikationsrate mit Leave-One-Out-Kreuzvalidierung) von 10,49% (vgl. Pesch et al., 2012) und 15,179 Kosten.



AdCa=Adenokarzinom, SCLC=Kleinzelliger Lungentumor, SqCC=Plattenepithelkarzinom, Kontrollen=Gesundes Lungengewebe

Abbildung 19: CART-Lösung des einkriteriellen Klassifikationsproblems zur Vorhersage der histologischen Lungenkrebssubtypen im medizinischen Datensatz

8.2 Ergebnisse des mehrkriteriellen *Wrapper*-Ansatzes

Im mehrkriteriellen *Wrapper*-Ansatz werden die drei EMOAs NSGA-II, *Steady-State* NSGA-II und SMS-EMOA (vgl. Abschnitt 5.3) miteinander verglichen. Der eingebettete Klassifikationsalgorithmus ist stets CART, der die durch den mehrkriteriellen *Wrapper* selektierten Variablenteilmengen mit einer fünffach kreuzvalidierten Fehlklassifikationsrate bewertet. Für die Individuen der finalen Pareto-Fronten wird abschließend mittels CART die LOOCV-Fahlklassifikationsrate bestimmt.

Die Parameter des mehrkriteriellen *Wrapper*-Ansatzes wurden mittels eines Latin-Hypercube-Versuchsplans ausgewählt. Tabelle 6 zeigt das Maximin-Latin-Hypercube-Design, das aus 10000 gebildeten Plänen die größte minimale Distanz zwischen den Versuchspunkten besitzt. So werden Läufe mit zehn verschiedenen Parametereinstellungen für Populationsgröße μ, Generationenanzahl g, Mutationsrate p_m, Rekombinationsrate p_r und Input-Rate π, die den Anteil der verwendeten Variablen bei der Initialisierung beschreibt, durchgeführt. Die Generationenzahl ist so angepasst, dass je Einstellung ungefähr 25000 Individuen durch den *Wrapper*-Ansatz berechnet werden. Bei Einstellung E_1 werden im NSGA-II bei einer Populationsgröße von 25 Individuen und 1000 Generationen 25000 neue Individuen betrachtet. Um eine Vergleichbarkeit mit den *Steady-State*-Algorithmen, die in jeder Generation nur einen Nachkommen erzeugen, herzustellen, werden in letzteren Fällen nicht 1000 Generationen, sondern 25000 Generationen berechnet, damit auch hier insgesamt 25000 Individuen betrachtet werden.

Nachfolgend werden die Ergebnisse je mehrkriteriellem *Wrapper* separat vorgestellt (vgl. Abschnitte 8.2.1 bis 8.2.3), wobei wegen der Ähnlichkeit der Ergebnisse lediglich

Einstellung	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}
μ	25	50	75	100	125	150	175	200	225	250
g	1000	500	333	250	200	167	143	125	111	100
p_r	0,60	0,90	0,85	0,55	0,75	0,80	1,00	0,95	0,65	0,70
p_m	0,015	0,010	0,025	0,075	0,001	0,150	0,125	0,005	0,100	0,050
π	0,60	0,90	0,30	0,50	0,10	0,80	0,20	0,70	0,40	0,95
Individuen	25000	25000	24975	25000	25000	25050	25025	25000	24975	25000

μ=Populationsgröße, g=Generationen, p_r=Rekombinationsrate, p_m=Mutationsrate, π=Input-Rate

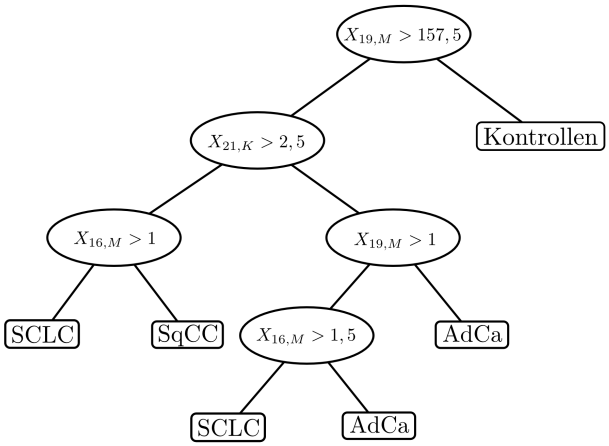
Tabelle 6: Parametereinstellungen für den mehrkriteriellen *Wrapper*-Ansatz nach dem Maximin-Latin-Hypercube-Design-Versuchsplan

die Resultate des NSGA-II-*Wrapper* im Detail und die der anderen mehrkriteriellen *Wrapper* verkürzt präsentiert werden. In Abschnitt 8.2.4 werden die Ergebniss der verschiedenen EMOA-*Wrapper* miteinander verglichen.

8.2.1 NSGA-II-*Wrapper*

Beim NSGA-II-*Wrapper* mit der Fehlklassifikationsrate und finanziellen Kosten als Fitnessfunktionen (NSGA-II-*Wrapper* (Finanzen)) sowie binärer Turnierselektion zur Elternauswahl, Ein-Punkt-Rekombination und Bitstring-Mutation beträgt die mittlere S-Metrik (vgl. Gleichung 5) der Pareto-Front-Approximation 9130. Somit sind 91,3% des potentiellen Hypervolumens abgedeckt. Die Pareto-Fronten der finalen Läufe bei verschiedenen Parametereinstellungen zeigen kaum Unterschiede. Die Differenz zwischen dem größten und der kleinsten mittleren S-Metrik liegt bei 1,8% (vgl. Tabelle 8). Parametereinstellungen mit niedrigeren Mutationsraten (z.B. Einstellung E_7) erreichen eine geringere mittlere S-Metrik als Parametereinstellungen mit hohen Mutationsraten. Die Spearman-Korrelation r_S zwischen Mutationsrate und S-Metrik liegt bei 0,84. Hohe Mutationsraten p_m, die für eine stärkere Variabilität innerhalb der Population sorgen, führen demnach zu besseren Ergebnissen. Für die Rekombinationsrate p_r zeigt sich keine Systematik in Bezug auf die Lösungsgüte.

Das *Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate von 6,99% und den Kosten 11,823 (vgl. Abbildung 20) wird mit allen Parametereinstellungen und verschiedenen Startwerten gefunden. Die Einstellungen E_1, E_5 und E_8 finden dieses Individuum im Gegensatz zu den anderen Einstellungen jedoch nicht in allen Läufen. Der NSGA-II-*Wrapper* (Finanzen) findet insgesamt drei nichtdominierte Individuen, die die Standard-CART-Lösung dominieren (vgl. Tabelle 7). Auch Zusatzexperimente mit reduzierten Generationen, sodass jeweils nur 2500 Individuen je Parametereinstellung berechnet werden, ergaben bessere Ergebnisse mit



AdCa=Adenokarzinom, SCLC=Kleinzelliger Lungentumor, SqCC=Plattenepithelkarzinom, Kontrollen=Gesundes Lungengewebe

Abbildung 20: *Wrapper*-Individuum mit minimaler LOOCV-Fehlklassifikationsrate

dem mehrkriteriellen *Wrapper*-Ansatz als mit CART. Der positive Zusammenhang zwischen Mutationsrate und Lösungsgüte konnte ebenfalls bestätigt werden.

Bei Betrachtung der Antikörperanzahl anstelle der finanziellen Kosten als zweite Fitnessfunktion (NSGA-II-*Wrapper* (Anzahl)) beträgt die mittlere S-Metrik der finalen Pareto-Fronten 1886. Dies entspricht einer Abdeckung von 85,7% des maximal möglichen Hypervolumens. Somit kann mit NSGA-II-*Wrapper* (Anzahl) im Vergleich zu NSGA-II-*Wrapper* (Finanzen) nur eine verminderte Lösungsqualität erreicht werden. Insgesamt sind die Pareto-Front-Approximationen bei den verschiedenen Parametereinstellungen sehr ähnlich und die Differenz zwischen der größten und der kleinsten S-Metrik beträgt 1,8% (vgl. Tabelle 8). Analog zum NSGA-II-*Wrapper* (Finanzen) werden leicht bessere Ergebnisse bei Einstellungen mit hohen Mutationsraten erzielt. Der Spearman-Korrelationskoeffizient zwischen Mutationsrate und S-Metrik beträgt 0,66.

Ferner lässt sich eine stark eingeschränkte Individuenvielfalt beobachten. Insgesamt findet der NSGA-II-*Wrapper* (Anzahl) zwölf verschiedene Individuen aus zehn verschiedenen Variablen, wovon drei Individuen nicht-dominiert sind: $\{X_{16,\mathrm{M}}, X_{19,\mathrm{M}}, X_{21,\mathrm{K}}\}$, $\{X_{19,\mathrm{M}}, X_{21,\mathrm{K}}\}$ und $\{X_{21,\mathrm{K}}\}$. Das *Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate $\{X_{16,\mathrm{M}}, X_{19,\mathrm{M}}, X_{21,\mathrm{K}}\}$ (vgl. Abbildung 20) wird nur einmal mit Hilfe der Einstellung E_9 gefunden. Die Individuen $\{X_{16,\mathrm{M}}, X_{19,\mathrm{M}}, X_{21,\mathrm{K}}\}$ und $\{X_{19,\mathrm{M}}, X_{21,\mathrm{K}}\}$ dominieren ferner die Standard-CART-Lösung (vgl. Tabelle 7).

Werden beim NSGA-II-*Wrapper* neben der Fehlklassifikationsrate sowohl die finanziellen Kosten als auch die Antikörperanzahl als Fitnessfunktionen verwendet (NSGA-II-*Wrapper* (Beides)), beträgt die mittlere S-Metrik der finalen Pareto-Front 190008,

Kostenfunktion	Individuum	Variablen	LOOCV-FKR [%]	Kosten	Antikörperanzahl
Finanzen	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
	3	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	10,49	9,172	3
Anzahl	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
Finanzen und Anzahl	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
	3	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	10,49	9,172	3
Standard-CART		$X_{13,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	10,49	15,179	3

Tabelle 7: Nicht-dominierte Individuen der NSGA-II-*Wrapper*-Ansätze, die die Standard-CART-Lösung dominieren

sodass 86,4% des potentiellen Hypervolumens abgedeckt sind. Analog zum NSGA-II-*Wrapper* (Finanzen) und NSGA-II-*Wrapper* (Anzahl) haben die Parametereinstellungen nur einen geringen Einfluss auf die S-Metriken der finalen Populationen (vgl. Tabelle 8). Die Individuenvielfalt ist ähnlich ausgeprägt wie beim NSGA-II-*Wrapper* (Finanzen). Lediglich die beiden Einstellungen mit den niedrigsten Mutationsraten (Einstellung E_5 und E_8) haben Schwierigkeiten, das Individuum mit minimaler Fehlklassifikationsrate (vgl. Abbildung 20) in jedem Lauf zu finden, sodass auch die S-Metriken für diese Parametereinstellungen geringer ausfallen. Der lineare Zusammenhang zwischen Mutationsrate und S-Metrik drückt sich in einem Spearman-Korrelationskoeffizienten von 0,42 aus und ist deutlich geringer als beim NSGA-II-*Wrapper* (Finanzen) und NSGA-II-*Wrapper* (Anzahl). NSGA-II-*Wrapper* (Beides) findet dieselben drei nicht-dominierte Individuen, die die Standard-CART-Lösung dominieren (vgl. Tabelle 7), wie NSGA-II-*Wrapper* (Finanzen).

Abbildung 21 zeigt für den NSGA-II-*Wrapper* mit den Einstellungen E_7, E_9 bzw. E_{10} die Pareto-Front-Approximation mit den finanziellen Kosten bzw. der Antikörperanzahl als zweiten Fitnessfunktion, sowie die drei-dimensionale Pareto-Front-Approximation, die bei simultaner Optimierung aller drei Ziele geschätzt wird. Die linke Abbildung zeigt die gute Approximation und hohe Diversität der zweikriteriellen Pareto-Front für Fehlklassifikationsrate und finanzielle Kosten. Die mittlere Abbildung zeigt die sehr beschränkte Lösungsvielfalt bei der zweikriteriellen Optimierung der Fehlklassifikationsrate und der Antikörperanzahl bestehend aus nur drei nicht-dominierten Lösungen. Die rechte Abbildung zeigt die dreidimensionale Pareto-Front-Approximation. Diese stellt vier „parallele Pareto-Fronten“ bezüglich der Fehlklassifikationsrate und der finanziellen Kosten, die nach der Antikörperanzahl aufgesplittet sind, ähnlich zur linken Abbildung dar.

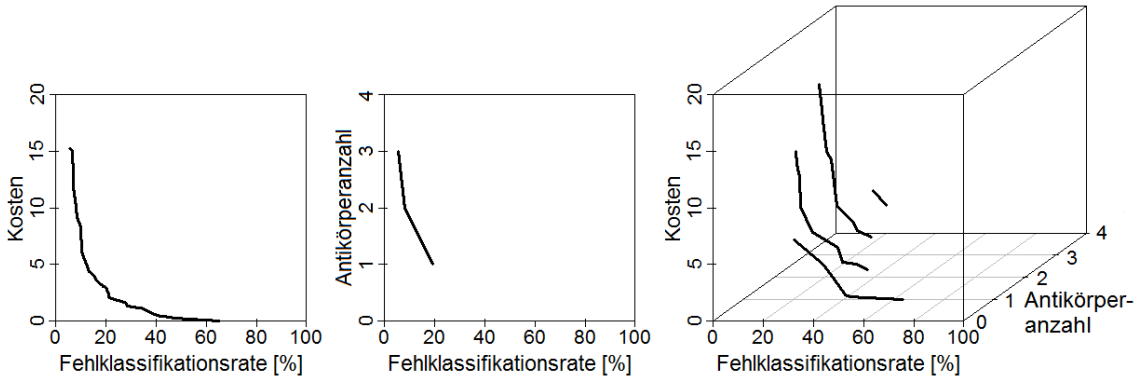


Abbildung 21: Vergleich der Pareto-Front-Approximationen bei verschiedenen optimierten Fitnessfunktionen im NSGA-II-*Wrapper*

Tabelle 9 gibt einen Überblick über die linearen Zusammenhänge zwischen der Mutationsrate p_m, der Rekombinationsrate p_r sowie der Populationsgröße μ und der S-Metrik. Dieser Zusammenhang ist in allen mehrkriteriellen *Wrapper*-Ansätzen bezüglich p_m stark ausgeprägt. Für p_r liegt kein solcher Zusammenhang vor. Die Korrelationen zwischen der S-Metrik und μ sind durchweg positiv.

Insgesamt zeigen die Ergebnisse, dass die Antikörperanzahl als zweite Fitnessfunktion ungeeignet ist. Die Modellierung mit drei Fitnessfunktionen liefert ähnliche Ergebnisse wie NSGA-II-*Wrapper* (Finanzen). Die zusätzliche Dimension und die daraus resultierende größere Komplexität führen zu einer Laufzeitverlängerung von etwa 10%. Des Weiteren wird bei Betrachtung der finanziellen Kosten mit 91,3% das meiste Hypervolumen dominiert. Daher wird im weiteren Verlauf der Fokus auf das zweikriterielle Optimierungsproblem bezüglich Fehlklassifikationsrate und finanzielle Kosten gelegt. Unabhängig von der Kostenfunktion enthalten die besten Einstellungen beim NSGA-II-*Wrapper* hohe Werte für die Mutationsrate p_m.

	NSGA-II			*Steady-State* NSGA-II			SMS-EMOA		
Einstellung	Finanzen	Anzahl	Beides	Finanzen	Anzahl	Beides	Finanzen	Anzahl	Beides
E_1	9056	1879	190300	9165	1880	187704	9115	1880	188460
E_2	9162	1880	190489	9164	1880	190272	9162	1880	190419
E_3	9164	1880	190485	9164	1889	190478	9163	1880	190403
E_4	9165	1880	189364	9167	1880	190339	9166	1880	190515
E_5	9006	1880	188183	9059	1880	188024	8991	1880	187988
E_6	9165	1887	190383	9158	1898	190462	9166	1898	190493
E_7	9170	1889	190419	9166	1898	190597	9166	1898	190465
E_8	9075	1880	189335	9163	1880	189679	9081	1880	188942
E_9	9165	1914	190518	9166	1880	190529	9126	1914	190699
E_{10}	9169	1889	190607	9165	1889	190473	9166	1889	190562
Arithm. Mittel	9130	1886	190008	9154	1885	189856	9130	1888	189895
Median	9165	1880	190401	9165	1880	190401	9163	1880	190442

Tabelle 8: Mittlere S-Metriken der finalen Populationen des mehrkriteriellen *Wrapper*-Ansatzes mit verschiedenen EAs, Fitnessfuntionen und Parametereinstellungen

	NSGA-II			*Steady-State* NSGA-II			SMS-EMOA		
	Finanzen	Anzahl	Beides	Finanzen	Anzahl	Beides	Finanzen	Anzahl	Beides
Mutationsrate p_m	0,84	0,66	0,42	0,49	0,66	0,73	0,81	0,80	0,75
Rekombinationsrate p_r	0,08	0,14	0,02	-0,39	0,41	0,25	0,07	0,05	-0,27
Populationsgröße μ	0,55	0,82	0,30	0,10	0,28	0,50	0,12	0,66	0,56

Tabelle 9: Spearman-Korrelationen zwischen Parametereinstellung und S-Metrik bei verschiedenen *Wrapper*-Ansätzen

8.2.2 *Steady-State* NSGA-II-*Wrapper*

Als weiterer EMOA wird der NSGA-II als *Steady-State*-Version untersucht. Die mittlere S-Metrik der finalen Pareto-Front-Approximationen beträgt bei Verwendung der finanziellen Kosten als zweite Fitnessfunktion (*Steady-State* NSGA-II-*Wrapper* (Finanzen)) 9154, sodass 91,5% des potentiellen Hypervolumens abgedeckt sind (vgl. Tabelle 8). Dies stimmt mit den Beobachtungen bei Verwendung des NSGA-II-*Wrappers* überein (vgl. Abschnitt 8.2.1). Zwischen den verschiedenen Parametereinstellungen zeigen sich kaum Unterschiede bezüglich der finalen Pareto-Front-Approximationen. Die Differenz zwischen der größten und der kleinsten mittleren S-Metrik liegt bei 1,2%, wobei dieser Unterschied hauptsächlich auf die geringe S-Metrik bei Einstellung E_5 mit der kleinsten Mutationsrate ($p_m = 0{,}001$) zurückzuführen ist. Der lineare Zusammenhang zwischen Mutationsrate und S-Metrik ist mit einer Spearman-Korrelation von 0,49 beim *Steady-State* NSGA-II-*Wrapper* schwächer als beim NSGA-II-*Wrapper* (vgl. Tabelle 9). Das *Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate (vgl. Abbildung 20) wird mit allen Parametereinstellungen gefunden, jedoch finden die Läufe mit Parametereinstellung E_5 diese nur einmal. *Steady-State* NSGA-II-*Wrapper* (Finanzen) findet zwei nichtdominierte Individuen, die die Standard-CART-Lösung dominieren (vgl. Tabelle A2 im Anhang).

Die Ergebnisse des *Steady-State* NSGA-II-*Wrapper* sowohl bei Verwendung der Antikörperanzahl als zweite Fitnessfunktion als auch bei Optimierung aller drei Fitnessfunktionen sind sehr ähnlich zu den entsprechenden Ergebnissen des NSGA-II-*Wrappers* (vgl. Tabelle 8, 9 und A2). Allerdings findet *Steady-State* NSGA-II-*Wrapper* (Anzahl) das *Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate (vgl. Abbildung 20) nicht. Ferner ist die Individuenvielfalt noch stärker eingeschränkt als die des NSGA-II-*Wrapper* (Anzahl) (vgl. Tabelle A2 im Anhang). Die Spearman-Korrelation zwischen Mutationsrate und S-Metrik ist beim *Steady-State* NSGA-II-*Wrapper* (Beides) mit $r_S = 0{,}73$ hingegen stark positiv (vgl. Tabelle 9).

8.2.3 SMS-EMOA-*Wrapper*

Neben dem NSGA-II wird der SMS-EMOA als weiterer mehrkriterieller *Wrapper* untersucht. Dessen mittlere S-Metrik beträgt bei Verwendung der finanziellen Kosten als zweite Fitnessfunktion (SMS-EMOA-*Wrapper* (Finanzen)) im Mittel 9130. Dies entspricht einer Abdeckung von 91,3% des maximal möglichen Hypervolumens und deckt sich mit den Beobachtungen des NSGA-II- und des *Steady-State* NSGA-II-*Wrapper*-Ansatzes. Die Pareto-Front-Approximationen der Läufe mit verschiedenen Parametereinstellungen sind sehr ähnlich. Die Differenz der S-Metriken liegt bei

1,7% (vgl. Tabelle 8). Analog zu den NSGA-II-*Wrappern* werden bei Parametereinstellungen mit niedrigen Mutationsraten geringere mittlere S-Metriken beobachtet ($r_S = 0{,}81$). Zwischen der Rekombinationsrate bzw. der Populationsgröße und den S-Metriken besteht kein linearer Zusammenhang (vgl. Tabelle 9). Das *Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate (vgl. Abbildung 20) wird mit Ausnahme von Einstellung E_5 ($p_m = 0{,}001$) bei allen Parametereinstellungen gefunden. Der SMS-EMOA-*Wrapper* (Finanzen) findet drei nicht-dominierte Individuen, die die Standard-CART-Lösung dominieren (vgl. Tabelle A3 im Anhang).

Die Ergebnisse des SMS-EMOA-*Wrapper* sowohl bei Verwendung der Antikörperanzahl als zweite Fitnessfunktion als auch bei Optimierung aller drei Fitnessfunktionen decken sich ebenfalls mit den Ergebnissen der anderen *Wrapper* (vgl. Tabelle 8 und 9). SMS-EMOA-*Wrapper* (Anzahl) findet dieselben nicht-dominierten Individuen wie der entsprechende NSGA-II-*Wrapper*. Der SMS-EMOA-*Wrapper* (Beides) findet hingegen ein nicht-dominiertes Individuum zusätzlich (vgl. Tabelle A3 im Anhang).

8.2.4 Vergleich der mehrkriteriellen *Wrapper*

Die Ergebnisse der drei mehrkriteriellen *Wrapper* sind im Allgemeinen sehr ähnlich (vgl. Tabelle 8). Je nach betrachteter Fitnessfunktion werden 85% bis 91% des maximal möglichen Hypervolumens durch die finalen Pareto-Fronten abgedeckt.

In Tabelle 10 sind die Anzahl der nicht-dominierten Individuen der finalen Populationen (N) und die Dominanzquotienten γ^D der drei mehrkriteriellen *Wrapper* mit den jeweils besten Parametereinstellungen aufgeführt. Demnach findet der NSGA-II-*Wrapper* (Finanzen) mit 26 Individuen die meisten nicht-dominierten Lösungen. Im Vergleich zum *Steady-State* NSGA-II-*Wrapper* (Finanzen) enthält die gemeinsame Pareto-Front jedoch nur ein Drittel der Individuen vom NSGA-II-*Wrapper* ($\gamma^D = 0{,}26$), aber 17% mehr Individuen vom NSGA-II-*Wrapper* im Vergleich zum SMS-EMOA-*Wrapper* (Finanzen).

Die Nicht-Eignung der Antikörperanzahl als zweite Fitnessfunktion drückt sich in allen mehrkriteriellen *Wrappern* durch eine stark eingeschränkte Lösungsgüte und -vielfalt aus. Die approximativen Pareto-Fronten bestehen jeweils aus nur drei Individuen (vgl. Tabelle 10). Da der Wertebereich dieser Fitnessfunktion lediglich die ganzen Zahlen zwischen 0 und 22 umfasst, kann diese Fitnessfunktion kaum differenzierte Werte hervorbringen und das Potential der mehrkriteriellen Optimierung kann kaum ausgeschöpft werden. Die Verwendung der finanziellen Kosten als zweite Fitnessfunktion erzeugt hingegen heterogenere Ergebnisse mit einer Vielzahl an Lösungen.

		NSGA-II		*Steady-State* NSGA-II		SMS-EMOA	
		N	γ^D	N	γ^D	N	γ^D
Finanzen	NSGA-II	26		5	0,29	14	1,17
	Steady-State NSGA-II	17	3,40	20		17	4,25
	SMS-EMOA	12	0,86	4	0,24	24	
Anzahl	NSGA-II	3		3	1,50	3	1,00
	Steady-State NSGA-II	2	0,67	3		3	1,00
	SMS-EMOA	3	1,00	3	1,00	3	
Beides	NSGA-II	27		21	3,00	19	1,06
	Steady-State NSGA-II	7	0,33	28		12	0,86
	SMS-EMOA	18	0,95	14	1,17	25	

N Individuen aus dem „Zeilen"-Ansatz sind nicht-dominiert im Vergleich zum „Spalten"-Ansatz, z.B. γ^D (*Steady-State* NSGA-II (Finanzen), NSGA-II (Finanzen)) $= \frac{17}{5} = 3{,}40$.

Tabelle 10: Anzahl nicht-dominierter Individuen N und Dominanzquotient γ^D im mehrkriteriellen *Wrapper*-Ansatz

Der Vergleich der nicht-dominierten Individuen des dreikriteriellen Optimierungsproblems zeigt einen klaren Vorteil für den NSGA-II-*Wrapper*. So sind etwa in der gemeinsamen Pareto-Front von NSGA-II-*Wrapper* (Beides) und *Steady-State* NSGA-II-*Wrapper* (Beides) 21 NSGA-II-Individuen und nur 7 *Steady-State* NSGA-II-Individuen enthalten ($\gamma^D = 3{,}00$).

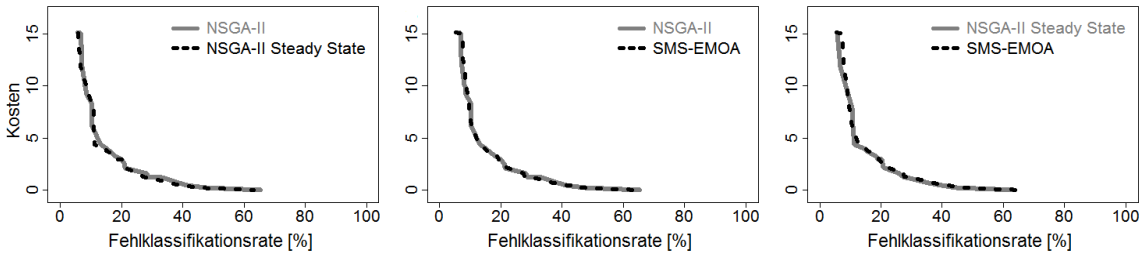


Abbildung 22: Vergleich der Pareto-Front-Approximationen des zweikriteriellen Optimierungsproblems bezüglich Fehlklassifikationsrate und finanziellen Kosten bei verschiedenen EMOAs im mehrkriteriellen *Wrapper*-Ansatz

Beim Vergleich der verschiedenen EMOAs als mehrkriterielle *Wrapper* mit den besten Parametereinstellungen zeigen sich trotz der verschiedenen Dominanzquotienten sehr ähnliche Pareto-Fronten. Abbildung 22 stellt die Pareto-Fronten des zweikriteriellen Optimierungsproblems mit den Fitnessfunktionen Fehlklassifikationsrate und finanziellen Kosten bei Verwendung der verschiedenen EMOAs dar. Die Beobachtungen von Emmerich et al. (2005), dass im Vergleich zum NSGA-II der SMS-EMOA zu besseren Kompromisslösungen im *Knick* einer konvexen Pareto-Front führt, können hier nicht bestätigt werden. Die um mehr als 10% kürzeren Laufzeiten des NSGA-II gegenüber dem *Steady-State* NSGA-II bzw. dem SMS-

EMOA sind vorteilhaft. Kürzere Laufzeiten des NSGA-II im Vergleich zum SMS-EMOA beobachteten ebenfalls Bader und Zitzler (2011). Ferner ist der NSGA-II der etablierteste der betrachteten EMOAs (vgl. Emmerich et al., 2005), sodass im weiteren Verlauf der Arbeit lediglich der NSGA-II mit generationsbasierter Umweltselektion betrachtet wird.

Alle drei mehrkriteriellen *Wrapper* haben einen großen Vorteil gegenüber der Standard-CART-Lösung und finden bessere Lösungen Die Verwendung des zweikriteriellen Optimierungsproblems mit der Fehlklassifikationsrate und den finanziellen Kosten als Fitnessfunktionen zeigt die besten Ergebnisse, sodass im weiteren Verlauf der Fokus auf der Analyse des zweikriteriellen Optimierungsproblems mit der Fehlklassifikationsrate und den finanziellen Kosten als Fitnessfunkionen liegt. Unabhängig von der Kostenfunktion und dem mehrkriteriellen *Wrapper* sind die besten Parametereinstellungen solche mit hohen Mutationsraten p_m. Bei den anderen Parametern ist kein solcher Zusammenhang beobachtbar (vgl. Tabelle 9).

8.3 NHEMOtree

Wie angekündigt wird mit NHEMOtree nur das zweikriterielle Optimierungsproblem bezüglich der gleichzeitigen Minimierung der Fehlklassifikationsrate und der finanziellen Kosten betrachtet, da diese Fitnessfunktionen im mehrkriteriellen *Wrapper*-Ansatz für den medizinischen Datensatz am geeignetsten erschienen. Wie bereits in Kapitel 6 erläutert, werden die Parameter des NHEMOtrees ebenfalls mittels eines Latin-Hypercube-Versuchsplans erstellt (vgl. Tabelle 11). Dies geschieht in Anlehnung an die Vorschläge von Koza (1992), Banzhaf et al. (1998), Zhao (2007) und Winkler et al. (2009), sowie den Erfahrungen aus den erzielten Ergebnissen des mehrkriteriellen *Wrapper*-Ansatzes (höhere Mutationsraten und Individuenzahlen erzielen bessere Ergebnisse). Es werden nur fünf verschiedene Parametereinstellungen getestet, da die Ergebnisse des mehrkriteriellen *Wrapper*-Ansatz keine diffizilen Unterschiede zwischen den Einstellungen aufwiesen. Aufgrund des Zusammenhangs zwischen der Mutationsrate und der S-Metrik beim mehrkriteriellen *Wrapper*-Ansatz werden in NHEMOtree stark verschiedene Mutationsraten untersucht. Außerdem wird die Gesamtindividuenanzahl von 25000 im mehrkriteriellen *Wrapper*-Ansatz auf 10000 in NHEMOtree reduziert, da in vorangegangenen Experimenten auch diese verringerte Individuenanzahl bereits zu guten Ergebnissen führte. Um *Bloating* entgegen zu wirken, wird ferner eine maximale Baumgröße von 20 Knoten eingeführt.

Die verwendeten Selektions- und Variationsoperatoren in NHEMOtree wurden bereits in Kapitel 6 beschrieben. Demnach sind die Art der Elternselektion (Turnierselektion der Größe $\tau = vier$ oder Winkler-Selektion) sowie der Rekombinationsopera-

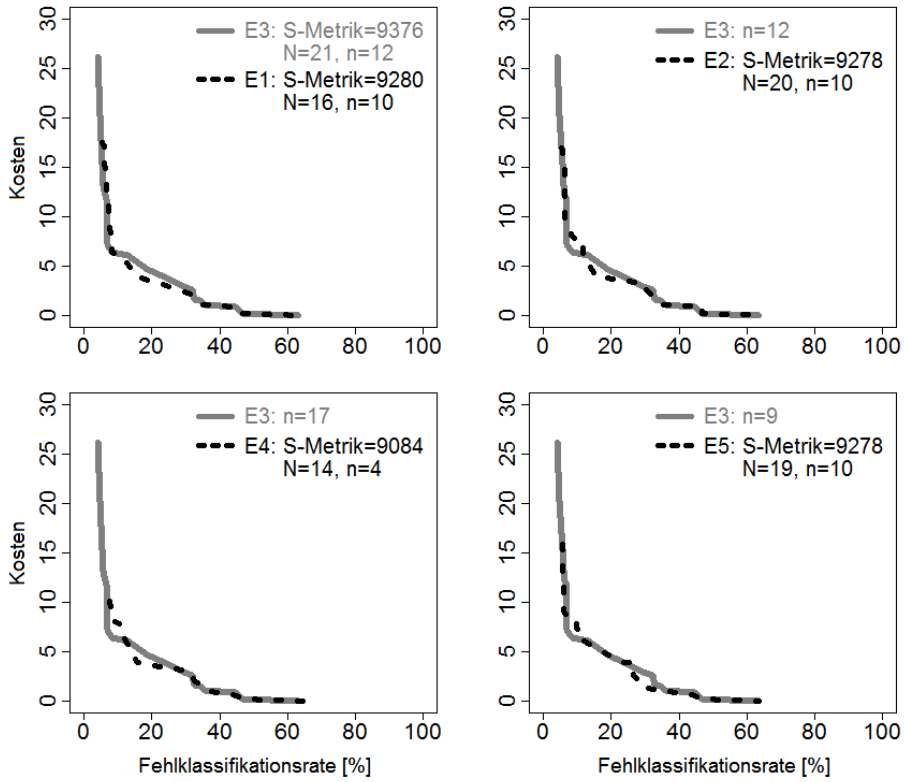
Einstellung	E_1	E_2	E_3	E_4	E_5
Generationen g	100	200	50	20	34
Populationsgröße μ	100	50	200	500	300
Rekombinationsrate p_r	0,80	0,70	0,50	0,06	0,90
Mutationsrate p_m	0,10	0,05	0,50	0,01	0,20
Individuen	10000	10000	10000	10000	10200

Tabelle 11: Parametereinstellungen für NHEMOtree nach dem Maximin-Latin-Hypercube-Design-Versuchsplan

tor (X_S, X_B, X_P oder X_{VIM}) je Lauf fest. Die verschiedenen Operatoren für die Mutation der Baumstruktur und der Knotenvariablen werden hingegen je Individuum per Zufall ausgewählt. Die Cutoffs der einzelnen Knotenvariablen werden standardmäßig in jeder Generation mittels Gauß-Mutation (vgl. Abschnit 4.3) angepasst. Im Gegensatz zum mehrkriteriellen *Wrapper*-Ansatz wird die Umweltselektion in NHEMOtree analog zum NSGA-II ausschließlich durch die nicht-dominierte Sortierung und die *Crowding*-Distanz durchgeführt, da sich der NSGA-II im mehrkriteriellen *Wrapper*-Ansatz als geeignet herausstellte und die geringsten Laufzeiten benötigte (vgl. Abschnitt 8.2). Die durch NHEMOtree untersuchten Szenarien sind in Tabelle A4 im Anhang zusammengefasst.

Abbildung 23 zeigt den Vergleich der finalen Pareto-Fronten der verschiedenen Parametereinstellungen in NHEMOtree bei Verwendung der Turnierselektion mit Größe $\tau = 4$ zur Elternauswahl und Standard-Rekombinationsoperator X_S. Die Überlegenheit der Einstellung E_3 mit Populationsgröße $\mu = 200$, $g = 50$ Generationen und Rekombinations- und Mutationsrate $p_r = p_m = 0{,}50$ ist unverkennbar. Vor allem im Bereich geringer Fehlklassifikationsraten und hoher Kosten ist E_3 den anderen Einstellungen weit überlegen. Mit einer mittleren S-Metrik von 9376 wird mit E_3 das größte Hypervolumen dominiert. Ferner werden $N = 21$ nicht-dominierte Individuen gefunden, die meistens mehr Individuen aus Läufen mit anderen Parametereinstellungen dominieren (vgl. Tabelle 12). Beispielsweise besteht die gemeinsame Pareto-Front von E_3 und E_4 aus $n = 17$ Individuen von E_3 und $n = 4$ Individuen von E_4, sodass der Dominanzquotient $\gamma^D(E_3,E_4 = 4{,}25)$ beträgt. Analog zum mehrkriteriellen *Wrapper*-Ansatz führt somit eine hohe Mutationsrate zu besseren Ergebnissen. Die Spearman-Korrelation zwischen Mutationsrate und S-Metrik beträgt 0,82.

Das NHEMOtree-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate von 4,22% und Kosten 26,20 erstellt mit E_3 und X_S ist in Abbildung 24 dargestellt. Es ist mit zehn internen Knoten größer als das *Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate (vgl. Abbildung 20). Beide Individuen sind nicht-dominiert.



N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 23: Vergleich der Pareto-Front-Approximationen bei verschiedenen Parametereinstellungen in NHEMOtree mit Standard-Rekombinationsoperator X_S

Nachfolgend werden verschiedene Operatoren und Einstellungen in NHEMOtree mit dem Ziel untersucht, verbesserte Ergebnisse zu erreichen, sodass ein finaler NHEMOtree-Algorithmus empfohlen werden kann. In Abschnitt 8.3.1 wird der Effekt verschiedener Startpopulationen untersucht. Abschnitt 8.3.2 beschäftigt sich mit der Reduzierung der maximal erlaubten Baumgröße in NHEMOtree und Abschnitt 8.3.3 untersucht den Effekt der Winkler-Selektion im Vergleich zur Turnierselektion auf die Lösungsgüte. In den Abschnitten 8.3.4 und 8.3.5 werden verschiedene Rekombinationsoperatoren miteinander verglichen und schließlich in den Abschnitten 8.3.6 und 8.3.7 die Ergebnisse des NHEMOtrees mit OCD-Abbruchkriterium bzw. mit lokaler Cutoff-Optimierung beschrieben.

8.3.1 NHEMOtree mit optimierter Startpopulation

Statt einer zufälligen Initialisierung durch die *ramped-half-and-half*-Methode kann NHEMOtree auch mit einer bereits optimierten Population gestartet werden. Als optimierte Startpopulation wird hier die finale Population eines NSGA-II-*Wrappers* genutzt. Es werden stets eine höhere Diversität und geringere Fehlklassifikations-

Einstellung	N	S-Metrik	FKR [%]	Kosten	$\gamma^D(E_3,E_i)$
E_1	16	9280	5,55 - 64,49	0,008 - 17,483	1,20
E_2	20	9278	5,53 - 65,04	0,008 - 16,974	1,20
E_3	21	9376	4,22 - 63,47	0,008 - 26,201	1
E_4	14	9084	7,68 - 64,98	0,008 - 10,148	4,25
E_5	19	9278	5,59 - 63,47	0,008 - 15,854	0,90

E_i: NHEMOtree mit Parametereinstellung E_i nach Tabelle 11

N: Nicht-dominierte Individuen in der approximierten Pareto-Front

FKR: Fehlklassifikationsrate

γ^D: Dominanzquotient für E_3 bezüglich E_i

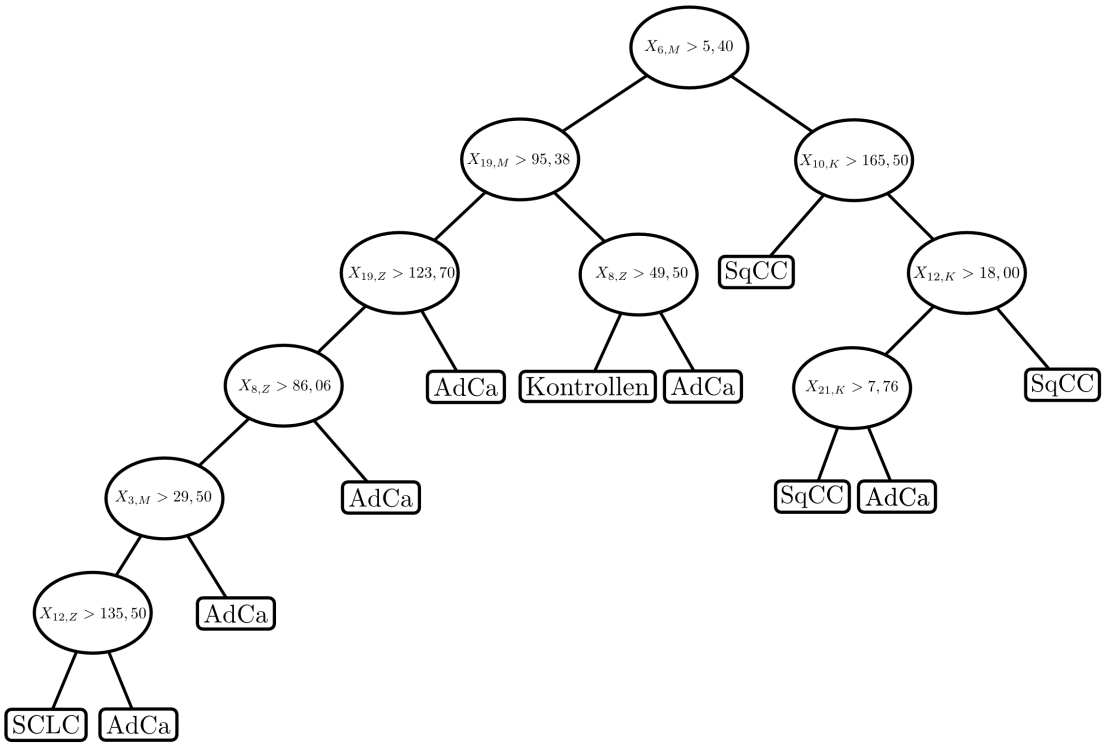
Tabelle 12: Vergleich der finalen Populationen mit verschiedenen Parametereinstellungen des NHEMOtrees

raten, aber auch höhere Kosten, meistens größere Bäume und bauchigere Pareto-Front-Approximationen bei Nutzung der optimierten Startpopulation erzielt (vgl. Abbildung 25(a)). Die Lösungen der optimierten Startpopulation dominieren viele finale Individuen bei Verwendung einer zufälligen Startpopulation.

Allerdings basiert die optimierte Startpopulation bereits auf einer vorgelagerten Optimierung mit 40000 Individuen, sodass insgesamt eine Optimierung mit 50000 Individuen erfolgt. Beim Vergleich zwischen den finalen NHEMOtree-Individuen mit optimierter Startpopulation zuzüglich 10000 neu erstellter Individuen und den finalen Individuen basierend auf einer zufälligen Initialisierung mit 50000 generierten NHEMOtree-Individuen liefert letztere Optimierung Lösungen mit höheren S-Metriken, verringerten Fehlklassifikationsraten und finanziellen Kosten. Vor allem im Bereich geringer Fehlklassifikationsraten ist NHEMOtree mit zufälliger Startpopulation der Kombination aus mehrkriteriellem *Wrapper*-Ansatz und NHEMOtree eindeutig überlegen (vgl. Abbildung 25(b)). Der Dominanzquotient beträgt jedoch 0,74 für die zufällige Initialisierung bezüglich einer optimierten Startpopulation. Insgesamt ist allerdings eine optimierte Startpopulation basierend auf einem mehrkriteriellen *Wrapper*-Ansatz für NHEMOtree nicht vorteilhaft.

8.3.2 NHEMOtree mit reduzierter Baumgröße

Eine stärkere Beschränkung der Baumgröße durch Reduzierung der maximal erlaubten Knotenanzahl je Baum von $\nu_{max} = 20$ auf $\nu_{max} = 10$ drückt sich durch eine geringere Diversität der Lösungen in der finalen Population, einen Dominanzquotienten kleiner Eins ($\gamma^D(\nu_{max} = 10, \nu_{max} = 20) = \frac{9}{11} = 0{,}82$), leicht reduzierte S-Metriken sowie höhere Fehlklassifikationsraten aus (vgl. Abbildung 26). Es existiert keine Parametereinstellung, nach der die Lösungen mit D schlechter sind als mit $\nu_{max} = 10$. Analog zu den Läufen mit $\nu_{max} = 20$ führt auch bei der reduzierten



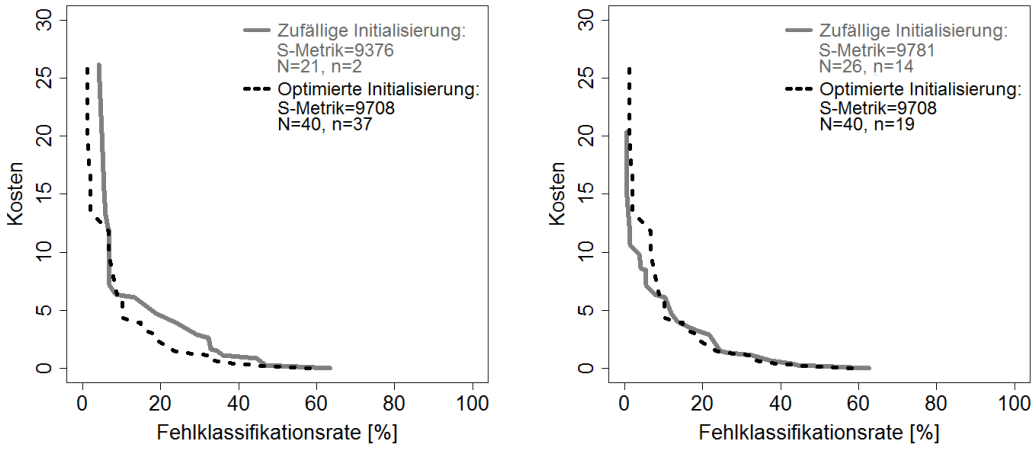
AdCa=Adenokarzinom, SCLC=Kleinzelliger Lungentumor, SqCC=Plattenepithelkarzinom, Kontrollen=Gesundes Lungengewebe

Abbildung 24: NHEMOtree-Individuum mit minimaler Fehlklassifikationsrate erstellt mit Standard-Rekombinationsoperator X_S und Parametereinstellung E_3

Knotenanzahl Parametereinstellung E_3 zu den höchsten S-Metriken. Die ähnliche Laufzeit zwischen den Läufen mit $\nu_{max} = 20$ und $\nu_{max} = 10$ (Die Laufzeiterhöhung für $\nu_{max} = 20$ beträgt 1,7%.) deutet bereits an, dass im Anwendungsbeispiel die Individuen selten mehr als zehn Knoten enthalten und nur vereinzelt die maximale Knotenanzahl erreicht wird. Aufgrund der höheren Individuenvielfalt bei größeren Bäumen und keiner Nachteile durch längere Laufzeiten, wird im weiteren Verlauf NHEMOtree mit $\nu_{max} = 20$ verwendet.

8.3.3 NHEMOtree mit Elternselektion nach Winkler

Die Kombination aus Roulette- und zufälliger Selektion (Winkler-Selektion) ist laut Winkler et al. (2009) besonders für die Elternselektion in EAs mit Baum-Repräsentation geeignet. In NHEMOtree zeigt der Vergleich der Turnier- mit der Winkler-Selektion jedoch, dass keine der beiden Selektionsarten zu besseren Pareto-Front-Approximationen führt. Die Turnierselektion bringt hauptsächlich bessere Individuen mit geringer Fehlklassifikationsrate hervor, während die Winkler-Selektion im Bereich der geringen Kosten bessere Lösungen findet (vgl. Abbildung 27). Aufgrund



(a) 10000 NHEMOtree-Individuen (b) 50000 NHEMOtree-Individuen

N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 25: Vergleich der Pareto-Front-Approximationen bei verschiedenen Initialisierungsarten in NHEMOtree

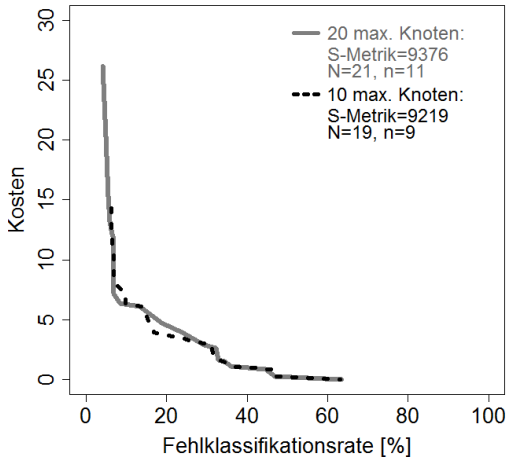
der leicht längeren Laufzeit bei Verwendung der Winkler-Selektion (2,3% verlängert) wird hier die Turnierselektion bevorzugt und im weiteren Verlauf verwendet.

8.3.4 NHEMOtree mit etablierten Rekombinationsarten

Wie bereits erwähnt, werden in NHEMOtree neben dem Standard-Rekombinationsoperator X_S noch weitere Rekombinationsarten untersucht. Der Einfluss der Rekombinationsart auf die Lösungsqualität des NHEMOtrees wird nachfolgend analysiert.

Brut-Rekombination (X_B)
Bei der Brut-Rekombination (X_B) mit Brutgröße $\rho = 4$ werden vier Mal mehr Nachkommen erstellt als bei X_S, sodass für die Brut-Rekombination hohe Rekombinationsraten besonders geeignet sind und die Häufigkeit der Rekombination einen starken Einfluss auf die Lösungsgüte hat. Als beste Einstellung stellt sich folglich Einstellung E_5 mit der höchsten Rekombinationsrate von 90% heraus. Mit der Brut-Rekombination erreicht NHEMOtree mit allen Einstellungen die höchsten S-Metriken (vgl. Tabelle 13).

Insgesamt zeigt sich bei der Brut-Rekombination im Vergleich zu den anderen Rekombinationsarten eine bessere Lösungsqualität. Im Allgemeinen ist die Diversität bei Verwendung von X_B gegenüber der Verwendung von X_S und X_P erhöht und die Fehlklassifikationsraten sind stets reduziert. Die Kosten sind wechselhaft (vgl. Tabelle 13).



N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

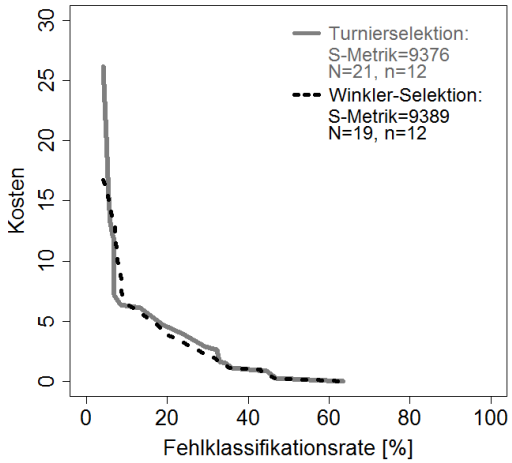
Abbildung 26: Vergleich der Pareto-Front-Approximationen bei verschiedenen maximal erlaubten Knoten je Individuum in NHEMOtree

Bei den Parametereinstellungen, die jeweils die höchsten S-Metriken erzielen, werden 20 Individuen bei Verwendung von X_B nicht durch die Individuen bei Verwendung von X_S dominiert, gleichzeitig sind aber nur vier Individuen von X_S nicht-dominiert. Der Dominanzquotient für X_B bezüglich X_S $\left(\gamma^D(X_B,X_S) = \frac{20}{4} = 5\right)$. Sowohl im Bereich der niedrigen Fehlklassifikationsrate als auch im Bereich der niedrigen Kosten ist der Unterschied zwischen den beiden approximierten Pareto-Fronten deutlich sichtbar (vgl. Abbildung 28(a)).

Beim Vergleich der Rekombinationsarten sollte jedoch bedacht werden, dass X_B bei identischen Algorithmus-Einstellungen stets mehr Nachkommen als X_S generiert. Wird hingegen bei X_S die Populationsgröße erhöht, sodass in beiden NHEMOtrees gleich viele Nachkommen während eines Laufs erzeugt werden, ist X_B nicht mehr überlegen (vgl. Abbildung 28(b)). Entsprechend ist bei einer Populationsgröße von $\mu = 400$ bei Verwendung von X_S die S-Metrik leicht gegenüber der S-Metrik bei Verwendung von X_B erhöht ($\gamma^S_{X_S} = 9596$ vs. $\gamma^S_{X_B} = 9523$). Die Pareto-Front-Approximation bei Verwendung von X_S ist insgesamt bauchiger und insbesondere im Bereich der niedrigen Fehlklassifikationsraten überlegen. Der Dominanzquotient zeigt jedoch einen Vorteil für X_B bezüglich X_S mit $\gamma^D(X_B,X_S) = \frac{13}{11} = 1{,}18$.

Ein-Punkt-Rekombination nach Poli und Langdon (X_P)

Bei der Ein-Punkt-Rekombination nach Poli und Langdon (1998a) (X_P) wird ein Rekombinationspunkt mit gleichverteilter Wahrscheinlichkeit aus der Menge aller potentiellen Punkte so ausgewählt, dass bei beiden Elternteilen der Rekombinationspunkt auf derselben Baumebene liegt. Analog zum X_S ist Einstellung E_3 mit der höchsten Mutationsrate gegenüber den anderen Parametereinstellungen vor-



N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 27: Vergleich der Pareto-Front-Approximationen bei verschiedenen Elternselektionsarten in NHEMOtree

teilhaft (vgl. Tabelle 13). Im Vergleich zur Verwendung von X_S liefert X_P häufig kostengünstigere Individuen. Allerdings sind damit meistens auch höhere Fehlklassifikationsraten und eine reduzierte S-Metrik verbunden. Der Vergleich der Pareto-Front-Approximation zeigt keinen Vorteil für X_P (vgl. Abbildung 29). Die finale Pareto-Front des NHEMOtrees mit X_S besteht aus 21 Individuen im Gegensatz zu 18 Individuen bei Verwendung von X_P. Ferner ist der Dominanzquotient von X_S bezüglich X_P größer Eins ($\gamma^D(X_S,X_P) = \frac{15}{9} = 1{,}67$). Die Berechnungsdauern bei Verwendung von X_S und X_P sind identisch.

Insgesamt ist X_S also sowohl gegenüber X_B als auch gegenüber X_P zu bevorzugen. Im weiteren Verlauf der Arbeit fungiert X_S als Vergleichsmaß für den neuen VIM-basierten Rekombinationsoperator X_{VIM}, der X_S mittels mehrkriterieller Variablenwichtigkeitsmaße adaptiert.

8.3.5 NHEMOtree mit VIM-basiertem Rekombinationsoperator

Analog zu den vorangegangenen Ergebnissen wird NHEMOtree bei Verwendung der VIM-basierten Rekombinationsoperatoren X_{VIM} mit den besten Parametereinstellungen dargestellt. Die Individuen werden mittels der *ramped-half-and-half*- Methode initialisiert, die Eltern mittels Turnierselektion der Größe $\tau = 4$ ausgewählt und die Parametereinstellung E_3 nach Tabelle 11 mit $\nu_{max} = 20$ Knoten je Baum verwendet.

Tabelle 14 stellt die Ergebnisse der finalen Populationen nach Anwendung von CART, dem NSGA-II-*Wrapper* mit Parametereinstellung E_7 und den verschiedenen

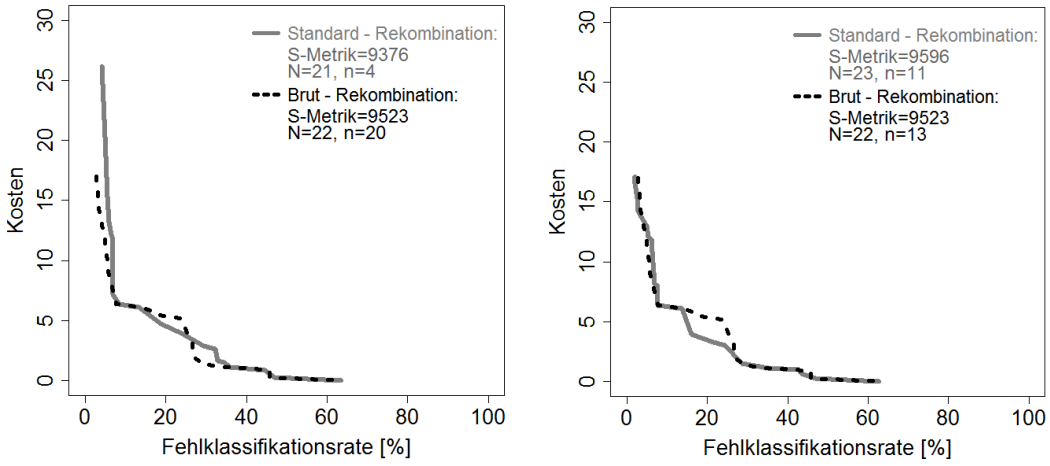
Einstellung	N	S-Metrik	FKR [%]	Kosten
Standard-Rekombination X_S				
1	16	9280	5,55 - 64,49	0,008 - 17,483
2	20	9278	5,53 - 65,04	0,008 - 16,974
3	21	9376	4,22 - 63,47	0,008 - 26,201
4	14	9084	7,68 - 64,98	0,008 - 10,148
5	19	9278	5,59 - 63,47	0,008 - 15,854
Brut-Rekombination X_B				
1	21	9453	3,80 - 62,71	0,008 - 14,963
2	20	9269	5,54 - 62,17	0,008 - 14,280
3	30	9455	3,49 - 63,47	0,008 - 19,023
4	20	9420	4,02 - 64,66	0,008 - 17,529
5	22	9523	2,80 - 63,14	0,008 - 17,007
Rekombination nach Poli X_P				
1	14	9224	6,31 - 64,47	0,008 - 9,766
2	21	9124	6,93 - 64,33	0,008 - 15,731
3	18	9324	4,89 - 64,85	0,008 - 23,570
4	19	9040	7,68 - 65,20	0,008 - 16,638
5	21	9313	4,89 - 63,11	0,008 - 19,209

N: Anzahl nicht-dominierter Individuen in der finalen Population
FKR: Fehlklassifikationsrate

Tabelle 13: Vergleich der finalen Populationen mit verschiedenen Parametereinstellungen und Rekombinationsarten des NHEMOtrees

NHEMOtree-Ansätzen auf die medizinischen Daten dar. Mit NHEMOtree lassen sich stets größere S-Metriken und häufig auch größere Dominanzquotienten finden. Die Fehlklassifikationsrate der Standard-CART-Lösung beträgt 10,49%, die Kosten belaufen sich auf 15,179, sodass ein dominiertes Hypervolumen von 7592 erreicht wird. Der NSGA-II-*Wrapper* findet eine finale Population aus insgesamt 17 nicht-dominierten Individuen mit Fehlklassifikationsraten von 6,99% bis 67,13% und Kosten von 0,008 bis 11,823, die ein Hypervolumen von 9174 dominieren. Von diesen 17 Individuen sind 12 gegenüber dem NHEMOtree mit Standard-Rekombinationsoperator X_S (NHEMOtree (X_S)) nicht-dominiert. Für den entsprechenden Dominanzquotient gilt $\gamma^D(Wrapper, \text{NHEMOtree } (X_S)) = 0{,}92$, sodass NHEMOtree ($X_S$) etwas bessere Ergebnisse liefert als der mehrkriterielle *Wrapper*-Ansatz.

Analog zu Tabelle 7 zeigt Tabelle 15 die NHEMOtree-Individuen mit den bisher besten Algorithmen-Einstellungen, die die Standard-CART-Lösung dominieren. Von den sieben Individuen sind vier Individuen (fett gedruckt) nicht-dominiert durch die finalen Individuen des NSGA-II-*Wrappers*. Die NHEMOtree-Individuen besitzen jedoch mehr Knoten als die Standard-CART-Lösung und die meisten *Wrapper*-Individuen.



(a) Originale Populationsgröße bei X_S (b) Populationsgröße = 400 bei X_S

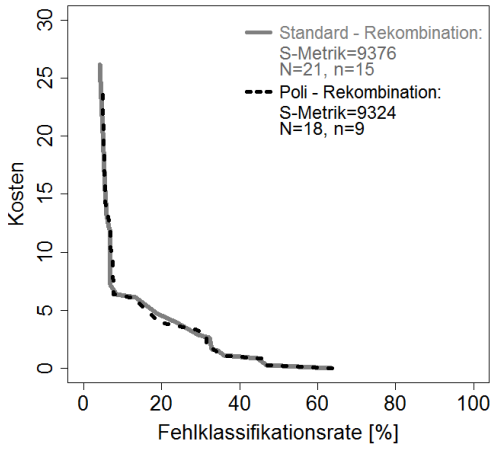
N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 28: Vergleich der Pareto-Front-Approximationen bei Verwendung des Standard- (X_S) und des Brut-Rekombinationsoperators (X_B) in NHEMOtree

Mit NHEMOtree können bei Verwendung von X_{VIM} die S-Metriken stets im Vergleich zum NSGA-II-*Wrapper* verbessert und mehr nicht-dominierte Individuen gefunden werden (vgl. Tabelle 14). Mit Ausnahme von X_{VIM_1} (X_S adaptiert mit der einfachen absoluten Variablenhäufigkeit) und X_{VIM_6} (X_S adaptiert mit der Permutierten Fehlerfreiheit) sind die S-Metriken bei Verwendung von X_{VIM} größer als bei NHEMOtree (X_S). Ferner sind lediglich bei Verwendung von X_{VIM_1} weniger Individuen der finalen Population nicht-dominiert als Individuen bei Gebrauch von X_S. Vor allem die Adaption durch die exponentiell gewichtete relative Variablenhäufigkeit (VIM_5) zeigt sehr gute Ergebnisse. Durch diese Adaption von X_S wird eine finale Population bestehend aus 29 nicht-dominierten Individuen gefunden, von denen 21 nicht durch die finalen Individuen des NHEMOtrees (X_S) ($\gamma^D(X_{VIM_5}, X_S) = 2{,}63$) und 18 nicht durch die finalen Individuen des NSGA-II-*Wrappers* (Finanzen) ($\gamma^D(X_{VIM_5}, Wrapper) = 1{,}50$) dominiert werden (vgl. Tabellen 14 und 16).

Insgesamt werden stets weniger nicht-dominierte Individuen mit dem NSGA-II-*Wrapper* als mit NHEMOtree gefunden (vgl. Tabelle 16). Vor allem im Bereich geringer Fehlklassifikationsraten (<10%) ist NHEMOtree eindeutig überlegen (vgl. Abbildung 30). Im Bereich geringerer Kosten scheint hingegen der mehrkriterielle *Wrapper*-Ansatz häufig besser zu sein.

Abbildung 31 zeigt die approximativen Pareto-Fronten des NHEMOtrees bei Verwendung von X_{VIM} im Vergleich zu NHEMOtree (X_S). Die Pareto-Front-Approximationen der NHEMOtrees sind sehr ähnlich. NHEMOtree (VIM_2) und NHEMOtree (VIM_5) finden Individuen mit ähnlichen geringen Fehlklassifikations-



N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 29: Vergleich der Pareto-Front-Approximationen bei Verwendung des Standard-Rekombinationsoperators (X_S) und des Rekombinationsoperators nach Poli und Langdon (X_P) in NHEMOtree

rate wie NHEMOtree (X_S). Ferner zeigen sich vor allem im Kostenbereich unter 5 häufig bauchigere Pareto-Fronten bei den Läufen mit X_{VIM}, die ebenfalls zu höheren S-Metriken führen.

Die Tabellen A6 bis A11 im Anhang listen analog zu Tabelle 15 die finalen Individuen der NHEMOtrees mit X_{VIM_1} bis X_{VIM_6} auf, die die Standard-CART-Lösung und das NSGA-II-*Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate dominieren. Entsprechend dem Vergleich der diversen NHEMOtree-Ansätze gegenüber dem NSGA-II-*Wrapper* findet NHEMOtree (VIM_5) auch beim Vergleich mit NHEMOtree (X_S) viele Individuen, die nicht durch die Individuen des NHEMOtrees (X_S) dominiert werden.

In den Abbildungen 41 bis 47 im Anhang sind ebenfalls die Individuen mit geringster LOOCV-Fehlklassifikationsrate der NHEMOtrees mit den verschiedenen X_{VIM} dargestellt. Mit Ausnahme von X_{VIM_2} sind die Individuen der NHEMOtrees mit X_{VIM} kleiner als das entsprechende Individuum mit geringster LOOCV-Fehlklassifikationsrate des NHEMOtrees (X_S). Nichtsdestotrotz sind diese Bäume stets größer als das NSGA-II-*Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate (vgl. Abbildung 20).

Insgesamt ist also NHEMOtree dem NSGA-II-*Wrapper* sowie CART überlegen (vgl. Tabelle 14 und 16). Vor allem mit der Adaption von X_S durch die exponentiell gewichtete relative Variablenhäufigkeit (VIM_5) erzielt NHEMOtree gute Ergebnisse und viele Individuen, die nicht durch die Individuen des NHEMOtrees (X_S) domi-

	N	S-Metrik	FKR [%]	Kosten	γ^D
CART	1	7592	10,49	15,179	
Wrapper	17	9174	6,99 - 67,13	0,008 - 11,823	0,92
NHEMOtrees					
X_S	21	9376	4,22 - 63,47	0,008 - 26,201	1
X_{VIM_1}	16	9302	5,43 - 63,76	0,008 - 17,058	1,71
X_{VIM_2}	28	9458	3,30 - 63,26	0,008 - 24,864	2,00
X_{VIM_3}	26	9429	3,50 - 62,59	0,008 - 18,861	1,21
X_{VIM_4}	20	9391	4,17 - 64,71	0,008 - 20,134	1,30
X_{VIM_5}	29	9494	2,81 - 62,38	0,008 - 24,796	2,63
X_{VIM_6}	22	9329	4,88 - 64,85	0,008 - 22,412	0,82

N: Nicht-dominierte Individuen in der finalen approximierten Pareto-Front
FKR: Fehlklassifikationsrate
γ^D: Dominanzquotient bezüglich NHEMOtree (X_S)

Tabelle 14: Vergleich der finalen Populationen von CART, NSGA-II-*Wrapper* und NHEMOtree mit verschiedenen Rekombinationsoperatoren

niert werden. Insbesondere im Kostenbereich unter 5 erzielt NHEMOtree mit X_{VIM} bauchigere Pareto-Fronten als bei Verwendung von X_S.

8.3.6 NHEMOtree mit Online Convergence Detection

Bisher wurde mit NHEMOtree stets eine feste Generationenanzahl durchlaufen. Wird hingegen das OCD-Abbruchkriterium (vgl. Abschnitt 5.3.3) als zusätzliches Abbruchkriterium in NHEMOtree integriert, zeigen sich für alle X_{VIM} weniger benötigte Generationen bis zum Abbruch des Laufs im Vergleich zu X_S (vgl. Tabelle 17) NHEMOtree mit OCD wird mit Parametereinstellung E_3 durchgeführt. Allerdings ist die maximale Generationenanzahl auf $g = 100$ erhöht, da NHEMOtree bei $g = 50$ häufig noch nicht konvergiert ist. Wegen der höheren Generationenanzahl bei Verwendung des OCD sind in dieser Anwendung die S-Metriken gegenüber NHEMOtree ohne OCD erhöht. NHEMOtree (VIM_5) benötigt mit durchschnittlich 80 Generationen wesentlich weniger Iterationen bis zur Konvergenz als die anderen NHEMOtrees. Der natürliche Konflikt zwischen Laufzeit und Güte der Pareto-Front ist in Tabelle 17 gut ersichtlich. Die Kosten für die schnellere Konvergenz bei Verwendung von X_{VIM_5} im Vergleich zu X_S drücken sich in einer geringeren S-Metrik aus. Die Spearman-Korrelation zwischen den benötigten Generationen und den S-Metriken liegt bei 0,65. Die beiden Pareto-Front-Approximationen von NHEMOtree (X_S) und NHEMOtree (VIM_5) sind sehr ähnlich. Im Bereich sehr geringer Fehlklassifikationsraten findet NHEMOtree (VIM_5) jedoch Individuen mit geringeren Kosten (vgl. Abbildung 38 im Anhang). Bei Verwendung von X_{VIM_2},

Individuum	Variablen	LOOCV-FKR [%]	Kosten	Antikörperanzahl
1	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ 2x $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$ 2x $X_{4,\mathrm{M}}$ $X_{1,\mathrm{Z}}$	5,95	13,189	6
2	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$ $X_{4,\mathrm{M}}$ $X_{3,\mathrm{M}}$	6,46	12,163	5
3	$X_{19,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 3x $X_{6,\mathrm{Z}}$ $X_{12,\mathrm{Z}}$	**6,95**	**11,789**	4
4	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$	**6,96**	**9,948**	4
5	$X_{19,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 2x $X_{6,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ $X_{7,\mathrm{M}}$	**6,96**	**11,386**	5
6	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	**6,97**	**7,148**	4
7	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	8,41	6,372	4

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle 15: NHEMOtree-Individuen, die die Standard-CART-Lösung dominieren

	X_S	X_{VIM_1}	X_{VIM_2}	X_{VIM_3}	X_{VIM_4}	X_{VIM_5}	X_{VIM_6}
NHEMOtree	13	15	21	18	14	18	14
Wrapper-Ansatz	12	9	10	13	13	12	12
γ^D	1,08	1,67	2,10	1,38	1,08	1,50	1,17

γ^D: Dominanzquotienten von NHEMOtree bezüglich des NSGA-II-*Wrappers*

Tabelle 16: Anzahl der paarweise nicht-dominierten Individuen des NHEMOtrees mit verschiedenen Rekombinationsoperatoren und des NSGA-II-*Wrappers*

X_{VIM_4} und X_{VIM_6} können finale Pareto-Front-Approximationen gefunden werden, die zum einen nach weniger Iterationen erreicht werden und trotzdem eine höhere S-Metrik besitzen als NHEMOtree (X_S).

	N	S-Metrik	FKR [%]	Kosten	Gen.	γ^D
X_S	24	9614	2,07 - 63,37	0,008 - 15,955	98,4	
X_{VIM_1}	30	9551	2,73 - 63,67	0,008 - 18,119	86,2	0,47
X_{VIM_2}	23	9675	1,38 - 63,26	0,008 - 19,112	93,2	1,78
X_{VIM_3}	32	9532	2,78 - 62,21	0,008 - 19,901	85,2	0,14
X_{VIM_4}	27	9619	1,92 - 63,78	0,008 - 16,655	89,8	1,89
X_{VIM_5}	28	9558	2,59 - 62,38	0,008 - 24,164	80,2	0,40
X_{VIM_6}	21	9648	1,39 - 63,60	0,008 - 60,988	91,4	0,86

N: Nicht-dominierte Individuen in der approximierten Pareto-Front
FKR: Fehlklassifikationsrate
Gen.: Durchschnittliche Generationenanzahl bis zur Konvergenz
γ^D: Dominanzquotient bezüglich X_S

Tabelle 17: Vergleich der finalen Populationen der NHEMOtrees mit verschiedenen Rekombinationsoperatoren und OCD-Abbruchkriterium

Eine Übersicht über die Anzahl der NHEMOtree-Individuen, die die Standard-CART-Lösung bzw. das NSGA-II-*Wrapper*-Individuum mit geringster FKR dominieren, liefert Tabelle 18. Es zeigt sich, dass insbesondere die Adaption durch VIM_4 erfolgreich ist, da die finale CART-Lösung und das NSGA-II-*Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate durch die meisten finalen Individuen des NHEMOtree (VIM_4) dominiert werden. Allerdings liegt die durchschnittliche Generationenanzahl bis zur Konvergenz mit 89,9 über der Anzahl bei den NHEMOtrees mit X_{VIM_1}, X_{VIM_3} oder X_{VIM_5}. Die Tabellen der NHEMOtree-Individuen (analog zu Tabelle 15) bei Verwendung des OCD-Abbruchkriteriums, die die Standard-CART-Lösung dominieren, befinden sich im Anhang (vgl. Tabelle A12 bis A18).

Insgesamt erscheint die Adaption von X_S auch bei Verwendung des OCD-Abbruchkriteriums basierend auf allen VIMs als sinnvoll. Vor allem die Adaption von X_S bezüglich VIM_4 und VIM_5 ist erfolgreich mit hohen S-Metriken und geringen durchschnittlichen Generationenanzahlen.

	X_S	X_{VIM_1}	X_{VIM_2}	X_{VIM_3}	X_{VIM_4}	X_{VIM_5}	X_{VIM_6}
Standard-CART-Lösung	10	15	10	11	15	9	9
NSGA-II-*Wrapper*	6	5	4	4	10	4	2

Tabelle 18: Anzahl der NHEMOtree-Individuen bei verschiedenen Rekombinationsarten und bei Verwendung des OCD-Abbruchkriteriums, die die Standard-CART-Lösung bzw. das NSGA-II-*Wrapper*-Individuum mit geringster LOOCV-Fehlklassifikationsrate dominieren

8.3.7 NHEMOtree mit lokaler Cutoff-Optimierung

In NHEMOtree werden die Cutoffs in den Knoten standardmäßig mittels der Gauß-Mutation (vgl. Abschnitt 4.3) optimiert. Vor dem Hintergrund, dass laut Hunter (2002) EAs mit Baum-Repräsentation die Cutoffs im Vergleich zur Baumstruktur weniger gut optimieren wird eine lokale Optimierung zum Auffinden der optimalen Cutoffs je Knotenvariable nach dem Vorbild von CART in NHEMOtree untersucht (vgl. Abschnitt 6.5). Der optimale Cutoff wird in NHEMOtree mit lokaler Cutoff-Optmierung sukzessive ausgehend vom Wurzelknoten für die einzelnen Knoten entweder basierend auf der Reduzierung der Fehlklassifikationsrate oder auf der Maximierung der Gini-Wichtigkeit (vgl. Gleichung (1) in Abschnitt 2.1) gesucht.

Lokale Cutoff-Optimierung basierend auf der Fehlklassifikationsrate

Mit NHEMOtree unter Verwendung der lokalen Cutoff-Optimierung basierend auf der Fehlklassifikationsrate ($\text{NHEMOtree}_{\text{FKR}}$) lassen sich stets größere S-Metriken im Vergleich zum Standard-NHEMOtree (NHEMOtree ohne lokale Cutoff-Optimierung) finden. Es werden mit allen X_{VIM} stets mehr als 97% des maximal möglichen Hypervolumens dominiert. Im Vergleich dazu waren es bei Verwendung des Standard-NHEMOtrees maximal 94,9% mit X_{VIM_5} (vgl. Tabelle 14 und 19). Diese stark erhöhten S-Metriken resultieren vor allem aus den deutlich geringeren Fehlklassifikationsraten von unter einem Prozent (vgl. Abbildung 32). $\text{NHEMOtree}_{\text{FKR}}$ (VIM_5) findet sogar ein Individuum mit einer kreuzvalidierten Fehlklassifikationsrate von 0%. Dieser Baum ist in Abbildung 53 im Anhang dargestellt und präsentiert sich als großer Baum mit vielen Knoten, schwieriger Interpretation und lässt trotz der Kreuzvalidierung auf Überanpassung der Daten vermuten. Die Bäume mit der jeweils geringsten Fehlklassifikationsrate, die durch $\text{NHEMOtree}_{\text{FKR}}$ unter Verwendung der verschiedenen X_{VIM} gefunden werden, sind ebenfalls im Anhang in den Abbildungen 48 bis 54 dargestellt. Diese sind um durchschnittlich 6 Knoten größer als die Individuen mit geringster LOOCV-Fehlklassifikationsrate der Standard-NHEMOtrees (vgl. Abbildungen 41 bis 47 im Anhang). Allerdings sind die finalen

	N	S-Metrik	FKR [%]	Kosten	γ^D
Lokale Cutoff-Optimierung auf Basis der FKR					
X_S	33	9752	0,68 - 62,40	0,008 - 23,778	
X_{VIM_1}	34	9757	0,68 - 62,36	0,008 - 21,890	1,85
X_{VIM_2}	30	9779	0,68 - 62,54	0,008 - 28,865	3,43
X_{VIM_3}	32	9773	0,68 - 62,32	0,008 - 31,991	1,83
X_{VIM_4}	32	9796	0,52 - 61,69	0,008 - 24,813	3,29
X_{VIM_5}	30	9823	0 - 62,21	0,008 - 20,143	1,83
X_{VIM_6}	33	9768	0,68 - 62,18	0,008 - 25,619	2,00
Lokale Cutoff-Optimierung auf Basis der Gini-Wichtigkeit					
X_S	27	9775	0,68 - 62,66	0,008 - 20,465	
X_{VIM_1}	19	9836	0 - 61,86	0,008 - 16,940	1,89
X_{VIM_2}	25	9835	0 - 62,43	0,008 - 13,936	0,93
X_{VIM_3}	25	9843	0 - 62,07	0,008 - 17,962	3,17
X_{VIM_4}	24	9846	0 - 62,51	0,008 - 8,722	2,17
X_{VIM_5}	30	9844	0 - 62,36	0,008 - 16,596	2,56
X_{VIM_6}	28	9826	0 - 63,08	0,008 - 25,441	2,86

N: Nicht-dominierte Individuen in der approximierten Pareto-Front
FKR: Fehlklassifikationsrate
γ^D: Dominanzquotient bezüglich X_S

Tabelle 19: Vergleich der finalen Populationen der NHEMOtrees mit lokaler Cutoff-Optimierung und verschiedenen Rekombinationsoperatoren

Populationen mit 30 und mehr Individuen bei $\text{NHEMOtree}_{\text{FKR}}$ vielfältiger und die Laufzeit mit der zusätzlichen lokalen Cutoff-Optimierung verlangsamt. Durch die Verwendung der X_{VIM} werden auch in $\text{NHEMOtree}_{\text{FKR}}$ schwach verbesserte Ergebnisse (Dominanzquotienten größer Eins und höheren S-Metriken) mit mehr nicht-dominierten Individuen im Vergleich zu $\text{NHEMOtree}_{\text{FKR}}$ (X_S) erzielt (vgl. Tabelle 19 und Abbildung 39).

Lokale Cutoff-Optimierung basierend auf der Gini-Wichtigkeit

Bei der lokalen Cutoff-Optimierung basierend auf der Gini-Wichtigkeit ($\text{NHEMOtree}_{\text{Gini}}$) sind die S-Metriken ebenfalls gegenüber Standard-NHEMOtree erhöht. Mit $\text{NHEMOtree}_{\text{Gini}}$ in Verbindung mit X_{VIM} werden stets mehr als 98% des maximal möglichen Hypervolumens dominiert. Analog zum $\text{NHEMOtree}_{\text{FKR}}$ erreicht $\text{NHEMOtree}_{\text{Gini}}$ unter Verwendung der VIMs, die die Variablenposition im Baum berücksichtigen, die höchsten S-Metriken. Die minimalen Fehlklassifikationsraten betragen mit Ausnahme von $\text{NHEMOtree}_{\text{Gini}}$ (X_S) 0%, sodass mit jedem X_{VIM} mindestens ein Individuum mit einer fehlerfreien, kreuzvalidierten Klassifikation gefunden wird. Die zugehörigen Bäume sind jedoch bei Verwendung aller X_{VIM}

groß (vgl. Abbildungen 55 bis 61 im Anhang). NHEMOtree$_{\text{Gini}}$ (X_S) enthält mit 19 Knoten bestehend aus 11 Variablen die meisten. Ebenfalls viele Knoten enthalten die Individuen von NHEMOtree$_{\text{Gini}}$ (VIM_3) und NHEMOtree$_{\text{Gini}}$ (VIM_6) (16 bzw. 17 Knoten). Das Individuum mit 0% Fehlklassifikationsrate und den wenigsten Knoten findet NHEMOtree$_{\text{Gini}}$ (VIM_1). Allerdings ist das Individuum insgesamt teurer als die entsprechenden Individuen unter Verwendung von X_{VIM_2}, X_{VIM_4} und X_{VIM_5}.

Auch in NHEMOtree$_{\text{Gini}}$ führt die Verwendung von X_{VIM} noch zu leichten Verbesserungen. Alle Pareto-Front-Approximationen der NHEMOtree$_{\text{Gini}}$ sind sehr ähnlich mit leichtem Vorteil im Bereich sehr niedriger Fehlklassifikationsraten, vor allem bei X_{VIM_4} (vgl. Abbildung 40 im Anhang). Die approximierten Pareto-Fronten von NHEMOtree$_{\text{Gini}}$ (X_S), NHEMOtree$_{\text{Gini}}$ (VIM_2) und NHEMOtree$_{\text{Gini}}$ (VIM_5) besitzen einen sehr ähnlichen Verlauf, jedoch ist die finale Population bei Verwendung von X_{VIM_5} vielfältiger und etwas näher am Koordinatenkreuz als bei Verwendung der anderen genannten Rekombinationsarten. Mit Ausnahme von NHEMOtree$_{\text{Gini}}$ (VIM_2) werden bei Verwendung von X_{VIM} mehr Individuen nicht-dominiert im Vergleich zum NHEMOtree$_{\text{Gini}}$ (X_S). Dies gilt insbesondere für X_{VIM_5}.

Vergleich der lokalen Cutoff-Optimierungsverfahren in NHEMOtree

Trotz höherer S-Metriken findet NHEMOtree$_{\text{Gini}}$ meistens Pareto-Front-Approximationen bestehend aus weniger nicht-dominierten Individuen als NHEMOtree$_{\text{FKR}}$ (vgl. Tabelle 19). Eine Ausnahme ist NHEMOtree$_{\text{Gini}}$ (VIM_5) mit gleich vielen nicht-dominierten Individuen und einer leicht höheren S-Metrik. Ein weiterer Vergleich der finalen nicht-dominierten Lösungen zeigt, dass trotz der häufig größeren finalen Populationen des NHEMOtree$_{\text{FKR}}$ die Individuen des NHEMOtree$_{\text{Gini}}$ häufiger nicht-dominiert sind (vgl. Tabelle 20). Am Beispiel von X_{VIM_5} zeigt sich mit $\gamma^D(FKR, Gini) = 0{,}48$ die Überlegenheit von NHEMOtree$_{\text{Gini}}$ gegenüber NHEMOtree$_{\text{FKR}}$. Dies spiegelt sich ebenfalls in Abbildung 32 mit den Pareto-Front-Approximationen mit und ohne lokale Cutoff-Optimierung bei verschiedenen Rekombinationsoperatoren wider. Die Laufzeiten von NHEMOtree$_{\text{FKR}}$ und NHEMOtree$_{\text{Gini}}$ sind ähnlich.

Allerdings ist eine Überanpassung trotz Kreuzvalidierung bei NHEMOtree mit lokaler Cutoff-Optimierung wahrscheinlich. Durch die lokale Cutoff-Optimierung wird an jedem Knoten versucht, neue Knoten oder Blätter zu generieren, die maximal homogen sind. Ein Stopp-Mechanismus, wie etwa das Pruning in CART, ist in NHEMOtree nicht integriert. Nur die maximale Knotenanzahl und eine zweite Fitnessfunktion, die große Bäume bestraft, können ein übermäßiges Aufteilen der

		X_S	X_{VIM_1}	X_{VIM_2}	X_{VIM_3}	X_{VIM_4}	X_{VIM_5}	X_{VIM_6}
Standard / FKR	N	0 / 33	2 / 31	0 / 30	0 / 32	0 / 32	0 / 30	0 / 33
	γ^D	0	0,06	0	0	0	0	0
Standard / Gini	N	4 / 26	4 / 19	4 / 25	3 / 22	3 / 24	2 / 30	4 / 26
	γ^D	0,15	0,21	0,16	0,14	0,13	0,07	0,15
FKR / Gini	N	14 / 21	13 / 14	16 / 16	8 / 17	15 / 10	12 / 25	14 / 18
	γ^D	0,67	0,93	1	0,47	1,50	0,48	0,78

Tabelle 20: Anzahl der paarweise nicht-dominierten NHEMOtree-Individuen N bei verschiedenen Cutoff-Optimierungen und Rekombinationsoperatoren, sowie die Dominanzquotienten γ^D von NHEMOtree mit dem ersten Verfahren bezüglich des zweiten Verfahrens

Beobachtungen verhindern. Im Standard-NHEMOtree hingegen führen ungünstige Cutoffs, d.h. Cutoffs, die einen Knoten nicht sinnvoll teilen bzw. nicht in dem Knoten optimal teilen, zu leeren Blättern, die dann durch den Algorithmus entfernt werden. Auf diese Weise werden die Bäume häufiger verkleinert und der Überanpassung wird so indirekt entgegengewirkt.

Die lokale Cutoff-Optimierung in NHEMOtrees führt zwar zu höheren S-Metriken und vielfältigeren Individuen in der finalen Population im Vergleich zum Standard-NHEMOtree (vgl. Tabelle 19 und Abbildung 32) und erzielt bei Verwendung von X_{VIM} sogar noch bessere Ergebnisse, jedoch erhöht die zusätzliche Cutoff-Optimierung die Gefahr der Überanpassung. Außerdem führt die zusätzliche Cutoff-Optimierung wie vermutet zu einer verlangsamten Laufzeit (vgl. Abschnitt 6.5), sodass die Cutoff-Optimierung in NHEMOtree durch die Gauß-Mutation bevorzugt werden sollte.

8.4 Fazit

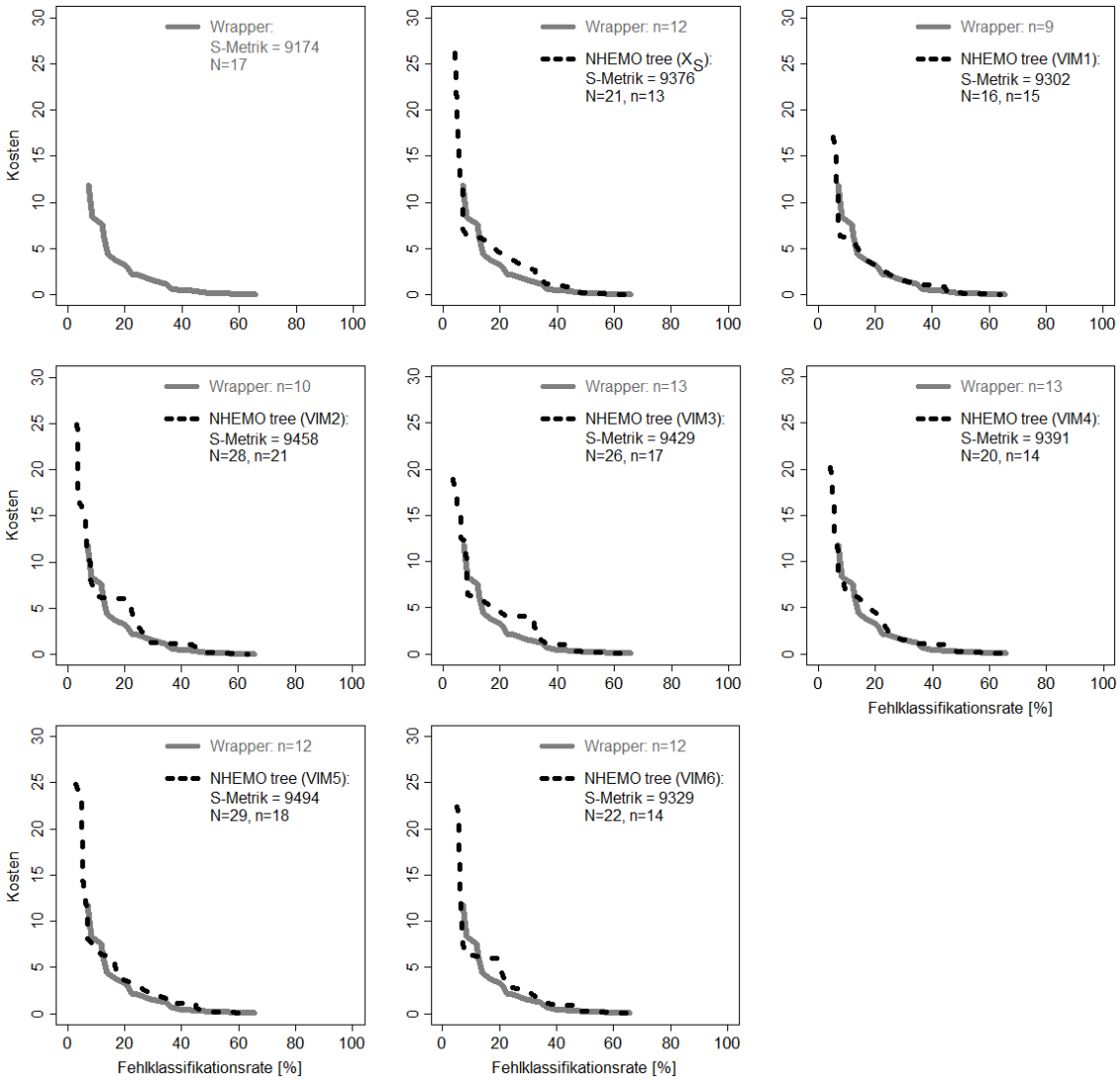
Bei der Auswertung des medizinischen Datensatzes zeigt sich ein großer Vorteil des mehrkriteriellen *Wrapper*-Ansatzes bei Verwendung des zweikriteriellen Optimierungsproblems mit der Fehlklassifikationsrate und den finanziellen Kosten als Fitnessfunktionen gegenüber der Standard-CART-Lösung (vgl. Tabellen 7, A2 und A3). Die Ergebnisse des mehrkriteriellen *Wrapper*-Ansatzes sind für die verschiedenen EMOAs (NSGA-II, *Steady-State* NSGA-II, SMS-EMOA) sehr ähnlich (vgl. Tabellen 8 und 10). Allerdings ist die Laufzeit des NSGA-II im Vergleich zu den anderen beiden EMOAs kürzer, sodass der NSGA-II-*Wrapper* im weiteren Verlauf der Arbeit bevorzugt wird. Ferner liefern vor allem Parametereinstellungen mit hohen Mutationsraten gute Ergebnisse (vgl. Tabellen 6 und 8).

Im Vergleich zum mehrkriteriellen *Wrapper*-Ansatz mit den besten Operatoren und Parametereinstellungen liefert der hier entwickelte NHEMOtree bessere Ergebnisse. Die approximierten Pareto-Fronten bestehen aus mehr nicht-dominierten Individuen, erzielen höhere Dominanzquotienten und führen zu höheren S-Metriken (vgl. Abbildung 30 und Tabellen 14 und 16). Vor allem im Bereich geringer Fehlklassifikationsraten (<10%) ist NHEMOtree überlegen. In NHEMOtree sind ebenfalls Einstellungen mit hohen Mutationsraten erfolgreich (vgl. Tabellen 11 und 12, sowie Abbildung 23). Zur Initialisierung bestätigt sich die *ramped-half-and-half*-Methode im Vergleich zu bereits optimierten initialen Populationen (vgl. Abbildung 25). Bei der Selektion der Eltern schneiden Turnier- und die Winkler-Selektion gleich gut ab (vgl. Abbildung 27), jedoch ist die Laufzeit bei der Winkler-Selektion leicht erhöht. Bei der Rekombination erzielt der Standard-Rekombinationsoperator X_S gute Ergebnisse, die weder durch die Brut-Rekombination X_B noch durch die Rekombination X_P nach Poli und Langdon (1998a) verbessert werden kann (vgl. Abbildungen 28 und 29). Da jedoch der Rekombinationsoperator in EAs mit Baum-Repräsentation als Ursache für die erfolgreiche Erstellung von *Building Block*s und somit auch für deren Effektivität angesehen wird (vgl. Abschnitt 4.3), wird in dieser Arbeit X_S adaptiert, um dadurch bessere *Building Block*s und schlussendlich auch bessere Pareto-Front-Approximationen zu generieren.

Der VIM-basierte Rekombinationsoperator X_{VIM} kann die Ergebnisse des NHEMOtrees tatsächlich verbessern (vgl. Tabelle 14). Insbesondere durch X_{VIM} mit VIMs, die die Variablenposition im Baum betrachten (VIM_4 und VIM_5), werden Pareto-Front-Approximationen mit guter Diversität und mit höheren S-Metriken sowohl im Vergleich zur Standard-CART-Lösung als auch zum mehrkriteriellen *Wrapper*-Ansatz (vgl. Tabelle 14 und Abbildung 31) erzielt. Außerdem kann bei Verwendung von X_{VIM_5} die Konvergenz des NHEMOtrees beschleunigt werden (vgl. Tabelle 17). Die Berücksichtigung der Häufigkeit einer Variable und insbesondere die Berücksichtigung ihrer Position im Baum erscheint besser für die Adaption von X_S geeignet zu sein als die alleinige Bewertung der Anwesenheit einer Variable im Entscheidungsbaum oder die Permutierte Fehlerfreiheit.

Die zusätzliche Optimierung der Cutoffs innerhalb des NHEMOtrees führt zu höheren S-Metriken im Vergleich zum Standard-NHEMOtree (vgl. Tabelle 19 und Abbildung 32). Die Verwendung von X_{VIM} insbesondere basierend auf VIM_5 weist Verbesserungspotential mit großen S-Metriken und vielen finalen Individuen in der approximierten Pareto-Front auf. Ein Vergleich der finalen Lösungen zeigt, dass trotz der oft größeren finalen Populationen des $NHEMOtree_{FKR}$ die Individuen des $NHEMOtree_{Gini}$ häufiger nicht-dominiert und die S-Metriken leicht erhöht sind (vgl. Tabelle 20). $NHEMOtree_{FKR}$ (VIM_5) sowie die meisten anderen $NHEMOtree_{Gini}$

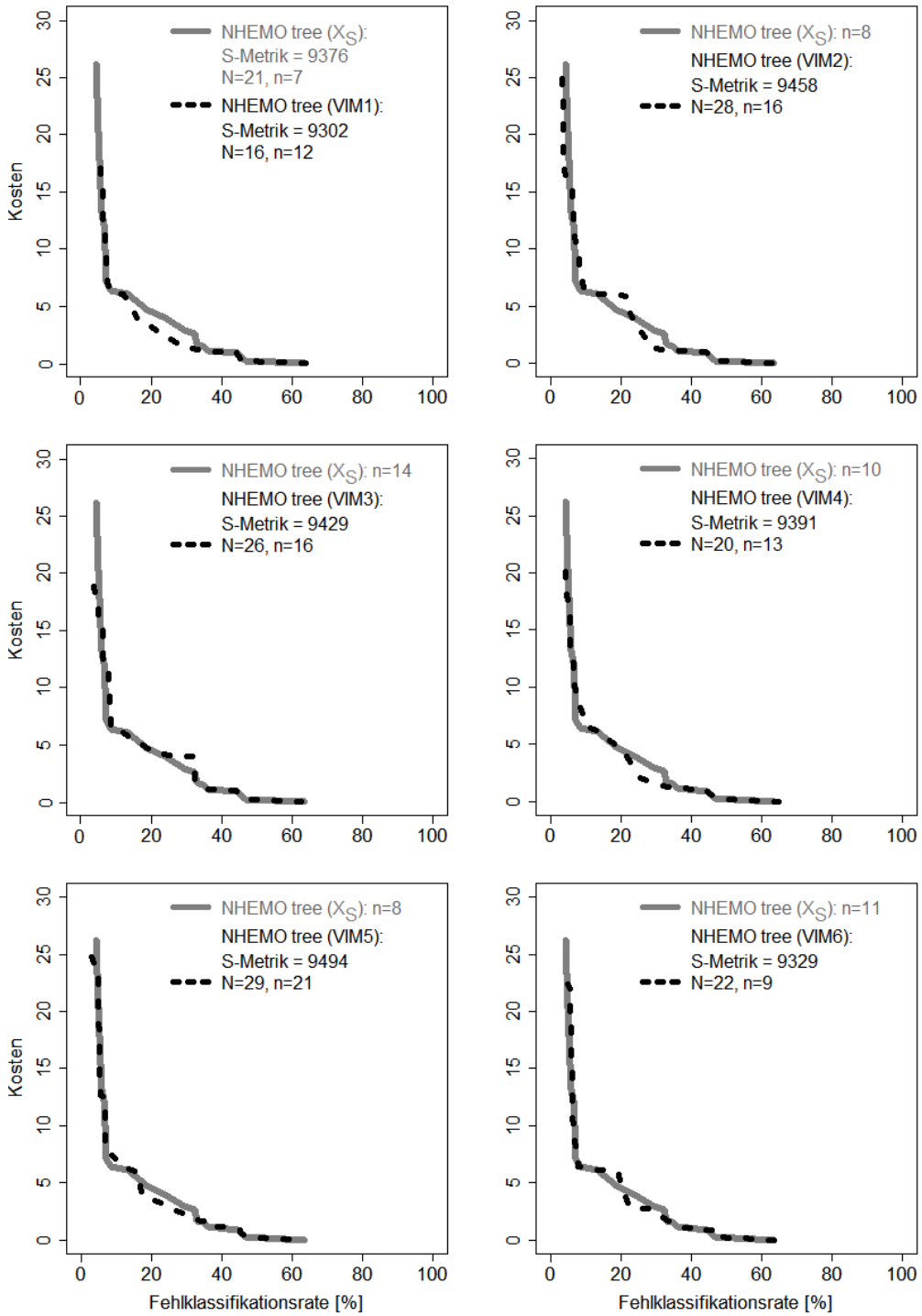
finden sehr große Individuen mit kreuzvalidierten Fehlklassifikationsraten von 0% (vgl. Abbildungen 48 bis 61 im Anhang). Diese zusätzliche Cutoff-Optimierung erhöht somit die Gefahr der Überanpassung.



N: Anzahl nicht-dominierter Individuen in der finalen Population

n: Anzahl nicht-dominierter Individuen im Vergleich zum mehrkriteriellen *Wrapper*-Ansatz

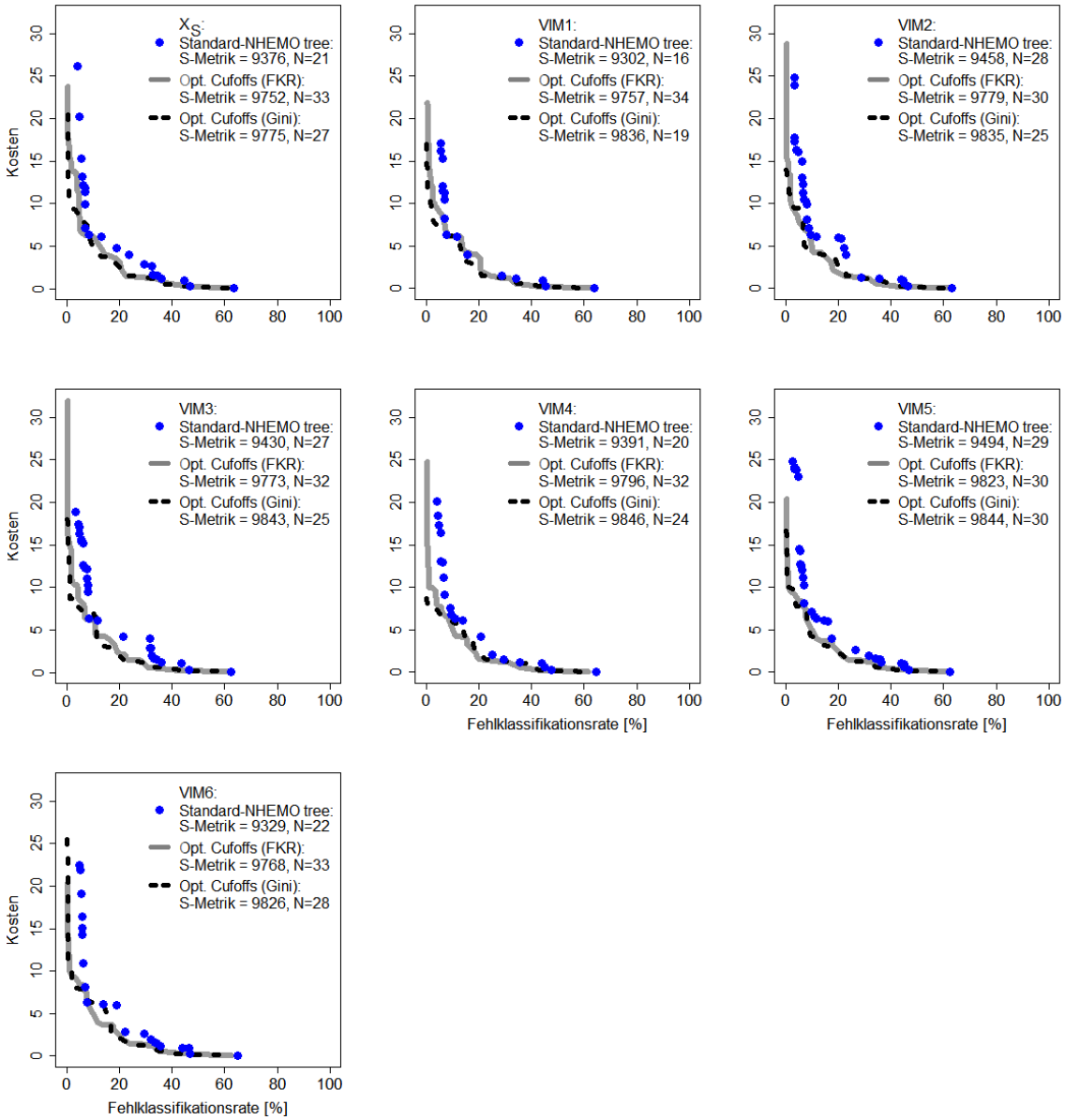
Abbildung 30: Vergleich der Pareto-Front-Approximationen des NSGA-II-*Wrapper* und NHEMOtree mit verschieden adaptierten Rekombinationsoperatoren



N: Anzahl nicht-dominierter Individuen in der finalen Population

n: Anzahl nicht-dominierter Individuen im Vergleich zum NHEMOtree (X_S)

Abbildung 31: Vergleich der Pareto-Front-Approximationen des NHEMOtrees mit und ohne Adaption des Standard-Rekombinationsoperators



N: Anzahl nicht-dominierter Individuen in der finalen Population

n: Anzahl nicht-dominierter Individuen im Vergleich zum mehrkriteriellen *Wrapper*-Ansatz

Abbildung 32: Vergleich der Pareto-Front-Approximationen des NHEMOtrees mit und ohne lokale Cutoff-Optimierung und verschieden adaptierten Rekombinationsoperatoren

9 Simulationsstudie

Die Hypothese der Simulationsstudie ist, dass NHEMOtree mit VIM-basiertem Rekombinationsoperator X_{VIM} schneller konvergiert als mit dem Standard-Rekombinationsoperator X_S. Dies wird sowohl für den Standard-NHEMOtree als auch für $NHEMOtree_{FKR}$ und $NHEMOtree_{Gini}$ überprüft. In Abschnitt 9.1 wird zunächst die Simulation der Daten beschrieben. Im nachfolgenden Abschnitt 9.2 werden die Ergebnisse der Simulationsstudie erläutert und in Abschnitt 9.3 zusammengefasst.

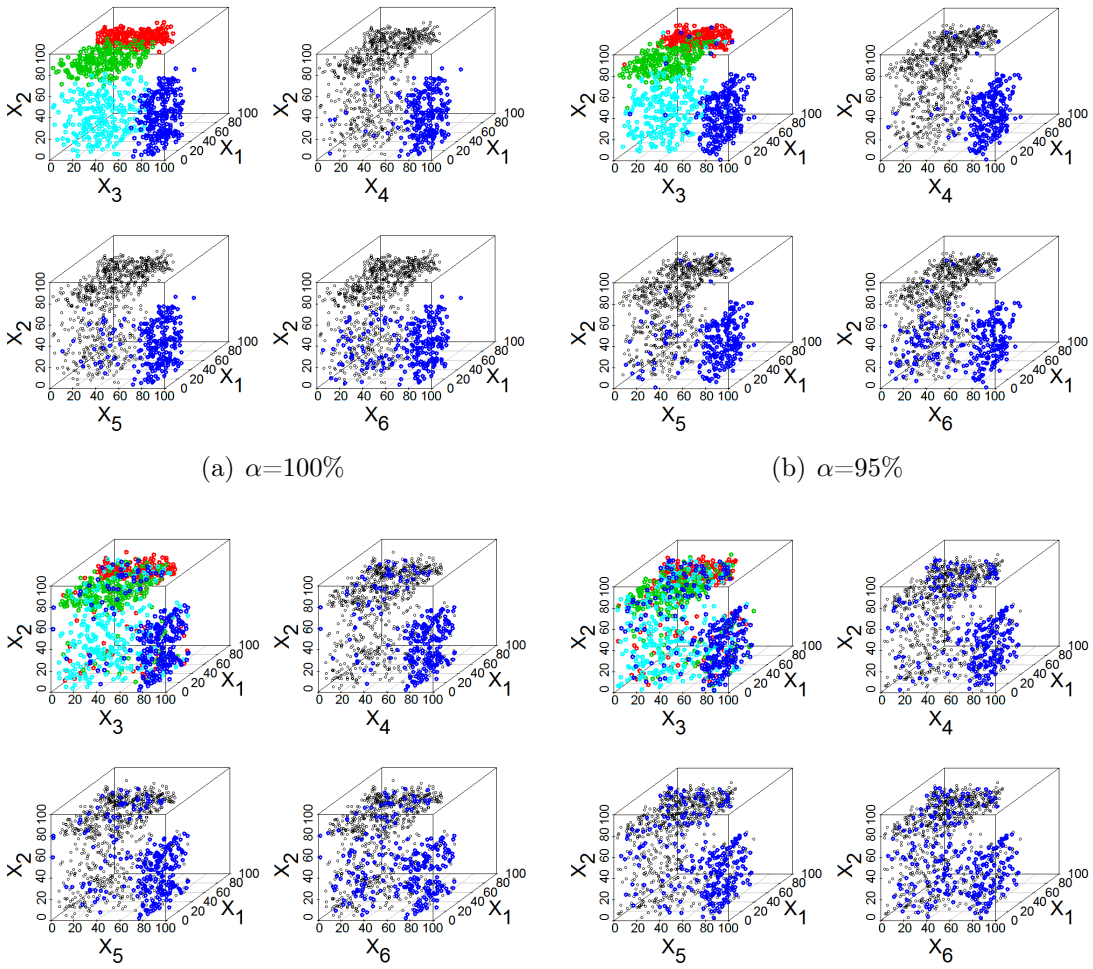
9.1 Simulation der Daten

Die Simulationsstudie besteht aus vier Szenarien für 300 Datensätze à 20 Variablen und 1000 Beobachtungen aus vier gleichgroßen Klassen. Die Simulation der Daten basiert auf einem Vier-Klassen-Problem, das durch drei diskriminierende Variablen X_1, X_2 und X_3 gut trennbar ist. X_1, X_2 und X_3 werden für die Beobachtungen der zugehörigen Klasse aus einer Normalverteilung mit Mittelwert $\mu = 80$ und Varianz $\sigma^2 = 25$ gezogen. Für Beobachtungen, die nicht der entsprechenden Klasse zugeordnet werden sollen, werden X_1, X_2 und X_3 aus einer Gleichverteilung mit Minimum 0 und einem variablen Maximum in Abhängigkeit des minimalen Wertes (M_i) der normalverteilten Zufallszahlen X_i, $i = 1, \ldots, 3$, gezogen.

Beobachtungen mit $X_1 \geq M_1$ werden der Klasse 1, Beobachtungen mit $X_1 < M_1 \wedge X_2 \geq M_2$ der Klasse 2, Beobachtungen mit $X_1 < M_1 \wedge X_3 \geq M_3$ der Klasse 3 und alle weiteren Beobachtungen der Klasse 4 zugeordnet. Das Verrauschen der Klassenzugehörigkeit erfolgt mittels einer Binomialverteilung, sodass die Klassen nur mit Wahrscheinlichkeit α korrekt zugeteilt sind.

Ferner werden die Variablen X_4, X_5 und X_6 basierend auf X_3 gebildet. Das Verrauschen der drei Variablen erfolgt durch Zuordnung eines gleichverteilten Wertes zwischen 0 und M_3 für 5%, 10% und 30% der Beobachtungen aus Klasse 3. X_4, X_5 und X_6 trennen die dritte Klasse entsprechend schlechter von der vierten Klasse als X_3. X_7 bis X_{20} sind Störvariablen und werden als Rauschen aus einer Normalverteilung gezogen. Abbildung 33 zeigt die simulierten Daten mit verschiedenen Trenngüten α für die Variablentrios $\{X_1, X_2, X_3\}$, $\{X_1, X_2, X_4\}$, $\{X_1, X_2, X_5\}$, $\{X_1, X_2, X_6\}$. Beobachtungen, die der vierten Klasse zugeordnet sind, sind durch blaue Kreise dargestellt. Es ist gut zu sehen, wie die Streuung der Beobachtungen dieser Klasse in Abhängigkeit von der dritten Variable im Variablentrio zunimmt.

Die Variablenkosten korrelieren mit der Vorhersagegüte, sodass informative Variablen teurer sind als solche mit geringem Informationsgehalt. Für die Variablen X_7 bis X_{20} werden identische Kosten festgesetzt. Variable X_6 ist doppelt so teuer wie



(a) α=100% (b) α=95%

(c) α=70% (d) α=50%

Abbildung 33: Simulationsdaten mit verschiedenen Trenngüten α der vier farbig markierten Klassen. Abbildung (a) zeigt perfekt trennbare Klassen bei Verwendung des Variablentrios $\{X_1, X_2, X_3\}$. Die Trennbarkeit der Klassen nimmt sukzessive von Abbildung (a) bis (d) ab, sowie innerhalb jeder Teilabbildung in Abhängigkeit von der dritten Variable im Variablentrio.

die Störvariablen, Variable X_5 1,5-mal teurer als X_6 usw.. Abschließend werden die Kosten für eine bessere Übersichtlichkeit auf 100 normiert. Die Kosten und die Fehlklassifikationsraten der einzelnen Variablen bei verschiedenen Trenngüten der Daten zeigt Tabelle 21.

Aufgrund der Konstruktion der Variablen und der Kosten sollte das Individuum mit geringster LOOCV-Fehlklassifikationsrate und höchsten Kosten aus dem Variablentrio $\{X_1, X_2, X_3\}$ bestehen. Insgesamt sollten die Individuen $\{X_1, X_2, X_3\}$, $\{X_1, X_2, X_4\}$, $\{X_1, X_2, X_5\}$ und $\{X_1, X_2, X_6\}$ sowie Individuen bestehend aus weniger Variablen durch NHEMOtree gefunden werden.

Die Optimierung der simulierten Daten erfolgt mittels NHEMOtree mit den für den medizinischen Datensatz bewährten Parametereinstellungen ($g = 50$ Generationen,

Variable	Kosten	Trennung von Rest vs.	$\alpha = 100\%$ Trenngüte lokal	$\alpha = 100\%$ Trenngüte total	$\alpha = 95\%$ Trenngüte lokal	$\alpha = 95\%$ Trenngüte total	$\alpha = 70\%$ Trenngüte lokal	$\alpha = 70\%$ Trenngüte total	$\alpha = 50\%$ Trenngüte lokal	$\alpha = 50\%$ Trenngüte total
X_1	13,16	Klasse 1	0%	50%	2%	51%	11%	57%	20%	62%
X_2	13,16	Klasse 1/2	0%	50%	2%	51%	15%	57%	25%	73%
X_3	13,16	Klasse 3	0%	50%	2%	51%	11%	57%	19%	59%
X_4	10,53	Klasse 3	1%	51%	3%	52%	12%	58%	20%	61%
X_5	7,89	Klasse 3	3%	52%	4%	54%	13%	59%	21%	62%
X_6	5,26	Klasse 3	8%	58%	9%	58%	17%	63%	23%	66%
X_7 - X_{20}	2,63	-	-	>70%	-	>70%	-	>70%	-	>70%

lokal: Trenngüte bezüglich der entsprechenden Klasse im Vergleich zu den restlichen Klassen
total: Trenngüte des Vier-Klassen-Problems

Tabelle 21: Kosten und Fehlklassifikationsraten der simulierten Variablen bei verschiedenen Trenngüten α der Daten

50% Mutations- und Rekombinationsrate, $\nu_{max} = 10$ maximal erlaubten Knoten je Baum). Anstelle der 200 Individuen je Population wird jedoch eine geringere Populationsgröße von 100 Individuen untersucht, um den Rechenaufwand zu verringern. Die Elternauswahl erfolgt durch eine Turnierselektion mit Turniergröße $\tau = 4$ und es werden die Rekombinationsoperatoren X_S und X_{VIM} verwendet. Die Vorhersagegüte wird mit einer fünffachen Kreuzvalidierung berechnet. Ein Abbruch der Optimierung erfolgt durch das OCD-Abbruchkriterium oder bei Erreichen der maximalen Generationenanzahl.

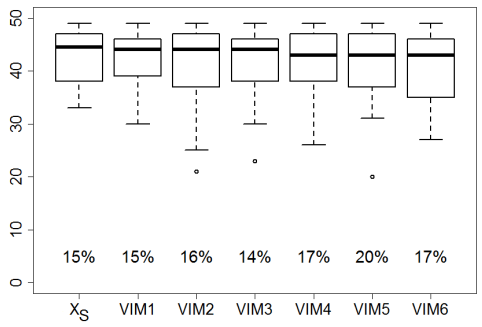
9.2 Ergebnisse der Simulationsstudie

9.2.1 NHEMOtree mit standardmäßiger Cutoff-Optimierung

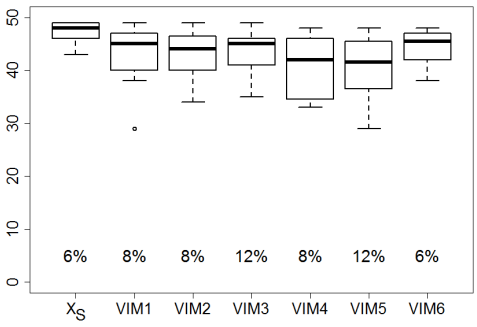
Die Ergebnisse der Simulationsstudie von NHEMOtree mit standardmäßiger Cutoff-Optimierung (Gauß-Mutation zur Adaption der Cutoffs) bezüglich vier verschiedenen Trenngüten der Daten sind in Tabelle 22 dargestellt. Die höhere Komplexität der Separierung in den Szenarien mit schlechterer Trenngüte ist klar ersichtlich. Die S-Metriken sowie die Häufigkeit der Läufe mit vorzeitiger Konvergenz nimmt sukzessize mit schlechter werdenden Trenngüten der Daten ab. Gleiches gilt für die Auswertung der Simulationsdaten mit CART und dem mehrkriteriellen *Wrapper*-Ansatz. Je schlechter die Trenngüte der Daten ist, desto geringer ist die erreichte S-Metrik. Ferner drückt sich der Nachteil der CART-Lösung sowie der mehrkriteriellen *Wrapper*-Lösung gegenüber NHEMOtree in den deutlich geringeren S-Metriken γ^S aus (vgl. Tabelle 22). Eine eindeutige Aussage ist anhand des Dominanzquotienten nicht zu treffen. Jedoch wird der mehrkriterielle *Wrapper*-Ansatz durch das OCD-Abbruchkriterium häufiger und früher beendet als NHEMOtree.

Im Simulationsszenario mit optimaler Trenngüte der Daten ($\alpha = 100\%$) sind die Ergebnisse zwischen NHEMOtree mit und ohne Adaption von X_S sehr ähnlich mit S-Metriken von 8600 und 14% bis 20% vorzeitig beendeten Läufen. Ein Vorteil für NHEMOtree mit X_{VIM} besteht insbesondere bei Verwendung von VIM_5. Mit NHEMOtree (VIM_5) werden am häufigsten die Läufe durch das OCD-Abbruchkriterium abgebrochen. Ferner erfolgt die vorzeitige Konvergenz bereits nach weniger Generationen als bei der Adaption durch andere VIMs (vgl. Abbildung 34(a)).

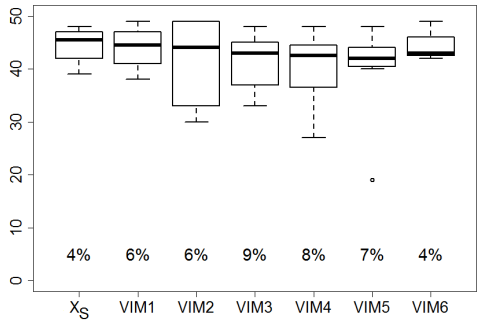
Bei schlechter werdender Trenngüte konvergiert NHEMOtree mit jedem Rekombinationsoperator seltener vorzeitig und der Vorteil der Adaption von X_S wird deutlicher (vgl. Abbildungen 34(b), 34(c) und 34(d)). Die Läufe mit X_{VIM_2} bis X_{VIM_5} konvergieren doppelt so häufig vorzeitig als bei Verwendung von X_S. Ferner senkt die Adaption von X_S die mittlere Anzahl nötiger Generationen bis zur Konvergenz. Dies gilt insbesondere für die VIMs, die die Variablenposition im Baum berücksichtigen (VIM_4 und VIM_5). Die Güte der Adaption basierend auf VIM_6 (Permutierte Fehlerfreiheit) fällt hingegen weniger gut aus. Ursächlich hierfür könnte die Fokussierung auf nur ein Zielkriterium bei der Bewertung der Variablengüte sein. VIM_6 misst die Güte einer Variable lediglich basierend auf ihrer Vorhersagegenauigkeit, ohne die Kosten für die Vorhersagegenauigkeit zu beachten.



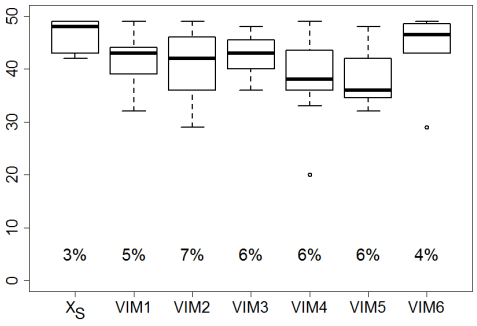
(a) Trenngüte $\alpha = 100\%$



(b) Trenngüte $\alpha = 95\%$



(c) Trenngüte $\alpha = 70\%$



(d) Trenngüte $\alpha = 50\%$

Die Prozentzahlen unterhalb der Boxplots geben jeweils den Anteil der Läufe mit vorzeitigem Abbruch durch das OCD-Abbruchkriterium an.

Abbildung 34: Boxplots notwendiger Generationen des NHEMOtrees bei Abbruch durch das OCD-Abbruchkriterium, simulierten Daten mit verschiedenen Trenngüten α und verschiedenen Rekombinationsoperatoren

	Trenngüte $\alpha = 100\%$				Trenngüte $\alpha = 95\%$			
	γ^S	γ^D	g	$< g_{\max}$	γ^S	γ^D	g	$< g_{\max}$
CART	6053				6051			
Wrapper	8400	0,08	5 - 25	100%	8083	1,17	5 - 33	100%
NHEMOtree								
X_S	8605		33 - 50	15%	8372		43 - 50	6%
X_{VIM_1}	8601	0,55	30 - 50	15%	8377	1,50	29 - 50	8%
X_{VIM_2}	8604	0,67	21 - 50	16%	8382	2,17	34 - 50	8%
X_{VIM_3}	8609	1,29	23 - 50	14%	8379	3,67	35 - 50	12%
X_{VIM_4}	8603	1,00	26 - 50	17%	8373	0,88	33 - 50	8%
X_{VIM_5}	8603	0,80	20 - 50	20%	8373	1,14	29 - 50	12%
X_{VIM_6}	8597	0,50	27 - 50	17%	8373	1,00	38 - 50	6%

	Trenngüte $\alpha = 70\%$				Trenngüte $\alpha = 50\%$			
	γ^S	γ^D	g	$< g_{\max}$	γ^S	γ^D	g	$< g_{\max}$
CART	4921				3988			
Wrapper	6592	1,00	5 - 33	100%	5431	0,56	6 - 21	100%
NHEMOtree								
X_S	7024		39 - 50	4%	5854		42 - 50	3%
X_{VIM_1}	7022	1,14	38 - 50	6%	5880	0,78	32 - 50	5%
X_{VIM_2}	7020	0,75	30 - 50	6%	5869	2,75	29 - 50	7%
X_{VIM_3}	7023	0,70	33 - 50	9%	5861	0,88	36 - 50	6%
X_{VIM_4}	7017	0,60	27 - 50	8%	5860	2,20	20 - 50	6%
X_{VIM_5}	7019	0,50	19 - 50	7%	5860	1,50	32 - 50	6%
X_{VIM_6}	7023	0,56	42 - 50	4%	5846	3,33	29 - 50	4%

γ^S: S-Metrik

γ^D: Dominanzquotient bezüglich NHEMOtree (X_S)

g: Generationen

$< g_{\max}$: Häufigkeit der Konvergenz nach weniger als den maximal möglichen Generationen

Tabelle 22: Ergebnisse der Simulationsstudie mit verschiedenen Datentrenngüten α

9.2.2 NHEMOtree mit lokaler Cutoff-Optimierung

Die lokale Cutoff-Optimierung wird in der Simulationsstudie nur bei den Daten mit schlechter Trenngüte ($\alpha = 50\%$) untersucht, da bei besserer Trenngüte der Daten der Standard-NHEMOtree bereits sehr erfolgreich ist und kaum Optimierungspotential besteht. Tabelle 23 zeigt die Ergebnisse für die lokale Cutoff-Optimierung basierend auf der Fehlklassifikationsrate und der Gini-Wichtigkeit. Im Vergleich zur Cutoff-Optimierung durch den NHEMOtree liefern beide lokale Cutoff-Optimierungen leicht höhere S-Metriken und mit jedem Rekombinationsoperator auch mehr vorzeitige Abbrüche durch das OCD-Abbruchkriterium.

	Fehlklassifikationsrate				Gini-Wichtigkeit			
	γ^S	γ^D	g	$< g_{\max}$	γ^S	γ^D	g	$< g_{\max}$
X_S	5899		35 - 50	7%	5879		31 - 50	13%
X_{VIM_1}	5917	0,73	30 - 50	13%	5889	1,00	33 - 50	10%
X_{VIM_2}	5895	0,50	34 - 50	13%	5880	0,54	33 - 50	19%
X_{VIM_3}	5881	0,67	30 - 50	8%	5886	1,86	37 - 50	16%
X_{VIM_4}	5883	0,46	30 - 50	12%	5876	0,73	32 - 50	11%
X_{VIM_5}	5897	0,58	28 - 50	18%	5868	0,57	17 - 50	14%
X_{VIM_6}	5891	0,75	30 - 50	10%	5891	2,00	38 - 50	13%

γ^S: S-Metrik
γ^D: Dominanzquotient bezüglich NHEMOtree (X_S)
g: Generationen
$< g_{\max}$: Häufigkeit der Konvergenz nach weniger als den maximal möglichen Generationen

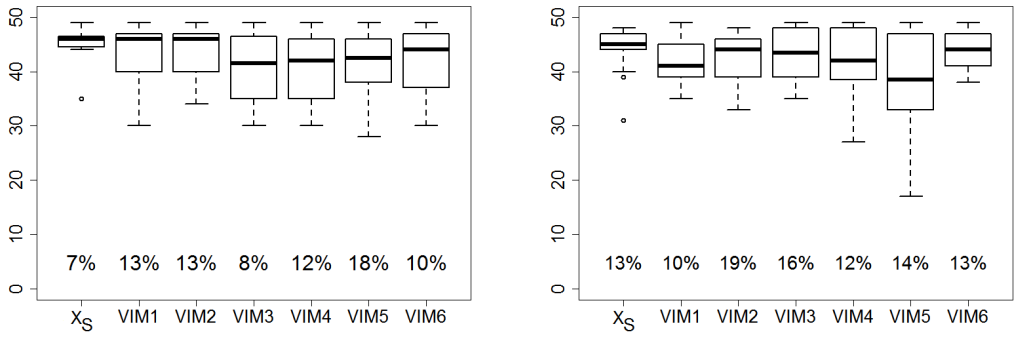
Tabelle 23: Ergebnisse der Simulationsstudie bei lokaler Cutoff-Optimierung in NHEMOtree und schlechter Trenngüte der Daten ($\alpha = 50\%$)

Der Vorteil von X_{VIM} wird auch hier deutlich. Bei der lokalen Cutoff-Optimierung basierend auf der Fehlklassifikationsrate werden mit allen X_{VIM} mehr Läufe durch das OCD-Abbruchkriterium abgebrochen als mit $NHEMOtree_{FKR}$ (X_S). Insbesondere die Verwendung von X_{VIM_5} erzielt eine erhöhte Abbruchrate durch das OCD-Abbruchkriterium. Außerdem zeigen sich bei den vorzeitig abgebrochenen Läufen bei X_{VIM_3}, X_{VIM_4} und X_{VIM_5} geringere mittlere verwendete Generationen als bei den anderen X_{VIM} und X_S (vgl. Abbildung 35(a)).

Bei $NHEMOtree_{Gini}$ ist der Unterschied in der Anzahl durch das OCD-Abbruchkriterium vorzeitig abgebrochener Läufe zwischen X_S und X_{VIM} weniger stark ausgeprägt als bei $NHEMOtree_{FKR}$. Der Vorteil von X_{VIM_5} zeigt sich aber auch hier durch die geringe mittlere Anzahl verwendeter Generationen bei vorzeitigem Abbruch durch das OCD-Abbruchkriterium (vgl. Abbildung 35(b)).

Es ist offensichtlich, dass bei schlecht separierbaren Daten die Adaption von X_S die Anzahl nötiger Generationen bis zur Konvergenz senkt. Über alle Simulationsläufe

mit den verschiedenen X_{VIM} hinweg erfolgt der vorzeitige Abbruch eines Laufs durch das OCD-Abbruchkriterium durchschnittlich häufiger bei der lokalen Cutoff-Optimierung auf Basis der Gini-Wichtigkeit als auf Basis der Fehlklassifikationsrate. Auch ist die mittlere Anzahl benötigter Generationen bei vorzeitigem Abbruch durch das OCD-Abbruchkriterium geringer bei der lokalen Cutoff-Optimierung auf Basis der Gini-Wichtigkeit (vgl. Abbildung 35), sodass ein leichter Vorteil für die lokale Cutoff-Optimierung basierend auf der Gini-Wichtigkeit besteht.



(a) Fehlklassifikationsrate

(b) Gini-Wichtigkeit

Abbildung 35: Boxplots notwendiger Generationen des NHEMOtrees mit lokaler Cutoff-Optimierung und verschiedenen Rekombinationsoperatoren bei Abbruch durch das OCD-Abbruchkriterium bei simulierten Daten mit schlechter Trenngüte ($\alpha = 50\%$)

9.3 Fazit

In der Simulationsstudie sollte die Güte des VIM-basierten Rekombinationsoperators X_{VIM} im Vergleich zum Standard-Rekombinationsoperator X_S und die mit X_{VIM} in Zusammenhang gebrachte Laufzeitreduktion im Standard-NHEMOtree untersucht werden. Es zeigt sich ein Vorteil für den NHEMOtree mit X_{VIM}, insbesondere bei Verwendung von VIMs, die die Variablenposition im Baum berücksichtigen (VIM_4 und VIM_5). Mit NHEMOtree (VIM_5) werden am häufigsten die Läufe durch das OCD-Abbruchkriterium abgebrochen. Ferner erfolgt die vorzeitige Konvergenz bereits nach weniger Generationen als bei anderen VIMs (vgl. Tabelle 22 und Abbildung 34). Die Rekombination basierend auf VIM_6 (Permutierte Fehlerfreiheit) fällt hingegen weniger gut aus. Ursächlich hierfür könnte die stärkere Fokussierung auf die Vorhersagegenauigkeit bei der Bewertung der Variablengüte sein, ohne die Kosten zu beachten. Ferner sind die Lösungen des NHEMOtrees besser als die des CART- und des NSGA-II-*Wrapper*-Ansatzes. Allerdings benötigt NHEMOtree die meisten Generationen bis zum Greifen des OCD-Abbruchkriteriums.

Im Vergleich zur Cutoff-Optimierung durch die Gauß-Mutation liefern beide lokale Cutoff-Optimierungen leicht höhere S-Metriken und mit jedem Rekombinationsoperator auch mehr vorzeitige Abbrüche durch das OCD-Abbruchkriterium. Der Vorteil von X_{VIM} gegenüber X_S ist ebenfalls bei Verwendung der lokalen Cutoff-Optimierung in NHEMOtree$_{FKR}$ und NHEMOtree$_{Gini}$ beobachtbar (vgl. Abbildung 35). Bei Verwendung von X_{VIM} wird das OCD-Abbruchkriterium häufiger erfüllt und bei den vorzeitig abgebrochenen Läufen im Mittel weniger Generationen verwendet. Dies geschieht etwas häufiger bei NHEMOtree$_{Gini}$ und insbesondere bei Verwendung von X_{VIM_5}. Die Ergebnisse mit X_{VIM_1} und X_{VIM_6} fielen hingegen weniger gut aus (vgl. Tabelle 23 und Abbildung 35).

10 Diskussion und Ausblick

Ihre Fähigkeit, ohne parametrische Voraussetzungen nicht-lineare Zusammenhänge zwischen Einfluss- und Zielvariablen zu modellieren und gut interpretierbare Klassifikationsmodelle zu erstellen, macht Entscheidungsbaumalgorithmen in vielen Wissenschaften sehr beliebt. Entscheidungsbäume werden häufig durch Greedy-Algorithmen erstellt, sodass im Allgemeinen die global beste Lösung nicht gefunden wird, denn die lokal optimalen Verbesserungen in den einzelnen Schritten eines Greedy-Algorithmus führen nicht zwangsläufig zur global optimalen Lösung. Vor allem können übliche Entscheidungsbaumalgorithmen keine mehrkriteriellen Klassifikationsprobleme lösen.

Zur Lösung mehrkriterieller Optimierungsprobleme werden häufig EMOAs verwendet. Soll wie in dieser Arbeit eine kostensensitive Klassifikation erfolgen, eignet sich die Verwendung eines mehrkriteriellen *Wrapper*-Ansatzes basierend auf einem EMOA mit Bitstring-Repräsentation und einem umhüllten Klassifikationsalgorithmus. Dabei erstellt der EMOA die Individuen bestehend aus Variablenteilmengen, auf deren Basis der Klassifikationsalgorithmus Klassifikationsmodelle erstellt. In dieser Arbeit wurde CART aufgrund seiner einfach zu interpretierenden Lösungen und aufgrund seiner Popularität zur Erstellungen der Entscheidungsbäume verwendet. Aber auch die Verwendung anderer Klassifikationsalgorithmen im mehrkriteriellen *Wrapper*-Ansatz ist möglich (vgl. Hamdani et al., 2007; Alba et al., 2007; Oliveira et al., 2003). Durch den *Wrapper*-Ansatz können die Vorhersagegüte und die Kosten der Variablenteilmengen in einem Optimierungsproblem gleichzeitig betrachtet werden. Unabhängig vom umhüllten Klassifikationsalgorithmus leiden jedoch mehrkriterielle *Wrapper*-Ansätze immer unter der Hierarchie der verschiedenen Zielkriterien. A priori wird der Vorhersagegüte als Zielkriterium eine höhere Wichtigkeit zugesprochen, da der umhüllte Klassifikationsalgorithmus nur auf Basis dieses einen Zielkriteriums eine Lösung entwickelt. Nachträglich werden die anderen Zielkriterien, wie etwa die Vorhersagekosten oder die Variablenanzahl, für diese Lösung ermittelt. Das Klassifikationsmodell wird also nicht gleichberechtigt bezüglich aller Zielkriterien erstellt. In dieser Arbeit wurde erstmals diese Hierarchie formalisiert und untersucht.

Des Weiteren besteht im mehrkriteriellen *Wrapper*-Ansatz eine eingeschränkte Lösungsvielfalt, da CART eine interne Variablenselektion durchführt, die aufgrund der Greediness zu weniger Lösungen führt. Innerhalb des mehrkriteriellen *Wrappers* wählt CART an jedem Knoten die lokal beste Variable mit entsprechendem Cutoff aus. Somit befinden sich dominante Variablen, d.h. Variablen mit einer höheren Vorhersagegüte, mit größerer Wahrscheinlichkeit in der Nähe des Wurzelknotens,

wenn sie durch den EMOA ausgewählt werden. Dadurch geht viel Flexibilität verloren und der mehrkriterielle *Wrapper*-Ansatz konvergiert vermutlich in einem lokalen Optimum. Ferner tendieren mehrkriterielle *Wrapper*-Ansätze bei kleinen Trainingsdatensätzen zur Überanpassung (vgl. Das, 2001).

Eine Alternative zum mehrkriteriellen *Wrapper*-Ansatz bieten EMOAs mit Baum-Repräsentation, wie der hier entwickelte nicht-hierarchische evolutionäre mehrkriterielle Baumlerner (NHEMOtree). NHEMOtree optimiert die verschiedenen Fitnessfunktionen gleichzeitig, sodass keine Hierarchie zwischen den Zielkriterien besteht. In dieser Arbeit wurden zwei wichtige Zielkriterien in der medizinischen Anwendung, die Fehlklassifikationsrate und die finanziellen Kosten der Variablen, betrachtet. Alternative Fitnessfunktionen zur Messung der Vorhersagegüte neben der in der Literatur am häufigsten verwendeten Fehlklassifikationsrate (vgl. Bhowan et al., 2012), wie etwa Sensitivität und Spezifität (Garcia-Nieto et al., 2009), können durch NHEMOtree ebenso optimiert werden. Des Weiteren ist NHEMOtree nicht auf nur zwei Fitnessfunktionen beschränkt. Problemlos können auch mehr als zwei Zielkriterien optimiert werden.

Nach bestem Wissen wurden in dieser Arbeit erstmalig ein mehrkriterieller *Wrapper*-Ansatz und ein EMOA mit Baum-Repräsentation systematisch miteinander verglichen. NHEMOtree erzielte sowohl im medizinischen Anwendungsbeispiel zur Diskriminierung von Lungenkrebssubtypen als auch in der Simulationsstudie stets bessere Ergebnisse in Form von höheren Dominanzquotienten und S-Metriken, mehr nicht-dominierten Lösungen und einer größeren Individuenvielfalt als der mehrkriterielle *Wrapper*-Ansatz (vgl. Tabelle 14, 16 und 22). Letzteres gilt insbesondere, da NHEMOtree nicht greedy ist und EMOAs mit Baum-Repräsentation flexibler als mehrkriterielle *Wrapper*-Ansätze mit Bitstring-Repräsentation sind (vgl. Jabeen und Baig, 2011). NHEMOtree war im medizinischen Datensatz vor allem im Bereich geringer Fehlklassifikationsraten ($<10\%$) dem *Wrapper*-Ansatz überlegen (vgl. Abbildung 30).

Flexibilität und Vielseitigkeit werden zwar in NHEMOtree wie in anderen EMOAs mit Baum-Repräsentation durch *Bloating* beeinflusst, aber in dieser Arbeit konnten Gegenmaßnahmen aufgezeigt werden, sodass NHEMOtree für die Lösung mehrkriterieller Optimierungsprobleme erfolgreich eingesetzt werden kann. Es zeigte sich, dass es bei der mehrkriteriellen Optimierung mit Zielfunktionen bezüglich der Baumgröße und der Kosten zu einer automatischen Verringerung des *Bloatings* kommt und kleinere Bäume verfügbar sind. Maßnahmen, wie die Beschränkung der maximalen Knotenanzahl je Baum, das Löschen redundanter Knoten oder die Hoist-Mutation, halfen ferner, dem *Bloating* in NHEMOtree entgegen zu wirken.

Da der Rekombinationsoperator in EMOAs mit Baum-Repräsentation als Ursache für die erfolgreiche Erstellung von *Building Block*s und somit auch für deren Effektivität angesehen wird (vgl. Abschnitt 4.3), wurde in dieser Arbeit durch verschiedene mehrkriterielle Variablenwichtigkeitsmaße (VIMs) der Standard-Rekombinationsoperator adaptiert, um dadurch bessere *Building Block*s und schlussendlich auch eine bessere Approximation der Pareto-Front zu generieren. Mit dem so entstandenen VIM-basierten Rekombinationsoperator X_{VIM} erzielte NHEMOtree verbesserte Ergebnisse (vgl. Tabelle 14) und konvergierte schneller (vgl. Tabelle 17 und 22). In X_{VIM} wird die Auswahl der Knoten für die Rekombination basierend auf den mehrkriteriellen VIMs optimiert. Je wichtiger eine Knotenvariable ist, desto höher ist die Wahrscheinlichkeit, dass diese als Wurzelknoten bei der Rekombination fungiert. Wie von Ito et al. (1998) und Xie und Zhang (2011) dargestellt, konnte auch in dieser Arbeit die Wichtigkeit der Variablenposition im Baum bestätigt werden, sodass häufig die mehrkriteriellen VIMs, die die Variablenposition im Baum berücksichtigen (VIM_4 und VIM_5), erfolgreicher waren als die VIMs ohne Tiefenabhängigkeit. Die hier vorgestellte Adaption ist entgegengesetzt zu dem Ansatz von Papagelis und Kalles (2001), in dem Knoten in fitteren Subbäumen mit einer geringeren Wahrscheinlichkeit für die Rekombination und Mutation ausgewählt wurden, denn Papagelis und Kalles (2001) konnten keine Fitness-Verbesserung der Individuen im Vergleich zu Standard-Variationsoperatoren beobachten.

Die in dieser Arbeit entwickelten mehrkriteriellen VIMs basieren mit Ausnahme von der Permutierten Fehlerfreiheit (VIM_6) auf den Variablenhäufigkeiten in den Individuen nach der Umweltselektion. Neben der Zählung einzelner Variablen könnten ebenfalls Interaktionen untersucht werden. So würde nicht mehr nur das Vorkommen einer Variable, sondern von Variablenpaaren oder -tripletts betrachtet werden, wie es beispielsweise Kooperberg und Ruczinski (2005) oder Schwender et al. (2011) machen. Diese Information ist etwa bei SNP-Assoziationsstudien von Interesse, bei denen insbesondere hohe Interaktionen von SNPs einen Unterschied zwischen Hoch- und Niedrig-Risikogruppen für die Entwicklung einer Erkrankung zeigten (vgl. Garte, 2001). Die Verwendung eines solchen VIMs wäre in NHEMOtree mit X_{VIM} leicht umsetzbar.

In NHEMOtree werden die Variablen-Cutoffs üblicherweise durch den Gauß-Mutationsoperator optimiert (Standard-NHEMOtree). Da jedoch laut Hunter (2002) Evolutionäre Algorithmen mit Baum-Repräsentation weniger gut die Cutoffs für die Knotenvariablen optimieren können als die Baumstruktur, wurde in NHEMOtree ebenfalls eine lokale Cutoff-Optimierung in Anlehnung an CART untersucht. Die Experimente im medizinischen Datensatz und in der Simulationsstudie zeigten eine Verbesserung der Ergebnisse, wenn in NHEMOtree eine lokale

Cutoff-Optimierung basierend auf der Fehlklassifikationsrate ($\text{NHEMOtree}_{\text{FKR}}$) oder der Gini-Wichtigkeit ($\text{NHEMOtree}_{\text{Gini}}$) vollzogen wurde. Der Vorteil von X_{VIM} insbesondere von X_{VIM_5} gegenüber des Standard-Rekombinationsoperators war dennoch in NHEMOtree mit lokaler Cutoff-Optimierung beobachtbar.

Allerdings führte die lokale Cutoff-Optimierung auch zu längeren Berechnungsdauern und stark vergrößerten Individuen, sodass eine Überanpassung an die Daten nicht auszuschließen ist. Trotz der Kreuzvalidierung fanden $\text{NHEMOtree}_{\text{FKR}}$ (VIM_5) und auch die meisten $\text{NHEMOtree}_{\text{Gini}}$ Individuen mit kreuzvalidierten Fehlklassifikationsraten von 0%. Durch die lokale Cutoff-Optimierung wird an jedem Knoten versucht, neue Knoten oder Blätter zu generieren, die maximal homogen sind. Ein Stopp-Mechanismus, wie etwa das Pruning in CART, ist bisher nicht in NHEMOtree integriert. Nur die maximale Knotenanzahl und eine Fitnessfunktion bezüglich der Bäumgröße können ein übermäßiges Aufteilen der Beobachtungen verhindern. Im Standard-NHEMOtree hingegen führen ungünstige Cutoffs, die einen Knoten nicht sinnvoll teilen bzw. nicht in dem Knoten optimal teilen, zu redundanten Knoten. Diese nichtterminalen Knoten mit nur einem Nachkommen werden dann durch den Algorithmus entfernt. Auf diese Weise werden die Bäume häufiger verkleinert und der Überanpassung wird so indirekt entgegengewirkt.

Weiterhin führt die lokale Cutoff-Optimierung zu einer Beschränkung der Variabilität des NHEMOtrees, da die lokale Cutoff-Wahl stets von den oberen Knoten abhängt. Außerdem widerspricht die integrierte lokale Optimierung der Idee der Metaheuristik. Metaheuristiken leiten die Optimierung mit dem Ziel, den Suchraum erfolgreich zu erkunden, um optimale Lösungen zu finden. Sie optimieren ein Problem durch iteratives Ausprobieren, um eine Kandidatenlösung zu verbessern (vgl. Mucherino und Seref, 2009; Blum und Roli, 2003). Im Gegensatz dazu ist die vollständige Iteration über alle Möglichkeiten, wie bei der lokalen Cutoff-Optimierung, unelegant und sehr zeitaufwendig, sodass NHEMOtree die Gauß-Mutation mit adaptiver Schrittweitenanpassung standardmäßig zur Cutoff-Optimierung verwendet.

Neben dem NSGA-II können auch andere EMOAs, wie der *Steady-State* NSGA-II oder der SMS-EMOA, zur Umweltselektion in NHEMOtree eingesetzt werden. Bisher wurden konträre Beobachtungen bei dem Vergleich der verschiedenen EMOAs gemacht. Etwa Brockhoff und Zitzler (2007), Emmerich et al. (2005) und Nebro und Durillo (2009) beobachteten bessere Ergebnisse für *Steady-State*-Ansätze, was Kinnear Jr. (1993) der Populationsdiversität und der direkten Verfügbarkeit von überlegenen Individuen zugeschriebt. Eggermont et al. (1999) erzielten hingegen bessere Resultate für den generationsbasierten Ansatz und Bhanu und Lin (2002) fanden keine signifikanten Unterschiede zwischen beiden Ansätzen.

In dieser Arbeit waren im mehrkriteriellen *Wrapper*-Ansatz ebenfalls kaum Unterschiede zwischen den Lösungen des NSGA-II, *Steady-State* NSGA-II und SMS-EMOA mit CART als umhüllten Klassifikationsalgorithmus beobachtbar. Jedoch war die Laufzeit des NSGA-II am kürzesten. In NHEMOtree zeigte die Umweltselektion nach dem Vorbild des SMS-EMOA gegenüber des NSGA-II im medizinischen Datensatz leichte Vorteile mit schwach höheren S-Metriken. Die Lösungsgüten von NHEMOtree mit einer Umweltselektion nach Vorbild des NSGA-II und des *Steady-State* NSGA-II waren hingegen ähnlich. Die Laufzeiten der *Steady-State*-Ansätze in NHEMOtree waren allerdings um ein Vielfaches langsamer als bei Verwendung des NSGA-II-Ansatzes zur Umweltselektion. Aufgrund der geringeren Laufzeit des NSGA-II-Ansatzes (vgl. Bader und Zitzler, 2011), dessen hoher Popularität (vgl. Coello Coello, 2009) und den ähnlichen Ergebnissen zwischen den verschiedenen EMOAs wurde in dieser Arbeit standardmäßig die nicht-dominierte Sortierung und die *Crowding*-Distanz analog zum NSGA-II zur Umweltselektion in NHEMOtree verwendet. Der finale Vorschlag des Standard-NHEMOtrees mit den entsprechenden Operatoren und mehrkriteriellem VIM entspricht dem Pseudocode aus Algorithmus 9 in Kapitel 7.

Der Umgang mit redundanten Knoten in EMOAs mit Baum-Repräsentation ist auf verschiedene Weisen möglich. In dieser Arbeit wurden nach dem Vorbild von Haruyama und Zhao (2002) redundante Knoten während eines Laufs entfernt und die Bäume somit stets verkleinert. Laut Cherkauer und Shavlik (1996) sollten jedoch nach Möglichkeit die Fitnessfunktionen von Bäumen mit leeren Blättern bestraft werden. Dies könnte problemlos in NHEMOtree integriert werden. Allerdings würden sich die Ergebnisse beider Umgangsweisen wahrscheinlich nicht maßgeblich unterscheiden. In beiden Fällen werden Individuen mit redundanten Knoten früher oder später aus der Population entfernt. In dem Beispiel von Cherkauer und Shavlik (1996) geschieht dies über den Umweg der verschlechterten Fitnessfunktionswerte, sodass solche Individuen im Vergleich zu Individuen ohne redundante Knoten mit höherer Wahrscheinlichkeit dominiert und dann aus der Population entfernt würden.

Es existieren auch einige Limitationen bezüglich des NHEMOtrees. Schwachstellen des NHEMOtrees sind wie bei allen Evolutionären Algorithmen die lange Rechenzeit sowie die notwendige Wahl der Parametereinstellungen. Allerdings ist die erhöhte Berechnungsdauer sowohl beim mehrkriteriellen *Wrapper*-Ansatz als auch bei NHEMOtree gering im Vergleich zur besseren Verständlichkeit der Modelle und der mehrkriteriellen Optimierung (vgl. Pappa et al., 2002). Im Allgemeinen nimmt die Statistik in epidemiologischen und klinischen Studien an dem Prozess der Wissensentdeckung im Vergleich zur Probandenrekrutierung und Bestimmung der biologischen Parameter wenig Zeit in Anspruch. Somit ist es in vielen An-

wendungen akzeptabel, wenn der Algorithmus auch über mehrere Tage läuft. Bei EMOAs besteht die Möglichkeit, die Rechenzeiten durch parallele und verteilte Implementierungen zu reduzieren (vgl. Espejo et al., 2010). Dies könnte auch in NHEMOtree für eine Verringerung der Rechenzeit sorgen.

Eine gute Wahl der Kontrollparameter trägt wesentlich zu einer guten Leistung der EMOAs bei und hängt vor allem von den zugrunde liegenden Daten ab. Im mehrkriteriellen *Wrapper*-Ansatz und in NHEMOtree wurde versucht, mittels Latin-Hypercube-Designs eine gute Kombination zwischen Populationsgröße, maximaler Generationenanzahl, Rekombinations- und Mutationsrate zu erzielen. Dennoch könnte eine dynamische Parameterkontrolle, bei der sich die Parameter über die Generationen hinweg verändern, oder eine selbst-adaptive Parameterkontrolle, bei der sich die anzupassenden Parameter ebenfalls der Rekombination und Mutation unterziehen, sinnvoll und erfolgreich sein. Trotz der Schwierigkeit bei der praktischen Umsetzung flexibler und selbst-adaptiver Parameter und der Skepsis gegenüber der Verbesserung durch dynamische Parameter (vgl. De Jong, 2007), könnte dies ein Schritt sein, um Latin-Hypercube-Designs zu umgehen und verbesserte Ergebnisse zu erzielen.

Die Adaption der Parameter könnte ferner im VIM-basierten Rekombinationsoperator X_{VIM} zum Einsatz kommen. So könnten die Gewichte der Baumebenen in VIM_4 beziehungsweise VIM_5 an die Laufzeit oder die Güte der erzielten Nachkommen gekoppelt werden. Denkbar wäre etwa die Erhöhung der Wichtigkeit von Variablen nahe des Wurzelknotens im Laufe der Evolution, um weiteres *Bloating* einzuschränken. Denn durch die wichtigeren Variablen in höheren Baumebenen sinkt die Wahrscheinlichkeit, dass tiefe Rekombinationspunkte gewählt und dadurch größere Nachkommen erstellt werden.

Neben den verwendeten Mutationsoperatoren zur Adaption der Cutoff-Werte in NHEMOtree existieren weitere vielversprechende Methoden. Die Integration der vor allem in den Evolutionsstrategien gebräuchlichen Kovarianzmatrix-Adaption (engl. *covariance matrix adaptation*, CMA) wäre eine mögliche Erweiterung von NHEMOtree. Die neuen Cutoff-Werte würden mit einer Zufallszahl aus einer Gauß-Verteilung mit adaptierter Kovarianzmatrix, die die Abhängigkeiten der Variablen beschreibt, gezogen. Die Adaptation der Kovarianzmatrix beruht auf der Idee, die Wahrscheinlichkeit von vormals erfolgreichen Schritten zu erhöhen. Dazu wird die Kovarianzmatrix so verändert, dass sich die Wahrscheinlichkeit des erfolgreichen Selektionsschritts aus der letzten Generation für die aktuelle Generation vergrößert, sodass die Konvergenz zum Optimum gesteigert wird (vgl. Hansen et al., 1995). Darüberhinaus existieren weitere CMA-basierte Mutationsmethoden, beispielsweise mit verringerter Zeit- und Speicherkomplexität (vgl. Ros und Hansen, 2008).

Des Weiteren ist die Validierung des NHEMOtrees in unbalancierten Datensätzen mit verschiedenen Klassengrößen wünschenswert. Dies bedarf einer Anpassung der Fitnessfunktionen, da die Fehlklassifikationsrate bei unbalancierten Klassengrößen nur dann erfolgreich ist, wenn die größte Klasse gut vorhergesagt wird. Die Vorhersage der kleineren Klassen ist dann kaum ausschlaggebend für die Vorhersagegüte. Bhowan et al. (2012) etwa entwickelten Fitnessfunktionen für binäre Klassifikationsprobleme, aber auch die Optimierung mehrerer Fitnessfunktionen bezüglich der Vorhersagegüte erleichtert den Umgang mit unbalancierten Daten. So können etwa gleichzeitig die Sensitivität und Spezifität oder die Fehlklassifikationsrate jeder Klasse gleichzeitig optimiert werden (vgl. Bhowan et al., 2011). Dies ist ohne Probleme mit NHEMOtree durchführbar.

Um einen gerechten Vergleich zwischen NHEMOtree und dem mehrkriteriellen *Wrapper*-Ansatz mit CART zu ziehen, wurde NHEMOtree analog zu CART als univariater Entscheidungsbaumalgorithmus mit genau einer Variable je Splittingkriterium konzipiert. Die Flexibilität von NHEMOtree könnte jedoch einfach erhöht werden, wenn die Splittingkriterien in den internen Knoten frei modellierbar wären, ähnlich zu Bot und Langdon (2000). Würden mit NHEMOtree etwa multivariat nicht-lineare Entscheidungsbäume erstellt, würde der Suchraum durch nicht-lineare Entscheidungsfunktionen partitioniert (vgl. Ittner und Schlosser, 1996) und NHEMOtree könnte in diesem stark vergrößerten Suchraum operieren.

Die Ergebnisse des medizinischen Datensatz und der Simulationsstudie zeigen, dass durch NHEMOtree eine nicht-hierarchische mehrkriterielle Optimierung möglich ist. Diese ist erfolgreicher als der mehrkriterielle *Wrapper*-Ansatz, der die Vorhersagegüte als Zielkriterium bei der Lösung der mehrkriteriellen Optimierungsprobleme bevorzugt. Der Erfolg der nicht-hierarchischen mehrkriteriellen Optimierung von Vorhersagegüte und -kosten spricht für eine sinnvolle Herangehensweise und lässt vermuten, dass eine Erweiterung dieser Idee für andere Klassifikationsalgorithmen ebenfalls aussichtsreich ist. Mit Hilfe von Syntaxbäumen (vgl. Hopcroft et al., 1979) könnte zum Beispiel die Variablenselektion in logistischen Regressionsmodellen bei verschieden teuren Einflussvariablen nicht-hierarchisch erfolgen. Offensichtlich beherbergt der innovative Ansatz der nicht-hierarchischen Optimierung ein hohes Forschungspotential, das es in Zukunft zu erkunden gilt.

Anhang

A Versuchsplanung in Computer-Experimenten

Die Auswahl geeigneter Parameter ist in EAs wie bei anderen Methoden von großer Bedeutung für das Auffinden guter Lösungen (vgl. Parsons und Johnson, 1997). Vor allem in mehrkriteriellen Problemen ist dies keine leichte Aufgabe. Da viele reale Probleme rechenintensiv sind, können nur wenige Optimierungsläufe durchgeführt werden, sodass diese mit guten Parametereinstellungen erfolgen sollten. Das Ziel ist somit, gute Parametereinstellungen mit wenigen Läufen zu erreichen.

Mit Hilfe der statistischen Versuchsplanung wird versucht, mit minimalem Aufwand, d.h. wenigen Experimenten, möglichst viel Information über die Beziehung zwischen Einflussvariablen und Zielgröße zu gewinnen. Anders als in der klassischen Versuchsplanung von physischen Experimenten müssen bei Computer-Experimenten Prinzipien wie Blockbildung und Randomisierung nicht beachtet werden (vgl. Santner et al., 2003). Da nicht bekannt ist, in welcher Datensituation welche Parametereinstellung die beste ist, sollten die Versuchspunkte den Raum der möglichen Datensituationen gut repräsentieren.

Eine sehr einfache Strategie ist die Auswahl aller Parameterwerte auf einem gleichmäßigen Gitter über der Experimentationsregion. In diesem Fall liegt ein *One-Factor-At-a-Time*-Versuchsplan zugrunde, bei dem immer nur ein Faktor verändert wird, während die anderen konstant bleiben. Eine bessere Alternative stellen raumfüllende Versuchspläne dar, die speziell für die Analyse von Computerexperimenten entwickelt wurden (vgl. Santner et al., 2003). Hierzu zählen etwa die einfache Zufallsauswahl, die stratifizierte Zufallsauswahl oder das Latin-Hypercube-Sampling.

Latin-Hypercube-Designs (LHDs) basieren auf Latin-Hypercube-Stichproben, werden laut Santner et al. (2003) im Rahmen von Computer-Experimenten sehr häufig verwendet und gehören zu den populärsten raumfüllenden Versuchsplänen. Gründe hierfür sind ihre einfache Erzeugung und ihre gute Projektionseigenschaft. Diese Eigenschaft garantiert, nach einer beliebigen Projektion der Versuchspunkte des LHDs auf eine der Koordinatenachsen eine äquidistante Verteilung der Punkte. Zur Generierung eines LHDs mit n Versuchspunkten für d Faktoren wird der Wertebereich für jeden der d Faktoren in n (gleich große) Intervalle geteilt. Die Menge der kartesischen Produkte dieser Intervalle unterteilt den Versuchsraum in $n \times d$ d-dimensionale Hypercubes.

Die Bestimmung der n Versuchspunkte des LHDs kann als Urnenexperiment aufgefasst werden. Dabei wird aus d Urnen mit den Zahlen $1, \ldots, n$ n-mal ohne Zu-

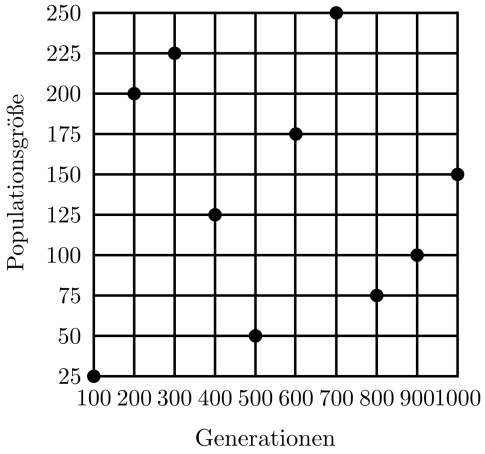


Abbildung 36: Maximin-Latin-Hypercube-Design bzgl. Generationenanzahl und Populationsgröße

rücklegen gezogen. Die i-te gezogene Zahlenkombination definiert die i-te Zelle des Versuchsplans. Die Versuchspunkte können durch die Mittelpunkte der ausgewählten Zellen oder auch zufällig innerhalb einer Zelle ausgewählt werden. Eine Alternative, die hier angewendet wird, ist, die gezogenenen Zahlenkombinationen nicht als Zellen eines Versuchsplans sondern als dessen Gitterpunkte aufzufassen. Auf diese Weise entstehen Versuchspunkte (Gitterpunkte) mit zuvor festgesetzten Parameterwerten. Dies hat den Vorteil, dass Versuchspunkte mit sinnvollen Werten entstehen. Beispielsweise wäre eine Populationsgröße $\mu = 197$ sonderbar, der Anwender würde eher $\mu = 200$ Individuen pro Population betrachten. Für die Auswahl der Versuchspunkte auf einem LHD-Gitter vergleiche Abbildung 36.

Allerdings füllt nicht jedes LHD den Raum der Versuchspunkte gleichmäßig aus, sodass aus mehreren LHDs basierend auf einem Kriterium ein LHD ausgewählt werden muss. In dieser Arbeit wird derjenige Versuchsplan mit größtem minimalen Abstand verwendet. Ein solcher „Maximin-Versuchsplan“ wird in Definition 7 vorgestellt (vgl. Santner et al., 2003). Andere Optimalitätskriterien als das Maximin-Abstandsmaß zur Auswahl von Versuchsplänen sind etwa durch Quasi-Monte-Carlo-Methoden motiviert (vgl. Fang et al., 2000).

Definition 7 (Maximin-Versuchsplan)
Ein Versuchsplan $\mathcal{D} \in \mathbb{R}^{n\times d}$ mit Versuchspunkten $x_i \in \mathbb{R}^d$, $i = 1,\ldots,n$, ist ein Maximin-Versuchsplan ($\mathcal{D}_M$), wenn der minimale Abstand δ zwischen den Versuchspunkten maximiert wird, d.h. wenn gilt

$$\min_{x_i,x_j\in\mathcal{D}_M} \delta(x_i,x_j) = \max_{\mathcal{D}\in\mathbb{R}^{n\times d}} \min_{x_i,x_j\in\mathcal{D}} \delta(x_i,x_j).$$

Dabei kann etwa das Abstandsmaß $\delta(x_i,x_j) = \sqrt{\sum_{k=1}^{d} |x_{i_k} - x_{j_k}|^2}$ gewählt werden.

B Medizinischer Datensatz

Der medizinische Datensatz umfasst die Informationen der immunhistochemischen Färbung von 22 Proteinen in der Zellmembran, im Zytoplasma und im Zellkern von 143 Lungengewebsproben. Die Auswahl der untersuchten Proteine erfolgte auf der Grundlage vorhandenem Wissens in der Literatur und wird in Abschnitt B.1 näher erläutert. Die Färbung je Protein und Zellbestandteil (vgl. Abschnitt B.2) wird durch je einen Score quantitativ erfasst (vgl. Abschnitt B.3), sodass insgesamt pro Gewebe 66 Farbintensitätsscores mit Ausprägungen zwischen 0 und 300 vorliegen, um die histologischen Lungenkrebssubtypen zu klassifizieren. Die finanziellen Kosten für die verwendeten Antikörper zur Proteinfärbung werden in Abschnitt B.4 beschrieben.

B.1 Protein-Auswahl

Für das Auffinden von Protein-Signaturen zur Charakterisierung der histologischen Lungenkrebssubtypen werden bei der Proteinauswahl das in der Literatur vorhandene Wissen über potentielle Marker, sowie über die Krebs-Stammzell-Hypothese, generelle Krebsmodelle und lungenspezifische Modelle genutzt.

Krebs-Stammzell-Hypothese

Der Krebs-Stammzell-Hypothese zufolge wird die Tumorentwicklung und das Tumorwachstum durch Tumorstammzellen vorangetrieben (vgl. Polyak und Hahn, 2006). Einige der Hanahan-Weinberg-Eigenschaften (vgl. Hanahan und Weinberg, 2000, 2011) sind bereits bei Stammzellen, deren Tochterzellen bzw. Progenitorzellen vorhanden, sodass möglicherweise durch wenige Mutationen aus diesen Zellen Tumore entstehen können. Von Bedeutung für die Stammzell-Hypothese ist ferner, dass Krebs überwiegend in regenerativen Geweben nach Gewebeschädigung auftritt. Abgestorbenes Gewebe wird aus undifferenzierten Vorläuferzellen bzw. Stammzellen regeneriert. Dabei werden Gene aktiviert, die in der Organentwicklung wichtige ontogenetische Prozesse steuern. Bei der Krebsentstehung erfolgt die Differenzierung nicht mehr plangemäß, jedoch können die Krebszellen dennoch gewisse Stammzell-Signaturen aufweisen. Mögliche Stammzell-Signaturen werden bereits in einer Reihe von Tumoren diskutiert (vgl. Kim et al., 2005; Brabletz et al., 2005; Glinsky et al., 2005).

Generelle Krebsmodelle

Nach dem klassischen Modell von Hanahan und Weinberg (2000) für die Schlüsselprozesse der Krebsentstehung müssen Gewebe mindestens sechs grundlegende Eigenschaften besitzen, um maligne zu entarten. Diese sind die Unabhängigkeit von externen Wachstumssignalen, die Unempfindlichkeit gegenüber wachstumshemmen-

den Signalen, das Ausschalten der Apoptose, die uneingeschränkte Fähigkeit der Genomreplikation, die eigenständige Bildung von Blutgefäßen, sowie das Eindringen in Gewebe und die Metastasenbildung.

Lungenspezifische Modelle
Für die Proteinauswahl sind ferner gewebespezifische Kenntnisse über Stammzellen der Lunge, die Lungenentwicklung, die Sauerstoffversorgung und die Regenerationsmechanismen der Lunge bei Gewebeschädigung wichtig. In der Lunge ist der regenerative Durchsatz relativ gering. Allerdings sind die Atemwege auskleidenden Epithelien der Außenwelt ausgesetzt, sodass die Lunge möglicherweise Stammzell-Nischen vorhält oder Transdifferenzierungen von Bedeutung sind.

B.2 Immunhistochemische Färbung

Neben den biologischen Kriterien (vgl. Abschnitt B.1) ist für die Proteinauswahl auch entscheidend, dass die entsprechenden Antikörper zur Verfügung stehen und zur immunhistochemischen Färbung des vorhandenen Probenmaterials geeignet sind. Bei der immunhistochemischen Färbung werden Antigene (Proteine) mittels Antikörper, die gegen das Protein gerichtet und mit einem Farbstoff beladen sind, sichtbar gemacht und so ein Proteinnachweis auf der Zellmembran, im Zytoplasma und im Zellkern erbracht. Auf diese Weise ist erkennbar, ob ein Protein im Gewebe enthalten und in welchem Zellbestandteil es vorwiegend lokalisiert ist. Aufgrund des unterschiedlichen Vorkommens der Proteine im Gewebe sind nach der Einfärbung die entsprechenden Proteinstrukturen erkennbar. Für die immunhistochemische Färbung existieren verschiedene Möglichkeiten. Hier erfolgt die Einfärbung der Gewebeschnitte mit der Labeled-Streptavidin-Biotin-Methode (vgl. Elias et al., 1989).

Die Einfärbung, Sichtung und Beurteilung der Gewebeschnitte erfolgt lichtmikroskopisch durch Pathologen des Instituts für Pathologie der Ruhr-Universität Bochum am Berufsgenossenschaftlichen Universitätsklinikum Bergmannsheil.

B.3 Score-Bildung

Für die statistische Analyse werden immunhistochemische Ergebnisse oftmals in Scores zusammengefasst (vgl. McCarty Jr et al. (1985) oder Remmele et al. (1986)), die die Anwendung von Klassifikationsverfahren vereinfachen. Im vorliegenden Fall wird der in der Medizin häufig verwendete histochemische Score (H-Score) von McCarty Jr et al. (1985) verwendet. Der H-Score wurde für die Analyse des Östrogenrezeptors bei der Betrachtung von Brustkrebsfällen entwickelt und verbindet sowohl die Intensität als auch die Verteilung der Färbung. Im Gegensatz zum ori-

ginalen H-Score mit fünf Intensitätskategorien werden hier vier Kategorien *keine Färbung (0)*, *schwache Färbung (1)*, *mittlere Färbung (2)* und *starke Färbung (3)* mit den zugehörigen Anteilen gefärbter Zellen der jeweiligen Kategorie betrachtet. Der Scorewert ergibt sich als Summe der Anteile multipliziert mit der gewichteten Färbeintensität und liegt zwischen 0 und 300:

$$\begin{aligned}\text{H-Score} = \; & 0 \times \text{Zellanteil ohne Färbung} \\ & + 1 \times \text{Zellanteil mit schwacher Färbung} \\ & + 2 \times \text{Zellanteil mit mittlerer Färbung} \\ & + 3 \times \text{Zellanteil mit starker Färbung} \qquad \in [0{,}300].\end{aligned}$$

Eine Alternative zum H-Score wäre etwa der Immunoreaktive Score von Remmele et al. (1986), bei dem neben der Farbintensität auch der Anteil der gefärbten Zellen pro Intensität kategorisiert wird.

B.4 Kostenparameter

Die Kosten der Variablen werden zum einen als die Materialkosten der Antikörper, die für die Klassifikation verwendet wurden, zum anderen als die Anzahl der verwendeten Antikörper betrachtet. Die Antikörperanzahl kann stellvertretend für den Zeitaufwand der Auswertung betrachtet werden. Ferner wird eine 3-kriterielle Optimierung mit der Klassifikationsgüte und den beiden Kostenfunktionen betrachtet.

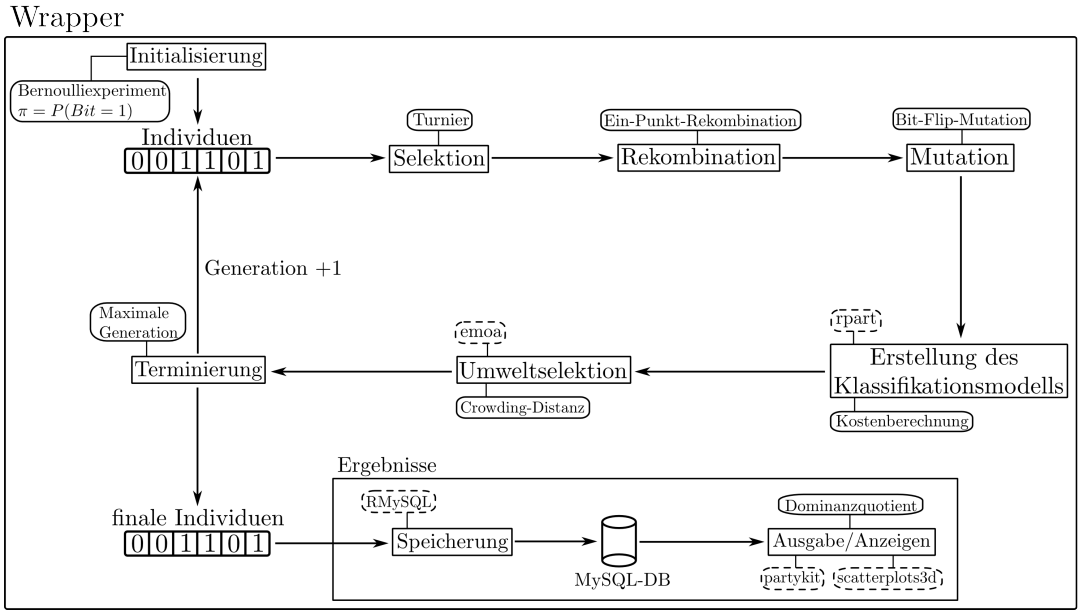
Antikörper	Kosten	Antikörper	Kosten	Antikörper	Kosten
X_1	1,027	X_9	0,008	X_{17}	6,181
X_2	1,671	X_{10}	5,536	X_{18}	5,150
X_3	1,761	X_{11}	3,602	X_{19}	5,227
X_4	4,030	X_{12}	3,911	X_{20}	37,757
X_5	0,887	X_{13}	4,132	X_{21}	5,820
X_6	0,891	X_{14}	0,412	X_{22}	4,815
X_7	3,097	X_{15}	0,255		
X_8	3,054	X_{16}	0,776		
Σ					100

Tabelle A1: Antikörperkosten je Färbung

Je gefärbtem Schnitt werden $200\mu l$ pro verwendeten Antikörper benutzt. Die Antikörperkosten pro Färbung eines Schnitts sind in Tabelle A1 aufgeführt. Je nachdem wie viele verschiedene und welche Antikörper zur Anfärbung des Schnittes verwendet werden, variieren die Kosten zwischen 0,008 bei Anfärbung mittels X_9 und 100 bei

Anfärbung mittels aller Antikörper. Das zeigt, dass eine kostensensitivere Variablenselektion sinnvoll ist. Vor allem vor dem Hintergrund immer leerer werdender Krankenkassenkassen, steigender Gesundheitsausgaben und der Altersverteilung, ist es notwendig kostengünstige Diagnosemethoden und Therapien aufzuzeigen. Mit den EMOAs und den daraus resultierenden approximierten Pareto-Fronten wird es möglich sein, die für den aktuellen Geldbeutel der Volkswirtschaft passende Klassifikationsgüte und somit auch die zu den möglichen Konditionen optimale Antikörperauswahl zu treffen.

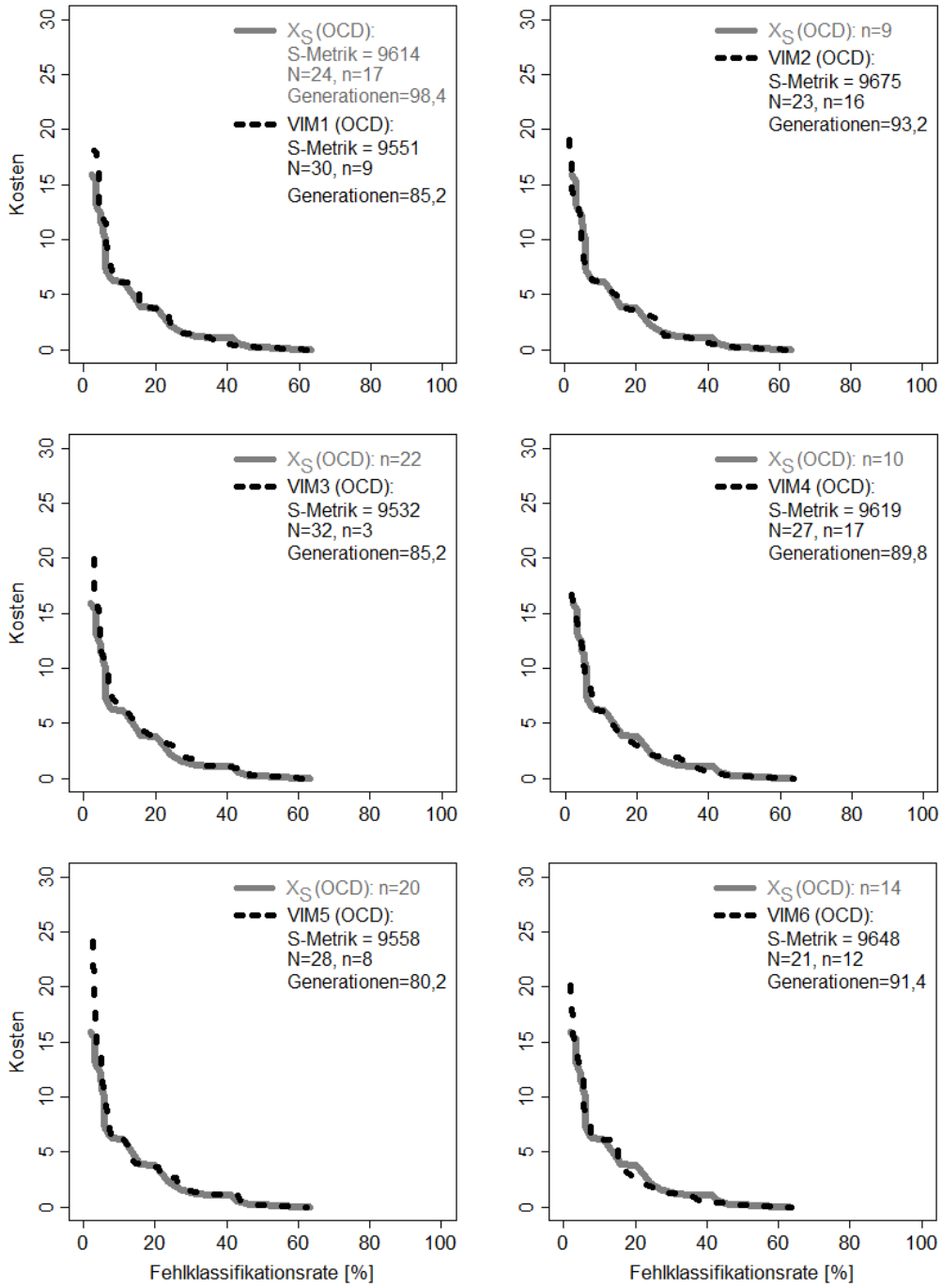
C Abbildungen



Legende:

Rechtecke:	Einzelne Programmierungsschritte
Abgerundete Rechtecke:	Selbstgeschriebene Funktionen
Gestrichelte Umrandungen:	Verwendete R-Pakete

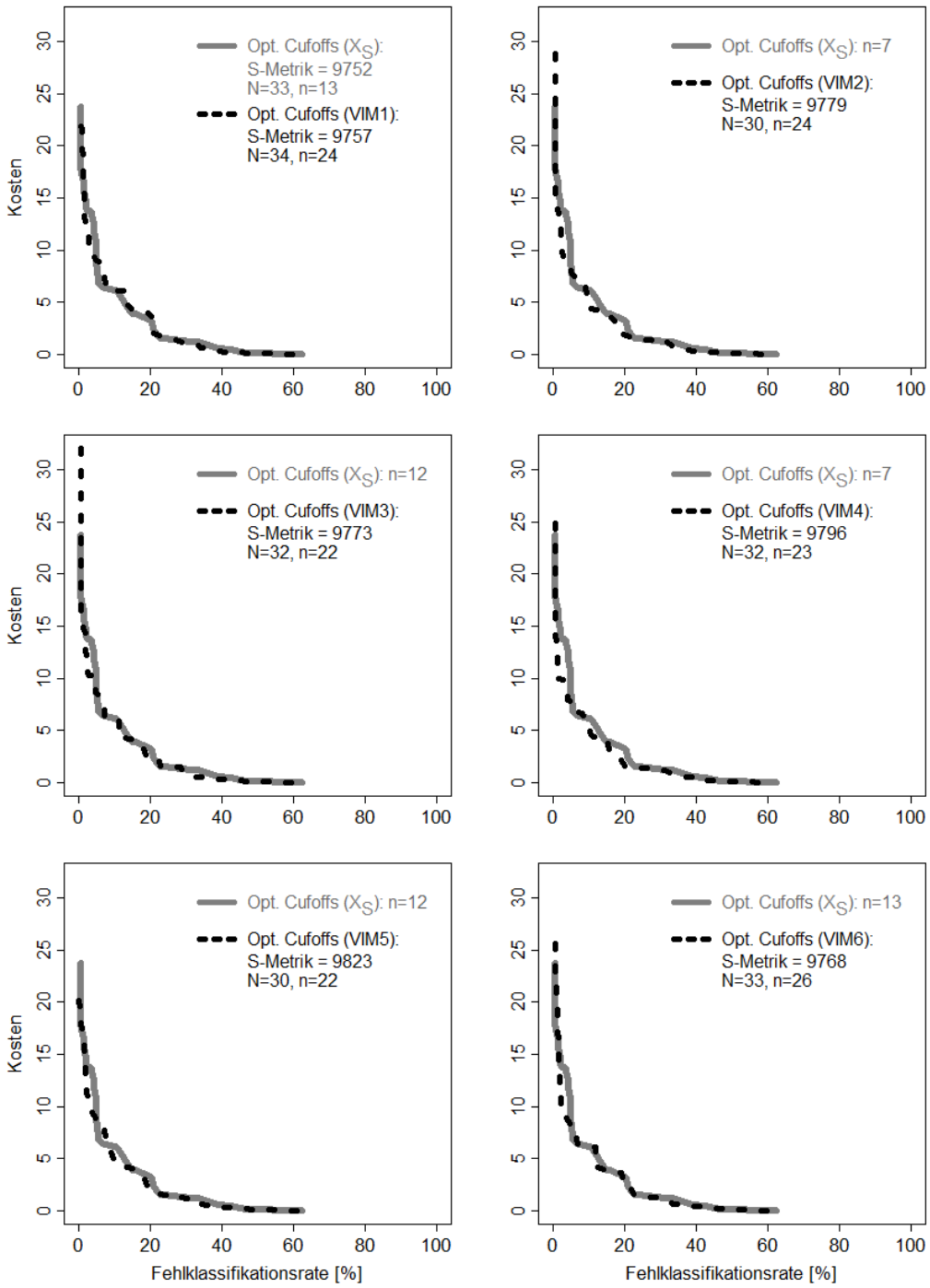
Abbildung 37: Programmierungsablauf für den *Wrapper*-Ansatz



N: Anzahl nicht-dominierter Individuen in der finalen Population

n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

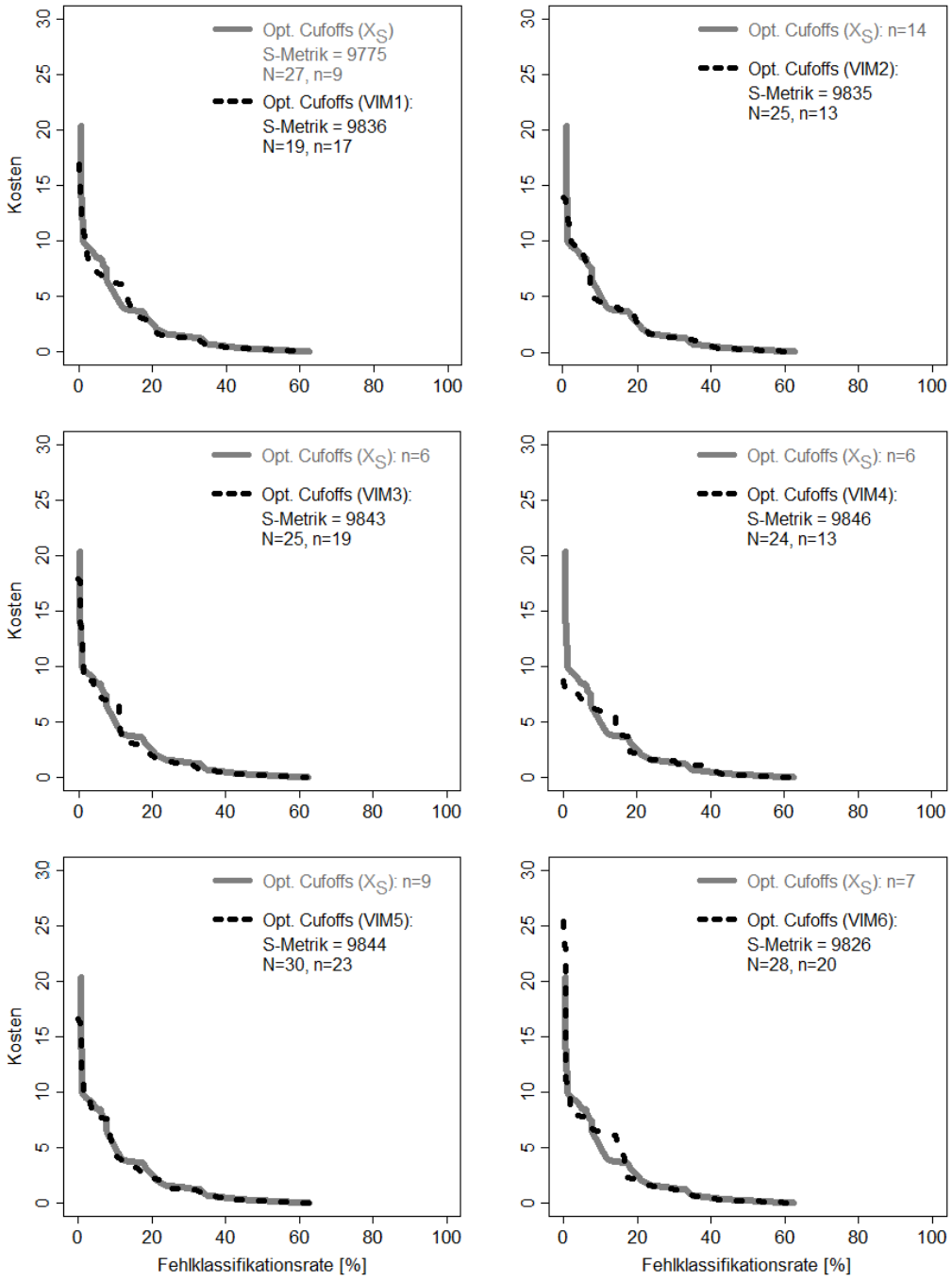
Abbildung 38: Vergleich der Pareto-Front-Approximationen der NHEMOtrees mit OCD-Abbruchkriterium und verschiedenen Rekombinationsoperatoren



N: Anzahl nicht-dominierter Individuen in der finalen Population

n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 39: Vergleich der Pareto-Front-Approximationen des NHEMOtrees mit lokaler Cutoff-Optimierung basierend auf der Fehlklassifikationsrate und verschiedenen Rekombinationsoperatoren



N: Anzahl nicht-dominierter Individuen in der finalen Population
n: Anzahl nicht-dominierter Individuen auf der gemeinsamen Pareto-Front

Abbildung 40: Vergleich der Pareto-Front-Approximationen des NHEMOtrees mit lokaler Cutoff-Optimierung basierend auf der Gini-Wichtigkeit und verschiedenen Rekombinationsoperatoren

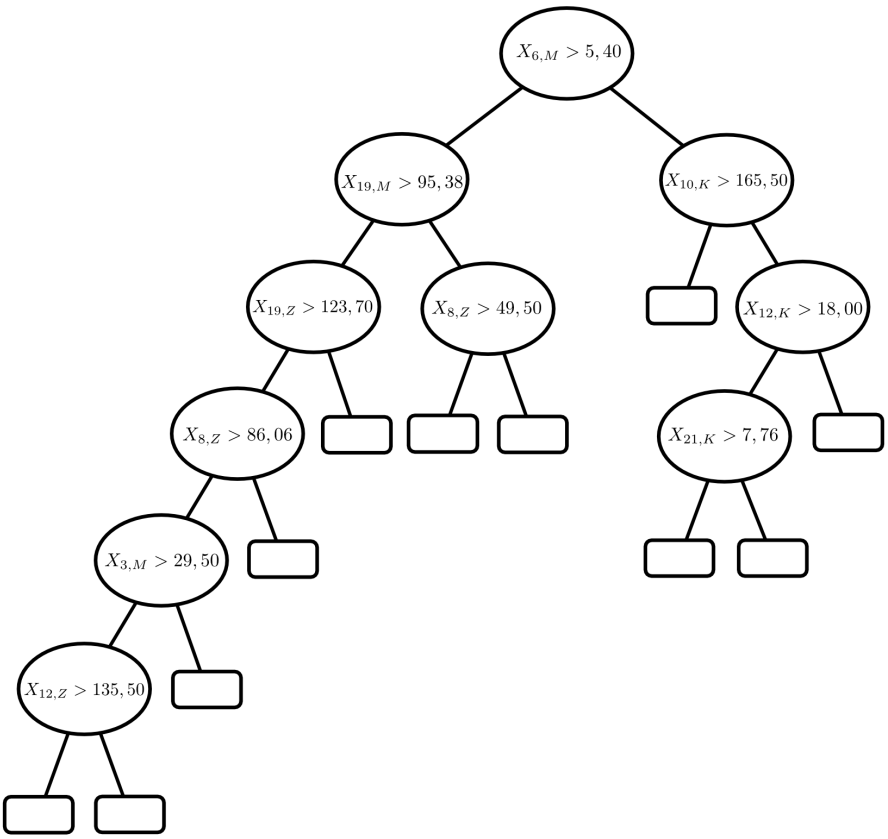


Abbildung 41: NHEMOtree-Individuum mit X_S und minimaler Fehlklassifikationsrate

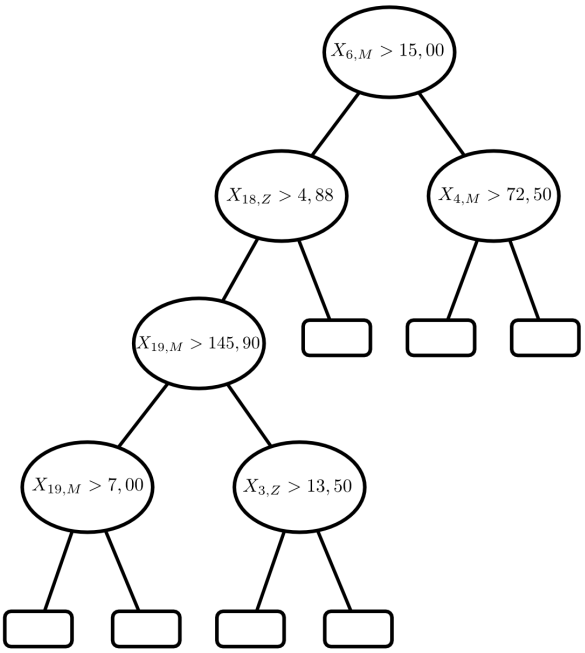


Abbildung 42: NHEMOtree-Individuum mit X_{VIM_1} und minimaler Fehlklassifikationsrate

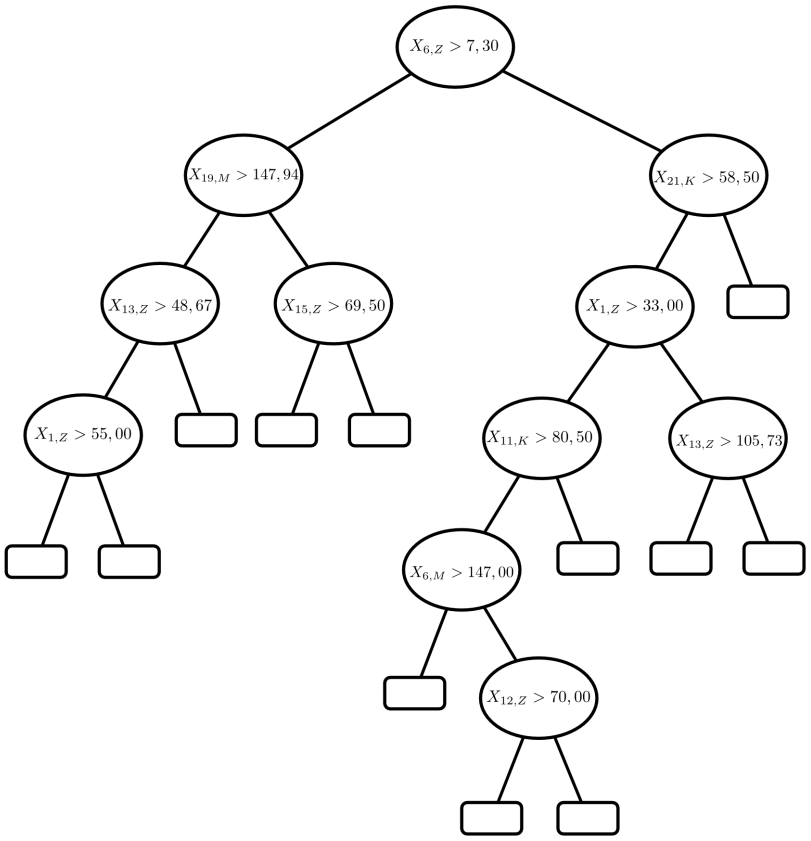


Abbildung 43: NHEMOtree-Individuum mit X_{VIM_2} und minimaler Fehlklassifikationsrate

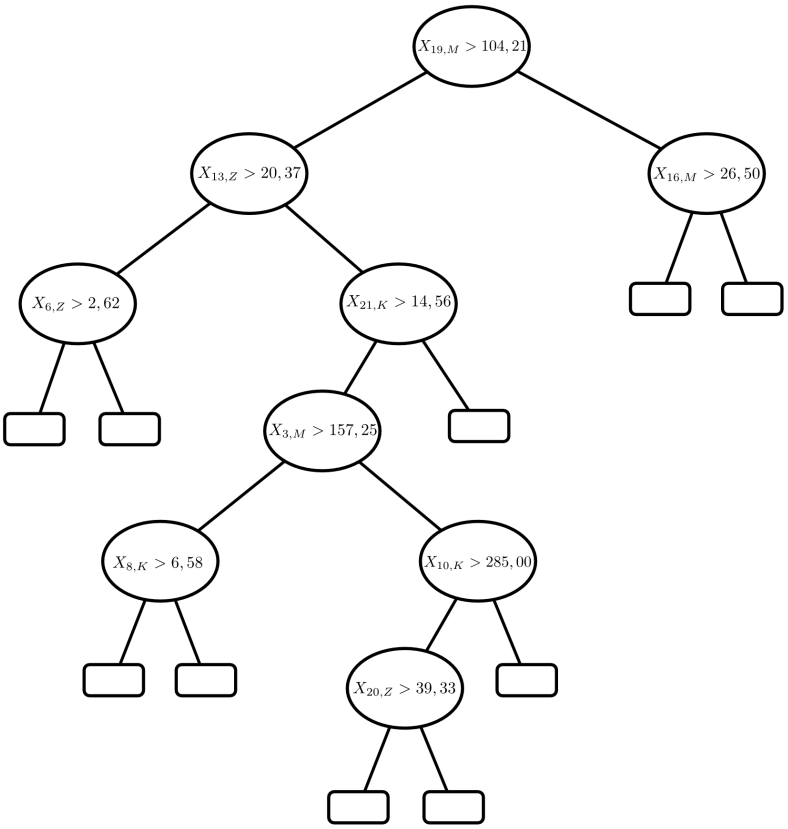


Abbildung 44: NHEMOtree-Individuum mit X_{VIM_3} und minimaler Fehlklassifikationsrate

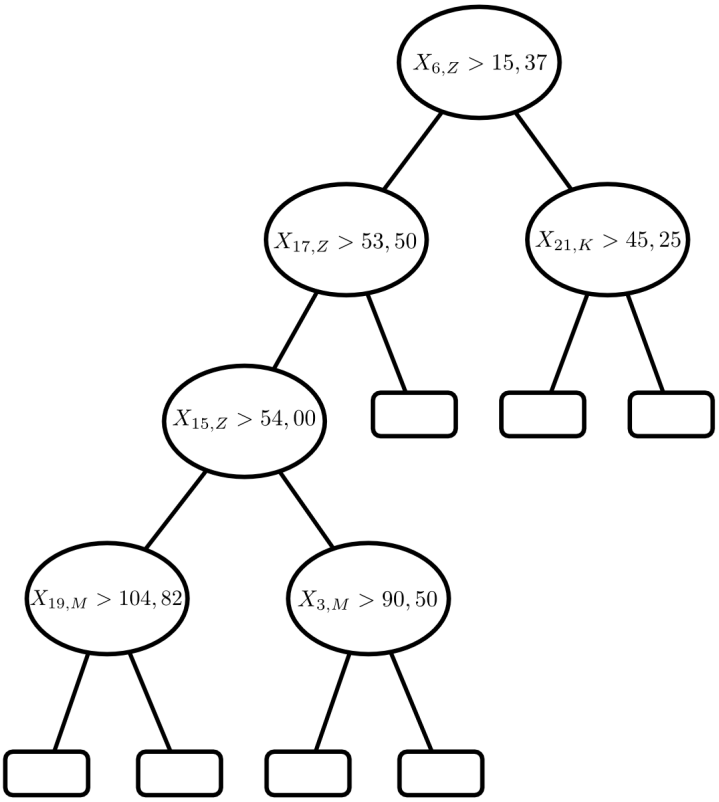


Abbildung 45: NHEMOtree-Individuum mit X_{VIM_4} und minimaler Fehlklassifikationsrate

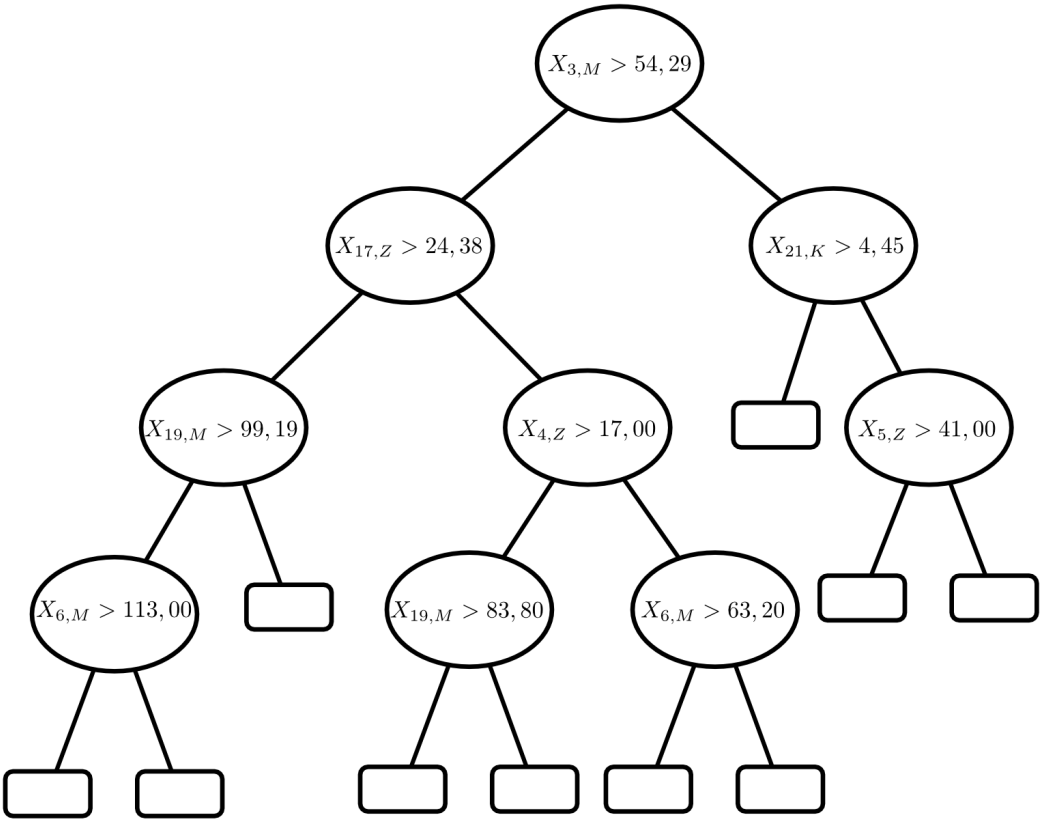


Abbildung 46: NHEMOtree-Individuum mit X_{VIM_5} und minimaler Fehlklassifikationsrate

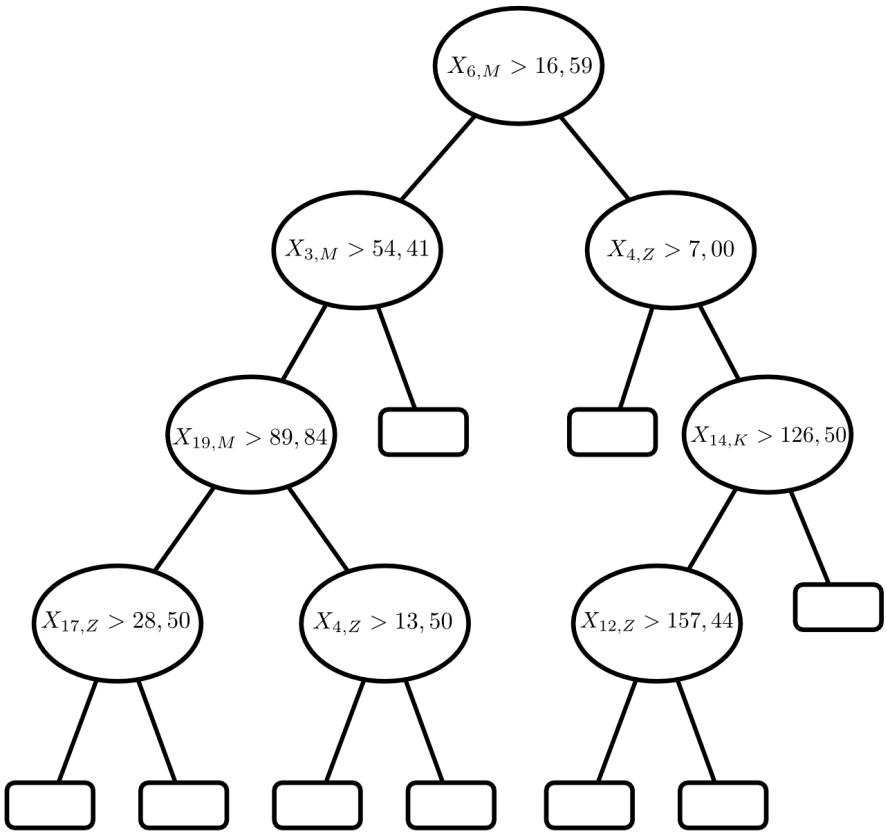


Abbildung 47: NHEMOtree-Individuum mit X_{VIM_6} und minimaler Fehlklassifikationsrate

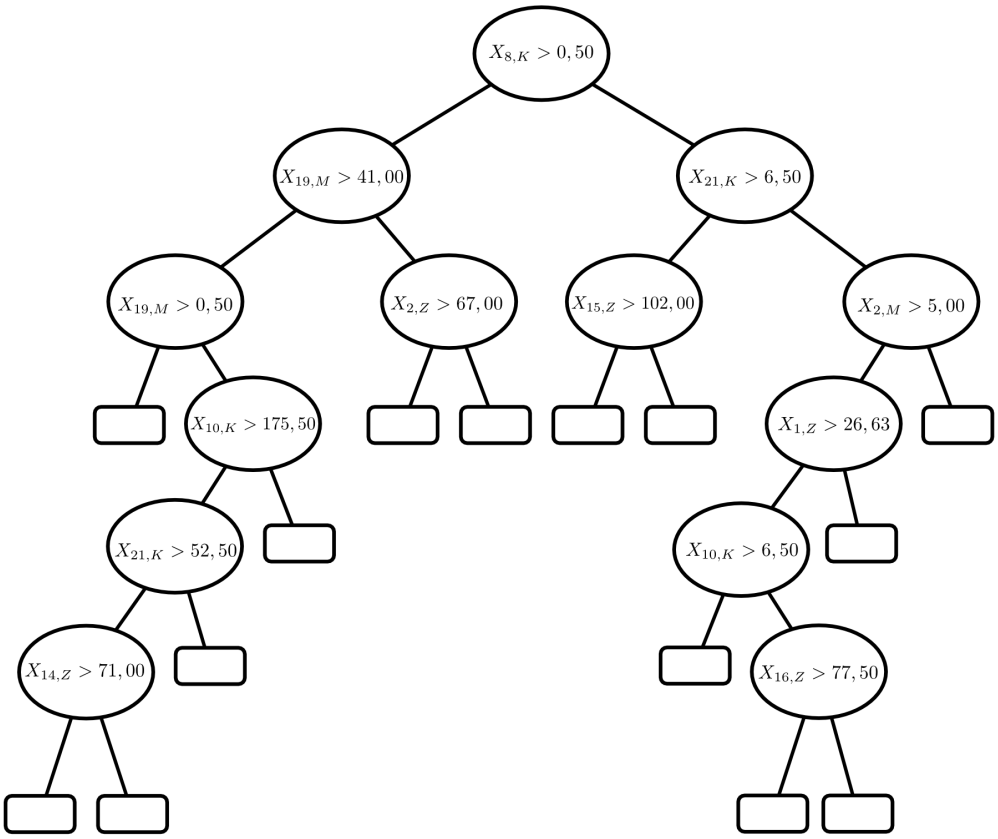


Abbildung 48: NHEMOtree$_{\text{FKR}}$-Individuum mit X_S und minimaler Fehlklassifikationsrate

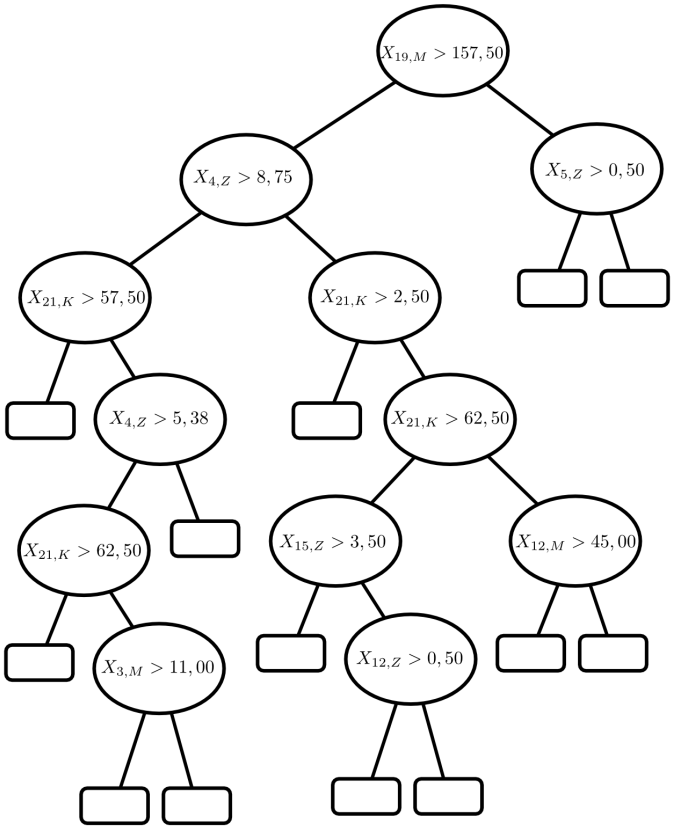


Abbildung 49: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_1} und minimaler Fehlklassifikationsrate

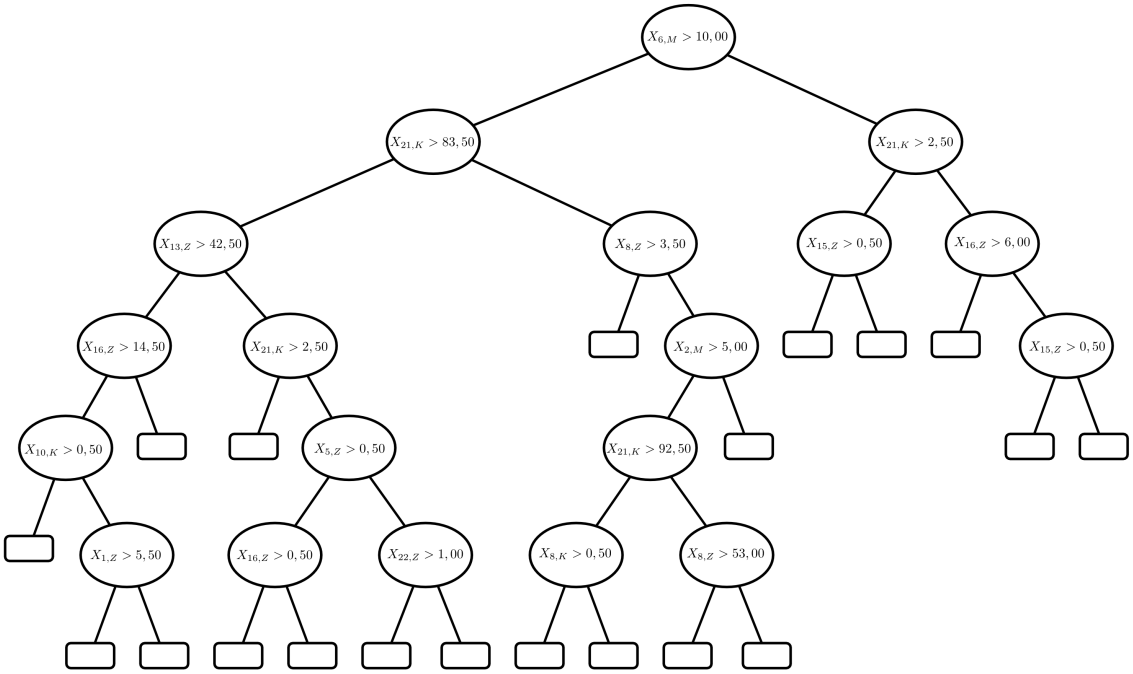


Abbildung 50: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_2} und minimaler Fehlklassifikationsrate

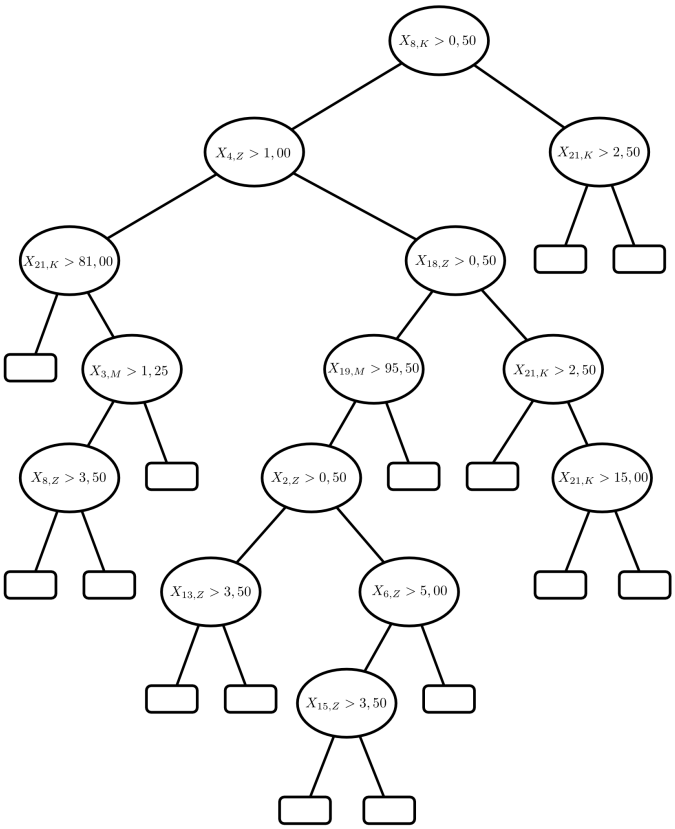


Abbildung 51: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_3} und minimaler Fehlklassifikationsrate

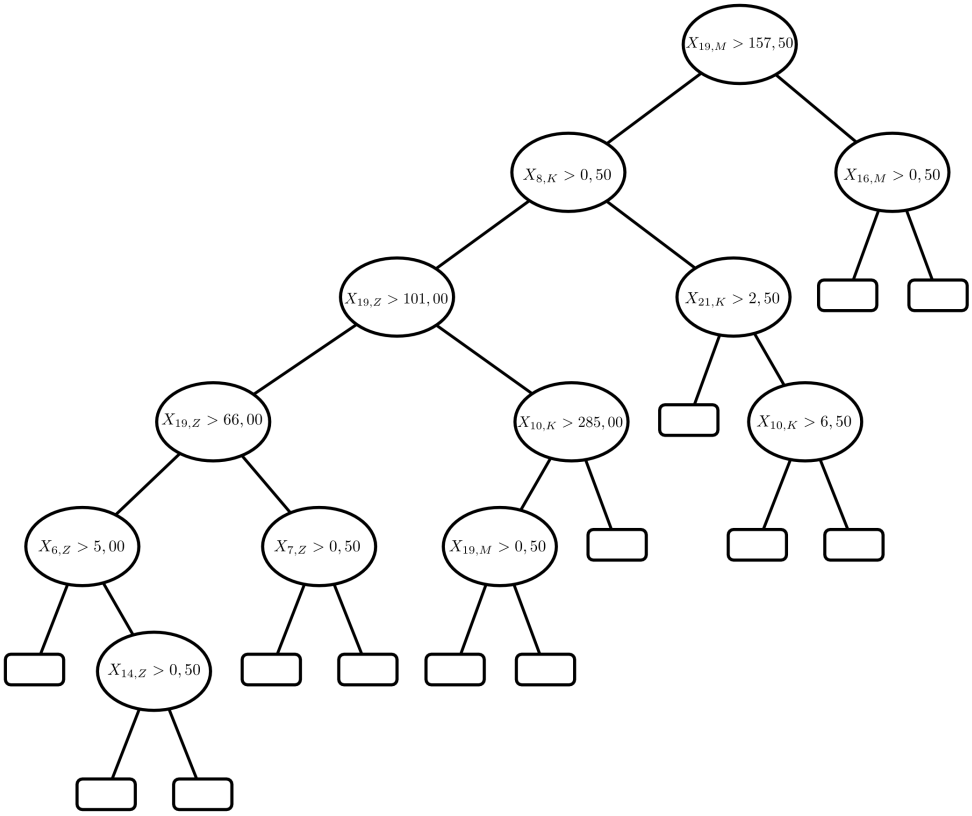


Abbildung 52: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_4} und minimaler Fehlklassifikationsrate

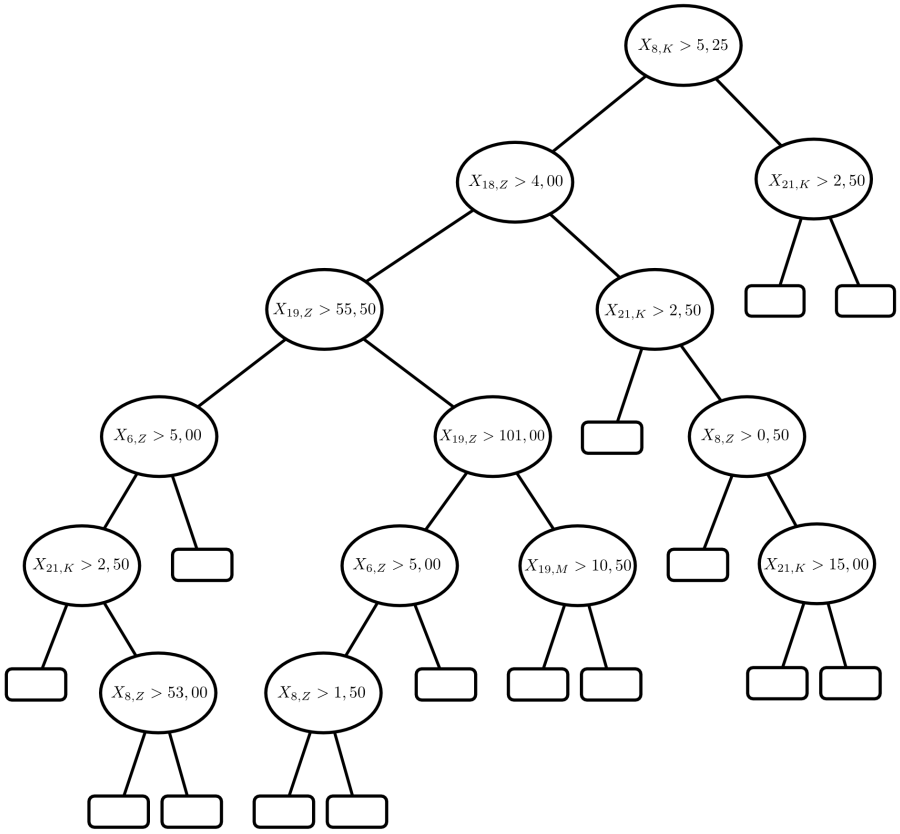


Abbildung 53: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_5} und minimaler Fehlklassifikationsrate

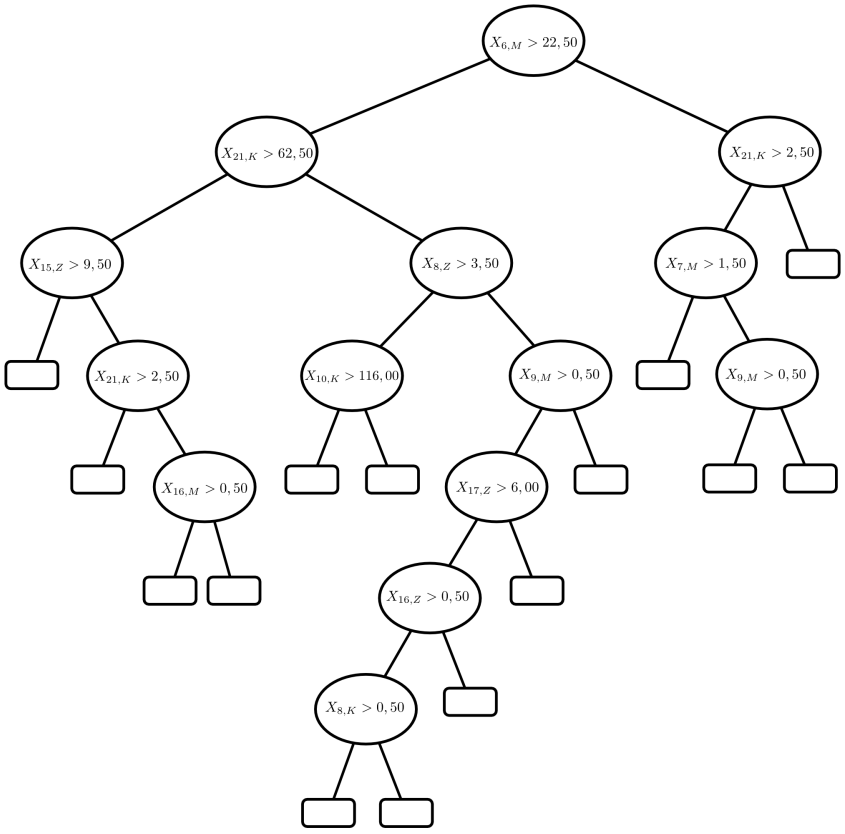


Abbildung 54: NHEMOtree$_{\text{FKR}}$-Individuum mit X_{VIM_6} und minimaler Fehlklassifikationsrate

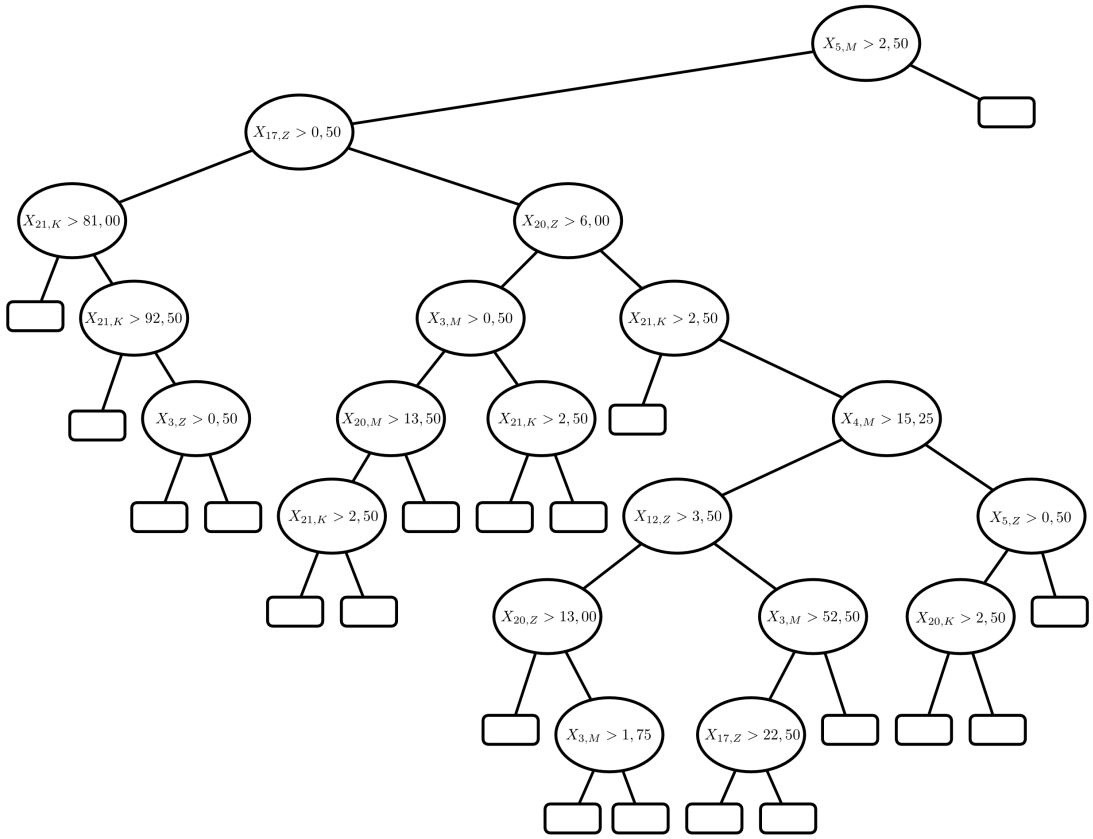


Abbildung 55: NHEMOtree$_{\text{Gini}}$-Individuum mit X_S und minimaler Fehlklassifikationsrate

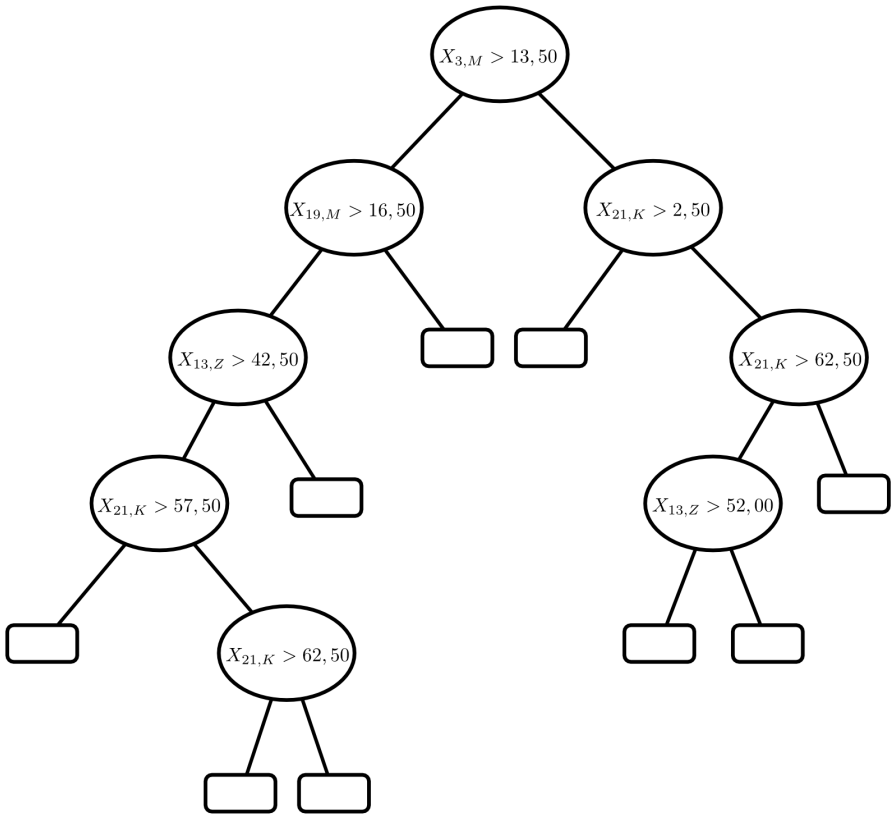


Abbildung 56: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_1} und minimaler Fehlklassifikationsrate

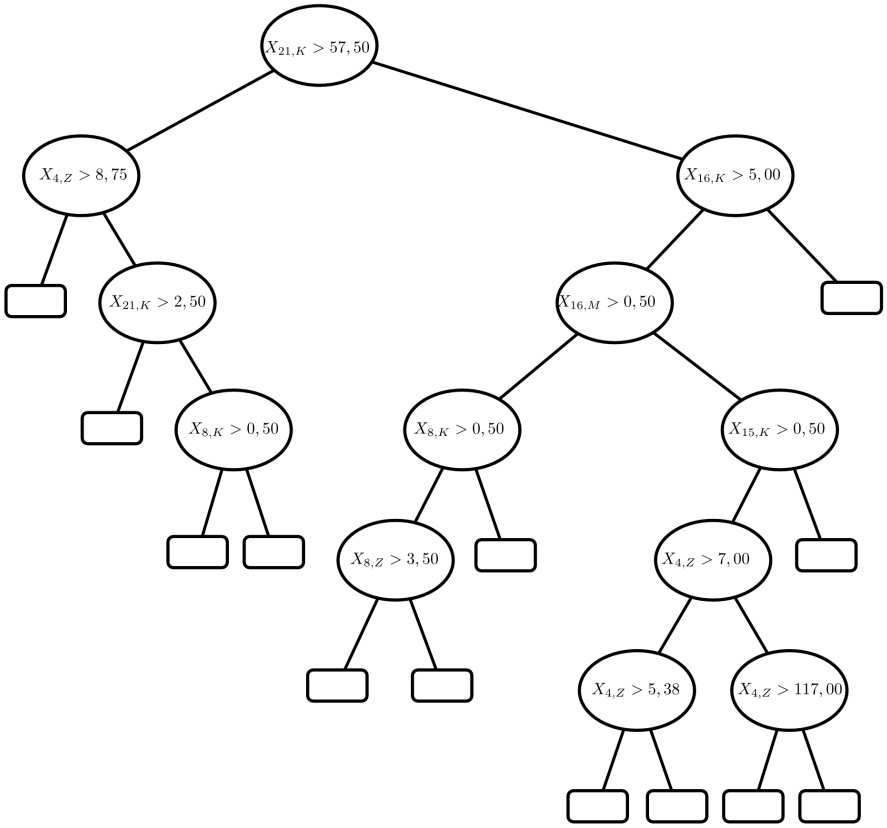


Abbildung 57: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_2} und minimaler Fehlklassifikationsrate

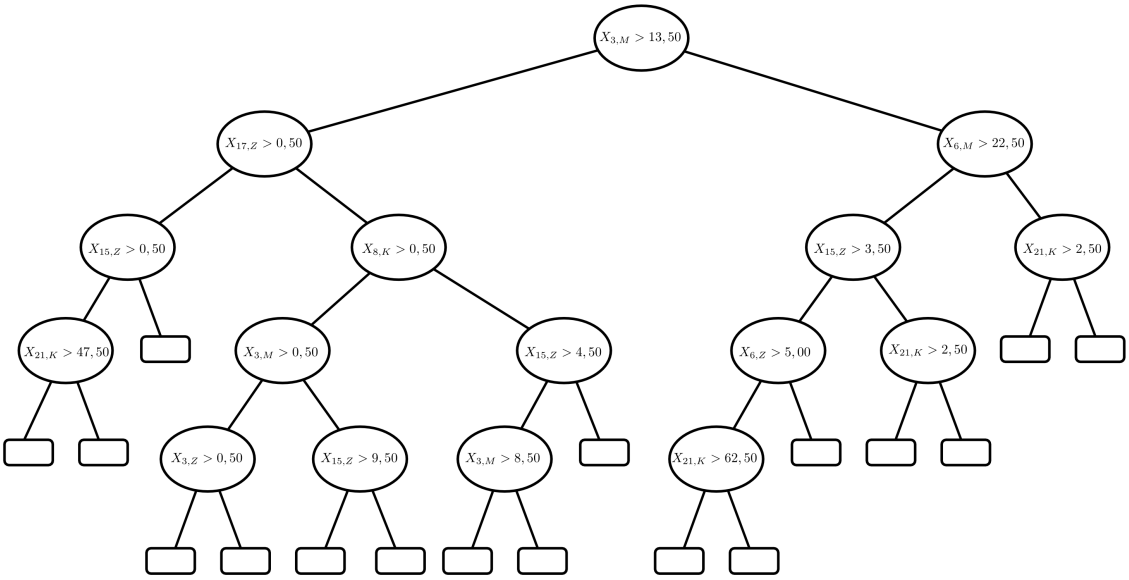


Abbildung 58: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_3} und minimaler Fehlklassifikationsrate

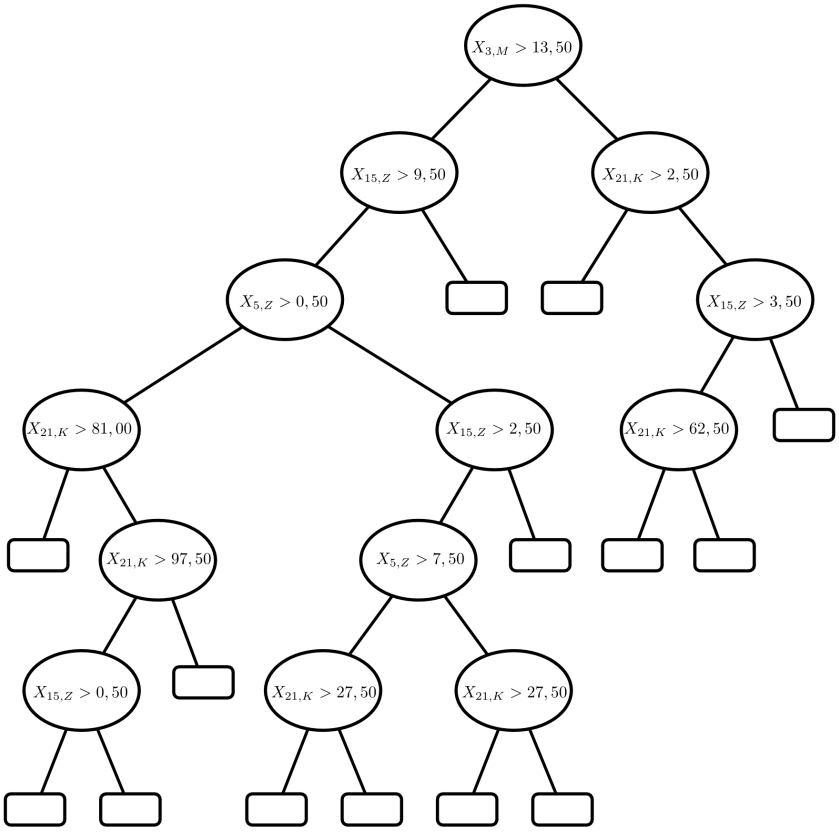


Abbildung 59: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_4} und minimaler Fehlklassifikationsrate

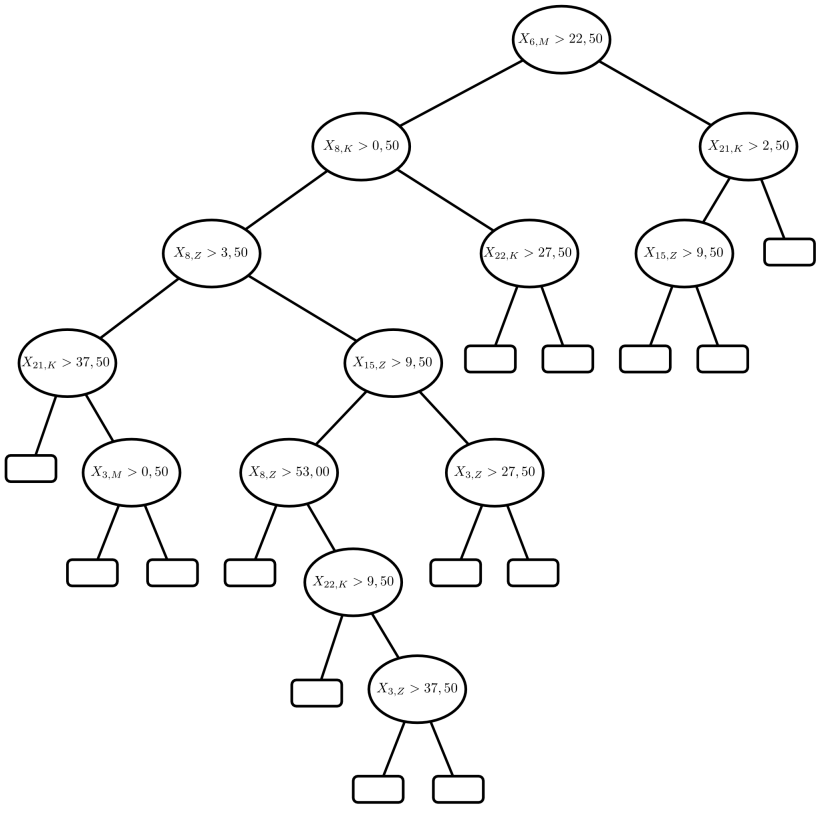


Abbildung 60: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_5} und minimaler Fehlklassifikationsrate

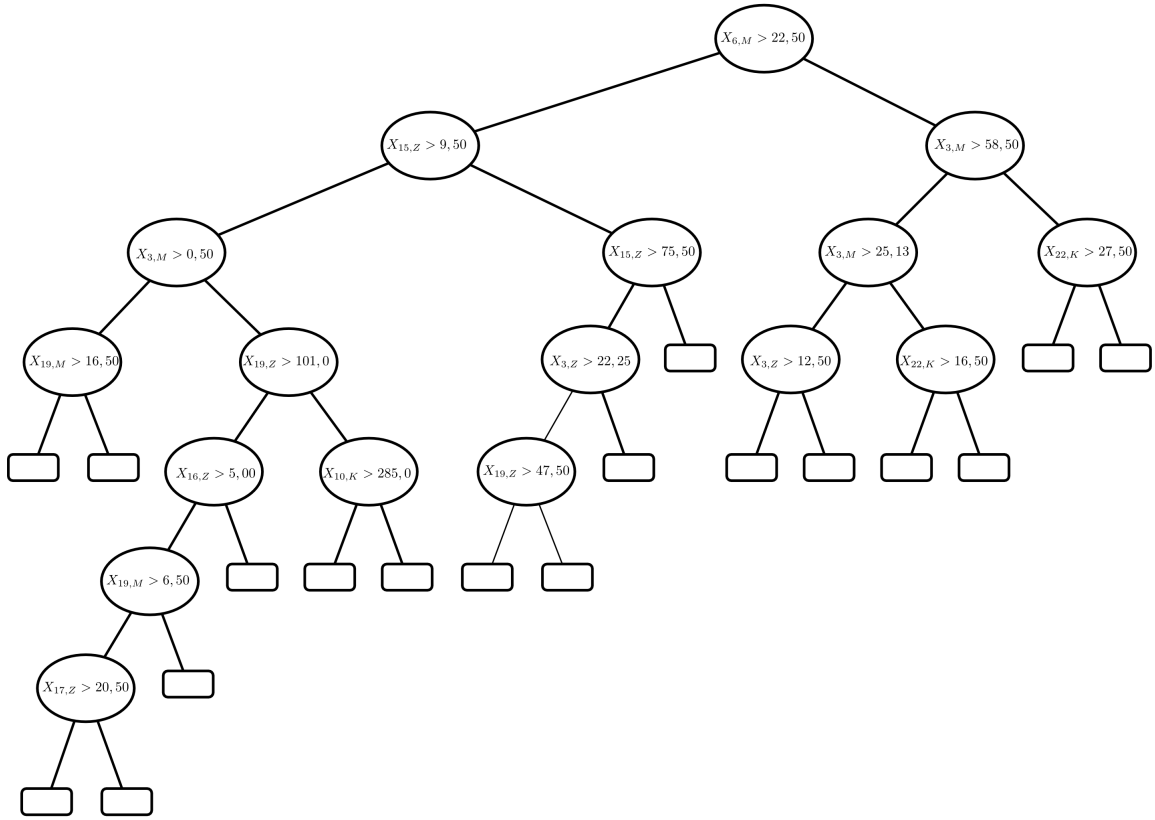


Abbildung 61: NHEMOtree$_{\text{Gini}}$-Individuum mit X_{VIM_6} und minimaler Fehlklassifikationsrate

D Tabellen

D.1 Ergebnisse des mehrkriteriellen *Wrapper*-Ansatzes

Kostenfunktion	Individuum	Variablen	LOOCV-FKR [%]	Kosten	Antikörperanzahl
Finanzen	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{3,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ $X_{21,\mathrm{K}}$	8,39	8,357	3
Anzahl	1	$X_{4,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	8,39	15,077	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
Finanzen und Anzahl	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
	3	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	10,49	9,172	3

Tabelle A2: Nicht-dominierte Individuen der *Steady-State* NSGA-II-*Wrapper*, die die Standard-CART-Lösung dominieren

Kostenfunktion	Individuum	Variablen	LOOCV-FKR [%]	Kosten	Antikörperanzahl
Finanzen	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
	3	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	10,49	9,172	3
Anzahl	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
Finanzen und Anzahl	1	$X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	6,99	11,823	3
	2	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$	9,79	11,047	2
	3	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$	10,49	9,172	3
	4	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	10,49	9,172	3

Tabelle A3: Nicht-dominierte Individuen der SMS-EMOA-*Wrapper*, die die Standard-CART-Lösung dominieren

D.2 Ergebnisse des NHEMOtrees

Initialisierung	Elternselektion	Rekombination	VIM	ν_{max}	Cutoff-Wahl
Ramped-half-and-half	Turnier, $\tau = 4$	X_S	0	20	Gauß-Mutation
Optimiert	Turnier, $\tau = 4$	X_S	0	20	Gauß-Mutation
Ramped-half-and-half	Winkler	X_S	0	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_S	0	10	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_B	0	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_P	0	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	1	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	2	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	3	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	4	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	5	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	6	20	Gauß-Mutation
Ramped-half-and-half	Turnier, $\tau = 4$	X_S	0	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	1	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	2	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	3	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	4	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	5	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	6	20	Lokal mit γ^F
Ramped-half-and-half	Turnier, $\tau = 4$	X_S	0	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	1	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	2	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	3	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	4	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	5	20	Lokal mit γ^G
Ramped-half-and-half	Turnier, $\tau = 4$	X_{VIM}	6	20	Lokal mit γ^G

ν_{max}: Maximale Knotenanzahl im Baum
γ^F: Fehlklassifikationsrate
γ^G: Gini-Wichtigkeit

Tabelle A4: Untersuchte Szenarien mit NHEMOtree und NSGA-II als Umweltselektionsoperator und der Fehlklassifikationsrate und finanzielle Kosten als Fitnessfunktionen

NHEMOtree-Individuen mit maximaler Generationenanzahl als Abbruchkriterium

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{10,\mathrm{K}}$ $X_{12,\mathrm{Z}}$ $X_{12,\mathrm{K}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$	4,22	26,201
2	$X_{6,\mathrm{M}}$ $X_{1,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{12,\mathrm{Z}}$	4,86	20,257
3	$X_{19,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 2x $X_{14,\mathrm{Z}}$ $X_{6,\mathrm{Z}}$ $X_{7,\mathrm{M}}$ $X_{12,\mathrm{Z}}$	5,58	15,298
4	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ 2x $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$ 2x $X_{4,\mathrm{M}}$ $X_{1,\mathrm{Z}}$	5,95	13,189
5	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$ $X_{4,\mathrm{M}}$ $X_{3,\mathrm{M}}$	6,46	12,163
6	$X_{19,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 3x	6,95	11,789

Tabelle A5 - *Fortsetzung auf nächster Seite*

Individuum	Variablen	LOOCV-FKR [%]	Kosten
	$X_{6,\mathrm{Z}}$ $X_{12,\mathrm{Z}}$		
7	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$	6,96	9,948
8	$X_{19,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 2x $X_{6,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ $X_{7,\mathrm{M}}$	6,96	11,386
9	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	6,97	7,148
10	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	8,41	6,372
11	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 3x	13,26	6,117
12	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{8,\mathrm{Z}}$	18,86	4,722
13	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ 2x	23,81	3,945
14	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{3,\mathrm{Z}}$	29,41	2,906
15	$X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ 2x	32,22	2,651
16	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{16,\mathrm{Z}}$	32,83	1,667
17	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$	34,45	1,557

Tabelle A5 - *Fortsetzung auf nächster Seite*

Individuum	Variablen	LOOCV-FKR [%]	Kosten
18	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ 2x	39,01	1,145
19	$X_{6,\mathrm{M}}$ $X_{9,\mathrm{Z}}$	44,57	0,899
20	$X_{15,\mathrm{Z}}$ 3x	46,85	0,255
21	$X_{9,\mathrm{M}}$ $X_{9,\mathrm{Z}}$	63,47	0,008

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A5: Finale Individuen des NHEMOtrees ($\mathrm{X_S}$)

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{18,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ 2x $X_{19,\mathrm{M}}$ 2x	6,27	12,044
2	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{18,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	**6,28**	**11,522**
3	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{16,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$	**6,95**	**11,280**
4	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{13,\mathrm{Z}}$	**6,96**	**10,504**
5	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	**6,98**	**8,175**
6	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	7,68	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A6: Finale Individuen des NHEMOtrees (VIM_1), die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{19,\mathrm{M}}$ $X_{6,\mathrm{M}}$ $X_{2,\mathrm{Z}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{Z}}$ 2x $X_{13,\mathrm{Z}}$	6,21	14,975
2	$X_{1,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ $X_{3,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{13,\mathrm{Z}}$	6,33	13,037
3	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{3,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$	6,76	12,265
4	$X_{6,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{Z}}$	**6,83**	**11,276**
5	$X_{6,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{13,\mathrm{Z}}$	**6,97**	**10,504**
6	$X_{1,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{M}}$	7,66	10,199
7	$X_{19,\mathrm{M}}$ $X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ 2x $X_{16,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	8,18	9,948
8	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	8,23	8,133
9	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$	9,03	7,148
10	$X_{6,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	9,76	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A7: Finale Individuen des NHEMOtrees (VIM_2), die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{19,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ $X_{21,\mathrm{K}}$	6,27	15,179
2	$X_{6,\mathrm{Z}}$ 2x $X_{17,\mathrm{Z}}$ 2x $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	6,29	12,553
3	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{17,\mathrm{Z}}$ 2x	6,98	12,298
4	$X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{15,\mathrm{Z}}$ 2x $X_{6,\mathrm{Z}}$	7,68	12,192
5	$X_{6,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{16,\mathrm{M}}$	7,69	11,026
6	$X_{6,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	8,06	10,249
7	$X_{6,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ 3x $X_{19,\mathrm{M}}$ 2x $X_{7,\mathrm{Z}}$ $X_{9,\mathrm{Z}}$	8,22	9,477
8	$X_{6,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	8,41	6,372

Tabelle A8: Finale Individuen des NHEMOtrees (VIM_3), die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{12,\mathrm{Z}}$ $X_{8,\mathrm{K}}$	5,55	13,083
2	$X_{6,\mathrm{Z}}$ $X_{17,\mathrm{Z}}$ 2x $X_{21,\mathrm{K}}$ 2x	6,31	12,892
3	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{K}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$	**6,80**	**11,187**
4	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,98**	**9,172**
5	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ 2x $X_{15,\mathrm{Z}}$ $X_{16,\mathrm{M}}$	9,08	7,560
6	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ 2x	9,76	6,783

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A9: Finale Individuen des NHEMOtrees (VIM_4), die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{6,\mathrm{M}}$ $X_{17,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ 2x	5,20	14,471
2	$X_{4,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{1,\mathrm{M}}$ $X_{6,\mathrm{Z}}$ 2x	5,57	14,229
3	$X_{6,\mathrm{Z}}$ 2x $X_{14,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{17,\mathrm{Z}}$ 2x	5,61	12,710
4	$X_{6,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{6,\mathrm{M}}$ $X_{17,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	6,11	12,553
5	$X_{6,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{12,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	6,26	12,044
6	$X_{6,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{K}}$	**6,83**	**11,187**
7	$X_{6,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{M}}$ $X_{6,\mathrm{M}}$	**6,93**	**10,199**
8	$X_{6,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,95**	**8,133**
9	$X_{6,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ 2x $X_{16,\mathrm{M}}$	9,78	7,148

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A10: Finale Individuen des NHEMOtrees (VIM_5), die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{17,\mathrm{Z}}$ $X_{8,\mathrm{M}}$	5,96	15,090
2	$X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{17,\mathrm{Z}}$ $X_{6,\mathrm{Z}}$	6,09	14,314
3	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$ $X_{14,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$	**6,27**	**10,915**
4	$X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,98**	**8,133**
5	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	7,67	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A11: Finale Individuen des NHEMOtrees (VIM_6), die CART dominieren

NHEMOtree mit OCD-Abbruchkriterium

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{21,\mathrm{K}}$ $X_{19,\mathrm{Z}}$ 4x $X_{6,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$ $X_{16,\mathrm{M}}$ 2x $X_{19,\mathrm{M}}$ $X_{14,\mathrm{Z}}$	3,47	15,141
2	$X_{21,\mathrm{K}}$ $X_{19,\mathrm{Z}}$ 4x $X_{6,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$	3,47	12,969
3	$X_{6,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ 2x	4,69	12,265
4	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ 2x $X_{14,\mathrm{Z}}$ 2x	**4,86**	**11,437**
5	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ 3x $X_{15,\mathrm{Z}}$	**5,35**	**11,280**
6	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ 3x	**5,42**	**11,026**
7	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$ 4x $X_{19,\mathrm{M}}$	**5,42**	**10,504**

Tabelle A12 - *Fortsetzung auf nächster Seite*

Individuum	Variablen	LOOCV-FKR [%]	Kosten
	$X_{6,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$		
8	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{1,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{9,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{3,\mathrm{Z}}$	**6,08**	**9,944**
9	$X_{6,\mathrm{M}}$ 3x $X_{16,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,09**	**7,148**
10	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 3x	7,63	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A12: Finale Individuen des NHEMOtrees ($\mathrm{X_S}$) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ 2x	4,02	14,992
2	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ 2x $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ 2x $X_{14,\mathrm{K}}$ $X_{15,\mathrm{Z}}$	4,15	13,380
3	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	4,18	12,969
4	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{21,\mathrm{K}}$ 2x $X_{19,\mathrm{M}}$ 2x	4,88	12,714
5	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ 2x	4,89	12,192
6	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{18,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	6,22	11,522
7	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$	6,25	11,026
8	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{3,\mathrm{M}}$ 2x $X_{1,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$	6,28	9,9936
9	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$	6,94	9,062

Tabelle A13 – *Fortsetzung auf nächster Seite*

Individuum	Variablen	LOOCV-FKR [%]	Kosten
	$X_{5,\mathrm{Z}}$ $X_{1,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$ 2x		
10	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{Z}}$ $X_{15,\mathrm{Z}}$	6,95	8,175
11	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{1,\mathrm{Z}}$	7,00	7,920
12	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	7,65	7,148
13	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$	8,35	6,894
14	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$	8,39	6,372
15	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 3x $X_{19,\mathrm{Z}}$	10,43	6,117

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A13: Finale Individuen des NHEMOtrees (VIM_1) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ 2x $X_{1,\mathrm{M}}$ $X_{3,\mathrm{M}}$	2,08	14,980
2	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ 2x $X_{1,\mathrm{M}}$	2,76	13,219
3	$X_{3,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{15,\mathrm{Z}}$	4,19	13,062
4	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 3x $X_{21,\mathrm{K}}$ $X_{19,\mathrm{Z}}$	4,20	12,192
5	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 5x $X_{3,\mathrm{Z}}$ 2x $X_{1,\mathrm{M}}$ 2x	**4,87**	**9,159**
6	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ $X_{1,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{16,\mathrm{M}}$	**5,41**	**8,175**
7	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{1,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,10**	**7,399**
8	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{16,\mathrm{M}}$	**6,92**	**7,148**
9	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	7,66	6,372
10	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$	10,44	6,117

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A14: Finale Individuen des NHEMOtrees (VIM_2) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{7,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 3x	4,67	14,123
2	$X_{6,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 3x $X_{1,\mathrm{Z}}$ 2x $X_{15,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ 2x	**4,88**	**11,531**
3	$X_{6,\mathrm{M}}$ $X_{13,\mathrm{Z}}$ 2x $X_{15,\mathrm{Z}}$ 2x $X_{19,\mathrm{M}}$ 2x	**6,26**	**10,504**
4	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{K}}$ $X_{15,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ $X_{1,\mathrm{Z}}$	**6,31**	**10,453**
5	$X_{6,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	**6,97**	**9,172**
6	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{Z}}$	7,00	8,175
7	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{3,\mathrm{M}}$	7,70	8,133
8	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{14,\mathrm{Z}}$	7,73	7,560
9	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$	8,33	7,148
10	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x	9,06	6,894
11	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	9,74	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A15: Finale Individuen des NHEMOtrees (VIM_3) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ 2x	2,79	14,992
2	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ 2x $X_{3,\mathrm{M}}$ 2x $X_{5,\mathrm{Z}}$ 2x $X_{8,\mathrm{K}}$	**4,69**	**11,819**
3	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{K}}$ 2x $X_{3,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$	**4,72**	**11,709**
4	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{K}}$ $X_{3,\mathrm{M}}$	**4,72**	**11,187**
5	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 3x $X_{19,\mathrm{Z}}$ 2x $X_{8,\mathrm{K}}$ 2x $X_{3,\mathrm{M}}$ 2x	**4,85**	**10,932**
6	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ 2x $X_{14,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{K}}$ 2x $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$	**4,91**	**10,360**
7	$X_{6,\mathrm{M}}$ $X_{6,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{K}}$ 2x $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{5,\mathrm{Z}}$	**5,44**	**10,059**

Tabelle A16 – *Fortsetzung auf nächster Seite*

Individuum	Variablen	LOOCV-FKR [%]	Kosten
8	$X_{6,\mathrm{M}}$ $X_{14,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{K}}$ 2x	**5,53**	**9,583**
9	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$ $X_{8,\mathrm{K}}$	**5,55**	**9,426**
10	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ 2x $X_{19,\mathrm{Z}}$	**5,57**	**9,172**
11	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{16,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{1,\mathrm{Z}}$	**6,99**	**8,175**
12	$X_{6,\mathrm{M}}$ 2x $X_{3,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	7,00	8,133
13	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x $X_{16,\mathrm{M}}$	7,66	7,148
14	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$ 2x	7,69	6,372
15	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{19,\mathrm{Z}}$	10,47	6,117

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A16: Finale Individuen des NHEMOtrees (VIM_4) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{19,\mathrm{M}}$ $X_{4,\mathrm{Z}}$ $X_{4,\mathrm{M}}$ 2x $X_{6,\mathrm{M}}$ 2x $X_{13,\mathrm{Z}}$ 3x $X_{19,\mathrm{Z}}$	3,85	14,280
2	$X_{4,\mathrm{M}}$ $X_{6,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ 2x $X_{8,\mathrm{K}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	4,87	13,978
3	$X_{3,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$	4,91	13,062
4	$X_{6,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{14,\mathrm{Z}}$ $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{M}}$	**5,60**	**10,610**
5	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$ $X_{13,\mathrm{Z}}$	**6,10**	**10,504**
6	$X_{6,\mathrm{Z}}$ $X_{6,\mathrm{M}}$ $X_{8,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{M}}$	**6,27**	**10,199**
7	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{3,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{1,\mathrm{M}}$	**6,30**	**9,159**
8	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 3x $X_{15,\mathrm{Z}}$ $X_{16,\mathrm{M}}$	7,48	7,148
9	$X_{6,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 4x $X_{15,\mathrm{Z}}$	7,65	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A17: Finale Individuen des NHEMOtrees (VIM_5) mit OCD-Abbruchkriterium, die CART dominieren

Individuum	Variablen	LOOCV-FKR [%]	Kosten
1	$X_{6,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{1,\mathrm{M}}$	2,77	14,980
2	$X_{6,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$	2,80	14,729
3	$X_{6,\mathrm{M}}$ $X_{21,\mathrm{K}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$	3,49	13,953
4	$X_{6,\mathrm{Z}}$ $X_{21,\mathrm{K}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	4,20	12,969
5	$X_{6,\mathrm{Z}}$ $X_{16,\mathrm{M}}$ X_7, Z $X_{19,\mathrm{M}}$ 3x $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$	5,56	12,006
6	$X_{6,\mathrm{M}}$ $X_{3,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ $X_{19,\mathrm{Z}}$ $X_{1,\mathrm{M}}$	5,59	9,159
7	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{Z}}$	6,25	8,133
8	$X_{6,\mathrm{M}}$ $X_{16,\mathrm{M}}$ $X_{19,\mathrm{M}}$ 2x $X_{15,\mathrm{Z}}$	7,66	7,148
9	$X_{6,\mathrm{M}}$ $X_{15,\mathrm{Z}}$ $X_{19,\mathrm{M}}$ 2x	7,67	6,372

Fett gedruckte Individuen dominieren das NSGA-II-*Wrapper*-Individuum mit der geringsten LOOCV-Fehlklassifikationsrate.

Tabelle A18: Finale Individuen des NHEMOtrees (VIM_6) mit OCD-Abbruchkriterium, die CART dominieren

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Literatur

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

E. Alba, J. Garcia-Nieto, L. Jourdan und E.G. Talbi. Gene selection in cancer classification using pso/svm and ga/svm hybrid algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 284–290, 2007.

L. Altenberg. Emergent phenomena in genetic programming. In *Proceedings of the Annual Conference on Evolutionary Programming*, 233–241, 1994.

P.J. Angeline. An investigation into the sensitivity of genetic programming to the frequency of leaf selection during subtree crossover. In *Proceedings of the Annual Conference on Genetic Programming*, 21–29, 1996.

J. Bader und E. Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

K. Badran und P. Rockett. Integrating categorical variables with multiobjective genetic programming for classifier construction. *LNCS - Genetic Programming*, 4971:301–311, 2008.

W. Banzhaf, P. Nordin, R. Keller und F. Francone. *Genetic Programming: An Introduction*. Morgan Kaufmann, 1998.

T. Bartz-Beielstein, J. Ziegenhirt, W. Konen, O. Flasch, M. Friese, P. Koch, M. Zaefferer und B. Naujoks. *SPOT: Sequential Parameter Optimization*, 2012. `http://CRAN.R-project.org/package=SPOT`.

S.C. Berney, I.R. Gordon, H.I. Opdam und L. Denehy. A classification and regression tree to assist clinical decision making in airway management for patients with cervical spinal cord injury. *Spinal Cord*, 49(2):244–250, 2010.

N. Beume, B. Naujoks und M. Emmerich. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3): 1653–1669, 2007.

H.G. Beyer. An alternative explanation for the manner in which genetic algorithms operate. *BioSystems*, 41(1):1–15, 1997.

H.G. Beyer und H.P. Schwefel. Evolution strategies–a comprehensive introduction. *Natural computing*, 1(1):3–52, 2002.

B. Bhanu und Y. Lin. Learning composite operators for object detection. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1003–1010, 2002.

U. Bhowan, M. Johnston und M. Zhang. Evolving ensembles in multi-objective genetic programming for classification with unbalanced data. In *Proceedings of the Annual Conference on Genetic and Evolutionary Computation*, 1331–1338, 2011.

U. Bhowan, M. Johnston und M. Zhang. Developing new fitness functions in genetic programming for classification with unbalanced data. In *IEEE Transactions on Systems, Man und Cybernetics, Part B: Cybernetics*, 42:406–421, 2012.

J. Biles. Genjam: A genetic algorithm for generating jazz solos. In *Proceedings of the International Computer Music Conference*, 131–131, 1994.

J.M. Bland und D.G. Altman. Multiple significance tests: the Bonferroni method. *BMJ*, 310(6973):170, 1995.

C. Blum und A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

C.C. Bojarczuk, H.S. Lopes, A.A. Freitas und E.L. Michalkiewicz. A constrained-syntax genetic programming system for discovering classification rules: application to medical data sets. *Artificial Intelligence in Medicine*, 30(1):27–48, 2004.

A. Bosin, N. Dessí und B. Pes. A cost-sensitive approach to feature selection in micro-array data classification. *LNCS - Applications of Fuzzy Sets Theory*, 4578: 571–579, 2007.

M.C.J. Bot und W.B. Langdon. Application of genetic programming to induction of linear classification trees. *LNCS - Genetic Programming*, 1802:247–258, 2000.

V.J. Bowman. On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. *Multiple Criteria Decision Making*, 135:76–85, 1976.

T. Brabletz, A. Jung, S. Spaderna, F. Hlubek und T. Kirchner. Migrating cancer stem cells - an integrated concept of malignant tumour progression. *Nature Reviews Cancer*, 5(9):744–749, 2005.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman, J. Friedman, C. J. Stone und R.A. Olshen. *Classification and regression trees*. Chapman and Hall/CRC, 1998.

H.J. Bremermann. Optimization through evolution and recombination. In *Self-organizing systems*, 93–106. Spartan, 1962.

D. Brockhoff und E. Zitzler. Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2086–2093, 2007.

L.T. Bui, S. Wesolkowski, A. Bender, H.A. Abbass und M. Barlow. A dominance-based stability measure for multi-objective evolutionary algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 749–756, 2009.

A. Bureau, J. Dupuis, K. Falls, K.L. Lunetta, B. Hayward, T.P. Keith und P. Van Eerdewegh. Identifying snps predictive of phenotype using Random Forests. *Genetic Epidemiology*, 28(2):171–182, 2005.

E. Carreno, G. Leguizamón und N. Wagner. Evolution of classification rules for comprehensible knowledge discovery. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1261–1268, 2007.

S. Casjens. *NHEMOtree: Non-hierarchical evolutionary multi-objective tree learner to perform cost-sensitive classification*, 2013. `http://CRAN.R-project.org/package=NHEMOtree`.

M.G. Castillo Tapia und C.A. Coello Coello. Applications of multi-objective evolutionary algorithms in economics and finance: A survey. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 532–539, 2007.

N.S. Chaudhari, A. Purohit und A. Tiwari. A multiclass classifier using genetic programming. In *Proceedings of the International Conference on Control, Automation, Robotics and Vision*, 1884–1887, 2008.

K. Chellapilla. Evolving computer programs without subtree crossover. *IEEE Transactions on Evolutionary Computation*, 1(3):209–216, 1997.

K.J. Cherkauer und J.W. Shavlik. Growing simpler decision trees to facilitate knowledge discovery. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 315–318, 1996.

K. Chrysostomou, S.Y. Chen und X. Liu. Identifying user preferences with wrapper-based decision trees. *Expert Systems with Applications*, 38(4):3294–3303, 2011.

C.A. Coello Coello. Evolutionary multi-objective optimization: a historical view of the field. *IEEE Transactions on Computational Intelligence Magazine*, 1:28–36, 2006.

C.A. Coello Coello. Evolutionary multi-objective optimization: some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China*, 3(1):18–30, 2009.

C.A. Coello Coello, G.B. Lamont und D.A. Van Veldhuizen. *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.

T.H. Cormen, C.E. Leiserson, R.L. Rivest und C. Stein. *Introduction to algorithms*. MIT Press, 2001.

C. Cortes und V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

C. Darwin. *On the origin of species*. John Murray, 1859.

I. Das und J.E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural and Multidisciplinary Optimization*, 14(1):63–69, 1997.

R. Das und A. Sengur. Evaluation of ensemble methods for diagnosing of valvular heart disease. *Expert Systems with Applications*, 37(7):5110–5115, 2010.

S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *Proceedings of the International Conference on Machine Learning*, 74–81, 2001.

M. Dash und H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(3):131–156, 1997.

E.D. De Jong und J.B. Pollack. Multi-objective methods for tree size control. *Genetic Programming and Evolvable Machines*, 4(3):211–233, 2003.

K.A. De Jong, W.M. Spears und D.F. Gordon. Using Markov chains to analyze GAFOs. In *Foundations of Genetic Algorithms 3*, 115–137. Morgan Kaufmann, 1995.

K.A. De Jong. *Evolutionary computation: A unified approach.* MIT Press, 2006.

K.A. De Jong. Parameter setting in EAs: A 30 year perspective. *Studies in Computational Intelligence - Parameter Setting in Evolutionary Algorithms*, 54: 1–18, 2007.

K. Deb. *Multi-objective optimization using evolutionary algorithms.* John Wiley, 2001.

K. Deb, A. Pratap, S. Agarwal und T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6 (2):182–197, 2002.

K. Deb, M. Mohan und S. Mishra. Towards a quick computation of well-spread pareto-optimal solutions. *LNCS - Evolutionary Multi-Criterion Optimization*, 2632:222–236, 2003.

P. Deuflhard und M. Weiser. *Numerische Mathematik 3 - Adaptive Lösung partieller Differentialgleichungen.* de Gruyter, 2011.

R. Díaz-Uriarte und A. de Andrés. Gene selection and classification of microarray data using Random Forest. *BMC Bioinformatics*, 7(3), 2006.

A. Dobra und J. Gehrke. Bias correction in classification tree construction. In *Proceedings of the International Conference on Machine Learning*, 90–97, 2001.

M. Dorigo und G. Di Caro. Ant colony optimization: A new meta-heuristic. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 11–32, 1999.

S. Dubuisson, F. Davoine und M. Masson. A solution for facial expression representation and recognition. *Signal Processing: Image Communication*, 17(9): 657–673, 2002.

J.G. Dy und C.E. Brodley. Feature selection for unsupervised learning. *The Journal of Machine Learning Research*, 5:845–889, 2004.

J. Eggermont, A. Eiben und J. van Hemert. A comparison of genetic programming variants for data classification. *LNCS - Advances in Intelligent Data Analysis*, 1642:281–290, 1999.

A.E. Eiben und G. Rudolph. Theory of evolutionary algorithms: A birds eye view. *Theoretical Computer Science*, 229:3–9, 1999.

A.E. Eiben und J.E. Smith. *Introduction to evolutionary computing.* Springer, 2003.

A.E. Eiben, R. Hinterding und Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, 1999.

J.M. Elias, M. Margiotta, und D. Gaborc. Sensitivity and detection efficiency of the peroxidase antiperoxidase (pap), avidin-biotin peroxidase complex (abc), and peroxidase-labeled avidin-biotin (lab) methods. *American Journal of Clinical Pathology*, 92(1):62–67, 1989.

C. Emmanouilidis, A. Hunter und J. MacIntyre. A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 309–316, 2000.

M. Emmerich, N. Beume und B. Naujoks. An emo algorithm using the hypervolume measure as selection criterion. *LNCS - Evolutionary Multi-Criterion Optimization*, 3410:62–76, 2005.

E.P. Ephzibah. Cost effective approach on feature selection using genetic algorithms and ls-svm classifier. *International Journal of Computer Applications*, 1(1):16–20, 2010.

P.G. Espejo, S. Ventura und F. Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man und Cybernetics, Part C: Applications and Reviews*, 40(2):121–144, 2010.

W. Fan, E.A. Fox, P. Pathak und H. Wu. The effects of fitness functions on genetic programming-based ranking discovery for web search. *Journal of the American Society for Information Science and Technology*, 55(7):628–636, 2004.

K.-T. Fang, D.K.J. Lin, P. Winker und Y. Zhang. Uniform design: Theory and application. *Technometrics*, 42(3):237–248, 2000.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

R. Fletcher. *Practical Methods of Optimization: Vol. 2: Constrained Optimization.* John Wiley, 1987.

L.J. Fogel, A.J. Owens und M.J. Walsh. *Artificial intelligence through simulated evolution.* John Wiley, 1966.

A.A. Freitas. *Data mining and knowledge discovery with evolutionary algorithms.* Springer, 2002.

Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, 1995.

J.H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001.

M.E. Garber, O.G. Troyanskaya, K. Schluens, S. Petersen, Z. Thaesler, M. Pacyna-Gengelbach, M. Van De Rijn, G.D. Rosen, C.M. Perou, R.I. Whyte, R.B. Altman, P.O. Brown, D. Botstein und I. Petersen. Diversity of gene expression in adenocarcinoma of the lung. *Proceedings of the National Academy of Sciences*, 98 (24):13784–13789, 2001.

J. Garcia-Nieto, E. Alba, L. Jourdan und E. Talbi. Sensitivity and specificity based multiobjective approach for feature selection: Application to cancer diagnosis. *Information Processing Letters*, 109(16):887–896, 2009.

S. Garte. Metabolic susceptibility genes as cancer risk factors. *Cancer Epidemiology Biomarkers & Prevention*, 10(12):1233–1237, 2001.

G.V. Glinsky, O. Berezovska und A.B. Glinskii. Microarray analysis identifies a death-from-cancer signature predicting therapy failure in patients with multiple types of cancer. *Journal of Clinical Investigation*, 115(6):1503–1521, 2005.

J.B. Gray und G. Fan. Classification tree analysis using target. *Computational Statistics & Data Analysis*, 52(3):1362–1372, 2008.

J.J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man und Cybernetics*, 16(1):122–128, 1986.

J.J. Grefenstette. Deception considered harmful. In *Foundations of Genetic Algorithms 2*, 75–91. Morgan Kaufmann, 1993.

J.L. Guerrero, J. Garcia, L. Marti, J.M. Molina und A. Berlanga. A stopping criterion based on Kalman estimation techniques with several progress indicators. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 587–594, 2009.

S. Gustafson, A. Ekárt, E. Burke und G. Kendall. Problem difficulty and code growth in genetic programming. *Genetic Programming and Evolvable Machines*, 5(3):271–290, 2004.

I. Guyon und A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

Y.Y. Haimes, L.S. Lasdon und D.A. Wismer. On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man und Cybernetics*, 1(3):296–297, 1971.

T. Hamdani, J.M. Won, A. Alimi und F. Karray. Multi-objective feature selection with nsga-ii. *LNCS - Adaptive and Natural Computing Algorithms*, 4431:240–247, 2007.

J. Han und M. Kamber. *Data mining: Concepts and techniques*. Morgan Kaufmann, 2006.

D. Hanahan und R.A. Weinberg. The hallmarks of cancer. *Cell*, 100(1):57–70, 2000.

D. Hanahan und R.A. Weinberg. Hallmarks of cancer: the next generation. *Cell*, 144(5):646–674, 2011.

M.P. Hansen und A. Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Mathematical Modeling, Technical University of Denmark, 1998.

N. Hansen, A. Ostermeier und A. Gawelczyk. On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. In *Proceedings of the International Conference on Genetic Algorithms*, 57–64, 1995.

K. Harries und P. Smith. Exploring alternative operators and search strategies in genetic programming. In *Proceedings of the 2nd Annual Conference on Genetic Programming*, 147–155, 1997.

J. Hartung, B. Elpelt und K.H. Klösener. *Statistik*. Oldenbourg, 2002.

S. Haruyama und Q. Zhao. Designing smaller decision trees using multiple objective optimization based gps. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 5, 2002.

T. Hastie, R. Tibshirani, M.B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W.C. Chan, D. Botstein und P. Brown. Gene shaving as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology*, 1(2):1–21, 2000.

T. Hastie, R. Tibshirani und J. Friedman. *The elements of statistical learning: Data mining, inference, and prediction*. Springer, 2009.

J.H. Holland. *Adaptation in natural and artificial systems*. The University of Michigan Press, 1975.

J.-H. Hong und S.-B. Cho. The classification of cancer based on DNA microarray data that uses diverse ensemble genetic programming. *Artificial Intelligence in Medicine*, 36(1):43–58, 2006.

R. Hooke und T.A. Jeeves. Direct search solution of numerical and statistical problems. *Journal of the Association for Computing Machinery*, 8(2):212–229, 1961.

J.E. Hopcroft, R. Motwani und J.D. Ullman. *Introduction to Automata Theory, Languages und Computation*. Addison-Wesley, 1979.

J. Horn, N. Nafpliotis und D.E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 82–87, 1994.

T. Hothorn und A. Zeileis. *partykit: A toolkit for recursive partytioning*, 2012. `http://CRAN.R-project.org/package=partykit`.

T. Hothorn, K. Hornik und A. Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3): 651–674, 2006.

A. Hunter. Expression inference - genetic symbolic classification integrated with non-linear coefficient optimisation. *Lecture Notes in Artificial Intelligence*, 2385: 117–127, 2002.

H. Ishibuchi und T. Nakashima. Multi-objective pattern and feature selection by a genetic algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1069–1076, 2000.

H. Ishwaran. Variable importance in binary regression trees and forests. *Electronic Journal of Statistics*, 1:519–537, 2007.

K. Iswandy und A. Koenig. Feature selection with acquisition cost for optimizing sensor system design. *Advances in Radio Science*, 4:135–141, 2006.

T. Ito, H. Iba und S. Sato. Depth-dependent crossover for genetic programming. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 775–780, 1998.

A. Ittner und M. Schlosser. Non-linear decision trees. In *Proceedings of the International Conference on Machine Learning*, 252–257, 1996.

H. Jabeen und A.R. Baig. A framework for optimization of genetic programming evolved classifier expressions using particle swarm optimization. *Lecture Notes in Artificial Intelligence*, 6076(2):56–63, 2010a.

H. Jabeen und A.R. Baig. Review of classification using genetic programming. *International Journal of Engineering Science and Technology*, 2(2):94–103, 2010b.

H. Jabeen und A.R. Baig. Depthlimited crossover in gp for classifier evolution. *Computers in Human Behavior*, 27(5):1475–1481, 2011.

A. Jain und D. Zongker. Feature selection: Evaluation, application und small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

A.K. Jain, R.P.W. Duin und J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

D.A. James und S. DebRoy. *RMySQL: R interface to the MySQL database*, 2009. `http://CRAN.R-project.org/package=RMySQL`.

A. Jemal, R. Siegel, E. Ward, Y. Hao, J. Xu und M.J. Thun. Cancer statistics, 2009. *CA: A Cancer Journal for Clinicians*, 59(4):225–249, 2009.

Y. Jin. *Multi-objective machine learning - Studies in Computational Intelligence*. Springer, 2006.

Y. Jin und B. Sendhoff. Pareto-based multiobjective machine learning: An overview and case studies. *IEEE Transactions on Systems, Man und Cybernetics, Part C: Applications and Reviews*, 38(3):397–415, 2008.

Y. Jin, M. Olhofer und B. Sendhoff. Dynamic weighted aggregation for evolutionary multi-objective optimization: Why does it work and how? In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1042–1049, 2001.

T. Juliusdottir, E. Keedwell, D. Corne und A. Narayanan. Two-phase ea/k-nn for feature selection and classification in cancer microarray datasets. In *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 1–8, 2005.

R.E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME - Journal of Basic Engineering*, 82(1):35–45, 1960.

J. Kennedy und R. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, 1942–1948, 1995.

M.W. Khan und M. Alam. A survey of application: Genomics and genetic programming, a new frontier. *Genomics*, 100(2):65–71, 2012.

T.M. Khoshgoftaar und Y. Liu. A multi-objective software quality classification model using genetic programming. *IEEE Transactions on Reliability*, 56(2):237–245, 2007.

C.F.B. Kim, E.L. Jackson, A.E. Woolfenden, S. Lawrence, I. Babar, S. Vogel, D. Crowley, R.T. Bronson und T. Jacks. Identification of bronchioalveolar stem cells in normal lung and lung cancer. *Cell*, 121(6):823–835, 2005.

H. Kim und W.Y. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96(454):589–604, 2001.

T.-W. Kim, D.-H. Koh und C.-Y. Park. Decision tree of occupational lung cancer using classification and regression analysis. *Safety and Health at Work*, 1(2):140–148, 2010.

K.E. Kinnear Jr. Generality and difficulty in genetic programming: Evolving a sort. In *Proceedings of the International Conference on Genetic Algorithms*, 287–294, 1993.

R. Kohavi und G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324, 1997.

C. Kooperberg und I. Ruczinski. Identifying interacting SNPs using Monte Carlo logic regression. *Genetic Epidemiology*, 28(2):157–170, 2005.

J.R. Koza. *Genetic programming: On the programming of computers by means of natural selection*. MIT Press, 1992.

D.H. Kraft, F.E. Petry, B.P. Buckles und T. Sadasivan. The use of genetic programming to build queries for information retrieval. In *Proceedings of the IEEE Conference on Evolutionary Computation*, 468–473, 1994.

K. Kshetrapalapuram und M. Kirley. Mining classification rules using evolutionary multi-objective algorithms. *LNCS - Knowledge-Based Intelligent Information and Engineering Systems*, 3683:959–965, 2005.

M. Kudo und J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(1):25–41, 2000.

D.J.N. Kumar, S.C. Satapathy und J.V.R. Murthy. A scalable genetic programming multi-class ensemble classifier. In *Proceedings of the IEEE World Congress on Nature & Biologically Inspired Computing*, 1201–1206, 2009.

T.N. Lal, O. Chapelle, J. Weston und A. Elisseeff. Embedded methods. *Studies in Fuzziness and Soft Computing Volume - Feature Extraction, Foundations and Applications*, 207:137–165, 2006.

W.B. Langdon und R. Poli. *Foundations of genetic programming*. Springer, 2002.

L. Li, C.R. Weinberg, T.A. Darden und L.G. Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the ga/knn method. *Bioinformatics*, 17(12):1131, 2001.

U. Ligges und M. Mächler. Scatterplot3d - an R package for visualizing multivariate data. *Journal of Statistical Software*, 8(11):1–20, 2003.

H. Liu und L. Yu. Toward integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 17(4): 491–502, 2005.

K.H. Liu und C.G. Xu. A genetic programming-based approach to the classification of multiclass microarray datasets. *Bioinformatics*, 25(3):331–337, 2009.

W.Y. Loh und Y.S. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7:815–840, 1997.

T. Loveard und V. Ciesielski. Representing classification problems in genetic programming. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1070–1077, 2001.

K. Lunetta, L.B. Hayward, J. Segal und P. Van Eerdewegh. Screening large-scale association study data: Exploiting interactions using Random Forests. *BMC Genetics*, 5(32), 2004.

H. Majeed und C. Ryan. A less destructive, context-aware crossover operator for gp. *LNCS - Genetic Programming*, 3905:36–48, 2006.

L. Martí, J. García, A. Berlanga und J.M. Molina. A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2835–2842, 2007.

K.S. McCarty Jr, L.S. Miller, E.B. Cox, J. Konrath und K.S. McCarty. Estrogen receptor analyses: correction of biochemical and immunohistochemical methods using monoclonal antireceptor antibodies. *Archives of Pathology & Laboratory Medicine*, 109(8):716–721, 1985.

M.D. McKay, R.J. Beckman und W.J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21:239–245, 1979.

B.H. Menze, W. Petrich und F.A. Hamprecht. Multivariate feature selection and hierarchical classification for infrared spectroscopy: serum-based detection of bovine spongiform encephalopathy. *Analytical and Bioanalytical Chemistry*, 387(5):1801–1807, 2007.

O. Mersmann. *emoa: Evolutionary Multiobjective Optimization Algorithms*, 2011. `http://CRAN.R-project.org/package=emoa`.

O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs und G. Rudolph. Exploratory landscape analysis. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 829–836, 2011.

D. Meyer und K. Hornik. Generalized and customizable sets in R. *Journal of Statistical Software*, 31(2):1–27, 2009.

Z. Michalewicz. *Genetic algorithms + Data structures = Evolution programs*. Springer, 1992.

J.N. Morgan und J.A. Sonquist. Problems in the analysis of survey data und a proposal. *Journal of the American Statistical Association*, 58(302):415–434, 1963.

A. Mucherino und O. Seref. Modeling and solving real-life global optimization problems with meta-heuristic methods. *Advances in Modeling Agricultural Systems*, 25:403–419, 2009.

E. Mugambi und A. Hunter. Multi-objective genetic programming optimization of decision trees for classifying medical data. *LNCS - Knowledge-Based Intelligent Information and Engineering Systems*, 2773:293–299, 2003.

D.P. Muni, N.R. Pal und J. Das. A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation*, 8(2):183–196, 2004.

D.P. Muni, N.R. Pal und J. Das. Genetic programming for simultaneous feature selection and classifier design. *IEEE Transactions on Systems, Man und Cybernetics, Part B: Cybernetics*, 36(1):106–117, 2006.

W. Nachtigall. *Bionik als Wissenschaft*. Springer, 2010.

A. Nebro und J. Durillo. On the effect of applying a steady-state selection scheme in the multi-objective genetic algorithm NSGA-II. *Studies in Computational Intelligence - Nature-Inspired Algorithms for Optimisation*, 193:435–456, 2009.

A.E. Nix und M.D. Vose. Modeling genetic algorithms with markov chains. *Annals of Mathematics and Artificial Intelligence*, 5(1):79–88, 1992.

E. Noda, A.A. Freitas und H.S. Lopes. Discovering interesting prediction rules with a genetic algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1322–1329, 1999.

P. Nordin, F. Francone und W. Banzhaf. Explicitly defined introns and destructive crossover in genetic programming. In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, 6–22, 1995.

L.S. Oliveira, R. Sabourin, F. Bortolozzi und C.Y. Suen. A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 17(6):903–929, 2003.

U.M. O'Reilly und F. Oppacher. Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing. *LNCS - Parallel Problem Solving from Nature*, 866:397–406, 1994.

U.M. O'Reilly und F. Oppacher. Hybridized crossover-based search techniques for program discovery. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, 573–578, 1995a.

U.M. O'Reilly und F. Oppacher. The troubling aspects of a building block hypothesis for genetic programming. In *Foundations of Genetic Algorithms 3*, 73–88. Morgan Kaufmann, 1995b.

F. Otero, M. Silva, A. Freitas und J. Nievola. Genetic programming for attribute construction in data mining. *LNCS - Genetic Programming*, 2610:101–121, 2003.

A. Papagelis und D. Kalles. Breeding decision trees using evolutionary techniques. In *Proceedings of the International Conference on Machine Leraning*, 393–400, 2001.

G. Pappa, A. Freitas und C. Kaestner. Attribute selection with a multi-objective genetic algorithm. *LNCS - Advances in Artificial Intelligence*, 2507:743–751, 2002.

R. Parsons und M.E. Johnson. A case study in experimental design applied to genetic algorithms with applications to DNA sequence assembly. *American Journal of Mathematical and Management Sciences*, 17(3):369–396, 1997.

B. Pesch, S. Casjens, I. Stricker, D. Westerwick, D. Taeger, S. Rabstein, T. Wiethege, A. Tannapfel, T. Brüning und G. Johnen. Notch1, hif1a and other cancer-related proteins in lung tissue from uranium miners - variation by occupational exposure and subtype of lung cancer. *PLoS ONE*, 7(9):e45305, 2012.

H. Petersohn. *Data Mining: Verfahren, Prozesse, Anwendungsarchitektur*. Oldenbourg, 2005.

R. Poli. Exact schema theory for genetic programming and variable-length genetic algorithms with one-point crossover. *Genetic Programming and Evolvable Machines*, 2(2):123–163, 2001a.

R. Poli. General schema theory for genetic programming with subtree-swapping crossover. *LNCS - Genetic Programming*, 2038:143–159, 2001b.

R. Poli. A simple but theoretically-motivated method to control bloat in genetic programming. *LNCS - Genetic Programming*, 2610:43–76, 2003.

R. Poli und W.B. Langdon. An experimental analysis of schema creation, propagation and disruption in genetic programming. In *Proceedings of the Internationel Conference on Genetic Algorithms*, 18–25, 1997.

R. Poli und W.B. Langdon. Schema theory for genetic programming with one-point crossover and point mutation. *Evolutionary Computation*, 6(3):231–252, 1998a.

R. Poli und W.B. Langdon. On the search properties of different crossover operators in genetic programming. In *Proceedings of the Annual Conference on Genetic Programming*, 293–301, 1998b.

R. Poli und N.F. McPhee. General schema theory for genetic programming with subtree-swapping crossover: Part ii. *Evolutionary Computation*, 11(2):169–206, 2003.

R. Poli, W.B. Langdon und N.F. McPhee. *A field guide to genetic programming.* Lulu Enterprises, 2008.

W. B. Poli, R. und Langdon. A new schema theory for genetic programming with one-point crossover and point mutation. In *Proceedings of the Annual Conference on Genetic Programming*, 278–285, 1997.

K. Polyak und W.C. Hahn. Roots and stems: stem cells in cancer. *Nature Medicine*, 12(3):296–300, 2006.

J.R. Quinlan. Discovering rules by induction from large collections of examples. *Expert Systems*, 174:168–201, 1979.

J.R. Quinlan. *C4.5: Programs for machine learning.* Morgan Kaufmann, 1993.

R Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, 2013. `http://www.R-project.org`.

N. J. Radcliffe und P. Surry. Formal memetic algorithms. *LNCS - Evolutionary Computing*, 865:1–16, 1994.

N.J. Radcliffe. Schema processing. In *Handbook of evolutionary computation*, B2.5–1 – B2.5–10. Oxford University Press, 1997.

I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Fromman-Holzboog, 1973.

W. Remmele, U. Hildebrand, H.A. Hienz, P.J. Klein, M. Vierbuchen, L.J. Behnken, B. Heicke und E. Scheidt. Comparative histological, histochemical, immunohistochemical and biochemical studies on oestrogen receptors, lectin receptors und barr bodies in human breast cancer. *Virchows Archiv*, 409(2): 127–147, 1986.

H.W. Ressom, R.S. Varghese, S.K. Drake, G.L. Hortin, M. Abdel-Hamid, C.A. Loffredo und R. Goldman. Peak selection from maldi-tof mass spectra using ant colony optimization. *Bioinformatics*, 23(5):619–626, 2007.

A. Reynolds und B. de la Iglesia. Rule induction for classification using multi-objective genetic programming. *LNCS - Evolutionary Multi-Criterion Optimization*, 4403:516–530, 2007.

G. Riddick, H. Song, S. Ahn, J. Walling, D. Borges-Rivera, W. Zhang und H.A. Fine. Predicting in vitro drug sensitivity using Random Forests. *Bioinformatics*, 27(2):220, 2011.

A. Righi, P. Agati, A. Sisto, G. Frank, M. Faustini-Fustini, R. Agati, D. Mazzatenta, A. Farnedi, F. Menetti, G. Marucci und M.P. Foschini. A classification tree approach for pituitary adenomas. *Human Pathology*, 43(10):1627–1637, 2012.

I. Rish. An empirical study of the naive Bayes classifier. In *Proceedings of the Workshop on Empirical Methods in Artificial Intelligence*, 41–46, 2001.

Robert-Koch-Institut. *Krebs in Deutschland 2007/2008*, 2012.

R. Ros und N. Hansen. A simple modification in cma-es achieving linear time and space complexity. *LNCS - Parallel Problem Solving from Nature*, 5199:296–305, 2008.

J.P. Rosca. Analysis of complexity drift in genetic programming. In *Proceedings of the Annual Conference on Genetic Programming*, 286–294, 1997.

F. Rothlauf. *Representations for genetic and evolutionary algorithms*. Springer, 2006.

I. Ruczinski, C. Kooperberg und M. LeBlanc. Logic regression. *Journal of Computational and Graphical Statistics*, 12(3):475–511, 2003.

O. Rudenko und M. Schoenauer. A steady performance stopping criterion for pareto-based evolutionary algorithms. In *Proceedings of the International Multi-Objective Programming and Goal Programming Conference*, 2004.

G. Rudolph. Finite Markov chain results in evolutionary computation: A tour d'horizon. *Fundamenta Informaticae*, 35(1):67–89, 1998.

G. Rudolph und A. Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1010–1016, 2000.

Y. Saeys, I. Inza und P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

H. Sanchez-Faddeev, M. Emmerich, F. Verbeek, A. Henry, S. Grimshaw, H. Spaink, H. van Vlijmen und A. Bender. Using multiobjective optimization and energy minimization to design an isoform-selective ligand of the 14-3-3 protein. *LNCS - Leveraging Applications of Formal Methods, Verification and Validation. Applications and Case Studies*, 7610:12–24, 2012.

M. Sandri und P. Zuccolotto. A bias correction algorithm for the Gini variable importance measure in classification trees. *Journal of Computational and Graphical Statistics*, 17(3):611–628, 2008.

T.J. Santner, B.J. Williams und W. Notz. *The design and analysis of computer experiments*. Springer, 2003.

R. Sarker, K.H. Liang und C. Newton. A new multiobjective evolutionary algorithm. *European Journal of Operational Research*, 140(1):12–23, 2002.

J.D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the International Conference on Genetic Algorithms*, 93–100, 1985.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

H.-P. Schwefel. *Numerische Optimierung von Computermodellen mittels der Evolutionsstrategie*. Birkhäuser, 1977.

H. Schwender und K. Ickstadt. Identification of SNP interactions using logic regression. *Biostatistics*, 9(1):187–198, 2008.

H. Schwender, I. Ruczinski und K. Ickstadt. Testing SNPs and sets of SNPs for importance in association studies. *Biostatistics*, 12(1):18–32, 2011.

C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 1948.

J. Sherrah, R.E. Bogner und B. Bouzerdoum. Automatic selection of features for classification using genetic programming. In *Proceedings of the Australian and New Zealand Conference on Intelligent Information Systems*, 284–287, 1996.

W. Siedlecki und J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10(5):335–347, 1989.

S. Silva und J. Almeida. Dynamic maximum tree depth. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 1776–1787, 2003.

M. Skurichina und R.P.W. Duin. Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis & Applications*, 5(2):121–135, 2002.

T. Soule und J.A. Foster. Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation*, 6(4):293–309, 1998.

H. Soyel, U. Tekguc und H. Demirel. Application of nsga-ii to feature selection for facial expression recognition. *Computers & Electrical Engineering*, 37(6):1232–1240, 2011.

M. Srinivas und L.M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6): 17–26, 1994.

N. Srinivas und K. Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

C. Strobl, A.-L. Boulesteix und T. Augustin. Unbiased split selection for classification trees based on the Gini index. *Computational Statistics & Data Analysis*, 52(1):483–501, 2007a.

C. Strobl, A.-L. Boulesteix, A. Zeileis und T. Hothorn. Bias in Random Forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(25), 2007b.

C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin und A. Zeileis. Conditional variable importance for Random Forest. *BMC Bioinformatics*, 9(307), 2008.

W.A. Tackett. *Recombination, selection und the genetic construction of computer programs*. PhD thesis, University of Southern California, 1994.

T.M. Therneau, B. Atkinson und B. Ripley. *rpart: Recursive Partitioning*, 2012. `http://CRAN.R-project.org/package=rpart`.

H. Trautmann, U. Ligges, J. Mehnen und M. Preuss. A convergence criterion for multiobjective evolutionary algorithms based on systematic statistical testing. *LNCS - Parallel Problem Solving from Nature*, 5199:825–836, 2008.

H. Trautmann, T. Wagner, B. Naujoks, M. Preuss und J. Mehnen. Statistical methods for convergence detection of multi-objective evolutionary algorithms. *Evolutionary Computation*, 17(4):493–509, 2009.

H. Trautmann, D. Steuer und O. Mersmann. *mco: Multi criteria optimization algorithms and related functions*, 2010. `http://CRAN.R-project.org/package=mco`.

A.M. Turing. Intelligent Machinery (1948). In *The Essential Turing*, 395–432. Oxford University Press, 2004.

P. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2: 369–409, 1995.

R. Varshavsky, A. Gottlieb, M. Linial und D. Horn. Novel unsupervised feature filtering of biological data. *Bioinformatics*, 22(14):e507–e513, 2006.

I. Vatolkin, M. Preuß, G. Rudolph, M. Eichhoff und C. Weihs. Multi-objective evolutionary feature selection for instrument recognition in polyphonic audio mixtures. *Soft Computing*, 16(12):2027–2047, 2012.

M. Vendrame, T. Loddenkemper, M. Zarowski, M. Gregas, H. Shuhaiber, D.P. Sarco, A. Morales, M. Nespeca, C. Sharpe, K. Haas, G. Barnes, D. Glaze und S.V. Kothare. Analysis of eeg patterns and genotypes in patients with angelman syndrome. *Epilepsy & Behavior*, 23(3):261–265, 2012.

T. Wagner und H. Trautmann. Online convergence detection for evolutionary multi-objective algorithms revisited. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 1–8, 2010.

T. Wagner, N. Beume und B. Naujoks. Pareto-, aggregation- und indicator-based methods in many-objective optimization. *LNCS - Evolutionary Multi-Criterion Optimization*, 4403:742–756, 2007.

T. Wagner, H. Trautmann und B. Naujoks. Ocd: Online convergence detection for evolutionary multi-objective algorithms based on statistical testing. *LNCS - Evolutionary Multi-Criterion Optimization*, 5467:198–215, 2009.

T. Wagner, H. Trautmann und L. Martí. A taxonomy of online stopping criteria for multi-objective evolutionary algorithms. *LNCS - Evolutionary Multi-Criterion Optimization*, 6576:16–30, 2011.

P.P. Wakabi-Waiswa und V. Baryamureeba. Mining high quality association rules using genetic algorithms. In *Proceedings of the Midwest Artificial Intelligence and Cognitive Science Conference*, 73–78, 2011.

I. Wegener. *Komplexitätstheorie: Grenzen der Effizienz von Algorithmen*. Springer, 2003.

P.A. Whigham. A schema theorem for context-free grammars. In *Proceedings of the IEEE Conference on Evolutionary Computation*, 178–182, 1995.

S.M. Winkler, M. Affenzeller und S. Wagner. Using enhanced genetic programming techniques for evolving classifiers in the context of medical diagnosis. *Genetic Programming and Evolvable Machines*, 10(2):111–140, 2009.

D.H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.

D.H. Wolpert und W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

B. Wyns und L. Boullart. Efficient tree traversal to reduce code growth in tree-based genetic programming. *Journal of Heuristics*, 15(1):77–104, 2009.

H. Xie und M. Zhang. Depth-control strategies for crossover in tree-based genetic programming. *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 15(9):1865–1878, 2011.

H. Xie, M. Zhang und P. Andreae. An analysis of depth of crossover points in tree-based genetic programming. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 4561–4568, 2007.

M. Xiong, X. Fang und J. Zhao. Biomarker identification by feature wrappers. *Genome Research*, 11(11):1878–1887, 2001.

V. Yang, J. und Honavar. Feature subset selection using a genetic algorithm. *IEEE Transactions on Intelligent Systems and Their Applications*, 13(2):44–49, 1998.

L. Yu und H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

M. Zhang, X. Gao, W. Lou und D. Qian. Investigation of brood size in gp with brood recombination crossover for object recognition. *LNCS - Trends in Artificial Intelligence*, 4099:923–928, 2006.

S. Zhang, H.S. Wong, Y. Shen und D. Xie. A new unsupervised feature ranking method for gene expression data based on consensus affinity. *IEEE Transactions on Computational Biology and Bioinformatics*, 9(4):1257–1263, 2012.

H. Zhao. A multi-objective genetic programming approach to developing pareto optimal decision trees. *Decision Support Systems*, 43(3):809–826, 2007.

E. Zitzler und L. Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. *LNCS - Parallel Problem Solving from Nature*, 1498: 292–301, 1998.

E. Zitzler, K. Deb und L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca und V.G. da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.