TECHNISCHE UNIVERSITÄT DARMSTADT

Vergleich verschiedener Methoden zur Behandlung fehlender Attributwerte im SeCo-Regellerner

Studienarbeit

von

Lars Wohlrab

Betreuer: Frederik Janssen

» ABSTRACT

Strategien zur Behandlung unbekannter Attributwerte wurden bereits in zahlreiche Arbeiten thematisiert. Oft zeigt sich darin, dass die angewandten Strategien in der Regel nicht auf allen Datensätzen gleichermaßen gut funktionieren – daher ist es für Lernverfahren von Vorteil, nicht auf die Anwendung einer bestimmten Strategie festgelegt zu sein. Ein im Vergleich zur Behandlung unbekannter Werte noch relativ neues und wenig beackertes Feld ist hingegen der Umgang mit Ungenauigkeit von bekannten numerischen Werten.

Grundlage dieser Arbeit war ein einfacher SeCo-Lerner, für den insgesamt acht Strategien zur Behandlung unbekannter Attributwerte implementiert wurden. In dieser Arbeit werden die einzelnen Strategien und deren Umsetzung beschrieben. Neben wohlbekannten und bereits gut untersuchten Strategien finden sich darunter auch bislang kaum untersuchte Ansätze – insbesondere auch die Adaptierung der von Entscheidungsbäumen bekannten Idee des »reduced information gain«. In dieser Routine kann zudem die angenommene numerische Ungenauigkeit spezifiziert werden. Die Leistungsfähigkeit aller implementierten Strategien wird umfassend evaluiert – sowohl an ausgewählten künstlich präparierten Datensätzen als auch an solchen mit natürlichem Attributausfall. Für die parametrisierbaren Strategien wird jeweils noch der Einfluss der Parameterwahl untersucht. Ferner werden auch die Auswirkungen der einzelnen Strategien auf die gebildeten Modelle diskutiert.

Die erzielten Ergebnisse untermauern die Erkenntnis, dass es keine durchgängig überlegene Strategie gibt – vielmehr zeigt sich ein relativ starker Einfluss des jeweiligen Datensatzes. Lediglich das komplette Verwerfen der Beispiele mit unbekannten Werten kann als eindeutig unterlegen identifiziert werden. Auf Datensätzen mit nicht-geringen Anteilen an unbekannten Werten erscheint eine sorgfältige Auswahl der eingesetzten Strategie daher als unerlässlich. Obschon die Untersuchungen zur numerischen Unschärfe nicht im Zentrum dieser Arbeit stehen, kann doch zumindest gezeigt werden, dass sich hierin selbst ohne domänenspezifische Kenntnisse ein nicht zu unterschätzendes Optimierungspotential verbirgt.

» Inhalt

» Tabellen

» ABBILDUNGEN

1 EINLEITUNG

Der Umgang mit unbekannten Attributwerten ist ohne Zweifel ein wichtiger Aspekt im Rahmen des induktiven Lernens. Bei nahezu jeder praktischen Anwendung maschinellen Lernens muss damit gerechnet werden, dass einzelne Angaben fehlen. Sei es, weil die Daten gar nicht erst erhoben wurden, sie verloren gingen oder absichtlich wieder entfernt wurden oder weil eine Angabe für gewisse Beispiele keinen Sinn ergibt. Aufgrund dieser praktischen Relevanz ist der Umgang mit unbekannten Werten kein neuer Untersuchungsgegenstand. So wurden bereits eine Reihe verschiedener Strategien für die Behandlung entwickelt und getestet – sowohl für Entscheidungsbäume ([Qui-89], [STP-07]) als auch für Regellerner [BF-96]. Im Allgemeinen lassen sich dabei vier mögliche Herangehensweisen identifizieren:

(1) Grundsätzlich keine Beispiele mit unbekannten Werten akzeptieren – der einfachste Ansatz, der sich in aller Regel grundsätzlich nur während der Lernphase und in der Praxis zudem nur bei einem sehr geringem Anteil an unbekannten Werten anwenden lässt.

(2) Keine Attribute mit unbekannten Werten testen – beschränkt man sich auf das Lernen genau eines Modells, praktisch nur anwendbar, wenn nur wenige, möglichst unwichtige Attribute betroffen sind. Lässt man die Beschränkung auf ein Modell fallen, ergibt sich der von Saar-Tsechansky & Provost beschriebene »reduced-model«-Ansatz [STP-07]. Durch die Beschränkung der zulässigen Anzahl zu lernender Modelle ist ein Trade-Off zwischen Aufwand und Genauigkeit möglich. Ein solcher Ansatz wurde im Rahmen dieser Arbeit jedoch (und als einziger) nicht verfolgt.

(3) Unbekannte Werte (durch »reguläre«) ersetzen – das lässt sich häufig als Schätzung des wahren Attributwerts interpretieren. Sowie

(4) Direkte Zuweisung einer Semantik an unbekannte Werte, also Festlegung expliziter Regeln für den Umgang mit unbekannten Werten in allen möglichen Situationen. Eine strikte Trennung von Ansatz (3) ist dabei nicht immer möglich, zumindest lassen sich bestimmte Verhaltensweisen auf beiderlei Arten interpretieren/implementieren.

Bruha & Franek beschrieben und untersuchten in [BF-96] fünf grundlegende Behandlungsstrategien (der obigen Einteilung nach aus den Kategorien 1, 3 und 4) für den CN2/CN4-Lerner – diese bildeten auch den Grundstock für diese Arbeit. An Stelle von CN4 wurde in dieser Arbeit jedoch der SeCo-Lerner eingesetzt (der im folgenden Kapitel näher vorgestellt wird).

Dabei wurden im Rahmen dieser Arbeit hauptsächlich zwei Ziele verfolgt. Zum einen sollte der SeCo-Lerner, der bislang nicht explizit mit unbekannten Werte umgehen konnte, um verschiedene Routinen zur Behandlung ebenjener erweitert werden. In einem zweiten Schritt ging es darum, die einzelnen Routinen empirisch umfassend zu evaluieren und so ihre Leistungsfähigkeit zu vergleichen. Zudem wurde untersucht, inwiefern und in welcher Weise sich die Anwendung der einzelnen Strategien auf die gelernten Modelle auswirkt.

Ein besonderes Augenmerk galt dabei dem in [GLF-08] vorgeschlagenen Ansatz zur Übertragung des von der TDIDT bekannten Prinzips des »reduced information-gain«, also der »Bestrafung« der heuristischen Evaluierungsfunktion für unbekannte Werte des getesteten Attributs, auf Regellerner. Dieser Ansatz wurde hier erstmals im Rahmen des Separate-and-con-

quer-Paradigmas untersucht. Darüber hinaus wird in besagtem Paper auch eine Methode für die Berücksichtigung von Ungenauigkeit bei numerischen Attributen beschrieben. Daraus ergibt sich eine gänzlich neue Möglichkeit, eine zu genaue Anpassung an die Trainingsdaten und insbesondere auch das Treffen anti-intuitiver Vorhersagen aufgrund der Unterteilung entlang harter numerischer Grenzen zu verhindern. Zusätzlich zu den verschiedenen Behandlungsstrategien für unbekannte Werte wurde auch diese Fähigkeit für den SeCo-Lerner implementiert und erste empirische Untersuchungen durchgeführt.

Die Arbeit ist dabei wie folgt strukturiert: Zunächst wird der verwendete SeCo-Lerner kurz vorgestellt und dessen im Rahmen dieser Arbeit verwendete Konfiguration beschrieben (Kapitel 2). Anschließend wird auf die für die empirische Evaluation verwendeten Datensätze eingegangen (Kapitel 3). Daran schließt sich in Kapitel 4 die Vorstellung der einzelnen implementierten und untersuchten Routinen für die Behandlung unbekannter Attributwerte an. Schließlich werden die mit den einzelnen Routinen auf den Datensätzen erzielten Ergebnisse präsentiert (Kapitel 5) und eingeordnet (Kapitel 6).

2 Der SeCo-Regellerner

Für die im Rahmen dieser Arbeit durchgeführten Untersuchungen wurde die Implementierung eines Separate-and-Conquer-Lerners [Für-99] eingesetzt und erweitert. Sein Vorgehen entspricht dabei dem des unten in Abbildung 1 dargestellten generischen Pseudocodes.

Es liegt auf der Hand, dass die gewählte Konfiguration des Lerners, also unter anderem die eingesetzte Suchstrategie und Bewertungsheuristik, das letztlich gelernte Modell entscheidend beeinflussen. Die Untersuchung der Vor- und Nachteile verschiedener Kombinationen von Suchstrategie und Evaluierungsheuristik und ihres etwaigen Einfluss' auf die Performanz der einzelnen Behandlungsstrategien für fehlende Attributwerte liegt jedoch klarerweise außerhalb des Umfangs dieser Arbeit. Stattdessen wurde hauptsächlich mit nur einer festen Konfiguration gearbeitet. Zu Kontrollzwecken wurde lediglich auch einmal eine alternative Evaluierungsheuristik verwendet. Die im Rahmen der hiesigen Untersuchungen eingesetzten konkreten Implementierungen der einzelnen Subroutinen werden im folgenden beschrieben.

- **InitializeRule/RefineRule**

  Der Regelraum wurde jeweils »top-down« durchsucht, es wurde also mit der allgemeinsten Regel angefangen und diese dann sukzessive spezialisiert. Da die Menge der möglichen Verfeinerungen eines Regelkandidaten unter Umständen von der MAVHPolicy beeinflusst werden kann, muss diese RefineRule als zusätzlicher Parameter übergeben werden.

- **SelectCandidates/FilterRules**

  Für die Realisierung dieser beiden Subroutinen wurde ein »Beam« unterhalten, eine Liste mit den (in diesem Falle) 5 momentan am besten bewerteten Regeln. Also filterte FilterRules die 5 besten Regeln aus der Liste der Regelkandidaten, während SelectCandidates einfach die eingegebene Liste unverändert zurücklieferte.

- **EvaluateRule/Cover**

  Als primäre Bewertungsheuristik für Regelkandidaten wurde das *Laplace*-Maß eingesetzt. Zusätzlich wurde zur Überprüfung der erzielten Ergebnisse auf einigen Datensätzen auch noch die *m-estimate*-Heuristik verwendet.

  - *Laplace:* $L(rule) = \frac{p+1}{p+n+2}$
  - *m-estimate:* $M(rule) = \frac{p+m\frac{P}{P+N}}{p+n+m}$ (mit Parameter m=22,466)
  - p/n ... Anzahl der als abgedeckt gewerteten positiven/negativen Beispiele
    P/N ... Gesamtzahl der positiven/negativen Beispiele

  Jedoch kann bei der heuristischen Bewertung von Regeln neben der Bewertungsfunktion auch der vorgegebene Umgang mit unbekannten Attributwerten eine Rolle

spielen, sofern die Werte für *p* und *n* dadurch beeinflusst werden. Dies ist auch eng verknüpft mit der Frage, welche Beispiele von einer Regel letzten Endes abgedeckt werden. Daher müssen sowohl Cover- als auch EvaluateRule um den entsprechenden MAVHPolicy-Parameter erweitert werden.

- **RuleStoppingCriterion**

  Das Lernen der Theorie konnte auf zwei Arten vorzeitig – also bevor alle positiven Beispiele abgedeckt waren – beendet werden. Zum einen, wenn der Aufruf von FindBestRule eine Regel ohne Bedingungen zurücklieferte. Zum anderen wurde eine Heuristik angewandt, die den Lernvorgang abbrach, wenn die neue beste Regel nicht mehr positive als negative Beispiele abdeckte.

- **StoppingCriterion**

  Auch die Verfeinerung eines Regelkandidaten konnte vorzeitig abgebrochen werden. Hierbei kamen jedoch ausschließlich konservative Verfahren (und keine Heuristiken) zum Einsatz. So wurden Kandidaten aussortiert, die keine Beispiele abdeckten, die zweimal auf dasselbe nominelle Attribut testeten oder wenn der durch weitere Verfeinerungen der Regel maximal noch erreichbare Wert der Evaluierungsheuristik geringer war als derjenige des momentan besten Kandidaten (konservatives Forward-Pruning).

- **PostProcess**

  Ein Post-Pruning wurde nicht durchgeführt – die PostProcess-Routine lieferte also lediglich die eingegebene Theorie unverändert zurück. Das Fehlen der Möglichkeit zu einer adäquaten Nachbearbeitung der gelernten Theorie führt, zusammen mit den auch sonst allenfalls als rudimentär zu bezeichnenden Maßnahmen gegen Overfitting, diesbezüglich zu einer gewissen Anfälligkeit.

```
procedure SeparateAndConquer(Examples, MAVHPolicy)
   Examples = PrepareExamples(Examples, MAVHPolicy)          ← Preparation-Phase
   Theory = Ø
   while (Positive(Examples) ≠ Ø)
      Rule = FindBestRule(Examples, MAVHPolicy)
      Covered = Cover(Rule, Examples, MAVHPolicy)            ← Separation-Phase
      if (RuleStoppingCriterion(Theory, Rule, Examples))
         exit while
      Examples = Examples\Covered
      Theory = Theory U Rule
   Theory = PostProcess(Theory)
return(Theory)

                                  ...

procedure FindBestRule(Examples, MAVHPolicy)
   InitRule = InitializeRule(Examples)
   InitVal = EvaluateRule(InitRule)
   BestRule = <InitVal, InitRule>
   Rules = {BestRule}
   while (Rules ≠ Ø)
      Candidates = SelectCandidates(Rules, Examples)
      Rules = Rules\Candidates
      for (Candidate ∈ Candidates)
         Refinements = RefineRule(Candidate, Examples, MAVHPolicy)
         for (Refinement ∈ Refinements)
            Evaluation = EvaluateRule(Refinement, Examples, MAVHPolicy)  ← Evaluation-Phase
            unless(StoppingCriterion(Refinement, Evaluation, Examples))
               NewRule = <Evaluation, Refinement>
               Rules = InsertSort(NewRule, Rules)
               if (NewRule > BestRule)
                  BestRule = NewRule
      Rules = FilterRules(Rules, Examples)
return(BestRule)
```

Abbildung 1: *generischer Pseudocode des Separate&Conquer-Lerners*

Darin gesondert gekennzeichnet sind die gegenüber der Ausgangsvariante ([Für-99]) vorgenommene Erweiterung für den Umgang mit unbekannten Attributwerten (MAVHPolicy – missing-attribute-value-handling). Ferner sind die drei für die Beschreibung des Verhaltens der Behandlungsroutinen während der Modellbildung relevanten Phasen lokalisiert (Kapitel 4: Behandlung unbekannter Attributwerte).

3 Datensätze

Für die durchgeführten Untersuchungen stand eine Reihe von bekannten UCI-Datensätzen zur Verfügung. Diese Datensätze teilten sich zu etwa gleichen Teilen auf in solche mit fehlenden Attributwerten und solche ohne. Für die durchzuführende Untersuchung von Strategien für den Umgang mit fehlenden Werten sind naturgemäß nur Datensätze mit fehlenden Attributwerten geeignet. Jedoch müssen auch die Datensätze ohne fehlende Attributwerte nicht notwendigerweise ungenutzt bleiben. Im Folgenden finden sich daher nähere Informationen sowohl zu den Datensätzen mit fehlenden Werten und ihrer Kategorisierung als auch zu den Datensätzen ohne fehlende Werte und ihrer Aufbereitung. Details zu den einzelnen Datensätzen sind dabei in den Appendix ausgelagert.

3.1 Datensätze mit fehlenden Attributwerten

Von diesen »von Natur aus« geeigneten Datensätzen standen insgesamt 24 für die Untersuchungen zur Verfügung. Der Umstand, dass in mindestens einem Beispiel eines Datensatzes mindestens ein Attributwert unbekannt ist, sagt alleine jedoch ungefähr nichts darüber aus, welche Rolle die unbekannten Werte (bzw. ihre Handhabung) im Rahmen der Modellbildung tatsächlich spielen. Um den potenziellen Einfluss der MAVH-Policy auf die gelernten Modelle a priori zumindest grob abschätzen zu können, wurde behelfsweise eine spezifische Kennzahl eingeführt – der so genannte MAV-Score. Je höher der Score, umso größer sollte der Einfluss der unbekannten Werte auf die gelernten Modelle sein. An der unteren Schranke des möglichen Einflussspektrums haben Datensätze ohne unbekannte Werte dabei einen Score von 0.

Um dies zu gewährleisten, fließen in den MAV-Score eines Datensatzes für jedes Attribut sowohl der jeweilige Anteil an unbekannten Werten als auch die mittels χ^2-Evaluierung ermittelte Unterscheidungskraft bezüglich der Klassenzugehörigkeit ein. (Der genaue Berechnungsalgorithmus ist im Appendix [A] beschrieben.) Dieser Ansatz beruht dabei auf den folgenden beiden grundsätzlichen Überlegungen: Der Einfluss der angewandten Behandlungsstrategie auf das gelernte Modell steigt mit dem Anteil der von der Behandlung betroffenen Beispiele – denn nur im Umgang mit diesen Beispielen können sich die Strategien überhaupt unterscheiden. Bei Attributen ohne unbekannte Werte ist die Behandlungsstrategie für ebensolche offenkundig ohne jede Relevanz. Damit sich der potentielle Einfluss auftretender unbekannter Werte auch praktisch auswirken kann, muss das entsprechende Attribut aber auch tatsächlich getestet werden. Unter der heuristischen Annahme, dass die Wahrscheinlichkeit, mit der ein Test auf ein bestimmtes Attribut gelernt wird, mit dessen Fähigkeit korreliert, die Klassen zu unterscheiden, kann die χ^2-Bewertung eines Attributs als grober Maßstab für dessen Wichtigkeit verwendet werden.

Anhand ihres MAV-Scores werden die Datensätze in drei Kategorien unterteilt – in solche mit geringem, mittlerem oder hohem Score. Die Liste der Datensätze mitsamt ihrem MAV-Score und der zugehörigen Kategorie findet sich im Appendix [B].

3.2 Datensätze ohne fehlende Attributwerte

Die Datensätze ohne fehlende Attributwerte eignen sich in ihrer ursprünglichen Form nicht für die im Rahmen dieser Arbeit durchzuführenden Untersuchungen. Man kann sie jedoch »nutzbar« machen, indem man künstlich einen Teil der Attributwerte entfernt (im Folgenden ist in diesem Kontext auch vom »Ausdünnen« von Attributen die Rede).

Die künstliche Erzeugung geeigneter Datensätze zieht jedoch eine Reihe von Konsequenzen nach sich. Einerseits übt die Anwendung eines bestimmten Erzeugungsalgorithmus' immer auch einen systematischen Einfluss auf die Datensatzcharakteristik aus, ist also nicht neutral und wird demzufolge einige Behandlungsstrategien tendenziell begünstigen, andere hingegen benachteiligen. Andererseits ermöglicht die Erzeugung der Datensätze unter »Laborbedingungen« die genaue Kontrolle sowohl über die betroffenen Attribute als auch über den jeweiligen Anteil an fehlenden Werten. Insbesondere existiert für alle generierten Datensätze stets das Vergleichsmodell, das ohne fehlende Werte gelernt wurde, sodass sich der Einfluss der fehlenden Werte auf Modellbildung und Genauigkeit explizit bestimmen lässt. Zur Entfernung der Attribute wurde im Rahmen dieser Arbeit dabei das folgende Verfahren angewendet:

- **Auswahl der Attribute**

  Damit die Behandlungsstrategien für fehlende Werte auch tatsächlich einen nicht zu vernachlässigenden Einfluss auf die Modellbildung haben, sollten möglichst relevante Attribute ausgewählt werden. Als Beurteilungsgrundlage wurde dabei das Ergebnis der χ^2-Evaluierung (gemittelt über 10 Partitionen) verwendet und die drei nach diesem Maßstab wichtigsten Attribute gewählt – diese werden im Folgenden auch zusammenfassend als KO-Attribute bezeichnet.

- **Entfernen der Attribute**

  Jedes der ausgewählten Attribute wurde zu gewissen Prozentsätzen »ausgeschaltet« – in Stufen von 15% bis hin zu maximal 90%. Die Auswahl der betroffenen Beispiele erfolgte dabei für jedes Attribut unabhängig und zufällig.

- **Ergebnis**

  Aus einem Datensatz ohne fehlende Attributwerte hat man so sechs Datensätze generiert, mit Ausfallraten von 15% bis 90% bei drei ausgewählten Attributen.

Durch die (rein) zufällige Auswahl der manipulierten Beispiele legt man bei der Erzeugung der Datensätze explizit ein MCAR-Szenario (missing completely at random) zugrunde. Diese Annahme über die Natur fehlender Attributwerte ist zwar im Bereich des maschinellen Lernens durchaus üblich, in der Praxis jedoch keinesfalls immer gerechtfertigt. Insofern ist eine gewisse Vorsicht angebracht, die auf solchen Datensätzen erzielten Ergebnisse auf reale Datensätze zu übertragen. Daher wird dieses Verfahren zur Erzeugung von Datensätzen mit fehlenden Attributwerten im Rahmen dieser Arbeit auch lediglich exemplarisch bei drei Datensätzen zum Einsatz gebracht. Um statistisch möglichst belastbare Aussagen zu erhalten, wurden hierfür die drei größten der verfügbaren Datensätze herangezogen: credit-g, krkp und segment. Auch hier sind Details zu den Datensätzen und den ausgewählten KO-Attributen im Appendix [C] angeführt.

4 Behandlung unbekannter Attributwerte

Das Fehlen von Attributwerten kann grundsätzlich vielfältige Ursachen haben – und entsprechend vielfältig kann auch die Semantik eines unbekannten Wertes sein. Bruha und Franek etwa unterschieden insgesamt vier verschiedenen Quellen für unbekannte Werte [BF-96]. So muss nicht für jeden unbekannten Wert zwangsläufig ein »wahrer« Wert existieren, der sich sinnvoll schätzen ließe. Insbesondere stellt ein unbekannter Wert nicht in jedem Fall einen Verlust an Information dar. Wenn das Fehlen eines Werts auf die Entscheidung eines Domänenexperten zurückgeht, kann sich darin stattdessen sogar implizites Wissen über das zu lernende Konzept manifestieren. (Im Gegensatz hierzu haben in den automatisch generierten Datensätzen aufgrund ihres Konstruktionsprinzips alle unbekannten Werte immer die selbe Semantik.)

Während der CN2-Lerner [CN-89] zumindest zwei Arten unbekannter Werte unterscheidet, kennt der hier verwendete SeCo-Lerner nur eine Kategorie. Daraus ergibt sich die inhärente Notwendigkeit, alle Arten unbekannter Werte ohne Berücksichtigung ihrer Semantik gleich zu behandeln. Folglich kann auch keine im SeCo-Lerner eingesetzte Routine, die von einer bestimmten Semantik der fehlenden Werte ausgeht, immer von der richtigen Semantik ausgehen. (inwiefern sich die Wahl der »richtigen« Semantik positiv auf die erzielte Genauigkeit auswirkt, soll hier allerdings nicht Gegenstand einer näheren Untersuchung sein)

Zur Beschreibung des Umgangs mit unbekannten Attributwerten muss differenziert werden zwischen den einzelnen Phasen des Lernvorgangs. Gemeinhin wird hierfür das Verhalten des Lerners in drei Situationen unterschieden [Qui-89], [BF-96]:

- Evaluation (Eval): Bewertung (und damit Auswahl) von Kandidatenregeln
- Separation (Sep): Abtrennung der von der ausgewählten Regel (nicht) abgedeckten Beispiele
- Classification (Cls): Klassifikation eines unbekannten Beispiels

Für die Beschreibung des praktischen Vorgehens ist es jedoch oftmals hilfreich, zusätzlich noch eine vierte Phase zu definieren: die Vorverarbeitung/Preparation der Beispiele vor dem Lernen (Pre-L) bzw. dem Klassifizieren (Pre-C). Die für die Modellbildung relevanten Phasen sind auch im Pseudo-Code des SeCo-Lerners (Abbildung 1, Seite 13) lokalisiert.

Die verfügbaren Optionen für die Handhabung unbekannter Werte hängen dabei offenkundig von der jeweiligen Phase ab – so ist beispielsweise die Klasseninformation nur während der Lernphase verfügbar. Prinzipiell sind die einzelnen Phasen bezüglich des Umgangs mit unbekannten Werten voneinander unabhängig – in [Qui-89] werden die Strategien für die einzelnen Phasen auch getrennt definiert. Zugleich liegt auf der Hand, dass nicht alle möglichen Kombinationen sinnvoll sind, sondern das Verhalten in den einzelnen Phasen vielmehr aufeinander abgestimmt sein sollte. Ein solches abgestimmtes Gesamtkonzept für den Umgang mit unbekannten Werten sei im Folgenden als

(Behandlungs-)Routine bezeichnet. Im Rahmen dieser Arbeit wurden für den SeCo-Lerner insgesamt acht solcher Routinen definiert, implementiert und evaluiert. Diese werden in den folgenden Abschnitten en Detail beschrieben.

4.1 DELETE

Der einfachst denkbare Ansatz für den Umgang mit fehlenden Werten ist es, alle Beispiele mit fehlenden Attributwerten schon im Vorfeld (genauer: in der PRE-L-Phase) aus der Trainingsmenge zu entfernen. Da in allen fürs Lernen verbleibenden Beispielen sämtliche Attributwerte bekannt sind, können in der EVAL- und der SEP-Phase keine fehlenden Attributwerte mehr auftreten. Beim Klassifizieren eines Beispiels mit fehlenden Attributwerten käme ein Entfernen des Beispiels in der PRE-C-Phase der Verweigerung einer Klassenvorhersage gleich und kommt also in der Regel nicht in Betracht. Für die CLS-Phase muss daher notwendigerweise ein konstruktives Verhalten spezifiziert werden. Beispiele mit fehlenden Werten werden ignoriert – fehlende Werte können also niemals von einer Bedingung abgedeckt werden. Ein Beispiel kann folglich nur von solch einer Regel abgedeckt werden, die keinen Test auf ein fehlendes Attribut enthält.

Es liegt auf der Hand, dass dieses Vorgehen nur sehr bedingt für den praktischen Einsatz geeignet ist, da die in den Trainingsbeispielen mit unbekannten Werten enthaltene Information über das zu lernende Konzept in jedem Fall verworfen wird – selbst dann, wenn das vom Attributausfall betroffene Attribut selber gar keine klassenrelevante Information enthält. Daher kann die DELETE-Routine im Grunde genommen nur als schlechtes Beispiel dienen und zu Vergleichszwecken eine untere Schranke für die mit dem SeCo-Lerner erzielbare Genauigkeit aufstellen.

4.2 IGNORE

Der entscheidende Schwachpunkt der DELETE-Routine ist die schlechte Ausnutzung der in den Trainingsbeispielen enthaltenen Klasseninformation. Beim Auftreten von unbekannten Werten wird das gesamte Beispiel verworfen – die in den restlichen Attributen enthaltene Information kann nicht verwertet werden. Einen der einfachsten Wege, dieses Problem zu beheben und die gesamte bekannte Information zum Lernen zu verwenden, stellt die IGNORE-Routine dar. Beispiele mit unbekannten Werten werden dabei nicht verändert. sondern diese gehandhabt wie Attributwerte, die von keiner Bedingung abgedeckt werden können. Unbekannte Werte werden beim Evaluieren und Anwenden (sowohl in der SEP- als auch in der CLS-Phase) der betreffenden Regeln also gewissermaßen »ignoriert«.

Anders als bei der TDIDT geht durch diese Art des Ignorierens keine Information aus den Trainingsbeispielen verloren – zum Lernen der folgenden Regeln werden die vormals ignorierten Beispiele wieder herangezogen, genauso wie alle noch nicht abgedeckten Beispiele. Auch wenn diese Routine auf den ersten Blick ausgesprochen simpel erscheinen mag, so erfüllt sie doch die wesentliche Anforderung, alle bekannten Attributwerte für das Lernen nutzen zu können.

4.3 AnyValue

Bei Anwendung der Ignore-Routine können unbekannte Werte niemals abgedeckt werden. Ein solches Vorgehen ist gewissermaßen »pessimistisch«: im Zweifelsfall wird immer gegen die Abdeckung eines Beispiels entschieden. Grundsätzlich spricht jedoch auch nichts gegen einen »optimistischen« Ansatz – Beispiele im Zweifelsfall (sprich: bei Tests auf Attribute mit unbekanntem Wert) also immer als abgedeckt zu werten. Die entsprechende Routine heißt AnyValue. Abgesehen von der gegenteiligen Weltsicht ist das Vorgehen identisch zu dem der Ignore-Routine.

So lange man nicht annimmt, dass zu jedem unbekannten Attributwert ein »wahrer« Wert existiert, gibt es kein schlüssiges Argument, warum die eine oder die andere der beiden Grundannahmen im Allgemeinen »richtiger« sein sollte. Das bedeutet allerdings nicht zwangsläufig, dass die beiden Routinen auch in der Praxis (im Mittel) gleich gut funktionieren. Tatsächlich könnte die AnyValue-Routine hier aufgrund der geringeren Selektivität der Bedingungen systematisch benachteiligt sein. Zum einen wird hierdurch generell das Lernen aus Attributen mit nennenswertem Anteil an unbekannten Werten erschwert; zum anderen werden, sollte doch ein Test auf ein solches Attribut gelernt werden, tendenziell mehr Bedingungen für die Regel benötigt, wodurch AnyValue anfälliger für Overfitting wäre – insbesondere, da im SeCo-Lerner keine Anti-Overfitting-Maßnahmen implementiert sind.

4.4 Common

Sowohl bei Ignore als auch bei AnyValue handelt es sich um sehr einfache Strategien, die fehlende Werte entweder niemals oder immer als abgedeckt werten. Insofern ist es nahe liegend, einen möglichst intelligenten Mittelweg zwischen diesen beiden Extremen beschreiten zu wollen. Will man nicht bloß blind raten, hat man im Wesentlichen zwei Möglichkeiten – entweder berücksichtigt man die anderen Attributwerte des Beispiels oder die Attributwerte anderer Beispiele. Grundsätzlich gilt: ersetzt man einen unbekannten Wert auf irgendeine Weise durch einen realen Wert, setzt man implizit immer die Existenz eines »wahren« Wertes voraus, den es zu approximieren gilt.

Die wahrscheinlich einfachste Möglichkeit für eine solche Einsetzung von Werten besteht darin, unbekannte Werte immer durch den häufigsten (*mode*, nominelle Attribute) bzw. den arithmetischen Mittelwert (*mean*, numerische Attribute), den so genannten *common value*, zu ersetzen. Theoretisch gerechtfertigt ist dieser Ansatz durch die einfache statistische Überlegung, dass (unter Annahme der Existenz eines wahren Wertes) der *common value* der Schätzer mit dem kleinsten Quadratmittelfehler ist.

Dieses Vorgehen hat je nach Attributtyp leicht unterschiedliche Seiteneffekte. Für nominelle Attribute werden alle Beispiele mit unbekanntem Wert unter dem ohnehin häufigsten Wert gesammelt, wodurch sich die oben bei AnyValue geschilderten Effekte auf diesen einen Wert beschränken. Insbesondere sind nicht die Tests auf die selteneren Werte betroffen, bei denen die Verzerrung der Evaluierungsfunktion naturgemäß besonders ausgeprägt ist. Zudem ist die vorgenommene Ersetzung gemäß Konstruktion des *common value* wahrscheinlich meist »richtig«. Für numerische Attribute sind die Auswirkungen in der Theorie hingegen nicht so schön. Zumindest beim ersten Test auf

das Attribut muss die Hälfte der generierten Features die unbekannten Werte abdecken. Auch ist der Mittelwert für das einzelne Beispiel nicht unbedingt eine brauchbare Approximation des wahren Werts.

Da die COMMON-Routine unbekannte Attributwerte bereits im Rahmen der Vorverarbeitung (PRE-L & -C) nach dem obig beschriebenen Mechanismus ersetzt, muss in den restlichen Phasen nicht mit fehlenden Werten umgegangen werden.

4.5 NEAREST-NEIGHBOR

Die COMMON-Routine ersetzt unbekannte Werte eines Attributs in allen Beispielen mit dem selben Wert, der unter Verwendung sämtlicher verfügbarer Trainingsbeispiele ermittelt wird. Ein tatsächlicher Bezug dieses Einheitswertes zu den einzelnen Beispielen mit unbekannten Werten – und damit auch zu deren angenommenen wahren Werten – besteht dabei jedoch nicht. Der wahre Attributwert eines konkreten Beispiels lässt sich auf diese Weise daher nur schwerlich abschätzen.

Man kann nun versuchen, die Schätzer für die unbekannten Werte zu verbessern, indem man den eingesetzten Wert an das jeweilige Beispiel anpasst. Eine Möglichkeit, dies zu erreichen besteht in einer intelligenteren Auswahl der für die Schätzung herangezogenen Beispiele. Konkret werden im Falle der NN-Routine statt der gesamten Trainingsmenge nur die jeweils K ähnlichsten Beispiele verwendet. Die Idee hinter diesem Vorgehen ist, dass die zu einem Beispiel ähnlichsten Beispiele für dieses repräsentativer sind als die Gesamtmenge und daher realistischere Schätzungen für die unbekannten Attributwerte liefern können. Zur Bestimmung der »ähnlichsten« Beispiele kann prinzipiell ein beliebiger NN-Suchalgorithmus verwendet werden. Die von diesem verwendete Abstandsfunktion legt dann auch fest, was Ähnlichkeit von Beispielen konkret bedeutet. Dabei ist evident, dass die NN-Routine in der Praxis nur dann (gut) funktionieren kann, wenn der von der Abstandsfunkton implizierte Ähnlichkeitsbegriff für die jeweilige Domäne Sinn ergibt.

Strukturell entspricht das Vorgehen der NN-Routine dem von COMMON. Letzteres lässt sich theoretisch sogar als Spezialfall von NN mit Parameter $K=\infty$ interpretieren. Demzufolge wird angestrebt, alle unbekannten Attributwerte bereits in den PRE-Phasen zu ersetzen. Im Gegensatz zur COMMON-Routine kann jedoch die Situation eintreten, dass das gesuchte Attribut in allen betrachteten Beispielen unbekannt ist. Eine Ersetzung des unbekannten Werts ist in diesem Fall faktisch nicht möglich, weshalb die NN-Routine auch in den nachgelagerten Phasen unbekannte Attributwerte handhaben können muss. Praktisch wurde dieses Problem dadurch gelöst, dass NN in diesen Fällen auf IGNORE als »Rückfallroutine« zurückgreift. (Ebenso gut hätte man hier beispielsweise COMMON einsetzen können.)

Die Alternative hierzu wäre gewesen, stattdessen die NN-Suche so zu modifizieren, dass die betrachtete Nachbarschaft bei Bedarf dynamisch vergrößert wird um zu garantieren, dass der eingesetzte Wert immer auf Grundlage von K bekannten Werten ermittelt wird. Diese Idee wurde hier jedoch aus zweierlei Gründen verworfen. Aus praktischer Sicht spricht dagegen, dass dieser Ansatz Modifikationen am NN-Suchalgorithmus erfordert und daher den Einsatz von Standardalgorithmen verhindert. Aus theoretischer

Sicht ist einzuwenden, dass man dadurch letztlich die Grundidee der NN-Routine aufweichen würde: dass nur die ähnlichsten Beispiele Einfluss haben sollen auf den eingesetzten Wert. Denn wenn alle Beispiele in der Nachbarschaft für ein bestimmtes Attribut unbekannte Werte aufweisen, warum sollte dies nicht als Indiz dafür gewertet werden, dass der NN-Ansatz in diesem Fall keine sinnvolle Schätzung liefern kann.

KONFIGURATION

Im Rahmen der hier durchgeführten Untersuchungen wurde als Abstandsfunktion durchgängig die LINEARNNSEARCH aus der Weka-Bibliothek eingesetzt. Für den Parameter K wurden Werte zwischen 3 und 15 gewählt – der jeweils verwendete Wert von K wird dabei durch die Bezeichnung NN.[K] zum Ausdruck gebracht (also zum Beispiel NN.3 für K=3).

4.6 VERTEILUNGSBASIERTE ERSETZUNG (DBI)

Die in der COMMON-Routine betriebene Ersetzung aller unbekannten Werte eines Attributs durch ein und denselben Wert ist insbesondere dann problematisch, wenn die verschiedenen (nominellen) Werte mit nahezu gleicher Häufigkeit auftreten. In diesem Fall wird der erwartete mittlere Ersetzungsfehler sehr groß – der »wahre« Wert des Attributs wird durch den eingesetzten nur sehr schlecht approximiert, die natürliche Verteilung der Attributwerte wird daher stark verzerrt.

Derlei Artefakte lassen sich vermeiden, indem man bei der Ersetzung der unbekannten Werte die ursprüngliche Verteilung berücksichtigt. Anstatt unbekannte Werte deterministisch durch einen bestimmten Attributwert zu ersetzen, werden alle möglichen Attributwerte eingesetzt, wobei das Gewicht des Beispiels proportional zu den Wahrscheinlichkeiten aufgeteilt wird – ein Beispiel mit unbekannten Werten wird so durch eine Wahrscheinlichkeitswolke aller möglichen Varianten ersetzt. Eine solche verteilungsbasierte Ersetzung der unbekannten Attributwerte, englisch: »Distribution-based imputation« (DBI), wurde beispielsweise bereits in [Qui-89] im Rahmen der TDIDT verwendet. Es lässt sich jedoch ebenso gut auf Regellernsysteme anwenden [BF-96]. Im Folgenden sind die Details zur Konfiguration und Implementierung der Routine in den einzelnen Phasen beschrieben.

LERNPHASE

Bei der Handhabung unbekannter Attributwerte wird unterschieden zwischen nominellen Attributen auf der einen und numerischen Attributen auf der anderen Seite. In nominellen Attributen können unbekannte Werte bereits im Rahmen der PRE-Phase ersetzt werden. Aufgrund der diskreten Wertigkeit kann ein unbekannter Attributwert gemäß der DBI-Philosophie a priori durch alle möglichen Werte ersetzt werden, wobei als Wahrscheinlichkeiten die relativen Häufigkeiten der bekannten Werte in der Trainingsmenge verwendet werden. Dieses Vorgehen kann jedoch, insbesondere wenn mehrere (nominelle) Attribute eines Beispiels unbekannte Werte aufweisen, aufgrund seiner exponentiellen Dynamik zu einer sehr starken Aufsplittung eines Beispiels führen und so die Menge der Trainingsbeispiele »explodieren« lassen. Von daher ist es für die praktische Anwendung unumgänglich, die Fragmentierung der Beispiele zu begrenzen. Die Implementierung der DBI-Routine im Rahmen des SeCo-Lerners verwendet hierfür die

Wahrscheinlichkeit der einzelnen Varianten als absolutes Kriterium – unterschreitet sie eine festgelegte untere Schranke wird die entsprechende Variante verworfen.

Unbekannte Werte in numerischen Attributen lassen sich hingegen aufgrund ihres stetigen Wertebereichs nicht in der PRE-Phase ersetzen. Zwar erfolgt die Aufspaltung theoretisch ebenfalls im Rahmen der Feature-Generierung, doch kann diese für numerische Attribute praktisch erst während der EVAL-Phase erfolgen. Ist ein Test (auf ein numerisches Attribut) zur Evaluierung ausgewählt, wird ein Beispiel mit unbekanntem Attributwert »on demand« aufgespalten in einen Teil, der die Bedingung erfüllt und einen, der nicht. Das Gewicht des Beispiels wird dabei im Sinne der DBI-Philosophie entsprechend der Verteilung des Attributs bezüglich des aktuellen Tests aufgeteilt. Diese Verteilung ist dann auch für die SEP-Phase maßgeblich – dabei wird weiterhin das Mindestgewichtskriterium angewendet: sinkt das Gewicht unter die spezifizierte Schranke, so wird der betreffende Teil komplett entfernt.

KLASSIFIKATIONSPHASE

Zum Klassifizieren eines neuen Beispiels werden zunächst analog zum Vorgehen in der Lernphase die fehlenden nominellen Attributwerte »ersetzt«, das zu klassifizierende Beispiel also durch alle hinreichend wahrscheinlichen Varianten ersetzt. Für jede dieser Varianten wird anschließend eine Vorhersage getroffen. Durch Tests auf numerische Attribute mit unbekannten Werten kann eine einzelne Regel dabei möglicherweise nur einen Teil des Beispiels abdecken (Maßgeblich für die Größe des Anteils ist – in Konsistenz mit der SEP-Phase – die Verteilung des Attributes zur Trainingszeit). In diesem Fall wird die Wahrscheinlichkeit für die von der Regel vorhergesagten Klasse entsprechend dem aktuellen Gewicht des Beispiels und dem Abdeckungsgrad der Regel erhöht – im Gegenzug wird das Gewicht des Beispiels um denselben Wert reduziert. So wird die gelernte Regelmenge sequentiell durchlaufen bis das Beispiel komplett abgedeckt ist. Auch für eine einzelne Variante des ursprünglichen Beispiels können also mehrere Klassen vorhergesagt werden. Die (möglicherweise mehrwertigen) Vorhersagen für die einzelnen Varianten des Beispiels werden aufsummiert und schließlich diejenige Klasse vorhergesagt, welche insgesamt das größte Gewicht auf sich vereinen konnte.

Es kann im Rahmen der Behandlung der unbekannten Werte in den nominellen Attributen jedoch auch vorkommen, dass alle möglichen Varianten das erforderliche Mindestgewicht unterschreiten, die Anwendung der DBI-Routine auf das betreffende Beispiel mithin nicht möglich ist. Aus diesem Grunde muss (wie für NN) eine Rückfallroutine hinterlegt werden, die für diesen Fall den Umgang mit dem Beispiel spezifiziert. Zu diesem Zwecke kommt (wie bei NN) wiederum die IGNORE-Routine zum Einsatz.

KONFIGURATION

Das Verhalten der DBI-Routine wurde mit sieben verschiedenen Mindestgewichten zwischen 0,01 und 0,95 untersucht. (Ein Mindestgewicht von 0,95 führt für nominelle Attribute im Wesentlichen zu einem zu DELETE identischen Verhalten.) Zur Kennzeichnung der einzelnen DBI-Varianten wurde der jeweils verwendete Mindestgewichtswert ohne die führende Vorkommanull angehängt – also beispielsweise DBI.01 für DBI mit einem Mindestgewicht von 0,01.

4.7 SpecialValue

Die zuvor beschriebenen Routinen Common, NN und DBI sind allesamt Ersetzungsmethoden und zielen im Kern darauf ab, auf die eine oder andere Weise den »wahren« Attributwert zu approximieren. Im Unterschied dazu versucht die Special-Routine aus dem Fehlen an sich Informationen zu gewinnen, indem unbekannte Werte grundsätzlich als eigenständige Kategorie interpretiert werden. Ein solcher Ansatz ist letztlich eng verwandt mit Ignore – das Behandeln von unbekannten Werten als eigenständige Kategorie führt dazu, dass diese genau wie in der Ignore-Routine von keiner normalen Bedingung abgedeckt werden können. Die Special-Routine erweitert nun Ignore dahingehend, dass zusätzlich zu den normalen Bedingungen auch auf die Sonderkategorie »unbekannt« getestet werden kann.

In theoretischen Szenarien – wie beispielsweise auch beim Präparieren der Datensätze ohne fehlende Attributwerte (vgl. Kapitel 3.2) – geht man häufig von einem komplett zufälligen Fehlen der Werte aus. Unter einer solchen MCAR-Annahme (*missing completely at random*) kann man von dieser zusätzliche Option keinerlei Gewinn erwarten – da in den unbekannten Werten tatsächlich keinerlei Information steckt, ließe sich ein gelernter Test auf die »unbekannt«-Kategorie nur als Artefakt von Overfitting erklären. Im Idealfall würde SpecialValue also keinen solchen Test lernen und sich damit effektiv äquivalent zu Ignore verhalten. Daher kann die SpecialValue-Routine überhaupt nur dann besser sein als Ignore, wenn die MCAR-Annahme nicht zutrifft – was in der Praxis allerdings keinesfalls die Ausnahme sein muss.

Die Realisierung des SpecialValue-Ansatzes gestaltet sich für nominelle Attribute sehr einfach. Bereits in den Pre-Phasen wird eine neue Kategorie für unbekannte Werte definiert und alle tatsächlich unbekannten Werte durch ebendiese neue Kategorie ersetzt. In der Folge werden die unbekannten Werte wie ganz normale Attributwerte behandelt. Für numerische Attribute mit ihrem stetigen Wertebereich lässt sich eine diskrete Kategorie hingegen nicht so nahtlos integrieren. Beim Evaluieren der möglichen Tests auf ein numerisches Attribut muss stattdessen zusätzlich zu den normalen numerischen Bedingungen immer auch noch explizit die »nominelle« Bedingung »Wert ist unbekannt« getestet werden. Unbekannte Werte werden nur beim Testen dieser Bedingung als abgedeckt gewertet. In der Sep-Phase und auch beim Klassifizieren verhält sich die SpecialValue-Routine dann ganz analog, wobei wiederum fallunterschieden werden muss – eine numerische Bedingung kann grundsätzlich nur Beispiele mit bekanntem Wert abdecken, wohingegen die nominelle Spezialbedingung alle Beispiele mit unbekanntem Attributwert erfasst.

4.8 Heuristic Penalty

Der Ansatz der HeuristicPenalty-Routine (HP) basiert auf der in [GLF-08] vorgeschlagenen Übertragung der Idee des von Entscheidungsbäumen bekannten Prinzips des »reduced information gains« [Qui-89] auf das Regellernen. Die Intention besteht darin, das Lernen von Tests auf Attribute mit vielen unbekannten Werten zu erschweren – in der Annahme, dass solche Attribute tendenziell unzuverlässiger sind. Erreicht werden soll dies durch die Abwertung der heuristischen Güteabschätzung der Kandidatenregeln

proportional zur Ausfallrate des (zuletzt) getesteten Attributs – also der Bestrafung der Bewertungsheuristik für unbekannte Werte des getesteten Attributs.

Evaluationsphase

Entsprechend dem in [GLF-08] beschriebenen Vorgehen wird hierfür in der Eval-Phase ein Test auf einen unbekannten Attributwert immer als Fehler gewertet: negative Beispiele werden als falsch-positiv gezählt (abgedeckt), positive Beispiele als falsch-negativ (nicht abgedeckt). Da jede sinnvolle Bewertungsheuristik tendenziell richtige Klassifikationen »belohnen« und falsche »bestrafen« muss, führt dieses Vorgehen effektiv zu der beabsichtigten Abwertung in Abhängigkeit von der Ausfallrate.

Klassifikationsphase

Da bei der Klassifikation neuer Beispiele naturgemäß keine Klasseninformation verfügbar ist (und eine absichtliche Fehlklassifikation ohnehin kaum erstrebenswert wäre), verhält sich HP an dieser Stelle äquivalent zu Ignore, fehlende Werte können also von keiner Bedingung abgedeckt werden. (An Stelle on Ignore kann man an dieser Stelle prinzipiell auch eine beliebige andere Routine anwenden, die fehlende Werte nicht bereits in der Pre-Phase ersetzt, wie etwa AnyValue – dies wurde im Rahmen dieser Arbeit jedoch nicht untersucht. Da die HP-Routine ihrem Wesen nach darauf abzielt, Tests auf unbekannte Attributwerte zu vermeiden, sollte diese Situation ohnehin relativ selten vorkommen.)

Separationsphase

Beim Abtrennen der abgedeckten Beispiele auf Grundlage der in der Eval-Phase ausgewählten Bedingung sind zwei verschiedene Vorgehensweisen möglich.

- Variante 1 – HP.c[onsistent]: die HP-Routine verhält sich in der Sep-Phase konsistent zur Evaluierungsphase. Das Verhalten ist damit insbesondere (und im Unterschied zu allen anderen Routinen) vom Klassenlabel der Beispiele abhängig. Die zum Zwecke der Heuristikbestrafung vorgenommenen absichtlichen Fehlklassifikationen werden dadurch beibehalten.
- Variante 2 – HP.s[imple]: die HP-Routine verhält sich beim Separieren genauso wie beim Klassifizieren (muss also insbesondere unabhängig vom Klassenlabel sein), sodass (sofern dort die Ignore-Routine angewendet wird) die negativen Beispiele mit unbekanntem Attributwert, die beim Evaluieren noch als falsch-positiv gewertet wurden, nicht abgedeckt und somit zu korrekt-negativen Beispielen werden.

Die zweite Variante ist dabei motiviert durch die Überlegung, dass die absichtliche Fehlzuordnung der Beispiele mit unbekannten Attributwerten mit der Bestrafung der Heuristik ihren eigentlichen Zweck bereits erfüllt hat – ein darüber hinaus gehender Nutzen der falschen Zuordnung sich hingegen kaum schlüssig begründen lässt. Hingegen ließe sich argumentieren, dass durch die Konsistenz des Verhaltens mit der Cls-Phase die Wahrscheinlichkeit steigt, dass die Verteilung der Daten beim Klassifizieren derjenigen beim Lernen entspricht.

Numerische Unschärfe

Parallel zur Behandlung unbekannter Attributwerte ermöglicht die HP-Routine auch den gezielten Umgang mit Ungenauigkeit von numerischen Attributen. Eine ausführliche

Beschreibung des Ansatzes findet sich ebenfalls in [GLF-08]. Es gibt prinzipiell zwei mögliche Szenarien um die Einführung von numerischer Unschärfe (NUS) zu motivieren:

Zunächst einmal ist die Größe des Unschärfebereichs nichts anderes als ein weiterer Optimierungsparameter des Lerners, der es ermöglicht, die Anzahl der Beispiele zu reduzieren, die bei Tests auf ein numerisches Attribut als abgedeckt gewertet werden – ist der Unschärfebereich genauso groß wie der Wertebereich des Attributs wäre die Wirkung äquivalent zur Entfernung des gesamten Attributs aus dem Datensatz. Insofern gibt es für jeden Datensatz eine optimale NUS-Konfiguration, welche die erzielte bzw. erwartete Genauigkeit des Lerners maximiert.

Alternativ kann die Einführung von NUS aber auch durch spezifisches Domänenwissen induziert sein (wobei dies natürlich nicht im Widerspruch zum Optimierungsansatz stehen muss, stehen sollte, sondern eher auf diesem aufbaut). So sind für Messungen physikalischer Größen in der Regel die Ungenauigkeiten der verwendeten Messgeräte bekannt. Zudem kann es vorkommen, dass man (selbst wenn die numerischen Daten an sich exakt vorliegen) Unterschiede bis zu einer bestimmten Höhe ganz bewusst nicht berücksichtigen möchte – etwa um zu verhindern, dass bei Altersangaben ein Unterschied von wenigen Wochen zu komplett verschiedenen Vorhersagen führt.

Insbesondere im Falle der domänenspezifischen Unschärfe liegt es auf der Hand, dass jedes numerische Attribut seinen individuellen Unschärfebereich benötigt. Für die bloße Parameteroptimierung ist dies hingegen nicht zwingend erforderlich, vergrößert jedoch den optimierbaren Parameterraum erheblich. Für die hier durchgeführten Untersuchungen zur NUS gelten dabei folgende zwei Punkte:

(1) Der Bemessung der Unschärfebereiche lag keinerlei Domänenwissen zugrunde.
(2) Die Unschärfebereiche waren nicht attributspezifisch, es wurde stets eine globale Schranke für alle Attribute verwendet.

KONFIGURATION

In den hier angeführten Untersuchungen wurde stets die Consistent-Variante der HP-Routine eingesetzt. Mit insgesamt sieben verschiedenen Unschärfeschranken zwischen 0 und 0,5. Zur Unterscheidung der Varianten wird dem Routinenkürzel der Nachkommaanteil des jeweils aktuellen Unschärfeparameterwerts angehängt – so steht beispielsweise HP.C.10 für EVAL-Phasen-konsistentes HP mit einem Unschärfeintervall von ±0,1.

5 Auswertung

Die acht vorangehend beschriebenen Behandlungsroutinen wurden auf eine Reihe von UCI-Datensätzen angewendet (für Details siehe Kapitel 3 sowie Appendix [B] und [C]). Die dabei erhaltenen Resultate werden im Folgenden präsentiert. Zunächst wird auf die auf den drei präparierten Datensätzen erzielten Ergebnisse eingegangen, daran schließt sich die Auswertung für die realen Daten an.

Die drei Routinen DBI, HP und NN lassen sich über Parameter konfigurieren. Für die Vergleiche mit den anderen Routinen wurde dabei jeweils eine – empirisch ermittelte – Standardkonfiguration verwendet. Wo aufschlussreich, wurde anschließend separat der Einfluss des Parameters evaluiert. Als Standard gelten dabei die folgenden Parametereinstellungen:

- DBI mit einem MIW von 0,05 (DBI.05)
- NN mit einer Nachbarschaft der Größe K=9 (NN.9)
- HP im consistent-Modus ohne Berücksichtigung numerischer Unschärfe (HP.C.0)

Neben der erzielten Genauigkeit wurden im Rahmen der durchgeführten Untersuchungen auch die benötigten Laufzeiten erhoben. Am Ende dieses 5. Kapitels findet sich auch hierzu eine kurze Auswertung.

5.1 Ergebnisse: Präparierte Datensätze

Um untersuchen zu können, wie sich die eingesetzten Behandlungsmethoden auf die gelernten Modelle auswirken, wurden in den Testreihen auf den präparierten Datensätzen weitere Kennzahlen erhoben – die Anzahl der gelernten Regeln und Bedingungen im Allgemeinen sowie im Besonderen derjenigen darunter, die auf eines der KO-Attribute testen bzw. solche Bedingungen enthalten.

5.1.1 »credit-g« – Bewertung deutscher Kredite

Augenscheinlichste Auffälligkeit der in Abbildung 2 dargestellten Genauigkeitskurven ist zweifelsohne der Verlauf der Delete-Kurve. Auf einem Ausfallniveau von 30% liegt die mit Delete erzielte Genauigkeit nur einen halben Prozentpunkt unterhalb des Ausgangswerts und selbst auf dem 60%-Niveau (also unter Verwendung von lediglich knapp 7% der ursprünglich verfügbaren Beispiele) beträgt der Genauigkeitsverlust noch weniger als 3 Pro-

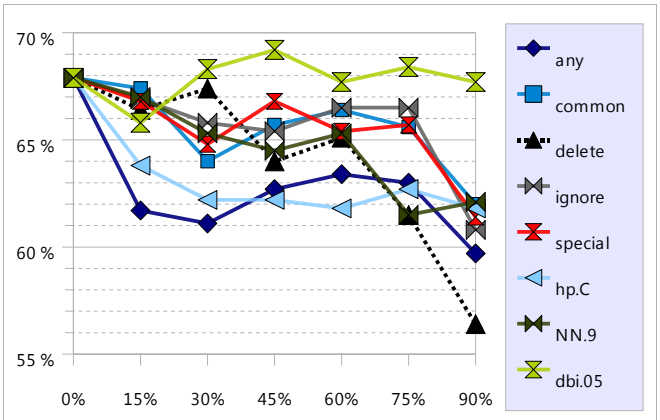

Abbildung 2: *Genauigkeiten der Standardroutinen auf dem credit-g-Datensatz*

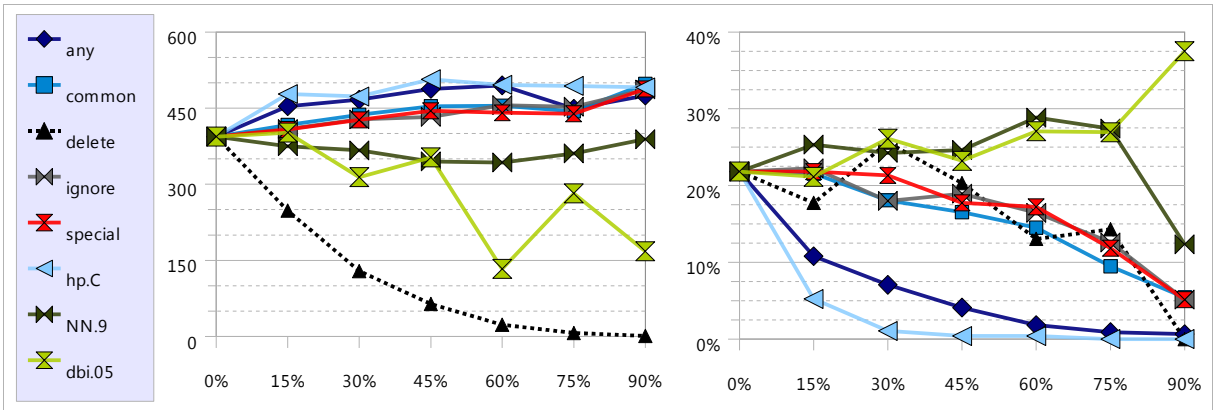

Abbildung 3: *Modellbildung auf den credit-g-Daten – Gesamtanzahl der gelernten Bedingungen (links) und Anteil der Tests auf KO-Attribute (rechts)*

zentpunkte. Jedoch spricht dies alles trotzdem nicht für Delete als Behandlungsroutine, sondern ist vielmehr Zeichen für extremes Overfitting – tatsächlich liegt die Zero-Rule-Genauigkeit auf dem credit-g-Datensatz bei 70%.

Es liegt auf der Hand, dass in einem solchen Szenario sowohl das Einsetzen von notwendig fehlerhaften Schätzwerten als auch der Versuch, vermehrt aus anderen – gemäß Konstruktion mutmaßlich schlechteren – Attributen zu lernen, kaum geeignet sein können, die Genauigkeit gegenüber Delete zu steigern. Insofern kann es nicht überraschen, dass sich die Genauigkeiten der meisten Routinen lange (nur) im Bereich von Delete bewegen. HP und AnyValue schneiden zunächst sogar deutlich schlechter ab. Lediglich DBI kann sich hier mehr oder weniger deutlich von den anderen Routinen absetzen und teilweise sogar höhere Genauigkeiten erzielen als auf dem originalen Datensatz.

Wirft man einen genaueren Blick auf die von den einzelnen Routinen gelernten Modelle, so offenbaren sich markante Parallelen zu den erzielten Genauigkeiten. So sind HP und AnyValue, die beiden auf diesem Datensatz ungenauesten Methoden, auch die einzigen Routinen, die den Anteil an Tests auf die KO-Attribute massiv und konsequent reduzieren (siehe Abbildung 3). Insbesondere ist die absolute Anzahl dieser Tests mit HP sogar geringer ist als mit Delete – obwohl das gelernte Modell um ein vielfaches größer ist. Im Gegensatz dazu steigt der Anteil (derartiger Tests) mit DBI, der fast immer optimalen Routine, sogar an. Das »natürlichste« Verhalten legen hingegen Common, Ignore und SpecialValue an den Tag, die den Anteil an Tests auf die KO-Attribute mit Zunahme der Ausfallrate kontinuierlich reduzieren und sich damit auch in puncto Genauigkeit zwischen den beiden Extremen positionieren.

5.1.2 »kr vs. kp« – Schachendspiel

Auf dem originalen krkp-Datensatz erreicht der SeCo-Lerner eine Genauigkeit von nahezu 99%, beschreibt die Domäne also nahezu perfekt. Hingegen erreicht ZeroRule lediglich 53% Genauigkeit – insofern stellt sich die Situation grundlegend anders dar als auf den »credit-g«-Daten. Entsprechend verschieden ist auch der Verlauf der Genauigkeitskurven für die verschiedenen Routinen, wie Abbildung 4 zeigt: für Delete als Referenzroutine ist ein näherungsweise lineares Absinken der Genauigkeit bis etwa auf ZeroRule-Niveau zu verzeichnen. Die restlichen Routinen lassen sich anhand der erzielten Genauigkeiten in zwei Gruppen unterteilen. Erneut sind es HP und AnyValue,

die insbesondere auf den unteren Ausfallniveaus stark an Genauigkeit verlieren und wiederum schlechter abschneiden als DELETE. Alle anderen Routinen weisen einen nahezu identischen Genauigkeitsverlauf auf, der durch relativ gleichmäßige Verluste geprägt ist, die jedoch im Vergleich mit DELETE wesentlich geringer ausfallen.

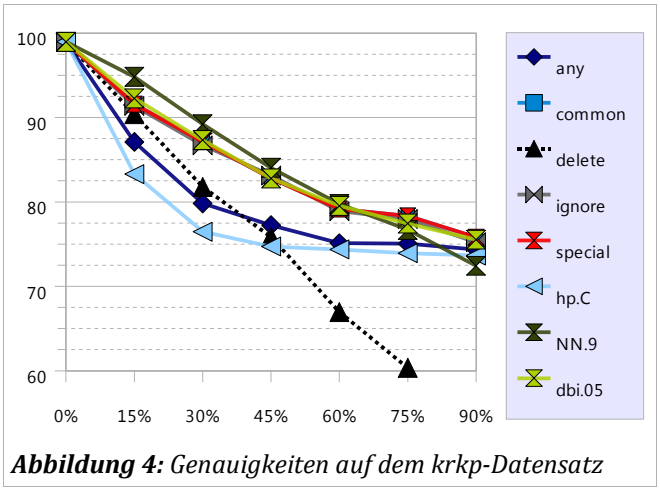

Abbildung 4: *Genauigkeiten auf dem krkp-Datensatz*

Untersucht man die mit den einzelnen Routinen gelernten Modelle genauer, lässt sich anhand der beobachteten Kennzahlen diesmal jedoch kein klarer Trend ablesen. Bezüglich der Modellgröße ist auf dem untersten Ausfallniveau eine Dreiteilung der Routinen zu verzeichnen (Tabelle 1) – während sich das mit IGNORE und NN gelernte Modell um weniger als 20% vergrößert, führt die Anwendung von SPECIAL, COMMON und HP zu einem Modellwachstum in der Größenordnung von 200% – mit ANYVALUE und DBI ist sogar ein Zuwachs von rund 400% zu verzeichnen. Im Gegensatz zu HP und ANYVALUE führt diese enorme Modellvergrößerung bei SPECIALVALUE, COMMON und DBI jedoch nicht zu einer eklatanten Verminderung der erzielten Genauigkeit.

Auch was den (in Abbildung 5 visualisierten) Anteil der Tests auf die KO-Attribute anbelangt, ergibt sich keine klare Korrelation zu den erzielten Genauigkeiten. Während der Anteil dieser Tests auf dem untersten Ausfallniveau mit HP leicht sinkt (wie sonst nur mit DELETE), ist beim vergleichbar ungenauen ANYVALUE analog zu den deutlich besseren Routinen DBI, NN und IGNORE ein Anstieg um 5 Prozentpunkte zu verzeichnen.

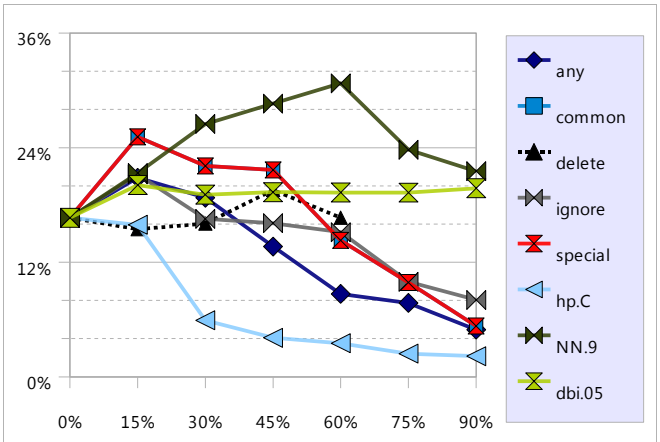

Abbildung 5: *Anteil der Tests auf KO-Attribute (krkp)*

DELETE	IGNORE	NN	COMMON	SPECIAL	HP	DBI	ANYVALUE
-30%	+9%	+19%	+186%	+186%	+191%	+391%	+414%

Tabelle 1: *Modellbildung auf den krkp-Daten bei 15%-igem Attributausfall – Größenänderung (bzgl. Anzahl gelernter Bedingungen) gegenüber dem auf den vollständigen Daten gelernten Basismodell*

5.1.3 »SEGMENT« – PIXELERKENNUNG

Im segment-Datensatz sind im Gegensatz zu den krkp-Daten alle Attribute numerisch. Die Beispiele gehören sechs verschiedenen Kategorien an, die alle mit der selben Häufigkeit auftreten – mit ZeroRule lässt sich daher nur eine Genauigkeit von $^1/_6$ erzielen. Hingegen erreicht der SeCo-Lerner auf den unveränderten Daten eine Genauigkeit von 94,33%.

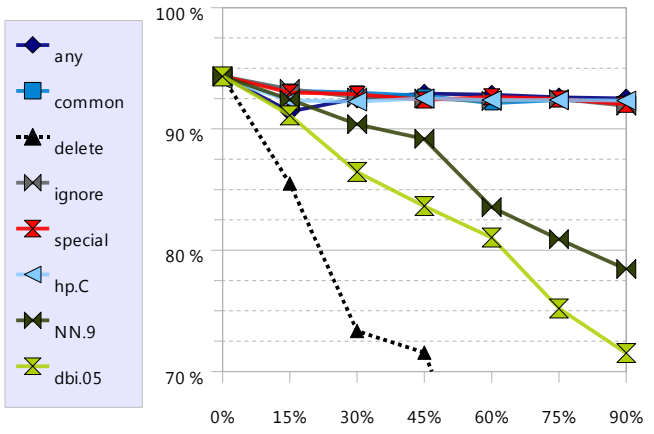

Abbildung 6: *Genauigkeiten auf dem segment-Datensatz*

Der Genauigkeitsverlauf von DELETE (Abbildung 6) entspricht im wesentlichen dem auf dem krkp-Datensatz, die Genauigkeit sinkt relativ gleichmäßig bis auf unter 50% bei 75% Attributausfall. Beim Blick auf das Verhalten der anderen Routinen ergibt sich jedoch ein anderes Bild. Während sich die Genauigkeit mit NN und DBI am Ende um knapp 16% bzw. 23% verringert, ist bei allen anderen Routinen fast kein Rückgang der Genauigkeit zu verzeichnen.

Im Gegensatz zu NN und DBI sind diese Routinen also offensichtlich in der Lage, die ursprünglich verwendeten KO-Attribute adäquat durch andere Attribute zu ersetzen. Dieser grundlegende Unterschied spiegelt sich auch in den gelernten Modellen wider. Wie in Abbildung 7 deutlich wird, bilden NN und DBI im Gegensatz zu den anderen Routinen mit zunehmenden Attributausfall immer kleinere Modelle, während zugleich der Anteil an Tests auf KO-Attribute konstant bleibt oder zunimmt. Hingegen lernen die restlichen Routinen immer größere Modelle mit mehr oder weniger stark sinkendem Anteil an Tests auf KO-Attribute. Die bei weitem stärkste Reduktion ist dabei wiederum bei HP und ANYVALUE zu beobachten. Durch die bereits konstatierte offenkundige Ersetzbarkeit der KO-Attribute wirkt sich dies – anders als auf den anderen Datensätzen – jedoch nicht negativ auf die Genauigkeit aus.

Insofern ist auch plausibel, dass NN und DBI auf diesem Datensatz deutlich schlechter abschneiden – diese Routinen sind offenbar nicht in der Lage, die in den anderen Attributen enthaltene Information effektiv auszunutzen, sondern stützen sich auch bei höheren Ausfallraten noch in starkem Maße auf die ursprünglich besten Attribute. Da deren Qualität jedoch zunehmend schlechter werden muss, sinkt nahezu zwangsläufig auch die mit einem daraus bestehenden Modell erzielbare Genauigkeit.

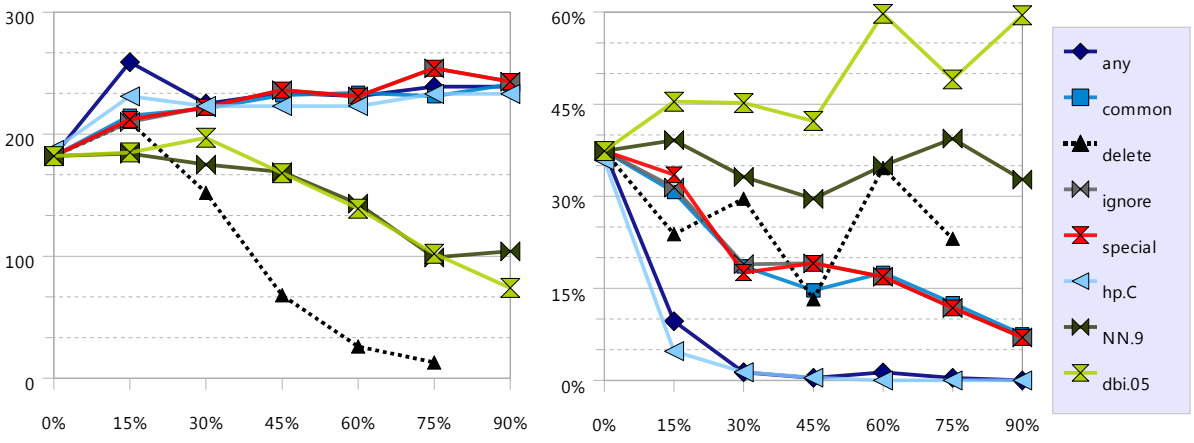

Abbildung 7: *Modellbildung auf den segment-Daten – Gesamtanzahl der gelernten Bedingungen (links) und Anteil der Tests auf KO-Attribute (rechts)*

5.1.4 Zwischenfazit – präparierte Daten

Insgesamt lässt sich festhalten, dass Delete nur dann konkurrenzfähig ist, wenn die offensichtlich bestehenden Nachteile, die aus der »Behandlung« fehlender Attributwerte in dieser Routine resultieren, durch die Verminderung von Overfitting aufgewogen werden können – wie es auf den »credit-g«-Daten der Fall ist.

Im Gegensatz hierzu erzielten die drei Routinen Ignore, SpecialValue und Common auf allen drei präparierten Datensätzen konstant gute Ergebnisse. Dabei war bei diesen drei Routinen in der Regel ein relativ gleichmäßiger Rückgang des Anteils an KO-Tests zu beobachten, was auch das am natürlichsten erscheinende Verhalten ist, da es am ehesten dem tatsächlichen Qualitätsverlust der Attribute entsprechen dürfte.

Demgegenüber fielen HP und AnyValue in erster Linie durch die auffallend starke Reduktion der Tests auf die KO-Attribute auf – bereits auf dem zweitniedrigsten Ausfallniveau von 30% werden, wie Abbildung 8 veranschaulicht, nahezu keine KO-Attribute mehr verwendet. Während dieses Verhalten im Rahmen von HP sogar explizit forciert wird, handelt es sich im Falle von AnyValue um einen – überraschend stark ausgeprägten – Nebeneffekt. Hier wirkt sich offenbar entscheidend die abnehmende Selektivität der KO-Attribute aus. Dabei erfolgt die Reduktion offenbar weitgehend unabhängig vom tatsächlichen Nutzen der KO-Attribute, ist sie doch ganz analog bei allen drei hier untersuchten Datensätzen zu beobachten. Dieses Verhalten erscheint ob seiner mangelnden Adaptivität zumindest zwiespältig: ist die in den KO-Attributen enthaltene Information aus den anderen Attributen reproduzierbar, ist es sicherlich von Vorteil, stattdessen Attribute ohne fehlende Werte zu verwenden – sind die KO-Attribute hingegen nicht zu ersetzen, büßt man durch den frühzeitigen Verzicht auf sie unnötig an Genauigkeit ein. Erschwerend kommt für die beiden Routinen dabei der Versuchsaufbau hinzu, da die KO-Attribute gemäß Konstruktion der präparierten Datensätze gerade die vermeintlich wertvollsten sind.

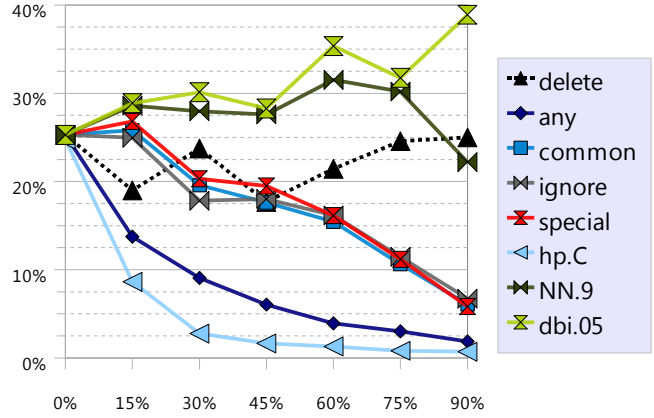

Abbildung 8: *durchschnittliche Anteile an KO-Tests mit den Standard-Routinen auf den präparierten Datensätzen*

Ein gegenteiliges Verhalten war hingegen mit DBI und NN zu beobachten: Diese beiden Routinen hielten den Anteil an Tests auf KO-Attribute im Wesentlichen konstant oder bauten ihn sogar aus. Dies sollte dann gut funktionieren, wenn die betroffenen Attribute tatsächlich nur schlecht zu ersetzen sind, aber eher suboptimale Ergebnisse liefern, wenn die enthaltene Information redundant ist und sich ebenso über Attribute ohne fehlende Werte abbilden lässt. Diese komplementären Eigenschaften von HP/AnyValue einer- sowie DBI/NN andererseits liefern zumindest eine plausible Erklärung dafür, dass die Gruppen auf allen drei untersuchten Datensätzen niemals gleich gut oder schlecht funktionierten.

Bei alledem gilt es aber zu beachten, dass eine Untersuchung auf drei Datensätzen grundsätzlich keine statistisch belastbaren Aussagen liefern kann, sondern in erster Linie dabei helfen soll, grundsätzliche Verhaltensweisen zu erkennen und entsprechende Thesen zu entwickeln.

5.2 Ergebnisse: Natürliche Daten

Zunächst gilt es, sich der im Vergleich zu den präparierten Daten deutlich veränderten Voraussetzungen für die Behandlungsroutinen bewusst zu werden. Zum einen ist aufgrund der ungleich größeren Anzahl an betrachteten Datensätzen eine wesentlich breitere Streuung der Datensatzcharakteristika gegeben. Insbesondere entfällt der unvermeidliche systematische Einfluss des Präparationsverfahrens. Eine Eins-zu-eins-Übertragung der auf den präparierten Datensätzen erzielten Ergebnisse kann daher von vornherein nicht erwartet werden. Ferner finden sich hier nur über mehrere Datensätze gemittelte Genauigkeitsangaben. Eine detaillierte Auflistung der Einzelergebnisse für alle Datensätze und Routinen ist jedoch im Appendix [D] enthalten.

5.2.1 Standardkonfiguration

Betrachtet man die Abweichungen von der Mediangenauigkeit, so wird ersichtlich, dass sechs von acht Routinen im Mittel in einem engen Bereich von nur ±0,5% um den Median liegen (Abbildung 9) – von HP.C (+0,41%) bis AnyValue (-0,45%). Deutlich unter dem Median liegt im Mittel neben dem zu erwartenden Delete lediglich die DBI-Routine.

Untersucht man die paarweisen Genauigkeitsdifferenzen zwischen den Routinen, so ergeben sich auf einem Signifikanzniveau von 0,01 signifikante Unterschiede zwischen Delete und allen anderen Routinen mit Ausnahme von DBI. Akzeptiert man einen Fehler von 0,075, wird auch noch DBI besser als Delete – weitere signifikanten Unterschiede sind jedoch nicht festzustellen. Durch die zahlreichen paarweisen Tests liegt die globale Irrtumswahrscheinlichkeit jedoch oberhalb des Niveaus für die einzelnen Tests.

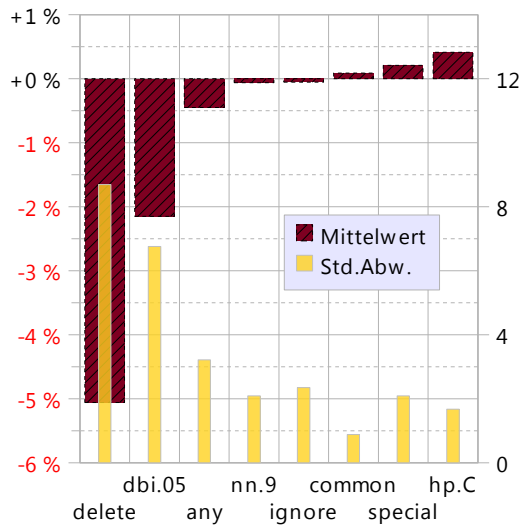

Abbildung 9: *durchschnittliche Abweichungen der mit den Heuristiken erzielten Genauigkeiten von der Median-Genauigkeit*

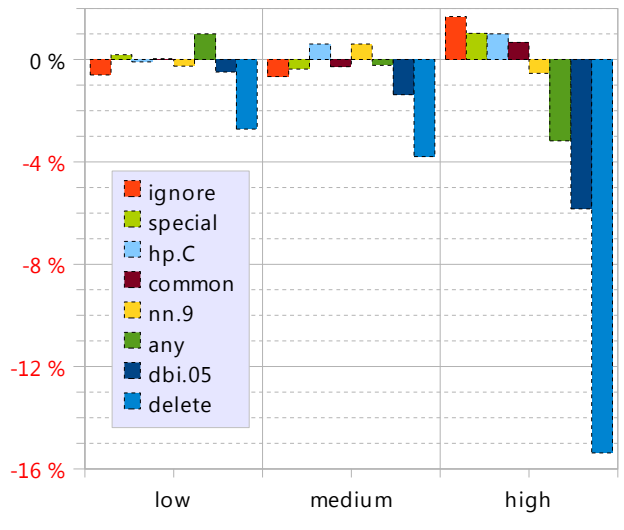

Abbildung 10: *mittlere Abweichungen von der Mediangenauigkeit, gruppiert nach MAV-Score*

Einen simultanen Vergleich aller Routinen mitsamt einem global gültigen Konfidenzniveau ermöglicht beispielsweise der Friedman-F-Test ([Dem-06],[ID-80]). Als nicht-parametrischer Test trifft dieser – im Unterschied zum zuvor durchgeführten t-Test oder einer ANOVA – keine impliziten Annahmen bezüglich der Verteilung der Daten (insbesondere wird keine Varianzhomogenität verlangt). Die in [Dem-06] präsentierten Untersuchungen legen zudem nahe, dass der Friedman-F-Test in der Praxis (des maschinellen Lernens) durchaus keine schwächeren Aussagen liefert als ANOVA. Da dieser Test nur die Platzierung der Routinen und nicht die Größe der Unterschiede berücksichtigt, erhalten hierbei insbesondere die zehn Datensätze mit geringem MAV-Score ein relativ großes Gewicht, gemessen an den dort zum Großteil nur marginalen Unterschieden. Dies ist auch der hauptsächliche Grund dafür, dass die AnyValue-Routine bezüglich des durchschnittlichen Rangs an zweiter Stelle rangiert (vgl. auch Abbildung 10 oben) – die sonstige Reihenfolge der Routinen entspricht hingegen der zuvor anhand der mittleren Genauigkeit ermittelten.

Aus den unten in Tabelle 2 aufgelisteten Rängen errechnet sich für die Friedman-F-Statistik ein Wert von 2,51. Da die Statistik F(7, 161)-verteilt ist, lässt sich damit die Nullhypothese, dass die Wahl der MAVH-Routine keinen Einfluss auf die erzielte Genauigkeit hat, auf einem Niveau α von 0,05 verwerfen. Daher kann man nunmehr versuchen, mittels eines post-hoc-Tests signifikante Unterschiede zwischen einzelnen Routinen aufzuspüren. Der Nemenyi-Test erfordert für k=8 Klassifizierer und N=24 Datensätze einen mittleren Rangunterschied, eine »critical difference« (CD_α), von 1,97 (für α=0,1) bzw. 2,14 (α=0,05) um einen signifikanten Unterschied festzustellen. Auf diese Weise lassen sich somit lediglich HP (α=0,05) sowie AnyValue (α=0,1) von Delete unterscheiden. Alle anderen Rangunterschiede unterschreiten die CD auch für α=0,1 und können demnach nicht als signifikant angesehen werden. Das liegt nicht zuletzt in der Tatsache begründet, dass es für jede Routine Datensätze gibt, auf denen der jeweils verfolgte Ansatz nur schlecht funktioniert. Dies spiegelt sich auch in der relativ breiten Streuung der gewonnenen Datensätze (ebenfalls Tabelle 2). Das zugehörige CD-Diagramm für den Nemenyi-Test und ein Signifikanzniveau von α=0,05 findet sich unten in Abbildung 12.

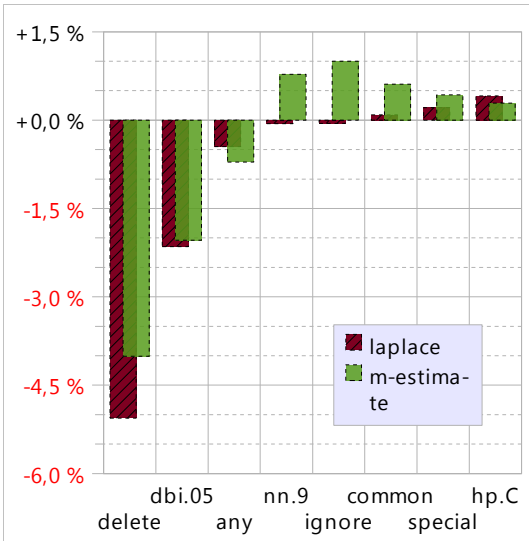

Abbildung 11: *durchschnittliche Abweichungen von der Mediangenauigkeit mit Laplace und m-estimate als Bewertungsheuristik*

Insgesamt lässt sich anhand der erzielten Ergebnisse lediglich die Delete-Routine mit Sicherheit als unterlegen einstufen. Die Unterschiede zwischen allen anderen Routinen sind insbesondere nicht konstant genug, hängen zu sehr vom jeweiligen Datensatz ab, als dass sich eine eindeutige Empfehlung rechtfertigen ließe.

		Delete	DBI	NN	Ignore	Common	Special	AnyValue	HP
L	# winner	2	4	3	4	3	6	4	7
	Ø rank	5,9	5,0	4,6	4,5	4,3	4,2	3,9	3,5
M	# winner	1	2	4	7	5	6	3	4
	Ø rank	6,4	5,4	4,9	4,2	4,1	3,7	3,7	3,7

Tabelle 2: *Anzahl der gewonnenen Datensätze und durchschnittlich belegter Rang der Standard-Routinen, mit dem Laplace-Maß (L, oben) und m-estimate (M, unten) als Bewertungsheuristik*

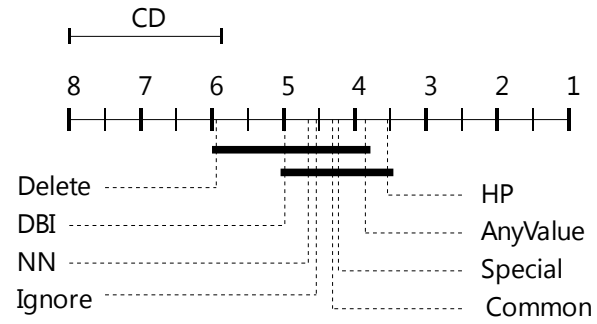

Abbildung 12: *Signifikanzintervalle gemäß Nemenyi-Test (α=0,05, Laplace-Heuristik)*

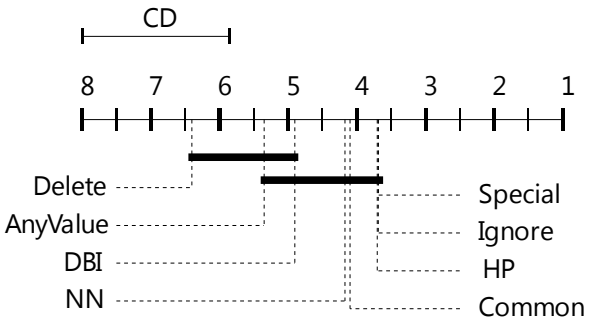

Abbildung 13: *Signifikanzintervalle gemäß Nemenyi-Test (α=0,05, m-estimate-Heuristik)*

5.2.2 Ergebnisse mit Alternativer Bewertungsheuristik

Die Modellbildung mit den acht Standardroutinen wurden wiederholt mit m-estimate an Stelle des Laplace-Maßes als Bewertungsfunktion für zu lernende Regeln. Auf diese Weise soll überprüft werden, inwiefern sich die bisher gemachten Beobachtungen übertragen lassen auf diese alternative SeCo-Konfiguration.

Grundsätzlich erhöhte sich die durchschnittlich erzielte Genauigkeit für alle Routinen zwischen 0,97 (DBI.05) und 2,46 Prozentpunkte (Ignore). Die mittleren (Median-zentrierten) Genauigkeiten weichen dabei nicht wesentlich von den vorherigen Ergebnissen ab (Abbildung 11) – auffallend ist allenfalls die vergrößerte Differenz zwischen AnyValue und den »Top-5«-Routinen, die von 0,4 auf über 1% anwächst. Dies manifestiert sich auch bei der Rank-Analyse – hier sinkt der mittlere Rang von AnyValue von 3,9 (und damit Platz 2) auf nur noch 5,4 (Platz 6). Auf den ersten Blick scheint hier daher eine deutlichere Trennung von guten und weniger guten Routinen stattzufinden.

Auch die Friedman-F-Statistik liefert einen deutlich höheren Wert von 4,44 – die Nullhypothese lässt sich damit bei Verwendung der m-estimate-Heuristik sogar auf einem Niveau von α=0,0005 verwerfen. Für die damit möglichen paarweisen Nemenyi-Tests bleiben die CDs unverändert – damit sind auf einem Niveau von 0,05 immerhin die fünf besten Routinen (NN, Common, Special, Ignore, HP) signifikant besser als Delete, wie auch dem zugehörigen CD-Diagramm (Abbildung 13) zu entnehmen ist. Weitere signifikante Rangdifferenzen sind jedoch auch mit m-estimate auch mit Verdoppelung des akzeptierten Fehlers nicht nachzuweisen.

5.2.3 Distribution-Based-Imputation

Neben dem zuvor als Standard verwendeten Minimalgewicht von $^1/_{20}$ wurden sechs weitere DBI-Konfigurationen mit Mindestgewichten (MIWs) zwischen 0,01 und 0,95 untersucht. Betrachtet man zunächst nur die Charakteristika der gelernten Modelle für die einzelnen DBI-Varianten (Tabelle 3), zeigen sich zwei klare Trends: so werden mit abnehmendem MIW zum einen die gelernten Modelle immer umfangreicher, zum anderen zugleich auch die gelernten Regeln immer komplexer. Das verwendete MIW spiegelt sich auch nahezu perfekt in den Durchschnittsrängen der einzelnen DBI-Varianten – hier ist eine monotone Zunahme zu beobachten von 2,85 für DBI.01 bis zur durchschnittlichen Platzierung von 5,13 für DBI.95 (Tabelle 4). Die Nullhypothese von der Gleichheit aller DBI-Varianten kann mit großer Sicherheit verworfen werden (F_F=5,44 bei 6 und 132 Freiheitsgraden). Unter Verwendung des Nemenyi-Tests lassen sich für α=0,05 jedoch lediglich zwischen DBI.01/05 und DBI.60/95 signifikante Unterschiede nachweisen ($CD_{0,05}$=1,88). Mit dem Bonferroni-Dunn-Test und DBI.01 als Vergleichsklassifizierer wird zudem der Unterschied zu DBI.40 als signifikant erkannt (CD=1,68).

Auch bezüglich der im Mittel erzielten Genauigkeiten ist der Einfluss des MIW sehr gut zu erkennen. Ausgehend von der Standardkonfiguration mit einem MIW von 0,05 führt eine Anhebung desselben offenkundig zu einer deutlichen Verringerung der mittleren Genauigkeit. Eine weitere Absenkung des MIW führt hingegen zu keiner weiteren Erhöhung der mittleren Genauigkeit – hier lohnt sich der Mehraufwand für eine genauere Abbildung der Verteilung der Attribute also nicht mehr. Möglicherweise bewirkt die immer genauere Betrachtung von Beispielfragmenten ab einem gewissen Punkt auch eine zu genaue Anpassung an die Trainingsbeispiele, wofür das beobachtete Lernen von immer komplexeren Regeln ein Indiz sein könnte.

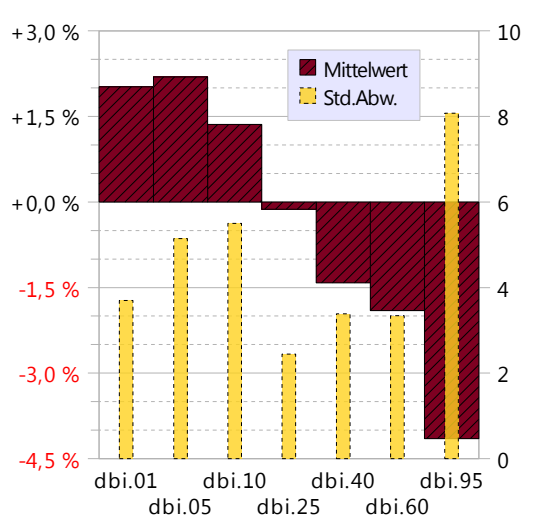

Abbildung 14: *mittlere Abweichungen der DBI-Varianten von der Mediangenauigkeit*

	DBI.01	DBI.05	DBI.10	DBI.25	DBI.40	DBI.60	DBI.95
#cond	+12,83	+7,96	+6,48	-0,13	-6,43	-9,26	-18,00
#cond/rule	+0,18	+0,08	+0,05	-0,02	+0,03	-0,12	-0,32

Tabelle 3: *Modellbildung mit DBI – Gesamtanzahl der gelernten Bedingungen und Anzahl von Bedingungen pro Regel (jeweils als mittlere Abweichung von den Medianwerten)*

	DBI.01	DBI.05	DBI.10	DBI.25	DBI.40	DBI.60	DBI.95
# winner	10	7	5	5	2	3	4
Ø rank	2,8	3,0	3,6	3,7	4,6	5,1	5,1

Tabelle 4: *Anzahl der gewonnenen Datensätze und durchschnittlicher Rang der DBI-Varianten*

5.2.4 Nearest-Neighbor

Da aus Stabilitäts- und Kompatibilitätsgründen stets LinearNNSearch als Suchalgorithmus verwendet wurde, verblieb als Variationsparameter für die NN-Routine in der Praxis nur die Anzahl der betrachteten Nachbarn. Wie den in Tabelle 4 dargestellten Genauigkeiten zu entnehmen ist, sind die Unterschiede zwischen den einzelnen untersuchten Varianten im Mittel jedoch relativ gering, im Einzelfall betrug die Differenz allerdings bis zu 14%. Auch die Betrachtung der durchschnittlichen Ränge (ebenfalls Tabelle 4) liefert keine grundlegend neuen Erkenntnisse – mit Hilfe eines Friedman-F-Tests lässt sich lediglich verifizieren, dass es einen Einfluss der Nachbarschaftsgröße auf die Genauigkeit geben muss, die paarweisen Differenzen sind jedoch allesamt nicht signifikant.

	NN.3	NN.5	NN.9	NN.15
accuracy	+0,17	-0,33	+0,27	-0,64
Ø rank	2,1	3,0	2,5	2,5

Tabelle 5: *mittlere Abweichungen von der Median-Genauigkeit und durchschnittlicher Rang der NN-Varianten*

5.2.5 Heuristic Penalty

Standardmäßig wurde die HP-Routine ohne Berücksichtigung numerischer Unschärfe (NUS) – also mit einem Unschärfebereich von 0 – durchgeführt, um die Vergleichbarkeit mit den anderen Routinen zu gewährleisten. Da die Einführung von NUS nicht im eigentlichen Sinne die Handhabung unbekannter Attributwerte, sondern den Umgang mit vorhandenen (numerischen) Werten betrifft, stand dieser Aspekt auch nicht im Zentrum der hier durchgeführten Untersuchungen.

Neben der Standard-HP-Routine wurden sechs weitere HP-Varianten mit NUS-Schranken zwischen 0,01 und 0,5 getestet. In die folgende Auswertung flossen dabei nur diejenigen 15 der 24 Datensätze ein, auf denen die Berücksichtigung von NUS beobachtbare Auswirkungen auf die erzielte Genauigkeit hatte. Obwohl Versuchsaufbau und Vorgehen (wie in Kapitel 4.8 beschrieben und begründet) hier wirklich sehr einfach gehalten wurden, stieg die durchschnittliche Genauigkeit für fast alle untersuchten Unschärfeschranken gegenüber der HP-Basisvariante an (vgl. Abbildung 15). Der höchste Genauigkeitsdurchschnitt konnte hierbei mit Schranken von 0,05 und 0,1 erzielt werden. Die Verwendung größerer wie kleinerer Schranken führte im Mittel jeweils zu einem Absinken der Genauigkeit. Bemerkenswerter jedoch als die verhältnismäßig geringe Steigerung der durchschnittlichen Genauigkeit:

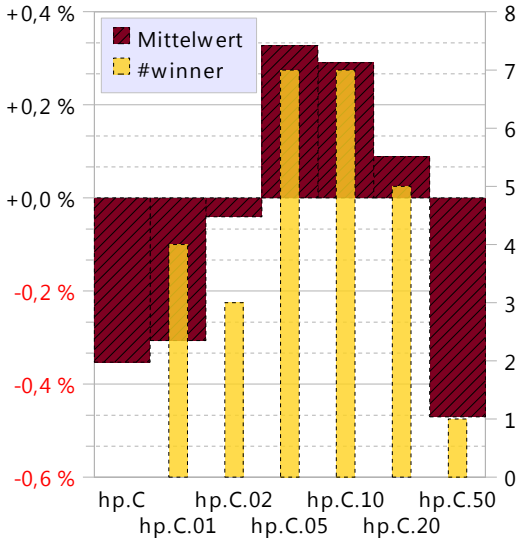

Abbildung 15: *Genauigkeiten der HP-Varianten (mittlere Abweichung vom Median) und Anzahl der jeweils gewonnenen Datensätze*

- der maximale Genauigkeitszuwachs betrug mehr als 8 Prozentpunkte (mit HP.05 auf dem echocardiogram-Datensatz);
- auf allen hier betrachteten Datensätzen erzielte mindestens eine der HP-Varianten eine höhere Genauigkeit als die Standardvariante ohne NUS – im Mittel konnte die Genauigkeit der Standardvariante um knapp 2 Prozentpunkte gesteigert werden

Betrachtet man auch hier wieder die durchschnittlich von den einzelnen Varianten belegten Ränge, so findet man die Reihenfolge der mittleren Genauigkeiten bestätigt. Die beobachteten Rangunterschiede sind allerdings nicht groß genug, um als signifikant gelten zu können – der maximale Rangunterschied zur HP-Basisvariante beträgt rund 1,87 und damit weniger als die mit dem Bonferroni-Dunn-Test für ein Konfidenzniveau von 0,1 notwendige CD von 1,89.

Die Zahl von 0 mit der Basisvariante »gewonnenen« Datensätzen (Abbildung 15) macht jedoch deutlich, dass eine sorgsam auf den jeweiligen Datensatz abgestimmte Wahl des Unschärfebereichs zuverlässig zu einer Erhöhung der erzielten Genauigkeit führen kann. Insbesondere erscheint eine genauere Untersuchung von NUS mit attributspezifischen Schranken als durchaus lohnenswert.

5.3 Laufzeit

Üblicherweise spielt speziell im Bereich des maschinellen Lernens die zur Modellbildung benötigte Laufzeit gegenüber der erzielten Genauigkeit eine eher untergeordnete Rolle. Hinzu kommt der Umstand, dass der SeCo-Lerner auf dem Java-basierten Weka-Framework aufsetzt und die beobachteten Laufzeiten daher den JRE-typischen Schwankungen unterworfen sind, weshalb die erhaltenen Zahlen auch nur als relativ grobe Anhaltspunkte dienen können. Ferner werden die absoluten Laufzeiten in erster Linie von der Größe des gelernten Modells beeinflusst. Da in einem Szenario ohne fehlende Attributwerte alle Behandlungsmethoden (außer den HP-Variationen) dasselbe Modell lernen müssen, können dort bei den jeweiligen Behandlungsansätzen allenfalls minimal verschiedene Initialisierungskosten auftreten, die – so sie überhaupt statistisch nachweisbar sind – auf jeden Fall keinerlei praktische Bedeutung haben. Um zumindest eine gewisse kontrollierte Vergleichbarkeit gewährleisten zu können, werden für den Laufzeitvergleich hier nur die drei präparierten Datensätze herangezogen, wobei für die einzelnen Ausfallniveaus jeweils das arithmetische Mittel der drei Laufzeiten gebildet wird.

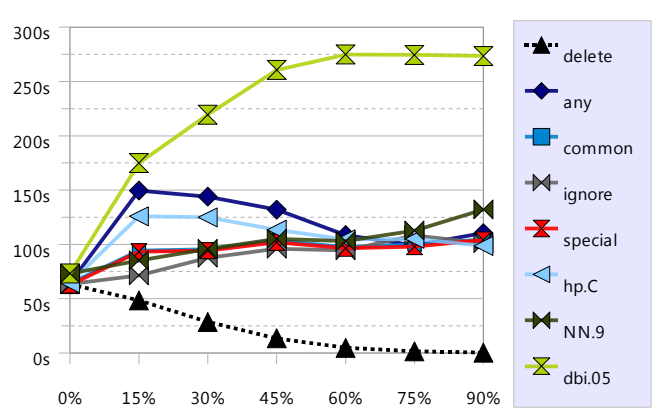

Abbildung 16: *mittlere Laufzeiten der Standard-Routinen*

Wie in Abbildung 16 dargestellt, ist bei Einsatz von Delete wenig überraschend die geringste Laufzeit zu erwarten, was auf die extreme Beschneidung der tatsächlich verwendeten Menge an Trainingsbeispielen zurückzuführen ist: bereits bei einem Ausfallniveau von 15% werden bei drei betroffenen Attributen (im Erwartungswert) fast 40% der

Trainingsbeispiele gelöscht. Die längsten Laufzeiten benötigte jeweils die DBI-Variante – hierbei wirken sich insbesondere fehlende nominelle Attributwerte laufzeitverlängernd aus, da sich hier die Trainingsmenge durch die proportionale Aufspaltung der betroffenen Beispiele vergrößert, während dieser Effekt bei fehlenden numerischen Attributen weniger stark ausgeprägt ist, da sich hier die Größe der Trainingsmenge zunächst nicht ändert.

Auffällig ist die (näherungsweise) Verdoppelung der Laufzeiten von AnyValue und HP bei 15% Ausfall gegenüber der Laufzeit auf den originalen Datensätzen. Die Ursache hierfür ist in den im Kapitel 5.1 dokumentierten größeren Änderungen des gelernten Modells zu sehen (die einhergehen mit eher unterdurchschnittlichen Ergebnissen) – analog zu den Genauigkeiten ist auch bezüglich der Laufzeiten mit Zunahme der Ausfallraten eine Angleichung zu beobachten. Auffallend unauffällig waren hingegen die Laufzeiten der NN-Variationen – anders als man gemeinhin für nachbarschaftsbasierte Ansätze erwarten könnte, bewegten sich die Zeiten immer im unteren Bereich (Delete außen vor gelassen). Auch die Anzahl der betrachteten Nachbarn spielte dabei kaum eine Rolle.

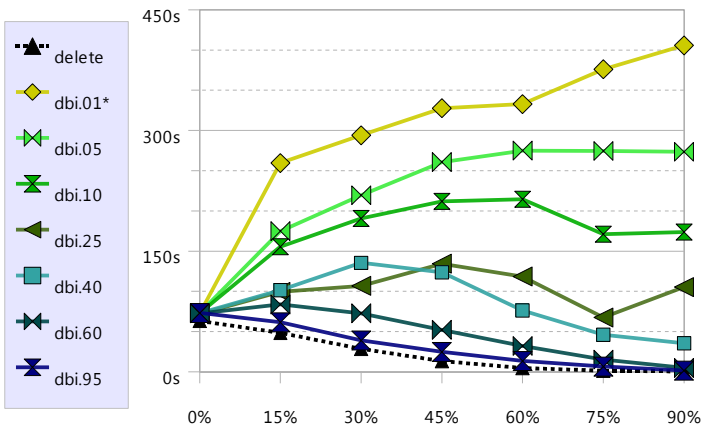

Abbildung 17: *Laufzeiten der DBI-Varianten (* bereinigt)*

Betrachtet man noch einmal im Detail die Laufzeiten der verschiedenen DBI-Varianten (Abbildung 17), lässt sich sehr deutlich die erwartete, mit Zunahme des Minimalgewichts einhergehende, Annäherung an die Delete-Laufzeiten erkennen. Dabei entsprechen die Laufzeiten von DBI.25 am ehesten denen der anderen in Abbildung 16 dargestellten Methoden – ein höheres Mindestgewicht verkürzt, ein geringeres verlängert die benötigten Laufzeiten tendenziell. Dabei wirken sich zwei gegenläufige Effekte auf die Laufzeit aus. Auf der einen Seite vergrößert sich die Menge der Trainingsbeispiele (und damit die Laufzeit) durch die Aufspaltung vorhandener Beispiele mit fehlenden Werten – andererseits verringert sich die Anzahl der Beispiele durch das Verwerfen „zu leichter“ Beispiele, insbesondere in Folge wiederholter Aufspaltungen von Beispielen mit mehreren fehlenden Werten. Besonders gut ist dies anhand der DBI.40-Kurve zu beobachten, wo der Schrumpfungseffekt ab einem Ausfallniveau von 45% überwiegt und die Laufzeit kontinuierlich absinkt.

5.4 Fazit

Das einzig klare und statistisch wasserdichte Resultat der Untersuchungen ist die generelle Unterlegenheit der Delete-Routine. Sie kann allenfalls von der verringerten Anfälligkeit für Overfitting profitieren, die mit dem notgedrungenen Lernen einfacherer Modelle einher geht. Alle anderen Routinen haben mehr oder weniger Stärken und Schwächen, sind für bestimmte Datensätze also gut und für andere hingegen eher schlecht geeignet.

Über alle Datensätze betrachtet sind folgerichtig keine weiteren signifikanten Unterschiede zu beobachten. Dementsprechend kann man für einen konkreten Datensatz a priori auch keine fundierte Empfehlung aussprechen.

Die im Durchschnitt höchste Genauigkeit – und auch die höchste Anzahl an optimalen Ergebnissen – konnte auf den natürlichen Daten mit der HP-Routine erzielt werden. Zudem legen die durchgeführten einfachen Untersuchungen bezüglich der numerischen Unschärfe nahe, dass bei einer intelligenteren und attributbezogenen Wahl der Schranken weitere Verbesserungen der Genauigkeit realisierbar sein könnten. Eine diesbezügliche weitere Untersuchung erscheint in jedem Fall vielversprechend. Vergessen sollte man dabei aber nicht die bestenfalls durchwachsen zu nennenden Resultate von HP auf den präparierten Datensätzen – dort verzichtete HP »zu Unrecht« frühzeitig auf die ausgedünnten Attribute. Möglicherweise fehlt es hier also grundsätzlich an der notwendigen Sensibilität, um auch in nicht vollständig gegebenen Attributen enthaltene Information effektiv nutzen zu können.

Ohne Probleme mit den präparierten Datensätzen, und auch auf den natürlichen Datensätzen im Durchschnitt kaum schlechter, bieten sich insbesondere SpecialValue, Ignore und Common als Alternativen an. Bemerkenswert ist dabei zweierlei. Zum einen, dass HP und SpecialValue zusammen immerhin auf der Hälfte der Datensätze das optimale Ergebnis lieferten. Zum anderen, dass Common zwar nur sehr selten die optimale Wahl darstellte, dafür aber unter allen Routinen die mit Abstand geringste Varianz aufwies.

Aber auch die im Durchschnitt am schlechtesten funktionierenden Routinen AnyValue und DBI können auf einzelnen Datensätzen deutlich bessere Ergebnisse erzielen als die restlichen Verfahren, sodass von ihrer Verwendung keinesfalls generell abgeraten werden kann. Insbesondere sollte berücksichtigt werden, dass nicht ganz klar ist, inwiefern die gemessenen Genauigkeiten tatsächlich im engeren Sinne die Fähigkeiten zum Umgang mit fehlenden Werten widerspiegeln. Oder ob nicht eher die indirekten Auswirkungen auf das Overfitting-Verhalten beobachtet werden – gerade DBI dürfte diesbezüglich durch die Vergrößerung der Trainingsmenge gegenüber den anderen Routinen etwas benachteiligt sein.

6 EINORDNUNG

Die oben beschriebenen Ergebnisse decken sich in weiten Teilen mit den Resultaten von Bruha und Franek aus [BF-96]. Darin definierten und untersuchten sie für den CN4-Regellerner fünf grundlegende Behandlungsstrategien für fehlende Attributwerte, welche auch den Grundstock der im SeCo-Lerner implementierten und in dieser Arbeit untersuchten Strategien bildeten, nämlich:

»ignore« (DELETE), »unknown« (IGNORE), »common«, »split« (DBI) und »anyvalue«

(in Klammern jeweils die im Rahmen des SeCo-Lerners für die Strategien verwendeten Bezeichnungen – sofern abweichend). Als eindeutig schlechteste Strategie identifizierten die Autoren »ignore«, von deren Einsatz sie als einziger Strategie generell abraten. Ins Mittelfeld stuften sie »common« und »split« ein, während sie im Durchschnitt die besten Ergebnisse mit »unknown« und »anyvalue« erzielen konnten. Dabei weisen Bruha und Franek darauf hin, dass die spezifischen Charakteristika der einzelnen Datensätze einen relativ großen Einfluss auf die mit den verschiedenen Methoden erreichbaren Genauigkeiten haben. Keine der Strategien wäre gleichermaßen für alle Datensätze geeignet. Abgesehen von schlechteren Abschneiden von ANYVALUE und DBI (»split«) im Rahmen des SeCo-Lerners, das vermutlich nicht zuletzt auch auf fehlende Maßnahmen gegen Overfitting zurückzuführen ist, konnten die Ergebnisse von Bruha und Franek vollumfänglich bestätigt werden.

Viele der verfügbaren Publikationen zur Behandlung fehlender Attributwerte beziehen sich auf das Lernen von Entscheidungsbäumen. Dass sich die dort erzielten Ergebnisse nicht ohne weiteres auf Regellerner übertragen lassen zeigt sich in [Qui-89]. Quinlan untersuchte darin am C4.5-Entscheidungsbaumlerner eine Reihe von Varianten der Behandlung – dabei kam er im Kern zu dem Ergebnis, dass das Aufspalten der Beispiele mit fehlenden Werten (also in etwa DBI) die mit Abstand beste Variante sei.

Ebenfalls mit Möglichkeiten der Behandlung fehlender Attributwerte im Rahmen des Entscheidungsbaumlernens (speziell zur Klassifizierungszeit) beschäftigen sich Saar-Tsechansky und Provost. Sie stellen in [STP-07] jedoch einen (grundlegend) neuen Ansatz zur Behandlung fehlender Attributwerte vor – die so genannten „reduzierten Modelle“ (RM). Für jedes auftretende Muster von fehlenden Werten wird dabei auf den Trainingsdaten ein eigenes Modell gelernt, dass nur die jeweils vorhandenen Attribute verwendet. Durch diesen Meta-Ansatz ist dieses Verfahren auch grundsätzlich unabhängig vom tatsächlich verwendeten Lernverfahren. Die beiden Autoren vergleichen es in ihrer Studie mit zwei klassischen Ersetzungsmethoden: DBI und der so genannten „predictive value imputation“ (PVI). Letzteres meint die Vorhersage fehlender Attributwerte auf Grundlage der anderen Attribute eines Beispiels mit Hilfe von vorher zu diesem Zweck gelernten Modellen. Der RM-Ansatz erwies sich dabei im Rahmen der von Saar-Tsechansky und Provost betriebenen Untersuchungen sowohl DBI als auch PVI gegenüber als deutlich überlegen. Zu Gute kommt den RM den Daten der Autoren zu Folge dabei vor allem seine Unempfindlichkeit gegenüber der Charakteristik der Datensätze.

Die Autoren stellen hierzu (der Frage der Eignung der eingesetzten Verfahren für bestimmte Klassen von Datensätzen) auch einige grundlegende theoretische Überlegungen an. Hierfür charakterisierten sie Datensätze mit unvollständig gegebenen Attributen anhand des Konzepts der Feature-Imputability (FI) – der Genauigkeit, mit der sich der Wert eines Attributs aus den anderen Attributen eines Beispiels vorhersagen lässt. DBI und PVI unterscheiden sich nun darin, wie sie statistische Abhängigkeiten zwischen einzelnen Features ausnutzen. Eine hohe FI impliziert, dass PVI mit hoher Wahrscheinlichkeit den »richtigen« Wert einsetzt, ein damit gelerntes Modell sollte also optimal sein. Hingegen kann DBI keinen Nutzen aus den Abhängigkeiten zwischen den Attributen ziehen und sollte daher einen größeren Fehler begehen. Hingegen wirkt sich PVI bei einer sehr geringen FI im Allgemeinen negativ aus – aufgrund der tatsächlich nur schwachen Korrelation zwischen den Attributen ist der vom PVI-Modell vorhergesagte Wert mit relativ hoher Wahrscheinlichkeit falsch. In diesem Fall dürfte es im Allgemeinen vorteilhafter sein, den fehlenden Wert stattdessen auf Grundlage der vorhandenen Attributwerte der anderen Beispiele zu schätzen, weshalb auf solchen Datensätzen DBI im Vorteil ist. RM hat den Untersuchungen von Saar-Tsechansky und Provost zufolge indessen das Potential, die Vorteile beider Ansätze zu kombinieren: bei hoher FI ist die Information über das fehlende Attribut implizit in den anderen Attributen enthalten, weshalb dem reduzierten Modell durch den Verzicht auf das betreffende Attribut keine Information verloren geht;hingegen kann der RM-Ansatz auch bei geringer FI immer noch von seiner vergleichsweise geringen Varianz profitieren ([STP-07], S.14).

Auch wenn sich die beschriebenen Ergebnisse möglicherweise nicht uneingeschränkt vom C4.5- auf den SeCo-Lerner übertragen lassen, erscheint doch eine solche Erprobung insbesondere insofern interessant, als dass die RM gerade das auch hier beobachtete Problem der relativ starken Datensatzsensibilität zu lösen verspricht, ohne dafür Kompromisse bei der erzielbaren Genauigkeit machen zu müssen.

7 LITERATUR

[**BF-96**] Ivan Bruha, Frantisek Franek. »Comparison of various routines for unknown attribute value processing: The covering paradigm«, *International Journal of Pattern Recognition and Artificial Intelligence*, 10(8): S.939–955, 1996

[**CN-89**] P. Clark, T. Niblett. »The CN2 induction algorithm«, *Machine Learning*, 3(4): S.261-283, 1989

[**Dem-06**] Janez Demšar. »Statistical Comparisons of Classifiers over Multiple Data Sets«, *Journal of Machine Learning Research*, 7: S.1–30, 2006

[**Für-99**] Johannes Fürnkranz. »Separate-And-Conquer Rule Learning«, *Artificial Intelligence Review*, 13(1): S.3-54, 1999

[**GLF-08**] Dragan Gamberger, Nada Lavrač, Johannes Fürnkranz. »Handling Unknown and Imprecise Attribute Values in Propositional Rule Learning: A Feature-Based Approach«, *Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence (PRICAI-08)*: S.636-645, Springer-Verlag 2008

[**ID-80**] R. L. Iman, J. M. Davenport. »Approximations of the critical region of the Friedman statistic«, *Communications in Statistics*: S.571–595, 1980

[**Qui-89**] J. R. Quinlan. »Unknown Attribute Values in Induction«, *Proceedings of the Sixth International Workshop on Machine Learning (ML-1989)*: S.164–168, 1989

[**STP-07**] Maytal Saar-Tsechansky, Foster Provost. »Handling Missing Values when Applying Classification Models«, *Journal of Machine Learning Research*, 1: S.1-48, 2007

8 Appendix

[A] Berechnung des MAV-Score

Die Berechnung des MAV-Scores für einen Datensatz erfolgt in mehreren Schritten.

(1) Bestimmung des »*average merit*« für alle Attribute mit Hilfe eines »χ^2-Evaluators« im Rahmen der Erstellung eines Attribut-Rankings. (→ *absMerit(i)*)

(2) Umwandlung der auf diese Weise erhaltenen absoluten *merits* in relative (Division durch die *merit*-Summe) – zwecks besserer Vergleichbarkeit der Werte zwischen verschiedenen Datensätzen mit möglicherweise stark unterschiedlicher Anzahl an Attributen (→ *relMerit(i)*)

(3) Bestimmung der Anteile an unbekannten Werten für alle Attribute (→ *missRate(i)*)

(4) Daraus berechnet sich der MAVScore eines Datensatzes dann einfach per:

$$MAVScore = 100 \cdot \sum_{i=1}^{n} (relMerit(i) \cdot missRate(i)) \quad \text{(wobei } n := \#Attribute\text{)}$$

[B] Datensätze mit fehlenden Attributwerten

Liste der 24 verwendeten Datensätze mit unbekannten Attributwerten, mit zugehörigem MAV-Score gemäß [A]. Auf einigen Datensätzen ließ sich nicht mit allen Routinen ein Modell lernen – die betroffenen Routinen sind in der letzten Spalte angeführt.

arff-Datei	relation	MAV-Score	Kategorie	nicht geeignet für
audiology	*audiology*	6,76	**Medium**	Delete, DBI
auto-mpg	*auto-mpg*	0,24	**Low**	
autos	*autos*	1,78	**Low**	
breast-cancer	*breast-cancer-data*	0,52	**Low**	
breast-w	*wisconsin-breast-cancer*	0,29	**Low**	
breast-w-d	*wisconsin-breast-cancer-discretized*	0,30	**Low**	
bridges2	*bridges2*	4,54	**Medium**	NN
cleveland-heart-disease	*cleveland-14-heart-disease*	0,34	**Low**	
colic	*horse-colic*	17,65	**High**	
colic.ORIG	*horse-colic.ORIG*	21,76	**High**	
credit	*credit*	0,33	**Low**	
credit-a	*credit-rating*	0,29	**Low**	
echocardiogram	*echo*	1,85	**Low**	NN

arff-Datei	relation	MAV-Score	Kategorie	nicht geeignet für
heart-h	*hungarian-14-heart-disease*	17,17	**High**	DELETE
hepatitis	*hepatitis-domain*	8,13	**Medium**	
hypothyroid	*hypothyroid*	10,80	**Medium**	DELETE
labor	*labor-neg-data*	36,36	**High**	DELETE
labor-d	*labor-discretized*	36,43	**High**	DELETE
mushroom	*mushroom*	1,00	**Low**	
primary-tumor	*primary-tumor*	8,38	**Medium**	
sick-euthyroid	*sick-euthyroid*	16,62	**High**	
soybean	*soybean*	9,80	**Medium**	
vote	*congress-voting-1984*	4,53	**Medium**	
vote-1	*congress-voting-1984*	4,93	**Medium**	

Zusätzlich zu den oben aufgeführten standen noch zwei weitere Datensätze zur Verfügung, die sich als Duplikate anderer Datensätze erwiesen:

- horse-colic.arff, identisch zu colic.arff und
- house-votes.arff, identisch zu vote.arff

[C] DATENSÄTZE OHNE FEHLENDE ATTRIBUTWERTE

Von den 31 zur Verfügung stehenden Datensätzen ohne fehlende Attributwerte wurden nur die drei umfangreichsten wie in Kapitel 3.2 beschrieben aufbereitet. Nachfolgend sind die Detailinformationen zu diesen Datensätzen aufgelistet.

arff-Datei	relation	#num. Att.	#nom. Att.	entfernte Attribute
credit-g	german_credit (dt. Kreditdaten)	7	13	1 (checking_status), 2 (duration), 3 (credit_history)
krkp	kr-vs-kp (Schachendspiel: König+Turm gegen König+Bauer)	0	37	10 (bxqsq), 21 (rimmx), 32 (wkna8)
segment	segment (Bildsegmentierungs-daten)	19	0	10 (intensity-mean), 11 (rawred-mean), 19 (hue-mean)

[D] Einzelergebnisse für alle Datensätze mit fehlenden Attributwerten

I » Ergebnisse mit Laplace-Heuristik

	any	common	delete	ignore	special	dbi.05	hp.C	nn.9	
auto-mpg	79,15	78,39	81,41	79,15	79,15	78,89	77,89	79,90	
credit-a	78,55	78,55	78,84	78,26	78,26	78,12	79,13	78,26	
breast-w	96,14	94,42	95,71	94,28	95,14	94,85	95,28	94,99	
breast-w-d	95,99	94,13	95,57	94,56	95,42	94,71	95,28	94,13	
credit	81,63	83,06	83,06	81,43	80,20	81,43	82,86	81,63	**LOW**
cleveland-heart-disease	72,61	72,61	72,61	71,29	73,93	73,27	70,96	71,62	
breast-cancer	70,63	69,58	66,78	71,33	68,53	69,93	69,58	68,53	
mushroom	100,00	100,00	81,88	100,00	100,00	100,00	100,00	100,00	
autos	81,95	82,93	77,07	82,44	81,95	80,00	81,46	81,95	
echocardiogram	67,57	60,81	54,05	55,41	63,51	58,11	60,81		
vote	94,71	94,02	94,25	93,33	94,25	94,94	94,71	94,48	
bridges2	59,05	58,10	60,00	55,24	60,00	63,81	60,95		**MEDIUM**
vote-1	89,43	87,59	83,68	86,21	88,28	88,97	89,66	88,51	
audiology	75,66	75,22		73,89	73,01		76,99	73,45	
hepatitis	76,77	76,77	79,35	77,42	74,84	80,65	78,71	80,00	
primary-tumor	28,02	30,68	27,43	32,45	30,68	29,79	31,86	34,51	
soybean	89,02	90,19	72,77	90,78	90,78	73,79	86,38	88,43	
hypothyroid	98,55	98,26		98,39	98,26	96,93	98,61	97,95	
sick-euthyroid	95,61	96,08	90,74	95,45	95,54	90,61	96,21	96,14	
heart-h	71,43	73,13		72,79	74,15	72,45	72,11	71,77	**HIGH**
colic	73,10	72,28	63,04	70,65	68,48	72,01	78,80	74,18	
colic.orig	67,66	70,11	36,96	75,27	75,54	72,55	68,21	62,23	
labor	75,44	87,72		91,23	91,23	59,65	87,72	85,96	
labor-d	80,70	87,72		87,72	84,21	80,70	85,96	89,47	

Tabelle 6: *Genauigkeiten der Standard-Routinen auf allen Datensätzen*

	dbi.01	dbi.05	dbi.10	dbi.25	dbi.40	dbi.60	dbi.95	nn.3	nn.5	nn.9	nn.15	
auto-mpg	78,14	78,89	78,64	78,89	78,39	78,39	78,39	79,15	79,90	79,90	79,90	
credit-a	78,84	78,12	77,68	77,39	77,97	77,97	77,25	80,14	79,71	78,26	76,96	
breast-w	95,42	94,85	94,71	94,85	94,99	94,85	94,85	95,57	95,14	94,99	94,99	
breast-w-d	94,56	94,71	93,99	94,42	94,42	94,42	95,57	94,85	94,13	94,13	94,71	
credit	82,24	81,43	82,86	83,67	81,63	81,63	83,47	82,04	80,82	81,63	82,24	**LOW**
cleveland-heart-disease	71,62	73,27	73,60	72,94	72,61	72,28	72,94	72,28	72,61	71,62	73,93	
breast-cancer	68,53	69,93	69,23	70,28	67,83	67,83	66,78	68,53	68,53	68,53	69,23	
mushroom	100,00	100,00	100,00	100,00	100,00	94,68	81,88	100,00	100,00	100,00	100,00	
autos	80,49	80,00	79,02	80,49	80,49	80,98	80,98	80,49	78,05	81,95	82,93	
echocardiogram	58,11	58,11	55,41	52,70	55,41	58,11	58,11					
vote	95,63	94,94	95,86	94,48	94,48	93,56	94,25	94,71	93,56	94,48	94,02	
bridges2	64,76	63,81	63,81	53,33	59,05	58,10	57,14					**MEDIUM**
vote-1	89,43	88,97	87,13	88,28	86,67	86,44	83,91	88,28	88,05	88,51	90,57	
audiology								78,32	77,43	73,45	71,68	
hepatitis	82,58	80,65	85,16	84,52	83,87	81,94	83,23	80,65	80,65	80,00	81,94	
primary-tumor	28,91	29,79	32,15	33,33	27,73	24,19	26,84	33,33	31,56	34,51	28,61	
soybean	75,11	73,79	72,91	72,77	72,77	72,77	72,77	89,31	89,17	88,43	87,85	
hypothyroid	97,06	96,93	96,93	97,03	97,09	96,78	96,46	98,07	97,85	97,95	97,53	
sick-euthyroid	90,74	90,61	90,61	90,61	90,55	90,55	90,58	96,08	95,92	96,14	95,70	
heart-h	72,11	72,45	72,45	60,20	59,18	53,06	41,84	71,43	71,09	71,77	72,45	**HIGH**
colic	72,01	72,01	70,38	66,58	63,32	72,01	72,01	71,74	71,20	74,18	69,02	
colic.orig	69,02	72,55	71,74	64,95	61,14	61,41	60,87	62,77	62,50	62,23	62,77	
labor	66,67	59,65	45,61	61,40	57,89	54,39	35,09	84,21	82,46	85,96	71,93	
labor-d	70,18	80,70	77,19	59,65	45,61	45,61	35,09	84,21	84,21	89,47	89,47	

Tabelle 7: *einzelne Genauigkeiten für alle DBI- und NN-Varianten*

	hp.C	hp.C.01	hp.C.02	hp.C.05	hp.C.10	hp.C.20	hp.C.50	hp.s	hp.S.01	hp.S.02	hp.S.05	hp.S.10	hp.S.20	hp.S.50	
auto-mpg	77,89	77,39	77,39	78,89	80,15	79,40	76,88	78,39	77,89	77,89	78,89	79,90	79,40	77,39	
credit-a	79,13	80,00	78,55	78,41	77,97	78,99	78,70	80,14	80,87	79,71	78,99	79,42	78,70	80,29	
breast-w	95,28	95,57	95,57	95,57	95,57	95,57	94,71	95,42	95,57	95,57	95,57	95,57	95,57	95,42	
breast-w-d	95,28	95,28	95,28	95,28	95,28	95,28	95,28	93,99	93,99	93,99	93,99	93,99	93,99	93,99	
credit	82,86	82,04	82,04	83,27	82,86	82,24	85,31	82,24	83,88	81,84	83,88	81,22	82,04	83,47	**LOW**
cleveland-heart-disease	70,96	70,96	70,96	71,29	73,60	73,60	71,95	73,27	72,94	72,94	72,28	73,60	73,27	73,27	
breast-cancer	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	69,58	
mushroom	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	100,00	
autos	81,46	81,95	82,93	83,41	80,98	82,44	82,44	80,49	80,49	81,46	81,95	80,49	82,44	83,41	
echocardiogram	60,81	59,46	66,22	68,92	66,22	63,51	67,57	62,16	64,86	62,16	70,27	67,57	63,51	66,22	
vote	94,71	94,71	94,71	94,71	94,71	94,71	94,71	94,25	94,25	94,25	94,25	94,25	94,25	94,25	
bridges2	60,95	61,90	61,90	61,90	61,90	61,90	56,19	57,14	57,14	57,14	57,14	56,19	57,14	55,24	**MEDIUM**
vote-1	89,66	89,66	89,66	89,66	89,66	89,66	89,66	87,82	87,82	87,82	87,82	87,82	87,82	87,82	
audiology	76,99	76,99	76,99	76,99	76,99	76,99	76,99	75,22	75,22	75,22	75,22	75,22	75,22	75,22	
hepatitis	78,71	78,71	78,71	76,77	80,00	80,00	77,42	81,94	81,94	81,94	81,94	84,52	80,65	81,29	
primary-tumor	31,86	31,86	31,86	31,86	31,86	31,86	31,86	30,68	30,68	30,68	30,68	30,68	30,68	30,68	
soybean	86,38	86,38	86,38	86,38	86,38	86,38	86,38	90,63	90,63	90,63	90,63	90,63	90,63	90,63	
hypothyroid	98,61	98,67	98,58	98,74	98,58	98,61	98,55	97,76	97,88	97,76	97,69	97,69	97,98	97,60	
sick-euthyroid	96,21	96,11	96,30	96,21	96,36	96,17	95,89	96,05	96,14	95,92	96,36	96,08	96,74	95,92	
heart-h	72,11	72,79	72,79	72,79	73,13	75,17	70,07	73,81	74,49	74,49	74,49	74,49	74,83	71,77	**HIGH**
colic	78,80	79,08	79,08	80,16	80,71	79,08	79,62	77,99	78,26	78,26	79,35	80,16	79,08	79,35	
colic.orig	68,21	68,48	68,48	68,48	67,66	67,66	67,39	68,21	68,48	68,48	68,48	67,66	67,66	67,39	
labor	87,72	87,72	87,72	91,23	89,47	85,96	84,21	84,21	84,21	84,21	84,21	80,70	84,21	78,95	
labor-d	85,96	85,96	85,96	85,96	85,96	85,96	85,96	84,21	84,21	84,21	84,21	84,21	84,21	84,21	

Tabelle 8: *einzelne Genauigkeiten sämtlicher HP-Varianten*

II » Ergebnisse mit m-estimate-Heuristik

	anyvalue	common	delete	ignore	special	DBI.05	HP	nn.9	
auto-mpg	76,13	77,89	78,64	79,40	79,15	80,40	77,39	79,40	
credit-a	82,61	84,49	83,33	83,19	84,06	82,90	83,48	84,35	
breast-w	95,71	95,14	95,14	95,42	95,42	94,56	95,99	94,28	
breast-w-d	95,57	95,28	95,42	95,28	95,71	95,42	95,71	95,57	
credit	82,04	83,88	82,65	87,14	84,08	81,63	84,69	83,27	**LOW**
cleveland-heart-disease	75,25	80,20	76,90	78,55	76,90	78,88	77,56	76,90	
breast-cancer	70,28	70,28	71,68	70,28	70,63	69,23	70,63	69,93	
mushroom	100,00	100,00	81,88	100,00	100,00	100,00	100,00	100,00	
autos	78,05	81,95	72,20	80,49	78,54	74,15	76,10	80,00	
echocardiogram	72,97	63,51	64,86	56,76	55,41	63,51	62,16		
vote	94,25	94,48	94,94	94,48	94,71	95,40	95,63	95,17	
bridges2	59,05	64,76	57,14	62,86	61,90	64,76	60,95		**MEDIUM**
vote-1	89,20	89,89	88,74	89,20	90,34	89,20	89,20	91,26	
audiology	81,86	83,63		82,74	84,07		82,30	78,76	
hepatitis	75,48	74,19	78,71	80,00	78,06	82,58	81,29	84,52	
primary-tumor	36,28	37,17	35,10	37,76	38,64	41,00	37,76	42,18	
soybean	92,09	93,70	75,11	94,29	94,14	78,04	87,26	92,53	
hypothyroid	98,89	98,83		98,89	98,86	97,09	98,96	98,70	
sick-euthyroid	96,93	97,25	90,74	97,00	97,03	90,55	97,06	97,31	
heart-h	76,87	75,85		76,87	75,85	78,23	74,83	73,81	**HIGH**
colic	74,73	74,73	63,04	81,25	76,09	82,34	85,87	75,54	
colic.orig	64,95	77,45	51,09	77,72	78,53	50,00	69,84	62,77	
labor	73,68	82,46		85,96	85,96	57,89	75,44	87,72	
labor-d	80,70	80,70		82,46	78,95	80,70	89,47	91,23	

Tabelle 9: *einzelne Genauigkeiten der Standard-Routinen unter Verwendung der m-estimate-Heuristik*