Hochschule Wismar
University of Technology, Business and Design
Fachbereich Wirtschaft

Hochschule Wismar
University of Technology, Business and Design
Faculty of Business

Christian Andersch, Jürgen Cleve

Data Mining auf Unfalldaten

Heft 01 / 2006

W D P

Wismarer Diskussionspapiere / Wismar Discussion Papers

Der Fachbereich Wirtschaft der Hochschule Wismar, University of Technology, Business and Design bietet die Präsenzstudiengänge Betriebswirtschaft, Management sozialer Dienstleistungen, Wirtschaftsinformatik und Wirtschaftsrecht sowie die Fernstudiengänge Betriebswirtschaft, International Management, Krankenhaus-Management und Wirtschaftsinformatik an. Gegenstand der Ausbildung sind die verschiedenen Aspekte des Wirtschaftens in der Unternehmung, der modernen Verwaltungstätigkeit im sozialen Bereich, der Verbindung von angewandter Informatik und Wirtschaftswissenschaften sowie des Rechts im Bereich der Wirtschaft.
Nähere Informationen zu Studienangebot, Forschung und Ansprechpartnern finden Sie auf unserer Homepage im World Wide Web (WWW): http://www.wi.hs-wismar.de/.

Herausgeber: Prof. Dr. Jost W. Kramer
Fachbereich Wirtschaft
Hochschule Wismar
University of Technology, Business and Design
Philipp-Müller-Straße
Postfach 12 10
D – 23966 Wismar
Telefon: ++49/(0)3841/753 441
Fax: ++49/(0)3841/753 131
e-mail: j.kramer@wi.hs-wismar.de

Vertrieb: HWS-Hochschule Wismar Service GmbH
Phillipp-Müller-Straße
Postfach 12 10
23952 Wismar
Telefon:++49/(0)3841/753-574
Fax: ++49/(0)3841/753-575
e-mail: info@hws-startupfuture.de
Homepage: www.hws-startupfuture.de

ISSN 1612-0884
ISBN 3-910102-87-5

JEL-Klassifikation C80, Z00

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Jährlich sterben etwa 1,2 Millionen Menschen weltweit im Straßenverkehr, weitere 50 Millionen werden zum Teil schwer verletzt [10]. Auch Mecklenburg-Vorpommern ist davon betroffen, und es ist dringend erforderlich, diese Anzahl zu reduzieren.

Ein Weg dorthin ist die Analyse gespeicherter Unfalldaten. Das vorliegende Dokument beschreibt solch eine Analyse von Verkehrsunfalldaten aus dem Großraum Rostock. Die Untersuchungen erfolgten im Rahmen der Veranstaltung *Data Mining* an der Hochschule Wismar. Schwerpunkt ist die Vorhersage der Unfallschwere in der Art *Wenn an diesem Ort ein Unfall geschähe, wie schwer wäre er?*

Es wird davon ausgegangen, dass bei ansonsten gleichen Bedingungen ein Unfall ungefähr den gleichen Schweregrad erreichen würde. Dieser Schweregrad wird als natürliche Zahl entsprechend Tab. 1 als Score auf einer Skala von 1 bis 9 angegeben (vgl. [2]). Personenschäden haben dabei Priorität; so hat ein Unfall mit 2 Leichtverletzten und 30.000 € Sachschaden den Score 4. Eine Score-Vorhersage könnte während der Fahrt fahrerabhängige Gefahrenschwerpunkte ermitteln und so das Unfallrisiko und damit Sach- und Personenschäden reduzieren. Auch andere Untersuchungen zeugen von der Wichtigkeit dieses Themas.[1]

Die verwandte Frage *Wie groß ist die Wahrscheinlichkeit eines Unfalls an diesem Ort?* kann hier aufgrund fehlender Angaben über die „Befahrenheit“ des Ortes nicht beantwortet werden – es liegen keine Daten ohne Unfall vor.

Tabelle 1: Unfallscores

Sachschaden [€]	Personenschaden	Score
$[0, 250)$	keiner	1
$[250, 500)$	keiner	2
$[500, 2\,000)$	keiner	3
$[2\,000, 5\,000)$	nur Leichtverletzte	4
$[5\,000, 10\,000)$	höchstens 2 Schwerverletzte	5
$[10\,000, 25\,000)$	mehr als 2 Schwerverletzte	6
$[25\,000, 50\,000)$	1 Toter	7
$\geq 50\,000$	mehrere Tote und Schwerverletzte	8
Massenkarambolage mit mindestens 10 Fahrzeugen und/oder mindestens 5 Schwerverletzten		9

Quelle: [6].

[1] Siehe „Improving road safety with data mining“, `http://soleunet.ijs.si/website/html/cocasesolutions.html`

1.2 Datengrundlage

Die Daten in Form einer Excel-Tabelle umfassen 10.813 anonymisierte Datensätze (Zeilen) mit 52 Attributen (Spalten) inkl. Score. Sie sind ein mehrjähriger Auszug der amtlichen Unfallstatistik aus dem Großraum Rostock, ergänzt durch weitere Attribute von [2]. Zu den 52 Attributen gehören u. a.:

- Art des Regelverstoßes, z. B. unerlaubte Geschwindigkeit,
- Merkmale des äußeren Milieus (der Umgebung), z. B. Dunkelheit,
- Altersklasse und Geschlecht des Unfallverursachers,
- Art und Anzahl der am Unfall beteiligten Personen und Fahrzeuge,
- Art und Anzahl der Verletzten.

Die genaue Bedeutung der einzelnen Attribute und ihrer Ausprägungen ist in Tab. 15 beschrieben. 36 der 52 Attribute sind binär und von den 8 Attributen der Art des Regelverstoßes ist immer exakt eines gesetzt. Der Score wurde ursprünglich entsprechend Tab. 1 berechnet, die dafür benötigte Höhe des Sachschadens ist in den gegebenen Daten jedoch nicht enthalten. Bei 194 Datensätzen (1,79%) sind Altersklasse und Geschlecht wegen Fahrerflucht nicht gegeben.

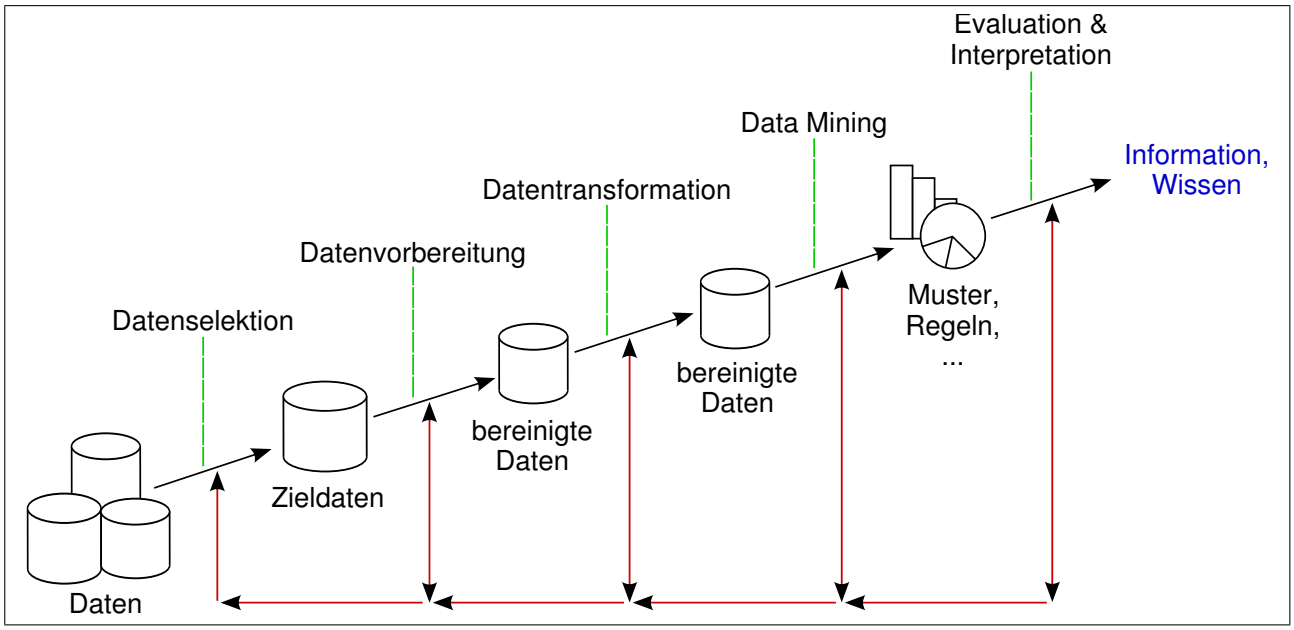
1.3 Data Mining

Data Mining bezeichnet das Umwandeln von implizitem zu explizitem Wissen. So können automatisch große Datenbanken z. B. nach Zusammenhängen zwischen Attributen oder Ähnlichkeiten von Datensätzen durchsucht und die Ergebnisse dem Benutzer präsentiert werden. Oft werden vorher unbekannte Zusammenhänge gefunden, die bei manueller Analyse nicht entdeckt worden wären.

Data Mining erfolgt in mehrstufigen Prozessen mit eher experimentellem Charakter. Um diese zu formalisieren, wurden diverse Prozessmodelle entwickelt, z. B. das inkrementelle Modell von [5], CRISP-DM [4] oder das 6-stufige DMKD [7]. Sie unterscheiden sich in Anzahl, Inhalt und den Verbindungen der einzelnen Stufen.

In diesem Projekt wurde nach dem in Abb. 1 dargestellten inkrementellen Modell vorgegangen. Darauf sei hier nur kurz eingegangen, Details dazu finden sich z. B. in [11]. In dessen eigentlicher Data-Mining-Stufe wird basierend auf Trainingsdaten gelernt und ein „Modell“ erstellt. Soll dieses auch zur Vorhersage eingesetzt werden, muss es auf ungelernten Daten evaluiert werden. Die

Abbildung 1: Inkrementelles Modell im Data Mining



Quelle: [5].

Fähigkeit zur Vorhersage unbekannter Daten wird Generalisierungsfähigkeit genannt.

Ein Problem beim Lernen ist das Overfitting, eine Überanpassung. Ab einem bestimmten Punkt führt weiteres Lernen auf den Trainingsdaten zu schlechterer Erkennung auf Testdaten. Da bei Vorhersage die Erkennung der Testdaten Priorität hat, ist ein Kompromiss zwischen *Lernen* und *Nicht-Lernen* einzugehen. Zur Aufteilung in Trainings- und Testdaten existieren verschiedene Varianten:

- **Test auf Trainingsdaten:** Hiermit kann überprüft werden, wie gut die Trainingsdaten gelernt wurden. In den nächsten Kapiteln meint Test auf Trainingsdaten immer Training und Test auf allen 10.813 Datensätzen, wenn nicht anders angegeben.
- Die $\frac{2}{3}$**-Regel** bezeichnet eine oft verwendete Aufteilung in $\frac{2}{3}$ Trainings- und $\frac{1}{3}$ Testdaten.
- **Crossvalidation:** Mittels der Kreuzvalidierung wird mehrfach in Test- und Trainingsdaten aufgeteilt und der Fehler gemittelt. Üblich ist 10-fach, nachfolgend mit CV10 abgekürzt.

Einige Algorithmen benutzen zur Vermeidung von Overfitting intern Crossvalidation. Weiterhin kann durch gezielte Verkleinerung des Modells („Pruning“) die Generalisierungsfähigkeit erhöht werden.

1.4 Weka

Während kommerzielle Data-Mining-Produkte wie der IBM Intelligent Miner[2] oder Lösungen von Prudsys[3] auf Performance und einfache Integration in die IT-Architektur optimiert sind, bieten freie Systeme wie Weka [12] oft spezielle oder eine große Auswahl an Algorithmen inkl. Zugriff auf ihre Parameter.

Weka ist eine javabasierte Sammlung von Algorithmen mit einem GUI-Aufsatz. Die Bedienung erfolgt per Kommandozeile, „Explorer“ oder Knowledge Flow. Der Explorer wie in Abb. 2 bietet Vorverarbeitung, Data-Mining-Algorithmen und Visualisierung unter einer Oberfläche. Im Knowledge Flow werden einzelne Data-Mining-Schritte visuell als Graph angeordnet. Automatisieren lassen sich Schritte jedoch am besten durch kombinierte Aufrufe per Kommandozeile. Datenimport in Weka kann über CSV-Dateien[4], eine Datenbankverbindung mittels SQL[5] oder Wekas eigenes Dateiformat ARFF[6] erfolgen.

Durch die Verfügbarkeit als Sourcecode und das einheitliche Format zur Verwendung der Algorithmen benutzen auch andere Programme Wekas Algorithmen, z. B. QuDA[7] oder YALE[8].

2 Methoden und Verfahren

2.1 Assoziationen

Mittels der Assoziationsanalyse kann herausgefunden werden, welche Attributkombinationen interessante Zusammenhänge darstellen. Diese ist verwandt mit der Korrelationsanalyse; Maße zur Angabe der Interessantheit sind aus der Statistik bekannt. Solche Maße sind:

- Support: $\text{supp}(A \to B) = P(A \cup B)$
- Konfidenz: $\text{conf}(A \to B) = P(B|A)$
- Lift: $\text{lift}(A \to B) = \frac{\text{supp}(A \to B)}{\text{supp}(A) \cdot \text{supp}(B)} = \frac{\text{conf}(A \to B)}{\text{supp}(B)}$

Assoziationsregeln werden die Attributkombinationen genannt, die vorgegebene Interessantheitsmaße erfüllen. Man kann die Regeln mit Implikationsregeln

[2] Siehe `http://www.software.ibm.com/data/iminer/`

[3] Siehe `http://www.prudsys.de/`

[4] Comma Separated Values

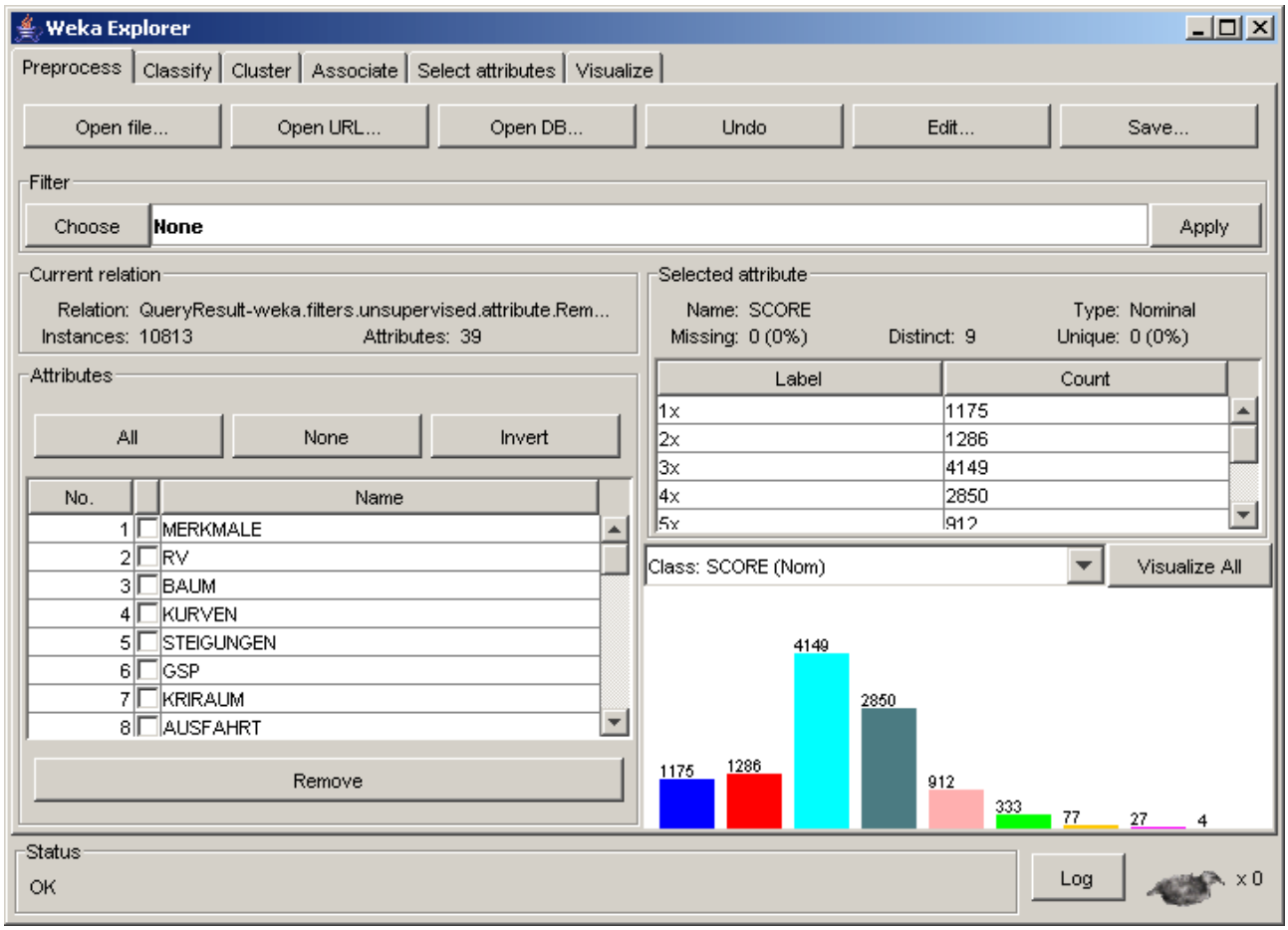
[5] Structured Query Language

[6] Attribute-Relation File Format. ARFF entspricht CSV mit zusätzlichen Informationen über Attribute mit ihren Ausprägungen.

[7] Siehe `http://kirk.intellektik.informatik.tu-darmstadt.de/~quda/`

[8] Siehe `http://www-ai.cs.uni-dortmund.de/SOFTWARE/YALE/`

Abbildung 2: Weka Explorer, Vorverarbeitung



Quelle: Darstellung in Weka mit Unfalldaten.

vergleichen inkl. statistischer Maße darüber: Wenn A und B, dann folgt daraus C in $x\%$ aller Fälle. A, B, C werden Items genannt, $A + B$ Itemset.

Durch Vorgabe der Interessantheitsmaße und Regellänge ist das Ergebnis einer Analyse eindeutig festgelegt. Algorithmen unterscheiden sich daher in Parametern und Ressourcenverbrauch und nicht im Ergebnis. Ein früher Algorithmus zum Finden der Itemsets ist der apriori-Algorithmus [1], welcher zuerst Itemsets der Länge 1 und dann immer längere bewertet.

2.2 Klassifikation

Während bei der Assoziation beliebige Attributkombinationen für ein weiteres beliebiges Zielattribut untersucht werden, wird bei der Klassifikation ein Zielattribut festgelegt und durch die restlichen $n - 1$ Attribute soll es determiniert werden. Dies entspricht einer Funktion

$$f(x_1, \ldots, x_{n-1}) = x_{ziel}$$

2.2.1 Algorithmen

Klassifikations-Algorithmen Unterschiedliche Klassifikations-Algorithmen können sehr unterschiedliche Ergebnisse liefern. Viele Algorithmen basieren auf Entscheidungsbäumen oder benutzen sie. Solche Bäume sind sehr beliebt, weil sie sich in für Menschen gut lesbarer Form darstellen lassen. Abb. 3 zeigt einen einfachen Baum für zwei numerische Attribute x_1, x_2 und das Zielattribut Farbe mit den Ausprägungen `rot` und `blau`.

Abbildung 3: Ein per J48 erstellter Entscheidungsbaum auf 128 Datensätzen

```
x1 <= 0.392539: blau (51.0)
x1 > 0.392539
|   x1 <= 0.619111
|   |   x2 <= 0.596739
|   |   |   x2 <= 0.289417: blau (6.0)
|   |   |   x2 > 0.289417: rot (15.0)
|   |   x2 > 0.596739: blau (9.0)
|   x1 > 0.619111: blau (47.0/2.0)
```

Quelle: Darstellung in Weka mit eigenen Daten.

Graphisch lässt es sich wie in Abb. 4 darstellen. Man erkennt, wie der Raum (hier nur 2D wegen zwei Attributen) geteilt wird. Eine Einteilung des Raumes ist das Ziel aller Klassifikations-Algorithmen. „Normale“ Entscheidungsbäume können den Raum nur mit achsenparallelen Ebenen teilen. Verschiedene Varianten unterscheiden sich darin, wo genau die Ebenen gesetzt werden.

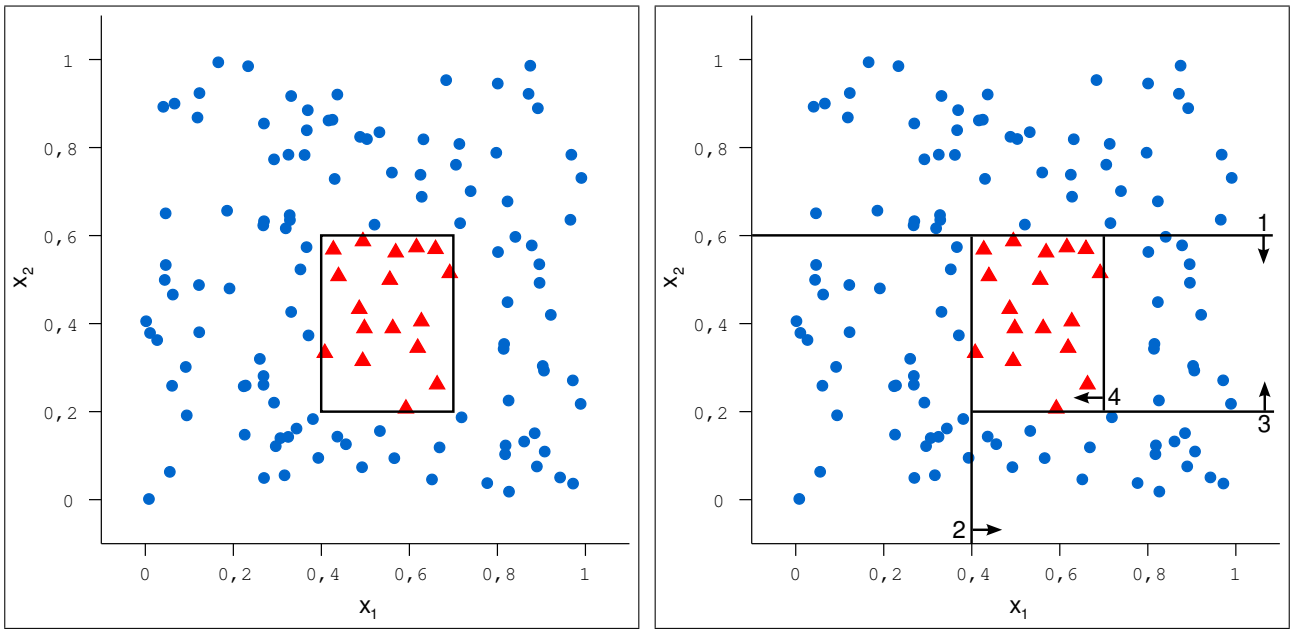
Entscheidungslisten und -tabellen sind ähnlicher Natur, werden aber anders generiert. Die Modelle von Entscheidungsbäumen, -listen und -tabellen können ineinander überführt werden.

Andere Klassifikations-Algorithmen wie bestimmte Neuronale Netze oder die Support Vector Machines (SVM) haben als Bedingung zum Lernen die lineare Separierbarkeit. Auf zwei Klassen bezogen bedeutet dies im zweidimensionalen Fall: Es muss eine Linie durch die Ebene gezogen werden können, die die Klassen genau trennt. Sie muss nicht achsenparallel sein. Nur einfachste Problemstellungen sind im zweidimensionalen Fall linear separierbar. Durch Dimensionserhöhung ergibt sich auch bessere lineare Separierbarkeit. Dies geschieht in SVMs mittels einer „Kernelfunktion“, die die Daten in einen Merkmalsraum transformiert. Gesucht ist somit ein Merkmalsraum, in dem mittels Hyperebenen möglichst optimal getrennt werden kann. Abb. 5 zeigt dies am Beispiel von kreisförmig getrennten Daten. Die Hyperebene wird vektoriell betrachtet von den ihr auf beiden Seiten am dichtesten liegenden Datensätzen unterstützt, daher der Name Support Vector Machine.

Abbildung 4: Klassifikation mittels Entscheidungsbaum

(a) Originaldaten mit den zu klassifizierenden Daten im Bereich $0{,}4 \leq x_1 \leq 0{,}7$ und $0{,}2 \leq x_2 \leq 0{,}6$

(b) Mögliche Entstehung eines Entscheidungsbaums in vier Schritten

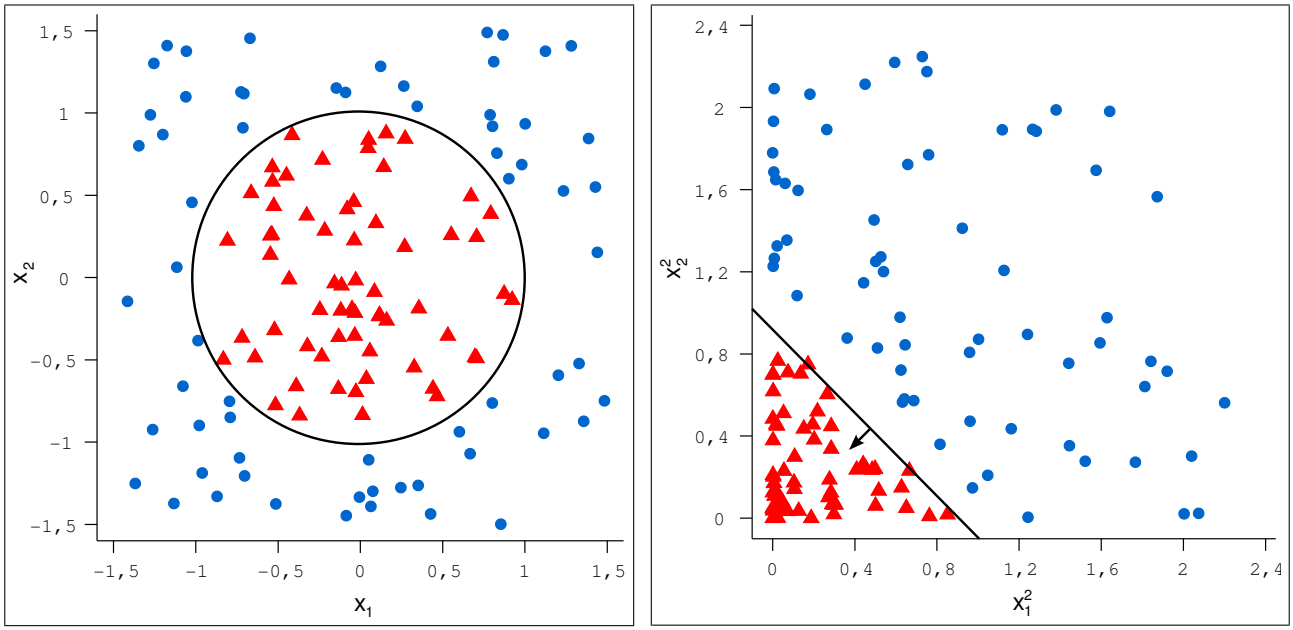


Quelle: Eigene Darstellung.

Abbildung 5: Klassifikation mittels Support Vector Machine

(a) Zu klassifizierende Daten der Art $x_1^2 + x_2^2 \leq 1$ sind in der Originalform nicht linear separierbar

(b) Transformierte Daten mit $x_1 \rightarrow x_1^2, x_2 \rightarrow x_2^2$ sind linear separierbar



Quelle: [9].

Für mehr Informationen zu Entscheidungsbäumen, SVMs und anderen Algorithmen sei [9] empfohlen. Nachfolgend eine kurze Erklärung der im Projekt verwendeten Klassifikations-Algorithmen von Weka.

- **ID3** ist ein einfacher Algorithmus für nominale Daten und ohne Einstellmöglichkeiten. Der Entscheidungsbaum wird sehr groß, da kein Pruning erfolgt und entspricht damit Auswendiglernen.
- **J48** ist Wekas Implementierung von C4.5, Revision 8[9]. Dies ist ein Entscheidungsbaum mit diversen Parametern, die auch Pruning ermöglichen.
- **Random Forest** implementiert einen Wald aus Zufallsbäumen [3].
- **NBTree** erzeugt einen Entscheidungsbaum basierend auf Klassifikationen nach Naive Bayes an den Blättern. Es sind keine Optionen möglich.
- **REPTree** verwendet Regression im Entscheidungsbaum.
- **Conjunctive Rule** ist eine regelbasierte Klassifikation aufbauend auf Konjunktionen.
- **Decision Table** erzeugt eine Entscheidungstabelle.
- **PART** gibt eine Entscheidungsliste aus. Dafür wird ein unvollständiger C4.5-Baum aufgebaut und pro Iteration aus dem jeweils besten Blatt eine Regel generiert.
- **DecisionStump** ist ein Entscheidungs-Stumpf, der die zwei häufigsten Klassen vorhersagt. Er wird meist in Verbindung mit Boosting-Algorithmen eingesetzt.
- **ZeroR** sagt immer den Durchschnitt bzw. Modalwert vorher. Dies eignet sich gut zum Vergleich; jeder andere Algorithmus sollte besser sein.
- **SMO** implementiert Sequential Minimal Optimization, einen Algorithmus für Support Vector Machines [8].

Es wurden die von Weka vorgeschlagenen Standardparameter verwendet.[10] Die höchst mögliche Erkennungsrate wird aufgrund des Auswendiglernens immer von ID3 auf der Trainingsmenge erzielt. So wird überprüft, wie gut die Trainingsmenge gelernt werden kann. Bei Crossvalidation zeigt ZeroR das Mindestmaß, welches von anderen Algorithmen überschritten werden sollte.

[9] Entwickelt von Ross Quinlan, siehe `http://www.rulequest.com/Personal/`

[10] SMO verwendet stattdessen radial basis functions (RBF) mit complexity 1, gamma 0,3

Boosting-Algorithmen Eine weitere Klasse von Algorithmen sind Boosting-Algorithmen, welche mehrere Modelle kombinieren. Dabei wird mindestens ein Klassifikations-Algorithmus als „Basisalgorithmus“ angegeben, dessen Modelle durch das Boosting schrittweise verbessert werden. Der Basisalgorithmus kann auch ein Boosting-Algorithmus sein. Somit sind beliebig komplexe Schachtelungen möglich. Nachfolgend eine Erklärung der benutzten Boosting-Algorithmen:

- **Vote** erlaubt die Angabe mehrerer Basisalgorithmen und macht eine Mehrheitsentscheidung, d. h. der Modalwert oder Durchschnitt wird ausgegeben.
- **AdaBoostM1**, kurz für Adaptive Boosting, benutzt intern Vote für mehrere Modelle vom gleichen Basisalgorithmus. Die später erzeugten Modelle sollen die vorher falsch klassifizierten Datensätze besser erkennen aufgrund ausgesuchter kleinerer Trainingsdaten.
- **Bagging**, kurz für Bootstrap Aggregating, benutzt ebenfalls intern Vote für mehrere Modelle vom gleichen Basisalgorithmus. Die Anzahl der Trainingsdaten bleibt konstant, aber ihre Zusammensetzung wird variiert.

Aufgrund teils vieler Parameter und beliebiger Kombinationen von Algorithmen sind unendlich viele Möglichkeiten für Experimente gegeben. Hier musste sich beschränkt werden. Auch wegen der erhöhten Rechenzeit war immer nur eine Auswahl möglich.

2.2.2 Gütemaße

Um verschiedene Ergebnisse hinsichtlich ihrer Brauchbarkeit zu bewerten, sind Gütemaße notwendig. Die einfachsten Maße sind die für die Gesamterkennung bzw. den Gesamtfehler, auch in Prozent üblich:

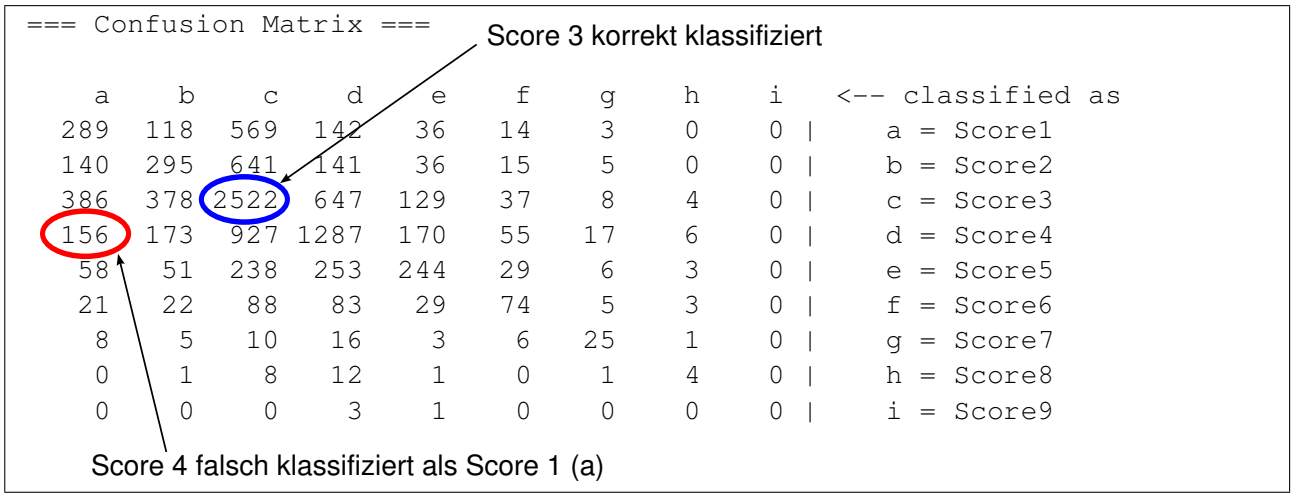
$$\text{Gesamterkennungsrate} = \frac{\sum \text{richtig klassifizierte}}{\text{Gesamtmenge}}$$

$$\text{Gesamtfehlerrate} = 1 - \text{Gesamterkennungsrate}$$

Klassifikationsergebnisse auf der Testmenge werden oft in einer quadratischen Confusion Matrix C dargestellt, in der ablesbar ist, wieviel von welcher Klasse (in diesem Fall Klasse = Score) als welche Klasse klassifiziert wurden. Abb. 6 zeigt ein Beispiel für Weka. Damit lassen sich die Gütemaße Recall, Precision und F-Measure definieren, welche jeweils für einen einzelnen Score-Wert gelten. Recall R (Erkennungsrate) ist ein normierter Anteil, wieviele von Score x auch als Score x klassifiziert wurden. Sei i der Zeilenindex und j der Spaltenindex:

$$\text{Recall}(x) = \frac{c_{x,x}}{\sum_{j=1}^{n} c_{x,j}}$$

Abbildung 6: Confusion Matrix C in Weka für ID3, CV10




```
=== Confusion Matrix ===

   a    b    c    d    e    f    g    h    i   <-- classified as
 289  118  569  142   36   14    3    0    0 |    a = Score1
 140  295  641  141   36   15    5    0    0 |    b = Score2
 386  378 2522  647  129   37    8    4    0 |    c = Score3
 156  173  927 1287  170   55   17    6    0 |    d = Score4
  58   51  238  253  244   29    6    3    0 |    e = Score5
  21   22   88   83   29   74    5    3    0 |    f = Score6
   8    5   10   16    3    6   25    1    0 |    g = Score7
   0    1    8   12    1    0    1    4    0 |    h = Score8
   0    0    0    3    1    0    0    0    0 |    i = Score9
```

Quelle: Darstellung in Weka mit Unfalldaten.

Im Gegenzug gibt Precision die Genauigkeit der Klassifikation an, also wieviele der als Score x klassifizierten auch wirklich Score x sind:

$$\text{Precision}(x) = \frac{c_{x,x}}{\sum_{i=1}^{n} c_{i,x}}$$

Beide entsprechen dem (prozentualen) Anteil des Hauptdiagonalelements an der Reihensumme. F-Measure F wird als „Durchschnitt“ aus Recall R und Precision P folgendermaßen berechnet:

$$\frac{1}{F} = \frac{1}{2}(\frac{1}{P} + \frac{1}{R}) \rightarrow F = 2 \cdot \frac{P \cdot R}{P + R}$$

Viele Data-Mining-Methoden versuchen, diesen Wert zu maximieren. Dies ist insbesondere dann sinnvoll, wenn keine Kostenmatrix verwendet wird. Bei einer Kostenmatrix werden die Elemente in C mit den an gleicher Stelle stehenden Elementen einer gleichartigen Kostenmatrix O multipliziert. Zur Anwendung kam folgende Kostenmatrix:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 2 & 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 1 & 0 & 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 2 & 1 & 0 & 1 & 2 & 3 & 4 \\ 5 & 4 & 3 & 2 & 1 & 0 & 1 & 2 & 3 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & 1 & 2 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 1 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{pmatrix}$$

Auf der Hauptdiagonalen liegen korrekt vorhergesagte Scores, die Kosten von 0 verursachen. Die Kosten bei einer falschen Vorhersage sind hier die Differenz aus

wahrem und vorhergesagtem Score, weswegen die Matrix symmetrisch ist. Die Summe aller Einzelkosten ergibt die Kostenfunktion, welche es zu minimieren gilt:

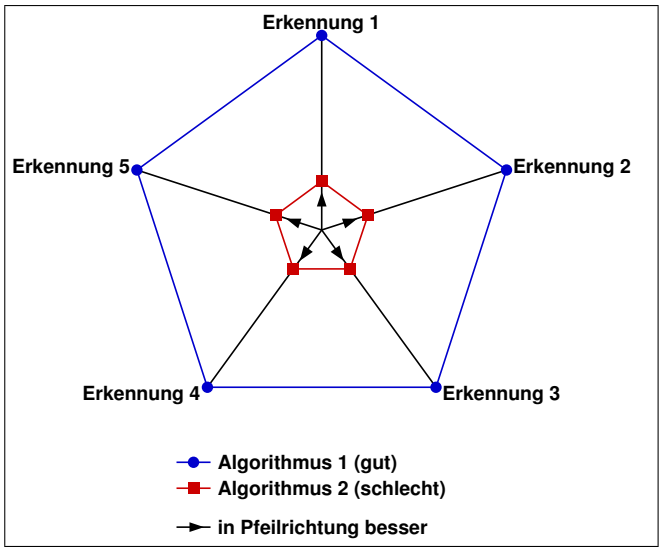
$$\text{Kosten} = \sum_{i=1}^{n} \sum_{j=1}^{n} o_{i,j} \cdot c_{i,j} \rightarrow \min$$

2.2.3 Visualisierung

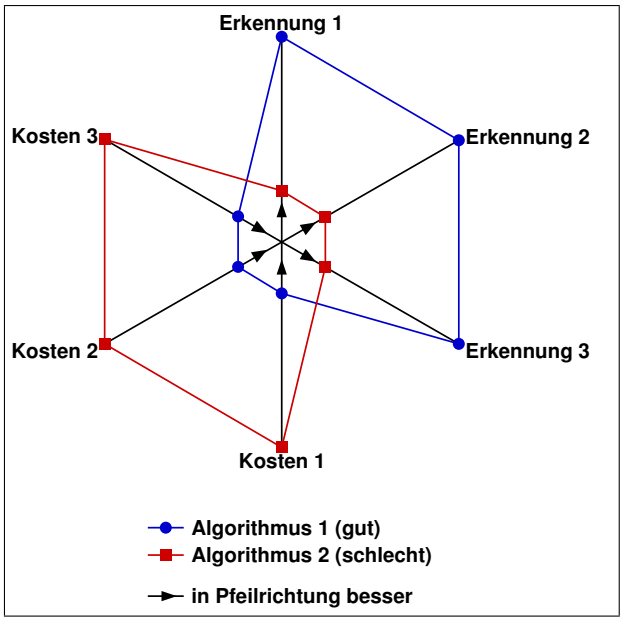
Einige der Tabelleninhalte sind zum besseren Verständnis als relatives Netzdiagramm visualisiert. Achsen sind dabei sternförmig mit nach außen größer werdenden Werten angeordnet. Aufgrund der teils ähnlichen absoluten Zahlen sind auf jeder Achse die relativen Werte abgetragen. Der schlechteste Algorithmus pro Achse ist entweder innen oder außen auf der Achse – ein Pfeil am Achsenbeginn (innen) zeigt die Richtung an für „bessere“ Werte. Somit ist immer ein Wert innen und einer außen. Die restlichen Werte verteilen sich linear auf der Achse. So können die Stärken und Schwächen der Algorithmen je Achse erkannt werden. Abb. 7 zeigt zwei Beispiele.

Abbildung 7: Zwei Beispiele für relative Netzdiagramme

(a) Außen besser



(b) Achsenabhängig innen oder außen besser



Quelle: Eigene Darstellung.

2.3 Clustering

Beim Clustering werden die Datensätze in Gruppen (Cluster) eingeteilt. Ziel ist es, möglichst ähnliche Datensätze in gleiche Cluster zu legen. Anders aus-

gedrückt: Der Unterschied zwischen Datensätzen in unterschiedlichen Clustern soll möglichst groß sein. Somit wird den Daten die Clusternummer als Attribut hinzugefügt. Die Ähnlichkeit zweier Datensätze u und v wird meist aufgrund einer Distanzfunktion ermittelt, z. B. per euklidischer Distanz

$$\mathrm{dist}(v,u) = \sqrt{\sum_{i=1}^{n}(v_i - u_i)^2}$$

Dabei entspricht v_i Attribut Nr. i von Datensatz v. Die Clustereinteilung hängt stark vom verwendeten Algorithmus ab. Einige Algorithmen können die für sie optimale Clusteranzahl selbst bestimmen. Beim hier verwendeten K-Means-Algorithmus muss sie vorgegeben werden. Für Details dazu sei auf [11] verwiesen.

3 Experimente

3.1 Erster Ansatz: Data Mining auf Originaldaten

3.1.1 Vorverarbeitung

Alle Werte wurden nominal bzw. binär codiert: leerer Wert → `0`, `x` → `1`. Fehlende Werte bei Altersklasse/Geschlecht wurden somit auch auf 0 gesetzt.

Entfernt wurden die Attribute cluster, SVT, Ursache, Leichtverletzte, Schwerverletzte, Tote. Diese Daten stehen erst nach einem Unfall zur Verfügung bzw. der Score wird aus ihnen berechnet.

Für die Attribute Merkmale und Fahrzeuge sollte eine Attributselektion in Weka Aussagen über deren Wichtigkeit liefern. Die dafür vorhandenen Algorithmen wie z. B. BestFirst oder GeneticSearch lieferten auf der Trainingsmenge und mit 10-facher Crossvalidation keine eindeutigen Aussagen. Ein Test mit J48 bei reduzierter Attributanzahl ergab eine schlechtere Gesamterkennungsrate. Deshalb wurde für Clustering und Klassifikation mit allen jetzt 46 Attributen weitergearbeitet.

3.1.2 Assoziationen

Ein Ergebnis der Assoziationsanalyse auf den Gesamtdaten zeigt Tab. 2. Es sind nur Regeln mit einem Item dargestellt. Mindestens eines der Maße Support, Konfidenz und Lift muss eine Regel interessant erscheinen lassen. Nachfolgend wird dies diskutiert.

Tabelle 2: Assoziationsregeln

Nr.	Regel A	→	B	absoluter Support A	B	A ∪ B	Support	Konfidenz	Lift
1	ohne FS	→	Alkohol	11	475	11	0,10%	100,00%	22,76
2	Alkohol	→	männlich	475	8 982	453	4,19%	95,37%	1,15
3	Fußg./Radf.	→	Score ≥ 4	837	4 203	680	6,29%	81,24%	2,09
4	Score ≥ 7	→	männlich	108	8 982	95	0,88%	87,96%	1,06
5	Score ≥ 7	→	erl. Geschw.	108	249	90	0,83%	83,33%	36,19
6	Score ≥ 7	→	Alkohol	108	475	15	0,14%	13,89%	3,16

- **Regel Nr. 1:** Die wenigen Fälle von Fahren ohne Führerschein (Support 0,10%) waren alle im Zusammenhang mit Alkohol (Konfidenz 100%). Bei statistischer Unabhängigkeit beider Ereignisse ergäbe sich der Lift 1. In der Gesamtmenge kommen $\frac{475}{10.813} = 4{,}39\%$ aller Unfälle unter Alkoholeinfluss zustande. Dies wäre bei statistischer Unabhängigkeit auch bei Fahren ohne Führerschein zu erwarten. Statt 4,39% entstehen jedoch 100% der Unfälle ohne Führerschein unter Alkoholeinfluss, so dass mit einem Lift von $\frac{100\%}{4{,}39\%} = 22{,}76$ ein übermäßig starker Zusammenhang dargestellt ist.

- **Nr. 2:** Von alkoholisierten, männlichen Fahrern werden 4,19% aller Unfälle verursacht. Die Konfidenz ist groß, der Lift jedoch nur leicht erhöht. Damit stellt diese Regel keinen außergewöhnlichen Zusammenhang dar.

- **Nr. 3:** Wenn Fußgänger oder Radfahrer vom Unfall betroffen sind, beträgt der Score mindestens 4. Diese Regel hat zwar einen hohen Support und einen interessant erscheinenden Lift, aber sie gibt nur die Definition des Scores wieder: Sobald Personenschaden entstanden ist, beträgt der Score mindestens 4.

Die weiteren Regeln beziehen sich auf Unfälle mit Score ab 7, d. h. es gab mindestens einen Toten. Diese betragen rund 1% aller Unfälle und haben damit einen geringen Support, jedoch sind sie wegen ihrer Schwere interessant.

- **Nr. 4:** Diese Regel zeigt, ob insbesondere männliche Fahrer solche Unfälle verursachen. Zwar ist dies zu 87,96% der Fall, jedoch liegt der Lift nur minimal über 1. Somit zeigt diese Regel nichts Unerwartetes.

- **Nr. 5:** Zu 83,33% geschehen die schweren Unfälle bei erlaubter Geschwindigkeit. Der extrem hohe Lift gibt an, dass dies wesentlich öfter ist als in der Gesamtmenge.

- **Nr. 6:** Bei einem schwerem Unfall war der Fahrer alkoholisiert. Die Konfidenz ist gering, allerdings zeigt der Lift, dass dies immer noch dreimal so häufig geschieht wie im Vergleich zur Gesamtmenge.

Eine Bemerkung zum geringen Lift bei männlichen Unfallverursachern sei erlaubt. 83,07% aller Unfälle wurden von Männern verursacht.[11] Diese Aussage bezieht sich auf vorhandene Unfalldaten. Aber: *Verursachen Männer öfter Unfälle als Frauen?* Diese Fragestellung muss sich auf die Verteilung der Geschlechter in einem anderen Maß beziehen, z. B. die Anzahl der Fahrer, Fahrten, Fahrtstunden oder -kilometer. Dafür liegen hier jedoch keine Daten vor und deshalb kann die Frage auch nicht beantwortet werden. Werden z. B. rund 85% aller Fahrtkilometer von Männern gefahren, so verursachen relativ gesehen Männer und Frauen gleich oft Unfälle.

3.1.3 Clustering

Beim Clustering in Weka kann keine Kostenmatrix angegeben werden. Als Optimierungskriterium wurde stattdessen die Rate der falsch geclusterten Datensätze angenommen, welche möglichst minimal sein sollte. Geclustert wurde direkt nach dem Score, Tab. 3 gibt einen Überblick über die Ergebnisse. Dabei wurde SimpleKMeans verwendet mit unterschiedlicher Anzahl an Clustern.

Tabelle 3: Clustering nach Score mit SimpleKMeans

Anzahl Cluster:	2	3	4	5	6	7	8	9
Fehlerrate [%]:	65,3	68,7	72,4	73,1	75,8	74,5	78,0	81,2

Die Fehlerrate steigt mit größerer Clusteranzahl ebenfalls an, außer von Clusteranzahl 6 auf 7. Da selbst das beste Ergebnis bei 2 Clustern mit einer Erkennungsrate von 34,7% unter dem schlechtesten Ergebnis beim Klassifizieren liegt, wurde Clustering nicht weiter verfolgt.

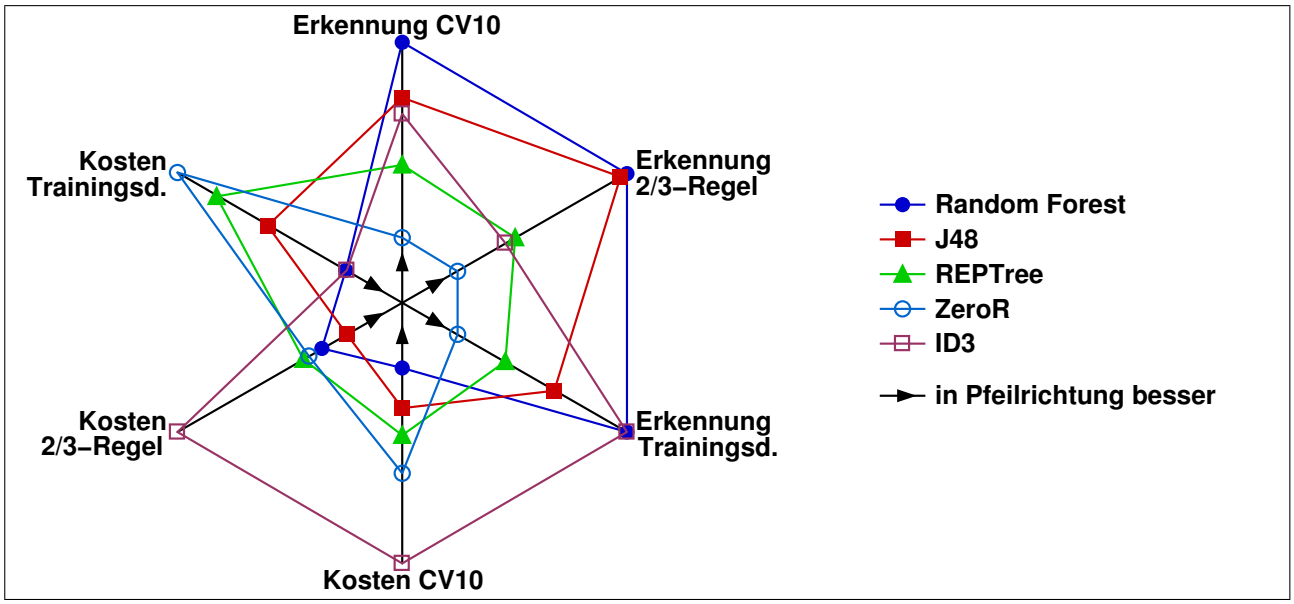
3.1.4 Klassifikation

Die Ergebnisse der Algorithmen sind in Tab. 4 zusammengefasst. Da die Kosten nur bei gleicher Testmenge vergleichbar sind, wurde die Tabelle entsprechend angeordnet, aufsteigend sortiert nach Kosten der Crossvalidation. Eine Visualisierung davon zeigt Abb. 8.

Es fällt auf, dass ID3 und Random Forest bei den Trainingsdaten annähernd gleiche Erkennungsraten und Kosten haben, sich bei Crossvalidation aber stark

[11] Die gleiche Quote bei den Fahrerflüchtigen angenommen ergibt insgesamt 84,56%.

Abbildung 8: Relatives Netzdiagramm der Ergebnisse vom ersten Ansatz



Quelle: Eigene Darstellung.

Tabelle 4: Erster Ansatz – Algorithmen, Erkennungsraten, Kosten

Algorithmus	Gesamterkennungsrate [%]			Kosten		
	Trainingsd.	CV10	$\frac{2}{3}$-Regel	Trainingsd.	CV10	$\frac{2}{3}$-Regel
Random Forest	77,3	46,9	44,4	3 697	9 005	3 208
J48	60,7	44,5	44,2	6 507	9 303	3 100
REPTree	49,2	41,6	41,1	8 352	9 496	3 291
ZeroR	38,4	38,4	39,4	9 776	9 776	3 266
ID3	77,5	43,8	40,8	3 678	10 426	3 858

unterscheiden. ID3 erkennt die Trainingsdaten am besten, allerdings auch nur zu 77,5%. Das deutet darauf hin, dass die Daten allgemein nicht besonders gut gelernt werden können.

3.2 Zweiter Ansatz: Klassifikation mit eindeutigen Trainingsdaten

3.2.1 Vorverarbeitung

Datenanalyse Die enttäuschenden Ergebnisse im vorigen Ansatz führten zu einer detaillierten Untersuchung der Datenbasis. Dabei fiel auf, dass die Daten nicht immer eindeutig sind. Mehrdeutig sind Datensätze, wenn sie (ohne Score) identische Attributausprägungen haben, sich im Score jedoch unterscheiden. Eindeutig bedeutet dementsprechend, dass der Score dann auch identisch ist.

Ohne Berücksichtigung des Scores existieren 5 993 (55,42%) verschiedene Datensätze, mit seiner Berücksichtigung 7 084 (65,51%). Tab. 5 zeigt die Anzahl der identischen Datensätze in Abhängigkeit vom Score. So gibt es z. B. 14 identische Datensätze, die sich auch trotz gleicher Personenschäden im Score um 5 unterscheiden. Datensätze können bei den verschiedenen Differenzen mehrfach enthalten sein, bspw. könnte zu Datensatz x ein Datensatz y mit Score-Differenz 2 und ein Datensatz z mit Score-Differenz 3 existieren.

Tabelle 5: Score-Differenzen und gleiche Datensätze

Score-Differenz:	0	1	2	3	4	5	6–8
Anzahl identisch:	3 729	2 795	1 953	537	108	14	0

Test- und Trainingsmenge Als Schlussfolgerung aus der Analyse wurden die Datensätze gezielt in Trainings- und Testmenge aufgeteilt. In der Trainingsmenge waren dabei nur eindeutige Datensätze, der Rest in der Testmenge. Es soll damit sozusagen die Trainingsmenge auswendig gelernt und die Testmenge vorhergesagt werden. Unvollständige Datensätze wurden ebenfalls nur in die Testmenge übernommen, was sicherlich diskussionswürdig ist.

Da es wieder die gleiche Kostenfunktion wie im ersten Ansatz zu minimieren galt, wurde bei den mehrdeutigen Daten derjenige Score zum Lernen ausgewählt, der am dichtesten am Durchschnitts-Score seiner sonst gleichen Daten liegt. Bei auch dann nicht eindeutiger Entscheidung wurde der höhere Score gewählt, da das Vorhersagen höherer Scores wichtiger erscheint.

Aufgrund der Mehrdeutigkeiten muss es dabei zu falschen Klassifikationen kommen. In der Trainingsmenge sind 5 891 (54,48%) eindeutige und in der Testmenge die restlichen 4 922 (45,52%) Datensätze. Somit wurden weniger als $\frac{2}{3}$ der Daten zum Trainieren benutzt.

Codierung Es wurde nominal codiert. Als Änderungen zum vorherigen Ansatz wurden unbekannte Werte bei Altersklasse/Geschlecht als `NULL` (nicht vorhanden) gesetzt und alle Regelverstöße zu einem Attribut rv zusammengefasst mit Ausprägungen `a` bis `h`, was insgesamt 39 Attribute ergibt.

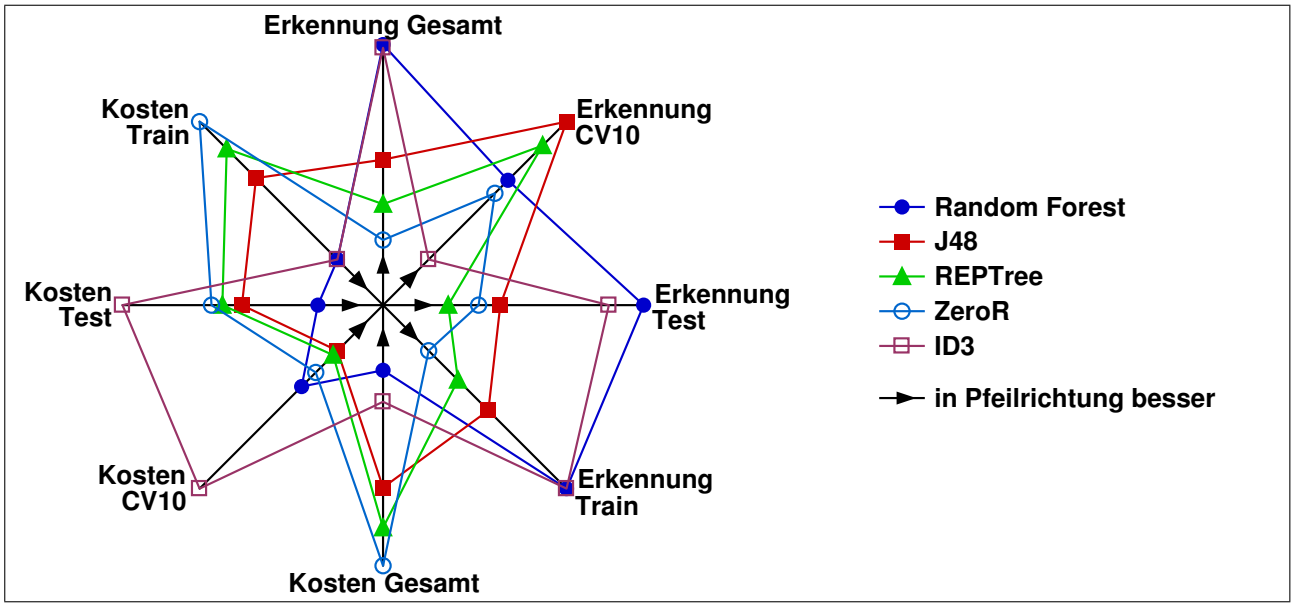
3.2.2 Ergebnisse

In Tab. 6 sind die Ergebnisse dargestellt. Die Algorithmen sind in gleicher Reihenfolge wie in Tab. 4 aufgeführt, wichtigstes Gütekriterium sind diesmal die Gesamtkosten. Kosten sind wieder nur innerhalb der gleichen Spalte vergleichbar.

Die Trainingsdaten werden von ID3 und Random Forest ca. 16% besser erkannt als im vorherigen Ansatz, bei den anderen Algorithmen war die Verbesserung unter 5%. Random Forest ist bei den Kosten wieder am besten, was auch an der Gesamterkennungsrate sichtbar ist. Zum besseren Vergleich mit dem ersten Ansatz wurde hier auch Crossvalidation auf der Trainingsmenge mit angegeben. Je besser die Trainingsmenge gelernt wurde, desto schlechter war die Erkennung bei Crossvalidation, was als Zeichen für Overfitting gewertet werden kann. ZeroR (klassifiziert immer als Score 3) und J48 sind davon ausgenommen.

Es fällt außerdem auf, dass die Werte für Crossvalidation bei jedem Algorithmus schlechter als im vorigen Ansatz sind. Dies lässt sich damit erklären, dass vorher viele Daten zwar mehrdeutig, aber auch mehrfach vorhanden waren. Diese Mehrfachen wurden öfter richtig erkannt und so wurde insgesamt ein besseres Ergebnis erzielt. Erst der aktuelle Ansatz mit Crossvalidation zeigt, wie wenig der Score eigentlich vorhergesagt werden kann.

Abbildung 9: Relatives Netzdiagramm der Ergebnisse vom zweiten Ansatz



Quelle: Eigene Darstellung.

Tabelle 6: Zweiter Ansatz – Algorithmen, Erkennungsraten, Kosten

Algorithmus	Gesamterkennungsrate [%]				Kosten			
	Train	Test	Gesamt	CV10	Train	Test	Gesamt	CV10
Rand. Forest	93,2	48,4	72,8	35,5	686	3 439	4 125	6 109
J48	60,1	43,8	52,7	40,7	3 715	3 804	7 519	5 405
REPTree	47,0	42,1	44,8	38,6	4 808	3 898	8 706	5 483
ZeroR	34,4	43,1	38,4	34,4	5 830	3 946	9 776	5 830
ID3	93,4	47,3	72,4	28,6	680	4 370	5 050	8 108

3.3 Dritter Ansatz: Klassifikation mit neuen Scores

3.3.1 Score-Varianten

Nach den bisherigen Ergebnissen konnte der Score nicht wirklich gut vorhergesagt werden. Im vorigen Ansatz wurden Mehrdeutigkeiten in der Trainingsmenge vermieden, in den Testdaten war dies nicht möglich. Der aktuelle Ansatz dagegen definiert andere Score-Varianten, um Mehrdeutigkeiten zu vermeiden. Diese werden direkt aus den Attributen Leichtverletzte, Schwerverletzte, Tote errechnet. Tab. 7 gibt eine Übersicht über mögliche Score-Varianten. ScoreC bis ScoreF lassen sich dabei auch nachträglich aus ScoreB berechnen, was in Kap. 3.3.5 genauer untersucht wird.

3.3.2 Vorverarbeitung

Codiert ist nominal, fehlende Werte bei Altersklasse und Geschlecht wurden auf einen eigenen Wert gesetzt. Regelverstöße wurden nicht zusammengefasst. Entfernt sind ID, cluster, SVT, Ursache, Leichtverletzte, Schwerverletzte, Tote. Anders als in Ansatz 1 wurden diesmal auch die Attribute Merkmale und Fahrzeuge entfernt, so dass mit der jeweiligen Score-Variante 44 Attribute bleiben.

Tabelle 7: Modifizierte Score-Varianten

Name	Wert	Personenschaden	Kommentar zum Score
ScoreB	1	keiner	Einteilung entspricht der des Original-Scores
	2	nur Leichtverletzte	
	3	höchstens 2 Schwerverletzte	
	4	mehr als 2 Schwerverletzte	
	5	1 Toter	
	6	mehrere Tote	
ScoreC	1	keiner	Unterscheidung zwischen kein Schaden, Leichtverletzte, Schwerverletzte, Tote
	2	nur Leichtverletzte	
	3	Schwerverletzte	
	4	Tote	
ScoreD	1	keiner	Einteilung in kein Schaden, Leichtverletzte, SVT
	2	Leichtverletzte	
	3	Schwerverletzte und Tote	
ScoreE	1	keiner	Unterscheidung zwischen kein Schaden, Verletzte, Tote
	2	Leicht- und Schwerverletzte	
	3	Tote	
ScoreF	1	keiner	Einteilung in nur Sachschaden und auch Personenschaden
	2	Verletzte und Tote	

Daraufhin wurden die Datensätze auf Mehrdeutigkeiten untersucht. Gezählt wurden die eindeutigen Datensätze inkl. mehrfacher Vorkommen. Von den mehrdeutigen wurde der jeweils häufigste (Modus) dazu addiert, inkl. mehrfacher Vorkommen. Die Summe ist dann die maximale Anzahl unterscheidbarer und damit richtig klassifizierbarer Datensätze. Das Ergebnis zeigt Tab. 8.

Tabelle 8: Maximale Unterscheidbarkeit bei verschiedenen Score-Varianten

Score-Variante:	Score	ScoreB	ScoreC	ScoreD	ScoreE	ScoreF
max. unterscheidbar:	8 384	10 487	10 487	10 489	10 523	10 527
in %:	77,5	97,0	97,0	97,0	97,3	97,4

3.3.3 Gesamterkennungsraten

Das Aufteilen in Trainings- und Testmenge wurde wieder Weka überlassen. Eine Übersicht der Gesamterkennungsraten bei unterschiedlichen Algorithmen und Score-Varianten bietet Tab. 9. Man sieht, dass die Erkennungsraten nach rechts, also bei Score-Varianten mit weniger Klassen, ansteigen. ScoreB ist dabei deutlich besser als der Originalscore, die restlichen steigen nur noch minimal an.

Man erkennt außerdem, dass ID3 auf den Trainingsdaten exakt die Werte aus Tab. 8 erreicht, was das Auswendiglernen beweist. Bei Crossvalidation und $\frac{2}{3}$-Regel ist SMO besser als ID3 und J48. Im Vergleich zu Tab. 4 vom ersten Ansatz zeigen ID3 und J48 beim Score jetzt etwas andere Werte, was auf das Weglassen zweier Attribute zurückzuführen ist.

Tabelle 9: Gesamterkennungsraten [%] aller Score-Varianten

Alg.	Test mit	Score	ScoreB	ScoreC	ScoreD	ScoreE	ScoreF
ID3	Trainingsdaten	77,5	97,0	97,0	97,0	97,3	97,4
	CV10	45,2	87,0	87,0	87,2	88,9	89,4
	$\frac{2}{3}$-Regel	42,7	85,7	85,7	85,9	88,0	88,8
J48	Trainingsdaten	59,9	90,7	90,7	91,0	91,8	91,9
	CV10	44,4	88,0	88,2	88,4	90,3	90,8
	$\frac{2}{3}$-Regel	43,6	87,9	87,8	87,9	90,5	90,6
SMO	Trainingsdaten	58,3	91,6	91,7	92,0	93,1	93,4
	CV10	46,5	88,7	88,7	88,8	90,7	91,1
	$\frac{2}{3}$-Regel	45,5	88,5	88,5	88,8	90,8	91,1

3.3.4 Erkennungsraten im Detail

Nachfolgend wird nur noch Crossvalidation wegen der besten Aussagekraft für Vorhersagen betrachtet. Tab. 10 und 11 geben einen detaillierteren Überblick über die Einzel- und Gesamterkennungsraten sowie Kosten bei ScoreB und ScoreD mit verschiedenen Algorithmen. Alle Angaben außer den Kosten sind prozentual angegeben. Die Kosten sind nur innerhalb der gleichen Score-Variante vergleichbar.

Am auffälligsten ist bei ScoreB, dass die meisten Algorithmen die höheren Scores komplett falsch klassifizieren. Außerdem hängt die Erkennungsrate eines Scores von seinem Anteil in der Gesamtmenge ab. Je häufiger er vorkommt, desto besser wird er klassifiziert. Für Score 4 bis 6 kann man dies aufgrund deren geringen Anzahl nicht verallgemeinern.

Auswertend lässt sich sagen, dass Random Forest und SMO die geringsten Kosten und die höchsten Gesamterkennungsraten erreichen. Es lassen sich zwei Optionen zur Weiterarbeit ableiten, welche in Kap. 3.3.6 kombiniert werden:

- Scores weiter zusammenfassen,
- Zusammensetzung der Trainingsmenge variieren, um höhere Scores besser zu erkennen.

Tabelle 10: Erkennungsraten [%] und Kosten ScoreB, CV10

ScoreB	Anteil	ID3	Dec.Table	PART	J48	NBTree	SMO	C.Rule	Rand.For.
1	85,5	94,8	98,2	96,4	98,1	97,3	98,6	98,1	97,3
2	9,5	44,4	39,3	36,9	33,1	41,9	40,1	42,3	43,5
3	4,2	38,9	13,1	26,0	21,4	8,1	12,0	—	38,4
4	0,3	18,2	9,1	—	—	—	—	—	18,2
5	0,3	36,1	—	—	—	—	—	—	39,5
6	0,1	50,0	8,3	—	—	5,6	—	—	50,0
unclassified	—	0,2	—	—	—	—	—	—	—
Gesamt:	100	87,0	88,3	87,0	88,0	87,5	88,7	87,9	89,2
Kosten:		1 878	1 720	1 890	1 772	1 857	1 671	1 764	1 545

Tabelle 11: Erkennungsraten [%] und Kosten ScoreE, CV10

ScoreE	Anteil	ID3	Dec.Table	PART	J48	NBTree	SMO	C.Rule	Rand.For.
1	85,5	94,8	97,6	96,2	97,7	96,7	98,0	98,1	96,7
2	14,0	55,9	47,7	51,7	48,1	51,2	49,1	41,9	56,2
3	0,4	35,4	—	—	—	6,3	—	—	35,4
unclassified	—	0,2	—	—	—	—	—	—	—
Gesamt:	100	88,9	90,2	89,6	90,3	90,0	90,7	89,8	90,8
Kosten:		1 206	1 080	1 159	1 071	1 112	1 022	1 132	1 011

3.3.5 Vorverarbeitung vs. Nachbereitung

Zuerst soll jedoch die Frage beantwortet werden, inwieweit sich nachträglich gebildete Score-Varianten von direkt trainierten/klassifizierten unterscheiden. ScoreB musste aus den Originaldaten neu gebildet werden. ScoreC bis ScoreF dagegen könnten direkt aus ScoreB abgeleitet werden.

Für J48 wurden ScoreC bis ScoreF sowohl gelernt als auch nachträglich berechnet, Tab. 12 zeigt dies demonstrativ für ScoreF. Spalte R enthält die Erkennungsrate (Recall) und Zeile P die Genauigkeit (Precision), je in Prozent.

Tabelle 12: Vergleich von berechnetem und gelerntem ScoreF bei J48, CV10

ScoreF berechnet

S	1	2	R
1	**9 078**	172	98,14
2	872	**691**	44,21
P	91,24	80,07	90,34

ScoreF gelernt

S	1	2	R
1	**9 047**	203	97,81
2	797	**766**	49,01
P	91,90	79,05	90,75

Es zeigt sich, dass vor allem hohe Scores bei der Version mit Vorverarbeitung besser erkannt werden. Die Gesamterkennungsrate ist ebenfalls minimal besser. Deshalb wurde bei der Vorverarbeitung geblieben.

3.3.6 Sach- oder Personenschaden

Bei einer Reduzierung auf ScoreF mit nur zwei Score-Ausprägungen wird nur noch zwischen Sach- und Personenschaden unterschieden. Da bietet es sich an, auf hohen F-Measure von Score 2 zu optimieren, also Personenschäden möglichst gut bzw. genau zu erkennen. Dafür wurden Algorithmen „alleine“, als auch in zwei Boostingvarianten durchlaufen. Das jeweils beste Ergebnis ist in Tab. 13 zusammen mit Erkennungsrate (R2) und Genauigkeit (P2) für Score 2, der Gesamterkennungsrate und den Kosten dargestellt. Weiterhin wurden für Vote einige Kombinationen getestet. Die Kostenmatrix bei ScoreF lautet

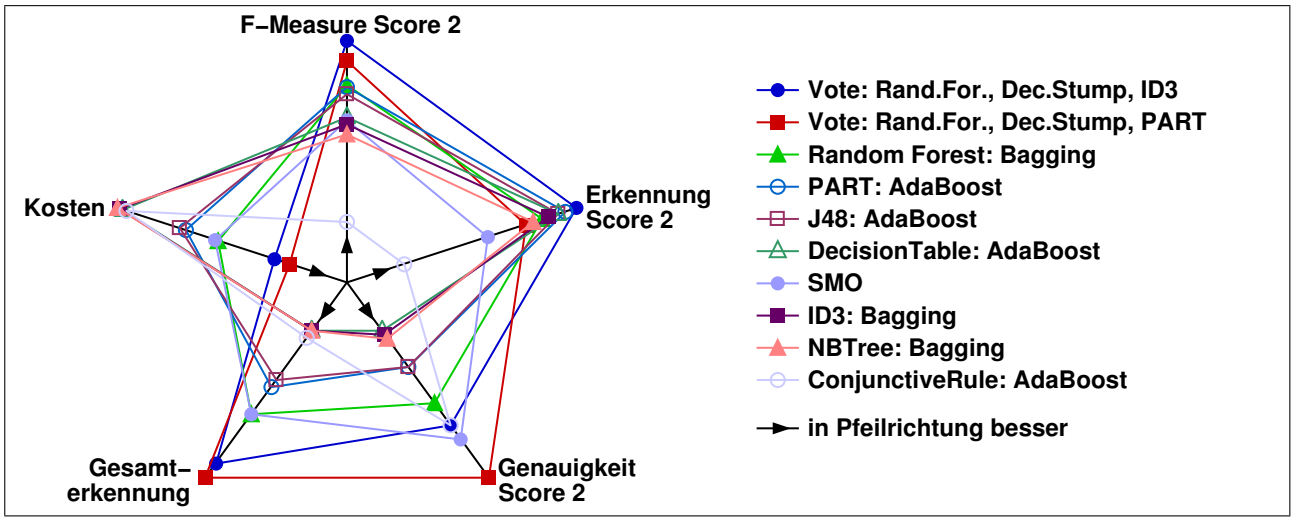
$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Damit entsprechen die Kosten der Anzahl der falschen Klassifikationen.

3.3.7 Interpretation

Ob AdaBoost oder Bagging bessere Ergebnisse liefert, hängt vom Basisalgorithmus ab. Überlegen war jedoch die Kombination mehrerer Basisalgorithmen

Abbildung 10: Relatives Netzdiagramm für geboosteten ScoreF, CV10



Quelle: Eigene Darstellung.

Tabelle 13: Geboosteter ScoreF, maximale F-Measure von Score 2, CV10

Algorithmus	F-M2	R2	P2	Gesamt	Kosten
Vote: Rand.For., Dec.Stump, ID3	0,677	59,4%	78,6%	91,8%	887
Vote: Rand.For., Dec.Stump, PART	0,663	54,4%	84,6%	92,0%	867
Random Forest: Bagging	0,646	56,0%	76,1%	91,1%	962
PART: AdaBoost	0,645	58,3%	72,0%	90,7%	1 005
J48: AdaBoost	0,640	57,6%	72,0%	90,6%	1 013
DecisionTable: AdaBoost	0,623	57,6%	67,8%	89,9%	1 089
SMO*	0,621	50,6%	80,3%	91,1%	966
ID3: Bagging†	0,618	56,6%	68,2%	89,9%	1 092
NBTree: Bagging	0,611	55,1%	68,7%	89,9%	1 095
ConjunctiveRule: AdaBoost	0,549	42,2%	78,6%	90,0%	1 083

* nicht geboostet aufgrund deutlich höherer Rechendauer

† 1 Datensatz nicht klassifiziert

mittels Vote. Die einfachen Algorithmen ID3 und DecisionStump in Kombination mit Random Forest lieferten das beste Ergebnis: knapp 60% Erkennung der Personenschäden bei etwas mehr als 21% Fehlalarm. Mit Kosten von 867 und einer Gesamterkennungsrate von 92% ist eine andere Vote-Kombination noch leicht besser. Diese Kombination für ScoreB durchgeführt ergab eine Gesamterkennung von 88,8% und Kosten von 1 660. Im Vergleich zu Tab. 10 zeigt sich, wie dabei das Ergebnis von Random Forest durch die Hinzunahme weiterer Algorithmen verschlechtert wird.

4 Zusammenfassung und Ausblick

Aufgabe war die Untersuchung von Unfalldaten, insbesondere die Vorhersage eines Unfallscores mittels Data-Mining-Methoden. Dazu wurde das inkrementelle Modell des Data Mining dreimal durchlaufen. Jeder Ansatz brachte neue Erkenntnisse. Somit war der letzte am aufschlussreichsten. Eine Übersicht mit den besten Ergebnissen zeigt Tab. 14, die Erkennungsraten sollten dabei möglichst hoch und die Kosten möglichst niedrig sein. Kosten sind nur relativ innerhalb des gleichen Ansatzes vergleichbar.

Für vergleichbare Zahlen beziehen sich die Daten von Ansatz 1 und 3 auf Crossvalidation, Ansatz 2 auf die Gesamtmenge. Ansatz 3 benutzt den neu gebildeten ScoreB mit 6 statt 9 Klassen. Deutlich sind die in jedem Ansatz ansteigenden Erkennungsraten und sinkenden Kosten erkennbar.[12]

Trotz der zuletzt hohen Erkennungsraten darf nicht übersehen werden, was diese eigentlich bedeuten. Hier können nur in verschiedenen Abstufungen schwere von leichten Unfällen unterschieden werden. Die hohe Anzahl leichter Unfälle sorgt bei deren guter Erkennung für eine hohe Gesamterkennungsrate.

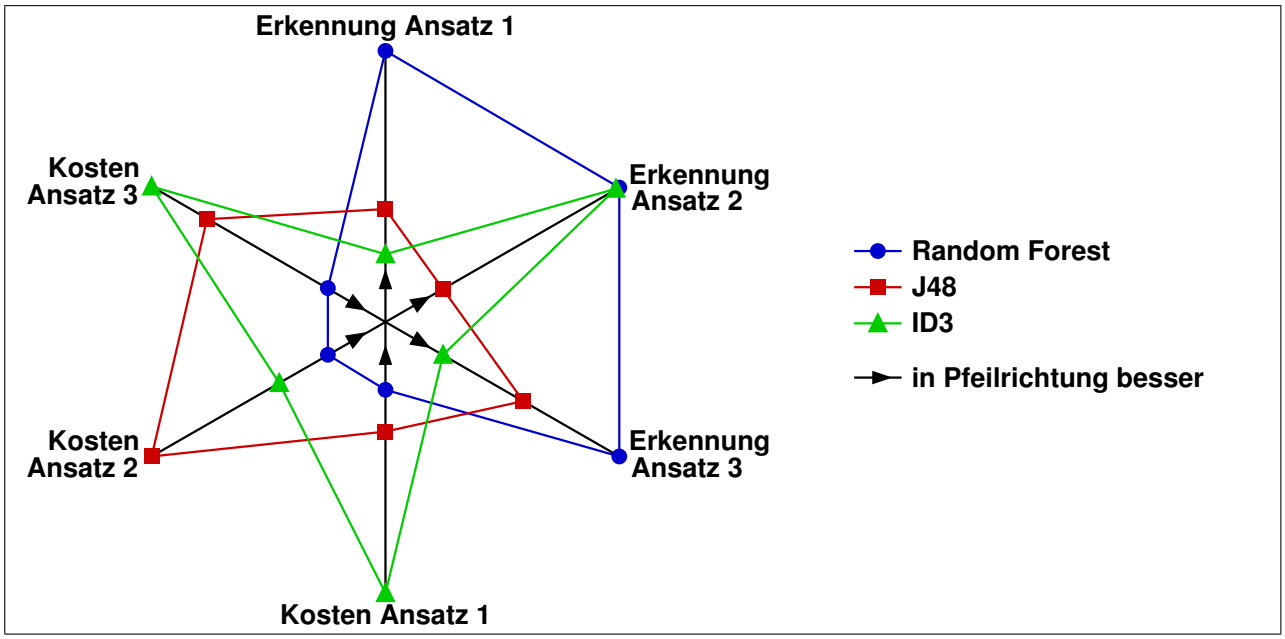
Wichtiger sind jedoch die schweren Unfälle mit Gefahr für Leib und Leben, welche deutlich schwieriger vorherzusagen sind. Je weniger Abstufungen im Score gemacht wurden, desto besser war die Vorhersage der hohen Scores. Bei nur zwei Unterscheidungen in Sachschaden und auch Personenschaden lag die Erkennungsrate für schwere Unfälle im besten Fall bei 59,4% mit einer Ungenauigkeit (Fehlalarm) von 21,4%. Dieses Ergebnis wurde mit einer Kombination aus Random Forest, ID3 und DecisionStump erreicht. Als bester einzelner Klassifikations-Algorithmus hat sich hier Random Forest herausgestellt.

Verbesserungen Mögliche Verbesserungen bieten sich in der Kombination aus Ansatz 2 und 3: Verringerung der Mehrdeutigkeiten durch eine geeignete Score-Variante und speziell beim Lernen durch gezielte Auswahl der Trainingsmenge. Auch ein höherer Anteil höherer Scores in der Trainingsmenge und eine unsymmetrische Kostenmatrix, die zu niedrig klassifizierte Scores stärker bestraft, erscheinen sinnvoll. Somit könnte ein eigener Boosting-Algorithmus für diese Problematik erdacht werden.

In jedem Schritt des Data-Mining-Prozesses sind weiterhin Optimierungen möglich: Attributauswahl, Verfeinerungen in der Codierung und Algorithmenparameter bieten Ansatzpunkte. Support Vector Machines konnten aufgrund ihrer Rechenzeit nicht ausreichend verglichen werden. Sie bieten einige Parameter, die – passend eingestellt – bessere Ergebnisse erwarten lassen als bisher,

[12] Abb. 6 zeigt die geringe Anzahl extrem falsch klassifizierter Datensätze. Daher sind Ergebnisse einer Kostenfunktion mit 6×6-Kostenmatrix in Ansatz 3 trotzdem vergleichbar mit den anderen Ansätzen.

Abbildung 11: Relatives Netzdiagramm der besten Ergebnisse



Quelle: Eigene Darstellung.

Tabelle 14: Beste Ergebnisse

Algorithmus	Gesamterkennungsrate [%]			Kosten		
	Ans. 1	Ans. 2	Ans. 3	Ansatz 1	Ansatz 2	Ansatz 3
Random Forest	46,9	72,8	89,2	9 005	4 125	1 545
J48	44,5	52,7	88,0	9 303	7 519	1 772
ID3	43,8	72,4	87,0	10 426	5 050	1 878

wofür weitere Experimente notwendig wären. Auch andere Algorithmen wie Neuronale Netze, die hier nicht beachtet wurden, mögen geeignet sein.

Praxiseinsatz Ein Einsatz in der Praxis stößt selbst bei idealer Vorhersage auf diverse Fragestellungen, so z. B. die Präsentation der Vorhersage. Soll das Fahrzeug gar selbst aktiv werden oder nur die Vorhersage anzeigen? Auch eine einfache Warnlampe sollte so dezent sein, dass ihr Aufleuchten nicht den Fahrer erschrickt und deswegen zu einem Unfall führt.

Ein noch größeres Problem ist die Datenerfassung. Wie können z. B. Alter und Alkoholspiegel des Fahrers während der Fahrt oder bei Fahrtantritt ermittelt werden? Man kann sicherlich davon ausgehen, dass mit zunehmender Technisierung mehr Daten erfasst werden und somit als Attribute zur Verfügung stehen. Eine gewisse Zufallskomponente wird man jedoch nie ganz ausschließen können.

A Tabellen

Tabelle 15: Aufbau der Exceltabelle. In Klammern gesetzte Attribute berechnen sich direkt aus anderen Attributen.

Nr.	Attribut	Beschreibung	Ausprägungen
A	Score	Schweregrad des Unfalls	1–9 (vgl. [6])
B	(cluster)	äußeres Milieu	Zeichenkette der Länge 1–9
C	(Merkmale)	Anzahl der x im äußeren Milieu	1–9
Regelverstoß			
D	unerlaubt	unerlaubte Geschwindigkeit	x=ja, leer=nein
E	unangemessen	unangemessene Geschwindigkeit	x=ja, leer=nein
F	WHP/DU	Fehler beim Abfahren, Wenden, Halten, Parken	x=ja, leer=nein
G	Überholen	Überholfehler	x=ja, leer=nein
H	Spur	Spurfehler	x=ja, leer=nein
I	ohne RV	ohne Regelverstoß	x=ja, leer=nein
J	Vorfahrt	Vorfahrtsfehler	x=ja, leer=nein
K	Abstand	Abstandsfehler	x=ja, leer=nein
L	(SVT)	Anzahl Schwerverletzte/Tote	x=ja, leer=nein
Äußeres Milieu			
M	Baum	Bäume und Masten	x=ja, leer=nein
N	Kurven	Kurven	x=ja, leer=nein
O	Steigungen	Steigungen	x=ja, leer=nein
P	GSP	Ausgeschilderte Gefahrenschwerpunkte	x=ja, leer=nein
Q	Kriraum	kritische Verkehrsräume	x=ja, leer=nein
R	Ausfahrt	Ausfahrten	x=ja, leer=nein
S	IFD	Fahrbahnmängel	x=ja, leer=nein
T	Vorfahrt	Vorfahrten	x=ja, leer=nein
U	Wild	Wildwechsel	x=ja, leer=nein
V	Baustellen	Baustellen	x=ja, leer=nein
W	nachts	Dunkelheit	x=ja, leer=nein
X	nass	Nässe	x=ja, leer=nein
Y	Nebel	Nebel	x=ja, leer=nein
Z	Wind	Wind	x=ja, leer=nein
AA	Glätte	Glätte	x=ja, leer=nein
AB	Mitfahrende	in gleiche Richtung fahrende Fahrzeuge	x=ja, leer=nein
AC	entgegen	entgegenkommende Fahrzeuge	x=ja, leer=nein
AD	Fg/Rf	Fußgänger/Radfahrer	x=ja, leer=nein
AE	MZR	motorisierte Zweiradfahrer	x=ja, leer=nein
AF	Altersklassen	Altersklassen in 10 Stufen, siehe Tab. 16	1–10, leer=unbekannt
AG	Geschlecht	männlich/weiblich	1–2, leer=unbekannt
AH	AWHP	Anfahren, Wenden, Halten, Parken	x=ja, leer=nein
AI	erlaubt	erlaubte Geschwindigkeit	x=ja, leer=nein
AJ	unerlaubt	unerlaubte Geschwindigkeit	x=ja, leer=nein
AK	FSP	Führerschein auf Probe	x=ja, leer=nein

Fortsetzung ...

Tabelle 15: (Fortsetzung)

Nr.	Attribut	Beschreibung	Ausprägungen
AL	Alkohol	unter Alkoholeinfluss	x=ja, leer=nein
AM	ohne FS	ohne Führerschein	x=ja, leer=nein
AN	Ursache	Amtliche Items der Unfallursachen	10–33, 35–55, 60–65, 67–70, 72–77, 81, 83–84, 86–90, 98–99
AO	(Fahrzeuge)	Anzahl der am Unfall beteiligten Fahrzeuge	1–6
Anzahl der am Unfall beteiligten ...			
AP	U1	Kleinkrafträder	1–2, leer=0
AQ	U2	Kräder	1, leer=0
AR	U3	PKWs	1–6, leer=0
AS	U4	Nutzfahrzeuge	1–3, leer=0
AT	U5	Radfahrer	1–2, leer=0
AU	U6	Fußgänger	1, leer=0
AV	U7	Schienenfahrzeuge	1, leer=0
AW	unbekannt	Unbekannten	x=mindestens 1, leer=0
Grad der Verletzung			
AX	Leichtverletzte	Anzahl Leichtverletzte	1–4, leer=0
AY	Schwerverletzte	Anzahl Schwerverletzte	1–5, leer=0
AZ	Tote	Anzahl Tote	1–2, leer=0

Quelle: [2].

Tabelle 16: Alter des Unfallverursachers

Klasse:	1	2	3	4	5	6	7	8	9	10
Alter [Jahre]:	18–21	22–24	25–29	30–34	35–39	40–44	45–49	50–54	55–59	≥ 60

Literatur

[1] **Agrawal**, Rakesh/**Srikant**, Ramakrishnan: Fast Algorithms for Mining Association Rules. In: **Bocca**, Jorge B. (Hrsg.)/**Jarke**, Matthias (Hrsg.) /**Zaniolo**, Carlo (Hrsg.): *Proc. 20th Int. Conf. Very Large Data Bases, VLDB* [Morgan Kaufmann] 1994, S. 487–499. – `http://www.almaden.ibm.com/software/projects/hdb/publications.shtml`

[2] **Bastian**, Dieter/**Stoll**, Regina: *Risikopotenziale und Risikomanagement im Straßenverkehr* [Forschungsinstitut für Verkehrssicherheit GmbH Schwerin, Universität Rostock] Juli 2004. – Forschungsbericht

[3] **Breiman**, Leo: Random Forests. In: *Machine Learning* 45 (2001), Nr. 1, S. 5–32. – `http://www.stat.berkeley.edu/users/breiman/RandomForests/`

[4] **CRISP-DM Consortium**: *CRISP-DM Process Model 1.0.* Version: 2000. `http://www.crisp-dm.org/Process/`. – Online-Ressource, Abruf: 2005-10-20

[5] **Fayyad**, Usama M. (Hrsg.)/**Piatetsky-Shapiro**, Gregory (Hrsg.)/**Smyth**, Padhraic (Hrsg.) /**Uthurusamy**, Ramasamy (Hrsg.): *Advanced Techniques in Knowledge Discovery and Data Mining* [AAAI/MIT Press] 1996

[6] **Fischer**, Th.: *Zur Koinzidenz menschlichen Versagens mit infrastrukturellen Defiziten.* Rostock, Medizinische Fakultät der Universität Rostock, Diss., 1994

[7] **Pal**, Nikhil R. (Hrsg.)/**Jain**, Lakhmi (Hrsg.): *Advances in Knowledge Discovery and Data Mining* [Springer] 2005

[8] **Platt**, John: *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines.* Version: 1998. `http://research.microsoft.com/users/jplatt/smo.html`. – Online-Ressource, Abruf: 2005-11-01. – Microsoft Research Technical Report MSR-TR-98-14

[9] **Russell**, Stuart J./**Norvig**, Peter: *Künstliche Intelligenz – Ein moderner Ansatz.* 2. Auflage [Pearson Studium] 2004

[10] **WHO**: *World Health Day: Road safety is no accident!* Version: 2004. `http://www.who.int/mediacentre/news/releases/2004/pr24/en/`. – Online-Ressource, Abruf: 2005-11-01. – Presserelease zum Weltgesundheitstag 2004

[11] **Wissuwa**, Stefan/**Cleve**, Jürgen /**Lämmel**, Uwe: *Analyse zeitabhängiger Daten durch Data-Mining-Verfahren* [Hochschule Wismar, FB Wirtschaft] 2005. – Wismarer Diskussionspapiere, Heft 21/2005

[12] **Witten**, Ian H./**Frank**, Eibe: *Data Mining: Practical Machine Learning Tools and Techniques.* 2nd Edition [Morgan Kaufmann, San Francisco] 2005. – `http://www.cs.waikato.ac.nz/~ml/weka/`

Autorenangaben

Christian Andersch
Studiengang Wirtschaftsinformatik 2003
Hochschule Wismar, Fachbereich Wirtschaft
Philipp-Müller-Straße
Postfach 12 10
D – 23952 Wismar
E-Mail: christian@andersch.net

Prof. Dr. rer. nat. Jürgen Cleve
Grundlagen der Informatik / Künstliche Intelligenz
Hochschule Wismar, Fachbereich Wirtschaft
Philipp-Müller-Straße
Postfach 12 10
D – 23952 Wismar
Telefon: ++49 / (0)3841 / 753 527
Fax: ++ 49 / (0)3841 / 753 131
E-Mail: j.cleve@wi.hs-wismar.de

WDP - Wismarer Diskussionspapiere / Wismar Discussion Papers

Heft 07/2005:	Melanie Pippig: Risikomanagement im Krankenhaus
Heft 08/2005:	Yohanan Stryjan: The practice of social entrepreneurship: Theory and the Swedish experience
Heft 09/2005:	Sebastian Müller/Gerhard Müller: Sicherheits-orientiertes Portfoliomanagement
Heft 10/2005:	Jost W. Kramer: Internes Rating spezieller Kundensegmente bei den Banken in Mecklenburg-Vorpommern, unter besonderer Berücksichtigung von Nonprofit-Organisationen
Heft 11/2005:	Rolf Steding: Das Treuhandrecht und das Ende der Privatisierung in Ostdeutschland – Ein Rückblick –
Heft 12/2005:	Jost W. Kramer: Zur Prognose der Studierendenzahlen in Mecklenburg-Vorpommern bis 2020
Heft 13/2005:	Katrin Pampel: Anforderungen an ein betriebswirtschaftliches Risikomanagement unter Berücksichtigung nationaler und internationaler Prüfungsstandards
Heft 14/2005:	Rolf Steding: Konstruktionsprinzipien des Gesellschaftsrechts und seiner (Unternehmens-)Formen
Heft 15/2005:	Jost W. Kramer: Unternehmensnachfolge als Ratingkriterium
Heft 16/2005:	Christian Mahnke: Nachfolge durch Unternehmenskauf – Werkzeuge für die Bewertung und Finanzierung von KMU im Rahmen einer externen Nachfolge –
Heft 17/2005	Harald Mumm: Softwarearchitektur eines Fahrrad-Computer-Simulators
Heft 18/2005:	Momoh Juanah: The Role of Micro-financing in Rural Poverty Reduction in Developing Countries
Heft 19/2005:	Uwe Lämmel/Jürgen Cleve/René Greve: Ein Wissensnetz für die Hochschule – Das Projekt ToMaHS
Heft 20/2005:	Annett Reimer: Die Bedeutung der Kulturtheorie von Geert Hofstede für das internationale Management
Heft 21/2005:	Stefan Wissuwa/Jürgen Cleve/Uwe Lämmel: Analyse zeitabhängiger Daten durch Data-Mining-Verfahren
Heft 22/2005:	Jost W. Kramer: Steht das produktivgenossenschaftliche Modell in Estland, Lettland und Litauen vor einer (Wieder-)Belebung?
Heft 23/2005:	Jost W. Kramer: Der Erfolg einer Genossenschaft. Anmerkungen zu Definition, Operationalisierung, Messfaktoren und Problemen
Heft 24/2005:	Katrin Heduschka: Ist die Integrierte Versorgung für Krankenhäuser und Rehabilitationskliniken das Modell der Zukunft?
Heft 01/2006:	Christian Andersch/Jürgen Cleve: Data Mining auf Unfalldaten