Prototyp einer generischen Recommendation Engine basierend auf Echtzeit-Assoziationsanalysen mit R

Masterarbeit

Zur Erlangung des Grades M.Sc. Informationsmanagement

vorgelegt von

Christian Meininger

211100038

cmeininger@uni-koblenz.de

1. Betreuer: Dr. Michael Möhring

2. Betreuer: Prof. Dr. Klaus G. Troitzsch

Koblenz, im Juli 2016

Inhalt

Abbildungsverzeichnis

Alle Abbildungen ohne Hinweise auf Quellen sind eigene Darstellungen ohne Anlehnung an Inhalte Anderer.

Tabellenverzeichnis

Akürzungsverzeichnis

BI……Business Intelligence

LHS……..…Left-Hand-Side

RE…….……Recommendation Engine

RHS……Right-Hand-Side

Zusammenfassung

In dieser Arbeit wurde unter Verwendung der Programmiersprache R ein Prototyp zur Erstellung einer Recommendation Engine zur Aufdeckung von Assoziationen innerhalb einer gegebenen Datenmenge entwickelt. Die Berechnung der Assoziationen findet hierbei in Echtzeit statt und des Weiteren wurden die Analysefunktionen generisch programmiert, um ein schnelles Einbinden und einfaches Parametrisieren von Datensätzen zu ermöglichen. Die Entwicklung fußte auf der grundlegenden Motivation, Data Mining Methoden wie das Assoziationsverfahren teilweise zu automatisieren, um damit generierte Lösungen effizienter umsetzen zu können. Der Entwicklungsprozess war insgesamt erfolgreich, sodass alle Grundfunktionalitäten im Sinne eines evolutionären Prototypings vorhanden sind.

1. Einführung

Die Arbeit wurde im Zuge einer Tätigkeit als Masterand bei Altran Deutschland S.A.S. & Co. KG verfasst. Dabei fand eine enge Zusammenarbeit zwischen der Universität Koblenz-Landau und Altran durch regelmäßige Zusammentreffen statt.

1.1 Motivation und Problemstellung

Die anhaltende Digitalisierung hat innerhalb der letzten Jahre erhebliche Veränderungen in Wirtschaft (vgl. Worthington, 2014, S. 55 ff.; Guerrieri & Bentivegna, 2011), Forschung (vgl. Owen, 2007, S. 223 ff.) sowie dem privaten Alltag (vgl. Holm, Jarrick, & Scott, 2015, S. 82 f.; Goldfarb & Tucker, 2012, S. 84 ff.) verursacht. Die grundlegende Innovation ist hierbei das Internet, welches als digitale Infrastruktur die weltweite Kommunikation von Daten ermöglicht (vgl. Berners-Lee, Cailliau, Groff, & Pollermann, 2010, S. 470 f.; Chen, Chiang, & Storey, 2012, S. 1168). Auf dieser Infrastruktur aufbauend, fand die Digitalisierung neben zahlreichen, mit dem Internet verbundenen Endgeräten vor allem durch Computer und Smartphones Einzug in nahezu alle Lebensbereiche, was als „Internet der Dinge“ definiert ist (vgl. daCosta, 2013, S. 23 ff.; Sun, Bie, Thomas, & Cheng, 2014). Eine derartige Infrastruktur eröffnet völlig neue Dimensionen in der Datenakquirierung, denn in jeglichen digitalen Strukturen können im Einklang mit den jeweilig geltenden Datenschutzregeln an geeigneten Stellen Daten erhoben werden (vgl. Hilbert, 2011). Diese Möglichkeit wurde innerhalb der letzten 15 Jahre sowohl innerhalb als auch außerhalb von Unternehmen ausgiebig genutzt und es lässt sich ein Trend zur exponentiell wachsenden Sammlung von Daten erkennen (siehe Abbildung 1).

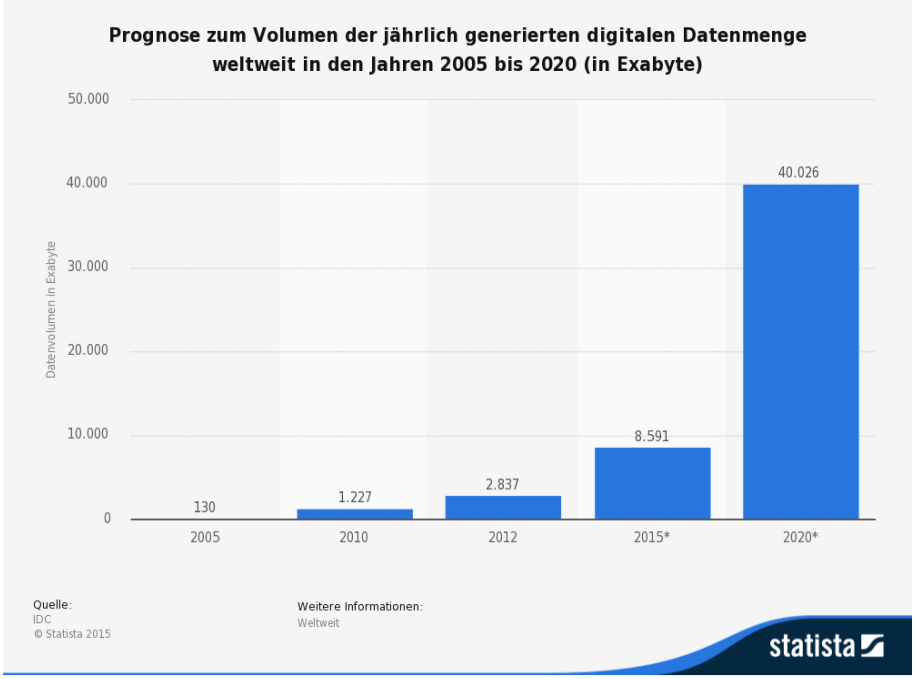


Abbildung 1: Prognose zum weltweit generierten digitalen Datenvolumen (vgl. Statista GmbH, 2016a)

Diese immense Datengrundlage wird im Allgemeinen als „Big Data“ betitelt (vgl. McKinsey Global Institute, 2011). Die Definition von Datensammlungen als Big Data basiert jedoch nicht nur auf der bloßen Menge an Daten, sondern auch auf dessen Vielfalt, Geschwindigkeit sowie Verwendung (siehe Abbildung 2). Insofern entspringt Big Data vielfältigen Quellen (z.B. Geschäftsanwendungen innerhalb einer Unternehmung) und besteht somit aus einer Fülle an diversen Formaten und Strukturiertheitsgraden, sodass es bei einer ganzheitlichen Betrachtung einer geeigneten Aggregation unter Berücksichtigung der unterschiedlichen Formate bedarf; eine derart integrierte Datengrundlage ermöglicht dann jedoch weitreichende Potenziale auf das Aufdecken von neuartigen Informationen (vgl. Bitkom, 2012, S. 23 ff.). Darüber hinaus handelt es sich bei Big Data vor allem auch um, durch die digitale Infrastruktur ermöglichte, Datensammlungen in Echtzeit. Dies offenbart einen maßgeblichen Unterschied zu traditionellen Datensammlungen, welche zu einem gewissen Zeitpunkt erhoben werden, denn Echtzeitakquirierung ermöglicht Analysemethoden in Echtzeit (vgl. Bitkom, 2012, S. 37 ff.). Die Faktoren Menge, Vielfalt und Geschwindigkeit von Big Data ermöglichen schließlich das Analysefeld „Data Mining“.

Data Mining beschreibt die Analyse von Big Data durch Aufdecken von Mustern und Trends zur Generierung von nützlichen Informationen (vgl. Linoff & Berry, 2011, S. 2 ff.). In Analogie zu dem Unterschied zwischen klassischen Datensammlungen und Big Data, ist zum jetzigen Zeitpunkt Data Mining ebenso von der klassischen Inferenzstatistik abzugrenzen.

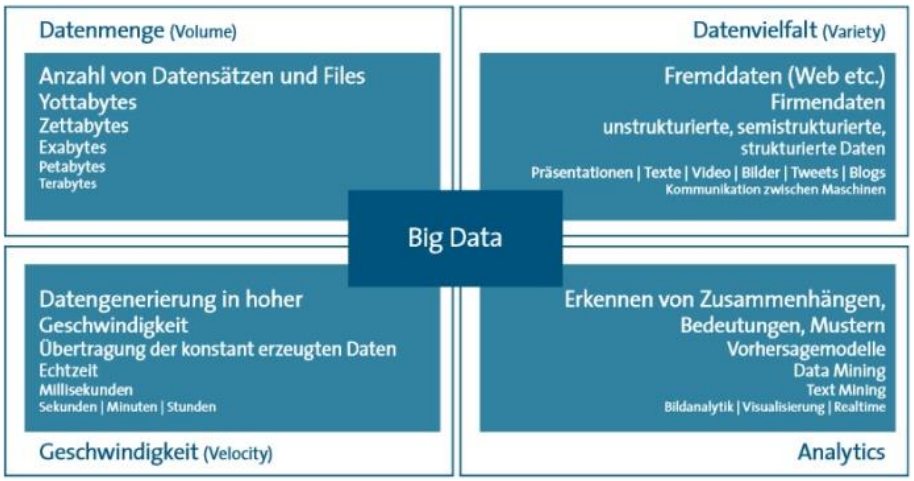


Abbildung 2: Merkmale von Big Data (vgl. Bitkom, 2012, S. 19)

Während in der Inferenzstatistik der Fokus auf dem Eliminieren von Unsicherheiten in vergleichsweise kleinen Datenmengen liegt, sind die Methoden des Data Mining auf die bereits beschriebenen Merkmale von Big Data ausgerichtet und ermöglichen somit adäquate Analysemethoden von großen Datenmengen (vgl. Kuonen, 2004, S. 6 f.). Dazu zählen Assoziationsanalysen, Segmentierungen, Klassifizierungen sowie Text-Mining (vgl. Kantardzic, 2011). Derartige Methoden werden abhängig von dem Einsatzfeld und dem zu eruierenden Ziel ausgewählt und anschließend ein Analysemodell erstellt, getestet, etabliert sowie evaluiert (vgl. Chapman, et al., 2000, S. 47 ff.). Diese Methoden bieten eine gleichermaßen ganzheitliche und tiefgehende Analyse, welche der klassischen Inferenzstatistik bei Big Data überlegen ist (vgl. Kuonen, 2004, S. 3), jedoch auch entsprechend höheren Aufwand erfordert (vgl. Campos, Stengard, & Milenova, 2005, S. 1).

Die vorherrschende Situation einer umfassenden, digitalen Infrastruktur zur Sammlung von Big Data und tiefergehenden Analyse durch Data Mining zur Unterstützung von Unternehmungen ist die grundlegende Motivation dieser Arbeit. Denn das Potenzial, was in Big Data liegt und durch Data Mining freigesetzt werden kann, ist immens (vgl. Vossen, 2014, S. 11 ff.; Bitkom, 2012, S. 34 ff.). Neben der stetig wachsenden Datengrundlage (siehe Abbildung 1) kann dieses Potenzial ebenso anhand des teilweise prognostizierten Umsatzes mit betreffenden Lösungen aufgezeigt werden (siehe Abbildung 3).

Hierbei zeigt sich neben der Vielfalt an Segmenten zur Bearbeitung von Big Data ebenfalls dessen wirtschaftliches Potenzial. Die für die Thematik dieser Arbeit in Form von Data Mining relevanten Segmente „Apps & Analytics“ sowie „Professional Services“ nehmen hierbei einen besonders hohen Anteil ein. Anhand der weiteren Segmente in der Abbildung lässt sich jedoch erkennen, dass die, der Analyse vorgeschalteten, Prozesse und Technologien ebenfalls einen hohen Anteil einnehmen (vgl. Chaudhuri & Dayal, 1997). Diese integrierte Gesamtheit

von der Akquirierung, Speicherung, Aggregation sowie Analyse von Big Data hin zu der Einbettung der gewonnenen Informationen in Unternehmen wird als „Business Intelligence“ (BI) beschrieben (vgl. Gluchowski, Gabriel, & Dittmar, 2008, S. 89 ff.).

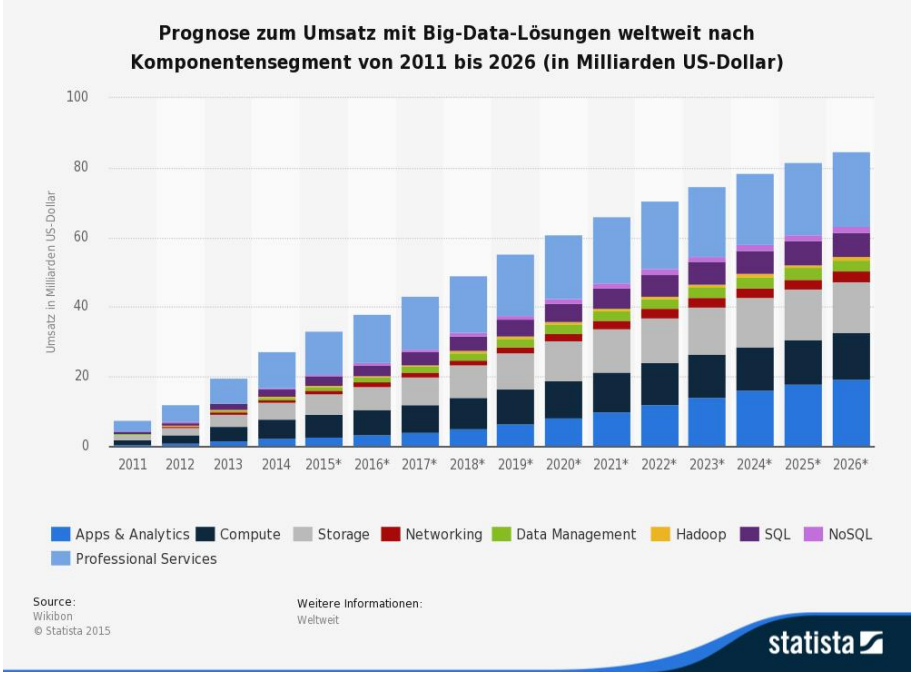


Abbildung 3: Umsatz mit Big Data Lösungen (vgl. Statista GmbH, 2016b)

Das Potenzial des Gesamtkonzepts BI ist für Unternehmen immens, da es Daten aus verschiedensten internen und externen Quellen aus diversen Bereichen zusammenführen kann und somit aufschlussreiche Informationen liefert, welche bei einer nicht-integrierten Betrachtung nicht zu Tage kommen würden (vgl. Chen, Chiang, & Storey, 2012, S. 1185 f.). Die durch BI generierten Informationen können außerdem in nahezu allen Bereichen eines Unternehmens vom Strategischen Management über Produktentwicklung bis hin zum Customer Relationship Management angewendet werden (vgl. Chen, Chiang, & Storey, 2012, S. 1168 ff.). Es erschließt sich somit ein Gesamtbild, wonach die in den letzten Jahren entstandenen Datenmengen Big Data in vielen Unternehmen hohe Potenziale durch dessen integrierten Managements und Data Mining im Zuge einer BI-Strategie ermöglichen. Jedoch bestehen einige Probleme, welche den weitreichenden Einsatz von BI vor allem jenseits großer Unternehmen bremsen.

Vordergründlich geht es dabei um die hohen Investitionskosten, welche für BI-Projekte aufgewendet werden müssen. Aufgrund des komplexen Themenfeldes und den diversifizierten Tätigkeiten (siehe Abbildung 3) benötigt die Durchführung solcher Projekte spezialisierte Experten (vgl. Campos, Stengard, & Milenova, 2005, S. 1), welche hohe Personalkosten ver-

ursachen. Hierbei nehmen das Erarbeiten von Wissen bezüglich der Unternehmung und Datengrundlage, die Datenvorbereitung sowie die Zielbestimmung einen großen Anteil des Gesamtaufwandes der Experten ein, während das Erstellen des jeweiligen Analysemodells und dessen Anwendung vergleichsweise wenig Aufwand erfordert (vgl. Chapman, et al., 2000, S. 10 ff.). Des Weiteren wird für die im BI durchgeführten Aktionen und vor allem derer des Data Mining spezielle Software benötigt (vgl. Neckel, 2011), welche aus dem kommerziellen Bereich hohe Lizenzkosten nach sich zieht. Vorherrschende BI-Softwarelösungen sind dabei vor allem darauf ausgelegt, möglichst viele Analysemethoden abzubilden, um eine gegebene Datengrundlage so ausgiebig wie möglich zu ergründen (vgl. Neckel, 2011). Hierfür müssen allgemeine Strukturen programmiert werden, welche erst durch das Expertenwissen des Analysten in Bezug auf das betreffende Unternehmen sowie der Datengrundlage zu konkreten Analysemodellen geformt werden können (vgl. Campos, Stengard, & Milenova, 2005, S. 1). Aus diesen Umständen an Personalaufwand sowie Lizenzkosten für Software entstehen Kosten, welche zugleich nach eigener Auffassung hohes Einsparpotenzial beinhalten.

Neben der Höhe der Kosten spielt jedoch auch die erschwerte Planbarkeit von BI-Projekten eine Rolle. Denn aufgrund der diversifizierten Datengrundlage und dem weiten Einsatzbereich innerhalb einer Unternehmung ist die Zielbestimmung und damit verbunden der Umfang des Data Mining in Form von Methodenauswahl und Modellgenerierung sowie der grundsätzlichen strategischen Vorbereitung nur schwierig abzusehen (vgl. McKinsey Global Institute, 2011, S. 114 ff.). Aus diesen Hürden in Bezug auf Kosten und Planbarkeit erwächst weiterhin ein Problem in Form von dem Verhältnis zwischen zu leistendem Aufwand und dem zu erwartenden Nutzen. Den hohen Investitionskosten stehen diesbezüglich zumeist vage Vorstellungen von dem Ziel und den damit verbundenem Nutzen gegenüber und nicht selten ist dieser selbst nach der Durchführung eines Projektes nicht genau messbar (vgl. Lönnqvist & Pirttimäki, 2006, S. 34 f.; Smith & Crossland, 2008, S. 172 f.) oder für Entscheidungsträger greifbar (vgl. IBM Institute for Business Value, 2013, S. 3 f.). Zwar haben die Themen Big Data und BI vor allem vor dem Hintergrund des zentralen Begriffs „Industrie 4.0“, welcher eine zentrale Strategie zur Unterstützung der Industrie durch moderne Informationstechnik darstellt (vgl. Bundesministerium für Bildung und Forschung, 2015a; Bundesministerium für Bildung und Forschung, 2015b), in den letzten Jahren an Bedeutung gewonnen, jedoch verhindern die aufgezeigten Problemstellungen bezüglich Kosten, Planbarkeit sowie erwartetem und messbaren Nutzen oftmals den praktischen Einsatz von BI-Lösungen, sodass ein Großteil des in diesem Kapitel aufgezeigten Potenzials brachliegt (vgl. Chen, Chiang, & Storey, 2012, S. 1169 ff.).

1.2 Zielsetzung und Vorgehensweise

Das Ziel dieser Arbeit ist die Erstellung eines Prototyps unter Verfolgung eines alternativen Ansatzes bezüglich der Entwicklung von BI-Lösungen, welcher die aufgezeigten Problemstellungen weitgehend eliminieren kann. Es soll hierbei der Vorgang der kostenintensiven, auf allgemeine Strukturen ausgelegten Vorgehensweise mittels etablierter BI-Lösungen durch kostengünstige Alternativvorgänge für spezielle Zielstellungen unter Verwendung des zu entwickelnden Prototyps mit hohem Automatisierungsgrad ersetzt werden. Durch die entsprechende Automatisierung soll das zuvor beleuchtete Einsparungspotenzial in Bezug auf Personal- und Softwarekosten (vgl. Campos, Stengard, & Milenova, 2005; Neckel, 2007) erschlossen werden. Denn statt des vorherrschenden, allgemeinen Ansatzes zur Abbildung möglichst aller Analysemethoden für jedwede Unternehmen und Datengrundlagen zur Erreichen von diversen Zielbestimmungen, wird in dieser Ausarbeitung der Ansatz verfolgt, für eine spezielle Zielgruppe inklusive einer spezifischen Datengrundlage (Daten mit Prozessen und Ausprägungen) mittels einer spezifischen Analysemethode (Assoziationsregelanalyse) ein bestimmtes Ziel (Recommendation Enigne) zu verfolgen. Zwar schränkt man somit die Zielgruppe im Gegensatz zu allgemeinen BI-Lösungen erheblich ein, jedoch wird der neue Prototyp weitestgehend automatisch arbeiten. Darüber hinaus wird bei der Erstellung des Prototyps ein maßgeblicher Fokus auf der Kommunikation der Ergebnisse liegen, um den Mehrwert des BI-Einsatzes zum einen für den Nutzer der RE und zum anderen für die jeweiligen Entscheidungsträger zu vermitteln. Konkret werden hierbei in Bezug auf die RE zahlreiche Visualisierungsmethoden herangezogen, damit der Nutzer möglichst ohne Fachkenntnis schnell und einfach die entsprechenden Ergebnisse des Prototyps interpretieren kann. Bezüglich der Vermittlung des Nutzens für die Entscheidungsträger werden geeignete Methoden zur Messbarkeit des Erfolges während des Einsatzes des Prototyps integriert. Schließlich wird der Prototyp mittels der Open-Source Programmierungssprache R geschrieben, sodass neben den aufgezeigten Einsparungen an Aufwand durch Spezialisierung und verbesserter Kommunikation des Nutzens auch die Lizenzkosten für die zu Grunde liegende Software minimiert werden.

Zur Erreichung des aufgezeigten Ziels werden in einem ersten Schritt thematische Grundlagen in der Literatur erarbeitet (siehe Abbildung 4). Der Fokus liegt hierbei verstärkt auf den Thematiken Crisp-DM (Vorgehendsmodell für BI-Projekte), Assoziationsregelanalyse, Recommendation Engines (RE), Prototyping, Visualisierungsmethoden in Bezug auf die Assoziationsregelanalyse sowie der Programmierumgebung R. Im Anschluss wird die bereits beschriebene Aneignung von Wissen bezüglich der für den Einsatz des Prototyps in Frage kommen-

den Unternehmen vorgenommen. Bei diesem Prozess wird verstärkt eine Verbindung zu dem zuvor erörterten Wissen bezüglich der Assoziationsregelanalyse hergestellt.

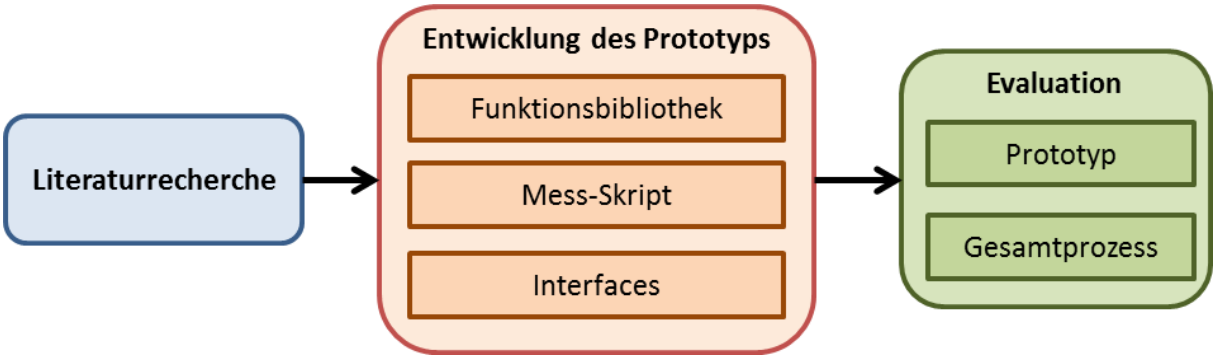


Abbildung 4: Forschungsvorgehen

Im Anschluss an die Literaturrecherche geht es im zweiten Schritt in die Entwicklung des Prototyps, welche grob in drei Phasen gegliedert werden kann. In der ersten Phase wird die eigentliche Analyse vorgenommen, in welcher zuerst die Integration und Preparation der Datengrundlage und weiterhin die Berechnung der Assoziationsregeln sowie Formatierung der Ergebnisse vorgenommen werden. Das Ergebnis ist eine Funktionsbibliothek, auf welche der Prototyp zugreift. In der zweiten Entwicklungsphase wird ein Mess-Skript erstellt, welches zeitpunktbezogen angewendet wird und Daten des Prototyps zum Zwecke des Monitoring und Nutzenmessung ausliest. Die dritte Entwicklungsphase beschäftigt sich mit der Integrationsmöglichkeit in Form von Interfaces für den Nutzer. An dieser Stelle kommen die bereits beschriebenen Visualisierungs- und Messbarkeitsmethoden zum Einsatz, um eine einfach zu integrierende und leicht verständliche RE zu liefern.

Der letzte Schritt des Forschungsvorgehens ist die Evaluierung des finalen Prototyps sowie des Gesamtprozesses der Entwicklung. Bei ersterem werden die verschiedenen Anpassungsmöglichkeiten des Prototyps getestet und vor allem auch die Performanz eruiert. Außerdem werden die entwickelten Funktionalitäten mit den angestrebten Zielen, welche aus der grundlegenden Thematik erwuchsen, verglichen. Schließlich wird der gesamte Prozess der Entwicklung mit Hinblick auf dem Potenzial des verfolgten Ansatzes hin zu einer verstärkten Entwicklung von spezialisierten und automatisierten BI-Lösungen rückblickend bewertet. Hierbei werden Implikationen aus dem Gesamtentwicklungsprozess für die Forschung, Praxis und die Weiterentwicklung des Prototyps gegeben.

1.3 Struktur

Aufgrund unterschiedlicher Umfänge und Prioritäten von Prozessen zwischen wissenschaftlichem und praktischem Bereich, unterscheidet sich das praxisorientierte Forschungsvorgehen marginal von der wissenschaftlichen Dokumentation. Die diesbezüglichen Verbindungen der unterschiedlichen Prozesse sind in Abbildung 5 illustriert und werden folgend innerhalb der präsentierten Ausarbeitungsstruktur erläutert.

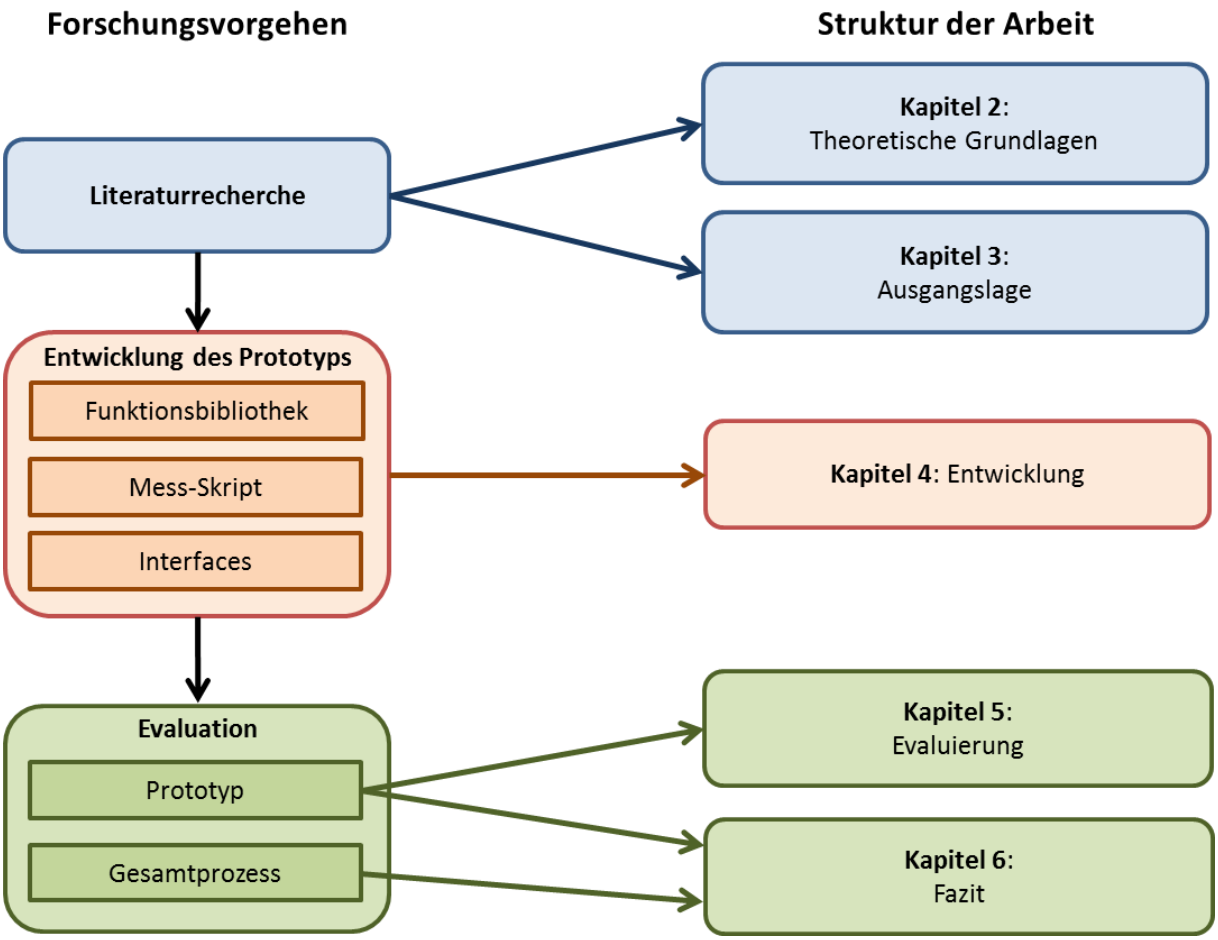


Abbildung 5: Zusammenhang zwischen Forschungsvorgehen und Struktur der Arbeit

Kapitel 2 enthält die aus der Literatur bezogenen Kenntnisse der grundlegenden Thematiken und eignet sich somit zur Schaffung eines Überblicks.

Kapitel 3 besteht aus zwei Unterkapiteln, welche die Ausgangslage vor dem Entwicklungsprozess, welche ebenfalls aus der Literaturrecherche entspringen, darstellen. Im ersten Teil wird die potenzielle Zielgruppe des Prototyps anhand der Anforderungen der Analysemethode und dem Data Mining Ziel erörtert. Im zweiten Unterkapitel werden die Programmierumgebung, Anforderungen und die grundlegende Systemstruktur des Prototyps beleuchtet.

Hier zeigt sich ein erster Unterschied zwischen dem im Forschungsdesign dargestellten Schritt der Literaturrecherche und der aufgeteilten Präsentation der Ergebnisse in dieser Aus-

arbeitung in Form von zwei separaten Kapiteln (2 und 3), da die Informationen bezüglich der Ausgangslage vor der Entwicklung von den allgemeinen Thematischen Grundlagen wissenschaftlich abzugrenzen sind.

Kapitel 4 ist der Hauptteil der Ausarbeitung und stellt die Dokumentation der Prototypentwicklung dar. Hierbei werden die Entwicklungsschritte aus dem Forschungsvorgehen auch für die Struktur übernommen, da die Entwicklungsschritte aufeinander aufbauen und daher eine praxisnahe Dokumentation am greifbarsten ist.

Kapitel 5 widmet sich der im Anschluss an die Entwicklung stattfindende Evaluierung des fertigen Prototyps. Hierbei werden zum einen die verschiedenen Anpassungen der Funktionen bezüglich der RE in Bezug auf eine allgemeine Verwendung getestet und weiterhin die Performanz beleuchtet. Außerdem wird überprüft, inwieweit die im Prototyp enthaltenen Funktionalitäten die zu Beginn der Ausarbeitung aufgestellten Zielstellungen erfüllen.

Kapitel 6 enthält die rückblickende Bewertung der gesamten Ausarbeitung, von der ursprünglichen, aus der Theorie erwachsenden Idee über die technische Entwicklung bis hin zu der Evaluation des fertigen Prototyps. Hierbei werden die spezifischen Erkenntnisse bezüglich des Prototyps auf die gesamte Entwicklung und den übergeordneten Ansatz von allgemeinen zu spezifischen BI-Lösungen projiziert, sodass Kapitel 6 in der Ausarbeitung durch Kenntnisse aus beiden Schritten des Forschungsdesign bezüglich Prototyp- und Gesamtevaluation entsteht. Diese Ergebnisse werden in greifbare Praxis-, Forschungs- und Weiterentwicklungsimplikationen überführt.

2. Theoretische Grundlagen

Innerhalb dieses Kapitels werden Themenblöcke behandelt, welche als Grundlage für die Entwicklung des Prototyps dienen. Zuerst wird mit „Crisp-DM“ das Standardmodell für die grundsätzliche Vorgehensweise von Data Mining Projekten vorgestellt. Denn die Entwicklung des Protoypen stellt den Versuch dar, eine weitgehend generische und automatisierte Projektdurchführung für den Einsatz von REs zu ermöglichen. Dies entspricht einer dem Crisp-Modell ähnlichen allgemeinen Anwendbarkeit, jedoch für ein spezielles Ziel und mit erheblich weniger Aufwand.

Die RE ist das Ziel des in dieser Arbeit gewählten Ansatzes eines partiell automatisierten Crisp-DM Modells. Die in diesem Kapitel vorgestellten Varianten dienen als Grundlage, um in Kapitel 3 eine spezifische Vorgehensweise für die Entwicklung der RE herauszustellen.

Die Assoziationsregelanalyse ist eine Modellierungsmethode aus dem Data Mining und eine von mehreren Möglichkeiten zur Berechnung der Datengrundlage für eine RE. Die Assoziationsregelanalyse ist die verwendete Modellierungsmethode innerhalb des Prototyps.

Da Assoziationsregeln und insbesondere deren Messfaktoren für den potenziellen Nutzer zu komplex sein könnten, sollen in der RE Visualisierungsmethoden zum Einsatz kommen. Diese sollen ein intuitives Verständnis gewährleisten, damit keine aufwändigen Schulungen bei dem Einsatz des Prototyps notwendig werden. Daher werden in diesem Kapitel mehrere Methoden vorgestellt, welche für diesen spezifischen Ansatz anwendbar sind.

Weiterhin wird das grundsätzliche Vorgehen bezüglich Prototyping als Rahmen für die Entwicklung behandelt. Dies liegt darin begründet, dass eine vollständig generische Entwicklung in einem komplexen Themenbereich wie der Assoziationsregelanalyse frühzeitige Evaluierungen bedarf. Daher werden verschiedene Prototyping Arten beleuchtet, um die Entwicklung des Prototyps optimal steuern zu können.

Abschließend wird die zur Entwicklung des Prototyps verwendete Anwendungsumgebung R vorgestellt.

2.1 Crisp DM

„Crisp-DM“ steht für „**CR**oss **I**ndustrial **S**tandard **P**rocess for **D**ata **M**ining“ und ist ein allgemeines Prozessmodell für Data Mining Projekte (vgl. Chapman, et al., 2000). Die Entstehung des Modells ist auf die Situation zurückzuführen, als Data Mining Methoden neu am Markt und in fast allen Industriezweigen einsetzbar waren, jedoch kein allgemeines Vorgehensverfahren existierte, welches einen standardisierten und effizienten Ablauf von Data Mining Projekten zentral aufzeigen konnte. Aus diesem Grund wurde ein EU Projekt im Zeit-

raum 1996 - 1999 durchgeführt, welches von einer „Special Interest Group“ mit vier Unternehmen aus der Wirtschaft[1] unterstützt wurde. Dabei wurde mit Erfolg eine in allen Industriefeldern einsetzbare (→„**CR**oss **I**ndustrial“) standardisierte Vorgehenweise (→„**S**tandard“) entwickelt und in die Form eines Prozessmodells (→„**P**rocess“) für Data Mining Projekte (→„for **D**ata **M**ining“) überführt.

Das Gesamtmodell ist in sechs Phasen unterteilt, welche in Abbildung 6 jeweils durch einen blauen Kasten dargestellt sind. Ein Data Mining Projekt startet insofern bei der Phase „Business Understanding“ und verläuft dann den durchgezogenen Pfeilen entsprechend bis zu der Endphase „Deployment“. Die dargestellten Doppelpfeile stehen für eine Interaktion bzw. Rückkopplung zwischen den jeweilig verbundenen Phasen. Das Modell wird nachfolgend im Detail erläutert.

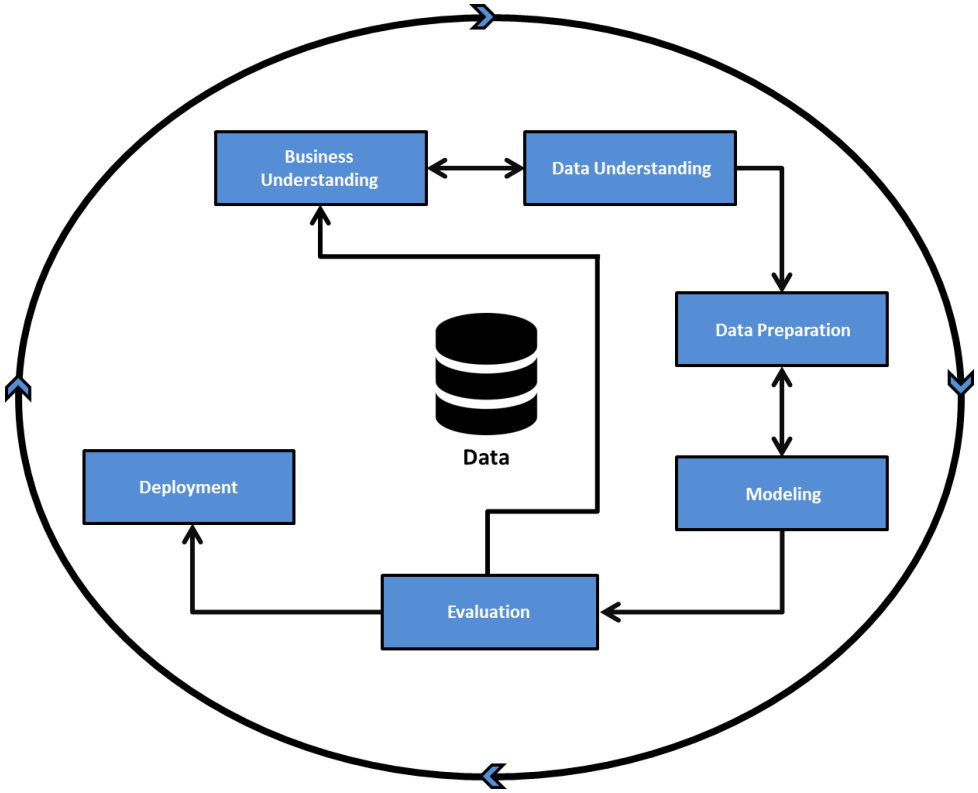


Abbildung 6: Gesamtmodell Crisp-DM (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

[1] Die Special Interest Group bestand aus: DaimlerChrysler, NCR Systems Copenhagen, OHRA Bank Groep B.V sowie SPSS Inc.

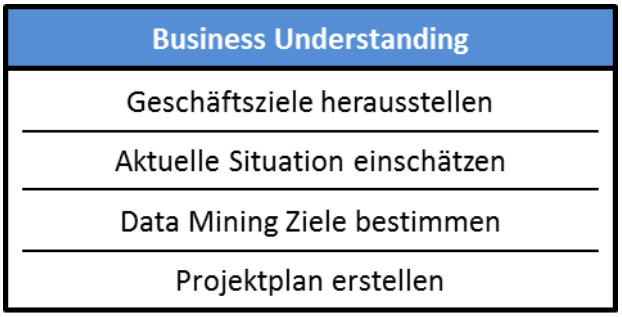


Abbildung 7: Aufgaben in der Phase Business Understanding (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

In der ersten Phase „**Business Understanding**“ geht es darum, dass sich der Analyst die Gegebenheiten des zu untersuchenden Unternehmens aneignet und daraus Problemstellungen für das Data Mining ableitet (siehe Abbildung 8). Dazu werden vordergründlich die Ziele des Unternehmens sowie darunter liegende Einflussfaktoren herausgestellt und dabei potenzielle Überschneidungen und Gegensätze identifiziert. Im Anschluss wird dieses Wissen durch die für das Data Mining Projekt relevanten, spezifischen Informationen der aktuellen Situation erweitert, indem Ressourcen, Anforderungen und Risiken erörtert werden. Auf dieser Wissensgrundlage basierend, werden dann die Ziele des Data Mining Projektes aufgestellt und ein Projektplan generiert (vgl. Chapman, et al., 2000, S. 14 ff.).

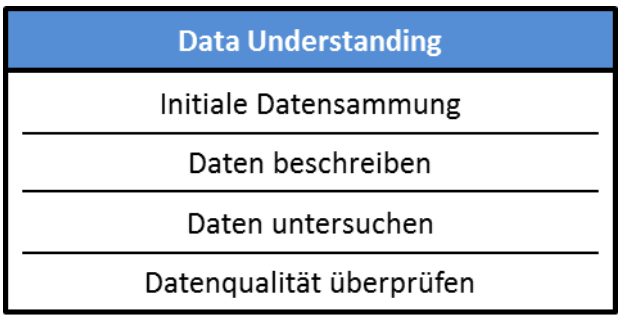


Abbildung 8: Aufgaben in der Phase Data Understanding (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

In der anschließenden Phase „**Data Understanding**“ verschafft sich der Analyst einen Überblick über die initiale Datensammlung des Unternehmens (siehe Abbildung 8). Dazu akquiriert der Analyst zuerst die Daten aus den jeweiligen Quellen. Dann werden die Daten bezüglich Format, Menge sowie Bedeutung beschrieben und weiterhin mittels Visualisierungen, Abfragen sowie statistischen Kennzahlen erörtert, um Hypothesen aufstellen zu können. Zuletzt wird außerdem die Datenqualität in Bezug auf fehlende Werte sowie Fehler und deren Ursprung überprüft. Hierbei besteht eine Interaktion zwischen den Phasen Business und Data Understanding, da das Wissen über das Unternehmen das Verständnis bezüglich der Daten fördert und umgekehrt (vgl. Chapman, et al., 2000, S. 17 ff.).

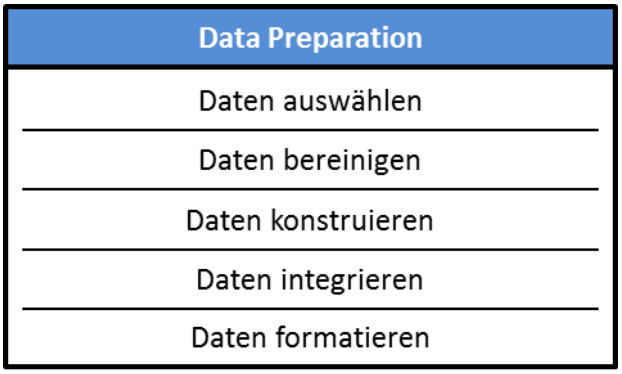


Abbildung 9: Aufgaben in der Phase Data Preparation (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

In der „**Data Preparation**“ Phase wird der initiale Datensatz durch zahlreiche Transformationen verändert, sodass auf Grundlage des transformierten Datensatzes Modellierungen stattfinden können (siehe Abbildung 9). Zu diesen Aktivitäten zählt zum einen das Auswählen von Variablen, welche für die herausgestellten Data Mining Ziele relevant sind. Diese werden nach Bedarf bereinigt, indem fehlende Werte behandelt, Teilmengen gebildet oder passende Werte erweitert werden. Darüber hinaus können gegebenenfalls neue Daten konstruiert werden, indem aus den existierenden Variablen durch Kombinationen oder Addition von Informationen neue Variablen erstellt werden. Da die Datengrundlage verschiedenen Quellen entspringen kann, ist in diesem Fall eine Integration der diversifizierten Daten notwendig. Mit Hinblick auf die Modellierung in der anschließenden Phase, müssen die Daten entsprechend der zu wählenden Modellierungsmethode formatiert werden. An dieser Stelle zeigt sich die Bedeutung des entsprechenden Doppelpfeils in Abbildung 6, denn die Aktivitäten in dieser Phase richten sich nach der Modellierung in der kommenden Phase, sodass hierbei ein interaktives Durchlaufen der beiden Phasen vorgesehen ist (vgl. Chapman, et al., 2000, S. 20 ff.).

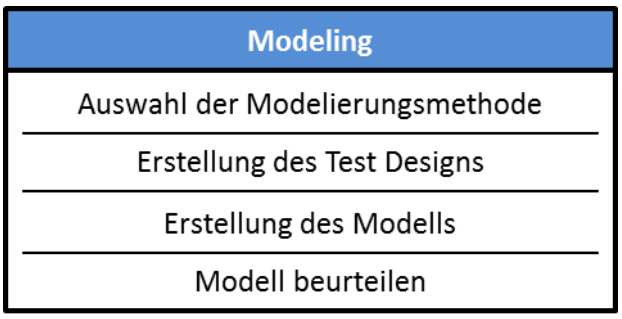


Abbildung 10: Aufgaben in der Phase Modeling (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

In der Phase „**Modeling**“ werden entsprechend der aufgestellten Ziele verschiedene Modellierungsmethoden angewendet, sodass diese Phase mehrmals und, wie bereits dargestellt, in

Interaktion mit der vorherigen Phase stattfindet (siehe Abbildung 10). Bezüglich einer Iteration wird zuerst eine Modellierungsmethode ausgewählt. Zur Auswahl stehen dabei Analysemethoden aus dem Data Mining (Assoziationsanalyse, Segmentierungen, Klassifizierungen, Text Mining, etc.). Im Anschluss wird ein „Test Design" der ausgewählten Methode unter Einbezug der vorliegenden Daten erstellt. Das Test Design dient zur Validierung und Prüfung des späteren Modells. Konkret wird dabei der vorliegende Datensatz in verschiedene Sets unterteilt. Neben dem Train-Set zur Generierung des Modells werden eine oder mehrere Test-Sets separiert. Es werden nun mittels des Train-Sets und der ausgewählten Analysemethode Modelle erstellt. Diese werden mittels der gleichen Analysemethode, jedoch mit verschiedenen Parametern berechnet. Die erhaltenen Modelle werden anschließend auf die Test-Sets angewendet, um die auf den Train-Sets basierenden Ergebnisse zu validieren. Auf diese Weise können Qualität und Fehlerraten der Modellvarianten herausgestellt werden. Die Auswahl und Dokumentation des geeigneten Modells anhand der Validierung stellt den abschließenden Prozess innerhalb eines Iterationszyklus dar. Zum Abschluss der Phase werden die in den verschiedenen Iterationszyklen erstellten Modelle in ein Ranking überführt (vgl. Chapman, et al., 2000, S. 23 ff.).

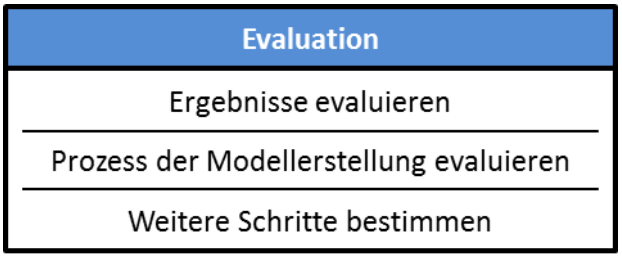


Abbildung 11: Aufgaben in der Phase Evaluation (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

In der anschließenden Phase „**Evaluation**" (siehe Abbildung 11) werden in einem ersten Schritt die Ergebnisse in Form von den aufgestellten Modellen mit Hinblick auf die aufgestellten Data Mining Ziele und deren Erfüllung der Unternehmensziele überprüft. Hierbei wird vor allem auch begutachtet, ob einige Modelle aus Sicht der Unternehmensziele besonders (un)geeignet sind. Außerdem wird in den Modellen nach Ergebnissen gesucht, welche sich nicht direkt auf die aufgestellten Ziele beziehen und als Nebenprodukt entstanden sind, diese würden sich für weitere Data Mining Prozesse anbieten. In einem weiteren Schritt wird der Prozess der Modellerstellung evaluiert. Hierbei geht es vordergründlich darum, aus Sicht des Data Mining die Erstellung der Modelle auf Basis der in den Test Designs erhaltenen Gütefaktoren zu eruieren und mögliche Versäumnisse oder Potenziale herauszustellen. Schließlich werden die Erkenntnisse aus den vorherigen Evaluierungen zu einem Gesamtbild zu-

sammengefügt, welches sowohl die Erfüllung der Unternehmensziele als auch Güte des Modellerstellungsprozesses beinhaltet. Dann wird entschieden, ob die Ergebnisse final implementiert werden, oder entsprechend des Pfeils zur initialen Business Understanding Phase unter Voraussetzung von genügend Zeit und Budget der Zyklus neugestartet wird. Falls hierbei die Unternehmensziele nicht zufriedenstellend erreicht wurden, Probleme oder Potenziale bei der Modellerstellung bestehen, oder besonders vielversprechende Ergebnisse als Nebenprodukt bei der erfolgreichen Modellierung vorliegen, ist ein derartiger Neustart des Zyklus naheliegend (vgl. Chapman, et al., 2000, S. 26 f.).

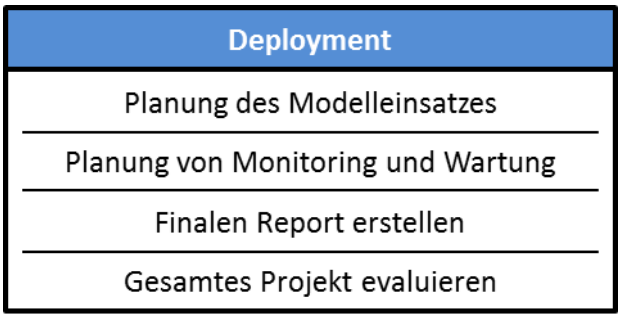


Abbildung 12: Aufgaben in der Phase Deployment (Quelle: eigene Darstellung in Anlehnung an Chapman et. al, 2000, S.10)

Andernfalls wird mit der finalen Phase „**Deployment**“ fortgefahren, in welcher die Ergebnisse zum einen dem Projektkunden verständlich gemacht und zum anderen unter dessen Einbezug in die Geschäftsprozesse eingebettet werden (siehe Abbildung 12). Dabei wird zuerst eine Deploymentstrategie entworfen, welche die notwendigen Schritte und Auswahl der zu präsentierenden Ergebnisse enthält sowie Ansätze zur deren Einbettung. Anschließend wird die Wartung und das Monitoring geplant, um während des späteren Einsatzes der eingebetteten Ergebnisse auf Fehler oder dynamische Änderungen des Umfeldes / der Ziele reagieren zu können sowie die Genauigkeit zu überprüfen. Die beiden vorherigen Planungsphasen werden dann in einen finalen Report zusammengefasst, welcher dem Kunden präsentiert wird und abschließend das gesamte Projekt evaluiert. Bei dieser Evaluierung geht es vor allem um interne Beurteilungen des Projektteams. Die jeweiligen Projektmitarbeiter sollen ihre Erfahrungen entlang des Projektes hinsichtlich besonders positiven und negativen Entwicklungen evaluieren, um Erkenntnisse für zukünftige Projekte zu sammeln (vgl. Chapman, et al., 2000, S. 28 f.).

2.2 Recommendation Engines

Recommendation Engines gehen auf die 1990er Jahre zurück (vgl. Hill, Stead, Rosenstein, & Furnas, 1995; Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Shardanand & Maes, 1995) und sind unter mehreren Synonymen wie Recommendation System / Platform“ oder „Recommender System“ bekannt. Eine RE wird eingesetzt, um innerhalb eines Systems mit zahlreichen Produkten / Items einem Nutzer gewisse Items vorzuschlagen, welche sich für einen Kauf / Nutzung anbieten. Häufig wird dies durch entsprechende Auflistungen an zentralen Stellen in dem jeweiligen Nutzerinterface bewerkstelligt (vgl. Ricci, Rockach, Shapira, & Kantor, 2011, S. 1). Durch diese Empfehlungen soll der Absatz an Items bzw. die Diversität der veräußerten Items gesteigert sowie die Kundenzufriedenheit erhöht werden, indem die zunehmende Informationsflut für den Kunden sinnvoll heruntergebrochen wird. Es existieren drei Möglichkeiten, wie Empfehlungen analytisch getroffen werden können. Dazu zählen die inhaltsbasierte, kollaborative sowie deren Kombination in Form von der hybriden Vorgehensweise (vgl. Adomavicius & Tuzhilin, 2005, S. 734 ff.).

Bei der **inhaltsbasierten Vorgehensweise** werden Empfehlungen auf Basis der Eigenschaften der Items getroffen (vgl. Ben Schafer, Frankowski, Herlocker, & Sem, 2007). Hierzu werden die Eigenschaften von Items in einer digitalisierten und maschinenlesbaren Form durch Keywords / Metadaten dargestellt. Einem Nutzer werden dann diejenigen Items empfohlen, welche die gleichen Eigenschaften wie bereits zuvor erstandene bzw. hoch bewertete Items besitzen. Bei diesem Ansatz ist es entscheidend, die Items innerhalb des Systems zu charakterisieren, denn die Empfehlungen werden basierend auf der charakteristischen Gleichheit von Items getroffen. Ein Beispielgebiet für den Einsatz solcher REs sind aktuelle Musik-Streamingdienste. Items sind hierbei einzelne Musiktitel, -alben oder –künstler, welche durch Metadaten wie Genre, Musiktexte oder Struktur des Audiosignals angereichert sind. Der Nutzer kann innerhalb des Systems Titel, Alben oder Künstler als Favorit markieren, welche dann in einem personalisierten Bereich in Playlisten eingeordnet und genutzt werden können. Die jeweiligen Metadaten der markierten Items werden dabei dem Nutzerprofil zugeordnet und es werden anschließend dem Nutzer Musiktitel mit den ähnlichen Metadaten empfohlen. Gerade in dem Bereich des Musikstreamings haben sich inhaltsbasierte REs als kritischer Erfolgsfaktor herausgestellt. Denn die Auswahl an jeweils verfügbaren Musiktiteln unterscheidet sich bei den diversen Anbietern marginal, gut funktionierende REs als Service zum optimierten Auffinden von Musiktiteln in der enormen Musikwelt haben sich insofern als Abgrenzungsfaktor entwickelt.

Bei der **kollaborativen Vorgehensweise** müssen keine Eigenschaften von Items bekannt sein, denn diese werden basierend auf in der Vergangenheit gesammelten Informationen empfohlen (vgl. Sarwar, Karypis, Konstan, & Riedl, 2001). Diese Informationen können explizit (= bewusste Nutzereingabe) oder implizit (= unbewuste Nutzereingabe) erhoben werden. Zu den Expliziten gehören bspw. Nutzerratings, Sucheingaben oder spezfische Itemlisten (z.B. Playlist bei einem Streamingdienst). Implizite Informationen wären hierbei Dauer der Betrachtung von Itemseiten oder Bestellungsdaten. Empfehlungen werden dann aus der Gesamtheit an Nutzerinformationen bezüglich Bestellungen von Items generiert (vgl. Ricci, Rockach, Shapira, & Kantor, 2011, S. 11 f.). Hierbei muss es im Gegensatz zu der inhaltsbasierten Vorgehensweise eine kritische Grundmenge in einer beachtlichen Größe vorliegen (vgl. Schein, Pepescul, Ungar, & Pennock, 2002), andernfalls können keine genauen Empfehlungen ausgesprochen werden („Cold Start Problem“).

Bei diesem Ansatz liegt die Herausforderung darin, die immensen Datenmengen performant zu bearbeiten und ohne inhaltliche Kenntnisse der Items passende Empfehlungen durch Eingrenzung der Analyseergebnisse zu treffen. Als Beispiel dienen hierbei Produktempfehlungen von Händlern. Items sind in diesem Fall Produkte und die Informationsgrundlage ist die Bestelldatenbank. Die Empfehlungen werden hierbei anhand von gekauften Warenkörben getroffen. Wenn bspw. zwei Produkte besonders oft zusammen bestellt wurden, werden diese potenziell gegenseitig empfohlen, wenn eines der Produkte in den Warenkorb gelegt wird. Hier zeigt sich der Unterschied zum inhaltsbasierten Ansatz. Denn es bestehen keine Informationen, ob Produkte inhaltlich zueinander passen. Wenn bspw. eine signifikante Anzahl an Bestellungen Kaffee und ein Smartphone enthalten, so wird potenziell beim Kauf von Kaffee ein Smartphone empfohlen – auch wenn dies inhaltlich wenig sinnvoll ist. Insofern offenbart sich die Herausforderung der kollaborativen RE. Es müssen enorme Mengen an Daten performant analysiert und ohne inhaltliche Kenntnis bezüglich der sinnvollen Verknüpfung mittels statistischen Parametern Empfehlungen getroffen werden.

Bei der **Hybriden Vorgehensweise** werden die beiden Vorherigen kombiniert, um verschiedene Problemstellungen zu eliminieren. Eine inhaltsbasierte RE bietet sich bspw. als Überbrückung des „Cold Start Problems“ zu Beginn eines Systems an, wenn keine ausreichenden Daten für eine kollaborative Engine vorliegen. Umgekehrt wäre es auch möglich, die potenziell ungenauen Ergebnisse einer kollaborativen Engine durch inhaltsbasierte Methoden zu verfeinern. Insofern kann entweder durch die Kombinierung der beiden separaten Systeme oder durch die partielle Anreicherung eines Systems mit Methoden des anderen Ansatzes Probleme

behoben und die Güte der Empfehlungen optimiert werden (vgl. Adomavicius & Tuzhilin, 2005, S. 740 f.).

2.3 Assoziationsanalyse

Der Ursprung der Assoziationsanalyse geht auf die erstmalige Erwähnung im Jahr 1993 zurück, in welchem der Begriff zuerst als funktioneller Teilbereich einer Betrachtung bezüglich effizientem Untersuchen von großen Datenbanken auftrat (vgl. Agrawal, Imielinski, & Swami, 1993a). Im gleichen Jahr wurde daraufhin eine separate Publikation zu dem eigenständigen Konstrukt der Assoziationsregeln veröffentlicht (vgl. Agrawal, Imilienski, & Swami, 1993b). Bis zum heutigen Zeitpunkt wurde die Assoziationsregelanalyse vor allem zur Warenkorbanalyse eingesetzt, das grundsätzliche Prinzip ist jedoch auch in weiteren Bereichen anwendbar (vgl. Brin, Motwani, & Silverstein, 1997) und wurde deshalb auch in IT-Sicherheit, Bio-Informatik sowie Produktionsmanagement angewendet.

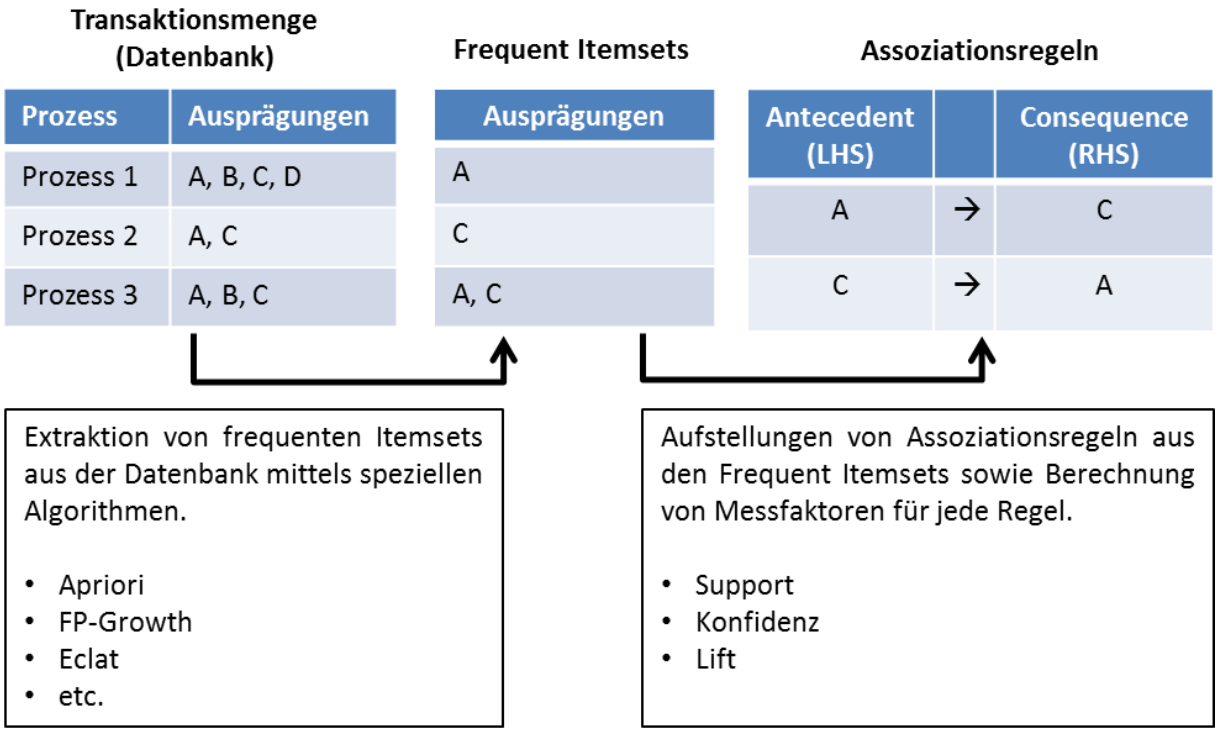


Abbildung 13: Übersicht der Assoziationsanalyse

Im Grundsatz eignet sich die Assoziationsregelanalyse dazu, in einer gegebenen Datenbank mit Prozessen und Ausprägungen Assoziationen zwischen den Ausprägungen in Form von Regeln zu finden (vgl. Agrawal, Imielinski, & Swami, 1993a, S. 917 f.). Dazu wird im Detail die Datenbank mittels eines gewählten Algorithmus untersucht und „frequente Itemsets“ extrahiert (vgl. Agrawal, Imilienski, & Swami, 1993b, S. 2 f.). Dies sind Kombinationen aus verschiedenen Ausprägungen, welche in einer bestimmten Häufigkeit in der Datenbank vorkommen. Aus diesen Itemsets werden dann Assoziationen in Form von Regeln erstellt (siehe

Abbildung 13) (vgl. Agrawal, Imilienski, & Swami, 1993b, S. 922 f.). Für diese Regeln werden drei maßgebliche Messfaktoren berechnet, welche deren Bedeutung widerspiegeln. Im weiteren Verlauf dieses Kapitels wird die Vorgangsweise von der Datenbankanalyse bis hin zu der Berechnung und Interpretation der Messfaktoren (Support, Konfidenz, Lift) erläutert.

Frequent Itemsets

Ein „Itemset" beschreibt eine Kombination aus Produkten, welche innerhalb einer Datenbank vorkommt. Dieses wird dann „frequent", wenn die Anzahl an Bestellungen, welches die entsprechende Produktkombination (folgend als „Itemset" geführt) enthält, in einer bestimmten Häufigkeit relativ zu der Gesamtanzahl an Bestellungen vorliegt (vgl. Borgelt, 2012). Diese Häufigkeit wird durch den ersten Messfaktor „Support" berechnet, dieser definiert sich wiefolgt (vgl. Zhao & Bhowmick, 2003, S. 6 f.):

$$\mathbf{Support}^{(X)} = \frac{|\{t \subseteq D; X \subseteq t\}|}{|D|}$$

„X" steht hierbei für die Ausprägung X, welche Teil einer Transaktion „t" ist, welche wiederum Teil einer Datenbank „D" ist. Zur Berechnung des Supports eines Itemsets zählt man demnach die Transaktionen, in welcher das Itemset vorkommt, und teilt diese Zahl durch die Gesamtanzahl an Transaktionen der Datenbank. Man erhält somit den Support bzw. das relative Vorkommen des Itemsets in der Datenbank. Der Supportwert liegt somit generell im Wertebereich von [0, 1].

Der erste Schritt einer Assoziationsregelanalyse ist die Extraktion aller frequenten Itemsets aus der Datenbank. Aufgrund der Größe heutiger Datenbanken wird vor der Extraktion der Itemsets ein Minimalsupport im Sinne der „bestimmten Häufigkeit" vorgegeben. Konkret bedeutet dies, dass vor der eigentlichen Extraktion mittels eines Minimalsupports von bspw. 0.3 lediglich jene Itemsets gebildet werden, welche in mindestens in 30% aller Bestellungen der Datenbank vorkommen (vgl. Agrawal & Srikant, 1994, S. 489). Somit wird die Menge an zu berechnenden Itemsets reduziert und die Analyse beschleunigt, jedoch fallen auf diesem Wege viele Ausprägungen ohne tiefere Begutachtung aus der Analyse heraus. Da jedoch auch Ausprägungen mit niedrigen Supportwerten sinnvolle Assoziationen enthalten können, wird der Minimalsupport in der Praxis niedrig angesetzt.

Die Bildung der Itemsets unter Berücksichtigung des Minimalsupports wird mittels eines Algorithmus vorgenommen. Es existiert eine Fülle an Algorithmen, welche sich unter anderem durch deren Speicher bzw. Prozessorauslastung sowie Anwendungsfeld unterscheiden. Des Weiteren können bestimmte Datenstrukturen und/oder spezielle Zielstellungen in Bezug auf den Analyseprozess mit verschiedenen Algorithmen spezifisch bedient werden. Der Greif-

barste und im späteren Prototyp aufgrund einer speziellen Eigenschaft (= Vorgabe einer spezifischen LHS) Verwendete ist der „Apriori"- Algorithmus, welcher mittels Kandidatengenerierung Itemsets extrahiert (vgl. Agrawal & Srikant, 1994, S. 492 ff.). Der Apriori Algorithmus iteriert mehrmals über eine Datenbank und generiert bei jeder Iteration Kandidaten für Itemsets mit einer gewissen Länge. In der ersten Iteration werden alle Itemsetkandidaten der Länge 1 gebildet. Es werden also für alle einzelnen Ausprägungen der Datenbank die jeweiligen Supportwerte berechnet. Diese Itemsets sind jedoch nicht frequent, sondern lediglich „Kandidaten", denn es wurde noch kein Abgleich mit dem Mindestsupport vorgenommen. Erst in der nachfolgenden Iteration, in welcher die Itemsets der Länge 2 gebildet werden, wird dies bewerkstelligt. Denn hier werden auf Grundlage der Itemsets der Länge 1 durch Kombinationen die Itemsets der Länge 2 gebildet. Bei der Aufstellung einer Kombination aus zwei Itemsets der Länge 1 wird nun der Mindestsupport abgeglichen. Dies bedeutet konkret, dass lediglich solche Kombinationen berechnet werden, in welchen beide Itemsets der Länge 1 den Mindestsupport erfüllen und somit frequent sind. Denn der maximale Support eines Itemsets ist gleichbedeutend mit dessen der niedrigsten Ausprägung. Daraus folgt, dass jedes frequente Itemset zwangsläufig aus frequenten Itemsets bestehen muss („schwächste Glied einer Kette"). Dieser Vorgang wird anschließend auf gleiche Weise für Itemsets der Länge n fortgeführt, bis alle möglichen Kombinationen gebildet wurden. Dies zeigt die grundsätzliche „Kandidatengenerierung" auf, weil der Algorithmus zuerst alle Itemsets einer Länge im Sinne von „Kandidaten" aufstellt sowie den jeweiligen Support berechnet und erst bei dem Übergang zu der nachfolgenden Iteration anhand des vorgegebenen Minimalsupports Kandidaten verwirft.

Zur Berechnung aller Itemsets einer Datenbank ist der Apriori Algorithmus im Vergleich zu anderen Algorithmen aufgrund der Kandidatengenerierung und potenziell unnötigen Berechnungen ineffizient. Außerdem wird bei jeder Iteration des Algorithmus die Datenbank vollständig durchlaufen, sodass in Abhängigkeit von der Anzahl der Items, deren Supportwerte sowie des Minimalsupports die Anzahl der Iterationen und aufwändigen Datenbankdurchläufe sehr hoch ausfallen kann (vgl. Han, Pei, & Yin, 2000, S. 1 f.). Im Laufe der Zeit wurden deshalb effizientere Algorithmen entworfen, welche weniger Datenbankiterationen vornehmen und eine restriktivere Anwendung des Minimalsupports bewerkstelligen. Die am weitesten Verbeiteten sind hierbei der „Freuqent Pattern Growth"-Algorithmus („FP-Growth") (vgl. Han, Pei, & Yin, 2000) sowie der „Eclat"-Algorithmus (vgl. Zaki, 2000). Neben diesen Alternativen existieren weitere Algorithmen, welche durch verschiedenen Such- und Strukturmethoden Itemsets extrahieren. Diese unterscheiden sich zwar grundsätzlich durch deren Metho-

den, jedoch liefern alle Algorithmen bei gleicher Vorgabe des Mindestsupports die gleichen frequenten Itemsets, welche anschließend für die Regelaufstellungen verwendet werden.
Diese Algorithmen werden an dieser Stelle nicht näher beleuchtet, da der Apriori Algorithmus trotz dessen grundsätzlich ineffizienten Charakters bei der Berechnung aller Itemsets einer Datenbank Vorteile hinsichtlich einer partiellen Echtzeitberechnung von Itemsets beinhaltet und aus diesem Grund im Prototyp zum Einsatz kommt. Dies wird im späteren Entwicklungsteil näher beleuchtet.

Regelaufstellung und Messfaktorenberechnung

Der zweite Schritt innerhalb einer Assoziationsregelanalyse ist die Aufstellung von Regeln aus den Itemsets mit einer Länge größer 1 (vgl. Tan, Steinebach, & Kumar, 2006, S. 349 ff.). Die Itemsets der Länge 1 dienen lediglich der Berechnung der weiteren Messfaktoren. Abbildung 14 ist die Grundstruktur einer Regel zu entnehmen, bestehend aus „Antecedent“ bzw. „Left-Hand-Side“ (LHS) sowie Consequence bzw. „Right-Hand-Side“ (RHS). Eine solche Regel ist zu lesen als „wenn X vorliegt, empfiehlt sich Y“.

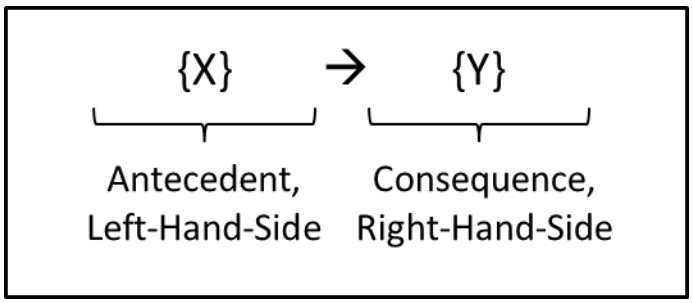


Abbildung 14: Grundstruktur einer Assoziationsregel

Bei der Aufstellung von Regeln aus einem Itemset ist zu beachten, dass stets alle Kombinationen der Teile eines Itemsets gebildet werden. Aus einem Itemset {X, Y} werden somit die Regeln {X} → {Y} und {Y} → {X} aufgestellt. Daraus folgt, dass aus Itemsets, welche eine Anzahl von Z Items enthält, bis zu 2^z-2 Regeln erstellt werden. Diese Notwendigkeit offenbart sich im weiteren Verlauf bei der Berechnung des nächsten Messfaktors der „Konfidenz“.
Die Konfidenz misst die Eintrittswahrscheinlichkeit einer Regel und ist definiert als (vgl. Zhao & Bhowmick, 2003, S. 7 f.):

$$\mathbf{Konfidenz}(\mathbf{X} \rightarrow \mathbf{Y}) \quad = \frac{Support(X \cup Y)}{Support(X)}$$

Man dividiert den Support des gesamten, der Regel zu Grunde liegenden Itemsets und teilt diesen durch den Support des Itemsets der LHS. Es verringert/vergrößert sich die Konfidenz

somit bei steigendem/fallendem Support von X. Um dies zu durchdringen, lohnt ein Blick auf zwei konkrete Situationen. Wenn der Support von X gleich dessen des gesamten Itemsets ist, dann war in jeder Transaktion von X auch Y enthalten. Der Konfidenzwert wäre in diesem Falle 1, was gleichbedeutend das Optimum in Form einer Wahrscheinlichkeit von 100% darstellt. Wenn jedoch der Support von X höher als dessen des gesamten Itemsets liegt, dann stellt die Differenz der Supportwerte Transaktionen dar, in welcher X ohne Y auftrat. Dies äußert sich durch einen höheren Wert im Nenner und verringert somit den Konfidenzwert. Insofern ist die Konfidenz ein zentraler Messfaktor, welcher die Wahrscheinlichkeit des Eintretens der Regel beschreibt. Je höher die Konfidenz, desto wahrscheinlicher ist das Eintreten einer Regel.

Aufgrund der Bildung von Assoziationsregeln aus allen Kombinationen eines Itemsets und dem Umstand, dass der Beispieldatensatz im Vergleich zu Datenbanken in der Praxis verschwindend klein ist, liegt auf der Hand, dass die insgesamte Regelanzahl trotz Minimalvorgabe des Supports in der Praxis immense Größen erreichen kann. Aus diesem Grund wird der Messfaktor Konfidenz abermals als Reduzierungsfaktor in Form einer Minimalvorgabe zu Beginn des Assoziationsregelverfahrens eingesetzt.

Der verbleibende Messfaktor „Lift“ bestimmt die Abweichung des Zustandekommens der Regel von der statistischen Zufälligkeit und ist somit nahe der Signifikanz- und Korrelationstests aus der Inferenzstatistik zuzuordnen (vgl. Brin, Motwani, & Tsur, 1997). Dieser Definiert sich wie folgt:

$$\begin{aligned}\mathbf{Lift(X \rightarrow Y)} &= \frac{Konfidenz(X \rightarrow Y)}{Support(Y)} \\ &= \frac{Support(X \cup Y)}{Support(X) * Support(Y)} \\ &= \frac{P(X \wedge Y)}{P(X) * P(Y)}\end{aligned}$$

Man teilt die Konfidenz einer Regel durch den Support des Itemsets der RHS, der Wertebereich liegt bei [0, ∞]. Auch hier lohnt ein Verständnis fördernder Blick auf konkrete Konstellationen. Wenn der Konfidenzwert der Regel gleich dem Support von Y ist, so wird das Zustandekommen dieser Konstellation als zufällig erachtet. Insofern beschreibt ein korrespondierender Liftwert von 1, dass X und Y genauso häufig auftreten, wie es bei geltender Unabhängigkeit der Ausprägungen unter Zufall zu erwarten wäre. Folgerichtig wäre eine solche

Regel zu verwerfen, da diese eine Korrelation zwischen X und Y und damit Abhängigkeit unterstellt. Wenn der Support von Y höher liegt als die Konfidenz, so erhält man einen Liftwert < 1. Dies bedeutet, dass X und Y weniger häufig auftreten, als es bei geltender Unabhängigkeit von X und Y durch Zufall zu erwarten wäre. Ein Liftwert kleiner 1 beschreibt also eine negative Korrelation von X und Y. Da dies das Gegenteil der Aussage der aufgestellten Regel ist, sind Regeln mit Liftwerten kleiner 1 unbedingt zu verwerfen. Wenn der Support von Y kleiner ist als die Konfidenz der Regel, treten X und Y analog häufiger auf, als es bei Unabhängigkeit unter Zufall zu erwarten wäre. Es wird also mit einem Liftwert größer 1 eine positive Korrelation von X nach Y beschrieben, was der Intention der Regel entspricht. Je nach Konstellation der Ausprägungen in der Datenbank kann der Liftwert potenziell nahezu unendlich sein, wobei gilt, dass bei steigendem Liftwert die Bedeutung der Regel zunimmt. In dem Analyseprozess wird der Lift nicht wie Support und Konfidenz als Minimalparameter genutzt, sondern auf die potenziell finale Regelmenge angewendet, um anhand der Korrelationen wichtige Regeln herauszustellen.

Beispiel

Im Folgenden wird ein vollständiger Analyseprozess anhand eines Beispiels mittels des Apriori Algorithmus durchgeführt. Der Mindestsupport wird hierbei mit **0.4**, die Minimalkonfidenz mit **0.8** festgelegt. Die Höhe der Werte wurde so gewählt, damit in dem Beispiel sinnvolle Konstellationen zum Aufzeigen der Vorgehensweise entstehen.

Tabelle 1: Beispieldatensatz zur Assoziationsregelanalyse

Einzelne Bestellungen

		Produkte (Items)			
		Rasierer	Rasier-klingen	Zahn-bürste	Haar-föhn
Bestellung	1	x	x	x	
	2		x		
	3		x		
	4	x	x	x	
	5	x	x		x
	6	x	x	x	
	7	x	x	x	
	8			x	x
	9	x	x		
	10				x
Σ	**10**	**6**	**8**	**5**	**3**

Produktkombinationen

Produkte (Itemsets)			Bestellungen	Σ
Rasierer	Rasier-klinge	Zahn-bürste		
x			1, 4 - 7, 9	**6**
	x		1 - 7, 9	**8**
		x	1, 4, 6 - 8	**5**
x	x		1, 4, 5 - 7, 9	**6**
x		x	1, 4, 6, 7	**4**
	x	x	1, 4, 6, 7	**4**
x	x	x	1, 4, 6, 7	**4**

Tabelle 1 enthält einen Beispieldatensatz mit zehn Bestellungen sowie vier Produkten: Rasierer, Rasierklingen, Zahnbürste sowie Haarföhn. Auf der linken Seite ist eine tabellarische Ansicht der einzelnen Bestellungen zu sehen, wobei auf der Vertikalen die Produkte und auf der Horizontalen die Bestellungen aufgetragen sind. Ein „x“ bedeutet hierbei, dass das jeweilige Produkt in der Bestellung enthalten war. Bestellung 1 enthält somit die drei Produkte Rasierer, Rasierklinge sowie Zahnbürste. Am unteren Ende der Tabelle sind außerdem die Summen der einzelnen Produkte zu finden.

Der rechte Teil von Tabelle 1 zeigt eine alternative Ansicht der Bestellungen mit Hinblick auf die möglichen Kombinationen von Produkten. Hierbei sind auf der Vertikalen Informationen bezüglich der Produktkombinationszusammensetzung, der Zuordnung zu den Bestellungen sowie der entsprechenden Summe und auf der Horizontalen die Produktkombinationen aufgetragen. Ein „x“ bedeutet hierbei, dass das jeweilige Produkt innerhalb der Produktkombination enthalten ist. Die ersten drei Zeilen stellen hierbei die einzelnen Produkte und deren Verortung in den Bestellungen dar, sodass hierbei die Summen denen des linken Teils der Tabelle entsprechen. In den weiteren Zeilen sind Kombinationen der Produkte enthalten, sodass beispielsweise die Kombination „Rasierer, Zahnbürste“ aus Zeile 5 in den Bestellungen 1, 4, 6 und 7 (der linken Seite), also insgesamt vier Mal bestellt wurde.

Im ersten Schritt des Apriori Algorithmus werden die Supportwerte der Itemsets der Länge 1 berechnet. Bezüglich des Beispieldatensatzes entspricht dies den Summen in Tabelle 1, geteilt durch die Anzahl der Gesamtbestellungen von 10. Es ergeben sich folgende Supportwerte:

Tabelle 2: Itemsetkandidaten nach der ersten Apriori-Iteration

Itemset	{Rasierer}	{Rasierklinge}	{Zahnbürste}	{Haarfön}
Support	0,6	0,8	0,5	0,3

Anhand der roten Markierung ist zu erkennen, dass das Produkt Haarfön unterhalb des Mindestsupports von 0,4 liegt und deshalb bei den weiteren Iterationen nicht einbezogen wird. In der zweiten Iteration werden nun die Itemsets der Länge 2 durch Kombination der frequenten Itemsets der ersten Iteration (Rasierer, Rasierklinge, Zahnbürste) gebildet. Hierbei werden die Summen aus dem rechten Teil von Tabelle 1 durch die Transaktionsgesamtanzahl von 10 geteilt (Tabelle 2).

Tabelle 3: Itemsetkandidaten nach der zweiten Apriori-Iteration

Itemset	**Support**

{Rasierer, Rasierklingen}	0,6
{Rasierer, Zahnbürste}	0,4
{Rasiererklingen, Zahnbürste}	0,4

In der dritten Iteration werden alle Itemsets der Länge 3 gebildet. In dem Beispieldatensatz ist dies lediglich ein Itemset {Rasierer, Rasierklingen, Zahnbürste} mit einem Support von 0,4. Hierbei wird kein Kandidat aus der zweiten Iteration verworfen, da alle dortigen Kandidaten den Mindestsupport erfüllen. Für den Beispieldatensatz ist die Aufstellung der frequenten Itemsets nun abgeschlossen, da keine weiteren Kombinationen aus Itemsets mehr gebildet werden können. Tabelle 2 enthält alle frequenten Itemsets aus dem Beispieldatensatz.

Tabelle 4: Frequente Itemsets aus dem Beispieldatensatz

Itemset	Support
{Rasierer}	0,6
{Rasierklingen}	0,8
{Zahnbürste}	0,5
{Rasierer, Rasierklingen}	0,6
{Rasierer, Zahnbürste}	0,4
{Rasiererklingen, Zahnbürste}	0,4
{Rasierer, Rasierklingen, Zahnbürste}	0,4

Tabelle 5: Regelaufstellungen des Beispieldatensatzes

Assoziationsregeln			Messfaktoren		
Antecedent (LHS)		**Consequence (RHS)**	**Support**	**Konfidenz**	**Lift**
{Rasierer}	→	{Rasierklingen}	0,6	**1,00**	1,25
{Rasierklingen}	→	{Rasierer}	0,6	0,75	1,25
{Rasierer}	→	{Zahnbürste}	0,4	0,60	-
{Zahnbürste}	→	{Rasierer}	0,4	0,80	1,33
{Rasierklingen}	→	{Zahnbürste}	0,4	0,50	-
{Zahnbürste}	→	{Rasierklingen}	0,4	0,80	1,00
{Rasierer}	→	{Rasierklingen, Zahnbürste}	0,4	0,67	-
{Rasierklingen, Zahnbürste}	→	{Rasierer}	0,4	**1,00**	1,67
{Rasierklingen}	→	{Rasierer, Zahnbürste}	0,4	0,50	-
{Rasierer, Zahnbürste}	→	{Rasierklingen}	0,4	**1,00**	1,25
{Zahnbürste}	→	{Rasierer, Rasierklingen}	0,4	0,80	1,33
{Rasierer, Rasierklingen}	→	{Zahnbürste}	0,4	0,67	-

Aus den frequenten Itemsets werden nun Regeln aufgestellt und die jeweiligen Messfaktoren berechnet. Dies wird anhand des Itemsets {Rasierer, Rasierlingen} exemplarisch aufgezeigt.

Es werden zuerst aus dem Itemset die beidseitigen Regeln aufgestellt. In diesem Fall also {Rasierer}→{Rasierklingen} und {Rasierklingen}→{Rasierer}. Betreffend der ersten Regel {Rasierer}→{Rasierklingen} dividiert man den Support des gesamten Itemsets {Rasierer, Rasierklingen} von 0,6 durch den Support des Itemsets der LHS {Rasierer} von 0,6. Man berechnet somit 0,6 / 0,6 = **1,0**. Dieser Konfidenzwert ist zugleich der Optimalfall und kann als „ein Kunde, welcher einen Rasierer kauft, kauft zu 100% Wahrscheinlichkeit auch Rasierklingen" interpretiert werden. Wenn man dieses Schema analog auf die zweite Regel {Rasierklingen}→{Rasierer} anwendet, wird ersichtlich, wieso eine beidseitige Regelgenerierung aus dem Itemset {Rasierer, Rasierklingen} notwendig war. Denn während die Konfidenz der Regel {Rasierer}→{Rasierklingen} **1,0** beträgt, ergibt die Rechnung bezüglich der Regel {Rasierklingen}→{Rasierer} 0,6 / 0,8 = **0,75**. Dies bedeutet, dass in nur 75% der Fälle, in welchen Rasierklingen gekauft wurden, auch ein Rasierer gekauft wurde. In 25% der Fälle wurden Rasierklingen ohne Rasierer gekauft, was den niedrigeren Konfidenzwert erklärt. Hier würde die Interpretation „wer Rasierklingen kauft, kauft zu 75% auch einen Rasierer" lauten. Dies offenbart den potenziellen Unterschied zwischen verschiedenen Regeln, die aus dem gleichen Itemset aufgestellt wurden. Tabelle 3 sind alle analog berechneten Konfidenzwerte zu entnehmen, wobei alle Werte unterhalb der Mindestkonfidenz von 0,7 rot markiert sind.
Im nächsten Schritt werden die Liftwerte berechnet. Bezogen auf die Beispielregel {Rasierer}→{Rasierklingen} würde die Rechnung demnach Konfidenz (1,0) dividiert durch den Support der RHS (0,8), also 1,0 / 0,8 = 1,25 betragen. Dies deutet auf eine positive Korrelation von {Rasierer} nach {Rasierklingen} hin, bei höherem Vorkommen von {Rasierer} steigt also das Vorkommen von {Rasierklingen}. Bezüglich des Liftwertes kann diese Regel somit als nützlich erachtet werden. Tabelle 3 enthält alle analog berechneten Liftwerte, wobei dies nur für diejenigen Regeln durchgeführt wurde, welche nicht durch die Mindestkonfidenz ausgeschlossen wurden.

Tabelle 6: Finale Regeln aus dem Beispieldatensatz

Assoziationsregeln			Messfaktoren		
Antecedent (LHS)		**Consequence (RHS)**	**Support**	**Konfidenz**	**Lift**
{Rasierer}	→	{Rasierklingen}	0,6	**1,00**	1,25
{Rasierklingen, Zahnbürste}	→	{Rasierer}	0,4	**1,00**	1,67
{Rasierer, Zahnbürste}	→	{Rasierklingen}	0,4	**1,00**	1,25
{Zahnbürste}	→	{Rasierer, Rasierklingen}	0,4	0,80	1,33
{Zahnbürste}	→	{Rasierer}	0,4	0,80	1,33
{Rasierklingen}	→	{Rasierer}	0,6	0,75	1,25

Es verbleiben nach Berücksichtigung der Minimalkonfidenz sowie den Liftwerten die in Tabelle 6 aufgelisteten Regeln. Mittels der Messfaktoren können entsprechend die Bedeutungen der Regeln ermittelt und durch deren Interpretation Aktionen vollzogen werden. Eine Möglichkeit ist hierbei die bereits vorgestellte RE, welche beispielsweise auf einer Produktseite die RHS aller Regeln als Kaufoption empfiehlt, welche das angezeigte Produkt in der LHS enthalten (bspw. anzeigen von Rasierklingen auf der Produktseite Rasierer).

2.4 Visualisierung von Assoziationsregeln

In diesem Kapitel wird eine Auswahl an Visualisierungsmethoden vorgestellt, welche dazu dienen, Assoziationsregeln inklusive deren Messfaktoren auf eine verständliche Art und Weise darzustellen. Es existiert eine Fülle an Visualiserungsmethoden (siehe Tabelle 7), welche jedoch überwiegend spezifisch dazu entwickelt wurden, Analysten während der Assoziationsanalyse bei der Behandlung der immensen Regelanzahlen zu unterstützen (vgl. Bruzzese & Davino, 2008). Hierbei standen vor allem die Interaktionsmöglichkeit sowie der Anspruch zur Darstellung großer Regelmengen mit zahlreichen Informationen im Vordergrund. Die Selektion der in dieser Arbeit behandelten Visualisierungsmethoden wurde jedoch mit Bezug auf die Thematik der Darstellung von kleineren Regelmengen innerhalb einer RE für Personen ohne Expertise über Assoziationsregeln getroffen. Zu den schnell verständlichen, die vorgestellten Messfaktoren übersichtlich darstellenden und auf kleinere Regelmengen ausgelegten Visualisierungsmethoden zählen Tabellen, 3D-Matrizen, Graphen sowie Parallelkoordinaten.

Tabelle 7: Übersicht aktueller Visualisierungsmethoden

	Darstellbare Regelmenge	Regeltyp	Komplexität
Tabellen	hoch	alle	niedrig
Scatterplot	hoch	one-to-one many-to-one	mittel
Matrizen	mittel	one-to-one many-to-one	niedrig
Parallelkoordinaten	mittel	alle	mittel
Graphen	niedrig	one-to-one many-to-one	niedrig
Mosaikmuster	niedrig	one-to-one many-to-one	hoch

Tabellen

Die einfachste und zugleich intuitivste Visualisierung ist nach eigener Auffassung jene der Tabelle. Diese Form wurde bereits im vorangegangenen Kapitel ausgiebig genutzt. Tabelle 6 dient hierbei als Beispiel für eine Visualisierung von den sechs Assoziationsregeln aus dem Beispieldatensatz. Hierbei werden die in LHS und RHS befindlichen Itemsets jeweils in einer Spalte verortet und darüber hinaus ebenfalls spaltenweise die Messfaktoren aufgetragen. Die in dieser Visualisierungsmethode ermöglichte Sortierung trägt weiterhin zu dem Durchdringen der Regeln bei (vgl. Bruzzese & Davino, 2008, S. 107; Sekhavat & Hoeber, 2013, S. 36). Jedoch sind in dieser Visualisierungsmethode alle Informationen in Textform vorhanden, was ein besonders schnelles Verstehen zumindest erschwert. Daher bieten sich Tabellen vor allem zusätzlich zu anderen Visualisierungen an, sodass mittels einer weiteren Visualisierungsmethode durch abstrahierte Darstellungen der Informationen ein schnelles Verständnis erwirkt wird und die jeweiligen Detailinformationen einer Tabelle entnommen werden können. Dieser Ansatz hat sich bereits in mehreren Visualisierungsimplementierungen von Assoziationsregeln als erfolgreich erwiesen und wird folgerichtig auch in dieser Arbeit verfolgt (vgl. Sekhavat & Hoeber, 2013). In der zu entwickelnden RE soll demnach eine Kombinierte Visualisierung mittels einer Tabelle sowie einer weiteren Notation mit höherem Visualisierungsgrad enthalten sein.

3D-Matrizen

3D-Matrizen visualisieren kleine bis mittlere Regelanzahlen in einer dreidimensionalen Graphenstruktur (siehe Abbildung 15). Hierbei befinden sich die Items auf der Horizontalen und aus den Markierungen in der Vertikalen wird die jeweilige Regel zusammengesetzt. Die Markierungen sind durch Farbgebung in zwei Kategorien zu unterscheiden, wobei die Kategorien jeweils für die LHS und RHS stehen. Die Messfaktoren sind als in die Höhe ragende Balken am oberen Ende des Graphen abgebildet und werden durch verschiedene Farben in Support und Konfidenz unterteilt. In der Beispielabbildung wurden jeweils farbliche Differenzierungen genutzt (siehe Legende in Abbildung 15). Die Höhe der Messfaktoren darstellenden Balken wird außerdem durch geeignete Linien mit den Messwerten versehen. Insgesamt wird eine Regel somit durch das vertikale Lesen der Matrix sichtbar, in welcher man anhand der Markierungen die Regelzusammensetzung und anhand der obigen Balken die Messfaktoren ablesen kann (vgl. Bruzzese & Davino, 2008, S. 107 f.; Sekhavat & Hoeber, 2013, S. 36 f.). In der ersten vertikalen Spalte von rechts ist somit die Regel {Rasierer} → {Rasierklingen} enthalten. Hier zeigt sich der bei der Tabellennotation angeschnittene Unterschied zwischen ei-

ner textlichen und visuellen Darstellung von Informationen bezüglich Assoziationsregeln. In einer 3D-Matrix sind die Informationen ausschließlich visualisiert und damit schnell überblickbar, was dazu geeignet ist, auf einen Blick alle interessanten Regeln zu identifizieren. Sobald dies jedoch vorgenommen wurde, und man detaillierte Informationen bezüglich der genauen Messwerte herausfinden möchte (wie bspw. dem genauen Konfidenzwert der Regel {Rasierer} → {Rasierklingen}), reicht eine Matrizennotation alleine nicht mehr aus. Denkbar wären zwar auch Interaktionsformen wie bspw. Hovereffekte innerhalb einer RE oder Anreicherung der Notation durch mehrere Messwertlinien, jedoch wäre eine separate Anzeige der Informationen in Tabellenform nach eigener Meinung aufgrund der potenziellen Überladung der Grafik oder möglichen Wartezeite- bzw. Steuerungszeiten bis zum Erscheinen von Hovers besser geeignet.

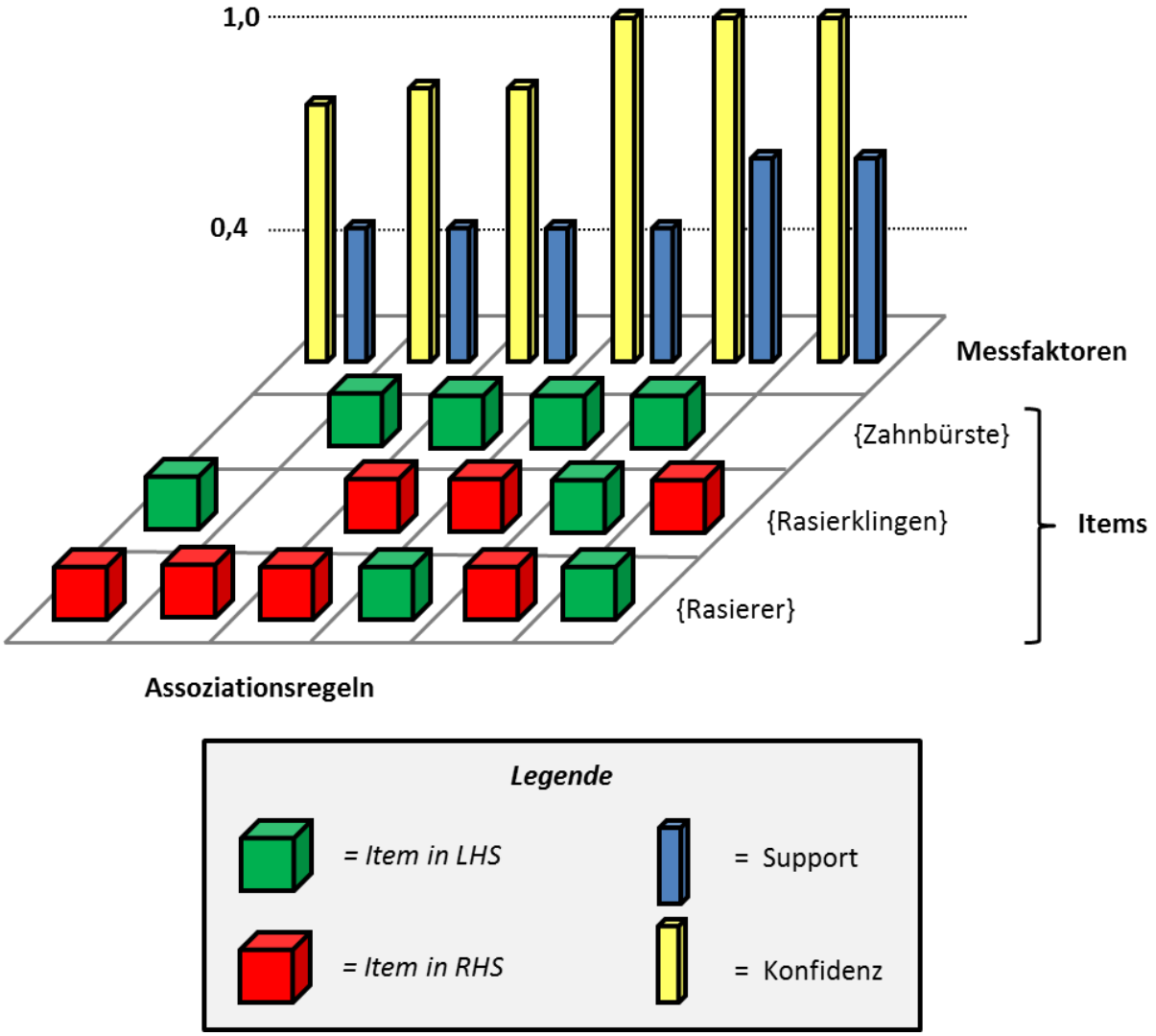


Abbildung 15: Visualisierung der Beispielregeln in einer 3D-Matrix

Graphen

In Graphen werden Assoziationsregeln in Form eines Netzes dargestellt (siehe Abbildung 16). Das Netz enthält zum einen alle Items, welche in den visualisierten Regeln beinhaltet sind. Zum anderen sind die Messwerte einer Regel durch einen Knoten in Form eines Kreises dargestellt. Der Durchmesser dieses Kreises ist die Konfidenz und die farbliche Kennzeichnung des jeweiligen Kreises ist der Support. Es liegt somit mit Bezug auf die Beispielabbildung auf der Hand, dass gleichermaßen viele Kreise wie Regeln in der Notation existieren. Die Zusammensetzung einer Regel wird über Pfeile realisiert. Pfeile von Items zu einem Knoten stellen die LHS der Regel und Pfeile von dem angesteuerten Knoten zu weiteren Items markieren die RHS (vgl. Bruzzese & Davino, 2008, S. 108 f.; Appice & Buono, 2005; Sekhavat & Hoeber, 2013, S. 37). Im rechten unteren Bereich ist somit die Regel {Zahnbürste}→{Rasierklingen} abgebildet.

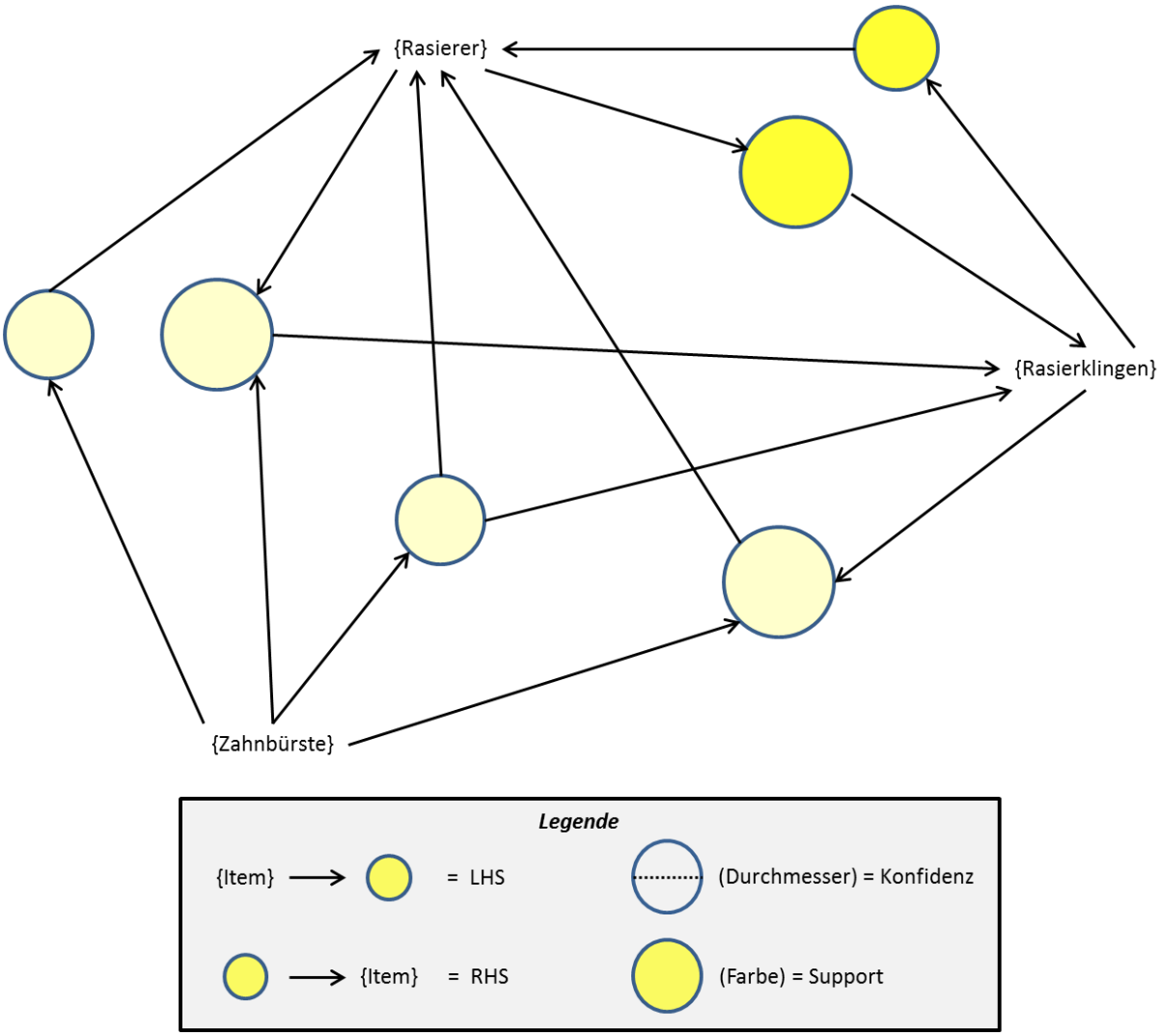


Abbildung 16: Visualisierung der Beispielregeln in einem Graphen

Der maßgebliche Vorteil dieser Notation liegt darin, dass man Gruppierungen von Knoten durch die Kreisformatierungen schnell interpretieren und deren Assoziationen gut abschätzen kann. Anhand der beiden Kreise zwischen {Rasierer} und {Rasierklingen} kann man bspw. durch die kräftigere Farbgebung erkennen, dass die Assoziationen zwischen diesen Items aufgrund eines hohen Supports innerhalb des Netzes vergleichsweise intensiv sind. Anhand des größeren Durchmessers des Knotens von {Rasierer} zu {Rasierklingen} kann außerdem auf einen Blick die höhere Konfidenz im Vergleich zu dem Knoten der umgekehrten Regel festgestellt werden. Demnach lässt sich bereits nach kurzer Zeit die Interpretation treffen, dass Rasierer und Rasierklingen besonders häufig in der Datenbank vorkommen, und hierbei vor allem eine Empfehlung von Rasierklingen zusätzlich zu dem Rasierer sinnvoll wäre.
Ebenfalls dienlich ist die ungefähre, räumliche Position von den Kreisen. Die Länge der Pfeile oder genaue Position der Kreise liegen hierbei keinen spezifischen Messwerten zugrunde, jedoch werden die Positionen der Kreise im Gesamten insofern optimiert positioniert, sodass die Anzahl an Überschneidungen minimiert wird. Insofern entstehen indirekt räumliche Cluster, welche einen ungefähren Überblick der Assoziationen durch deren Position liefern.
Auch in dieser Visualisierungsnotation können die Regeln schnell überblickt werden und im Gegensatz zu der Matrizennotation dient hierbei die räumliche Komponente zusätzliche Unterstützung bei der Interpretation der Assoziationen zwischen den Items. Jedoch verhält es sich bei dem Genauigkeitsgrad der Darstellungen der Messfaktoren ähnlich zu dem herausgestellten Sachverhalt bei 3D-Matrizen. Die Messwerte sind lediglich durch Größe und Farbe der Kreisknoten dargestellt, sodass sich auch bei dieser Visualisierungsmethode eine Kombination mit einer Tabelle anbietet.

Parallelkoordinaten

In der Parallelkoordinatennotation werden Regeln in einer baumähnlichen Struktur visualisiert (siehe Abbildung 17). Die Notation besteht hierbei aus vertikalen, parallel zueinander positionierten Linien. Diese Linien stellen jeweils eine Itemposition innerhalb einer Regel dar. Die Items sind auf der Horizontalen aufgetragen und durch Bereiche (weiß und grau) abgegrenzt. Eine Regel wird innerhalb dieser Struktur durch eine Kombination von gerichteten Pfeilen und einfachen Linien visualisiert. Jede Regel besitzt genau einen gerichteten Pfeil sowie potenziell mehreren Linien. Für den Fall, dass eine Regel aus nur zwei Items besteht, wird die Regel durch einen einzigen gerichteten Pfeil dargestellt. Hierbei sind der Ursprung des Pfeils die LHS und das Ziel die RHS. Bei Regeln mit mehr als zwei Items werden jeweils vor und / oder nach dem Pfeil einfache Linien hinzugefügt, um mehr Items in die Regel einzuschließen. Hierbei gilt, dass alle von Linien und dem gerichteten Pfeil durchlaufenen Items bis und in-

klusive der Pfeilspitze die LHS darstellen. Alle Items ab und exklusive der Pfeilspitze, welche von einfachen Linien durchlaufen werden, sind die RHS. Der obere und dickere Pfeil, welcher in der Beispielabbildung von dem Item {Rasierklingen} in Position 1 ausgeht und bei dem Item {Rasierer} in Position 2 endet, stellt somit die Regel {Rasierklingen}→{Rasierer} dar (vgl. Bruzzese & Davino, 2008, S. 112 ff.; Yang, 2003; Sekhavat & Hoeber, 2013, S. 36). Die dünnere, ebenfalls von dem Item {Rasierklingen} in Position 1 ausgehende Linie, welche ab dem Item {Zahnbürste} in Position 2 als gerichteter Pfeil hin zu dem Item {Rasierer} in Position 3 läuft, stellt die Regel {Rasierklingen, Zahnbürste} → {Rasierer} dar.

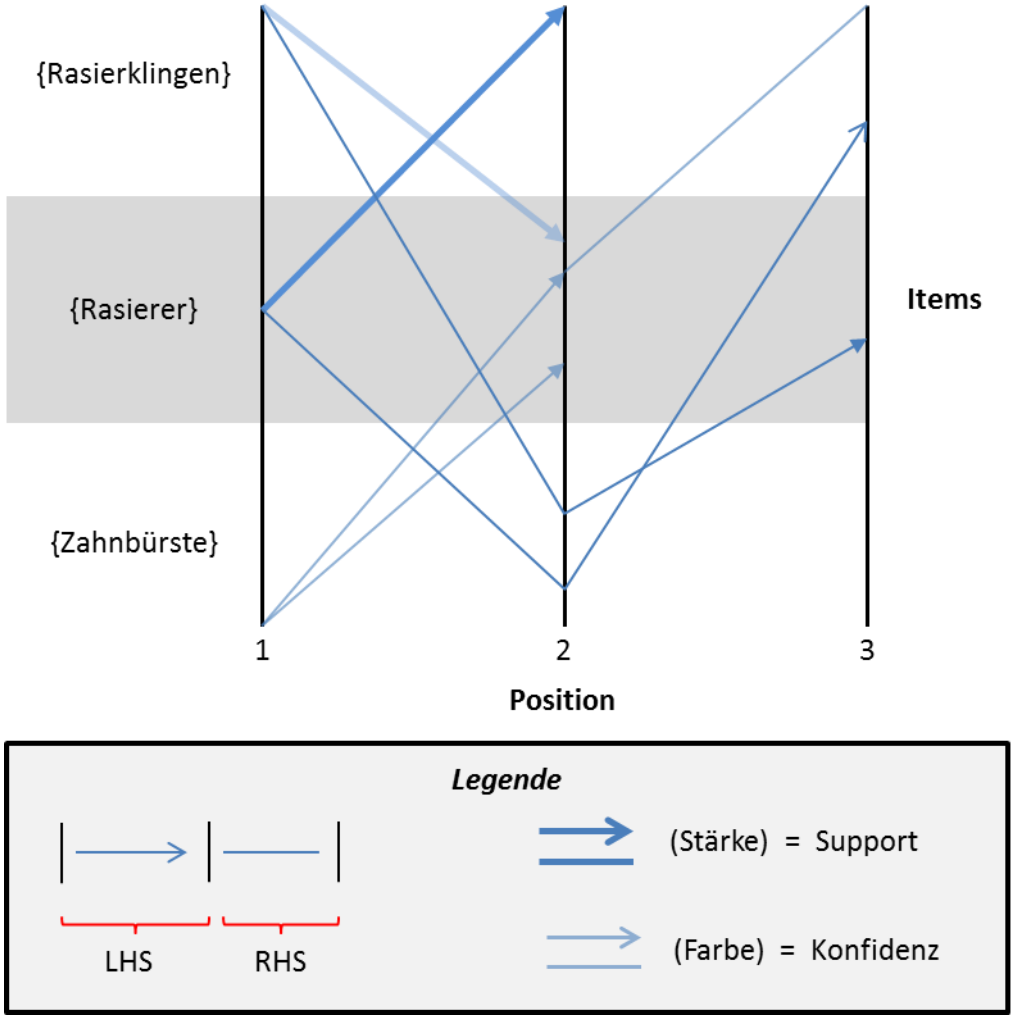


Abbildung 17: Visualisierung der Beispielregeln mittels Parallelkoordinaten

Auch in dieser Visualisierungsmethode kann man anhand von räumlich positionierten Pfeilgruppen Assoziationen zwischen zwei Itembereichen schnell identifizieren und interpretieren. Die hohe Assoziation zwischen {Rasierer} und {Rasierklingen} ist auch hier schnell durch die Stärke der jeweiligen Pfeile erkennbar und der Konfidenzunterschied ist ebenso durch die

Farbe der Pfeile identifizierbar. Auch diese Notation kommt ohne textlich genaue Informationen bezüglich der Messwerte aus.

2.5 Prototyping

In diesem Unterkapitel werden verschiedene Arten des Prototypings im Kontext der Softwareentwicklung vorgestellt, um die Entwicklung des in dieser Arbeit zu erstellenden Prototyps zu steuern (vgl. Carr & Verner, 1993, S. 3). Prototyping definiert im Kontext der Softwareentwicklung im Grundsatz eine Entwicklungsweise, in welcher bei einem komplexen und aufwändigen Ziel durch entsprechende initiale Fokussierung auf Teilbereiche schnell erste Ergebnisse erzielt werden können (vgl. Hallmann, 1990, S. 1). Diese Ergebnisse werden dann frühzeitig genutzt, um den Entwicklungsansatz in Kongruenz mit dem Ziel zu evaluieren, damit langfristige Fehlentwicklungen in einem komplexen Entwicklungsprozess vermieden werden. Es existieren fünf etablierte Arten des Protoypings, welche alle diesen Ansatz bedienen. Die Existenz mehrerer Arten begründet sich durch unterschiedliche Herangehensweisen bezüglich der Fokussierung auf einen Teilbereich des Ziels (vgl. Hallmann, 1990, S. 23 ff.). Folgend werden diese Herangehensweisen erläutert.

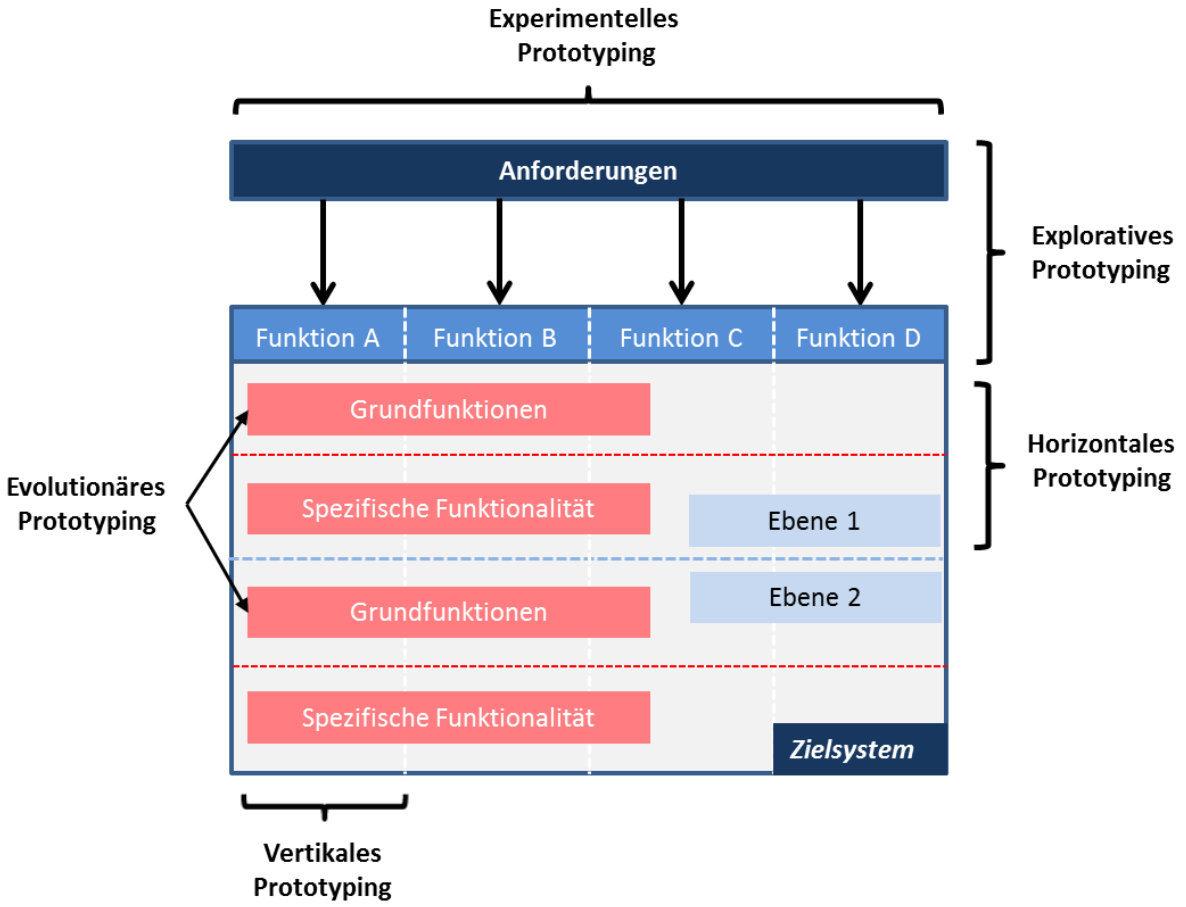


Abbildung 18: Prototyping Arten

Abbildung 18 enthält einen Überblick über alle etablierten Prototyping-Arten. Diese legen die Definition eines Zielsystems zu Grunde, welches aus drei Dimensionen besteht. Ein System wird vordergründlich in verschiedene Funktionsbereiche unterteilt. Die Funktionen wiederum sind unterteilbar in Ebenen, welche wiederum in grund- und spezifische Funktionaliät untergliedert sind.

Bei dem **horizontalen Prototyping** implementiert man die unterste Ebene aller Funktionen, und zwar sowohl Grund- als auch spezifische Funktionalität. Das Ziel bei dieser Vorgehensweise ist das Beurteilen des Zielansatzes anhand einer fertigen Ebene, welche dann als Ausgangspunkt für weitere Ebenen genommen werden kann (vgl. Manhartsberger, 1998, S. 20).

Bei dem **vertikalen Prototyping** wird ein Funktionsbereich vollständig entwickelt. Dieser kann dann mit dem Nutzer getestet werden um Anpassungen für die weiteren Funktionsbereiche zu erhalten. Diese Vorgehensweise ist vor allem bei unklaren Anforderungsspezifikationen bezüglich der weiteren Funktionen sinnvoll (vgl. Manhartsberger, 1998, S. 20).

Bei dem **evolutionären Prototyping** werden alle Grundfunktionalitäten des Systems über alle Funktionen und Ebenen erstellt. Man erhält somit einen im Grundsatz umfänglich funktionierendes Systemgerüst, welches beurteilt und im Anschluss mit den spezifischen Funktionalitäten angereichert werden kann (vgl. Carr & Verner, 1993, S. 10).

Bei dem **explorativen Prototyping** werden einzelne Spezifikationen aus den Anforderungen implementiert, um die Tauglichkeit der Anforderung aus Nutzersicht zu überprüfen. Hierbei orientiert sich die Implementierung an den Anforderungen, welche als nachprüfbar erachtet werden (vgl. Carr & Verner, 1993, S. 8; Manhartsberger, 1998, S. 20).

Letztlich verbleibt das vor allen Dingen in der Forschung eingesetzte **experimentelle Prototyping**. Bei diesem wird die Heuristik bezüglich Systementwicklung und Anforderungen umgekehrt. Denn es wird hierbei ein Prototyp entwickelt, in welchem verschiedene unklare Implementationsmöglichkeiten erschlossen werden. An diesen Versuchen können dann sehr genaue Anforderungen für ein Zielsystem abgeleitet werden. (vgl. Carr & Verner, 1993, S. 9).

2.6 R

„R“ ist eine weit verbreitete, statistische Programmiersprache aus dem Open Source Bereich. Der Ursprung geht auf das Jahr 1992 zurück, in welchem die Statistiker Ross Ihaka und Robert Gentleman (die Bezeichnung der Sprache „R“ ist auf die Vornamen der Entwickler zurückzuführen) an der Universität Auckland R als eine Erweiterung der Programmiersprache S, welche wiederum auf C basiert, entwickelten (vgl. Ihaka & Gentleman, 1996). S ist gleichermaßen eine statistische Programmiersprache und zu großen Teilen mit R kompatibel. Der Erfolg in Form einer weitreichenderen Verwendung von R gegenüber S basiert auf der freien

Verfügbarkeit in Form von der Veröffentlichung unter der GNU Lizenz im Jahre 1995 (vgl. Ihaka, 1998). Denn während S durch kommerzielle Verbreitung die Zielgruppe vordergründlich auf die Wirtschaft reduzierte, wurde die Weiterentwicklung und Nutzung von R in der weltweiten Open Source Community vorangetrieben. Dies hat zu einer Bildung zahlreicher Communities geführt, in welchen Programmierer an der Weiterentwicklung von R und R-Packages arbeiten und somit eine hohe Dynamik erzeugen. Konkret kann jeder eigene R Funktionen schreiben und die entsprechende Funktionsbibliothek in ein R Package überführen. Dieses beinhaltet neben der Funktionsbibliothek vor allem Dokumentationen und muss in einer Standardstruktur erstellt werden. Das Package wird dann in das zentrale „Comprehensive R Archive Network“ (CRAN) eingespeist, über welches andere Nutzer das Package über einen R Client der Wahl beziehen und einbinden können. Diese dynamische Struktur einer kollaborativen Open Source Software hat zu einer weitreichenden Nutzung geführt. Denn systemübergreifend sind aktuell 8376 Packages verfügbar (Stand: 20.05.16, Abfrage des Cloud Mirror), welche diverse Analysebereiche abdecken. Diese können bspw. Packages sein, welche Funktionen einzelner Datenoperationen beinhalten und als eine Art Toolbox fungieren, von welcher man einzelne Funktionen an verschiedenen Stellen nutzen kann. Es können aber auch weitaus mächtigere Packages enthalten sein, z.B. „arules“, in welchem eine vollständige Assoziationsanalyse nach aktuellen Erkenntnissen durchgeführt werden kann (vgl. Hahsler, Buchta, Gruen, & Hornik, 2015) oder „shiny“, welches ein Framework zur Programmierung von Webapplikationen ist.

3. Ausgangslage

In diesem Kapitel wird die grundsätzliche Ausgangslage beschrieben, anhand derer sich die Entwicklung des Prototyps ausrichtet. Dazu wird zuerst die Zielgruppe sowie Zielsetzungen des Data Minings bestimmt, was der ersten Crisp-DM Phase Business Understanding entspricht. Im Anschluss wird detailliert aufgezeigt, wie das Zielsystem des Prototyps aufgebaut wurde. Dabei werden verstärkt die verschiedenen Thematiken aus den Theoretischen Grundlagen in Zusammenhang gesetzt, um die konkludierte Ausgangslage vor der Entwicklung aufzuzeigen.

3.1 Zielgruppe

Der Prototyp zielt aufgrund des generischen Anspruchs nicht auf eine spezifische Zielgruppe ab, sondern bedient potenziell zahlreiche Anwendungsfälle. In diesem Kapitel werden die allgemeine Zielgruppe sowie die Anforderungen an die Daten abgesteckt.

3.1.1 Eigenschaften & Zielsetzung durch Data Mining (Business Understanding)

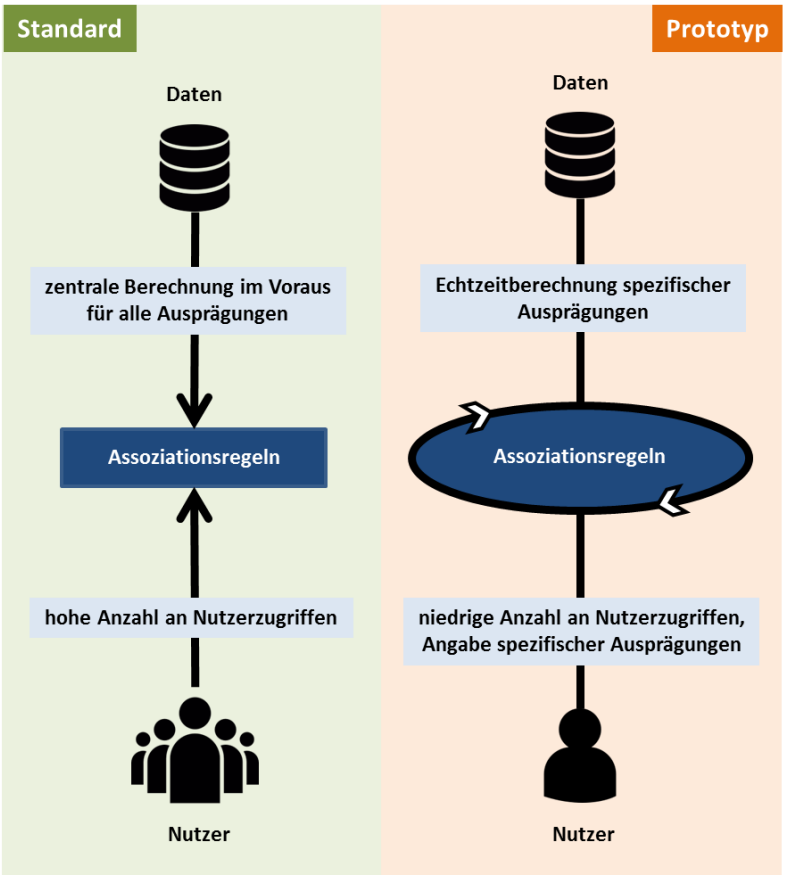


Abbildung 19: Zielgruppenspezifische Abgrenzung des Prototyps

Bei der Zielgruppe handelt es sich um jegliche Unternehmen, welche eine große Anzahl an Produkten / Services aufweisen. Je größer die Anzahl an zu verwaltenden Ressourcen ist, des-

to höher liegt das Potenzial, durch den Einsatz des Prototyps einen Mehrwert zu generieren. Dabei müssen die zu koordinierenden Elemente nicht zwingend Produkte / Services sein, welche dem Kunden verkauft werden, sondern können auch interne Ressourcen sein. Die Eigenschaften der zu bedienenden Zielgruppe lassen sich anhand der Abgrenzungen des Prototyps vom Status Quo aufzeigen (siehe Abbildung 19).

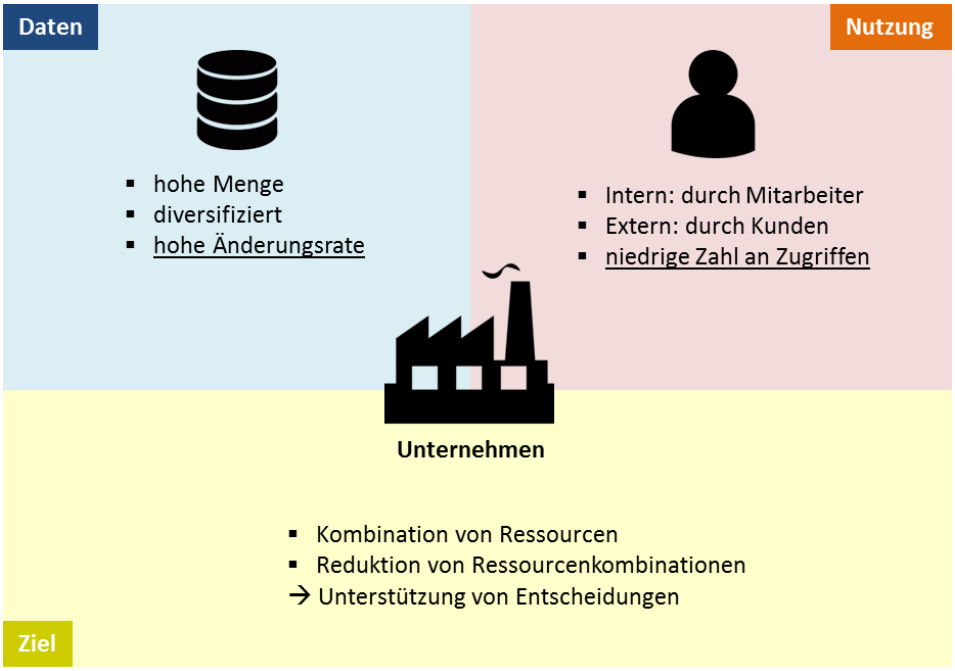


Abbildung 20: Zielgruppe des Prototyps

Bei der Assoziationsanalyse wird standardmäßig eine zeitpunktbezogene Berechnung der Assoziationsregeln unter Verwendung von performanten Algorithmen vorgenommen. Die zeitliche Differenz zwischen Regelberechnung und Verwendung durch den Nutzer wird hierbei in Kauf genommen, weil eine jeweilige Echtzeitberechnung bei jedem Nutzer aufgrund hoher Nutzerzugriffe zu aufwändig wäre. Der Prototyp bedient die umgekehrte Situation. Dabei wird zu Grunde gelegt, dass die Anzahl der Nutzerzugriffe niedrig (individuell nach Hardware zu bestimmen) ausfällt. Dafür können jedoch die Assoziationsregeln in Echtzeit berechnet werden und darüber hinaus auch solche mit mehreren Ausprägungen in der RHS, was in der standardmäßigen Praxis ebenfalls aufgrund des hohen Mehraufwandes nicht vorgenommen wird. Insofern ist der Prototyp zum einen in solchen Unternehmen einzusetzen, in welchen die Interaktionsanzahl mit dem Kundenstamm niedrig ist. Als Beispiel dienen hierbei Händler von teuren Produkten, bei welchen die Anzahl der Kaufvorgänge niedriger ist als bei Händlern mit Massenprodukten. Zum anderen ist ein Einsatz des Prototyps zur internen Verwendung möglich. Als Beispiel hierfür können Ersatzteillieferanten angeführt werden, welche intern Warenkörbe an Ersatzteilen mittels des Prototyps zusammenstellen, um Ineffzienzen

durch fehlende/unnötige Artikel zu minimieren. Durch die Echtzeitberechnung sind vor allem Unternehmen anzusprechen, welche eine hohe Datenänderungsrate aufweisen und die zeitliche Differenz zwischen Regelberechnung und Nutzung im Standardverfahren zu hoch ausfallen würde. Hier wären als Beispiel Unternehmungen aus dem Finanzsektor anzusprehen. Des Weiteren eignet sich der Prototyp vor allem auch für Unternehmen, welche besonders diversifizierte Daten vorliegen haben. Denn die generische Programmierung und automatisierte Datenaufbereitung erlaubt eine schnelle Exploration von zahlreichen Variablenkonstellationen. Damit können bspw. Unternehmungen mit einer Vielzahl an unterschiedlichen Systemen bedient werden, was eine diversifizierte Datengrundlage nach sich zieht.
Der Einsatz des Prototyps ermöglicht schließlich das Ziel, live durch die Assoziationen von Daten zu iterieren und durch diese Entscheidungen im Unternehmen zu unterstützen. Es können dabei zwei Motive verfolgt werden. Zum einen kann der Prototyp verwendet werden, um zusätzliche Ausprägungen zu bereits Ausgewählten aufzufinden. Zum anderen kann die Nutzung auch darauf abzielen, besonders große Sets an Ausprägungen durch die Assoziationen zu reduzieren. Ersteres ist in der Praxis vor allem in Handel eingesetzt, bspw. in Form von Produktanzeigen auf bereits ausgewählten Produkten. Das zweite Motiv ist hingegen vergleichsweise wenig eingesetzt, beinhaltet jedoch ebenfalls hohes Potenzial. Denn vor allem bei Vorgängen, in welchen Ressourcen unter Kosten des Unternehmens kombiniert werden, ist eine Reduktion von unnötigen Elementen sinnvoll.

3.1.2 Datengrundlage (Data Understanding)

Im vorangegangenen Kapitel wurden bereits die Eigenschaften der grundsätzlichen Datenlage zur ökonomischen Verwendung im Prototyp aufgezeigt. In diesem Kapitel geht es um die spezifischen Anforderungen an die Datenstruktur. Um Assoziationen in Daten aufdecken zu können, benötigt man Prozesse sowie Ausprägungen. Prozesse stellen hierbei inhaltlich abgeschlossene Kombinationen von Ausprägungen dar, welche als Grundlage zur Bildung der Assoziationen dienen. Im Fall eines Einsatzes im Handel würden sich bspw. Bestellungen als Prozesse anbieten, welche Ausprägungen in Form von Produkten enthalten. Denkbar wären auch Konstellationen in denen Zeiträume als Prozesse verwendet werden, um das Auftreten von Ereignissen im Sinne von Ausprägungen in Verbindung zu setzen. Grundsätzlich können zahlreiche Kombinationen aus Prozessen und Ausprägungen verwendet werden, unter der Voraussetzung, dass die Prozesse eine Gruppierung der Ausprägungen enthalten. Es existieren hierbei zwei Formen, in welchen ein Datensatz die benötigten Informationen enthalten kann: vertikal und horizontal (siehe Abbildung 21).

Die vertikale Variante dürfte nach eigener Auffassung bei der späteren Verwendung am gängigsten sein. Hierbei liegen Prozesse und Ausprägungen in jeweils einer Variable vor. Die Prozessvariable enthält eine Information (ID, Datum, etc.), welche die Zugehörigkeit einer Ausprägung zu einem Prozess aufzeigt. Demzufolge wird ein Prozess in mehreren Zeilen durch die gleiche Information in der Prozessvariable und unterschiedlichen Informationen in der Ausprägungsvariable zusammengesetzt. Bei der horizontalen Variante sind die Ausprägungen zeilenweise aufgetragen und jede Zeile stellt einen Prozess dar. Dabei ist der Zeilenindex die Prozessvariable.

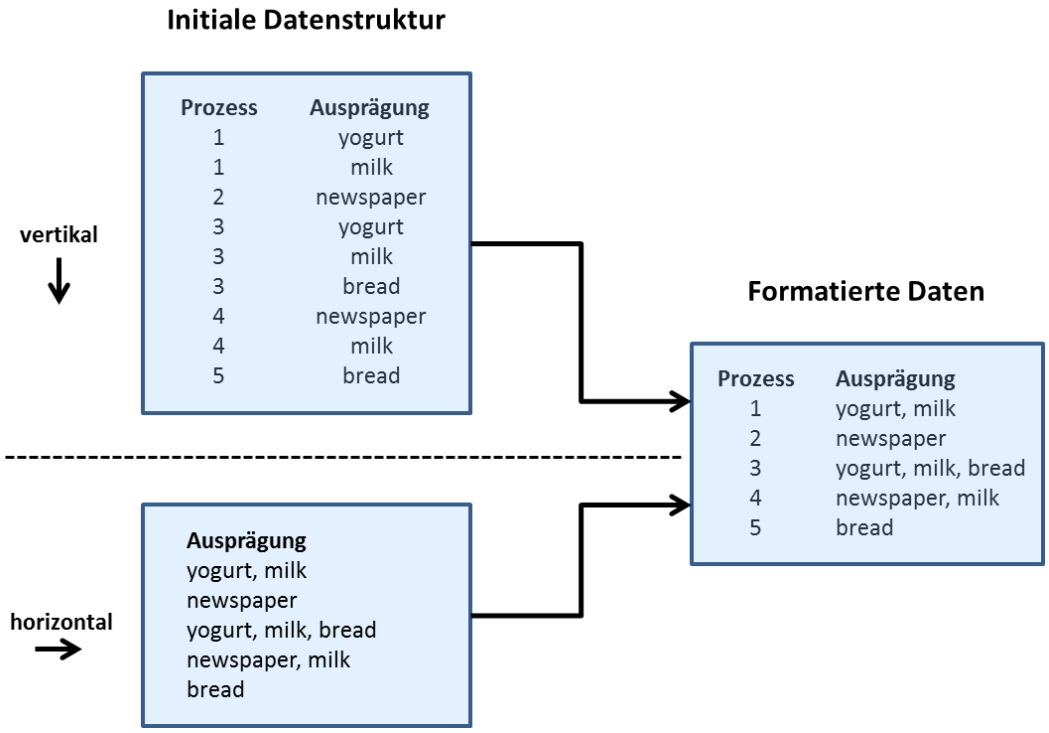


Abbildung 21: Mögliche Strukturen der Datengrundlage

Die aufgezeigten Anforderungen könnten den Eindruck einer sehr restriktiven Beschränkung der Zielgruppe entstehen lassen. Jedoch können unpassende Daten potenziell umstrukturiert werden, um diese für den Prototyp verwendbar zu machen. Dabei geht es um die Konstruktion von Prozessvariablen durch Gruppierungen. Insofern kann eine geeignete Variable (z.B. Datum) individuell gruppiert werden, um Aufschlüsse über die Ausprägungen (z.B. Kundeninteraktion etc.) zu erhalten (siehe Abbildung 22). Dann könnten Assoziationen zwischen Kundeninteraktionen in den gruppierten Zeiträumen aufgedeckt werden. Des Weiteren kann auch eine Assoziationsanalyse über zahlreichen Variablen stattfinden, indem ggfs. Variablen in geeignete Intervalle gruppiert und eine horizontale Datenstruktur unter Verwendung des Zeilenindex als Prozessvariable angewendet wird (siehe Abbildung 23). Dann können die jeweiligen Variablen kombiniert werden, bspw. durch Auswahl eines Kunden sowie einem

hohem Umsatz, um den wahrscheinlichen Wochentag solcher Transaktionen zu erhalten. Derartige Datenkonstruktionen sind nicht Teil des Prototyps und müssen vom Analysten vorab vollzogen werden, der Prototyp unterstützt diesen Vorgang jedoch durch die generische und schnelle Integrationsmöglichkeit verschiedener Kombinationen aus Prozess- und Ausprägungsvariablen.

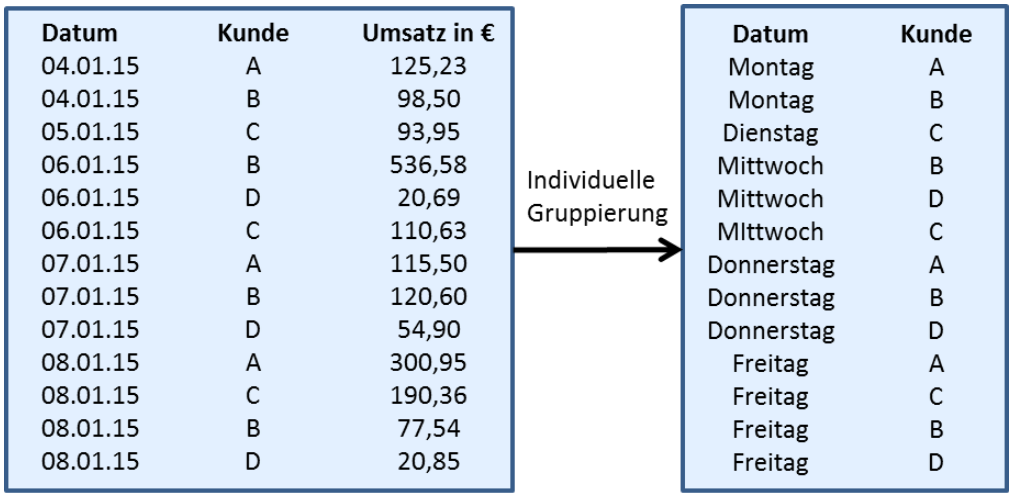


Abbildung 22: Beispiel einer zeitlichen Gruppierung für eine vertikale Dateneinbindung

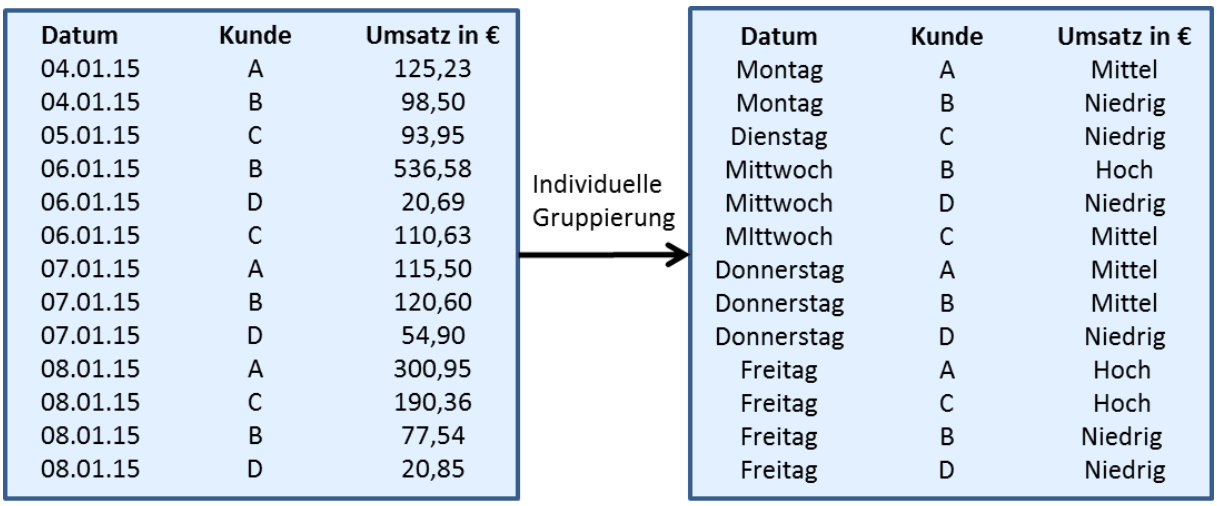


Abbildung 23: Beispiel der horizontalen Einbindung eines gesamten Datensatzes durch Gruppierung von Variablen

3.2 Rahmenbedingungen des Prototyps

An dieser Stelle werden die konkreten Rahmenbedingungen der Entwicklung aufgezeigt. Dazu wird zuerst der grundsätzliche Programmieransatz hinsichtlich der Automatisierung von Data Mining Prozessen aufgezeigt. Darauf folgend wird die Auswahl der Anwendungsumgebung begründet und anschließend ein Überblick über die initialen Systembereiche hinsichtlich der übergeordneten Zielsetzung gegeben. Abschließend wird dann ein Implementierungsplan angegeben, welcher die vergangene Entwicklung auf einem Zeitstrahl aufzeigt.

3.2.1 Programmieransatz

Neben der Anwendung des Prototyps innerhalb der zuvor vorgestellten Zielgruppe stellt die Gesamtentwicklung den Versuch dar, durch Automatisierungen von Data Mining Prozessen entsprechende Projekte effizienter zu machen. Dies lässt sich anhand des in Kapitel 2.1 vorgestellten Vorgehensmodells Crisp-DM aufzeigen.

Das Crisp-DM Modell stellt die allgemeine Vorgehensweise von Data Mining Projekten dar und hat sich in der Praxis etabliert. Folgerichtig wurden Softwarelösungen zur Durchführung der Analysemethoden auf die allgemeine Anwendbarkeit ausgerichtet und beinhalten eine Fülle an Analysemethoden. Dies begründet sich auch auf den zahlreichen Anwendungskontexten und Datengrundlagen, welche eine breite Masse an Analysemethoden bedingen. Vor diesem Hintergrund sind Projektdurchführungen jedoch mit hohem Aufwand verbunden. Wie bereits in der Einleitung dieser Arbeit dargelegt, sind hohe Kosten für die jeweiligen Experten und Softwarelösungen zu erwarten. Weiterhin sind solche Projekte aufgrund von ungenauen Zielbestimmungen schwierig zu planen.

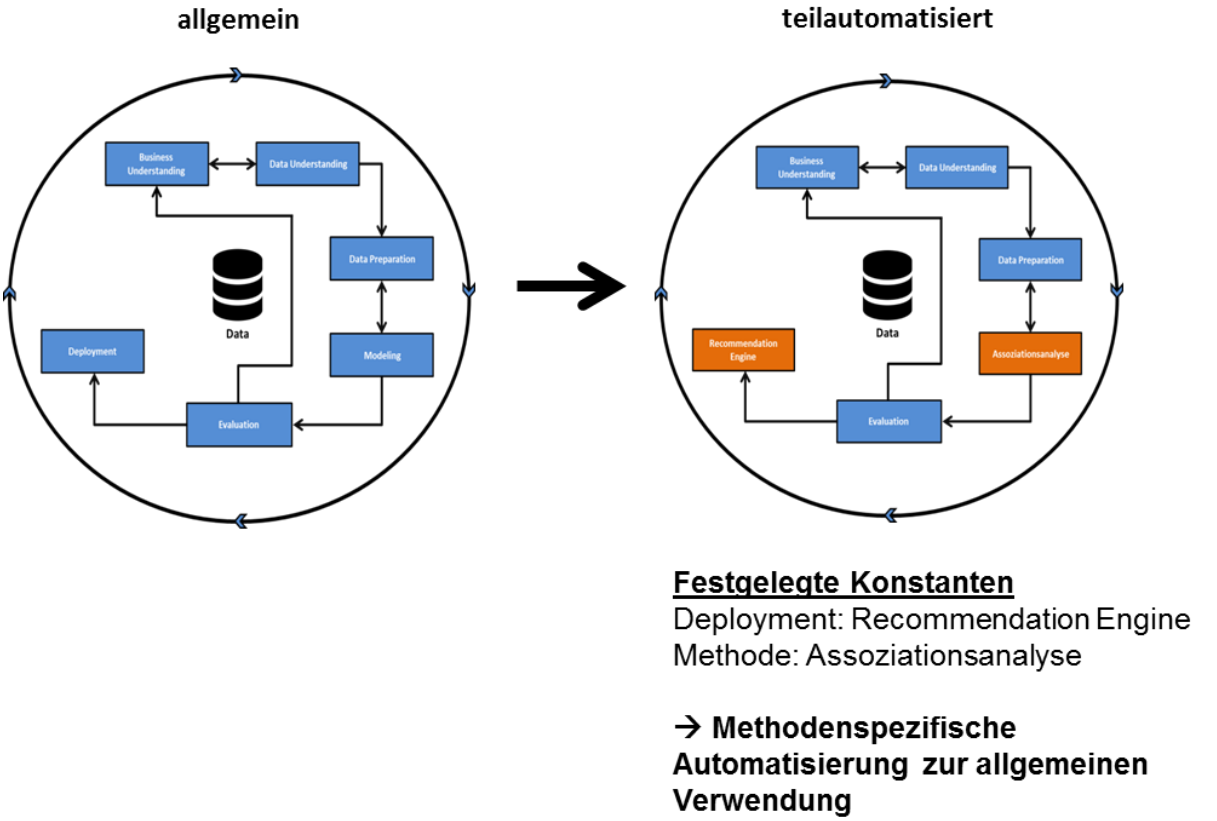


Abbildung 24: Ansatz zur Effizienzsteigerung von Data Mining Projekten mittels Automatisierungen

Es ist davon auszugehen, dass bei Projekten mit gleichen Zielsetzungen und/oder gleichen Datengrundlagen Best Practices in der Evaluationsphase vergangener Projekte erarbeitet wurden und somit effizientere Durchführungen ermöglicht werden. An dieser Stelle ist zu hinter-

fragen, ob derartige Erkenntnisse innerhalb von Softwarelösungen durch Automatisierungen realisiert werden können. Um diese grundsätzliche Möglichkeit zu testen, wird mit dem Prototyp eine exemplarische Automatisierung für Projektdurchführungen mit dem Ziel einer RE unter Anwendung der Assoziationsanalyse versucht (siehe Abbildung 24). Dazu werden die spezifischen Methoden hinsichtlich auf Assoziationsanalysen basierenden REs automatisiert und dabei gleichzeitig eine generische Verwendbarkeit gewährleistet. Es soll schlussendlich möglich sein, diverse Daten in den Prototyp einfügen zu können und mittels grundlegenden Parametereinstellungen ad hoc eine RE zum Laufen zu bringen.
Sofern dies erfolgreich realisiert werden kann, besteht ein Potenzial zur analogen Anwendung für weitere Data Mining Ziele und der Möglichkeit, Erfahrungen im Sinne von Best Practices innerhalb dieser Lösungen etablieren zu können. Infolgedessen können durch solche Lösungen effizientere Projektdurchführungen ermöglicht und eine breitere Masse an Kunden erschlossen werden.

3.2.2 Auswahl der Anwendungsumgebung

R eignet sich zum einen sehr gut für Programmentwicklungen im Bereich der Forschung. Denn aufgrund der freien Verfügbarkeit und bereits aufgezeigten, immensen Nutzeranzahl, welche entweder an Erweiterungen des Core Programms oder eigenständigen Packages arbeiten, weist R eine hohe Dynamik auf. Wissenschaftliche Erkenntnisse im Bereich der Data Mining Methoden werden vor diesem Hintergrund besonders schnell implementiert und können durch die freie Verfügbarkeit kollaborativ erarbeitet bzw. modifziert werden. Im Kontext dieser Arbeit spiegelt sich dies vor allem in bereits verfügbaren Packages wider, welche im Protoyp zum Einsatz kommen und teilweise modifiziert werden (siehe Tabelle 8). Die grün unterlegten Packages werden hierbei besonders umfassend genutzt, allen voran „arules“ und „arulesViz“ bei der Analyse sowie „shiny“ bezüglich der Interfaces. Die weiteren Packages wurden zum Teil wegen einzelnen Methoden hinsichtlich Datenoperationen eingebunden. Etablierte Softwarelösungen kommerzieller Hersteller und andere Open Source Lösungen enthalten hingegen wenige Funktionalitäten hinsichtlich der Assoziationsanalyse bzw. nur spezifische Lösungen, welche nicht wiederverwendet werden können. Insofern wurde R zu einem großen Teil deshalb gewählt, weil hier die überwiegende Grundlagenprogrammierung hinsichtlich der Assoziationsanalyse in einer besonders wiederverwendbaren und modifizierbaren Art enthalten ist.

Tabelle 8: Übersicht in dieser Arbeit verwendeter R Packages

Package	Beschreibung
arules	Beinhaltet eine ganzheitliche Funktionalität hinsichtlich der Assoziationsregelanalyse mit maximal einem Item in der RHS. Zur Berechnung der Itemsets sind der Apriori sowie Eclat Algorithmen implementiert (vgl. Hahsler, Buchta, Gruen, & Hornik, 2015).
arulesViz	Enthält mehrere Visualisierungsfunktionalitäten für mit dem arules Package berechneten Assoziationsregeln (vgl. Hahsler & Chelluboina, 2015).
caTools	Toolbox mit Funktionalitäten bezüglich Visualisierungen (vgl. Tuszynski, 2014).
gtools	Enthält Funktionen bezüglich der Bearbeitung und des Verwaltens von R Packages sowie zu zahlreichen Datentransformationen (vgl. Warnes, Bolker, & Lumley, 2015).
plyr	Toolbox mit Funktionen zum Separieren und Integrieren von Daten (vgl. Wickham, 2011).
shiny	Framework zur Erstellung von Webapplikationen. Integrierte Anwendungsumgebung mit R, HTML/CSS sowie Javascript (vgl. Chang, Cheng, Allaire, Xie, & McPherson, 2016).

Letztlich begründet sich die Auswahl abermals auf der freien Verfügbarkeit, jedoch hier hinsichtlich des Prototypeinsatzes. Denn R weist aufgrund der weitreichenden Nutzung durch verschiedene Communitis/Nutzern mit verschiedenen Systemen und Anwendungskontexten äußerst generische Strukturen hinsichtlich der Verwendung auf. R läuft insofern auf allen gängigen Betriebssystemen, bietet zahlreiche Schnittstellen und lässt sich in andere Programmierumgebungen integrieren. Da das Ziel ein generischer Prototyp zur potenziellen Anwendung in zahlreichen Kontexten ist, eignet sich R durch dessen Flexibilität somit besonders als Anwendungsumgebung.

3.2.3 Grundlegende Systemstruktur

Der Prototyp besteht aus zwei Systemen, der RE als Hauptsystem sowie einem Reportmodul zur Messung und Überwachung der im Betrieb befindlichen RE (siehe Abbildung 25). Während die RE auf den Echtzeitdaten des Nutzers arbeitet, werden im Reportmodul statistische Daten der RE zeitpunktbezogen durch ein Mess-Skript gemessen und aufbereitet.

Die RE kann grob in zwei Systembereiche unterteilt werden (siehe Abbildung 26). Es wurde zuerst eine Funktionsbibliothek erstellt, in welcher der gesamte Analyseprozess von der Datenakquirierung über Modellierung bis hin zu dem Aufbereiten und Bereitstellen der finalen Analyseergebnisse auf Basis einer funktionalen Programmierung implementiert wurde. Diese Funktionsbibliothek kann aufgrund der Flexibilität von R grundsätzlich generisch eingesetzt

werden. Denkbar wäre hierbei eine Integration in andere Analysesoftware, Einbindung der Funktionen als Hintergrundprozesse innerhalb von Geschäftsanwendungen oder Erstellen einer individuellen RE.

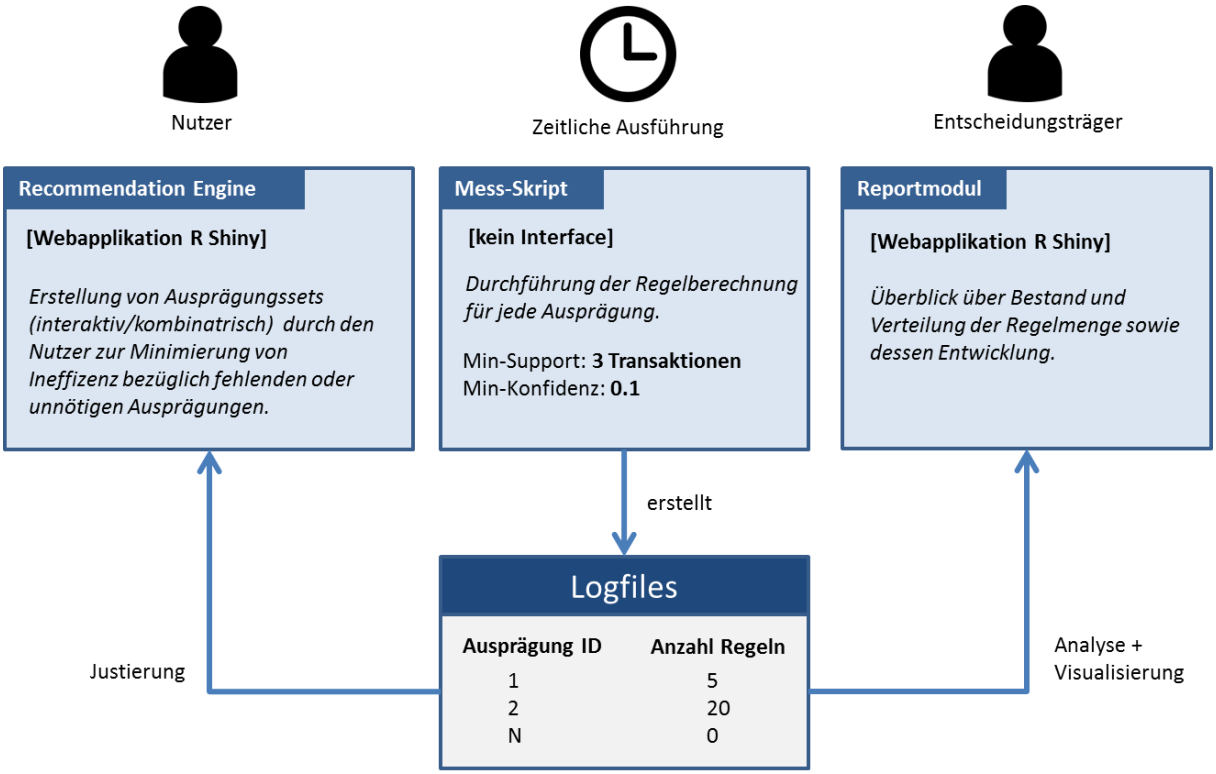


Abbildung 25: Systemüberblick

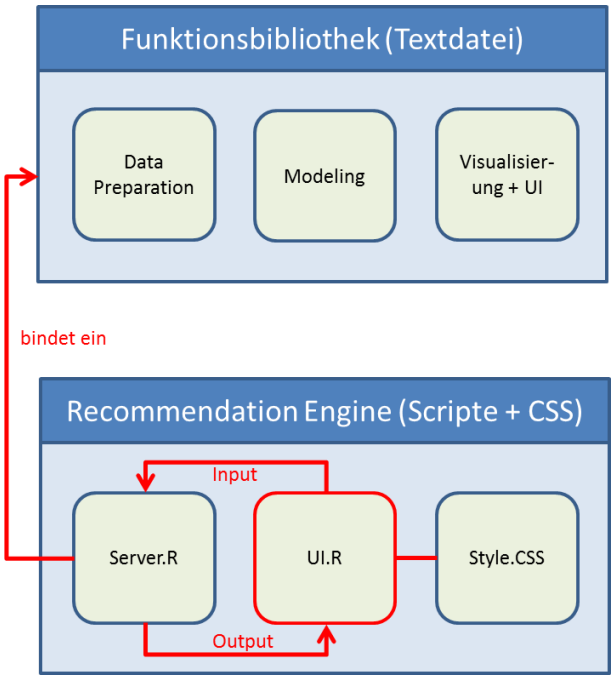


Abbildung 26: Struktur der RE

Letzteres stellt die in dieser Arbeit verwendete Lösung und zugleich den zweiten Systembereich der RE dar. Mittels des bereits vorgestellten Packages „shiny“ wurde eine Webapplikation erstellt, welche als Userinterface fungiert. Mit dem Interface steuert der Nutzer insofern die in der Funktionsbibliothek enthaltenen Analysefunktionen, indem er Daten importiert und Eingaben bezüglich der zu untersuchenden Ausprägungen und Parametern macht. Das Interface besteht aus zwei Scripten sowie einer Cascading Stylesheet (CSS) Datei. Das UI Script bestimmt in Kombination mit der CSS Datei das Design des Interfaces und das Serverscript steuert die Funktionalität unter Einbindung der Funktionsbibliothek.

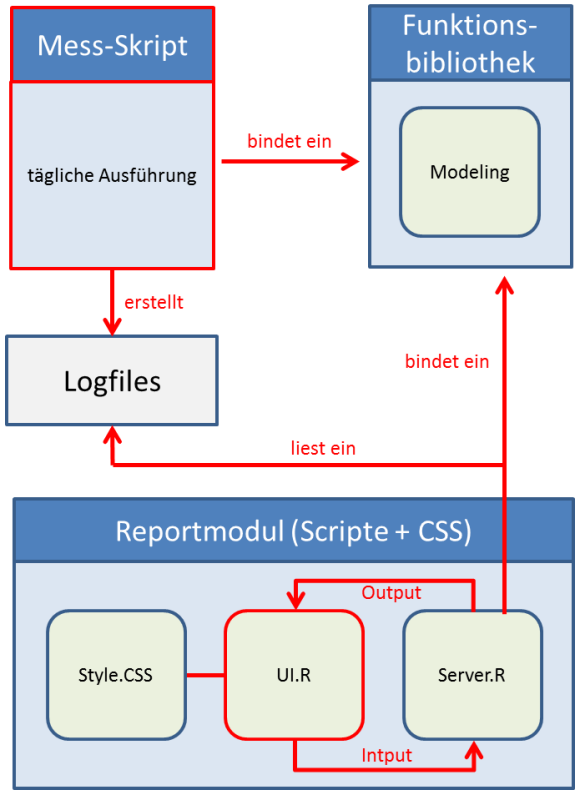


Abbildung 27: Struktur des Reportmoduls

Das Reportmodul wurde analog zu der RE implementiert. Jedoch arbeitet das Reportmodul auf von einem Mess-Skript berechneten, statistischen Daten, welche mittels der Analysefunktionen der RE erstellt werden. Diese statistischen Daten werden in Logfiles gespeichert und abermals mittels eines Webinterfaces durch das Package „shiny“ für den Entscheidungsträger aufbereitet. Die statistischen Daten geben dann Aufschluss über den Regelbestand sowie die Entwicklung des Systems im laufenden Betrieb.

3.2.4 Anforderungen & Implementierungsplan

Die RE muss zwei zentrale Anforderungen erfüllen (siehe Tabelle 9). Zum einen musste generisch programmiert werden, um eine Wiederverwendbarkeit zu ermöglichen und zum ande-

ren muss die Analyse in Echtzeit laufen. Dies grenzt den Prototyp insofern von der grundsätzlichen Assoziationsanalyse in der Praxis ab, denn hierbei werden zumeist alle Regeln einer Datenbank zentral zu einem Zeitpunkt berechnet und die somit im Voraus berechneten Regeln im laufenden Betrieb genutzt. Diese Verwendung lässt sich auf die Anzahl an Zugriffen zurückführen. Denn je höher die Anzahl der Nutzung der Regeln, desto performanter ist eine zentrale Berechnung gegenüber der Einzelanalyse. Der Prototyp zielt jedoch auf Nutzer ab, bei welchen die Aktualität der Regeln im Vordergrund steht und darüber hinaus die Anzahl an Zugriffen niedrig ist.

Tabelle 9: Funktionale Anforderungen an die RE

Prio	Anforderung	Beschreibung
1	Das System muss generisch einsetzbar sein.	Das System muss generisch programmiert werden, um eine allgemeine Nutzung in verschiedenen Kontexten ermöglichen zu können.
1	Die Analyse muss in Echtzeit durchführbar sein.	Entgegen dem verbeiteten Ansatz der Berechnung aller Regeln aus einer Datenbank, soll das System spezifische Regeln durch Auswahl des Nutzers berechnen, jedoch in Echtzeit und nicht durch Vorberechnung der Regeln
2	Es muss eine Schnittstelle für Datenimporte vorhanden sein.	Damit der Prototyp dynamisch einsetzbar ist, muss ein Datenimport möglich sein.
2	Das System muss die Datenaufbereitung automatisch durchführen.	Um eine generische Nutzung zu ermöglichen, muss das System die spezifische Datenaufbereitung automatisch durchführen und notwendige Input-Informationen auf für den Nutzer verständliche Art und Weise herunterbrechen.
3	Die Analyseergebnisse müssen mittels Visualisierung und geeigneter Formatierung für den Nutzer intuitiv verständlich gemacht werden.	Es muss eine geeignete Kombination aus Visualisierung und textlicher Formatierung implementiert werden, sodass Nutzer das System ohne notwendige Schulungsmaßnahmen anwenden können.
4	Es müssen alle Regeltypen berechnet werden können.	Neben einer iterativen Verknüpfung von Berechungen von many-to-one Regeln müssen ebenfalls many-to-many Regeln analysierbar sein.

Weiterhin ist zur generischen Nutzung des Protoypen ein Datenimport inklusive automatischer Datenaufbereitung notwendig. Der Prototyp muss neben allgemeinen Preparationen auch die für die Assoziationsanalyse Notwendigen weitgehend automatisch durchführen. Weiterhin ist der Einsatz von Visualisierungen wichtig, um das abstrakte Ergebnis der Assoziationsanalyse und vor allem die Messfaktoren dem Nutzer verständlich zu machen. Auch dies ist eine indirekte Folge der generischen Einsetzbarkeit. Schließlich muss das System alle Regeltypen bedienen. Der Fokus liegt zwar auf der iterativen Verknüpfung von Single-RHS-Analysen, jedoch ist auch eine Berechnung von vollständigen Sets an Ausprägungen für den

Nutzer sinnvoll. Auch dies grenzt den Prototyp von der allgemeinen Anwendung der Assoziationsanalyse in der Praxis ab. Denn der notwendige Aufwand gegenüber dem Nutzen von Multiple-RHS-Analysen verhindert den Einsatz in der Praxis oftmals. Dies trifft jedoch nur auf den zu Beginn aufgezeigten Fall einer zentralen Berechnung aller Regeln zu. Bei einer Echtzeitanalyse mit vergleichsweise wenigen Zugriffen ist eine optionale Berechnung von Regeln mit mehreren Ausprägungen in der RHS durchaus performant zu bewerkstelligen.
Aufgrund des potenziellen Einsatzgebietes des Prototyps, ist die performante Anwendung des Interfaces wichtig (siehe Tabelle 10). Daher sollen die Berechnungsschritte maximal 2 Sekunden in Anspruch nehmen. Dieser Wert ist eine Standardgröße im Web Development und fußt auf Studien bezüglich des biologischen Kurzzeitgedächtnis bei der Nutzung von Computern (vgl. Nah, 2004, S. 17; Shneiderman, 1984, S. 283). Da dies von der Datenlage sowie der verwendeten Hardware abhängt, wird die zeitliche Vorgabe an eine maximale Transaktionsgröße von 1.000.000 sowie die Hardwaredaten des zur Entwicklung verwendeten Gerätes (i5-3200, 16Gb Ram) geknüpft. Des Weiteren sollen dem Nutzer geeignete Systemmeldungen gegeben werden, um vor allem Fehleingaben abzufangen. Sollte der Nutzer trotz dessen kritische Fehleingaben machen, so sollte der Prototyp frühzeitige Konsistenzchecks und gegebenenfalls Abbrüche durchführen, um Abstürze des gesamten Systems zu vermeiden.

Tabelle 10: Nicht-Funktionale Anforderungen an die RE

Prio	Anforderung	Beschreibung
1	Die Reaktionszeiten der Analyseschritte sollten unter 2 Sekunden liegen.	Vor allem bei der Nutzung der iterativen Berechnung sollte das System eine besonders schnelle Reaktionszeit aufweisen, um ein explorativen Durchsuchen der Transaktionsmenge zu gewährleisten.
2	Das System sollte an zentralen Stellen durch Rückmeldungen den Nutzer bei Fehlern unterstützen.	Vor allem im Bereich des Datenimportes sollte der Nutzer bei Eingaben der Parameter durch textliche Hilfestellungen vor Fehleinstellungen gewarnt werden.
3	Systemabstütze durch Fehleingaben sollten durch frühzeitige Maßnahmen verhindert werden.	Bei Fehlkonfigurationen in Kombination mit hohem Aufwand aufgrund großer Datenmengen sollte das System frühzeitig automatische Abbrüche herbeiführen, um Systemabstütze durch hohe Fehlertoleranz zu vermeiden.

Die hauptsächliche Funktionalität des Reportmoduls ist die Darstellung von geeigneten Messfaktoren, welche die Kalibrierung der RE im konkreten Einsatz verkörpern (siehe Tabelle 11). Es handelt sich dabei um einfache statistische Kennzahlen zur Verteilung der Regelmenge innerhalb der Datenbank. Eine weitere Anforderung stellt die Selektierbarkeit der Daten dar. Es wird als wichtig erachtet, dass Nutzer des Systems in Form eines Entscheidungsträgers mit Überblick über die gesamten Geschäftsprozesse, den Status der RE in verschiedenen Sparten,

Zeiträumen und etwaigen individuellen Variablen begutachten kann. Schließlich muss der Zugriff auch mobil möglich sein, um einen spontanen und schnellen Zugriff auf die Kennzahlen des Systems an jedem Ort möglich zu machen.

Tabelle 11: Funktionale Anforderungen an das Reportmodul

Prio	Anforderung	Beschreibung
1	Das System muss geeignete Messfaktoren zur Darstellung des Systemstatus enthalten.	Damit der Entscheidungsträger den Mehrwert des Systems erkennen kann, müssen geeignete Messfaktoren konstruiert werden.
2	Die Daten des Systems müssen selektierbar sein.	Um genauere Erkenntnisse bezüglich des Mehrwertes zu erhalten, muss eine Selektion nach Zeitraum, Sparten und individuellen Variablen möglich sein.
3	Das System muss mobil erreichbar sein.	Damit die Systemdaten stets eingesehen werden können, muss das Reportmodul auch mobil anwendbar sein.

Auch in dem Reportmodul soll durch geeignete Visualisierungen der Messfaktoren das einfache Verständnis ermöglicht werden sowie die Reaktionszeiten der Selektionen auf unter zwei Sekunden gehalten werden (siehe Tabelle 12).

Tabelle 12: Nicht-Funktionale Anforderungen an das Reportmodul

Prio	Anforderung	Beschreibung
1	Die Messfaktoren sollten durch Visualisierung einfach und schnell verständlich sein.	Der Mehrwert des Systems sollte insbesondere Personen ohne Kenntniss über das Analyseverfahren verständlich nähergebracht werden.
2	Das System sollte bei Selektionen eine Reaktionszeit von unter 2 Sekunden aufweisen.	Durch schnelle Reaktionszeiten soll eine komfortable Nutzung der Reporting Funktion gewährleistet werden.

Abbildung 28 enthält eine Übersicht des vergangenen Entwicklungsprozesses. Diese beinhaltet vier Meilensteine, welche zugleich für die Fertigstellung der einzelnen Systemkomponenten stehen. Der erste Meilenstein ist die Fertigstellung der Funktionsbibliothek, sodass die gesamte Analyse mittels des R Clients durchgeführt und getestet werden konnte. Dieser Schritt hat mit drei Monaten einen großen Teil des Entwicklungszeitraumes eingenommen, da zum einen ein Einarbeiten in R notwendig war und zum anderen eine weitestgehend explorative Vorgehensweise von verschiedenen Ansätzen durchgeführt wurde. Die Fertigstellung des Mess-Skriptes dauerte hingegen lediglich einen Monat, da hierbei ein Großteil den Funktionen aus der Bibliothek entnommen wurde. Die Entwicklung der RE veranschlagte zwei Monate. Auch hierbei war ein Teil der Zeit für die grundlegende, explorative Einarbeitung in das shiny Package notwendig, die anschließende Entwicklung des Reportmoduls war in 15 Tagen abgeschlossen.

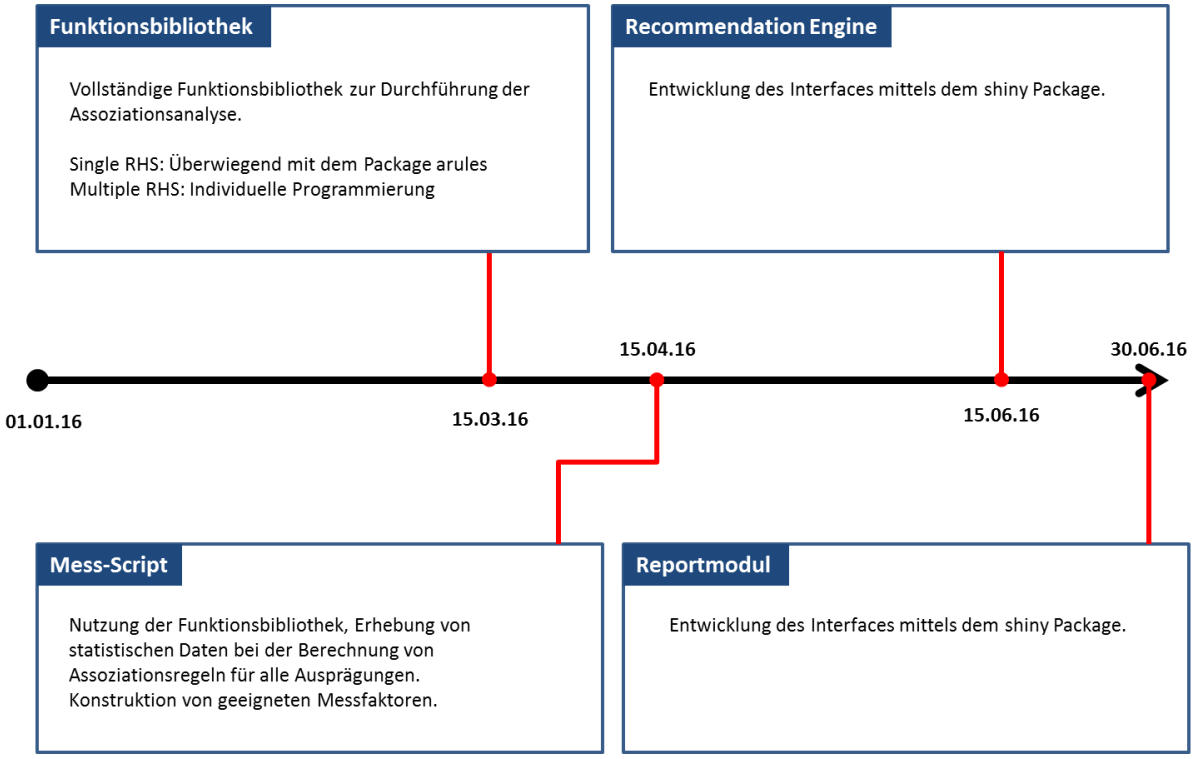


Abbildung 28: Implementierungsplan

4. Entwicklung

In diesem Kapitel wird die Entwicklung des Prototyps dokumentiert. Die Struktur orientiert sich direkt an dem im vorangegangenen Kapitel erörterten Implementierungsplan. In den Beispielen wird ein durchgehender Datensatz verwendet, welcher bei einer Beobachtung von 20 Supermärkten über einen Zeitraum von 30 Tagen entstanden ist. Der „Groceries“ Datensatz enthält 169 eindeutige Produkte, welche über 9835 Transaktionen verteilt sind (vgl. Hahsler, Inside-R, 2006). Der Datensatz hat sich aufgrund des Einsatzgebietes bezüglich Warenkorbanalysen als Standardbeispiel in der Assoziationsanalyse etabliert (vgl. EMC Education Services, 2015; GitHub Inc., 2015).

4.1 Funktionsbibliothek

Die Funktionsbibliothek ist eine Textdatei, welche generische Funktionen von der Datenakquise über Analyse bis hin zu dem Aufbereiten und Steuern der Analyseergebnisse beinhaltet. Es lassen sich hierbei drei Funktionsgruppen unterscheiden. Ein Fokus wird dabei auf die Integration und Aufbereitung der Datengrundlage gelegt. Denn aufgrund des generischen Einsatzes sollen Nutzer eigene Daten einbinden können, was eine entsprechend automatisierte Aufbereitung der Datengrundlage bedingt. Die zweite Funktionsgruppe ist der Hauptteil bestehend aus dem Assoziationsanalyseverfahren, welches entsprechend der Anforderungen aus dem vorherigen Kapitel in das iterative sowie kombinatorische Verfahren unterteilt werden kann. Die dritte Funktionsgruppe ist das Mess-Skript, welches unter Verwendung der Funktionen des Analyseteils alle Regeln eines Datensatzes berechnen und dabei statistische Kennzahlen festhalten soll, welche im Reportmodul zur Darstellung des Systemstatus und -nutzen dienen sollen.

4.1.1 Integration der Datengrundlage (Data Preparation)

Im Sinne des evolutionären Prototypings wird der Datenimport exemplarisch mittels Einbinden von CSV-Dateien ermöglicht (siehe Abbildung 29, vertikale Datenstruktur mit fehlenden Werten). Für den Echtzeitbetrieb sind Importe durch Datenbankzugriffe oder Streams vorgesehen, welche mittels existierender R Packages vollzogen werden können (vgl. Ripley & Lapsley, 2015; Hahsler, Bolanos, & Forrest, 2015).

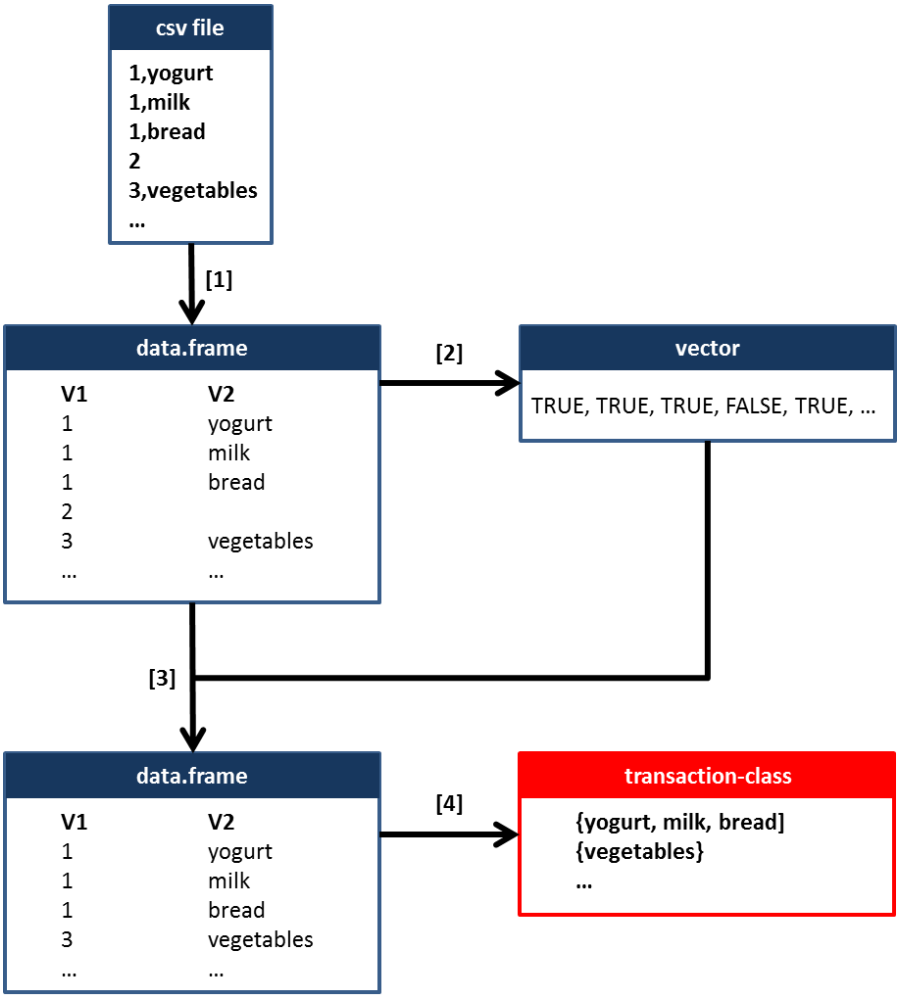


Abbildung 29: Darstellung der Data Preparation

Das grundsätzliche Einbinden einer CSV Datei ist in R mittels der Funktion

[1] read.csv(file, header, separator, quotation)

möglich, die eingelesene Datei wird in ein „data.frame“ überführt. Neben der Pfadangabe der Datei können Parameter hinsichtlich Variablennamen in der ersten Zeile, dem Zeichen des Separators sowie potenziellen Einklammerungen durch Anführungszeichen angegeben werden. Auf dem data.frame werden dann die für die Analyse benötigten Daten selektiert und in einem ersten Schritt auf fehlende Werte überprüft. Konkret wird auf die selektierten Variablen jeweils eine apply Funktion

[2] sapply(selected_variable,

function(x) if(invalid(x)){return(FALSE)}else{return(TRUE)}

)

angewendet, welche die einzelnen Werte mittels der Funktion

invalid(value)

aus dem Package gtools überprüft. Die Funktion gibt einen boolschen Vektor aus, welcher dann für die Behandlung der fehlenden Werte genutzt werden kann. Fehlende Werte sind in Bezug auf die spätere Assoziationsanalyse nicht durch andere Variablen ersetzbar oder grundsätzlich zu schätzen (bspw. durch Mittelwerte etc.). Aus diesem Grund werden fehlende Werte in selektierten Variablen grundsätzlich mittels des berechneten Vektors ausgeschlossen.

[3] data.frame[vector.bool,]

Die selektierten Variablen müssen dann in eine spezielle S4 Klasse „transaction-class“ des arules Packages überführt werden. Dies wird mittels der Funktion

[4] read.transactions(file, format, separator, duplicates, cols)

bewerkstelligt. Bei der Überführung in die Transaktionsklasse werden mehrere Datenaufbereitungen vorgenommen. Zum einen werden die Daten entsprechend dem Format (vertikal, horizontal) in einzelne Transaktionen geschrieben. Dazu müssen neben der Vorgabe der Struktur auch die jeweilig selektierten Variablen angegeben werden (cols). Bei dem Prozess werden die Werte der Variablen zum anderen jeweils in den Datentyp factor konvertiert. Damit ist die Datenaufbereitung abgeschlossen. Die Transaktionsklasse ist das Ausgangsobjekt für die Assoziationsanalyse.

4.1.2 Assoziationsanalyse (Modeling & Evaluation)

Die Assoziationsanalyse wird in zwei separate Analyseschritte unterteilt. Die Hauptanalyse wird iterativ durch Verknüpfungen mehrerer Assoziationsanalysen mit jeweils einer Ausprägung in der RHS durchgeführt. Hierbei wird die Analyse vollständig mittels der Funktionalitäten aus dem arules Packages durchgeführt. Die optionale Analyse von vollständigen Sets an Ausprägungen in Form von Assoziationsregeln mit mehreren Ausprägungen in der RHS wird durch individuelle Funktionen implementiert. Bei beiden Analysen wird im Sinne der Echtzeitberechnung eine Nutzereingabe benötigt. Denn entgegen der ganzheitlichen Berechnung von Regeln für jede Ausprägung einer Datenbank soll in dem Prototyp eine Echtzeitberechnung von Assoziationsregeln für spezielle, durch den Nutzer angegebene Ausprägungen stattfinden.

Single-RHS-Analyse (iterativ)

Der erste Schritt der Assoziationsregelanalyse ist die Berechnung der frequenten Itemsets. Wie bereits in den theoretischen Grundlagen erwähnt, existieren mehrere Algorithmen hierfür (siehe Kapitel 2.2). Im Prototyp kommt der Apriori Algorithmus zum Einsatz. Diese Auswahl fußt vor allen Dingen auf der Flexibilität, denn neben den grundsätzlichen Parametern hin-

sichtlich Support, Konfidenz und Regellänge können bei dem Apriori Algorithmus spezifische Ausprägungen für die LHS oder RHS vorgegeben werden. Somit wird die Regelberechnung auf die relevanten Ausprägungen reduziert. Die weiteren Algorithmen wie Eclat und FP-Growth sind auf die Analyse von allen Regeln einer Transaktionsmenge ausgerichtet und bieten sich insofern nicht für Echtzeitberechnungen an.

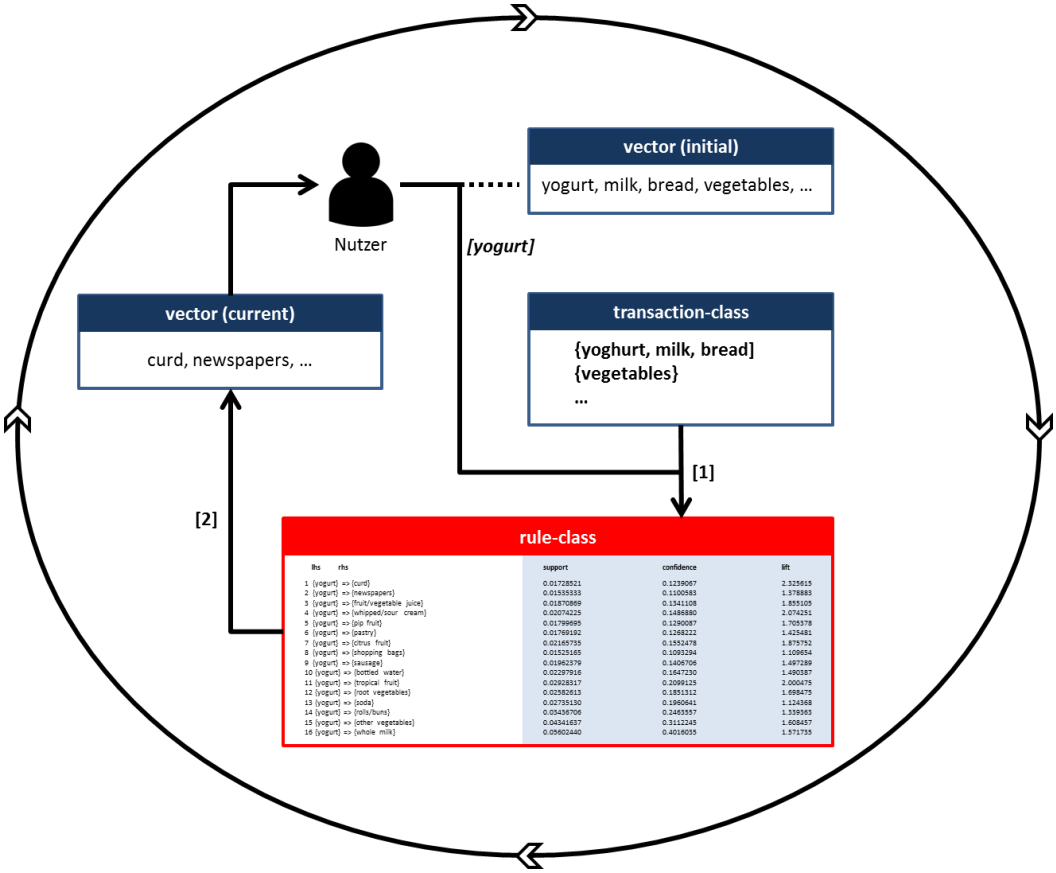


Abbildung 30: Überblick Single-RHS-Analyse (Detailansicht der Rule-Class in Abb. 32)

Der Apriori Algorithmus ist bereits in dem arules Package enthalten und wird wiefolgt aufgerufen.

[1] apriori(transactions,

parameter = list(support, confidence, minlen, maxlen),

appearance = list(lhs, rhs)

)

Insofern muss zwingend eine Transaktionsklasse als Datengrundlage angegeben werden sowie die Parameter Support und Konfidenz. Minlen und Maxlen stehen für optionale Parameter hinsichtlich der Regellänge in Form von Mindest- und Maximallänge. Besonders wichtig sind jedoch die optionalen Parameter der appearance. Denn hier können konkrete Ausprägungen eingetragen werden, sodass der Algorithmus lediglich jene Itemsets berechnet, welche die

angegebenen Ausprägungen enthält. Konkret werden bei der Nutzung der Funktion die Transaktionsklasse, eine Regelmindestlänge, Konfidenz sowie die LHS dynamisch bei der Nutzung der RE parametrisiert. Der Minimalwert des Supports kann entsprechend des Einsatzfeldes und des Motives bei der Nutzung der RE fest vorgegeben werden. Hierbei wird dieser jedoch nicht statisch in Form einer Prozentvorgabe sondern relativ in Bezug zu der sich dynamisch verändernden Transaktionsmenge angegeben. Bspw. „3/Transaktionslänge“ statt „0.1“. Somit kann mit dem Nutzer eine konstante, kritische Minimallänge festgesetzt werden.
Die Regelaufstellung und -berechnung wird innerhalb der Apriori Funktion automatisch durchgeführt, sodass das Ergebnis die entsprechenden Assoziationsregeln in einer speziellen S4 Klasse „rule-class“ ausgegeben wird. Abbildung 31 zeigt ein Beispiel, in welchem die apriori Funktion wiefolgt parametrisiert wurde:

```
regeln <- apriori(Groceries,
                  parameter = list(support =0.015, confidence = 0.1, minlen=2),
                  appearance = list(lhs="yogurt",default="rhs")
                 )
```

Es wurde der Beispieldatensatz „Groceries“ mit einem Support von 0.015, einer Konfidenz von 0.1, einer Minimalregelänge von 2 sowie der spezifischen Ausprägung “yogurt” als LHS analysiert und in der Variable „regeln“ gespeichert. Abbildung 31 enthält das Ergebnis.
Der weiße Bereich steht hierbei für die Assoziationsregeln, wobei die LHS entsprechend der Vorgabe statisch die Ausprägung „yogurt“ enthält. Die Regeln aus dem weißen Bereich sind jeweils vom Datentyp factor. Die im blauen Bereich befindlichen Messfaktoren sind ein data.frame, welches mit dem Befehl

regeln$quality

aus der Rule-Class bezogen und verarbeitet werden kann. Es lassen sich ebenso die Assoziationsregeln mittels

regeln$rules

extrahieren, jedoch erhält man lediglich eine Variable vom Typ factor. Das separate Auslesen der LHS oder RHS ist somit nicht vorgesehen. Da jedoch im Sinne der iterativen Verknüpfung die RHS separat vorliegen muss, um basierend auf dieser weitere Berechnungsschritte vorzunehmen, wurde eine Funktion zum Extrahieren der Regelkomponenten implementiert.

rule-class

	lhs	rhs	support	confidence	lift
1	{yogurt}	=> {curd}	0.01728521	0.1239067	2.325615
2	{yogurt}	=> {newspapers}	0.01535333	0.1100583	1.378883
3	{yogurt}	=> {fruit/vegetable juice}	0.01870869	0.1341108	1.855105
4	{yogurt}	=> {whipped/sour cream}	0.02074225	0.1486880	2.074251
5	{yogurt}	=> {pip fruit}	0.01799695	0.1290087	1.705378
6	{yogurt}	=> {pastry}	0.01769192	0.1268222	1.425481
7	{yogurt}	=> {citrus fruit}	0.02165735	0.1552478	1.875752
8	{yogurt}	=> {shopping bags}	0.01525165	0.1093294	1.109654
9	{yogurt}	=> {sausage}	0.01962379	0.1406706	1.497289
10	{yogurt}	=> {bottled water}	0.02297916	0.1647230	1.490387
11	{yogurt}	=> {tropical fruit}	0.02928317	0.2099125	2.000475
12	{yogurt}	=> {root vegetables}	0.02582613	0.1851312	1.698475
13	{yogurt}	=> {soda}	0.02735130	0.1960641	1.124368
14	{yogurt}	=> {rolls/buns}	0.03436706	0.2463557	1.339363
15	{yogurt}	=> {other vegetables}	0.04341637	0.3112245	1.608457
16	{yogurt}	=> {whole milk}	0.05602440	0.4016035	1.571735

Abbildung 31: Beispiel einer Rule-Class

Diese nutzt die Funktion

gsub(Bedingung,Ersetzung,Variable)

mit welcher man in einer gegebenen Variable Zeichen einer bestimmten Vorgabebedingung durch ein weiteres, vorgegebenes Zeichen ersetzt. Die RHS einer Variable „regeln“ vom Typ rule-class wird mit

gsub(".*=> |\\{|\\}", "", regeln$rules)

ausgelesen. Hierbei werden alle Zeichen vor und inklusive des Pfeils mit einem nachfolgenden Leerzeichen sowie die jeweils durch ein Escape „\\“ gekennzeichneten, geschweiften Klammern durch den String „“, also nichts ersetzt. Es verbleibt somit lediglich die RHS. Analog wird die LHS mit

gsub("\\{|\\}| =>.*", "", regeln$rules)

ausgelesen. Diese Einzeloperationen werden in eine konkludierte Funktion

[2] decode.ItemsFromRules(variable,modus=*lhs/rhs*)

zusammengefasst.

Mittels der aufgezeigten Funktionalität lässt sich nun bereits eine iterative Regelberechnung durchführen. Der Nutzer erhält initial alle eindeutigen Ausprägungen aus der Transaktionsklasse. In dem gezeigten Beispiel wählt er hierbei „yogurt“ aus. Es werden dann mit der apri-

ori Funktion die Assoziationsregeln in Form von der Rule-Class unter Einbezug der statisch vorgegebenen LHS und der Regellänge von 2 berechnet (die Regelklasse wird in der später vorgestellten RE entsprechend visualisiert). Aus der Rule-Class werden dann die Ausprägungen der RHS ausgelesen und dem Nutzer als neue Ausprägungsauswahl zur Verfügung gestellt. Damit endet der Iterationszyklus und der Nutzer kann nun neben der Ausprägung „yogurt“ eine zusätzliche Ausprägung auswählen. Würde er bspw.“newspaper“ auswählen, würde der Iterationszyklus mit der statischen LHS „yogurt, newspaper“ neugestartet. Auf diese Weise können durch iterative Verknüpfungen von einzelnen Assoziationsanalysen mit einer Ausprägung in der RHS mittels der bestehenden Packages in R performant Assoziationsregeln mit potenziell vielen Ausprägungen berechnet werden. Durch die Iterationsschritte und den jeweiligen Visualisierungen wird der Nutzer somit transparent schrittweise über die Assoziationen der ausgewählten Ausprägungen informiert.

Multiple-RHS-Analyse (kombinatorisch)

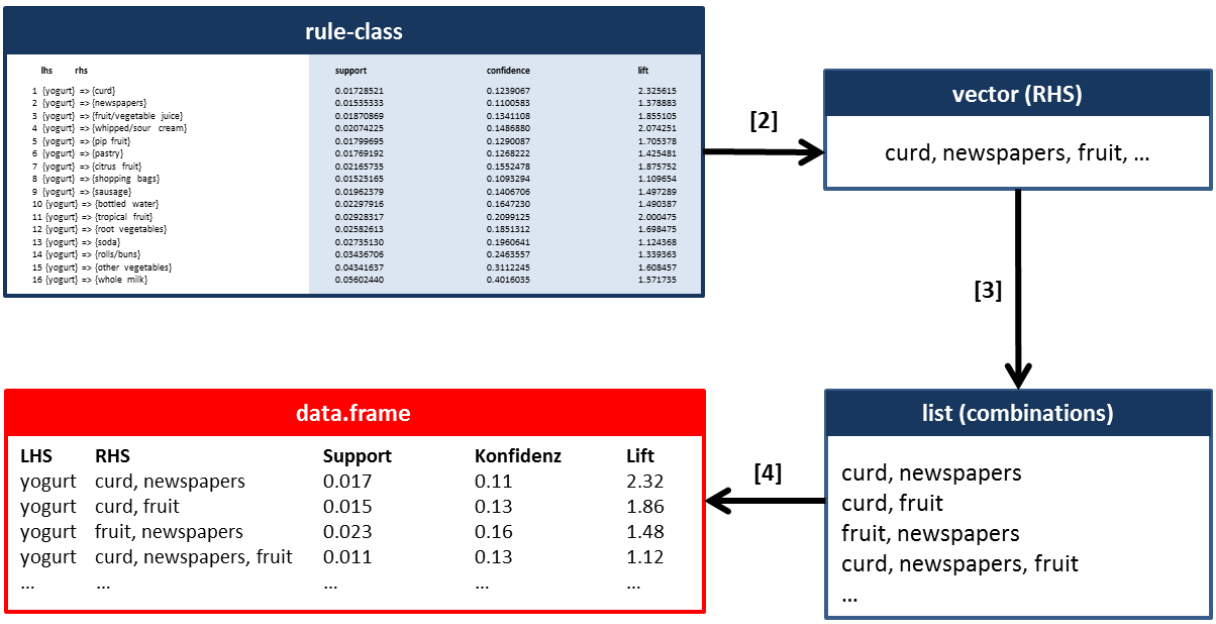


Abbildung 32: Überblick Multiple-RHS-Analyse

Neben der iterativen Berechnung von Assoziationsregeln sollen mit dem Prototyp auch Regeln mit mehreren Items in der RHS berechnet werden können. Dies soll als optionaler Prozess zur Verfügung stehen. Wie bereits zu Beginn des Kapitels dargelegt, ist diese Berechnungsweise in der Praxis kaum eingesetzt, da der Aufwand bei der Berechnung aller Regeln einer Datenbank sehr hoch ist. In dem Prototyp wird die Kombinationsberechnung jedoch innerhalb der iterativen Berechnungsweise optional zur Verfügung gestellt. Des Weiteren wird der Rechenaufwand anhand der Anzahl der Regeln der Single-RHS-Analyse eingegrenzt. Im Gegensatz zu den im arules Package bestehenden Funktionen hinsichtlich der Sin-

gle-RHS-Analyse, wurden keine Funktionen zur Multiple-RHS-Analyse implementiert, sodass es einer gänzlichen Neuentwicklung bedarf.

Das Ausgangsobjekt ist in diesem Falle die Rule-Class der iterativen Berechnung (siehe Abbildung 32). Es werden hierbei die Ausprägungen der RHS aus der Rule-Class mittels der bereits vorgestellten Funktion „decode.ItemsFromRules“ extrahiert **[2]**. Anschließend werden mittels Kombinatorik alle Kombinationen dieser Ausprägungen gebildet. Konkret wird dies mittels der im Package gtools befindlichen Funktion

combinations(x, l, v, r)

bewerkstelligt. Hierbei werden alle Kombinationen der vorgegebenen Länge „l“ der Werte „x“ des Inputvektors „v“ gebildet und dabei Wiederholungen „r“ zu-/weggelassen. Um alle möglichen Kombinationen verschiedener Längen zu erhalten, muss eine entsprechende Schleifenkonstruktion gebildet werden. Wie viele Kombinationslängen hierbei berechnet werden, hängt von der Anzahl der Ausprägungen ab. Die potenzielle Anzahl an Kombinationen ist das hauptsächliche Problem der Multiple-RHS-Analyse (siehe Abbildung 33). Mit steigender Anzahl der Ausprägungen („x“) steigt die Anzahl möglicher Kombinationen („combinations“) exponentiell (ohne Reihenfolge, ohne Wiederholung).

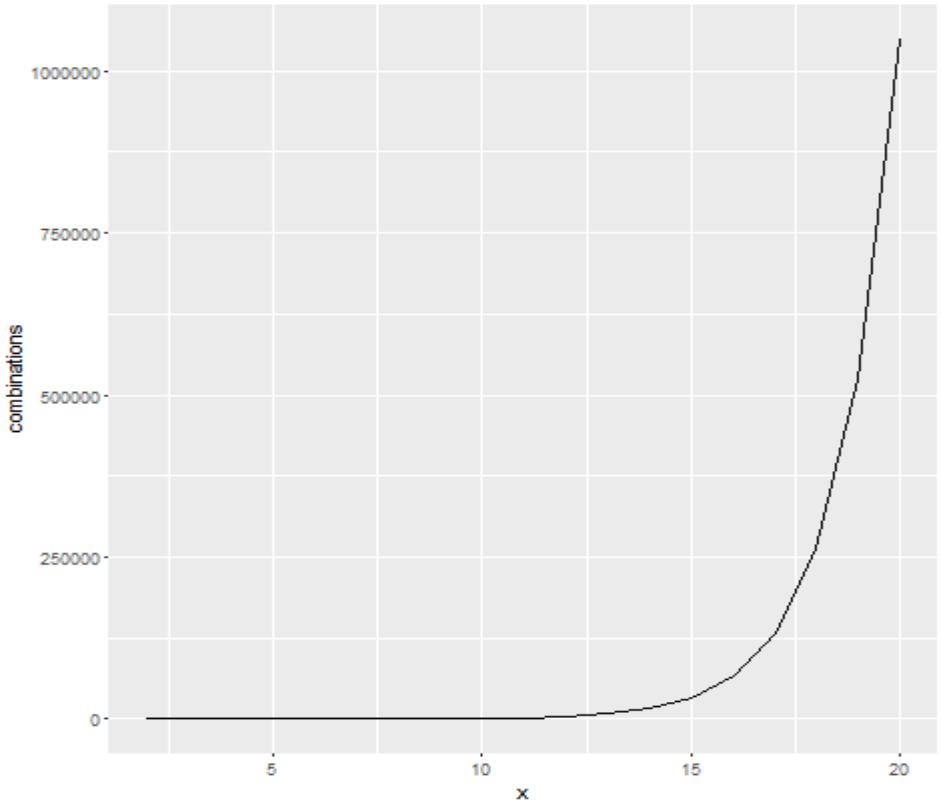


Abbildung 33: Entwicklung von Kombinationsanzahlen

Daher wird an dieser Stelle in Abhängigkeit von der Anzahl der aus der Rule-Class ausgelesenen RHS-Ausprägungen die Kombinationsberechnung gesteuert. Wenn die Anzahl der Ausprägungen größer als ein festgelegter Wert ist, wird nicht die gesamte Kombinationsberechnung vorgenommen. Der Wert ist individuell nach Ansprüchen der Berechnungszeit innerhalb des Prototyps sowie vorhandener Hardware festzusetzen. Sobald der Wert überschritten wird, bieten sich verschiedene Schritte für die Reduzierung der Kombinationen an. Zum einen sind Manipulationen hinsichtlich der Kombinationslängen denkbar. Die Reduzierung der Kombinationslänge auf zwei, im Sinne der Unterstützung der iterativen Berechnung, und/oder auf die Maximallänge, im Sinne von vollständigen Sets, grenzen die Anzahl der Kombinationen stark ein. Zum anderen sind Manipulationen hinsichtlich der Ausprägungen denkbar, indem hier eine Auswahl getroffen wird. Eine Eingrenzung durch Selektion der wichtigsten Ausprägungen (nach Support/Konfidenz/Lift) hat ebenfalls einen Reduzierungseffekt wie die Eingrenzung der Kombinationslängen. Die Grundeinstellung im Prototyp ist die Eingrenzung der Kombinationslänge auf zwei, bei einer Überschreitung des Grenzwertes von zehn. Dies begründet sich darauf, dass der Prototyp iterativ genutzt wird. Daher sollen dem Nutzer zumindest Kombinationen der **Länge 2**, jedoch für alle Ausprägungen ausgegeben werden. Der **Grenzwert von 10** obliegt der vorher durchgeführten Kombinationsberechnung (siehe Abbildung 33).

```
[3] if(length(auspraegungen)>10){
combinations(length(auspraegungen), 2, auspraegungen[1:n], repeats.allowed=FALSE);
}else{
for(i in 2:length(auspraegungen)){
combinations(length(auspraegungen), i, auspraegungen[1:n], repeats.allowed=FALSE);
}
```

Wenn also mehr als zehn Assoziationsregeln in einem iterativen Berechnungsschritt vorliegen und der Nutzer die Kombinationsberechnung zuschaltet, werden lediglich die Kombinationen der Länge zwei zur Unterstützung der iterativen Berechnung berechnet, andernfalls werden in einer for-Schleife alle Kombinationslängen berechnet. Die gesamte Berechnung der Kombinationen ist in einer Funktion

[3] get.rule.candidates(rule.class)

zusammengefasst.

Das Ergebnis der Kombinationsberechnung ist eine Liste von Vektoren, welche jeweils eine Kombination in Form eines Regelkandidaten enthält. Für diese Regelkandidaten müssen nun

die Messfaktoren berechnet werden. Der Support wird dabei mittels der im arules Packages enthaltenen Funktion

support(itemMatrix, transaction-class)

berechnet. Während die Transaktionsklasse bereits aus der Single-RHS-Analyse vorliegt, müssen die zu berechnenden Ausprägungen in eine für das arules Package spezifische item-Matrix überführt werden. Diese ist eine speziell strukturierte Matrix, welche die Ausprägungen enthält. Die Konvertierung des Ausprägungsvektoren in eine itemMatrix kann mit der ebenfalls im arules Package enthalten Funktion

as(Ausprägungsvektor,“itemMatrix“)

durchgeführt werden.
Insofern wird zuerst die Ausprägung der statischen LHS in ein itemMatrix überführt und anschließend der Support berechnet.

itemMatrix.LHS ← as(LHS,“itemMatrix“)
support.LHS ← support(itemMatrix.LHS,transaction-class)

Anschließend werden auf die gleiche Weise die Supportwerte der Ausprägungsvektoren (RHS), sowie der Regelitemsets (LHS+RHS) berechnet. Nun können die weiteren Messfaktoren mittels der Supportwerte berechnet werden.

Konfidenz = support.itemset / support.LHS
Lift = Konfidenz / support.RHS

Die jeweiligen Regeln sowie Messfaktoren werden in ein data.frame überführt, welche das finale Ergebnis der Multiple-RHS-Analyse darstellt. Die Funktionen hinsichtlich der Berechnung der Assoziationsregeln aus den Regelkandidaten sind in der Funktion

[4] get.rules.multiple(LHS, Liste[Kombinationen], transaction-class)

konkludiert.

4.2 Mess-Skript

Das Mess-Skript führt eine Assoziationsanalyse im Sinne der Berechnung aller Regeln aus der Datenbank durch. Dieses wird in regelmäßigen, zeitlichen Abständen durchgeführt, sodass der Fokus hier nicht auf der Echtzeitanalyse liegt. Es sollen stattdessen alle Regeln der Datenbank berechnet und aus diesen statistische, zeitpunktbezogene Kennzahlen herausgestellt werden, welche Entscheidungsträgern den Status und die Entwicklung der im Betrieb befind-

lichen RE liefern, sowie zum Monitoring dienen sollen. Nachfolgend werden im ersten Schritt die Kennzahlen und im Weiteren die Messung dargestellt.

4.2.1 Kennzahlen

Das Reportmodul enthält drei Kennzahlen: Regelbestand, Regelbestandsentwicklung sowie Regelabdeckung. Grundlage für die Berechnung der Kennzahlen ist eine Assoziationsanalyse der gesamten Datenbank. Die Parameter, welche der Regelberechnung zu Grunde liegen, können vom Entscheidungsträger anhand des Anwendungskontextes vorgegeben werden und orientieren sich an den Einstellungen der RE. Je nach Motiv, können entweder starke oder auch schwache Regeln bezüglich der Messfaktoren berechnet werden. Wenn eine geringe Datenlage vorliegt und der Prototyp explorativ mit hohem Nutzereinbezug verwendet wird, sind niedrige Messfaktoren anzuwenden. Bei einer ausreichend großen Datengrundlage und starken Assoziationen könnte der Prototyp als Hintergrundprozess mit niedrigem Einbezug des Nutzers in Form einer Liste an assoziierten Ausprägungen als Ergebnis verwendet werden. Dann sind höhere Messfaktoren einzustellen. Die Grundeinstellung des Protoyps liegt bei niedrigen Messfaktoren in Form von 3/Transaktionsmenge (Support), sowie einer Konfidenz von 0,1. Denn grundsätzlich sollten zuerst alle möglichen Regeln berechnet werden, um den Informationsverlust durch Erhöhung der Messfaktoren vorab abschätzen zu können. Der Support muss dem der Grundeinstellung der RE gleich sein, damit möglichst die gleichen Assoziationsregeln berechnet werden. Die Konfidenz wird allerdings durch den Nutzer individuell verändert und kann somit nicht genau abgeglichen werden. Der Nutzer kann in der RE bis zu einem Minimalwert von 0.1 heruntergehen. Im Sinne der explorativen Regelberechnung wurde dieser Wert entsprechend für das Messskript übernommen.

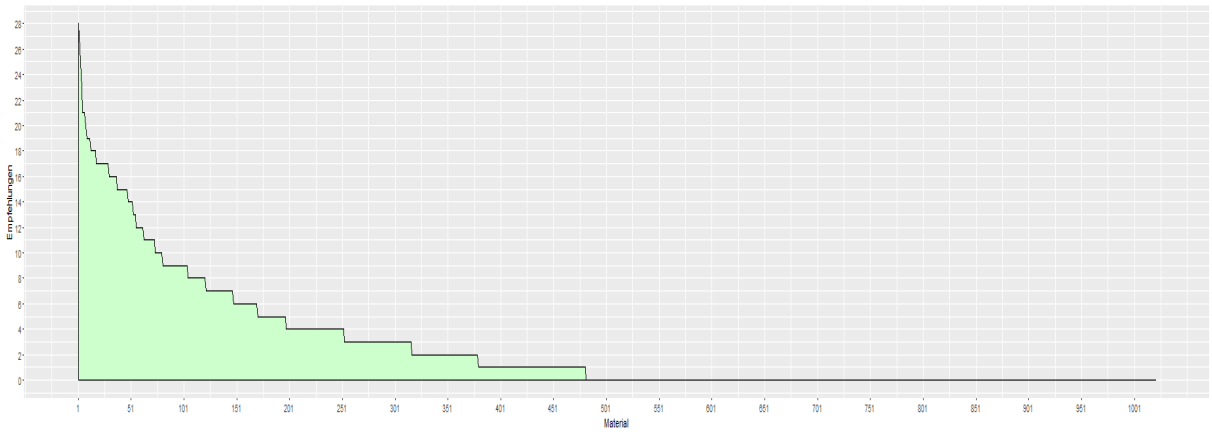


Abbildung 34: Darstellung der Regelbestände eines spezifischen Zeitpunktes

Der Regelbestand ist definiert als die Gesamtheit an Assoziationsregeln aller Ausprägungen aus der Datenbank, strukturiert nach den Ausprägungen. Es wird insofern für jede Ausprä-

gung aus der Datenbank die Menge der jeweiligen Assoziationsregeln festgehalten. Entscheidend für den Einsatz dieser Kennzahl ist die geeignete Visualisierung. Denn der Regelbestand soll in einem sortierten Balkendiagramm dargestellt werden (siehe Abbildung 34). Hierbei können nun nach kumulierten Ausprägungen verschiedene Aussagen getroffen werden. In dem Beispiel würde bspw. ersichtlich, dass für 300 Ausprägungen mindestens drei Assoziationen vorliegen. Auf diese Weise lassen sich verschiedene Ausprägungssektoren ausfindig machen. Der Sektor mit mindestens 20 Assozationen, welcher für 50 Ausprägungen zutrifft, könnte bspw. als negativ angesehen werden, weil diese Ausprägungen besonders häufig verwendet werden und möglicherweise durch ineffizientes Hinzufügen zu zahlreichen Prozessen charakterisiert sind. Insofern könnten anhand des visualisierten Regelbestandes Ziele ausgesprochen werden, bspw. durch den Einsatz der RE eben jenen Sektor zu reduzieren.

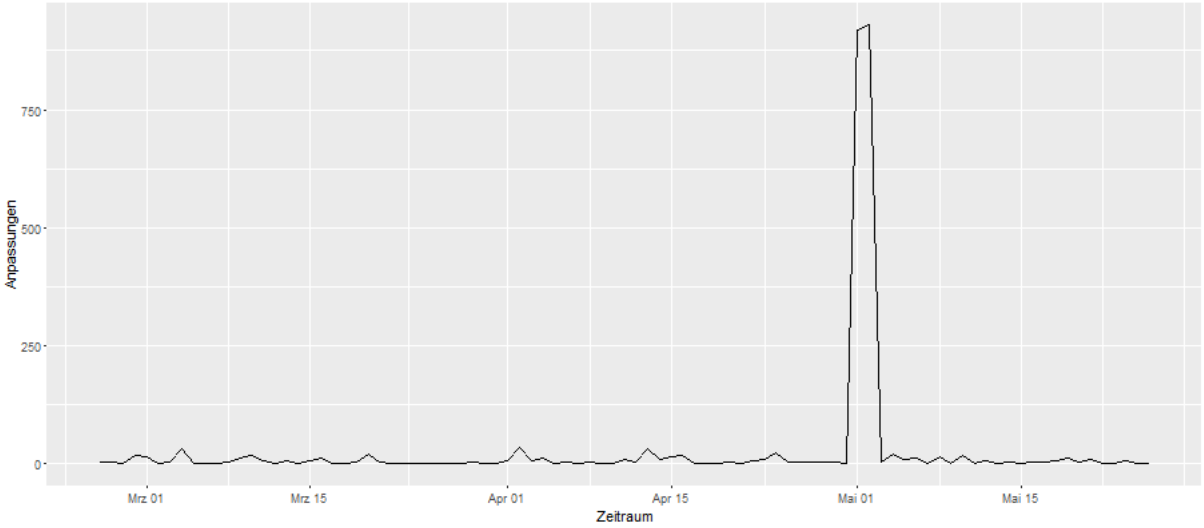


Abbildung 35: Darstellung der Regelbestandsentwicklung innerhalb eines Quartals

Die Regelbestandsentwicklung beschreibt die Entwicklung des gesamten Regelbestandes. Es wird die Menge an Regelbeständen einer Datenbank zum Zeitpunkt t-1 von der Menge an Regelbeständen der Datenbank zum Zeitpunkt t subtrahiert. Der Betrag des Ergebnisses stellt insofern die Veränderung der Regelbestände einer Datenbank in Bezug zu der letzten Messung dar. Abbildung 35 enthält die im Reportmodul genutzte Visualisierung und kann als eine Art „Pulsschlag“ des Systems interpretiert werden. Dem Entscheidungsträger soll somit aufgezeigt werden, wie stark sich das System im Betrieb kalibriert. Bei besonders hohen Veränderungen in den Regelbeständen (siehe Peak bei 01.Mai in Abbildung 35) wäre diese Kennzahl vor allem auch für das Monitoring zu nutzen. Denn ungewöhnlich hohe Veränderungen könnten auf Systemfehler zurückzuführen sein.

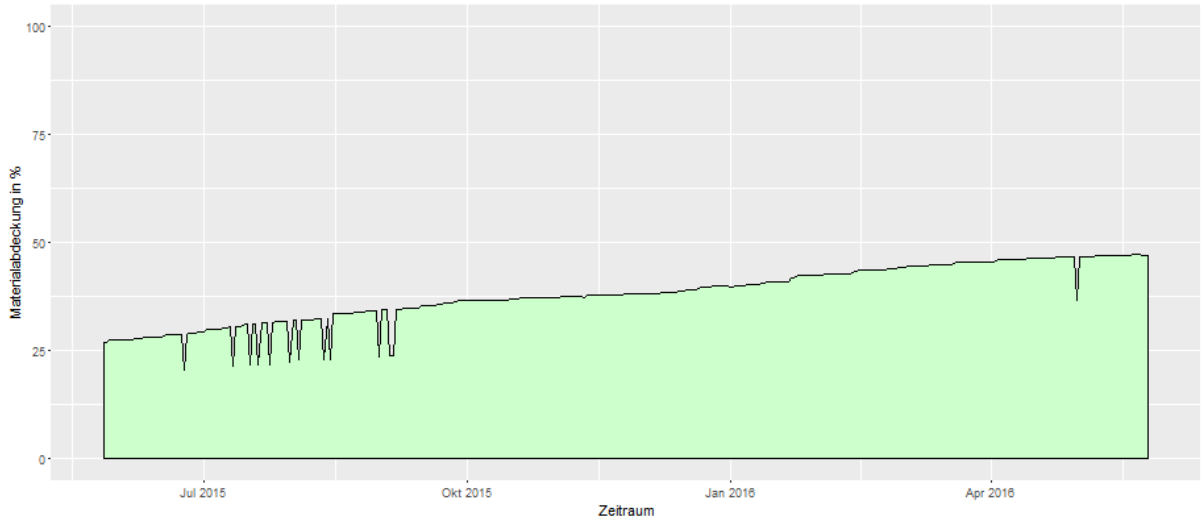


Abbildung 36: Darstellung der Regelabdeckung innerhalb eines Jahres

Die Regelabdeckung ist ein prozentualer Wert im Bereich [0,100], welcher das Verhältnis von Ausprägungen mit mindestens einer Assoziationsregel gegenüber allen Ausprägungen darstellt. Man teilt die Anzahl an Ausprägungen mit mindestens einer Assoziationsregel durch die Gesamtanzahl an Ausprägungen. Liegt die Regelabdeckung bspw. bei 50%, so hat die Hälfte aller Ausprägungen mindestens eine Assoziation. Abbildung 36 zeigt eine beispielhafte Darstellung einer Regelabdeckungssmessung über einen Zeitraum von einem Jahr. Hierbei soll dem Entscheidungsträger vermittelt werden, wie die Assoziationen innerhalb der Datenbank verteilt sind, ob bspw. viele Regeln auf einen kleinen Teil der gesamten Ausprägungen oder über die gesamte Ausprägungsmenge verteilt sind und welcher Entwicklung die Verteilung unterliegt. Auch dieser Messfaktor kann für das Monitoring genutzt werden, denn die in Abbildung 36 enthaltenen Ausschläge können auf Systemfehler hinweisen.

4.2.2 Messung

Das Mess-Skript ist zeitpunktbezogen auf die Datenbank anzuwenden und berechnet für jede Ausprägung die Assoziationsregeln im Sinne der Single-RHS-Analyse. Die Multiple-RHS-Analyse wird nicht vorgenommen, da für das Reportmodul lediglich die grundsätzlichen Assoziationen interessant sind. Es wird also konkret die Single-RHS-Analyse für jede Ausprägung vorgenommen, sodass dies weitestgehend mit den bereits implementierten Funktionen durchgeführt werden kann.

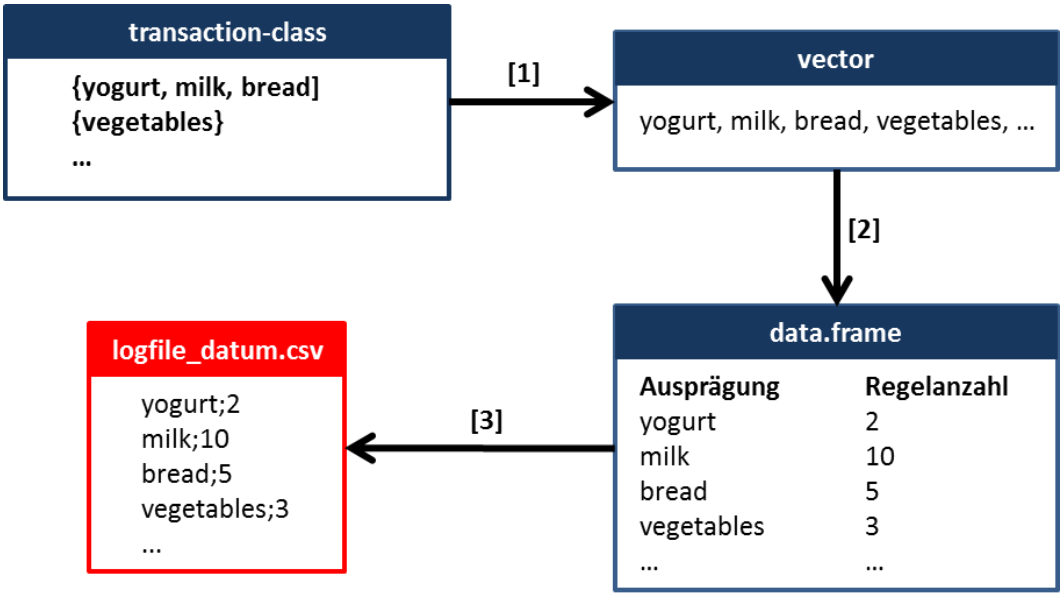


Abbildung 37: Übersicht Mess-Skript

Grundlage ist die in der Datenaufbereitung (siehe Kapitel 4.1.2) aus der Datenbank ausgelesene transaction-class (siehe Abbildung 37). Im ersten Schritt werden alle eindeutigen Ausprägungen aus der transaction-class extrahiert.

[1] unlist(as(itemInfo(transactions),"vector"))

Dazu wird die in dem arules Package enthaltene Funktion

itemInfo(transactions)

verwendet, welcher alle eindeutigen Ausprägungen in der bereits bekannten itemMatrix ausgibt. Der Befehl

as(itemMatrix,“vector“)

konvertiert diese in eine Liste von Vektoren, welche jeweils eine Ausprägung enthält, und mit

unlist(Liste[Vektoren])

in einen einzelnen Vektor überführt wird.

Auf diesen Vektor wird anschließend eine apply Funktion ausgeführt, welcher die Single-RHS-Analyse für jede Ausprägung veranlasst.

[2] apply(vector[ausprägungen],function(x) single.rhs.analysis(x))

Die Funktion single.rhs.analysis führt die bereits vorgestellte Single-RHS-Analyse durch und speichert hierbei jedoch lediglich die Anzahl an Assozationsregeln. Es wäre ebenfalls denkbar, genauere Informationen hinsichtlich Support, Konfidenz sowie Lift festzuhalten. Im Sinne des evolutionären Protoypings wurde dies jedoch zurückgestellt. Das Ergebnis in Form eines data.frames mit Ausprägungen und der Regelanzahl wird dann in ein Logfile gesichert.

[3] write.table(data.frame,file="logfile_datum.csv",sep=";")

Die Logfiles dienen anschließend als Grundlage für das Reportmodul.

Neben der verwendeten Lösung der verketteten Single-RHS-Analyse in der apply Funktion wäre alternativ eine Assoziationsanalyse mittels Eclat- oder FP-Growth-Algorithmus denkbar (der Eclat-Algorithmus ist ebenfalls im arules-Package enthalten) bzw. performanter. Jedoch bietet die verwendete, individuelle Lösung eine höhere Flexibilität. Konkret können beispielsweise zusätzliche Messfaktoren bezüglich der Assoziationsanalyse einfach integriert werden. Denn bei der Single-RHS-Analyse per Apriori-Algorithmus können an jeglichen Stellen Modifikationen vorgenommen werden, welche bei den weiteren Algorithmen aufgrund der speziellen Datenstrukturen schwierig/nicht möglich sind. Analog können außerdem Funktionen bezüglich Monitoring eingebettet werden. Insofern ist die verwendete Lösung zwar weniger performant, jedoch sehr flexibel. Da das Mess-Skript zeitpunktbezogen und ohne Involvierung des Nutzers ausgeführt wird, wurde hierbei die Performanz der generischen Verwendung/Weiterentwicklung untergeordnet.

4.3 Interfaces (Deployment)

In diesem Kapitel werden die jeweils mit dem R shiny Package erstellten Interfaces vorgestellt. Einführend werden an dieser Stelle einige Besonderheiten von shiny vorgestellt.

Eine shiny App besteht aus einem Server- sowie Interfaceskript. In dem Interfaceskript werden Inputdaten definiert (z.B. durch Texteingaben, Radio Buttons etc.) und dann im Serverskript zu Outputdateien (Text, Tabellen, Grafiken, etc.) verarbeitet, welche anschließend wiederum im Interfaceskript angezeigt werden. Im Interfaceskript könnte beispielsweise ein Input in Form einer Zahlenangabe in einem Textfeld eingelesen werden.

***Interface:* textInput(„squareNumber", „Zu quadrierende Zahl")**

In dem Serverskript wird der Input dann verwendet und ein Output berechnet.

***Server:* output$squareResult ← renderText({squareFunction(input$squareNumber)})**

Der Output wird dann im Interfaceskript ausgegeben.

***Interface:* textOutput(„squareResult")**

Ein maßgebliches Konstrukt von shiny sind dabei reaktive Objekte, welche wiederholende Methodenaufrufe basierend auf Nutzerinputs steuern. Es lässt sich beispielsweise ein reaktives Objekt „barplot" erstellen, welches durch eine individuelle Funktion get.barplot() konstruiert wird.

***Server:* output$barplot ← renderPlot({get.barplot(input$variable1, input$variable2)})**

„renderPlot" ist hierbei genau wie „renderText" eine spezifische Funktion des shiny Packages, mit welcher ein Output erzeugt wird. Ein Outputobjekt ist stets reaktiv und wird immer dann neu berechnet, wenn sich eine der Inputvariablen ändert. Insofern würde der im Interface abgebildete Barplot immer dann neu berechnet, wenn der Nutzer eine der Inputvariablen „variable1" oder „variable2" ändert. Reaktive Objekte können jedoch auch ohne direkte input-output Beziehungen gebildet werden.

***Server:* tmpResult ← reactive({customFunction(input$x, y())})**

„tmpResult" ist hierbei ein individuelles reaktives Objekt, welches nicht für einen direkten Output sondern weiterführende Berechnungen konstruiert wurde. Dieses Objekt wird von R als Funktion behandelt und daher in der weiteren Verwendung mit „tmpResult()" angesprochen. Die hinzugefügten Funktionsklammern sind dabei stets leer, da die Inputparameter in der Objektkonstruktion definiert werden. Insofern würde das Objekt „tmpResult()" immer dann neu berechnet, wenn sich die Inputvariable „x" oder das reaktive Inputobjekt „y()" ändert.

Es können bei der Verbindung von reaktiven Objekten auch Fälle auftreten, in welchen nicht alle Inputparameter eine erneute Berechnung auslösen sollen. Dies kann mit dem Befehl „isolate()" gesteuert werden.

***Server:* tmpResult2 ← reactive({customFunction2(isolate(input$x), y())})**

In dem Beispiel würde „tmpResult2" nicht neu berechnet, wenn ein Nutzer die Variable „x" ändert, sondern nur, wenn sich das reaktive Objekt „y()" ändert.

Insofern kann mit shiny aufgrund der reaktiven Konstrukte in Kombination mit einer flexiblen, partiellen Steuerung der Auslösungsstruktur ein ideal auf den Nutzer zugeschnittener Programmablauf erstellt werden.

4.3.1 Recommendation Engine

Das Interface der RE ist in zwei Bereiche unterteilt. Ein Bereich dient der Datenintegration (siehe Abbildungen 38 und 39) und der Weitere enthält das Hauptinterface (Abbildung 40).

Datenintegration

(1) stellt eine Statuszeile dar, in welcher der Nutzer während der Datenintegration über den aktuellen Stand informiert wird (siehe Abbildung 38). (2) enthält eine Möglichkeit des Uploadens einer CSV Datei, dies ist im Prototyp die einzige Möglichkeit, Daten zu importieren. Des

Weiteren kann die Anzahl der in der Datenvorschau (5) abgebildeten Zeilen angegeben werden (siehe Abbildung 39). (3) enthält die standardmäßigen Einstellungen zur Konfiguration einer CSV Datei. (4) enthält die spezifischen Einstellungen der Datenformatierung für die Assoziationsanalyse. Der Nutzer muss angeben, in welchem Format sich die Daten befinden und welche Variablen untersucht werden sollen (siehe Kapitel 3.1.2).

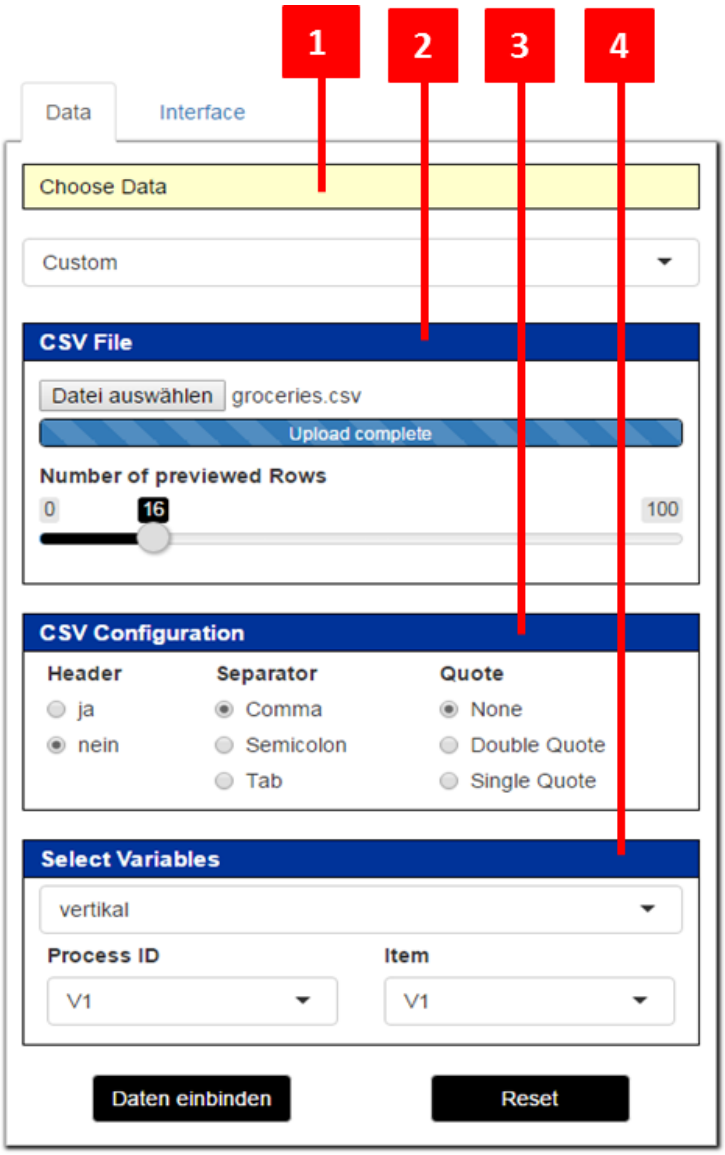
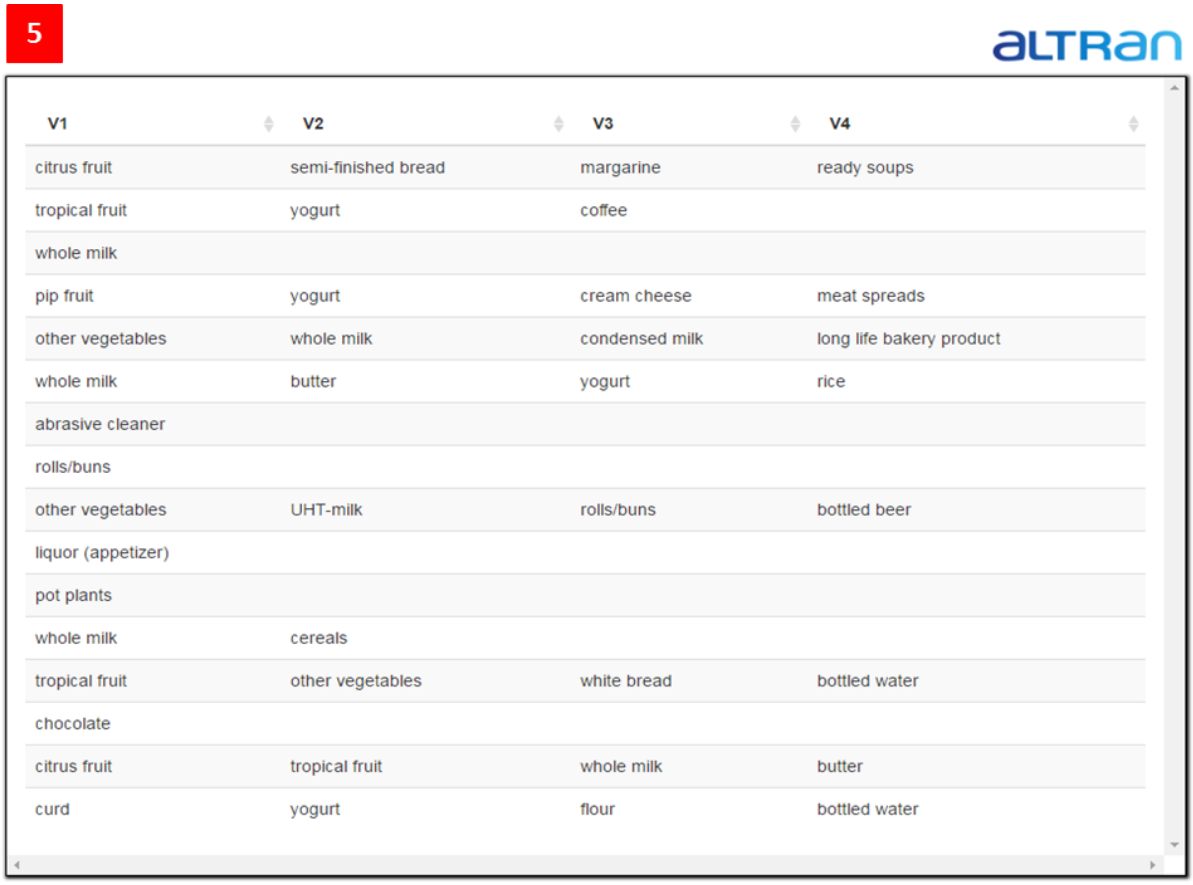


Abbildung 38: Datenintegration innerhalb der RE (Konfiguration)



V1	V2	V3	V4
citrus fruit	semi-finished bread	margarine	ready soups
tropical fruit	yogurt	coffee	
whole milk			
pip fruit	yogurt	cream cheese	meat spreads
other vegetables	whole milk	condensed milk	long life bakery product
whole milk	butter	yogurt	rice
abrasive cleaner			
rolls/buns			
other vegetables	UHT-milk	rolls/buns	bottled beer
liquor (appetizer)			
pot plants			
whole milk	cereals		
tropical fruit	other vegetables	white bread	bottled water
chocolate			
citrus fruit	tropical fruit	whole milk	butter
curd	yogurt	flour	bottled water

Abbildung 39: Datenintegration innerhalb der RE (Ausgabe)

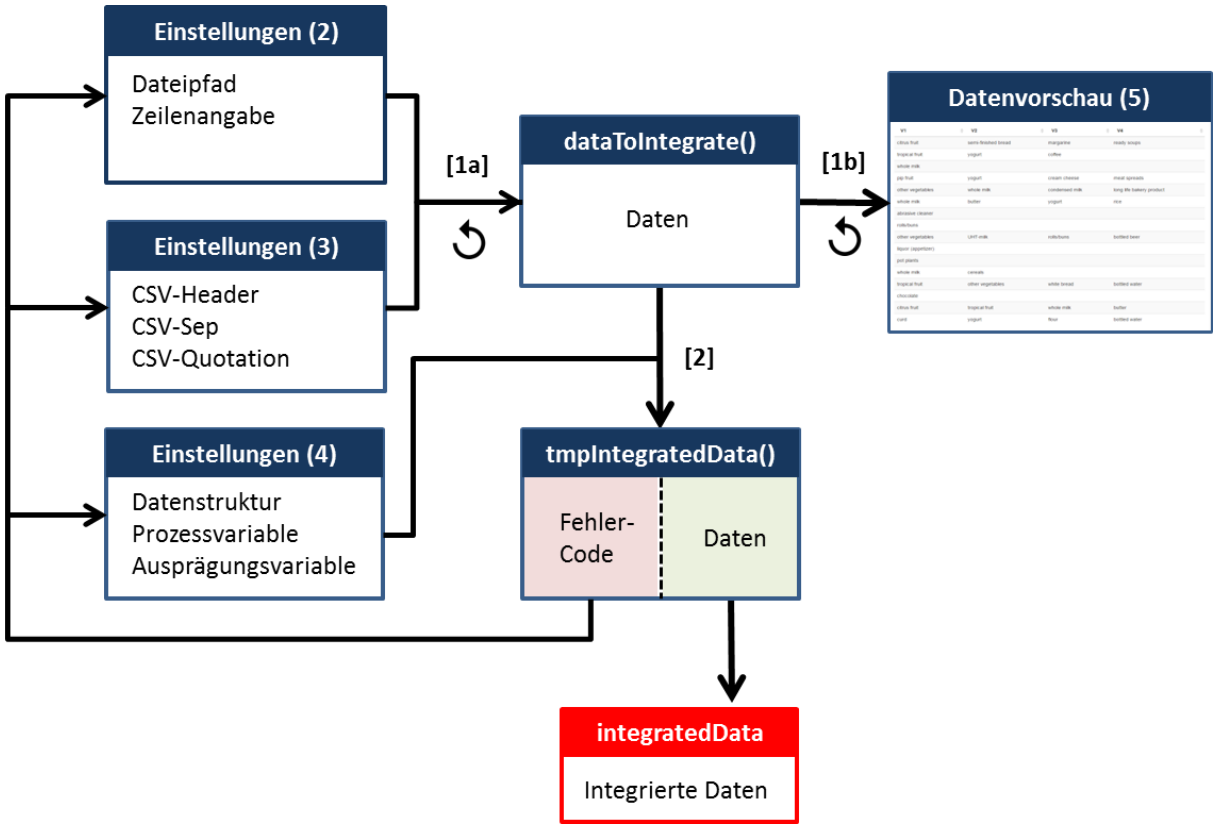


Abbildung 40: Überblick der Datenaufbereitung in der RE

Die Einstellungen werden in einem reaktiven Objekt zusammengeführt, in welchem die Funktion get.data() mit den Parametern Dateipfad, CSV-Header, CSV-Separator, CSV-Quotation sowie der Anzahl der anzuzeigenden Zeilen aufgerufen wird (siehe Abbildung 40).

[1a] reactive({get.data(path(), input$header, input$sep, input$quoted, input$rows)})

Der Dateipfad „path()“ ist hierbei ein reaktives Objekt, welches die temporäre URL der hochgeladenen Datei enthält. Die Funktion get.data() führt dann den bekannten read.csv() Befehl durch. Der eingelesene Datensatz wird dann für die Datenvorschau in eine entsprechende Tabelle überführt.

[1b] renderDataTable({data_preview()},options = list(pageLength = input$rows))

Durch diese Konstruktion erhält der Nutzer einen Überblick seiner konfigurierten Daten in Echtzeit und kann Änderungen on-the-fly vornehmen. Dies bezieht sich auf die Einstellungen (2) und (3), da diese in der Datenvorschau sichtbar sind. Die notwendigen Einstellungen zur Datenstruktur und Variablenselektion in (4) werden erst bei dem Einbinden überprüft.

[2] try(get.customTransactions(dataToIntegrate(), input$auspraegung, input$prozess, input$datastructure), silent=TRUE)

Hierbei werden die konfigurierten Daten in einer try() Funktion integriert. Dabei wird die in Kapitel 4.1.1 vorgestellte Funktion read.transactions() innerhalb der get.CustomTransactions() ausgeführt. Durch den Silent-Modus hat das Ergebnis des Funktionsaufrufes keinen direkten Einfluss auf den Programmablauf, sondern speichert entweder die korrekten Daten oder einen individuellen Fehlercode im erzeugten Objekt. Auf Basis dieses Objektes wird dann der weitere Ablauf gesteuert. Im Falle eines Fehlers wird die Datenintegration gestoppt und eine Fehlermeldung unter (1) angezeigt (die jeweilige Statusmeldung grenzt den Fehler auf einen der Einstellungensbereiche 2-4 ein). Durch diese Vorgehensweise werden Systemabstürze verhindert, was die in den Anforderungen enthaltene Fehlertoleranz erfüllt. Wenn die Datenintegration erfolgreich war, wird eine entsprechende Meldung in (1) angezeigt und die RE kann mit den integrierten Daten mit dem Hauptinterface genutzt werden.

Hauptinterface

Das Hauptinterface (siehe Abbildung 41) besteht aus einem Konfigurationsbereich (6) + (7) sowie dem Ausgabebereich (8) – (10). (6) ist hierbei eine combobox, in welcher sich die statische LHS einstellen lässt. Unter (7) kann man die Mindestkonfidenz einstellen, die Visualisierungsmethode für die Ausgabe auswählen, sowie die optionale Multiple-RHS-Analyse hinzuschalten.

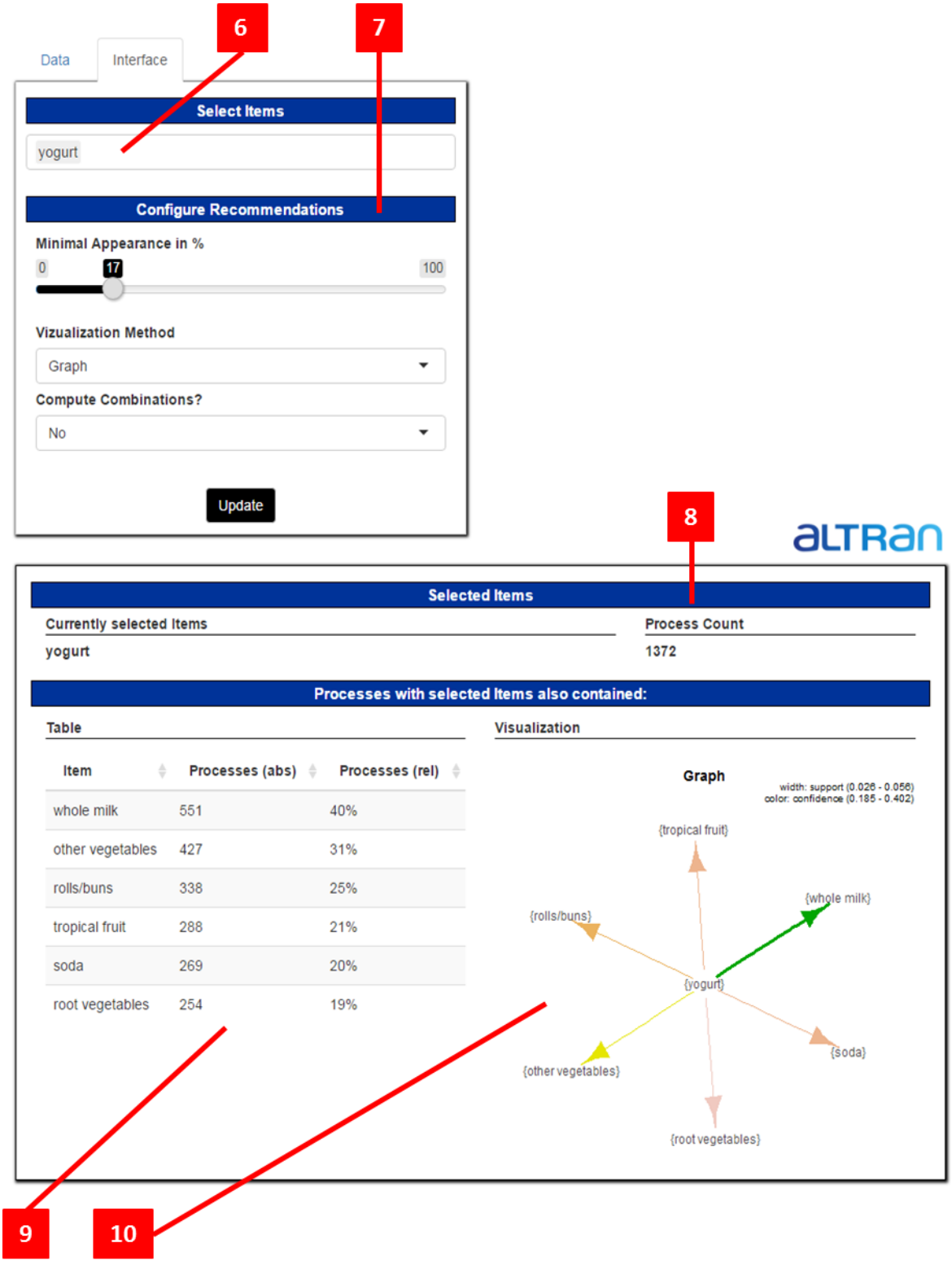


Abbildung 41: Hauptinterface der RE

Der Ausgabebereich enthält eine Statusleiste (8), in welcher die momentane Auswahl an Ausprägungen sowie der entsprechende Support aufgelistet werden. (9) enthält die berechneten Assoziationsregeln. Da die LHS statisch bei allen Regeln gleich ist, wurde diese durch die Statusleiste (8) ersetzt und in der Tabelle (9) werden lediglich die RHS sowie Support und Konfidenz angezeigt. Es ist hierbei zu beachten, dass die Begrifflichkeiten aus der Assoziati-

onsanalyse in allgemein verständliche Texte umgewandelt wurden. (10) enthält schließlich die Visualisierung der berechneten Regeln auf Basis der Nutzerauswahl.

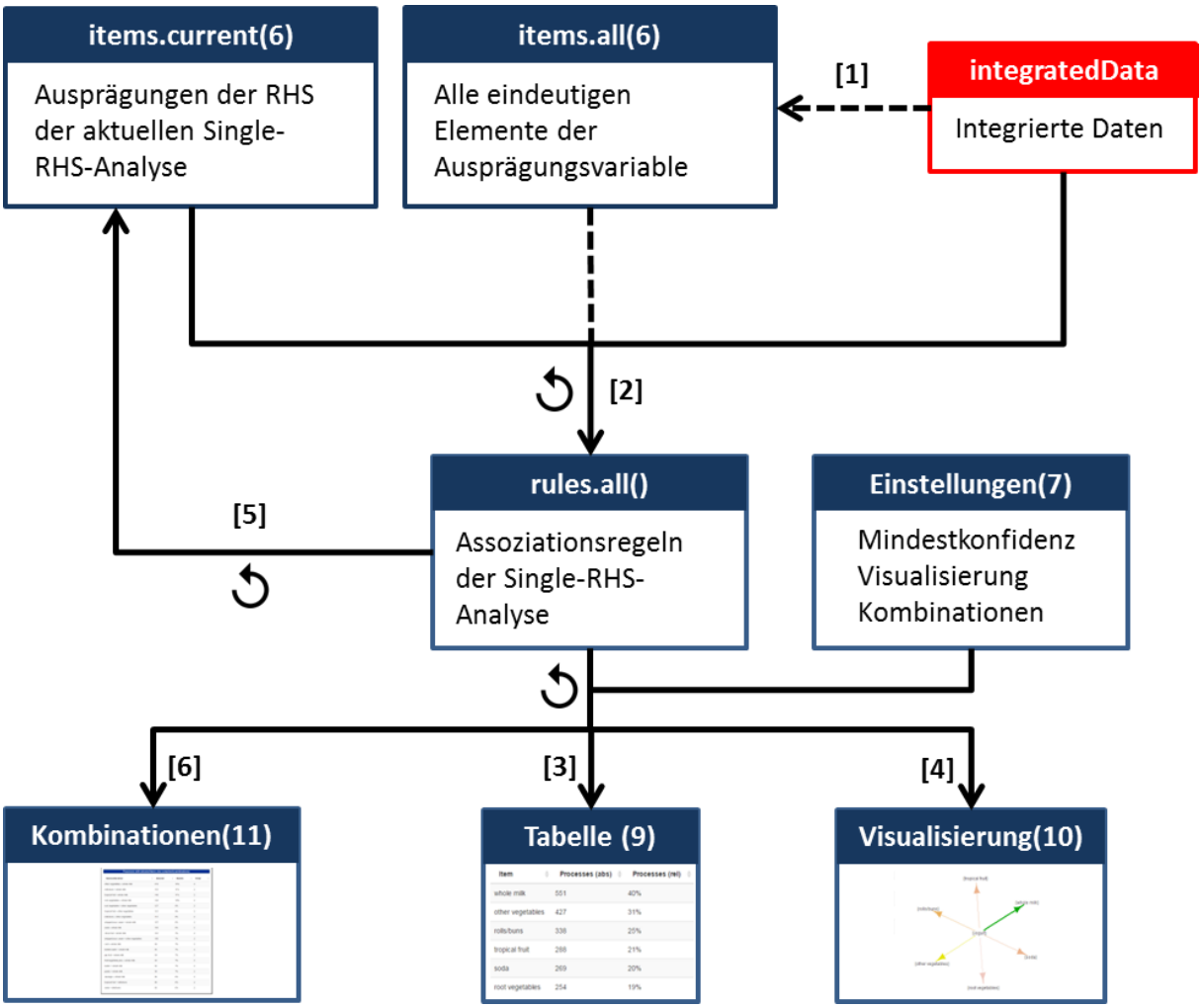


Abbildung 42: Berechnung im Hauptinterface

Der Nutzer erhält initial alle eindeutigen Ausprägungen der Datenbank als Auswahl in der combobox (siehe Abbildung 42).

[1] itemInfo(integratedData())

Nachdem der Nutzer eine Ausprägung ausgewählt und per Aktivierung des Update-Buttons bestätigt hat, wird eine Single-RHS-Analyse (siehe Kapitel 4.1.2) durchgeführt. Hierbei werden die berechneten Regeln abermals in ein reaktives Objekt überführt.

[2] rules.all ← reactive({get.rules.single(input$material,integratedData()})

Die Single-RHS-Analyse wird hierbei mit einem Mindestsupport von 3/Transaktionslänge sowie einer Konfidenz von 0.1 durchgeführt. Die Parameter sind absichtlich derart niedrig ausgelegt, dass alle in dem Datensatz enthaltenen, interessanten Regeln mit einem Analyse-

schritt berechnet werden. Bei Änderungen der Mindestkonfidenz durch den Nutzer werden keine neuen Regeln berechnet, sondern lediglich eine Auswahl auf den einmalig berechneten Regeln vorgenommen. Insofern wird nur bei einer Änderung der Ausprägungen eine Assoziationsanalyse durchgeführt, bei weiteren Einstellungen werden lediglich subsets auf die berechneten Regeln getätigt. Das jeweilige Ergebnis wird dann im Ergebnisteil in Form der gerenderten Tabelle (9) angezeigt.

[3] renderDataTable({rules.all(), subset = rules.all@quality[2]>conf)})

Des Weiteren werden die Regeln mit einer unter (7) ausgewählten Visualisierungsmethode mittels des Packages arulesViz dargestellt (10).

[4] reactive({visualize.rules(rules(),input$vismethod,transactions())})

Im Falle der ausgewählten Graphen-Notation würde in der Funktion visualize.rules() die Visualisierung mittels

```
plot(rules(),
     method="graph",
     shading = "confidence",
     control=list(
                  type="itemsets",
                  main="Graph",
                  edgeCol=terrain.colors(9),
                  arrowSize=2,
                  alpha=1
                )
    )
```

berechnet werden. Unter „method“ können dabei die jeweiligen Visualisierungsmethoden aus Kapitel 2.4 ausgewählt werden, die restlichen Optionen variieren je nach Methode. „Shading“ gibt an, welcher Messfaktor als primäre Messgröße zur Konstruktion verwendet werden soll. Man könnte beispielsweise auch einen Graphen konstruieren, der die Pfeile nach dem jeweiligen Liftwert konstruiert. In der control-list werden grundsätzlich Parameter angegeben, welche die Erscheinung jedoch nicht die Struktur des Graphen verändern. Unter Type wird festgelegt, ob die gesamte Regelbeschriftung oder nur die jeweiligen Itemsets als Knotenbeschriftung verwendet werden sollen. „Main“ ist der Titel, „edgeCol“ die Farbe der Pfeile, „arrowSize“ die Größe der Pfeile und „alpha“ die Transparenz.

Anhand der Visualisierung und der Tabelle kann der Nutzer nun die weiteren Ausprägungen identifizieren, welche mit der ausgewählten Ausprägung assoziieren. In der combobox (1) wird die Auswahl dabei auf die RHS der zuletzt berechneten Regeln reduziert. Die Auswahl wird mittels der bereits vorgestellten Funktion

[5] decode.ItemsFromRules(rules.all(),"rhs")

befüllt. Hierbei ist zu beachten, dass die Auswahl basierend auf den bereits durch Konfidenz reduzierten Regeln vorgenommen wird. Insofern stehen genau die Ausprägungen in (1) zur Wahl, welche in der Tabelle (9) enthalten sind. Somit kann der Nutzer besonders einfach durch eine zusätzlich ausgewählte Ausprägung einen neuen Iterationsschritt starten.
Es kann dabei jederzeit unter (7) die Kombinationsberechnung hinzugeschaltet werden. Hierbei wird eine Multiple-RHS-Analyse (siehe Kapitel 4.1.2) durchgeführt.

[6] rules.combinations ← renderDataTable({get.rules.multiple(input$material, rules.all()})

Die Vorgehensweise ist dabei derer der Single-RHS-Analyse gleich, es wird jedoch aufgrund der potenziell sehr großen Menge an Kombinationen auf eine Visualisierung verzichtet und lediglich eine Tabelle vorgegeben (siehe Abbildung 43) und unterhalb der Ergebnisse der Single-RHS-Analyse anezeigt.

11

Processes with selected Items also contained (combinations):

Itemcombination	Absolut	Relativ	Order
other vegetables + whole milk	219	16%	2
rolls/buns + whole milk	153	11%	2
tropical fruit + whole milk	149	11%	2
root vegetables + whole milk	143	10%	2
root vegetables + other vegetables	127	9%	2
tropical fruit + other vegetables	121	9%	2
rolls/buns + other vegetables	113	8%	2
whipped/sour cream + whole milk	107	8%	2
soda + whole milk	103	8%	2
citrus fruit + whole milk	101	7%	2
whipped/sour cream + other vegetables	100	7%	2
curd + whole milk	99	7%	2
bottled water + whole milk	95	7%	2

Abbildung 43: Abbildung der Kombinationsberechnung in der RE

Hierbei wird zusätzlich zu den Werten der Single-RHS-Analyse die Order angegeben, welche die Anzahl der RHS Ausprägungen darstellt. In der Beispielabbildung ist dies nicht relevant, weil hier aus Gründen der Performanz (siehe Kapitel 4.1.2) lediglich die Kombinationen der Länge 2 berechnet wurden.

4.3.2 Reportmodul

Abbildung 44 enthält einen Überblick über das Interface des Reportmoduls. Dabei sind drei Einstellungen verfügbar (1) – (3), sowie die drei bereits vorgestellten Kennzahlen (4) – (6). In den Einstellungen können die Ausprägungen nach zusätzlichen Informationen strukturiert werden (z.B. Produktsparten) (1) sowie nach verschiedenen Zeiträumen selektiert werden (2). Außerdem kann man eine detaillierte und eine einfache Version unterscheiden (3). (4) enthält die Systemabdeckung, (5) die Regelbestandsveränderung und (6) den Regelbestand.

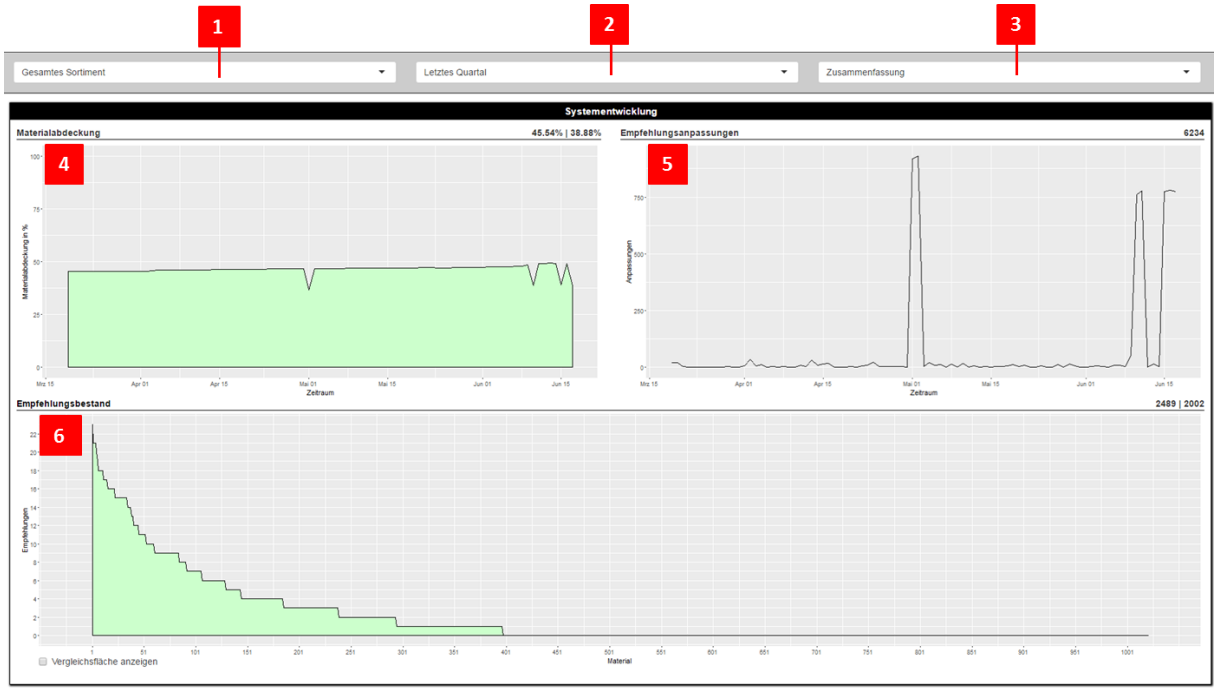


Abbildung 44: Interface des Reportmoduls

Die Informationen werden aus den Logdateien des in dem vergangenen Zeitraum regelmäßig ausgeführten Mess-Skriptes zusammengetragen (siehe Abbildung 45). Dazu wird eine Schleifenkonstruktion ausgeführt, welche in jedem Durchlauf eine Logdatei ausliest und die zeitraumbezogenen Kennzahlen Materialabdeckung und Regelbestandsänderung berechnet (der Regelbestand ist zeitpunktbezogen auf das aktuelle Datum). Die Schleifenvariable enthält dabei das aktuell zu analysierende Datum und wird an jedem Durchlaufsende inkrementiert.

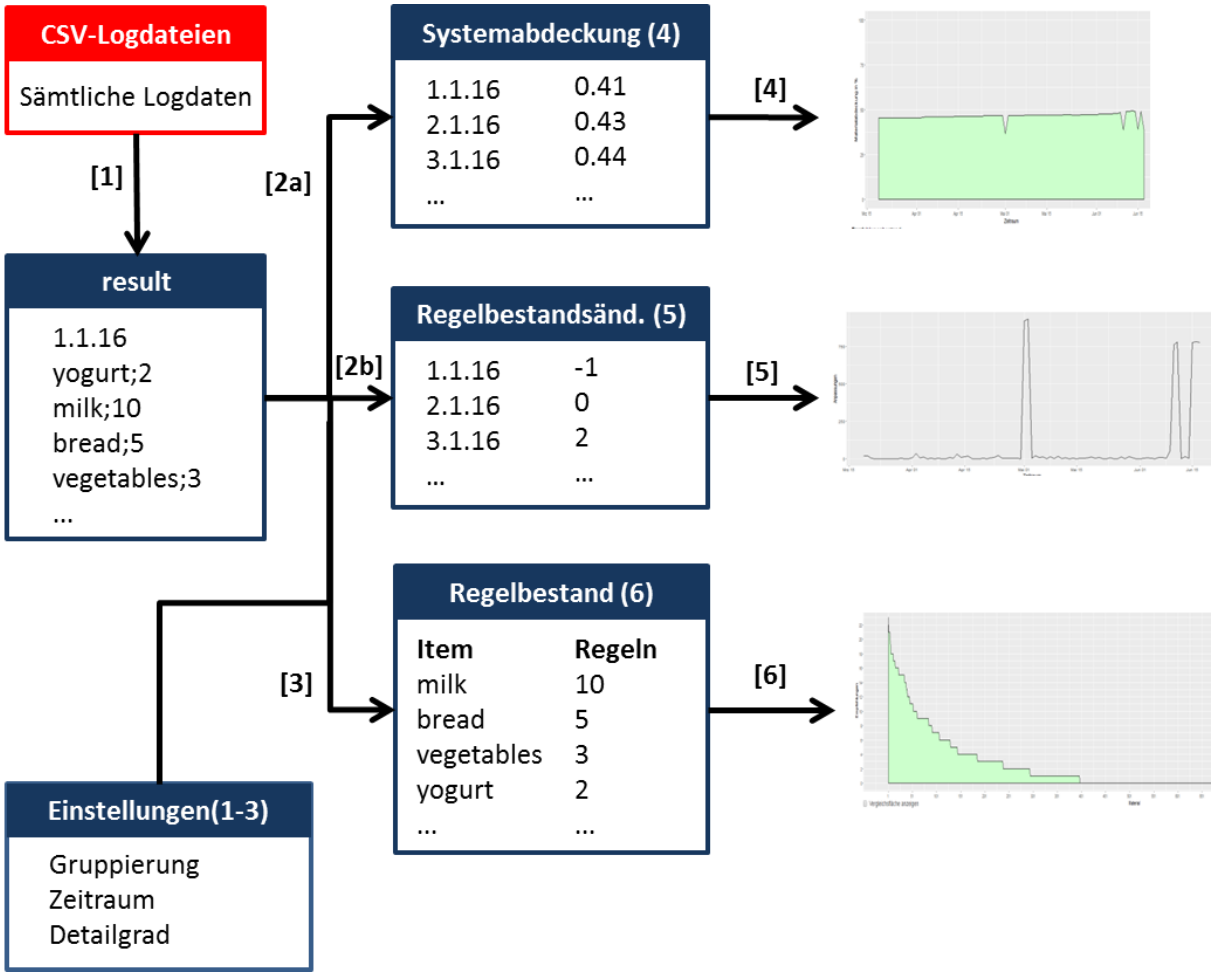


Abbildung 45: Berechnungen im Reportmodul

Es wird zuerst der Dateiname mit dem aktuellen Datum zusammengesetzt und anschließend die entsprechende Logdatei eingelesen.

```
[1] filename <- paste0("logs_result//logfile_",datum,".csv");
    result <- read.csv(filename,sep=";");
```

Es wird die Systemabdeckung berechnet, indem zuerst die Anzahl aller Ausprägungen festgestellt wird, welche mindestens eine Regel haben und dieser Wert anschließend durch die Anzahl aller Ausprägungen dividiert wird.

```
[2a] hasrules <-length(result[result[,2]>0,]);

allrules <- length(result[,2]);

systemabdeckung <- round((hasrules/allrules)*100);
```

Im Anschluss wird die Regelbestandsänderung berechnet, indem die Regeln des unmittelbar vergangenen Logfiles von denen des aktuellen Logfiles subtrahiert werden. Die jeweiligen Differenzen werden in Betrag gesetzt und summiert.

[2b] vec.difftmp <- result[,2]-result_old[,2];

vec.difftmp <- abs(vec.difftmp[!(vec.difftmp==0)]);

regelbestandsänderung <- sum(vec.difftmp);

Der Regelbestand wird zeitpunktbezogen nach der Schleife aus dem aktuellen Logfile bezogen und absteigend sortiert.

[3] sort(result[,2],decreasing=TRUE)

Dann kann bereits die Visualisierung vorgenommen werden. Alle Visualisierungen des Reportmoduls werden mit dem ggplot2 Package bewerkstelligt. Mit nachfolgendem Befehl wird die Systemabdeckung visualisiert.

[4] ggplot(data = stats, aes(x=datum, y=sa))

+ geom_area(position = 'stack',colour="#000",bg="#ccffcc")

+ xlab("Zeitraum")

+ ylab("Systemabdeckung in %")

+ ylim(0,100)

Neben den anzugebenden Daten wird mit geom_area erwirkt, dass die Daten in einer Fäche dargestellt werden (mit der Position im Vordergrund, einer schwarzen Kontur sowie grüner Farbe). „xlab“ und „ylab“ stehen für die Beschriftungen der Achsen und mit „ylim“ wird die Skalierung der Ordinate. Die Regelbestandsänderungen werden wiefolgt visualisiert.

[5] ggplot(data = stats,aes(x=datum, y=adjusts))

+ geom_line()

+ xlab("Zeitraum")

+ ylab("Anpassungen")

Der einzige Unterschied besteht hier in der Form, denn geom_line erzeugt eine Linie statt Fläche. Hierbei müssen keine Parameter angegeben werden, weil die Standardwerte in Form einer schwarzen Linie passend sind. Der Regelbestand wird abschließend visualisiert.

```
[6] ggplot(data = value, aes(x=mat, y=empCurrent))
+ geom_area(position="stack",bg="#ccffcc",color="#000")
+ xlab("Material")
+ ylab("Empfehlungen")
+ scale_x_continuous(breaks = round(seq(min(value$mat), max(value$mat), by = 50),0))
+ scale_y_continuous(breaks = round(seq(min(value$empCurrent),
  max(value$empCurrent), by = 2),0))
```

Neben den bereits bekannten Parametern sind hier „scale_x_continous“ sowie scale_y_continous“ interessant, welche eine äußerst flexible Skalierung der Achsen ermöglicht. Hierbei werden der Minimal- und Maximalwert angegeben sowie die Größe der Intervalle.
Die Steuerung der Einstellungen unter (1) – (3) bedingt subsets auf die Daten der Logdateien. Das Auswählen der Sparten bedingt subsets auf die Ausprägungsvariable und das Ändern des Zeitraums grenzt die Auswahl der Logdateien anhand des Datums ein.

5. Evaluierung

In diesem Kapitel wird der Prototyp hinsichtlich dreier Faktoren evaluiert. In 5.1 und 5.2 werden die beiden maßgeblichen Eigenschaften beleuchtet, welche den Prototyp von der standardmäßigen Vorgehensweise bei der Berechnung von Assoziationsregeln unterscheiden (siehe Kapitel 3.1). In 5.1 liegt der Fokus auf der generischen Verwendung hinsichtlich des schnellen und einfachen Einbindens von zahlreichen Datensätzen. In 5.2 wird die Performance des Prototyps evaluiert, indem die durchschnittliche Berechnungsdauer von Items in verschiedenen Datensätzen gemessen wird. 5.3 enthält schließlich die rückblickende Überprüfung, welche Phasen des Crisp-DM Modells durch den Protoypen automatisiert werden können, um REs für eine breite Masse an Kunden durch die Verknüpfung der generischen Eigenschaft in Kombination mit der Echtzeitanalyse und einem teilautomatisierten Crisp-DM Modell greifbar zu machen.

5.1 Datenintegration

Um die generische Verwendung des Prototyps aufzuzeigen, werden nachfolgend drei Datensätze vorgestellt, welche aus verschiedenen Anwendungsbereichen entstammen und außerdem unterschiedliche Datenstrukturen aufweisen. Die Datensätze werden jeweils kurz hinsichtlich des Anwendungsgebietes vorgestellt und anschließend die Parametrisierung zum Einbinden in den Prototyp aufgezeigt.

Groceries

Der Groceries Datensatz ist frei verfügbar und aufgrund seines Anwendungsgebietes als Standard in der Entwicklung bezüglich Assoziationsanalysen eingesetzt. Dieser ist deshalb auch als Standard im Protoytpen zum exemplarischen aufzeigen der Funktionalitäten im Prototyp eingebunden.

```
1  citrus fruit,semi-finished bread,margarine,ready soups
2  tropical fruit,yogurt,coffee
3  whole milk
4  pip fruit,yogurt,cream cheese ,meat spreads
5  other vegetables,whole milk,condensed milk,long life bakery product
6  whole milk,butter,yogurt,rice,abrasive cleaner
7  rolls/buns
8  other vegetables,UHT-milk,rolls/buns,bottled beer,liquor (appetizer)
9  pot plants
10 whole milk,cereals
11 tropical fruit,other vegetables,white bread,bottled water,chocolate
12 citrus fruit,tropical fruit,whole milk,butter,curd,yogurt,flour,bottled water,di
13 beef
14 frankfurter,rolls/buns,soda
15 chicken,tropical fruit
```

Abbildung 46: Ausschnitt aus dem Groceries Datensatz

Der Datensatz entstammt aus dem Bereich der Warenkorbanalyse. Es wurde über einen gewissen Zeitraum alle Bestellungen von 20 Supermärkten aufgezeichnet. Der Datensatz enthält lediglich eine Variable, welche eine beliebige Anzahl an Produkten enthalten kann. Jede Zeile stellt hierbei eine Bestellung dar. Insgesamt enthält der Datensatz 9835 Zeilen (= Bestellungen), in welchen 169 eindeutige Produkte verteilt sind.
Zum Einbinden in den Prototyp wird hierbei das horizontale Format unter Verwendung des Zeilenindex als Prozess-ID verwendet. Hierbei sind keine vorherigen Datenumformungen notwendig. Somit kann man im Prototyp Assoziationen zwischen den Produkten und deren Zusammensetzungen innerhalb der Bestellungen aufdecken.

Ersatzteile

Dieser Datensatz wurde innerhalb eines Unternehmens aufgezeichnet, in welchem zu Geräten Ersatzteile bestellt wurden. Der Datensatz ist anonymisiert und enthält lediglich zwei Variablen: eine Transactions-ID sowie eine Material-ID.

```
id;Material
1;16715
2;56684
3;35499
4;61678
5;68178
6;61684
7;47961
7;32617
8;32857
9;36487
10;34877
11;35971
12;34178
13;51977
```

Abbildung 47: Ausschnitt des Ersatzteil Datensatzes (anonymisiert durch geänderte IDs)

Der Datensatz enthält 9000 Transaktionen, in welchen 1021 eindeutige Materialien verteilt sind. Zur Einbindung wurde der Datensatz vertikal parametrisiert unter der Zuordnung der Transaktions-ID als Process-ID und der Material-ID als Ausprägungsvariable. Somit können im Prototyp Material-Sets erschlossen werden, basierend auf den Assoziationen zwischen Materialien in den Transaktionen.

Zoo

Der Datensatz enthält die Eigenschaften von Tieren eines Zoos. Zu jedem Tier existieren 18 Eigenschaften wie bspw. Größe oder Lebensraum. Die Eigenschaften wurden für insgesamt 101 Tiere aufgezeichnet.

```
"hair","milk","predator","toothed","backbone","breathes","legs","catsize"
"hair","milk","toothed","backbone","breathes","legs","tail","catsize"
"eggs","aquatic","predator","toothed","backbone","fins","tail"
"hair","milk","predator","toothed","backbone","breathes","legs","catsize"
"hair","milk","predator","toothed","backbone","breathes","legs","tail","catsize"
"hair","milk","toothed","backbone","breathes","legs","tail","catsize"
"hair","milk","toothed","backbone","breathes","legs","tail","domestic","catsize"
"eggs","aquatic","toothed","backbone","fins","tail","domestic"
"eggs","aquatic","predator","toothed","backbone","fins","tail"
"hair","milk","toothed","backbone","breathes","legs","domestic"
"hair","milk","predator","toothed","backbone","breathes","legs","tail","catsize"
"feathers","eggs","airborne","backbone","breathes","legs","tail","domestic"
"eggs","aquatic","predator","toothed","backbone","fins","tail"
"eggs","predator"
"eggs","aquatic","predator","legs"
```

Abbildung 48: Ausschnitt aus dem Zoo Datensatz

Dieser Datensatz kann auf die gleiche Art und Weise eingebunden werden, wie der Groceries Datensatz: horizontal unter Verwendung des Index als Process ID. Somit können Assoziationen zwischen Eigenschaften gefunden werden.

5.2 Performanz

In diesem Kapitel wird die Berechnungsdauer der Assoziationsregeln innerhalb des Prototyps gemessen. Die Messungen wurden mit einem Intel i5-3500u durchgeführt, welcher zwei Physische Kerne mit jeweils 2.3Ghz, aufgeteilt auf jeweils zwei Threads enthält. Es wurde außerdem mit der 64bit Variante des R Clients 3.2.3 mit einem maximal verfügbaren Arbeitsspeicher von 16 Gb gearbeitet.

In Tabelle 13 sind die durchschnittlichen Berechnungszeiten der einzelnen Elemente der im vorherigen Kapitel vorgestellten Datensätze aufgeführt. Diese beziehen sich auf die reine Regelberechnung mittel der Funktion „get.rules.single“. Alle weiteren Interfaceoperationen werden vor dem Hintergrund des individuellen Einsatzes nicht gemessen. Die Datenintegration der CSV Datei wird an dieser Stelle ebenfalls nicht gemessen, da für die Echtzeitberechnung eine Anbindung per Datenbank/Stream vorgenommen werden muss.

Innerhalb der getesteten Datensätze konnte ein gutes Ergebnis erzielt werden. Die durchschnittlichen Berechnungszeiten liegen deutlich unter einer Sekunde. Jedoch beinhalten die Datensätze lediglich zwischen 100 und 10.000 Transaktionen. Daher sind zur praxisorientieren Feststellung der Performanz weitere Tests mit größeren Datensätzen notwendig.

Tabelle 13: Performanz der Einzelberechnungen

Datensatz	Anzahl Transaktionen	Ø Proc.time	Max Proc.time	Min Proc.time
Groceries	9835	0,0269	0,2	0
Ersatzteile	9000	0,0157	0,2	0
Zoo	101	0,0258	0,2	0

Zur Messung der Analysedauer wurde der vorliegende Ersatzteildatensatz mehrmals vervielfältigt. Es wurden vier unterschiedliche Datensätze generiert, welche durch geeignete Manipulationen verschiedene Faktoren wie Transaktionsgröße, Produktanzahl und Bestellzyklen und deren Auswirkung auf die Performanz messbar machen. Nachfolgend werden die Manipulationen aufgezeigt.

- **Datensatz 1 (original)**: Vervielfältigung auf vier Millionen Transaktionen ohne Manipulation.
- **Datensatz 2 (Transaktionsgröße)**: Zu 3000 zufällig ausgewählten Transaktionen des Originaldatensatzes wurden jeweils fünf zufällig gewählte Produkte hinzugefügt, um die durchschnittliche Transaktionsgröße des Datensatzes zu erhöhen. Die manipulierten Transaktionen wurden dann vervielfältigt.
- **Datensatz 3 (Produktanzahl)**: Die Anzahl der Produkte wurde verdoppelt, indem der Orginialdatensatz einmal vervielfältigt und die Produktnamen der vervielfältigten Hälfte durch Hinzufügen einer zusätzlichen Zahl „1“ verändert wurden. Der somit verdoppelte Datensatz wurde dann auf vier Millionen Transaktionen hochgerechnet.
- **Datensatz 4 (Produktposition)**: Es wurden 3000 Transaktionen zufällig dem orginialen Datensatz entzogen und am Ende hinzugefügt, um die Positionen innerhalb des Datensatzes zu verändern. Dies soll annäherungsweise den Einfluss der Position in der Datenbank messen, was bspw. durch veränderte Bestellzyklen auftreten kann. Die manipulierten Transaktionen wurden dann vervielfältigt.

Nach den Datensimulationen ergeben sich folgende Konstellationen (siehe Tabelle 14).

Tabelle 14: Merkmale der simulierten Datensätze zur Performanzmessung

Datensatz	Anzahl Transaktionen	Anzahl Ausprägungen	Ø Transaktionsgröße
1	4.000.000	1021	1,6
2	4.000.000	1021	**2,6**
3	4.000.000	**2042**	1,6
4	4.000.000	1021	1,6

Die Berechnungszeiten (siehe Tabelle 15) betragen bei vier Millionen Transaktionen zwischen fünf und sieben Sekunden. Die jeweiligen Varianzen liegen vergleichsweise niedrig, sodass die Durchschnittszahlen aussagekräftig sind. Eine Veränderung der Produktanzahl und den Positionen der Bestellungen innerhalb des Datensatzes hat keine deutliche Veränderung der Berechnungszeiten verursacht. Bei dem Datensatz mit erhöhter Transaktionsgröße wurde eine Erhöhung der durchschnittlichen Berechnungsdauer von ca. zwei Sekunden verzeichnet.

Vor dem Hintergrund der Echtzeitberechnung ist die Performanz bei größeren Datensätzen somit noch verbesserungswürdig. Ansätze zur Verbesserung der Performanz können durch Optimierungen der Berechnungsfunktion, Interface oder R Client Auswahl vorgenommen werden.

Tabelle 15: Berechnungszeiten der simulierten Datensätze

Datensatz	Ø Proc.time	Var Proc.time	Min Proc.time	Max Proc.time
1	5,40	0,0026	5,33	5,74
2	**7,60**	0,0130	7,47	8,87
3	5,92	0,0061	5,83	7,23
4	5,57	0,1377	5,43	5,82

Grundsätzlich wäre eine Verbesserung der Analysefunktion sinnvoll. R eignet sich nur bedingt für effiziente Analysen von großen Datenmengen. Im Prototyp wird die Regelberechnung hauptsächlich mittels des arules Packages vorgenommen. Die Apriori-Funktion wird hierbei zum Zwecke der Performanz bereits in C durchgeführt. Zur Verbesserung der Performanz müsste demzufolge eine gänzlich neue Funktion innerhalb von C geschrieben werden, welche einen effizienteren Algorithmus beinhaltet. Dies wäre nach eigener Abschätzung die effizienteste, jedoch auch aufwändigste Methode zur Minimierung der Berechnungszeiten.
Die Performanz kann weiterhin durch eine Optimierung des Interfaces verbessert werden. Momentan wird eine Regelberechnung immer dann vorgenommen, wenn der Nutzer ein Item in der LHS verändert. Hierbei werden lediglich die Regeln der aktuellen LHS berechnet. Da der Nutzer im Sinne der explorativen Nutzung allerdings potenziell einige Zeit auf die Begutachtung der Regeln verwendet, würde eine Vorausberechnung der Regeln für die aktuell berechneten RHS während der Begutachtungszeit Sinn machen. Vor dem Hintergrund, dass der Nutzer zu Beginn eine Ausprägung als Start benötigt, liegt es nahe, dass ein Wechsel der Startausprägung im laufenden Betrieb unwahrscheinlich ist. Die Vorausberechnung des Regelbaumes der Startausprägung könnte somit die Wartezeiten bei der Berechnung minimieren.
Es ist außerdem eine Verbesserung durch einen Wechsel des R Clients möglich. Die Berechnungen wurde mit der R Base Version 3.2.3 vorgenommen. Diese unterstützt kein Multithreading, sodass die Berechnungen nur mit einem Thread durchgeführt werden. Es existieren jedoch bereits weitere Versionen von R, welche Multithreading unterstützen, sodass ein Wechsel auf eine andere R Version unter Anpassungen des Funktionsablaufes Sinn macht. Dies würde vor allem in Kombination mit einer neu zu entwickelnden Analysefunktion durch Ausrichtung der Funktionsschritte auf Multithread-Berechnungen zur Minimierung der Berechnungszeiten führen.

5.3 Programmieransatz

An dieser Stelle wird der Programmieransatz bei der Entwicklung des Prototyps, Erstellungen von REs zu vereinfachen und einer breiten Masse an Kunden zugänglich zu machen, evaluiert. Durch den Prototyp werden hinsichtlich des Crisp-DM Modells vor allem Aktivitäten der Phasen Data Preparation, Modeling sowie Deployment automatisiert.

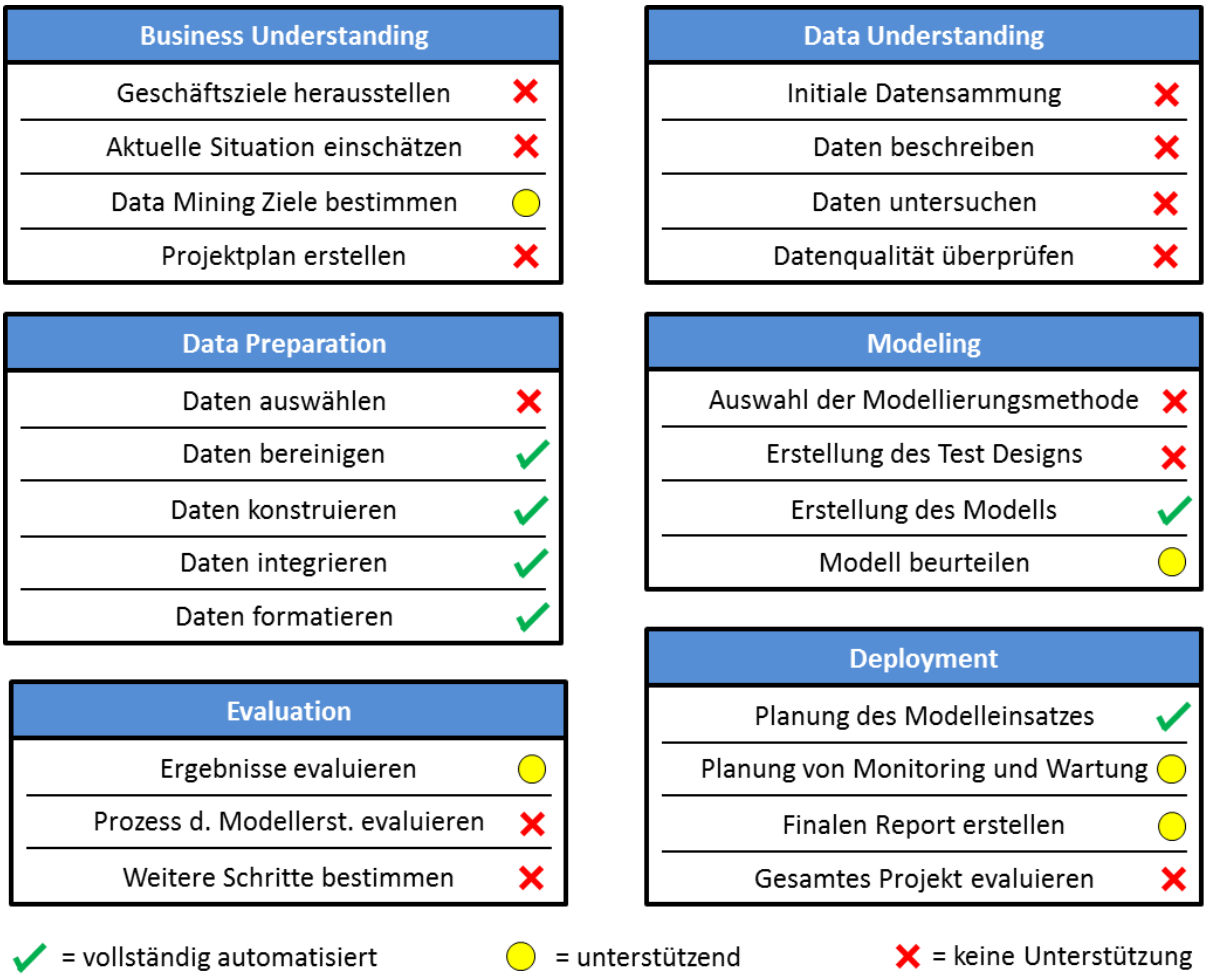


Abbildung 49: Teilautomatisierung des Crisp-DM Modells

Die Business Understanding Phase muss auch bei Nutzung des Prototyps weitestgehend manuell durchgeführt werden. Lediglich im Bereich der Zielbestimmung können erste Entwürfe einer RE genutzt werden, um die Geeignetheit der Daten abzuschätzen. Die Data Understanding Phase obliegt weiterhin dem Nutzer, zwar können Daten auch im Prototyp innerhalb einer Tabelle dargestellt werden, jedoch nur CSV-Dateien und ohne weitergehende Funktionen hinsichtlich der Datenuntersuchung. Die Data Preparation Phase wird vollständig vom Prototyp übernommen, nachdem der Nutzer die entsprechenden Daten selektiert hat. In der Phase Modeling wird die hauptsächliche Modellerstellung durch den Prototyp vorgenommen. Die Modellbeurteilung kann mit Hilfe des Report Moduls unterstützt werden. Eine Auswahl

der Modellierungsmethode oder Erstellung eines Test-Designs wird nicht unterstützt. In der Phase Evaluation können die Ergebnisse mit Hilfe des Report Moduls evaluiert und außerdem weitere Schritte geplant werden. In der letzten Phase Deployment kann die Phase der Modelleinsatzplanung durch die bereits erstellte RE ersetzt werden. Monitoring, Wartung sowie Erstellung des finalen Moduls können durch das Report Modul unterstützt werden.

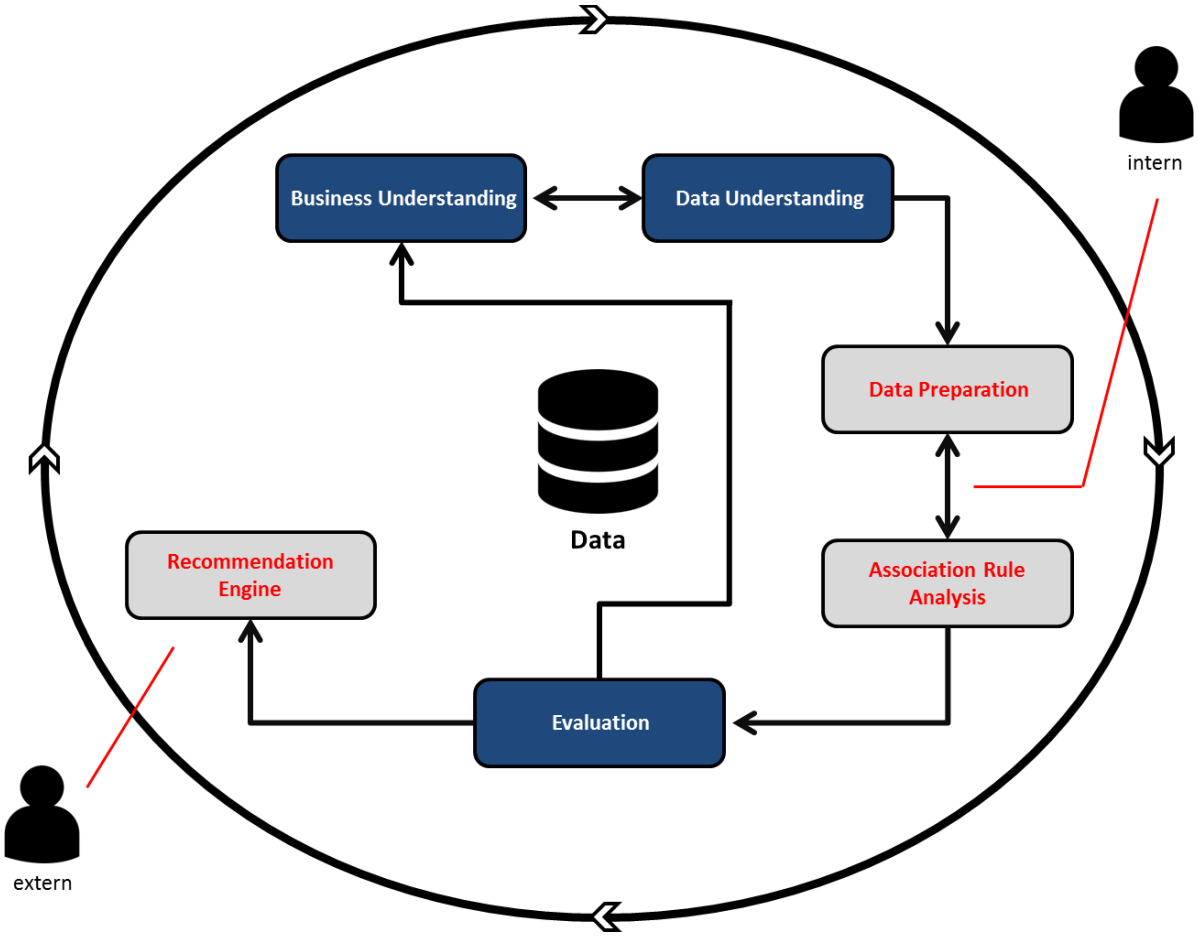


Abbildung 50: Modifiziertes Crisp-DM Modell unter Verwendung des Prototyps

Innerhalb von Crisp-DM Projekten sind durch die Automatisierungen zwei maßgebliche Verwendungen des Prototyps möglich. Zum einen können Analysten den Prototyp intern dazu nutzen, vorliegende Daten zur Eignung für eine RE zu testen. Durch die generische Funktionalität können verschiedene Variablenkonstellationen schnell eingebunden, auf Eignung überprüft und dem Kunden ggfs. ein frühzeitiger Einblick gegeben werden. Dies beschleunigt insofern nicht nur den Projektdurchlauf sondern verbessert auch die Planbarkeit des Projektes. Denn bezüglich der zweiten Verwendungsmöglichkeit ist der externe Einsatz des Prototyps bei dem Kunden gemeint. Durch die dynamischen Integrationsmöglichkeiten von R kann der Prototyp im Gesamten oder lediglich einzelne Funktionen in die Geschäftsprozesse des Kunden eingebunden werden. Bei einer somit beschleunigten Feststellung der Eignung von Daten für eine RE und einem frühzeitigem Aufzeigen von Ergebnissen für den Kunden, kann ein

Data Mining Projekt schließlich hinsichtlich Zeit, Kosten und Planbarkeit optimiert werden. Dies wird letztlich durch die geschaffene Möglichkeit der Überführung von Best Practices innerhalb der Automatisierungen unterstützt.

6. Fazit

Die Entwicklung des Prototyps kann insgesamt als erfolgreich bewertet werden. Die finalen Funktionalitäten decken alle zuvor aufgestellten Anforderungen ab und das übergeordnete Ziel der Teilautomatisierung von Data Mining Projekten bezüglich REs zur Minimierung von Zeit- und Kostenaufwand kann mittels des Prototyps vollzogen werden. Während des Entwicklungsprozesses sind einige Sachverhalte aufgefallen, welche sich für weitergehende Forschungen anbieten und nachfolgend beleuchtet werden. Wie der erstellte Prototyp in der Praxis eingesetzt werden kann, wird weiterführend präsentiert und abschließend werden Empfehlungen für Weiterentwicklungen getroffen.

6.1 Implikationen für die Forschung

Anhand der Evaluierung des Prototyps kann festgestellt werden, dass der in dieser Arbeit angewendete, alternative Programmieransatz sinnvoll war. Denn mit dem Prototyp kann in kürzester Zeit anhand verschiedener Daten und Variablenselektionen durch die einfache Datenintegration und Echtzeitberechnung geprüft werden, ob eine RE für den jeweiligen Kunden anwendbar ist. Innerhalb des klassischen Crisp-DM Modells unter Verwendung der vorherrschenden, auf allgemeine Analysen ausgelegte Software, wäre dieser Vorgang erheblich aufwändiger. Somit erschließt sich ein Einsparungspotenzial durch Automatisierungen von Data Mining Lösungen. Dabei geht es nicht um eine gänzliche Ersetzung von allgemeinen Softwarelösungen, sondern um zusätzliche Werkzeuge, die eine Reihe an standardmäßigen Data Mining Lösungen schnell und einfach hinsichtlich der Geeignetheit der Daten testet. Auf diese Weise könnten erheblich Kosten und Zeit eingespart werden. Die Anwendung dieses Ansatzes könnte insofern Data Mining Projekte für eine breite Masse an Kunden finanzierbar machen. Vor allem bei kleinen und mittleren Unternehmen wäre ein schneller Überblick über die Geeignetheit der Daten für Standardlösungen des Data Mining interessant. Daher wird an dieser Stelle empfohlen, weitere Automatisierungen im Bereich von häufig verwendeten und herunterbrechbaren Lösungen vorzunehmen, um den Data Minig Sektor einer breiteren Zielgruppe zugänglich zu machen.

Des Weiteren sollte nach eigener Auffassung der Assoziationsanalyse von Regeln mit mehreren Items in der RHS mehr Aufmerksamkeit zugesprochen werden. Die in der aktuellen Forschung vorherrschende Meinung, dass der Mehrwert solcher Regeln nicht mit dem immensen Aufwand der Berechnung im Einklang steht, wird hierbei nicht in Frage gestellt. Es ist jedoch durchaus realistisch, den Sektor der zu berechnenden Regeln einzugrenzen. Das dies sinnvoll umgesetzt werden kann, ist Kapitel 4.1.2 zu entnehmen. Dabei wurden mehrere Alternativen

des Herunterbrechens der Kombinationsanazahlen vorgestellt, welche grundsätzlich auch in anderen Assoziationsanalysen angewendet werden können. Der Vorteil von solchen Assoziationsanalysen dürfte im Praxiseinsatz vor allem der Faktor Zeit sein, da man aufwändige, iterative Berechnungen spart. Außerdem wird das Verständnis über die Assoziationen erhöht, denn in einigen Fällen lösen Assoziationsregeln mit einem Element in der RHS möglicherweise ein falsches Verständnis aus.

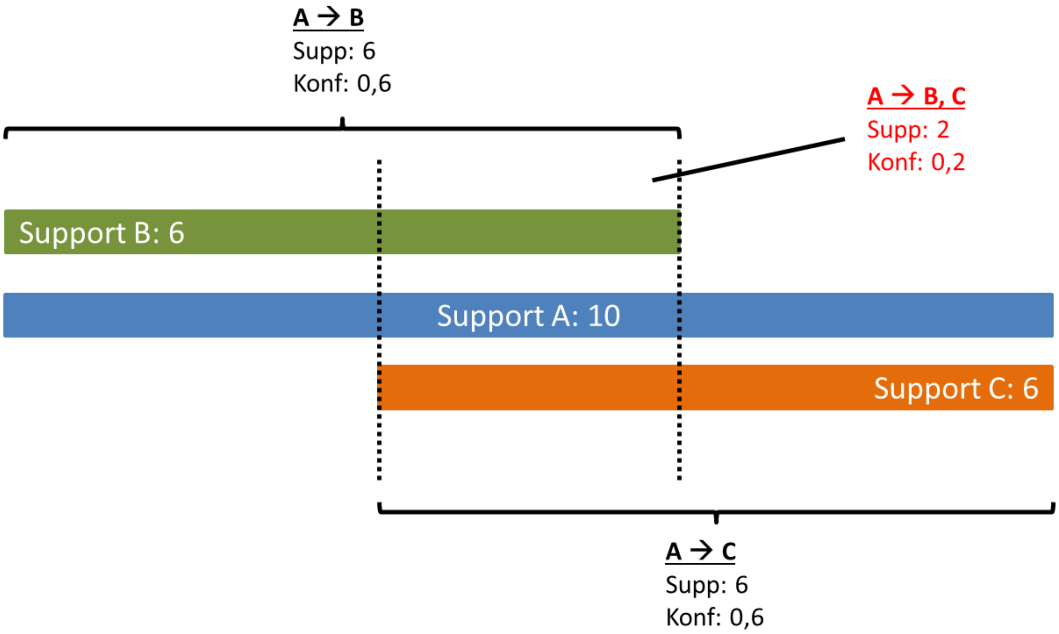


Abbildung 51: Häufig aufgetretener Interpretationsfehler bei Assoziationsregeln mit einem Element in der RHS

Wenn ein Nutzer zwei Regeln A→B und A→C mit jeweils gleichen Support- und Konfidenzwerten erhält, wird möglicherweise das Verständnis vermittelt, dass beide Ausprägungen B und C auf der gleichen Assoziation zu A beruhen und gleichermaßen in ein Set eingeordnet werden. Abbildung 51 zeigt jedoch einen Fall auf, welcher im Prototyp häufig vorkam und bei welchem die getroffene Interpretation falsch ist. Daher und zur Optimierung der schnellen Bedienung von REs wird empfohlen, Assoziationsregeln mit mehreren Elementen in der RHS durch geeignete Eingrenzungen der Kombinationsmöglichkeiten in der Forschung einen höheren Stellenwert zuzuschreiben.

6.2 Implikationen für die Praxis

Der vordergründliche und bereits mehrfach angesprochene Verwendungsansatz des Prototyps ist das schnelle Integrieren von Kundendaten und durch die Echtzeitanalyse ermöglichte ad hoc Präsentation einer RE zur Optimierung von Data Mining Projekten. Mittels des Mess-Skriptes und dem Reportmodul kann ein initialer Überlick für den Analysten gegeben werden und mittels der RE kann dem Kunden ein schneller Einblick in die Funktionsweise gegeben

werden. Dies soll in der Anfangsphase dabei helfen, REs als potenzielles Ziel abzuschätzen. Sofern dabei die Entscheidung für den Einsatz einer RE getroffen wird, kommen mehrere Einsatzszenarien der generischen Funktionsbibliothek des Prototyps in Frage.

Der naheliegendste Einsatz ist die Verwendung der RE des Prototyps durch deren Anpassung und Einbettung in die Systemarchitektur des Kunden. Hierbei spielt die weite Verbreitung von R und dadurch geschaffene Integrationsmöglichkeiten eine große Rolle, denn R bietet hierfür zahlreiche Schnittstellen. Denkbar wäre insofern, dass Nutzer innerhalb des Kundensystems die RE mit aktuell ausgewählten Ausprägungen aufrufen. Anschließend können dann Ausprägungssets erstellt und nach Beendigung der RE in das laufende System des Kunden zurückgeführt werden. Hierbei können mit dem Prototyp je nach Motiv sowohl die iterative als auch kombinatorische Vorgehensweise gewählt werden.

Ein weiteres Verwendungsszenario wäre der Einsatz der Funktionsbibliothek als Hintergrundprozess ohne primäre Einbindung des Nutzers. Konkret kann dabei die Funktionsbibliothek bei Betätigen eines Buttons zum Abschließen eines Prozesses eingebunden werden, welche dann durch eine Assoziationsanalyse prüft, ob zu dem Ausprägungsset weitere Elemente passen. Sofern vorhanden, würden diese Elemente entsprechend für den Nutzer angezeigt. Dieser Einsatz hätte den Vorteil, dass der Nutzer keinerlei Kenntnisse über das Assoziationsverfahren oder die RE besitzen muss. Auch ist die Integration der Funktionsbibliothek als Hintergrundprozess wenig aufwändig und die Performanz der Single-RHS-Analyse reicht für einen reibungslosen Vorgang aus. Daher wird dieser Verwendung nach eigener Auffassung das größte Potenzial zugeschrieben.

Unabhängig von dem gewählten Verwendungsansatz der RE kann mit dem Mess-Skript sowie dem Reportmodul die im Einsatz befindliche RE überwacht werden. Der Entscheidungsträger des Kunden kann im Reportmodul durch einfach verständliche Visualisierungen und Kennzahlen die Entwicklung der RE nachvollziehen. Anhand des Regelbestandes können hier außerdem Ziele bezüglich der Verwendung der RE aufgestellt und evaluiert werden. Somit kann dem Entscheidungsträger der Nutzen der RE aufgezeigt werden.

6.3 Implikationen für die Weiterentwicklung

Die Ausweitung der Datenintegration stellt die nach eigener Auffassung wichtigste Erweiterung des Prototyps dar. Hinsichtlich der Echtzeitberechnung wären hier vor allem Datenbankanbindungen sowie Streams sinnvoll. Diesbezüglich existieren bereits R Packages mit entsprechenden Funktionalitäten, auf den aufgebaut werden kann.

Um die generische Anwendbarkeit bezüglich Datenintegration zu beschleunigen, wäre eine entsprechende Optimierung der Performanz bei der Dateneinbindung zielführend. Erste Ver-

suche durch Vorab-Kompilierungen in Byte Code haben keine signifikante Verbesserung der Performance bewerkstelligt. Die Funktionen der Dateneinbindung wurden außerdem bereits hinsichtlich einer minimalen Memory Allokation optimiert.

Die Performanz der Regelberechnung sollte ebenfalls verbessert werden. Entsprechende Empfehlungen sind dem Kapitel 5.2 zu entnehmen.

Hinsichtlich der Analysemethoden wäre vor allem das Motiv der Reduzierung von Ausprägungssets im Sinne des Post-Pruning für Weiterentwicklungen zu empfehlen. Diesbezüglich enthält der Prototyp keine spezifischen Methoden, da ein effizientes Hinzufügen durch die iterative Erstellung der Ausprägungssets Ineffizienzen vorbeugt. Wenn die Funktionsbibliothek jedoch als Hintergrundprozess am Ende eines ineffizienten Prozesses eingesetzt wird, wären Methoden zur Reduzierung sinnvoll. Hier wäre nach eigener Auffassung eine Implementation von Messwerten aus den Visualisierungsmethoden zielführend. Eine Berechnung des „Item-Utility-Index“ aus der Parallelkoordinatennotation oder der „Differences of Confidences“ aus Mosaikmustern können hierbei zur Reduzierung herangezogen werden. Darüber hinaus wäre eine Möglichkeit zur Einbindung von bereits aufgezeichneten Fehlentwicklungen wie bspw. Rücksendungen sinnvoll. Da dies jedoch eine entsprechende Datenlage beim Kunden voraussetzt, wäre dies den vorherigen Implementationsempfehlungen unterzuordnen.

Die Erweiterung des Mess-Skriptes hin zu einem Initialisierungsskript zur effizienten Überprüfung von Daten für eine RE wäre außerdem sinnvoll. Dabei ist nach eigener Auffassung lediglich eine Erweiterung des bereits bestehenden Interfaces des Reportmoduls durch Methoden zur automatischen Überprüfung mehrerer Variablenkonstellationen ausreichend. Konkret soll der Analyst mehrere Konstellationen angeben können, für welche dann der Regelbestand berechnet wird. Dieser sollte dann durch tiefergehende Statistiken hinsichtlich verschiedener Regelklassen („schwach“ bis „stark“) und der Möglichkeit zur Separierungen durch bspw. Produktsparten erweitert werden. Auf diese Weise könnte der Prozess der Datenüberprüfung zur Eignung für eine RE abermals beschleunigt werden.

Schließlich kann das Mess-Skript und Reportmodul auch zum Zwecke des Monitoring eingesetzt werden. Durch Definierung von Fehlentwicklungen (z.B. Grenzwert bei der Regelbestandsänderung) könnten neben entsprechenden Meldungen an den Analysten bereits automatische Methoden zur temporären Behebung (z.B. Ausklammern aller Transaktionen bis zum zuletzt funktionierenden Stand) des Fehlers zur Gewährleistung der Funktionalität vorgenommen werden. Diese und weitere Monitoringfunktionen stellen die abschließende Empfehlung zur Weiterentwicklung des Prototyps dar.

7. Literatur

Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engeneering, Vol.17* , S. 734-748.

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules. *VLDB '94 Proceedings of the 20th International Conference on Very Large Data Bases* , 487-499.

Agrawal, R., Imielinski, T., & Swami, A. (1993a). Database mining: a performance perspective. *IEEE Transactions on Knowledge and Data Engineering, Vol.5* , S. 914-925.

Agrawal, R., Imilienski, T., & Swami, A. (1993b). Mining association rules between sets of items in large databases. *Proceedings of the 1993 ACM SIGMOD international conference on Management of data* , S. 207-216.

Appice, A., & Buono, P. (2005). Analyzing Multi-level Spatial Association Rules Through a Graph-Based Visualization. *Innovations in Applied Artificial Intelligence, Vol.3533* , S. 448-458.

Ben Schafer, J., Frankowski, D., Herlocker, J., & Sem, S. (2007). Collaborative Filtering Recommender Systems. *Lecture Notes in Computer Science, Vol.4321* , S. 291-324.

Berners-Lee, T., Cailliau, R., Groff, J.-F., & Pollermann, B. (2010). World-wide web: the information universe. *Internet Research, Vol. 20* , S. 461-471.

Bitkom. (2012). *Big Data im Praxiseinsatz – Szenarien, Beispiele, Effekte.*

Borgelt, C. (2012). Frequent item set mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* , S. 437-456.

Brin, S., Motwani, R., & Silverstein, C. (1997). Beyond market baskets: generalizing association rules to correlations. *Proceedings of the 1997 ACM SIGMOD international conference on Management of data* , S. 265-276.

Brin, S., Motwani, R., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data* , S. 265-276.

Bruzzese, D., & Davino, C. (2008). Visual Mining of Association Rules. In *Visual Data Mining* (S. 103-122). Heidelberg/Berlin: Springer.

Bundesministerium für Bildung und Forschung. (2015a). Abgerufen am 22. März 2015 von http://www.bmbf.de/de/9072.php

Bundesministerium für Bildung und Forschung. (2015b). Abgerufen am 22. März 2015 von http://www.hightech-strategie.de/de/Industrie-4-0-59.php

Campos, M. M., Stengard, P. J., & Milenova, B. L. (2005). Data-Centric Automated Data Mining. *Machine Learning and Applications* .

Carr, M., & Verner, J. (1993). *Prototyping and Software Development Approaches.*

Chang, W., Cheng, J., Allaire, J., Xie, Y., & McPherson, J. (2016). *shiny: Web Application Framework for R. R package version 0.13.1.* Von https://CRAN.R-project.org/package=shiny abgerufen

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., et al. (2000). *Crisp-DM 1.0.* Crisp-DM Consortium.

Chaudhuri, S., & Dayal, U. (1997). An overview of data warehousing and OLAP technology. *ACM SIGMOD Record, Vol.26* , S. 65-74.

Chen, H., Chiang, R. L., & Storey, V. C. (2012). Business Intelligence and Analytics: From Big Data to Big Impact. *MIS Quartely, Vol.36* , S. 1165-1188.

daCosta, F. (2013). *Rethinking the Internet of Things: A Scalable Approach to Connecting Everything.* Apress Media.

EMC Education Services. (2015). *Data Science & Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data.* Indianapolis: John Wiley & Sons.

GitHub Inc. (2015). *GitHub*. Abgerufen am 25. Juli 2016 von https://github.com/stedy/Machine-Learning-with-R-datasets/blob/master/groceries.csv

Gluchowski, P., Gabriel, R., & Dittmar, C. (2008). *Management Support Systeme und Business Intelligence.* Springer.

Goldfarb, A., & Tucker, C. (2012). Privacy and Innovation. *Innovation Policy and the Economy, Vol.12* , S. 65-90.

Guerrieri, P., & Bentivegna, S. (2011). *The Economic Impact of Digital Technologies: Measuring Inclusion and Diffusion in Europe.* Northampton: Edward Elgar Publishing Inc.

Hahsler, M. (2006). *Inside-R*. Abgerufen am 25. Juli 2016 von http://www.inside-r.org/packages/cran/arules/docs/Groceries

Hahsler, M., & Chelluboina, S. (2015). *arulesViz: Visualizing Association Rules and Frequent Itemsets. R package version 1.1-0.* Von https://CRAN.R-project.org/package=arulesViz abgerufen

Hahsler, M., Bolanos, M., & Forrest, J. (2015). *stream: Infrastructure for Data Stream Mining. R package version 1.2-2.* Von https://CRAN.R-project.org/package=stream abgerufen

Hahsler, M., Buchta, C., Gruen, B., & Hornik, K. (2015). *arules: Mining Association Rules and Frequent Itemsets. R package version 1.3-1.* Von https://CRAN.R-project.org/package=arules abgerufen

Hallmann, M. (1990). *Prototypig komplexer Softwaresyteme.* Wiesbaden: Springer.

Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *SIGMOD '00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data* , S. 1-12.

Hilbert, M. (2011). The World's Technological Capacity to Store, Communicate, and Compute Information. *Science, Vol.332* , 60-65.

Hill, W., Stead, L., Rosenstein, M., & Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. *Proceedings on Conference Human Factors in Computing Systems.*

Holm, P., Jarrick, A., & Scott, D. (2015). The Digital Humanities. *Humanities World Report* , S. 64-83.

IBM Institute for Business Value. (2013). *Analytics: A blueprint for value.* Somers: IBM Global Services.

Ihaka, R. (1998). R: Past and Future History. *Proceedings of the 30th Symposium on the Interface* , S. 392-369.

Ihaka, R., & Gentleman, R. (1996). R: A Language for Data Analysis and Graphics. *Journal of Computational and Graphical Statistics, Vol.5* , S. 299-314.

Kantardzic, M. (2011). *Data Mining: Concepts, Models, Methods and Algorithms.* Hoboken: John Wiley & Sons.

Kuonen, D. (2004). *Data Mining and Statistics: What is the Connection?* Lausanne: Statoo Consulting.

Linoff, G. S., & Berry, M. J. (2011). *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management.* John Wiley & Sons.

Lönnqvist, A., & Pirttimäki, V. (2006). The Measurement of Business Intelligence. *Information Systems Management, Vol.23* , S. 32-40.

Manhartsberger. (1998). Prototyping - Theorie und Praxis. *Ergonomie und Informatik, Vol.33* , S. 19-23.

Matloff, N. (2011). *The Art of R Programming: A Tour of Statistical Software Design.* San Francisco: William Pollock.

McKinsey Global Institute. (2011). *Big data: The next frontier for innovation, competition, and productivity.* McKinsey & Company.

Nah, F. F.-H. (2004). A study on tolerable waiting time: how long are Web users willing to wait? *Behaviour & Information Technology, Vol.23* , S. 153-163.

Neckel, P. (2011). *Data-Mining-Tools auf dem Prüfstand.*

Neckel, P. (2007). *Self-Acting Data Mining: Das neue Paradigma der Datenanalyse.*

Owen, M. J. (2007). Digitalization and the Evolution of Scientific Communication. *Information Science and Knowledge Management, Vol.11* , S. 191-225.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, (S. 175-186).

Ricci, F., Rockach, L., Shapira, B., & Kantor, P. (2011). *Recommender Systems Handbook.* Springer.

Ripley, B., & Lapsley, M. (2015). *RODBC: ODBC Database Access. R package version 1.3-12.* Von https://CRAN.R-project.org/package=RODBC abgerufen

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based Collaborative Filtering Recommendation Algorithms. *Proceedings of Tenth International World Wide Web Conference* .

Schein, A. I., Pepescul, A., Ungar, L. H., & Pennock, D. M. (2002). *Methods and Metrics for Cold-Start Recommendations.* Pennsylvania.

Sekhavat, Y. A., & Hoeber, O. (2013). Visualizing Association Rules Using Linked Matrix, Graph, and Detail Views. *International Journal of Intelligence Science, Vol.3* , S. 34-49.

Shardanand, U., & Maes, P. (1995). Social Information Filtering: Algorithms for Automating ‘Word of Mouth’. *Proceedings of Conference Human Factors in Computing Systems.*

Shneiderman, B. (1984). Response Time and Display Rate in Human Performance with Computers. *Computing Surveys, Vol.16* , S. 265-285.

Smith, D., & Crossland, M. (2008). Realizing the Value of Business Intelligence. *IFIP Advances in Information and Communication Technology, Vol.274* , S. 163-174.

Statista GmbH. (2016a). *Statista*. Abgerufen am 2. Februar 2016 von http://de.statista.com/statistik/daten/studie/267974/umfrage/prognose-zum-weltweit-generierten-datenvolumen/

Statista GmbH. (2016b). *Statista*. Abgerufen am 9. Februar 2016 von http://de.statista.com/statistik/daten/studie/257974/umfrage/weltweiter-umsatz-mit-big-data-loesungen/

Sun, Y., Bie, R., Thomas, P., & Cheng, X. (2014). Advances on data, information, and knowledge in the internet of things. *Personal and Ubiquitous Computing, Vol.18* , S. 1793-1795.

Tan, P.-N., Steinebach, M., & Kumar, V. (2006). *Association Analysis: Basic Concepts and Algorithms.*

Tuszynski, J. (2014). *caTools: Tools: moving window statistics, GIF, Base64, ROC AUC, etc.. R package version 1.17.1.* Von https://CRAN.R-project.org/package=caTools abgerufen

Vossen, G. (2014). Big data as the new enabler in business and other intelligence. *Vietnam Journal of Computer Science, Vol.1* , S. 3-14.

Warnes, R. G., Bolker, B., & Lumley, T. (2015). *gtools: Various R Programming Tools. R package version 3.5.0.* Von https://CRAN.R-project.org/package=gtools abgerufen

Wickham, H. (2011). The Split-Apply-Combine Strategy for Data Analysis. *Journal of Statistical Software, Vol.40* , S. 1-29.

Worthington, R. (2014). Digitization and Sustainability. *State of the World* , S. 53-62.

Yang, L. (2003). Visualizing Frequent Itemsets, Association Rules, and Sequential Patterns in Parallel Coordinates. *Computational Science and Its Applications — ICCSA 2003* , 21-30.

Zaki, M. J. (2000). Scalable Algorithms for Association Mining. *IEEE Transactions on Knowledge and Data Engineering, Vol.12* , S. 372-390.

Zhao, Q., & Bhowmick, S. S. (2003). Association Rule Mining:A Survey. *Computer Science, Vol.4* .