technische universität dortmund

IT in Produktion und Logistik

Vergleich von Methoden zur Behandlung fehlender Werte und zur Datentransformation für die Datenvorverarbeitung der Wissensentdeckung in Datenbanken der Produktion

Masterarbeit zur Erlangung des Grades Master of Science (M. Sc.)

Autor: Jan Wilken
Matrikel-Nr.: 210281
Studiengang: Maschinenbau
[redacted]

Ausgabedatum: 22.04.2025
Abgabedatum: 15.10.2025

Erstprüferin: Dr.-Ing. Dipl.-Inform. Anne Antonia Scheidler
Zweitprüfer: Florian Hochkamp, M. Sc.

Inhaltsverzeichnis

Abbildungsverzeichnis I

Tabellenverzeichnis II

Algorithmenverzeichnis III

Abkürzungsverzeichnis IV

1 Einleitung 1

2 Grundlagen zur Datenvorverarbeitung für die Wissensentdeckung in Produktionsdatenbanken 4

2.1 Wissensentdeckung in Datenbanken . . . 4
2.1.1 Daten, Informationen und Wissen . . . 4
2.1.2 Statistische Grundlagen . . . 7
2.1.3 Vorgehensmodelle zur Wissensentdeckung in Datenbanken . . . 13
2.1.4 Data Mining . . . 18
2.2 Datenvorverarbeitung . . . 19
2.2.1 Fehlende Werte . . . 21
2.2.2 Methoden zur Behandlung von fehlenden Werten . . . 23
2.2.3 Aufgaben der Datentransformation . . . 26
2.2.4 Methoden der Datentransformation . . . 27
2.2.5 Metriken zum Vergleich von Datenvorverarbeitungsmethoden . . . 29
2.3 Wissensentdeckung in der Produktion . . . 30
2.3.1 Produktion . . . 31
2.3.2 Datenerfassung und -speicherung in der Produktion . . . 32
2.3.3 Eigenschaften von Produktionsdaten . . . 35
2.3.4 Data-Mining-Anwendungsgebiete in der Produktion . . . 37

3 Entwicklung des Experimentierplans zum Vergleich von Datenvorverarbeitungsmethoden 39

3.1 Ziel des Experimentierplans zum Vergleich von Datenvorverarbeitungsmethoden . . . 39
3.2 Ausarbeitung der Anforderungen an den Experimentierplan . . . 40
3.2.1 Anforderungen an den Experimentierplan . . . 41
3.2.2 Anforderungen von Produktionsdaten . . . 43
3.2.3 Anforderungen von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation . . . 44
3.2.4 Anforderungen von Vergleichsmetriken . . . 45

3.3 Erstellung eines Experimentierplans zum Vergleich von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation 47

4 Exemplarische Anwendung des Experimentierplans 61
4.1 Vorstellung des Produktionsdatensatzes zur exemplarischen Anwendung 61
4.2 Implementierung des Experimentierplans in Python 63
4.3 Anwendung des Experimentierplans auf den Produktionsdatensatz 74
4.4 Auswertung der Ergebnisse 79

5 Diskussion und Fazit 86

6 Zusammenfassung und Ausblick 89

Literaturverzeichnis 91

A Anhang A: Vollständiger Experimentierplan 97

B Anhang B: Python Programmcode der exemplarischen Implementierung 101
B.1 Experiment.py 101
B.2 Methoden.py 109
B.3 SVP.py 115
B.4 Auswertung.ipynb 116

C Anhang C: Zusatzmaterial aus der Anwendung des Experimentierplans 124
C.1 Anzahl und Prozentangabe der fehlenden Werte aller Merkmale und Zielvariablen . . 124
C.2 Ausschnitt des ausgefüllten statistischen Versuchsplans 126

Abbildungsverzeichnis

2.1 Wissenstreppe angelehnt an North (2021, S. 37) . . . 5
2.2 Konfusionsmatrix zur Bewertung eines Klassifikationsmodells angeleht an Ayaz (2022, S. 36) . . . 9
2.3 Schematische Darstellung eines untersuchten Systems nach Ghazi (2019, S. 1) . . . 12
2.4 Vorgehensmodell des KDD-Prozesses angelehnt an Fayyad et al. (1996a, S. 41) und Fayyad et al. (1996b, S. 29) . . . 14
2.5 Bewertungsschema für den Vergleich der Datenvorverarbeitungsmethoden . . . 30
2.6 Produktionssystem angelehnt an Bloech et al. (2014, S. 3) . . . 31
2.7 Automatisierungspyramide angelehnt an Gesellschaft Mess- und Automatisierungstechnik (2013, S. 4) . . . 33

3.1 Legende für benutzte Symbole im Experimentierplan angeleht an die *DIN 66001* (1983) 40
3.2 Die Säulen der Anforderungen an den Experimentierplan . . . 41
3.3 Start und einleitende Schritte im Experimentierplan . . . 49
3.4 Abgleich der Eigenschaften und Voraussetzungen und Datenvorverarbeitung im Experimentierplan . . . 51
3.5 Auswahl der Methoden für das Experiment . . . 55
3.6 Initiale Überprüfung der Ergebnisse des Data-Mining-Algorithmus . . . 57
3.7 Durchführung des Experiments im Experimentierplan . . . 59
3.8 Experimentierplan in kompakter Darstellung . . . 60

4.1 Aufbau der Produktionslinie aus dem der Datensatz von Liveline Technologies (2025) erhoben worden ist . . . 62
4.2 Programmaufbau der Implementierung des Experimentierplans . . . 65
4.3 Effekt-Diagramm mit Haupteffekten ausgewählter Methoden zur Behandlung fehlender Werte . . . 80
4.4 Effekt-Diagramm mit Haupteffekten von Datentransformationsmethoden zur Skalierung 81
4.5 Effekt-Diagramm mit Haupteffekten von Datentransformationsmethoden zur Aggregation, Datenglättung und Merkmalskonstruktion . . . 81
4.6 Haupteffekte auf den R^2-Durchschnitt der Datenvorverarbeitungsmethoden . . . 82
4.7 136 Wechselwirkungen der Datenvorverarbeitungsmethoden . . . 83
4.8 Vergleich der Haupteffekte auf den R^2-Durchschnitt mit und ohne Train-Test-Split . . . 85

A.1 Experimentierplan Seite 1 . . . 98
A.2 Experimentierplan Seite 2 . . . 99
A.3 Experimentierplan Seite 3 . . . 100

Tabellenverzeichnis

3.1 Anforderungen an den Experimentierplan 46
3.2 Ausgewählte Vergleichsmetriken der Modellgüte für den Experimentierplan 48
3.3 Voraussetzungen von Data-Mining-Algorithmen: Ausgewählte Beispiele 49
3.4 Eigenschaften des Datensatzes ausgefüllt mit Beispielen 50
3.5 Vorauswahl an Methoden zur Behandlung von fehlenden Werten für das Experiment . 52
3.6 Vorauswahl an Methoden zur Datentransformation für das Experiment 53
3.7 Vorlage für statistischen Versuchsplan 54

4.1 Eigenschaften der Funktionsgruppen von Datenvorverarbeitungsmethoden 67

C.1 Anzahl und Prozentangabe der fehlenden Werte pro Merkmal 125
C.2 Anzahl und Prozentangabe der fehlenden Werte pro Zielvariable 126
C.3 Ausschnitt der ersten 20 Zeilen des Versuchsplans 127
C.4 Ausschnitt der ersten 20 Zeilen des Versuchsplans mit Ergebnissen 128

Algorithmenverzeichnis

4.1 Auszug der Gruppierung der Datenvorverarbeitungsmethoden und Einteilung der Anwendung vor oder nach dem Train-Test-Split 66
4.2 Code der Funktion fw_KNNImputation 68
4.3 Code der Funktion dt_Differenz_bilden 69
4.4 Bewertungsfunktion für die Data-Mining-Aufgabe in Pseudocode 70
4.5 Python-Funktion für Effektberechnung mit Funktionsaufruf 73
B.1 Experimentdurchlauf 101
B.2 Implementierung der Datenvorverarbeitungsmethoden als Funktionen 109
B.3 Experimentdurchlauf Funktion zur Erstellung des statistischen Versuchsplans 115
B.4 Auswertung: Effektberechnung und Haupteffektdiagramme erstellen 116

Abkürzungsverzeichnis

ACID Atomicy, Consistency, Isolation, Durability

CNC Computerized Numerical Control

CPS Cyber-Physisches System

CRISP-DM Cross Industry Standard Process for Data Minig

CRUD Create, Read, Update, Delete

CSV Comma Seperated Value

DBSCAN Density-Based Spatial Clustering of Applications with Noise

ERP Enterprise Resource Planning

KNN K-Nearest-Neighbor

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

MAR Missing At Random

MCAR Missing Completely At Random

MES Manufacturing Execution System

MICE Multiple imputations by chained equations

MNAR Missing Not At Random

MSE Mean Squared Error

RFID Radio-Frequency Identification

RMSE Root Mean Squared Error

SCADA Supervisory Control And Data Acquisition

SPS Speicherprogrammierbare Steuerung

SVP Statistische Versuchsplanung

1 Einleitung

Im Zeitalter der Industrie 4.0 wird die Produktion zunehmend digitalisiert und vernetzt (Lasi et al. 2014). Dies wird hauptsächlich mit cyber-physischen Systemen (CPS) durchgeführt, die mit Sensoren und Aktuatoren die physische und digitale Welt verbinden (Bauernhansl 2014). In den CPS werden Daten generiert, indem Zustände von Aktuatoren ausgelesen oder Messungen von Sensoren in der physischen Welt gespeichert werden. Diese Daten bieten das Potenzial, neues Wissen aus ihnen zu generieren (Runkler 2015). Seit den 90er Jahren hat sich die Wissensentdeckung in Datenbanken (Knowledge Discovery in Databases, KDD) verbreitet und wird mit steigender Datenmenge vermehrt angewendet (Han et al. 2023). KDD strebt mithilfe von Datenanalysemodellen, in diesem Kontext auch Data-Mining-Algorithmen genannt, die Entdeckung von Mustern in großen Datenmengen an (Fayyad et al. 1996a). Diese können neue Erkenntnisse liefern, die Einordnung neuer Datenpunkte ermöglichen und als Grundlage für fundierte Entscheidungen oder Prozessoptimierungen dienen (Runkler 2015).

Hierfür ist eine qualitativ hochwertige Datenbasis mit Daten in der richtigen Form wichtig (Zhang et al. 2003). In der Praxis ist dies selten gegeben, weshalb die Datenvorverarbeitung ein essenzieller Schritt für KDD ist (García et al. 2015). Des Weiteren wird der Datenvorverarbeitung die meiste Arbeitszeit zugesprochen (Munson 2012). Aufgrund der großen Auswirkung der Datenvorverarbeitung auf den KDD-Prozess besteht in diesem Schritt das Potenzial, den gesamten Prozess zu verbessern. Die Datenvorverarbeitung dient der Konvertierung von Daten aus der realen Welt in ein Format, das für Data-Mining-Algorithmen geeignet ist (García et al. 2015). Sie kann in verschiedene Aufgaben eingeteilt werden. Zu den Standardaufgaben gehören die Datenbereinigung, die Datentransformation, die Datenintegration, die Behandlung von fehlenden Werten und die Rauschbewältigung in Daten (García et al. 2015). Vor allem die Behandlung von fehlenden Werten und die Datentransformation werden in dieser Arbeit behandelt, da die Anwendung dieser Datenvorverarbeitungsschritte für einige Data-Mining-Algorithmen Voraussetzung sein kann, wie Çetin und Yıldız (2022) für die Behandlung von fehlenden Werten und García et al. (2015) für die Datentransformation herausstellen.

Es existieren viele verschiedene Methoden mit unterschiedlichen Ansätzen, fehlende Werte zu behandeln (Little und Rubin 2020). Zusätzlich existieren verschiedene Datentransformationsaufgaben, die jeweils mit unterschiedlichen Methoden auf unterschiedliche Weise gelöst werden (Cleve und Lämmel 2024). Für die Kombination der Behandlung fehlender Werte und der Datentransformation gibt es folglich eine sehr große Auswahl an Methoden, die in der Datenvorverarbeitung angewendet werden können, um die Ergebnisse des Data-Mining-Algorithmus zu ermöglichen und zu verbessern. Eine Unterstützung beim Vergleich der Methoden trägt dazu bei, eine fundierte Auswahl für die Datenvorverarbeitung treffen zu können. Nach Kenntnisstand des Autors ist diese Unterstützung

in der Literatur noch nicht vorhanden. Ein generischer Vergleich dieser Methoden hilft nur bedingt weiter, da unter anderem von Çetin und Yıldız (2022) aufgezeigt wird, dass die Güte der Datenvorverarbeitungsmethoden von vielen Faktoren abhängig ist, zum Beispiel vom Datensatz oder dem Data-Mining-Algorithmus. Ein Methodenvergleich, der diese Faktoren berücksichtigt und darauf angepasste Erkenntnisse für die Behandlung von fehlenden Werten und für die Datentransformation liefert, wird benötigt.

Das Hauptziel dieser Arbeit besteht deshalb in der Entwicklung eines Experimentierplans, der den Vergleich verschiedener Methoden zur Behandlung fehlender Werte und zur Datentransformation ermöglicht. Um dieses Ziel zu erreichen, müssen mehrere Teilziele erfüllt werden. Das initiale Teilziel besteht in der Ausarbeitung von Anforderungen an den Experimentierplan. Die Identifizierung der Anforderungen erfolgt dabei aus unterschiedlichen Perspektiven. Das zweite Teilziel besteht in der Konzeption des Experimentierplans. Die Konzeption stützt sich dabei auf die zuvor erarbeiteten Anforderungen. Das dritte Teilziel ist die Implementierung und Anwendung des Experimentierplans auf einen adäquaten Produktionsdatensatz. Idealerweise ein Datensatz mit fehlenden Werten, auf den Datentransformationen angewendet werden müssen.

Die Realisierung der Ziele erfolgt gemäß der nachfolgend dargelegten Vorgehensweise. In Kapitel 2 wird der aktuelle Wissensstand erfasst, der für die Entwicklung des Experimentierplans grundlegend ist. Zunächst werden Grundlagen für die Wissensentdeckung in Datenbanken dargestellt, da diese für eine tiefere Betrachtung der Datenvorverarbeitung und der Spezialisierung der Produktion benötigt werden. Dabei werden auch statistische Grundlagen aufgegriffen, da diese für die Versuche im Experimentierplan benötigt werden. In der Datenvorverarbeitung werden die Behandlung fehlender Werte, die Datentransformation und die Methoden dieser Datenvorverarbeitungsschritte behandelt, da sich die Arbeit auf den Vergleich von Methoden dieser Datenvorverarbeitungsschritte fokussiert. Um eine objektive Bewertung der Versuche im Experimentierplan zu gewährleisten, werden auch Metriken zum Vergleich der Methoden vorgestellt. Abschließend wird die Spezialisierung der Wissensentdeckung in der Produktion vorgestellt, um die Anforderungen an den Experimentierplan auch aus der Perspektive der Domäne artikulieren zu können.

In Kapitel 3 wird der Experimentierplan entwickelt und erstellt. Dafür werden zuerst das Ziel des Experimentierplans sowie die Vorgehensweise bei der Entwicklung erläutert. Anschließend werden die Anforderungen an den Experimentierplan ausgearbeitet, um das erste Teilziel zu erreichen. Die Anforderungen werden, in vier Unterkapitel aufgeteilt, aus verschiedenen Sichtweisen artikuliert. Diese gehen aus Kapitel 2 und den Zielen dieser Arbeit hervor. Es handelt sich um Anforderungen an den Experimentierplan, Anforderungen von Produktionsdaten, Anforderungen von den zu vergleichenden Methoden und Anforderungen von den Vergleichsmetriken. Aufbauend auf den Anforderungen wird im Anschluss der Experimentierplan konzeptioniert und grafisch festgehalten, um die Anwendung zu vereinfachen.

Die anschließende exemplarische Anwendung des Experimentierplans wird in Kapitel 4 behandelt. Dafür werden zunächst der Datensatz und dessen Eigenschaften vorgestellt, weil diese für den Experimentierplan und den Vergleich der Datenvorverarbeitungsmethoden benötigt werden. Im Anschluss wird die Implementierung des Experimentierplans in Python vorgestellt. Einerseits, weil neben den Schritten im KDD-Prozess auch viele weitere Funktionen in Python umgesetzt werden

können. Andererseits, weil die Implementierung auf den Datensatz angepasst ist. Die Anwendung des Experimentierplans auf den vorgestellten Datensatz wird nach der Implementierung vorgestellt, um die Entscheidungen im Experimentierplan zu dokumentieren und die Hintergründe für die Implementierung zu erläutern. Die generierten Ergebnisse aus der Anwendung werden schließlich ausgewertet, um die untersuchten Datenvorverarbeitungsmethoden untereinander zu vergleichen.

In Kapitel 5 wird der Experimentierplan, dessen Anwendung und die erhaltenen Ergebnisse diskutiert und ein Fazit gezogen. Abschließend wird in Kapitel 6 die gesamte Arbeit zusammengefasst und es wird ein Ausblick auf angrenzende Forschungsarbeiten gewagt.

2 Grundlagen zur Datenvorverarbeitung für die Wissensentdeckung in Produktionsdatenbanken

Im Folgenden werden die theoretischen Grundlagen dargestellt, die für die Entwicklung des Experimentierplans erforderlich sind. Zuerst wird KDD mithilfe einer anfänglichen Begriffsabgrenzung vorgestellt. Des Weiteren werden benötigte statistische Grundlagen, Vorgehensmodelle und die Data-Mining-Phase dafür näher betrachtet. Der Fokus im Stand der Technik wird auf den Datenvorverarbeitungsschritten Behandlung von fehlenden Werten und Datentransformation liegen. Metriken, um einen Vergleich von Methoden aus beiden Aufgaben zu ermöglichen, werden ebenso dafür vorgestellt. Abschließend wird eine Verbindung zur Produktion hergestellt, wobei die Erhebung von Daten in der Produktion, die Eigenschaften von Produktionsdaten sowie die Data-Mining-Anwendungsgebiete in der Produktion erörtert werden.

2.1 Wissensentdeckung in Datenbanken

In diesem Abschnitt wird der Prozess des KDD erklärt. Der Begriff KDD hat sich insbesondere in der internationalen Fachliteratur etabliert und wird auch in dieser Arbeit verwendet. Fayyad et al. (1996a) beschreibt den KDD-Prozess als einen Transformationsprozess, in dem Daten in andere Formen umgewandelt werden, die abstrakter, kompakter oder hilfreicher sind. Damit werden aus Daten nützliche Informationen extrahiert, mit denen neues Wissen entdeckt werden kann. Für ein klares Verständnis ist eine scharfe Begriffsabgrenzung von Daten, Informationen und Wissen nötig, die dafür zu Beginn erfolgt.

2.1.1 Daten, Informationen und Wissen

Obwohl eine Abgrenzung der Begriffe Daten, Informationen und Wissen im Folgenden vorgenommen wird, werden diese in der wissenschaftlichen Literatur zumeist in Abhängigkeit voneinander definiert (Liew 2007). Für das Grundverständnis der drei Begriffe wird die Wissenstreppe von North (2021) herangezogen, die nachfolgend in Abbildung 2.1 gezeigt wird.

In Abbildung 2.1 werden die Zusammenhänge von Zeichen, Daten, Informationen und Wissen modelliert. Die Treppe symbolisiert dabei, dass die Stufen aufeinander aufbauen und eine Grundlage für die nächste Stufe darstellen. Zeichen stehen ganz unten auf der Wissenstreppe und werden

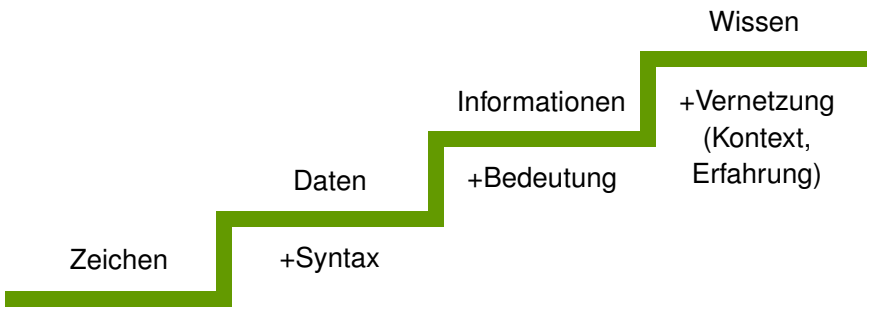

Abbildung 2.1: Wissenstreppe angelehnt an North (2021, S. 37)

erst durch Ordnungsregeln, wie eine Syntax, zu Daten (North 2021). Daten sind also geordnete Zeichen, die noch nicht interpretiert sind, wie z. B. die Zahl 100 (North 2021). Erst mit der Zugabe einer Bedeutung, zum Beispiel mit einer Einheit, wird aus den Daten eine Information (North 2021). Aus der Zahl 100 wird die Temperatur 100 °C. Aus Informationen wird Wissen, wenn diese hilfreich miteinander vernetzt werden (North 2021). Die Temperatur 100 °C wird mit anderen Informationen vernetzt. Dabei entsteht das Wissen, dass Wasser ab dieser Temperatur anfängt zu kochen.

Daten bilden die Grundlage für Informationen und diese sind der Rohstoff für die Generierung von Wissen (North 2021). Deshalb und weil sie in Computern gespeichert und verarbeitet werden können, wird ihnen eine grundlegende Bedeutung zugesprochen. Sie dienen somit als Ausgangspunkt für weiterführende Analysen und Entscheidungsprozesse. Angesichts dieser zentralen Rolle von Daten ist es im Folgenden sinnvoll, sich näher mit deren Eigenschaften und Funktionen auseinanderzusetzen.

Daten sind Objekte, mit denen Anwendungen auf Computern interagieren (Gumm et al. 2011). Sie besitzen verschiedene Datentypen und können je nach Datentyp mit unterschiedlichen Operationen behandelt werden (Gumm et al. 2011). Im Bereich der Datenspeicherung existieren vier Grundoperationen, die auf alle Daten gleichermaßen Anwendung finden: Create (Erschaffen), Read (Lesen), Update (Aktualisieren) und Delete (Löschen), die in Kurzform auch als CRUD bezeichnet werden (Reis und Housley 2023). In vielen Fällen besteht die Notwendigkeit, umfangreiche Datenbestände dauerhaft zu speichern und diese von mehreren Benutzern[1] simultan zu bearbeiten. In solchen Situationen werden Datenbanksysteme eingesetzt (Gumm et al. 2011). Ein Datenbanksystem besteht aus den Daten, die in einer Datenbank gespeichert werden, und einem Datenbankverwaltungssystem (DBVS) (Mertens et al. 2023). Datenbanksysteme sind entwickelt worden, um die Speicherung der Daten von den Anwendungen, die sie benutzen, zu entflechten (Mertens et al. 2023). Ohne Datenbanken würden die Daten, auf die jeweilige Anwendung abgestimmt, separat von jeder Anwendung gespeichert werden (Mertens et al. 2023). Im Falle des Zugriffs unterschiedlicher Anwendungen auf übereinstimmende Datenbestände besteht die Gefahr einer unkontrollierten Redundanz und inkonsistenter Datenbestände (Mertens et al. 2023). Ein Zugriff auf ein Datenbanksystem kann mehrere Operationen beinhalten, die gebündelt als Transaktion bezeichnet werden (Gumm et al. 2011).

Gumm et al. (2011) stellt heraus, dass für einen Mehrbenutzerbetrieb, von verschiedenen Anwen-

[1] Hinweis zur Sprachverwendung: In dieser Arbeit wird aus Gründen der Lesbarkeit ausschließlich die generisch-maskuline Form verwendet. Diese ist jedoch als geschlechtsneutral zu verstehen und schließt alle Geschlechter gleichermaßen ein.

dungen oder verschiedenen Benutzern, folgende Bedingungen an ein DBVS von essenzieller Bedeutung sind. Die Bedingungen sind Atomarität, Konsistenz, Isolation und Dauerhaftigkeit und werden in der Bedeutung von Gumm et al. (2011) übernommen. In der Fachliteratur werden die Bedingungen mit den englischen Übersetzungen als ACID abgekürzt.

A: Atomicy (Atomarität). Eine Transaktion wird ganz oder gar nicht ausgeführt.

C: Consistency (Konsistenz). Der Datenbestand wird von einen konsistenten Zustand in einen anderen konsistenten Zustand überführt.

I: Isolation (Isolation). Die Transaktionen werden isoliert von anderen Transaktionen ausgeführt. Das Ergebnis einer Transaktion kann deshalb nicht von einer anderen Transaktion beeinflusst werden.

D: Durability (Dauerhaft). Nach Abschluss einer Transaktion sind die Änderungen dauerhaft in der Datenbank.

Datenbanksysteme können verschiedene Datenbankmodelle umfassen, die sich durch unterschiedliche Eigenschaften auszeichnen. Die Eignung der Modelle für ein spezifisches Aufgabengebiet kann variieren (Kemper und Eickler 2015). Das am häufigsten in der Praxis verwendete Datenbankmodell ist, auch dank seiner einfachen Struktur, das relationale Datenbankmodell (Kemper und Eickler 2015). In diesem Fall basiert die Datenbank auf Relationen, die in zweidimensionalen Tabellen mit einer festen Spaltenanzahl und einer beliebigen Zeilenanzahl abgebildet werden (Mertens et al. 2023). Die Zeilen einer Tabelle stellen Entitäten dar, die als Tupel bezeichnet werden (Kemper und Eickler 2015). In der Darstellung einer Tabelle als Matrix wird bei Tupeln auch von Datenpunkten gesprochen (Runkler 2015). In Python besteht die Möglichkeit, eine Datenmatrix beispielsweise mithilfe eines Pandas DataFrames zu ver- und zu bearbeiten. Dabei werden die Zeilen und Spalten mittels Identifikatoren ausgezeichnet, um ein gezieltes Abrufen zu ermöglichen (The pandas development team 2025). Die Reihenfolge der Tupel in einer Tabelle ist egal und es gibt keine zwei identischen Tupel (Mertens et al. 2023). Deshalb sind die Tupel durch einen Schlüssel, den Primärschlüssel, eindeutig identifizierbar (Kemper und Eickler 2015). Die Darstellung der Merkmale, in der Fachliteratur ebenso als Attribute bezeichnet, einer Relation erfolgt in den Spalten (Kemper und Eickler 2015). Auch bei den Spalten ist die Reihenfolge innerhalb der Tabelle irrelevant (Mertens et al. 2023). Darüber hinaus sind die Merkmale von Relationen atomar, was impliziert, dass sie aus einer einelementigen Menge bestehen, und die Spalten einer Tabelle sind homogen, was bedeutet, dass alle Werte innerhalb einer Spalte denselben Datentyp aufweisen (Mertens et al. 2023).

Datentypen dienen der Strukturierung von Daten. Sie legen fest, dass Daten in Form von Zahlen, Wörtern, Buchstaben, Datumsangaben oder booleschen Werten dargestellt werden. Interessanter ist jedoch eine Unterteilung nach den Kriterien Ordnung und Rechnen, wie sie Cleve und Lämmel (2024) vornimmt:

- Nominale Daten können nicht geordnet werden und können nur auf Gleichheit geprüft werden. Rechenoperationen können ebenso nicht auf sie angewendet werden.
- Ordinale Daten können nach einer Reihenfolge geordnet werden, zum Beispiel: klein, mittel,

groß. Damit ist ein Vergleich auf Gleichheit und Relation zueinander möglich. Es ist nicht möglich, mit ihnen zu rechnen.

- Metrische Daten besitzen die Eigenschaften von reellen Zahlen. Vergleiche und Rechenoperationen können mit ihnen durchgeführt werden.
  - Metrische diskrete Daten können nur Werte in bestimmter Schrittreihenfolge annehmen.
  - Metrische kontinuierliche Daten können jeden Wert innerhalb des Definitionsbereichs annehmen.

Informationen sind Nachrichten, die eine relevante Bedeutung, Implikation oder einen Input für eine Entscheidung oder eine Handlung enthalten (Liew 2007). Sie sind außerdem besonders, weil sie das Bindeglied zwischen Daten und Wissen sind. Im Computer sind Informationen mit Daten verbunden und gespeichert und im menschlichen Gehirn wird Wissen, aufbauend auf verknüpften Informationen gespeichert (Liew 2007). Die Parallele zwischen dem KDD-Prozess und Informationen ist demnach die Verknüpfung von Daten und Wissen. Das damit zusammenhängende Informationsmanagement umfasst laut Liew (2007) die Sammlung von historischen, aktuellen und möglichen prognostizierten zukünftigen Ereignissen. Außerdem stellt der Autor heraus, dass die Analyse von Informationen für Entscheidungsfindungen oder Problemlösungen auch zum Informationsmanagement zählt.

Wissen definiert Probst et al. (2012, S. 23) als „[...] die Gesamtheit der Kenntnisse und Fähigkeiten, die Individuen zur Lösung von Problemen einsetzen“. Wissen basiert auf Daten und Informationen, ist jedoch, im Gegensatz zu diesen, individuell (Probst et al. 2012). Dies ist auf den individuellen Prozess der Wissensgenerierung zurückzuführen (North 2021). Da Datenbanken meist nicht nur Daten für individuelle Personen beinhalten, ist auch der organisationale Wissensbegriff von Bedeutung. Laut Probst et al. (2012) umfasst dieser sowohl die individuellen als auch die kollektiven Wissensbestände, auf die eine Organisation zugreifen kann, um Probleme zu lösen oder Aufgaben zu bewältigen. Darüber hinaus schließt der organisationale Wissensbegriff auch die Daten- und Informationsbestände einer Organisation mit ein, da diese die Grundlage für individuelles und organisationales Wissen bilden (Probst et al. 2012).

Nach der Erläuterung der Grundlagen und Abgrenzungen der Begriffe werden im nächsten Abschnitt die statistischen Grundlagen behandelt. Diese sind von Bedeutung, da sie in verschiedenen Bereichen rund um die Datenanalyse und Experimente benötigt werden.

2.1.2 Statistische Grundlagen

In der Statistik wird die Datensammlung, Datenanalyse, Interpretation oder Erklärung und die Datenpräsentation behandelt (Han et al. 2023). Wegen dieser Gemeinsamkeiten zeigt Han et al. (2023) eine inhärente Verbindung zwischen dem KDD-Prozess und der Statistik auf. Ein statistisches Modell ist eine Reihe von mathematischen Funktionen, die ein Verhalten von Objekten in einer Zielklasse beschreiben und dies in Form von Zufallsvariablen sowie Wahrscheinlichkeitsverteilungen darstellen (Han et al. 2023). Welche Rolle die Statistik in der KDD einnimmt und welche weiteren statistischen

Grundlagen zur Experimentplanung notwendig sind, wird in diesem Unterkapitel erörtert.

Statistik in der Wissensentdeckung in Datenbanken

Zu Beginn des KDD-Prozesses ist ein Überblick über die Daten essenziell. Dafür werden Standardbeschreibungen aus der Statistik benutzt und auf einzelne Merkmale angewendet (Han et al. 2023). Um zu untersuchen, wo die Werte liegen, werden Maße der zentralen Tendenz verwendet (Han et al. 2023). Dazu gehören der Mittelwert (Summe aller Werte / Anzahl der Werte, vgl. Gleichung 2.1), der Median (mittlerer Wert einer geordneten Menge) und der Modus (am häufigsten auftretender Wert), auch Modalwert genannt (Sachs 1993). Um die Verteilung der Werte zu untersuchen, wird die Dispersion der Daten herangezogen. Dazu gehören die Spannweite (vom kleinsten bis zum größten Wert), das Quartil, die interquartile Spannweite, der Boxplot und die Varianz mit der Standardabweichung (Han et al. 2023). Es gibt drei Quartile (Q1, Q2, Q3), die eine geordnete Menge in 4 gleich große Teile mit jeweils 25% der Werte einteilen und deshalb die Menge bei 25% (Q1), 50% (Q2) und 75% (Q3) der Werte trennen (Ribeiro et al. 2017). Die interquartile Spannweite bildet die 50% der Werte in der Mitte zwischen Q1 und Q3 ab (Sachs 1993). Mit den Werten der Spannweite, der interquartilen Spannweite und dem Median kann ein Boxplot gebildet werden (Sachs 1993). Die Varianz (σ^2) für N Werte, wenn N groß ist, wird in Gleichung 2.2 dargestellt und gibt an, wie weit verteilt die Werte vom Mittelwert ($\tilde{x}$) sind (Han et al. 2023). Die Standardabweichung wird gebildet, indem die Quadratwurzel der Varianz gezogen wird und ist eine gute Kenngröße, um die Verteilung der Werte in einer Menge zu bemessen (Han et al. 2023). Eine geringe Standardabweichung indiziert, dass alle Werte nah am Mittelwert liegen. Sollte die Standardabweichung null sein, gibt es nur einen Wert in der Menge.

$$\tilde{x} = \frac{\sum_{i=1}^{N} x_i}{N} \tag{2.1}$$

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \tilde{x})^2 = \left(\frac{1}{N}\sum_{i=1}^{N} x_i^2\right) - \tilde{x}^2, \tag{2.2}$$

Um Korrelationen in Datensätzen herauszufinden, werden beispielsweise die Kovarianz, der Korrelationskoeffizient (bei numerischen Werten) und der Wert des x^2 Korrelationstests (bei nominalen Werten) ausgerechnet (Han et al. 2023). Für Korrelationen zwischen Merkmalen wird der Pearson-Korrelationskoeffizient benutzt. Dieser greift auf die Kovarianz und Standardabweichungen der Merkmale zurück und bestimmt die lineare Korrelation zueinander (Plevris et al. 2022). Die Ergebnisse liegen zwischen -1 und 1 und deuten am Rand des Bereichs auf einen starken negativen oder starken positiven Zusammenhang hin (Plevris et al. 2022). Werte, die nahe an der Null liegen, weisen auf keine Korrelation hin (Plevris et al. 2022).

Neben Anwendungen in der Datenvorverarbeitung des KDD-Prozesses ist die Statistik auch im Schritt des Data-Mining zu finden. Statistische Modelle können das Ergebnis eines KDD-Prozesses sein und ein Data-Mining-Algorithmus kann auf Statistik beruhen (Han et al. 2023). Zudem wird die

		Tatsächliche Ergebnisse	
		Positiv	Negativ
Vorhersage	Positiv	True Positive (TP)	False Positive (FP)
	Negativ	False Negative (FN)	True Negative (TN)

Abbildung 2.2: Konfusionsmatrix zur Bewertung eines Klassifikationsmodells angeleht an Ayaz (2022, S. 36)

Statistik bei der Bewertung der Ergebnisse herangezogen. Die sogenannten Evaluierungsmetriken, wie Naidu et al. (2015) sie nennt, sind statistische Kennzahlen, die dazu benutzt werden, die Modelle zu validieren und untereinander zu vergleichen. Naidu et al. (2015) und Botchkarev (2019) listen viele Kennzahlen auf, die bei der Klassifikation und Regression angewendet werden können. An dieser Stelle werden einige erwähnt, die von den Autoren herausgestellt werden, weil sie häufiger in der Praxis Anwendung finden.

Für die Klassifikation sind das die Genauigkeit, die Sensitivität, der F1-Score und die Präzision (Naidu et al. 2015). Diese können alle mit den Einträgen einer Konfusionsmatrix errechnet werden. In einer Konfusionsmatrix werden die vorhergesagten Ergebnisse den tatsächlichen Ergebnissen gegenübergestellt, wie in Abbildung 2.2 dargestellt.

Naidu et al. (2015) beschreibt die Kennzahlen und deren mathematischen Grundlagen, wie nachfolgend dargelegt. Die nachfolgenden Gleichungen sind ebenso diesem Artikel entnommen worden. Die Genauigkeit gibt den Anteil der korrekten positiven Vorhersagen im Vergleich zu allen positiven Vorhersagen an und kann mit Gleichung 2.3 berechnet werden (Naidu et al. 2015). Die Sensitivität, die mit Gleichung 2.4 bestimmt wird, misst die Fähigkeit eines Modells, tatsächlich positive Ergebnisse zu erkennen, indem die richtig vorhergesagten positiven mit allen tatsächlich positiven Ergebnissen in Beziehung gesetzt werden (Naidu et al. 2015). Mit der Präzision, die mit Gleichung 2.5 errechnet wird, wird gemessen, wie viel Prozent insgesamt, also positiv wie auch negativ, richtig vorhergesagt wird (Naidu et al. 2015). Außerdem wird auch der F1-Score oft bei der Bewertung von Klassifikationsmodellen herangezogen. Dieser bildet das harmonische Mittel zwischen Präzision und Sensitivität und wertet somit beide mit gleicher Bedeutung (Taha und Hanbury 2015). Aus der Berechnungsvorschrift ergibt sich Gleichung 2.6 (Taha und Hanbury 2015).

$$\text{Genauigkeit} = \frac{TP}{TP + FP} \tag{2.3}$$

$$\text{Sensitivität} = \frac{TP}{TP + FN} \tag{2.4}$$

$$\text{Präzision} = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.5}$$

$$\text{F1-Score} = \frac{2 * \text{Präzision} * \text{Sensitivität}}{\text{Präzision} + \text{Sensitivität}} \tag{2.6}$$

Die Güte von Regressionsmodellen wird mithilfe von Kennzahlen bewertet, die in verschiedener Art und Weise die Vorhersage mit dem realen Ergebnis miteinander verrechnen (Plevris et al. 2022). Botchkarev (2019) ordnet 31 verschiedene in einer Tabelle ein, von denen hier der Root Mean Squared Error (RMSE), der Mean Absolute Error (MAE) und der Mean Absolute Percentage Error (MAPE) vorgestellt werden. Diese werden in der Praxis am häufigsten zur Bewertung verwendet, wobei früher RMSE und heutzutage MAPE am meisten verwendet werden (Botchkarev 2019).

Im Folgenden werden die Gleichungen aufgeführt, die eine Kennzahl für die Bewertung von Regressionen berechnen. In diesen Fällen werden dabei immer die realen Werte r mit den vorhergesagten Werten des Modells p verglichen. Diese werden mit der Hilfe eines Testdatensatzes erhoben, der neben der Vorhersage auch reale Werte beinhaltet, um die Vorhersage zu bewerten. N steht für die Anzahl der Datenpunkte, die verglichen werden können, weil reale und vorhergesagte Werte für den Datenpunkt vorhanden sind. Die Gleichungen sind aus Plevris et al. (2022) entnommen worden.
Der RMSE, der in Gleichung 2.7 dargestellt wird, nimmt die Wurzel aus dem Mean Squared Error (MSE), welcher den Abstand der Vorhersage zum realen Wert quadriert, damit sowohl positive als auch negative Differenzen betrachtet werden können (Plevris et al. 2022). Die Differenzen werden alle aufsummiert und durch die Anzahl der Datenpunkte geteilt, um einen Mittelwert zu bilden. Die Wurzel sorgt dann dafür, dass die Größenverhältnisse der Fehler den eigentlichen Werten wieder angepasst werden (Plevris et al. 2022). Der MAE wird in Gleichung 2.8 dargestellt und benutzt den Betrag bei der Differenzberechnung zwischen Vorhersage und realem Wert, summiert die Beträge auf und bildet auch den Mittelwert über alle Datenpunkte (Plevris et al. 2022). Der MAPE wird von Plevris et al. (2022) mit Gleichung 2.9 abgebildet, ist eine Weiterentwicklung des MAE und gibt die Differenz von Vorhersage zum realen Wert in Prozentangaben vom realen Wert zurück. Die Autoren weisen außerdem darauf hin, dass die realen Werte nicht nah an der Null liegen sollten, da in diesem Fall kleine Abweichungen zu großen prozentualen Abweichungen führen können und die Kennzahl dadurch verfälschen. Plevris et al. (2022) erklärt, dass das Bestimmtheitsmaß R^2 die Varianz, die durch das Modell erklärt werden kann, mit der echten Varianz in ein Verhältnis setzt (siehe Gleichung 2.10). Die Werte von R^2 liegen für gewöhnlich zwischen 1 und 0, können jedoch theoretisch auch negativ sein. Eine 1 signalisiert demnach eine perfekte Abbildung der realen Werte, während 0 oder negative Werte auf eine schlechte Abbildung der realen Werte durch das Modell hindeuten, wie die Autoren weiter ausführen.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - r_i)^2} \tag{2.7}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|p_i - r_i| \tag{2.8}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|p_i - r_i|}{r_i} \tag{2.9}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(r_i - p_i)^2}{\sum_{i=1}^{N}(r_i - \tilde{r})} \tag{2.10}$$

Abschließend muss erwähnt werden, dass keine Kennzahl alleine Aussagekraft über die Güte eines Modells hat. Die Kombination von mehreren Kennzahlen, unter Berücksichtigung der Fragestellung und des Kontextes, ist für eine Bewertung der Ergebnisse essenziell (Botchkarev 2019). Bei einer Klassifikation sollte beispielsweise die Genauigkeit bei kritischen Vorhersagen sehr hoch sein. Ein Beispiel hierfür ist die letzte Qualitätskontrolle vor dem Versand eines Produkts an den Kunden. Eine falsche positive Vorhersage hätte in diesem Fall schlimmere Folgen als eine falsche negative Vorhersage.

Grundlagen der statistischen Versuchsplanung

Wie für KDD ist auch für die Versuchs- und Experimentierplanung die Kenntnis über einige statistische Grundlagen erforderlich. Aus diesem Grund wird im Folgenden auf die statistischen Grundlagen eingegangen, die für die Planung eines Experiments notwendig sind, damit wissenschaftlich belastbare Ergebnisse erzeugt werden können.

Die statistische Versuchsplanung (SVP) ist ein systematischer Ansatz zur Planung, Durchführung und Analyse von Experimenten, die mit Daten aus realen Laborversuchen oder aus computergestützten Simulationen durchgeführt werden (Ghazi 2019). Zugleich wird das Ziel verfolgt „[...] mit einem möglichst geringen Versuchs- bzw. Simulationsaufwand möglichst genauere Aussagen über die Wirkzusammenhänge von Eingangs- bzw. Einflussparametern (Inputs), Störgrößen (stochastische Inputs) und Zielgrößen (Outputs) zu gewinnen“ (Ghazi 2019, S. 1). Der Zusammenhang dieser Ein- und Ausgänge im System wird in der folgenden Abbildung 2.3 veranschaulicht.

Abbildung 2.3 zeigt die schematische Darstellung eines untersuchten Systems. Die Grundbegriffe daraus werden im folgenden vorgestellt. Zielgrößen, die Siebertz et al. (2017) auch als Qualitätsmerkmale bezeichnet, sind die messbaren Ergebnisse oder Ausgänge eines Systems, die im Rahmen des Experiments betrachtet werden. Mithilfe der Qualitätsmerkmale lässt sich ein gutes von einem schlechten System unterscheiden. In der Praxis sind außerdem meist mehrere Qualitätsmerkmale relevant und ein Merkmal reicht nicht aus, um die Güte eines Systems zu bewerten (Siebertz et al. 2017). Um diese Bewertung zu ermöglichen, ist eine kontinuierliche Ausprägung der Qualitätsmerkmale essenziell, damit ein qualitativer Vergleich innerhalb eines Qualitätsmerkmals möglich ist (Siebertz et al. 2017).

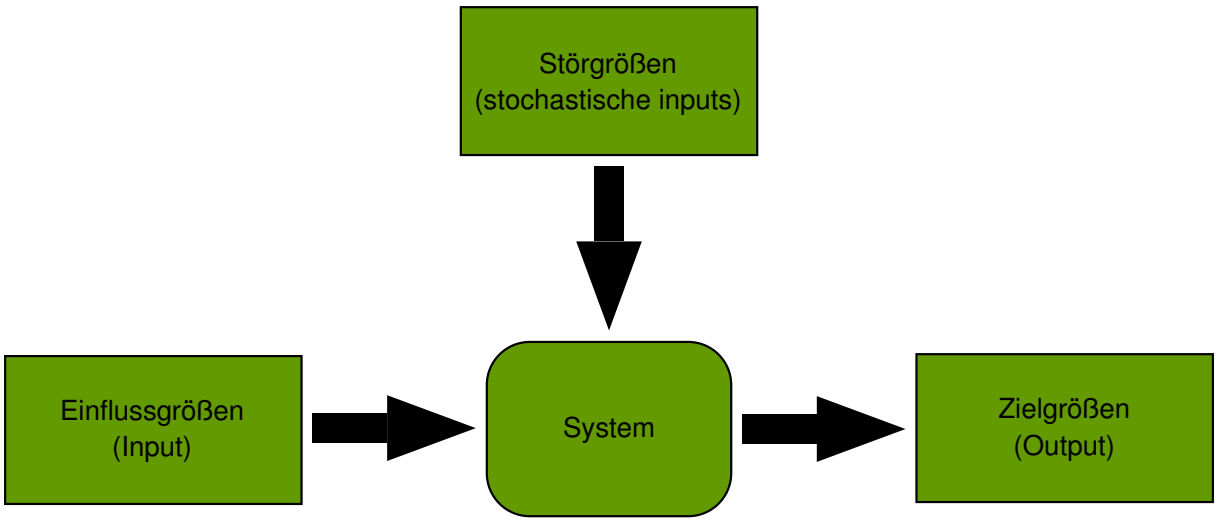

Abbildung 2.3: Schematische Darstellung eines untersuchten Systems nach Ghazi (2019, S. 1)

Alle Eingangsgrößen in das System werden Parameter genannt. Faktoren sind eine Teilmenge aller Parameter und besitzen die Eigenschaft, im Experimentierplan enthalten zu sein und in diesem systematisch verändert zu werden (Siebertz et al. 2017). In Abbildung 2.3 sind Faktoren als Einflussgrößen gekennzeichnet. Die übrigen Parameter sind Störgrößen, die im besten Fall eliminiert oder konstant gehalten werden können (Ghazi 2019). Jeder Faktor wird über verschiedene Stufen hinweg untersucht und soll auf mindestens zwei unterschiedlichen getestet werden (Wieczorek 2009). Eine Stufe repräsentiert dabei einen spezifischen Wert oder Zustand des Faktors (Wieczorek 2009). Zum Beispiel kann ein Faktor die Drehzahl eines Bohrers sein. Verschiedene Stufen hätten dann die Ausprägung 500, 1500 und 3000 min^{-1}. Die Auswirkung der Faktoren auf die Qualitätsmerkmale ist abhängig von den Abständen der Ausprägungen auf den Stufen und sollte möglichst groß gewählt werden (Ghazi 2019). Die Auswahl geeigneter Stufen für die Faktoren bildet die Grundlage für die Erstellung eines SVP. Ein Versuchsplan definiert, welche Kombinationen von Faktoren in bestimmten Stufen in den Versuchen untersucht werden sollen (Siebertz et al. 2017). Die Durchführung einer bestimmten Kombination an Faktoren in einem Versuch wird Versuchslauf genannt. Die Versuchsläufe können zum Beispiel in einer Tabelle dokumentiert werden (Siebertz et al. 2017).

Die Auswirkungen von Faktoren auf Zielgrößen werden in der SVP Effekte genannt (Ghazi 2019). Innerhalb der Effekte wird zwischen den Haupteffekten und den Wechselwirkungen unterschieden. Die Haupteffekte sind die direkten Auswirkungen eines Faktors auf die Zielgrößen (Ghazi 2019). Von einer Wechselwirkung wird gesprochen, wenn ein Effekt auch von der Einstellung eines weiteren Faktors abhängig ist (Ghazi 2019). Effekte für ausgesuchte Zielgrößen werden berechnet, indem die Differenz von zwei Mittelwerten bestimmt wird (Siebertz et al. 2017). Von einem Mittelwert der Zielgröße, wenn ein Faktor auf einer Stufe ist, wird der Mittelwert der Zielgröße, wenn der Faktor auf einer anderen Stufe ist, abgezogen (Siebertz et al. 2017). Sowohl Haupteffekte als auch Wechselwirkungen können in einem Effekt-Diagramm standardisiert dargestellt werden, wie Siebertz et al. (2017) erklärt. Dabei wird der Faktor mit seinen Faktorstufen Dimensionslos auf der horizontalen Achse und die Zielgröße auf der vertikalen Achse dargestellt. Die Autoren führen aus, dass die Ausprägungen der Zielgrößen in den Faktorstufen verbunden werden und die Steigung der Verbindungslinie den Effekt kennzeichnet.

Es existieren unterschiedliche Ansätze, um systematisch im SVP vorzugehen und die Versuchsläufe zu bestimmen. Ein klassischer und einfacher Ansatz ist der vollfaktorielle Versuchsplan, der alle möglichen Kombinationen der Faktoren in den unterschiedlichen Stufen testet (Schiefer und Schiefer 2018). Dies hat den Vorteil, dass sowohl Haupteffekte als auch Wechselwirkungen zwischen allen Faktoren vollständig analysiert werden können. Der große Nachteil des vollfaktoriellen Versuchsplans ist die erforderliche Anzahl an Experimenten, die durchgeführt werden muss. Diese kann mit der Gleichung $(\text{Anzahl der Stufen})^{\text{Anzahl der Faktoren}}$ berechnet werden (Siebertz et al. 2017). Aufgrund des exponentiellen Wachstums erweisen sich vollfaktorielle Versuchspläne als zu rechenaufwändig und werden daher in der Praxis oft nicht angewandt. Um den Aufwand zu reduzieren, stellen teilfaktorielle Versuchspläne eine effiziente Alternative dar. Diese testen nur eine bestimmte Auswahl an Kombinationen der Faktoren und Stufen, sodass trotzdem noch aussagekräftige Ergebnisse erschlossen werden können (Siebertz et al. 2017). Ein Beispiel für einen teilfaktoriellen Versuchsplan ist der Plackett-Burmann-Versuchsplan (Ghazi 2019). Bei den teilfaktoriellen Versuchsplänen ist zu beachten, dass es zu Vermischungen von Haupteffekten und Wechselwirkungen kommen kann. Deshalb gibt es einige Computerprogramme oder Pakete in Programmiersprachen, die bei der Erstellung und Auswertung von teilfaktoriellen Versuchsplänen unterstützen (Siebertz et al. 2017). Unterstützend für die Anwendung von SVP mit Computerprogrammen können Anforderungen sein, die erfüllt werden müssen.

Gemäß Rupp und Pohl (2021) wird der Begriff der Anforderung als die dokumentierte Darstellung eines Bedürfnisses, einer Fähigkeit oder einer Eigenschaft definiert. Zudem sagt die Quelle aus, dass ein System die Fähigkeiten und Eigenschaften erfüllen muss. Anforderungen können in funktionale Anforderungen, Qualitätsanforderungen und Randbedingungen eingeteilt werden. Funktionale Anforderungen sind Vorgaben für das Ergebnis oder das Verhalten, das eine Funktion im System erbringen soll (Rupp und Pohl 2021). Qualitätsanforderungen legen fest, wie gut ein System in einem bestimmten Bereich sein muss (Rupp und Pohl 2021). Die Autoren nennen Performance, Verfügbarkeit, Zuverlässigkeit oder Skalierbarkeit als Beispiele für diese Bereiche. Randbedingungen werden nicht umgesetzt, sondern schränken den Lösungsraum zur Umsetzung der funktionalen und Qualitätsanforderungen ein (Rupp und Pohl 2021).

Die statistischen Grundlagen stellen die Basis für Experimentierpläne, Auswertungen und einen Überblick über die Daten dar. Für einen Überblick über den KDD-Prozess gibt es in der Literatur verschiedene Vorgehensmodelle, die im nächsten Abschnitt vorgestellt werden.

2.1.3 Vorgehensmodelle zur Wissensentdeckung in Datenbanken

Für den KDD-Prozess gibt es in der Literatur eine Vielzahl an Vorgehensmodellen, die den Prozess aus unterschiedlichen Sichtweisen abbilden (Mariscal et al. 2010). Diese Vorgehensmodelle eignen sich gut, um den gesamten KDD-Prozess zu überblicken, und leiten die Durchführung der Wissensentdeckung Schritt für Schritt an. In diesem Abschnitt werden anhand des Vorgehensmodells von Fayyad et al. (1996a) der Prozess und die Ziele von KDD erläutert. Die Verwendung dieses Modells an dieser Stelle erfolgt, da es den Fokus auf die technische Vorgehensweise richtet (Cleve und Lämmel 2024). Zusätzlich wird die Datentransformation in einem eigenen Schritt behandelt, was einen

Vorteil mit Blick auf das Ziel dieser Arbeit bietet.

KDD ist ein nichttrivialer Prozess zur Identifizierung von Mustern in Daten, die neu, valide, nützlich und verständlich sein sollten (Fayyad et al. 1996b). Diese Muster sollen dem Ziel von KDD dienen, wie der Name bereits vermuten lässt, die Generierung von nützlichem Wissen aus großen Datenmengen zu ermöglichen (Fayyad et al. 1996a). Dieser Prozess wird nach Brachman und Anand (1994) durch ein hohes Maß an Interaktivität ausgezeichnet. Vorgehensmodelle geben deswegen nur Rahmenbedingungen, ohne den Anwender im Handeln einzuschränken. Der KDD-Prozess ist interdisziplinär und umfasst nach Fayyad et al. (1996a, S. 39) die Disziplinen „[...] maschinelles Lernen, Mustererkennung, Datenbanken, Statistik, Künstliche Intelligenz, Wissenserwerb für Expertensysteme, Datenvisualisierung und Hochleistungsrechnen." In diesem Prozess können verschiedene Schritte identifiziert werden (Ester und Sander 2000). In der folgenden Abbildung 2.4 werden die Schritte im Vorgehensmodell nach Fayyad et al. (1996a) dargestellt.

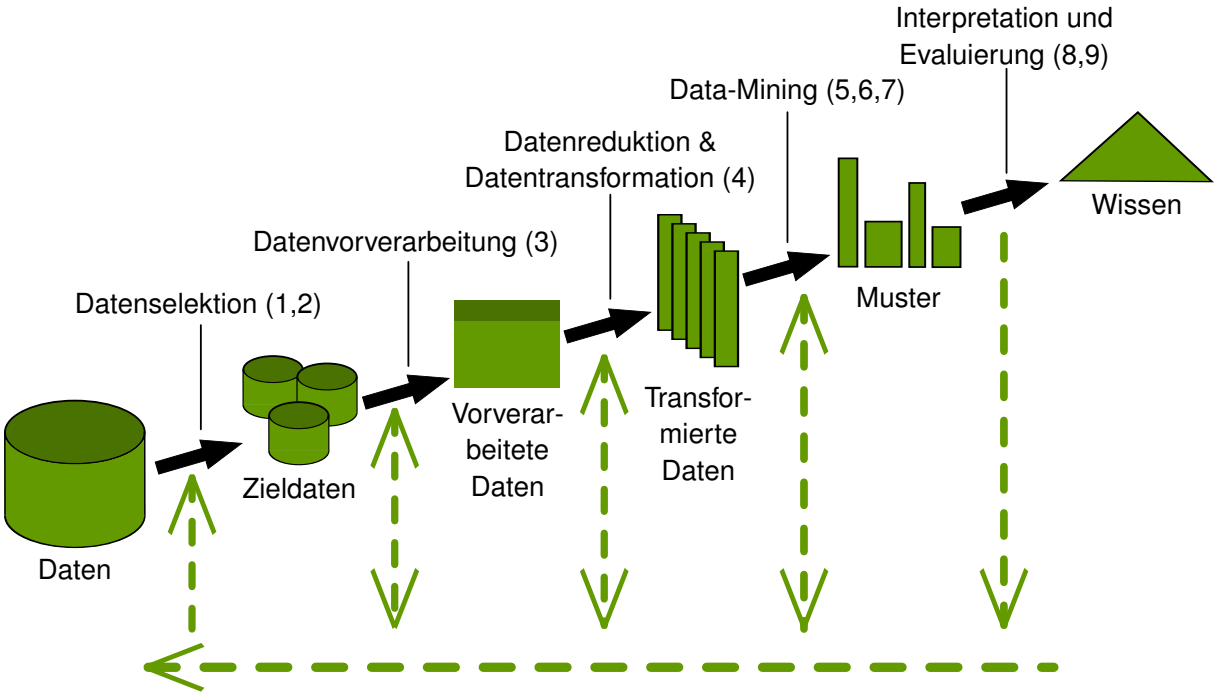

Abbildung 2.4: Vorgehensmodell des KDD-Prozesses angelehnt an Fayyad et al. (1996a, S. 41) und Fayyad et al. (1996b, S. 29)

In Abbildung 2.4 werden die Schritte im KDD-Prozess dargestellt. Gemäß Fayyad et al. (1996a) wird dieser Prozess als iterativ beschrieben, da nach jedem Schritt des Prozesses eine Rückkehr zu einem beliebigen früheren Schritt möglich ist. Dies wird im Modell mit den Pfeilen angedeutet. Fayyad et al. (1996a) stellt im Text neun Schritte im KDD-Prozess heraus, die im Folgenden den fünf übergeordneten Phasen aus dem Modell zugeteilt werden. Die fünf Phasen sorgen dafür, dass die Daten im Prozess den jeweils nächsten Reifegrad erreichen. Sie heißen Datenselektion, Datenvorverarbeitung, Datentransformation, Data-Mining und Interpretation & Validierung.

Datenselektion

Der erste Schritt behandelt die Daten noch nicht direkt, in ihm werden die Grundlagen für den KDD-Prozess geschaffen. Dafür wird das nötige Vorwissen erlangt, sodass die Anwendungsdomäne verstanden wird (Fayyad et al. 1996a). Des Weiteren ist es unerlässlich, das Ziel aus Sicht der Anwendung zu definieren (Ester und Sander 2000).

Im zweiten Schritt werden die Daten dafür ausgewählt, die für das Ziel als notwendig erachtet werden (Ester und Sander 2000). Es besteht die Möglichkeit, ausgewählte Datensätze oder Teile eines Datensatzes zu exportieren (Fayyad et al. 1996a). Die Grunddatenmenge, aus der Daten exportiert werden können, setzt sich aus den vorhandenen Datenquellen des Anwenders oder des Kunden sowie aus externen Daten zusammen, die beschafft werden können (Cleve und Lämmel 2024). In jedem Datensatz können viele Merkmale sein, die für das Ziel irrelevant oder redundant sind (Han et al. 2023). Eine anfängliche Datenselektion kann deshalb noch viele Merkmale beinhalten, die überflüssig sind und erst durch die Iterationen im KDD-Prozess herausgefiltert werden. Insgesamt sind alle verfügbaren Daten in der Datenselektion in einen (komprimierten) Zieldatenbestand überführt worden.

Datenvorverarbeitung

Der dritte Schritt im KDD-Vorgehensmodell nach Fayyad et al. (1996a) ist Datenvorverarbeitung und die Datenbereinigung. Ziel dieses Schrittes ist es, die Rohdaten so aufzubereiten, dass sie für die anschließende Data-Mining-Analyse geeignet sind (Fayyad et al. 1996a). Dazu gehören grundlegende Operationen wie das Entfernen von Rauschen, wenn dies angemessen erscheint, sowie der Umgang mit fehlenden Werten in den Daten (Mariscal et al. 2010). Zudem können bei zeitabhängigen Daten Zeitsequenzinformationen berücksichtigt werden (Mariscal et al. 2010).

Die Notwendigkeit einer sorgfältigen Vorverarbeitung ergibt sich daraus, dass die Daten in ihrer Ursprungsform oft noch nicht für Data-Mining-Analysen geeignet sind (Cleve und Lämmel 2024). Daher ist es essenziell, die Qualität des Zieldatenbestands zu untersuchen und gegebenenfalls durch geeignete Verfahren zu verbessern (Cleve und Lämmel 2024). Beispielsweise wird angenommen, dass etwa fünf Prozent der Angaben in einem typischen Datensatz fehlerhaft sein können. Ein weiterer wesentlicher Aspekt ist die Kenntnis der Stärken und Schwächen des Datenbestandes. Nur so kann die Zuverlässigkeit und Korrektheit der Ergebnisse gewährleistet werden (Cleve und Lämmel 2024).

Datenreduktion & Datentransformation

Den vierten Schritt benennt Fayyad et al. (1996a) als die Datenreduktion und Projektion. In dem vorliegenden Modell kann dies dem vierten Schritt zugeordnet werden (vgl. Abbildung 2.4). Das Ziel besteht darin, nützliche Merkmale für die Data-Mining-Analyse zu identifizieren, die die Daten unter Berücksichtigung des angestrebten KDD-Ziels repräsentieren (Ester und Sander 2000). Dieser Pro-

zess erfolgt unter Anwendung von Dimensionsreduktions- oder Transformationsmethoden, welche dazu dienen, die effektive Anzahl an Variablen zu reduzieren oder invariante Repräsentationen der Daten zu ermitteln (Fayyad et al. 1996b).

In dieser Arbeit wird zwischen Datenreduktion und Datentransformation unterschieden, weshalb die Benennung in Abbildung 2.4 zu Fayyad et al. (1996a) und Fayyad et al. (1996b) abweicht. Laut Cleve und Lämmel (2024) wird in der Datentransformationsphase der Datenbestand in ein Datenbankschema transformiert, sodass dieser im nächsten Schritt analysiert werden kann. Darüber hinaus stellen die Autoren heraus, dass die Merkmale spezifischen Anforderungen der Data-Mining-Analyse unterliegen, was eine Transformation erforderlich macht. Im vorliegenden Kontext besteht die Möglichkeit, neue Merkmale oder Datensätze zu generieren (Cleve und Lämmel 2024). Insgesamt bildet die Datentransformation mit der Datenvorverarbeitung eine wichtige Grundlage für den Erfolg des gesamten KDD-Prozesses: Nur durch eine gründliche Bereinigung und Aufbereitung der Daten kann gewährleistet werden, dass das Data-Mining belastbare und aussagekräftige Ergebnisse liefert (Cleve und Lämmel 2024).

Data-Mining

In der Data-Mining-Phase werden die Data-Mining-Algorithmen angewendet, die Muster in den Daten erkennen (Ester und Sander 2000). Da diese Phase die angestrebten Ergebnisse produziert, wird der Gesamtprozess teilweise auch als Data-Mining bezeichnet (Fayyad et al. 1996a). Dadurch wird jedoch die Wichtigkeit der anderen Phasen unterschlagen, weshalb Fayyad et al. (1996a) eine klare Unterscheidung zwischen KDD und Data-Mining vornimmt und Data-Mining nur eine Phase des KDD-Vorgehensmodells zuteilt.

In Schritt fünf wird ausgehend vom Ziel des KDD-Prozesses eine Data-Mining-Aufgabe ausgesucht, die später angewendet wird (Mariscal et al. 2010). Zum Beispiel Klassifikation, Clustering, Regression oder Assoziationsanalyse. Diese Methoden können in die Gruppen des überwachten und unüberwachten Lernens eingeteilt werden (Han et al. 2023). Das überwachte Lernen benötigt Label, auch Zielvariablen genannt, im Datensatz, in denen reale Ergebnisse gespeichert sind, mit denen der Data-Mining-Algorithmus lernen kann (Han et al. 2023). Demnach werden Prognosen über vergleichbare Daten erstellt, für die das Label nicht vorhanden ist (Han et al. 2023). Dazu gehören die Klassifikation und die Regression. Demgegenüber ist beim unüberwachten Lernen kein Einsatz von Labels erforderlich (Han et al. 2023). Hier liegt die Intention vor, Gruppen und Regeln in den Daten zu identifizieren, ohne dass ein Ergebnis vorliegt (Han et al. 2023). Clustering und Assoziationsanalyse sind hier einzuordnen.

Im sechsten Schritt erfolgt gemäß Fayyad et al. (1996a) eine explorative Datenanalyse. Darüber hinaus wird der spezifische Data-Mining-Algorithmus festgelegt, um Muster zu finden (Mariscal et al. 2010). Bei der Selektion des Algorithmus sind insbesondere das Ziel des KDD-Prozesses sowie die Eigenschaften der Daten zu berücksichtigen (Ester und Sander 2000). Der Anwender muss sich der Frage bewusst sein, ob die Vorhersage oder das Modell, welches der Vorhersage zugrunde liegt, zum Ziel führt (Fayyad et al. 1996a).

Den siebten Schritt bezeichnet (Fayyad et al. 1996a) als Data-Mining und betont noch einmal, dass die Qualität der Ergebnisse dieses Schrittes maßgeblich von den vorherigen Schritten abhängig ist. In diesem Schritt werden Data-Mining-Algorithmen angewendet, um Muster in den Daten zu identifizieren (Mariscal et al. 2010). Die so identifizierten Muster werden anschließend in zuvor definierten Formen dargestellt (Mariscal et al. 2010). Diese sind z. B. als Entscheidungsbäume oder Cluster vorzufinden.

Interpretation und Evaluierung

Im achten Schritt werden die Ergebnisse des siebten Schritts, die entdeckten Muster, interpretiert und bewertet (Cleve und Lämmel 2024). Falls die Ergebnisse nicht aussagekräftig sind oder eine Verbesserung durch Anpassungen im vorherigen KDD-Prozess vermutet wird, wird zu einem der ersten sieben Schritte zurückgekehrt, um den KDD-Prozess von dort mit Anpassungen neu zu starten (Fayyad et al. 1996a). Eine potenzielle Visualisierung der gefundenen Muster, um die Verständlichkeit zu verbessern, wird auch in diesem Schritt vorgenommen (Fayyad et al. 1996a). In diesem Schritt wird aus den Informationen Wissen generiert, um eine Einordnung auf der Wissenstreppe nach North (2021) zu geben (vgl. Abbildung 2.1).

Cleve und Lämmel (2024) bewertet die gefundenen Muster mit dem Kriterium der Interessantheit, welches die Dimensionen Gültigkeit, Neuartigkeit, Nützlichkeit und Verständlichkeit bewertet. Oft sind gefundene Muster der Data-Mining-Analyse „[...] irrelevant, trivial, bedeutungslos oder bereits bekannt [...]“ (Cleve und Lämmel 2024, S. 10). Außerdem ist es möglich, dass Muster nicht nachvollziehbar und deshalb unbrauchbar sind (Cleve und Lämmel 2024).

Der neunte Schritt und gleichzeitig letzte Schritt des KDD-Prozessmodells von Fayyad et al. (1996a) beschreibt das Handeln, welches auf dem neu gewonnenen Wissen aus Schritt acht aufbaut. Das Wissen wird beispielsweise in Systeme und Prozesse integriert und kann für Interessierte dokumentiert werden. Es ist wichtig, das neu erlangte Wissen auf Konflikte mit bereits vorhandenem Wissen zu prüfen.

Nachdem das Vorgehensmodell von Fayyad et al. (1996a) im Detail vorgestellt wurde, wird zum Abschluss des Kapitels das Cross Industry Standard Process for Data-Mining (CRISP-DM) Modell von Wirth und Hipp (2000) der Vollständigkeit halber namentlich aufgeführt, weil es in der Fachliteratur und besonders in der produzierenden Industrie ebenso weit verbreitet ist (Mariscal et al. 2010). Dieses legt den Fokus auf die Anwendersicht des gesamten Prozesses der Wissensentdeckung in einem Unternehmen mit dem Ziel, ein Modell zu entwickeln, welches einen Mehrwert in Unternehmensprozessen einbringt. Die technische Umsetzung spielt eine untergeordnete Rolle im Vergleich zum KDD-Prozessmodell nach Fayyad et al. (1996a). Eine Gemeinsamkeit der beiden Modelle ist der iterative Charakter, der von Wirth und Hipp (2000) im CRISP-DM Vorgehensmodell aufgegriffen wird. Die Autoren stellen das Modell in Kreisform dar, um den zyklischen Charakter im Data Mining herauszustellen.

Im KDD-Prozess nimmt der Data-Mining-Schritt eine entscheidende Rolle ein, da in diesem Schritt die Muster aus den Daten extrahiert werden. Mariscal et al. (2010) stellen heraus, dass im Allge-

meinen in der Industrie und im Journalismus der Data-Mining-Begriff den gesamten KDD-Prozess abdeckt. Wegen dieser zentralen Rolle wird dieser Schritt im nächsten Abschnitt vertieft.

2.1.4 Data Mining

In diesem Abschnitt werden weitere Informationen und Begriffe zum Data Mining dargestellt, die im Verlauf der Arbeit benötigt werden. Eine Einordnung in den KDD-Prozess ist in Kapitel 2.1.3 mithilfe eines Vorgehensmodells vorgenommen worden. Data Mining ist ein multidisziplinäres Feld, das Ansätze aus der Statistik, dem maschinellen Lernen, der Mustererkennung, der Datenvisualisierung und vielen weiteren Bereichen nutzt (Han et al. 2023).

Wie im Vorgehensmodell von Fayyad et al. (1996a) beschrieben, wird zu Beginn des KDD-Prozesses ein Ziel festgelegt, welches im weiteren Verlauf der Arbeit als Data-Mining-Aufgabe bezeichnet wird, die im KDD-Prozess verfolgt wird. Grundlegende Data-Mining-Aufgaben sind zum Beispiel Klassifikationen, Regressionen, Clusteranalysen und das Erkennen von Assoziationsregeln, die dann im Kontext weiter spezifiziert werden (Cleve und Lämmel 2024). Ein Beispiel mit mehr Kontext ist die Vorhersage von Ausschussteilen anhand gegebener Prozessparameter in einer Klassifikation. Eine Klassifikation ist ein überwachtes Lernverfahren, bei dem Daten in vordefinierte Kategorien oder Klassen eingeteilt werden (Runkler 2015). Ziel ist es, aus vorhandenen Daten ein Modell zu erstellen, das neue Daten korrekt klassifiziert (Runkler 2015). Die Regression ist ebenfalls ein überwachtes Lernverfahren zur Modellierung von Beziehungen zwischen Variablen, wobei meist eine kontinuierliche Zielvariable vorhergesagt wird (Han et al. 2023). Die Clusteranalyse ist ein unüberwachtes Lernverfahren, das keine vorgegebenen Label benötigt und Objekte basierend auf ihren Ähnlichkeiten gruppiert (Runkler 2015). Die Assoziationsanalyse untersucht Beziehungen zwischen Elementen innerhalb eines Datensatzes und identifiziert häufig auftretende Muster (Cleve und Lämmel 2024). Die Assoziationsregeln sind ähnlich zu Regeln, die in einer Klassifikation aufgestellt werden, beschränken sich aber nicht nur auf die Zielvariable (Cleve und Lämmel 2024).

Um Data-Mining-Aufgaben zu bearbeiten, werden Data-Mining-Algorithmen benutzt (Larose und Larose 2014). Für eine Data-Mining-Aufgabe gibt es verschiedene Data-Mining-Algorithmen, die diese Aufgabe auf unterschiedliche Weise bearbeiten und auch leicht unterschiedliche Ergebnisse erarbeiten (Larose und Larose 2014). Die Data-Mining-Aufgaben und -Algorithmen stellen unterschiedliche Voraussetzungen an die Eigenschaften der Daten (García et al. 2015). Fayyad et al. (1996a) heben hervor, dass die Ergebnisse der Algorithmen neben den schon erwähnten Mustern auch Modelle sein können. Die Autoren merken dabei an, dass Muster Komponenten von Modellen sind, wie zum Beispiel eine Regel in einem Klassifikationsmodell.

Um die Ergebnisse zu bewerten und einen Vergleich untereinander zu ermöglichen, können diese Modelle bewertet werden. Bei den Data-Mining-Algorithmen, die den überwachten Lernverfahren zugeordnet werden können, wird dies mit dem Vergleich von Vorhersagen und echten Werten der Zielvariable gemacht (Larose und Larose 2014). Die Datenpunkte mit den echten Werten werden dabei vorher bei der Modellerstellung des Data-Mining-Algorithmus nicht berücksichtigt (Larose und Larose 2014). Eine Möglichkeit für die Bewertung ist die Aufteilung des Datensatzes in zwei Datensätze mit einem sogenannten Train-Test-Split (Tan et al. 2021). In der Praxis wird der Datensatz am

häufigsten durch Zufall mit einer gleichmäßigen Verteilung aufgeteilt, weil diese Methode effizient und einfach zu implementieren ist (Reitermanova 2010). Daraus resultieren der Trainingsdatensatz, auf dem der Data-Mining-Algorithmus lernt und ein Modell erstellt, und der Testdatensatz, von dem die realen Zielvariablen getrennt sind (Larose und Larose 2014). Reitermanova (2010) haben 25% der Daten dem Testdatensatz zugeordnet. Das erlernte Modell wird auf den Testdatensatz angewendet, sodass prognostizierte Werte für den Testdatensatz erschaffen werden, die mit den zuvor getrennten Zielvariablen verglichen werden können (Larose und Larose 2014). Einige Berechnungsvorschriften, die für den Vergleich verwendet werden, sind in Kapitel 2.1.2 aufgelistet und berechnen Kennzahlen, die die Güte des Modells beschreiben. Das sind zum Beispiel der RMSE, der MAE, der MAPE, und der R^2-Wert für die Regression oder die Genauigkeit, die Sensitivität, die Präzision und der F1-Score für die Klassifikation.

Wenn ein Modell aus einem Data-Mining-Algorithmus zu gut an die Daten angepasst ist und auch das Rauschen und die Ausreißer erfasst, ist die Generalisierungsfähigkeit nicht gegeben und es wird von Überanpassung (Overfitting) gesprochen (Runkler 2015). Dies führt dazu, dass das Modell auf neuen, unbekannten Daten schlecht abschneidet, da es den zugrundeliegenden Zusammenhang schlecht abbildet, weil es zu stark an die spezifischen Trainingsdaten angepasst ist (Runkler 2015). Durch die Aufteilung des Datensatzes und der anschließenden Bewertung kann Overfitting deswegen entdeckt werden.

Abschließend muss in diesem Abschnitt erwähnt werden, dass das Finden von Wissen in den Daten mithilfe von Mustern und Modellen nicht garantiert ist (Larose und Larose 2014). Damit die Data-Mining-Algorithmen angewendet werden können, muss der Datensatz, je nach Algorithmus, bestimmte Voraussetzungen erfüllen. Die Vorbereitung des Datensatzes für diese Anforderungen wird in der Datenvorverarbeitung erfüllt, die im nächsten Abschnitt vorgestellt wird.

2.2 Datenvorverarbeitung

Nachdem in Kapitel 2.1.3 die Datenvorverarbeitung als Teilschritt des KDD-Vorgehensmodells kurz behandelt wurde, wird diese im Folgenden vertieft. Die Schritte der Datenvorverarbeitung, die sich mit der Behandlung fehlender Werte und der Transformation von Daten befassen, werden jeweils in einem gesonderten Unterkapitel behandelt und in diesem Abschnitt nur der Vollständigkeit halber aufgezählt.

Obwohl Fayyad et al. (1996a) nur den dritten Schritt der Datenvorverarbeitung zuordnet, wird in diesem Kapitel der Begriff weiter gefasst und alles zwischen Datenselektion und Data-Mining wird zur Datenvorverarbeitung gezählt (vgl. Abbildung 2.4). Diese Einteilung zur Datenvorverarbeitung nimmt zum Beispiel auch García et al. (2016) vor. Der Vollständigkeit halber soll erwähnt werden, dass andere Quellen, wie beispielsweise Han et al. (2023), die Datenselektion auch noch zur Datenvorverarbeitung dazu zählen.

Die Datenvorverarbeitung bezieht sich auf alle Techniken, die vor der Anwendung des Data-Mining eingesetzt werden (García et al. 2016). Das Hauptziel ist es, die Daten so aufzubereiten, dass sie

für eine Data-Mining-Analyse geeignet sind (Fayyad et al. 1996a). Außerdem stellt Borrohou et al. (2025) heraus, dass ohne eine Datenvorverarbeitung die Rohdaten nicht nur schwerer zu analysieren sind, sondern die Ergebnisse der Data-Mining-Analyse auch im Endeffekt falsche Erkenntnisse herbeiführen. Vorteile einer Datenvorverarbeitung sind nach García et al. (2015) die Ermöglichung, Daten optimal auf den Data-Mining-Algorithmus zuzuschneiden, die Datenmenge und Analysezeit zu reduzieren sowie die Vorhersagegenauigkeit und -effektivität zu steigern. Zudem sagt die Quelle aus, dass das Verständnis der Daten verbessert wird und dadurch das Erkennen von Mustern und Zusammenhängen erleichtert wird. Die Datenvorverarbeitung kann benötigt werden, weil Rohdaten unvollständig sind, Inkonsistenzen oder Redundanzen beinhalten, Rauschen aufweisen oder Dopplungen enthalten (Çetin und Yıldız 2022). Sie möchte dies beseitigen und setzt sich aus den Schritten Datenintegration, Datenbereinigung, Behandlung von fehlenden Werten, Rauschbeseitigung, Datenreduktion und Datentransformation zusammen (García et al. 2016).

In Abhängigkeit der vorliegenden Daten kann zum Start der Datenvorverarbeitung eine Datenintegration erforderlich sein. Im Rahmen dieses Schrittes erfolgt die Zusammenführung von Daten aus diversen Quellen und in unterschiedlichen Formaten (Runkler 2015). Das Ziel besteht darin, eine zusammenhängende Datenbasis für die Datenanalyse zu gewährleisten. In diesem Zusammenhang sollen Merkmale abgeglichen und Inkonsistenzen in den Daten entfernt werden (Runkler 2015).

Da Daten aus der echten Welt Unreinheiten, wie fehlende Werte, Rauschen oder falsche Werte beinhalten, müssen diese Fehler in dem Schritt Datenbereinigung entfernt werden (García et al. 2015). Die Behandlung von fehlenden Werten erfolgt durch die Löschung der fehlenden Werte oder durch das Ersetzen der fehlenden Werte mit einem anderen Wert (Cleve und Lämmel 2024). Çetin und Yıldız (2022) erklärt, dass Rauschen durch fehlerhafte Messungen oder menschliche Fehler entsteht. Dieses wird durch sogenannte Filter behoben, wobei oft verschiedene Filter angewendet werden, um die Vorteile dieser zu kombinieren (Çetin und Yıldız 2022). Interessant ist auch, dass verschiedene Algorithmen unterschiedlich stark auf Rauschen in den Daten reagieren. Entscheidungsbäume weisen im Vergleich zu Algorithmen, die Distanzmessungen anwenden, wie beispielsweise der K-Nearest-Neighbor (KNN)-Algorithmus, eine geringere Anfälligkeit auf (García et al. 2015). Zu falschen Werten werden jegliche anderen Fehler, wie zum Beispiel Ausreißer, Rechtschreibfehler oder falsche Formate, gezählt. Die Entfernung dieser ist sehr individuell und wird an dieser Stelle nicht weiter ausgeführt (Runkler 2015).

Im Zuge der Datenreduktion erfolgt eine Selektion der zu analysierenden Daten, wobei der Fokus auf der Reduktion der Datenmenge liegt, ohne dabei signifikante Informationen aus den Daten zu verlieren (Han et al. 2023). Zwei Ansätze für eine Datenreduktion sind die Merkmalsauswahl und Instanzauswahl (García et al. 2015). In der Merkmalsauswahl werden irrelevante oder doppelte Merkmale entfernt, bis ein minimal möglicher Satz an Merkmalen übrig ist (García et al. 2015). Dadurch wird das Verständnis des extrahierten Musters aus der Data-Mining-Phase erleichtert und die Rechenzeit dieser Phase verkürzt (García et al. 2015). In der Instanzauswahl wird eine Menge an Datentupeln aus dem Datensatz ausgesucht. Bestimmte Regeln oder Heuristiken können zum Einsatz kommen, um einen kleineren, für den Data-Mining-Algorithmus verbesserten Datensatz zu erschaffen (García et al. 2015). Das sogenannte Sampling ist auch eine Art der Instanzauswahl. Hierbei werden vom Data-Mining-Algorithmus zufällig einige Datentupel vom kompletten Datensatz

ausgesucht, um eine interne Validierung zu ermöglichen und somit Overfitting vorzubeugen (García et al. 2015).

Die Datentransformation ist ein Überbegriff für verschiedene Transformationen, die auf die Daten angewendet werden, um diese in eine andere Form zu bringen, die für die Analyse vorausgesetzt wird oder zum besseren Verständnis der Daten beiträgt (Cleve und Lämmel 2024). Dabei wird für gewöhnlich auf mathematische Formeln zurückgegriffen und zusätzlich können in der Datentransformation auch neue Merkmale erschaffen werden (García et al. 2015).

Die zuvor genannten Schritte der Datenvorverarbeitung dienen der Verbesserung der Datenlage, um potenziell belastbares Wissen aus Daten mit dem KDD-Prozess zu gewinnen. Die Anwendung der folgenden Schritte der Datenvorverarbeitung, die Behandlung von fehlenden Werten und die Datentransformation sind von noch entscheidender Bedeutung, da die Anwendung dieser eine Voraussetzung für einige Data-Mining-Algorithmen ist.

2.2.1 Fehlende Werte

Fehlende Werte sind einer der am häufigsten auftretenden Qualitätsmängel in Datensätzen (Jäger et al. 2021). Die Behandlung dieser ist demnach ein bedeutender Schritt in der Datenvorverarbeitung. In dieser kann sie der Datenbereinigung zugeordnet werden. Da zusätzlich viele Datenanalysemethoden mit fehlenden Werten im Datensatz nicht angewendet werden können, steigt dadurch die Wichtigkeit der Behandlung von fehlenden Werten (Rockel 2022). In diesem Abschnitt werden zuerst allgemeine Begrifflichkeiten geklärt, bevor die Behandlung von fehlenden Werten thematisiert wird. Spezifische Methoden für die Behandlung von fehlenden Werten, die verschiedene Behandlungsarten abdecken, werden im nächsten Abschnitt separat behandelt.

In Datensätzen können fehlende Werte in verschiedenen Ausprägungen auftreten. Han et al. (2023) merken an, dass der Eintrag entweder leer sein oder eine Art Platzhalter enthalten kann. Beispiele, die die Autoren für Platzhalter aufführen, sind *NULL*, *?* oder *don't know*. Ihre Präsenz ist auf die mangelnde Beobachtung von Daten zurückzuführen (Little und Rubin 2020). Das kann verschiedene Ursachen haben, die von technischen Ausfällen von Sensoren bis hin zu nicht beantworteten Fragen auf Fragebögen reichen (Little und Rubin 2020). Die Gründe, die hinter den Ursachen stehen, können zufällig sein oder bedingt von anderen Faktoren (Little und Rubin 2020). Deswegen definiert Little und Rubin (2020) fehlende Werte als unbeobachtete Werte, die für die Analyse aussagekräftig wären, wenn sie beobachtet würden; mit anderen Worten: In einem fehlenden Wert ist eine bedeutende Aussage versteckt.

Fehlende Werte besitzen Eigenschaften, die den Überkategorien Ausfallursachen und Ausfallmechanismus zugeordnet werden können (Bankhofer 1995). Lang und Little (2018) betrachtet zusätzlich noch das Ausfallmuster und die Ausfallrate. Der Umgang mit den fehlenden Werten in der Datenvorverarbeitung hängt von diesen Eigenschaften ab. Es existiert eine große Anzahl verschiedener Ausfallursachen. An dieser Stelle werden die Verwendung mehrerer Sekundärquellen und das mangelhafte Untersuchungsdesign von Bankhofer (1995) exemplarisch vorgestellt. Im Falle der Zusammenführung mehrerer Sekundärquellen in der Datenintegration und der Abbildung von Objekten in

unterschiedlicher Detailtiefe in den betreffenden Quellen können bei der Zusammenführung Lücken im Datensatz entstehen, da eine Merkmalsausprägung nicht immer in beiden Quellen vorhanden ist (Bankhofer 1995). Ebenso können nicht alle Merkmale von allen Objekten ausgefüllt werden, sofern das Untersuchungsdesign mangelhaft ist. Dies resultiert in leeren Einträgen und demzufolge fehlenden Werten. Eine Datenbank eines Metallbauers, in der verfügbare Schrauben und deren Eigenschaften hinterlegt sind und in der diese Ausfallursache vorliegt, müsste bei einem Bolzen das Feld für die Gewindeart leer lassen und würde folglich einen fehlenden Wert in der Datenbank haben. Weitere Ausfallursachen stehen meist im Zusammenhang mit manuellen Erhebungen und sind an dieser Stelle nicht weiter relevant (Bankhofer 1995).

Der Ausfallmechanismus beschreibt, inwiefern das Fehlen der Werte von der vorhandenen Datenmatrix abhängt (Rockel 2022). Es wird zwischen Missing Completely At Random (MCAR), Missing At Random (MAR) und Missing Not At Random (MNAR) unterschieden (Rockel 2022). Wie der Name bereits vermuten lässt, beschreibt MCAR einen vollständig zufälligen Ausfallmechanismus, bei dem keine Abhängigkeiten vom Fehlen des Werts zu beobachteten Werten bestehen (Rockel 2022). Der MAR-Ausfallmechanismus sagt aus, dass das Fehlen der Werte von den beobachteten Werten abhängt (Rockel 2022). Hängt das Fehlen der Werte jedoch von den fehlenden Werten selbst ab, wird von einem MNAR-Ausfallmechanismus gesprochen (Rockel 2022). Das Ausfallmuster beschreibt, an welchen Stellen in der Datenmatrix fehlende Werte auftreten (Enders und Little 2010). Wenn nur in einem Merkmal Werte fehlen, wird das Muster als univariates Ausfallmuster bezeichnet (Enders und Little 2010). Sobald in mehr als einem Merkmal Werte fehlen, wird es als multivariates Ausfallmuster angesehen (Van Buuren 2018). Es gibt einige Unterscheidungen innerhalb der multivariaten Ausfallmuster. Am häufigsten tritt in der Praxis jedoch auf, dass Werte augenscheinlich willkürlich über die gesamte Datenmatrix verteilt fehlen (Enders und Little 2010). Die Ausfallrate beschreibt die Anzahl der fehlenden Werte. Diese kann durch verschiedene Kennzahlen ausgedrückt werden. Am einfachsten ist eine Prozentangabe, wie viele Werte im Vergleich zu allen möglichen beobachteten Werten fehlen (Lang und Little 2018). Es existieren weitere Kennzahlen, wie die Kovarianzabdeckung oder der Anteil fehlender Informationen, die zusätzlich berücksichtigen, inwiefern mithilfe der beobachteten Daten fehlende Informationen wiederhergestellt werden können (Lang und Little 2018).

Die Behandlung von fehlenden Werten kann in verschiedene Strategien eingeteilt werden, wobei sich die Strategien nicht gegenseitig ausschließen (Bankhofer 1995). Zudem sind in der Literatur die Strategien nicht vollständig einheitlich aufgeteilt und unterscheiden sich je nach Autor, wie im Vergleich von Bankhofer (1995) zu Little und Rubin (2020) zu erkennen ist. In dieser Arbeit werden die Verfahren in die Strategien Eliminierungsverfahren, Imputationsverfahren, Anpassung von Analyseverfahren und Sensitivitätsbetrachtung eingeteilt.

„Bei den Eliminierungsverfahren werden Objekte oder Merkmale mit fehlenden Werten von der Analyse ausgeschlossen“ (Rockel 2022, S. 21). Dabei gibt es Unterschiede, ob alle vollständigen oder alle verfügbaren Objekte (Datentupel) oder Merkmale betrachtet werden (Rockel 2022). Wenn nur vollständige Datentupel oder Merkmale betrachtet werden, reicht ein fehlender Wert aus, um das gesamte Objekt oder Merkmal zu eliminieren (Rockel 2022). Bei verfügbaren Datentupeln oder Merkmalen würde aus der Betrachtung des Datentupels oder des Merkmals nur der fehlende Wert ent-

fernt werden (Rockel 2022). In der wissenschaftlichen Literatur wird die Eliminierung von Merkmalen vergleichsweise selten behandelt, da grundsätzlich alle erhobenen Merkmale in die Datenanalyse einfließen sollen, wenngleich dies nicht immer im vollen Umfang realisierbar ist (Bankhofer 1995).

Imputationsverfahren setzen Werte in die Datenmatrix an Stelle der fehlenden Werte ein, sodass diese wieder gefüllt wird (Rockel 2022). Die eingesetzten Werte werden mit unterschiedlichen Berechnungsvorschriften erstellt, die in der Komplexität variieren. Diese Berechnungsvorschriften greifen auf die vorhandenen Werte, sowohl vom gleichen Merkmal als auch von anderen Merkmalen, zurück (Rockel 2022). Ein großer Vorteil gegenüber den Eliminierungsverfahren ist, dass alle beobachteten Werte erhalten bleiben und keine Informationen verloren gehen. Außerdem können durch Imputationsverfahren implizite Informationen wiederhergestellt werden (García et al. 2015). Die Imputation bedarf jedoch mehr Vorbereitung als die Eliminierung und kann unter Umständen statistisches Verhalten, statistische Kennzahlen und Beziehungen zwischen Daten verzerren (Little und Rubin 2020). Besonders bei systematischem Fehlen der Werte, wie zum Beispiel bei einem MNAR-Ausfallmechanismus, sollte die Imputation deshalb bedacht angewendet werden, um die Aussagekraft, warum diese Werte fehlen, nicht zu unterschlagen (Little und Rubin 2020). Zu den Imputationsverfahren zählen nach García et al. (2015) auch die Parameterschätzverfahren. Diese sind für Bankhofer (1995) Methoden zur Schätzung bestimmter Parameter wie Mittelwerte und Varianzen auf Basis unvollständiger Datenmatrizen. Für den Autor liegt der Fokus dabei auf Verfahren, die direkt Verteilungsparameter schätzen, wie zum Beispiel dem Expectation-Maximization-Algorithmus.

Bei der Anpassung von Analyseverfahren wird der Datensatz nicht verändert, sondern das Analyseverfahren wird so angepasst, dass dies auch trotz fehlender Werte funktioniert (Bankhofer 1995). Dabei ist die Anpassung individuell vom Verfahren abhängig. In dieser Arbeit wird diese Strategie nicht weiter verfolgt und wird nur der Vollständigkeit halber aufgezählt.

Das Verfahren der Sensitivitätsbetrachtung behandelt fehlende Werte in einem Datensatz mit verschiedenen Ansätzen, sodass mehrere unterschiedliche Datensätze entstehen (Bankhofer 1995). Auf diese Weise ergeben sich schlussendlich unterschiedliche Analyseergebnisse, die daraufhin in Anbetracht der übergeordneten Fragestellung abgewogen werden (Bankhofer 1995). Da dieses Verfahren kein eindeutiges Ergebnis liefert, wird dieses ebenso für den Experimentierplan nicht weiter berücksichtigt.

Abschließend wird erwähnt, dass es auch Ansätze gibt, bei denen fehlende Werte gar nicht in der Datenvorverarbeitung behandelt werden. In diesem Fall wird darauf vertraut, dass die Data-Mining-Algorithmen trotz fehlender Werte funktionieren oder diese eigenständig beseitigen (García et al. 2015). Somit ist es möglich, die Verfahren in die drei Kategorien *Löschen*, *Auffüllen* und *nichts unternehmen* einzuteilen. Im nächsten Abschnitt werden einige spezifische Methoden zur Behandlung fehlender Werte aus verschiedenen Strategien vorgestellt.

2.2.2 Methoden zur Behandlung von fehlenden Werten

Dieser Abschnitt stellt Methoden zur Behandlung fehlender Werte vor, die für den Vergleich im Experimentierplan verwendet werden können. Dieser Abschnitt beschäftigt sich mit der prinzipiellen

Funktionsweise der Methoden und geht nicht auf die Implementierung der Methoden in Datenverarbeitungsprogrammen ein.

Wie in Kapitel 2.2.1 erläutert, gibt es bei den Eliminierungsverfahren jeweils die Möglichkeit, die vollständigen oder verfügbaren Datentupel oder Merkmale zu betrachten. Insgesamt gibt es daher vier verschiedene Methoden, die den Eliminierungsverfahren zugeordnet sind. An dieser Stelle werden die zwei Methoden vorgestellt, die jeweils alle Datentupel oder Merkmale löschen, die nicht vollständig sind. Das Löschen von Datentupeln hat den Nachteil, dass selbst bei einem kleinen Anteil an fehlenden Werten, die zufällig auf jedes Merkmal verteilt sind, die Gesamtanzahl der Datentupel schnell reduziert wird . Bei 5% fehlenden Werten und 12 Merkmalen sind nur noch ungefähr 54% $(0,95^{12} = 0,54)$ der Datentupel vorhanden (Rockel 2022). Die Merkmalseliminierung ist zum Beispiel von Cho et al. (2022) dazu genutzt worden, Merkmale mit vielen fehlenden Daten zu löschen.

Methoden der Imputation greifen auf Techniken zurück, die auch in der Statistik oder bei Modellen in der Data-Mining-Phase des KDD-Prozesses zu finden sind (Lin und Tsai 2020). Zu den statistischen Techniken zählt die Berechnung der Lageparameter oder die stochastische Regression (Little und Rubin 2020). Zu den Modellen zählt zum Beispiel der KNN-Algorithmus oder der Random-Forest-Algorithmus (García et al. 2015).

Eine einfache Art der Imputation ist die Imputation mit Lageparametern (Bankhofer 1995). Zu den Lageparametern gehören der Mittelwert, der Median und der Modalwert (vgl. Kapitel 2.1.2). Diese werden mit den vorhandenen Werten eines Merkmals berechnet, bevor die fehlenden Werte mit dem Wert des ausgewählten Lageparameters ersetzt werden (Sun et al. 2023). Bei numerischen Werten wird der Mittelwert oder der Median verwendet, bei nominalen Werten der Modalwert (Sun et al. 2023). Bei ordinalen Werten können aufgrund der Ordnung der Median und der Modalwert als Lageparameter verwendet werden(vgl. Kapitel 2.1.1). Eine potenzielle Abhängigkeit zu anderen Merkmalen kann bei dieser Imputation nicht berücksichtigt werden.

Ein mathematisches Modell für die Imputation ist die Interpolation, die Runkler (2015) als besonders geeignet für Zeitreihendaten herausstellt. Eine einfache Art der Interpolation ist die lineare Interpolation. Diese wird innerhalb eines Merkmals auf die Lücken in den Daten angewendet. Eine Lücke in den Daten besteht aus mindestens einem fehlenden Wert, kann aber auch aus mehreren fehlenden Werten nacheinander bestehen (Junninen et al. 2004). Dabei wird eine Gerade zwischen den angrenzenden vorhandenen Daten der Lücke gezogen, mit der die fehlenden Werte berechnet und imputiert werden können (Junninen et al. 2004).

Mit den Data-Mining-Algorithmen, die im KDD-Prozess zur Erstellung von Vorhersagemodellen verwendet werden, können auch Modelle zur Imputation fehlender Werte erstellt werden (García et al. 2015). Diese können auch, abhängig vom Modell, nominale und metrische Daten vorhersagen (García et al. 2015). Die KNN-Imputation bedient sich des KNN-Algorithmus, der die K nächsten Nachbarn mithilfe einer Distanzmessung bestimmt (Troyanskaya et al. 2001). Die in der Literatur am häufigsten vertretene Distanzmessung ist die euklidische Norm (García et al. 2015). Die Anzahl K der Nachbarn muss vom Anwender der Methode gewählt werden (Sun et al. 2023). Von den K nächsten Nachbarn werden die Werte des Merkmals verwendet, in dem der fehlende Wert liegt. Aus diesen Werten wird ein gewichteter Durchschnittswert in Abhängigkeit vom Abstand zum fehlenden

Wert gebildet (Troyanskaya et al. 2001). Dieser Wert wird zur Imputation des fehlenden Wertes genutzt (Troyanskaya et al. 2001). Bei nominalen Werten wird der Modalwert imputiert (García et al. 2015).

Einige Methoden zur Behandlung fehlender Werte sind iterativ strukturiert. Zum Beispiel der Multivariate Imputation by Chained Equations-Algorithmus von Buuren und Groothuis-Oudshoorn (2011). In anderen Artikeln sowie in dieser Arbeit wird der Algorithmus auch als Multiple imputations by chained equations (MICE) bezeichnet (Seu et al. 2022). Der MICE-Algorithmus beginnt typischerweise mit einer einfachen Startimputation, wie der Mittelwertimputation, um initiale Werte und einen Platzhalter für die fehlenden Daten zu erzeugen (Seu et al. 2022). Anschließend werden Modelle für jedes Merkmal mit fehlenden Werten erstellt, wobei die übrigen Merkmale als Eingabedaten dienen (Seu et al. 2022). Diese Modelle, beispielsweise ein Regressionsmodell, können flexibel gewählt werden (Buuren und Groothuis-Oudshoorn 2011). Die Imputation erfolgt iterativ, wobei die zuletzt imputierten Werte in den folgenden Durchläufen berücksichtigt werden (Seu et al. 2022). Dieser Prozess wird so lange wiederholt, bis sich die imputierten Werte stabilisieren und eine konsistente Schätzung erreicht ist (Buuren und Groothuis-Oudshoorn 2011).

Eine weitere iterative Imputationsmethode ist MissForest von Stekhoven und Bühlmann (2012). Bei dieser Methode wird der Random-Forest-Algorithmus als Regressionsmodell für die Vorhersage der fehlenden Werte benutzt (Stekhoven und Bühlmann 2012). Diese Methode funktioniert ähnlich wie MICE, da die fehlenden Werte ebenfalls in Iterationen neu vorhergesagt werden, bis ein Abbruchkriterium erreicht ist (Stekhoven und Bühlmann 2012). Stekhoven und Bühlmann (2012) haben den Random-Forest-Algorithmus gewählt, da durch diesen eine Imputation auf gemischten Variablen mit metrischen und kategorischen Wertausprägungen ermöglicht wird. Außerdem erklären die Autoren, dass dieser Algorithmus unter Bedingungen wie hohen Dimensionen, komplexen Interaktionen in den Daten und nichtlinearen Datenstrukturen Vorteile gegenüber anderen Algorithmen aufweist. Der Random-Forest-Algorithmus kann auch ohne Iteration für eine Regression verwendet werden. In dem Fall werden die fehlenden Werte mit einer Regressionsimputation aufgefüllt.

Abschließend werden Methoden des Deep Learning zur Imputation der Vollständigkeit halber erwähnt. Diese sind im Vergleich zu bisher vorgestellten Methoden jünger, wie Sun et al. (2023) herausstellen. Deep Learning basiert auf künstlichen neuronalen Netzen, die aus miteinander verbundenen Eingabe- und Ausgabeeinheiten bestehen (Han et al. 2023). Während der Lernphase passt das Netz die Gewichte der Verbindungen an, um Eingaben so zu verarbeiten, dass korrekte Zielwerte vorhergesagt werden können (Han et al. 2023). Sun et al. (2023) stellen unter anderem Generative adversarial imputation nets (GAIN) und variational auto-encoders (VAE) als Imputationsmethoden vor, kommen aber zu dem Schluss, dass diese Methoden nicht so gute Ergebnisse erzielen wie herkömmliche Modellbasierte Imputationsmethoden, zum Beispiel MICE oder MissForest. Nur bei sehr großen Datenmengen ist es Dong et al. (2021) mit GAIN gelungen, bessere Ergebnisse in der Vorhersagegenauigkeit der fehlenden Werte zu erzielen.

Neben dem zuvor behandelten Thema der fehlenden Werte wird in dieser Arbeit auch die Datentransformation als weiterer Schritt der Datenvorverarbeitung untersucht. Im nächsten Unterkapitel werden deshalb zunächst die theoretischen Grundlagen der Datentransformation dargelegt.

2.2.3 Aufgaben der Datentransformation

Die Datentransformation wird im KDD-Vorgehensmodell von Fayyad et al. (1996a) nach der Datenvorverarbeitung eingeordnet (vgl. Abbildung 2.4 in Kapitel 2.1.3). Je nach Definition ist die Datentransformation auch Teil der Datenvorverarbeitung, zum Beispiel in García et al. (2015). Im Vergleich zu anderen Datenvorverarbeitungsschritten beinhaltet die Datentransformation viele verschiedene Aufgaben, die bearbeitet werden können (García et al. 2015). Für jede Datentransformationsaufgabe gibt es unterschiedliche Methoden (Han et al. 2023). Die Datentransformation kann auch als Werkzeug in anderen Schritten der Datenvorverarbeitung wiedergefunden werden (Borrohou et al. 2025). Dies wird in dieser Arbeit jedoch nicht weiter behandelt, da es um den Vergleich der Datentransformationsmethoden geht. In diesem Abschnitt werden zunächst die Ziele der Datentransformation sowie die entsprechenden Aufgaben vorgestellt, bevor im Anschluss die unterschiedlichen Methoden der Aufgaben vorgestellt werden.

Das Ziel einer geeigneten Datentransformation ist, den KDD-Prozess effizienter und die gefundenen Muster und Modelle verständlicher zu gestalten (Han et al. 2023). Im Gegensatz zur Behandlung fehlender Werte ist es üblich, eine Reihe von Datentransformationen auf einen Datensatz anzuwenden (Han et al. 2023). Cleve und Lämmel (2024) stellt zudem heraus, dass einige Transformationen an den Data-Mining-Algorithmus angepasst werden, während andere verfahrensunabhängig angewendet werden. García et al. (2015) zählen zu den Aufgaben der Datentransformation die Normalisierung, die Datenglättung, die Aggregation, die Merkmalskonstruktion und die Diskretisierung. Cleve und Lämmel (2024) zählen zusätzlich die Datencodierung auf. Außerdem verwenden diese Autoren den Überbegriff Skalierung, der die Normalisierung mit einschließt. Dies wird in dieser Arbeit so übernommen.

Bei der Skalierung werden die Werte in einem Datensatz auf einen beliebigen Wertebereich projiziert (Cleve und Lämmel 2024). Die Normalisierung zählt auch zur Skalierung, jedoch werden die Werte aller Merkmale auf einen bestimmten Wertebereich Skaliert (Cleve und Lämmel 2024). Typische Wertebereiche dafür sind [0,1] oder [-1,1] (García et al. 2015). Die Normalisierung wird angewendet, um die Größenunterschiede in der Merkmalsausprägung zu entfernen und einen einheitlichen Skalenbereich zu erhalten (Zheng und Casari 2019). Die Standardisierung, die auch eine Skalierung ist, projiziert die Werte in eine homogene Verteilung (Kamalov et al. 2023). Dabei wird der Mittelwert zur Null und die Standardabweichung eins (Larose und Larose 2014). Insgesamt sollen mit der Skalierung Verzerrungen in der Data-Mining-Analyse vermieden werden, die nur aufgrund von unterschiedlich großen Wertausprägungen verschiedener Merkmale auftreten (Borrohou et al. 2025).

Bei Merkmalen, die metrisch kontinuierlich ausgeprägt sind, kann es zu Rauschen in den Daten kommen. Die kleinen Veränderungen in den rauschenden Werten sind meist nicht relevant, können aber die Leistung und die Ergebnisse der Data-Mining-Algorithmen beeinflussen (Kantardzic 2020). Um diesem Problem vorzubeugen, können die Daten mit einer Datenglättung behandelt werden (Kantardzic 2020). Mit den Methoden der Datenglättung werden Rauschen und auch Ausreißer entfernt (Runkler 2015).

Die Datenaggregation ist eine weitere Aufgabe der Datentransformation. Dabei erfolgt eine Zusam-

menfassung von Daten auf einer höheren Ebene, wodurch auch eine Verkleinerung der Größe des Datensatzes erreicht werden kann (Çetin und Yıldız 2022). Deshalb kann es sinnvoll sein die Datenaggregation aus Sicht der Datenkompression oder aus inhaltlichen Gründen anzuwenden, weil ein neuer Blickwinkel auf die Daten erlangt wird (Cleve und Lämmel 2024)

Die Merkmalskonstruktion ist eine Datentransformationsaufgabe, die mit unterschiedlichen Methoden neue Merkmale konstruiert (Borrohou et al. 2025). Dies geschieht mit der Nutzung von bereits vorhandenen Merkmalen und (Rechen-) Operationen. Außerdem weisen Cleve und Lämmel (2024) darauf hin, dass mit einer Kombination oder Separierung von vorhandenen Merkmalen ebenso neue Merkmale konstruiert werden können. Ein Beispiel wäre das Zusammenfügen oder Trennen von Informationen in Datumseinheiten.

Die Diskretisierung ordnet metrisch kontinuierliche Daten in Intervalle ein und gibt diesen Bezeichnungen (Han et al. 2023). Dadurch werden die Werte in nominale und ordinale Daten umgewandelt (Cleve und Lämmel 2024). Eine Diskretisierung muss auf metrische Daten angewendet werden, falls der Data-Mining-Algorithmus nur mit diskreten oder nominalen Werten und einer endlichen Anzahl an Intervallen lernen kann (García et al. 2013). Zusätzlich kann diese auch bei Algorithmen, die mit kontinuierlichen Werten umgehen können, das Lernen von Mustern beschleunigen (García et al. 2013).

Im Gegensatz zur Diskretisierung werden in der Datencodierung nominale und ordinale Werte in metrische umgewandelt (Cleve und Lämmel 2024). Auch die Datencodierung wird angewendet, wenn bestimmte Daten im Datensatz an den Data-Mining-Algorithmus angepasst werden müssen (Li 2019). Bei der Anwendung einer Datencodierung auf ordinale Daten darf die Ordnungsrelation nicht verändert werden (Cleve und Lämmel 2024).

In diesem Abschnitt wurde ein Ausschnitt aller möglichen Aufgaben der Datentransformation vorgestellt, die in der Literaturrecherche zu Datentransformationen häufig vertreten waren. Nach dieser Vorstellung werden im nächsten Abschnitt einige Methoden dieser Aufgaben vorgestellt. Wie zu Beginn des Abschnitts erläutert wurde, können die Aufgaben mit verschiedenen Methoden bearbeitet werden.

2.2.4 Methoden der Datentransformation

In diesem Abschnitt werden verschiedene Methoden zu den Datentransformationsaufgaben vorgestellt, die für den Vergleich im Experimentierplan herangezogen werden können. Cleve und Lämmel (2024) listet kleinere Anpassungen im Datensatz, wie Anpassungen des Datentyps, von Zeichenketten oder der Datumsangaben und Maßeinheiten als Methoden der Datentransformation auf. Diese sind jedoch keiner übergeordneten Aufgabe aus Kapitel 2.2.3 zugeordnet.

Zuerst werden vier Methoden der Skalierung vorgestellt und der Normalisierung oder Standardisierung zugeordnet. Die erste Methode stellt die Min-Max-Normalisierung dar, die alle Werte innerhalb eines Merkmals von der originalen Spannweite $[o_{min}, o_{max}]$ auf eine neue, vorgegebene Spannweite $[n_{min}, n_{max}]$ projiziert (Li 2019). Dafür wird Gleichung 2.11 aus Li (2019) herangezogen. Der

normalisierte Wert wird mit v' und der originale Wert mit v dargestellt. Die neue Spannweite ist in dieser Arbeit $[0, 1]$, sodass nur der Bruch von der Normalisierung übrig bleibt.

$$v' = \frac{v - o_{min}}{o_{max} - o_{min}}(n_{max} - n_{min}) + n_{min} \tag{2.11}$$

Eine weitere Möglichkeit der Normalisierung besteht darin, ein Merkmal als Vektor zu betrachten und diesen mit einer Vektornormierung auf die Einheitslänge 1 zu bringen (García et al. 2016). Die Z-Score-Standardisierung ist in der Statistik weit verbreitet und funktioniert, indem die Differenz zwischen dem aktuellen Wert und dem Mittelwert des Merkmals ermittelt und mit der Standardabweichung des Merkmals skaliert wird (Larose und Larose 2014). Die Berechnung wird in Gleichung 2.12 aufgezeigt.

$$v' = \frac{v - \tilde{v}}{\sigma_x} \tag{2.12}$$

Anstatt den Mittelwert als Nullpunkt und die Standardabweichung als Streuung um den Nullpunkt zu verwenden, wie es bei der Z-Score-Standardisierung der Fall ist, kann diese auch medianbasiert sein. In diesem Fall wird der Median als neuer Nullpunkt gesetzt und die interquartile Spannweite wird für die Verteilung der Werte um den Nullpunkt verwendet. Diese Methode ist unter dem Namen RobustScaler in der Bibliothek Scikit-Learn von Pedregosa et al. (2011) zu finden.

Für die Datenglättung, die in Runkler (2015) auch als Filterung bezeichnet wird, kann der gleitende Mittelwert herangezogen werden. Dieser berechnet den Mittelwert aller Werte in einem Intervall, dessen Breite bestimmt werden kann, eines Merkmals (Runkler 2015). In dieser Arbeit wird der symmetrische gleitende Mittelwert mit Intervallbreite q in Gleichung 2.13 dargestellt, die aus Runkler (2015) entnommen wurde. Dabei steht das x für die Wertausprägungen der Merkmale, das k für die Position des Wertes im Merkmal und das Ergebnis y_k ist der gleitende Mittelwert an der k-ten Position.

$$y_k = \frac{1}{q} \sum_{i=k-\frac{q-1}{2}}^{k+\frac{q-1}{2}} x_i \tag{2.13}$$

Die Methode der exponentiellen Datenglättung gewichtet näher liegende Werte stärker, indem vorhergehende Werte mit dem Faktor $(1 - \eta)^i$ eingehen (Runkler 2015). Der Faktor η kann zwischen 0 und 1 gewählt werden (Runkler 2015). Außerdem fließen im Kontrast zum gleitenden Mittelwert nur vorangegangene Werte in die Glättung mit ein (Runkler 2015).

Weitere Methoden der Datentransformation können der Aggregation zugeordnet werden. Die Summenbildung stellt eine Methode der Datenaggregation dar (Cleve und Lämmel 2024). Han et al. (2023) stellen auch die Durchschnittsbildung als Aggregationsmethode dar. Ein Beispiel für eine Aggregation ist die Zusammenfassung von täglichen Produktionszahlen auf einer zeitlich höheren Ebene, wie einer Woche oder einem Monat. Dies schafft trotz einer Verkleinerung des Datensatzes eine neue Perspektive, die neue Informationen für das Data-Mining-Verfahren beinhalten kann.

Die Auswahl an Methoden der Merkmalskonstruktion ist sehr groß und vielfältig, wie Zheng und Casari (2019). Je nach Ansatz können Merkmale aller Art konstruiert werden. Dabei werden sowohl nominale als auch metrische Daten benutzt (Zheng und Casari 2019). Auf metrische Merkmale können zum Beispiel mathematische Funktionen wie ein Logarithmus angewendet werden (Zheng und

Casari 2019). Es ist auch möglich, mehrere Merkmale zu kombinieren, um ein neues Merkmal zu erschaffen, das durch die Kombination eine neue Aussage tätigt (Han et al. 2023). Außerdem weisen Zheng und Casari (2019) darauf hin, dass in einigen Fällen auch Zielvariablen mit der Merkmalskonstruktion angepasst werden können.

Für die Diskretisierung werden zwei Methoden vorgestellt, die dem Binning zugeordnet werden (Han et al. 2023). Dabei werden die Merkmale in eine bestimmte Anzahl von Intervallen eingeteilt, die festgelegt werden kann (Larose und Larose 2014). Die Methoden unterscheiden sich in der Intervallerstellung. Entweder werden die Intervalle mit einer gleichen Breite erstellt oder mit gleicher Häufigkeit (Han et al. 2023). Wenn die Intervalle mit gleicher Breite aufgeteilt wurden, sind die Ränder aller Intervalle gleich weit voneinander entfernt. Wenn die Intervalle gleiche Häufigkeiten haben, sind alle Intervalle mit gleich vielen Werten gefüllt. Die Intervall-Namen können nach der Füllung mit dem Mittelwert oder Median des jeweiligen Intervalls gebildet werden (Han et al. 2023).

Abschließend werden Methoden der Datencodierung vorgestellt. Mit dem One-Hot-Encoding werden nominale Werte in neue metrisch diskrete Merkmale umgewandelt, die dann die Werte 0 oder 1 annehmen (Li 2019). Im Allgemeinen ist One-Hot-Encoding für Merkmale mit wenigen möglichen Wertausprägungen angedacht, da der Datensatz sonst nach der Transformation sehr groß wird und nur spärlich mit Einsen ausgeprägt ist (Li 2019). Die Label-Codierung kann nur bei ordinalen Daten angewendet werden, da sonst eine nicht existierende Ordnung in die Daten eingebracht werden würde (Larose und Larose 2014). Bei dieser werden auch Werte zwischen 0 und 1 vergeben, um die Ordnung in den Merkmalen zu bewahren (Cleve und Lämmel 2024). Ein Beispiel für eine Label-Codierung ist die Umwandlung von Qualitätsstufen, wie *gut*, *mit Mängeln* und *Ausschuss* in die Werte $[1; 0,5; 0]$.

Es wurden jeweils verschiedene Methoden zu den Datentransformationsaufgaben und Methoden zur Behandlung von fehlenden Werten vorgestellt. Für einen Vergleich, um zu entscheiden, welche Methoden einen Datensatz für das Data-Mining-Verfahren besser vorbereiten, ist Wissen über deren Bewertung notwendig. Metriken, die dafür genutzt werden können, werden im nächsten Abschnitt vorgestellt.

2.2.5 Metriken zum Vergleich von Datenvorverarbeitungsmethoden

In diesem Abschnitt werden die Metriken vorgestellt, die benutzt werden, um die Datenvorverarbeitungsverfahren zur Behandlung von fehlenden Werten und zur Datentransformation untereinander zu vergleichen. Die Metriken sind nicht speziell auf einen Schritt in der Datenvorverarbeitung ausgelegt, sondern sollen alle untersuchten Datenvorverarbeitungsschritte mitberücksichtigen. Dafür wird das Muster oder das Modell, welches aus der Data-Mining-Phase als Ergebnis hervorgeht, mit verschiedenen Metriken bewertet. Die Metriken beschreiben die Modellgüte. Dieses Vorgehen, die Datenvorverarbeitung nicht direkt zu messen, sondern die Modellgüte der nachfolgenden Data-Mining-Phase, ist in der Literatur verbreitet und wird zum Beispiel von García et al. (2015) vorgeschlagen oder von Bilal et al. (2022) benutzt. Das Schema der Bewertung wird in der nachfolgenden Abbildung 2.5 dargestellt.



Abbildung 2.5: Bewertungsschema für den Vergleich der Datenvorverarbeitungsmethoden

Die Abbildung 2.5 veranschaulicht das Bewertungsschema, welches in der Literatur und in den genannten Quellen angewendet wird. Dazu werden Daten nach der Datenvorverarbeitung in eine Art Black-Box gegeben, die ein Ergebnis zum Vergleichen ausgibt. Die Black-Box wird mit einem schwarzen Rahmen dargestellt, in dem eine feste Abfolge an Ereignissen dargestellt ist. Auf den Daten wird ein Data-Mining-Algorithmus angewendet, mit dem ein Modell erzeugt wird (vgl. Kapitel 2.1.4). Dieses Modell wird mit einer vorher festgelegten Bewertungsmetrik bewertet und als Ergebnis zurückgegeben. Wenn dies öfter mit unterschiedlichen Datenvorverarbeitungen angewendet wird, werden dazu verschiedene Ergebnisse erzeugt, die untereinander verglichen werden können. Darüber ist ein Rückschluss auf die Güte der Datenvorverarbeitung unter den Voraussetzungen aus der Black-Box möglich (García et al. 2015). In Kapitel 2.1.2 werden einige Bewertungsmetriken für überwachte Lernverfahren vorgestellt.

Die Bewertung von überwachten Lernverfahren ist in Kapitel 2.1.4 näher vorgestellt worden. Dabei ist aufgezeigt worden, dass der Datensatz in einen Trainings- und einen Testdatensatz aufgeteilt wird. Dabei kann es zu einem Datenleck kommen, wie Kaufman et al. (2012) herausstellen. Die Autoren erklären ein Datenleck als die Einbringung von Informationen über eine Zielvariable in die Lernphase des Data-Mining-Algorithmus, die nicht rechtmäßig verfügbar sein sollten, um daraus zu lernen. Außerdem sagen sie, dass die Einbringung der Informationen unabsichtlich geschieht und auch in der Datenvorverarbeitung geschehen kann. Dies führt dann dazu, dass nicht optimale Lösungen für die Data-Mining-Aufgabe gefunden werden können und durch Datenlecks die Modellgüten überschätzt werden (Kaufman et al. 2012). Deshalb weisen Kaufman et al. (2012) darauf hin, dass keine Informationen aus dem Testdatensatz in der Lernphase mit einbezogen werden sollten. Datenvorverarbeitungsmethoden, die auf statistische Informationen des Datensatzes zurückgreifen, dürfen diese demnach nur vom Trainingsdatensatz erhalten.

Nachdem in diesem Abschnitt die Datenvorverarbeitung und vor allem die Behandlung fehlender Werte, die Datentransformation, die Methoden für beide Schritte und die Vergleichsmetrik für die Methoden vorgestellt wurden, fehlt noch die Betrachtung der Domäne. Dafür wird im nächsten Abschnitt die Wissensentdeckung in der Produktion vorgestellt. Dieser Abschnitt schließt die Lücke zwischen der Theorie zum KDD-Prozess und der spezifischen Anwendung des Prozesses auf Produktionsdatenbanken.

2.3 Wissensentdeckung in der Produktion

Nachdem KDD im Allgemeinen und detaillierter noch die Datenvorverarbeitung vorgestellt wurde, wird in diesem Kapitel die Verbindung zur Produktion hergestellt. Allgemein sorgt Data Mining in

der Produktion für eine Wissensentdeckung, die viele kompetitive Vorteile mit sich bringt (Dogan und Birant 2021). Diese sind die höhere Produktqualität, niedrigere Kosten und verbesserte Prozesse in der Produktion (Dogan und Birant 2021). Außerdem ist Data Mining für die Entwicklung von wissensbasierten Systemen in der Produktion wichtig (Dogan und Birant 2021).

Zunächst werden die Grundlagen der Produktion vorgestellt. Im weiteren Verlauf erfolgt eine Erläuterung der charakteristischen Erhebung und Speicherung von Daten in der Produktion sowie eine Darstellung der Eigenschaften von Produktionsdaten. Abschließend erfolgt die Präsentation der Anwendung des Data-Mining in der Produktion sowie der dazugehörigen Fragestellungen.

2.3.1 Produktion

Die Produktion kann allgemein als der „[...] gelenkte Einsatz von Gütern und Dienstleistungen, den sogenannten Produktionsfaktoren, zum Abbau von Rohstoffen oder zur Herstellung bzw. Fertigung von Gütern und zur Erzeugung von Dienstleistungen [...]“ (Bloech et al. 2014, S.3) definiert werden. Im allgemeinen Sprachgebrauch werden Fertigung und Produktion oft synonym verwendet. Die Fertigung beschreibt jedoch nur den Herstellungsprozess und wird von der Produktion umfasst (Grün und Jammernegg 2019). In dieser Arbeit wird der Begriff Produktion auf die Fertigung von materiellen Gütern in industriellen Unternehmen eingeschränkt. Industrielle Unternehmen weisen einen hohen Grad an Mechanisierung und Automatisierung in der Produktion auf (Lasi et al. 2014). Die Veranschaulichung eines Produktionssystems wird in Abbildung 2.6 in Anlehnung an Bloech et al. (2014) im Folgenden dargestellt.

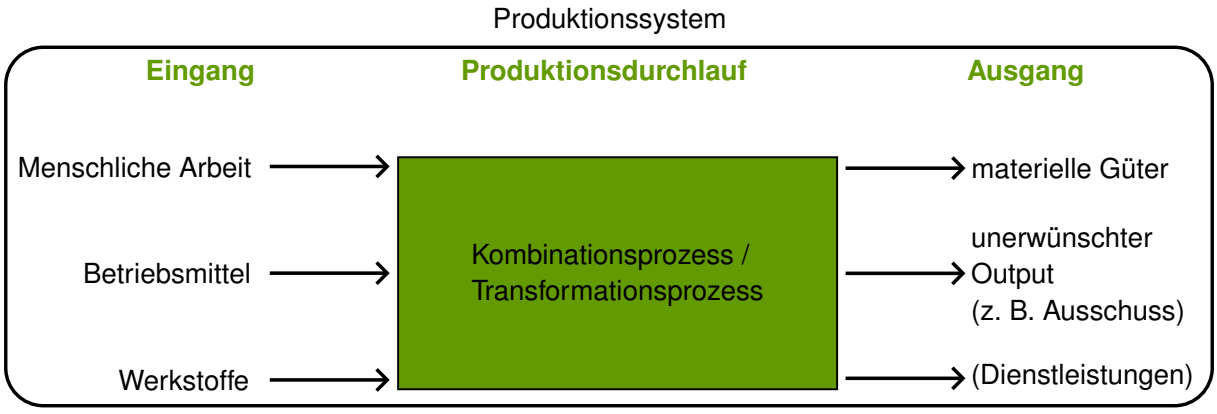

Abbildung 2.6: Produktionssystem angelehnt an Bloech et al. (2014, S. 3)

Wie in Abbildung 2.6 zu sehen, werden im Produktionsdurchlauf die Eingänge durch eine Kombination dieser in ein neues Gut oder eine neue Dienstleistung transformiert (Bloech et al. 2014). Währenddessen können auch unerwünschte Nebenprodukte entstehen, wie zum Beispiel Fehlteile oder Emissionen (Bloech et al. 2014). Die Eingänge in den Kombinations- oder Transformationsprozess werden Produktionsfaktoren genannt (Bloech et al. 2014). Bloech et al. (2014) teilt die Produktionsfaktoren in Betriebsmittel, Werkstoffe und menschliche Arbeit ein. Diese Einteilung wird beispielsweise auch von Gutenberg (1983) und Günther und Tempelmeier (2014) vorgenommen. Betriebsmittel lassen sich in materielle Betriebsmittel, wie Gebäude, Anlagen und Kapital, sowie in immaterielle Betriebsmittel, wie Informationen, Patente und Wissen, unterteilen. Zu den Werkstoffen

zählen Hilfs-, Betriebs- und Rohstoffe. Hilfsstoffe und Rohstoffe sind nach der Produktion Bestandteile des Produkts, wobei der Rohstoff die Hauptkomponente ausmacht. Betriebsstoffe werden im Prozess der Produktion verbraucht und sind danach nicht im Produkt wiederzufinden. Die menschliche Arbeit ist jede Aktivität in körperlicher oder geistiger Form, die zur Produktion von Sachgütern beiträgt (Gutenberg 1983).

In der Produktion hat die fortschreitende Digitalisierung Wandlungsprozesse angestoßen. Nach den Phasen der Mechanisierung, der intensiven Nutzung elektrischer Energie und der Digitalisierung isolierter Prozesse wird nun die Vernetzung der Prozesse als ein Schritt hin zur vierten industriellen Revolution betrachtet (Lasi et al. 2014). Das Bundesministerium für Bildung und Forschung hat dazu den Begriff Industrie 4.0 geprägt, der als Bezeichnung für eine Zukunftsstrategie erstellt wurde und sich im Anschluss etabliert hat (Lasi et al. 2014). Ziel von Industrie 4.0 ist die Herstellung individualisierter Produkte unter Massenproduktionsbedingungen (Lasi et al. 2014). Daraus folgt eine zunehmende Komplexität in der Produktion, die eine steigende Autonomisierung und Dezentralisierung nach sich zieht (Bauernhansl 2014). Mithilfe von CPS in einer smarten Fabrik ist es möglich, diese Herausforderungen zu bewältigen. Ein CPS kann über das Internet kommunizieren und Daten übertragen und auswerten (Bauernhansl 2014). Die Daten werden von Sensoren erfasst und von Aktoren in eine Auswirkung auf die physikalische Welt umgewandelt (Bauernhansl 2014). Je ausgereifter diese Systeme sind, desto intelligenter sind die Fabriken, in denen sie arbeiten. Die unterschiedlichen Reifegrade der CPS reichen von der Identifikation von Bauteilen, z. B. durch Radio-Frequency Identification (RFID) Chips, bis hin zur autonomen Entscheidungsfindung durch Informationsaustausch zwischen kooperierenden CPS (Bauernhansl 2014). Heutzutage sind fortgeschrittene CPS schon in der Produktion etabliert, wie Singh (2024) herausstellt. Dabei werden zum Beispiel viele verschiedene Daten eines Computerized Numerical Control (CNC) Fräsvorgangs gesammelt, um das Werkzeug in der Produktion kontinuierlich zu überwachen. Auf Basis von validierten Modellen werden diese Daten genutzt, um potenzielle Fehlerquellen frühzeitig zu erkennen und das Werkzeug vor einer Beschädigung auszutauschen (Singh 2024).

Die Generierung großer Datenmengen durch die zunehmende Vernetzung und Digitalisierung in der Produktion bildet die Grundlage für diese Entwicklungen. Im Nachfolgenden wird dieses Thema daher detaillierter untersucht.

2.3.2 Datenerfassung und -speicherung in der Produktion

Historisch betrachtet wurden Daten traditionell auf Papier dokumentiert und aufbewahrt. Da die Produktion zu dieser Zeit ausschließlich handwerklich erfolgte, erschien eine Kombination beider Elemente zu Beginn wenig sinnvoll (Tao et al. 2018). Mit der ersten industriellen Revolution wurden Daten zu Werkzeugmaschinen und den Arbeitern, die diese bedienten, zwar niedergeschrieben, jedoch noch manuell verarbeitet (Tao et al. 2018). Die Entwicklung zur Massenproduktion in der zweiten industriellen Revolution hat die Effektivität der Produktion erhöht, jedoch noch nicht die Effektivität der Datenverarbeitung (Hu 2013). Diese kam mit der dritten industriellen Revolution und dem Aufschwung von Informationstechnologien, die auch in der Produktion vermehrt benutzt wurden (Balakrishnan et al. 1999). Dies hat dazu geführt, dass Daten auf Computern in Informations-

systemen gespeichert wurden, die jedoch noch als Informationssilos agierten und die Daten noch nicht untereinander austauschten (Tao et al. 2018). Die Verknüpfung der einzelnen Systeme und Prozesse über die Daten erfolgte mit der vierten industriellen Revolution, die als moderne Produktion betrachtet wird (vgl. Kapitel 2.3.1).

In der modernen Produktion werden verschiedene Systeme benutzt (Siepmann und Graef 2016). Für die Herleitung werden das Enterprise-Resource-Planning (ERP)-System, das Manufacturing-Execution-System (MES)-System, das Supervisory Control and Data Acquisition (SCADA)-System und die speicherprogrammierbaren Steuerung (SPS) benötigt. Die moderne Produktion kann mithilfe der nachfolgend abgebildeten Automatisierungspyramide von der Gesellschaft Mess- und Automatisierungstechnik (2013) in Abbildung 2.7 strukturiert werden.

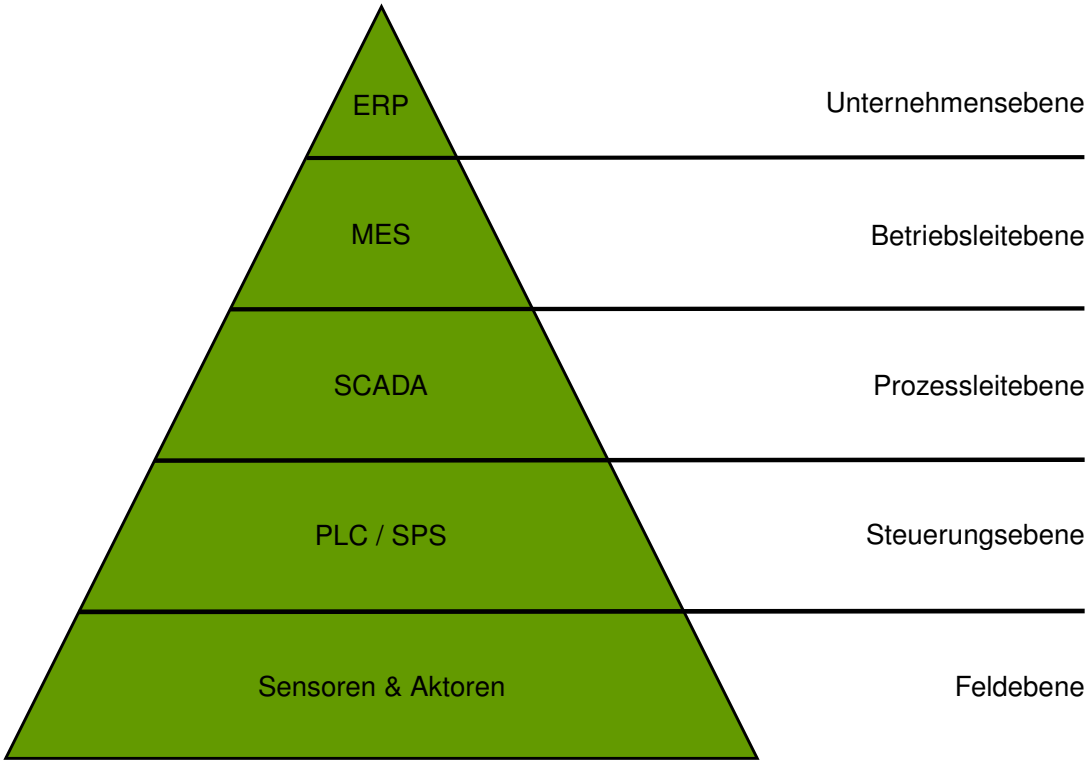

Abbildung 2.7: Automatisierungspyramide angelehnt an Gesellschaft Mess- und Automatisierungstechnik (2013, S. 4)

Abbildung 2.7 strukturiert die Ebenen und die dahinterliegenden Systeme im Unternehmen. Die Automatisierungspyramide dient dazu, Techniken und Systeme in das betriebliche Umfeld einzuordnen (Siepmann und Graef 2016). Die Automatisierungspyramide zeigt den meist noch vorherrschenden Stand in den Unternehmen, aber mit dem Fortschreiten der vierten industriellen Revolution wird und ist die hierarchische Struktur von einer flachen, untereinander vernetzten Struktur abgelöst (Gesellschaft Mess- und Automatisierungstechnik 2013). Für die nachfolgenden Erläuterungen zur Datenerfassung in der Produktion erfüllt diese jedoch weiterhin ihren Zweck.

Daten werden in der modernen Produktion auf verschiedene Weise und auf allen Ebenen im Unternehmen gesammelt (Tao et al. 2018). Die Datenerfassung in der Produktion wird auf der Feldebene eingeleitet (vgl. Abbildung 2.7) (Stock 2022). Anschließend werden die Daten an die höheren Ebenen weitergegeben und verarbeitet. Der Datenfluss erfolgt in der Automatisierungspyramide dem-

nach von unten nach oben. Die Daten können an jeder Stelle der Automatisierungspyramide abgegriffen werden. Die technischen Grundlagen des Auslesens und Weitersendens der Daten sind in dieser Arbeit nicht von Bedeutung. Im Folgenden wird beschrieben, welche Daten an welcher Stelle in der Produktion vorliegen:

- Auf der Feldebene werden Produktionsrelevante Daten in Form von Ein- und Ausgangssignalen der Sensoren und Aktoren ausgelesen und zurückgeführt (Siepmann und Graef 2016). Stock (2022) charakterisiert diese als Quelle für alle relevanten Daten, die aus dem Produktionsprozess erhoben werden. Daten, die von Sensoren kommen, sind physikalische Größen, die in elektrische Signale umgewandelt werden (z. B. : Temperatur [℃], Größe [mm] oder Gewicht [Kg]). Bei den Aktoren können Betriebszustände ausgelesen werden (z. B.: Drehgeschwindigkeit [min^{-1}] oder anliegende Spannung [V]).

- Gemäß dem vorgegebenen Prozess werden die Eingangssignale auf der Steuerungsebene von der Feldebene verarbeitet und in Ausgangssignale in Form von Ergebnisdaten umgewandelt (Siepmann und Graef 2016).

- Auf der Prozessleitebene werden die Daten in Prozess- und Zustandsdaten umgewandelt, die hauptsächlich dazu benutzt werden, um Informationen über den Prozess für Menschen anschaulich darzustellen (Stock 2022). Gleichzeitig dient die Prozessleitebene auch umgekehrt dazu menschliche Eingaben in Prozess- und Zustandsdaten umzuwandeln und an die unteren Ebenen weiterzugeben (Siepmann und Graef 2016).

- Auf der Betriebsleitebene werden im MES Betriebs-, Maschinen- und Personaldaten aus den anderen Ebenen zusammengefasst (Stock 2022).

- Auf der Unternehmensebene fließen, neben den Daten aus der Betriebsleitebene, auch externe Daten zur Unternehmerischen Planung mit in den Entscheidungsprozess ein (Stock 2022). Dazu gehören Beispielsweise Daten zu aktuellen Bestellungen, Ressourcen oder Personal. Zusätzlich sind Dokumente, die wichtige Informationen zu Produkten enthalten auf dieser Ebene abrufbar (Gröger 2022).

Vor dem Start der Datenerfassung sollte ein ausbaufähiges Konzept für Daten aus der Produktion erstellt werden, da in Zukunft vermutlich noch mehr Daten benötigt werden und auch größere Datenmengen strukturiert werden müssen (Dietrich 2021). Die Datenerfassung erfolgt größtenteils automatisch, kann aber auch an einigen Stellen noch manuell erfolgen (Dietrich 2021). Die automatische Datenerfassung erfolgt mittels Sensoren und Aktoren. Dies kann über die SPS geschehen, welche die Daten der zugeordneten Sensoren und Aktoren ohnehin ausliest und diese dann weitersendet (Dietrich 2021). Das direkte Auslesen der Sensordaten kann ebenfalls von Vorteil sein und wird durch den Einsatz entsprechender Adapter realisiert (Dietrich 2021). Des Weiteren können zusätzliche Sensoren, auch multifunktionale Sensoren, Messgrößen aus dem Fertigungsprozess aufnehmen. Der Vorteil von multifunktionalen Sensoren ist die gleichzeitige Betrachtung verschiedener Messgrößen, die mit mehreren Sensoren, die nur eine Messgröße aufnehmen, schwerer zu erlangen ist (Dietrich 2021). Von SCADA-Systemen können gleichzeitig Auftragsdaten mit Fortschritt und Ist-Zeiten gespeichert werden, aus denen Zeiten, wie Produktivzeiten, Rüstzeiten, Wartezeiten und

Transportzeiten errechnet werden können (Dietrich 2021). Die manuelle Datenerfassung erfolgt im Zusammenspiel von Arbeiterinnen und Arbeitern in der Produktion und einem technischen System für die Dateneingabe. Um zeiteffizient zu sein, ist die Dateneingabe meist individuell auf einen Arbeitsschritt oder einen Arbeitsplatz angepasst (Dietrich 2021). Zum Beispiel kann ein Einloggen am Arbeitsplatz über eine Mitarbeiterkarte dabei helfen, die gefertigten Produkte einem Mitarbeiter in der Produktion zuzuordnen. Auch ist es möglich, Messwerte von Prüfschritten manuell einzugeben oder die Messwerte mit einem Scanner den Produkten zuzuordnen. Daten aus unternehmensinternen Dokumenten können ebenso manuell übertragen werden. Außerdem können externe Quellen, die nicht unmittelbar aus der Produktion kommen, weitere Daten dem KDD-Prozess beisteuern und mit zusätzlichen Informationen helfen, Wissen zu generieren (Gröger 2022). Das sind zum Beispiel Daten aus dem Internet oder von den Produkten, die beim Kunden im Einsatz sind.

In modernen Produktionen erfolgt die Datenerfassung mit Hilfe leistungsfähiger Datenerfassungssysteme, die die Daten an den Prozessen erfassen und in großen Datenbanken zusammenfassen (Dogan und Birant 2021). Zusätzlich werden in der modernen Produktion Datenplattformen benutzt, die nicht nur die Speicherung der Daten umsetzen, sondern auch die Daten managen (Gröger 2022). Dabei werden zum Beispiel neben den Daten auch Metadaten zum Unternehmen und den Daten selbst gespeichert (Gröger 2022).

Nach der Erfassung und Speicherung von Daten in der Produktion werden im nächsten Unterkapitel die allgemeinen Eigenschaften von Produktionsdaten erörtert. Die Identifikation dieser Eigenschaften erfolgt durch eine Betrachtung des gesamten Datensatzes, der Daten aus verschiedenen Ebenen der Automatisierungspyramide enthält.

2.3.3 Eigenschaften von Produktionsdaten

In diesem Unterkapitel werden die typischen Eigenschaften von Produktionsdaten aufgelistet. Zuerst spezifische Eigenschaften von Merkmalen in Produktionsdatenbanken und im Anschluss allgemeine Eigenschaften, die typisch für Datensätze aus der Produktion sind. Zusätzlich werden besondere Auswirkungen oder Beschränkungen, die deswegen im KDD-Prozess, besonders bei der Datenvorverarbeitung zu beachten sind, hervorgehoben. Für einen erfolgreichen Data-Mining-Schritt im KDD-Prozess hält Weichert et al. (2019) es für essenziell, die Daten zuerst zu charakterisieren, um eine passende Data-Mining-Analyse auswählen zu können, da nicht alle Produktionsdaten die gleichen Eigenschaften besitzen. Meistens besitzen nicht alle Merkmale eines Datensatzes die gleichen Eigenschaften. Lieber et al. (2013) belegen dies, indem sie eine starke Heterogenität in Datensätzen aus der Produktion herausstellen. Weichert et al. (2019) ordnet die Produktionsdaten in Kategorien mit jeweils zwei unterschiedlichen Eigenschaften, die teilweise konträr zueinander sind, ein. Das genannte Paper ist ein Review-Paper, welches sich mit der Optimierung von Produktionsprozessen befasst. Aussagen, welche Eigenschaften öfter vorkommen und deshalb charakteristisch für Produktionsdaten sind, weisen folglich eine Tendenz zu den Prozessen in der Produktion auf und können bei anderen Blickwinkeln auf die Produktion ihre Gültigkeit verlieren.

Die Daten können **qualitativ oder quantitativ** ausgeprägt sein (Weichert et al. 2019). Eine qualitative Ausprägung eines Merkmals weist nominale oder ordinale Daten auf. Daten, die quantitativ

ausgeprägt sind, können jeden Wert in einem Intervall einnehmen (Weichert et al. 2019). Weitere Informationen dazu sind in Kapitel 2.1.1 aufgelistet. Prozessparameter sind beispielsweise hauptsächlich quantitative Daten (Weichert et al. 2019).

Zeitreihendaten oder werkstückbezogene Daten bilden die nächste Kategorie von Weichert et al. (2019). In Zeitreihen werden die Datentupel in regelmäßigem Abstand abgefragt und gespeichert, sodass in jedem Datentupel ein Zeitpunkt gespeichert ist, an dem dieses beobachtet wurde. Bei werkstückbezogenen Daten sind die Daten im Datentupel vom Werkstück abhängig. Diese Daten können jedoch zusätzlich auch Informationen zur Zeit beinhalten und als Zeitreihendaten charakterisiert werden (Weichert et al. 2019). Meist sind nur werkstückbezogene Daten verfügbar, falls die Daten nicht zur Anomalieerkennung verwendet werden (Weichert et al. 2019).

Eine Einteilung in **kontrollierbare und unkontrollierbare** Daten macht auch Sinn, da bei Produktionsprozessen oft Systemparameter kontrolliert werden können (Weichert et al. 2019). Kontrollierbare Daten können manuell oder automatisch direkt oder indirekt angepasst werden, während unkontrollierbare Daten höchstens indirekt angepasst werden können (Weichert et al. 2019). Das Muster herauszufinden, welches beschreibt, wie die unkontrollierbaren Daten indirekt angepasst werden können, ist oft ein Ziel des KDD-Prozesses (vgl. Kapitel 2.1.3).

Die Unterteilung in **beobachtbare Größen und Prozesszustandsvariablen** stellt eine weitere Kategorie dar. Oft ist es nicht möglich, Prozesszustandsvariablen direkt zu messen. Stattdessen werden sie über beobachtbare und messbare Größen errechnet (Weichert et al. 2019). Zum Beispiel kann die mechanische Spannung nur mit der Kraft und der Querschnittsfläche berechnet und nicht direkt gemessen werden.

Nachdem spezifische Eigenschaften von Merkmalen in Produktionsdaten aufgezählt wurden, werden im Anschluss noch Eigenschaften aufgelistet, die allgemein für Produktionsdaten typisch sind. Zu diesen stellen Lieber et al. (2013) heraus, dass neben der zuvor erwähnten Heterogenität auch unvollständige Daten mit Inkonsistenzen typisch für Produktionsdaten sind. Diese treten laut den Autoren oft aufgrund historisch gewachsener Systeme auf. Auch Cheng et al. (2018) weisen auf spärlich gefüllte Datensätze in der Produktion hin. Zusätzlich wird die hohe Dimensionalität der Daten erwähnt und darauf hingewiesen, dass der Fluch der Dimensionalität bei Data-Mining-Aufgaben berücksichtigt werden muss und mit Datenreduktionsverfahren, wie einer Hauptachsentransformation, behandelt werden kann (Dogan und Birant 2021). In der Literatur wird außerdem herausgestellt, zum Beispiel von Dogan und Birant (2021), dass Produktionsdaten oft als Big Data charakterisiert werden können. Die darunter fallenden Eigenschaften können mit den fünf Vs aufgelistet werden: Volume, Velocity, Variety, Veracity und Value (Olariu et al. 2020). Gemäß García et al. (2016) stellen insbesondere die großen Datenmengen, die laut den Autoren mindestens fünf Gigabyte groß sein sollten, um wirklich als groß zu gelten, eine Herausforderung für herkömmliche Datenvorverarbeitungsalgorithmen dar. Aus diesem Grund werden Lösungsansätze für die Datenvorverarbeitungsschritte gesucht. Im Folgenden werden aus diesem Artikel Lösungsansätze für die Behandlung von fehlenden Werten und zur Datentransformation vorgestellt.

Ein allgemeiner Ansatz ist die Nutzung eines MapReduce-Frameworks für die parallele Datenverarbeitung von großen Datenmengen (García et al. 2016). Das Hadoop MapReduce-Framework ist

eines davon und kann in der Industrie und Wissenschaft als Standard bei großen Datenmengen gesehen werden (Dittrich und Quiané-Ruiz 2012). Der Benutzer muss nur im Vorhinein die Map- und Reduce-Funktionen bestimmen (Dittrich und Quiané-Ruiz 2012). Die Map-Funktion verarbeitet zuerst die Eingabedaten und erzeugt somit neue Zwischenergebnisse, die im Anschluss von der Reduce-Funktion zum finalen Ergebnis aggregiert werden (Dittrich und Quiané-Ruiz 2012). Diese Funktionen können die Behandlung von fehlenden Werten und die Datentransformation ermöglichen (García et al. 2016).

Luengo et al. (2020) schlägt für die Imputation von fehlenden Werten bei großen Datensätzen einen angepassten KNN-Algorithmus vor, den k-Nearest-Neighbor Local Imputation (KNN-LI). Dieser bedient sich auch der MapReduce-Funktionsweise und führt den KNN-Algorithmus auf kleineren Partitionen des Datensatzes durch und führt die Partitionen danach ohne fehlende Werte wieder zu einem großen Datensatz zusammen (Luengo et al. 2020). Die Qualität der Imputation ist folglich abhängig von der Anzahl der Partitionen (Luengo et al. 2020).

Insgesamt kann festgehalten werden, dass in Produktionsdatenbanken viele verschiedene Eigenschaften von Daten auftreten können und eine hohe Komplexität zu erwarten ist. Damit wird die Wichtigkeit einer ordentlich durchgeführten Datenvorverarbeitung noch einmal betont. Typische Data-Mining-Anwendungsgebiete und -Fragestellungen in der Produktion, die nach der Datenvorverarbeitung angewendet werden, werden im nachfolgenden Unterkapitel aufgelistet.

2.3.4 Data-Mining-Anwendungsgebiete in der Produktion

In diesem Unterkapitel werden typische Anwendungsgebiete von Data Mining in der Produktion dargestellt. Dogan und Birant (2021) teilen die Anwendungsgebiete in die vier Oberkategorien Terminplanung, Überwachung, Qualität und Fehler ein. Im Anschluss werden diese weiter ausgeführt. Einen Zusammenhang zwischen Data-Mining-Algorithmen und Oberkategorien können Cheng et al. (2018) und Dogan und Birant (2021) nicht herstellen.

Zur **Terminplanung** gehört die Auftragsabwicklung, die Shop-Planung, die Sequenzierung, die Ressourceneinteilung sowie die Fertigungsplanung (Dogan und Birant 2021). Für die Terminplanung ist es möglich, komplett neue Terminierungsregeln in historischen Daten von Terminplänen zu entdecken und den Terminplan komplett neu aufzubauen oder die aktuellen Terminierungsregeln dynamisch anzupassen (Cheng et al. 2018).

Die **Überwachung** umfasst Systeme, die Entscheidungen unterstützen und Prozesse überwachen. Das Ziel besteht darin, die Key Performance Indicators (KPI) stabil zu halten und die Sichtbarkeit sowie das Verständnis von Fertigungssystemen zu erhöhen (Dogan und Birant 2021). Cheng et al. (2018) zeigen in ihrer Untersuchung verschiedener Ansätze zur Prozessüberwachung, dass Signale, die außerhalb definierter Prozessgrenzen liegen, mit Data Mining weiter analysiert werden können. So ist es möglich, Fehler zu klassifizieren und Ursachen für die Grenzüberschreitung zu ermitteln, sodass diese in Zukunft vermieden werden können (Cheng et al. 2018).

Zur **Qualität** zählt die datengetriebene Vorhersage der Qualität von Produkten, die Qualitätsverbes-

serung in großen und komplexen Produktionsprozessen, die Qualitätskontrolle und die Erkennung von Defekten in der Fertigung (Dogan und Birant 2021). Gröger (2022) führt als Anwendung dieser Kategorie die predictive process quality auf. Dabei werden Daten von Produktionsprozessen mit Data Mining ausgewertet, um die komplexen Abhängigkeiten zwischen Prozess- und Maschinenparametern sowie zwischen Werkzeugeigenschaften und Materialeigenschaften und deren Zusammenhang mit der Prozess- und Produktqualität besser zu verstehen (Gröger 2022). Dadurch soll eine Vorhersage der Produktqualität am Ende des Produktionsprozesses möglich werden und die Ursachen erkannt werden, die für eine schlechtere Produktqualität ausschlaggebend sind (Gröger 2022). Eine weitere Anwendung,die die Qualität des Produkts verbessert, ist das engineering in the loop, die ebenso Gröger (2022) aufführt. Dabei werden Daten im Produktlebenszyklus gesammelt, aus denen neues Wissen für die Weiterentwicklung des Produktes generiert werden kann (Gröger 2022).

Die **Fehler**-Kategorie umfasst die Erkennung von Abnormalitäten, die Wartung von Maschinen, die Fehlererkennung und die systematische Analyse von Produktionsequipment (Dogan und Birant 2021). Die Wartung von Maschinen benennt Gröger (2022) als predictive machine maintenance. Dabei werden laut dem Autor mit historischen Maschinen- und Wartungsdaten Vorhersagen zur Wartung von Maschinen getroffen. Dank dieser Vorhersagen können ungeplante Ausfallzeiten reduziert und unnötige Wartungsarbeiten an Verschleißteilen vermieden werden (Gröger 2022). Es ist anzumerken, dass Anwendungen nicht nur eine Oberkategorie bedienen können. Mit predictive machine maintenance wird ebenso die zeitliche Einplanung von Wartungen in den Terminkalender verbessert (Gröger 2022). Damit könnte es ebenso der Terminplanung untergeordnet werden.

Nachdem in diesem Kapitel der Stand der Technik mit den Grundlagen zur Wissensentdeckung, Datenvorverarbeitung und Data Mining in der Produktion dargestellt wird, wird im nächsten Kapitel der Experimentierplan auf Grundlage von Anforderungen, die auf dem Stand der Technik beruhen, erstellt.

3 Entwicklung des Experimentierplans zum Vergleich von Datenvorverarbeitungsmethoden

Im dritten Kapitel dieser Arbeit wird der Experimentierplan entwickelt und erstellt, der zum Vergleich der Datenvorverarbeitungsmethoden dient. Es werden Methoden zur Behandlung von fehlenden Werten und zur Datentransformation systematisch verprobt. Zuerst wird das methodische Vorgehen im Allgemeinen erläutert, bevor der Experimentierplan nach diesem Vorgehen aufgebaut wird. Für die Entwicklung des Experimentierplans sind Anforderungen nötig, die vorher dafür erhoben werden.

3.1 Ziel des Experimentierplans zum Vergleich von Datenvorverarbeitungsmethoden

Das Ziel eines Experimentierplans ist es, die systematische Durchführung eines Experiments anzuleiten. Mithilfe des Ergebnisses nach Ausführung dieses Experimentierplans ist eine Beurteilung möglich, welche Datenvorverarbeitungsmethoden zur Behandlung von fehlenden Werten und zur Datentransformation den größten Effekt auf die Wissensentdeckung im vorhandenen Datensatz haben. Darüber hinaus wird eine bestimmte Kombination aus Behandlung fehlender Werte und Datentransformation als die für den vorhandenen Datensatz mit definierter Data-Mining-Aufgabe und definiertem -Algorithmus beste herausgestellt. Dadurch werden individuelle Ergebnisse erzielt, die sich nicht eins zu eins aus anderen, generellen Experimenten, die Datenvorverarbeitungsschritte vergleichen, ableiten lassen.

Um dieses Ziel zu erreichen, wird der Experimentierplan in mehreren klar definierten Schritten entwickelt. Dadurch wird sichergestellt, dass die Entwicklung systematisch und nachvollziehbar erfolgt. Zuerst müssen dafür Anforderungen erhoben werden, um sicherzustellen, dass der Experimentierplan den spezifischen Gegebenheiten der Wissensentdeckung in der Produktion gerecht wird. Neben generellen Anforderungen an einen Experimentierplan fallen darunter Anforderungen, die von den Produktionsdaten ausgehen. Ebenso sind Anforderungen an die Methoden zur Behandlung von fehlenden Werten und zur Datentransformation wichtig. Zuletzt werden auch Anforderungen, die von den Vergleichsmetriken ausgehen, für die Erstellung des Experimentierplans herausgearbeitet. Auf Grundlage der Anforderungen wird der Experimentierplan erstellt. Da das Experiment im Experimentierplan auf den Grundlagen der SVP erfolgt, werden dabei Faktoren und Zielgrößen festgelegt. Für die Faktoren werden außerdem Faktorstufen festgelegt (vgl. Kapitel 2.1.2). Des Weiteren wird

definiert, wie und mit welcher Metrik die Zielgrößen gemessen und verglichen werden.

Um den Ablauf des Experimentierplans besser zu veranschaulichen und als Anleitung für Anwender bereitzustellen, wird ein Ablaufplan in Form eines Flussdiagramms erstellt. Es basiert auf den Grundlagen nach *DIN 66001* (1983). Ein Ausschnitt mit den Symbolen, die in diesem Kapitel verwendet werden, stellt Abbildung 3.1 dar.

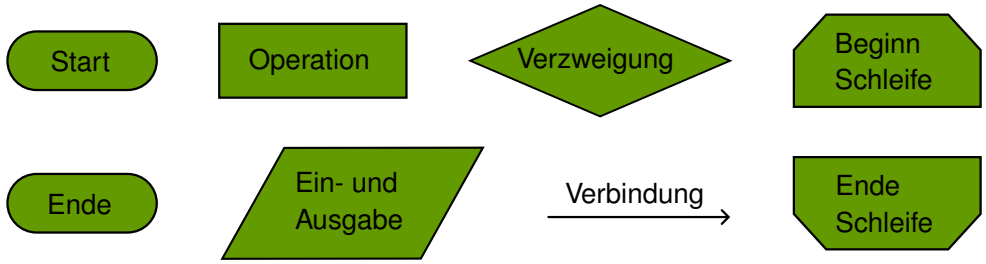

Abbildung 3.1: Legende für benutzte Symbole im Experimentierplan angeleht an die *DIN 66001* (1983)

Die Symbole des Ablaufplans, die in der Abbildung 3.1 dargestellt werden, verbinden die Form mit der Art der Aktion, auf die das jeweilige Symbol hinweist. Jedes Symbol wird im Ablaufplan mit einem kurzen Text ergänzt, der den jeweiligen Schritt beschreibt. Eine detailliertere Beschreibung erfolgt in der Herleitung des Schrittes im Text. Die Größe der Symbole ist im Ablaufplan variabel und passt sich der jeweiligen Beschreibung im Symbol an. Dies wurde für eine kompaktere Gesamtdarstellung und eine bessere Lesbarkeit der Beschreibungen so entschieden.

Für den SVP im Experimentierplan wird eine Tabelle bereitgestellt, in der die Faktoren in den verschiedenen Stufen und die Ergebnisse der Zielgrößen eingetragen werden können. Vorerst werden im nächsten Unterkapitel die Anforderungen herausgearbeitet, die grundlegend für den Experimentierplan sein werden.

3.2 Ausarbeitung der Anforderungen an den Experimentierplan

In diesem Unterkapitel werden Anforderungen an den Experimentierplan, zur Sicherstellung der Anwendbarkeit unter den spezifischen Gegebenheiten des KDD-Prozesses in der Produktion, gestellt. Die Anforderungen folgen den Grundlagen zu Anforderungen aus Kapitel 2.1.2. Erforderliche Anforderungen werden mit dem Signalwort *muss*, optionale Anforderungen mit dem Signalwort *soll* artikuliert. Zunächst erfolgt die Spezifikation jeder Anforderung, anschließend deren Diskussion und Herleitung. Die Struktur der Anforderungen an den Experimentierplan wird in der folgenden Abbildung 3.2 dargestellt.

Abbildung 3.2 zeigt eine vereinfachte Darstellung eines Hauses, bei dem das Fundament die grundlegenden Anforderungen an den Experimentierplan symbolisiert, die durch die Fragestellung bedingt sind. Auf dem Fundament stehen die drei zentralen Säulen, die auch jeweils Anforderungen an den Experimentierplan stellen. Dies sind Anforderungen von Produktionsdaten, von Methoden zur Behandlung fehlender Werte und zur Datentransformation sowie von Vergleichsmetriken. Das Dach des Hauses, das von den Anforderungen gehalten wird, symbolisiert den Experimentierplan,

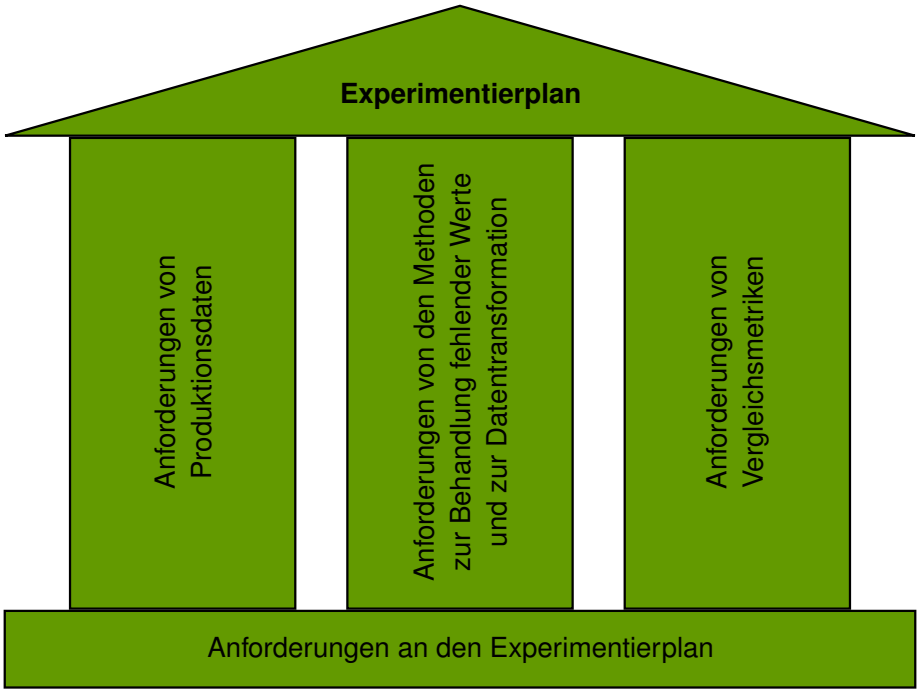

Abbildung 3.2: Die Säulen der Anforderungen an den Experimentierplan

da dieser mithilfe der Anforderungen erstellt wird und sich auf diese stützt. Zunächst werden in diesem Kapitel grundlegende Anforderungen an den Experimentierplan formuliert. Anschließend werden die Anforderungen der drei Säulen aufgenommen.

3.2.1 Anforderungen an den Experimentierplan

In diesem Unterkapitel werden sechs Anforderungen an den Experimentierplan dargelegt. Diese Forderungen werden aus einer neutralen Perspektive artikuliert und beinhalten die Verfolgung grundlegender Ziele der Arbeit sowie die Durchführung eines aussagekräftigen Experiments.

Anforderung 1: Der Experimentierplan muss die Anwendung von Methoden zur Behandlung fehlender Werte und zur Datentransformation vorsehen
Diese funktionale Anforderung stellt einen wichtigen Aspekt aus dem Titel der Arbeit dar und wird darüber hinaus mit dem Hauptziel begründet. Für einen Vergleich verschiedener Methoden muss sichergestellt werden, dass beim Durchlaufen des Experimentierplans Methoden aus beiden Schritten der Datenvorverarbeitung auf die Daten angewendet werden. Dies impliziert auch, dass es notwendig ist, fehlende Werte zu behandeln und Daten zu transformieren. Falls diese beiden Datenvorverarbeitungsschritte für den Datensatz nicht notwendig sind, sollte ein anderer Datensatz gewählt werden. Im Fall eines kompletten Datensatzes kann es sinnvoll sein, fehlende Werte nach einem passenden Ausfallmechanismus (vgl. Kapitel 2.2.1) künstlich zu erzeugen. Der Experimentierplan wird diese Entscheidungen mit gestellten Fragen anleiten.

Anforderung 2: Die Versuchsläufe im Experimentierplan müssen qualitativ vergleichbar sein
Auch diese Anforderung geht aus dem Titel der Arbeit und dem dazugehörigen Hauptziel hervor. Die objektive Vergleichbarkeit der Experimentergebnisse muss durch den Experimentierplan vor-

gegeben sein. Um die Anforderung umzusetzen, wird auf die Modellgüte des angewendeten Data-Mining-Algorithmus zurückgegriffen. Dieses Vorgehen und die dafür relevanten Metriken werden in Kapitel 2.2.5 dargestellt. Ein besseres Modell mit einem besseren Bewertungsergebnis impliziert somit eine bessere Datengrundlage, die in der Datenvorverarbeitung geschaffen wird.

Anforderung 3: Im Experimentierplan müssen die Methoden systematisch untersucht werden
Diese Anforderung wird mit der Reproduzierbarkeit begründet, die für wissenschaftliche Arbeiten vorausgesetzt wird. Darüber hinaus gewährleistet ein systematischer Ansatz eine effiziente Durchführung der Versuchsläufe sowie eine objektive Auswahl und Auswertung der Faktoren. Mithilfe der SVP, die in Kapitel 2.1.2 dargestellt wird, wird die Systematik im Experiment gewährleistet. Die Erstellung einer Tabelle für die SVP, die im Experiment die Stufen der Faktoren in den Versuchsläufen dokumentiert, wird deshalb im Experimentierplan vorgesehen.

Anforderung 4: Der Experimentierplan sollte von einer Person mit Erfahrung im KDD-Prozess angewendet werden
Diese Qualitätsanforderung ergibt sich aus der Tatsache, dass der Experimentierplan keine Ansprüche an die Vermittlung des KDD-Prozesses an den Nutzer stellt. Da im Experimentierplan aufgrund von Anforderung 2 jedoch die Anwendung eines Data-Mining-Algorithmus für die Bewertung vorgesehen wird, ist Vorerfahrung im KDD-Prozess impliziert gefordert. Darüber hinaus erfordert der Experimentierplan Einschätzungen und Einordnungen, die auf Erfahrungen im KDD-Prozess basieren.

Anforderung 5: Der Experimentierplan sollte an das Vorgehen eines KDD-Vorgehensmodells angelehnt sein
Da der Experimentierplan für KDD in der Produktion vorgesehen ist, sollte er einem etablierten Vorgehensmodell folgen, welches den KDD-Prozess beschreibt. In dieser Arbeit wird auf das KDD-Vorgehensmodell nach Fayyad et al. (1996a) zurückgegriffen, welches in Kapitel 2.1.3 dargestellt und erläutert wird. Die ersten sieben Schritte bis zu den Mustern, die in den Daten gefunden werden sollen, werden im Experimentierplan durchlaufen.

Anforderung 6: Innerhalb des Experimentierplans muss eine Rückführung an einen vorherigen Schritt ermöglicht werden
Diese Anforderung vertieft Anforderung 5 und fokussiert sich auf das iterative Element im KDD-Prozess, welches in verschiedenen Vorgehensmodellen wiederzufinden ist (vgl. Kapitel 2.1.3). Daraus lässt sich die Schlussfolgerung ableiten, dass auch im Experimentierplan eine Rückführung zu vorherigen Schritten möglich sein muss, um Anpassungen vornehmen zu können.

Nach den sechs Anforderungen, die das Fundament für den Experimentierplan darstellen, werden weitere Anforderungen erhoben, um diesen zu spezifizieren. Diese stehen als Säulen auf dem Fundament. Die erste Säule besteht aus den Anforderungen von Produktionsdaten, die im nächsten Abschnitt aufgestellt werden.

3.2.2 Anforderungen von Produktionsdaten

Nach den fundamentalen Anforderungen an den Experimentierplan werden in diesem Abschnitt drei Anforderungen von den Produktionsdaten erhoben. Diese Anforderungen sind für äußere Rahmenbedingungen wichtig und werden deshalb den Anforderungen aus den anderen Säulen vorgezogen.

Anforderung 7: Der Experimentierplan muss eine geeignete grundlegende Datenbasis erschaffen
Diese Anforderung wird an den Experimentierplan gestellt, da eine einheitliche Grundlage für alle Versuchsläufe geschaffen werden muss. Das heißt, dass der zu untersuchende Datensatz einer Datenselektion unterzogen wird und nötige Datenvorverarbeitungsschritte, die nicht verglichen werden sollen, auf die selektierten Daten angewendet werden. Diese Schritte werden in Kapitel 2.2 vorgestellt. Datensätze aus der Produktion weisen eine hohe Heterogenität auf. Zusätzlich können einige Fehler und Ungenauigkeiten vorkommen (vgl. Kapitel 2.3.3). Ein präparierter Datensatz, der nur noch die Behandlung von fehlenden Werten und Datentransformationen für ein nachfolgendes Data-Mining-Verfahren benötigt, muss daher ein Zwischenziel im Experimentierplan sein.

Anforderung 8: Der Experimentierplan muss die Eigenschaften des Datensatzes abfragen, damit die Wahl erforderlicher Datenvorverarbeitungsmethoden möglich ist
Diese Anforderung ist nötig, weil die Auswahl der passenden Datenvorverarbeitungsschritte von den Gegebenheiten des Datensatzes abhängig ist (vgl. Kapitel 2.2). Der Experimentierplan muss den Anwender also zur Anwendung der benötigten Datenvorverarbeitungsschritte anleiten. Wie in Kapitel 2.2 beschrieben, kann die Anwendung dieser Datenvorverarbeitungsschritte eine Voraussetzung für die Durchführung des Data-Mining-Algorithmus sein. Die Auswahl der geeigneten Datentransformationsaufgaben, die in Kapitel 2.2.3 dargelegt werden, ist von entscheidender Bedeutung für den erfolgreichen Vergleich der verschiedenen Methoden einer Datentransformationsaufgabe. Dies ist darauf zurückzuführen, dass je nach Art der Data-Mining-Aufgabe unterschiedliche Eigenschaften des Datensatzes erforderlich sind, die in der Datenvorverarbeitung geschaffen werden können (vgl. Kapitel 2.1.4). Dafür wird auch die Datentransformation und die Behandlung von fehlenden Werten benötigt.

Anforderung 9: Der Experimentierplan sollte auf Big Data angewendet werden können
Diese Qualitätsanforderung wird mit den Eigenschaften von Produktionsdaten begründet. Wie in Kapitel 2.3.3 herausgestellt wird, ist Big Data ein häufig vorkommendes Merkmal bei Produktionsdaten. Deshalb sollte die potenzielle Anwendung des Experimentierplans auf Big Data vorgesehen sein und ein Weg mit dem Umgang von Big Data aufgezeigt werden. Dass große Datenmengen, eine Eigenschaft von Big Data, in der Datenvorverarbeitung mit angepassten Algorithmen oder der Anwendung von MapReduce-Verfahren besser behandelt werden können, wird in Kapitel 2.3.3 aufgezeigt.

Im KDD-Prozess wird nach der Datenselektion die Datenvorverarbeitung und die Datentransformation durchgeführt (vgl. Kapitel 2.1.3). Aus diesem Grund werden im folgenden Abschnitt, nach den Anforderungen der Produktionsdaten, die Anforderungen der zu vergleichenden Datenvorverarbeitungsmethoden an den Experimentierplan erörtert.

3.2.3 Anforderungen von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation

In diesem Unterkapitel werden Anforderungen an den Experimentierplan gestellt, die von den Methoden zur Behandlung von fehlenden Werten und zur Datentransformation ausgehen. Da der Experimentierplan mehrere Methoden vergleichen soll, werden die Methoden als Gesamtheit betrachtet.

Anforderung 10: Die Methoden müssen den Datensatz für die Data-Mining-Aufgabe vorbereiten
Diese funktionale Anforderung folgt Anforderung 2 und der Bedingung, dass eine Data-Mining-Aufgabe mit einem festgelegten Algorithmus bestimmte Voraussetzungen an die Eigenschaften des Datensatzes stellt (vgl. Kapitel 2.1.4). Der Experimentierplan muss dazu anleiten, dass vor Anwendung des Data-Mining-Algorithmus diese Eigenschaften gegeben sein werden. Die Umsetzung dieser Anforderung steht im Zusammenhang mit Anforderung 8, die fordert, dass die Datenvorverarbeitung im Experimentierplan an den Datensatz angepasst werden kann, indem Eigenschaften abgefragt werden.

Anforderung 11: Der Experimentierplan kann nur eine Methode zur Behandlung von fehlenden Werten pro Versuchslauf anwenden
Dies ist eine logische Randbedingung, die beachtet werden muss, da nach der korrekten Behandlung von fehlenden Werten der Datensatz komplett ist. Eine weitere Behandlung der fehlenden Werte ist demnach in diesem Versuchslauf nicht mehr möglich. Dies muss im Experimentierplan berücksichtigt werden, wenn die Tabelle für den SVP erstellt wird, die aus Anforderung 3 geschlussfolgert wird.

Anforderung 12: Der Experimentierplan muss die Aufgaben der Datentransformationen festlegen
Diese Anforderung fordert, dass der Experimentierplan dazu anleitet, die erforderlichen Aufgaben der Datentransformation für den angedachten Data-Mining-Algorithmus zu bestimmen. Für den Prozess wird Anforderung 8 herangezogen. Einerseits wird damit Anforderung 10 erfüllt, andererseits wird damit bestimmt, welche Datentransformationsaufgaben auf jeden Fall im Experimentierplan untersucht werden müssen. Analog zu Anforderung 11 wird mit dieser Anforderung außerdem festgelegt, dass in jedem Versuchslauf je Datentransformationsaufgabe nur eine Methode pro Aufgabe angewendet wird.

Nach den Schritten der Datenvorverarbeitung und Datentransformation wird im KDD-Prozess das Data Mining angewendet, um Muster zu finden (vgl. Kapitel 2.1.3). Vergleichsmetriken werden zur Bewertung der Muster genutzt und gewährleisten somit die Vergleichbarkeit der Methoden im Experimentierplan. Aus diesem Grund werden nachfolgend auch Anforderungen der Vergleichsmetriken an den Experimentierplan erhoben.

3.2.4 Anforderungen von Vergleichsmetriken

Als letzte Säule werden Anforderungen an den Experimentierplan aus der Sicht der Vergleichsmetriken gestellt. Das sind zwei Anforderungen, die erfüllt sein müssen, damit der Vergleich der Methoden und Methodenkombinationen funktioniert. Diese Anforderungen folgen der Anforderung 2, die einen Vergleich mit der Bewertung der Modellgüte des Data-Mining-Algorithmus im Experimentierplan festlegt.

Anforderung 13: Die Datenvorverarbeitungsmethoden, die verglichen werden, müssen in eine Anwendung vor oder nach dem Train-Test-Split eingeteilt werden
Diese Anforderung wird von den Vergleichsmetriken an den Experimentierplan gestellt, da ein Train-Test-Split für die Bewertung nach dem Vorgehen aus Kapitel 2.2.5 benötigt wird. Wie dort erklärt wird, kann es zu Datenlecks kommen, wenn Eigenschaften aus dem Testdatensatz in die Lernphase des Data-Mining-Algorithmus mit einfließen. Aufgrund dessen können die Datenvorverarbeitungsmethoden, die verglichen werden sollen, je nach Funktion der Methode, teilweise nur vor oder nur nach der Aufteilung angewendet werden. Folglich müssen diese Methoden vor der Anwendung im SVP in zwei Gruppen eingeteilt werden, die auch in der Durchführung des SVPs berücksichtigt werden.

Anforderung 14: Der Datensatz muss mit einem Data-Mining-Algorithmus brauchbare Ergebnisse liefern
Da nicht in jedem Datensatz Wissen entdeckt werden kann und es auch keine Garantie gibt, dass Data-Mining-Algorithmen brauchbare Muster finden (vgl. Kapitel 2.1.4), muss durch den Experimentierplan sichergestellt werden, dass diese Anforderung im vorliegenden Datensatz erfüllt wird. Die Durchführung des Experimentierplans ist nur dann sinnvoll, wenn Wissen in dem Datensatz entdeckt werden kann, da die Ergebnisse des Experimentierplans individuell für den vorliegenden Datensatz aus der Produktion sind. Falls die Anforderung nicht direkt erfüllt werden kann, sollte der Experimentierplan den Anwender anleiten, mithilfe von anderen Data-Mining-Algorithmen oder einer anderen Data-Mining-Aufgabe eine funktionsfähige Mustererkennung zu ermöglichen. Sollte dies nicht gelingen oder nicht mehr dem Ziel entsprechen, welches der Anwender verfolgt, muss der Experimentierplan den Nutzer darauf hinweisen, dass die Nutzung des Experimentierplans mit diesem Datensatz nicht möglich oder sinnvoll ist.

Anforderung 15: Die initialen Ergebnisse ohne Anpassung des Datensatzes müssen Spielraum für Verbesserung lassen
Diese qualitative Anforderung ist für den Vergleich wichtig. Ein vorheriger Versuchslauf sollte, wenn möglich, ohne die Behandlung fehlender Werte oder Datentransformationen durchgeführt werden, um einen Referenzwert der Modellgüte zu erhalten (vgl. Kapitel 2.2.5). Falls andere Datenvorverarbeitungsmethoden für die Anwendung des Data-Mining-Algorithmus erforderlich sind, müssen diese angewendet werden. Der erlangte Referenzwert sollte nicht zu hoch sein, damit die Auswirkungen einer Datenvorverarbeitung messbar sind. Wenn die initialen Ergebnisse zu gut sind, kann es vorkommen, dass eine Datenvorverarbeitung eine zu geringe Auswirkung auf die Ergebnisse hat, weil die Bereiche oder Merkmale in den Daten vorverarbeitet werden, nur einen geringen oder gar keinen Einfluss auf das Modell haben, welches mit dem Data-Mining-Algorithmus erstellt wird. Dann

muss entweder die Datenvorverarbeitung angepasst werden, oder die Datenselektion, damit irrelevante oder redundante Merkmale nicht weiter vorverarbeitet werden können (vgl. Kapitel 2.1.3). Eine Rückführung, wie sie auch Anforderung 6 fordert, wird dafür benötigt. Die Entscheidung, wann ein Modell zu gut ist und eine Verbesserung der Ergebnisse mit einer Datenvorverarbeitung nicht mehr zu erkennen ist, hängt von mehreren Faktoren, wie zum Beispiel der Data-Mining-Aufgabe, ab und ist deshalb vom Anwender zu tätigen.

Um die Erhebung der Anforderungen abzuschließen, werden alle Anforderungen zusammengetragen, um eine zentrale Übersicht zu schaffen. Dafür werden alle erhobenen Anforderungen in folgender Tabelle 3.1 zusammen dargestellt.

Tabelle 3.1: Anforderungen an den Experimentierplan

Kategorie	Anforderung	Beschreibung
Experimentierplan	Anforderung 1	Der Experimentierplan muss die Anwendung von Methoden zur Behandlung fehlender Werte und zur Datentransformation vorsehen
Experimentierplan	Anforderung 2	Die Versuchsläufe im Experimentierplan müssen qualitativ vergleichbar sein
Experimentierplan	Anforderung 3	Im Experimentierplan müssen die Methoden systematisch untersucht werden
Experimentierplan	Anforderung 4	Der Experimentierplan sollte von einer Person mit Erfahrung im KDD-Prozess angewendet werden
Experimentierplan	Anforderung 5	Der Experimentierplan sollte an das Vorgehen eines KDD-Vorgehensmodells angelehnt sein
Experimentierplan	Anforderung 6	Innerhalb des Experimentierplans muss eine Rückführung an einen vorherigen Schritt ermöglicht werden
Produktionsdaten	Anforderung 7	Der Experimentierplan muss eine geeignete grundlegende Datenbasis erschaffen
Produktionsdaten	Anforderung 8	Der Experimentierplan muss die Eigenschaften des Datensatzes abfragen, damit die Wahl erforderlicher Datenvorverarbeitungsmethoden möglich ist
Produktionsdaten	Anforderung 9	Der Experimentierplan sollte auf Big Data angewendet werden können
Methoden	Anforderung 10	Die Methoden müssen den Datensatz für die Data-Mining-Aufgabe vorbereiten
Methoden	Anforderung 11	Der Experimentierplan kann nur eine Methode zur Behandlung von fehlenden Werten pro Versuchslauf anwenden
Methoden	Anforderung 12	Der Experimentierplan muss die Aufgaben der Datentransformationen festlegen
Vergleichsmetriken	Anforderung 13	Die Datenvorverarbeitungsmethoden, die verglichen werden, müssen in eine Anwendung vor oder nach dem Train-Test-Split eingeteilt werden
Vergleichsmetriken	Anforderung 14	Der Datensatz muss mit einem Data-Mining-Algorithmus brauchbare Ergebnisse liefern
Vergleichsmetriken	Anforderung 15	Die initialen Ergebnisse ohne Anpassung des Datensatzes müssen Spielraum für Verbesserung lassen

Die Tabelle 3.1 zeigt alle hergeleiteten Anforderungen und ordnet sie in den Kategorien ein, aus de-

nen sie erhoben worden sind. Die Nummer der Anforderung und die Beschreibung werden ebenso mit aufgeführt. Tabelle 3.1 enthält sechs grundlegende Anforderungen, die an den Experimentierplan gestellt werden, um die Ziele dieser Arbeit zu erfüllen. Dazu kommen jeweils drei Anforderungen, die von den Produktionsdaten, Methoden und Vergleichsmetriken an den Experimentierplan gestellt werden. Insgesamt werden 15 Anforderungen erhoben, auf die der Experimentierplan sich stützt.

Die Erhebung der Anforderungen an den Experimentierplan stellt den initialen Schritt der Erstellung des Experimentierplans dar. In der zweiten Stufe des Prozesses wird auf Grundlage der zuvor identifizierten Anforderungen ein Experimentierplan erstellt. Dieser wird im nachfolgenden Unterkapitel erörtert.

3.3 Erstellung eines Experimentierplans zum Vergleich von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation

In diesem Unterkapitel wird der Experimentierplan erstellt, für den in Kapitel 3.2 die Anforderungen erhoben wurden. Der Experimentierplan wird als Ablaufplan dargestellt und wird auf Tabellen zurückgreifen, die vom Experimentierplan bereitgestellt werden. In Anforderung 4 wird gefordert, dass der Anwender des Experimentierplans Vorwissen im KDD-Prozess vorweisen sollte. Dadurch ist es dem Experimentierplan möglich, an einigen Stellen Einschätzungen zu verlangen, die ohne Erfahrung nur schwer zu treffen sind. Auch die im Experimentierplan getroffenen Auswahlen werden durch diese Anforderung fundierter. Dadurch kann die Komplexität des Experimentierplans geringer gehalten werden. Mithilfe der Anforderungen, die alle in Tabelle 3.1 aufgeführt werden, und dem theoretischen Wissen aus Kapitel 2 wird der Experimentierplan in diesem Unterkapitel Stück für Stück hergeleitet und aufgebaut. Abschließend wird der gesamte Plan in kompakter Form dargestellt, wobei die zuvor hergeleiteten Schritte in dieser Version als einzelne Bausteine zu sehen sind. Eine komplette kleinschrittige Version mit den zusammenhängenden hergeleiteten Stücken ist in Anhang A hinterlegt.

Zum Start des Experimentierplans wird festgelegt, nach welcher Struktur dieser aufgebaut wird und mit welcher Aktion dieser anfängt. Dafür wird Anforderung 5 herangezogen. In dieser ist festgelegt worden, dass der Experimentierplan an die Struktur des KDD-Vorgehensmodells von Fayyad et al. (1996a) angelehnt wird und einen großen Teil davon durchläuft. In Kapitel 2.1.3 wird dieses Vorgehensmodell dargestellt. So wie im Vorgehensmodell wird der Datensatz zu Beginn des Experimentierplans geladen und es erfolgt eine Datenselektion, die aus Anforderung 7 hervorgeht und in Kapitel 2.2 als Schritt der Datenvorverarbeitung erwähnt wird. Diese wird auch vorgenommen, um den Datensatz potenziell zu verkleinern und den nachfolgenden Aufwand der Datenvorverarbeitung im Experimentierplan zu verringern. Um sicherzustellen, dass in diesem Schritt keine Informationen verloren gehen, sollte der Anwender besser zu viele Merkmale auswählen als zu wenige.

Zu Beginn muss für die objektive Vergleichbarkeit der Methoden, die in Anforderung 2 gefordert

wird, die Data-Mining-Phase spezifiziert werden. Dafür wird eine Data-Mining-Aufgabe festgelegt, an der der KDD-Prozess ausgelegt werden kann (vgl. Kapitel 2.1.4). Beispiele für Data-Mining-Aufgaben in der Produktion sind in Kapitel 2.3.4 aufgelistet. Nach der Definition der Aufgabe muss ein spezifischer Data-Mining-Algorithmus gewählt werden, der die Aufgabe lösen soll. Zusätzlich muss nach der Festlegung eines Data-Mining-Algorithmus für den späteren Vergleich der Algorithmen eine Vergleichsmetrik oder verschiedene Metriken festgelegt werden. Dies ist der erste Schritt für Anforderung 2, die eine objektive Vergleichbarkeit der Datenvorverarbeitungsgüte fordert. Dabei hilft die Tabelle 3.2, die Vergleichsmetriken für verschiedene Data-Mining-Aufgaben vorschlägt.

Tabelle 3.2: Ausgewählte Vergleichsmetriken der Modellgüte für den Experimentierplan

Data-Mining-Aufgabe	Vergleichsmetriken
Klassifikation	F1-Score, Präzision, Sensitivität und Genauigkeit
Regression	RMSE, MAE, MAPE und Bestimmtheitsmaß R^2
Clusteranalyse	Adjusted Rand Index und Silhouettenkoeffizient
Effizienz	Laufzeitmessung
Andere	Weitere

Die Vergleichsmetriken in Tabelle 3.2 werden in Kapitel 2.1.2 vorgestellt und in Kapitel 2.2.5 im Kontext zur Bewertung des Data Mining betrachtet. Es ist dem Anwender überlassen, welche und wie viele Vergleichsmetriken er für den Vergleich benutzen möchte. Die Benutzung einer anderen Vergleichsmetrik, die nicht in Tabelle 3.2 aufgelistet ist, ist ebenso möglich. Weil die Nutzbarkeit der einzelnen Vergleichsmetriken auch von der Data-Mining-Aufgabe abhängt, wie in Kapitel 2.1.2 herausgestellt wird, sollte diese bei der Auswahl ebenso berücksichtigt werden. Unter der Prämisse, dass die Effizienz der Analyse relevant ist, obliegt es dem Anwender, die Laufzeit des Data-Mining-Algorithmus zu messen und zu vergleichen. Bei mehreren Vergleichsmetriken muss der Anwender festlegen, wie wichtig jede einzelne Metrik ist und wie sie in einen Gesamtvergleich einfließt. Insgesamt ergibt sich für den Start des Experimentierplans folgender Ablaufplan, der in Abbildung 3.3 dargestellt wird.

Die Abbildung 3.3 zeigt den Start des Experimentierplans, der aus dem Startfeld, der Dateneingabe und 4 Operationen besteht, die die Data-Mining-Phase spezifizieren und somit festlegen, wie die Methoden in dem Experimentierplan verglichen werden sollen. Außerdem ist auf der linken Seite von den Operationen eine Rückführung mit einem gestrichelten Pfeil angedeutet, die aus Anforderung 6 motiviert ist und aufgrund von Anforderung 5 an das Vorgehensmodell aus Kapitel 2.1.3 angelehnt ist. Mit dieser ist es möglich, von einem späteren Zeitpunkt im Experimentierplan zu einem Schritt am Anfang zurückzukehren und Anpassungen vorzunehmen. Auch innerhalb des Starts ist ein Schritt zurück bei der Spezifizierung der Data-Mining-Phase und der Datenselektion möglich.

Für Anforderung 7, Anforderung 8, Anforderung 10 und Anforderung 12 ist es nötig, die Voraussetzungen für den Data-Mining-Algorithmus mit der Data-Mining-Aufgabe und die Eigenschaften des Datensatzes zu kennen. Dafür werden diese im Experimentierplan abgefragt und dokumentiert. Zuerst wird der Anwender des Experimentierplans aufgefordert, die Voraussetzungen zu dokumentieren. Wie in Kapitel 2.1.4 herausgestellt wurde, haben Data-Mining-Algorithmen unterschiedliche Voraussetzungen, die an die Eigenschaften der Daten gestellt werden. Als Anhaltspunkt wird dafür Tabelle 3.3 bereitgestellt.

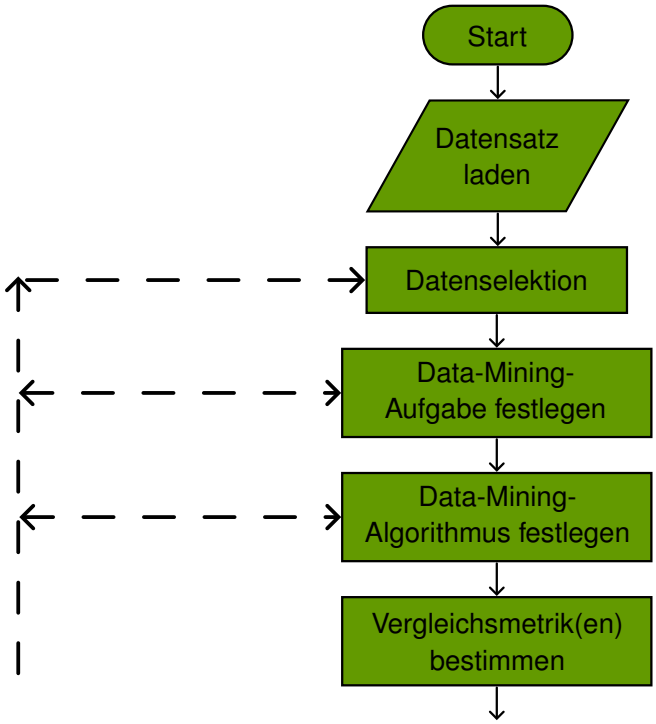

Abbildung 3.3: Start und einleitende Schritte im Experimentierplan

Tabelle 3.3: Voraussetzungen von Data-Mining-Algorithmen: Ausgewählte Beispiele

Data-Mining-Aufgabe	Algorithmus	benötigte Datenstruktur
Klassifikation	Naive Bayes	nominale oder ordinale Merkmale Unabhängigkeit zwischen Merkmalen
Regression	KNN	metrische Merkmale einheitliche Skalierung für bessere Ergebnisse
Clusteranalyse	DBSCAN	metrische Merkmale normalisierte Merkmale nicht zu hohe Dimensionalität
Assoziationsregeln	Apriori	binäre Merkmalsausprägung

Die Tabelle 3.3 ist mit ein paar Beispielalgorithmen in Kombination mit einer Data-Mining-Aufgabe gefüllt und listet die benötigte Datenstruktur für diese Kombination auf. Die Data-Mining-Aufgaben können auch mit anderen Algorithmen bearbeitet werden und die Algorithmen können auch für andere Aufgaben benutzt werden (vgl. Kapitel 2.1.4). Die Tabelle 3.3 soll dem Anwender als Unterstützung dienen und aufzeigen, welche Voraussetzungen an die Datenstruktur gestellt werden können. Die Dokumentation der Voraussetzungen muss nicht zwingend in Form einer Tabelle erfolgen.

Um Anforderung 8 zu erfüllen, müssen nach der Dokumentation der Voraussetzungen auch die Eigenschaften von den Daten abgefragt und dokumentiert werden, die für einen Abgleich von Eigenschaften und Voraussetzungen notwendig sind. Als Anhaltspunkt wird dafür die Tabelle 3.4 vom Experimentierplan bereitgestellt.

Tabelle 3.4 beinhaltet drei Spalten für die Eigenschaften oder die Datenstruktur, den Bereich, in dem die Eigenschaften auftreten, und eine Spalte für Anmerkungen. Sie beinhaltet Beispiele aus einem

Tabelle 3.4: Eigenschaften des Datensatzes ausgefüllt mit Beispielen

Eigenschaftsbereich	Eigenschaften / Datenstruktur	Anmerkungen
ganzer Datensatz	Big-Data-Eigenschaften	z. B. Volume, Variety, Velocity
ganzer Datensatz	hochdimensionale Daten	Datenreduktion anwenden
Merkmal 0	Zeitstempel	Datentyp: String
Merkmal 7	WekstückID	Datentyp: Int64
Merkmale 2 und 4	ordinale Daten	
Merkmale 1 und 3	metrische Daten	
Merkmale 5 und 6	binäre Daten	
Merkmale 3 und 4	Fehlende Werte über 70 %	
Merkmale 1,2,5 und 6	Fehlende Werte unter 10 %	

imaginären Datensatz, um zu zeigen, wie eine Dokumentation der Eigenschaften eines Datensatzes aussehen könnte. In dieser Tabelle sind die Beispiele der generellen Datenstrukturen Kapitel 2.1.1 und die der Eigenschaften Kapitel 2.3.3 entnommen worden. Es können Eigenschaften vom gesamten Datensatz oder nur von bestimmten Merkmalen für Anforderung 8 wichtig sein und in der Tabelle eingetragen werden. Die Eigenschaften sollen dem Anwender eine Hilfestellung und einen Überblick geben, welche Datenvorverarbeitungsschritte nötig sind und welche Merkmale dafür in Frage kommen. In der Spalte für Anmerkungen können zusätzliche Informationen eingetragen werden, die zum Beispiel bei der Datenvorverarbeitung, die nicht im SVP untersucht wird, helfen.

Um Anforderung 9 zu berücksichtigen, sollte in Tabelle 3.4 ebenso dokumentiert werden, falls der Datensatz Big-Data-Eigenschaften aufweist, weil dies Auswirkungen auf die Datenvorverarbeitung haben kann. Besonders große Datenmengen, die eine Big-Data-Eigenschaft darstellen, haben Einfluss auf die Auswahl von Datenvorverarbeitungsmethoden, die im SVP untersucht werden. Es wird von großen Datenmengen gesprochen, wenn die Datenmenge in Tabellenform fünf Gigabyte übersteigt, wie in Kapitel 2.3.3 festgelegt wurde.

Vor dem Abgleich, den 8 vorsieht, von Eigenschaften und Voraussetzungen, wird mit einer Verzweigung überprüft, ob fehlende Werte vorhanden sind. Sollte dies nicht der Fall sein, werden in einem Zwischenschritt fehlende Werte nach dem MAR-, MCAR- oder MNAR-Ausfallmechanismus erzeugt. Diese werden in Kapitel 2.2.1 erläutert. Die Entscheidung zur Auswahl des Ausfallmechanismus und wie viele Werte entfernt werden, obliegt dem Anwender. Es wird jedoch empfohlen, MAR oder MNAR zu wählen, da diese in der Praxis häufiger vertreten sind (vgl. Kapitel 2.2.1). Die Ergebnisse des Experimentierplans dienen in diesem Fall nur der Information und sind weniger relevant, da alle Werte vorhanden sind. Die Abfrage und anschließende künstliche Erzeugung fehlender Werte ist dennoch im Experimentierplan enthalten, um die Methoden zur Behandlung von fehlenden Werten anwenden zu können und somit Anforderung 1 zu erfüllen. Nach dem Erzeugen der fehlenden Werte muss die Dokumentation der Eigenschaften aktualisiert werden, da die Eigenschaften des Datensatzes geändert werden (vgl. Tabelle 3.4). Im Anschluss geht es im Ablaufplan des Experimentierplans weiter, als ob fehlende Werte direkt im Datensatz vorhanden gewesen wären.

Die nächste Operation im Experimentierplan ist das Abgleichen der Voraussetzungen, die der Data-Mining-Algorithmus fordert, mit den Eigenschaften des Datensatzes. Dafür werden die beiden Dokumentationen verwendet, die mithilfe von Tabelle 3.3 und Tabelle 3.4 angefertigt wurden. Mit den

Informationen aus dem Abgleich kann der Anwender die Datenvorverarbeitungsschritte, die notwendig für die Data-Mining-Phase sind und nicht untersucht werden sollen, auf den Datensatz anwenden und damit eine grundlegende Datenbasis für die Untersuchungen schaffen, wie in Anforderung 7 gefordert wird. Dazu zählen auch strukturelle Anpassungen, die in Kapitel 2.2.3 vorgestellt werden. Diese zählen zwar zu den Datentransformationen, bieten im Vergleich jedoch keinen Mehrwert und müssen für das Data Mining angewendet werden. Mit dieser Operation beginnt die Datenvorverarbeitung, dem nächsten Schritt im KDD-Prozess aus Kapitel 2.1.3, dessen Vorgehensmodell als Vorlage für die Struktur des Experimentierplans dient (vgl. Anforderung 5). Außerdem ist dieser Abgleich eine Bedingung für die Auswahl der benötigten Datentransformationsaufgaben, damit später die verschiedenen Methoden der benötigten Aufgaben untersucht werden können. Der aktuelle Stand dieses Datensatzes wird im nächsten Schritt als Ausgangsdatensatz für die SVP gespeichert, um diesen Stand vor jedem Versuchslauf abrufen zu können. Zusammenfassend werden alle Operationen, Verzweigungen und Datenausgaben von der Dokumentation der Voraussetzungen des Data-Mining-Algorithmus bis zur Speicherung des Ausgangsdatensatzes in Abbildung 3.4 als zweiter Teil des Ablaufplans dargestellt.

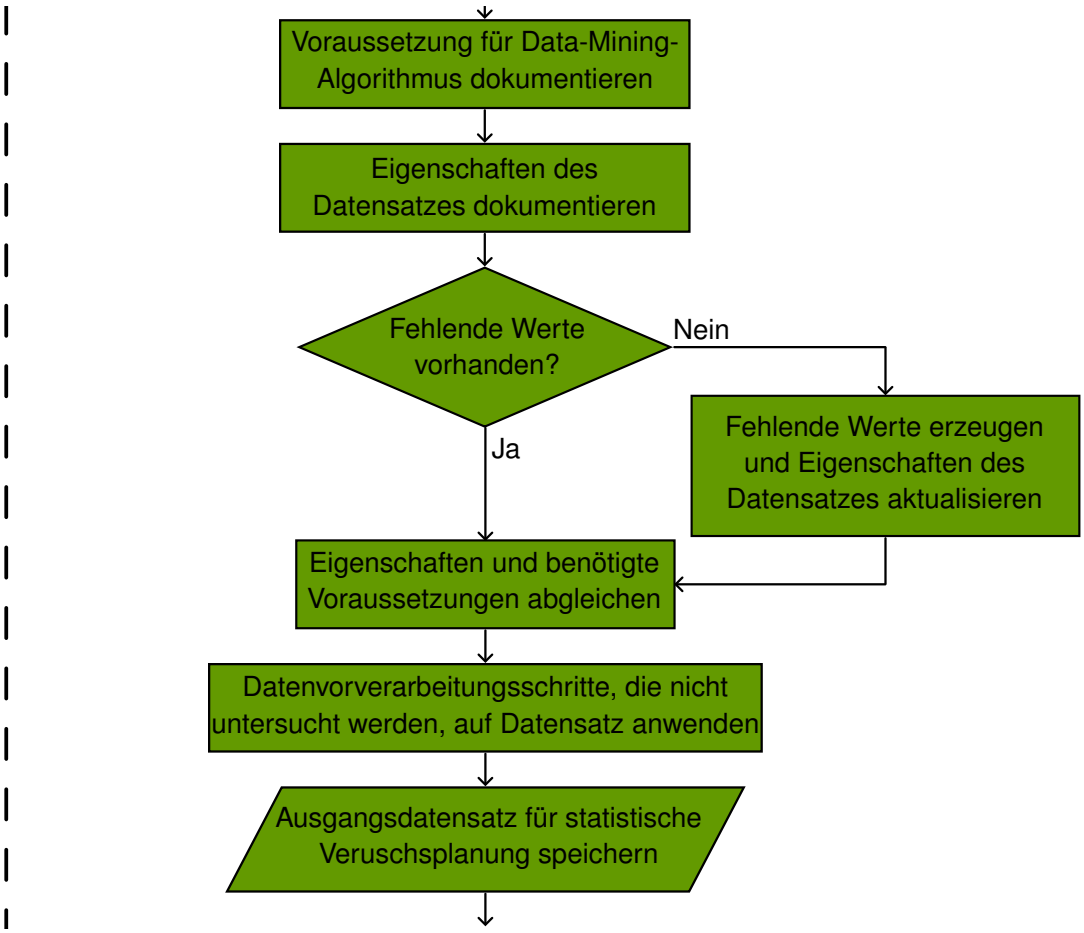

Abbildung 3.4: Abgleich der Eigenschaften und Voraussetzungen und Datenvorverarbeitung im Experimentierplan

In Abbildung 3.4 wird der zweite Teil des Ablaufplans vom Experimentierplan dargestellt. Dieser behandelt die Dokumentation von Voraussetzungen für den Data-Mining-Algorithmus und den Eigenschaften des Datensatzes. Sollten keine fehlenden Werte vorhanden sein, werden diese künstlich erzeugt. Nachfolgend können die Eigenschaften und die Voraussetzungen abgeglichen werden. Mit

den Informationen daraus ist es möglich, Datenvorverarbeitungsschritte auf den Datensatz anzuwenden, die nicht untersucht werden, und den dann vorliegenden Datensatz als Ausgang für den SVP zu speichern. Außerdem wird der Abgleich für die Auswahl der zu untersuchenden Datenvorverarbeitungsmethoden im Experimentierplan wieder aufgegriffen. Zusätzlich ist mit der gestrichelten Linie die Fortsetzung der Rückführung im Experimentierplan zu erkennen.

Nachdem der Ausgangsdatensatz für das Experiment fertiggestellt wurde, müssen die Methoden ausgewählt werden, die untersucht werden sollen, um Anforderung 1 zu erfüllen. Für die Behandlung fehlender Werte und die Datentransformation wird der Experimentierplan jeweils eine Vorauswahl bereitstellen, die ein Spektrum an unterschiedlichen Methoden abdeckt. Zuerst werden die Methoden zur Behandlung fehlender Werte ausgesucht. Die Vorauswahl dafür ist in Tabelle 3.5 aufgelistet.

Tabelle 3.5: Vorauswahl an Methoden zur Behandlung von fehlenden Werten für das Experiment

Behandlungsart	Methode zur Behandlung fehlender Werte
Löschen	Löschen von Merkmalen Löschen von Datentupeln
Imputation Lageparameter	Imputation des Mittelwert Imputation des Median Imputation des Modus
Modellgestützte Imputation	Lineare Interpolation (Bei Zeitreihen) KNN-Imputation MICE MissForest Regressionsimputation GAIN-Imputation KNN-LI (für Big Data > 5 GB)

In Tabelle 3.5 werden ausgesuchte Methoden zur Behandlung fehlender Werte zusammen mit ihrer Behandlungsart aufgelistet. Die Behandlungsart wird aufgeführt, damit der Anwender bei der Auswahl auf die verschiedenen Funktionsweisen der Methoden hingewiesen wird und eine möglichst diverse Auswahl an Methoden treffen kann. Eine Erklärung zur Funktionsweise der Methoden ist im Kapitel 2.2.2 hinterlegt. Der Anwender kann auch andere Methoden zur Behandlung von fehlenden Werten, die nicht in der Vorauswahl stehen, für das Experiment auswählen.

Da die Experimentdauer von der Anzahl der Faktoren abhängt und damit auch von der Anzahl der durchgeführten Methoden, sollte der Anwender die Menge der zu vergleichenden Methoden bedacht auswählen und darauf achten, alle Behandlungsarten zu untersuchen. Da die Methoden der modellgestützten Imputation, die auf verschiedenen Modellen beruhen, zueinander auch sehr unterschiedlich sind, sollten von diesen immer mehrere untersucht werden (vgl. Kapitel 2.2.2). Falls die Methoden der modellgestützten Imputation eine Abhängigkeit zu anderen Merkmalen im Datensatz ausnutzen, muss sichergestellt werden, dass alle Methoden zu den gleichen Merkmalen eine Abhängigkeit herstellen, damit ein Vergleich der Methoden untereinander möglich ist. In diesem Experimentierplan werden alle Merkmale mit fehlenden Werten mit derselben Methode behandelt, um die Anzahl der Faktoren im Experiment überschaubar zu halten. Außerdem muss stets eine

Methode zur Behandlung fehlender Werte angewendet werden, um die Anforderung 10 zu erfüllen. Andernfalls ist es möglich, dass der Data-Mining-Algorithmus zur Bewertung nicht angewendet werden kann (vgl. Kapitel 2.1.4). Sollte der Datensatz die Eigenschaften von Big Data aufweisen, ist in Tabelle 3.5 eine spezielle Methode dafür aufgelistet, um Anforderung 9 zu erfüllen. Alternativ ist auch die Verwendung von MapReduce mit einer anderen Methode eine Hilfe, um mit großen Datenmengen umzugehen, wie in Kapitel 2.3.3 erläutert wird.

Anschließend werden die Methoden der Datentransformation ausgewählt. Die Methoden und Aufgaben der Datentransformation in der Vorauswahl sind dem Kapitel 2.2.4 entnommen worden. Die Vorauswahl der Methoden ist in Tabelle 3.6 mit den dazugehörigen Datentransformationsaufgaben aufgelistet.

Tabelle 3.6: Vorauswahl an Methoden zur Datentransformation für das Experiment

Datentransformationsaufgabe	Datentransformationsmethode
Skalierung: Normalisierung	Min-Max Normalisierung Vektornormierung mit euklidischer Norm (Normalizer)
Skalierung: Standardisierung	Z-Score Median-basierte Skalierung (RobustScaler)
Datenglättung	Gleitender Mittelwert (Moving-Average) Exponentielle Glättung
Aggregation	Durchschnittsbildung Summenbildung
Merkmalskonstruktion	Interaktionen zwischen Merkmalen berechnen Logarithmische oder exponentielle Transformation
Diskretisierung	Intervalle mit gleicher Breite Intervalle mit gleicher Häufigkeit
Datencodierung	One-Hot-Encoding Label-Encoding

In Tabelle 3.6 werden von jeder Datentransformationsaufgabe jeweils zwei Datentransformationsmethoden aufgelistet, damit ein Vergleich der Methoden einer Aufgabe in der Vorauswahl sichergestellt wird. Die Datentransformationsaufgaben, die aufgelistet werden, sind Normalisierung, Standardisierung, Datenglättung, Aggregation, Merkmalskonstruktion, Datencodierung und Diskretisierung. Abschließend wird darauf hingewiesen, dass diese Tabelle nur eine Vorauswahl darstellt und es noch mehr Datentransformationsaufgaben gibt und demnach auch noch deutlich mehr Datentransformationsmethoden.

Die Auswahl der Methoden erfolgt nach erneuter Betrachtung des Abgleichs der Voraussetzungen des Data-Mining-Algorithmus, die zum Start des Experimentierplans dokumentiert wurden, mit den Eigenschaften des Datensatzes, die in Tabelle 3.4 eingetragen wurden. Dadurch wird deutlich, welche Datentransformationsaufgaben erforderlich sind, um Anforderung 10 zu erfüllen. Demzufolge sind die Methoden dieser erforderlichen Aufgaben auf die entsprechenden Merkmale anzuwenden und der Experimentierplan hat sicherzustellen, dass diese Aufgaben in jedem Versuchslauf erfüllt werden. In Tabelle 3.6 sind für jede Datentransformationsaufgabe mindestens zwei Methoden aufgelistet, um selbst bei den erforderlichen Datentransformationen einen Vergleich zu ermöglichen.

Darüber hinaus ist es dem Anwender möglich und explizit erwünscht, unterschiedliche Methoden der Datentransformation auf verschiedene Merkmale anzuwenden, bei denen er einen Mehrwert durch diese Datentransformation vermutet. Die Kombination aus Datentransformation und Merkmal ist im Experiment ein Faktor. Eine Datentransformationsmethode kann auf verschiedene Merkmale angewendet werden und bildet somit verschiedene Faktoren für das Experiment. Bei den Datentransformationen wird keine Methode vorgeschlagen, die besonders für Big Data geeignet ist (vgl. Kapitel 2.2.4). Um die Anforderung 9 trotzdem zu erfüllen, wird an dieser Stelle für diese Eigenschaft des Datensatzes ebenso die Verwendung von MapReduce in Verbindung mit einer Datentransformationsmethode empfohlen, wie in Kapitel 2.3.3 vorgestellt wurde.

Nachdem alle Methoden zur Behandlung fehlender Werte und zur Datentransformation ausgesucht wurden, müssen diese in Abhängigkeit von ihrem Anwendungszeitpunkt gruppiert werden, um Anforderung 13 zu erfüllen. Der Anwendungszeitpunkt unterscheidet sich in eine Anwendung der Datenvorverarbeitungsmethoden unmittelbar vor oder nach dem Train-Test-Split. Dies sind die beiden Gruppen, in denen die Methoden eingeteilt werden. Die Anforderung besteht, weil ein Train-Test-Split für die Bewertung der Modellgüte in der Data-Mining-Phase erforderlich ist (vgl. Kapitel 2.2.5). Dieses Vorgehen wird in Anforderung 2 für den Experimentierplan festgelegt. Auf diese beiden Gruppen wird in der Implementierung des Experimentierplans zurückgegriffen, damit die Methoden zum richtigen Zeitpunkt auf den kompletten Datensatz oder auf den Train- und den Test-Split des Datensatzes angewendet werden.

Im Experimentierplan wird auf die SVP zurückgegriffen, um Anforderung 3 zu erfüllen. Wichtige Grundlagen zur SVP werden in Kapitel 2.1.2 erläutert. Im Experimentierplan wird mit der Tabelle 3.7 eine Vorlage bereitgestellt, die für den Experimentaufbau genutzt werden kann.

Tabelle 3.7: Vorlage für statistischen Versuchsplan

Faktoren:	Methode 1	Methode 2	Zielgröße
	-	-	2
	+	-	3
	-	+	4
	+	+	5
Mittelwert + Mittelwert -	$\frac{3+5}{2} = 4$ $\frac{2+4}{2} = 3$	$\frac{4+5}{2} = 4,5$ $\frac{2+3}{2} = 2,5$	
Effekt:	$4 - 3 = 1$	$4,5 - 2,5 = 2$	

In der Vorlage, die in Tabelle 3.7 dargestellt wird, müssen alle Faktoren eingetragen werden, die untersucht werden sollen. Die Faktoren bestehen aus den Methoden und den Merkmalen, auf denen sie angewendet werden. Wenn die Methode auf den kompletten Datensatz angewendet wird, ist es ausreichend, nur die Methode als Faktor zu benutzen. Die Zielgröße wird mit der ausgewählten Vergleichsmetrik ausgefüllt. Es ist auch möglich, mehrere Vergleichsmetriken oder eine Kombination mehrerer Vergleichsmetriken zu benutzen. Die Tabelle 3.7 ist mit einem Beispiel, bestehend aus zwei Methoden, der Zielgröße und der Effektberechnung gefüllt. Die Anwendung einer Datenvorver-

arbeitungsmethode auf ein Merkmal wird im SVP mit einem Plus (+) gekennzeichnet. Sollte sie nicht angewendet werden, wird dies mit einem Minus (-) im SVP vermerkt. Dies sind die beiden Faktorstufen im SVP. Die Zeilen im SVP zeigen somit, welche Methoden in einem Versuchslauf angewendet werden und welche nicht, und wie sich dies auf die Zielgröße auswirkt. Der Effekt errechnet sich aus der Differenz der Mittelwerte der Zielgrößen, wenn die Methode angewendet wird, im Vergleich zum Weglassen der Methode (vgl. Kapitel 2.1.2).

Mit der Tabelle 3.7 kann der SVP erstellt werden. Dabei ist es wichtig, auf die Randbedingungen zu achten, die in Anforderung 11 oder Anforderung 12 gestellt werden. Zusätzlich können Aufgaben der Datentransformation erforderlich sein, um den Data-Mining-Algorithmus anzuwenden. In diesem Fall muss auch dies berücksichtigt werden, um Anforderung 10 zu erfüllen. Aus diesen Gründen wird empfohlen, ein Computerprogramm zu verwenden, das bei der Erstellung eines SVP mit Randbedingungen unterstützt. Die Tabelle 3.7 sollte anschließend in digitaler Form ausgefüllt vorliegen, sodass ein Datenverarbeitungsprogramm auf die Inhalte zugreifen kann und das Experiment automatisiert abläuft. Die Auswahl und Gruppierung der Methoden und anschließende Erstellung des SVP unter Berücksichtigung der Randbedingungen wird in Abbildung 3.5 als nächster Teil des Ablaufplans vom Experimentierplan abgebildet.

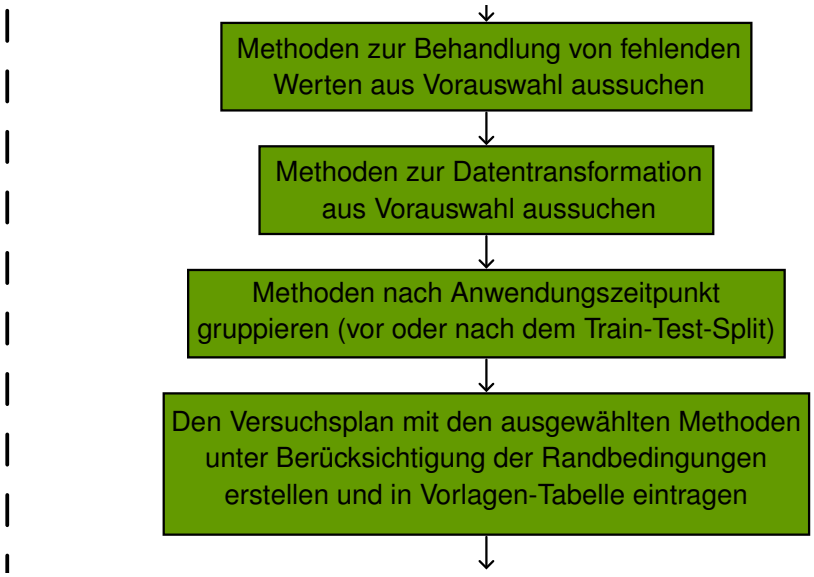

Abbildung 3.5: Auswahl der Methoden für das Experiment

Abbildung 3.5 zeigt die nächsten vier Operationen im Ablaufplan vom Experimentierplan, die den Anwender dazu auffordern, die Datenvorverarbeitungsmethoden zur Untersuchung auszusuchen, diese dann in Abhängigkeit von ihrem Anwendungszeitpunkt zu gruppieren und den SVP unter Berücksichtigung der Randbedingungen und in Anlehnung an die Vorlage aus Tabelle 3.7 zu erstellen. Außerdem ist auch in diesem Teil des Ablaufplans die Rückführung wieder mit einer gestrichelten Linie angedeutet.

Nach der Erstellung des SVP muss eine initiale Überprüfung der Ergebnisse erfolgen, um sicherzustellen, dass das Experiment aussagekräftige Ergebnisse liefert. Sollte dies nicht der Fall sein, muss der Anwender wieder zu einem früheren Zeitpunkt im Experimentierplan geleitet werden, um Änderungen vorzunehmen. Die Möglichkeit der Rückführung wird von Anforderung 6 gefordert. Auch

im Vorgehensmodell zum KDD-Prozess aus Kapitel 2.1.3, an das sich der Experimentierplan aufgrund von Anforderung 5 anlehnt, ist eine Rückkehr in vorherige Schritte möglich. Für die Überprüfung muss zunächst der Ausgangsdatensatz für das Experiment aufgerufen werden. Damit der Data-Mining-Algorithmus auf den Datensatz angewendet werden kann, müssen eventuell Datenvorverarbeitungsmethoden, die noch untersucht werden sollen, angewendet werden (vgl. Anforderung 10). Sollte keine Datenvorverarbeitungsmethode erforderlich sein, ist es auch möglich, keine Methode anzuwenden. Zu diesen Methoden zählen für die Behandlung fehlender Werte zum Beispiel das Löschen von Merkmalen oder Datentupeln mit fehlenden Werten oder die Imputation eines festen Wertes, wie der Null (vgl. Kapitel 2.2.2). Sollte es dazu kommen, dass die initiale Überprüfung aufgrund der Rückführung im Experimentierplan mehrmals durchlaufen wird, kann die Datenvorverarbeitung angepasst werden, bis die Ergebnisse passen, oder die Daten, die Data-Mining-Aufgabe oder der Data-Mining-Algorithmus geändert werden.

Auf diesen hinreichend vorverarbeiteten Datensatz wird der Data-Mining-Algorithmus unter Berücksichtigung der Data-Mining-Aufgabe angewendet. Anschließend wird das Ergebnis des Algorithmus mithilfe der festgelegten Vergleichsmetriken bewertet, sodass ein Referenzwert für die Modellgüte des Algorithmus errechnet wird. Zuerst muss entschieden werden, ob die Ergebnisse allgemein sinnvoll sind und somit Anforderung 14 erfüllt wird. Sollte dies nicht der Fall sein, wird der Anwender gemäß der Anforderung zurückgeleitet, um die Datenvorverarbeitung anzupassen oder am Start einen anderen Data-Mining-Algorithmus mit einer potenziell anderen Data-Mining-Aufgabe auszuwählen, um Wissen in den Daten zu entdecken (vgl. Abbildung 3.3). Ob die Ergebnisse des Data Mining sinnvoll sind, hängt von der Data-Mining-Aufgabe ab und ist vom Anwender des Experimentierplans zu entscheiden.

Im Anschluss muss Anforderung 15 geprüft werden, um sicherzustellen, dass durch eine Datenvorverarbeitung die Ergebnisse des Data Mining messbar verbessert werden können. Dafür muss der Referenzwert von den Vergleichsmetriken, der bei der initialen Anwendung des Data-Mining-Algorithmus bestimmt wurde, vom Anwender des Experimentierplans bewertet werden. Sollte die Modellgüte ohne gründliche Datenvorverarbeitung sehr gut sein, sollte dies durch Anpassungen an der Datenselektion, der Data-Mining-Aufgabe und am Data-Mining-Algorithmus gelöst werden (vgl. Abbildung 3.3). Eine mögliche Erklärung dafür ist das Overfitting, welches auch mit diesen Maßnahmen behandelt werden kann (vgl. Kapitel 2.1.4). Aus diesem Grund fordert auch Anforderung 15 eine mögliche Rückführung im Experimentierplan an dessen Start.

Sobald Anforderung 15 und Anforderung 14 erfüllt sind, wird die initiale Überprüfung als erfolgreich angesehen und der Anwender kann das Experiment mit dem SVP durchführen. Sollte der Experimentierplan jedoch nach einigen Durchläufen und Anpassungen der Datenvorverarbeitung, der Datenselektion und der Data-Mining-Phase immer noch an diesen Bewertungen scheitern, wird dem Anwender an dieser Stelle mitgeteilt, dass die Durchführung des Experimentierplans nicht sinnvoll ist, da die Datenvorverarbeitung ohne funktionierendes Data Mining nicht benötigt wird und ein Vergleich der Methoden somit sinnlos ist. Zusammengefasst wird die initiale Überprüfung in Abbildung 3.6.

Der nächste Teil des Ablaufplans wird in Abbildung 3.6 gezeigt und behandelt die initiale Überprüfung der Ergebnisse, die der Data-Mining-Algorithmus auf dem Datensatz erzeugt. Dafür wird zuerst

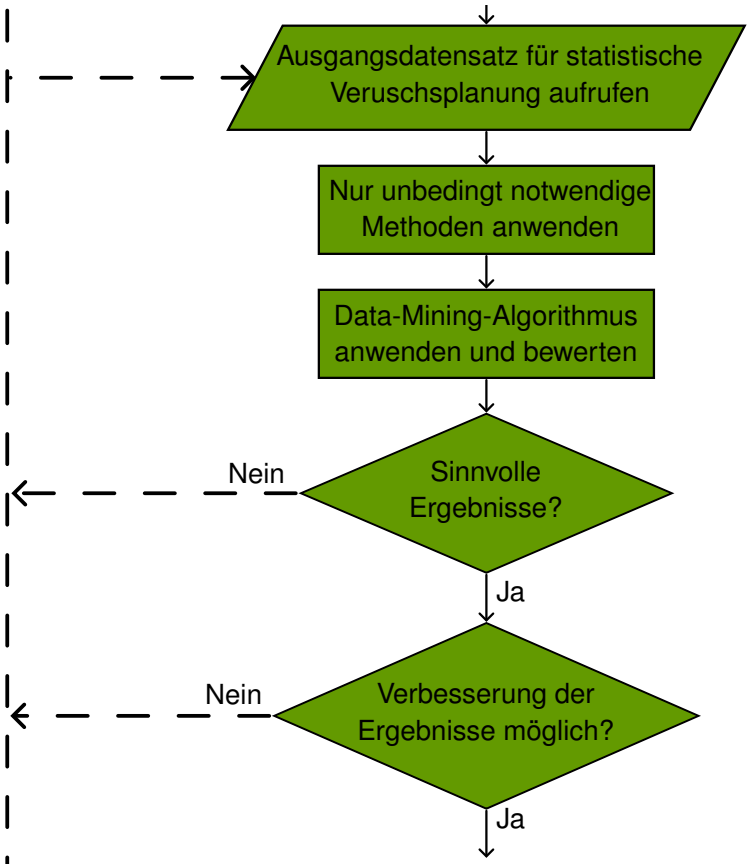

Abbildung 3.6: Initiale Überprüfung der Ergebnisse des Data-Mining-Algorithmus

der Ausgangsdatensatz aufgerufen, auf den im Anschluss nur notwendige Datenvorverarbeitungsmethoden angewendet werden. Danach kann der Data-Mining-Algorithmus angewendet und dessen Ergebnisse, in Form von Mustern und Modellen, mit der Vergleichsmetrik bewertet werden. Abschließend können die Ergebnisse des Data-Mining-Algorithmus und das Ergebnis der Vergleichsmetrik in zwei Vergleichen im Ablaufplan bewertet werden. Sollte einer der beiden Vergleiche mit *Nein* beantwortet werden, erfolgt eine Rückführung zum Anfang der initialen Überprüfung oder zum Start. Die komplette Verbindung der Rückführung wird im gesamten Ablaufplan in Anhang A dargestellt.

Im letzten Abschnitt des Experimentierplans wird das Experiment durchgeführt. Dafür wird die SVP benutzt, die angelehnt an Tabelle 3.7 die Versuchsläufe dokumentiert. Eine Zeile der Tabelle entspricht einem Versuchslauf. Um alle Versuchsläufe durchzugehen, wird eine Schleife im Ablaufplan verwendet. Bei einer Implementierung des Experimentierplans wird empfohlen, ebenso auf eine Zähl- oder While-Schleife zurückzugreifen.

Der Ablauf jedes Schleifendurchgangs ist gleich und wird nachfolgend erläutert. Zuerst wird der Ausgangsdatensatz aufgerufen, der für die SVP vorbereitet wurde, um Anforderung 7 zu erfüllen. Im Anschluss werden die für diesen Versuchslauf vorgesehenen Datenvorverarbeitungsmethoden auf den Datensatz angewendet. Diese sind mit einem (+) gekennzeichnet. Es ist darauf zu achten, dass aufgrund von Anforderung 13 einige Methoden vor und einige nach dem Train-Test-Split, der demnach während der Datenvorverarbeitung in dieser Schleife erstellt wird, angewendet werden. Innerhalb der Anwendung der Methoden vor oder nach dem Train-Test-Split sollte probiert werden, die fehlenden Werte zuerst zu behandeln, da einige Methoden der Datentransformation einen voll-

ständigen Datensatz benötigen, wie in Kapitel 2.2.3 erläutert wird. Fehlende Werte können zum Beispiel nicht normalisiert oder diskretisiert werden. Anschließend werden die vorgesehenen Datentransformationsmethoden angewendet.

Aufgrund der Erfüllung von Anforderung 14 erfolgt, wie bereits im initialen Durchlauf, nachfolgend die Anwendung des Data-Mining-Algorithmus. Das erstellte Modell unterliegt einer Bewertung anhand der definierten Vergleichsmetriken. Das Vorgehen bei der Bewertung der Modellgüte des Data-Mining-Algorithmus wird in Kapitel 2.2.5 erläutert. Die daraus resultierenden Ergebnisse sind in die aktuelle Zeile der Tabelle 3.7 einzutragen. Im Anschluss wird, wenn es durch die SVP weiterhin vorgesehen wird, in die nächste Zeile der Tabelle gewechselt und die Schleife beginnt von vorne. Nach dem Durchlauf der letzten Zeile ist die Tabelle gefüllt und die Schleife wird verlassen.

Im Anschluss an das Experiment kann die gefüllte Tabelle 3.7 für eine Auswertung genutzt werden. Mit dieser können die Effekte der Faktoren berechnet werden, wie in Kapitel 2.1.2 dargestellt wird. Damit lassen sich die Datenvorverarbeitungsmethoden mit dem größten Einfluss auf die Data-Mining-Aufgabe bestimmen und untereinander vergleichen, wie der Titel der Arbeit fordert. Sollte ein vollfaktorieller Versuchsplan angewendet worden sein, ist allein mit der Zielgröße ebenso die beste Kombination an zuvor ausgewählten Datenvorverarbeitungsmethoden der Behandlung von fehlenden Werten und der Datentransformation zu bestimmen. Bei anderen Versuchsplänen ist die Anwendung der besten Kombination nicht garantiert und kann nur anhand der berechneten Effekte geschätzt werden. An dieser Stelle im Experimentierplan besteht die Möglichkeit, mithilfe des neu erlangten Wissens weitere Versuchsläufe durchzuführen, um die beste Kombination dieser Methoden zu verifizieren. Dieser Schritt ist optional, da es hierbei nicht mehr nur um den Vergleich der Methoden geht. Sollten die Ergebnisse nicht ausreichen, ist es auch am Ende nach der Auswertung möglich, an den Start des Experimentierplans zurückzukehren und Anforderung 6 zu ermöglichen. Die Rückführung vom Ende zum Start ist auch im Vorgehensmodell aus Kapitel 2.1.3 möglich, an das sich der Experimentierplan aufgrund von Anforderung 5 orientiert. Nach der Auswertung endet der Ablaufplan des Experimentierplans und ergibt den finalen Abschnitt, der in Abbildung 3.7 dargestellt wird.

In Abbildung 3.7 wird der letzte Teil des Ablaufplans vom Experimentierplan dargestellt. Dieser zeigt die Schleife, die den SVP durchläuft und dabei die Datenvorverarbeitungsmethoden der Versuchsläufe anwendet und die Ergebnisse des nachfolgenden Data-Mining-Algorithmus bewertet und im SVP dokumentiert. Im Anschluss an die Schleife werden die Ergebnisse ausgewertet und die Effekte der Methoden als Faktoren berechnet, um einen Vergleich zu ermöglichen. Der Nutzer hat damit den kompletten Experimentierplan angewendet und kann mit den Ergebnissen fundierte Entscheidungen treffen, welche Datenvorverarbeitungsmethoden zur Behandlung fehlender Werte und zur Datentransformation für die Wissensentdeckung in seiner Produktionsdatenbank angewendet werden sollen.

Abschließend wird der gesamte Experimentierplan in kompakter Form dargestellt, um einen Überblick zu verschaffen. Die zuvor hergeleiteten Abschnitte stellen dabei jeweils die Schritte dar, sodass diese für eine detaillierte Betrachtung weiterhin herangezogen werden können. Der gesamte Experimentierplan in kompakter Form wird in Abbildung 3.8 dargestellt.

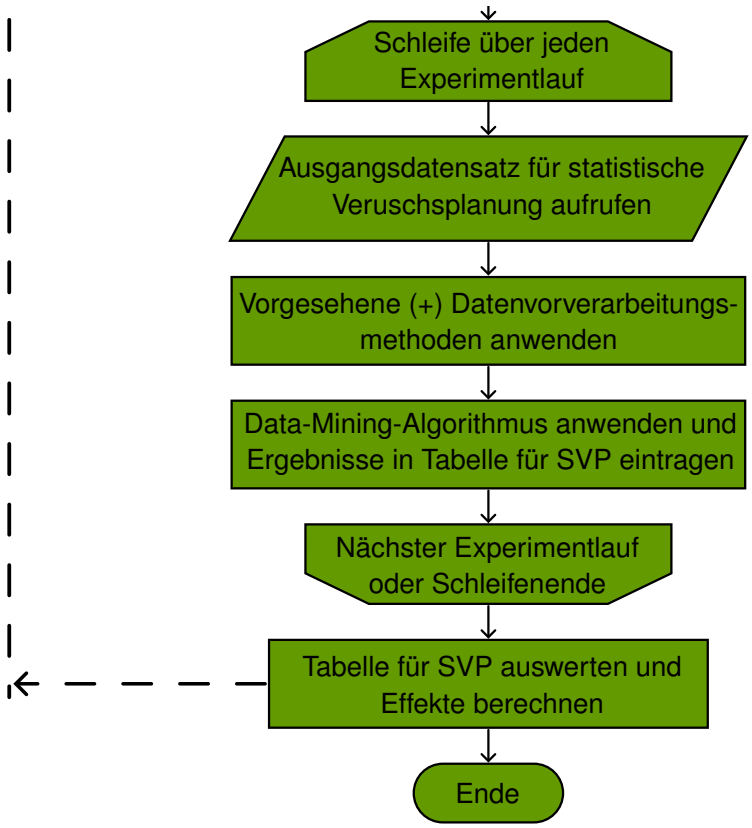

Abbildung 3.7: Durchführung des Experiments im Experimentierplan

Abbildung 3.8 zeigt den kompletten Experimentierplan, der den Anwender dazu anleitet, Datenvorverarbeitungsmethoden zur Behandlung fehlender Werte und zur Datentransformation zu vergleichen. Im gesamten Überblick wird ebenso deutlich, dass Anforderung 6 und Anforderung 5 erfüllt sind, da Iterationen im Experimentierplan möglich sind, wie im KDD-Vorgehensmodell nach Fayyad et al. (1996a), welches in Kapitel 2.1.3 vorgestellt wird.

Im ersten Schritt wird der Datensatz aufgenommen und die umfassende Data-Mining-Aufgabe mit weiteren Spezifikationen, wie dem Algorithmus und der Vergleichsmetrik festgelegt (vgl. Abbildung 3.3). Der zweite Schritt behandelt die Abfrage und Dokumentation der Datensatz-Eigenschaften sowie den Voraussetzungen des Data-Mining-Algorithmus, um den Datensatz auf die Datenvorbereitungsschritte vorzubereiten, die untersucht und verglichen werden sollen. Damit ist ein Abgleich von Voraussetzungen und Eigenschaften möglich (vgl. Abbildung 3.4). Außerdem wird der Abgleich für Entscheidungen im dritten Schritt des kompakten Experimentierplans benötigt, um die Auswahl der Methoden zur Behandlung von fehlenden Werten und zur Datentransformation zu tätigen. Die Ausgewählten Methoden werden im selben Schritt verwendet, um den SVP zu füllen und diesen unter Berücksichtigung der Randbedingungen zu erstellen (vgl. Abbildung 3.5). Im Anschluss kann ein initialer Durchgang durchgeführt und bewertet werden. Für den initialen Durchgang werden nur die nötigsten Datenvorbereitungsmethoden angewendet, damit sinnvolle Ergebnisse erreicht werden können. Diese initiale Datenvorverarbeitung kann innerhalb des Schrittes angepasst werden. Sollten die Ergebnisse auch nach den Anpassungen nicht als Ausreichend bewertet werden, ist eine Rückführung zum Start vorgesehen um dort Anpassungen zu tätigen (vgl. Abbildung 3.6). Im letzten Schritt wird eine Schleife durch die Zeilen des SVP iteriert und die vorgesehenen Methoden

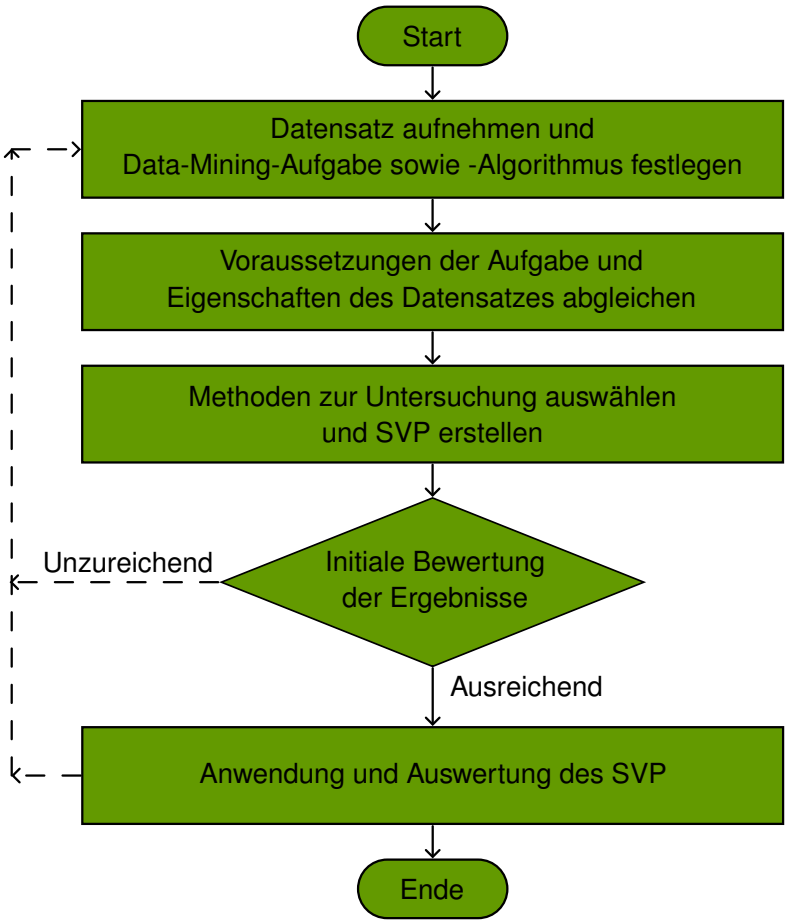

Abbildung 3.8: Experimentierplan in kompakter Darstellung

werden auf den Datensatz angewendet. Nach der Datenvorverarbeitung wird in jedem Schleifendurchgang der Data-Mining-Algorithmus auf die Daten angewendet. Die Ergebnisse des Algorithmus werden anschließend mit der Vergleichsmetrik für jede Zeile des SVP bewertet. Nach Beendigung der Schleife kann der SVP mithilfe der Vergleichsmetriken ausgewertet werden. Dabei wird die Effektberechnung angewendet, um die Methoden untereinander zu vergleichen. Mit diesem Vergleich der Methoden ist das Ende des Experimentierplans erreicht (vgl. Abbildung 3.7).

In diesem Kapitel wurde auf Grundlage der zuvor erhobenen Anforderungen ein Experimentierplan erstellt. Im nächsten Kapitel wird dieser validiert, indem er exemplarisch auf einen Datensatz aus der Produktion angewendet wird.

4 Exemplarische Anwendung des Experimentierplans

In diesem Kapitel wird der zuvor erstellte Experimentierplan exemplarisch angewendet. Dazu wird zunächst ein Datensatz mit Produktionsdaten ausgewählt und vorgestellt. Im Anschluss wird die Implementierung des Experimentierplans in der Programmiersprache Python in der Version 3.13.5 von der Python Software Foundation (2025) durchgeführt und beschrieben. Die Durchführung des Experiments erfolgt auf Grundlage des Datensatzes sowie der Implementierung. Im Anschluss werden die Ergebnisse des Experiments einer abschließenden Auswertung unterzogen. Durch dieses Vorgehen wird eine exemplarische Validierung durchgeführt, die das Vertrauen in den Experimentierplan stärken soll.

4.1 Vorstellung des Produktionsdatensatzes zur exemplarischen Anwendung

Um einen realen Datensatz mit Daten aus der Produktion zu bekommen, wird auf eine frei zugängliche Datenbank zurückgegriffen, die reale Datensätze aus der Produktionsumgebung für Machine-Learning-Aufgaben sammelt und bereitstellt. Zusammengestellt wird diese Übersicht vom Fraunhofer IPT und Fraunhofer FFB (2025). Dort gibt es die Möglichkeit, durch Filter die Suche einzuschränken, weil das Feld der Produktionsumgebung ziemlich groß und vielfältig ist. Um einen passenden Datensatz zu finden, werden nur Datensätze angezeigt, in denen Daten in tabellarischer Form vorliegen. Dies wird gefordert, damit die vorgestellten Grundlagen aus Kapitel 2 auf den Datensatz angewendet werden können. Außerdem werden einige Kategorien aus den Anwendungsgebieten, die nur im entfernteren Sinne mit der Produktion in Verbindung stehen, nicht berücksichtigt, sodass nur Datensätze aus den Anwendungsgebieten *Production Processes* und *Machinery & Equipment* übrig bleiben. Zuerst wurden die 36 tabellarischen Datensätze der Produktionsprozesse betrachtet, von denen 10 Datensätze zur freien Verfügung standen. Von diesen war einer zu groß, um mit der zur Verfügung stehenden Hardware ein Experiment mit vielen Versuchsdurchläufen durchzuführen, ohne sehr lange Rechenzeiten in Kauf zu nehmen. Die neun Übrigen sind genauer betrachtet und einer explorativen Datenanalyse unterzogen worden. Außerdem wurde darauf geachtet, dass fehlende Werte im Datensatz vorhanden sind, damit diese nicht künstlich erzeugt werden müssen. Insbesondere die Ergebnisse eines Datensatzes waren dabei vielversprechend daher wird auf diesen der Experimentierplan angewendet. Im Folgenden wird dieser Datensatz vorgestellt.

Der Datensatz, auf den der Experimentierplan angewendet wird, ist aus einer realen Produktions-

anlage, die in Michigan in den USA steht, und wird von Liveline Technologies (2025) auf Kaggle (2025) zur Verfügung gestellt. Die Daten sind anonymisiert, sodass keine genauere Zuordnung auf das Unternehmen dieser Produktionslinie möglich ist. Die Produktionslinie ist ein kontinuierlicher Produktionsprozess und beinhaltet sowohl parallele als auch serielle Bearbeitungsschritte. Zur Veranschaulichung wird die Linie in Abbildung 4.1 dargestellt.

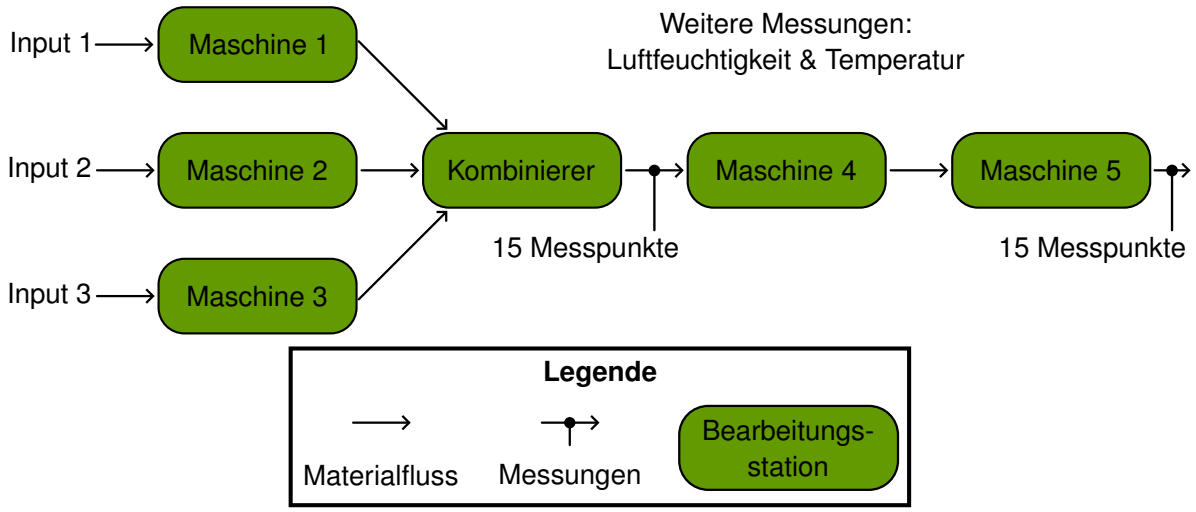

Abbildung 4.1: Aufbau der Produktionslinie aus dem der Datensatz von Liveline Technologies (2025) erhoben worden ist

Die Abbildung 4.1 zeigt eine Produktionslinie, die mit drei parallelen Eingängen in drei verschiedene Maschinen beginnt. Die Eingänge stellen Materialeingänge für die jeweiligen Maschinen dar. Nach den parallelen Maschinen werden die Materialflüsse in einer weiteren Bearbeitungsstation zu einem Materialfluss kombiniert. Am Ausgang des Kombinierers wird der Materialfluss an 15 Messpunkten gemessen. Anschließend wird der Materialfluss in den Maschinen 4 und 5 bearbeitet und nach Maschine 5 erneut an den gleichen 15 Messpunkten gemessen. Die Daten sind für eine Dauer von fast vier Stunden mit der Frequenz von einem Hertz aufgezeichnet worden, sodass insgesamt 14.088 Datenpunkte im Datensatz enthalten sind. Jeder Datenpunkt hat deshalb ein Zeit-Merkmal, in dem die aktuelle Zeit, auf die Sekunde genau, gespeichert wird. Insgesamt weist jeder Datenpunkt 116 verschiedene Merkmale auf, die alle metrisch kontinuierlich ausgeprägt sind. Neben dem Zeitpunkt werden auch die Luftfeuchtigkeit in % und die Temperatur in ℃ als generelle Merkmale aufgenommen, die keinem bestimmten Punkt im Prozess zugeordnet sind. Die 116 Merkmale sind auf unterschiedlichen Werteskalen ausgeprägt.

Für die drei Materialeingänge am Anfang der Produktionslinie liegen jeweils vier Merkmale mit den Materialeigenschaften des rohen Eingangsmaterials vor. Durch die Anonymisierung ist nicht zu erkennen, welche Eigenschaften die metrisch ausgeprägten Merkmale darstellen sollen. Die drei parallelen Maschinen, in denen das Material zuerst bearbeitet wird, speichern zudem jeweils acht Prozessvariablen. Diese beinhalten verschiedene Temperaturen, den Stromverbrauch und die Umdrehungszahl des Motors sowie den Druck, den das Material auf die Maschine ausübt. Am Kombinierer werden drei Temperaturen ausgelesen und gespeichert, während an den Maschinen 4 und 5 jeweils sieben Prozessvariablen im Datensatz dokumentiert werden. Maschine 4 misst den Druck des Materials und verschiedene Temperaturen in der Bearbeitungsstation, während Maschine 5 nur verschiedene Temperaturen misst. Hinzu kommen die 15 Messpunkte, die an zwei Stellen im Material-

fluss erfasst werden. Diese messen einen Abstand in Millimetern. Die Steuerung des Materialflusses gibt außerdem Sollwerte vor, die an den 15 Messpunkten an beiden Stellen in der Produktionslinie erzielt werden sollen. Diese 30 Sollwerte sind ebenso als Merkmale im Datensatz wiederzufinden. Alle Sollwerte werden auf der Prozessleitebene erhoben (vgl. Kapitel 2.3.2)

Der Datensatz wurde mit zwei verschiedenen vorgegebenen Aufgaben von Liveline Technologies (2025) zur Verfügung gestellt. Beide aufgaben können der datengetriebenen Vorhersage der Qualität von Produkten zugeordnet werden (vgl. Kapitel 2.3.4) Die erste Aufgabe besteht darin, die Messungen der ersten 15 Messpunkte auf Basis der zuvor in der Produktionslinie erhobenen Merkmale vorherzusagen. Für die Vorhersage können die generellen Umgebungsparameter, die Materialeigenschaften an den Eingängen sowie alle Prozessparameter der parallelen Maschinen und des Kombinierers verwendet werden. Die zweite Aufgabe sieht eine Vorhersage der Messungen an den zweiten 15 Messpunkten vor. Dafür können wieder die Umgebungsvariablen benutzt werden. Zusätzlich werden die Werte der ersten Messung und die Prozessparameter von Maschine 4 und Maschine 5 berücksichtigt. Damit stehen insgesamt 31 Eingabe-Merkmale und 15 Zielvariablen zur Verfügung. Für jede Zielvariable können alle 31 Eingabe-Merkmale verwendet werden. In dieser Arbeit wird die zweite Aufgabe bearbeitet, da in den ersten 15 Messpunkten einige fehlende Werte vorkommen, die für den Vergleich von Datenvorverarbeitungsmethoden zur Behandlung von fehlenden Werten benötigt werden. Damit liegt ein multivariates Ausfallmuster vor (vgl. Kapitel 2.2.1). Außerdem beschreibt Liveline Technologies (2025) die Daten für die zweite Vorhersage als verrauschter als für die erste Vorhersage, was auch für den Vergleich einiger Datentransformationen einen Vorteil darstellen kann.

Fehlende Werte können, wie in Kapitel 2.2.1 erklärt worden ist, in verschiedenen Formen im Datensatz auftreten. In diesem Datensatz sind fehlende Werte als *0.0* ausgeprägt. Damit ist der Datensatz technisch gesehen schon vollständig, da keine leeren Felder vorhanden sind. In der Aufgabe, die bearbeitet wird, sind nur in den ersten 15 Messpunkten nach dem Kombinierer fehlende Werte. Von diesen besitzen 8 Merkmale eine Ausfallrate von weniger als 6% und 7 Merkmale eine Ausfallrate zwischen 22% und 95% (vgl. Anhang C.1). Es wird also eine große Spanne von wenigen bis vielen fehlenden Werten abgedeckt. In den Zielvariablen haben 13 Messungen eine Ausfallrate von unter 9% und zwei Messungen haben 30% und 91% (vgl. Anhang C.2).

Nachdem der Datensatz vorgestellt wurde, wird im Anschluss die Implementierung des Experimentierplans in Python vorgenommen. Der Experimentierplan wird mit diesem Datensatz angewendet.

4.2 Implementierung des Experimentierplans in Python

In diesem Abschnitt wird die Implementierung des Experimentierplans aus Kapitel 3 mit Python vorgestellt. Die Implementierung ist exemplarisch für den zuvor in Abschnitt 4.1 vorgestellten Datensatz durchgeführt worden. Zusätzlich kann sie als Unterstützung für die Anwendung des Experimentierplans auf andere Datensätze zur Hilfe herangezogen werden. Der Programmcode des Experimentierplans ist in drei Module unterteilt. Das Hauptmodul ist der Experimentierplan, der weitere Funktionen aus den Modulen für die Datenvorverarbeitungsmethoden und die Erstellung des

Experimentierplans importiert. Diese Funktionen und andere Analysen des Datensatzes sind mit einem Jupyter Notebook von Kluyver et al. (2016) getestet und angewendet worden, bevor sie in die Module für den Experimentierplan überführt wurden. Solch ein Notebook eignet sich gut für die Datenanalyse, weil die Ergebnisse zwischengespeichert werden und somit bei Änderungen oder Anpassungen nicht der komplette Code wieder von vorne durchlaufen werden muss.

Zusätzlich sind die frei verfügbaren Bibliotheken Scikit-Learn von Pedregosa et al. (2011), Numpy von Harris et al. (2020) und Pandas von The pandas development team (2025) installiert und verwendet worden, um den Experimentierplan zu implementieren. Scikit-Learn ist eine Bibliothek, die Methoden für den KDD-Prozess bereitstellt. Die bereitgestellten Methoden decken dabei nahezu alle Schritte des gesamten KDD-Prozesses ab. In der vorliegenden Implementierung werden unter anderem Imputationen, Skalierungen, Regressionsalgorithmen und Bewertungsalgorithmen aus dieser Bibliothek eingesetzt. Für Methoden, die zusätzlich zur generellen Datenverarbeitung der Datenmatrix benötigt werden, werden Numpy oder Pandas herangezogen. Zu den betreffenden Befehlen zählen etwa Anweisungen, die einzelne Spalten oder Zeilen der Datenmatrizen aufrufen oder CRUD-Funktionen. Diese sind zum Beispiel für die Transformationen zwischen Datenmatrizen in Python und Comma-Separated-Value-Dateien (CSV-Dateien) zuständig.

Die Implementierung wird nachfolgend mithilfe eines Programmablaufplans dargestellt. Um diesen vom Ablaufplan des Experimentierplans zu unterscheiden, wird dieser in orange anstatt grün dargestellt. Anschließend werden bei der Herleitung ausgewählte Code-Ausschnitte dargestellt. Der komplette Programmcode ist in Anhang B zu finden. Der Aufbau der Implementierung richtet sich nach der Abfolge im Experimentierplan (vgl. Kapitel 3.3 und Anhang A) und wird nachfolgend in Abbildung 4.2 dargestellt.

Abbildung 4.2 zeigt den Ablauf des Experimentierplans auf den Datensatz aus Kapitel 4.1 angewendet und in Python implementiert. Der Ablaufplan benutzt auch die Symbole der *DIN 66001* (1983), weicht aber vor der Durchführung des SVP von der ordnungsgemäßen Darstellung der Norm ab, um Platz zu sparen. Die Reihenfolge der Schritte in der Implementierung des Experimentierplans unterscheidet sich teilweise von der im ursprünglichen Experimentierplan festgelegten Reihenfolge. Dies liegt daran, dass die Implementierung darauf abzielt, die praktische Durchführung anwenderfreundlich und übersichtlich zu gestalten. Im nächsten Abschnitt, in dem der Experimentierplan angewendet wird, wird jedoch wieder die ursprüngliche Reihenfolge des Experimentierplans eingehalten.

Zunächst werden die erforderlichen Funktionen der zuvor angeführten Bibliotheken Scikit-Learn, Numpy und Pandas importiert. Darüber hinaus werden Funktionen aus anderen Modulen importiert, die für die Implementierung programmiert wurden. Im Anschluss wird die CSV-Datei des Datensatzes geladen und mithilfe von Pandas in ein DataFrame überführt. Wie in Kapitel 2.1.1 erwähnt wird, speichert ein Pandas DataFrame Informationen in tabellarischer Form und weist dabei den Zeilen und Spalten Identifikatoren zu, über die diese gezielt aufgerufen werden können. Diese Eigenschaft wird im Verlauf der Implementierung benötigt. Nachdem der Datensatz in Form eines DataFrames vorhanden ist, wird dieses in mehrere kleinere DataFrames aufgeteilt. Die Aufteilung erfolgt nach der Dokumentation des Datensatzes, sodass jeweils für beide Aufgaben die Eingabedaten in Form der Eingabe-Merkmale, der Zielvariablen, der dazugehörigen Sollwerte der Steuerung

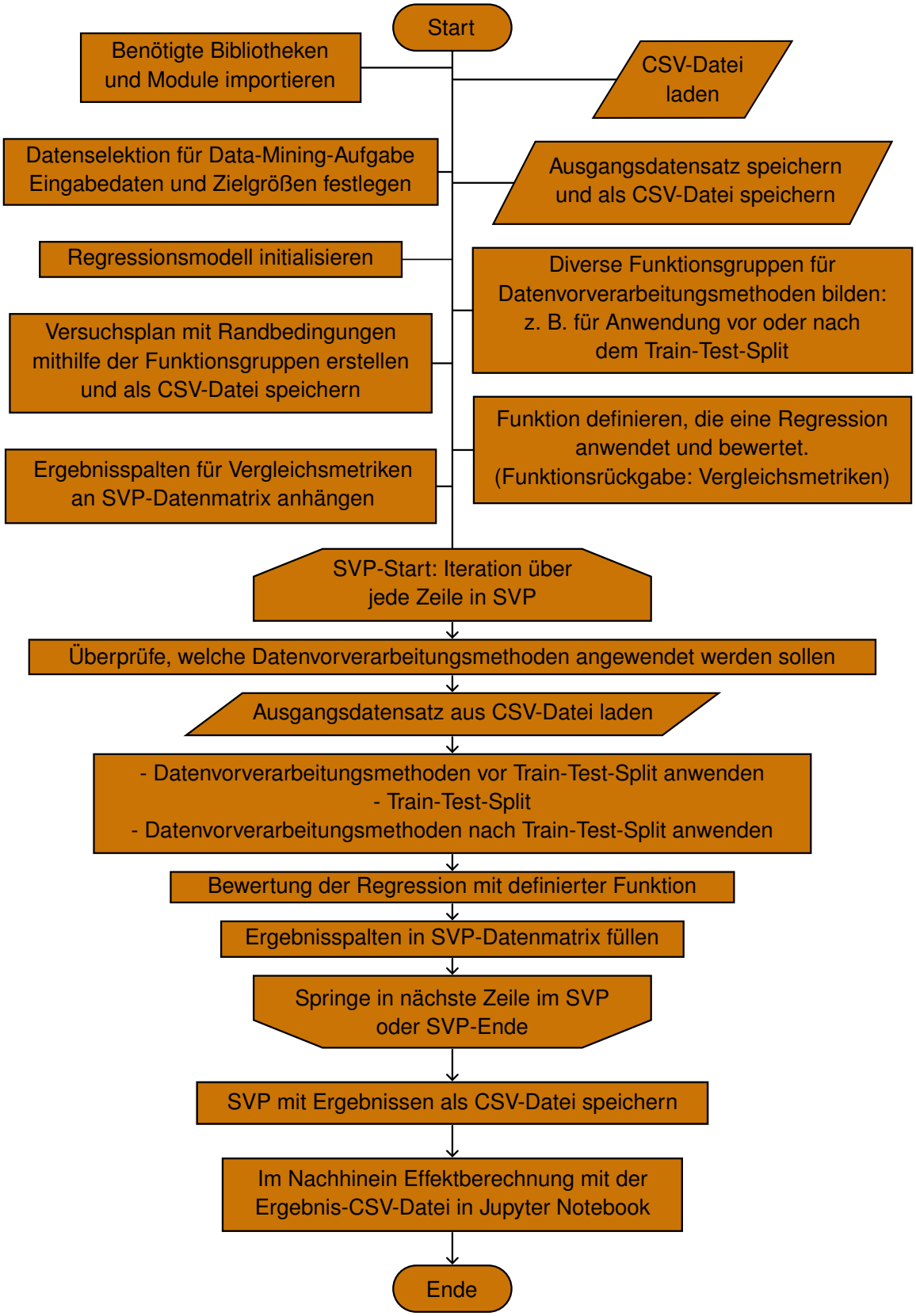

Abbildung 4.2: Programmaufbau der Implementierung des Experimentierplans

und der Zeitpunkte zur Verfügung stehen. Im Rahmen der Datenselektion, die gemäß des Experimentierplans nach dem Laden des Datensatzes erfolgt, werden die für die Data-Mining-Aufgabe erforderlichen Daten selektiert und festgelegt, welche Daten als Eingabedaten und welche als Ziel-

größen fungieren. Darüber hinaus werden weitere Daten als Zusatzinformationen festgelegt, da diese potenziell in der Datenvorverarbeitung benötigt werden. Weitere Datenvorverarbeitungsschritte werden bei diesem Datensatz nicht benötigt, sodass nach der Datenselektion die Daten für den Ausgangsdatensatz zur Verfügung stehen. Der Ausgangsdatensatz, bestehend aus den Eingabedaten, den Zielvariablen und den dazugehörigen Zusatzdaten, wird anschließend in verschiedenen CSV-Dateien gespeichert. Die CSV-Dateien dienen einerseits der Dokumentation des Zwischenstandes und andererseits der Dokumentation der Daten im Experiment. Damit ist die Aufbereitung der Produktionsdaten für den Experimentierplan beendet.

Nach der Aufbereitung der Produktionsdaten werden in den nächsten Schritten in der Implementierung, andere essentielle Initialisierungen vorgenommen (vgl. Abbildung 4.2). Zuerst wird das Regressionsmodell initialisiert, welches für die Data-Mining-Aufgabe als Data-Mining-Algorithmus festgelegt worden ist. In dieser Implementierung wird der *KNeighborsRegressor* von Scikit-Learn verwendet. Dabei handelt es sich um ein Regressionsmodell, das auf dem KNN-Algorithmus basiert. An dieser Stelle besteht die Möglichkeit, das Modell mithilfe von Parametern anzupassen. In dieser Implementierung werden die Standardeinstellungen von Scikit-Learn verwendet. Darüber hinaus müssen Datenvorverarbeitungsmethoden zur Behandlung von fehlenden Werten und zur Datentransformation festgelegt werden. Diese sind in einem anderen Modul in Form von Funktionen definiert und werden zu Beginn des Programmcodes importiert. Für die weitere Verwendung der Datenvorverarbeitungsmethoden ist es nötig, diese in verschiedene Gruppen einzuteilen. Zunächst werden die Funktionen der Methoden in Funktionsgruppen eingeteilt, wobei die Methoden nach den gleichen Ein- und Ausgaben der Funktionen sortiert werden. Dafür wird auf ein Dictionary in Python zurückgegriffen. Die Funktionsgruppen sind: fehlende Werte, Imputation, Skalierung, Aggregation, Datenglättung und Merkmalskonstruktion. Jeder Funktionsgruppe sind die entsprechenden Funktionen der Datenvorverarbeitungsmethoden zugeordnet. Es ist notwendig, die genauen Namen der Funktionen zu verwenden, da diese später über die Namen in den Funktionsgruppen aufgerufen werden. Eine weitere Gruppierung der Funktionen erfolgt nach der Anwendung vor oder nach dem Train-Test-Split (vgl. Anforderung 13). Zu diesem Zweck werden zwei Listen erstellt, die alle Funktionen einer Gruppe der zugehörigen Liste über die Gruppen der Funktionen im Python-Dictionary zuordnen. Der Programmcode für diese Einteilung wird in Auszügen in Algorithmus 4.1 mit den Funktionsgruppen der Skalierung und Aggregation dargestellt.

Algorithmus 4.1: Auszug der Gruppierung der Datenvorverarbeitungsmethoden und Einteilung der Anwendung vor oder nach dem Train-Test-Split

```
1 function_groups = {
2     "Skalierung": [dt_MinMaxSkalierung, dt_Normalisierung,
          dt_RobustScale, dt_ZScore],
3     "Aggregation": [dt_Summe, dt_Mittelwert], ...}
4 function_groups_pre_split = [
5     *function_groups["Aggregation"], ...]
6 function_groups_post_split = [
7     *function_groups["Skalierung"], ...]
```

Algorithmus 4.1 zeigt die Verwendung des Dictionarys, welches Listen mit Funktionsnamen einer

definierten Gruppe zuordnet. In diesem Auszug werden zwei Gruppen dargestellt, die im Anschluss in die Listen überführt werden, indem die komplette Gruppe in der entsprechenden Gruppe abgerufen wird.

Die letzte Gruppierung der Funktionen ist die Einteilung der Datenvorverarbeitungsmethoden für den SVP unter Berücksichtigung der Randbedingungen, die in Kapitel 3.2.3 gefordert werden. Um diese Anforderungen zu erfüllen, werden alle Datenvorverarbeitungsmethoden in Listen eingetragen, die als Faktoren für die Erstellung des SVP benötigt werden. In einer Liste stehen die Funktionsnamen der Datenvorverarbeitungsmethoden, von denen nur eine aus jeder Gruppe pro Versuchslauf angewendet werden darf. Zusätzlich wird eine Liste mit Datenvorverarbeitungsmethoden angelegt, die keiner Randbedingung unterliegen. In dieser Implementierung resultieren daraus die Listen *fehlendeWerte, skalierung, aggregation, datenglättung* und *other_factors*. Damit ist die Einteilung der Datenvorverarbeitungsmethoden in diverse Funktionsgruppen abgeschlossen.

Da die Anwendung der Datenvorverarbeitungsmethoden zur Behandlung von fehlenden Werten und zur Datentransformation den zentralen Schritt im Experimentierplan und im Ablaufplan aus Abbildung 4.2 darstellt, werden die Funktionen, die die Methoden auf den Datensatz anwenden, im Folgenden näher erläutert. Die Funktionen werden in einem separaten Modul definiert und zu Beginn des Programms importiert. Der vollständige Python-Code des Moduls ist in Anhang B.2 zu finden. Im nachfolgenden Text wird der generelle Aufbau der Funktionen und ausgewählte Beispiele näher erläutert. Alle Funktionen sind nach der gleichen Konvention benannt worden. Diese beginnt mit dem Schritt der Datenvorverarbeitung, *dt* für Datentransformation und *fw* für Behandlung von fehlenden Werten. Gefolgt wird diese Einordnung von einem Unterstrich und endet mit dem Namen der Methode, die in der Funktion auf den Datensatz angewendet wird. Die Ein- und Ausgaben der Funktionen unterscheiden sich, sind aber innerhalb der Funktionsgruppen, deren Einteilung in Algorithmus 4.1 vorgestellt worden ist, gleich. Auch die Einordnung in unterschiedliche Anwendungszeitpunkte der Funktionen vor oder nach dem Train-Test-Split ist dort gezeigt worden. In der folgenden Tabelle 4.1 werden die Ein- und Ausgaben der Funktionsgruppen sowie der Anwendungszeitpunkt im Versuchsdurchlauf in Relation zum Train-Test-Split aufgelistet.

Tabelle 4.1: Eigenschaften der Funktionsgruppen von Datenvorverarbeitungsmethoden

Funktionsgruppen	Funktionseingabe	Funktionsausgabe	Anwendung in Relation zum Split
FehlendeWerte	Eingabedaten und Form der fehlenden Werte	Eingabedaten ohne fehlende Werte	davor
Imputation	Trainingsdaten, Testdaten und Form der Fehlenden Werte	Trainingsdaten und Testdaten ohne fehlende Werte	danach
Skalierung	Trainingsdaten und Testdaten	skalierte Trainings- und Testdaten	danach
Aggregation	Daten und Intervallanzahl	aggregierte Daten	davor
Datenglättung	Eingabedaten	geglättete Daten	davor
Merkmalskonstruktion	Zielgrößen und Sollwerte der Anlage	neu berechnete Zielgrößen	davor

Die Tabelle 4.1 zeigt die Eigenschaften aller Funktionsgruppen, in denen die Funktionen der Datenvorverarbeitungsmethoden eingeteilt sind. Mögliche Funktionseingaben sind die Eingabedaten vor dem Train-Test-Split oder die Trainings- und Testdaten, die mit dem Train-Test-Split erzeugt werden. Zusätzlich verlangen die Funktionen die Form der fehlenden Werte, die in dem verwendeten Datensatz 0 ist oder die Intervallanzahl bei Aggregationen. Auch die Zielvariablen können von der Merkmalskonstruktion mit den dazugehörigen Sollwerte der Anlage benötigt werden. Die Funktionen wenden die Datenvorverarbeitungsmethode auf und mit den Funktionseingaben an und geben die Daten in aufbereiteter Form zurück. Zur Veranschaulichung werden beispielhaft vier Funktionen, die Datenvorverarbeitungsmethoden anwenden, vorgestellt. Zwei müssen vor dem Train-Test-Split und zwei danach angewendet werden. Der komplette Python-Code für die Implementierung der Methoden ist in Anhang B.2 dargestellt. Für die Vorstellung im Text werden nur Ausschnitte benutzt.

Zuerst wird eine Funktion, die eine KNN-Imputation auf den Trainings- und Testdatensatz anwendet, vorgestellt. Die Grundlagen und die Vorgehensweise bei einer KNN-Imputation werden in Kapitel 2.2.2 behandelt. Die Funktionsdefinition ist ein Ausschnitt aus dem Modul Methoden.py, welcher in Algorithmus 4.2 dokumentiert wird.

Algorithmus 4.2: Code der Funktion fw_KNNImputation

```
1 def fw_KNNImputation(data_local_train, data_local_test,
      fehlende_werte= np.nan):
2       imputer = KNNImputer(n_neighbors=5, missing_values=
           fehlende_werte)
3       imputed_data_train = imputer.fit_transform(data_local_train)
4       imputed_data_test = imputer.transform(data_local_test)
5       return imputed_data_train, imputed_data_test
```

Der Algorithmus 4.2 zeigt die Definition der Funktion *fw_KNNImputation*, die Trainingsdaten, Testdaten und die Form der fehlenden Werte als Funktionseingabe aufnimmt. Die Form der fehlenden Werte wird in der nächsten Zeile bei der Initialisierung des *KNNImputer* aus der Scikit-learn-Bibliothek benötigt. Da dieser Importer die Möglichkeit bietet, die Form der fehlenden Werte zu wählen, wird die Variable an dieser Stelle weitergegeben. Einstellungsmöglichkeiten des Imputers werden nicht verändert und behalten den Standard der Bibliothek. Die Standard-Anzahl der Nachbarn (K) ist fünf. In den nächsten beiden Zeilen wird der initialisierte Imputer auf den Trainings- und Testdatensatz angewendet, um fehlende Werte mit Schätzungen des KNN-Algorithmus zu füllen. Auf den Trainingsdatensatz wird mit der *fit_transform()*-Funktion des Imputers erst eine Anpassung des Modells vorgenommen und anschließend der Trainingsdatensatz transformiert, indem fehlende Werte imputiert werden. Mit der Anpassung wird das Imputationsmodell auf dem Trainingsdatensatz trainiert, sodass die Eigenschaften des Modells auf diesen Datensatz zurückzuführen sind. Auf den Testdatensatz wird nur die *transform()*-Funktion angewendet, bei der das schon trainierte Modell für die Imputation der fehlenden Werte sorgt. Dadurch wird verhindert, dass Eigenschaften aus dem Testdatensatz in den Data-Mining-Algorithmus mit einfließen und die Vorhersagen dadurch verzerrt werden (vgl. Kapitel 2.2.5). Abschließend gibt Algorithmus 4.2 die imputierten Trainings- und Testdatensätze in gleicher Reihenfolge zurück, wie sie in die Funktion eingegeben wurden.

Eine weitere Funktion zur Behandlung fehlender Werte trägt dem Namen *fw_Interpolation* und wendet eine lineare Interpolation auf die Eingabedaten an. Der Python-Code der Funktion ist in Anhang B.2 zu finden. In dieser Implementierung werden alle Eingabedaten vor dem Train-Test-Split mitgegeben, damit die Datenpunkte für die Interpolation in der korrekten Reihenfolge erhalten bleiben. Dies ist ein Unterschied zu anderen Imputationsmethoden, die nach dem Train-Test-Split angewendet werden. Die Funktion sorgt zuerst dafür, dass alle fehlenden Werte in die Form *Not a Number* überführt werden, falls sie diese Form noch nicht haben. Mit dieser Form kann die interpolate-Funktion aus der Pandas-Bibliothek benutzt werden, die das lineare Interpolieren zwischen zwei vorhandenen Datenpunkten übernimmt. Die interpolierten Daten ohne fehlende Werte werden abschließend von der Funktion zurückgegeben.

In dieser Implementierung wird eine Merkmalskonstruktion als Datentransformation angewandt, die speziell auf den Datensatz zugeschnitten ist. Es wird die Differenz zwischen den gemessenen Werten und den Sollwerten, die die Prozessleitebene vorgibt, gebildet und als neue Zielvariable zurückgegeben. Der nachfolgende Algorithmus 4.3 zeigt die Umsetzung für die Differenzbildung in Form einer Python-Funktion.

Algorithmus 4.3: Code der Funktion dt_Differenz_bilden

```
1 def dt_Differenz_bilden(actual_local,setpoints_local):
2     #Bei beiden Df Indizes und Spaltennamen angleichen
3     actual_local.columns = [f"Abweichung␣{i+1}" for i in range(
          len(actual_local.columns))]
4     setpoints_local.columns = [f"Abweichung␣{i+1}" for i in
          range(len(setpoints_local.columns))]
5     #Dataframe mit Differenzen berrechnen. Nur mit gleichen
          Spaltennamen moeglich
6     differenz = actual_local - setpoints_local
7     return differenz
```

Der Algorithmus 4.3 zeigt die Funktion, die eine Merkmalskonstruktion auf die Zielvariablen anwendet (vgl. Kapitel 2.2.4). Für die Ausführung der Funktion ist die Mitgabe der gemessenen Zielvariablen und der Sollwerte aus der Prozessleitebene als Dataframes erforderlich. Im Zuge der Vorbereitung erfolgt eine Gleichbenennung der Spalten beider Dataframes, um eine Subtraktion der beiden Datenbestände zu ermöglichen. Um eine erneute Umbenennung der Spalten zu vermeiden, werden diese direkt als *Abweichung X* bezeichnet, wobei das *X* für die X-te Spalte im Dataframe steht. Damit ist es auch möglich, die Sollwerte von den gemessenen Werten abzuziehen und ein Dataframe, bestehend aus den Differenzen, zu erhalten. Bei der Subtraktion der Dataframes werden korrespondierende Einträge in der gleichen Zeile und Spalte voneinander abgezogen. Voraussetzung dafür ist, dass die Dataframes die gleiche Größe haben und die gleiche Reihenfolge der Mess- und Sollwerte aufweisen. Da der Datensatz geordnet ist, ist dies durch die Aufteilung des Datensatzes in der Datenselektion bereits vorgegeben (vgl. Abbildung 4.2). Abschließend gibt die Funktion die Differenz als Dataframe mit den neuen Spaltenüberschriften zurück.

Die Datentransformationsmethode der Min-Max-Normalisierung wird ebenso in einer Funktion die-

ser Implementierung angewendet. Die Funktion zählt zur Funktionsgruppe der Skalierungen und wird nach dem Train-Test-Split angewendet, wie die Imputationen. Zusätzlich ist der Aufbau fast identisch zu der zuvor in Algorithmus 4.2 vorgestellten Funktion, die eine KNN-Imputation anwendet. Es werden Trainings- und Testdaten eingegeben und skalierte Daten zurückgegeben. Anstatt eines Imputers wird ein Skalierer der Scikit-Learn-Bibliothek initialisiert. Dies ist der Min-Max-Skalierer, der mit den benutzten Standardeinstellungen auf den Wertebereich von 0 bis 1 skaliert und somit eine Min-Max-Normalisierung durchführt. Auch bei der Skalierung ist es wichtig, die Trainingsdaten mit *fit_transform()* und die Testdaten mit *transform()* zu transformieren, um ein Datenleck zu vermeiden (vgl. Kapitel 2.2.5). Alle anderen Funktionen der Skalierungs-Funktionsgruppe sind auch so aufgebaut und unterscheiden sich nur im initialisierten Skalierer der Scikit-Learn-Bibliothek (vgl. Anhang B.2).

Nachdem die Funktionen, die die Datenvorverarbeitungsmethoden anwenden, vorgestellt worden sind, wird der Ablaufplan des Experiments weiter verfolgt. Wie in Abbildung 4.2 zu sehen ist, werden die gebildeten Funktionsgruppen bei der Erstellung des SVP herangezogen. Die Funktion, die den SVP erstellt, wird zu Beginn importiert und ist auf diese Implementierung zugeschnitten. Sie erhält die fünf Listen *fehlendeWerte, skalierung, aggregation, datenglättung* und *other_factors* als Eingabe und erstellt mit allen Datenvorverarbeitungsmethoden aus diesen Listen einen vorläufigen vollfaktoriellen Versuchsplan. Im Anschluss wird dieser Versuchsplan gekürzt. Dabei wird jede Zeile gelöscht, in der mehr als eine Datenvorverarbeitungsmethode einer Gruppe, die einer Randbedingung unterliegt, angewendet werden soll. Der gekürzte SVP wird von der Funktion zurückgegeben und zur Dokumentation als CSV-Datei abgespeichert. Der Programmcode der Funktion wird in Anhang B.3 dargestellt.

Bevor die Versuchsläufe der SVP durchgeführt werden können, muss eine Funktion definiert werden, die die Data-Mining-Ergebnisse bewertet, wie in Kapitel 2.2.5 beschrieben und im Experimentierplan gefordert wird. Diese Funktion bearbeitet die Data-Mining-Aufgabe mit dem festgelegten Data-Mining-Algorithmus und gibt Vergleichsmetriken zurück. In dieser Implementierung des Experimentierplans wird die zweite gestellte Aufgabe, die in Kapitel 4.1 vorgestellt wird, gelöst. Der Pseudocode der besagten Funktion zur Bewertung und folglich zum Vergleich wird mit dem Algorithmus 4.4 dokumentiert.

Algorithmus 4.4: Bewertungsfunktion für die Data-Mining-Aufgabe in Pseudocode

```
1 def data_mining_regression(Trainingsdaten, Zielvariablen,
      Testdaten, Test-Zielvariablen, Regressionsmodell):
2       r2mean, r2sum = 0
3       r2median_liste = []
4       Schleife ueber jede Spalte in den Zielvariablen:
5           Zeilen mit gueltigen (Test-)Zielvariablen != 0
              herausfiltern
6           Auch Trainings- und Testdaten mit diesen Zeilen filtern
7           Ueberspringe die Spalte, wenn keine gueltigen (Test-)
              Zielvariablen vorhanden sind
8
```

```
9           Regressionsmodell lernt mit gueltigen Trainingsdaten und
                Zielvariablen der aktuellen Spalte
10          Test-Vorhersage = Regressionsmodell(Testdaten)
11
12          Vergleichsmetriken = Berechnung(Test-Vorhersage, Test-
               Zielvariable)
13          r2sum += r2
14          r2median_liste.anhaengen(r2)
15      r2mean = r2sum/Schleifendurchlaeufe
16      r2median_value = median(r2median_liste)
17      return r2mean, r2median_value, r2sum
```

Algorithmus 4.4 zeigt in Pseudocode eine Funktion mit dem Namen *data_mining_regression*, die Eingangsdaten benutzt, um Vergleichsmetriken zu berechnen. Die Daten, die in die Funktion eingegeben werden, sind Trainingsdaten und Testdaten mit deren zugehörigen Zielvariablen. Zusätzlich benötigt die Funktion ein zuvor initialisiertes Regressionsmodell, welches als Eingabe mitgegeben wird. Zu Beginn der Funktion werden die Vergleichsmetriken initialisiert. In dieser Implementierung sind das Formen der R^2-Werte, die in Kapitel 2.2.5 und Kapitel 2.1.2 erläutert werden. Da die Data-Mining-Aufgabe eine Vorhersage von 15 verschiedenen Messwerten vorsieht, werden die Mittelwerte und der Median des R^2-Wertes ermittelt, um die Regression für die gesamte Data-Mining-Aufgabe zu bewerten. Die Summen der Werte und eine Liste für den Median werden dafür als Hilfsvariablen gebraucht und in dem Zuge auch initialisiert. Den Hauptbestandteil der Funktion stellt die Schleife dar, die über alle Spalten der Zielvariablen, also über die 15 verschiedenen Messwerte, iteriert. In jeder Iteration sind die Zielvariablen somit eine Spalte aus der Datenmatrix der eingegebenen Zielvariablen und stellen einen Messwert dar. In der weiteren Beschreibung der Funktion aus Algorithmus 4.4 ist mit den Zielvariablen die aktuelle Spalte in der Schleife gemeint.

Die Zielvariablen beinhalten noch fehlende Werte, wie in Kapitel 4.1 vorgestellt wurde. Diese werden gelöscht, um das Regressionsmodell nur auf gültigen Werten zu trainieren und zu bewerten. Dafür werden diese am Anfang jedes Schleifendurchgangs für die aktuelle Spalte herausgefiltert. Der Filter sucht nach fehlenden Werten in den Zielvariablen und Test-Zielvariablen und entfernt diese und die korrespondierenden Zeilen der Trainings- und Testdaten. Die fehlenden Werte in den Zielvariablen werden nicht imputiert, um Verzerrungen durch die Imputation in den Daten vorzubeugen. Zur Sicherheit wird abgefragt, ob nach dem Filter in Zielvariablen noch Werte vorhanden sind, ansonsten wird die Spalte übersprungen. Die Abfrage ist notwendig, weil eine Spalte 91% fehlende Werte aufweist und es theoretisch möglich wäre, dass nach dem Train-Test-Split keine gültigen Werte in (Test-)Zielvariablen übrig bleiben.

Nachdem fehlende Werte in den Zielvariablen herausgefiltert sind, wird das Regressionsmodell auf den Trainingsdaten und Zielvariablen trainiert. Mit dem trainierten Modell wird eine Vorhersage für die Testdaten getroffen. Im Anschluss kann diese Vorhersage mit den richtigen Zielvariablen der Testdaten verglichen und bewertet werden. Im Programmcode wird der R^2-Wert mit dafür vorgefertigten Funktionen berechnet, die die Scikit-Learn-Bibliothek zur Verfügung stellt. In jedem Schleifendurchgang wird diese Vergleichsmetrik aufaddiert und an eine Liste angehangen, sodass nach der Schleife ein Durchschnitt und ein Median der Werte berechnet werden können.

Diese Vergleichsmetriken werden von der Funktion aus Algorithmus 4.4 zurückgegeben und später als Zielgrößen in den SVP eingetragen, um die Methoden vergleichen zu können. Auch wenn nicht alle Vergleichsmetriken für den Vergleich der Datenvorverarbeitungsmethoden gebraucht werden, werden alle zurückgegeben, um dem Benutzer keine Informationen vorzuhalten. Um die Vergleichsmetriken in den SVP eintragen zu können, werden leere Spalten mit den Bezeichnungen der Vergleichsmetriken an das Dataframe des SVP angehangen, die in den Versuchsdurchläufen gefüllt werden können. Dies ist die letzte Vorbereitung auf die SVP, wie in Abbildung 4.2 zu erkennen ist.

Die Schleife, die den SVP durchläuft, wird nach der Erweiterung des SVP-Dataframes um die Spalten für die Vergleichsmetriken initialisiert. Diese bildet den zentralen Anteil des Experimentierplans, da in der Schleife die Versuchsläufe durchgeführt werden. Im Rahmen der Iteration durch die Zeilen des SVP übernimmt jede Zeile im aktuellen Schleifendurchlauf die Rolle der *aktuellen Zeile*. Diese Begrifflichkeit wird in der anschließenden Erläuterung der Schleife verwendet. Zur Veranschaulichung der Schritte innerhalb der Schleife wird auf Abbildung 4.2 verwiesen.

Der erste Schritt der Schleife besteht aus der Überprüfung, welche Datenvorverarbeitungsmethoden im aktuellen Versuchslauf angewendet werden sollen. Dafür wird der Wert jeder Spalte in der aktuellen Zeile überprüft, ob er den Wert 1 besitzt. Damit wird verdeutlicht, dass dieser Faktor im SVP positiv ausgeprägt ist und somit die Datenvorverarbeitungsmethode angewendet werden soll. Falls die Überprüfung positiv ausfällt, wird die Funktion, deren Namen dem Spaltennamen entspricht, in einer von zwei Listen gespeichert. Die zwei Listen unterscheiden die Funktionen in einer Anwendung der Datenvorverarbeitungsmethode vor oder nach dem Train-Test-Split. Für die Unterscheidung werden die Funktionsgruppen benutzt, deren Einteilung in Algorithmus 4.1 vorgestellt wurde. Damit hat die aktuelle Zeile des SVP zwei Listen, in denen die Funktionen stehen, die im Versuchslauf angewendet werden.

Im Anschluss wird der Ausgangsdatensatz aufgerufen, der am Anfang der Implementierung als CSV-Datei gespeichert wurde. Die dafür benötigten Dateien werden mit Funktionen aus der Pandas-Bibliothek von CSV-Dateien in DataFrames umgewandelt. Auf und mit den DataFrames werden die Datenvorverarbeitungsmethoden angewendet. Zuerst werden alle Methoden angewendet, die in der Liste für die Anwendung vor dem Train-Test-Split stehen. Dafür wird jede Funktion in der Liste auf ihre Funktionsgruppe überprüft und entsprechend mit den passenden Funktionseingaben und Funktionsausgaben ausgeführt. Die Reihenfolge der Funktionsausführung ist demnach mit der Reihenfolge der Datenvorverarbeitungsmethoden im SVP gleich. Eine Datenvorverarbeitungsmethode, die als erster Faktor im SVP aufgelistet wird, wird demnach eher angewendet als eine Methode, die als zweiter Faktor im SVP steht. Vorausgesetzt, beide Methoden werden vor dem Train-Test-Split angewendet. Diese Einteilung ist aufgrund der Reihenfolge in der Implementierung übergeordnet.

Nach der Anwendung aller Methoden, die vor dem Train-Test-Split vorgesehen sind, wird dieser durchgeführt. Dafür wird eine Funktion der Scikit-Learn-Bibliothek genutzt. Diese benötigt die Eingabedaten und Zielvariablen und teilt sie jeweils in Trainings- und Testdaten auf. Zusätzlich wird der Funktion mitgegeben, wie groß der Anteil der Testdaten im Verhältnis zur Gesamtgröße des Datensatzes ist. In dieser Implementierung werden den Testdaten ein Viertel des gesamten Datensatzes zugeschrieben, so wie es auch in Kapitel 2.1.4 beschrieben wurde. Die anderen drei Viertel werden den Trainingsdaten zugewiesen. Die Funktion mischt die Datenpunkte vor der Zuweisung

zu Trainings- und Testdaten, sodass diese zufällig erfolgt. Nach der abgeschlossenen Einteilung werden weitere Datenvorverarbeitungsmethoden angewendet, sollten deren Funktionen in der aktuellen Zeile in der vorgesehenen Liste dafür eingetragen sein. Diese Anwendung erfolgt im gleichen Schema, wie die der zuvor angewendeten Datenvorverarbeitungsmethoden, mit dem einzigen Unterschied, dass die Trainings- und Testdatensätze in den Funktionen eingegeben werden. Damit ist die Datenvorverarbeitung abgeschlossen und in der Implementierung kann mit der Bewertung der Datenvorverarbeitung begonnen werden, sodass ein Vergleich der angewendeten Datenvorverarbeitungsmethoden möglich ist.

Die Bewertung passiert in dieser Implementierung mit der Hilfe einer Regression, die in der Funktion angewendet wird, die dafür nach dem Schema von Algorithmus 4.4 definiert wurde und in Anhang B.1 in Python-Code dokumentiert ist. Die Bewertungsfunktion wird für die aktuelle Zeile aufgerufen. In die Funktion werden die Trainingsdaten, die Zielvariablen, die Testdaten, die Test-Zielvariablen und das initialisierte Regressionsmodell eingegeben. Die Ergebnisse der festgelegten Vergleichsmetriken sind die Ausgabe der Funktion. Diese werden in Variablen gespeichert, um sie im nächsten Schritt in den Ergebnisspalten der aktuellen Zeile des SVP einzutragen. Dies ist der letzte Schritt der Schleife. Anschließend springt die Implementierung entweder in die nächste Zeile im SVP oder verlässt die Schleife, sobald der letzte Versuchslauf durchlaufen wurde. Nach der Schleife sind die Ergebnisspalten im SVP vollständig gefüllt, sodass es sinnvoll ist, auch den SVP mit den Ergebnissen als CSV-Datei zu speichern und die Ergebnisse auf diese Weise zu dokumentieren.

Mit den Ergebnissen der Vergleichsmetriken in der CSV-Datei ist es anschließend möglich, die Auswertung vorzunehmen. In dieser Implementierung wird diese in einem separaten Jupyter-Notebook durchgeführt. Die Auslagerung der Auswertung der zwischengespeicherten Ergebnisse hat den Vorteil, dass das Experiment nicht noch einmal durchgeführt werden muss, um Anpassungen bei der Auswertung vorzunehmen. Für die Auswertung ist eine Funktion implementiert worden, die die Effekte nach dem Vorgehen aus Kapitel 2.1.2 und Tabelle 3.7 berechnet. Die Funktion wird in folgendem Algorithmus 4.5 in Python-Code dargestellt.

Algorithmus 4.5: Python-Funktion für Effektberechnung mit Funktionsaufruf

```
1  svp_results = pd.read_csv("Ergebnisse/SVP_mit_Ergebnissen.csv")
2
3  def berechne_effekte(df, zielspalte="r2_mean"):
4      ergebnisse = {}
5      for spalte in df.columns:
6          if spalte == zielspalte:
7              continue
8          # Mittelwert fuer Zielspalte, wenn Spalte == 1
9          mean_1 = df[df[spalte] == 1][zielspalte].mean()
10         # Mittelwert fuer Zielspalte, wenn Spalte == 0
11         mean_0 = df[df[spalte] == 0][zielspalte].mean()
12         ergebnisse[spalte] = {"Mittelwert+": mean_1, "Mittelwert
               -": mean_0, "Effekt": mean_1 - mean_0}
13     return ergebnisse
```

```
14
15  # Beispielaufruf:
16  effekte_r2_mean = berechne_effekte(svp_results, zielspalte="
        r2_mean")
17  print(effekte_r2_mean)
18  effekte_r2_mean_df = pd.DataFrame(effekte_r2_mean).T
```

Algorithmus 4.5 zeigt die Definition der Funktion, die für die Effektberechnung benutzt wird. In dieser Funktion wird ein DataFrame und der Name der Zielspalte im DataFrame als String eingegeben. Die Standardeinstellung gibt die Spalte mit den durchschnittlichen R^2-Werten als Zielspalte mit. Zu Beginn wird ein Dictionary für die Ergebnisse initialisiert, welches in der Funktion gefüllt wird und am Ende zurückgegeben wird. Eine Schleife iteriert durch die Spalten im eingegebenen DataFrame und überspringt die Zielspalte selbst. In jeder anderen Spalte wird gefiltert, in welchen Feldern eine 1 steht. Es werden die Ergebnisse, die in den gleichen Zeilen von diesen Feldern in der Zielspalte stehen, gemittelt und in einer Variable gespeichert. Dieses Vorgehen wird analog zu den Feldern in der Spalte, in denen eine 0 eingetragen ist, wiederholt. Die Variablen der Mittelwerte werden in dem Ergebnis-Dictionary in Verbindung mit dem Spaltennamen und der Bezeichnung Mittelwert+ oder Mittelwert- gespeichert. Zusätzlich wird der Effekt als Differenz zwischen den beiden Mittelwerten gespeichert. Nach der Schleife wird das gefüllte Ergebnis-Dictionary von der Funktion zurückgegeben.

Außerdem wird in Algorithmus 4.5 gezeigt, wie vorher die Ergebnisse aus einer CSV-Datei in ein Pandas DataFrame umgewandelt werden. Im Anschluss an die Funktionsdefinition wird die Funktion mit diesem DataFrame und den Mittelwerten der R^2-Werte als Zielspalte aufgerufen. Das Ergebnis-Dictionary wird umgewandelt und die berechneten Effekte werden in einem separaten DataFrame gespeichert. Die Funktion wird auch für die Berechnung von Wechselwirkungen (vgl. Kapitel 2.1.2) benutzt, nachdem Spalten mit den Faktorstufen 0 und 1 entsprechend der Wechselwirkung zweier Faktoren hinzugefügt wurden. Für die Erzeugung von Diagrammen zum Vergleich der Effekte ist die Matplotlib-Bibliothek von Hunter (2007) in Python benutzt worden. Der Code für die Erstellung der Diagramme und der Vorbereitung dafür ist in Anhang B.4 zu finden.

Nachdem in diesem Abschnitt die Implementierung des Experimentierplans vorgestellt wurde, wird im nächsten Abschnitt dessen Anwendung auf den zuvor vorgestellten Datensatz durchgeführt. Dabei werden die im Verlauf des Experimentierplans getroffenen Entscheidungen erläutert, die für die Datenvorverarbeitungsmethoden, den SVP und die Vergleichsmetriken verantwortlich sind.

4.3 Anwendung des Experimentierplans auf den Produktionsdatensatz

In diesem Abschnitt wird die Anwendung des Experimentierplans auf den Datensatz aus Abschnitt 4.1 vorgestellt. Dabei wird auf den Experimentierplan aus Kapitel 3 sowie Anhang A zurückgegriffen. Für die Umsetzung wird die Implementierung, die in Kapitel 4.2 sowie Anhang B vorgestellt wurde, zurückgegriffen. Die Entscheidungen im Experimentierplan werden in diesem Kapitel zuerst

hergeleitet und anschließend in Stichpunkten zusammengefasst und hervorgehoben.

Zu Beginn wird im Experimentierplan der Datensatz geladen und es folgt eine Datenselektion mit einer darauffolgenden Festlegung der Data-Mining-Aufgabe. Für diesen Datensatz sind, wie in Kapitel 4.1 vorgestellt, diese beiden Schritte schon vorgegeben. Die Data-Mining-Aufgabe ist die Vorhersage von 15 Messwerten nach Maschine 5 mit den Werten, die davor in der Produktionslinie erfasst werden. Da die Variablen vom Anfang der Produktionslinie in den 15 Messpunkten nach dem Kombinierer zusammengefasst werden, werden alle Variablen in der Produktionslinie vor diesen Messpunkten nicht weiter berücksichtigt. Eingangsdaten für die Vorhersage sind demnach die Umgebungsvariablen, die ersten 15 Messpunkte und die Prozessparameter der Maschinen 4 und 5. Der Data-Mining-Algorithmus, der für diese Aufgabe festgelegt wird, ist die KNN-Regression. Neben den Informationen zum Datensatz ist auf Kaggle von anderen Nutzern der Seite ein Vergleich von verschiedenen Regressionsalgorithmen durchgeführt worden, bei dem die KNN-Regression die 15 Messwerte im Durchschnitt am besten vorhergesagt hat. Dieses Vorwissen ist an dieser Stelle in die Entscheidung für den Data-Mining-Algorithmus mit eingeflossen. Für die Bewertung der KNN-Regression ist der R^2-Wert zur Bewertung für Regressionen vorgesehen (vgl. Kapitel 2.2.5). Die Vergleichsmetriken RMSE und MAE sind auch in Betracht gezogen worden, wurden vor der Auswertung aber verworfen, weil die Fehlergrößen durch verschiedene Datentransformationen nicht einheitlich groß waren. Anschließend ist der MAPE aufgrund von den uneinheitlichen Größen in Betracht gezogen worden. Dieser liefert jedoch auch keine aussagekräftigen Ergebnisse, weil einige Werte nach einer Datentransformation nahe an der Null sind. Dieses Problem wurde in Kapitel 2.1.2 herausgestellt.

- **Data-Mining-Aufgabe:** Messungen nach Maschine 5 mit den Umgebungsvariablen, den Prozessparametern von Maschine 4 und 5 und den 15 Messungen vor Maschine 4 vorhersagen
- **Data-Mining-Algorithmus:** KNN-Regression zur Vorhersage von metrisch kontinuierlichen Werten
- **Vergleichsmetrik:** R^2-Wert zur Bewertung der Regression

Gemäß dem Experimentierplan werden die Voraussetzungen für den KNN-Algorithmus dokumentiert. Für die Anwendung der KNN-Regression werden metrisch kontinuierliche Daten benötigt (vgl. Kapitel 2.1.1). Zudem kann die Prognose durch heterogene Skalen in den Merkmalen der Eingabedaten verzerrt werden, weshalb eine einheitliche Skalierung der Merkmale erforderlich ist. Für den KNN-Algorithmus ist die Verwendung einer vollständigen Datenmatrix ebenso essenziell, da dieser für die Berechnung der Abstände zwischen den Datenpunkten Werte benötigt (vgl. Kapitel 2.2.2).

- **Voraussetzung für KNN-Regression:**
  - metrische kontinuierliche Daten für Abstandsmessung
  - Einheitliche Skalierung der Daten
  - Vollständige Datenmatrix

Nach den Voraussetzungen des Data-Mining-Algorithmus müssen auch die Eigenschaften des Da-

tensatzes dokumentiert werden, um einen späteren Abgleich von Voraussetzungen und Eigenschaften zu ermöglichen, wie es der Experimentierplan vorsieht. Wie schon bei der Vorstellung des Datensatzes in Kapitel 4.1 herausgestellt wurde, sind alle Merkmale metrisch kontinuierlich ausgeprägt (vgl. Kapitel 2.1.1). Ebenfalls wurde vorgestellt, dass die Merkmale des Datensatzes auf unterschiedlichen Werteskalen ausgeprägt sind. Des Weiteren sind die natürlich auftretenden fehlenden Werte erwähnt worden. In den Eingabedaten sind in den ersten 15 Messwerten fehlende Werte vorhanden. Acht Merkmale davon haben unter 6% fehlende Werte und 7 Merkmale weisen mehr fehlende Werte, zwischen 23 und 95%, auf. In den Prozessparametern von Maschine 4 und 5 sind alle Werte vorhanden. Eine Übersicht über die fehlenden Werte in den Eingabedaten ist in Anhang C.1 gegeben. Die Vorstellung des Datensatzes erklärt auch, dass jeder Datenpunkt einen Zeitstempel hat, an dem dieser aufgenommen wurde. Deshalb ist der Datensatz mit Zeitreihendaten gefüllt. Eine Besonderheit des Datensatzes ist, dass für die Zielvariablen gemessene Werte und Sollwerte vorhanden sind. Der Datensatz besteht zudem aus einer Mischung aus kontrollierbaren und unkontrollierbaren Daten. Die 15 Messwerte sind unkontrollierbar bzw. nicht direkt über die SPS steuerbar. Die Prozessparameter von Maschine 4 und 5 sind direkt über die SPS kontrollierbar (vgl. Kapitel 2.3.2).

- **Eigenschaften des Datensatzes:**
  - Alle Merkmale metrisch kontinuierlich ausgeprägt
  - Merkmale mit unterschiedlichen Werteskalen
  - Fehlende Werte in den ersten 15 Messwerten zwischen 0,5 und 95%
  - Zeitreihendaten
  - Gemessene Werte und Sollwerte vorhanden
  - unkontrollierbare oder nicht direkt kontrollierbare Daten: 15 Messwerte
  - kontrollierbare Daten: Prozessparameter von Maschine 4 und 5 und Umgebungsvariablen

Als nächstes wird im Experimentierplan geprüft, ob es fehlende Werte gibt. Da dies im vorherigen Abschnitt bestätigt wurde, können im Anschluss die Eigenschaften des Datensatzes mit den Voraussetzungen des Data-Mining-Algorithmus abgeglichen werden. Der Abgleich ergibt, dass die metrisch kontinuierliche Datenausprägung des Datensatzes nicht angepasst werden muss. Aufgrund der geforderten einheitlichen Skalierung der Daten und der unterschiedlichen Werteskalen im Datensatz muss die Skalierung als Datentransformationsaufgabe auf den Datensatz angewendet werden (vgl. Kapitel 2.2.3). Zudem wird eine vollständige Datenmatrix vorausgesetzt, die noch nicht gegeben ist und durch die Behandlung von fehlenden Werten hergestellt werden muss (vgl. Kapitel 2.2.1). Außer den Datenvorverarbeitungsschritten der Datentransformation und Behandlung von fehlenden Werten, die im SVP verglichen werden, müssen keine weiteren Datenvorverarbeitungsschritte vor dem SVP angewendet werden, um einen ausreichenden Ausgangsdatensatz zu erhalten. Es wurde überlegt, die Anzahl der unkontrollierbaren Daten durch eine Datenreduktion zu minimieren, um das

Regressionsmodell besser in die Praxis umsetzen zu können. Da die fehlenden Werte jedoch in den unkontrollierbaren Daten dieses Datensatzes liegen, wurde auf eine Datenreduktion verzichtet, um die Auswirkungen von Methoden zur Behandlung fehlender Werte möglichst groß zu halten und somit aussagekräftige Ergebnisse zu erhalten. Der Ausgangsdatensatz wurde anschließend, wie in der Implementierung gezeigt, gespeichert.

Der Experimentierplan fordert, dass die Methoden zur Behandlung fehlender Werte und zur Datentransformation ausgesucht werden. Dafür wird eine Vorauswahl in Tabelle 3.5 und Tabelle 3.6 bereitgestellt. Es ist auch möglich andere Methoden zu wählen. Für diesen Datensatz werden die Methoden zur Behandlung von fehlenden Werten nachfolgend aufgeführt (vgl. Kapitel 2.2.2). Für die Behandlungsart, bei der fehlende Werte gelöscht werden, werden Merkmale mit mehr als 20% fehlender Werte gelöscht. Die Methode, dass Datentupel mit fehlenden Werten gelöscht werden, wird nicht angewandt, da es zu wenig vollständige Datentupel im Datensatz gibt. Von der Imputation von Lageparametern werden die Methoden Mittelwertimputation und Medianimputation ausgewählt. Die Imputation des Modalwerts ergibt bei metrisch kontinuierlich ausgeprägten Merkmalen wenig Sinn (vgl. Kapitel 2.2.2). Außerdem werden die lineare Interpolation, die KNN-Imputation, die MICE, die MissForest-Imputation und die Regressionsimputation als modellgestützte Imputationen zum Vergleich ausgewählt. Die lineare Interpolation ist bei der Auswahl besonders hervorzuheben, weil der Datensatz mit Zeitreihendaten gefüllt ist, wie bei den Eigenschaften des Datensatzes herausgestellt wurde. Wie in Kapitel 2.2.2 herausgestellt wurde, ist die Interpolation dafür besonders geeignet. Insgesamt werden acht Methoden zur Behandlung fehlender Werte ausgewählt.

Bei dem Abgleich von Voraussetzungen der Data-Mining-Methode und den Eigenschaften des Datensatzes ist herausgestellt worden, dass eine Skalierung auf den Datensatz angewendet werden muss. Deshalb werden die Datentransformationsmethoden Min-Max-Normalisierung, Normalisierung, Z-Score-Standardisierung und eine Median-basierte Skalierung (RobustScaler) zum Vergleich ausgewählt. Da die Daten für die Data-Mining-Aufgabe als verrauscht beschrieben wurden, werden Datentransformationsmethoden aus den Datentransformationsaufgaben der Datenglättung und Aggregation ausgewählt Kapitel 2.2.4. Für die Aggregation sind die Durchschnitts- sowie die Summenbildung auf 2800 neue Datenpunkte ausgewählt worden. Für die Aufgabe der Datenglättung wird der Moving-Average und die exponentielle Glättung, beide mit der Fenstergröße 5, ausgewählt. Die Anzahl der Datenpunkte bei der Aggregation und die Fenstergröße bei der Datenglättung sind aufeinander abgestimmt, um die Vergleichbarkeit der Methoden auch außerhalb der eigenen Datentransformationsaufgabe zu gewährleisten. Zuletzt wird eine Merkmalskonstruktion ausgewählt, die mit den Messwerten und Sollwerten eine Differenz berechnet und dadurch die Abweichung als neues Merkmal bildet. Diese Abweichung wird dann anstelle der Messwerte von der Regression vorhergesagt. Insgesamt werden neun Datentransformationsmethoden ausgewählt.

- **Methoden zur Behandlung von fehlenden Werten:**
  - Löschen: Löschen von Merkmalen (>20% fehlende Werte)
  - Imputation Lageparameter: Mittelwert und Median
  - Modellgestützte Imputation: Lineare Interpolation, KNN-Imputation, MICE, MissForest

und Regressionsimputation

- **Datentransformationsmethoden:**
  - Skalierung: Min-Max-Normalisierung, Normalisierung, Z-Score-Standardisierung und Median-basierte Skalierung (RobustScaler)
  - Aggregation: Durchschnittsbildung und Summenbildung (2800 Datenpunkte)
  - Datenglättung: Moving-Average, exponentielle Glättung (Fenstergröße 5)
  - Merkmalskonstruktion: Differenzbildung zwischen Mess- und Sollwerten

Nachdem die Methoden ausgewählt wurden, fordert der Experimentierplan die Erstellung des SVP und stellt dafür eine Vorlage in Tabelle 3.7 bereit. Der SVP wird als vollfaktorieller Versuchsplan mit Randbedingungen als Einschränkungen erstellt, sodass immer nur eine Methode zur Behandlung von fehlenden Werten und eine Methode je Datentransformationsaufgabe angewendet wird (vgl. Kapitel 3.2.3). Insgesamt besteht der SVP aus 17 Faktoren und beinhaltet 810 Versuchsläufe. Der SVP ist, wie in der Implementierung beschrieben, mit einer Python-Funktion erstellt worden, die den SVP als Datenmatrix zurückgibt. Da dieser auch gespeichert wird, ist ein Ausschnitt des SVP in Anhang C.3 zu finden.

Der initiale Durchgang, den der Experimentierplan vorsieht, wird durchgeführt, indem die Bewertungsfunktion direkt auf den Ausgangsdatensatz angewendet wird. Zunächst werden keine weiteren Datenvorverarbeitungsmethoden angewendet, weil das bei diesem Datensatz nicht notwendig ist. Der Datensatz ist technisch gesehen schon vollständig, weil fehlende Werte als 0 dargestellt werden (vgl. Kapitel 4.1) und ohne Datentransformation ist die Vorhersage zwar verzerrt, aber mit dem Data-Mining-Algorithmus nicht unmöglich. Die vorhergesagten Messwerte für die Test-Variablen sind sinnvoll und weichen nur leicht von den tatsächlichen Test-Zielwerten ab. In den Randwerten ist die Vorhersage tendenziell extremer als die tatsächliche Messung. Der durchschnittliche R^2-Wert für die Bewertung der Regression zur Vorhersage der 15 Messwerte im initialen Durchgang beträgt 0,57 und der Median 0,60. Somit können beide Fragen in den Verzweigungen im Experimentierplan positiv beantwortet werden. Es werden sinnvolle Ergebnisse für die Data-Mining-Aufgabe geliefert und da eine perfekte Vorhersage einen R^2-Wert von 1 hat, ist eine Verbesserung der Ergebnisse auf jeden Fall möglich. Abschließend sieht der Experimentierplan die Anwendung des SVP vor, die mit der Implementierung umgesetzt wird.

- **Initialer Durchgang:**
  - Vorhersage der 15 Messwerte mit R^2-Wert bewertet: Durchschnitt 0.57 und Median 0.60
  - Es werden Messwerte vorhergesagt, die nah an den echten tatsächlichen Messwerten liegen
- **Versuchsläufe werden durchlaufen um die Datenvorverarbeitungsmethoden zu vergleichen**

Mit der Durchführung des SVP sind die Ergebnisspalten der SVP-Datenmatrix ausgefüllt worden. Zusätzlich ist, wie in Kapitel 4.2 gezeigt, die Datenmatrix mit den Ergebnissen für die Auswertung als CSV-Datei gespeichert worden. Ein Ausschnitt dieser Datenmatrix ist in Anhang C.4 zu finden. Im Folgenden wird die Datenmatrix mit den Ergebnissen als Ergebnismatrix bezeichnet. Der durchschnittliche R^2-Wert in der Ergebnismatrix ist von -0,01 bis 0,84 ausgeprägt, wobei 71,36% der Versuchsläufe ein besseres Ergebnis als der initiale Durchgang erzielt haben (R^2 = 0,57). Die Mediane der R^2-Werte sind von -0,01 bis 0,88 leicht besser ausgeprägt und von diesen sind 77,04% besser als der Median R^2-Wert des initialen Durchlaufs (R^2 = 0,6).

Nachdem der Experimentierplan aus Kapitel 3 fast vollständig auf den Datensatz mit der Implementierung aus Kapitel 4 angewendet wurde und die Ergebnisse des SVP dokumentiert wurden, erfolgt nun die Auswertung im nächsten Abschnitt. Dort wird die Anwendung des Experimentierplans mit der Auswertung der Ergebnisse abgeschlossen.

4.4 Auswertung der Ergebnisse

In diesem Abschnitt werden die Ergebnisse, die durch die Ausführung des SVP erzielt worden sind, vorgestellt und ausgewertet. Die relevante Zielgröße für die Auswertung ist der R^2-Wert für die Regressionsbewertung. Für die vorliegende Auswertung wurde der durchschnittliche R^2-Wert für die 15 Messwerte und nicht der Median herangezogen. Dies hatte den Grund, dass dadurch die Messwerte, die aufgrund unreiner Daten schwerer vorherzusagen waren, stärker in die Zielgröße einbezogen werden konnten. Es ist schließlich die Aufgabe der angewendeten Datenvorverarbeitungsmethoden, unreine Daten zu behandeln. Mit dieser Entscheidung werden aussagekräftigere Ergebnisse in der Auswertung erhofft. Zuerst werden die Haupteffekte der angewendeten Datenvorverarbeitungsmethoden vorgestellt, sodass ein Vergleich der Methoden untereinander möglich ist. Im Anschluss werden die Zweifach-Wechselwirkungen vorgestellt, bevor abschließend die besten Kombinationen für die untersuchte Data-Mining-Aufgabe herausgestellt werden. Dies ist möglich, weil ein vollfaktorieller Versuchsplan angewendet wurde.

Die Haupteffekte der 17 angewendeten Datenvorverarbeitungsmethoden sind nach dem Vorgehen aus Kapitel 2.1.2 berechnet worden. Für die Effektberechnung wird auf die Implementierung zurückgegriffen, die am Ende von Kapitel 4.2 in Form von Algorithmus 4.5 vorgestellt wird. Der komplette Code, der für die Effektberechnung und die weitere Auswertung herangezogen worden ist, ist in Anhang B.4 zu finden. Zunächst werden die Ergebnisse der Haupteffekte in Effekt-Diagrammen dargestellt (vgl. Kapitel 2.1.2). Die Zielgröße, die auf der y-Achse der Effekt-Diagramme zu finden ist, ist der durchschnittliche R^2-Wert für die Vorhersage der 15 Messwerte. Die Darstellung der Haupteffekte in den Diagrammen ist so sortiert, dass die Datenvorverarbeitungsmethode mit dem größten Haupteffekt links und die mit dem geringsten Haupteffekt rechts im Diagramm dargestellt wird. Sie zeigen die Mittelwerte der Zielgrößen an, in welchen Versuchsläufen die Methoden angewendet werden und in welchen nicht. Die Anwendung einer Methode stellt die positive Faktorstufe (+) im SVP dar und ist in der Implementierung mit einer 1 in der Datenmatrix des SVP gekennzeichnet. Die Nicht-Anwendung wird durch die negative Faktorstufe (-) und mit einer 0 in der Datenmatrix symbolisiert. Die beiden Mittelwerte werden durch unterschiedlich farbige Punkte dargestellt und im

Diagramm durch eine Linie verbunden. Die Differenz zwischen den Mittelwerten stellt den Haupteffekt dar. In den Effekt-Diagrammen lässt sich ein stärkerer Haupteffekt demnach an der höheren Steigung der Verbindungslinie zwischen dem Mittelwert - und dem Mittelwert + intuitiv feststellen. Zu Beginn werden die Haupteffekte von verschiedenen Methoden zur Behandlung von fehlenden Werten ausgewertet und in Abbildung 4.3 dargestellt.

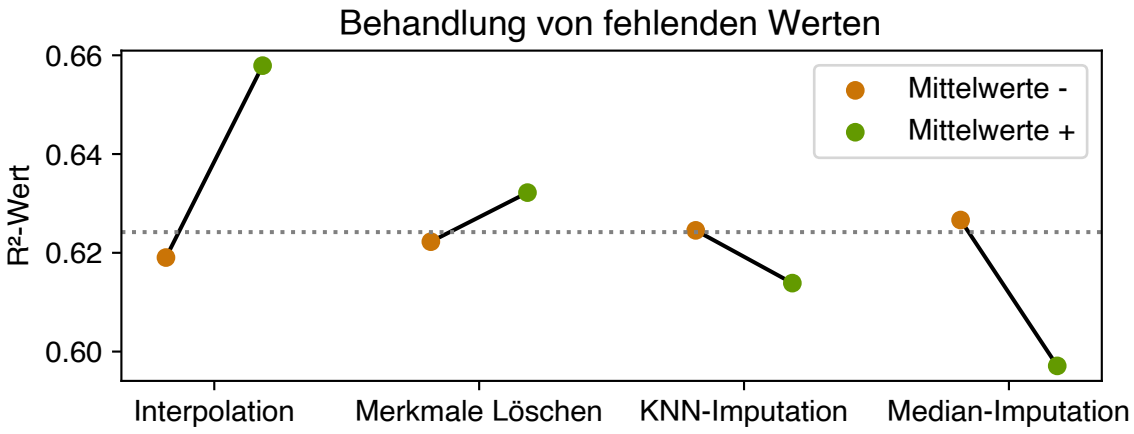

Abbildung 4.3: Effekt-Diagramm mit Haupteffekten ausgewählter Methoden zur Behandlung fehlender Werte

In Abbildung 4.3 werden die Haupteffekte der Methoden Interpolation, Merkmale löschen, KNN-Imputation und der Median-Imputation dargestellt. Um die Darstellung kompakt zu halten, ist dies nur eine Auswahl aller angewendeten Methoden, die jedoch den besten und schlechtesten Haupteffekt sowie alle Kategorien der Methoden aus Tabelle 3.5 repräsentiert. Zwei ausgewählte Methoden haben bei Anwendung einen positiven Einfluss und somit einen positiven Haupteffekt auf die Zielgröße und zwei einen negativen Einfluss. Ein negativer Einfluss bedeutet in dieser exemplarischen Anwendung des Experimentierplans, dass im Mittel bei den Versuchsläufen, bei denen eine KNN-Imputation oder eine Median-Imputation in der Datenvorverarbeitung angewendet wurde, die Vorhersage der Messwerte ungenauer gewesen ist, als bei den Versuchsläufen, wo diese Datenvorverarbeitungsmethoden nicht angewendet worden sind. Die lineare Interpolation hat bei Anwendung den größten positiven Einfluss auf die Zielgröße. Dies bestätigt die Tauglichkeit der Interpolation bei Zeitreihendaten (vgl. Kapitel 2.2.2). Die Methode, ganze Merkmale zu löschen, wenn diese über 20% fehlende Werte haben, hat auch einen positiven Haupteffekt, der jedoch deutlich kleiner ausfällt, als der der linearen Interpolation. Es kann also festgehalten werden, dass die lineare Interpolation für diesen Datensatz mit Abstand am besten die fehlenden Werte behandelt.

Für die Methoden der Datentransformation werden zwei Effekt-Diagramme benutzt, die verschiedene Datentransformationsaufgaben beinhalten. Die Haupteffekte der Datentransformationen, die der Skalierung zugeordnet sind, werden im ersten Effekt-Diagramm nur unter sich verglichen, weil die Skalierung der Daten beim Abgleich von den Eigenschaften des Datensatzes und den Voraussetzungen der KNN-Regression gefordert worden ist (vgl. Kapitel 4.3). Dieses Effekt-Diagramm wird nachfolgend in Abbildung 4.4 dargestellt.

Abbildung 4.4 zeigt die Haupteffekte aller angewendeten Skalierungsmethoden, die im Diagramm verglichen werden können. Die Normalisierung verbessert die Vorhersage nur minimal, sodass nur

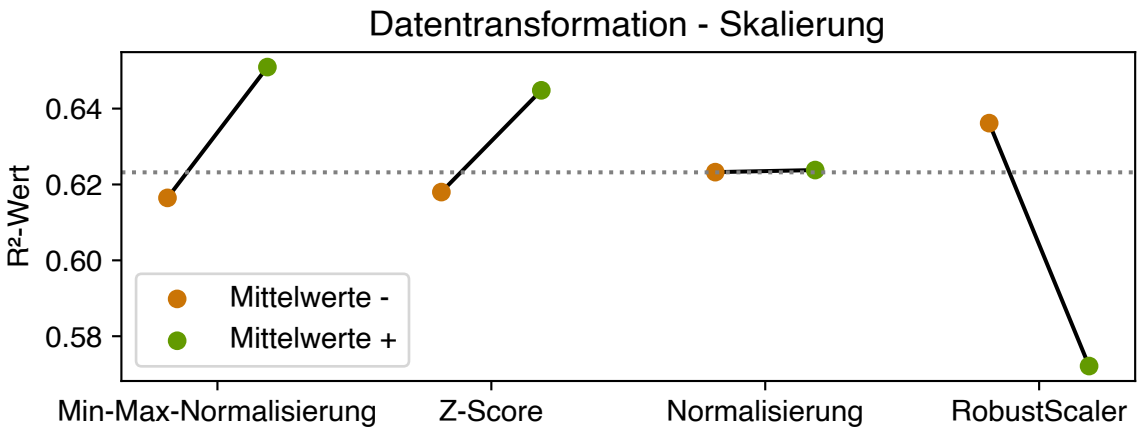

Abbildung 4.4: Effekt-Diagramm mit Haupteffekten von Datentransformationsmethoden zur Skalierung

eine sehr geringe Steigung im Diagramm erkennbar ist. Der RobustScaler hat einen stark negativ ausgeprägten Haupteffekt und verschlechtert die Zielgröße in jedem Versuchslauf, in dem dieser angewendet wird. Die Min-Max-Normalisierung und die Z-Score-Standardisierung schneiden beide ähnlich ab und verbessern den R^2-Wert im Durchschnitt um mehr als 2%. Wenn in der Datenvorverarbeitung nur eine Skalierung angewendet werden sollte, kann die Min-Max-Normalisierung als beste Skalierung geschlussfolgert werden. Da die Z-Score-Standardisierung jedoch ähnlich gute Ergebnisse verspricht, kann es aufgrund von Wechselwirkungen zu anderen Datenvorverarbeitungsschritten auch dazu kommen, dass diese Methode schlussendlich bei dem Versuchslauf angewendet wird, der die beste Datenvorverarbeitung verspricht.

Im Gegensatz zur Skalierung wurden die Datentransformationsaufgaben Aggregation, Datenglättung und Merkmalskonstruktion nicht gefordert und werden mit der Überlegung angewendet, die Vorhersagegenauigkeit zu verbessern. Die Haupteffekte dieser Datentransformationsaufgaben werden in Abbildung 4.5 angezeigt.

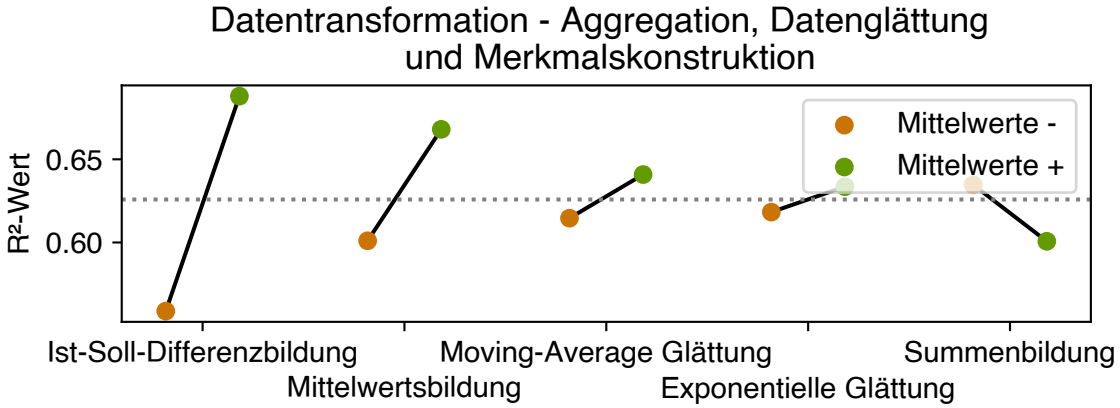

Abbildung 4.5: Effekt-Diagramm mit Haupteffekten von Datentransformationsmethoden zur Aggregation, Datenglättung und Merkmalskonstruktion

Die Abbildung 4.5 zeigt die Haupteffekte der restlichen Datenvorverarbeitungsmethoden. Die fünf

abgebildeten Methoden können den Aufgaben Aggregation, Datenglättung und Merkmalskonstruktion zugeordnet werden. Von diesen haben vier einen positiven und eine negativen Haupteffekt. Im Vergleich zu den Methoden aus den vorherigen Effekt-Diagrammen sind diese Haupteffekte im Allgemeinen größer, was durch die Behebung des Rauschens in den Daten durch diese Methoden erklärt werden kann (vgl. Kapitel 4.1). Die Methoden der Aggregation, wie etwa die Mittelwertsbildung und die Summenbildung, sind dabei unterschiedlich ausgeprägt. Die Bildung des Mittelwerts kann die Vorhersage verbessern, während die Bildung der Summe diese verschlechtert. Die beiden Glättungsmethoden resultieren in einem leicht positiven Haupteffekt, wobei die Moving-Average-Glättung den größeren positiven Haupteffekt auf die Vorhersage aufweist. Die Merkmalskonstruktion mit der Bildung der Ist-Soll-Differenz erweist sich als am vorteilhaftesten. Diese ermöglicht die Prognose neuer Zielwerte und erlaubt mittels Addition der Sollwerte eine theoretische Umkehrung der Rechnung, sodass sich die absoluten Messwerte zurückrechnen lassen.

Die Präsentation der Haupteffekte wird mit der Darstellung aller Haupteffektgrößen in einem zusammenfassenden Balkendiagramm finalisiert. Dies stellt das Hauptergebnis der Arbeit dar, da es einen Vergleich aller angewendeten Methoden auf einen Blick ermöglicht. Der Größe nach sortiert, werden alle berechneten Haupteffekte der Datenvorverarbeitungsmethoden zur Übersicht in der nächsten Abbildung 4.6 gezeigt.

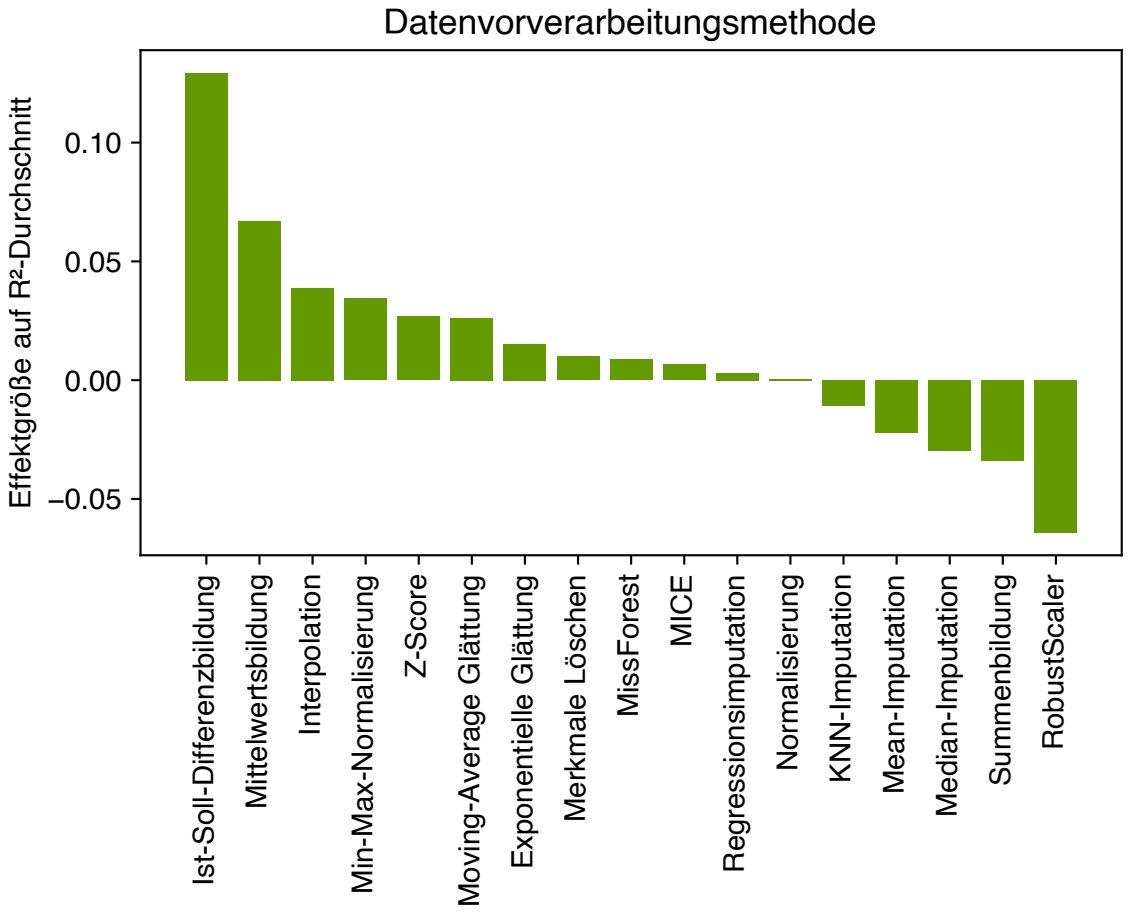

Abbildung 4.6: Haupteffekte auf den R^2-Durchschnitt der Datenvorverarbeitungsmethoden

Abbildung 4.6 zeigt die Haupteffekte der Datenvorverarbeitungsmethoden auf die Zielgröße, den

durchschnittlichen R^2-Wert. Die Methoden sind von links nach rechts nach ihrer Größe geordnet. Zwölf der 17 Datenvorverarbeitungsmethoden haben einen positiven Haupteffekt auf die Zielgröße, was für die generelle Anwendung einer Datenvorverarbeitung spricht, jedoch auch verdeutlicht, dass die Anwendung der richtigen Methoden essenziell ist. Den insgesamt kleinsten Haupteffekt hat die Anwendung des RobustScalers auf den Datensatz, gefolgt von der Summenbildung, den Imputationen der Lageparameter und der KNN-Imputation. Dies sind alle untersuchten Datenvorverarbeitungsmethoden, die sich negativ auf die Vorhersagegenauigkeit ausgewirkt haben. Die Anwendung der Normalisierung hat eine vernachlässigbar kleine Auswirkung auf die Vorhersagegenauigkeit gehabt. Einen leicht besseren Haupteffekt konnten die Methoden Regressionsimputation, MICE, MissForest und Merkmale löschen erzielen, die zur Behandlung von fehlenden Werten benutzt werden. Im Mittel wurde durch diese vier Methoden eine Verbesserung der Vorhersage um 1% nicht erreicht. Die zwei angewendeten Datenglättungen folgen in der aufsteigenden Reihenfolge der Haupteffekte. Anschließend kommen die beiden Skalierungen Z-Score und Min-Max-Normalisierung, bevor die Interpolation als drittbeste angewendete Datenvorverarbeitungsmethode aufgelistet wird. Bis hier ist die Effektgröße für den R^2-Durchschnitt unter vier Prozent. Die besten Methoden, die sich im Vergleich zu den anderen Methoden durch höhere Haupteffekte auszeichnen, sind die Mittelwertbildung, die der Datenaggregation zugeordnet wird, sowie insbesondere die Ist-Soll-Differenzbildung, die eine neue Zielvariable konstruiert. Nach dem Vergleich der Haupteffekte sind die fünf Methoden, mit denen jede gewählte Aufgabe der Datentransformation bearbeitet wird und mit denen fehlende Werte behandelt werden können, die Ist-Soll-Differenzbildung, die Bildung von Mittelwerten, die Interpolation, die Min-Max-Normalisierung und die Moving-Average-Glättung.

Nach den Haupteffekten werden die Zweifach-Wechselwirkungen der Methoden betrachtet. Aus den 17 angewendeten Methoden können 136 verschiedene Wechselwirkungen abgeleitet werden. Diese werden in der folgenden Abbildungen 4.7 als Gesamtheit dargestellt.

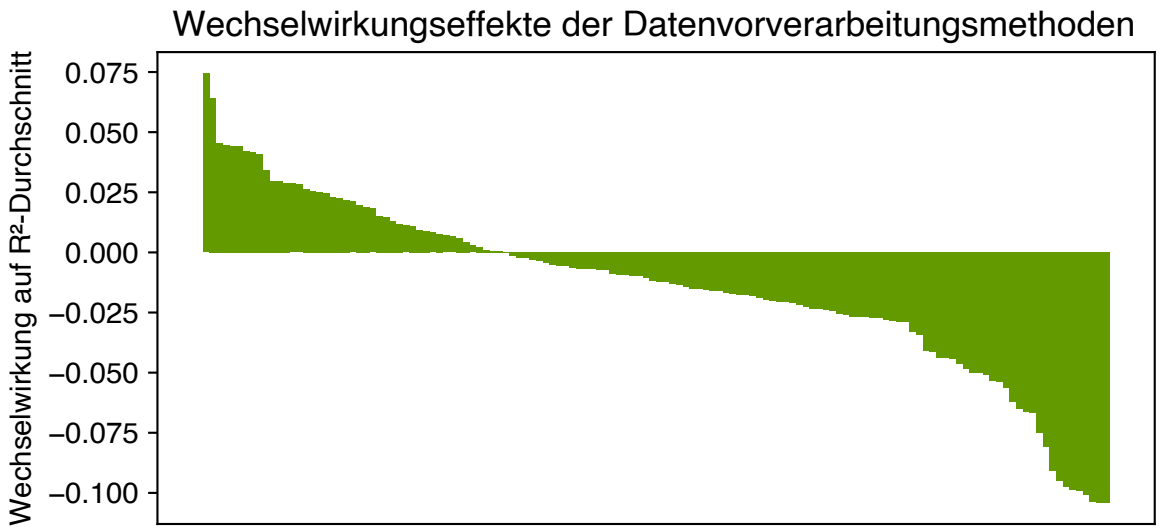

Abbildung 4.7: 136 Wechselwirkungen der Datenvorverarbeitungsmethoden

In Abbildung 4.7 werden alle Wechselwirkungen von jeweils zwei untersuchten Datenvorverarbeitungsmethoden dargestellt. Für eine bessere Lesbarkeit des Diagramms wird die Achsenbeschriftung mit den Wechselwirkungen nicht dargestellt. Die genauen Werte der Wechselwirkungen sind

nicht relevant, weil diese nicht direkt für einen Vergleich der Datenvorverarbeitungsmethoden herangezogen werden. Aus der Abbildung geht jedoch hervor, dass die Mehrheit der Wechselwirkungen der Datenvorverarbeitungsmethoden einen negativen Einfluss auf die Vorhersagegenauigkeit und somit auf den R^2-Wert haben. Im Kontrast dazu stehen die überwiegend positiven Haupteffekte der Methoden. Es wird die These aufgestellt, dass viele Methoden sich gegenseitig hemmen. Die beste Kombination an Datenvorverarbeitungsmethoden ist demnach nicht allein mit Abbildung 4.6 herauszufinden, sondern benötigt die Berücksichtigung der Wechselwirkungen. Da ein vollfaktorieller Versuchsplan durchgeführt worden ist, können die besten Kombinationen an Methoden zur Behandlung fehlender Werte und zur Datentransformation aus den Ergebnissen des SVP abgelesen werden.

In den Ergebnissen des SVP haben die besten neun Versuchsläufe einen identischen Eintrag in der Ergebnisspalte für den durchschnittlichen R^2-Wert. Dieser liegt bei 0,8445. In diesen neun Versuchsläufen werden stets die Ist-Soll-Differenzbildung, die exponentielle Datenglättung, die Mittelwertsbildung und der RobustScaler angewendet. Der Unterschied zwischen den Versuchsläufen liegt in der Behandlung fehlender Werte. Für das Ergebnis ist es unerheblich, welche Methode zur Behandlung fehlender Werte angewendet wird. Auch das Nichtbehandeln fehlender Werte, sodass die Nullen im Datensatz erhalten bleiben, hat keine Auswirkung. Dies liegt für die Imputationen daran, dass durch die Trennung der Methoden in die Anwendung vor oder nach dem Train-Test-Split die Aggregation vor der Imputation angewendet wird. Dabei werden fehlende Werte, die im Datensatz als 0 ausgeprägt sind (vgl. Kapitel 4.1), mit anderen Werten zusammengefasst und deshalb indirekt behandelt.

Um zu untersuchen, ob die Einschränkung der Anwendungsreihenfolge der Methoden durch den Train-Test-Split weitere Auswirkungen auf das Ergebnis hat, wurde der Experimentierplan erneut durchgeführt. Dieses Mal wurden alle Datenvorverarbeitungsmethoden vor dem Train-Test-Split angewendet, wodurch ein Datenleck entstand (vgl. Kapitel 2.2.5). Die feste Reihenfolge dabei ist: 1) Behandlung fehlender Werte, 2) Skalierung der Werte, 3) Merkmalskonstruktion, 4) Aggregation und 5) Datenglättung. Die Unterschiede in den Ergebnissen werden in Abbildung 4.8 dargestellt.

Abbildung 4.8 zeigt ein Balkendiagramm, das die Haupteffekte der Methoden aus den beiden Durchgängen des Experimentierplans vergleicht. Die Methoden werden in einem Durchgang alle vor dem Train-Test-Split angewendet. Ihre Haupteffekte werden mit orangen Balken dargestellt und sind absteigend von links nach rechts geordnet. Die grünen Balken daneben stammen aus dem Durchgang, in dem die Methoden vor oder nach dem Train-Test-Split angewendet wurden, und sind identisch zu denen in Abbildung 4.6. Ein Vergleich der Haupteffekte zeigt, dass Unterschiede in der Höhe der Haupteffekte und somit auch in der Reihenfolge der Methoden vorhanden sind, jedoch vier der fünf besten Methoden beider Durchgänge identisch sind. Hierzu zählen die Ist-Soll-Differenzbildung, die Mittelwertbildung, die Z-Score-Standardisierung und die Interpolation. Der beste Versuchslauf ohne Aufteilung der Methoden erreicht einen durchschnittlichen R^2-Wert von 0,82 und ist damit etwas schlechter als der beste Versuchslauf mit Aufteilung der Methoden. In diesem Versuchslauf werden eine Interpolation, eine Min-Max-Normalisierung, eine Mittelwertbildung, eine Datenglättung mit dem Moving-Average und eine Ist-Soll-Differenzbildung auf der Zielvariablen angewendet. Das Phänomen, dass fehlende Werte durch eine Aggregationsmethode indirekt behandelt werden, und, dass dadurch mehrere Versuchsdurchläufe den gleichen R^2-Wert aufweisen, treten nicht ein.

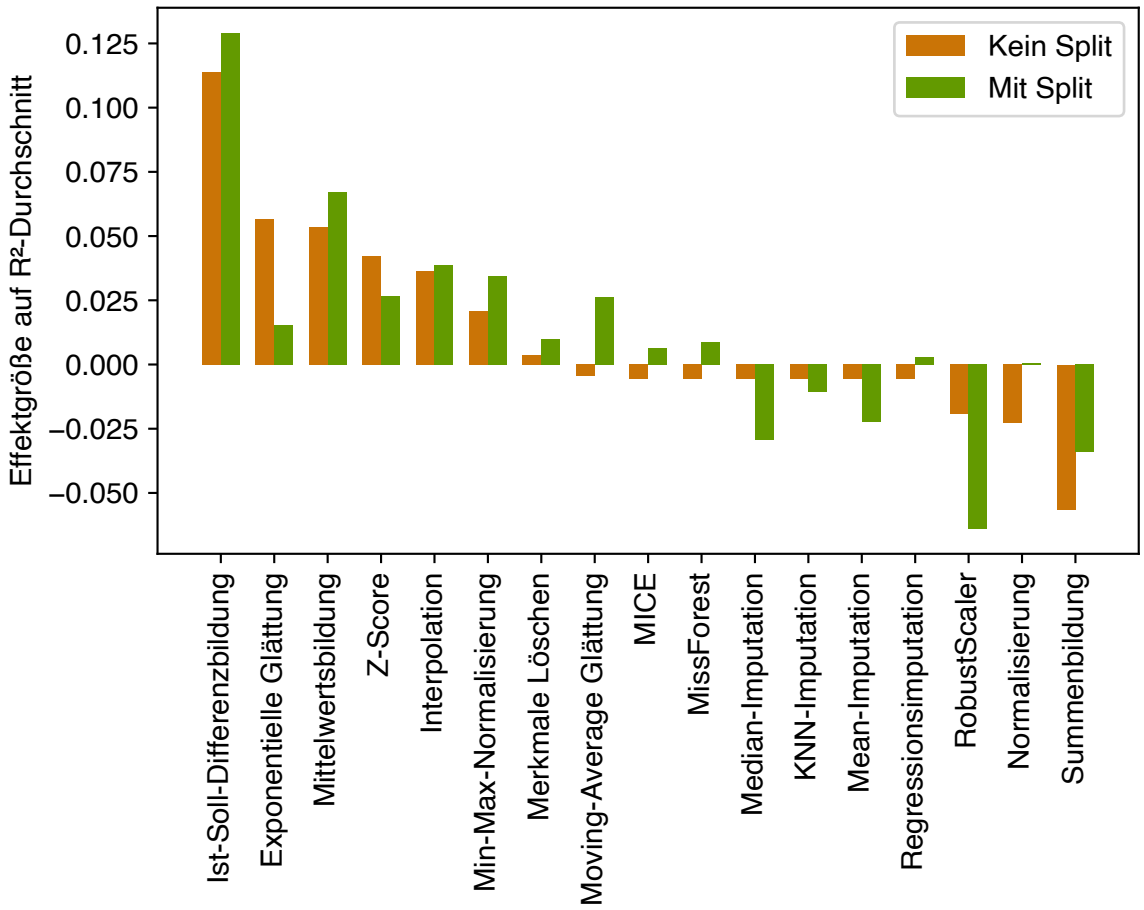

Abbildung 4.8: Vergleich der Haupteffekte auf den R^2-Durchschnitt mit und ohne Train-Test-Split

In beiden Durchgängen des Experimentierplans liegen die Ergebnisse vieler Kombinationen von Methoden in der Vorhersagegenauigkeit nah beieinander. Nach der Auswertung besteht demnach noch die Möglichkeit, die Datenvorverarbeitung anzupassen, indem eine andere gute Kombination gewählt wird, wodurch nur geringe Verluste in der Vorhersagegenauigkeit in Kauf genommen werden müssen. Dabei können auch andere Kriterien berücksichtigt werden, die nicht zu den Vergleichsmetriken zählen.

Nach der Auswertung der Ergebnisse, die durch die exemplarische Anwendung des Experimentierplans erhalten wurden, werden die Ergebnisse mitsamt dem Experimentierplan und der exemplarischen Implementierung diskutiert. Mit den Erkenntnissen aus der Diskussion wird auch ein abschließendes Fazit gezogen.

5 Diskussion und Fazit

In diesem Kapitel wird der in Kapitel 3 entwickelte Experimentierplan diskutiert, bevor auf dieser Grundlage ein abschließendes Fazit gezogen wird. Zuerst wird dafür der Experimentierplan in der aktuellen wissenschaftlichen Literatur eingeordnet. Anschließend werden die Anforderungen an den Experimentierplan aus Kapitel 3.2 abschnittsweise erneut betrachtet und mithilfe der Erkenntnisse, die während der exemplarischen Anwendung (vgl. Kapitel 4) erlangt wurden, diskutiert.

In der bestehenden wissenschaftlichen Literatur existieren bereits Vergleiche von Methoden, die Daten transformieren oder fehlende Werte Behandeln. Diese wurden in der Regel nur auf einer geringen Anzahl an Datensätzen durchgeführt. Für die Datentransformation sind die Vergleiche nur innerhalb einer Datentransformationsaufgabe durchgeführt worden. Zum Beispiel haben García et al. (2013) Methoden der Diskretisierung verglichen. Amin et al. (2019) und Pan et al. (2016) haben Methoden verglichen, die der Skalierung zugeordnet werden können. García et al. (2013) und Amin et al. (2019) haben die Methoden unter der Anwendung verschiedener Data-Mining-Algorithmen verglichen und somit auch den spezifischen Einfluss der Datentransformationen auf den Data-Mining-Algorithmus untersucht. Für die Behandlung von fehlenden Werten sind zum Beispiel von Jäger et al. (2021), Sun et al. (2023) oder Rockel (2022) Vergleiche von Methoden durchgeführt worden. Letzterer stellt zudem als finale Implikation seiner umfassenden Güteuntersuchung heraus: „Es gibt nicht das universell beste Imputationsverfahren. Vielmehr sollte in Abhängigkeit von der Fragestellung bzw. dem Analyseziel und der Datenmatrix stets ein geeignetes Imputationsverfahren ausgewählt werden.“ (Rockel 2022, S. 232). Es wird deutlich, dass es nur schwer möglich ist, alle äußeren Umstände, wie beispielsweise den Data-Mining-Algorithmus oder die Fragestellung, in einen Vergleich mit einzubeziehen. Die heterogene Struktur von Produktionsdaten, die in Kapitel 2.3.3 erwähnt wird, sorgt dafür, dass die Ergebnisse eines Vergleichs auf Produktionsdaten nicht allgemein übertragbar sind. Zusätzlich gibt es nach dem Kenntnisstand des Verfassers dieser Arbeit keine oder nur sehr wenige Vergleiche, die Methoden mehrerer Aufgaben der Datenvorverarbeitung zusammen untersuchen und vergleichen. In Kapitel 1 wurde herausgestellt, dass die Behandlung fehlender Werte und die Datentransformation für die Anwendung eines Data-Mining-Algorithmus verpflichtend sein können. Deshalb wurde mit dem Experimentierplan ein anpassbarer Vergleichsrahmen in Kapitel 3.3 dieser Arbeit erstellt, um die Lücke in der Literatur durch den Vergleich von Methoden zur Behandlung fehlender Werte und Datentransformationen zu verkleinern.

Zu Beginn wird überprüft, ob die grundlegenden Anforderungen an den Experimentierplan aus Kapitel 3.2.1 als erfüllt gelten. Anforderung 1 ist erfüllt, weil der Experimentierplan die Auswahl und Anwendung von Methoden zur Behandlung fehlender Werte und zur Datentransformation vorsieht. Im Zweifelsfall werden fehlende Werte erzeugt, um sicherzustellen, dass die Behandlung fehlender Werte durchgeführt werden kann. Anforderung 2 wird erfüllt, indem eine ausgewählte Vergleichs-

metrik das Muster oder Modell des Data-Mining-Algorithmus bewertet. Die Versuchsdurchführung mit einem SVP ermöglicht eine systematische Untersuchung und erfüllt Anforderung 3. Die Anforderung 4 verlangt, dass der Experimentierplan von einer Person mit Erfahrung im KDD-Prozess durchgeführt wird. Diese Anforderung kann nicht direkt im Experimentierplan überprüft werden. Trotz Erfahrung im KDD-Prozess ist es möglich, dass diese nicht ausreicht. Die exemplarische Anwendung hat zudem gezeigt, dass die Person neben der Erfahrung im KDD-Prozess auch tiefes Wissen über den Produktionsprozess, aus dem die Daten erhoben wurden, benötigt. Dadurch kann es sein, dass zwei Anwender mit unterschiedlich gutem Wissen über den KDD- und den Produktionsprozess unterschiedliche Methoden vergleichen und somit nicht das gleiche Ergebnis im Experimentierplan erhalten. Sollte die Durchführung des Experimentierplans nicht in einer Programmiersprache möglich sein, weil die Programmierkenntnisse des Anwenders nicht ausreichend sind, kann dieser auf ein Programm für Datenanalysen zurückgreifen. Dadurch wird die Auswahl der Methoden, die verglichen werden sollen, ebenso beeinflusst und auf die Auswahl des Programms beschränkt. Obwohl der Experimentierplan kleinschrittiger ist als das KDD-Vorgehensmodell von Fayyad et al. (1996a) und zusätzliche Schritte enthält, ist die Ähnlichkeit zu diesem Vorgehensmodell gegeben, sodass Anforderung 5 erfüllt ist. Die Möglichkeit, zu einem vorherigen Schritt zurückzugehen und Anpassungen vorzunehmen, ist ein Teil der Ähnlichkeit zum Vorgehensmodell. Zudem wird dadurch die Anforderung 6 erfüllt, die eine Rückführungsmöglichkeit fordert.

Weiterführend wird die Erfüllung der Anforderungen von Produktionsdaten aus Kapitel 3.2.2 überprüft. In der exemplarischen Anwendung aus Kapitel 4.3 wurde eine grundlegende Datenbasis für die Durchführung des SVP geschaffen. In dieser Anwendung ist Anforderung 7 demnach erfüllt worden. Im allgemeinen wird diese Anforderung erfüllt, falls der Anwender genügend Erfahrung im KDD-Prozess und in der Datenvorverarbeitung besitzt. Damit die richtigen Datenvorverarbeitungsmethoden angewendet werden, werden im Experimentierplan neben den Voraussetzungen des Data-Mining-Algorithmus auch die Eigenschaften des Datensatzes abgefragt, womit Anforderung 8 erfüllt wird. Auch wenn der Datensatz die Big Data Eigenschaften erfüllt, soll der Experimentierplan auf diesen angewendet werden können, wie Anforderung 9 fordert. In der Theorie wird dies erfüllt, weil darauf hingewiesen wird, dass es Methoden gibt, die auf Big Data spezialisiert sind. In der exemplarischen Anwendung wurde diese Theorie jedoch nicht überprüft. Bei Big Data könnte es dazu kommen, dass die Rechenzeiten im SVP lange dauern. Die gesamte Rechenzeit kann in diesem Fall mit teilfaktoriellen Versuchsplänen verkürzt werden, da der Experimentierplan keinen bestimmten Versuchsplan vorschreibt.

Die Anforderungen von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation werden in Kapitel 3.2.3 erhoben und nachfolgend diskutiert. Anforderung 10 wird mit der richtigen Methodenauswahl und der korrekten Erstellung des SVP erfüllt. In der exemplarischen Anwendung zeichnete sich dies dadurch aus, dass der Data-Mining-Algorithmus in jedem Versuchslauf angewendet werden konnte und Ergebnisse erzielte. Die korrekte Erstellung des SVP beinhaltet ebenso die Berücksichtigung der Randbedingungen. Eine Randbedingung geht von Anforderung 11 aus und fordert, dass nur eine Methode zur Behandlung von fehlenden Werten pro Versuchslauf angewendet werden kann. Mit der Implementierung aus Kapitel 4.2 ist dies erfüllt (vgl. Anhang B.3). Dies gilt Analog für die Anwendung von nur einer Datentransformationsmethode je Datentransformationsaufgabe. Diese Randbedingung ist ein Teil von Anforderung 12. Zusätzlich wird gefordert, die erfor-

derlichen Datentransformationsaufgaben zu bestimmen. Durch den Abgleich der Voraussetzungen des Data-Mining-Algorithmus mit den Eigenschaften des Datensatzes und unter Zuhilfenahme ausreichender Erfahrung können die passenden Aufgaben der Datentransformation abgeleitet werden, wodurch diese Anforderung erfüllt wird. Die Randbedingungen, die in Kapitel 3.2.3 erhoben wurden, schränken die Bildung des SVP ein. In der exemplarischen Anwendung war es möglich, einen vollfaktoriellen Versuchsplan umzusetzen, ohne dass umfassende Erfahrung in der Erstellung solcher Pläne erforderlich war. Die Bildung teilfaktorieller Versuchspläne unter diesen Randbedingungen würde hingegen mehr Erfahrung in der Erstellung von Versuchsplänen voraussetzen.

Zuletzt werden die Anforderungen von Vergleichsmetriken aus Kapitel 3.2.4 diskutiert. Die Anforderung 13, die Methoden vor oder nach dem Train-Test-Split einzuteilen und anzuwenden, wird erfüllt. In der exemplarischen Anwendung hat sich jedoch gezeigt, dass es dadurch nicht möglich ist, eine Reihenfolge bei der Anwendung der Methoden vorzugeben. Ein zweiter Durchgang des Experimentierplans, bei dem alle Methoden vor dem Train-Test-Split angewendet wurden, hat zu anderen Ergebnissen geführt. Anforderung 14 fordert, dass der Datensatz mit einem Data-Mining-Algorithmus brauchbare Ergebnisse liefern muss. Diese Anforderung ist durch die Rückführungsmöglichkeit erfüllt, da die Data-Mining-Aufgabe mit dem Algorithmus so lange angepasst werden kann, bis die Ergebnisse brauchbar sind. Alternativ kann der Experimentierplan abgebrochen werden, wodurch die Anforderung hinfällig wird. Die brauchbaren Ergebnisse sollten für Anforderung 15 Spielraum für Verbesserungen lassen, damit die Auswirkungen einer Datenvorverarbeitung messbar sind. In der exemplarischen Anwendung (vgl. Kapitel 4.3) waren die Ergebnisse des initialen Durchgangs klar verbesserungsfähig und die Anforderung war erfüllt. Es kann jedoch dazu kommen, dass die Entscheidung, ob eine Verbesserung durch eine Datenvorverarbeitung möglich ist, nicht so einfach ist. Dafür ist es möglich auch nach der Durchführung des SVP mit einer Rückführung Anpassungen im Experimentierplan vorzunehmen.

Auf Grundlage der Diskussion kann im Folgenden ein Fazit gezogen werden. Die Erstellung des Experimentierplans und die damit verbundene Schließung einer Lücke in der wissenschaftlichen Literatur waren erfolgreich. Alle erhobenen Anforderungen konnten erfüllt werden. Während der Diskussion gab es zwar Anmerkungen zu drei Anforderungen, die einen erfolgreichen Vergleich der Methoden zur Behandlung fehlender Werte und zur Datentransformation jedoch nicht verhindert haben. Sie zeigen das Potenzial auf, den Experimentierplan weiterzuentwickeln. Die Weiterentwicklung würde auf einer strukturierten Basis stattfinden, die jedoch individuell anpassbar ist und somit auch in der Zukunft noch angewendet werden kann. Außerdem wurde herausgestellt, dass die Erfahrung des Anwenders im KDD-Prozess eine nicht zu vernachlässigende Rolle spielt und nicht vollständig durch den Experimentierplan ersetzt werden kann.

Insgesamt war es möglich, fast alle Anforderungen der exemplarischen Anwendung in der Praxis zu überprüfen und als erfüllt zu bewerten. Das Ergebnis aus Kapitel 4.4, dass die Methoden, die das Rauschen im Datensatz beseitigen, gut abschneiden und die Interpolation die fehlenden Werte am besten behandelt, passt zu den Eigenschaften des Datensatzes, die in Kapitel 4.1 vorgestellt wurden. Das Hauptziel der Arbeit, einen Experimentierplan zu entwickeln, der den Vergleich verschiedener Methoden zur Behandlung fehlender Werte und zur Datentransformation ermöglicht, wurde erreicht und in der exemplarischen Anwendung wurden sinnvolle Ergebnisse erzielt.

6 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Experimentierplan entwickelt, der darauf abzielt, Methoden zur Behandlung fehlender Werte sowie zur Datentransformation systematisch zu vergleichen. Diese Schritte gehören zur Datenvorverarbeitung, einem essenziellen Prozess, um Datensätze für Data-Mining-Anwendungen vorzubereiten. Die Datenvorverarbeitung und Data Mining sind Bestandteile des KDD-Prozesses. Der entwickelte Experimentierplan wurde speziell für die Anwendung auf Produktionsdatenbanken konzipiert.

Zu Beginn wurde der KDD-Prozess mit Grundlagen zu Daten, Informationen und Wissen, statistischen Grundlagen sowie einem KDD-Vorgehensmodell dargestellt. Anschließend wurde gesondert auf die Data-Mining-Phase eingegangen. Im Anschluss wurde die Datenvorverarbeitung vertieft und der Fokus auf die Behandlung fehlender Werte, die Datentransformation und die Bewertung dieser Schritte gelegt. Das Kapitel schloss mit einer umfassenden Darstellung der Wissensentdeckung in der Domäne der Produktion ab, wobei sowohl die Datenerfassung als auch die Eigenschaften der Daten und die Data-Mining-Anwendungsgebiete auf die Spezialisierung in der Produktion ausgerichtet waren.

Die Erstellung des Experimentierplans basiert auf diesen Grundlagen. Nachdem das Ziel des Experimentierplans offengelegt wurde, wurden Anforderungen an den Experimentierplan ausgearbeitet. Sechs Anforderungen wurden direkt an den Experimentierplan gestellt. Jeweils drei weitere Anforderungen kamen von den Produktionsdaten, den Datenvorverarbeitungsmethoden und den Vergleichsmetriken. Abschließend wurde der Experimentierplan zum Vergleich von Methoden zur Behandlung von fehlenden Werten und zur Datentransformation mithilfe der 15 Anforderungen erstellt. Zur Veranschaulichung wurde der Experimentierplan mit einem Flussdiagramm dargestellt.

Nach seiner Erstellung wurde der Experimentierplan exemplarisch auf einen originalen Datensatz aus der Produktion angewendet. Dazu wurden der Produktionsdatensatz, die Herkunft der Daten, die Dateneigenschaften und die angedachte Data-Mining-Aufgabe vorgestellt. Anschließend wurde die auf den Datensatz angepasste Version des Experimentierplans in Python implementiert, um diesen auf den Datensatz anzuwenden. Die Implementierung wurde nach der Vorstellung des Datensatzes detailliert beschrieben. Im Anschluss wurde die Anwendung des Experimentierplans mit allen getroffenen Entscheidungen und Wahlen dargestellt. Die produzierten Ergebnisse aus der Anwendung des Experimentierplans wurden ausgewertet, um einen Vergleich der Methoden herzustellen.

Abschließend wurde die Gesamtarbeit diskutiert. Dabei wurde herausgestellt, dass sowohl die Erstellung des Experimentierplans als auch dessen exemplarische Anwendung erfolgreich waren. Dazu wurde die Erfüllung der Anforderungen überprüft. Alle Anforderungen wurden erfüllt, es gab je-

doch auch Anmerkungen, die eine Weiterentwicklung des Experimentierplans motivieren.

Eine potenzielle Weiterentwicklungsmöglichkeit ist die Einbindung des Experimentierplans in den KDD-Prozess, sodass die untersuchte Datenvorverarbeitung automatisch auf den Datensatz angewendet wird. Dies würde eine Untersuchung der Anwendungsreihenfolge der Datenvorverarbeitungsmethoden mit sich bringen. Wie in der Auswertung der Ergebnisse (vgl. Kapitel 4.4) gezeigt wurde, hat die Anwendungsreihenfolge der Datenvorverarbeitungsmethoden eine Auswirkung auf das Ergebnis der Datenvorverarbeitung. Außerdem ist bei der Implementierung aufgefallen, dass die Datenvorverarbeitungsmethoden einige Eingabeparameter benötigen. Eine zusätzliche Variation der Eingabeparameter müsste für ein bestmögliches Ergebnis deshalb ebenso untersucht werden. Da diese zwei Erweiterungen die Faktoren im SVP und damit die Rechenzeit deutlich erhöhen würden, ist eine Aufteilung der Untersuchungen und der abschließenden finalen Anwendung der Datenvorverarbeitung in verschiedenen Stufen denkbar.

Da in der Diskussion herausgestellt wurde, dass die Erfahrung im KDD-Prozess nur schwer zu quantifizieren ist und ein wenig Erfahrung für den Experimentierplan vorausgesetzt wird, wäre die Untersuchung spannend, inwieweit der Experimentierplan Erfahrung im KDD-Prozess ersetzen kann. Es könnte die Forschungsfrage gestellt werden, ob ein weniger Erfahrener Data Scientist mit dem Experimentierplan vorgegebene Daten besser für eine Data-Mining-Aufgabe vorbereitet als ein Erfahrener Data Scientist.

Literaturverzeichnis

Amin, Adnan, Babar Shah, Asad M. Khattak, Fernando J. Lopes Moreira, Gohar Ali, Alvaro Rocha und Sajid Anwar (2019). „Cross-company customer churn prediction in telecommunication: A comparison of data transformation methods“. In: *International Journal of Information Management* 46, S. 304–319.

Ayaz, Baris (2022). „Maschinelles Lernen“. In: *Analytics in der Industrie: Schlüsseltechnologie für die digitale Transformation*. Hrsg. von Thomas Schulz. 1. Aufl. Würzburg: Vogel Communications Group.

Balakrishnan, Anantaram, Soundar R. T. Kumara und Shankar Sundaresan (1999). „Manufacturing in the Digital Age: Exploiting Information Technologies for Product Realization“. In: *Information Systems Frontiers* 1, S. 25–50.

Bankhofer, Udo (1995). „Unvollständige Daten- und Distanzmatrizen in der multivariaten Datenanalyse“. Diss. Universität Augsburg.

Bauernhansl, Thomas (2014). „Die Vierte Industrielle Revolution - Der Weg in ein wertschaffendes Produktionsparadigma“. In: *Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung · Technologien · Migration*. Hrsg. von Thomas Bauernhansl, Michael Ten Hompel und Birgit Vogel-Heuser. Wiesbaden: Springer Fachmedien.

Bilal, Mehwish, Ghulam Ali, Muhammad Waseem Iqbal, Muhammad Anwar, Muhammad Sheraz Arshad Malik und Rabiah Abdul Kadir (2022). „Auto-Prep: Efficient and Automated Data Preprocessing Pipeline“. In: *IEEE Access* 10, S. 107764–107784.

Bloech, Jürgen, Ronald Bogaschewsky, Udo Buscher, Anke Daub, Uwe Götze und Folker Roland (2014). *Einführung in die Produktion*. Berlin: Springer Gabler.

Borrohou, Sanae, Rachida Fissoune und Hassan Badir (2025). „Critical Role of Data Transformation in Preprocessing: Methods, Algorithms, and Challenges“. In: *Model and Data Engineering*. Hrsg. von Carlos Ordonez, Giancarlo Sperlì, Elio Masciari und Ladjel Bellatreche. Cham: Springer Nature Switzerland, S. 108–122.

Botchkarev, Alexei (2019). „Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology“. In: *Interdisciplinary Journal of Information, Knowledge, and Management* 14, S. 45–76.

Brachman, Ronald J. und Tej Anand (1994). „The Process of Knowledge Discovery in Databases: A First Sketch“. In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. AAAI Conference on Artificial Intelligence. AAAI Workshop Papers 1994. Seattle: AAAI Press, S. 1–12.

Buuren, Stef van und Karin Groothuis-Oudshoorn (2011). „mice: Multivariate imputation by chained equations in R“. In: *Journal of Statistical Software* 45.3, S. 1–67.

Çetin, Volkan und Oktay Yıldız (2022). „A comprehensive review on data preprocessing techniques in data analysis“. In: *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi* 28.2, S. 299–312.

Cheng, Ying, Ken Chen, Hemeng Sun, Yongping Zhang und Fei Tao (2018). „Data and knowledge mining with big data towards smart production“. In: *Journal of Industrial Information Integration* 9, S. 1–13.

Cho, Eunnuri, Tai-Woo Chang und Gyusun Hwang (2022). „Data Preprocessing Combination to Improve the Performance of Quality Classification in the Manufacturing Process“. In: *Electronics* 11.3, S. 477.

Cleve, Jürgen und Uwe Lämmel (2024). *Data Mining: Datenanalyse für Künstliche Intelligenz*. 4. Aufl. De Gruyter Studium. Berlin: De Gruyter.

Dietrich, Manfred (2021). „Daten in der Produktion“. In: *Digitales Shopfloor Management in SAP-Systemumgebungen: Roadmap und Lösungsalternativen für die Umsetzung*. Hrsg. von Manfred Dietrich. Wiesbaden: Springer Fachmedien, S. 39–63.

DIN 66001 (1983). *DIN 66001:1983-12, Informationsverarbeitung; Sinnbilder und ihre Anwendung*. Berlin: DIN Media GmbH.

Dittrich, Jens und Jorge-Arnulfo Quiané-Ruiz (2012). „Efficient big data processing in Hadoop MapReduce“. In: *Proceedings of the VLDB Endowment* 5.12, S. 2014–2015.

Dogan, Alican und Derya Birant (2021). „Machine learning and data mining in manufacturing“. In: *Expert Systems with Applications* 166, Article 114060.

Dong, Weinan, Daniel Yee Tak Fong, Jin-sun Yoon, Eric Yuk Fai Wan, Laura Elizabeth Bedford, Eric Ho Man Tang und Cindy Lo Kuen Lam (2021). „Generative adversarial networks for imputing missing data for big data clinical research“. In: *BMC Medical Research Methodology* 21.1, S. 78.

Enders, Craig K. und Todd D. Little (2010). *Applied missing data analysis*. Methodology in the social sciences. New York: Guilford Press.

Ester, Martin und Jörg Sander (2000). *Knowledge Discovery in Databases*. Berlin: Springer Verlag.

Fayyad, Usama, Gregory Piatetsky-Shapiro und Padhraic Smyth (1996a). „From data mining to knowledge discovery in databases“. In: *AI magazine* 17.3, S. 37–54.

– (1996b). „The KDD process for extracting useful knowledge from volumes of data“. In: *Communications of the ACM* 39.11, S. 27–34.

Fraunhofer IPT und Fraunhofer FFB (2025). *Machine Learning in production*. Fraunhofer Big Data and Artificial Intelligence Alliance. URL: https://www.bigdata-ai.fraunhofer.de/en/business-areas/production_industry/production-datasets.html (besucht am 28. 08. 2025).

García, Salvador, Julián Luengo und Francisco Herrera (2015). *Data Preprocessing in Data Mining*. Bd. 72. Intelligent Systems Reference Library. Cham: Springer International Publishing.

García, Salvador, Julián Luengo, José Antonio Sáez, Victoria López und Francisco Herrera (2013). „A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning“. In: *IEEE Transactions on Knowledge and Data Engineering* 25.4, S. 734–750.

García, Salvador, Sergio Ramírez-Gallego, Julián Luengo, José Manuel Benítez und Francisco Herrera (2016). „Big data preprocessing: methods and prospects“. In: *Big Data Analytics* 1, 9:1–9:22.

Gesellschaft Mess- und Automatisierungstechnik (2013). *Cyber Physical Systems: Chancen und Nutzen aus Sicht der Automation*. Düsseldorf: VDI Verein Deutscher Ingenieure e.V.

Ghazi, Mohamed El (2019). „Statistische Versuchsplanung für Einsteiger“. In: *Unveröffentlicht (DOI: 10.13140/RG.2.2.31748.68487)*.

Gröger, Christoph (2022). „Industrial analytics – An overview“. In: *it - Information Technology* 64.1, S. 55–65.

Grün, Oskar und Werner Jammernegg (2019). *Grundzüge der Beschaffung, Produktion und Logistik*. Hrsg. von Sebastian Kummer. 4. Aufl. Hallbergmoos: Pearson.

Gumm, Heinz-Peter, Manfred Sommer und Wolfgang Hesse (2011). *Einführung in die Informatik*. 9. Aufl. München: Oldenbourg.

Günther, Hans-Otto und Horst Tempelmeier (2014). *Produktion und Logistik*. 11. Aufl. Bd. 1. Produktion und Logistik. Berlin: Springer.

Gutenberg, Erich (1983). *Grundlagen der Betriebswirtschaftslehre*. 24. Aufl. Bd. 1: Die Produktion. Berlin, Heidelberg, New York: Springer.

Han, Jiawei, Jian Pei und Hanghang Tong (2023). *Data mining: concepts and techniques*. 4. Aufl. Cambridge: Morgan Kaufmann Publishers, an imprint of Elsevier.

Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke und Travis E. Oliphant (2020). „Array programming with NumPy“. In: *Nature* 585.7825, S. 357–362.

Hu, S. Jack (2013). „Evolving Paradigms of Manufacturing: From Mass Production to Mass Customization and Personalization“. In: *Procedia CIRP*. 46th CIRP Conference on Manufacturing Systems (29 - 30 May 2013). Bd. 7. Setubal: Elsevier, S. 3–8.

Hunter, J. D. (2007). „Matplotlib: A 2D graphics environment“. In: *Computing in Science & Engineering* 9.3, S. 90–95.

Jäger, Sebastian, Arndt Allhorn und Felix Bießmann (2021). „A Benchmark for Data Imputation Methods“. In: *Frontiers in Big Data* 4:693674.

Junninen, Heikki, Harri Niska, Kari Tuppurainen, Juhani Ruuskanen und Mikko Kolehmainen (2004). „Methods for imputation of missing values in air quality data sets“. In: *Atmospheric Environment* 38.18, S. 2895–2907.

Kaggle (2025). *Multi-stage continuous-flow manufacturing process*. Kaggle. URL: https://www.kaggle.com/datasets/supergus/multistage-continuousflow-manufacturing-process/data (besucht am 02. 09. 2025).

Kamalov, Firuz, Sherif Moussa und Jorge Avante Reyes (2023). „Data Transformation in Machine Learning: Empirical Analysis“. In: *International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies*. 3ICT 2023 (Virtual Conference: 20 - 21 November 2023). Bahrain, S. 115–120.

Kantardzic, Mehmed (2020). *Data mining: concepts, models, methods, and algorithms*. 3. Aufl. Hoboken: John Wiley & Sons.

Kaufman, Shachar, Saharon Rosset, Claudia Perlich und Ori Stitelman (2012). „Leakage in data mining: Formulation, detection, and avoidance“. In: *ACM Transaction on Knowledge Discovery from Data* 6.4, 15:1–15:21.

Kemper, Alfons Heinrich und André Eickler (2015). *Datenbanksysteme: eine Einführung*. 10. Aufl. Berlin: De Gruyter Oldenbourg.

Kluyver, Thomas, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing und Jupyter development team (2016). „Jupyter Notebooks – a publishing format for reproducible computational workflows“. In: 20th International Conference on Electronic Publishing (07-09 Jun 2016). Hrsg. von Fernando Loizides und Birgit Scmidt. Unter Mitarb. von Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing, Jupyter development team, Fernando Loizides und Birgit Scmidt. IOS Press, S. 87–90.

Lang, Kyle M. und Todd D. Little (2018). „Principled Missing Data Treatments“. In: *Prevention Science* 19.3, S. 284–294.

Larose, Daniel T. und Chantal D. Larose (2014). *Discovering Knowledge in Data: an introduction to Data Mining*. 1. Aufl. Hoboken: John Wiley & Sons.

Lasi, Heiner, Peter Fettke, Hans-Georg Kemper, Thomas Feld und Michael Hoffmann (2014). „Industrie 4.0“. In: *Wirtschaftsinformatik* 56, S. 261–264.

Li, Canchen (2019). „Preprocessing Methods and Pipelines of Data Mining: An Overview“. In: *arXiv preprint arXiv:1906.08510*.

Lieber, Daniel, Olga Erohin und Jochen Deuse (2013). „Wissensentdeckung im industriellen Kontext: Herausforderungen und Anwendungsbeispiele“. In: *Zeitschrift für wirtschaftlichen Fabrikbetrieb* 108.6, S. 388–393.

Liew, Anthony (2007). „Understanding Data, Information, Knowledge And Their Inter-Relationships“. In: *Journal of Knowledge Management Practice* 7.2.

Lin, Wei-Chao und Chih-Fong Tsai (2020). „Missing value imputation: a review and analysis of the literature (2006–2017)“. In: *Artificial Intelligence Review* 53.2, S. 1487–1509.

Little, Roderick J. A. und Donald B. Rubin (2020). *Statistical analysis with missing data*. 3. Aufl. Wiley series in probability and statistics. Hoboken: John Wiley & Sons.

Liveline Technologies (2025). *Liveline Technologies*. URL: https://www.liveline.tech (besucht am 28. 08. 2025).

Luengo, Julián, Diego García-Gil, Sergio Ramírez-Gallego, Salvador García und Francisco Herrera (2020). *Big Data Preprocessing: Enabling Smart Data*. Cham: Springer Nature Switzerland.

Mariscal, Gonzalo, Óscar Marbán und Covadonga Fernández (2010). „A survey of data mining and knowledge discovery process models and methodologies“. In: *The Knowledge Engineering Review* 25.2, S. 137–166.

Mertens, Peter, Peter Buxmann, Thomas Hess, Oliver Hinz, Jan Muntermann und Matthias Schumann (2023). *Grundzüge der Wirtschaftsinformatik*. Berlin: Springer Gabler.

Munson, M. Arthur (2012). „A study on the importance of and time spent on different modeling steps“. In: *ACM SIGKDD Explorations Newsletter* 13.2, S. 65–71.

Naidu, Gireen, Tranos Zuva und Elias M. Sibanda (2015). „A Review on Evaluation Metrics for Data Classification Evaluations“. In: *International Journal of Data Mining & Knowledge Management Process* 5.2, S. 1–11.

North, Klaus (2021). *Wissensorientierte Unternehmensführung: Wissensmanagement im digitalen Wandel*. Wiesbaden: Springer Fachmedien.

Olariu, Eliza Maria, Ramona Tolas, Raluca Portase, Mihaela Dinsoreanu und Rodica Potolea (2020). „Modern approaches to preprocessing industrial data“. In: *2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP)*. ICCP (03 - 05 September 2020). Cluj-Napoca: IEEE, S. 221–226.

Pan, Jiaqi, Yan Zhuang und Simon Fong (2016). „The impact of data normalization on stock market prediction: Using SVM and technical indicators“. In: *International Conference on Soft Computing in Data Science*. Communications in Computer and Information Science (21 - 22 September 2016, Kuala Lumpur). Hrsg. von Michael W. Berry, Azlinah Hj. Mohamed und Bee Wah Yap. Bd. 652. Singapore: Springer Nature Singapore, S. 72–88.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot und E. Duchesnay (2011). „Scikit-learn: Machine learning in Python“. In: *Journal of Machine Learning Research* 12, S. 2825–2830.

Plevris, Vagelis, German Solorzano, Nikolaos P. Bakas und Mohamed El Amine Ben Seghier (2022). „Investigation of performance metrics in regression analysis and machine learning-based prediction models“. In: *The 8th European Congress on Computational Methods in Applied Sciences and Engineering (05 - 09 June 2022)*. ECCOMAS Congress 2022. Oslo.

Probst, Gilbert, Steffen Raub und Kai Romhardt (2012). *Wissen managen: Wie Unternehmen ihre wertvollste Ressource optimal nutzen*. Wiesbaden: Springer Gabler.

Python Software Foundation (2025). *Python*. URL: https://www.python.org (besucht am 02. 09. 2025).

Reis, Joe und Matt Housley (2023). *Handbuch data engineering: robuste Datensysteme planen und erstellen*. Übers. von Helena Schock. 1. Aufl. Heidelberg: O'Reilly.

Reitermanova, Zuzana (2010). „Data splitting“. In: *World Data Symposium*. Bd. 10. Matfyzpress Prague, S. 31–36.

Ribeiro, Vitor, Andre Rocha, Rui Peixoto, Filipe Portela und Manuel F. Santos (2017). „Importance of Statistics for Data Mining and Data Science“. In: *2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. FiCloud (21 - 23 August 2017). Prague: IEEE, S. 156–163.

Rockel, Tobias (2022). „Güteuntersuchung von Imputationsverfahren für unvollständige Datenmatrizen“. Diss. Universitätsverlag Ilmenau.

Runkler, Thomas A. (2015). *Data Mining: Modelle und Algorithmen intelligenter Datenanalyse*. Wiesbaden: Springer Fachmedien.

Rupp, Chris und Klaus Pohl (2021). *Basiswissen Requirements Engineering: Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level*. 5. Aufl. Heidelberg: dpunkt.verlag.

Sachs, Lothar (1993). *Statistische Methoden*. 7. Aufl. Berlin: Springer.

Schiefer, Hartmut und Felix Schiefer (2018). „Statistische Versuchsplanung, Design of Experiments (DoE)“. In: *Statistik für Ingenieure: Eine Einführung mit Beispielen aus der Praxis*. Hrsg. von Hartmut Schiefer und Felix Schiefer. Wiesbaden: Springer Fachmedien, S. 1–20. (Besucht am 06. 10. 2025).

Seu, Kimseth, Mi-Sun Kang und HwaMin Lee (2022). „An Intelligent Missing Data Imputation Techniques: A Review“. In: *JOIV : International Journal on Informatics Visualization* 6.1, S. 278–283.

Siebertz, Karl, David Van Bebber und Thomas Hochkirchen (2017). *Statistische Versuchsplanung*. Berlin: Springer Vieweg.

Siepmann, David und Norbert Graef (2016). „Industrie 4.0 - Grundlagen und Gesamtzusammenhang“. In: *Einführung und Umsetzung von Industrie 4.0*. Hrsg. von Armin Roth. Berlin: Springer Gabler, S. 17–82.

Singh, Nitin (2024). „Cyber-physical systems: a bibliometric analysis of literature“. In: *Journal of Intelligent Manufacturing* 36, S. 2335–2371.

Stekhoven, Daniel J. und Peter Bühlmann (2012). „MissForest—non-parametric missing value imputation for mixed-type data“. In: *Bioinformatics* 28.1, S. 112–118.

Stock, Daniel (2022). „Authentifizierungsverfahren auf Basis von Selbstbeschreibungsmerkmalen cyber-physischer Produktionssysteme“. Diss. Stuttgart: Universität Stuttgart.

Sun, Yige, Jing Li, Yifan Xu, Tingting Zhang und Xiaofeng Wang (2023). „Deep learning versus conventional methods for missing data imputation: A review and comparative study“. In: *Expert Systems with Applications* 227, Article 120201.

Taha, Abdel Aziz und Allan Hanbury (2015). „Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool“. In: *BMC Medical Imaging* 15.29.

Tan, Jimin, Jianan Yang, Sai Wu, Gang Chen und Jake Zhao (2021). *A critical look at the current train/test split in machine learning*.

Tao, Fei, Qinglin Qi, Ang Liu und Andrew Kusiak (2018). „Data-driven smart manufacturing“. In: *Journal of Manufacturing Systems* 48, S. 157–169.

The pandas development team (2025). *pandas-dev/pandas: Pandas*. Version v2.3.2. URL: https://doi.org/10.5281/zenodo.16918803.

Troyanskaya, Olga, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein und Russ B. Altman (2001). „Missing value estimation methods for DNA microarrays“. In: *Bioinformatics* 17.6, S. 520–525.

Van Buuren, Stef (2018). *Flexible Imputation of Missing Data*. 2. Aufl. New York: Chapman und Hall/CRC.

Weichert, Dorina, Patrick Link, Anke Stoll, Stefan Rüping, Steffen Ihlenfeldt und Stefan Wrobel (2019). „A review of machine learning for the optimization of production processes“. In: *The International Journal of Advanced Manufacturing Technology* 104.5, S. 1889–1902.

Wieczorek, Dorothee (2009). „Statistische versuchsplanung“. In: *„Das ist gar kein modell!“: Unterschiedliche modelle und modellierungen in betriebswirtschaftslehre und ingenieurwissenschaften*. Hrsg. von Gerhard Bandow und Hartmut H. Holzmüller. Wiesbaden: Gabler, S. 167–187.

Wirth, Rüdiger und Jochen Hipp (2000). „CRISP-DM: Towards a standard process model for data mining“. In: *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. Bd. 1. Manchester: AAAI Press, S. 29–39.

Zhang, Shichao, Chengqi Zhang und Qiang Yang (2003). „Data Preparation for Data Mining“. In: *Applied Artificial Intelligence* 17.5, S. 375–381.

Zheng, Alice und Amanda Casari (2019). *Merkmalskonstruktion für machine learning - Prinzipien und Techniken der Datenaufbereitung*. Übers. von Thomas Lotze. 1. Aufl. Heidelberg: O'Reilly.

A Anhang A: Vollständiger Experimentierplan

Auf den nächsten drei Seiten wird der Ablaufplan des Experimentierplans in voller Länge und vollem Detailgrad abgebildet.

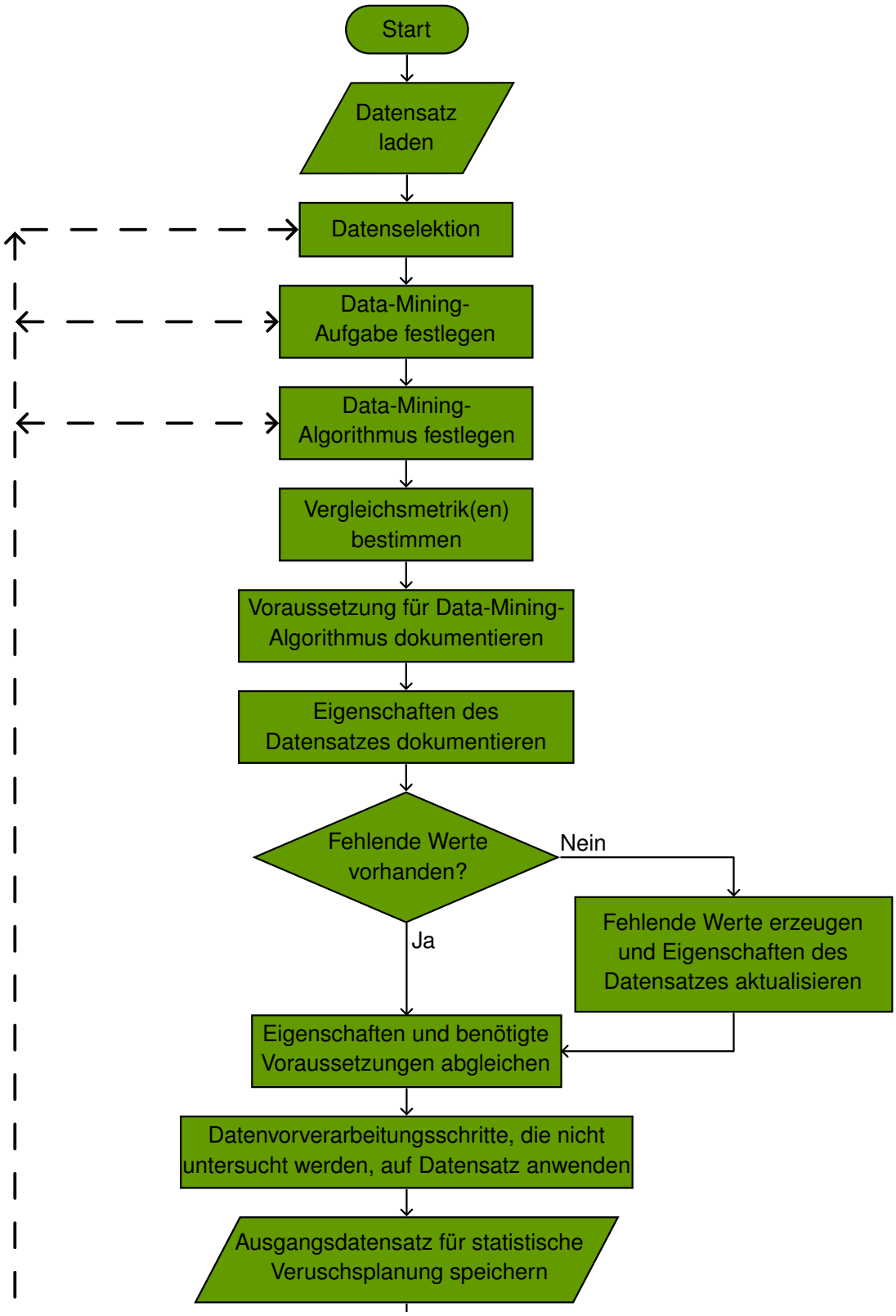

Abbildung A.1: Experimentierplan Seite 1

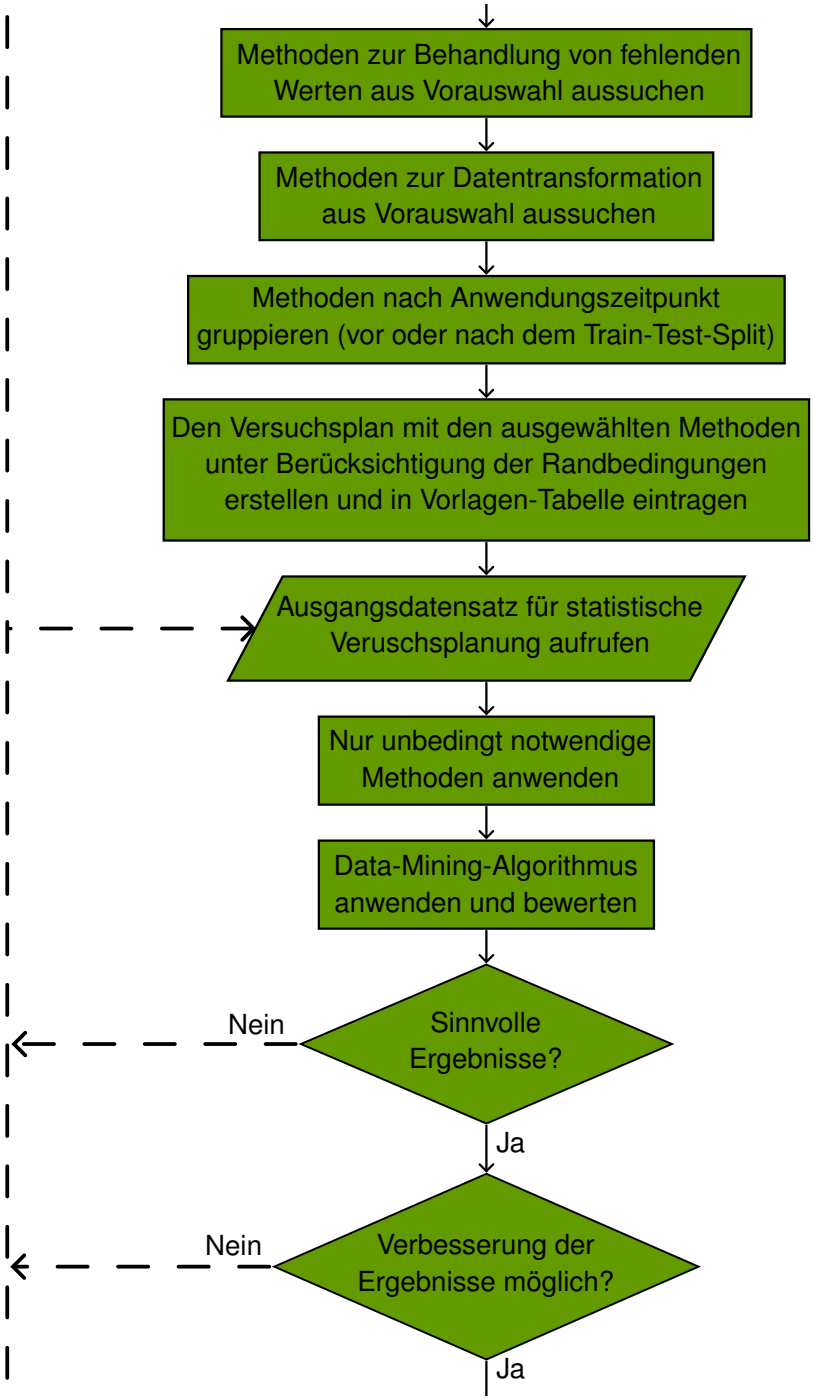

Abbildung A.2: Experimentierplan Seite 2

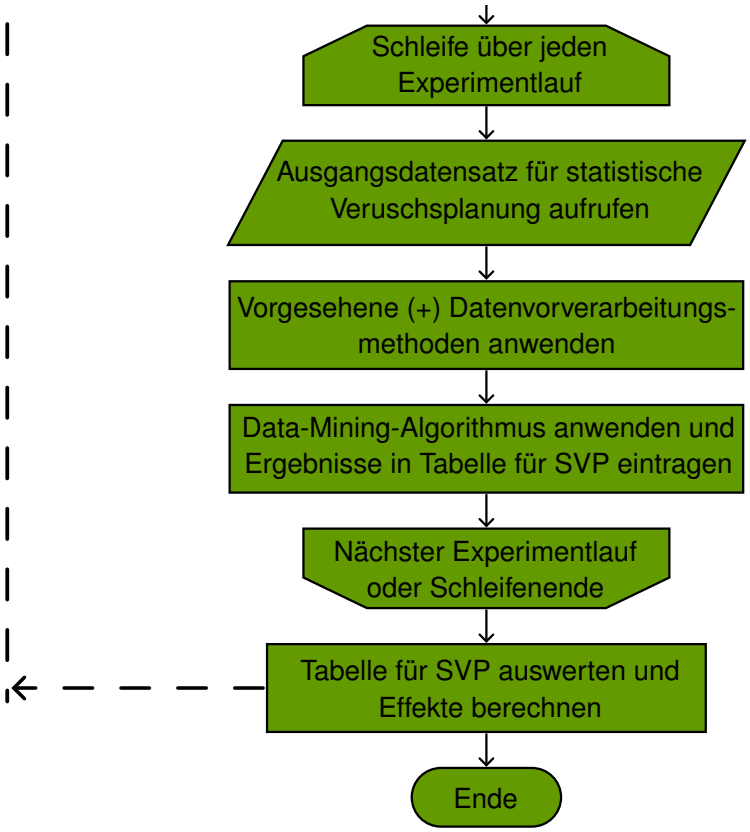

Abbildung A.3: Experimentierplan Seite 3

B Anhang B: Python Programmcode der exemplarischen Implementierung

B.1 Experiment.py

Algorithmus B.1: Experimentdurchlauf

```
1  import pandas as pd
2  from sklearn.ensemble import RandomForestRegressor
3  from sklearn.model_selection import train_test_split
4  from SVP import generiere_versuchsplan
5  from sklearn.preprocessing import StandardScaler, MinMaxScaler,
       RobustScaler, Normalizer, PolynomialFeatures
6  from sklearn.experimental import enable_iterative_imputer  #
       noqa
7  from sklearn.impute import SimpleImputer, KNNImputer,
       IterativeImputer
8  # from missforest import MissForest --> Funktiniert nicht. Ich
       habe Missforest in IterativeImputer mit RandomForestRegressor
        umgewandelt
9  import numpy as np
10 from sklearn.neighbors import KNeighborsRegressor
11 from sklearn.tree import DecisionTreeRegressor
12 from sklearn.metrics import root_mean_squared_error, r2_score,
       mean_absolute_percentage_error
13 from Methoden import *
14
15 def load_data(file_path):
16     try:
17         data = pd.read_csv(file_path)
18         return data
19     except Exception as e:
20         print(f"Error loading data: {e}")
21         return None
22
23 df = load_data("continuous_factory_process.csv")
```

```
24 #============================================
25 #benoetigte Datenvorverarbeitung
26 #============================================
27 #Daten Aufteilen
28 time = df.iloc[:, 0] #erste Spalte
29 factory_ambient = df.iloc[:, 1:3] #Spalten 1 und 2
30 variables = df.iloc[:, 1:42] #Spalten 1 bis 41
31 variable_second_stage_pre = df.iloc[:, 72:86] #Spalten 72 bis 85
32
33 measurements = df.iloc[:, 42:72]
34 target_indices = [42,44,46,48,50,52,54,56,58,60,62,64,66,68,70]
35 setpoint_indices =
      [43,45,47,49,51,53,55,57,59,61,63,65,67,69,71]
36
37 measurements_second_stage = df.iloc[:, 86:116]
38 target_indices_second_stage =
      [86,88,90,92,94,96,98,100,102,104,106,108,110,112,114]
39 setpoint_indices_second_stage =
      [87,89,91,93,95,97,99,101,103,105,107,109,111,113,115]
40
41 targets = df.iloc[:,target_indices]
42 setpoints = df.iloc[:,setpoint_indices]
43 targets_second_stage = df.iloc[:,target_indices_second_stage]
44 setpoints_second_stage = df.iloc[:,setpoint_indices_second_stage
      ]
45 variable_second_stage = pd.concat([factory_ambient,targets,
      variable_second_stage_pre], axis=1)
46
47 #Daten Aufteilen in Eingabe, Zielgroessen und Sollwerte fuer
      Aufgabe zwei
48 X_use = variable_second_stage
49 y_use = targets_second_stage
50 setpoints_use = setpoints_second_stage
51
52 #Ausgangsdatensatz fuer Experiment speichern
53 X_use.to_csv("CSV/X.csv", index=False)
54 y_use.to_csv("CSV/y.csv", index=False)
55 setpoints_use.to_csv("CSV/setpoints.csv", index=False)
56 time.to_csv("CSV/time.csv", index=False)
57
58 # Data Mining Funktion hier festlegen
59 knn_regr = KNeighborsRegressor()
60
```

```
61 #=============================================
62 #initialer Durchgang --> Durchgefuehrt in Experiment_notebook
63 #=============================================
64 #Werte ohne jegliche Vorverarbeitung:
65 # Mean R2 Score = 0.5754503273751702
66 # R2-Sum Score = 8.631754910627553
67
68 # RMSE-Summe: 7.737449205312799
69 # RMSE-Durchschnitt: 0.5158299470208533
70
71 #=============================================
72 #Funktionen Gruppieren
73 #=============================================
74 # Gruppiere Funktionen,wo der Input und Output gleich ist
75 function_groups = {
76     "FehlendeWerte": [fw_LoeschenDatentupel, fw_LoeschenMerkmale
          , fw_Interpolation],
77     "Imputation": [fw_MedianImputation, fw_MeanImputation,
          fw_ModeImputation, fw_KNNImputation,
          fw_RegressionImputation, fw_MICE, fw_MissForest,
          fw_GAINImputation],
78     "Skalierung": [dt_MinMaxSkalierung, dt_Normalisierung,
          dt_RobustScale, dt_ZScore],
79     "Aggregation": [dt_Summe, dt_Mittelwert],
80     "Datenglaettung": [dt_Glaettung_MovingAverage,
          dt_Glaettung_Exponential],
81     "Merkmalskonstruktion": [dt_Differenz_bilden]
82 }
83 function_groups_pre_split = [
84     *function_groups["Merkmalskonstruktion"],
85     *function_groups["Aggregation"],
86     *function_groups["Datenglaettung"],
87     *function_groups["FehlendeWerte"]
88 ]
89 function_groups_post_split = [
90     *function_groups["Skalierung"],
91     *function_groups["Imputation"]
92 ]
93
94 # Definiere die Gruppen von Faktoren und ihre Randbedingungen
95 fehlendeWerte = ["fw_MedianImputation", "fw_MeanImputation",
96                  "fw_LoeschenMerkmale", "fw_KNNImputation", "
                        fw_RegressionImputation",
```

```
97                  "fw_Interpolation", "fw_MICE", "fw_MissForest"]
98  #"fw_LoeschenDatentupel", entfernt, weil sonst nur 16
       Datenpunkte uebrig bleiben oder so
99
100 skalierung = ["dt_MinMaxSkalierung", "dt_Normalisierung", "
       dt_RobustScale", "dt_ZScore"]
101
102 aggregation = ["dt_Summe", "dt_Mittelwert"]
103
104 datenglaettung = ["dt_Glaettung_MovingAverage","
       dt_Glaettung_Exponential"]
105
106 other_factors = ["dt_Differenz_bilden"]
107
108 #=============================================
109 #Versuchsplan generieren oder laden
110 #=============================================
111 # SVP Generieren
112 svp = generiere_versuchsplan(fehlendeWerte, skalierung,
       aggregation, datenglaettung, other_factors)
113
114 # Ausgabe des Versuchsplans
115 print("Versuchsplan:")
116 print(svp)
117 # Speichere den Versuchsplan als CSV-Datei
118 svp.to_csv("Versuchsplan.csv", index=False)
119
120 #Wenn Versuchsplan mit den Faktoren und Gruppen schon vorhanden,
       dann nur noch laden
121 #svp = pd.read_csv("Versuchsplan.csv")
122
123 #=============================================
124 #definiere Funktion, um Data Mining Algorithmus zu bewerten
125 #=============================================
126 def data_mining_regression(train_data, train_targets, test_data,
       test_targets, model):
127     #Rueckgabe der Metriken:
128     #return r2mean, r2median_value, r2sum, mapemean, mapesum
129
130     r2sum = 0
131     r2mean = 0
132     r2_median = []
133     mapesum = 0
```

```
134     mapemean = 0
135     schleifendurchlaeufe = 0
136     df = pd.DataFrame() #leerer Dataframe fuer Ergebnisse
137     for col in train_targets.columns:
138         # Nur Zeilen mit gueltigen Zielwerten (ausserhalb einer
                Toleranz um 0) verwenden
139         tolerance = 1e-5  # Toleranzwert, kann angepasst werden
140         train_mask = (train_targets[col].abs() > tolerance) | (
                train_targets[col] < -tolerance)
141         test_mask = (test_targets[col].abs() > tolerance) | (
                test_targets[col] < -tolerance)
142
143         X_train_valid = train_data[train_mask]
144         y_train_valid = train_targets[col][train_mask]
145         X_test_valid = test_data[test_mask]
146         y_test_valid = test_targets[col][test_mask]
147
148         if len(y_train_valid) == 0 or len(y_test_valid) == 0:
149             print(f"Keine␣gueltigen␣Werte␣fuer␣Spalte␣{col}.")
150             continue
151
152         model.fit(X_train_valid, y_train_valid)
153         y_col_pred = model.predict(X_test_valid)
154
155         # # Testergebnisse und Predictions speichern - Fuer
                initialen Durchgang
156         # df_temp = pd.DataFrame({
157         #     f"{col}_true": y_test_valid.values,
158         #     f"{col}_pred": y_col_pred
159         # })
160         # df = pd.concat([df, df_temp], axis=1)
161
162         #score = model.score(X_test_valid, y_test_valid)
163         #rmse = root_mean_squared_error(y_test_valid, y_col_pred
                )
164         #print(f"Test_valid: {y_test_valid}")
165         #print(f"Predicted: {y_col_pred}")
166         mape = mean_absolute_percentage_error(y_test_valid,
                y_col_pred)
167         r2 = r2_score(y_test_valid, y_col_pred)
168         #print(f"R2 Score fuer Spalte {col} ist = {r2}")
169         #print(f"MAPE = {mape}| R2Score = {r2}")
170         mapesum += mape
```

```
171             r2sum += r2
172             r2_median.append(r2)
173             schleifendurchlaeufe += 1
174         r2mean = r2sum / schleifendurchlaeufe
175         mapemean = mapesum / schleifendurchlaeufe
176         r2median_value = np.median(r2_median)
177         print(f"Mean␣R2␣Score␣=␣{r2mean}")
178         print(f"Median␣R2␣Score␣=␣{r2median_value}")
179         print(f"Sum␣of␣R2␣Score␣=␣{r2sum}")
180         print(f"Mean␣MAPE␣=␣{mapemean}")
181         print(f"Sum␣of␣MAPE␣=␣{mapesum}")
182         #df.to_csv("Predictions_initialer_durchgang.csv", index=
                False) # Fuer Initialen durchgang gemacht
183
184         return r2mean, r2median_value, r2sum, mapemean, mapesum
185
186 #SVP Ergebnisspalten anhaengen
187 svp_with_results = svp.copy()
188 svp_with_results['r2_mean'] = None  # Neue Spalte fuer
        Ergebnisse
189 svp_with_results['r2_sum'] = None  # Neue Spalte fuer Ergebnisse
190 svp_with_results['r2_median'] = None  # Neue Spalte fuer
        Ergebnisse
191 svp_with_results['mape_mean'] = None  # Neue Spalte fuer
        Ergebnisse
192 svp_with_results['mape_sum'] = None  # Neue Spalte fuer
        Ergebnisse
193 #....
194
195 #=============================================
196 #Experiment mit Faktoren aus SVP durchfuehren
197 #=============================================
198
199 #Damit nicht alles auf einmal gemacht werden muss, nur bestimmte
        zeilen
200 start_zeile = 0
201 end_zeile = 100 #oder svp.shape[0] fuer alle
202 # Verarbeite jede Zeile im Versuchsplan
203 for index, row in svp.iterrows():
204     if index < start_zeile or index >= end_zeile:
205         continue  #ueberspringe Zeilen ausserhalb des
                gewuenschten Bereichs
206     print(f"Verarbeitung␣der␣Zeile␣{index␣+␣1}:")
```

```
207     #Aktive Funktionen die vor und nach dem Split aktiv sind
208     active_functions_pre_split = []
209     active_functions_post_split = []
210     for column_name, value in row.items():
211         if value == 1:  # Wenn Methode aktiv ist
212             function_to_call = globals().get(column_name)
213             # Pruefe, ob die Funktion in der jeweiligen Gruppe
                    ist (Vergleich ueber Namen)
214             if function_to_call and any(f.__name__ ==
                    column_name for f in function_groups_pre_split):
215                 active_functions_pre_split.append(
                        function_to_call)
216             elif function_to_call and any(f.__name__ ==
                    column_name for f in function_groups_post_split):
217                 active_functions_post_split.append(
                        function_to_call)
218     print(active_functions_pre_split)
219     print(active_functions_post_split)
220     #Ausgangdaten aufrufen und kopieren
221     X = pd.read_csv("CSV/X.csv")
222     y = pd.read_csv("CSV/y.csv")
223     setpoints = pd.read_csv("CSV/setpoints.csv")
224
225     #Methoden vor Train_test split anwenden
226     for func in active_functions_pre_split:
227         if func in function_groups["FehlendeWerte"]:
228             X = func(X, fehlende_werte=0)
229             print(f"FehlendeWertefunktion␣{func.__name__}␣wurde␣
                    ausgefuehrt.")
230         elif func in function_groups["Merkmalskonstruktion"]:
231             y = func(y,setpoints) #kann hier nur die
                    Differenzbildung der Zielvariablen sein
232             print(f"Merkmalskonstruktionsfunktion␣{func.__name__
                    }␣wurde␣ausgefuehrt.")
233         elif func in function_groups["Aggregation"]:
234             X = func(X,interval_anzahl=2800) #Intervalanzahl
                    kann gewaehlt werden
235             y = func(y,interval_anzahl=2800) #Muss dann
                    natuerlich auch auf y angewendet werden
236             setpoints = func(setpoints,interval_anzahl=500) #und
                     fuer setpoints falls es zur Merkmalskonstruktion
                     kommt
```

```
237             print(f"Aggregationsfunktion␣{func.__name__}␣wurde␣
                    ausgefuehrt.")
238         elif func in function_groups["Datenglaettung"]:
239             X = func(X)
240             print(f"Datenglaettungsfunktion␣{func.__name__}␣
                    wurde␣ausgefuehrt.")
241
242
243     #Train Test Split
244     X_train, X_test, y_train, y_test = train_test_split(X, y,
            test_size=0.25, random_state=42)
245
246     #Methoden nach Train_test split anwenden
247     for func in active_functions_post_split:
248         if func in function_groups["Imputation"]:
249             X_train, X_test = func(X_train, X_test,
                    fehlende_werte=0) #Form der fehlenden Werte ist 0
250             print(f"Imputationsfunktion␣{func.__name__}␣wurde␣
                    ausgefuehrt.")
251         elif func in function_groups["Skalierung"]:
252             X_train, X_test = func(X_train, X_test)
253             print(f"Skalierungsfunktion␣{func.__name__}␣wurde␣
                    ausgefuehrt.")
254
255     #Nach der ganzen Datenvorverarbeitung, Data Mining aufrufen
256     r2mean, r2median, r2sum, mapemean, mapesum =
            data_mining_regression(X_train, y_train, X_test, y_test,
            knn_regr)
257     svp_with_results.at[index, 'r2_mean'] = r2mean
258     svp_with_results.at[index, 'r2_median'] = r2median
259     svp_with_results.at[index, 'r2_sum'] = r2sum
260     svp_with_results.at[index, 'mape_mean'] = mapemean
261     svp_with_results.at[index, 'mape_sum'] = mapesum
262
263 svp_with_results.to_csv(f"Ergebnisse/2
        _Versuchsplan_mit_Ergebnissen{start_zeile}_{end_zeile}.csv",
        index=False)
264 print("Experiment␣abgeschlossen.␣Ergebnisse␣gespeichert.")
```

B.2 Methoden.py

Algorithmus B.2: Implementierung der Datenvorverarbeitungsmethoden als Funktionen

```
1  import numpy as np
2  import pandas as pd
3  from sklearn.ensemble import RandomForestRegressor
4  from sklearn.experimental import enable_iterative_imputer  #
       Notwendig fuer IterativeImputer
5  from sklearn.impute import IterativeImputer, KNNImputer,
       SimpleImputer
6  from sklearn.preprocessing import StandardScaler, MinMaxScaler,
       Normalizer, PolynomialFeatures, RobustScaler
7
8
9  #Fehlende Werte - loeschen oder interpolation
10 def fw_LoeschenMerkmale(data_local,fehlende_werte = np.nan): #
       Spalten mit mehr als 20% fehlenden Werten loeschen
11     for col in data_local.columns:
12         # ueberpruefen auf "fehlende Werte", damit ich auch
               fehlende_werte =0 beruecksichtigen kann
13         if (data_local[col] == fehlende_werte).sum() / len(
               data_local) > 0.2:
14             data_local = data_local.drop(columns=[col])
15     return data_local
16
17 def fw_Interpolation(data_local,fehlende_werte= np.nan):
18     if fehlende_werte != np.nan:
19         data_local = data_local.replace(fehlende_werte, np.nan)
20
21     interpolated_data = data_local.interpolate(method='linear',
           limit_direction='both')
22
23     return interpolated_data
24
25 #Fehlende Werte - Imputation
26 def fw_MedianImputation(data_local_train, data_local_test,
       fehlende_werte= np.nan):
27     imputer = SimpleImputer(strategy='median', missing_values=
           fehlende_werte)
28     imputed_data_train = imputer.fit_transform(data_local_train)
29     imputed_data_test = imputer.transform(data_local_test)
30     return imputed_data_train, imputed_data_test
```

```
31
32 def fw_MeanImputation(data_local_train, data_local_test,
       fehlende_werte= np.nan):
33     imputer = SimpleImputer(strategy='mean', missing_values=
           fehlende_werte)
34     imputed_data_train = imputer.fit_transform(data_local_train)
35     imputed_data_test = imputer.transform(data_local_test)
36     return imputed_data_train, imputed_data_test
37
38 def fw_ModeImputation(data_local_train, data_local_test,
       fehlende_werte= np.nan):
39     imputer = SimpleImputer(strategy='most_frequent',
           missing_values=fehlende_werte)
40     imputed_data_train = imputer.fit_transform(data_local_train)
41     imputed_data_test = imputer.transform(data_local_test)
42     return imputed_data_train, imputed_data_test
43
44
45 def fw_KNNImputation(data_local_train, data_local_test,
       fehlende_werte= np.nan): #neighbors =5 festgelegt
46     imputer = KNNImputer(n_neighbors=5, missing_values=
           fehlende_werte)
47     imputed_data_train = imputer.fit_transform(data_local_train)
48     imputed_data_test = imputer.transform(data_local_test)
49     return imputed_data_train, imputed_data_test
50
51
52 def fw_RegressionImputation(train_data, test_data,
       fehlende_werte=np.nan):
53     train = train_data.copy()
54     test = test_data.copy()
55     regr = RandomForestRegressor(n_estimators=25, random_state
           =0)
56     for col in train.columns:
57         # Nur wenn im Training fehlende Werte vorhanden sind
58         if (train[col] == fehlende_werte).any():
59             train_valid = train[train[col] != fehlende_werte]
60             train_missing = train[train[col] == fehlende_werte]
61             if not train_missing.empty:
62                 regr.fit(train_valid.drop(columns=[col]),
                       train_valid[col])
63                 predicted_train = regr.predict(train_missing.
                       drop(columns=[col]))
```

```
64                  train.loc[train[col] == fehlende_werte, col] =
                        predicted_train
65          # Imputation auf Testdaten
66          if (test[col] == fehlende_werte).any():
67              test_missing = test[test[col] == fehlende_werte]
68              if not test_missing.empty:
69                  # Trainiere auf allen gueltigen Trainingsdaten
70                  train_valid = train[train[col] != fehlende_werte
                        ]
71                  regr.fit(train_valid.drop(columns=[col]),
                        train_valid[col])
72                  predicted_test = regr.predict(test_missing.drop(
                        columns=[col]))
73                  test.loc[test[col] == fehlende_werte, col] =
                        predicted_test
74      return train, test
75
76  def fw_MICE(data_local_train, data_local_test, fehlende_werte=
        np.nan):
77      imputer = IterativeImputer(max_iter=10, random_state=0,
            missing_values=fehlende_werte, initial_strategy='mean') #
            Hier auch? MaXIter und tol anpassen?
78      imputed_data_train = imputer.fit_transform(data_local_train)
79      imputed_data_test = imputer.transform(data_local_test)
80      return imputed_data_train, imputed_data_test
81
82  def fw_MissForest(data_local_train, data_local_test,
        fehlende_werte= np.nan):
83      rf = RandomForestRegressor(n_estimators=10,
84              max_depth=10,
85              bootstrap=True,
86              max_samples=0.5,
87              n_jobs=2,
88              random_state=0) #fuer MissForest
                    RandomForestRegressor #Werte aus sklearn-Doku
89      imputer = IterativeImputer(estimator=rf, max_iter=3, tol=1e
            -2, verbose=2, random_state=0, missing_values=
            fehlende_werte, initial_strategy='mean') #MaxIter =3,
            geht hier nicht um den besten Missforest. Probleme mit
            convergence online bekannt
90      imputed_data_train = imputer.fit_transform(data_local_train)
91      imputed_data_test = imputer.transform(data_local_test)
92      return imputed_data_train, imputed_data_test
```

```
93
94
95  #Skalierung
96  def dt_ZScore(train,test):
97      scaler = StandardScaler()
98      scaled_train = scaler.fit_transform(train)
99      scaled_test = scaler.transform(test)
100     return scaled_train, scaled_test
101
102 def dt_Normalisierung(train,test):
103     scaler = Normalizer()
104     scaled_train = scaler.fit_transform(train)
105     scaled_test = scaler.transform(test)
106     return scaled_train, scaled_test
107
108 def dt_MinMaxSkalierung(train,test):
109     scaler = MinMaxScaler()
110     scaled_train = scaler.fit_transform(train)
111     scaled_test = scaler.transform(test)
112     return scaled_train, scaled_test
113
114 def dt_RobustScale(train,test):
115     scaler = RobustScaler()
116     scaled_train = scaler.fit_transform(train)
117     scaled_test = scaler.transform(test)
118     return scaled_train, scaled_test
119
120 #Aggregation
121 def dt_Summe(data_local, interval_anzahl=1):
122     if interval_anzahl <= len(data_local):
123         # Berechne die Groesse jedes Intervalls basierend auf
              der Anzahl der Zeilen
124         total_rows = len(data_local)
125         interval_size = total_rows // interval_anzahl
126
127         # Erstelle eine Gruppierungsvariable basierend auf dem
              Index
128         data_local['interval'] = (data_local.index //
              interval_size).astype(int)
129
130         # Gruppiere nach dem neuen Interval und aggregiere mit
              Summe
```

```
131         result = data_local.groupby('interval').sum().
                reset_index()
132         result = result.drop(columns=['interval'])  # Entferne
                die Hilfsspalte 'interval'
133
134         return result
135     else:
136         # Wenn die Anzahl der Intervalle groesser ist als die
                Anzahl der Zeilen, gebe die Originaldaten zurueck
137         return data_local
138
139 def dt_Mittelwert(data_local, interval_anzahl=1):
140     if interval_anzahl <= len(data_local):
141         # Berechne die Groesse jedes Intervalls basierend auf
                der Anzahl der Zeilen
142         total_rows = len(data_local)
143         interval_size = total_rows // interval_anzahl
144
145         # Erstelle eine Gruppierungsvariable basierend auf dem
                Index
146         data_local['interval'] = (data_local.index //
                interval_size).astype(int)
147
148         # Gruppiere nach dem neuen Interval und bilde den
                Durchschnitt
149         result = data_local.groupby('interval').mean().
                reset_index()
150         result = result.drop(columns=['interval'])  # Entferne
                die Hilfsspalte 'interval'
151
152         return result
153     else:
154         return data_local
155
156 #Merkmalskonstruktion
157 def dt_Differenz_bilden(actual_local,setpoints_local):
158     # Benoetigt die Zielgroessen und die Sollwerte
159     #Bestimmt die Differenz zwischen allen eintraegen im
            Dataframe
160     #bei beiden Df Indizes und Spaltennamen angleichen
161     actual_local.columns = [f"Abweichung␣{i+1}" for i in range(
            len(actual_local.columns))]
```

```
162     setpoints_local.columns = [f"Abweichung␣{i+1}" for i in
            range(len(setpoints_local.columns))]
163     #Gibt einen Dataframe mit den Differenzen
164     differenz = actual_local - setpoints_local
165     return differenz
166
167 def dt_PolynomialFunktion(features): #Benoetigt Zwei Merkmale
168     poly = PolynomialFeatures(degree=2, include_bias=False) #
169     poly_features = poly.fit_transform(features)
170     return poly_features
171
172 #Datenglaettung
173 def dt_Glaettung_MovingAverage(data_local): #Fenstergroesse
        festgelegt auf 5
174     smoothed_data = data_local.rolling(window=5, min_periods=1,
            center=True).mean()
175     return smoothed_data
176
177 def dt_Glaettung_Exponential(data_local): #Fenstergroesse auch
        auf 5, Adjust = True damit Startwerte nicht so stark
        gewichtet werden
178     smoothed_data = data_local.ewm(span=5, adjust=True).mean()
179     return smoothed_data
```

B.3 SVP.py

Algorithmus B.3: Funktion zur Erstellung des statistischen Versuchsplans

```
1  import pandas as pd
2  from itertools import product
3
4  def generiere_versuchsplan(fehlendeWerte, skalierung,
       aggregation, datenglaettung, other_factors):
5
6      # Alle Faktoren zusammen
7      all_factors = fehlendeWerte + skalierung + aggregation +
           datenglaettung + other_factors
8
9      # Erzeuge alle moeglichen Kombinationen (0/1) fuer alle
           Faktoren
10     alle_kombinationen = list(product([0, 1], repeat=len(
           all_factors)))
11     df = pd.DataFrame(alle_kombinationen, columns=all_factors)
12
13     # Filterfunktion: Nur Zeilen behalten, in denen pro Gruppe
           hoechstens ein Faktor aktiv ist
14     def nur_einer_aktiv(row, gruppe):
15         return row[gruppe].sum() <= 1
16
17     # Filter anwenden
18     mask = df.apply(lambda row:
19         nur_einer_aktiv(row, fehlendeWerte) and
20         nur_einer_aktiv(row, skalierung) and
21         nur_einer_aktiv(row, aggregation) and
22         nur_einer_aktiv(row, datenglaettung),
23         axis=1
24     )
25     df_filtered = df[mask].reset_index(drop=True)
26     return df_filtered
```

B.4 Auswertung.ipynb

Algorithmus B.4: Effektberechnung

```
from itertools import combinations
import pandas as pd

svp_results = pd.read_csv("Ergebnisse/2
    _Versuchsplan_mit_Ergebnissen0_1000.csv")

svp_results.drop(columns=["r2_sum","mape_mean", "mape_sum"],
    axis=1, inplace=True)

# Kopiere das DataFrame, falls du das Original behalten willst
svp_results_ww = svp_results.copy()

# Bestimme die Faktor-Spalten (alle ausser Ergebnisspalten)
ergebnis_spalten = ["r2_mean", "r2_median"]  # ggf. anpassen
faktor_spalten = [col for col in svp_results_ww.columns if col
    not in ergebnis_spalten]

# Wechselwirkungen berechnen und als neue Spalten anhaengen
for a, b in combinations(faktor_spalten, 2):
    # 1, wenn beide Werte gleich sind (also beide 0 oder beide
        1), sonst 0
    svp_results_ww[f"{a}*{b}"] = (svp_results_ww[a] ==
        svp_results_ww[b]).astype(int)

# Nur die Spalten mit Wechselwirkungen und die Ergebnis-Spalten
    behalten
svp_ww_only = svp_results_ww[ergebnis_spalten + [col for col in
    svp_results_ww.columns if '*' in col]]
#Effektberechnung fuer r2_mean

def berechne_effekte(df, zielspalte="r2_mean"):
    ergebnisse = {}
    for spalte in df.columns:
        if spalte == zielspalte:
            continue
        # Mittelwert fuer r2_mean, wenn Spalte == 1
        mean_1 = df[df[spalte] == 1][zielspalte].mean()
        # Mittelwert fuer r2_mean, wenn Spalte == 0
        mean_0 = df[df[spalte] == 0][zielspalte].mean()
```

```
33          ergebnisse[spalte] = {"MW+": mean_1, "MW-": mean_0, "
                Effekt": mean_1 - mean_0}
34      return ergebnisse
35
36  # Beispielaufruf:
37  effekte_r2_mean = berechne_effekte(svp_results, zielspalte="
        r2_mean")
38  print(effekte_r2_mean)
39  effekte_r2_mean_df = pd.DataFrame(effekte_r2_mean).T
40  #effekte_r2_mean_df.to_csv("Effekte/Effekte_r2_mean.csv")
41
42  wechselwirkungen_r2_mean = berechne_effekte(svp_ww_only,
        zielspalte="r2_mean")
43  print(wechselwirkungen_r2_mean)
44  wechselwirkungen_r2_mean_df = pd.DataFrame(
        wechselwirkungen_r2_mean).T
45  #wechselwirkungen_r2_mean_df.to_csv("Effekte/
        Wechselwirkungen_r2_mean.csv")
46
47  # Effektberechnung fuer r2_median
48  effekte_r2_median = berechne_effekte(svp_results, zielspalte="
        r2_median")
49  print(effekte_r2_median)
50  effekte_r2_median_df = pd.DataFrame(effekte_r2_median).T
51
52  wechselwirkungen_r2_median = berechne_effekte(svp_ww_only,
        zielspalte="r2_median")
53  print(wechselwirkungen_r2_median)
54  wechselwirkungen_r2_median_df = pd.DataFrame(
        wechselwirkungen_r2_median).T
55
56  #Zusammenfuehren der Ergebnisse
57  effekte_df = pd.concat([effekte_r2_mean_df.add_suffix('_r2_mean'
        ), effekte_r2_median_df.add_suffix('_r2_median')], axis=1)
58  wechselwirkungen_df = pd.concat([wechselwirkungen_r2_mean_df.
        add_suffix('_r2_mean'), wechselwirkungen_r2_median_df.
        add_suffix('_r2_median')], axis=1)
59  # Speichern der Ergebnisse in CSV-Dateien
60  effekte_df.to_csv("Ergebnisse/Effekte.csv")
61  wechselwirkungen_df.to_csv("Ergebnisse/Wechselwirkungen.csv")
```

Algorithmus B.5: Haupteffektdiagramme erstellen

```
1  import matplotlib.pyplot as plt
2
3  # Daten vorbereiten: Mittelwerte fuer negative (-) und positive
       (+) Auspraegungen
4  faktoren = ['Interpolation', 'Merkmale␣Loeschen', 'KNN-
       Imputation', 'Median-Imputation']  # Faktoren
5  mittelwerte_plus = [0.6579035986945996, 0.6321904957893713,
       0.6138621827733832, 0.5971014616285214]  # Mittelwerte bei
       negativer Auspraegung (-)
6  mittelwerte_minus = [0.6190396584698891, 0.6222537963330426,
       0.6245448354600411, 0.6266399256031489]   # Mittelwerte bei
       positiver Auspraegung (+)
7
8  abstand = 3*(max(mittelwerte_plus) - min(mittelwerte_plus))  #
       Halber Abstand zwischen den Punkten
9  abstand_unten = 0.1*(max(mittelwerte_plus) - min(
       mittelwerte_plus))  # Abstand von plus und minus zu den
       Beschriftungen
10 farbe2 = "#639A00"
11 farbe1 = "#CA7406"
12
13 # Plot erstellen
14 plt.figure(figsize=(6.2, 2.5))
15 for i in range(len(faktoren)):
16     plt.plot([i - abstand, i + abstand], [mittelwerte_minus[i],
          mittelwerte_plus[i]], 'k-', zorder=1)  # Linie zeichnen
17     plt.scatter(i - abstand, mittelwerte_minus[i], color=farbe1,
           zorder=2)# Punkt fuer "-"
18     plt.scatter(i + abstand, mittelwerte_plus[i], color=farbe2,
          zorder=2)# Punkt fuer "+"
19
20 # Achsenbeschriftungen und Titel
21 plt.xticks(range(len(faktoren)), faktoren)
22 plt.ylabel('R2-Wert')
23 plt.title('Behandlung␣von␣fehlenden␣Werten')
24
25 # Legende fuer "+" und "-"
26 plt.scatter([], [], color=farbe1, label='Mittelwerte␣-', zorder
       =2)
27 plt.scatter([], [], color=farbe2, label='Mittelwerte␣+', zorder
       =2)
28 plt.legend(loc='upper␣right')
```

```
29
30 # Horizontale Linie zur Orientierung
31 plt.axhline(y=sum(mittelwerte_minus + mittelwerte_plus)/len(
      mittelwerte_minus + mittelwerte_plus), color="gray",
      linestyle="dotted")
32
33 # Diagramm anzeigen
34 plt.tight_layout()
35 plt.savefig("Plots/Behandlung_FW_Haupteffektdiagramm.pdf",
      format='pdf')
36 plt.show()
37
38 ###Zweiter Plot - Skalierung
39 # Daten vorbereiten: Mittelwerte fuer negative (-) und positive
      (+) Auspraegungen
40 faktoren = ['Min-Max-Normalisierung', 'Z-Score', 'Normalisierung
      ', 'RobustScaler']  # Faktoren
41 mittelwerte_plus = [0.6509410350260595, 0.6448165050245752,
      0.6237988749925146, 0.5721361412578939]  # Mittelwerte bei
      negativer Auspraegung (-)
42 mittelwerte_minus = [0.6164620838065007,0.6179932163068719,
      0.6232476238148869, 0.636163307248542]   # Mittelwerte bei
      positiver Auspraegung (+)
43
44 # Plot erstellen
45 plt.figure(figsize=(6.2, 2.5))
46 for i in range(len(faktoren)):
47     plt.plot([i - abstand, i + abstand], [mittelwerte_minus[i],
         mittelwerte_plus[i]], 'k-', zorder=1)  # Linie zeichnen
48     plt.scatter(i - abstand, mittelwerte_minus[i], color=farbe1,
          zorder=2)# Punkt fuer "-"
49     plt.scatter(i + abstand, mittelwerte_plus[i], color=farbe2,
         zorder=2)# Punkt fuer "+"
50
51 # Achsenbeschriftungen und Titel
52 plt.xticks(range(len(faktoren)), faktoren)
53 plt.ylabel('R2-Wert')
54 plt.title('Datentransformation␣-␣Skalierung')
55
56 # Legende fuer "+" und "-"
57 plt.scatter([], [], color=farbe1, label='Mittelwerte␣-', zorder
      =2)
58 plt.scatter([], [], color=farbe2, label='Mittelwerte␣+', zorder
```

```
    =2)
59 plt.legend(loc='lower left')
60
61 # Horizontale Linie zur Orientierung
62 plt.axhline(y=sum(mittelwerte_minus + mittelwerte_plus)/len(
    mittelwerte_minus + mittelwerte_plus), color="gray",
    linestyle="dotted")
63
64 # Diagramm anzeigen
65 plt.tight_layout()
66 plt.savefig("Plots/Behandlung_DTS_Haupteffektdiagramm.pdf",
    format='pdf')
67 plt.show()
68
69 ###Dritter Plot - Datentransformation Rest
70 # Daten vorbereiten: Mittelwerte fuer negative (-) und positive
    (+) Auspraegungen
71 faktoren = ['Ist-Soll-Differenzbildung', 'Mittelwertsbildung', '
    Moving-Average Glaettung', 'Exponentielle Glaettung', '
    Summenbildung']  # Faktoren
72 mittelwerte_plus = [0.6879710169033486,
    0.668023572413199,0.6408410546989187, 0.6335475694999463,
    0.600706349305059]  # Mittelwerte bei negativer Auspraegung
    (-)
73 mittelwerte_minus = [0.5587447311974764,0.6010250248690192,
    0.6146162837261594, 0.6182630263256457, 0.6346836364230892]
      # Mittelwerte bei positiver Auspraegung (+)
74
75 # Plot erstellen
76 plt.figure(figsize=(6.2, 2.5))
77 for i in range(len(faktoren)):
78     plt.plot([i - abstand, i + abstand], [mittelwerte_minus[i],
        mittelwerte_plus[i]], 'k-', zorder=1)  # Linie zeichnen
79     plt.scatter(i - abstand, mittelwerte_minus[i], color=farbe1,
        zorder=2)# Punkt fuer "-"
80     plt.scatter(i + abstand, mittelwerte_plus[i], color=farbe2,
        zorder=2)# Punkt fuer "+"
81
82 # Achsenbeschriftungen und Titel
83 plt.xticks(range(len(faktoren)), faktoren)
84 ax = plt.gca()
85 for label in ax.get_xticklabels():
86     if label.get_text() == 'Mittelwertsbildung' or label.
```

```
            get_text() == 'Exponentielle Glaettung':
87              label.set_y(-0.15)  # weiter nach unten
88
89
90  plt.ylabel('R2-Wert')
91  plt.title('Datentransformation - Aggregation, Datenglaettung \n
        und Merkmalskonstruktion')
92
93  # Legende fuer "+" und "-"
94  plt.scatter([], [], color=farbe1, label='Mittelwerte -', zorder
        =2)
95  plt.scatter([], [], color=farbe2, label='Mittelwerte +', zorder
        =2)
96  plt.legend(loc='upper right')
97
98  # Horizontale Linie zur Orientierung
99  plt.axhline(y=sum(mittelwerte_minus + mittelwerte_plus)/len(
        mittelwerte_minus + mittelwerte_plus), color="gray",
        linestyle="dotted")
100
101 # Diagramm anzeigen
102 plt.tight_layout()
103 plt.savefig("Plots/Behandlung_DTR_Haupteffektdiagramm.pdf",
        format='pdf')
104 plt.show()
```

Algorithmus B.6: Haupteffekte Balkendiagramme

```
1  import matplotlib.pyplot as plt
2  import pandas as pd
3
4  # Effekte aufrufen von CSV - Namen geaendert
5  effekte_r2_mean_df = pd.read_csv("Effekte/Effekte_r2_mean.csv")
6
7  # Effekte sortieren (optional, z.B. nach Betrag des Effekts)
8  effekte_sorted = effekte_r2_mean_df.dropna().sort_values("Effekt
       ", ascending=False)
9
10 plt.figure(figsize=(6.2, 5))
11 # Schriftart und -groesse festlegen
12 plt.rcParams["pdf.fonttype"] = 42  # TrueType fuer LaTeX-
       Kompatibilitaet
13 plt.rcParams["font.family"] = "Helvetica"
14 plt.rcParams["font.size"] = 11
15 plt.bar(effekte_sorted["Faktor"], effekte_sorted["Effekt"],
       color ="#639A00")
16 plt.xticks(rotation=90)
17 plt.ylabel("Effektgroesse␣fuer␣R2-Durchschnitt")
18 plt.title("Datenvorverarbeitungsmethode")
19 plt.tight_layout()
20 plt.savefig("Plots/effekte_r2_mean.pdf", format='pdf')
21 plt.show()
22 #Aufgabe die Balken nach den Funktionsgruppen einfaerben
23
24 wechselwirkungen_r2_mean_df = wechselwirkungen_r2_mean_df.dropna
       ().sort_values("Effekt", ascending=False)
25 plt.figure(figsize=(6.2, 3))
26 # Schriftart und -groesse festlegen
27 plt.rcParams["pdf.fonttype"] = 42  # TrueType fuer LaTeX-
       Kompatibilitaet
28 plt.rcParams["font.family"] = "Helvetica"
29 plt.rcParams["font.size"] = 11
30 plt.bar(wechselwirkungen_r2_mean_df.index,
       wechselwirkungen_r2_mean_df["Effekt"], color ="#639A00",
       width=1) #Edgecolor="black" ist noch moeglich
31 plt.xticks([])  # <-- X-Achse ohne Beschriftung
32 plt.ylabel("Wechselwirkung␣auf␣R2-Durchschnitt")
33 plt.title("Wechselwirkungseffekte␣der␣
       Datenvorverarbeitungsmethoden")
34 plt.tight_layout()
```

```
35 plt.savefig("Plots/wechselwirkungen_r2_mean.pdf", format='pdf')
36 plt.show()
```

C Anhang C: Zusatzmaterial aus der Anwendung des Experimentierplans

C.1 Anzahl und Prozentangabe der fehlenden Werte aller Merkmale und Zielvariablen

In diesem Abschnitt des Anhangs wird die Anzahl der fehlenden Werte in zwei Tabellen angegeben. In Tabelle C.1 stehen die Anzahl und Prozentangabe der fehlenden Werte pro Merkmal und in Tabelle C.2 pro Zielvariable.

Tabelle C.1: Anzahl und Prozentangabe der fehlenden Werte pro Merkmal

Merkmalsbezeichung	Anzahl fehlender Werte	Angabe in %
AmbientConditions.AmbientHumidity.U.Actual	0	0.00
AmbientConditions.AmbientTemperature.U.Actual	0	0.00
Stage1.Output.Measurement0.U.Actual	70	0.50
Stage1.Output.Measurement1.U.Actual	5900	41.88
Stage1.Output.Measurement2.U.Actual	85	0.60
Stage1.Output.Measurement3.U.Actual	135	0.96
Stage1.Output.Measurement4.U.Actual	174	1.24
Stage1.Output.Measurement5.U.Actual	13400	95.12
Stage1.Output.Measurement6.U.Actual	4703	33.38
Stage1.Output.Measurement7.U.Actual	8761	62.19
Stage1.Output.Measurement8.U.Actual	777	5.52
Stage1.Output.Measurement9.U.Actual	722	5.12
Stage1.Output.Measurement10.U.Actual	268	1.90
Stage1.Output.Measurement11.U.Actual	10462	74.26
Stage1.Output.Measurement12.U.Actual	3188	22.63
Stage1.Output.Measurement13.U.Actual	341	2.42
Stage1.Output.Measurement14.U.Actual	5005	35.53
Machine4.Temperature1.C.Actual	0	0.00
Machine4.Temperature2.C.Actual	0	0.00
Machine4.Pressure.C.Actual	0	0.00
Machine4.Temperature3.C.Actual	0	0.00
Machine4.Temperature4.C.Actual	0	0.00
Machine4.Temperature5.C.Actual	0	0.00
Machine4.ExitTemperature.U.Actual	0	0.00
Machine5.Temperature1.C.Actual	0	0.00
Machine5.Temperature2.C.Actual	0	0.00
Machine5.Temperature3.C.Actual	0	0.00
Machine5.Temperature4.C.Actual	0	0.00
Machine5.Temperature5.C.Actual	0	0.00
Machine5.Temperature6.C.Actual	0	0.00
Machine5.ExitTemperature.U.Actual	0	0.00

Tabelle C.2: Anzahl und Prozentangabe der fehlenden Werte pro Zielvariable

Bezeichnung Zielvariable	Anzahl fehlender Werte	Angabe in %
Stage2.Output.Measurement0.U.Actual	1224	8.69
Stage2.Output.Measurement1.U.Actual	676	4.80
Stage2.Output.Measurement2.U.Actual	604	4.29
Stage2.Output.Measurement3.U.Actual	772	5.48
Stage2.Output.Measurement4.U.Actual	12754	90.53
Stage2.Output.Measurement5.U.Actual	189	1.34
Stage2.Output.Measurement6.U.Actual	188	1.33
Stage2.Output.Measurement7.U.Actual	346	2.46
Stage2.Output.Measurement8.U.Actual	973	6.91
Stage2.Output.Measurement9.U.Actual	4198	29.80
Stage2.Output.Measurement10.U.Actual	635	4.51
Stage2.Output.Measurement11.U.Actual	628	4.46
Stage2.Output.Measurement12.U.Actual	256	1.82
Stage2.Output.Measurement13.U.Actual	193	1.37
Stage2.Output.Measurement14.U.Actual	912	6.47

C.2 Ausschnitt des ausgefüllten statistischen Versuchsplans

Die CSV-Datei des komplett ausgefüllten SVP trägt den Namen *Versuchsplan.csv* und die mit den Ergebnissen *Versuchsplan_mit_Ergebnissen0_810.csv*. Die ersten 20 Zeilen beider Dateien werden in Tabelle C.3 und Tabelle C.4 hier im Anhang dargestellt.

Tabelle C.3: Ausschnitt der ersten 20 Zeilen des Versuchsplans

fw_MedianImputation	fw_MeanImputation	fw_LöschenMerkmale	fw_KNNImputation	fw_RegressionImputation	fw_Interpolation	fw_MICE	fw_MissForest	dt_MinMaxSkalierung	dt_Normalisierung	dt_RobustScale	dt_ZScore	dt_Summe	dt_Mittelwert	dt_Glättung_MovingAverage	dt_Glättung_Exponential	dt_Differenz_bilden
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1

Tabelle C.4: Ausschnitt der ersten 20 Zeilen des Versuchsplans mit Ergebnissen

fw_MedianImputation	fw_MeanImputation	fw_LöschenMerkmale	fw_KNNImputation	fw_RegressionImputation	fw_Interpolation	fw_MICE	fw_MissForest	dt_MinMaxSkalierung	dt_Normalisierung	dt_RobustScale	dt_ZScore	dt_Summe	dt_Mittelwert	dt_Glättung_MovingAverage	dt_Glättung_Exponential	dt_Differenz_bilden	r2_mean	r2_sum	r2_median	rmse_mean	rmse_sum
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.58	8.64	0.60	0.26	3.86
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.68	10.23	0.72	1.50	22.47
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0.59	8.86	0.67	0.24	3.59
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0.72	10.86	0.75	1.36	20.42
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0.57	8.54	0.64	0.24	3.60
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0.72	10.84	0.75	1.36	20.39
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0.51	7.63	0.52	1.20	17.94
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0.70	10.43	0.81	1.95	29.22
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0.61	9.10	0.64	1.07	16.00
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0.78	11.64	0.86	1.54	23.05
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0.59	8.86	0.65	1.10	16.57
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0.74	11.08	0.83	1.65	24.70
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0.51	7.64	0.54	6.02	90.32
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0.51	7.60	0.56	12.00	179.99
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0.61	9.12	0.64	5.37	80.56
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0.78	11.63	0.86	7.68	115.21
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0.58	8.74	0.64	5.58	83.75
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0.35	5.25	0.60	12.45	186.80
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0.53	7.96	0.58	0.27	4.12
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0.65	9.81	0.67	1.59	23.88