Anwendungspotential von Maschinellen Lernprozessen auf XML-Standardisierten Mess- und Prüfdaten

Application potential of machine learning for XML standardised measurement and test data

Bachelorarbeit

vorgelegt am: 29.10.2018

Institut für Informationsmanagement im Ingenieurwesen

Prof. Dr. Dr.-Ing. Dr. h. c. Jivka Ovtcharova

Institut für Angewandte Informatik und Formale Beschreibungsverfahren

Prof. Dr. Andreas Oberweis

Betreuer:
Andreas Kimmig

Sommersemester 2018

von

Simon Richebächer

E-Mail: simon.richebaecher@parsqube.de

Aufgabenstellung

Zielsetzung der Arbeit ist es, die vorhandene Datenerfassung einer realen Produktionsstätte zu untersuchen, deren Potential für eine Anwendung von Maschinellem Lernen (ML) zu evaluieren und die angewandte Methodik für zukünftige Evaluationen gleicher Problemstellungen zu erfassen. Der Fokus liegt dabei in der Verarbeitung von XML-Datensätzen aus Mess- und Prüfdaten einer Produktion. Die Arbeit umfasst dabei die folgenden Aspekte:

- Erarbeitung einer standardisierten Methodik zur Verarbeitung und Evaluierung von Mess- und Prüfdaten aus XML-Datensätzen im Hinblick auf deren Verwendung in ML-Verfahren
- Aufzeigen der bestehenden Datenerfassung und Verarbeitung innerhalb des Anwendungsfalles
- Untersuchung und Testen der vorhandenen Datensätze auf Informationsgewinn durch ML

Eidesstattliche Erklärung

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, den 27.10.2018

(Simon Richebächer)

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Abkürzung	Bedeutung
ML	Maschinelles Lernen
SEL	Symbolisch Empirisches Lernen
EBL	Erklärungsbasiertes Lernen
SVM	Support Vector Machines
OLS	Ordenary Least Square
SGD	Stochastic Gradient Descent
PCA	Principal Component Analysis
LOF	Local Outlier Factor
NN	Neuronales Netz
OLAP	Online Analytical Processing
XML	eXtensible Markup Language
CRISP-DM	Cross Industry Standard Process for Data Mining
IDE	Integrated Developer Environment
GUI	Graphical User Interphase
NaN	Not a Number
ROC	Receiver Operating Characteristic
KPI	Key Performance Indicator

1. Einleitung

"Discovery is no longer limited by the collection and processing of data, but rather management, analysis, and visualization."
— Damian Mingle

Durch die zunehmende digitale Integration in Produktionsabläufen erfassen industrielle Unternehmen eine immer größere Menge an Daten. Diese Integration ermöglicht es den Unternehmen Daten abteilungsübergreifend auszutauschen und auszuwerten, ebenso eröffnet es vielfältige Möglichkeiten zur intelligenten Datenanalyse. Einen wachsenden Einfluss auf die Umsetzung von intelligenten Systemen hat der Bereich des Maschinellen Lernens (ML). Mit der stetigen Datenverfügbarkeit durch die Digitalisierung und durch signifikante Entwicklungen im Forschungsfeld des ML, ist der Einsatz dieser Technologie immer attraktiver geworden. Sie bietet neue Ansatzpunkte zur Identifikation von kritischen Prozessabschnitten, ungenutzten Ressourcen und anderweitigem Optimierungspotential innerhalb der Produktion. Dementsprechend groß ist das Interesse dieses Potential für die automatisierte Auswertung von Mess- und Prüfdaten aus Produktionsprozessen zu nutzen.

Grundlage für den effizienten Einsatz von ML auf Messdaten ist eine standardisierte Erfassung, ausreichend signifikante Datenmengen und eine anwendungsspezifische Vorverarbeitung der Daten. Diese Voraussetzungen sind für Unternehmen die im Begriff sind ihre Produktion digital zu integrieren zunächst zu erfüllen. Die Evaluation des Status Quo und des einhergehenden Anwendungspotentials von ML ist daher ein erster Schritt in die Richtung einer intelligenten Datenanalyse.

Zum unternehmens- und bereichsübergreifenden Austausch von Daten ist der Einsatz des XML-Formates ein etablierter Standard. Das „World Wide Web Consortium" (W3C) empfiehlt XML als Austauschformat für strukturierte Daten jeder Art. Diese Gewichtung des Formates im Zusammenhang mit Internetdiensten und seine flexible Verwendbarkeit werden auch in den industriellen Sektor getragen. Der Einsatz von XML übernimmt daher auch in der Erfassung von produktionsbezogenen Mess- und Prüfdaten eine immer wichtigere Rolle.

Aus den erklärten Zusammenhängen lässt sich die zunehmende Relevanz von ML-Potential im Rahmen von XML-Standardisierten Mess- und Prüfdaten feststellen. Diese Arbeit behandelt die Untersuchung der Thematik in ihrer Vielschichtigkeit. Durch den Anwendungsfall einer realen Produktion wird Praxisnähe erzeugt. Innerhalb einer Machbarkeitsstudie wird die XML-Standardisierte Datengrundlage des Unternehmens im Hinblick auf ihre Verwendbarkeit für ML untersucht. Alle daraus gewonnenen Erfahrungswerte fließen in die Erstellung einer Methodik zur Durchführung gleichartiger Projekte. Der Mehrwert dieser Arbeit liegt daher in dem Erkenntnisgewinn für das gewählte Unternehmen und in der Bereitstellung einer vielschichtigen Informationsbasis für kommende Vergleichsprojekte.

2. Grundlagen und Anwendungen des Maschinellen Lernens

Um die Natur des Maschinellen Lernens (ML) in seiner vielfältigen Zusammensetzung zu verstehen, hilft es die Geschichte des Forschungsbereiches in den Phasen seiner großen Entwicklungsschübe zu betrachten. Die erste Phase im Zeitraum von 1940 bis 1960 konzertierte sich auf Neuronale Modelle und Entscheidungstheoretische Methoden. Das Ziel war es, die Erkenntnisse der neuronalen Forschung auf maschinelle Systeme zu übertragen. Durch Manipulation von Wahrscheinlichkeiten einer Signalweiterleitung, ähnlich der Funktionsweise eines Neurons, erhielten solche Systeme die ersten Ansätze zur Lernfähigkeit (Michalski, et al., 1983, p. 14). Die jedoch noch in den Kinderschuhen stehende Computertechnik schränkte die Forschungsmöglichkeiten ein und führte zu einem eher theoretischen Erkenntnisgewinn nebst experimentellen Hardware Entwicklungen.

Eine zweite Forschungswelle begann in den 1960ern, als man nun nicht mehr nur auf numerische oder statistische Methoden zurückgriff. Das symbolische, konzept-orientierte Lernen wurde aus der Forschung von Menschlichem Lernverhalten gewonnen und erlaubte es maschinellen Systemen durch symbolische Beschreibungen mit hochwertigeren Wissensrepräsentationen zu arbeiten (Michalski, et al., 1983, p. 15). Mit der zunehmend zugänglichen und vergleichsweise leistungsstarken Computertechnik der späten 70er und frühen 80er, eröffneten sich neue Möglichkeiten. Die Disziplin weitete sich mit der Erforschung von vielfältigen, informationsintensiven Aufgabenbereichen für lernende Systeme. Dies führte zu einer Bandbreite von Lernstrategien, welche auch noch heute die Basis für grundlegende ML Verfahren darstellen.

In den vergangenen zwei Jahrzehnten hat das Forschungsfeld des ML enorme Fortschritte in den Computerwissenschaften und den datenintensiven Industrien ermöglicht. Dazu beigetragen hat nicht nur die kontinuierliche Kostensenkung in der Informationstechnologie, sondern auch die Datenexplosion durch die zunehmende digitale Vernetzung im privaten und öffentlichen Bereich. Im Umgang mit diesen großen und immer gefragteren Datenmengen, hat sich das ML als unabdingbares Werkzeug erwiesen. Wegweisende Forscher des Maschinellen Lernens wie T.M. Mitchell bezeichnen es deshalb als eine der wichtigsten, transformativen Technologien des 21 Jahrhunderts (Mitchell & Jordan, 2018, p. 260).

2.1 Von der Lerntheorie zum Maschinellen Lernen

Der klassische Einstieg in das ML befasst sich zunächst mit der Frage nach der Begrifflichkeit des Lernens. Eine natürliche Assoziation verbindet das Lernen mit einer gewissen Zunahme an Erfahrung in einem bestimmten Aufgabenbereich. Formalisiert hat dies Tom Mitchell (1997, p. 2) wie folgt:

„Ein System lernt aus Erfahrung E, in Bezug auf eine Klasse T von Aufgaben und dem Bewertungsmaß P, falls es seine Leistung in Aufgaben von T, gemessen durch P, um Erfahrung E verbessert."

Es stellt sich damit die Frage nach der Art der Erfahrung und der Art von Aufgabenklassen. Eine klassische Unterteilung in zwei grundlegenden Lernarten ist uns als Menschen nur allzu vertraut. Unterschieden wird zwischen dem Erwerb neuen Wissens und dem Erwerb neuer Fähigkeiten, beschrieben durch eine Verbesserung des existierenden Wissens während seiner Anwendung (Briscoe & Caelli, 1996, p. 6). Aus diesen zwei Grundsätzen resultieren zwei wichtige Ansätze des Forschungsfeldes. Das Symbolisch Empirische Lernen (SEL) fokussiert sich auf induktive Lernmethoden zum Erwerb neuen Wissens. Die meisten Algorithmen des SEL lernen dabei aus den gemeinsamen Eigenschaften aller beobachteten Instanzen eines Konzeptes und tragen daher die Bezeichnung „Gleichheitsbasierend" (Briscoe & Caelli, 1996, p. 7). Der zweite Ansatz ist das Erklärungsbasierte Lernen (EBL), welches im Umkehrschluss eher deduktive Lernmethoden beinhaltet. Es wird also vorhandenes Wissen ausgenutzt um bestehende Regeln zu verbessern oder die Suche nach spezifischen Regeln effizienter zu gestalten. Das EBL findet in der maschinellen Sprachverarbeitung seine Anwendung. Es bietet darüber hinaus aber wenig Potenzial für ein breites Anwendungsspektrum und wird daher nicht tiefgreifender behandelt werden.

Um eine weitere Unterteilung und folglich auch Klassifikation von Lernstrategien zu ermöglichen, ist eine Unterscheidung nach der vom System angewandten Inferenz eine gängige Variante (Michalski, et al., 1983, p. 7). Ein ML-Verfahren leistet geringe Inferenz auf übergebene Informationen, falls die gewonnene Erkenntnis nur der Vorgabe von schon bekannten Mustern entstammt. Hingegen leistet es hohe Inferenz, falls es auch ohne diese Vorgaben neue Muster erkennen kann. Je mehr das System zu Inferenz fähig ist, desto geringer ist der Aufwand zur Überwachung und Steuerung. Dieser Umstand führt zur gängigen Klassifikation in überwachte und unüberwachte Lernstrategien.

In Abbildung 1 ist eine Auswahl von gängigen ML Verfahren in ihren Klassen zu sehen. Diese werden in den folgenden Unterkapiteln des Grundlagenteils behandelt werden.

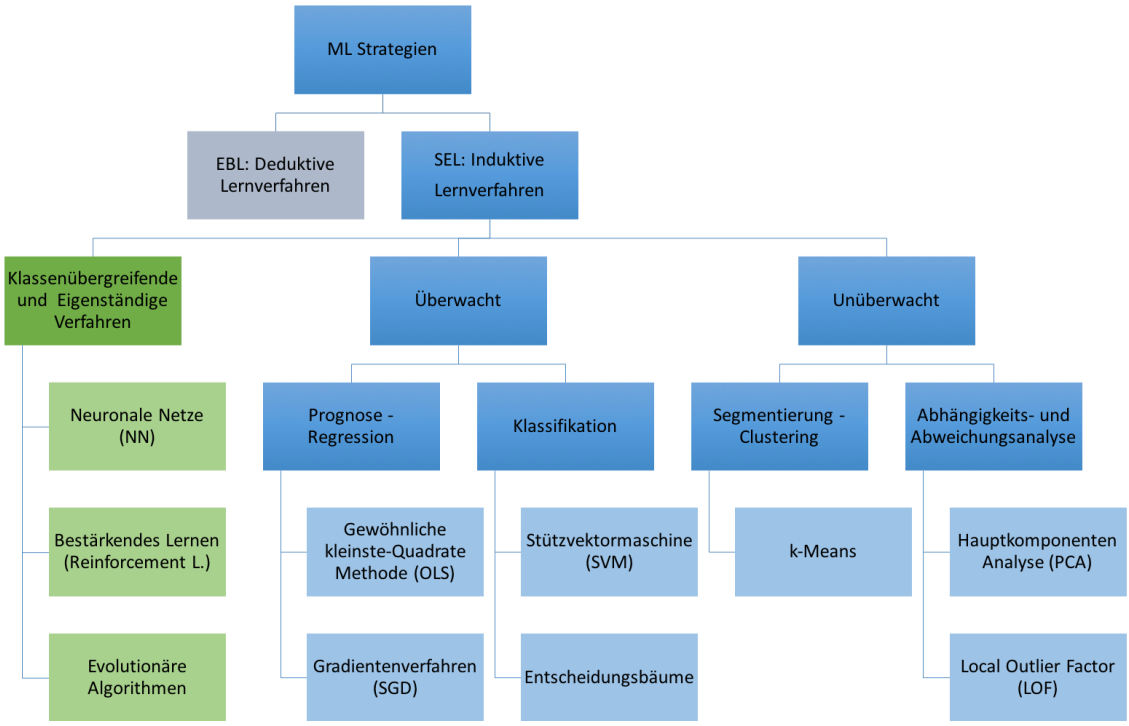


Abbildung 1: Klassifikation gängiger ML Verfahren

Die in Abbildung 1 gesondert aufgeführten Verfahren Neuronale Netze, Bestärkendes Lernen und Evolutionäre Algorithmen werden am Ende von Kapiteln 2.4 kurz erläutert. Mit der stetigen Zunahme an verfügbaren Rechenkapazitäten erhalten diese Verfahren immer mehr Bedeutung. Ihre Verwendung beschränkt sich jedoch auf spezifische Problemstellungen, welche auf den in der Arbeit untersuchten Anwendungsfall nicht zutreffen. Ein kurzer Überblick über die klassischen Aufgabengebiete von ML Verfahren ermöglicht folgende Unterteilung (Mitchell & Jordan, 2018, p. 17):

a) Data Mining Probleme mit impliziten Mustern, welche erst durch automatisierte (ML) Verfahren zu Tage treten

b) (Forschungs-)Bereiche mit vielen Unbekannten und dem fehlenden Wissen zur Entwicklung effektiver Algorithmen

c) Anwendungsumgebungen in denen sich das Programm dynamisch an kontinuierlich geänderte Bedingungen anpassen muss

Dabei finden überwachte und unüberwachte Lernverfahren vor allem in Bereich a) ihre Verwendung. Dort bilden sie das gängige Werkzeug von Datenwissenschaftlern zur Erkennung von verborgenen Mustern und Zusammenhängen in großen Datenmengen. In einer Arbeitswelt mit stetiger Datenzunahme gewinnt der Bereich des Data Science als Schnittstelle zwischen Computerwissenschaften, Statistik und Wirtschaft, daher nicht ohne Grund an Bedeutung (Davenport & Patil, 2012).

2.2 Anforderungen und Vorgehensweise bei ML-Problemen

Die Grundlage zur Datenverarbeitung durch ML Verfahren liegt zunächst in dem Design eines funktionierenden Lernsystems. Ziel dieses Kapitels ist es, einen ersten Überblick über die benötigten Bausteine eines Lernsystems zu bekommen. Der Designprozess ist entnommen aus „Machine Learning“ (Mitchell, 1997) und beginnt mit der Wahl einer sogenannten Trainingserfahrung. Diese stellt mit der Datengrundlage und Problemstellung die Weichen für die spätere Auswahl einer passenden Lernstrategie.

Ein oft verwendetes Anwendungsbespiel bietet das Brettspiel Dame, welches man einem Lernsystem beibringen möchte zu spielen. Wie gut das lernende System dabei abschneidet, ist nicht zuletzt von den verwendeten Trainingsbeispielen abhängig. Unterschieden wird dabei zwischen direkten und indirekten Trainingsbeispielen. Erstere sind Daten, welche schon vor dem Training erhoben wurden und direkt dem Lernsystem zur Verfügung stehen. Auf das Beispiel bezogen könnten dies bestimmte Steinpositionen und deren Aussagengehalt sein. Die indirekten Daten werden hingegen aus den Ergebnissen des lernenden Systems gewonnen, um es iterativ zum Lernziel zu führen (Mitchell, 1997, p. 5). Ein passendes Beispiel wären die verwendeten Spielzüge im Dame-Spiel und das damit zusammenhängende Spielergebnis. Auf Grund der Relevanz für die spätere Anwendung sei speziell auf das Aufteilen des vorhandenen Datensatzes in Trainings- und Testdaten hingewiesen. Mit dieser Technik lässt sich auch ohne neue Datenquellen der Lernerfolg des Systems nach dem Training an Hand der Testdaten überprüfen.

Je nach Art des lernenden Systems müssen die Daten außerdem schon im Vorhinein auf ihren Informationsgehalt durch einen (menschlichen) „Lehrer“ überprüft und entsprechend sortiert werden (siehe Inferenz in Kapitel 2.1). Die Wertegruppen oder Eigenschaften mit der die Ausgangsdaten klassifiziert wurden, sind im Englischen unter den Begriffen Test Features und Training Features bekannt. Die Güte, mit der sie das Szenario der finalen Anwendung repräsentieren auf welches das lernende System trainiert wird, ist dabei ein

wichtiges Attribut. ML-Verfahren verlassen sich dabei auf die Annahme, dass sich die Training Features den abschließenden Test Features in ihrer Art und Verteilung ähneln (Mitchell, 1997, p. 6). Sollte dies nicht der Fall sein, kann dies zu den am Ende des Kapitels beschriebenen Problemen im ML führen.

Der zweite Schritt auf dem Weg zum Lernsystem ist die Wahl einer Zielfunktion. Je nach Art des vermittelten Wissens und des verwendeten Performance-Maß kann diese unterschiedlich komplex formuliert sein. Kurzgefasst kann eine solche Funktion als Abbruchkriterium für ein ML-Verfahren verstanden werden. Der Wert oder die Werte dieser Zielfunktion sind essentielle Ausgangsinformationen und werden im Folgenden als Labels bezeichnet werden. Während die Features die Eingabe des Datensatzes charakterisieren, so sind die Labels der Ausgabe des Lernverfahrens zugehörig. Auch hier wird wie in Schritt eins in zwei Gruppen aufgeteilt: Training Labels und Test Labels. Um mit dem Beispiel des Dame Spiels abzuschließen, folgt eine kurze Erläuterung zur Anwendung der Datengrundlage: Nach Aufteilung der Ausgangsdaten in Trainings- und Testdaten wird der ML-Algorithmus zunächst mit Hilfe der Training Features auf die Training Labels geschult. Er ordnet z.B. den Steinpositionen gewisse Spielbewertungen zu und kann somit den aktuellen Spielstatus einschätzen. Als nächstes testet man den Algorithmus, in dem man ihm die Test Features übergibt und die daraus resultierenden Ergebnisse mit den Test Labels vergleicht, um Aussagen über die Güte seiner Einschätzungen zu treffen. Ein erfolgreicher ML-Algorithmus würde in diesem Fall eine Spielbewertung aus der neu übergebenen Steinposition generieren, welche der wahren Bewertung des Test Labels möglichst ähnlich ist.

Das gerade verwendete Beispiel hat einen schon fertig erstellten Lernalgorithmus vorausgesetzt. Zum vollständigen Design eines lernenden Systems werden daher noch zwei Schritte benötigt: Das Bestimmen einer Repräsentation des Lernverfahrens und das Bestimmen eines daraus abgeleiteten Lernalgorithmus. Die Repräsentation lässt sich als Mathematisches Konzept des Lernverfahrens beschreiben, während der Algorithmus eine konkrete, funktionsfähige Ablaufbeschreibung dessen ist. Da sich diese Arbeit mit der Anwendung und nicht mit der Herleitung von ML-Algorithmen beschäftigt, wird auf die detaillierte Ausführung dieser Schritte verzichtet. Im Anwendungsfall stehen für bestimmte Problemstellungen schon vordefinierte Lernalgorithmen zur Verfügung. Bei der Auswahl dieser hilft es jedoch, sich an den zwei Schritten zu orientieren. So stellen bestimmte Algorithmen wie z.B. Support Vektor Machines (SVM) (siehe Kapitel 2.3) eine eigene

Klasse von Lernverfahren mit gleichem Grundkonzept dar. Sie unterscheiden sich aber untereinander durch ihre Optimierungsverfahren (bei SVM linear oder nichtlinear), welche sie auf unterschiedlichen Wegen und mit unterschiedlichem Erfolg zum Ziel führen. Je nach Problemstellung sollte man sich daher bei der Auswahl zunächst vom allgemeinen Konzept zum konkreten ML-Verfahren vorarbeiten.

Bezüglich des beispielhaften Lernsystems, welches lernt verschiedene Steinpositionen einzuschätzen, wurde schon die Frage nach der Güte seiner Spieleinschätzung gestellt. Ein optimal lernendes System besitzt eine auf den Anwendungsfall ausreichende Fähigkeit zur Generalisierung. Es sollte die ihm übergebenen Features überhaupt zunächst in verwertbare Resultate wandeln können. Dies setzt ein ausreichendes Training voraus. Andererseits sollte das Programm aber auch genügend Flexibilität besitzen, um sich nicht von unbekannten Feature-Werten allzu stark beeinflussen lassen. Es wurde also nicht auf einen Spezialfall hin übertrainiert. Wird dieses gewünschte Level an Generalisierung nicht erreicht, spricht man im Englischen von Underfitting oder Overfitting. Dies sind zwei klassische ML-Probleme, welche die Güte der Resultate einer ML-Anwendung negativ beeinflussen.

Es können noch viele weitere anwendungsspezifische Komplikationen auftreten, in Tabelle 1 der folgenden Seite wurde sich jedoch auf allgemeine und besonders häufig auftretende Problemtypen konzentriert (Duda, et al., 2001, pp. 11-16). Darüber hinaus wird sich diese Arbeit im Anwendungsfall nur mit der Verarbeitung von numerischen Werten beschäftigen, weshalb an dieser Stelle die Problemtypen bezüglich Sprach- und Texterkennung vernachlässigt werden.

Mangelhafte Feature Extraktion	Bezüglich der Ausgangsdaten stellt sich die Grundsatzfrage: Was ist die optimale Menge an Features und welche besitzen Relevanz für das Modell? Ohne repräsentative Datengrundlage wird das ML-Programm keine guten Resultate liefern.
Rauschen und nicht plausible Ausreißer	Während der Datenerfassung kann es zu zufälligen Umwelteinflüssen kommen. Diese beeinflussen ggf. das betrachtete Szenario und gefährden den Aussagengehalt.
Overfitting	Das ML-Programm ist perfekt auf den Trainingsdatensatz geschult, besitzt im Test oder Anwendungsfall jedoch eine schlechte Performance. Es wird auch von einer starken Voreingenommenheit, dem High Bias gesprochen. Die Ursache ist meist eine mangelhafte Feature Extraktion.
Underfitting	Durchweg schlechte Performance, meist auf Grund einer ungenügenden Menge an Trainingsdaten oder mangelhafter Feature Extraktion.
Fehlende Werte, Dubletten, uneinheitliche Bezeichnungen und Einheiten	Ob während der Datenerfassung oder dem Datentransfer, es kann immer zu einer fehlerhaften Dokumentation kommen. Diese Störfaktoren müssen kompensiert oder eliminiert werden.
Rechenaufwand und Problemkomplexität	Welche Skalenfaktoren herrschen für die Anwendung des ML-Algorithmus? Die Arbeit mit größeren Datenmengen und anspruchsvollen Algorithmen muss vorhandene Kapazitäten und Rechenzeiten berücksichtigen.

Tabelle 1: Dominante Problemklassen numerischer ML Aufgaben

Zur Vorbeugung und einem effizienten Umgang mit solchen Problemtypen ist ein strukturierter Ablaufplan für die Datenvorverarbeitung, dem sogenannten Pre-Processing, von Nutzen. Dieses ist jedoch nur einer der ersten Schritte auf dem Weg ein passendes Lernsystem für eine spezifische Problemstellung aufzusetzen. Bezüglich dem in Kapitel 4 erläuterten Anwendungsfall, wird im methodischen Teil ein detaillierter Ablaufplan erarbeitet werden.

2.3 Überwachtes Lernen

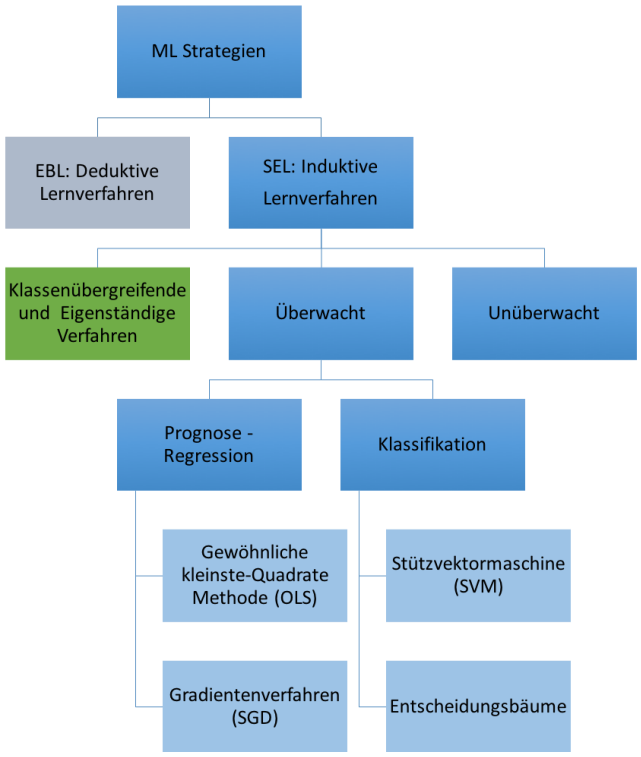


Abbildung 2: Repräsentative Auswahl überwachter ML Verfahren

In Kapitel 2.1 wurde eine erste Unterteilung der heute üblichen Lernstrategien in Überwacht und Unüberwacht eingeführt. Dabei ist die breite Masse der heutzutage verwendeten Lernstrategien dem überwachten Typ zugehörig. Dies lässt sich darauf zurückführen, dass viele der überwachten Lernalgorithmen ein praktikables Werkzeug für eine grundlegende statistische Untersuchung und Sortierung der Ausgangsdaten darstellen. Daraus lassen sich auch gleich die zwei Aufgabengebiete der überwachten Lernverfahren herauslesen: Regression und Klassifikation (Müller & Guido, 2017, p. 27).

Der jeweilige Typ der Aufgabe ist eng mit dem Zielverständnis des Verfahrens verbunden. Während die Regression eine (zeitlich) kontinuierliche Unterscheidung der Datenpunkte verwendet, setzt die Klassifikation eindeutig definierte Klassengrenzen voraus. Das Vorhandensein von Kontinuität in den Daten ist also ein guter Indikator für die Art des Aufgabentypus. Weiterhin benötigen überwachte Lernalgorithmen, ihrer geringen Inferenzfähigkeit entsprechend, mehr Informationen über Ihre Ausgangsdaten in Form von a priori gelabelten Daten (Oettinger, 2017, p. 93). Die Einzeldaten besitzen also schon eine eindeutige Verknüpfung der Feature-Werte mit den zugehörigen Labels.

Einen guten Einstieg in überwachte ML Algorithmen bietet die lineare Regression, mit der kleinste-Quadrate Methode (eng. OLS). Graphisch beschrieben hat dieser ML-Algorithmus das Ziel den mittleren quadratischen Abstand (R^2) zwischen den Datenpunkten der Vorhersage und jenen der tatsächlichen Werte zu minimieren (siehe Tabelle 3). Es wird also eine Vorhersagefähigkeit aus der Schulung mit den Training-Features und Labels gewonnen, deren Aussagekraft dann mit Hilfe der Test-Features und Labels evaluiert wird. Gewünscht ist hier ein R^2-Wert der Vorhersage, welcher möglichst hoch ist und nahe an dem Wert des Testdatensatzes liegt. Ist der R^2-Wert der Vorhersage um einiges höher, liegt hier ein klassisches Overfitting vor.

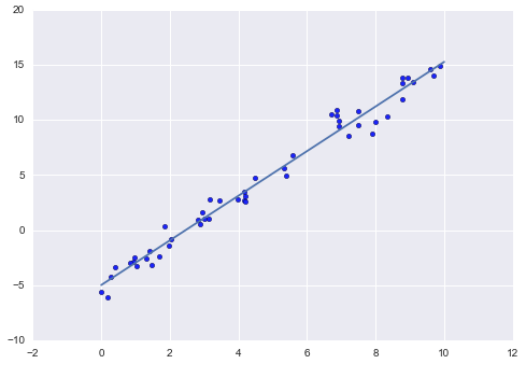
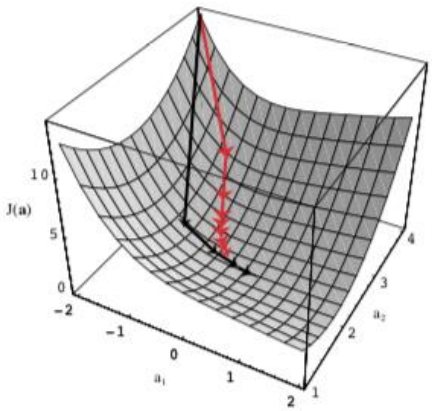
	
Lin. Reg. (VanderPlas, 2016, p. 391)	SGD (Duda, et al., 2001, p. 15)

Tabelle 2: Visualisierung überwachter ML Verfahren Teil 1

Während die OLS Methode die optimalen Parameter zur Vorhersage aus dem linearen Zusammenhang der Daten bezieht, geht das Gradientenverfahren (engl. SGD) anders vor. Die Startparameter des Lernverfahrens sind zufällig ausgewählt und der Algorithmus erlernt erst nach mehreren Iterationen die optimalen Parameter. Entscheidend ist hierbei der

Gradient der sogenannten Loss Funktion, welche als Maß für die Abweichung zwischen den wahren und den Vorhersage-Werten dient. Je nach aktueller Steigung der Funktion wandert das Verfahren nun in die Richtung geringerer Abweichung, mit dem Ziel ein lokales oder globales Minimum zu erreichen (siehe Tabelle 2). Das Verfahren ist für den Fall der linearen Regression recht ineffizient, da der optimale lineare Zusammenhang schon recht eindeutig ist. Es ist jedoch ein gängiges Optimierungsverfahren sehr vieler ML-Algorithmen, weil lineare Zusammenhänge und eindeutige Startparameter eher die Ausnahme darstellen. Muss sich ein ML-Algorithmus also iterativ an die Extrempunkte eines Problems heranarbeiten, so ist SGD ein übliches Verfahren.

Auch zur Klassifikation können lineare Modelle verwendet werden. Der Grundgedanke dabei ist, den Raum der Datenpunkte durch lineare Funktionen in die gewünschten Räume der Klassen zu zerteilen. Bezogen auf multidimensionale Probleme mit mehr als nur zwei Features spricht man daher auch von Hyperebenen oder -flächen, welche den Feature Raum aufteilen. Da es bei der Menge an Datenpunkten im Regelfall zu Übergangsbereichen zwischen den Klassengrenzen kommt, wird zur Bestimmung dieser gemischt besetzten Grenzen die Wahrscheinlichkeitsverteilung der Logistischen Funktion genutzt. Die Minimierung dieser Funktion lässt sich bei dem Verfahren der Logistischen Regression also als Minimierung der Missklassifikationswahrscheinlichkeit verstehen. Sie ist eine der fundamentalen Techniken zur Klassifikation und Grundlage für viele weitere Klassifikationsverfahren.

Ein weiteres essentielles Verfahren zur Unterteilung des Feature Raumes sind Support Vector Machines (SVM). Im Gegensatz zur Logistischen Regression wird hier nicht mit einer Wahrscheinlichkeitsverteilung im Trennungsband des Übergangsbereiches gearbeitet. Die SVM Methode berechnet die Trenngerade oder Hyperfläche, welche den größtmöglichen Abstand zu den nächstliegen Punkten der Klassen besitzt. Dazu verwendet werden die Abstände der im Trennungsband enthaltenen Punkte, die sogenannten Stützvektoren (siehe Tabelle 3). Je nach gewünschter Toleranz gegenüber einer Missklassifikation durch den Übergangsbereich ist diese Methode flexibel verstellbar. Weiterhin können SVM auch von nichtlinearer Natur sein. Durch Polynomiale Annäherung mit der Radial-Basis-Function und dem Kerneltrick (Müller & Guido, 2017), lassen sich auch ohne großen Rechenaufwand nichtlineare Klassifikationsgrenzen in höherdimensionalen Feature Räumen berechnen. Wie auch bei den Methoden aus der Regression, ist bei den SVM-Verfahren darauf zu achten, dass es nicht zu Overfitting

kommt. Der zugehörige SVM-Score erlaubt es, genau wie der R^2 Vergleich, den Unterschied zwischen Training und Test Labels festzustellen.

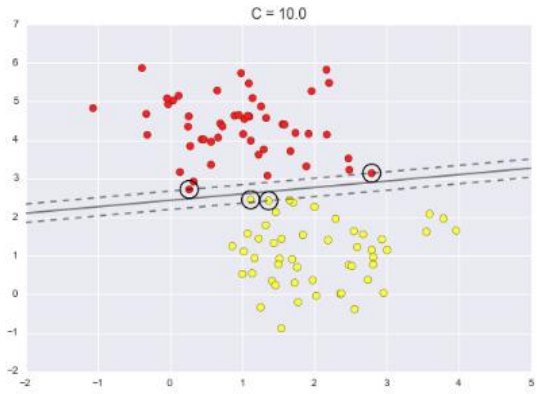
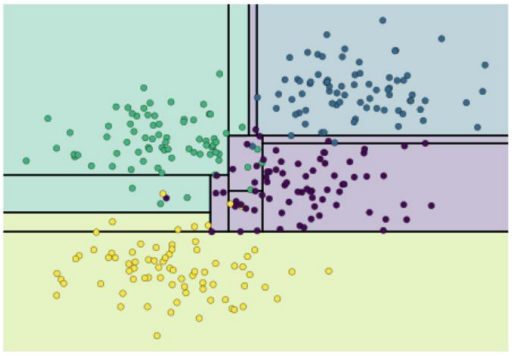
 SVM (VanderPlas, 2016, p. 410)	 Decision Tree (VanderPlas, 2016, p. 424)

Tabelle 3: Visualisierung überwachter ML Verfahren Teil 2

Für Klassifikationsaufgaben wird auch häufig die Methode der Entscheidungsbäume genutzt. Der Lernalgorithmus erlernt dabei schrittweise den Feature-Raum anhand der Feature Werte aufzuteilen. So beginnt er im „ersten Level“ bei einer Aufteilung des Raumes in zwei Unterräume, welche anhand einer Feature Ausprägung alle Punkte eindeutig einem der zwei Räume zuordnen kann. Auf jedem weiteren Level werden die Unterräume wieder anhand eindeutiger Merkmale zerteilt, bis alle Datenpunkte einer a priori gelabelten Klasse durch einen Entscheidungszweig charakterisiert werden (siehe Tabelle 3). Eine solche vollständige Zerteilung ist in der Anwendung jedoch selten gewünscht, da bei einer hohen Anzahl an räumlichen Aufspaltungen die Fähigkeit zur Generalisierung immer mehr verloren geht. Entscheidungsbäume sind daher auch recht anfällig für Overfitting (Müller & Guido, 2017, p. 71) und benötigen eine spezielle Anpassung von Parametern wie z.B. dem maximalen Aufspaltungs-Level. Ein Vorteil von Entscheidungsbäumen sind dahingegen die geringen Anforderungen an die Vorverarbeitung. Es können nämlich auch nicht numerische, kategorische Werte direkt in den Entscheidungsbaum eingelesen werden.

Eine nennenswerte Erweiterung dieser Methode ist die sogenannte Random-Forest Klassifikation. Diese teilt den Trainingsdatensatz in Untergruppen auf, welche jeweils eine Entscheidungsbaum-Klassifikation durchlaufen. Nach Zusammenführung der Teilergebnisse wird ein Durchschnitt für das Gesamtergebnis berechnet. Dieses hilft dem Overfitting entgegenzuwirken, wobei hier jedoch auf den Trade-off zwischen gesenkter

Varianz und erhöhtem Bias geachtet werden muss. Mit den Entscheidungsbäumen sind die gängigen überwachten Methoden abschließend aufgeführt. Es stellt sich nun die Frage welche Lernalgorithmen man auf Daten ohne a priori Label anwenden kann. Dies wird im folgenden Kapitel thematisiert werden.

2.4 Unüberwachtes Lernen

Kann ein Lernalgorithmus ohne vorherige Angabe der Labels einen Informationsgewinn aus den Ausgangsdaten erzeugen, so gehört er zu den unüberwachten Lernmethoden. Die Mustererkennung von solchen Verfahren ermöglicht es, die fehlenden Labels und damit die vorher unbekannte Struktur des Datensatzes zu ermitteln. Unüberwachte Algorithmen eignen sich durch ihren explorativen Charakter darüber hinaus als Hilfsmittel für die effiziente Feature-Auswahl, Datenbereinigung und -korrektur. Erreicht wird dies durch das Erkennen von Segmenten, Abhängigkeiten und Abweichungen innerhalb des Datensatzes. Diese drei klassischen Aufgabenbereiche sind folglich besonders hilfreich zur Vorverarbeitung der Daten und werden im Folgenden durch jeweils ein gängiges Verfahren repräsentiert werden (siehe Abbildung 3).

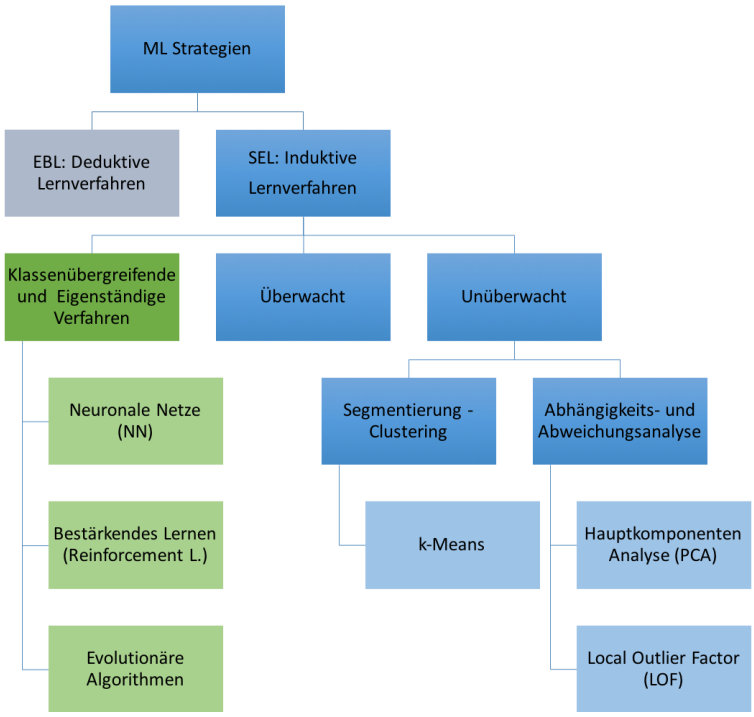
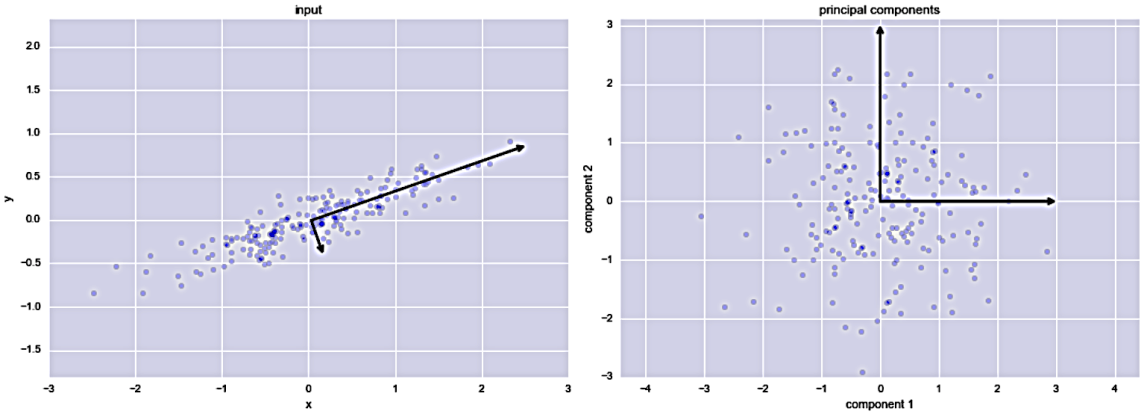


Abbildung 3: Repräsentative Auswahl unüberwachter ML Verfahren

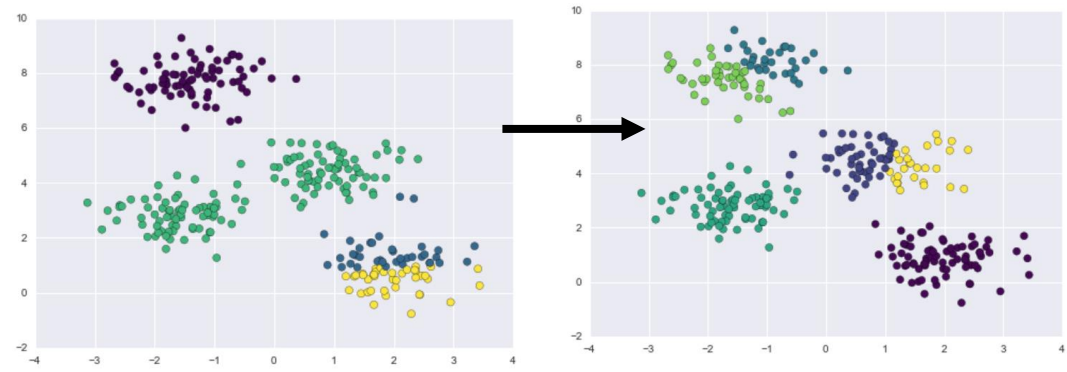
Die Visualisierung der Daten ist ein wichtiges Werkzeug zur Datenexploration und erleichtert dem Menschen die Interpretation enorm. Es muss sich jedoch auf eine niederdimensionale Darstellung, meist 2D oder 3D, beschränkt werden. Die Projektion des hochdimensionalen Datensatzes auf einen niederdimensionalen Raum wird durch die Hauptkomponentenanalyse (engl. PCA) ermöglicht (Müller & Guido, 2017, p. 132). Dabei wird sich an der Koordinatenachse mit der größten Varianz innerhalb der Daten orientiert. Im Anschluss werden die weiteren, zur Hauptachse orthogonalen, Achsen nach Varianz absteigend sortiert und je nach Darstellungsdimension mit ihr als neues Koordinatensystem verwendet. Die ursprünglichen Variablen werden somit orthogonal transformiert und bilden anschließend die neuen, unkorrelierten Hauptkomponenten (siehe Tabelle 5). Mit dieser Technik lassen sich schnell Muster innerhalb des Datensatzes identifizieren und daher ist die PCA ein gängiges Mittel zur Abweichungsanalyse. Sie eignet sich bei niedrigdimensionalen Datensätzen auch zur Abweichungsanalyse, da die Ausreißer grafisch besonders schnell identifiziert werden können. In hochdimensionalen Datensätzen müssen ansonsten Verfahren wie der Local Outlier Factor (LOF) genutzt werden. Dieser kann kurz umfasst als Kennzahl für die Anzahl an nächsten Nachbarn eines Datenpunktes beschrieben werden. Ist diese „Nachbarschaftsdichte“ im Gegensatz zu den nächsten Nachbarn besonders gering, kann ein Ausreißer identifiziert werden.



PCA (VanderPlas, 2016, p. 435)

Tabelle 4: Visualisierung unüberwachter ML Verfahren Teil 1

Liegt ein Bedarf nach Segmentierung von ungelabelten Daten vor, so werden Clustering Algorithmen verwendet. Solche Lernalgorithmen ordnen Datenpunkte den Gruppierungen, welche erst während des Prozesses entstehen, schrittweise zu. Besonders entscheidend ist dabei die Art der Gruppeneinteilung und das verwendete Ähnlichkeitsmaß. Eine Gruppeneinteilung kann Eindeutig, Überlappend oder Wahrscheinlichkeitsbasierend erfolgen und je nach Ähnlichkeitsmaß können Cluster unterschiedliche Formen bilden. Als Beispiel soll das eindeutige, abstandsbasierte k-Means-Clustering dienen. Der Lernalgorithmus verfolgt dabei das Ziel, sich iterativ an die Mittelpunkte der Cluster anzunähern. Die Initialisierung beginnt mit zufällig gewählten Datenpunkten als Dummy-Mittelpunkte. Nun werden alle Datenpunkte dem nächstliegenden Mittelpunkt zugeordnet, sodass die ersten Gruppierungen entstehen. Mit einem neu berechneten Mittelpunkt dieser Gruppierungen lässt sich im zweiten Durchlauf dann eine präzisere Zuordnung erreichen. Wiederholt man die Neuberechnung und Zuordnung bis es zu keiner Veränderung der Mittelpunkte mehr kommt, so endet die Segmentierung des Datensatzes (siehe Tabelle 5). Schwachstellen des k-Means sind die Ungewissheit über die „richtige“ Anzahl der k Mittelpunkte, komplexe Verteilungsformen der Datenpunkte und die randomisierte Auswahl der Startpunkte.



k-Means (VanderPlas, 2016, pp. 467, 468)

Tabelle 5: Visualisierung unüberwachter ML Verfahren Teil 2

Abschließend zum Grundlagenteil zu ML Verfahren werden die klassenübergreifenden und eigenständigen Verfahren (siehe Abbildung 1) erläutert. Ein übergreifendes Verfahren ist ein Neuronales Netz (NN), welches sowohl für überwachtes als auch für unüberwachtes Lernen verwendet werden kann. Durch Matrizenrechnung und das Ablaufschema eines Lernalgorithmus ermöglichen NN iteratives Training für vielfältige Einsatzmöglichkeiten

(Oettinger, 2017, p. 112). Das NN lernt, in dem es die Gewichtungen seiner Untereinheiten durch den Input von Trainingsdaten anpasst. Der dabei verwendete Lernalgorithmus wird durch die Struktur des Netzes (Topologie) und die Art der verwendeten Verrechnungsfunktionen realisiert. Genutzt werden NN auch für Verfahren des Bestärkenden Lernens. Neben überwachten und unüberwachten Lernmethoden wird das Bestärkende Lernen auch als eigenständige Klasse aufgefasst. Seine Anwendungen sind von besonders interaktivem Charakter, da das Lernsystem situationsbedingt durch positive oder negative Rückmeldung auf eine gewisse Handlung ein verbessertes Verhalten erlernt. Die Erläuterung solcher komplexen Systeme übersteigt den Umfang dieser Arbeit, es wird daher nicht weiter darauf eingegangen.

Ebenfalls komplex in der Erläuterung sind Evolutionäre Algorithmen, welche hier daher nur kurz eingeführt werden. Der Nutzen dieser Lernalgorithmen liegt vor allem in der Lösung von Optimierungsproblemen, welche Schwierigkeiten mit exponentieller Komplexitätszunahme und dem Erreichen von globalen Extrempunkten besitzen. Wie bei der natürlichen Selektion werden bei solchen Verfahren die vielversprechendsten Lösungsvorschläge einer Generation nach ihrer „Fitness“ selektiert, rekombiniert und mutiert, um sich schrittweise einem Optimum zu nähern. Die Laufzeit des Verfahrens und die Qualität der Endlösung sind jedoch im Vorhinein unbekannt (Oettinger, 2017, p. 148).

Die grundlegende Einführung in die ML Algorithmen ist mit diesem Kapitel abgeschlossen. Der durch das Kapitel 2 erarbeitete Einblick in die Thematik des ML wird den kommenden Inhalten häufig als Grundlage dienen. Kapitel 3 wird mit einer methodischen Erläuterung in die forschungsorientierte Arbeit einsteigen. Diese lässt sich als Wechselwirkung zwischen Methodik und Anwendung beschreiben. Die Inhalte der Kapitel 3 und 4 sind parallel, mit gegenseitigem Erkenntnisgewinn erarbeitet worden. Das übergreifende Prozessmodell aus Kapitel 3 hat von der praktischen Projektarbeit der Anwendung profitiert. Ergänzend hat sich der Projektablauf aus Kapitel 4 an dem Prozessmodell orientieren können. Die daraus resultierte Struktur bildet das Fundament der kommenden zwei Kapitel.

3. Data Mining nach CRISP-DM

Im Rahmen dieser Abhandlung soll ein allgemeingültiges Prozessmodell für ein definiertes ML Potential erarbeitet werden. Der Fokus liegt hierbei auf der Ermittlung des ML Potentials XML-Standardisierter (eXtended Markup Language) Mess- und Prüfdaten in der Produktion. Diese Zielthematik entstammt dem Bedarf eines realen Anwendungsfalls. Nach Betrachtung der repetitiven Natur dieses Bedarfes, wurde der Mehrwert eines Referenzmodells für gleichartige Projekte erkannt und innerhalb einer CRSIP-DM Methodik umgesetzt. Der Grund für die zukünftige Verwendbarkeit des Referenzmodells liegt in der Bedeutung des XML-Formates für Web Services im Rahmen von Online Analytical Processing (OLAP) Systemen. Seit der Jahrtausendwende nimmt die Bedeutung der unternehmensinternen Datenanalyse und der Verwendung von Web Applikationen zur Datenauswertung verstärkt zu. Auch in industriellen Unternehmungen nehmen Web Services in der vertikalen Datenintegration einen wichtigen Stellenwert ein. Web Services werden verstanden als „Software Applikationen deren Schnittstellen dazu fähig sind von XML-Artefakten definiert, beschrieben und erkannt zu werden" (Kalogeras, et al., May 2006, p. 120). Der Kontext einer Datengrundlage von produktionsbezogenen Mess- und Prüfwerten im XML-Format ist kein einzigartiger Fall für Studien im Bereich des ML.

Data Mining befasst sich mit dem Erkennen und Beschreiben von Mustern innerhalb von Daten, um wertvolle Informationen für einen bestimmten Zweck zu extrahieren (Witten & Frank, 2001, p. 3). Das Maschinelle Lernen kann hier als Informationstechnisches Werkzeug eingesetzt werden. Bei der Auswahl einer Methodik für den effizienten Ablauf eines ML Projektes, empfiehlt es sich zunächst an der allgemeinen Struktur von Data Mining Projekten zu orientieren. Etabliert hat sich diesbezüglich der industrieübergreifende, standardisierte Prozess für Data Mining (engl. **CR**oss **I**ndustry **S**tandard **P**rocess for **D**ata **M**ining) (Derksen, et al., 2013, p. 41). Ziel dieses Prozessmodells ist es, die Reliabilität und Effizienz einer Art von reproduzierbaren Data Mining Projekten zu sichern. Es zielt auf eine Balance zwischen fixer Struktur und flexiblen Elementen, um die fachübergreifenden Hintergründe zukünftiger Projektteilnehmer abzudecken. Dabei stützt sich CRISP-DM auf ein hierarchisches Modell, um mit absteigendem Abstraktionslevel von generischen zu sehr spezifischen Prozessschritten, die benötigte Bandbreite abzudecken (siehe Abbildung 4).

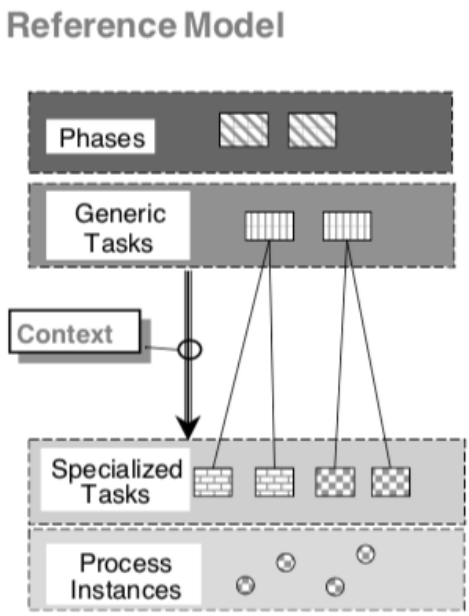


Abbildung 4: CRISP-DM Struktur (Wirth & Hipp, 2000, p. 32)

CRISP-DM unterscheidet zwei Vorgehen mit unterschiedlichen Detailgraden: Dem Referenzmodell und dem Userguide. Ersterer fokussiert sich auf eine kompakte Prozessübersicht mit definiertem Ablaufplan. Der User Guide ist eine detaillierte Herleitung aller notwendigen Arbeitsschritte. Die hier vorliegende explorative Arbeit benötigt weniger Detailtiefe und verwendet daher das Referenzmodell.

Die ersten zwei Abstraktionslevel im Referenzmodell beschreiben die generischen Phasen und Aufgabenbereiche. Dieser Teil sichert die Vollständigkeit und Stabilität des Prozesses, auch im Hinblick auf unvorhergesehene Ereignisse und zeitliche Weiterentwicklungen. Für klassische Data Mining Projekte bietet CRISP-DM daher bis zu diesem Level eine Vorlage (siehe Abbildung 5). Ab dem dritten Level werden problemspezifische Aktivitäten detaillierter dargelegt. Im Rahmen dieser Arbeit werden von dem dritten Abstraktionslevel an neue Aufgaben und Empfehlungen definiert. Dabei ist zu beachten, dass CRISP-DM absichtlich eine idealisierte Sequenz von Ereignissen verlangt. Dies vermeidet eine unproduktive Komplexität des Modells, auch wenn in der Realität Rück- und Vorgriffe auf bestimmte Schritte erlaubt oder sogar von Nöten sind. Das vierte Level dokumentiert die genauen Aktionen, Entscheidungen und Ergebnisse, kurzgefasst die Projektinstanzen des realen Anwendungsfalles. Solche Projektinstanzen sind auf Grund ihrer umfangreichen Natur primär auf den Anhang dieser Arbeit ausgelagert und werden nur in zusammengefasster Form erläutert.

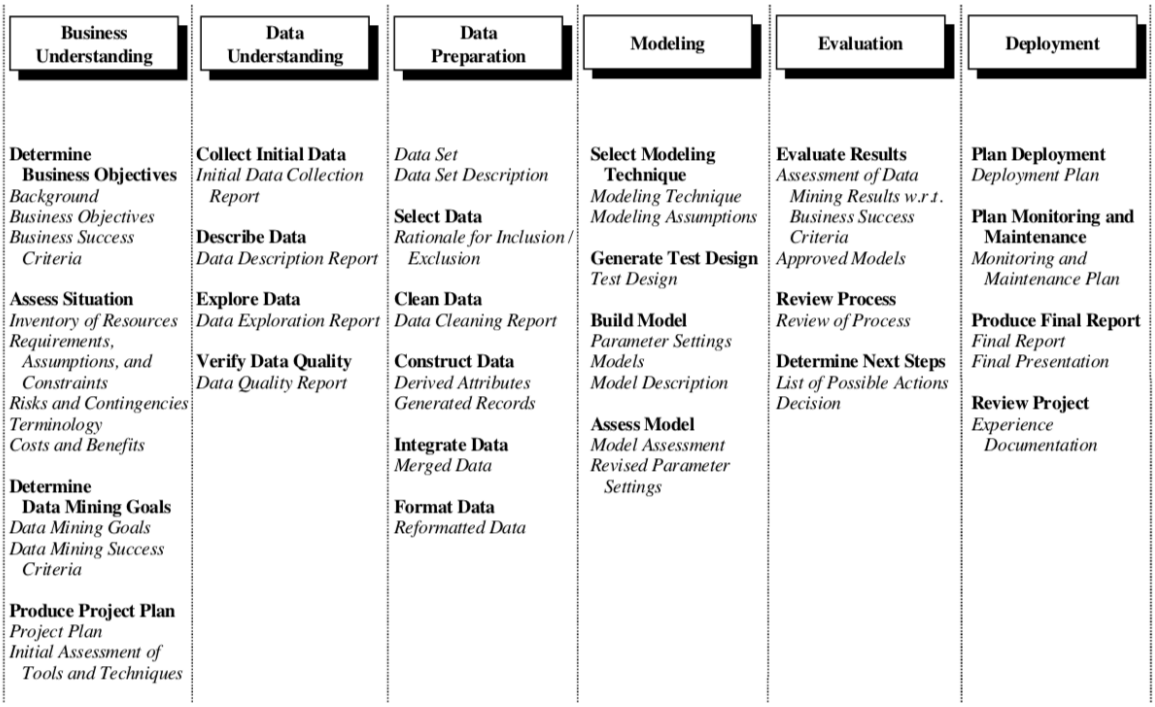


Abbildung 5: Phasen und generische Aktivitäten nach CRISP-DM

Das Modell dient vor allem dem Verständnis unterschiedlicher Fachebenen und unterstützt die Interaktionen der Projektbeteiligten entlang des Prozesses. Kunden werden dazu befähigt das erworbene Produkt und die Dienstleistung einzuschätzen, zu bewerten und das gewonnene Wissen effizienter im Unternehmen zu verbreiten. Hierdurch wird ein produktiverer Austausch zwischen Kunde und Dienstleister gefördert, da eine gemeinsame Sprache entsteht. Auf fachlicher Seite ist CRISP-DM für Datenanalysten hilfreich. Es bietet eine solide Orientierungsstruktur und vollständige Checklisten um Prozesslücken zu vermeiden. Zusammenfassend liegt die Aufgabe des Prozessmodells in der Unterstützung von Planung, Kommunikation und Dokumentation, den Grundpfeilern eines effektiven und effizienten Projektablaufes (Wirth & Hipp, 2000, p. 30).

Die Struktur der Folgekapitel entstammt den sechs Phasen von CRSIP-DM (siehe Abbildung 6). Die zugehörigen Abstraktionslevel werden in den Unterkapiteln indirekt durchlaufen. Im Anschluss werden die erläuterten Prozessschritte auf die Fallstudie angewandt.

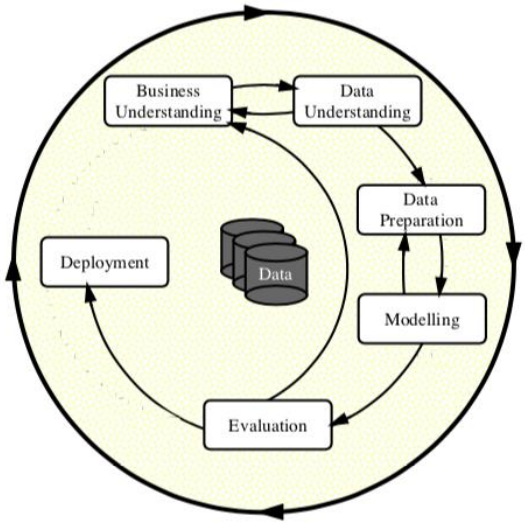


Abbildung 6: Phasen CRISP-DM (Wirth & Hipp, 2000, p. 33)

3.1 Verstehen des Unternehmens

Zum Kennenlernen der Projektziele und Anforderungen befasst sich die erste Phase mit der Ausgangslage des Unternehmens. Die branchen- und fachspezifische Aufgabenstellung muss in eine Data Mining Problematik und schlussendlich auch in eine ML-Problematik übersetzt werden. Ist dieser Schritt erfolgt, so lassen sich auf Basis dieser Grundlagen die notwendigen Arbeitsschritte in einem vorläufigen Projektplan festhalten.

3.1.1 Ermittlung der Unternehmensziele

Schritt 1: Annäherung durch Hintergrundinformationen

Die Ziele, unter welchen ein Data Mining Projekt für das betreffende Unternehmen steht, sind zu Beginn oft sehr vage. Auch wenn sich ein Mehrwert durch statistische Analyse und Maschinelle Lernprozesse auf die Ganzheit von unternehmensinternen Prozessen erahnen lässt, so ist der Ansatzpunkt jedoch schwer zu lokalisieren. Die Erfassung von großen Datenmengen eignet sich für ML Anwendungen, gerade ihr Umfang erschwert jedoch das Definieren von genauen Projektzielen. Zum effizienten Bearbeiten von Projektzielen empfiehlt Vermeulen zu priorisieren. Hierzu bietet er eine Klassifikation nach dem MoSCoW-Schema an (Vermeulen, 2018, p. 54) (siehe Abbildung 7). Generische Unternehmensziele hinsichtlich Daten aus Produktionsabläufen, lassen sich im Regelfall auf wirtschaftliche, Qualitäts- und Optimierungsziele zurückführen. Nach diesen Kategorien lassen sich erste Ziele im Zusammenhang mit der Datenanalyse definieren.

Must have	Requirements with the priority "must have" are critical to the current delivery cycle.
Should have	Requirements with the priority "should have" are important but not necessary to the current delivery cycle.
Could have	Requirements prioritized as "could have" are those that are desirable but not necessary, that is, nice to have to improve user experience for the current delivery cycle.
Won't have	Requirements with a "won't have" priority are thoseidentified by stakeholders as the least critical, lowest payback requests, or just not appropriate at that time in the delivery cycle.

Abbildung 7: MoSCoW Zielklassifikation (Vermeulen, 2018, p. 54)

Schritt 2: Messbarkeit durch Akzeptanzkriterien

Für die Messbarkeit der projektbezogenen Unternehmensziele werden Akzeptanzkriterien benötigt. Nach dem Zielorientierten Ansatz im Anforderungsmanagement können diese durch eine detaillierte und konkrete Beschreibung der Unternehmensziele aus Schritt 1 gewonnen werden (Prakash & Prakash, 2018, p. 12). Ein Akzeptanzkriterium sollte seiner Formulierung nach mit einem erfüllt oder nicht erfüllt zu bewerten sein. Es werden nur Zustände, keine Aufgaben definiert. Letztere werden erst durch die Definition von Data Mining Zielen festgelegt.

3.1.2 Lagebeurteilung

Projekte mit Machbarkeitsstudiencharakter stellen meist den Anfang von weiterführenden Data Mining Projekten. Betrachtet man diese Studien daher als gesonderte Projekte mit relativ kurzfristiger Laufzeit und verhältnismäßig geringem Aufwand, rentieren sich ausführliche Projektvorbereitungen kaum. Die folgenden drei Schritte können in diesem Fall in einer Zusammenfassung behandelt werden.

Schritt 1: Auflisten von Ressourcen, Annahmen und Beschränkungen

Zu Beginn des Projektes kann zunächst eine Bestandsliste über vorhandene und benötigte Ressourcen erstellt werden. Diese reichen vom Humankapital und dessen Fähigkeiten bis hin zu Hard- und Software, auf welche sich das Data Mining stützt. Wird eine wirtschaftliche Betrachtung verlangt, müsste das Budget definiert werden. Hierzu können Methoden aus dem klassischen Projektmanagement angewandt werden (Litke, 2007, p. 126).

Schritt 2: Risiken und Eventualitäten

Während der Auflistung von Ressourcen können bereits Risiken identifiziert werden, welche anschließend explizit bewertet werden. Ein genaues Verständnis über mögliche Problemfälle lässt eine Wenn-Dann Definition im Projektplan zu und sichert dessen flexibles Reaktionsverhalten. Handlungsalternativen können in präventive, korrektive und selbsttragende Maßnahmen unterschieden werden (DIN IEC 62198, 2002, pp. 15-17). Erfahrungswerte und Lessons Learned von gleichartigen Projekten können hier hilfreich sein.

Schritt 3: Kosten und Nutzen

Nach Schritt 1 und 2 können, je nach wirtschaftlicher Betrachtung bereits abgeschätzte Kostenpunkte dem erwarteten Nutzen gegenübergestellt werden. Eine direkte monetäre Gegenrechnung ist zum Projektstart sehr schwierig, da die Ungewissheit über Qualität und Tragweite des Data Mining diese erschweren. In diesem Fall ist der Nutzen aus den Projektzielen abzuleiten, welcher bei Machbarkeitsstudien unter anderem durch die hochklassige Aufbereitung von Informationen gegeben ist. Weitere sind projektspezifisch zu definieren.

3.1.3 Festlegen von Data Mining Zielen

Übersetzung aus Unternehmenszielen

Nachdem die Akzeptanzkriterien eine Erfolgsmessung definiert haben, lassen sich aus ihnen konkrete Aufgabenstellungen definieren. Im Hinblick auf Data Mining Projekte ergeben sich Zielfragen, welche durch die Bearbeitung der Aufgaben zu beantworten sind. Es muss z.B. eine begründete Antwort darauf gefunden werden, ob die Datengrundlage in ihrer Menge, Zusammensetzung und in ihrem Informationsgehalt den Kriterien einer sinnvollen ML Anwendung entspricht. Die erfolgreiche Umsetzung solcher Data Mining Ziele ist schwer zu quantifizieren. Zunächst besitzt sowohl das Aufzeigen von Mangel, als auch der Nachweis von Potential einen Informationsgehalt. Je nach Kontext der Problemstellung kann dies eine wertvolle Erkenntnis darstellen. Es geht daher in erster Linie um die möglichst vollständige Beantwortung der Zielfragen hinter den Aufgabenstellungen.

3.1.4 Erstellen des Projektplans

Der Projektplan orientiert sich in seiner Struktur am Referenzmodell nach CRSIP-DM. Wie schon im Kapitel zur Lagebeurteilung erwähnt, sind Machbarkeitsstudien auf Grund ihres geringen Umfangs für ausführliche Planungsarbeiten ungeeignet. Einfache

Projektablaufpläne nach Petri-Netz oder BPMN Konvention können als Kompromisslösung mit übersichtsvorteil genutzt werden.

3.2 Aufbau eines Datenverständnisses

3.2.1 Sammeln von initialen Daten

Zu Beginn der Datenbeschaffung wird die Struktur und Funktionsweise der vorhandenen Datenquelle untersucht. Handelt es sich um einen Speicher, welcher große Datenmengen in einem einheitlichen, nativen Datenformat bereitstellt, so wird auch von einem Data Lake gesprochen. Charakterisiert wird ein Data Lake durch seine Schema-On-Read-Architektur (Vermeulen, 2018, p. 4). Diese weist jedem gespeicherten Element eine spezifische Identifikation und eine Menge an Metadaten zu, welche von einer Suchanfrage mit einem individuellem Ad-hoc-Schema für den Datenzugriff genutzt wird (siehe Abbildung 8). Ein solcher Aufbau unterscheidet sich von einer eher starren, Schema-On-Write-Architektur. Dieses Data-Warehouse-Prinzip speichert die Daten in vordefinierten Datenbanken und Schemata, welche dem Ausleseprozess eine fest vorgegebene Struktur geben. Auch hier bilden Metadaten eine Orientierung, sie können in ihrer Komposition jedoch nicht verändert werden.

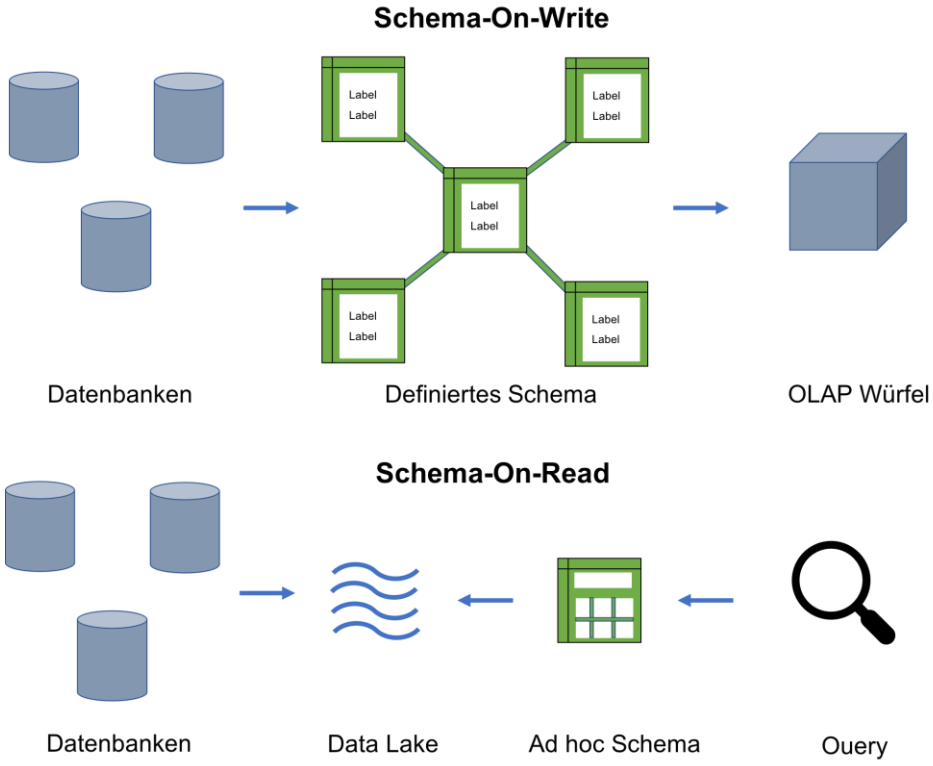


Abbildung 8: Schaubild Datenbank Architekturen

Unabhängig von der Architektur einer Datenquelle, wird auf Metadaten zurückgegriffen um die für die Data Mining Ziele relevanten Daten zu beziehen. Diese Metadaten besitzen in ihrer allgemeinsten Variante drei Dimensionen: Objekt/Person, Ort und Zeit. Liegen große Mengen von Mess- und Prüfdaten aus Produktionsprozessen vor, so sind die Metadaten also nach dem gesuchten Produktionsobjekt, seiner räumlichen oder reihenfolgespezifischen Kennzeichnung und dem betrachteten Zeitraum zu filtern. Das Produktionsobjekt ist an dieser Stelle absichtlich allgemein definiert. Dieses Objekt kann von einer gesamten Fertigungsreihe bis hin zu einem einzelnen Sensor/Prüfschritt reichen. Da es an dieser Stelle jedoch noch nicht um eine konkrete Datenanalysen geht, orientiert man sich an dem größtmöglichen gemeinsamen Nenner, um einen möglichst vollständigen Datenpool für die weitere Verarbeitung zu erhalten. Kommunikationsunterstützend und zur Gewährleistung der Nachvollziehbarkeit sollten alle angewandten Schritte und Filter einheitlich dokumentiert werden. Dies gilt für alle weiteren Schritte, welche für das Datenverständnis grundlegend sind.

3.2.2 Datenbeschreibung

Ist die vorläufige Datenbeschaffung abgeschlossen, so kann die für das Datenverständnis essentielle Datenbeschreibung beginnen. Hier werden nicht nur Eckdaten zum Format angegeben, sondern die Gründe für die Wahl des Formates und die verwendete Datenstruktur erläutert. Die charakteristische Strukturiertheit und Vielschichtigkeit einer Produktion, spiegeln sich in den aufgezeichneten Daten wieder. Es ist daher ratsam viel Wert auf eine gründliche Erklärung der verwendeten Bezeichnungen und der Hierarchie hinter den einzelnen Datenpunkten zu legen. Die Logik des zu Grunde liegenden Produktionsablaufes kann dazu verwendet werden diese Erklärung verständlicher zu gestalten.

3.2.3 Datenexploration

Der Schritt der Datenexploration ist ein zusammengefasster Vorgriff auf die Schritte der Datenvorverarbeitung und Modellierung. Wie der Name verdeutlicht, geht es um eine erste, explorative Analyse mit dem Zweck das Datenverständnis zu erweitern. So sind z.B. trotz einer ausführlichen Datenbeschreibung die inhaltlichen Zusammenhänge zwischen den Merkmalen schwer aus den reinen Zahlenwerten zu erfassen. Wie in Kapitel 2.4 erwähnt, lässt sich durch Visualisierungen eine wesentlich zugänglichere Interpretation erarbeiten. Je nachdem müssen aber selbst für diese zunächst Vorverarbeitungsschritte durchlaufen werden. Dies zeigt, dass auch bei der Datenexploration auf die Reihenfolge der zu Grunde

liegenden Phasen und deren Arbeitsschritte geachtet werden sollte. Eine solche vorgreifende Exploration findet falls notwendig innerhalb der Phasen statt, da sie durch das Testen von Hypothesen und Prototypen gewisse Handlungsentscheidungen verkürzen und fundieren kann. Während der ersten Datenexploration werden die informationstechnischen Werkzeuge, einschließlich Intrigierten Entwicklungsumgebungen (IDEs) und Grafischen Benutzeroberflächen (GUIs) aufgesetzt.

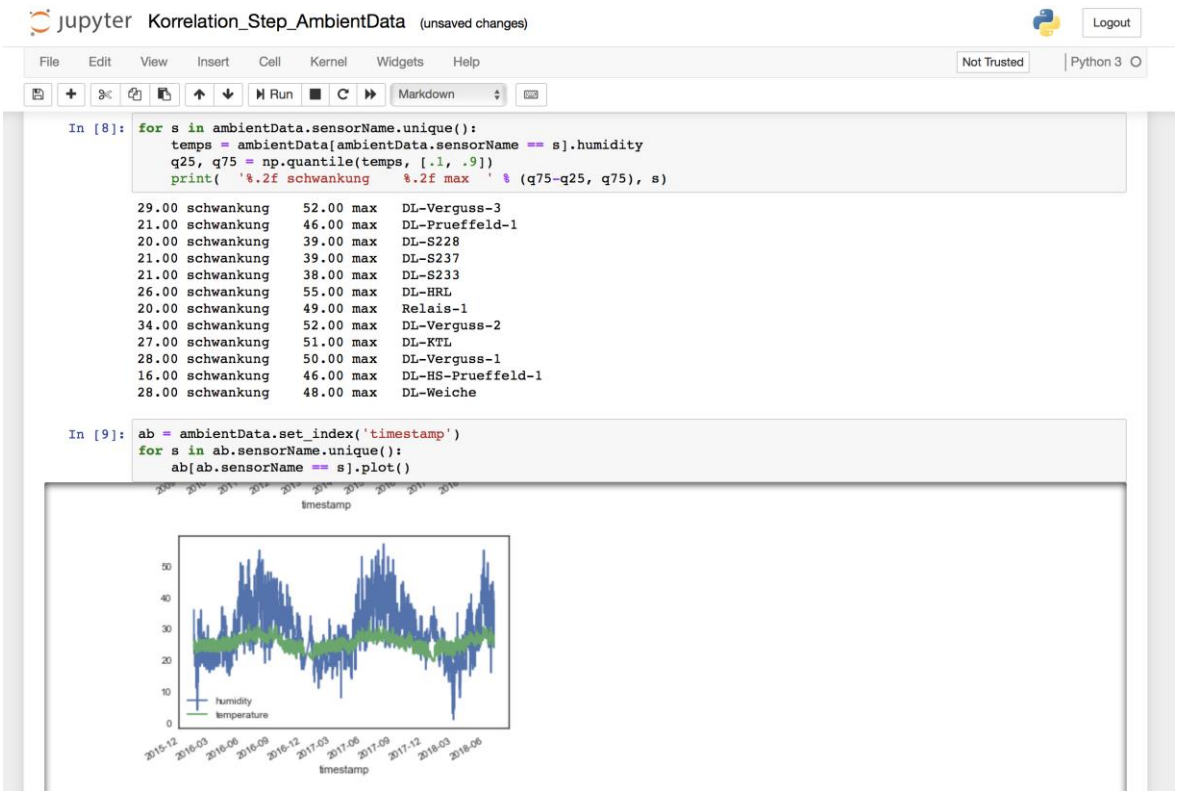


Abbildung 9: Beispielauszug aus der Jupyter IDE – Projekt (Kapitel 4)

Da die Kapitel 3.3 und 3.4 die Arbeitsschritte der Datenexploration abdecken, wird an dieser Stelle auf eine Erläuterung dieser verzichtet. Ein essentieller Bestandteil der Datenexploration ist die grafische Gegenüberstellung einzelner Merkmale. Auf Grund der strikten Grenzwerte für Messwerte sind auf diese Weise Muster und Ausreißer im zeitlichen Verlauf besonders gut auszumachen (siehe Abbildung 9).

3.2.4 Verifikation der Datenqualität

Eine zuverlässige Informationsextraktion verlangt eine verifizierte Datenbasis. Die Verifizierung überprüft die Datenbasis an Hand der sechs Qualitätsdimensionen nach Vermeulen (Vermeulen, 2018, pp. 277-279).

1) Vollständigkeit: Das Verhältnis von fehlerhaften und fehlenden Einträgen zu der Gesamtzahl. Der Kontext ist bei der Bestimmung von mangelhaften Einträgen zu berücksichtigen, da es ansonsten zu einer inkorrekten Einordnung kommen kann.

2) Einzigartigkeit: Als Fragen innerhalb von drei Ebenen zu betrachten. In wie weit unterscheiden sich einzelne Werte innerhalb eines Merkmals? Haben diese Werte Ähnlichkeiten zu Werten anderer Merkmale? Welche Merkmale treten in welcher Häufigkeit in den Datenquellen auf?

3) Zeitliche Kontinuität: Unterscheidung zwischen Perioden stabiler und instabiler Datenerfassung.

4) Gültigkeit: Festhalten von Abweichungen in Bezug auf den genutzten oder etablierten Standard.

5) Genauigkeit: Beurteilt wie akkurat die Messungen und beschreibenden Merkmale eine Person oder ein Objekt darstellen. Auch hier ist der (unternehmensspezifische) Kontext entscheidend.

6) Konsistenz: Als Maß für die Änderung der Datenstruktur über einen Zeitraum oder über die Datenquellen hinweg. Die Veränderung sollte an Hand eines festgelegten Musters gemessen werden.

Die strikten Standards und Arbeitspläne einer Produktion bedeuten, dass bei der Untersuchung der Datenqualität von Mess- und Prüfdaten ein besonderes Augenmerk auf Einzigartigkeit, zeitliche Kontinuität und Konsistenz gelegt wird. Gibt es in diesen Dimensionen überdurchschnittlich viele oder auch sehr wenige Abweichungen, so lassen sich essentielle Aussagen über die Datenqualität und damit den extrahierbaren Informationsgehalt treffen.

3.3 Datenvorverarbeitung

Datensatzbeschreibung:

Bevor es an die eigentliche Datenverarbeitung geht, ist es hilfreich sich einen Überblick über den zu bearbeitenden Datensatzes zu verschaffen. Man beachte den Unterschied zwischen der Datenbeschreibung aus Kapitel 3.2.2 und der Datensatzbeschreibung. Es wird nun ein spezifischer Datensatz und nicht die Gesamtheit der Daten betrachtet. Da der Datenbeschreibungsreport schon Informationen zu Inhalt, Format und Struktur der Daten im

Allgemeinen liefert, ist an dieser Stelle auf die Umstände der spezifischen Auswahl des Datensatzes einzugehen.

Da Produktionsabläufe und somit auch die resultierenden Datenstrukturen streng hierarchisch aufgebaut sind, lassen sich Datensätze leicht aus einer spezifischen Ebene dieser Struktur entnehmen. So eignen sich Kennzeichnungen von Bauteilgruppen, Maschinentypen und Prozessabläufen als Ansatz um einen spezifischen Datensatz zu entnehmen. Eine solche Selektion sollte im Zusammenhang mit den Data Mining Zielen begründet dargelegt werden. Erscheint die Auswahl unpassend, so kann je nach benötigter Dimension ein Datensatz aus der nächst höheren oder niedrigeren Hierarchieebene entnommen werden. Es wird empfohlen den Datensatz diesbezüglich nicht nur auf inhaltliche Übereinstimmung mit den Analysezielen zu testen, sondern auch auf eine ausreichende Menge an Datenpunkten und deren möglichst konsistente zeitliche Abfolge.

3.3.1 Datenselektion

Darlegung von Ausschluss und Einbeziehung:

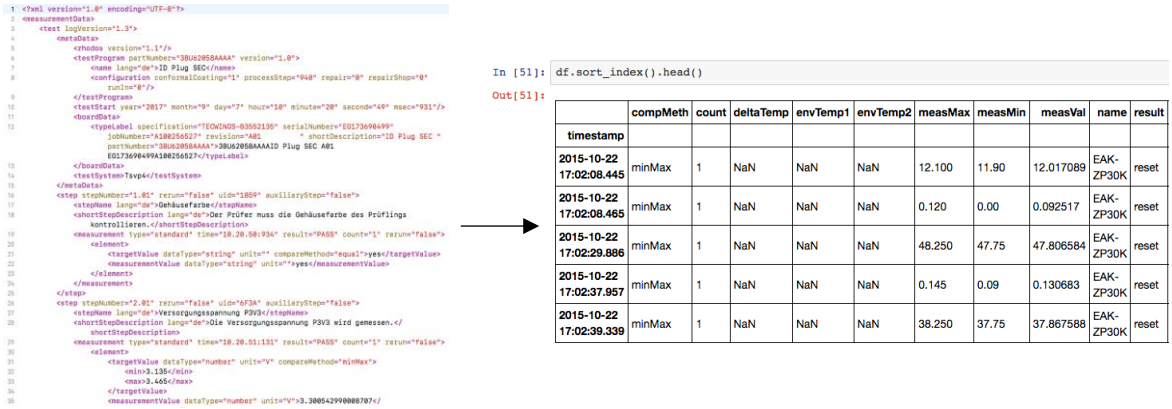
Die Arbeit mit großen Datenmengen erfordert in den häufigsten Fällen eine Selektion der zu analysierenden Variablen. Auch wenn durch die Auswahl eines Datensatzes bereits eine gewisse Selektion vorgenommen wurde, so ist diese nur der erste Schritt auf dem Weg eine verwertbare Datengrundlage für die Analyse zu schaffen. Gibt es Variablen, welche unvollständige, fehlerhafte oder gar irrelevante Daten im Datensatz aufweisen, so kann dies den Analyseprozess negativ beeinflussen. Die Reihe von Problemtypen für ML Verfahren aus Kapitel 2.2 sind eine mögliche Konsequenz solcher Unzulänglichkeiten, falls diese nicht adressiert werden. Bevor eine aufwendige Bereinigung von solchen Variablen vorgenommen wird, sollte daher zunächst zwischen Ausschuss und Einbeziehung gewisser Variablen entschieden werden.

Im Falle von Sensordaten aus Produktionsprozessen liegt durch akribische, technische Planung und Kontrolle im Regelfall eine geringere Anzahl von Variablen mit irrelevanter oder problematischer Datengrundlage vor. Im Gegensatz dazu kann es eher vorkommen, dass im Verlauf der Modellierung und Analyse ein Mangel an Variablen festgestellt wird. Daher ist es empfehlenswert bei der Datenselektion nach dem Vorsichtsprinzip zu handeln, so dass sich eine Aufbereitung ggf. aufwendiger gestaltet, die inhaltliche Vollständigkeit aber gesichert wird.

3.3.1 Konstruktion neuer Datenstrukturen

Generierte Datensätze:

Das native Dateiformat des Datensatzes ist für die zukunftssichere und hierarchisch korrekte Speicherung von Daten optimiert. Ein Data Mining Prozess profitiert darüber hinaus von Datenstrukturen, aus denen sich übersichtliche Darstellungen generieren lassen und auf denen sich Operationen an den Daten unkompliziert bewerkstelligen lassen. Beide Anforderungen lassen sich schwer vereinen und daher ist es ratsam aus dem nativen Dateiformat eine neue Datenstruktur für das Data Mining zu konstruieren. Dieser Prozess wird von einer eigens angepassten Funktion, dem Parser, vollführt. Der Parser liest die benötigten Einzelwerte des nativen Formates aus und überführt sie in das neue Format.



timestamp	compMeth	count	deltaTemp	envTemp1	envTemp2	measMax	measMin	measVal	name	result
2015-10-22 17:02:08.445	minMax	1	NaN	NaN	NaN	12.100	11.90	12.017089	EAK-ZP30K	reset
2015-10-22 17:02:08.465	minMax	1	NaN	NaN	NaN	0.120	0.00	0.092517	EAK-ZP30K	reset
2015-10-22 17:02:29.886	minMax	1	NaN	NaN	NaN	48.250	47.75	47.806584	EAK-ZP30K	reset
2015-10-22 17:02:37.957	minMax	1	NaN	NaN	NaN	0.145	0.09	0.130683	EAK-ZP30K	reset
2015-10-22 17:02:39.339	minMax	1	NaN	NaN	NaN	38.250	37.75	37.867588	EAK-ZP30K	reset

Abbildung 10: Auszüge einer XML- und Pandas-Struktur – Projekt (Kapitel 4)

Im Falle von XML standardisierten Mess- und Prüfdaten werden die Informationen vom Parser in ein praktikables, meist tabellarisches Format überführt (siehe Abbildung 10). Auf Grund der einzigartigen Natur eines jeden Data Mining Projektes, muss ein Parser problemspezifisch angepasst werden. Während dieser Spezifizierung kann die Datenselektion in diesen Arbeitsschritt mit eingebunden werden. Der Parser kann durch seine selektive Aus- und Einlese diesen Arbeitsschritt automatisieren. Auch zukünftige Selektionsänderungen können auf diese Weise schnell eingearbeitet und umgesetzt werden. Der praktische Hintergrund dieser Empfehlung lässt sich in Kapitel 4.3.1 einsehen.

Abgeleitete Attribute:

Zur Anreicherung der Informationslage können die vorhandenen Variablen des Datensatzes auf (tiefere) Zusammenhänge untersucht werden. Es lassen sich z.B. durch zeitliche Aggregation indirekte Informationen über neue Sachverhalte herausarbeiten. Versteht man die Variablen eines Datensatzes als dessen Attribute, so stellen die aus den indirekten

Informationen abgeleiteten Attribute eine zusätzliche Datengewinnung dar. Die Suche nach indirekten Informationen findet meist in der Datenexploration statt und wird exemplarisch in Kapitel 4.2.3 behandelt.

3.3.2 Datenbereinigung

Nach der Konstruktion neuer Datenstrukturen mit leichterer Handhabung, können besonders fehlerbehaftete Variablen effektiver erkannt und bereinigt werden. Eine der Hauptaufgaben der Datenbereinigung ist dafür zu sorgen, dass der Datentyp und die Wertebezeichnungen einer Variablen einheitlich sind. Gibt es fehlerhafte oder fehlende Werte, muss entschieden werden, ob der Wert entfernt, durch einen Repräsentanten ersetzt oder gar ignoriert werden soll (Petersohn, 2005, p. 62). Diesbezüglich ist zu beachten, dass auch fehlende Werte für die Vollständigkeit der Datenreihe relevant sein können und einen einheitlichen Datentyp benötigen. Ein dafür gängiger Datentyp ist das numerische NaN, welches von Funktionen zur Datenbearbeitung direkt erkannt und eigens behandelt werden kann (VanderPlas, 2016, p. 141). Der Aufwand und die Dringlichkeit einer gewissenhaften Filterung und Bereinigung ist nicht zu unterschätzen. Beseitigt man die Mängel nicht, kann es während der Analyse zu Problemen kommen, welche eine erneut zeitaufwendige Suche und Bereinigung nach sich ziehen.

Numerische Mess- und Prüfdaten sind an ein hohes Maß an Präzision gebunden. Der Austausch von fehlerhaften oder fehlenden Werten durch Repräsentanten wie Mittel- oder Durchschnittswerte können den Aussagengehalt verfälschen. Das Entfernen bzw. Löschen eines Wertes ist aus dem gleichen Grund zu hinterfragen. In bestimmten Anwendungsfällen ist der Ausgleich durch NaN-Werte eine plausiblere Technik zur Bereinigung (siehe Kapitel 4.3.1).

3.3.3 Integration von Daten

Durch die Auswahlkriterien bei der Entnahme des Datensatzes aus der Datengrundlage wird die Informationslage begrenzt. Es kommt zu einem Trade-off zwischen möglichst vollständiger Information und handhabbarer Informationslast. Es ist daher ggf. notwendig zunächst kleinere, spezifische Datenpakete zu verarbeiten und diese dann im Nachhinein zusammenzuführen. Gleiches gilt für Datensätze aus unterschiedlichen Quellen, mit unterschiedlichem Format und Inhalt. Es erscheint sinnvoll diese zunächst in das einheitliche Zielformat für das Data Mining zu überführen und Sie im Anschluss nach Selektion, Bereinigung und Konstruktion zusammenzuführen.

3.3.4 Datenformatierung

Sind die inhaltliche Vollständigkeit und Genauigkeit nach Bereinigung und Integration gesichert, kann die Datenstruktur in die für das ML benötigte Form gebracht werden. Diese Formatierung ist nicht mit dem Wechsel zu einem anderen Dateiformat zu vergleichen. Es handelt sich um die Veränderung der strukturellen Beziehung zwischen den Variablen/Attributen. In einem tabellarischen Zusammenhang stellt sich die Frage nach welcher Variablen indexiert werden soll. Diese Indexierung bestimmt in welcher Weise die Daten gelesen und interpretiert werden. So kann z.B. eine Tabelle mit Kundendaten nach Alter, Geschlecht, usw. indexiert werden und stellt dementsprechend ganz unterschiedliche Inhaltliche Bezüge und Körnigkeiten innerhalb der Datengrundlage dar. Weiterhin ist auch die Wahl der Attribute, welche die Spalten darstellen, für die Form relevant. Die Umsortierung der Tabelle nach einem schon existierenden Spaltenattribut nennt sich Pivotisierung. Indexierung und Pivotisierung sind für die Vorverarbeitung der Daten im Hinblick auf ihren Einsatz in ML-Verfahren entscheidend. Durch die Umstrukturierung des Datensatzes kreieren sie die Spalten der Labels und Features.

3.4 Modellierung

Nach einer meist aufwendigen Vorbereitungsphase kommt es während der Modellierung nun zur eigentlichen Analyse und Suche nach Lösungskonzepten. Der im Folgenden dargestellte Prozess wird linear für ein undefiniertes ML-Verfahren durchlaufen werden. In der Realität gibt es häufig mehrere Verfahren, welche diesen Prozess parallel und in unterschiedlicher Reihenfolge durchlaufen. Wie in der Einleitung zur CRISP-DM Methodik erwähnt, kann es zu einem Vor- oder Rückgriff auf gewisse Arbeitsschritte kommen. Es kann z.B. während der Modellierungsphase erkannt werden, dass eine Datengrundlage erneut die Bearbeitungsphase durchlaufen muss um für ein gewisses ML-Verfahren geeignet zu sein.

3.4.1 Auswahl einer Modellierungstechnik

Schritt 1: Auswahl eines Verfahrens

Die erklärten Ziele eines Data Mining Projektes sind ein erster Ansatzpunkt um eine engere Auswahl für mögliche ML-Verfahren zu erstellen. Je nach Projekt lässt sich aus der Aufgabenstellung ggf. schon die Art des Verfahrens erkennen, z.B. ob eine regressionsbasierte Vorhersage oder eine Klassifikation von Daten gefordert ist. Diese Verfahrensklassen des überwachten ML wurden bereits in Kapitel 2.3 behandelt und sind recht eindeutig in ihrer Arbeitsweise. Ist ein Data Mining Ziel jedoch nicht eindeutig mit der

Verfahrensweise verknüpf oder besitzt explorativen Charakter, so empfiehlt es sich zunächst die in dem Datensatz enthaltenen Muster zu untersuchen. Sind eine große Anzahl an Features im Datensatz vorhanden, kann sich dies jedoch auf Grund der in Tabelle 6 aufgezeigten Konsequenzen als schwierig erweisen.

„Fluch der hohen Dimensionen“: Die Berücksichtigung vieler Features resultiert in komplexen Modellen, welche wiederum zu Overfitting neigen (siehe Kapitel 2.2). Kann diese Mehrdimensionalität nicht durch eine besonders große Datengrundlage kompensiert werden, lassen sich statistisch signifikante Aussagen nur beschränkt treffen.
Längere Trainingszeiten: Mit der zunehmenden Anzahl an Features steigt auch die einzuplanende Zeit zum Einlernen der ML-Verfahren. Eine SVM muss z.B. mit einer höheren Anzahl an Dimensionen komplexere Funktionen für die Raumtrennung berechnen (siehe Kapitel 2.3).
Verlust der Übersichtlichkeit und Interprätierbarkeit: Werden Datenpunkte durch eine große Anzahl an Features beschrieben, lassen sich die Zusammenhänge und Muster innerhalb der Daten nur schwierig erkennen und deuten.

Tabelle 6: Mögliche Konsequenzen eines multidimensionalen Datensatzes

Durch die große Anzahl an erfassten Messwerten ziehen Datensätze aus Produktionsprozessen diese Problematik häufig nach sich (siehe Kapitel 4.4.1). Eine explorative Analyse kann dabei helfen Ansätze für ein ML-Verfahren zu finden. Zeitreihenanalysen, Histogramme und Korrelationsanalysen sind klassische Mittel um relevante Features zu identifizieren und erste Modellansätze zu entwickeln. In Kapitel 2.4 wurde auch die besondere Eignung von unüberwachten ML-Verfahren zur explorativen Analyse dargelegt. Mit ihren Fähigkeiten zur Abweichungsanalyse und Segmentierung können sie unterstützend verwendet werden. Verfahren wie die PCA helfen bei mehrdimensionalen Datensätzen sich einen Überblick über vorhandene Muster und die entscheidendsten Features zu verschaffen. Trotz aufbereitender Analysen lassen sich passende ML-Verfahren nicht immer im Vorhinein identifizieren. Es muss daher nicht selten eine breite Auswahl an Verfahren den Modellierungsprozess durchlaufen, um erst am Ende durch die Modellbeurteilung selektiert zu werden.

Schritt 2: Modellierungsannahmen

Die getroffene Vorauswahl an ML-Verfahren zieht gewissen Bedingungen und Annahmen nach sich, welche der Datensatz erfüllen sollte. Eine zu testende Bedingung von ML-Verfahren ist die verlangte Skalierung (Petersohn, 2005, p. 64). Der Datensatz muss meist kardinalskaliert, oder zumindest ordinalskaliert sein. Eine Nominalskalierung lässt keine Distanzen oder gewichtete Unterscheidungen zwischen den Datenpunkten zu. Neben der Skalierung können noch weitere Annahmen wie minimale Stichprobengröße und spezifische Formatierung relevant sein. Im optimalen Fall wurden diese jedoch schon während der Datenvorverarbeitung antizipiert und berücksichtigt. Können die Annahmen trotz Vorverarbeitung nicht erfüllt werden, schränkt dies die Auswahl der Verfahren möglicherweise weiter ein.

Numerische Daten aus Produktionsprozessen sind von Grund auf kardinalskaliert und können daher ohne Hindernis in den meisten ML-Verfahren verwendet werden. Es gibt durch die Ergebniskennzeichnungen (z.B. pass/fail) und die Metadaten ggf. aber auch rein kategorische Labels und Features, welche so nicht akzeptiert werden. Diese müssen folglich in die benötigte (kardinale) Skalierung übersetzt werden. In Sprachen wie Python gibt es Bibliotheken mit Funktionen, welche diesen Arbeitsschritt automatisiert und ohne aufwendige manuelle Arbeit umsetzen können.

3.4.2 Erstellung eines Test-Designs

Für die spätere Modellbeurteilung ist das Test-Design ein vorgelagerter Arbeitsschritt zur Schaffung einer Vergleichsbasis. Es werden Anwendungsfälle (engl. test cases) mit fest definierten Gütekriterien und Erfolgskennzahlen geschaffen, welche später zur fundierten Bewertung des fertigen Modells herangezogen werden. Der Nutzen des Test Designs liegt also im Vergleich von Soll- und Ist-Zustand (Copeland, 2004, p. 2).

Festlegung von Gütekriterien:

Probleme wie Overfitting und Underfitting wurden in Kapitel 2.2 als Ursachen für die mangelhafte Leistung eines ML-Verfahrens behandelt. Mit Hilfe von Metriken lässt sich das Vorhandensein oder auch die Abwesenheit solcher Probleme erkennen. Die in Kapitel 2.2 erwähnte Aufteilung des Datensatzes in Trainings- und Testdaten ermöglicht die praxisbezogene Leistungsbeurteilung eines ML-Verfahrens auch ohne neue Datenzufuhr. Mit Hilfe randomisierter Zusammensetzungen und dem Vergleich mehrerer unterschiedlicher Trainings-Iterationen, kann diese Aufteilung erweitert werden. Diese als (verschaltete) Kreuzvalidierung bezeichnete Testerweiterung kann einer möglichen

Voreingenommenheit entgegenwirken (Müller & Guido, 2017, pp. 246-260). Der Einsatz solcher umfangreichen Tests muss jedoch auf Praxistauglichkeit untersucht werden, da Ansätze wie die Kreuzvalidierung z.B. aus Laufzeitgründen nicht immer rentabel sind. Nachdem das trainierte ML-Verfahren Ergebnisse für den Testdatensatz vorhersagt hat, lassen sich diese mit den wahren Ergebnissen vergleichen. Für die Berechnung von Gütekennzahlen nutzen spezifische Metriken das Verhältnis zwischen den richtig und falsch vorhergesagten Werten. In Tabelle 7 ist eine Auswahl der besonders gängigen Metriken für Klassifikationsverfahren (Müller & Guido, 2017, pp. 260-281) zu sehen.

$$Genauigkeit = \frac{Anzahl\ der\ richtig\ vorausgesagten\ Labels}{Anzahl\ aller\ Datenpunkte}$$ (engl. Accuracy) Eine simple, aber viel verwendete Metrik für die meisten Klassifikationsverfahren.
Konfusionsmatrix: $\begin{pmatrix} tc1 & fc1 \\ fc2 & tc2 \end{pmatrix}$ Gelesen: tc1/fc1 = richtig/falsch klassifizierte Datenpunkte der Klasse 1. Hier nur binäre Klassenunterscheidung, die quadratische Matrix kann um zusätzliche Klassen/Labels erweitert werden. Übersichtlichkeit und tiefere Einsicht bei mehreren Klassen sind ein Vorteil der Matrixdarstellung.

Tabelle 7: Auswahl gängiger Metriken für die Klassifikation Teil 1

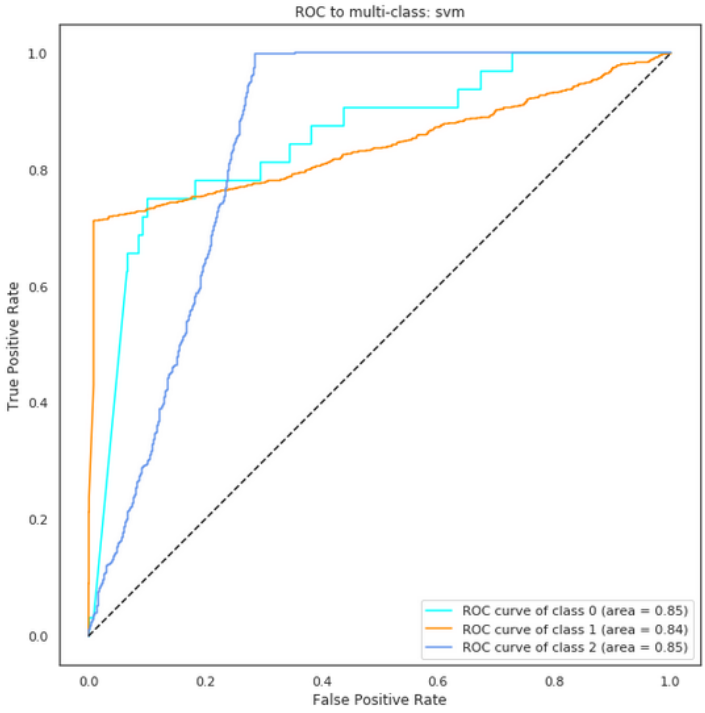
Precision: $\frac{tc1}{tc1+fc1}$ Die Präzision der Vorhersage bzgl. einer Klasse

Recall: $\frac{tc1}{tc1+fc2}$ Die Wiedererkennungsrate oder auch Effizienz bzgl. einer Klasse

F1 Wert: $\frac{2 * precision * recall}{(precision + recall)}$ Ein harmonisches Mittel zwischen beiden Werten

Der Einsatz kann sowohl für die allgemeine Betrachtung, als auch für die klassenspezifische Optimierung des Verfahrens genutzt werden. Zu beachten: Es kommt immer zu einer binären Betrachtung. Bei mehreren Labels werden die falschen Vorhersagen zu einer Klasse zusammengefasst.

ROC (Receiver Operating Characteristic) – Kurve:



Gegensätzlich zu Precision und Recall wird bei dieser grafischen Analyse die Ausfallrate $\left(\frac{fc1}{fc1+tc2} : false\ positive\ rate\right)$ und Sensibilität $\left(\frac{tc1}{tc1+fc1} : true\ positive\ rate\right)$ gegeneinander abgetragen. Auch hier liegt immer eine binäre Betrachtung vor.

Tabelle 8: Auswahl gängiger Metriken für die Klassifikation Teil 2

Anwendungsfall und Erfolgskennzahlen

Bei der Entwicklung von Anwendungsfällen ist es hilfreich einen Fokus auf die existierenden, industriespezifischen Kennzahlen zu legen. In der Praxis sind solche Kennzahlen auch als KPIs (engl. Key Performance Indicator) geläufig. Nach der Identifikation der Arbeitsschritte mit einem Einfluss auf die KPIs, kann der theoretische Mehrwert einer ML-Anwendung innerhalb dieser Prozesse untersucht werden. Ist ein potentieller Ansatzpunkt gefunden, können Szenarien mit konkreten Kennzahlen für das trainierte ML-Modell entworfen werden.

Für den Fall einer Produktion kann dies wie folgt aussehen:

1) Recherche oder Definition von KPIs für einen spezifischen Arbeitsbereich
2) (Hypothetisches) Szenario mit einer ML-Anwendung zur effizienteren Gestaltung der betrachteten Arbeitsschritte
3) Definition einer verlangten Mindestgröße für die Güte des ML Modells aus den festgelegten Metriken. Berechnung der damit einhergehenden Verbesserung der KPIs aus Schritt 1) als Zielwert für die Umsetzung.

3.4.3 Aufsetzen eines ML-Modells

Informationstechnische Werkzeuge:

Nachdem der Ansatzpunkt für die Anwendung eines ML-Verfahrens durch das Test Design gegeben ist, kann die Modellierung umgesetzt werden. Für die Modellierung können zwei Arten von informationstechnischen Werkzeugen verwendet werden. Im Regelfall eignet sich dafür die Entwicklungsumgebung, in welcher schon die explorativen und datenverarbeitenden Arbeitsschritte durchlaufen wurden. Eine alternative Modellierungsoberfläche bieten kommerzielle oder Open Source Plattformen für das Data Mining. Diese Softwarepakete vereinfachen durch automatisierte Extraktions-, Aufbereitungs- und Manipulationsfunktionen die Arbeit mit der Datengrundlage (Oettinger, 2017, p. 42). Sie können jedoch nicht als Allheilmittel für den gesamten Data Mining Prozess gesehen werden, da sie entweder zum einmaligen Abtasten der Daten oder für stark integrierte Datenströme verwendet werden.

Zwischen den beiden Werkzeugklassen kann ein Vergleich zu dem Unterschied zwischen Individualsoftware und Standardsoftware gezogen werden. Während eine Plattformlösung ein schnelles, unkompliziertes Aufsetzen und Testen von ML-Verfahren ermöglicht, so

bietet die Modellierung in der Entwicklungsumgebung eine problemspezifischere Anpassung an den Anwendungsfall. Je nach Menge und Komplexität der zu testenden ML-Verfahren sollte der Trade-Off zwischen gewonnenen Erkenntnissen und investierter Arbeitszeit beachtet werden. Gegebenenfalls können beide Ansätze auch parallel und für bestimmte Modelle selektiv betrieben werden. Das in Kapitel 4 präsentierte Projekt nutzt beide Modellierungsmethoden um eine Vergleichsmöglichkeit für Ergebnisse zu schaffen.

Training eines ML-Verfahrens und Schnittstellendesign:

Das Ziel der Modellierung ist das Trainieren eines speziell selektierten ML-Verfahrens. Mit seiner auf den Anwendungsfall spezifizierten Prognostizierfähigkeit, simuliert das trainierte ML-Verfahren ein aus der Realität abstrahiertes Modell. Den praktischen Rahmen stellt dabei eine programmierte Schnittstelle, welche sowohl für das Training des Modells, als auch für die praktische Anwendung dessen verantwortlich ist. Die Schnittstelle deckt dabei folgende Aufgabenbereiche ab:

1) Es werden die benötigten Lernalgorithmen importiert und problemspezifische (Parameter-) Einstellungen vorgenommen.
2) Der Datensatz wird eingelesen und durchläuft die noch benötigten Vorverarbeitungsschritte für die einwandfreie Verwendung in den ausgewählten ML-Verfahren.
3) Es wird eine (randomisierte) Aufteilung in Trainings- und Testdaten vorgenommen, sowie in Labels und Features.
4) Das ML-Verfahren wird mit den Trainingsdaten trainiert.
5) Nach Übergabe der Test-Features geben die ML-Verfahren eine Vorhersage aus.
6) Die Vorhersage wird an Hand der Test-Labels mit Hilfe von Metriken auf Güte überprüft. Die Ergebnisse werden (grafisch aufgewertet) ausgegeben.

Modellbeschreibung:

Durch Dokumentation wird das Nachvollziehen des Programmcodes für kommende Anpassungsprozesse erleichtert. Je nach Zielgruppe und fachlichem Verständnis reicht dies von Kommentaren zwischen den Codeblöcken, bis hin zu zusammenfassenden Handbüchern. Während Ersteres nach guter IT Praxis schon während des gesamten Entwicklungsprozesses stattfindet, so findet Letzteres erst mit der gesammelten Erfahrung am Ende der Entwicklung seinen Platz.

An dieser Stelle sei der interaktive Charakter von Testumgebungen wie Jupyter Notebook (siehe Project Jupyter: jupyter.org) ins Licht zu rücken. Mit der Möglichkeit zur Dokumentation in Markdown (Auszeichnungssprache) und der direkten Ausgabe von Rechenergebnissen und Grafiken, unterstützen solche Applikationen die Dokumentation. Nicht ohne Grund nutzen viele Webdokumentationen für (Python) Code deshalb Jupyter Notebook.

3.4.4 Modellbeurteilung

Schritt 1: Beurteilung des Modells

In der Beurteilung wird das trainierte ML-Modell an Hand des Test Designs auf seinen Mehrwert für das Unternehmen geprüft. Die Ist- und Soll-Güte des ML-Modells wird zunächst verglichen und etwaige Abweichungen werden beurteilt. Abschließend wird geprüft, ob die im Vorhinein überschlagenen Mindestanforderungen für die KPIs erreicht werden. Der Aussagengehalt der Kennzahlen unterstützt dabei die Entscheidung: Ist das Modell ungeeignet, sind die verwendeten Parameter noch zu optimieren, oder kann das Modell mit seinen aktuellen Einstellungen verwendet werden?

Schritt 2: Anpassung und Ergänzungen

Identifiziert die Modellbeurteilung gewisse Unzulänglichkeiten kann versucht werden diese zu beheben. Es bietet sich an, den Datensatz noch einmal einer kritischen Datenvorverarbeitung zu unterziehen oder Parameteränderungen und Schnittstellenerweiterungen vorzunehmen. Eine erneute Modellbeurteilung entscheidet im Anschluss darüber, ob weitere Maßnahmen getroffen werden müssen. Der Anpassungsprozess repräsentiert somit eine Ist-Soll-Schleife über die Data Mining Prozessschritte hinweg.

3.5 Evaluation und Modelleinsatz

Die klassische CRISP-DM Planung betrachtet die zwei finalen Prozessschritte Evaluation und Modelleinsatz (engl. deployment) getrennt. Geht es um die Entwicklung einer vollwertigen Softwarelösung, sollte der Modelleinsatz durchaus eine ausführliche Planung und Umsetzung erhalten. Da es sich aber mit der Untersuchung von ML-Potential innerhalb einer Machbarkeitsstudie nicht um einen solchen Fall handelt, sei der Modelleinsatz auf eine kurze Abhandlung in Kapitel 3.5.3 beschränkt.

Die Umsetzung der Data Mining Ziele hat in einer aufbereiteten Datengrundlage und einem trainierten ML-Modell resultiert. Nun geht es um die Analyse der gewonnenen Ergebnisse und die Zusammenfassung aller gewonnenen Erfahrungswerte. Diese zwei Auswertungen sind in die Kapitel 3.5.1 und 3.5.2 aufgeteilt. Ihre getrennte Bearbeitung ist ratsam, da sich ihr Urteil über den Nutzen des Projektes unterscheiden kann. So lässt sich z.B. trotz negativer Evaluation des ML-Potentials ggf. ein Mehrwert aus den Projekterfahrungen generieren.

3.5.1 Beurteilung der Ergebnisse

Schritt 1: Beurteilung der Data Mining und ML-Ergebnisse

Für die Beantwortung der Fragen aus den Data Mining Zielen, sind nicht nur die gewonnenen Informationen bezüglich ML-Potential von Nutzen. Maschinelle Lernverfahren sind nur eine Auswahl aus der Werkzeugsammlung des Data Mining. Während der Datenexploration oder Datenvorverarbeitung wurden vermutlich schon wertvolle Informationen aus simpleren Methoden wie Zeitreihenanalysen oder Filteranfragen generiert. Auch diese sollten dokumentiert werden und in die Bewertung mit einfließen. Geht es an die Beurteilung der Ergebnisse aus den ML-Verfahren, so sind diese eng mit den Aussagen der Modellbeurteilung verknüpft. Erst nachdem ein Regressions- oder Klassifikationsverfahren mit der geforderten Güte eine Vorhersage oder Einordnung vornehmen kann, lassen sich die Verfahren zur fundierten Erkenntnisgewinnung verwenden. Je nach Zielsetzung besitzen dabei nicht nur die Resultate, sondern auch das Verfahren mit seiner Schnittstelle selbst einen Mehrwert für die Unternehmung.

Ist die Untersuchung des Anwendungspotentials von ML die zentrale Aufgabenstellung eines Data Mining Projektes, so geht es um das Zusammenspiel zwischen der Modellgüte und der Rentabilität der Ergebnisse. Fällt eine der zwei Variablen negativ aus, so ist das Potential für ML Anwendungen begrenzt. Es wird also sowohl bewertet wie effizient das Modell die gesuchten Muster erlernen und vorhersagen kann, als auch welchen Stellenwert

diese Muster für das Unternehmen haben. Da die Evaluation den Data Mining Kreislauf schließt, sollten die beim Projektstart definierten Data Mining Ziele ein abschließendes Fazit erhalten.

Schritt 2: Umsetzung der Unternehmensziele

Nachdem die Beurteilung der Data Mining Ergebnisse die bearbeiteten Aufgabenbereiche evaluiert hat, können die anfänglich definierten Akzeptanzkriterien untersuch werden. Auch wenn ein Kriterium durch seine Formulierung mit einem einfachen erfüllt oder nicht erfüllt bewertet werden kann, sollte hier eine kurze Ergebniszusammenfassung stattfinden.

Schritt 3: Anerkennung des Modells

Der letzte Schritt der Beurteilungsphase reflektiert noch einmal über die Gesamtheit des Modells, es wird nun auch die praktische Umsetzung beurteilt. Zur Vermeidung von Voreingenommenheit, ist darauf zu achten, dass die Umsetzung unabhängig von den inhaltlichen Ergebnissen der Analyse beurteilt wird. Der Fokus liegt darauf mit welcher Qualität das selektierte Verfahren im Hinblick auf die ursprünglichen Data Mining Ziele integriert und umgesetzt wurde. Für die Bewertung der Qualität einer ML-Anwendung hat das repräsentative Projekt aus Kapitel 4 die folgenden Qualitätskriterien generiert:

- Inhaltliche Zielgenauigkeit: Werden alle für die Untersuchung der Data Mining Ziele benötigten Informationen vollständig und unverändert verarbeitet?
- Angestrebter Formfaktor: Ist die Darstellung der Informationen korrekt umgesetzt worden? Vermittelt sie die benötigte Informationsmenge mit dem richtigen Grad an Lesbarkeit?
- Performance: Wie schnell lässt sich eine geforderte Menge an Daten verarbeiten und welche Mittel werden dafür benötigt?
- Flexibilität und Anpassungsfähigkeit: In wie weit und mit welchem Aufwand lassen sich Parameter- und Inputwechsel vornehmen? Ist die Anwendung auch für anderweitige, zukünftige Anpassungen ausgelegt?

3.5.2 Rezension des Prozessablaufes

Neben den durch die Projektziele geformten Ergebnissen und Erfahrungswerten, erweisen sich auch solche als wertvoll, welche aus der reinen Projektausführung gewonnenen werden. Besonders aufschlussreich für die Analyse von Arbeitsprozessen einer Machbarkeitsstudie ist die Betrachtung der Prozessschritte in ihrer Gesamtheit. Da Unterprozesse häufig einen Regelkreis-Charakter besitzen und es nicht selten zu prozessübergreifenden Vor- und Nachgriffen kommt, kann eine Effizienzbetrachtung im Hinblick auf Wiederholungen und Sprünge zu verwertbaren Erkenntnissen führen.

3.5.3 Bestimmen der nächsten Schritte

Abgesehen von möglichen Prozessschritten zur Vorbereitung eines Softwareeinsatzes ist der Data Mining Prozess nach der abschließenden Beurteilung zu Ende. Nicht ohne Grund wird dieser Prozess jedoch wie in vielen informationstechnischen Entwicklungsprozessen als Kreislauf definiert. Liegt der Bedarf nach Erneuerungen und Anpassungen des Modells vor, so werden die Arbeitsschritte des Zyklus ein weiteres Mal durchlaufen.

4. Machbarkeitsstudie nach Referenzmodell

Nachdem das Referenzmodell nach CRISP-DM einen anwendungsbezogenen Rahmen für die Untersuchung des ML Potentials gesteckt hat, geht es in diesem Kapitel um dessen Umsetzung im realen Betriebsumfeld. Die folgende Aufarbeitung eines repräsentativen Projektes beinhaltet zwei Hauptaufgaben. Erstere ist die Zusammenfassung der Arbeitsschritte und (Zwischen-) Ergebnisse entlang des Referenzmodells. Dies ermöglicht es zusätzlich den Nutzen und die Realitätstauglichkeit des Referenzmodells zu testen. Als Zweites gilt es, das erklärte Ziel der Potentialermittlung für den repräsentativen Anwendungsfall zu erreichen. Zur Erhaltung von Übersichtlichkeit und Verständlichkeit ist eine starke Kondensation der projektbezogenen Geschehnisse notwendig. Aus diesem Grund wird der im Anhang ausgelagerte Programmcode als referenzierte Stütze dienen, um auf theoretische und praktische Hintergründe zu verweisen. Der kommentierte Code stellt damit auch einen signifikanten Teil der aus dem Projekt gewonnenen logischen und praktischen Erzeugnisse dar.

Das repräsentative Projekt ist unter dem Schirm eines Kooperationsprojektes von Unternehmen X mit der parsQube GmbH abgelaufen. Innerhalb des übergreifenden Projektes wurde das IT-Büro parsQube beauftragt, eine integrierte XML-Datenbank für Unternehmen X aufzusetzen. Diese Datenbank erfasst die in der Produktion anfallenden Messwerte und Sensordaten, um deren (grafische) Auswertung innerhalb eines Web Service zentral zugänglich zu machen. Ergänzend besteht das Interesse die zunehmende Erfassung von solchen Daten im Hinblick auf zukünftige Analysemöglichkeiten mit ML untersuchen zu lassen. Diese Anfrage wurde von parsQube aufgenommen und im Rahmen dieser Bachelorarbeit als Machbarkeitsstudie umgesetzt.

4.1 Verstehen des Unternehmens

Der Kunde Unternehmen X produziert Leit- und Sicherungstechnik für den Schienennah- und -fernverkehr. Eine Hauptkomponente für die Bahnsicherungstechnik sind mechanische Relais oder elektronische Leiterplatten. Da sie eine kritische Rolle in der sicheren Steuerung des Schienenverkehrs einnehmen, werden hohe Anforderung an ihre Funktionsfähigkeit gestellt. Die Produktion dieser Bauteile beinhaltet daher eine Reihe an intensiven Tests. Leiterplatten durchlaufen z.B. von Optischen Inspektionen über Klimakammertests bis hin zu Hochspannungsprüfungen unterschiedliche Tauglichkeitstests.

Der Ausbau von digitalen Schnittstellen und der verstärkte Einsatz von Umgebungssensoren findet auch in den Abteilungen dieser Testbereiche statt. Eine Pilotfunktion übernimmt dabei der Arbeitsbereich der Funktionsprüfung, in dem die Leiterplatten auf ihre finale Funktionsfähigkeit getestet werden. Die seit 2015 erfassten Daten aus den Testsystemen dieses Bereiches werden die Grundlage der Untersuchungen auf ML-Potential darstellen.

4.1.1 Ermittlung der Unternehmensziele

Mit der Betrachtung der Mess- und Prüfwerte aus der Produktion von Unternehmen X, ist der Mehrwert einer Potentialuntersuchung den Unternehmenszielen aus der Qualitätssicherung untergeordnet. Diese zieht als analytische Abteilung mit Kontroll- und prozessoptimierenden Funktionen den größten Nutzen aus den erfassten Daten. Die erklärten Ziele liegen damit in einer möglichst effizienten Aufbereitung und Auswertung der Daten, so dass die Abteilung alle Produktionsprozesse erfolgreich steuern und weiterentwickeln kann. Soll in Zukunft eine ML-Anwendung die Aufgaben der Qualitätssicherung unterstützen, so muss der erzielte Vorteil den investierten Aufwand übersteigen. Anhand dieser Hintergründe lassen sich folgende Ziele und Bedingungen festhalten, welche die Machbarkeitsstudie untersuchen soll:

1) Must have	Die für eine ML Implementierung benötigten Ressourcen sind vorhanden und der damit verbundene Aufwand ist wirtschaftlich vertretbar.
2) Should have	Die Anwendung ist in der Lage die Qualitätssicherung in der Datenaufbereitung und -analyse zumindest zu entlasten.
3) Could have	Mit Hilfe der Anwendung erlangt die Abteilung neue Erkenntnisse und tiefere Einsichten.
4) Won’t have	Eine ML Anwendung darf die bisherigen Prozesse nicht verkomplizieren und tiefgreifende Umschulungen verlangen.

An Hand der übergreifenden Ziele lassen sich nun Akzeptanzkriterien für die Machbarkeitsstudie definieren. Neben den wirtschaftlichen und informationstechnischen Bedingungen ist vor allem die Güte der Aufbereitungs- und Analysefähigkeiten einer ML-Anwendung relevant. Einen Ansatz für die Feststellung dieser Güte bieten die Merkmale, welche die Qualitätssicherung zu erfassen sucht (siehe Zelle 2 und 3).

Zu 1)	• Die vorhandene Datengrundlage ist für die Anwendung in ML-Verfahren geeignet. Es lassen sich verwertbare Informationen ausarbeiten. • Die IT-Infrastruktur und die nativen Formate lassen eine mögliche (Schnittstellen-) Implementierung zu. • Arbeitsaufwand und Kostenpunkte lassen sich in der Theorie durch den Mehrwert einer Implementierung wieder erwirtschaften.
Zu 2)	• Durch die ML-Anwendung können repetitive Muster innerhalb der Daten erkannt werden, um gewisse Arbeitsschritte zu automatisieren. Als relevante Erkennungsmerkmale gelten: Fehlerrate, Messgenauigkeit, Wiederholungsrate.
Zu 3)	• Das ML ermöglicht es noch unbekannte Muster zu identifizieren oder zu prognostizieren. Die dafür relevanten Merkmale sind Umgebungsdaten, Ergebnislabel von Prüfschritten und -abläufen und weitere, indirekt gewonnene Kennzahlen wie jene aus 2).
Zu 4)	• Die Anwendung von ML erfordert keine Umstellungen der aktuellen Arbeitsprozesse und stellt nur eine hilfreiche Ergänzung für diese dar. • Falls nützliche Informationen und Kennzahlen aus der Anwendung gewonnen werden können, sollten diese für die Zielgruppe verständlich und verwertbar sein.

4.1.2 Lagebeurteilung

Durch den Machbarkteitsstudien-Charakter des Projektes, ist im Zuge der Lagebeurteilung keine ausführliche Projektplanung notwendig. Der Kern des Projektteams besteht aus zwei Personen, einen betreuenden Mitarbeiter von parsQube in seiner Rolle als Softwarearchitekt und Datenanalyst, und den Autor dieser Arbeit. Zum erweiterten Kreis der Beteiligten gehören einige Mitarbeiter der Qualitätssicherung und IT-Abteilung von Unternehmen X. Im Sinne einer explorativen Machbarkeitsstudie kann das Projekt daher mit geringem Planungs- und Ressourcenaufwand durchgeführt werden. Die Hardwarenutzung stützt sich auf die Verwendung von handelsüblichen Rechnern. Werden größere Rechenleistungen und Speicher benötigt, steht der firmeneigene Server von parsQube zur Verfügung. Als Softwaretools für die Schnittstellenentwicklung und das allgemeine Data Mining werden die Open Source Applikationen Jupyter Notebook (siehe jupyter.org) und Orange (siehe orange.biolab.si) verwendet.

In Bezug auf Risiken werden zwei mögliche kritische Ereignisse identifiziert. Wird schon in der Phase der Datenexploration eine zu geringe oder irreparable Datenmenge gefunden, kann dies zum frühzeitigen Ende der praktischen Arbeiten führen. Des Weiteren kann es trotz gewissenhafter Datenvorverarbeitung zu mangelhaften Ergebnissen bei der Verwendung von ML-Verfahren kommen. Beide Problematiken würden einen Erkenntnisgewinn für das Projekt bedeuten, allerdings den Wert der Studie vermindern. Die Betrachtung der vorgelegten Informationen führt zu dem Schluss, dass ein solch kleines Data Mining Projekt mit geringem Ressourcenaufwand, wertvollem Knowhow und geringem Risiko einen Mehrwert erzeugen kann.

4.1.3 Festlegen von Data Mining Zielen

Alle in Kapitel 4.1.1 erfassten Ziele, welche das Data Mining und ML betreffen, können jetzt als Grundlage für die Überführung in Data Mining Ziele verwendet werden:

Zu 1)	• Überprüfung auf ausreichende Datenquantität und -qualität im Hinblick auf ML-Verfahren. • Entwicklung eines Parsers zur Überführung der relevanten Informationen zwischen den Formaten. Prototypischer Schnittstellenbau für importierte ML-Verfahren und Kompatibilitätsanalyse mit dem existierenden System.

Zu 2)	• Untersuchung von Korrelationen und (zeitlichen) Mustern innerhalb der Daten. Austausch über Informationsgehalt mit Analysten aus Unternehmen X. Anschließendes Testen von ML-Verfahren auf die Fähigkeit hin ähnliche Ergebnisse zu Tage zu bringen.
Zu 3)	• Berechnung von Vorhersagen auf Testdaten mit Hilfe von trainierten ML-Verfahren. Evaluation der Modellgüte an Hand von ausgewählten Metriken.

Die Aufgabenpakete stehen für indirekte Zielfragen an das Data Mining. Je mehr von diesen Fragen beantwortet werden können, desto erfolgreicher ist das Data Mining. Ob die Antwort positiv oder negativ ausfällt ist dabei nicht zu berücksichtigen. Die finale Evaluation des Mehrwertes ist separat zu betrachten.

4.1.4 Erstellen des Projektplans

In Anbetracht der durchzuführenden Prozessschritte nach CRISP-DM wird ein Projektablaufplan erstellt (siehe Abbildung 11). Dieser bietet eine Orientierung während der Bearbeitung von Projektaufgaben und verschafft eine Übersicht über die vielschichtige und zyklusartige Projektstruktur. Es ist anzumerken, dass die als Dokumente dargestellten Objekte im Rahmen dieser Arbeit nicht als gesonderte Dokumente aufzufassen sind. Ihr Inhalt wird innerhalb der zugehörigen Kapitel dieser Arbeit behandelt.

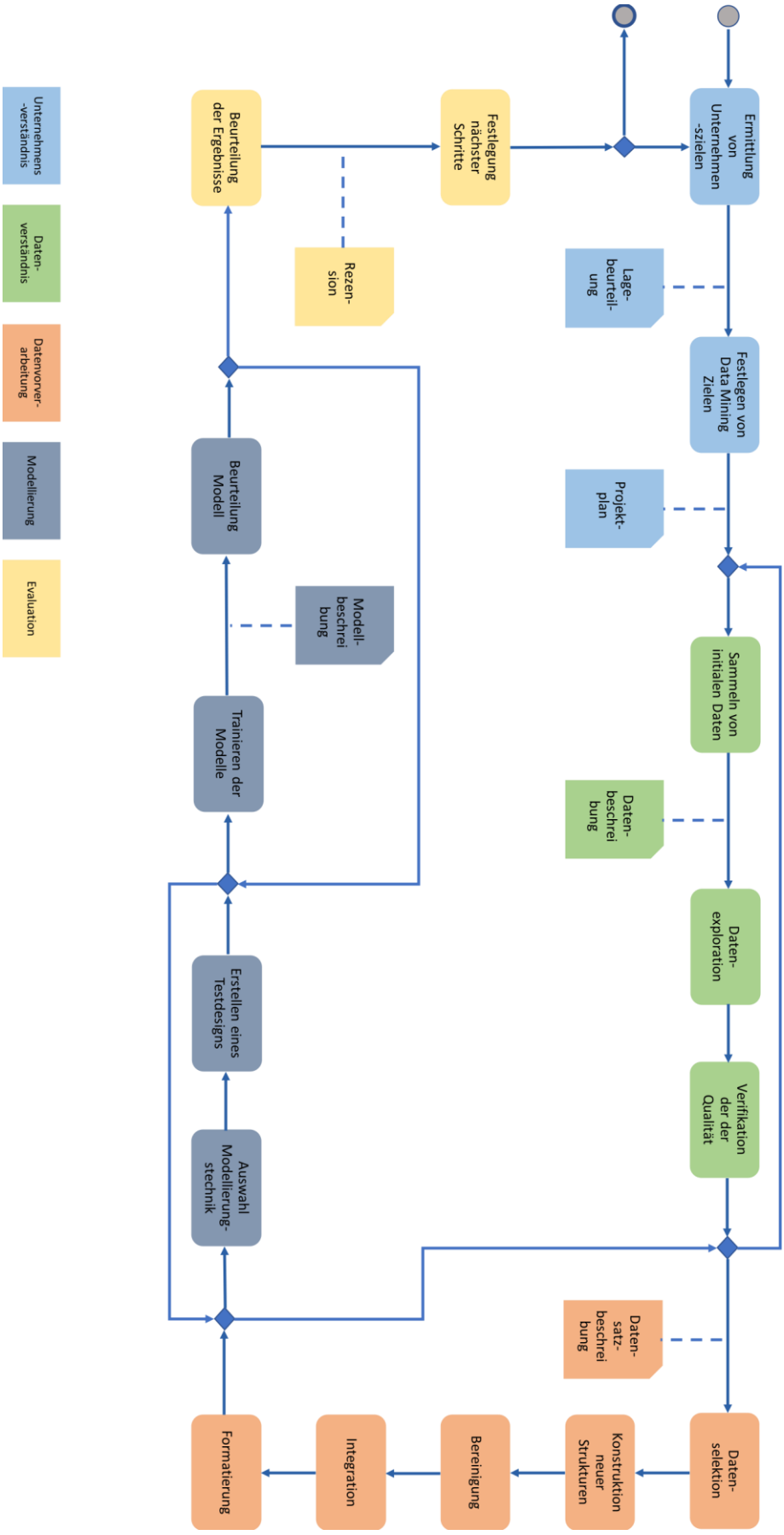


Abbildung 11: Projektablaufplan nach BPMN

4.2 Aufbau eines Datenverständnisses: Mess- und Prüfdaten

Als Grundlage für die Erarbeitung des Datenverständnisses ist eine Übersicht über die bei Unternehmen X vorhandene Datenerfassung hilfreich. Betrachtet wird die Funktionsprüfung einer Leiterplatte (siehe Abbildung 12). Eine Maschine dieses Arbeitsbereiches ist das Testsystem tsvp. Es besitzt eine Schnittstelle für das zentrale Datenmanagement und erfasst die Mess- und Prüfwerte aller Tests in Form von XML-Logdateien. Auf Grund der repräsentativen Messdaten die durch das System erzeugt werden, wird es als zentrale Datenquelle für die Machbarkeitsstudie verwendet

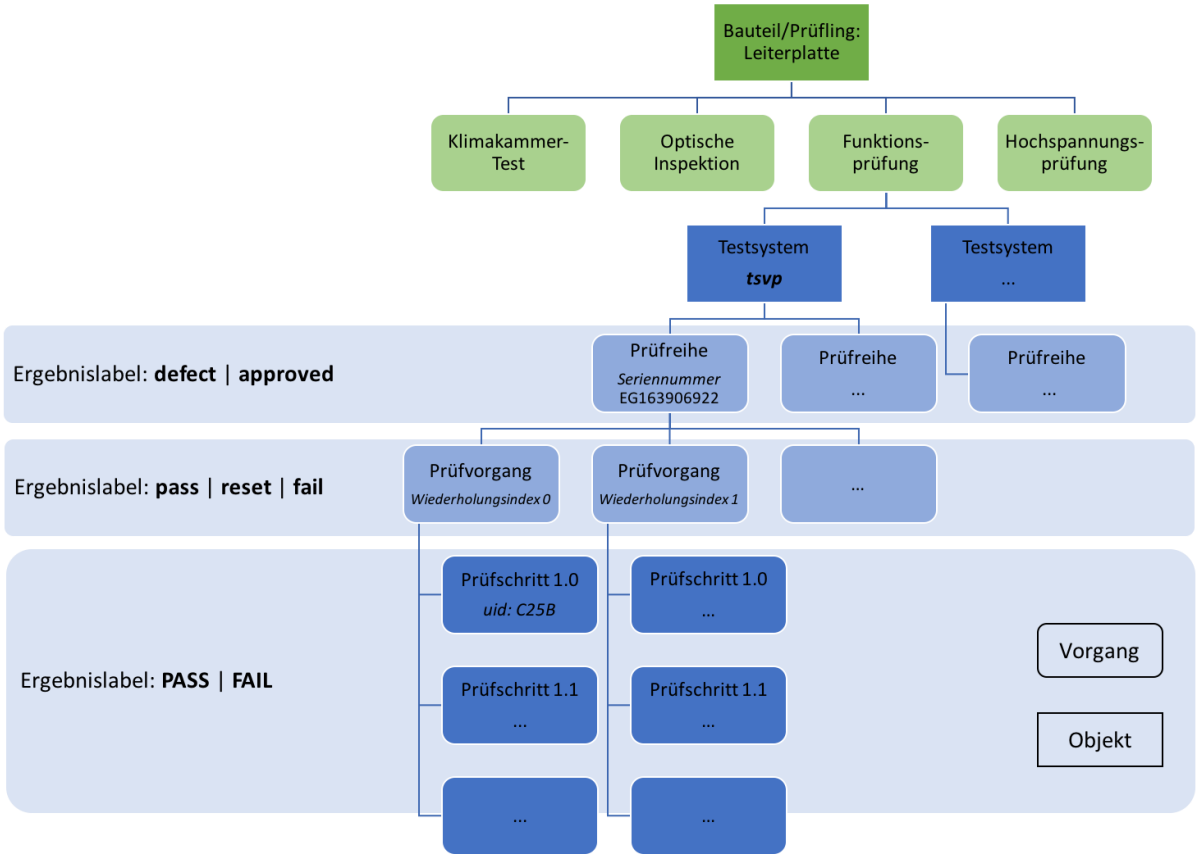


Abbildung 12: Prozessübersicht Unternehmen X – Funktionsprüfung

4.2.1 Sammeln von initialen Daten

Nachdem die Mess- und Prüfdaten als XML-Logdatei dem Testsystem entnommen sind, speichert Unternehmen X diese in einem lokalen Server. Die dabei verwendete Ordnerstruktur ist an die produzierten Bauteilgruppen angelehnt. Innerhalb eines Bauteilordners befinden sich damit alle verfügbaren Logdateien, welche seit Aufnahmebeginn des Formates gespeichert wurden. Zeitliche Lücken lassen sich auf begrenzte Auftragszeiträume und (wartungsbedingte) Aufnahmestopps zurückführen.

Standardisierte Metadaten charakterisieren eine Logdatei und setzen sich zusammen aus:

- Testprogramm (testProgramm), definiert durch die Bauteilgruppe (partNumber)
- Zeitstempel (testStart), von Jahres- bis Millisekunden-Angabe
- Typenlabel (typeLabel) für das zu prüfende Bauteil mit Seriennummer (serialNumber) und Auftragsnummer (jobNumber)
- Testsystembezeichnung (testSystem)

7HA02205ABAA_EG180701227_20180327_104219.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <measurementData>
3      <test logVersion="1.3">
4          <metaData>
5              <rhodos version="1.1"/>
6              <testProgram partNumber="7HA02205ABAA" version="V2.0">
7                  <name lang="de">EAK-ZP30K</name>
8                  <configuration conformalCoating="0" processStep="860" repair="0" repairShop="0" runIn=
9              </testProgram>
10             <testStart year="2018" month="3" day="27" hour="10" minute="42" second="19" msec="452"/>
11             <boardData>
12                 <typeLabel specification="83552135" jobNumber="A100262875" serialNumber="EG180701227"
13             </boardData>
14             <testSystem>Tsvp1</testSystem>
15             <environmentData>
16                 <measurement lowerLimit="15" upperLimit="50" module="hardware.thales.M1998256239" id="
17                 <measurement lowerLimit="18.25" upperLimit="27.75" module="hardware.thales.M1998256239
18             </environmentData>
19         </metaData>
```

Abbildung 13: Metadaten eines Prüfvorgangs - einer XML-Logdatei

Damit definiert eine Logdatei den Prüfvorgang eines spezifischen Bauteils in einem festen zeitlichen und (auf das Testsystem bezogenen) örtlichen Rahmen. Die aufgenommenen Umgebungsdaten (environmentData) können als erweiterte Metadaten betrachtet werden. Eine Erläuterung der Einzelmessungen, welche der Prüfvorgang enthält, wird in der Datenbeschreibung behandelt. Die Datengrundlage für die Data Mining Arbeiten fokussiert sich wie erwähnt auf das Testsystem tsvp. Die Ordner einer Bauteilgruppe stellen daher die oberste Hierarchieebene innerhalb der Daten dar. Die Datenvollständigkeit ist mit dem Vorhandensein aller Bauteilordner damit gewährleistet.

Zur Verwaltung der Logdateien in einer nativen XML-Datenbank wird die Open Source Software eXist-db verwendet. Mit Hilfe dieser kann Unternehmen X abteilungsübergreifende Datenintegration und zentrale Datenbankanfragen ermöglichen. Im Hinblick auf die Architektur ist die vorhandene Datenbank als Hybrid zwischen Schema-On-Read und Schema-On-Write zu verstehen (siehe Kapitel 3.2.1). Während es ein

vordefiniertes Schema durch die standardisierte XML-Struktur gibt, lassen sich Datenbankanfragen auch Ad-hoc definieren. Solche sogenannten Querys ermöglichen es Mitarbeitern von Unternehmen X kontextspezifische Informationen aus der zentralen Datenquelle zu beziehen.

Neben dem Aufbau der eXist-Datenbank ist parsQube beauftragt einen Web Service für weitere Analysezwecke zu entwickeln. Dieser baut auf selbiger Datengrundlage auf und ermöglicht es innerhalb einer Browseranwendung einfache Filtermechanismen und aufbereitete Visualisierungen zu nutzen. Dies ermöglicht automatisierte Analyseanfragen und erweitert die Zielgruppe im Vergleich zu komplexeren Datenbank-Querys. Als rein auswertender Bestandteil der Datenverwaltung wird die Webanwendung nicht weiter eine Rolle für die Datenakquirierung spielen. Sie wird in den kommenden Kapiteln jedoch als potentieller Ansatzpunkt für die gewonnenen Erkenntnisse dieser Arbeit aufgegriffen werden.

4.2.2 Datenbeschreibung

Unternehmen X hat sich im Rahmen seiner Standardisierung von Mess- und Prüfdaten für das XML-Format entschieden. Diese Formatwahl liegt nach Unternehmen X in den folgenden Punkten begründet:

- Das XML-Format erlaubt durch seine Verwendbarkeit in unterschiedlichen Abteilungen ein zentrales Sammeln und Verarbeiten der Daten. Darüber hinaus besitzt es eine Kompatibilität für Web Services.
- Das XML-Format mit seiner natürlichen Strukturiertheit ermöglicht einheitlich definierten Logdateien, welche von den spezifischen Prüfsystemen unabhängig sind.
- Da in Zukunft neue Prüfsysteme integriert und neue Anwendungsfälle berücksichtigt werden müssen, braucht es Möglichkeiten zur Erweiterung der Standardisierung. XML ist durch seine hierarchische Struktur in der Lage, neue Verfeinerungen zu integrieren.

Die Logik und Reihenfolge hinter den Einzelmessungen eines Prüfvorgangs können gut an Hand der XML-Struktur nachvollzogen werden (siehe Abbildung 14). Ein Prüfvorgang setzt sich aus einer Reihe von Prüfschritten zusammen, welche von Startziffer 1.0 an numerisch geordnet sind. Eine solche „stepNumber“ ist jedoch nur als Kennzeichnung im aktuellen Prüfplan zu verstehen. Sie beschreibt die Reihenfolge, in welcher bestimmte Komponenten

und Kontakte der untersuchten Leiterplatte geprüft oder vermessen werden. Eindeutig identifiziert wird der Prüfschritt durch seine ID (uid), welche die Informationen über Leiterplatten-Position und Art der Testkomponente kodiert. Zur übersichtlicheren Darstellung werden die Kindelemente wie folgt aufgelistet:

- Name (stepName), beschreibt kurzgefasst den vollzogenen Test
- Messung (measurement), beinhaltet das binäre Resultat des Prüfschrittes (PASS/FAIL) und einen eigenen Zeitstempel. Weitere Kindelemente sind:
    - Zielwert (targetValue), definiert durch die gemessene Einheit und die Vergleichsmethode. Die Kindelemente sind Minimum und Maximum.
    - Messwert (measurementValue), auch hier die gemessene Einheit und Vergleichsmethode, jedoch bezogen auf den wahren Messwert.

7HA02205ABAA_EG180701227_20180327_104219.xml

```
19          </metaData>
20          <step stepNumber="1.0" rerun="false" uid="C25B" auxiliaryStep="false">
21              <stepName lang="de">Adapterhilfsspannung 12 V</stepName>
22              <measurement type="standard" time="10.42.20:304" result="PASS" count="1" rerun="false">
23                  <element>
24                      <targetValue dataType="number" unit="V" compareMethod="minMax">
25                          <min>11.9</min>
26                          <max>12.1</max>
27                      </targetValue>
28                      <measurementValue dataType="number" unit="V">12.017089325907683</measurementValue>
29                  </element>
30              </measurement>
31          </step>
32          <step stepNumber="1.1" rerun="false" uid="158B" auxiliaryStep="false">
33              <stepName lang="de">Adapterhilfsspannung 12 V Stromaufnahme</stepName>
34              <measurement type="standard" time="10.42.20:315" result="PASS" count="1" rerun="false">
35                  <element>
36                      <targetValue dataType="number" unit="A" compareMethod="minMax">
37                          <min>0</min>
38                          <max>0.12</max>
39                      </targetValue>
40                      <measurementValue dataType="number" unit="A">0.08631306539162556</measurementValue>
41                  </element>
42              </measurement>
43          </step>
```

Abbildung 14: Einzelmessungen eines Prüfvorgangs – einer XML-Logdatei

Das abschließende Element des Prüfvorgangs (nach der Dokumentation der Prüfschritte) ist das Testresultat (testResult). Das Ergebnis des gesamten Prüfvorgangs ist in drei Kategorien gegliedert:

- „pass“: Alle Prüfschritte wurden durchlaufen und als „PASS“ befunden
- „fail“: Trotz eines „FAIL“ entscheidet der Mitarbeiter den Prüfvorgang zu Ende zu führen um weitere Informationen über die Leiterplatte in Erfahrung zu bringen.

- „reset“: Entweder wird direkt nach einem „FAIL“ der Vorgang frühzeitig beendet, oder nachdem der Mitarbeiter ein paar weitere Schritte durchlaufen hat. Wird im Anschluss ein neuer Vorgang gestartet, so beginnt dieser je nach Prüfprogrammvorschrift entweder genau mit dem zuletzt getesteten Prüfschritt oder mit einem verpflichtenden Vorgänger des Prüfschrittes.

Solange kein „pass“ das Bauteil, bzw. die Leiterplatte als einwandfrei begutachtet, werden nach einem „fail“ oder „reset“ erneut Prüfvorgänge durchlaufen. Es liegt im Ermessen des Mitarbeiters zu entscheiden, nach welcher Anzahl von wiederholten Durchläufen eine Leiterplatte als Defekt aussortiert wird.

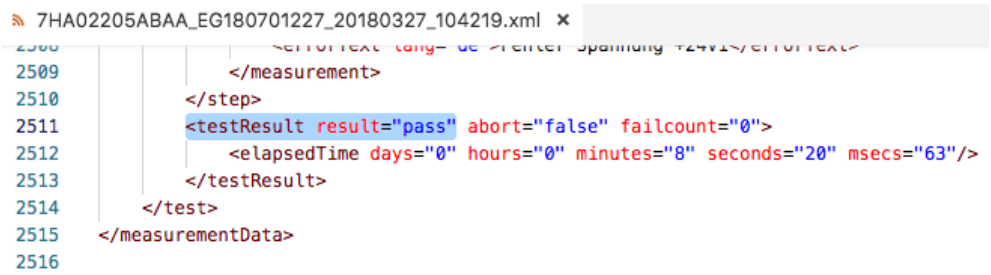


Abbildung 15: Testergebnis eines Prüfvorgangs / einer XML-Logdatei

Die Kapitel der Datensammlung und Datenbeschreibung haben sich mit den relevanten Bestandteilen einer Logdatei beschäftigt. Alle weiteren XML-Elemente und Attribute haben für die weiteren Prozessschritte keine Relevanz.

4.2.3 Datenexploration

Die Betrachtung der XML-Struktur und des Inhaltes einzelner Logdateien ist für das tiefere Datenverständnis zu Verarbeitungs- und Analysezwecke nicht ausreichend. Relevante Informationen für kommende Analysen müssen zunächst aus den Logdateien extrahiert, selektiert und neu konstruiert werden. Die Behandlung dieser Schritte zur Vorverarbeitung wird in den Kapiteln 4.3.1 und 4.3.2 behandelt und ist grundlegend für die erste Datenexploration. Der thematischen Kapitelsortierung nach werden sie erst später behandelt, während die Ergebnisse der Datenexploration auf Grund der Relevanz für das Datenverständnis in diesem Kapitel ihren Platz finden. Für die nun beschriebenen Untersuchungen wird daher ein neu konstruierter Datensatz im Tabellenformat (siehe Abbildung 16), im Folgenden auch als Dataframe bezeichnet, als gegeben vorausgesetzt. Dieser Datensatz beinhaltet die aggregierten Attribute, welche bereits in der Datenbeschreibung erläutert wurden. Es handelt sich darüber hinaus um den Datensatz des

Bauteilordners „7HA02205ABAA“. Dieser ist auf Grund seiner Dateigröße am besten für statistisch solide Analysen geeignet und wird daher in der weiteren Arbeit als Anwendungsbeispiel dienen.

```
df.tail()
```

timestamp	testStart	compMeth	count	envTemp1	envTemp2	measMax	measMin	measVal	name	result	partNbr	jobNbr
2018-04-16 10:33:54.296	2018-04-16T10:23:08.546	minMax	1	26.8125	22.0625	2.000000	-2.000000	0.332158	EAK-ZP30K	pass	7HA02205ABAA	A100262875
2018-04-16 10:34:10.169	2018-04-16T10:23:08.546	minMax	1	26.8125	22.0625	1.000000	0.764705	0.846631	EAK-ZP30K	pass	7HA02205ABAA	A100262875
2018-04-16 10:34:24.903	2018-04-16T10:23:08.546	minMax	1	26.8125	22.0625	-0.235295	-1.000000	-0.338554	EAK-ZP30K	pass	7HA02205ABAA	A100262875
2018-04-16 10:35:01.094	2018-04-16T10:23:08.546	minMax	1	26.8125	22.0625	2.000000	-2.000000	-0.352028	EAK-ZP30K	pass	7HA02205ABAA	A100262875
2018-04-16 10:35:36.434	2018-04-16T10:23:08.546	minMax	1	26.8125	22.0625	25.400000	23.000000	24.885368	EAK-ZP30K	pass	7HA02205ABAA	A100262875

Abbildung 16: Tabellierte Informationen der Logdatei 7HA02205ABAA

In Kapitel 3.4.3 ist bereits auf die Testumgebung Jupyter Notebook verwiesen worden. Sie ist das Hauptwerkzeug der kommenden Datenverarbeitungsschritte und wird für die Erarbeitung der hier präsentierten Ergebnisse genutzt. Erste Ansätze einer Datenexploration sind meist von zusammenfassender und visualisierender Natur. Da die Ergebnisse der Prüfvorgänge und Prüfschritte eine zentrale Rolle für die Qualitätssicherung darstellen, lohnt es sich aus Übersichtsgründen diese Informationen zunächst zeitlich zu aggregieren und grafisch abzutragen. Innerhalb des Notebooks Implizite_Datenanalyse (siehe Anhang) wurde zu diesem Zweck der prozentuale Anteil von „fail“ und „reset“ an der Gesamtanzahl an Ergebnissen im zeitlichen Verlauf festgehalten (siehe Abbildung 17). Mit Hilfe dieser Darstellung lässt sich ein erster Eindruck über den Anteil an mangelhaften Prüfvorgängen und deren Verlauf über die Aufträge hinweg bilden.

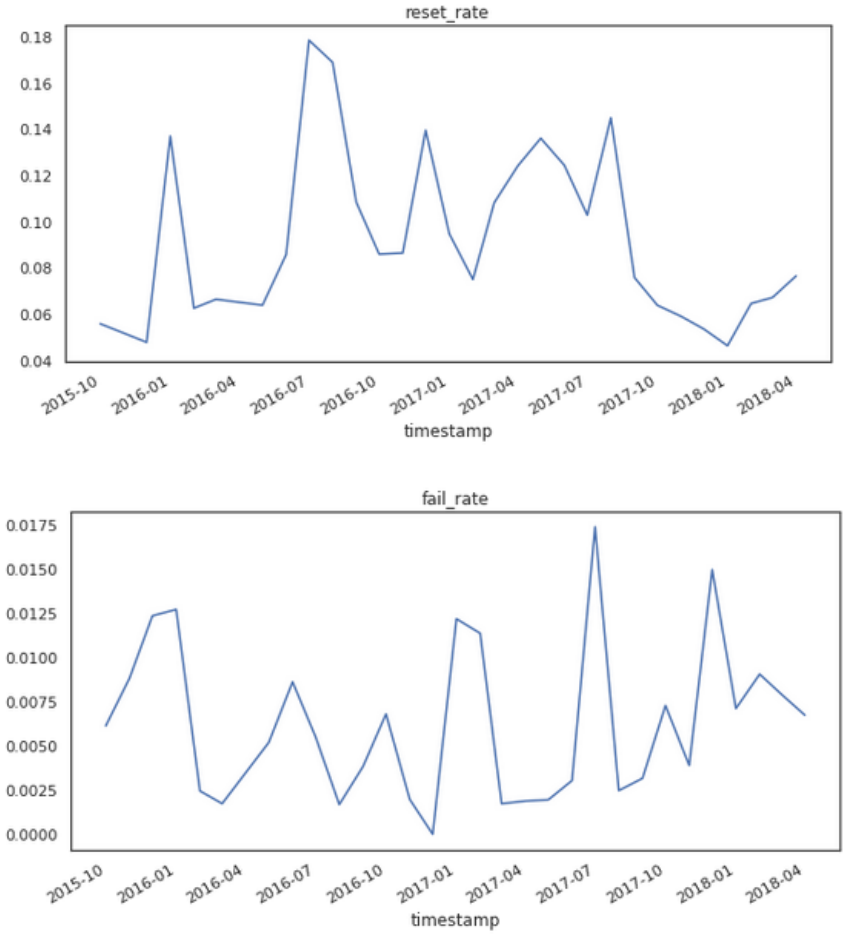


Abbildung 17: Prozentuale Fehler- und Reset-Rate im zeitlichen Verlauf

Gewisse Informationen können nicht direkt aus den Logdateien ausgelesen werden. In weiteren Bearbeitungsschritten lassen sich die impliziten Informationen aus der Datenbasis generieren. Ein solches Beispiel ist das neu erstellte Attribut final_result, welches das Endergebnis einer Reihe an Prüfvorgängen beschreibt. Mit Hilfe dieses neuen Attributes kann das finale Ergebnis einer getesteten Leiterplatte (defect/approved) zusammenfassen. Da final_result die Reihe aller Prüfvorgänge einer Seriennummer zusammenfasst, wird die Bezeichnung Prüfreihe eingeführt. In der Übersicht zum Anfang des Überkapitels wurde dieser Sachverhalt schon dargestellt (siehe Abbildung 12). Es lässt sich mit diesem Attribut nun auch die Defekt-Rate (defect_rate) zeitlich abtragen.

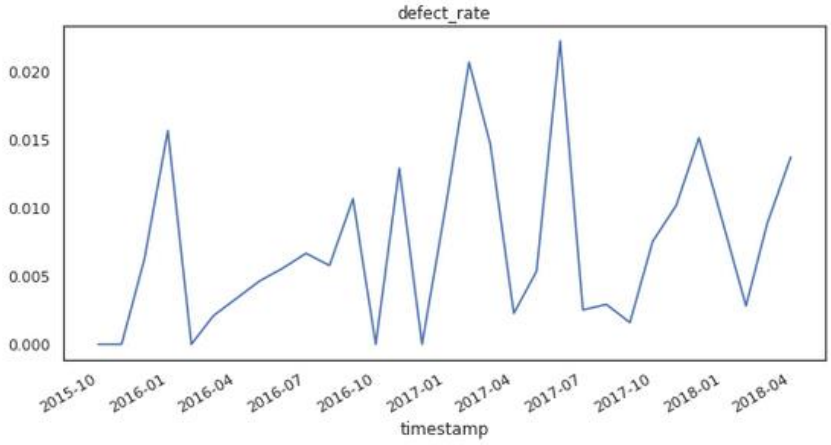


Abbildung 18: Prozentuale Defekt-Rate im zeitlichen Verlauf

Weiterer solcher impliziten Informationen lassen sich nun schrittweise aggregieren und können im Anschluss für weitere Auswertungen genutzt werden. Interessant ist z.B. der Beitrag eines jeden Prüfschrittes zur kumulierte Fehlerrate über den Prüfvorgang hinweg. Abbildung 19 veranschaulicht diesen prozentualen Zuwachs, ausgelöst durch die spezifische Anzahl an gelabelten „FAIL“ pro Prüfschritt. Auf diese Weise lassen sich schnell kritische Prüfschritte wie 1.5.1 und 1.8.1 identifizieren.

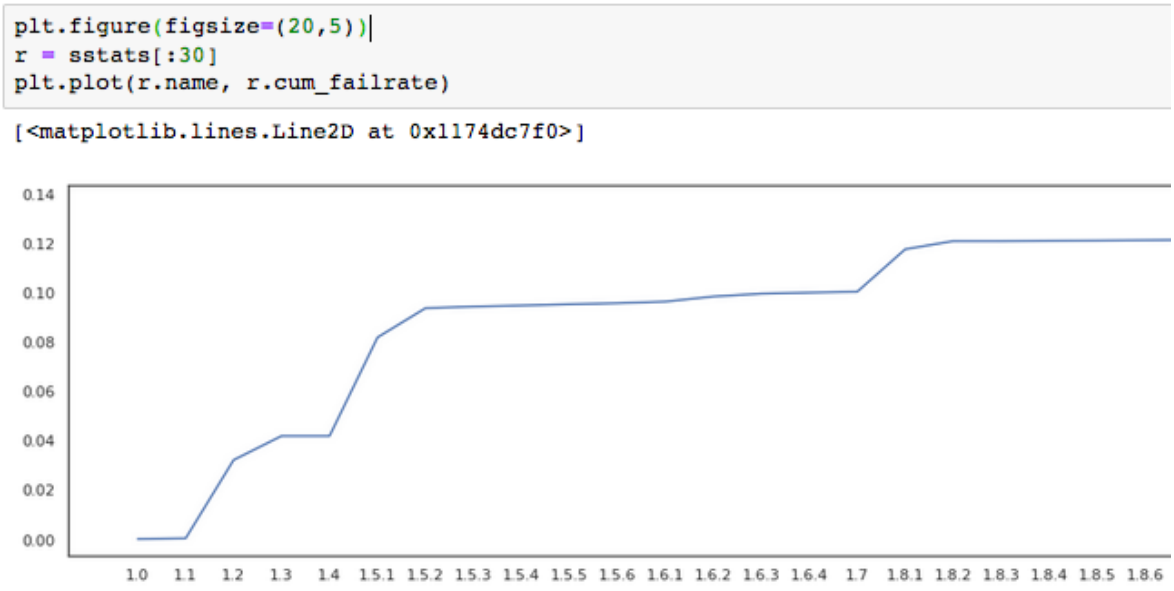


Abbildung 19: Verlauf der kumulierten Fehlerrate über die Einzelmessungen

Möchte man die Zusammenhänge zwischen gewissen Attributen untersuchen, ist eine Korrelationsanalyse sinnvoll. Ziel ist es nicht nur die gängige Korrelationsgröße nach Pearson (R^2, siehe Kapitel 2.3) zu berechnen, sondern auch die grafische Darstellung der zwei Vergleichsgrößen auf auffällige Muster zu untersuchen. Innerhalb der

Machbarkeitsstudie wurden diesbezüglich drei Korrelationsanalysen in jeweils einem Notebook betrachtet (siehe Anhang). Für ausführliche Arbeitsbeschreibungen und Herleitungen kann auf die interaktiven Notebooks selbst zugegriffen werden. Es folgt eine Kurzbeschreibung und das relevante Ergebnis des jeweiligen Notebooks:

- „Korrelation_Steps_untereinander“: untersucht selbstbezeichnend alle möglichen Korrelationsbeziehungen der Prüfschritte untereinander. Es wurden sowohl identische Prüfschritte mit besonderer Messgleichheit, als auch unerwartete Beeinflussungen der Prüfschritte untereinander identifiziert. Ein Mehrwert verbirgt sich hinter der Identifikation von stark korrelierenden, gleichartigen Prüfschritten wie 1.5.1, 1.8.1, 11.1 .
- „Korrelation_Steps_Temperatur“: berechnet Korrelationskoeffizienten, Standardabweichung und lineares Modell für die Beziehungen zwischen Prüfschritten und Temperaturdaten des Prüfvorgangs (siehe Metadaten in Kapitel 4.2.1). Identifiziert werden sollten mögliche Temperatureinflüsse. Die vielversprechendsten Kandidaten besaßen jedoch einen zu niedrigen Korrelationskoeffizienten und eine zu große Streuung.
- „Korrelation_Steps_Ambient“: beinhaltet das gleiche Vorgehen wie das Notebook zur Temperaturkorrelation. Anstatt den Daten von Temperatursensoren direkt am Testsystem werden Umgebungsdaten von Raumsensoren innerhalb der Produktionsstätte verwendet. Diese Untersuchung von möglichen Zusammenhängen mit „externen“ (nicht aus der Logdatei stammenden) Metadaten, ermöglicht die Ausschöpfung des vorhandenen Informationsangebotes. Genau wie bei der vorherigen Temperaturanalyse konnten hier keine aussagekräftigen Korrelationen der Prüfschritte mit Temperatur- und Feuchtigkeitsdaten gefunden werden.

Zusammenfassend hat die Datenexploration drei grundlegende Erkenntnisse gewonnen. Es gibt ein Bild über die Anzahl an mangelhaften Prüfschritten und Defekten, kritische Prüfschritte wurden identifiziert und interne sowie externe Zusammenhänge der Prüfvorgänge wurden erforscht. Darüber hinaus ermöglichen die neu aggregierten Daten eine erweiterte Datengrundlage für die kommende Modellierungsphase.

4.2.4 Verifikation der Datenqualität

Ist ein grundlegendes Datenverständnis durch Beschreibung und Exploration vorhanden, kann nun auf Basis dessen die Datenqualität analysiert werden. Als Qualitätskriterien werden die 6 Dimensionen nach Vermeulen verwendet (siehe Kapitel 3.2.4).

Vollständigkeit	Es konnten keine fehlenden Einträge, bzw. Messwert-Lücken innerhalb eines Prüfdurchlaufes gefunden werden. Als Nachweis gilt die iterative Funktionsfähigkeit des Parsers über alle Bauteilordner (siehe Anhang). Die „fehlenden“ Prüfschritte nach Abbruch eines Prüfvorganges sollten an dieser Stelle nicht als fehlende Einträge missverstanden werden. Auch ihr Nicht-Vorhandensein charakterisiert einen spezifischen „reset“ Fall und ist kein Fehler der Datenaufnahme. Bezüglich der Fehlerhaftigkeit von Messung müssen Numerische Messwerte im Kontext ihres Messbereiches untersucht werden. Ausreißer können fehlerhafte Messungen und folglich falsche Einträge bedeuten. Ob ein Ausreißer tatsächlich einer fehlerhaften Messung entstammt, oder doch ein wahres Phänomen beschreibt, ist schwer zu identifizieren.
Einzigartigkeit	1 Ebene: Auch wenn die numerischen Messwerte eines Prüfschrittes sich im gleichen Messintervall befinden, unterscheiden sie sich durch ihre präzise Messgenauigkeit. Es liegt eine hohe Einzigartigkeit vor. 2 Ebene: Das Vorhandensein von gleichartigen Prüfschritten kann die Einzigartigkeit einschränken. Es gibt insgesamt sieben unterschiedliche Messeinheiten, welche einzeln jedoch auch noch vielfältige Messwertgrenzen besitzen. Dokumentiert eine Logdatei also sehr viele Prüfschritte, ist die merkmalsübergreifende Einzigartigkeit durch ein breites Spektrum an Messintervallen gesichert. 3 Ebene: Durch die (auftragsabhängigen) Spezifikationen der Leiterplatten besitzen die Bauteilordner unterschiedliche Kompositionen und Variationen von Prüfschritten. Durch diese Vielfalt ist eine hohe Einzigartigkeit auf der Ordnerebene vorhanden.
Zeitliche Kontinuität	Die zeitliche Kontinuität ist stark vom Bauteilordner abhängig. Je nach Auftragslage und Bedarf an bestimmten Leiterplatten gibt es daher große Unterschiede. Bei vermehrt produzierten Leiterplatten lässt sich, abgesehen von zeitlichen Lücken durch Auftragspausen, aber eine recht solide Kontinuität feststellen (siehe Abbildung 20).

Gültigkeit	Seit der Aufnahme des XML-Standards in 2015 hat die XML-Struktur einige verfeinernde Ergänzungen durchlaufen. Da sich diese jedoch nicht auf die oberen Hierarchiestufen auswirken, ist die Gültigkeit von älteren Datensätzen nur geringfügig belastet.
Genauigkeit	An der Genauigkeit der Mess- und Prüfwerte lassen sich nur schwer Mängel erkennen, es handelt sich um besonders exakte maschinelle Prüfvorgänge. Jeder Prüfschritt hat durch seine kritische Relevanz einen Einfluss auf Beurteilung der Leiterplatte.
Konsistenz	Hier gilt die gleiche Beurteilungsgrundlage wie bei der Gültigkeit. Konsistenz ist somit auch über die unterschiedlichen Datenquellen, bzw. Bauteilordner größtenteils gesichert.

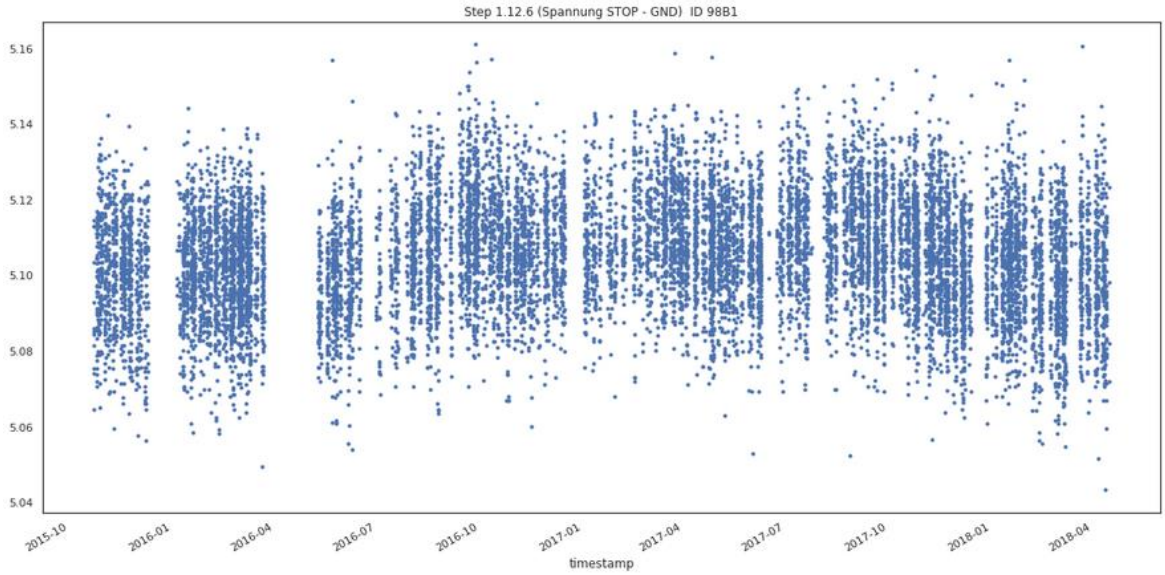


Abbildung 20: Prüfschrittbeispiel zeitlicher Messwertverlauf

In den Kriterien Gültigkeit, Genauigkeit, Konsistenz und Einzigartigkeit erweist sich die Datengrundlage der Logdateien von Unternehmen X als besonders solide. Die Bereiche Vollständigkeit und zeitliche Kontinuität schneiden in den verschiedenen Bauteilordnern unterschiedlich ab.

4.3 Datenvorverarbeitung: Arbeitsprozess mit Jupyter Notebook

Datensatzbeschreibung:

Auf Grund der starken Abhängigkeit der Datenqualität und -quantität von der Größe des Bauteilordners, ist für eine explorative Machbarkeitsstudie ein großes Exemplar zu verwenden. Mit einer Anzahl von rund 15.500 Logdateien, welche jeweils bis zu 131 Prüfschritte zählen, ist „7HA02205ABAA“ einer dieser Kandidaten. Er bietet eine große Datenmenge mit zeitlicher Kontinuität und Vollständigkeit. Darüber hinaus deckt er mit einem Satz von 131 Prüfschritten auch die benötigte Einzigartigkeit für kommende Analyseschritte ab.

4.3.1 Datenselektion und Neukonstruktion

In Kapitel 3.3.1 wurde erläutert, dass sich im Rahmen der Entwicklung eines Parsers für XML-Dateien die Prozessschritte der Datenselektion und Konstruktion effizient vereinen lassen. Es wird daher ein Parser geschrieben, welcher mit Hilfe einer Elementbaum-Funktion die Elemente einer Logdatei hierarchisch durchläuft und dabei selektiv die relevanten Informationen ausliest (Abbildung 21).

```
# direktes Auslesen der Metadaten
name = root.findtext('./test/metaData/testProgram/name')
day  = int(root.find('./test/metaData/testStart').get('day'))
month  = int(root.find('./test/metaData/testStart').get('month'))
year  = int(root.find('./test/metaData/testStart').get('year'))
date = dt.datetime(year, month, day)
serNbr = root.find('./test/metaData/boardData/typeLabel').get('serialNumber')
testResult = root.find('./test/testResult')
if testResult is not None:
    result = root.find('./test/testResult').get('result')
```

Abbildung 21: Auszug Parser - Auslesen von Metadaten

Elementbaum-Funktionen sind ein gängiges Instrument der Informationsrückgewinnung im Umgang mit XML-Dateien (Salem, et al., 2011, pp. 73,74). Die vom Parser ausgelesenen Informationen wurden bereits als Attribute der Logdateien in den Kapiteln 4.2.1 und 4.2.2 vorgestellt. Ihre Selektion ist so definiert, dass sie vollständig alle potentiell wertvollen Informationen beinhaltet. Alle ausgelesenen Informationen werden anschließend vom Parser in das native Python Tabellenformat Pandas geformt.

Das resultierende Dataframe ist für vielfältige Manipulations- und Filterfunktionen geeignet und wird in den Programmausschnitten als simples *df* erkennbar sein. Bevor jedoch weitere Operationen an dem Dataframe vorgenommen werden, wird es mit Hilfe einer übergreifenden Funktion als CSV-Datei im zugehörigen Bauteilordner zwischengespeichert (siehe Abbildung 22). Automatisiert man diesen Vorgang mit einer Schleife über alle

Bauteilordner, lässt sich innerhalb von drei bis vier Stunden die aggregierte CSV-Datei für jeden Ordner erzeugen. Aus der Auswahl von unterschiedlichen Dateiformaten zur Zwischenspeicherung wurde CSV gewählt, da es im Rahmen von Laufzeittests für den automatisierten Parse-Vorgang das beste Ergebnis lieferte (siehe Anhang).

```
In [2]: %%time
        from path import Path
        import csv
        root = Path("data/rhodos/tsvp")

        def process_dir(d):
            print("processing",d)
            df = parse_dir(d)
            f = d/'data.csv'

            if f.exists(): f.remove()

            df.to_csv( f, quoting=csv.QUOTE_NONNUMERIC)
```

Abbildung 22: Auszug automatisierte Parsefunktion

Der Nutzen dieser Zwischenspeicherung liegt in dem Vorteil alle kostspieligen Parse-Vorgänge auszulagern und den aktuellen Datensatz jederzeit ohne Verzögerung aus dem Bauteilordner importieren zu können. Auch die abgeleiteten Attribute aus dem Notebook zur impliziten Datenanalyse stehen nach Integration in das grundlegende Dataframe als neuer, erweiterter Datensatz durch die CSV-Zwischenspeicherung jederzeit zur Verfügung.

4.3.2 Umformatierung und Datenbereinigung

Die restlichen Arbeitsschritte der Datenvorverarbeitung werden in dem Notebook ML_Preprocessing (siehe Anhang) bearbeitet. Im Hinblick auf die ML-Auswertung der Messdaten eines Prüfvorgangs, ist es naheliegend sich zunächst auf die relevanten Attribute zu fokussieren. Diese sind die Prüfschritte selbst, da sie in ihrer Gesamtheit einen Prüfvorgang eindeutig charakterisieren. Im Rahmen des ML-Hintergrundes sind die unterschiedlichen Prüfschritte also die Features und das Ergebnis des Prüfvorgangs/der Prüfreihe ist das Label. Es wird also nach einem Dataframe mit folgender Struktur verlangt:

- Die Reihen/der Index: Die Seriennummer; als Kennung für eine Prüfreihe besitzt sie einen zusammenfassenden Charakter (bzgl. der Prüfvorgänge, welche zu einem spezifischen Bauteil gehören)
- Die Spalten:
    - Target: Das Ergebnis eines einzelnen Prüfvorganges (*result*) oder das Endergebnis einer ganzen Prüfreihe (*final_result*)

- Hauptfeatures: Jeder einzelne Prüfschritt (jeweiliger Name)
- Weitere Features: Wiederholungsindex (*loopindex*) und je nach Bedarf weitere Attribute welche einem Prüfvorgang eindeutig zugeordnet werden können.

Generiert wird dieses Dataframe durch die Pivotisierung der Messwerte nach den Prüfschritten (siehe Abbildung 23). Alle weiteren Attribute wie das Ergebnis und der Wiederholungsindex werden, nachdem sie vom Ursprungs-Dataframe separiert wurden, an das neue Dataframe geheftet. Das Ursprungs-Dataframe muss dafür nach Zeitstempel sortiert und nach Seriennummer gruppiert sein, um eine passgenaue Zusammenführung zu ermöglichen.

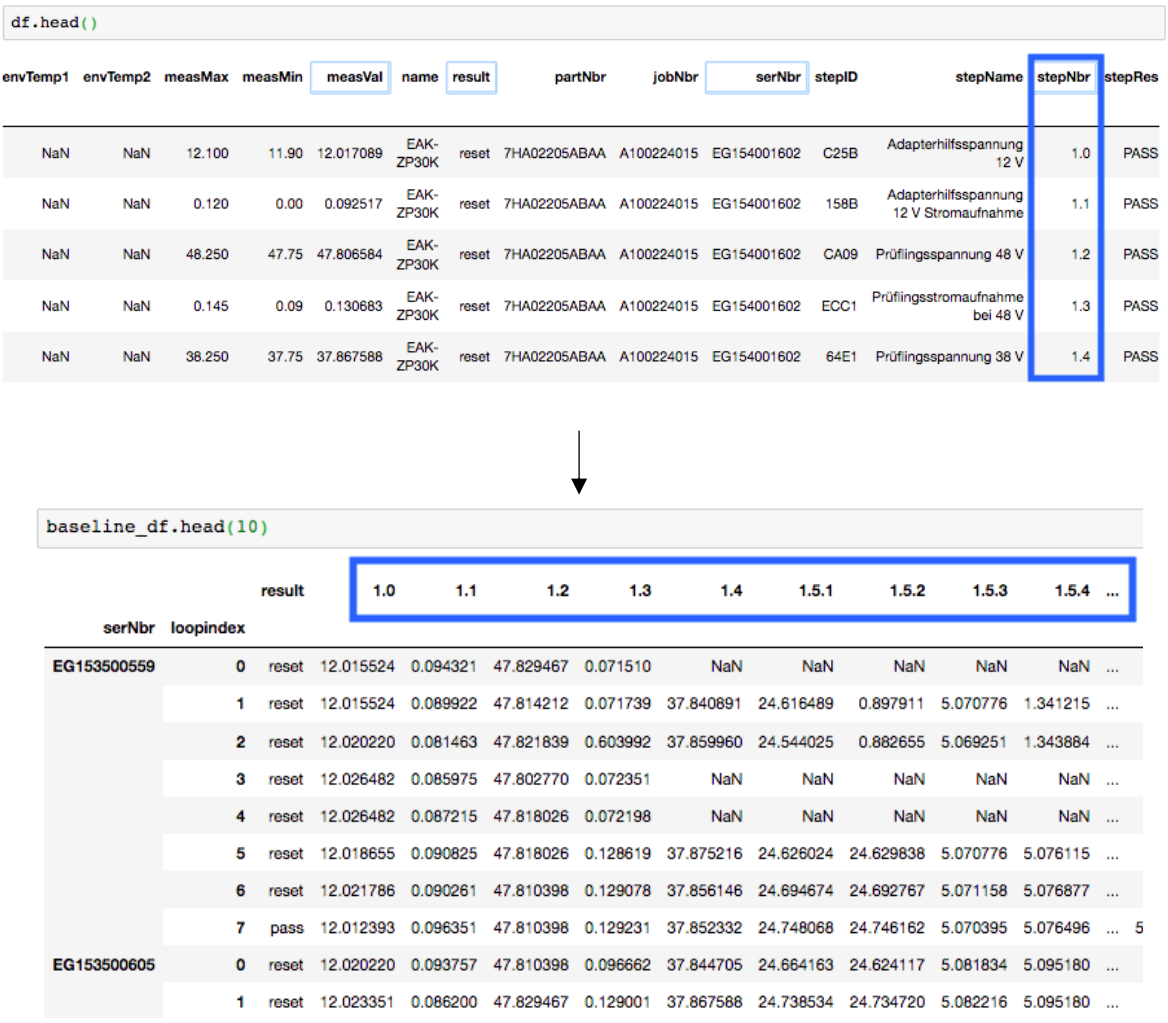


Abbildung 23: Pivotisierung der Messwerte nach Prüfschritten

Die Beurteilung der Datenqualität hat ergeben, dass in den Datensätzen keine Messwert-Lücken existieren und fehlerhafte Messwerteinträge nur schwer zu identifizieren ist (siehe Kapitel 3.2.4). Diesbezüglich wird der Datensatz nicht weiter bereinigt, da auch davon ausgegangen wird, dass die eingesetzten ML-Verfahren trotz der natürlich vorkommenden Ausreißer eine Fähigkeit zur Generalisierung entwickeln müssen.

Zur Behandlung bleiben schlussendlich nur die fehlenden Messwerte, welche durch den Abbruch und anschließenden Neustart eines Prüfdurchlaufes (Label *reset*) entstehen. Diese sind durch den Parser bereits mit einem NaN-Wert belegt worden. Auch wenn diese Werte keinen Messwert beinhalten, charakterisierten sie einen Prüfdurchlauf durch das Aufzeigen von Start und Ende eines mangelhaften Durchlaufes. Einen Ersatz durch Mittelwerte oder anderweitige Dummy-Werte ist daher nicht im Sinne des Aussagengehaltes dieser fehlenden Werte. Die Entscheidung fällt daher auf einen Ersatzwert außerhalb des Wertebereiches jeder Messung: eine **-1**. Durch diesen Wert ist es den ML-Verfahren möglich, einen fehlenden Messwert eindeutig zu identifizieren und dementsprechend zu gewichten. Des Weiteren werden die Labels der Ergebnisspalte noch in Zahlen kodiert, da die ML-Verfahren aus der Python Bibliothek scikit nur rein numerische Werte als Input akzeptieren (Abbildung 24).

```
data_int_res = baseline_df.replace(['fail','reset','pass'],[0, 1, 2])

data_int_res.fillna(value = -1, inplace=True)
```

```
data_int_res.head()
```

serNbr	loopindex	result	1.0	1.1	1.2	1.3	1.4	1.5.1	1.5.2	1.5.3	...	8.11	9.1
EG153500559	0	1	12.015524	0.094321	47.829467	0.071510	-1.000000	-1.000000	-1.000000	-1.000000	...	-1.0	-1.0
EG153500559	1	1	12.015524	0.089922	47.814212	0.071739	37.840891	24.616489	0.897911	5.070776	...	-1.0	-1.0
EG153500559	2	1	12.020220	0.081463	47.821839	0.603992	37.859960	24.544025	0.882655	5.069251	...	-1.0	-1.0
EG153500559	3	1	12.026482	0.085975	47.802770	0.072351	-1.000000	-1.000000	-1.000000	-1.000000	...	-1.0	-1.0
EG153500559	4	1	12.026482	0.087215	47.818026	0.072198	-1.000000	-1.000000	-1.000000	-1.000000	...	-1.0	-1.0

Abbildung 24: Rein numerisches Dataframe nach Bereinigung

Integration von Daten:

Ebenso wie der Wiederholungsindex an das pivotisierte Messwert-Dataframe angefügt wird, lassen sich auch weitere Attribute eines Prüfdurchlaufes integrieren. Eine Erweiterung könnten z.B. die Temperaturmessungen pro Prüfdurchlauf darstellen. Mit der Hinzunahme von zusätzlichen (Metadaten-) Attributen wird jedoch bis zur Fertigstellung des ersten Modellierungsdurchlaufes gewartet. Lassen sich die Auswirkungen von zusätzlichen

Attributen auf das Ergebnis der ML-Verfahren nicht im Vorhinein einschätzen, ist es ratsam die Modellierung zunächst nur mit dem Basis-Datensatz an reinen Messwerten zu durchlaufen. Anschließend können Attribute schrittweise hinzugenommen werden und ihre Auswirkungen auf das Ergebnis untersucht werden.

4.4 Modellierung: Test Design und Training von ML-Verfahren

4.4.1 Auswahl einer Modellierungstechnik

Als explorative Machbarkeitsstudie ist aus der Aufgabenstellung und den Data Mining Zielen des Projektes zunächst keine engere Auswahl an ML-Verfahren zu erschließen. Die in der Vorverarbeitung aufbereiteten Daten besitzen den Zweck einen Prüfschritt, Prüfvorgang oder eine gesamte Prüfreihe eindeutig zu charakterisieren. Jeder dieser Elemente besitzt dazu ein eigenes Label, welches durch den Einsatz von Klassifikationsverfahren bestimmt werden könnte. Im Rahmen dieser Betrachtung tritt jedoch ein Problem auf. Mit 131 Prüfschritten als Features, verfällt der ausgewählte Datensatz (Bauteil „7HA02205ABAA“) den Problemkategorien aus Kapitel 3.4.1. Im Sinne einer breiten Potentialbetrachtung und mit dem Ziel auf Grund der kritischen Mehrdimensionalität viele Ergebnisse zu vergleichen, werden alle gängigen Klassifikationsverfahren getestet. Die benötigte Kardinalskalierung für die ML-Verfahren ist auf Grund der rein numerischen Messwerte und der Codierung von anderweitigen Attributen (siehe Kapitel 4.3.2) gegeben.

4.4.2 Erstellung eines Test Designs

Ziel-KPIs

Die Prüfdurchläufe an dem Testsystem für Leiterplatten werden von Mitarbeitern des Unternehmens genutzt, um die Einzelkomponenten auf Mängel zu untersuchen. Wird bei einem Prüfschritt ein Mangel („FAIL“) festgestellt, so wird das Bauteil durch den Mitarbeiter einer Reparaturmaßnahme oder einem weiteren Prüfvorgang unterzogen. Im Wechsel mit Reparaturen testen erneute Prüfdurchläufe die Leiterplatte, bis alle Tests erfolgreich waren oder der Mitarbeiter das Bauteil als Defekt aussortiert. Als KPI für diesen Vorgang eignet sich die Durchsatzrate (engl. Throughput Rate) nach ISO 22400-2 (2014) (Kang, et al., 2016, p. 6340):

$$\text{Durchsatz} = \frac{\text{Anzahl bearbeiteter Teile}}{\text{Bearbeitungszeit}} \left[\frac{\text{Einheit}}{\text{Stunde}}\right]$$

Zur Berücksichtigung der durch die Reparatur entstehenden Materialkosten, werden diese Kennzahl auch als KPI aufgenommen:

$$\text{Materialkosten} = \frac{\text{verwendetes Material}}{\text{geprüfte Einheit}} \left[\frac{€}{\text{Einheit}}\right]$$

Damit wurden zwei KPIs identifizieren, welche es gilt durch den (theoretischen) Einsatz einer ML-Anwendung zu verbessern.

Szenario:

Falls es für einen Mitarbeiter möglich ist, eine defekte Leiterplatte und ggf. die „Problemklasse“ dahinter früher zu erkennen, kann die Durchsatzrate erhöht und der Materialverbrauch gesenkt werden. Durch ein optimal trainiertes ML-Verfahren wäre der Mitarbeiter auf Grund von Hinweisen in der Lage, zu bestimmten kritischen Stellen im Prüfprogramm zu springen. Auch die maximale Anzahl an Prüfdurchläufen für eine Defekt-Kategorisierung könnte durch einen solchen Hinweis besser eingeschätzt werden. Der Test auf die Lernfähigkeit eines ML-Verfahrens für diesen Anwendungsfall wird in zwei Komplexitätsstufen unterteilt:

1) Test auf die Fähigkeit defekte von funktionsfähigen Leiterplatten zu unterscheiden: Mit welcher Güte können Klassifikationsverfahren *defect* und *approved* vorhersagen.

2) Test auf die Fähigkeit Problemklassen in Form von Seriennummern defekter Trainingsbeispiele zu erkennen. Dies gilt als optionaler Folgetest, falls Test 1) zufriedenstellende Ergebnisse geliefert hat.

Angesetzte Erfolgskennzahlen

Für eine möglichst vollständige Gütebetrachtung werden die in Kapitel 3.4.2 behandelten Metriken zur Untersuchung der zwei Lernfähigkeitstests genutzt. Zur Festlegung von verbindlichen Gütekennzahlen werden die Metriken der Präzision (Precision) und der Wiedererkennungsrate (Recall) verwendet. Von Test 1) wird als Mindestgrößen für die Wiedererkennungsrate der defekten Teile ein Wert von 0.8 festgelegt. Zur Vermeidung einer verstärkten Fehlklassifikation der funktionsfähigen Teile durch das Training auf Defekterkennung, wird auch der verlangte Präzisionswert auf mindestens 0,8 gesetzt. Im harmonischen Mittel ergibt sich ein damit ein solider F1-Wert von 0,8.

Diese Kennzahlen bedeuten für die Klassifikation, dass das folgende Verhältnis als minimaler Standard gilt:

$$f_{approved} + f_{defect} = 0{,}5 * t_{defect}$$

mit $f_{approved} \,/\, f_{defect} = 0{,}25 * \ t_{defect}$ aus Recall und Precision gleich 0,8 (siehe Formeln aus Kapitel 3.4.2). Auf diese Weise wird gesichert, dass die Anzahl an falsch klassifizierten Leiterplatten maximal die Hälfte der korrekt klassifizierten Defekte beträgt.

Der Autor besitzt durch seine Tätigkeit im dienstleistenden Unternehmen keinen direkten Zugang zu den Kennzahlen der Qualitätssicherung und Produktion bei Unternehmen X. Die angesetzten Materialkosten sind daher fiktive Zahlen. Alle restlichen Zahlenwerte entstammen den Ergebnissen aus der Datenexploration (siehe Anhang: Notebook Implizite_Datenanalyse). Der hier beschriebene Anwendungsfall ist als Veranschaulichung zu verstehen:

- Für das Bauteil 7HA02205ABAA (spez. Leiterplatte) liegt das Verhältnis von Defekt zu Funktionsfähig über die gemessenen 3 Jahre hinweg bei 84:11788. Test 1) soll zunächst die geforderten Mindestgrößen für die Klassifikationsgüte erreichen (s.o.). Es werden also mindestens 67 defekte Teile richtig klassifiziert und von beiden Klassen jeweils maximal 16 Teile falsch klassifiziert.

- Es wird nun von der Annahme ausgegangen, dass sich der Zeitaufwand für die weitere Prüfung von falsch klassifizierten Teilen mit der gewonnenen Zeit durch die korrekt klassifizierten Defekte gegenrechnen lässt. Bezogen auf den aktuellen Datensatz bleibt demnach ein Mehrwert durch die frühe Erkennung von 35 korrekt klassifizierten Defekten (aus 67-(2*16)).

- Die Gesamtheit der Leiterplatten besitzt im Durchschnitt ca. 1,3 Prüfvorgänge. Ein Prüfvorgang inklusive Reparaturarbeiten dauert durchschnittlich ca. 10 min. Bezogen auf die Durchsatzrate ergeben sich dadurch ca. 4,6 geprüfte und bearbeitete Einheiten pro Stunde. Betrachtet man die Defekte separat, besitzen sie durchschnittlich ca. 2,5 Prüfvorgänge. Bei den Defekten liegt damit eine Durchsatzrate von 2,4 Einheiten pro Stunde vor. Falls ein Mitarbeiter mit Hilfe des ML-Verfahrens den Defekt schon nach den gängigen 1,3 Prüfvorgängen erkennen könnte, würde sich die Durchsatzrate von defekten Bauteilen um 2,2 Einheiten pro

Stunde erhöhen. Werden fiktive Materialkosten von 100€ für den Reparaturbedarf je Prüfvorgang beispielhaft angesetzt, so lassen sich die Materialkosten pro defekte Leiterplatte von 250€ auf 130€ um ca. 50 Prozent senken.

- Im Gesamtbild besitzt diese beispielhafte Verbesserung der KPIs wesentlich geringere Auswirkungen. Auf den gemessenen 3 Jahreszeitraum besitzen die defekten Leiterplatten nur einen Anteil von ca. 0,7 Prozent (siehe Punkt 1). Bezieht sich der Mehrwert einer frühen Erkennung wie in Punkt 2 beschrieben nur auf einen gewissen Teil der Defekte, so wirkt sich die KPI Verbesserung auf ca. 0,3 Prozent aller Prüfvorgänge aus.

4.4.3 Aufsetzen eines ML-Modells

Informationstechnische Werkzeuge:

Zur Unterstützung des Modellierungsprozesses ist eine modulare, schrittweise Ausführung von einzelnen Codebausteinen besonders hilfreich. Die Testumgebung Jupyter Notebook ist eine interaktive Shell mit der ein sequentielles Ausführen in einer Browseranwendung möglich ist. Zur Unterstützung der Tests über verschiedene ML-Verfahren wurde überdies die Plattformlösung Orange verwendet. Als Open Source Software für den Bildungssektor ist sie nur eingeschränkt für die Verarbeitung großer Datensätze nutzbar. Der einfache und schnelle Modellierungsvorgang erlaubt jedoch einen frühzeitigen Einblick in die Trainingsergebnisse von ML-Verfahren. Darüber hinaus sind die Funktionen der Plattform auf Python Code basierend und verwenden die gängigen ML-Algorithmen aus scikit. Damit lassen sich die Ergebnisse der Jupyter Anwendung mit denen aus Orange vergleichen und ggf. verifizieren.

Training eines ML-Verfahrens und Schnittstellendesign:

Die Aufgabenbereiche der zu programmierenden Schnittstelle werden in Jupyter umgesetzt. Sie sind jeweils einzeln als Funktionen (siehe ml_functions.py im Anhang) ausgelagert und können dadurch innerhalb des Notebooks ML_Prototype als automatisierter Prozess ausgeführt werden. In Orange werden die einzelnen Arbeitsschritte und Aufgabenbereiche in einem grafischen Netzplan veranschaulicht. Zur Übersicht folgen Verweise auf anschauliche Darstellungen, mit der linken Seite für Jupyter (Abbildung 25 und Abbildung 26) und der rechten Seite für Orange (Abbildung 27).

1) Import Lernalgorithmen und Parameter-Einstellungen:

Siehe *classifiers*	Rot gefärbte Elemente

2) Einlesen des Datensatzes und Vorverarbeitungsschritte:

Siehe Importzelle in ML_Prototype und preprocess	Gelb gefärbte Elemente

3) Aufteilung in Trainings- und Testdaten, sowie in Labels und Features:

Siehe train_test_split	Gelbes Element Data Sampler

4) Training der Klassifikationsverfahren:

Siehe train_classifiers	Blaues Element Test & Score

5) Vorhersage der ML-Verfahren:

Siehe predict_from	Blaues Element Predict

6) Überprüfung der Klassifikationsgüte:

Siehe print_score_info	Blaue Elemente Test & Score, ROC Analysis, Confusion Matrix

```
clf_tree = tree.DecisionTreeClassifier(criterion="gini",max_depth=5)
clf_svm = svm.SVC(kernel='linear')
clf_bayes = naive_bayes.GaussianNB()
clf_randf = RandomForestClassifier(criterion="gini", max_depth=5)
clf_adab = AdaBoostClassifier()
clf_knn = KNeighborsClassifier()
clf_NN = MLPClassifier(solver='adam', alpha=1e-5, hidden_layer_sizes=(100, ))


classifiers = {'tree': clf_tree,'svm': clf_svm, 'bayes': clf_bayes
                , 'random_f': clf_randf, 'adaboost':clf_adab, 'knn':clf_knn, 'neuralnet': clf_NN}
```

Abbildung 25: Verwendete Klassifikationsverfahren in Jupyter

```
cut_selection = data_fin_res.loc[:,'1.0':'1.5.1'].columns.values

for cut in cut_selection:

    print("---------------------------------- Cut-Step: " + cut + " ----------------------------------------")

    X, y = preprocess(data_fin_res, cut)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

    clfs = train_classifiers(classifiers, X_train, y_train)

    predictions = predict_from(clfs, X_test)

    print_scoreinfo(classifiers, predictions, X_train, X_test, y_train, y_test)
```

Abbildung 26: Vorverarbeitung und Training der ML-Verfahren in Jupyter

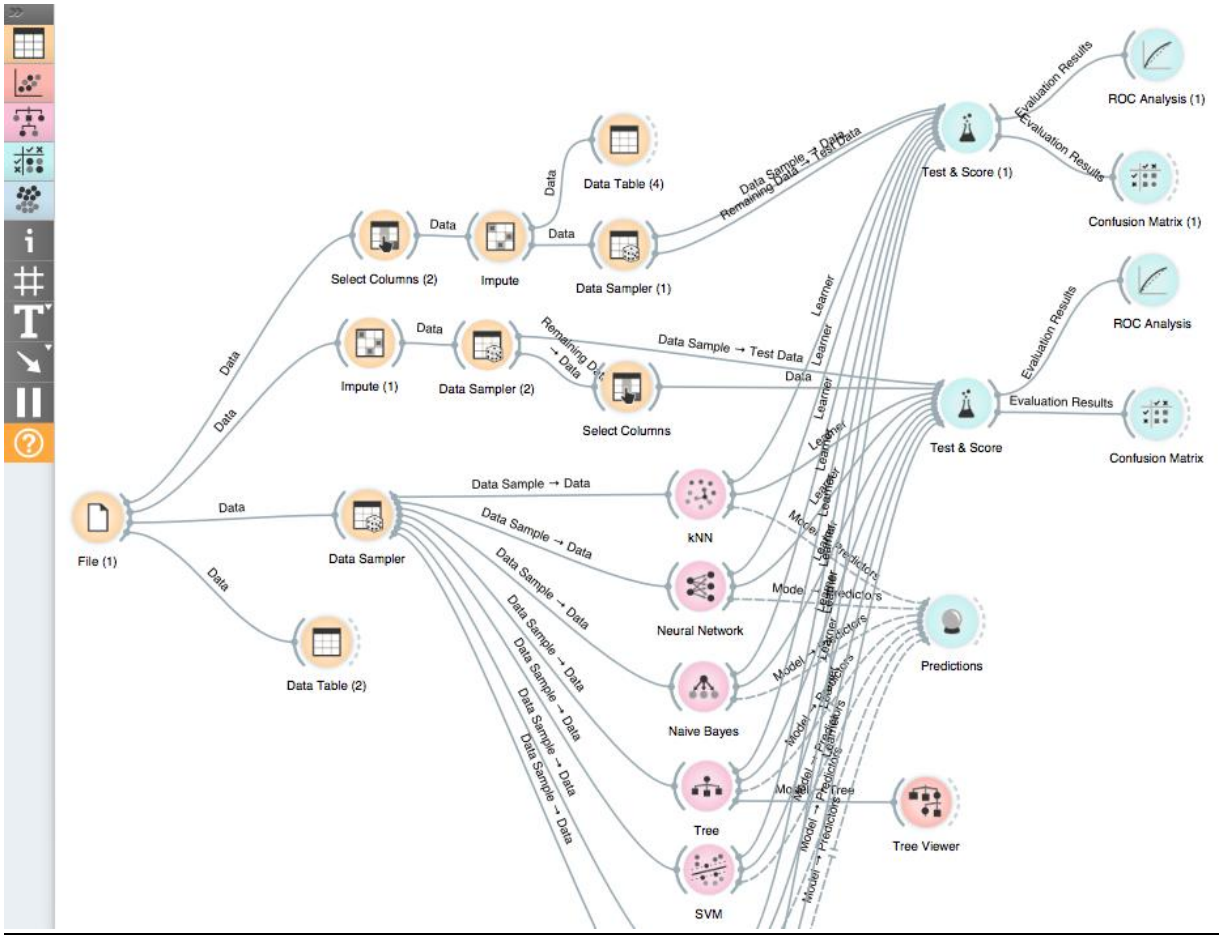


Abbildung 27: Modellierung innerhalb von Orange (Datei ML_testing.ows)

Modellbeschreibung:

Innerhalb von Jupyter Notebook wurde sowohl mit Python Kommentaren (Kennzeichnung # oder ‘‘‘) als auch mit Markdown Zellen das Vorgehen dokumentiert. Der Code und die zugehörigen Ausgaben sind als im Anhang zu finden. Darüber hinaus lassen sich die Notebook und Orange Dateien nach den Anweisungen der ReadMe-Datei auf gängigen PCs interaktiv nachvollziehen.

4.4.1 Modellbeurteilung

Beurteilung des Modells:

Das Szenario des Test Designs verlangt nach einer ML-Anwendung, welche eine Defekt-Erkennung nicht nur am Ende eines Prüfvorganges erlaubt, sondern eine Neuklassifikation auch nach jedem erfolgten Prüfschritt ermöglicht. Schon während des Prüfvorganges könnten dem Mitarbeiter damit frühzeitig Hinweise gegeben werden. Dieses Szenario wird in Jupyter simuliert, in dem der Datensatz für das ML-Training und Testen sowohl nach Wiederholungsindex, als auch nach Prüfschritt gefiltert wird. Der *preprocess* Funktion (siehe Anhang: ML_Prototype) kann ein gewünschter Wiederholungsindex und Prüfschritt übergeben werden, sodass der Fall eines Prüfprogrammes im n-ten Prüfdurchlauf an Prüfschritt-Stelle X simuliert wird.

Die für Test 1) angesetzten Erfolgskennzahlen für die Klassifikationsgüte werden von keinem der getesteten ML-Verfahren erreicht. Die Wiedererkennungsrate und Präzision nimmt mit zunehmender Anzahl an Prüfvorgängen und Prüfschritten zu, doch selbst bei der maximalen Anzahl an Prüfvorgängen und bei der Berücksichtigung aller Prüfschritte wird nicht der Zielwert von 0,8 erreicht. Eines der bestmöglichen Ergebnisse für eine Defekt-Erkennung in der beschriebenen Endauswahl erreicht die NN-Klassifikation mit einem F1-Wert von 0,25 (siehe Abbildung 28). Zum Vergleich mit den verlangten Kennzahlen aus dem Test Design ist das Ergebnis für unterschiedliche Metriken aus Jupyter in Abbildung 28 dargestellt.

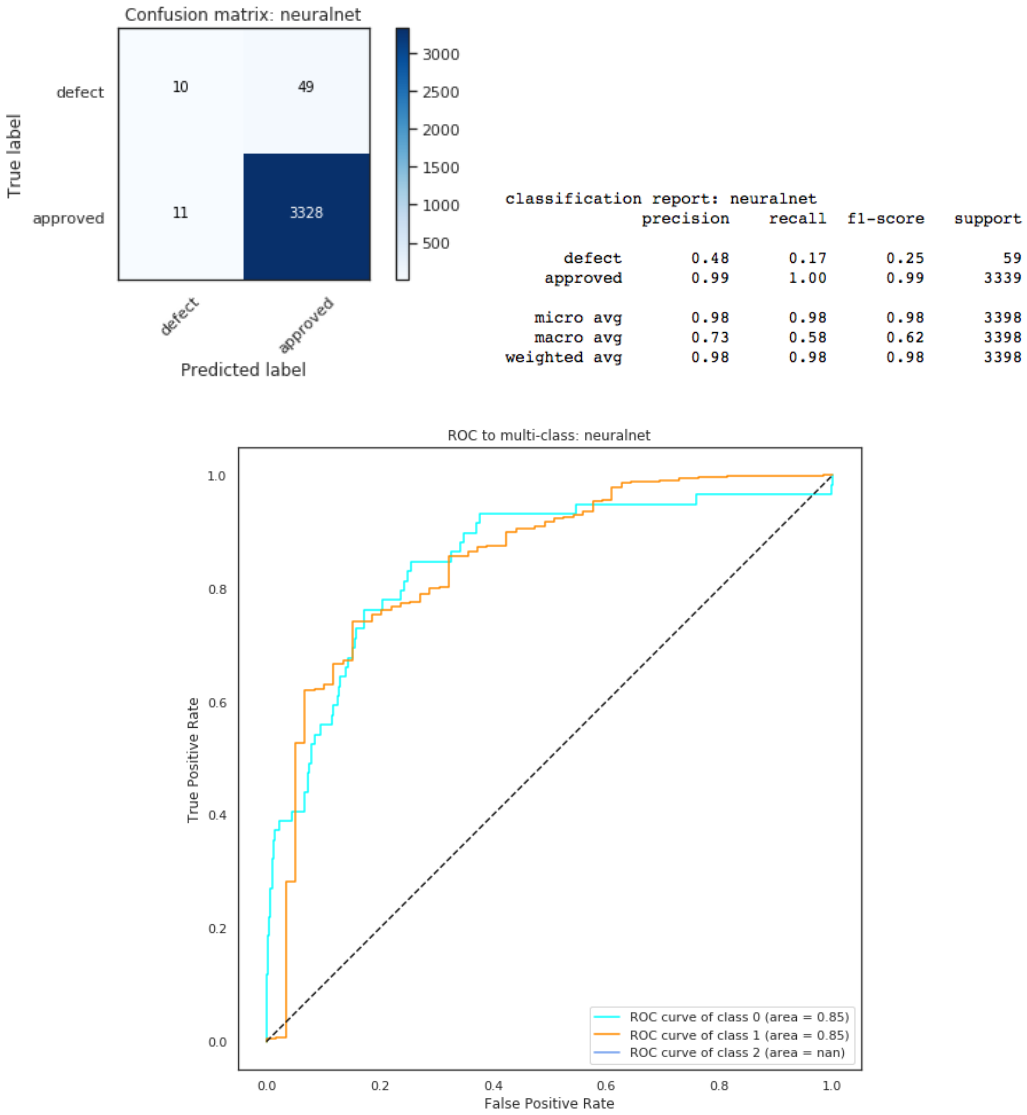


Abbildung 28: Auszug der Klassifikationsergebnisse aus ML_Prototype

Die Gesamtzahl der falsch klassifizierten Leiterplatten dominiert über alle Tests hinweg die Anzahl an korrekt klassifizierten Defekten. Für den im Test Design angesetzten Anwendungsfall lässt sich mit diesen Ergebnissen keine Verbesserung der zwei Ziel-KPIs erreichen. Es wird daher auch von Folgetest 2) abgesehen.

In ihrer breiten Masse werden die funktionsfähigen Leiterplatten mit einer sehr hohen Wiedererkennungsrate und Präzision klassifiziert. Daher ist es naheliegend, dass ein ausreichender Unterschied zwischen den Features beider Klassen vorliegt. Eine mangelhafte Feature Extraktion oder eine zu geringe Anzahl an Features ist daher nicht festzustellen. Als Ursache für die fehlerhafte Einordnung von defekten Bauteilen bleibt noch die geringe Anzahl an Daten über defekte Leiterplatten. Das im Test Design erwähnte Verhältnis von

84:11788 fundiert diese Annahme. Die mangelhafte Klassifikationsgüte für defekte Teile lässt sich daher auf ein klassisches Underfitting zurückführen (siehe Kapitel 2.2).

Anpassung und Ergänzungen

Da die Anzahl an Daten zu defekten Bauteilen eine feste Größe ist, lässt sich zunächst keine nennenswerte Verbesserung der Klassifikationsgüte durch weitere Verarbeitungsschritte oder Parametereinstellungen erreichen. Es können jedoch vielfältige Überlegungen zur Abänderung des Modells angestellt werden.

Das bisher betrachtete Modell nutzte mit dem Ziellabel einer gesamten Prüfreihe (defect/approved) ein besonders hohes Aggregationslevel. Die Ergebnislabel von Prüfvorgängen (pass/reset/fail) und Prüfschritten (PASS/FAIL) bieten weitere Ansatzpunkte für Klassifikationsverfahren. Eine leichte Umgestaltung des bisherigen Modells ermöglicht die Klassifikation der Labels von einzelnen Prüfvorgängen. Die Klassifikationsgüte für *pass* und *reset* ist, wie in den folgenden Abbildungen zu sehen, auch schon nach dem Durchlaufen von wenigen Prüfschritten sehr gut. Das Entscheidungsbaum-Verfahren schneidet dabei über alle Iterationen hinweg am besten ab und könnte zumindest für die frühzeitige Erkennung eines *reset* oder *pass* verwendet werden.

classification report: tree

	precision	recall	f1-score	support
fail	1.00	0.09	0.16	23
reset	0.97	0.65	0.78	651
pass	0.91	1.00	0.95	2299
micro avg	0.92	0.92	0.92	2973
macro avg	0.96	0.58	0.63	2973
weighted avg	0.92	0.92	0.91	2973

classification report: tree

	precision	recall	f1-score	support
fail	1.00	0.11	0.19	19
reset	0.97	0.84	0.90	707
pass	0.95	0.99	0.97	2247
micro avg	0.95	0.95	0.95	2973
macro avg	0.97	0.65	0.69	2973
weighted avg	0.95	0.95	0.95	2973

Abbildung 29: Entscheidungsbaum - 2 Prüfdurchlauf, oben Prüfschritt 2.9, unten 5.9

Rein aus der Bestimmung dieser Ergebnislabels lässt sich jedoch noch kein Mehrwert gewinnen. Der frühzeitige Hinweis auf ein vermutetes Endergebnis verhindert nicht, dass alle Komponenten der Leiterplatte auf Funktionsfähigkeit getestet werden müssen, solange

keine endgültige Aussage über Defekt oder Funktionsfähigkeit möglich ist. Durch eine Informationsanreicherung und Verfeinerung der Labels könnten jedoch nützliche Hinweise gegeben werden. So wäre, ähnlich wie bei der Problemklassifikation eines Defektes (siehe Test 2 des Test Designs), eine Unterteilung der *reset* Klasse möglich. Es könnte beispielsweise in Spannungsfehler, Frequenzfehler, etc. unterteilt werden. Kann der Mitarbeiter frühzeitig gewarnt werden, auf welchen gängigen Problemfall die Abbrüche des Prüfprogrammes zurückzuführen sind, liegt auch in dieser Modellierung ein Potential für die Verbesserung von Durchsatzrate und Materialverbrauch.

4.5 Evaluation und Modelleinsatz: Ergebnisse und Potential

4.5.1 Beurteilung der Ergebnisse

Beurteilung der Data Mining und ML Ergebnisse

Auch wenn die Untersuchung von ML-Potential innerhalb der Machbarkeitsstudie im Vordergrund steht, tragen die Erkenntnisse aus der Datenexploration auch zum Mehrwert der Arbeit bei. Die Erfassung von impliziten Informationen, welche zuvor noch nicht direkt den Logdateien entnommen werden konnten, stellt dabei eine Basis für Ansatzmöglichkeiten der Datenanalyse dar. Neue Attribute wie der Wiederholungsindex (loopindex) und die finale Prüfreihenklassifizierung (final_result) sind zentral für aussagenkräftige Analysen. Darüber hinaus unterstützen die Ergebnisse der Korrelationsuntersuchungen die Suche nach kritischen Einflussfaktoren auf die Mess- und Prüfvorgänge. Nachweisliche Umwelteinflüsse können an Hand der aktuellen Datengrundlage nicht festgestellt werden. Die Korrelationsuntersuchung der Prüfschritte untereinander hat hingegen aussagekräftige Zusammenhänge identifiziert. Mit der Kenntnis über den repetitiven Ausgang von gleichartigen Prüfschritten lassen sich z.B. neue Ansatzpunkte für Stellschrauben der Qualitätssicherung erforschen.

Die Untersuchung der frühzeitigen Detektion von defekten Leiterplatten mit Hilfe von ML-Verfahren hat zentrale Kenntnisse für die Einschätzung des ML-Potentials erbracht. Auf Grund der mangelhaften Datengrundlage zum Trainieren der Defekt-Erkennung, ist dieser Anwendungsfall aktuell nicht umsetzbar. Des Weiteren ist durch den geringen Einfluss einer solchen Anwendung auf die KPIs die Rentabilität kritisch zu betrachten. Diese Beurteilungen betrifft einen recht spezifisches ML-Potential. Im Gegensatz dazu konnte für die frühzeitigen Erkennung von Prüfvorgang-Labels eine vielversprechende

Klassifikationsgüte festgestellt werden. Durch diese Option wurden weitere Anwendungsmöglichkeiten für ML-Potential identifiziert.

Es lässt sich Rückblickend die Bearbeitung der Data Mining Ziele begutachten (grüne Markierung: erfüllt, gelbe Markierung: teilweise erfüllt):

Zu 1)	• Ausreichende Datenquantität und -qualität: Die Begutachtung aus Kapitel 4.2.4 hat diese mit Beschränkung auf große Bauteilordner zunächst bestätigt. Durch das in Kapitel 4.4.2 angesetzte ML-Modell wurde jedoch eine Schwachstelle bezüglich der Datenquantität defekter Bauteile festgestellt. • Entwicklung eines Parsers, prototypischer Schnittstellenbau und Kompatibilitätsanalyse: Der Parser wurde entwickelt und seine Funktionsweise hat bereits Einsatz in dem von parsQube entwickelten Web Service gefunden. Die ML-Schnittstelle kann erfolgreich auf spezifische Anwendungsfälle trainiert werden und ist auf Grund ihrer modularen Funktionen auch für kommende Modellierungen flexibel einsetzbar.
Zu 2)	• Untersuchung von Korrelationen und Mustern: Durch die Exploration erarbeitet. Austausch mit Unternehmen X: Die Korrelationsergebnisse bzgl. der Prüfschritte untereinander wurde von Unternehmen X als informativ bewertet. Zur Erarbeitung einer Deutung müssen die Zusammenhänge noch weiter untersucht werden. Testen von ML-Verfahren auf die Fähigkeit zur Mustererkennung: Innerhalb des Notebooks ML_Prototype und in Orange ansatzweise untersucht. Da erste Tests nicht erfolgreich waren und der Fokus auf Vorhersagefunktionen gelegt wurde, gibt es zu diesem Punkt keine Ergebnisse vorzuweisen.
Zu 3)	• Vorhersagen mit Hilfe von trainierten ML-Verfahren und resultierende Modellgüte: Die Vorhersagen der Zielklasse waren auf Grund der mangelhaften Klassifikationsgüte nicht verwertbar. Hingegen konnte für

	ein Alternativmodell eine hohe Güte und damit ein hohes Potential für möglichen Folgekonzepte festgestellt werden.

Umsetzung der Unternehmensziele

An Hand der in Kapitel 4.1.1 definierten Akzeptanzkriterien werden die Projektergebnisse in den Zusammenhang mit den Unternehmenszielen gebracht:

Zu 1)	• Eignung der Datengrundlage: Trotz der Schwächen in der Erkennung von Bauteil-Eigenschaften aus kleinen Datenmengen, erweist sich die Datenmenge im Ganzen als geeignet für ML-Anwendungen. • Potentielle Implementierung von ML-Schnittstellen: Durch den Nutzen des Parsers für den Web Service, scheint auch für die ML-Schnittstelle aus Jupyter eine zukünftige Integration möglich zu sein. • Rentabilität einer möglichen Implementierung: Das ausgewählte Testszenario konnte keine Rentabilität nachweisen. Auf Grund des spezifischen Anwendungsfalles kann diesbezüglich jedoch noch kein abschließendes Urteil gefällt werden. Das ausstehende ML-Potential ist für die breite Masse an Bauteil-Prüfungen interessant und hätte bei erfolgreicher Weiterentwicklung einen positiven Einfluss auf die betrachteten KPIs.
Zu 2)	• Mustererkennung durch ML: Bei der Erkennung von Klassen ist auf das Aggregationslevel der zu klassifizierenden Eigenschaften zu achten. Es wurde jeweils ein Positiv- und Negativbeispiel vorgezeigt. Ein funktionierender und rentabler Erkennungsmechanismus ist nicht vorhanden, das Positivbeispiel zeigt jedoch Möglichkeiten in diese Richtung auf.
Zu 3)	• Identifikation/Prognostizierung von neuartigen Mustern durch ML: Spezifisch durch ML konnten keine neuen Zusammenhänge identifiziert werden. Die Korrelationsuntersuchungen zu den Prüfschrittzusammenhängen bieten jedoch Ansatzpunkte für die Entwicklung solcher Anwendungen.

Zu 4)	• Keine geforderte Umstellungen der aktuellen Arbeitsprozesse: Die im Testszenario entwickelte Beispielanwendung fungierte als rein informative Ergänzung für die Funktionsfähigkeitstests. Auch ähnliche Anwendungen würden keine Umstrukturierung der Arbeitsprozesse nach sich ziehen, sondern höchstens Abkürzungsmöglichkeiten aufweisen. • Verständlichkeit für die Zielgruppe: Das Testszenario fokussierte sich auf ein Informationstechnische Anwendung für Mitarbeiter im Bereich der Funktionsfähigkeitstests. Durch die Darstellung eines Hinweises für einfache Problemklassen wären der Aussagengehalt verständlich und die Nutzung ohne große Umschulungsmaßnahmen möglich.

Anerkennung des Modells

Die Betrachtung der Data Mining und Unternehmensziele hat einen Mehrwert durch das ML-Modell und seine Alternativmodellierung feststellen können. Die identifizierte Schwachstelle lässt sich nach Modellbeurteilung nicht auf eine falsche Konzeptionierung und Umsetzung zurückführen, sondern besitzt ihren Ursprung in den gegebenen Umständen (bezgl. Datengrundlage). Es folgt die Bewertung der Umsetzung des ML-Modells und der zugehörigen Schnittstelle:

- Inhaltliche Zielgenauigkeit: Im Rahmen der Datenselektion (siehe Kapitel 4.3.1) wurde durch das umfangreiche Auslesen aus den Logdateien auf größtmögliche Vollständigkeit und Unverfälschtheit geachtet. Die Schnittstelle wird somit den Anforderungen nach Inhaltlicher Zielgenauigkeit gerecht.

- Angestrebter Formfaktor: Bei der Auswertung der Informationen wurde sowohl in den Notebooks der Datenexploration, als auch in ML_Prototype auf möglichst anschauliche Direktausgaben geachtet. Die Grafiken und Tabellen wurden in einem möglichst übersichtlichen Stil gestalten und die Arbeitsprozesse durch Kommentare erläutert. Da es keine direkten Formvorgaben gab, ist diese Beurteilung recht subjektiv. Das positive Feedback von Unternehmen X und parsQube Mitarbeitern lässt jedoch auf einen passenden Formfaktor schließen.

- Performance: Zur Beurteilung der Geschwindigkeit der Datenverarbeitung muss zwischen der Performance des Parsers und der einer ML-Anwendung unterschieden werden. Für den Parser wurde auf Grund der langen Laufzeit von ca. vier Stunden (siehe Anhang: Parser_Initiation) eine Zwischenspeicherung eingerichtet. Hingegen kann der anschließende Import nahezu in Echtzeit erfolgen. Die Vorverarbeitungsschritte und das Training der ML-Verfahren liegen in einem für Analysezwecke ausreichenden Zeitraum von wenigen Minuten. Dies ist akzeptabel, da zur Echtzeitverarbeitung von Vorhersagetests schon fertig trainierte ML-Verfahren genutzt werden. Die Schnittstelle wurde somit im Hinblick auf Performance auf den Anwendungsfall optimiert.

- Flexibilität und Anpassungsfähigkeit: Durch den modularen Aufbau der Schnittstelle mit flexiblen Funktionen für unterschiedliche Verarbeitungsschritte, sind Parameter und Inputwechsel durch einfache Austauscharbeiten möglich. Gerade bei der Pfadverwaltung für den Zugriff auf Bauteilordner wurde auf Interoperabilität zwischen Betriebssystemen und Speicherlokalitäten geachtet. Voraussetzung für die Durchführung von Anpassungsarbeiten ist jedoch eine gewisse informationstechnische Vertrautheit, welche durch die Verständlichkeit des Python Codes und mit Hilfe der eingearbeiteten Kommentare nicht allzu ausgeprägt sein muss.

4.5.1 Rezension des Prozessablaufes

Als Machbarkeitsstudie mit explorativem Charakter, kam es an vielen Stellen des Projektablaufes zu einem besonderen Orientierungsaufwand. Die Notwendigkeit das Datenverständnis zunächst von Grund auf zu erarbeiten und die breite Definition der Analyseaufgaben hatte zur Folge, dass die Datenexploration einen Großteil der gesamten Arbeitszeit beanspruchte. Erschwerend kam dazu die Distanz zwischen dem beim Dienstleister parsQube eingestellten Autor und dem Kundenunternehmen X. Die Möglichkeit zur Erfassung von Hintergrundinformationen zu den unternehmensinternen Prozessen bei Unternehmen X, war nur durch stoßweise Austauschmöglichkeiten vorhanden. Auf Grund dessen mussten teilweise mehrere Feedback-Schleifen durchlaufen werden, bis ein Sachverhalt in seiner Tiefe verstanden wurde.

Bewertet man rückblickend die Arbeitsschritte der Datenvorverarbeitung, so war der Arbeitsaufwand geringer als zu Beginn erwartet. Dies lässt sich auf zwei Faktoren zurückführen. Zunächst weißt die Datengrundlage eines streng überwachten und stark

automatisierten Prüfprozesses eine geringe Anzahl an Fehlern auf. Darüber hinaus gestaltete sich die Datenextraktion aus den Logdateien auf Grund der strukturellen Übersichtlichkeit des XML-Formates als sehr unkompliziert. Der zweite hilfreiche Faktor waren die vielfältigen und benutzerfreundlichen Verarbeitungsfunktionen des tabellarischen pandas-Formates. Sie ließen flexible und schnell ausführbare Filteranfragen auch bei größeren Datensätzen zu.

Für den Aufgabenbereich der Modellierung im Rahmen einer Machbarkeitsstudie lässt sich eine essentielle Schwierigkeit festhalten. Nachdem sich durch die Datenexploration und Vielfalt an Analysemöglichkeiten unterschiedliche Modellierungsansätze ergeben haben, fällt die Wahl eines Modells zur ausführlichen Betrachtung nicht leicht. Im Falle dieser Studie wurde zum Ende hin deutlich, dass ein Modell zwar ein größeres Potential besaß, es aber an einem Anwendungsfall für ein konkretes Test Design fehlte. Für das Test Design musste sich demnach auf ein weniger rentables, aber dafür konkreteres Modell gestützt werden. Neben den Ergebnissen aus dem Test Design konnte das vielversprechendere Modell aber als Kandidat für die weitere Potentialbetrachtung verwendet werden. Der zentrale Erfahrungswert ist daher, dass nicht nur trotz, sondern gerade wegen dem Kompromiss ein ausgewogenes und fundiertes Studienergebnis erzeugt werden konnte.

5. Zusammenfassung und Ausblick

Wie in der Einleitung der Arbeit beschrieben, lässt sich der Mehrwert der Abhandlung zweiteilen. Der Erkenntnisgewinn über das aktuelle ML-Potential für den spezifischen Anwendungsfall von Unternehmen X ist der erste, direkt verwertbare Nutzen. Die Zusammenfassung aller diesem Projekt zugehörigen Zielsetzungen, Arbeitsschritte und Ergebnisse wurde bereits in Kapitel 4.5 ausführlich behandelt. Es bleibt daher noch die rückblickende Betrachtung des in Kapitel 3 präsentierten Referenzmodells. Im Fokus der Zielsetzung für dieses Modell stand die Anforderung, dass die erarbeitete Methodik als praxisnahe Orientierung für Vergleichsprojekte dienen kann. Die Arbeitsschritte der Methodik sollten sich dabei auf eine standardisierte und etablierte Struktur stützen.

Der anfänglich durch CRISP-DM definierte Rahmen, unterstützte den Autor dabei aus Erfahrungswerten des Beispielprojektes und Techniken des Data Mining ein vollwertiges Referenzmodell zu entwickeln. Während dem Ablauf des Projektes wurden die generischen Prozessschritte nach CRSIP-DM durchlaufen und das Modell mit problemspezifischen Aktivitäten und Empfehlungen gefüllt. Gleichzeitig konnten die Praxisnähe und

Leitfähigkeit des Referenzmodells getestet werden. In das Endresultat aus Kapitel 3 wurden nur die essentiellen Arbeitsschritte und Empfehlungen aufgenommen, deren Mehrwert zuvor durch den Anwendungsfall verifiziert worden war. Das entstandene Modell und seine Methodik sind damit durch eine standardisierte Struktur und die Bestätigung eines realen Anwendungsfalles fundiert. Es ist anzumerken, dass diese Arbeitsschritte in dem Aufbau eines etablierten Modellstandards nur den Anfang repräsentieren. Zur Durchführung einer wissenschaftlich akzeptierten Verifikation müssten noch viele weitere Anwendungsfälle das Referenzmodell durchlaufen und seinen praktischen Mehrwert nachweisen. Im Rahmen der Aufgabenstellung wurde die Zielsetzung aber erreicht. Das Testen des Referenzmodells in unterschiedlichen Anwendungsszenarien mit gleichen Projektcharakteristiken (XML-Standardisierte Mess- und Prüfdaten) könnte eine Zielsetzung für zukünftige Arbeiten darstellen.

5.1.1 Mögliche nächste Schritte - Ausblick

Im Folgenden werden mögliche Folgeschritte gemäß Data Mining Kreislaufes nach Kapitel 4 abgeleitet. Die Untersuchung des ML-Potentials für eine Anwendung bei Unternehmen X kann, bis auf die durch das Test Design aufgewiesenen Restriktionen, als positiv bewertet werden. Besonders die solide Datenerfassung und die zunehmende Datenmenge durch fortschreitende digitale Integration lassen auf eine gute Grundlage für mögliche ML-Anwendungen schließen. Das Data Mining Projekt hat darüber hinaus eine Bandbreite an Ansatzpunkten für weitere, vielversprechende Potentialuntersuchungen ergeben. Im Sinne des Data Mining Kreislaufes wären Folgeprojekte, welche eben diese Ansatzpunkte aufnehmen, eine Möglichkeit das Potential bei Unternehmen X weiter auszuloten. Das erarbeitete Referenzmodell könnte gerade in solchen Projekten seinen Nutzen finden und gleichzeitig auch verifiziert oder optimiert werden.

Von den zukünftigen Data Mining Untersuchung abgesehen, lassen sich durch die Machbarkeitsstudie auch schon konkrete Möglichkeiten für eine praktische Umsetzung erkennen. Die Funktionen der entwickelten Schnittstelle konnten im Falle des Parsers schon teilweise in den von parsQube aufgesetzten Web Service bei Unternehmen X integriert werden. Diese Applikation hat zwei Implementierungen für unterschiedliche Zwecke. Eine Browseranwendung ermöglicht rückblickende Datenanalysen und grafische Aufbereitungen für die Qualitätssicherung. Die zweite Anwendung ist für Displayanzeigen direkt bei den Testsystemen der Produktion konzipiert und ermöglicht fortlaufend aktualisierte Auswertungen. Falls die ML-Funktionen der Jupyter-Schnittstelle weiterentwickelt und in

den Web Service integriert werden könnten, würde die zweite Anwendung in der Produktion von zusätzlichen Funktionalitäten profitieren.

Werden zum Abschluss die Arbeit und das zugehörige Projekt noch einmal in ihrer Gesamtheit betrachtet, lässt sich der eindeutige Charakter einer Machbarkeitsstudie erkennen. Es wurde ein Anwendungsfall mit viel Explorationsbedarf auf praktische Umsetzungen eines umfangreichen Themas untersucht. Das erarbeitete Ergebnis bestätigt die Anwendbarkeit von ML-Verfahren auf produktionserzeugte Mess- und Prüfdaten von Unternehmen X. Es stellt gleichzeitig neue Fragen, welche sowohl für das entwickelte Referenzmodell, als auch für das übergreifende Projekt von parsQube und Unternehmen X weiteren Forschungsbedarf feststellen. In diesem Sinne lässt sich auf kommende Erkenntnisse durch einen beständigen Entdeckergeist hoffen.

> “The business would do good to understand that success or failure is not final in Data Science. For this reason, the business should develop a persistent spirit.”
> — Damian Mingle

Literaturverzeichnis

Berthold, M. & Hand, D. J., 2003. *Intelligent Data Analysis: An Introduction.* 2 ed. Berlin: Springer.

Briscoe, G. & Caelli, T., 1996. *A Compenium of Machine Learning - Volume 1: Symbolic Machine Learning.* 1 ed. New Jersey: Norwood.

Copeland, L., 2004. *A practitioner's guide to software test design.* Boston; London: Artech House.

Davenport, T. H. & Patil, D. J., 2012. Data Scientist: The Sexiest Job of the 21st Century. *Harvard Business Review,* October.

Derksen, O. et al., 2013. *Big Data - Systeme und Prüfung.* Berlin: Erich Schmidt Verlag.

DIN IEC 62198, D. I. f. N., 2002. *DIN IEC 62198.* s.l. Patent No. AC-Code: DE45760721.

Duda, R. O., Hart, P. E. & Stork, D. G., 2001. *Pattern Classification.* XX ed. New York: John Wiley & Sons Inc..

Kalogeras, A. P., Gialelis, J. V. & Alexakos, C. E., May 2006. Vertical Integration of Enterprise Industrial Systems Utilizing Web Services. *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS,* 2(2).

Kang, N., Zhao, C., Li, J. & Horst, . J. A., 2016. A Hierarchical structure of key performance indicators for operation management and continuous improvement in production systems. *International Journal of Production Research,* Issue 54, p. 6333–6350.

Litke, H.-D., 2007. *Projektmanagement - Methoden, Techniken, Verhaltensweisen.* 5. Auflage ed. München: Carl Hanser Verlag.

Müller, A. C. & Guido, S., 2017. *Einführung in Machine Learning mit Python - Praxiswissen Data Science.* 1 ed. Heidelberg : dpunkt.verlag Gmbh.

Michalski, R. S., Mitchell, T. & Carbonell, J. G., 1983. *Machine Learning - An Artificial Intelligence Approach.* 1 ed. Palo Alto(Calif.): Tioga Publ. Co..

Mitchell, T. M., 1997. *Machine Learning.* XVII ed. Boston(Mass.): McGraw-Hill Science.

Mitchell, T. M. & Jordan, M. I., 2018. Machine Learning: Trends, perspecitives and prospects. *Science,* 17 July, 349(6245), pp. 255-261.

Oettinger, M., 2017. *Data Science - Eine praxisorientierte Einführung im Umfeld von Machine Learning, künstlicher Intelligenz und Big Data.* 1 ed. Hamburg: tredition GmbH.

Petersohn, H., 2005. *Data Mining - Verfahren, Prozesse, Anwendungsarchitektur.* München: Oldenbourg Wissenschaftsverlag GmbH.

Prakash, N. & Prakash, D., 2018. *Data Warehouse Requirements Engineering.* s.l.:Springer Nature.

Salem, M. V. M. et al., 2011. *Information Retrieval Technology.* Heidelberg: Springer-Verlag.

VanderPlas, J., 2016. *Python Data Science Handbook.* 1. Auflage ed. s.l.:O'Reilly Media.

Vapnik, V. N., 1998. *Statistical Learning Theory.* XXIV ed. New York: John Wiley & Sons Inc..

Vermeulen, A. F., 2018. *Practical Data Science: A Guide to Building the Technology Stack for Turning Data Lakes into Business Assets.* 1 ed. Berkeley, CA: Apress.

Wirth, R. & Hipp, J., 2000. CRISP-DM: Towards a Standard Process Model for Data Mining. *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining,* pp. 29-39.

Witten, I. H. & Frank, E., 2001. *Data Mining - Pratkrische Werkzeuge und Techniken für das maschinelle Lernen.* 1 ed. San Francisco: Carl Hanser Verlag München Wien.

Anhang

Jupyter Notebooks, Python und Orange Dateien	
Parser_Initiation.ipynb	Initialisiert den Parser über alle Bauteilordner – im Anhang 3 Seiten (1-3)
Implizite_Datenanalyse.ipynb	Datenexploration zur Erkundung des Datensatzes – im Anhang 19 Seiten (4-22)
Korrelation_Step_Ambient.ipynb	Korrelationsuntersuchung mit den Umgebungsdaten von Raumsensoren – im Anhang 13 Seiten (23-35)
Korrelation_Step_Temperatur.ipynb	Korrelationsuntersuchung mit Temperaturdaten direkt vom Testsystem – im Anhang 9 Seiten (36-44)
Korrelation_Steps_untereinander.ipynb	Korrelationsuntersuchung bezüglich der gegenseitigen Beeinflussung der Prüfschritte – im Anhang 7 Seiten (45-51)
ML_Preprocessing.ipynb	Datenvorverarbeitung als Vorarbeit für ML_Prototype – im Anhang 6 Seiten (52-57)
ML_Prototype.ipynb	Aufsetzen von ML-Verfahren zur Klassifikation – im Anhang 8 Seiten (58-65)
rhodos.py	Ausgelagerte Import und Verarbeitungsfunktionen – im Anhang 3 Seiten (66-68)
regression_functions.py	Ausgelagerte Regressionsfunktionen – im Anhang 8 Seiten (69-76)
ml_functions.py	Ausgelagerte Vorverarbeitung und mehr – im Anhang 5 Seiten (77-81)

ambient.py	Importfunktion für Raumsensordaten – im Anhang 1 Seite (82)
ML_testing.ows	Orange Datei zur ML-Modellierung, als Netzwerkdarstellung nur digital mit der mitgelieferten Datei einsehbar