BERGISCHE
UNIVERSITÄT
WUPPERTAL

Fakultät für Maschinenbau und Sicherheitstechnik

Masterthesis

im Studiengang **Sicherheitstechnik / Qualitätsingenieurwesen**
beim Fachgebiet für **Verkehrssicherheit und Zuverlässigkeit**

zur Erreichung des akademischen Grades

Master of Science

Thema:**Zuverlässigkeitstechnik in der Industrie 4.0 und Prädiktive Instandhaltung mit Anwendung von maschinellen Lernalgorithmen**

Autor*in: **Hossein Nafar**

MatNr: 1838552

Bearbeitungszeitraum: **15. März 2024** bis **05. Juli 2024**

Betreuer*in: Herr Jun.-Prof. Dr. Tordeux, Antoine

Erstprüfer*in: Herr Jun.-Prof. Dr. Tordeux, Antoine

Zweitprüfer*in: Herr M.Sc. Julitz, Tim M.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die von mir eingereichte Abschlussarbeit (Master-Thesis) selbstständig verfasst und keine andere als die angegebenen Quelle und Hilfsmittel benutzt sowie Stellen der Abschlussarbeit, die anderen Werken dem Wortlaut oder Sinn nach entnommen wurden, in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ich bin damit einverstanden, dass die Arbeit durch Dritte eingesehen und unter Wahrung urheberrechtlicher Grundsätze zitiert werden darf.

Ort und Datum: wuppertal. Den 05.07.2024

Unterschrift: 

Abkürzungsverzeichnis

- **Bagging**: Bootstrap Aggregation
- **Big Data**: Große Datenmengen
- **Coolant**: Kühlmittel (Coolant)
- **CPS**: Cyber-Physische Systeme
- **DL**: Deep Learning
- **EN**: Europäische Norm (European Norm)
- **FN**: False Negatives (Falsche Negative)
- **FP**: False Positives (Falsche Positive)
- **Fuel**: Kraftstoff (Fuel)
- **I4.0**: Industrie 4.0
- **IoT**: Internet der Dinge
- **IQR**: Interquartilsabstand (Interquartile Range)
- **KI**: Künstliche Intelligenz
- **KNN**: k-Nearest Neighbors
- **Lub oil**: Schmieröl (Lub oil)
- **ML**: Maschinelles Lernen
- **NLP**: Verarbeitung natürlicher Sprache (Natural Language Processing)
- **P(y|x)**: Bedingte Wahrscheinlichkeit
- **PdM**: Predictive Maintenance (Prädiktive Instandhaltung)
- **PM**: Preventive Maintenance (Präventive Instandhaltung)
- **RF**: Reinforcement Learning (verstärkendes Lernen)
- **RM**: Reactive Maintenance (Reaktive Instandhaltung)
- **Rpm**: Motordrehzahl (Engine rpm)
- **SML**: Supervised Machine Learning (überwachtes maschinelles Lernen)
- **SVM**: Support Vector Machine
- **Temp**: Temperatur (temp)
- **TN**: True Negatives (Wahre Negative)
- **TP**: True Positives (Wahre Positive)
- **UML**: Unsupervised Machine Learning (unüberwachtes maschinelles Lernen)

Tabellenverzeichnis

Abbildungsverzeichnis

Inhaltsverzeichnis

Zusammenfassung

Zusammenfassung

Die industrielle Entwicklung von 'Industrie 1.0' zu 'Industrie 4.0' hat bedeutende technologische Fortschritte hervorgebracht. Im Zeitalter von Industrie 4.0 spielen Automatisierung, Vernetzung und intelligente Systeme eine zentrale Rolle, insbesondere in Form von cyber-physischen Systemen und vorausschauender Wartung. Letztere ermöglicht durch maschinelles Lernen und das Internet der Dinge eine proaktive Wartung und die Vermeidung von Ausfällen. Unternehmen stehen vor der Herausforderung, die Zuverlässigkeit ihrer Anlagen durch diese Technologien zu verbessern. Diese Masterarbeit untersucht die Anwendung von Algorithmen des maschinellen Lernens zur Effizienzsteigerung von vorausschauenden Instandhaltungsstrategien, um die Zuverlässigkeit und Produktivität in der modernen Industrie zu erhöhen.

Summary

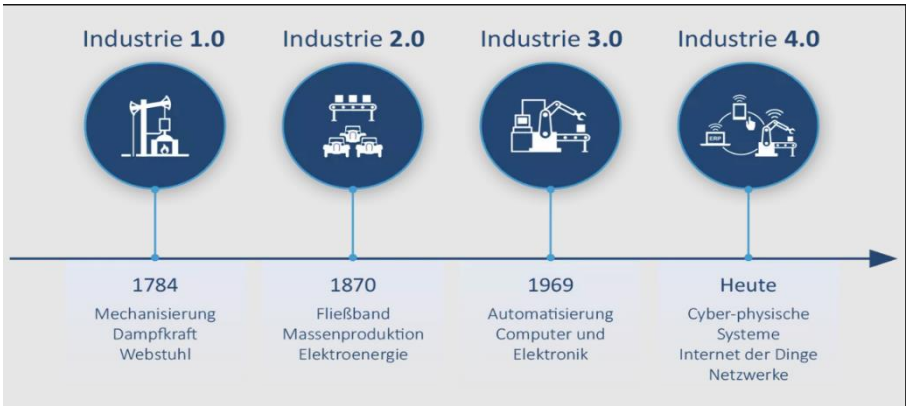
The industrial development from 'Industry 1.0' to 'Industry 4.0' has resulted in significant technological advances. In the age of Industry 4.0, automation, networking and intelligent systems play a central role, particularly in the form of cyber-physical systems and predictive maintenance. The latter uses machine learning and the Internet of Things to enable proactive maintenance and avoid breakdowns. Companies are faced with the challenge of improving the reliability of their systems using these technologies. This master's thesis investigates the application of machine learning algorithms to increase the efficiency of predictive maintenance strategies in order to increase reliability and productivity in modern industry.

1 Einleitung

1.1 Ausgangspunkt

Die industrielle Landschaft hat im Laufe der Geschichte transformative Phasen durchlaufen, die von bedeutenden technologischen Fortschritten geprägt waren. Von der Mechanisierung der Arbeit in der Ära der 'Industrie 1.0' bis zur Automatisierung von Prozessen in der 'Industrie 3.0' wurde unaufhaltsam nach Effizienzsteigerung und Produktivitätsverbesserung gestrebt [23].

Mit dem Übergang von 'Industrie 3.0' zu 'Industrie 4.0' betreten wir eine neue Ära der industriellen Revolution. Die Entwicklung von Automatisierung, Vernetzung und Intelligenz hat zur Entstehung von 'cyber-physischen Systemen' (CPS) und intelligenten Fabriken geführt. Diese basieren auf der Verbindung von intelligenten Maschinen, dem Internet der Dinge (IoT) und leistungsstarken Algorithmen [23]. Die Abbildung 1.1 veranschaulicht den Verlauf des industriellen Wandels von der ersten bis zur vierten industriellen Revolution.



1.1: Industrieller Wandel und Entwicklungsverlauf [24].

In einem anspruchsvollen Geschäftsumfeld mit globaler Vernetzung und zunehmendem Variantenreichtum streben Unternehmen nach kundenindividuellen Produkten und verkürzten Produktlebenszyklen. Zur Bewältigung dieser Anforderungen setzen sie auf Lean-Prinzipien zur Prozessoptimierung und hoch automatisierte Maschinen. In diesem Zusammenhang hat die Instandhaltung eine entscheidende Funktion. Sie sorgt für die Verfügbarkeit und Effizienz von Maschinen [40]. Diese Dimension gewinnt zunehmend an Relevanz und unterstreicht die zentrale Bedeutung des Themas Instandhaltung in der modernen industriellen Welt.

In dieser fortschrittlichen Umgebung spielt die prädiktive Instandhaltung eine entscheidende Rolle in der industriellen Landschaft. Sie ermöglicht proaktive Eingriffe und Reparaturen, um Ausfälle von Anlagen sowie Qualitätsminderungen frühzeitig zu antizipieren.

Durch die Integration von Sensoren und Übertragungstechnik werden erhebliche Datenmengen generiert, was im Kontext der Industrie 4.0 und des Internet der Dinge (IoT) von großer Bedeutung ist. Dies ermöglicht Ingenieuren und Ingenieurinnen, detaillierte Einblicke in Betriebszustände, Verwendungszwecke, Energieverbrauch und weitere Parameter von Geräten zu gewinnen [44].

1.2 Problemstellung

Die fortschreitende Einführung von Industrie 4.0-Technologien stellt Unternehmen vor neuen Herausforderungen, insbesondere im Bereich der Zuverlässigkeitstechnik und prädiktiven Instandhaltung. Es ist wichtig, die Sicherheit und Funktionsfähigkeit von Anlagen zu gewährleisten, insbesondere angesichts der wachsenden Bedeutung von Schlüsselthemen wie maschinelles Lernen. In dieser dynamischen Umgebung liegt der Fokus auf der Entwicklung prädiktiver Instandhaltungsstrategien, die den speziellen Anforderungen von Industrie 4.0 gerecht werden.

Die vorliegende Masterarbeit untersucht, wie Unternehmen in der Ära von Industrie 4.0 die Zuverlässigkeit ihrer Technik durch den gezielten Einsatz maschineller Lernalgorithmen für prädiktive Instandhaltung verbessern können. Unternehmen stehen vor der Herausforderung, die Möglichkeiten und Grenzen des maschinellen Lernens zu verstehen und effektiv zu nutzen. Die zentrale Frage lautet: Wie können maschinelle Lernalgorithmen dazu beitragen, die Zuverlässigkeit von Anlagen in Industrie 4.0 zu steigern und prädiktive Instandhaltungsstrategien effizienter zu gestalten?

1.3 Zielsetzung

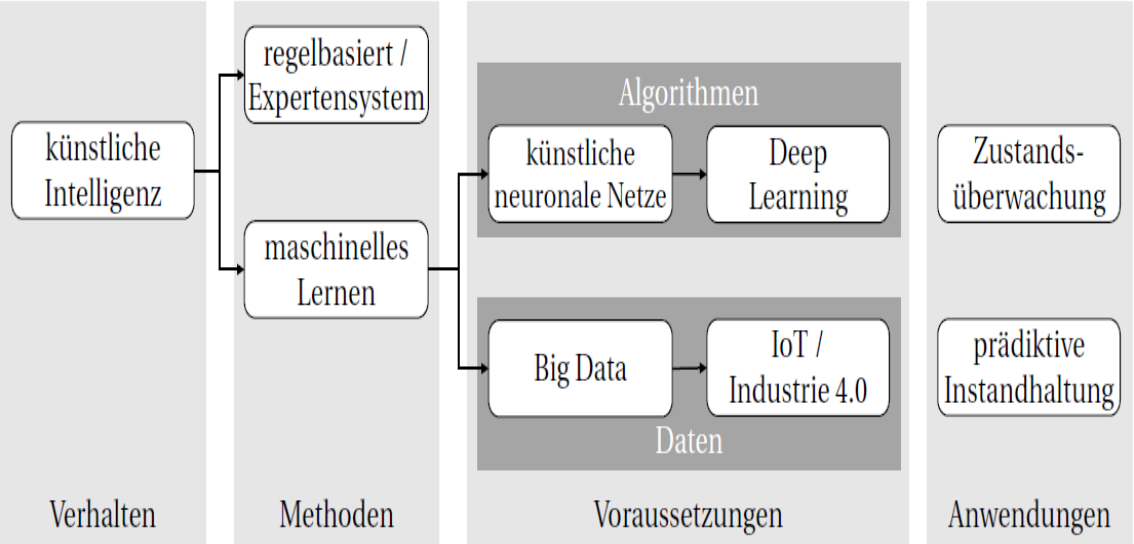
Das Hauptziel dieser Masterarbeit besteht darin, einen umfassenden Einblick in das Thema prädiktive Instandhaltung zu geben und dabei insbesondere die Anwendung von maschinellen Lernalgorithmen zur Entwicklung prädiktiver Instandhaltungsstrategien zu erforschen. Dies wird durch die Durchführung eines Anwendungsbeispiels mit Hilfe eines Datensatzes und maschineller Lernalgorithmen erreicht.

1.4 Aufbau der Arbeit

Die vorliegende Masterarbeit ist in vier Hauptteile gegliedert. Der erste Abschnitt behandelt die Einleitung, in der der Ausgangspunkt, die Problemstellung und die übergeordnete Zielsetzung der Arbeit erörtert werden. Der zweite Abschnitt widmet sich den theoretischen Grundlagen. Diese Arbeit behandelt Themen wie Industrie 4.0, Zuverlässigkeitstechnik, prädiktive Instandhaltung, maschinelles Lernen in der Instandhaltung sowie weitere relevante Aspekte. Der dritte Abschnitt fokussiert sich auf die Anwendung. Hier wird die angewandte Methodik detailliert erläutert, Anwendungsbeispiele werden durchgeführt, und die erzielten Ergebnisse werden präsentiert. Abschließend wird im vierten und letzten Teil ein Fazit präsentiert, das die erlangten Ergebnisse reflektiert.

2 Theoretische Grundlagen

Um die Zusammenhänge zwischen verschiedenen Themenbereichen im Kontext der Zuverlässigkeitstechnik innerhalb der Industrie 4.0 und den damit verbundenen Konzepten, wie beispielsweise der prädiktiven Instandhaltung mittels maschineller Lernalgorithmen, zu erläutern, wird eine grafische Darstellung herangezogen. Die Abbildung 2.1 veranschaulicht die dargestellten Zusammenhänge und verdeutlicht die gegenseitigen Abhängigkeiten, um das Verständnis des Gesamtzusammenhangs zu erleichtern. Im weiteren Verlauf dieses Abschnitts erfolgt eine Definition der genannten Themen, da diese den Kontext und den Fokus der vorliegenden Arbeit am besten widerspiegeln.



2.1Terminologie [43].S.3.

Im Vorfeld ist darauf hinzuweisen, dass der Begriff „Industrie 4.0“ in unterschiedlichen Kontexten verwendet wird. Im Zentrum der Anwendung von künstlicher Intelligenz im Produktionsumfeld steht die sogenannte Smart Factory, auch als „intelligente Fabrik“ oder „vernetzte Fabrik“ bezeichnet. In Deutschland wurde hierfür der Begriff „Industrie 4.0“ geprägt. Der Fokus liegt dabei auf der Vernetzung von Fertigungstechnologien [41].

2.1 Industrie 4.0

2.1.1 Smart Manufacturing

Smart Manufacturing bezeichnet eine moderne Produktionsform, die gegenwärtige und zukünftige Fertigungsanlagen durch die Integration von Sensoren, Computerplattformen, Kommunikationstechnologien, Steuerungssystemen, Simulation, datenintensiver Modellierung und prädiktivem Engineering optimiert [33].

Im Vergleich zu traditionellen Fertigungssystemen weist Smart Manufacturing eine Reihe von Vorteilen in Echtzeit auf, die sich auf die Qualität, Zeit, Ressourcen und Kosten beziehen. Die genannten Vorteile führen dazu, dass Smart Manufacturing nach nachhaltigen und serviceorientierten Geschäftspraktiken konzipiert wird. Der Ansatz basiert auf den Prinzipien der Anpassungsfähigkeit, Flexibilität, Selbstanpassungsfähigkeit, Lernfähigkeit, Fehlertoleranz und des Risikomanagements, welche durch ein flexibles Netzwerk von CPS-basierten Fertigungseinheiten ermöglicht werden [49].

Im vorliegenden Zusammenhang werden, wie oben erwähnt, Konzepte cyber-physischer Systeme verwendet, die durch das Internet der Dinge, Cloud Computing, serviceorientiertes Computing, künstliche Intelligenz und Datenwissenschaft geprägt werden. Im weiteren Verlauf der Arbeit erfolgt eine detaillierte Betrachtung cyber-physischer Systeme im Kontext der Technologien der Industrie 4.0.

Smart Manufacturing führt zu einer Minimierung von Verschwendungen jeglicher Art, fördert kontinuierliche Verbesserungsprozesse, setzt das Null-Fehler-Prinzip um und optimiert visuelles Management. Die in Echtzeit erfassten Produktionsdaten liefern zuverlässige Kennzahlen und stets aktuelle Soll-Ist-Vergleiche für das Shop Floor Management. Die Anwendung von Analyse- und Prognosetechniken in der Fertigungssteuerung ermöglicht die Realisierung einer hohen Prozesseffizienz, auch bei der Fertigung von Produkten in kleineren Losgrößen und mit kürzeren Produktlebenszyklen. [69].

Das Konzept des Smart Manufacturing lässt sich in sechs zentrale Säulen unterteilen:

Fertigungstechnologie und -prozesse, Materialien, Daten, Predictive Engineering, Nachhaltigkeit sowie die gemeinsame Nutzung und Vernetzung von Ressourcen stellen die wesentlichen Säulen des Konzepts dar [34].

2.1.1 Vernetzte Systeme und IoT (Internet of Things)

Die Abbildung 2.1.2.1 veranschaulicht die Entwicklung von Internet-Wellen und die daraus hervorgegangenen Geschäftsmodelle im Zeitverlauf. Ein neues digitales Geschäftsmodellmuster Web.3.0, das in diesem Kontext entstanden ist, ist das Internet der Dinge (Internet of Things, IoT). In diesem Kontext wird das Internet der Dinge (IoT) als ein Modell definiert, bei dem Sensoren zur Wertschöpfung beitragen [20].

3-2.1.1: Internet-Wellen und daraus neu entstandene digitale Geschäftsmodellmuster [20].S.814.

Eine allgemein anerkannte Definition des Begriffs „Internet der Dinge" existiert nicht. Verschiedene Gruppen wie Akademiker, Forscher, Praktiker, Innovatoren, Entwickler und Unternehmensleute haben jeweils ihre eigenen Definitionen formuliert. Zusammengefasst lässt sich sagen, dass das Internet der Dinge aus verschiedenen Perspektiven unterschiedlich betrachtet und interpretiert wird [42].

Der Begriff „Internet der Dinge" (IoT) bezeichnet ein globales Netzwerk von Geräten, zu denen unter anderem Sensoren und Aktoren zählen. Diese Geräte sind mittels standardisierter Internet-Protokolle miteinander verbunden [57], wobei der Informationsaustausch innerhalb des Netzwerks oder mit einem zentralen Speichersystem erfolgt. Dieser Informationsaustausch ermöglicht es den Geräten, in Zusammenarbeit zu agieren oder bestimmte Funktionen zu erfüllen. Folglich generiert das IoT-Netzwerk eine signifikante Anzahl an Transaktionen zwischen verschiedenen Sensoren und Geräten. Die Daten werden anschließend an eine Cloud-Datenbank übermittelt, um dort analysiert und überwacht zu werden [51].

Das Internet der Dinge eröffnet eine Vielzahl von Anwendungsmöglichkeiten und fördert die Integration von physischer und digitaler Welt. Dies führt insbesondere zu einer optimierten Kommunikation und Interaktion zwischen Individuen, Objekten und Unternehmen [30].

2.2 Technologien der Industrie 4.0

2.2.1 Cyber-physische Systeme (CPS)

Zur Verbesserung der Instandhaltung sind vor allem die Verfügbarkeit von Informationen, vorausschauende Instandhaltungsstrategien und eine optimierte Informationsbereitstellung von Bedeutung. Diese Aspekte können auf technischer Ebene durch den Einsatz spezieller cyber-physischer Systeme umgesetzt werden [41].

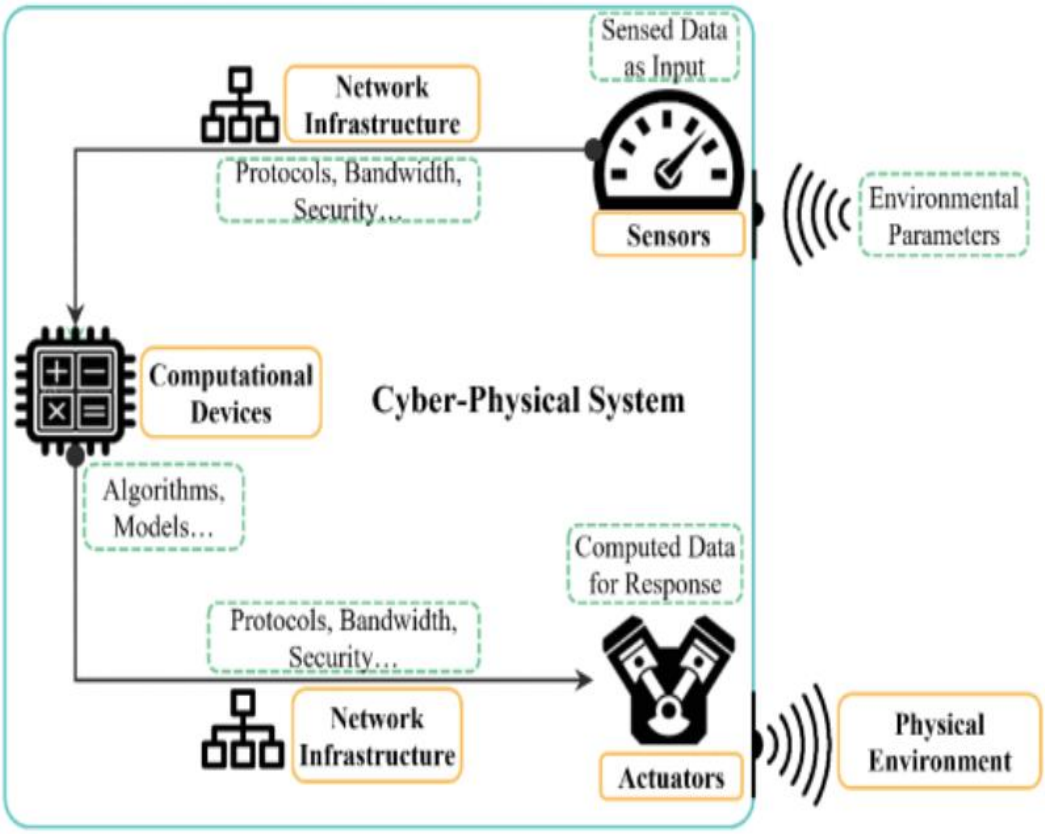
Eine besondere Form von Cyber-Physical Systems (CPS), die als intelligente Bestandteile von Maschinen fungieren, sind Smart Components (intelligente Komponenten). Sie können als Smart Devices oder als verteilte Anwendungen realisiert werden. In der Instandhaltung erhöhen intelligente Komponenten die Verfügbarkeit von Informationen und bilden damit die Grundlage für die Umsetzung prädiktiver Instandhaltungsstrategien [41].

Cyber-physische Systeme (CPS) tragen wesentlich zur digitalen Transformation der industriellen Wertschöpfung im Kontext von Industrie 4.0 bei. Diese Systeme integrieren verschiedene technologische Ansätze, einschließlich Big-Data-Analyse und künstlicher Intelligenz, und ermöglichen so eine verbesserte Echtzeitüberwachung und -steuerung von Fertigungsprozessen [56].

Dabei gelten Cyber-physische Systeme (CPS) als entscheidende Triebkraft für eine neue Ära der internetbasierten Echtzeit-Kommunikation und Kooperation zwischen den Akteuren der Wertschöpfungskette, einschließlich Geräten, Systemen, Organisationen und Menschen. Die Einführung von CPS in industriellen Umgebungen hat das Potenzial, die Art und Weise, wie Unternehmen ihre Prozesse aus einer ganzheitlichen Perspektive gestalten, grundlegend zu verändern. Dies gilt für alle Bereiche, von der Fertigung über die Beziehungen zu Lieferanten und Kunden bis hin zu geschäftlichen Interaktionen und von der Entwicklung bis zum Support über den gesamten Lebenszyklus von Produkten und Dienstleistungen [11].

Die Abbildung 2.2.1.1 veranschaulicht den Aufbau und die Funktionsweise eines Cyber-Physical Systems (CPS). Diese Systeme nutzen fortschrittliche Techniken, um die von den Sensoren erfassten Rohdaten zu verarbeiten und in praktisch nutzbare Informationen umzuwandeln, die oft erst durch intelligente Analyse zugänglich gemacht werden [60]. Die exakte und verlässliche Erfassung von Daten aus Maschinen und ihren Komponenten bildet den ersten Schritt bei der Entwicklung von Anwendungen für

Cyber-Physische Systeme [38]. Eine schematische Darstellung veranschaulicht die Verknüpfung von physischen Komponenten wie Sensoren, Aktoren, Rechengeräten und Netzwerkinfrastruktur mit den Cyber-Komponenten. Letztere umfassen erfasste und berechnete Daten, Algorithmen, Systemmodelle, Netzwerkprotokolle und Bandbreite.[60].



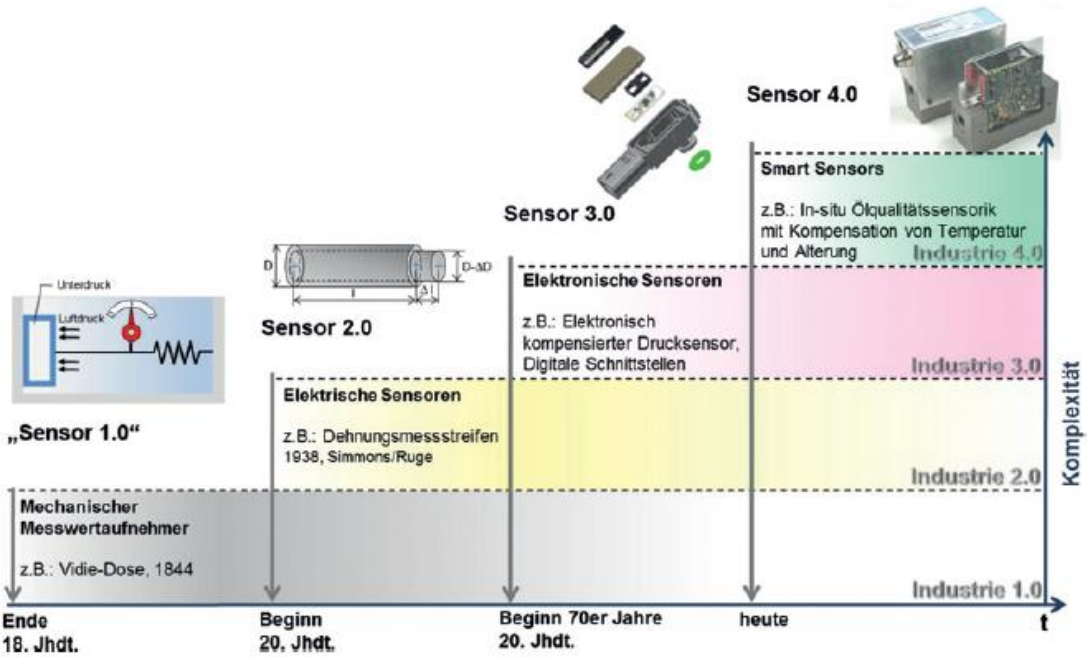
4-2.2.1.1: Ein cyber-physisches System in schematischer Darstellung [58].S.2.

2.2.1 Industrie 4.0 Sensorik und Messtechnik

Im vorhergehenden Abschnitt wurde die Bedeutung von cyber-physischen Systemen (CPS) hervorgehoben. Diese Systeme nutzen fortgeschrittene Techniken, um die von Sensoren erfassten Rohdaten zu verarbeiten und in praktisch nutzbare Informationen umzuwandeln [60]. Dies unterstreicht die Relevanz der Sensorik im Kontext von Industrie 4.0. Im Zusammenhang mit Industrie 4.0 und den notwendigen Voraussetzungen taucht häufig der Begriff Sensorik auf. Die Sensorik und die damit verbundene Messtechnik spielen eine zentrale Rolle, um die wertvollen Daten im

Zeitalter der Digitalisierung zu erfassen und daraus die notwendigen Informationen zu gewinnen. Intelligente Sensoren bieten im Vergleich zu herkömmlichen elektronischen Sensoren erweiterte interne Fähigkeiten. Dazu gehören die Erfassung und Verknüpfung mehrerer Parameter sowie die Fähigkeit zur Selbstdiagnose. Darüber hinaus verfügen sie über erweiterte Kommunikationsfähigkeiten, die nicht nur die Ausgabe von Messwerten, sondern auch die Parametrierung des Messsystems ermöglichen.

Die historische Entwicklung der Sensorik, wie in Abbildung 2.2.2.1 dargestellt, zeigt deutlich den engen Zusammenhang zwischen der Entwicklung der Sensorik und der allgemeinen Industrialisierung [65].



5-2.2.2.1: Die historische Entwicklung des „Sensor 1.0“ zu dem Sensor „4.0“ [63]S.311.

Im nachfolgenden Abschnitt erfolgt eine Auseinandersetzung mit der Bedeutung von Big Data und Datenanalyse, welche als Schlüsseltechnologien zu betrachten sind, um die immense Datenmenge effizient zu verarbeiten und wertvolle Erkenntnisse zu gewinnen.

2.2.1 Big Data und Datenanalyse

Der Begriff „Big Data“ bezeichnet Datensätze, die aufgrund ihrer Größe und Komplexität mit herkömmlichen Datenbankverwaltungskonzepten und -werkzeugen nicht mehr effizient verwaltet werden können. Die Herausforderungen erstrecken sich auf die Erfassung, Speicherung, Suche, Freigabe, Analyse und Visualisierung der Daten [68].

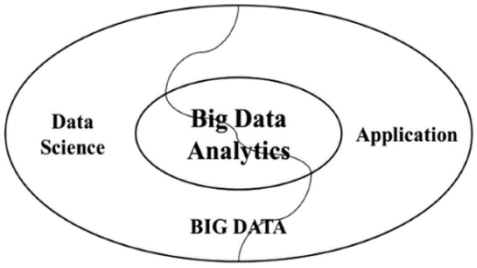
Die umfangreichen Datenmengen werden auf Basis diverser Quellen generiert, darunter Handheld-Geräte, soziale Netzwerke, das Internet der Dinge, Multimedia und zahlreiche weitere moderne Anwendungen. Diese Daten sind gekennzeichnet durch ihre immense Menge, ihre hohe Geschwindigkeit der Generierung sowie ihre Diversität [73]. Die Anwendung von Big-Data-Analysetechniken ermöglicht es Unternehmen und Organisationen, wertvolle Erkenntnisse aus großen und komplexen Datensätzen zu gewinnen. Die genannten Instrumente sind von entscheidender Bedeutung für die Identifizierung von Trends, die Analyse komplexer Systeme sowie die Gewinnung wichtiger Einblicke in datengesteuerte Ökosysteme [13]. Zu den zentralen Big Data-Analysetechniken zählen Data Mining, maschinelles Lernen, Verarbeitung natürlicher Sprache (NLP), Datenvisualisierung, Predictive Analytics, statistische Analyse, Clustering und Segmentierung sowie Echtzeitanalysen [13]. Die in diesem Text vorgestellten Techniken erweisen sich in Hinblick auf das Thema dieser Arbeit als besonders bedeutsam.

Im Rahmen der prädiktiven Analyse erfolgt eine Prognose kontinuierlicher Variablen sowie eine Klassifikation kategorialer Ergebnisse. Vor der Entwicklung prädiktiver Modelle ist es von entscheidender Bedeutung, deskriptive Analysetechniken anzuwenden, um die Daten zu verstehen, zu bereinigen und vorzubereiten. Zu den zahlreichen verfügbaren prädiktiven Analysemodellen zählen Techniken wie die multiple Regression, Entscheidungsbäume und neuronale Netzwerke [1]. Im Verlauf der vorliegenden Arbeit werden diese sowohl als Grundlagen in diesem Kapitel als auch als Anwendungen im nachfolgenden Kapitel behandelt.

Die folgende Abbildung 2.2.3.1 veranschaulicht die Beziehung zwischen Big Data, Big Data Analytics und Data Science und erläutert ihre Zusammenhänge.

Die Nutzung mathematischer Methoden und Algorithmen stellt einen essenziellen Bestandteil von Data Science dar, welcher im nächsten Kapitel detaillierter erörtert wird.

Big Data Analytics stellt einen Teilbereich von Big Data dar und wird sowohl in der Data Science als auch in praktischen Anwendungen eingesetzt, um spezifische Probleme zu lösen [25].



6-2.2.3.1: Zusammenhang von Big Data, Big Data Analytics, Data Science [67].S.15.

2.2.1 Künstliche Intelligenz und maschinelles Lernen

Vor der Definition des Begriffes der Künstlichen Intelligenz (KI) ist es unerlässlich, zunächst die Unterschiede zwischen einigen verwandten Themenfeldern zu klären, um Missverständnisse und Verwechslungen zu vermeiden. Die Begriffe „Künstliche Intelligenz", „Maschinelles Lernen" und „Deep Learning" werden in der öffentlichen Diskussion häufig synonym verwendet, obwohl sie sich in ihrer Bedeutung unterscheiden. In der Tat stehen sie jedoch in einem hierarchischen Verhältnis zueinander. [31]. Die Abbildung 2.2.4.1 veranschaulicht dies.

7-2.2.4.1: Leistung Bestandteile von KI [31].S.10.

❖ künstlichen Intelligenz

Der Begriff der künstlichen Intelligenz (KI) wird in unterschiedlichen Kontexten mit unterschiedlichen Schwerpunktsetzungen definiert. In der Tat besteht weitgehende Einigkeit darüber, dass es darum geht, Computerprogramme oder Maschinen zu entwickeln, die Verhaltensweisen zeigen können, die als intelligent gelten würden, wenn sie von Menschen ausgeführt würden [28]. John McCarthy, einer der Begründer dieses Fachgebiets [28], definierte künstliche Intelligenz in einem Artikel aus dem Jahr 2004 folgendermaßen: „Es handelt sich um die Wissenschaft und Technik der Entwicklung intelligenter Maschinen, insbesondere intelligenter Computerprogramme. Dies steht im Zusammenhang mit der Aufgabe, Computer zu nutzen, um menschliche Intelligenz zu verstehen. Allerdings ist KI nicht darauf beschränkt, biologisch beobachtbare Methoden anzuwenden [27]“.

In der künstlichen Intelligenz wird im Wesentlichen zwischen zwei Formen unterschieden.

- Schwache KI
- Starke KI

Der Begriff „schwach“ beschreibt in diesem Kontext Systeme mit begrenzter Zielvorgabe und eindimensionaler Funktionalität, die ihre Fähigkeiten nur innerhalb eines spezifischen Aufgabengebietes weiterentwickeln und lernen können. Diese Systeme sind auf eine spezifische Anwendung beschränkt und alle aktuell verfügbaren Anwendungsmöglichkeiten dieser Art stellen Formen schwacher Systeme dar [54].

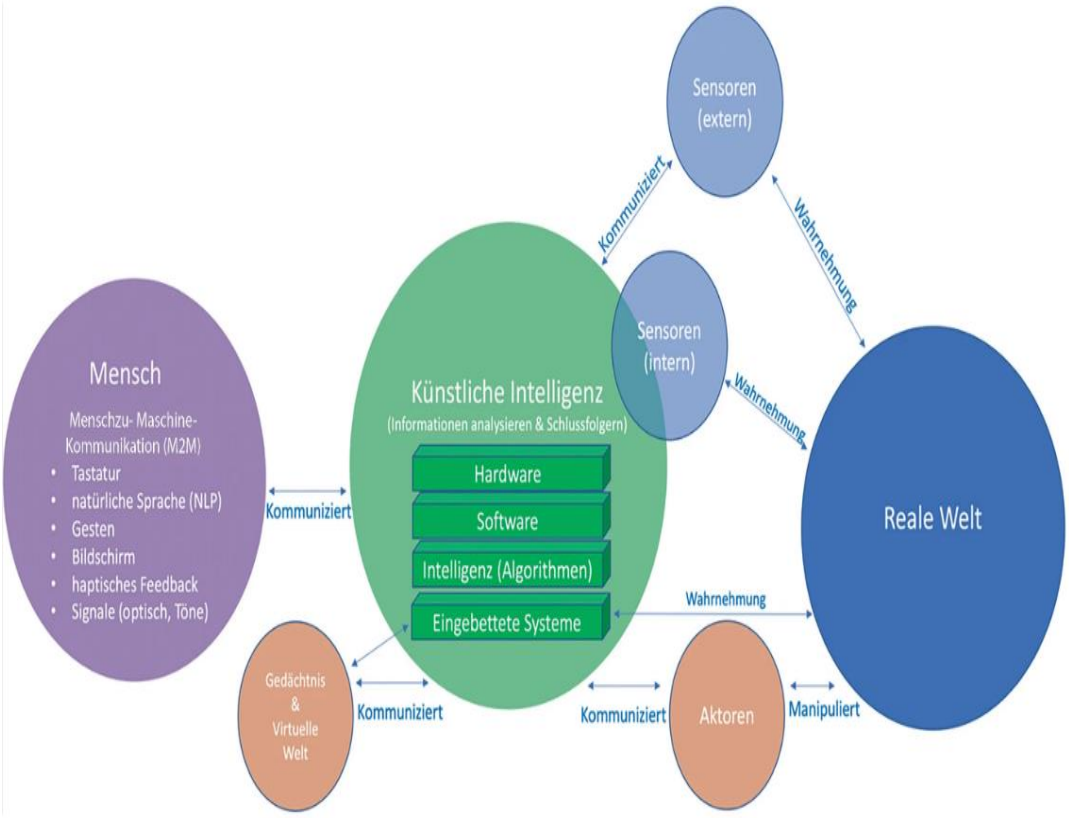
Die Theorie einer starken oder allgemeinen Intelligenz bezieht sich auf eine zukünftige Form der künstlichen Intelligenz, die menschliche Fähigkeiten wie Anpassungsfähigkeit und Problemlösung aufweist.

Eine solche Form der künstlichen Intelligenz würde ohne menschliche Anleitung auskommen und wäre folglich nicht an die Vorgaben der Entwickler gebunden.

Derzeit ist unklar, ob und wann eine solche Intelligenz entstehen könnte. Die bisherigen Szenarien sind spekulativ und fiktiv [54].

Ein KI-System besteht grundlegend aus mehreren wesentlichen Komponenten, wie in Abbildung 2.2.4.2 die Interaktionen dargestellt. Die wesentlichen Komponenten eines KI-Systems umfassen:

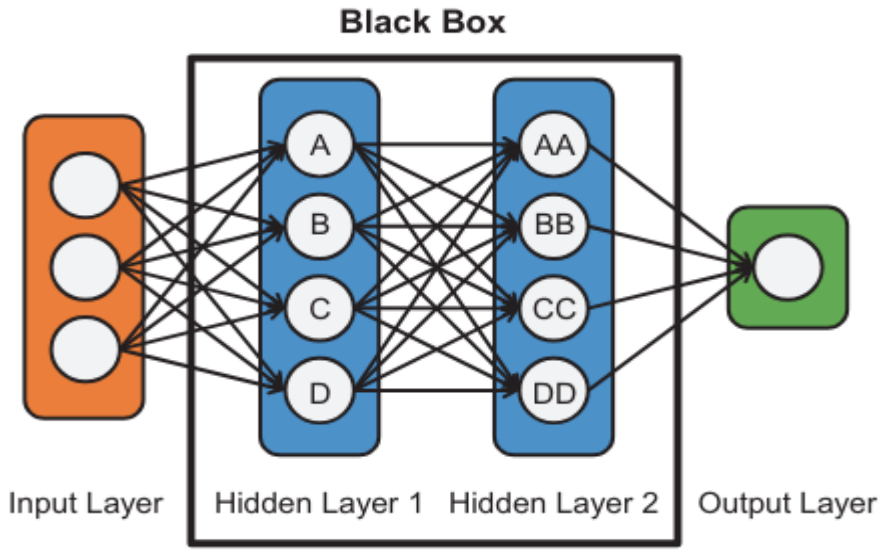
- Computersysteme (Hardware)
- Programmcode (Software)
- Algorithmen: Die Logiken zur Verarbeitung und Entscheidungsfindung
- Sensoren: Diese können extern, intern oder als eingebettete Systeme (Embedded Systems) ausgeführt sein und dienen der Erfassung von Umwelteinflüssen.
- Kommunikationstechnik: Dazu zählen Netzwerke und Mensch-Maschine-Schnittstellen, die die Interaktion mit der Umwelt ermöglichen.
- Aktoren: Diese Elemente, wie Räder, Greifer und Roboter, sind für das Handeln und die Manipulation in der realen Welt verantwortlich.
- Gedächtnis: Hierzu gehören Modelle, Simulationen, der digitale Zwilling sowie interne Datenspeicher und Wissensspeicher.
- Zugriff auf die virtuelle Welt: Dies umfasst den Zugang zum Internet, zu externen Datenquellen und zu Wissen sowie zu Virtual Reality [46].



8-2.2.4.2: Aufbau eines KI-Systems [45].S.59.

❖ Neuronale Netzwerk

Ein neuronales Netzwerk bezeichnet ein System aus Hardware und Software, dessen Aufbau an das menschliche Gehirn angelehnt ist. Ein neuronales Netzwerk besteht in der Regel aus einer großen Anzahl von Prozessoren, die parallel arbeiten und in mehreren Schichten organisiert sind. Die erste Schicht (Input Layer oder Eingabeschicht) erhält die Rohdaten und kann mit den Sehnerven in der menschlichen Sehverarbeitung verglichen werden. Die nachfolgenden Schichten (beispielsweise Hidden Layer 1 und 2) verarbeiten den Output der vorhergehenden Schicht, nicht jedoch die ursprünglich eingegebenen Daten. Die letzte Schicht (Output Layer oder Ausgabeschicht) erzeugt schließlich die endgültigen Ergebnisse des Systems [32].Die Abbildung 2.2.4.3 veranschaulicht verschiedene Schichten bei neuronalen Netzwerken.



9-2.2.4.3: Verschiedene Schichten bei neuronalen Netzwerken [31].S.11.

❖ Maschinelles Lernen:

Im Folgenden wird eine ausführliche Erläuterung des Themenbereichs Maschinelles Lernen gegeben. An dieser Stelle wird eine vorläufige Definition bereitgestellt, um die Unterschiede zwischen den Begriffen „Künstliche Intelligenz (KI)“, „Maschinelles Lernen“ und „Deep Learning“ zu verdeutlichen. Maschinelles Lernen hat historische Wurzeln in der Statistik und der künstlichen Intelligenz (KI). Aus statistischer Sicht konzentriert es sich auf die Entdeckung von Mustern und Zusammenhängen in Daten sowie auf die

Erstellung von Modellen zur Datenerzeugung und Vorhersage. Diese Modelle verbessern das Verständnis von Zusammenhängen und liefern wichtige Erkenntnisse in verschiedenen Domänen. Aus der Perspektive der künstlichen Intelligenz betrachtet, stellt maschinelles Lernen die Grundlage für die Entwicklung intelligenter, lernfähiger Systeme dar, die aus Erfahrungen lernen und ihre Leistung bei der Lösung von Aufgaben kontinuierlich verbessern können. Dabei dienen menschliche Lernprozesse als Vorbild [8].

- ❖ Deep Learning:

Die Fähigkeit komplexer Computermodelle, Daten auf mehreren Abstraktionsebenen zu verarbeiten, wird durch Deep Learning ermöglicht. Im Bereich des maschinellen Lernens werden diese Modelle genutzt, um Objekte in Bildern zu erkennen, gesprochene Sprache in Text umzuwandeln und Inhalte an die Nutzerinteressen anzupassen. Diese Anwendungen greifen verstärkt auf Deep Learning-Techniken zurück, die eine Weiterentwicklung traditioneller Methoden darstellen. In der Vergangenheit waren Ansätze des maschinellen Lernens jedoch begrenzt in ihrer Fähigkeit, natürliche Daten in ihrer ursprünglichen Form zu verarbeiten [37]. Der entscheidende Unterschied zwischen traditionellem maschinellem Lernen (ML) und modernem Deep Learning besteht darin, dass tiefe neuronale Netzwerke im Verlauf des Trainingsprozesses eigenständig Darstellungen (sogenannte „Features“) der Daten erlernen, anstatt diese von Experten manuell erstellen zu lassen, wie es vor der Ära des Deep Learnings die gängige Praxis war [74].

Die vorliegende Arbeit fokussierte bislang die Industrie 4.0 sowie die zugehörigen Technologien und deren Anwendungsbereiche. In den nachfolgenden Abschnitten wird der Schwerpunkt auf den Bereich der Instandhaltung und den aktuellen Stand der Technik in diesem Bereich gelegt. Des Weiteren wird detailliert das Thema Maschinelles Lernen behandelt und dessen Anwendung in der Instandhaltung als prädiktive Instandhaltung erörtert, welches den Hauptteil dieser Arbeit darstellt.

2.3 Instandhaltung

Die Instandhaltung nimmt eine zentrale Rolle in der industriellen Praxis ein und hat maßgeblichen Einfluss auf die Kostenstruktur sowie die Zuverlässigkeit von Produktionsprozessen. Ihre Bedeutung für die Wettbewerbsfähigkeit eines Unternehmens manifestiert sich insbesondere in den Bereichen Preisgestaltung, Qualität und Leistungsfähigkeit. Unerwartete Stillstände von Maschinen und Anlagen können das operative Geschäft eines Unternehmens erheblich beeinträchtigen. Folglich ist die Entwicklung und Implementierung einer effektiven Instandhaltungsstrategie von essentieller Bedeutung. Die Zielsetzung dieser Strategie liegt in der Vermeidung ungeplanter Ausfälle, der Erhöhung der Zuverlässigkeit von Betriebsabläufen und der Verringerung der Betriebskosten [76]. Somit erlangt die Relevanz dieser Arbeit eine deutliche Gewichtung.

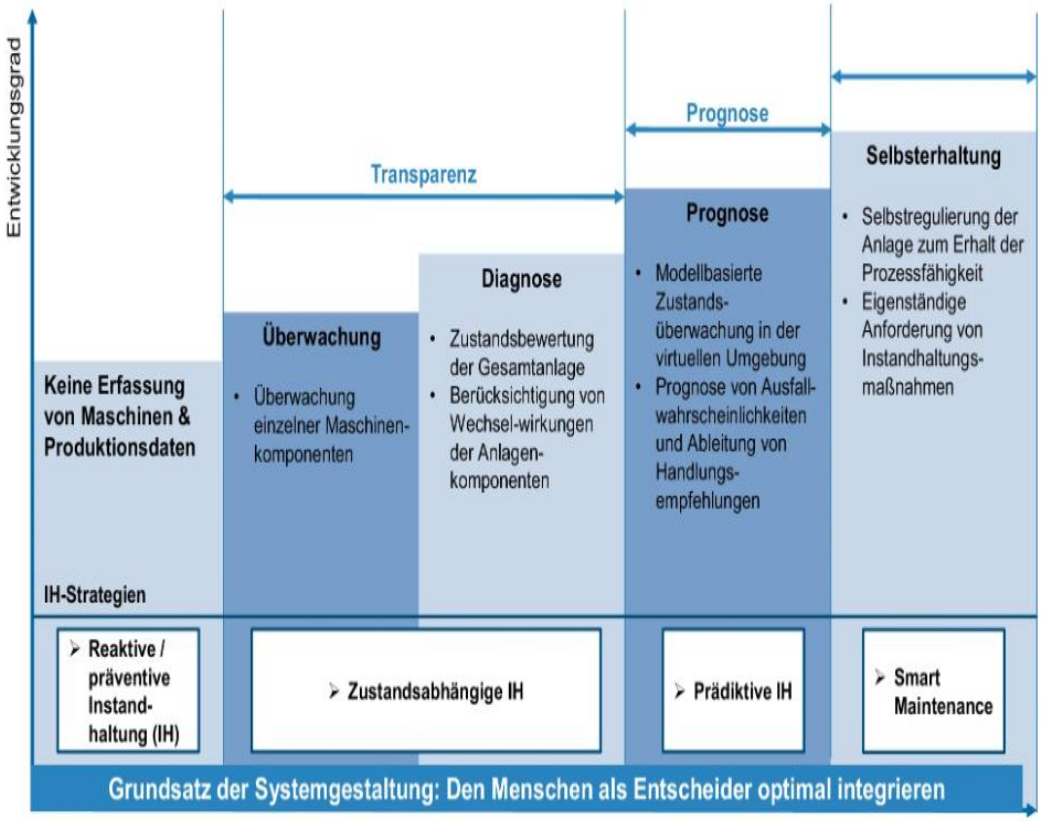
Die zunehmende technische Komplexität von Maschinen und Anlagen hat in den letzten Jahren die Bedeutung der Instandhaltung als zentralen Faktor zur Sicherstellung der Verfügbarkeit von Maschinen und Anlagen erhöht [41]. Laut DIN 31051 wird die Instandhaltung als "Kombination aller technischen und administrativen Maßnahmen sowie Maßnahmen des Managements während des Lebenszyklus einer Einheit, die dem Erhalt oder der Wiederherstellung ihres funktionsfähigen Zustands dient, sodass sie die geforderte Funktion erfüllen kann" [14] definiert. Abbildung 2.3.1 zeigt die Einteilung der grundlegenden Instandhaltungsmaßnahmen nach DIN 31051 und erläutert deren Definitionen und Ziele.

10-2.3.1: Begriff der Instandhaltung nach DIN 31051 [70].S.3.

2.3.1 Stand der Technik in der Instandhaltung

Die Abbildung 2.3.1.1 veranschaulicht den aktuellen Stand der Technik in der Instandhaltung durch ein dargestelltes Referenzmodell und verdeutlicht die Integration von Menschen, Technologie und Organisation zu einem komplexen Gesamtsystem der Wertschöpfung. Im weiteren Verlauf erfolgt eine vergleichende Betrachtung der unterschiedlichen Instandhaltungsstrategien. Ein Thema, das nicht im Rahmen dieser Arbeit behandelt wird und nicht zum Scope dieser Arbeit passt, ist die Selbsterhaltung.

Kurz definiert: Die Selbsterhaltung bezeichnet einen Prozess, bei dem Anlagen durch Selbstregulierung die Erhaltung der Prozessfähigkeit gewährleisten. Das Ziel der Selbsterhaltung besteht in der Entwicklung von Anlagen, die durch Selbstregulierung die Erhaltung der Prozessfähigkeit gewährleisten. In diesem Zusammenhang erfolgt eine kontinuierliche Anpassung der Instandhaltungsstrategie, wobei sowohl die aktuellen als auch die zu erwartenden Belastungen berücksichtigt werden. Sobald ein erster Ausfall droht, werden Maßnahmen zur Sicherung der Prozessfähigkeit automatisch eingeleitet [50].

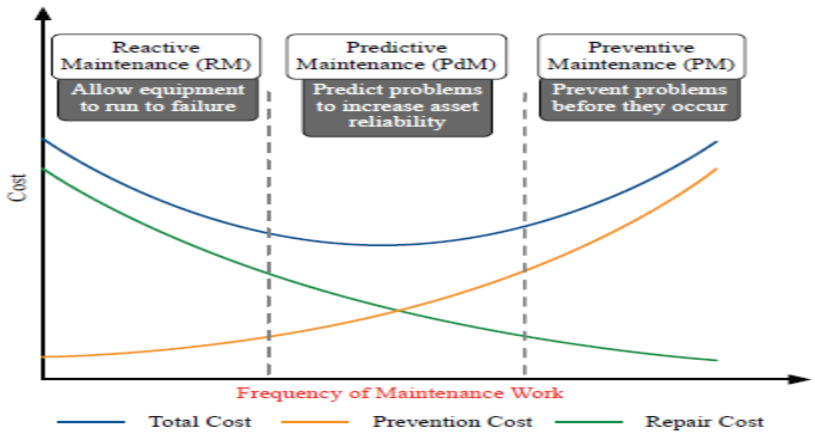


11-2.3.1.1: Referenzmodell [49].S.353.

2.3.1 Unterschiede zwischen RM, PM, PdM

Die reaktive Instandhaltung (Reactive Maintenance, RM) erfolgt nach dem Eintritt eines Ausfalls mit dem Ziel, den Betriebszustand der betroffenen Ausrüstung wiederherzustellen. Dieses Vorgehen kann zu signifikanten Verzögerungen und hohen Reparaturkosten führen. Demgegenüber wird die präventive Instandhaltung (Preventive Maintenance, PM) nach einem festgelegten Zeitplan durchgeführt, der auf Zeit- oder Prozessintervallen basiert. Das Ziel präventiver Instandhaltung ist die Vermeidung von Ausfällen, wobei jedoch das Risiko besteht, dass unnötige Maßnahmen durchgeführt werden, was zu hohen Präventionskosten führen kann [76].

Um einen optimalen Kompromiss zwischen RM und PM zu erreichen, wird die prädiktive Instandhaltung (Predictive Maintenance, PdM) implementiert. Die prädiktive Instandhaltung basiert auf der kontinuierlichen Überwachung und Online-Schätzung des „Gesundheitszustandes" der Ausrüstung. Auf diese Weise können rechtzeitig Eingriffe vor einem potenziellen Ausfall vorgenommen werden. Dies führt zu einer Reduzierung der Häufigkeit von Instandhaltungsmaßnahmen, der Vermeidung ungeplanter Instandhaltungsmaßnahmen und zu einer Reduzierung der mit einer übermäßigen Planung verbundenen Kosten [76]. In diesem Kontext erscheint es sinnvoll, die Abbildung 2.3.2.1 zur Veranschaulichung eines Vergleichs von RM, PM und PdM hinsichtlich der Kosten und Häufigkeit der Instandhaltungsarbeiten zu nutzen.



12-2.3.2.1: Vergleich von RM, PM und PdM hinsichtlich der Kosten und Häufigkeit der Instandhaltungsarbeiten [72].S.5.

Die vorliegende Tabelle, in Anlehnung an [67, Seite 2], präsentiert in tabellarischer Form die Vorzüge, Herausforderungen sowie geeignete und ungeeignete Anwendungsbereiche von reaktiver Instandhaltung (RM), präventiver Instandhaltung (PM) und prädiktiver Instandhaltung (PdM).

	Vorteile	Herausforderungen	Geeignete Anwendungen	Ungeeignete Anwendungen
RM	• Maximale Nutzung und Produktionswert • Geringere Präventionskosten	• Ungeplante Ausfallzeiten • Hohe Ersatzteilbestandskosten • Potenziell weitere Schäden an den Anlagen • Höhere Reparaturkosten	• Redundante oder nicht-kritische Ausrüstung • Reparatur von kostengünstigen Anlagen nach Ausfall	• Ausfälle der Ausrüstung stellen ein Sicherheitsrisiko dar • 24/7 Verfügbarkeit der Ausrüstung ist notwendig
PM	• Geringere Reparaturkosten • Weniger Ausfälle und ungeplante Stillstände der Ausrüstung	• Bedarf an Inventar • Erhöhte geplante Stillstands Zeiten • Wartung von scheinbar einwandfreier Ausrüstung	• Anlagen, deren Ausfallwahrscheinlichkeit mit der Zeit oder Nutzung zunimmt	• Zufällige Ausfälle, die nicht mit der Wartung in Zusammenhang stehen
PdM	• Ganzheitlicher Überblick über den Zustand der Ausrüstung • Verbesserte Analysemöglichkeiten • Vermeidung von Ausfällen im Betrieb • Vermeidung des Austauschs von Komponenten mit verbleibender Nutzungsdauer	• Höhere anfängliche Infrastrukturkosten und Einrichtung (z.B. Sensoren) • Komplexeres System		

2.3.2 Prädiktive Instandhaltung

Das Prinzip der prädiktiven Instandhaltung basiert auf proaktiven Eingriffen und Reparaturen an Anlagen, bei denen in naher Zukunft ein Ausfall oder ein Absinken der Produktqualität erwartet wird. Diese Maßnahmen werden in Zeiten geplanter Produktionspausen durchgeführt, wodurch Produktionsausfälle aufgrund von Maschinenstörungen während der laufenden Produktion vermieden werden. Des Weiteren wird durch rechtzeitige Eingriffe verhindert, dass bei Störungen häufig entstehender Produktausschuss oder Schäden an Produktionseinrichtungen auftreten. Im Gegensatz zu klassischen, geplanten Instandhaltungsmaßnahmen, die nach einer bestimmten Einsatzdauer oder Laufleistung erfolgen, basiert die prädiktive Instandhaltung auf dem festgestellten Maschinenzustand oder der Prognose einer unmittelbar bevorstehenden Störung [44]. Prädiktive Instandhaltung zielt darauf ab, durch die Anwendung von Verschleißmodellen und die Bestimmung der Restlebensdauer, Instandhaltungsobjekte optimal bis zu ihrem Lebensende zu nutzen und anschließend planmäßig auszutauschen.

Die zustandsorientierte Instandhaltung, die auf einer Verschleiß- oder Lebensdauervorhersage basiert (EN 13306), ermöglicht eine vorausschauender Planung im Vergleich zur herkömmlichen diagnostikbasierten Strategie [40].

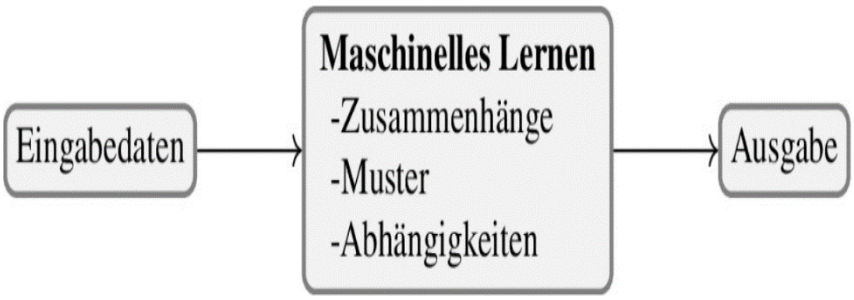
Die Grundlage dieses Ansatzes bildet die Erfassung einer Vielzahl von Maschinendaten sowie deren Analyse unter Zuhilfenahme von Methoden des maschinellen Lernens. Ziel ist die Modellierung komplexer Zusammenhänge zwischen Zustandsdaten und entsprechenden Zielvariablen, beispielsweise der Restlaufzeit. Die bloße Verfügbarkeit zahlreicher Überwachungsdaten garantiert jedoch nicht zwangsläufig eine solide Informationsgrundlage für die Entwicklung effektiver Prognosemodelle [7].

Im Folgenden wird die Anwendung von maschinellem Lernen erörtert, um zu untersuchen, auf welche Weise und mit welchen Methoden die erfassten Maschinendaten die Entwicklung prädiktiver Modelle für die prädiktive Instandhaltung mittels maschineller Lernalgorithmen unterstützen können.

2.4 Verfahren des Maschinellen Lernens

Wie bereits im Abschnitt über Künstliche Intelligenz und maschinelles Lernen dargelegt, stellt das maschinelle Lernen einen Teilbereich der KI dar, der darauf abzielt, Computer dazu zu befähigen, eigenständig Probleme zu lösen, ohne dass für jedes spezifische Problem eine explizite Programmierung notwendig ist. Auch wenn häufig die Verfahren, die es ermöglichen, aus Daten zu lernen, im Mittelpunkt stehen, umfasst das Themengebiet deutlich mehr als nur diese [36]. Das maschinelle Lernen befasst sich mit der Entwicklung lernfähiger Systeme und Algorithmen. Die Algorithmen identifizieren anhand umfangreicher Daten verschiedene Zusammenhänge. Das resultierende Modell kann anschließend auf neue, unbekannte Daten derselben Art angewendet werden [9].

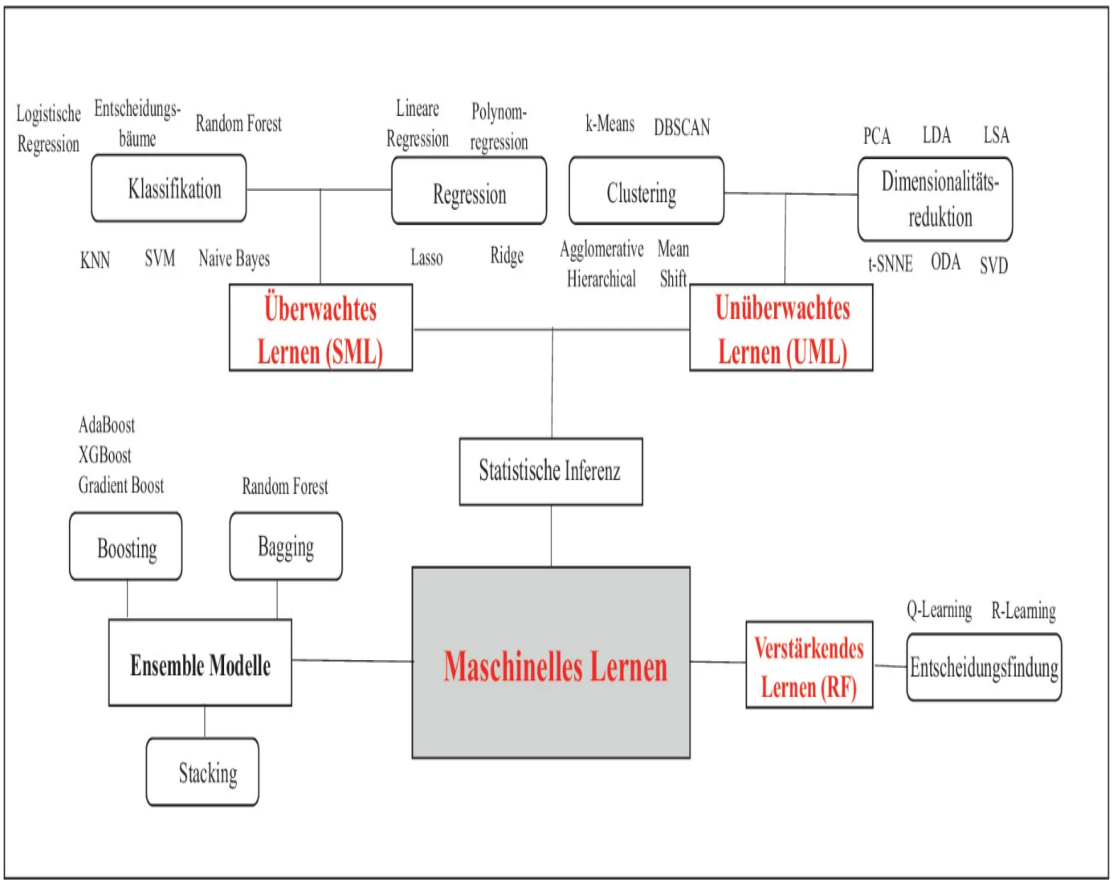
Wie in Abbildung 2.4.1, welche den schematischen Ablauf des maschinellen Lernprozesses veranschaulicht, dargestellt, werden für das Training eines Lernalgorithmus Eingabedaten benötigt. In der Regel handelt es sich dabei um wert- oder zeitdiskrete Daten. Die Daten dienen dem Algorithmus zur Erlernung einer mathematischen Funktion. Funktionen oder Abbildungen stellen eindeutige Zuordnungen zwischen zwei Mengen, X und Y, dar [9].



13-2.4.1: Abbildung 2.4.1 Schematische Darstellung des ML-Prozesses [9].S.2.

Die Abbildung 2.4.2 präsentiert zudem eine Übersicht über die verschiedenen Methoden des maschinellen Lernens. Die umfassende Darstellung veranschaulicht die Vielfalt und Komplexität der Ansätze im Bereich des maschinellen Lernens sowie deren Anwendungsmöglichkeiten. Die Methoden lassen sich in die Kategorien überwachte Lernverfahren (Supervised Machine Learning, SML), unüberwachte Lernverfahren (Unsupervised Machine Learning, UML) und verstärkendes Lernen (Reinforcement Learning, RF) unterteilen. Zu den überwachten Lernverfahren zählen unter anderem

Klassifikations- und Regressionsmethoden, während die unüberwachten Lernverfahren Techniken wie Clustering und Dimensionsreduktion umfassen. Im weiteren Verlauf der Arbeit erfolgt eine separate Behandlung einiger dieser Methoden. Verstärkendes Lernen fokussiert sich auf Entscheidungsfindungsprozesse, wobei häufig Q-Learning und R-Learning zum Einsatz kommen. Zusätzlich sind Ensemble-Modelle zu nennen, welche Methoden wie Boosting, Bagging und Stacking kombinieren, um die Vorhersagegenauigkeit zu verbessern und bessere Ergebnisse zu erhalten [3].



14-2.4.2: Modelle des maschinellen Lernens [3].S.6.

Im Rahmen der vorliegenden Untersuchung soll zunächst der Unterschied zwischen überwachten und unüberwachten Lernverfahren erörtert werden, um ein grundlegendes Verständnis für die anschließende Diskussion zu entwickeln. Als wesentliches Merkmal ist hierbei festzuhalten, dass die Ausgabedaten beim überwachten Lernen bekannt sind, während dies beim unüberwachten Lernen nicht der Fall ist [45].

2.4.1 Überwachtes Lernen (SML)

Das überwachte Lernen stellt das erste Lernkonzept dar, das bei der Entwicklung neuronaler Algorithmen eine signifikante Rolle spielte. Das Ziel besteht in der Erlernung einer Menge von Ein- und Ausgabedaten auf geeignete Weise [63]. Anders ausgedrückt zielt überwachtes Lernen darauf ab, Eingangsdaten x auf Ausgangsdaten y abzubilden, sodass sie eine bekannte Zielgröße d wiedergeben. Die Zielgröße wird in diesem Kontext häufig als Label bezeichnet, insbesondere im Kontext von Klassifikationsproblemen [52]. Ein Modell erlernt die überwachte Mustererkennung, wenn sowohl die Eingabedaten (unabhängige Variablen, Merkmale, Prädiktoren) als auch die Zielvariablen bereitgestellt werden, sodass Referenzwerte für korrekte Entscheidungen existieren [29]. Im Rahmen des überwachten Lernens erfolgt eine Aufteilung der Daten in Trainings- und Testdaten [64].

Anhand der vorliegenden Informationen aus der Abbildung 2.4.2 lässt sich eine Differenzierung der überwachten Verfahren in zwei Untergruppen vornehmen:

- Die Klassifikation bei diskreten Ausgabedaten
- Die Regression bei kontinuierlichen Ausgabedaten

stellen zwei Untergruppen des überwachten Lernens dar [45].

❖ Klassifikationsmodells

Das Ziel besteht in der Anwendung eines Klassifikators auf neue Daten, der sich entsprechend dem Verhalten verhält, das ihm anhand der Trainingsdaten beigebracht wurde. Im Rahmen dessen ist es erforderlich, dass für Datenpunkte konsistente Zuweisungen gemäß den in den Trainingsdaten gegebenen Klassenzuweisungen erfolgen. Die Auswahl der Beispiele muss so erfolgen, dass der Klassifikator in der Lage ist, eine Grenze zwischen den Klassen zu ziehen. Ein E-Mail-Spamfilter beispielsweise benötigt für das Training sowohl Beispiele für Spam als auch Beispiele für Nicht-Spam [6].

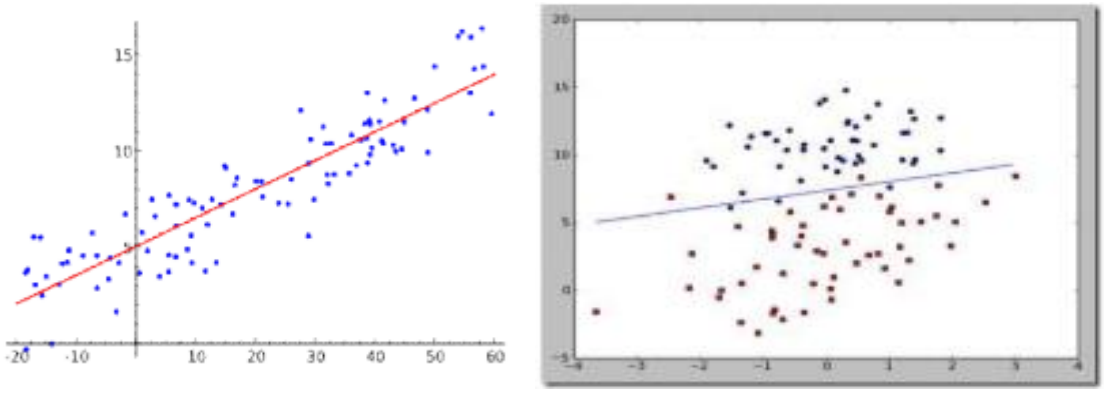
In der technischen Praxis werden Klassifikationen bei verschiedenen Problemstellungen mit diskreten Ausgabewerten angewandt. Zu den Anwendungsgebieten zählen beispielsweise:

- Die Bestimmung des aktuellen Betriebszustandes ist von zentraler Bedeutung.
- Die Vorhersage eines bevorstehenden Ereignisses.
- Die Festlegung einer adäquaten Reaktion [45].

❖ Regressionsmodells

Die lineare Regression stellt eine zentrale Methode des überwachten Lernens dar. Die lineare Regression findet in zahlreichen Anwendungsbereichen Anwendung, beispielsweise zur Abschätzung von Versicherungs- oder Kreditrisiken, in der personalisierten Medizin sowie in der Marktanalyse. Im Rahmen einer Regressionsaufgabe erfolgt die Vorhersage einer numerischen Zielvariable mithilfe mehrerer Prädiktorvariablen. Dies erfolgt durch das Erlernen eines Modells, dessen Ziel die Minimierung einer Verlustfunktion ist [25].

In Abbildung 2.4.1.1 wird die logistische Regression dargestellt, welche die Unterschiede zwischen der linearen und der logistischen Regression verdeutlicht. Es wird ein Beispiel für eine binäre Klassifizierung präsentiert, bei der die Ausgabe der Funktion entweder den Wert 0 oder 1 annimmt. Dabei steht der Wert 0 für die Zugehörigkeit zu einer der beiden Klassen, während der Wert 1 die Gegenklasse bezeichnet [77].



Lineare Regression logistische Regression

15-2.4.1.1: Unterschied zwischen linearer und logistischer Regression [73].S1,2.

Die Kernaufgabe des Regressionsproblems besteht in der Entwicklung eines Lernalgorithmus, der auf Basis vorhandener Daten trainiert wird und die Eingaben den entsprechenden Ausgabeergebnissen zuordnet. Dies dient dem Ziel der Vorhersage [22]. Innerhalb des technischen Umfelds werden Regressionen bei der Lösung diverser Problemstellungen mit kontinuierlichen Ausgabewerten eingesetzt. Zu den Anwendungsgebieten von Regressionen zählt beispielsweise die:

- Die Ermittlung eines Wertes ohne den Einsatz von Messungen oder Simulationen.

- Die Vorhersage der zeitlichen Entwicklung eines Wertes.
- Des Weiteren kann die automatische Optimierung von Prozessparametern durchgeführt werden [45].

2.4.2 Unüberwachtes Lernen (UML)

Im Rahmen des unüberwachten Lernens erfolgt die Identifikation von Strukturen in den Daten. Im Gegensatz zum überwachten Lernen stehen dabei keine Labels oder Zielgrößen zur Verfügung. Eine wesentliche Stärke des überwachten Lernens ist die Erstellung von Modellen für Variablen. Die Methode basiert auf der Bereitstellung von Eingangsdaten sowie gewünschten Ausgangswerten (Input und Labels) und ermöglicht somit das Erlernen des Zusammenhangs zwischen beiden [53]. Die beiden zentralen Aufgaben des unüberwachten Lernens sind das Erkennen von Mustern sowie das Nachbilden von Zusammenhängen [21].

Die Mustererkennung erfolgt mittels Deep-Learning-Algorithmen [71].

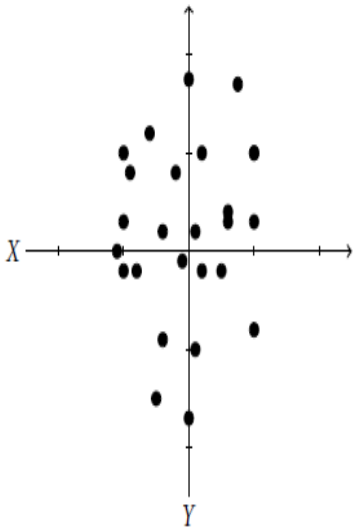
Anhand der vorliegenden Informationen aus der Abbildung 2.4.2 lässt sich eine Differenzierung der überwachten Verfahren in zwei Untergruppen vornehmen:

- Clustering
- Dreidimensional

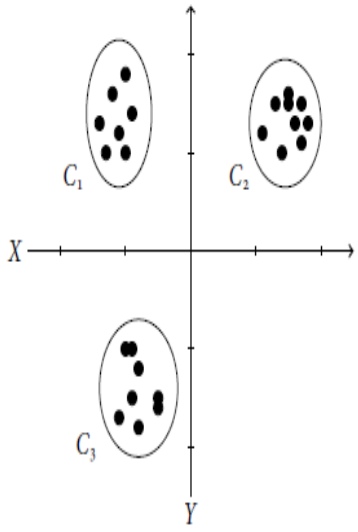
stellen zwei Untergruppen des überwachten Lernens dar.

❖ Clustering

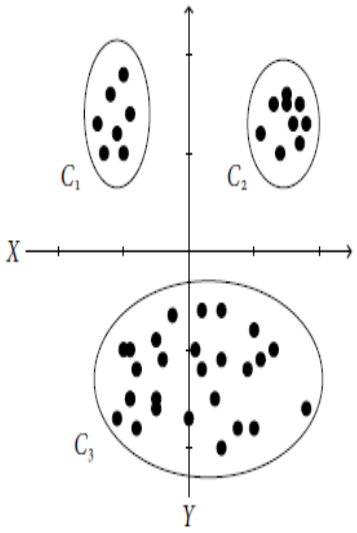
Das primäre Ziel clusteranalytischer Auswertungsverfahren besteht in der Unterteilung einer Menge von Klassifikationsobjekten in homogene Gruppen (Klassen, Cluster, Typen). Kurz gesagt, geht es darum, eine empirische Klassifikation (Gruppeneinteilung, Typologie) zu finden [5]. In den Abbildungen 2.4.2.1 (a) bis (d) werden verschiedene Datenkonstellationen präsentiert, um die Erkennbarkeit von Clusterstrukturen zu veranschaulichen. Diese Abbildungen 4.2.2.1 veranschaulichen das Prinzip der Clusterbildung, insbesondere die Relevanz der Homogenität als zentrales Prinzip bei der Bildung von Clustern der Gruppen.



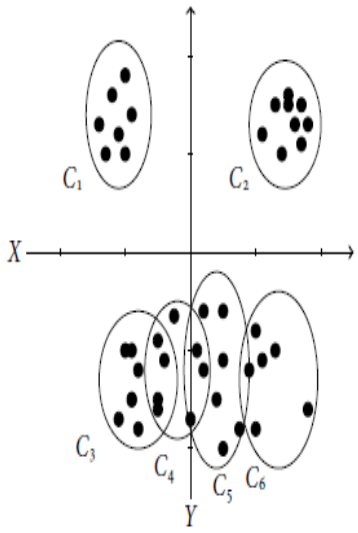
(a) Eine Clusterstruktur ist nicht erkennbar. Die Klassifikationsobjekte bilden eine große Punktewolke.



(b) Es sind drei Cluster erkennbar. Die Cluster sind homogen und voneinander gut getrennt.



(c) Es lassen sich drei Cluster erkennen. Cluster C_3 ist aber sehr lang gestreckt und wenig homogen.



(d) Im Unterschied zur Abbildung 1.1c wurde das lang gestreckte Cluster in vier überlappende Teilcluster zerlegt.

16-4.2.2.1: Datenkonstellationen mit erkennbarer und nicht erkennbarer Clusterstruktur [5].S.17.

- Dreidimensional

Die Reduktion der Datendimensionalität bezeichnet die Umstrukturierung der ursprünglichen Informationen in eine geringere Anzahl von Dimensionen. Die Umstrukturierung erfolgt in einer teilenden Weise, d. h. von oben nach unten (Top-down). Das Ziel besteht in der Verringerung der Anzahl unabhängiger Variablen bei gleichzeitiger Vermeidung eines erheblichen Informationsverlustes. Die Methoden zur Dimensionsreduktion komprimieren die Daten somit von einem hochdimensionalen in einen niedrigdimensionalen Raum [3].

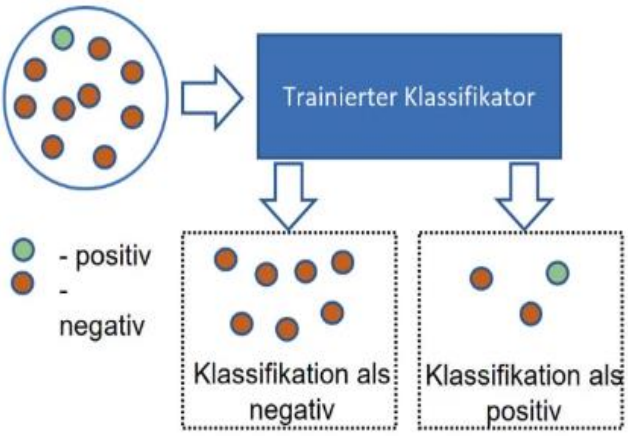
2.4.3 Verstärkendes Lernen („Reinforcement Learning“)

Diese Lernmethode fokussiert sich auf Agenten in spezifischen Situationen, die für spezifische Aktionen eine Belohnung erhalten. Diese kann in Form eines Rewards dargestellt werden, der durch verschiedene Eingabewerte und Ausgaben definiert wird. Die externe Bewertung der Ausgabe stellt dabei ein weiteres entscheidendes Element dar. Die zukünftigen Aktionen sollen so angepasst werden, dass die erhaltene Belohnung maximiert wird. Ziel ist es, durch diesen Lernprozess eine Steuerung zu entwickeln, die sich automatisch optimiert [39]. Ein Beispiel hierfür ist das System für autonomes Fahren in einem Fahrzeug, welches für Fahrmanöver, die zu einer Kollision führen, bestraft wird, während Manöver, die Unfälle vermeiden oder der Straßenverkehrsordnung entsprechen, belohnt werden. Der beschriebene Prozess führt zu einer Optimierung der Entscheidungsfindung des Systems [12].

2.5 Konfusionsmatrix

Zur Evaluierung der Leistungsfähigkeit eines Klassifikationsmodells findet in der Praxis häufig die Verwendung einer Konfusionsmatrix Anwendung. Die Konfusionsmatrix stellt die Anzahl der Fälle dar, in denen das Modell eine bestimmte Klasse korrekt oder inkorrekt vorhergesagt hat. Dabei wird in jeder Zelle die entsprechende Anzahl an Fällen angezeigt. Die Summe aller Zellen entspricht der Gesamtheit der Beobachtungen. Die Konfusionsmatrix stellt eine Grundlage zur Berechnung verschiedener Metriken dar, welche zur Leistungsbewertung von Modellen herangezogen werden können [15]. Die in Abbildung 2.5.1 dargestellte

Konfusionsmatrix veranschaulicht ein Beispiel zur Evaluierung von Klassifikationsmodellen und beinhaltet die Berechnung einiger wesentlicher Metriken.



Konfusionsmatrix		Real	
		positiv	negativ
Vorher-sage	positiv	1 wahr-positiv (TP)	2 falsch-negativ (FP)
	negativ	0 falsch-positiv (FN)	7 wahr-negativ (TN)

$$Genauigkeit = \frac{TP + TN}{TP + TN + FP + FN}$$

$$falsch{-}postiv\ Rate = \frac{FP}{FP + TN}$$

$$falsch{-}negativ\ Rate = \frac{FN}{FN + TP}$$

$$Sensitivität = \frac{TP}{TP + FN}$$

$$Spezifität = \frac{TN}{TN + TP}$$

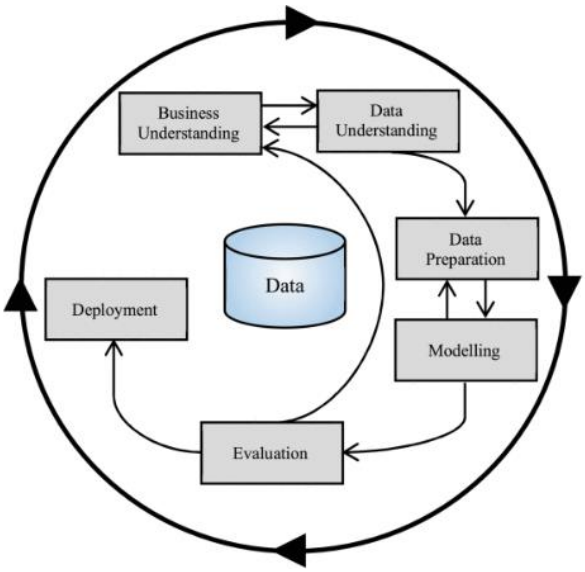
17-2.5.1: Beispiel einer Konfusionsmatrix [15].S.17.

3 Anwendung

3.1 Vorgehensweise

Im vorangegangenen Kapitel wurde eine umfassende Übersicht über Industrie 4.0, die Technologien der Industrie 4.0, Instandhaltungsstrategien sowie Verfahren des maschinellen Lernens im Hinblick auf Zuverlässigkeitstechnik in der Industrie 4.0 gegeben. Auf Basis der zuvor dargelegten Inhalte wird im Folgenden die Entwicklung eines prädiktiven Modells unter Zuhilfenahme maschineller Lernalgorithmen erörtert. Dies erfolgt anhand eines Anwendungsbeispiels.

W. Edwards Demings Ausspruch „In God we trust; all others must bring data" [26], [58] betont die zentrale Rolle von Daten in der Datenwissenschaft und bildet das Kernelement des CRISP-DM-Modells Abbildung 3.1.1. Im Folgenden wird das CRISP-DM-Modell als methodischer Rahmen dieser Arbeit angewendet. Im Folgenden werden der Datensatz sowie das Datenverständnis analysiert. Im Anschluss erfolgt die Datenaufbereitung, wobei verschiedene Algorithmen Modelle vorgestellt werden. In der Folge wird ein Vergleich der Genauigkeit der angewendeten Algorithmen vorgenommen.



18-3.1.1 Crisp Modell [2].S.2.

3.2 Datensatz und Data Understanding

Das vorliegende Kapitel präsentiert ein Anwendungsbeispiel, das auf einem Datensatz basiert, der auf der Plattform Kaggle zur Verfügung gestellt ist. Kaggle fungiert als Webportal, das der Datenwissenschafts-Community umfangreiche Werkzeuge und Ressourcen bereitstellt, um die Weiterentwicklung in diesem Bereich zu fördern [59]. Der analysierte Datensatz konzentriert sich auf den Zustand von Automotoren und wird im Kontext der vorausschauenden Wartung untersucht. Das vorliegende Anwendungsbeispiel veranschaulicht das Potenzial prädiktiver Wartungsmodelle in der Automobilindustrie. Diese ermöglichen datengetriebene Entscheidungen, welche Ausfälle vermeiden und die Leistungsfähigkeit der Fahrzeuge verbessern können.

Ein mögliches Projekt, das auf diesem Datensatz aufbaut, ist die Entwicklung eines prädiktiven Wartungsmodells für Automotoren. Die Analyse von Mustern und Trends in den Daten ermöglicht die Entwicklung maschineller Lernalgorithmen, die Vorhersagen über den Wartungs- und Reparaturbedarf eines Motors treffen können. Dadurch könnten Fahrzeugbesitzer und Mechaniker potenzielle Probleme frühzeitig erkennen und angehen, bevor sie sich verschlimmern. Dies würde nicht nur eine Optimierung der Fahrzeugleistung, sondern auch eine Verlängerung der Lebensdauer der Motoren bewirken [47].

Die Abbildung 3.2.1 präsentiert einen Ausschnitt des Datensatzes über den Zustand von Kraftfahrzeugen, welcher insgesamt 19.535 Datenpunkte und sieben Merkmale umfasst. Die Merkmale umfassen:

- Motordrehzahl (Engine rpm)
- Schmieröldruck (Lub oil pressure)
- Kraftstoffdruck (Fuel pressure)
- Kühlmitteldruck (Coolant pressure)
- Schmieröltemperatur (Lub oil temp)
- Kühlmitteltemperatur (Coolant temp)
- Motorzustand (Engine Condition).

Der Motorzustand stellt dabei die Zielvariable dar. Die Auswertung der genannten sechs Merkmale erlaubt schließlich eine Aussage über den Zustand des Motors. Die Werte 1 und 0 wurden verwendet, um den Zustand des Motors zu beschreiben. Der Wert 1 zeigt an, dass der Motor gesund ist, während der Wert 0 darauf hinweist, dass der Motor außer Betrieb oder ausgefallen ist.

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition
0	700	2.493592	11.790927	3.178981	84.144163	81.632187	1
1	876	2.941606	16.193866	2.464504	77.640934	82.445724	0
2	520	2.961746	6.553147	1.064347	77.752266	79.645777	1
3	473	3.707835	19.510172	3.727455	74.129907	71.774629	1
4	619	5.672919	15.738871	2.052251	78.396989	87.000225	0
5	1221	3.989226	6.679231	2.214250	76.401152	75.669818	0
6	716	3.568896	5.312266	2.461067	83.646589	79.792411	1
7	729	3.845166	10.191126	2.362998	77.921202	71.671761	1
8	845	4.877239	3.638269	3.525604	76.301626	70.496024	0
9	824	3.741228	7.626214	1.301032	77.066520	85.143297	0

19-3.2.1: Datentabelle

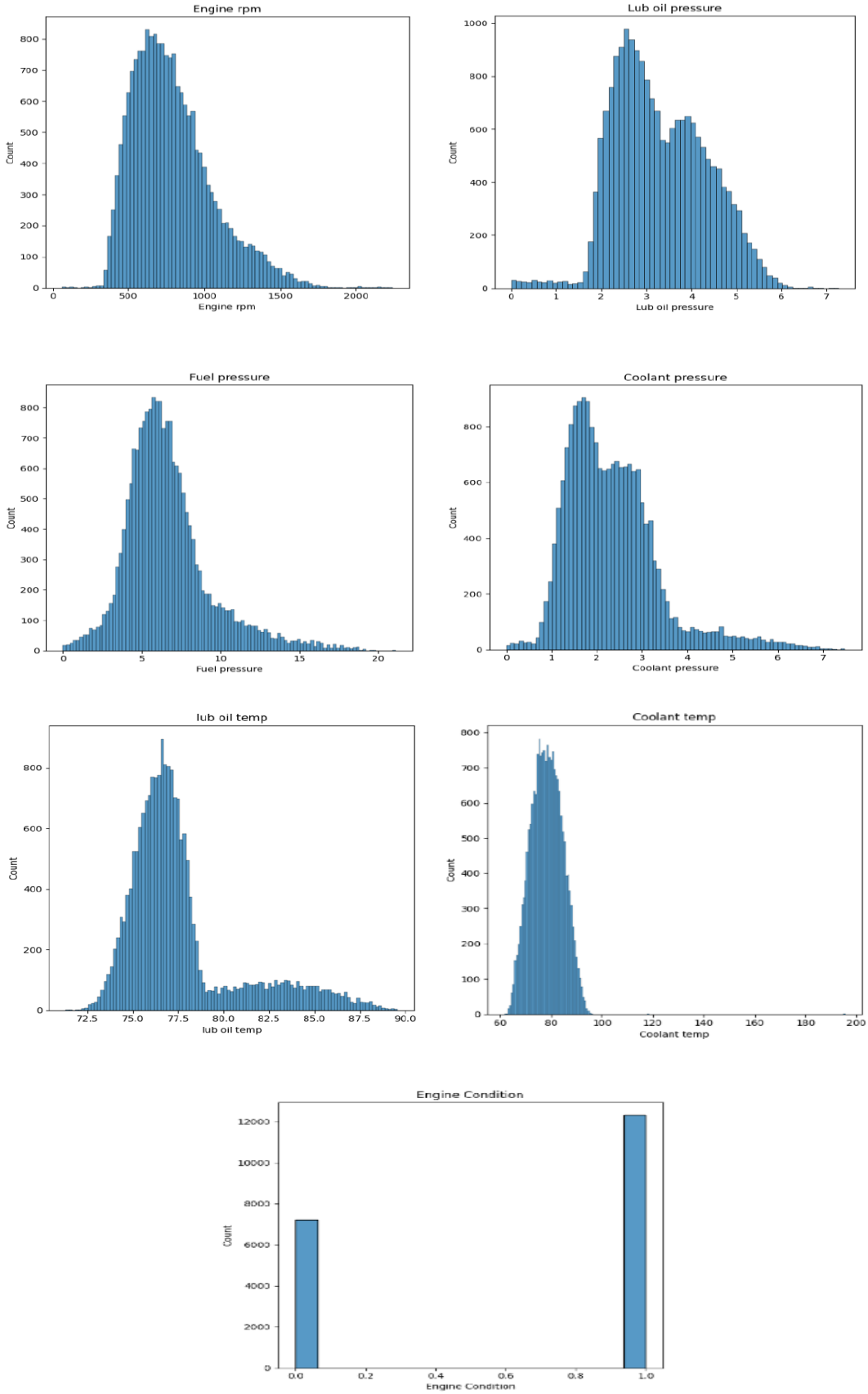
Die Abbildung 3.2.2 liefert zusätzliche Informationen über die Datentypen des Datensatzes. Die Abbildung verdeutlicht, welche Parameter als Ganzzahlen (int64) und welche als Fließkommazahlen (float64) vorliegen.

```
RangeIndex: 19535 entries, 0 to 19534
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Engine rpm         19535 non-null  int64  
 1   Lub oil pressure   19535 non-null  float64
 2   Fuel pressure      19535 non-null  float64
 3   Coolant pressure   19535 non-null  float64
 4   lub oil temp       19535 non-null  float64
 5   Coolant temp       19535 non-null  float64
 6   Engine Condition   19535 non-null  int64  
dtypes: float64(5), int64(2)
memory usage: 1.0 MB
None
```

20-3.2.2: Datentypen

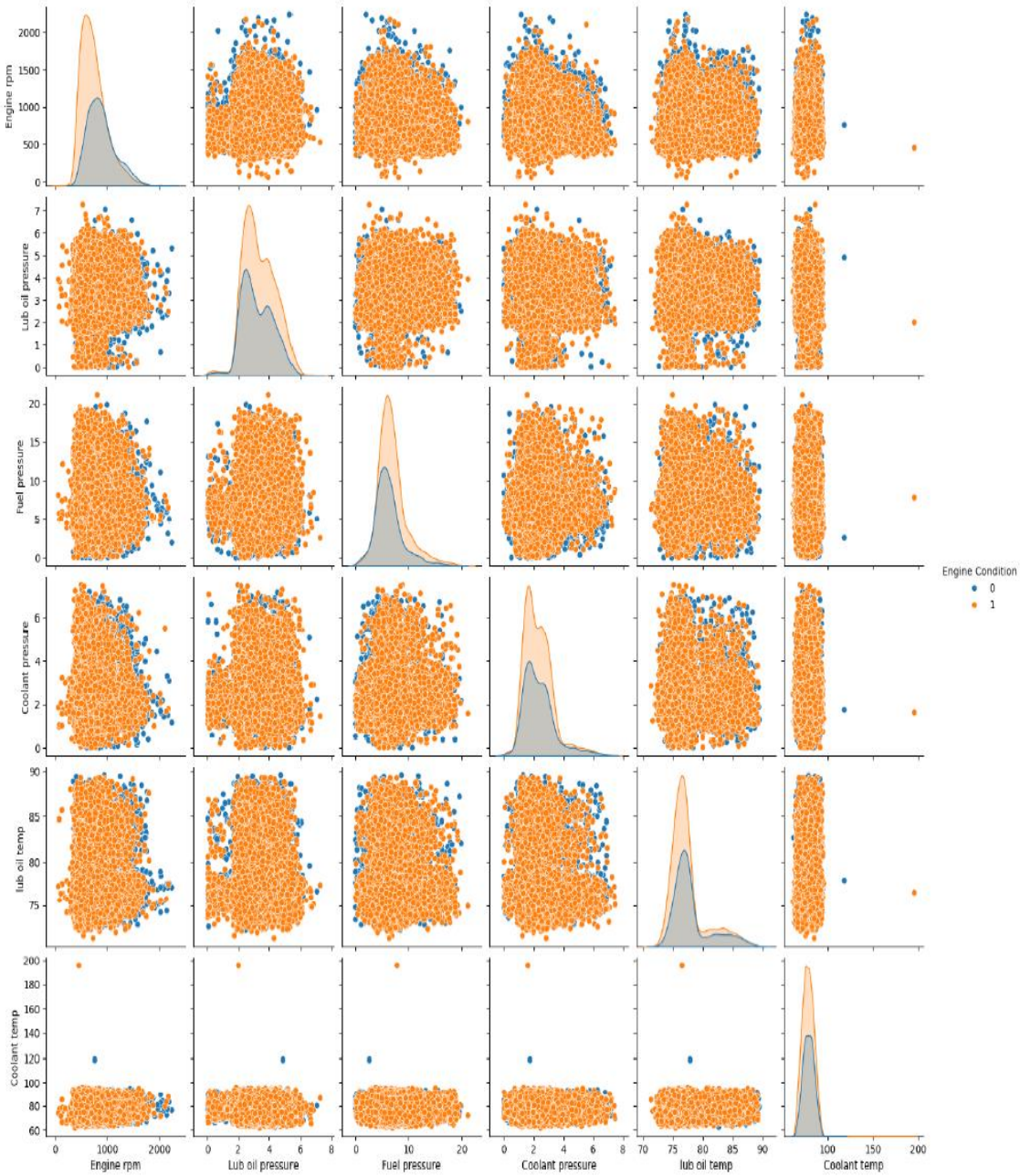
- ❖ Histogram der Merkmale

Die Abbildung 3.2.3 präsentiert ein Histogramm der Merkmale „Engine rpm", „Lub oil pressure", „Fuel pressure", „Coolant pressure", „Lub oil temp", „Coolant temp" und „Engine Condition" und veranschaulicht die Verteilung und Häufigkeit dieser Merkmale. Die X-Achse zeigt die Wertebereiche der jeweiligen Merkmale, während die Y-Achse die Anzahl der Vorkommen (Count) in den jeweiligen Kategorien anzeigt. Die dargestellten Histogramme bieten eine klare Visualisierung der Verteilung der Daten, wodurch Muster und Auffälligkeiten im Datensatz leichter erkennbar werden.



21-3.2.3: Histogramm der Merkmale

Die vorliegende Visualisierung Abbildung 3.2.4 präsentiert Scatterplot-Matrizen und Histogramme zur Verteilung der verschiedenen Motorparameter, unterteilt nach dem Motorzustand (Engine Condition). Die Motorzustände werden in zwei Kategorien unterteilt: 0 und 1. Die blauen Punkte repräsentieren den Zustand 1, während die orangefarbenen Punkte den Zustand 0 darstellen. Die Datenpunkte weisen eine sehr unausgewogene Verteilung innerhalb jeder Histogramm Analyse auf, wodurch sich eine signifikante Auswirkung auf die Genauigkeit der zugrundeliegenden Algorithmen ableiten lässt.



22-3.2.4 Verteilung der verschiedenen Motorparameter.

3.2.1 Datenaufbereitung

- Null Detektion

Im Rahmen der Datenanalyse wurde eine Nullwertanalyse für den betreffenden Datensatz durchgeführt, wie anhand der Abbildungen 3.3.1 ersichtlich ist. Hierbei konnte festgestellt werden, dass der Datensatz keine Nullwerte aufweist.

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
19530	False	False	False	False	False	False	False
19531	False	False	False	False	False	False	False
19532	False	False	False	False	False	False	False
19533	False	False	False	False	False	False	False
19534	False	False	False	False	False	False	False

23-3.2.1: Nullwertanalyse-Tabelle

- Duplikation Detektion

Im Rahmen der Datenaufbereitung wurde ebenfalls eine Duplikat Erkennung durchgeführt. Dabei wurde der Datensatz mit der gebotenen Sorgfalt auf doppelte Einträge überprüft, um sicherzustellen, dass keine redundanten Daten vorhanden sind, welche die Modellentwicklung und -evaluierung verfälschen könnten. Es wurden jedoch keine Duplikate festgestellt, sodass der Datensatz in seiner ursprünglichen Form für die weitere Verarbeitung zur Verfügung steht.

3.3 Korrelationanalyse

Die vorliegende Korrelationstabelle Abbildung 3.3.1 präsentiert die Pearson-Korrelationskoeffizienten zwischen verschiedenen Motorparametern. Die Werte erstrecken sich von -1 bis 1, wobei -1 eine perfekte negative Korrelation, 0 keine Korrelation und 1 eine perfekte positive Korrelation darstellt.

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition
Engine rpm	1.000000	0.025046	-0.001571	-0.024979	0.052134	0.029560	-0.268201
Lub oil pressure	0.025046	1.000000	0.043790	-0.009357	-0.008031	-0.060906	0.060904
Fuel pressure	-0.001571	0.043790	1.000000	0.033264	-0.025338	-0.042986	0.116259
Coolant pressure	-0.024979	-0.009357	0.033264	1.000000	-0.020761	0.033451	-0.024054
lub oil temp	0.052134	-0.008031	-0.025338	-0.020761	1.000000	0.072914	-0.093635
Coolant temp	0.029560	-0.060906	-0.042986	0.033451	0.072914	1.000000	-0.046326
Engine Condition	-0.268201	0.060904	0.116259	-0.024054	-0.093635	-0.046326	1.000000

24-3.3.1: Korrelationstabelle nach Pearson.

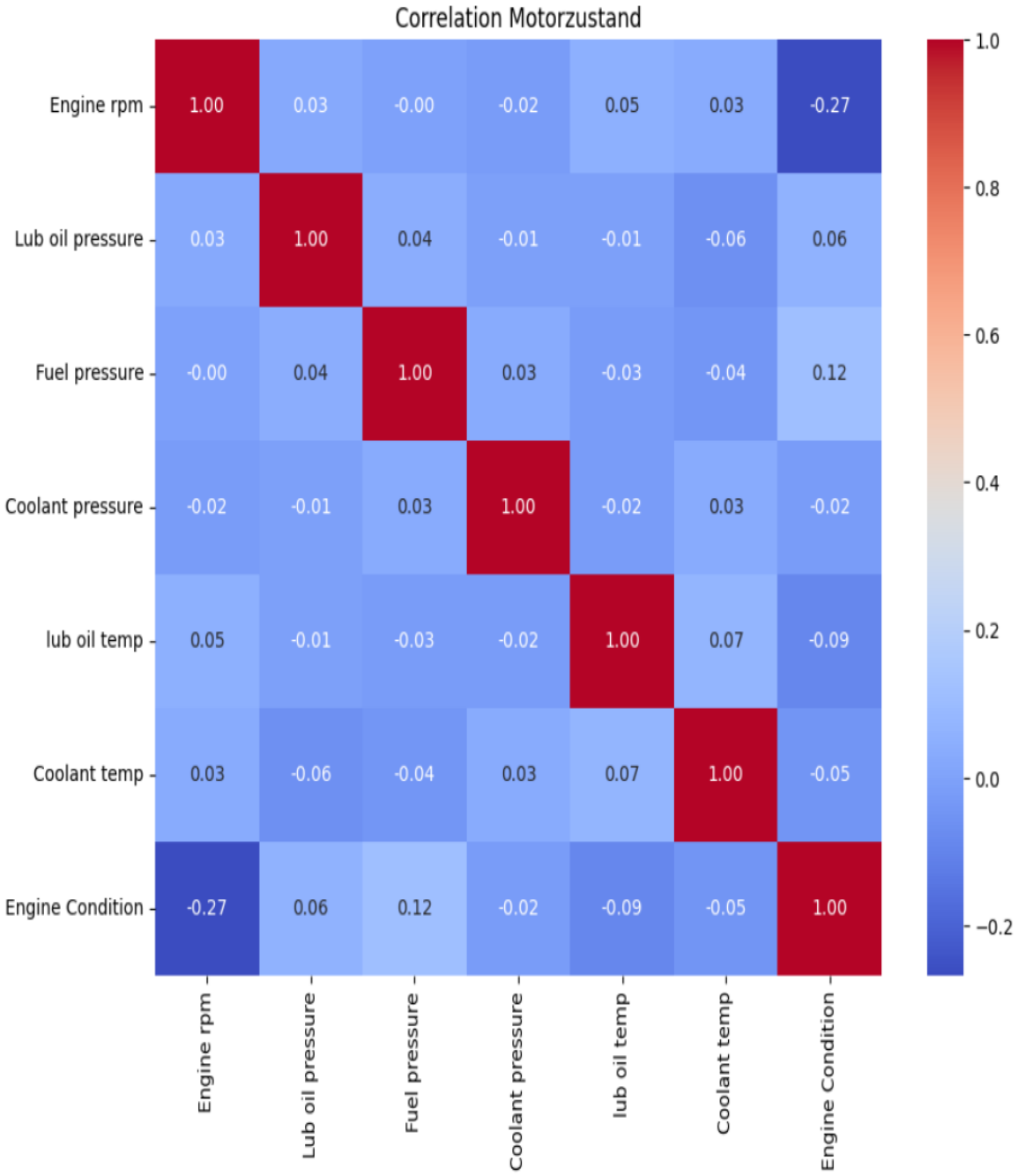
1. Engine rpm (Drehzahl des Motors):
    - Es besteht eine geringe positive Korrelation zum Schmieröldruck (0,025) und zur Schmieröltemperatur (0,052).
    - Eine geringe negative Korrelation zeigt sich beim Kühlmitteldruck (-0,025) und bei der Kühlmitteltemperatur (-0,030).
    - Eine moderate negative Korrelation besteht zum Motorzustand (-0,268).
2. Lub oil pressure (Schmieröldruck):
    - Eine geringe positive Korrelation besteht zum Kraftstoffdruck (0,044) und zum Motorzustand (0,061).

- Eine sehr geringe negative Korrelation zeigt sich beim Kühlmitteldruck (-0,009), bei der Schmieröltemperatur (-0,008) und bei der Kühlmitteltemperatur (-0,061).

3. Fuel pressure (Kraftstoffdruck):
    - Es besteht eine geringe positive Korrelation zum Kühlmitteldruck (0,033) und zum Motorzustand (0,116).
    - Eine geringe negative Korrelation zeigt sich bei der Schmieröltemperatur (-0,025) und bei der Kühlmitteltemperatur (-0,043).
4. Coolant pressure (Kühlmitteldruck):
    - Eine sehr geringe negative Korrelation besteht zur Schmieröltemperatur (-0,021) und zum Motorzustand (-0,024).
    - Eine geringe positive Korrelation zeigt sich bei der Kühlmitteltemperatur (0,033).
5. Lub oil temp (Schmieröltemperatur):
    - Eine sehr geringe negative Korrelation besteht zum Motorzustand (-0,094).
    - Eine geringe positive Korrelation zeigt sich bei der Kühlmitteltemperatur (0,073).
7. Engine Condition (Motorzustand):
    - Die stärkste negative Korrelation besteht zur Drehzahl des Motors (-0,268).
    - Eine geringe positive Korrelation zeigt sich beim Kraftstoffdruck (0,116) und beim Schmieröldruck (0,061).

Die Zusammenfassung der Ergebnisse zeigt, dass die stärkste Korrelation eine moderate negative Beziehung zwischen der Motordrehzahl und dem Motorzustand darstellt. Die meisten weiteren Parameter weisen lediglich geringe oder gar keine Korrelationen untereinander oder mit dem Motorzustand auf. Dies lässt den Schluss zu, dass der Motorzustand am stärksten von der Motordrehzahl beeinflusst wird, während andere Parameter eine geringere Relevanz aufweisen.

Die in Abbildung 3.3.2 dargestellte Heatmap bietet eine anschauliche Visualisierung der Korrelationsmatrix der Motorparameter. Die folgende Darstellung veranschaulicht die Zusammenhänge zwischen den verschiedenen Parametern. Die Farbgebung der Heatmap gibt Aufschluss über die Stärke und Richtung der Korrelationen.



25-3.3.2: Heatmap Korrelationsmatrix.

Die Farbgebung in Rot kennzeichnet positive Korrelationen. Die Intensität der roten Farbe korreliert mit der Stärke der positiven Korrelation zwischen den Parametern. Eine Korrelation von 1,00, wie sie bei den Diagonaleinträgen zu beobachten ist, bedeutet

eine perfekte positive Korrelation. Dies impliziert, dass ein Anstieg eines Parameters mit einem proportionalen Anstieg des anderen einhergeht.

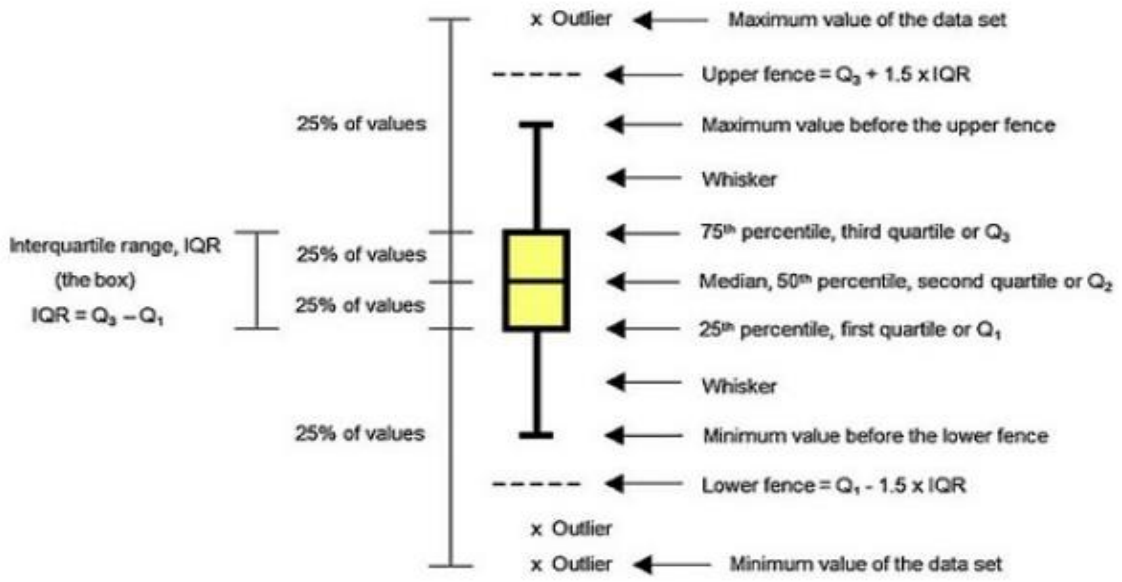
Die Farbe Blau steht für negative Korrelationen. Eine stärkere negative Korrelation wird durch dunklere Blautöne angezeigt. Eine Korrelation von -1,00 würde eine perfekte negative Korrelation bedeuten, jedoch ist dies in der betrachteten Matrix nicht vorhanden.

Weiße Farbtöne sowie hellere Blau- und Rottöne indizieren eine geringe oder gar keine Korrelation. Ein Wert nahe 0 impliziert, dass kein linearer Zusammenhang zwischen den beiden Parametern besteht.

3.4 Ausreißer

3.4.1 Ausreißer identifizieren

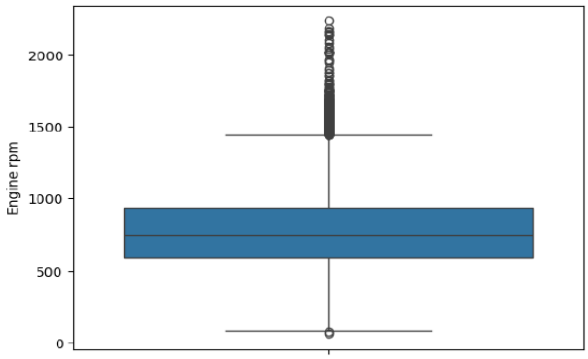
Ein Ausreißer ist ein Datenpunkt, der sich signifikant von der Mehrheit der übrigen Datenpunkte unterscheidet [35]. Im Rahmen dieser Untersuchung wurde die Boxplot-Methode zur Identifikation von Ausreißern angewendet. Die Wahl dieser Methode basiert auf der Erkenntnis, dass der Median und der Quartilsbereich gegenüber Ausreißern weniger anfällig sind als der Mittelwert und die Standardabweichung. Folglich gewährleisten diese robusten Statistiken eine verlässlichere Darstellung der Datenverteilung, auch unter Berücksichtigung von Ausreißern [35]. Die grafische Methode zur Erkennung von Ausreißern besticht durch ihre Einfachheit und Effektivität, da sie extreme potenzielle Ausreißer bei der Berechnung eines Streuungsmaßes unberücksichtigt lässt. Die Festlegung der inneren und äußeren Grenzen eines Boxplots erfolgt anhand der Quartile, wodurch eine Verzerrung durch extreme Werte vermieden wird [66]. Die folgende Abbildung 3.4.1.1 veranschaulicht die wesentlichen Bestandteile eines Boxplots. In diesem Zusammenhang werden zentrale Elemente wie der Median, das obere und untere Quartil sowie mögliche Ausreißer grafisch dargestellt. Die grafische Darstellung erlaubt eine übersichtliche Darstellung der Datenverteilung sowie die Identifikation von Ausreißern.



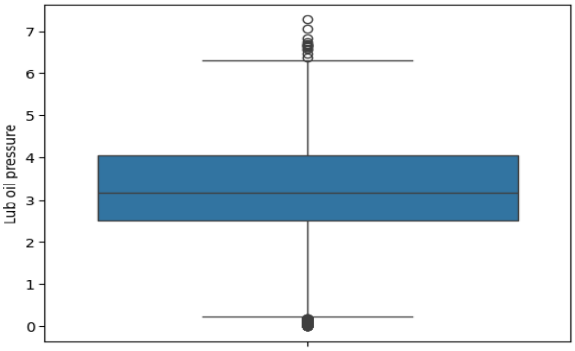
26-3.4.1.1: Die wichtigsten Bestandteile eines Boxplots [19]S.211.

In der Literatur werden drei Hauptmethoden zur Behandlung von Ausreißern in Datensätzen diskutiert: Als Methoden zur Behandlung von Ausreißern in Datensätzen können Trimmen, Winsorisierung und robuste Schätzverfahren angeführt werden. Die Methode des Trimmens beinhaltet die Eliminierung von Ausreißern, was eine Reduktion der Varianz zur Folge hat. Allerdings besteht das Risiko einer Verzerrung, da Ausreißer bei dieser Vorgehensweise unberücksichtigt bleiben. Im Rahmen der Winsorisierung erfolgt eine Anpassung der Gewichte von Ausreißern oder eine Ersetzung ihrer Werte durch geschätzte Werte, um den Einfluss der Ausreißer zu begrenzen. Robuste Schätzverfahren liefern konsistente Schätzungen, die gegenüber Ausreißern robust sind, wobei die bekannte Verteilung der Population als Grundlage dient. Allerdings ist ihre Anwendung aufgrund komplexer methodischer Aspekte begrenzt [35].

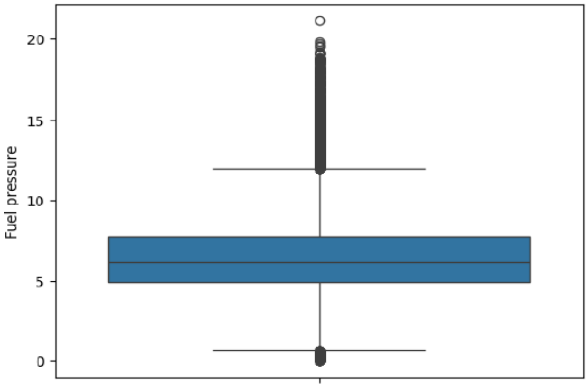
Die unten dargestellten Abbildung 3.4.1.2 Boxplots der verschiedenen Merkmale zeigen deutlich die potenziellen Ausreißer für jedes Merkmal. Die Abbildungen veranschaulichen die potenziellen Ausreißer in den einzelnen Merkmalen mittels der Boxplot-Methode. Die Box stellt den Interquartilsabstand (IQR) dar, welcher die mittleren 50 % der Daten umfasst. Der Median wird als Linie innerhalb der Box visualisiert. Die sogenannten „Whiskers" geben Aufschluss über die maximale und minimale Ausdehnung der Daten, die nicht als Ausreißer klassifiziert werden.



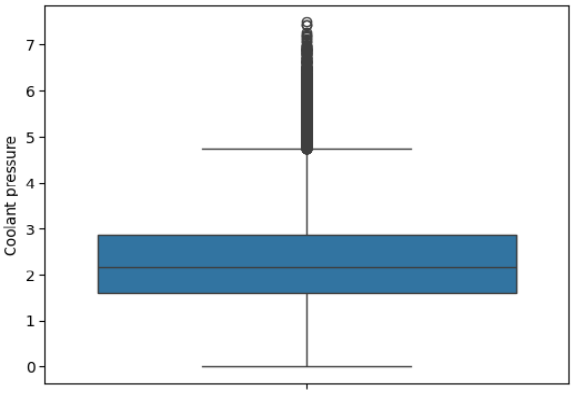
Engine rpm
Upper Outliers: 462
Lower Outliers: 2



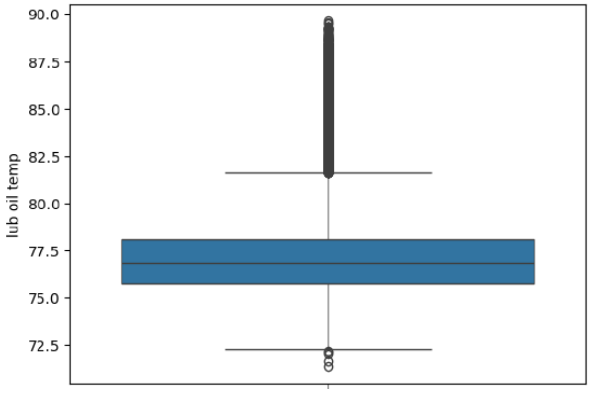
Lib oil pressure
Upper Outliers: 13
Lower Outliers: 53



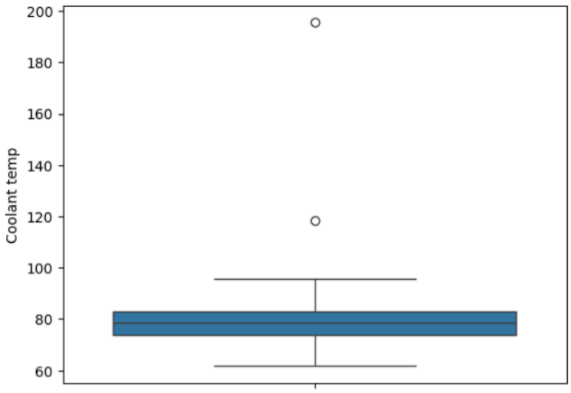
Fuel pressure
Upper Outliers: 1069
Lower Outliers: 66



Coolant pressure
Upper Outliers: 785
Lower Outliers: 0



Lib oil temp
Upper Outliers: 2612
Lower Outliers: 5

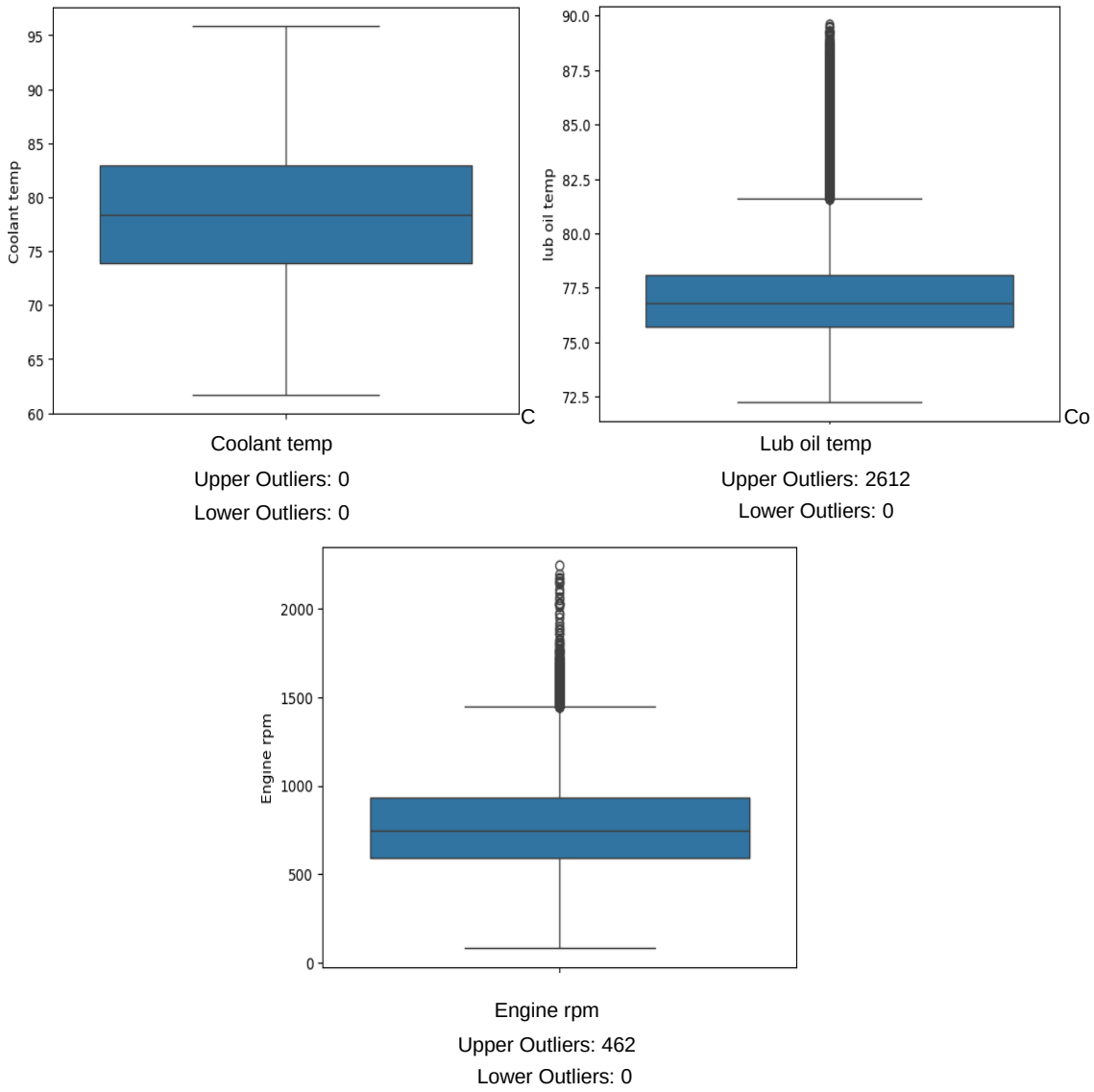


Lower Outliers: 0
Coolant temp
Upper Outliers: 2

27-3.4.1.2: Boxplots der verschiedenen Merkmale.

3.4.2 Ausreißer Behandlung

In der Realität enthalten Daten häufig Beobachtungen, die im Vergleich zur Gesamtheit der Daten als abnormal angesehen werden können [72]. So wurden im vorliegenden Datensatz im oberen Bereich der Schmieröltemperatur 2.612 Datenpunkte als Ausreißer identifiziert. Im Hinblick auf den Kraftstoffdruck wurden 1069 Datenpunkte im oberen Bereich und 66 im unteren Bereich als Ausreißer identifiziert. Beim Kühlmitteldruck wurden 785 Datenpunkte im oberen Bereich als Ausreißer klassifiziert. Dies trifft ebenfalls auf andere Datensatzmerkmale zu. Die Ursachen dieser Abnormalitäten sind vielfältig und eine klare Definition der Grenzen des „Normalen" ist oft schwierig. Solche Abweichungen könnten auf einen alternativen generativen Prozess hindeuten, was impliziert, dass sie möglicherweise einer anderen Verteilung angehören. Alternativ könnten es auch Extremfälle sein, die zwar statistisch selten, aber dennoch möglich sind [72]. Das Histogramm zeigt insbesondere bei der Schmieröltemperatur zwei Normalverteilungen von Datenpunkten, was die Frage aufwirft, wie 2612 Datenpunkte außerhalb des Bereichs liegen können. Bei Betrachtung des Kraftstoffdrucks lässt sich eine Normalverteilung erkennen, wobei 1069 Datenpunkte außerhalb des Bereichs liegen. Aufgrund der fehlenden Expertise im Bereich der Behandlung von Ausreißern sowie der nicht verfügbaren Daten erfassten Expertise war es schwierig, eine Entscheidung über den Umgang mit den Ausreißern zu treffen. Aufgrund des Fehlens eines Experten wurden in dieser Arbeit lediglich drei Merkmale behandelt: Im Rahmen der Ausreißer Behandlung wurden bei der Kühlmitteltemperatur zwei Datenpunkte, welche weit außerhalb des oberen Bereichs lagen, gelöscht. Bei der Schmieröltemperatur wurden im unteren Bereich fünf Datenpunkte, die außerhalb des Bereichs lagen, entfernt. Schließlich wurden bei der Motordrehzahl ebenfalls zwei Datenpunkte im unteren Bereich gelöscht. Die Abbildung 3.4.2.1 präsentiert die Boxplots der drei Merkmale nach der Ausreißer Behandlung.

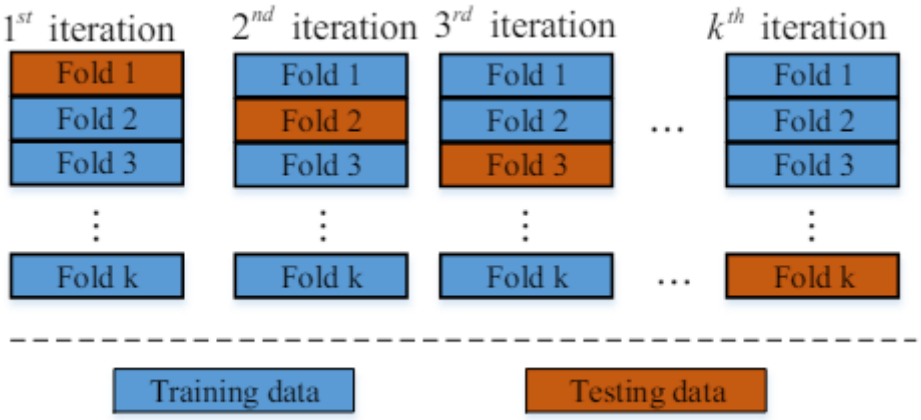


28-3.4.2.1: Die Boxplots der drei Merkmale nach der Ausreißer Behandlung.

3.5 Training Test-Methode

Die nachfolgende Abbildung 3.5.1 veranschaulicht eine Kreuzvalidierung (Cross-validation). Im Rahmen dieser Arbeit wird anstelle der herkömmlichen Aufteilung von 20–80 % in Test- und Trainingsdaten die Kreuzvalidierung angewendet. Die k-fache Kreuzvalidierung stellt eine Methode zur Evaluierung maschineller Lernmodelle dar. Im Rahmen dessen erfolgt eine Unterteilung des Datensatzes in k Teile. In jeder Iteration wird eine Falte als Testdaten verwendet, während die verbleibenden Falten als Trainingsdaten dienen [55]. Im Rahmen dieser Arbeit erfolgt eine Durchführung der k-fachen Kreuzvalidierung mit bis zu fünf Falten. Die Funktion „cross_val_score” liefert die Genauigkeitswerte für jedes der fünf Folds, welche im Anschluss zu einem

Mittelwert zusammengefasst werden, um eine allgemeine Einschätzung der Modellleistung zu ermöglichen.

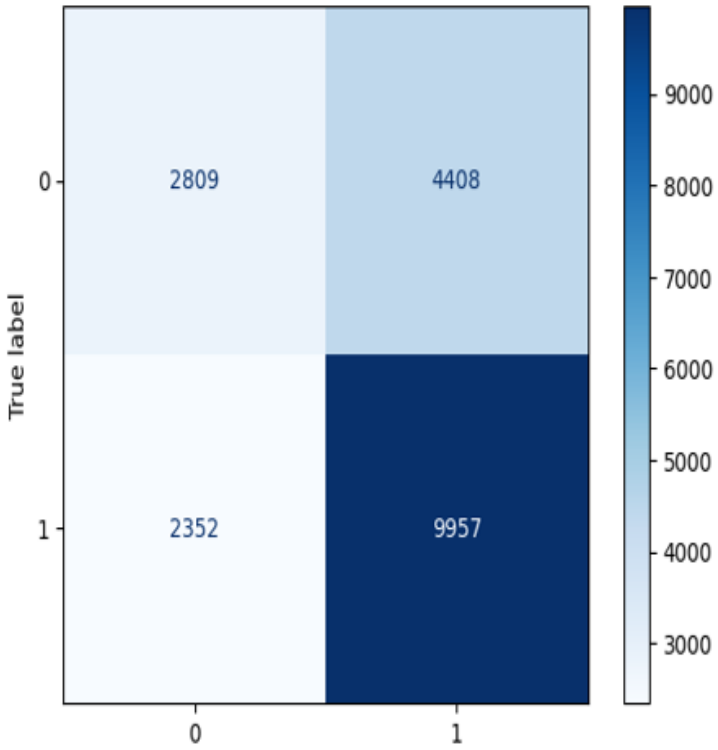


29-3.5.1 K-fold Cross Validition [52].S63.

3.6 Algorithmen

Im vorliegenden Anwendungsbeispiel wird ein Klassifikationsproblem behandelt, das verschiedene Algorithmen des überwachten Lernens einsetzt. Für die Beurteilung der Leistung von Klassifikationsalgorithmen wurde für jeden Algorithmus eine separate Konfusionsmatrix erstellt. Die Struktur der Konfusionsmatrix erlaubt eine systematische Analyse der Klassifikationsergebnisse sowie die Berechnung von Evaluations Metriken zur Leistungsbeurteilung [43]. Metriken stellen wesentliche Kennzahlen zur Evaluierung der Leistungsfähigkeit von Modellen des maschinellen Lernens dar. Mithilfe dieser Kennzahlen sind eine Bewertung und ein Vergleich verschiedener Modelle möglich, wodurch die Auswahl des optimalen Modells für eine spezifische Anwendung erleichtert wird [15]. Die in Abbildung 3.6 dargestellte Konfusionsmatrix veranschaulicht exemplarisch die Leistungsfähigkeit eines Klassifikators. Die Matrix präsentiert die Anzahl der tatsächlichen und vorhergesagten Klassen, um eine Evaluierung der Modellgenauigkeit zu ermöglichen. Zusätzlich erlaubt die Farbcodierung die Ermittlung der Häufigkeit der Werte in den jeweiligen Zellen. Hierbei repräsentieren dunklere Farben höhere Werte. Die Gesamtsummen der tatsächlich existierenden Klassen lassen sich wie folgt darstellen:

Die Gesamtsumme der Datenpunkte der Klasse 0 (außer Betrieb) beläuft sich auf 7217. Dieser Wert ergibt sich aus der Summe der ersten Zeile in der Konfusionsmatrix. Die Klasse 1 (gesund) umfasst 12.309 Datenpunkte, was sich aus der Summe der zweiten Zeile in der Konfusionsmatrix ergibt.



30-3.6: Beispiel einer Konfusionsmatrix

Im Rahmen der Analyse werden die Metriken „Genauigkeit" (Accuracy), „Sensitivität", „Spezifität", „Präzision" und „F1-Score" zur Untersuchung der Algorithmen auf dem Datensatz herangezogen. Das Performancemaß „Genauigkeit" (Accuracy) stellt eine grundlegende Kennzahl zur Beurteilung der Richtigkeit einer Klassifikation dar. Das Verhältnis der korrekt klassifizierten Objekte zur Gesamtanzahl aller Objekte wird in diesem Kontext beschrieben. Die Berechnung erfolgt gemäß der folgenden Formel: Die Accuracy lässt sich gemäß der folgenden Formel berechnen:

Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + True Negatives + False Negatives) [43]. Die Berechnung der Genauigkeit (Accuracy) für jeden Algorithmus erfolgt auf Basis der abgeleiteten Werte der Konfusionsmatrix, wobei die entsprechende Formel mathematisch wie folgt lautet:

$$Accuracy = \frac{TP + TN}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP} + \mathrm{TN}}$$

Die Sensitivität gibt den Anteil aller korrekt als positiv vorhergesagten Eingaben an und dient somit als Maß für die Entdeckungswahrscheinlichkeit. Die Präzision gibt den Anteil der korrekt vorhergesagten positiven Eingaben an. Die Berechnung erfolgt gemäß der Formel TP/(TP+FP). Eine alternative Formel zur Berechnung lautet TP/(TP+FN). Die Spezifität ermöglicht die Berechnung des Anteils aller negativen Proben, die korrekt als negativ vorhergesagt wurden, durch TN/(TN+FP). Der F1-Score kombiniert Präzision und Sensitivität zu einem Mittelwert und kann durch die Formel 2TP/(2TP+FP+FN) berechnet werden [48].

Die eingesetzten Algorithmen umfassen Random Forest, Entscheidungsbaum, logistische Regression, Support Vector Machine (SVM) und Naive Bayes. In Bezug auf die Klassifizierung von Daten lassen sich zwei grundlegende Ansätze unterscheiden. Der erste Ansatz basiert auf einer binären Klassifizierung, bei der ein unbekanntes Datenelement entweder der Klasse 0 oder der Klasse 1 zugewiesen wird. Der zweite Ansatz zielt auf die Modellierung von P (y | x) ab, was nicht nur eine Klassenbezeichnung für ein Datenelement liefert, sondern auch die Wahrscheinlichkeit der Klassenzugehörigkeit angibt.

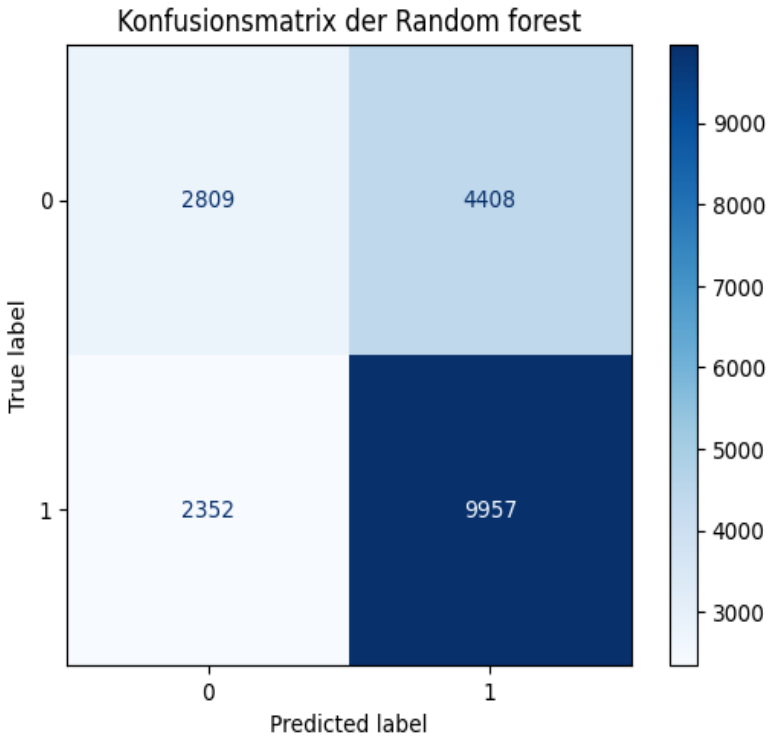
Zu den bekanntesten Vertretern des ersten Ansatzes zählen Support Vector Machines. Zu den Methoden des zweiten Ansatzes zählen logistische Regression, künstliche neuronale Netze, k-Nearest Neighbors (KNN) sowie Entscheidungsbäume. Die genannten Methoden unterscheiden sich erheblich in der Art und Weise, wie sie eine Näherung an P (y | x) aus den Daten ableiten [16]. Im Folgenden erfolgt eine detaillierte Betrachtung jedes einzelnen Algorithmus.

3.6.1 Random Forest

Die Random-Forest-Methode zeichnet sich durch die Anwendung von Bootstrap-Aggregation (Bagging) und der zufälligen Auswahl von Prädiktoren aus. Durch diese beiden Ansätze wird eine hohe Vorhersagegenauigkeit erreicht [61]. Bagging stellt eine Kurzform von Bootstrap-Aggregation dar und bezeichnet die zufällige Auswahl von Datenstichproben sowie die Aggregation der Vorhersagen mehrerer Entscheidungsbäume, mit dem Ziel, Verzerrungen durch Ausreißer zu minimieren. Diese Methode findet Anwendung im Random Forest Algorithmus zur Vorhersageerstellung. Der Begriff „Random” bezeichnet die zufällige Auswahl von Daten aus einem ursprünglichen Datensatz. Der Begriff „Forest” hingegen beschreibt

den Aufbau mehrerer Entscheidungsbäume. Jeder Baum basiert dabei auf einer eigenen, zufälligen Datenstichprobe [4].

Die in Abbildung 3.6.1.1 dargestellte Konfusionsmatrix visualisiert die Leistungsfähigkeit des Random-Forest-Klassifikators.



31-3.6.1.1: Konfusionmatrix der Random Forest

Im Folgenden werden die Werte der Konfusionsmatrix des Random Forest interpretiert:

- 2809 (TN): Das Modell hat 2809 Fälle korrekt als Klasse 0 (negative Klasse) identifiziert.
- 4408 (FP): Das Modell hat 4408 Fälle fälschlicherweise als Klasse 1 (positive Klasse) vorhergesagt, obwohl sie zur Klasse 0 gehören.
- 2352 (FN): Das Modell hat 2352 Fälle fälschlicherweise als Klasse 0 (negative Klasse) vorhergesagt, obwohl sie zur Klasse 1 gehören.
- 9957 (TP): Das Modell hat 9957 Fälle korrekt als Klasse 1 (positive Klasse) identifiziert.

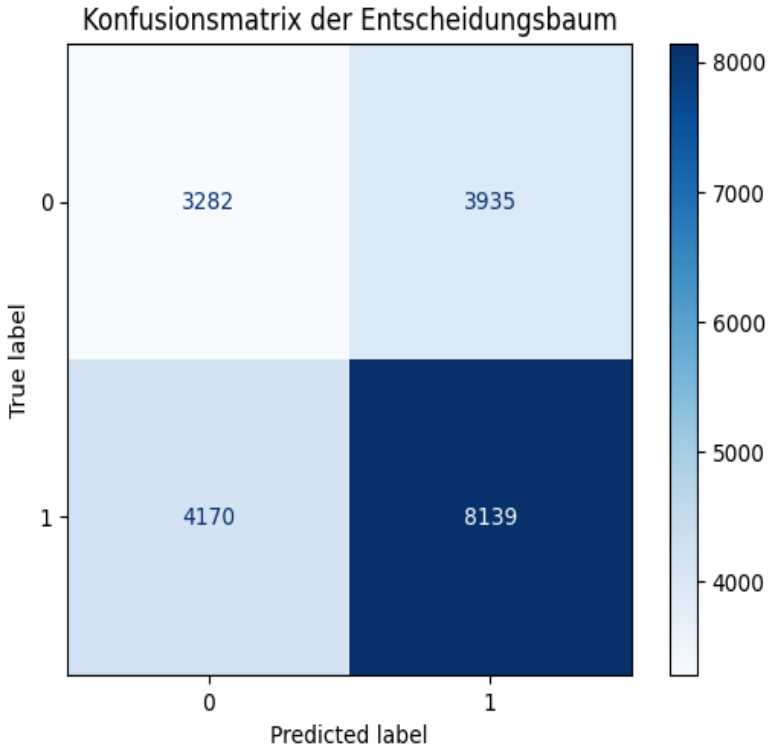
Die nachfolgende tabellarische Darstellung präsentiert die Ergebnisse der Kreuzvalidierung für jeden Testdurchlauf sowie die daraus abgeleiteten Metriken Sensitivität, Spezifität, Präzision und F1-Score. Zudem wird die durchschnittliche Genauigkeit des Random-Forest-Algorithmus für den Gesamtdatensatz ausgewiesen.

Table 1:Random Forest Ergebnistabelle.

K-Fold	Fold1	Fold2	Fold3	Fold4	Fold5
Cross-Validation Scores	0.65232975	0.65096031	0.64814341	0.65275288	0.66478873
Sensitivität	80.9%				
Spezifität	38.9%				
Präzision	69.3%				
F1-Score	74.7%				
Mean CV Accuracy	65.37%				

3.6.1 Entscheidungsbaum

Ein Entscheidungsbaum stellt einen komplexen, oft mehrstufigen Entscheidungsprozess in einer transparenten Weise dar, indem er alle möglichen Entscheidungsoptionen aufzeigt. Die Verästelungen des Baumes visualisieren die verketteten Entscheidungen, die sowohl in zeitlicher als auch in logischer Abfolge zueinanderstehen können. Dabei können die Entscheidungsbäume sowohl auf logischen als auch auf mathematischen Prinzipien basieren [62]. Entscheidungsbäume lassen sich grundsätzlich in zwei Typen unterteilen, nämlich in Klassifikationsbäume und Regressionsbäume. Klassifikationsbäume werden verwendet, wenn die abhängige Zielgröße nominal skaliert ist, während Regressionsbäume bei einer quantitativen Zielgröße zum Einsatz kommen [18]. Die in Abbildung 3.6.1.2 dargestellte Konfusionsmatrix visualisiert die Leistungsfähigkeit des Entscheidungsbaum-Klassifikators.



32-3.6.1.2: Konfusionmatrix des Entscheidungsbaums

Im Folgenden erfolgt eine Interpretation der Werte der Konfusionsmatrix des Entscheidungsbaums:

- 3282 (TN): Das Modell hat 3282 Fälle korrekt als Klasse 0 (negative Klasse) identifiziert.
- 3935 (FP): Das Modell hat 3935 Fälle fälschlicherweise als Klasse 1 (positive Klasse) vorhergesagt, obwohl sie zur Klasse 0 gehören.
- 4170 (FN): Das Modell hat 4170 Fälle fälschlicherweise als Klasse 0 (negative Klasse) vorhergesagt, obwohl sie zur Klasse 1 gehören.
- 8139 (TP): Das Modell hat 8139 Fälle korrekt als Klasse 1 (positive Klasse) identifiziert.

Die nachfolgende tabellarische Darstellung präsentiert die Ergebnisse der Kreuzvalidierung für jeden Testdurchlauf sowie die daraus abgeleiteten Metriken

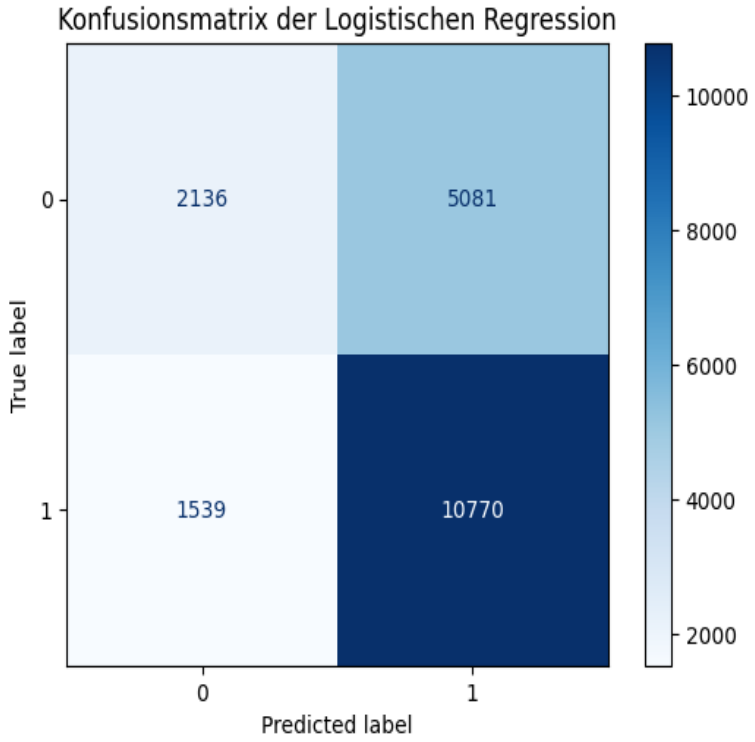
Sensitivität, Spezifität, Präzision und F1-Score. Zudem wird die durchschnittliche Genauigkeit des Entscheidungsbaum-Algorithmus für den Gesamtdatensatz ausgewiesen.

Table 2:Tabelle der Entscheidungsbaumergebnisse

K-Fold	Fold1	Fold2	Fold3	Fold4	Fold5
Cross-Validation Scores	0.58806964	0.59052497	0.58104994	0.58873239	0.57618438
Sensitivität	66.1%				
Spezifität	45.5%				
Präzision	67.4%				
F1-Score	66.6%				
Mean CV Accuracy	58.49%				

3.6.1 logistische Regression

Im Folgenden soll zunächst der Unterschied zwischen linearer und logistischer Regression verdeutlicht werden. Die lineare Regression ist ein statistisches Analyseverfahren, das auf der Regressionsanalyse der mathematischen Statistik basiert und dazu dient, die quantitative Beziehung zwischen zwei oder mehr Variablen zu ermitteln. Die betreffende Methode findet häufig Anwendung bei der Prognose von kontinuierlichen Daten. Der wesentliche Unterschied zur linearen Regression besteht darin, dass die Datenpunkte nicht in einer durchgehenden Linie angeordnet sind. Stattdessen können sie in Gruppen oder Stapeln vorliegen, wobei jeder Stapel eine bestimmte Kategorie repräsentiert und alle Datenpunkte innerhalb eines Stapels dieselbe Kategorie Bezeichnung tragen. Bei der logistischen Regression wird ein beliebiger Satz von Eingaben genommen, und die Ausgabe erfolgt über eine Funktion, die die Klassifizierung der Eingabedaten vornimmt [77]. Die in Abbildung 3.6.1.3 dargestellte Konfusionsmatrix visualisiert die Leistungsfähigkeit des logistische Regression-Klassifikators.



33-3.6.1.3: Konfusionsmatrix der logistischen Regression

Interpretation der Werte der Konfusionmatrix der logistischen Regression:

- 2136 (TN): Das Modell hat 2136 Fälle korrekt als Klasse 0 (negative Klasse) identifiziert.
- 5081 (FP): Das Modell hat 5081 Fälle fälschlicherweise als Klasse 1 (positive Klasse) vorhergesagt, obwohl sie zur Klasse 0 gehören.
- 1539 (FN): Das Modell hat 1539 Fälle fälschlicherweise als Klasse 0 (negative Klasse) vorhergesagt, obwohl sie zur Klasse 1 gehören.
- 10770 (TP): Das Modell hat 10770 Fälle korrekt als Klasse 1 (positive Klasse) identifiziert.

Die nachfolgende tabellarische Darstellung präsentiert die Ergebnisse der Kreuzvalidierung für jeden Testdurchlauf sowie die daraus abgeleiteten Metriken Sensitivität, Spezifität, Präzision und F1-Score. Zudem wird die durchschnittliche

Genauigkeit des logistischen Regression -Algorithmus für den Gesamtdatensatz ausgewiesen.

Table 3: Ergebnistabelle der logistischen Regression

K-Fold	Fold1	Fold2	Fold3	Fold4	Fold5
Cross-Validation Scores	0.65514593	0.66734955	0.66760563	0.65224072	0.66248399
Sensitivität	87.5%				
Spezifität	29.6%				
Präzision	67.9%				
F1-Score	76.4%				
Mean CV Accuracy	66.09%				

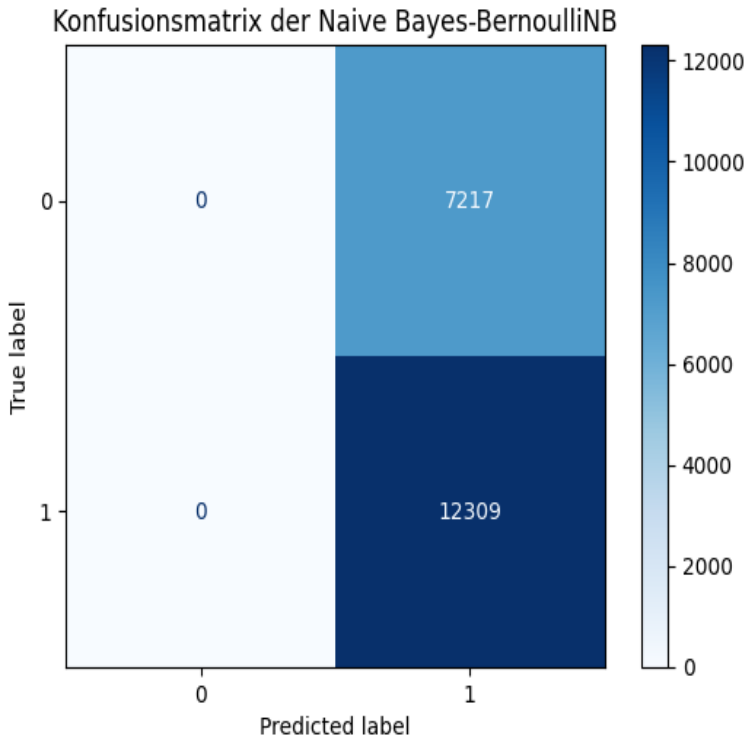
3.6.2 Naive Bayes

Es lassen sich drei Arten von Naive Bayes Modellen unterscheiden:

- Gaussian
- Bernoulli
- Multinomial

Die Anwendung der Gauß-Verteilung ist bei diskreten Datensätzen nicht möglich, da sie nur bei kontinuierlichen Trainingsdaten verwendet werden sollte. Diskrete Daten weisen lediglich eine begrenzte Anzahl möglicher Werte auf. Im Gegensatz dazu klassifiziert die Bernoulli-Verteilung das Ergebnis in nur zwei Merkmale oder Klassen, wie beispielsweise wahr oder falsch, ja oder nein sowie 0 oder 1. Wenn das Ergebnis als Klassentyp oder als bestimmter Typ mit zwei Variablen vorhergesagt werden muss, wird diese Art der Klassifizierung verwendet. Die Bernoulli-Verteilung bildet die Grundlage für die Algorithmen, die in dieser Arbeit angewendet werden [17]. Die in

Abbildung 3.6.1.4 dargestellte Konfusionsmatrix visualisiert die Leistungsfähigkeit des Naive Bayes -Klassifikators.



34-3.6.1.4: Konfusionsmatrix der BernoulliNB

Im Folgenden erfolgt eine Interpretation der Werte der Konfusionsmatrix des Bernoulli-Modells:

- 0 (TN): Das Modell hat 0 Fälle korrekt als Klasse 0 (negative Klasse) identifiziert.
- 7217 (FP): Das Modell hat 7217 Fälle fälschlicherweise als Klasse 1 (positive Klasse) vorhergesagt, obwohl sie zur Klasse 0 gehören.
- 0 (FN): Das Modell hat 0 Fälle fälschlicherweise als Klasse 0 (negative Klasse) vorhergesagt, obwohl sie zur Klasse 1 gehören.
- 12309 (TP): Das Modell hat 12309 Fälle korrekt als Klasse 1 (positive Klasse) identifiziert.

Das hier präsentierte Modell (Bernoulli) zeigt eine ausgeprägte Neigung, sämtliche Instanzen der Klasse 1 zuzuordnen, unabhängig von deren tatsächlicher Klassenzugehörigkeit. Dies resultiert in einer signifikanten Anzahl von falsch positiven Klassifizierungen und einer perfekten Klassifizierung der Instanzen der Klasse 1,

während Instanzen der Klasse 0 vollständig fehlklassifiziert werden. Das beobachtete Verhalten deutet auf eine signifikante Unausgewogenheit des Modells sowie eine starke Präferenz für die Klasse 1 hin. Eine Analyse der Konfusionsmatrix zeigt, dass das verwendete Naive Bayes-Bernoulli-Modell in diesem spezifischen Fall eine unzureichende Leistungsfähigkeit aufweist.

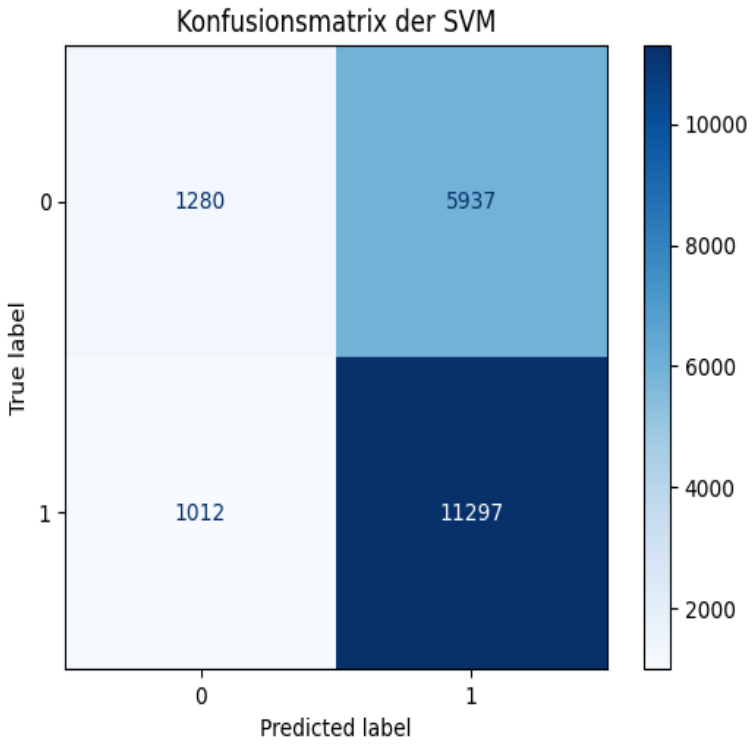
Die nachfolgende tabellarische Darstellung präsentiert die Ergebnisse der Kreuzvalidierung für jeden Testdurchlauf sowie die daraus abgeleiteten Metriken Sensitivität, Spezifität, Präzision und F1-Score. Zudem wird die durchschnittliche Genauigkeit des Naive Bayes -Algorithmus für den Gesamtdatensatz ausgewiesen.

Table 4:Naive Bayes-Ergebnistabelle

K-Fold	Fold1	Fold2	Fold3	Fold4	Fold5
Cross-Validation Scores	0.63031234	0.63047375	0.63047375	0.63047375	0.63021767
Sensitivität	100%				
Spezifität	0%				
Präzision	63.0%				
F1-Score	77.3%				
Mean CV Accuracy	63.03%				

3.6.1 Support Vector Maschine (SVM)

Eine Support Vector Maschine (SVM) ist eine Technik der binären linearen Klassifizierung im Bereich des Maschinellen Lernens. Die SVM zeichnet sich dadurch aus, dass sie die Klassen durch eine möglichst große Lücke, den sogenannten optimalen Rand, voneinander trennt. Die Instanzen, die diesen Rand bestimmen, werden als Support-Vektoren bezeichnet. Aufgrund dieser Eigenschaft wird die SVM auch als Optimal Margin Classifier bezeichnet [10]. Die Klassifizierung erfolgt durch die Implementierung einer linearen oder nichtlinearen Trennfläche im Eingangsraum [75]. Die in Abbildung 3.6.1.5 dargestellte Konfusionsmatrix visualisiert die Leistungsfähigkeit des Naive Bayes -Klassifikators.



35-3.6.1.5: Konfusionsmatrix der SVM

Interpretation der Werte der Konfusionmatrix der SVM Im Folgenden soll eine Interpretation der Werte der Konfusionsmatrix der SVM vorgenommen werden:

- 1280 (TN): Das Modell hat 1280 Fälle korrekt als Klasse 0 (negative Klasse) identifiziert.
- 5937 (FP): Das Modell hat 5937 Fälle fälschlicherweise als Klasse 1 (positive Klasse) vorhergesagt, obwohl sie zur Klasse 0 gehören.
- 1012 (FN): Das Modell hat 1012 Fälle fälschlicherweise als Klasse 0 (negative Klasse) vorhergesagt, obwohl sie zur Klasse 1 gehören.
- 11297 (TP): Das Modell hat 11297 Fälle korrekt als Klasse 1 (positive Klasse) identifiziert.

Die nachfolgende tabellarische Darstellung präsentiert die Ergebnisse der Kreuzvalidierung für jeden Testdurchlauf sowie die daraus abgeleiteten Metriken Sensitivität, Spezifität, Präzision und F1-Score. Zudem wird die durchschnittliche

Genauigkeit des Support Vector Maschine -Algorithmus für den Gesamtdatensatz ausgewiesen.

Table 5:Ergebnistabelle der Support-Vektor-Maschine (SVM)

K-Fold	Fold1	Fold2	Fold3	Fold4	Fold5
Cross-Validation Scores	0.63952893	0.64763124	0.65121639	0.64020487	0.64199744
Sensitivität	91.8%				
Spezifität	17.7%				
Präzision	65.6%				
F1-Score	76.5%				
Mean CV Accuracy	64.41%				

4 Fazit und Ausblick

Im Rahmen dieser Masterthesis wurde ein prädiktives Modell für die vorausschauende Wartung von Automotoren entwickelt. In diesem Zusammenhang wurde ein Datensatz der Webseite Kaggle verwendet, der verschiedene Motorparameter umfasst. Im Rahmen der Analyse wurden die Merkmale Motordrehzahl, Schmieröldruck, Kraftstoffdruck, Kühlmitteldruck, Schmieröltemperatur, Kühlmitteltemperatur und Motorzustand als Zielvariable betrachtet. Da es sich bei dem Anwendungsbeispiel um ein Klassifikationsproblem handelt, wurden verschiedene maschinelle Lernalgorithmen des überwachten Lernens eingesetzt. Die Evaluierung der Genauigkeit und Leistung der Klassifizierung erfolgte anhand der Algorithmen Random Forest, Entscheidungsbaum, logistische Regression, Support Vector Machine (SVM) und Naive Bayes. Die Evaluierung der Algorithmus-Genauigkeit erfolgte anhand einer Konfusionsmatrix, welche in der nachfolgenden Tabelle 6 die tatsächlichen und vorhergesagten Klassen von allen Algorithmen zusammenfassend darstellt.

Table 6:Zusammenstellung der Ergebnisse der Konfusionsmatrix der Algorithmen

0	2809	4408	Random Forest
	3282	3935	Entscheidungsbaum
	2136	5081	Log-Regression
	1280	5937	SVM
	0	7217	Naive Bayes
1	2352	9957	Random Forest
	4170	8139	Entscheidungsbaum
	1539	1077	Log-Regression
	1012	11297	SVM
	0	12309	Naive Bayes
	0	**1**	

Die Vorgehensweise umfasste das Datenverständnis, die Durchführung einer Korrelationsanalyse, die Datenaufbereitung, die Identifikation und Behandlung von Ausreißern, die Anwendung der Kreuzvalidierung als Training-Test-Methode sowie die

Anwendung von Algorithmen. Die nachfolgende Tabelle 7 präsentiert eine vergleichende Analyse der Leistungsfähigkeit angewendeter Klassifikationsalgorithmen, wobei die Kriterien Genauigkeit, Sensitivität, Spezifität, Präzision und F1-Score herangezogen werden. Der Random Forest-Algorithmus zeigt eine ausgewogene Sensitivität und Präzision, was zu einem hohen F1-Score von 74,7 % führt, obschon die Spezifität gering ist. Der Entscheidungsbaum zeigt die geringste Genauigkeit und den niedrigsten F1-Score, was auf eine suboptimale Effektivität in diesem Szenario hinweist. Die logistische Regression erreicht die höchste Genauigkeit von 66,096 % sowie einen hohen F1-Score von 76,4 %. Dies ist auf eine ausgewogene Balance zwischen Sensitivität und Präzision zurückzuführen. Der SVM-Algorithmus zeigt eine hohe Sensitivität von 91,8 % sowie eine niedrige Spezifität von 17,7 %. Dies impliziert eine hohe Anzahl an falsch-positiven Klassifikationen. Der Naive Bayes-Algorithmus weist die höchste Sensitivität von 100 % auf, allerdings werden negative Fälle nicht korrekt erkannt, sodass es zu einer Spezifität von 0 % kommt. Dennoch erreicht der Algorithmus einen hohen F1-Score von 77,3 %.

Table 7:Vergleichstabelle der Algorithmus Ergebnisse

Algorithmen	Genauigkeit	Sensitivität	Spezifität	Präzision	F1-Score
Random Forest	65.379%	80.9%	38.9%	69.3%	74.7%
Entscheidungsbaum	58.491%	66.1%	45.5%	67.4%	66.6%
logistische Regression	66.096%	87.5%	29.6%	67.9%	76.4%
SVM	64.411%	91.8%	17.7%	65.6%	76.5%
Naive Bayes	63.039%	100%	0%	63.0%	77.3%

Ein weiteres potenzielles Anwendungsgebiet des Datensatzes ist die Leistungsanalyse verschiedener Motoren- und Fahrzeugtypen als Clustering-Problem. Forscherinnen und Forscher könnten die Daten beispielsweise nutzen, um die Leistung von Motoren verschiedener Hersteller zu vergleichen oder die Effizienz unterschiedlicher Wartungsstrategien zu bewerten. Auf diese Weise könnten Innovationen und Verbesserungen in der Automobilindustrie gefördert werden.

Anhang

In [1]:
```
import pandas as pd
import numpy as np
#Initial data frame
df = pd.read_csv("engine_data.csv")
df. head(10)
```

Out[1]:

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition
0	700	2.493592	11.790927	3.178981	84.144163	81.632187	1
1	876	2.941606	16.193866	2.464504	77.640934	82.445724	0
2	520	2.961746	6.553147	1.064347	77.752266	79.645777	1
3	473	3.707835	19.510172	3.727455	74.129907	71.774629	1
4	619	5.672919	15.738871	2.052251	78.396989	87.000225	0
5	1221	3.989226	6.679231	2.214250	76.401152	75.669818	0
6	716	3.568896	5.312266	2.461067	83.646589	79.792411	1
7	729	3.845166	10.191126	2.362998	77.921202	71.671761	1
8	845	4.877239	3.638269	3.525604	76.301626	70.496024	0
9	824	3.741228	7.626214	1.301032	77.066520	85.143297	0

In [2]:
```
#Shape of Dataframe
print(df.shape)
print(df.info())
```

```
(19535, 7)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19535 entries, 0 to 19534
Data columns (total 7 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Engine rpm        19535 non-null  int64  
 1   Lub oil pressure  19535 non-null  float64
 2   Fuel pressure     19535 non-null  float64
 3   Coolant pressure  19535 non-null  float64
 4   lub oil temp      19535 non-null  float64
 5   Coolant temp      19535 non-null  float64
 6   Engine Condition  19535 non-null  int64  
dtypes: float64(5), int64(2)
memory usage: 1.0 MB
None
```

In [3]:
```
# Nulling Detection
df.isnull()
```

Out[3]:

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
19530	False	False	False	False	False	False	False
19531	False	False	False	False	False	False	False
19532	False	False	False	False	False	False	False
19533	False	False	False	False	False	False	False
19534	False	False	False	False	False	False	False

19535 rows × 7 columns

In [4]:
```
df.isnull().sum()
```

Out[4]:
```
Engine rpm          0
Lub oil pressure    0
Fuel pressure       0
Coolant pressure    0
lub oil temp        0
Coolant temp        0
Engine Condition    0
dtype: int64
```

In [5]:
```
#duplicate Detection
duplicate_rows= df[df.duplicated()]
duplicate_rows
```

Out[5]:

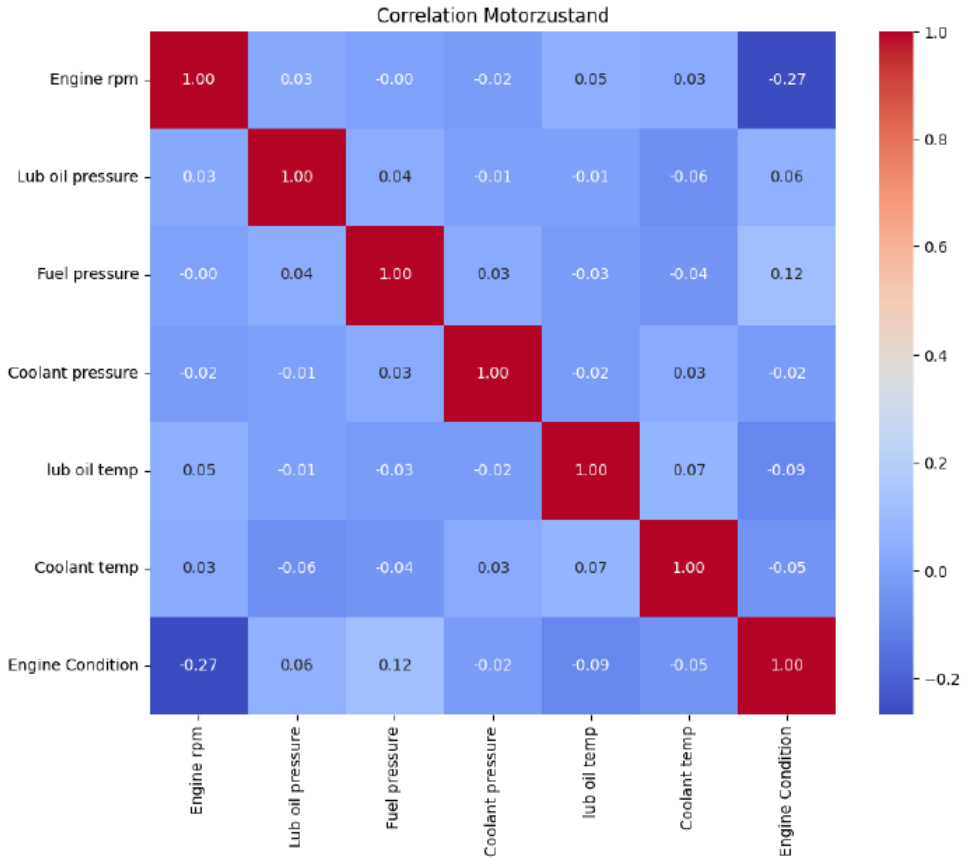
	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	Engine Condition

In [6]:
```
#Correlation
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
df.corr()
```

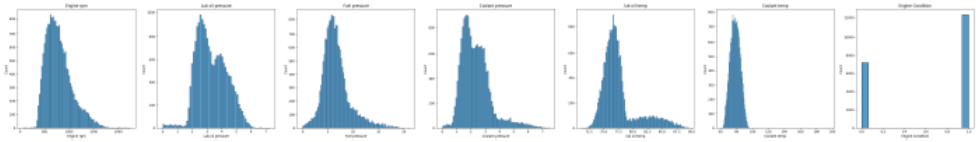
Out[6]:

	Engine rpm	Lub oil pressure	Fuel pressure	Coolant pressure	lub oil temp	Coolant temp	C
Engine rpm	1.000000	0.025046	-0.001571	-0.024979	0.052134	0.029560	-
Lub oil pressure	0.025046	1.000000	0.043790	-0.009357	-0.008031	-0.060906	
Fuel pressure	-0.001571	0.043790	1.000000	0.033264	-0.025338	-0.042986	
Coolant pressure	-0.024979	-0.009357	0.033264	1.000000	-0.020761	0.033451	-
lub oil temp	0.052134	-0.008031	-0.025338	-0.020761	1.000000	0.072914	-
Coolant temp	0.029560	-0.060906	-0.042986	0.033451	0.072914	1.000000	-
Engine Condition	-0.268201	0.060904	0.116259	-0.024054	-0.093635	-0.046326	

In [7]:
```
# Heatmap Correlation-Matrix
corr_matrix = df.corr()

plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f', an
plt.title('Correlation Motorzustand')
plt.show()
```

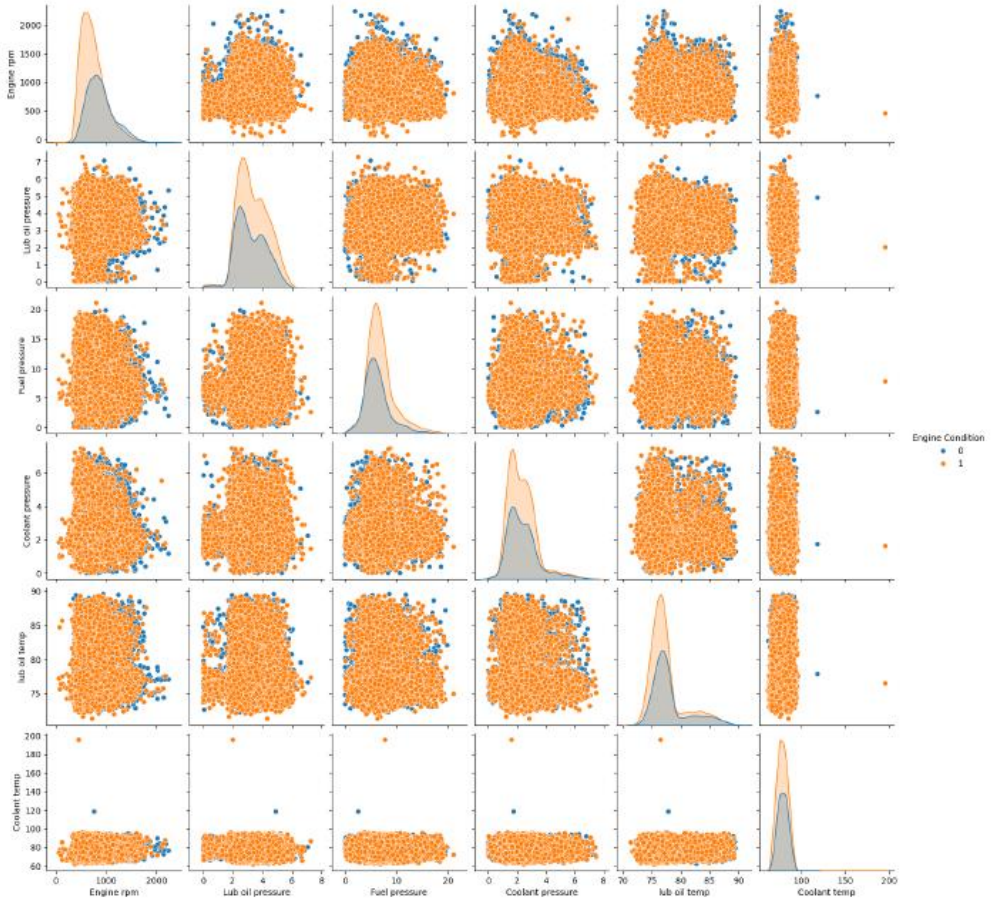


In [8]:
```
#Distribution plot
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

num_cols = len(df.columns)
fig, axes = plt.subplots(nrows=1, ncols=num_cols, figsize=(6*num_co
for i, column in enumerate(df.columns):

  if (df[column].dtypes==int) | (df[column].dtypes==float):
    sns.histplot(df[column], ax=axes[i])
  else:
    sns.countplot(data=df, x=column, ax=axes[i])
  axes[i].set_title(column)
plt.tight_layout()
plt.show()
```



In [9]:
```
#Visualisation von Distribution, anhand 0,1 als Engine Condition
condition_0_data = df[df['Engine Condition'] == 0]
```

```
condition_1_data = df[df['Engine Condition'] == 1]
filtered_data = pd.concat([condition_0_data, condition_1_data])
sns.pairplot(filtered_data, hue='Engine Condition')
plt.show()
```



In [10…

```
#Outlier/IQR for Standardization

def count_outliers(series):
    Q1 = series.quantile(0.25)
    Q3 = series.quantile(0.75)
    IQR = Q3 - Q1
    U = Q3 + 1.5 * IQR
    l = Q1 - 1.5 * IQR
    U = series[series > U].count()
    l = series[series < l].count()
    return U, l


outlier_counts = {}
for column in df.select_dtypes(include=['float64', 'int64']).column
    upper_count, lower_count = count_outliers(df[column])
    outlier_counts[column] = {'Upper Outliers': upper_count, 'Lower

for column, counts in outlier_counts.items():
```

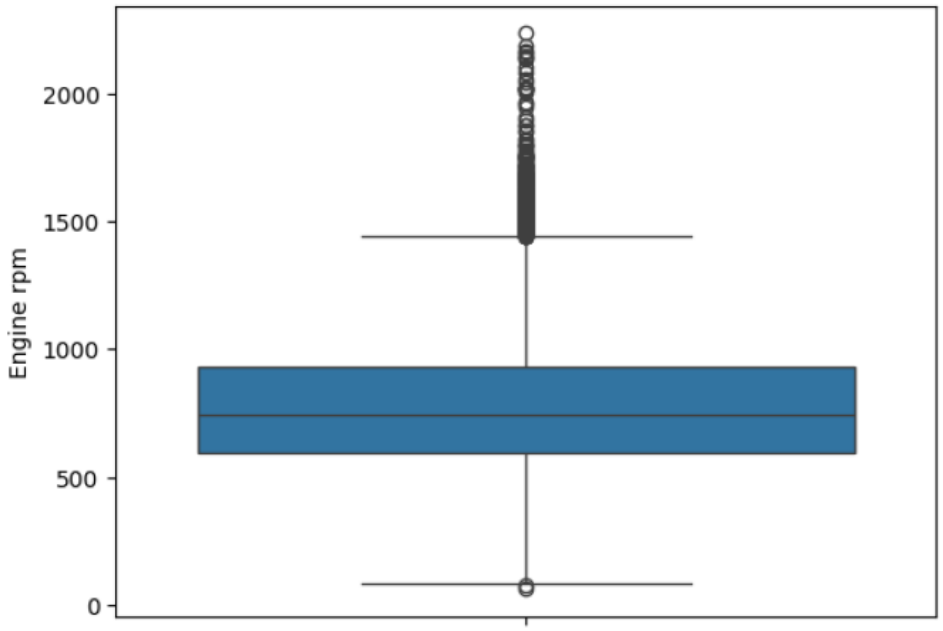
```
    print(f"Column: {column}")
    print(f"Upper Outliers: {counts['Upper Outliers']}")
    print(f"Lower Outliers: {counts['Lower Outliers']}")
    print()
```

```
Column: Engine rpm
Upper Outliers: 462
Lower Outliers: 2

Column: Lub oil pressure
Upper Outliers: 13
Lower Outliers: 53

Column: Fuel pressure
Upper Outliers: 1069
Lower Outliers: 66

Column: Coolant pressure
Upper Outliers: 785
Lower Outliers: 0

Column: lub oil temp
Upper Outliers: 2612
Lower Outliers: 5

Column: Coolant temp
Upper Outliers: 2
Lower Outliers: 0

Column: Engine Condition
Upper Outliers: 0
Lower Outliers: 0
```

In [11…

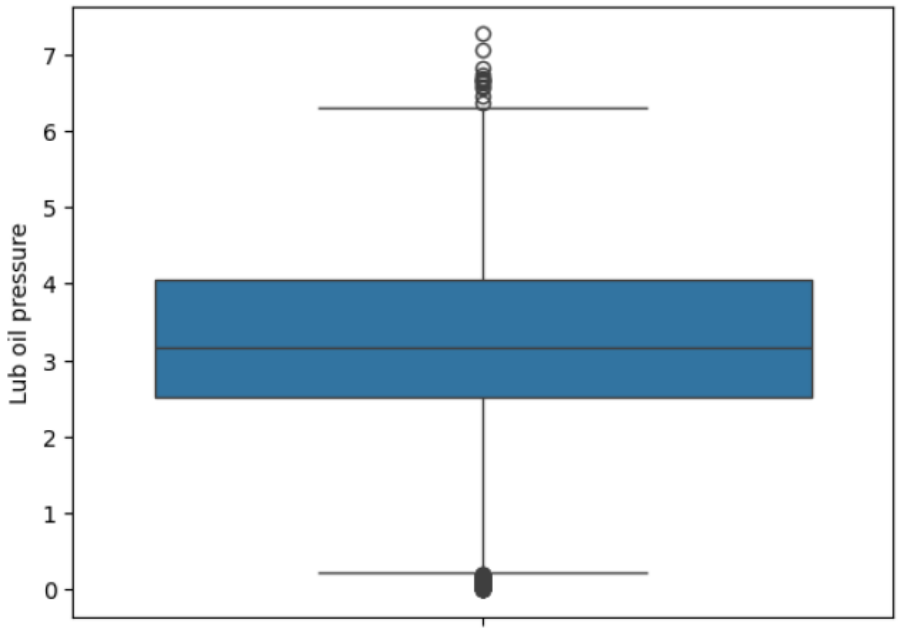
```
#Outlier/IQR for Standardization
# Box Plot Engine rpm
import seaborn as sns
sns.boxplot(df['Engine rpm'])
```

Out[11]: <Axes: ylabel='Engine rpm'>



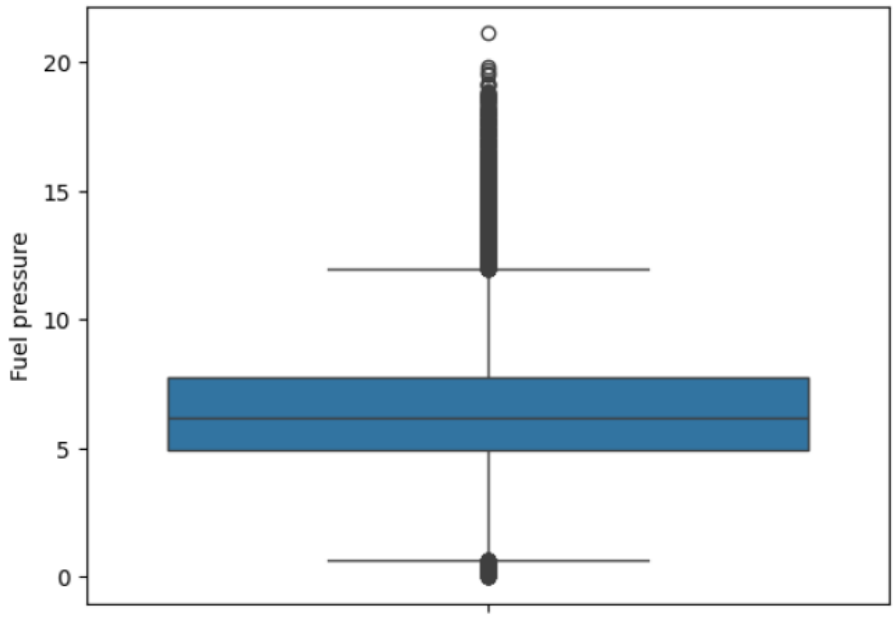
In [12…
```
#Outlier/IQR for Standardization
# Box Plot Lub oil pressure
import seaborn as sns
sns.boxplot(df['Lub oil pressure'])
```

Out[12]: <Axes: ylabel='Lub oil pressure'>



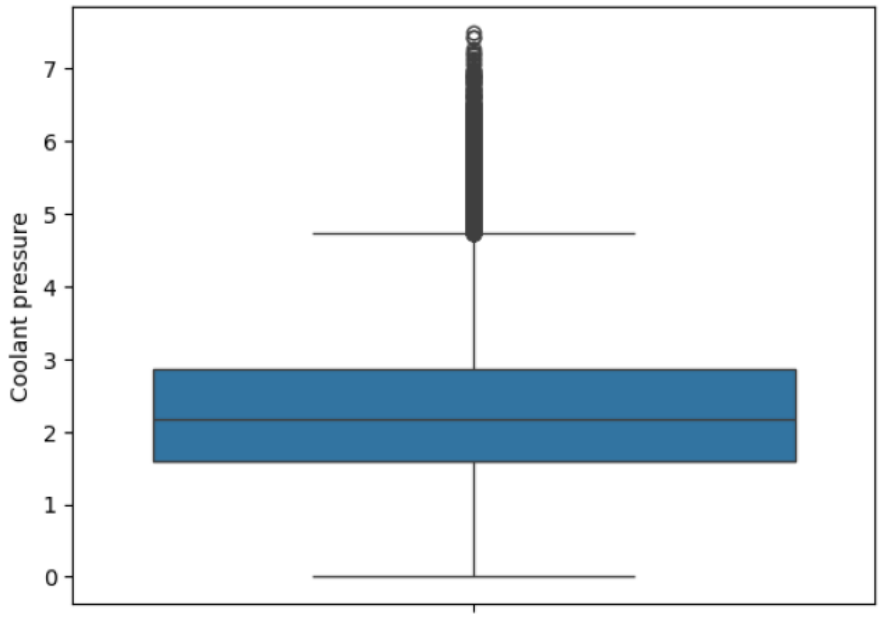
In [13…
```
#Outlier/IQR for Standardization
# Box Plot Fuel pressure
sns.boxplot(df['Fuel pressure'])
```

Out[13]: <Axes: ylabel='Fuel pressure'>



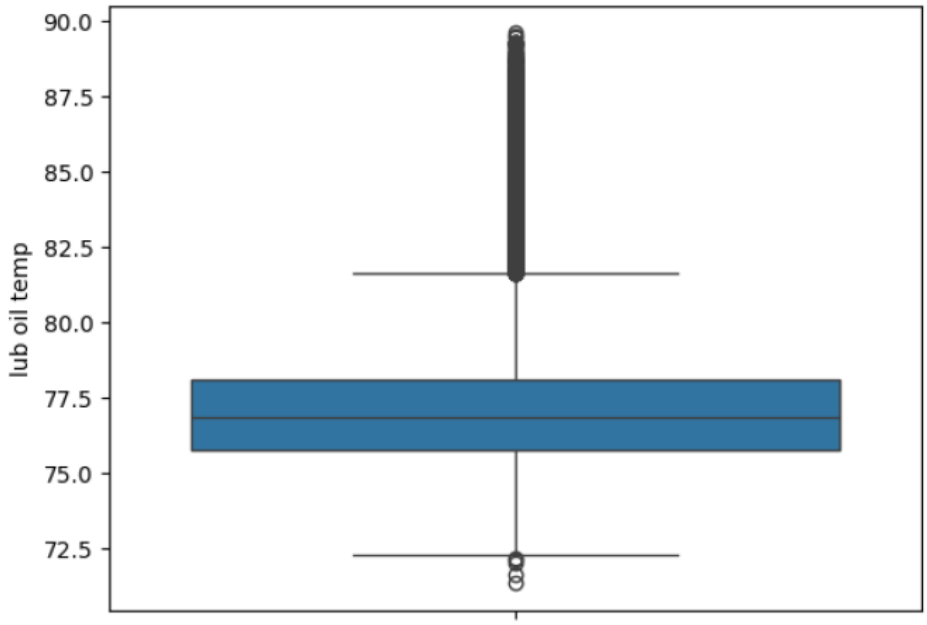
In [14…
```
#Outlier/IQR for Standardization
# Box Plot Fuel pressure
sns.boxplot(df['Coolant pressure'])
```

Out[14]: <Axes: ylabel='Coolant pressure'>



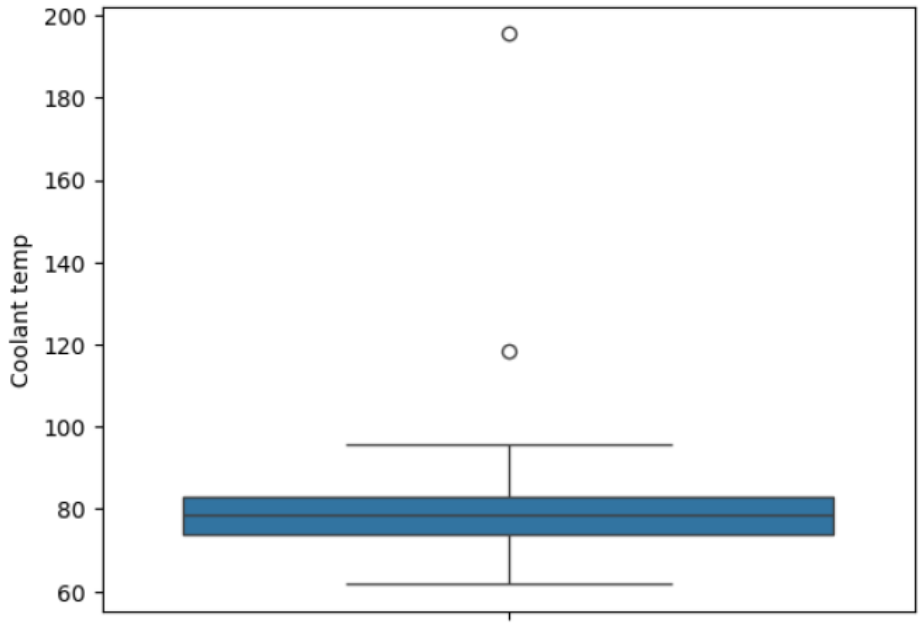
In [15…
```
#Outlier/IQR for Standardization
# Box Plot lub oil temp
import seaborn as sns
sns.boxplot(df['lub oil temp'])
```

Out[15]: <Axes: ylabel='lub oil temp'>



In [16…
```
#Outlier/IQR for Standardization
# Box Plot Coolant temp
import seaborn as sns
sns.boxplot(df['Coolant temp'])
```

Out[16]: <Axes: ylabel='Coolant temp'>



In [17…
```
# Random forest mit Training test CV
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score
import numpy as np

X = df[['Engine rpm', 'Lub oil pressure', 'Fuel pressure', 'Coolant
y = df['Engine Condition']

# Model RandomForestClassifier
model = RandomForestClassifier(n_estimators=200, random_state=52)

cv_scores = cross_val_score(model, X, y, cv=5) #K-fold 5

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6570
```

```
Cross-Validation Scores: [0.65881751 0.65165088 0.65062708 0.656513
95 0.66777579]
Mean CV Accuracy: 0.657077041208088
```

In [18…

```
# LogesticRegration mit Tarining Test CV

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
import numpy as np

X = df[['Engine rpm', 'Lub oil pressure', 'Fuel pressure', 'Coolant
y = df['Engine Condition']


model = LogisticRegression(random_state=52, max_iter=3000)

cv_scores = cross_val_score(model, X, y, cv=5) #K-fold 5


print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores))  #6591
```

```
Cross-Validation Scores: [0.65318659 0.65932941 0.66726389 0.655490
15 0.66060916]
Mean CV Accuracy: 0.6591758382390581
```

In [19…

```
# SuppertVektorMaschin mit Tarining Test CV

from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score
import numpy as np

X = df[['Engine rpm', 'Lub oil pressure', 'Fuel pressure', 'Coolant
y = df['Engine Condition']


model = SVC(random_state=52)
```

```
cv_scores = cross_val_score(model, X, y, cv=5)  #K-fold 5


print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6443
```

```
Cross-Validation Scores: [0.63987714 0.64832352 0.65088303 0.640389
05 0.6421807 ]
Mean CV Accuracy: 0.6443306885078066
```

In [20…

```
# Naive Bayes-BernoulliNB mit Tarining Test CV

from sklearn.naive_bayes import BernoulliNB
from sklearn.model_selection import cross_val_score
import numpy as np

X = df[['Engine rpm', 'Lub oil pressure', 'Fuel pressure', 'Coolant
y = df['Engine Condition']


model = BernoulliNB()


cv_scores = cross_val_score(model, X, y, cv=5)  #K-fold 5

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6305
```

```
Cross-Validation Scores: [0.63066291 0.63066291 0.63040696 0.630406
96 0.63040696]
Mean CV Accuracy: 0.6305093422062964
```

In [21…

```
# Die obige Ergebnisse sind ohne Outlier Handeling, Weiter hin wird
```

In [21…

```
# Outlier Handling- Remove

def remove_upper_outliers(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    upper_bound = Q3 + 1.5 * IQR
    return column[column <= upper_bound]

df['Coolant temp'] = remove_upper_outliers(df['Coolant temp']) # 2


def remove_lower_outliers(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    return column[column >= lower_bound]
```

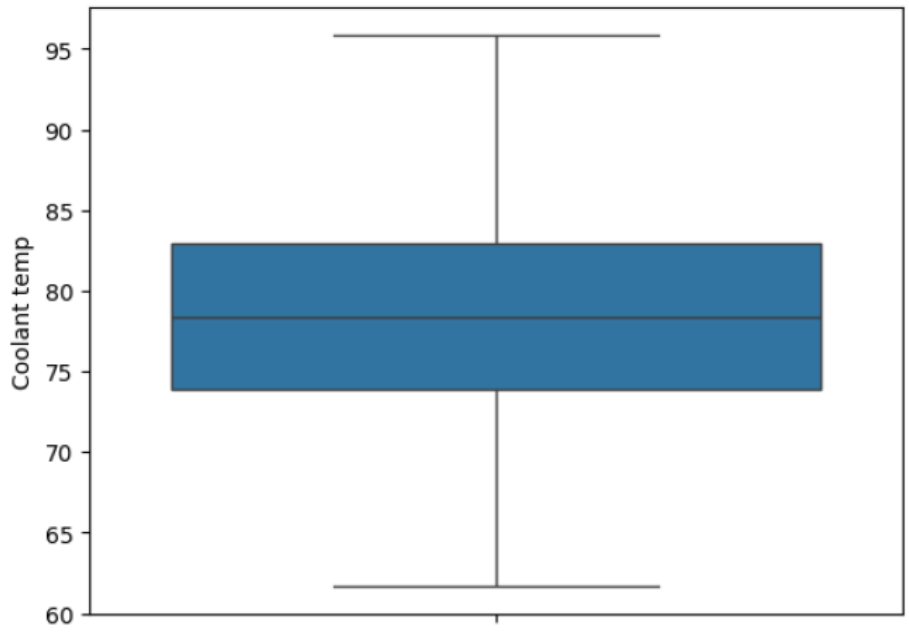
```
df['lub oil temp'] = remove_lower_outliers(df['lub oil temp']) # 5
df['Engine rpm'] = remove_lower_outliers(df['Engine rpm']) # 2 Date
```

In [22…
```
#Shape of Dataframe

df_cleaned = df.dropna()


print("Ursprüngliche Form der Daten:", df.shape)
print("Form der bereinigten Daten:", df_cleaned.shape)
```

```
Ursprüngliche Form der Daten: (19535, 7)
Form der bereinigten Daten: (19526, 7)
```

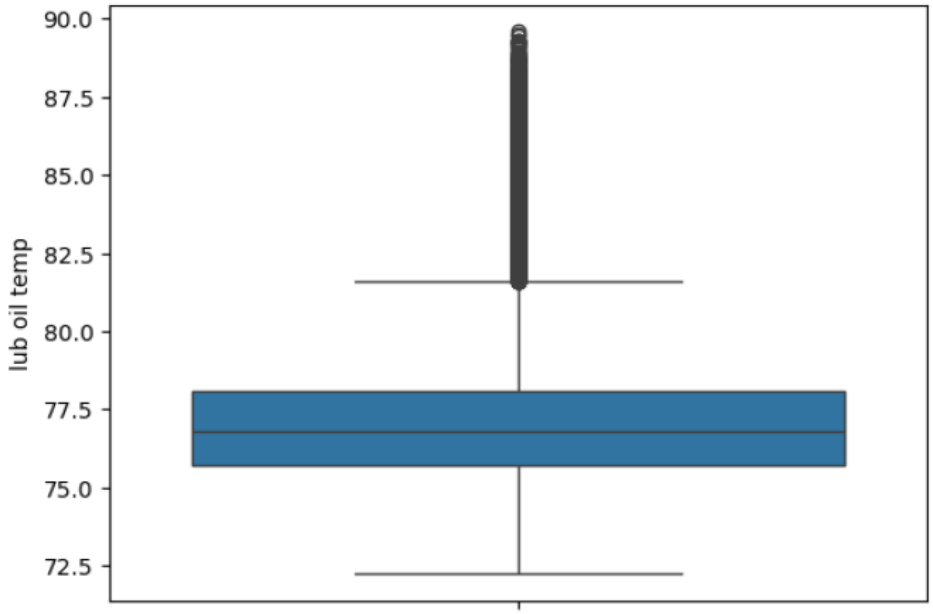
In [23…
```
#Outlier/IQR for Standardization
# Box Plot Coolant temp nach Data Cleaning
import seaborn as sns
sns.boxplot(df['Coolant temp'])
```

Out[23]: <Axes: ylabel='Coolant temp'>



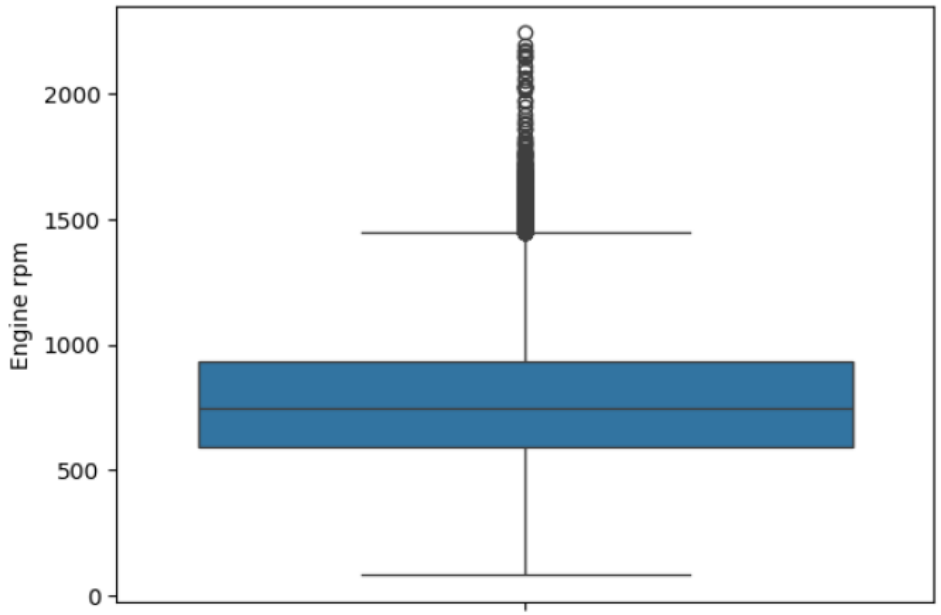
In [24…
```
#Outlier/IQR for Standardization
# Box Plot lub oil temp nach Data Cleaning
import seaborn as sns
sns.boxplot(df['lub oil temp'])
```

Out[24]: <Axes: ylabel='lub oil temp'>



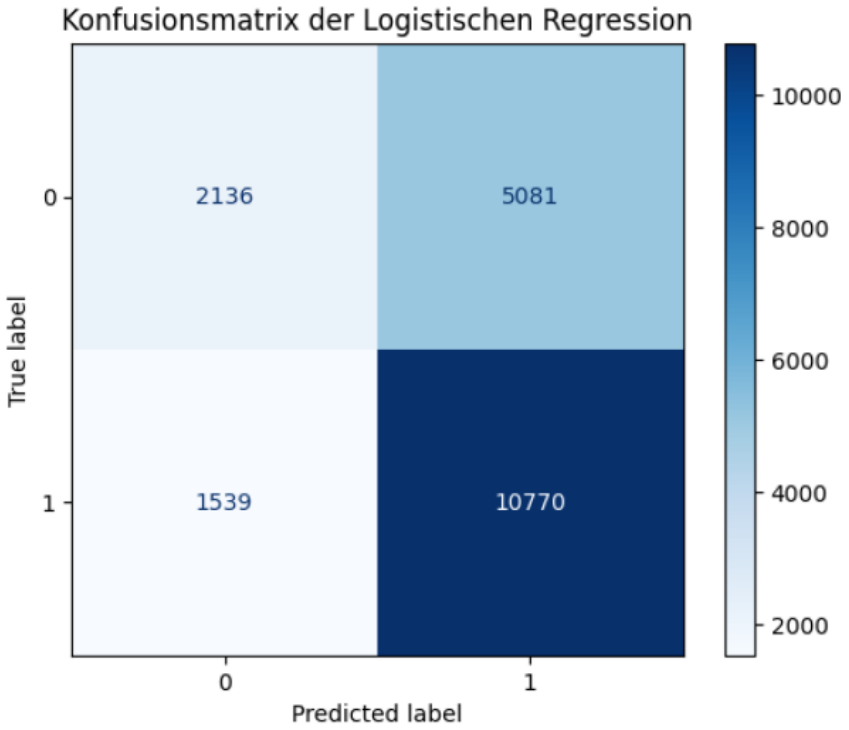
In [25…
```
#Outlier/IQR for Standardization
# Box Plot Engine rpm nach Data Cleaning
import seaborn as sns
sns.boxplot(df['Engine rpm'])
```

Out[25]: <Axes: ylabel='Engine rpm'>



In [26…
```
# LogesticRegration mit Tarining Test CV nach Outlier Handling

from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score, cross_val_pred
```

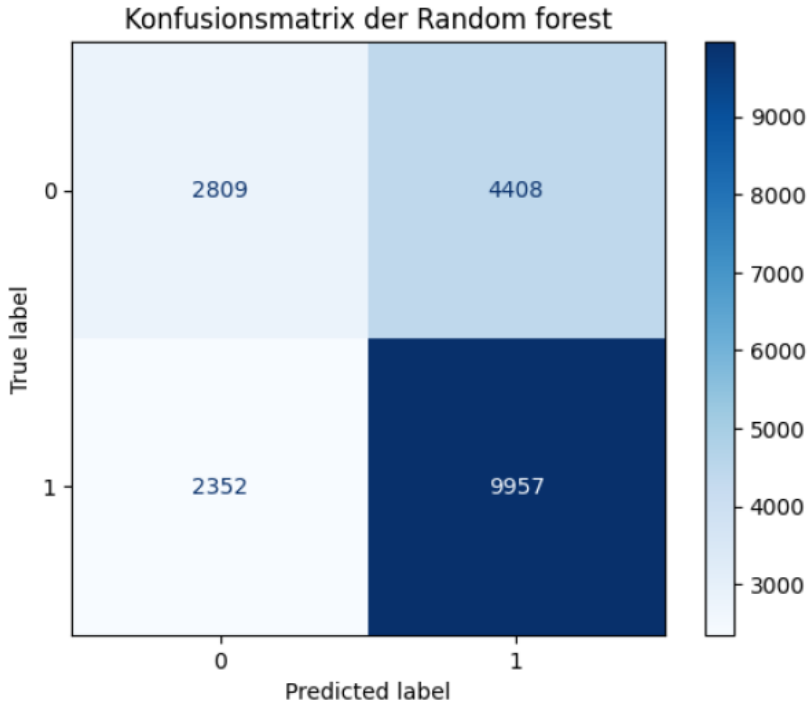
```python
from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla


X = df_cleaned[['Engine rpm', 'Lub oil pressure', 'Fuel pressure',
y = df_cleaned['Engine Condition']


model = LogisticRegression(random_state=52, max_iter=3000)

cv_scores = cross_val_score(model, X, y, cv=5) #K-fold 5

y_prediction_cv = cross_val_predict(model, X, y, cv=5)

cm = confusion_matrix(y, y_prediction_cv)


model.fit(X, y)


disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=m
disp.plot(cmap=plt.cm.Blues)
plt.title('Konfusionsmatrix der Logistischen Regression')
plt.show()

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores))  #6591-0.6609-0.6580
```



```
Cross-Validation Scores: [0.65514593 0.66734955 0.66760563 0.652240
72 0.66248399]
Mean CV Accuracy: 0.6609651653813399
```

In [27…

```python
# Random forest mit Training test CV nach Outlier Handling

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score,cross_val_predi

from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla


X = df_cleaned[['Engine rpm', 'Lub oil pressure', 'Fuel pressure',
y = df_cleaned['Engine Condition']


model = RandomForestClassifier(n_estimators=200, random_state=52)

cv_scores = cross_val_score(model, X, y, cv=5) #K-fold 5

y_prediction_cv = cross_val_predict(model, X, y, cv=5)

cm = confusion_matrix(y, y_prediction_cv)


model.fit(X, y)


disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=m
disp.plot(cmap=plt.cm.Blues)
plt.title('Konfusionsmatrix der Random forest')
plt.show()
print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6570-0.6537-0.654
```

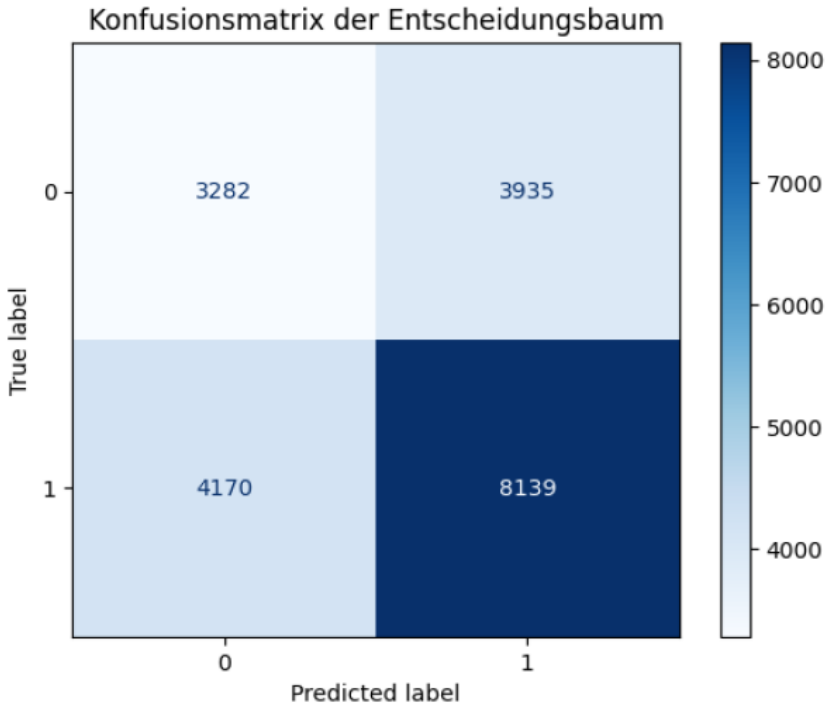



```
Cross-Validation Scores: [0.65232975 0.65096031 0.64814341 0.652752
88 0.66478873]
Mean CV Accuracy: 0.6537950151216847
```

In [28…

```python
# Entscheidungsbaum mit Training Test CV nach Outlier Handling

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score,cross_val_predi

from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla


X = df_cleaned[['Engine rpm', 'Lub oil pressure', 'Fuel pressure',
y = df_cleaned['Engine Condition']


model = DecisionTreeClassifier(random_state=52)


cv_scores = cross_val_score(model, X, y, cv=5) #K-fold 5

y_prediction_cv = cross_val_predict(model, X, y, cv=5)

cm = confusion_matrix(y, y_prediction_cv)


model.fit(X, y)
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=m
disp.plot(cmap=plt.cm.Blues)
plt.title('Konfusionsmatrix der Entscheidungsbaum')
plt.show()
print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores))  #0.5857-0.5849-0.58
```

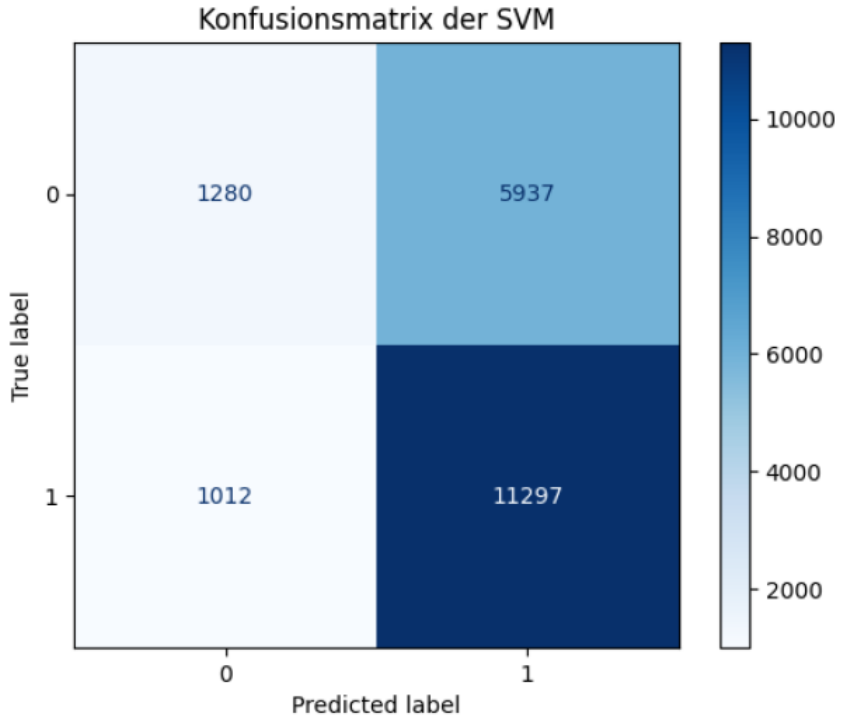



```
Cross-Validation Scores: [0.58806964 0.59052497 0.58104994 0.588732
39 0.57618438]
Mean CV Accuracy: 0.5849122627586963
```

In [29…

```
# SVM mit Training Test CV nach Outlier Handling

from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score,cross_val_predi

from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla

X = df_cleaned[['Engine rpm', 'Lub oil pressure', 'Fuel pressure',
y = df_cleaned['Engine Condition']


model = SVC(random_state=52)


cv_scores = cross_val_score(model, X, y, cv=5)  #K-fold 5

y_prediction_cv = cross_val_predict(model, X, y, cv=5)

cm = confusion_matrix(y, y_prediction_cv)
```

```
model.fit(X, y)


disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=m
disp.plot(cmap=plt.cm.Blues)
plt.title('Konfusionsmatrix der SVM')
plt.show()

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6443-0.6441-0.644
```

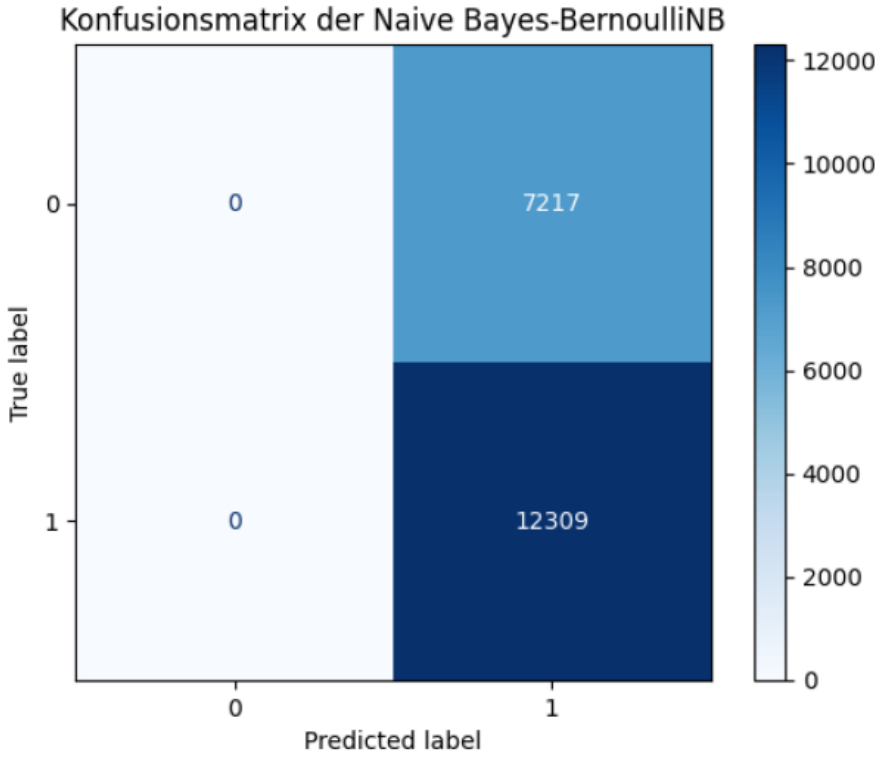



```
Cross-Validation Scores: [0.63952893 0.64763124 0.65121639 0.640204
87 0.64199744]
Mean CV Accuracy: 0.6441157731662048
```

In [30…

```
# Naive Bayes-BernoulliNB mit Tarining Test CV nach Outlier Handlin

from sklearn.naive_bayes import BernoulliNB
from sklearn.model_selection import cross_val_score

from sklearn.metrics import confusion_matrix, ConfusionMatrixDispla

X = df_cleaned[['Engine rpm', 'Lub oil pressure', 'Fuel pressure',
y = df_cleaned['Engine Condition']


model = BernoulliNB()
```

```
cv_scores = cross_val_score(model, X, y, cv=5)  #K-fold 5

y_prediction_cv = cross_val_predict(model, X, y, cv=5)

cm = confusion_matrix(y, y_prediction_cv)


model.fit(X, y)


disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=m
disp.plot(cmap=plt.cm.Blues)
plt.title('Konfusionsmatrix der Naive Bayes-BernoulliNB')
plt.show()

print("Cross-Validation Scores:", cv_scores)
print("Mean CV Accuracy:", np.mean(cv_scores)) #0.6305-0.6303
```




```
Cross-Validation Scores: [0.63031234 0.63047375 0.63047375 0.630473
75 0.63021767]
Mean CV Accuracy: 0.630390252889117
```

5 References

[1] Acito, F. 2023. Introduction to Analytics. In *Predictive Analytics with KNIME*. *Analytics for Citizen Data Scientists*, F. Acito, Ed. Springer, Cham, 1–9. DOI=10.1007/978-3-031-45630-5_1.

[2] Alogogianni, E. and Virvou, M. uuuu-uuuu. Data Mining for Targeted Inspections Against Undeclared Work by Applying the CRISP-DM Methodology. In *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE, 1–8. DOI=10.1109/IISA52424.2021.9555543.

[3] Andrae, S. 2023. Grundlagen des maschinellen Lernens. In *Ökonometrie und maschinelles Lernen*. *Basiswissen für Ökonomen*, S. Andrae, Ed. essentials. Springer Gabler, Wiesbaden, Germany, 5–25. DOI=10.1007/978-3-658-41362-0_2.

[4] Ayyadevara, V. K. 2018. Random Forest. In *Pro machine learning algorithms*. *A hands-on approach to implementing algorithms in Python and R*, V. K. Ayyadevara, Ed. Apress, [Berkeley], 105–116. DOI=10.1007/978-1-4842-3564-5_5.

[5] Bacher, J., Pöge, A., and Wenzig, K. 2011. *Clusteranalyse*. *Anwendungsorientierte Einführung in Klassifikationsverfahren*. Oldenbourg, München.

[6] Biemann, C., Heyer, G., and Quasthoff, U. 2022. Maschinelles Lernen für Sprachverarbeitung. In *Wissensrohstoff text - konzepte, algorithmen, ergebnisse*. *Eine einfhrung in das text mining*, C. Biemann, Ed. Springer, Germany, 257–309. DOI=10.1007/978-3-658-35969-0_6.

[7] Bink, R. and Zschech, P. 2018. Predictive Maintenance in der industriellen Praxis. *HMD* 55, 3, 552–565.

[8] Borth, D., Hüllermeier, E., and Kauermann, G. 2023. Maschinelles Lernen. In *KUNSTLICHE INTELLIGENZ UND DATA SCIENCE IN THEORIE UND PRAXIS*. *Von*, A. Gillhuber, G. Kauermann and W. Hauner, Eds. SPRINGER SPEKTRUM, [S.l.], 19–49. DOI=10.1007/978-3-662-66278-6_4.

[9] Botsch, B., Ed. 2023. *MASCHINELLES LERNEN - GRUNDLAGEN UND ANWENDUNGEN*. *Mit beispielen in python*. SPRINGER SPEKTRUM, [S.l.].

[10] Chauhan, V. K., Dahiya, K., and Sharma, A. 2019. Problem formulations and solvers in linear SVM: a review. *Artif Intell Rev* 52, 2, 803–855.

[11] Colombo, A. W., Karnouskos, S., Kaynak, O., Shi, Y., and Yin, S. 2017. Industrial Cyberphysical Systems: A Backbone of the Fourth Industrial Revolution. *EEE Ind. Electron. Mag.* 11, 1, 6–16.

[12] Dahm, M. H. and Zehnder, V. 2023. Grundlagen der KI. In *Moderne Personalführung mit Künstlicher Intelligenz*, M. H. Dahm and V. Zehnder, Eds. essentials. Springer Nature, Wiesbaden, 3–16. DOI=10.1007/978-3-658-43138-9_2.

[13] Demirbaga, Ü., Aujla, G. S., Jindal, A., and Kalyon, O. 2024. Introduction. In *Big data analytics*. *Theory, techniques, platforms, and applications*, Ü. Demirbaga, G. S. Aujla, A. Jindal and O. Kalyon, Eds. Springer, Cham, 1–8. DOI=10.1007/978-3-031-55639-5_1.

[14] DIN 31051:2012-09, S.4.

[15] Dindorf, C., Bartaguiz, E., Gassmann, F., and Fröhlich, M. 2023. Eine kurze Einführung in die Künstliche Intelligenz. In *KUNSTLICHE INTELLIGENZ IN SPORT UND SPORTWISSENSCHAFT*. *Potentiale*, C. Dindorf, E. Bartaguiz, F. Gassmann and M. Fröhlich, Eds. essentials. SPRINGER SPEKTRUM, [S.l.], 7–22. DOI=10.1007/978-3-662-67419-2_2.

[16] Dreiseitl, S. and Ohno-Machado, L. 2002. Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics* 35, 5-6, 352–359.

[17] Eguturi Manjith Kumar Reddy, Akash Gurrala, Vasireddy Bindu Hasitha, and Korupalli V Rajesh Kumar. 2022. Introduction to Naive Bayes and a Review on Its Subtypes with Applications. In *Bayesian reasoning and Gaussian processes for machine learning applications*, H. K., S. Tayal, P. M. George, P. Singla and U. Kose, Eds. CRC Press, Taylor & Francis Group, Boca Raton, 1–14. DOI=10.1201/9781003164265-1.

[18] 2008. Entscheidungsbäume. In *Datenanalyse und Statistik*. *Eine Einführung für Ökonomen im Bachelor*, U. Bankhofer and J. Vogel, Eds. Lehrbuch. Gabler, Wiesbaden, 273–284.

[19] Ferreira, J. E. V., Pinheiro, M. T. S., dos Santos, W. R. S., and Da Maia, R. S. 2016. Graphical representation of chemical periodicity of main elements through boxplot. *Educación Química* 27, 3, 209–216.

[20] Fleisch, E., Weinberger, M., and Wortmann, F. 2014. Geschäftsmodelle im Internet der Dinge. *HMD* 51, 6, 812–826.

[21] Haim, M. 2023. Maschinelles Lernen ohne Goldstandard („unüberwachtes Lernen“). In *Computational Communication Science*. *Eine Einführung*, M. Haim, Ed. Studienbücher zur Kommunikations- und Medienwissenschaft. Springer Fachmedien Wiesbaden GmbH; Springer VS, Wiesbaden, 257–277. DOI=10.1007/978-3-658-40171-9_11.

[22] Huang, J.-C., Ko, K.-M., Shu, M.-H., and Hsu, B.-M. 2020. Application and comparison of several machine learning algorithms and their integration models in regression problems. *Neural Comput & Applic* 32, 10, 5461–5469.

[23] Hüning, F. 2019. *Embedded Systems für IoT*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[24] *IoT & Industrie 4.0*. https://www.btelligent.com/themen/industrie-40/. Accessed 25 January 2024.

[25] Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., and Li, B. 2018. Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In *2018 IEEE Symposium on Security and Privacy*. *SP 2018 : proceedings : 21-23 May 2018, San Francisco, California, USA*. IEEE Computer Society, Los Alamitos, CA, 19–35. DOI=10.1109/SP.2018.00057.

[26] Jing Peng, A. F. 2016. *In God We Trust. All Others Must Bring Data.-W. Edwards Deming. Using Word Embeddings to Recognize Idioms*.

[27] John McCarthy. 2004. WHAT IS ARTIFICIAL INTELLIGENCE? (Nov. 2004), 2.

[28] Kaplan, J. 2017. *Künstliche Intelligenz: Eine Einführung*. mitp Professional. MITP-Verlags GmbH & Co. KG, 2017.

[29] Knuth, T. 2021. Lernende Entscheidungsbäume. *Informatik Spektrum* 44, 5, 364–369.

[30] König, U. M., Röglinger, M., and Urbach, N. 2021. Industrie 4.0 in kleinen und mittleren Unternehmen – Lösungsansatz und Handlungsempfehlungen für die Integration smarter Geräte. In *IOT BEST PRACTICES*. *Internet der dinge, geschftsmodellinnovationen, iot*, S. Meinhardt and F. Wortmann, Eds. Edition HMD. MORGAN KAUFMANN, [S.l.], 141–157. DOI=10.1007/978-3-658-32439-1_8.

[31] Krauss, P. 2023. Was ist Künstliche Intelligenz? In *Künstliche Intelligenz und Hirnforschung*. Springer, Berlin, Heidelberg, 117–123. DOI=10.1007/978-3-662-67179-5_11.

[32] Kreutzer. 2019. *Künstliche Intelligenz verstehen*. Springer Fachmedien Wiesbaden, Wiesbaden.

[33] Kusiak, A. 2018. Smart manufacturing. *International Journal of Production Research* 56, 1-2, 508–517.

[34] Kusiak, A. 2018. Smart manufacturing. *International Journal of Production Research* 56, 1-2, 508–517.

[35] Kwak, S. K. and Kim, J. H. 2017. Statistical data preparation: management of missing values and outliers. *Korean journal of anesthesiology* 70, 4, 407–411.

[36] Lanquillon, C. 2020. Grundzüge des maschinellen Lernens. In *Blockchain und Maschinelles Lernen*. *Wie das Maschinelle Lernen und Die Distributed-Ledger-Technologie Voneinander Profitieren*, S. Schacht and C. Lanquillon, Eds. Springer Vieweg. in Springer Fachmedien Wiesbaden GmbH, Berlin, Heidelberg, 89–142. DOI=10.1007/978-3-662-60408-3_3.

[37] LeCun, Y., Bengio, Y., and Hinton, G. 2015. Deep learning. *Nature* 521, 7553, 436–444.

[38] Lee, J., Bagheri, B., and Kao, H.-A. 2015. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. *Manufacturing Letters* 3, 18–23.

[39] Lorenz, U. 2024. Verstärkendes Lernen als Teilgebiet des Maschinellen Lernens. In *REINFORCEMENT LEARNING*. *Aktuelle ansatze verstehen- mit beispielen in java und greenfoot*, U. W. LORENZ, Ed. Springer Vieweg, [S.l.], 1–12. DOI=10.1007/978-3-662-68311-8_1.

[40] Lucke, D., Defranceski, M., and Adolf, T. 2017. Cyberphysische Systeme für die prädiktive Instandhaltung. In *Handbuch Industrie 4.0 Bd.1*. Springer Vieweg, Berlin, Heidelberg, 75–91. DOI=10.1007/978-3-662-45279-0_28.

[41] Lucke, D., Defranceski, M., and Adolf, T. 2023. Cyberphysische Systeme für die prädiktive Instandhaltung. In *Handbuch Industrie 4.0*. *Band 1: Produktion*, T. Bauernhansl, Ed. Springer Berlin Heidelberg; Springer Vieweg, Berlin, Heidelberg, 27–43. DOI=10.1007/978-3-662-58532-0_28.

[42] Madakam, S., Ramaswamy, R., and Tripathi, S. 2015. Internet of Things (IoT): A Literature Review. *JCC* 03, 05, 164–173.

[43] Matthies, B. 2020. Performancemaße von Business Analytics Methoden. *CON* 32, 4, 79–80.

[44] Matzka, S. 2021. *Künstliche Intelligenz in den Ingenieurwissenschaften*. *Maschinelles Lernen verstehen und bewerten*. Lehrbuch. Springer Vieweg, Wiesbaden, Heidelberg.

[45] Matzka, S. 2021. Überwachtes Lernen. In *Künstliche Intelligenz in den Ingenieurwissenschaften*, S. Matzka, Ed. Springer Fachmedien Wiesbaden, Wiesbaden, 99–169. DOI=10.1007/978-3-658-34641-6_4.

[46] Mockenhaupt, A. 2024. Grundlagen der Künstlichen Intelligenz (KI). In *Digitalisierung und Künstliche Intelligenz in der Produktion*. *Grundlagen und Anwendung*, A. Mockenhaupt and T. Schlagenhauf, Eds. Springer Fachmedien Wiesbaden, Weisbaden, 53–104. DOI=10.1007/978-3-658-41935-6_3.

[47] Modi, P. 2023. *Automotive Vehicles Engine Health Dataset*. https://www.kaggle.com/datasets/parvmodi/automotive-vehicles-engine-health-dataset?select=engine_data.csv. Accessed 16 July 2024.

[48] Möller, D. 2023. Der Einsatz Künstlicher Intelligenz bei der Suche nach kinderpornographischen Dateien. In *Sexuelle Gewalt gegen Kinder*, F. Lüttig and J. Lehmann, Eds. Schriften der Generalstaatsanwaltschaft Celle 8. Nomos Verlagsgesellschaft mbH & Co. KG, Baden-Baden, 35–54. DOI=10.5771/9783748919940-35.

[49] Möller, D. P. F. 2016. Digital Manufacturing/Industry 4.0. In *Guide to computing fundamentals in cyber-physical systems*. *Concepts, design methods, and applications*, D. Möller, Ed. Computer Communications and Networks. Springer, Switzerland, 307–375. DOI=10.1007/978-3-319-25178-3_7.

[50] Mühlnickel, H., Kurz, C. M., Jussen, P., and Emonts-Holley, R. 2018. Smart Maintenance. In *Betriebliche Instandhaltung*, Reichel, Ed. VDI-Buch. Springer Berlin Heidelberg, Berlin, Heidelberg, 349–360. DOI=10.1007/978-3-662-53135-8_24.

[51] Nacer, M. I., Prakoonwit, S., and Alarab, I. uuuu-uuuu. Blockchain as a Complementary Technology for the Internet of Things: A Survey. In *Internet of Things : Cases and Studies*, F. P. García Márquez, Ed. International Series in Operations Research & Management Science. Springer, Cham, 1–24. DOI=10.1007/978-3-030-70478-0_1.

[52] Neuer, M. J. 2024. Überwachtes Lernen. In *Maschinelles Lernen für die Ingenieurwissenschaften*, M. J. Neuer, Ed. Springer Nature, Berlin, Heidelberg, 93–149. DOI=10.1007/978-3-662-68216-6_4.

[53] Neuer, M. J. 2024. Unüberwachtes Lernen. In *Maschinelles Lernen für die Ingenieurwissenschaften*, M. J. Neuer, Ed. Springer Nature, Berlin, Heidelberg, 151–183. DOI=10.1007/978-3-662-68216-6_5.

[54] Neuhöfer, S. 2023. *Grundrechtsfähigkeit Künstlicher Intelligenz*. Internetrecht und Digitale Gesellschaft 42. Duncker & Humblot, Berlin.

[55] Nti, I. K., Nyarko-Boateng, O., and Aning, J. 2021. Performance of Machine Learning Algorithms with Different K Values in K-fold CrossValidation. *IJITCS* 13, 6, 61–71.

[56] Oks, S. J., Jalowski, M., Lechner, M., Mirschberger, S., Merklein, M., Vogel-Heuser, B., and Möslein, K. M. 2022. Cyber-Physical Systems in the Context of Industry 4.0: A Review, Categorization and Outlook. *Inf Syst Front*, 1–42.

[57] Osterhage, W. 2023. Internet of Things. *Vom Ding an sich zum Internet der Dinge*, 109–116.

[58] Peng, J., & Feldman, A. 2016. *In God We Trust. All Others Must Bring Data.-W. Edwards Deming. Using Word Embeddings to Recognize Idioms*.

[59] R, A. 2023. Kaggle: Alles, was Du über diese Plattform wissen musst. *DataScientest* (Sep. 2023).

[60] Rai, R. and Sahu, C. K. 2020. Driven by Data or Derived Through Physics? A Review of Hybrid Physics Guided Machine Learning Techniques With Cyber-Physical System (CPS) Focus. *IEEE Access* 8, 71050–71073.

[61] Rigatti, S. J. 2017. Random Forest. *Journal of insurance medicine (New York, N.Y.)* 47, 1, 31–39.

[62] Schawel, C. and Billing, F. 2018. Entscheidungsbaum. In *Top 100 Management Tools*. *Das wichtigste Buch eines Managers von ABC-Analyse bis Zielvereinbarung*, C. Schawel and F. Billing, Eds. Springer Gabler, Wiesbaden, 121–124. DOI=10.1007/978-3-658-18917-4_31.

[63] Scherer, A. 1997. Überwachtes Lernen. In *Neuronale Netze*. *Grundlagen und Anwendungen*, A. Scherer, Ed. Computational intelligence. Vieweg, Braunschweig, Wiesbaden, 71–92. DOI=10.1007/978-3-322-86830-5_6.

[64] Schneider, L. C. 2023. Einführung in Machine Learning. In *Das MAI-Tool als Untersuchungsinstrument von Lösungsprozessen beim mathematischen Modellieren*, L. C. Schneider, Ed. Mathematikdidaktik im Fokus. SPRINGER SPEKTRUM, Wiesbaden, Germany, 175–179. DOI=10.1007/978-3-658-41732-1_11.

[65] Schütze, A. and Helwig, N. 2017. Sensorik und Messtechnik für die Industrie 4.0. *tm - Technisches Messen* 84, 5, 310–319.

[66] Schwertman, N. C., Owens, M. A., and Adnan, R. 2004. A simple more general boxplot method for identifying outliers. *Computational Statistics & Data Analysis* 47, 1, 165–174.

[67] Shi, Y. 2022. Big Data and Big Data Analytics. In *Advances in Big Data Analytics*. *Theory, Algorithms and Practices*, Y. Shi, Ed. Springer Nature, Singapore, 3–21. DOI=10.1007/978-981-16-3607-3_1.

[68] Singh, S. and Singh, N. 2012. Big Data analytics. In *2012 International Conference on Communication, Information & Computing Technology (ICCICT)*. IEEE / Institute of Electrical and Electronics Engineers Incorporated, 1–4. DOI=10.1109/ICCICT.2012.6398180.

[69] Sinsel, A. 2020. Smart Manufacturing. In *Das Internet der Dinge in der Produktion*. *Smart Manufacturing für Anwender und Lösungsanbieter*, A. Sinsel, Ed. Springer Vieweg, Berlin, 1–35. DOI=10.1007/978-3-662-59761-3_1.

[70] Strunz, M. 2012. *Instandhaltung*. *Grundlagen - Strategien - Werkstätten*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[71] Taulli, T. 2023. Maschinelles Lernen. In *Grundlagen der Kunstlichen Intelligenz*. *Eine nichttechnische Einfuhrung*, T. Taulli, Ed. Springer-Verlag Berlin and Heidelberg GmbH & Co. KG, Berlin, 45–77. DOI=10.1007/978-3-662-66283-0_3.

[72] Thériault, R., Ben-Shachar, M. S., Patil, I., Lüdecke, D., Wiernik, B. M., and Makowski, D. 2024. Check your outliers! An introduction to identifying statistical outliers in R with easystats. *Behavior research methods* 56, 4, 4162–4172.

[73] Tsai, C.-W., Lai, C.-F., Chao, H.-C., and Vasilakos, A. V. 2015. Big data analytics: a survey. *Journal of Big Data* 2, 1, 1–32.

[74] Vasudevan, R. K., Ziatdinov, M., Vlcek, L., and Kalinin, S. V. 2021. Off-the-shelf deep learning is not enough, and requires parsimony, Bayesianity, and causality. *npj Comput Mater* 7, 1.

[75] Vishwanathan, S. and Narasimha Murty, M. 2002. SSVM: a simple SVM algorithm. In *Proceedings of the 2002 International Joint Conference on Neural Networks*. IEEE Service Center, Piscataway, NJ, 2393–2398. DOI=10.1109/IJCNN.2002.1007516.

[76] Zhu, T., Ran, Y., Zhou, X., and Wen, Y. 2019. *A Survey of Predictive Maintenance: Systems, Purposes and Approaches*.

[77] Zou, X., Hu, Y., Tian, Z., and Shen, K. uuuu-uuuu. Logistic Regression Model Optimization and Case Analysis. In *2019 IEEE 7th International Conference on Computer Science and*

Network Technology (ICCSNT). IEEE, 135–139.
DOI=10.1109/ICCSNT47585.2019.8962457.